RMIT UNIVERSITY

Technical Solution Design for DriverCity(CarShare)

Version: V1.0
Date: 29.04.19
Sponsor: Mr Homy
Author: Ahnaf Shahriar Abir, Khatina Haidari, Mahima Shrestha

Document Control

Distribution

Version	Issued	Recipient	Position
V 1.0	29.06.19	Mahima	Team member

Amendment History

Section	Page	Version	Comment
Known Issues and Risks	15	1	More risks and issues needed to e added
Functionality	7	1	Flowchart needed to be made

.

Staff or Entities Consulted

NAME	Position / Organization
Ahnaf Shahriar Abir Khatina Haidari Mahima Shrestha	Scrum master Team member Team member

Related Documents

Name	Author	Description
Development guide	Ahnaf	https://drive.google.com/open?id=1IcH5zpjgVPpy6OGuaNJ6r7Ne2lsBNhJ6

Preface

The purpose of this document is to show the technicality behind the web framework.

Table of Contents

1 Introduction

Car Share is when people rent cars for a short period of time. DriverCity is a web build project for the company that is running this car share scheme. This scheme, DriverCity, was developed to attract different types of customers such as tourists, families on holiday, people who are new to the city, friends or people who cannot afford to buy a car yet. DriverCity lets people login or register their details, see their location and the cars location near them in a map, see different ways of getting to the car destination, see what type of car is available near them and the price.

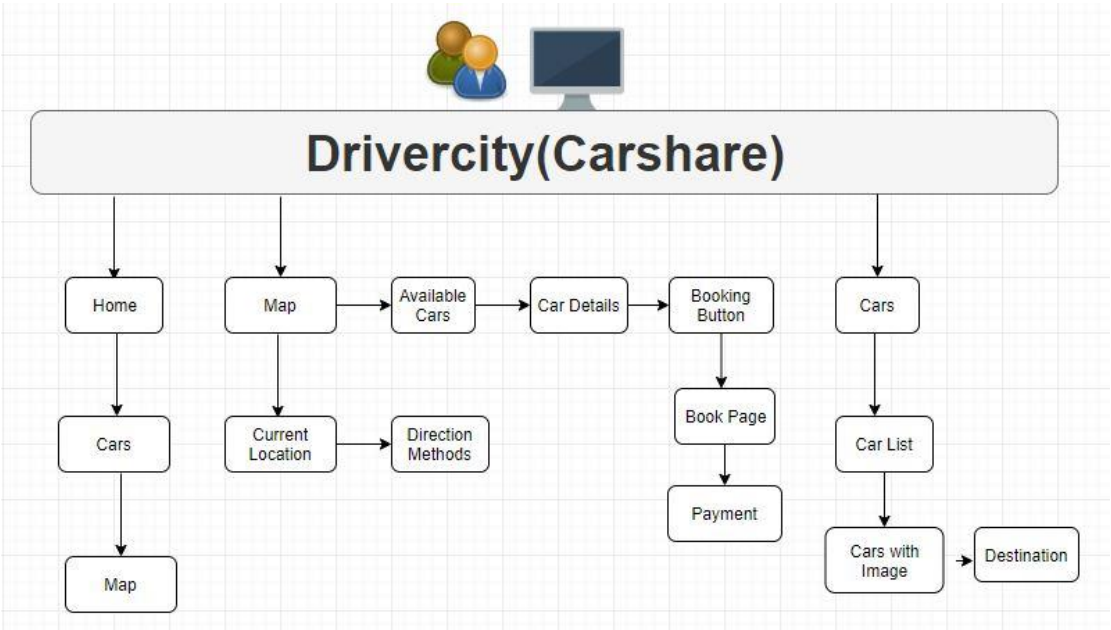
Various software was used for the development of the car share project, DriverCity. For the framework Laravel was used. Version Control, GitHub, was used for the sharing of the codes, Google cloud, MySQL, phpMyAdmin was used for the database of this project. The complexity level was high for the team as the team had to learn to use the software while they were developing the project. Although saying that Laravel was the best choice for the team to use as it had some already implemented codes for register and login which helped ease the process of the project for the team.

2 Technical Environment

The team brought their laptops which had all the information on the project plus could be carried and used anywhere they wanted. The project needed to be done where there was a high level of network connection as some software that was used by the team needed internet access. Message-enabled application were also needed for the team to be able to communicate with one another. Back-up systems were also in place. For the back-up USB Flash Drives were also used. Laravel php web framework was used for the implementation of the codes of the project. Laravel was used as it had some codes already implemented which helped ease the process of the project. GitHub was used for version control as it provides features of bug on the codes and task management.

