DRIVECARE CANCER SUPPORT TRANSPORT ORGANISER

Project Report

Student: Adam Lambert
Student Number: C00257510
Supervisor: Dr. Christopher Staff

Submission Date: 19/04/2024

Ollscoil Teicneolaíochta an Oirdheiscirt

South East Technological University

Table of Contents

Abstract

This report details both the personal and technical aspects of the development of Drivecare, a transport organisation web application for local Cancer Supports around Ireland. The importance of a simple and intuitive platform for these Cancer Supports to utilise to ensure that patients arrive to their cancer treatments is essential. DriveCare aims to deliver this functionality, transferring the currently tedious process in place to a more modern online implementation.

Project Overview

In many cancer support centres located around Ireland, transport to and from cancer treatments is the number one service that they can offer to improve the lives of cancer patients in their local communities. This service is essential for many, being the only way that they can feasibly receive their cancer treatments from hospitals that can often be located quite some way from their homes. As this is such an important service, it is shocking to learn that in most cases the staff of these cancer support centres operate a paper-based transportation system, logging patients (referred to as clients in this document), volunteer drivers and upcoming trips in a paper diary.

After years of operation, these paper diaries are full of confidential client information that is near impossible to navigate, adding unnecessary difficulty to the lives of the office staff (transport administrators) within these support centres. DriveCare aims to provide a solution to this problem, offering a web-based application which allows transport administrators to access and maintain client information using a web interface, and to organise hospital trips for multiple clients at the click of a button.

The Client

To deliver as successful of a product as possible, I met with the staff of my local Cancer Support. Over several meetings, I discussed the current manual process in place to manage transportation within their Cancer Support to gain as much of an understanding as possible as to how the process currently works.

They broke down in detail the amount of work, effort, and time it takes to complete these tasks that on the surface look like something that should be simple and quick. We talked about what their ideal scenario would be for an automated replacement, and after many meetings we had agreed on a system that would work for both of us.

Having a prospective client that I could meet with on a regular basis was invaluable throughout the entirety of the planning phase of this project. They provided me with answers to any questions that I had in relation to the current system in place and in what ways it could be improved. This valuable nature continued right throughout the entire process of development, allowing me constant access to feedback on designs, certain implementations, and general functionality of the application.

Project Planning

Although the discussions with the client were extremely informative in terms of the necessary functionality, all decisions of the actual implementation of the project were left in my hands. The biggest part of this decision making was deciding on the technologies to use to develop the project.

Ultimately, I settled on the Django REST Framework (DRF) for the backend of my application. Django is a Python web framework that is stable, secure, regularly updated and has a strong community built around it. The DRF allowed me to leverage Django to build web API's that I could use as a connection between my frontend and my chosen database, as well as the API's necessary to produce the functionality that I desired. Having used a very small amount of Angular during my 3rd year work placement, it had piqued my interest, and I ultimately chose it for the frontend of my project. Research led me to understand that Angular and the DRF integrated quite well together for making a comprehensive application, which further reinforced my decision. The final aspect was choosing a database, where Supabase was chosen due to its ability to integrate with Django. More information on these choices can be found in the Functional Specification and Design Document of this project.

Dealing with an actual client one on one was not something that I had ever done before. Managing these discussions with the client and translating the ideas that we decided on into a cohesive, tangible product has taught me valuable skills for the future. The entire process of planning a project of this scale was also something that I was unfamiliar with, and looking back I feel proud of how I handled the choosing of technologies and issues that arose from these choices. I feel much more confident in my ability to build a project up from nothing but verbal ideas, through an entire planning process to an actual product that both myself and the client are happy with.

Technical Aspects

Project Differences

DriveCare is an administrative web application used by transport coordinators within the cancer support to view and maintain driver and client information and to manage planned hospital trips. Originally, this would be accompanied by an Android application to be utilised by volunteer drivers and clients, however after further consideration it was decided to drop the native Android app in favour of developing DriveCare as a web application which could be used on device. This freed up time and removed the constraint of being stuck in the Android ecosystem. Instead, one project could be used for all devices with minimal code changes required.

