Project Report **30/04/2021**

Application Green Quake

Student Name: Peter Lucan
Student Number: C00228946
Supervisor: Chris Meudec

Github Link:
https://github.com/PeterX12/Application-Green-Quake.git
APK Github Link:
https://github.com/PeterX12/Application-Green-Quake/blob/master/ApplicationGreenQuake.apk
APK Google Drive Link:
https://drive.google.com/file/d/1MlPChFA_Z_XOzcy3yi_Q1Gt6JyNJcC1e/view?usp=sharing

Abstract

This document serves as the concluding report to my final year project. The entire project has been documented in this report. It provides technical and non technical insights into the project. It clearly explains the entire process involved in creating this application and the changes made along the way.

Table of Contents

1 Introduction

This document begins by describing both the non technical and technical aspects of the project followed by the UI-UX designs and the database structure of the project. This clearly explains what the project is about and how it looks and works.

Following on, the document talks about how the agile process was thoroughly followed throughout the project, how feedback was gathered, listened to and implemented into the application and all the testing that was performed on the application throughout its development.

Finally the document ends with a personal reflection on the problems encountered during this project and how they were solved and the lesson that has been derived from said problems. The document ends by reviewing the project and speaking about what was and wasn't achievend and other possibilities.

2 Project Description

This document concluded the development of the Green Quake mobile application. The goal of Green Quake was to make being environmentally friendly feel more easier, fun and rewarding to do by gamifying doing environmentally friendly activities and by providing the right tools and information relating to the environment to the user in an attractive manner. The main goal was to help the environment and in turn aid Europe in achieving the aims of the Green Deal by 2050.

The delivered application kept its promise and was able to do so although some modification of the original plan and design had to be made due to time and financial constraints. The original plan was to develop two versions of the application. One for the general public and one for companies to monitor their employees environmental activities and award them for them but only the first version of the application was developed as two versions was simply too much for a single developer.

The application allows users to log in and log eco friendly activities that they perform and earn points for them. In addition the users can use the Water Refill Station Tool to find the nearest places where they can refill a water bottle encouraging them to use reusable water bottles instead of plastic ones as plastic waste is a huge problem in 2021. A user can lvl up and unlock pieces of mosaics, place on a global or national leaderboard and also earn live trophies, badges and achievements as well as edit their profile.

3 Technical Project Description

The Green Quake Mobile application was developed using the agile process with 2 week to 4 week long iterations and a deliverable at the end of each iteration. It also closely followed the specifications laid out in the earlier stages of the project and changed with the changes in the specification. Regular commits and pushes were made to github from my local machine whenever changes were applied to the code.

This application was developed using Xamarin Forms for the front end and C# for the back end. This allowed for cross platform mobile app development with one solution for both Android and iOS. These languages were chosen as they are maintained and well documented by Microsoft and are very powerful languages for cross mobile app development.

The database used for this application was Firebase which was a great decision as it is a real time database. It is a cloud-hosted NoSQL database that lets you store and sync data between your users in real time. This means that it allows fluid synchronization across all mobile devices that have the application and that if your devices go offline they can get all the data they need once more by reconnecting to the database.

Firebase Authentication was used for managing the credentials of the users. A strong password validation system was implemented when creating an account that required a password which had at least 8 characters, at least one lower case and upper case character, at least one number and finally at least one symbol. The password is then sent to firebase where it gets a salt prepended to it and it gets hashed.

Firebase Storage was used to store the users images. This allowed the application real time access to the images of the users.

In order to implement the water refill station locator tool Google Api had to be used. This tool loads a map focused on the users current location and displays pins of places where they can refill their water bottle and information about those places.

In order to monitor and provide reports on app crashes and other bugs.The Microsoft App Center API had to be used and implemented by downloading the relevant SDK and making some code changes.

A number of Nuget Packages had to be downloaded and installed to make the app function to the desired effect and these packages and details about them are listed below.

- **Acr.UserDialogs (Entire Project):** A cross platform library that allows you to call for standard user dialogs from a shared/portable library.
- **FirebaseDatabase.net (Entire Project):** A complex C# library for Firebase Real Time Database and it is built on top of the REST API.
- **FirebaseStorage.net (Entire Project):** Allows the uploading of files to Firebase.
- **Microsoft.AppCenter.Analytics (Entire Project):** Microsoft App Center package that provides analytics capabilities for the Green Quake mobile application.
- **Microsoft.App.Center.Analytics (Entire Project):** Provides Crash reporting capabilities for the application.
- **NETStandard.Library (Entire Project):** Just all the.NET APIs that work together.
- **Plugin.CurrentActivity (Entire Project):** Provides a simple solution for getting access to the current Activity of the application when developing a Plugin for Xamarin.

- **Rg.Plugins.Popup (Entire Project):** Plugin for Xamarin forms. Allows you to open any page as a popup.
- **Syncfusion.Xamarin.SfPicker (Entire Project):** The Syncfusion Picker for Xamarin.Forms allows users to pick an item from a list of items that can be customized using a template or custom view.
- **Xam.Plugin.Media (Entire Solution):** Allows the app to take and store images as well as pick them from the gallery.
- **Xamarin.Android.Support.Annotations (Android):** Xamarin.Android bindings for Android Support Library.
- **Xamarin.Android.Support.Compat (Android):** Xamarin.Android bindings for Android Support Library.
- **Xamarin.Android.Support.Core.UI (Android):** Xamarin.Android bindings for Android Support Library.
- **Xamarin.Android.Support.Core.Utils (Android):** Xamarin.Android bindings for Android Support Library.
- **Xamarin.Android.Support.Design (Android):** Xamarin.Android bindings for Android Support Library.
- **Xamarin.Android.Support.Fragment (Android):** Xamarin.Android bindings for Android Support Library.
- **Xamarin.Android.Support.VersionedParcelable (Android):** Xamarin.Android bindings for Android Support Library.
- **Xamarin.Android.Volley (Android):**
- **Xamarin.AndroidX.Browser (Android):** Xamarin.Android bindings for AndroidX.
- **Xamarin.AndroidX.Legacy.Support.V4 (Android):** Xamarin.Android bindings for AndroidX.
- **Xamarin.AndroidX.Lifecycle.LiveData (Android):** Xamarin.Android bindings for AndroidX.
- **Xamarin.Essentials (Entire Solution):** Xamarin.Essentials: a kit of essential API's for the app.
- **Xamarin.Firebase.Auth (Entire Solution):** Xamarin.Android Bindings for Google Play Services.
- **Xamarin.Firebase.Common (Entire Solution):** Xamarin.Android Bindings for Google Play Services.

- **Xamarin.Firebase.Core (Entire Solution):** Xamarin.Android Bindings for Google Play Services.
- **Xamarin.Firebase.Database (Entire Solution):** Xamarin.Android Bindings for Google Play Services.
- **Xamarin.Firebase.iOS.Auth (iOS):** C# bindings for Firebase APIs Auth iOS Library.
- **Xamarin.Firebase.iOS.Database (iOS):** C# bindings for Firebase APIs Database iOS Library.
- **Xamarin.Forms (Entire Solution):** Allows the building of native UIs for iOS, Android, UWP, macOS, Tizen and many more from a single, shared C# codebase.
- **Xamarin.Forms.GoogleMaps (Entire Solution):** A Maps library for Xamarin.Forms that is optimized for Google maps.
- **Xamarin.Google.Android.Material (Android):** Xamarin.Android bindings for AndroidX.
- **Xamarin.Google.iOS.Places (Entire Solution):** C# bindings for Google APIs Places iOS Library.
- **Xamarin.GooglePlayServices.Maps (Entire Solution):** Xamarin.Android Bindings for Google Play Services.

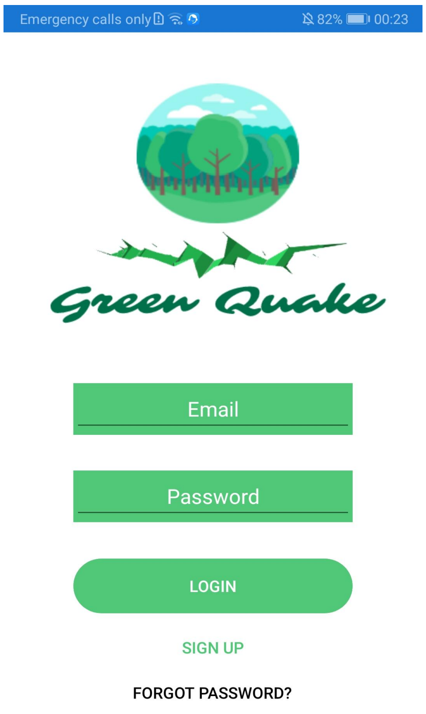
The application was developed using Visual Studio 2019 for software development and tools like Befunky [1], Gravitat [2] and BadgeBuilder [3] were used for graphics design and Miro [4] and Jira [5] were used for planning, designing and brainstorming as well as aiding in the agile process by storing all the use cases and user stories and acting as the backlog. All the images in the application were images that are free for commercial use from Pixabay [6], Pexels [7] and Unsplash [8]. Original Icons Templates were taken from Flaticon [9], Iconfinder [10] and Icon8 [11] and are also free for commercial use.

Git and Github were used via the terminal to backup the code for this application and push it to a remote repository to keep it safe if something was to occur to the local machine.

The application also uses 3 APIs. The Firebase API for the database, file storage and authentication functionality, the Google Maps API for the maps functionality and App Centers API to monitor and provide analysis on app errors and crashes to the developer so it can be updated regularly and fixed after it is released.

