applied sciences

MDPI

Article

Mathematical Background and Algorithms of a Collection of Android Apps for a Google Play Store Page

Roland Szabo

Faculty of Electronics, Telecommunications and Information Technologies, Politehnica University Timisoara, Vasile Parvan Av., No. 2, 300223 Timisoara, Romania; roland.szabo@upt.ro; Tel.: +40-256-40-3351

Abstract: This paper discusses three algorithmic strategies tailored for distinct applications, each aiming to tackle specific operational challenges. The first application unveils an innovative SMS messaging system that substitutes manual typing with voice interaction. The key algorithm facilitates real-time conversion from speech to text for message creation and from text to speech for message playback, thus turning SMS communication into an audio-focused exchange while preserving conventional messaging standards. The second application suggests a secure file management system for Android, utilizing encryption and access control algorithms to safeguard user privacy. Its mathematical framework centers on cryptographic methods for file security and authentication processes to prevent unauthorized access. The third application redefines flashlight functionality using an optimized touch interface algorithm. By employing a screen-wide double-tap gesture recognition system, this approach removes the reliance on a physical button, depending instead on advanced event detection and hardware control logic to activate the device's flash. All applications are fundamentally based on mathematical modeling and algorithmic effectiveness, emphasizing computational approaches over implementation specifics.

Keywords: electronic messaging; file systems; mobile applications; mobile computing; utility programs

check for updates

Academic Editor: Douglas O'Shaughnessy

Received: 7 March 2025
Revised: 31 March 2025
Accepted: 1 April 2025
Published: 17 April 2025

Citation: Szabo, R. Mathematical Background and Algorithms of a Collection of Android Apps for a Google Play Store Page. *Appl. Sci.* **2025**, *15*, 4431. https://doi.org/10.3390/app15084431

Copyright: © 2025 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

This study emphasizes the mathematical precision underlying efficient and scalable solutions by removing platform-specific concerns. The varied applications illustrate how algorithmic principles can be adapted to different fields, such as human–computer interaction and information security.

The study investigates the mathematical principles and algorithmic structures behind three separate applications, each tackling specific computational problems. Although these implementations are designed for Android platforms, the emphasis remains on their theoretical foundations rather than on the deployment details.

The primary scientific advancement lies in the creation of innovative algorithms that drive these applications, examined through mathematical modeling and performance improvements. Each application utilizes its own unique methods. For visually impaired individuals, voice SMS writing and reading depend on speech recognition algorithms for immediate conversion from voice to text and synthesis from text to speech, structured within probabilistic language models and signal processing. Private file management ensures data security by applying cryptographic protocols, using encryption algorithms, and access control systems based on computational complexity theory. The torch activated by double-tapping anywhere on the screen employs touch-event classification algorithms,

including detection of double-tapping using temporal pattern analysis and hardware interface refinement.

In order to adapt the mathematical precision, the algorithms and apps should focus on security, to simplify their use and increase their privacy. Therefore, apps are selected to demonstrate our approach.

1.1. Vocal SMS Writing and Reading App for Visually Impaired People

We will present the mathematical background and algorithm for a voice SMS app. SMS (Short Message Service) messaging is one of the most used means of communication, due to its simplicity and speed [1]. SMS messaging is quite an old form of communication, being introduced in the 1980s. It has survived quite a few generations of mobile networks, and it is quite often used even in 5G networks [2]. It has the advantage that the receiver is not interrupted in their work, giving them the possibility to read the message and respond when they have some free time [3]. SMS messaging outlived MMS (Multimedia Message Service) messaging and coexists with modern chat clients that communicate over the Internet [4]. SMS is still popular due to its accessibility and the possibility of reaching places without Internet access, with much greater coverage than chat clients. It also has a guarantee that the message it is sending is the same, since it is a paid service [5]. The fees for SMS messaging are not high; some operators have prepaid options to send a specific number of SMS messages at a fixed price, which greatly reduces the price of a message, and SMS transmission is almost free [6]. SMS messaging is a very inexpensive and easy way to communicate, but there may be a way to make it even simpler [7]. There are problems such as the fact that there are people who cannot use the system or situations in which it cannot be used [8]. This can be the case for visually impaired people, children who have not yet learned to write, busy people who have no free hands, or car drivers. It would be a good idea to create a system in which SMS messages can be handled vocally, like a phone call, but for the price of an SMS message. This can be achieved using speech recognition and text-to-speech software. The interaction is made vocally through speaking and hearing, the system performs the speech-to-text and text-to-speech transformations, and the message is sent for the reduced price of an SMS message.

Nayak et al. (2020) introduced an application designed to manage schedules, send emails, and read SMS messages, all through voice commands. This system allows users to execute tasks by vocalizing commands that are processed by a Speech Recognition Engine (SRE) and translated into text or corresponding actions [9].

In Table 1, a comparison of the presented vocal SMS app with the state of the art can be seen.

Prasanna et al. (2021) concentrated on creating a mobile application aimed at automating SMS systems via voice conversion methods, particularly tailored for people with visual impairments. This app allows users to compose and access messages using voice commands, thus improving their independence [10].

Creating a vocal SMS app requires the possibility to access the user's SMS database, which is harder in each newer Android version due to restrictions for the privacy of the user. Harder-to-obtain databases mean more coding and sometimes forms to be filled in and accepted by Google. Another requirement is to have a good text-to-speech and speech-to-text software to generate text for SMS vocally and read a received SMS out loud. The goal is to make an SMS client that can be used for vocal phone conversations by visually impaired people, drivers, or busy people. Until now, no app has succeeded in this for some reason, and none of them in the presented way.

Table 1. Comparison of the presented vocal SMS app with the state of the art.

Feature	Vocal SMS App	Nayak et al. (2020) [9]	Prasanna et al. (2021) [10]
Objective	Exchange of messages via text for individuals with visual disabilities	Mobile application designed to assist individuals with visual impairments	Service for automated text messages featuring voice translation tailored for visually impaired individuals
Input Method	Voice input incorporated for SMS creation and sending	Auditory guidance for navigation and engagement	Converting spoken words into text for SMS messaging
Output Method	Text to speech for reading messages aloud	I'm sorry, but I need the LaTeX text you want to be paraphrased.	Text to speech for reading SMS messages
Additional Features	Facilitates the use of various languages, includes message preview, and allows for editing	Object identification, navigational support	An automated system for sending SMS using preset messages
Target Audience	Mobile users who are visually impaired	Individuals with visual impairments requiring help with everyday activities	Individuals with visual impairments needing SMS communication
Platform	Android	Mobile application running on Android	Mobile application running on Android
Evaluation	User evaluation with individuals who are visually impaired	Testing usability with users who are visually impaired	Evaluation of performance and precision assessment

1.2. Private File Manager App

We will present the mathematical background and algorithm for a private file manager app. In the current world, smartphones and tablets have entered at light speed; nowadays, hardly anyone in cities has not owned at least one smartphone or tablet [11]. Smartphones or tablets can also be very intimate, storing a lot of sensitive user information [12]. Having access to someone's smartphone or tablet is almost like having access to that person's digital life [13]. Smartphones and tablets can store many accounts, passwords, and almost any information that can be hacked if the person is the target of a cyber attack [14]. If someone installs an app, it will allow the creator of the app to have access to a lot of sensitive user information, that is, the app can access a very intimate part of the user's digital life [15]. As time passes, many new applications appear, each of them wanting access to many intimate data [16]. Today, data privacy is very important; there have many scandals of private information being released, a lot of privacy consent is given, and GDPR declaration forms have to be accepted [17]. As time passes, privacy will become more important and much effort will be made to protect privacy [18]. After each version of the Android operating system is released, the developers analyze privacy issues and perform software updates to increase privacy [19]. In other words, each version is more and more secure [20]. To come in handy for this privacy issue, this article presents the creation of an app that increases privacy in a way that was not thought of when creating updates to the Android operating system until now (Android 12L (API 32) (Google, Mountin View, CA, USA)). The app uses the functionality of Linux (because Android is a Linux-based operating system), where files beginning with a dot (".") are hidden. The app renames the files to begin with a dot (".") To unhide them or reverse the process, the dot (".") is removed from

the beginning of the filename. The app can be used to hide images or videos from the Gallery app on the Android phone or tablet. The app can also hide other files, but images or videos are perhaps some of the most sensitive data for the average user of a smartphone or tablet. The app can be accessed with a password, so only the owner of the smartphone or tablet can hide images or videos from the Gallery app on the Android smartphone or tablet. After the images and videos are hidden in the private file manager app, anyone who accesses the Gallery app of the Android smartphone or tablet will be able to view only the files the user allows anyone to view, and the files the user had posted or would post to social media sites. The app is special because it is not very easy to create an app that shows all hidden files, and on Android, the standard file manager cannot really hide all files, especially not images or video files. The method of hiding files is carried out with external apps with the ".nomedia" file in a folder, but this will hide an entire folder, not each folder/file selected individually by the user. Also, file hiding can be problematic in an Android system because sometimes the file system needs to be refreshed or the operating system needs to be rebooted. Sometimes, the file is renamed to begin with a dot ("."), but is still visible; in this situation, a phone reboot is needed. Because the app deals with sensitive data, such as images and videos, the app must be robust to prevent the user from losing any of their files. There is also the issue that not all file managers let the user rename a file to begin with a dot (".") to hide it. Simply, doing this will not rename the file; it will allow one to rename the file to anything else, but not begin with a dot (".). A command must be found that will rename a file to begin with a dot (".").

Ontology-driven strategies have been suggested to improve file management by supporting semantic structure and retrieval. Rompa et al. (2018) presented OntoFM, a personal file manager grounded in ontology, which allows users to perform semantic searches within their information space. OntoFM leverages ontology relationships to display files linked to distinct concepts, thus helping to intuitively search for and retrieve information [21].

Decentralized file storage systems provide a different approach to conventional centralized storage by spreading data over numerous nodes, which improves security and resilience. Zichichi et al. (2020) explored the effectiveness of decentralized file storage in managing personal information systems by employing Distributed Ledger Technologies (DLTs) and decentralized file storage (DFS) frameworks. Their architecture guarantees the immutability, traceability, and verifiability of the data, complying with regulations such as GDPR [22].

Merkle-tree-based file systems utilize cryptographic frameworks to guarantee data integrity and allow efficient verification. In 2019, Kan et al. introduced MTFS, a file system built on Merkle trees tailored for private file storage within decentralized settings. MTFS integrates sophisticated proxy re-encryption techniques to enable secure file transactions, maintaining balanced data distribution across service nodes without relying on a central authority [23].

In Table 2, a comparison of the presented file manager application with the state of the art can be seen.

User-centric methodologies focus on allowing individuals to control their data, highlighting the importance of confidentiality, integrity, and performance. In 2024, Bozorgi et al. presented UPSS, a user-centric private sharing framework that operates either as a traditional file system or as a base for security-sensitive applications. UPSS ensures robust security and effective performance in untrusted storage, supporting applications such as integrity-preserving redaction and private revision management [24].

Table 2. Comparison of the presented file manager app with the state of the art.

Feature	Private File Manager App	Rompa et al. (2018) [21]	Zichichi et al. (2020) [22]	Kan et al. (2019) [23]	Bozorgi et al. (2024) [24]
Purpose	Management of secure files on mobile devices	Organizing files using ontologies	Personal data storage using decentralized systems	File system based on Merkle trees for ensuring integrity	User-focused personal storage framework
Data Storage	Local storage	Structured storage based on ontology	Blockchain-enabled and distributed storage	Hierarchical storage using Merkle trees	User-managed, secure storage
Security	AES encryption	Mechanisms for controlled file access	Blockchain-enabled redundancy and cryptography	Tamper resistance and hash-based verification	User-specific access control combined with robust encryption
Accessibility	Interface optimized for mobile use with offline capabilities	Desktop system utilizing ontology-driven search	Access that is both decentralized and interconnected	Decentralized and secure retrieval system	Secure access across multiple devices
Platform	Android	Desktop	Distributed network	System integrated with blockchain technology	Personalized confidential storage setting
Key Advantage	Improved confidentiality and protection for individual documents	Smart document arrangement	Robustness and resilience to failures	Verification and integrity of immutable files	Comprehensive oversight of personal data paired with strengthened security measures

The requirement for creating a private file manager app is to have access to the users' files, which is harder in any new Android version, due to increasing user privacy. Harder-to-obtain files mean more coding and sometimes forms to be filled in and accepted by Google. Then, some encryption is needed and some Unix file operation knowledge to easily hide and show sensitive user files, like photos and videos. Based on the fact that it seems to be so simple, too few apps have managed to implement this functionality and none of them in this simple way.

1.3. Novel Torch App Which Can Be Switched by Double-Tapping Anywhere on the Screen

We will present the mathematical background and algorithm for a novel flashlight app, which is a much-needed tool and can be used when the user wants to search for an object in darkness, to find the lock on a door, or even to send signals at night [25]. In today's world driven by mobile devices, the flashlight is a very popular app, because almost all devices have a camera with a flashlight [26]. The flashlight can be used to take a picture, but not only [27]. It can be used separately as a light in torch mode [28]. Because the flashlight is connected to the camera, camera permissions are needed to use this, and this can sometimes be uncomfortable for the user [29]. The flashlight app is one of the most popular apps, but one of the apps that can fail very much on the Google Play Store, because there are many versions and it is also included in the newer Android distributions [30].

In Table 3, a comparison of the presented novel torch app with the state of the art can be seen.

It is far too hard to come up with something new to make the app attractive or to convince many users to download the app [31]. The Google Play Store is working on marketing many apps that are not downloaded, because even in the beginning, there were more than 2000 flashlight apps on the Google Play Store with 0 downloads and the number is growing. This fact can discourage young programmers with start-up ideas that fail before they even start, and this cannot be beneficial to the app developer community. Based on the facts presented above, it is much too hard to make a flashlight app that can even receive more than 1 download. The only way to create a flashlight app that can have more downloads is to analyze as many flashlight apps as possible and find what functions or features they are missing. During this work, it was discovered that one of the main functions that many mobile apps and implicitly flashlight apps lack is accessibility. The aim of this work is to find a method to control the flashlight with one tap on the touchscreen of the mobile device in any place, regardless of whether the mobile device is locked or not. In this way, the hardest task is to protect the app from accidentally turning the flashlight on/off.

