FINAL YEAR PROJECT

Singapore Vacation Planner App for Tourist

FINAL REPORT

Submitted by: Tee Shi Jun

Matriculation Number: U1320438G

FYP Supervisor (NTU): Prof Shum Ping

FYP Examiner (NTU): Prof Linbo

School of Electrical & Electronic Engineering

Contents

Abstract 4
Acknowledgement 5
List of Figures 6
Chapter 1 Introduction 7
1.1 Background 7
1.2 Project Objective 7
Chapter 2 Literature Review 8
2.1 Current Situation 8
2.2 Mobile Application Development Framework 9
2.2.1 Research 9
2.2.2 Adaptability 10
2.3 Ionic 2/3 Framework 11
2.3.1 HTML (Ionic 2) 12
2.3.2 TypeScript .ts 12
2.3.3 Sassy CSS .scss 13
2.3.4 JavaScript Object Notation .json 13
2.3.5 Ionic Component 14
2.3.6 API 14
2.3.7 Ionic Native 14
2.4 Node.js 15
2.5 Cordova 16
2.6 Android SDK 16
2.7 Xcode (iOS SDK) 17
Chapter 3 Existing Application 18
3.1 Tourist Applications 18
3.1.1 Singapore City Guide Trip.com 19
3.1.2 Singapore Guide by Triposo 20
3.1.3 Singapore Travel Guide-Tourias 20
Chapter 4 Materials and Procedures 21
4.1 Methodology 21
4.1.1 Preparation of material 22

4.1.2 Implementing new page 25
4.1.3 Inserting google map 25
4.1.4 Creating list of places of interest and placing of markers 27
4.1.5 Calculation formula 28
4.1.6 Design and user navigation 29
4.1.7 Troubleshooting 30
4.1.8 Exporting to native devices – iOS & Android 31
4.2 Test 33
4.3 Summary and results 34
Chapter 5 Discussion 35
5.1 Limitation and Recommendation 35
5.2 Conclusion 35
Chapter 6 Future Development 36
References 37
Appendix 39

Abstract

With the increased spending power of the world, people in this 21st century like to travel all around the world. They require something that can be useful to them and at the same time, help them during their travel. With the popular usage of smartphone, more people are in favors of a lightweight application that acts like a swiss-army knife. Hence to meet the demand, especially in Singapore, there is a need for a highly useful application to be developed so as to save time and costs for travelers. Be it be a personal or business trip. A smart tourist will have this application in the phone. Existing applications in the market were research and found to have lack of support for Singapore. Places of interest were also researched, and data are collected to feed the application. In conclusion, this project is a success. It has been tested repeatedly to show constant results. The application is undergoing debugging while new features are added regularly. The ideal outcome of this application is to be an official partner of Singapore Tourism Board.

Acknowledgement

This final year project has given me a chance to work with the ionic 2/3 components which include Html, SCSS and TypeScript from zero knowledge. I am able to apply what I learnt from Web Application Design (EE4717) and applied the fundamental knowledge to an actual mobile application. It was an enriching and fulfilling journey.

I would like to express my deepest gratitude to Prof. Shum Ping for his supervision towards the success of the project.

I would like to express my deepest gratitude to NTU Alumni Victor Tan for his continuous support, guidance and patience throughout the whole period of final year project. His persistence and encouragement has inspired me to become consistent and hardworking.

Being a part time student in EEE, I would also like to express my gratitude towards the other fellow part time students for the sharing of useful information, cooperation, and supporting one another to bring our projects to success despite having a limited timeline.

Lastly, I would like to thank the great ionic community, their expertise had inspired me for more ideas for this project.

List of Figures

Figure 1. Passenger movements at Changi Airport 8
Figure 2. Application developed using Ionic 2 10
Figure 3. Ionic 2 Logo 11
Figure 4. Navigation Stack Push and Pop 12
Figure 5. Comparison between scss and css 13
Figure 6. node.js 15
Figure 7. Command-line user interface (CLI) Node.JS 15
Figure 8.view of the Cordova application architecture 16
Figure 9.Icon of Android Studio inclusive of SDK tools 17
Figure 10. Xcode 17
Figure 11. Playstore Singapore tourism-friendly application 18
Figure 12. Singapore City Guide trip.com screenshot preview 19
Figure 13. Singapore City Guide trip.com screenshot preview 20
Figure 14. Iterative method and incremental build model for EzVacation@SG 21
Figure 15. Software required 22
Figure 16. Directory of EzVacation@SG 23
Figure 17. Common Error Message: Typescript Error 24
Figure 18. Editing on Notepad++ 24
Figure 19. Generating google map API developer key 25
Figure 20. Map.ts calling GoogleMaps and Locations 26
Figure 21. Map and Places to Visit Tabs 27
Figure 22. Haversine formula model 28
Figure 23. Comparison of conceptual design (sketchup) with the actual design 29
Figure 24. app.module.ts 31
Figure 25. Successfully built APK 32
Figure 26. Mobile application testing methods 33

Chapter 1 Introduction

1.1 Background

Singapore vacation planner is a smart application that is made to assist tourists. It served as a smart swiss army knife for tourists to travel in Singapore. Even though Singapore have a strong tourism market, it does not have a powerful yet simple to use “free” application that can be a great support to current or future tourists. Therefore, this smart application is created to ease the planning of travelling in Singapore. This application will continue to improve the tourism market in Singapore. This smart application does not rely heavily on user participation, instead user rely heavily on its functions.

1.2 Project Objective

- To assist tourist or business visitor for an efficient and interesting tour in Singapore
- Application will allow user to select multiple different tourism locations in Singapore and auto-generate the details information for the User
- Application will make use of Google Map API and developer’s computed formula in the background to operate

The smart application will be ported to both Android and IOS.

Chapter 2 Literature Review

2.1 Current Situation

According to Changi Airport traffic statistic, the average passenger movements in year 2017 is approximal 5 million people per month. [1A] With Singapore Tourism collaborating with Changi Airport to market and promote Singapore as an attractive stopover and twinning destination for travelers, there will be more tourists visiting Singapore. [1B] And with the increased tourist, the demand for a useful application come into the picture. Upon surveying tourist in a particular Guesthouse in Chinatown, a light-weight mobile application was chosen as the most popular choice amongst the other. In order to satisfy the user's need, the application is noted to include place of interests, with auto-itinerary generation as a bonus feature. An application that is able to show some tips for the location such as the deal of the entrance tickets, must-buy-items and must-try-foods at the spot were feedbacked by the guests. General public also shown interest in the mobile application. The application comes into good use when the public are expecting foreigner friends or business partners and are unsure of where to go or what to do.

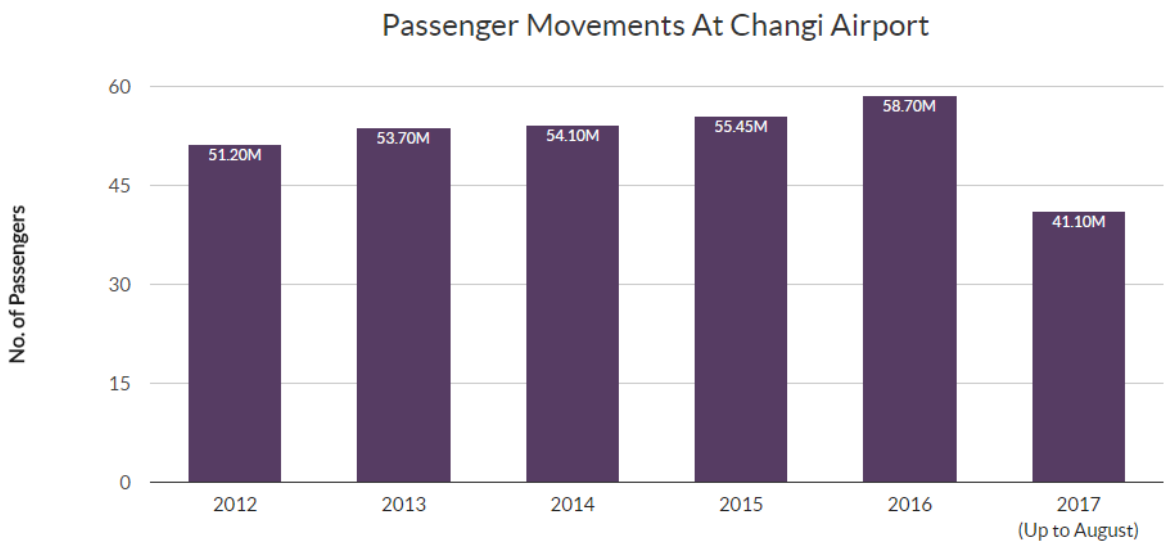

Figure 1. Passenger movements at Changi Airport

2.2 Mobile Application Development Framework

In the 21st century, variant mobile phones are starting to appear. Many different brands and operating systems are in the current market. The two-major mobile operating systems, which are the IOS and the Android, have the most market shares in the world. Due to the nature of the two operating systems, coding for one platform cannot be easily port over to the another. Different operating mobile systems such as the Android, Windows and iOS need to be coded separately. Ionic was developed in order to ease the development hybrid application. Using the same coding to export to different firmware by using general interface method. The Ionic Html and TypeScript (AngularJS), Sassy CSS are component inside Ionic framework. It is one of the top few frameworks for constructing a hybrid app that is suitable for all major platforms. This additional tool is easily adapted to the protocol of different native mobile, hence creates a more efficient and easy management platform for the app developer.

2.2.1 Research

Ionic 2, React Native and NativeScript are some of the popular frameworks use web tools for mobile development and use the native API of iOS and Android. The three platforms provide similar methods of developing mobile applications through website application technologies mainly JavaScript, HTML, CSS and etc. However, the different between them is their features as well as the APIs communication between individual platforms.

Various reasons are taken into consideration upon choosing Ionic 2. Firstly, Ionic 2 employs a wide range of user supports from the Ionic community. Secondly, Ionic 2 has been popular for the past 2 years, with millions of developers using Ionic 2 as their framework to create their application. Some example of those popular applications includes MacDonald, Baskin Robbins and etc. Lastly, compared to the other two, Ionic 2 have the most stable Hybrid web development support which is suitable for the project which is target at Android and iOS. For the ease of the future development of this project, Ionic 2 is selected.

