ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
Indian Journal of Science and Technology, *Vol 11(16), DOI: 10.17485/ijst/2018/v11i16/121410, April 2018*

Mitigating Storage Security Threats in Mobile Phones

Sabahat Hussain*, Rashidah F. Olanrewaju and Ahmad Fadzil Bin Ismail

Department of Electrical and Computer Engineering, Kulliyah of Engineering, IIUM, Malaysia; Sabahathussain903@gmail.com, frashidah@iium.edu.my, af_ismail@iium.edu.my

Abstract

Objectives: This paper discusses the proposed system which provides the user the ability to run the application on Android phones for encrypting all types of files before they are stored. **Methods/Statistical Analysis**: In the proposed system, Advanced Encryption Standard (AES) is employed for encryption as well as decryption of the files stored by users in mobile phones. All types of files including docx, text, pdf, image, ppt, audio and video files can be encrypted and later regenerated as well. **Findings:** The use of AES in encrypting and decrypting data in this system provides good security as well as higher speed. An added advantage is that it is implementable on several platforms particularly in small gadgets like smartphones. As compared to the traditional computers, smartphones can be carried more easily and provide similar functionalities as that of a computer like data storage and processing, communication and other services including video call, wireless network, web browser, GPS, audio or video player. **Application/Improvements:** Data is transmitted, shared and stored for various purposes including production, banking, development, and research. Therefore, security is a must for information which can be provided by encryption using AES as proposed in this system.

Keywords: Encryption, Decryption, Mobile Phones, Security, Threats, AES

1. Introduction

After the advances in technology, it can be said with certainty that smartphones have replaced the standard mobile phones in the market. A combination of personal digital assistant and cellular phone, smartphone use has increased over the past few years. The smartphones are built on a mobile operating system with computing capabilities and advanced connectivity. They are very portable and help the users perform their tasks easily and simultaneously. Instead of carrying Personal Digital Assistants or personal computers, the user can just carry the smartphone, which is much more portable and are capable of multitasking – they can send a text message or an email or make a phone call from the smartphone; it doesn't merely perform one duty only[1]. Smartphones use a variety of operating systems. Some of the very popular mobile operating systems are Windows, Apple's IOS, Nokia's Symbian, Android, RIM's BlackBerry OS, etc. Android is a Linux kernel-based mobile operating system that is presently developed by Google. It was first used on a smartphone in October 2008 on HTC's Dream[2]. Developed mainly for handheld devices like wrist gear, tablets, smartphones, etc., it is open source and has been outselling its competitor's viz. Windows, Blackberry, and IOS[3]. Furthermore, the Android operating system allows different third-party developers to develop applications for it, unlike Apple where the developer must be registered with the development site[4]. With the growing popularity of mobile devices and smart gadgets like tablets, the use of electronic mails and social networking on such mobile devices has increased immensely in recent years and thus

*Author for correspondence

special applications had to be built for serving the public demands[5,6]. For Android, OneNote application provides the most commonly used integration of social media (LinkedIn, Twitter and Facebook) and e-mail[7]. In general, a user remains logged in to his/her application whenever he/she logs in using a mobile phone. Thus, for ensuring security, the user must log out from the applications.

Android mobile devices have become a popular alternative to traditional PCs. However, there have been observed some vulnerability in the Android platform that received major media attention[8]:

i. Adups data transmission
There are several device manufacturers of Android that employ over-the-air (OTA) third-party update services for deploying software updates on their gadgets. But this OTA does not form a part of the main Android platform. It was discovered in November 2016 that manufacturers using this Adups OTA service were wrongly implementing a configuration which transmitted crucial information from gadgets to Adups servers. After receiving the notification, device manufacturers updated their gadgets not to transmit data of such type to Adups.

ii. CVE-2015-1805
The security researchers found a means of exploiting CVE-2015-1805 in February 2016 and later informed Google that a famous device-rooting application namely Kingroot was externally using it. Then, an emergency patch was released by Google in March for addressing this issue and rules were added to the PHA detection systems for alarming users from the applications that exploit them.

iii. Dirty Cow
Dirty Cow or CVE-2016-5195 was vulnerability in the Linux kernel related to privilege-escalation due to which Android devices became vulnerable to rooting. This vulnerability could have been exploited by local attackers for gaining write access to ROM including the executable files' memory-cached version thereby enhancing their privileges to the system. Al that was required for the user was to download an application for gaining root access by taking advantage of this loophole.

iv. Quadrooter
It is formed of four vulnerabilities - CVE-2016-2503, CVE-2016-2059, CVE-2016-5340 and CVE-2016-2504 - which badly affect a Qualcomm chipset popularly utilized in Android devices. These were disclosed to the public in August 2016 at DEF CON 24.

With time, enhancements have been made in the operating system security technology for preventing issues related to confidentiality, availability, and integrity in a network[9]. The Android system (and other operating systems of mobile phones) has a "system-centric" security model. The permissions governing the rights to data and interfaces are identified by applications statically at the time of installation. Nevertheless, the developer/application cannot control the access to these rights and how they are then exercised. Basically, permissions are assumed to be imprecise suggestions related to the type of protection desired by the application. Thus, the applications must rely on the user and the operating system that the user won't make any wrong choice regarding which application the permission is granted, and in several instances, it has been found to be impossible since they lack enough contexts for doing so[10].

Taking those mentioned above and various other vulnerabilities into consideration, security becomes a crucial aspect of Android devices. The rise in the tasks being performed on these devices suggests that sensitive information is sometimes stored on these devices. Due to this, they have become targets for criminal exploitation[11]. Logged in social networking apps and email apps are very vulnerable to misuse. These apps may contain a lot of sensitive information and secure data. Vulnerabilities in these apps will attract misuse of the data and information in these apps[12]. When a user is logged in to the email or any social app, all the notifications and updates within the app can be accessed. When the app is exited, it will not log out. The email and social apps on these smartphones are live apps. Once logged in, they are always logged in, and the user has to log out to ensure security[13]. The existing security can be changed or improved by android modifications, which can reduce the potential of criminals to harm. This study discusses a system that can safeguard the data stored on the Android devices from attackers.

1.1 Problem Statement

Security in smartphones and tablets is essential and can be applied at various levels; the security of the phone in general, and the security of applications in the phone such as social networking apps, online banking apps, email apps, online shopping apps, etc[14,15]. In the Android environment, some degree of rushing and floppiness can be seen in security issues like unintended data leakage through application's log files. Some categories, like Intents in IPC controls, are something that most un-professional developers might not understand completely, and thus they cause security holes that way. It is common for users storing their vital documents and secret data in their mobile phones to avoid the burden of having to carry them every time. Another advantage mobile phones provide is the ability to form ad hoc networks effortlessly thereby providing an economical solution for communication for numerous applications like emergency services, tactical networks, enterprise/home networking, commercial and civilian environment, and etc[16]. Legal documents such as Passport, Birth Certificate, etc. and other sensitive information like user credentials are essential and can thus be misused in the presence of malicious nodes[17,18]. Uploading these to any server is a risk-taking action as they can be hacked and keeping these on our phone is also dangerous[19]. In the existing system, there are few applications which can provide security for locking the system. Basically, locks are used for locking doors, etc. but this type of system is not proving to be completely secure. Therefore, we propose to develop a system that saves the uploaded documents in an encrypted form and store them in the Internal Memory, to avoid any kind of hacking internal or external.

1.2 Research Objectives and Scope

The research objectives and the scope of the proposed system have been enlisted below:

- Digitally empowering the residents by presenting them the Security Digital Locker.
- Minimize the practice of using physical documents.
- Ensuring the authenticity of the electronic documents (in any form – docx, pdf, mp3, mp4 or image) generated thus eradicating the use of fake documents.
- A secure and safe access to the documents issued by the government via a mobile application for all.
- Minimizing the administrative overhead for government agencies and other departments besides making their services easily accessible to all.
- Offline access to all the files anytime and anywhere.
- Enabling secure access by setting up a PIN for login to access documents and making them available electronically whenever required.

1.3 System Lifecycle Details

The Waterfall model (Figure 1) is followed in the development of this system which has been discussed as:

The Waterfall Model is a sequential, linear flow that progresses through the various developmental phases of the software in a downward manner, steadily resembling a waterfall. Thus, a new phase commences only after the completion of the previous phase. This approach does not provide the developer the provision to move back to a previous stage for handling any changes in the requirement. It is the most primitive approach which was employed for software development.

The entire paper is comprised of 5 sections. Section 2 is entirely devoted to the concise survey of various existing encryption/decryption techniques. Section 3 covers the illustration of the system design and implementation details of the proposed system. Section 4 provides the analysis of results, and finally, the concluding remarks along with future scope have been elucidated in Section 5.

