

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
“JNANA SANGAMA”, BELAGAVI - 590 018



A MINI PROJECT REPORT
on
“WEATHER FORECASTING APPLICATION”

Submitted by

Karthik J	4SF20IS041
Mohammed Uyam Abbas	4SF20IS055

In partial fulfillment of the requirements for the VI semester

MOBILE APPLICATION DEVELOPMENT

of

BACHELOR OF ENGINEERING

in

INFORMATION SCIENCE & ENGINEERING

Under the Guidance of

Mrs. Shwetha S Shetty

Assistant Professor, Department of ISE

at



SAHYADRI

College of Engineering & Management

An Autonomous Institution

MANGALURU

2022 - 23

SAHYADRI
College of Engineering & Management
An Autonomous Institution
MANGALURU

Department of Information Science & Engineering



CERTIFICATE

This is to certify that the **Mini Project** entitled “**Weather Forecasting Application**” has been carried out by **Karthik J (4SF20IS041)** and **Mohammed Uyam Abbas (4SF20IS055)**, the bonafide students of Sahyadri College of Engineering & Management in partial fulfillment of the requirements for the VI semester **Mobile Application Development (18ISMP68)** of **Bachelor of Engineering in Information Science & Engineering** of Visvesvaraya Technological University, Belagavi during the year 2022 - 23. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of mini project work.

Mrs. Shwetha S Shetty
Assistant Professor
Dept. of ISE, SCEM

Dr. Mustafa Basthikodi
Professor & Head
Dept. of ISE & CSE(DS), SCEM

External Practical Examination:

Examiner's Name

Signature with Date

1.

.....

2.

.....

SAHYADRI
College of Engineering & Management
An Autonomous Institution
MANGALURU

Department of Information Science & Engineering



DECLARATION

We hereby declare that the entire work embodied in this Mini Project Report titled **“Weather Forecasting Application”** has been carried out by us at Sahyadri College of Engineering and Management, Mangaluru under the supervision of **Mrs. Shwetha S Shetty** as the part of the VI semester **Mobile Application Development (18ISMP68)** of **Bachelor of Engineering in Information Science & Engineering**. This report has not been submitted to this or any other University.

Karthik J (4SF20IS041)

Mohammed Uyam Abbas (4SF20IS055)

SCEM, Mangaluru

Abstract

A weather forecasting application typically involves integrating a third-party weather API into the app. The API will provide real-time weather data for a user's current location or any other location they choose. The app also includes features such as daily weather forecasts, weather alerts, and a user interface to display the weather data in an easily understandable way. The user interface of a weather forecasting application is typically simple and intuitive, allowing users to easily navigate through the different sections and view the weather information they need. To ensure accuracy and reliability of the weather data, the app uses multiple APIs and data sources, and implement caching and error handling mechanisms to handle cases of data unavailability or inconsistencies. Overall, a weather forecast feature can enhance the user experience of a mobile app by providing valuable and timely information, and can be a valuable addition for apps related to travel, outdoor activities, or any other context where weather conditions are relevant.

Acknowledgement

It is with great satisfaction and euphoria that we are submitting the Mini Project Report on “**Weather Forecasting Application**”. We have completed it as a part of the VI semester **Mobile Application Development (18ISMP68)** of **Bachelor of Engineering in Information Science & Engineering** of Visvesvaraya Technological University, Belagavi.

We are profoundly indebted to our guide, **Mrs. Shwetha S Shetty**, Assistant Professor, Department of Information Science & Engineering for innumerable acts of timely advice, encouragement and We sincerely express our gratitude.

We express our sincere gratitude to **Dr. Mustafa Basthikodi**, Professor & Head, Department of Information Science & Engineering for his invaluable support and guidance.

We sincerely thank **Dr. Rajesha S**, Principal, Sahyadri College of Engineering & Management who have always been a great source of inspiration.

Finally, yet importantly, We express our heartfelt thanks to our family & friends for their wishes and encouragement throughout the work.