2.2.2 Adaptability

Ionic 2 make use of web technologies to develop mobile application. Learning the basic of Ionic 2 requires plenty of reading and practical hand on. The learning progress of Ionic 2 will be easier for developer with previous website building background knowledge, they can pick up Ionic 2 easily. Ionic 2 includes some special functions and commands which required programmers to study and have a good knowledge on how the infrastructure is, before starting to make an application. The learning curve is deep, however with more practice, the user can get a good grasp on how this amazing this platform can perform. Moreover, a lot of reputable company app is based on Ionic 2 & Angular JS 2. New Ionic 2 developer can easily adapt and continue other's project. Some of the concern point to be noted:

1. Understand the different between website & mobile app using ionic.
2. Different between Truescript and JavaScript; Ionic 2 protocol system
3. Angular 2 and Angular JS
4. Syntax and structural changes to the original framework

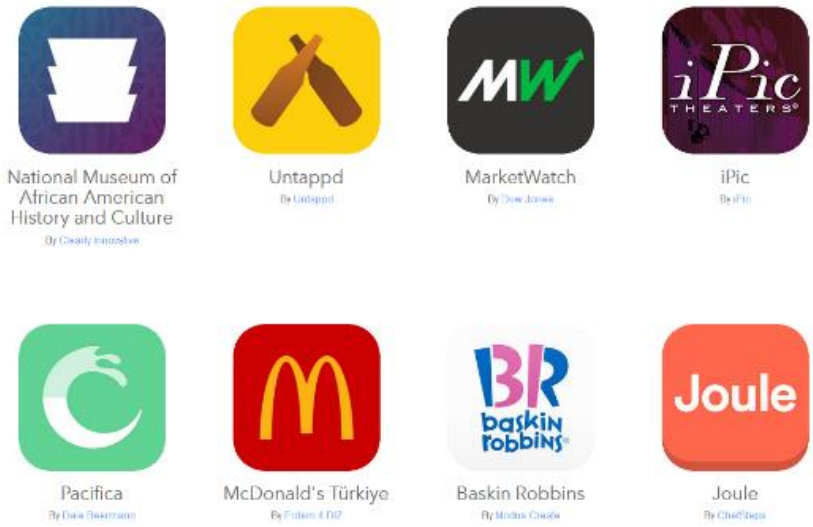

Figure 2. Application developed using Ionic 2

2.3 Ionic 2/3 Framework

Ionic 2 Framework, renamed as Ionic 3 on April 2017, is a hybrid mobile development framework that is built on top of Angular 2 with some changes. [2] Using HTML, CSS, and JavaScript, various simple web application can be ported to a hybrid mobile application with minimum effort. The framework also comes with a set of UI components and functions which is useful for developing wonderful applications. Cordova needs to be employed to build the mobile application while Ionic handles the user interface part. Ionic, together with Angular and Cordova, will be able to build the UI, handles the logic and compile into mobile installation files which will be then deployed into mobile phones. The different between Ionic 2/3 framework and Ionic 1 is that Ionic 1 was built solely for hybrid mobile application while Ionic 2/3 is built to support progressive web applications. The syntax used in Ionic 2/3 is more user friendly and concise. The details of individual technology will be mentioned in the subsections.

Figure 3. Ionic 2 Logo

2.3.1 HTML (Ionic 2)

Hypertext Markup Language (HTML), together with With Cascading Style Sheets (CSS) and JavaScript, helps to create web pages and web applications as the standard language.

The HTML in Ionic 2 includes the 'ion-' prefixed directives. Prefixed directives such as 'ion-list','ion-item' and etc are angular directives that enable extended features. Some of the features include swipe to delete, reorder and listing. HTML in Ionic 2 employ the push and pop stack to navigate between pages.

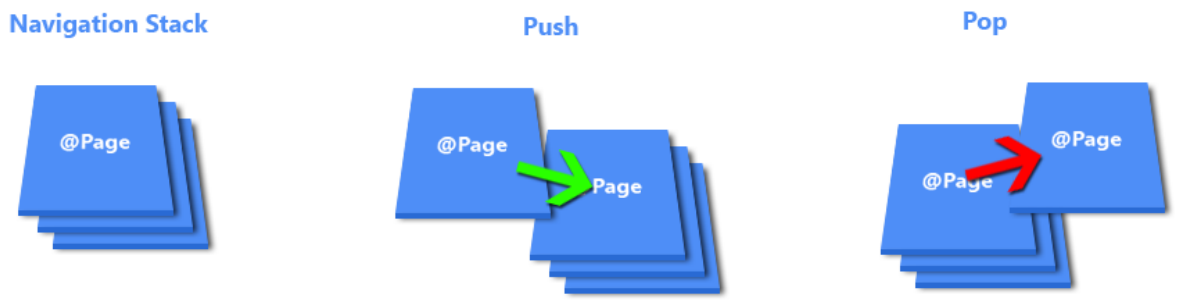

Figure 4. Navigation Stack Push and Pop

2.3.2 TypeScript .ts

TypeScript, well maintained by Microsoft, is one of the main web technologies used in Ionic 2 framework. TypeScript is claimed to be a superset of JavaScript. Most of the JavaScript code can be compiled by TypeScript complier. [3] The complier set a base for a smooth migration of a project to TypeScript and it integrates well with the JavaScript ecosystem. Web tools like TypeScript can be easily upgraded and maintained during the process of mobile application development. In Ionic 2, TypeScript's static types can be used to annotate the code. Definition files can be added as well for Ionic 2 and will be getting code completion in the editor.

2.3.3 Sassy CSS .scss

Sass is a stylesheet language is used by Ionic. It is compiles to CSS. Sass is like CSS, but with additional features such as mixins, loops and variables. [4] It is a commonly known as the superset of CSS3's syntax. Ionic is built on top of it and styling is done on a single file to make theming easier. The responsive grid in SCSS enable the built mobile application to run flawlessly in various sizes of mobile devices. SCSS is also known as SASS. Typically, these two has the same functions and are similar.

Figure 5. Comparison between scss and css

2.3.4 JavaScript Object Notation .json

JSON is a syntax for storing and exchanging data. As Ionic 2 uses Typescript, using JSON to exchange data in between components will be more convenient. JSON is also lightweight and is easily read or write by human. [5] It is also easy for devices to parse and generate. This project make use of JSON to store most of the information, serving as a database.

2.3.5 Ionic Component

Ionic 2 applications are made of high-level building blocks called components. [6] A quickly construction of an interface for the application can be done by components. Ionic includes several components, such as popups, modals and cards. There are about 28 different components to be added, with more coming up over the time.

Ionic Components	
Action Sheets	Menus
Alerts	Modals
Badges	Navigation
Buttons	Popover
Cards	Radio
Checkbox	Range
DateTime	Searchbar
FABs	Segment
Gestures	Select
Grid	Slides
Icons	Tabs
Inputs	Toast
Lists	Toggle
Loading	Toolbar

Table 1. List of Ionic Component

2.3.6 API

API, also known as Application Programming Interface, is a set of functions and procedures that allow the creation of applications which access the features or data of an operating system, application, or other service. [7] API specifies how software components interact. API are often used during programming GUI components. API acts like a smart communication portal allow user to complete the action without leaving the original place.

2.3.7 Ionic Native

Ionic Native acts as a TypeScript wrapper for Cordova plugins to enable native functionalities into Ionic 2 mobile application. Some of the popular Ionic Native that developer used are Geolocation (GPS), camera, contacts and Google Maps.

2.4 Node.js

Node.js is a JavaScript runtime that is execute on the server-side instead of the traditional client side. It is an open-source and was released on the 27th May 2009. [8] Node.js allows the creation of Web servers and networking tools using JavaScript and a collection of "modules" that handle various core functionality. Modules are provided for file system I/O, networking (DNS, HTTP, TCP, TLS/SSL, or UDP), binary data (buffers), cryptography functions, data streams, and other core functions. Node.js's modules use an API designed to reduce the complexity of writing server applications. [9]

Node.js applications can run on Linux, macOS, Microsoft Windows, NonStop and Unix servers.

Figure 6. node.js

Node.js NPM was introduced on January 2010 which is the package manager for Node.js packages. Popular package includes Cordova, Ionic, React, Angular etc.

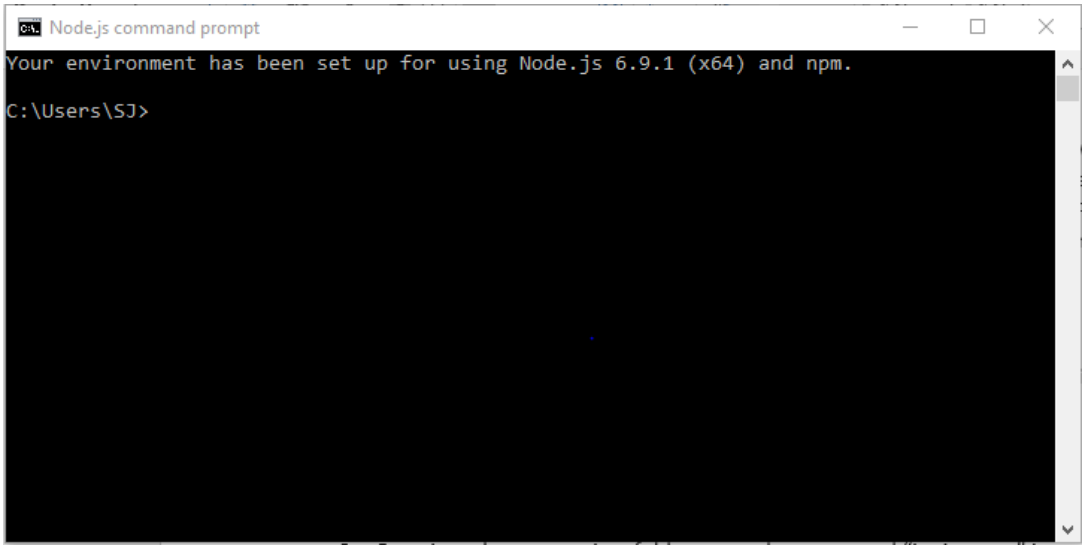

Figure 7. Command-line user interface (CLI) Node.JS

2.5 Cordova

Apache Cordova formerly known as PhoneGap. [10] Cordova is a mobile application distribution network that Ionic built on top of it. It is a wrapper for CSS3, HTML5 and JavaScript to build into a mobile application. Hybrid application can be done with Cordova, however native components are not able to function in the early release of Cordova. Cordova is able to support some native features in the latest update.

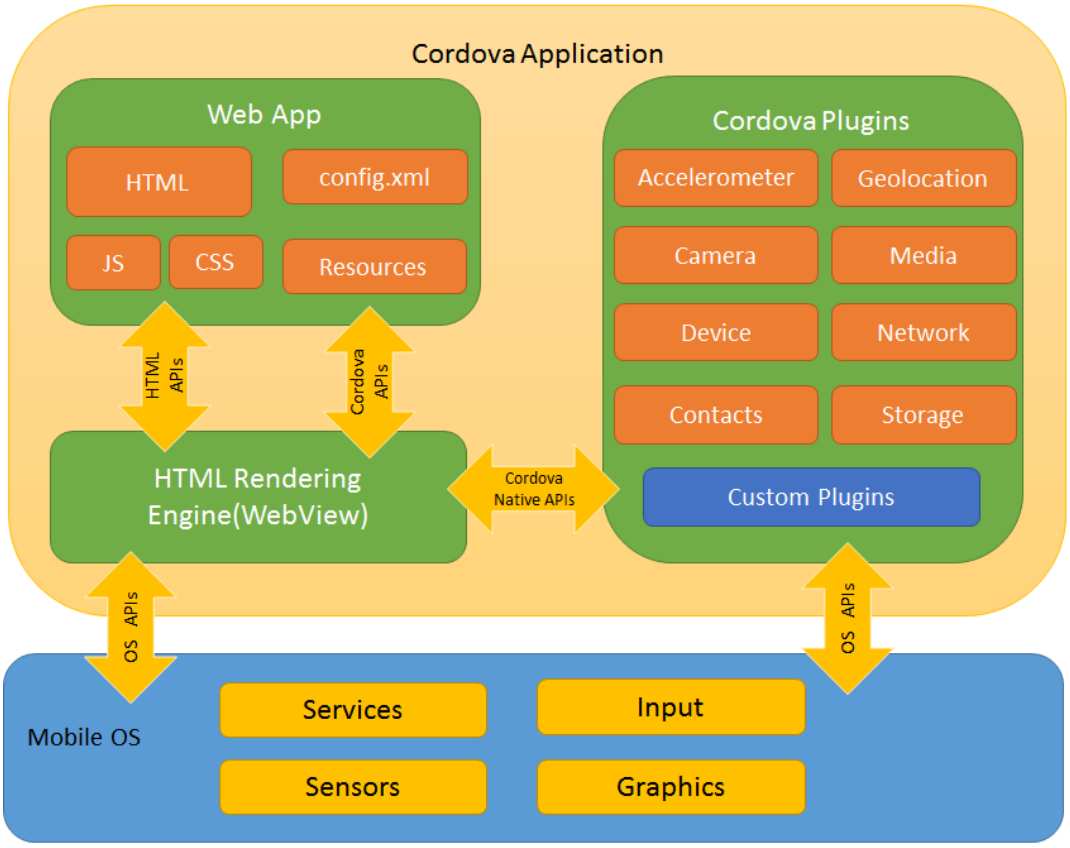

Figure 8.view of the Cordova application architecture

2.6 Android SDK

A software development kit that enables developers to create applications for the Android platform. The Android SDK includes sample projects with source code, development tools, an emulator, and required libraries to build Android applications. Applications are written using the Java programming language and run on Dalvik, a custom virtual machine designed for embedded use which runs on top of a Linux kernel. [11]