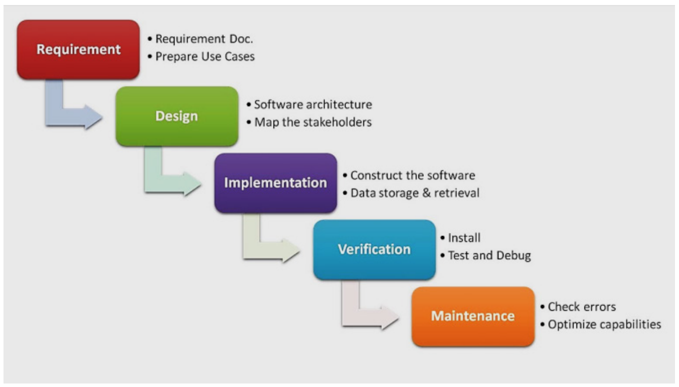


Figure 1. Waterfall model.

2. Literature Survey

In this section, some of the encryption techniques most frequently used by different researchers have been discussed.

A thorough study[20] of the popularly used symmetric-key encryption schemes like AES, Blowfish, DES and Triple DES was presented by authors. They observed the performance of symmetric key algorithms to be faster than the asymmetric key algorithms, and their memory requirement was lesser as compared to asymmetric algorithms. Besides, symmetric key encryption was found to be superior to asymmetric key encryption in terms of security. The authors concluded that Blowfish algorithm was better than DES, Triple DES, and AES when security and key size is considered.

In[21] compared three algorithms - DES, RSA, and AES based on parameters like computation time, output byte, and memory usages. As per the study, the longest encryption time was found in RSA, and it also had very high memory usage, but it had the least output byte. Furthermore, AES was observed to have the least memory usage, and DES had the least encryption time whereas these had minimal difference in their encryption time.

In a study[22], the authors compared four popular symmetric key algorithms – DES, AES, Triple DES, and Blowfish. The parameters analyzed included key size, round block size, CPU process time (in terms of power consumption and throughput) and encryption/decryption time. The authors concluded that triple DES showed the least performance of all and Blowfish was better compared to all the algorithms considered. Besides, it was found that AES is better than DES and triple DES based on the decryption time and throughput.

The authors in a study[23] analyzed symmetric and asymmetric algorithms viz. DES, RSA and triple DES based on the time they take for encrypting data, the ability to secure data and the throughput required. Based on the inputs, the algorithms showed varied performances. The study concluded that triple DES offers higher scalability and confidentiality than RSA and DES. However, DES takes less time for encryption/decryption as well as less power memory but is vulnerable to brute-force attack which makes it the least secure algorithm.

In another study[24], authors performed a comparative analysis of six algorithms – RC2, DES, AES, RC6, Blowfish and triple DES. The comparison was based on battery power consumption, encryption/decryption speed, data block size, key size and data types. They concluded that the results did not show any significant difference when displayed in Base 64 encoding or Hexadecimal Base encoding. Besides, they found Blowfish followed by RC6 performing better than the other algorithms when the packet size was changed. When data type was changed, RC6, Blowfish, and RC2 were observed to be disadvantageous based on time consumption. Furthermore, triple DES showed lesser performance than DES.

In[25] performed a comparison of symmetric key algorithms namely CAST-256, Blowfish, AES, and Two fish based on varying data loads. The parameters considered were key size, block size and speed. Taking the least time, Blowfish was concluded to be superior to other three algorithms. Although this difference was not clear for small data size, it was visible when file size is more than 100 Kb.

3. System Design and Implementation

3.1 Existing System

i. Problem with the current scenario
   - Lack of strong security while transmission of sensitive documents and other files using mobile phones
   - It is challenging for everyone to submit multiple copies of documents for availing some services.
   - Huge administrative overhead associated with the documents in physical form.
   - It is a challenge to share physical documents.
   - Verification of the authenticity of physical documents and other shared files is a challenge as well.

ii. Limitations of the existing system
   - It is very difficult to maintain the system.
   - There is every possibility of acquiring results that are erroneous.
   - It is very less user-friendly.
   - Minor bugs like sometimes the file upload is failing.
   - It consumes more time for processing the task.
   - Offline access is not provided.

3.2 Proposed System

The proposed security digital locker is an advanced application in which the file storage process is made very efficient and reliable as we don't have to carry every document with us every time. This application is a stand-alone app, and it does not have its own server. It makes use of the SQLite database server to store the username and secure pin of the user. The proposed system saves the uploaded document in an encrypted form and stores it in the Internal Memory, to avoid any kind of hacking internal or external. The files are secured in many ways which no intruder can access them in any case, such as the usage of AES Encryption. The main benefit of this system is that it asks for Secure Pin to access the file which we enter while registering ourselves. The Login validation checks the username and pin entered with the ones in the database and confirms or rejects login accordingly. This authentication mechanism employed ensures that if our phone is in someone's hand, they won't be able to access the files. The system then auto decrypts the files. Moreover, files cannot be lost using this locker system.

The system architecture is shown in Figures 2,3 shows the process of encryption and decryption of the plaintext file. The encryption process takes the plaintext file uploaded by the user as the input, and the decryption process takes the ciphertext file as the input. The input file is encrypted using AES algorithm with the help of a key and generates ciphertext file which is stored. After decryption, the ciphertext file generated can be passed through AES again employing the same key to generate back the plaintext file.

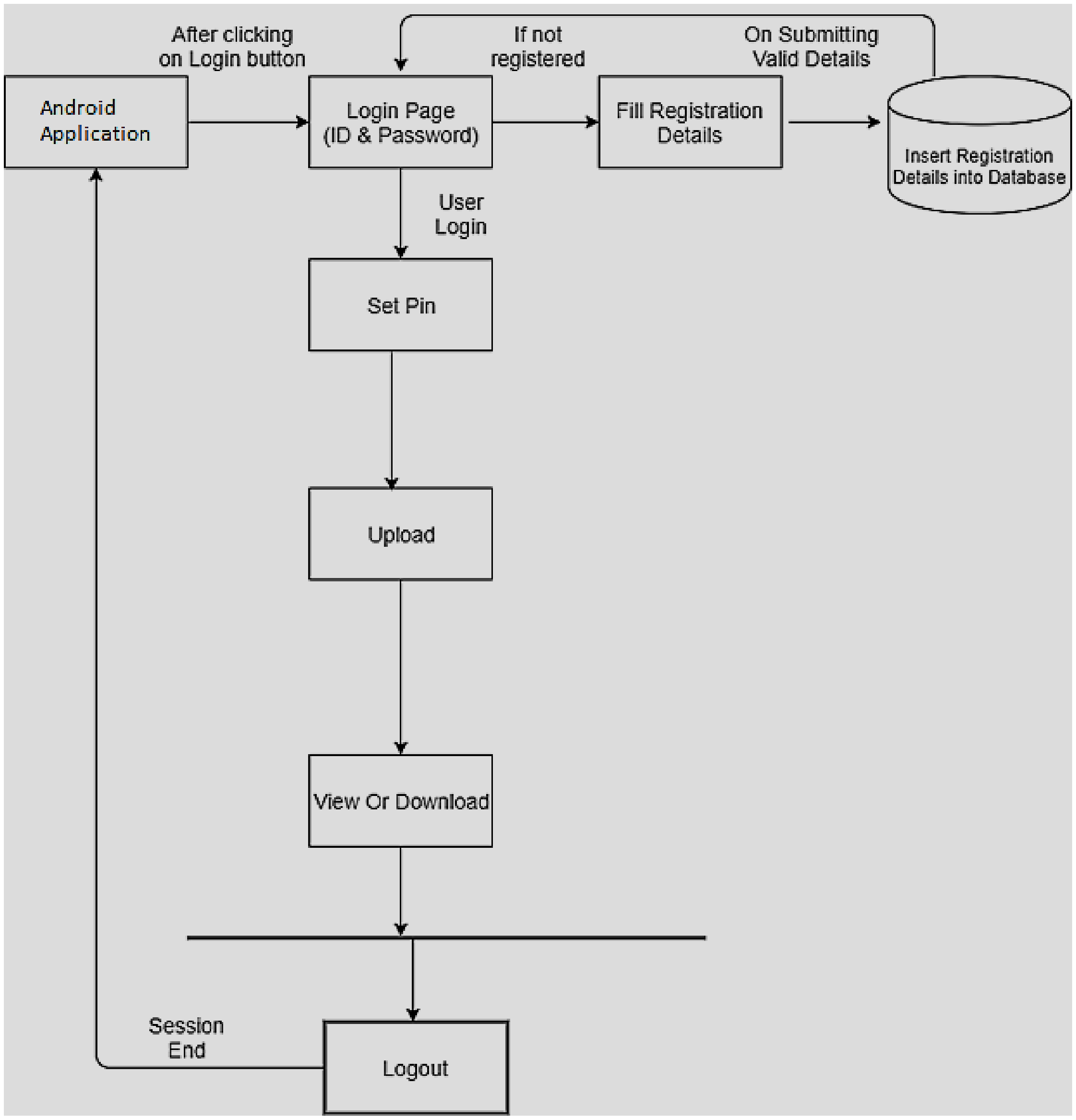

After clicking on Login button
If not registered
On Submitting Valid Details
Android Application
Login Page (ID & Password)
Fill Registration Details
Insert Registration Details into Database
User Login
Set Pin
Upload
View Or Download
Session End
Logout

Figure 2. System architecture.

