Chapter 1

INTRODUCTION

1.1 Overview of Data Encryption app

A textual data encryption application is a software tool or system that is designed to encrypt and decrypt textual data to ensure its confidentiality and security. Here is an overview of how such an application typically works. The application allows the user to input the text that needs to be encrypted. This can be done through a graphical user interface (GUI) or a command-line interface (CLI). The application utilizes an encryption algorithm to convert the plaintext into cipher text. Common encryption algorithms used for textual data include AES, RSA, and Blowfish. The algorithm takes the plaintext and a secret encryption key as input and produces the cipher text. The user may be required to provide an encryption key or password that will be used to encrypt and decrypt the data. The key should be kept secret and should be strong enough to resist attacks.

1.2 Applications of Data Encryption app

Textual data encryption applications have numerous applications in various domains where data security and confidentiality are of utmost importance. Encryption applications are widely used to secure communication channels, such as email, messaging apps, and chat platforms. By encrypting the text messages, the application ensures that only the intended recipient can read the content, protecting it from eavesdropping or interception. Textual data encryption applications are used to secure sensitive information stored in databases, files, or cloud storage. By encrypting the data before storage, unauthorized access to the information is prevented, even if the storage medium is compromised Encryption applications can be used to encrypt files before sharing them with others. This ensures that only authorized individuals with the decryption key can access and decrypt the file, maintaining its confidentiality during transit.

1.3 Problem statement

Many individuals and organizations handle sensitive textual data that needs to be protected from unauthorized access and potential data breaches. The current lack of a reliable and user-friendly textual data encryption application poses a significant security risk, as sensitive information can be easily intercepted or accessed by malicious actors. Existing encryption solutions may be complex, difficult to implement, or lack proper integration into commonly used communication platforms and storage systems. Therefore, there is a need for a robust, user-friendly, and efficient textual data encryption application that can securely encrypt and decrypt sensitive textual data. This application should provide strong encryption algorithms, easy-to-use interfaces, and seamless integration with communication platforms, storage systems, and other relevant software. It should ensure that only authorized individuals with the correct decryption key can access the encrypted data, protecting it from unauthorized disclosure, interception, or misuse.

1.3 Objectives

The main objective of our project is to encrypt/decrypt the textual files for personal and professional security. Encryption and Decryption protects privacy of our email messages, documents and sensitive files by encrypting them using AES algorithm to provide high protection against unauthorized data access. Every day hundreds and thousands of people interact electronically, whether it is through emails, e-commerce, etc. through internet. The Internet is comprised of millions of interconnected communication and transfer of information around the world. People use emails to correspond with one another . The www is used for online business, data distribution, marketing, research, learning and a myriad of other activities. Sending sensitive messages over the Internet is very dangerous as all emails are transmitted in an unsecured form and anybody - ISP, your boss, etc. can read your emails. If you want to send sensitive information via email, simply paste the encrypted text into your email or attach the encrypted file, all the recipient has to do is to decrypt your text or file. Encryption and Decryption works with text information and files. Just select what you want to encrypt, and Encryption and Decryption software helps you keep documents, private information and files in a confidential way.

1.5 Overview of android

Architecture:

Android operating system is a stack of software components which is roughly divided into five sections and four main layers.

Linux kernel:

At the bottom of the layers is Linux - Linux 3.6 with approximately 115 patches. This provides a level of abstraction between the device hardware and it contains all the essential hardware drivers like camera, keypad, display etc. Also, the kernel handles all the things that Linux is really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.

Libraries:

On top of Linux kernel there is a set of libraries including open-source Web browser engine Web Kit, well known library, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc.

Android Libraries:

This category encompasses those Java-based libraries that are specific to Android development. Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access.

Android Runtime:

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called Dalvik Virtual Machine which is a kind of Java Virtual Machine specially designed and optimized for Android. The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine. The Android runtime also provides a set of core libraries which enable Android application developers to write Android, EV applications using standard Java programming language.

Application Framework:

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.

Features of android:

Android is a powerful operating system competing with Apple 4GS and supports great features.Few of them are listed below.

Beautiful UI: - Android OS screen provides a beautiful and intuitive user interface.

Connectivity: - GSM/EDGE, IDEN, CDMA, EV -DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC and wimax.

Storage: - SQLite, a lite weight relational database, is used for data storage purpose.

Media Support: - H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1,MP3, MIDI, WAV, JPEG, PNG, GIF, and BMP.

Messaging: - SMS and MMS.

Web Browser: - Based on the open-source Web Kit layout engine, coupled with Chrome's V8JavaScript engine supporting HTML5 and CSS3.

