Summer Internship at Internship at NIT Mizoram

in the Department of "Computer Science & Engineering"

June – July 2024

Lakhya Borah

CSM23061

Tezpur University

National Institute of Technology Mizoram

July 2024

Aizawl, India
22 July 2024

To
The Training and Placement Office
National Institute of Technology Mizoram

Dear Sir,

I submit this report entitled "*Optical English Character Recognition Android Application*" for the summer internship project undertaken in the Department of Computer Science & Engineering from 24 June 2024 to 24 July 2024. I hereby declare that this report has been prepared based on the internship project and has not been copied from any other offline or online resources.

Sincerely,

[illegible]

Signature of the student

Lakhya Borah

Checked by

Signature of the project supervisor

Lenin Laitonjam

Department of Computer Science & Engineering

National Institute of Technology Mizoram

Abstract

This report presents the development and implementation of an Android application for Optical English Character Recognition, completed during my internship at the Department of Computer Science & Engineering, NIT Mizoram. The application features include user authentication (login and signup), capturing or uploading images for English text extraction, copying extracted text, and saving scanned text records as notes. Developed using Java, the application leverages Firebase for storing user credentials and records. For optical English character recognition, Google's developer API is utilized, ensuring accurate and efficient text extraction. This project aims to provide a robust solution for converting images of text into editable digital text, facilitating better accessibility and usability of printed materials. The application's potential uses range from digitizing documents to aiding visually impaired users by converting printed text to digital format. The report details the project's background, methodology, results, and future work prospects, reflecting on the skills and knowledge gained during the internship.

Nomenclature

Abbreviations

AR	Aspect Ratio
BR	Blockage ratio
HAWT	Horizontal Axis Wind Turbine

Greek Letters

ρ	Air density	kg/m^3
ω	Rotational speed	rpm
σ	Solidity	-

Symbols

A	Frontal area	m^2
c	Chord length	m
C_m	Coefficient of moment	-

List of Figures

List of Tables

Table of Contents

INTRODUCTION Chapter 1

1.1 Background about Mobile Technologies

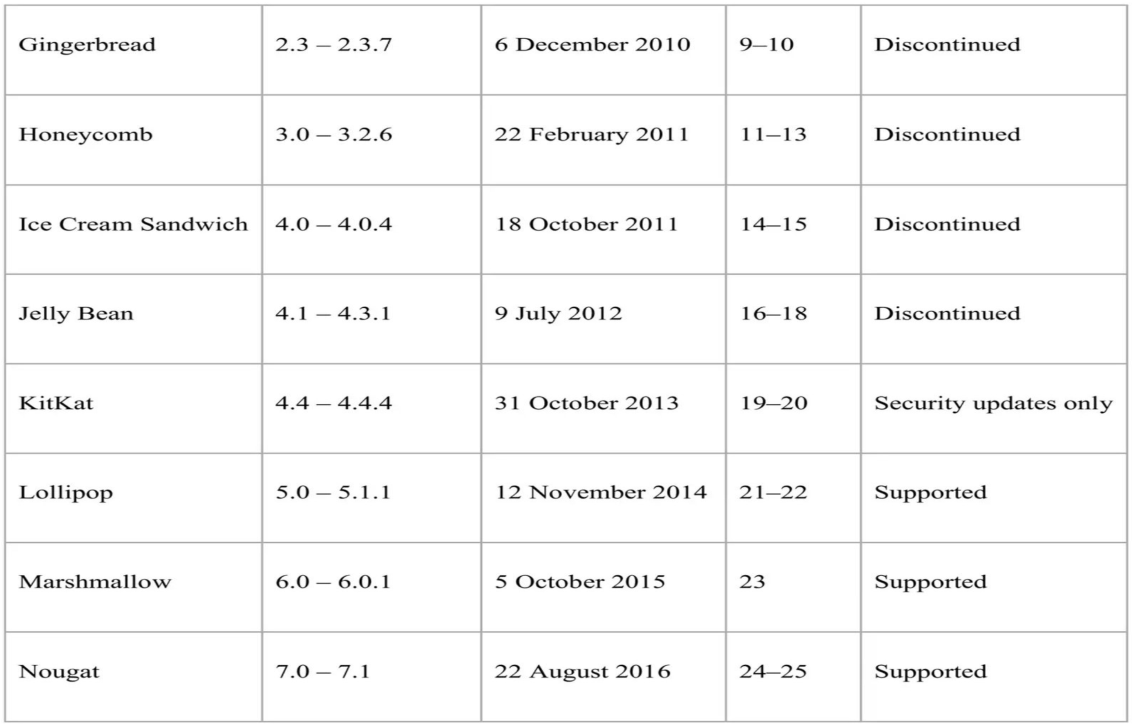
Mobile technology is the technology used for cellular communication. Since the start of this millennium, a standard mobile device has gone from being no more than a simple two-way pager to being a mobile phone, GPS navigation device, an embedded web browser and instant messaging client, and a handheld game console. Many types operating systems are available for smart phones, including Android, BlackBerry OS, iOS, Symbian, Windows Mobite and Bada of mobile.

1.2 Android

Android is an operating system based on Linux with a Java programming interface. Android is a mobile operating system (0S) developed by Google. Android is the first completely opensource mobile OS. Building on the contributions of the open-source Linux community and more than 300 hardware, software, and carrier partners, Android has rapidly become the fastest-growing mobile OSs.