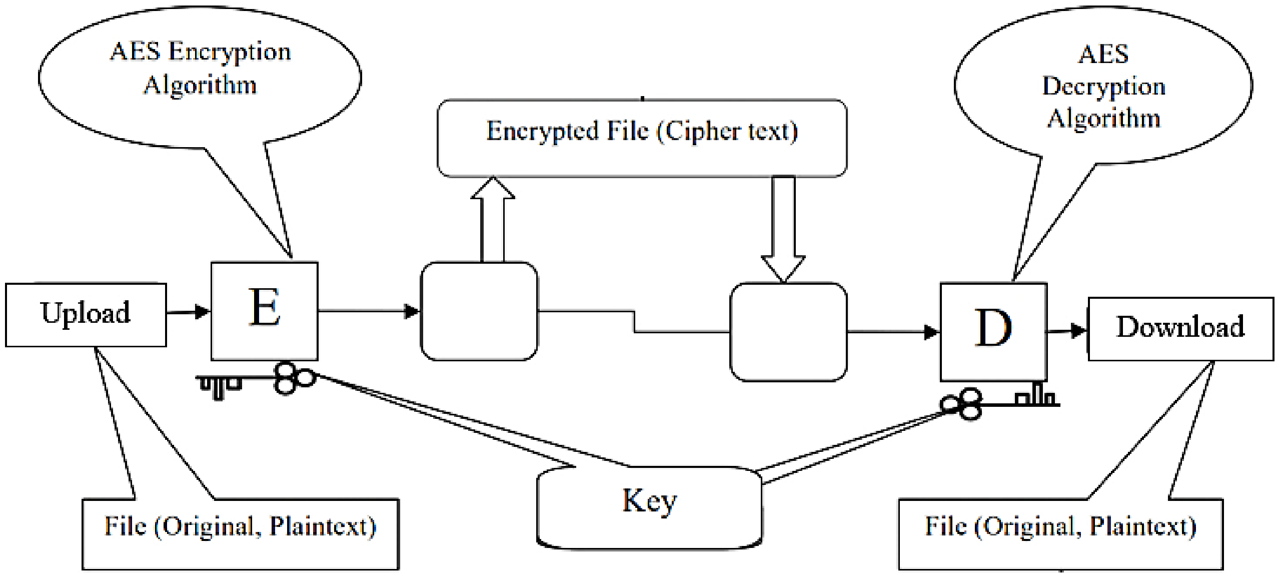

AES Encryption Algorithm
Encrypted File (Cipher text)
AES Decryption Algorithm
Upload
E
D
Download
Key
File (Original, Plaintext)
File (Original, Plaintext)

Figure 3. Encryption and decryption process.

3.3 System Design (Description of System Modules)

The system comprises of 5 major modules as follows:

i. Registration
The user has to register into the system with his basic details and create valid login credentials.
ii. Login
The user has to log in with credentials into the system, keeping the data secure.
iii. Set Pin
The user has to set a PIN for their account to enable the security of the documents and keeping the data safe from others to access.
iv. Upload
The user can upload a file after entering secured PIN, which is encryption and stored.
v. View/Download
The user can see all Encrypted file list of his/her document. The user can view any of the documents and also can download whenever required.

Decryption: Reverse operation of the file takes place on download and only after entering the secured pin, and the source file is downloaded and converted into the actual format.

The working of these modules can be better depicted with the help of the below-given representations of E-R diagram (Figure 4), use case model (Figure 5), sequence diagram (Figure 6), activity diagram (Figure 7), class diagram (Figure 8) and data-flow-diagrams (Figures 9–11).

3.4 Implementation

3.4.1 Implementation Technology

The proposed application is loaded in Android Studio. Android Studio has been used for designing and coding of the proposed system. All the databases were created as well as maintained into SQLite; there, tables were created, queries for storing data were written, and a record of the proposed system was maintained.

3.4.1.1 Hardware Requirements

i. PC or Laptop
- Based on i3 Processor
- 1GB Random Access Memory (RAM)
- 5 GB Hard Disk

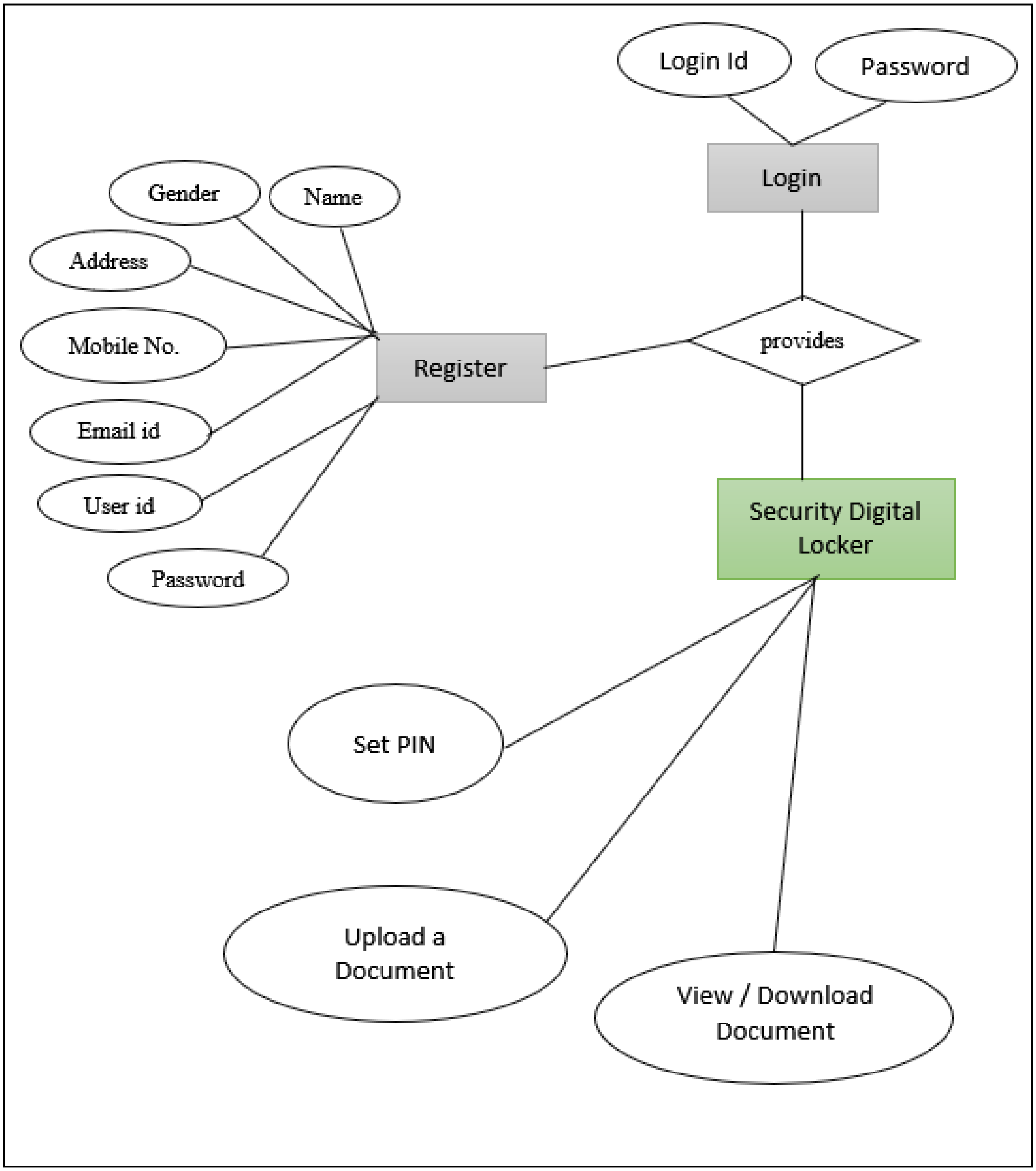


Figure 4. E-R diagram of the system.

