
UNIVERSITY of LIMERICK

OLLSCOIL LUIMNIGH

Department of Electronic and Computer Engineering

Interim Report

# Electric Vehicle Charging Crowdsourcing Framework


| | |
|---|---|
| Student Name: | **Colin Hanily** |
| Student ID: | **14159031** |
| Supervisor: | **Thomas Conway** |
| Course: | **Electronic and Computer Engineering** |
| Academic Year | **2017/2018** |

# Abstract

# Introduction

Electric vehicles are becoming more and more common as technology continues to progress. Over the last decade the prominence of electric vehicles has slowly increased. Although they are still but a small portion of the market, it is hard to ignore their growing prominence.

The aim of the project is to prepare for this growing usage with the vision that in years to come, electric vehicles will dominate the market. Electric vehicles are unique in that they can be charged from the pre-existing grid. This gives owners of charging points in places such as home and offices the ability to profit from increased electric vehicle usage by renting out there charging points to electric vehicle owners. As opposed to fossil fuel powered vehicles which require large stores of petrol/diesel etc. which would be hazardous for consumers themselves to store. This fresh view of vehicle fuelling which comes with electric cars could change the market as we know it in years to come

The project aims to create a framework which can be used by electric vehicle owners and charging point owners alike so that both can benefit. This will come in mobile app form so that a driver will be able to rent out a private charging point and the owner will be able to profit from said charging points' usage. The aim is to get the basic functionality of the app working first so that a simulation of charge point selection and renting integrated with google maps is built first. I then aim to expand onto as many useful app features as possible such as charge point rating systems, map filters, charge point searching, in app payment system, charge point and user usage stats etc.

# Project Aims

1. Create a mobile application mapping out all private charging points which can be used, allowing users to rent out these points for personal usage.
2. Create a database/webserver which will be able to store user/charging point data.
3. Link the database and mobile application together so that they can work in tandem.

4. If the above goals are achieved, begin to add in extra features such as a charge point rating system, in app payments, route planning and possibly link the mobile app and database to microcontroller which will simulate a plug so that app users will get real-time feedback as to which charging points are being used.

## Similar Systems

Many systems similar to the one designed in this project. As the main objective of the project is to develop a mobile application for electric vehicle charger renting, many systems which involve renting come to mind. For example, Airbnb.

Airbnb is an online marketplace which lets people rent out their properties or spare rooms to guests. Airbnb takes 3% commission of every booking from hosts, and between 6% and 12% from guests [61].

There's plenty of criteria to list for/search a property: from a shared room to an entire house, to having a swimming pool to having a washing machine. There are photos of the property, and the hosts/guests, with full map listings [61]. The Airbnb framework utilises MySQL as its main database, this was taken into account when taking into consideration which database to use for this project [62].

Many other apps which are similar to Airbnb but applied to a different field were also analysed when taking into consideration the best approach for the project, for example, Uber which allows registered drivers to act as payed drivers [64] and JustPark which allows user to rent out there parking space [63].

All of these applications have a similar goal, which is achieved in a different way. This diversity helped a lot during the design and implementation stages.

Many mobile applications also exist for electronic cars such as NextCharge and Plugshare, Both of which allow the reviewing and rating of charge points but have no charge point booking integrated as of yet. User can "check in" at charge points to show that they have used them or been present at them. The function of this is to allow other user to know the operational state of the charge point. Plugshare allows user to put their home chargers on the database, but again these charge points cannot be rented as of yet but are available for free.

As the electric vehicle industry progresses, these charge points will become more in demand. This project aims to prepare for this growing demand by making charge points bookable/rentable as a proof of concept.


*Figure 1:PlugShare UI [65]*

## Analytical Background

The first step taken in analysing was the project was the approach which would be taken and the feasibility of the project. Inspiration was taken from similar apps such as MyTaxi, AirBnb and PlugShare. These apps were seen as a proof of concept that the project was indeed feasible. Although they touched on similar topics. Concepts from each were taken and remoulded to be suitable for the projects use case.

In terms of designing the framework, the first step was to choose a suitable mobile application platform to design the app. The problem with this is that in most cases, once a platform has been chosen, the app will only run on the operating system for which it was designed for e.g. IOS or Android. To overcome this problem, the ionic framework was chosen.

Ionic is an open source SDK released under a permissive MIT license. This means you can use Ionic in your own personal or commercial projects for free. It is built on top of AngularJS and Apache Cordova. It provides tools and for this purpose using web technologies such as CSS HTML5 and Sass [1]. The framework allows for the development of hybrid apps which can be deployed on Android, iOS or Windows platforms

Previous experience with Ionic provided confidence in the suitability of the framework, although a project of this scale had not been attempted before. Therefore, Ionic was the client side framework of choice, with the intention of switching to Android studio and developing for the android framework if roadblocks were encountered with Ionic.

Android development was chosen over IOS development as a fallback for due to past experience which would speed up the development process.

## Client Side

### Ionic Framework

As previously discussed, the Ionic framework is built on top of AngularJS and Apache Cordova, the latter enabling the development of hybrid applications.

**Apache Cordova**, formerly known as PhoneGap enables applications for mobile devices using CSS3, HTML5, and JavaScript to be built instead of relying on platform-specific APIs like those in Android, iOS, or Windows Phone [4]. It enables wrapping up of CSS, HTML, and JavaScript code depending upon the platform of the device. It extends the features of HTML and JavaScript to work with the device. The resulting applications are hybrid, meaning that they are neither truly native mobile application (because all layout rendering is done via Web views instead of the platform's native UI framework) nor purely Web-based (because they are not just Web apps, but are packaged as apps for distribution and have access to native device APIs) [26].


*Figure 2: Hybrid Application Development [2]*

**HTML** stands for hype text mark-up language. It describes the structure of web pages using mark-up language. Every webpage is written in HTML so therefore this language is an integral part of the project [45]. Web browsers receive HTML documents from a web server or local storage and renders them into multimedia web pages. HTML can embed programs written in a scripting language such as JavaScript which affect the behaviour of the content of web pages [46]. In this framework, HTML5 will be used.

**Sass** (syntactically awesome stylesheets) is a style sheet language used by the ionic framework. It is a scripting language that is interpreted or compiled into CSS. Sass Script is the scripting language itself [11]. This language, in conjunction with CSS is responsible for perfecting the look and design of each page of the application.

**CSS** (cascading style sheets) is used for describing the presentation of a document written in mark-up language [12]. It is most often used to set the visual style of web pages and user interface. CSS is designed primarily to enable the separation of presentation and content, including the layout colours and fonts [43]. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, and enable multiple HTML pages to share formatting by specifying the relevant .css file, which

reduces complexity and repetition in the structural content [44]. To get the look and feel of the app correct, this language will be very useful in conjunction with Sass.

**JavaScript** is used in conjunction with HTML for a lot of the functionality of the app. It is a high-level, dynamic, weakly typed, object based, multi-paradigm and interpreted programming language [47]. A large majority of websites employ it to make webpages interactive and provide online programs. All modern web browsers support it without the need for plug-ins by means of a built-in JavaScript engine. JavaScript supports event driven, functional and imperative programming styles. It does not include any I/O such as networking, storage or graphics facilities as these features rely on the environment in which it is embedded [48]. For example the google maps JavaScript API will be used in order to integrate google maps, a key part of the apps uses. A lot of the core functionality of the app will be written in typescript.

**TypeScript** is an open-source programming language developed and maintained by Microsoft [29]. It is a strict syntactical superset of JavaScript, and adds optional static typing to the language [30]. TypeScript supports definition files that can contain type information of existing JavaScript libraries, much like C++ header files can describe the structure of existing object files. This enables other programs to use the values defined in the files as if they were statically typed TypeScript entities. TypeScript headers for the Node.js basic modules are also available, allowing development of Node.js programs within TypeScript [31]. An official extension allows Visual Studio 2012 to support TypeScript. This is especially useful as Visual Studio Code is the IDE of choice for the project [32]. As TypeScript is a superset of JavaScript, on compilation, it is compiled to JavaScript [43]. Therefore, JavaScript and Typescript are used interchangeably throughout the project.

Although the ionic framework is built on top of Apache Cordova and uses AngularJS and Node.js for a lot of its core functionality [1], [27]. HTML5 dictates the content, structure [7] and alongside JavaScript/TypeScript the functionality of the app whereas CSS and Sass modifies the design and display of HTML elements [11], [12].

## Ionic Framework Layout

Ionic has a unique structure when it comes to the file structure during development.

Each page of the app is assigned its own folder. Inside each folder are the files that make up the structure and function of the pages elements i.e. for this project, .html, .scss and .ts files.

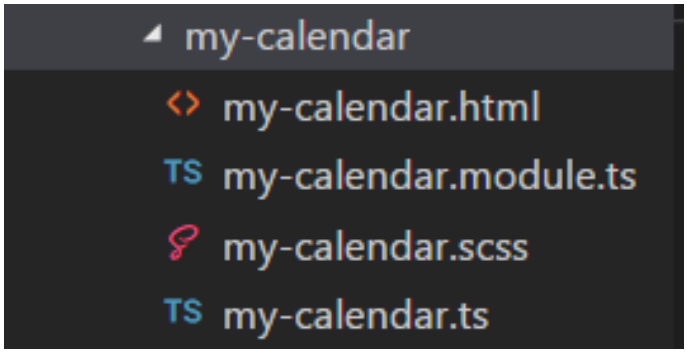



*Figure 3: Ionic Page Folder Layout*

Page folders may also contain a module.ts file which declares the page so that it can be used in conjunction with other pages.