Certain studies indicate that some flashlight applications seek permission for sensitive device functionalities, such as contacts, location, and microphone, that do not pertain to their core purpose. Such extensive permission requests may result in the collection and exploitation of unauthorized data by malicious actors. For example, Collobert et al. (2002) emphasize the importance of modular software libraries in safeguarding against unauthorized access and protecting user privacy [32].

Table 3. Comparison of the presented novel torch app with the state of the art.

Feature	Novel Torch App	Ma et al. (2016) [33]
Purpose	Offers an elementary flashlight capability	Development and deployment of a smart LED lighting system controlled via a mobile application
Technology Used	LED integrated into the mobile device	Incorporation of an LED lighting setup connected to a mobile application using Bluetooth or Wi-Fi
Control Method	Toggle manual control on/off	The intelligent LED light is operated via a mobile app, facilitating brightness and color modifications.
Power Source	Device's power source	LED light relies on an external power source; the mobile app operates on a smartphone.
Smart Features	Simple toggle mechanism for the flashlight	Smart functionalities include adjusting brightness, altering colors, and optimizing energy use
Connectivity	No additional LEDs are required, utilizes its internal LED	Connectivity between the mobile application and the LED system can be established via Bluetooth or Wi-Fi
Application Area	Emergency, fundamental illumination	Personalized LED management in smart home lighting systems
Hardware Requirements	Mobile phone featuring LED	External LED lighting system compatible with mobile applications
Software Requirements	App for flashlight	An innovative mobile application designed for LED light management, offering smart functionalities
User Interaction	Turn the flashlight on/off with a simple touch	Enhanced interface for managing LED light properties including brightness and color

The requirement for creating a torch app is to have camera permission, because the flashlight on the mobile phone is made for the video camera on the phone. This permission

is harder to obtain for every new Android version, due to the fact that it is declared to be a sensitive permission. Harder to obtain means more coding and sometimes forms to be filled in and accepted by Google. It is quite strange that there are so many flashlight apps on the Google Play Store and none of them has made an implementation where the torch can be toggled while pressing anywhere on the mobile phone's screen. This is useful because we usually use a flashlight in dark conditions where we need to search for the turn on flashlight button, but we can just tap anywhere on the screen.

The three apps share a similar mathematical background, because all three apps are focused on user privacy, encoding, and also optimizing the mobile phone's battery usage.

2. Problem Formulation

The current trends in mobile applications are mostly related to artificial intelligence and machine learning. The following apps have not incorporated artificial intelligence, but there is the possibility to do so.

2.1. Vocal SMS Writing and Reading App for Visually Impaired People

The task was to present the mathematical background and algorithm for a new and reinvented SMS application for mobile devices. The app should be able to access the main SMS database from the mobile device and be able to send and receive SMS messages with vocal assistance. SMS writing should be carried out with speech recognition and text transformation. SMS reading should be carried out with text-to-speech software, so the received SMS is read aloud by the software. In this way, the app could be used by anyone who cannot type or read for any reason.

2.2. Private File Manager App

The task was to present the mathematical background and algorithm for a file manager app that will increase user privacy. The file manager app should be able to hide any files, but in our situation, specifically the images and videos of the Gallery app of an Android smartphone or tablet. The app needs to have a password to be able to be accessed only by the legitimate owner of the smartphone or tablet. The app will hide any user-selected image or video file from the smartphone or tablet. In this way, in the Gallery app on an Android smartphone or tablet, the selected files will be hidden. In the private file manager app, all files can be seen, even if they are not hidden. After entering the app password, the user can choose which files to hide or unhide. In the unpleasant case where the user loses their phone, the finder will not be able to access those files the legitimate user hid. In the case where the user borrows a phone from someone, that person will see only the images and videos that the legitimate user allows anyone to see. The app will try to maintain the legitimate user's privacy as much as possible.

2.3. Novel Torch App Which Can Be Switched by Double-Tapping Anywhere on the Screen

The task was to present the mathematical background and algorithm for a mobile application that can control the flashlight on a mobile device as easily as possible. This is good to use when the user wants to focus on lighting and not the mobile device or app itself. The easiest way to control the flashlight is to touch it anywhere on the screen when it is needed. The current applications require the user to pull down different menus to control the flashlight. This app should work only by double-tapping on its icon or anywhere on the screen to control the flashlight. The app can have a shortcut on the main screen and even a notification on the lock screen to control the flashlight when the mobile device is locked without the need to unlock it.

3. Problem Solving

The research methodology is based on studying the mathematical background of each app and creating each application to be launched on the Google Play Store.

The design approach is user-centered design, so the applications were tested to be built on what users would need to use on their phone for on a daily basis, with a great emphasis on user feedback. Decisions to make changes in apps and implement new functionalities were always made after carefully checking the user comments on the Google Play Store.

The programming language chosen to create the mobile apps was Java, because it has the widest support among all mobile programming languages for the Android operating system.

3.1. Vocal SMS Writing and Reading App for Visually Impaired People

The methodology of creating the voice SMS app is to first study mathematical aspects such as the spectrogram of the audio signal and then implement all the findings in the mobile phone app.

The scientific impact of the vocal SMS application advances the research fields of speech recognition, natural language processing, human–computer interaction, accessibility, cybersecurity, and behavioral linguistics. It also enhances AI-driven communication, fosters digital inclusion, and optimizes data communication. In addition, it helps to advance multilingual technologies.

An app is needed for visually impaired people to use SMS functionalities vocally; a solution would be this vocal SMS app. The creation method and the mathematical background will be presented next.

This Android application creates text when recording, sends it as an SMS message, and reads the received SMS messages using text-to-speech software. The aim is to accelerate the composition of SMS messages, which is helpful when the user is in a hurry or driving a vehicle, or for visually impaired people. The application loads all contacts into a list, but additionally, a contact can be searched (with a speech command as well). When the user long-presses a contact, recording starts immediately. To end the recording, the user should stop speaking or remove the mobile device from near the mouth. If the result is not satisfactory, the recording can be repeated. The recording result will be better if the default language of the mobile device is set to the spoken language. When the message is OK, it can be sent via an SMS message or copied to the clipboard to send it through other messaging applications. The received SMS message can also be read aloud with text-to-speech software, so the user can hear the message rather than read it. When the user wants to answer a received message, they can simply long-press on the contact name from the received SMS pop-up. The application needs an Internet connection. It needs Internet to run the speech-to-text and text-to-speech engine from Google. The implication is that if the user has no Internet access, the app cannot work. This exception was handled, so the app will not crash, but an error message that says that the app cannot work without an Internet connection will pop up.

Algorithm 1 shows the most important steps of the vocal SMS app.

Figure 1 shows the flowchart of the vocal SMS app.

All SMS messages are saved in the SMS database of the mobile device. The application shows only the current SMS, not all old messages. Older SMS messages can be read from the default SMS application on the mobile device. When the application is installed, the application must become the default SMS application on the mobile device by tapping OK on the initial pop-up message. This can be easily changed by tapping on OK in the default SMS application. Everything is stored on the user's mobile device, so the user should not worry about the data being present in any other third-party servers. The goal

of this application is to make SMS writing similar to normal phone talk and to reduce the writing time. The user must make text to speech functional on their mobile device and configure the Google text-to-speech engine in their language of choice.

Algorithm 1 Speech-to-text conversion in vocal SMS app.

```
procedure ConvertSpeechToText(audioInput)
    audioData ← CaptureAudio(audioInput)
    processedAudio ← PreprocessAudio(audioData)
    textOutput ← SpeechRecognition(processedAudio)
    if textOutput is valid then
        Return textOutput
    else
        Return "Error: Could not transcribe speech"
    end if
end procedure
```

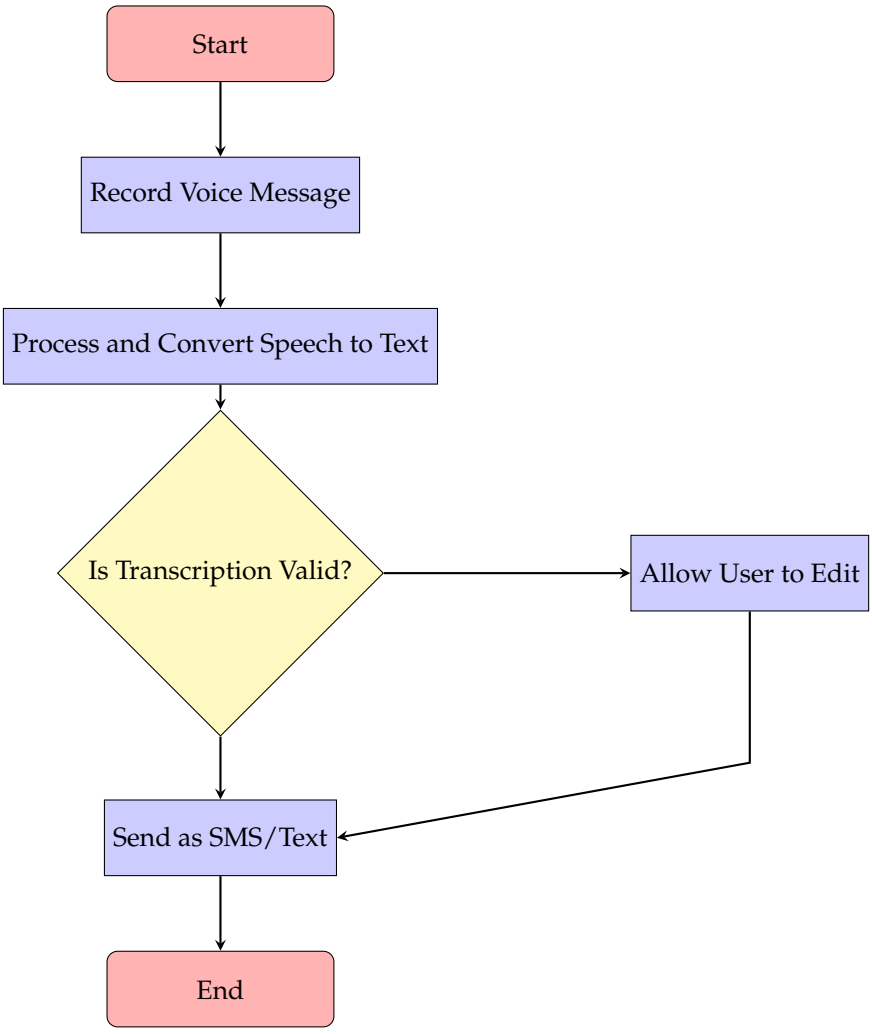


Figure 1. Flowchart of the vocal SMS app.

The sending of voice messages can be represented with the following functions.

$$\text{RecordVoiceMessage}()$$

This function starts the voice message recording process.

$$\text{PlayVoiceMessage}(\text{message})$$

This function plays the voice message that was received.

$$\text{DeleteVoiceMessage}(\text{message})$$

The designated voice message is removed using this function.

$$\text{TranscribeVoiceMessage}(\text{message})$$

This function converts the audio message that must be sent to text.

$$\text{SendTextSMS}(\text{recipient}, \text{text})$$

The text message is sent while using this function. The TTS (text-to-speech) function can be represented as TTS (text), where text is the text input that must be spoken.

$$\text{TTS}(\text{text}) = \text{audio}$$

This function converts text input into audio, which is indicated by audio. Likewise, STT (audio) represents the STT (speech-to-text) function, where audio is the speech input that needs to be translated into text.

$$\text{STT}(\text{audio}) = \text{text}$$

"Text" is the result of this function's conversion of the audio input to text. Integrating SMS messaging with these features is the function shown below.

$$\text{SendSMS_STT}(\text{recipient}, \text{text}) = \text{SendSMS_STT}(\text{recipient}, \text{STT}(\text{audio}))$$

Using STT, this function converts audio to text and sends an SMS to the designated receiver.

Text to speech translates written words into spoken language using various methods. One essential model for this purpose is the Tacotron 2 architecture, which encompasses both sequence-to-sequence models and vocoders.

The Tacotron 2 architecture includes the following steps:

1. **Encoder**: As demonstrated in Equation (1), the encoder converts the character sequence **c** into a series of hidden states **h**.

$$\mathbf{h} = \text{Encoder}(\mathbf{c}) \tag{1}$$

2. **Attention Mechanism**: The attention mechanism matches the hidden states **h** with the decoder states, as illustrated in Equation (2).

$$\mathbf{a}_i = \text{Attention}(\mathbf{h}, \mathbf{s}_{i-1}) \tag{2}$$

3. **Decoder**: The decoder produces Mel spectrogram frames using the attended hidden states as described in Equation (3)

$$\mathbf{m}_i = \text{Decoder}(\mathbf{a}_i, \mathbf{s}_{i-1}) \tag{3}$$

4. **Vocoder (e.g., WaveNet)**: The vocoder transforms the Mel spectrogram frames into the time-domain waveform as depicted in Equation (4).

$$\mathbf{y} = \text{Vocoder}(\mathbf{m}) \tag{4}$$

Converting spoken words into written text is the essence of speech to text. A typical method employs a Recurrent Neural Network (RNN) paired with a Connectionist Temporal Classification (CTC) loss function.

The following steps are involved in the RNN-CTC model:

1. **Feature Extraction**: Derive the acoustic features $\mathbf{x}$ from the audio signal $\mathbf{y}$ as depicted in Equation (5).

$$\mathbf{x} = \text{FeatureExtraction}(\mathbf{y}) \tag{5}$$

2. **Recurrent Neural Network**: Implement an RNN to capture the temporal dependencies present in the features as demonstrated in Equation (6).

$$\mathbf{h}_t = \text{RNN}(\mathbf{x}_t, \mathbf{h}_{t-1}) \tag{6}$$

3. **CTC Loss**: Determine the likelihood of the output sequence $\mathbf{z}$ using the CTC method as illustrated in Equation (7).

$$\mathcal{L}_{\text{CTC}} = -\log P(\mathbf{z}|\mathbf{h}) \tag{7}$$

A model such as Deep Speech streamlines the process by employing a single deep learning model trained from start to finish.

The Deep Speech model includes the following procedures.

1. **Feature Extraction**: Derive characteristics from the audio signal as described by Equation (5).