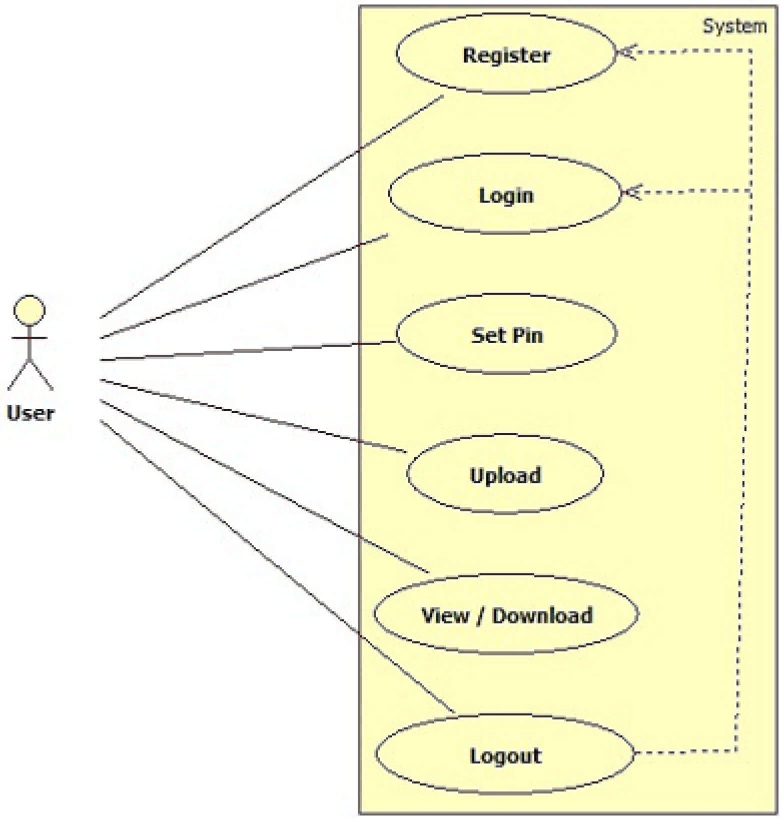


Figure 5. Use case diagram of the user.

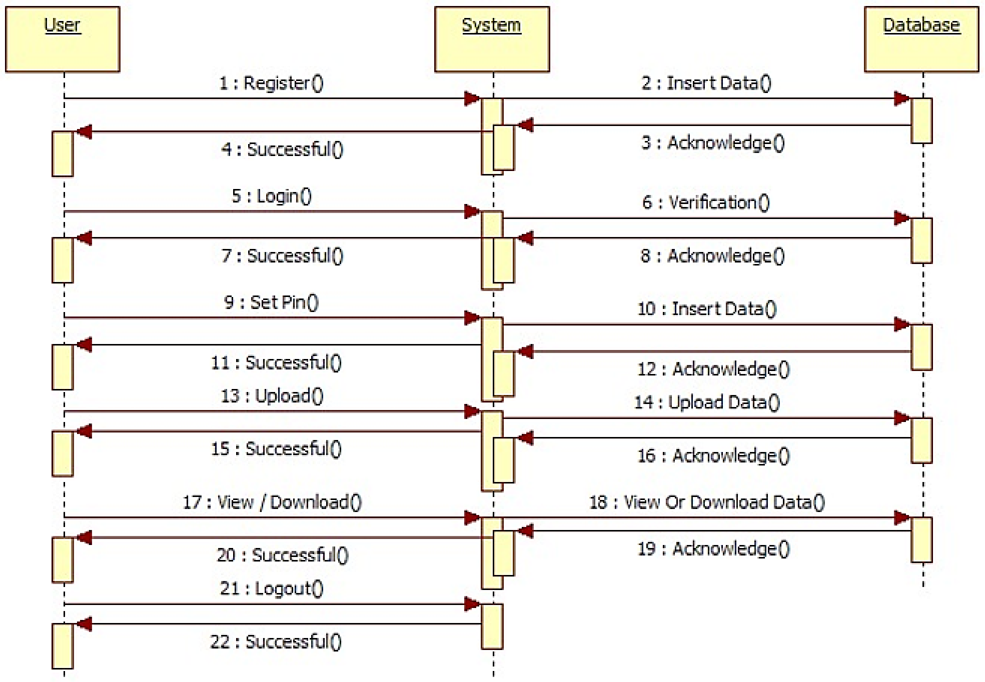


Figure 6. Sequence diagram of the user.

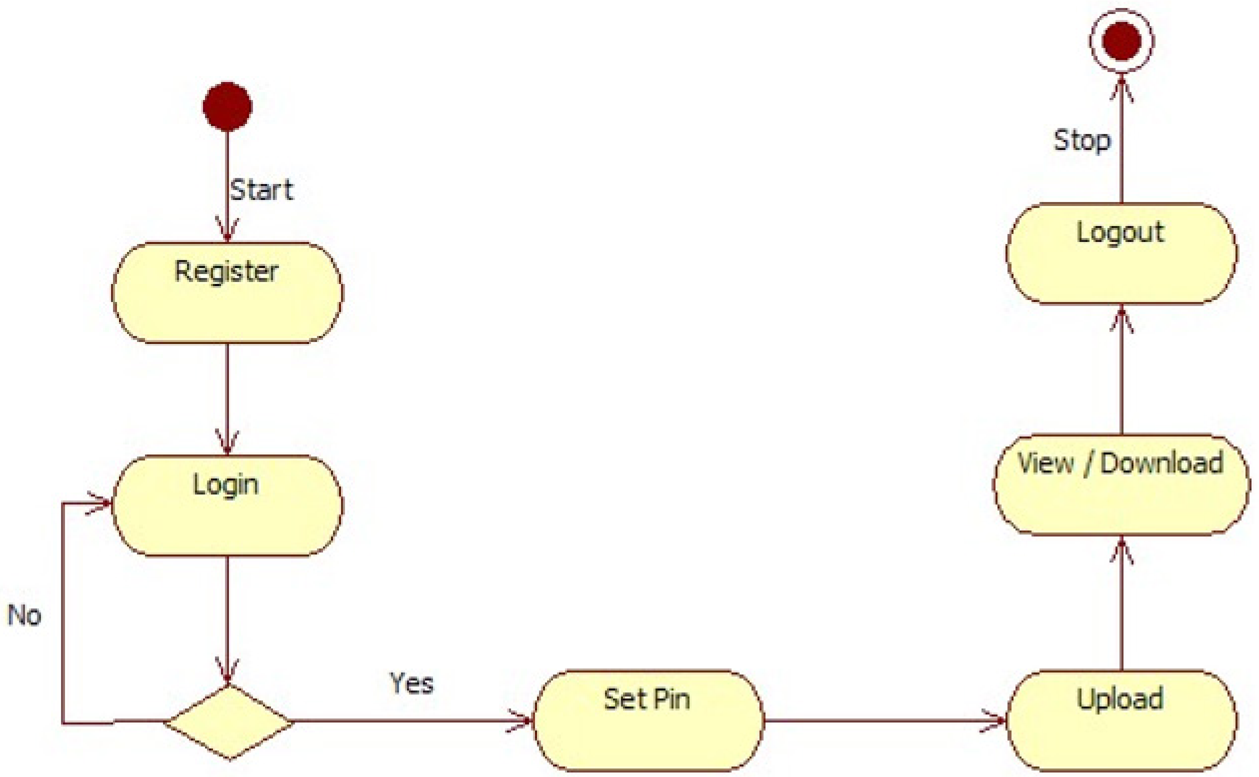


Figure 7. Activity diagram of the system.

