Department of Informatics, University of Leicester

CO7501 Individual Project

City Tour Planner – Android Application

Author Name: Muhammad Sohaib Furqan

University Email: msf7@student.le.ac.uk

University ID: msf7

Interim Report

Word Count: 4066

Supervisor: Dr. Rayna Dimitrova

Second Marker: Prof. Reiko Heckel

Submission Date: 17 May 2019

DECLARATION

All sentences or passages quoted in this report, or computer code of any form whatsoever used and/or submitted at any stages, which are taken from other people s work have been specifically acknowledged by clear citation of the source, specifying author, work, date and page(s). Any part of my own written work, or software coding, which is substantially based upon other people s work, is duly accompanied by clear citation of the source, specifying author, work, date and page(s). I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this module and the degree examination as a whole.

Name: Muhammad Sohaib Furqan

Signed: *Sohaib*

Date: 17 May, 2019

Abstract

This document builds on the preliminary report. It is hoped that the reader of this document will develop an appreciation of the motivation behind the project and the main challenges that may arise over the course of implementation. A tentative list of requirements (both high-level and detailed), has been laid out. A detailed specification of the design and the core functionality of the project, including a discussion of how project objectives will be measured is included. The foundations of the project rest upon a comprehensive survey of the background material involved (including the study and analysis of applications of a similar nature), and hence made a part of this report. Care has been taken to evaluate and state how the literature survey has influenced project progression. The report also outlines the progress that has been made towards fulfilment of project objectives, discussing the artifacts that have been developed up to this point in time, including an analysis of the challenges that were encountered along the way and the strategies adopted to address them. The report concludes with an initial outline of the dissertation, with the tentative contents documented as a table of contents.

Table of Contents

DECLARATION i
Abstract ii
Section 1: Aims, objectives and challenges 5
Motivation 5
Aims and objectives 5
Challenges 5
Integration of relevant information from various sources: 5
Provide an estimate of the cost of a generated trip: 6
Section 2: Literature Survey 6
TripIt 6
RoadTrippers 6
Sygic Travels 6
Reading List 7
Outcomes of the background research 7
Section 3: Specification and design 7
High-level requirements: 7
Detailed Requirements: 7
1. Allow users to set up a profile: 7
2. Allow users to customize trip experience: 8
3. Retrieve and integrate relevant information from different sources: 8
4. Generate tour suggestions based on user interests and constraints: 8
5. Optimize search results based on user preferences: 8
6. Provide an estimate of the cost incurred by a specific trip: 8
7. Include pictures of venues along with user reviews of the most popular spots: 8
8. Incorporate user feedback and suggestions for improving the user experience of the application: 9
Architecture Overview 9
Technologies: 9
Platform and programming languages 9

Database selection 10
API selection 10
Libraries and packages 11
Other specifications: 11
System design 12
Section 4: Measuring project progress 14
Section 5: Project status 14
1. Explore relevant technologies and platforms 14
2. Literature review 15
3. Gained familiarity with the appropriate APIs and libraries 15
4. Implementation 15
Implemented user sign-up and login with FirebaseUI 15
Implemented Navigation drawer and tabbed layout for main activity 15
Implemented Google Places autocomplete functionality 16
Retrieved nearby place data using Google Places web service 16
Parsed JSON response into recyclerview 16
Section 6: Reflective analysis 16
Section 7: Dissertation outline 16
Explanation 18
References: 19

Section 1: Aims, objectives and challenges

This section of the report provides a detailed description of the rationale behind the project, its aims and objectives in the broader context, and the fundamental challenges involved.

Motivation

Ever since the use of the internet has become widespread and the computing industry has experienced a fundamental paradigm shift towards the use of mobile applications instead of traditional desktop software that were prevalent in the past, we see a raft of mobile applications across platforms that address a wide range of user needs. These mobile applications span almost all areas where desktop programs were customarily deployed and have changed the fundamental way people go about doing their daily tasks. Where a user would have had to bind themselves to a computer screen for carrying out their tasks, the ease of access and portability offered by smart phones have given a new dimension to routine computing.

