INDUSTRY INTERNSHIP
SUMMARY REPORT

GOOGLE AICTE Virtual Internships Program 2023

VIRTUAL INTERNSHIP PROGRAM IN ANDROID DEVELOPER

Submitted in partial fulfillment of the
requirement for the award of the degree of

Master of Computer Applications

GALGOTIAS
UNIVERSITY
(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of
Name of Supervisor:
Designation

Submitted By

KULDEEP GOSWAMI- 22SCSE2030487

SCHOOL OF COMPUTER APPLICATION AND TECHNOLOGY
GALGOTIAS UNIVERSITY, GREATER NOIDA

November -2023

<u>CERTIFICATE</u>

I hereby certify that the work which is being presented in the Internship project report entitled "GOOGLE AICTE Virtual Internship Program 2023 - VIRTUAL INTERNSHIP PROGRAM IN ANDROID DEVELOPER" is in partial fulfilment of the requirements for the award of the degree of Master of Computer Applications Galgotias University, Greater Noida, is an authentic record of my work carried out in the industry.

To the best of my knowledge, the matter embodied in the project report has not been submitted to any other University/Institute for the award of any Degree.

KULDEEP GOSWAMI (22SCSE2030487)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Signature of Internships Coordinator
Dr. N. Partheeban
Professor &IIIC
School of Computing Science &Engineering
Galgotias University Greater Noida

Signature of Program Chair

Signature of Student
Kuldeep Goswami
22SCSE2030487

Date: November 2023

Place: Greater Noida

TABLE OF CONTENTS

Abstract

VERTUAL INTERNSHIP PROGRAM IN ANDROID DEVELOPMENT

This internship is provided by AICTE, GOOGLE ANDROID, and Ed-create. The main objective of this internship is to provide the students with the knowledge and skills in the Android app development domain. This internship provides two online courses at Google Developer. The name of the courses is India Internship Google android developer. These courses mainly focus on the different types of services provided by GOOGLE and teaches basics terminology and implementation of those different services. For example, ANDROID APP.

It has eight modules in the android essentials course and two modules in the Introduction to android course. Each module focuses on different types of technologies in the domain and services provided by google to implement that. Each module has explanatory text slides that explain the concept and then we have a summary of what we have learned from those slides.

Each summary contains a short brief that we have learned at the end of the module.

The Android development course mainly focuses on that how we understand key terms and concepts in Android and Analyse new tools and risks within context of the android.

An internship in Android development at Google would likely involve working on real-world projects related to the Android operating system or other mobile technologies. Here are some aspects you might expect.

CHAPTERS 1
INTRODUCTION

1.1 Objective of the Internship: -

The main objective of this internship is to provide the students with the knowledge and skills in the domain of Android developers and about studios tools.
The main objective of this internship (VIRTUAL INTERNSHIP PROGRAM IN ANDROID DEVELOPER)
The main objectives of a virtual internship program in Android development are likely to align with providing interns with practical experience, exposure to real-world projects, and the opportunity to enhance their skills in Android app development. While specific objectives can vary, here are some common goals for such programs.
Hands-On Experience: The internship aims to provide interns with hands- on experience in developing Android applications. This involves working on actual projects, writing code, and gaining practical skills that are valuable in a professional setting.
Skill Development: The program typically focuses on enhancing the technical skills of interns in Android development. This includes proficiency in programming languages such as Java or Kotlin, understanding the Android framework, and learning about best practices in mobile app development.
Project Contribution: Interns may be assigned specific tasks or features within ongoing projects. The goal is to contribute to the development of real-world applications, providing a sense of accomplishment and a tangible addition to their portfolio.

1.2 Problem Statement and Research Objectives: -

During the internship we got a project with the problem statement stating "Design Secure network for your institution - Select your college or a building of your college and study the network topology of the same and design the same using on packet tracer tool (Please don't configure just design the network). Once done, apply your learnings of Neta cad cyber security course to upgrade / secure the existing network of your college" and we had to create this with the help of Cisco Packet Tracer by applying the knowledge we gained from the course Introduction to packet tracer.