Figure 9.Icon of Android Studio inclusive of SDK tools

2.7 Xcode (iOS SDK)

A software development kit (SDK) released by Apple that allows people to write applications for both the iPhone and the iPod Touch. The iPhone SDK includes the Xcode IDE, Instruments, iPhone simulator, frameworks and samples, compilers, Shark analysis tool, and more. [12] To download the iPhone SDK Apple ID is a must. Developers have long insisted that by providing a development platform for iPhone is one way that Apple can work towards preventing people from hacking the operating system used by the device, which is a stripped down version of Apple's own OS X operating system.

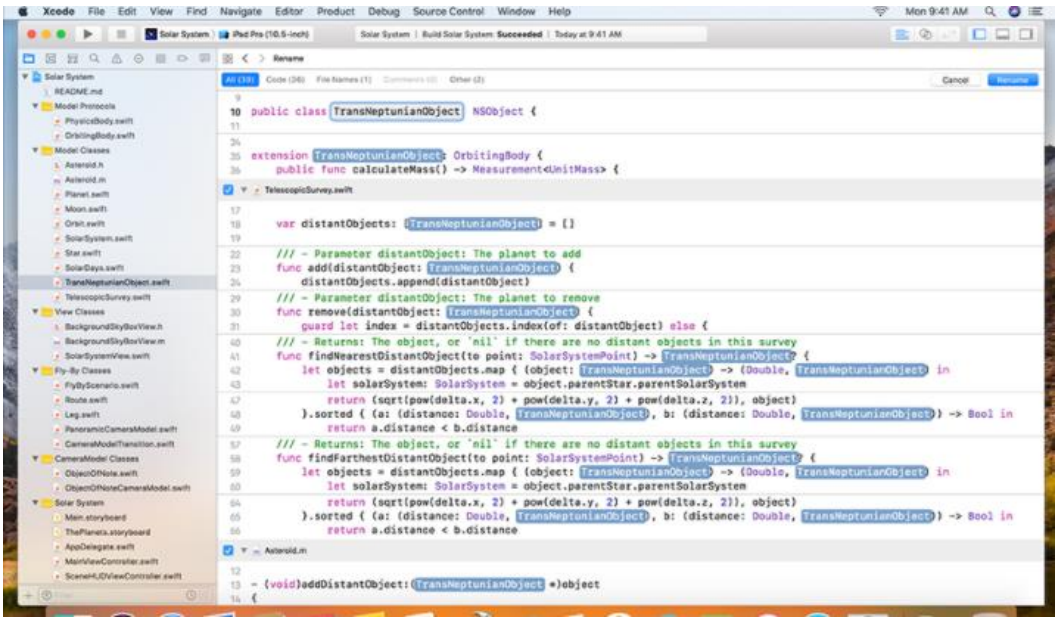

Figure 10. Xcode

Chapter 3 Existing Application

3.1 Tourist Applications

As of 2017 August, the number of available tourism applications for Singapore with rating above 4 out of 5 is three. Two applications out of the three is outdated, and the last being too commercialized which is not user-friendly. The applications that are available are ‘Singapore City Guide Trip.com’, ‘Singapore Guide by Triposo’, and ‘Singapore Travel Guide-Tourias’. Bigger cities, such as Barcelona, Vancouver, London etc., have their tourist application supported by the government. Moreover, the cities have a significant huge landscape and various iconic visitor spots which will be tougher to develop.

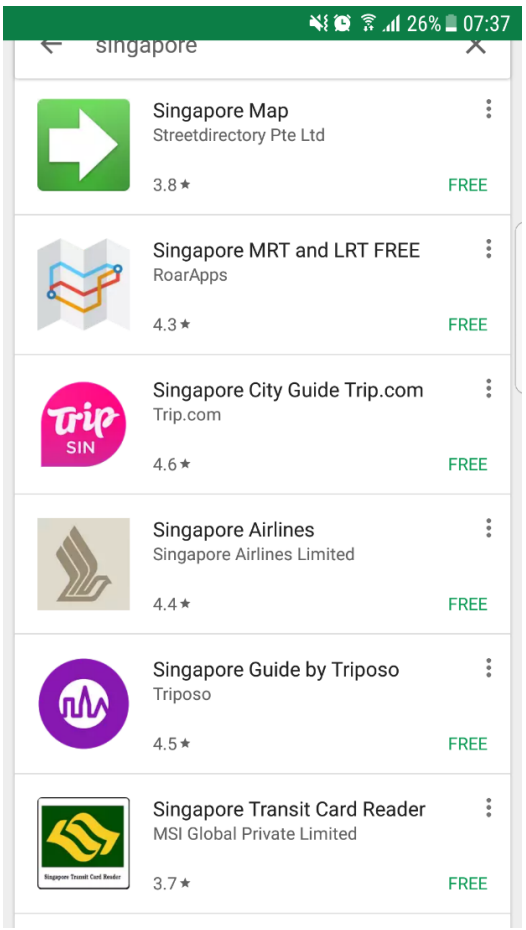

Figure 11. Playstore Singapore tourism-friendly application

3.1.1 Singapore City Guide Trip.com

Singapore City Guide by Trip.com is a clear and well-organized application. With Trip.com as the main frame, it has already been stabled. However, due to the diversities of the application, Singapore City Guide by Trip.com has lost the local Singaporean touch which might not be appealing for majority of the market in the tourism sector. Also, this application is too commercialize and lost the true purpose of helping tourist in Singapore. Currently this application is the leader in the Singapore tourism market, but with a total of about 10,000-50,000 downloads, it is still not an ideal application yet.

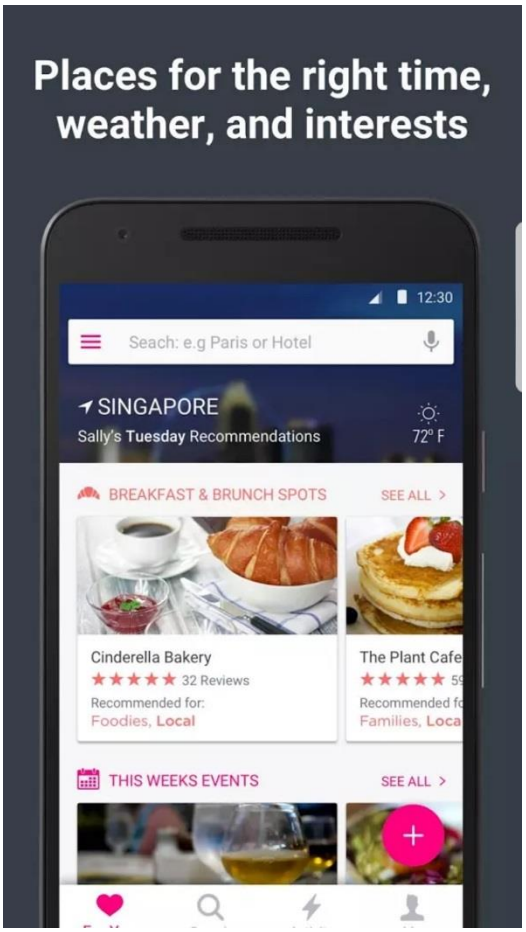

Figure 12. Singapore City Guide trip.com screenshot preview

3.1.2 Singapore Guide by Triposo

Singapore Guide by Triposo was a good mobile application. The application has not been updating consistently and it is not supported by newer version of Android and iOS. The lack of support cause user to have a bad experience. First user impression is also important but the Singapore Guide by Triposo is in short of this. The design of Triposo is rather bad. The colour choices do not shows Singapore's unique style. The downloaded record is 3 times less than Singapore City Guide by Trip.com.

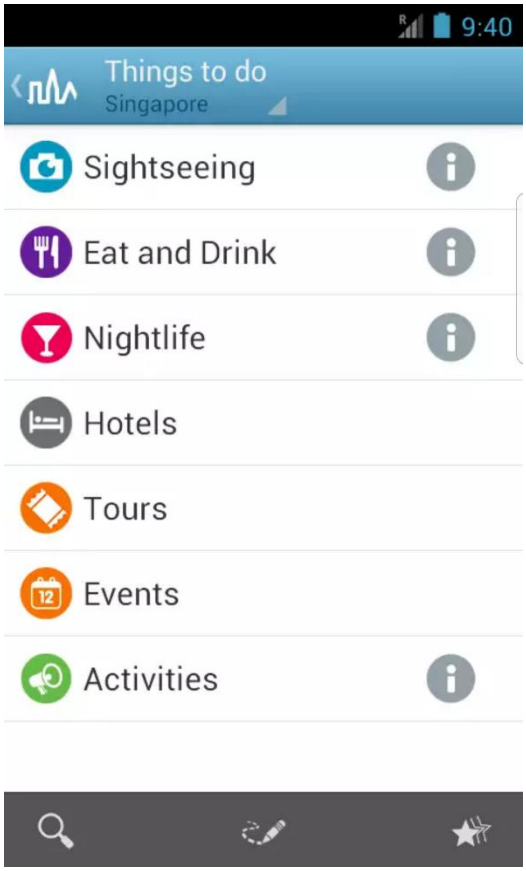

Figure 13. Singapore City Guide trip.com screenshot preview

3.1.3 Singapore Travel Guide-Tourias

Similar with the Singapore Guide by Triposo, Tourias has not been updating the application for more than a year, with the last update being July 2016. It managed to have a great user base many year back but due to the inactivity and lack of support, causing the application to go down.

Chapter 4 Materials and Procedures

4.1 Methodology

An iterative method and incremental build model for software development approach was chosen to apply for EzVacation@SG application. The constructing of this application come in detailed planning.

