2021

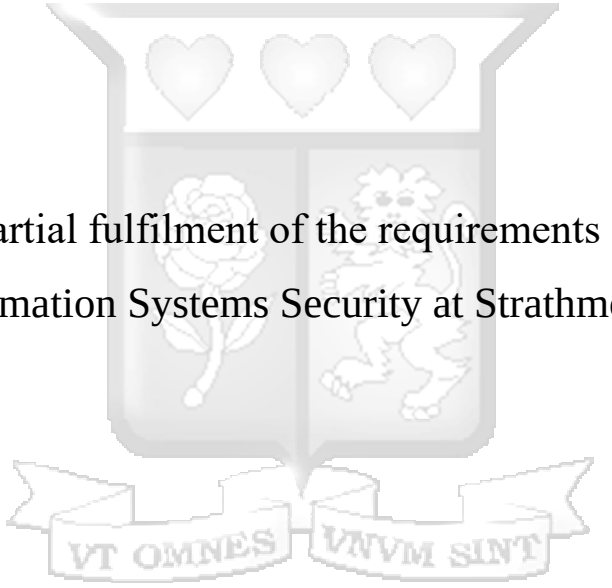
Application for enhancing confidentiality and availability for sensitive user data using AES algorithm in smartphone devices.

Nyamwaro, Valentine Nyaboke
School of Computing and Engineering Sciences
Strathmore University

APPLICATION FOR ENHANCING CONFIDENTIALITY AND AVAILABILITY FOR SENSITIVE USER DATA USING AES ALGORITHM IN SMARTPHONE DEVICES

Nyamwaro Valentine Nyaboke

Submitted in Partial fulfilment of the requirements for a Master's Degree of Information Systems Security at Strathmore University



Faculty of Information Technology

Strathmore University

Nairobi, Kenya

2021

Declaration

I declare that this work has not been previously submitted and approved for the award of a degree by this or any other University. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

Name of Candidate: Valentine Nyaboke Nyamwaro
Signature: 
Date: 9th September 2021

Approval

The thesis of Valentine Nyaboke Nyamwaro was reviewed and approved by the following:

Dr Humphrey Njogu,
Senior Lecturer, Faculty of Information Technology,
Strathmore University

Dr Julius Butime,
Dean, Faculty of Information Technology,
Strathmore University

Dr Bernard Shibwabo,
Director of Graduate Studies,
Strathmore University

Abstract

Today's world has seen a rapid increase in the number of users of mobile devices. Since the first mobile launch in the last quarter of the 19th century, mobile devices have evolved from their weight cost and functionalities to become pervasive tools. Mobile technology has provided us with mobile and flexible tools to work with, communicate, and store data. As mobility and flexibility increase, so are the risks to the information accessed from mobile devices. The devices host a lot of sensitive personal data which attackers can illegally access.

The sensitive personal data focuses on a category of select data, which the user gets to identify as sensitive on the mobile devices in the format of contacts, text messages, audios, videos, and documents such as word, pdf, excel, among others. Once exposed to unauthorised personnel or access by applications can cause exposure or harm to the user and the subjects related. Mobile devices and their storage have come under increasing attack, putting the sensitive data on the device in jeopardy. In addition, the data in the device's local storage is at risk of threats associated with the mobile, such as spam, virus, spyware, theft and loss of device and unauthorised access due to non-utilisation of the basic security measures deployed by mobile manufacturers.

The proposed solution is an Android application tool that secures all sensitive personal data on mobile devices by securely storing them in the remote cloud using cryptographic techniques. The research adopted the Agile methodology to develop the proposed solution. The methodology is more flexible and adaptable with making changes to the tool while allowing for faster delivery within a short time. The tool uses the AES 256 algorithm, and this is because from analysis of the symmetric algorithms, it is secure and with high computational complexity, and thus, any access to the encrypted data by an intruder requires comparatively more time decrypting. The tested and validated prototype provides a mechanism for restricting user access to the data with a set of authentications in the system. The tests evaluated the system performance in which it showed the encryption process and access of data averaged to a few seconds depending on the size of the file, leading to a high rating of performance. Furthermore, it exhibited a high accuracy result for confidentiality in the storage of data in the system. Compatibility tests further showed that the tool could be accessible in the different versions of the Android operating system.

Keywords: Mobile Data, Encryption, Confidentiality, Authentication, Privacy,

Table of Contents

List of Figures

List of Tables

Abbreviations

AES – Advanced Encryption Standard

CVE – Common Vulnerabilities and Exposure

CVSS – Common Vulnerability Scoring System

DES – Data Encryption Standard

IDEA – International Data Encryption Algorithm

OWASP – Open Web Application Security Project

PDA – Personal Digital Assistant

SHA – Secure Hash Algorithm

Acknowledgements

Firstly, I would like to thank the Almighty God for the strength, wisdom, and guidance he has granted me throughout the project period.

Secondly, I would like to acknowledge the continued guidance, advice and support that has been given to me by my supervisor Dr Humphrey Njogu, who has been there to provide me with valuable suggestions, ideas and corrections on how best to improve the dissertation while I have been working on it. Appreciation to Dr Drahansky Martin for his support during the writing of my proposal and crafting the idea in the initial stages.

To Strathmore University and Ilab Africa, thank you for the opportunity to pursue my masters in their program. The exposure I got to new technologies, skills, knowledge, and networks created are much appreciated.

Finally, to my larger support system, my sincere gratitude and appreciation to my family, friends and classmates for the support, understanding and encouragement throughout the dissertation duration and entire masters course. Special mention to Conseray Mabeya, Alfred, Susan and my former work supervisor Emmanuel Kwambai; they have constantly challenged me to rise in the higher academic realm.

Chapter 1: Introduction

1.1 Background of Study

As technologies advance, mobile phones and tablets are becoming more common tools for everyday use. Mobile devices are tools capable of storing and processing large amounts of information and can be carried around easily by users. Thus, mobile devices have become essential devices in modern lives. The devices tend to hold numerous types of user data, ranging from public to private data. Some personal data include contacts, text messages, videos, photos, and documents (Boyles et al., 2012).

With the growing use of the internet and smartphones, there have been considerable increases in applications developed to meet users' needs and demands. These applications have been found to gain access and disclose personal information from the users' devices by accessing them through the android permissions granted (Hutchinson & Varol, 2018). Despite google security measures in ensuring that malicious applications do not make it to the store, it is inevitable that some infected disguised applications still make it through the security screening process and to the users' mobile devices. Research conducted on privacy and data management on mobile devices by Boyles et al. (2012) indicates that there has been an increase in the malicious mobile applications and other repackaged applications acting on the adversary's behalf that execute on the mobile devices. File systems are easily accessible; once malware gets to inspect user's data, they steal sensitive information, thus invading the users' privacy with a breach of their confidentiality.

The theft and loss of mobile devices are rampant, and it is an area of security concern regarding confidentiality for users (Prey, 2020). Another area of security concern is mobile devices with unprotected local storage, and this is usually due to vulnerabilities from operating system or manufacturer model defects that can be exploited and situations where users do not get to utilise the basic authentication methods such as pin and lock patterns to protect the data from being accessed. Once such a device falls into the wrong hands, it becomes easier for unauthorised personnel to access the device local storage and access the sensitive data that can easily be used in ways that can negatively harm or impact the user (Kearns, 2016).

Confidentiality is such an essential element in maintaining humanity, social responsibilities, and relationships, and as such, humans value their privacy and the protection of their sensitive data. Data can reveal information about a person, including their relations in life, through contacts, images, videos, messages, and documents stored on mobile devices. There is a possibility of hijacking this data for potential misuse, and once this happens, the hijacked sensitive data threatens the privacy of the users with a chance of a range of negative consequences. Hoven et al. (2019) discuss moral reasons for limiting access to personal data and giving users control over it. The moral reasons include prevention from harm, informational inequality, informational justice and discrimination, and the encroachment on moral autonomy and human dignity. Ciphr (2020) reports that several cases of mobile data breaches have been experienced over the years; some of them have been reported, while for others, the users ignored and dealt with the consequences privately. Among the people who have been reported to have experience mobile data breaches on their personal sensitive data, they include the celebrities Emma Watson, Tiger Woods, and Olympics Skier Lindsey Vonn. Due to hacking, their data from their local mobile storage was leaked online, costing them their reputation. This act shows the need for mobile data protection as a security concern for extensive consideration.

Security measures such as passwords are utilised as a standard measure in securing sensitive data on mobile devices. Although the use of a password as a security mechanism may contribute to confidentiality, its protection is instrumental to the safety of the data. Whilst technology is often considered the cause of the confidentiality problem, it also provides several technological ways to help prevent the issue. These technologies include communication anonymising tools such as Tor and Brave browsers. They allow users to browse the web and share content securely. Cryptography is a technology for confidentiality used to protect data through encryption to transform it into an unreadable form. Other technologies include hashing for secure storage of passwords and virtual private networks to create secure connections over networks and the use of antiviruses in devices for detection, neutralisation and eradication of viruses, malware, and spyware. (Hoven et al., 2019)

This study strives to achieve confidentiality on the sensitive personal mobile data, focusing on securing the select data files identified by the user as sensitive, ensuring to secure them remotely and making them available. The designed tool provides a solution for enhancing the confidentiality of the users' sensitive data on the mobile device storage. The tools use the encryption algorithm

AES 256 to encrypt the data and then get uploaded to the cloud, where it is stored for availability and protection from unauthorised personnel and applications.

1.2 Problem Statement

Several threats are associated with smartphone devices, leading to unauthorised access of the mobile user device, putting the data in the mobile device in jeopardy. In the research conducted by Rashidi and Fung (2015), several security threats have been identified on android devices. Further analysis and research by CVE details (2021) identify the following rising security threats on android mobile devices, information leakage, permission escalation attacks, denial of services, and malware attack through repackaged applications. When an adversary acts upon these threats, they strip the users of the right to privacy by compromising the confidentiality of data in mobile devices. In the bid to secure data at rest on smartphones, developers have come up with applications such as lock patterns, pins, and fingerprint readers to prevent unauthorised access to it. However, further research established that hackers could easily crack patterns and pins to grant them access to the data in smartphones (Zednet, 2016). Therefore, this poses a problem as the sensitive data stored in the mobile device in unencrypted format can be accessed and read by malicious hackers and other users in case of mobile loss. In the event of a mobile ransomware attack, the malware steals sensitive data or locks the mobile device permanently, demanding payment before unlocking it or returning the data to the user, thus making the data unavailable

1.3 Research Objectives

The study aims to design and develop a user-friendly tool to enhance confidentiality for the sensitive personal user data stored in smartphone mobile devices.

1.4 Specific Objectives

1. To identify the threats for breaching the confidentiality of data stored in mobile devices and their effects
2. To review the current tools and solutions for enhancing confidentiality and availability in sensitive personal data on mobile devices and identify the existing gaps.
3. To design, develop and test a mobile application tool for enhancing confidentiality of sensitive data on the mobile device and securely store them in the cloud.

4. To validate the security effectiveness for providing confidentiality of the data through encryption and availability via cloud storage of the application.

1.5 Research Questions

1. What common threats exist for breaching the confidentiality of sensitive personal data stored in mobile devices?
2. What tools and solutions exist for enhancing confidentiality in sensitive personal data on mobile devices for users?
3. How will the application be designed, developed, and tested to enhance the confidentiality of sensitive data on the mobile device and making it available?
4. How will the application be validated for security effectiveness?

1.6 Scope and limitation

The mobile data encryption tool is an android virtual drive encryption utility application that streamlines the confidentiality and availability of data stored in android smartphone devices. This tool enables users to store their sensitive data without worrying about confidentiality when the mobile device is lost and the availability of the data. It gets to encrypt the data and uploads it to an online server where it is stored in an encrypted form, in which it can be retrieved and viewed from any android smartphone mobile device with the use of the encryption key created by the user.

The data stored can be retrieved using a different mobile device using the user's logins credentials to access the application in case of a device loss.

Owing to the popularity of Android-based smartphones in the market, as they hold most of the market share, the scope of this tool is limited to the android operating system.

1.7 Significance of the study

Mobile devices are at risk and vulnerable to attacks for unauthorised access, there have been several solutions so far that have been provided to safeguard access to the mobile devices, but with time they are proving not to be the most robust ways of providing confidentiality of the data in storage. With these tools, there are processes through which attackers can access the security features on the mobile devices, and to some extent, some users do not utilise the basic features

such as passcodes, making it even more straightforward for the attackers to get access to the information.

Mobile devices theft and loss are also quite rampant to be overlooked. With the amount of sensitive and personal information stored on mobile devices, it is essential to protect sensitive data from unauthorised access and get exposed by malicious applications and users.

Chapter 2: Literature Review

2.1 Introduction

Mobile devices such as smartphones and PDAs are now essential personal and business communication tools. Among other functions, mobile devices are used to make calls, exchange multimedia messages, surf the web, check emails, pay bills, download games, download music, and more. To support its functions, mobile devices today carry with them confidential information such as emails, messages, pictures and contact numbers, which exposes the owners of these devices to security risks and threats unknown before. In a survey by the computer security professional, 42 per cent cited theft of mobile devices as a top information technology threat (Grover, 2016). However, the loss of the device itself is perceived to be small compared to its content value.

Data has got much value in the digital world, and thus protecting data is of utmost importance. Not only must the transactional data be safeguarded, but the sensitive personal data as well, as it gets to reveal relationships of individuals and cooperates. In realising the importance of protecting confidential information, some entities and users have taken vital steps to secure the information and data within their premises and computers. However, often preserving the confidentiality of data residing in mobile devices is either overlooked or taken lightly, resulting in the loss of essential and confidential data. Since, in this case, most data and information are no longer just residing in static devices like servers and storage, protecting the data and information in the mobile devices is thus becoming a critical issue. A report published on mobile security indicates that 2 out of 10 companies have experienced a mobile cyber-attack with a larger utterly unaware of whether they have been breached (Dimensional Research, 2017).

2.2 Mobile Devices and Personal Data

With the increase in mobile device sophistication, smartphones hold much power due to the amount of data available in them. The information generated and stored on these devices can either be classified as public or private information based on the sensitive nature of the data. It is essential to protect the sensitive data in the device from prying eyes.

According to Ohm (2015), Sensitive data is described as data that can be used to enable the privacy or security of an individual when placed in the wrong hands. When lost or accessed unauthorised,

sensitive information can lead to significant contractual or legal liabilities, especially at the organisation level, as it can cause severe damage to the organisation image and reputation and individual negative reputation. In this research, sensitive data can be identified based on the users' knowledge of the compelling form of harm it can cause to their daily lives. It can also be found on the state of damage it exposes the user to and the harm that the data can cause and tell most of about the subjects involved in the data from the relationships that can be linked.

There is the idea that the variety of laws that define sensitive data have a common approach underneath the surface. It derives the unstated factor that helps determine whether some specific type of data is sensitive and worthy of protection. This thou get to serve as a descriptive and normative restatement of the factors that add up to the conclusion of sensitivity. Sensitive data can be further classified into at least three types. The three types of classification entail.

Inherently sensitive information: This data can cause concrete harm merely by being known to another. This category of information can usually be connected to a category of ancient damage such as blackmail and harassment. The information can be often considered embarrassing, and as such, such kinds of data tend to be held closely in confidence. The second category of sensitive data can be classified as inferential sensitive data; This type of data is connected to harm through at least one inferential or predictive step. This can be from past or present documents of individuals relating to education, ethnicity, and creditworthiness. The third category of sensitive data can be classified as Instrumentally sensitive data. The data in this category can lead to harm, but not, merely through the knowledge or inference of another person, the damage it causes if from being used, example in this category could be hackers obtaining a document bearing your home address, which gets to be used with stalkers and predators to harm someone.

2.3 Mobile Device Platforms

The mobile environment is perceived to be vulnerable due to its features such as open medium, network topology and decentralised management. The threats associated with mobile devices vary depending on the operating systems (Kearns, 2016). A mobile operating system is a software that allows smartphones and other devices to run applications and other programs, and they manage cellular and wireless connectivity and access to the device. The mobile operating systems in the market include Apple IOS, Android, Blackberry and Windows. Android mobile

devices are the most popular used operating system in the market, as they take up to 75% - 80% of the mobile devices in the market (Zhernakov & Gavrilov, 2017). Despite Android having the largest market share, Apple operating system has been considered for some time to be more secure. This is primarily attributed to Apple being a closed system while Android offers an open-source system (Garg & Baliyan, 2021). The open-source code is available to developers and device manufacturers to build applications for the platform and modify the system to the needs of the hardware performance.

Farooq (2018) described the Android system structure as a Linux kernel that implements a monolithic architecture whose components are interconnected and composed as one piece. All the kernel components such as shared memory, power management and drivers run in the same layer with no isolation between them. Figure 2.1 show an android operating system architecture. Device manufacturers often use custom drivers to support their hardware, and this customisation gives a chance of having poor quality made drivers that could pose a security threat. Unlike Android Operating System, Apple IOS cannot be installed on any device that is not produced by Apple as there is no license for that, and this gives it a security advantage in managing the tests and quality of drivers being implemented in the IOS architecture.

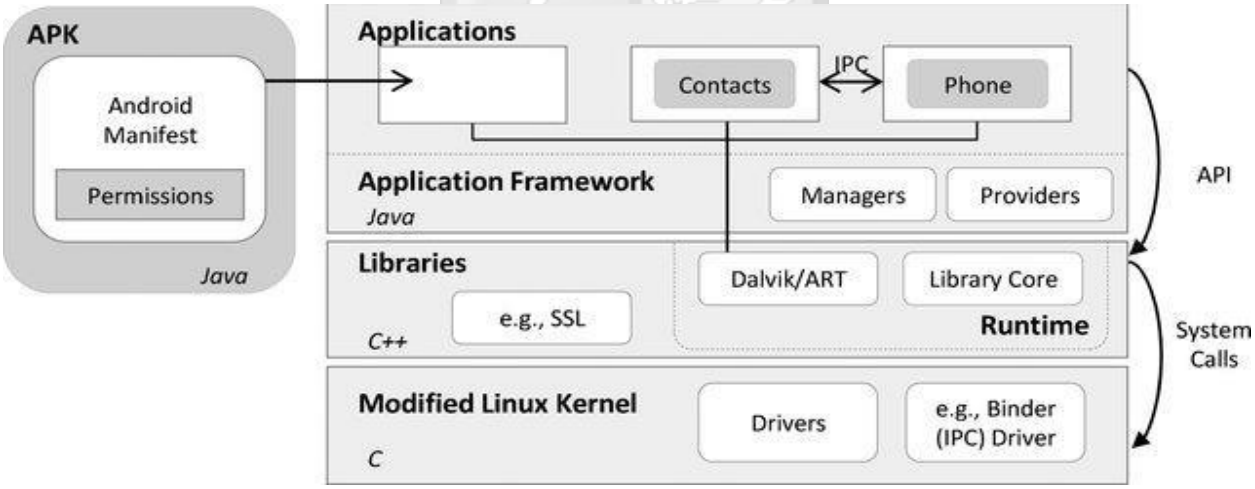


Figure 2.1: Android Operating system architecture

Source: (Tam et al., 2017)

2.3.1 Operating System Protection

The operating system communicates directly with the hardware or the drivers. Its main principle is to provide hardware features to users of its system. The security of the operating system is thus focused on protection mechanism and its control access. Protection mechanism access control to a system works by limiting the types of file access being granted to users. This mechanism ensures that only authorised processes from the operating system can operate on the memory segments, the processors and other device resources. Protection is provided by a mechanism that controls the access to applications, processes or users to the defined resources in the system (Wessel et al., 2015). Security in the system ensures the authentication of the system users and in protecting the integrity of the information in the system. The security system thus prevents unauthorised access, malicious alteration of data and system modification to introduce inconsistency.

As the Operating System become more sophisticated and pervasive in its applications, the need to protect its integrity has risen (Furnell et al., 2008). Protection was initially conceived as an adjunct to multiprogramming operating systems, so untrustworthy applications could safely share a standard logical to a directory of files. With such, modern security concepts would now need to evolve as there is an increase in the use of shared resources in the operating system.

While the application makes access requests to gain the necessary permissions to protected resources granted by the kernel, the user also controls some of the permissions. One central problem, however, is the inability of users to make good security choices. According to Li and Clark (2013), the typical users neither have the necessary understanding of the available security mechanisms nor the ability to utilise those protection mechanisms to their full benefit properly. Thus, users have little trouble granting dangerous permissions when installing free Android applications. The burden placed on users to manage arcane security options is a risk opportunity for attackers to take advantage of the gap between the perception and reality of risk.

2.4 Threats for personal data in mobile devices

Today, mobile devices have come under increasing attack, and no one is immune. Mobile devices, just like personal computers, are vulnerable to spam, viruses, spyware, theft, loss, and even phishing attacks. The data breach may not be as widespread as reported, but it is rising (Collett, 2017).

2.4.1 Loss or Theft of Device

The convenience of mobile devices, has led to information being stored in it that is both sensitive to the user and confidential to corporates. This is quite a challenge as it further aggravates the risk of loss of confidential information. With the devices being tiny, it makes it relatively easy to move around with, and in most cases, the devices tend to be stolen or sometimes misplaced, and with the confidential information stored in it, it can get to be misused if it falls into the wrong hands (Hernandez et al., 2018).

2.4.2 Malware and Spyware

Mobile Malware and Spyware is also a significant threat to mobile devices. In research conducted and a report published in the first quarter of the year on mobile threats, McAfee labs could detect more than 1.5 million new mobile malware incidents of mobile devices (McAfee, 2017). Hackers use mobile malware to gain access to devices to access unauthorised data. This malware can find its way to mobile devices through malicious applications installed on the devices. The malware could have codes that get to execute on the mobile devices, and with the various access permission granted through the android permissions, they can get to monitor, track activities, steal sensitive data from the mobile device, including making unauthorised calls and sending text messages on behalf of the user.

2.4.3 Permission Escalation

Android applications do not usually have permissions associated with them by default. They are usually enforced with android mandatory access control in the form of a permission system, and thus, to gain access to the applications, they request the user for the permissions needed during installation. Rashidi and Fung (2015) classify android permissions into four categories. These categories include normal, dangerous, signatures and signatures or system permissions. Normal permission is the lower risk permission granted to the applications to access the isolated application features with minimal risk to other applications in the system and the user. Dangerous permission is higher-risk permission granted to applications to access and control the private user data, which can negatively impact the user. Signature permission is that type of permission given to the application by the system based on whether the requesting application has a signed certificate similar to the application declaring the permission. Signature or system is the permission granted

to applications in the android system image or signed with a certificate similar to the application declaring the permission. The threat thus emanates from giving irrelevant access to applications during the installation process. During installation, applications usually request the user access to system resources such as contacts, SMS, gallery, camera, and storage. The applications requesting permission often are designed with no options for the user to grant access selectively; it usually either in whole or not granted at all, which leads to the successful or unsuccessful installation of the application (Farooq, 2018). The granting of permission leads to many applications gaining access to personal data on a mobile device.

2.4.4 Insecure Data Storage

In the OWASP, Mobile Top 10 list that identifies the type of security vulnerabilities that are faced by mobile apps globally, it enlists insecure data storage as the second most vulnerability on mobile devices (Qian et al., 2019). It categorises the insecure data storage risk as easy, with common prevalence, average detectability, and severe impact if exploited. It further explains the easy ways through which an adversary can access the insecure data storage by gaining access to a stolen device or by using malware that can be facilitated through repackaged applications.