4 Final Application Screens

4.1 Login Screen

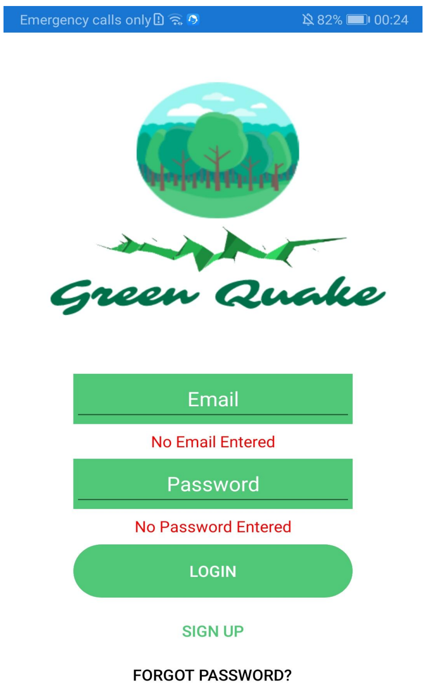


As you can see this is the **login** screen**.** This will be the opening screen for the application when the user is not signed in. A user can enter their email and password and click the login button to attempt to sign in or they can tap the Sign up option to navigate to the sign up page or they can tap on the Forgot Password? Option to navigate to the Forgot Password screen.

Upon successful login a nice splash screen appears which can not be documented as it is an animated event. After this splash screen the user is brought to the main menu.

Figure 1: Login Screen.

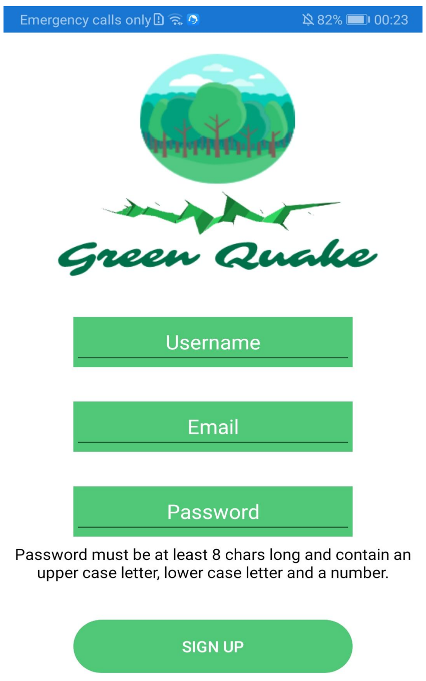
4.2 Login Screen With Error Messages


Emergency calls only 82% 00:24

As you can see this is the **login** screen once more but now displays error messages for the invalid input fields. These error messages vary depending on the situation. If the user is not connected to the internet an alert pops up asking them to connect to the internet.

Figure 2: Login Screen With Error Messages.

4.3 Sign Up Screen

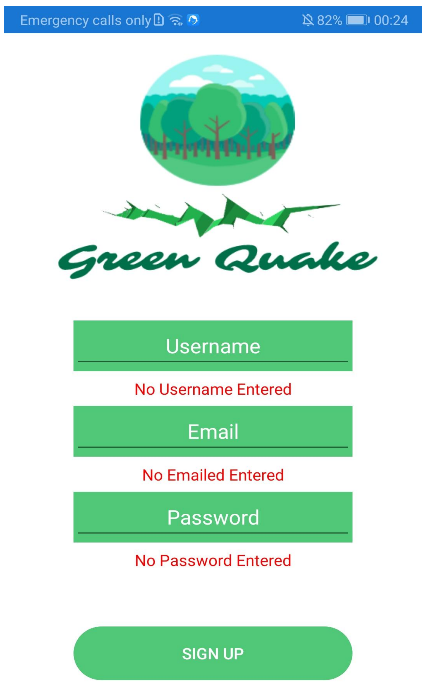

Emergency calls only 82% 00:23

Password must be at least 8 chars long and contain an upper case letter, lower case letter and a number.

As you can see this is the **Sign Up** screen. A user can enter any username they wish and only a valid non duplicate email and a password which complies with the rules listed under the password fields. Upon successful signup an alert pops us saying your account has been created and redirects the user to the login page.

Figure 3: Sign Up Screen.

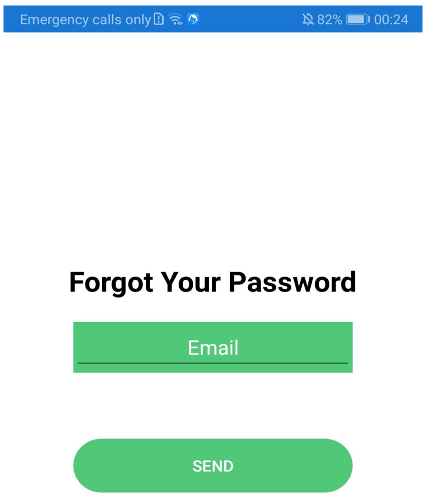
4.4 Sign Up Screen With Error Messages



As you can see this is the **Sign Up** screen once more but now displays error messages for the invalid input fields. These error messages vary depending on the situation. If the user is not connected to the internet an alert pops up asking them to connect to the internet. If the email already exists an alert pops up saying the email already exists and if there is an error a generic error alert pops up.

Figure 4: Sign Up Screen With Error Messages.

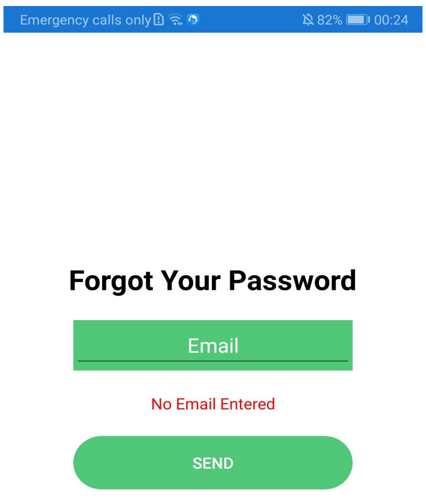
4.5 Forgot Password Screen



As you can see this is the **Forgot Password** screen. A user can enter their email and if the email exists a password reset email will be sent to that email address. An alert saying "If the email exists a password reset email has been sent to your email address". Then the user is redirected to the login page.

Figure 5: Forgot Password Screen.

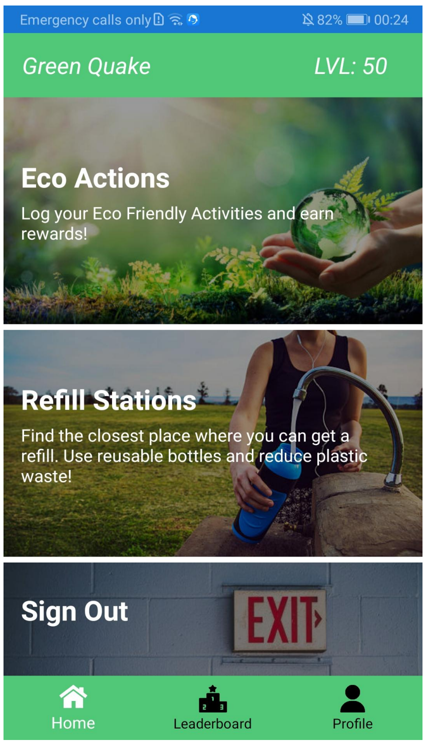
4.6 Forgot Password Screen With Error Messages



As you can see this is the **Forgot Password** screen once more but now displays an error message for the invalid input field. If the user is not connected to the internet or the email is invalid the corresponding alert box pops up.

Figure 6: Forgot Password Screen With Error Messages.

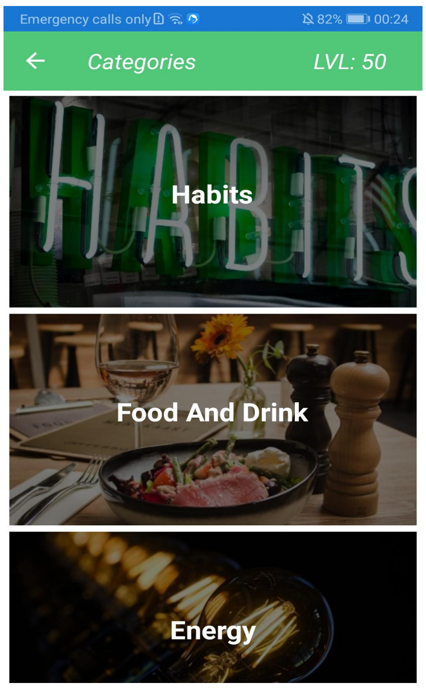
4.7 Main Menu Screen


Emergency calls only
82% 00:24
Green Quake
LVL: 50
Eco Actions
Log your Eco Friendly Activities and earn rewards!
Refill Stations
Find the closest place where you can get a refill. Use reusable bottles and reduce plastic waste!
Sign Out
EXIT
Home
Leaderboard
Profile

This is the **main menu** screen that appears after successful login. The user can navigate to the Eco Actions section by tapping on it and slo to the Refill Station section by tapping on it. In addition the user can navigate to the Leaderboard or Profile section using the navigation bar at the bottom of the page. The level of the user is displayed in the top right of the screen.

Figure 7: Main Menu Screen.