1.2 Description of Domain: -

ANDROID DEVELOPMENT: -

The domain of Android development involves creating applications (apps) that run on the Android operating system, which is primarily used in mobile devices such as smartphones and tablets. Android development is a specialized field within software development that focuses on building applications using the Android platform, typically using programming languages like Java or Kotlin.

Here is a breakdown of key aspects within the domain of Android development:

1. Android Operating System:
    - Android is an open-source operating system developed by Google for mobile devices.
    - Android provides a rich set of features and libraries that developers can leverage to create diverse applications.
2. Programming Languages:
    - Java and Kotlin are the primary programming languages used for Android app development.
    - Kotlin, introduced by JetBrains, is gaining popularity as an official language for Android development due to its concise syntax and modern features.
3. Integrated Development Environment (IDE):
    - Android Studio is the official IDE for Android development. It provides tools for designing, coding, testing, and debugging Android applications.
    - Android Studio supports both Java and Kotlin, offering a comprehensive environment for building and testing Android apps.
4. User Interface (UI) Design:
    - Android apps have a diverse range of user interfaces, from simple to complex.
    - XML is commonly used for designing layouts in Android, and developers can use the Android Studio's visual editor to create and customize UI elements.
5. Application Components:
    - Android applications are composed of various components, including activities, services, broadcast receivers, and content providers.

- Activities represent the UI and user interactions, services handle background processes, broadcast receivers respond to system-wide events, and content providers manage data access.

6. Application Lifecycle:
    - Understanding the Android app lifecycle is crucial for managing resources efficiently.
    - Activities go through different states (e.g., on Create, on Resume, on Pause) based on user interactions and system events.
7. Data Storage:
    - Android apps often need to store and retrieve data. This can be done using various mechanisms, such as SQLite databases, Shared Preferences, or external storage options.
8. Networking:
    - Android apps commonly interact with remote servers to fetch or send data.
    - APIs, HTTP requests, and libraries like Retrofit are used for networking in Android development.
9. Testing and Debugging:
    - Android Studio provides tools for testing and debugging, including emulators for testing on virtual devices and real devices.
    - Unit testing and instrumentation testing are commonly used in Android development.
10. Security and Permissions:
    - Android apps need to adhere to security best practices.
    - Permissions are used to control access to sensitive device features, and developers need to consider security aspects in areas like data storage and network communication.
11. Publishing to Google Play:
    - Once an app is developed, it can be published on the Google Play Store for distribution to users.
    - Developers need to adhere to guidelines, create appealing app listings, and manage updates through the store.

All of these roles can be part of the our work in the exciting system , ever-changing high-demand of android applications.

1.3 A Brief Introduction about an Organization: -

All India Council for Technical Education (AICTE):

AICTE is a national-level apex advisory and regulatory body for technical education in India. It operates under the Department of Higher Education, Ministry of Education, Government of India. AICTE is responsible for planning and coordinating technical education and management education in the country. It plays a crucial role in formulating and implementing policies to ensure the quality and relevance of technical education.

Google:

Google is a multinational technology company that specializes in internet-related services and products. It is renowned for its search engine but has expanded its portfolio to include a wide range of products and services, including Android, one of the most widely used mobile operating systems in the world. Google is known for its innovative approach to technology, and it actively supports educational initiatives and programs to promote skill development in various fields, including software development.

Google Android Development Internship (AICTE):

If there is a collaboration between AICTE and Google for an Android development internship, it would likely be designed to provide students with practical experience in Android app development. Here's what such an internship might involve:

Objective: The internship program would likely aim to enhance the skills of participants in Android development, preparing them for careers in software development.

Curriculum: The curriculum may cover various aspects of Android development, including programming languages (Java or Kotlin), Android Studio IDE, user interface design, application components, data storage, networking, and best practices in app development.

