AUTOMATIC PARKING CHARGE SYSTEM

WHITE PAPER

BSc Computing (Hons) 3rd Year Project

CMP311: PROFESSIONAL PROJECT DEVELOPMENT & DELIVERY

CREATED BY ERROR 418
Michael McMillan
Jamie Beecroft
Nicole Notarangelo

1 Abstract

This paper describes the process behind the development of a parking system that scans the registration plates of vehicles when they enter and leave a car park, and automatically charges customers based on the length of their stay. This system was requested by the client Smart Parking Ltd, represented by Dr Gavin Hales. The client desires a modern and elegant parking system solution that will be far more convenient to customers than solutions currently available through competitors, and that will allow changes to parking prices to be quickly and easily implemented throughout the entire system.

To achieve this, a system was created involving a website, client-server setup, a license-plate recognising API (known as OpenALPR; Automatic License Plate Recognition), and a Raspberry Pi device. The website is what a customer will use to create an account, register vehicles to that account, and purchase season permits. The server contains details of registered customers and vehicles owned by the customers. The Raspberry Pi controls a camera to capture images of license plates and interacts with the API that identifies the license number in the images. Once the API has identified a valid plate, the device sends this plate to the server database via a client API. If this plate is registered to a customer, the device would allow entry to the car park. If it is not registered, entry is denied. The API also keeps track of whether the given vehicle is entering or leaving, and if it is leaving the customer who owns it is charged accordingly.

The result of this project was a robust system with a secure and functioning front-end, and a mostly-accurate license plate detector. It adequately demonstrates all the core functionality requested and forms a solid basis for a viable real-world car park technology. It would need to be scaled up appropriately and refactored for commercial use, but in its current form it successfully carries out all necessary tasks on a minor scale.

2 CONTENTS

3 Introduction

3.1 Background

This white paper is for the Automatic Parking Charge System project to be created for the client, Dr. Gavin Hales, representative of Smart Parking Ltd.

In urban spaces, the flow and control of vehicle traffic is a major concern, especially in large city centres like Dundee. This includes providing adequate facilities for commuters to park their vehicles once they arrive. However, many city car parks suffer from inefficient systems and inadequate flow of incoming and outgoing traffic. Most of the core problems of urban parking are caused by miscalculation of required space and overall ineffective use of land (Ibrahim, 2017). Therefore, they are not easily fixed as the issues stem from the foundation level of an affected parking area. Even so, parking companies still have an obligation to reduce the negative impact of poor initial planning on their customers by streamlining and simplifying the overall process of using the car parks.

Dundee in particular has three major car parks, each with their own flaws and difficulties of use. As of the writing of this paper, it takes an excessive amount of time for most commuters to enter and exit these car parks. Furthermore, these car parks offer only a single traditional method of tracking and charging customers as they get in and out; by giving them a ticket when they arrive and then checking it when they leave, and thereby billing them for parking based on the time taken between the ticket being dispensed and then being returned. If a customer were to lose their ticket or have it damaged or stolen, then they would have to spend even more time to get this issue resolved and then consequently have to pay the full ticket price, even if the length of their stay would have cost less than this price.

Smart Parking Ltd want to lead the way in modern parking solutions by getting rid of these paper tickets in favour of a fully automated system. Instead of receiving a ticket at the barrier and then subsequently paying at a machine on the way back out, the system will automatically read the vehicle number plate upon arrival and then raise the barrier if this number plate is registered with a customer. The customers will then be sent a bill charging them for their time spent in the car park.

This new system proposed by Smart Parking Ltd will greatly reduce the amount of time it takes to make use of a car park. It will reduce frustration in customers who lose or damage their tickets and ensure that the bills they receive for parking are accurate for the length of their stay by only charging once their vehicle has left the car park. Smart Parking Ltd will then gain a fantastic reputation of having easy-to-use and hassle-free car parks. Furthermore, Smart Parking Ltd will be taking a step forward in becoming eco-friendlier by completely cutting out paper waste.

3.2 Aim

The aim of the project is to develop an automatic parking charge system whereby a customer can enter and exit the car parks without having to use a machine and to have their bills sent to them automatically so that Smart Parking Ltd can:

1. Increase their reputation.
2. Gain a competitive edge over rival parking companies in Dundee.
3. Gain more customers.
4. Increase customer satisfaction.
5. Eliminate the need for paper tickets, which will reduce environmental impact.

To meet these aims set out by Smart Parking Ltd, the development team gathered requirements in the planning phase to create a project proposal that clearly set out all the features that Smart Parking Ltd would like in their final working product. It had been decided that completing the following objectives will allow the development team to meet the aims of the client, Smart Parking Ltd:

1. Create a customer control panel where customers can register vehicles, purchase season permits, and pay their bills.
2. Create an administrator control panel where the client can have full control over the pricing of their car parks.
3. Offer an incentive to customers by way of season permits that allow the customer with full access to the car parks for the given month without having to pay bills for each use.
4. Allow registered customers access to the car parks, with the barriers opening automatically after reading the number plate and the customers sent a bill after they have left the car park.

4 PROCEDURE

4.1 OVERVIEW OF PROCEDURE

To meet the aims and objectives set out by the client, the project has been split into the following parts:

1. Web application
2. Client-server
3. API
4. Physical system
5. Project management

Each of these parts were relegated to handling an aspect of the overall application, so that bringing them all together would result in a robust and functioning system for tracking entries and exits into a parking area. Partitioning the overall project in this way also helped to reduce time spent fixing errors, as it was simple to see which part of the whole system was malfunctioning and then fix it accordingly.

4.2 WEB APPLICATION

The website was developed in PHP using the Model-View-Controller application architecture, allowing for a good separation of concerns because the logic, front-end, and database are not fully dependent on each other, only an interface.

The database was already designed and fully normalised in the design stage. It was then implemented through the phpMyAdmin database wizard on a MariaDB engine.

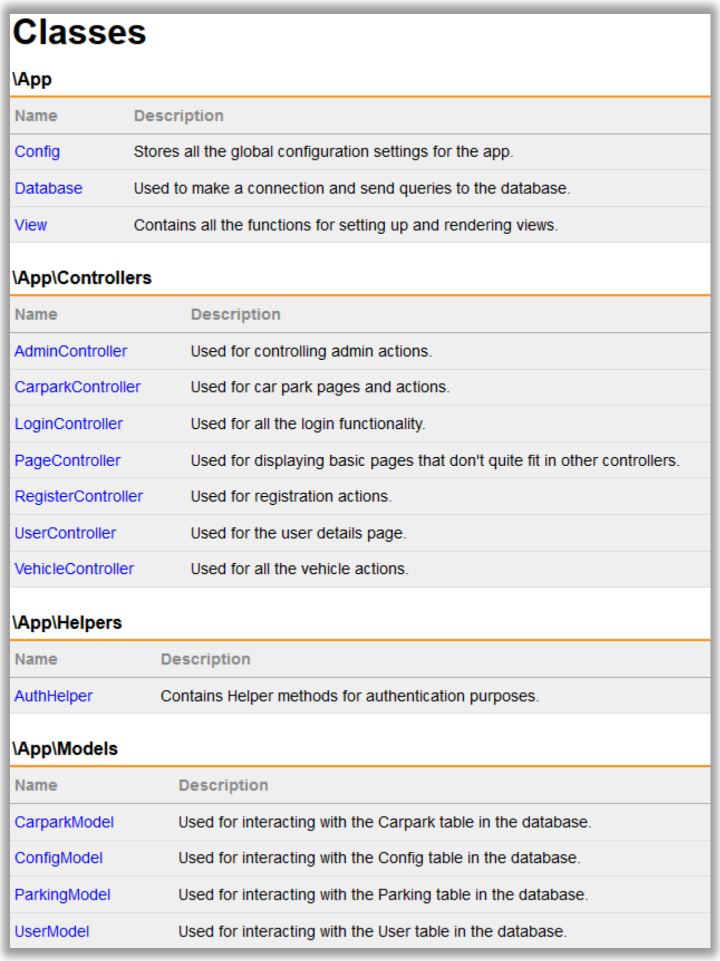
In the website source code, model classes were created for each of the tables in the database. These classes contain the functions to communicate with the database and carry out actions such as creating, reading, updating, and deleting records. This has provided good abstraction over the database so that it can be completely separate from the application logic – useful in the event that the database engine has to be changed.

View files were created for each of the required pages. At first, the views only contained basic HTML for testing purposes, so the developer could focus on the functionality, but were later updated with a new design at a later date using the Bootstrap CSS library (Bootstrap team, 2019). The view and database were connected together by creating the controller classes to handle interactions with the database API, which would communicate with the Raspberry Pi device.

Composer (Adermann & Boggiano, 2019) was installed so that autoloading could be used. Autoloading is more or less a requirement for setting up an object-oriented PHP project because it automatically loads in all the classes without the developer having to specify them using the include function. Composer was also used as the package manager, sourcing the required dependencies for the application.

One such dependency is the Stripe payments service (stripe, 2019) used to develop the payment system. This was implemented using the PHP library that can be found on GitHub (stripe, 2019). To use the Stripe API, an account was registered on the website so that an API key could be obtained.

Another dependency obtained by utilising Composer is phpDox (theseer, 2019). phpDox generates HTML documentation by reading the files to gather classes, methods, and descriptions. This requires that the code is commented in a standard way, detailing each method and each method parameter. This has made it useful to the current development team and any future teams undertaking new versions of the project because they can look at the documentation to get a good understanding of how the code works.



Classes

\App

Name	Description
Config	Stores all the global configuration settings for the app.
Database	Used to make a connection and send queries to the database.
View	Contains all the functions for setting up and rendering views.

\App\Controllers

Name	Description
AdminController	Used for controlling admin actions.
CarparkController	Used for car park pages and actions.
LoginController	Used for all the login functionality.
PageController	Used for displaying basic pages that don't quite fit in other controllers.
RegisterController	Used for registration actions.
UserController	Used for the user details page.
VehicleController	Used for all the vehicle actions.

\App\Helpers

Name	Description
AuthHelper	Contains Helper methods for authentication purposes.

\App\Models

Name	Description
CarparkModel	Used for interacting with the Carpark table in the database.
ConfigModel	Used for interacting with the Config table in the database.
ParkingModel	Used for interacting with the Parking table in the database.
UserModel	Used for interacting with the User table in the database.

Figure 1: The PHP documentation generated by phpDox.

4.2.1 Security

There are people out there who are determined to crack into insecure websites and compromise databases for personal gain. It is important to the development team that all recommended security measures have been applied to the system. OWASP's top 10 security risks (OWASP, 2017) have been reviewed by the developers and will be used as a guideline for the team to understand what needs to be done to ensure a fully protected system that cannot be compromised so Smart Parking Ltd's reputation and income remain intact.

Password Encryption

Users require a password in order to log in and make use of the web application. As a result of this, the passwords have been encrypted before being stored in the database to add an extra line of defence in the event that an attacker compromises the database. This was implemented using the built-in PHP function using the BCrypt algorithm which is good for slowing down attackers who are carrying out a brute force attack. A salt was also added to the encrypted passwords to make it more difficult for attackers to crack the password (Arias, 2018).

Data Validation and Integrity

The website makes use of HTML forms with JavaScript validation for things like registration and login, making the site potentially vulnerable to cross-site scripting (XSS) attacks. Therefore, all user input is escaped before being processed by the server in order to prevent XSS attacks.

The web application was also potentially vulnerable to SQL injection attacks because it made use of the previously mentioned HTML forms but also the GET parameters in the URL can be altered by an attacker in attempt to execute queries on the database. SQL injection attacks have thus been prevented by using prepared SQL statements with the PDO PHP module.

Access Control

Access control has been implemented on the web application by way of user accounts with roles indicating whether the user is a customer or an admin. Only users who have the admin role can view the admin panel. This was implemented with a check in the server-side code on the web application for certain pages to check if the user is a guest, logged in customer, or logged in admin.

4.3 CLIENT-SERVER

It was originally planned to have a client-server architecture in which the Raspberry Pi had a Python TCP client and the central server had a .NET TCP server. The .NET TCP server was developed following the Microsoft guide (Microsoft, 2017) with the OpenALPR Cloud API easily integrated by following the official documentation (OpenALPR Technology Inc, 2017).