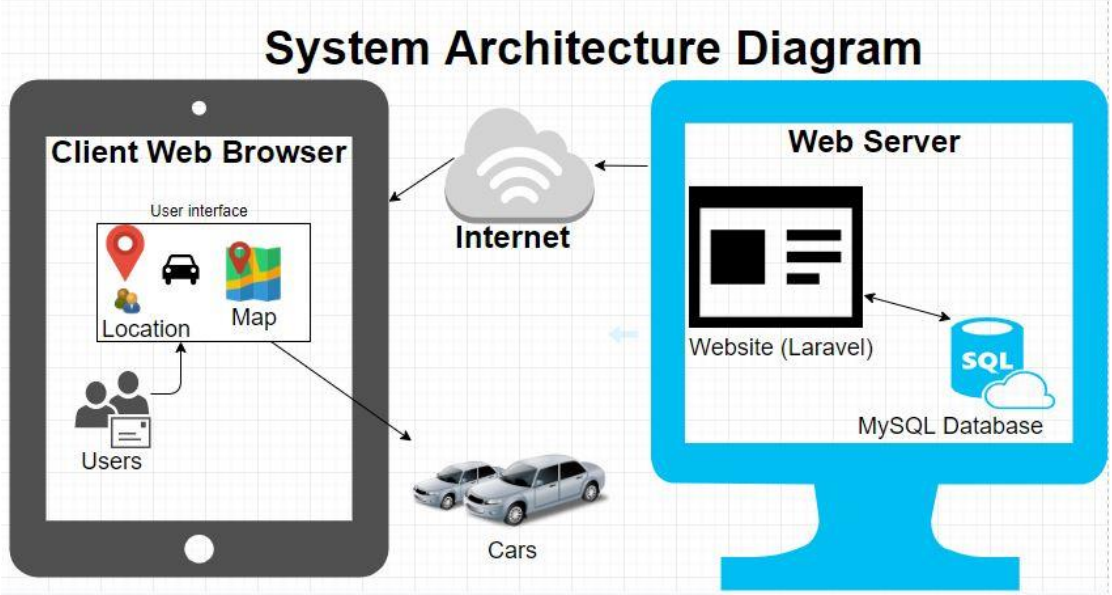
3 Overall Architecture

The architecture diagram below shows the details of what the website will contain. It gives the detailed version of what the website will look like and what each page in the website will contain and be used for. The main menu option will have homes, cars and map page. Then map page will contain the location, direction methods, available cars, booking option. Then after the booking, there will e a payment option. The car page will contain the list of individual cars with their images.



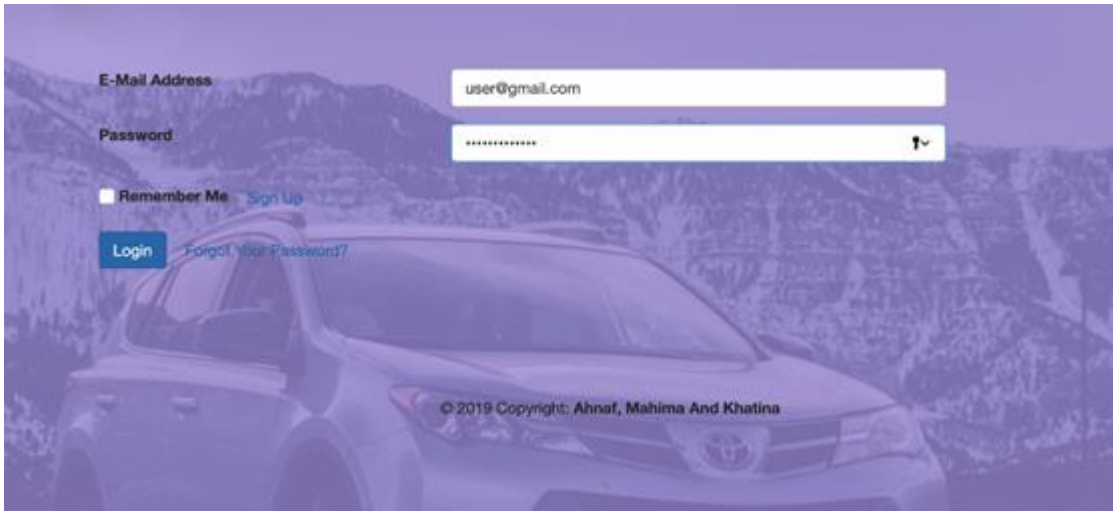
4 System Architecture

Software architecture is a blueprint of the system. The diagram below describes how the website will operate on the client and user side. The website used laravel and database for storing the details of users and cars. The website will need internet to work then the client and users can see the interface that has the map that will have all the details of the car for the users to see, the location of the car plus their own will also be visible in the interface for the users and client to look at.

