BSc (Hons) Computer Science

IN3007 – Final Year Project Report



Car Park management system

Written and submitted by Claudiu Andrei

Email: [claudiu.andrei@city.ac.uk](mailto:claudiu.andrei@city.ac.uk)

ACZH866

Table of Contents

[BSc (Hons) Computer Science 1](#_Toc72205918)

[IN3007 – Final Year Project Report 1](#_Toc72205919)

[Car Park management system 1](#_Toc72205920)

[Written and submitted by Claudiu Andrei 1](#_Toc72205921)

[Email: 1](#_Toc72205922)

[ACZH866 1](#_Toc72205923)

[Acknowledgements 7](#_Toc72205924)

[Chapter 1 - Introduction 8](#_Toc72205925)

[1.1 Project Overview 8](#_Toc72205926)

[1.2 Problem to be solved 8](#_Toc72205927)

[1.3 Project Objectives 8](#_Toc72205928)

[1.4 S.M.A.R.T. Objectives 9](#_Toc72205929)

[Chapter 2 - Output 11](#_Toc72205930)

[2.1 WebPortal Application 11](#_Toc72205931)

[2.2 Number Plate Recognition and Source Code 11](#_Toc72205932)

[2.3 Database 12](#_Toc72205933)

[Chapter 3 - Literature Review 13](#_Toc72205934)

[3.1 Overview 13](#_Toc72205935)

[3.2 Digital Accessibility 13](#_Toc72205936)

[3.3 Edge Detection 13](#_Toc72205937)

[3.4 OpenCV (Python) 14](#_Toc72205938)

[3.5 Existing Tools and Programs 14](#_Toc72205939)

[3.5.1 Neural Labs VPAR 15](#_Toc72205940)

[3.5.2 OPENALPR by Rekor 15](#_Toc72205941)

[Chapter 4 - Method 16](#_Toc72205942)

[4.1 Requirements Specification 16](#_Toc72205943)

[4.2 Analysis 16](#_Toc72205944)

[4.3 Software Development Methodology 17](#_Toc72205945)

[4.4 Tools 18](#_Toc72205946)

[4.5 Languages and framework 18](#_Toc72205947)

[4.6 Design 18](#_Toc72205948)

[4.6.1 Use case specifications 19](#_Toc72205949)

[4.6.2 Use Case diagram 19](#_Toc72205950)

[4.6.3 Entity relationship diagram 19](#_Toc72205951)

[4.6.4 Wireframes 19](#_Toc72205952)

[4.7 Web Portal 20](#_Toc72205953)

[4.7.1 Login page 20](#_Toc72205954)

[4.7.2 Register Page 20](#_Toc72205955)

[4.7.2.1 Password 20](#_Toc72205956)

[4.7.2.2 Username 21](#_Toc72205957)

[4.7.2.3 Registration Number 21](#_Toc72205958)

[4.7.2.4 Email address 21](#_Toc72205959)

[4.8 Python Number Plate recognition system 21](#_Toc72205960)

[4.8.1 Image processing 21](#_Toc72205961)

[4.8.2 Object Detection 21](#_Toc72205962)

[4.8.3 Character recognition 22](#_Toc72205963)

[4.8.4 Implementation in the web portal 22](#_Toc72205964)

[4.9 Pillow 22](#_Toc72205965)

[Chapter 5 - Results 23](#_Toc72205966)

[5.1 Methodology 23](#_Toc72205967)

[5.2 Analysis 23](#_Toc72205968)

[5.3 Tools 23](#_Toc72205969)

[5.3.1 Programming languages and ide 23](#_Toc72205970)

[5.3.1.1 Tools used for the Python number plate recognition system 23](#_Toc72205971)

[5.3.1.2 Tools Used for the Web Portal 24](#_Toc72205972)

[5.3.2 Model-template-views architectural pattern 24](#_Toc72205973)

[5.3.3 Bootstrap 25](#_Toc72205974)

[5.3.4 Setting up the project and working environment 25](#_Toc72205975)

[5.3.4.1 Number Plate Recognition System 25](#_Toc72205976)

[5.3.4.2 Web Portal 26](#_Toc72205977)

[5.3.5 Django Syntax 28](#_Toc72205978)

[5.3.6 Wireframes 28](#_Toc72205979)

[5.3.7 Use Case Diagram 29](#_Toc72205980)

[5.4 Implementation 29](#_Toc72205981)

[5.4.1 Python Recognition System Prototype 29](#_Toc72205982)

[5.4.2 Web Portal 37](#_Toc72205983)

[5.4.3 Implementation of the Web Portal 38](#_Toc72205984)

[5.4.3.1 Base File 38](#_Toc72205985)

[5.4.3.2 Applications and Routes 40](#_Toc72205986)

[5.4.3.3 Authentication 41](#_Toc72205987)

[5.4.4 Database Development 41](#_Toc72205988)

[5.4.5 Edit User Profile 42](#_Toc72205989)

[5.4.6 Editing the profile Image 43](#_Toc72205990)

[5.5 Parking Sessions Logic 43](#_Toc72205991)

[5.5.1 Step 1 44](#_Toc72205992)

[5.5.2 Step 2 44](#_Toc72205993)

[5.5.3 Step 3 44](#_Toc72205994)

[5.6 Admin Panel 45](#_Toc72205995)

[5.7 Testing 45](#_Toc72205996)

[Chapter 6 - Conclusion and Discussion 45](#_Toc72205997)

[6.1 Main objective 45](#_Toc72205998)

[6.2 Sub objectives 46](#_Toc72205999)

[6.2.1 Registration and Login 46](#_Toc72206000)

[6.2.2 Admin panel 46](#_Toc72206001)

[6.2.3 Plate Recognition 46](#_Toc72206002)

[6.2.4 Parking session automation 46](#_Toc72206003)

[6.3 Future Work 46](#_Toc72206004)

[6.4 Knowledge Gained 46](#_Toc72206005)

[6.5 Conclusion 47](#_Toc72206006)

[Glossary 47](#_Toc72206007)

[References 48](#_Toc72206008)

[Appendix A – Project Description Document 53](#_Toc72206009)

[Project proposal 55](#_Toc72206010)

[Problem to be solved 55](#_Toc72206011)

[Project Objectives 55](#_Toc72206012)

[S.M.A.R.T. Objectives 56](#_Toc72206013)

[Beneficiaries 57](#_Toc72206014)

[Risks 58](#_Toc72206015)

[Work Plan 59](#_Toc72206016)

[Research Ethics Review Form: BSc, MSc and MA Projects 60](#_Toc72206017)

[PART A: Ethics Checklist 60](#_Toc72206018)

[PART B: Ethics Proportionate Review Form 63](#_Toc72206019)

[Changes 66](#_Toc72206020)

[Templates for Consent and Information 66](#_Toc72206021)

[Further Information 67](#_Toc72206022)

[Appendix B – Requirements Documentation 68](#_Toc72206023)

[Use case prioritization 79](#_Toc72206024)

[Appendix C – Design Documentation 80](#_Toc72206025)

[C.1 Use Case Diagram 80](#_Toc72206026)

[C.2 ERD (Entity Relationship Diagram) 81](#_Toc72206027)

[C.3 Wireframes 82](#_Toc72206028)

[C.3.1 Home Page 82](#_Toc72206029)

[C.3.2 Login Page 83](#_Toc72206030)

[C.3.3 Register Page 84](#_Toc72206031)

[C.3.4 Contact Us Page 85](#_Toc72206032)

[C.3.5 Profile Page 86](#_Toc72206033)

[C.3.6 Help Requests Page 87](#_Toc72206034)

[C3.7 Logout Confirmation Page 88](#_Toc72206035)

[C.3.8 About Page 89](#_Toc72206036)

[Appendix D – Test Results 90](#_Toc72206037)

[Appendix E – Implementation Document 97](#_Toc72206038)

[Base Html file 97](#_Toc72206039)

[Contact Html file 99](#_Toc72206040)

[Post Detail Html file 99](#_Toc72206041)

[Post Form html file 100](#_Toc72206042)

[Requests page html file 100](#_Toc72206043)

[Help Request Model 101](#_Toc72206044)

[Registration Recognition 102](#_Toc72206045)

[Carpark app routes 103](#_Toc72206046)

[Parking Logic file 104](#_Toc72206047)

[Carpark app views 105](#_Toc72206048)

[Django Project Settings 106](#_Toc72206049)

[Django Project Routing 109](#_Toc72206050)

[Login page html file 109](#_Toc72206051)

[Logout page html file 110](#_Toc72206052)

[Profile page html file 111](#_Toc72206053)

[Register page html file 112](#_Toc72206054)

[Forms file 113](#_Toc72206055)

[Models of the parking logic 114](#_Toc72206056)

[Signals file 115](#_Toc72206057)

[Views file 116](#_Toc72206058)

[Appendix F – Running the solution 117](#_Toc72206059)

[Method used by the author 117](#_Toc72206060)

# Acknowledgements

I want to start by thanking everyone in the computer science department for all the guidance provided for the realization of this project as without them a task this big would have been impossible to accomplish.

Thanks to my friend Usman who helped me relieve stress by going out to exercise with me almost every day for the past 6 months. I would also like to immensely thank my friends who I consider by brothers in Italy for supporting me morally and emotionally by keeping me company during the late-night sessions and listening to my rants: Mirko Roscani, Ismael Ziva, Francesco Zara, Baldo, Samuele Grasso, Hiruka and especially a big thank you goes to Diego “Panda” Neroni for always being there for me.

I thank my parents, Daniela and Vasile, for being there for me and coping with me during the last 4 months, where I have been really stressed but have always comforted me with their warmth and love.

Author:

Claudiu Andrei

# Chapter 1 - Introduction

## 1.1 Project Overview

This project resulted in the creation of a platform for car park management by including a number plate recognition system and a web portal for users. Below is an overview of the project for the read discussing the underlying problem meant to be solved with this solution and the objectives the author has identified for reaching the desired result.

## 1.2 Problem to be solved

The society we currently live in uses travel means more than ever before, therefore an always increasing number of personal vehicles is present in the UK each year. In a world where the ability to move quickly is critical, no one wants to waste time in a society where most people's free time is restricted. As a result, almost every time we leave our homes, we choose to travel to our destination by car. For obvious reasons, we just travel by car when we need to go anywhere where there is car parking. Since we live in a country like the United Kingdom, where anybody who wants to own a car can, it is normal for the number of cars on the road to increase year after year. Furthermore, as older vehicles depreciate year after year, an increasing number of people are able to afford and maintain them, resulting in a much higher demand for car parking space. Because of this, the car park sector is one of the most prolific business types these days, with car parks being built everywhere for people to park their car when they reach their destination or store them when they are not using them. However, in the industry the methods used to operate are often outdated even if recently new ways of operating car parks have appeared. These however don’t result in an easier process for customers in most cases, and are still very frustrating and fidgety to operate, for this reason the author of this project has decided to tackle this issue and create a system that would make this process easier for customers while also giving more freedom and peace of mind to the owners.

## 1.3 Project Objectives

The author aimed at creating a system that allows car parks to receive payments with minimal effort while also allowing customers have a more enjoyable experience. As a result, everything should be implemented smoothly to function in synchrony with the various parts of the system; this means that there should be a number of systems interacting and exchanging information. Building software that receives input as an image and can extrapolate information from the it, such as a number plate or the shape of a car, was how the author visualized the implementation of this. Once the program detects the existence of a number plate, it will deduce the characters and send this information to different system.

This will save a record of the number plate as well as the moment it was first recognized, enabling it to be used later on for further processing. There should also be a database with a list of registered customers, where the registration of a vehicle that has just entered the car park can be reviewed against the records, with no need to create a new instance if the vehicle is already registered.

When the number plate recognition device detects that they have entered the parking lot, the amount due is calculated and displayed to the customers to complete and review the session payment. If a customer has not registered, an error message stating that it was not possible to connect the number plate to a user would be displayed. This approach improves upon current methods by allowing cars to enter and leave without halting or asking the customer to take any extra steps, making it much more enjoyable in the long run while also allowing business managers to spend less money on payment booths and employees.

## 1.4 S.M.A.R.T. Objectives

* Number plate recognition system: This objective aims at implementing a solution that can look at an image and recognize that a number plate is present in it, then return the characters to database. As at the beginning of this project the author did not have the knowledge to implement this, they will have to used the material learned in the Introduction to A.I module as well as some online tutorials. The test for this consists of several images containing car number plates, and if they are recognised correctly under certain circumstances the test will be considered a pass.
* Component to work out how much each customer gets charged for the amount of time they have used the car park for. In order to achieve this, the number plate recognition system will submit the registration number of the vehicle to this program which will then start to work out the amount to be paid when the customer leaves the car park. Research on how to implement this and the programming language to use will be described in the following chapters. In order to test for this, a set amount of time will be decided and if the number returned at the end matches the amount that should be paid the test will be considered successful.
* Customer registration system for customers to register their vehicle and other personal information, if they change their vehicle in the future, they will be allowed to edit their car registration number in the system so that their new vehicle will be recognised when parked. To achieve this, a web portal accessible to customers will be implemented. In order to test for this, dummy data to create accounts will be used and tested to check if the process of registration works as intended.
* Admin panel: this should be present in order to allow the administrator of the system to manage the data present in the system as well as the customers information. Furthermore, it will be used to determine if the customer has any unpaid sessions by viewing their profile as well as for editing information and dealing with customer contact queries if they are experiencing issues or need to edit information.

# Chapter 2 - Output

## 2.1 WebPortal Application

|  |  |
| --- | --- |
| Description | Successfully implementing the Web Portal with the user related features such as registering, logging in and editing profile as well as the parking logic which will allow the registration number recognition to function in synchrony with it. |
| Usage | Customers, Admin, Non-registered Users |
| Beneficiaries | Car Park owner, Customers, Reader |
| Link | Moodle Submission |

## 2.2 Number Plate Recognition and Source Code

|  |  |
| --- | --- |
| Description | The output of the source code was comprised of 1400 lines of code written by the author, this includes python files, html and css and Django Syntax and does not include any libraries used. 48 files of code were generated in 11 directories for the final solution, without counting prototypes and tests. |
| Usage | Python code used to develop and implement all the features and functionalities of the system. Manages database and logic of the system as well as the looks of the web portal. |
| Beneficiaries | Car Park owner, Customers, Reader |
| Link | Moodle Submission, Appendix E |

## 2.3 Database

|  |  |
| --- | --- |
| Description | Configuration of the Django model files that create the database structure |
| Usage | Creates the structure for the data of customers and parking sessions to be handled by the system. |
| Beneficiaries | Car Park owner, Customers, Reader |
| Link | Moodle Submission, Appendix C.2, Appendix E |

**2.4 Project Definition Document**

|  |  |
| --- | --- |
| Description | Document describing the objectives and requirements of the project |
| Usage | Used to receive approval of project and compare final results with the original requirements. |
| Beneficiaries | Reader, Author |
| Link | Moodle Submission, Appendix A |

**2.5 Testing Documentation**

|  |  |
| --- | --- |
| Description | Provides proof and documentation of the tests carried out |
| Usage | Used to document the outcome of the testing phase |
| Beneficiaries | Reader, Author |
| Link | Moodle Submission, Appendix D |

# Chapter 3 - Literature Review

## 3.1 Overview

In order to produce a solution that satisfies the requirements and follows the best practices the author had to embark into further research to gain the required knowledge needed to understand design patterns and ways of developing this solution in the most efficient way possible. Below will be listed noteworthy sources used.

## 3.2 Digital Accessibility

Accessibility refers to the extent to which a product, device, service, or environment is available and navigable for persons with disabilities, or for persons with other special needs or functional limitations. Digital accessibility, the focus of the present round table, centers on access to technology products, resources, and services across hardware and software (Carnegie Melon University 2015, Lazar et al., 2015). This solution will contain parts that the user will have to interact with, such as the registration portal, where they will be able to register their vehicle number plate and add a payment method that they will be automatically charged from. The author had to make sure that the portal meets the digital accessibility guidelines set by the World Wide Web Consortium (www.w3.org). Digital/Web accessibility implies the removal of barriers when it comes to accessing digital resources, put alternately, it implies perception, comprehension, navigation of, and contributions to the Web easier for people who, for different reasons, might find it harder to complete such actions.

## 3.3 Edge Detection

Edge detection is an image processing technique used to find boundaries of objects within images. It works by detecting discontinuities in brightness. It is used in the field of computer science for image segmentation and data extraction like applications such as image processing, computer vision and machine vision. “Contains adaptations from: International Journal of Computer Science & Information Technology (IJCSIT) Vol 3, No 6, Dec 2011” Image segmentation is what allows for edge detection: it is an essential step in image analysis as segmentation separates the image into its component parts and objects. The extent to which the segmentation is carried out depends on the problem being solved. When objects that are being processed by an application become inaccessible, segmentation must stop. Algorithms used for segmentation of images are based on the discontinuity and ad similarity of image intensity values, therefore different approaches may be used by the segmentation algorithms. Discontinuity approaches aim to partition an image based on abrupt changes in intensity and similarity by partitioning an image into similar regions according to a set of predefined criteria. Furthermore, because edge detection is a part of image segmentation, the effectiveness of many image processing and computer vision tasks are dependent on the accuracy of detecting meaningful edges. Image Segmentation is the process of partitioning a digital image into multiple regions or sets of pixels. Essentially, in image partitions are different objects which have the same texture or color. The image segmentation results are a set of regions that cover the entire image together and a set of contours extracted from the image. All of the pixels in a region are similar with respect to some characteristics such as color, intensity, or texture. Adjacent regions are considerably different with respect to the same individuality. The different approaches are (i) by finding boundaries between regions based on discontinuities in intensity levels, (ii) thresholds based on the distribution of pixel properties, such as intensity values, and (iii) based on finding the regions directly. Thus the choice of image segmentation technique is depends on the problem being considered.