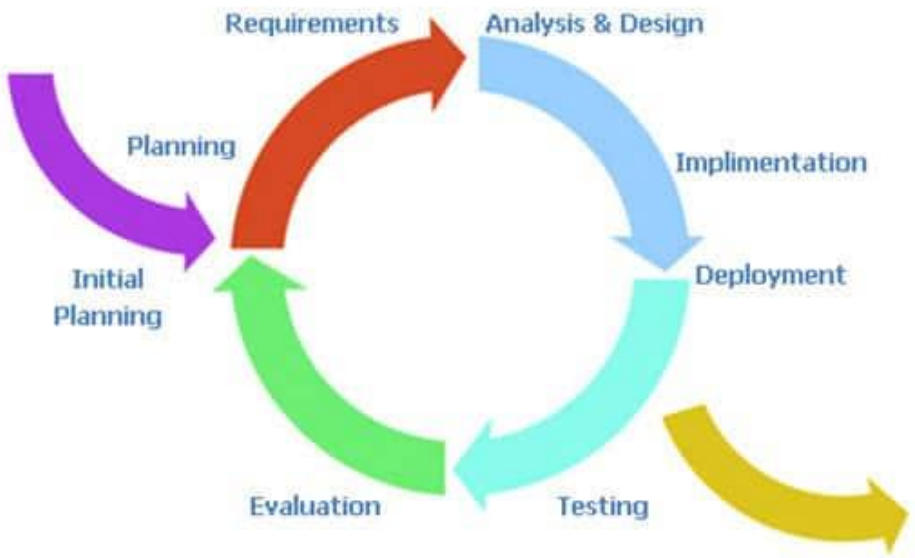

Figure 14. Iterative method and incremental build model for EzVacation@SG

Before the start of the development of the application, many researches and studies were done. EE4717 Web Application Design was studied at the initial stage of this project. The basics of website programming such as the Html, JavaScript, SQL, PHP and CSS are learned during the process. At the same period, *Learning PHP MySQL Javascript CSS HTML5 Robin Nixon by O'Relly* was used as a material for an efficient read up. Udacity.com and W3schools.com served as a good guide and reference during the development of simple demonstration web application. Ionic 2 learning were started during the 2nd phase of the project. Ionic 2 basic components and features were studied via the Ionic 2 online courses in the official website. Service and component APIs were learned through various examples from the official ionic community forum. Typescript is studied through Microsoft partner site: https://www.typescriptlang.org/docs/ and the family of Angular JS2 and Apache Cordova are also learnt through their official website. The transition to Ionic 2 was not easy due to additional of components that need to be understand

well in order to start the development of mobile application. Overall, learning Ionic 2 is a deep-learning curve for beginner, guidance from mentor can speed things up. Understandings and practices are the keys toward a good development of Ionic 2 Mobile Application.

4.1.1 Preparation of material

Materials, hardware and software, are important for the development of Ionic 2 mobile application.

In this project, a total of two computers are used, one being a personal desktop and the second being a laptop that is connected to a domain. Due to the restriction of external software install on the laptop, the personal desktop is served as the host computer for the Ionic 2 mobile application developments. Chrome Remote Desktop will need to be installed on the host computer to enable the slave computer to be able to work on the development remotely.

The software that are used in this mobile application development are Windows 10 Pro, Node.js, Ionic 3 Framework (previously known as Ionic 2), Cordova CLI, Microsoft Visual Studio Code, Notepad++, Android Studio and SDK, Xcode, Chrome Browser, Chrome Remote Desktop and Dropbox. All software is updated to latest version when possible.

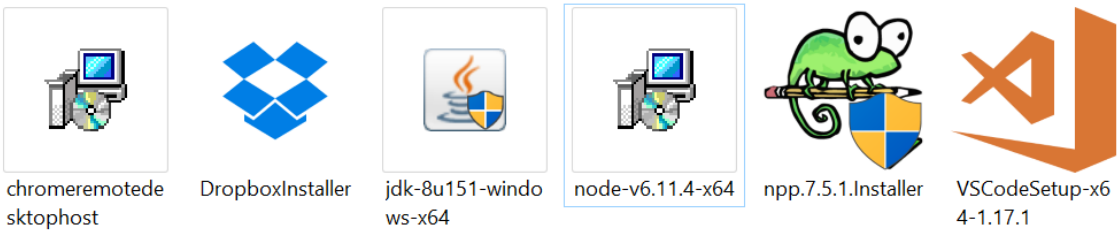

Figure 15. Software required

1. Installing of Ionic 2 is through Node.js, a command line "`npm install -g cordova ionic`" is used to install Ionic 2 together with Cordova. Another command line "`ionic start ezvac tabs`" , a new project named as ezvac with the default 'tabs' template included. All the files will be stored at local drive in which in this case under "*user\dropbox\school\fyp\ezvac*".

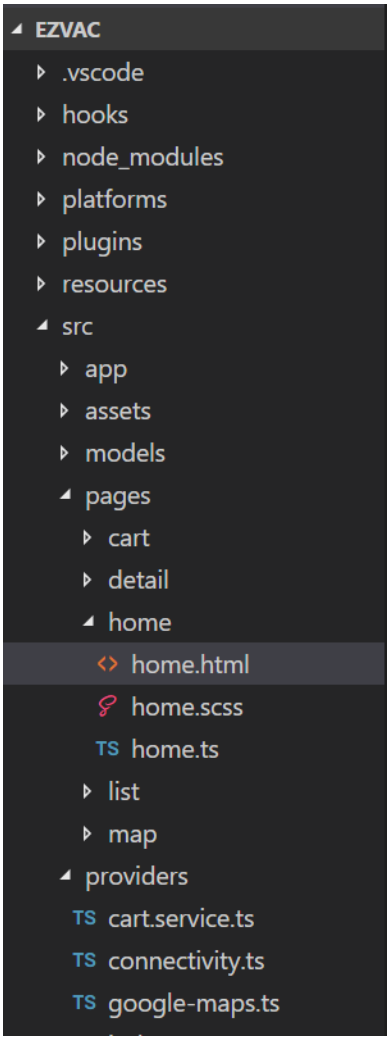

Figure 16. Directory of EzVacation@SG

2. To start the project, navigation to the directory will need to be executed. Command "cd dropbox\school\fyp\ezvac" is executed to move to the directory and the command "ionic serve" is entered to compile the file and the sample mobile application that is under development will be shown on the browser. The sample mobile application has a real-time update function and when changes are made in the project directory, it will auto update and refresh immediately. The application will be compiled unsuccessfully and any error will be reflected if there is any wrong argument.

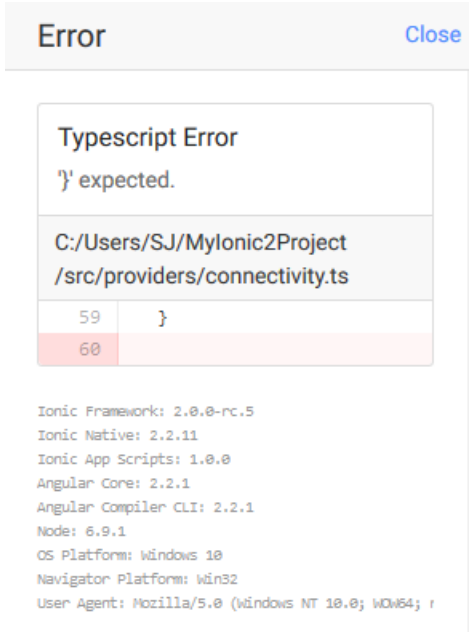

Figure 17. Common Error Message: Typescript Error

3. Notepad+ is used to add, amend and edit the files inside the folder. The most frequent edited files are HTML, SCSS, TypeScript which are located under 'src'(Source) folder. Notepad+ was recently replaced by Visual Studio Code to edit the coding. Visual Studio Code is a lightweight code editor with fantastic features. It provides support for Node.js and for HTML, SCSS, TypeScript codes and these scripts can be executed from within Visual Studio Code. With Visual Studio Code, different editors are not required for multiple programming languages.

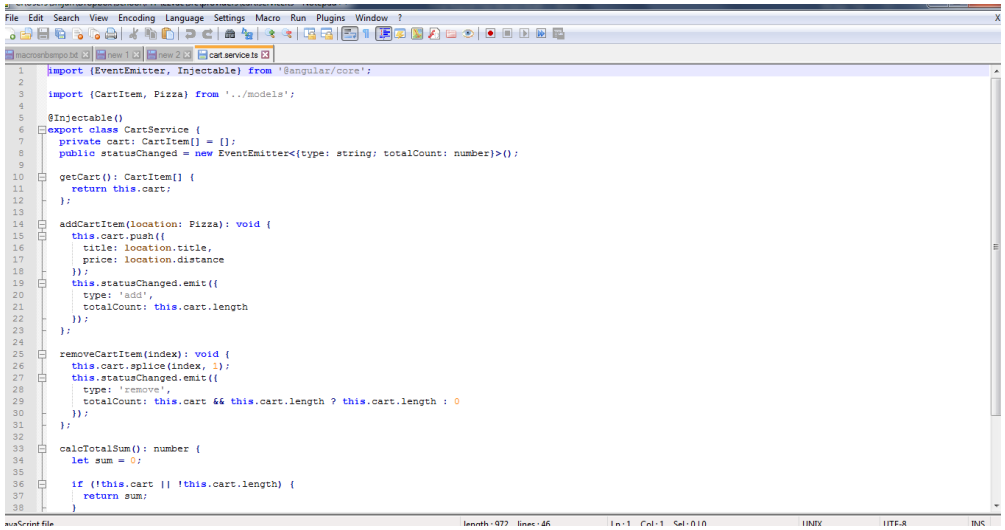

Figure 18. Editing on Notepad++

4.1.2 Implementing new page

The procedure of inserting a new page Ionic2 framework is more tedious as compared to adding a normal HTML. By entering the command “ionic g page Page1”, a folder named Page1 will be generated with the basic files inside it. The basic files include the HTML, SCSS and TypeScript with predefined code. The generated page can then be linked up with the mainframe by editing the base home page.

The pages in ionic2 is an application focused platform, it employs the push-stack-pop method. A page is supported by files .html (HTML), .ts (TypeScript), and .scss (Style Sheet) together.

4.1.3 Inserting google map

To insert google map, the API developer key need to be generated from Google. The google map API developer key is generated from https://developers.google.com/maps/. Once logged in, click on the GET A KEY from Google Maps JavaScript API. The generated key will be ready to use. The key is free for a limit and the limit is huge enough for testing purpose.