The Google Routes API was ultimately replaced by the Google Directions API, a slightly older but much more documented API that allowed for similar functionality. This choice was made after much difficulty attempting to work with the project, and with many more resources available to learn about the Google Directions API, it became clear that the Directions API was a better choice for this project. However, as the Routes API continues to improve in the future, there is room to replace the Direction API within this project in favour of its more modern alternative.

The SMS integration was put on hold after much discussion with the client and the providers of the API 47Elks. Due to personal financial cost that I would incur to simply test the API, it was deemed more suitable to leave this functionality for future production if the application was monetarily supported.

Technical Problems

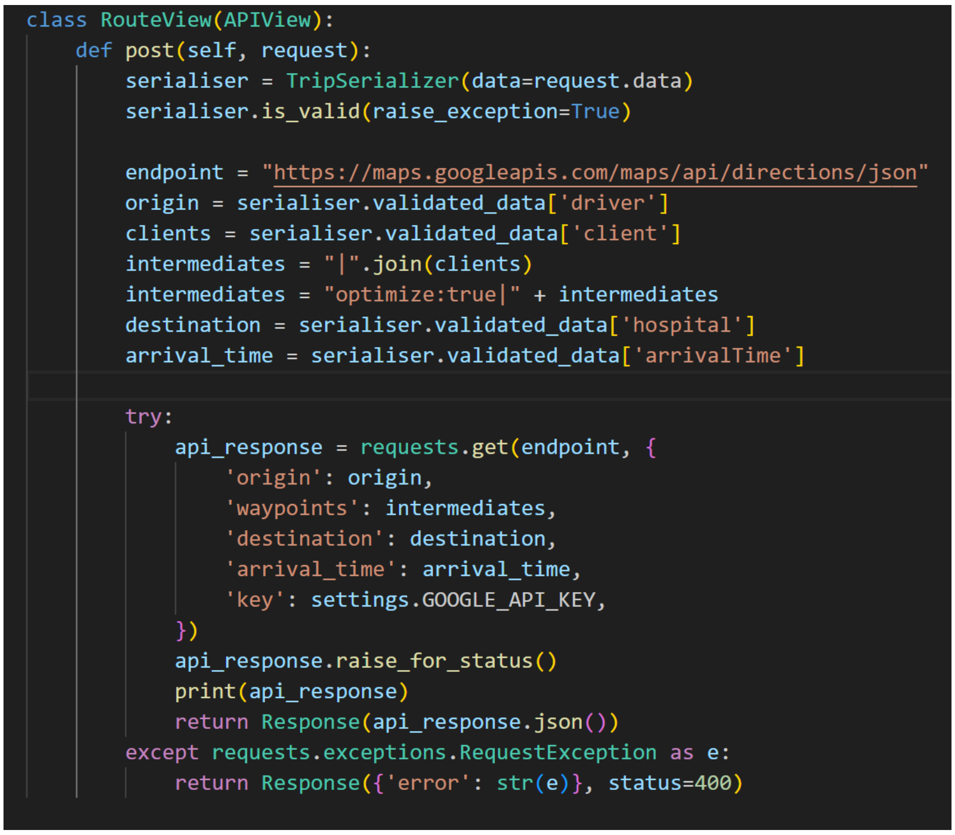
Many technical problems were met throughout the development of this project. The largest of these being a compatibility issue discovered between Angular 17 and the Django 5 version of the DRF that I was using. This led to issues when attempting to integrate the two technologies together, resulting in a halt to the project. Due to how much work had already been completed in those versions of the technologies and being completely blindsided by the problems, I had to take a step back from the project temporarily and began to doubt my choices of technology. However, returning from this hiatus I proceeded by downgrading my Angular version to Angular 16 and completely overhauling the Angular codebase to function with this downgraded version. While I lost quite a lot of development time to this problem, it gave me valuable insight into how to manage my own personal self as well as the technologies when an issue like this arises. As these issues can occur quite late into a project without any warning beforehand, this knowledge is very valuable to have for any future endeavours.