2.5 Trends Increase in Vulnerabilities

Further study on the android vulnerabilities has revealed a drastic increase in android vulnerabilities over the years (Umasankar, 2017). Statistics from the ultimate security vulnerability data source supports the claim. The CVE details are an all-vulnerability database that uses common identifiers to represent the vulnerabilities using a common vulnerability scoring system score (CVSS). Based on the CVSS, further classification of the identified vulnerability is done based on its severity level. Table 2.1 indicates the vulnerability increase over time.

Table 2. 1: Number of Vulnerabilities reported on Android over the years

YEAR	No of Vulnerabilities
2013	7
2014	13
2015	125
2016	523

2017	842
2018	613
2019	414
2020	859

Source: (CVE details, 2021)

Further statistics on the vulnerabilities provide an overview of the most occurring and reported threats over the years on android devices. The highly reported is the code execution vulnerability, followed closely by the denial-of-service attack and gaining information for storage of mobile devices. Figure 2.2 gives an overview of the vulnerability analysis.

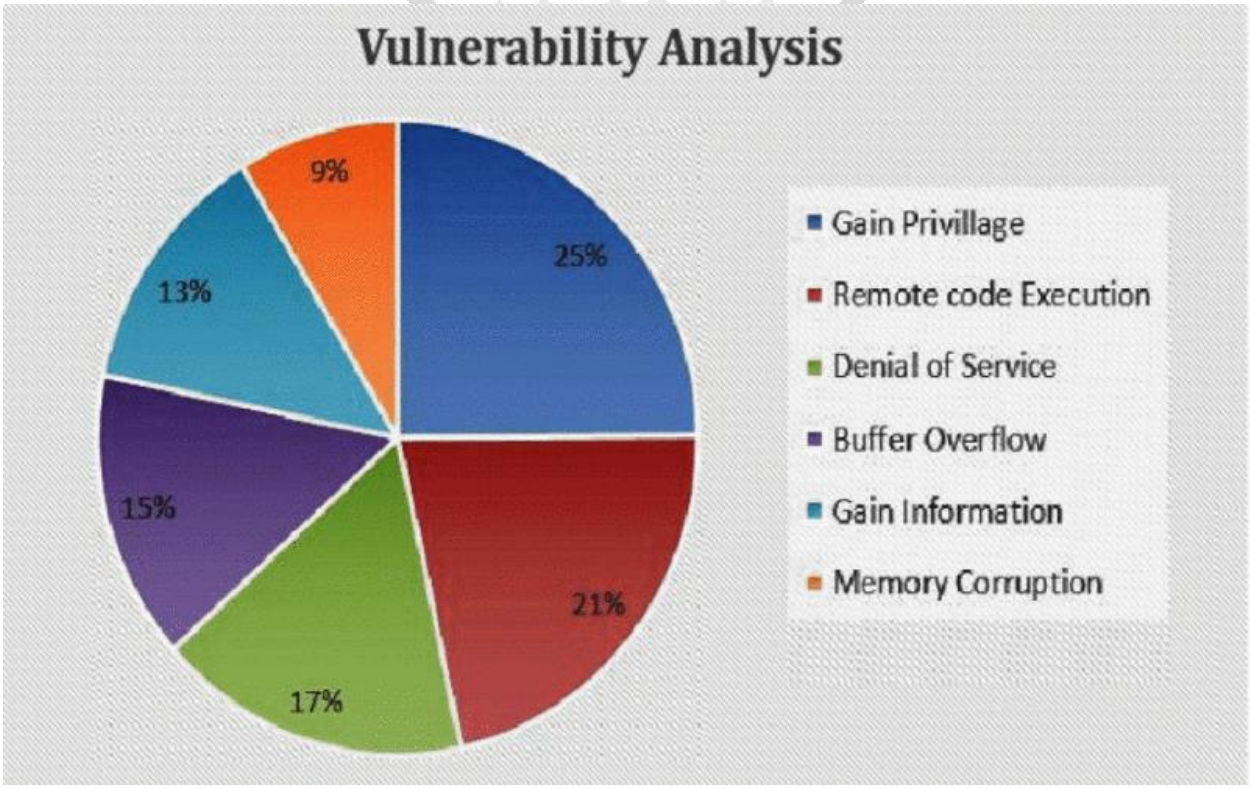


Figure 2.2 Android Vulnerability Analysis

Source: (CVE Details, 2019)

The identified vulnerabilities have been rising over the years. The figure 2.3 gives an overview of the common trends by year and rise.

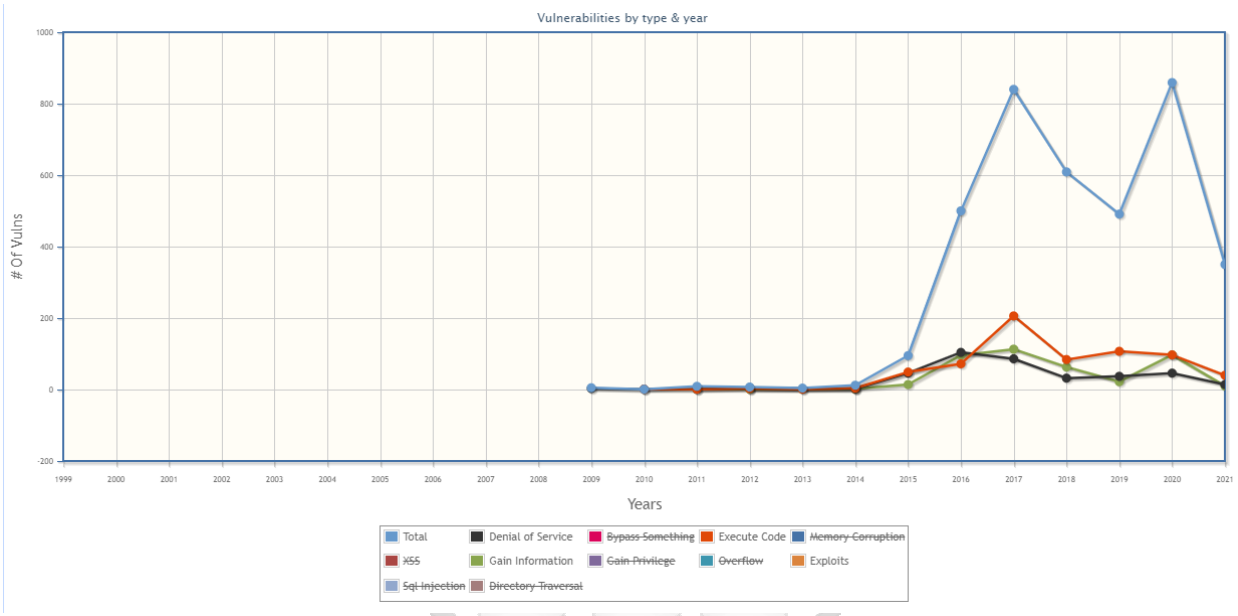


Figure 2.3: Vulnerability Trend Over the Years

The identified vulnerabilities have been exploited over time by hackers using various malware and viruses to gain access to mobile devices.

Hence with more sensitive data in devices against the vulnerabilities in the device storage, safeguarding it is a requirement, and the best techniques are required in offering the necessary protection for the data at rest on the mobile devices. Cryptography algorithms are widely available and recommended for use to accomplish data security (Irwan et al., 2016).

2.6 Cryptography

During storage and transmission of data, it is crucial to secure the data only to be read or processed by those for whom it is intended, and cryptography achieves this. Cryptography is a method in which information and communication are protected with codes only to be accessed by those it is intended. The protection is achieved using mathematical concepts, and rule-based calculations called algorithms to transform the data in hard to decipher ways, as shown in figure 2.4 (Kessler).

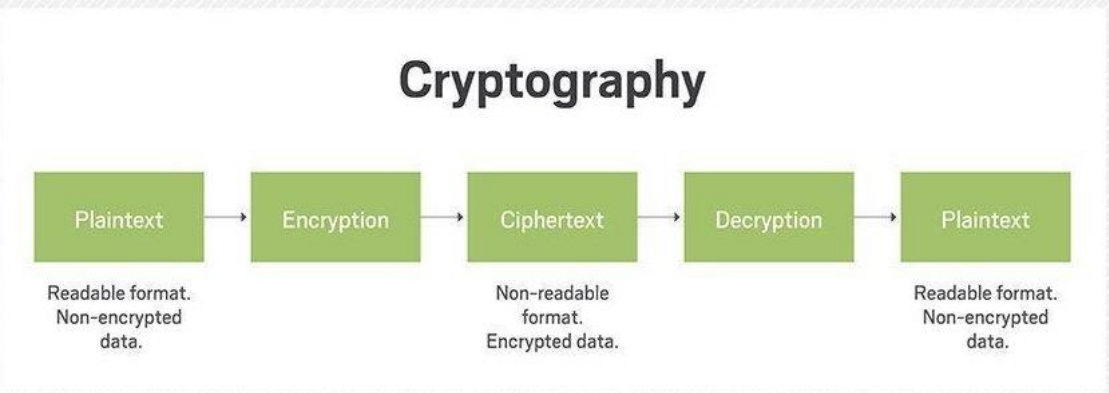


Figure 2.4: Cryptography process

(Source: TechTarget)

There are many processes of achieving cryptography using algorithms, but the most common and widely used methods can be grouped into two major categories: Asymmetric and Symmetric Cryptography.

2.6.1 Asymmetric Encryption

Asymmetric encryption is also known as public-key encryption. In this type of cryptography, encryption is usually accomplished using public and the other key private. The private key is used to encrypt, decrypt and digitally sign files, and the public key can be used to do the same. The public key is freely distributed and widely accessed for use, while the private key is only known to the user. Despite the broadcast of the public key being that they are generated as a pair, it does not make the private key insecure, as the knowledge of one does not allow someone to determine the other key quickly. To accomplish the encryption process for using the technique, it does not matter which key is applied first, as both keys are required for the process to be complete (Solanki & Patel, 2012).

Merkle, Diffie and Hellman introduced the idea of Public Key Cryptography. Its main goal was to provide privacy for two parties to communicate securely over a non-secure channel.

2.6.2 Symmetric Key Cryptography

This cryptography is also known as private key cryptography, in which encryption and decryption are performed using the same key. Symmetric cryptography is based on a block cipher or a stream

cipher concept. The majority of encryption applies the block concepts as they are mostly used to encrypt large data sets. In this concept, the set lengths of the data sets are usually encrypted in blocks of data to achieve the final encryption (Kaur & Mahajan 2013). The Symmetric algorithm in use among them includes Data Encryption Standard (DES), Advanced Encryption Standard (AES), International Data Encryption Algorithm (IDEA) and Blowfish.

2.6.3 Comparative Analysis of Encryption Algorithms

As electronic data rises, there is an expanding essential for data protection, and in selecting the most effective algorithm, the encryption must ensure complete protection without compromising efficiency (Toldinas et al., 2011). In a research conducted by Yegireddi and Kumar (2016) on the performance evaluation of different cryptographic algorithms, they determined that the security of an algorithm is based on the length of the key in use. The comparison of the private and public-key cryptography showed that private key algorithms require less memory than the public key algorithm, while in terms of speed, it was determined that the computational speed of the private key algorithm is much faster than the public key algorithm.

In further comprehensive evaluation research by Patil et al. (2016), a comparative analysis of the commonly used encryption algorithms' performance efficiency and security measures is carried out. The comparative analysis is of the algorithms AES, Blowfish, DES and RSA using the parameters key length, turnability, speed, encryption ratio and security attacks. The implementation results reveal that the algorithm Blowfish scores the highest average entropy speed per byte of encryption, while RSA is the least efficient in terms of encryption speed. Blowfish takes the least memory in the memory required, followed closely by AES and DES, while RSA requires the most extensive memory. In evaluating the algorithms based on the parameter of avalanche effects, the AES algorithm scores the highest, thus making it the most efficient for use in applications where confidentiality and integrity are of the highest priority (Patil et al., 2016). The common algorithm on energy consumption cost analysis of mobile data encryption and decryption research conducted by Pry and Lomotey (2016) concluded that the AES algorithm is the most effective for mobile devices due to its low power usage.

2.7 Mobile Confidentiality Best Practices

As an IT strategy for ensuring confidentiality, integrity and availability of the data, encryption provides the solution to protect the data that lives on and between the devices. Encryption is one of the most powerful ways to keep the user's data safe, and while it isn't impenetrable, it's a significant deterrent to hackers (Kumari, 2017). Even if data does end up getting stolen, it will be unreadable and nearly useless if it's encrypted. According to Heath (2018), mobile phone makers and operating systems developers have incorporated encryptions on the devices to be utilised by the users. The devices can achieve the available encryptions solutions in the following ways.

Full device encryption (FDE**)**: This is the primary way to protect the mobile devices and the at-rest data. Any files saved to the device are automatically encrypted.

Pre-encrypting data that's synced with the cloud: there are some software available that can get to pre-encrypt data before it is synced to the cloud, making it unreadable by the cloud or anyone who hacks into it. This leaves the files still stored on the device unencrypted and are still vulnerable.

File encryption: This provides a way to encrypt at-rest data on a file-by-file basis, not to be read if intercepted. This isn't automatic, but it's beneficial because it stays encrypted after leaving its place of origin.

As a common solution to users on enhancing confidentiality of data, different mobile models and researchers have come up with various mobile tools to solve the challenge.

2.8 Review of the Existing Solutions

2.8.1 Samsung Knox Platform

This is a security feature developed by the Samsung Electronics Group for the protection of data in the devices. A whitepaper published by the Samsung Knox platform (2015) states that the Samsung Knox platform is an integrated suite of security features that protects sensitive data on a Knox enabled device. Knox is built into the hardware and software protection of the Samsung flagship devices. The Samsung Knox is serviced by the principles of software integrity, data storage protection, data isolation and the least privilege to provide security for the mobile user. The system is developed to detect any modification in the kernel, and it offers secure boot, trusted boot, and security enhancements for android protection. The whitepaper by Samsung Knox on the

Knox overview reveals that for the software to ensure data protection with the least privilege, it ensures a minimum level of access for applications in the device to do their job. Furthermore, it provides on-device encryption for Knox sensitive data protection and further isolates data by providing a secure workspace that stores private data away from the untrusted application (Dorjmyagmar et al., 2017). Figure 2.5 shows a standard Knox architecture.

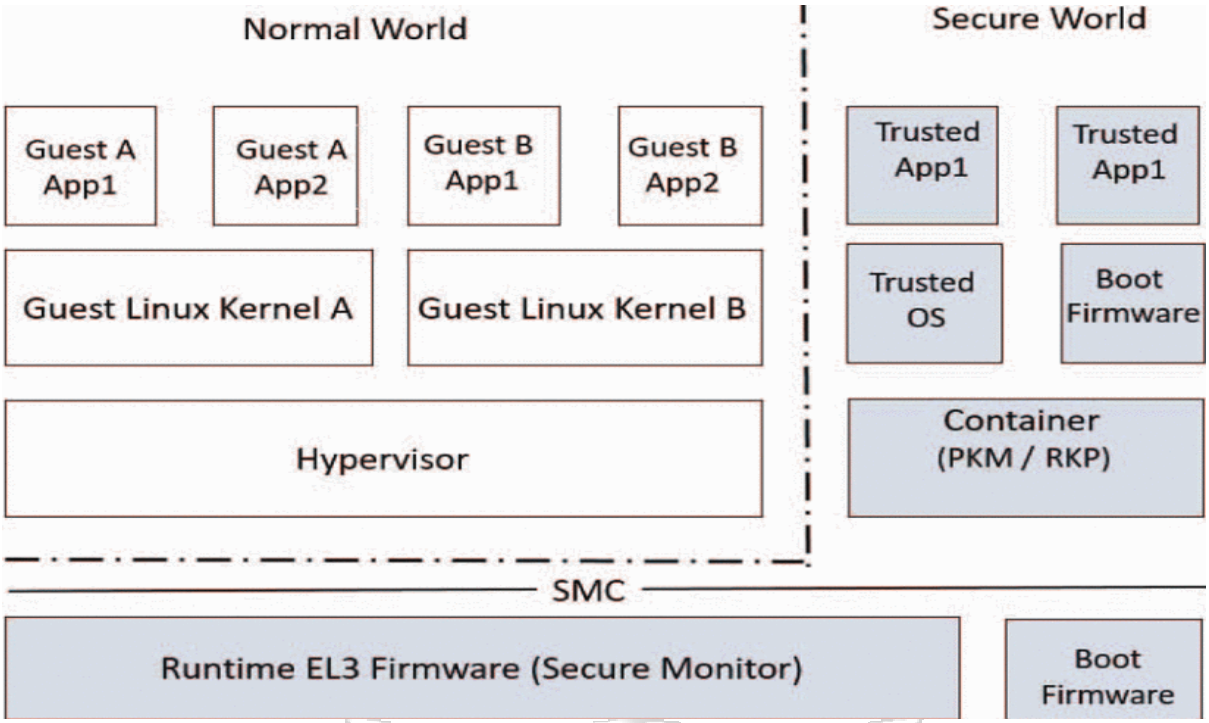


Figure 2.5: Samsung Knox architecture

Source: (Dorjmyagmar et al., 2017)

The Multi-layered mobile security platform consists of the Trusted Execution Environment that resides in the secure world in ensuring storage of sensitive data, processing and protection. The trust zone Trusted Execution Environment performs attestation and provision of the secure storage mechanism for crypto keys (Krishnan, 2015)

Despite the security measures implemented in the system in ensuring confidentiality and integrity, further research conducted by Dorjmyagmar et al. (2017) on the security analysis on the Samsung Knox reveals that there are attacks scenarios that can be utilised to get access to the user device. Among the vulnerabilities include clipboard data exposure with CVE 2016-3996 that an adversary can use to intercept the contents of the Knox clipboard. Furthermore, ecryptfs Key Exposure with

CVE-2016-1919 reveals a gap in which an adversary can decrypt the Knox data without the user password. This is because the key used for encryption, the master key, is generated by the key derivation algorithm PBKDF2 that is susceptible due to the inadequate generation of the ecryptFS key. Knox USB attack using ADB and man in the middle attack is also the other attack scenarios identified for Samsung Knox.

2.8.2 AppLock for Efficient storage Encryption

AppLock is an android mobile-based solution that provides a secure framework for storing data on mobile devices. The developed solution was provided by Lu, Kuo and Fang in their research for efficient storage encryption for Android mobile devices. The solution integrates AES, Bcrypt, PBKDF2 encryption and MD5 hash schemes to provide efficient storage (Lu et al., 2016).

The *AppLock* solution consists of three components for the design of privacy: The Activity lock subsystem, data protection, and authentication system. The *Applock* activity lock subsystem implements an interface that enables the user to lock applications in the device using a password. The password is defined by the user and protects in storage by the MD5 hashing algorithm. The data protection subsystem implements an interface that allows the user to select a list of files from the data stored on the mobile device to protect it. The files are protected with the use of a password that the user enters. The subsystem then uses BCrypt to generate the first set of ciphertext code which becomes Key 1 and is stored locally on the device. The data protection subsystem further uses the PBKDF2 to generate the second set of ciphertext code, which becomes Key 2 and is used to encrypt the data with AES encryption. Figure 2.6 provides an overview of the overall procedure for the AppLock data protection system, while figure 2.7 gives an overview of the encryption procedure.

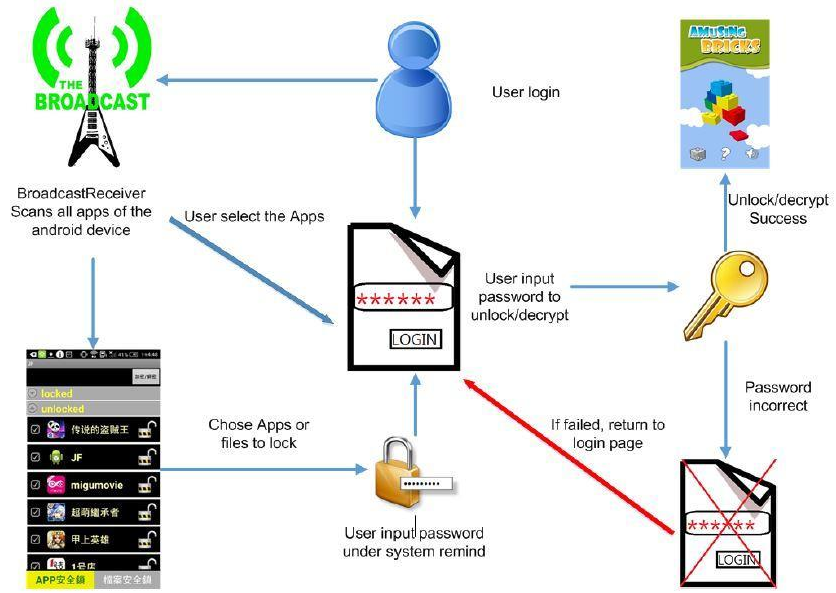


Figure 2.6: AppLock Data Protection Procedure

Source: (Lu et al., 2016)

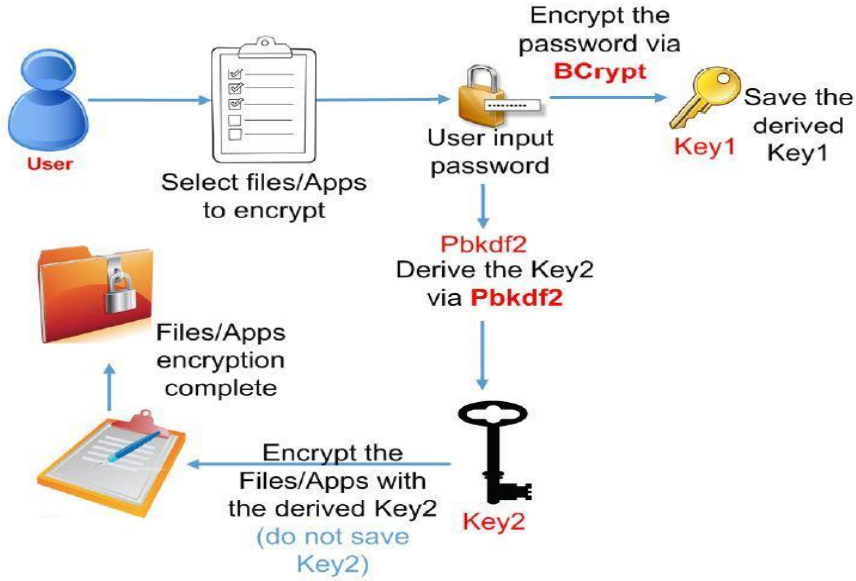


Figure 2.7: Encryption Procedure.

Source: (Lu et al., 2016)

2.8.3 CryptAll Application for Data Security

A CryptAll android solution is a tool developed to provide data privacy that uses AES 256-bit encryption and decryption algorithm to accomplish confidentiality on mobile devices. It was developed by Vishare et al. (2017) in their research on data security using authenticated encryption and decryption algorithm for android devices. The tool provides the features for registration, encryption, and decryption with an option for cloud backup of the encrypted files. The figure 2.8 provides an overview of the CryptAll application interface. It functions in a way that the user gets to mark the sensitive files in the device, and in the event of an intrusion, the files marked as sensitive are automatically encrypted on the device. Furthermore, the solution offers a backup to cloud option for the encrypted files to prevent data loss. However, to accomplish this, the file has to be manually encrypted and manually uploaded to the cloud.