2. **Deep RNN**: Implement multiple layers of RNNs, including GRUs or LSTMs, as described in Equation (8).

$$\mathbf{h}_t = \text{RNN}_n(\cdots \text{RNN}_2(\text{RNN}_1(\mathbf{x}_t))) \tag{8}$$

3. **Softmax Layer**: Estimate the probability distribution of the characters at each time step as depicted in Equation (9).

$$\mathbf{p}_t = \text{Softmax}(\mathbf{W}\mathbf{h}_t + \mathbf{b}) \tag{9}$$

4. **CTC Loss**: Employ CTC for alignment and training as described in Equation (7).

SMS messages can be encoded with various character sets, including GSM 7-bit, UCS-2 (16-bit), and UTF-8. For GSM 7-bit encoding, the count of septets (7-bit characters) is crucial.

The equation for the number of septets, N_s, derived from a string M consisting of L characters, is presented in Equation (10).

$$N_s = \left\lceil \frac{L \cdot 7}{8} \right\rceil \tag{10}$$

For SMS messages that contain multiple parts, it is necessary to determine the maximum character count per segment and the total number of segments.

When the message length L is more than 160 characters, it must be divided into segments. Each segment can contain a maximum of 153 characters because some characters are used for segmentation details.

Equation (11) provides the value of the number of parts N_p.

$$N_p = \left\lceil \frac{L}{153} \right\rceil \tag{11}$$

Error-correcting codes like Reed–Solomon codes are useful for maintaining message integrity.

In a Reed–Solomon code $RS(n, k)$, with n as the block length and k as the message length, the quantity of errors that can be corrected t is given by Equation (12).

$$t = \left\lfloor \frac{n-k}{2} \right\rfloor \tag{12}$$

Communicating with the SMSC (Short Message Service Center) requires the use of protocols such as SMPP (Short Message Peer-to-Peer) to transmit messages.

The overall length of a message L_{msg}, which encompasses both the header and the body, is represented in Equation (13).

$$L_{msg} = L_{header} + L_{body} \tag{13}$$

where L_{header} represents the length of the SMPP header, and L_{body} indicates the length of the encoded message.

Modulation and demodulation processes are describable utilizing signal-processing techniques.

The modulated signal, denoted as $s(t)$, is shown in Equation (14).

$$s(t) = A \cdot m(t) \cdot \cos(2\pi f_c t + \phi) \tag{14}$$

In which:

- A represents the amplitude;
- $m(t)$ denotes the message signal;
- f_c stands for the carrier frequency;
- ϕ symbolizes the phase.

The function represents the TTS system as follows:

$$y(t) = S(F(x)) \tag{15}$$

Where:

- x serves as the input text.
- $F(x)$ is the feature extraction function responsible for transforming text into linguistic features.
- S is the synthesis function responsible for creating the audio signal $y(t)$ using the features.

The synthesis function S is usually represented as

$$y(t) = \sum_{i=1}^{N} a_i \cdot \sin(2\pi f_i t + \phi_i) \tag{16}$$

Where:

- a_i represents the amplitude for the i-th sine wave.
- f_i denotes the frequency.
- ϕ_i denotes the phase offset.

- N represents the count of sine waves used in the synthesis process.

Figure 2 shows the sine wave components in TTS.

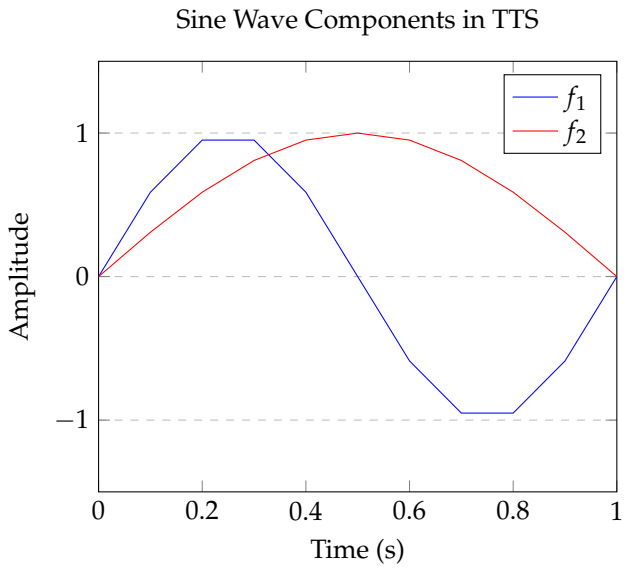


Figure 2. Sine wave components in TTS.

Figure 3 shows the spectrogram of audio signal.

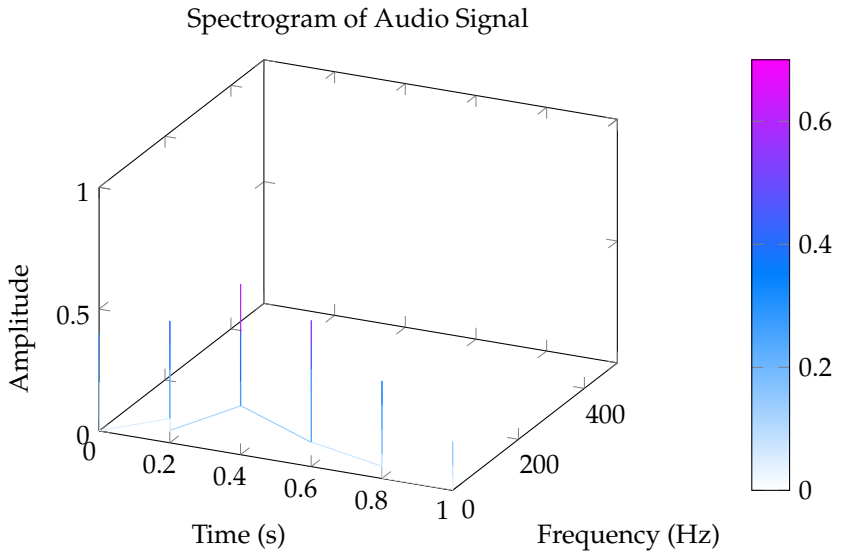


Figure 3. Spectrogram of audio signal.

The STT system can be characterized as follows:

$$x = G(y(t)) \tag{17}$$

Where:

- $y(t)$ represents the audio signal input.
- G is the function responsible for translating the audio signal into text.

The recognition function, denoted by G, operates through multiple layers:

$$G(y(t)) = \text{Dec}(\text{Enc}(y(t))) \tag{18}$$

Where:

- Enc is the function responsible for extracting features from the audio signal.
- Dec is the function responsible for converting features into text.

A vocal SMS application enables users to both send and receive text messages through voice commands. The functionality of this system depends on different mathematical models that address signal processing, energy usage, and data transfer. To represent these critical components of the app, we will employ techniques from differential equations, linear algebra, analytic geometry, and mathematical analysis.

An auditory signal can be described as a time-dependent function $x(t)$, with t representing time and $x(t)$ indicating the signal's amplitude at that moment. This audio signal undergoes sampling at specific time points $t_n = n\Delta t$, where Δt denotes the interval between samples. The sampled values of the signal are represented by $x_n = x(t_n)$.

The Fourier transform is employed to change a signal from the time domain to the frequency domain, a process crucial for compression and transmission. The Fourier transform of the function $x(t)$ is expressed as

$$\hat{x}(f) = \int_{-\infty}^{\infty} x(t) e^{-2\pi i f t}\, dt \tag{19}$$

The discrete Fourier transform (DFT) is employed to analyze the discrete signal:

$$\hat{x}_k = \sum_{n=0}^{N-1} x_n e^{-2\pi i \frac{kn}{N}}, \quad k = 0, 1, \ldots, N-1 \tag{20}$$

To reconstruct the signal, the inverse Fourier transform is applied:

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} \hat{x}_k e^{2\pi i \frac{kn}{N}} \tag{21}$$

By transitioning the signal to the frequency domain, we can achieve compression by discarding minor Fourier coefficients, which are associated with less significant frequencies. Let $\mathbf{X} = [\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_{N-1}]^T$ denote the vector of Fourier coefficients.

We can achieve signal compression by using a projection matrix $\mathbf{P}$ to eliminate minor coefficients:

$$\mathbf{Y} = \mathbf{PX} \tag{22}$$

Let $\mathbf{Y}$ denote the compressed signal, while $\mathbf{P}$ represents a diagonal matrix assigning 0 to small coefficients and 1 to large ones. The compression ratio is calculated as follows:

$$\text{Compression Ratio} = \frac{\text{Number of Non-Zero Coefficients}}{N} \tag{23}$$

Let C represent the battery capacity of the phone, measured in mAh, while $P(t)$ denotes the power consumption of the vocal SMS application at a given time t, expressed in watts. This power consumption depends on the intensity of audio processing and SMS transmission.

The rate at which battery discharges can be represented by this differential equation:

$$\frac{dC(t)}{dt} = -\frac{P(t)}{V} \tag{24}$$

Given that V represents the constant voltage of the phone battery, resolving this equation provides the battery level at any time point t:

$$C(t) = C_0 - \int_0^t \frac{P(\tau)}{V}\, d\tau \tag{25}$$

When the app consistently uses power P_0, the battery level diminishes linearly:

$$C(t) = C_0 - \frac{P_0}{V}t \tag{26}$$

Here, C_0 denotes the original battery capacity.

Analytic geometry offers a framework for modeling signal transmission. Imagine a phone sending a voice signal across a span d. The signal power diminishes following the inverse-square law:

$$P_{\text{received}}(d) = \frac{P_{\text{transmitted}}}{d^2} \tag{27}$$

where $P_{\text{transmitted}}$ denotes the signal power in the phone speaker, and $P_{\text{received}}(d)$ represents the power received at a distance d.

To enhance signal quality, we reduce attenuation by selecting the best transmission angle. When the phone antenna emits the signal at an angle of θ to the horizontal, it arrives at the receiver over a distance $d(\theta)$, represented by

$$d(\theta) = \frac{h}{\sin(\theta)} \tag{28}$$

Given that h represents the height of the phone with respect to the receiver, the power received can be expressed as a function of θ as follows:

$$P_{\text{received}}(\theta) = \frac{P_{\text{transmitted}} \sin^2(\theta)}{h^2} \tag{29}$$

In order to enhance the performance of the vocal SMS application, our objective is to reduce energy usage $E(t)$ while preserving the quality of the signal $Q(t)$. This leads to the formulation of the optimization problem as follows:

$$\min \int_0^T P(t)\,dt \quad \text{subject to} \quad Q(t) \geq Q_{\min}\ \forall t \in [0, T] \tag{30}$$

Employing Lagrange multipliers, we establish the Lagrangian as follows:

$$\mathcal{L}(P, \lambda) = \int_0^T P(t)\,dt + \lambda(Q_{\min} - Q(t)) \tag{31}$$

By varying $\mathcal{L}$ concerning both $P(t)$ and λ, the resulting equations are

$$\frac{\delta \mathcal{L}}{\delta P(t)} = 0, \quad \frac{\delta \mathcal{L}}{\delta \lambda} = 0 \tag{32}$$

Solving these equations results in determining the best power distribution $P(t)$ along with the associated signal quality $Q(t)$.

A detailed mathematical model for a mobile phone vocal SMS application was formulated in this section. The approach involved employing techniques from differential equations, linear algebra, analytic geometry, and mathematical analysis to represent signal processing, battery usage, signal transmission, and system optimization. These models aim to boost the app's efficiency and enrich the user experience.

In Figure 4, the initial screen of the mobile app can be seen. It has a search box to search for a name in the mobile phone's address book. It also has an 'X' button to clear an error in the search box. The application loads the names from the mobile phone address book. As the hint says, the user must long-press any contact in the address book or from

the search box, and the recording will begin. After the recording is finished, the speech is transformed into text that can be sent as an SMS message.



Figure 4. Loading address book in the vocal SMS app.

In Figure 5, it can be seen how the search box functions. The search box has a drop-down list with search suggestions and an autocomplete function to ease the searching. After the second character is written, the search drop-down list appears; this way, the user can select the most appropriate user to send the message to. The users can also be searched vocally; in this way, it is even easier for visually impaired people, car drivers, or children who do not know how to write to use the app.

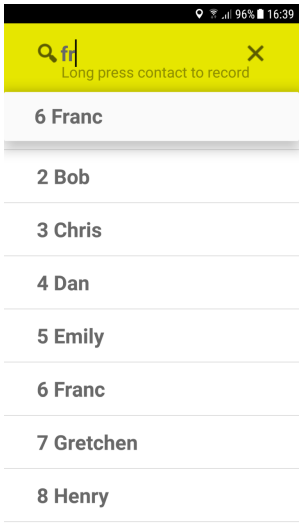


Figure 5. Searching in address book with the vocal SMS app.

In Figure 6, recording of the user's speech is shown. There is some indication of what to take into account to make a good recording. In the first line, the name to whom the message will be sent can be seen. The user does not need to press any microphone button, because the recording starts when the user speaks and stops automatically when the user stops speaking. The green button must be pressed when indicated by the software when the user makes an unclear recording. There is also an indication that the audio file will be sent to Google to provide a speech recognition service. The speech recognition service will transform the speech into text that can be sent as an SMS message. If the recording is not good, it can be repeated as many times as necessary. For the system to work correctly,

the best practice is to keep the mobile phone close to the mouth when speaking and to remove the device after speaking ends.

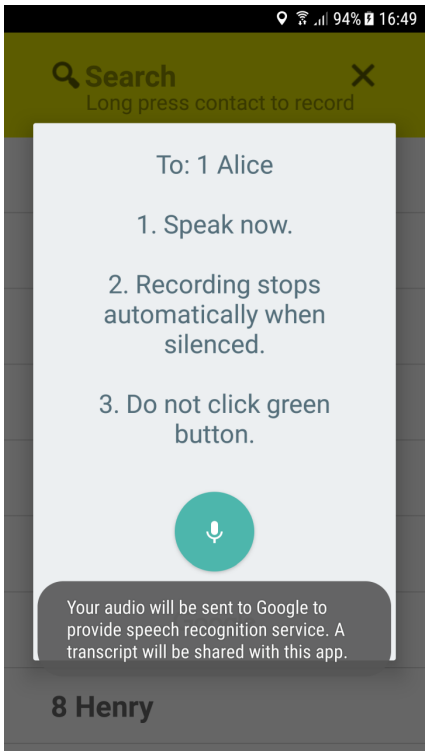


Figure 6. Recording a message with the vocal SMS app.