These pages are themselves stored in a folder named “src”, in which, the majority of files the developer will need in app development are kept e.g. index.html in which, scripts and can be imported for API’s such as the google maps API or the theme folder which defines global variables for the apps overall theme and colour scheme. Other folders for service providers etc. can be created as necessary.

The src folder also contains an app folder, in which pages all pages, providers and services are declared.

Folders outside of the ones named above, such as “npm_modules” where libraries were installed, were used sparingly throughout the project.

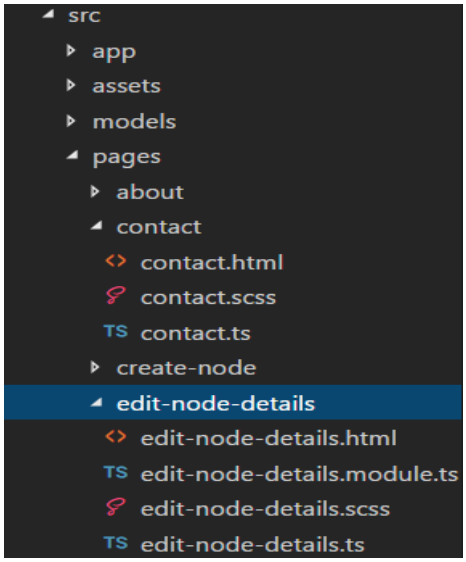



*Figure 4: Ionic src Folder Layout*

## Charge Point and User Calendar Plugin

In order to make the mobile Application easier to follow and understand, it was apparent that a user interface for charge point booking would need to be provided. It was found that the process of creating a custom user calendar from scratch, although more flexible, would take quite a lot of time and not allow for the completion of the overall project. However many plugins were found to be provided by both Ionic in documentation as well as third parties on GitHub. Many of these had drawbacks such as only allowing access to the user phone calendar, or not allowing the user to see a more refined view other than a monthly view such as the one below:


*Figure 5Date Picker Plugin for Ionic [60]*

The applications calendar would need to provide an hourly view of charge point bookings so as to make booking a simple process. After quite some research, a suitable calendar plugin was found on GitHub, developed by a user called ‘twinssbc’ [67]. The plugin allows for month, week and day views of the calendar and also for events to be implemented, this was suitable for calendar bookings. The plugin also has quite an in depth API, is easily customisable as well and question on the plugin are answered swiftly by twinssbc [68].

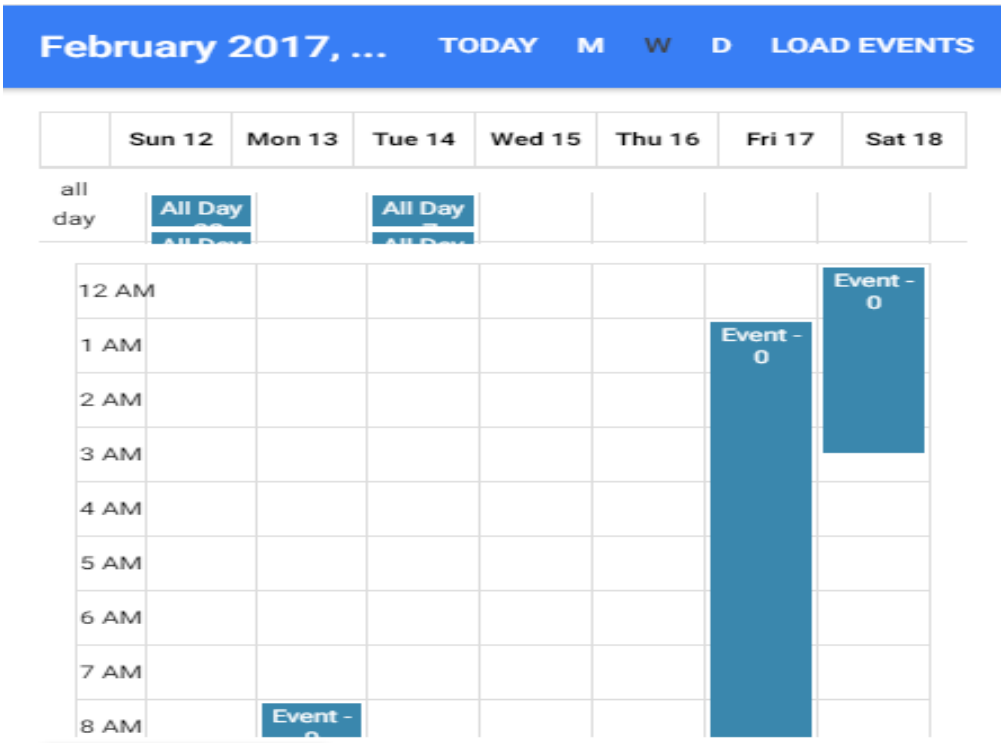



*Figure 6: twinssbc Ionic Calendar Plugin [69]*

## Google Maps JavaScript API

Use of the Google maps JavaScript application programming interface is an integral part of the project. In order to visualise much of the data used and integrate a map component into the application, google maps JavaScript API was used. The APIs’ documentation provides clear instructions on the use of its many functions [28].

Although ionic has a supporting Apache Cordova google maps plugin, the google maps JavaScript API was chosen for use due to large amounts of documentation, making roadblocks easier to overcome. Although as described by the author of the google maps native plugin for ionic on the ionic forum, the google maps plugin is faster but more difficult to code than the

JavaScript API [33]. Given the relatively short time for project work, my beginner knowledge as a JavaScript developer and facts previously stated, the JavaScript API was chosen for use in the project. Once the application had been completed and tested, the native plugin could be swapped to in to replace the JavaScript API to better performance speed if time allowed.

### Development Environment

The IDE chosen for this project was the Visual Studio Code editor. This was chosen for a number of reasons. Many other IDEs were tested such as Visual Studio, NetBeans and Atom. For simplicity's and to keep budget spending's to a minimum, a free IDE was preferred. Visual Studio code was chosen over Visual studio as it is a more lightweight text editor solution and from personal testing is much easier and faster to work with when compared to Visual Studio and NetBeans for Ionic development. Visual Studio Code has many ionic supporting plugins for file extensions as well as the IntelliSense feature which allows for code completion, parameter info, quick info, and member lists which allow for faster development [34]. From reading many forum threads on the ionic forum [36], as well as following YouTube tutorials provided by professionals such as Josh Morony, it could be seen that Visual Studio Code Seemed to be the editor of choice for many. Visual Studio code is also on the top of a list of recommended IDEs and Editors [35].

## Server Side

The server side of the framework will be used to store user data including users' profiles, charge points bookings and charge point information. For the development stages of the application, this server was hosted on the same computer which client side development was taking place to increase the fluidity of testing.

### Database

Analysis was firstly taken to decide what database management system would be used. This helped to narrow down the two most prominent choices to:

1. MySQL
2. MongoDB

**MongoDB** is a free and open-source cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schemas [38].

**MySQL** is a freely available open source relational database management system (RDBMS) that uses Structured Query Language (SQL). SQL is the most popular language for adding, accessing and managing content in a database and MySQL is one of the most popular management systems that utilises it [4].

MongoDB and MySQL were then compared using the table below

| Date | MySQL | MongoDB |
|---|---|---|
| Written in | C++, C | C++, C and JavaScript |
| Type | RDBMS | Document-oriented |
| Main points | - Table<br>- Row<br>- Column | - Collection<br>- Document<br>- Field |
| License | GPL v2 / Commercial licenses available OD | GNU AGPL v3.0 / Commercial licenses available OD |
| Schemas | Strict | Dynamic |
| Scaling | Vertically | Horizontally |
| Key features | - Full-text searching and indexing<br>- Integrated replication support<br>- Triggers<br>- SubSELECTs<br>- Query caching<br>- SSL support<br>- Unicode support<br>- Different storage engines with various performance characteristics | - Auto-sharding<br>- Native replication<br>- In-memory speed<br>- Embedded data models support<br>- Comprehensive secondary indexes<br>- Rich query language support<br>- Various storage engines support |
| Best used for | - Data structure fits for tables and rows<br>- Strong dependence on multi-row transactions<br>- Frequent updates and modifications of large volume of records<br>- Relatively small datasets | - High write loads<br>- Unstable schema<br>- Your DB is set to grow big<br>- Data is location based<br>- HA (high availability) in unstable environment is required<br>- No database administrators (DBAs) |
| Examples | NASA, US Navy, Bank of Finland, UCR, Walmart, Sony, S2 Security Corporation, Telenor, Italtel, iStock, Uber, Zappos, Booking.com, Twitter, Facebook, others. | Expedia, Bosch, Otto, eBay, Gap, Forbes, Foursquare, Adobe, Intuit, Metlife, BuzzFeed, Crittercism, CitiGroup, the City of Chicago, others. |

*Figure 7MySQL/MongoDB Comparison [39]*

Both Management systems would be suitable for the project, although the fact that MySQL has been released for nearly 22 years compared to MongoDB's relatively short life of 9 years, the decision was made to use MySQL due to its large amount of documentation and community testing, meaning development problems encountered could be solved quickly by using community sites such as Stackoverflow.com

### Webserver

As development was to take place firstly solely on a laptop, a suitable way to host and interact with the database had to be chosen keeping in mind that the database would eventually be hosted online so that the framework could be used on a mobile from any location with internet access and the application installed. Initial testing was done by following online tutorial to

install MySQL server but after some short use were found unsuitable as they development seemed to be taking quite a long time which a sharp learning curve for installation as many components had to be installed separately [14] i.e. MySQL, phpMyAdmin, and Apache.