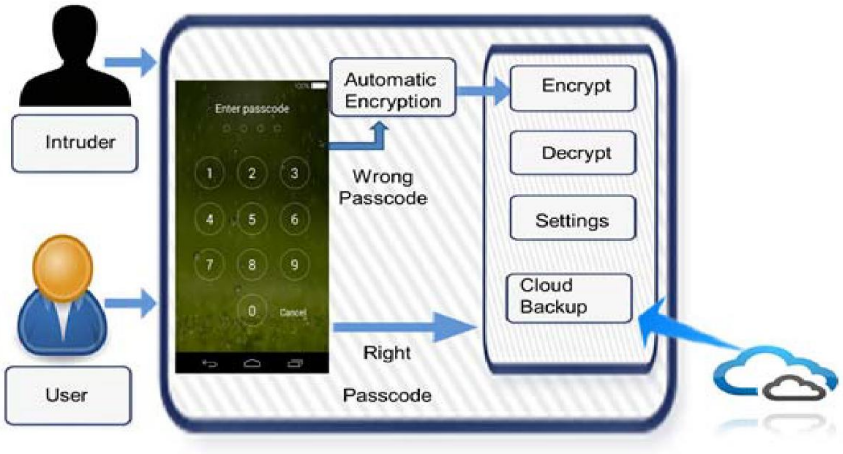


Figure 2.8 CryptAll System architecture

Source: (Vishare et al., 2017)

2.8.4 Mobiflage for Deniable Storage

The Mobiflage is a solution designed for data confidentiality on mobile devices using plausible deniability encryption. It enables Plausible Deniability Encryption by hiding encrypted volumes with random data in the device free storage to conceal the existence of additional volumes. Mobiflage implementation is in two variants, the Mobiflage-SD designed for use on devices with FAT-32 removable SD cards, while the

Mobiflage-MTP variant is designed for use on devices that share a single internal partition for both applications and user-accessible data (Skillen & Mannan, 2014).

Mobiflage consists of features that define the mode of operations of the device into two, that is, the standard mode and the Plausible Deniable Encryption mode. The standard mode is used for the device's day-to-day operation by providing storage encryption without deniability. In this mode, the user supplies their decoy password at boot time, and the storage media is mounted in the default way. The plausible deniable encryption mode is used whenever the user needs to store sensitive data whose existence may need to be denied when coerced. The user supplies their true password during boot to activate the PDE mode and mount the hidden volumes. Figure 2.9 illustrates the PDE mode. In the storage layout, the entire file is encrypted with a decoy and formatted for regular use. This is referred to as the outer volume. Hidden volumes are additional file systems created at different offsets within the disk and encrypted with different keys. Hidden volumes are the decoys, but there exists at least one volume containing the actual sensitive data that s encrypted with the true key.

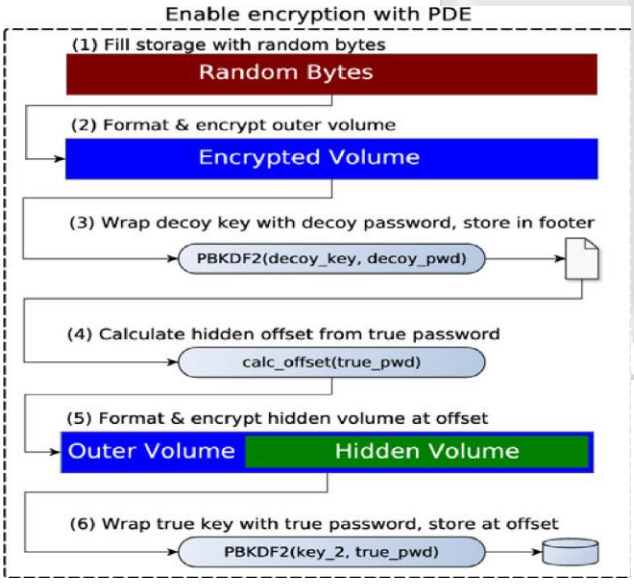


Figure 2.9: Mobiflage Initialization Process

Source: (Skillen and Mannan, 2014)

2.8.5 Mobile Data Self Encryption Scheme for Data Security

In a research paper by Chen and Ku (2009) on a self-encryption scheme for data security in mobile devices, they proposed a novel data encryption scheme to address the issue of data security. Looking at the data as a binary bitstream, the self-encryption scheme would generate a keystream by randomly extracting bits from the stream. The length of the keystream is dependent on the user security requirement. The bitstream is encrypted and the ciphertext stored on the mobile device, whereas the keystream is stored separately, making it computationally complex to recover the original data from the ciphertext. Gurdhalkar et al. (2018) further developed a system to provide security to the data on mobile. In the self-encryption system, the data would be browsed or added from a micro SD, encrypt the file, and store the encrypted file on the Micro SD or the internal mobile storage, and the encryption key would then be uploaded to the server for storage. The system utilised the AES 128 Algorithm for encryption of the data.

2.9 Summary of Gaps with Existing Solutions

The tools discussed above are available to provide confidentiality of sensitive data captured on the mobile device. However, the tools still leave a gap in which the files in the device are vulnerable to threats in the android operating system being exploited. The encryption keys stored in the device's internal memory are susceptible to being accessed by intruders or leaked by malware as they reside on the libraries in the same execution environment as the malware and spyware, which often infect the device. Rooted devices are more often at risk in this case.

Some solutions have limiting access to the users. For example, the Samsung Knox is only available on Samsung model devices and high-end devices, limiting other low-level Samsung phones and other models of android devices. For business resiliency, data should always be available to enable continuity; this needs to apply to users' mobile device data. In the event of loss of device for users, the data is often lost. Thus, this data can only be available for tools that store the data on the cloud. Unlike all the other solutions provided, CryptAll gets to keep a copy of its encrypted files in an online database. This makes it easier for users to retrieve the files when there is a need. However, the retrieval process is consuming since the files can’t be viewed directly from the tool. Instead, it should be decrypted and then saved on the phone drive for viewing. The users should then delete the file once again after viewing.

2.10 Conceptual Framework

According to Jabareen (2009), a conceptual framework represents a network of interlinked concepts that offer a comprehensive understanding of a situation. It creates an illustration of the concepts derived from the research to provide a conceptual distinction while organising the ideas to depict the relationships between the system elements, as shown in figure 2.10.

Figure 2.9 is divided into three phases: inputs, processes, and outputs of the proposed system. The inputs are independent variables, mainly the sensitive data in the mobile devices that the tool depends on to accomplish its function. The inputs also include the user login credentials and encryption key, determining the user's access to the system.

In the second phase, security is applied to the inputs to the system. The security process includes secure authentication for user logins and encryption keys to the system for access, encryption and decryption of the files added to the system and hashing the passwords and keys created for secure storage. The data added to the system and encrypted on the mobile device level is transmitted to the webserver for storage in the cloud.

The files uploaded to the server are encrypted, and the access passwords and encryption keys created by the user are encrypted using a password-based key derivation function and stored in hashed form. In addition, a salt function is added to the hash before storage to help strengthen the security of the encrypted passwords so that in the event of identical passwords created by users, they would not have the same hash during storage.

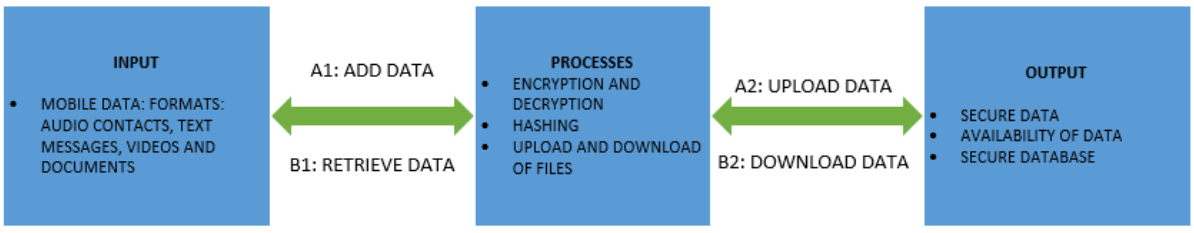


Figure 2.10: Conceptual Framework for Developed System

Source (Researcher)

Chapter 3: Methodology

3.1 Introduction

Prior to developing the mobile data encryption tool for securing and storing personal sensitive data remotely, extensive research needed to be undertaken to provide the foundation on which the system's design and development will be achieved. This chapter seeks to give the formal documentation for the phases and design methodology that the mobile data encryption tool was subjected to during its development life cycle to achieve the research objectives. Agile Software development methodology was adopted in this development.

3.2 Research approach for Objectives 1 and 2

As part of this study, descriptive research was considered to achieve the research objectives one and two, with a survey form on mobile data confidentiality sent to users who represented a sample of the population of mobile users. This was to collect information from the mobile users that would contribute to understanding the confidentiality of mobile data and habits concerning the storing of sensitive data on mobile devices.

Further research was carried out, with a broad reading of documents regarding mobile systems and security and discussed in the literature review to identify the common threats associated with the breach of confidentiality and availability of data stored in the android mobile devices. The common threats identified included memory corruption, privilege escalation, malware and spyware, and the loss of mobile devices. In the research, several solutions have been identified that are available for providing confidentiality of data. The tools and solutions reviewed include Samsung Knox, Mobiflage solution for plausible data deniability, Applock for efficient storage encryption, CryptAll and mobile self-encryption Solution. A review of the tools has been further discussed in the literature review.

3.3 Research approach for objectives 3 and 4

To achieve the objective three and four of the research, a mobile application tool for enhancing confidentiality of data was designed, developed, and further validated for the solution's effectiveness. This was accomplished by following a software methodology process, and for the design and development of the Mobile data encryption tool, agile software methodology was used.

Agile Software development life cycle methodology, as illustrated by figure 3.1, is an iterative methodology that breaks the developed product into small increments to minimise the amount of upfront planning and design. This methodology involved continuous planning, testing, and integration to ensure the delivery of a quality and secure mobile application tool. It offers a realistic approach to software development in an environment where technology and user needs are continuously changing, user-focused, more flexible in making changes and faster implementing it. (Schell Deborah, 2018).

The steps that were followed in the methodology included.

- Planning
- Requirement Analysis
- Design
- Development
- Testing

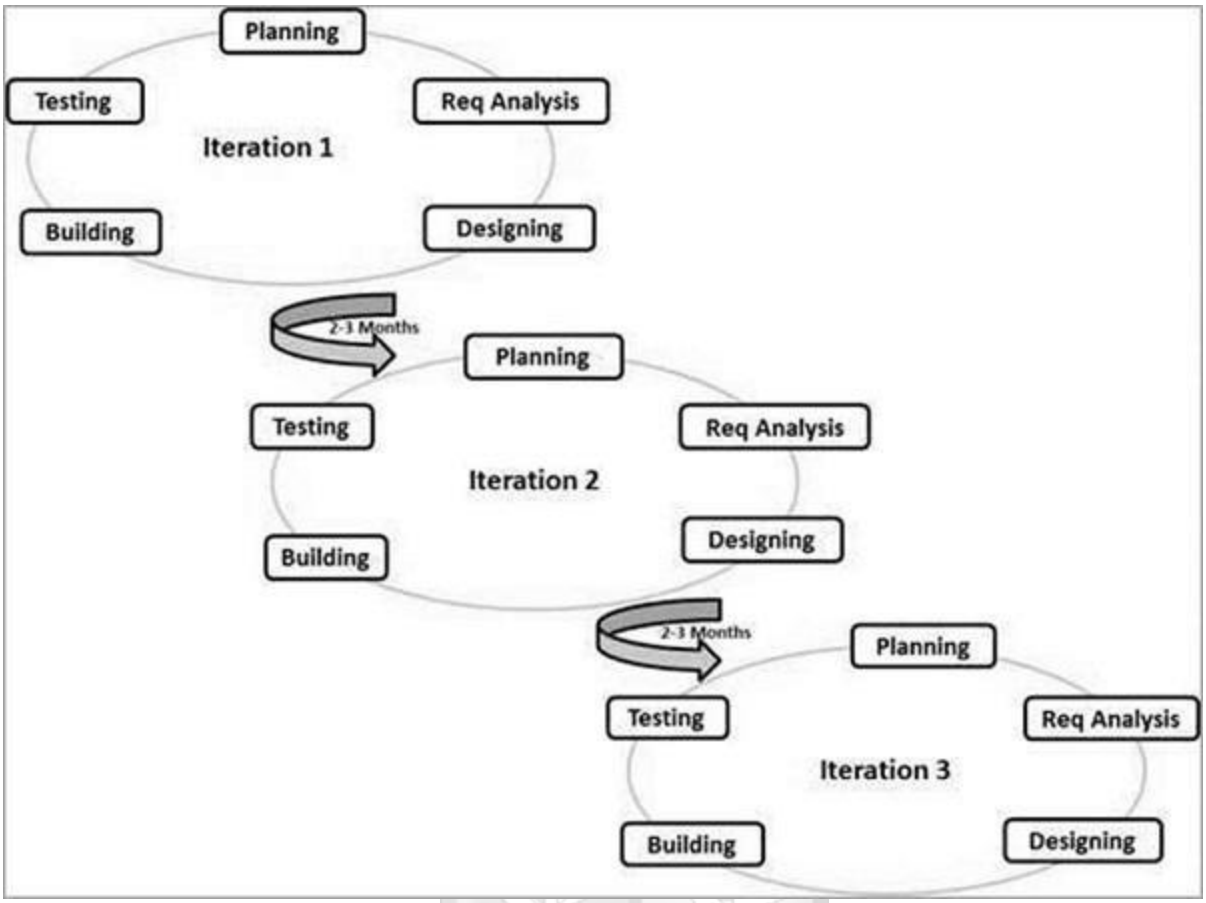


Figure 3.1 Agile Software development Methodology

(Source: Tutorialspoint, 2019)

3.3.1 Planning

Planning is the initial phase of agile methodology, which sought to outline and facilitate the process through which the development of the mobile data encryption system was undertaken. This phase entailed defining the mobile data encryption system's problem, objectives, functionality, and quality requirements. A plan was developed further to estimate the project scope and application requirements for the tool.

3.3.2 Requirement Analysis

This is the second phase of agile methodology that focuses on gathering the requirements for the developed tool. This phase was used to identify the requirements for the design and development of the mobile data encryption tool based on the analysis of the various gaps identified in the

existing solutions. The outcome of this phase was used to define the technical approaches that were followed to develop the tool successfully. With successive iterative, changes were made to the requirements of the tool to achieve the specific objectives defined. For the implementation of a successful system, it is paramount to apply a proper analysis of the data that was collected.

The various methods through which the requirement analysis achieved this was through;

3.3.2.1 Literature review

To further understand the area of study, a literature review was done focusing on the Android Operating System security and then on the platform's mobile data storage. This was achieved by reading much literature about the development of systems providing similar solutions and the scope of their functionalities. The goal was to obtain more knowledge on how the platforms are built and work. However, privacy being the focus, the literature review further focused on identifying the significant threats that exist in the Android mobile platform. In accomplishing the study's second objective, a review of existing solutions was done in the literature review to identify the gaps in the solutions available in providing data confidentiality for the mobile users on the android platform.

This was helpful in a way that:

• It helped in getting new ideas that one may not have thought about the research.

• It helped preview what security solutions others have used and derive their strengths and weaknesses.

• It helped give more insight into how similar systems had been developed previously and their various components.

• It eased the development process.

3.3.2.2 Observation and applications review

In this requirement analysis technique, I installed the available existing tools identified from the literature review that provide the solutions. Such available tools included the *AppLock, CrypAll, Mobiflage* and *Samsung Knox.* Their documentation was reviewed to understand more their

functionalities and discover the areas where improvement is needed. The documents reviewed included academic papers, online articles, and journals.

3.3.2.3 Questionnaire

A feasibility study was conducted using survey forms on mobile data privacy focused on confidentiality and user habits. This was done using an online questionnaire form created with the help of google forms and sent to a sample of the population of mobile users. The responses received were analysed using the google form analysis tool and represented using graphs and charts. See Appendix A.

3.3.3 Design

In this phase, requirements identified in the prerequisite stage of requirement analysis gets to be interpreted. The database schema has been defined, and Unified Modelling Language diagrams modelled to create a visual representation of the developed tool.

The design phase was vital as it guided the subsequent steps of the actual building of the system.

The analysis and design tools that were utilised for the design phase of the data and process include:

- Entity-relationship diagrams – This was used to model the database schema to show the relationships between the entities in the mobile data encryption tool during database design and development
- Use Case diagrams – This diagram was used to show how the mobile users interact with the mobile encryption tool.
- Sequence diagrams – This diagram gets to show how objects interact with one another and in the order of interaction.
- Class diagrams – This diagram was used to model the static view of the proposed tool by showing the system's classes, attributes, operations, and the relationship that exists among the defined objects.
- Wireframe diagram – This was used for sketching of the Android mobile application to give a visual representation of the structure and system functionality.

3.3.4 Development

In this phase, the actual development of a prototype of the mobile encryption tool was carried out. This includes the mobile application and database on the backend. This development was facilitated using the following tools:

3.3.4.1 Backend Development

MySQL was used as the backend tool for the database design and management of the mobile encryption tool and as well as establishing a connection between the server and mobile application device. The reason for using MySQL is that it is scalable and flexible as it handles deeply embedded applications and can handle large amounts of data. In addition to its scalability and flexibility, it allows for the customisation of requirements for specific functionalities. It also offers the implementation of solid security features with high performance to meet the demanding performance for requirements. (Oracle, 2019)

The Application is hosted on an online Apache HTTP server. This is because PHP is an Open-Source platform, independent, it supports all major web servers and databases, and it has multiple layers of security to prevent threats and malicious attacks

3.3.4.2 Mobile Application Tools for Programming

The developed tool is an Android mobile application, and It is implemented in a Java Environment. The source code is written in Java utilising the android classes. The JavaScript Object Notation (JSON) has been used to provide the interface between the Android application and the database. The reasons for choosing Android as the client application included availability of flexible software development kit, Android Development Tools availability and support from online developer communities.

3.3.5 Deployment

3.3.5.1 Implementation

For the successful implementation of this tool, a direct methodology was used. This involved installing the tool in smartphone devices running various Android versions and later trying to correct any upcoming problems and errors.

3.3.5.2 Testing

In this stage, system acceptance testing was conducted to test the system functionality of the tool regarding the system functional and non-functional requirements and the objectives specified.

Penetration testing on the tool was carried out to test the security features of the proposed tool being implemented to ensure the objectives have been met.

3.4 Summary

Research Methodology is core in the successful development of the mobile encryption tool. It ensured that the data obtained from literature and application review was accurately interpreted to come up with the requirement specifications of the developed mobile application. The iterative agile software development methodology ensured flexibility in developing a well-supported mobile application tool for encryption and availability of data that would effectively ensure confidentiality and availability of data in the android smartphone devices

Chapter 4: System Architecture and Design

4.1 Introduction

This chapter aims to cover the system design and architecture of the mobile sensitive data encryption tool, which satisfies the requirements that were discovered and gathered during the requirement analysis phase. The developed system allows android smartphone users to create accounts together with an encryption and decryption key, add confidential data from their phones to the system for secure storage. During the addition of the data to the system, the files are encrypted and stored in a secure location on the server. The users get to access the data from the mobile application tool with the use of a decryption key. The system has been developed using the agile methodology.

4.2 Functional Requirements

The functional requirements are the various functions and capabilities that the system must perform to achieve the objective of the user. The functional requirements for the developed mobile data encryption system include;

i. Account Registration: This function ensures that a mobile device user is registered into the system by setting up an account with a valid username, email, and password.
ii. Login to System: This function gets to grant the user access to the system by authenticating the provided email and password from the system.
iii. Creation of Cryptography Keys: This function enables users to create cryptographic keys that the system uses for encryption and decryption of data. The key is of a valid length as enforced in the system security policy.
iv. Add Data to the System: This function enables the users to access the data stored on their mobile device in different formats and select the required sensitive files that the user would need to be added to the system.
v. Encryption and Upload of data: This function enables the user to add a valid encryption key to encrypt the files selected by the user and upload them for storage in the cloud.
vi. Decryption and download of data: This function enables the user to access the files stored in the system. It will allow the user to select the file they would like to access, which are

in an encrypted format, enter a valid decryption key, and allow time to be decrypted. The user gets to download the file decrypted for storage on the mobile device.

vii. Logout from System: The user should be able to log out of the system to allow for disconnection of all operations in the system.

4.3 Non-Functional Requirements

They are requirements that describe the system operational capabilities and the constraints that enhance its capabilities to perform efficiently. These requirements do not affect the system's main functions; however, they are essential in the operation of the proposed system.

The following are the non-functional requirements that were identified in the requirement phase;

i. Security: The system should be able to address the concepts of confidentiality, integrity, and authentication. These concepts ensure that the information in the system would be accessed only by the authorised users, ensure that it has not been altered from its original state and be able to authenticate that the user is whom they say they are by using the encryption and decryption keys that are only known to them to access the data stored.

ii. Performance: The system should be able to respond to the various user interactions in the system within a short time.

iii. Reliability: The system should be able to perform its functions without failure and downtimes.

iv. Availability: The functionalities and services offered by the system should be available for use with all the required operations. It should also enable the users to access their data at any given time and from any given device that is compatible whenever it is needed.

v. Scalability: The system should allow for the addition of new features without affecting its services and performance.

vi. Compatibility: The mobile application should be able to run on the different available versions of the Android Operating system.

vii. Usability: The system should offer ease of use for all users across the board regardless of their technical knowledge and minimal steps to accomplish tasks.

4.4 System Architecture

System Architecture is a conceptual representation that describes the structure, behaviour, and views of a system. The developed system is based on a three-tier client-server architecture. These consists of the user interface layer, an application layer and the data storage layer, as shown in figure 4.1.