Other than this major derailment, no massive issues occurred throughout the project. The occasional issue arose due to my lack of initial understanding of these new technologies, however as I began to become more proficient with them these small issues were slowly phased out.

Module Descriptions

The main module within this project is the Google Direction API. This API takes a HTTP request containing an origin location, a destination location and intermediate ‘waypoint’ locations and returns JSON formatted directions between these locations. Accessing this API involved setting up a Google Cloud project, which required the setting up of a Google Developer account. This gave me access to a Google API key, which had to be included with every API call made to the Directions API.

Within my project, this module is used to generate a route for drivers to take to collect clients of the cancer support and bring them to their hospital visits most efficiently. Using the Directions API’s inbuilt ‘optimize:’ call allowed me to order the client waypoints in a way that was most efficient to travel as shown in Figure 1.



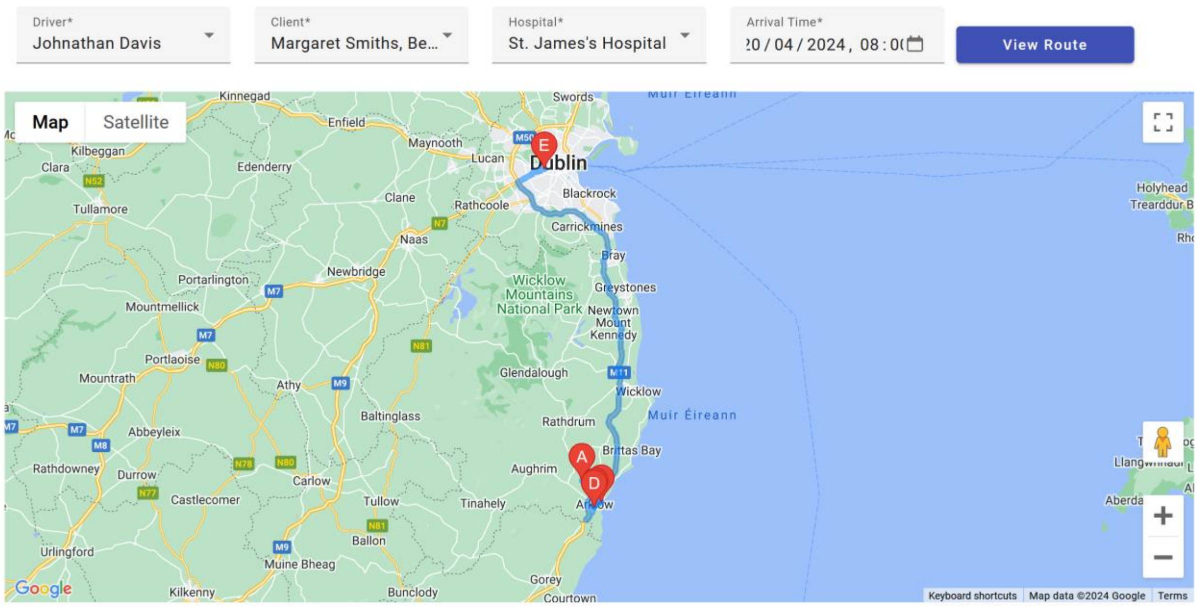
```python
class RouteView(APIView):
    def post(self, request):
        serialiser = TripSerializer(data=request.data)
        serialiser.is_valid(raise_exception=True)

        endpoint = "https://maps.googleapis.com/maps/api/directions/json"
        origin = serialiser.validated_data['driver']
        clients = serialiser.validated_data['client']
        intermediates = "|".join(clients)
        intermediates = "optimize:true|" + intermediates
        destination = serialiser.validated_data['hospital']
        arrival_time = serialiser.validated_data['arrivalTime']

        try:
            api_response = requests.get(endpoint, {
                'origin': origin,
                'waypoints': intermediates,
                'destination': destination,
                'arrival_time': arrival_time,
                'key': settings.GOOGLE_API_KEY,
            })
            api_response.raise_for_status()
            print(api_response)
            return Response(api_response.json())
        except requests.exceptions.RequestException as e:
            return Response({'error': str(e)}, status=400)
```

Figure 1. Call from the DRF to the Google Directions API with the specified locations.