Mentorship: Interns may have the opportunity to work closely with experienced mentors from Google or other partnering organizations, providing guidance and support throughout the internship.

Projects: Interns might work on real-world projects or contribute to ongoing development efforts, gaining hands-on experience in building Android applications.

Learning Resources: The program may provide access to learning resources, workshops, and training sessions to help interns develop a strong foundation in Android development.

For the most accurate and up-to-date information on this specific internship program, I recommend checking the official websites of AICTE and Google or contacting the relevant authorities at AICTE for details on any collaboration and internship opportunities. Additionally, students interested in this program may inquire with their educational institutions for guidance on application processes and eligibility criteria.

Technical Description

A technical description of Android app development involves understanding the various components, tools, and processes involved in creating applications for the Android platform. Here's a detailed technical overview:

1. Development Environment:
    - IDE (Integrated Development Environment): Android Studio is the official IDE for Android development. It provides a comprehensive set of tools for designing, coding, testing, and debugging Android applications.
2. Programming Languages:
    - Java: Traditionally, Java has been the primary language for Android development.
    - Kotlin: Kotlin is a modern programming language that is now officially supported by Google for Android development. It offers concise syntax and additional features compared to Java.
3. User Interface (UI) Design:
    - XML: Android apps use XML for designing layouts. The visual editor in Android Studio allows developers to create and customize UI elements.
4. Application Components:
    - Activities: Represent individual screens in an Android app.
    - Services: Handle background processes independently of UI.
    - Broadcast Receivers: Respond to system-wide broadcast announcements.
    - Content Providers: Manage shared sets of app data.
5. Android Manifest:
    - The AndroidManifest.xml file contains essential information about the app, such as app components, permissions, and hardware requirements.
6. UI Components:
    - Android apps use a variety of UI components, including buttons, text fields, images, lists, and more.
    - These components are defined in XML layout files and can be manipulated programmatically.
7. Activities and Fragments:
    - Activities represent screens with a user interface, while fragments are modular components within activities.

    - Fragment usage allows for more modular and reusable UI components.

8. Intents:
    - Intents are a messaging system that enables communication between components.
    - They are used to start activities, services, and broadcast receivers.
9. Data Storage:
    - SQLite Database: Android includes a built-in SQLite database for local data storage.
    - shared Preferences: Lightweight data storage for key-value pairs.
    - File Storage: Apps can also store data in files on internal or external storage.
10. Networking:
    - Android apps often need to communicate with servers to fetch or send data.
    - APIs and libraries such as Retrofit are commonly used for networking tasks.
11. App Lifecycle:
    - Understanding the Android app lifecycle is crucial for managing resources efficiently.
    - Activities go through various states (e.g., on Create, on Resume, on Pause) based on user interactions and system events.
12. Debugging and Testing:
    - Android Studio provides tools for debugging, including emulators for testing on virtual devices and real devices.
    - Unit testing and instrumentation testing are commonly used for testing Android apps.
13. Version Control:
    - Version control systems like Git are often used to manage source code changes and collaborate with other developers.
14. Publishing to Google Play:
    - Once the app is developed and tested, it can be published on the Google Play Store.
    - The app's listing includes descriptions, screenshots, and other metadata.

This technical overview provides a glimpse into the multifaceted process of Android app development. Developers need to have a strong understanding of these components and tools to create robust and effective Android applications.

CHAPTER 3

Modules Leaning

Unit 1: Your first Android app

Learn programming basics and create your first Android app.

- Write simple Kotlin programs that display text output.
- Download and install Android Studio.
- Build an Android app with a simple user interface that displays text and images.
- Run the app on a device or emulator.

Unit 2: Building app UI

Continue learning the fundamentals of Kotlin and start building more interactive apps.