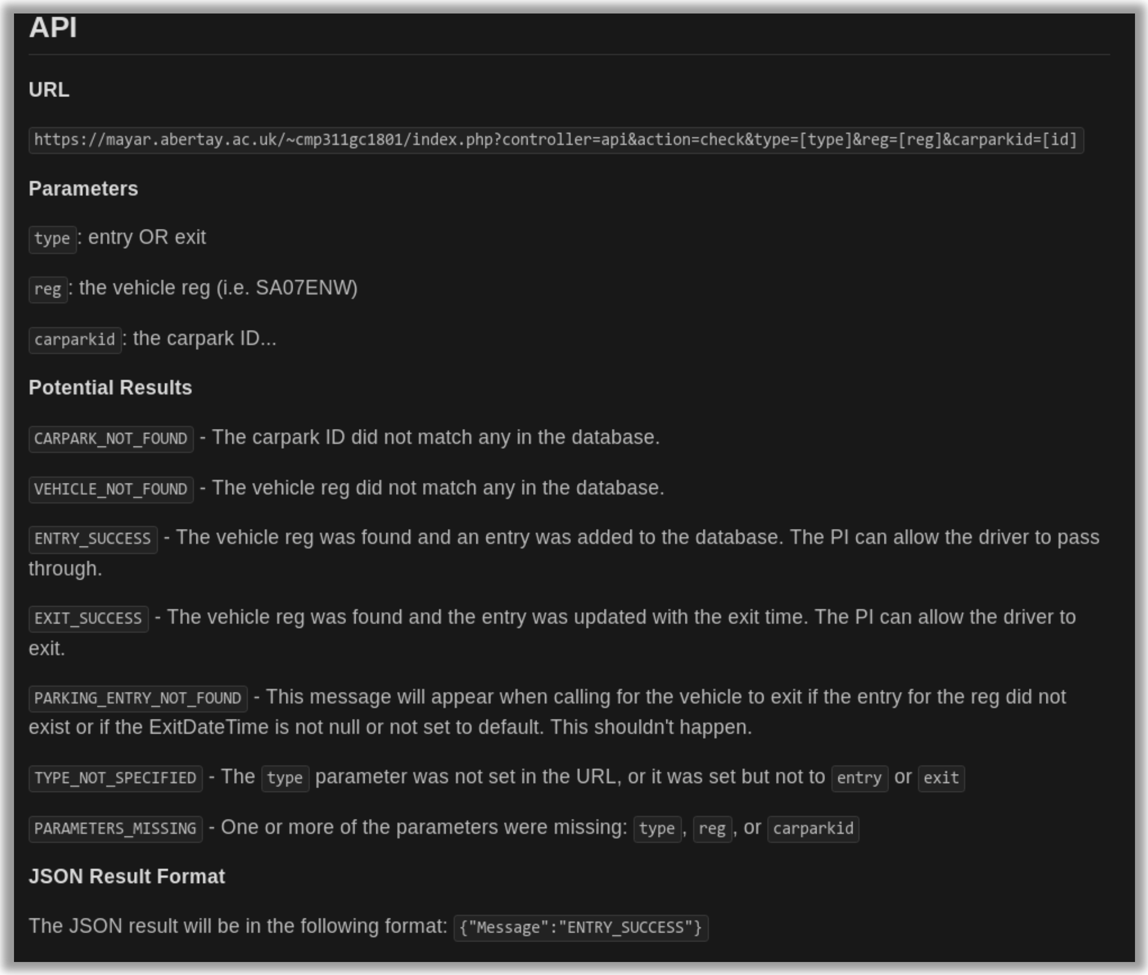
This worked very successfully, with the TCP client sending a picture from the Raspberry Pi to the TCP central server, and having the server read the number plate from the picture and send back a result indicating whether the vehicle should be let in.

The TCP server worked effectively when only connecting to one client, but it did not support working with multiple clients simultaneously, which puts a real constraint on how many customers can enter or exit the car parks at any on time. Instead, an asynchronous version was developed. Even with the asynchronous server, there were still issues in sending an image from one device on the network to another because sometimes the image got corrupted.

However, it was discovered at this stage that the system would become considerably more reliable by having the Raspberry Pi perform the number plate recognition, eliminating the need for sending an image over the network, which could cause major downtime in a production environment with insufficient network speeds. Instead, only the number plate string needs to be sent over the network. Doing it this way means there was no longer any need for a client-server set-up like this. Instead, the Raspberry Pi client now communicates with a PHP API that has been developed alongside the web application to determine whether the vehicle is to be let in or not.

4.4 API

As a result of getting rid of the TCP server, an API had to be designed and developed for the Raspberry Pi to communicate with. The design of the API can be seen in the following screenshot. When a vehicle successfully enters the car park (as confirmed by the API), a new record is added into the database. When a vehicle successfully exits the car park, the record is updated with the exit date and time and a new invoice is generated via Stripe and sent to the customer in an email, but only if they do not currently possess a season permit for the vehicle. The customer will then be able to click the link in the email to make the payment for their time in the car park.



API

URL

`https://mayar.abertay.ac.uk/~cmp311gc1801/index.php?controller=api&action=check&type=[type]&reg=[reg]&carparkid=[id]`

Parameters

`type`: entry OR exit

`reg`: the vehicle reg (i.e. SA07ENW)

`carparkid`: the carpark ID...

Potential Results

`CARPARK_NOT_FOUND` - The carpark ID did not match any in the database.

`VEHICLE_NOT_FOUND` - The vehicle reg did not match any in the database.

`ENTRY_SUCCESS` - The vehicle reg was found and an entry was added to the database. The PI can allow the driver to pass through.

`EXIT_SUCCESS` - The vehicle reg was found and the entry was updated with the exit time. The PI can allow the driver to exit.

`PARKING_ENTRY_NOT_FOUND` - This message will appear when calling for the vehicle to exit if the entry for the reg did not exist or if the ExitDateTime is not null or not set to default. This shouldn't happen.

`TYPE_NOT_SPECIFIED` - The `type` parameter was not set in the URL, or it was set but not to `entry` or `exit`

`PARAMETERS_MISSING` - One or more of the parameters were missing: `type`, `reg`, or `carparkid`

JSON Result Format

The JSON result will be in the following format: `{"Message":"ENTRY_SUCCESS"}`

Figure 2: Documentation of the API.

4.5 PHYSICAL SYSTEM

Once the Raspberry Pi was obtained, the SD card was flashed with the Raspbian operating system before installation was carried out after connecting the Pi to a display. The latest version of Python was then installed on the Pi.

The Python code was written for the Raspberry Pi client to consume the previously mentioned PHP API. Once the Python script was activated, the Raspberry Pi camera captured an image whenever it detected a change in the pixels of its view (in other words, when it detected motion).

Originally, the device was to use an external infrared sensor to detect proximity of objects to the camera, but the required tools to safely connect the sensor to the Pi were unavailable and so the motion-activated camera was used as a compromise. This has its benefits because it reduces the hardware cost for each unit.

The picture of the vehicle that was captured by the camera was saved on the device and processed through the OpenALPR API (OpenALPR, 2019) to detect if there were any registration number present. If the registration number was found, then it was extracted and sent to the PHP API link using the aforementioned Python code. The API then returned a result indicating whether to allow or deny access to the car park, which depended on whether the detected registration was present in the database of registered car park user vehicles. If no registration plate was detected in the image, neither API is called, and an error message is printed, indicating that the barrier should not open.

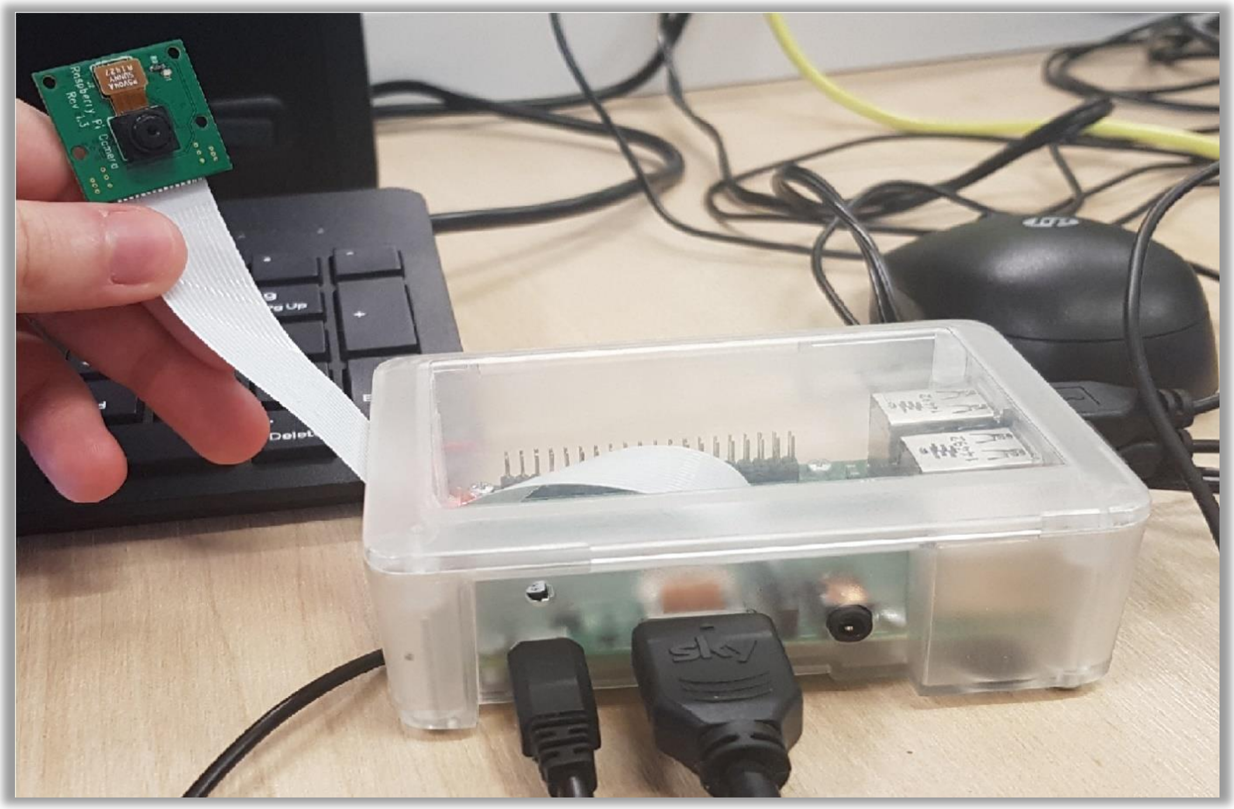


Figure 3: The setup of the physical Raspberry Pi with the camera attached.

4.6 PROJECT MANAGEMENT

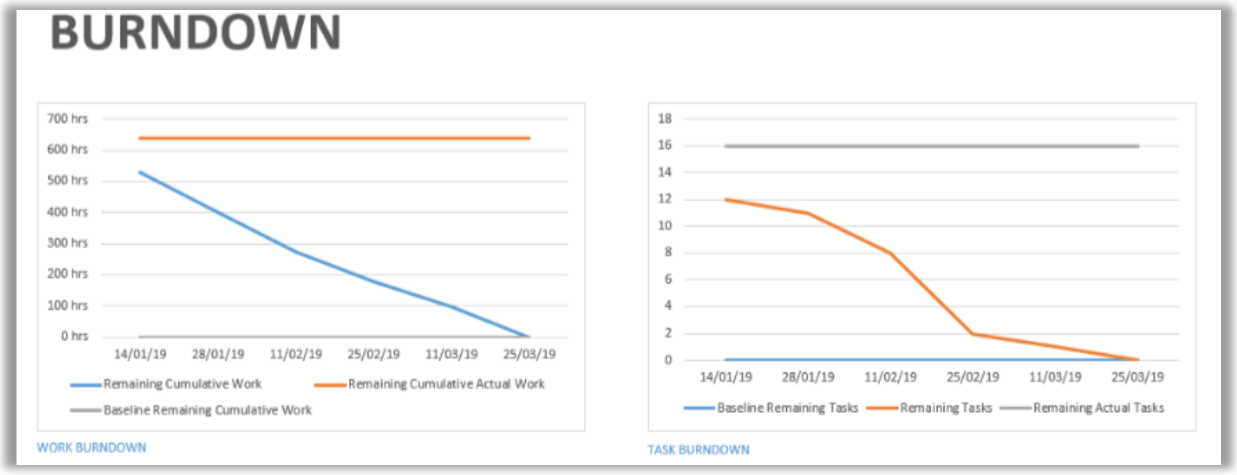


Figure 4: The project burndown chart.