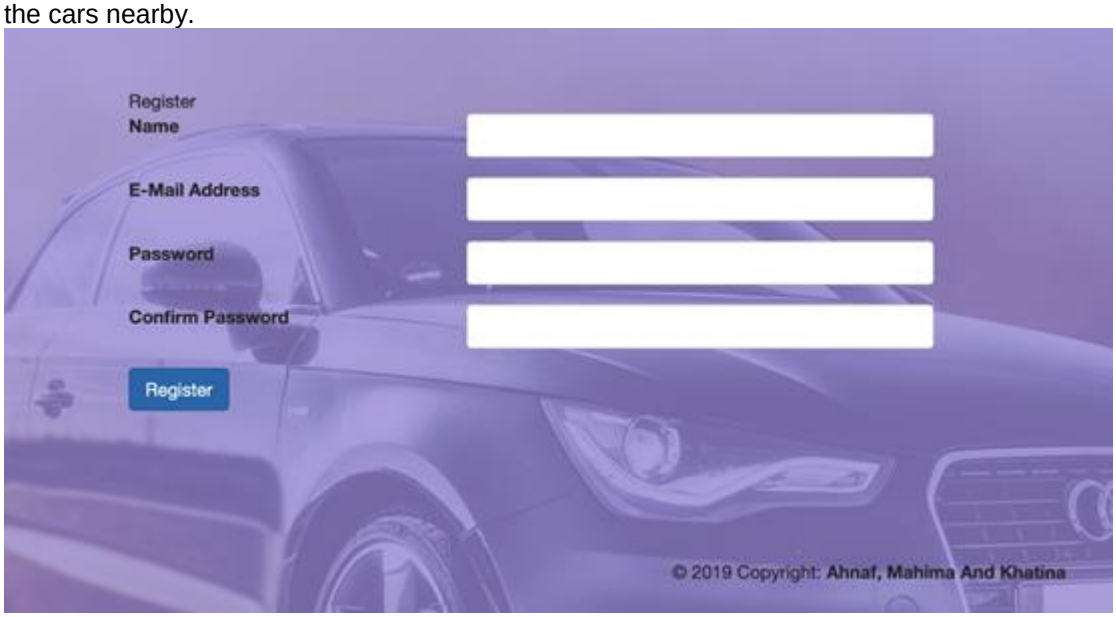


4.1 FUNCTIONALITIES/FEATURES

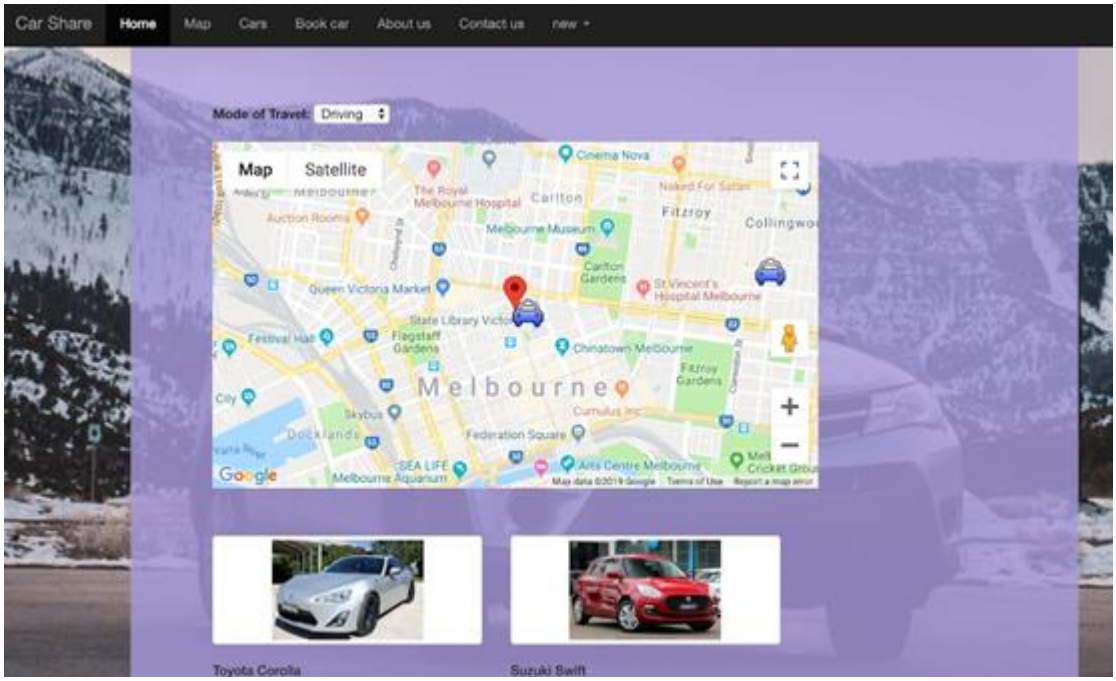
At first the user is asked to provide with credentials, If the user is already registered, he can login with his credentials.
Else, users can create a new account by clicking into sign up button from login page.