Karthik J

4SF20IS041

VI Sem, B.E., ISE

SCEM, Mangaluru

Mohammed Uyam Abbas

4SF20IS055

VI Sem, B.E., ISE

SCEM, Mangaluru

Table of Contents

Abstract	i
Acknowledgement	ii
Table of Contents	iv
List of Figures	v
1 Introduction	1
1.1 Overview	1
1.2 Purpose	2
1.3 Scope	2
2 Requirements Specification	3
2.1 Hardware Specification	3
2.2 Software Specification	3
3 System Design	4
3.1 Architecture Diagram	4
3.2 Application Modules	5
3.3 End Users	6
3.4 Limitations	7
4 Implementation	8
4.1 Overview	8
4.2 Language Used	8
4.2.1 Java	8
4.2.2 XML	9
4.3 Android Studio	9
4.4 Pseudo-Codes	10
4.4.1 Pseudocode for Searching Location	10

4.4.2	Pseudocode for Temperature Convertor	11
4.4.3	Pseudocode for 7 Day Forecast	12
4.4.4	Pseudocode for Fetching current location	13
5	Results and Disscussion	14
5.1	Registration Page	14
5.2	Home Screen	15
5.3	Weather for Region in a City	16
5.4	7 Day Weather Forecast	17
5.5	Celcius to Fahrenheit Convertor	18
6	Conclusion and Future work	19
	References	20

List of Figures

3.1	Architecture Diagram of Weather Forecasting Application	4
4.1	Pseudocode for Searching Location	10
4.2	Pseudocode for Temperature Convertor	11
4.3	Pseudocode for 7 Day Forecast	12
4.4	Pseudocode for Fetching current location	13
5.1	Registration Page	14
5.2	Home Screen	15
5.3	Weather for Region in a City	16
5.4	7 Day Weather Forecast	17
5.5	Celcius to Fahrenheit Convertor	18

Chapter 1

Introduction

Weather forecasting is a critical aspect of our daily lives, impacting numerous sectors including agriculture, transportation, tourism, and emergency management. Accurate and reliable weather predictions are essential for making informed decisions, planning activities, and ensuring the safety and well-being of individuals and communities. With the advent of advanced technology and the abundance of weather data, weather forecasting applications have become invaluable tools for accessing up-to-date and personalized weather information.

Location-based forecasting is a prominent feature of the application, leveraging geolocation services to provide users with weather forecasts specific to their current or desired location. Users can manually enter their location or allow the application to access their device's GPS for automatic detection. This ensures that users receive tailored weather information for their precise location.

1.1 Overview

The weather forecasting application is a comprehensive tool that provides users with accurate and reliable weather forecasts. It integrates real-time weather data from various sources, utilizes advanced forecasting algorithms, and offers location-specific forecasts. The user-friendly interface allows easy access to current weather conditions, hourly and daily forecasts, and extended forecasts. Customizable notifications and alerts keep users informed about significant weather changes. Additional features such as radar images, air quality index, UV index, and pollen forecasts enhance the user experience. Overall, the weather forecasting application empowers users with timely and personalized weather information for making informed decisions and planning their activities.

1.2 Purpose

The main purpose of the weather forecasting application is to provide users with accurate, reliable, and location-specific weather information. By integrating real-time data, employing advanced forecasting algorithms, and offering a user-friendly interface, the application aims to empower users to make informed decisions based on current and predicted weather conditions. Whether it's planning outdoor activities, optimizing agricultural practices, or ensuring safety during severe weather events, the application serves the purpose of delivering timely and personalized weather forecasts to enhance user decision-making and preparedness.

1.3 Scope

The scope of the weather forecasting application encompasses a range of features and functionalities to cater to diverse user needs. It includes real-time data integration from multiple sources, utilization of advanced forecasting algorithms, location-based forecasting, user-friendly interfaces with intuitive displays, customizable notifications and alerts, and supplementary information such as radar images, air quality index, UV index, and pollen forecasts. The application aims to provide comprehensive and personalized weather information to users, enabling them to make informed decisions, plan activities, and stay prepared for changing weather conditions.

Chapter 2

Requirements Specification

2.1 Hardware Specification

- Processor : Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz
- RAM : 8GB
- Hard Disk : 500GB
- Input Device : Standard keyboard and Mouse
- Output Device : Monitor

2.2 Software Specification

- Programming Language :Java and XML
- IDE :Android Studio
- Database: MySQL

Chapter 3

System Design

3.1 Architecture Diagram

The architecture diagram of the application is as shown in the below figure:

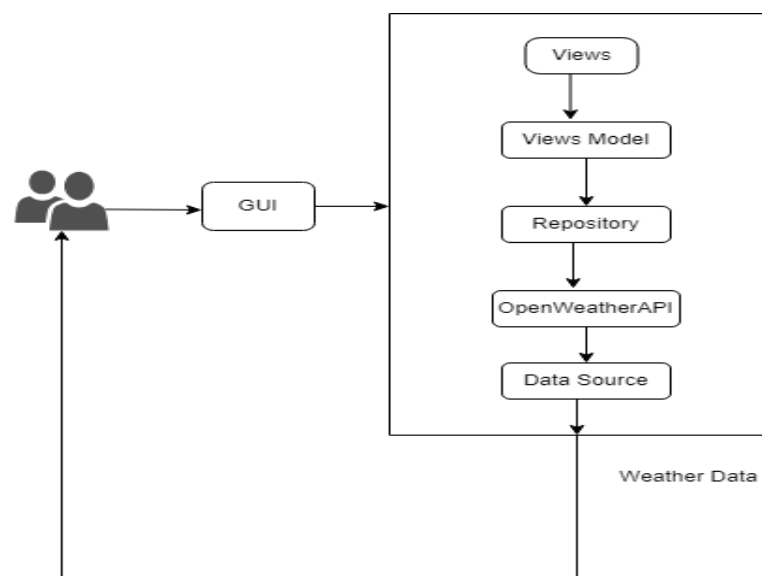


Figure 3.1: Architecture Diagram of Weather Forecasting Application

The user initiates an interaction with the graphical user interface (GUI) by requesting the current location of their device. The GUI captures the user's input and triggers the process of retrieving the device's current location. The obtained location data is then parsed and utilized as a parameter in the request made to the OpenWeatherAPI. The OpenWeatherAPI, acting as a gateway to various data sources, fetches the relevant weather data based on the provided location. Once the weather data is retrieved, it is presented to the user through the GUI, enabling them to conveniently access and view the up-to-date weather information.

3.2 Application Modules

- **Current Location Fetching:** This module is responsible for fetching the user's current location using the device's GPS or location services. It retrieves the latitude and longitude coordinates of the user's location.
- **OpenWeather API Integration:** This module integrates with the OpenWeather API to retrieve real-time weather data for the user's current location. It sends requests to the API and receives responses containing weather information such as temperature, humidity, pressure, and more.
- **Weather Display:** This module is responsible for displaying the current weather conditions of the user's location. It showcases the temperature, description of the weather (e.g., sunny, cloudy), and other relevant weather parameters like humidity or pressure.
- **Calendar Integration:** This module incorporates a calendar feature into the application, allowing users to view the current date. It ensures that the displayed date is synchronized with the device's system clock.
- **7-Day Forecast:** This module provides users with a 7-day forecast of the weather for the user's current location. It displays temperature highs and lows for each day, along with other relevant weather information. Users can get an overview of the expected weather trends over the next week.
- **Celsius to Fahrenheit Converter:** This module enables users to convert temperature values between Celsius and Fahrenheit. It allows users to input a temperature value in one scale and provides the corresponding converted value in the other scale.

3.3 End Users

- **General Public:** Everyday individuals who want to check the weather forecast for their location or any other place they plan to visit. They use weather apps to plan their activities, such as outdoor events, vacations, commuting, and dressing appropriately for the weather.
- **Travelers and Tourists:** People who are planning trips, whether locally or internationally, rely on weather forecasting apps to make informed decisions about packing, choosing destinations, and planning outdoor activities during their travels.
- **Outdoor Enthusiasts:** Hikers, campers, runners, cyclists, and other outdoor enthusiasts rely on weather forecasts to plan their activities safely and enjoyably. They check information like temperature, wind speed, precipitation, and UV index to determine the best time for their outdoor pursuits.
- **Farmers and Agriculturists:** Weather forecasts are crucial for farmers and agriculturists as they rely on accurate predictions to plan planting and harvesting schedules, manage irrigation, prevent crop diseases, and make decisions about crop protection measures.
- **Airlines and Pilots:** Weather conditions have a significant impact on aviation operations. Airlines and pilots use weather forecasting applications to assess weather patterns along their flight paths, make informed decisions about routes and altitudes, and plan for potential turbulence or severe weather conditions.
- **Media and Broadcasters:** Weather forecasts are an essential part of news reporting and broadcasting. Meteorologists and media professionals use weather applications to gather real-time data and present weather information to the public through TV, radio, and online platforms.