Android versions

Code name	Version number	Initial release date	API level	Support status
N/A	1.0	23 September 2008	1	Discontinued
	1.1	9 February 2009	2	Discontinued
Cupcake	1.5	27 April 2009	3	Discontinued
Donut	1.6	15 September 2009	4	Discontinued
Eclair	2.0 – 2.1	26 October 2009	5–7	Discontinued
Froyo	2.2 – 2.2.3	20 May 2010	8	Discontinued



Gingerbread	2.3 – 2.3.7	6 December 2010	9–10	Discontinued
Honeycomb	3.0 – 3.2.6	22 February 2011	11–13	Discontinued
Ice Cream Sandwich	4.0 – 4.0.4	18 October 2011	14–15	Discontinued
Jelly Bean	4.1 – 4.3.1	9 July 2012	16–18	Discontinued
KitKat	4.4 – 4.4.4	31 October 2013	19–20	Security updates only
Lollipop	5.0 – 5.1.1	12 November 2014	21–22	Supported
Marshmallow	6.0 – 6.0.1	5 October 2015	23	Supported
Nougat	7.0 – 7.1	22 August 2016	24–25	Supported

Table 1: Android Versions

1.3 Global present scenario

The global landscape of mobile technology is constantly evolving, marked by rapid advancements and widespread adoption across various sectors. Currently, mobile technologies are integral to daily life, influencing communication, business, healthcare, education, and entertainment. Smartphones have become ubiquitous, with over 3.8 billion users worldwide as of 2021, reflecting a penetration rate exceeding 48%. This widespread usage is driven by the proliferation of affordable devices, improved network infrastructures, and increasing availability of mobile internet.

Technological advancements such as 5G networks are revolutionizing the mobile experience by providing faster data speeds, reduced latency, and enhanced connectivity. These improvements are enabling new applications, including the Internet of Things (IoT), augmented reality (AR), and virtual reality (VR), which rely on robust and reliable mobile networks.

Moreover, mobile payment systems and digital wallets are gaining traction globally, transforming the financial sector and facilitating cashless transactions. Regions like Asia and

Africa are witnessing significant growth in mobile banking, providing financial inclusion to previously unbanked populations.

In the healthcare sector, mobile technologies are enhancing patient care and accessibility through telemedicine, remote monitoring, and mobile health applications. Educational institutions are leveraging mobile platforms for remote learning, making education more accessible to students worldwide.

1.4 Mobile Application

A mobile Application is a software application designed to run on smart phones, tablets computers and other mobile devices. Mobile apps were originally offered for generally productivity and information retrieval, including email, calendar, contacts, and stock market and weather information.

1.5 Android Structure

➢ **Linux kernel**

Linux Kernel is heart of the android architecture. It manages all the available drivers such as display drivers, camera drivers, Bluetooth drivers, audio drivers, memory drivers, etc. which are required during the runtime. The Linux Kernel will provide an abstraction layer between the device hardware and the other components of android architecture. It is responsible for management of memory, power, devices etc. The features of Linux kernel are:

- **Security:** The Linux kernel handles the security between the application and the system.
- **Memory Management:** It efficiently handles the memory management thereby providing the freedom to develop our apps.
- **Process Management:** It manages the process well, allocates resources to processes whenever they need them.
- **Network Stack:** It effectively handles the network communication.

- **Driver Model:** It ensures that the application works properly on the device and hardware manufacturers responsible for building their drivers into the Linux build.

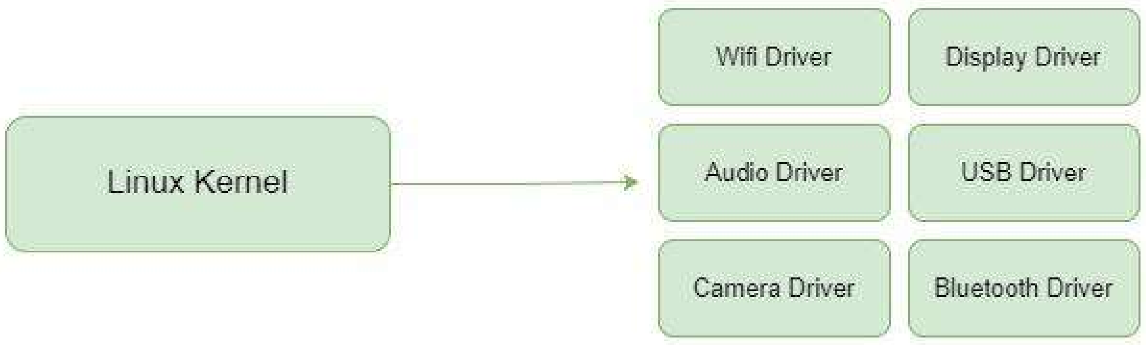


Figure 1: Linux Kernel

➢ **Platform libraries**

The Platform Libraries includes various C/C++ core libraries and Java based libraries such as Media, Graphics, Surface Manager, OpenGL etc. to provide a support for android development.

- **Media** library provides support to play and record an audio and video formats.
- **Surface manager** responsible for managing access to the display subsystem.
- **SGL** and **OpenGL** both cross-language, cross-platform application program interface (API) are used for 2D and 3D computer graphics.
- **SQLite** provides database support and **FreeType** provides font support.
- **Web-Kit** This open source web browser engine provides all the functionality to display web content and to simplify page loading.
- **SSL (Secure Sockets Layer)** is security technology to establish an encrypted link between a web server and a web browser.

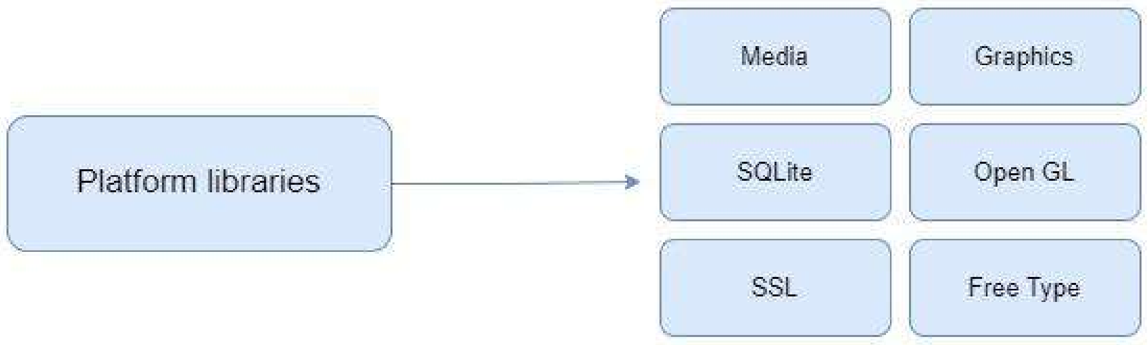


Figure 2: Platform Libraries

➢ **Application Runtime**

Android Runtime environment is one of the most important part of Android. It contains components like core libraries and the Dalvik virtual machine (DVM). Mainly, it provides the base for the application framework and powers our application with the help of the core libraries. Like Java Virtual Machine (JVM), **Dalvik Virtual Machine (DVM)** is a register-based virtual machine and specially designed and optimized for android to ensure that a device can run multiple instances efficiently. It depends on the layer Linux kernel for threading and low-level memory management. The core libraries enable us to implement android applications using the standard JAVA or Kotlin programming languages.