The project manager, Jamie Beecroft, regularly kept the tasks up to date by using Microsoft Project. This is the burndown chart generated by Microsoft Project that indicates the progress of the project accurately.

The original project plan with regards to the time management aspect had to be altered due to members leaving the group and the unreliability of other group members not being able to complete tasks and communicate to the rest of the team in a reasonable amount of time.

Owen Ellis was originally designated with the tasks for the physical system, including design, installation, coding, and testing. After it was discovered that Owen would no longer be continuing on the course, these tasks were re-assigned to Chloe Brown, with Jamie Beecroft taking over some of Chloe’s duties in relation to the web application.

A new member, Calum Newton, was then assigned to the group. This was very useful because the group was now back to four members, meaning that the workload could be spread back out. Therefore, Newton was assigned with the previously mentioned web application tasks that Beecroft took over.

Very shortly after commencement of the course, Calum refused to communicate with the group and due to the likelihood that he would not be continuing, all of his tasks were reallocated again to Jamie who would complete whatever was outstanding.

Chloe then also decided to not continue on the course, with no tasks completed the remainder of the work was then split between Jamie and Michael, however they were already aware that this was likely to happen and had began to complete the tasks prior to this. This had left the group with just two members.

Nicole Notarangelo was then assigned to the group. Nicole took on the physical system tasks, which were the only remaining tasks with the exception of administration work.

These issues were rectified easily thanks to the team's effective project management plan which included developing a risk management plan in the planning phase.

4.6.1 Risk Management

There were nine risks that were identified during the planning phase of the project. This section discusses these risks, how they affected the project during the development phase, and how they were managed to ensure that the project stays on track for a complete and high-quality product to be delivered to Smart Parking Ltd on the planned date.

1. Member absences.
2. External commitments.
3. Travel issues.
4. Miscommunication.
5. Schedule flaws.
6. Requirements change.
7. Failure to meet requirements.
8. External dependencies.
9. Underestimation of development complexity.

These risks were assessed at every meeting to determine whether their impact or likelihood has changed.

In relation to member absences, there were a number of occasions where group members did not attend the meetings with no apologies. It resulted in three members in total leaving the group after lengthy unexplained absences. This did have a big impact on the project because tasks had to be assigned to different members of the group. However, the workload was spread out accordingly and development went forward as planned.

External commitments were a non-issue due to each member being able to flexibly carry out their work for this project around any other commitments that they had.

Travel issues did not really arise, however because all the group members lived out with Dundee, most communication took place electronically in the form of online chat, with Trello being used to keep the project on track.

The remaining members of the development team communicated well with each other and each member understood what was required of them. Keeping in regular contact to keep each group member up to date was an essential part of the process. This worked in all cases apart from those that did not regularly attend meetings.

There were no flaws with the schedule as the development team left an adequate amount of time at the end of the project timeline, allowing for the time to be used to fix any issues that arose.

There were no changes in requirements from the client.

The development believes that the requirements for the project have been met with additional suggestions being implemented with further points that can be worked on in the future should the client request it.

All of the external dependencies are currently working as intended. The OpenALPR system is open-source and has been installed onto the hardware. The rest of the system is managed by the in-house API that was created by the development team.

The project was fairly complex however there were adequate resources and knowledge present within the group in order to combat the problem. Furthermore, any issues that could not be solved by group member knowledge were researched and solutions were found online.

5 Results

5.1 Web Application

The Stripe payments service was implemented successfully and has provided for a very convenient form of payment for Smart Parking Ltd's customers. Customers using the car parks who do not possess a season permit are automatically sent an email when their vehicle leaves any one of the car parks, with a link to pay their bill that is accumulated from their time in the car park. The link takes them to the web application, where they will login and be presented with the Stripe payments interface.

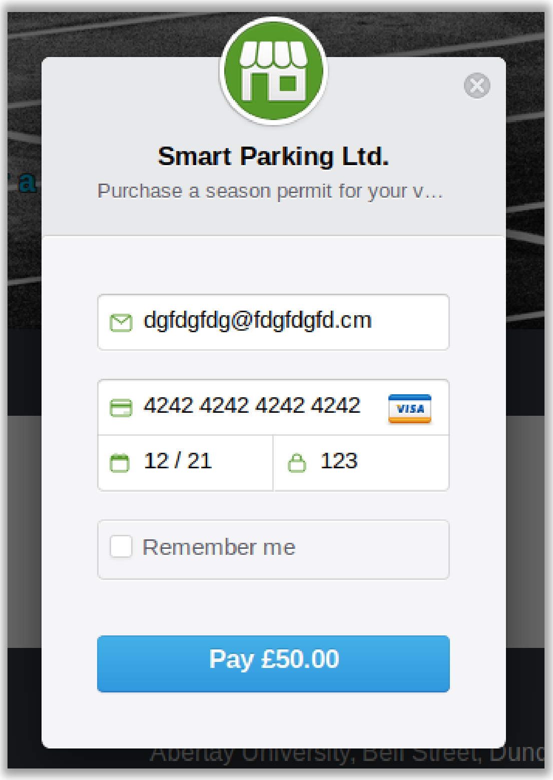
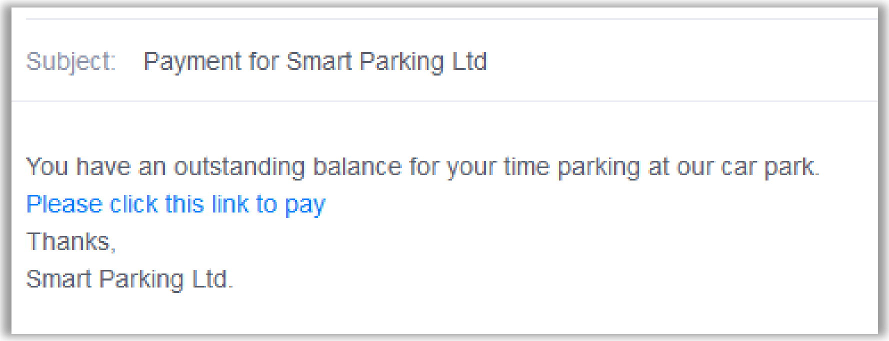


Figure 5: Stripe payments interface for the web application.

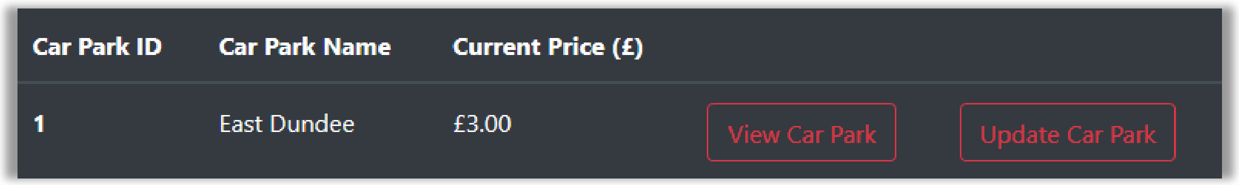


Subject: Payment for Smart Parking Ltd

You have an outstanding balance for your time parking at our car park.
Please click this link to pay
Thanks,
Smart Parking Ltd.

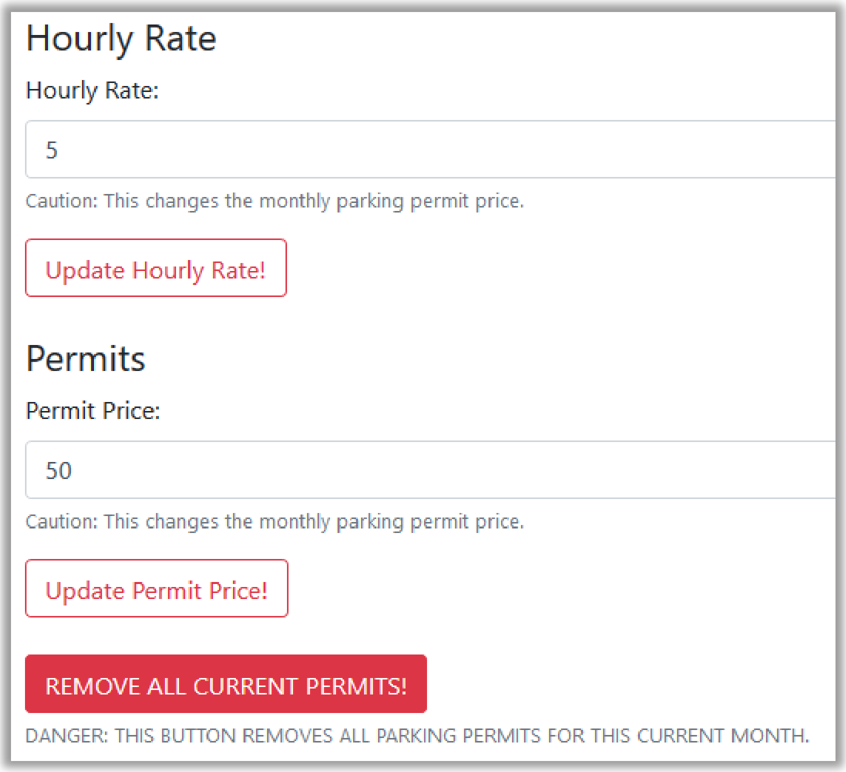
Figure 6: The email sent to the customer after leaving the car park.

The staff at Smart Parking Ltd can be satisfied that they have a convenient way of managing their car parks details and prices through the administration panel on the web application.



Car Park ID	Car Park Name	Current Price (£)		
1	East Dundee	£3.00	View Car Park	Update Car Park

Figure 7: The managing car parks section of the admin panel.



Hourly Rate

Hourly Rate:

5

Caution: This changes the monthly parking permit price.

Update Hourly Rate!

Permits

Permit Price:

50

Caution: This changes the monthly parking permit price.

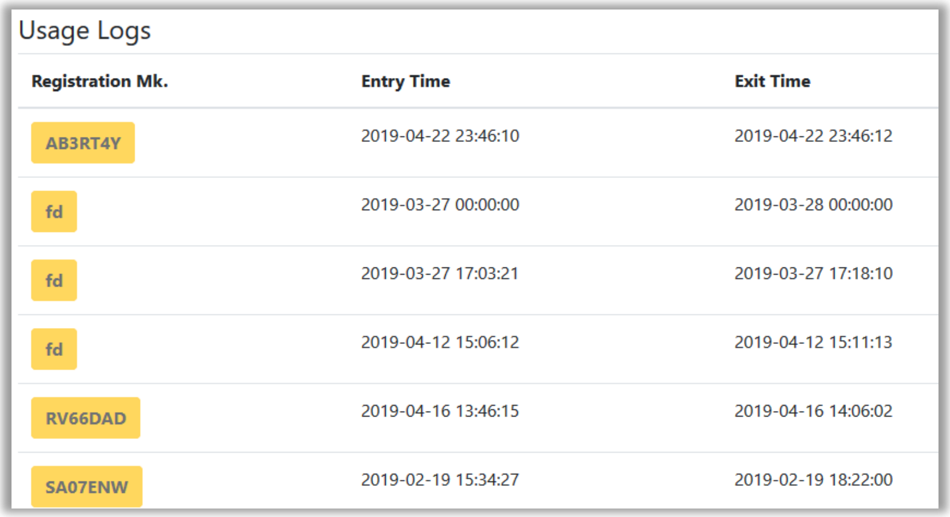
Update Permit Price!

REMOVE ALL CURRENT PERMITS!

DANGER: THIS BUTTON REMOVES ALL PARKING PERMITS FOR THIS CURRENT MONTH.

Figure 8: The updating prices section of the admin panel.

A low priority requirement was to implement a way in which Smart Parking Ltd staff can view usage statistics that show vehicles entering and exiting the car parks. The project had been going as planned so this requirement was added successfully, so now the staff can be pleased that they can keep a history of who is using the car park through the user interface rather than manually checking the database.