The resulting JSON return from this module was parsed in my Angular code and used to produce the route as seen in Figure 2.



Driver* Johnathan Davis

Client* Margaret Smiths, Be...

Hospital* St. James's Hospital

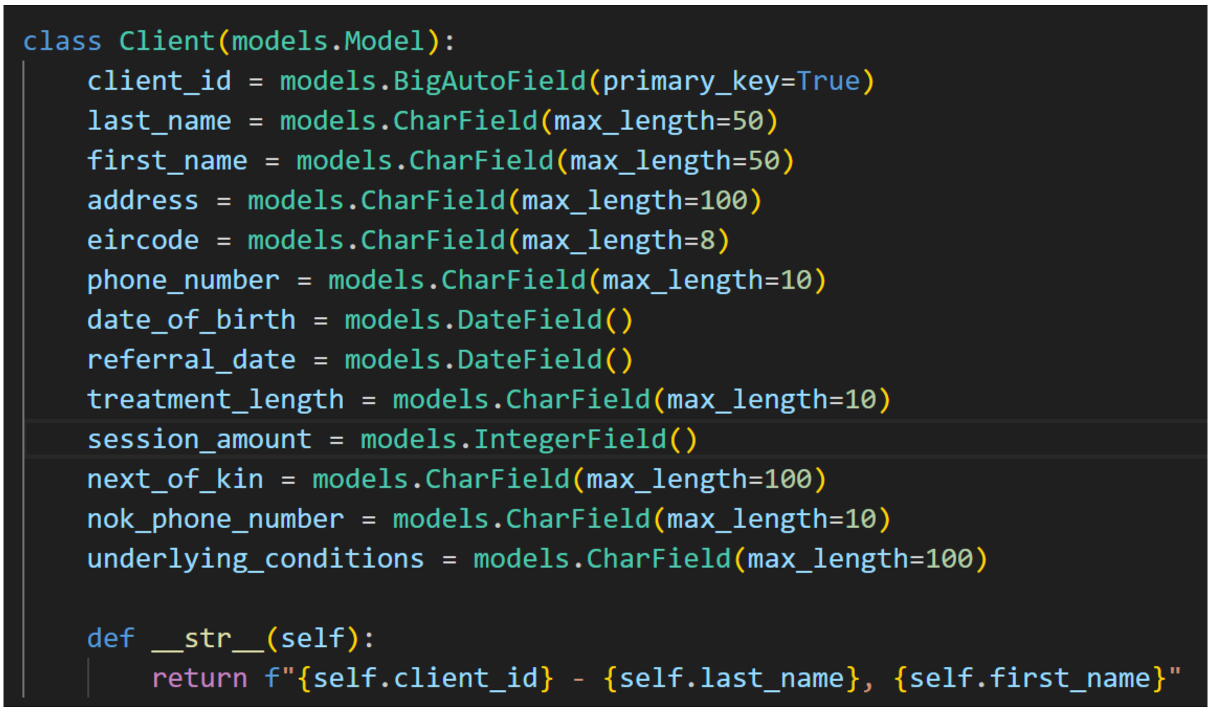
Arrival Time* 20 / 04 / 2024, 08 : 0(

Figure 2. Route received from Directions API after being parsed in Angular code.

Data Structures

This project mostly utilised the DRF's ability to handle models, serialise them and pass them directly to the database using inbuilt functions. For this functionality to work, the DRF had to be informed of the database language type, in this case PostgreSQL, and the database's secret key, which is generated automatically upon creation of the database in Supabase.