An easy install package called **XAMPP** which simplified the installation was then found. XAMPP is a completely free, easy to install Apache distribution containing MySQL, PHP, and Perl and Apache. The XAMPP open source package has been set up to be incredibly easy to install and to use [50]. XAMPP makes it extremely easy for developers to create a local web server for testing and deployment purposes. Everything needed to set up a web server – server application (Apache), database (MySQL), and scripting language (PHP) – is included in an extractable file [49]. This made development and testing for the project a much simpler process

**PhpMyAdmin** which is was included in the XAMPP stack, was found to be very useful in order to speed up the development process. It allows the MySQL database to be managed using an intuitive GUI as well as allowing SQL commands to be inputted. With no experience with database management in general, this easy learning curve was welcome.

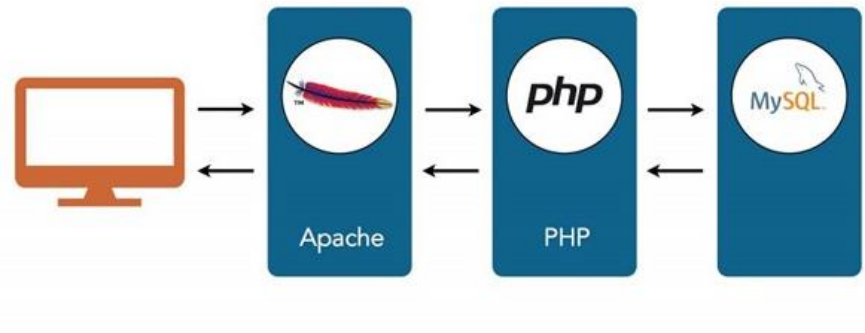



*Figure 8: Server Side Setup [37]*

**PHP** (recursive acronym: Hypertext Pre-processor) is a widely used open source general purpose scripting language that is especially suited for web development and can be embedded into HTML [25]. PHP will be used in order to create the REST API that will be used by the server and client side in order to communicate with each other. It will be used for example, to create charge points and charge point bookings

A **REST API** (representational state transfer) web service is a way of providing interoperability between computer systems on the internet. Web services that are REST-compliant allow requesting systems to access and manipulate textual representations of web resources using a uniform and predefined set of stateless operations [9]. In the case of our applications

framework, a REST API will be used so that the database on the mobile application can communicate and perform READ/WRITE operations. This API will be written in php.

### User Authentication

In order for user to be able to create and book charging points, they must first be able to create a unique account for authentication and so that they're unique details can be accessed. Users should be able to login, after registering their account using a unique username and password. This was found to be relatively simple if the user's details, including their password, was stored on the database, after some research. Storing both username and password details in the database was found to be insecure if it was not encrypted in some form. Therefore, it was seen as best to use a third party authentication service. Many options were considered such as Okta, OAuth and SAML, however after some basic testing of each, the documentation and implementations seemed as though it would take some time to understand completely. As each service seemed quite in depth. Therefore Firebases authentication service was used instead. From previous experience on a project, Firebase authentication was relatively straightforward to implement, although it lacks many of the features given by OAuth, although OAuth can be used in conjunction with Firebase for further security. However, it was thought that for the time being the application should have only basic user authentication due to time constraints. If time allowed, further security could be added by using OAuth.

**Firebase Authentication** provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more.

Firebase Authentication integrates tightly with other Firebase services, and it leverages industry standards like OAuth 2.0 and OpenID Connect, so it can be easily integrated with your custom backend [51]. Firebase also has a dedicated section in Ionics documentation as plugin and appears on multiple blogs and the ionic forum making for a lot of resources and examples of other implementations for an improved learning curve [53], [54].

**OAuth 2.0** is the industry-standard protocol for authorization. OAuth 2.0 supersedes the work done on the original OAuth protocol created in 2006. OAuth 2.0 focuses on client developer simplicity while providing specific authorization flows for web applications, desktop applications, mobile phones, and living room devices. This specification and its extensions are being developed within the IETF OAuth Working Group [52].

### Third Party Charge Points

As the basis of the framework is for users to be able to rent their own charge points to other users as well as find charge points for their own electrical vehicles, making all charge points

which are not bookable and owned by a third party source such as the ESB or car dealerships was another aspect of the project which would prove vital to include. From past use of google maps, it could be seen that many of these charge points are already catalogued on google maps, therefore, a database of these chargers must exist. This was a fair assumption due to the fact that there exists a google maps "Places" API which is readily accessible for locations such as bars and restaurants, and some similar apps also have these charge points mapped out [55], [56]. The places API however did not make charge point data accessible by third parties.

After much searching, a publically usable API provided by OpenChargeMap.org, which made collected public charge points readily available to the public all public charge points was found [57]. This API is well documented, providing example API calls. It contains many parameters for a lot of customisations for functions such as filtering results to a certain country, charger type and distance from a certain point etc. Call-backs can be formatted in either XML, JSON or KML format making for a lot of data portability. This API proved quite fluid in conjunction with the apps user generated charge points

### Cloud Hosting Services

Once the framework had been developed and tested on a local machine, the final step would be to host the database in the cloud using a third party hosting service. This would allow the mobile application to be used anywhere with an internet connection. This is a necessity if the framework is to be demonstrated to the designated panel on campus. For ease of use it was preferable for the cloud server used to have XAMPP installed so that it the local database could be quickly imported and set up

Many options for cloud hosting exist which are all capable of hosting the server side of the framework. Of the many options available, Digital Ocean was first tested. This was mainly due to the fact that GitHub provides a free student pack with vouchers for many applications, one of which is Digital Ocean. After setting up a Digital Ocean, "droplet" (Digital Oceans name for cloud servers) [58], the service was compared and contrasted with a similarly set up cloud server using Amazon Web Service.

Both hosts were suitable for the chosen application, although the Amazon web service was chosen as the UI and setup seemed slightly easier to implement and provides free usage of 750 hours a month for a tier 2 micro device which provides 1 virtual CPU and 2GiB of memory [59].

XAMPP was installed onto the cloud server by using Bitnami. Bitnami is used for application packaging which simplifies the process of installing packages onto a cloud server [76]. Bitnami was also used for the management of the cloud server.

### Version Control

After background analysis had begun and some basic testing of the ionic framework was started, it was found that a version control system would be needed. This was due to the fact that it was apparent the design and implementation of the framework would change vastly throughout the project. Keeping files saved locally could get messy and unorganised quickly, changes would be difficult to revert and control, and the entire project could be lost if a system failure on the machine occurred.

The version control system chosen was git. Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows [79].

Git was used in conjunction with BitBucket which is a git repository management service [80]. BitBucket was chosen over other repository management services such as GitHub due to the fact that it allows for the creation of free private repositories for up to 5 users whereas if Github was chosen, private repositories start at $7 a month. Although public repositories are free, this was not desired due to privacy [81].

## System Analysis and Design

Although the development of the project was done by only one person, the evolution of the project followed a typical system development style similar to the below cycle:

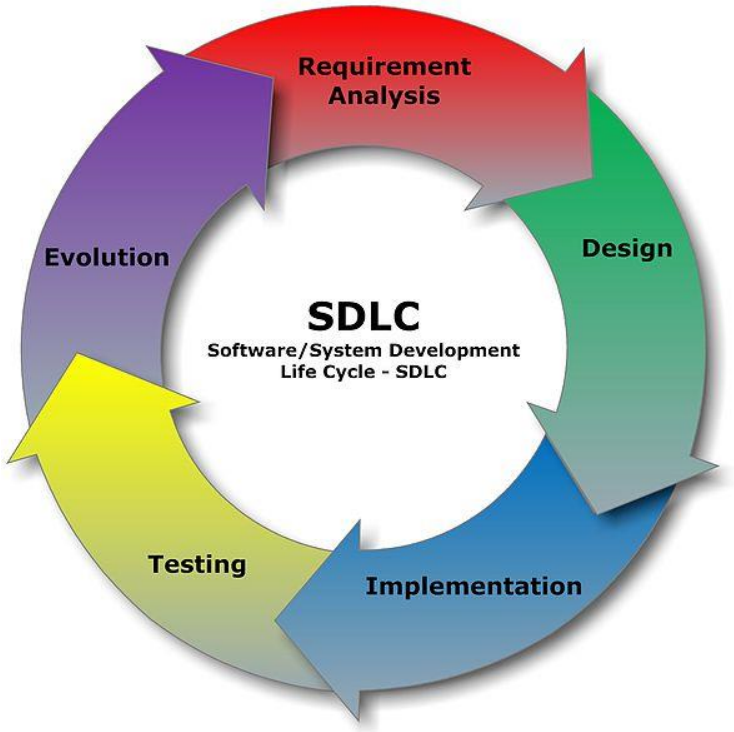



*Figure 9: Software Development Cycle [66]*

This was incredibly important as with a project like this there is quite a lot room for 'scope creep' i.e. there were many features which could have been added which are not essential but would take quite a lot of time to implement. This would in turn allow less time for key features to be implemented correctly, due to limited time. If the key components of the framework did not function correctly, these extra features would be futile.