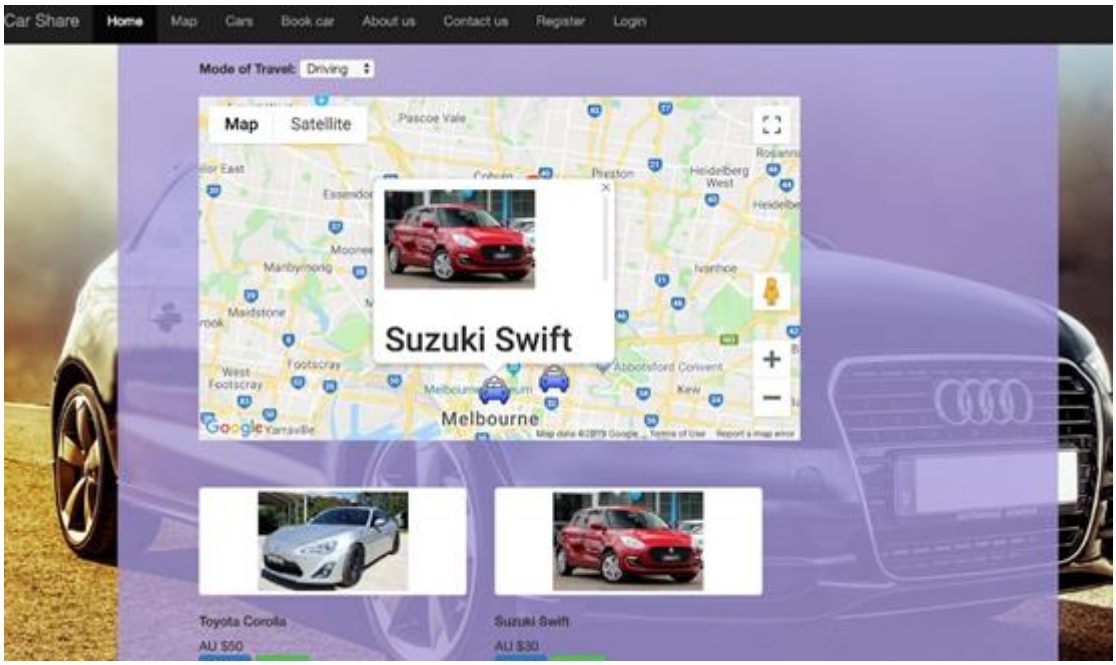
The user account is created when the user enters required fields with validation successful. The user is then redirected to the homepage where he can allow current location to see all the cars nearby.

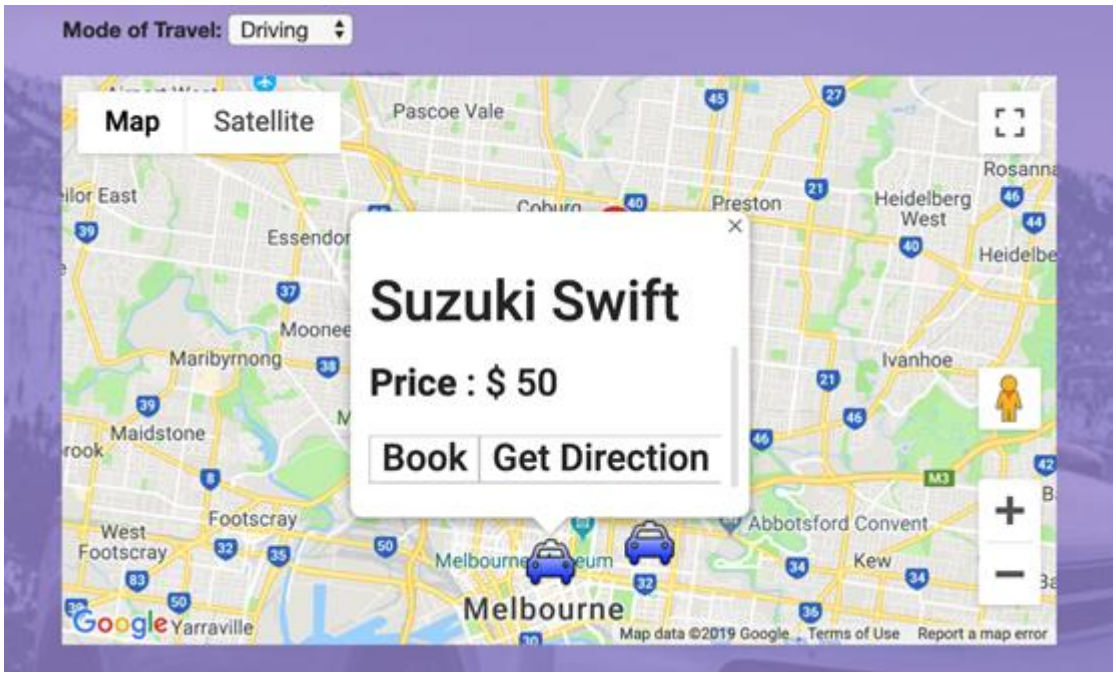


Car details are shown under the map.