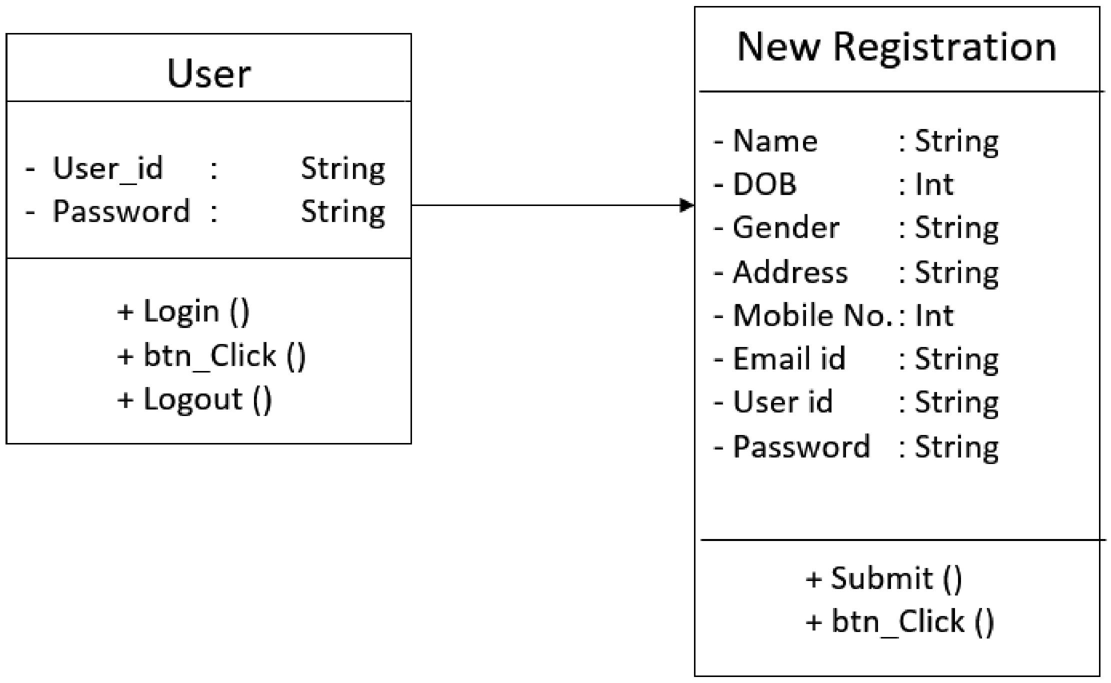


Figure 8. Class diagram of the system.

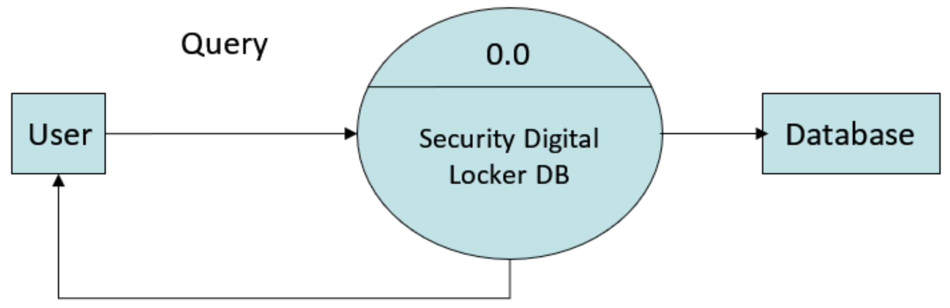


Figure 9. Database detail.

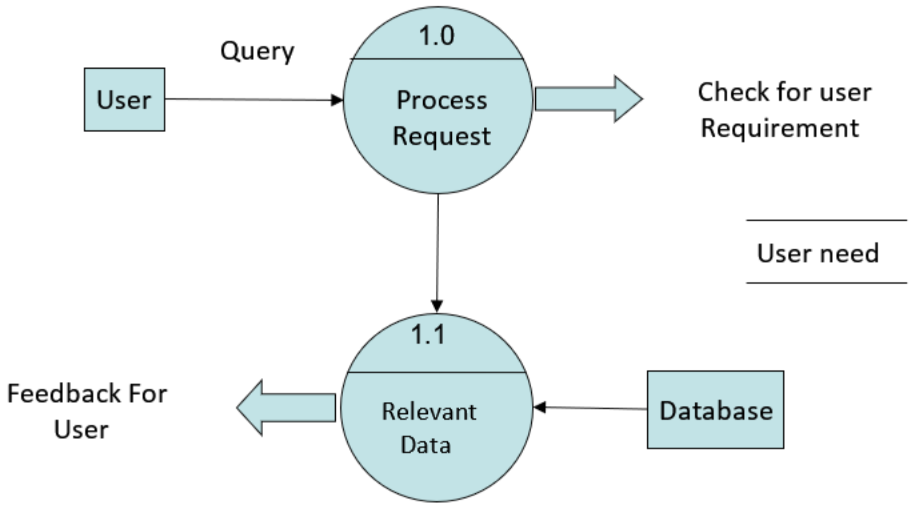


Figure 10. Level 1 DFD.

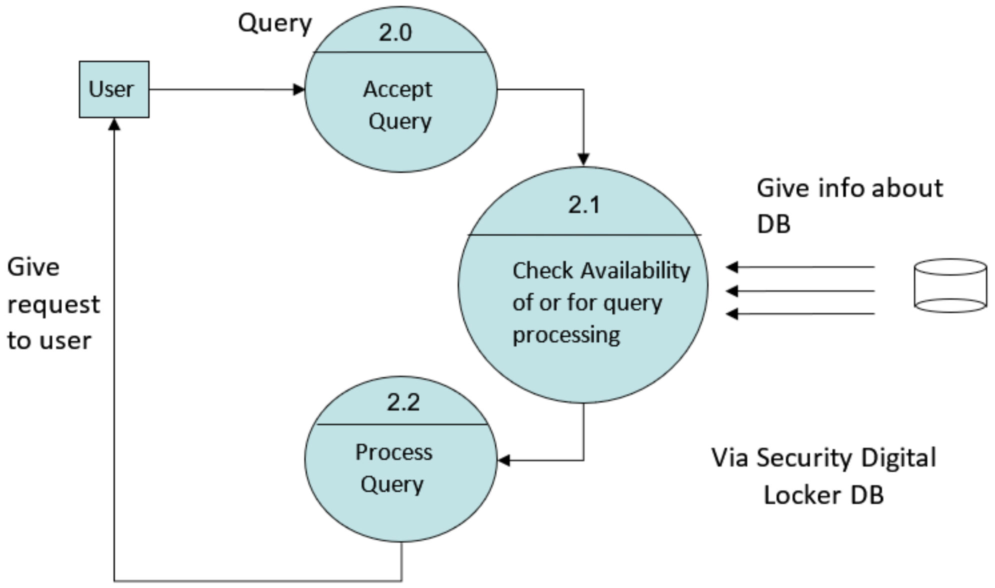


Figure 11. Level 2 DFD.

ii. Tablet or Android Phone
   - 1.2 Quad core or higher Processor
   - 1 GB Random Access Memory (RAM)

3.4.1.2 Software Requirements

i. Laptop or PC
   - Windows 7 or higher
   - SQLite
   - Java
   - Android Studio

ii. Tablet or Android Phone
   - Android version 5.0 or higher

The implementation details have been highlighted right from the installation to setting everything up. The Android SDK includes all tools as well as APIs required for writing Android applications. What remains is setting everything up in a proper manner so that the depths of application development can be delved into. To begin with Android development, the requirements are the Java Development Kit (JDK), the Android SDK and Java Integrated Development Environment (IDE) such as Eclipse for making the development process more comfortable and easier. For using Eclipse, the Android Development Tools (ADT) is also required besides a plug-in which provides additional support to IDE for Android development.

When using Eclipse, we also need the Android Development Tools (ADT), a plug-in for Eclipse that adds support for Android development to the IDE. Besides, the database support is provided by SQLite. The installation is recommended to proceed in the following order:

1. JDK - Java Development Kit
2. Eclipse
3. Android SDK
4. ADT - Android Development Tools
5. SQLite

3.4.2 Screenshots

The following screenshots may help to understand the working of the proposed system better:

i. The application opens up at the login page (Figure 12) where only registered users can sign in. The user has to register some details before being able to store any files. The registration page appears as shown in Figures 13,14 shows the error message generated in case already existing email is used by a new user.

After registration is complete, the user can sign in using his/her username and password (Figure 15). The user is then prompted to enter the encryption key (Figure 16) that is used by the AES algorithm for encrypting and decrypting the files. A user can upload any file (img, mp3, mp4, docx, pdf) as is visible in Figures 17,18. The app gives the provision of deleting the source file as well. Two options are provided for every file viz. Download and Delete (Figure 19) that let the user download or delete the files from the security digital locker.

The files uploaded are converted in text format which can be seen in Figure 20, and the encrypted version is shown in Figure 21.