Each component of the framework followed the same iterative cycle as above before moving onto the next component for example, the user login and authentication page was designed before being implemented and tested. The next component would then be designed, implemented and tested before being integrated with the previous component if need

## Overall System Design

The below diagram shows the overall design of the framework from the database being hosted on the local host, to the database being hosted on a cloud server. This developed over time as a greater understanding of how the different components of the system interacted with each other became apparent.

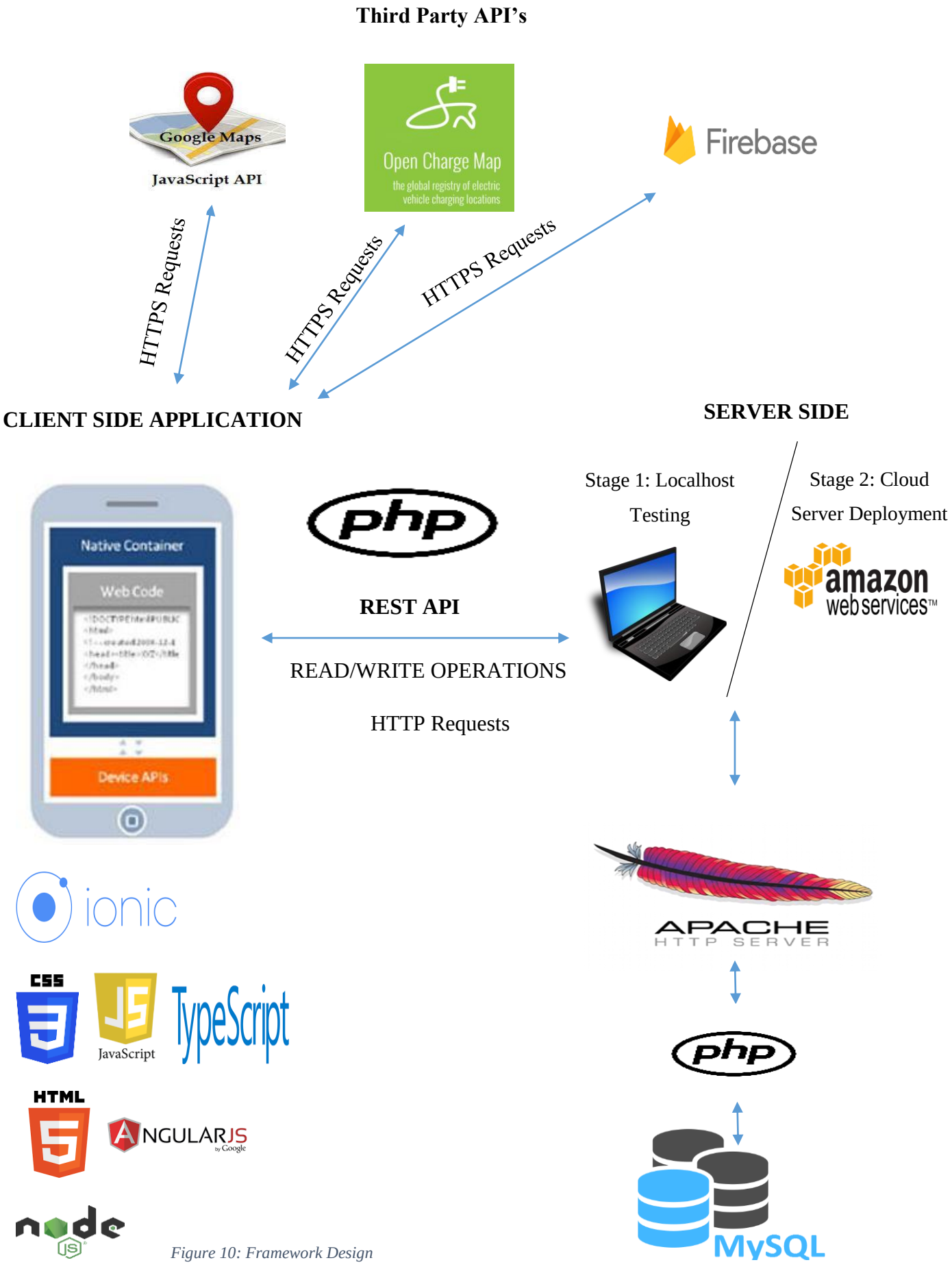



*Figure 10: Framework Design*

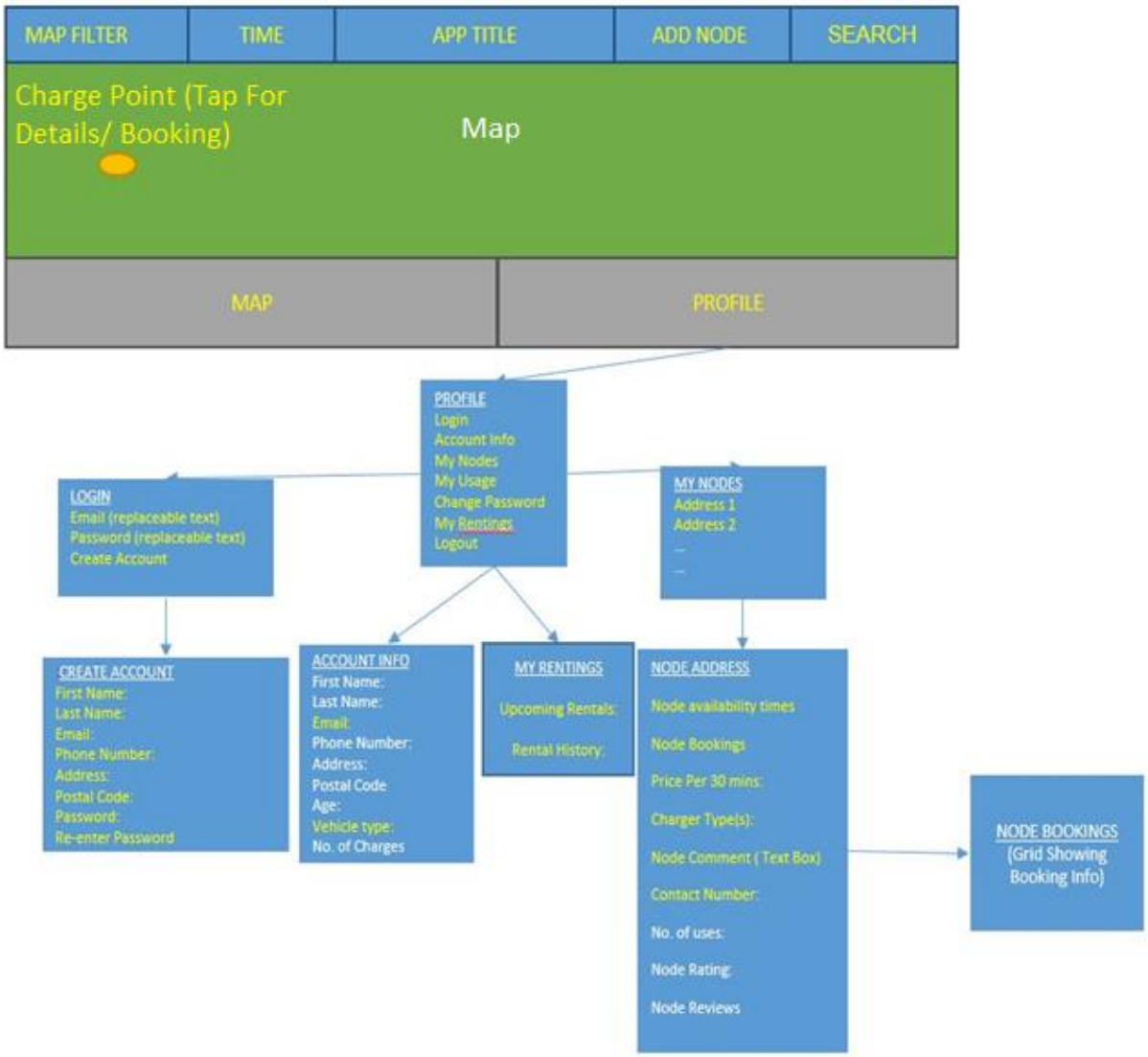
## Client Side

Multiple different designs for the mobile application were made before deciding on a user interface. The goal was to make the UI as simple and easy to understand as possible for new users to integrate quickly.

It was important to have a rough idea of the UI design first before coding began due to the scale of the system. Below is the first design of the client side UI before implementation began with all of the initially desirable features.