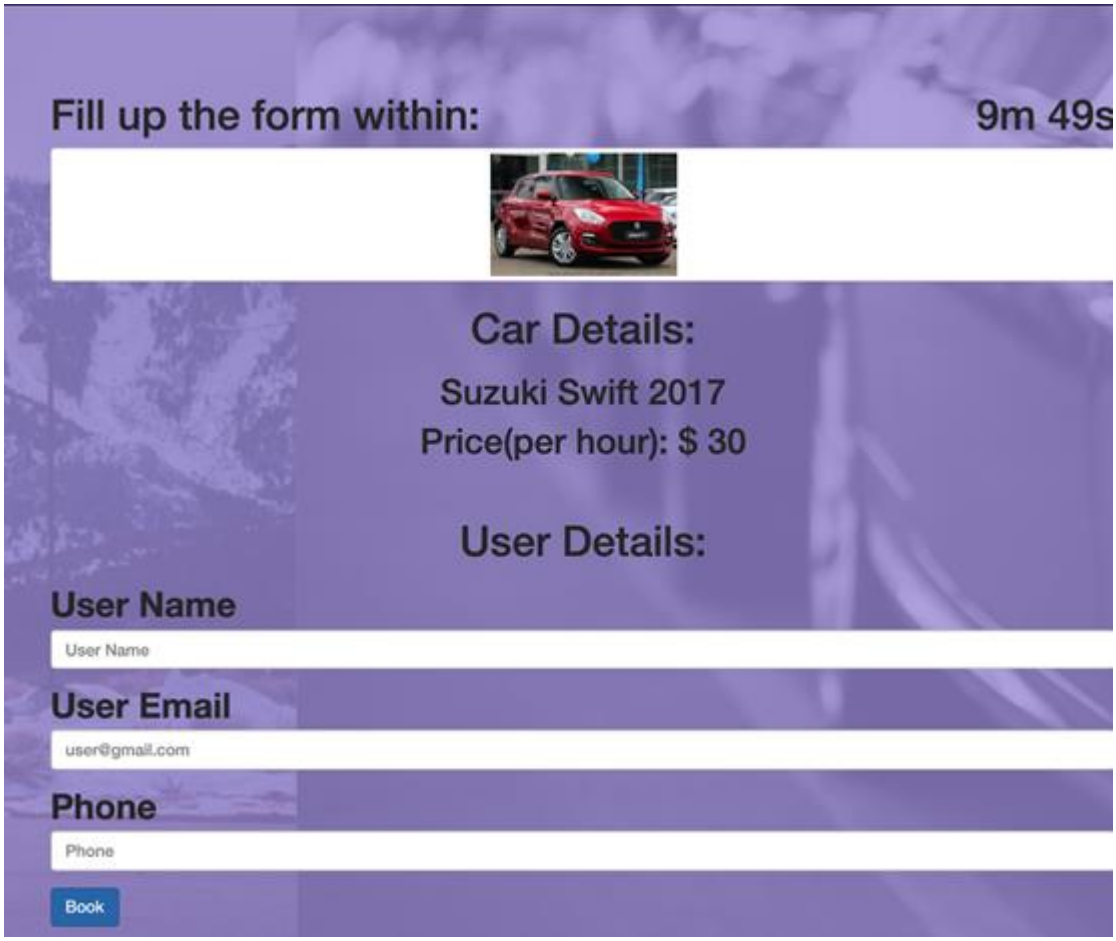


When the user selects any car icon, a popup window shows the picture of the car with model, price. The user can select to book the car from that point or get directions to that car which will show the way from user's location to the car's location. The user can choose the mode of travel too, he can choose either, by walking, bicycling or in transit.