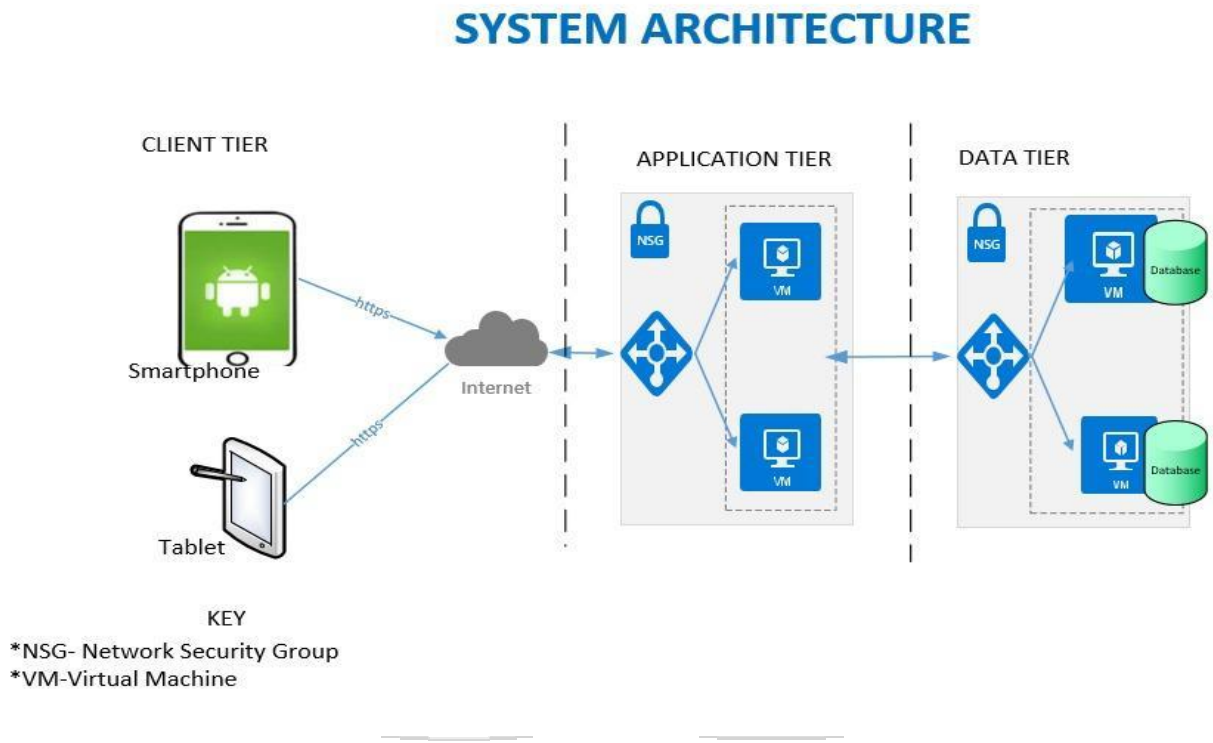


Figure 4.1: Three-Tier Client-Server Architecture

4.4.1 Layer 1/Tier 1: Client layer

The client layer is the presentation layer in the developed mobile data encryption tool and consists of the Android mobile application. The user interacts with the system through this layer. The application contains the interface through which the user performs the various functions on the tool. This function includes the account setup and login, creating a private encryption key used to encrypt data in the system, adding confidential data, and retrieving data to an android mobile device from the server. The types of data files supported by the system include Audio files, Videos, Images, Contacts, Text messages, and documents.

4.4.2 Layer 2/Tier 2: Application layer

The application layer is the middle tier between the client and the data server. The layer consists of the system logic that implements the access rules and data processing. In the tool, the middle tier consists of the transcoder, which performs the encryption and decryption functions to enhance data confidentiality. The encryptor validates the encryption key provided by the user against the hash for the key created by the user and stored in the server and proceeds to encrypt the file selected by the user and then transmits it for storage in the server. The decryptor functions so that the user gets to download a copy of the encrypted file to the device and then provide the decryption key, which is validated and utilised to decrypt the file to a form that is readable by the user. The decrypted file does not get saved on the device, and for the encrypted file downloaded, once the user logs out of the system, the file is automatically deleted from the mobile device's internal storage.

4.4.3 Layer 3/Tier 3: Data Layer

The third tier is the data server layer, and this is the resource manager that carries out the function of data storage. The data server is protected from direct access by the clients, and the application layer acts as a medium of exchange of the processed data between the server and the client. This layer is composed of MySQL relational database server that is protected from direct access by the users. The server hosts all data and files uploaded to the system and those required by the user to access the system. The files include audio, videos, text messages, pictures, contacts, and documents in an encrypted format. The user logins and encryption key are stored in an encrypted form in a database on the online server. Interaction to the data layer is only accessible through the application tier. The database is hosted in a virtual machine to provide redundancy and recovery for the system in enhancing its availability.

4.5 System Design

In this phase of Agile Methodology, the system development focused on the design structure of the mobile data encryption tool. It will entail the modelling of the conceptual structure and behaviour of the system, with the use of diagrams to logically represent the system and provide insight into the actual implementation of the system. The system design is supported by a use case diagram, a sequence diagram, a class diagram, and an entity-relationship diagram to model the database schema.

4.5.1 Use Case Diagram

A use case is a behavioural diagram used to show the different ways in which the user is represented as an actor to interact with the various system functionalities. Figure 4.2 illustrates the actor who is the smartphone device owner, interactions with the mobile data encryption tool and the relationships that exist with the use case involved.

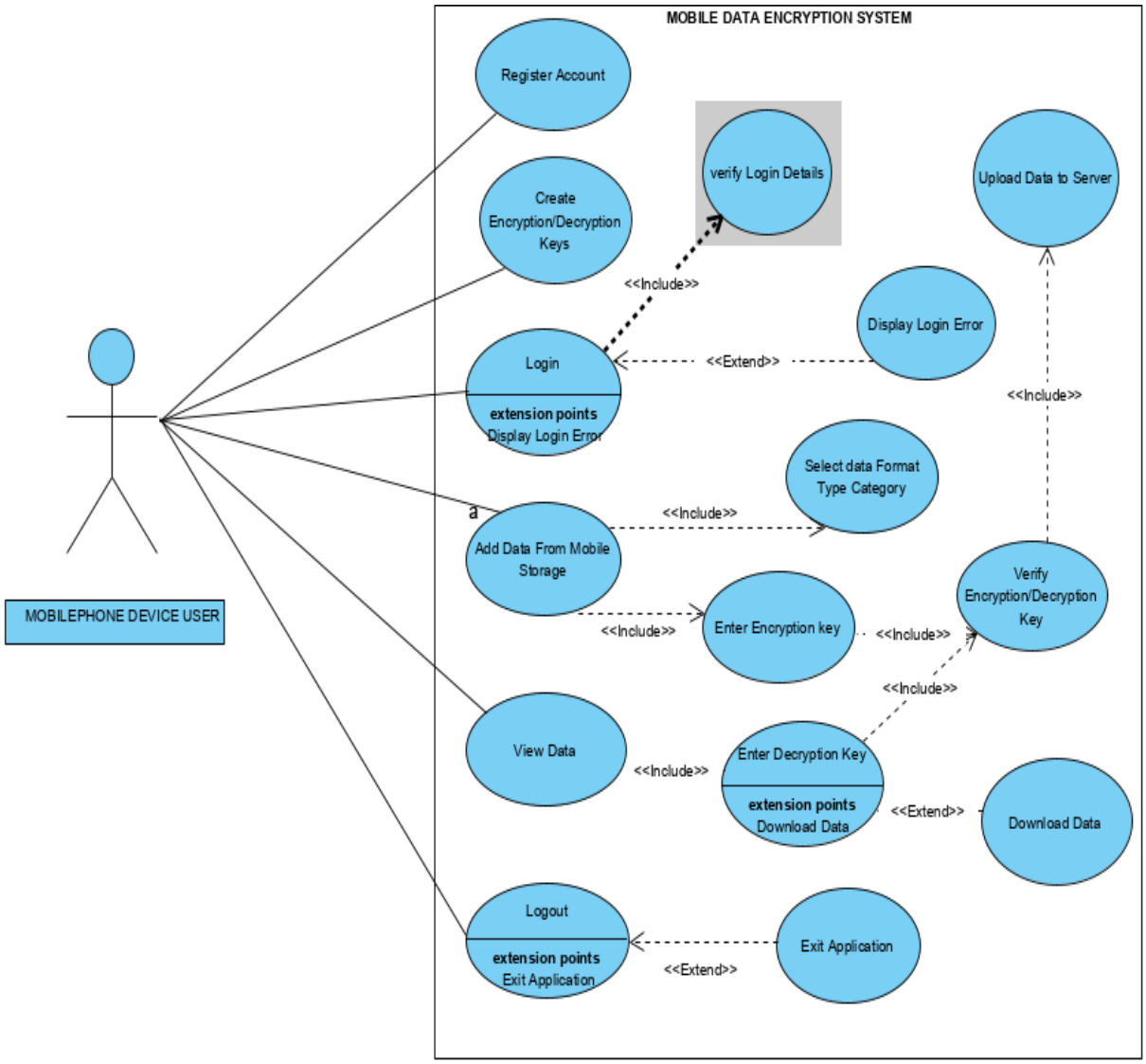


Figure 4.2: Use Case Diagram

4.5.2 Use Case Descriptions

Table 4. 1: Account Register Use Case Description

Use Case Name	Account Registration
Use case Description	This use case allows the user to create a user account to the mobile encryption tool system, and it is triggered by the user sending an account registration request. They provide the personal information required to the system, and they are saved in the system.
Primary Actor	Mobile Phone User
Stakeholder	Mobile Owner
Preconditions	The user should provide valid information for the creation of an account
Post Conditions	New Account successfully registered. Unique Id for the user is generated and saved in a database for access during the login process.
Main Flows	The user provides a unique user ID. The user provides a valid email address. The user provides a valid password. The user submits information to the system
Alternate Flows	Invalid email address, username or password provided; system displays an error “invalid email/password” message. Missing email address, username or id, system display an error message prompting for a valid email or username to be entered.

Table 4. 2: Login Use case Description

Use Case Name	Login
Use case Description	This use case allows the user to login into the mobile decryption tool system to be able to access the functionalities being offered by the system. To login to the system, all the users need to enter a valid email address and password in which upon successfully authentication of the credentials entered, the users will be redirected to the homepage where they get to access the functionalities provided by the system
Primary Actor	Mobile Phone User
Stakeholder	Mobile Owner
Preconditions	The user should provide a valid account already created.
Post Conditions	Successful authentication of the credentials provided will direct the user to the homepage.
Main Flows	The user enters the email address. The user provides a password. The system validates the account details provided. The system authenticates the user and redirects to the homepage.
Alternate Flows	Invalid email address, username or password provided, system displays an error "invalid email/password" message Missing email address, username or id, system display an error message prompting for a valid email or username to be entered.

Table 4. 3: Creation of Encryption/Decryption Key Description

Use Case Name	Create encryption and Decryption keys
Use case Description	This use case allows the user to create a cryptography key that will be used by the system to perform the function of encryption and decryption of data. The system uses the symmetric algorithm AES 256 in accomplishing its task; thus, only a single key will be created for use in both encryption and decryption.
Primary Actor	Mobile Phone User
Stakeholder	Mobile Owner
Pre-conditions	The user should provide a valid account already created. The user should provide a key with a valid length as enforced in the system policy.
Post Conditions	Successful creation of cryptography key. Cryptography key saved in an encrypted format
Main Flows	The user enters the cryptography key. The users re-enter the cryptography key. The system validates the length of the key as it is entered. The user submits a key of valid length to the system. The key is successfully created and stored in the system in an encrypted format.
Alternate Flows	The length of the key is invalid, and the submit button is inactive.

Table 4. 4: Encrypt Data Use Case Description

Use Case Name	Encrypt & Upload Data
Use case Description	This use case allows the user to add confidential data in various formats to the system for storage to the server in the cloud in an encrypted format. The use case is triggered by the need to add data to the system.
Primary Actor	Mobile Phone User
Stakeholder	Mobile Owner
Preconditions	The user should be logged in to the system. The user should have valid cryptography keys created. User should have data available on the mobile device storage.
Post Conditions	Successful encryption of data and uploaded to the cloud server for storage
Main Flows	The user selects the type of data category to add to the system. The user selects the files needed to be stored from the mobile storage The users enter the encryption key. The system validates the encryption key. The system encrypts the files added to the system. The system uploads the encrypted files to the cloud for storage.
Alternate Flows	Invalid encryption key provided, the system displays an error “invalid email/password” message

Table 4. 5: Table Decryption of Data Use Case Description

Use Case Name	Decrypt View and Download Data
Use case Description	This use case enables the user to be able to access the data that already stored in the server. The users get to select the file they need to access and add a decryption key that will decrypt the file to a format that is readable by the user. This use case is triggered by the need for the user to access and view the data that is in the system.
Primary Actor	Mobile Phone User
Stakeholder	Mobile Owner
Preconditions	The user should be logged in to the system. The user should have a valid decryption key. The user should have data encrypted and stored in the system.
Post Conditions	Successful decryption of file and ready for download.
Main Flows	The user selects the file to access The user enters the decryption key. The system validates the decryption key provided. The system downloads a shadow file to the device and decrypts it. The user can view the file and ready to save it to mobile storage.
Alternate Flows	Invalid decryption key provided, the system displays an error “invalid key” message

4.5.3 Sequence Diagram

A Sequence Diagram is used to show the interaction between objects in the system arranged in sequential order. Figure 4.3 shows the interactions between the users and the three different entities in the mobile encryption tool. The three entities are the mobile user interface, authorization server and content server. The diagram gets to show the different messages that were exchanged in sequential order, and the responses received.

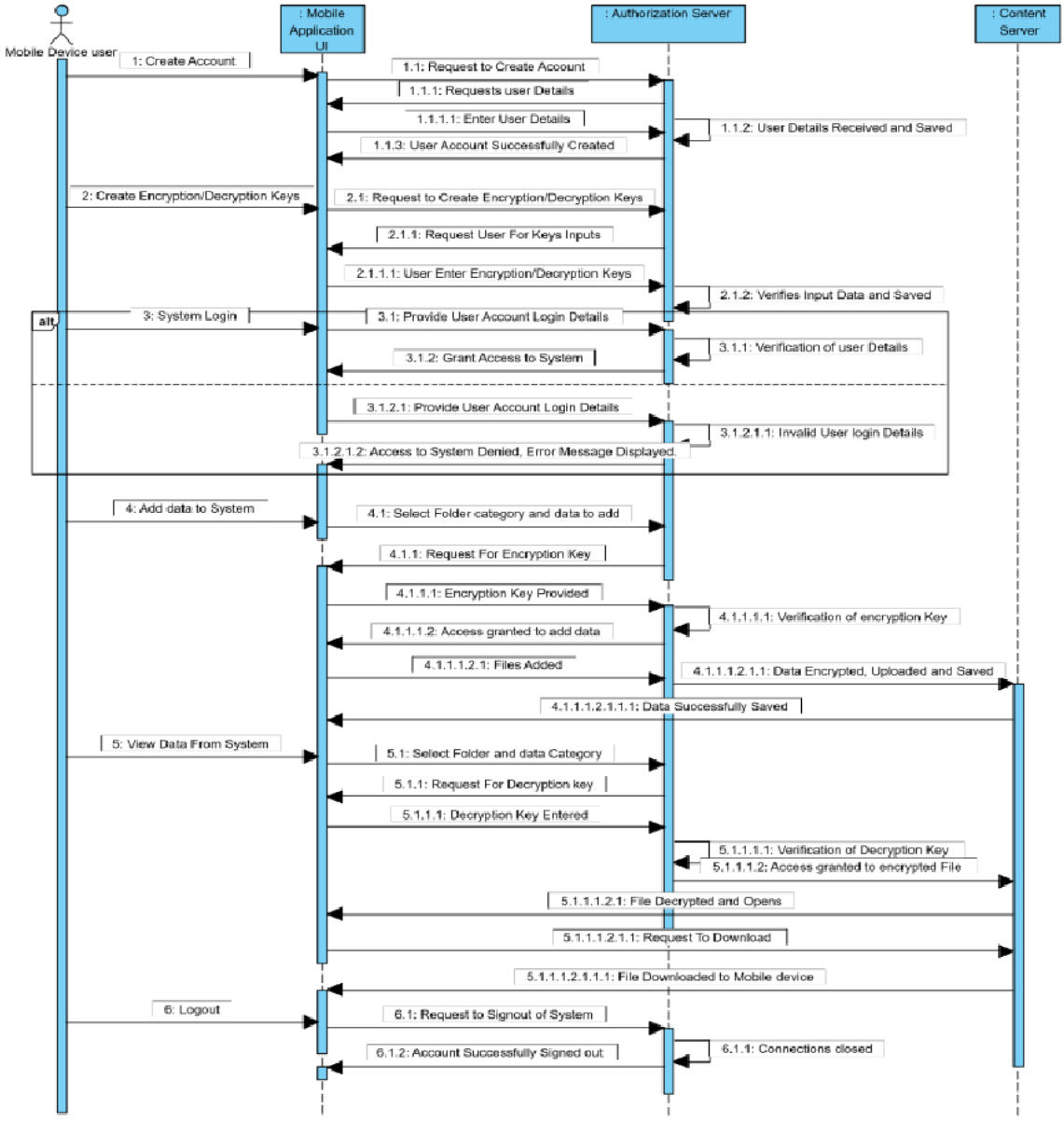


Figure 4.3: Sequence Diagram for Mobile Data Encryption Tool

4.5.4 Class Diagram

A class diagram is used to describe the system structure by showing the system classes, the attributes and operations of these classes. It also shows the relationships that exist among the objects. The figure 4.4 shows the class diagram for the Mobile data cryptography tool.

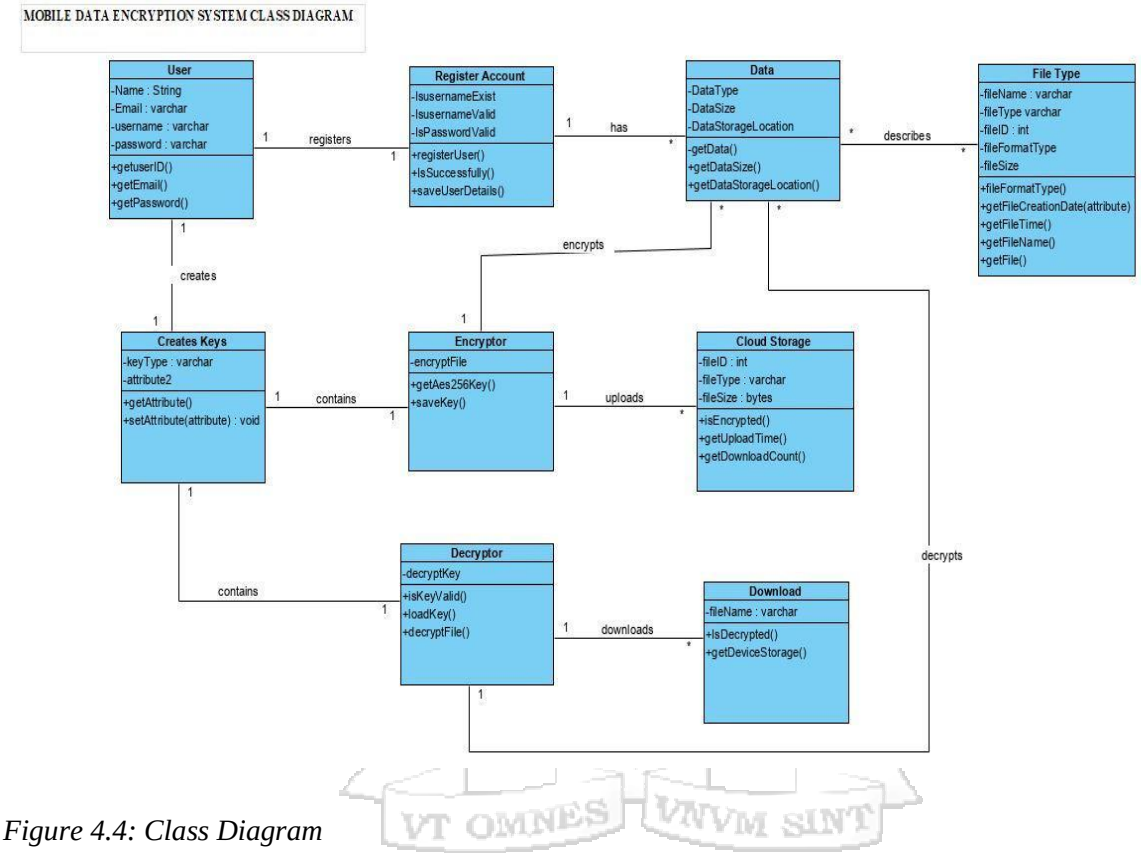


Figure 4.4: Class Diagram

4.5.5 Database Schema

A database schema represents the relationship between the objects and information in the database. The database design for the mobile data encryption tool has been depicted in figure 4.5 with an entity-relationship diagram. It consists of three tables that represent the main entities in the system. The tables are users, files and encrypt. Each table has a primary key that provides a unique identifier for the table and is set to auto-increment and generates a unique number whenever a new record is inserted into the table.

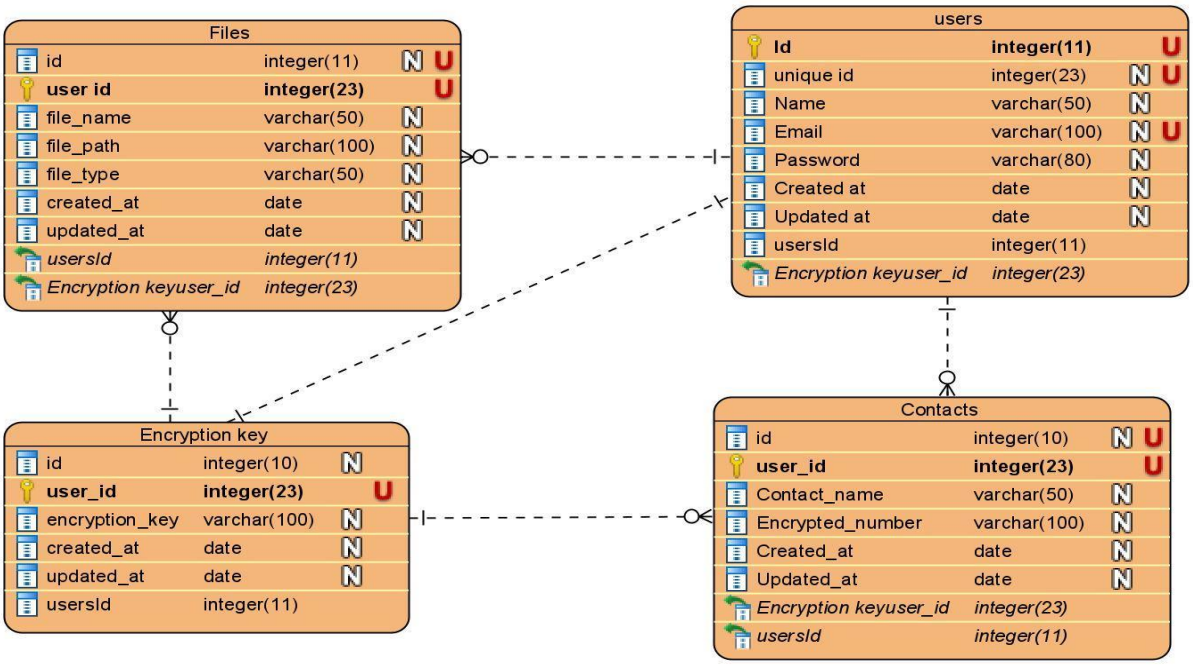


Figure 4.5: Entity Relationship Diagram

The tables below describe the main attributes of the tables shown in the entity-relationship diagram.

4.5.5.1 User Table Description

The table below contains information provided by the user to the system for the successful setup of their account.

Table 4. 6: User Table

Field	Data Type	Index	Notes
Id	int	Primary Key,	Auto increments with every new record inserted
Unique_id	varchar	Foreign key	Generates a unique id for the account created

Name	varchar		Captures the username of the user account
Email	varchar		
Encrypted_password	varchar		Password hashed with salt and stored in an encrypted format
Created_at	DateTime		
Modified_at	datetime		

4.5.5.2 Encrypt Table Description

The table below holds the cryptographic key created by the user for use in the system for encrypting and decryption of data

Table 4. 7: Encrypt Table

Field	Data Type	Index	Notes
Id	int	Primary Key	Auto
User_id	varchar	Foreign Key	A unique id generated by the system for each key created that is used for retrieval of data from the system
Encrypted_key	varchar		The encryption key is hashed with salt and stored in an encrypted format

4.5.5.3 Files Table Description

The table below is used for the storage of data that has been uploaded to the server.