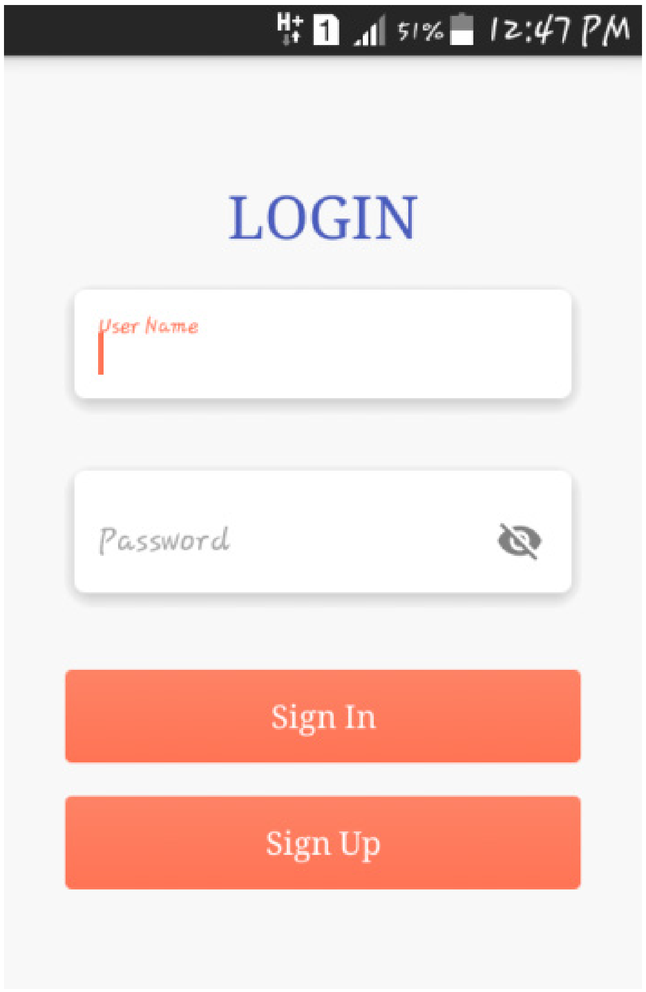


Figure 12. Login page.

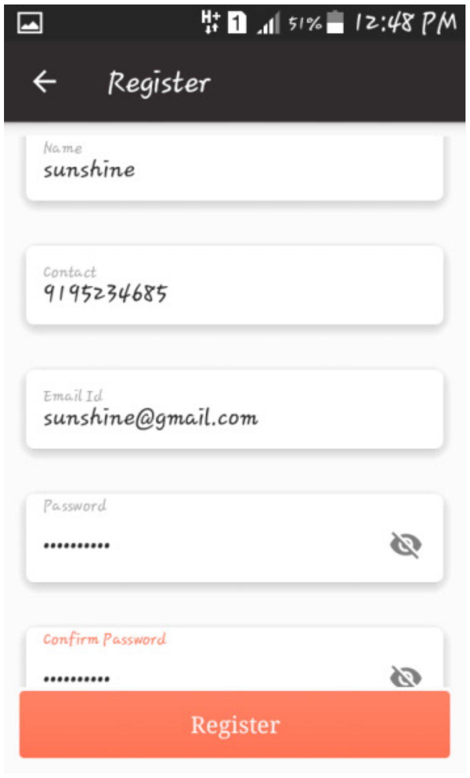


Figure 13. Registration page.

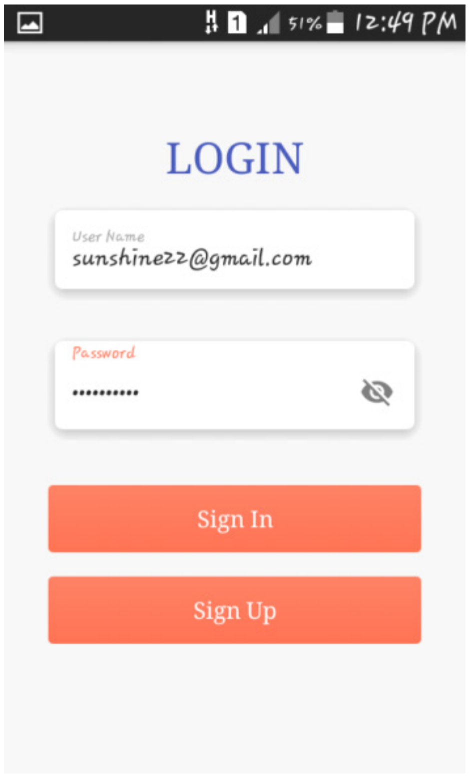
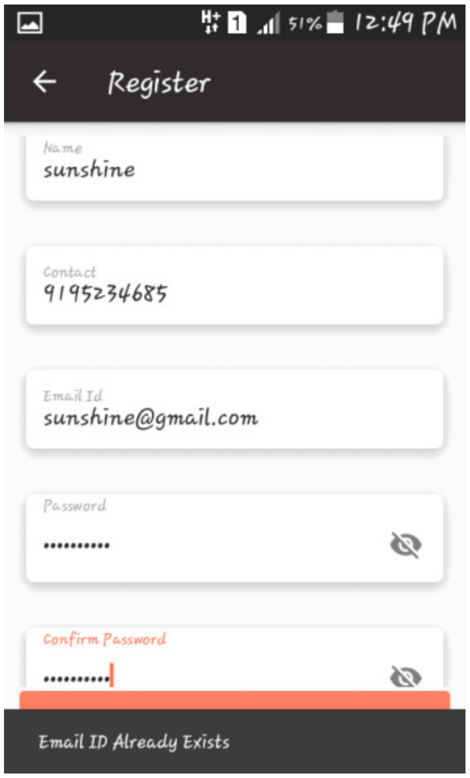


Figure 15. Registered user signing in.



Email ID Already Exists

Figure 14. Error message for wrong entry.

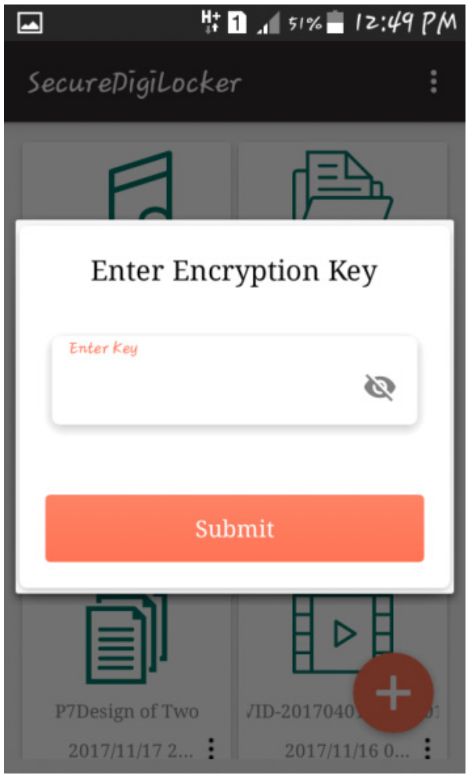


Figure 16. Setting the encryption key.

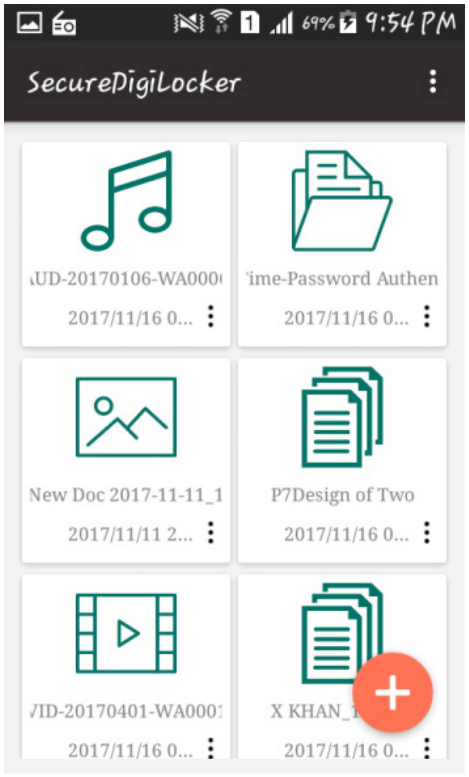


Figure 17. Types of files uploaded.