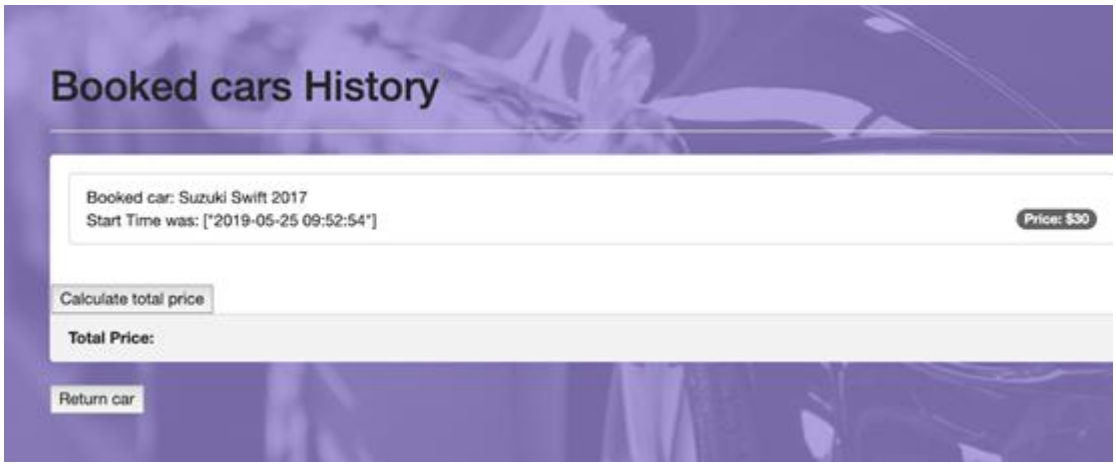




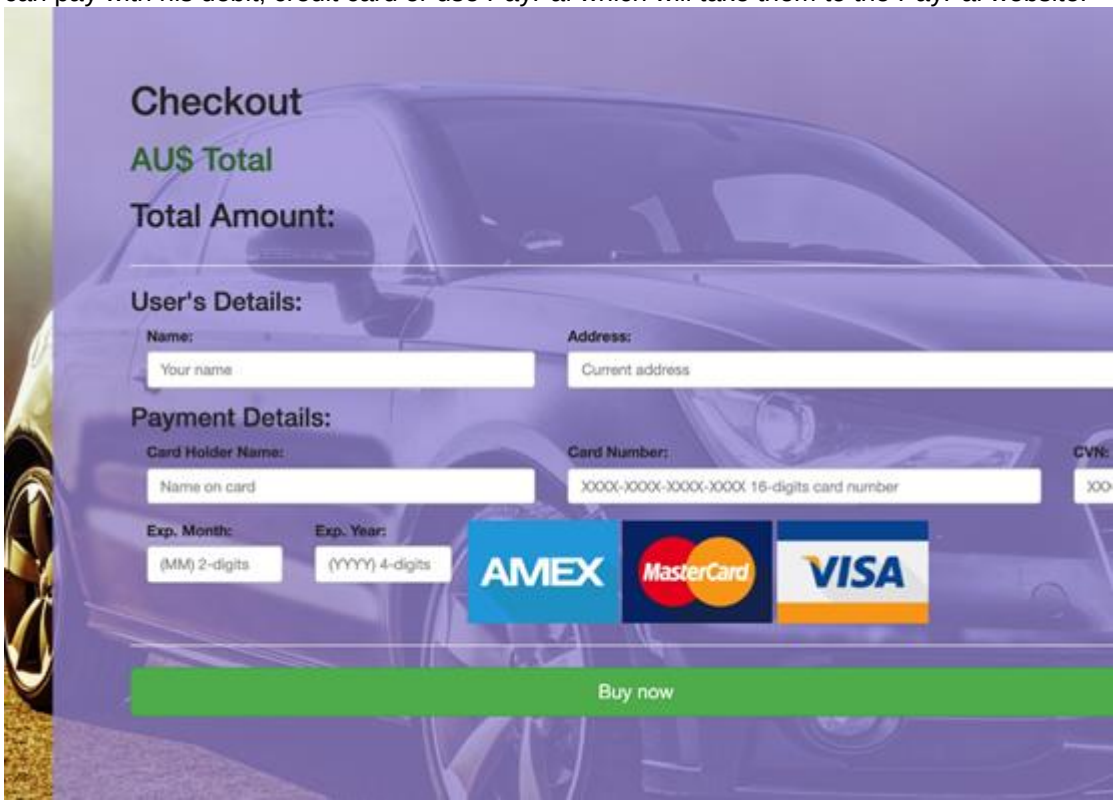
The user is redirected to this page when book is clicked. Then the user is given a 10 mins of time to fill up the form. To ease out user's hassle, logged in user's email and name will be automatically fetched to the form.



Once the book was done and the user wants to return the car, he can see his booking history by going to history option.



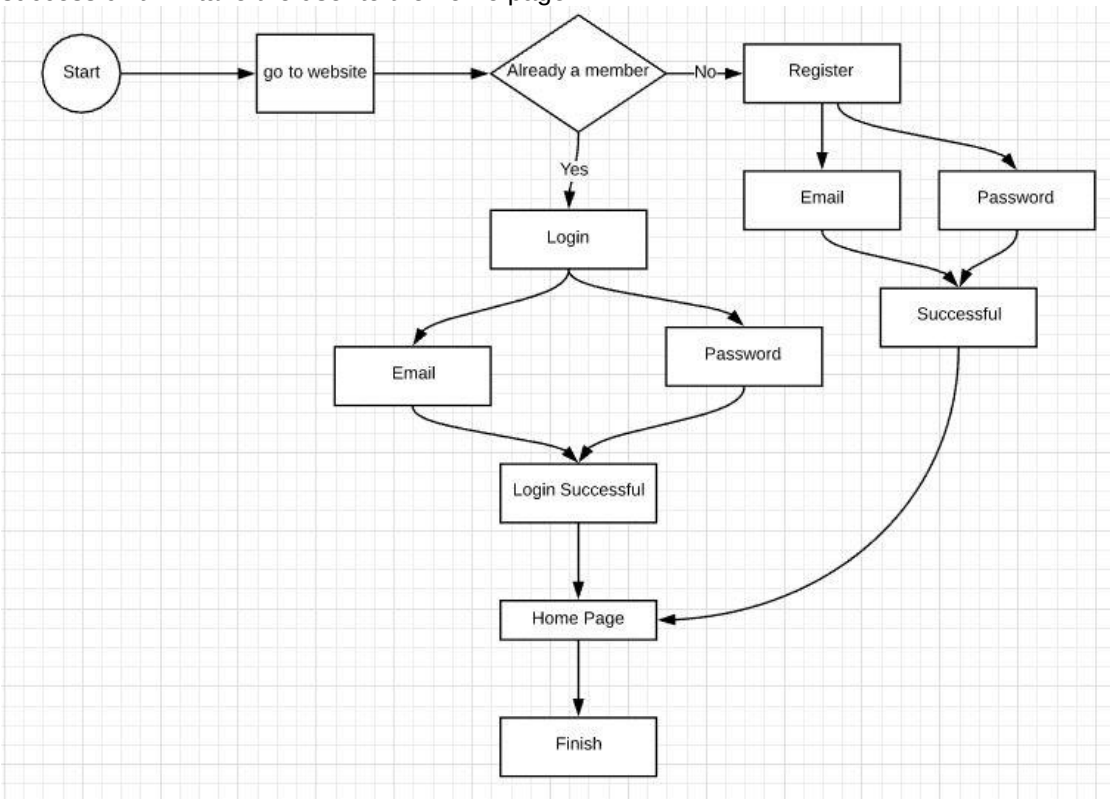
After the total amount is calculated, the user is redirected to the payment page where the user can pay with his debit, credit card or use PayPal which will take them to the PayPal website.



