**Climate Forecasting**

A

Project Report Submitted to Department of Computer Science

Lamar University



In Partial Fulfilment

Of the Requirement for the Degree

Of Master of Science in Computer Science

By

Nidhi ShaileshKumar Patel

Project Advisor: Dr. Timothy Roden

Associate Professor

August 2017

CLIMATE FORECASTING

A Project report submitted in partial fulfilment of the requirements for Masters of Science in Computer Science, has been approved and accepted by:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr. Timothy Roden,

Associate Professor

Supervising Professor

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_­­\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr. Stefan Andrei,

Chair and Associate Professor

Committee Member

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr. Jing Zhang,

Associate Professor

Committee Member

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr. Stefan Andrei,

Chair, Department of Computer Science,

Lamar University – Beaumont, Texas.

**ACKNOWLEDGEMENT**

It is with my great pleasure to present this report on “**Climate Forecasting**”. I would like to express my deep gratitude to Professor **DR. Timothy Roden**, my project supervisor for his valuable guidance, continuous support, and encouragement. His availability and support were invaluable.

I would like to thank the chair and Associate Professor **Dr. Stefan Andrei**, graduate committee member, for his valuable suggestions and encouragement to finish this project.

I am grateful to Associate Professor **Dr. Jing Zhang**, graduate committee member, for her support and suggestions.

**Table of Contents**

**Acknowledgment** **3**

**Chapter 1 Introduction….................................................................** **5**

**Chapter 2 Design and Implementation** **6**

**2.1 Programming Environment……….** **6**

**2.2 Specifications of project…………...........** **7**

**2.3 Design of project.** **7**

**2.4 UML Diagram** **8**

**2.5 JSON Parsing……………………………………………………………………….12**

**Chapter 3 Screenshots** **13**

**Chapter 4 Conclusion and Future Work……............................................................** **15**

**Chapter 5 Reference….............................................................** **16**

**Chapter 1**

**Introduction**

Climate Forecasting is the application which combines science and technology to predict the state of atmosphere for future time at a given location. Climate Forecasting is very important because it can be used to protect life and property. Climate Forecasting is based on temperature; wind speed and relative humidity are very important attributes in agriculture sector as well as many industries. The mobile app allows people to check out the weather in multiple cities worldwide. The data is dynamic, which means that users can see the weather anytime.

My project is an android application for weather info service. It will help users to see weather information at anytime and anywhere. The mobile app not only show the weather, temperature, and humidity, but it also uses various icons to represent the weather accordingly. It will be easy to read and use. They can search any location in this world and, they can add that city name in list. So, they don’t have to search every time just click on that city name and they will get it weather information. This app is user friendly and easy to understand.

Throughout the remainder of this project I am going to explain the different methodologies used in designing this project, analysis of the project, future work that can be done etc. The organization of the report is as follow: Section 2 Shows different technologies I have used in my project. Section 3 shows how climate forecasting app looks like. And Section 4 concludes the report.

**Chapter 2**

**Design and Implementation**

**2.1 Programming Environment**

To develop Climate Forecasting application, I used different programming surroundings like Java and XML. Java and XML is used for application design and Openweather map is used for server-side database connection. Furthermore, the whole application is developed in Android Studio IDE. Since the project used weather details integration, it required openweather map API key.

**2.1.1 Android Studio**

Android Studio is the official integrated development environment (IDE) for the Android platform. Android Studio provides the fastest tools for building apps on every type of Android device. World-class code editing, debugging, performance tooling, a flexible build system, and an instant build/deploy system all allow you to focus on building unique and high-quality apps. We have used android studio version 2.2.3

**2.1.2 JAVA**

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture.

**2.1.3 JSON**

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. Virtually all modern programming languages support them in one form or another. In JSON, they take on these forms: An object is an unordered set of name/value pairs. An object begins with {(left brace) and ends with} (right brace). Each name is followed by: (colon) and the name/value pairs are separated by, (comma). We also used JSON parsing to retrieve and save data into database. A common use of JSON is to exchange data to/from a web server. When receiving data from a web server, the data is always a string. Parse the data with JSON. Parse (), and the data becomes a JavaScript object.