- Use conditionals, function types, classes, and lambda expressions in Kotlin.
- Understand how composition and recompositing works.
- Add a button to an app UI and respond to user taps.
- Create an app that works with data entered by the user.
- Learn how to use state to display data and reflect the changes automatically when the data gets updated.
- Write unit tests to test isolated functions.

Unit 3: Display lists and use Material Design

Build apps that display a list of data and learn how to make your apps more beautiful with Material Design.

- Use data classes, functions, and collections in Kotlin.
- Create a scrollable list in an app that displays both text and images.
- Add click listeners to interact with list items.
- Add an app bar to the app and modify the app theme.
- Use Material Design to build modern and intuitive user interfaces, using colours, shapes and typography.

Unit 4: Navigation and app architecture

Learn the best practices of app architecture to build more complex apps. Enhance your users' ability to navigate across, into and back out from the various screens within your app for a consistent and predictable user experience.

- Explain activities and their lifecycles.
- Understand Modern Android architecture.
- Use State Flow and UDF pattern to work with state and events.
- Add a View Model to save data and state.
- Set up and use the Navigation component with Compose.
- Understand what responsive UI is.
- Use window class sizes to build for different screen sizes.
- Add a navigation drawer to an app.

Unit 5: Connect to the internet

Use Kotlin coroutines to perform multiple tasks at once, and learn about HTTP and REST to get data from the internet using Retrofit. Then use the Coil library to display images in your app.

- Describe the basics of concurrency and how to use coroutines in an Android app.
- Define and understand the data layer in Modern Android app architecture.
- Implement a repository to centralize data access.
- Use Retrofit to retrieve data from a remote server.
- Load and display images using the Coil library.
- Implement dependency injection to decouple the classes, making it easier to test, maintain, and scale the app.

Unit 6: Data persistence

Learn how to store data locally on the device and keep your apps working through any network disruptions for a smooth and consistent user experience.

- Learn the basics of SQL to insert, update, and delete data from a SQLite database.
- Use the Room library to add a database to an Android app.
- Use Database Inspector to test and debug database issues.
- Use Preference Datastore to store user preferences.

Unit 7: Work Manager

Use Android Jetpack's Work Manager API to schedule necessary background work, such as data backups or fresh content downloads, that keeps running even if the app exits or the device restarts.

- Define long running tasks that need to run in background work.
- Add Work Manager to an Android app.
- Create a Worker object and enqueue work.
- Create constraints on Work Requests.
- Use the Background Task Inspector to inspect and debug Work Manager.

Unit 8: Views and compose

Learn how to use Compose and the older UI toolkit based on Views side-by-side in the same app. In this unit, you will learn interoperability APIs and best practices to add a new feature to an existing app in Views, use an existing library that uses Views, or use a UI component that is not yet available in Compose.

- Understand the View-based UI toolkit and build app UI using XML.
- Add a composable in an app built with Views.
- Add Navigation component to the app and use it to navigate between fragments.
- Use Android View to display views.
- Add existing View-based UI components in a Compose app.

CHAPTER 4

System implementation

Problem Statement: -

This project is to create a Weather Report application with a third-party server and users to enable the users to Forecast Weather... The project should be very easy to use, enabling even a novice person to use it. This application is also developed to make people's life hustle free by introducing some of the extraordinary features like Weather updates which are going on all around the world and getting the weather forecast before going out of the home. This project is to design an application which consists of all the basic features which the users can access with a single click.

DESIGN OF PROJECT

4.1 LIST OF MODULES4.1.1 MODEL

A model stores data that is retrieved according to commands from the controller and displayed in the view. Here the model module contains the four classes clouds, current condition, place, weather, wind. Cloud class contains the data of precipitation, current condition contains the data such as description, maximum, minimum weather temperature, pressure, humidity, icon and condition. Place class contains the data such as latitude, longitude, sunset, sunrise, country, city and last updated data. Wind class contains the speed and degree of the wind. Weather class contains the initialization of the other classes of place, cloud and wind so that we don't have to write the code in the main class.