3.4 Limitations

Weather forecasting applications have some limitations that users should be aware of. Firstly, forecasting accuracy is not 100 percent guaranteed, as weather conditions can change rapidly and be influenced by numerous factors. Uncertainty and errors can arise due to incomplete data, inaccuracies in weather models, or unforeseen atmospheric phenomena. Secondly, localized variations can pose challenges, as weather patterns can differ significantly within a small geographical area. The generalized forecasts provided by applications may not capture these nuances accurately. Thirdly, data availability and quality can impact forecast accuracy, as some regions may have limited data coverage or issues with data integrity. The forecast horizon has limitations, with longer-term predictions being less precise and more uncertain. Lastly, the weather app requires an internet connection to provide accurate and up-to-date forecast details. The availability of internet access is essential for the app to retrieve real-time weather data from meteorological sources and deliver the most current forecast information to users.

Chapter 4

Implementation

4.1 Overview

Android is an open-source mobile operating system developed by Google. It was initially released in September 2008 and has since become one of the most popular mobile operating systems worldwide. Android is based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and tablets. Android's popularity, flexibility, and extensive developer ecosystem have contributed to its widespread adoption in the mobile industry. It provides a platform for creating innovative and feature-rich applications that cater to a diverse range of user needs. Android provides a suite of developer tools, including Android Studio, the official integrated development environment (IDE) for Android development.

4.2 Language Used

4.2.1 Java

Java is a widely used, high-level programming language that was developed by Sun Microsystems (now owned by Oracle) in the mid-1990s. It is known for its simplicity, portability, and versatility, making it popular among developers for various types of applications, including desktop, web, and mobile. Java's versatility, robustness, and extensive ecosystem have made it a popular choice for a wide range of applications, from enterprise software to Android app development. Its object-oriented nature and platform independence make it suitable for large-scale projects and contribute to its enduring popularity in the software development community.

4.2.2 XML

XML(eXtensible Markup Language) is a markup language designed to store and transport data. It is a standard format for representing structured information, making it easy to share and exchange data between different systems.XML is commonly used in various domains, including web services, data storage, configuration files, and data exchange formats. It provides a flexible and standardized way to represent structured data, enabling interoperability and facilitating communication between different systems and platforms.

4.3 Android Studio

Android Studio is the official integrated development environment (IDE) for developing Android applications. It provides a comprehensive set of tools and features that aid in the development, testing, and debugging of Android apps. Android Studio is available for free and is widely used by developers around the world.is specifically designed to streamline the process of building Android apps and provides a comprehensive set of tools and features to assist developers throughout the development life cycle.Android Studio is continually updated and improved by Google, incorporating the latest Android platform features and development tools. It serves as the primary IDE for Android app development, offering developers a comprehensive and efficient environment for building high-quality Android applications.

4.4 Pseudo-Codes

4.4.1 Pseudocode for Searching Location

Fig: 4.1 Shows a pseudocode for searching of a location in a weather forecasting application. The application typically provides a search bar or a location input field where users can enter the name of a city, town, or even specific coordinates to find weather data for that particular area.

Figure 4.1: Pseudocode for Searching Location

4.4.2 Pseudocode for Temperature Convertor

Figure 4.2 depicts the pseudocode for a temperature converter, a vital and indispensable feature in any weather forecasting application. The ability to convert temperature measurements between Celsius and Fahrenheit is crucial to cater to the diverse preferences of end users.

```
sharedPref= getActivity().getSharedPreferences( name: "Switch",Context.MODE_PRIVATE);
SharedPreferences.Editor editor = sharedPref.edit();

switchcon= sharedPref.getString( s: "switch", s1: "");
if(switchcon.equals("ON"))
{
    simpleSwitch.setChecked(true);
}
else
{
    simpleSwitch.setChecked(false);
}
submit.setOnClickListener(view1 -> {

    if (simpleSwitch.isChecked()) {
        statusSwitch1 = simpleSwitch.getTextOn().toString();
        System.err.println(statusSwitch1);
        editor.putString( s: "switch", statusSwitch1);
        editor.apply();
        Toast.makeText(getActivity(), text: "Value successfully changed", Toast.LENGTH_SHORT).show();
    }
    else {
        statusSwitch1 = simpleSwitch.getTextOff().toString();
        System.err.println(statusSwitch1);
        editor.putString( s: "switch", statusSwitch1);
        editor.apply();
        Toast.makeText(getActivity(), text: "Value successfully changed", Toast.LENGTH_SHORT).show();
    }
});
```