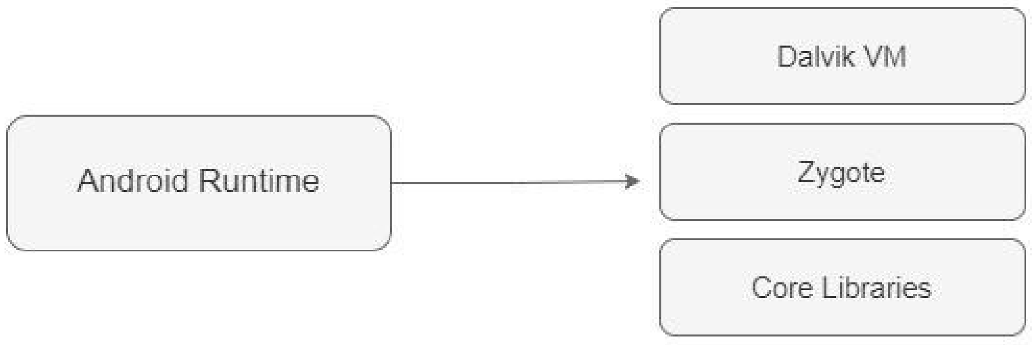


Figure 3: Android Runtime

➢ Application Framework

Application Framework provides several important classes which are used to create an Android application. It provides a generic abstraction for hardware access and also helps in managing the user interface with application resources. Generally, it provides the services with the help of which we can create a particular class and make that class helpful for the Applications creation. It includes different types of services activity manager, notification manager, view system, package manager etc. which are helpful for the development of our application according to the prerequisite.

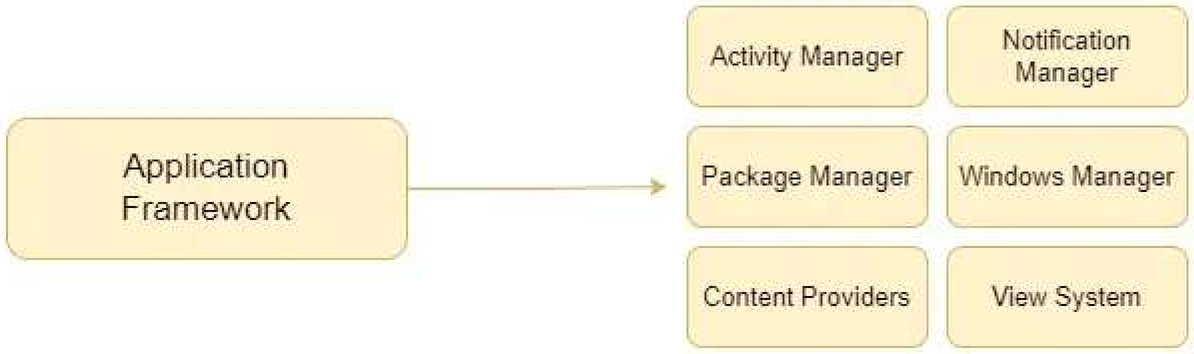


Figure 4: Application Framework

➢ Applications

Applications is the top layer of android architecture. The pre-installed applications like home, contacts, camera, gallery etc and third party applications downloaded from the play store like chat applications, games etc. will be installed on this layer only. It runs within the Android run time with the help of the classes and services provided by the application framework.

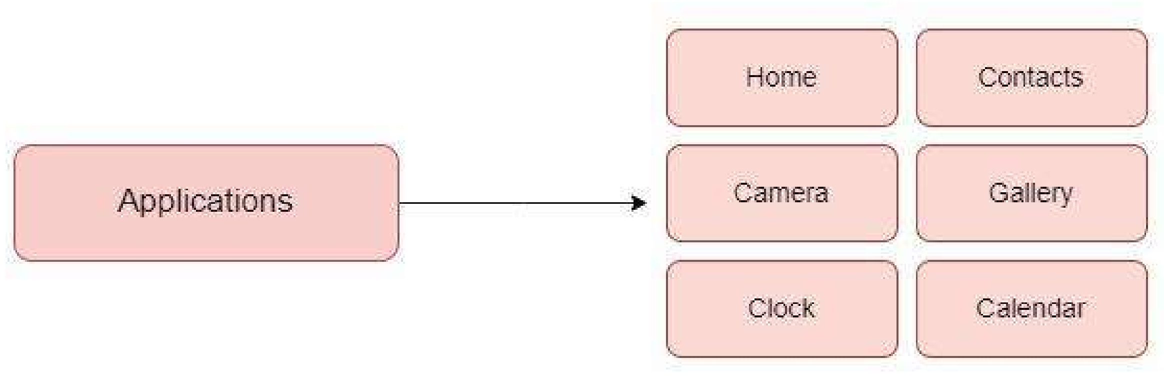


Figure 5: Applications

1.6 Introduction to Optical Text Recognition (OCR)

Optical English Text Recognition (OCR) is a transformative technology that converts printed or handwritten text into machine-readable data. In the context of mobile applications, OCR enables the seamless integration of text recognition capabilities, allowing users to extract and interact with text from images or scanned documents directly from their smartphones.

