# Gym Analytics and Personal Training Application

## Technical Manual


Author: Jamie Hawthorne

Supervisor: Greg Doyle

Submission Date: 30/04/2021

# Abstract

The purpose of this “Gym Analytics and Personal Training application” is to develop a mobile application for Android and iOS with two sections available to the public. Firstly, we have a client section to help the average person hire a personal trainer, purchase a workout plan whether it is a pre-made plan or a requested plan. The client also has the option to set motivational personal goals/ milestones for them to achieve.

Secondly, we have the trainer side of this application, it is here where a Personal Trainer can grow their business by posting pre-made plans, create custom plans for clients, reach a wide range of potential clients and receive payments directly into their PayPal accounts.

# Contents

# Introduction

The purpose of this Technical Manual document is to outline the requirements, installation procedure and show all relevant code for my application. Due to time restraints and the lack of access to an Apple MAC for development and testing, the currently working version of the application is for Android only. The installable APK file and all associated code for my "Gym Analytics and Personal Training Application" is currently available on the GitHub linked below.

https://github.com/Jamie-Source/GymApp

As the trainer receives money from the applications business account and this application is currently using PayPal Sandbox the only trainer registered who can receive money is:

Email: testemailjamie@gmail.com.

Password: Password12

To install the current version of this application, simply download the APK file onto your android device and run it through the local installer.

All code associated with this application is displayed below and is in the same format as it can be found in Microsoft Visual Studio to give the reader an accurate representation of the code.

# Application Code

## C#

### Data

*FireAssistant.cs*

```
1.  using System;
2.  using System.Collections.Generic;
3.  using System.Diagnostics;
4.  using System.IO;
5.  using System.Linq;
6.  using System.Net;
7.  using System.Threading.Tasks;
8.  using Firebase.Database;
9.  using Firebase.Database.Query;
10. using Firebase.Storage;
11. using GymAndPAapp.Tables;
12. using Xamarin.Forms;
13.
14. namespace GymAndPAapp.ViewModels
15. {
16.     public class FireAssistant
17.     {
18.         public static FirebaseClient firebase = new FirebaseClient("https://gymanalyticsapp-default-rtdb.firebaseio.com/");
19.
20.         //Insert image
21.         public static async Task<string> SaveImage(Stream imgStream, string username)
22.         {
23.             try
24.             {
25.                 var webClient = new WebClient();
26.                 var image = await new FirebaseStorage("gymanalyticsapp.appspot.com")
27.                 .Child("ProfilePics")
28.                 .Child(username + ".jpeg")
29.                 .PutAsync(imgStream);
30.                 string imgurl = image;
31.                 return image;
32.             }
33.             catch (Exception e)
34.             {
35.                 Debug.WriteLine($"Error:{e}");
36.                 return null;
37.             }
38.         }
39.         //remove an image from storage
40.          public static async Task<bool> DelImage(string username)
41.          {
42.              try
43.              {
44.                 var webClient = new WebClient();
45.                 await new FirebaseStorage("gymanalyticsapp.appspot.com")
46.                 .Child("ProfilePics")
47.                 .Child(username + ".jpeg")
48.                 .DeleteAsync();
49.                     return true;
50.                 }
51.                 catch (Exception e)
52.                 {
53.                     Debug.WriteLine($"Error:{e}");
54.                     return false;
55.                 }
56.          }
57.         //Read All exercises
58.         public static async Task<List<Exercises>> GetExcercises()
59.         {
60.             try
61.             {
62.                 var exerciselist = (await firebase
```

```
63.             .Child("Exercises")
64.             .OnceAsync<Exercises>()).Select(item =>
65.             new Exercises
66.             {
67.                 id = item.Object.id,
68.                 name = item.Object.name,
69.             }).ToList();
70.             return exerciselist;
71.         }
72.         catch (Exception e)
73.         {
74.             Debug.WriteLine($"Error:{e}");
75.             return null;
76.         }
77.     }
78.     //Add an exercise to the database
79.     public static async Task<bool> AddExercise(string name)
80.     {
81.         try
82.         {
83.             var current = GetExcercises();
84.             int ID;
85.                 if(current.Id == 1)
86.                   {
87.                 ID = 1;
88.                   }
89.                 else
90.             {
91.                 ID = current.Id;
92.             }
93.             await firebase
94.             .Child("Exercises")
95.             .PostAsync(new Exercises() { id = ID, name = name });
96.             return true;
97.         }
98.         catch (Exception e)
99.         {
100.            Debug.WriteLine($"Error:{e}");
101.            return false;
102.        }
103.    }
104.    //Read All clients
105.    public static async Task<List<Clients>> GetAllClients()
106.    {
107.        try
108.        {
109.            var clientlist = (await firebase
110.            .Child("Clients")
111.            .OnceAsync<Clients>()).Select(item =>
112.            new Clients
113.            {
114.                UserName = item.Object.UserName,
115.                Token = item.Object.Token,
116.                TrainerName = item.Object.TrainerName,
117.                PlanTitle = item.Object.PlanTitle,
118.                Email = item.Object.Email
119.            }).ToList();
120.            return clientlist;
121.        }
122.        catch (Exception e)
123.        {
124.            Debug.WriteLine($"Error:{e}");
125.            return null;
126.        }
127.    }
128.    //Read All trainers
129.    public static async Task<List<Trainers>> GetAllTrainers()
130.    {
131.        try
132.        {
133.            var trainerlist = (await firebase
```

```
            .Child("Trainers")
            .OnceAsync<Trainers>()).Select(item =>
            new Trainers
            {
                UserName = item.Object.UserName,
                Token = item.Object.Token,
                PaypalAddress = item.Object.PaypalAddress
            }).ToList();
            return trainerlist;
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return null;
        }
    }
    //Read All plans
    public static async Task<List<Plans>> GetAllPlans()
    {
        try
        {
            var planlist = (await firebase
            .Child("Plans")
            .OnceAsync<Plans>()).Select(item =>
            new Plans
            {
                PlanTitle = item.Object.PlanTitle,
                Monday = item.Object.Monday,
                Tuesday = item.Object.Tuesday,
                Wednesday = item.Object.Wednesday,
                Thursday = item.Object.Thursday,
                Friday = item.Object.Friday,
                Saturday = item.Object.Saturday,
                Sunday = item.Object.Sunday,
                TrainerName = item.Object.TrainerName,
                Description = item.Object.Description,
                Calorie = item.Object.Calorie,
                Fee = item.Object.Fee,
                Link = item.Object.Link,
                Client = item.Object.Client,
                active = item.Object.active
            }).ToList();
            return planlist;
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return null;
        }
    }
    //Read All goals
    public static async Task<List<Goals>> GetAllGoals()
    {
        try
        {
            var planlist = (await firebase
            .Child("Goals")
            .OnceAsync<Goals>()).Select(item =>
            new Goals
            {
                Goal = item.Object.Goal,
                Done = item.Object.Done,
                UserName = item.Object.UserName
            }).ToList();
            return planlist;
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return null;
        }
```

```
205.        }
206.        //Read All requests
207.        public static async Task<List<Requests>> GetAllRequests()
208.        {
209.            try
210.            {
211.                var planlist = (await firebase
212.                .Child("Requests")
213.                .OnceAsync<Requests>()).Select(item =>
214.                new Requests
215.                {
216.                    ClientName = item.Object.ClientName,
217.                    TrainerName = item.Object.TrainerName,
218.                    Age = item.Object.Age,
219.                    Height = item.Object.Height,
220.                    Goals = item.Object.Goals,
221.                    Equipment = item.Object.Equipment,
222.                    Weight = item.Object.Weight
223.                }).ToList();
224.                return planlist;
225.            }
226.            catch (Exception e)
227.            {
228.                Debug.WriteLine($"Error:{e}");
229.                return null;
230.            }
231.        }
232.        //Read All payments
233.        public static async Task<List<Payments>> GetAllPayments()
234.        {
235.            try
236.            {
237.                var paymentlist = (await firebase
238.                .Child("Payments")
239.                .OnceAsync<Payments>()).Select(item =>
240.                new Payments
241.                {
242.                    ClientName = item.Object.ClientName,
243.                    TrainerName = item.Object.TrainerName,
244.                    PlanTitle = item.Object.PlanTitle,
245.                    Amount = item.Object.Amount,
246.                    Date = item.Object.Date
247.                }).ToList();
248.                return paymentlist;
249.            }
250.            catch (Exception e)
251.            {
252.                Debug.WriteLine($"Error:{e}");
253.                return null;
254.            }
255.        }
256.
257.        //get admin details from db
258.        public static async Task<List<Admin>> GetAdminDetails()
259.        {
260.            try
261.            {
262.                var adminlist = (await firebase
263.                .Child("Admin")
264.                .OnceAsync<Admin>()).Select(item =>
265.                new Admin
266.                {
267.                    UserName = item.Object.UserName,
268.                    Token = item.Object.Token
269.                }).ToList();
270.                return adminlist;
271.            }
272.            catch (Exception e)
273.            {
274.                Debug.WriteLine($"Error:{e}");
275.                return null;
```

```
276.
277.         }
278.     }
279.     //Read Admin
280.     public static async Task<Admin> GetAdmin(string token)
281.     {
282.         try
283.         {
284.             var getAdminDetails = await GetAdminDetails();
285.             await firebase
286.             .Child("Admin")
287.             .OnceAsync<Admin>();
288.             return getAdminDetails.Where(a => a.Token == token).FirstOrDefault();
289.         }
290.         catch (Exception e)
291.         {
292.             Debug.WriteLine($"Error:{e}");
293.             return null;
294.         }
295.     }
296.     //Read Requests
297.     public static async Task<List<Requests>> GetRequests(string trainername)
298.     {
299.         try
300.         {
301.             var getRequests = await GetAllRequests();
302.             await firebase
303.             .Child("Requests")
304.             .OnceAsync<Requests>();
305.             return getRequests.Where(a => a.TrainerName == trainername).Select(item =>
306.             new Requests
307.             {
308.                 ClientName = item.ClientName,
309.                 TrainerName = item.TrainerName,
310.                 Age = item.Age,
311.                 Height = item.Height,
312.                 Goals = item.Goals,
313.                 Equipment = item.Equipment,
314.                 Weight = item.Weight
315.             }).ToList();
316.         }
317.         catch (Exception e)
318.         {
319.             Debug.WriteLine($"Error:{e}");
320.             return null;
321.         }
322.     }
323.     //Read Client
324.     public static async Task<Clients> GetClient(string token)
325.     {
326.         try
327.         {
328.             var allClients = await GetAllClients();
329.             await firebase
330.             .Child("Clients")
331.             .OnceAsync<Clients>();
332.             return allClients.Where(a => a.Token == token).FirstOrDefault();
333.         }
334.         catch (Exception e)
335.         {
336.             Debug.WriteLine($"Error:{e}");
337.             return null;
338.         }
339.     }
340.     //check username taken or not
341.     public static async Task<Clients> CheckUsernameClient(string username)
342.     {
343.         try
344.         {
345.             var allClients = await GetAllClients();
346.             await firebase
```

```
            .Child("Clients")
            .OnceAsync<Clients>();
            return allClients.Where(a => a.UserName == username).FirstOrDefault();
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return null;
        }
    }
    //check username taken or not
    public static async Task<Trainers> CheckUsernameTrainer(string username)
    {
        try
        {
            var allTrainers = await GetAllTrainers();
            await firebase
            .Child("Trainers")
            .OnceAsync<Trainers>();
            return allTrainers.Where(a => a.UserName == username).FirstOrDefault();
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return null;
        }
    }
    //get clients of trainer
    public static async Task<List<Clients>> GetTrainerClients(string username)
    {
        try
        {
            var allClients = await GetAllClients();
            await firebase
            .Child("Clients")
            .OnceAsync<Clients>();
            return allClients.Where(a => a.TrainerName == username).Select(item =>
            new Clients
            {
                UserName = item.UserName,
                PlanTitle = item.PlanTitle
            }).ToList();
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return null;
        }
    }
    //Read Trainer
    public static async Task<Trainers> GetTrainer(string token)
    {
        try
        {
            var allTrainers = await GetAllTrainers();
            await firebase
            .Child("Trainers")
            .OnceAsync<Trainers>();
            return allTrainers.Where(a => a.Token == token).FirstOrDefault();
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return null;
        }
    }
    //Read Plans for a given trainer
    public static async Task<List<Plans>> GetPlan(string trainername, string username)
    {
        try
        {
```

```
418.             var planlist = await GetAllPlans();
419.             await firebase
420.             .Child("Plans")
421.             .OnceAsync<Plans>();
422.             var check1 = planlist.Where(a => a.TrainerName == trainername && a.Client == null && a.active ==
    true).Select(item =>
423.             new Plans
424.             {
425.                 PlanTitle = item.PlanTitle,
426.                 Monday = item.Monday,
427.                 Tuesday = item.Tuesday,
428.                 Wednesday = item.Wednesday,
429.                 Thursday = item.Thursday,
430.                 Friday = item.Friday,
431.                 Saturday = item.Saturday,
432.                 Sunday = item.Sunday,
433.                 TrainerName = item.TrainerName,
434.                 Calorie = item.Calorie,
435.                 Fee = item.Fee,
436.                 Description = item.Description,
437.                 Link = item.Link
438.             }).ToList();
439.             var check2 = planlist.Where(a => a.TrainerName == trainername && a.Client == username && a.active ==
    true).Select(item =>
440.                 new Plans
441.                 {
442.                     PlanTitle = item.PlanTitle,
443.                     Monday = item.Monday,
444.                     Tuesday = item.Tuesday,
445.                     Wednesday = item.Wednesday,
446.                     Thursday = item.Thursday,
447.                     Friday = item.Friday,
448.                     Saturday = item.Saturday,
449.                     Sunday = item.Sunday,
450.                     TrainerName = item.TrainerName,
451.                     Calorie = item.Calorie,
452.                     Fee = item.Fee,
453.                     Description = item.Description,
454.                     Link = item.Link
455.                 }).ToList();
456.             var list = check1.Concat(check2).ToList();
457.             return list;
458.         }
459.         catch (Exception e)
460.         {
461.             Debug.WriteLine($"Error:{e}");
462.             return null;
463.         }
464.     }
465.     //Get All plans for a given trainer
466.     public static async Task<List<Plans>> GetPlanTrainer(string trainername)
467.     {
468.         try
469.         {
470.             var planlist = await GetAllPlans();
471.             await firebase
472.             .Child("Plans")
473.             .OnceAsync<Plans>();
474.             return planlist.Where(a => a.TrainerName == trainername && a.active == true).Select(item =>
475.              new Plans
476.              {
477.                 PlanTitle = item.PlanTitle,
478.                 Monday = item.Monday,
479.                 Tuesday = item.Tuesday,
480.                 Wednesday = item.Wednesday,
481.                 Thursday = item.Thursday,
482.                 Friday = item.Friday,
483.                 Saturday = item.Saturday,
484.                 Sunday = item.Sunday,
485.                 TrainerName = item.TrainerName,
486.                 Calorie = item.Calorie,
```

```
                Fee = item.Fee,
                Description = item.Description,
                Link = item.Link,
                Client = item.Client
            }).ToList();
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return null;
        }
    }
    //Read goals for a given client
    public static async Task<List<Goals>> GetGoal(string username)
    {
        try
        {
            var planlist = await GetAllGoals();
            await firebase
            .Child("Goals")
            .OnceAsync<Goals>();
            return planlist.Where(a => a.UserName == username).Select(item =>
            new Goals
            {
                Goal = item.Goal,
                Done = item.Done,
            }).ToList();
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return null;
        }
    }
    // gets clients workout (plans even deleted stay here)
    public static async Task<Plans> GetPlan2(string trainername, string plantitle)
    {
        try
        {
            var planlist = await GetAllPlans();
            await firebase
            .Child("Plans")
            .OnceAsync<Plans>();
            return planlist.Where(a => a.TrainerName == trainername && a.PlanTitle == plantitle).FirstOrDefault();
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return null;
        }
    }
    //Read Payments for a given trainer
    public static async Task<List<Payments>> GetPayments(string trainername)
    {
        try
        {
            var paymentlist = await GetAllPayments();
            await firebase
            .Child("Payments")
            .OnceAsync<Payments>();
            return paymentlist.Where(a => a.TrainerName == trainername).Select(item =>
            new Payments
            {
                ClientName = item.ClientName,
                TrainerName = item.TrainerName,
                PlanTitle = item.PlanTitle,
                Amount = item.Amount,
                Date = item.Date
            }).ToList();
        }
        catch (Exception e)
```

```
558.          {
559.              Debug.WriteLine($"Error:{e}");
560.              return null;
561.          }
562.      }
563.      //Insert client
564.      public static async Task<bool> AddClient(string username, string token, string email)
565.      {
566.          try
567.          {
568.              await firebase
569.              .Child("Clients")
570.              .PostAsync(new Clients() { UserName = username, Token = token, Email=email });
571.              return true;
572.          }
573.          catch (Exception e)
574.          {
575.              Debug.WriteLine($"Error:{e}");
576.              return false;
577.          }
578.      }
579.      //Insert Trainer
580.      public static async Task<bool> AddTrainer(string username, string token, string paypal)
581.      {
582.          try
583.          {
584.              await firebase
585.              .Child("Trainers")
586.              .PostAsync(new Trainers() { UserName = username, Token = token, PaypalAddress = paypal });
587.              return true;
588.          }
589.          catch (Exception e)
590.          {
591.              Debug.WriteLine($"Error:{e}");
592.              return false;
593.          }
594.      }
595.      // add admin
596.      public static async Task<bool> AddAdmin(string username, string token)
597.      {
598.          try
599.          {
600.              await firebase
601.              .Child("Admin")
602.              .PostAsync(new Admin() { UserName = username, Token = token });
603.              return true;
604.          }
605.          catch (Exception e)
606.          {
607.              Debug.WriteLine($"Error:{e}");
608.              return false;
609.          }
610.      }
611.      //Insert Plan
612.      public static async Task<bool> AddPlan(string username, string title, string mon, string tue, string wed, string thur,
      string fri, string sat, string sun, string description,string calorie, string link, string fee, string client)
613.      {
614.          try
615.          {
616.              await firebase
617.              .Child("Plans")
618.              .PostAsync(new Plans() { TrainerName = username, PlanTitle = title, Monday = mon, Tuesday = tue,
      Wednesday=wed,Thursday=thur,Friday=fri,Saturday=sat,Sunday=sun, Description = description, Calorie = calorie, Link
      = link, Fee = fee, Client = client, active = true });
619.              return true;
620.          }
621.          catch (Exception e)
622.          {
623.              Debug.WriteLine($"Error:{e}");
624.              return false;
625.          }
```

```
626.        }
627.        //Insert Goal
628.        public static async Task<bool> AddGoal(string username, string goal)
629.        {
630.            try
631.            {
632.                await firebase
633.                .Child("Goals")
634.                .PostAsync(new Goals() { UserName = username, Goal = goal });
635.                return true;
636.            }
637.            catch (Exception e)
638.            {
639.                Debug.WriteLine($"Error:{e}");
640.                return false;
641.            }
642.        }
643.        //Insert Payment
644.        public static async Task<bool> AddPayment(string username, string amount, String date, string trainername, string
    plan)
645.        {
646.            string dateNow = date.ToString();
647.            try
648.            {
649.                await firebase
650.                .Child("Payments")
651.                .PostAsync(new Payments() { TrainerName = trainername, Amount = "€" + amount, ClientName = username,
    Date = dateNow, PlanTitle = plan });
652.                return true;
653.            }
654.            catch (Exception e)
655.            {
656.                Debug.WriteLine($"Error:{e}");
657.                return false;
658.            }
659.        }
660.        //Make Request
661.        public static async Task<bool> MakeRequest(string username, string trainername, string age, string height, string
    goals, string equipment, string weight)
662.        {
663.            try
664.            {
665.                await firebase
666.                .Child("Requests")
667.                .PostAsync(new Requests() { TrainerName = trainername, Age = age, ClientName = username, Height =
    height, Goals = goals, Equipment = equipment, Weight = weight });
668.                return true;
669.            }
670.            catch (Exception e)
671.            {
672.                Debug.WriteLine($"Error:{e}");
673.                return false;
674.            }
675.        }
676.        //Update Client add trainer
677.        public static async Task<bool> UpdateClient2(string username,string TrainerName)
678.        {
679.            try
680.            {
681.                var toUpdateClient = (await firebase
682.                .Child("Clients")
683.                .OnceAsync<Clients>()).Where(a => a.Object.UserName == username).FirstOrDefault();
684.                await firebase
685.                .Child("Clients")
686.                .Child(toUpdateClient.Key)
687.                .PutAsync(new Clients() { UserName = username, Token = toUpdateClient.Object.Token, TrainerName =
    TrainerName, PlanTitle = toUpdateClient.Object.PlanTitle });
688.                return true;
689.            }
690.            catch (Exception e)
691.            {
```

```
692.                 Debug.WriteLine($"Error:{e}");
693.                 return false;
694.             }
695.         }
696.         //Update Client add trainer
697.         public static async Task<bool> UpdateClient3(string token, string Title)
698.         {
699.             try
700.             {
701.                 var toUpdateClient = (await firebase
702.                 .Child("Clients")
703.                 .OnceAsync<Clients>()).Where(a => a.Object.Token == token).FirstOrDefault();
704.                 await firebase
705.                 .Child("Clients")
706.                 .Child(toUpdateClient.Key)
707.                 .PutAsync(new Clients() { UserName = toUpdateClient.Object.UserName, Token =
    toUpdateClient.Object.Token, TrainerName = toUpdateClient.Object.TrainerName, PlanTitle = Title });
708.                 return true;
709.             }
710.             catch (Exception e)
711.             {
712.                 Debug.WriteLine($"Error:{e}");
713.                 return false;
714.             }
715.         }
716.         //Update Goal
717.         public static async Task<bool> UpdateGoal(string username, string goal)
718.         {
719.             try
720.             {
721.                 var toUpdateGoal = (await firebase
722.                 .Child("Goals")
723.                 .OnceAsync<Goals>()).Where(a => a.Object.UserName == username && a.Object.Goal ==
    goal).FirstOrDefault();
724.                 await firebase
725.                 .Child("Goals")
726.                 .Child(toUpdateGoal.Key)
727.                 .PutAsync(new Goals() { UserName = username, Goal = goal,Done = true });
728.                 return true;
729.             }
730.             catch (Exception e)
731.             {
732.                 Debug.WriteLine($"Error:{e}");
733.                 return false;
734.             }
735.         }
736.         //Update Plan
737.         public static async Task<bool> UpdatePlan(string trainername, string mon, string tue,string wed,string thur,string
    fri, string sat, string sun, string plantitle, string oldtitle, string description,string calorie ,string fee, string client, string
    link)
738.         {
739.             try
740.             {
741.                 var toUpdatePlan = (await firebase
742.                 .Child("Plans")
743.                 .OnceAsync<Plans>()).Where(a => a.Object.TrainerName == trainername && a.Object.PlanTitle ==
    oldtitle).FirstOrDefault();
744.                 var countClients = (await firebase
745.                 .Child("Clients")
746.                 .OnceAsync<Clients>()).Where(a => a.Object.TrainerName == trainername && a.Object.PlanTitle ==
    oldtitle).Count();
747.                 for( int i = 0; i< countClients;i++)
748.                 {
749.                  var toUpdateClient = (await firebase
750.                 .Child("Clients")
751.                 .OnceAsync<Clients>()).Where(a => a.Object.TrainerName == trainername && a.Object.PlanTitle ==
    oldtitle).FirstOrDefault();
752.                     await firebase
753.                     .Child("Clients")
754.                     .Child(toUpdateClient.Key)
```

```
                .PutAsync(new Clients() { UserName = toUpdateClient.Object.UserName, Token =
toUpdateClient.Object.Token, TrainerName = toUpdateClient.Object.TrainerName, PlanTitle = plantitle });
            }
                await firebase
                .Child("Plans")
                .Child(toUpdatePlan.Key)
                .PutAsync(new Plans() { TrainerName = trainername, PlanTitle = plantitle, Monday = mon,
Tuesday=tue,Wednesday=wed,Thursday=thur,Friday=fri,Saturday=sat,Sunday=sun, Fee = fee, Calorie = calorie,
Description = description, Client = client, Link = link, active = toUpdatePlan.Object.active  });
                return true;
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return false;
        }
    }
    //Delete Client
    public static async Task<bool> DeleteClient(string username, string token)
    {
        try
        {
            var toDeleteClient = (await firebase
            .Child("Clients")
            .OnceAsync<Clients>()).Where(a => a.Object.UserName == username && a.Object.Token ==
token).FirstOrDefault();
            await firebase.Child("Clients").Child(toDeleteClient.Key).DeleteAsync();
            return true;
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return false;
        }
    }
    //Delete Trainer
    public static async Task<bool> DeleteTrainer(string username, string token)
    {
        try
        {
            var toDeleteTrainer = (await firebase
            .Child("Trainers")
            .OnceAsync<Trainers>()).Where(a => a.Object.UserName == username && a.Object.Token ==
token).FirstOrDefault();
            await firebase.Child("Trainers").Child(toDeleteTrainer.Key).DeleteAsync();
            return true;
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return false;
        }
    }
  /*   public static async Task<bool> DeleteAdmin(string username, string token)
    {
        try
        {
            var toDeleteTrainer = (await firebase
            .Child("Admin")
            .OnceAsync<Admin>()).Where(a => a.Object.UserName == username && a.Object.Token ==
token).FirstOrDefault();
            await firebase.Child("Admin").Child(toDeleteTrainer.Key).DeleteAsync();
            return true;
        }
        catch (Exception e)
        {
            Debug.WriteLine($"Error:{e}");
            return false;
        }
    }
  */
```

```
820.        //Delete Goal
821.        public static async Task<bool> DeleteGoal(string username, string goal)
822.        {
823.            try
824.            {
825.                var toDeleteGoal = (await firebase
826.                .Child("Goals")
827.                .OnceAsync<Goals>()).Where(a => a.Object.UserName == username && a.Object.Goal ==
    goal).FirstOrDefault();
828.                await firebase.Child("Goals").Child(toDeleteGoal.Key).DeleteAsync();
829.                return true;
830.            }
831.            catch (Exception e)
832.            {
833.                Debug.WriteLine($"Error:{e}");
834.                return false;
835.            }
836.        }
837.        //Delete Request
838.        public static async Task<bool> DeleteRequest(string trainername, string clientname)
839.        {
840.            try
841.            {
842.                var toDeleteGoal = (await firebase
843.                .Child("Requests")
844.                .OnceAsync<Requests>()).Where(a => a.Object.ClientName == clientname && a.Object.TrainerName ==
    trainername).FirstOrDefault();
845.                await firebase.Child("Requests").Child(toDeleteGoal.Key).DeleteAsync();
846.                return true;
847.            }
848.            catch (Exception e)
849.            {
850.                Debug.WriteLine($"Error:{e}");
851.                return false;
852.            }
853.        }
854.        //Delete Plan (but allow paid users to still own it)
855.        public static async Task<bool> DeletePlan(string trainer, string plantitle)
856.        {
857.            try
858.            {
859.                var toDelPlan = (await firebase
860.                .Child("Plans")
861.                .OnceAsync<Plans>()).Where(a => a.Object.TrainerName == trainer && a.Object.PlanTitle ==
    plantitle).FirstOrDefault();
862.                await firebase
863.                    .Child("Plans")
864.                    .Child(toDelPlan.Key)
865.                    .PutAsync(new Plans() { TrainerName = trainer, PlanTitle = plantitle, Monday = toDelPlan.Object.Monday,
    Tuesday = toDelPlan.Object.Tuesday, Wednesday = toDelPlan.Object.Wednesday, Thursday =
    toDelPlan.Object.Thursday, Friday = toDelPlan.Object.Friday, Saturday = toDelPlan.Object.Saturday, Sunday =
    toDelPlan.Object.Sunday, Fee = toDelPlan.Object.Fee, Calorie = toDelPlan.Object.Calorie, Description =
    toDelPlan.Object.Description, Client = toDelPlan.Object.Client, Link = toDelPlan.Object.Link, active = false });
866.                return true;
867.            }
868.            catch (Exception e)
869.            {
870.                Debug.WriteLine($"Error:{e}");
871.                return false;
872.            }
873.        }
874.    }
875. }
876.
```

## *PayPalClient.cs*

```
1.  using System;
2.  using PayoutsSdk.Core;
```

```
using PayoutsSdk.Payouts;
using PayPalHttp;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace GymAndPAapp.Data
{
    public class PayPalClient
    {
        /**
            Set up PayPal environment with sandbox credentials.
            In production, use LiveEnvironment.
         */
        public static PayPalEnvironment environment()
        {
            return new SandboxEnvironment(
                System.Environment.GetEnvironmentVariable("PAYPAL_CLIENT_ID") != null ?
                System.Environment.GetEnvironmentVariable("PAYPAL_CLIENT_ID") :
"AbQjSy5RAXfTI0C9qSjfOSORy9YIRvORgKMkmxvwL0xGcwhqqGj_Tzqrq1OOGv1T0SxXL2ZYUgMQ_Ay1",
                System.Environment.GetEnvironmentVariable("PAYPAL_CLIENT_SECRET") != null ?
                System.Environment.GetEnvironmentVariable("PAYPAL_CLIENT_SECRET") :
"ENjFx5t8fBwNbNxiRFO3_y6XJdxQikUH0Sf83kOeE45NcR0SXYNQhD0vbPwhX1f_1axVYra0PwQo9KYk");
        }
        /**
            Returns PayPalHttpClient instance to invoke PayPal APIs.
         */
        public static HttpClient client()
        {
            return new PayPalHttpClient(environment());
        }
        public static HttpClient client(string refreshToken)
        {
            return new PayPalHttpClient(environment(), refreshToken);
        }
    }
}

```

## Tables

### *Admin.cs*

```
namespace GymAndPAapp.Tables
{
    public class Admin
    {
        public string UserName { get; set; }
        public string Token { get; set; }
    }
}
```

### *Clients.cs*

```
namespace GymAndPAapp.Tables
{
    public class Clients
    {
        public string UserName { get; set; }
        public string Token { get; set; }
        public string TrainerName { get; set; }
        public string PlanTitle { get; set; }
        public string Email { get; set; }

    }
}
```

## *Exercises.cs*

```
1.  namespace GymAndPAapp.Tables
2.  {
3.      public class Exercises
4.      {
5.          public int id { get; set; }
6.          public string name { get; set; }
7.      }
8.  }
```

## *Goals.cs*

```
1.  namespace GymAndPAapp.Tables
2.  {
3.      public class Goals
4.      {
5.          public string UserName { get; set; }
6.          public string Goal { get; set; }
7.          public bool Done { get; set; }
8.
9.      }
10. }
```

## *Payments.cs*

```
1.  namespace GymAndPAapp.Tables
2.  {
3.      public class Payments
4.      {
5.          public string ClientName { get; set; }
6.          public string Amount { get; set; }
7.          public string TrainerName { get; set; }
8.          public string Date { get; set; }
9.          public string PlanTitle { get; set; }
10.     }
11. }
```