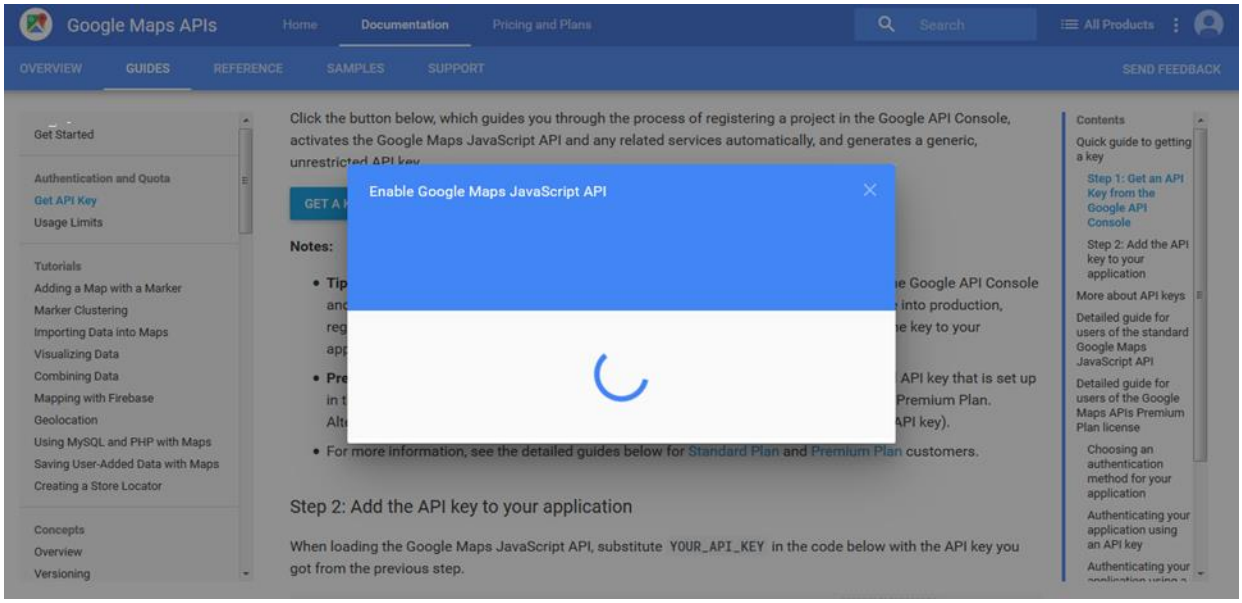

Figure 19. Generating google map API developer key

Adding script together with the API key is to be inputted into src\map.ts. Once API is ready for the project, the map components are added into the page by modifying the class definition and

etc. The main file for the implementation of Google map is google-maps.ts which is responsible for the map to appear on our mobile application. Two other files, map.html and map.ts, are responsible for displaying the map together with the markers. Geolocation component is also included to make use of our GPS location. In order to make it appear well on the page, styling of mobile application need to be done.

```
import { Component, ElementRef, ViewChild } from '@angular/core';
import { Locations } from '../../providers/locations';
import { GoogleMaps } from '../../providers/google-maps';
import { NavController, Platform } from 'ionic-angular';

@Component({
  selector: 'page-map',
  templateUrl: 'map.html'
})
export class MapPage {

    @ViewChild('map') mapElement: ElementRef;
    @ViewChild('pleaseConnect') pleaseConnect: ElementRef;

    constructor(public navCtrl: NavController, public maps: GoogleMaps, public platform: Platform, public locations: Locations) {

    }

    ionViewDidLoad(){

        this.platform.ready().then(() => {

            let mapLoaded = this.maps.init(this.mapElement.nativeElement, this.pleaseConnect.nativeElement);
            let locationsLoaded = this.locations.load();

            Promise.all([
                mapLoaded,
                locationsLoaded
            ]).then((result) => {

                let locations = result[1];

                for(let location of locations){
                    this.maps.addMarker(location.latitude, location.longitude);
                }

            });

        });

    }

}
```

Figure 20. Map.ts calling GoogleMaps and Locations

4.1.4 Creating list of places of interest and placing of markers

The list is created with the information such as Title, Description, Latitude, Longitude. The information is stored inside JSON file. 10 sample places of interest are included in the application. The list can be updated to include more places of interest. User will be able to select the place of interest from the list or description of the place of interest. Decision will need to be made whether the information should be store locally in the application or on a webserver which will be better for updating of the database files in the future

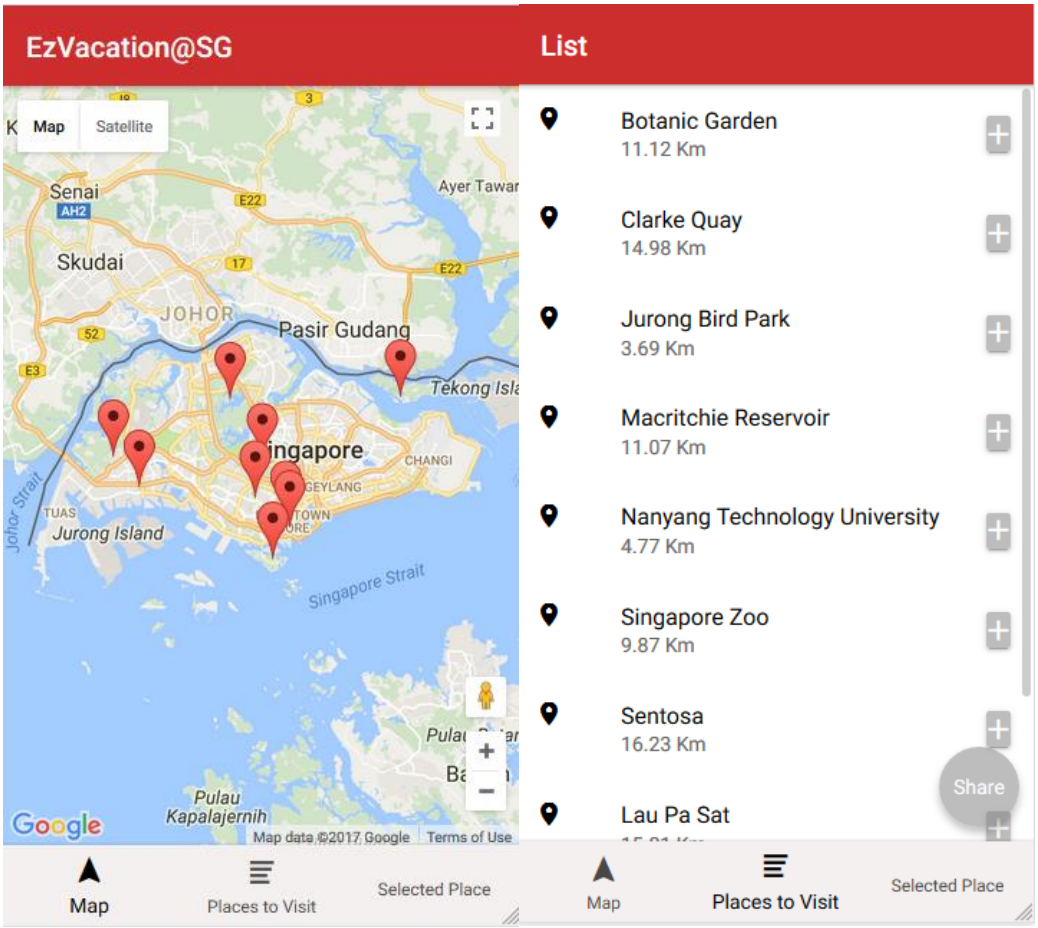

Figure 21. Map and Places to Visit Tabs

4.1.5 Calculation formula

The formula is generated by combining and stacking Haversine Formula to calculate distances together with some variables. Haversine formula is mainly used for navigation, giving great-circle distances between points on the earth sphere. Although Singapore is small to calculate distance through Haversine formula, it is a good practice to implement this for future work. The mobile application made use of the coordination of information of the place to plot in the map. Distance between the user location and the place of interest is calculated using Haversine formula. The distance can be fixed if global positioning system is not available in the device.

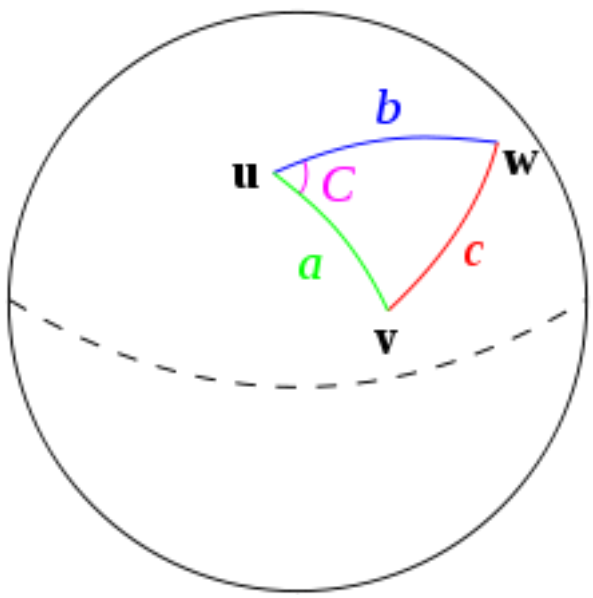

Figure 22. Haversine formula model

4.1.6 Design and user navigation

This mobile application is designed minimalistic which is an in-trend design that mobile users love. The user interface is improved through simply decluttering the interface. When there is lesser colour to focus on, navigate within the interface is eased without clicking on an incorrect link or otherwise getting struck in the navigation. The actual design is trendier compared to the conceptual design. Most of the function is not shown at the user interface and is actually processed in the background. Icon served as a great guide to user. Function like adding the place to selected place is represented by a simple '+' button and clicking on the listing object will show more information about that place. The functions are supported by TypeScript in the background. TypeScript are usually located in the local HTML page or the service component side. TypeScript is also import and export to be used together with other TypeScript. TypeScript communication needs to be handled carefully to ensure conciseness of the calling of components. Selected places list tab is then being processed and an itinerary with required information is generated and the user can choose to download it in PDF file format. User will have some options to either share it or export it.

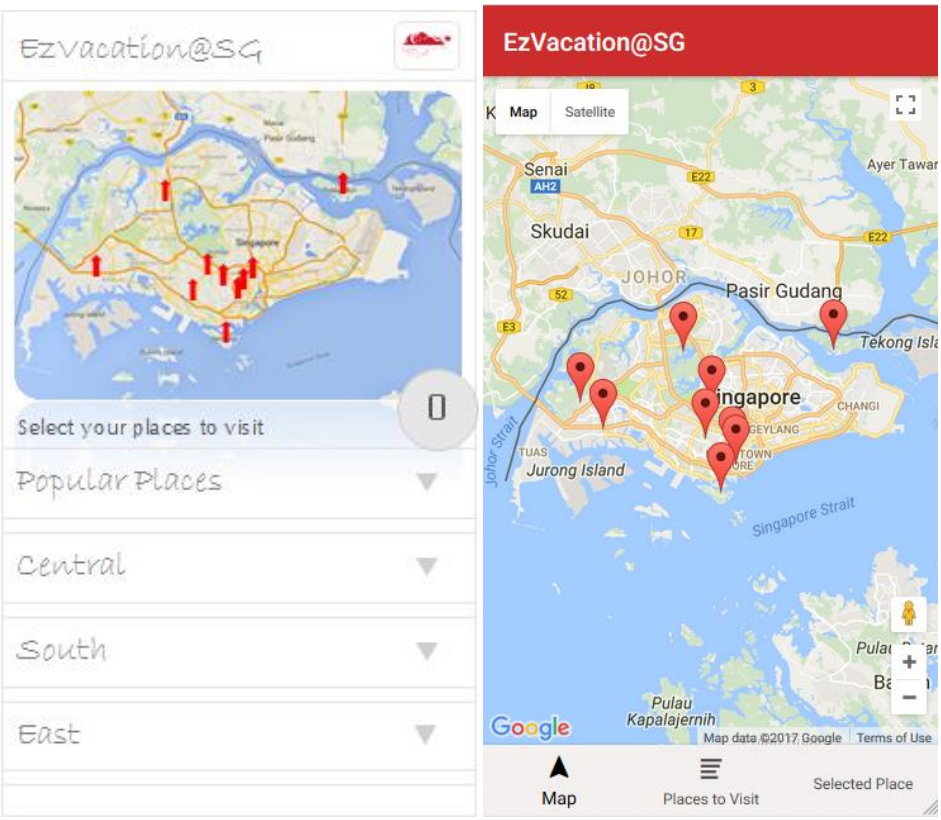

Figure 23. Comparison of conceptual design (sketchup) with the actual design

4.1.7 Troubleshooting

The project consists of many working files which are interrelated and correlated with the other codes. During combination of different function, a single fault will lead to the other component malfunctioning. The software used in developing the need to be updated when error occurred. Although this is not always but it is of good practices. The troubleshooting guide *https://ionicframework.com/docs/troubleshooting/* is followed when error occurred. If the error is not solved, a flow chat of troubleshooting will be jotted down, and then different methods will be performed to solve the issue. In the scenario when there is no method that is working, the last resort is to seek help from expert or engage into discussion with the Ionic Community. The troubleshooting time taken for this project compared to the actual coding is 3:2. Troubleshooting took time more than the actual coding but while doing the troubleshooting, knowledge and experience are acquired which will eventually improve the coding ability of the programmer.