Yellow Text – Clickable Options

White Text – Headings/Labels


MAP FILTER
TIME
APP TITLE
ADD NODE
SEARCH
Charge Point (Tap For Details/ Booking)
Map
MAP
PROFILE
PROFILE
Login
Account Info
My Nodes
My Usage
Change Password
My Rentings
Logout
LOGIN
Email (replaceable text)
Password (replaceable text)
Create Account
MY NODES
Address 1
Address 2
...
...
CREATE ACCOUNT
First Name:
Last Name:
Email:
Phone Number:
Address:
Postal Code:
Password:
Re-enter Password
ACCOUNT INFO
First Name:
Last Name:
Email:
Phone Number:
Address:
Postal Code:
Age:
Vehicle type:
No. of Charges
MY RENTINGS
Upcoming Rentals:
Rental History:
NODE ADDRESS
Node availability times
Node Bookings
Price Per 30 mins:
Charger Type(s):
Node Comment ( Text Box)
Contact Number:
No. of uses:
Node Rating:
Node Reviews
NODE BOOKINGS
(Grid Showing Booking Info)

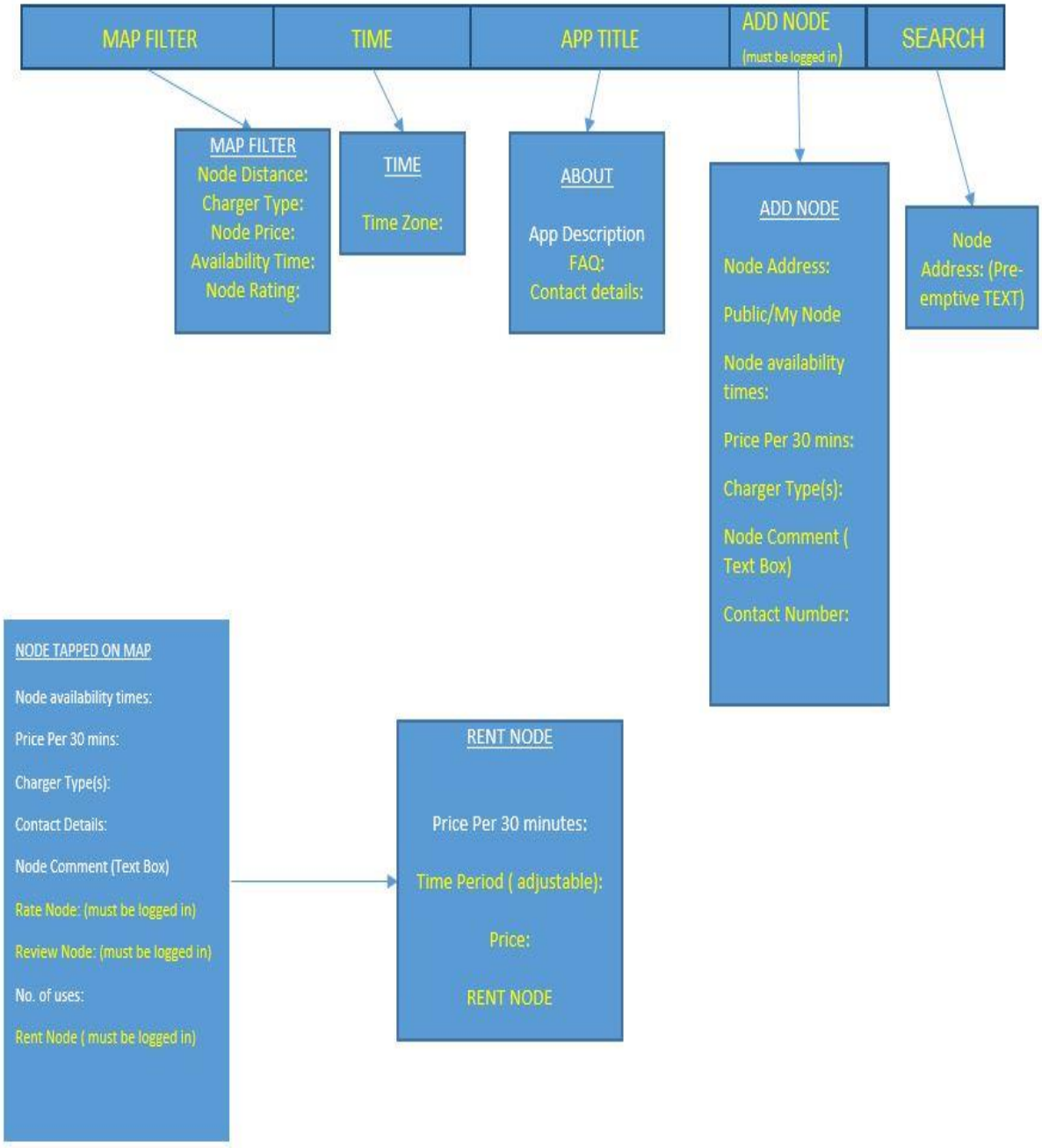



*Figure 11: Mobile Application UI Design*

After some development cycles, the above design gradually changed into the below. As described previously, there was quite a lot of room for scope creep, therefore many of the features which were not critical for functionality were removed and made a low priority. These features would be implemented of enough time allowed.

Below is the design that was finally decided upon:

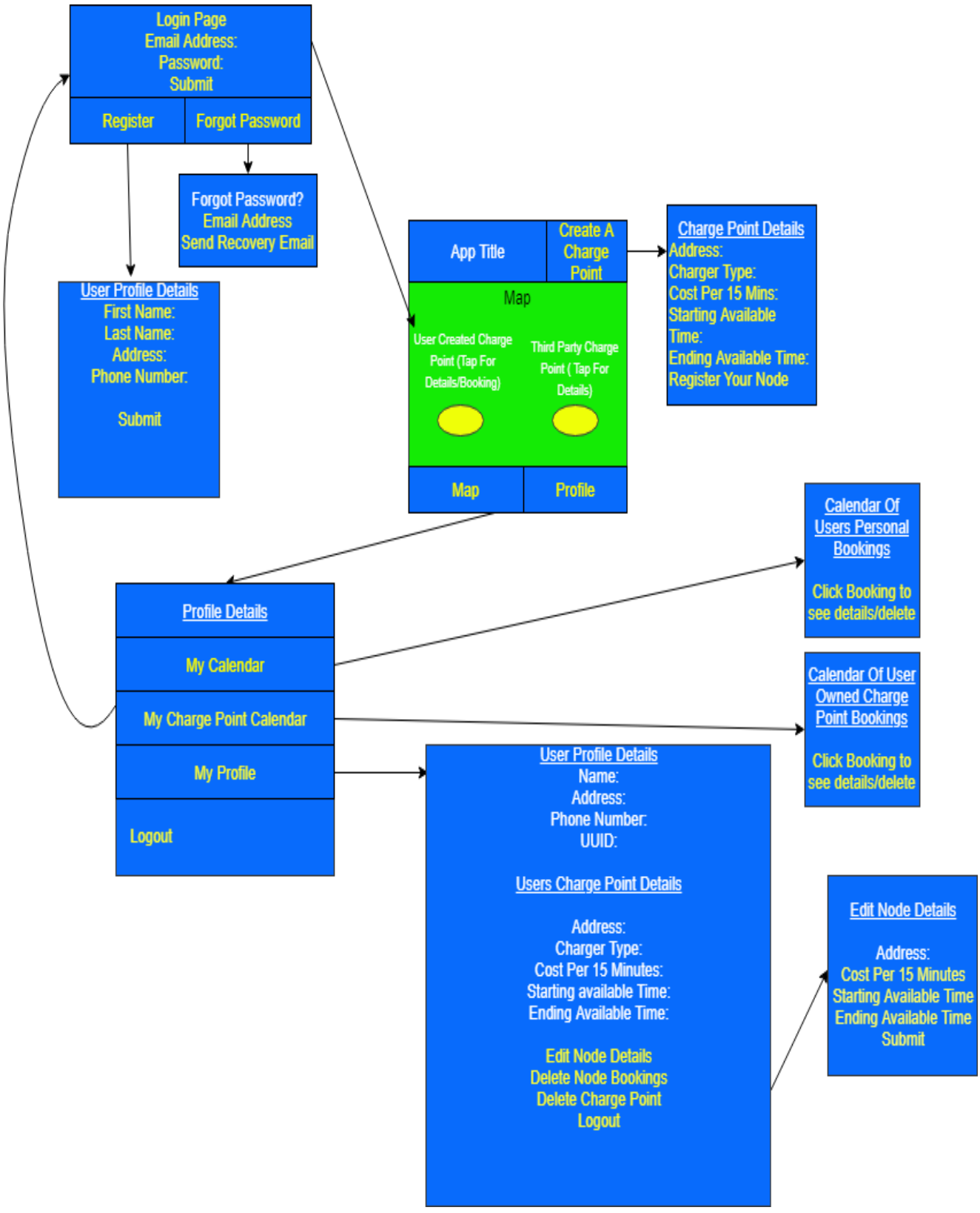



*Figure 12: Final App UI Design*

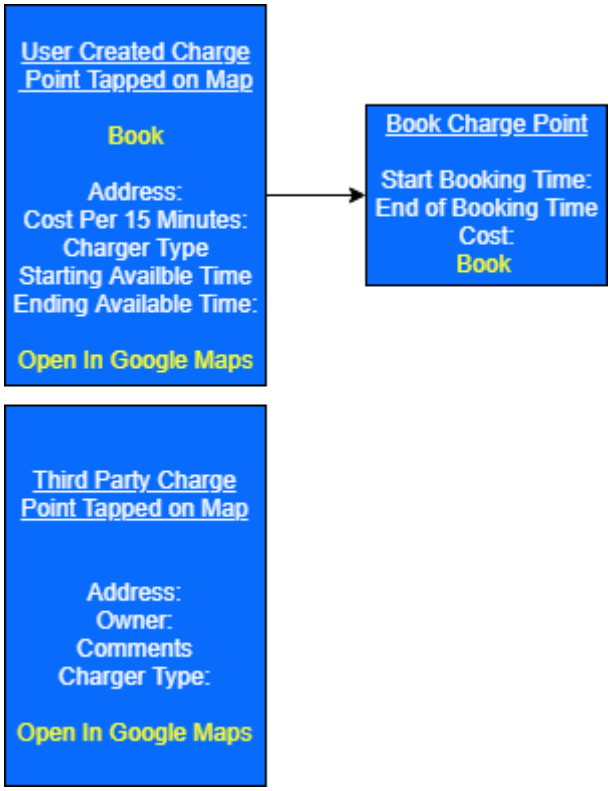



*Figure 13: Final App UI*

Each page of the application was thought through

## Server Side

Although Development and design of the server and client was completed together, implementation of the server side did not take place until later in the project. This was due to the fact that the client side of the framework was changing rapidly with different implementation being tested, therefore, it did not make sense to develop the server side until it got to a stage in the project where it was needed.