## *Plans.cs*

```
1.  namespace GymAndPAapp.Tables
2.  {
3.      public class Plans
4.      {
5.          public bool active { get; set; }
6.          public string TrainerName { get; set; }
7.          public string PlanTitle { get; set; }
8.          public string Description { get; set; }
9.          public string Monday { get; set; }
10.         public string Tuesday { get; set; }
11.         public string Wednesday { get; set; }
12.         public string Thursday { get; set; }
13.         public string Friday { get; set; }
14.         public string Saturday { get; set; }
15.         public string Sunday { get; set; }
16.         public string Calorie { get; set; }
17.         public string Fee { get; set; }
18.         public string Link  { get; set; }
19.         public string Client { get; set; }
20.     }
21. }
```

### *Requests.cs*

```
1.  namespace GymAndPAapp.Tables
2.  {
3.      public class Requests
4.      {
5.          public string ClientName { get; set; }
6.          public string TrainerName { get; set; }
7.          public string Age { get; set; }
8.          public string Height { get; set; }
9.          public string Weight { get; set; }
10.         public string Goals { get; set; }
11.         public string Equipment { get; set; }
12.     }
13. }
14.
```

### *Trainers.cs*

```
1.  namespace GymAndPAapp.Tables
2.  {
3.      public class Trainers
4.      {
5.          public string UserName { get; set; }
6.          public string Token { get; set; }
7.          public string PaypalAddress { get; set; }
8.
9.      }
10. }
```

## ViewModels

### *ChangPassViewModel.cs*

```
1.  using GymAndPAapp.Views;
2.  using GymAndPAapp.Views.Admin;
3.  using System;
4.  using System.Collections.Generic;
5.  using System.ComponentModel;
6.  using System.Text;
7.  using Xamarin.Forms;
8.
9.  namespace GymAndPAapp.ViewModels
10. {
11.     class ChangePassViewModel
12.     {
13.         IAuthenticationService auth;
14.         public ChangePassViewModel()
15.         {
16.             auth = DependencyService.Get<IAuthenticationService>();
17.         }
18.         public string Username = App._variable;
19.         public string NewPassword { get; set; }
20.         public string ConfirmNewPassword { get; set; }
21.
22.         public Command ChangePasswordCommand
23.         {
24.             get
25.             {
26.                 return new Command(ChangePassword);
27.             }
28.         }
29.         public Command Cancel
30.         {
31.             get
32.             {
```

```
33.             return new Command(CancelPass);
34.         }
35.     }
36.     private async void CancelPass()
37.     {
38.         var token = auth.GetUser();
39.         var clientCheck = await FireAssistant.GetClient(token);
40.         var trainerCheck = await FireAssistant.GetTrainer(token);
41.         if (clientCheck != null)
42.         {
43.             await App.Current.MainPage.Navigation.PushAsync(new TabbedClient());
44.         }
45.         else if (trainerCheck != null)
46.         {
47.             await App.Current.MainPage.Navigation.PushAsync(new TabbedTrainer());
48.         }
49.         else
50.         {
51.             await App.Current.MainPage.Navigation.PushAsync(new TabbedAdmin());
52.         }
53.     }
54.     private async void ChangePassword()
55.     {
56.         if ( string.IsNullOrEmpty(NewPassword) || string.IsNullOrEmpty(ConfirmNewPassword))
57.             await App.Current.MainPage.DisplayAlert("Empty Values", "Please ensure all fields completed", "OK");
58.         else if(NewPassword != ConfirmNewPassword)
59.             await App.Current.MainPage.DisplayAlert("Dont match", "New passwords dont match", "OK");
60.         else
61.         {
62.             auth.updatePassword(NewPassword);
63.             CancelPass();
64.         }
65.     }
66.
67. }
68. }
```

## *GoalsViewModel.cs*

```
1.  using GymAndPAapp.Views;
2.  using GymAndPAapp.Views.Clients;
3.  using Rg.Plugins.Popup.Services;
4.  using System.ComponentModel;
5.  using Xamarin.Forms;
6.
7.  namespace GymAndPAapp.ViewModels
8.  {
9.      public class GoalsViewModel : INotifyPropertyChanged
10.     {
11.         public event PropertyChangedEventHandler PropertyChanged;
12.         public GoalsViewModel()
13.         {
14.
15.         }
16.         public string Username = App._variable;
17.
18.         public string Goal { get; set; }
19.
20.         public Command AddCommand
21.         {
22.             get
23.             {
24.                 return new Command(check);
25.             }
26.         }
27.         private async void check()
28.         {
29.             if (Goal == null)
30.             {
```

```
31.                 App._goalsVar = true;
32.                 await App.Current.MainPage.DisplayAlert("Error", "Goal Value is empty, please try again", "Ok");
33.                 await PopupNavigation.Instance.PopAllAsync();
34.                 App.Current.MainPage = new NavigationPage(new TabbedClient());
35.             }
36.             else
37.             {
38.                 createGoal();
39.             }
40.         }
41.
42.         private async void createGoal()
43.         {
44.
45.             var addgoal = await FireAssistant.AddGoal(Username, Goal);
46.             if (addgoal)
47.             {
48.                 App._goalsVar = true;
49.                 await App.Current.MainPage.DisplayAlert("Goal Created", "", "Ok");
50.                 await PopupNavigation.Instance.PopAllAsync();
51.                 App.Current.MainPage = new NavigationPage(new TabbedClient());
52.             }
53.             else
54.             {
55.                 App._goalsVar = true;
56.                 await App.Current.MainPage.DisplayAlert("Error", "Failed to add goal", "OK");
57.                 await PopupNavigation.Instance.PopAllAsync();
58.                 App.Current.MainPage = new NavigationPage(new TabbedClient());
59.             }
60.
61.         }
62.     }
63. }
64.
```

## *IAuthenticationServices.cs*

```
1.  using GymAndPAapp.Tables;
2.  using System;
3.  using System.Collections.Generic;
4.  using System.Text;
5.  using System.Threading.Tasks;
6.
7.  namespace GymAndPAapp.ViewModels
8.  {
9.      public interface IAuthenticationService
10.     {
11.         Task<string> LoginWithEmail(string email, string password);
12.
13.         Task<string> SignUpWithEmail(string email, string password);
14.         Task<string> AdminRecoveryAsync(string token);
15.         // Task<string> AdminCloseAccount(string token); //currently unsupported
16.         string GetUser();
17.         string GetEmail();
18.         bool CloseAccount();
19.         bool UpdateEmail(string email);
20.         bool SignOut();
21.         bool IsSignIn();
22.         bool resetEmail(string email);
23.         bool updatePassword(string password);
24.     }
25. }
```

*LoginPageViewModel.cs*

```
1.  using GymAndPAapp.Views;
2.  using GymAndPAapp.Views.Admin;
3.  using GymAndPAapp.Views.Shared;
4.  using System.ComponentModel;
5.  using Xamarin.Forms;
6.
7.  namespace GymAndPAapp.ViewModels
8.  {
9.      public class LoginPageViewModel
10.     {
11.         IAuthenticationService auth;
12.
13.         // public event PropertyChangedEventHandler PropertyChanged;
14.         public LoginPageViewModel()
15.         {
16.             auth = DependencyService.Get<IAuthenticationService>();
17.         }
18.
19.         public string Email { get; set; }
20.         public string Password { get; set; }
21.
22.         public Command LoginCommand
23.         {
24.             get
25.             {
26.                 return new Command(Login);
27.             }
28.         }
29.         public Command SignUpCommand
30.         {
31.             get
32.             {
33.                 return new Command(() => { App.Current.MainPage.Navigation.PushAsync(new RegistrationPage()); });
34.             }
35.         }
36.         public Command ResetPassCommand
37.         {
38.             get
39.             {
40.                 return new Command(() => { App.Current.MainPage.Navigation.PushAsync(new ResetPasswordPage()); });
41.             }
42.         }
43.
44.         private async void Login()
45.         {
46.
47.             if (string.IsNullOrEmpty(Email) || string.IsNullOrEmpty(Password))
48.                 await App.Current.MainPage.DisplayAlert("Empty Values", "Please enter Username and Password", "OK");
49.             else
50.             {
51.                 string token = await auth.LoginWithEmail(Email, Password);
52.
53.                 if(token != string.Empty)
54.                 {
55.                     var Clients = await FireAssistant.GetClient(token);
56.                     var Trainers = await FireAssistant.GetTrainer(token);
57.                     if (Clients != null)
58.                     {
59.                         App._variable = Clients.UserName;
60.                         App._variable2 = Password;
61.                         if (App.DayOfWeek.DayOfWeek.ToString() != App._variable8)
62.                         {
63.                             App._variable7 = 0;
64.                             App._variable6 = 0;
65.                         }
66.                         await App.Current.MainPage.Navigation.PushAsync(new TabbedClient());
67.                     }
68.                     else if (Trainers != null)
```

```
69.                 {
70.                     App._variable = Trainers.UserName;
71.                     App._variable2 = Password;
72.                     if (App.DayOfWeek.DayOfWeek.ToString() != App._variable8)
73.                     {
74.                         App._variable7 = 0;
75.                         App._variable6 = 0;
76.                     }
77.                     await App.Current.MainPage.Navigation.PushAsync(new TabbedTrainer());
78.                 }
79.                 else
80.                 {
81.                     var Admin = await FireAssistant.GetAdmin(token);
82.                     if (Admin != null)
83.                     {
84.                         App._variable = Admin.UserName;
85.                         App._variable2 = Password;
86.                         await App.Current.MainPage.Navigation.PushAsync(new TabbedAdmin());
87.                     }
88.                 }
89.             }
90.             else
91.             {
92.                 await App.Current.MainPage.DisplayAlert("Login Fail", "Please enter correct Username and Password",
    "OK");
93.             }
94.         }
95.     }
96. }
97. }
```

## *PayoutPaypal.cs*

```
1.  using System;
2.  using System.Collections.Generic;
3.  using System.Threading.Tasks;
4.
5.  using PayoutsSdk.Payouts;
6.  using PayoutsSdk.Core;
7.  using PayPalHttp;
8.
9.  using System.IO;
10. using System.Text;
11. using GymAndPAapp.Data;
12.
13. namespace GymAndPAapp.ViewModels
14. {
15.     public class PayoutPaypal
16.     {
17.         private static CreatePayoutRequest buildRequestBody(string email, string amount, string title, string Username)
18.         {
19.             var request = new CreatePayoutRequest()
20.             {
21.
22.                 SenderBatchHeader = new SenderBatchHeader()
23.                 {
24.                     EmailMessage = "You have recieved payment of €" + amount + " from " + Username + " for " + title,
25.                     EmailSubject = "You recieved a payment"
26.                 },
27.                 Items = new List<PayoutItem>(){
28.                 new PayoutItem()
29.                 {
30.                 RecipientType="EMAIL",
31.
32.                 Amount=new Currency(){
33.                     CurrencyCode="EUR",
34.                     Value=amount,
35.                  },
36.                 Receiver=email,
```

```
37.
38.             }
39.             }
40.             };
41.             return request;
42.
43.         }
44.         public async static Task<HttpResponse> CreatePayout(string email, string amount, string title, string Username)
45.         {
46.             PayoutsPostRequest request = new PayoutsPostRequest();
47.             request.RequestBody(buildRequestBody(email,amount,title,Username));
48.
49.             var response = await PayPalClient.client().Execute(request);
50.             var result = response.Result<CreatePayoutResponse>();
51.             return response;
52.         }
53.
54.     }
55. }
```

## *RegistrationViewModel.cs*

```
1.  using GymAndPAapp.Views;
2.  using System.Linq;
3.  using Xamarin.Forms;
4.
5.  namespace GymAndPAapp.ViewModels
6.  {
7.      class RegistrationViewModel
8.      {
9.          IAuthenticationService auth;
10.         public string UserName { get; set; }
11.         public string Password { get; set; }
12.         public string ConfirmPassword { get; set; }
13.         public string Email { get; set; }
14.         public bool CheckedBox { get; set; }
15.
16.         public RegistrationViewModel()
17.         {
18.             auth = DependencyService.Get<IAuthenticationService>();
19.         }
20.         public Command RegisterCommand
21.         {
22.             get
23.             {
24.                 return new Command(() =>
25.                 {
26.                     if (Password == ConfirmPassword)
27.                         register();
28.                     else
29.                         App.Current.MainPage.DisplayAlert("", "Password do not match!", "OK");
30.                 });
31.             }
32.         }
33.         private async void register()
34.         {
35.
36.             var clientusername = await FireAssistant.CheckUsernameClient(UserName);
37.             var trainerusername = await FireAssistant.CheckUsernameTrainer(UserName);
38.             if (string.IsNullOrEmpty(UserName) || string.IsNullOrEmpty(Password) ||
    string.IsNullOrEmpty(ConfirmPassword) || string.IsNullOrEmpty(Email))
39.             {
40.                 await App.Current.MainPage.DisplayAlert("Empty Values", "All fields required", "OK");
41.             }
42.             else if (clientusername != null || trainerusername != null)
43.             {
44.                 await App.Current.MainPage.DisplayAlert("Username Take", "This username is currently taken please try
    another", "OK");
45.             }
```

```
46.             else if (Password.Length > 8 || Password.Any(ch => char.IsUpper(ch)) || Password.Any(ch =>
    char.IsNumber(ch)))
47.             {
48.                 var user = auth.SignUpWithEmail(Email, Password);
49.                 string token = await auth.LoginWithEmail(Email, Password);
50.
51.                 if (user !=null)
52.                 {
53.                     if (CheckedBox == true)
54.                     {
55.
56.                         var trainer = await FireAssistant.AddTrainer(UserName, token, Email);
57.                         if(trainer)
58.                         {
59.                             await App.Current.MainPage.DisplayAlert("Success", "Account created", "ok");
60.                         }
61.                     }
62.                     else
63.                     {
64.                         var client = await FireAssistant.AddClient(UserName, token, Email);
65.                         if(client)
66.                         {
67.                             await App.Current.MainPage.DisplayAlert("Success", "Account created", "ok");
68.                         }
69.                     }
70.                     var signout = auth.SignOut();
71.                     if (signout)
72.                     {
73.                         await App.Current.MainPage.Navigation.PushAsync(new LoginPage());
74.                     }
75.                 }
76.                 else
77.                 {
78.                     await App.Current.MainPage.DisplayAlert("ERROR", "Account Not Created", "ok");
79.                     await App.Current.MainPage.Navigation.PushAsync(new RegistrationPage());
80.                 }
81.             }
82.             else
83.             {
84.                 await App.Current.MainPage.DisplayAlert("Password Invaild", "Password must meet requirments.\n - Atleast 8
    characters long.\n - Must contain a number", "OK");
85.             }
86.         }
87.     }
88. }
89.
```

## *ValueProgressBar.cs*

```
1.  using System;
2.  using Xamarin.Forms;
3.
4.  namespace CircularProgressApp
5.  {
6.      public class ValueProgressBar : IValueConverter
7.      {
8.          public object Convert(object value, Type targetType, object parameter,    System.Globalization.CultureInfo culture)
9.          {
10.             return (double)value / 100;
11.
12.         }
13.         public object ConvertBack(object value, Type targetType, object parameter, System.Globalization.CultureInfo
    culture)
14.         {
15.             return new NotImplementedException();
16.         }
17.     }
```

```
18. }
```

## Views

### *Admin*

### Addexe.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.              xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.              xmlns:d="http://xamarin.com/schemas/2014/forms/design"
5.              xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
6.              mc:Ignorable="d"
7.              x:Class="GymAndPAapp.Views.addexe"
8.              Shell.NavBarIsVisible="False">
9.      <ContentPage.Content>
10.         <ScrollView BackgroundColor ="#002d40">
11.             <StackLayout>
12.                 <Grid VerticalOptions="CenterAndExpand" Margin="40" RowSpacing="10">
13.                     <Grid.RowDefinitions>
14.                         <RowDefinition/>
15.                         <RowDefinition/>
16.                         <RowDefinition/>
17.                         <RowDefinition/>
18.                     </Grid.RowDefinitions>
19.                     <Grid.ColumnDefinitions>
20.                         <ColumnDefinition/>
21.                         <ColumnDefinition/>
22.                     </Grid.ColumnDefinitions>
23.                     <Entry Placeholder="Exercise name" TextColor="White"  x:Name="EntryEmail" Grid.Row="0"
    Grid.ColumnSpan="2"/>
24.                     <Button Text="Add Exercise" Grid.Row="2" Grid.ColumnSpan="2" FontSize="15"  HeightRequest="50"
    CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
25.                     <Button Text="Return" Grid.Row="3" Grid.ColumnSpan="2" FontSize="15"  HeightRequest="50"
    CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked_2"/>
26.                 </Grid>
27.             </StackLayout>
28.         </ScrollView>
29.     </ContentPage.Content>
30. </ContentPage>
31.
```

### Addexe.xaml.cs

```
1.  using GymAndPAapp.ViewModels;
2.  using GymAndPAapp.Views.Admin;
3.  using System;
4.  using System.Collections.Generic;
5.  using System.Linq;
6.  using System.Text;
7.  using System.Threading.Tasks;
8.
9.  using Xamarin.Forms;
10. using Xamarin.Forms.Xaml;
11.
12. namespace GymAndPAapp.Views
13. {
14.     [XamlCompilation(XamlCompilationOptions.Compile)]
15.     public partial class addexe : ContentPage
16.     {
17.         public addexe()
18.         {
19.             SetValue(NavigationPage.HasNavigationBarProperty, false);
20.             InitializeComponent();
21.         }
22.
23.         private async void Button_Clicked(object sender, EventArgs e)
```

```
        {
            var result = await FireAssistant.AddExercise(EntryEmail.Text);

        }
        private async void Button_Clicked_2(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new TabbedAdmin());

        }
    }
}

```

## AdminHome.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:Image="clr-namespace:ImageCircle.Forms.Plugin.Abstractions;assembly=ImageCircle.Forms.Plugin"
             x:Class="GymAndPAapp.Views.Admin.AdminHome">
    <ContentPage.Content>
        <ScrollView BackgroundColor="#002d40">
            <StackLayout>
                <Grid VerticalOptions="Center" Margin="30"  RowSpacing="30">
                    <Grid.RowDefinitions>
                        <RowDefinition Height="0"/>
                        <RowDefinition Height="220"/>
                        <RowDefinition Height="60"/>
                        <RowDefinition Height="50"/>
                        <RowDefinition Height="50"/>
                        <RowDefinition Height="50"/>
                        <RowDefinition Height="50"/>
                    </Grid.RowDefinitions>
                    <StackLayout Grid.Row="1">
                        <Image:CircleImage x:Name="imgChoose" Source="stock.jpg" HeightRequest="300" WidthRequest="300" Aspect="AspectFill">
                            <Image.GestureRecognizers>
                                <TapGestureRecognizer
                                    Tapped="TapGestureRecognizer_Tapped"
                                    NumberOfTapsRequired="1"/>
                            </Image.GestureRecognizers>
                        </Image:CircleImage>
                        <ActivityIndicator Grid.Row="1" x:Name="loading" IsRunning="False" Color="#ff4140"/>
                    </StackLayout>
                    <Button Text="Manage Clients"  FontSize="15" HeightRequest="50" Grid.Row="3" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked"/>
                    <Button Text="Manage Trainers"  FontSize="15" HeightRequest="50" Grid.Row="4" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked_1"/>
                    <Button Text="Add Exercise"  FontSize="15" HeightRequest="50" Grid.Row="5" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked_2"/>
                </Grid>
            </StackLayout>
        </ScrollView>
    </ContentPage.Content>
</ContentPage>

```

## AdminHome.xaml.cs

```
using Firebase.Storage;
using GymAndPAapp.Views.Shared;
using Rg.Plugins.Popup.Services;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Net;
```

```
9.  using System.Text;
10. using System.Threading.Tasks;
11. using Xamarin.Essentials;
12. using Xamarin.Forms;
13. using Xamarin.Forms.Xaml;
14.
15. namespace GymAndPAapp.Views.Admin
16. {
17.     [XamlCompilation(XamlCompilationOptions.Compile)]
18.     public partial class AdminHome : ContentPage
19.     {
20.         public AdminHome()
21.         {
22.             App._variable5 = null;
23.             SetValue(NavigationPage.HasNavigationBarProperty, false);
24.             InitializeComponent();
25.         }
26.         protected async override void OnAppearing()
27.         {
28.             var storage = await Permissions.CheckStatusAsync<Permissions.StorageRead>();
29.             var camera = await Permissions.CheckStatusAsync<Permissions.Camera>();
30.
31.             if (storage != PermissionStatus.Granted)
32.             {
33.                 storage = await Permissions.RequestAsync<Permissions.StorageRead>();
34.             }
35.             if (storage != PermissionStatus.Granted)
36.             {
37.                 return;
38.             }
39.             if (camera != PermissionStatus.Granted)
40.             {
41.                 camera = await Permissions.RequestAsync<Permissions.Camera>();
42.             }
43.             if (camera != PermissionStatus.Granted)
44.             {
45.                 return;
46.             }
47.
48.             var name = App._variable;
49.             base.OnAppearing();
50.             loading.IsRunning = true;
51.             if (App.profile == null)
52.             {
53.                 try
54.                 {
55.                     var webClient = new WebClient();
56.                     var getimage = await new FirebaseStorage("gymanalyticsapp.appspot.com")
57.                     .Child("ProfilePics")
58.                     .Child(name + ".jpeg")
59.                     .GetDownloadUrlAsync();
60.                     string imgurl = getimage;
61.                     byte[] imgbyte = webClient.DownloadData(imgurl);
62.                     App.profile = ImageSource.FromStream(() => new MemoryStream(imgbyte));
63.                     imgChoose.Source = ImageSource.FromStream(() => new MemoryStream(imgbyte));
64.                     loading.IsRunning = false;
65.                 }
66.                 catch (Exception e)
67.                 {
68.                     imgChoose.Source = "stock.jpg";
69.                     loading.IsRunning = false;
70.                 }
71.             }
72.             else
73.             {
74.                 imgChoose.Source = App.profile;
75.                 loading.IsRunning = false;
76.             }
77.
78.         }
79.
```

```
        private void Button_Clicked(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new ManageClients());
        }

        private void Button_Clicked_1(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new ManageTrainers());
        }
        private void Button_Clicked_2(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new addexe());
        }

        private async void TapGestureRecognizer_Tapped(object sender, EventArgs e)
        {
            await PopupNavigation.Instance.PushAsync(new ProfilePicOptions());
        }
    }
}

```

## AdminSettings.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="GymAndPAapp.Views.Admin.AdminSettings">
    <ContentPage.Content>
        <StackLayout BackgroundColor="#002d40">
            <Grid VerticalOptions="CenterAndExpand" Margin="60" RowSpacing="30">
                <Grid.RowDefinitions>
                    <RowDefinition/>
                    <RowDefinition/>
                    <RowDefinition/>
                </Grid.RowDefinitions>
                <Button Text="Change Password"  FontSize="15"  HeightRequest="50" Grid.Row="0" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked_1"/>
                <Button Text="Logout" FontSize="15"  HeightRequest="50" Grid.Row="1" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
            </Grid>
        </StackLayout>
    </ContentPage.Content>
</ContentPage>

```

## AdminSettings.xaml.cs

```
using GymAndPAapp.ViewModels;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views.Admin
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class AdminSettings : ContentPage
    {
        IAuthenticationService auth;
        public AdminSettings()
        {
            auth = DependencyService.Get<IAuthenticationService>();
```

```
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
        }

        private async void Button_Clicked_1(object sender, EventArgs e)
        {
            await Navigation.PushAsync(new ChangePassword());
        }

        private async void Button_Clicked(object sender, EventArgs e)
        {
            App._variable = "";
            App._variable2 = "";
            App._variable4 = "";
            App._variable5 = "";
            App._VarArray.Clear();
            var signout = auth.SignOut();
            if (signout)
            {
                await Navigation.PushAsync(new LoginPage());
            }
        }
    }
}
```

## ClientOptions.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<pages:PopupPage x:Class="GymAndPAapp.Views.Admin.ClientOptions"
                 xmlns="http://xamarin.com/schemas/2014/forms"
                 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
                 xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
                 xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup">

    <StackLayout Margin="50,120,50,220"
                 Padding="30">

        <Grid VerticalOptions="Center">
            <Grid.RowDefinitions>
                <RowDefinition Height="120"/>
                <RowDefinition Height="auto"/>
            </Grid.RowDefinitions>
            <StackLayout Grid.Row="1" Grid.ColumnSpan="2" >
                <Button Text="Send Password Recovery" FontSize="15" HeightRequest="50"  CornerRadius="80"
FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked_2"/>
            </StackLayout>
        </Grid>
    </StackLayout>
</pages:PopupPage>
```

## ClientOptions.xaml.cs

```
using GymAndPAapp.ViewModels;
using Rg.Plugins.Popup.Services;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views.Admin
{
```

```
14.     [XamlCompilation(XamlCompilationOptions.Compile)]
15.     public partial class ClientOptions
16.     {
17.         public string Username;
18.         IAuthenticationService auth;
19.         public ClientOptions( string username)
20.         {
21.             Username = username;
22.             auth = DependencyService.Get<IAuthenticationService>();
23.             SetValue(NavigationPage.HasNavigationBarProperty, false);
24.             InitializeComponent();
25.         }
26.
27.         private async void Button_Clicked_2(object sender, EventArgs e)
28.         {
29.             var client = await FireAssistant.CheckUsernameClient(Username);
30.             if (client != null)
31.             {
32.                 await auth.AdminRecoveryAsync(client.Email);
33.             }
34.         }
35.     }
36. }
37.
```

## ManageClients.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.               xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.               x:Class="GymAndPAapp.Views.Admin.ManageClients">
5.      <ContentPage.Content>
6.          <StackLayout BackgroundColor="#002d40" Padding="12">
7.              <Grid>
8.                  <Grid.RowDefinitions>
9.                      <RowDefinition Height="10"/>
10.                     <RowDefinition Height="30"/>
11.                     <RowDefinition Height="20"/>
12.                     <RowDefinition Height="500"/>
13.                     <RowDefinition Height="auto"/>
14.                 </Grid.RowDefinitions>
15.                 <Grid.ColumnDefinitions>
16.                     <ColumnDefinition Width="*"/>
17.                     <ColumnDefinition Width="*"/>
18.                     <ColumnDefinition Width="*"/>
19.                 </Grid.ColumnDefinitions>
20.                 <StackLayout Grid.Row="1" Grid.ColumnSpan="3">
21.                     <Label Text="Clients" TextColor="White"  FontSize="23" FontAttributes="Bold"
    HorizontalOptions="Center"/>
22.                 </StackLayout>
23.                 <StackLayout Grid.Row="2" Grid.ColumnSpan="3">
24.                     <Label Grid.Row="2" TextColor="White" Grid.Column="0" Text="   Username          Trainer
    Plan            " FontSize="18" FontAttributes="Bold"/>
25.                 </StackLayout>
26.                 <Frame BackgroundColor="White" Padding="2" Grid.Row="3" Grid.ColumnSpan="3">
27.                     <ListView x:Name="clientlist" Grid.Row="3" Grid.ColumnSpan="3" BackgroundColor="#002d40"
    SeparatorColor="White">
28.                         <ListView.ItemTemplate>
29.                             <DataTemplate>
30.                                 <ViewCell>
31.                                     <ViewCell.View>
32.                                         <StackLayout >
33.                                             <Grid>
34.                                                 <Grid.ColumnDefinitions>
35.                                                     <ColumnDefinition Width="130"/>
36.                                                     <ColumnDefinition Width="120"/>
37.                                                     <ColumnDefinition Width="5"/>
38.                                                 </Grid.ColumnDefinitions>
39.                                                 <Label HorizontalOptions="Start"
```

```
                                    Grid.Column="0"
                                    VerticalOptions="Center"
                                    VerticalTextAlignment="Center"
                                    TextColor="White"
                                    Text="{Binding UserName}"/>
                                     <Label HorizontalOptions="Start"
                                    Grid.Column="1"
                                    VerticalOptions="Center"
                                    VerticalTextAlignment="Center"
                                    TextColor="White"
                                    Text="{Binding TrainerName}"/>
                                     <Label HorizontalOptions="Start"
                                    Grid.Column="3"
                                    VerticalOptions="Center"
                                    VerticalTextAlignment="Center"
                                    TextColor="White"
                                    Text="{Binding PlanTitle}"/>
                                </Grid>
                            </StackLayout>
                        </ViewCell.View>
                    </ViewCell>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
    </Frame>
    <Button Text="Return" FontSize="15" HeightRequest="50" Grid.Row="4" Grid.ColumnSpan="3"
CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked"/>
    </Grid>
  </StackLayout>
 </ContentPage.Content>
</ContentPage>
```

## ManageClients.xaml.cs

```
using GymAndPAapp.ViewModels;
using Rg.Plugins.Popup.Services;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views.Admin
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class ManageClients : ContentPage
    {
        public ManageClients()
        {
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
        }
        protected async override void OnAppearing()
        {
            base.OnAppearing();
            var allClients = await FireAssistant.GetAllClients();
            clientlist.ItemsSource = allClients;
            clientlist.ItemTapped += Clientlist_ItemTapped;

        }

        private async void Clientlist_ItemTapped(object sender, ItemTappedEventArgs e)
        {
            var details = e.Item as Tables.Clients;
            await PopupNavigation.Instance.PushAsync(new ClientOptions(details.UserName));
```

```
        }

        private void Button_Clicked(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new TabbedAdmin());
        }
    }
}

```

## ManageTrainers.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="GymAndPAapp.Views.Admin.ManageTrainers">
    <ContentPage.Content>
        <StackLayout BackgroundColor="#002d40" Padding="12">
            <Grid>
                <Grid.RowDefinitions>
                    <RowDefinition Height="10"/>
                    <RowDefinition Height="30"/>
                    <RowDefinition Height="20"/>
                    <RowDefinition Height="500"/>
                    <RowDefinition Height="auto"/>
                </Grid.RowDefinitions>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="100"/>
                    <ColumnDefinition Width="250"/>
                </Grid.ColumnDefinitions>
                <StackLayout Grid.Row="1" Grid.ColumnSpan="3">
                    <Label Text="Trainers" TextColor="white"  FontSize="23" HorizontalOptions="Center" FontAttributes="Bold" Padding="20,0,20,0"/>
                </StackLayout>
                <StackLayout Grid.Row="2" Grid.ColumnSpan="3">
                    <Label Grid.Row="2" TextColor="White" Grid.Column="0" Text="    Username            Paypal Address " FontSize="18" FontAttributes="Bold"/>
                </StackLayout>
                <Frame BackgroundColor="White" Padding="2" Grid.Row="3" Grid.ColumnSpan="3">
                    <ListView x:Name="trainerlist" Grid.Row="3" Grid.ColumnSpan="3" SeparatorColor="White" BackgroundColor="#002d40">
                    <ListView.ItemTemplate>
                        <DataTemplate>
                            <ViewCell>
                                <ViewCell.View>
                                    <StackLayout >
                                        <Grid>
                                            <Grid.ColumnDefinitions>
                                                <ColumnDefinition Width="10"/>
                                                <ColumnDefinition Width="140"/>
                                                <ColumnDefinition Width="250"/>
                                            </Grid.ColumnDefinitions>
                                            <Label HorizontalOptions="Start"
                                              Grid.Column="1"
                                              VerticalOptions="Center"
                                              VerticalTextAlignment="Center"
                                              TextColor="White"
                                              Text="{Binding UserName}"/>
                                            <Label HorizontalOptions="Start"
                                              Grid.Column="2"
                                              VerticalOptions="Center"
                                              VerticalTextAlignment="Center"
                                              TextColor="White"
                                              Text="{Binding PaypalAddress}"/>
                                        </Grid>
                                    </StackLayout>
                                </ViewCell.View>
                            </ViewCell>
                        </DataTemplate>
```

```
                </ListView.ItemTemplate>
            </ListView>
            </Frame>
            <Button Text="Return" FontSize="15" HeightRequest="50" Grid.Row="4" Grid.ColumnSpan="3" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked"/>
        </Grid>
    </StackLayout>
  </ContentPage.Content>
</ContentPage>
```