In Figure 7, the pop-up that shows after the recording ends can be seen. The person who receives the message and the recorded message, in our case "hello", can be seen. After this, the user can decide whether or not to send the message. When tapping on the blue arrow, the message will be sent as an SMS message. If the user clicks on the "X" button to close, the pop-up closes and the message will not be sent. The user can then make a new recording if they choose to. As the hint says, all incoming and outgoing messages can be viewed in the default SMS app. The app uses the database from the default SMS application and stores all incoming and outgoing messages in that database. Of course, longer messages can also be recorded. The recording will work better if the user sets the default language of the phone to the language that they speak when making a recording with this app. The default language can be easily set in the Settings app on the Android device.

In Figure 8, the pop-up that appears when an SMS message is received can be seen. It also shows a notification in the notification area of the mobile device, but also a pop-up with the message. When the notification is clicked, the system takes the user to the voice SMS app. In the pop-up, the name of the user who sent the message and, of course, the text of the message itself can be seen.

If the user clicks on the green arrow, the system will read the text of the message aloud using text-to-speech software. Reading the text message can be stopped by clicking the green arrow again. The green arrow has the functions "Play" and "Pause". This function can be helpful for visually impaired people, drivers, and even children who have not yet learned to read. As the hint says, the user can respond to the received SMS message simply by pressing the name of the pop-up that shows the received SMS message. This received SMS message is also stored in the database of the default SMS app and can also be read from there if needed. The app can also be used without voice support, but with this, it loses all its fun.

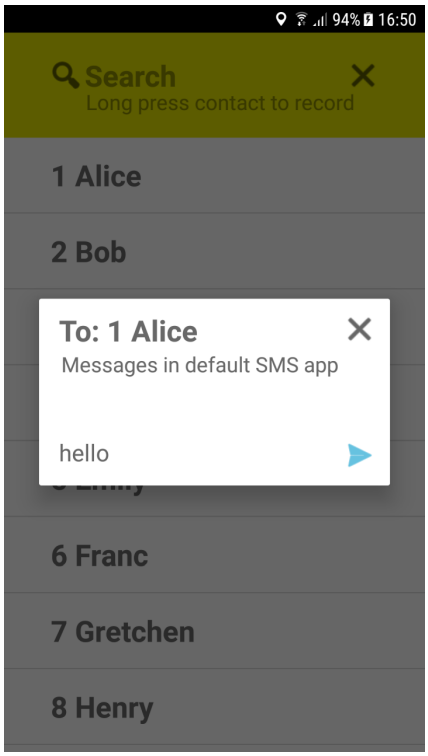


Figure 7. Sending an SMS message with the app.

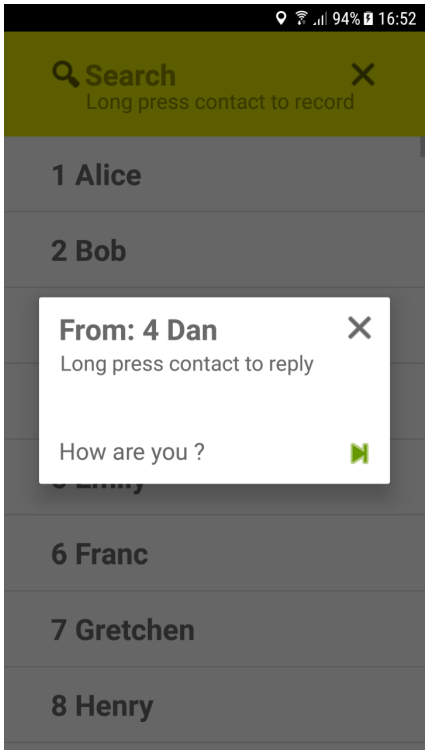


Figure 8. Receiving and reading an SMS message aloud with the app.

3.2. Private File Manager App

The methodology of the private file manager app is to first study mathematical aspects such as the storage usage of different files and then implement all the findings in the mobile phone app.

The scientific impact of the private file manager app is that it advances studies in cybersecurity, encryption, AI-driven file categorization, and human–computer interaction,

enhancing data privacy, secure access methods, and energy-efficient digital management while complying with global data protection regulations.

Algorithm 2 shows the most important steps of the private file manager app.

Algorithm 2 Hide file in private file manager.

```
procedure HIDEFILE(filePath)
    if FileExists(filePath) then
        hiddenPath ← AppendHiddenAttribute(filePath)
        MoveFile(filePath, hiddenPath)
        Return "File Hidden Successfully"
    else
        Return "Error: File Not Found"
    end if
end procedure
```

Figure 9 shows the flowchart of the private file manager app.

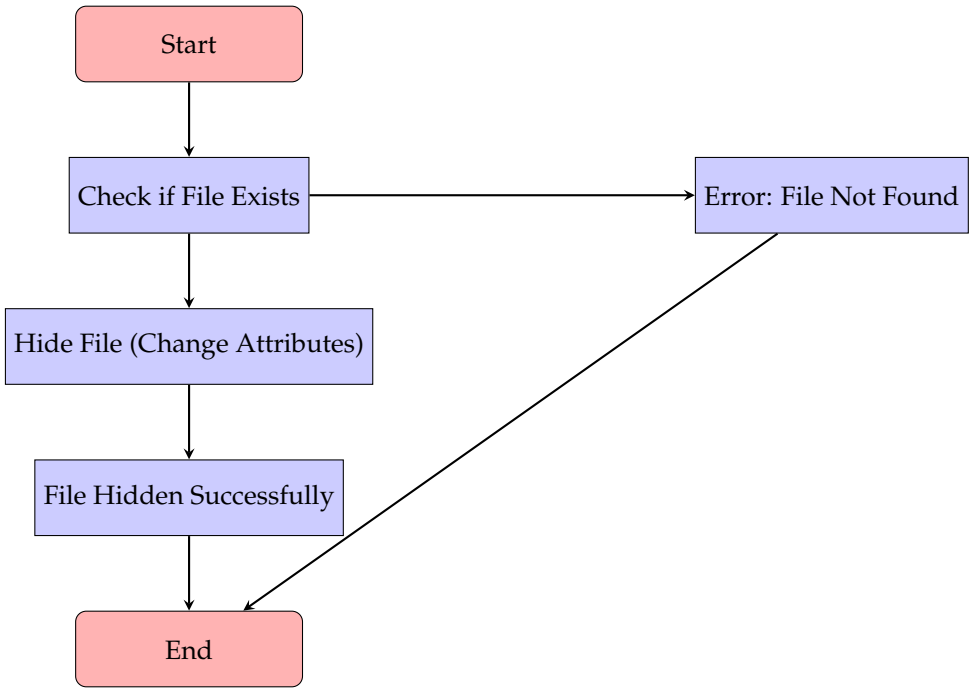


Figure 9. Flowchart of the private file manager app.

There is a need for an app for hiding personal files, mostly photos and videos; a solution would be this private file manager app. The creation method and the mathematical background will be presented next.

The initial task was to create a file manager for Android devices. Today, the task can be performed quite well due to the capabilities of the Android operating system. As Android is based on Linux, it can use the Linux shell commands to work with the file system (create, delete, rename, copy, move folders/files). If we think back on the olden days to a similar application, called DOS (Disk Operating System), Microsoft made a multibillion-dollar corporation. The funny thing is that the application did not even have a graphical user interface at that time.

For the file manager app, we used the following formulas.

The file existence check is presented in Equation (33).

$$\text{Exists}(f) = \begin{cases} 1, & \text{if file } f \text{ exists} \\ 0, & \text{otherwise} \end{cases} \tag{33}$$

If the file f exists, this function returns 1, and if not, it returns 0.
File size calculation is presented in Equation (34).

$$\text{FileSize}(f) = \text{size of file } f \tag{34}$$

The size of the file is f.
File copying is presented in Equation (35).

$$\text{CopyFile}(f, g) = \text{Copy file } f \text{ to } g \tag{35}$$

The contents of file f are copied to file g by this function.
File deletion is presented in Equation (36).

$$\text{DeleteFile}(f) = \text{Delete file } f \tag{36}$$

The file f is deleted by this function.
File renaming is presented in Equation (37).

$$\text{RenameFile}(f, g) = \text{Rename file } f \text{ to } g \tag{37}$$

File f is renamed to file g using this method.
File moving is presented in Equation (38).

$$\text{MoveFile}(f, \text{directory}) = \text{Move file } f \text{ to directory} \tag{38}$$

File f is moved to the designated directory using this function.
File listing is presented in Equation (39).

$$\text{ListFiles}(\text{directory}) = \{f_1, f_2, \ldots, f_n\} \tag{39}$$

A list of files in the given directory is the return value of the function.
File creation is presented in Equation (40).

$$\text{CreateFile}(f) = \text{Create file } f \tag{40}$$

The new file f is created by this function.

Effective file indexing is essential for quick access and retrieval. A common method is to use an inverted index.

Given a collection of documents $D = \{d_1, d_2, \ldots, d_N\}$ and a vocabulary $V = \{t_1, t_2, \ldots, t_M\}$, the inverted index I associates each term t_i with a list of documents D_{t_i} that contain the term, as illustrated in Equation (41).

$$I(t_i) = \{d_j \mid t_i \in d_j\} \tag{41}$$

Effective file searches can be achieved with algorithms such as binary search trees (BSTs) or B-trees.

A B-tree of order m is a type of tree structure where each node can have up to m children, making it ideal for file indexing and searching.

Consider a B-tree node x containing n keys; the properties listed below are satisfied:

1. $n \leq 2m - 1$.
2. If x is an internal node that is not the root, it must have a minimum of $\lceil m/2 \rceil$ children.

Securing data involves encrypting files. One popular encryption technique is the Advanced Encryption Standard (AES). The AES encryption was implemented in the file manager app.

AES encryption comprises several transformation rounds. Each round r of AES-128 encryption includes the following:

1. SubBytes: A substitution step with non-linear properties.
2. ShiftRows: A permutation of rows.
3. MixColumns: A mixing operation performed on the columns.
4. AddRoundKey: An XOR operation using the round key.

In round r, the state matrix S is transformed by S_r, as demonstrated in Equation (42).

$$S_r = \text{AddRoundKey}(\text{MixColumns}(\text{ShiftRows}(\text{SubBytes}(S)))) \tag{42}$$

File compression minimizes file size for optimal storage. Huffman coding is a widely used algorithm for lossless file compression.

Huffman encoding allocates codes of varying lengths to symbols according to their occurrence rates.

Consider a set of characters $C = \{c_1, c_2, \ldots, c_n\}$ with associated frequencies $f(c_i)$. The expected length $E(L)$ of the encoded message is depicted in Equation (43).

$$E(L) = \sum_{i=1}^{n} P(c_i) \cdot l(c_i) \tag{43}$$

$P(c_i) = \frac{f(c_i)}{\sum_{j=1}^{n} f(c_j)}$, where $l(c_i)$ denotes the length of the code for c_i.

Synchronizing files guarantees consistency between different devices. The rsync algorithm is a popular technique for this purpose.

The rsync algorithm employs a rolling checksum to identify differences between files. The rolling checksum, denoted as R, for a block of size b beginning at position i, is illustrated in Equation (44).

$$R_i = \sum_{j=0}^{b-1} (d_j \cdot 2^{b-j}) \tag{44}$$

To detect changes, the hash of this block is compared later with the hashes of blocks in the target file.

Let the total storage capacity be denoted by S, and let s_i represent the size of the i-th file. The total storage utilized, U, is defined as:

$$U = \sum_{i=1}^{n} s_i \tag{45}$$

Where:

- n indicates the complete count of files.

The proportion of storage in use, denoted as P, is determined by:

$$P = \frac{U}{S} \times 100\% \tag{46}$$

Figure 10 shows the storage usage over time.

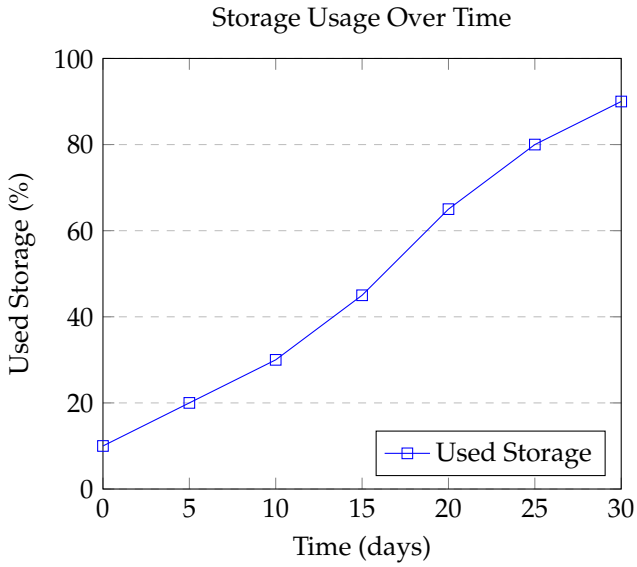


Figure 10. Storage usage over time.

The rate at which a file of size s is transferred over a duration t is represented by R:

$$R = \frac{s}{t} \tag{47}$$

Where:

- s represents the size of the file in megabytes (MB).
- t refers to the duration of the transfer measured in seconds (s).

Figure 11 shows the file transfer rates.

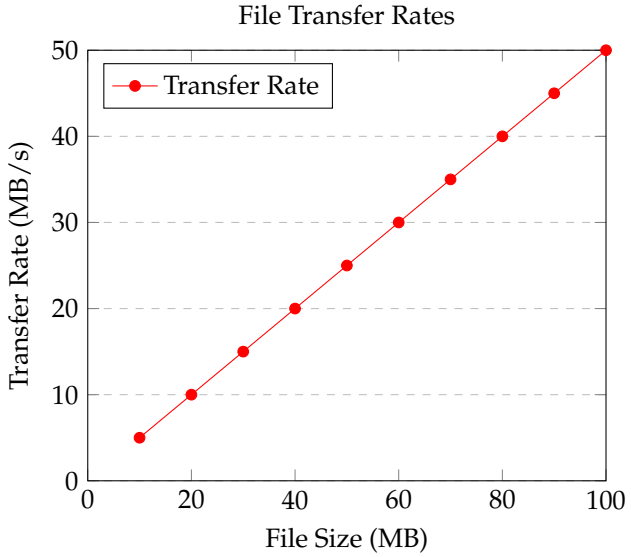


Figure 11. File transfer rates.

The effectiveness of arranging files can be measured by the mean number of files per directory E, calculated as:

$$E = \frac{n}{f} \tag{48}$$

Where:

- n represents the complete count of files.
- f denotes the aggregate number of directories.