Table 4. 8: Files Table

Field	Data Type	Index	Notes
Id	int	Primary Key,	Autoincrement key with every new record created
User_id	varchar	Foreign Key	id generated for each key that is used for retrieval of files from the system
File_name	varchar		
File_type	string		
File_ext	varchar		
File_size	int		
Updated_at	datetime		

4.5.6 Network Design

The network design topology implemented in the system is the server-based network design. The client connects the smartphone device that contains the application to the internet for connection to the server and transmission of data for storage. The server is hosted online, providing centralized storage for the user files and data. The requests sent to and from the server for storage and access of the data are transmitted and responded to over the HTTPS protocol to avoid interception by hackers.

4.5.7: Security Design

In the security design, the system has implemented access controls such as user logins, authentication, and encryption to provide confidentiality for the user by denying access to unauthorized users. The user logins and private encryption keys are created by the user during sign up to the system and are transmitted to the server over a secure HTTPS channel and stored in the server in an encrypted format. The sensitive personal data added to the system is first encrypted at the device level using the private keys known to the user, using the encryption algorithm AES 256 and transmitted to the server for secure storage.

4.5.8 Wireframes Diagrams

A wireframe diagram is a sketch that is used to give a visual representation of the structure and functionalities of a system. The figure below shows a wireframe diagram for the proposed mobile data encryption system to provide an overview of the mobile application user interface design.

4.5.8.1 Account Registration

The figure 4.6 shows a user account registration screen, where the user gets to set up their account, the login page and the home interface that will be displayed once the user successfully registers and logins into the account.

Figure 4.6: Account Registration and Login

4.5.8.2 Cryptography Key Creation

Figure 4.7 shows the screen in which the user gets to create the encryption keys that will be used by the system for the encryption and decryption of the data.

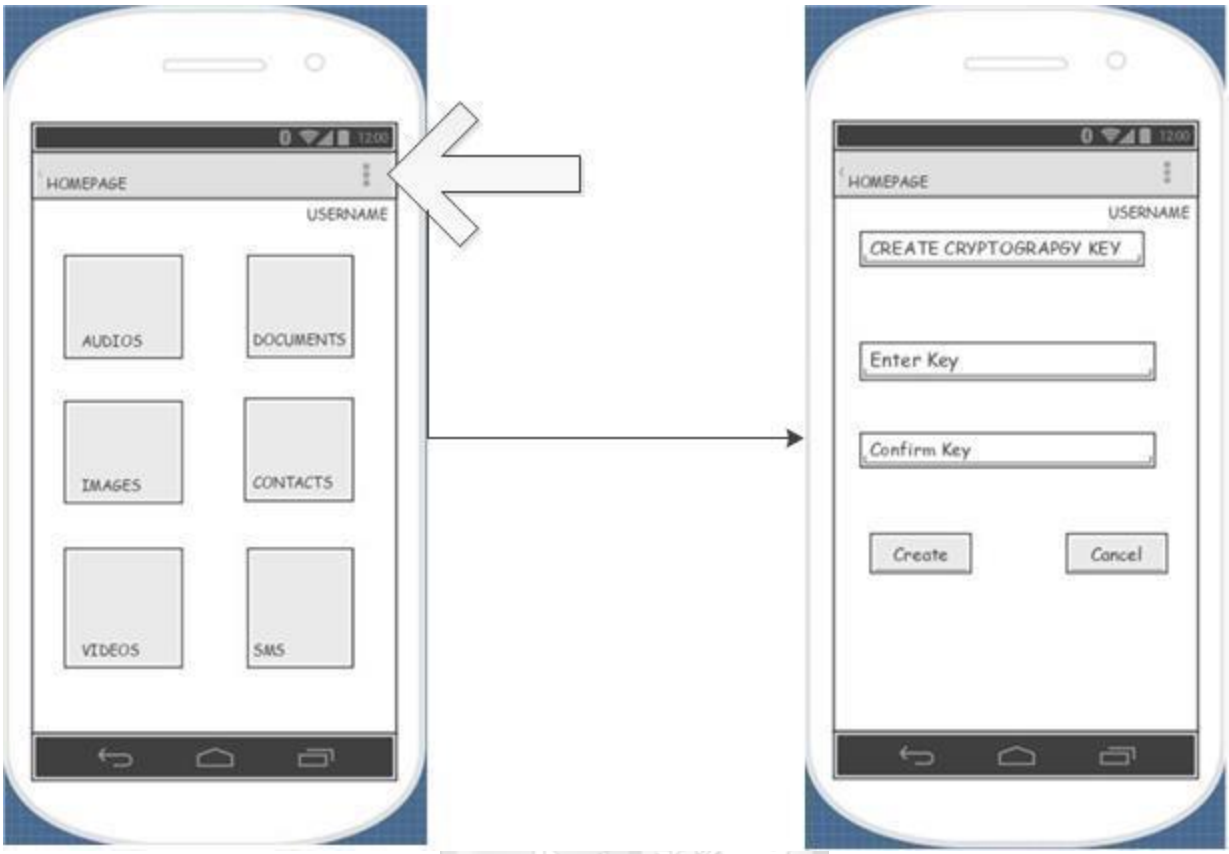


Figure 4.7: Cryptography Key Creation

4.5.8.3 Encryption of Data

Figure 4.8 shows the screen on which the user will be able to add data by selecting from among the data contained in the mobile storage and enter a valid encryption key for the data to be encrypted and automatically uploaded to the cloud server for storage.

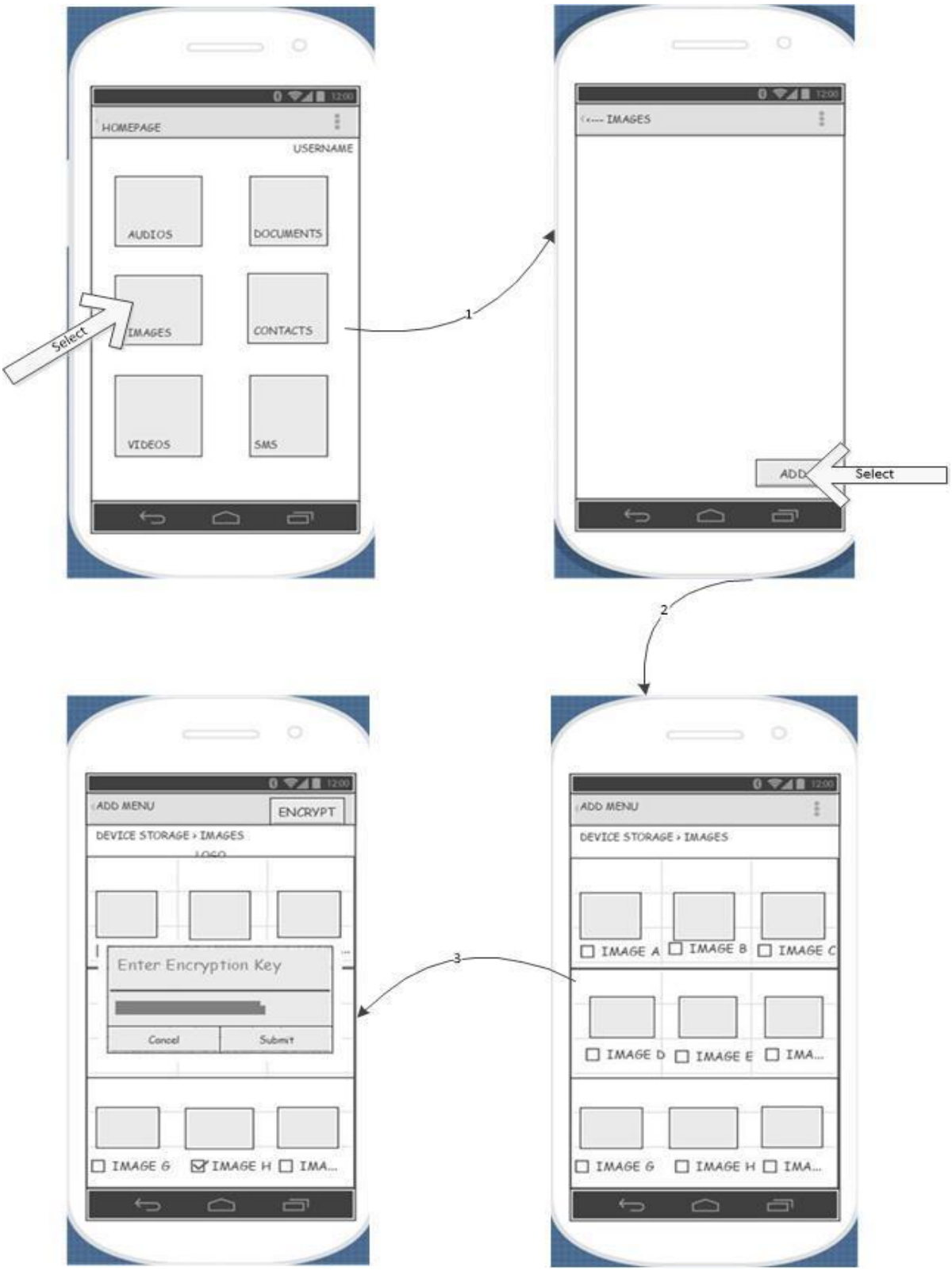


Figure 4.8: Encryption of Data

4.5.8.4 Decryption of Data

This is the interface from which the user gets to decrypt and view the data that is already stored in the server. The user enters a valid decryption key, and the data is automatically decrypted by the system for the user to access it in a readable format. Figure 4.9 show the screen interface.

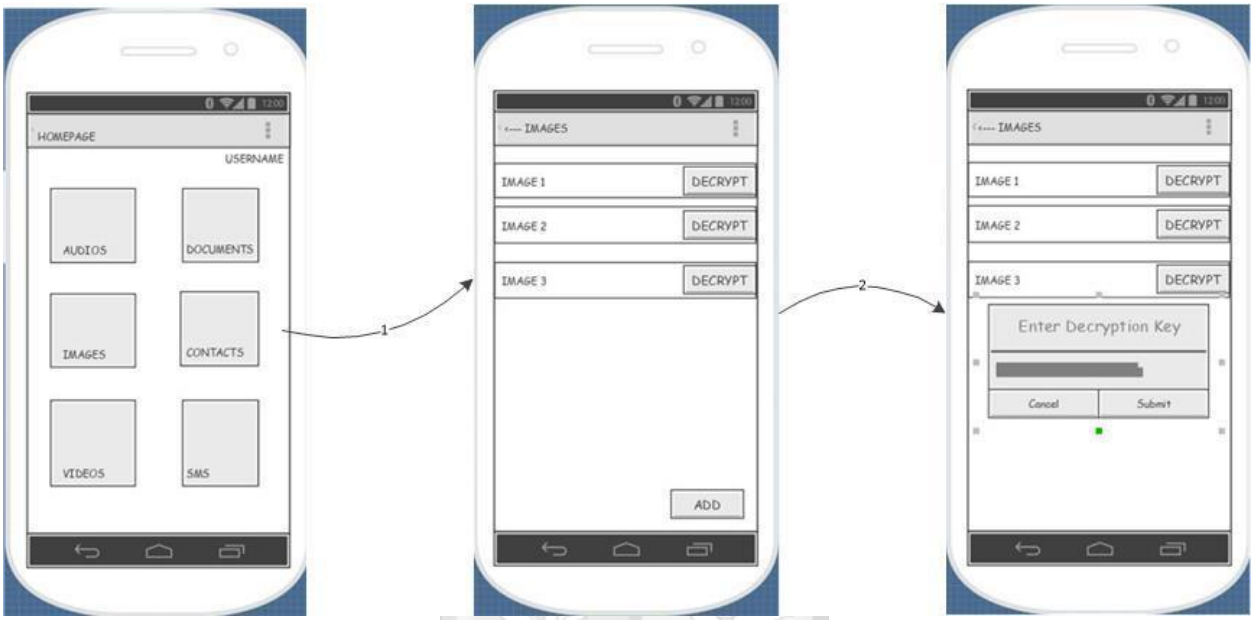


Figure 4.9: Decryption of Data

Chapter 5: System Implementation and Testing

5.1 Introduction

This chapter focuses on implementing the mobile data encryption tool, highlighting the significant functionalities and tests carried out to validate the tool. This section includes screenshots of important user interfaces and the tests carried out.

5.2 Implementation Environment

5.2.1 Hardware requirements

The mobile data encryption tool development was done by a computer running a 64-bit version of windows and tested on an android mobile device with the following hardware requirements.

1. The processor was a core i5 dual-core with 2.4GHz speed.
2. RAM of 8GB
3. Hard drive of 320GB
4. Android Device with Android OS version 7 and above.

5.2.2 Software Requirements

The system was divided into two modules, the backend and the frontend modules. To support the development of these two modules, I utilized the following tools.

Backend development

The proposed mobile data encryption tool's backend system development and implementation used the following tools:

1. PHP v7.1.1.1.
2. MySQL v5.6.43

Programming Tools

The development of the mobile application used the following.

1. Android Studio
2. Java 8
3. Gradle 3.6.1

5.3 System Modules

The main components of the system modules are as shown below.

5.3.1 Account Creation and Login

On installing the application to an android mobile device, a new user has to create an account by providing valid details as requested and submitting them. Once an account has been set up, the user can log in to the system by providing a valid email address and password, which would need to be authenticated by the system before being granted access. Figure 5.1 shows an account registration screen and the login screen for the mobile application.

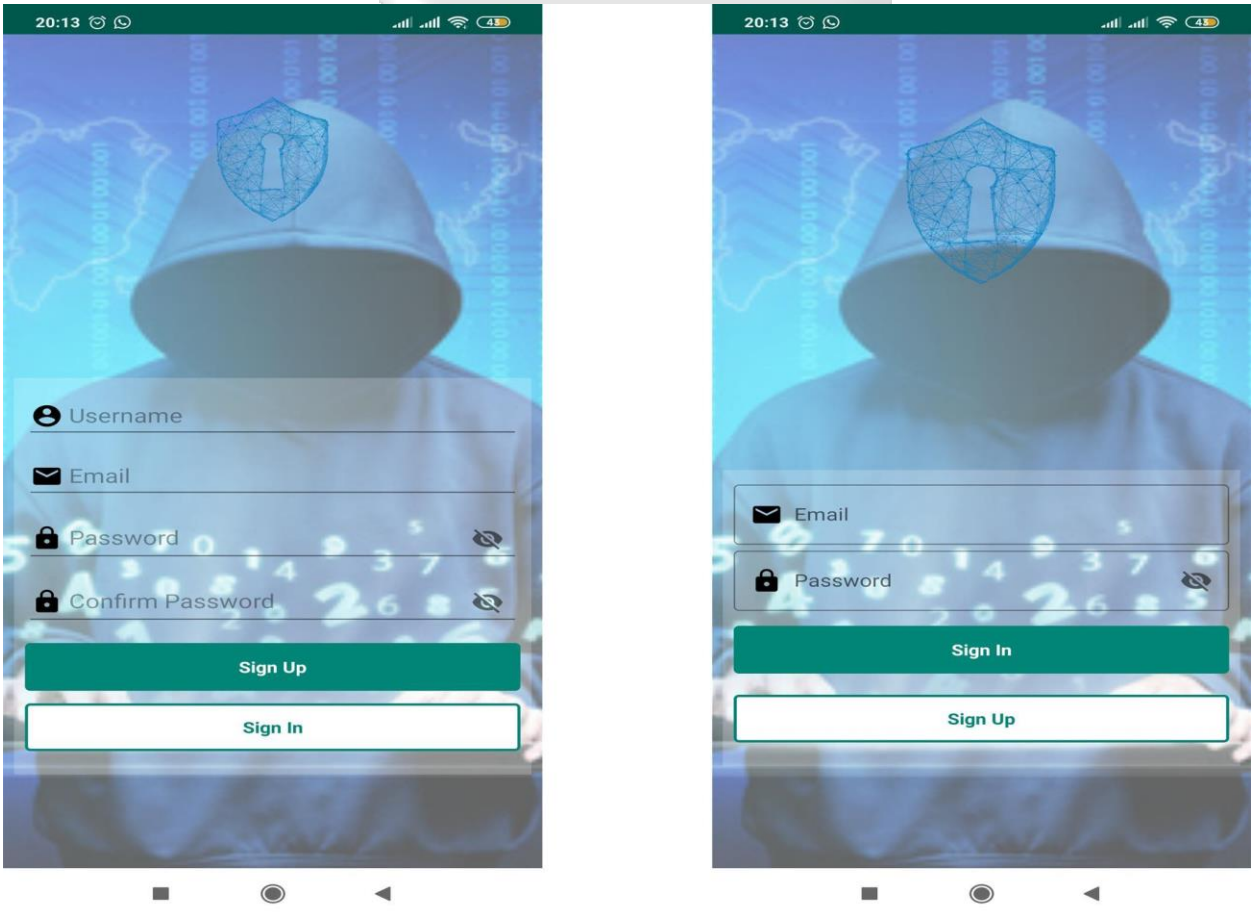


Figure 5. 1: Account Registration and Login Screens

5.3.2 Create Cryptography Key

On the first-time successful login to the screen, the user will be prompted for the creation of cryptography keys that would be utilized for encryption and decryption of data. The keys need to be of valid length for the system to accept. Figure 5.2 shows a cryptography key creation screen for the mobile application.

```
public static byte[] getKey(String password) throws CryptoException {
    try {
        byte[] salt = "!a3edr45!a3edr45".getBytes(StandardCharsets.UTF_8);
        SecretKeyFactory secKeyFactory = SecretKeyFactory.getInstance("PBKDF2withHmacSHA1");
        KeySpec spec = new PBEKeySpec(password.toCharArray(), salt, iterationCount: 156017, keyLength: 128);
        SecretKey pbeSecretKey = secKeyFactory.generateSecret(spec);
        return pbeSecretKey.getEncoded();

    } catch (NoSuchAlgorithmException | InvalidKeySpecException ex) {
        throw new CryptoException("Error encrypting/decrypting file", ex);
    }

}
```

Figure 5. 2: Creation of Cryptography key

5.3.3 Home Screen

After successful login to the system, the user will be redirected to the home page of the mobile application. The home page screen is shown in figure 5.3. The home page screen contains the categories of data from which the user navigates to add data to the system.

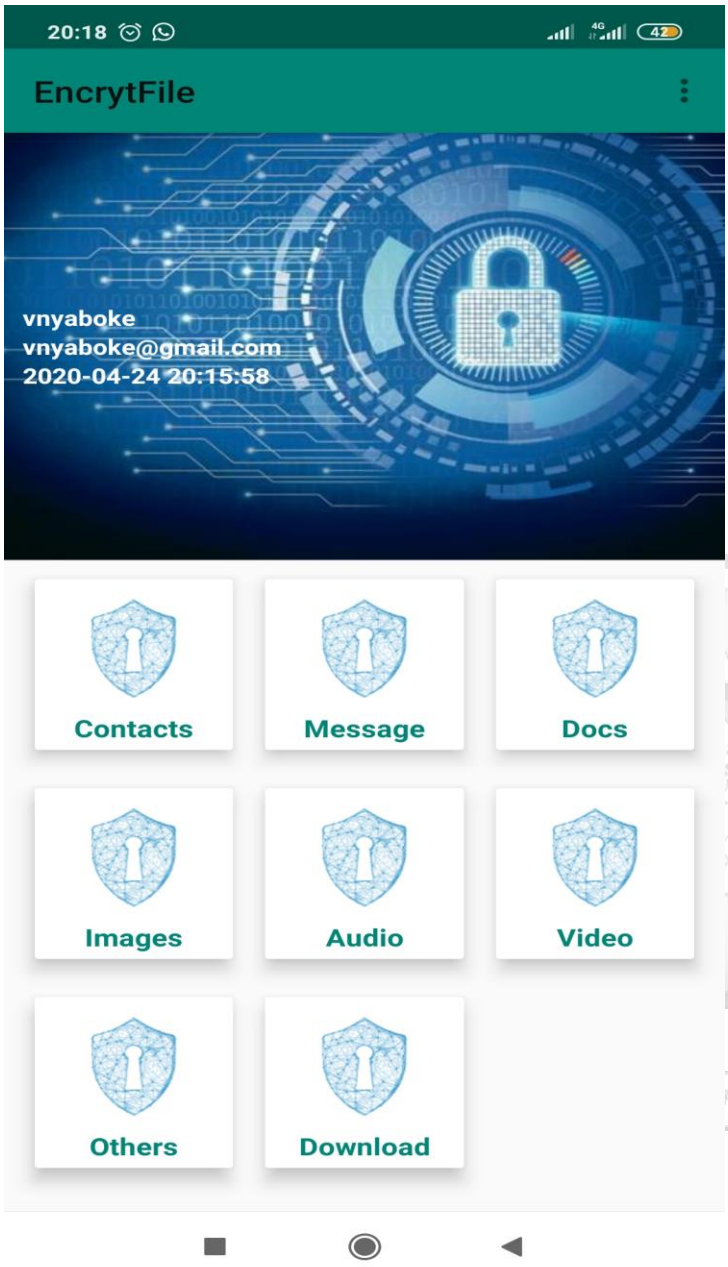


Figure 5. 3: Home Screen Screen

5.3.4 Encrypting data

Data will be added from the mobile device storage. Figures 5.4 shows a screen from the mobile application interface for encryption of data process. Once the user has selected the files that need to be added to the system, the user will be prompted for an encryption key that will be validated by the system before the encryption of the file can begin. The figure 5.5 shows the source code for

key validation before once the user inputs key. The data will then automatically upload to the cloud for storage in the database in an encrypted format as shown by figure 5.6 source code on the function for data upload once the file has been encrypted.