The flip-side of this increasingly technology driven era though, is that there is hardly any spare time for people to attend to their hobbies. It would be good if this technology dominance can be molded and used to facilitate users in a way that allows them an efficiently managed break from their schedules, without impacting daily timetables in a negative way.

There are many applications that enable users to plan and organize trips, and all of them offer distinct features to facilitate a greater user experience. Redesigning the wheel and coming up with a similar application would amount to a waste of valuable resources and time.

Aims and objectives

Hence, this project is directed towards building upon the offerings of existing applications, albeit with a different objective. The aim is to provide smartphone users with a software-driven solution which allows them to plan a one-day tour of a city. Success in doing so will be of value to users in that they will not be required to set aside a dedicated time period for trips, but instead will be able to organize short recreational trips around their daily workflows. While similar applications work around organizing and managing traditional, long-duration trips, our aim in this project is for the user to be able to explore everything that a city has to offer while adhering to individual time constraints and work commitments.

Challenges

This project poses two fundamental challenges:

Integration of relevant information from various sources:

The application will use information from various available sources (e.g. Google Maps, foursquare, tripadvisor etc.) to generate a customized trip based on user preferences. This will require filtering out available information from and using it in an efficient and effective manner

to provide maximum value for the user. Furthermore, integration of filtered (relevant) information from the various sources to generate the best available trip deals for the user will be a challenge. Time-constraints set by users will also have to be taken into account while developing the application.

Provide an estimate of the cost of a generated trip:

Costs of leisure trips can escalate very quickly because of the large number of factors that can be involved. Users may opt for all kinds of different facilities and experiences, and the choices can cause costs to vary dramatically. Also, different geographical locations have different prices for the same thing. What might be cheaper in one area might be relatively expensive in the others. The application should provide a close approximation of the costs. Given the diversity involved, such a task can pose a significant challenge in the development of the actual software.

Section 2: Literature Survey

Similar android applications including TripIt, RoadTrippers and Sygic Travels were studied as part of the literature review for this project. The following is a brief description of the main features offered by each of these applications. The reading list section indicates resources used to study android programming required for this project. This section concludes with a look at the ways in which project design and implementation has been influenced by this background research.

TripIt

TripIt is a travel planning application which allows for creating an itinerary for each trip. Among other features, this application allows users to forward trip confirmation emails to a dedicated email address (plans@tripit.com). It then creates an itinerary out of the forwarded emails, separately for each trip. Users also have access to a master itinerary where they can access every trip. These plans can then be added to a calendar or shared with friends.

RoadTrippers

This is a web and mobile based application intended to help users in planning road trips. Focusing specifically on areas in the US and Canada, users of this application can choose from over 5 million locations and can use category filters to customize preferences. The application also provides users with turn-by-turn navigation of their desired location.

Sygic Travels

A GPS navigation software that works along the lines of Google Maps but was the first to offer offline maps. Though Google has also started offering this feature now, there is still a huge difference. Sygic maps are far more memory efficient compared to google and also allow for storing larger maps for offline usage.

Reading List

1. Google Codelabs – android fundamentals
2. Android developer guides
3. Android material design guidelines

Outcomes of the background research

A detailed study of the aforementioned applications provided the knowledge and the background necessary for designing a robust, state-of-the-art application. The best practices pertinent in industry were also easily evident. Although details of the coding practices and design patterns could not be retrieved from the high-level study of the applications, sufficient insight was gained to identify technologies being used, or in some cases, their alternatives. Specifically, the research led to getting familiar with the use of material design guidelines, which are extensively used in all modern applications and enable the design of rich UI/UX artifacts. Since the project involves fetching data from various APIs, the study also prompted a comparison of the various options that were available in this regard and helped in making a more informed decision. This is detailed in the "specification and design" section under "technologies".

Section 3: Specification and design

This section of the report discusses the high level and detailed requirements, the system architecture and the technologies that used for the development of the different components of the system. Details and considerations regarding selection of various development artifacts are also laid out. This section concludes with diagrams that show various aspects of system design, including use-case diagrams, activity diagrams and system architecture diagrams.