Due to the above fact, and the fact that a greater technical knowledge grew as the project progressed. The server side design changed vastly from beginning to end. The following diagram is the first design of the server side database. At this starting stage, only a beginner understanding of database design had been attained. Below the first database design for the framework can be seen

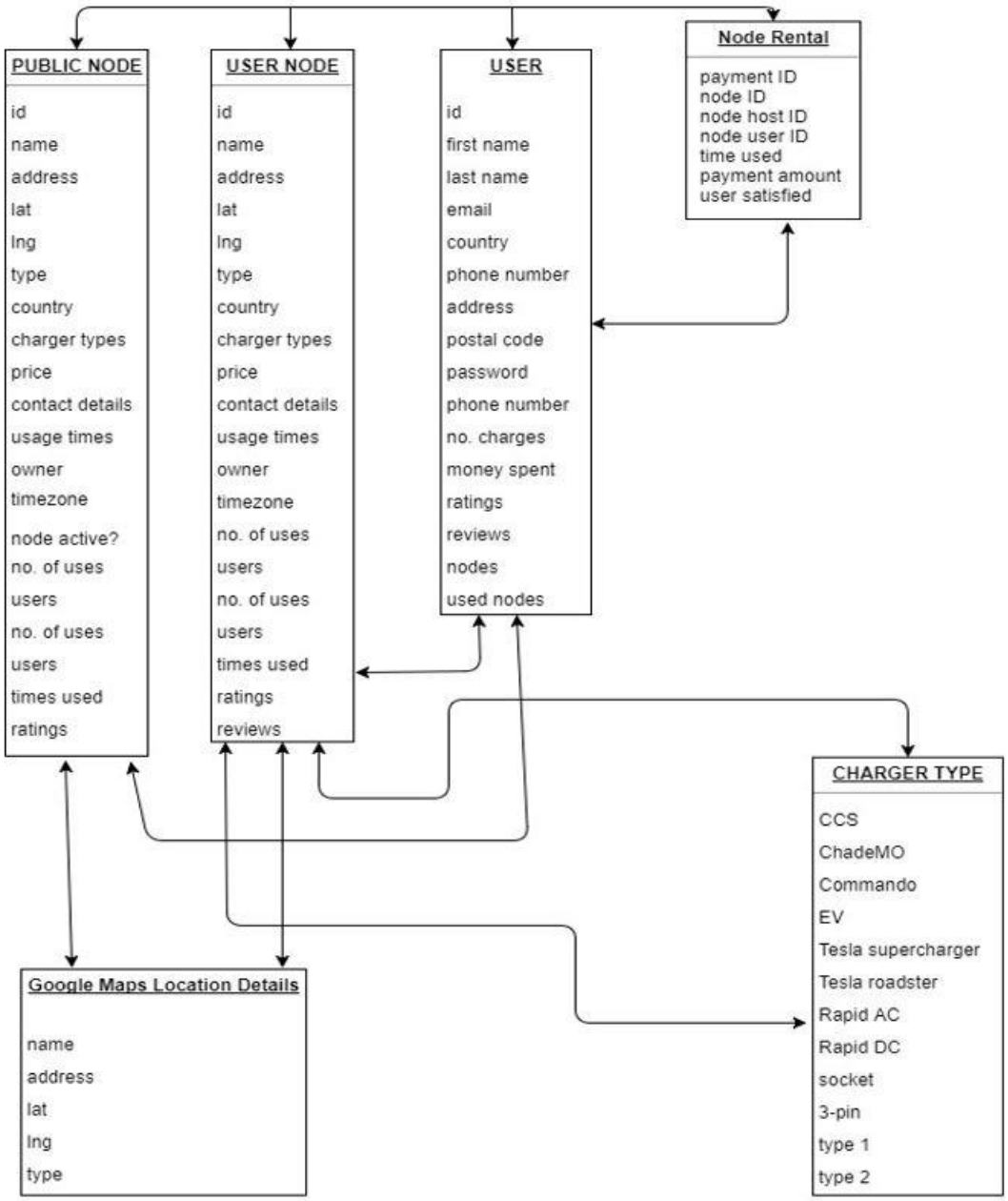



*Figure 14: First Server Side Database Design*

This design, which changed at various stages throughout the project was relatively simple and but displays a beginner understanding of how MySQL databases works is straightforward, but was gradually changed as knowledge of MySQL databases grew.

The following graph shows the final database design:

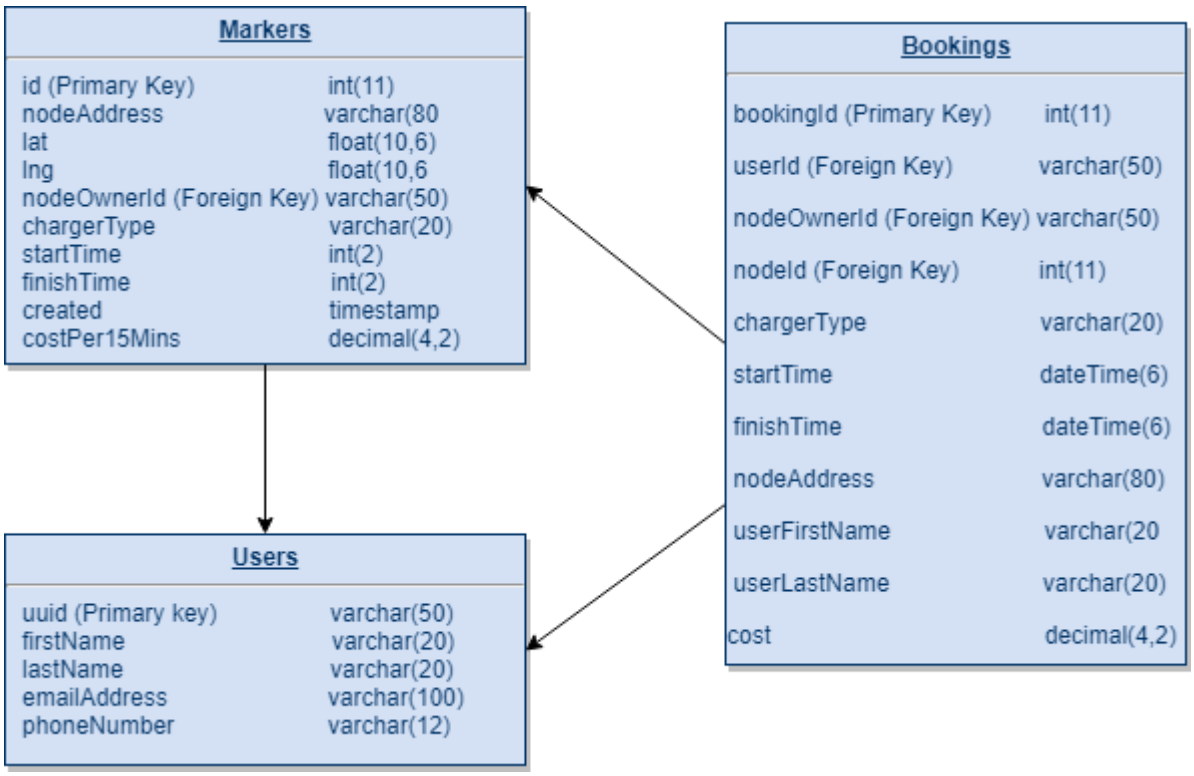



*Figure 15: Final Database Design*

As can be seen, this design is much simpler and efficient. This design could without doubt be made even more efficient and more flexibility could be added, but for the scope of the project it was found to be fit for purpose.

## Local Testing

Testing, as described previously went through 2 stages. The first of which was testing on a local environment. This involved designing and implementing the framework on a single device (Dell 7000 Laptop). On the device, Ionic was installed by following the documentations installation instructions [74], and XAMPP was easily installed using the XAMPP installer for windows [75]. The ionic application was easily linked with the local XAMPP server through HTTP GET/POST request to the localhost i.e. 127.0.0.1

### Cloud Server Testing

Once the framework had been implemented and tested on the local device, the cloud server was set up, the framework database was imported, and connected to the client side, replacing the local host database. The cloud server was set up by following a very detailed documentation created by Apache Friends [77]. The documentation is quite intricate and involves the creation of an SSH tunnel using a program called Putty, an SSH client software [78].