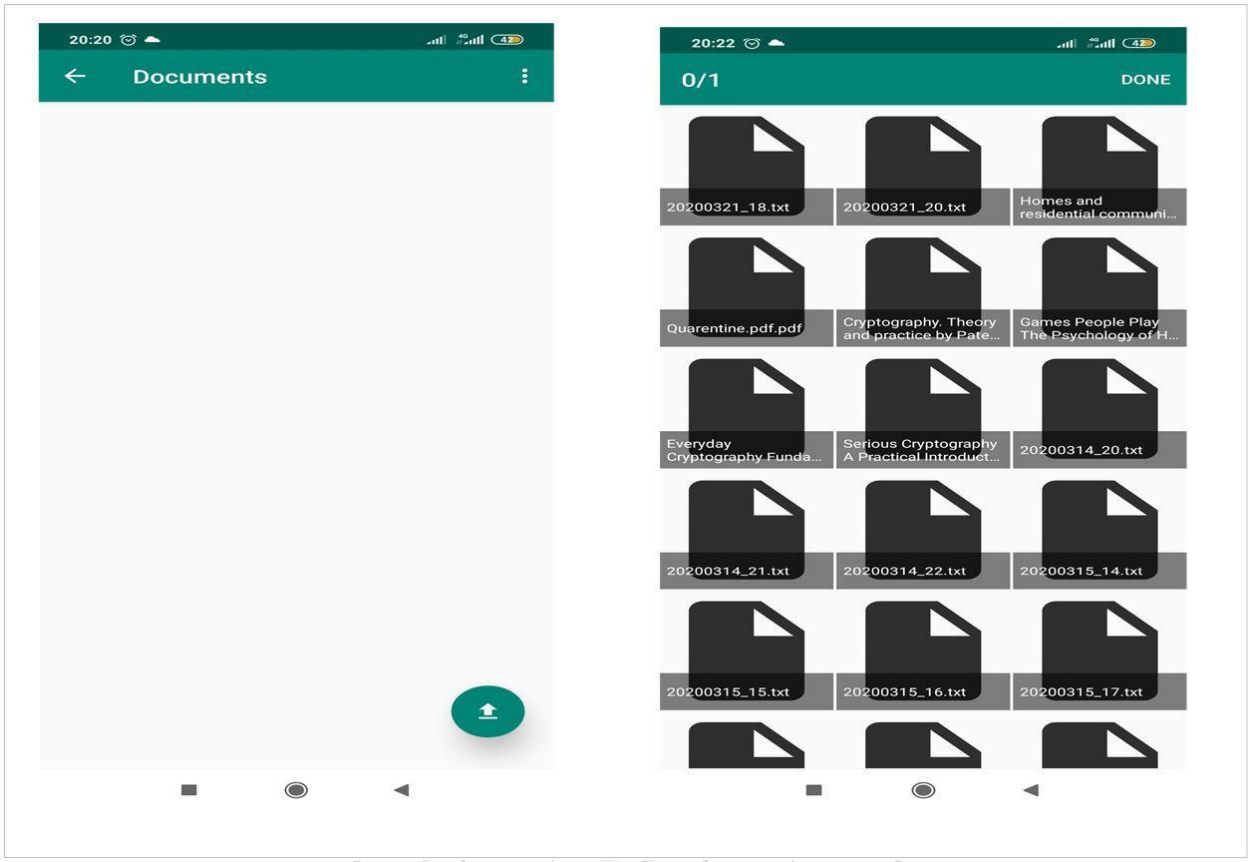


Figure 5. 4: Add data from Mobile Storage Screen

```java
public static void encrypt(byte[] key, File inputFile, CryptoListener listener)
        throws CryptoException {
    doCrypto(Cipher.ENCRYPT_MODE, key, inputFile, listener);
}

public static String encrypt(byte[] key, String phoneNumber)
        throws CryptoException {
    try {
        Key secretKey = new SecretKeySpec(key, ALGORITHM);
        @SuppressLint("GetInstance") Cipher cipher = Cipher.getInstance(TRANSFORMATION);
        cipher.init(Cipher.ENCRYPT_MODE, secretKey);
        byte[] encryptedText = cipher.doFinal(phoneNumber.getBytes(StandardCharsets.UTF_8));
        return Base64.encodeToString(encryptedText, Base64.DEFAULT);
    } catch (InvalidKeyException | NoSuchPaddingException | NoSuchAlgorithmException | IllegalBlockSizeException | BadPaddingException e) {
        e.printStackTrace();
        throw new CryptoException("Error Encrypting Text",e);
    }

}
```

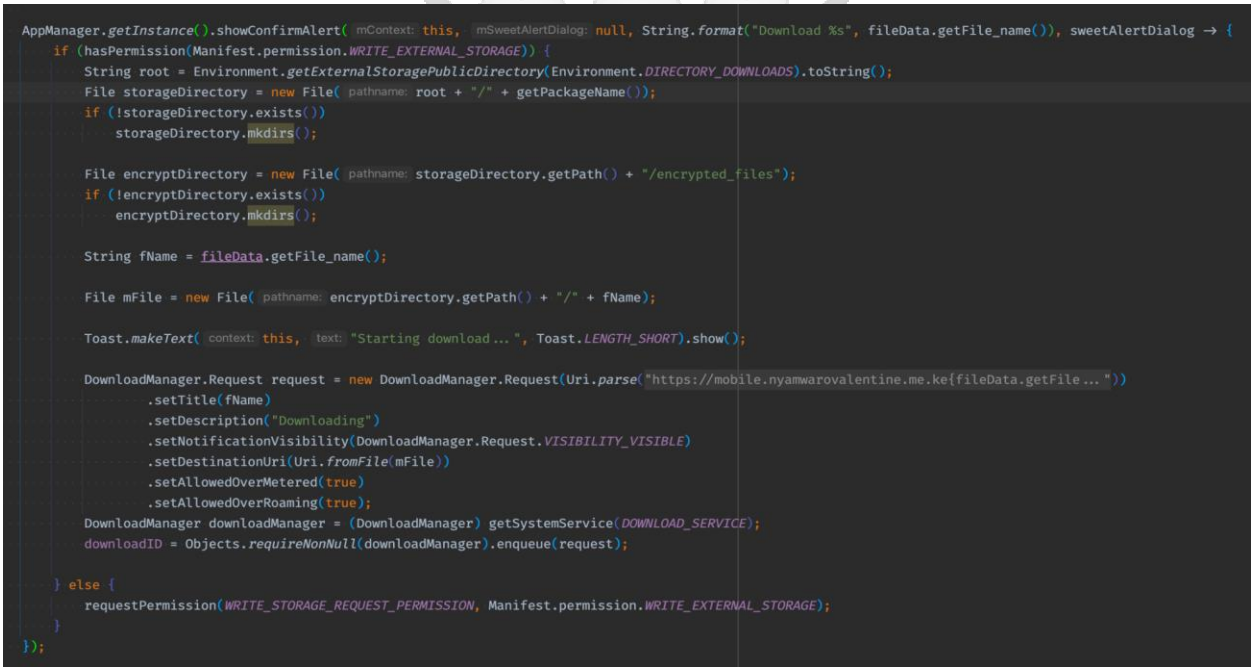
Figure 5. 5: Encryption Key Validation and File Encryption Source Code

```java
public void saveFileToServer(String fileName, File mFile, String fileType, Context mContext) {
    String userID = SessionHandler.getInstance().getSessionUserID(mContext);
    Toast.makeText(mContext, String.format("Uploading %s started...", fileName), Toast.LENGTH_SHORT).show();
    AsyncTask.execute(() -> {

        OkHttpClient client = new OkHttpClient().newBuilder()
                .connectTimeout( timeout: 5, TimeUnit.MINUTES) // connect timeout
                .writeTimeout( timeout: 5, TimeUnit.MINUTES) // write timeout
                .readTimeout( timeout: 5, TimeUnit.MINUTES) // read timeout
                .protocols(Collections.singletonList(Protocol.HTTP_1_1))
                .retryOnConnectionFailure(true)
                .build();

        RequestBody body = new MultipartBody.Builder().setType(MultipartBody.FORM)
                .addFormDataPart( name: "api",  value: "upload_file")
                .addFormDataPart( name: "user", userID)
                .addFormDataPart( name: "name", fileName)
                .addFormDataPart( name: "file_type", fileType)
                .addFormDataPart( name: "file", fileName, RequestBody.create(mFile, MediaType.parse(getMimeType(mFile))))
                .build();

        Request request = new Request.Builder()
                .url("https://mobile.nyamwarovalentine.me.ke/api.php")
                .method( method: "POST", body)
                .addHeader( name: "Content-Type",  value: "multipart/form-data")
                .addHeader( name: "Connection",  value: "close")
                .build();
        okhttp3.Response response;
        try {
            response = client.newCall(request).execute();
            GeneralResponse fileResponse = new Gson().fromJson(Objects.requireNonNull(response.body()).string().trim(), GeneralResponse.class);
            new Handler(Looper.getMainLooper()).post(() -> {
                if (fileResponse.IsSuccessFul()) {
                    getAllFiles(ContextHelper.getInstance().getContext());
                }
                Toast.makeText(ContextHelper.getInstance().getContext(), fileResponse.getMsg(), Toast.LENGTH_LONG).show();
            });
```

Figure 5. 6: Uploading File to Server Source Code

5.3.5 Decrypting data

The user gets to access the data that has been added to the system by decrypting the files. The user logins into the system select the category of data from which the file they would need to access is contained. From the select category, the file is downloaded by the user, and it will be available in the download category in the system. On the download category, a click on the file to view it will prompt for a decryption key to complete the process. Successfully validating the key will lead to the user accessing the file in a format that is readable. Figure 5.7 shows the source code implementing the functionality for download of data, while figure 5.8 shows the source code process for validation of keys to server and decryption of the file selected by the user



```
AppManager.getInstance().showConfirmAlert( mContext: this, mSweetAlertDialog: null, String.format("Download %s", fileData.getFile_name()), sweetAlertDialog → {
    if (hasPermission(Manifest.permission.WRITE_EXTERNAL_STORAGE)) {
        String root = Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS).toString();
        File storageDirectory = new File( pathname: root + "/" + getPackageName());
        if (!storageDirectory.exists())
            storageDirectory.mkdirs();

        File encryptDirectory = new File( pathname: storageDirectory.getPath() + "/encrypted_files");
        if (!encryptDirectory.exists())
            encryptDirectory.mkdirs();

        String fName = fileData.getFile_name();

        File mFile = new File( pathname: encryptDirectory.getPath() + "/" + fName);

        Toast.makeText( context: this, text: "Starting download ... ", Toast.LENGTH_SHORT).show();

        DownloadManager.Request request = new DownloadManager.Request(Uri.parse("https://mobile.nyamwarovalentine.me.ke{fileData.getFile ... "))
                .setTitle(fName)
                .setDescription("Downloading")
                .setNotificationVisibility(DownloadManager.Request.VISIBILITY_VISIBLE)
                .setDestinationUri(Uri.fromFile(mFile))
                .setAllowedOverMetered(true)
                .setAllowedOverRoaming(true);
        DownloadManager downloadManager = (DownloadManager) getSystemService(DOWNLOAD_SERVICE);
        downloadID = Objects.requireNonNull(downloadManager).enqueue(request);

    } else {
        requestPermission(WRITE_STORAGE_REQUEST_PERMISSION, Manifest.permission.WRITE_EXTERNAL_STORAGE);
    }
});
```

Figure 5.7: File Download Source Code

```java
public static void decrypt(byte[] key, File inputFile, CryptoListener listener)
        throws CryptoException {
    doCrypto(Cipher.DECRYPT_MODE, key, inputFile, listener);
}
public static String decrypt(byte[] key, String phoneNumber)
        throws CryptoException {
    try {
        Key secretKey = new SecretKeySpec(key, ALGORITHM);
        @SuppressLint("GetInstance") Cipher cipher = Cipher.getInstance(TRANSFORMATION);
        cipher.init(Cipher.DECRYPT_MODE, secretKey);
        byte[] result = cipher.doFinal(Base64.decode(phoneNumber, Base64.DEFAULT));
        return new String(result, StandardCharsets.UTF_8);
    } catch (NoSuchAlgorithmException | NoSuchPaddingException | InvalidKeyException | BadPaddingException | IllegalBlockSizeException e) {
        e.printStackTrace();
        throw new CryptoException("Error decrypting Text", e);
    }
}
private static void doCrypto(int cipherMode, byte[] key, File inputFile, CryptoListener listener) throws CryptoException {
    try {
        File outputFile = File.createTempFile( prefix: "Crypt", inputFile.getName());
        Key secretKey = new SecretKeySpec(key, ALGORITHM);
        @SuppressLint("GetInstance") Cipher cipher = Cipher.getInstance(TRANSFORMATION);
        cipher.init(cipherMode, secretKey);

        FileInputStream inputStream = new FileInputStream(inputFile);
        byte[] inputBytes = new byte[(int) inputFile.length()];
        inputStream.read(inputBytes);
        byte[] outputBytes = cipher.doFinal(inputBytes);

        FileOutputStream outputStream = new FileOutputStream(outputFile);
        outputStream.write(outputBytes);

        inputStream.close();
        outputStream.close();
        listener.onComplete(outputFile);
    } catch (NoSuchPaddingException | NoSuchAlgorithmException | InvalidKeyException | BadPaddingException | IllegalBlockSizeException | IOException ex) {
        throw new CryptoException("Error encrypting/decrypting file" + ex.getMessage(), ex);
    }
}
```

Figure 5. 8: File Decryption Source Code

5.3.6 Created User Account Details

The user account created to get to be stored on the cloud server database. Figure 5.9 shows the table in which the user accounts credentials are stored once a new account has been created. The new record created is identified with a unique user id that is randomly generated. The username and email provided by the user are stored against the user id generated and passwords which are in an encrypted format.

Showing rows 0 - 21 (22 total, Query took 0.0002 seconds.)

```
SELECT * FROM `users`
```

Profiling [Edit inline] [Edit] [Explain

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	id	unique_id	name	email	encrypted_password	salt	created_at	updated_at
Edit Copy Delete	1	5e9d575578c6e4.37737036	victor	oyandovic@gmail.com	lldubvC/2ZyA3yY0YWkgNrQcC1M3OTQwN2E3NzU1	79407a7755	2020-04-20 11:03:33	NULL
Edit Copy Delete	2	5e9e039f1fd1c1.61842830	briee	mawindavic@gmail.com	ECH3ZugnYs2lY8uOf4HfSzaQRB05MTkzNzM1OTdh	919373597a	2020-04-20 23:18:39	NULL
Edit Copy Delete	3	5ea1ee15582ff7.62713086	briee	mawindavi@gmail.com	UluPe9xl4Y+6sw2UZBkF3rDOl6pjMGFhMjg5YjM2	c0aa289b36	2020-04-23 22:35:49	NULL
Edit Copy Delete	4	5ea1f1f753fb87.80340978	briee	mawindavid@gmail.com	5wJql0MkGa0W0MXiCYGsw9K+COVjZDU5ZjNhNjgz	cd59f3a683	2020-04-23 22:52:23	NULL
Edit Copy Delete	5	5ea1f580037aa1.18064358	briee	mawindavit@gmail.com	xX7CSF7WmRfXo4DD+GVdMLDz0G9iZmJhZGUzZTYz	bfbade3e63	2020-04-23 23:07:28	NULL
Edit Copy Delete	6	5ea1f5c9ce31e8.50463805	briee	mawindavil@gmail.com	5JSL+8+DgNOqtEAdvvYfk6L38k81NWU1YTA0ZTVi	55e5a04e5b	2020-04-23 23:08:41	NULL
Edit Copy Delete	7	5ea30e8dd70367.18686989	Black	123@gmail.com	+ipLSowxTbx9plLx3y5UY2nczdkzMzJiMjUxOTE2	332b251916	2020-04-24 19:06:37	NULL
Edit Copy Delete	8	5ea312e95d9ef8.64304547	Akumu	akumubrendah@gmail.com	wU7Q4AN2mDdi6x3LulDAP20wA081ZmlwYzFlNmU3	5fb0c1e6e7	2020-04-24 19:25:13	NULL
Edit Copy Delete	9	5ea31eced6eef1.57559982	vnyaboke	vnyaboke@gmail.com	9Fcx0E7ZtmZXn7asHIjO2FSe9Wl2NzFjOTZjMzZm	671c96c36f	2020-04-24 20:15:58	NULL
Edit Copy Delete	10	5ea31fff5d32c7.78670005	Briee	brendahakumu2@gmail.com	DB7GML9odYEhQ8qRKiMPRaR7B6hmNDI2MTgwZjY1	f426180f65	2020-04-24 20:21:03	NULL
Edit Copy Delete	11	5ebfb685d1e6b4.33773926	ValentineN	valentine.nyamwaro@strathmore.edu	iAliWtkIEvI3iExYK84gonHq2540YTg2ZTlIOTk3	4a86e9e997	2020-05-16 12:46:45	NULL
Edit Copy Delete	12	5ebfc7ab66b600.42650549	Briee	brendahakumu@gmail.com	L2x/MP65KFvzUnjvdrjvTPES1Q1mZWU0YjE5ZDQ1	fee4b19d45	2020-05-16 13:59:55	NULL
Edit Copy Delete	13	5ec616ebc95b96.13811796	vmaithya	vmaithyaa@gmail.com	73Aay6h5XZbxLzqf1TsbYbBO8R43MGU4ZTNkNWVm	70e8e3d5ef	2020-05-21 08:51:39	NULL
Edit Copy Delete	14	5ec6a73594ff56.97688218	Vale	valentinen@gmail.com	DBNv9tTxgEjBChQgzJ6j3UHaB+djYTVkYTNmNjRl	ca5da3f64e	2020-05-21 19:07:17	NULL
Edit Copy Delete	15	5ec7552b73ac54.69957486	Bonfic	nyamwaro95@gmail.com	xOwTPpnh57MAD4VJ0CO+uDOgQVU4NDdlM2I3YjMz	847e3b7b33	2020-05-22 07:29:31	NULL
Edit Copy Delete	16	5ecbf3d7380bb6.75420360	gonoar	alfred.gono@strathmore.edu	JGUXvPnDC8Fl4H+JGOs5Q2cTtP40ZWE3ODdjNDcw	4ea787c470	2020-05-25 19:35:35	NULL
Edit Copy Delete	17	5ed0d6191acb29.62476452	gloria kerubo	kerubogloria27@gmail.com	uPj5pzThSC2yK7o+VF9PWQdjG8VjNjRiMDM1M2Yx	c64b0353f1	2020-05-29 12:30:01	NULL
Edit Copy Delete	18	5ed0dcb9257cf8.04568770	oscarlito	karugaoscar@gmail.com	hYPji3kRrzqwCvlRIEObd2O5J3Q3ODlhNjBmNzJk	789a60f72d	2020-05-29 12:58:17	NULL
Edit Copy Delete	19	5ed0de652613a9.59475524	Valentine	valenyam@gmail.com	1kyl95JrP6fNo7eOYVMohxgi3VphM2FkMDlmM2Nj	a3ad09f3cc	2020-05-29 13:05:25	NULL
Edit Copy Delete	20	5ed5e73da35600.27234929	vmaithya	vixtreme09@gmail.com	G7DMRsq5RThfluwKNDlaYT8EZ9l0NTRmZTI1MmY4	454fe252f8	2020-06-02 08:44:29	NULL
Edit Copy Delete	21	5ed7703434b0d5.68902960	vnyamwaro	vnyamwaro@yahoo.co.uk	6061LFagPL4acLKj4mpF9EvooP01OGQxNWU5YzM5	58d15e9c39	2020-06-03 12:41:08	NULL
Edit Copy Delete	22	5ed772f1576034.17519546	Valentine Moraa	vmoraa@gmail.com	d+iRtRUpriqt6rlAUCXZvqGZ35g0ODU4MDI3YTdj	4858027a7c	2020-06-03 12:52:49	NULL

Check all With selected: Edit Copy Delete Export

Figure 5.9: User accounts Records Created in System

5.3.7 Encryption Key Storage

The System uses symmetric key cryptography for both encryption and decryption of the sensitive data that is selected by the user for storage. The symmetric key is defined by the user during sign up, and the key is encrypted and transmitted to the cloud for storage in an encrypted format. Figure 5.10 gives an overview of the storage of the key. The key is stored against a unique identifier key created for each user during sign up. No other user details are stored with it, which makes it hard to identification if unauthorized access occurs.

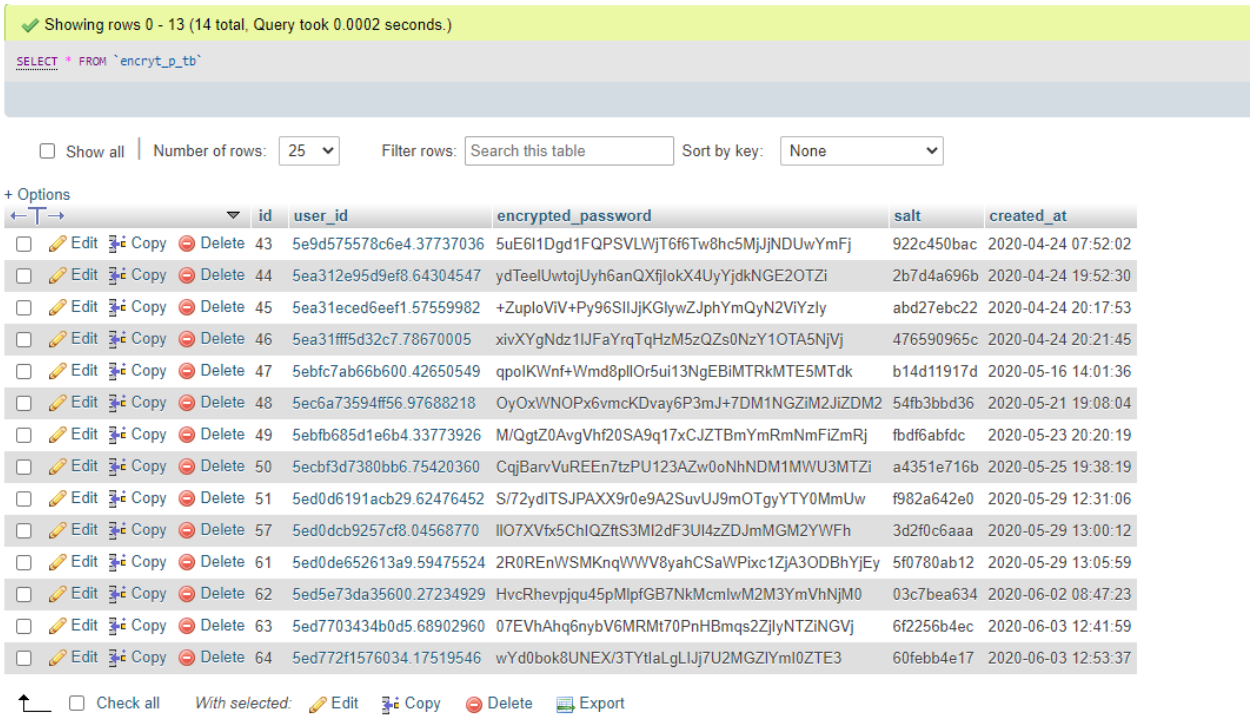


Figure 5.10: Symmetric Key Stored

5.4 System Testing

This section describes the tests that were performed on the developed mobile data encryption tool to ascertain if the set objectives were achieved. It describes the different tests that were carried out to validate the functional and non-functional requirements of the system.

5.4.1 Functional Requirements

Functional tests were done to ascertain whether the system functions work as per the system requirement. The use case tests were done in which testing measures were set to determine the success or failure of the functions implemented. The tables below show the main use cases and the results obtained from the test.

Table 5. 1: Account Creation Test Case

Identifier	1
Test case	Account Creation

Description	A user creates an account using their personal information
Utilized use case	Register Account
Results	Successful creation of an account
Pass / Fail	Pass

Table 5. 2: Login and Logout Test Case

Identifier	2
Test case	Login and Logout
Description	A user creates an account using their personal information
Utilized use case	Login, Logout
Results	Successful login to the system and access is granted, successful logout from the system and directed to login screen
Pass / Fail	Pass

Table 5. 3: Cryptography Key Creation Test Case

Identifier	3
Test case	Cryptography key creation
Description	A user creates an encryption and decryption key of valid length
Utilized use case	Create encryption and decryption key
Results	Successful creation of cryptography key
Pass / Fail	Pass

Table 5. 4: Encrypt Data & Upload to Server Test case

Identifier	4
Test case	Encrypt data and upload for server
Description	Aser adds files to the system, they are encrypted and uploaded to the server for storage
Utilized use case	Add data from mobile storage
Results	Successful encryption and upload of data to the server for storage
Pass / Fail	Pass

Table 5. 5: Download and Decryption of Data Test Case

Identifier	5
Test case	Decryption of data
Description	A user downloads data from the server decrypts using the key to a format that is readable
Utilized use case	View data
Results	Successful decryption of data for viewing
Pass / Fail	Pass

5.4.2 Compatibility Testing

Compatibility tests were carried out in two phases. The first phase was in the development environment, while the second phase was in the production environment once the system was stable for deployment. This test was done to check whether the developed Android mobile

application is capable of running on different Android platforms that exist on smartphones in the market. Table 5.6 show the compatibility results against the Android platforms.