Figure 4.2: Pseudocode for Temperature Convertor

4.4.3 Pseudocode for 7 Day Forecast

Figure 4.3 showcases the pseudocode for 7-day forecast feature, an incredibly valuable tool within the weather forecasting application. This feature provides users with crucial information about the upcoming weather conditions for their current location. With the ability to anticipate weather patterns over the next week, users can effectively plan and prepare for their future activities and engagements.

```
@Override
public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
    view = inflater.inflate(R.layout.sevendaysweather, container, attachToRoot: false);

    progressDialog = new ProgressDialog(getActivity());
    progressDialog.setCancelable(false);
    progressDialog.setCanceledOnTouchOutside(false);
    progressDialog.setMessage("Getting 7 Days Weather Data ");
    progressDialog.show();
    uname= view.findViewById(R.id.sevenusername);

    sharedPreferences= getActivity().getSharedPreferences( name: "Switch", Context.MODE_PRIVATE);
    uname.setText("Hello, "+sharedPref.getString( s: "Username", s1: ""));

    new Handler().post(new Runnable() {
        @Override
        public void run() {
            getWeatherInfo();
        }
    });
    return view;
}

public void getWeatherInfo() {
    //saving data
    SharedPreferences sharedPreferences = getActivity().getSharedPreferences( name: "Switch", Context.MODE_PRIVATE);
    Double latitude = Double.valueOf(sharedPref.getString( s: "Latitude", s1: ""));
    Double longitude = Double.valueOf(sharedPref.getString( s: "Longitude", s1: ""));
}
```

Figure 4.3: Pseudocode for 7 Day Forecast

4.4.4 Pseudocode for Fetching current location

Figure 4.4 showcases the pseudocode for of fetching the current location in a weather forecasting application. This feature plays a crucial role in providing users with real-time weather information tailored to their specific whereabouts.

```
public void getWeatherInfo(double latitude, double longitude) {
    OkHttpClient client = new OkHttpClient();
    String ou_response;
    SharedPreferences.Editor editor = sharedPreferences.edit();
    editor.putString(s: "latitude", String.valueOf(latitude));
    editor.putString(s: "longitude", String.valueOf(longitude));
    editor.commit();
    Request request = new Request.Builder()
        .url("https://api.openweathermap.org/data/2.5/weather?lat="+ latitude+"&lon="+longitude+"&appid=a62d425c25ae4413b28bfb832b82ca7e")
        .get()
        .addHeader( name: "weatheriano", value: "a62d425c25ae4413b28bfb832b82ca7e")
        .build();

    Response response = null;
    try {
        response = client.newCall(request).execute();

        ou_response = response.body().string().trim();

        JSONObject result = new JSONObject(ou_response);
        System.err.println(result);
        String name= result.getString( name: "name");
        JSONArray weather= result.getJSONArray( name: "weather");
        //getting array at 0 index
        JSONObject o = weather.getJSONObject( index: 0);
        weatherinfo.setText(o.getString( name: "main"));
        desweatherinfo.setText(o.getString( name: "description"));
        //setting Icon
        String iconUrl = "http://openweathermap.org/img/w/" + o.getString( name: "icon") + ".png";
        Glide.with(getActivity()).load(iconUrl).placeholder(R.drawable.img).listener(new RequestListener<Drawable>()
            @Override
```

Figure 4.4: Pseudocode for Fetching current location

Chapter 5

Results and Discussion

5.1 Registration Page

Figure 5.1 illustrates the registration process in which users are required to provide their name as the initial step. Registration is a fundamental component of many applications, including those that require personalized user experiences or secure access to specific features. This information is typically used to create a unique user profile or to personalize the user's interactions within the application.