Usage Logs

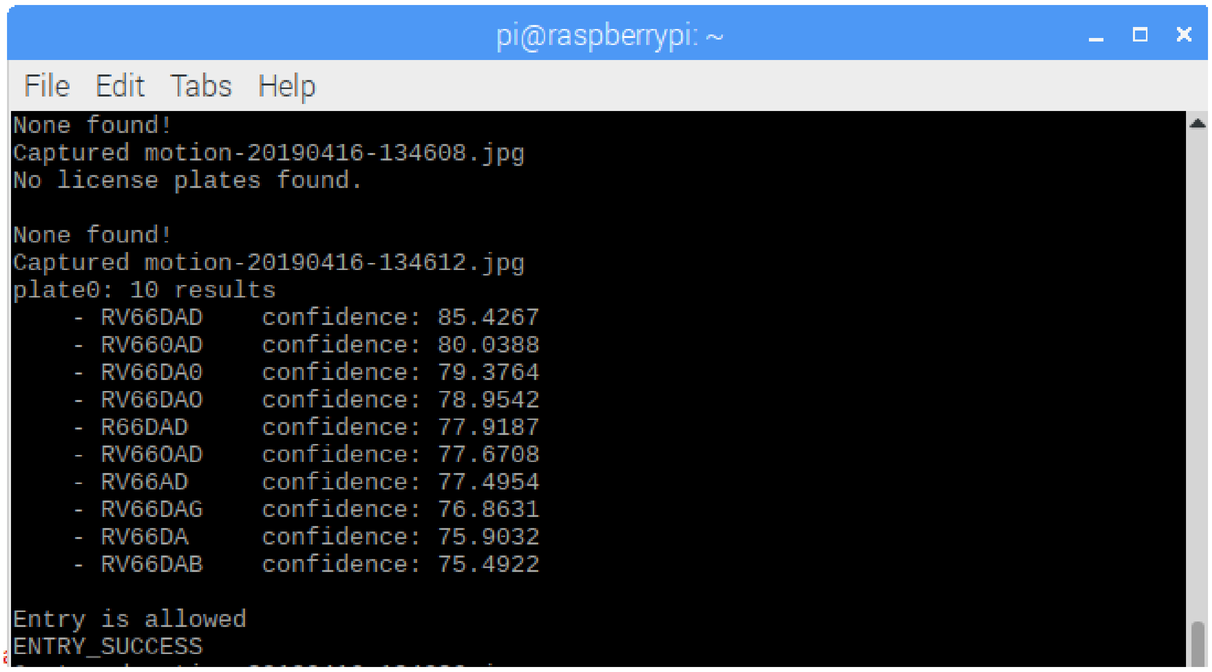
Registration Mk.	Entry Time	Exit Time
AB3RT4Y	2019-04-22 23:46:10	2019-04-22 23:46:12
fd	2019-03-27 00:00:00	2019-03-28 00:00:00
fd	2019-03-27 17:03:21	2019-03-27 17:18:10
fd	2019-04-12 15:06:12	2019-04-12 15:11:13
RV66DAD	2019-04-16 13:46:15	2019-04-16 14:06:02
SA07ENW	2019-02-19 15:34:27	2019-02-19 18:22:00

Figure 9: The car park usage logs section of the admin panel.

Although the system has been designed intuitively so that minimal training is required to use it, a user manual was created to guide a user through the process of using Smart Parking Ltd's car parks. This can be found in Appendix E.

5.2 Plate Recognition

One of the most important aspects of the system is the accuracy and reliability of the hardware that is being used. It is essential to the team and to the client that the hardware detects approaching cars and can accurately interpret the vehicle registration number of all vehicles it is presented with. The system that has been developed can detect the correct identifiers on the approaching vehicle by taking a picture of the vehicle's registration plate. The accuracy varies due to the quality of the image taken and the proximity of the plate in relation to the camera.



```
None found!
Captured motion-20190416-134608.jpg
No license plates found.

None found!
Captured motion-20190416-134612.jpg
plate0: 10 results
    - RV66DAD    confidence: 85.4267
    - RV660AD    confidence: 80.0388
    - RV66DA0    confidence: 79.3764
    - RV66DAO    confidence: 78.9542
    - R66DAD     confidence: 77.9187
    - RV660AD    confidence: 77.6708
    - RV66AD     confidence: 77.4954
    - RV66DAG    confidence: 76.8631
    - RV66DA     confidence: 75.9032
    - RV66DAB    confidence: 75.4922

Entry is allowed
ENTRY_SUCCESS
```

Figure 10: The Pi successfully identifying a license plate image and displaying the list of detected values by confidence score.

The average accuracy of the ALPR was calculated by having it detect the same number plate ten times and tracking how many correct predictions it made on what the number is.

PLATE NUMBER	ACCURACY OVER 10 DETECTIONS (%)
RV66 DAD	70
H160 JKF	80
GZGI CPD	90
VI2 LAF	90

Figure 11: The Pi successfully identifying a number plate image and displaying the list of detected values by confidence score.

Over four different plates, the average accuracy came to be 82.5%. A higher average accuracy of at least 90% would be ideal, but for the purposes of concept demonstration this score will be adequate.

5.3 Testing

Testing is essential to ensure a high-quality product for Smart Parking Ltd. Each module was tested individually before integration tests were carried out.

The web application was tested by using heuristic test logs, component test logs, and functional test logs. The tests included checking the forms to see if invalid data could be inputted and ensuring that links on the pages directed the user to the correct location. Initially, most of the tests were carried out successfully, however there were a few minor issues that were rectified in order to achieve a 100% pass rate.

All test logs can be found in Appendix B.

5.3.1 Quality Assurance

In the project proposal that was written up during the planning phase, it was stated that "when the quality assurance and testing phases have been completed to an appropriate standard and the product has been delivered on time, within budget, and of the required quality with only few minor issues after deployment, then it can ultimately be said that the project has been successful".

The ISO 9126 software quality standards (SQA, 2018) were used in determining whether this project has been successful. The six quality characteristics defined by ISO 9126 have been reviewed:

1. **Functionality:** The requirements and component testing has been carried out to ensure that the functionality is there and working to the client's standard, but also that it is secure and free from any flaws.

2. **Reliability:** Regarding number plate recognition accuracy, this was tested by running a plate 10 times to discover the average recognition accuracy. This was deemed successful, however leaves room for improvement, but this is dependent on external factors such as brightness, contrast, and the position

of the number plate in relation to the camera.

3. **Usability:** The web application has been developed with understandability, learnability, and operability in mind. This has ensured that the final working product is user-friendly and can be operated with minimal training.

4. **Efficiency:** All actions on the web application execute within the time frame set out in the system performance requirements. The physical system is able to read a plate before communicating with the web application API very quickly and allows the customer to enter or exit the car park within five seconds of approaching the barrier.

5. **Maintainability:** The web application has been developed using the Model-View-Controller architecture and object-oriented best practices to ensure that both the development team and any future development teams undertaking the project can easily pick up on how it works and smoothly integrate new features. This has been backed up by generating documentation through phpDox that will allow the developers to explore the API.

6. **Portability:** The application was developed using PHP with the database utilising the MariaDB engine, meaning that the web application can be hosted on servers with any operating system, whether its Windows, Mac, or Linux. The physical system was developed in Python, also meaning that it can be run on any operating system.

5.3.2 Requirements

The development team and Smart Parking Ltd signed two agreement forms at the beginning of the development phase, one for project deliverables and another for requirements. These agreement forms were reviewed to ensure that the development team has met what has been set out. These can be found in Appendix B.

As seen in the requirements trace log in Appendix B, all the must-have requirements agreed upon by the development team and Smart Parking Ltd have been completed to a sufficient standard. Smart Parking Ltd should be pleased to know that a low priority requirement to show car park usage logs on the admin panel was successfully implemented. However, another low priority requirement for pulling vehicle details from the DVLA was not implemented due to time constraints and should be brought forward for a future version if Smart Parking Ltd insists on having this feature.

5.3.3 System Performance

Performance requirements were defined in the planning phase. These are:

1. Communications with the database need to be performed within three seconds. This is especially required during peak times where multiple users could be using the web application and car parks at any one time. At any one time, the maximum number of customers using the car parks is equal to how many entry and exit barriers there are.

2. The pages on the web application must load within three seconds.

This was tested to the best of the development team's ability. However, system performance testing is constrained because there is only one Raspberry Pi to act as the physical system. When multiple Raspberry Pi's are available, system performance testing will be more accurate. Ideally, when the systems are installed in each car park and before full release, every entry and exit should be tested with a car simultaneously with multiple users executing actions on the web application to check server stress under the most extreme conditions.

6 DISCUSSION

6.1 GENERAL DISCUSSION

The web application has been developed with user-friendliness in mind which has resulted in well-designed web pages that makes it easy for users of all types, whether they are technically adept or are not confident with computers, to make use of all parts of the web application.

The physical system is very cost efficient, even more so than what was originally envisaged due to the realisation that detecting movements in pixels within the camera to detect motion was better than using a motion sensor. This means that each physical system unit is now cheaper to build. The system performs much better and is much more maintainable than what was originally planned due to the elimination of the TCP client and server which meant that images did not need to be sent over a network, instead the Raspberry Pi only needs to send the number plate string to an API on the web server.

The project management carried out by the development team was very effective in ensuring a high-quality product for Smart Parking Ltd. Key performance indicators were identified in the planning phase with the intention of being regularly reviewed during the development phase.

1. *If the time needed for each developer to complete a task is longer than the estimations made in the planning phase. If the task takes longer than expected, then the project may not be delivered on time.*

   Some tasks, like setting up the Raspberry Pi, took longer than expected due to developer inexperience. However, other tasks such as development of the web application were carried out a lot quicker than expected, so the extra time was available to keep the project on track.

2. *If the time that each developer has spent on the project in total is sufficient enough to complete all tasks in time for the deadline. If the developers have not been putting enough time into the project, then it may not be completed on time.*

   A few group members did not put in enough hours, so the remaining group members had to take on more hours in order to complete the tasks in time. This was helped further when a new group member was assigned to take on some overdue tasks.

3. *If there are a large number of adjustments that need to be made to the schedule to accommodate for any flaws. This could cause the project manager and stakeholders to lose sight of the final product and cause confusion and stress, resulting in a delayed and potentially low-quality product being delivered.*

   Any adjustments to the schedule were few and very minor, meaning that the overall plan was not hugely affected, and the product has been completed on time.

4. *The satisfaction of customers. A large number of complaints means that the customers are not happy with the final product and will likely take their business elsewhere, hurting the company's profits and reputation.*

   This key performance indicator is not yet applicable to the project. This must be reviewed on release and regularly thereafter.

5. *If there are a high number of errors. A good quality product will only have a few minor issues and bugs, ensuring the company maintain their reputation as a reliable car parking service.*

   The product has been tested extensively, with any errors that arose have now been fixed.

6. *If the project milestones are completed in a timely manner, then it will show that the project is still on the correct course and that the delivery of the final product on time is likely, ensuring that the customer can get their new system deployed as soon as possible to stay on top of the game in terms of modern-day parking solutions in Dundee.*

   In the weekly meetings, the milestones were reviewed and were confirmed to be on track.

7. *If extensive training were required for anybody to use the product. This would result in extra costs for the client to provide training plus customers may not want to use the products if they must learn a new way of doing things that they have done many times before.*

   The interfaces to the system, the barrier at the physical system and the web application, have both been developed with intuition in mind so that the customers can easily find their way about how to work with the system. With the web application being designed using Bootstrap, it has allowed for a design that is common amongst many websites, meaning that users will already be familiar with the controls. The physical system is very easy to use, with the driver simply only needing to approach the barrier for everything to happen automatically thereafter.

When this system is implemented, it will greatly out-perform rival parking companies in Dundee because customers will be able to get in and out of the car parks smoothly and in short time, meaning that they will be less stressed when faced with the daunting task of finding parking and navigating through rush-hour traffic. The customers who use the car parks will be so pleased that word-of-mouth will spread to other potential customers in Dundee, thus increasing the demand for people wanting to use Smart Parking Ltd's car parks of the future.

6.2 CONCLUSIONS

The client, Smart Parking Ltd, identified their initial requirements for this project which was essentially requiring a system that would allow users of their parking solutions to utilise a "smarter" system that would allow for easier entry and exit and provided them with a monthly permit instead of the traditional ticket-based system. The development team for this project, group Error 418, were able to offer extra facilities to the client. Facilities such as a website that would allow their customers to manage and pay for their account usages. This would also provide a platform for the company to administrate all of their own car parks providing them with usage logs and statistics of what vehicles are where. This has all been proposed and implemented with efficiency and accuracy kept in mind. This project meets all of the requirements initially stipulated by the client, Smart Parking Ltd, and ultimately provides a realistic solution to their current problems which will boost their company's reputation within the parking solutions industry.