## Implementation

Throughout the project many different implementations were designed tested and integrated together. This process was made much simpler through the use of Git and Bitbucket. Below is a list of the various git adjustments and implementations of the project as it progressed from start to finish.

| Date | Message | Commit ID |
|---|---|---|
| 22/12/2017 | 3 tab layout only no layers _1_ | 80a9d50 |
| 23/12/2017 | first tab level added template | 0f0e68b |
| 30/12/2017 | added loginpage with functionality to check if user is logged in | 15d4fad |
| 30/12/2017 | multiple levels of navstack tested | 7894bd5 |
| 02/01/2018 | added geolocation and adding map markers (need if statement for geolocation fault) | e890b41 |
| 02/01/2018 | barebones google maps implementation added | 7280ddb |
| 05/01/2018 | login/account creationm screen added | 3a0231f |
| 10/01/2018 | solved map marker overload problem | 1a86d46 |
| 10/01/2018 | thirdPartyNodesAdded - loading dynamically - need to add delete dynamicaly | 5070f06 |
| 14/01/2018 | user login/logout authentication added | 52a5f14 |
| 16/01/2018 | user create account, login/logout, first and second name details for display name, reset password, email verification added | f45eb69 |
| 19/01/2018 | added user addable nodes by address, can drag node to change address | 9ed9864 |
| 20/01/2018 | added fixes for delete/add node buttons. added autofill address field. fixed numerous bugs<br>(double page load, looping calls, disable button) | 2c09de3 |
| 22/01/2018 | loginpage fixed | bad3fbc |
| 22/01/2018 | charging point details added. login broken | b65a148 |
| 26/01/2018 | Calendar Added | 2374782 |
| 02/02/2018 | added basic node creation from database with infowindow | c625cfd |
| 05/02/2018 | node creation linked to database, nodes loaded from database | 6628fed |
| 07/02/2018 | fixed calendar bugs - overlapping bookings and days | 7026b96 |
| 07/02/2018 | fixed user authentication with authState, added node start and finish times | 70daf75 |
| 08/02/2018 | more error checking in calendar added. step + click bug persists. removed nodes uncommented volume increase.<br>app noode titles refomatted. prototype almost ready | c90599d |

| | | |
|---|---|---|
| 12/02/2018 | solved node creation 12 AM PM probem, added get method for node start times for booking..<br>broken nodebookingpage (fix by removing http.get, error as no data recieved (null) ) | 810a724 |
| 13/02/2018 | node start/finish times loaded from db. solved add/remove event on load button problem (make 2 separate calendar pages,<br>same instance on 2 tabs were crashing | a988bea |
| 14/02/2018 | node bookings pushed to server. Still need to pull down and perform overlap checks | 47d2123 |
| 15/02/2018 | node bookings from databse now load onto calendar in app. overlap checking and title needed | b7b8b5e |
| 16/02/2018 | Initial commit | 9c0d3e2 |
| 18/02/2018 | login screen added, calendar fully functional, need to add in user delet marker then complete | 9227d6d |
| 18/02/2018 | multiple change major bugs fixed elements removed | a2aa888 |
| 23/02/2018 | multiple changes readded | dcea8c3 |
| 24/02/2018 | create Node cleaned up, one node per user | 359e1e2 |
| 24/02/2018 | profile page added, needs fixing | ba33300 |
| 24/02/2018 | myCalendarPage added | 7a23b07 |
| 25/02/2018 | prototype complete | 06e7a7c |
| 25/02/2018 | delete bookings, checks added | 8f3594c |
| 27/02/2018 | booking edited | b43df5d |
| 27/02/2018 | calendar updated, booking updated | d89e91e |
| 27/02/2018 | changes to calendar booking, booking creation/deletion shown straight away, color layout changed | e9f8a46 |
| 03/03/2018 | charge point costs are now correct | 18a0ccc |
| 08/03/2018 | minor bug fixes in calendar and costing | a7d228a |

As can be seen above, a lot of effort was put into the project and commits were pushed regularly to the repository in order to prevent unnecessarily long rollbacks

Although design and basic testing of the ionic framework and various elements of the project began in September 2018. The first versions of the project were pushed to the repository until. 22nd December 2018. This was due to the fact that git was not used until multiple failures had occurred storing the files locally on the device at the beginning of the project. Effort also had to be diverted to study for the upcoming end of term examinations.

On 16/2/2018, as noticeable in the list of git submissions for the project above, the message is titled “initial commit”. This is due to the face that problems were encountered, the easiest solution to this was to create a new repository. This will be explained in detail in the “Problems Encountered” section below.

## Map Page

Per the design of the application and the from analytical background, it was clear that the function of the entire framework, from a user's perspective, would hinge on the map page. There are over 800 lines of code in the Map Page alone, therefore, only key functions will be discussed in detail

### Map Creation

```
initMap(mapElement) {
    this.infoWindowObservable.true = 0;
    this.geolocation.getCurrentPosition().then((position) => {

      //let defaultLatLng = new google.maps.LatLng(40.4040, 60.4040);

      this.myLat = position.coords.latitude;
      this.myLng = position.coords.longitude

      let latLng = new google.maps.LatLng(position.coords.latitude,
position.coords.longitude);

      let mapOptions = {
        center: latLng,
        zoom: 14,
        maxZoom: 30,
        streetViewControl: false,
        fullscreenControl: false,
        rotateControl: false,
        mapTypeControl: false,
        mapTypeId: google.maps.MapTypeId.ROADMAP,
        clickableIcons: false,
      };

      this.map = new google.maps.Map(mapElement, mapOptions);
```

The above code snippet is only half of the initMap() function which is responsible for creating the google maps instance. The map opens initially on the current location of the user through the getCurrentPosition() function and using the coordinates returned in instantiating the map.

The initMap function also include multiple listeners such as the one below which is used for loading the charge points on the map when the user is not scrolling through the app. This is to counteract charge points being loaded unnecessarily as the user scroll through the map.

The removeMarkers() function is also called  which removes charging points once a certain threshold of markers have been loaded.

```
        google.maps.event.addListener(this.map, 'idle', () => {
          this.loadThirdPartyMarkers();
          this.removeMarkers();
          this.loadAppMarkers();
```

### Addition of Charge Points to the Map

```
loadAppMarkers() {


    this.subscriber =
this.http.get('http://colinfyp.bitnamiapp.com/data_marker/appMarkerData.php')
      .map(res => res.json())
      .subscribe(appMarkers => {

        this.appMarkers = appMarkers;

        console.log("hello" + appMarkers)

        if (appMarkers == null) {
          console.log("problem");
        }
        this.addAppMarkers(appMarkers);

      });


  }


  addAppMarkers(markers) {

    let costPer15Mins;
    let markerLatLng;
    let lat;
    let lng;
    let address;
    let Title;
    let AddressLine1;
    let AddressLine2;
    let Town;
    let StateOrProvince;
    let Country;
```

```
    let Membership;
    let NumberOfPoints;
    let GeneralComments;
    let Connections;
    var bookButton;
    let chargerType;
    let start;
    let finish;
    var nodeOwnerId;
    var id;

    for (let marker of markers) {

      try {
        costPer15Mins = marker.costPer15Mins;
        nodeOwnerId = marker.nodeOwnerId;
        id = marker.id;
        lat = marker.lat;
        lng = marker.lng;
        Title = marker.name;
        AddressLine1 = marker.nodeAddress
        chargerType = marker.chargerType
        start = marker.startTime;
        var startFormat = start + ":00";
        finish = marker.finishTime;
        var finishFormat = finish + ":00";
        console.log("OWNER: " + nodeOwnerId);
        if (start == 0) {
          startFormat == "00:00";
        }



      }
      catch (error) {

      }

      address = "<br />" + "Cost per 15 mins: €" + costPer15Mins + "<br />" +
"Address: " + AddressLine1 + "<br />" + " Available from: " + startFormat + "-
" + finishFormat;


      markerLatLng = new google.maps.LatLng(lat, lng);

      if (!this.markerExists(lat, lng)) {
        var image = "http://maps.google.com/mapfiles/ms/icons/green-dot.png";

        marker = new google.maps.Marker({
```

```
          map: this.map,
          animation: google.maps.Animation.DROP,
          position: markerLatLng,
          clickable: true,
          icon: image,


        });

        var self = this;
        bookButton = '<button id="book"
style="height:45px;width:45px">Book</button>';
        var open = '<button id="open">Open Directions In Google
Maps</button>';

        google.maps.event.addListener(marker, 'click', (function (marker,
content, appInfoWindow, nodeAddress, id, chargerType, nodeOwnerId, start,
finish, costPer15Mins, lat, lng) {
          return function () {
            appInfoWindow.setContent(bookButton + " \n \n" + content + "<br
/>" + open);
            appInfoWindow.open(map, marker);
            self.updateBookingNodeAddress(nodeAddress, id, chargerType,
nodeOwnerId, start, finish, costPer15Mins, lat, lng);

            self.thirdPartyMarkerInfoWindow.close();



          };
})(marker, address, this.appInfoWindow, AddressLine1, id, chargerType,
nodeOwnerId, start, finish, costPer15Mins, lat, lng));

        let markerData = {

          lat: lat,
          lng: lng,
          marker: marker
        };
        this.existingThirdPartyMarkers.push(markerData);
      }
    }
  }
```

The functions described above are responsible for generating app user created charge points which are added to the map. The loadAppMarker() function makes a http.get request to the cloud server which in returns the details on the chargepoints i.e. coordinates, address, charger

type etc. this information is then used in the addAppMarkers() function which is responsible for creating the markers for each charge point on the map. Each marker is then given a unique listener which recognises the marker has been clicked. When a marker is clicked, an info window loads which displays the charge point's details, as well as 2 buttons. One for to open the charge points booking page, and the other for opening direction to the charge point in google maps These markers are added to an array of currently loaded markers which is used so that duplicate markers are not reloaded which will effect performance.