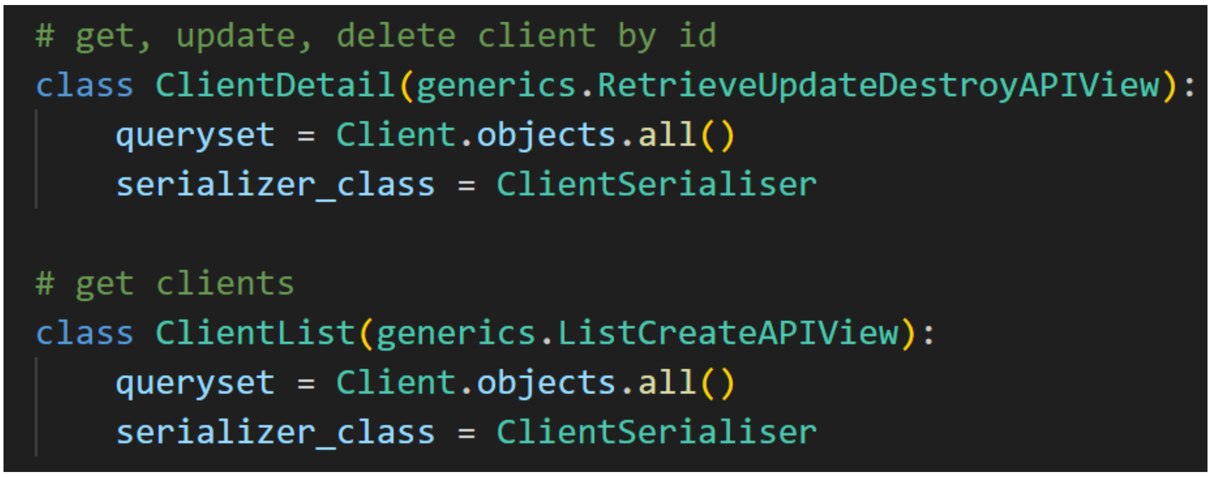
Figure 3 shows an example of one of these models, which after being sent information from the Angular frontend, could be serialised, and used to manipulate the database with the DRF's inbuilt functions.



```python
class Client(models.Model):
    client_id = models.BigAutoField(primary_key=True)
    last_name = models.CharField(max_length=50)
    first_name = models.CharField(max_length=50)
    address = models.CharField(max_length=100)
    eircode = models.CharField(max_length=8)
    phone_number = models.CharField(max_length=10)
    date_of_birth = models.DateField()
    referral_date = models.DateField()
    treatment_length = models.CharField(max_length=10)
    session_amount = models.IntegerField()
    next_of_kin = models.CharField(max_length=100)
    nok_phone_number = models.CharField(max_length=10)
    underlying_conditions = models.CharField(max_length=100)

    def __str__(self):
        return f"{self.client_id} - {self.last_name}, {self.first_name}"
```

Figure 3. Example model from the DRF codebase.



```python
# get, update, delete client by id
class ClientDetail(generics.RetrieveUpdateDestroyAPIView):
    queryset = Client.objects.all()
    serializer_class = ClientSerialiser

# get clients
class ClientList(generics.ListCreateAPIView):
    queryset = Client.objects.all()
    serializer_class = ClientSerialiser
```

Figure 4. Inbuilt DRF calls to manipulate the database.

Testing

Most of the application's testing was done manually. A large part of this manual testing involved allowing the client to use the application in a way that they normally would. This allowed me to understand not only if any functionalities were not working, but also if any aspects of the UI or any flows that a user followed needed to be modified to be more intuitive.

Personal Aspects

In this section, I would like to discuss my personal experiences with the Final Year Project as a whole, how I approached it, where this lead me and how I would approach it again knowing what I know now.