At the commencement of this project, it was made clear to the development team the issues that the client was facing, and the struggle within the parking solutions business. The solution that has been developed by the team is an ideal one that meets all of the goals and outcomes set out originally by the client. This system will ensure "smarter" parking for all of the client's customers, making it more convenient for them to park at all of the sites operated by Smart Parking Ltd.

If the client, Smart Parking Ltd, decides not to use the system that has been developed then they will be faced with the same issues that they currently face – they would not be taking any steps forward and thus no competitive edge on rivals would be gained. Furthermore, one of the key outcomes of this project was to enhance Smart Parking Ltd company reputation. Using the system will ensure their reputation increases amongst customers and other competitor companies. Failure to use this system will result in no increased reputation and therefore not meeting one of the essential elements of this project.

During the course of the development of this project it was made possible to keep refining the system. This was done by reworking the way in which the hardware of the system communicates and sends data to the server-side part of the system. This had made the system cheaper to implement but equally as important is a faster and more efficient system when taking into account the network requirements and the workload placed onto the small chip inside the Raspberry Pi.

6.3 Future Work

As the project currently stands, there is nothing that would need to be added in order to reach completion for this project. However, during the course of the development, the development team, group Error 418, discovered that there would be some extra developments that could be done to the project if given a sufficient amount of funding in order to undertake these said developments. Some of these developers that the development team would be keen to undertake are:

Expanding upon the current web platform and developing it to a point where the client, Smart Parking Ltd, can use this as a front page for their company. Developments that would be planned to this is to add and expand upon the pages on the site, pages such as adding company details to give a more person feel, a gallery page to allow customers to see what purchasing a permit will provide them. Finally, a contact page to allow viewers to the site a convenient method of contacting the business.

Social media integration for Smart Parking Ltd's social media pages. For example, showing the latest posts from Facebook or Twitter on the side of the page to make the customer feel integrated and informed about the business and what their goals and mission is.

An overhaul of the way that the hardware handles the images. Currently, the images would just be deleted, however the development team would like the images to be sent to a file server. These images could then be displayed either for customers to view or only accessible to the administrators and used as evidential material in the event of a dispute between the customers and the business over parking charges.

Integration of the web platform to the DVLA registration number database. This database is publicly accessible with an application of their API key. This would allow the client, Smart Parking Ltd, to check the details of the vehicles that are within their premises. This would allow checks such as basic vehicle details and the ability for more advanced checks such as road tax, MOT status, and other details about the vehicle such as if its reported stolen, scrapped, or written off.

6.4 Call to Action

The development team, group Error 418, could offer the client, Smart Parking Ltd, free removal and disposal of the old and antiquated system whilst the installation of the newly developed system is being integrated. This could also be done at a convenient time for Smart Parking Ltd to avoid loss of business and disruption of their customer base. An installation time period can be negotiated between the development team and the client to allow for a quick and accurate implementation of the developed project.

Additionally, the development team are keen to continue development of this project and complete all of the suggested developments for the client. Therefore, the development team would offer the development of any additional features to be done at a discounted rate. This would mean a better system for the client without the expense of contacting another development team who would need to re-assess the client's needs when the current development team are already well aware of Smart Parking Ltd's business needs and what they would like added to the project.

If the client, Smart Parking Ltd, would like to take up any of these offers then they can do so by contacting the development team with the following contact information:

C/O Error 418
Abertay University
Bell Street
Dundee
DD81HG
error418@notarealemail.co.uk

7 TABLE OF FIGURES

8 REFERENCES

Adermann, N. & Boggiano, J., 2019. *Composer.* [Online]
Available at: https://getcomposer.org/
[Accessed 20 April 2019].

Arias, D., 2018. *Hashing in Action: Understanding bcrypt.* [Online]
Available at: https://auth0.com/blog/hashing-in-action-understanding-bcrypt/
[Accessed 16 April 2019].

Bootstrap team, 2019. *Bootstrap - The most popular HTML, CSS, and JS library in the world.* [Online]
Available at: https://getbootstrap.com/
[Accessed 16 April 2019].

Ibrahim, H., 2017. *Car Parking Problem in Urban Areas, Causes and Solutions,* s.l.: SSRN.

Microsoft, 2017. *Synchronous Server Socket Example.* [Online]
Available at: https://docs.microsoft.com/en-us/dotnet/framework/network-programming/synchronous-server-socket-example
[Accessed 20 March 2019].

OpenALPR Technology Inc, 2017. *OpenALPR Documentation.* [Online]
Available at: http://doc.openalpr.com/
[Accessed 20 April 2019].

OpenALPR, 2019. *Cloud API- openalpr 2.6.101 documentation.* [Online]
Available at: http://doc.openalpr.com/cloud_api.html
[Accessed 15 March 2019].

OWASP, 2017. *OWASP Top 10 Most Critical Web Application Security Risks.* [Online]
Available at: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
[Accessed 16 March 2019].

SQA, 2018. *ISO9126 - Software Quality Characteristics.* [Online]
Available at: http://www.sqa.net/iso9126.html
[Accessed 16 April 2019].

stripe, 2019. *Online payment processing for internet businesses - Stripe.* [Online]
Available at: https://stripe.com/en-GB/
[Accessed 20 April 2019].

stripe, 2019. *stripe/stripe-php: PHP library for the Stripe API.* [Online]
Available at: https://github.com/stripe/stripe-php
[Accessed 20 April 2019].

theseer, 2019. *theseer/phpdox: Documentation generator for PHP Code using standard technology (SRC, DOCBLOCK, XML and XSLT).* [Online]

Available at: https://github.com/theseer/phpdox
[Accessed 20 April 2019].

9 APPENDICES

9.1 APPENDIX A – WEB APPLICATION CODE

The following code snippets capture the key areas of the code.

Submitting Permit Payment

```php
public function submitPayment()
{
    if (AuthHelper::isLoggedIn())
    {
        if (isset($this->params['reg']) &&
            isset($this->params['entrydatetime']) &&
            isset($this->params['cost']))
        {
            $reg = $this->params['reg'];
            $entryDateTime = $this->params['entrydatetime'];
            $cost = $this->params['cost'];

            \Stripe\Stripe::setApiKey(Config::STRIPE_SECRET_KEY);
            $token = $this->params['stripeToken'];

            $charge = \Stripe\Charge::create(
                ['amount' => $cost,
                 'currency' => 'gbp',
                 'source' => $token,
                 'description' => 'Parking payment for ' . $reg . ' at ' . $entryDateTime]
            );

            $this->parkingModel->setPaid($reg, $entryDateTime);
            header('Location: index.php?controller=vehicle&action=index');
        }
    }
}
```

Updating Prices in Admin Panel

```php
public function updatePermitPrice()
{
    if (!AuthHelper::isAdmin() || !isset($this->params['price']))
        exit(header('Location: index.php'));

    $price = $this->params['price'];
    $this->configModel->setConfigValue(ConfigModel::PERMIT_PRICE, $price);

    header('Location: index.php?controller=admin&action=permits');
}
```

API for Using Car Park

```php
public function check()
{
    // Need to check the IP of the client so not just anyone and their nan can do this
    if (isset($this->params['reg']) && isset($this->params['carparkid']))
    {
        $reg = $this->params['reg'];
        $carparkId = $this->params['carparkid'];

        // Check the carpark actually exists
        $carpark = $this->carparkModel->getCarparkById($carparkId);
        if ($carpark == null)
        {
            // Return error json;
            $json = new Json(array('Message' => 'CARPARK_NOT_FOUND'));
            $json->execute();
            return;
        }

        // Check the vehicle exists
        $vehicle = $this->vehicleModel->getVehicle($reg);
        if ($vehicle == null)
        {
            $json = new Json(array('Message' => 'VEHICLE_NOT_FOUND'));
            $json->execute();
            return;
        }

        // Check if the vehicle is already in the car park,
        // this will tell us that the vehicle wants to leave
        if ($this->parkingModel->isInCarpark($reg))
        {
            $this->parkingModel->addExit($reg);

            $parking = $this->parkingModel->getLatestParking($reg);

            $entryDateTime = $parking->EntryDateTime;

            $this->sendPaymentEmail($vehicle->UserID, $reg, $entryDateTime);

            $json = new Json(array('Message' => 'EXIT_SUCCESS'));
            $json->execute();
```

```
                return;
            }
            // If not already in car park, then the vehicle is trying to enter
            else
            {
                $this->parkingModel->addEntry($reg, $carparkId);

                $json = new Json(array('Message' => 'ENTRY_SUCCESS'));
                $json->execute();
                return;
            }
        }

        $json = new Json(array('Message' => 'PARAMETERS_MISSING'));
        $json->execute();
}
```

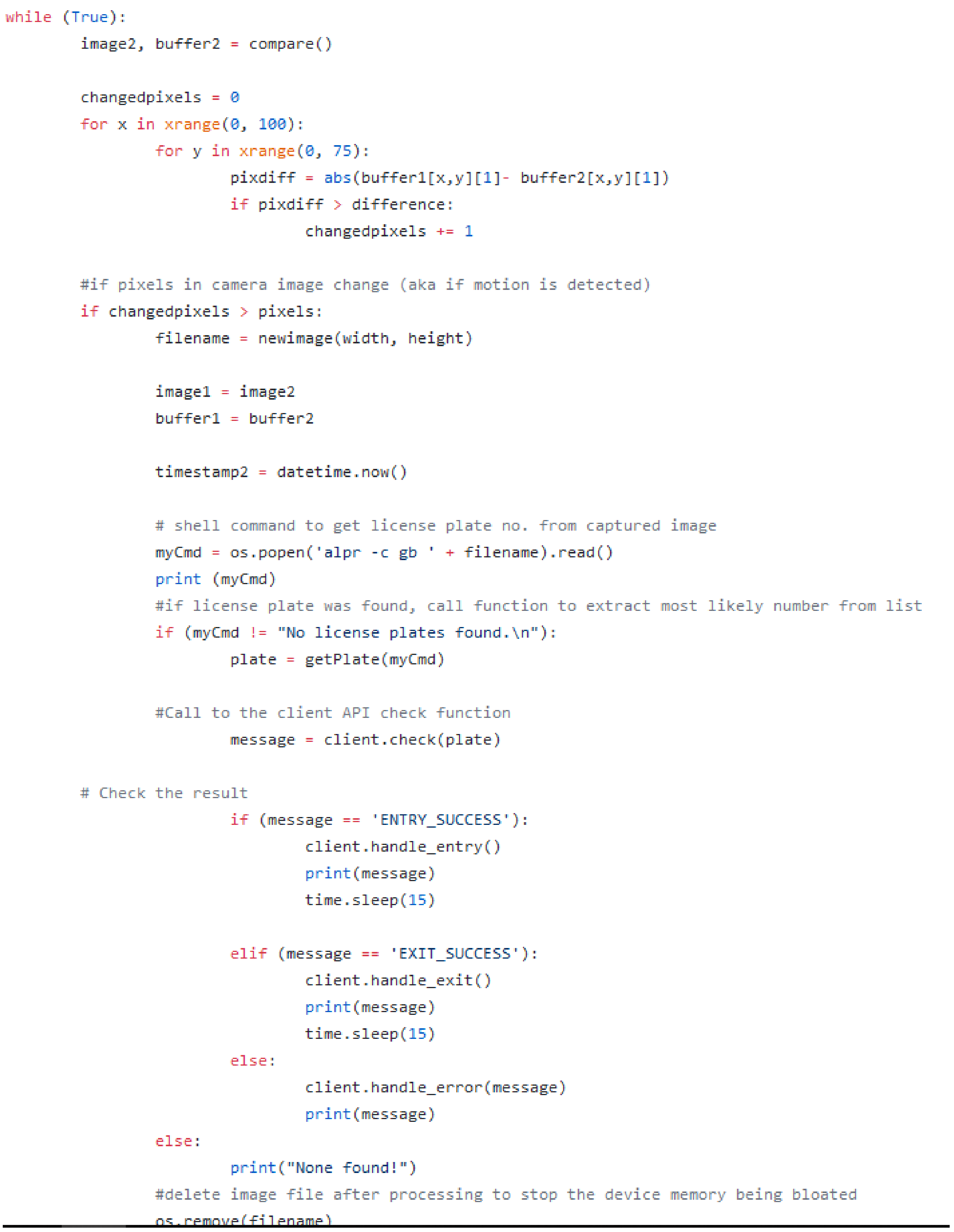
Adding a Parking Entry to the Database

```
public function addEntry($reg, $carparkId)
{
    $db = Database::getInstance();

    $sql = "INSERT INTO `Parking` (`Reg`, `EntryDateTime`, `CarparkID`)"
        . " VALUES (:reg, NOW(), :carparkId)";
    $query = $db->prepare($sql);
    $query->bindParam(':reg', $reg, PDO::PARAM_STR);
    $query->bindParam(':carparkId', $carparkId, PDO::PARAM_INT);
    $query->execute();
}
```