**2.1.4 Openweather map API**

Openweather is an online service that provides weather data, including current weather data, forecasts, and historical data to the developers of web services and mobile applications. For data sources, it utilizes meteorological broadcast services, raw data from airport weather stations, raw data from radar stations, and raw data from other official weather stations. All data is processed by openweather in a way that it attempts to provide accurate online weather forecast data and weather maps, such as those for clouds or precipitation.

**2.2 Specifications of Project**

**2.2.1 Purpose**

The main target of this application is to provide a weather information anytime and anywhere. Also, it shows humidity and pressure.

**2.2.2 Software Requirement:**

* Android Studio version: 2.2.3
* Minimum SDK version for android: 19
* Target SDK version for android: 25
* API: Openweather Map API

**2.2.3 Software Interfaces**

* Operating System: Android
* Databases: MYSQL
* Middleware: PHP

**2.3 Design of the project:**

Amid my outline stage, I concentrated on the operation of the framework. In this stage, I planned the design and interface of the site alongside the application and store some data and how it ought to stream. In this way began with the general programming framework, and the various levelled structure of the web content.

My android application will contain following layout structure, activities:

**2.3.1 List view of cities:**

This is the launcher activity of my application. As application start user can see the list of cities. The cities which user added. Now user can select any city and see the details of weather. And user can search another city and user can add that city in list too.

**2.3.2 Main activity/Main page:**

In this page user can see all the details for city weather. Like humidity, temperature, Pressure etc. If user want to see atmosphere of another city user can search that city name in search field.

**2.4 UML**

A UML diagram is a limited graphical illustration of a model of a system under design, implementation. UML diagram comprises graphical elements. UML nodes connected with edges that represent elements in the UML model of the designed system. A diagram which shows use cases and actors is use case diagram. A sequence diagram shows sequence of message exchanges between lifelines.

**2.4.1 Use case diagram**

Use case diagrams outline the usage requirements for a system. They are beneficial for presentations to management and/or project investors, but for real development you will find that use cases deliver significantly more value.

**2.4.1.1 Use cases**

A use case depicts a sequence of activities that provide something of measurable value to an actor and is drawn as a horizontal ellipse.

**2.4.1.2 Actors**

An actor is a person, organization, or external system that plays a role in one or more connections with your system. Actors are drawn as stick figures.

**2.4.1.3 Associations**

Associations between actors and use cases are shown in use case diagrams by solid lines. An association happens whenever an actor is involved with a communication described by a use case. Associations are modelled as lines linking use cases and actors to one another, with an optional arrowhead on one end of the line. The arrowhead is frequently used to represent the direction of the initial invocation of the relationship or to specify the primary actor within the use case.

**2.4.1.5 Packages (optional)**.

Packages are UML paradigms that allow you to organize model elements such as use cases.

Packages are represented as file folders and can be used on any of the UML diagrams.

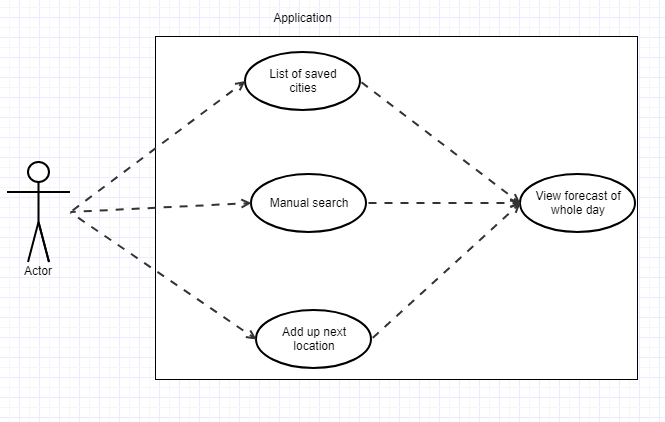


Figure 2.1 Use case Diagram

**2.4.2 Class Diagram**

A Class Diagram is a type of static structure diagram that describes the structure of a system by showing the system's [classes,](https://en.wikipedia.org/wiki/Class_(computer_science)) their attributes, operations (or methods), and the relationships among objects. The class diagram is the main building block of [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) modelling. It is used both for general [conceptual modelling](https://en.wikipedia.org/wiki/Conceptual_model) of the systematics of the application, and for detailed modelling translating the models into [programming code.](https://en.wikipedia.org/wiki/Programming_code)