A mobile file manager equipped with a concealment feature allows users to handle their files while ensuring that certain ones remain hidden for privacy purposes. This app incorporates encryption, file storage structuring, energy consumption regulation, and enhanced access speed. This paper presents a mathematical model that uses techniques from differential equations, linear algebra, analytic geometry, and mathematical analysis.

Figure 12 shows the file organization efficiency.

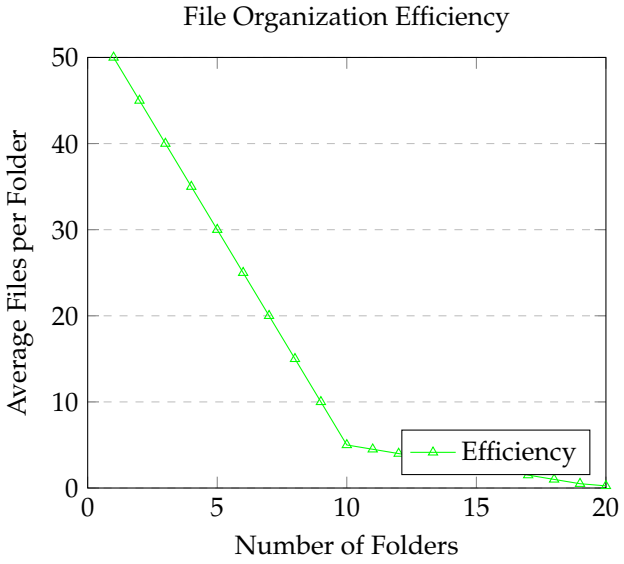


Figure 12. File organization efficiency.

The process of hiding data generally involves encrypting files. Consider $\mathbf{F} \in \mathbb{R}^n$, which denotes the file content as a vector of dimension n (e.g., a vector format of the file's binary data). The encryption can be represented by a matrix transformation $\mathbf{E}$, where $\mathbf{E}$ is an encryption matrix of size $n \times n$. The resulting encrypted file $\mathbf{F}_{\text{encrypted}}$ is:

$$\mathbf{F}_{\text{encrypted}} = \mathbf{EF} \tag{49}$$

To conceal the file, we utilize a function $\mathbf{H}$ applied to the encrypted data, where $\mathbf{H}$ serves to relocate or encode the encrypted file securely. The concealed file $\mathbf{F}_{\text{hidden}}$ is:

$$\mathbf{F}_{\text{hidden}} = \mathbf{H}(\mathbf{F}_{\text{encrypted}}) \tag{50}$$

We retrieve the concealed file by executing the reverse operations of the encryption and hiding functions:

$$\mathbf{F} = \mathbf{E}^{-1}\mathbf{H}^{-1}(\mathbf{F}_{\text{hidden}}) \tag{51}$$

A file manager organizes files within a confined storage area, conceptualized as a three-dimensional geometric space. This storage area is depicted as a bounded region $\Omega \subset \mathbb{R}^3$, where files reside at specific coordinates $(x, y, z) \in \Omega$. The position of a hidden file is indicated by the coordinates (x_h, y_h, z_h).

$$d = \sqrt{x_h^2 + y_h^2 + z_h^2} \tag{52}$$

Retrieving the file requires an amount of time that is proportional to this distance; therefore, the access time T_{access} is represented by:

$$T_{\text{access}}(x_h, y_h, z_h) = \alpha d = \alpha \sqrt{x_h^2 + y_h^2 + z_h^2} \tag{53}$$

A proportionality constant, denoted by α, is determined by the efficiency of the search algorithm employed by the file manager.

The impact of encryption and hiding activities on file manager battery usage is significant. Let $C(t)$ denote the battery capacity of the mobile device at time t, with $P_{\text{enc}}(t)$ and $P_{\text{hide}}(t)$ indicating the power consumption for encryption and hiding tasks, respectively.

The rate of battery depletion is modeled by the following differential equation:

$$\frac{dC(t)}{dt} = -\frac{P_{\text{enc}}(t) + P_{\text{hide}}(t)}{V} \tag{54}$$

In this equation, the battery level can be determined at any time t by integrating, where V is the constant voltage of the phone battery.

$$C(t) = C_0 - \int_0^t \frac{P_{\text{enc}}(\tau) + P_{\text{hide}}(\tau)}{V}\, d\tau \tag{55}$$

Assuming each operation uses fixed power, specifically $P_{\text{enc}} = P_{\text{enc},0}$ and $P_{\text{hide}} = P_{\text{hide},0}$, the level of the battery decreases linearly:

$$C(t) = C_0 - \frac{(P_{\text{enc},0} + P_{\text{hide},0})t}{V} \tag{56}$$

In order to enhance system performance, our objective is to reduce cumulative access time while ensuring minimal battery usage. Let $T_{\text{total}}(t)$ represent the total time required to access hidden files by time t, and let $C(t)$ denote the battery level at that time. The optimization challenge is defined as follows:

$$\min_{x_h, y_h, z_h} T_{\text{total}}(t) = \sum_{i=1}^{N} T_{\text{access}}(x_h^i, y_h^i, z_h^i) \tag{57}$$

$$\text{subject to} \quad C(t) \geq C_{\min}, \quad \forall t \in [0, T] \tag{58}$$

In this context, $C_{\min}$ denotes the lowest necessary battery level for the system's functionality, while T_{total} represents the complete duration needed to retrieve N files.

The solution to this optimization was achieved using the Lagrange multiplier method. Consider the Lagrangian:

$$\mathcal{L}(x_h, y_h, z_h, \lambda) = T_{\text{total}}(t) + \lambda(C_{\min} - C(t)) \tag{59}$$

By differentiating with respect to x_h, y_h, z_h, and λ, we derive the following set of equations:

$$\frac{\partial \mathcal{L}}{\partial x_h^i} = 0, \quad \frac{\partial \mathcal{L}}{\partial y_h^i} = 0, \quad \frac{\partial \mathcal{L}}{\partial z_h^i} = 0, \quad \frac{\partial \mathcal{L}}{\partial \lambda} = 0 \tag{60}$$

Solving these equations yields the ideal coordinates (x_h^i, y_h^i, z_h^i) for each file, thereby reducing the total access time and maintaining battery efficiency.

Besides encrypting files, the file manager can also compress hidden files to optimize storage usage. Define $\mathbf{F}_{\text{hidden}}$ as an n-dimensional vector representing the hidden

file. The compression process can be described by a matrix transformation $\mathbf{C} \in \mathbb{R}^{m \times n}$, with $m < n$, transforming the file into its compressed form $\mathbf{F}_{\text{compressed}}$:

$$\mathbf{F}_{\text{compressed}} = \mathbf{C}\mathbf{F}_{\text{hidden}} \tag{61}$$

The process of decompression is carried out utilizing the pseudoinverse of the compression matrix, denoted as $\mathbf{C}^{+}$:

$$\mathbf{F}_{\text{hidden}} = \mathbf{C}^{+}\mathbf{F}_{\text{compressed}} \tag{62}$$

A detailed mathematical model for a mobile phone file manager with a file-hiding feature was introduced. This model uses linear algebra for encryption and hiding, employs analytic geometry for file access timing and storage layout, applies differential equations to gauge battery usage, and utilizes mathematical analysis for system optimization. These models shed light on how the file manager can effectively manage hidden files while reducing both power consumption and access time.

In Figure 13, the initial screen of the application, the file manager, can be seen. As can be seen, the app was made as simple as possible, and the types of files and folders that the file manager can access are at the top: SD card, images, audio, videos, documents, downloads, and DCIM (photos). The sub-folders and files that are in the selected folders are shown below. All files whose names begin with a dot (".") are hidden. At the top, there is a back button for navigation, the actions that can be selected, the hidden/unhidden files of folders, and the info button, which shows information/help messages about the app. Initially, when the app is installed, the user must allow the app to access the directories and files of the smartphone or tablet. The user can check the directories and files of the smartphone or tablet after allowing the app to access the internal and/or external memory of the device. The app also needs to have some protection in order not to access some sensitive files, located mainly in the root folder. It is not good to hide a file that is used by the operating system in a critical operation for other purposes. It is not very easy to create an app that can show all hidden files, because the Android operating system is designed in such a way that certain sensitive files are not accessible or cannot be seen (hidden). With some tweaks, it can be changed so that the application shows all files, hidden or not, but does not access the root file system or the folders where the sensitive files of the Android operating system are stored. The app was made to hide sensitive files for the user, not for the operating system. The sensitive files for the user are images and videos, but the app can also hide other folders/files. The author found a limitation in the Android operating system file manager and implemented it in this app. The limitation was that the Android operating system does not really have a good hide/show function. This may have been implemented for safety reasons, to ensure that standard users do not wreck something in the operating system of their Android device. The operating system does not really have a function to hide/show files. The method of hiding/showing files can be carried out with external applications by creating a file ".nomedia" in a folder. This method will hide all folders/files from the folder in which it is placed, but not each individual folder/file selected by the user. If we see it in another way, users of the Android operating system need an app that can make their sensitive files (images, videos) more private. The way to do this is to hide sensitive user files.

In Figure 14, the pop-up shown when clicking on the "Select Action" button can be seen. From the pop-up, the user can choose if they want to Hide" or "Show" a certain selected file. The files can be selected/deselected by long-pressing them. The actions there are only "hide" and "show" actions for safety purposes. The app is not made to destroy the user's operating system; it is made just to compensate for a functionality that the Android

operating system file manager does not have. This could also be implemented, but was not, for safety purposes; this way, the user will not accidentally take some unwanted action. All actions (create, delete, rename, copy, move folders/files) can be performed in the file manager of the Android operating system, so it is not necessary to implement them in this app as well. This app has implemented only hide/show actions, which are not present in the Android operating system file manager, so it can be used to hide/show images/videos or any other folders/files of the user.

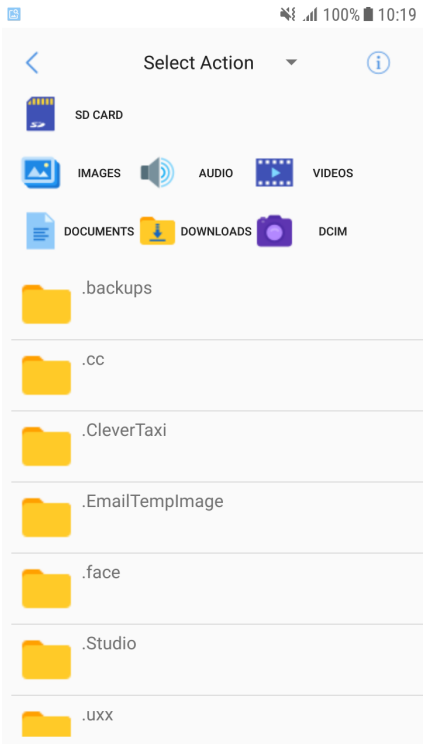


Figure 13. Initial screen of the file manager application for the Android operating system.

In Figure 15, it can be seen that the “SimpleScanner” directory was hidden after selecting the "Hide" operation of the app. After this, the “SimpleScanner” directory will not be visible in the Gallery app on the Android smartphone or tablet. The app automatically refreshes the folder contents, so all files are alphabetically sorted after each change. This is why the “.SimpleScanner” directory became the first, because its name starts with a dot (“.”), which is before the other letters in the ASCII system, and alphabetical sorting is based on the ASCII table. The other actions (create, delete, rename, copy, move folders/files) can be performed in the file manager of the Android operating system.

Actually, the hide/show action is a rename action, but not all users know that all files beginning with a dot (“.”) are hidden in Linux and, in this case, in Android. This app tries to make use of this functionality easy for the average user.

In Figure 16, it can be seen that the pop-up with the information/help message is displayed when the user taps the information button. It is explained that the user has to long-press a folder/file to select it and long-press again to deselect it. All folders/files whose names begin with a dot (“.”) are hidden. The app actually hides or shows folders/files in the smartphones or tablet’s Gallery app; this way, in this app, all folders/files are visible, even if they are hidden. The app is suggested to be used for images or videos of the user in the Gallery app of the Android operating system. Sometimes, it is necessary to turn off the phone after turning it on, not rebooting, to see changes in the Gallery app of the Android operating system. Sometimes, the refreshing of the files requires turning the phone off and then on.

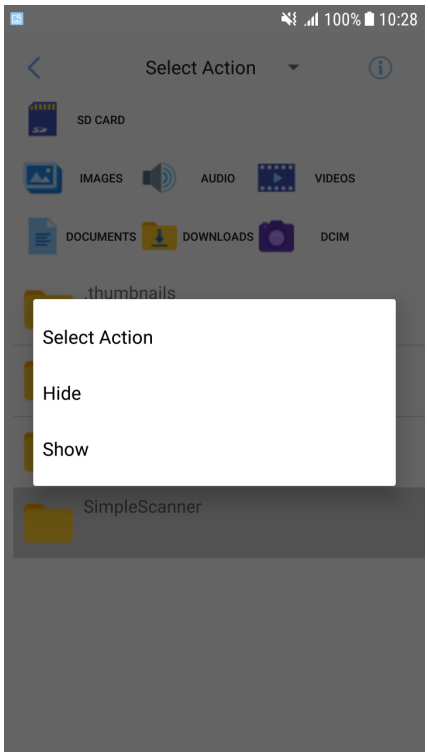


Figure 14. Selecting the action to hide/show folders/files.

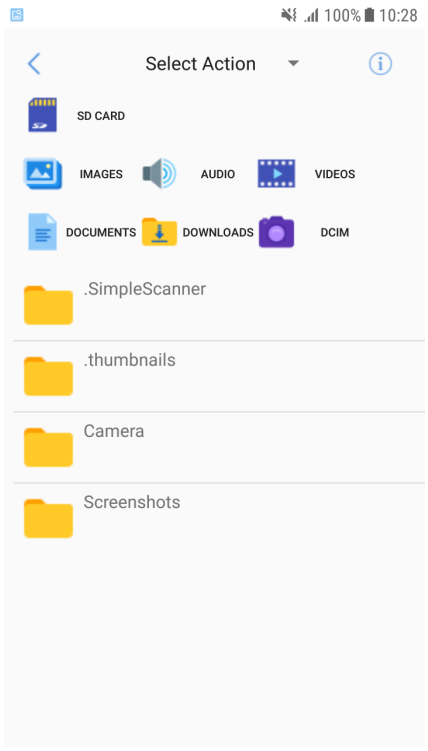


Figure 15. The "SimpleScanner" directory is hidden after selecting the "Hide" operation in the app.