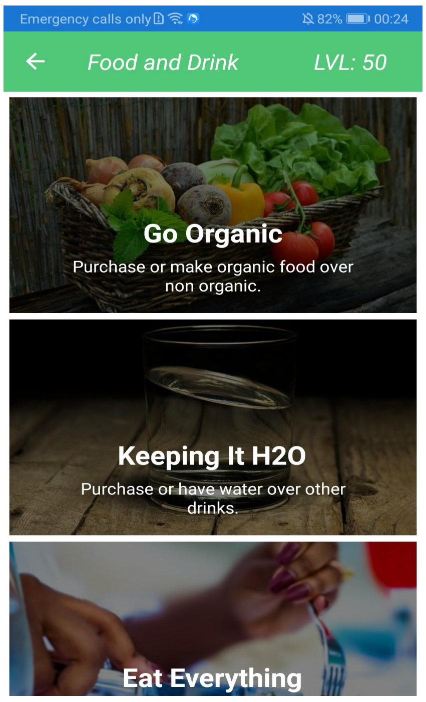
4.8 Categories Screen



This is the **Categories** screen that appears after a user selects Eco Actions from the previously mentioned main menu. From here the user can select one of 12 categories to log and eco action for. These categories are: Habits, Food And Drink, Energy, Travel, Shopping, Water, Home, Outdoors, Community, Waste, Work and Advanced categories.

Figure 8: Categories Screen.

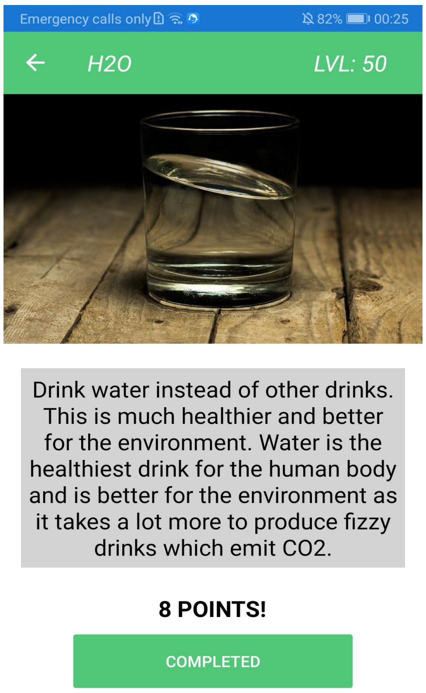
4.9 Food and Drink Subcategories Screen


Emergency calls only
82% 00:24
Food and Drink
LVL: 50
Go Organic
Purchase or make organic food over non organic.
Keeping It H2O
Purchase or have water over other drinks.
Eat Everything

This is the **Food and Drink** subcategories screen that appears after a user selects Food and Drink from the previously mentioned categories screen.

Figure 9: Food and Drink Subcategories Screen.

4.10 H20 Action Screen



Drink water instead of other drinks. This is much healthier and better for the environment. Water is the healthiest drink for the human body and is better for the environment as it takes a lot more to produce fizzy drinks which emit CO2.

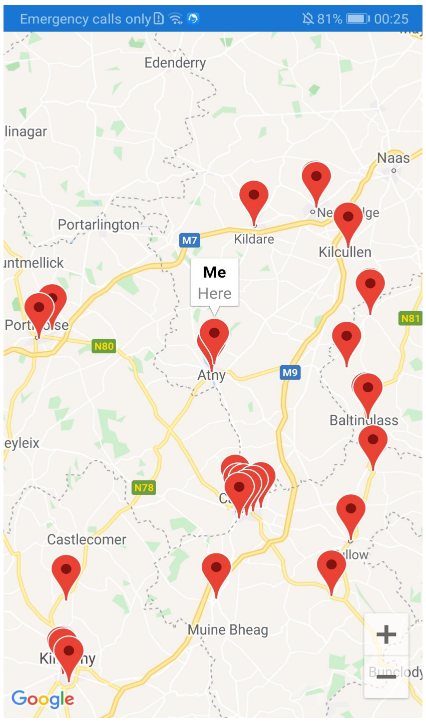
8 POINTS!

This is the **H20** action screen. This screen provides some details and interesting information about this particular action. The amount of points that it is worth is displayed below the text. The user can press the Completed button to log the action. Once this is pressed either the points get posted into the database and an alert box with the successful message is displayed to the user before redirecting them to the main menu or one of two error message alert boxes pop up.

The application does not allow more than 15 posts per day and an alert box saying this will be shown to the user and the post will not be submitted if the user tries to submit more than 15 actions per day. The app also only allows one submission per minute to prevent spamming and an alert box saying this also pops up for the user and does not submit the post.

Figure 10: H20 Action Screen.

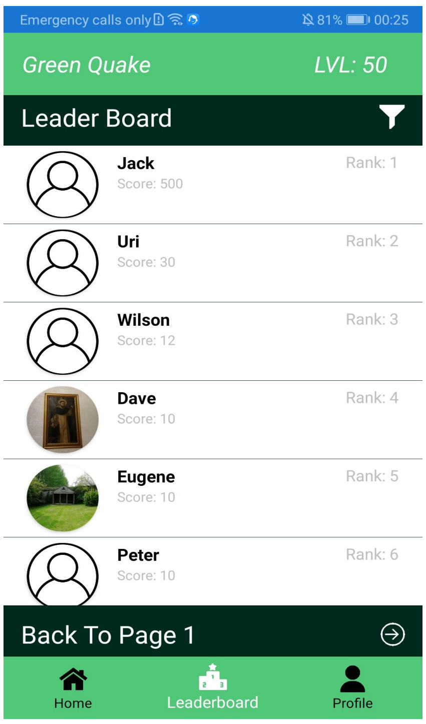
4.11 Refill Stations Screen



This is the **Refill Stations** screen that appears after a user selects Refill Station from the previously mentioned main menu. This screen loads a map at your current location and displays all the Refill Station on the map. The user can clearly see where the refill stations are located in retrospect to their location and can tap on each pin to view more details about it.

Figure 11: Refill Stations Screen.

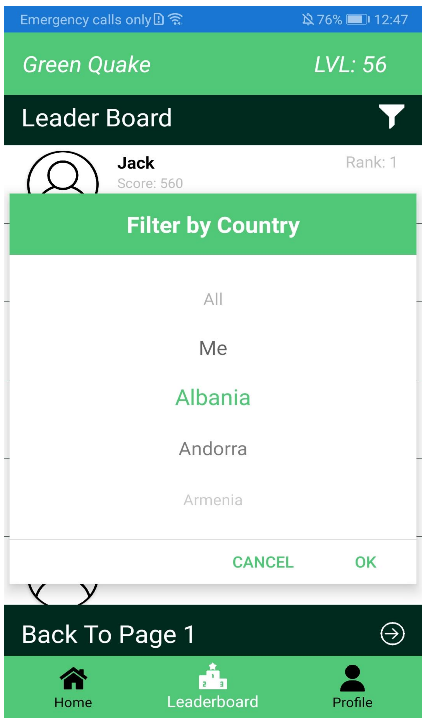
4.12 Leaderboard Screen



This is the **Leaderboard** screen that appears after a user selects the Leaderbaord from the navigation menu at the bottom of the screen on the previously mentioned main menu. On this screen the leaderboard displays 10 rows per page. The user can tap on a profile to view more information about the user. The user can also tap the icon in the top right to filter the leaderboard. The user can also tap the right arrow at the bottom right of the page to see the next 10 entries and tap the Back to Page on text to return to the first page. If there are no more pages to loads then an alert box with the appropriate message gets displayed.

Figure 12: Leaderboard Screen.

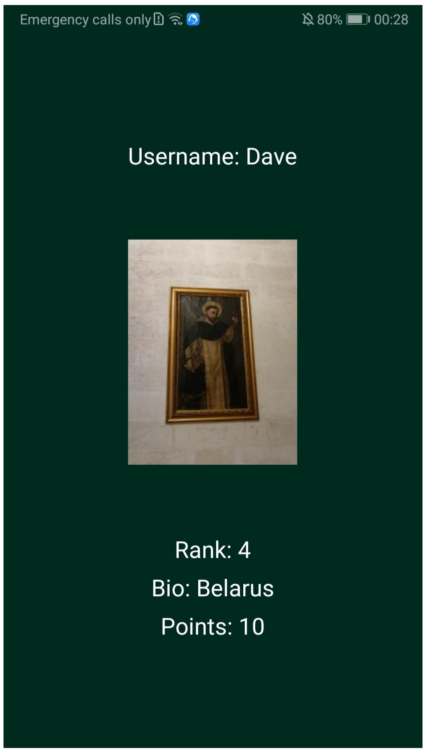
4.13 Leaderboard Screen When The Filter Button Is Tapped



This is the **Leaderbaord** screen when the **filter icon** is tapped. A picker with the options All, Me and the list of nations appears. The user is allowed to filter to display all the users on the leaderboard, filter by nation or view their own position on the leaderboard global.

Figure 13: Leaderboard Screen When The Filter Button Is Tapped.

4.14 Leaderboard Pop Up Screen When A Profile Is Tapped On The Leaderboard


Emergency calls only 80% 00:28

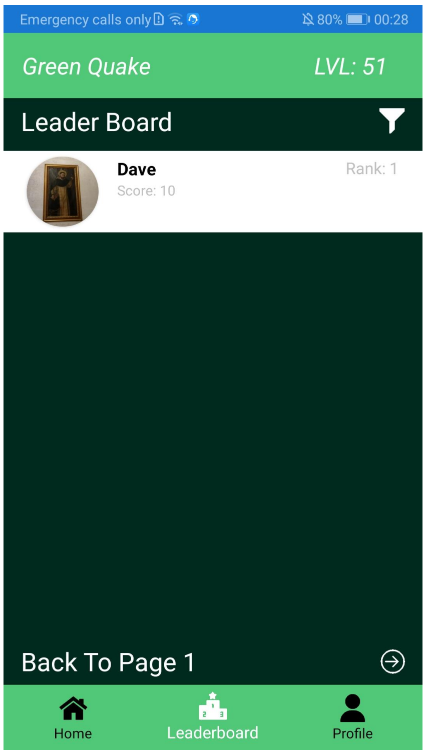
Username: Dave

Rank: 4
Bio: Belarus
Points: 10

This **popup** screen appears when a profile is tapped on from the leaderboard. It displays more information about the tapped on user. The information displayed is the username, the profile picture, the rank, the bio and the points.