Table 5. 6: Android Platform Compatibility Test

Android OS Code Name	Version No	API Level	Test Results
Oreo	8	26	Pass
Oreo	8.1	27	Pass
Pie	9	28	Pass
Android 10	10	29	Pass

5.4.3 Usability Testing

This test was conducted to determine whether the developed application met the users objective. In order to determine the goal of this, the test was performed by a sample of 10 users whereby they got to install the application on their mobile devices, navigate through it and after which respond to a questionnaire. The areas of focus for the tests were: user-friendliness, functionality, aesthetic and acceptability.

5.4.3.1 User Friendliness

This involves the ease of learning and use of the system. The APK of the system was shared with ten potential mobile users. They were able to install the tool on their android mobile devices and test the ease of learning and use of the application on the mobile devices. The respondents provided their feedback with responses ranging from easy, average and difficult. 60% of the respondents found the application easy to use, while 40% found it to be average. Figure 5.11 provides a summary of the results.

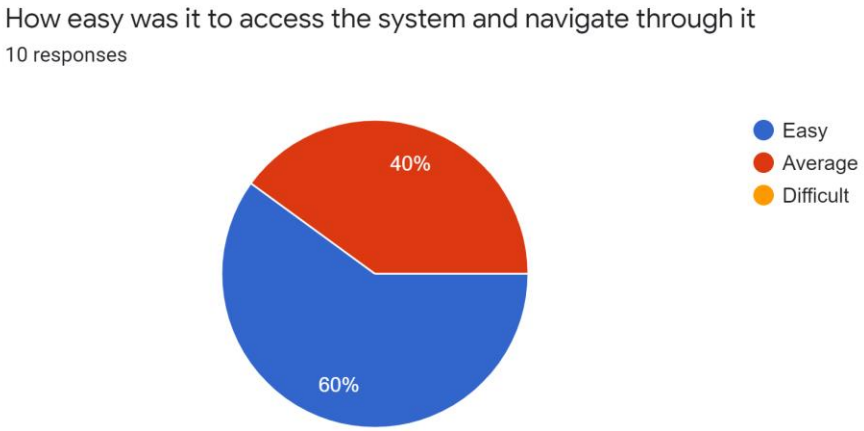


Figure 5. 11: User Friendliness Testing Feedback

5.4.3.2 Functionality

The Mobile users tested how the system functioned. This entailed testing the functionality of the system against the user needs. The respondents provided feedback on the functionality of the system after testing it, and they ranged from their responses from very satisfied, satisfied, neutral, dissatisfied and very dissatisfied. 40% of the respondents were very satisfied with the functionalities the system provided, 50% were satisfied, and 10% of the users were neutral. Figure 5.12 provides a summary of the results.

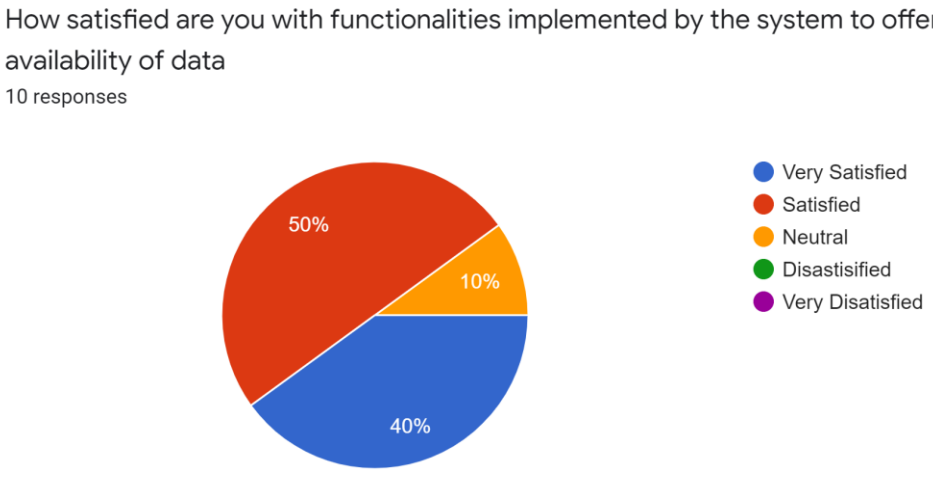


Figure 5. 12: Functionality Testing Feedback

5.4.3.3 User Interface Aesthetic

User aesthetic is defined by the look and feel of the application design and flows to the users. The system was tested for its appearance to the users. 90% of the respondents found the system to be attractive, while 10% of the respondents found the appearance of the system to be average. Figure 5.13 provides the summary of the results.

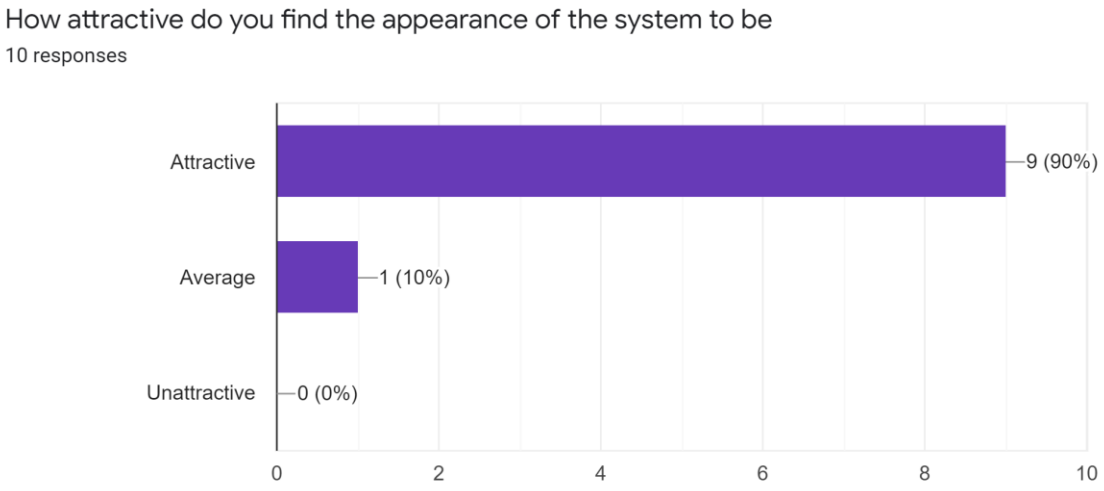


Figure 5. 13: User Interface Aesthetic Testing Feedback

5.4.3.4 Acceptability

This test was carried out in ensuring that the system was a success and was well accepted by the users. 90% of the respondents indicated that they would use the system for enhancing confidentiality of the sensitive data on their mobile devices, while 10% of the respondents were not sure if they would use it. Figure 5.14 provides a summary of the results.

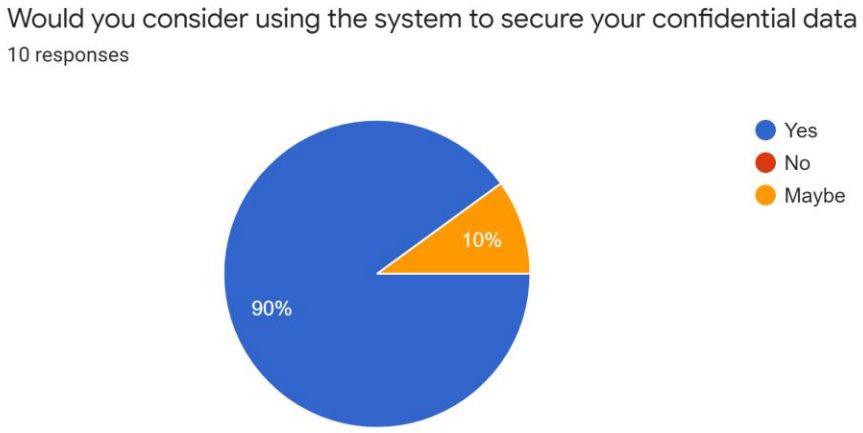


Figure 5. 14: System Acceptability Testing Feedback

Conclusion

The data gathered at the requirement gathering and analysis stage provided fundamental information that was used in the implementation of the system. The system design aided in the development of the mobile data encryption tool in ensuring that the developed tool met the research objective of the study. Various tests were carried out on the tool to validate the functional and non-functionality requirements of the system.

Chapter 6: Discussion of Results

6.1 Introduction

This chapter aims to discuss the findings of the research concerning the specific objectives that were set.

6.2 Research Objective One Discussion

From the specific objectives set in chapter one of this research objective, the first objective was to identify the common threats that are associated with mobile data storage for breaching confidentiality. As discussed in the literature review, it has revealed that smartphones have gained quite a market share at a rapid rate as they have got so many functions to aid in the day-to-day life of the user, and with this, they have attracted a high number of security threats. Among the threats identified in the research, they include the loss of data due to insecure data storage and loss of mobile phone device, malware and spyware on the phones, lack of use of the inbuilt security mechanism on the phone by users.

Insecurity storage of mobile storage is a high risk in the study, especially for android based mobile devices. As discussed in the study, the android operating system architecture has standard security issues noted in its operating system architecture due to its inherent code complexity. The Operating System libraries or the applications on the platform can be utilized by attackers to get access to the private user data stored in the shared internal device storage. Most attackers have managed to compromise devices by using malware and spyware, which are often incorporated in the legitimate application being installed from application stores.

Due to the physical size of mobile devices, they easily get lost or stolen, and with this, they subsequently fall under the control of others. This act leaves the data on the device vulnerable to access by unauthorized users. With some users lack of security awareness and utilization of the inbuilt security mechanisms for securing devices, private data stored on the devices can easily be accessed from the storage, while for those devices with security mechanisms such as pins, patterns and password utilized, third party applications can be used to break the security mechanism for access to data storage.

6.3 Research Objective Two Discussion

The second objective of the research was to investigate and study the current tools and solutions in use for mobile data confidentiality. This specific objective was intended to give an insight into the existing solutions being used to address the problem of the security of data at the rest of the mobile device's storage. To understand the gap in the issue, several applications in use were reviewed, and their strengths and gaps acknowledged. In this study, it was noted that the available applications offering the solution for data privacy are primarily available in flagships phone models, and thus they were unavailable for middle-level phones or the old models of devices already in use. Among those that cut across all models of phones, the mobile devices were noted to lack the concept of availability of data, as the data was stored on the mobile device storage after encryption, and due to the risk of the operating system security, this posed a threat of data leakage where the encryption key stored on the mobile storage can be leaked by malware. Further research and evaluation on the cryptography method were done to determine the most appropriate algorithms that can be implemented. There is a valuable algorithm available that would be ideal for the task, including AES and blowfish, but in comparison to achieving both speed and efficiency while getting the job done, AES came out to be the most appropriate algorithm.

6.4 Research Objective Three Discussion

The third objective of the research was to design, develop and test a working mobile application tool for data confidentiality. This was achieved through the design, implementation and testing of the tool developed. The tool was developed to address the main vulnerability of secure data storage. To achieve this, the tool was developed to be able to enable the user to store their sensitive data in a secure location in a format that would not be readable by unauthorized parties. The data would be stored in a secure location in the cloud, where the user can be able to access the data at any given time. The password and cryptography key for the user is stored in a server that is not accessible by the client. Connection to the server is via a middle-tier application that handles the logics enforced in the application, and it creates the connection between the client and the server to be able to access the data whenever needed and from any android enabled device.

6.5 Research Objective Four Discussion

The fourth objective of the research project was to validate the effectiveness of the solution provided by the tool. The objective was to be achieved by validating the data security concepts of

confidentiality. The developed tools security solutions were tested in which it was noted that for confidentiality, all the confidential user data were stored in an encrypted format using the key provided by the user to encrypt it; thus, the files would not be able to be accessed by any unauthorized person. For the storage of the key and password created by the user, all the information were stored in a server using the SHA3 hash algorithm and combined with salt to make it more secure during storage. Encryption of data was carried out using AES 256 Algorithm, which is quite a solid algorithm to break. For the availability of data, the files were encrypted and uploaded to a cloud-based server from which they can be accessed wherever they are needed.

6.6: Comparison of the tools for enhancing confidentiality

The table below provides a comparative overview of the features provided by the available tools for enhancing confidentiality and the developed system features as well.

Table 6.1: Comparative analysis of Available Similar Tools

Applications Comparison against Features	***MobiFlage For Plausible Deniability***	***Crypt4All***	***Samsung Knox***	***App Lock***	***FileCrypt (Proposed Developed System***
Supported by a wide range of smartphone	YES	YES	NO (Limited to Samsung High-End Models)	YES	YES
Range of File Supported	Limited range of files (Documents)	Limited range of files (Documents)	All types of files in local device storage	All kinds of files in local device storage	All kinds of files in local device storage
Key Generation for Encryption	Derived by the system	Generated by the user	Derived with PBKDF2	Generated by user	Generated by user
Storage of Encryption Key	On-device	On-device	On-device	On-device	Cloud
Backup of the files to the cloud for availability	NO	YES (has to be done manually)	NO	NO	YES (Automatic)

Chapter 7: Conclusions, Recommendations and Future Work

7.1 Conclusions

The research of this project was focused on the development of a mobile application that would provide security for the users' confidential data while making it available. The tool was intended for Android device users to enable the users to store the selected sensitive files from their mobile devices in an encrypted format using the algorithm AES 256 to encrypt the data with a key only known to the user and upload it cloud for secure storage. The Mobile application system was developed and was able to fulfil the requirement specification successfully.

7.2 Recommendations

Security is a significant concern in mobile devices for data at rest. The proposed approach aims to not only give the user satisfaction of storing their personal user data securely on the devices but also available wherever it is needed. To provide more security, the encryption key and the files can be deployed on different servers to make it more complex in case of an attack on the server as there will be two separate machines to be attacked.

The focus of the system was for the storage of selected files in a remote location; however, further study can be done, and improvements done to secure the local data storage on the device and secure different storage levels from being accessed without an encryption key.

7.3 Future Work

For further research on mobile privacy and confidentiality, the application can be modified to provide more functionality in the scope of secure communication. Communication allows the users to be able to share messages and data within the application without any interception in an encrypted format. This can get to incorporate the use of combined algorithms such as AES 256 and RSA for authentication. The future study would also focus on the need to deploy the application on mobile devices across the different platforms to be able to provide confidentiality, availability across the different devices.

References

Allen, J. (2010, December). ROAD WARRIOR: Data Security in a Mobile World. The Greening of Your Law Practice, Volume 27(No 8), 4 - 6. Retrieved 8th April 2019 from http://www.jstor.org/stable/23630279

Anand, A., Raj, A., Kohli, R. and Bibhu, V., "Proposed symmetric key cryptography algorithm for data security," 2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH), Noida, 2016, pp. 159-162, DOI: 10.1109/ICICCS.2016.7542294.

Armel, A.S.R and Thavavel, V., "Ghost encryption: Mobile data security model encrypting data before moving it to the cloud service provider," 2013 Fifth International Conference on Advanced Computing (ICoAC), Chennai, 2013, pp. 512-516, DOI: 10.1109/ICoAC.2013.6922004.

B. Debnath, A. Das, S. Das and A. Das, "Studies on Security Threats in Waste Mobile Phone Recycling Supply Chain in India," *2020 IEEE Calcutta Conference (CALCON)*, 2020, pp. 431-434, doi: 10.1109/CALCON49167.2020.9106531.

Boyles, J. L., Smith, A., & Madden, M. (2012, September 5). Privacy and Data Management on Mobile Devices. Retrieved from Pew Research Centre internet & technology: Retrieved 22nd April 2019 from http://www.pewinternet.org/

Chen, Y. and Ku, W. "Self-Encryption Scheme for Data Security in Mobile Devices," 2009 6th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, 2009, pp. 1-5, DOI: 10.1109/CCNC.2009.4784733.

Ciphr Team (2020). Celebrity Examples of Data Breaches. Retrieved from https://ciphr.io/blog/post/celebrity-examples-data-breaches on 27th August 2021

Collett, S. (2017, August 1). Five new threats to your mobile security. Retrieved 8th March 2020 from https://www.csoonline.com/article/2157785/

Dimensional Research. (2017 April). The Growing threat of mobile Device security breaches; A global security survey of security professionals. San Carlos: Check Point Software Technologies. Retrieved from https://blog.checkpoint.com/wp-content/uploads/2017/04/

Farooq, Umer. (2018). Android Operating System Architecture. Retrieved from: 10.13140/RG.2.2.20829.72169.

Fernando, E., Agustin, D., Irsan, M., Murad, D.F., Rohayani, H. and Sujana, D. "Performance Comparison of Symmetries Encryption Algorithm AES and DES With Raspberry Pi," 2019

International Conference on Sustainable Information Engineering and Technology (SIET), Lombok, Indonesia, 2019, pp. 353-357, DOI: 10.1109/SIET48054.2019.8986122.

Furnell, S., Clarke, N., & Karatzouni, S. (2008). Beyond the pin: Enhancing user authentication for mobile devices. Computer fraud & security, 2008(8), 12-17. DOI: 10.1016/S1361-3723(08)70127-1

Garg, S., & Baliyan, N. (2021). Comparative analysis of Android and iOS from security viewpoint. *Computer Science Review*, *40*, 100372. https://doi.org/10.1016/j.cosrev.2021.100372

Gary .C. Kessler, An Overview of Cryptography. Retrieved 23rd March 2020 from: https://www.garykessler.net/library/crypto.html

Ghallali, M. and Ouahidi, B.E. "Security of mobile phones: Prevention methods for the spread of malware," 2012 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Sousse, 2012, pp. 648-651, DOI: 10.1109/SETIT.2012.6481989.

Hernández, M., Baquero, L., & Gil, C. (2018). Ethical Hacking on Mobile Devices: Considerations and practical uses. *International Journal of Applied Engineering Research*, *13*(23), 16637-16647.

Hutchinson, S. and Varol, C. "A Survey of Privilege Escalation Detection in Android," 2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York City, NY, USA, 2018, pp. 726-731, DOI: 10.1109/UEMCON.2018.8796550.

Kearns, G. S. (2016, June). Countering Mobile Device Threats: A Mobile Device Security Model. Journal of Forensic & Investigative Accounting, Volume 8(Issue 1), 36 - 48. Retrieved 3rd April 2019 from http://web.nacva.com/JFIA/Issues/JFIA-2016-4.pdf

Khandelwal, S. (2015, April 29). Review on the Best Android Privacy and Security Apps. Retrieved 6th May 2020 from The Hacker News: https://thehackernews.com/2015/04/android-privacy-security-apps.

Kumar, A. (2016, June 3). Risk of mobile threats and privacy concerns grow. Retrieved 28th March 2019 from https://www.csoonline.com/article/3078815/security/risk-of-mobile-threats-and-privacy-concerns-grow.html

Kumari, S. (2017). A research paper on cryptography encryption and compression techniques. International Journal Of Engineering And Computer Science, 6(4).

Lee, J., Pelechrinis, K. and Zeinalipour-Yazti, D., "Human Mobility Computing and Privacy: Fad or Reality?," 2015 16th IEEE International Conference on Mobile Data Management, Pittsburgh, PA, 2015, pp. 42-43. https://doi.org/10.1109/MDM.2015.86

Li, Q. and Clark, G. "Mobile Security: A Look Ahead," in IEEE Security & Privacy, vol. 11, no. 1, pp. 78-81, Jan.-Feb. 2013, DOI: 10.1109/MSP.2013.15.

McAfee. (2017). McAfee Labs Threat Report. Santa Clara, CA: McAfee LLC. Retrieved from https://www.mcafee.com/us/resources/reports/rp-quarterly-threats-jun-2017.pdf

Ohm, P. (2015). Sensitive Information. *Southern California Law Review*, *88*. Retrieved from https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2501002 on 2nd May 2021

Power, E., & Trope, R. L. (2006, November). The 2006 Survey of Legal Developments in Data Management, Privacy, and Information Security: The continuing Evolution of Data Governance. The Business Lawyer, Volume 62(Issue 1), 251 - 294. Retrieved from http://www.jstor.org/stable/40688420

Prey (2020) Mobile Theft & Loss Report 2020 Retrieved from https://preyproject.com/uploads/2020/03/Mobile-Theft-Loss-Report_2020.pdf on 1st September 2021

Pry, J.C, and Lomotey, R.K (2016) "Energy Consumption Cost Analysis of Mobile Data Encryption and Decryption," 2016 IEEE International Conference on Mobile Services (MS), San Francisco, CA, 2016, pp. 178-181, DOI: 10.1109/MobServ.2016.35.

Qian, K., Parizi, R.M. and Lo, D. "OWASP Risk Analysis Driven Security Requirements Specification for Secure Android Mobile Software Development," 2018 IEEE Conference on Dependable and Secure Computing (DSC), Kaohsiung, Taiwan, 2018, pp. 1-2. DOI: 10.1109/DESEC.2018.8625114.

Raymond, V.J. and Sushmitha, E. "Google drive based secured anti-theft android application," 2017 International Conference on IoT and Application (ICIOT), Nagapattinam, 2017, pp. 1-8, DOI: 10.1109/ICIOTA.2017.8073623.

SINGH, Y. K., Milan, R., & RATAWA, S. P. (2017). Advance cryptography algorithm for improving data security. IJRDO - Journal of Computer Science Engineering (ISSN: 2456-1843), 3(6), 22-29. Retrieved from https://www.ijrdo.org/index.php/cse/article/view/480

Symmetric Cryptography, Retrieved from https://www.sciencedirect.com/topics/computer-science/symmetric-cryptography

Umasankar. (2017). Analysis of latest vulnerabilities in android. International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. pp. 1236-1241). Udupi, India: IEEE. doi:10.1109/ICACCI.2017.8126011

Vichare, A., Jose, T., Tiwari, J., & Yadav, U. (2017, May). Data security using authenticated encryption and decryption algorithm for Android phones. In 2017 International Conference on Computing, Communication and Automation (ICCCA) (pp. 789-794). IEEE.

Weichbroth, P., & Łysik, Ł. (2020). Mobile security: Threats and best practices. *Mobile Information Systems, 2020*. Retrieved from https://www.hindawi.com/journals/misy/2020/8828078/ on 3rd September 2021

Wessel, S., Huber, M., Stumpf, F., & Eckert, C. (2015). Improving mobile device security with operating system-level virtualization. Computers & Security, 52, 207-220. https://doi.org/10.1016/j.cose.2015.02.005

Yalew, S. D., Maguire, G. Q., Haridi, S., & Correia, M. (2017). Hail to the Thief: Protecting Data from Mobile Ransomware with ransomSafeDroid. (pp. pp. 1 - 8). Cambridge, MA, USA: 2017 IEEE 16th International Symposium on Network Computing and Applications (NCA). DOI: 10.1109/NCA.2017.8171377

Yoon, S., Jeon, Y. and Kim, J. "Mobile security technology for smart devices," 2015 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, 2015, pp. 1171-1173, DOI: 10.1109/ICTC.2015.7354766.