## 3.4 OpenCV (Python)

OpenCV stands for Open Source Computer Vision Library and is a computer vision and machine learning software library for python. It is used vastly in areas dealing with image processing as it offers a variety of methods that allow to alter the image and make it much easier to be used for the intended purpose.

## 3.5 Existing Tools and Programs

After choosing to resolve this problem and coming up with a solution the author made some research to identify any potential competitors. There are not many that directly deal with the issue in the same way intend by this project, however there are number of software and apps that in some way resemble idea.

### 3.5.1 Neural Labs VPAR

This is a number plate recognition system developed by the company Neural Labs. They offer the service as a simple software that recognizes car number plates which can have many applications; however, they do not offer any further type of assistance when it comes to implementing the software into a car park business. This project tries to tackle this issue by embedding this type of system with another one that will allow for automatic payment collection and calculation of the sum to be collected from the customer. On the VPAR website it isn’t directly specified whether or not the system can recognize UK number plates, as the program demonstration video they provide only displays the system being used with cars which are fitted USA registration numbers; this program, on the other hand, will be designed to work specifically with UK cars.

### 3.5.2 OPENALPR by Rekor

This is a service offered by an organization called Rekor. The way their application differs from the Neural Labs one is the way in which the service is provided, in fact, they have different plans on their website, which charge customers a different amount of money based on the number of number plates that can be identified each month. An important feature missing in VAPR is present here, that is the recognition of UK plates, however they mainly specialize in the recognition of USA vehicle plates, which might imply that UK vehicles might have a higher chance of error when being scanned. This service offers a plan very similar to what this proposed solution describes: Rekor allows customers to also register their vehicle and pay within the app beforehand. The one feature that isn’t mentioned on their website is the computation of the time a customer uses the car park for, therefore working out the amount that the customer has to be charged isn’t possible with OPENALPR.

# Chapter 4 - Method

## 4.1 Requirements Specification

As this project has not been devised for a specific client it is to be researched and looked into exclusively with the sole initiative of the author, as they will be the only one working on this solution. Its development and discretion as to what is required, from its functionalities and features to its implementation and testing will not be judged by external sources. Furthermore, the author was unable to locate many sources within literature review chapter as there are not many documented software that encapsulate the specific requirements that this project intends to represent.

## 4.2 Analysis

As a result of this brainstorming sessions had to be conducted with guidance from a supervisor at times, to decide the design and most efficient way to achieve the looked after result. The author has researched image segmentation, edge detection and image processing in the lit review and has come upon challenges when it comes to implementing this in a project, therefore in the method section they will investigate solving these issues and experiment methods of doing so.

Since the objective of this project is to create a solution which is well documented and feasible with the skillset at the author’s disposal, they have decided to keep the main areas of the program functional but with lack of redundancy by showcasing the effectiveness of the solution without having to implement features which do not aid the functioning of the system. One of the main challenges when working with image processing is the development of object detection: this requires many optimizations to work fluently, therefore, images will be processed before applying any sort of detection to them.

One feature which would not be included is the implementation of a payment portal; this will not need to be present in the final version of the project as this software is not intended for use by any client. In case this needs to be made into a feature in the future, it is easily achievable using several payment processors available online, however these are not free to use and since this program is not intended to be used to generate profit, the author overlooked this aspect. Therefore, the amount that the customer should be charged after their stay can just be showcased on their profile page, accessible after they register using the portal.

Dealing with Digital Accessibility is also something that the author will need to take into account as the web portal will be used by people when registering. It is important that they tackle the issue by offering a solution which is accessible from a variety of devices, such as smartphones, tablets and computers which use different type of browsers and operating systems such as Android, iOS, Windows, Linux and MacOs. The main premise of this will be creating a website that includes responsive design and utilizes a platform that is conventional on most devices.

## 4.3 Software Development Methodology

Time was spent researching different methodologies; however, they would not really be the most suited to developing the intended solution for the problem that is being solved by this thesis. With limited experience when it comes to using various software development methodologies, other than Waterfall, the author chose to use this method as it seems to be the most suitable for the project decided to undertake. While most of the developing methodologies focus mainly on customer satisfaction and make of that the priority, it was chosen not to use one such as agile as the issue of satisfying the client is not encountered, that is because this solution was developed without having a fixed client. Furthermore, by using the waterfall methodology the author was able to spend as much time as they deemed necessary on the design phase and only once they felt confident with the design they focused exclusively on the coding and implementation of the solution, without having to spend more time revisiting what had been already researched and perfect.

The way in which the development of this solution was planned was spending a good amount of time on the design phase in order to make sure there are not any inconsistencies in the structure, as this allowed the author to have a clear plan to follow and a “blueprint” to work with. Since this solution involves 2 main pieces of work, it had to be decided which part to work on first and as the main aim of project is to develop a system that can recognize car number plates, it was decided to firstly focus on developing the python solution that performs the image processing and extracts characters of the number plate. Once this was be completed, it was possible to focus on the web portal that customers use in order to register their details and car number plate.

## 4.4 Tools

In order to help myself visualize and better understand how to implement this solution the author had to spend an important amount of time analyzing requirements and developing Unified Modeling Language (UML) diagrams which help to develop a good starting point for how details and functionalities of the software are developed and the way they interact with each other. The tool chosen to create some of the diagrams such as: Use Case Diagrams, Design Class Diagrams, Sequence Diagrams, and Activity diagram was Visual Paradigm.

## 4.5 Languages and framework

The author concluded that the best way to implement the number plate recognition system is by doing so using python; from the results of their research, it was decided that python offered the best tools and frameworks for this type of application. There are many tools tailored specifically for image processing such as OpenCV, which was researched and talked about in the lit review chapter, it is an open-source python library used widely for image processing tasks. After carrying out further research, it was found that OpenCV allows image manipulation which made carrying out the task of number plate recognition much easier. As in regards for the web portal there were many options available to achieve the desired result, however since python is used for the first part of the software, it was decided that a framework such as Django would be more suited for this as it revolves around using Python to create web applications. The Django is the framework of choice when developing web applications and is used by many popular websites such as: Mozzilla, Youtube and Spotify. It offers many functionalities which allow to save time and resources. This was used for the main outline of the website, also known as the front end, where it was integrated with HTML and CSS (cascading style sheet) language for the styling and appearance of the website; it also allows to implement ways of dealing with the logic of the website.

## 4.6 Design

Once the analysis phase has been completed, the requirements specifications and methodologies outlined, the internal components as well as the visual appearance and interaction with the program need to be established.

### 4.6.1 Use case specifications

Use cases are used to outline different outcomes for different behaviors and scenarios that are implemented within the software. These help out by giving some points and functionalities that are to be covered and taken into account. These will allow for consistent and accuracy during the development using the waterfall orientated methodology.

### 4.6.2 Use Case diagram

The author developed a use case diagram for the program to allow for better planification and a representation of how the different parts of the system interact with each other and how users (actors) will navigate and interact with the various components of the software as well as their response to the inputs given.

### 4.6.3 Entity relationship diagram

An entity relationship diagram was created by the author, this is what helps visualize and describe the structure of the database. The ERD provides a blueprint or design of the database which is to be developed and therefore makes the implementation much easier. For this project the author had to develop these so that during the implementation phase it was clear how the various parts of the database interacted with each other as well as with the methods developed which manipulate this data.

### 4.6.4 Wireframes

A website wireframe, also called a page schematic or screen blueprint, is a visual representation of a website's skeletal structure. Wireframes were made with the intention of organizing elements in the most efficient way possible to achieve a specific goal. A business goal and an innovative idea are typically used to inform the intent. The wireframe illustrates the website's content page structure, including design elements and navigational systems, as well as how they interact. Since the main emphasis is on functionality, actions, and content priority, the wireframe typically lacks typographic design, color, or graphics. Wireframes may be sketches made with pencil or whiteboards, or designs created with the aid of a variety of free software.

## 4.7 Web Portal

The main interaction with the system is to be done through the web portal page that customers use to register their vehicle on as well as to manage their account, personal information and preferences. In order to give off a good impression and clear vision of what the available actions are, it is important that the user is able to easily navigate through the various sections of the website. Designs and prototypes for the look of pages are essential to plan how the displayed data is laid out prior to the start of development which help when building the various sections of the page. A very convenient way of developing we pages is by using CSS with the grid layout method, which involves splitting the page into different sections and using each section to display a certain function or information. The reasons for using this style of development are various, but mainly it is because it allows standardization of code across browsers. The visual appeal of the page is also important; Bootstrap allows for easy web customization, and it contains many designs already widely used on almost every website on the internet today, allowing to save time on customizing the CSS of the site.

### 4.7.1 Login page

The login page is to allow customers to input their details in order to access their account, view their details and edit their preferences. Blank fields are completed by the user with the details used when they created and registered their account. The fields which are requested in order to allow for login are the customer email address and their password, the details are checked against the database data and if they are correct the customer will be logged into their account. These checks happen after the LOGIN button has been pressed, which then triggers the authentication process and if this is successful the customer will be able to access their account information.

### 4.7.2 Register Page

For the system to function in the intended way it is necessary that cars that enter the car park are assigned to a customer and in order for that to happen it is essential that customers are able to get their details logged into the system. These are to be taken as input within the web portal, using a form that users can fill in with the relevant information, such as:

4.7.2.1 Password: This is selected by the user during the registration phase and is what allows them to access the functionalities of the website after logging in. It is important that customers select a password that is not too similar to their username as this is not allowed and returns an error.

4.7.2.2 Username: This is usually needed in database, as it is used in the system to link their accounts to the correct information. The username is also what allows the authentication to function correctly as it is required together with the password when logging into the website.

4.7.2.3 Registration Number: this is required to allow the process of number plate recognition to associate the output to a certain user in order to process their data and deal with the logic of the parking sessions.

4.7.2.4 Email address: When customer create their account their email address is used to send emails to them and it is also the field which is used to allow the customer to login into their account.

## 4.8 Python Number Plate recognition system

The number plate recognition system is the main backend of the system, as it is what allows for the entire application to function according to the specification. Various methods for implementing the character recognition of the number plate, the image processing process, masking of the file as well as the object detection methods were experimented by the author.

### 4.8.1 Image processing

In order to start the recognition process an image containing a number plate is required, and it therefore needs to be loaded in the program. This will happen using a library in python called OpenCV, which was talked about in the literature review; which amongst allowing to import the image into the program will allow to further work with it in order to allow for the next part of the program to execute.

### 4.8.2 Object Detection

In order to be able to start the process of number plate recognition it is necessary to identify the position in the image of the number plate of a car, this can be achieved in various ways, however the 2 experimented with are OpenCv and YoloV4. These work by allowing to define a certain pattern to be recognized in an image, which by doing that, allow to cut out the portion of the image containing the number plate which can then be used to extract the characters of the registration.

### 4.8.3 Character recognition

This is what ultimately allows the characters of the registration of the vehicle to be extracted from the image and converted into a string, which can then be used to work with database entries and allow to manipulate data in order to calculate the time the customer has been using the car park and how much they are due to pay. A way for data to be fetched from images was implemented separately and tested, after which it could be implemented in the system without having to further deal with bugs and errors.

### 4.8.4 Implementation in the web portal

In order to allow the python side of the system to communicate with the database and therefore the web portal, it was necessary to implement means of communication between the two. This was achieved by having methods that acted as a bridge between the number plate recognition system and the web portal. These deal with the logic of the parking sessions by using the output received from the number plate recognition system and associate it with a user in the database, after which the necessary actions can be performed to manage the logic of the parking.

## 4.9 Pillow

Pillow, also known as PIL, is an imaging and manipulation library that provides additional functionalities to the Python interpreter. The library provides extensive file format support, efficient internal representation and powerful image processing capabilities. Its use is implemented to deal with the profile picture that customers are able to upload on their profile page.

# Chapter 5 - Results

## 5.1 Methodology

As mentioned already the application is mainly composed of 2 components, which include the number plate recognition as well as the online portal for users to register in. The development methodology chosen for this project was waterfall, therefore it was essential that a good plan and design was thought of before starting development. This is also shown in the PDD, where it can be seen that the design and planification took a lot longer than the actual development. However, this has paid off as the coding and implementation of the solution went as expected and eventual problems encountered didn’t take excessive time and effort to fix.

## 5.2 Analysis

The research carried out during the literature review has been useful as it has allowed to identify which techniques and technologies to implement in order to achieve the desired result. Most important has been the research carried out regarding the python object detection and character recognition from images.

## 5.3 Tools

In order to ensure consistency within the design and development phase, UML diagrams created with Visual Paradigm were used. These graphs helped by giving an overview on the entirety of the project and allowed a smoother development process and implementation.

### 5.3.1 Programming languages and ide

#### 5.3.1.1 Tools used for the Python number plate recognition system

A variety of technologies, libraries and frameworks were used in order to develop the system. These will be described in further detail in different paragraphs of the results chapter, however an overview of programming languages used is necessary. A particularly useful piece of software is Anaconda, which was also introduced to students during the introduction to A.I. module; Anaconda is a distribution of the Python programming language with tools aimed at aiding scientific computing and working with A.I. for those reasons the Spyder IDE provided with Anaconda was the most suited to be used for this project. When developing the python system, the python Notebook provided within Anaconda was also used to create prototypes and experiment with various ways of executing the object detection and character recognition.

#### 5.3.1.2 Tools Used for the Web Portal

For development of the web portal, it was decided to use Django, an amazing framework for building web applications using Python. Django was chosen as the web framework because of all the features and functionalities it offers as well as its flexibility; working with databases is really nicely implemented and optimized and the model-template-views architectural pattern is something that was found to be extremely practical. For the rest of the website, HTML, CSS such custom CSS classes as well as Bootstrap were used. It was decided to use Bootstrap since it includes nicely designed elements which are already widely used in many websites and trusted by many web developers.

### 5.3.2 Model-template-views architectural pattern

As mentioned previously Django was used in order to develop the web portal for the solution. Django utilizes the MVT, short for Model View Template, which is a software design pattern composed of three main elements: Model, View and Template.

The model is what helps handle the database, particularly it is the data access layer which deals with the data; the model is a class which contains essential fields and methods, in Django each model class maps a single table in the database. In order to create, delete, retrieve and edit records from the database field Django provides a database-abstraction API.

Views are documents where the business logic of the application is located. These are python functions that perform business logic and return a response to the user, such as HTML contents of a Web Page, redirects, or errors. For example, they may be used to display different content to users who are logged in and display an error to user who are not.

The template is a layer that deals with presentation, specifically it handles the User Interface part completely, this is what helps generate dynamic HTML pages. Templates are HTML documents composed of static Markup Language parts outputting the desired content as well as Django syntax that can display content dynamically. Templates are useful because they can be reused within other templates in order to make writing code more optimized and efficient.

### 5.3.3 Bootstrap

When developing the web application design choices had to be made in order to identify the optimal method to be used for the looks of the website. Bootstrap offers a web framework widely used for developing responsive user interfaces. It is constituted of a set of CSS and JavaScript files that the author used for designing templates of the different elements composing the front-end of the website, such as tables, navigation bars and footers. Bootstrap offers specific styles by defining several classes that can be applied to various elements of the application.

### 5.3.4 Setting up the project and working environment

#### 5.3.4.1 Number Plate Recognition System

In order to get started with the development and implementation of this project it was needed to setup the environment to work in. Following, are the python libraries that had to be installed for the development of the project

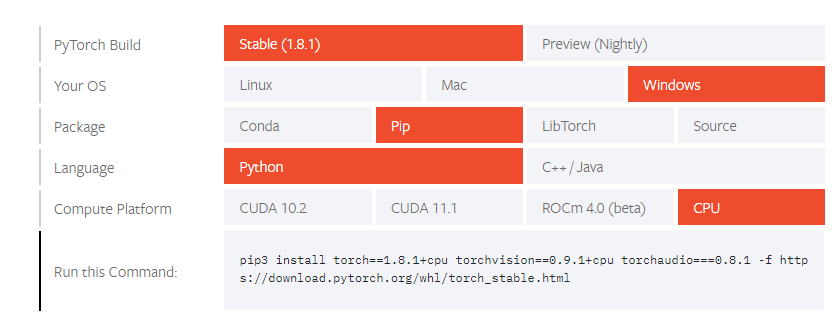


Figure 5.3.4.1.1

First, Pytorch was installed (Figure 5.3.4.1.1) to facilitate the use of EasyOcr, which was installed with the code in figure 5.3.4.1.2.

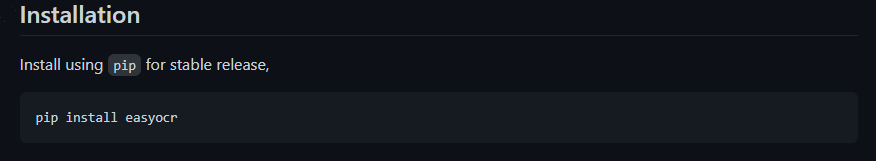


Figure 5.3.4.1.2

Another requirement for the plate recognition was imutils which was installed with the code in figure 5.3.4.1.3.



Figure 5.3.4.1.3

Another requirement was OpenCV which allows to work with images, the installation process for OpenCV requires a few steps. Firstly, numpy needs to be installed as well at matplotlib. After these requirements are satisfied in order to install the latest version of OpenCV it is needed to download the package and clone it in the python directory. This should be the last step and OpenCV should be available to import within projects.

After all the required libraries are installed, the IDEs can be setup. Firstly, a new .ipynb file was created in order to be utilized within the Jupyter notebook, which was required to create prototypes of the number plate recognition system. The Jupyter notebook allows for easier debugging as it is able to run snippets of code bit by bit making it easier to follow the output throughout and facilitating debugging. The Spyder IDE was also used when finalizing the development of the python system, therefore a new python file was created where the final version of the document would written and then used to import in Django.