Figure 14: Leaderboard Pop Up Screen When A Profile Is Tapped On The Leaderboard.

4.15 Filtered Leaderboard Screen


Emergency calls only 80% 00:28

Green Quake LVL: 51

Leader Board

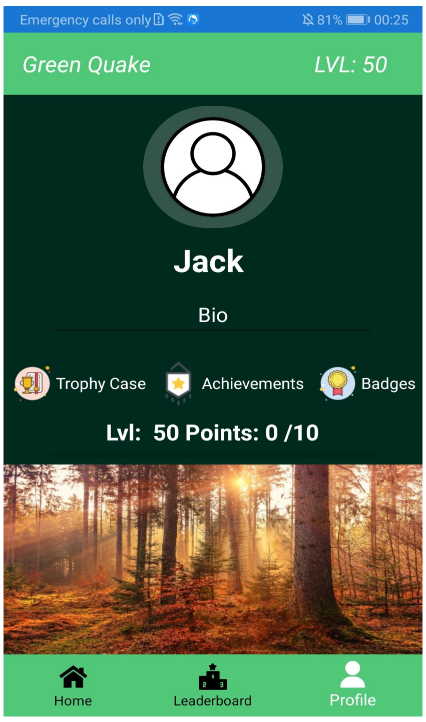
Dave Rank: 1
Score: 10

Back To Page 1

This is the **Leaderboard** after it has been **Filtered by Nation**. Only the profiles with the correct nation are displayed and they are ranked against other accounts of the same nation.

Figure 22: Filtered Leaderboard Screen.

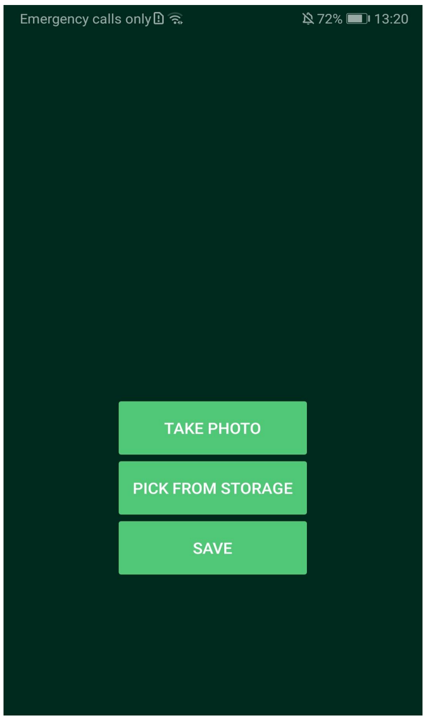
4.16 Profile Screen



This is the **Profile** screen that appears after a user selects the Profile from the navigation menu at the bottom of the screen on the previously mentioned main menu. The Profile screen has the profile picture and bio that the user can tap to change and their username. Below this are the Trophies, Achievements and Badges. By tapping on each of these the corresponding screen is displayed showing the Trophies, Achievements and Badges. Below this, the level of the user is displayed with a progress bar to the next level indicating how many points out of 10 the user needs to level up. Finally below this is the live avatar that is a mosaic that gets unlocked piece by piece on each level up. The mosaic gets unlocked after each level so each mosaic has 5 stages and there are 20 mosaics to unlock which makes them end at lvl 100.

Figure 16: Profile Screen.

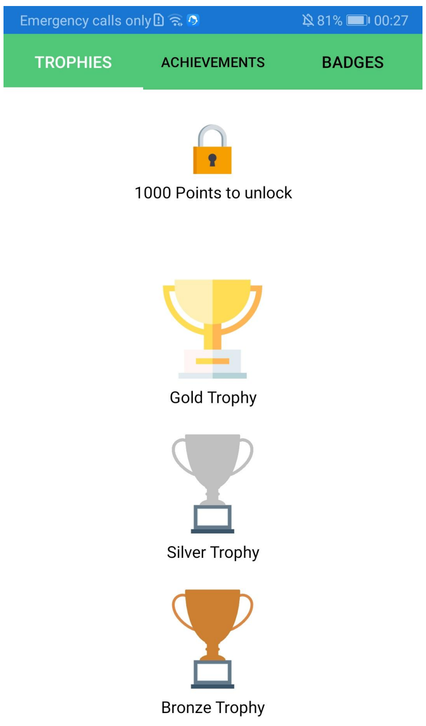
4.17 Popup Screen After The Profile Image Is Tapped



This **popup** screen appears when the user taps on the profile image on the profile screen. This popup allows the user to take a photo and save it as their profile picture or pick an image from the phone's storage and save it as their profile picture. Then the user is redirected to the profile screen.

Figure 17: Popup Screen After The Profile Image Is Tapped.

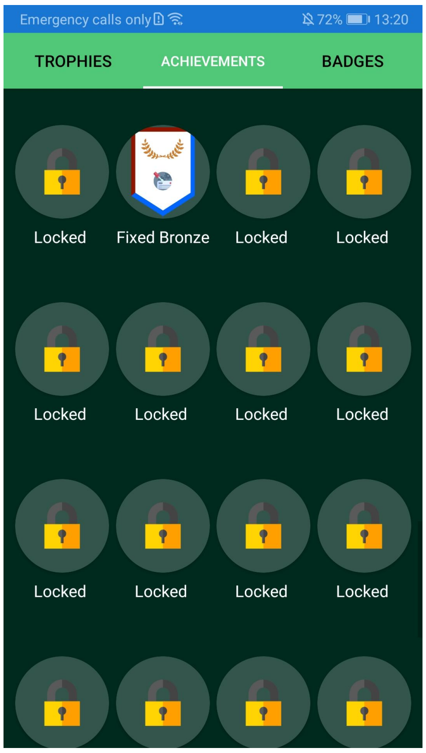
4.18 The Trophies Screen



This is the **Trophies** Screen that can be navigated to from the profile screen by tapping on the trophy case icon and text. The trophies are initially locked and get replaced with trophies as the requirements get met. Bronze trophy for 100 points, Silver for 250, Gold for 500 and Diamond for 1000.

Figure 18: The Trophies Screen.

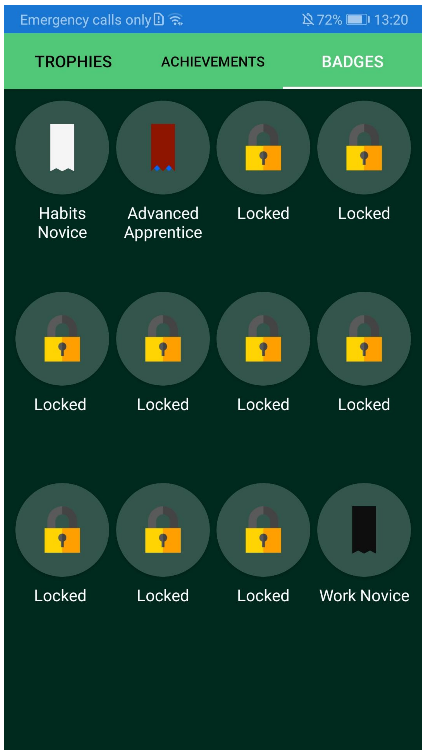
4.19 The Achievements Screen



This is the **Achievements** Screen that can be navigated to from the profile screen by tapping on the achievements icon and text. There are achievements for every action that can be logged which makes it over 80 achievements that the user can earn. In addition to this these are live achievements which can be bronze, silver and gold so there are over 260 achievements to earn in total. The achievements are initially locked and get unlocked by meeting the qualifying requirements. A bronze achievement is unlocked when a certain action gets logged 5 or more times, a silver achievement is unlocked when a certain action gets logged 15 or more times and finally a gold achievement is unlocked when a certain action gets logged 25 or more times.

Figure 19: The Achievements Screen.

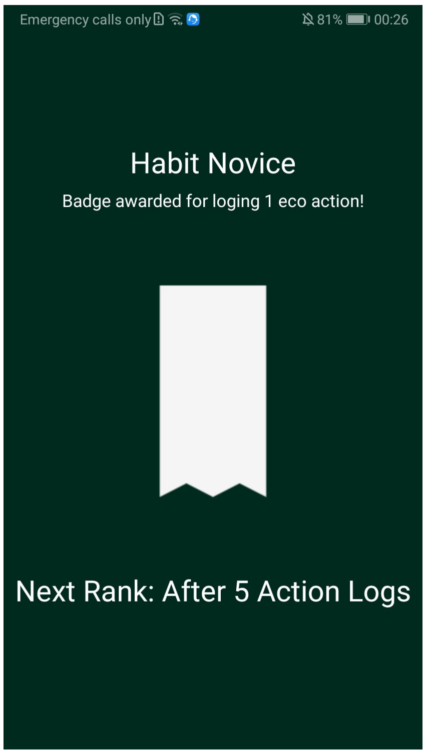
4.20 The BadgesScreen



This is the **Badges** Screen that can be navigated to from the profile screen by tapping on the badges icon and text. There are badges for every category that a user makes a login. There are 12 categories and there are 12 badges that can be unlocked and displayed. These are live badges and there are 6 levels in each category making it 72 badges that the user can earn. Each badge has a different design. The badges are initially locked and get unlocked when certain conditions are met. The Novice badge for a certain category gets unlocked when the user makes a single log or more in that category. Apprentice badge for 5 or more, Adept badge for 10 or more, Expert badge for 25 or more, Master badge for 50 or more and Legend Badge for 100 or more. All these badges can also be tapped on to get a close up and view more information.

Figure 20: The Badges Screen.