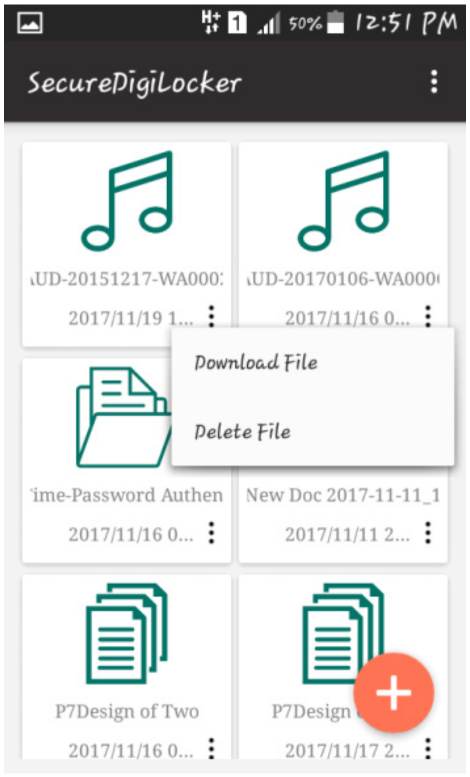
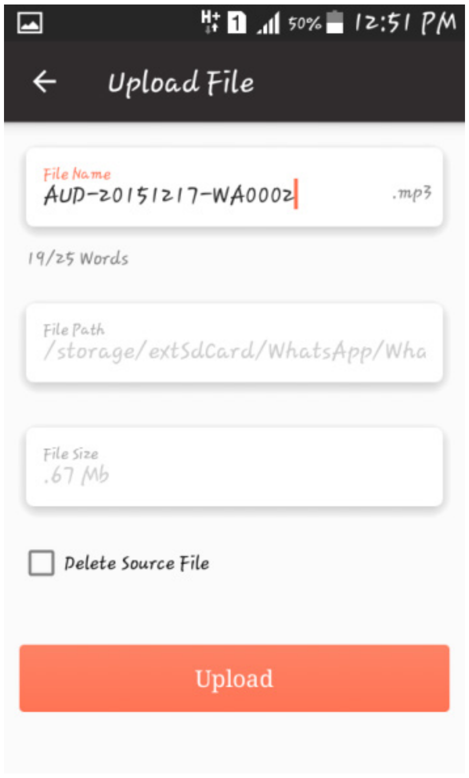
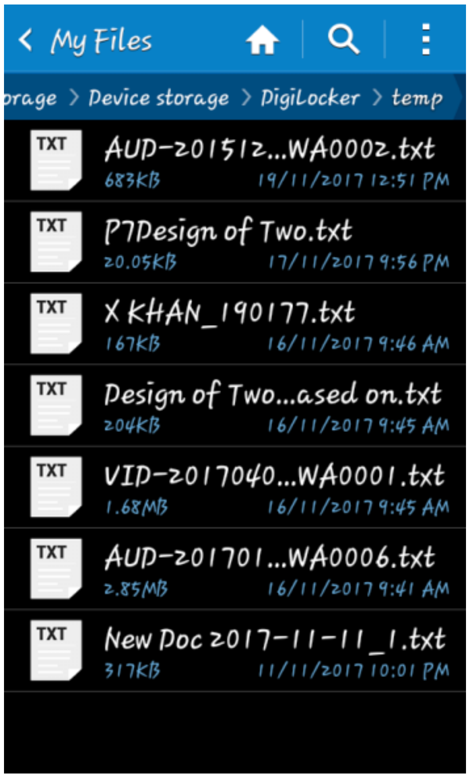


Figure 19. Options offered to user.



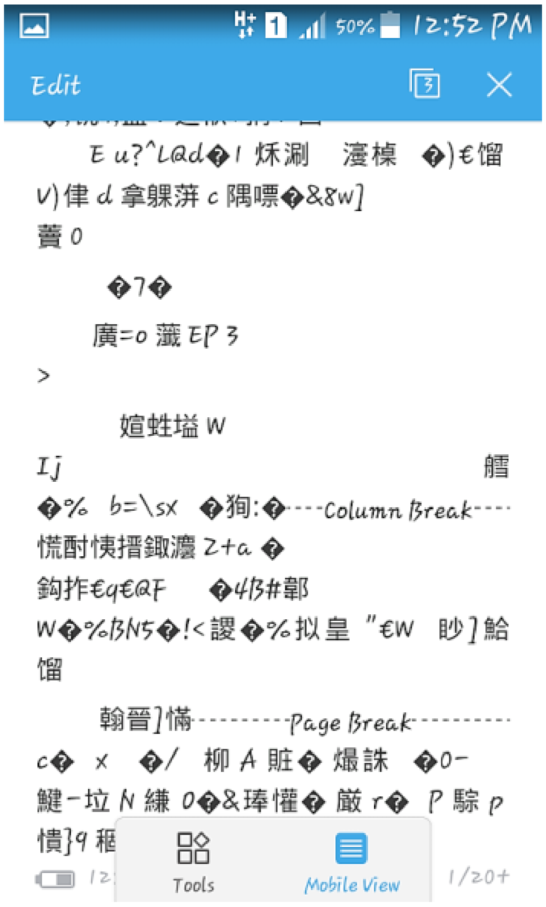
File Name
AUD-20151217-WA0002 .mp3
19/25 Words
File Path
/storage/extSdCard/WhatsApp/Wha
File Size
.67 Mb
Delete Source File
Upload

Figure 18. File uploaded by user.



TXT	AUD-201512...WA0002.txt	683KB	19/11/2017 12:51 PM
TXT	P7Design of Two.txt	20.05KB	17/11/2017 9:56 PM
TXT	X KHAN_190177.txt	167KB	16/11/2017 9:46 AM
TXT	Design of Two...ased on.txt	204KB	16/11/2017 9:45 AM
TXT	VID-2017040...WA0001.txt	1.68MB	16/11/2017 9:45 AM
TXT	AUD-201701...WA0006.txt	2.85MB	16/11/2017 9:41 AM
TXT	New Doc 2017-11-11_1.txt	317KB	11/11/2017 10:01 PM

Figure 20. Files saved in text format.



E u?^L@d�ı 烁涮 瀯榠 �)€馏
V)俥 d 拿躶荓 c 隅嘌�&8w]
薵 0
�7�
廣=o 瀻 EP 3
>
媗蛀塧 W
Ij 艚
�% b=\sx �狗:�----Column Break----
慌酎洟揺鍶瀘 Z+a �
鉤拃€q€@F �4B#鄢
W�%BN5�!<謖�%拟皇 "€W 眇]鮯
馏
翰晋]慉---------Page Break---------
c� x �/ 柳 A 赃� 熥誅 �0-
鞬-垃 N 繂 0�&琫懽� 敵 r� P 騌 p
慣}9 秬

Figure 21. Encrypted version of file.

4. Results

Since the proposed system is rather large-scale, testing is essential for making it successful. The proposed system shall be successful if all the components work properly in every respect and produce the desired result for any input. Thus, it can be concluded that the system has to be tested to make it successful.

In this case, System Testing was performed, i.e., it was checked if the requirements given by the user were met. The programming for the proposed system has been done using Java with front-end designing inculcated utilizing the interface of Android Studio. The proposed system was tested with the help of users, and every activity in the application was verified exclusively by the users.

Though some errors were found in the application but those have been rectified before implementation. The flow in the application forms has been observed to conform to the actual data flow.

4.1 Testing Levels

For uncovering the errors in various phases of the proposed system, testing can be performed on different levels. The most fundamental testing levels are given in Figure 22 as:

The proposed system underwent a series of testing phases before being submitted to the user for acceptance testing. The various steps involved the process of testing includes:

i. Unit Testing
Also referred to as "Module Testing," it is the process of verification on a module, i.e., the testing of the smallest unit in the system. In this kind of testing, each module is tested individually and is carried out while the system is being programmed[26]. When the proposed system was subjected to unit testing, it was found that every module was working in compliance with the output expected from the module.

ii. Integration Testing
If data is brought in using an interface, modules may have undesirable effects on one another. Integration testing is a type of systematic testing for constructing the structure of the program as well as conducting tests for uncovering errors in the interface. In this testing, all the modules are grouped and tested[27]. The provision to correct any errors is not easy in this testing since it is very difficult to locate the cause in the whole program. Therefore, in this stage of testing, the errors discovered are rectified for the subsequent testing steps.

iii. System testing
It is the implementation stage which focusses on ensuring the correct as well as the efficient working of the system for live operation[28]. System testing logically assumes that the goal shall be achieved successfully only if every component of the proposed system is working correctly.

iv. Validation Testing
After integration testing, the entire system is assembled as a package, discovery, and correction of interfacing errors has been done, and then the final stages of testing commence. Validation test is said to be successful when the proposed system works in the same way as expected by the end-user[29]. After conducting the validation test, there can be two conditions possible: one is that the performance characteristics comply with

the requirements specified and are accepted, the other condition being the discovery of deviations from the requirements which are enlisted. The proposed system was observed to work satisfactorily when validation testing was conducted on it.

v. Output Testing

If the system proposed does not generate the required results, it is of no use, and thus, output testing is performed after validation testing. This testing involves asking the end-users if the output produced by the system is in the pre-specified format. The format of the output is examined in two respects – the printed format and the screen. The proposed system was found to conform with the user's requirements in both respects. Thus, no corrections are made in the system during output testing.

vi. User Acceptance Testing (UAT)

The acceptance of the user is the main factor for the success of a system[30]. The proposed system was tested by keeping constant contact with the end users during development, and the changes were made as required.

4.2 Test Cases

User Login and Registration: Before logging in, the user has to register his/her details. Various fields are given in the registration form, all of which have to be filled up. The user is not allowed to enter a character in the field of login id.