## ManageTrainers.xaml.cs

```
using GymAndPAapp.ViewModels;
using Rg.Plugins.Popup.Services;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views.Admin
{
  [XamlCompilation(XamlCompilationOptions.Compile)]
  public partial class ManageTrainers : ContentPage
  {
    public ManageTrainers()
    {
      SetValue(NavigationPage.HasNavigationBarProperty, false);
      InitializeComponent();
    }
    protected async override void OnAppearing()
    {
      base.OnAppearing();
      var allTrainers = await FireAssistant.GetAllTrainers();
      trainerlist.ItemsSource = allTrainers;
      trainerlist.ItemTapped += Trainerlist_ItemTapped;

    }

    private async void Trainerlist_ItemTapped(object sender, ItemTappedEventArgs e)
    {
      var details = e.Item as Tables.Trainers;
      await PopupNavigation.Instance.PushAsync(new TrainerOptions(details.UserName));
    }

    private void Button_Clicked(object sender, EventArgs e)
    {
      App.Current.MainPage = new NavigationPage(new TabbedAdmin());
    }
  }
}
```

## TabbedAdmin.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<TabbedPage xmlns="http://xamarin.com/schemas/2014/forms"
        xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml" xmlns:views="clr-namespace:GymAndPAapp.Views"
        xmlns:views2="clr-namespace:GymAndPAapp.Views.Admin"
        x:Class="GymAndPAapp.Views.Admin.TabbedAdmin"
        xmlns:android="clr-namespace:Xamarin.Forms.PlatformConfiguration.AndroidSpecific;assembly=Xamarin.Forms.Core" xmlns:views1="clr-namespace:GymAndPAapp.Views.Clients"
```

```
            android:TabbedPage.ToolbarPlacement="Bottom"
            android:TabbedPage.BarItemColor ="White"
            android:TabbedPage.BarSelectedItemColor ="#ff4140">

    <!--Pages can be added as references or inline-->


    <views2:AdminHome Title="Home"></views2:AdminHome>
    <views2:AdminSettings Title="Settings"></views2:AdminSettings>

</TabbedPage>

```

## TabbedAdmin.xaml.cs

```
using Xamarin.Forms;
using Xamarin.Forms.Xaml;
using Xamarin.Forms.PlatformConfiguration.AndroidSpecific;
using TabbedPage = Xamarin.Forms.TabbedPage;
using Xamarin.Forms.PlatformConfiguration;

namespace GymAndPAapp.Views.Admin
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class TabbedAdmin : TabbedPage
    {
        public TabbedAdmin()
        {
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
        }

    }
}

```

## TrainerOptions.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<pages:PopupPage x:Class="GymAndPAapp.Views.Admin.TrainerOptions"
                 xmlns="http://xamarin.com/schemas/2014/forms"
                 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
                 xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
                 xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup">

    <StackLayout Margin="50,120,50,220"
                 Padding="30">

        <Grid VerticalOptions="Center">
            <Grid.RowDefinitions>
                <RowDefinition Height="120"/>
                <RowDefinition Height="auto"/>
            </Grid.RowDefinitions>
            <StackLayout Grid.Row="1" Grid.ColumnSpan="2" >
                <Button Text="Send Password Recovery" FontSize="15" HeightRequest="50"  CornerRadius="80"
FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked_2"/>
            </StackLayout>
        </Grid>
    </StackLayout>
</pages:PopupPage>

```

## TrainerOptions.xaml.cs

```
using GymAndPAapp.ViewModels;
using Rg.Plugins.Popup.Services;
using System;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views.Admin
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class TrainerOptions
    {
        public string Username;
        IAuthenticationService auth;
        public TrainerOptions(string username)
        {
            Username = username;
            auth = DependencyService.Get<IAuthenticationService>();
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
        }
        private async void Button_Clicked_2(object sender, EventArgs e)
        {
            var trainer = await FireAssistant.CheckUsernameTrainer(Username);
            if (trainer != null)
            {
                var update = await auth.AdminRecoveryAsync(trainer.PaypalAddress);
                await App.Current.MainPage.DisplayAlert("", "Reset link sent", "OK");
                await PopupNavigation.Instance.PopAllAsync();
            }
        }
    }
}

```

## *Clients*

## ClientGoals.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:control="clr-namespace:ProgressRingControl.Forms.Plugin;assembly=ProgressRing.Forms.Plugin"
             xmlns:local="clr-namespace:CircularProgressApp" xmlns:local1="clr-namespace:GymAndPAapp.ViewModels"
             x:Class="GymAndPAapp.Views.Clients.ClientGoals"
             BackgroundColor="#002d40">

    <ContentPage.Resources>
        <ResourceDictionary>
            <local:ValueProgressBar x:Key="ValueProgressBar" />
        </ResourceDictionary>
    </ContentPage.Resources>
    <Grid>
    <Grid.RowDefinitions>
        <RowDefinition Height="90"/>
        <RowDefinition Height="40"/>
        <RowDefinition Height="100"/>
        <RowDefinition Height="100"/>
        <RowDefinition Height="30"/>
        <RowDefinition Height="200"/>
        <RowDefinition/>
```

```
23.             <RowDefinition/>
24.         </Grid.RowDefinitions>
25.         <StackLayout Grid.ColumnSpan="2" Grid.Row="0" VerticalOptions="CenterAndExpand">
26.             <Label FontAttributes="Bold" TextColor="White" FontSize="18" VerticalOptions="CenterAndExpand"
    HorizontalOptions="CenterAndExpand" Text="Personal Goals Progress"/>
27.         </StackLayout>
28.         <control:ProgressRing Grid.ColumnSpan="2" Grid.Row="2" RingProgressColor="#ff4140" Scale="2"
    RingThickness="8"  Progress="{Binding ProgressValue , Converter={StaticResource
    ValueProgressBar},Mode=TwoWay}"/>
29.         <StackLayout Grid.ColumnSpan="2" Grid.Row="2" VerticalOptions="CenterAndExpand">
30.             <Label FontAttributes="Bold" FontSize="18" VerticalOptions="CenterAndExpand"
    HorizontalOptions="CenterAndExpand" TextColor="#ff4140" Text="{Binding ProgressValue, StringFormat='{0}'}"/>
31.             <Label FontAttributes="Bold" FontSize="18" VerticalOptions="CenterAndExpand"
    HorizontalOptions="CenterAndExpand" TextColor="#ff4140" Text="%"/>
32.         </StackLayout>
33.         <StackLayout Grid.ColumnSpan="2" Grid.Row="4">
34.             <Label FontAttributes="Bold" FontSize="18" VerticalOptions="Start" TextColor="White" Text="Personal
    Goals"/>
35.         </StackLayout>
36.         <Frame BackgroundColor="White" Padding="2" Grid.Row="5" Grid.ColumnSpan="2">
37.             <ListView x:Name="goal_list" HasUnevenRows="True" BackgroundColor="#002d40" HeightRequest="300"
    SeparatorColor="White">
38.                 <ListView.ItemTemplate>
39.                     <DataTemplate>
40.                         <ViewCell>
41.                             <ViewCell.View>
42.                                 <Grid>
43.                                     <Grid.ColumnDefinitions>
44.                                         <ColumnDefinition/>
45.                                         <ColumnDefinition/>
46.                                     </Grid.ColumnDefinitions>
47.                                     <Label Grid.Column="0" HorizontalOptions="Start"
48.                                            FontSize="18"
49.                                            TextColor="White"
50.                                            FontAttributes="Bold"
51.                                            Text="{Binding Goal}"/>
52.                                     <CheckBox Grid.Column="1" HorizontalOptions="End"
53.                                               Color="#ff4140"
54.                                               IsEnabled="False"
55.                                               CheckedChanged="CheckBox_CheckedChanged"
56.                                               x:Name="testing"
57.                                               IsChecked="{Binding Done}"/>
58.                                 </Grid>
59.                             </ViewCell.View>
60.                         </ViewCell>
61.                     </DataTemplate>
62.                 </ListView.ItemTemplate>
63.             </ListView>
64.         </Frame>
65.         <StackLayout Grid.Row="6" Grid.ColumnSpan="2" Padding="30" >
66.             <Button Text="Add"  FontSize="15" HeightRequest="50" CornerRadius="80" FontAttributes="Bold"
    TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
67.         </StackLayout>
68.     </Grid>
69.
70. </ContentPage>
71.
```

## ClientGoals.xaml.cs

```
1.  using GymAndPAapp.Tables;
2.  using GymAndPAapp.ViewModels;
3.  using Rg.Plugins.Popup.Services;
4.  using System;
5.  using System.Collections.Generic;
6.  using System.Collections.ObjectModel;
7.  using System.Linq;
8.  using System.Text;
9.  using System.Threading.Tasks;
```

```
10. using System.Timers;
11. using Xamarin.Forms;
12. using Xamarin.Forms.Xaml;
13.
14. namespace GymAndPAapp.Views.Clients
15. {
16.     [XamlCompilation(XamlCompilationOptions.Compile)]
17.     public partial class ClientGoals : ContentPage
18.     {
19.         public ObservableCollection<Goals> selectedList;
20.
21.         private double _ProgressValue;
22.         public string percentageString { get; set; }
23.         public double ProgressValue
24.         {
25.             get
26.             {
27.                 return _ProgressValue;
28.             }
29.             set
30.             {
31.                 _ProgressValue = value;
32.                 OnPropertyChanged();
33.             }
34.         }
35.         private double _Maximum;
36.         public double Maximum
37.         {
38.             get
39.             {
40.                 return _ProgressValue;
41.             }
42.             set
43.             {
44.                 _ProgressValue = value;
45.                 OnPropertyChanged();
46.             }
47.         }
48.         private double _countList;
49.         public double countList
50.         {
51.             get
52.             {
53.                 return _countList;
54.             }
55.             set
56.             {
57.                 _countList = value;
58.                 OnPropertyChanged();
59.             }
60.         }
61.         private double _checkedlist;
62.         public double checkedlist
63.         {
64.             get
65.             {
66.                 return _checkedlist;
67.             }
68.             set
69.             {
70.                 _checkedlist = value;
71.                 OnPropertyChanged();
72.             }
73.         }
74.
75.         public object CheckBoxTest { get; private set; }
76.
77.         public ClientGoals()
78.         {
79.             App._goalsVar = false;
80.             SetValue(NavigationPage.HasNavigationBarProperty, false);
```

```
            InitializeComponent();
            BindingContext = this;
            Maximum = 100;
            percentageString = ProgressValue.ToString() + "%";

        }
        protected async override void OnAppearing()
        {
            base.OnAppearing();
            var goals = await FireAssistant.GetGoal(App._variable);
            goal_list.ItemsSource = goals;
            countList = goals.Count;
            goal_list.ItemTapped += goal_list_ItemTapped;
        }

        private void goal_list_ItemTapped(object sender, ItemTappedEventArgs e)
        {
            var details = e.Item as Goals;
            PopupNavigation.Instance.PushAsync(new PopUpCompletedGoals(details.Goal));
            OnPropertyChanged();
        }

        private async void Button_Clicked(object sender, EventArgs e)
        {
            await PopupNavigation.Instance.PushAsync(new PopupAddGoal());
            OnPropertyChanged();

        }

        private async void CheckBox_CheckedChanged(object sender, CheckedChangedEventArgs e)
        {
            var test = e.Value;
            if(test == true)
            {
                checkedlist++;
            }
            else
            {
                checkedlist--;
            }
            if (checkedlist == 0 && countList == 0)
            {
                ProgressValue = 100;
            }
            else
            {
                ProgressValue = (checkedlist /countList ) * 100;
                ProgressValue = Math.Round(ProgressValue,2);
            }
        }
    }
}

```

## ClientHome.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:Image="clr-namespace:ImageCircle.Forms.Plugin.Abstractions;assembly=ImageCircle.Forms.Plugin"
             x:Class="GymAndPAapp.Views.ClientHome">
    <ContentPage.Content>
        <ScrollView BackgroundColor="#002d40">
            <StackLayout>
                <Grid VerticalOptions="Center" Margin="30"  RowSpacing="30">
                    <Grid.RowDefinitions>
                        <RowDefinition Height="0"/>
                        <RowDefinition Height="220"/>
                        <RowDefinition Height="60"/>
```

```
                    <RowDefinition Height="50"/>
                    <RowDefinition Height="50"/>
                    <RowDefinition Height="50"/>
                </Grid.RowDefinitions>
                <StackLayout Grid.Row="1">
                    <Image:CircleImage x:Name="imgChoose" Source="stock.jpg" HeightRequest="300" WidthRequest="300" Aspect="AspectFill">
                        <Image.GestureRecognizers>
                            <TapGestureRecognizer
                                Tapped="TapGestureRecognizer_Tapped"
                                NumberOfTapsRequired="1"/>
                        </Image.GestureRecognizers>
                    </Image:CircleImage>
                    <ActivityIndicator Grid.Row="1" x:Name="loading" IsRunning="False" Color="#ff4140"/>
                </StackLayout>
                <Button Text="Trainers"  FontSize="15" HeightRequest="50" Grid.Row="3" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked"/>
                <Button Text="Plans" x:Name="PlansButton"  FontSize="15" HeightRequest="50" Grid.Row="4" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked_1"/>
                <Button Text="Daily Workout" x:Name="DP"  FontSize="15" HeightRequest="50" Grid.Row="5" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked_2"/>
            </Grid>
        </StackLayout>
    </ScrollView>
  </ContentPage.Content>
</ContentPage>

```

## ClientHome.xaml.cs

```
using Firebase.Storage;
using GymAndPAapp.ViewModels;
using GymAndPAapp.Views.Clients;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.IO;
using System.Text;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Essentials;
using Xamarin.Forms.Xaml;
using GymAndPAapp.Views.Shared;
using Rg.Plugins.Popup.Services;

namespace GymAndPAapp.Views
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class ClientHome : ContentPage
    {
        IAuthenticationService auth;
        public ClientHome()
        {
            auth = DependencyService.Get<IAuthenticationService>();
            App._variable5 = null;
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
        }

        protected async override void OnAppearing()
        {
            var storage = await Permissions.CheckStatusAsync<Permissions.StorageRead>();
            var camera = await Permissions.CheckStatusAsync<Permissions.Camera>();

            if (storage != PermissionStatus.Granted)
            {
                storage = await Permissions.RequestAsync<Permissions.StorageRead>();
```

```
40.             }
41.             if(storage != PermissionStatus.Granted)
42.             {
43.                 return;
44.             }
45.             if(camera != PermissionStatus.Granted)
46.             {
47.                 camera = await Permissions.RequestAsync<Permissions.Camera>();
48.             }
49.             if (camera != PermissionStatus.Granted)
50.             {
51.                 return;
52.             }
53.             var info = auth.GetUser();
54.             var details = await FireAssistant.GetClient(info);
55.             if(details.PlanTitle != null)
56.             {
57.                 DP.IsVisible = true;
58.             }
59.             else
60.             {
61.                 DP.IsVisible = false;
62.             }
63.             if(details.TrainerName != null)
64.             {
65.                 PlansButton.IsVisible = true;
66.             }
67.             else
68.             {
69.                 PlansButton.IsVisible = false;
70.             }
71.             App._variable = details.UserName;
72.             var name = App._variable;
73.             base.OnAppearing();
74.             loading.IsRunning = true;
75.             if(App.profile == null)
76.             {
77.                 try
78.                 {
79.                     var webClient = new WebClient();
80.                     var getimage = await new FirebaseStorage("gymanalyticsapp.appspot.com")
81.                     .Child("ProfilePics")
82.                     .Child(name + ".jpeg")
83.                     .GetDownloadUrlAsync();
84.                     string imgurl = getimage;
85.                     byte[] imgbyte = webClient.DownloadData(imgurl);
86.                     App.profile = ImageSource.FromStream(() => new MemoryStream(imgbyte));
87.                     imgChoose.Source = ImageSource.FromStream(() => new MemoryStream(imgbyte));
88.                     loading.IsRunning = false;
89.                 }
90.                 catch (Exception e)
91.                 {
92.                     imgChoose.Source = "stock.jpg";
93.                     loading.IsRunning = false;
94.                 }
95.             }
96.             else
97.             {
98.                 imgChoose.Source = App.profile;
99.                 loading.IsRunning = false;
100.            }
101.
102.        }
103.
104.        private void Button_Clicked(object sender, EventArgs e)
105.        {
106.            App.Current.MainPage = new NavigationPage(new TrainerList());
107.        }
108.
109.        private void Button_Clicked_1(object sender, EventArgs e)
110.        {
```

```
111.            App.Current.MainPage = new NavigationPage(new PlanList());
112.        }
113.
114.        private void Button_Clicked_2(object sender, EventArgs e)
115.        {
116.            App.Current.MainPage = new NavigationPage(new DailyWorkout());
117.        }
118.
119.        private async void TapGestureRecognizer_Tapped(object sender, EventArgs e)
120.        {
121.            await PopupNavigation.Instance.PushAsync(new ProfilePicOptions());
122.        }
123.    }
124. }
125.
```

## ClientPayment.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.              xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.              x:Class="GymAndPAapp.Views.Clients.ClientPayment"
5.              BackgroundColor="#2a2a2a">
6.      <ContentPage.Content>
7.          <ScrollView>
8.              <StackLayout>
9.                  <Grid Margin="50" RowSpacing="10">
10.                     <Grid.RowDefinitions>
11.                         <RowDefinition Height="80"/>
12.                         <RowDefinition Height="30"/>
13.                         <RowDefinition/>
14.                         <RowDefinition/>
15.                         <RowDefinition/>
16.                         <RowDefinition Height="60"/>
17.                         <RowDefinition/>
18.                         <RowDefinition/>
19.                         <RowDefinition Height="80"/>
20.                     </Grid.RowDefinitions>
21.                     <Grid.ColumnDefinitions>
22.                         <ColumnDefinition/>
23.                         <ColumnDefinition/>
24.                     </Grid.ColumnDefinitions>
25.                     <Label Text="Card Details" Grid.Row="1" Grid.ColumnSpan="2" FontSize="23" TextColor="White"
    FontAttributes="Bold" />
26.                     <StackLayout Orientation="Horizontal" HorizontalOptions="FillAndExpand" Grid.Row="2"
    Grid.ColumnSpan="2">
27.                         <Entry  Placeholder="4242 4242 4242 4242" x:Name="Number" MaxLength="16" TextColor="White"
    Keyboard="Numeric" WidthRequest="230"/>
28.                     </StackLayout>
29.                     <StackLayout Orientation="Horizontal" HorizontalOptions="FillAndExpand" Grid.Row="3"
    Grid.ColumnSpan="2">
30.                         <Entry  Grid.Column="0" Placeholder="MM" Keyboard="Numeric" x:Name="Month" MaxLength="2"
    TextColor="White" />
31.                         <Entry  Grid.Column="1" Placeholder="YY" Keyboard="Numeric" x:Name="Year" MaxLength="2"
    TextColor="White"/>
32.                     </StackLayout>
33.                     <StackLayout Orientation="Horizontal" HorizontalOptions="FillAndExpand" Grid.Row="4"
    Grid.ColumnSpan="2">
34.                         <Entry Grid.Column="0" Placeholder="CVC"  Keyboard="Numeric" MaxLength="4"
    x:Name="VerificationCode" TextColor="White"/>
35.                     </StackLayout>
36.                     <StackLayout Grid.Row="5" Grid.ColumnSpan="2">
37.                         <Image Source="cardProviders.png" HeightRequest="70"/>
38.                     </StackLayout>
39.                     <StackLayout Grid.Row="6" Grid.Column="0">
40.                         <Button Text="Pay Now"  FontSize="15" HeightRequest="50" CornerRadius="80"
    FontAttributes="Bold" TextColor="White" Background="green" Clicked="checkdetails"/>
41.                     </StackLayout>
42.                     <StackLayout Grid.Row="6" Grid.Column="1">
```

```
43.                     <Button Text="Cancel" FontSize="15" HeightRequest="50" CornerRadius="80" FontAttributes="Bold"
    TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
44.                 </StackLayout>
45.             </Grid>
46.         </StackLayout>
47.     </ScrollView>
48.   </ContentPage.Content>
49. </ContentPage>
50.
```

## ClientPayment.xaml.cs

```
1.  using GymAndPAapp.Data;
2.  using GymAndPAapp.ViewModels;
3.  using Stripe;
4.  using System;
5.  using System.Collections.Generic;
6.  using System.Linq;
7.  using System.Text;
8.  using System.Threading.Tasks;
9.
10. using Xamarin.Forms;
11. using Xamarin.Forms.Xaml;
12.
13. namespace GymAndPAapp.Views.Clients
14. {
15.     [XamlCompilation(XamlCompilationOptions.Compile)]
16.     public partial class ClientPayment : ContentPage
17.     {
18.         IAuthenticationService auth;
19.         public string TestApiKey =
    "sk_test_51IkJqaLuQaYsYnIw9XVrQ0MhFZjSFGf6wOeMGUxbAv4AS4cghiJ2QUkusVX2rCpX1rUpPg7r1XQgpwM
    AcYxx1s9o00cXwxPROl";
20.         public string name;
21.         public string email;
22.         public int AmountPaid;
23.         public string Title;
24.         public string TrainerEmail;
25.
26.         public ClientPayment(int amount, string title, string trainer)
27.         {
28.             Title = title;
29.             AmountPaid = amount;
30.             TrainerEmail = trainer;
31.             SetValue(NavigationPage.HasNavigationBarProperty, false);
32.             InitializeComponent();
33.             auth = DependencyService.Get<IAuthenticationService>();
34.             getdetails();
35.         }
36.
37.         private async void getdetails()
38.         {
39.             var token = auth.GetUser();
40.             string user = auth.GetEmail();
41.             var details = await FireAssistant.GetClient(token);
42.             name = details.UserName;
43.             email = user;
44.         }
45.
46.         string customer;
47.         string chargedID;
48.         string refundID;
49.
50.         public void checkdetails(object sender, EventArgs e)
51.         {
52.             if (Number.Text.Length < 16)
53.             {
54.                 App.Current.MainPage.DisplayAlert("Error", "Card number is to short, please try again", "OK");
55.                 App.Current.MainPage = new NavigationPage(new ClientPayment(AmountPaid, Title, TrainerEmail));
```

```
56.             }
57.             else if (Number.Text == null)
58.             {
59.                 Error();
60.             }
61.             else if (Year.Text == null)
62.             {
63.                 Error();
64.             }
65.             else if (Month.Text == null)
66.             {
67.                 Error();
68.             }
69.             else if (VerificationCode.Text == null)
70.             {
71.                 Error();
72.             }
73.             else
74.             {
75.                 NewEventHandler(sender,e);
76.             }
77.
78.         }
79.         public void NewEventHandler(object sender, EventArgs e)
80.         {
81.
82.             StripeConfiguration.SetApiKey(TestApiKey);
83.             TokenCardOptions stripcard = new TokenCardOptions();
84.             stripcard.Number = Number.Text;
85.             stripcard.ExpYear = long.Parse(Year.Text);
86.             stripcard.ExpMonth = long.Parse(Month.Text);
87.             stripcard.Cvc = VerificationCode.Text;
88.
89.             TokenCreateOptions token = new TokenCreateOptions();
90.             token.Card = stripcard;
91.             TokenService serviceToken = new TokenService();
92.             Token newToken = serviceToken.Create(token);
93.
94.             var options = new SourceCreateOptions
95.             {
96.                 Type = SourceType.Card,
97.                 Currency = "EUR",
98.                 Token = newToken.Id
99.             };
100.
101.            var service = new SourceService();
102.            Source source = service.Create(options);
103.
104.            CustomerCreateOptions customer = new CustomerCreateOptions()
105.            {
106.                Name = name,
107.                Email = email,
108.                Description = "Payment for Personal Trainer" + TrainerEmail,
109.            };
110.            var customerService = new CustomerService();
111.            Customer stripeCustomer = customerService.Create(customer);
112.
113.            // customer = stripeCustomer.Id;
114.
115.            var chargeoptions = new ChargeCreateOptions
116.            {
117.                Amount = AmountPaid *100,
118.                Currency = "EUR",
119.                ReceiptEmail = email,
120.                Customer = stripeCustomer.Id,
121.                Source = source.Id
122.            };
123.
124.            var service1 = new ChargeService();
125.            Charge charge = service1.Create(chargeoptions); //this makes payment
126.
```

```
            chargedID = charge.Id;
            PayTrainer();

        }

        public async void PayTrainer()
        {
            var response = await PayoutPaypal.CreatePayout(TrainerEmail,AmountPaid.ToString(),Title,name);
            if(response != null)
            {
                updateDB();
            }
        }
        public async void updateDB()
        {
            await App.Current.MainPage.DisplayAlert("Success", "Payment has been processed", "OK");
            App._variable7 = 0;
            App._variable6 = 0;
            App._VarArray.Clear();
            var token = auth.GetUser();
            var update = await FireAssistant.UpdateClient3(token, Title);
            var getdetails = await FireAssistant.GetClient(token);
            var updatetrainer = await FireAssistant.AddPayment(App._variable, AmountPaid.ToString(),
    DateTime.Now.Date.ToString("dd/MM/yyyy"), getdetails.TrainerName, Title);
            App.Current.MainPage = new NavigationPage(new TabbedClient());
        }

        private void Button_Clicked(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new TabbedClient());
        }
        public async void Error()
        {
            App.Current.MainPage.DisplayAlert("Error", "All fields are required to process payments", "OK");
            App.Current.MainPage = new NavigationPage(new ClientPayment(AmountPaid,Title,TrainerEmail));
        }
    }
}

```

## ClientSelectedPlan.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<pages:PopupPage xmlns="http://xamarin.com/schemas/2014/forms"
                 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
                 xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
                 xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup"
                 x:Class="GymAndPAapp.Views.ClientSelectedPlan">
    <StackLayout Margin="15,10,15,80"
                 Padding="24"
                 BackgroundColor="#2a2a2a">
        <Grid>
            <Grid.RowDefinitions>
                <RowDefinition Height="auto"/>
                <RowDefinition Height="auto"/>
                <RowDefinition Height="90"/>
                <RowDefinition Height="auto"/>
                <RowDefinition Height="auto"/>
                <RowDefinition Height="140"/>
                <RowDefinition Height="auto"/>
                <RowDefinition Height="140"/>
                <RowDefinition Height="auto"/>
            </Grid.RowDefinitions>
            <StackLayout Grid.Row="0">
                <Label x:Name="title" HorizontalOptions="CenterAndExpand" FontSize="23" FontAttributes="Bold"
    TextColor="white"/>
            </StackLayout>
            <StackLayout Grid.Row="1">
```

```
            <Label Text="Description:" TextColor="White" HorizontalOptions="Start" FontSize="18"
FontAttributes="Bold"/>
        </StackLayout>
        <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="2">
            <StackLayout Grid.Row="2" BackgroundColor="#2a2a2a">
                <Label x:Name="description" TextColor="White"  HorizontalOptions="Start"/>
            </StackLayout>
        </Frame>
        <StackLayout Grid.Row="3">
            <Label Text="Preview of plan" TextColor="White" HorizontalOptions="Start" FontSize="18"
FontAttributes="Bold"/>
        </StackLayout>
        <StackLayout Grid.Row="4">
            <Label Text="Monday" TextColor="White" HorizontalOptions="Start" FontSize="16"/>
        </StackLayout>
        <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="5">
            <StackLayout Grid.Row="5">
                <ListView x:Name="monlist" BackgroundColor="#2a2a2a" SeparatorColor="White">
                    <ListView.ItemTemplate>
                        <DataTemplate>
                            <ViewCell IsEnabled="False">
                                <ViewCell.View>
                                    <StackLayout>
                                        <Label HorizontalOptions="Start"
                                            VerticalOptions="Center"
                                            VerticalTextAlignment="Center"
                                            TextColor="White"
                                            Text="{Binding}"/>
                                    </StackLayout>
                                </ViewCell.View>
                            </ViewCell>
                        </DataTemplate>
                    </ListView.ItemTemplate>
                </ListView>
            </StackLayout>
        </Frame>
        <StackLayout Grid.Row="6">
            <Label Text="Tuesday" TextColor="white" HorizontalOptions="Start" FontSize="16"/>
        </StackLayout>
        <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="7">
            <StackLayout Grid.Row="7">
                <ListView x:Name="tuelist" BackgroundColor="#2a2a2a" SeparatorColor="White">
                    <ListView.ItemTemplate>
                        <DataTemplate>
                            <ViewCell IsEnabled="False">
                                <ViewCell.View>
                                    <StackLayout>
                                        <Label HorizontalOptions="Start"
                                            VerticalOptions="Center"
                                            VerticalTextAlignment="Center"
                                            TextColor="White"
                                            Text="{Binding}"/>
                                    </StackLayout>
                                </ViewCell.View>
                            </ViewCell>
                        </DataTemplate>
                    </ListView.ItemTemplate>
                </ListView>
            </StackLayout>
        </Frame>
        <StackLayout Grid.Row="8">
            <Label x:Name="fee" TextColor="White" HorizontalOptions="CenterAndExpand" FontSize="18"
FontAttributes="Bold"/>
        </StackLayout>
        <StackLayout Grid.Row="9">
            <Button Text="Purchase Plan"  FontSize="15"  HeightRequest="50" CornerRadius="80"
FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
        </StackLayout>
    </Grid>
  </StackLayout>
</pages:PopupPage>
```

```
93.
```