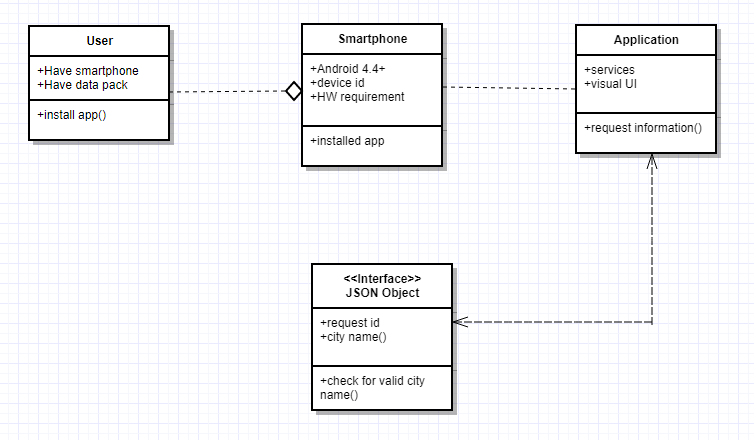


Figure 2.2 Class Diagram

**2.4.3 Sequence Diagram**

UML sequence diagrams model the flow of logic inside your system in a graphic means, it usually used for analysis and design determinations. Sequence diagrams are the most widespread UML artefact for dynamic modelling, which emphases on recognizing the behavior within your system. Other dynamic modelling techniques include activity diagramming, communication diagramming, timing diagramming, and interaction overview. Sequence diagrams are characteristically used to model three things i.e. usage scenarios, the logic of methods and the logic of services.

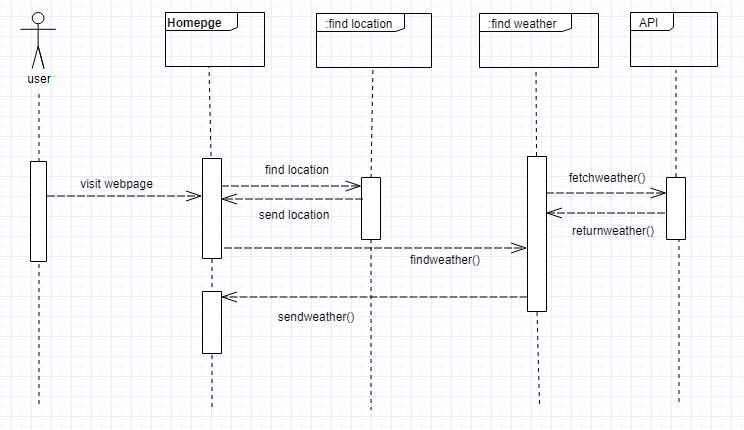


Figure 2.3 when user visit first time (sequence diagram)

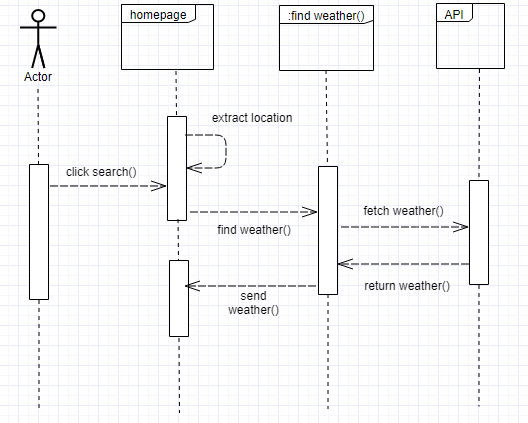


Figure 2.4 Manual search for city (sequence diagram)

**2.5 JSON Object**

JavaScript Object Notation (JSON) is a standard format for representing structured data as JavaScript objects, which is commonly used for representing and transmitting data on web sites (i.e. sending some data from the server to the client, so it can be displayed on a web page). You'll come across it quite often, so in this article we give you all you need to work with JSON using JavaScript, including accessing data items in a JSON object and writing your own JSON.