4.1.1 Functionality 1

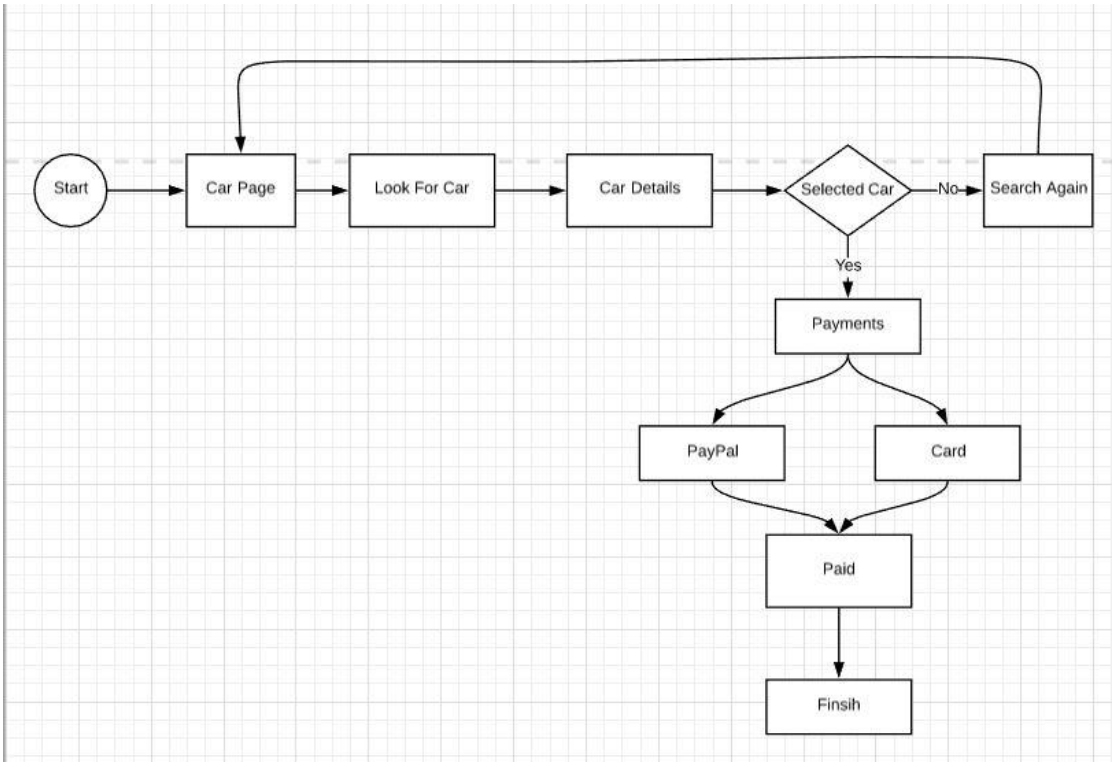
Login and Register functionality:

The User will go to the Website and login with their credentials, if the user does not have login details, they will need to register their email and password and if it is successful it will take the user to the home page.