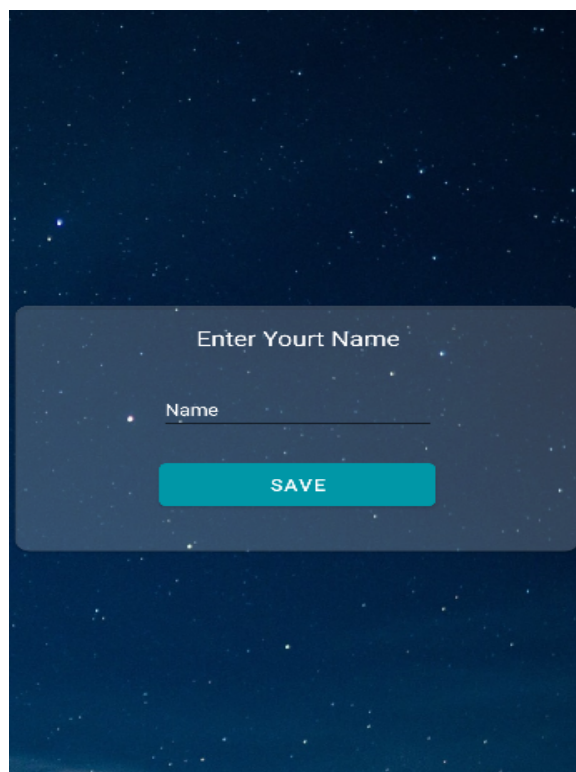
A registration form titled "Enter Yourt Name" (note the typo) is displayed on a dark blue background with a starry pattern. The form is a semi-transparent grey rectangle. It contains a text input field with the placeholder text "Name" and a teal "SAVE" button below it.

Figure 5.1: Registration Page

5.2 Home Screen

Figure 5.2 showcases the home screen of a weather forecasting application, providing users with an overview of the current weather conditions and various related parameters. The home screen serves as the main interface where users can access essential weather information at a glance. These parameters on the home screen of the weather forecasting application provide users with important weather-related information, allowing them to quickly assess the current conditions and plan their activities accordingly.

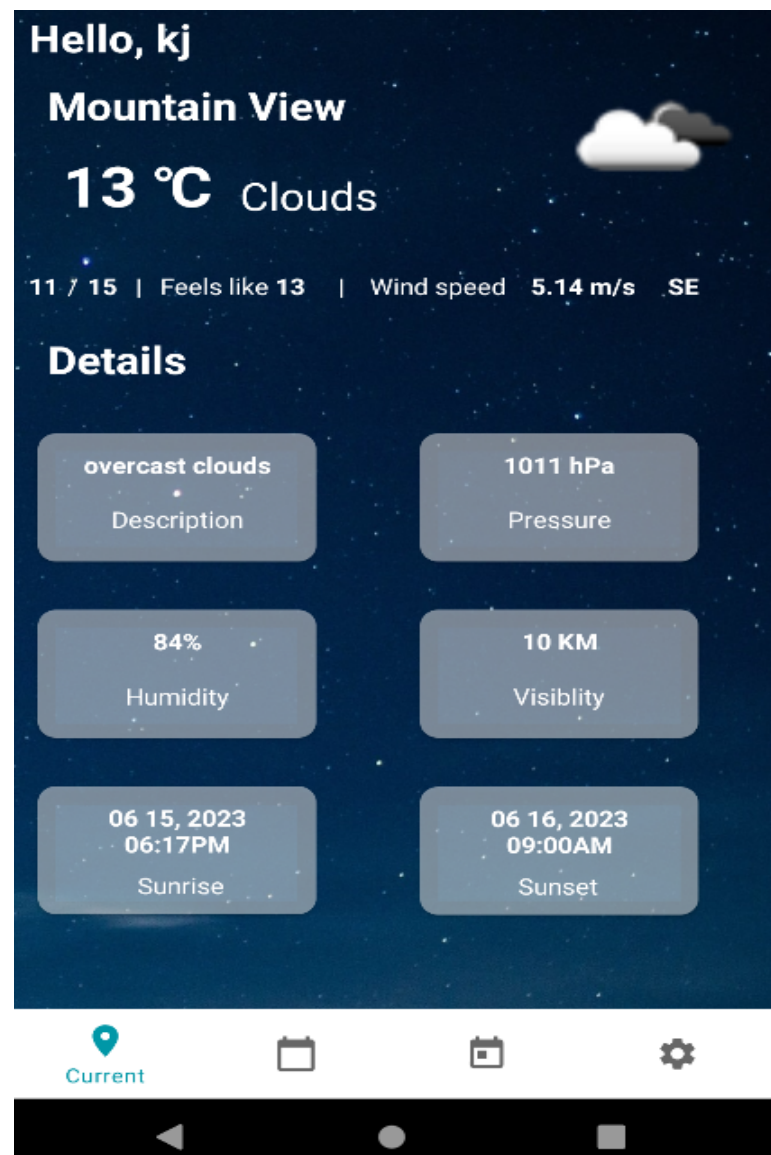


Figure 5.2: Home Screen

5.3 Weather for Region in a City

Figure 5.3 illustrates the weather forecast for major regions covered within a city, along with the current date. This enhanced feature allows users to select multiple regions according to their needs and retrieve specific weather data for those areas. By enabling users to customize their weather information, this functionality provides a more personalized and tailored experience. The inclusion of the current date further enhances the user experience, ensuring that the displayed weather data corresponds to the desired time frame. This real-time information keeps users informed about the latest weather conditions, enabling them to make timely decisions and adjust their plans accordingly.