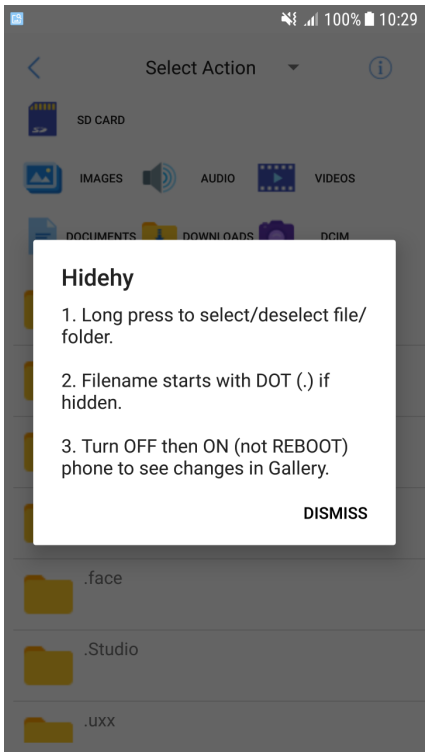


Figure 16. The pop-up with the information/help message shown after tapping the information button.

3.3. Novel Torch App Which Can Be Switched by Double-Tapping Anywhere on the Screen

The methodology of the novel torch app is to first study mathematical aspects such as the power consumption of the mobile phone, while the torch LED is turned on, and then implement all the findings in the mobile phone app.

The scientific impact of the novel torch app is that it improves research in fields such as energy-efficient mobile technology, adaptive brightness management, human–computer interaction, and technologies for emergency response. Moreover, it contributes to progress in optimizing low-power LEDs and improving accessibility features.

Figure 17 shows the flowchart of the novel torch app.

Algorithm 3 shows the most important steps of the novel torch app.

Algorithm 3 Toggle flashlight in novel torch app.

```
procedure ToggleFlashlight
    if FlashlightStatus = OFF then
        TurnOnFlashlight()
        FlashlightStatus ← ON
        Return "Flashlight Turned ON"
    else
        TurnOffFlashlight()
        FlashlightStatus ← OFF
        Return "Flashlight Turned OFF"
    end if
end procedure
```

There is a need for a torch app that can be toggled really easily by tapping anywhere on the phone screen; a solution would be this novel torch app. The creation method and mathematical background will be presented next.

With this application, the user can turn the flashlight of the mobile device on/off by double-tapping anywhere on the screen. The problem of the app starting accidentally was solved by adding a double-tap to turn the flashlight on and a double-tap to turn

the flashlight off. The application was designed to be extremely simple and easy to use. The application also has a notification that turns the flashlight on/off by tapping on the notification. To make the application usable, the user must allow camera access for the application, as the application uses the flashlight near the camera of the mobile device.

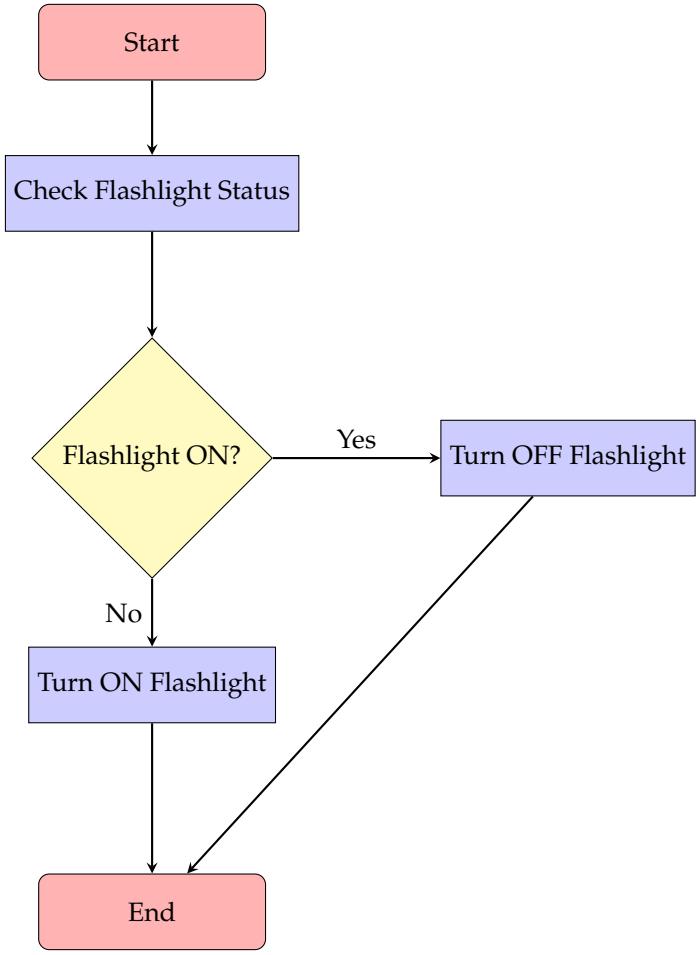


Figure 17. Flowchart of the novel torch app.

Next, the implementation of the application will be shown.

Equation (63) shows a mathematical expression of the flashlight toggling app.

$$\text{Flashlight}(\text{state}) = \begin{cases} \text{ON}, & \text{if state} = \text{TRUE} \\ \text{OFF}, & \text{if state} = \text{FALSE} \end{cases} \tag{63}$$

The flashlight function is represented by a flashlight. The current state of the flashlight is represented by the TRUE or FALSE state. If the state is TRUE, the function will return ON, showing that the flashlight is turned ON, and OFF if the state is FALSE, showing that the flashlight is turned OFF.

The illumination E (measured in lux) at a distance d from the flashlight is determined by Equation (64).

$$E = \frac{I}{d^2} \tag{64}$$

In this context, I represents the luminous intensity in candelas (cd).

Battery life t can be determined using Equation (65).

$$t = \frac{C \times \eta}{P} \tag{65}$$

Here:

- C represents the battery capacity in milliampere-hours (mAh).
- η denotes the LED driver's efficiency (dimensionless, usually ranging from 0 to 1).
- P corresponds to the power usage in watts (W).

The thermal resistance, denoted as R_θ and measured in °C/W, can be determined using Equation (66).

$$R_\theta = \frac{T_j - T_a}{P} \tag{66}$$

Where:

- T_j is the junction temperature of the LED (°C).
- T_a is the ambient temperature (°C).
- P is the power dissipation in watts (W).

As demonstrated in Equation (67), the total luminous flux Φ (measured in lumens) can be determined using the luminous efficacy η_l (measured in lumens per watt) and the power consumption P (measured in watts).

$$\Phi = \eta_l \times P \tag{67}$$

The flashlight app's power usage is described by

$$P = V \times I \tag{68}$$

Where:

- V represents the voltage level of the cell phone battery.
- I is the amount of current consumed by the LED.

The duration for which the flashlight can be powered by the battery is

$$T = \frac{C}{I} \tag{69}$$

Where:

- C represents the battery capacity measured in ampere hours (Ah).

Luminous efficacy (LE), which denotes the LED's ability to transform electrical energy into light, is determined by

$$LE = \frac{L}{P} \tag{70}$$

Where:

- L represents the luminous flux measured in lumens.

Figure 18 shows the battery life vs. current.

The intensity of brightness (B) can be regulated by altering the current:

$$B = k \times I \tag{71}$$

Where:

- k represents a constant of proportionality that depends on the properties of the LED.

Figure 19 shows the brightness vs. current.

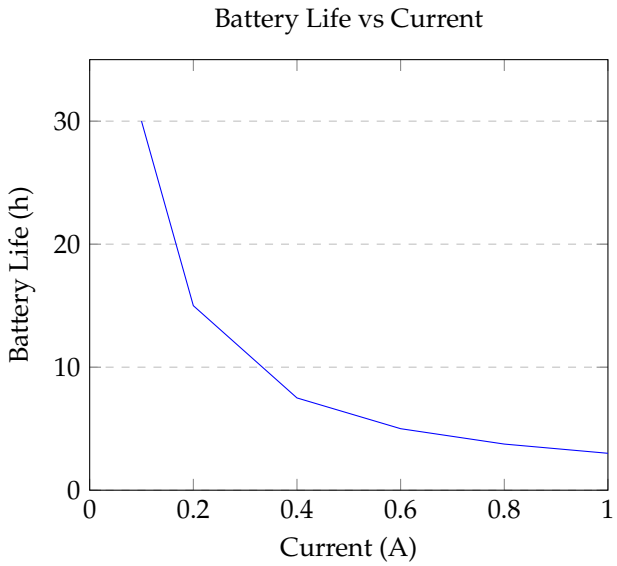


Figure 18. Battery life vs. current.

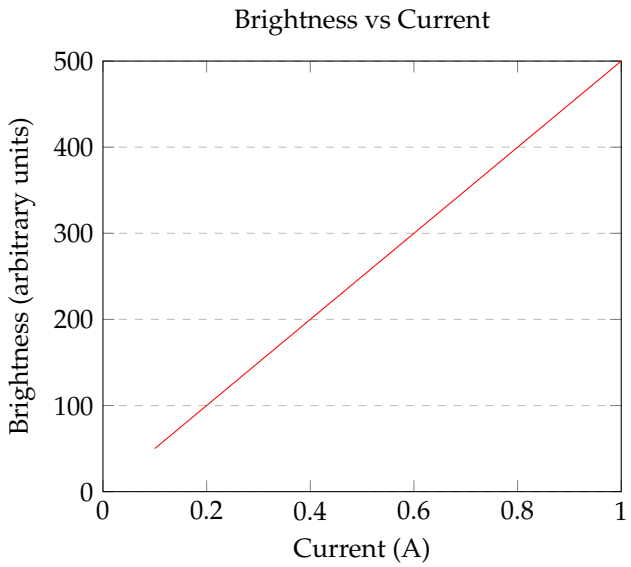


Figure 19. Brightness vs. current.

The amount of heat produced by the LED can be determined using

$$H = P - L \times E \tag{72}$$

Where:

- E represents the efficiency of energy conversion.

Figure 20 shows the heat vs. current.

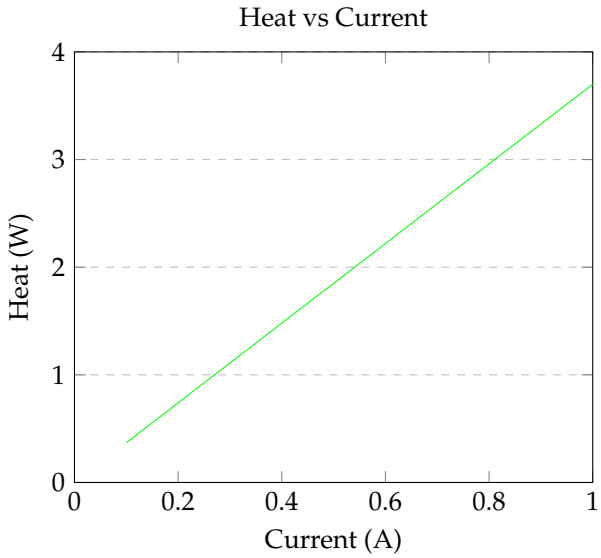


Figure 20. Heat vs. current.

The rise in temperature, denoted as ΔT, can be expressed as

$$\Delta T = \frac{H}{C_{\text{thermal}}} \tag{73}$$

Where:

- C_{thermal} denotes the thermal capacity of the LED.

Figure 21 shows the temperature increase vs. current.

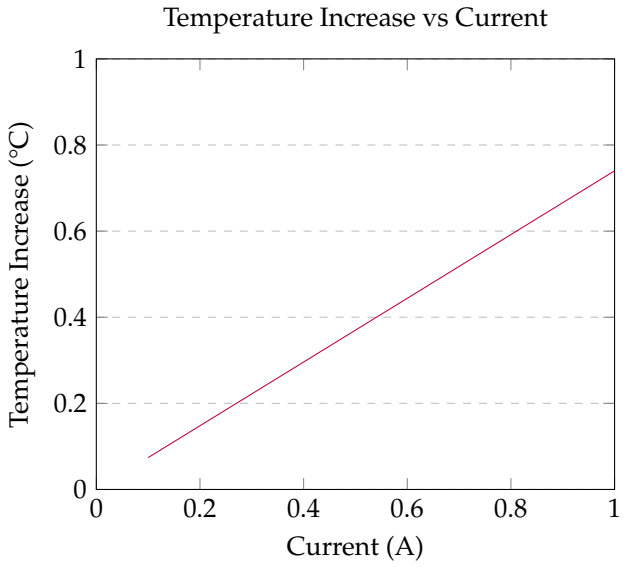


Figure 21. Temperature increase vs. current.

The flashlight application on a smartphone utilizes battery energy to emit light. This paper presents a model that captures the physical and mathematical considerations, which include power usage, light intensity, the geometry of the light cone, and signal processing for adjusting brightness.

Let C denote the capacity of the phone battery in mAh, and let $P(t)$ represent the power consumed by the flashlight at any given time t (in watts). With the assumption of a steady voltage V, the current $I(t)$ (in amperes) is associated with the power through:

$$P(t) = VI(t) \tag{74}$$

The overall energy used by the flashlight, $E(t)$, can be determined by integrating the power with respect to time:

$$E(t) = \int_0^t P(\tau)\, d\tau = V \int_0^t I(\tau)\, d\tau \tag{75}$$

The battery life T refers to the duration at which the battery is fully discharged, thus:

$$\int_0^T I(\tau)\, d\tau = C \tag{76}$$

Assuming the flashlight operates with a steady power consumption of P_0, we have the following:

$$I_0 = \frac{P_0}{V}, \quad T = \frac{C}{I_0} = \frac{CV}{P_0} \tag{77}$$

The flashlight projects light in the shape of a cone. The conical geometry is described using spherical coordinates (r, θ, ϕ), where r denotes the distance from the light source, θ represents the azimuthal angle, and ϕ is the polar angle.