With the advancement of OCR technology, it is now possible to accurately digitize text from various sources, including paper documents, receipts, and even books. This capability not only enhances user experience but also opens up new possibilities for applications in fields such as document management, data entry automation, and accessibility.

In this guide, we'll explore how integrating OCR into your Android app can provide powerful text recognition features. We'll cover the basics of OCR, its benefits, and how you can leverage an OCR API to bring this technology to your application. Whether you're looking to enhance productivity, streamline data capture, or offer innovative features to your users, understanding and implementing OCR will be a key step in achieving these goals.

1.7 Background of Optical text Recognition (OCR)

Optical Character Recognition (OCR) has a rich history rooted in the quest to digitize and automate the processing of written text. Here's an overview of the evolution and significance of OCR technology:

Early Beginnings

- **1930s–1950s**: The origins of OCR can be traced back to the early 20th century, when inventors and researchers began exploring methods to read printed text automatically. The first practical OCR systems were developed in the 1930s, using mechanical devices and early electronic components.

The Rise of Digital OCR

- **1960s**: The development of electronic OCR technology gained momentum with the advent of digital computers. Pioneering systems used light sensors and early algorithms to recognize characters, though they were limited in accuracy and versatility.

- **1970s–1980s**: Advances in computing power and image processing led to significant improvements in OCR accuracy. During this period, OCR systems began to find applications in various fields, including data entry, document management, and banking.

Modern OCR Technology

- **1990s**: The introduction of more sophisticated algorithms and machine learning techniques marked a new era for OCR. Digital cameras and scanners became widely available, further boosting the adoption of OCR technology.

- **2000s**: The rise of open-source OCR engines like Tesseract and the development of commercial solutions enhanced the accuracy and flexibility of text recognition. OCR became a standard tool for digitizing books, newspapers, and other printed materials.

Contemporary Advances

- **2010s–Present**: Recent advancements in artificial intelligence (AI) and machine learning have transformed OCR into a powerful tool for various applications. Modern OCR systems leverage deep learning and neural networks to achieve high accuracy in recognizing text from diverse sources and complex layouts.

- **Mobile Integration**: With the proliferation of smartphones and mobile applications, OCR technology has become an integral feature in many apps. Mobile OCR enables users to capture and process text on-the-go, making it a valuable tool for personal and professional use.

Significance and Impact

- **Efficiency and Automation**: OCR streamlines the process of converting printed or handwritten text into digital formats, reducing manual data entry and improving workflow efficiency.
- **Accessibility**: OCR technology plays a crucial role in making printed content accessible to individuals with visual impairments, enabling text-to-speech and braille conversion.
- **Information Retrieval**: By digitizing documents and making them searchable, OCR enhances information retrieval and management, facilitating easier access to large volumes of text-based data.

METHODOLOGY Chapter 2

2.1 Requirements Analysis

In developing our Android application, we identified and addressed the following requirements to ensure the app meets user needs and expectations:

2.1 Text Recognition

Accurate OCR: Users require precise optical character recognition to extract English text from images with minimal errors.

2.2 User Interface

Intuitive Design: The app must feature a user-friendly and intuitive interface, allowing easy navigation and interaction.

Responsive Layout: The UI should adapt to different screen sizes and orientations, ensuring a consistent experience on all devices.

2.3 Text Processing

Real-Time Text Extraction: The app should process and display extracted text from images in real-time, enhancing user efficiency.

Text Editing: Users should have the ability to edit and save the extracted text for further use.

2.4 Data Storage and Management

Cloud Integration: Users require reliable cloud storage for saving and syncing data across devices.

2.5 Performance and Reliability

Fast Processing: The app should offer quick text recognition and response times to ensure a smooth user experience.

2.6 Security and Privacy

Data Protection: User data must be securely stored and transmitted, adhering to privacy regulations and best practices.

Permission Management: The app should request and manage permissions responsibly, ensuring user consent for accessing camera and storage features.

By addressing these requirements, the application aims to deliver a reliable, user-centric experience focused on English text recognition.

2.2 Software Development Lifecycle Model

Here the waterfall Software Development Life Cycle model is used.

- **Structured Approach**: Clear and organized, with specific deliverables for each phase.
- **Easy to Manage**: Linear and sequential, making progress easy to track.
- **Client Involvement**: Clients are involved at key milestones, reducing scope creep.
- **Ideal for Well-Defined Projects**: Best for projects with clear, stable requirements.
- **Simplified Testing**: End-to-end testing is more thorough after development is complete.
- **Easier Maintenance**: Detailed documentation and structure facilitate future changes.