### Problems encountered

## Results

## Discussion of Results

## Conclusion

## References

[1] Welcome to Ionic – Ionic Framework (2017). Available: http://ionicframework.com/docs/v1/guide/preface.html

[2] Sanket, Devlekar Hybrid Application Development (April 3, 2015). Available: https://www.google.ie/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=0ahUKEwj4v82u7vnWAhXJKlAKHZLvAMIQjRwIBw&url=https%3A%2F%2Fwww.slideshare.net%2Fsamde94849%2Fhybrid-app-in-ionic-framework-overview&psig=AOvVaw10JsOeUIyQEAz0w5TiY4VC&ust=1508405438343924

[3] Sweeney, Austin. The Must Read HTML vs CSS Infographic (March 11, 2015). Available: http://www.codingdojo.com/blog/html-vs-css-inforgraphic/

[4] SiteGround Hosting. MySQL Tutorial - Learn more about MySQL and how to use it (2017). Available: https://www.siteground.com/tutorials/php-mysql/mysql.htm

[5] Spring.io. Understanding: REST (2017). Available: https://spring.io/understanding/REST

[6] zap-map.com. Charging Speeds and Connectors (2017). Available: https://www.zap-map.com/charge-points/connectors-speeds/

[9] "Web Services Architecture". World Wide Web Consortium. 11 February 2004. 3.1.3 Relationship to the World Wide Web and REST Architectures. Available: https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#relwwwrest

[11] Media Mark (3.2.12). "Sass - Syntactically Awesome Stylesheets". Available: Sass-lang.com.

[12] "CSS developer guide". Mozilla Developer Network. Available: https://developer.mozilla.org/en-US/docs/Learn/CSS

[14] Google Inc. Google Maps JavaScript API Logo (February 8th 2005). Available: https://www.beginnersheap.com/wp-content/uploads/2016/09/Google-Maps-JavaScript-API-Logo.png

[15] Netscape Communications Corporation. JavaScript Logo (4th December 1995). Available: https://www.eduonix.com/blog/wp-content/uploads/2015/04/JavaScript.png

[16] Catlin Hampton. Sass Logo (2006). Available: http://sass-lang.com/styleguide/brand

[17] Microsoft Corporation. Typescript Logo (1st October 2012). Available: https://images.g2crowd.com/uploads/product/image/social_landscape/social_landscape_1489714904/typescript.png

[18] Google Inc. AngularJS Logo (20th October 2010). Available: https://upload.wikimedia.org/wikipedia/commons/thumb/c/ca/AngularJS_logo.svg/1000px-AngularJS_logo.svg.png

[19] Node.js Developers, Joyent. Node.js Logo (27th May2009). Available: https://nodejs.org/static/images/logos/nodejs-new-pantone-black.png

[20] W3C, WHATWG. HTML5 Logo (28th October 2014). Available: https://www.w3.org/html/logo/downloads/HTML5_Logo_512.png

[21] Oracle Corporation. MySQL Logo (23rd May 1995). Available: https://www.handybackup.net/images/icons/mysql-backup.png

[22] ClipartPanda.com. Laptop ClipArt (2017). Available: http://images.clipartpanda.com/laptop-clipart-4393-laptop-design.png

[24] The PHP Development Team, Zend Technologies. PHP Logo (8th June 1995). Available from: https://maxcdn.icons8.com/Share/icon/ios7/Logos//php_logo1600.png

[25] The PHP Group What is PHP? (2017). Available from: http://php.net/manual/en/intro-whatis.php

[26] Jose Fermoso (April 5, 2009). "PhoneGap Seeks to Bridge the Gap Between Mobile App Platforms". GigaOM. Available: https://gigaom.com/2009/04/05/phonegap-seeks-to-bridge-the-gap-between-mobile-app-platforms/

[27] https://ionicframework.com/getting-started

[28] https://developers.google.com/maps/documentation/javascript/tutorial

[29] *Foley, Mary Jo (1 October 2012). Microsoft takes the wraps off TypeScript, a superset of JavaScript.* ZDNet. CBS Interactive. Available:

http://www.zdnet.com/article/microsoft-takes-the-wraps-off-typescript-a-superset-of-javascript/

[30] Bright, Peter (3 October 2012). Microsoft TypeScript: the JavaScript we need, or a solution looking for a problem? Ars Technica. Condé Nast. Available:

https://arstechnica.com/information-technology/2012/10/microsoft-typescript-the-javascript-we-need-or-a-solution-looking-for-a-problem/

[31] borisyankov/DefinitelyTyped". GitHub. Available:

https://github.com/DefinitelyTyped/DefinitelyTyped

[32] TypeScript 1.0 Tools for Visual Studio 2012. Available: https://marketplace.visualstudio.com/items?itemName=TypeScriptTeam.TypeScript10ToolsforVisualStudio2012

[33] https://forum.ionicframework.com/t/google-maps-native-plugin-vs-javascript-sdk/85651/3

[34] https://code.visualstudio.com/docs/editor/intellisense

[35] https://ionicframework.com/docs/developer-resources/editors_and_ides

[36] https://forum.ionicframework.com/t/which-ide-is-better-for-production-and-development/76596

[37] https://toster.ru/q/319515

[38] https://www.mongodb.com/what-is-mongodb

[39] https://hackernoon.com/mongodb-vs-mysql-comparison-which-database-is-better-e714b699c38b

[40] https://www.youtube.com/watch?v=UgHRay7gN1g

[41] https://www.apachefriends.org/index.html

[43] "What is CSS?". World Wide Web Consortium. Available: https://www.w3.org/standards/webdesign/htmlcss#whatcss

[44] https://thimbleprojects.org/niallobrien/112064/

[45] https://www.codeschool.com/beginners-guide-to-web-development/how-does-a-website-work

[46] HTML 4.0 Specification — W3C Recommendation — Conformance: requirements and recommendations". World Wide Web Consortium. December 18, 1997.

[47] https://medium.com/@El_Nino/javascript-for-dummies-part-1-aebe9422c0a

[48] http://academy.trendonix.com/courses/web-development/introduction-javascript

[49] https://www.thecodedeveloper.com/how-to-install-xampp-on-windows/

[50] https://www.apachefriends.org/index.html

[51] https://firebase.google.com/docs/auth/

[52] https://oauth.net/2/

[53] https://blog.paulhalliday.io/2017/09/08/ionic-3-and-firebase-user-authentication/

[54] https://forum.ionicframework.com/t/ionic-demo-with-firebase-auth-and-database-usage/84859

[55] https://developers.google.com/places/

[56] https://www.esb.ie/our-businesses/ecars/charge-point-map

[57] https://openchargemap.org/site/develop/api

[58] https://www.digitalocean.com/community/tutorials/how-to-create-your-first-digitalocean-droplet

[59] https://aws.amazon.com/ec2/instance-types/

[60] https://market.ionicframework.com/plugins/ionicdatepicker

[61] http://home.bt.com/lifestyle/travel/travel-advice/what-is-airbnb-11363981595930

[62] https://medium.com/airbnb-engineering/how-we-partitioned-airbnb-s-main-database-in-two-weeks-55f7e006ff21

[63] https://www.justpark.com/

[64] https://www.uber.com/en-IE/

[65] http://www.plugincars.com/sites/default/files/plugshare-app-map-search-filters.png

[66] https://commons.wikimedia.org/wiki/File:SDLC_-_Software_Development_Life_Cycle.jpg

[67] https://github.com/twinssbc

[68] https://github.com/twinssbc/Ionic2-Calendar

[69] https://lh6.googleusercontent.com/nlNze871RqFpbY0n8Nxu1C7xxhKagryn_IJ-50SoMKgZ-vjddKXkd_DlsgvUIaOzKt5LhpLV6agKhp7knADrISVZA-ia_aFPjzCCxFCBwOSopHqeO9nJ4EtIDcFPlUDW7VXid1oD

[70] https://cdn-images-1.medium.com/max/1000/1*ipwpqQrHz0Lkd_5setXQCQ.jpeg

[71] http://tecdistro.com/wp-content/uploads/2015/03/apache318x2601.png

[72] http://www.aapnainfotech.com/wp-content/uploads/2015/11/AmazonWebservices_Logo1-1024x410.png

[73] https://lh4.googleusercontent.com/-t8mgyfl9mFw/AAAAAAAAAAI/AAAAAAAAAAc/t_bMvUyjSbM/photo.jpg

[74] https://ionicframework.com/docs/intro/installation/

[75] https://www.apachefriends.org/download.html

[76] https://bitnami.com/about-us

[77] https://www.apachefriends.org/docs/hosting-xampp-on-aws.html

[78] https://www.putty.org/

[79] https://git-scm.com/

[80] https://bitbucket.org/

[81] https://www.upguard.com/articles/github-vs-bitbucket

---

## Acknowledgements

## Appendices

## Poster