## ClientSelectedPlan.xaml.cs

```
1.  using System;
2.  using System.Collections.Generic;
3.  using System.Linq;
4.  using System.Text;
5.  using System.Threading.Tasks;
6.  using GymAndPAapp.ViewModels;
7.  using GymAndPAapp.Views.Clients;
8.  using Rg.Plugins.Popup.Services;
9.  using Xamarin.Forms;
10. using Xamarin.Forms.Xaml;
11.
12. namespace GymAndPAapp.Views
13. {
14.     [XamlCompilation(XamlCompilationOptions.Compile)]
15.     public partial class ClientSelectedPlan
16.     {
17.         public int cost;
18.         IAuthenticationService auth;
19.         public ClientSelectedPlan(string Title,string Monday,string Tuesday, string Description, string Fee)
20.         {
21.             cost = int.Parse(Fee);
22.             auth = DependencyService.Get<IAuthenticationService>();
23.             SetValue(NavigationPage.HasNavigationBarProperty, false);
24.             InitializeComponent();
25.             title.Text = Title;
26.             description.Text = Description;
27.             List<string> MondayArray = new List<string>();
28.             if (Monday.Contains(",") == true)
29.             {
30.                 MondayArray = Monday.Split(new char[] { ',' }).ToList();
31.                 monlist.ItemsSource = MondayArray;
32.             }
33.             List<string> TuesdayArray = new List<string>();
34.             if (Tuesday.Contains(",") == true)
35.             {
36.                 TuesdayArray = Tuesday.Split(new char[] { ',' }).ToList();
37.                 tuelist.ItemsSource = TuesdayArray;
38.             }
39.             fee.Text = "€ " + Fee;
40.         }
41.
42.         private async void Button_Clicked(object sender, EventArgs e)
43.         {
44.             var token = auth.GetUser();
45.             var details = await FireAssistant.GetClient(token);
46.             if(details.PlanTitle != null)
47.             {
48.                 bool answer = await DisplayAlert("Question?", "You currently have a plan purchased are you sure youd like to
    continue previous plan will be overwritten?", "Yes", "No");
49.                 if(answer == true)
50.                 {
51.                     var trainer = await FireAssistant.CheckUsernameTrainer(details.TrainerName);
52.                     App._variable7 = 0;
53.                     App._variable6 = 0;
54.                     await PopupNavigation.Instance.PopAsync();
55.                     App.Current.MainPage = new NavigationPage(new ClientPayment(cost, title.Text, trainer.PaypalAddress));
56.                 }
57.                 else
58.                 {
59.                     await PopupNavigation.Instance.PopAsync();
60.                     App.Current.MainPage = new NavigationPage(new PlanList());
61.                 }
62.             }
```

```
63.             else
64.             {
65.                 var trainer = await FireAssistant.CheckUsernameTrainer(details.TrainerName);
66.                 App._variable7 = 0;
67.                 App._variable6 = 0;
68.                 await PopupNavigation.Instance.PopAsync();
69.                 App.Current.MainPage = new NavigationPage(new ClientPayment(cost, title.Text, trainer.PaypalAddress));
70.             }
71.         }
72.     }
73. }
74.
```

## DailyWorkout.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.               xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.               xmlns:control="clr-namespace:ProgressRingControl.Forms.Plugin;assembly=ProgressRing.Forms.Plugin"
5.               xmlns:local="clr-namespace:CircularProgressApp" xmlns:local1="clr-namespace:GymAndPAapp.ViewModels"
6.               x:Class="GymAndPAapp.Views.Clients.DailyWorkout"
7.               BackgroundColor="#002d40">
8.
9.     <ContentPage.Resources>
10.        <ResourceDictionary>
11.            <local:ValueProgressBar x:Key="ValueProgressBar" />
12.        </ResourceDictionary>
13.    </ContentPage.Resources>
14.    <Grid>
15.        <Grid.RowDefinitions>
16.            <RowDefinition Height="90"/>
17.            <RowDefinition Height="40"/>
18.            <RowDefinition Height="100"/>
19.            <RowDefinition Height="100"/>
20.            <RowDefinition Height="30"/>
21.            <RowDefinition Height="30"/>
22.            <RowDefinition Height="200"/>
23.            <RowDefinition/>
24.            <RowDefinition/>
25.        </Grid.RowDefinitions>
26.        <Grid.ColumnDefinitions>
27.            <ColumnDefinition Width="*"/>
28.            <ColumnDefinition Width="*"/>
29.            <ColumnDefinition Width="*"/>
30.        </Grid.ColumnDefinitions>
31.        <StackLayout Grid.ColumnSpan="3" Grid.Row="0" VerticalOptions="CenterAndExpand">
32.            <Label FontAttributes="Bold" FontSize="23" VerticalOptions="CenterAndExpand"
    HorizontalOptions="CenterAndExpand" TextColor="white" Text="Daily Workout Plan"/>
33.        </StackLayout>
34.        <control:ProgressRing Grid.ColumnSpan="3" Grid.Row="2" RingProgressColor="#ff4140" Scale="2"
    RingThickness="8"  Progress="{Binding ProgressValue , Converter={StaticResource
    ValueProgressBar},Mode=TwoWay}"/>
35.        <StackLayout Grid.ColumnSpan="3" Grid.Row="2" VerticalOptions="CenterAndExpand">
36.            <Label FontAttributes="Bold" FontSize="18" VerticalOptions="CenterAndExpand"
    HorizontalOptions="CenterAndExpand" TextColor="#ff4140" Text="{Binding ProgressValue, StringFormat='{0}'}"/>
37.            <Label FontAttributes="Bold" FontSize="18" VerticalOptions="CenterAndExpand"
    HorizontalOptions="CenterAndExpand" TextColor="#ff4140" Text="%"/>
38.        </StackLayout>
39.        <StackLayout Grid.ColumnSpan="3" Grid.Row="4">
40.            <Label FontAttributes="Bold" FontSize="18" VerticalOptions="Start" TextColor="White" Text="Workout
    Plan"/>
41.        </StackLayout>
42.        <StackLayout Grid.ColumnSpan="3" Grid.Row="6"  IsVisible="{Binding restday2}">
43.            <Label FontAttributes="Bold" FontSize="18" VerticalTextAlignment="Center" VerticalOptions="Center"
    TextColor="White" Text="This is a Rest Day"/>
44.        </StackLayout>
45.        <StackLayout Grid.ColumnSpan="3" Grid.Row="6"  IsVisible="{Binding completedDay}">
46.            <Label FontAttributes="Bold" FontSize="18" HorizontalOptions="Center" HorizontalTextAlignment="Center"
    TextColor="White" Text="Todays Workout has been completed"/>
```

```
47.         </StackLayout>
48.         <Label Grid.Row="5" Grid.ColumnSpan="2" IsVisible="{Binding restday}">
49.             <Label.FormattedText>
50.                 <FormattedString>
51.                     <Span Text="Click here for Tutorial" FontAttributes="Bold" TextColor="#ff4140"
    TextDecorations="Underline">
52.                         <Span.GestureRecognizers>
53.                             <TapGestureRecognizer Tapped="TapGestureRecognizer_Tapped"/>
54.                         </Span.GestureRecognizers>
55.                     </Span>
56.                 </FormattedString>
57.             </Label.FormattedText>
58.         </Label>
59.         <Frame BackgroundColor="White" Padding="2" Grid.Row="6" Grid.ColumnSpan="3" IsVisible="{Binding
    restday}">
60.             <ListView x:Name="workout_list" ItemTapped="workout_list_ItemTapped" HasUnevenRows="True"
    BackgroundColor="#002d40" HeightRequest="300">
61.                 <ListView.ItemTemplate>
62.                     <DataTemplate>
63.                         <ViewCell>
64.                             <ViewCell.View>
65.                                 <Grid>
66.                                     <Grid.ColumnDefinitions>
67.                                         <ColumnDefinition Width="10"/>
68.                                         <ColumnDefinition Width="20"/>
69.                                         <ColumnDefinition Width="*"/>
70.                                     </Grid.ColumnDefinitions>
71.                                     <Label Grid.Column="1" HorizontalOptions="Start"
72.                                            FontSize="18"
73.                                            FontAttributes="Bold"
74.                                            TextColor="White"
75.                                            Text="-"/>
76.                                     <Label Grid.Column="2" HorizontalOptions="Start"
77.                                            FontSize="18"
78.                                            FontAttributes="Bold"
79.                                            TextColor="White"
80.                                            Text="{Binding }"/>
81.                                 </Grid>
82.                             </ViewCell.View>
83.                         </ViewCell>
84.                     </DataTemplate>
85.                 </ListView.ItemTemplate>
86.             </ListView>
87.         </Frame>
88.         <StackLayout Grid.Row="7" Grid.ColumnSpan="3" Padding="30" >
89.             <Button Text="Return"  FontSize="15" HeightRequest="50" CornerRadius="80" FontAttributes="Bold"
    TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
90.         </StackLayout>
91.     </Grid>
92.
93. </ContentPage>
94.
```

## DailyWorkout.xaml.cs

```
1.  using GymAndPAapp.Tables;
2.  using GymAndPAapp.ViewModels;
3.  using Rg.Plugins.Popup.Services;
4.  using System;
5.  using System.Collections.Generic;
6.  using System.Linq;
7.  using System.Text;
8.  using System.Threading.Tasks;
9.  using Xamarin.Essentials;
10. using Xamarin.Forms;
11. using Xamarin.Forms.Xaml;
12.
13. namespace GymAndPAapp.Views.Clients
14. {
```

```
15.     [XamlCompilation(XamlCompilationOptions.Compile)]
16.     public partial class DailyWorkout : ContentPage
17.     {
18.         IAuthenticationService auth;
19.         private bool _completedDay;
20.         public bool completedDay
21.         {
22.             get
23.             {
24.                 return _completedDay;
25.             }
26.             set
27.             {
28.                 _completedDay = value;
29.                 OnPropertyChanged();
30.             }
31.         }
32.         private bool _restday;
33.         public bool restday
34.         {
35.             get
36.             {
37.                 return _restday;
38.             }
39.             set
40.             {
41.                 _restday = value;
42.                 OnPropertyChanged();
43.             }
44.         }
45.         private bool _restday2;
46.         public bool restday2
47.         {
48.             get
49.             {
50.                 return _restday2;
51.             }
52.             set
53.             {
54.                 _restday2 = value;
55.                 OnPropertyChanged();
56.             }
57.         }
58.         private double _ProgressValue;
59.         public double ProgressValue
60.         {
61.             get
62.             {
63.                 return _ProgressValue;
64.             }
65.             set
66.             {
67.                 _ProgressValue = value;
68.                 OnPropertyChanged();
69.             }
70.         }
71.         private double _Maximum;
72.         public double Maximum
73.         {
74.             get
75.             {
76.                 return _ProgressValue;
77.             }
78.             set
79.             {
80.                 _ProgressValue = value;
81.                 OnPropertyChanged();
82.             }
83.         }
84.         public DailyWorkout()
85.         {
```

```
86.             auth = DependencyService.Get<IAuthenticationService>();
87.             SetValue(NavigationPage.HasNavigationBarProperty, false);
88.             InitializeComponent();
89.             BindingContext = this;
90.             Maximum = 100;
91.             App.DayOfWeek =  DateTime.Now;
92.             if (App._variable7 == 0 && App._variable6 !=0)
93.             {
94.                 completedDay = true;
95.                 ProgressValue = 100;
96.                 App._variable8 = App.DayOfWeek.DayOfWeek.ToString();
97.             }
98.             else
99.             {
100.                completedDay = false;
101.                if (App._variable6 / App._variable7 > 0)
102.                {
103.                    ProgressValue = Math.Round((App._variable6 / App._variable7) * 100);
104.                }
105.                else
106.                {
107.                    ProgressValue = 0;
108.                }
109.
110.            }
111.        }
112.        protected async override void OnAppearing()
113.        {
114.            base.OnAppearing();
115.            var token = auth.GetUser();
116.            var userdata = await FireAssistant.GetClient(token);
117.            var workoutinfo = await FireAssistant.GetPlan2(userdata.TrainerName, userdata.PlanTitle);
118.
119.            if (completedDay == false)
120.            {
121.                if (App._VarArray.Count == 0)
122.                {
123.                    if (App.DayOfWeek.DayOfWeek.ToString() == "Monday")
124.                    {
125.                        if (workoutinfo.Monday.Contains(",") == true)
126.                        {
127.                            App._VarArray = workoutinfo.Monday.Split(new char[] { ',' }).ToList();
128.                            App._VarArray.RemoveAt(App._VarArray.Count - 1);
129.                            workout_list.ItemsSource = App._VarArray;
130.                            App._variable7 = App._VarArray.Count;
131.                        }
132.                    }
133.                    if (App.DayOfWeek.DayOfWeek.ToString() == "Tuesday")
134.                    {
135.                        if (workoutinfo.Tuesday.Contains(",") == true)
136.                        {
137.                            App._VarArray = workoutinfo.Tuesday.Split(new char[] { ',' }).ToList();
138.                            App._VarArray.RemoveAt(App._VarArray.Count - 1);
139.                            workout_list.ItemsSource = App._VarArray;
140.                            App._variable7 = App._VarArray.Count;
141.                        }
142.                    }
143.                    if (App.DayOfWeek.DayOfWeek.ToString() == "Wednesday")
144.                    {
145.                        if (workoutinfo.Wednesday.Contains(",") == true)
146.                        {
147.                            App._VarArray = workoutinfo.Wednesday.Split(new char[] { ',' }).ToList();
148.                            App._VarArray.RemoveAt(App._VarArray.Count - 1);
149.                            workout_list.ItemsSource = App._VarArray;
150.                            App._variable7 = App._VarArray.Count;
151.                        }
152.                    }
153.                    if (App.DayOfWeek.DayOfWeek.ToString() == "Thursday")
154.                    {
155.                        if (workoutinfo.Thursday.Contains(",") == true)
156.                        {
```

```
157.                    App._VarArray = workoutinfo.Thursday.Split(new char[] { ',' }).ToList();
158.                    App._VarArray.RemoveAt(App._VarArray.Count - 1);
159.                    workout_list.ItemsSource = App._VarArray;
160.                    App._variable7 = App._VarArray.Count;
161.                }
162.            }
163.            if (App.DayOfWeek.DayOfWeek.ToString() == "Friday")
164.            {
165.                if (workoutinfo.Friday.Contains(",") == true)
166.                {
167.                    App._VarArray = workoutinfo.Friday.Split(new char[] { ',' }).ToList();
168.                    App._VarArray.RemoveAt(App._VarArray.Count - 1);
169.                    workout_list.ItemsSource = App._VarArray;
170.                    App._variable7 = App._VarArray.Count;
171.                }
172.            }
173.            if (App.DayOfWeek.DayOfWeek.ToString() == "Saturday")
174.            {
175.                if (workoutinfo.Saturday.Contains(",") == true)
176.                {
177.                    App._VarArray = workoutinfo.Saturday.Split(new char[] { ',' }).ToList();
178.                    App._VarArray.RemoveAt(App._VarArray.Count - 1);
179.                    workout_list.ItemsSource = App._VarArray;
180.                    App._variable7 = App._VarArray.Count;
181.                }
182.            }
183.            if (App.DayOfWeek.DayOfWeek.ToString() == "Sunday")
184.            {
185.                if (workoutinfo.Sunday.Contains(",") == true)
186.                {
187.                    App._VarArray = workoutinfo.Sunday.Split(new char[] { ',' }).ToList();
188.                    App._VarArray.RemoveAt(App._VarArray.Count - 1);
189.                    workout_list.ItemsSource = App._VarArray;
190.                    App._variable7 = App._VarArray.Count;
191.                }
192.            }
193.            if (App._VarArray[0] == "Rest")
194.            {
195.                restday = false;
196.                restday2 = true;
197.                ProgressValue = 100;
198.            }
199.            else
200.            {
201.                restday = true;
202.                restday2 = false;
203.            }
204.        }
205.        else
206.        {
207.            if (App._VarArray[0] == "Rest")
208.            {
209.                restday = false;
210.                restday2 = true;
211.                ProgressValue = 100;
212.            }
213.            else
214.            {
215.                restday = true;
216.                restday2 = false;
217.            }
218.            workout_list.ItemsSource = App._VarArray;
219.        }
220.    }
221. }
222.
223. private void Button_Clicked(object sender, EventArgs e)
224. {
225.     App.Current.MainPage.Navigation.PushAsync(new TabbedClient());
226. }
227.
```

```
228.        private void workout_list_ItemTapped(object sender, ItemTappedEventArgs e)
229.        {
230.            string details = e.Item.ToString();
231.            var details2 = App._VarArray;
232.            List<string> temp = new List<string>();
233.            for (int i = 0; i < App._VarArray.Count(); i++)
234.            {
235.                if (details2.ElementAt(i) == details)
236.                {
237.                    details = "empty";
238.                }
239.                else
240.                {
241.                    temp.Add(details2.ElementAt(i));
242.                }
243.            }
244.            App._VarArray = temp;
245.            if(App._VarArray.Count == 0)
246.            {
247.                App._variable7 = 0;
248.                App._variable6 = App._variable6 + 1;
249.            }
250.            else
251.            {
252.                App._variable6 = App._variable6 + 1;
253.            }
254.            App.Current.MainPage.Navigation.PushAsync(new DailyWorkout());
255.        }
256.
257.        private async void TapGestureRecognizer_Tapped(object sender, EventArgs e)
258.        {
259.            var token = auth.GetUser();
260.            var userdata = await FireAssistant.GetClient(token);
261.            var workoutinfo = await FireAssistant.GetPlan2(userdata.TrainerName, userdata.PlanTitle);
262.            Uri uri = new Uri(workoutinfo.Link);
263.            await Launcher.OpenAsync(uri);
264.
265.        }
266.    }
267. }
268.
```

## PlanList.xaml

```
1.    <?xml version="1.0" encoding="utf-8" ?>
2.    <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.                 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.                 x:Class="GymAndPAapp.Views.PlanList">
5.
6.        <ContentPage.Content>
7.            <StackLayout BackgroundColor="#002d40">
8.                <Grid>
9.                <Grid.RowDefinitions>
10.                   <RowDefinition Height="10"/>
11.                   <RowDefinition Height="30"/>
12.                   <RowDefinition Height="20"/>
13.                   <RowDefinition Height="550"/>
14.                   <RowDefinition Height="auto"/>
15.                   <RowDefinition Height="auto"/>
16.                   </Grid.RowDefinitions>
17.               <Grid.ColumnDefinitions>
18.                   <ColumnDefinition Width="*"/>
19.                   <ColumnDefinition Width="*"/>
20.               </Grid.ColumnDefinitions>
21.               <StackLayout Grid.Row="1" Grid.ColumnSpan="2">
22.                   <Label Text="Available Plans" FontSize="23" TextDecorations="Underline" TextColor="White"
      FontAttributes="Bold" HorizontalOptions="Center"/>
23.               </StackLayout>
24.               <StackLayout Grid.Row="2" Grid.ColumnSpan="2">
```

```
25.                 <Label Grid.Row="2" TextColor="White" Grid.Column="0" Text="   Plan Title          Description
    " FontSize="18" FontAttributes="Bold"/>
26.             </StackLayout>
27.                 <Frame BackgroundColor="White" Padding="2" Grid.Row="3" Grid.ColumnSpan="2">
28.                     <ListView x:Name="planlist" Grid.Row="3" Grid.ColumnSpan="2" SeparatorColor="White"
    BackgroundColor="#002d40">
29.                         <ListView.ItemTemplate>
30.                             <DataTemplate>
31.                                 <ViewCell>
32.                                     <ViewCell.View>
33.                                         <StackLayout >
34.                                             <Grid>
35.                                                 <Grid.ColumnDefinitions>
36.                                                     <ColumnDefinition Width="10"/>
37.                                                     <ColumnDefinition Width="130"/>
38.                                                     <ColumnDefinition Width="250"/>
39.                                                 </Grid.ColumnDefinitions>
40.                                                 <Label HorizontalOptions="Start"
41.                                                 Grid.Column="1"
42.                                                 VerticalOptions="Center"
43.                                                 VerticalTextAlignment="Center"
44.                                                 FontAttributes="Bold"
45.                                                 TextColor="White"
46.                                                 Text="{Binding PlanTitle}"/>
47.                                                  <Label HorizontalOptions="Start"
48.                                                 Grid.Column="2"
49.                                                 VerticalOptions="Center"
50.                                                 VerticalTextAlignment="Center"
51.                                                 FontAttributes="Bold"
52.                                                 TextColor="White"
53.                                                 Text="{Binding Description}"/>
54.                                             </Grid>
55.                                         </StackLayout>
56.                                     </ViewCell.View>
57.                                 </ViewCell>
58.                             </DataTemplate>
59.                         </ListView.ItemTemplate>
60.                     </ListView>
61.                 </Frame>
62.                 <Button Text="Request Custom Plan"  FontSize="15" Grid.Row="4" Grid.ColumnSpan="2"
    HeightRequest="50" TextColor="White" FontAttributes="Bold" BackgroundColor="#002d40" BorderWidth="2"
    BorderColor="White" Clicked="Button_Clicked_1"/>
63.                 <Button Text="Return"  FontSize="15" HeightRequest="50" Grid.Row="5" Grid.ColumnSpan="2"
    CornerRadius="80" FontAttributes="Bold" TextColor="white" Background="#ff4140" Clicked="Button_Clicked"/>
64.             </Grid>
65.         </StackLayout>
66.     </ContentPage.Content>
67. </ContentPage>
68.
```

## PlanList.xaml.cs

```
1.  using System;
2.  using System.Collections.Generic;
3.  using System.Linq;
4.  using System.Text;
5.  using System.Threading.Tasks;
6.
7.  using Xamarin.Forms;
8.  using Xamarin.Forms.Xaml;
9.  using GymAndPAapp.Tables;
10. using GymAndPAapp.ViewModels;
11. using Rg.Plugins.Popup.Services;
12. using GymAndPAapp.Views.Clients;
13.
14. namespace GymAndPAapp.Views
15. {
16.     [XamlCompilation(XamlCompilationOptions.Compile)]
17.     public partial class PlanList : ContentPage
```

```
    {
        IAuthenticationService auth;
        public PlanList()
        {
            auth = DependencyService.Get<IAuthenticationService>();
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
        }

        protected async override void OnAppearing()
        {
            base.OnAppearing();
            var token = auth.GetUser();
            var details = await FireAssistant.GetClient(token);
            if (details != null)
            {
                var plans = await FireAssistant.GetPlan(details.TrainerName,details.UserName);
                planlist.ItemsSource = plans;
            }
            else
            {
                var label = new Label { Text = "Error please try again." };
            }
            planlist.ItemTapped += Planlist_ItemTapped;
        }

        private async void Planlist_ItemTapped(object sender, ItemTappedEventArgs e)
        {
            var details = e.Item as Plans;
            await PopupNavigation.Instance.PushAsync(new
    ClientSelectedPlan(details.PlanTitle,details.Monday,details.Tuesday, details.Description,details.Fee));
        }

        private void Button_Clicked(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new TabbedClient());
        }

        private void Button_Clicked_1(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new RequestPlan());
        }
    }
}

```

## PopupAddGoal.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<pages:PopupPage xmlns="http://xamarin.com/schemas/2014/forms"
                 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
                 xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
                 xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup"
                 x:Class="GymAndPAapp.Views.Clients.PopupAddGoal">


    <StackLayout Margin="15,10,15,80"
                 Padding="24"
                 BackgroundColor="#2a2a2a" >

        <Grid VerticalOptions="Center">
            <Grid.RowDefinitions>
                <RowDefinition Height="auto"/>
                <RowDefinition Height="150"/>
                <RowDefinition Height="auto"/>
            </Grid.RowDefinitions>
            <StackLayout Grid.Row="0" Grid.ColumnSpan="2">
```

```
20.             <Label  HorizontalTextAlignment="Center" TextColor="White" HorizontalOptions="Center" Text="Add a
    New Goal" FontSize="18" FontAttributes="Bold"/>
21.         </StackLayout>
22.         <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="1"  Grid.ColumnSpan="2">
23.             <Editor Placeholder="Entry a new goal" TextColor="White" PlaceholderColor="AliceBlue"
    HeightRequest="150" HorizontalOptions="Center" WidthRequest="330" BackgroundColor="#2a2a2a" FontSize="16"
    Text="{Binding Goal}"/>
24.         </Frame>
25.         <StackLayout Grid.Row="3" Grid.ColumnSpan="2" >
26.             <Button Text="Add"  FontSize="15" HeightRequest="50" WidthRequest="80" CornerRadius="80"
    FontAttributes="Bold" TextColor="white" Background="#ff4140" Command="{Binding AddCommand}"/>
27.         </StackLayout>
28.     </Grid>
29.   </StackLayout>
30. </pages:PopupPage>
31.
```

## PopupAddGoal.xaml.cs

```
1.  using GymAndPAapp.ViewModels;
2.  using System;
3.  using System.Collections.Generic;
4.  using System.Linq;
5.  using System.Text;
6.  using System.Threading.Tasks;
7.
8.  using Xamarin.Forms;
9.  using Xamarin.Forms.Xaml;
10.
11. namespace GymAndPAapp.Views.Clients
12. {
13.     [XamlCompilation(XamlCompilationOptions.Compile)]
14.     public partial class PopupAddGoal
15.     {
16.         public PopupAddGoal()
17.         {
18.             SetValue(NavigationPage.HasNavigationBarProperty, false);
19.             InitializeComponent();
20.             BindingContext = new GoalsViewModel();
21.         }
22.     }
23. }
24.
```

## PopUpCompletedGoals.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <pages:PopupPage x:Class="GymAndPAapp.Views.Clients.PopUpCompletedGoals"
3.                   xmlns="http://xamarin.com/schemas/2014/forms"
4.                   xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
5.                   xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
6.                   xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup">
7.
8.     <StackLayout Margin="15,10,15,80"
9.                   Padding="24"
10.                  BackgroundColor="#2a2a2a" >
11.
12.        <Grid VerticalOptions="Center">
13.            <Grid.RowDefinitions>
14.                <RowDefinition Height="auto"/>
15.                <RowDefinition Height="150"/>
16.                <RowDefinition Height="auto"/>
17.                <RowDefinition Height="auto"/>
18.            </Grid.RowDefinitions>
19.            <StackLayout Grid.Row="0" Grid.ColumnSpan="2">
20.                <Label  HorizontalTextAlignment="Center" TextColor="White" HorizontalOptions="Center" Text="Selected
    Goal" FontSize="18" FontAttributes="Bold"/>
```

```
21.             </StackLayout>
22.             <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="1"  Grid.ColumnSpan="2">
23.                 <StackLayout>
24.                     <Label x:Name="description" HeightRequest="150" HorizontalTextAlignment="Start"
    WidthRequest="330" TextColor="White" FontAttributes="Bold" BackgroundColor="#2a2a2a"
    HorizontalOptions="Center" Text="{Binding}"/>
25.                 </StackLayout>
26.               </Frame>
27.             <StackLayout Grid.Row="2" Grid.ColumnSpan="2">
28.                <Button Text="Remove Goal" FontSize="15" HeightRequest="50"  CornerRadius="80"
    FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked"/>
29.             </StackLayout>
30.             <StackLayout Grid.Row="3" Grid.ColumnSpan="2">
31.                <Button Text="Complete Goal" FontSize="15" HeightRequest="50"  CornerRadius="80"
    FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked_1"/>
32.             </StackLayout>
33.         </Grid>
34.         </StackLayout>
35. </pages:PopupPage>
36.
```

## PopUpCompletedGoals.xaml.cs

```
1.  using GymAndPAapp.ViewModels;
2.  using Rg.Plugins.Popup.Services;
3.  using System;
4.  using System.Collections.Generic;
5.  using System.Linq;
6.  using System.Text;
7.  using System.Threading.Tasks;
8.
9.  using Xamarin.Forms;
10. using Xamarin.Forms.Xaml;
11.
12. namespace GymAndPAapp.Views.Clients
13. {
14.     [XamlCompilation(XamlCompilationOptions.Compile)]
15.     public partial class PopUpCompletedGoals
16.     {
17.         public PopUpCompletedGoals(string goal)
18.         {
19.             SetValue(NavigationPage.HasNavigationBarProperty, false);
20.             InitializeComponent();
21.             description.Text = goal;
22.
23.         }
24.
25.         private async void Button_Clicked(object sender, EventArgs e)
26.         {
27.             App._goalsVar = true;
28.             var remove = await FireAssistant.DeleteGoal(App._variable, description.Text);
29.             await PopupNavigation.Instance.PopAllAsync();
30.             App.Current.MainPage = new NavigationPage(new TabbedClient());
31.
32.         }
33.
34.         private async void Button_Clicked_1(object sender, EventArgs e)
35.         {
36.             App._goalsVar = true;
37.             var remove = await FireAssistant.UpdateGoal(App._variable, description.Text);
38.             await PopupNavigation.Instance.PopAllAsync();
39.             App.Current.MainPage = new NavigationPage(new TabbedClient());
40.
41.         }
42.     }
43. }
44.
```

## RequestPlan.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.               xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.               x:Class="GymAndPAapp.Views.Clients.RequestPlan">
5.      <ContentPage.Content>
6.          <ScrollView BackgroundColor="#002d40">
7.              <StackLayout>
8.                  <Grid Margin="20,0,20,0" VerticalOptions="CenterAndExpand" RowSpacing="25">
9.                      <Grid.RowDefinitions>
10.                         <RowDefinition Height="Auto"/>
11.                         <RowDefinition Height="Auto"/>
12.                         <RowDefinition Height="Auto"/>
13.                         <RowDefinition Height="Auto"/>
14.                         <RowDefinition Height="Auto"/>
15.                         <RowDefinition Height="Auto"/>
16.                         <RowDefinition Height="Auto"/>
17.                     </Grid.RowDefinitions>
18.                     <StackLayout Grid.Row="0">
19.                         <Entry Placeholder="Age" PlaceholderColor="AliceBlue" FontSize="16" TextColor="white"
    x:Name="EnteredAge" />
20.                     </StackLayout>
21.                     <StackLayout Grid.Row="1">
22.                         <Entry Placeholder="Height" PlaceholderColor="AliceBlue" FontSize="16" TextColor="white"
    x:Name="EnteredHeight"/>
23.                     </StackLayout>
24.                     <StackLayout Grid.Row="2">
25.                         <Entry Placeholder="Weight" PlaceholderColor="AliceBlue" FontSize="16" TextColor="white"
    x:Name="EnteredWeight"/>
26.                     </StackLayout>
27.                     <StackLayout Grid.Row="3">
28.                         <Editor Placeholder="Goals" PlaceholderColor="AliceBlue" FontSize="16" TextColor="white"
    x:Name="EnteredGoal"/>
29.                     </StackLayout>
30.                     <StackLayout Grid.Row="4">
31.                         <Label Text="Access to equipment E.g dumbells, bench, etc." FontSize="16" TextColor="white"
    FontAttributes="Italic"/>
32.                         <Editor Placeholder="Equipment" PlaceholderColor="AliceBlue" FontSize="16" TextColor="white"
    x:Name="EnteredEquipment"/>
33.                     </StackLayout>
34.                     <StackLayout Grid.Row="5">
35.                         <Button Text="Request Plan"  FontSize="15" HeightRequest="50" CornerRadius="80"
    FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked"/>
36.                     </StackLayout>
37.                     <StackLayout Grid.Row="6">
38.                         <Button Text="Cancel"  FontSize="15" HeightRequest="50" CornerRadius="80" FontAttributes="Bold"
    TextColor="white" Background="#ff4140"  Clicked="Button_Clicked_1"/>
39.                     </StackLayout>
40.                 </Grid>
41.             </StackLayout>
42.         </ScrollView>
43.     </ContentPage.Content>
44. </ContentPage>
45.
```