Motion Activated Camera

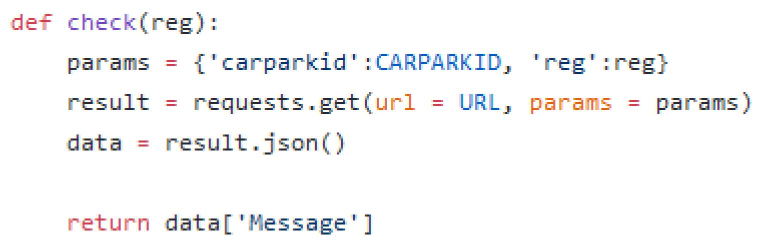
```
def compare():
        camera.resolution = (100, 75)
        stream = io.BytesIO()
        camera.capture(stream, format = 'bmp')
        stream.seek(0)
        im = Image.open(stream)
        buffer = im.load()
        stream.close()
        return im, buffer
def newimage(width, height):
        time = datetime.now()
        filename = 'motion-%04d%02d%02d-%02d%02d%02d.jpg' % (time.year, time.month,time.day, time.hour,time.minute, time.second)
        camera.resolution = (width, height)
        camera.capture(filename)
        print 'Captured %s' % filename
        return filename
```

Physical System Main Loop



```
while (True):
        image2, buffer2 = compare()

        changedpixels = 0
        for x in xrange(0, 100):
                for y in xrange(0, 75):
                        pixdiff = abs(buffer1[x,y][1]- buffer2[x,y][1])
                        if pixdiff > difference:
                                changedpixels += 1

        #if pixels in camera image change (aka if motion is detected)
        if changedpixels > pixels:
                filename = newimage(width, height)

                image1 = image2
                buffer1 = buffer2

                timestamp2 = datetime.now()

                # shell command to get license plate no. from captured image
                myCmd = os.popen('alpr -c gb ' + filename).read()
                print (myCmd)
                #if license plate was found, call function to extract most likely number from list
                if (myCmd != "No license plates found.\n"):
                        plate = getPlate(myCmd)

                #Call to the client API check function
                        message = client.check(plate)

        # Check the result
                        if (message == 'ENTRY_SUCCESS'):
                                client.handle_entry()
                                print(message)
                                time.sleep(15)

                        elif (message == 'EXIT_SUCCESS'):
                                client.handle_exit()
                                print(message)
                                time.sleep(15)
                        else:
                                client.handle_error(message)
                                print(message)
                else:
                        print("None found!")
                #delete image file after processing to stop the device memory being bloated
                os.remove(filename)
```

Raspberry Pi Code to Consume API



```python
def check(reg):
    params = {'carparkid':CARPARKID, 'reg':reg}
    result = requests.get(url = URL, params = params)
    data = result.json()

    return data['Message']
```

9.2 Appendix B – Test Logs

9.2.1 Requirements Tracing

Functional Requirements Table

ID	List of Agreed Requirements	Completed	Reason for not completing (if applicable)
APCS F1.1	The car park barrier must open for registered customers and subsequently close once the driver has safely passed through.	Yes	
APCS F1.2	The number plate of the vehicle entering the car park must be scanned to determine whether the customer is registered and allowed access.	Yes	
APCS F1.3	Vehicle details could be pulled from the DVLA database in order to send bills without the customer having to register on the web application.	No	This was a low priority requirement that was only to be implemented trusting there was extra time to be allocated to the task. This should be moved forward as a requirement for version 2.
APCS F1.4	Customers must be automatically billed for their time spent in the car park unless the vehicle is associated with a season permit. An email will be sent to them with the bill and option to pay through Stripe.	Yes	
Web F2.1	Customers must be able to register one account for multiple vehicles. This will require the SQL database to be designed correctly additionally, the PHP should also allow for the display of multiple vehicles on the page.	Yes	
Web F2.2	Customers must be able to add or remove vehicles from their account. There must be functionality on the PHP page to allow a user to do this. A query must also be running on the SQL database to update this.	Yes	
Web F2.3	Customers must be able to update their account details. There must be a PHP page that allows the user to do this and the SQL must query the database accordingly also.	Yes	
Web F2.4	Customers must be able to purchase a season permit that is valid for one vehicle only. The customers will purchase the permits through the PHP User Web Panel.	Yes	
Web F2.5	Season permits must last from month-begin to month-end. At the end of the month, the permit must	Yes	

	automatically expire. There must be no discounts for buying a permit closer to the end of the month.		
Web F2.6	Administrators must be able to change the prices of each of the car parks. This must be done through the PHP Administration Web Panel.	Yes	
Web F2.7	Administrators should be able to check a log of vehicles using the car park. The logs should contain the vehicle identifier, entry and exit dates, and pictures. This must be done through the PHP Administration Web Panel.	Yes	

Non-Functional Requirements Table

ID	List of Agreed Requirements	Completed	Reason for not completing (if applicable)
APCS NF1.1	Only registered customers must be allowed access to the car parks.	Yes	
APCS NF1.2	Season permits must be valid for one vehicle only.	Yes	
APCS NF1.3	Season permits must only last from month-begin to month-end.	Yes	
APCS NF1.4	The camera must be able to read and recognise the vehicle registration number within five seconds to prevent delays at the entrances and exits of each of the car parks.	Yes	
Web NF2.1	Helpful messages will be given to the users of the application.	Yes	
Web NF2.2	Must support all of the latest versions of browser applications.	Yes	
Web NF2.3	The interface of the application must be user friendly.	Yes	
Web NF2.4	The interface must be accessible to all users.	Yes	
Web NF2.5	The website must comply with regulations such as GDPR, therefore only necessary details about each customer and vehicle should be stored.	Yes	
Web NF2.6	Actions requiring communication with the database need to be performed within 3 seconds, especially at peak times when numerous users could be using the web application and the car parks concurrently.	Yes	

9.2.2 System Testing

Initial Test Logs

#	Function Under Test	Test Case	Expected	Actual	Pass/ Fail
1.1	**Login**	Valid email invalid password	Refuse login	Refused login	Pass
1.2		Valid email valid password	Allow login	Allowed login	Pass
1.3		Invalid email	Refuse login	Refused login	Pass
2.1	**Register**	Empty fields	Refuse register	Refused register	Pass
2.2		Existing email	Refuse register	Redirected to home page	Fail
2.3		Passwords don't match	Refuse register	Allowed register	Fail
2.4		Unique email, valid names, matching passwords	Allow register (and add new record to DB)	Allowed register and new user added to database	Pass
3.1	**Logout**	Logout clicked when logged in	Logout successfully	Logout successful	Pass
3.2		Logout clicked when not logged in (page is accessed directly through URL because logout won't appear when not logged in)	Redirect to homepage	Redirected to homepage	Pass
4.1	**Update password**	Passwords match	Change password successfully	Password changed successfully	Pass
4.2		Passwords don't match	Refuse password change	Password changed successfully	Fail
5.1	**Add vehicle**	Unique registration number	Add vehicle successfully	Vehicle added successfully	Pass
5.2		Registration number taken by another user	Refuse to add vehicle	Refused to add vehicle	Pass
5.3		Registration number already on list of user's vehicles	Refuse to add vehicle	Refused to add vehicle	Pass
5.4		Registration numbers case insensitive. Ex: User adds vehicle with reg 'sm53yxn' where another user has 'SM53YXN'.	Refuse to add vehicle	Refused to add vehicle	Pass
6.1	**Remove vehicle**	Button on user's list of vehicles	Removes vehicle successfully	Removed vehicle successfully	Pass
6.2		Removing reg specified via URL (i.e. manually going to index.php?controller=vehicle&action=remove&reg=[REG]) with a reg that belongs to the user	Removes vehicle successfully	Removed vehicle successfully	Pass

6.3		As above but with a reg not belonging to the user	Refuse to remove vehicle	Refused to remove vehicle	Pass
7.1	**Purchase permit**	Purchase permit for user’s own vehicle that doesn’t have a permit	Take to permit payment page (payments tested separately) and permit added once payment made	Taken to purchase page and permit was added once payment was made	Pass
7.2		Purchase permit for user’s own vehicle that already has a permit	Refuse to add permit	Refuse to add permit	Pass
7.3		Purchase a permit for a vehicle the user doesn’t own	Refuse to add permit	Refuse to add permit	Pass
8.1	**Update Hourly Rate**	Admin logged in and update the hourly rate with valid value (i.e. between 0-100)	Update successfully	Updated successfully	Pass
8.2		Admin logged in and update the hourly rate with invalid value (i.e. below 0 and above 100)	Refuse to update	Updated successfully	Fail
8.3		Admin not logged in	Refuse to update	Refuse to update	Pass
9.1	**Update Permit Price**	Admin logged in and update the permit price with a valid value (between 0-100)	Update successfully	Updated successfully	Pass
9.2		Admin logged in and update the permit price with an invalid value (below 0 or above 100)	Refuse to update	Updated successfully	Fail
9.3		Admin not logged in	Refuse to update	Refused to update	Pass
10.1	**Remove All Current Permits**	Button on admin panel with admin logged in	Removes all the permits	Removed all permits successfully	Pass
10.2		Admin not logged in	Refuse to remove all permits	Refused to remove all permits	Pass
	API	Type parameter not specified	Return “PARAMETERS_MISSING”	Returned “PARAMETERS_MISSING”	Pass
		Reg parameter not specified	Return “PARAMETERS_MISSING”	Returned “PARAMETERS_MISSING”	Pass

		Carpark ID parameter not specified	Return "PARAMETERS_MISSING"	Returned "PARAMETERS_MISSING"	Pass
		Type specified but not "entry" or "exit"	Return "TYPE_NOT_SPECIFIED"	Returned "TYPE_NOT_SPECIFIED"	Pass
		Non-existing carpark ID	Return "CARPARK_NOT_FOUND"	Returned "CARPARK_NOT_FOUND"	Pass
		Non-existing registration	Return "VEHICLE_NOT_FOUND"	Returned "VEHICLE_NOT_FOUND"	Pass
		Request entry to car park with valid reg	Return "ENTRY_SUCCESS"	Returned "ENTRY_SUCCESS"	Pass
		Request exit from car park with valid reg and existing parking logs (i.e. vehicle has entered car park)	Return "EXIT_SUCCESS"	Returned "EXIT_SUCCESS"	Pass
		Request exit from car park with valid reg but no existing logs (i.e. vehicle has not been marked to have entered the car park)	Return "PARKING_ENTRY_NOT_FOUND"	Returned "PARKING_ENTRY_NOT_FOUND"	Pass
		Case insensitive registration plate	Return "ENTRY_SUCCESS" or "EXIT_SUCCESS"	Returned "ENTRY_SUCCESS" with lowercase reg that is stored uppercase in DB and then subsequently "EXIT_SUCCESS" with random combination of upper and lower.	Pass

Resolution Test Logs

#	Function Under Test	Test Case	Reason for failure	Resolution	Pass/Fail
2.2	**Register**	Existing email	When register form submitted, user was redirected to home page.	Discovered the code will redirect the user to the home page if they are not logged in. This was changed to redirect if the user is logged in.	Pass