High-level requirements:

1. Allow users to set up a profile
2. Allow users to customize trip experience
3. Retrieve and integrate relevant information from different sources
4. Generate tour suggestions based on user interests and constraints
5. Optimize search results based on user preferences
6. Provide an estimate of the cost incurred by a specific trip
7. Include pictures of venues along with user reviews of the most popular spots
8. Incorporate user feedback and suggestions for improving the user experience of the application

Detailed Requirements:

1. Allow users to set up a profile:

The application should allow each user to enter profile information including personal information such as name, age, gender, contact number etc. and also information pertaining

to trip preferences and time constraints. This will also include the places and the type of activities that the user is interested in.

2. Allow users to customize trip experience:

Users should be able to customize settings for individual trips based on various search filters .e.g. budget, location, type of trip, activities, food choices, specific facilities etc. If no available trip fulfils the customized criteria, the user should be asked to change the search criteria and try again.

3. Retrieve and integrate relevant information from different sources:

The application will strive to provide the best user experience by using different sources to gather information. The information will be retrieved based on the preferences and search criteria specified by the user. All sources will be queried against the same criteria and the results of the search will be combined in the best possible way to ensure that the user gets the best suited customized trip. The search results should then be displayed in a user-friendly manner.

4. Generate tour suggestions based on user interests and constraints:

The application will allow for storing user preferences and time constraint settings. These should be used to suggest the users regarding available trips. These trip suggestions will be generated automatically and be visible to users if they select the option of “view suggested trips”.

5. Optimize search results based on user preferences:

An appropriate level of detail should be presented to the user initially, with the choice to view more. This is especially important, as too little or too much detail is likely to have a negative impact on the user. Also, only the most relevant information should be retrieved and used by the application.

6. Provide an estimate of the cost incurred by a specific trip:

The application should calculate the estimated cost of the selected trip with the selected settings. This estimate is liable to change based on changes to the user’s search criteria and preferences.

7. Include pictures of venues along with user reviews of the most popular spots:

The application should provide a complete user experience by displaying pictures and descriptions of the famous spots in a city, and should also have the option for tourists to post reviews about their experience at the specific venue. These reviews should then be available for other users to see so that they can make a more well-informed choice.

8. **Incorporate user feedback and suggestions for improving the user experience of the application:**

   Users should be allowed to provide feedback regarding their experience of using the application. This could include comments or requests for add-ons or new features to be made part of the application in subsequent releases.

Architecture Overview

The application is intended for use on the Android OS and will run on almost all android-powered devices. The target audience is any user interested in automated trip-planning. The aim of the application is for users to be able to quickly and efficiently find and customize trip experiences to their liking. Users will be able to set custom search criteria, and retrieve related information on the basis of these settings from multiple sources. Users will be asked to set preferences and these preferences will be taken into account during the search. The information will then be presented in a user-friendly way.

The user is the primary point of interaction for the application. The user may perform operations on the application (such as specifying a city name), to which a response will be returned (such as a list of places of interest within that city). The user could then use these results to create trips. Moreover, the application will also provide appropriate options for trip management (CRUD operations) and keeping history of past trips.

The application will use third-party APIs for retrieving place information. At the time of writing, the Google Places API web service and the Google Places SDK for Android is being used in conjunction to provide the necessary user experience, however, this is likely to be enhanced in future.

The project uses the Firebase database for persistent data storage, the merits of which have been laid out in significant detail in the following discussions. It is worth noting that, due to the nature of the project, changes and revisions are likely to be made throughout the course of project completion.

Technologies:

Although, the downsides of a particular technology depend largely on the nature of the project being developed, making the correct decision is crucial to timely completion of the project. An informed choice of the technologies to use can also greatly reduce the development effort involved.

Platform and programming languages

This application targets the android platform and therefore, a significant part of it will be developed in Java using the Java Platform Standard Edition (Java SE). The database technology for the purposes of this application was initially decided to be SQLite and Object-Relational Mapping (ORM) systems such as ROOM might but as implementation progressed it become

clear that the firebase real-time database will be a much more viable option. Hence the switch to that was made.