#### 5.3.4.2 Web Portal

Firstly Django needed to be installed and a project had to be setup by running the command “startproject ProjectName”. This creates a folder within the directory where the command was ran with all the files the Django application needs in order to function properly. The project folder that Django creates looks by default like in figure 5.3.4.2.1:

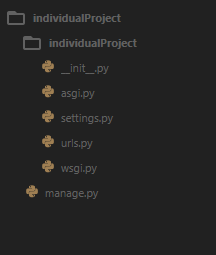
The ” \_\_init\_\_.py” is an empty file used in order to allow python do identify the project as a python package.    
The next file generated by Django is the ”asgi.py”, this together with the ”wsgi.py” file are used for the deployment of the application, as they allow the python web application and the web server hosting the app to communicate. They are both generated by Django in order to give more freedom to the developer as ASGI is becoming more widely used than WSGI. The ”settings.py” file contains a number of options that allow to configure the application by adding, removing and editing settings of the environment. Django also provides a file that makes the navigation of users within the app much easier to set up; this is done through the ”urls.py” file, which allows to set the mapping to certain URLs which will redirect the user to the correct page and display the information intended for the user to access when following a certain path within the web app. ”manage.py” is the file containing Django's command-line utility for administrative tasks, this is left untouched in this project as it does not need the implementation of custom admin commands.

Figure 5.3.4.2.1

The barebone structure of the project is generated with the command “startproject ProjectName”, however in order to be able to create an application within the project, Django utilizes the command “python manage.py startapp AppName”. This is used multiple times as this django project contains a number of applications.

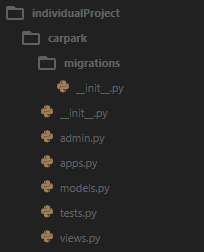
When creating a new app Django, the files necessary to develop a completely functional app are created as seen in figure 5.3.4.2.2. The 2 \_\_init.py\_\_ serve the only purpose of allowing python to detect the package as a python package. The ”admin.py” file allows to edit the admin control panel view for the application. ”apps.py” contains the app configurations and allows to edit various behaviors of the application. The use of ”models.py” and ”views.py” have been described previously when introducing the MVT architecture. Lastly ”tests.py” is used to run tests in order to verify that the various features of the application function as intended by using unit tests (tests that verify the integrity of a specific function) and Integration tests (tests that focus on user behavior and testing entire applications).

Figure 5.3.4.2.2

The IDE used for the development of the Web App is Sublime Text 3, a widely used software that makes writing code more enjoyable than using a normal text editor. Sublime Text 3 does not exactly fit in the category of IDEs as it does not provide debugging, therefore, it is mostly considered a syntax checker and code editor, however by enabling Debug Mode within the Django project settings, the use of a debugger within the Code Editor is superfluous. Additionally, Sublime Text 3 offers a powerful Python API that allows plugins to be installed that augment built-in functionality and a package eco system that offers useful addons which make writing code more optimized.

### 5.3.5 Django Syntax

Django allows to display dynamic elements on web pages by using its own custom syntax. This is particularly useful when displaying information for a particular user on their profile page since it can fetch data from database, work with variables, add filters and configurations. In this project the author has made use of the Django syntax throughout the entire solution, and it is easily recognizable because of the code wrapped around double curly brackets: {{‘code’}}. Code wrapped around one curly bracket followed by a % sign “{% ‘code’ %} are tags that provide arbitrary logic during the rendering process.

### 5.3.6 Wireframes

After requirements were identified and wireframes were generated; these give additional help when developing the solution as they help visualize the various screen/page the user is presented with. These were created by the author in blank sheets that contain the various contents and interactable components, such as buttons and forms, that the user is able to see when landing on a specific page under certain circumstances, such as displaying different contents for user who are logged in and users who are not. These are remodeled, adapted and the layout of elements is recreated using markup language and Django. The final version of if the application wireframes can be found in the Appendix C.3.

### 5.3.7 Use Case Diagram

The use case diagram together with the rest of the UML documentation including wireframes and ERD can be found at Appendix C.

## 5.4 Implementation

Succeeding completion of the analysis and design of the application is the implementation phase, where the various tools and languages described in earlier parts of this documents are used to build the application. The various layers such as the front end, back end and database are built utilizing the waterfall methodology therefore the first component to be developed was the number plate recognition system which was subsequently implemented in the web portal that users have access to. The structure of the application as well as the list of classes can be found in the Appendix E.

### 5.4.1 Python Recognition System Prototype

The component that allows the system to function in its intended way is the number plate recognition. In order for this to function as intended it is necessary to import the dependencies that are used (Figure 5.4.1.1).

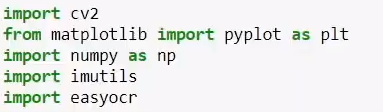


Figure 5.4.1.1

The author has experienced with various ways of carrying out the process of recognition of the number plate, the process however remains similar and only minor tweaks are carried out to the values used when calling functions, and it consists of the following steps:

1. Reading an image from a directory within the python function
2. Greyscale the image and apply blur
3. Using several filters on the image
4. Identify the edges of the image
5. Identifying the contours of the number plate the reading of the text in ocr
6. Applying a mask
7. Isolate the segment containing the number plate
8. Read the text off the image using easy OCR
9. Processing the result returned

OpenCV is used to read the image into python firstly, after which the image is converted into greyscale, thus removing color and making it easier to work with.

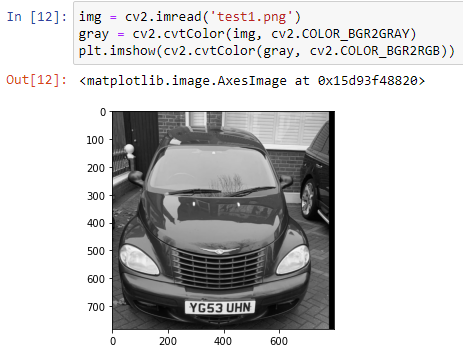
The output obtained as a result of importing the image in Python and converting it to greyscale (Figure 5.4.1.2): OpenCV offers a way of converting images from one color space to another with a function that can be used by calling cv2.cvtColor() and providing the source variable containing the image that the conversion needs to be applied to. The second field of the function is the color space that the image is being converted into and these are provided in the OpenCv documentation.

Figure 5.4.1.2

Another library that was used during the development of the prototype was Matplotlib, which is what allows to display the image within the Jupyter Notebook and facilitates understanding the flow of the execution of the code. Figure 5.4.1.3 shows the image used as test when experimenting with the various ways of implementing this system.

Figure 5.4.1.3

The color filtered image needs to undergo additional processing as mentioned above, therefore a bilateral filter is applied to the image which smoothens the edges and reduces noise in the image. This is a necessary step as it would not be possible to execute the following steps if the image contains too much noise or unwanted sharp edges. The author has achieved noise level reductions by making use of a function offered by the OpenCV library, more precisely the cv2.bilateralFilter() function which takes as arguments the image on which the filter is wished to be applied onto as well as values which help calibrate the strength and effectiveness of the filter. The filtered image is then processed further by applying a canny filter which outlines the edges of the image by making use of edge detection, a technology discussed more in detail by the author in the literature review chapter of this document. The Canny edge detection filter is also provided by the OpenCV library and can be applied to images by making use of the following function: cv2.Canny() where its arguments are a source image, the high threshold value of intensity gradient and the Low threshold value of intensity gradient. This step is necessary for enabling the object detection of the number plate in the image, even if there are several object detection libraries available which already provide ways of identifying number plates, the author has decided to experiment with implementing number plate detection using different libraries. Figure 5.4.1.4 below shows the result after applying the filters mentioned above.

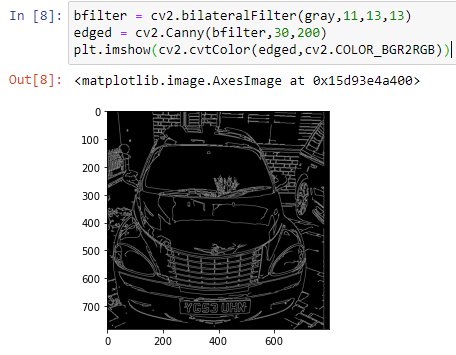


Figure 5.4.1.4

It now becomes easier to detect where the number plate of the vehicle is located within the image as all the straight edges are easily identifiable. The manner in which the author has decided to perform this is by using contour detection, which is allows to identify where an object is within an image by identifying its borders, therefore recognizing polygons and shapes obtained in the previous steps. What the program needs to be looking for, particularly, are 4 points which form a shape resembling a rectangle; these in most cases represent where the number plate is located.

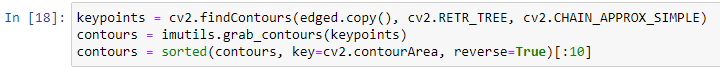


Figure 5.4.1.5

The process of locating the number plate in the image (Figure 5.4.1.5) begins with finding the contours using the cv2.findContours() function, which goes through the edged image and finds shapes within it. The argument of the function “cv2.RETR\_TREE” is utilized to return the contours in a tree format, allowing the author to the traverse the tree and find different levels of contours within it. The “cv2.CHAIN\_APPROX\_SIMPLE” argument is necessary as it instructs the function to return the simplified version of the contour, as without this instead of returning the main points constructing the polygon, the function would return every single point composing the shape, which would unnecessarily use extra resources and computation power. Using imutils, the author has passed the key points of the contours to a variable in order to be able to sort them using the Python function sorted(). The top 10 results are sorted by a key, represented by the contour area, whilst setting reverse to TRUE makes the function run in descending order. After identifying the best contours, It is necessary to loop through each one once they are sorted, in order to find the ones that effectively represent a square/rectangle or a number plate.

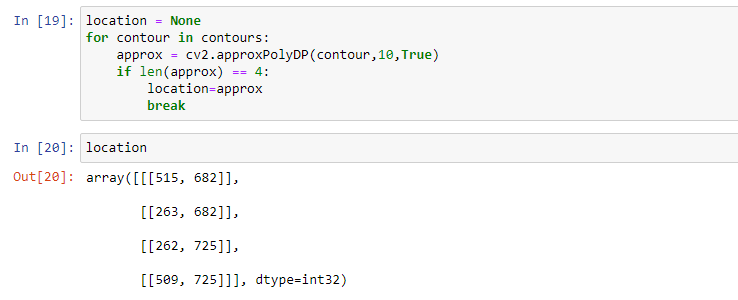


Figure 5.4.1.6

The output obtained from the process is an array containing the coordinates defining the position of the number plate in the image in Figure 5.4.1.6. During the process of looping through the contours the polygons are approximated using the cv2.approxPolyDP() which is passed the contour at a point in time and the argument “10” specifies how accurate the approximation is, therefore allowing for polygons containing imperfections or dents to still be detected correctly and if the approximated contour contains 4 key points the coordinates are returned. The coordinates returned represent the pixels in the image where the number plate is located, in fact if plotted onto the image it can be seen that they contour the number plate of the vehicle correctly. Once the coordinates of the number plate are identified the next step can be commenced: the masking can be applied to isolate the section of the image defined by the coordinates.



Figure 5.4.1.7

The key steps to applying the masking to the image are shown in figure 5.4.1.7, where the NumPy library is used, more precisely the np.zeros() function. To this function the arguments passed are the shape of original image by using “gray.shape” and a parameter that helps specify how to fill in the image, which in this case was specified to be filled with blank zeros by using “np.uint8”. After the image is filled in, the contours need to be drawn within it for the mask to be applied; this was achieved by making use of the cv2.drawContours() function, what is passed to the function here are the temporary image, which is the mask in this case, the contour that is being drawn “[location]” and the way in which the mask is intended to be drawn and displayed is specified. The mask however is not overlayed on the original image at this stage, and this is necessary for the characters contained in the number plate to be identified. A function is provided by OpenCv that the author has used in order overlay the mask onto the original image: cv2.bitwise\_and(), which allows to return the segment of the image that represents the number plate, however, this is processed further since the number plate needs to be isolated in order to facilitate the character recognition which is performed in the following steps.

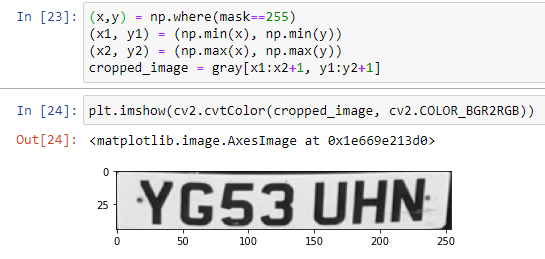
In order to achieve this the author has first defined where the image is not black, by identifying the coordinates of all the pixels which are present in the section representing the number plate. The minimum x and y values are then found as well as the maximum values, which are used to crop the image to the desired size (Figure 5.4.1.8).

Figure 5.4.1.8

When the number plate has been cropped and recognized within the image, the characters present in it need to be fetched. This is done by using the library called EasyOCR and the text is extracted using the easyocr.Reader() method, which defines the language and alphabet format that the results is expected to be extracted in. The readtext method is the applied to the cropped image and the output obtained is a list of values representing coordinates of the text, the characters of the number plate and the confidence score. For the image used as a test during the development phase, EasyOCR has identified a score of 0.944, meaning that the output is 94.4% accurate. (Figure 5.4.1.9)

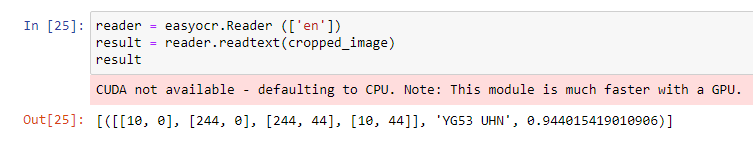


Figure 5.4.1.9

The number plate recognition phase is complete after the steps mentioned above, however, the current output does not allow for it to be used with a database and manipulated further in the intended manner; this is because the only required part of the output is the string containing the text extracted from the image, therefore one last step is required for the output to be returned as desired.

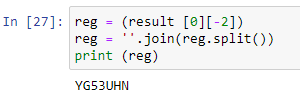
As the output will be checked against entries in the database it is important that these would match correctly, hence the author has only extracted the text string from list obtained as output from EasyOCR (the second to last element) and removed any blank spaces using the code showed in Figure 5.4.1.10. The reason for that is that entries of number plates in the database do not contain any spaces, and the output given by OCR would not allow it to match a user’s plate. Once the number plate recognition system has been fully developed, the author began development of the web portal and the in order to implement the two systems together.

Figure 5.4.1.10

### 5.4.2 Web Portal

Figure 5.4.2.2

Figure 5.4.2.1

As the work environment for the web portal was set it was necessary to set additional configurations within Django. Firstly, as mentioned previously, a Django project can contain several applications, therefore, within the main project folder, two additional apps were created, “users” and “carpark”; the “user” application is responsible for dealing with user logic, such as user profile, parking sessions, details and additional actions that users can perform. The “carpark” application is responsible for managing the main views of the website, where the appearance of web pages is specified, as well as logic dealing with actions the carpark system is expected to perform, such as recognizing the number plate of parked vehicles. These applications were created using the “startapp” admin command (Figure 5.4.2.1) and the result is a folder containing the necessary files as showed earlier with the “carpark” app (Figure 5.4.2.2). Within Anaconda the terminal is launched and after navigating to the Django project directory and running the command above a new application is generated. The final barebone structure for the project as it appears in the Sublime Text 3 editor is seen on the figure on the left. The subdirectories within the newly generated app folders ”migrations” are meant to contain migrations; which are needed to be performed after specifying the structure of the database and its contents in the ”models.py” files. To update the database with any changes made the author has first created the migrations that need to be applied or ”migrated” onto the database. Amongst the admin commands offered with Django the ones used by the author to obtain the desired results were ”makemigrations” and "migrate”, the ”makemigrations” command generates the code file within the migrations folder, which when run results in the changes specified appearing in the database. The ”migrate” command is used to obtain the specified changes in the database, this converts the code in the migrations file to SQL and runs it in order to add or edit tables to the database. The author has also made use of another command (Figure 5.4.2.3), which aids debugging in case the obtained results are not satisfactory: ”sqlmigrate appname migration”



Figure 5.4.2.3

This outputs the SQL code that would run when executing the migration generated by the “makemigrations” command.

With the structure of the project set, the next desired result is obtaining the database that would allow the system to function correctly. After careful analysis of the requirements, the author has defined models within Django and applied the migrations.

### 5.4.3 Implementation of the Web Portal

The main pages of the web portal are:

* Home Page
* About Page
* Profile Page
* Register Page
* Login Page

#### 5.4.3.1 Base File

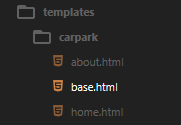
In order to have a page appear on the website Django requires some key steps: firstly, the author has created a base.html (Figure 5.4.3.1.1) file acting as a template for every page in the website, this allows to save time and avoid repetitions in the code, as the common features present on every page can be included at the top of the file each time it is needed. This document is contained in the templates folder within the “carpark” application folder in the Django project. The details contained in the base.html describe the base appearance of the website; this includes a declaration of the website’s metadata and declaration of the CSS as well as Bootstrap. The page title that appears in the browser of the user is also defined here, in order to define the correct title for each page that the base template is applied to, the author has made use of the Django syntax as seen in figure 5.4.3.1.2 below.

Figure 5.4.3.1.1

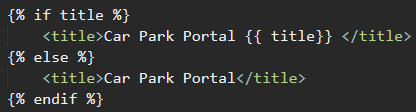


Figure 5.4.3.1.2

An if statement (Figure 5.4.3.1.2) is defined in this occasion by using “{% if title %}“ to check if the page where the base.html document is being used has a title specified, and if it does it will be displayed in the browser preceded by the ”Car Park Portal” text. The {{ title }} used here is defined in the ”views.py” file, which as previously explained, defines the appearance of the content on a page.



Figure 5.4.3.1.3

In figure 5.4.3.1.3 it can be seen how a page’s title is defined in Django, in this case the page title is defined to be ”Home”.