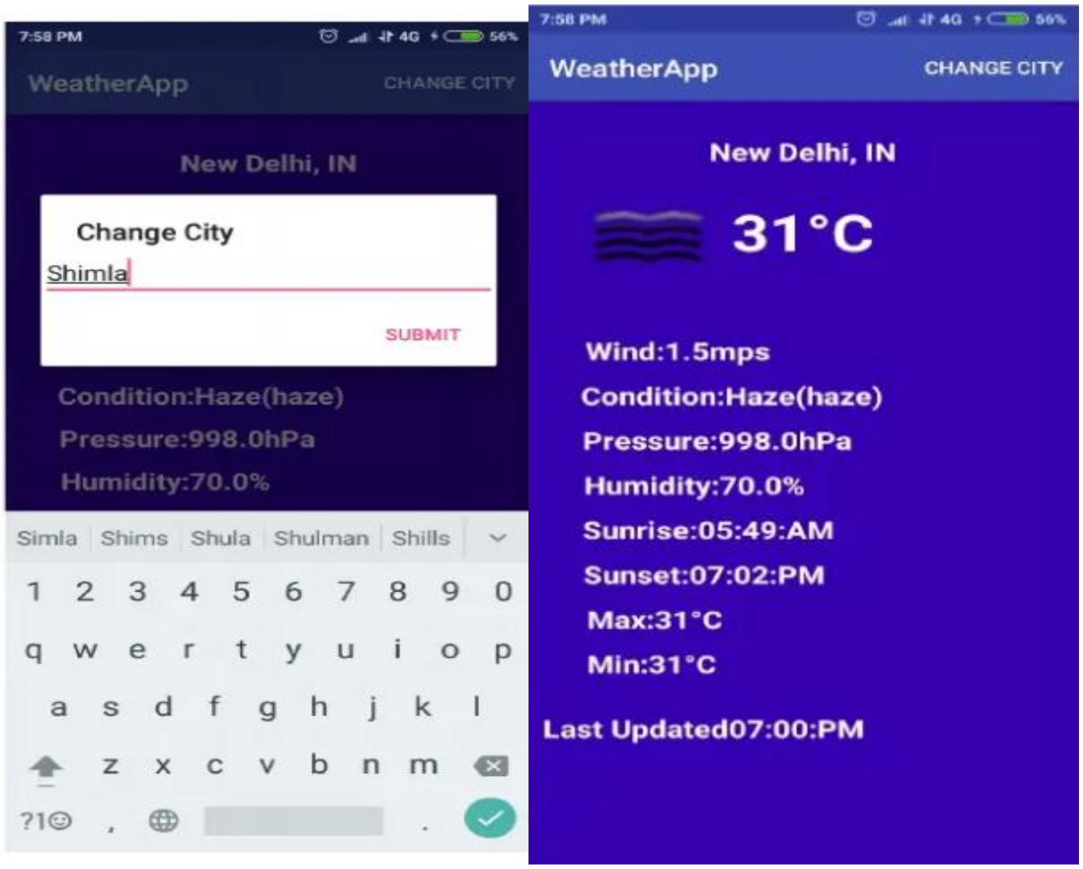
4.1.2 DATA

Data modules contains the classes for downloading the JSON data from the api it downloads the data and reads the data from the API and sets the data in model module. This is the important module as the API is downloaded and read here. City preferences class saves the default city, which is New Dehi, IN. Here the city is saved in the memory of application. Json Weather object class reads the data from API and sets the data to the model class. Weather HTTP client connects to the API and downloads the data if API is not found then it will give the exceptions and will tell the project that it cannot download the data.

4.1.3 UTILS

Utils module contains the link of the API and the link to download the icons for the weather and it contains the code which converts the JSON to the datatype as it is needed for example it converts the JSON object of string type to string and for the same number to int and float value to float.