Database selection

There are a few obvious advantages that Firebase offered over SQLite in the context of this project. While SQLite happens to be the traditional, and still the most popular way of persistent data storage within android applications, it comes with a fair share of downsides. One striking limitation of SQLite which prompted use of firebase for this project is that synching SQLite data across devices can prove to be difficult. A SQLite database operates on the physical device running the application. As a consequence, data that is stored is internal to each device and cannot be easily transferred from one device to the other. This can give rise to user dissatisfaction as changes made on one device, will most likely not reflect when the same user logs in to the same application from a different device. Firebase uses cloud storage, and hence makes the process automatic with instantaneous synchronization across devices.

API selection

The application makes use of APIs to fetch the necessary data regarding places. This includes (but is not limited to) Google Maps, Trip Advisor and Foursquare. These technologies will be used as deemed appropriate at the time of development so as to enhance the user experience of the application. At the time of writing, the Google Places API is being used to fetch points of interest in a city selected by the user.

There were some choices available with regards to available APIs. One of the apps studied during the literature review, Sygic Travels, used the Sygic Travel API in their app, but for the purposes of this project, Google's offering was preferred because it has a richer places database and is generally more popular.

The following is a comparison of the various APIs as noted during the literature review.

Place API Comparison

Feature	Google	Twitter	Yahoo	Foursquare	Factual	Facebook
Service Name	Places	Places	GeoPlanet	Venues	Global Places	GraphLocation
First Released	Nov. 2010	Jun. 2010	May 2009	Nov. 2009		Aug. 2010
Identifier	Place ID	GEO ID	WOE ID	Venue ID	Factual ID	Numeric ID
Data Size	95 millions data from 70 countries		6 millions from 150 countries		65 millions data from 50 countries	
Data Structure	JSON, XML	JSON	JSON, XML	JSON, JSONP	JSON	JSON
Checkin	No	No	No	Yes		No
Create Place	No	No	No	Yes		No
Geo Coding	Yes	Yes	Yes	No		No
Reverse Geo	Yes				Yes	
Nearby Data	Yes (Mandatory)	Yes	Yes	Yes (Mandatory)		Yes (Mandatory)
Text Search	Yes	No	Yes	Yes		Yes
Popular Place	No	Yes	No	Yes	No	No
Autocomplete	Yes	No	No	Yes	No	Yes
More About API	https://developers.google.com/places	https://dev.twitter.com/docs/platform-objects/places	http://developer.yahoo.com/geo/geoplanet/	https://developer.foursquare.com/	http://www.factual.com/products#location-data	https://developers.facebook.com/docs/reference/android/3.0/interface/GraphLocation/

Places API Comparison

Libraries and packages

The application makes use of various libraries to perform otherwise tedious tasks. These are detailed below:

Android dependencies

- Android support library: implementation 'com.android.support:support-v4:28.0.0'
- Android support compatibility library: implementation 'com.android.support:appcompat-v7:28.0.0'
- Constraint Layout: implementation 'com.android.support.constraint:constraint-layout:1.1.3'
- Recyclerview: implementation 'com.android.support:recyclerview-v7:28.0.0'

Firebase dependencies

- Firebase core: implementation 'com.google.firebase:firebase-core:16.0.8'
- Firebase database: implementation 'com.google.firebase:firebase-database:16.1.0'
- Firebase auth-UI: implementation 'com.firebaseui:firebase-ui-auth:4.3.1'