At a point (r, θ, ϕ), the light intensity $I_L(r, \theta, \phi)$ is determined by the inverse-square law:

$$I_L(r, \theta, \phi) = \frac{I_0}{r^2} \tag{78}$$

Here, I_0 represents the initial intensity of light when $r = 1$. The complete luminous flux Φ produced by the flashlight is determined by integrating on the surface of the cone, within the angular boundaries $\theta \in [0, 2\pi]$ and $\phi \in [0, \phi_0]$ (where ϕ_0 denotes the opening angle of the cone):

$$\Phi = \int_0^{2\pi} \int_0^{\phi_0} I_L(r, \theta, \phi) r^2 \sin\phi \, d\phi \, d\theta \tag{79}$$

The intensity of the flashlight is adjustable through Pulse Width Modulation (PWM), where the duty cycle dictates the proportion of time the flashlight remains active. Let $\mathbf{X} = [x_1, x_2, \ldots, x_n]^T$ denote duty cycles over n specific intervals and $\mathbf{Y} = [y_1, y_2, \ldots, y_n]^T$ indicate the brightness levels aimed for during these intervals. The relationship connecting the duty cycle to the brightness level is expressed by the linear system:

$$\mathbf{AX} = \mathbf{Y} \tag{80}$$

In this context, $\mathbf{A}$ represents a matrix depicting the response of the system, essentially illustrating the relationship between the duty cycle and the resulting brightness. To determine the optimal duty cycles, we solve for $\mathbf{X}$:

$$\mathbf{X} = \mathbf{A}^{-1}\mathbf{Y} \tag{81}$$

To achieve the desired brightness while reducing power usage, we employ the Lagrange multiplier method. Define $P(x)$ as the power consumption associated with brightness x, and let $B(x)$ denote the target brightness. The corresponding Lagrange function is:

$$\mathcal{L}(x, \lambda) = P(x) + \lambda(B(x) - \text{desired brightness}) \tag{82}$$

Differentiating with respect to x and λ, we obtain the following system of equations:

$$\frac{\partial \mathcal{L}}{\partial x} = P'(x) + \lambda B'(x) = 0 \tag{83}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = B(x) - \text{desired brightness} = 0 \tag{84}$$

Determining the solution for this system yields the most efficient power consumption for a specified level of brightness.

Key mathematical components of a mobile phone flashlight app were modeled, such as power usage, light output, and brightness optimization. These models can improve the performance of the app and extend battery life.

In Figure 22, a screenshot with the flashlight ON can be seen. When the rear-view camera flashlight is turned ON, a message is shown in "Toast" that the flashlight is turned ON; this way, the user will know the status of the flashlight. The flashlight can be changed by double-tapping anywhere on the screen. This is possible because there is a large invisible button on the entire screen. The flashlight can also be toggled off the lock screen from a notification. This function was implemented because the flashlight needs to have the ability to be controlled even when the mobile device is locked.

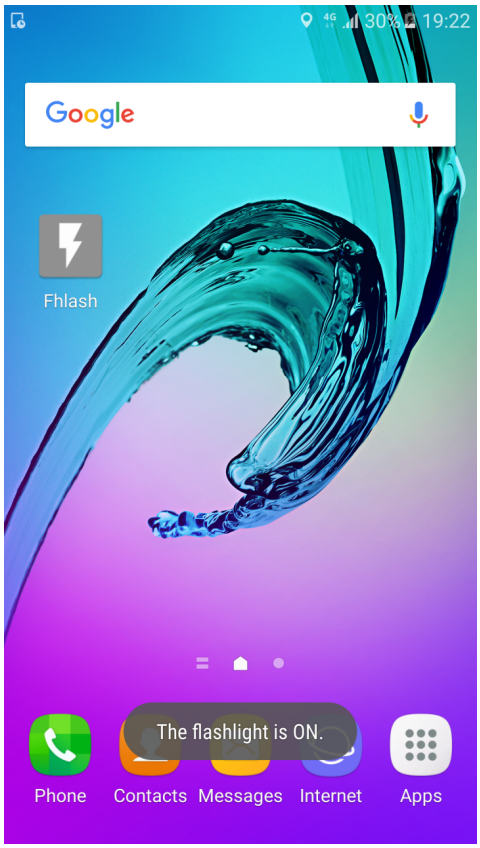


Figure 22. Turning the flashlight ON.

In Figure 23, the message that the flashlight is OFF can be seen. This message appears when the flashlight is turned OFF to inform the user. To turn the flashlight ON, it can be tapped anywhere on the screen. The flashlight can be toggled in two modes, one by double-tapping anywhere on an unlocked screen or by tapping on the lock screen on the flashlight notification.

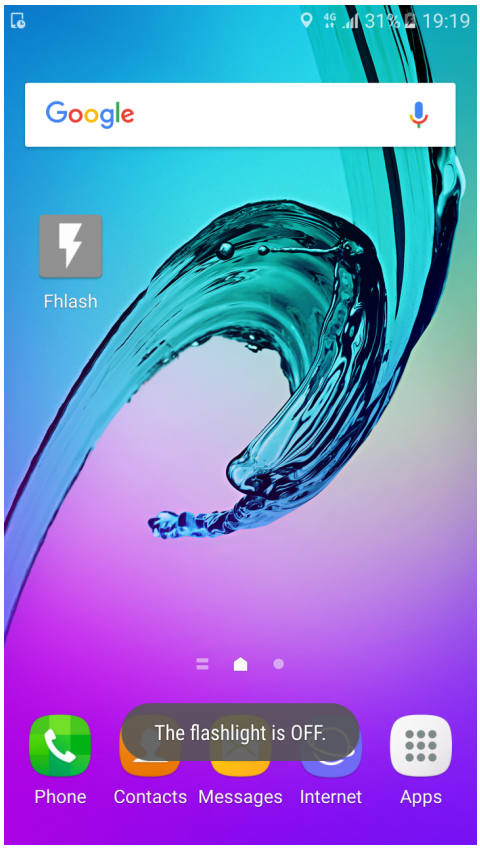


Figure 23. Turning the flashlight OFF.

3.4. Results

The apps were tested by the author and verification was carried out through experiments and presented in the graphs shown previously. An analysis of the audio signal spectrum for the voice SMS app, an analysis of the storage for the private file manager app, and an analysis of the power consumption for the new torch app were carried out. Other testing will be carried out by the users, whose comments are present on the Google Play Store and also listed below. The user experience is also expressed by their reviews and ratings. The results are not sufficient to perform statistical analysis.

Next, real user reviews and ratings of the apps, extracted from the Google Play Store, will be presented.

Some of the apps are quite new, so there are no or not too many reviews.

The vocal SMS app has an average rating of 3.667 and a review that says: "It works at first".

The private file manager app has an average rating of 2 and a review that says: "improve the back-end in file explorer but only leave the storage permissions so I give 5 stars and many more good comments, could you answer quickly, I can give more stars for that".

The novel torch app has an average rating of 2 and no reviews.

While these ratings may not be the best, comparing them with well-known Google LLC apps such as the Blogger app with a 3.7 rating and the Google Classroom app with a 2.5 rating shows that the ratings of the presented apps are not so bad. Often, users of mobile apps are harsh in their reviews.

Initially, the apps were tested by the author and beta testers. Later on, these reviews from the Google Play Store show the evaluations of the apps. These reviews also technically evaluated the performance and efficiency of the apps. Their reviews tell honestly if the apps are worth using or not.

4. Discussion

This paper presented the creation process and the mathematical background of three mobile phone apps. The paper, in addition to presenting some mobile phone apps that can ease a user's life in one way or another, also emphasizes the mathematical formulas and different evaluations of the apps using graphs. The graphs represent the spectrogram for the vocal SMS app, the storage usage of the file manager app, and the power consumption of the flashlight app. These graphs place the paper more in the mathematical sector rather than only the standard programming of mobile apps.

4.1. Vocal SMS Writing and Reading App for Visually Impaired People

As presented, a simple app was created, which redefines the meaning of SMS messaging with the use of voice interaction. The app can record a voice message, transform it into text, and send it as an SMS message. The app can also read an SMS message that is received aloud with the implemented text-to-speech software. In this way, the app can be used for vocal conversation, but with the reduced price of SMS messaging.

The adaptation is that it added voice functionality to an SMS client. The mathematics and algorithms presented here make the text-to-speech and speech-to-text functionalities. This app is better than the ones that exist, due to the fact that it is kept as a simple SMS client, but it adds only a voice option in order to be used by visually impaired people, drivers, or busy people. The idea is to use the app almost as we would make a phone call, but with the price of an SMS.

4.2. Private File Manager App

As presented, a private file manager application was created for the Android operating system. There was a limitation in the Android operating system file manager because it did not have a function to hide/show individual folders/files. The app was suggested to be used for sensitive user files, to hide selected images/videos from the Gallery app on the smartphone or tablet. This is useful if the phone is lost and found by another user or if the phone is simply borrowed from someone else. There are some data that must remain private all the time. The hardest part of the implementation was making all files visible, but with responsibility, in this way, the average user does not have access to sensitive files of the Android operating system.

The adaptation is that a show/hide functionality was added to a simple file manager. The mathematics and algorithms presented here are the parts that encode the hidden files. This app is better than the ones that exist, because the simplicity of a standard file manager was retained but with show/hide functions for sensitive files of the user, like pictures or videos.

4.3. Novel Torch App Which Can Be Switched by Double-Tapping Anywhere on the Screen

As presented, a simple flashlight app was shown, which is used to turn the flashlight ON/OFF. The flashlight of a phone is easy to use, and allows the user to concentrate on the lit object, not on the app itself. In this way, the app is very easily accessible from the phone. It also has a shortcut icon on the main screen and is also present in the notification area. When launched, the app creates a large invisible button on the entire phone screen and the flashlight can be turned ON/OFF by double-tapping anywhere on the phone. Flashlight apps are not new; they are almost the first apps that users install if they do not have a newer version of the OS that already has a built-in flashlight. The creation of a flashlight app is not the simplest task, due to camera permission, which could cause users to refuse to install the app. The flashlight was put on mobile devices to create light for pictures, not for use as a torch, but the need for light led to flashlight apps appearing. This is why

flashlight applications need camera permissions, and trusted flashlight applications will not take pictures in secret or will not spy on users who install a simple flashlight application. Camera permission is only needed to access the flashlight.

The adaptation is that the functionality to toggle the flashlight by tapping anywhere on the screen of the mobile phone was added. The mathematics and algorithms presented here are the parts that deal with the battery consumption of the mobile phone and try to optimize battery usage. This app is better than the ones that exist, because there is no app which can toggle the flashlight so easily by tapping anywhere on the screen of the mobile phone.

5. Conclusions

The presented apps contributed to the development of new algorithms or methodologies. The vocal SMS app is a simple standalone SMS app that can be used without typing. The private file manager app is a standalone file that can hide sensitive files on the mobile phone. The novel torch app can be used to toggle the torch anytime while tapping anywhere on the mobile phone's screen.

5.1. Vocal SMS Writing and Reading App for Visually Impaired People

The vocal SMS app converts talking into text and can read text messages, so one can send voice SMS messages (speech-to-text) and listen to responses (text-to-speech). The vocal SMS app is best for visually impaired people, car drivers, children who have not yet learned to write or read, or anyone who is too busy to type SMS messages.

Further enhancements to the app could be porting it to other platforms, like iOS, and working on the simplicity of the app, making it even more user-friendly.

The limitation of the vocal mobile phone app is that in the newer Android version, there should be only one default SMS app. It is hard to convince a user to switch from the default SMS app to this vocal SMS app. For this, the brand of the app needs more attention in order to increase credibility. Also, in the Google Play Store, a really good explanation is needed for applications using SMS permissions, as SMS permissions are sensitive permissions.

Further enhancement of the app should include the integration of AI capabilities to better decode speech and generate a more human voice.

5.2. Private File Manager App

The secure file manager allows one to hide files that the user does not want anyone who has access to the user's phone to see. In particular, the secure file manager intends to hide images or videos that the user wants to keep private. The file manager app is actually a file manager that renames some folders/files, but the idea of hiding/showing folders/files by renaming them to begin with a dot (".") has not been implemented in an Android app until now, for certain reasons. The idea was that Android is based on Linux, where files beginning with a dot (".") are hidden, and all this will function on Android mobile devices, where a lot of restrictions are present and even more restrictions are made after each new release.

Further enhancements would be to implement the application on iOS to be used on Apple smartphones and tablets. This would be even harder because Apple devices have even more restrictions.

The limitations of the private file manager app is that this app can only hide or unhide files, and it cannot carry out any other file operations, due to the fact that these operations are present in common file manager apps.

Further enhancement can include implementing all file manager functions to convince a user to switch to this app and make it their only file manager app. Other enhancements can be to implement AI functionalities, such as detecting unused files and archiving them to free up space.

5.3. Novel Torch App Which Can Be Switched by Double-Tapping Anywhere on the Screen

The novelty of the flashlight app is that there is no button to turn it ON/OFF; it was attempted to make it more accessible, so the flashlight can be switched ON/OFF by double-tapping anywhere on the screen or tapping on the notification of a locked screen.

Further enhancements to the app can be made to port it to more platforms, not only on Android but also on iOS. An animation icon can be added for the flashlight, not only for writing the status of the flashlight through "Toast". The animated icon can have a turned-OFF flashlight when the flashlight is turned OFF and a turned-ON flashlight when the flashlight is turned ON. In this way, there is no need to write the status of the flashlight in "Toast". Other functionalities such as flash patterns, blink patterns, and a screen-mounted flashlight can also be added to have a frontal flashlight. It can also be used to create different distress signals with a blinking flashlight.

A new enhancement can be made by making a flashlight control on the lock screen similar to the controls for music media, allowing the entire application to be on the lock screen and not requiring the unlocked mobile device to control the application.

The limitation of the novel torch app is that it can sometimes be toggled accidentally.

Further enhancement of the torch app can be to implement AI functionality to detect accidental switching of the flashlight.

Funding: This research was funded by Politehnica University Timisoara, Romania.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All data is available on request from the corresponding author.