SCREENSHOT: -



New Delhi, IN

Shimla

Condition:Haze(haze)

Pressure:998.0hPa

Humidity:70.0%

New Delhi, IN

31°C

Wind:1.5mps

Condition:Haze(haze)

Pressure:998.0hPa

Humidity:70.0%

Sunrise:05:49:AM

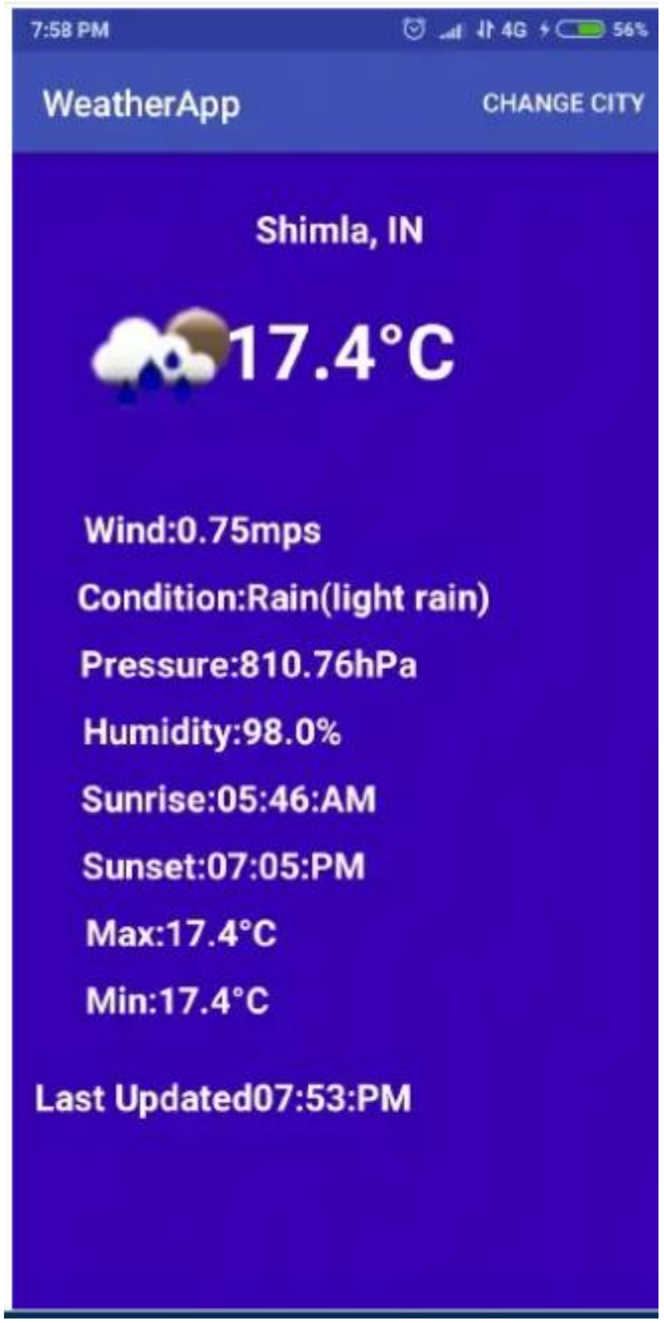
Sunset:07:02:PM

Max:31°C

Min:31°C

Last Updated07:00:PM

Figure 1



Shimla, IN

17.4°C

Wind:0.75mps

Condition:Rain(light rain)

Pressure:810.76hPa

Humidity:98.0%

Sunrise:05:46:AM

Sunset:07:05:PM

Max:17.4°C

Min:17.4°C

Last Updated07:53:PM

Figure 2

CHAPTER 5

FUTURE SCOPE & CONCLUSION

Android is the future of the world day by day its demand is getting increased, and their users are increasing every day in numbers. As android devices are increasing so the demand for the good applications is also increasing day by day. Everyday new apps are launched in google play store. “Android” Weather app” will get the four days forecasting and some other features like it will be updated in every hour it’s user interface will be changed. This app will check the pollution in the city. Android is the future of the world day by day its demand is getting increased, and their users are increasing every day in numbers. As android devices are increasing so the demand for the good applications is also increasing day by day. The project will be useful for checking the weather of your city and for getting started in android for a person.