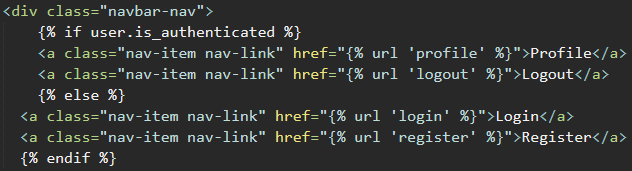


Figure 5.4.3.1.4

The next element that is defined in base.html is the navigation bar, which contains the anchor text that users can utilize to get redirected to the page they wish to visit (figure 5.4.3.1.4). However, its appearance changes based on whether the user is logged in or not. This is because a logged in user should not be able to see the register and login buttons, viceversa, a logged-out user should not be able to view content intended for registered users. In order to achieve this the author has implemented another if statement and by using ”user\_is.authenticated” the base.html template is able to understand which version of the navigation bar to display for the current user’s session. When using the web portal’s features the user may encounter different type of messages, an example of this is a confirmation message that appears when the users successfully logs into their account. In order to allow for these messages to be displayed correctly they are fetched from the views file and displayed accordingly. The final step in setting up the layout for the webpages was specifying where the actual content of the pages is to be displayed, therefore within the base.html file the Django syntax {% block content %} {% endblock %} is used to specify that the main body of the website has to be rendered withing those tags, which are placed beneath the code for the navigation bar and side bar of the website but above the footer section; the author has done this since the content would have to be displayed in the middle of the page. After the base layout for the pages was defined this was imported in the various html documents by using {% extends "carpark/base.html" %} at the top of each document; and in order to define the document as the body of the page the html code was wrapped around {% block content %} tags to be recognized by the base.html file.

#### 5.4.3.2 Applications and Routes

After the Html code was written for the appearance of the different pages, the author had to make them available to display within the browser and accessible to the users, a process that required working with the “urls.py” documents present in the different directories. The way in which Django fetches the information to display is by trying to traverse the urls present in the urls.py document and following the path until it reaches the end where content to render is found. Django will begin traversing the urls beginning at the default project urls.py file.



Figure 5.4.3.2.1

As seen in figure 5.4.3.2.1, the root of the project is localhost, in this case ”127.0.0.1:8000”, therefore anything after a ”/” will try to be matched to a path in the urls.py file.

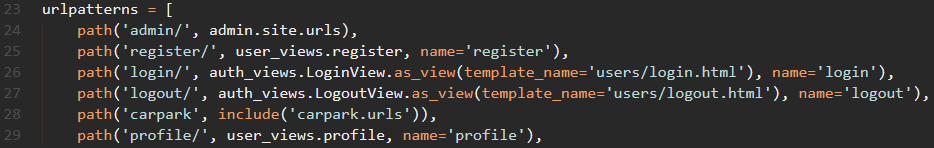


Figure 5.4.3.2.2

The paths present in the main project directory are showcased in figure 5.4.3.2.2 above. To give an example, when trying to navigate to the url “127.0.0.1:8000/login” Django checks the urls.py file for instructions on what to render when encountering “/login” after the main url. In this case the result rendered will be the content of the path named “login”. In case of the “/carpark” string Django will be redirected to fetch the content do display using the urls.py file of the “carpark” app. The reason for which the author has mapped the urls in different apps is because if there was need to change the route to the carpark application for future maintenance it would simply be enough to change where its components are mapped to just once, for example instead of ‘carpark’, the path could be renamed ‘carpark\_debugging’. A full list of the project's routes can be found in Appendix E.

#### 5.4.3.3 Authentication

Users will be able to login the web portal by using the details they provided during the registration process. The registration process is a requirement as the in order for the Django app to create a parking session, a user needs to be assigned to it, therefore once customers have registered their account, they will be able to also login to view and edit their details. The login process is made simpler for the author to implement as Django provides the necessary libraries and functionalities for basic authentication processes, the author has then modified these to suit the use intended for this project.

### 5.4.4 Database Development

As described during the methods chapter, a database is utilized for holding data regarding users and parking information. When developing the database, the author has created a design that would be suited for the requirements described for the project, mainly the challenging part of this implementation was found to be the interaction with the database on the user end as well as the automation of the parking sessions. The database models were split between two main applications, the “users” application and “carpark”. The “users” application was defined by the author to contain models related to the data processed when dealing users information, therefore the users application contains the models: User, which contains the fields given as input by the user during the registration process. Profile, which contains additional information about the users such as fields needed for the parking automation process. A field containing the registration of the car owned by the user is used to associate vehicles to their owners when they are identified by the number plate recognition. Furthermore, it is necessary to keep track of the amount of time customers are parked for, “parktime” is defined as DateTimeField to resolve this issue. “UserSession” is also defined in the models of the users app, this model class resulted from research on how to implement the individual parking sessions of single users. This associates each parking instance to a user and allows customers to manage their sessions from their profile page, such as checking the amount due, time spent in the car park and more.  The carpark app models contain fields that are used for when users create a new contact query for the admin to deal with. Once all the necessary requirements were met the author was able to create the migrations and apply them to the database in order to have a functional system. Appendix C.2

### 5.4.5 Edit User Profile

As a requirement of the system the author had to implement a way for customers to modify the information related to their profile. These will be available on the user profile page and some are already filled in automatically as they were provided during the registration process, such as email address, username, and email address, however, additional information is available to fill in such as the registration number as this is not given by the user during the registration process, which can be edited by the customer in case they change their vehicle in the future. On the profile page an “edit” button was implemented by the author, when the form is completed with the new information and the button is pressed, the data is registered and altered in the database. The user edit process is implemented by defining a view within the user application of the project which renders the “profile.html” file, the view that allows profiles to be edited is defined as “profile” and works by providing a form for both the user and the profile. The user profile form is passed a method called “UserUpdateForm” which is provided by Django, while for the profile, the process is similar as the form uses the method “ProfileUpdateForm”. The form fetches the information provided when editing the profile by requesting the information provided using POST, after which checks are carried out to see if the forms are valid, if they are not the changes will not take place; if they are the form contents are saved and the user is prompted with a message confirming that the changes have applied.

### 5.4.6 Editing the profile Image

Within the profile model, the “image” field was specified by the author in order to allow storage of the profile picture of the user, which initially sets the image to the default.jpg file. The user is able to select the new image on their profile page after uploading a new file with the “Choose file” button as seen in figure 5.4.6.1 below. To allow images to be saved in the system and be displayed the author has defined within the Django project a media folder containing a new directory called “profile\_pics”. Once a user updates their profile picture and upload an image this get processed further using a functionality which is provided by a Python library discussed in the method chapter called Pillow, which was used to crop any file to a 300 by 300 pixels image; as a result smaller files which are easier to store and require less resources are obtained. The author has implemented this to optimize the navigation of the website as smaller images take less time to load, furthermore it avoids possible misuse by some users who might try to upload very large images to impact the performance of the website.

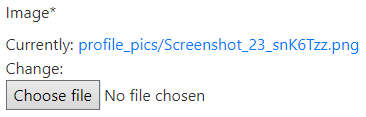


Figure 5.4.6.1

## 5.5 Parking Sessions Logic

The feature that required the most time to develop was the implementation of the car park recognition system within the Django application. Many obstacles were encountered during this phase of development as the logic of the parking system had to be centered around the python system built previously. The functioning Following is an explanation of how the author has decided to incorporate the recognition in the Django application.

### 5.5.1 Step 1

Firstly a command was defined (“makepark”), which when called triggers the start of the process and displays a success message if successful or an error message otherwise.

### 5.5.2 Step 2

The subsequent step began in carpark application folder by creating a python file containing the refined version of the recognition system created in the Spyder IDE. Within the “registration\_recognition.py” file is defined a method called “get\_registration(img\_path)” which runs the recognition on the image of the vehicle and outputs a string containing the registration of the vehicle as discussed earlier in the results chapter.

### 5.5.3 Step 3

In order to operate the parking logic of the system, within the utils.py file a method called parking is defined. This triggers the number plate recognition of the system by calling the “get\_registration” function on a given image. This then checks if the registration plate is associated to a customer, returning false to the “makepark” command and therefore an error. If a user matches the output of the “get\_registration” method when called, a check is carried out in order to verify whether the customer is currently parked or not. This is needed to understand if the customer is entering or exiting the carpark: a boolean field “parked” specifies the current parking status of the user and if found to be false, the park method sets it to true and saves the current time using the “timezone.now()” method in the in the “parktime” filed of the user profile. If the customer is found to currently be parked, the elapsed time of their stay is calculated by subtracting the time at entry from the time at which the parking sessions ended. The amount due is then worked out by multiplying the number of minutes elapsed by 0.05 (a minute of parking costs £0.05) after which a new UserSession associated with the user is created and saved in the database specifying the elapsed time and amount due. Finally, the customer’s parking status is set to false and their “parktime” field is cleared, therefore setting up the user’s profile for an eventual new parking session. The “parking” method returns a value of “True” confirming that the operation has been carried out successfully.

## 5.6 Admin Panel

Django provides an admin panel which allows the Administrator of the system to manage the various aspects of the application. The url path to access this is provided by default and is set to domain/admin, where a login page appears and after filling this in with admin credentials the panel can be accessed. It is possible to add functionalities, this the author has done so by modifying the admin.py document and importing the models that are required to display information regarding user profiles and parking sessions.

## 5.7 Testing

To verify the integrity of the system the author has carried out tests using different methods. These were also needed to ensure that requirements are met, and bugs dealt with. The types of testing carried out consisted of tasks that had to output the desired result in order to be marked as passed. Extended documentation of testing carried out is available in Appendix D.

# Chapter 6 - Conclusion and Discussion

This chapter aims at giving a reflection on the project objectives as well as evaluating its success by considering the research carried out, design phase and the implementation. The extent to which the author has been able to meet requirements and accomplish the desired results are discussed in order to is used to consider prospects for any future work, knowledge gained and alternative executions that could have led to an improved result.

## 6.1 Main objective

The main objectives of this project as stated in section (introduction) were: “To create a system that integrates a number plate recognition system with a system that can automate the process of working out the stay of customers. Integrated into web portal application where users can register and execute payments in order to ease the process of parking their vehicle within the premises of a car park.” After carrying out the necessary tests and ensuring they are passed, the author considers this object met as the final result is able to correctly recognize number plates, automate the process of working out the time for which the customer was parked, output the amount due, provide a web portal for users to register where to manage their account, provide an admin panel for administration tasks and a contact function for users.

## 6.2 Sub objectives

### 6.2.1 Registration and Login

The requirements were met as users are able to make use of this feature as intended.

### 6.2.2 Admin panel

This requirement was also met as the admin panel provides the required functionalities.

### 6.2.3 Plate Recognition

The system recognizing car number plate was successfully developed and functions as intended.

### 6.2.4 Parking session automation

User car park session duration is calculated automatically and the amount due is also worked out correctly.

## 6.3 Future Work

The system may have to be implemented for real world task in the future as it could be incorporated in existing car parks where the systems currently in place are considered obsolete and need to be upgraded. It would be possible to adapt the system to function in a variety of car parks and the number plate system used for different purposes such as verifying that assigned parking spaces are only being used by the entitled person in business car parks, or that only vehicles with permission access certain areas. The scalability of the system also means that it could be adapted to handle personal use as well as cases where it will have to deals with constant streams of vehicles.

## 6.4 Knowledge Gained

Being this project’s magnitude something that the author has never undertaken before it was a very constructive experience. Previous to this the author had only embarked on small projects mainly consisting of one system with a set of basic functionalities. This project helped them adapt to problems and learn how to solve issues when encountering difficulties during development. Furthermore, the skill level prior to this project’s development the author had never made use of the Django framework and had to carry out extensive research on its functionalities in order to learn how to integrate different systems and have them work in with each other seamlessly. This project has allowed the author to be much more confident with their Python skills which included only basics knowledge of the Python syntax no real-world scenarios experience and has allowed them to develop a real passion for this programming language as well as discover the limitless possibilities available with Python and its libraries.

## 6.5 Conclusion

The project has helped the author mature on a professional level by imposing the use of best practices during the various phases of development, furthermore, this experience has been really an incredible journey that gifted the opportunity of learning new skills and adapting to unforeseen situations. The author was able to prove to themselves that they are able of achieving goals on their own without the guidance that they would have required in the past; this is something that they could have never imagined when embarking on their university studies in 2018 and is very proud of the level reached, to a point where they can consider themselves ready for the work environment in the field of computer science.

# Glossary

The following provides a guide of the meaning of certain terms with which the viewer might not be familiar.

|  |  |
| --- | --- |
| Term | Definition |
| Android | Operating system used by many smartphones available on the market today. |
| MVT | Development architecture that stands for Model View Template, used by frameworks such as Django. |
| PIL | Python library, short for pillow. |
| Backend | Software development terminology referring to the logic of the application that is not visible to the end user. |
| Frontend | Software development terminology referring to the part of the application that is visible to the user and can be interacted with. |
| iOS | Operating system used by devices manufactured by Apple. |
| Meta | Used to specify metadata used by certain Django classes. Specifies data to be passed by forms. |
| Windows | Operating system commonly used by many personal computers |
| Linux | Operating system commonly used by personal computers and server sided applications. |
| .ipynb | Files extension used by the Jupyter Notebook. |
| .py | .py is used at the end certain files as their extension to signify that the file is to be interpreted as a Python file. |

# References

Below are listed all the online resource accessed by the author to help create this project.

En.wikipedia.org. 2021. *Bilateral filter - Wikipedia*. [online] Available at: <https://en.wikipedia.org/wiki/Bilateral\_filter> [Accessed 2 February 2021].

docs.opencv.org. (n.d.). *OpenCV: Object Detection*. [online] Available at: <https://docs.opencv.org/4.5.0/d5/d54/group\_\_objdetect.html> [Accessed February 2021].

Docs.djangoproject.com. 2021. *Writing your first Django app, part 1 | Django documentation | Django*. [online] Available at: <https://docs.djangoproject.com/en/3.2/intro/tutorial01/> [Accessed February 2021].

‌Docs.djangoproject.com. 2021. *Django documentation | Django documentation | Django*. [online] Available at: <https://docs.djangoproject.com/en/3.2/> [Accessed February 2021].

template, l., Beasley, J. and Naeem, K., 2021. *loop backwards using django template*. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/12680691/loop-backwards-using-django-template> [Accessed March 2021].

DateTimeFields, D., Mansouri, J., Vandelay, A. and Weyne, L., 2021. *Django model elapsed time between two DateTimeFields*. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/55150925/django-model-elapsed-time-between-two-datetimefields> [Accessed March 2021].

interface, D., Jan, Z. and R, R., 2021. *Django. How to add an extra field in User model and have it displayed in the admin interface*. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/42478191/django-how-to-add-an-extra-field-in-user-model-and-have-it-displayed-in-the-adm> [Accessed March 2021].

CharField with fixed length, h. and Milia, G., 2021. *CharField with fixed length, how?*. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/2470760/charfield-with-fixed-length-how> [Accessed March 2021].

TemplateDoesNotExist - file exists, n., Bauer, J. and Belchak, B., 2021. *TemplateDoesNotExist - file exists, no permissions issue*. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/4975842/templatedoesnotexist-file-exists-no-permissions-issue> [Accessed March 2021].

GeeksforGeeks. 2021. *Python OpenCV | cv2.cvtColor() method - GeeksforGeeks*. [online] Available at: <https://www.geeksforgeeks.org/python-opencv-cv2-cvtcolor-method/> [Accessed March 2021].

Docs.djangoproject.com. 2021. *Writing your first Django app, part 2 | Django documentation | Django*. [online] Available at: <https://docs.djangoproject.com/en/dev/intro/tutorial02/#creating-models> [Accessed March 2021].

forms?, w., S, V. and Tomita, Y., 2021. *whats the difference between Django models and forms?*. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/5481713/whats-the-difference-between-django-models-and-forms> [Accessed March 2021].

Vаиітy, A. and Pinsard, A., 2021. *What does on\_delete do on Django models?*. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/38388423/what-does-on-delete-do-on-django-models?rq=1> [Accessed March 2021].

Coding for Entrepreneurs. 2021. *Install Python 3.8, Virtual Environments using Pipenv, Django 3+ on Windows // Blog // Coding for Entrepreneurs*. [online] Available at: <https://www.codingforentrepreneurs.com/blog/install-python-django-on-windows> [Accessed March 2021].

GitHub. 2021. *JaidedAI/EasyOCR*. [online] Available at: <https://github.com/JaidedAI/EasyOCR> [Accessed March 2021].

Pytorch.org. 2021. *PyTorch documentation — PyTorch 1.8.1 documentation*. [online] Available at: <https://pytorch.org/docs/stable/index.html> [Accessed May 2021].

Circuit Digest. 2021. *License Plate Recognition using Raspberry Pi and OpenCV*. [online] Available at: <https://circuitdigest.com/microcontroller-projects/license-plate-recognition-using-raspberry-pi-and-opencv> [Accessed April 2021].

Renotte, N., 2021. *Optical Character Recognition with EasyOCR and Python | OCR PyTorch*. [online] Youtube.com. Available at: <https://www.youtube.com/watch?v=ZVKaWPW9oQY> [Accessed April 2021].

Dsouza, J., 2021. *jasmcaus/opencv-course*. [online] GitHub. Available at: <https://github.com/jasmcaus/opencv-course> [Accessed May 2021].

Renotte, N., 2021. *nicknochnack/EasyOCR*. [online] GitHub. Available at: <https://github.com/nicknochnack/EasyOCR> [Accessed May 2021].