Third-party libraries and dependencies

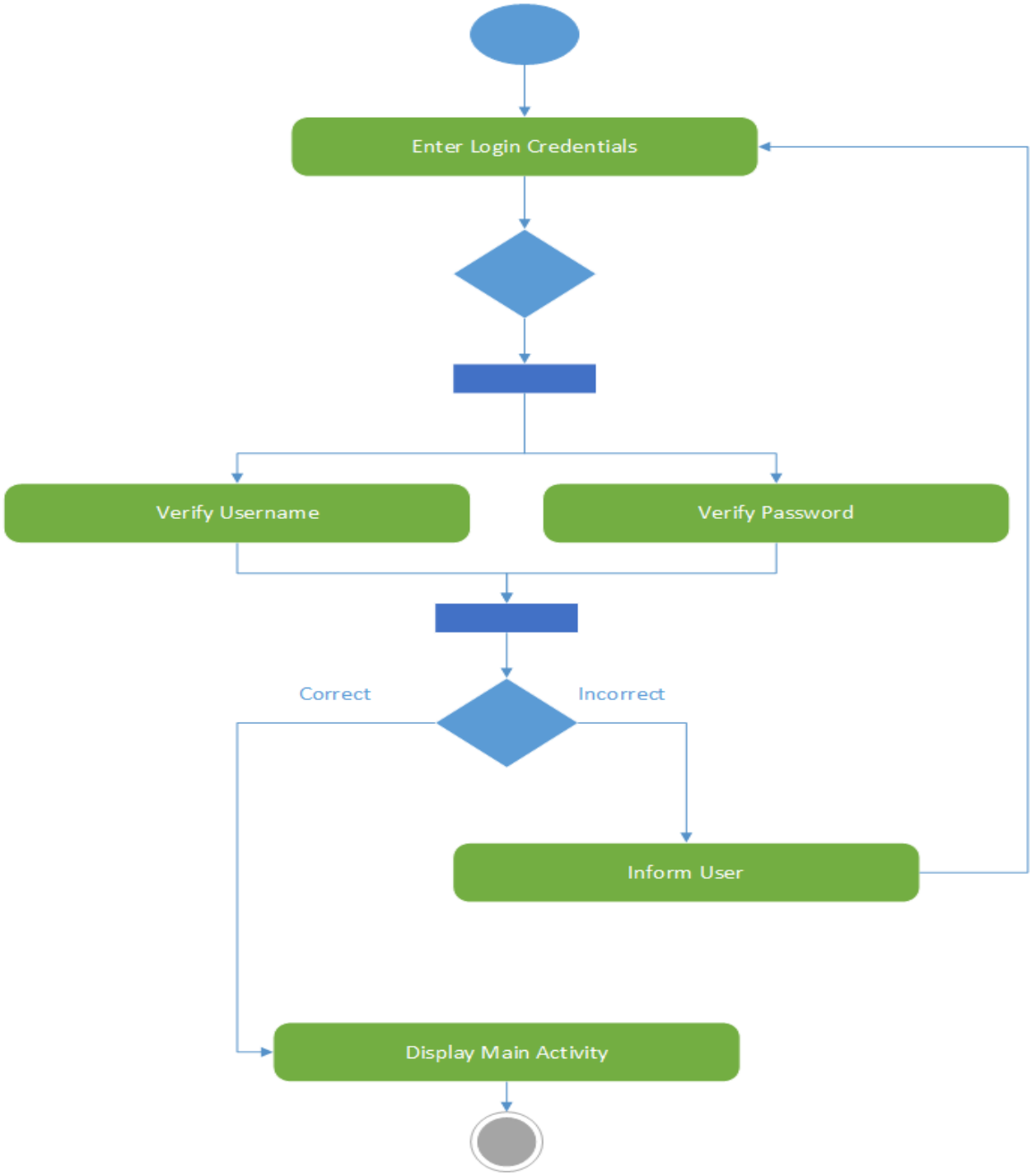
- Ted permissions: This library offers easy permissions handling for Android Marshmallow and above versions. It is available at https://github.com/ParkSangGwon/TedPermission
- Volley: An HTTP library that simplifies networking tasks for android applications and makes it faster. For this project, the library is being used to parse JSON data from APIs so that it can be displayed locally within the application. It is available at https://github.com/google/volley
- Picasso: It is a common task for android applications to load images from online sources. Picasso is an image loader and caching library for android that greatly simplifies this task. It is available at https://github.com/square/picasso

Other specifications:

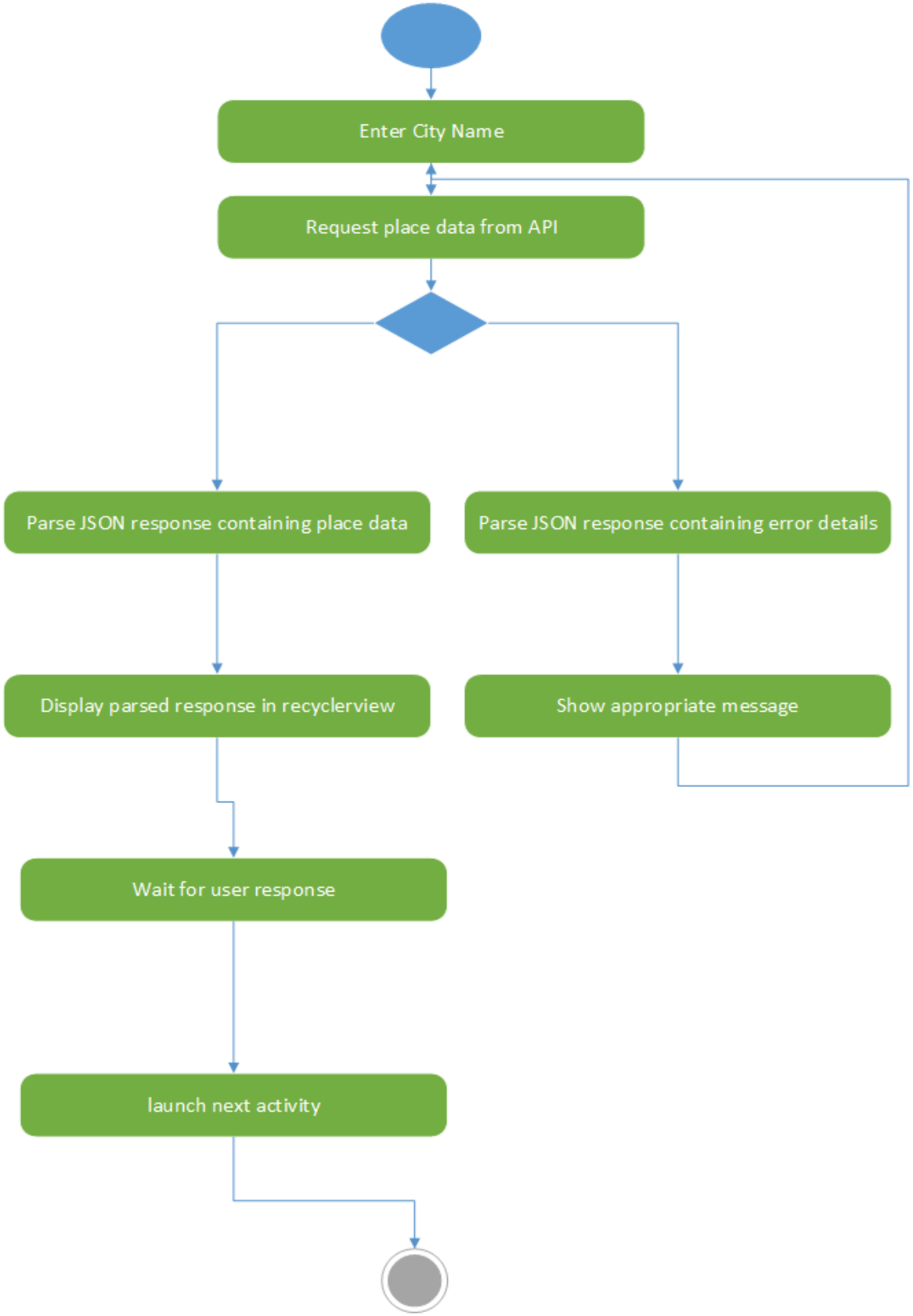
The application will run on any device running Android 4.0.3 (IceCreamSandwich) or higher.

System design

This section depicts system design using activity diagrams. The representations in this document are not exhaustive and changes are likely to be made in the future as the implementation of the project progresses.