Login: Login id, as well as a password, is the compulsory fields, and an error message shall be generated if the id and password do not match.

4.3 Validation Criteria

1. No field in the form (if not nullable) should be left empty.
2. No non-numeric values should be allowed in any numeric field. Likewise, no numeric characters should be allowed in text fields such as names, etc.
3. The user should be prevented from existing keys by generating all primary keys automatically.
4. Error handling for operations likes Edit, Save, Delete, etc.
5. Whenever data is entered in any textbox, there should be a provision of validating it, and a proper message should be flashed in case it is invalid.

5. Conclusion

This paper shows the successful implementation of file encryption as well as decryption for its secure storage in the mobile phones. The file may be image, text, pdf, ppt, mp3 or mp4. The users have observed faster file encryption or decryption since the AES algorithm runs faster on

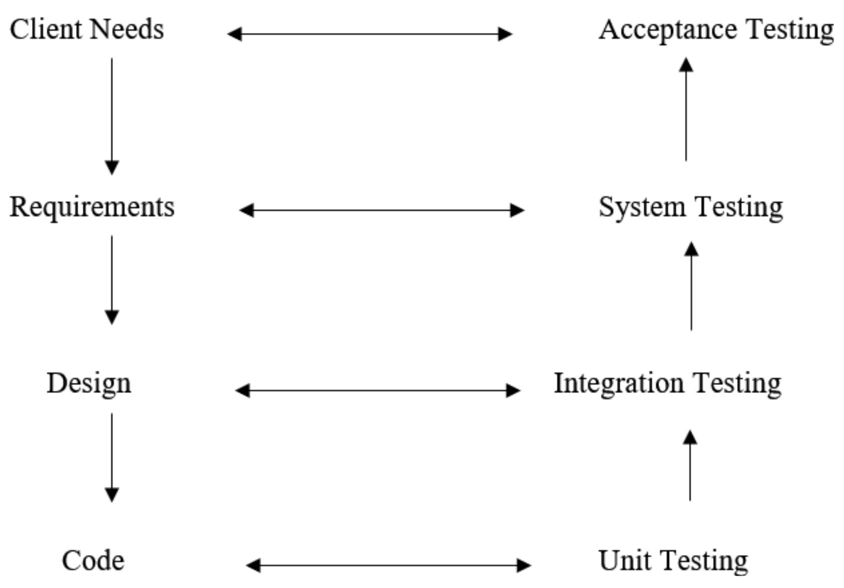


Figure 22. Levels of testing.

an Android phone. It provides better security for mobile phones from unauthorized access. This application ensures secure storage of data without any corrupt data. In future, the work might be extended by employing a stronger encryption algorithm that has higher speed and less memory usage.

6. References

1. Mobile Fact Sheet [Internet]. [cited 2018 Feb 05]. Available from: http://www.pewinternet.org/fact-sheet/mobile/.
2. T-Mobile G1 Event Round-up [Internet]. [cited 2008 Sep 23]. Available from: http://www.talkandroid.com/260-t-mobile-g1-details/.
3. A brief history of Android phones [Internet]. [cited 2011 Aug 02]. Available from: https://www.cnet.com/news/a-brief-history-of-android-phones/.
4. Android vs iOS: how they compare (updated for Android Nougat and iOS 10) [Internet]. [cited 2016 Oct 24]. Available from: https://www.androidauthority.com/android-vs-ios-682005/.
5. Mehraj T, Rasool B, Khan BU, Baba A, Lone AG. Contemplation of effective security measures in access management from adoptability perspective. International Journal of Advanced Computer Science and Applications. 2015; 6(8):188–200. Crossref.
6. Rasool B, Mehraj T, Khan BU, Baba A, Najar ZA. Securely eradicating cellular dependency for e-banking applications. International Journal of Advanced Computer Science and Applications. 2018; 9(2):385–98. Crossref.
7. Android Application to Integrate and Secure Social Networks and Emails [Internet]. [cited 2018 Feb 26]. Available from: https://www.lifewire.com/social-media-apps-for-managing-everything-3486302.
8. Android Security 2016 Year in Review [Internet]. [cited 2017 Mar]. Available from: https://source.android.com/security/reports/Google_Android_Security_2016_Report_Final.pdf.
9. Olanrewaju RF, Khan BU, Najeeb AR, Zahir KN, Hussain S. Snort-based smart and swift intrusion detection system. Indian Journal of Science and Technology. 2018 Jan; 11(4):1–9. Crossref.
10. Ongtang M, McLaughlin S, Enck W, Mc Daniel P. Semantically rich application-centric security in Android. Security and Communication Networks. 2012 Jun; 5(6):658–73. Crossref.
11. Joshi J, Parekh C. Android smartphone vulnerabilities: a survey. International Conference on Advances in Computing, Communication and Automation; 2016. p. 1–5. Crossref.
12. Mir MS, Suhaimi B, Adam M, Khan BUI, Mattoo MUI, Olanrewaju RF. Critical security challenges in cloud computing environment: An appraisal. Journal of Theoretical and Applied Information Technology. 2017 May; 95(10):2234–48.
13. Olanrewaju RF, Khan BU, Mattoo MM, Anwar F, Nordin AN, Mir RN. Securing electronic transactions via payment gateways–a systematic review. International Journal of Internet Technology and Secured Transactions. 2017; 7(3):245–69.Crossref.
14. Masihuddin M, Khan BU, Mattoo MM, Olanrewaju RF. A survey on e-payment systems: elements, adoption, architecture, challenges and security concepts. Indian Journal of Science and Technology. 2017 May; 10(20):1–19. Crossref.
15. Hussain S, Khan BU, Anwar F, Olanrewaju RF. Secure annihilation of out-of-band authorization for online transactions. Indian Journal of Science and Technology. 2018; 11(5):1–9. Crossref.
16. Khan BU, Olanrewaju RF, Mir RN, Yusoff SH, Sanni ML. Trust and resource oriented communication scheme in mobile ad hoc networks. Proceedings of SAI Intelligent Systems Conference; 2016. p. 414–30.
17. Khan BU, Olanrewaju RF, Baba AM, Langoo AA, Assad S. A compendious study of online payment systems: Past developments, present impact, and future considerations. International Journal of Advanced Computer Science and Applications. 2017 May; 8(5):256–71.
18. Olanrewaju RF, Khan BU, Mir RN, Shah A. Behaviour visualization for malicious-attacker node collusion in MANET based on probabilistic approach. American Journal of Computer Science and Engineering. 2015 Mar; 2(3):10–19.
19. Khan BUI, Olanrewaju RF, Mir RN, Baba A, Adebayo BW. Strategic profiling for behaviour visualization of malicious node in manets using game theory. Journal of Theoretical and Applied Information Technology. 2015 Jul; 77(1):25–43.
20. Agrawal M, Mishra P. A comparative survey on symmetric key encryption techniques. International Journal on Computer Science and Engineering. 2012 May; 4(5): 877–82.
21. Seth SM, Mishra R. Comparative analysis of encryption algorithms for data communication. International Journal of Computer Science and Technology. 2011; 2(2):292–4.
22. Mandal PC. Superiority of Blowfish algorithm. International Journal of Advanced Research in Computer Science and Software Engineering. 2012 Sep; 2(9): 196–201.
23. Marwaha M, Bedi R, Singh A, Singh T. Comparative analysis of cryptographic algorithms. International Journal of Advanced Engineering Technology. 2013; 4(3):16–18.

24. Elminaam DA, Kader HA, Hadhoud MM. Performance evaluation of symmetric encryption algorithms. Communications of the IBIMA. 2009 Apr; 8:58–64.
25. Apoorva YK. Comparative study of different symmetric key cryptography. International Journal of Application or Innovation in Engineering and Management. 2013; 2(7):204–6.
26. Software Testing [Internet]. [cited 2018 Mar 06]. Available from: https://en.wikipedia.org/wiki/Software_testing. Date accessed: 06/03/2018.
27. INTEGRATION Testing Tutorial: Big Bang, Top Down & Bottom Up [Internet]. [cited 2018 Feb 07]. Available from: https://www.guru99.com/integration-testing.html. Date accessed: 07/02/2018.
28. System Testing: What? Why? & How? [Internet]. [cited 2012 Oct 01]. Available from: www.softwaretestingclass.com/system-testing-what-why-how/.
29. Validation Testing [Internet]. [cited 2016 Jun 22]. Available from: https://www.quora.com/What-is-validation-testing.
30. What is User Acceptance Testing (UAT) and How to Perform It Effectively? — Software Testing Help [Internet]. [cited 2018 Feb 22]. Available from: http://www.softwaretestinghelp.com/what-is-user-acceptance-testing-uat/.