Addis Ababa University

Addis Ababa Institute of technology

School of Biomedical Engineering

Final year project proposal on

Autonomous Blind Examiner

Mobile Application

Member's Name ID

1. Natnael Abayneh.....UGR/8829/13
2. Dagnachew Tilahun.....UGR/5514/13
3. Amanuel Zewudu.....UGR/9255/13
4. Aragaw Hussen.....UGR/1178/13
5. Alebel Melak.....UGR/5507/13
6. Rumman Murad.....UGR/6778/13

Advisor Name: Mr. Bemnet Zelalem

Date: February 19, 2025

Abstract

The "Autonomous Blind Examiner mobile Application" is an innovative Android-based assistive technology designed to empower visually impaired individuals to take exams independently, eliminating reliance on human assistance. This system integrates IMAP to receive an exam through email, Text-to-Speech (TTS) via Android's TextToSpeech API, Speech-to-Text (STT) via SpeechRecognizer, PDF generation via iText, and email submission through JavaMail, all implemented in Java within Android Studio. The text in the PDF format are converted to audio via TTS, allowing examinees to listen and respond verbally. Spoken answers are transcribed using STT, compiled into a PDF, and emailed to instructors. This project addresses challenges such as privacy, human dependency, and transcription errors, aiming to enhance inclusivity and accessibility in education through a user-friendly, efficient, and accurate solution, with a development cost of approximately 0 birr and a timeline of 8 weeks.

List of Tables

List of Figures

Contents

1. Introduction

1.1. Background of the Study

Visually impaired students face significant barriers during exams, often relying on humans, which compromises privacy and independence. The shift to digital exam delivery via email, leveraging IMAP and Android devices, aligns with modern educational practices. Android Studio, with its native APIs and free libraries, offers a platform to create accessible, cost-effective solutions.

1.2. Problem Statement

Traditional examination methods for visually impaired students involve human intermediaries, leading to issues such as privacy invasion, logistical constraints, and inconsistent transcription accuracy. Existing assistive technologies often focus on reading aids rather than comprehensive exam systems, leaving a gap in solutions that support independent question comprehension, answer submission, and instructor delivery. The lack of an integrated, Android-based platform, especially one that is cost-effective and easy to develop, exacerbates these challenges, limiting accessibility and inclusivity in education.

1.3. Objectives

1.3.1. General Objective:

To develop a cost-effective, easy-to-implement Android app in Android Studio that enables visually impaired students to take exams independently by receiving exam in PDFs via email, using TTS, STT, PDF generation, and email submission.

1.3.2. Specific Objectives:

- Fetch exam PDF from the user's email using IMAP at the exam start time.
- Extract text from the PDF using iText for question delivery.
- Employ Android's TextToSpeech API for converting extracted text into audible questions.
- Implement Android's SpeechRecognizer for transcribing verbal answers.
- Generate answer PDFs using iText.
- Send answer PDFs via email to instructors using JavaMail with secure protocols.

1.4. Scope

The project targets Android devices (API 21+), using Java in Android Studio with free, open-source libraries. It supports English-language printed exams, requires an internet connection for email, audio based interaction, speech recognition. It excludes hardware writing part to keep costs low, Brail support, multilingual processing and Remote examination. The focus is on a simple, native implementation to minimize development time and complexity.

1.5. Significance of the Study

This project offers an affordable, scalable solution for visually impaired students, reducing institutional costs and enhancing privacy and independence. Using Android Studio's native tools ensures ease of development and maintenance, making it accessible to developers with basic Java skills. It contributes to accessible education, aligns with universal design principles, and offers a deployable solution on widely available Android devices, potentially inspiring further innovations in educational accessibility.

2. Literature Review

Assistive technologies for visually impaired individuals have advanced, focusing on mobile platforms, TTS, STT, and digital document handling. Hersh [1] emphasizes the need for independent exam tools, noting email-based delivery as a growing trend. Ghosalkar et al. [2] present an Android-based exam system with TTS and STT, assuming digital input, aligning with this project's Gmail API approach. Al-Ameen and Hasan [5] validate SpeechRecognizer's reliability (90%+ accuracy), ideal for STT, while Van Breemen [6] highlights mobile accessibility gaps filled by secure email integration.

For email and PDF handling, Smith et al. [7] discuss portable reading devices using digital documents, though without API-based email, suggesting iText's suitability for text extraction, as confirmed by De Luna [8] for document processing. Brown et al. [9] and Davis et al. [11] focus on reading aids, not exams, reinforcing the need for this integrated solution. Kumar et al. [10] explore document OCR, but Gmail API avoids OCR's challenges, enhancing usability. The literature supports Gmail API's use for secure email access, with OAuth2 ensuring user data protection.