Activity Diagram for user login



Activity diagram for "fetch places data"

Section 4: Measuring project progress

To keep track of the project in a sufficiently appropriate manner, in addition to deliverables and periodic supervisor feedback, testing techniques will be applied rigorously. This will ensure that the system thus produced functions as expected and will also expose and anomalies that might need rectification. These are elaborated in the context of the project in the following discussion.

Unit testing

This will be performed throughout the development of the project. This will span testing of individual methods as well as a group of methods. Test cases will be developed to ensure proper behavior under various input combinations. This will be carried out using built-in support of JUnit in android studio. In an android project, this phase typically includes two parts: the backend code is written in Java and needs to be tested using the traditional JUnit way. The front-end GUI is in XML and needs to be tested using tests specific to the android platform. This can also include automated GUI testing, which ensures that UI elements operate as expected and produce the correct outputs.

Functional testing

Functional testing may be used to verify API behavior. A third-party solution such as Postman (or another appropriate solution) might be employed if the need arises.

GUI testing

User Interface will be tested manually but automated testing techniques might also be employed as mentioned in the unit testing section. The application will be test both on an android emulator as well as on a physical android device to ensure consistency. Elements with which the user interacts commonly will be continuously tested to ensure that the user faces no difficulties. Common scenarios which produce errors will be reproduced and monitored to verify that the presentation of errors to the user is as expected.

Section 5: Project status

At the time of writing, the progress made towards project objectives is as stated below:

1. Explore relevant technologies and platforms

Work on the project started with careful consideration of the technologies and platforms necessary. This involved a detailed study of the various options available for developing mobile applications. Specifically, a choice had to be made between using a hybrid mobile application development framework (such as Xamarin and Ionic, among others) and using the traditional native approach. Xamarin was initially adopted as the platform for developing this project, but the downsides became quickly evident. One such drawback, and perhaps the

biggest one, was lack of a community. Tutorials and solutions to common errors were hard to find, and although the developer community is steadily growing, the hard troubleshooting could potentially have been a risk towards timely project completion. Hence it was decided that the traditional approach be taken – native android development with android studio was chosen and a sufficient level of proficiency was gained to ensure a smooth sailing during the implementation phase. Android is a huge platform and can be daunting to first-timers, hence this phase took up a good deal of the time spent on the project.

2. Literature review

A review of literature was performed in which existing applications of a similar nature were critically evaluated. Various important aspects and techniques prevalent in professional mobile application development became clear as a result of this study. To be specific, it enabled an informed decision regarding the difference between this application and the existing ones. While those applications offer a traditional trip-booking experience which spans a certain time duration, the "City Tour Planner" being developed in this project is focused on single-day trips of a single city, thus not necessarily having to be time-bound. This process of the literature survey is detailed in section 2 of the report.

3. Gained familiarity with the appropriate APIs and libraries

As with any other modern application dealing with data from remote sources and carrying out otherwise demanding tasks, the project makes use of various third-party artifacts (APIs and libraries). Consequently, time had to be spent to identify and evaluate the suitability and feasibility of the necessary offerings. Considerations made during this phase included dependency size with respect to the needed features, and where possible, a smaller artifact was preferred due to direct impact on the overall application size. Changes are likely to be made as the development of the project progresses. This is discussed in detail in section 3 of the report.

4. Implementation

Implemented user sign-up and login with FirebaseUI

FirebaseUI is built on top of the core authentication SDK and provides readymade UI flows for use in applications. It provides many features, manual implementation of which would require a great deal of work. Consequently, developer effort is reduced. Some features offered by FirebaseUI include multiple providers (email, google, facebook, twitter etc), account management and custom theming

Implemented Navigation drawer and tabbed layout for main activity

The main screen of the application is displaying after a user logs in to the application and is responsible for showing the scheduled trips along with old and deleted trips. This is achieved through a combination of a tabbed layout and associated fragments. The main activity also contains a navigation drawer which serves a twofold purpose: it provides a quick way for users to easily navigate through the most commonly used areas of the application and also ensures that the user experience is in line with the modern best practices.