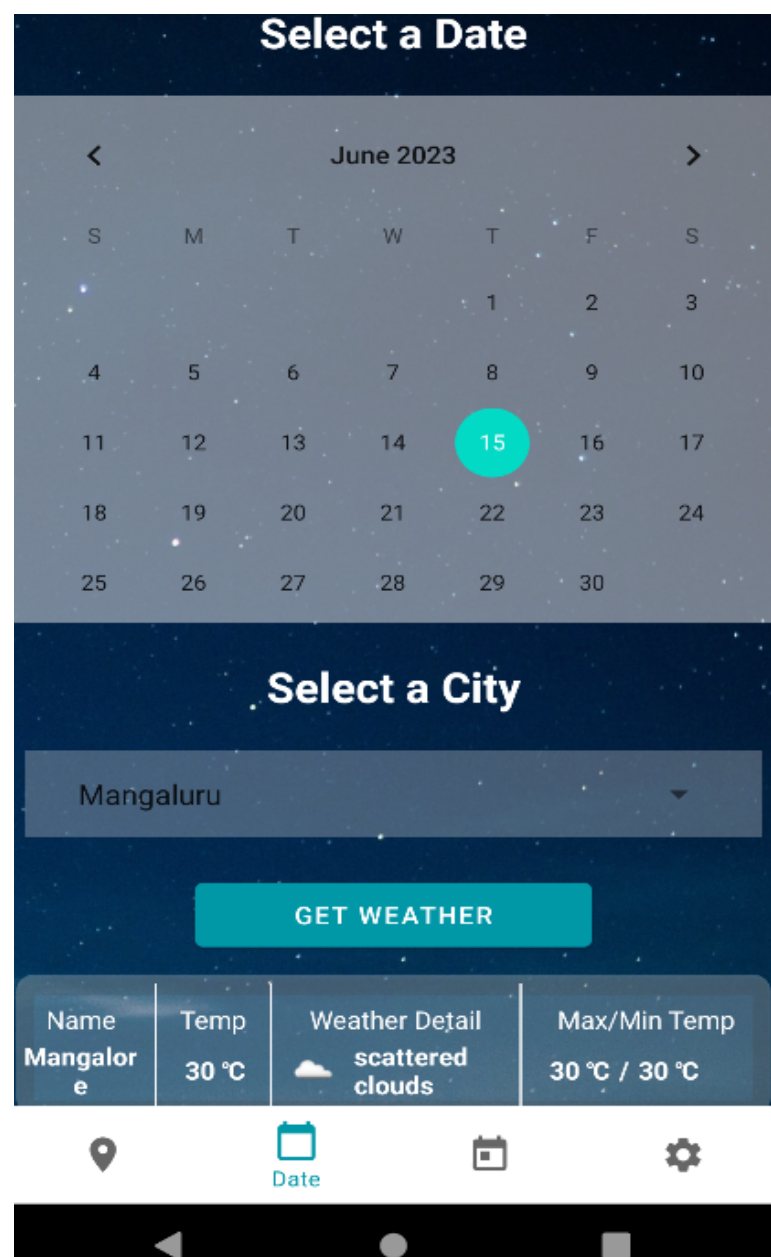


Figure 5.3: Weather for Region in a City

5.4 7 Day Weather Forecast

Figure 5.4 illustrates the 7-day forecast feature in a weather forecasting application. This feature presents users with an outlook of anticipated weather conditions for the upcoming week, allowing them to plan their activities and make informed decisions based on the expected weather trends. The 7-day forecast provides users with essential information such as temperature highs and lows, precipitation chances, and any notable weather events or phenomena. By displaying this data, users can gain insights into the expected weather patterns over the next seven days and adjust their plans accordingly.