Schafer, C., 2021. *User-Registration-Form*. [online] GitHub. Available at: <https://github.com/CoreyMSchafer/code\_snippets/blob/master/Django\_Blog/06-User-Registration-Form/django\_project/users/forms.py> [Accessed April 2021].

cse.unr.edu. 2021. *(Trucco, Chapt 4 AND Jain et al., Chapt 5)*. [online] Available at: <https://www.cse.unr.edu/~bebis/CS791E/Notes/EdgeDetection.pdf> [Accessed February 2021].

Muthukrishnan, R. and Radha, M., 2021. *EDGE DETECTION TECHNIQUES FOR IMAGE SEGMENTATION*. [online] Airccse Publishing Corporation. Available at: <http://airccse.org/journal/jcsit/1211csit20.pdf> [Accessed February 2021].

Schafer, C. 2021. *CoreyMSchafer/code\_snippets*. [online] Available at: <https://github.com/CoreyMSchafer/code\_snippets/blob/master/Django\_Blog/snippets/main.css> [Accessed 18 May 2021].

# Appendix A – Project Description Document

BSc (Hons) Computer Science

IN3007 – Final Year Project Report

Project Description Document



Car Park management system

**Written and submitted by Claudiu Andrei**

**Email:** [**claudiu.andrei@city.ac.uk**](mailto:claudiu.andrei@city.ac.uk)

**ACZH866**

Table Of Contents

[Appendix A – Project Description Document 52](#_Toc72200428)

[Project proposal 54](#_Toc72200432)

[Problem to be solved 54](#_Toc72200433)

[Project Objectives 54](#_Toc72200434)

[S.M.A.R.T. Objectives 55](#_Toc72200435)

[Beneficiaries 56](#_Toc72200436)

[Risks 57](#_Toc72200437)

[Work Plan 58](#_Toc72200438)

[Research Ethics Review Form: BSc, MSc and MA Projects 59](#_Toc72200439)

[PART A: Ethics Checklist 59](#_Toc72200440)

[PART B: Ethics Proportionate Review Form 62](#_Toc72200441)

[Changes 65](#_Toc72200442)

[Templates for Consent and Information 65](#_Toc72200443)

[Further Information 66](#_Toc72200444)

## Project proposal

### Problem to be solved

We live in a world where being able to travel quickly is essential, no one wants to waste time in a society where free time is limited for most people. Due to that reason, almost every time we leave our homes, we decide to reach our destination by driving there. For obvious reasons we only chose to travel by car when heading to places with available car parking.

Thanks to new technologies being developed around cars every year, we can now purchase new cars that are safer than ever. Since we are in a country like the United Kingdom, where every person who wants to own a car is able to do so, it is to be expected for the number of cars on the streets to increase on a yearly basis. Furthermore, as older cars become depreciated more and more every year, a greater number of people are able to afford and maintain them, resulting in a much greater need for car parking space. Because of this, the car park sector is one of the most prolific business types these days, with car parks being built everywhere for people to park their car when they reach their destination or store them when they are not using them.

The problem with modern car parks is one that all of us have encountered at times: the experience when it comes to paying for parking can be at times frustrating and time consuming. Even if many car parks are becoming very advanced technologically, an even greater number are still utilizing outdated methods to collect payments from their customers and operate the stream of cars entering and exiting the car park. This issue is not a let down for the customers only, but also for the people managing the establishments, as it results in a waste of resources.

## Project Objectives

The way I intend to develop this is by having a system which allows cars parks to collect payments without effort, whilst also allowing customers to be carefree when entering or leaving the premises. Consequently, everything should be seamlessly implemented to work in synchrony with the different parts of the system, by this I mean that there should be a variety of systems communicating with each other and exchange information. The way I visualized my implementation of this was building a software that receives inputs as an image and is able to the extrapolate information from the input, such as a number plate, or the shape of a vehicle. Once the software has recognised the presence of a number plate it can then work out what its characters are and submit this information to a different system.

This will save a log of the number plate as well as the time it was first recognised, allowing to use this information in the future. There should also be a database that has a list of registered customers, where the registration of a vehicle that has just entered the car park will be checked against the records and if the vehicle is registered there won’t be any need to create a new instance.

Customers who have registered through the website will have to provide payments details in order for them to be able to pay for their stay. When the number plate recognition system logs that they have left the car park, thereafter, the amount due is worked out and provided to the customers to complete and check the session as paid for. If a customer has not registered there would need to be some sort of error message communicating that it was not possible to associate the number plate to a user. This solution improves upon existing methods because it doesn’t use any gates when allowing cars to enter and exit without stopping or requiring the customer to have to do any additional steps, making it much more enjoyable in the long run, whilst also allowing the business managers to spend less resources on payment booths and staff workers.

### S.M.A.R.T. Objectives

* Number plate recognition system: This objective aims at implementing a solution that can look at an image and recognize that a number plate is present in it, then return the characters to database. As I don’t have the knowledge to implement this as of now, I will have to use the material learned in the Introduction to A.I module as well as some online tutorials. I will test this with several images containing car number plates, and if they are recognised correctly under certain circumstances the test will be considered a pass.
* Component to work out how much each customer gets charged for the amount of time they have used the car park for. In order to achieve this, I will have the number plate recognition system submit the registration number of the vehicle to this program which will then start to work out the amount to be paid when the customer leaves the car park. I will investigate how to implement this and will decide the programming language to use at a later date. In order to test for this, I will use a set amount of time and then see if the number returned at the end matches the amount that should be paid.
* Customer registration system for customers to register their vehicle other personal information, if they change their vehicle in the future, they will be allowed to edit their car registration number in the system so that their new vehicle will be recognised when parked. To achieve this, I will create a web portal accessible to customers. In order to test for this, I will use dummy data to create accounts and test if the process of registration works as intended.
* Admin panel: this should be present in order to allow the administrator of the system to manage the data present in the system as well as the customers information. Furthermore, it will be used to determine if the customer has any unpaid sessions by viewing their profile as well as for editing information and dealing with customer contact queries if they are experiencing issues or need to edit information.

## Beneficiaries

|  |  |
| --- | --- |
| Beneficiaries | Reason |
| General Public. | Anyone with interest in this such of number plate recognition system might find this project useful for their research. |
| Companies and organisations. | This project may be beneficial for other companies as it may be used for workers to upskill their knowledge and become accustomed to using different working environments. |
| Developers and Engineers | For new developers this project might be used to |
| Researchers. | To develop the system further and investigate how it could be made better. |
| Organisations that might use the software for non-profit reasons. | The software might be used by non-profit organisations like charities for free. |

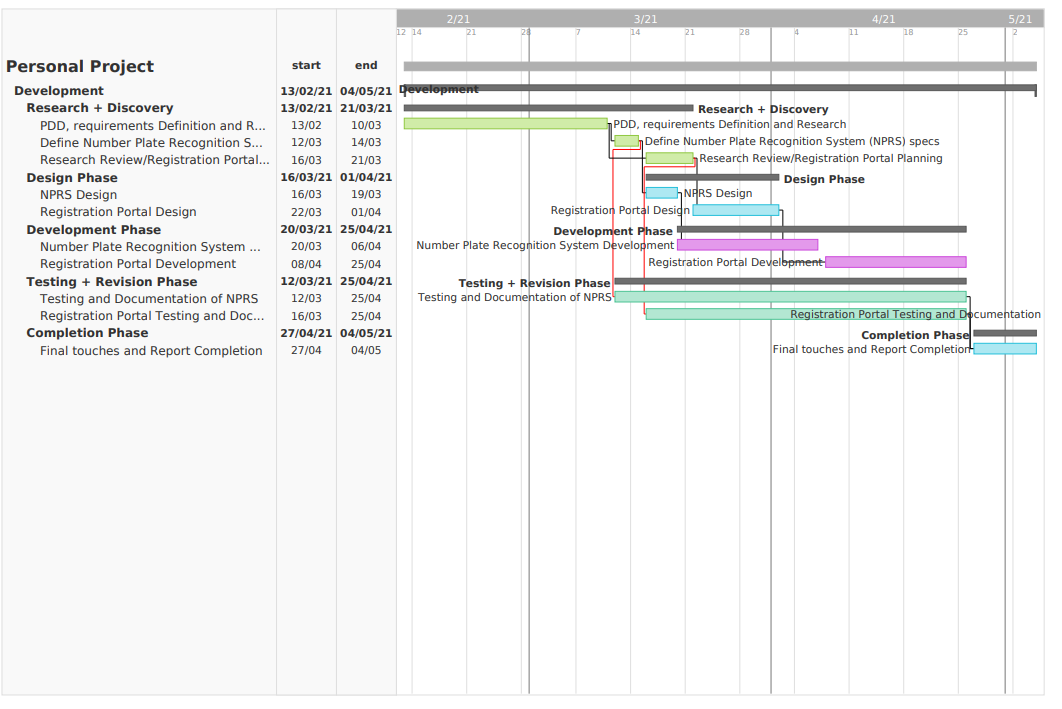
## Risks

There are several risks that could affect the outcome of this project. The ones that are controllable have been identified and listed in the table below. Furthermore, their Likelihood (1 = low, 5 = high), Impact (1 = low, 5 = high), Risk Score (Likelihood \* Impact: 1 = low, 25 = high)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk Description** | **Likelihood** | **Impact** | **Risk Score** | **Mitigation** |
| **Learning how to build the number plate recognition system:** This is going to be one of the hardest tasks which will require extensive research and testing. | 5 | 4 | 20 | I will start to look into creating this sort of system early on, and allow myself time to understand how to implement this to work with the rest of the program. |
| **Customer registration portal:** Since I have never built this sort of system, creating something that will work with the rest of the project components will be a difficult challenge. | 4 | 3 | 12 | This will not be the most difficult component to implement therefore I can expect to mitigate its impact by looking at online tutorials and guides. |
| **Database creation:** Ensuring everything works well with the rest of the components and there are no mistakes in the database. | 4 | 4 | 16 | Creating the database will be one of the most challenging parts of the project since there is much data to be dealt with, therefore I will have to plan everything meticulously in order to ensure there are no errors. |
| **Other modules**: Throughout the project I will also have to focus and allow time to work on the other module for this semester. | 5 | 2 | 10 | In order to make sure I allow enough time for both my other modules and this project; pre-emptive planning is going to be really important. |

## Work Plan

For my work plan I have decided to implement an agile methodology since I recon it will allow me to be as flexible with my development as possible.



|  |
| --- |
| Research Ethics Review Form: BSc, MSc and MA Projects **Computer Science Research Ethics Committee (CSREC)**  <http://www.city.ac.uk/department-computer-science/research-ethics> |

Undergraduate and postgraduate students undertaking their final project in the Department of Computer Science are required to consider the ethics of their project work and to ensure that it complies with research ethics guidelines. In some cases, a project will need approval from an ethics committee before it can proceed. Usually, but not always, this will be because the student is involving other people (“participants”) in the project.

In order to ensure that appropriate consideration is given to ethical issues, all students must complete this form and attach it to their project proposal document. There are two parts:

***PART A: Ethics Checklist***. All students must complete this part. The checklist identifies whether the project requires ethical approval and, if so, where to apply for approval.

***PART B: Ethics Proportionate Review Form****.* Students who have answered “no” to all questions in A1, A2 and A3 and “yes” to question 4 in A4 in the ethics checklist must complete this part. The project supervisor has delegated authority to provide approval in such cases that are considered to involve MINIMAL risk. The approval may be ***provisional*** *– identifying the planned research as*likely to involve MINIMAL RISK. In such cases you must additionally seek ***full approval*** from the supervisor as the project progresses and details are established. ***Full approval*** must be acquired in writing, before beginning the planned research.

## PART A: Ethics Checklist

**Part A: Ethics Checklist**

|  |  |  |
| --- | --- | --- |
| **A.1 If you answer YES to any of the questions in this block, you must apply to an appropriate external ethics committee for approval and log this approval as an External Application through Research Ethics Online - https://ethics.city.ac.uk/** | | *Delete as appropriate* |
| 1.1 | Does your research require approval from the National Research Ethics Service (NRES)?  e.g. because you are recruiting current NHS patients or staff?  If you are unsure try - https://www.hra.nhs.uk/approvals-amendments/what-approvals-do-i-need/ | **NO** |
| 1.2 | Will you recruit participants who fall under the auspices of the Mental Capacity Act?  Such research needs to be approved by an external ethics committee such as NRES or the Social Care Research Ethics Committee - http://www.scie.org.uk/research/ethics-committee/ | **NO** |
| 1.3 | Will you recruit any participants who are currently under the auspices of the Criminal Justice System, for example, but not limited to, people on remand, prisoners and those on probation?  Such research needs to be authorised by the ethics approval system of the National Offender Management Service. | **NO** |
| **A.2 If you answer YES to any of the questions in this block, then unless you are applying to an external ethics committee, you must apply for approval from the Senate Research Ethics Committee (SREC) through Research Ethics Online -**  **https://ethics.city.ac.uk/** | | *Delete as appropriate* |
| 2.1 | Does your research involve participants who are unable to give informed consent?  For example, but not limited to, people who may have a degree of learning disability or mental health problem, that means they are unable to make an informed decision on their own behalf. | **NO** |
| 2.2 | Is there a risk that your research might lead to disclosures from participants concerning their involvement in illegal activities? | **NO** |
| 2.3 | Is there a risk that obscene and or illegal material may need to be accessed for your research study (including online content and other material)? | **NO** |
| 2.4 | Does your project involve participants disclosing information about special category or sensitive subjects?  *For example, but not limited to: racial or ethnic origin; political opinions; religious beliefs; trade union membership; physical or mental health; sexual life; criminal offences and proceedings* | **NO** |
| 2.5 | Does your research involve you travelling to another country outside of the UK, where the Foreign & Commonwealth Office has issued a travel warning that affects the area in which you will study?  *Please check the latest guidance from the FCO -* [*http://www.fco.gov.uk/en/*](http://www.fco.gov.uk/en/) | **NO** |
| 2.6 | Does your research involve invasive or intrusive procedures?  These may include, but are not limited to, electrical stimulation, heat, cold or bruising. | **NO** |
| 2.7 | Does your research involve animals? | **NO** |
| 2.8 | Does your research involve the administration of drugs, placebos or other substances to study participants? | **NO** |
| **A.3 If you answer YES to any of the questions in this block, then unless you are applying to an external ethics committee or the SREC, you must apply for approval from the Computer Science Research Ethics Committee (CSREC) through Research Ethics Online - https://ethics.city.ac.uk/**  **Depending on the level of risk associated with your application, it may be referred to the Senate Research Ethics Committee.** | | *Delete as appropriate* |
| 3.1 | Does your research involve participants who are under the age of 18? | **NO** |
| 3.2 | Does your research involve adults who are vulnerable because of their social, psychological or medical circumstances (vulnerable adults)?  This includes adults with cognitive and / or learning disabilities, adults with physical disabilities and older people. | **NO** |
| 3.3 | Are participants recruited because they are staff or students of City, University of London?  For example, students studying on a particular course or module.  If yes, then approval is also required from the Head of Department or Programme Director. | **NO** |
| 3.4 | Does your research involve intentional deception of participants? | **NO** |
| 3.5 | Does your research involve participants taking part without their informed consent? | **NO** |
| 3.5 | Is the risk posed to participants greater than that in normal working life? | **NO** |
| 3.7 | Is the risk posed to you, the researcher(s), greater than that in normal working life? | **NO** |
| **A.4 If you answer YES to the following question and your answers to all other questions in sections A1, A2 and A3 are NO, then your project is deemed to be of MINIMAL RISK.**  **If this is the case, then you can apply for approval through your supervisor under PROPORTIONATE REVIEW. You do so by completing PART B of this form.**  **If you have answered NO to all questions on this form, then your project does not require ethical approval. You should submit and retain this form as evidence of this.** | | *Delete as appropriate* |
| 4 | Does your project involve human participants or their identifiable personal data?  *For example, as interviewees, respondents to a survey or participants in testing.* | **YES** |

## PART B: Ethics Proportionate Review Form

If you answered YES to question 4 and NO to all other questions in sections A1, A2 and A3 in PART A of this form, then you may use PART B of this form to submit an application for a proportionate ethics review of your project. Your project supervisor has delegated authority to review and approve this application under proportionate review. You must receive final approval from your supervisor in writing before beginning the planned research.