CHAPTER 6
Appendices

6.1 Learning Experiences: -

1. Beautiful UI:
Android OS basic screen provides a beautiful and intuitive user interface.
2. Connectivity:
It provides connectivity with GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC and WiMAX.
3. Storage:
SQLite, a lightweight relational database, is used for data storage purposes.
4. Media support:
H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1, MP3, MIDI, Ogg Verbids, WAV, JPEG, PNG, GIF, and BMP.
5. Open Source:
Since it is open source, it has a larger community support if anyone gets into any problem there is a big community to support it.
6. multi-tasking:
User can jump from one task to another, and same time various application can run simultaneously.
7. multi-language:
supports single direction and bi-directional text.

6.2 SWOT Analysis: -

Strengths:

1. **Accurate Weather Information:** Providing accurate and reliable weather forecasts will be a significant strength for the application, ensuring user trust and satisfaction.
2. **User-Friendly Interface:** A simple and intuitive user interface will enhance the user experience, making it easy for users to access and understand weather information.
3. **Multi-Platform Accessibility:** Developing the application for multiple platforms (iOS, Android, web) will increase its reach and cater to a broader audience.

4. **Real-Time Updates:** Offering real-time weather updates will set the application apart, making it more valuable for users who require up-to-the-minute information.
5. **Customization Options:** Allowing users to customize their weather preferences, such as location-based alerts and favorite locations, can enhance the personalization of the app.

Weaknesses:

1. **Dependence on External Data Sources:** The application's accuracy is highly dependent on the data it receives from external weather services. Any disruptions in these data sources could affect the app's reliability.
2. **Technical Issues:** Potential technical glitches, such as server downtimes or bugs in the application, could lead to user dissatisfaction and impact the app's reputation.
3. **Limited Geographic Coverage:** If the application focuses on a specific region or lacks global coverage, it may limit its appeal to users outside that area.
4. **Competition:** The weather app market is competitive, and standing out from established competitors may be a challenge.
5. **Monetization Challenges:** Determining a sustainable and user-friendly monetization strategy without compromising the app's core user experience can be challenging.

Opportunities:

1. **Integration with Other Apps:** Collaborating with other applications, such as travel or event planning apps, can provide opportunities for cross-promotion and enhanced user engagement.
2. **Innovative Features:** Adding unique features, such as weather-related games, augmented reality experiences, or social sharing options, can attract new users and set the app apart.
3. **Global Expansion:** Expanding the app's coverage to include more regions globally can tap into new markets and increase the user base.
4. **IoT Integration:** Exploring integration with IoT devices, such as smart home systems or wearables, can open up new possibilities for user interaction and convenience.
5. **Data Analytics:** Leveraging user data for analytics can provide insights into user behaviour, helping to improve the app and tailor features to user preferences.

Threats:

1. **Data Security Concerns:** As the app deals with location data, ensuring robust data security and privacy measures is crucial to avoid potential breaches and legal issues.

2. **Changing Technology Trends:** Rapid advancements in technology may make certain features or aspects of the app obsolete. Staying abreast of technological changes is essential.
3. **Weather Prediction Challenges:** The inherent unpredictability of weather patterns poses a threat to the app's accuracy, and any inaccuracies may lead to user dissatisfaction.
4. **Regulatory Changes:** Changes in regulations related to data privacy or the use of location-based services could impact the app's functionality and compliance.
5. **Negative User Reviews:** Negative reviews or feedback, especially in the early stages of the app, could harm its reputation and deter potential users.

CHAPTER 7

REFERENCES

https://developer.android.com/courses/android-basics-compose/course