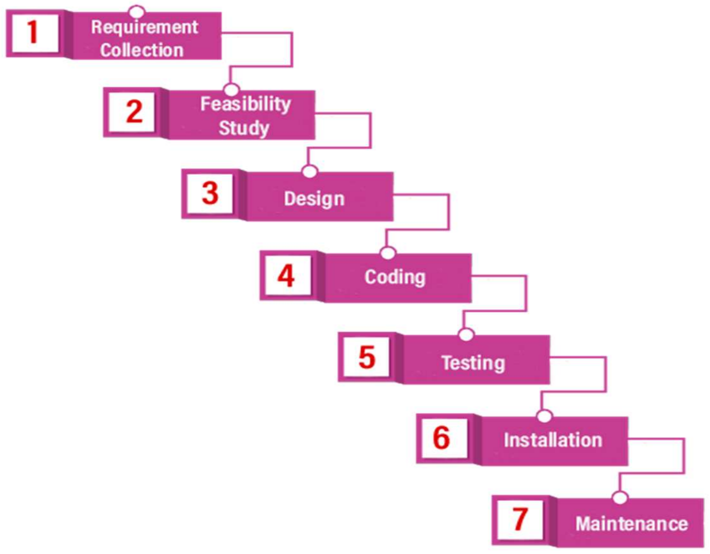


Figure 6: Waterfall SDLC Model

2.3 Flow Chart

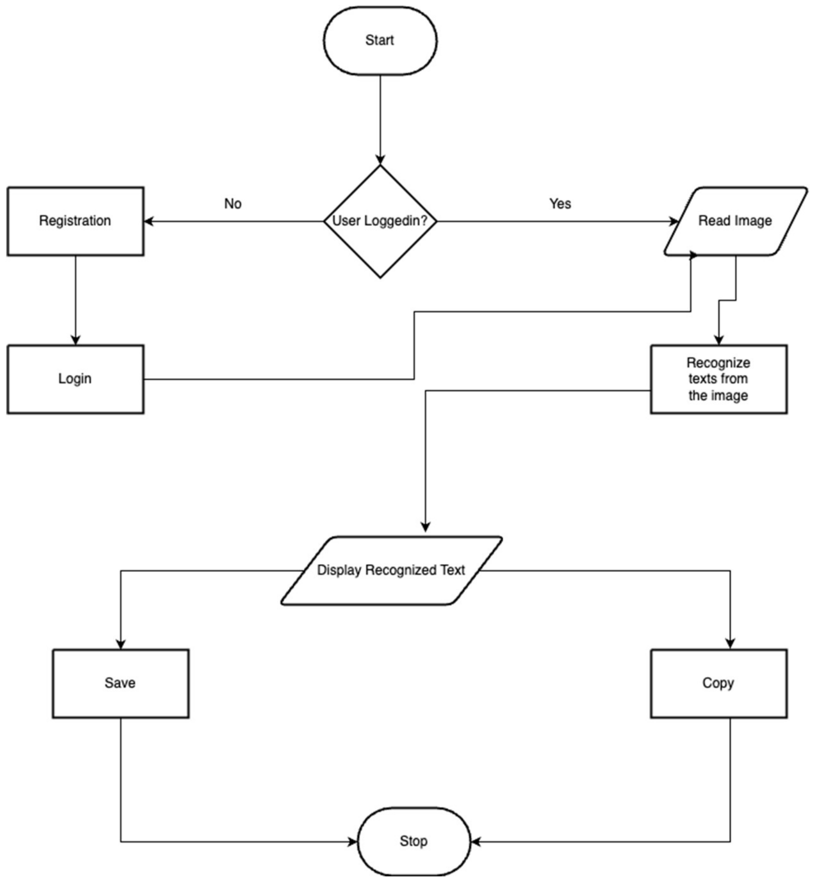


Figure 7: Flowchart of the Application

2.4 Database Design

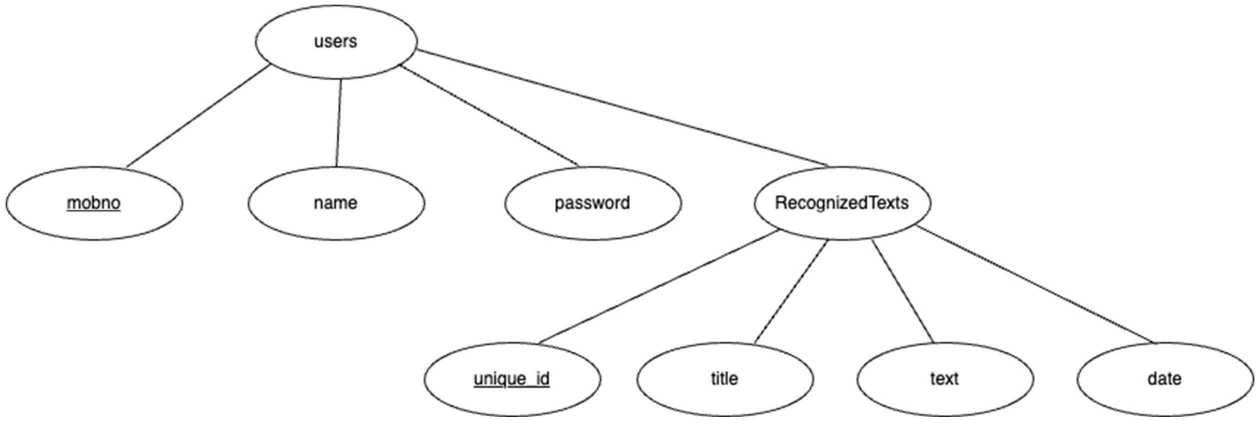


Figure 8: Database Design (Firebase)

2.5 User Interface Design

The user interface (UI) was meticulously designed to ensure a user-friendly experience. Key considerations included:

- **Intuitive Navigation**: Simplified navigation to enhance usability and ensure that users can easily access and use the text recognition features.
- **Responsive Design**: The UI adapts to various screen sizes and orientations, providing a consistent experience across different devices.

By combining these technologies, our application delivers a powerful and efficient text recognition solution, enhancing user experience and providing seamless integration of OCR capabilities within a modern Android environment.

2.6 Technology Used

In the development of our Android application, we leveraged a combination of cutting-edge technologies to deliver a robust and efficient text recognition solution. Here's a breakdown of the technologies and tools used:

2.6.1 Java for Android Development

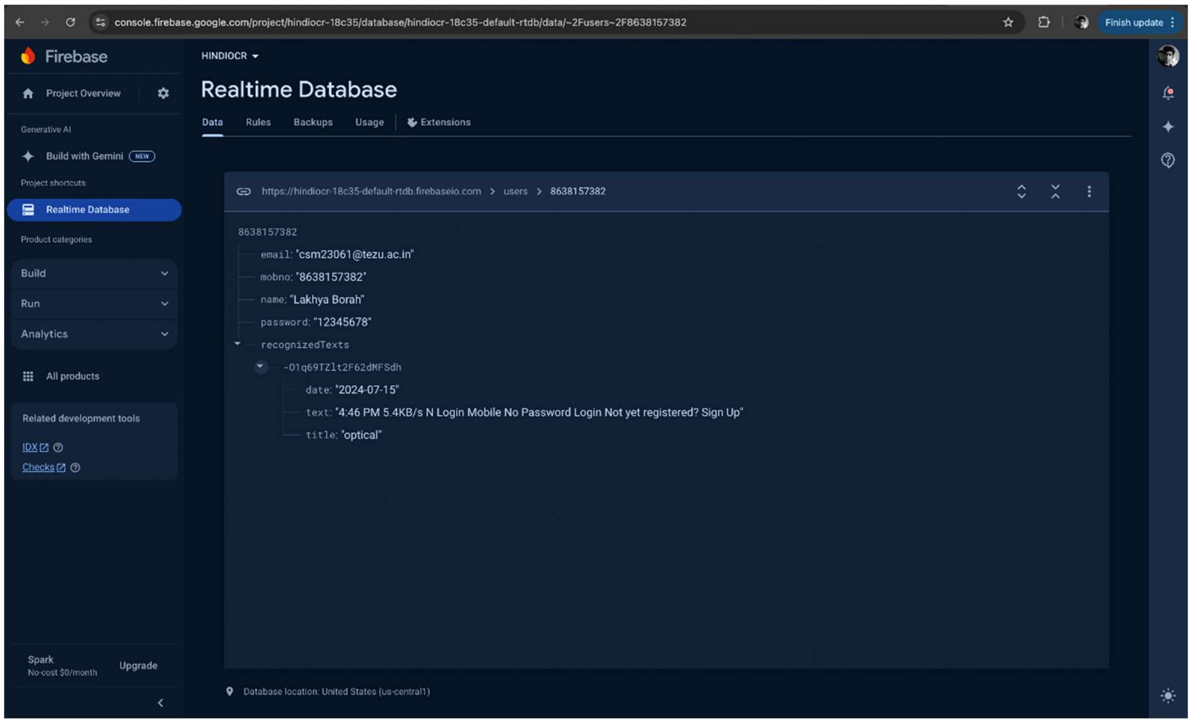
Our application was developed using Java, a versatile and widely-used programming language for Android development. Java provides a strong foundation for building scalable and maintainable mobile applications. Key aspects include:

- **Rich Libraries**: Java offers a rich set of libraries and frameworks that streamline development tasks.
- **Performance**: It provides reliable performance and stability, essential for mobile applications.
- **Community Support**: Extensive community support and resources are available, facilitating problem-solving and innovation.

2.6.2 Firebase for Authentication and Database Management

Firebase, a comprehensive platform developed by Google, was used for authentication and database management. It offers a suite of tools and services that enhance app functionality and security:

- **Firebase Authentication**: Simplifies user sign-in and sign-up processes with support for various authentication methods, including email/password, Google, and social media logins.
- **Firebase Realtime Database**: Provides a cloud-hosted NoSQL database that allows real-time synchronization of data across all clients. This ensures that user data is updated instantaneously and consistently.


console.firebase.google.com/project/hindiocr-18c35/database/hindiocr-18c35-default-rtdb/data/~2Fusers~2F8638157382
Finish update
Firebase
Project Overview
Generative AI
Build with Gemini NEW
Project shortcuts
Realtime Database
Product categories
Build
Run
Analytics
All products
Related development tools
IDX
Checks
Spark
No-cost $0/month
Upgrade
HINDIOCR
Realtime Database
Data
Rules
Backups
Usage
Extensions
https://hindiocr-18c35-default-rtdb.firebaseio.com > users > 8638157382
Database location: United States (us-central1)

```
8638157382
    email: "csm23061@tezu.ac.in"
    mobno: "8638157382"
    name: "Lakhya Borah"
    password: "12345678"
    recognizedTexts
        -O1q69TZlt2F62dMFSdh
            date: "2024-07-15"
            text: "4:46 PM 5.4KB/s N Login Mobile No Password Login Not yet registered? Sign Up"
            title: "optical"
```

Figure 9: Firebase Database Integration

2.6.3 Google's OCR API for Text Recognition

For text recognition, we integrated Google's OCR API, a powerful tool that enables accurate and efficient extraction of text from images. Key features include:

- **High Accuracy**: Google's OCR API leverages advanced machine learning algorithms to deliver high-accuracy text recognition, even with varied fonts and complex layouts.

- **Ease of Integration**: The API is designed to be easily integrated into mobile applications, providing seamless text extraction with minimal configuration.
- **Support for Multiple Languages**: It supports a wide range of languages, making it versatile for applications with diverse user bases.

2.6.4 Development Environment

For developing our Android application, we used Android Studio, the official IDE for Android development. Here are the key features:

- **Code Editor**: Offers intelligent code completion, real-time error checking, and refactoring tools for Java, Kotlin, and XML.
- **Layout Editor**: Provides a visual interface for designing UI with drag-and-drop functionality.
- **Android Emulator**: Enables testing on virtual devices with various configurations and Android versions.
- **Gradle Build System**: Manages the build process, dependencies, and custom build configurations.
- **Version Control Integration**: Supports Git for source code management and collaboration.
- **Performance Profiler**: Monitors CPU, memory, and network usage to optimize app performance.
- **Debugging Tools**: Includes breakpoints, step-through debugging, and logcat for troubleshooting.
- **APK Analyzer**: Inspects APK contents to analyze size and structure.

Android Studio's comprehensive tools streamline development, testing, and deployment, making it an essential environment for building Android apps.

Summery:

Java Version: JDK 21

Android studio version: Android Studio Koala | 2024.1.1

OCR API: ‘com.google.android.gms:play-services-mlkit-text-recognition:19.0.0’

Image Cropping API: ‘com.github.Dhaval2404:imagepicker:2.1’

RESULTS AND DISCUSSION Chapter 3

The development and implementation of the Optical English Character Recognition (OECR) Android application yielded several outcomes. This chapter discusses the results achieved during the project and reflects on their implications, challenges encountered, and potential areas for improvement.

3.1 Application Functionality and Performance

The OECR application was designed with several key features, including user authentication (login and signup), image capture or upload for text extraction, and the ability to copy and save the extracted text as notes. These functionalities were successfully implemented and tested across various scenarios.