However, if you cannot provide all the required attachments (see B.3) with your project proposal (e.g. because you have not yet written the consent forms, interview schedules etc), the approval from your supervisor will be ***provisional***. You **must** submit the missing items to your supervisor for approval prior to commencing these parts of your project. Once again, you must receive written confirmation from your supervisor that any provisional approval has been superseded by with ***full approval*** of the planned activity as detailed in the full documents. **Failure to follow this procedure and demonstrate that final approval has been achieved may result in you failing the project module.**

Your supervisor may ask you to submit a full ethics application through Research Ethics Online, for instance if they are unable to approve your application, if the level of risks associated with your project change, or if you need an approval letter from the CSREC for an external organisation.

|  |  |  |
| --- | --- | --- |
| **B.1 The following questions must be answered fully.**  **All grey instructions must be removed.** | | *Delete as appropriate* |
| 1.1. | Will you ensure that participants taking part in your project are fully informed about the purpose of the research? | **YES** |
| 1.2 | Will you ensure that participants taking part in your project are fully informed about the procedures affecting them or affecting any information collected about them, including information about how the data will be used, to whom it will be disclosed, and how long it will be kept? | **YES** |
| 1.3 | When people agree to participate in your project, will it be made clear to them that they may withdraw (i.e. not participate) at any time without any penalty? | **YES** |
| 1.4 | Will consent be obtained from the participants in your project?  Consent from participants will be necessary if you plan to involve them in your project or if you plan to use identifiable personal data from existing records. “Identifiable personal data” means data relating to a living person who might be identifiable if the record includes their name, username, student id, DNA, fingerprint, address, etc.  *If YES, you must attach drafts of the participant information sheet(s) and consent form(s) that you will use in section B.3 or, in the case of an existing dataset, provide details of how consent has been obtained.*  *You must also retain the completed forms for subsequent inspection. Failure to provide the completed consent request forms will result in withdrawal of any earlier ethical approval of your project.* | **YES** |
| 1.5 | Have you made arrangements to ensure that material and/or private information obtained from or about the participating individuals will remain confidential? | **YES** |

|  |  |  |
| --- | --- | --- |
| **B.2 If the answer to the following question (B2) is YES, you must provide details** | | *Delete as appropriate* |
| 2 | Will the research be conducted in the participant’s home or other non-University location?  *If* ***YES****, you must provide details of how your safety will be ensured.* | **YES** |

|  |  |  |  |
| --- | --- | --- | --- |
| **B.3 Attachments**  **ALL of the following documents MUST be provided to supervisors if applicable.**  **All must be considered prior to final approval by supervisors.**  **A written record of final approval must be provided and retained.** | ***YES*** | ***NO*** | ***Not Applicable*** |
| Details on how safety will be assured in any non-University location, including risk assessment if required (see B2) | **X** |  |  |
| Details of arrangements to ensure that material and/or private information obtained from or about the participating individuals will remain confidential (see B1.5)  *Any personal data must be acquired, stored and made accessible*  *in ways that are GDPR compliant.* | **X** |  |  |
| Full protocol for any workshops or interviews\*\* |  |  | **X** |
| Participant information sheet(s)\*\* |  |  | **X** |
| Consent form(s)\*\* |  |  | **X** |
| Questionnaire(s)\*\*  *sharing a Qualtrics survey with your supervisor is recommended.* |  |  | **X** |
| Topic guide(s) for interviews and focus groups\*\* |  |  | **X** |
| Permission from external organisations or Head of Department\*\*  *e.g. for recruitment of participants* |  |  | **X** |

*\*\*If these items are not available at the time of submitting your project proposal, then* ***provisional approval*** *can still be given, under the condition that you must submit the final versions of all items to your supervisor for approval at a later date.* ***All*** *such items* ***must*** *be seen and approved by your supervisor before the activity for which they are needed begins. Written evidence of* ***final approval*** *of your planned activity must be acquired from your supervisor before you commence.*

## Changes

If your plans change and any aspects of your research that are documented in the approval process change as a consequence, then any approval acquired is invalid. If issues addressed in Part A (the checklist) are affected, then you must complete the approval process again and establish the kind of approval that is required. If issues addressed in Part B are affected, then you must forward updated documentation to your supervisor and have received written confirmation of approval of the revised activity before proceeding.

## Templates for Consent and Information

You must use the templates provided by the University as the basis for your participant information sheets and consent forms. You **must** adapt them according to the needs of your project before you submit them for consideration.

Participant Information Sheets, Consent Forms and Protocols must be consistent. Please ensure that this is the case prior to seeking approval. Failure to do so will slow down the approval process.

We strongly recommend using Qualtrics to produce digital information sheets and consent forms.

## Further Information

<http://www.city.ac.uk/department-computer-science/research-ethics>

https://www.city.ac.uk/research/ethics/how-to-apply/participant-recruitment

https://www.city.ac.uk/research/ethics

# Appendix B – Requirements Documentation

Below are listed the main use cases and flows of the system

|  |  |
| --- | --- |
| Use Case: Login | |
| ID: | UC01 – Login |
| Description: | The user can access their user account |
| Primary Actors: | User |
| Secondary Actors: | Django System |
| Preconditions: | User has registered an account |
| Main Flow: | The user navigates to the login page using the buttons on the website.  1) The user is prompted with form requesting authentication data  2) The user fills in the fields with the required information  3) The information is processed by the system and appears to be correct  4) The user is able to access their profile |
| Post Conditions: | User is redirected to their profile page |
| Alternate Flows: | The system outputs an error message if it fails to authenticate the user |

|  |  |
| --- | --- |
| Use Case: Logout | |
| ID: | UC02 – Logout |
| Description: | The currently logged in user is able to log out of their account |
| Primary Actors: | User |
| Secondary Actors: | Django System |
| Preconditions: | The user is logged into their account |
| Main Flow: | The use case begins once the user attempts to log out  1) The user selects the Log Out button from the navigation bar  2) The application ends the user session and logs them out  3) The user is redirected to a different page  4) A message confirming the action is displayed |
| Post Conditions: | User is logged out of their account |
| Alternate Flows: | N/A |

|  |  |
| --- | --- |
| Use Case: Register | |
| ID: | UC03 – Register |
| Description: | The user registers their account in the system |
| Primary Actors: | User |
| Secondary Actors: | Django System |
| Preconditions: | The user is not logged into the system |
| Main Flow: | The use case begins when the user navigates to the register page of the site and attempts to create a new account  1) The user presses the Register button on the navigation bar  2) The user is prompted to a form requesting the required information  3) The user enters a username, email address, password, and password confirmation  4) User submits the form with the allocated button  5) Account is created in the database |
| Post Conditions: | 1) User account is created  2) User is redirected to home page  3) User is showed a success message confirming account creation |
| Alternate Flows: | 1) The user inputs an invalid username  1.1) The user is not able to register and will be showed an error  2) Password selected by the user does not meet criteria  2.1) The user will be showed and error  3) Email is not valid  3.1) The user will see and error and asked to fill the form again  4) Passwords do not match  4.1) Account is not created and the user will be asked to try again  5) The user is to access the login page |

|  |  |
| --- | --- |
| Use Case: Profile Page | |
| ID: | UC04 – Profile Page |
| Description: | The user manages their account using the Profile page |
| Primary Actors: | User |
| Secondary Actors: | Django System |
| Preconditions: | The user is logged into the system |
| Main Flow: | The use case begins when the user navigates to their Profile page  1) The user presses the My Account button on the navigation section  2) The user is prompted to the form showcasing their information |
| Post Conditions: | The user has access to their information |
| Alternate Flows: | N/A |

|  |  |
| --- | --- |
| Use Case: Edit Username | |
| ID: | UC05 – Edit Username |
| Description: | The user edits their username |
| Primary Actors: | User |
| Secondary Actors: | Django System |
| Preconditions: | The user is logged into the system |
| Main Flow: | The use case begins when the user navigates to their Profile page  1) The user presses the My Account button on the navigation section  2) The user is prompted to the form showcasing their information  3) The user edits the username field  4) The user presses the Update button  5) The information is saved in the system’s database |
| Post Conditions: | 1) The user is redirected to their profile page  2) The user is showed a success message |
| Alternate Flows: | 1) The user has selected an invalid username  2) The user presses the update button  3) The user is showed an error message |

|  |  |
| --- | --- |
| Use Case: Edit Email | |
| ID: | UC06 – Edit Email |
| Description: | The user edits their Email address |
| Primary Actors: | User |
| Secondary Actors: | Django System |
| Preconditions: | The user is logged into the system |
| Main Flow: | The use case begins when the user navigates to their Profile page  1) The user presses the My Account button on the navigation section  2) The user is prompted to the form showcasing their information  3) The user edits the Email field  4) The user presses the Update button  5) The information is saved in the system’s database |
| Post Conditions: | 1) The user is redirected to their profile page  2) The user is showed a success message |
| Alternate Flows: | 1) The user has selected an invalid email  2) The user presses the update button  3) The user is showed an error message |

|  |  |
| --- | --- |
| Use Case: Edit Image | |
| ID: | UC07 – Edit Image |
| Description: | The user edits their profile image |
| Primary Actors: | User |
| Secondary Actors: | Django System |
| Preconditions: | The user is logged into the system |
| Main Flow: | The use case begins when the user navigates to their Profile page  1) The user presses the My Account button on the navigation section  2) The user is prompted to the form showcasing their information  3) The user presses the chose file option  4) The user uploads and image  5) The user presses the update button  6) The new image is saved in the database and displayed on their profile |
| Post Conditions: | 1) The user is redirected to their profile page  2) The user is showed a success message |
| Alternate Flows: | 1) The user has selected an invalid file  2) The user presses the update button  3) The user is showed an error message |

|  |  |
| --- | --- |
| Use Case: Edit Registration Number | |
| ID: | UC08 – Edit Registration Number |
| Description: | The user edits their profile image |
| Primary Actors: | User |
| Secondary Actors: | Django System |
| Preconditions: | The user is logged into the system |
| Main Flow: | The use case begins when the user navigates to their Profile page  1) The user presses the My Account button on the navigation section  2) The user is prompted to the form showcasing their information  3) The user enters a new registration number  4) The user presses the update button  5) The new registration is saved in the system |
| Post Conditions: | 1) The user is redirected to their profile page  2) The user is showed a success message |
| Alternate Flows: | 1) The user has selected an invalid file  2) The user presses the update button  3) The user is showed an error message |

|  |  |
| --- | --- |
| Use Case: Admin Panel | |
| ID: | UC09 – Admin Panel |
| Description: | The admin accesses the admin panel |
| Primary Actors: | Admin |
| Secondary Actors: | Django System |
| Preconditions: | The user is logged in as a staff user |
| Main Flow: | The use case begins when the user logs into the admin panel  1) The user is able to control all aspects of the database made accessible to them by the author |
| Post Conditions: | The altered data is saved by the system |
| Alternate Flows: | N/A |

|  |  |
| --- | --- |
| Use Case: Submit Contact Request | |
| ID: | UC10 – Submit Contact Request |
| Description: | Users can submit contact requests |
| Primary Actors: | User |
| Secondary Actors: | Django System |
| Preconditions: | Users are logged into the system |
| Main Flow: | The use case begins with the user navigating to the contact section of the website  1) The user presses the contact us button  2) The user Fills in the information required  3) The user presses the submit button  4) The information is stored, and the request can be viewed by the staff members |
| Post Conditions: | 1) The data is stored in the system  2) Request is created  3) The staff users are able to view the request |
| Alternate Flows: | N/A |

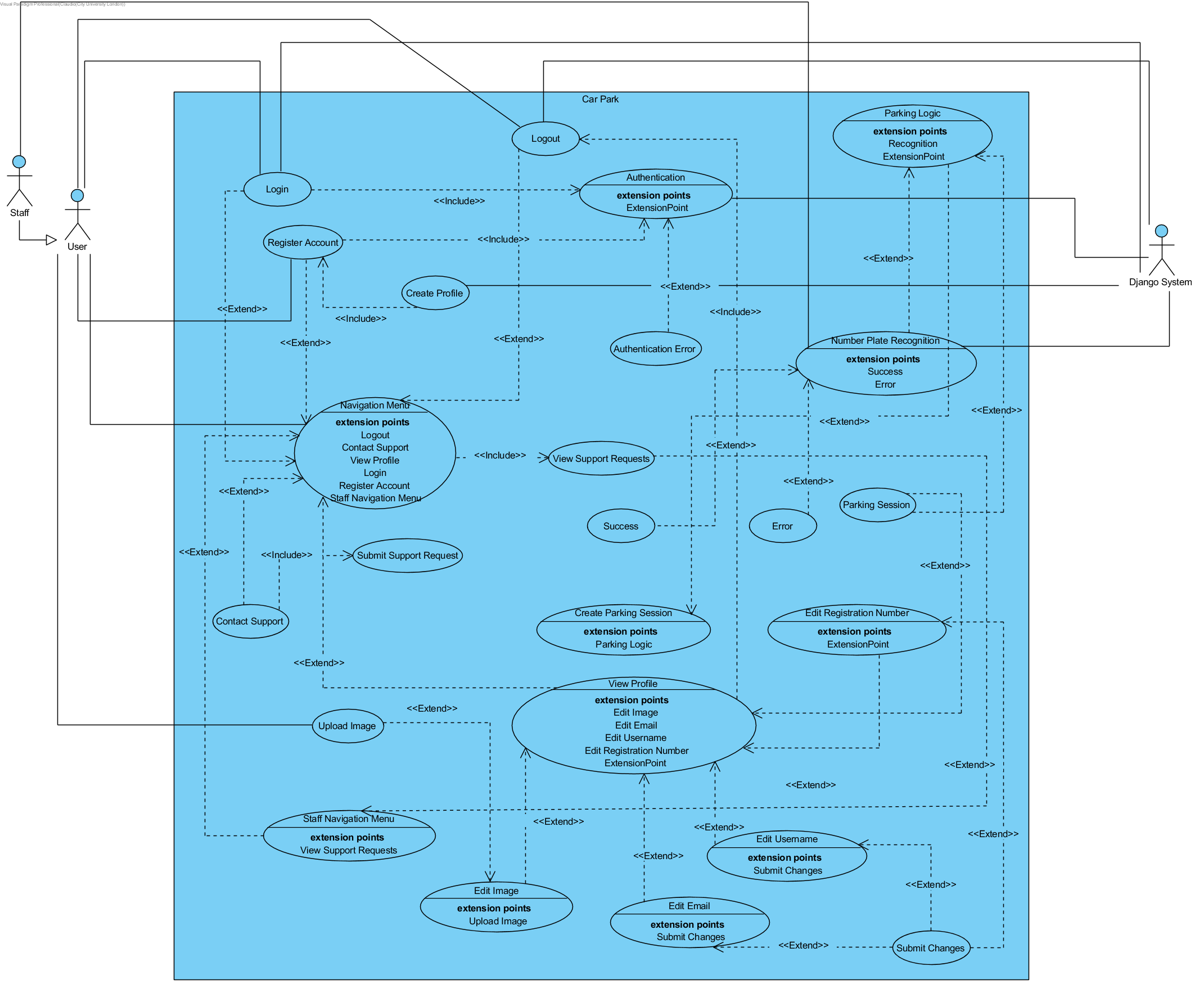
|  |  |
| --- | --- |
| Use Case: Navigate Site | |
| ID: | UC11 – Navigate Site |
| Description: | The user is able to access the intended pages using the navigation buttons |
| Primary Actors: | User |
| Secondary Actors: | N/A |
| Preconditions: | User is using a supported browser to visit the website |
| Main Flow: | The use case begins when the user lands on any page of the website  1) The user is not logged in  1.1) The user will have access to the Home, About, Contact Us, Register and Login pages  2) The user is logged in  2.1) The user will have access to the Home, About, Contact Us, Profile and Logout pages  3) The user is logged in as Staff  3.1) The user will have access to the Home, About, Contact Us, Profile, Logout and Contact Requests pages |
| Post Conditions: | N/A |
| Alternate Flows: | N/A |

## Use case prioritization

|  |  |
| --- | --- |
| Requirement ID | Priority |
| UC01 | High |
| UC02 | High |
| UC03 | High |
| UC04 | Medium |
| UC05 | Medium |
| UC06 | Low |
| UC07 | Low |
| UC08 | High |
| UC09 | High |
| UC10 | Medium |
| UC11 | Medium |

# Appendix C – Design Documentation

## C.1 Use Case Diagram



## C.2 ERD (Entity Relationship Diagram)



## C.3 Wireframes

### C.3.1 Home Page

### C.3.2 Login Page

### C.3.3 Register Page

### C.3.4 Contact Us Page

### C.3.5 Profile Page

### C.3.6 Help Requests Page

### C3.7 Logout Confirmation Page

### C.3.8 About Page

# Appendix D – Test Results

|  |  |
| --- | --- |
| **Use Case ID:** 1.1 | **Use Case Name:** Login |
| **Test Number:** 1 | |
| **Objective:** Test Login Function | |
| **Set Up:** User with valid credentials logs in | |
| **Expected Results:**  1) User is successfully logged in  2) User is redirected to home page | |
| **Test:**  1) The user navigates to the login page  2) The user inputs the login data and submits it to the system | |
| **Test Record:** Expected Results | |
| **Date:** 27 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 1.2 | **Use Case Name:** Incorrect Login |
| **Test Number:** 2 | |
| **Objective:** Test Login Function for erroneous input | |
| **Set Up:** User with invalid credentials tries to logs in | |
| **Expected Results:**  1) User is showed an error message  2) Page is reloaded | |
| **Test:**  1) The user navigates to the login page  2) The user inputs the invalid login data and submits it to the system | |
| **Test Record:** Expected Results | |
| **Date:** 27 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 1.3 | **Use Case Name:** Logout |
| **Test Number:** 3 | |
| **Objective:** Test Logout Function | |
| **Set Up:** User is successfully logged into their account | |
| **Expected Results:**  1) User is logged out of their account  2) User is showed a logout confirmation page | |
| **Test:**  1) The user presses the logout button at the top of the page | |
| **Test Record:** Expected Results | |
| **Date:** 27 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 1.4 | **Use Case Name:** Register |
| **Test Number:** 4 | |
| **Objective:** Test Register Function | |
| **Set Up:** User is not logged in | |
| **Expected Results:**  1) User Data is saved into the system  2) User is able to log into the account they created  3) User is redirected to the login page | |
| **Test:**  1) The user fills the registration form with valid data  2) The user submits the form | |
| **Test Record:** Expected Results | |
| **Date:** 27 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 1.5 | **Use Case Name:** Register Invalid Input |
| **Test Number:** 5 | |
| **Objective:** Test Register Function with invalid input | |
| **Set Up:** User is not logged in | |
| **Expected Results:**  1) User is prompted an error message  2) Data from the Registration form is not saved  3) Page is reloaded | |
| **Test:**  1) The user fills the registration form with invalid data  2) The user submits the form | |
| **Test Record:** Expected Results | |
| **Date:** 27 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 1.6 | **Use Case Name:** Edit Username |
| **Test Number:** 6 | |
| **Objective:** Test Edit Username Function | |
| **Set Up:** User is logged in | |
| **Expected Results:**  1) User is shown a success message  2) Data from the username form is saved  3) Page is reloaded | |
| **Test:**  1) The user fills the username field with vaild data  2) The user submits the form | |
| **Test Record:** Expected Results | |
| **Date:** 27 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 1.7 | **Use Case Name:** Edit Username Invalid |
| **Test Number:** 7 | |
| **Objective:** Test Edit Username Function with invalid input | |
| **Set Up:** User is logged in | |
| **Expected Results:**  1) User is shown an error message  2) Data from the Username form is not saved  3) Page is reloaded | |
| **Test:**  1) The user edits the username form with invalid data  2) The user submits the form | |
| **Test Record:** Expected Results | |
| **Date:** 27 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 1.8 | **Use Case Name:** Edit Email |
| **Test Number:** 8 | |
| **Objective:** Test Edit Email Function | |
| **Set Up:** User is logged in | |
| **Expected Results:**  1) User is shown a success message  2) Data from the email field is saved  3) Page is reloaded | |
| **Test:**  1) The user fills the email field with valid data  2) The user submits the form | |
| **Test Record:** Expected Results | |
| **Date:** 27 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 1.9 | **Use Case Name:** Edit Email Invalid |
| **Test Number: 9** | |
| **Objective:** Test Edit Email Function with invalid input | |
| **Set Up:** User is logged in | |
| **Expected Results:**  1) User is shown an error message  2) Data from the email form is not saved  3) Page is reloaded | |
| **Test:**  1) The user edits the email form with invalid data  2) The user submits the form | |
| **Test Record:** Expected Results | |
| **Date:** 27 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 1.10 | **Use Case Name:** Edit Image |
| **Test Number:** 10 | |
| **Objective:** Test Edit Email Function | |
| **Set Up:** User is logged in | |
| **Expected Results:**  1) User is shown a success message  2) File from the Image field is saved and downsized  3) Page is reloaded | |
| **Test:**  1) The user presses the Choose file button  2) The user uploads an image  3) The user submits the form | |
| **Test Record:** Expected Results | |
| **Date:** 27 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 2.1 | **Use Case Name:** Contact Support |
| **Test Number:** 11 | |
| **Objective:** Test Contact Support Function | |
| **Set Up:** User is logged in | |
| **Expected Results:**  1) User is shown a success message  2) Data from the Support from is saved  3) Support request becomes available on the Requests section accessible by the staff | |
| **Test:**  1) The user presses the Contact us button  2) The user fills the form with their request  3) The user submits the form | |
| **Test Record:** Expected Results | |
| **Date:** 29 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