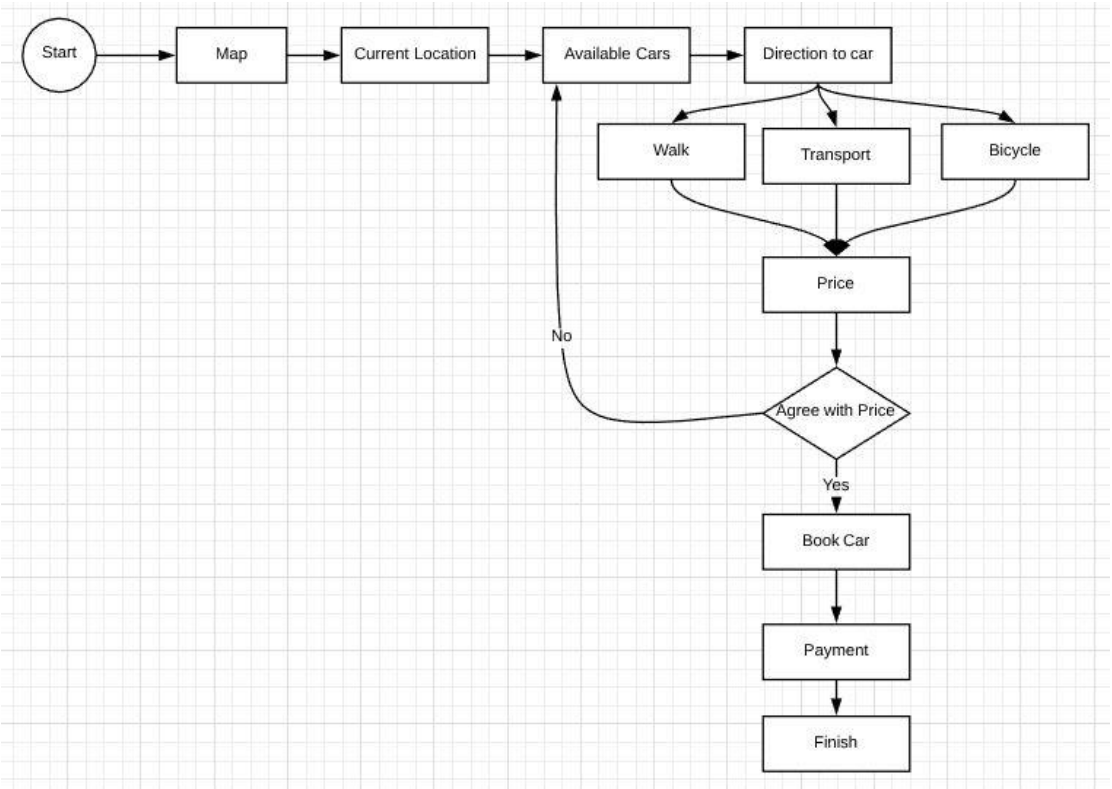
4.1.2 Functionality 2

Payment and book Car:
The user can look for car with its information such as model, make and year. Once the user selects a car they will proceed to the payment section where they can pay either with their debit/credit card or through PayPal.



4.1.3 Functionality 3

Map Page:
In the map page, users can see their current location and the location of the car. They can also see the available cars and the method of getting to the car by walk, transport or bicycle. The user can see the car price and if they agree with the price of the car selected by them they can start booking the car and proceed to the payment section if they are not happy with the price of their selected car then they can go back to the available car page and re-select the car.



5 Database Architecture

Database schema may help keeping track of information that is present or not in the database and help maintain it for later. It describes what data is stored in database, the entity and the relationship among the data. Internal level/storage which is a physical representation of the database. It describes how the data is stored in database and covers the data structure plus the file organisation. It gives the team a view of the structure and the relationship of the tables in the database.

The team created a database which contains two tables one for users which stores all of user's details when they register. Another table was created for the car data. Several data were added for the car as well as the images of each car were added. At first the team used phpMyAdmin for their database creation but later realised that this can only be used on the local device and cannot be shared with team members. Then the team moved to using google cloud SQL for the database which allowed the team to share the database and work in it together.

6 Implementation Instructions

Database used: Google MySql

Link: ----------Link _____----------

1. Create Cloud SQL instance:
    a. Go to the Cloud SQL Instances page in Google Cloud Platform console
    b. Click Create instance
    c. Select MySQL and click Next
    d. Click Second Generation
    e. Enter a name for the instance. (This will be used later)
    f. Create a new user (if don't want to use root)
    g. Enter password

h. Choose Automatic backup on. This will backup the database at a particular time regularly. This ensures that there's always a backup of the database and it's updated one.
i. Click Create

2. Create Database from the DATABASES tab.

3. Connect to database using CLOUD SQL PROXY

a. Enable the API
b. Install Proxy client on local machine

```
curl -o cloud_sql_proxy
https://dl.google.com/cloudsql/cloud_sql_proxy.darwin.amd64
```

c. `./cloud_sql_proxy -instances=<INSTANCE_CONNECTION_NAME>=tcp:3306`

7 Non-functional specifications

- Accessibility: The website is available 24/7 from anywhere across the globe

- Efficiency: The system should produce relevant outputs depending on user input.

- Easy to use: Users of all ages and web experience should be able to use the website without facing any difficulties

- Security: Use of payment gateways to protect customer data.

- Low perceived workload: The amount of attempts that would be needed by the users to accomplish a particular task.

-Intuitiveness:How easy it is for users to understand the interfaces, buttons, headings, and more.

8 Summary of test results

<Provide a summary of the test cases and results in tabular format.>

9 Known Issues & Risks

Some risks and issues that the user or the website may face could be hacking of the website and the payment card details being stolen. The website speed may get slow, taking time to load the pages. Outdated website design. Issues the team faced when building this website were technical issue, where the laptop were updating and the team member had to open all their work and write all the commands to open software. Some risks that the team could have faced could be: Software failure would have a risk that the team could have faced when building this website, Lack of communication, delays on project or even scope creep.

10 Appendix

<Refer the tool that is used to capture the functional requirement and if possible provide the references to the tool and also a summary of the functional requirement. It can be simply compilation or copy of the user stories from JIRA/Rally.>

<NOTE: These headings are guidelines only. Based on your project, you might require additional headings, so feel free to add headings as required.>

11 References

d.

Cloud - https://cloud.google.com/sql/docs/mysql/connect-external-app#proxy
Cloud database - https://cloud.google.com/sql/docs/mysql/create-manage-databases
Issues- https://www.proofhub.com/articles/project-management-challenges
Database Architecture- https://medium.com/oceanize-geeks/concepts-of-database-architecture-dfdc558a93e4
https://www.techopedia.com/definition/30601/database-schema
Flowchart - https://www.lucidchart.com/pages/landing/free-flowchart-maker?utm_source=bing&utm_medium=cpc&utm_campaign=flowchart_maker_australia&msclkid=e1a757fc4584160074180aeaa5380033
Architecture-http://sungsoo.github.io/2014/04/28/overall-architecture-and-design-philosophy.html
https://www.tutorialspoint.com/software_architecture_design/introduction.htm