Zinkus, M., Jois, T. M., & Green, M. (2021). Data Security on Mobile Devices: Current State of the Art, Open Problems, and Proposed Solutions. *arXiv preprint arXiv:2105.12613*. Retrieved from: https://arxiv.org/pdf/2105.12613.pdf on 1st September 2021

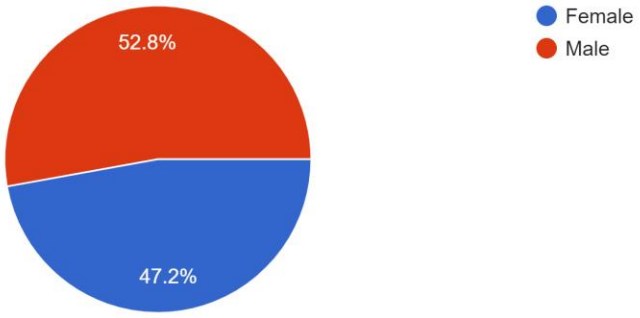
Appendices

Appendix A : Mobile Data Questionnaire Responses

An online survey form was created with the use of google form to obtain data from users. A total of 123 responses were successfully received to help in answering some of the research questions about mobile data privacy and in designing the system. Responses from the online questionnaire were analysed using the Google forms analysis tool. Graphs and Charts were used to depict the answers from respondents, as shown below.

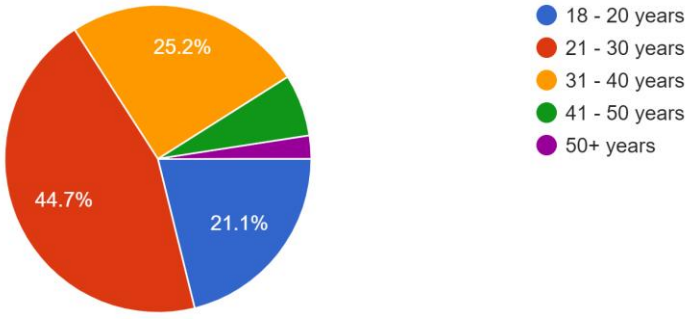
1. What is your Gender?

123 responses



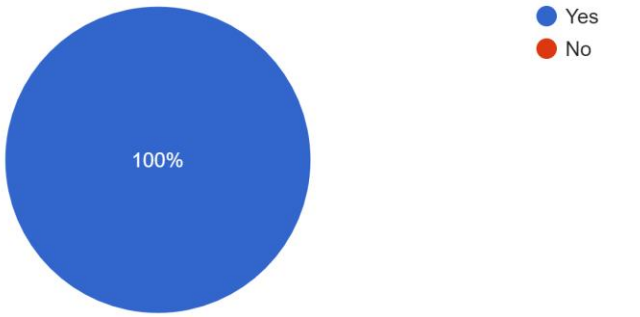
2. Age Group

123 responses



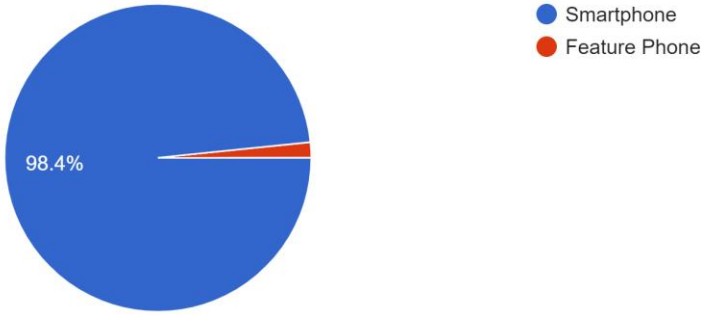
3. Do you own a mobile device?

123 responses



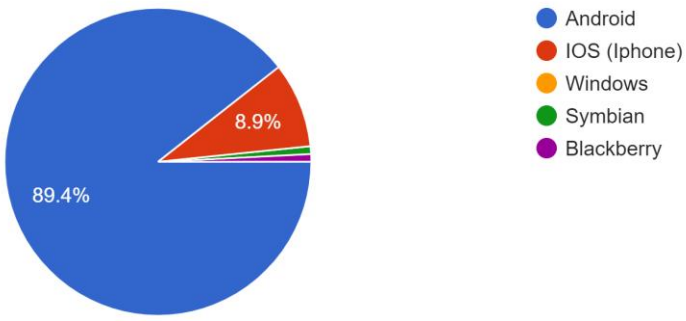
4. What type of mobile phone do you own?

123 responses



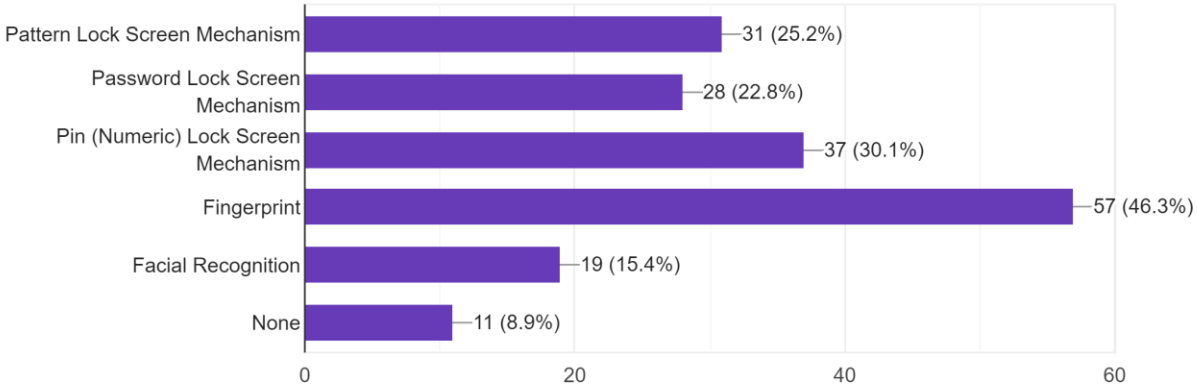
5. Which operating system does your mobile device operate on?

123 responses



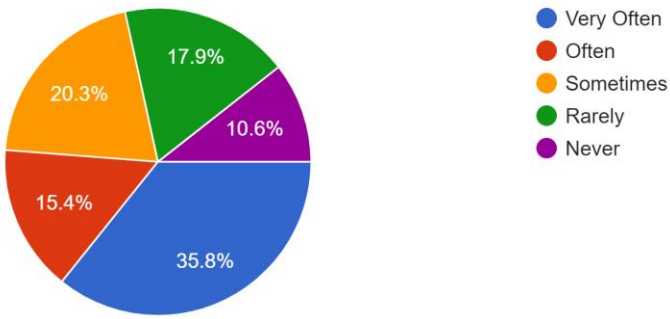
6. What security mechanism do you use on locking and unlocking your screen to secure your phone

123 responses



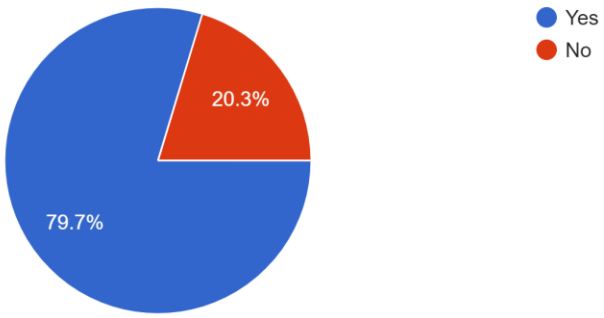
7. How often do you store confidential/sensitive data on your mobile device?

123 responses



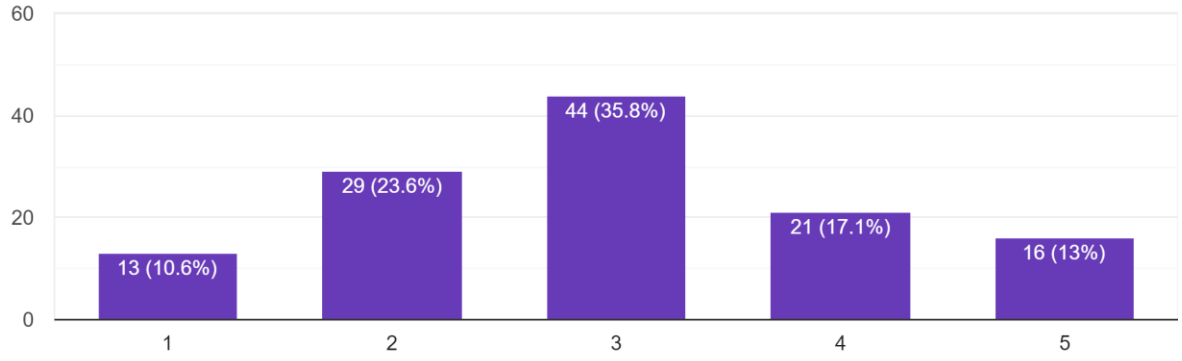
7 (b) Do you feel if these sensitive / confidential data stored on your phone is accessed by unauthorized personnel or leaked it can cause a security breach?

123 responses



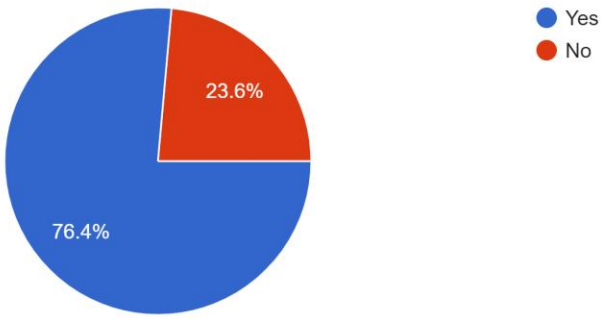
8. Rate the confidentiality or sensitivity of types of data you have stored on your mobile device? (With 5 being highly confidential and 1 not confidential)

123 responses



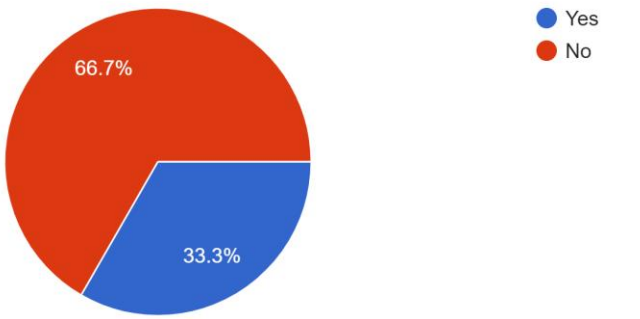
10. Have you ever lost or misplaced your mobile device?

123 responses



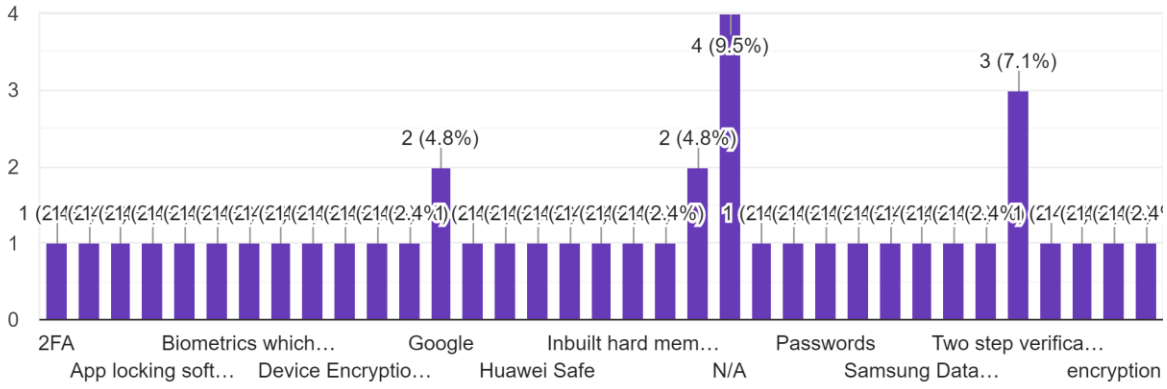
11. Are you aware of existing technologies that you can use to protect data on your mobile device

123 responses



12. If yes, what technologies do you use to protect data in your mobile phone? (List the technologies)

42 responses



Appendix B: Encryption and Decryption Code

```
private File processFile(@NonNull boolean encrypt, @NonNull File inFile, @NonNull String inputKey) throws Exception {
    String fileName = inFile.getName();
    String IV = "!a3edr45!a3edr45";
    IvParameterSpec ivSpec = new IvParameterSpec(IV.getBytes(StandardCharsets.UTF_8));
    byte[] keyStart = inputKey.getBytes(StandardCharsets.UTF_8);

    KeyGenerator kgen = KeyGenerator.getInstance("AES");
    SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");
    sr.setSeed(keyStart);
    kgen.init( keysize: 128, sr);
    SecretKey skey = kgen.generateKey();
    byte[] key = skey.getEncoded();

    SecretKeySpec skeySpec = new SecretKeySpec(key,  algorithm: "AES");
    Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");

    if (encrypt) {
        cipher.init(Cipher.ENCRYPT_MODE, skeySpec, ivSpec);
    } else {
        cipher.init(Cipher.DECRYPT_MODE, skeySpec, ivSpec);
    }
    // Read input file into byte array
    FileInputStream fileInputStream = new FileInputStream(inFile);
    byte[] inputBytes = new byte[(int) inFile.length()];
    fileInputStream.read(inputBytes);
    byte[] outputBytes = cipher.doFinal(inputBytes);

    String prefix = encrypt ? "ENC_" : "DECRYPT_";

    File mFile = File.createTempFile(prefix, fileName);

    // Write the output byte array to the output file
    FileOutputStream fileOutputStream = new FileOutputStream(mFile);
    fileOutputStream.write(outputBytes);

    // Close file streams
    fileInputStream.close();
    fileOutputStream.close();
    return mFile;
}
```

Appendix C: Downloaded File Auto Deletion Code

```
void deleteRecursive(@Nullable File storageDirectory) {

    //finds the file storage directory
    if (storageDirectory == null) {
        String root = Environment.getExternalStorageDirectory().toString();
        storageDirectory = new File( pathname: root + "/" + BuildConfig.APPLICATION_ID);
    }

    if (storageDirectory.exists()) {
        if (storageDirectory.isDirectory())
            for (File child : Objects.requireNonNull(storageDirectory.listFiles()))

                //deletes storage files recursively
                deleteRecursive(child);

        //delete files
        storageDirectory.delete();
    }
}
```

In the event of logout from the system Code

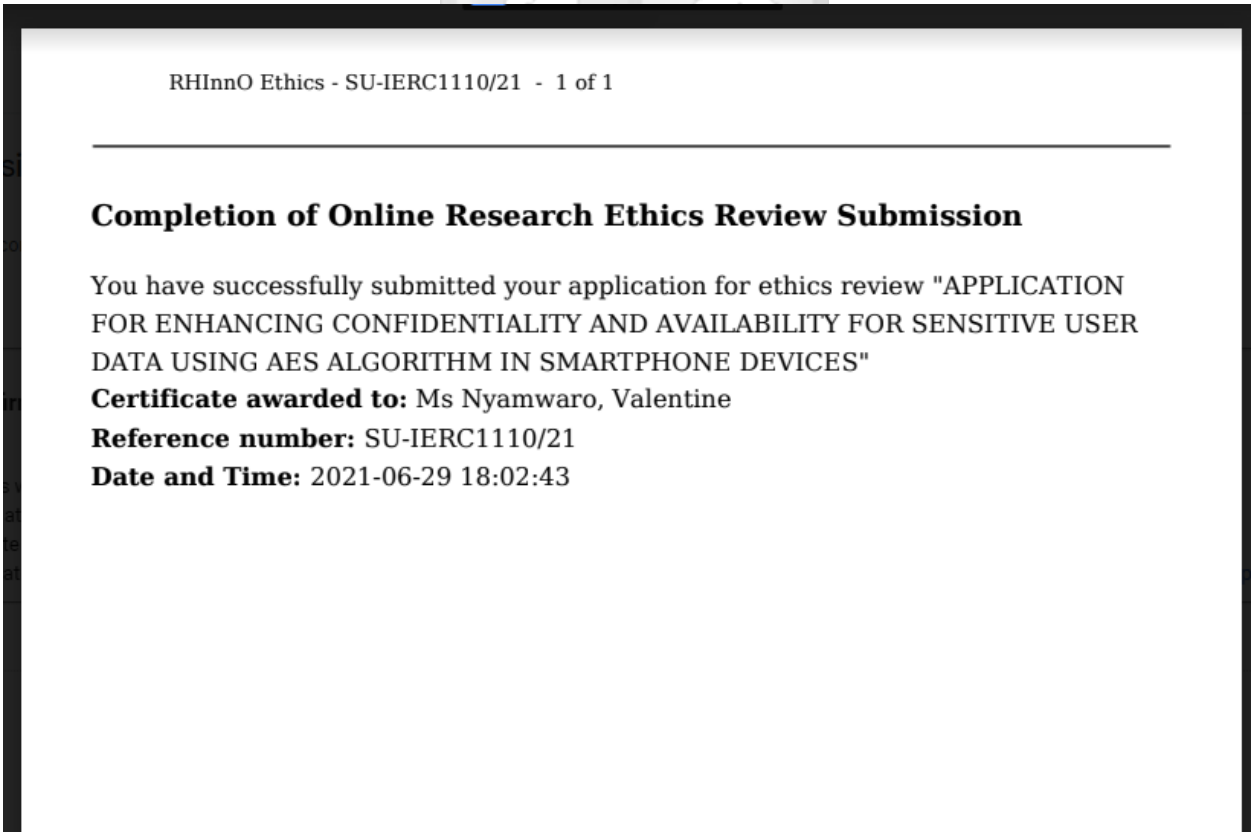
```
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    if (item.getItemId() == R.id.action_log_out) {

        //Confirm log out
        AppManager.getInstance().showConfirmAlert( mContext: this, sweetAlertDialog,  msg: "Log Out from this App", sweetAlertDialog -> {
            //Deletes downloaded files
            AsyncTask.execute(() -> AppManager.getInstance().deleteRecursive( storageDirectory: null));

            //clear session and return to login page
            if (SessionHandler.getInstance().destroySession( mContext: MainActivity.this)) {
                startActivity(new Intent( packageContext: MainActivity.this, LoginActivity.class));
                MainActivity.this.finishAffinity();
            }

        });
    }
    return super.onOptionsItemSelected(item);
}
```

Work Schedule in case user exits the application without logging out

```
OneTimeWorkRequest oneTimeWorkRequest = new OneTimeWorkRequest.Builder(DelWorker.class).setInitialDelay( duration: 60, TimeUnit.MINUTES).build();
WorkManager.getInstance().enqueueUniqueWork( uniqueWorkName: "DEL", ExistingWorkPolicy.REPLACE, oneTimeWorkRequest);
```

```
public class DelWorker extends Worker {
    public DelWorker(@NonNull Context context, @NonNull WorkerParameters workerParams) {
        super(context, workerParams);    }

    @NonNull
    @Override
    public Result doWork() {
        AppManager.getInstance().deleteRecursive( storageDirectory: null);
        return Result.success();
    }
}
```

Appendix D: Ethical Review Report

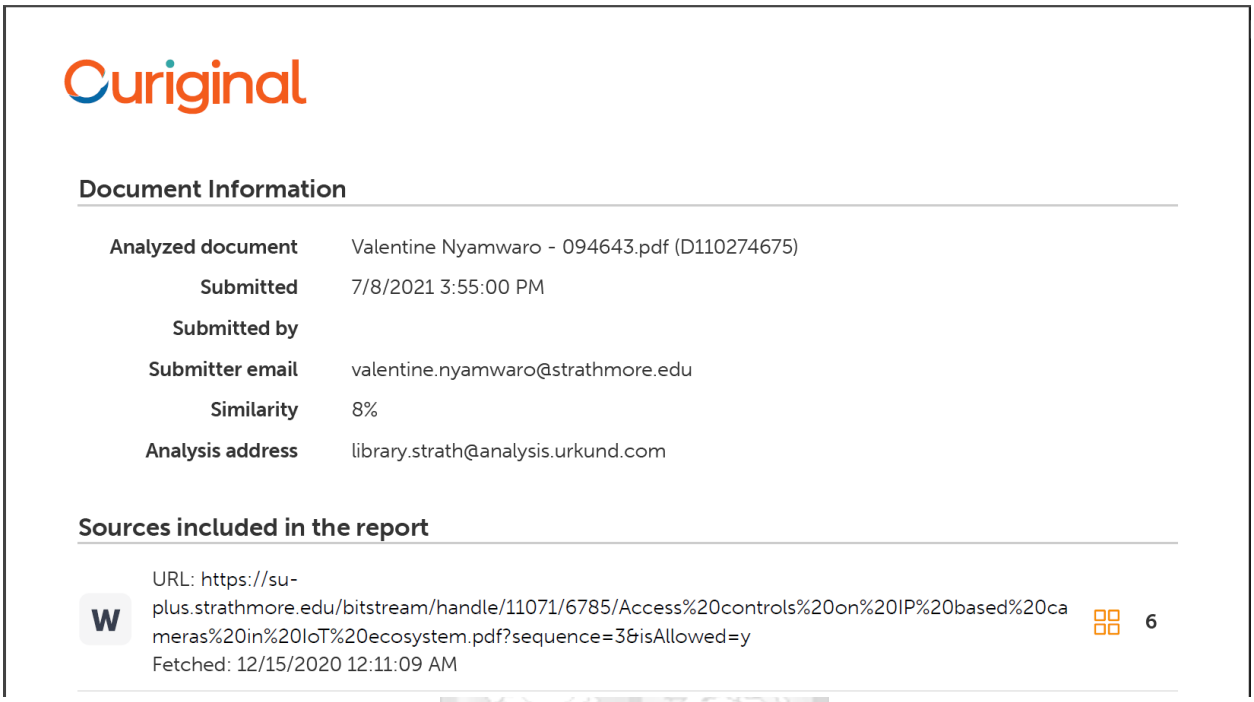


RHInnO Ethics - SU-IERC1110/21 - 1 of 1

Completion of Online Research Ethics Review Submission

You have successfully submitted your application for ethics review "APPLICATION FOR ENHANCING CONFIDENTIALITY AND AVAILABILITY FOR SENSITIVE USER DATA USING AES ALGORITHM IN SMARTPHONE DEVICES"
Certificate awarded to: Ms Nyamwaro, Valentine
Reference number: SU-IERC1110/21
Date and Time: 2021-06-29 18:02:43

Appendix E: Similarity Report



Curiginal

Document Information

Analyzed document	Valentine Nyamwaro - 094643.pdf (D110274675)
Submitted	7/8/2021 3:55:00 PM
Submitted by	
Submitter email	valentine.nyamwaro@strathmore.edu
Similarity	8%
Analysis address	library.strath@analysis.urkund.com

Sources included in the report

W URL: https://su-plus.strathmore.edu/bitstream/handle/11071/6785/Access%20controls%20on%20IP%20based%20cameras%20in%20IoT%20ecosystem.pdf?sequence=3&isAllowed=y
Fetched: 12/15/2020 12:11:09 AM 6