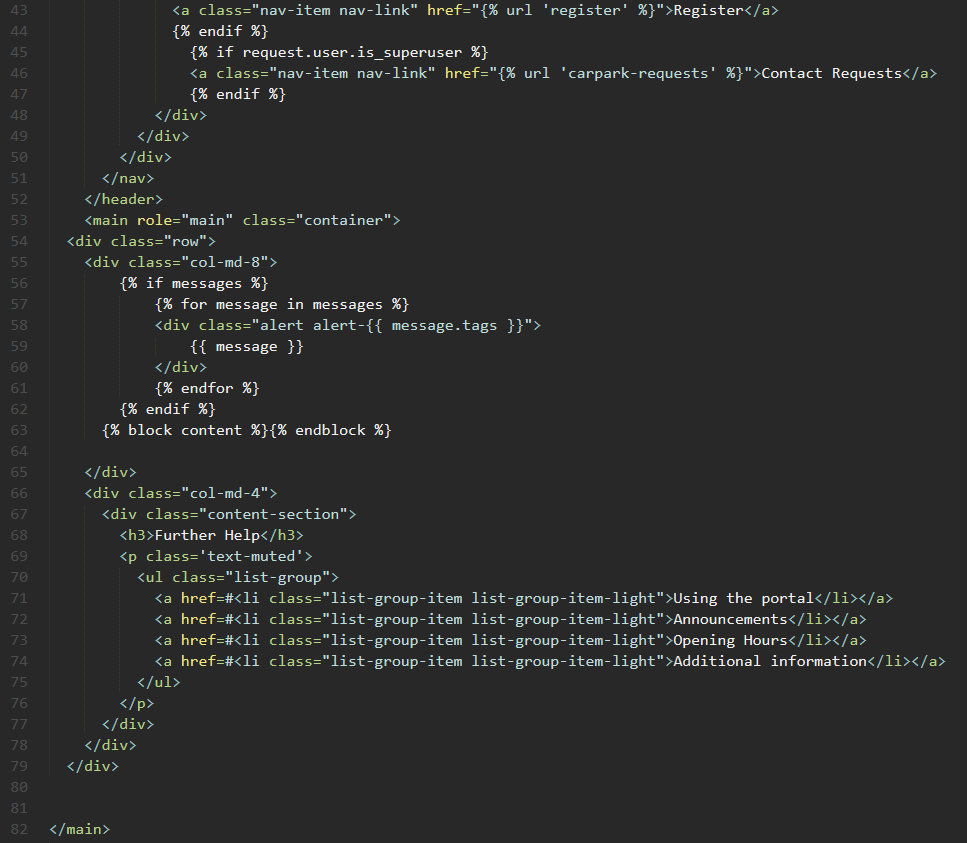
|  |  |
| --- | --- |
| **Use Case ID:** 2.2 | **Use Case Name:** Navigation Bar |
| **Test Number:** 12 | |
| **Objective:** Test navigation bar | |
| **Set Up:** User is using a supported browser | |
| **Expected Results:**  1) Navigation bar items redirect the user to the intended route | |
| **Test:**  1) The user presses the navigation bar buttons | |
| **Test Record:** Expected Results | |
| **Date:** 29 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

|  |  |
| --- | --- |
| **Use Case ID:** 2.4 | **Use Case Name:** Help Requests Page |
| **Test Number:** 13 | |
| **Objective:** Test Help Request Function | |
| **Set Up:** StaffUser is logged into the system | |
| **Expected Results:**  1) User is able to see and select different help requests | |
| **Test:**  1) The user presses Contact Requests button  2) They select a request to verify | |
| **Test Record:** Expected Results | |
| **Date:** 29 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

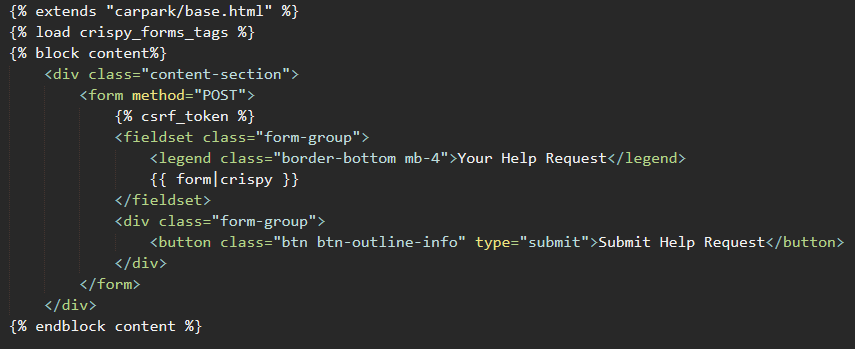
|  |  |
| --- | --- |
| **Use Case ID:** 2.5 | **Use Case Name:** Number Plate Recognition |
| **Test Number:** 14 | |
| **Objective:** Test Number plate recognition Function | |
| **Set Up:** StaffUser is logged into the system | |
| **Expected Results:**  1) Staff user runs the check on an image and the expected text is recognized  2) Accordingly, user session is started or ended | |
| **Test:**  1) The staff runs the parking command | |
| **Test Record:** Expected Results | |
| **Date:** 29 April 2021 | **Tester:** Claudiu Andrei |
| **Result:** Passed | |