4.1.7.1 Troubleshooting of error

Some common errors include the syntax error, typescript error and the API component calling error. Some of the error encountered in this project is Undefined Member Variables. The error occurred by attempting to access a member variable that hasn't been defined or hasn't been defined properly. The problem is solved by tracing through the error code line and identify the member is setup correctly. Another common problem is Component, Pipe, or Directive being not declared. Component, Pipe and Directive must be declared in the declarations array of the app.module.ts file.

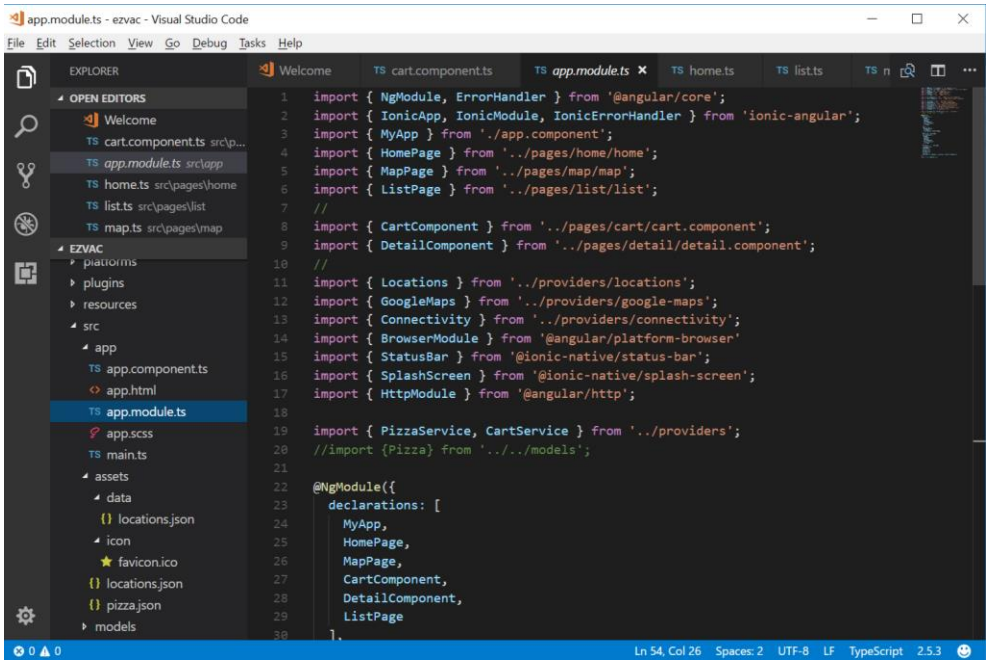

Figure 24. app.module.ts

4.1.7.2 Troubleshooting on simulator and actual devices

Troubleshooting is mostly done on simulator before exporting to actual device. The process between exporting to devices and testing on devices need significant time to troubleshoot. One error encountered was the unable to find required JDK (Java Development Kit) software to build the application. The reason is because there is multiple version of JDK installed on the PC and only one of the version is accepted. The solution is to remove all existing JDK on the PC and reinstall the only version that is accepted. Error also occurred in the native devices, some API component do not show up on the screen and lead to more error due to that. The device troubleshooting process is still undergoing.

4.1.8 Exporting to native devices – iOS & Android

4.1.8.1 Apple iOS

Using the command "*ionic run ios*" resulting in an error "You cannot run iOS unless you are on Mac OSX.". For most PC user, access to an Mac OSX is very limited. A possible solution is to install a virtual Mac OSX which will be using to export the app into iOS package. A Macbook air and an iPhone is loaned to deploy the application to the device, however due to some technical license issue, iPhone deployment is on a halt.

4.1.8.2 Google Android

Android publishing is less complicated. In order to build the package, Android studio and SDK tools need to be installed. After that, the command "*ionic cordova build --release android*" is used to create the apk. Java Development Kit will need to be installed and JAVA_HOME will need to be directed in order for the unsigned apk to be exported out. Testing of the application can be done once installed on the device.

Figure 25. Successfully built APK

4.2 Test

Mobile application testing is extremely important because of the following reasons: mobile application testing is required to point out the issues and errors that were made during the development phases. Mobile application testing is essential since it makes sure of the Customer's reliability and their satisfaction in the application in order to fulfill user acceptance criteria. The mobile application undergoes various type of testing to ensure that what is being developed does what it is supposed to do. The testing method employed includes: Functional Testing, Performance Testing, Usability Testing and Installation Testing. Some minor bugs and errors persisted in the program during the testing phases. Also, some errors appear intermittently and simulate the occurrence has to be done rapidly for troubleshooting before the testing.

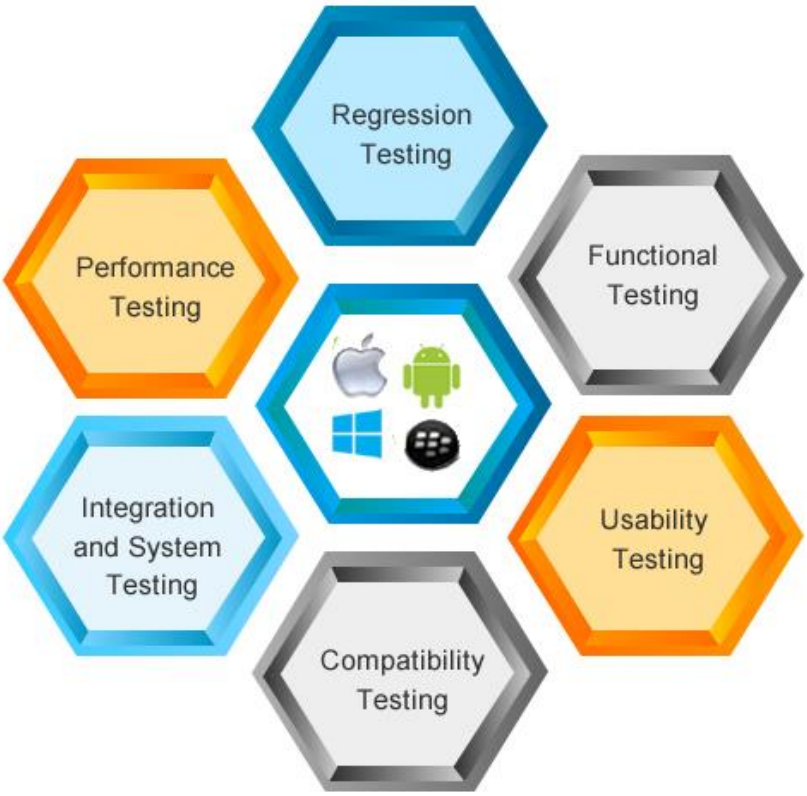

Figure 26. Mobile application testing methods

4.3 Summary and results

A total of 14 months is spent on the project with an average of 4 hour per week. 50% of the time is spent for studies and researches, while another half is the actual development of the mobile application. The test experiments are done on Chrome browser developer tools simulation.

Functional Testing	**Results**
Loading the map	Successful
Navigating to 'Places of interest' tab	Successful
Click titles for details	Successful
Share button	Successful but actual link for the social media
Click '+' icon (single time) Click '+' icon (multiple times)	Successful Successful
Navigating to 'Selected Places' tab	Successful
Show empty when there is no selected place	Successful
Single entry of selected place Multiple entries of selected places	Successful Semi-bugged
Click generate button	Successful
Itinerary generated	Successful
Export to PDF file	Fail

Performance Testing	**Results**
Low Battery	Able to on
No GPS	Less accuracy for calculation
No network	Map cannot be loaded
Low memory	Usable

The testing done shows promising result. The main objective is met and the web application can perform the main requirement. It is

Chapter 5 Discussion

5.1 Limitation and Recommendation

The project is started with zero knowledge in Ionic 2 or web application. Time and efforts are required as this process is steep learning curve. In order to speed up the progress of learning, a mentor with the expertise in Ionic 2 will be of great help. On top of that, learning within a community will be good as idea can be shared and helping each other in it.

The laptop that is used in this project is tied to a domain, due to that 3rd party software cannot be installed. A lightweight fast personal laptop is recommended to have. Coding can be done easily whenever new solution or idea is discovered. Another workaround is to use a host computer that is on for 24/7 hour. This way the portable computer can be used to remote control the host computer to program. This method requires a fast and reliable internet connection.

Focusing on a single project is difficult especially for part time student. Supports from school, classmates, project supervisor, work colleagues are important, project developer will be more motivated to continue on despite difficulties. Discipline to work on the project is also crucial, regular breather is recommended to ensure the development continue to flow.

Although this project can be development singlehanded. A good mobile application development is better to do with a team. With a good coordinator, a good mobile application can be efficiently developed.

5.2 Conclusion

Hybrid mobile application today required a high level of functionality and feature as mobile users demand a high quality yet simple application to use, Ionic 2 is a good framework to work on base on the market trend. The mobile application built for this project has been discussed very thoroughly on how test experiments are constructed and performed to get substantial results. In conclusion, the project is a success and was tested repeatedly to ensure a constant result by users. The features of the mobile application are constantly update on. The insurance of the quality of the mobile application is done by regular compatibility test performance. Although, the project achieved the objective, the application has more room for improvement.

Chapter 6 Future Development

Even though the project is proven to be successful. As an application, there is plenty of possibilities for the future development.

1. New algorithm can be improved to show the travel distance instead of straight line distance between two locations.
2. Login feature to store history and record in the cloud system so users will be able to retrieve their account once logged in
3. User moment posting, and user review can be implemented.
4. Recommendation or affordable ticket deals can be implemented for to collaborate with local merchant.

A tedious study on obtaining deep learning together with the google map API on the live transport guide can be integrated into the system to understand the input response of the user. Complex machine learning and data collection from the active can be used to improve the development. Self-learning mobile application can also report back on the market trend which will also be able to assist Singapore Tourism Board on their next move on marketing Singapore as a global city.