Challenges Encountered

As is to be expected of anything in life, the Final Year Project was not devoid of plenty of problems and challenges, some of which were anticipated and some of which were not. The first challenge that I was faced with in this project was the actual conception of the project itself, coming up with an idea that I felt was practical and could be applied to a real-world problem, while also being something that was achievable in the available timeframe. Ensuring that I chose a project that I was interested in was also essential, as I didn't want to reach a point where I fell out of interest with the chosen topic and cause my work to suffer as a result. This led me to choose something that I felt would be beneficial to something that I cared about in my community, and as I already had contact with some of the staff in my local cancer support, the choice practically made itself. This gave me the advantage of essentially having a 'client' who I was developing the project for, allowing me to get second opinions on modifications made, such as certain parts of the project that were not essential when time constraints began to arise.

A challenge that I admittedly imposed upon myself was choosing to develop DriveCare in languages that I was not entirely familiar with. Having limited experience with Angular from my 3rd Year internship, no experience with Django or the Django Rest Framework, and no experience with a Postgres database like Supabase, a large portion of my project development time was spent familiarising myself with the basics of each language/service and attempting to expand on this basic knowledge in order to achieve the deliverables that I had set out for myself. While self-imposed, I had greatly underestimated the effort that would be required to utilise these new languages in the way I intended to, impeding the amount of these deliverables that I could manage to complete. However, I feel that I have made a large amount of progress in understanding these new languages and how they are used from where I was before starting the project, and I hope that pushing myself to experience challenges like this now will make the process of learning new languages in my future endeavours a much smoother experience.

Time management was another issue that I encountered continuously throughout the project. Between unexpected problems like the ones mentioned in this section and this being my first large scale project, making time estimation a difficult procedure that I was in no way familiar with, I often underestimated the resources that would be required to complete certain parts of the project, and failed to meet deliverables in the time frame that I wanted to. This coupled with the constant need to keep up with work and study for brand new modules started in semester two of 4th Year resulted in constant crunches to produce the final product. This project has certainly changed my view of time management required for projects and has helped me to understand that much more time needs to be allocated to any aspects of a project than is expected. This project has also demonstrated to me that unforeseen circumstances must also be considered, with a contingency plan in place to handle them as best as possible, with issues such as illness popping up multiple times during the project's development.

The final, and probably most impactful, personal challenge that I encountered was the mental strain of the entire project. I often reached points where I felt that I had nowhere left to go and that I would never overcome the issues that I was encountering, continually piling onto each other as the big picture continued to look even bigger. As the project continued, I began to take solace in the small aspects that I did get working. This project gave me a better understanding of the fact that nothing is ever going to work as we expect it to, but there is much more to be learned from trying and failing something repeatedly until it works than there is in something working immediately, although that did feel nice when it happened.

What I Achieved

Other than the actual product that I delivered; I also achieved many learning outcomes throughout the duration of the project. I have gained valuable experience working with Angular, Django, APIs and other technical aspects of the project. Taking into consideration that these technologies had to be learned essentially from scratch, I am proud of how much more comfortable I have gotten with using these technologies, and I feel that it has given me more confidence to work with technologies that I am unfamiliar with.

I also feel that I have made great personal growth in areas such as time and stress management and important decision making as discussed in the personal challenges section above. I feel that I have vastly improved as both a software developer and a person over the course of this project despite the many challenges that I encountered. If anything, these challenges have enhanced this learning even further, as challenges will be encountered in all walks of life.

What I Would Do Differently

If I was to tackle this project again, I would certainly begin to vastly overestimate exactly how much time a certain aspect of the project will take, as I often found myself underestimating the time required to complete something, which led to constant dragging of certain problems over to what should have been the next iteration of the project. The time management skills that I have obtained from this project would certainly be put to use if I attempted it again.

Another thing that I would do differently is try to stop myself from getting caught up in one specific section of the project. I lost quite a great deal of time attempting to produce a solution to the travelling salesman problem, a problem which has been solved many times before. This extra time may have benefitted me greatly in later point of the project.