# Appendix E – Implementation Document

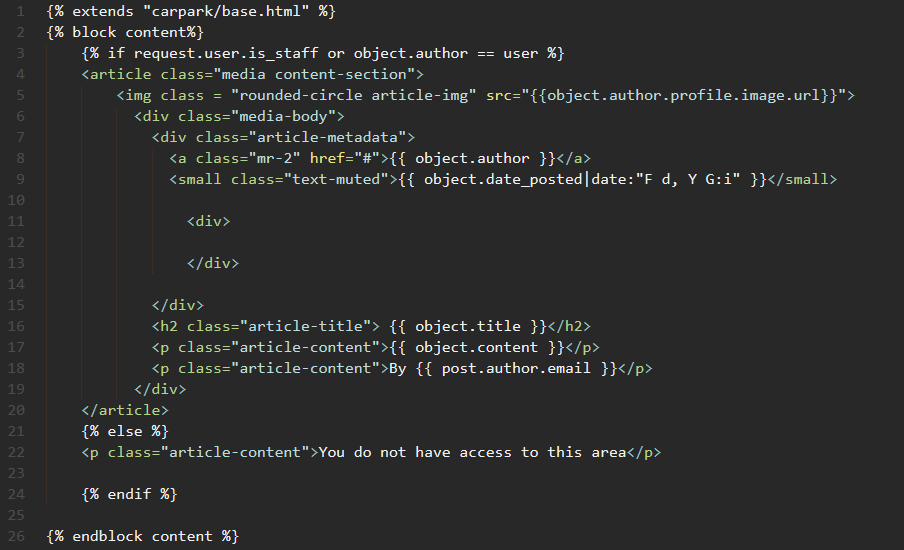
## Base Html file



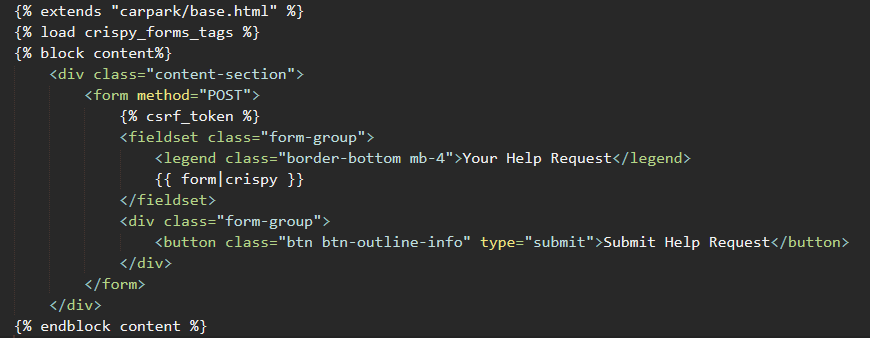
## Contact Html file



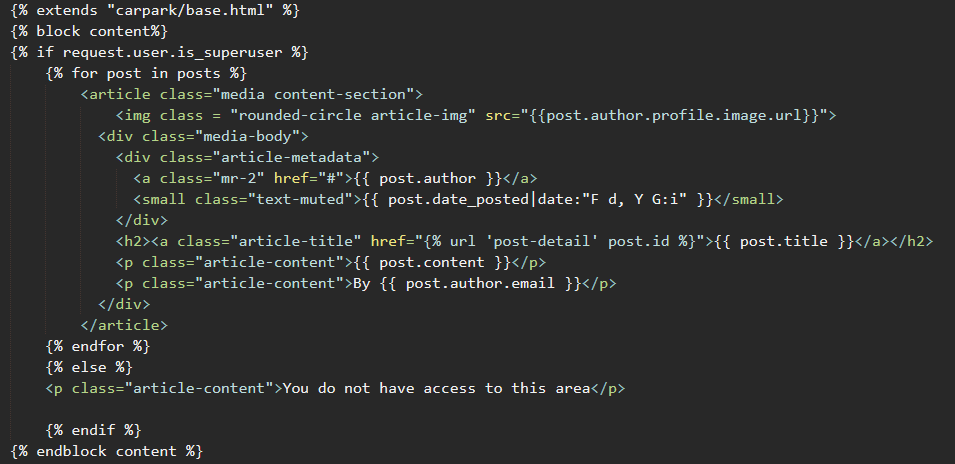
## Post Detail Html file



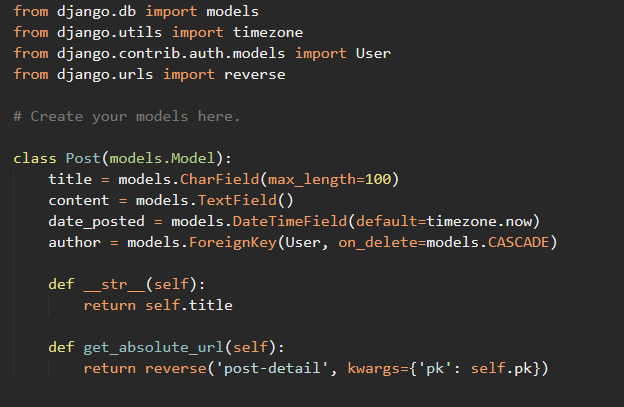
## Post Form html file



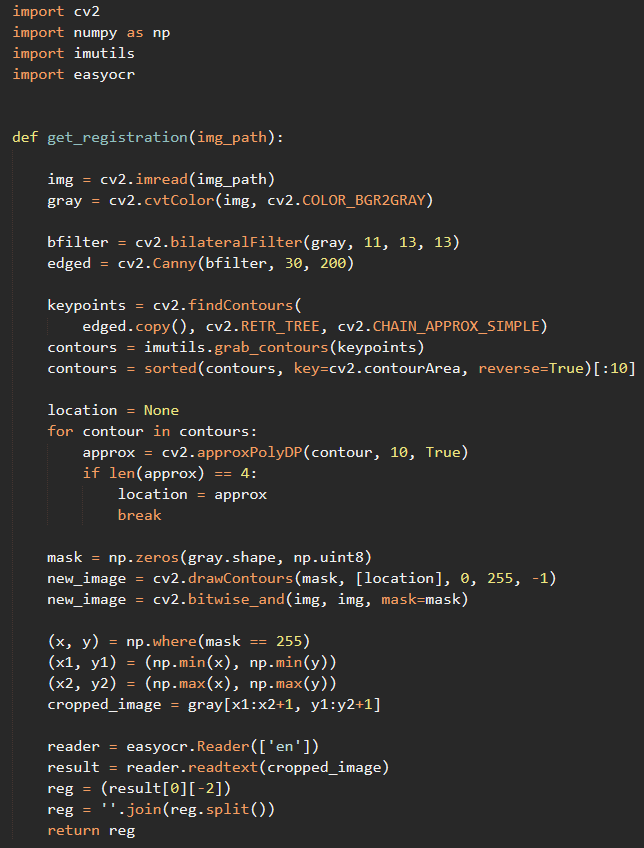
## Requests page html file



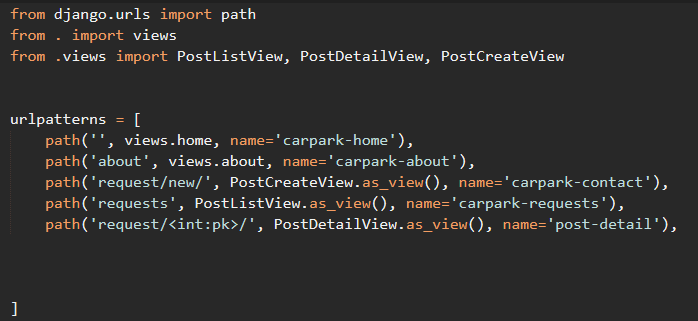
## Help Request Model



## Registration Recognition



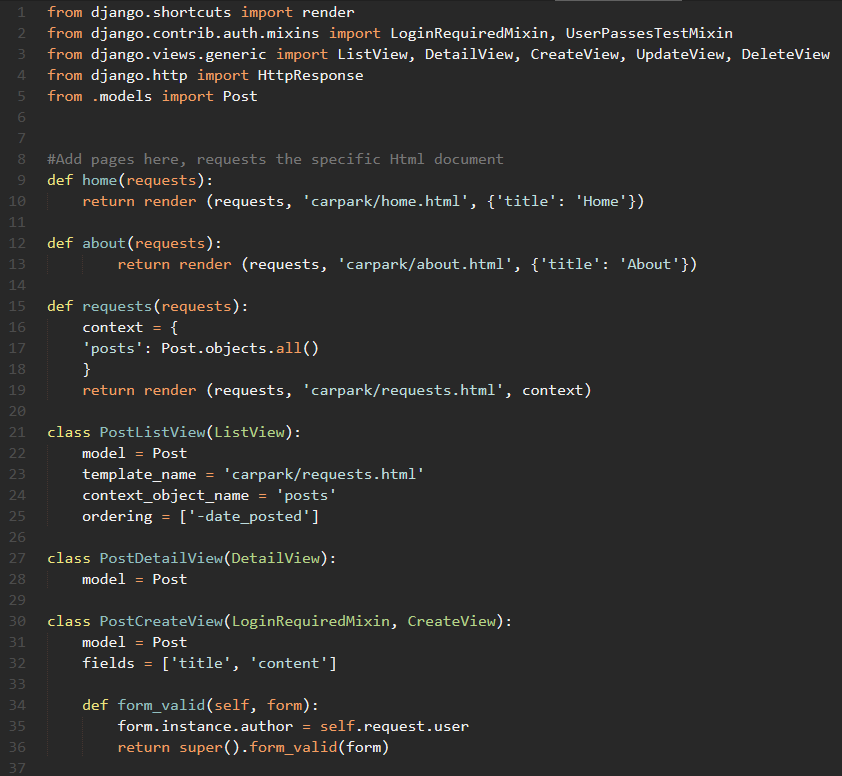
## Carpark app routes



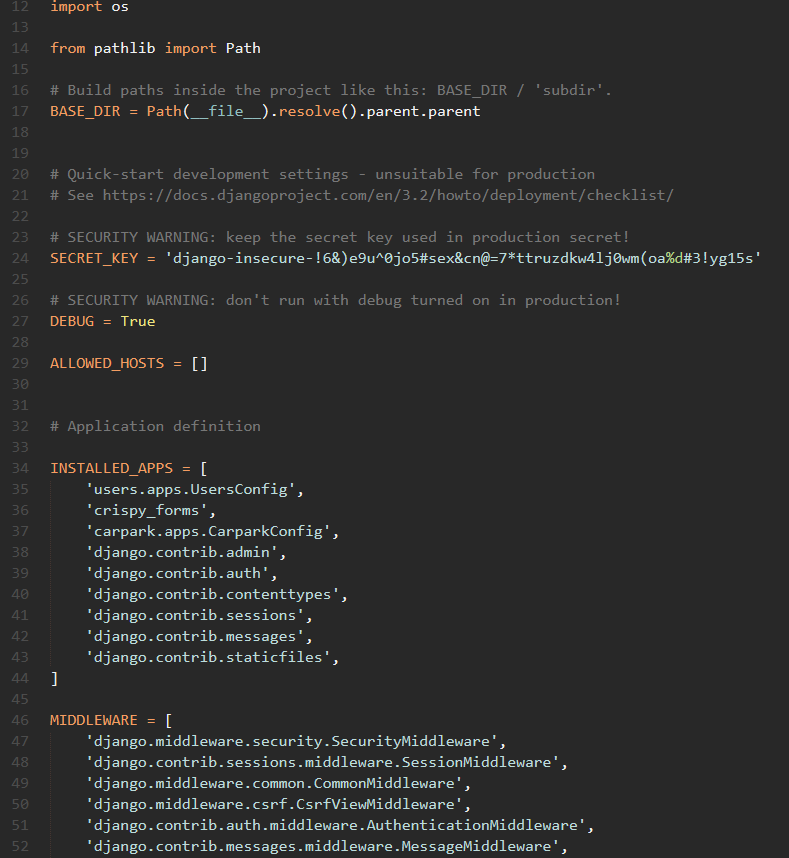
## Parking Logic file



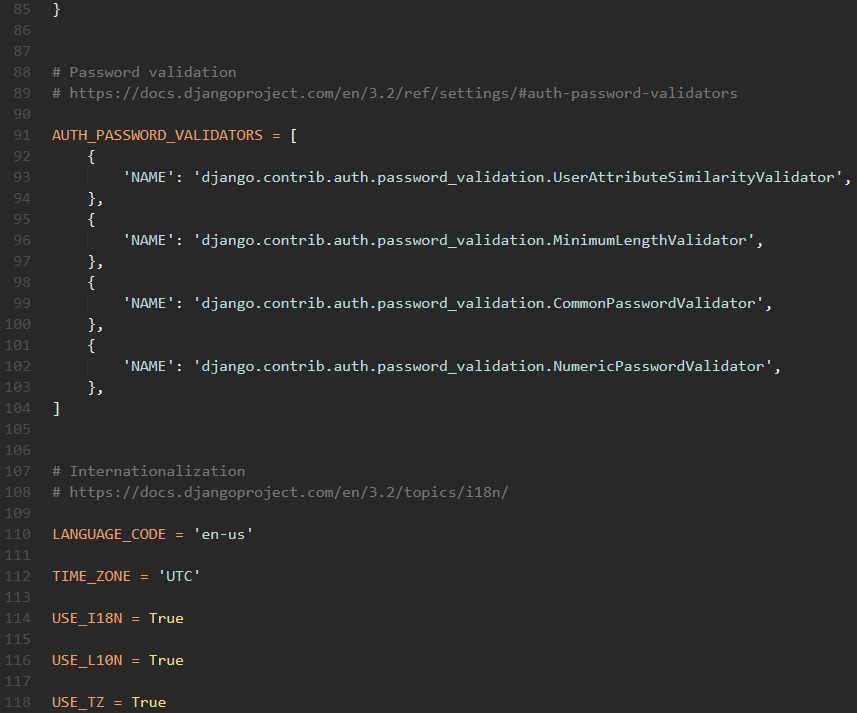
## Carpark app views

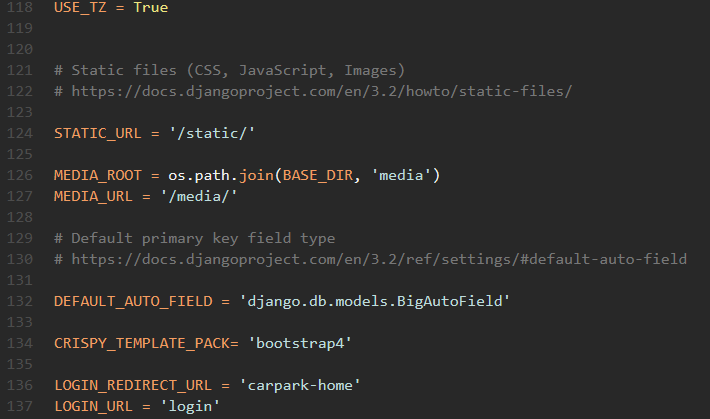


## Django Project Settings

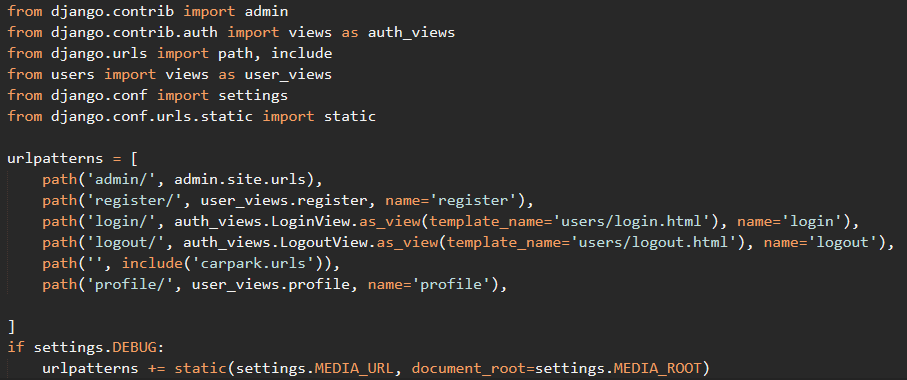






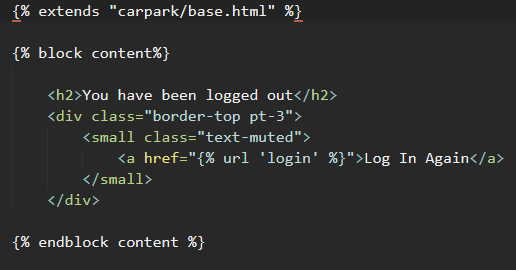


## Django Project Routing

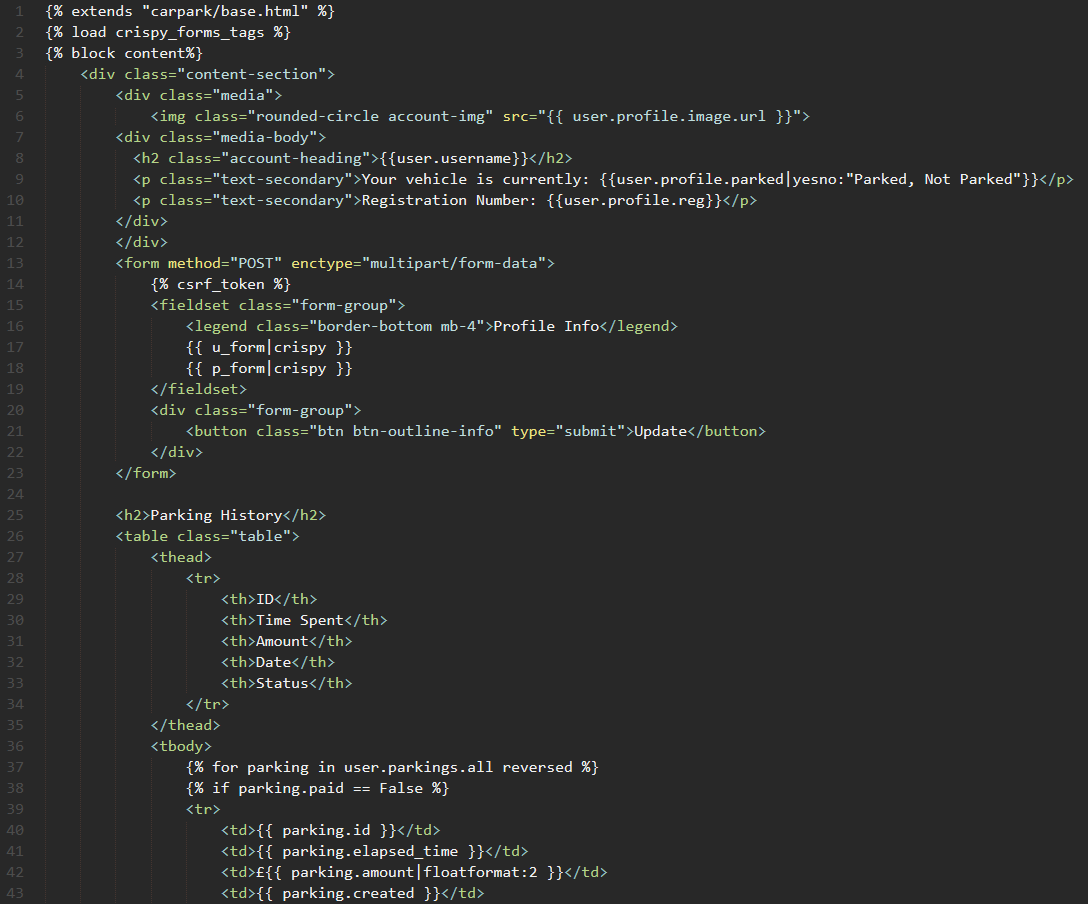


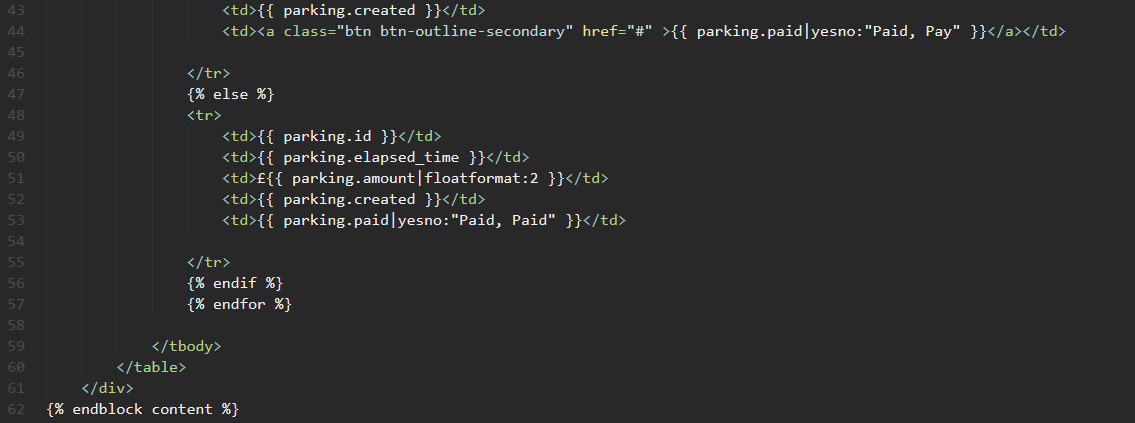
## Login page html file

## Logout page html file

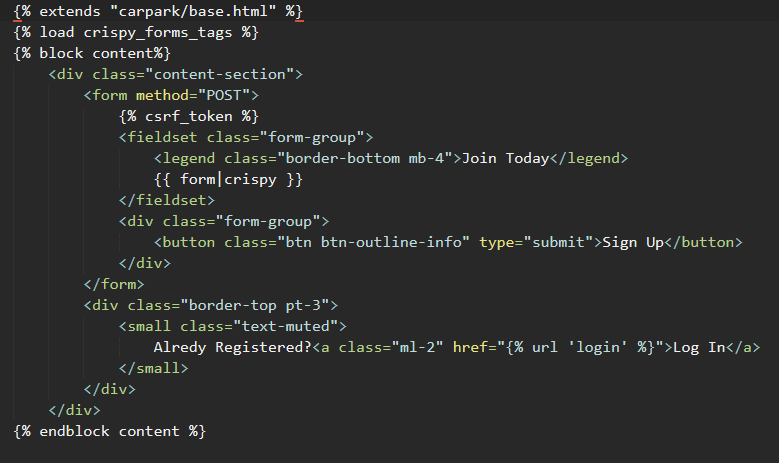


## Profile page html file

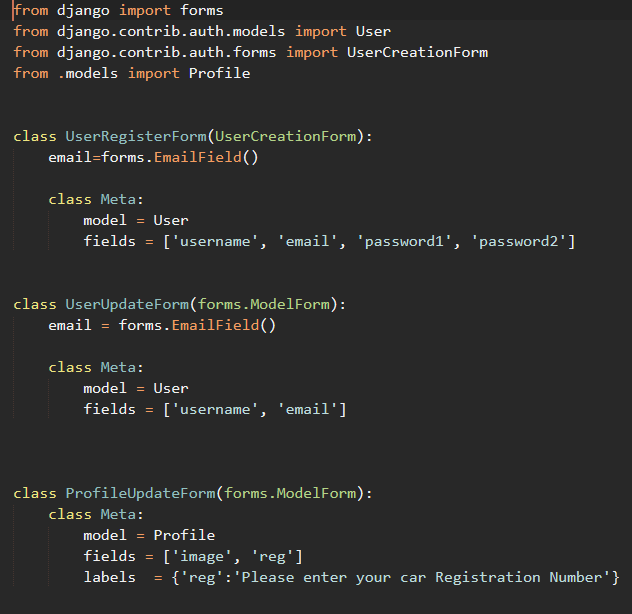




## Register page html file



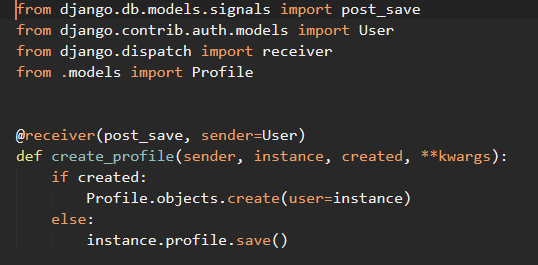
## Forms file



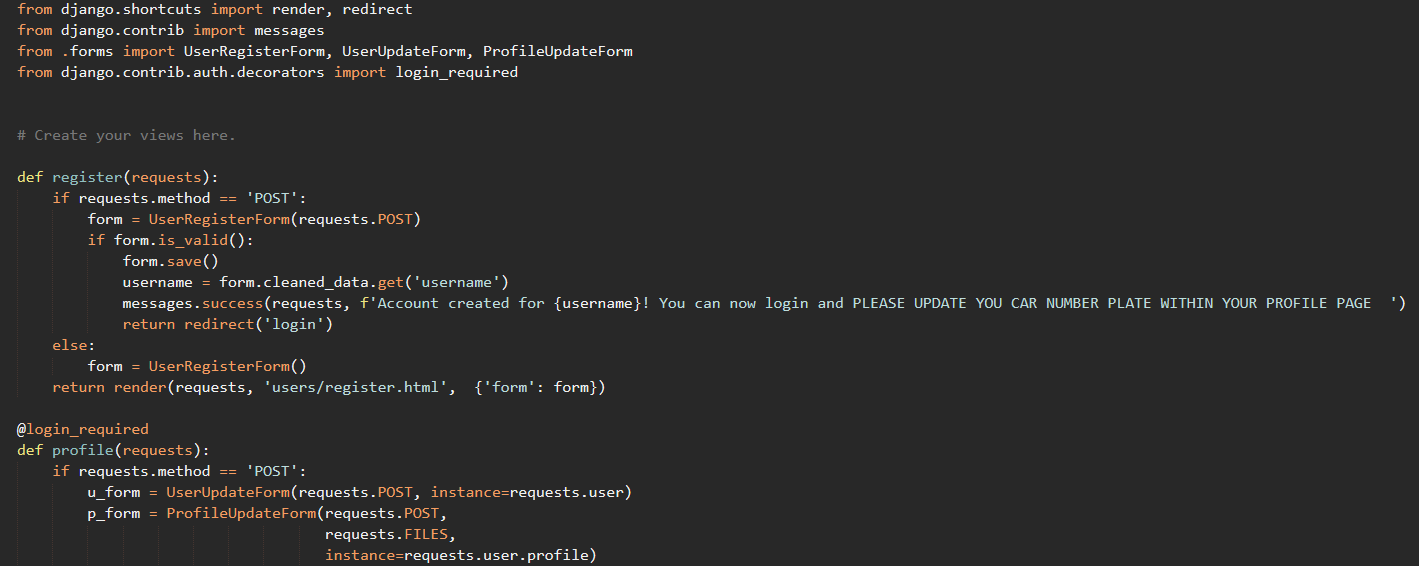
## Models of the parking logic

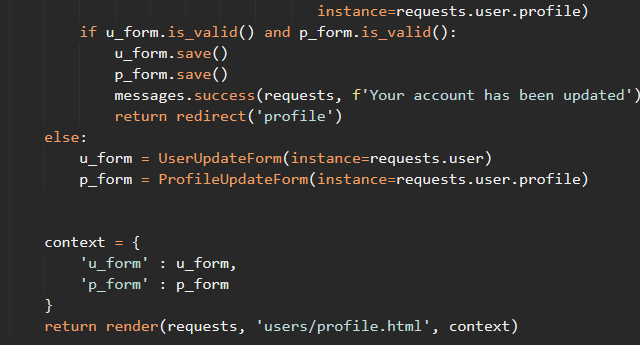


## Signals file



## Views file





# Appendix F – Running the solution

## Method used by the author

In order to run this solution, some requirements on the machine need to be installed.

1. Django
2. Anaconda
3. OpenCV
4. Imutils
5. EasyOCR
6. Matplotlib
7. Pillow
8. Python

After these are installed in order the following steps are required:

1. Extract the files of the project in a folder
2. In windows using the Anaconda command prompt navigate to the folder
3. Run the “python manage.py runserver” command to start the server
4. In a browser navigate to the localhost url to be able to access the site
5. To run the number plate recognition function, open a new command prompt and navigate to the project folder
6. Run the command “makepark” to begin the recognition process
7. Run the command again with different inputs in the “utils.py” file for different tests