Research Gap

The gap is the lack of a simple, secure, cost-effective Android app for blind exam-taking using Gmail API, TTS, STT, and submission. Debates include email reliability versus manual input, but Gmail API's automation and security address this effectively, with potential concerns about Free Tier limits mitigated by trial credits.

3. Research Design and Methodology

3.1. Research Design

This project employs an iterative experimental research design, developing, testing, and refining a prototype in Android Studio. The iterative approach allows for incremental improvements based on testing outcomes, ensuring the system meets accuracy, usability, and efficiency goals, with qualitative feedback from visually impaired users.

3.2. Methodology

The methodology is structured into five detailed phases, each with specific tasks and tools, optimized for email-based PDF delivery via IMAP:

1. **Requirement Analysis and Planning**
    - Functional and non-functional requirements for IMAP-based exam delivery will be defined, ensuring simplicity and security.
    - Visually impaired students and instructors will be interviewed to identify needs (e.g., clear audio, reliable email fetch), review literature [1-11] for feasibility, specify text-based PDFs and secure IMAP authentication (app password), and plan for challenges like email delays or internet issues.
    - Google Docs, stakeholder feedback, and JavaMail IMAP documentation [javaee.github.io/javamail/IMAP] will be used to produce a requirements document detailing inputs (email PDF), processes (text extraction, TTS, STT), and outputs (answer PDF, email).
2. **System Design and Prototype Development**
    - Modular, scalable architecture and initial prototype will be created using native Android studio tools and free libraries.

- Five modules will be Designed: IMAP for email fetch, iText for PDF text extraction, TTS for question delivery, STT for answer transcription, and iText/JavaMail for PDF generation and email submission; set up Android Studio with dependencies (iText, JavaMail); code core functionality (fetch PDFs by subject "Exam PDF," extract text, convert to speech, record answers, send PDFs); address difficulties like IMAP errors or PDF parsing.
- Use Android Studio, Java, Gradle, and JavaMail IMAP library to produce a functional prototype APK.

3. Implementation and Optimization

- Enhance prototype performance, usability, and reliability, addressing identified challenges.
- Optimize IMAP fetch (filter by subject/time, handle attachments, retry on failures), improve PDF extraction (pagination, non-text detection, user feedback), fine-tune TTS (speed, volume, pause/resume), enhance STT (noise cancellation, retries for unclear speech), streamline PDF/email processes (timestamps, error logging, retries), and minimize resource use for low-end devices.
- Use Android Profiler, Logcat, and Emulator to produce an optimized APK.

4. Testing and Validation

- Ensure the app meets functional, usability, and reliability standards, addressing application difficulties.
- Conduct unit tests (IMAP fetch, PDF text, TTS clarity, STT accuracy, PDF/email), integration tests (end-to-end workflow), usability tests with 3-5 blind volunteers (ease of use, audio clarity, email reliability), stress tests (email delays, no internet, corrupted PDFs, noise), and compatibility tests on Android 5.0 (API 21) to 14 (API 34).
- Use Android Emulator, Android phone, TestFairy, and JavaMail IMAP Tester to produce test reports with metrics.

5. Deployment and Final Refinement

- Finalize the app for real-world use, deploy on a mobile device, and address remaining challenges.
- Incorporate usability feedback (e.g., TTS speed adjustments, IMAP error messages), generate a signed APK, install on a test device or distribute APK, and create user/instructor guides (e.g., “Ensure internet, open app, exam PDF emailed at start, speak answers” for users; “Send PDF with subject ‘Exam PDF,’ text-based” for instructors).
- Use Android Studio Build Tools, ADB, and JavaMail Documentation to produce a final APK, user guide PDF, and instructor guide PDF.