4.21 The Badge Popup Screen



Habit Novice

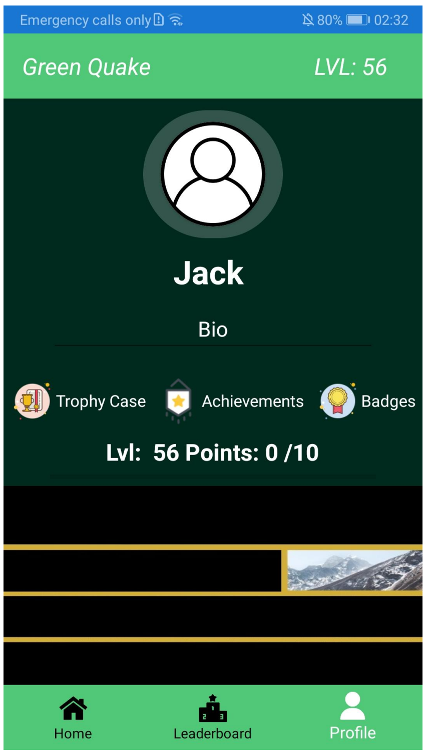
Badge awarded for loging 1 eco action!

Next Rank: After 5 Action Logs

This **Popup** screen appears when a user taps on the badges on the badges page. This image displays a close up of the badge and a description for the badge and what it was awarded for. It also tells the user the requirements to get the next badge.

Figure 21: The Badges Screen Popup When A Badge Is Tapped.

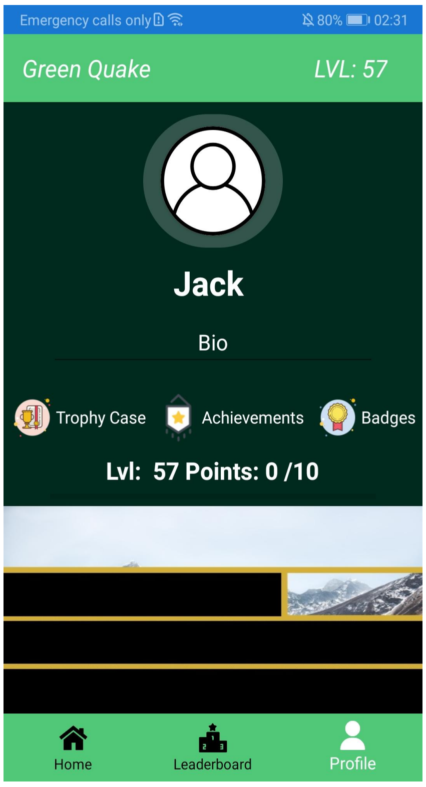
4.22 The Live Avatar Mosaics on Lvl 56



As you can see the mosaic only has one piece unlocked on level 56.

Figure 22: The Live Avatar Mosaics on Lvl 56.

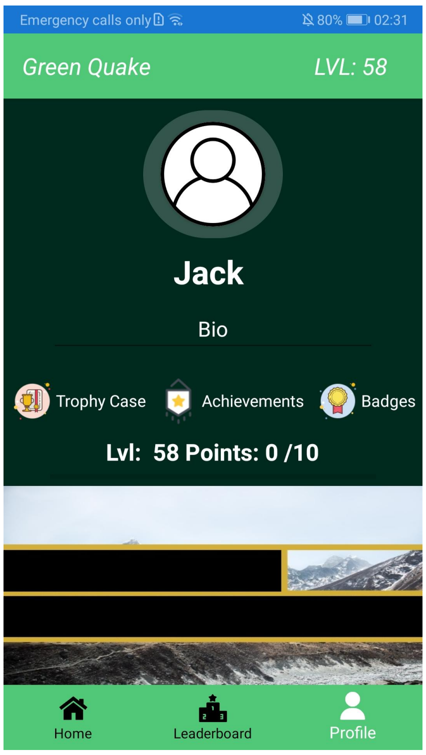
4.23 The Live Avatar Mosaics on Lvl 57



As you can see the mosaic only has two pieces unlocked on level 57.

Figure 23: The Live Avatar Mosaics on Lvl 57.

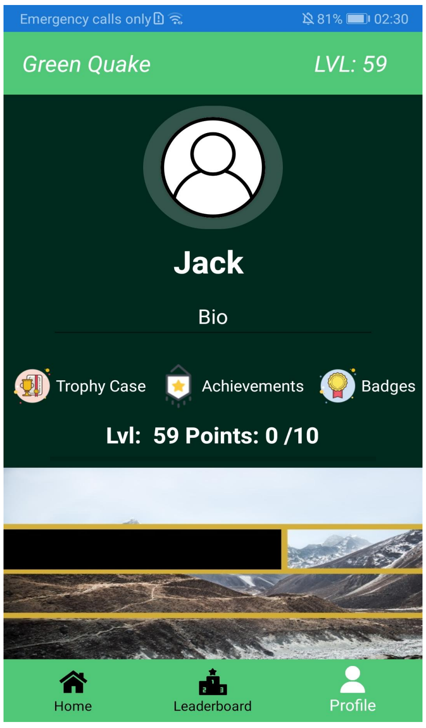
4.24 The Live Avatar Mosaics on Lvl 58



As you can see the mosaic has three pieces unlocked on level 58.

Figure 24: The Live Avatar Mosaics on Lvl 58.

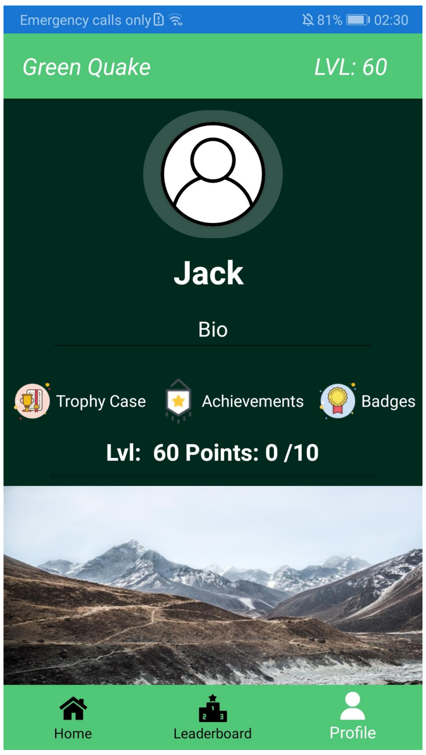
4.25 The Live Avatar Mosaics on Lvl 59



As you can see the mosaic has four pieces unlocked on level 59.

Figure 25: The Live Avatar Mosaics on Lvl 59.

4.26 The Live Avatar Mosaics on Lvl 60



As you can see the mosaic has all pieces unlocked on level 60. The process repeats itself every 5 levels with a new image each time all the way to level 100.

Figure 26: The Live Avatar Mosaics on Lvl 60.

5 Final Database Structure

The Firebase Database is of a JSON format. The structure that was created below is the best way to represent the database schema of the project.

```
application-green-quake-default-rtdb
-|AdvancedPoints
        -|UID
                -|fixCount
                -|numberOfLogs
                -|points
                -|username
-|EnergyPoints
        -|UID
                -|draftSealCount
                -|ductSealCount
                -|efficientThermostatCount
                -|fridgeCount
                -|fullDryerCount
                -|fullMachineCount
                -|hangDryCount
                -|insulateWaterCount
                -|isolateHomeCount
                -|ledLightBulbCount
                -|microwaveCount
                -|numberOfLogs
                -|offSocketCount
                -|points
                -|reBatteriesCount
                -|solarPanelCount
                -|username
```

```
-|FoodAndDrinkPoints
        -|UID
                -|eatAllCount
                -|foodDeliverCount
                -|noMeatCount
                -|numberOfLogs
                -|organicCount
                -|ownCoffeeCount
                -|points
                -|reCoffeeMugCount
                -|saveLeftOversCount
                -|steelStrawCount
                -|username
                -|waterOverFizzyCount
-|HabitsPoints
        -|UID
                -|brushingCount
                -|fullWasherCount
                -|matchesCount
                -|numberOfLogs
                -|offLigtsCount
                -|points
                -|showerCount
                -|timedShowerCount
                -|username
```

```
-|Points
        -|UID
                -|points
                -|username
-|SecuritySchecks
        -|UID
                -|counter
                -|date
                -|time


-|Station
        -|ID
                -|description
                -|label
                -|latitude
                -|longitude
-|Travel
        -|UID
                -|carpoolCount
                -|cycleCount
                -|ecoCarCount
                -|numberOfLogs
                -|points
                -|transportCount
                -|username
                -|walkCount
```

```
-|WastePoints
        -|UID
                -|billsCount
                -|bioBinBagsCount
                -|compostCount
                -|numberOfLogs
                -|points
                -|recyclingBinCount
                -|setUpRecyclingBinCount
                -|username
-|WorkPoints
        -|UID
                -|numberOfLogs
                -|offElectronicsCount
                -|paperCount
                -|points
                -|remoteWorkCount
                -|username
-|usernames
        -|username
                -|Uid
-|Users
        -|UID
                -|bio
                -|nation
                -|username
```

Firebase Storage is used to store images for users and it's structure can be seen below:

-|gs://application-green-quake.appspot.com

-|UID

-|filename

6 Following The Agile Process

From day one this project was developed following the agile process. Work was planned and portioned out into use cases and user stories which were then placed in a backlog. These were then ranked from the Highest to the lowest priority and then the highest priority tasks were taken into the sprints which lasted from 2 to 4 weeks at most and produced a working deliverable at the end of each iteration. A software tool called Jira was used to manage the iterations, Use cases and User stories and a software tool called Miro was also used. Jira is a tool that allows the management of Agile and Miro is an online infinite virtual whiteboard that was used for planning and brainstorming. Whenever a change was made to the code the code was committed using git and pushed to Github using git. My Miro workspace is shown below and keep in mind that this workspace was ever changing and a ton of work was done on it and Jira.