## RequestPlan.xaml.cs

```
1.  using GymAndPAapp.ViewModels;
2.  using System;
3.  using System.Collections.Generic;
4.  using System.Linq;
5.  using System.Text;
6.  using System.Threading.Tasks;
7.
8.  using Xamarin.Forms;
9.  using Xamarin.Forms.Xaml;
10.
```

```
namespace GymAndPAapp.Views.Clients
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class RequestPlan : ContentPage
    {
        public RequestPlan()
        {
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();

        }
        private async void Button_Clicked(object sender, EventArgs e)
        {
            var details = await FireAssistant.CheckUsernameClient(App._variable);
            var request = await FireAssistant.MakeRequest(App._variable, details.TrainerName, EnteredAge.Text,
EnteredHeight.Text, EnteredGoal.Text, EnteredEquipment.Text,EnteredWeight.Text);
            if(request)
            {
                await App.Current.MainPage.DisplayAlert("Success", "Plan reuqest successful", "Continue");
                App.Current.MainPage = new NavigationPage(new TabbedClient());
            }
            else
            {
                await App.Current.MainPage.DisplayAlert("Error", "Plan reuqest failed, please try again", "Continue");
                App.Current.MainPage = new NavigationPage(new RequestPlan());
            }

        }

        private void Button_Clicked_1(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new TabbedClient());
        }
    }
}

```

## SelectedTrainer.xaml

```
 <?xml version="1.0" encoding="utf-8" ?>
<pages:PopupPage xmlns="http://xamarin.com/schemas/2014/forms"
                 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
                 xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
                 xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup"
                 x:Class="GymAndPAapp.Views.SelectedTrainer">
    <StackLayout Margin="15,10,15,80"
                 Padding="24"
                 BackgroundColor="#2a2a2a">
        <Grid>
            <Grid.RowDefinitions>
                <RowDefinition Height="auto"/>
                <RowDefinition Height="auto"/>
                <RowDefinition Height="450"/>
                <RowDefinition Height="20"/>
                <RowDefinition Height="auto"/>
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="100"/>
                <ColumnDefinition/>
            </Grid.ColumnDefinitions>
            <StackLayout Grid.Row="0" Grid.Column="0">
                <Label Text="    Trainer:" HorizontalOptions="Start" FontSize="18" FontAttributes="Bold"
TextColor="White"/>
            </StackLayout>
            <StackLayout Grid.Column="1" Grid.Row="0">
                <Label x:Name="trainer" HorizontalOptions="Center" FontSize="18" FontAttributes="Bold"
TextColor="White"/>
            </StackLayout>
            <StackLayout Grid.Row="1" Grid.ColumnSpan="2">
```

```
29.             <Label Text="    Plans:        Description:" HorizontalOptions="Start" FontSize="18" FontAttributes="Bold"
    TextColor="White"/>
30.         </StackLayout>
31.         <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="2"  Grid.ColumnSpan="2">
32.           <StackLayout Grid.Row="2" Grid.ColumnSpan="2" BackgroundColor="#2a2a2a">
33.             <ListView x:Name="planlist" ItemSelected="planlist_ItemSelected" IsEnabled="False">
34.               <ListView.ItemTemplate>
35.                 <DataTemplate>
36.                   <ViewCell>
37.                     <ViewCell.View>
38.                       <StackLayout>
39.                         <Grid>
40.                           <Grid.ColumnDefinitions>
41.                             <ColumnDefinition Width="10"/>
42.                             <ColumnDefinition Width="100"/>
43.                             <ColumnDefinition Width="250"/>
44.                           </Grid.ColumnDefinitions>
45.                           <Label Grid.Column="1" HorizontalOptions="Start"
46.                           VerticalOptions="Start"
47.                           VerticalTextAlignment="Center"
48.                           TextColor="White"
49.                           Text="{Binding PlanTitle}"/>
50.                            <Label Grid.Column="2" HorizontalOptions="Start"
51.                           VerticalOptions="Start"
52.                           VerticalTextAlignment="Center"
53.                           TextColor="White"
54.                           Text="{Binding Description}"/>
55.                         </Grid>
56.                       </StackLayout>
57.                     </ViewCell.View>
58.                   </ViewCell>
59.                 </DataTemplate>
60.               </ListView.ItemTemplate>
61.             </ListView>
62.           </StackLayout>
63.         </Frame>
64.         <StackLayout Grid.Row="4" Grid.ColumnSpan="2">
65.           <Button Text="Select Trainer"  FontSize="15"  HeightRequest="50" CornerRadius="80"
    FontAttributes="Bold" TextColor="white" Background="#ff4140" Clicked="Button_Clicked"/>
66.         </StackLayout>
67.       </Grid>
68.       </StackLayout>
69. </pages:PopupPage>
70.
```

## SelectedTrainer.xaml.cs

```
1.  using GymAndPAapp.ViewModels;
2.  using Rg.Plugins.Popup.Services;
3.  using System;
4.
5.  using Xamarin.Forms;
6.  using Xamarin.Forms.Xaml;
7.
8.  namespace GymAndPAapp.Views
9.  {
10.     [XamlCompilation(XamlCompilationOptions.Compile)]
11.     public partial class SelectedTrainer
12.     {
13.         public SelectedTrainer(string Trainer)
14.         {
15.             SetValue(NavigationPage.HasNavigationBarProperty, false);
16.             InitializeComponent();
17.             trainer.Text = Trainer;
18.         }
19.
20.         protected async override void OnAppearing()
```

```
21.         {
22.             base.OnAppearing();
23.             var plans = await FireAssistant.GetPlan(trainer.Text,App._variable);
24.             planlist.ItemsSource = plans;
25.         }
26.
27.         private async void Button_Clicked(object sender, EventArgs e)
28.         {
29.             var update = await FireAssistant.UpdateClient2(App._variable,trainer.Text);
30.             if(update)
31.             {
32.                 await PopupNavigation.Instance.PopAsync();
33.                 App.Current.MainPage = new NavigationPage(new PlanList());
34.             }
35.             else
36.             {
37.                 await App.Current.MainPage.DisplayAlert("ERROR", "Failed to select Trainer, please try again", "OK");
38.                 await PopupNavigation.Instance.PopAsync();
39.                 App.Current.MainPage = new NavigationPage(new PlanList());
40.             }
41.         }
42.
43.         private void planlist_ItemSelected(object sender, SelectedItemChangedEventArgs e)
44.         {
45.             var list = (ListView)sender;
46.             list.SelectedItem = null;
47.         }
48.     }
49. }
50.
```

## TabbedClient.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <TabbedPage xmlns="http://xamarin.com/schemas/2014/forms"
3.              xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml" xmlns:views="clr-namespace:GymAndPAapp.Views"
4.              x:Class="GymAndPAapp.Views.TabbedClient"
5.              xmlns:android="clr-
    namespace:Xamarin.Forms.PlatformConfiguration.AndroidSpecific;assembly=Xamarin.Forms.Core" xmlns:views1="clr-
    namespace:GymAndPAapp.Views.Clients"
6.              android:TabbedPage.ToolbarPlacement="Bottom"
7.              android:TabbedPage.BarItemColor ="White"
8.              android:TabbedPage.BarSelectedItemColor ="#ff4140">
9.
10.     <!--Pages can be added as references or inline-->
11.
12.         <views1:ClientGoals Title="Goals"></views1:ClientGoals>
13.         <views:ClientHome Title="Home"></views:ClientHome>
14.         <views:Settings Title="Settings"></views:Settings>
15.
16. </TabbedPage>
17.
```

## TabbedClient.xaml.cs

```
1.  using Xamarin.Forms;
2.  using Xamarin.Forms.Xaml;
3.  using Xamarin.Forms.PlatformConfiguration.AndroidSpecific;
4.  using TabbedPage = Xamarin.Forms.TabbedPage;
5.  using Xamarin.Forms.PlatformConfiguration;
6.
7.  namespace GymAndPAapp.Views
8.  {
9.      [XamlCompilation(XamlCompilationOptions.Compile)]
10.     public partial class TabbedClient : TabbedPage
11.     {
```

```
        public bool goals = false;
        public TabbedClient()
        {
            goals = App._goalsVar;
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
            if(goals == true)
            {
                CurrentPage = Children[0];
            }
            else
            {
                CurrentPage = Children[1];
            }

        }

    }
}

```

## TrainerList.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="GymAndPAapp.Views.TrainerList">

    <ContentPage.Content>
        <StackLayout BackgroundColor="#002d40">
            <Grid>
                <Grid.RowDefinitions>
                    <RowDefinition Height="10"/>
                    <RowDefinition Height="30"/>
                    <RowDefinition Height="580"/>
                    <RowDefinition Height="auto"/>
                </Grid.RowDefinitions>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="*"/>
                    <ColumnDefinition Width="*"/>
                </Grid.ColumnDefinitions>
                <StackLayout Grid.Row="1" Grid.ColumnSpan="3">
                    <Label Text="Available Trainers" FontSize="23" TextDecorations="Underline" TextColor="White" FontAttributes="Bold" HorizontalOptions="Center"/>
                </StackLayout>
                <Frame BackgroundColor="White" Padding="2" Grid.Row="2" Grid.ColumnSpan="3">
                    <ListView x:Name="trainerlist" Grid.Row="2" Grid.ColumnSpan="2" SeparatorColor="White" BackgroundColor="#002d40">
                        <ListView.ItemTemplate>
                            <DataTemplate>
                                <ViewCell>
                                    <ViewCell.View>
                                        <StackLayout >
                                            <Label HorizontalOptions="Center"
                                              VerticalOptions="Center"
                                              VerticalTextAlignment="Center"
                                              FontAttributes="Bold"
                                              TextColor="White"
                                              Text="{Binding UserName}"/>
                                        </StackLayout>
                                    </ViewCell.View>
                                </ViewCell>
                            </DataTemplate>
                        </ListView.ItemTemplate>
                    </ListView>
                </Frame>
                <Button Text="Return" FontSize="15" HeightRequest="50" Grid.Row="4" Grid.ColumnSpan="3" CornerRadius="80" FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked"/>
            </Grid>
```

```
        </StackLayout>
    </ContentPage.Content>
</ContentPage>

```

## TrainerList.xaml.cs

```
using System;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;
using GymAndPAapp.Tables;
using System.Collections.ObjectModel;
using System.Collections;
using GymAndPAapp.ViewModels;
using Rg.Plugins.Popup.Services;

namespace GymAndPAapp.Views
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class TrainerList : ContentPage
    {

        public TrainerList()
        {
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
        }

        protected async override void OnAppearing()
        {
            base.OnAppearing();
            var allTrainers = await FireAssistant.GetAllTrainers();
            trainerlist.ItemsSource = allTrainers;

            trainerlist.ItemTapped += Trainerlist_ItemTapped;

        }

        private async void Trainerlist_ItemTapped(object sender, ItemTappedEventArgs e)
        {
            var details = e.Item as GymAndPAapp.Tables.Trainers;
            await PopupNavigation.Instance.PushAsync(new SelectedTrainer(details.UserName));
        }

        private void Button_Clicked(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new TabbedClient());
        }
    }
}

```

# *Shared*

## AddImages.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<pages:PopupPage xmlns="http://xamarin.com/schemas/2014/forms"
                 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
                 xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
                 xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup" xmlns:Image="clr-namespace:ImageCircle.Forms.Plugin.Abstractions;assembly=ImageCircle.Forms.Plugin"
                  x:Class="GymAndPAapp.Views.Shared.AddImage">


    <StackLayout Margin="50,50,50,200"
                 Padding="30">
```

```
11.
12.         <Grid VerticalOptions="Center">
13.             <Grid.RowDefinitions>
14.                 <RowDefinition Height="60"/>
15.                 <RowDefinition Height="auto"/>
16.                 <RowDefinition Height="auto"/>
17.             </Grid.RowDefinitions>
18.             <StackLayout Grid.Row="1" Grid.ColumnSpan="2" >
19.                 <Button Text="Camera"  FontSize="15" HeightRequest="50" WidthRequest="80" CornerRadius="80"
    FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked_1"/>
20.             </StackLayout>
21.             <StackLayout Grid.Row="2" Grid.ColumnSpan="2" >
22.                 <Button Text="Gallery"  FontSize="15" HeightRequest="50" WidthRequest="80" CornerRadius="80"
    FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
23.             </StackLayout>
24.         </Grid>
25.     </StackLayout>
26. </pages:PopupPage>
27.
```

## AddImages.xaml.cs

```
1.  using Firebase.Storage;
2.  using GymAndPAapp.ViewModels;
3.  using GymAndPAapp.Views.Admin;
4.  using Plugin.Media;
5.  using Plugin.Media.Abstractions;
6.  using Rg.Plugins.Popup.Services;
7.  using System;
8.  using System.Collections.Generic;
9.  using System.Diagnostics;
10. using System.IO;
11. using System.Linq;
12. using System.Net;
13. using System.Text;
14. using System.Threading.Tasks;
15.
16. using Xamarin.Forms;
17. using Xamarin.Forms.Xaml;
18.
19. namespace GymAndPAapp.Views.Shared
20. {
21.     [XamlCompilation(XamlCompilationOptions.Compile)]
22.     public partial class AddImage
23.     {
24.         MediaFile file;
25.         IAuthenticationService auth;
26.         public AddImage()
27.         {
28.             App._variable5 = this.GetType().Name;
29.             auth = DependencyService.Get<IAuthenticationService>();
30.             InitializeComponent();
31.         }
32.
33.         private async void Button_Clicked(object sender, EventArgs e)
34.         {
35.             await CrossMedia.Current.Initialize();
36.             try
37.             {
38.                 file = await Plugin.Media.CrossMedia.Current.PickPhotoAsync(new
    Plugin.Media.Abstractions.PickMediaOptions
39.                 {
40.                     PhotoSize = Plugin.Media.Abstractions.PhotoSize.Large
41.                 });
42.                 await PopupNavigation.Instance.PopAllAsync();
43.                 await FireAssistant.SaveImage(file.GetStream(), App._variable);
44.                 App.profile = null;
45.                 var token = auth.GetUser();
46.                 var x = await FireAssistant.GetClient(token);
```

```
            var y = await FireAssistant.GetTrainer(token);
            if (x != null)
            {
                App.Current.MainPage = new NavigationPage(new TabbedClient());

            }
            else if (x!= null)
            {
                App.Current.MainPage = new NavigationPage(new TabbedTrainer());

            }
            else
            {
                App.Current.MainPage = new NavigationPage(new TabbedAdmin());
            }
        }
        catch (Exception ex)
        {
            Debug.WriteLine(ex.Message);
        }
    }
    private async void Button_Clicked_1(object sender, EventArgs e)
    {
        await CrossMedia.Current.Initialize();
        try
        {
            if (CrossMedia.Current.IsCameraAvailable && CrossMedia.Current.IsTakePhotoSupported)
            {
                file = await CrossMedia.Current.TakePhotoAsync(new StoreCameraMediaOptions
                {
                    SaveToAlbum = true,
                    PhotoSize = PhotoSize.Large
                });
            }
            await PopupNavigation.Instance.PopAllAsync();
            await FireAssistant.SaveImage(file.GetStream(), App._variable);
            var token = auth.GetUser();
            var x = await FireAssistant.GetClient(token);
            var y = await FireAssistant.GetTrainer(token);
            if (x != null)
            {
                App.Current.MainPage = new NavigationPage(new TabbedClient());

            }
            else if (x != null)
            {
                App.Current.MainPage = new NavigationPage(new TabbedTrainer());

            }
            else
            {
                App.Current.MainPage = new NavigationPage(new TabbedAdmin());
            }
        }
        catch (Exception ex)
        {
            Debug.WriteLine(ex.Message);
        }
    }

}
}

```

## ChangePassword.xaml

<?xml version="1.0" encoding="utf-8" ?>

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
```

```
3.              x:Class="GymAndPAapp.Views.ChangePassword">
4.      <ContentPage.Content>
5.          <ScrollView BackgroundColor="#002d40">
6.              <StackLayout>
7.                  <Grid Margin="20,0,20,0" VerticalOptions="CenterAndExpand" RowSpacing="25">
8.                      <Grid.RowDefinitions>
9.                          <RowDefinition Height="Auto"/>
10.                         <RowDefinition Height="Auto"/>
11.                         <RowDefinition Height="Auto"/>
12.                         <RowDefinition Height="Auto"/>
13.                         <RowDefinition Height="Auto"/>
14.                     </Grid.RowDefinitions>
15.                     <StackLayout Grid.Row="0">
16.                         <Entry Placeholder="New Password" PlaceholderColor="AliceBlue" TextColor="White"
    IsPassword="True" FontSize="16" Text="{Binding NewPassword}"/>
17.                     </StackLayout>
18.                     <StackLayout Grid.Row="1">
19.                         <Entry Placeholder="Confirm Password" PlaceholderColor="AliceBlue" TextColor="White"
    IsPassword="True" FontSize="16" Text="{Binding ConfirmNewPassword}"/>
20.                     </StackLayout>
21.                     <StackLayout Grid.Row="2">
22.                         <Button Text="Confirm" FontSize="15"  HeightRequest="50" Grid.Row="1" CornerRadius="80"
    FontAttributes="Bold" TextColor="White" Background="#ff4140" Command="{Binding
    ChangePasswordCommand}"/>
23.                     </StackLayout>
24.                     <StackLayout Grid.Row="3">
25.                         <Button Text="Cancel" FontSize="15"  HeightRequest="50" Grid.Row="1" CornerRadius="80"
    FontAttributes="Bold" TextColor="White" Background="#ff4140" Command="{Binding Cancel}"/>
26.                     </StackLayout>
27.                 </Grid>
28.             </StackLayout>
29.         </ScrollView>
30.     </ContentPage.Content>
31. </ContentPage>
32.
```

## ChangePassword.xaml.cs

```
1.  using GymAndPAapp.Tables;
2.  using GymAndPAapp.ViewModels;
3.  using System;
4.  using System.IO;
5.
6.  using Xamarin.Forms;
7.  using Xamarin.Forms.Xaml;
8.
9.  namespace GymAndPAapp.Views
10. {
11.     [XamlCompilation(XamlCompilationOptions.Compile)]
12.     public partial class ChangePassword : ContentPage
13.     {
14.         public ChangePassword()
15.         {
16.             SetValue(NavigationPage.HasNavigationBarProperty, false);
17.             InitializeComponent();
18.             BindingContext = new ChangePassViewModel();
19.         }
20.     }
21. }
22.
```

## CloseAccount.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.              xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.              x:Class="GymAndPAapp.Views.Shared.CloseAccount">
```

```
    <ContentPage.Content>
        <ScrollView BackgroundColor="#002d40">
            <StackLayout>
                <Grid Margin="20,0,20,0" VerticalOptions="CenterAndExpand" RowSpacing="25">
                    <Grid.RowDefinitions>
                        <RowDefinition Height="Auto"/>
                        <RowDefinition Height="Auto"/>
                        <RowDefinition Height="Auto"/>
                        <RowDefinition Height="Auto"/>
                    </Grid.RowDefinitions>
                    <StackLayout Grid.Row="0">
                        <Frame Grid.Row="0" BackgroundColor="#ff4140" Padding="1">
                            <Label HorizontalOptions="Center" BackgroundColor="#002d40" TextColor="white" TextDecorations="Underline" FontAttributes="Bold"  Text="In order to permanently close your account please click the Confirm Button. Please Note this action cannot be undone and you will be required to register again to use the application."/>
                        </Frame>
                    </StackLayout>
                    <StackLayout Grid.Row="2">
                        <Button Text="Confirm" FontSize="15"  HeightRequest="50" Grid.Row="1" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
                    </StackLayout>
                    <StackLayout Grid.Row="3">
                        <Button Text="Cancel" FontSize="15"  HeightRequest="50" Grid.Row="1" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked_1"/>
                    </StackLayout>
                </Grid>
            </StackLayout>
        </ScrollView>
    </ContentPage.Content>
</ContentPage>

```

## CloseAccount.xaml.cs

```
using GymAndPAapp.ViewModels;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views.Shared
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class CloseAccount : ContentPage
    {
        IAuthenticationService auth;
        public CloseAccount()
        {
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
            auth = DependencyService.Get<IAuthenticationService>();
        }

        private async void Button_Clicked(object sender, EventArgs e)
        {
            var token = auth.GetUser();
            var accountType = await FireAssistant.GetClient(token);
            var accountType2 = await FireAssistant.GetTrainer(token);
            if(accountType != null)
            {
                var updatedb = await FireAssistant.DeleteClient(accountType.UserName, token);
                if(updatedb)
                {
                    auth.CloseAccount();
```

```
35.                 await App.Current.MainPage.Navigation.PushAsync(new LoginPage());
36.             }
37.             else
38.             {
39.                 await App.Current.MainPage.DisplayAlert("ERROR", "Error Deleating account please try again", "OK");
40.             }
41.         }
42.         else if(accountType2 !=null)
43.         {
44.             var updatedb = await FireAssistant.DeleteTrainer(accountType.UserName, token);
45.             if (updatedb)
46.             {
47.                 auth.CloseAccount();
48.                 await App.Current.MainPage.Navigation.PushAsync(new LoginPage());
49.             }
50.             else
51.             {
52.                 await App.Current.MainPage.DisplayAlert("ERROR", "Error Deleating account please try again", "OK");
53.             }
54.         }
55.         else
56.         {
57.             await App.Current.MainPage.DisplayAlert("ERROR", "Error Deleating account please try again", "OK");
58.         }
59.     }
60.
61.     private async void Button_Clicked_1(object sender, EventArgs e)
62.     {
63.         var token = auth.GetUser();
64.         var accountType = await FireAssistant.GetClient(token);
65.         if(accountType != null)
66.         {
67.             await App.Current.MainPage.Navigation.PushAsync(new TabbedClient());
68.         }
69.         else
70.         {
71.             await App.Current.MainPage.Navigation.PushAsync(new TabbedTrainer());
72.         }
73.     }
74. }
75. }
76.
```

## LoginPage.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.              xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.              xmlns:d="http://xamarin.com/schemas/2014/forms/design"
5.              xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
6.              mc:Ignorable="d"
7.              x:Class="GymAndPAapp.Views.LoginPage"
8.              Shell.NavBarIsVisible="False">
9.      <ContentPage.Content>
10.         <ScrollView BackgroundColor ="#002d40">
11.             <StackLayout>
12.                 <Grid VerticalOptions="CenterAndExpand" Margin="40" RowSpacing="10">
13.                     <Grid.RowDefinitions>
14.                         <RowDefinition/>
15.                         <RowDefinition/>
16.                         <RowDefinition/>
17.                         <RowDefinition/>
18.                         <RowDefinition Height="80"/>
19.                     </Grid.RowDefinitions>
20.                 <Grid.ColumnDefinitions>
21.                     <ColumnDefinition/>
22.                     <ColumnDefinition/>
23.                 </Grid.ColumnDefinitions>
```

```
24.                 <Entry Placeholder="Email" PlaceholderColor="AliceBlue" TextColor="White"  x:Name="EntryEmail"
    Grid.Row="1" Grid.ColumnSpan="2" Text="{Binding Email}"/>
25.                 <Entry Placeholder="Password" PlaceholderColor="AliceBlue"  TextColor="White" x:Name="EntryPass"
    IsPassword="True" Grid.Row="2" Grid.ColumnSpan="2" Text="{Binding Password}"/>
26.                 <Button Text="Login" Grid.Row="3" Grid.ColumnSpan="2" FontSize="15"  HeightRequest="50"
    CornerRadius="80" TextColor="white" FontAttributes="bold" Background="#ff4140" Command="{Binding
    LoginCommand}"/>
27.                 <Button Text="Dont have an account? SignUp here" BorderWidth="1" BorderColor="#ff4140"
    BackgroundColor="#002d40" Grid.Row="4" Grid.Column="0" TextColor="white" Command="{Binding
    SignUpCommand}"/>
28.                 <Button Text="Forgot Password" BorderWidth="1" BorderColor="#ff4140" BackgroundColor="#002d40"
    Grid.Row="4" Grid.Column="1" TextColor="white" Command="{Binding ResetPassCommand}"/>
29.                 </Grid>
30.             </StackLayout>
31.         </ScrollView>
32.     </ContentPage.Content>
33. </ContentPage>
34.
```

## LoginPage.xaml.cs

```
1.  using GymAndPAapp.Tables;
2.  using GymAndPAapp.ViewModels;
3.  using System;
4.  using System.IO;
5.  using Xamarin.Forms;
6.  using Xamarin.Forms.Xaml;
7.
8.  namespace GymAndPAapp.Views
9.  {
10.     [XamlCompilation(XamlCompilationOptions.Compile)]
11.     public partial class LoginPage : ContentPage
12.     {
13.         public LoginPage()
14.         {
15.             SetValue(NavigationPage.HasNavigationBarProperty, false);
16.             InitializeComponent();
17.             BindingContext = new LoginPageViewModel();
18.         }
19.     }
20. }
21.
```

## ProfilePicOptions.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <pages:PopupPage xmlns="http://xamarin.com/schemas/2014/forms"
3.                   xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.                   xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
5.                   xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup"
6.                   x:Class="GymAndPAapp.Views.Shared.ProfilePicOptions">
7.
8.
9.     <StackLayout Margin="50,50,50,200"
10.                  Padding="30">
11.
12.         <Grid VerticalOptions="Center">
13.             <Grid.RowDefinitions>
14.                 <RowDefinition Height="60"/>
15.                 <RowDefinition Height="auto"/>
16.                 <RowDefinition Height="auto"/>
17.             </Grid.RowDefinitions>
18.             <StackLayout Grid.Row="1" Grid.ColumnSpan="2" >
19.                 <Button Text="Remove Image"  FontSize="15" HeightRequest="50" WidthRequest="80" CornerRadius="80"
    FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked_1"/>
20.             </StackLayout>
21.             <StackLayout Grid.Row="2" Grid.ColumnSpan="2" >
```

```
22.             <Button Text="Add Image"  FontSize="15" HeightRequest="50" WidthRequest="80" CornerRadius="80"
    FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
23.         </StackLayout>
24.       </Grid>
25.   </StackLayout>
26. </pages:PopupPage>
27.
```

## ProfilePicOptions.xaml.cs

```
1.  using GymAndPAapp.ViewModels;
2.  using Rg.Plugins.Popup.Services;
3.  using System;
4.  using System.Collections.Generic;
5.  using System.Linq;
6.  using System.Text;
7.  using System.Threading.Tasks;
8.
9.  using Xamarin.Forms;
10. using Xamarin.Forms.Xaml;
11.
12. namespace GymAndPAapp.Views.Shared
13. {
14.     [XamlCompilation(XamlCompilationOptions.Compile)]
15.     public partial class ProfilePicOptions
16.     {
17.         IAuthenticationService auth;
18.         public ProfilePicOptions()
19.         {
20.             auth = DependencyService.Get<IAuthenticationService>();
21.             App._variable5 = this.GetType().Name;
22.             InitializeComponent();
23.         }
24.
25.         private async void Button_Clicked(object sender, EventArgs e)
26.         {
27.             await PopupNavigation.Instance.PushAsync(new AddImage());
28.         }
29.
30.         private async void Button_Clicked_1(object sender, EventArgs e)
31.         {
32.             await FireAssistant.DelImage( App._variable);
33.             App.profile = null;
34.             var token = auth.GetUser();
35.             var x = await FireAssistant.GetClient(token);
36.             if (x != null)
37.             {
38.               await PopupNavigation.Instance.PopAllAsync();
39.               App.Current.MainPage = new NavigationPage(new TabbedClient());
40.             }
41.             else
42.             {
43.               await PopupNavigation.Instance.PopAllAsync();
44.               App.Current.MainPage = new NavigationPage(new TabbedTrainer());
45.             }
46.         }
47.     }
48. }
49.
```

## RegistrationPage.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.               xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.               x:Class="GymAndPAapp.Views.RegistrationPage">
5.     <ContentPage.Content>
```

```
6.          <ScrollView BackgroundColor="#002d40">
7.             <StackLayout>
8.                <Grid Margin="20,0,20,0" VerticalOptions="CenterAndExpand" RowSpacing="25">
9.                   <Grid.RowDefinitions>
10.                     <RowDefinition Height="Auto"/>
11.                     <RowDefinition Height="Auto"/>
12.                     <RowDefinition Height="Auto"/>
13.                     <RowDefinition Height="Auto"/>
14.                     <RowDefinition Height="Auto"/>
15.                     <RowDefinition Height="Auto"/>
16.                     <RowDefinition Height="Auto"/>
17.                  </Grid.RowDefinitions>
18.                  <StackLayout Grid.Row="0">
19.                     <Entry Placeholder="UserName" PlaceholderColor="AliceBlue" FontSize="16" TextColor="white"
    Text="{Binding UserName}"/>
20.                  </StackLayout>
21.                  <StackLayout Grid.Row="1">
22.                     <Entry Placeholder="Password" PlaceholderColor="AliceBlue" IsPassword="True" TextColor="white"
    FontSize="16" Text="{Binding Password}"/>
23.                  </StackLayout>
24.                  <StackLayout Grid.Row="2">
25.                     <Entry Placeholder="Confirm Password" PlaceholderColor="AliceBlue" IsPassword="True"
    TextColor="white" FontSize="16" Text="{Binding ConfirmPassword}"/>
26.                  </StackLayout>
27.                  <StackLayout Grid.Row="3">
28.                     <Label Text="For Trainers please enter a vaild paypal email address to recieve payments"
    TextColor="white" FontSize="16" FontAttributes="Italic"/>
29.                     <Entry Placeholder="Email" PlaceholderColor="AliceBlue" TextColor="white" FontSize="16"
    Text="{Binding Email}"/>
30.                  </StackLayout>
31.                  <StackLayout Grid.Row="4">
32.                     <Label Text="Are you a Trainer?" TextColor="white" FontAttributes="Bold"/>
33.                     <CheckBox IsChecked="{Binding CheckedBox}" Color="#ff4140"/>
34.                  </StackLayout>
35.                  <StackLayout Grid.Row="5">
36.                     <Button Text="Register"  FontSize="15" HeightRequest="50" CornerRadius="80" FontAttributes="Bold"
    TextColor="white" Background="#ff4140"  Command="{Binding RegisterCommand}"/>
37.                  </StackLayout>
38.                  <StackLayout Grid.Row="6">
39.                     <Button Text="Login Page"  FontSize="15" HeightRequest="50" CornerRadius="80"
    FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked"/>
40.                  </StackLayout>
41.               </Grid>
42.            </StackLayout>
43.         </ScrollView>
44.      </ContentPage.Content>
45.   </ContentPage>
46.
```