Figure 5.4: 7 Day Weather Forecast

5.5 Celcius to Fahrenheit Convertor

Figure 5.5 demonstrates the presence of a Celsius to Fahrenheit converter in a weather forecasting application. This converter allows users to easily convert temperature values between Celsius and Fahrenheit units. Users can input a temperature value in Celsius, and the converter will provide the corresponding Fahrenheit value. In summary, Figure 5.5 represents the inclusion of a Celsius to Fahrenheit converter in a weather forecasting application, offering users the ability to convert temperature values between the two scales effortlessly. This feature enhances user experience and facilitates better understanding and interpretation of temperature data within the application.

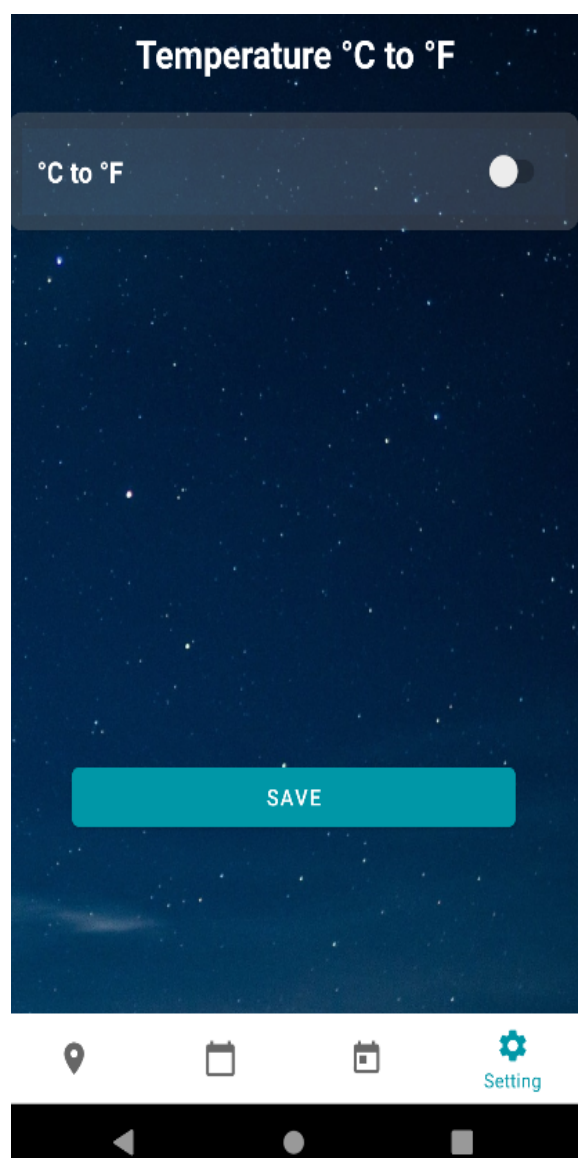


Figure 5.5: Celcius to Fahrenheit Convertor

Chapter 6

Conclusion and Future work

In conclusion, weather forecasting applications have become an integral part of our lives, providing us with real-time weather information and forecasts. These applications use advanced data collection, processing, and visualization techniques to deliver accurate and personalized weather updates to users. With features like location-based services, 7-day forecasts, temperature converters, and more, these applications enhance our ability to plan activities, make informed decisions, and stay prepared for weather changes.

As for future work, weather forecasting applications can continue to improve by incorporating more precise and localized data sources, such as IOT devices and crowd-sourced data. Enhanced machine learning algorithms can be implemented to improve the accuracy and reliability of forecasts. Additionally, integrating additional weather parameters like air quality, UV index, and pollen count could provide users with a more comprehensive understanding of environmental conditions. Furthermore, incorporating predictive analytics and historical data analysis can help users identify long-term weather patterns and plan accordingly. Overall, the future of weather forecasting applications lies in continuous advancements in data collection, analysis, and personalized user experiences to provide even more accurate and valuable weather information.

References

- [1] Google Developer Training, "Android Developer Fundamentals Course-Concept Reference", Google Developer Training Team, 2017. <https://www.gitbook.com/book/googledeveloper-training/android-developer-fundamentals-course-concepts/details>.
- [2] Erik Hellman, "Android Programming-Pushing the limits", 1st Edition, Wiley India Pvt Ltd, 2014. ISBN-13: 978-8126547197.
- [3] Dawn Griffiths and David Griffiths, "Head First Android Development", 1st SPD Publishers, 2015. ISBN-13: 978-9352131341.
- [4] Bill Phillips, Chris Stewart and Kristin Marsicano, "Android Programming: The Big Berd Ranch Guide", 3rd Edition, Big Nerd Ranch Guides, 2017. ISBN-13: 978-0134706054.