In general, all the JSON nodes will start with a square bracket or with a curly bracket. The difference between [ and {is, the square bracket ([) represents starting of an JSONArray node whereas curly bracket ({) represents JSONObject. So, while accessing these nodes we need to call appropriate method to access the data.

If your JSON node starts with [, then we should use getJSONArray() method. Same as if the node starts with {, then we should use getJSONObject() method.

The method getJSONObject returns the JSON object. The method getString returns the string value of the specified key. Following is the example of our JSON file.

The put methods adds values to an object.

myString = new JSONObject().put("JSON", "Hello, World!").toString();

produces the string {"JSON": "Hello, World"}.

**Chapter 3**

**Screenshots**

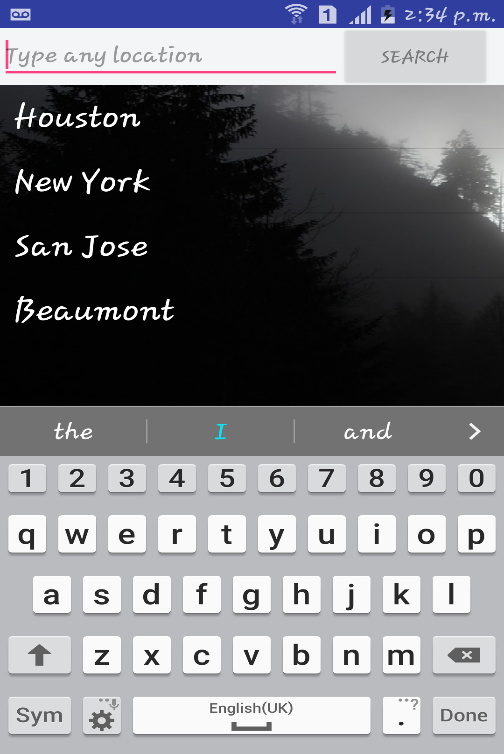
 

Figure 3.1 Homepage (City list) Figure 3.2 City Selected from list

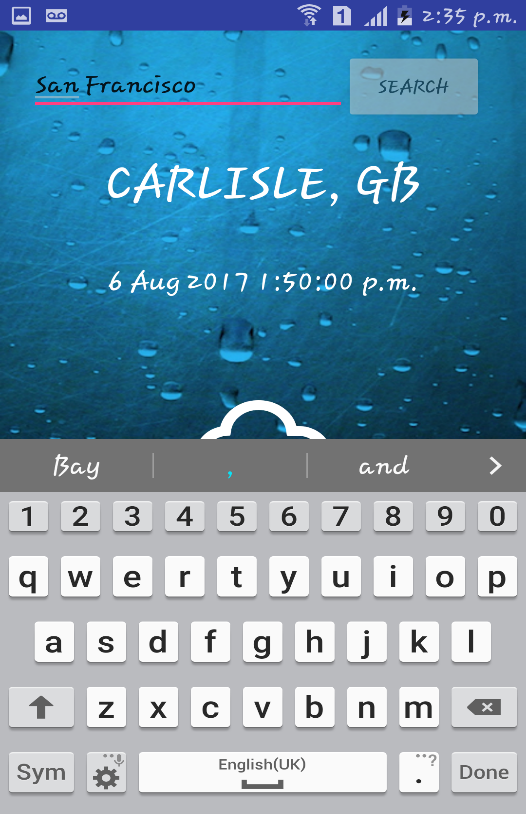
 

Figure 3.4 Figure 3.3

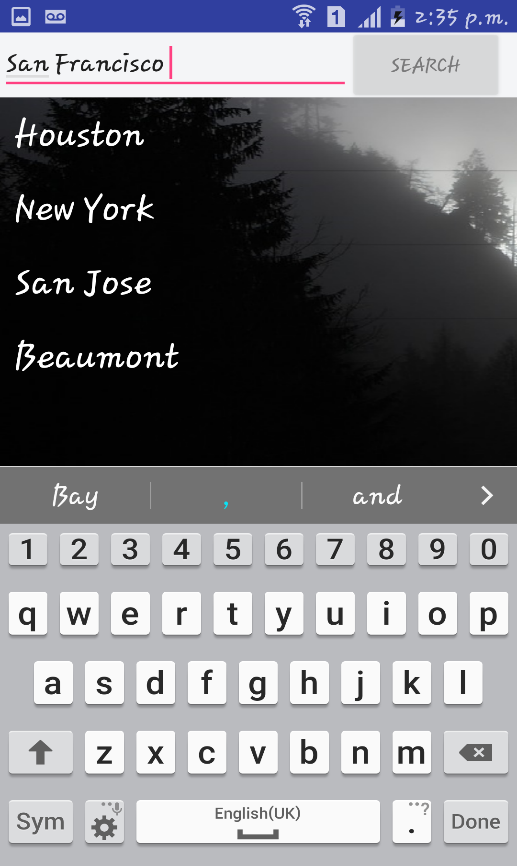


Figure 3.5 Adding up another city in list

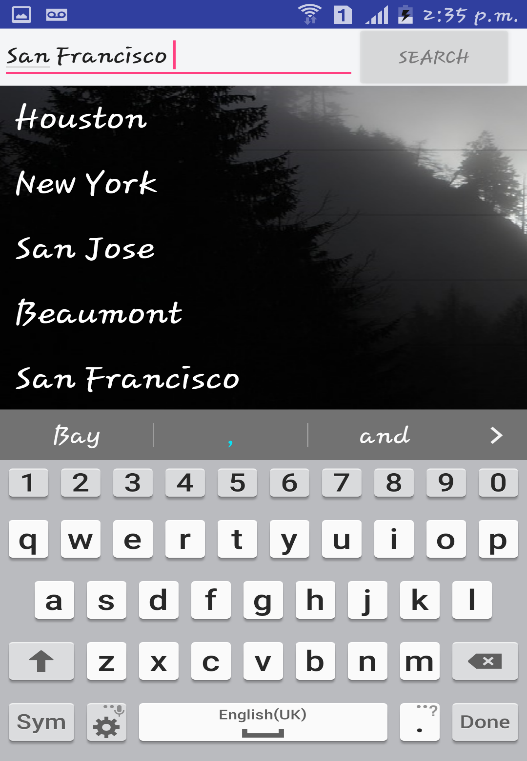


Figure 3.6 City added

**Chapter 4**

**Conclusion and Future work**

In Climate Forecasting app people can easily see atmosphere and make a plan accordingly. And this app is user friendly and easy to operate. People can add city in the city list for time consume and easy to see directly. In this application, I learned so many things and I planning to make advance level of this application.

In future work, I am planning to make notification in this application so that user will get important notification regarding weather. And want to add some more functionality like this app also can work offline too etc.

**Chapter 5**

**References**

1. **Openweather Map API:** <https://home.openweathermap.org/api_keys>
2. **Phillips, Bill, and Brian Hardy. *Android programming: The Big Nerd Ranch guide*.**

**Atlanta, GA: Big Nerd Ranch, 2015.**

1. **For Android Developers:** <https://developer.android.com/index.html>
2. **Weather icons:**

[https://github.com/erikflowers/weather-icons/blob/master/font/weathericons-regular-webfont.ttf](ttps://github.com/erikflowers/weather-icons/blob/master/font/weathericons-regular-webfont.ttf)