## RegistrationPage.xaml.cs

```
1.    using GymAndPAapp.Tables;
2.    using GymAndPAapp.ViewModels;
3.    using System;
4.    using System.IO;
5.
6.    using Xamarin.Forms;
7.    using Xamarin.Forms.Xaml;
8.
9.    namespace GymAndPAapp.Views
10.   {
11.      [XamlCompilation(XamlCompilationOptions.Compile)]
12.      public partial class RegistrationPage : ContentPage
13.      {
14.         public RegistrationPage()
15.         {
16.            SetValue(NavigationPage.HasNavigationBarProperty, false);
```

```
            InitializeComponent();
            BindingContext = new RegistrationViewModel();
        }

        private void Button_Clicked(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new LoginPage());
        }
    }
}

```

## ResetPasswordPage.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:d="http://xamarin.com/schemas/2014/forms/design"
             xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
             mc:Ignorable="d"
             x:Class="GymAndPAapp.Views.Shared.ResetPasswordPage"
             Shell.NavBarIsVisible="False">
    <ContentPage.Content>
        <ScrollView BackgroundColor ="#002d40">
            <StackLayout>
                <Grid VerticalOptions="CenterAndExpand" Margin="40" RowSpacing="10">
                    <Grid.RowDefinitions>
                        <RowDefinition/>
                        <RowDefinition/>
                        <RowDefinition/>
                    </Grid.RowDefinitions>
                    <Entry Placeholder="Enter Email Address" PlaceholderColor="AliceBlue" TextColor="White"
x:Name="EntryEmail" Grid.Row="0"/>
                    <Button Text="Reset Password" Grid.Row="1" FontSize="15"  HeightRequest="50" CornerRadius="80"
FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
                    <Button Text="Return" Grid.Row="2" FontSize="15"  HeightRequest="50" CornerRadius="80"
FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked_1"/>
                </Grid>
            </StackLayout>
        </ScrollView>
    </ContentPage.Content>
</ContentPage>

```

## ResetPasswordPage.xaml.cs

```
using GymAndPAapp.ViewModels;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views.Shared
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class ResetPasswordPage : ContentPage
    {
        IAuthenticationService auth;
        public ResetPasswordPage()
        {
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            auth = DependencyService.Get<IAuthenticationService>();
            InitializeComponent();
```

```
        }

        private async void Button_Clicked(object sender, EventArgs e)
        {
            auth.resetEmail(EntryEmail.Text);
            await App.Current.MainPage.DisplayAlert("", "Password Reset link sent", "OK");
            App.Current.MainPage = new NavigationPage(new LoginPage());
        }

        private void Button_Clicked_1(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new LoginPage());
        }
    }
}

```

## Settings.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="GymAndPAapp.Views.Settings">
    <ContentPage.Content>
        <StackLayout BackgroundColor="#002d40">
            <Grid VerticalOptions="CenterAndExpand" Margin="60" RowSpacing="30">
                <Grid.RowDefinitions>
                    <RowDefinition/>
                    <RowDefinition/>
                    <RowDefinition/>
                </Grid.RowDefinitions>
                <Button Text="Change Password"  FontSize="15"  HeightRequest="50" Grid.Row="0" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked_1"/>
                <Button Text="Logout" FontSize="15"  HeightRequest="50" Grid.Row="1" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked_2"/>
                <Button Text="Close Account" FontSize="15"  HeightRequest="50" Grid.Row="2" CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
            </Grid>
        </StackLayout>
    </ContentPage.Content>
</ContentPage>

```

## Settings.xaml.cs

```
using GymAndPAapp.ViewModels;
using GymAndPAapp.Views.Shared;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class Settings : ContentPage
    {
        IAuthenticationService auth;
        public Settings()
        {
            InitializeComponent();
            auth = DependencyService.Get<IAuthenticationService>();
            SetValue(NavigationPage.HasNavigationBarProperty, false);
```

```
        }
        async void Button_Clicked_1(object sender, EventArgs e)
        {
            await Navigation.PushAsync(new ChangePassword());
        }
        async void Button_Clicked_2(object sender, EventArgs e)
        {
            App._variable = "";
            App._variable2 = "";
            App._variable4 = "";
            App._variable5 = "";
            App._VarArray.Clear();
            var signout = auth.SignOut();
            if(signout)
            {
                await Navigation.PushAsync(new LoginPage());
            }
        }

        async void Button_Clicked(object sender, EventArgs e)
        {
            await Navigation.PushAsync(new CloseAccount());
        }
    }
}

```

## *Trainer*

## EditPlan.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<pages:PopupPage xmlns="http://xamarin.com/schemas/2014/forms"
                 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
                 xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
                 xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup"
                 x:Class="GymAndPAapp.Views.EditPlan">
    <ScrollView>
        <StackLayout Margin="12"
                     Padding="35"
                     BackgroundColor="#2a2a2a">
            <Grid>
                <Grid.RowDefinitions>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                    <RowDefinition Height="Auto"/>
                </Grid.RowDefinitions>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition/>
                    <ColumnDefinition/>
                </Grid.ColumnDefinitions>
                <StackLayout Grid.Row="0" Grid.ColumnSpan="2">
                    <Label Text="Monday" FontSize="23" TextColor="White" HorizontalOptions="Center"
FontAttributes="Bold" x:Name="Monday"/>
                    <Label Text="Tuesday" FontSize="23" TextColor="White" HorizontalOptions="Center"
FontAttributes="Bold" x:Name="Tuesday"/>
                    <Label Text="Wednesday" FontSize="23" TextColor="White" HorizontalOptions="Center"
FontAttributes="Bold" x:Name="Wednesday"/>
                    <Label Text="Thursday" FontSize="23" TextColor="White" HorizontalOptions="Center"
FontAttributes="Bold" x:Name="Thursday"/>
```

```
35.             <Label Text="Friday" FontSize="23" TextColor="White" HorizontalOptions="Center"
    FontAttributes="Bold" x:Name="Friday"/>
36.             <Label Text="Saturday" FontSize="23" TextColor="White" HorizontalOptions="Center"
    FontAttributes="Bold" x:Name="Saturday"/>
37.             <Label Text="Sunday" FontSize="23" TextColor="White" HorizontalOptions="Center"
    FontAttributes="Bold" x:Name="Sunday"/>
38.         </StackLayout>
39.         <Frame Grid.Row="0" BackgroundColor="#ff4140" Padding="2" Grid.ColumnSpan="2" x:Name="frame2" >
40.           <StackLayout >
41.             <Entry x:Name="PlanTitleXaml" Placeholder="Plan Title" BackgroundColor="#2a2a2a"
    PlaceholderColor="AliceBlue" FontSize="16" TextColor="White" Text="{Binding Title}"/>
42.           </StackLayout>
43.         </Frame>
44.         <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="1"  Grid.ColumnSpan="2" x:Name="frame1" >
45.           <Editor x:Name="DescriptionHide" Placeholder="Description" HeightRequest="180"
    VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand" BackgroundColor="#2a2a2a"
    PlaceholderColor="AliceBlue" FontSize="16" TextColor="Black" Text="{Binding Description}"/>
46.         </Frame>
47.         <StackLayout Grid.Row="2" Grid.ColumnSpan="2">
48.           <Picker x:Name="picker2"
49.                   Title="Select Exercise"
50.                   TextColor="White"
51.                   TitleColor="White"
52.                   SelectedIndexChanged="picker2_SelectedIndexChanged"
53.                   ItemDisplayBinding="{Binding name}">
54.           </Picker>
55.         </StackLayout>
56.         <StackLayout Grid.Row="3" Grid.Column="1">
57.           <Picker x:Name="picker"
58.                   Title="Select Reps"
59.                   TextColor="White"
60.                   TitleColor="White"
61.                   SelectedItem="{Binding Amount}">
62.             <Picker.ItemsSource>
63.               <x:Array Type="{x:Type x:String}">
64.                 <x:String>1</x:String>
65.                 <x:String>2</x:String>
66.                 <x:String>3</x:String>
67.                 <x:String>4</x:String>
68.                 <x:String>5</x:String>
69.                 <x:String>6</x:String>
70.                 <x:String>7</x:String>
71.                 <x:String>8</x:String>
72.                 <x:String>9</x:String>
73.                 <x:String>10</x:String>
74.                 <x:String>11</x:String>
75.                 <x:String>12</x:String>
76.                 <x:String>13</x:String>
77.                 <x:String>14</x:String>
78.                 <x:String>15</x:String>
79.                 <x:String>16</x:String>
80.                 <x:String>17</x:String>
81.                 <x:String>18</x:String>
82.                 <x:String>19</x:String>
83.                 <x:String>20</x:String>
84.                 <x:String>21</x:String>
85.                 <x:String>22</x:String>
86.                 <x:String>23</x:String>
87.                 <x:String>24</x:String>
88.                 <x:String>25</x:String>
89.                 <x:String>26</x:String>
90.                 <x:String>27</x:String>
91.                 <x:String>28</x:String>
92.                 <x:String>29</x:String>
93.                 <x:String>30</x:String>
94.                 <x:String>31</x:String>
95.                 <x:String>32</x:String>
96.                 <x:String>33</x:String>
97.                 <x:String>34</x:String>
98.                 <x:String>35</x:String>
99.                 <x:String>36</x:String>
```

```
                    <x:String>37</x:String>
                    <x:String>38</x:String>
                    <x:String>39</x:String>
                    <x:String>40</x:String>
                    <x:String>41</x:String>
                    <x:String>42</x:String>
                    <x:String>43</x:String>
                    <x:String>44</x:String>
                    <x:String>45</x:String>
                    <x:String>46</x:String>
                    <x:String>47</x:String>
                    <x:String>48</x:String>
                    <x:String>49</x:String>
                    <x:String>50</x:String>
                  </x:Array>
                </Picker.ItemsSource>
              </Picker>
            </StackLayout>
            <StackLayout Grid.Row="3" Grid.Column="0">
              <Picker x:Name="picker1"
                  Title="Select Sets"
                  TextColor="White"
                  TitleColor="White"
                  SelectedItem="{Binding Sets}">
                <Picker.ItemsSource>
                  <x:Array Type="{x:Type x:String}">
                    <x:String>1</x:String>
                    <x:String>2</x:String>
                    <x:String>3</x:String>
                    <x:String>4</x:String>
                    <x:String>5</x:String>
                  </x:Array>
                </Picker.ItemsSource>
              </Picker>
            </StackLayout>
            <StackLayout Grid.Row="4" Grid.Column="0" HorizontalOptions="Center" >
              <Button Text="Add"  FontSize="15" HeightRequest="50" x:Name="AddButton" Grid.Row="3"
CornerRadius="80" FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked"/>
            </StackLayout>
            <StackLayout Grid.Row="4" Grid.Column="1" HorizontalOptions="Center" >
              <Button Text="Next Day"  FontSize="15" HeightRequest="50" x:Name="NextDayButton" Grid.Row="3"
CornerRadius="80" FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked_1"/>
            </StackLayout>
            <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="7" Grid.ColumnSpan="2" x:Name="frame3">
              <StackLayout>
                <Entry x:Name="Fees" Placeholder="Fee " BackgroundColor="#2a2a2a" TextColor="White"
PlaceholderColor="AliceBlue" FontSize="16" Text="{Binding Fee}"/>
              </StackLayout>
            </Frame>
            <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="8" Grid.ColumnSpan="2" x:Name="frame4">
              <StackLayout>
                <Entry x:Name="DCI" Placeholder="Daily Calorie intake" BackgroundColor="#2a2a2a"
PlaceholderColor="AliceBlue" TextColor="White" FontSize="16" Text="{Binding Calorie}"/>
              </StackLayout>
            </Frame>
            <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="9" Grid.ColumnSpan="2" x:Name="frame5">
              <StackLayout>
                <Entry  x:Name="LET" Placeholder="Link to Excercise Tutorials" BackgroundColor="#2a2a2a"
PlaceholderColor="AliceBlue" TextColor="White" FontSize="16" Text="{Binding Link}"/>
              </StackLayout>
            </Frame>
            <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="10" Grid.ColumnSpan="2" x:Name="frame6">
              <StackLayout>
                <Entry  x:Name="CS" Placeholder="Personal client username(optional)" BackgroundColor="#2a2a2a"
PlaceholderColor="AliceBlue" TextColor="White" FontSize="16" Text="{Binding Client}"/>
              </StackLayout>
            </Frame>
            <StackLayout Grid.Row="11" Grid.ColumnSpan="2" >
              <Button Text="Update Plan"  FontSize="15" HeightRequest="50" Grid.Row="3" x:Name="UP"
CornerRadius="80" FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked_2"/>
            </StackLayout>
```

```
            </Grid>
        </StackLayout>
    </ScrollView>
</pages:PopupPage>

```

## EditPlan.xaml.cs

```
using GymAndPAapp.Tables;
using GymAndPAapp.ViewModels;
using Rg.Plugins.Popup.Services;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class EditPlan
    {
        public string UserName = App._variable;
        public string Title { get; set; }
        public string Plan { get; set; }
        public string Description { get; set; }
        public string Fee { get; set; }
        public string Amount { get; set; }
        public string Calorie { get; set; }
        public string Sets { get; set; }
        public string Link { get; set; }
        public string Client { get; set; }
        public string OldPlan = App._variable4;
        private string temp;
        public string cost;
        string mon;
        string tue;
        string wed;
        string thur;
        string fri;
        string sat;
        string sun;
        public EditPlan(string PlanTitle, string Description, string Cost,string Calories,string Client,string Link)
        {
            OnAppearing();
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            App._variable5 = this.GetType().Name;
            InitializeComponent();

            cost = Cost;
            Sunday.IsVisible = false;
            Saturday.IsVisible = false;
            Friday.IsVisible = false;
            Thursday.IsVisible = false;
            Wednesday.IsVisible = false;
            Tuesday.IsVisible = false;
            AddButton.IsVisible = true;
            NextDayButton.IsVisible = true;
            DCI.IsVisible = false;
            LET.IsVisible = false;
            UP.IsVisible = false;
            CS.IsVisible = false;
            Fees.IsVisible = false;
            DescriptionHide.IsVisible = false;
            PlanTitleXaml.IsVisible = false;
            frame1.IsVisible = false;
```

```
            frame2.IsVisible = false;
            frame3.IsVisible = false;
            frame4.IsVisible = false;
            frame5.IsVisible = false;

            PlanTitleXaml.Text = PlanTitle;
            DescriptionHide.Text = Description;
            Fees.Text = "€" + Cost;
            if(Calories != null)
            {
                DCI.Text = Calories;
            }
            if(Client != null)
            {
                CS.Text = Client;
            }
            if (Link != null)
            {
                LET.Text = Link;
            }
        }
        protected async override void OnAppearing()
        {
            base.OnAppearing();
            var result = await FireAssistant.GetExcercises();
            picker2.ItemsSource = result;
        }
        private void Button_Clicked(object sender, EventArgs e) // add exercise
        {

            temp += Plan + " " + picker1.SelectedItem + " Sets of " + picker.SelectedItem + " Reps ,";
        }

        private async void Button_Clicked_1(object sender, EventArgs e) //new day
        {
            if (App._variable9 == 0)
            {
                if (temp == null)
                {
                    mon = "Rest,";
                    App._variable9++;
                    temp = null;
                    Monday.IsVisible = false;
                    Tuesday.IsVisible = true;
                }
                else
                {
                    mon = temp;
                    App._variable9++;
                    temp = null;
                    Monday.IsVisible = false;
                    Tuesday.IsVisible = true;
                }
            }
            else if (App._variable9 == 1)
            {
                if (temp == null)
                {
                    tue = "Rest,";
                    App._variable9++;
                    temp = null;
                    Tuesday.IsVisible = false;
                    Wednesday.IsVisible = true;
                }
                else
                {
                    tue = temp;
                    App._variable9++;
                    temp = null;
                    Tuesday.IsVisible = false;
                    Wednesday.IsVisible = true;
```

```
133.            }
134.        }
135.        else if (App._variable9 == 2)
136.        {
137.            if (temp == null)
138.            {
139.                wed = "Rest,";
140.                App._variable9++;
141.                temp = null;
142.                Wednesday.IsVisible = false;
143.                Thursday.IsVisible = true;
144.            }
145.            else
146.            {
147.                wed = temp;
148.                App._variable9++;
149.                temp = null;
150.                Wednesday.IsVisible = false;
151.                Thursday.IsVisible = true;
152.            }
153.        }
154.        else if (App._variable9 == 3)
155.        {
156.            if (temp == null)
157.            {
158.                thur = "Rest,";
159.                App._variable9++;
160.                temp = null;
161.                Thursday.IsVisible = false;
162.                Friday.IsVisible = true;
163.            }
164.            else
165.            {
166.                thur = temp;
167.                App._variable9++;
168.                temp = null;
169.                Thursday.IsVisible = false;
170.                Friday.IsVisible = true;
171.            }
172.        }
173.        else if (App._variable9 == 4)
174.        {
175.            if (temp == null)
176.            {
177.                fri = "Rest,";
178.                App._variable9++;
179.                temp = null;
180.                Friday.IsVisible = false;
181.                Saturday.IsVisible = true;
182.            }
183.            else
184.            {
185.                fri = temp;
186.                App._variable9++;
187.                temp = null;
188.                Friday.IsVisible = false;
189.                Saturday.IsVisible = true;
190.            }
191.        }
192.        else if (App._variable9 == 5)
193.        {
194.            if (temp == null)
195.            {
196.                sat = "Rest,";
197.                App._variable9++;
198.                temp = null;
199.                Saturday.IsVisible = false;
200.                Sunday.IsVisible = true;
201.            }
202.            else
203.            {
```

```
204.                 sat = temp;
205.                 App._variable9++;
206.                 temp = null;
207.                 Saturday.IsVisible = false;
208.                 Sunday.IsVisible = true;
209.             }
210.         }
211.         else if (App._variable9 == 6)
212.         {
213.             if (temp == null)
214.             {
215.                 sun = "Rest,";
216.                 App._variable9++;
217.                 temp = null;
218.                 Sunday.IsVisible = false;
219.                 DCI.IsVisible = true;
220.                 LET.IsVisible = true;
221.                 UP.IsVisible = true;
222.                 CS.IsVisible = true;
223.                 Fees.IsVisible = true;
224.                 DescriptionHide.IsVisible = true;
225.                 PlanTitleXaml.IsVisible = true;
226.                 frame1.IsVisible = true;
227.                 frame2.IsVisible = true;
228.                 frame3.IsVisible = true;
229.                 frame4.IsVisible = true;
230.                 frame5.IsVisible = true;
231.                 picker.IsVisible = false;
232.                 picker1.IsVisible = false;
233.                 picker2.IsVisible = false;
234.                 AddButton.IsVisible = false;
235.                 NextDayButton.IsVisible = false;
236.             }
237.             else
238.             {
239.                 sun = temp;
240.                 App._variable9++;
241.                 temp = null;
242.                 Sunday.IsVisible = false;
243.                 DCI.IsVisible = true;
244.                 LET.IsVisible = true;
245.                 UP.IsVisible = true;
246.                 CS.IsVisible = true;
247.                 Fees.IsVisible = true;
248.                 DescriptionHide.IsVisible = true;
249.                 PlanTitleXaml.IsVisible = true;
250.                 frame1.IsVisible = true;
251.                 frame2.IsVisible = true;
252.                 frame3.IsVisible = true;
253.                 frame4.IsVisible = true;
254.                 frame5.IsVisible = true;
255.                 picker.IsVisible = false;
256.                 picker1.IsVisible = false;
257.                 picker2.IsVisible = false;
258.                 AddButton.IsVisible = false;
259.                 NextDayButton.IsVisible = false;
260.             }
261.         }
262.     }
263.
264.     private async void Button_Clicked_2(object sender, EventArgs e) // complete
265.     {
266.         Title = PlanTitleXaml.Text;
267.         Description = DescriptionHide.Text;
268.         Calorie = DCI.Text;
269.         Link = LET.Text;
270.         Fee = Fees.Text;
271.         Client = CS.Text;
272.         if (string.IsNullOrEmpty(Title) || string.IsNullOrEmpty(Description))
273.             await App.Current.MainPage.DisplayAlert("Empty Values", "All fields required", "OK");
274.         else
```

```
275.            {
276.                var update = await FireAssistant.UpdatePlan(UserName, mon, tue, wed, thur, fri, sat, sun, Title, OldPlan,
    Description, Calorie, cost, Client,Link);
277.                if (update)
278.                {
279.                    App._variable9 = 0;
280.                    await App.Current.MainPage.DisplayAlert("Plan updated", "Success", "Ok");
281.                    await PopupNavigation.Instance.PopAllAsync();
282.                    await App.Current.MainPage.Navigation.PushAsync(new TrainerPlanPage());
283.                }
284.            }
285.        }
286.
287.        private void picker2_SelectedIndexChanged(object sender, EventArgs e)
288.        {
289.            Picker picker = (Picker)sender;
290.            var selected = picker.SelectedItem;
291.            Exercises ex = selected as Exercises;
292.            Plan = ex.name;
293.        }
294.    }
295. }
296.
```

## Payments.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.                xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.                xmlns:Syncfusion="clr-namespace:Syncfusion.SfDataGrid.XForms;assembly=Syncfusion.SfDataGrid.XForms"
5.                x:Class="GymAndPAapp.Views.Payments">
6.      <ContentPage.Content>
7.          <Syncfusion:SfDataGrid x:Name="paymentlist"
8.              ColumnSizer="Star"
9.              AutoGenerateColumns="False"
10.             ItemsSource="{Binding Payments}"
11.             AllowSorting="True"
12.             AllowMultiSorting="False"
13.             AllowPullToRefresh="True">
14.
15.             <Syncfusion:SfDataGrid.Columns x:TypeArguments="syncfusion:Columns">
16.                 <Syncfusion:GridTextColumn HeaderText="ClientName"
17.                                 MappingName="ClientName" />
18.                 <Syncfusion:GridTextColumn HeaderText="PlanTitle"
19.                                 MappingName="PlanTitle" />
20.                 <Syncfusion:GridTextColumn HeaderText="Date"
21.                                 MappingName="Date" />
22.                 <Syncfusion:GridTextColumn HeaderText="Amount"
23.                                 MappingName="Amount" />
24.             </Syncfusion:SfDataGrid.Columns>
25.         </Syncfusion:SfDataGrid>
26.     </ContentPage.Content>
27. </ContentPage>
28.
```

## Payments.xaml.cs

```
1.  using GymAndPAapp.Tables;
2.  using GymAndPAapp.ViewModels;
3.  using Syncfusion.SfDataGrid.XForms;
4.  using System;
5.  using System.Collections.Generic;
6.  using System.Linq;
7.  using System.Text;
8.  using System.Threading.Tasks;
9.
10. using Xamarin.Forms;
```

```
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class Payments : ContentPage
    {
        public Payments()
        {
            App._variable5 = this.GetType().Name;
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
        }

        protected async override void OnAppearing()
        {
            base.OnAppearing();
            var allPayments = await FireAssistant.GetPayments(App._variable);
            paymentlist.ItemsSource = allPayments;
        }
    }
}

```

## PlanRequests.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="GymAndPAapp.Views.Trainer.PlanRequests"
             BackgroundColor="#002d40">
    <ContentPage.Content>
        <StackLayout BackgroundColor="#002d40" Padding="12">
            <Grid>
                <Grid.RowDefinitions>
                    <RowDefinition Height="10"/>
                    <RowDefinition Height="30"/>
                    <RowDefinition Height="20"/>
                    <RowDefinition Height="550"/>
                    <RowDefinition Height="auto"/>
                </Grid.RowDefinitions>
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="100"/>
                    <ColumnDefinition Width="250"/>
                </Grid.ColumnDefinitions>
                <StackLayout Grid.Row="1" Grid.ColumnSpan="3">
                    <Label Text="Custom Plan Requests" TextColor="white"  FontSize="23" HorizontalOptions="Center"
FontAttributes="Bold" Padding="20,0,20,0"/>
                </StackLayout>
                <StackLayout Grid.Row="2" Grid.ColumnSpan="3">
                    <Label Grid.Row="2" TextColor="White" Grid.Column="0" Text="    Username          Goal          "
FontSize="18" FontAttributes="Bold"/>
                </StackLayout>
                <Frame BackgroundColor="White" Padding="2" Grid.Row="3" Grid.ColumnSpan="3">
                    <ListView x:Name="planlist" ItemTapped="planlist_ItemTapped_1" Grid.Row="3" Grid.ColumnSpan="3"
SeparatorColor="White" BackgroundColor="#002d40">
                        <ListView.ItemTemplate>
                            <DataTemplate>
                                <ViewCell>
                                    <ViewCell.View>
                                        <StackLayout >
                                            <Grid>
                                                <Grid.ColumnDefinitions>
                                                    <ColumnDefinition Width="10"/>
                                                    <ColumnDefinition Width="160"/>
                                                    <ColumnDefinition Width="250"/>
                                                </Grid.ColumnDefinitions>
                                                <Label HorizontalOptions="Start"
                                                Grid.Column="1"
```

```
                                    VerticalOptions="Center"
                                    VerticalTextAlignment="Center"
                                    TextColor="White"
                                    Text="{Binding ClientName}"/>
                                     <Label HorizontalOptions="Start"
                                    Grid.Column="2"
                                    VerticalOptions="Center"
                                    VerticalTextAlignment="Center"
                                    TextColor="White"
                                    Text="{Binding Goals}"/>
                                </Grid>
                            </StackLayout>
                        </ViewCell.View>
                    </ViewCell>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
    </Frame>
    <Button Text="Return" FontSize="15" HeightRequest="50" Grid.Row="4" Grid.ColumnSpan="3"
CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked"/>
    </Grid>
  </StackLayout>
 </ContentPage.Content>
</ContentPage>
```

## PlanRequests.xaml.cs

```
using GymAndPAapp.Tables;
using GymAndPAapp.ViewModels;
using Rg.Plugins.Popup.Services;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views.Trainer
{
  [XamlCompilation(XamlCompilationOptions.Compile)]
  public partial class PlanRequests : ContentPage
  {
    IAuthenticationService auth;
    public PlanRequests()
    {
      auth = DependencyService.Get<IAuthenticationService>();
      SetValue(NavigationPage.HasNavigationBarProperty, false);
      InitializeComponent();
    }
    protected async override void OnAppearing()
    {
      base.OnAppearing();
      var plans = await FireAssistant.GetRequests(App._variable);
      planlist.ItemsSource = plans;
    }


    private void Button_Clicked(object sender, EventArgs e)
    {
      App.Current.MainPage = new NavigationPage(new TabbedTrainer());
    }

    private async void planlist_ItemTapped_1(object sender, ItemTappedEventArgs e)
    {
      var details = e.Item as Requests;
```