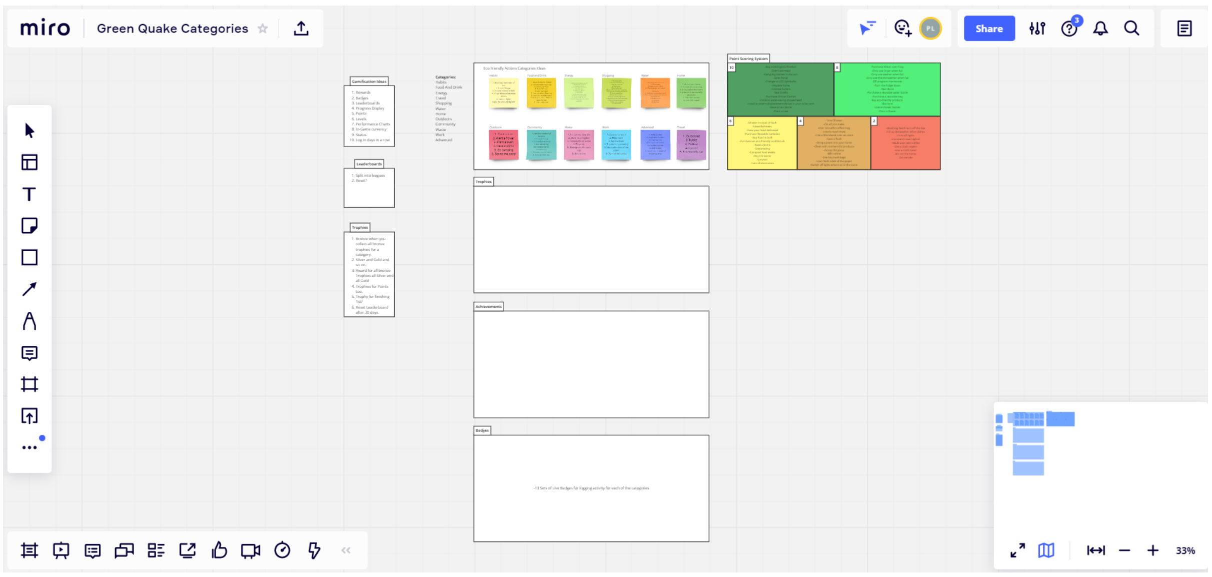
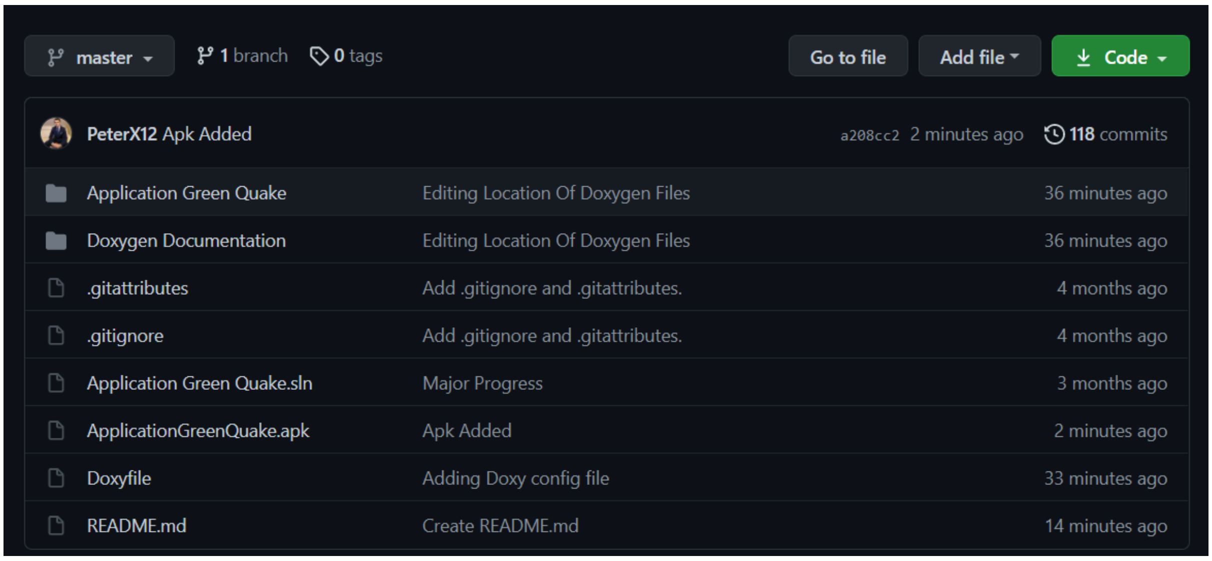


Figure 27: My Miro Workspace.

I have used Github throughout and made over 110 commits as can be seen in the screenshot below. In addition to this the entire solution is documented by Doxygen.


master 1 branch 0 tags Go to file Add file Code

PeterX12 Apk Added a208cc2 2 minutes ago 118 commits

Application Green Quake	Editing Location Of Doxygen Files	36 minutes ago
Doxygen Documentation	Editing Location Of Doxygen Files	36 minutes ago
.gitattributes	Add .gitignore and .gitattributes.	4 months ago
.gitignore	Add .gitignore and .gitattributes.	4 months ago
Application Green Quake.sln	Major Progress	3 months ago
ApplicationGreenQuake.apk	Apk Added	2 minutes ago
Doxyfile	Adding Doxy config file	33 minutes ago
README.md	Create README.md	14 minutes ago

Figure 28: My Github Repository.

The entire solution has been documented in detail using doxygen and it's comments. Documentation is provided in all the available formats. The entire solution is documented and there are quite a lot of files so just some searching is required to get through all these files. As can be seen above this documentation is in a folder called "Doxygen Documentation" and can be found on my github in the root of my project.

7 Feedback & Feedback Implementation

After the final presentation was completed feedback was received from Paul Barry to implement mitigation against malicious users. Checks and limitations based on time were advised. I listened to this advice and implemented checks and limitations on the logging of the eco actions. A user can only log a maximum of 15 logs per day and a maximum of 1 log per 60 seconds.

I have also received feedback from me whenever I give him a working deliverable to review. The feedback that I received was that I should implement a filtering feature in the leaderboard and improve the loading times of the badges and achievements and I did just that. It was also suggested to improve the error messages and I did that too.

8 Testing

Regression testing was regularly conducted through the development of this project and the final regression tests are documented below.

8.1 Login Screen

Test ID	Action	Inputs	Expected Result	Actual Result	Test Result
1	No email or password		No Email Entered No	No Email Entered No	Pass

			Password Entered	Password Entered	
2	No email entered	Password: 1	No Email Entered	No Email Entered	Pass
3	No password entered	Email: p@p.com	No Password Entered	No Password Entered	Pass
4	Input invalid email and/or password	Email: p@p.com Password: 1	Email or Password are incorrect	Email or Password are incorrect	Pass
5	Input valid email and password	Email: y@y.com Password1234As!	Login Successful redirect to main menu	Login Successful redirect to main menu	Pass

8.2 Sign Up Screen

Test ID	**Action**	**Inputs**	**Expected Result**	**Actual Result**	**Test Result**
1	No username, email and password entered		No Username Entered No Email Entered No Password Entered	No Username Entered No Email Entered No Password Entered	Pass
2	No username entered valid password and email entered	email: a@a.com password:1234aS1!	No Username Entered	No Username Entered	Pass
3	No email entered valid password and username entered	Username: peter password:1234aS1!	No Email Entered	No Email Entered	Pass
4	No Password entered valid username and email entered	Username: peter Email: a@A.com	No Password entered	No Password entered	Pass

5	Valid Username entered no password or email entered	Username: peter	No Email Entered No Password Entered	No Email Entered No Password Entered	Pass
6	Valid email entered no username or password entered	Email: a@A.cp,	No Username Entered No password Entered	No Username Entered No password Entered	Pass
7	Valid Password no username or email entered	password:1234aS1!	No Username Entered No Email Entered	No Username Entered No Email Entered	Pass
8	Invalid email entered	Username: John Email: p password :As12!	Email is invalid	Email is invalid	Pass
9	Password less than 8 chars but has lower and upper char a number and a special character entered valid email and password entered	Username: Peter Email: a@a.com Pasword: 1As!	Password must be at least 8 characters	Password must be at least 8 characters	Pass
10	Password with no number entered valid email and username entered	Username: Peter Email: a@a.com Pasword: ppLL££aa	Password must have at least one number	Password must have at least one number	Pass
11	Password with no upper case letter entered valid username and email	Username: Peter Email: a@a.com Pasword: pp11££aa	Password must have at least one uppercase character	Password must have at least one uppercase character	Pass

	entered				
12	Password with no lowercase letter entered valid username and email entered	Username: Peter Email: a@a.com Pasword: PP11££AA	Password must have at least one lowercase character	Password must have at least one lowercase character	Pass
13	Password with no special character entered valid email and username entered	Username: Peter Email: a@a.com Pasword: PP11aa11	Password must have at least one special character	Password must have at least one special character	Pass
14	Valid Username Password and Email entered	Username: Peter Email:p@p.com Password: aaSS11!!	New user created and redirected to the login screen	New user created and redirected to the login screen	Pass
15	Login with the same credentials as test 14	Credentials from test 14	Login Successful redirect to main menu	Login Successful redirect to main menu	Pass