1. **User Authentication:** The user authentication system, powered by Firebase. The signup process allowed users to create accounts, and the login system correctly authenticated users based on stored credentials. The seamless integration of Firebase ensured secure storage and retrieval of user data, contributing to the overall reliability of the application.
2. **Image Capture and Upload:** The application provided users with the option to either capture an image using the device camera or upload an existing image from their gallery. This flexibility was well-received during testing, as it catered to different user preferences and use cases. The image capture functionality was responsive, and the image upload process was smooth, with no significant delays or crashes observed.
3. **Text Extraction Accuracy:** The core functionality of the OECR application—extracting text from images—was evaluated using Google's developer API for Optical Character Recognition (OCR). The results were highly accurate, especially for clear, printed English text. The OCR engine efficiently recognized and converted the text into digital format, with minimal errors. The accuracy did, however, vary slightly with the quality of the image; low-resolution or poorly lit images resulted in reduced precision, highlighting the importance of clear image input for optimal results.

4. **Text Management:** Once extracted, the text could be copied to the clipboard or saved as a note within the application. This feature was particularly useful for users who needed to store and retrieve scanned text for later use. The text management system performed as expected, with all saved notes accessible through the application's user interface.

3.2 User Experience and Feedback

During testing, the application was shared with a small group of users for feedback. The general response was positive, with users appreciating the ease of use, intuitive interface, and the accuracy of text extraction. Some users noted the convenience of having an all-in-one solution for scanning and managing text, which could be beneficial for both personal and professional use.

However, the feedback also highlighted some areas for improvement:

- **Image Quality Dependency:** As previously mentioned, the accuracy of text extraction was closely tied to the quality of the image. Users suggested incorporating a built-in image enhancement feature to improve OCR results for lower-quality images.
- **Language Support:** Currently, the application only supports English text recognition. Users expressed interest in having the application support multiple languages, which would significantly broaden its utility.
- **Accessibility Features:** While the application has the potential to aid visually impaired users, the absence of voice feedback or screen reader compatibility was noted. Integrating these features would enhance the application’s accessibility.

3.3 Challenges and Limitations

Throughout the project, several challenges were encountered, particularly in optimizing the OCR process for different image qualities and ensuring the application's stability across various Android devices. Managing different screen sizes and resolutions required additional design considerations to maintain a consistent user experience.

Another limitation was the dependency on Firebase for user data storage. While Firebase provided a reliable backend solution, it introduced a dependency on internet connectivity, which could be a limitation for users in areas with unstable network access.

3.4 Future Work and Enhancements

Based on the results and feedback, several potential enhancements have been identified for future versions of the OECR application:

- **Image Enhancement:** Implementing an image processing module to automatically enhance image quality before text extraction could improve OCR accuracy, especially for lower-quality images.
- **Multilingual Support:** Expanding the OCR capabilities to recognize multiple languages would make the application more versatile and useful to a broader audience.
- **Accessibility Improvements:** Adding voice output and screen reader compatibility would significantly enhance the application's accessibility, making it a more inclusive tool for users with visual impairments.

3.5 Screenshots

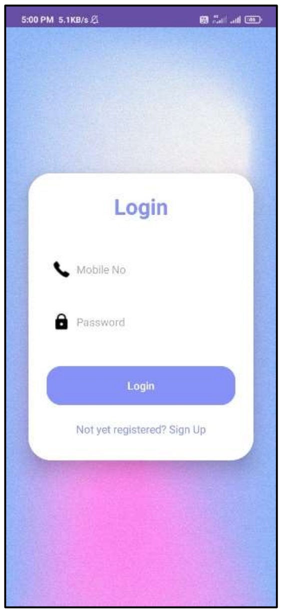


Figure 10: Login

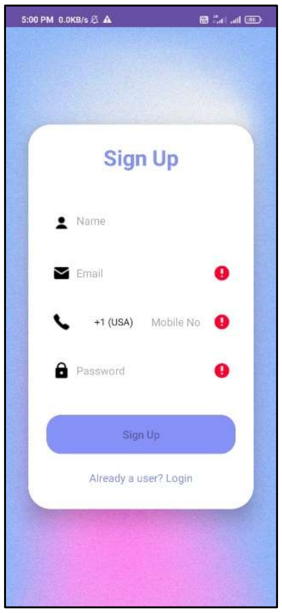


Figure 11: Registration

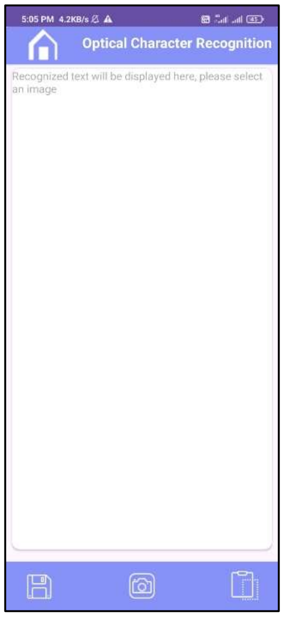

Optical Character Recognition
Recognized text will be displayed here, please select an image

Figure 12: HomePage

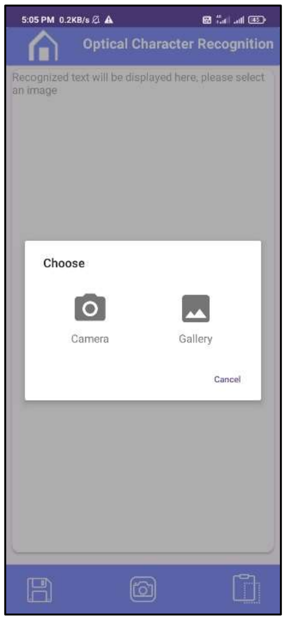
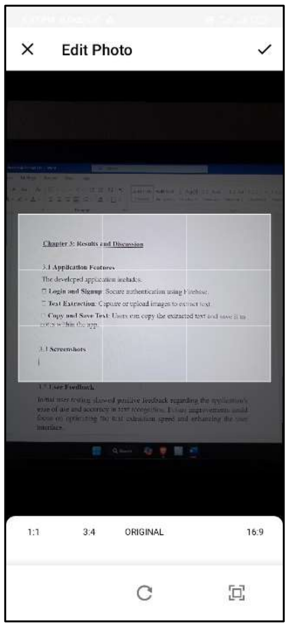

Optical Character Recognition
Recognized text will be displayed here, please select an image
Choose
Camera
Gallery
Cancel

Figure 13: Reading Image



Chapter 3: Results and Discussion

3.1 Application Features

The developed application includes:

□ **Login and Signup**: Secure authentication using Firebase.