```
41.             await PopupNavigation.Instance.PushAsync(new ViewRequestPage(details.ClientName, details.Height,
    details.Age, details.Goals, details.Equipment, details.Weight));
42.         }
43.     }
44. }
45.
```

## PopUpTrainerAddPlan.xaml

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <pages:PopupPage x:Class="GymAndPAapp.Views.PopUpTrainerAddPlan"
3.                   xmlns="http://xamarin.com/schemas/2014/forms"
4.                   xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
5.                   xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
6.                   xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup">
7.
8.       <ScrollView>
9.          <StackLayout Margin="12"
10.                      Padding="35"
11.                      BackgroundColor="#2a2a2a">
12.             <Grid>
13.                <Grid.RowDefinitions>
14.                   <RowDefinition Height="Auto"/>
15.                   <RowDefinition Height="Auto"/>
16.                   <RowDefinition Height="Auto"/>
17.                   <RowDefinition Height="Auto"/>
18.                   <RowDefinition Height="Auto"/>
19.                   <RowDefinition Height="Auto"/>
20.                   <RowDefinition Height="Auto"/>
21.                   <RowDefinition Height="Auto"/>
22.                   <RowDefinition Height="Auto"/>
23.                   <RowDefinition Height="Auto"/>
24.                   <RowDefinition Height="Auto"/>
25.                   <RowDefinition Height="Auto"/>
26.                </Grid.RowDefinitions>
27.                <Grid.ColumnDefinitions>
28.                   <ColumnDefinition/>
29.                   <ColumnDefinition/>
30.                </Grid.ColumnDefinitions>
31.                <StackLayout Grid.Row="0" Grid.ColumnSpan="2">
32.                   <Label Text="Monday" FontSize="23" TextColor="White" HorizontalOptions="Center"
    FontAttributes="Bold" x:Name="Monday"/>
33.                   <Label Text="Tuesday" FontSize="23" TextColor="White" HorizontalOptions="Center"
    FontAttributes="Bold" x:Name="Tuesday"/>
34.                   <Label Text="Wednesday" FontSize="23" TextColor="White" HorizontalOptions="Center"
    FontAttributes="Bold" x:Name="Wednesday"/>
35.                   <Label Text="Thursday" FontSize="23" TextColor="White" HorizontalOptions="Center"
    FontAttributes="Bold" x:Name="Thursday"/>
36.                   <Label Text="Friday" FontSize="23" TextColor="White" HorizontalOptions="Center"
    FontAttributes="Bold" x:Name="Friday"/>
37.                   <Label Text="Saturday" FontSize="23" TextColor="White" HorizontalOptions="Center"
    FontAttributes="Bold" x:Name="Saturday"/>
38.                   <Label Text="Sunday" FontSize="23" TextColor="White" HorizontalOptions="Center"
    FontAttributes="Bold" x:Name="Sunday"/>
39.                </StackLayout>
40.                <Frame Grid.Row="0" BackgroundColor="#ff4140" Padding="2" Grid.ColumnSpan="2" x:Name="frame2" >
41.                   <StackLayout >
42.                      <Entry x:Name="PlanTitle" Placeholder="Plan Title" BackgroundColor="#2a2a2a"
    PlaceholderColor="AliceBlue" FontSize="16" TextColor="White" Text="{Binding Title}"/>
43.                   </StackLayout>
44.                </Frame>
45.                <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="1"  Grid.ColumnSpan="2" x:Name="frame1" >
46.                   <Editor x:Name="DescriptionHide" Placeholder="Description" HeightRequest="180"
    VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand" BackgroundColor="#2a2a2a"
    PlaceholderColor="AliceBlue" FontSize="16" TextColor="White" Text="{Binding Description}"/>
47.                </Frame>
48.                <StackLayout Grid.Row="2" Grid.ColumnSpan="2">
49.                   <Picker x:Name="picker2"
50.                           Title="Select Exercise"
```

```
                    TextColor="White"
                    TitleColor="White"
                    SelectedIndexChanged="picker2_SelectedIndexChanged"
                    ItemDisplayBinding="{Binding name}">
                </Picker>
            </StackLayout>
            <StackLayout Grid.Row="3" Grid.Column="1">
                <Picker x:Name="picker"
                    Title="Select Reps"
                    TextColor="White"
                    TitleColor="White"
                    SelectedItem="{Binding Amount}">
                    <Picker.ItemsSource>
                        <x:Array Type="{x:Type x:String}">
                            <x:String>1</x:String>
                            <x:String>2</x:String>
                            <x:String>3</x:String>
                            <x:String>4</x:String>
                            <x:String>5</x:String>
                            <x:String>6</x:String>
                            <x:String>7</x:String>
                            <x:String>8</x:String>
                            <x:String>9</x:String>
                            <x:String>10</x:String>
                            <x:String>11</x:String>
                            <x:String>12</x:String>
                            <x:String>13</x:String>
                            <x:String>14</x:String>
                            <x:String>15</x:String>
                            <x:String>16</x:String>
                            <x:String>17</x:String>
                            <x:String>18</x:String>
                            <x:String>19</x:String>
                            <x:String>20</x:String>
                            <x:String>21</x:String>
                            <x:String>22</x:String>
                            <x:String>23</x:String>
                            <x:String>24</x:String>
                            <x:String>25</x:String>
                            <x:String>26</x:String>
                            <x:String>27</x:String>
                            <x:String>28</x:String>
                            <x:String>29</x:String>
                            <x:String>30</x:String>
                            <x:String>31</x:String>
                            <x:String>32</x:String>
                            <x:String>33</x:String>
                            <x:String>34</x:String>
                            <x:String>35</x:String>
                            <x:String>36</x:String>
                            <x:String>37</x:String>
                            <x:String>38</x:String>
                            <x:String>39</x:String>
                            <x:String>40</x:String>
                            <x:String>41</x:String>
                            <x:String>42</x:String>
                            <x:String>43</x:String>
                            <x:String>44</x:String>
                            <x:String>45</x:String>
                            <x:String>46</x:String>
                            <x:String>47</x:String>
                            <x:String>48</x:String>
                            <x:String>49</x:String>
                            <x:String>50</x:String>
                        </x:Array>
                    </Picker.ItemsSource>
                </Picker>
            </StackLayout>
            <StackLayout Grid.Row="3" Grid.Column="0">
                <Picker x:Name="picker1"
                    Title="Select Sets"
```

```
                    TextColor="White"
                    TitleColor="White"
                    SelectedItem="{Binding Sets}">
                <Picker.ItemsSource>
                    <x:Array Type="{x:Type x:String}">
                        <x:String>1</x:String>
                        <x:String>2</x:String>
                        <x:String>3</x:String>
                        <x:String>4</x:String>
                        <x:String>5</x:String>
                    </x:Array>
                </Picker.ItemsSource>
            </Picker>
        </StackLayout>
        <StackLayout Grid.Row="4" Grid.Column="0" HorizontalOptions="Center" >
            <Button Text="Add"  FontSize="15" HeightRequest="50" x:Name="AddButton" Grid.Row="3" CornerRadius="80" FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked"/>
        </StackLayout>
        <StackLayout Grid.Row="4" Grid.Column="1" HorizontalOptions="Center" >
            <Button Text="Next Day"  FontSize="15" HeightRequest="50" x:Name="NextDayButton" Grid.Row="3" CornerRadius="80" FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked_1"/>
        </StackLayout>
        <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="7" Grid.ColumnSpan="2" x:Name="frame3">
            <StackLayout>
                <Entry x:Name="Fees" Placeholder="Fee €" TextColor="White" BackgroundColor="#2a2a2a" PlaceholderColor="AliceBlue" FontSize="16" Text="{Binding Fee}"/>
            </StackLayout>
        </Frame>
        <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="8" Grid.ColumnSpan="2" x:Name="frame4">
            <StackLayout>
                <Entry x:Name="DCI" Placeholder="Daily Calorie intake" TextColor="White" BackgroundColor="#2a2a2a" PlaceholderColor="AliceBlue" FontSize="16" Text="{Binding Calorie}"/>
            </StackLayout>
        </Frame>
        <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="9" Grid.ColumnSpan="2" x:Name="frame5">
            <StackLayout>
                <Entry  x:Name="LET" Placeholder="Link to Excercise Tutorials" BackgroundColor="#2a2a2a" PlaceholderColor="AliceBlue" TextColor="White" FontSize="16" Text="{Binding Link}"/>
            </StackLayout>
        </Frame>
        <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="10" Grid.ColumnSpan="2" x:Name="frame6">
            <StackLayout>
                <Entry  x:Name="CS" Placeholder="Personal client username(optional)" BackgroundColor="#2a2a2a" PlaceholderColor="AliceBlue" TextColor="White" FontSize="16" Text="{Binding Client}"/>
            </StackLayout>
        </Frame>
        <StackLayout Grid.Row="11" Grid.ColumnSpan="2" >
            <Button Text="Create Plan"  FontSize="15" HeightRequest="50" Grid.Row="3" x:Name="CP" CornerRadius="80" FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked_2"/>
        </StackLayout>
    </Grid>
  </StackLayout>
 </ScrollView>
</pages:PopupPage>

```

## PopUpTrainerAddPlan.xaml.cs

```
using GymAndPAapp.Tables;
using GymAndPAapp.ViewModels;
using Newtonsoft.Json;
using Rg.Plugins.Popup.Services;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.Net.Http;
using System.Threading.Tasks;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;
```

```
12.
13.  namespace GymAndPAapp.Views
14.  {
15.     [XamlCompilation(XamlCompilationOptions.Compile)]
16.     public partial class PopUpTrainerAddPlan
17.     {
18.        public string UserName = App._variable;
19.        public string Title { get; set; }
20.        public string Plan { get; set; }
21.        public string Description { get; set; }
22.        public string Fee { get; set; }
23.        public string Amount { get; set; }
24.        public string Calorie { get; set; }
25.        public string Sets { get; set; }
26.        public string Link { get; set; }
27.        public string Client { get; set; }
28.        private string temp;
29.        string mon;
30.        string tue;
31.        string wed;
32.        string thur;
33.        string fri;
34.        string sat;
35.        string sun;
36.        public PopUpTrainerAddPlan()
37.        {
38.           OnAppearing();
39.           SetValue(NavigationPage.HasNavigationBarProperty, false);
40.           App._variable5 = this.GetType().Name;
41.           InitializeComponent();
42.
43.           Sunday.IsVisible = false;
44.           Saturday.IsVisible = false;
45.           Friday.IsVisible = false;
46.           Thursday.IsVisible = false;
47.           Wednesday.IsVisible = false;
48.           Tuesday.IsVisible = false;
49.           AddButton.IsVisible = true;
50.           NextDayButton.IsVisible = true;
51.           DCI.IsVisible = false;
52.           LET.IsVisible = false;
53.           CP.IsVisible = false;
54.           CS.IsVisible = false;
55.           Fees.IsVisible = false;
56.           DescriptionHide.IsVisible = false;
57.           PlanTitle.IsVisible = false;
58.           frame1.IsVisible = false;
59.           frame2.IsVisible = false;
60.           frame3.IsVisible = false;
61.           frame4.IsVisible = false;
62.           frame5.IsVisible = false;
63.           frame6.IsVisible = false;
64.        }
65.
66.        protected async override void OnAppearing()
67.        {
68.           base.OnAppearing();
69.           var result = await FireAssistant.GetExcercises();
70.           picker2.ItemsSource = result;
71.        }
72.
73.        private void Button_Clicked(object sender, EventArgs e) // add exercise
74.        {
75.
76.           temp += Plan + " " + picker1.SelectedItem + " Sets of " + picker.SelectedItem + " Reps ,";
77.        }
78.
79.        private async void Button_Clicked_1(object sender, EventArgs e) //new day
80.        {
81.           if (App._variable9 == 0)
82.           {
```

```
83.             if (temp == null)
84.             {
85.                 mon = "Rest,";
86.                 App._variable9++;
87.                 temp = null;
88.                 Monday.IsVisible = false;
89.                 Tuesday.IsVisible = true;
90.             }
91.             else
92.             {
93.                 mon = temp;
94.                 App._variable9++;
95.                 temp = null;
96.                 Monday.IsVisible = false;
97.                 Tuesday.IsVisible = true;
98.             }
99.         }
100.        else if (App._variable9 == 1)
101.        {
102.            if (temp == null)
103.            {
104.                tue = "Rest,";
105.                App._variable9++;
106.                temp = null;
107.                Tuesday.IsVisible = false;
108.                Wednesday.IsVisible = true;
109.            }
110.            else
111.            {
112.                tue = temp;
113.                App._variable9++;
114.                temp = null;
115.                Tuesday.IsVisible = false;
116.                Wednesday.IsVisible = true;
117.            }
118.        }
119.        else if (App._variable9 == 2)
120.        {
121.            if (temp == null)
122.            {
123.                wed = "Rest,";
124.                App._variable9++;
125.                temp = null;
126.                Wednesday.IsVisible = false;
127.                Thursday.IsVisible = true;
128.            }
129.            else
130.            {
131.                wed = temp;
132.                App._variable9++;
133.                temp = null;
134.                Wednesday.IsVisible = false;
135.                Thursday.IsVisible = true;
136.            }
137.        }
138.        else if (App._variable9 == 3)
139.        {
140.            if (temp == null)
141.            {
142.                thur = "Rest,";
143.                App._variable9++;
144.                temp = null;
145.                Thursday.IsVisible = false;
146.                Friday.IsVisible = true;
147.            }
148.            else
149.            {
150.                thur = temp;
151.                App._variable9++;
152.                temp = null;
153.                Thursday.IsVisible = false;
```

```
154.                 Friday.IsVisible = true;
155.             }
156.         }
157.         else if (App._variable9 == 4)
158.         {
159.             if (temp == null)
160.             {
161.                 fri = "Rest,";
162.                 App._variable9++;
163.                 temp = null;
164.                 Friday.IsVisible = false;
165.                 Saturday.IsVisible = true;
166.             }
167.             else
168.             {
169.                 fri = temp;
170.                 App._variable9++;
171.                 temp = null;
172.                 Friday.IsVisible = false;
173.                 Saturday.IsVisible = true;
174.             }
175.         }
176.         else if (App._variable9 == 5)
177.         {
178.             if (temp == null)
179.             {
180.                 sat = "Rest,";
181.                 App._variable9++;
182.                 temp = null;
183.                 Saturday.IsVisible = false;
184.                 Sunday.IsVisible = true;
185.             }
186.             else
187.             {
188.                 sat = temp;
189.                 App._variable9++;
190.                 temp = null;
191.                 Saturday.IsVisible = false;
192.                 Sunday.IsVisible = true;
193.             }
194.         }
195.         else if (App._variable9 == 6)
196.         {
197.             if (temp == null)
198.             {
199.                 sun = "Rest,";
200.                 App._variable9++;
201.                 temp = null;
202.                 Sunday.IsVisible = false;
203.                 DCI.IsVisible = true;
204.                 LET.IsVisible = true;
205.                 CP.IsVisible = true;
206.                 CS.IsVisible = true;
207.                 Fees.IsVisible = true;
208.                 DescriptionHide.IsVisible = true;
209.                 PlanTitle.IsVisible = true;
210.                 frame1.IsVisible = true;
211.                 frame2.IsVisible = true;
212.                 frame3.IsVisible = true;
213.                 frame4.IsVisible = true;
214.                 frame5.IsVisible = true;
215.                 frame6.IsVisible = true;
216.                 picker.IsVisible = false;
217.                 picker1.IsVisible = false;
218.                 picker2.IsVisible = false;
219.                 AddButton.IsVisible = false;
220.                 NextDayButton.IsVisible = false;
221.             }
222.             else
223.             {
224.                 sun = temp;
```

```
                App._variable9++;
                temp = null;
                Sunday.IsVisible = false;
                DCI.IsVisible = true;
                LET.IsVisible = true;
                CP.IsVisible = true;
                CS.IsVisible = true;
                Fees.IsVisible = true;
                DescriptionHide.IsVisible = true;
                PlanTitle.IsVisible = true;
                frame1.IsVisible = true;
                frame2.IsVisible = true;
                frame3.IsVisible = true;
                frame4.IsVisible = true;
                frame5.IsVisible = true;
                frame6.IsVisible = true;
                picker.IsVisible = false;
                picker1.IsVisible = false;
                picker2.IsVisible = false;
                AddButton.IsVisible = false;
                NextDayButton.IsVisible = false;
            }
        }
    }

    private async void Button_Clicked_2(object sender, EventArgs e) // complete
    {
        Title = PlanTitle.Text;
        Description = DescriptionHide.Text;
        Calorie = DCI.Text;
        Link = LET.Text;
        Fee = Fees.Text;
        Client = CS.Text;
        if (string.IsNullOrEmpty(Title) || string.IsNullOrEmpty(Description))
            await App.Current.MainPage.DisplayAlert("Empty Values", "All fields required", "OK");
        else
        {
            var plans = await FireAssistant.AddPlan(UserName, Title, mon, tue, wed, thur, fri, sat, sun, Description,
Calorie, Link, Fee, Client);
            if (plans)
            {
                App._variable9 = 0;
                await App.Current.MainPage.DisplayAlert("Success", "Plan added successfully", "Ok");
                await PopupNavigation.Instance.PopAllAsync();
                await App.Current.MainPage.Navigation.PushAsync(new TrainerPlanPage());

            }
            else
            {
                App._variable9 = 0;
                await App.Current.MainPage.DisplayAlert("ERROR", "Plan was not added, Please try again", "Ok");
                await PopupNavigation.Instance.PopAllAsync();
                await App.Current.MainPage.Navigation.PushAsync(new TrainerPlanPage());
            }
        }
    }

    private void picker2_SelectedIndexChanged(object sender, EventArgs e)
    {
        Picker picker = (Picker)sender;
        var selected = picker.SelectedItem;
        Exercises ex = selected as Exercises;
        Plan = ex.name;
    }
  }
}
```

## TabbedTrainer.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <TabbedPage xmlns="http://xamarin.com/schemas/2014/forms"
3.              xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml" xmlns:views="clr-namespace:GymAndPAapp.Views"
4.              x:Class="GymAndPAapp.Views.TabbedTrainer"
5.              xmlns:android="clr-
    namespace:Xamarin.Forms.PlatformConfiguration.AndroidSpecific;assembly=Xamarin.Forms.Core"
6.              android:TabbedPage.ToolbarPlacement="Bottom"
7.              android:TabbedPage.BarItemColor ="white"
8.              android:TabbedPage.BarSelectedItemColor ="#ff4140">
9.
10.     <!--Pages can be added as references or inline-->
11.
12.     <views:Payments Title="Payments"></views:Payments>
13.     <views:TrainerHome Title="Home"></views:TrainerHome>
14.     <views:Settings Title="Settings"></views:Settings>
15.
16. </TabbedPage>
17.
```

## TabbedTrainer.xaml.cs

```
1.  using Xamarin.Forms;
2.  using Xamarin.Forms.Xaml;
3.  using Xamarin.Forms.PlatformConfiguration.AndroidSpecific;
4.  using TabbedPage = Xamarin.Forms.TabbedPage;
5.
6.  namespace GymAndPAapp.Views
7.  {
8.      [XamlCompilation(XamlCompilationOptions.Compile)]
9.      public partial class TabbedTrainer : TabbedPage
10.     {
11.         public TabbedTrainer()
12.         {
13.             SetValue(NavigationPage.HasNavigationBarProperty, false);
14.             InitializeComponent();
15.             CurrentPage = Children[1];
16.
17.         }
18.
19.     }
20. }
21.
```

## TrainerHome.xaml

```
1.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
2.               xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
3.               xmlns:Image="clr-namespace:ImageCircle.Forms.Plugin.Abstractions;assembly=ImageCircle.Forms.Plugin"
4.               x:Class="GymAndPAapp.Views.TrainerHome">
5.      <ContentPage.Content>
6.          <ScrollView BackgroundColor ="#002d40">
7.              <StackLayout>
8.                  <Grid VerticalOptions="Center" Margin="30"  RowSpacing="40">
9.                      <Grid.RowDefinitions>
10.                         <RowDefinition Height="0"/>
11.                         <RowDefinition Height="220"/>
12.                         <RowDefinition Height="60"/>
13.                         <RowDefinition Height="50"/>
14.                         <RowDefinition Height="50"/>
15.                         <RowDefinition Height="50"/>
16.                     </Grid.RowDefinitions>
17.                     <StackLayout Grid.Row="1">
18.                         <Image:CircleImage x:Name="imgChoose" Source="stock.jpg" HeightRequest="300"
    WidthRequest="300" Aspect="AspectFill">
```

```
                        <Image.GestureRecognizers>
                            <TapGestureRecognizer
                                Tapped="TapGestureRecognizer_Tapped"
                                NumberOfTapsRequired="1"/>
                        </Image.GestureRecognizers>
                    </Image:CircleImage>
                    <ActivityIndicator Grid.Row="1" x:Name="loading" IsRunning="False" Color="#ff4140"/>
                </StackLayout>
                <StackLayout Grid.Row="3">
                    <Button Text="Clients" FontSize="15"  HeightRequest="50" Grid.Row="2" CornerRadius="80"
FontAttributes="Bold" TextColor="white" Background="#ff4140" Clicked="Button_Clicked"/>
                </StackLayout>
                <StackLayout Grid.Row="4">
                    <Button Text="Plans" FontSize="15" HeightRequest="50" Grid.Row="3" CornerRadius="80"
FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked_1"/>
                </StackLayout>
                <StackLayout Grid.Row="5">
                    <Button Text="Requests" FontSize="15" HeightRequest="50" Grid.Row="3" CornerRadius="80"
FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked_2"/>
                </StackLayout>
            </Grid>
        </StackLayout>
    </ScrollView>
  </ContentPage.Content>
</ContentPage>

```

## TrainerHome.xaml.cs

```
using Firebase.Storage;
using GymAndPAapp.ViewModels;
using GymAndPAapp.Views.Shared;
using GymAndPAapp.Views.Trainer;
using GymAndPAapp.Views.Trainers;
using Rg.Plugins.Popup.Services;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Net;
using System.Text;
using System.Threading.Tasks;
using Xamarin.Essentials;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp.Views
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class TrainerHome : ContentPage
    {
        IAuthenticationService auth;
        public TrainerHome()
        {
            auth = DependencyService.Get<IAuthenticationService>();
            App._variable5 = this.GetType().Name;
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
        }

        protected async override void OnAppearing()
        {
            var storage = await Permissions.CheckStatusAsync<Permissions.StorageRead>();
            var camera = await Permissions.CheckStatusAsync<Permissions.Camera>();

            if (storage != PermissionStatus.Granted)
            {
                storage = await Permissions.RequestAsync<Permissions.StorageRead>();
            }
```

```
41.             if (storage != PermissionStatus.Granted)
42.             {
43.                 return;
44.             }
45.             if (camera != PermissionStatus.Granted)
46.             {
47.                 camera = await Permissions.RequestAsync<Permissions.Camera>();
48.             }
49.             if (camera != PermissionStatus.Granted)
50.             {
51.                 return;
52.             }
53.             var info = auth.GetUser();
54.             var details = await FireAssistant.GetTrainer(info);
55.             App._variable = details.UserName;
56.             var name = App._variable;
57.             base.OnAppearing();
58.             loading.IsRunning = true;
59.             if (App.profile == null)
60.             {
61.                 try
62.                 {
63.                     var webClient = new WebClient();
64.                     var getimage = await new FirebaseStorage("gymanalyticsapp.appspot.com")
65.                     .Child("ProfilePics")
66.                     .Child(name + ".jpeg")
67.                     .GetDownloadUrlAsync();
68.                     string imgurl = getimage;
69.                     byte[] imgbyte = webClient.DownloadData(imgurl);
70.                     App.profile = ImageSource.FromStream(() => new MemoryStream(imgbyte));
71.                     imgChoose.Source = ImageSource.FromStream(() => new MemoryStream(imgbyte));
72.                     loading.IsRunning = false;
73.                 }
74.                 catch (Exception e)
75.                 {
76.                     imgChoose.Source = "stock.jpg";
77.                     loading.IsRunning = false;
78.                 }
79.             }
80.             else
81.             {
82.                 imgChoose.Source = App.profile;
83.                 loading.IsRunning = false;
84.             }
85.         }
86.
87.         private void Button_Clicked(object sender, EventArgs e)
88.         {
89.             App.Current.MainPage = new NavigationPage(new TrainersClients());
90.         }
91.
92.         private void Button_Clicked_1(object sender, EventArgs e)
93.         {
94.             App.Current.MainPage = new NavigationPage(new TrainerPlanPage());
95.         }
96.         private async void TapGestureRecognizer_Tapped(object sender, EventArgs e)
97.         {
98.             await PopupNavigation.Instance.PushAsync(new ProfilePicOptions());
99.              OnPropertyChanged();
100.        }
101.
102.        private void Button_Clicked_2(object sender, EventArgs e)
103.        {
104.            App.Current.MainPage = new NavigationPage(new PlanRequests());
105.        }
106.    }
107. }
108.
```

## TrainerPlanPage.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.               xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.               x:Class="GymAndPAapp.Views.TrainerPlanPage">
5.     <ContentPage.Content>
6.        <StackLayout BackgroundColor="#002d40">
7.           <Grid>
8.              <Grid.RowDefinitions>
9.              <RowDefinition Height="10"/>
10.             <RowDefinition Height="20"/>
11.             <RowDefinition Height="20"/>
12.             <RowDefinition Height="500"/>
13.             <RowDefinition Height="auto"/>
14.             <RowDefinition Height="auto"/>
15.             </Grid.RowDefinitions>
16.          <Grid.ColumnDefinitions>
17.             <ColumnDefinition Width="100"/>
18.             <ColumnDefinition Width="250"/>
19.          </Grid.ColumnDefinitions>
20.          <StackLayout Grid.Row="1" Grid.ColumnSpan="3">
21.             <Label Text="Current Plans" TextColor="white"  FontSize="23" HorizontalOptions="Center"
    FontAttributes="Bold" Padding="20,0,20,0"/>
22.          </StackLayout>
23.          <StackLayout Grid.Row="2" Grid.ColumnSpan="3">
24.             <Label Grid.Row="2" TextColor="White" Grid.Column="0" Text="   Title                  Discription
    " FontSize="18" FontAttributes="Bold"/>
25.          </StackLayout>
26.          <Frame BackgroundColor="White" Padding="2" Grid.Row="3" Grid.ColumnSpan="3">
27.                <ListView x:Name="planlist" ItemTapped="Planlist_ItemTapped" SeparatorColor="White"
    BackgroundColor="#002d40">
28.                   <ListView.ItemTemplate>
29.                      <DataTemplate>
30.                         <ViewCell>
31.                            <ViewCell.View>
32.                               <StackLayout>
33.                                  <Grid>
34.                                     <Grid.ColumnDefinitions>
35.                                        <ColumnDefinition Width="10"/>
36.                                        <ColumnDefinition Width="140"/>
37.                                        <ColumnDefinition Width="250"/>
38.                                     </Grid.ColumnDefinitions>
39.                                     <Label HorizontalOptions="Start"
40.                                    Grid.Column="1"
41.                                    VerticalOptions="Center"
42.                                    VerticalTextAlignment="Center"
43.                                    TextColor="White"
44.                                    Text="{Binding PlanTitle}"/>
45.                                     <Label HorizontalOptions="Start"
46.                                    Grid.Column="2"
47.                                    VerticalOptions="Center"
48.                                    VerticalTextAlignment="Center"
49.                                    TextColor="White"
50.                                    Text="{Binding Description}"/>
51.                                  </Grid>
52.                               </StackLayout>
53.                            </ViewCell.View>
54.                         </ViewCell>
55.                      </DataTemplate>
56.                   </ListView.ItemTemplate>
57.                </ListView>
58.             </Frame>
59.             <Button Text="Add Plan"   FontSize="15" HeightRequest="50" Grid.Row="4" Grid.ColumnSpan="3"
    CornerRadius="80" FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked_1"/>
60.             <Button Text="Return"  FontSize="15" HeightRequest="50" Grid.Row="5" Grid.ColumnSpan="3"
    CornerRadius="80" FontAttributes="Bold" TextColor="white" Background="#ff4140"  Clicked="Button_Clicked"/>
61.          </Grid>
62.       </StackLayout>
63.    </ContentPage.Content>
```

```
64.  </ContentPage>
65.
```

## TrainerPlanPage.xaml.cs

```
1.  using GymAndPAapp.Tables;
2.  using GymAndPAapp.ViewModels;
3.  using Rg.Plugins.Popup.Services;
4.  using System;
5.  using Xamarin.Forms;
6.  using Xamarin.Forms.Xaml;
7.
8.  namespace GymAndPAapp.Views
9.  {
10.     [XamlCompilation(XamlCompilationOptions.Compile)]
11.     public partial class TrainerPlanPage : ContentPage
12.     {
13.         public TrainerPlanPage()
14.         {
15.             App._variable5 = this.GetType().Name;
16.             SetValue(NavigationPage.HasNavigationBarProperty, false);
17.             InitializeComponent();
18.         }
19.         protected async override void OnAppearing()
20.         {
21.             base.OnAppearing();
22.             var plans = await FireAssistant.GetPlanTrainer(App._variable);
23.             planlist.ItemsSource = plans;
24.             planlist.ItemTapped += Planlist_ItemTapped;
25.         }
26.
27.         private async void Planlist_ItemTapped(object sender, ItemTappedEventArgs e)
28.         {
29.             var details = e.Item as Plans;
30.             PopupNavigation.Instance.PushAsync(new TrainerSelectedPlan(App._variable, details.PlanTitle,
    details.Monday,details.Tuesday,details.Wednesday,details.Thursday,details.Friday,details.Saturday,details.Sunday,
    details.Description,details.Calorie, details.Fee,details.Client,details.Link));
31.         }
32.
33.         private void Button_Clicked(object sender, EventArgs e)
34.         {
35.             App.Current.MainPage = new NavigationPage(new TabbedTrainer());
36.         }
37.
38.         private void Button_Clicked_1(object sender, EventArgs e)
39.         {
40.
41.             PopupNavigation.Instance.PushAsync(new PopUpTrainerAddPlan());
42.         }
43.     }
44. }
45.
```

## TrainersClients.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3.               xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.               x:Class="GymAndPAapp.Views.Trainers.TrainersClients">
5.      <ContentPage.Content>
6.          <StackLayout BackgroundColor="#002d40" Padding="12">
7.              <Grid>
8.                  <Grid.RowDefinitions>
9.                      <RowDefinition Height="10"/>
10.                     <RowDefinition Height="30"/>
11.                     <RowDefinition Height="20"/>
12.                     <RowDefinition Height="550"/>
```

```
13.                 <RowDefinition Height="auto"/>
14.             </Grid.RowDefinitions>
15.             <Grid.ColumnDefinitions>
16.                 <ColumnDefinition Width="*"/>
17.                 <ColumnDefinition Width="*"/>
18.                 <ColumnDefinition Width="*"/>
19.             </Grid.ColumnDefinitions>
20.             <StackLayout Grid.Row="1" Grid.ColumnSpan="3">
21.                 <Label Text="Clients" TextColor="White"  FontSize="23" FontAttributes="Bold"
    HorizontalOptions="Center"/>
22.             </StackLayout>
23.             <StackLayout Grid.Row="2" Grid.ColumnSpan="3">
24.                 <Label Grid.Row="2" TextColor="White" Grid.Column="0" Text="  Username          Plan Title          "
    FontSize="18" FontAttributes="Bold"/>
25.             </StackLayout>
26.             <Frame BackgroundColor="White" Padding="2" Grid.Row="3" Grid.ColumnSpan="3">
27.                 <ListView x:Name="clientlist" IsEnabled="False" Grid.Row="3" Grid.ColumnSpan="3"
    BackgroundColor="#002d40" SeparatorColor="White">
28.                     <ListView.ItemTemplate>
29.                         <DataTemplate>
30.                             <ViewCell>
31.                                 <ViewCell.View>
32.                                     <StackLayout >
33.                                         <Grid>
34.                                             <Grid.ColumnDefinitions>
35.                                                 <ColumnDefinition Width="150"/>
36.                                                 <ColumnDefinition Width="120"/>
37.                                                 <ColumnDefinition Width="5"/>
38.                                             </Grid.ColumnDefinitions>
39.                                             <Label HorizontalOptions="Start"
40.                                             Grid.Column="0"
41.                                             VerticalOptions="Center"
42.                                             VerticalTextAlignment="Center"
43.                                             TextColor="White"
44.                                             Text="{Binding UserName}"/>
45.                                              <Label HorizontalOptions="Start"
46.                                             Grid.Column="1"
47.                                             VerticalOptions="Center"
48.                                             VerticalTextAlignment="Center"
49.                                             TextColor="White"
50.                                             Text="{Binding PlanTitle}"/>
51.                                         </Grid>
52.                                     </StackLayout>
53.                                 </ViewCell.View>
54.                             </ViewCell>
55.                         </DataTemplate>
56.                     </ListView.ItemTemplate>
57.                 </ListView>
58.             </Frame>
59.             <Button Text="Return" FontSize="15" HeightRequest="50" Grid.Row="4" Grid.ColumnSpan="3"
    CornerRadius="80" FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked"/>
60.         </Grid>
61.     </StackLayout>
62.   </ContentPage.Content>
63. </ContentPage>
64.
```

## TrainersClients.xaml.cs

```
1.  using GymAndPAapp.ViewModels;
2.  using System;
3.  using System.Collections.Generic;
4.  using System.Linq;
5.  using System.Text;
6.  using System.Threading.Tasks;
7.
8.  using Xamarin.Forms;
9.  using Xamarin.Forms.Xaml;
10.
```

```
namespace GymAndPAapp.Views.Trainers
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class TrainersClients : ContentPage
    {
        public TrainersClients()
        {
            App._variable5 = this.GetType().Name;
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
        }

        protected async override void OnAppearing()
        {
            base.OnAppearing();
            var allClients = await FireAssistant.GetTrainerClients(App._variable);
            clientlist.ItemsSource = allClients;