Implemented Google Places autocomplete functionality

The Places SDK for Android provides an autocomplete widget that can be used to provide the user with suggestions as they type the name of a place into a textview. This helps improve user experience and is also a common feature in most apps. For the purposes of this project, the autocomplete widget is configured to return only cities as place suggestions, thus ensuring that the user can select only cities.

Retrieved nearby place data using Google Places web service

Once a city has been specified, nearby place data is fetched using the Places API web service. The user may then add the retrieved places to their trips. At the time of writing, the place data being retrieved as part of this process is configured to return only those places which are marked by Google to be “points of interest”. This helps in making sure that irrelevant data is not returned to the end user. The data is returned in the Javascript Object Notation (JSON) format.

Parsed JSON response into recyclerview

The data returned by the places API is parsed into Plain Old Java Object (POJO) format and displayed inside the application using a recyclerview, which arranges items linearly in card-shaped holders, rendered by using a cardview widget.

Section 6: Reflective analysis

At the time of writing, there has been no change to the activities outlined in the initial plan. Strategies to counter challenges include internet tutorials and articles. Where necessary, questions were posted on developer help forums like Stack Overflow. It was tried that the necessary aid be taken from the most relevant sources (.e.g. the official documentation).

Section 7: Dissertation outline

The tentative outline for the dissertation in the form of a table of contents (with sample page numbers) is presented below.

Chapter 1: Introduction **1**

Introduction 2

Objectives 3

Scope 2

Literature Review 3

Problem Statement....2
Summary....3

Chapter 2 Methodology and Design....1
Introduction....2
Approach....3
Architecture....2
Milestones....3
Risk Management....2

Chapter 3 System Specification....1
Introduction....2
Business Requirements....3
High-Level Requirements....2
Functional Requirements....3
Non-functional requirements....2
Assumptions and constraints....3
Actors....3
Use-Cases....3

Chapter 4 Technical Details....1
Introduction....2
Platform and programming language....3
Database Selection....3
API selection....3
Libraries, packages and dependencies....3

Chapter 5 Project Modules....1
Introduction....2
Module descriptions....3

Chapter 6 Testing....1

Introduction 2
Unit Testing 3
Functional Testing 3
GUI Testing 2
Summary 3

Chapter 7 Conclusion and feedback **1**
Code Snippets 2
Future Work 3
Closing Remarks 3
References 3

Explanation

Chapter 1 of the dissertation will look at the motivation behind the project, its objectives and scope. It also looks at the literature survey conducted as part of the background study for the project.

Chapter 2 will be directed at the foundations laid down for starting the project, including a discussion of the general approach throughout the course of the project, the software architecture, the Milestones and the plans for risk management.

Chapter 3 will be devoted to detailing the requirements specifications for the project, covering high level and detailed requirements. Non-Functional Requirements will also be included.

Chapter 4 covers the various details of the project at the technical level

Chapter 5 provides a walkthrough of the various modules (classes, methods etc.) that constitute the project. The purpose served by each module in the broader context of the overall application will be elaborated

Chapter 6 discusses the approaches used for testing the application at length

Chapter 7 closes the dissertation with a glance at some code snippets from the application, ideas for future enhancements to improve the application, and some concluding remarks. The references for writing the dissertation will also be provided in this section.

References:

Android Developer Guides, available at http://developer.android.com/guides (Accessed: 10 February, 2019)

Material Design Guidelines, available at https://material.io/design/guidelines-overview/ (Accessed: 23 February, 2019)

Google Codelabs – Android Fundamentals, available at https://codelabs.developers.google.com/android-training (Accessed: 10 February, 2019)

The Java Tutorials by Oracle, available at https://docs.oracle.com/javase/tutorial/ (Accessed 10 February, 2019)

Places API Comparison, available at https://docs.google.com/document/d/1f_3r8Ebx94ojzeXpOb9-P6ehnSHV5P-t3WlgA0ryzYE/edit (Accessed 9 May, 2019)