□ **Text Extraction**: Capture or upload images to extract text.

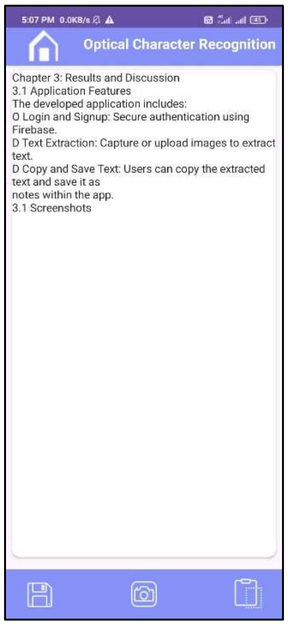
□ **Copy and Save Text**: Users can copy the extracted text and save it as notes within the app.

3.1 Screenshots

3.2 User Feedback

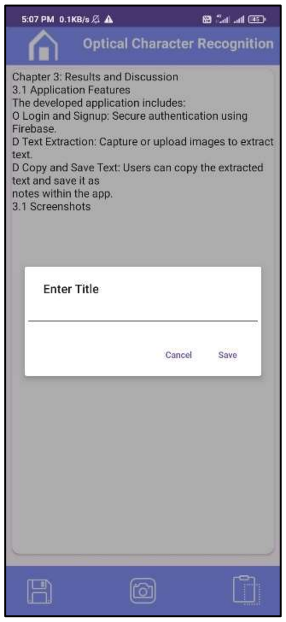
Initial user testing showed positive feedback regarding the application's ease of use and accuracy in text recognition. Future improvements could focus on optimizing the text extraction speed and enhancing the user interface.

Figure 14: Cropping



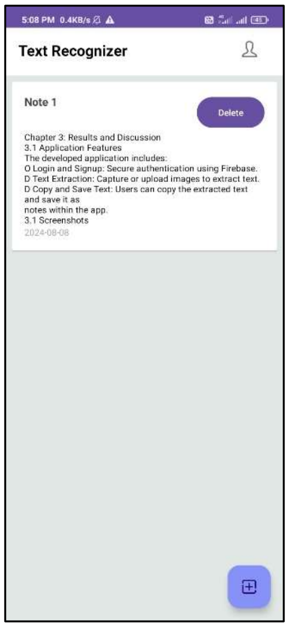
Chapter 3: Results and Discussion
3.1 Application Features
The developed application includes:
O Login and Signup: Secure authentication using Firebase.
D Text Extraction: Capture or upload images to extract text.
D Copy and Save Text: Users can copy the extracted text and save it as
notes within the app.
3.1 Screenshots

Figure 15: Recog. Text



Chapter 3: Results and Discussion
3.1 Application Features
The developed application includes:
O Login and Signup: Secure authentication using Firebase.
D Text Extraction: Capture or upload images to extract text.
D Copy and Save Text: Users can copy the extracted text and save it as
notes within the app.
3.1 Screenshots

Figure16 : Saving Text



Chapter 3: Results and Discussion
3.1 Application Features
The developed application includes:
O Login and Signup: Secure authentication using Firebase.
D Text Extraction: Capture or upload images to extract text.
D Copy and Save Text: Users can copy the extracted text and save it as
notes within the app.
3.1 Screenshots
2024-08-08

Figure 17: Records

CONCLUSIONS Chapter 4

The development of the Optical English Character Recognition (OECR) Android application during my internship at the Department of Computer Science & Engineering, NIT Mizoram, has been a significant learning experience. The project provided an opportunity to apply theoretical knowledge in a practical setting, leading to the successful implementation of a functional mobile application that addresses the challenge of converting printed English text into editable digital format.

Through this project, I gained hands-on experience in Android app development using Java and enhanced my understanding of integrating third-party APIs, specifically Google's developer API for Optical Character Recognition (OCR). The use of Firebase for user authentication and data storage further broadened my knowledge of backend services and cloud-based data management.

The application successfully meets its objectives by offering features such as user authentication, image capture and upload for text extraction, and the ability to copy or save the extracted text as notes. The accuracy and efficiency of the text extraction process underscore the effectiveness of the implemented technology, making the application a viable tool for digitizing printed materials.

Moreover, the potential applications of this OECR app extend beyond document digitization. It can serve as an essential tool for visually impaired individuals, enabling them to access printed text in a digital format. The project not only addresses a current technological need but also opens up avenues for future enhancements, such as expanding language support and integrating voice output for accessibility.

In conclusion, this internship project has provided me with a comprehensive understanding of the software development lifecycle, from initial concept and design to implementation and testing. It has reinforced my problem-solving skills, fostered creativity, and equipped me with the technical skills necessary for future endeavors in the field of software engineering. The experience has been invaluable in preparing me for a career in technology, with a particular focus on developing solutions that enhance accessibility and usability for diverse user groups.

References

1. Kamal, R. (2009). **Mobile Computing: Principles, Designing and Development**. Oxford University Press.

2. Phillips, B., & Stewart, C. (2019). **Android Programming: The Big Nerd Ranch Guide** (4th ed.). Big Nerd Ranch Guides.

3. Bunke, H., & Wang, P. S. P. (1997). **Handbook of Character Recognition and Document Image Analysis**. World Scientific.

4. Murphy, M. L. (2018). **Firebase Essentials: Android Edition**. CommonsWare, LLC.

5. Firebase Documentation https://firebase.google.com/docs/android/setup

6. https://developers.android.com

7. https://developers.google.com