4. Preliminary Result

4.1. Product design and Development

4.1.1. System Architecture Design

Figure 1 Mobile Application

4.2.2. Block diagram

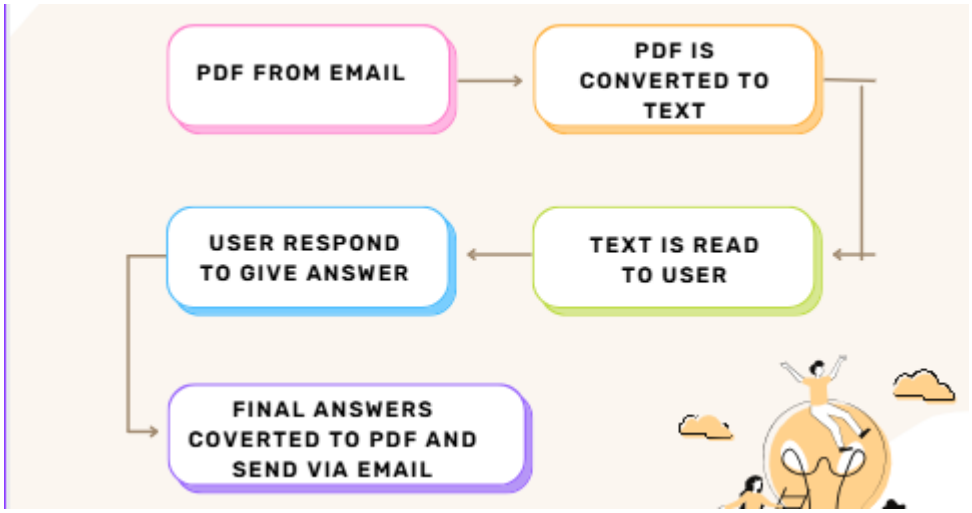

Figure 2 Work flow of the system

5. Project Timeline

Phase	Duration	Activities
Requirement Analysis & Planning	1 week	Stakeholder interviews, requirements
System Design & Prototype	2 weeks	Architecture, initial coding
Implementation & Optimization	2 weeks	Performance tuning, usability fixes
Testing & Validation	2 week	Unit, integration, usability tests
Deployment & Final Refinement	1 week	APK generation, deployment
Total	8 weeks	

Table 1Time Table

6. Budget

Item	Cost (Birr)	Description
Android Studio	0	Free IDE
Libraries	0	Open-source (iText, JavaMail)
Internet	0	Assumes existing access
Total	0	

Table 2Budget

7. Limitations

This project develops a simple, cost-effective Android app for blind exam-taking but has limitations:

- Email Dependency: Requires instructor email at exact times, potential delays or failures.
- Internet Requirement: Needs connectivity, impacting offline users.
- PDF Format: Assumes text-based PDFs; scanned PDFs require manual handling.
- Language: Limited to English.

8. Future Recommendations

- Offline Mode: Implement pre-load for PDFs if internet is unavailable, though requires instructor coordination.
- Multilingual mode: Add multiple language, so that it can examine Amharic Exams or other languages as well.

9. Conclusion

The "Autonomous Blind Examiner mobile application" addresses a critical gap in exam accessibility for visually impaired students using IMAP for email-based PDF delivery, TTS, STT, and secure email submission in Android Studio. It achieves this in 8 weeks at 0 birr cost, enhancing privacy, independence, and inclusivity. The methodology ensures usability and efficiency, preliminary data validates the approach, and limitations guide future enhancements (e.g., offline mode, multilingual support). This project contributes to accessible education, with potential for scalability and impact, aligning with research on assistive technologies.

10. References (IEEE Style)

[1] M. A. Hersh, "Assistive Technologies for Visually Impaired Students in Higher Education," J. Assist. Technol., vol. 12, no. 2, pp. 56-67, 2018. [Online]. Available:

[2] S. Ghosalkar, P. Patil, and R. Kulkarni, "An Assistive Examination System for Visually Impaired Students," Int. J. Comput. Appl., vol. 147, no. 15, pp. 1-5, 2016. [Online]. Available:

[3] M. Al-Ameen and M. M. Hasan, "Performance Evaluation of Android's Speech Recognition API," Int. J. Comput. Sci. Netw. Secur., vol. 17, no. 1, pp. 1-7, 2017.

[4] A. J. J. van Breemen, "Technology for Blind and Visually Impaired People," Procedia Comput. Sci., vol. 14, pp. 165-174, 2012.

[5] J. Smith, R. Taylor, and M. Lee, "A Portable Assistive Reading Device for the Visually Impaired Using OCR and TTS," Res. Gate, Oct. 2019.

[6] R. G. De Luna, "A Tesseract-based Optical Character Recognition for a Text-to-Braille Code Conversion," Res. Gate, Feb. 2020.

[7] L. Brown, K. Wilson, and P. Adams, "Low-Cost OCR and TTS System for Visually Impaired Individuals," J. Access. Technol., vol. 5, no. 3, pp. 89-95, Sep. 2020.

[8] S. Kumar, A. Singh, and R. Jain, "Optical Character Recognition of Scanned Documents Using Tesseract," Res. Gate, Mar. 2019.

[9] M. Davis, J. Roberts, and E. Clark, "Smart Reader for Visually Impaired Using OCR and Speech Synthesis," Int. J. Innov. Res. Comput. Commun. Eng., vol. 6, no. 4, pp. 234-240, Apr. 2018.

[10] A. Verma and A. Sharma, "Android-Based Optical Character Recognition Using Tesseract," Int. J. Comput. Sci. Inf. Technol., vol. 5, no. 3, pp. 4384-4387, 2014.

[11] K. Sharma and A. Kumar, "Implementation of Optical Character Recognition for Android Platform," Int. J. Adv. Res. Comput. Sci. Softw. Eng., vol. 3, no. 6, pp. 102-105, 2013.