        }

        private void Button_Clicked(object sender, EventArgs e)
        {
            App.Current.MainPage = new NavigationPage(new TabbedTrainer());
        }
    }
}

```

## TrainerSelectedPlan.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<pages:PopupPage xmlns="http://xamarin.com/schemas/2014/forms"
                 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
                 xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
                 xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup"
                 x:Class="GymAndPAapp.Views.TrainerSelectedPlan">
    <ScrollView>
        <StackLayout Margin="12"
                     Padding="30"
                     BackgroundColor="#2a2a2a">
            <StackLayout>
                <Grid VerticalOptions="CenterAndExpand">
                    <Grid.RowDefinitions>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="60"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="100"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="100"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="100"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="100"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="100"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="100"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="100"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="100"/>
                        <RowDefinition Height="10"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="auto"/>
                        <RowDefinition Height="auto"/>
                    </Grid.RowDefinitions>
```

```
39.             <Label Grid.Row="0" x:Name="title" TextColor="white" Grid.ColumnSpan="2" FontAttributes="Bold"
    WidthRequest="330" HorizontalOptions="Center" BackgroundColor="#2a2a2a" HorizontalTextAlignment="Center"
    FontSize="23"/>
40.               <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="1"  Grid.ColumnSpan="2">
41.               <StackLayout BackgroundColor="#2a2a2a">
42.                 <Label x:Name="description" TextColor="White" FontSize="16" WidthRequest="330"
    BackgroundColor="#2a2a2a" HorizontalOptions="Center"/>
43.               </StackLayout>
44.             </Frame>
45.             <StackLayout Grid.Row="2">
46.               <Label Text="Monday" HorizontalOptions="Start" TextColor="White" FontAttributes="Bold"
    FontSize="16"/>
47.             </StackLayout>
48.             <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="3" Grid.ColumnSpan="2" >
49.               <StackLayout>
50.                 <ListView x:Name="monlist" SeparatorColor="White">
51.                   <ListView.ItemTemplate>
52.                     <DataTemplate>
53.                       <ViewCell IsEnabled="False">
54.                         <ViewCell.View>
55.                           <StackLayout>
56.                             <Label HorizontalOptions="Start"
57.                                 VerticalOptions="Center"
58.                                 VerticalTextAlignment="Center"
59.                                 BackgroundColor="#2a2a2a"
60.                                 HeightRequest="110"
61.                                 WidthRequest="330"
62.                                 TextColor="White"
63.                                 Text="{Binding}"/>
64.                           </StackLayout>
65.                         </ViewCell.View>
66.                       </ViewCell>
67.                     </DataTemplate>
68.                   </ListView.ItemTemplate>
69.                 </ListView>
70.               </StackLayout>
71.             </Frame>
72.             <StackLayout Grid.Row="4">
73.               <Label Text="Tuesday" HorizontalOptions="Start" TextColor="White" FontAttributes="Bold"
    FontSize="16"/>
74.             </StackLayout>
75.             <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="5" Grid.ColumnSpan="2">
76.               <StackLayout>
77.                 <ListView x:Name="tuelist" SeparatorColor="White">
78.                   <ListView.ItemTemplate>
79.                     <DataTemplate>
80.                       <ViewCell IsEnabled="False">
81.                         <ViewCell.View>
82.                           <StackLayout>
83.                             <Label HorizontalOptions="Start"
84.                                 VerticalOptions="Center"
85.                                 VerticalTextAlignment="Center"
86.                                 BackgroundColor="#2a2a2a"
87.                                 HeightRequest="110"
88.                                 WidthRequest="330"
89.                                 TextColor="White"
90.                                 Text="{Binding}"/>
91.                           </StackLayout>
92.                         </ViewCell.View>
93.                       </ViewCell>
94.                     </DataTemplate>
95.                   </ListView.ItemTemplate>
96.                 </ListView>
97.               </StackLayout>
98.             </Frame>
99.             <StackLayout Grid.Row="6">
100.              <Label Text="wednesday" HorizontalOptions="Start" TextColor="White" FontAttributes="Bold"
    FontSize="16"/>
101.            </StackLayout>
102.            <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="7" Grid.ColumnSpan="2">
103.              <StackLayout>
```

```
                    <ListView x:Name="wedlist" SeparatorColor="White">
                        <ListView.ItemTemplate>
                            <DataTemplate>
                                <ViewCell IsEnabled="False">
                                    <ViewCell.View>
                                        <StackLayout>
                                            <Label HorizontalOptions="Start"
                                                VerticalOptions="Center"
                                                VerticalTextAlignment="Center"
                                                BackgroundColor="#2a2a2a"
                                                HeightRequest="110"
                                                WidthRequest="330"
                                                TextColor="White"
                                                Text="{Binding}"/>
                                        </StackLayout>
                                    </ViewCell.View>
                                </ViewCell>
                            </DataTemplate>
                        </ListView.ItemTemplate>
                    </ListView>
                </StackLayout>
            </Frame>
            <StackLayout Grid.Row="8">
                <Label Text="Thursday" HorizontalOptions="Start" TextColor="White" FontAttributes="Bold"
    FontSize="16"/>
            </StackLayout>
            <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="9" Grid.ColumnSpan="2">
                <StackLayout>
                    <ListView x:Name="thurlist" SeparatorColor="White">
                        <ListView.ItemTemplate>
                            <DataTemplate>
                                <ViewCell IsEnabled="False">
                                    <ViewCell.View>
                                        <StackLayout>
                                            <Label HorizontalOptions="Start"
                                                VerticalOptions="Center"
                                                VerticalTextAlignment="Center"
                                                BackgroundColor="#2a2a2a"
                                                HeightRequest="110"
                                                WidthRequest="330"
                                                TextColor="White"
                                                Text="{Binding}"/>
                                        </StackLayout>
                                    </ViewCell.View>
                                </ViewCell>
                            </DataTemplate>
                        </ListView.ItemTemplate>
                    </ListView>
                </StackLayout>
            </Frame>
            <StackLayout Grid.Row="10">
                <Label Text="Friday" HorizontalOptions="Start" TextColor="White" FontAttributes="Bold"
    FontSize="16"/>
            </StackLayout>
            <Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="11" Grid.ColumnSpan="2">
                <StackLayout>
                    <ListView x:Name="frilist" SeparatorColor="White">
                        <ListView.ItemTemplate>
                            <DataTemplate>
                                <ViewCell IsEnabled="False">
                                    <ViewCell.View>
                                        <StackLayout>
                                            <Label HorizontalOptions="Start"
                                                VerticalOptions="Center"
                                                VerticalTextAlignment="Center"
                                                BackgroundColor="#2a2a2a"
                                                HeightRequest="110"
                                                WidthRequest="330"
                                                TextColor="White"
                                                Text="{Binding}"/>
                                        </StackLayout>
```

```
                        </ViewCell.View>
                    </ViewCell>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
    </StackLayout>
</Frame>
<StackLayout Grid.Row="12">
    <Label Text="Saturday" HorizontalOptions="Start" TextColor="White" FontAttributes="Bold"
FontSize="16"/>
</StackLayout>
<Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="13" Grid.ColumnSpan="2">
    <StackLayout>
        <ListView x:Name="satlist" SeparatorColor="White">
            <ListView.ItemTemplate>
                <DataTemplate>
                    <ViewCell IsEnabled="False">
                        <ViewCell.View>
                            <StackLayout>
                                <Label HorizontalOptions="Start"
                                    VerticalOptions="Center"
                                    VerticalTextAlignment="Center"
                                    BackgroundColor="#2a2a2a"
                                    HeightRequest="110"
                                    WidthRequest="330"
                                    TextColor="White"
                                    Text="{Binding}"/>
                            </StackLayout>
                        </ViewCell.View>
                    </ViewCell>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
    </StackLayout>
</Frame>
<StackLayout Grid.Row="14">
    <Label Text="Sunday" HorizontalOptions="Start" TextColor="White" FontAttributes="Bold"
FontSize="16"/>
</StackLayout>
<Frame BackgroundColor="#ff4140" Padding="2" Grid.Row="15" Grid.ColumnSpan="2">
    <StackLayout>
        <ListView x:Name="sunlist" SeparatorColor="White">
            <ListView.ItemTemplate>
                <DataTemplate>
                    <ViewCell IsEnabled="False">
                        <ViewCell.View>
                            <StackLayout>
                                <Label HorizontalOptions="Start"
                                    VerticalOptions="Center"
                                    VerticalTextAlignment="Center"
                                    BackgroundColor="#2a2a2a"
                                    HeightRequest="110"
                                    WidthRequest="330"
                                    TextColor="White"
                                    Text="{Binding}"/>
                            </StackLayout>
                        </ViewCell.View>
                    </ViewCell>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
    </StackLayout>
</Frame>
<StackLayout Grid.Row="17"  Grid.ColumnSpan="2">
    <Label x:Name="calorie" TextColor="White" FontAttributes="Bold" WidthRequest="330"
BackgroundColor="#2a2a2a" HorizontalOptions="Center" FontSize="16"/>
</StackLayout>
<StackLayout Grid.Row="18" Grid.ColumnSpan="2">
    <Label x:Name="client" TextColor="White" FontAttributes="Bold" WidthRequest="330"
BackgroundColor="#2a2a2a" HorizontalOptions="Center" FontSize="16"/>
</StackLayout>
```

```
240.                 <StackLayout Grid.Row="19" Grid.ColumnSpan="2">
241.                     <Label x:Name="tutorial" TextColor="White" FontAttributes="Bold" WidthRequest="330"
    BackgroundColor="#2a2a2a" HorizontalOptions="Center" FontSize="16"/>
242.                 </StackLayout>
243.                 <StackLayout Grid.Row="20" Grid.ColumnSpan="2">
244.                     <Label x:Name="fee" TextColor="White" FontAttributes="Bold" WidthRequest="330"
    BackgroundColor="#2a2a2a" HorizontalOptions="Center" FontSize="16"/>
245.                 </StackLayout>
246.                 <StackLayout Grid.Row="21" Grid.ColumnSpan="2">
247.                     <Button Text="Edit" FontSize="15" HeightRequest="50"  CornerRadius="80" FontAttributes="Bold"
    TextColor="White" Background="#ff4140"  Clicked="Button_Clicked_1"/>
248.                 </StackLayout>
249.                 <StackLayout Grid.Row="22" Grid.ColumnSpan="2">
250.                     <Button Text="Delete Plan" FontSize="15" HeightRequest="50"  CornerRadius="80"
    FontAttributes="Bold" TextColor="White" Background="#ff4140"  Clicked="Button_Clicked"/>
251.                 </StackLayout>
252.             </Grid>
253.         </StackLayout>
254.     </StackLayout>
255.   </ScrollView>
256. </pages:PopupPage>
257.
```

## TrainerSelectedPlan.xaml.cs

```
1.  using GymAndPAapp.ViewModels;
2.  using Rg.Plugins.Popup.Services;
3.  using System;
4.  using System.Collections.Generic;
5.  using System.Linq;
6.  using System.Text;
7.  using System.Threading.Tasks;
8.
9.  using Xamarin.Forms;
10. using Xamarin.Forms.Xaml;
11.
12. namespace GymAndPAapp.Views
13. {
14.     [XamlCompilation(XamlCompilationOptions.Compile)]
15.     public partial class TrainerSelectedPlan
16.     {
17.         public string cost;
18.         public string clientinfo;
19.         public string linkinfo;
20.         public string calinfo;
21.         public string monday;
22.         public string tuesday;
23.         public string wednesday;
24.         public string thursday;
25.         public string friday;
26.         public string saturday;
27.         public string sunday;
28.
29.         public TrainerSelectedPlan( string TrainerName,string PlanTitle, string Mon, string Tue,string Wed,string
    Thur,string Fri,string Sat,string Sun, string Description,string Calorie, string Fee, string Client, string Link)
30.         {
31.             SetValue(NavigationPage.HasNavigationBarProperty, false);
32.             App._variable5 = this.GetType().Name;
33.             InitializeComponent();
34.             cost = Fee;
35.             clientinfo = Client;
36.             linkinfo = Link;
37.             calinfo = Calorie;
38.             monday = Mon;
39.             tuesday = Tue;
40.             wednesday = Wed;
41.             thursday = Thur;
42.             friday = Fri;
43.             saturday = Sat;
```

```
44.             sunday = Sun;
45.
46.             client.Text = "Specific client plan: " + Client;
47.             tutorial.Text = "Link to tutorial: " +Link;
48.             title.Text = PlanTitle;
49.             description.Text = Description;
50.             List<string> MondayArray = new List<string>();
51.             if (Mon.Contains(",") == true)
52.             {
53.                 MondayArray = Mon.Split(new char[] { ',' }).ToList();
54.                 monlist.ItemsSource = MondayArray;
55.             }
56.
57.             List<string> TuesdayArray = new List<string>();
58.             if (Tue.Contains(",") == true)
59.             {
60.                 TuesdayArray = Tue.Split(new char[] { ',' }).ToList();
61.                 tuelist.ItemsSource = TuesdayArray;
62.             }
63.
64.             List<string> WednesdayArray = new List<string>();
65.             if (Wed.Contains(",") == true)
66.             {
67.                 WednesdayArray = Wed.Split(new char[] { ',' }).ToList();
68.                 wedlist.ItemsSource = WednesdayArray;
69.             }
70.
71.             List<string> ThursdayArray = new List<string>();
72.             if (Thur.Contains(",") == true)
73.             {
74.                 ThursdayArray = Thur.Split(new char[] { ',' }).ToList();
75.                 thurlist.ItemsSource = ThursdayArray;
76.             }
77.
78.             List<string> FridayArray = new List<string>();
79.             if (Fri.Contains(",") == true)
80.             {
81.                 FridayArray = Fri.Split(new char[] { ',' }).ToList();
82.                 frilist.ItemsSource = FridayArray;
83.             }
84.
85.             List<string> SaturdayArray = new List<string>();
86.             if (Sat.Contains(",") == true)
87.             {
88.                 SaturdayArray = Sat.Split(new char[] { ',' }).ToList();
89.                 satlist.ItemsSource = SaturdayArray;
90.             }
91.             List<string> SundayArray = new List<string>();
92.             if (Sun.Contains(",") == true)
93.             {
94.                 SundayArray = Sun.Split(new char[] { ',' }).ToList();
95.                 sunlist.ItemsSource = SundayArray;
96.             }
97.             calorie.Text = "Daily Calorie intake: " + Calorie +" cal" ;
98.             fee.Text = "Plan Cost: €" + Fee;
99.             App._variable4 = PlanTitle;
100.        }
101.
102.        private async void Button_Clicked_1(object sender, EventArgs e)
103.        {
104.            await PopupNavigation.Instance.PushAsync(new EditPlan(title.Text, description.Text, cost,
    calinfo,clientinfo,linkinfo));
105.        }
106.
107.        private async void Button_Clicked(object sender, EventArgs e)
108.        {
109.            var request = await FireAssistant.DeletePlan(App._variable, title.Text);
110.            if(request)
111.            {
112.                await PopupNavigation.Instance.PopAllAsync();
113.                await App.Current.MainPage.Navigation.PushAsync(new TrainerPlanPage());
```

```
114.            }
115.            else
116.            {
117.                await App.Current.MainPage.DisplayAlert("ERROR", "Plan not deleted please try again", "OK");
118.                await PopupNavigation.Instance.PopAllAsync();
119.            }
120.        }
121.    }
122. }
123.
```

## ViewRequestPage.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
2.  <pages:PopupPage xmlns="http://xamarin.com/schemas/2014/forms"
3.                   xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4.                   xmlns:animations="clr-namespace:Rg.Plugins.Popup.Animations;assembly=Rg.Plugins.Popup"
5.                   xmlns:pages="clr-namespace:Rg.Plugins.Popup.Pages;assembly=Rg.Plugins.Popup"
6.                   x:Class="GymAndPAapp.Views.Trainer.ViewRequestPage">
7.      <StackLayout Margin="15,60,15,120"
8.                   Padding="24"
9.                   BackgroundColor="#2a2a2a">
10.         <Grid>
11.             <Grid.RowDefinitions>
12.                 <RowDefinition Height="auto"/>
13.                 <RowDefinition Height="30"/>
14.                 <RowDefinition Height="auto"/>
15.                 <RowDefinition Height="auto"/>
16.                 <RowDefinition Height="auto"/>
17.                 <RowDefinition Height="auto"/>
18.                 <RowDefinition Height="100"/>
19.                 <RowDefinition Height="auto"/>
20.                 <RowDefinition Height="auto"/>
21.             </Grid.RowDefinitions>
22.             <StackLayout Grid.Row="0">
23.                 <Label x:Name="ClientName" HorizontalOptions="CenterAndExpand" FontSize="23" FontAttributes="Bold"
    TextColor="White"/>
24.             </StackLayout>
25.             <StackLayout Grid.Row="2">
26.                 <Label x:Name="ClientAge" HorizontalOptions="Start" FontSize="18" TextColor="White"/>
27.             </StackLayout>
28.             <StackLayout Grid.Row="3">
29.                 <Label x:Name="ClientHeight" HorizontalOptions="Start" FontSize="18" TextColor="White"/>
30.             </StackLayout>
31.             <StackLayout Grid.Row="4">
32.                 <Label x:Name="ClientWeight" HorizontalOptions="Start" FontSize="18" TextColor="White"/>
33.             </StackLayout>
34.             <StackLayout Grid.Row="5">
35.                 <Label x:Name="ClientGoals" HorizontalOptions="Start" FontSize="18" TextColor="White"/>
36.             </StackLayout>
37.             <StackLayout Grid.Row="6">
38.                 <Label x:Name="ClientEquip" HorizontalOptions="Start" FontSize="18" TextColor="White"/>
39.             </StackLayout>
40.             <StackLayout Grid.Row="8">
41.                 <Button Text="Completed"  FontSize="15"  HeightRequest="50" CornerRadius="80" FontAttributes="Bold"
    TextColor="White" Background="#ff4140" Clicked="Button_Clicked"/>
42.             </StackLayout>
43.             <StackLayout Grid.Row="9">
44.                 <Button Text="Begin Creating Plan"  FontSize="15"  HeightRequest="50" CornerRadius="80"
    FontAttributes="Bold" TextColor="White" Background="#ff4140" Clicked="Button_Clicked_1"/>
45.             </StackLayout>
46.         </Grid>
47.     </StackLayout>
48. </pages:PopupPage>
49.
```

## ViewRequestPage.xaml.cs

```
1.  using GymAndPAapp.ViewModels;
2.  using Rg.Plugins.Popup.Services;
3.  using System;
4.  using System.Collections.Generic;
5.  using System.Linq;
6.  using System.Text;
7.  using System.Threading.Tasks;
8.
9.  using Xamarin.Forms;
10. using Xamarin.Forms.Xaml;
11.
12. namespace GymAndPAapp.Views.Trainer
13. {
14.     [XamlCompilation(XamlCompilationOptions.Compile)]
15.     public partial class ViewRequestPage
16.     {
17.         public string Name;
18.         public string height;
19.         public string weight;
20.         public string Age;
21.         public string Goals;
22.         public string Equipment;
23.
24.
25.         public ViewRequestPage(string clientName, string clientheight, string age, string goals, string equipment, string
    clientweight)
26.         {
27.             SetValue(NavigationPage.HasNavigationBarProperty, false);
28.             InitializeComponent();
29.             Name = clientName;
30.             Age = age;
31.             Equipment = equipment;
32.             height = clientheight;
33.             weight = clientweight;
34.             Goals = goals;
35.         }
36.         protected async override void OnAppearing()
37.         {
38.             base.OnAppearing();
39.             ClientName.Text = Name;
40.             ClientAge.Text = "Client age: " + Age;
41.             ClientEquip.Text = "Equipment access: " + Equipment;
42.             ClientHeight.Text = "Client height: " + height;
43.             ClientWeight.Text = "Client weight: " + weight;
44.             ClientGoals.Text = "Client goals: " + Goals;
45.         }
46.
47.             private async void Button_Clicked(object sender, EventArgs e)
48.         {
49.             var done = await FireAssistant.DeleteRequest(App._variable, ClientName.Text);
50.             await PopupNavigation.Instance.PopAsync();
51.             App.Current.MainPage = new NavigationPage(new PlanRequests());
52.         }
53.
54.         private async void Button_Clicked_1(object sender, EventArgs e)
55.         {
56.             await PopupNavigation.Instance.PopAsync();
57.             await PopupNavigation.Instance.PushAsync(new PopUpTrainerAddPlan());
58.         }
59.     }
60. }
61.
```

## App.xaml

```
1.  <?xml version="1.0" encoding="utf-8" ?>
```

```
<Application xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="GymAndPAapp.App">
    <!--
        Define global resources and styles here, that apply to all pages in your app.
    -->
    <Application.Resources>
        <ResourceDictionary>
            <Color x:Key="Primary">white</Color>
            <Style TargetType="Button">
                <Setter Property="TextColor" Value="Black"></Setter>
                <Setter Property="VisualStateManager.VisualStateGroups">
                    <VisualStateGroupList>
                        <VisualStateGroup x:Name="CommonStates">
                            <VisualState x:Name="Normal">
                                <VisualState.Setters>
                                    <Setter Property="BackgroundColor" Value="{StaticResource Primary}" />
                                </VisualState.Setters>
                            </VisualState>
                            <VisualState x:Name="Disabled">
                                <VisualState.Setters>
                                    <Setter Property="BackgroundColor" Value="#999999" />
                                </VisualState.Setters>
                            </VisualState>
                        </VisualStateGroup>
                    </VisualStateGroupList>
                </Setter>
            </Style>
            <Style TargetType="Label">
                <Setter Property="TextColor" Value="Black" />
            </Style>
            <Style TargetType="Entry">
                <Setter Property="TextColor" Value="Black" />
            </Style>
        </ResourceDictionary>
    </Application.Resources>
</Application>

```

## App.xaml.cs

```
using GymAndPAapp.ViewModels;
using GymAndPAapp.Views;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace GymAndPAapp
{
    public partial class App : Application
    {
        IAuthenticationService auth;
        public static bool _goalsVar { get; set; }
        public static ImageSource profile { get; set; } // profile picture
        public static string _variable { get; set; }    //USERNAME
        public static string _variable2 { get; set; }   //PASSWORD
        public static string _variable4 { get; set; }   //Title
        public static string _variable5 { get; set; }   //page
        public static double _variable6 { get; set; }  //excercise done
        public static double _variable7 { get; set; }  //excercise list size
        public static string _variable8 { get; set; }  //Previous Day
        public static int _variable9 { get; set; }  // changing days plans pages
        public static DateTime DayOfWeek { get; set; } //Current Day
        public static List<string> _VarArray = new List<string>() ;

        public App()
        {
```

```
            SetValue(NavigationPage.HasNavigationBarProperty, false);
            InitializeComponent();
            auth = DependencyService.Get<IAuthenticationService>();
              if (auth.IsSignIn())
              {
                 auth.SignOut();
                 MainPage = new NavigationPage(new LoginPage());

              }
              else
              {
                 MainPage = new NavigationPage(new LoginPage());
              }

          //   MainPage = new NavigationPage(new addexe());
        }

        protected override void OnStart()
        {
            SetValue(NavigationPage.HasNavigationBarProperty, false);
        }

        protected override void OnSleep()
        {
        }

        protected override void OnResume()
        {
        }

    }
```

## Android

### FirebaseAuthentication.cs

```
using Firebase.Auth;
using GymAndPAapp.Droid;
using GymAndPAapp.ViewModels;
using System;
using System.Threading.Tasks;
using Xamarin.Forms;

[assembly: Dependency(typeof(FirebaseAuthentication))]
namespace GymAndPAapp.Droid
{
    public class FirebaseAuthentication : IAuthenticationService
    {
      /* public async Task<string> AdminCloseAccount(string token)
       {
          UserRecord userRecord = await FirebaseAdmin.Auth.FirebaseAuth.DefaultInstance.GetUserAsync(token);
          var user = FirebaseAdmin.Auth.FirebaseAuth.DefaultInstance.DeleteUserAsync(token);
          return token;
       } */

       // currently unsupported with current version

       public async Task<string> AdminRecoveryAsync(string email)
       {
          var user = Firebase.Auth.FirebaseAuth.Instance.SendPasswordResetEmail(email);
          return email;
       }

       public bool CloseAccount()
       {
          var currentUser = Firebase.Auth.FirebaseAuth.Instance.CurrentUser;
          currentUser.DeleteAsync();
          return true;
        }
```

```
34.
35.         public string GetEmail()
36.         {
37.             var currentUser = Firebase.Auth.FirebaseAuth.Instance.CurrentUser;
38.             return currentUser.Email;
39.         }
40.
41.         public bool IsSignIn()
42.         {
43.             var currentUser = Firebase.Auth.FirebaseAuth.Instance.CurrentUser;
44.             return currentUser != null;
45.         }
46.
47.         public async Task<string> LoginWithEmail(string email, string password)
48.         {
49.             try
50.             {
51.                 var newUser = await Firebase.Auth.FirebaseAuth.Instance.SignInWithEmailAndPasswordAsync(email,
    password);
52.                 var token = newUser.User.Uid;
53.                 return token;
54.
55.             }
56.             catch (FirebaseAuthInvalidUserException ex)
57.             {
58.                 ex.PrintStackTrace();
59.                 return string.Empty;
60.             }
61.             catch (FirebaseAuthInvalidCredentialsException ex)
62.             {
63.                 ex.PrintStackTrace();
64.                 return string.Empty;
65.             }
66.         }
67.
68.         public bool resetEmail(string email)
69.         {
70.             try
71.             {
72.                 Firebase.Auth.FirebaseAuth.Instance.SendPasswordResetEmail(email);
73.                 return true;
74.             }
75.             catch(Exception ex)
76.             {
77.                 Console.WriteLine(ex.ToString());
78.                 return false;
79.             }
80.
81.         }
82.         public bool SignOut()
83.         {
84.             try
85.             {
86.                 Firebase.Auth.FirebaseAuth.Instance.SignOut();
87.                 return true;
88.             }
89.             catch(Exception ex)
90.             {
91.                 return false;
92.             }
93.         }
94.         public async Task<string> SignUpWithEmail(string email, string password)
95.         {
96.             try
97.             {
98.                 var newUser = await Firebase.Auth.FirebaseAuth.Instance.CreateUserWithEmailAndPasswordAsync(email,
    password);
99.                 var token = newUser.User.Uid;
100.                return token;
101.
102.            }
```

```
103.            catch (FirebaseAuthInvalidUserException ex)
104.            {
105.                ex.PrintStackTrace();
106.                return string.Empty;
107.            }
108.            catch (FirebaseAuthInvalidCredentialsException ex)
109.            {
110.                ex.PrintStackTrace();
111.                return string.Empty;
112.            }
113.        }
114.        public bool updatePassword(string password)
115.        {
116.            try
117.            {
118.                var currentUser = Firebase.Auth.FirebaseAuth.Instance.CurrentUser;
119.                currentUser.UpdatePassword(password);
120.                return true;
121.            }
122.            catch (Exception ex)
123.            {
124.                Console.WriteLine(ex.ToString());
125.                return false;
126.            }
127.        }
128.        string IAuthenticationService.GetUser()
129.        {
130.            var currentUser = Firebase.Auth.FirebaseAuth.Instance.CurrentUser;
131.            var token = currentUser.Uid;
132.            return token;
133.        }
134.        bool IAuthenticationService.UpdateEmail(string email)
135.        {
136.            try
137.            {
138.                var currentUser = Firebase.Auth.FirebaseAuth.Instance.CurrentUser;
139.                currentUser.UpdateEmail(email);
140.                return true;
141.            }
142.            catch(Exception ex)
143.            {
144.                Console.WriteLine(ex.ToString());
145.                return false;
146.            }
147.        }
148.    }
149. }
150.
```

## MainActivity.cs

```
1.    using System;
2.
3.    using Android.App;
4.    using Android.Content.PM;
5.    using Android.Runtime;
6.    using Android.Views;
7.    using Android.Widget;
8.    using Firebase;
9.    using Android.OS;
10.   using Rg.Plugins.Popup.Services;
11.   using Android.Content;
12.   using Xamarin.Forms;
13.   using static Android.Renderscripts.Sampler;
14.   using GymAndPAapp.ViewModels;
15.
16.   namespace GymAndPAapp.Droid
17.   {
```

```
18.     [Activity(Label = "GymAndPAapp", Icon = "@mipmap/icon", Theme = "@style/MainTheme", MainLauncher = false,
    ConfigurationChanges = ConfigChanges.ScreenSize | ConfigChanges.Orientation | ConfigChanges.UiMode |
    ConfigChanges.ScreenLayout | ConfigChanges.SmallestScreenSize )]
19.     public class MainActivity : global::Xamarin.Forms.Platform.Android.FormsAppCompatActivity
20.     {
21.         protected override void OnCreate(Bundle bundle)
22.         {
23.             TabLayoutResource = Resource.Layout.Tabbar;
24.             ToolbarResource = Resource.Layout.Toolbar;
25.             base.OnCreate(bundle);
26.
27.             FirebaseApp.InitializeApp(BaseContext);
28.             Rg.Plugins.Popup.Popup.Init(this);
29.             Xamarin.Essentials.Platform.Init(this, bundle);
30.             global::Xamarin.Forms.Forms.Init(this, bundle);
31.             LoadApplication(new App());
32.         }
33.         public override void OnBackPressed()
34.         {
35.             if (App._variable5 == null)
36.             {
37.
38.             }
39.             else if(App._variable5 == "PopUpTrainerAddPlan")
40.             {
41.                 App._variable9 = 0;
42.                 PopupNavigation.Instance.PopAllAsync();
43.             }
44.             else if (App._variable5 == "EditPlan")
45.             {
46.                 App._variable9 = 0;
47.             }
48.             else if (App._variable5 != "PopUpTrainerAddPlan")
49.             {
50.                 if (App._variable5 != "EditPlan")
51.                 {
52.                     PopupNavigation.Instance.PopAllAsync();
53.                 }
54.                 else
55.                 {
56.                     Rg.Plugins.Popup.Popup.SendBackPressed(base.OnBackPressed);
57.                 }
58.             }
59.             else
60.             {
61.                 Rg.Plugins.Popup.Popup.SendBackPressed(base.OnBackPressed);
62.                 App._variable9--;
63.             }
64.         }
65.         public override void OnRequestPermissionsResult(int requestCode, string[] permissions, [GeneratedEnum]
    Android.Content.PM.Permission[] grantResults)
66.         {
67.             Xamarin.Essentials.Platform.OnRequestPermissionsResult(requestCode, permissions, grantResults);
68.
69.             base.OnRequestPermissionsResult(requestCode, permissions, grantResults);
70.         }
71.     }
72. }
73.
```

## SplashActivity.cs

```
1.  using Android.App;
2.  using Android.OS;
3.  using GymAndPAapp.Droid;
4.
5.  namespace MyApp.Droid
6.  {
```

```
7.      [Activity(Label = "GymAndPAapp", Icon = "@mipmap/icon", Theme = "@style/Theme.Splash", MainLauncher =
    true)]
8.    public class SplashActivity : Activity
9.    {
10.       protected override void OnCreate(Bundle bundle)
11.       {
12.          base.OnCreate(bundle);
13.
14.          SetContentView(Resource.Layout.SplashLayout);
15.          System.Threading.ThreadPool.QueueUserWorkItem(o => LoadActivity());
16.       }
17.
18.       private void LoadActivity()
19.       {
20.          System.Threading.Thread.Sleep(4000); // Simulate a long pause
21.          RunOnUiThread(() => StartActivity(typeof(MainActivity)));
22.       }
23.   }
24. }
25.
```

# Declaration

- I declare that all material in this submission, e.g., thesis/essay/project/assignment, is entirely my own work except where duly acknowledged.

- I have cited the sources of all quotations, paraphrases, summaries of information, tables, diagrams, or other material; including software and other electronic media in which intellectual property rights may reside.

- I have provided a complete bibliography of all works and sources used in the preparation of this submission.

- I understand that failure to comply with the Institute's regulations governing plagiarism constitutes a serious offense.

Student Name: Jamie Hawthorne

Student Number: C00226160

Student Signature: 

Date: 30/04/2021