8.3 Forgot Password Screen

Test ID	**Action**	**Inputs**	**Expected Result**	**Actual Result**	**Test Result**
1	No email entered		No Email Entered	No Email Entered	Pass
2	Invalid email entered	Email: t	Invalid Email	Invalid Email	Pass
3	Valid email entered	Email: peteroluc@gmail.com	Password reset email sent and received and filled out and password was changed	Password reset email sent and received and filled out and password was changed	Pass

8.4 Logout Functionality

Test ID	Action	Inputs	Expected Result	Actual Result	Test Result
1	Logout option tapped	Logout option tapped	Logged out and redirected to the Login Page	Logged out and redirected to the Login Page	Pass
2	Press the back button after logout	Press the back button after logout	App closes	App closes	Pass

8.5 Navigation Functionality

Test ID	Action	Inputs	Expected Result	Actual Result	Test Result
1	Try all navigational options and see and document if any produce errors	Every navigation path	Correct page displayed	Correct page displayed	Pass

8.6 Log Action Functionality

Test ID	Action	Inputs	Expected Result	Actual Result	Test Result
1	Log all points and see if the correct messages get displayed and the correct points get added	Every log points function triggered	Correct message displayed and correct points added	Correct message displayed and correct points added	Pass
2	Try log more than 15 actions in a single day	Try Log 16 actions	Error message displayed	Error message displayed	Pass
3	Try log more	Try log 2	Error	Error	Pass

	than 1 action in 60 seconds	actions in 60 seconds	message displayed	message displayed	

8.7 Refill Station Screen

Test ID	Action	Inputs	Expected Result	Actual Result	Test Result
1	Tap the Refill Station option	Tap the Refill Station option	The map opens up focused on the user's current location and all the pins on the map are displayed	The map opens up focused on the user's current location and all the pins on the map are displayed	Pass

8.8 Leaderboard Screen

Test ID	Action	Inputs	Expected Result	Actual Result	Test Result
1	Navigate to the Leaderbaord	Navigate to the Leaderbaord	First 10 entries get loaded	First 10 entries get loaded	Pass
2	Press the next page button	Press the next page button	Displays the next then entries	Displays the next then entries	Pass
3	Press the back to page on button	Press the back to page on button	Displays the first10 entries again	Displays the first10 entries again	Pass
4	Tap on an entry in the leaderboard	Tap on an entry in the leaderboard	Pop Up of accounts details shows up	Pop Up of accounts details shows up	Pass
5	Tap on the filter icon	Tap on the filter icon	Filter menu appears	Filter menu appears	Pass
6	Chose Belarus from the filter	Chose Belarus from the filter	Only Belarus accounts get displayed on the leaderboard	Only Belarus accounts get displayed on the leaderboard	Pass

7	Choose Me from the filter	Choose Me from the filter	The users account gets displayed	The users account gets displayed	Pass

8.9 Profile Screen

Test ID	Action	Inputs	Expected Result	Actual Result	Test Result
1	Tap on the profile picture	Tap on the profile picture	Image Screen pops up	Image Screen pops up	Pass
2	Tap on the Bio edit it and press back on the keyboard	Tap on the Bio edit it and press back on the keyboard	Bio gets updated	Bio gets updated	Pass

8.10 Image Screen

Test ID	Action	Inputs	Expected Result	Actual Result	Test Result
1	Choose the from storage option, pick an image and click save	Choose the from storage option, pick an image and click save	Profile picture gets updated	Profile picture gets updated	Pass
2	Choose Capture Image, Take a photo and click save	Choose Capture Image, Take a photo and click save	Profile picture gets updated	Profile picture gets updated	Pass
3	Tap outside the options	Tap outside the options	The popup closes	The popup closes	Pass

8.11 Trophies Badges And Achievements Screen

Test ID	Action	Inputs	Expected Result	Actual Result	Test Result

1	Set points to 100	Set points to 100	Bronze Badge Appears	Bronze Badge Appears	Pass
2	Set points to 250	Set points to 250	Silver Badge Appears	Silver Badge Appears	Pass
3	Set points to 500	Set points to 500	Gold Badge Appears	Gold Badge Appears	Pass
4	Set points to 1000	Set points to 1000	Diamond Badge Appears	Diamond Badge Appears	Pass
5	Set all category log count to 1	Set all category log count to 1	All Novice Badges Appear	All Novice Badges Appear	Pass
6	Set all category log count to 5	Set all category log count to 5	All Apprentice Badges Appear	All Apprentice Badges Appear	Pass
7	Set all category log count to 10	Set all category log count to 10	All Adept Badges Appear	All Adept Badges Appear	Pass
8	Set all category log count to 25	Set all category log count to 25	All Expert Badges Appear	All Expert Badges Appear	Pass
9	Set all category log count to 50	Set all category log count to 50	All Master Badges Appear	All Master Badges Appear	Pass
10	Set all category log count to 100	Set all category log count to 100	All Legend Badges Appear	All Legend Badges Appear	Pass
11	Set all cation logs to 5	Set all cation logs to 5	All Bronze badges appear	All Bronze badges appear	Pass
12	Set all cation logs to 15	Set all cation logs to 15	All Silver badges appear	All Silver badges appear	Pass
13	Set all cation logs to 25	Set all cation logs to 25	All Gold badges appear	All Gold badges appear	Pass

9 Problems & Learning Outcomes

Listed below a number of problems and learning outcomes derived from these problems. Some of the problems were solved and some were not possible to solve due to the Covid-19 pandemic.

9.1 Cross Platform Mobile App Development

The initial plan for the application was to develop a cross platform mobile application for Android and iOS as they control almost all of the market for mobile devices.

9.1.1 Problems

The problems that arose were that a macBook and an iOS developer license was needed to test and implement certain features on the iOS side of the application. Initially the plan was to purchase a macBook and the developer license but this was not possible as I have lost my job due to the Covid-19 pandemic and was short on finances. The application that was developed was developed for both iOS and Android but there a few things that need to be fixed, implemented and tested on the iOS side but this should not be a problem and only a couple of days of work at most as almost all the code is shared across iOS and Android and the only things that need to be added are the iOS native calls which are not possible to implement or test without an iOS device and a developer license.

9.1.2 Learning Outcomes

A developers license and a macbook is needed to develop applications for iOS which is a major disadvantage that iOS has compared to Android as this repels developers from developing iOS applications. This is the main reason why there are a ton more Android applications on the Playstore than there are iOS applications on the App Store.

9.2 Xamarin Forms & C#

Xamarin forms and C# were used to develop the mobile application.

9.2.1 Problems

During my time developing with Xamarin Forms and C# I have found out that as Xamarin Forms is Cross platform it is slower than other alternative native languages. Xaml is also a lot slower than using just C# for the front end and back end.

9.2.2 Learning Outcomes

I have learnt that it is best to use C# to implement the most of the front end as you can as pure code is faster than a markup language like xaml. During this project I have realised this pretty late and only change what I could to C#.

9.3 Firebase Database

The Firebase Database was used to store users data for this project.

9.2.1 Problems

The initial problems with firebase arose with implementing the API and setting up the project in firebase. A lot of research and work had to go into this as it was very new to me and I have never used JSON before. Figuring out the functions that perform CRUD operations to the database and how to structure the database was also very challenging to me. Once this was figured out I faced the challenge of processing the data in lists as the data was in JSON format as mentioned before. The final challenge using firebase was that it is a real time database and can take some time to load as things are done and loaded at real time.

9.2.2 Learning Outcomes

I have learned a lot while working with the Firebase database and JSON. I have learned how to set up my project and implement CRUD functions into the database and how to structure a JSON database. I have also learnt most importantly that I should have implemented a cache in the database and pre loaded all the data and only posted it to the database periodically and not instantly and do the same for downloading the data. This would have made the application much faster, severely reducing loading time. One I realised this it was too late in the project to amend it. The reason behind why it was figured out too late was that I have only used a small amount of data to test my application as I was developing it. A lesson for the future is to test my app with large amounts of data and test the performance and loading times.

9.3 Firebase Storage

Firebase Storage was used to store the uploaded images from the users.

9.3.1 Problems

Similar to the firebase database it was difficult to figure out how to store images into the storage in firebase and then retrieve them.

9.3.2 Learning Outcomes

I have learnt that the best way to do so in my opinion is to create a folder with a name the same as the user UID and then put their files there. This is the easiest and fastest way to locate the users files quickly.

9.4 Firebase Authentication

Firebase Authentication was used to safely store the users credentials.

9.4.1 Problems

There were a number of problems with using this approach. Firstly setting it all up was difficult and the database seemed not to work and it took me a long while to figure out why that was.

9.4.2 Learning Outcomes

I have learned that Firebase Authentication does not allow passwords that are shorter than 6 characters or should I say it does allow them but just does nothing with them and does not tell the developer that this is not acceptable. This is a major flaw in Firebase Authentication in my opinion as I spent a long time figuring why my user was not being saved and I know a lot of other students using Firebase also faced this issue. Strong validation was implemented on the password fields and then firebase takes this password and prepends a salt to it, hashes it and saves it in the database.

9.5 Google Maps

Google Maps and the Google Maps Api was used to display the pins where a user can refill their water bottle on a map.

9.5.1 Problems

The problem faced here was figuring out a way of how to store and display these pins as I have never implemented maps on any projects that I have worked on previously.

9.5.2 Learning Outcomes

The solution to this problem was quite simple but took a while to figure out and implement. The details about a pin were stored in the database including their latitude and longitude. Then this data is loaded into the application as a list and the pins are taken from that list and displayed on the map.

9.6 Images

Storing, Displaying and Capturing Images.

9.6.1 Problems

The issue was figuring out how to display images, how to take them and how to capture images.

9.6.2 Learning Outcomes

I have found out that to display images they need to be saved as embedded resources in the solution and a class has to be implemented to process this image. Images can also be displayed by downloading them directly from firebase but this is slower than having them saved locally. Functions had to be made to Capture and choose images from storage by using bytestreams.