Acknowledgments: The author thanks the Politehnica University Timisoara, Romania, for the support given.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following symbols and abbreviations are used in this manuscript:

SMS	Short Message Service
5G	fifth generation of broadband cellular network technology
MMS	Multimedia Messaging Service
SRE	Speech Recognition Engine
CV	Curriculum Vitae
GDPR	General Data Protection Regulation
DLTs	Distributed Ledger Technologies
DFS	Decentralized File Storage
MTFS	Merkle-tree-based file system
UPSS	User-centric Private Storage System
AI	Artificial intelligence
TTS	Text-to-speech
STT	Speech-to-text
c	characters
h	hidden states

$\mathbf{a}_i$	attention mechanism
$\mathbf{s}_i$	decoder states
$\mathbf{m}_i$	Mel spectrogram frames
RNN	Recurrent Neural Network
CTC	Connectionist Temporal Classification
$\mathcal{L}_{\text{CTC}}$	CTC Loss
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
$\mathbf{p}_t$	prediction
$\mathbf{W}$	character probabilities
GSM	Global System for Mobile Communications
UCS-2	Universal Coded Character Set—2-byte
UTF-8	Unicode Transformation Format—8-bit
N_s	number of septets
M	string
N_p	number of parts
RS	Reed–Solomon code
n	block length, B-tree keys, complete count of files, specific time points, dimension of vector
SMSC	Short Message Service Center
SMPP	Short Message Peer-to-Peer
L_{msg}	total message length
L_{header}	length of the SMPP header
L_{body}	length of the encoded message
$s(t)$	modulated signal
A	amplitude
$m(t)$	message signal
f_c	carrier frequency
ϕ	phase
x	input text, B-tree node
$F(x)$	feature extraction function responsible for transforming text into linguistic features
S	synthesis function responsible for creating audio signal $y(t)$ using the features, state matrix, total storage capacity
a_i	amplitude for the i-th sine wave
f_i	frequency
ϕ_i	phase offset
N	count of sine waves employed in the synthesis process
$y(t)$	audio signal input
G	function responsible for transcribing the audio signal into text
Enc	function responsible for extracting features from the audio signal
Dec	function responsible for converting features into text
$x(t)$	time-dependent function, signal's amplitude
t_n	sampling
Δt	interval between samples
x_n	sampled values of the signal
DFT	discrete Fourier transform
$\mathbf{X} = [\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_{N-1}]^T$	vector of Fourier coefficients
$\mathbf{P}$	projection matrix, diagonal matrix
$\mathbf{Y}$	compressed signal
C_0	original battery capacity
$P_{\text{transmitted}}$	signal power in the phone speaker
$P_{\text{received}(d)}$	power received at a distance d

$d(\theta)$	distance
h	height of the phone
$Q(t)$	quality of the signal
DOS	Disk Operating System
f, g	files, aggregate number of directories
SD card	Secure Digital Card
DCIM	Digital Camera Images
ASCII	American Standard Code for Information Interchange
$D = \{d_1, d_2, \ldots, d_N\}$	set of documents
$V = \{t_1, t_2, \ldots, t_M\}$	vocabulary
t_i	term
D_{t_i}	list of documents
d_j	document
BST or B-trees	binary search trees
m	B-tree order, matrix dimension
AES	Advanced Encryption Standard
XOR	Exclusive OR
r	round key
S_r	transformation for the round key r
$C = \{c_1, c_2, \ldots, c_n\}$	set of characters
$f(c_i)$	frequencies
$E(L)$	the expected length of the encoded message
$P(c_i)$	probability of the code for c_i
$l(c_i)$	length of the code for c_i
R	rolling checksum, rate at which a file is transferred
b	block of size
i	starting position, file
s_i	size of the i-th file
U	total storage utilized
s	size of the file in megabytes (MB)
$\mathbf{F}$	file content as a vector of dimension n
$\mathbf{E}$	encryption matrix of size $n \times n$
$\mathbf{F}_{\text{encrypted}}$	encrypted file
$\mathbf{H}$	function to relocate or encode the encrypted file securely
$\mathbf{F}_{\text{hidden}}$	hidden file
Ω	bounded region
(x, y, z)	files specific coordinates
(x_h, y_h, z_h)	hidden file coordinates
T_{access}	access time
α	proportionality constant
$C(t)$	battery capacity of the mobile device at time t
$P_{\text{enc}}(t)$	power consumption for encryption
$P_{\text{hide}}(t)$	power consumption for hiding
$T_{\text{total}}(t)$	total time required to access hidden files by time t, complete duration needed to retrieve N files
T_{access}	time to access files
$C_{\min}(t)$	least necessary battery level for the system's functionality
(x_h^i, y_h^i, z_h^i)	ideal coordinates for each file
$\mathbf{C}$	matrix transformation of compression process
$\mathbf{F}_{\text{compressed}}$	compressed file
$\mathbf{C}^+$	pseudoinverse of the compression matrix
E	illumination (in lux), efficiency of energy conversion, effectiveness of arranging files
d	distance, Euclidean distance

I	luminous intensity in candelas (cd), amount of current consumed by the LED, inverted index
t	battery life, duration of the transfer measured in seconds (s)
C	battery capacity in milliampere-hours (mAh), Ampere-hours (Ah)
η	efficiency of the LED driver (dimensionless, typically between 0 and 1)
LED	Light-Emitting Diode
P	power consumption, power dissipation in watts (W), probability, proportion of storage in use
R_θ	thermal resistance (in °C/W)
T_j	junction temperature of the LED (°C)
T_a	ambient temperature (°C)
Φ	total luminous flux (in lumens)
η_l	luminous efficacy (in lumens per watt)
V	voltage level of the cell phone battery
T	duration for which the flashlight can be powered by the battery, battery life
LE	Luminous efficacy
L	luminous flux measured in lumens, number of characters (message length)
B	intensity of brightness
k	constant of proportionality that depends on the properties of the LED, message length
H	amount of heat produced by the LED
ΔT	rise in temperature
C_{thermal}	LED's thermal capacity
C	capacity of the phone battery in mAh
$P(t), P_0$	power consumed by the flashlight at any given time t (in watts)
V	voltage
$I(t)$	current (in amperes)
$E(t)$	energy used by the flashlight, app
τ	time
r	distance from the light source
θ	azimuthal angle
ϕ	polar angle
$I_L(r, \theta, \phi)$	light intensity
I_0	initial intensity of light when $r = 1$
Φ	complete luminous flux
ϕ_0	opening angle of the cone
PWM	Pulse Width Modulation
$\mathbf{X} = [x_1, x_2, \ldots, x_n]^T$	duty cycles over n specific intervals
$\mathbf{Y} = [y_1, y_2, \ldots, y_n]^T$	brightness levels aimed during these intervals
$\mathbf{A}$	a matrix depicting the system's response
$P(x)$	power consumption associated with brightness x
$B(x)$	target brightness
$\mathcal{L}(x, \lambda)$	Lagrange function
λ	Lagrange coefficient
iOS	iPhone Operating System

References

1. Traynor, P. Characterizing the Security Implications of Third-Party Emergency Alert Systems over Cellular Text Messaging Services. *IEEE Trans. Mob. Comput.* **2012**, *11*, 983–994. [CrossRef]
2. Mustafa, H.; Xu, W.; Sadeghi, A.R.; Schulz, S. End-to-End Detection of Caller ID Spoofing Attacks. *IEEE Trans. Dependable Secur. Comput.* **2018**, *15*, 423–436.

3. Wang, J.; Shang, Y.; Jiang, S.; Gowda, D.; Lv, K. Whispered Speech Detection Using Fusion of Group-Delay-Based Subband Modulation Spectrum and Correntropy Features. *IEEE Signal Process. Lett.* **2016**, *23*, 1042–1046. [CrossRef]
4. Wang, Z.Q.; Wichern, G.; Roux, J.L. Convolutive Prediction for Monaural Speech Dereverberation and Noisy-Reverberant Speaker Separation. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *29*, 3476–3490.
5. Wang, Z.Q.; Cornell, S.; Choi, S.; Lee, Y.; Kim, B.Y.; Watanabe, S. TF-GridNet: Integrating Full- and Sub-Band Modeling for Speech Separation. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2023**, *31*, 3221–3236.
6. Wang, Z.Q.; Wang, P.; Wang, D. Multi-microphone Complex Spectral Mapping for Utterance-wise and Continuous Speech Separation. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *29*, 2001–2014.
7. Dasgupta, A.; Rahman, D.; Routray, A. A Smartphone-Based Drowsiness Detection and Warning System for Automotive Drivers. *IEEE Trans. Intell. Transp. Syst.* **2019**, *20*, 4045–4054.
8. Gomez, E.A.; Elliot, N. Research Article Measuring Mobile ICT Literacy: Short-Message Performance Assessment in Emergency Response Settings. *IEEE Trans. Prof. Commun.* **2013**, *56*, 16–32.
9. Nayak, S.; Chandrakala, C.B. Assistive Mobile Application for Visually Impaired People. *Int. J. Interact. Mob. Technol.* **2020**, *14*, 52–62.
10. Prasanna, S.; Vani, K.; Venkateswari, V. Design and Development of a Mobile-Based Application for Automated SMS System Using Voice Conversion Technique for Visually Impaired People. In *Innovative Data Communication Technologies and Application*; Springer: Singapore, 2021; pp. 307–317.
11. Ahn, S.; Hyun, S.; Kim, T.; Bahn, H. A compressed file system manager for flash memory based consumer electronics devices. *IEEE Trans. Consum. Electron.* **2013**, *59*, 544–549.
12. Shin, I. Implementing secure file deletion in NANDbased block devices with internal buffers. *IEEE Trans. Consum. Electron.* **2012**, *58*, 1219–1224.
13. Reyna Ballesteros, A.; Gama Moreno, L.A. Embedded file manager for mobile devices design and implementation. *IEEE Lat. Am. Trans.* **2007**, *5*, 638–643. [CrossRef]
14. Kim, T.; Bahn, H. Implementation of the storage manager for an IPTV set-top box. *IEEE Trans. Consum. Electron.* **2008**, *54*, 1770–1775.
15. Ali, M.; Malik, S.U.R.; Khan, S.U. DaSCE: Data Security for Cloud Environment with Semi-Trusted Third Party. *IEEE Trans. Cloud Comput.* **2017**, *5*, 642–655.
16. Zhu, Z.; Jiang, R. A Secure Anti-Collusion Data Sharing Scheme for Dynamic Groups in the Cloud. *IEEE Trans. Parallel Distrib. Syst.* **2016**, *27*, 40–50.
17. Tang, Y., Lee, P.P.C.; Lui, J.C.S.; Perlman, R. Secure Overlay Cloud Storage with Access Control and Assured Deletion. *IEEE Trans. Dependable Secur. Comput.* **2012**, *9*, 903–916.
18. Catuogno, L.; Galdi, C.; Persiano, G. Secure Dependency Enforcement in Package Management Systems. *IEEE Trans. Dependable Secur. Comput.* **2020**, *17*, 377–390.
19. Liao, X.-L.; Hu, S.-M. Bridging the information gap between buffer and flash translation layer for flash memory. *IEEE Trans. Consum. Electron.* **2011**, *57*, 1765–1773.
20. Wu, T.-Y.; Pan, J.-S.; Lin, C.-F. Improving Accessing Efficiency of Cloud Storage Using De-Duplication and Feedback Schemes. *IEEE Syst. J.* **2014**, *8*, 208–218.
21. Rompa, J.; Lepouras, G.; Vassilakis, C.; Tryfonopoulos, C. OntoFM: A Personal Ontology-based File Manager for the Desktop. *arXiv* **2018**, arXiv:1307.2018.
22. Zichichi, M.; Ferretti, S.; D'Angelo, G. On the Efficiency of Decentralized File Storage for Personal Information Management Systems. In Proceedings of the 25th IEEE Symposium on Computers and Communications, Rennes, France, 7–10 July 2020; pp. 1–6.
23. Kan, J.; Kim, K.S. MTFS: Merkle-Tree-Based File System. In Proceedings of the IEEE International Conference on Blockchain and Cryptocurrency, Seoul, Republic of Korea, 14–17 May 2019; pp. 43–47.
24. Bozorgi, A.; Jadidi, M.S.; Anderson, J. UPSS: A User-centric Private Storage System with its applications. In Proceedings of the 10th International Conference on Information Systems Security and Privacy—Volume 1, Rome, Italy, 26–28 February 2024; pp. 660–671.
25. Aitkulov, A.; Tosi, D. Design of an All-POF-Fiber Smartphone Multichannel Breathing Sensor With Camera-Division Multiplexing. *IEEE Sens. Lett.* **2019**, *3*, 3500904. [CrossRef]
26. Leal-Junior, A.G.; Prado, A.; Frizera, A.; Pontes, M.G. Smartphone Integrated Polymer Optical Fiber Humidity Sensor: Towards a Fully Portable Solution for Healthcare. *IEEE Sens. Lett.* **2019**, *3*, 5000304. [CrossRef]
27. Castillo-Zamora, J.d.J.; Negre, A.; Ingargiola, J.-M.; Ndoye, A.; Pouthier, F.; Dumon, J.; Durand, S.; Marchand, N.; Ruffier, F. Synchronization of a New Light-Flashing Shield With an External-Triggered Camera. *IEEE Sens. Lett.* **2023**, *7*, 6500104. [CrossRef]
28. Chan, P.P.K.; Liu, W.; Chen, D.; Yeung, D.S.; Zhang, F.; Wang, X.; Hsu, C.-C. Face Liveness Detection Using a Flash Against 2D Spoofing Attack. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 521–534. [CrossRef]
29. Escobedo Araque, P.; Pérez de Vargas Sansalvador, I.M.; López Ruiz, N.; Erenas, M.M.; Carvajal Rodríguez, M.A.; Martínez Olmos, A. Non-Invasive Oxygen Determination in Intelligent Packaging Using a Smartphone. *IEEE Sens. J.* **2018**, *18*, 4351–4357. [CrossRef]

30. Hartl, A.D.; Arth, C.; Grubert, J.; Schmalstieg, D. Efficient Verification of Holograms Using Mobile Augmented Reality. *IEEE Trans. Vis. Comput. Graph.* **2016**, *22*, 1843–1851. [CrossRef]
31. Thakur, R.; Maheshwari, P.; Datta, S.K.; Dubey, S.K.; Shakher, C. Machine Learning-Based Rapid Diagnostic-Test Reader for Albuminuria Using Smartphone. *IEEE Sens. J.* **2021**, *21*, 14011–14026. [CrossRef]
32. Collobert, R.; Bengio, S.; Marithoz, J. *Torch: A Modular Machine Learning Software Library*; IDIAP: Martigny, Switzerland, 2002; pp. 1–7.
33. Ma, G.; He, W.; Gao, M.; He, Z. The Design and Implementation of Intelligent LED Light Based on Mobile APP. In Proceedings of the Sixth International Conference on Instrumentation & Measurement, Computer, Communication and Control, Harbin, China, 21–23 July 2016; pp. 838–843.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.