2.3		Passwords don't match	Register form submits successfully even when the passwords didn't match.	Discovered the "ID" parameter on the input HTML element had been changed to something different than is checked in the javascript validation. ID params were renamed to match the javascript validation.	Pass
4.2	**Update password**	Passwords don't match	Update password form submits successfully even when passwords didn't match.	Discovered the "ID" parameter on the input HTML element had been changed to something different than is checked in the javascript validation. ID params were renamed to match the javascript validation.	Pass
8.2	**Update hourly rate**	Admin logged in and update the hourly rate with invalid value (i.e. below 0 and above 100)	Update hourly rate form was submitted successfully even with values below 0 and above 100.	Added JS validation for ensuring price between 0-100 before form submitted.	Pass
9.2	**Update permit price**	Admin logged in and update the permit price with an invalid value (below 0 or above 100)	Update permit price form was submitted successfully even with values below 0 and above 100.	Added JS validation for ensuring price between 0-100 before form submitted.	Pass

9.2.3 Website Function Tests

Test Case	**Test Detail**	**Expected Result**	**Actual Result**	**Notes**
Home Page/Global				
Home Button	Press Button	Take to home page	As expected	
Gallery Button	Press Button	Nothing	As expected	
About Button	Press Button	Nothing	As expected	
Login Button	Press Button	Take to login page	As expected	
Register Button	Press Button	Take to register page	As expected	
Logout Button	Press Button	Logout of account	As expected	Only visible while logged in
Account Actions - Your Account	Press Button	Account management page	As expected	Only visible while logged in
Account Actions – Vehicles	Press Button	Vehicle management page	As expected	Only visible while logged in

Account Actions – Admin	Press Button	Administration panel page	As expected	Only visible while logged in
Email link	Press Link	Attempts to open email app on device	As expected	This may very between device
Phone Link	Press Link	Attempts to open phone app on device	As expected	This may very between device
Login Page				
Username field	Allows text input	Allows text input	As expected	
Password field	Allows password input	Allows password input	As expected	
Submit button	Submits values to the php script	Attempts login	As expected	
Registration Page				
First Name field	Text input	Allows text input	As expected	
Last Name field	Test input	Allows text input	As expected	
Email address field	Text input	Allows text input	As expected	
Password field	Password input	Allows password input	As expected	
Confirm Password field	Password input	Allows password input	As expected	
Register button	Press Button	Attempts registration	As expected	

Heuristic Test Logs

Website Navigation	**Compliance**			
	Always	**Sometimes**	**Never**	**Notes**
Clear indication of current page	✓			
Site structure is easily understandable	✓			
All major parts of the site are accessible from the home page	✓			
Interface is user friendly		✓		
All navigation links are easily understandable	✓			
All pages are consistent in layouts and styling	✓			

Website Control	**Compliance**			
	Always	**Sometimes**	**Never**	**Notes**
All features work as intended	✓			
All security control measures are met	✓			
All functions operate as labeled	✓			
A best practice approach has been adopted	✓			
Language	**Compliance**			
	Always	**Sometimes**	**Never**	**Notes**
The language used is simple and understandable	✓			
Jargon is avoided	✓			
Multi language support is offered			✓	
Accessibility options are there for those who need them		✓		
Errors	**Compliance**			
	Always	**Sometimes**	**Never**	**Notes**
Website handles all errors	✓			
API handles all errors	✓			
Hardware handles all errors	✓			
Errors make sense to the user	✓			
Errors give clear indication what is required from the user	✓			

9.3 Appendix C – Deliverables and Requirements

Agreement Form: Project Deliverables

Group	Error 418
Team Member	Michael McMillan, Jamie Beecroft, and Nicole Noterangelo
Programme	BSc Computing
Programme Specialist	Dr. Gavin Hales, Dr Lynsay Shepherd
The deliverables listed below will be submitted by the team by the due date.	

Part A deliverables	• Requirements Specification
	• Module Design Documentation
	• Completed Physical System Module
	  ◦ Design
	  ◦ Implementation
	  ◦ Testing
	• Completed Central Server Module
	  ◦ Design
	  ◦ Implementation
	  ◦ Testing
	• Completed Web Application Module
	  ◦ Design
	  ◦ Implementation
	  ◦ Testing
	• Module Testing Documentation
	• User Manual

Programme specialist's signature	[illegible] Amended to reflect group member changes. LS not available to sign - GH
Team members' signatures	[illegible] Jamie Beecroft [illegible] Michael McMillan [illegible] Nicole Notarangelo

Agreement Form: Requirements

Group Number: Error 418

Team members (print): Michael McMillan, Jamie Beecroft, and Nicole Notarangelo

Project Title: Automatic Parking Charge System

Please refer to the attached documentation for full details on the project. The requirements are listed in Table 1. The signatures below indicate that the requirements for this project have been agreed by the project stakeholders.

Any changes to the project documentation should be made using the correct change authorisation procedure agreed with the programme specialist.

Functional Requirements Table

ID	List of Agreed Requirements
APCS F1.1	The car park barrier must open for registered customers and subsequently close once the driver has safely passed through.
APCS F1.2	The number plate of the vehicle entering the car park must be scanned to determine whether the customer is registered and allowed access.
APCS F1.3	Vehicle details could be pulled from the DVLA database in order to send bills without the customer having to register on the web application.
APCS F1.4	Customers must be automatically billed for their time spent in the car park unless the vehicle is associated with a season permit. An email will be sent to them with the bill and option to pay through Stripe.
Web F2.1	Customers must be able to register one account for multiple vehicles. This will require the SQL database to be designed correctly additionally, the PHP should also allow for the display of multiple vehicles on the page.
Web F2.2	Customers must be able to add or remove vehicles from their account. There must be functionality on the PHP page to allow a user to do this. A query must also be ran on the SQL database to update this.
Web F2.3	Customers must be able to update their account details. There must be a PHP page that allows the user to do this and the SQL must query the database accordingly also.
Web F2.4	Customers must be able to purchase a season permit that is valid for one vehicle only. The customers will purchase the permits through the PHP User Web Panel.
Web F2.5	Season permits must last from month-begin to month-end. At the end of the month, the permit must automatically expire. There must be no discounts for buying a permit closer to the end of the month.
Web F2.6	Administrators must be able to change the prices of each of the car parks. This must be done through the PHP Administration Web Panel.

Web F2.7	Administrators should be able to check a log of vehicles using the car park. The logs should contain the vehicle identifier, entry and exit dates, and pictures. This must be done through the PHP Administration Web Panel.

Non-Functional Requirements Table

ID	**List of Agreed Requirements**
APCS NF1.1	Only registered customers must be allowed access to the car parks.
APCS NF1.2	Season permits must be valid for one vehicle only.
APCS NF1.3	Season permits must only last from month-begin to month-end.
APCS NF1.4	The camera must be able to read and recognise the vehicle registration number within five seconds to prevent delays at the entrances and exits of each of the car parks.
Web NF2.1	Helpful messages will be given to the users of the application.
Web NF2.2	Must support all of the latest versions of browser applications.
Web NF2.3	The interface of the application must be user friendly.
Web NF2.4	The interface must be accessible to all users.
Web NF2.5	The website must comply with regulations such as GDPR, therefore only necessary details about each customer and vehicle should be stored.
Web NF2.6	Actions requiring communication with the database need to be performed within 3 seconds, especially at peak times when numerous users could be using the web application and the car parks concurrently.

Stakeholders	**Signatures**	**Date**
Jamie Beecroft	[illegible]	22/02/2019

Nicole Notarangelo	[illegible]	22/02/2019
Michael McMillan	[illegible]	22/02/2019

Programme Specialist

Dr. Gavin Hales [illegible]

9.4 APPENDIX D – MINUTES

DATE: 15/01/19
TIME: 14:00
PLACE: Room 4510, Zone E
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Chloe Brown

ATTENDEES:
Michael McMillan
Jamie Beecroft
Chloe Brown

APOLOGIES FOR ABSENCE:
N/A

AGENDA:
- Project review
- Create questions for the client
- Get the client to answer the questions
- Set up Trello for project management
- Set up Google documents to allow file sharing within the team
- Start work on white paper

MINUTES:
- Decided roles by show of hand, Jamie wanted to lead, Michael wanted to develop, and Chloe Brown was the minute taker.
- Team decide that questions needed to be created right at the beginning and presented to the client to help understand the extent of the project.
- Trello was decided as the best project management system by Michael. This was due to the fact he had used this software before and knew it would help the team stay on track.
- Google documents was set up by Jamie as we could all edit and share documents. This would help us as a team.
- The proposal was discussed on when to start writing this up. As a team we decided that the best thing to do was start it straight away to get ahead of the game.

Date of next meeting: 22nd of January at 1:00 pm

AOCB
- No

DATE: 22/01/19
TIME: 13:00
PLACE: Room 4510, zone H
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Chloe Brown

ATTENDEES:
Michael McMillan
Jamie Beecroft
Chloe Brown
Calum Newton

APOLOGIES FOR ABSENCE:
N/A

AGENDA:
- Introduction of new team member
- Deciding tasks to be done for next week

MINUTES:
- Calum Newton joined the group
- Completed functional requirements
- Completed non-functional requirements
- Intro to be completed by next week
- Agreement form to be signed for next week

Date of next meeting: 29/01/2019 at 1:00pm

AOCB
- No

DATE:29/01/19

TIME: 13:00
PLACE: Room 4510, zone H
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Chloe Brown

ATTENDEES:
Michael McMillan
Jamie Beecroft
Chloe Brown

APOLOGIES FOR ABSENCE:
N/A

AGENDA:
- Agreement form
- Task organisation and distribution
- GitHub

MINUTES:
- Agreement form signed by client
- Organised tasks for each module on Trello
- Set up GitHub repositories

Date of next meeting: 05/02/2019 at 1:00pm

AOCB
- No

DATE:05/02/19
TIME: 13:00
PLACE: Room 4510, zone C
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Chloe Brown

ATTENDEES:
Michael McMillan
Jamie Beecroft
Chloe Brown

APOLOGIES FOR ABSENCE:
N/A

AGENDA:
- Raspberry Pi
- Update on what has been done on trello
- Update on what has been done in the white paper
- Find Calum

MINUTES:
- Raspberry pi has been asked for to help build the development of the parking system and will hopefully receive it in the next couple of weeks.
- Trello has been updated to show what has been completed since the end of last week's meeting.
- Background section has been started of the white paper by Michael and will be completed in the next few days.
- Discussed where Calum is, he has not signed the agreement form for the white paper or yet to start any of the documentation towards the final project.

Date of next meeting: 12/02/2019 at 1:15pm

AOCB
- No

DATE:12/02/19

TIME: 13:00
PLACE: Room 4510, zone B
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Jamie Beecroft

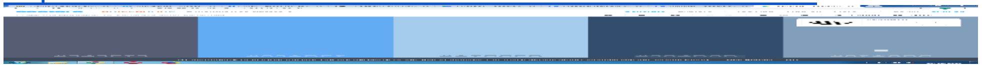
ATTENDEES:
Michael McMillan
Jamie Beecroft

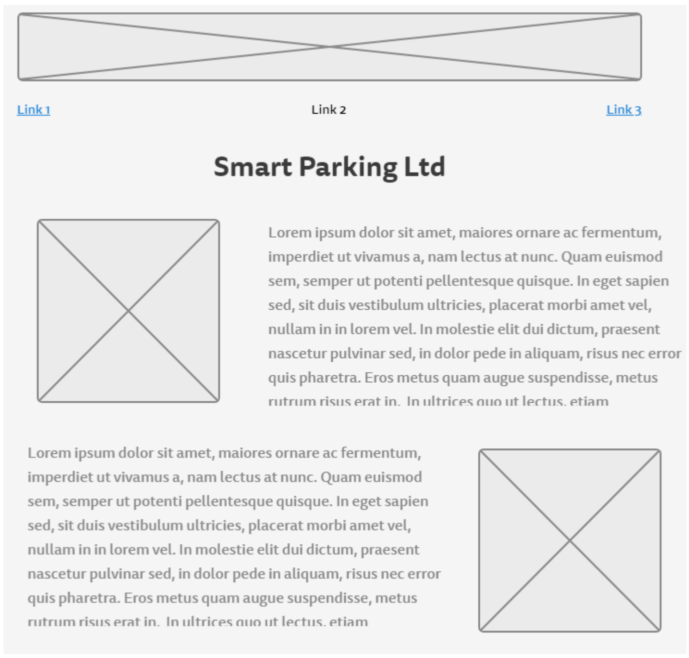
APOLOGIES FOR ABSENCE:
Chloe Brown

AGENDA:

- Discuss progress
- Discuss team member participation efforts
- Discuss user interface
    - Identify key feature requirements
    - Colour Schemes
    - Wireframes
    - Sketch ups