9.7 Error Handling

Error handling had to be implemented in this application.

9.7.1 Problems

Initially when implementing this application in since scenarios errors and exceptions were thrown and this had to be handled.

9.7.2 Learning Outcomes

The best solution that I have used for this was using try and catch blocks. I have previously not used these and after learning how to use them find them amazing and very useful. I have implemented these try and catch blocks to try to execute a piece of code and when that code throws an exception and executes something else that is appropriate to the exception thrown. This proved to be a very efficient and reliable way to handle any errors that arose during the app development. I also signed my app into app center to monitor crashes and provide reports on my application. This was done by implementing their API and downloadinge their Nuget Packages as well as refactoring the code.

9.8 Security and Validation

Every application needs to be secure and validated and my application is no different.

9.8.1 Problems

Initially the application allowed any password to be entered and any email to be entered. In addition a user could get back into the application by pressing the back button after logging out. This hat to be fixed. Malicious users had to also be mitigated.

9.8.1 Learning Outcomes

I have implemented features that I have learnt in Secure Applications Development and implemented strong validation on the password and email fields. The input email must be valid and the input password must be at least 8 characters long and contain at least one upper and lower case letter, a number and a special character. The error messages that display to the user must not give away useful information to an attacker. The navigation stack is reset after logging out which makes it impossible to return back to the application with the back button.

When it comes to malicious users the application only allows a maximum of 15 logs per day and at most 1 log every 60 seconds to prevent fake entries into the database and spamming in order to get a high placement on the leaderboard.

9.9 Performance

Performance had to be improved in my application.

9.9.1 Problems

I developed my application to load data whenever it needs it but later on in the project my supervisor pointed out and I realized that when the data gets larger this can cause a log waiting time to load certain pages.

9.9.1 Learning Outcomes

I have learnt that the best way to build an app is to pre load the data and I refactored my app to do so wherever possible. The apps performance improved greatly and the achievements and badges changed from loading for a couple of seconds to instantly appearing.

10 Project Review

The initial plan for this project was to develop a cross platform application that used gamification of eco friendly activities, had a water refill station locator tool and had a food tracker tool. In addition to this there were meant to be 2 versions of this application. One for the general user and one for companies. The company version would have been the same as the original version but with one additional feature. It would allow the company to monitor and reward their employees for being environmentally friendly. It would have come with an additional website to do this. I have realized throughout this project that this idea was way too ambitious and I would have needed an entire development team to get it finished in time. In order to release a working product the original plan was refactored quite a bit but still resulting in a very good application with the possibility of future development to add in the functionalities and versions mentioned above.

10.1 Achieved

A huge amount was achieved in this project. The core functionality of this project has been achieved bar the food tracker functionality which is not that important to the overall project anyway. In addition some bonus functionalities were added as the project went on.

A user is able to sign up for the application using a username, a valid email and a strong password. Their account is then created on Firebase and it is very secure. The user can also request to be sent a password reset email by entering their email. The user can log into the application.

Once logged into the application the user can do a multitude of things. Firstly a user can see their level and start logging eco friendly actions that they perform using the Eco Actions section. They select a category and the action that they have performed and information regarding that action is displayed and a user can choose to log it. A user is limited to a maximum of 15 logs per day and a maximum of 1 log every 60 days to protect against malicious users as mentioned above.

A user can also use the Refill Stations tool to load a map focused on their current location which displays the nearest water refill station to them. Not all refill stations are implemented yet as more research would have to be carried out but enough are loaded around Carlow.

A user can also load the leaderboard which loads 10 entries per page and the user can view the following pages and also return to the first page. The user can tap on each individual entry and more details about that user get displayed. The user can also filter the leaderboard by nationality or view their own position in it.

The user can also unlock different mosaic pieces in the profil page by leveling up and there are 20 images to unlock in total. The user can upload a new profile picture by either taking an image or selecting on from the gallery and also enter a bio.

Finally the user can look at and earn 4 trophies, 72 Badges and Over 260 Achievements. There are 12 badges and these are live badges with 6 levels to each of them for the user to progress and unlock the next level of the badge. There are 80+ Achievements with 3 levels to each achievement and these are Gold, Silver and Bronze. The Trophies are earned by points. The Badges are unlocked by logging actions in their respective categories and finally the achievements are unlocked by performing individual actions. Each version of the Badge or Achievement is unlocked after logging a certain amount of actions.

The project also works on all the tested Android devices and emulator all the way from Android 5.1 to the current version of Android 11.

10.2 Not Achieved

Due to the Covid-19 pandemic I was unable to gain access to iOS devices and a macBook to deploy and test the iOS version of the application. Saying this all the necessary procedures were carried out to make the application work on iOS. Everything that was possible to do without actually running the application was done. As most of the code is shared across the platforms I would estimate that only a couple of days of work of maximum a week is needed to make the iOS version work as only the native calls have to be implemented on the iOS side and then tested.

Due to time constraints and because the food tracker feature did not seem that lucrative or important to the application it was also dropped.

Due to the time constraints and as the original idea was simply too large for a single student to complete. The second version of the application was also dropped out of scope. Although not too much work would be needed to make the second version of the application as it would have been built on top of the original application.

10.3 What Would Have Been Achieved With More Time?

With more time I would have been able to achieve the second version of the application. I would have built it on top of the first version of the application. The leaderboard would have changed into a company leaderboard that could have been filtered by different teams. An accompanying website would have been built for the manager to view the statistics of each employee and would get informed when an employee has earned a reward. The manager would then send an award to the employee via an email and the employee would have been notified on their mobile device. The food tracker feature could also have been implemented.

All in all the estimated additional time required for this would have been roughly three wix. This would include deploying and testing the iOS device if a macbook was to be obtained.

10.4 How Would I Start From The Beginning Again?

If I was to start from the beginning possessing all the knowledge that I currently possess and knew that I would be financially hindered by the repeated Covid-19 lockdown I would have only developed an application for Android only. This is because native app developed allows the creation of a better application for the native platform compared to cross platform mobile app development. Native applications usually have better performance and more functionality can be implemented on them. Saying this I wanted to develop an app for both iOS and Android to target as much of the market as possible.

In addition to this, I would have started testing my application from the beginning of the project and started coding much sooner to face less stress at the end of the project but this is easy to say in hindsight.

I would have also implemented a leaderboard that filters by radius rather than nationality but most importantly I would have developed the app focusing on the storage and loading of data to increase the performance of the application and ran it with large amounts of data. I would have implemented a local cache that would download and save everything and only send the data to firebase periodically rather than sending and updating everything instantly like the way the current application is implemented. This causes a strain on performance as the app is constantly processing and sending data. A local cache would have definitely been the way to go.

10.5 Differences From Initial Proposal And Design.

As previously mentioned the the second version of the application and the food tracker tool was placed out of scope for the final application due to time constraints and the unrealistic goals set at the beginning of the project.

11 Conclusion

In conclusion, this project report has documented the Green Quake project entirely going into specific detail in each area discussed previously. This document began by describing both the non technical and technical aspects of the project followed by the UI-UX designs and the database structure of the project. This clearly explained what the project is about and how it looks and works.

Following on, this document spoke about how the agile process was thoroughly followed throughout the project, how feedback was gathered, listened to and implemented into the application and all the testing that was performed on the application throughout its development.

Finally this document ended with a personal reflection on the problems encountered during this project and how they were solved and the lesson that has been derived from said problems. The document concluded by reviewing the project and speaking about what was and wasn’t achievend and other possibilities.

12 Plagiarism Declaration Form

- 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.

1. Student **Name** : Peter Lucan
2. Student **Number** : C00228946
3. Student **Signature** : Peter Lucan

13 References

[1] BeFunky. n.d. Photo Editor | BeFunky: Free Online Photo Editing and Collage Maker. [online] Available at: <https://www.befunky.com/> [Accessed 21 April 2021].

[2] Gravit Designer. n.d. Gravit Designer - Start Designing Graphics or Editing Icons and Logos for Free. [online] Available at: <https://www.designer.io/en/> [Accessed 21 April 2021].

[3] Accredible.com. n.d. Badges. [online] Available at: <https://www.accredible.com/digital-badges/?utm_source=badgedesign&utm_medium=referral&utm_campaign=badge_design_logo&__hstc=141085997.10eaa744625351401bbea1522658d4e7.1617219154821.1619035213907.1619039810695.7&__hssc=141085997.2.1619039810695&__hsfp=1848490052&_ga=2.10731964.966390800.1619035211-597170466.1617219150> [Accessed 21 April 2021].

[4] n.d. [online] Available at: <https://miro.com/about/> [Accessed 21 April 2021].

[5] Atlassian. 2021. How to Use Jira Software | Official Buyer & User Guide. [online] Available at: <https://www.atlassian.com/software/jira/guides> [Accessed 28 April 2021].

[6] n.d. [online] Available at: <https://pixabay.com/> [Accessed 21 April 2021].

[7] n.d. [online] Available at: <https://www.pexels.com/> [Accessed 21 April 2021].

[8] Unsplash.com. n.d. Beautiful Free Images & Pictures | Unsplash. [online] Available at: <https://unsplash.com/> [Accessed 21 April 2021].

[9] Flaticon. n.d. Flaticon, the largest database of free vector icons. [online] Available at: <https://www.flaticon.com/> [Accessed 21 April 2021].

[10] Iconfinder. n.d. 5,425,000+ free and premium vector icons - Iconfinder. [online] Available at: <https://www.iconfinder.com/> [Accessed 21 April 2021].

[11] Icons8.com. n.d. Free Icons, Clipart Illustrations, Photos, and Music. [online] Available at: <https://icons8.com/> [Accessed 21 April 2021].