References

[1A] Changi Air Traffic Statistics Passenger Movements At Changi Airport

http://www.changiairport.com/corporate/about-us/traffic-statistics.html

[1B] Changi Airport, SIA, STB to invest S$34m to promote Singapore as travel destination

http://www.channelnewsasia.com/news/singapore/changi-aiport-sia-stb-to-invest-s-34m-to-promote-singapore-as/3683522.html

[2] Ionic (mobile app framework) - Wikipedia

https://en.wikipedia.org/wiki/Ionic_(mobile_app_framework)

[3] TypeScript - JavaScript that scales.

https://www.typescriptlang.org/

[4] How to use Sass Mixins and Loops

https://medium.freecodecamp.org/sass-mixins-and-loops-171122499a2

[5] Introducing JSON

http://www.json.org/

[6] Ionic Component Documentation - Ionic Framework

https://ionicframework.com/docs/components/

[7] Dictionary

http://www.dictionary.com

[8] Node.js – Wikipedia

https://en.wikipedia.org/wiki/Node.js

[9] WHY IS NODE.JS POPULAR IN REAL TIME APPLICATIONS

https://www.freelancinggig.com/blog/2017/01/26/node-js-popular-real-time-applications/

[10] Apache Cordova

https://cordova.apache.org/

[11] Meet Android Studio

https://developer.android.com/studio/intro/index.html

[12] iPhone SDK

https://www.webopedia.com/TERM/I/iPhone_SDK.html

Appendix

SRC\APP

app.component.ts

```
import { Component } from '@angular/core';
import { Platform } from 'ionic-angular';
import { StatusBar } from '@ionic-native/status-bar';
import { SplashScreen } from '@ionic-native/splash-screen';

import { HomePage } from '../pages/home/home';
@Component({
  templateUrl: 'app.html'
})
export class MyApp {
  rootPage:any = HomePage;

  constructor(platform: Platform, statusBar: StatusBar, splashScreen:
SplashScreen) {
    platform.ready().then(() => {
      // Okay, so the platform is ready and our plugins are available.
      // Here you can do any higher level native things you might need.
      statusBar.styleDefault();
      splashScreen.hide();
    });
  }
}
```

App.html

```
<ion-nav [root]="rootPage"></ion-nav>
```

app.module.ts

```
import { NgModule, ErrorHandler } from '@angular/core';
import { IonicApp, IonicModule, IonicErrorHandler } from 'ionic-angular';
import { MyApp } from './app.component';
import { HomePage } from '../pages/home/home';
import { MapPage } from '../pages/map/map';
import { ListPage } from '../pages/list/list';
//
import { CartComponent } from '../pages/cart/cart.component';
import { DetailComponent } from '../pages/detail/detail.component';
//
import { Locations } from '../providers/locations';
import { GoogleMaps } from '../providers/google-maps';
import { Connectivity } from '../providers/connectivity';
import { BrowserModule } from '@angular/platform-browser'
import { StatusBar } from '@ionic-native/status-bar';
```

```
import { SplashScreen } from '@ionic-native/splash-screen';
import { HttpModule } from '@angular/http';

import { PizzaService, CartService } from '../providers';
//import {Pizza} from '../../models';

@NgModule({
  declarations: [
    MyApp,
    HomePage,
    MapPage,
    CartComponent,
    DetailComponent,
    ListPage
  ],
  imports: [
    BrowserModule,
    HttpModule,
    IonicModule.forRoot(MyApp)
  ],
  bootstrap: [IonicApp],
  entryComponents: [
    MyApp,
    HomePage,
    MapPage,
    CartComponent,
    DetailComponent,
    ListPage
  ],
  providers: [    StatusBar,
    SplashScreen,
  GoogleMaps,
  Connectivity,
  CartService,
  PizzaService,
  Locations,
  {provide: ErrorHandler, useClass: IonicErrorHandler}]
})
export class AppModule {}
```

main.ts

```
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';

import { AppModule } from './app.module';

platformBrowserDynamic().bootstrapModule(AppModule);
```

src\assets\data

location.json

```
{
    "locations": [

        {
            "id": 1,
            "title": "Botanic Garden",
            "latitude": 1.309460,
            "longitude": 103.817200,
            "description": "Hello"
        },
        {
            "id": 2,
            "title": "Clarke Quay",
            "latitude": 1.290000,
            "longitude": 103.846500,
            "description": "Hello"
        },
        {
            "id": 3,
            "title": "Jurong Bird Park",
            "latitude": 1.319600,
            "longitude": 103.706500,
            "description": "Hello"
        },
        {
            "id": 4,
            "title": "Macritchie Reservoir",
            "latitude": 1.34500,
            "longitude": 103.82420,
            "description": "Hello"
        },
        {
            "id": 5,
            "title": "Nanyang Technology University",
            "latitude": 1.34834,
            "longitude": 103.68170,
            "description": "Hello"
        },
        {
            "id": 6,
            "title": "Singapore Zoo",
            "latitude": 1.40352,
            "longitude": 103.79340,
            "description": "Hello"
        },
```

```
        {
            "id": 7,
            "title": "Sentosa",
            "latitude": 1.25088,
            "longitude": 103.83410,
            "description": "Hello"
        },
        {
            "id": 8,
            "title": "Lau Pa Sat",
            "latitude": 1.28094,
            "longitude": 103.85030,
            "description": "Hello"
        },
        {
            "id": 9,
            "title": "Pulau Ubin",
            "latitude": 1.40566,
            "longitude": 103.95599,
            "description": "Hello"
        }
    ]
}
```

Src\models\

Cart-item-model.ts

```
export interface CartItem {
  title: string;
  price: number;
}
```

Pi.model.ts

```
export interface Pizza {
  id: number;
  title: string;
  latitude: number;
  longitude: number;
  description?: string;
  distance: number;
}
```

Src\pages\cart\

Cart.component.html

```
<ion-header>
```

```
  <ion-navbar>
    <button menuToggle ion-button icon-only>
      <ion-icon name="menu"></ion-icon>
    </button>
    <ion-title>Selected Place</ion-title>
  </ion-navbar>
</ion-header>

<ion-content>
  <ion-list *ngIf="cart.length">
    <ion-item-sliding *ngFor="let item of cart; let index=index">
      <ion-item>
        <h2>{{item.title}}</h2>
        <p>{{item.price | number:'1.0-3'}}</p>
      </ion-item>
      <ion-item-options>
        <button (click)="removeFromCart(index)" color="danger" ion-button
icon-left>
          <ion-icon name="trash"></ion-icon>
          Delete
        </button>
      </ion-item-options>
    </ion-item-sliding>
  </ion-list>
  <p *ngIf="!cart.length" padding text-center>
    There is nothing in your list!
  </p>
</ion-content>

<ion-footer>
  <ion-toolbar>
    <ion-row>
      <ion-col width-50>
        <b>Total Selected Place:</b>
      </ion-col>
      <ion-col width-50 text-right>
        <b>{{calcTotalSum()}}</b>
      </ion-col>
    </ion-row>
  </ion-toolbar>
</ion-footer>
```

Cart.component.ts

```
import {Component, OnInit} from '@angular/core';

import {AlertController} from 'ionic-angular';
```

```
import {CartService} from '../../providers';
import {CartItem} from '../../models';

@Component({
  templateUrl: 'cart.component.html'
})
export class CartComponent implements OnInit {
  cart: CartItem[] = [];

  constructor(private cartService: CartService, private alertCtrl:
AlertController) {}

  ngOnInit(): void {
    this.cart = this.cartService.getCart();
  }

  ionViewDidEnter(): void {
    if (this.cart.length) {
      return;
    }

    let alert = this.alertCtrl.create({
      title: '<b>Your list of places is empty!</b>',
      subTitle: 'Add interesting places to your list.',
      buttons: ['OK']
    });
    alert.present();
  }

  calcTotalSum() {
    return this.cartService.calcTotalSum();
  }

  removeFromCart(index: number): void {
    this.cartService.removeCartItem(index);
  }
}
```

Src\pages\details

Detail.component.html

```
import {Component, OnInit} from '@angular/core';

import {NavParams} from 'ionic-angular';
import 'rxjs/add/operator/toPromise';

import {PizzaService} from '../../providers';
```

```
import {Pizza} from '../../models';

@Component({
  templateUrl: 'detail.component.html'
})
export class DetailComponent implements OnInit {
  pizza: Pizza;

  constructor(
    private navParams: NavParams,
    private pizzaService: PizzaService
  ) {}

  ngOnInit(): void {
    this.pizzaService
      .getPizza(this.navParams.get('id'))
      .toPromise()
      .then(pizza => this.pizza = pizza);
  }
}
```

Detail.component.ts

```
import {Component, OnInit} from '@angular/core';

import {NavParams} from 'ionic-angular';
import 'rxjs/add/operator/toPromise';

import {PizzaService} from '../../providers';
import {Pizza} from '../../models';

@Component({
  templateUrl: 'detail.component.html'
})
export class DetailComponent implements OnInit {
  pizza: Pizza;

  constructor(
    private navParams: NavParams,
    private pizzaService: PizzaService
  ) {}

  ngOnInit(): void {
    this.pizzaService
      .getPizza(this.navParams.get('id'))
      .toPromise()
      .then(pizza => this.pizza = pizza);
```

```
  }
}
```

Src\pages\home

Home.html

```
<ion-tabs color="light">
  <ion-tab [root]="tab1Root" tabTitle="Map" tabIcon="navigate"></ion-tab>
  <ion-tab [root]="tab2Root" tabTitle="Places to Visit" tabIcon="list"></ion-
tab>
  <ion-tab [root]="tab3Root" tabTitle="Selected Place" tabIcon="list2"></ion-
tab>
</ion-tabs>
```

Home.ts

```
import { Component } from '@angular/core';
import { MapPage } from '../map/map';
import { ListPage } from '../list/list';
import { CartComponent } from '../cart/cart.component';

@Component({
  selector: 'page-home',
  templateUrl: 'home.html'
})
export class HomePage {

  tab1Root: any = MapPage;
  tab2Root: any = ListPage;
  tab3Root: any = CartComponent;

  constructor(){

  }

}
```

Src\pages\list

```
<ion-header>

  <ion-navbar color="secondary">
    <ion-title>List</ion-title>
  </ion-navbar>

</ion-header>
```

```
<ion-content>

 <div [hidden]="!loading" text-center padding>
  <ion-spinner></ion-spinner>
</div>
<ion-refresher (ionRefresh)="doRefresh($event)">
  <ion-refresher-content
    pullingText="Refreshing..."
  >
  </ion-refresher-content>
</ion-refresher>

    <ion-list no-lines>
        <a ion-item (click)="openPizza(location.id);" *ngFor="let location of
locations.data" >
            <ion-avatar item-left>
                <ion-icon name="pin"></ion-icon>
            </ion-avatar>
            <h2>{{location.title}}</h2>
            <p>{{location.distance}} Km</p>
            <button item-right (click)="addToCart($event, location);"  ion-
button icon-only>
              <ion-icon name="md-add"></ion-icon>
            </button>
          </a>
    </ion-list>

 <ion-fab bottom right >
  <button ion-fab>Share</button>
  <ion-fab-list side="top">
    <button ion-fab>Facebook</button>
    <button ion-fab>Twitter</button>
    <button ion-fab>Youtube</button>
  </ion-fab-list>
  <ion-fab-list side="left">
    <button ion-fab>Vimeo</button>
  </ion-fab-list>
</ion-fab>

</ion-content>
```