MINUTES:

- Reasonable progress has been made with the project
    - User interface has a basic look that allows us to test current functionality
    - Client and server have active communication, however, there is an issue with the client occasionally causing a timeout with the server.
    - Website has almost been completed regarding the backend system however still requires work on the front end to make it more user friendly.
    - Stripe payment is working.
- Identified some members of the group that have not been attending weekly meetings or making any effort to communicate with the group regarding apologies for absence.
- User interface on the website will use bootstrap to provide easy and reliable scalability across all devices.
    - The colour scheme chosen will be blue and shades around See: https://coolors.co/545e75-63adf2-a7cced-304d6d-82a0bc
    - 

- 
Link 1
Link 2
Link 3
Smart Parking Ltd
Lorem ipsum dolor sit amet, maiores ornare ac fermentum, imperdiet ut vivamus a, nam lectus at nunc. Quam euismod sem, semper ut potenti pellentesque quisque. In eget sapien sed, sit duis vestibulum ultricies, placerat morbi amet vel, nullam in in lorem vel. In molestie elit dui dictum, praesent nascetur pulvinar sed, in dolor pede in aliquam, risus nec error quis pharetra. Eros metus quam augue suspendisse, metus rutrum risus erat in. In ultrices quo ut lectus, etiam
Lorem ipsum dolor sit amet, maiores ornare ac fermentum, imperdiet ut vivamus a, nam lectus at nunc. Quam euismod sem, semper ut potenti pellentesque quisque. In eget sapien sed, sit duis vestibulum ultricies, placerat morbi amet vel, nullam in in lorem vel. In molestie elit dui dictum, praesent nascetur pulvinar sed, in dolor pede in aliquam, risus nec error quis pharetra. Eros metus quam augue suspendisse, metus rutrum risus erat in. In ultrices quo ut lectus, etiam

Date of next meeting: 19/02/2019 at 1:00pm

AOCB

- No

DATE:19/02/19
TIME: 13:15
PLACE: Room 4510, Zone E
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Jamie Beecroft

ATTENDEES:
Michael McMillan
Jamie Beecroft

APOLOGIES FOR ABSENCE:
None from Chloe Brown.

AGENDA:
- Continued Absence from group members.
- Discuss Progress

MINUTES:
- Calum has had a further week of nonattendance at the weekly meetings. Therefore, he will not be assigned any tasks as the previous ones have not been completed and he is not communicating with the group to update us on his progression.
- Reasonable progress has been made the project, no issues identified by the development team. This is to be continually reviewed in the coming weeks.

Date of next meeting: 19/02/2019 at 13:00

AOCB
- No

DATE:26/02/19
TIME: 13:15
PLACE: Room 4510, Zone E
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Jamie Beecroft

ATTENDEES:
Michael McMillan
Jamie Beecroft
Nicole Notarangelo

APOLOGIES FOR ABSENCE:
None from Chloe Brown

AGENDA:

- Group members attendance at participation with regards to the project.
- New group member to the project
- Project hardware
- Front end web platform

MINUTES:

- Several group members have not shown continued participation in the project, these being:
  - Calum Newton has not shown any initiative to contribute to the group project, therefore no further tasks have been assigned to him and all previous tasks have been taken by other group members.
  - Chloe Brown has not attended for a few weeks and has possibly withdrawn from the course. No new tasks will be assigned to her and all current tasks have been reallocated.
- Nicole has joined the team and has been given access to the following:
  - Google Docs (all information and documentation with regards to the project)
  - Trello (group tasks management system)
  - GitHub (Code collaboration)
  - Group chat (Facilitate effective communications between team members)
- As the project site and backend ins coming along smoothly, it is time to ensure that the hardware and the client side of the application on the barrier component is functioning as intended. This was the goal for this week however a HDMI cable is required for the visual display of the Raspberry PI.
- The front end for the web platform is in a functional state however still requires the front-end design, namely bootstrap to be applied to the system. This will be a continuing task in the coming weeks.

Date of next meeting: 05/03/2019 at 13:00

AOCB

- No

DATE:05/03/19
TIME: 13:15
PLACE: Room 4510, Zone E
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Jamie Beecroft

ATTENDEES:
Michael McMillan
Jamie Beecroft
Nicole Notarangelo

APOLOGIES FOR ABSENCE:
N/A

AGENDA:
- Meeting not held as not required.

MINUTES:
- Meeting not held as not required.

Date of next meeting: 12/03/2019 at 13:00

AOCB
- No

DATE:12/03/19
TIME: 13:15
PLACE: Room 4510, Zone E
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Jamie Beecroft

ATTENDEES:
Michael McMillan
Jamie Beecroft
Nicole Notarangelo

APOLOGIES FOR ABSENCE:
N/A

AGENDA:

- Website front end.
- Stripe Payment.
- Hardware (Pi Discussion)

MINUTES:

- The front end for the both the users and the administrators has been completed and has been uploaded to the web server for use. No changes were made to the backend other than bug fixing. All pages now have some form of user friendly display that would be acceptable for project completion.
- The stripe payment system has been fully implemented into the backend of the website.
- Additional work will need to begin on the Raspberry PI with further bug fixing taking place to ensure the PI can send images to the server without fail.

Date of next meeting: 19/03/2019 at 13:00

AOCB

- No

DATE: 19/03/19
TIME: 13:15
PLACE: Room 4510, Zone E
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Jamie Beecroft

ATTENDEES:
Michael McMillan
Jamie Beecroft
Nicole Notarangelo

APOLOGIES FOR ABSENCE:
N/A

AGENDA:

- Development of the PI.
- Backend Development.
- General point about group members.

MINUTES:

- Michael gave possession of the hardware to Nicole to allow her to make reasonable progress with the hardware of the system.
- The backend of the server-side development is around the point of completion with only minor bug fixing to do.
- It is to be assumed that only the three regularly attending group members are to remain on this course. Due to lack of involvement from the other group members they have been given no tasks to complete and will unlikely be given any if they return.

Date of next meeting: 26/03/2019 at 13:00

AOCB

- No

DATE:26/03/19
TIME: 13:15
PLACE: Room 4510, Zone E
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Jamie Beecroft

ATTENDEES:
Michael McMillan
Jamie Beecroft
Nicole Notarangelo

APOLOGIES FOR ABSENCE:
N/A

AGENDA:
- Development of the PI.

MINUTES:
- It has been decided by the group and approved by the client that the Raspberry PI (hardware) will no longer send whole images of the registration plates to the server-side system. This will sacrifice on evidential pictures in the event of a dispute however will be extremely beneficial in sending and response times on the network load.
  Additionally, the API for the Open ALPR has been implemented in order to allow the PI to have the capability to read number plates and then send them to the system.

Date of next meeting: 02/04/2019 at 13:00

AOCB
- No

DATE: 02/04/19
TIME: 13:15
PLACE: Room 4510, Zone E
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Jamie Beecroft

ATTENDEES:
Michael McMillan
Jamie Beecroft
Nicole Notarangelo

APOLOGIES FOR ABSENCE:
N/A

AGENDA:

- Unaware during the previous meeting that the group was set to be on annual leave. Meeting was not held because of this.

MINUTES:

- Unaware during the previous meeting that the group was set to be on annual leave. Meeting was not held because of this.

Date of next meeting: As Required.

AOCB

- No

DATE: 11/04/19
TIME: 12:00
PLACE: Abertay University, Various Places
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Jamie Beecroft

ATTENDEES:
Michael McMillan
Jamie Beecroft
Nicole Notarangelo

APOLOGIES FOR ABSENCE:
N/A

AGENDA:

- Introduction
- Hardware
- API
- Website
- Database
- Testing

MINUTES:

- Meeting was required for various points of discussion
- Device works as intended however there is a bug with the capturing of images, this was quickly rectified with some changes t the script that communicates with the OpenALPR system.
  It was decided to change the way the device sends data to the API. The API will now handle the entry/exit commands and return the result to the device.
- The API has been changed to allow the implementation of the above and has been pushed to the web server to be used as the active API.
- Website is working as intended and no work is required on it for now.
- Database is working as intended and is populated with test data.
- Testing will be conducted next week once more issues have been iron out with the system. Partial testing has already been done on the backend of the system and on the website.

Date of next meeting: 16/04/2019

AOCB

- No

DATE: 16/04/19
TIME: 13:00
PLACE: Room 4510, Zone E
MEETING CHAIR: Jamie Beecroft
MINUTE TAKER: Jamie Beecroft

ATTENDEES:
Michael McMillan
Jamie Beecroft
Nicole Notarangelo

APOLOGIES FOR ABSENCE:
N/A

AGENDA:
- Update
- Hardware
- Testing
- White paper
- Presentation
- Submission
- Individual Contribution

MINUTES:
- The entire system is working as intended on an initial inspection, no extensive testing has been done.
- The hardware has mostly been fixed however needs tested with a real number plate to identify if the issues are still present when used in a realistic scenario.
- Extensive testing will be done this week on all components of the system and any issues identified should be rectified before submission.
- The white paper will need to be finished and finalised and ensure that all group members are happy with what will be submitted.
- The presentation will need to be made and discussed who will be taking what responsibilities during the presentation.
- Michael will be submitting the final project.
- Everyone is reminded that they will need to submit their own sections of individual contributions at the completion of the project.

Date of next meeting: Not required, completion of project.

AOCB
- No

9.5 Appendix E – User Manual

Smart Parking Ltd

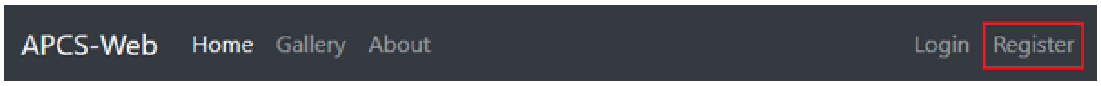
User Manual

Operating in our car parks

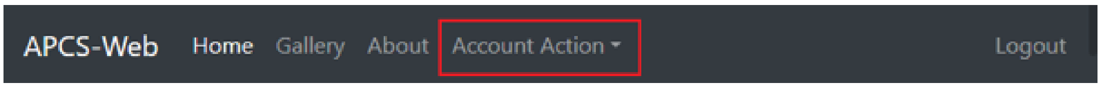
Using our car parks is easy and hassle-free. All you need to do is register an account with us on our website and then approach the entrance at any one of our car parks. Upon approaching the barrier, your vehicle's registration number will be read before allowing you to conveniently park without even having to open your window!

Registering with us

First, visit https://mayar.abertay.ac.uk/~cmp311gc1801/ and register an account. Make sure your email is valid.

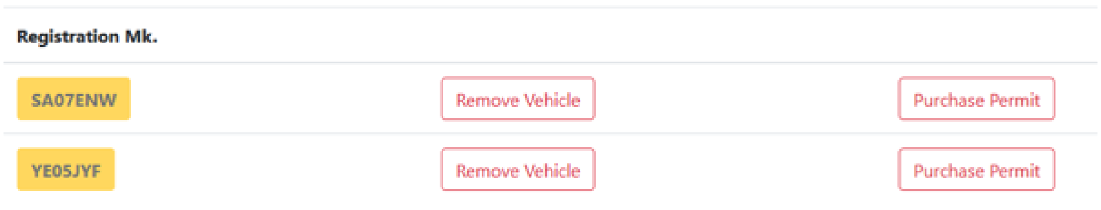


Next, login to your account and locate the "Account Action" drop-down menu on the navigation bar. Click on "Vehicles".



Here you will be presented with a form in which you can enter your vehicle's registration number that you wish to use our car parks.

Once you have a vehicle registered to Smart Parking Ltd, you can use any of our car parks! You will be charged an hourly rate for using our car parks with the bill being sent to you after you leave the car park.



Registration Mk.		
SA07ENW	Remove Vehicle	Purchase Permit
YE05JYF	Remove Vehicle	Purchase Permit

Season permits

If you are a regular customer of ours then you will benefit greatly from buying a season permit! With a season permit, you are given unlimited access to each one of our car parks for a small monthly fee.

If you'd like to buy a permit, simply locate your vehicles page on our website. From there, you will see the "Purchase Permit" button that will take you to a payment page where you can pay by card through the trusted and secure Stripe payments service.

9.6 Appendix F – Web Application Url

The web application can be found at this link:

https://mayar.abertay.ac.uk/~cmp311gc1801/