List.ts

```
import { Component, Input, OnInit } from '@angular/core';
import { NavController, Refresher } from 'ionic-angular';
import { Locations } from '../../providers/locations';
```

```
import {DetailComponent} from '../detail/detail.component';

import {CartComponent} from '../cart/cart.component';
import {PizzaService} from '../../providers';
import {CartService} from '../../providers';
import {Pizza} from '../../models';

@Component({
  selector: 'page-list',
  templateUrl: 'list.html'
})

export class ListPage implements OnInit {

  pizzas: Pizza[] = [];
  loading: boolean;

  constructor(public navCtrl: NavController, public locations: Locations,
private cartService: CartService,private pizzaService: PizzaService,) {}

  ngOnInit() {
    this.loading = true;
    const subscription = this.pizzaService.getPizzas().subscribe(pizzas => {
      this.pizzas = pizzas;
      this.loading = false;
      subscription.unsubscribe();
    }, () => this.loading = false);
  }

  doRefresh(refresher: Refresher) {
    const subscription = this.pizzaService.getPizzas().subscribe(pizzas => {
      this.pizzas = pizzas;
      refresher.complete()
      subscription.unsubscribe();
    }, () => refresher.complete());
  }

  ionViewDidLoad() {
    console.log('Hello ListPage Page');
  }

  openPizza(id: number) {
    this.navCtrl.push(DetailComponent, {
      id: id
    });
  }

  openCart() {
```

```
    this.navCtrl.push(CartComponent);
  }

  addToCart($event, pizza: Pizza) {
    $event.stopPropagation();

    this.cartService.addCartItem(pizza);
  }
}
```

Src\pages\map

Map.html

```
<ion-header>
  <ion-navbar color="secondary">
    <ion-title>
      EzVacation@SG
    </ion-title>
  </ion-navbar>

</ion-header>

<ion-content>

  <div #pleaseConnect id="please-connect">
    <p>Internet not detected :(</p>
  </div>

  <div #map id="map"></div>

</ion-content>
```

Map.ts

```
import { Component, ElementRef, ViewChild } from '@angular/core';
import { Locations } from '../../providers/locations';
import { GoogleMaps } from '../../providers/google-maps';
import { NavController, Platform } from 'ionic-angular';

@Component({
  selector: 'page-map',
  templateUrl: 'map.html'
})
export class MapPage {

    @ViewChild('map') mapElement: ElementRef;
    @ViewChild('pleaseConnect') pleaseConnect: ElementRef;
```

```
    constructor(public navCtrl: NavController, public maps: GoogleMaps, public
platform: Platform, public locations: Locations) {

    }

    ionViewDidLoad(){

        this.platform.ready().then(() => {

            let mapLoaded = this.maps.init(this.mapElement.nativeElement,
this.pleaseConnect.nativeElement);
            let locationsLoaded = this.locations.load();

            Promise.all([
                mapLoaded,
                locationsLoaded
            ]).then((result) => {

                let locations = result[1];

                for(let location of locations){
                    this.maps.addMarker(location.latitude,
location.longitude);
                }

            });

        });

    }

}
```

Src\providers\

Cart.service.ts

```
import {EventEmitter, Injectable} from '@angular/core';

import {CartItem, Pizza} from '../models';

@Injectable()
export class CartService {
  private cart: CartItem[] = [];
  public statusChanged = new EventEmitter<{type: string; totalCount:
number}>();

  getCart(): CartItem[] {
```

```
    return this.cart;
  };

  addCartItem(location: Pizza): void {
    this.cart.push({
      title: location.title,
      price: location.distance
    });
    this.statusChanged.emit({
      type: 'add',
      totalCount: this.cart.length
    });
  };

  removeCartItem(index): void {
    this.cart.splice(index, 1);
    this.statusChanged.emit({
      type: 'remove',
      totalCount: this.cart && this.cart.length ? this.cart.length : 0
    });
  };

  calcTotalSum(): number {
    let sum = 0;

    if (!this.cart || !this.cart.length) {
      return sum;
    }

    for (let i = 0; i < this.cart.length; i = i + 1) {
      sum = sum + this.cart[i].price;
    }

    return sum;
  }
}
```

Connectivity.ts

```
import { Injectable } from '@angular/core';
import { Network } from 'ionic-native';
import { Platform } from 'ionic-angular';

declare var Connection;

@Injectable()
export class Connectivity {
```

```
  onDevice: boolean;

  constructor(public platform: Platform){
    this.onDevice = this.platform.is('cordova');
  }

  isOnline(): boolean {
    if(this.onDevice && Network.connection){
      return Network.connection !== Connection.NONE;
    } else {
      return navigator.onLine;
    }
  }

  isOffline(): boolean {
    if(this.onDevice && Network.connection){
      return Network.connection === Connection.NONE;
    } else {
      return !navigator.onLine;
    }
  }

}
```

Google-maps.ts

```
import { Injectable } from '@angular/core';
import { Connectivity } from './connectivity';
import { Geolocation } from 'ionic-native';

declare var google;

@Injectable()
export class GoogleMaps {

  mapElement: any;
  pleaseConnect: any;
  map: any;
  mapInitialised: boolean = false;
  mapLoaded: any;
  mapLoadedObserver: any;
  markers: any = [];
  apiKey: string;

  constructor(public connectivityService: Connectivity) {

  }
```

```
  init(mapElement: any, pleaseConnect: any): Promise<any> {

    this.mapElement = mapElement;
    this.pleaseConnect = pleaseConnect;

    return this.loadGoogleMaps();

  }

  loadGoogleMaps(): Promise<any> {

    return new Promise((resolve) => {

      if(typeof google == "undefined" || typeof google.maps == "undefined"){

        console.log("Google maps JavaScript needs to be loaded.");
        this.disableMap();

        if(this.connectivityService.isOnline()){

          window['mapInit'] = () => {

            this.initMap().then(() => {
              resolve(true);
            });

            this.enableMap();
          }

          let script = document.createElement("script");
          script.id = "googleMaps";

          if(this.apiKey){
            script.src =
'http://maps.google.com/maps/api/js?key=AIzaSyCeyYz0tFP9E4tsBW1mr0NXQCr_qLOL8e
s ' + this.apiKey + '&callback=mapInit';
          } else {
            script.src =
'http://maps.google.com/maps/api/js?callback=mapInit';
          }

          document.body.appendChild(script);

        }
      }
      else {

        if(this.connectivityService.isOnline()){
```

```
          this.initMap();
          this.enableMap();
        }
        else {
          this.disableMap();
        }

      }

      this.addConnectivityListeners();

    });

  }

  initMap(): Promise<any> {

    this.mapInitialised = true;

    return new Promise((resolve) => {

      Geolocation.getCurrentPosition().then((position) => {

        // UNCOMMENT FOR NORMAL USE
        let latLng = new google.maps.LatLng(position.coords.latitude,
position.coords.longitude);

        //let latLng = new google.maps.LatLng(40.713744, -74.009056);

        let mapOptions = {
          center: latLng,
          zoom: 10,
          mapTypeId: google.maps.MapTypeId.ROADMAP
        }

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

      });

    });

  }

  disableMap(): void {

    if(this.pleaseConnect){
      this.pleaseConnect.style.display = "block";
```

```
  }

}

enableMap(): void {

  if(this.pleaseConnect){
    this.pleaseConnect.style.display = "none";
  }

}

addConnectivityListeners(): void {

  document.addEventListener('online', () => {

    console.log("online");

    setTimeout(() => {

      if(typeof google == "undefined" || typeof google.maps == "undefined"){
        this.loadGoogleMaps();
      }
      else {
        if(!this.mapInitialised){
          this.initMap();
        }

        this.enableMap();
      }

    }, 2000);

  }, false);

  document.addEventListener('offline', () => {

    console.log("offline");

    this.disableMap();

  }, false);

}

addMarker(lat: number, lng: number): void {

  let latLng = new google.maps.LatLng(lat, lng);
```

```
    let marker = new google.maps.Marker({
      map: this.map,
      animation: google.maps.Animation.DROP,
      position: latLng
    });

    this.markers.push(marker);

  }

}
```

Locations.ts

```
import { Injectable } from '@angular/core';
import { Http } from '@angular/http';
import 'rxjs/add/operator/map';
import { Geolocation } from 'ionic-native';

@Injectable()
export class Locations {

    data: any;

    constructor(public http: Http) {

    }

    load(){

        if(this.data){
            return Promise.resolve(this.data);
        }

        return new Promise(resolve => {

            this.http.get('assets/data/locations.json').map(res =>
res.json()).subscribe(data => {

                this.data = this.applyHaversine(data.locations);

                this.data.sort((locationA, locationB) => {
                    return locationA.distance - locationB.distance;
                });

                resolve(this.data);
            });
```

```
        });

    }

applyHaversine(locations){
let usersLocation;
Geolocation.getCurrentPosition().then((position) => {

   usersLocation = { lat :position.coords.latitude ,  lng :
position.coords.longitude }
   locations.map((location) => {

    let placeLocation = {
      lat: location.latitude,
      lng: location.longitude
    };

    location.distance = this.getDistanceBetweenPoints(
        usersLocation,
        placeLocation,
        'km'
    ).toFixed(2);
  });

});
return locations;
}

    getDistanceBetweenPoints(start, end, units){

        let earthRadius = {
            miles: 3958.8,
            km: 6371
        };

        let R = earthRadius[units || 'km'];
        let lat1 = start.lat;
        let lon1 = start.lng;
        let lat2 = end.lat;
        let lon2 = end.lng;

        let dLat = this.toRad((lat2 - lat1));
        let dLon = this.toRad((lon2 - lon1));
        let a = Math.sin(dLat / 2) * Math.sin(dLat / 2) +
        Math.cos(this.toRad(lat1)) * Math.cos(this.toRad(lat2)) *
        Math.sin(dLon / 2) *
        Math.sin(dLon / 2);
```

```
        let c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
        let d = R * c;

        return d;

    }

    toRad(x){
        return x * Math.PI / 180;
    }

}
```

Pizza.service.ts

```
import {Injectable} from '@angular/core';
import {Http, Response} from '@angular/http';
import {Observable} from 'rxjs/Observable';
import 'rxjs/add/operator/map';
import 'rxjs/add/operator/delay';
import { Locations } from '../providers/locations';
import {Pizza} from '../models';

@Injectable()
export class PizzaService {

  constructor(private http: Http) {}

  getPizzas(): Observable<Pizza[]> {
    return this.http
      .get('assets/pizza.json')
      .delay(2000)
      .map((res: Response) => res.json());
  }

  getPizza(id): Observable<Pizza> {
    return this.http
      .get('assets/pizza.json')
      .map((res: Response) => res.json())
      .map((pizzas: Pizza[]) => {
        for (let pizza of pizzas) {
          if (pizza.id === id) {
            return pizza;
          }
        }
      });
  }
}
```