Multitouch: - Android has native support for multi-touch which was initially made available inhandsets such as the HTC Hero.

Multi-Tasking: - User can jump from one task to another and same time various applicationcan run simultaneously.

Resizable Widgets: - Widgets are resizable, so users can expand them to show more content orshrink them to save space.

Multi Language: - Supports single direction and bi-directional text.

GCM: - Google Cloud Messaging (GCM) is a service that lets developers send short messagedata to their users on Android devices, without needing a proprietary sync solution.

Wi-Fi Direct: - A technology that lets apps discover and pair directly, over a high-bandwidthpeer-to-peer connection.

Android Beam: - A popular NFC-based technology that lets users instantly share, just bytouching two NFC-enabled phones together.

1.6 Overview of Java

Features:

1. **Cross-Platform Development:**

Java's platform independence allows developers to write code once and run it on multiple platforms. By using Java for mobile app development, you can target both Android and non-Android platforms (e.g., J2ME), reaching a wider audience with a single codebase.

2. **Native-Like Performance:**

Java's efficient bytecode execution and Just-In-Time (JIT) compilation capabilities enable mobile applications to achieve native-like performance. Java's performance optimizations make it suitable for resource-intensive tasks and demanding mobile app requirements. Using Java for mobile app development allows developers to achieve cross platform compatibility, performance, security, code reusability, and integration with existing systems.

3. **Rich Standard Library:**

Java provides a rich standard library with a wide range of classes and APIs that can be leveraged for mobile app development. This library includes networking, file I/O, database connectivity, user interface components, and much more, reducing the need for third-party libraries and simplifying development.

4. **Development Tools and Ecosystem:**

Java benefits from a robust and mature development ecosystem. It offers a variety of integrated development environments (IDEs), such as Android Studio and Eclipse, which provide comprehensive tools for debugging, testing, and profiling mobile applications. Additionally, Java has a vast community and extensive documentation, making it easier for developers to find support and resources.

5. **Security and Stability:**

Java places a strong emphasis on security, and this applies to mobile app development as well. Java provides features like sandboxing, secure coding practices, and cryptographic libraries, allowing developers to build secure mobile applications.

6. **Multithreading and Concurrency:**

Java's support for multithreading and concurrency is advantageous in mobile app development. Mobile apps often require background tasks, network requests, and parallel processing. Java's built-in support for threads and concurrent programming simplifies the implementation of such functionality.

7. **Code Reusability:**

Java's object-oriented nature promotes code reusability. With proper architecture and design patterns, developers can write reusable components and modules, saving time and effort in mobile app development. This is particularly useful when building multiple mobile apps or maintaining different versions of the same app.

8. **Integration with Existing Systems:**

Many enterprises already use Java for their backend systems. By leveraging Java for mobile app development, developers can seamlessly integrate mobile apps with existing Java-based backend systems, databases, and APIs.

Advantages over java:

1. Maturity and Ecosystem: Java has been around for a long time and has a mature ecosystem with a vast number of libraries, frameworks, and tools. It has a larger community and a more extensive range of resources, making it easier to find support and solutions to problems.

2. Compatibility: Java has a high level of compatibility with existing Java codebases, libraries, and frameworks. It allows developers to seamlessly integrate new Java code with existing Java systems. This is particularly beneficial when working on large-scale projects or maintaining legacy code.

3. Tooling and IDE Support: Java has excellent tooling and Integrated Development Environment (IDE) support, with robust IDEs like Eclipse, IntelliJ IDEA, and NetBeans. These IDEs provide powerful features, such as code auto completion, refactoring tools, debugging support, and project management capabilities.

4. Performance: Java's runtime performance is generally considered to be faster than Kotlin's. Java's long history of optimization and JVM's efficient Just-In-Time (JIT) compilation contribute to its performance advantage, especially in computationally intensive applications.

5. Adoption and Industry Support: Java has been widely adopted by the industry for many years. It is often the primary language for enterprise software development, and many large organizations and industries rely on Java-based systems. This widespread adoption ensures a strong job market and career opportunities for Java developers.

Role of Java in app development:

Java is the preferred language for Android development in 2021. Both Java and Kotlin can be used to build performant, useful applications, but Google's libraries, tooling, documentation, and learning resources, making it the better language for Android today, offering a robust and versatile platform for building applications.

Chapter2

DESIGN AND IMPLEMENTATION

2.1 Functional Requirements

Hardware Requirements:

1) Windows: Windows 11

2) Processor: Intel (R) Core(TM) i5-7200U CPU@ 2.50 GHz 2.70 GHz

3) RAM: 8GB

4) System type: 64-bit Operating system, x64-based Processor

Software Requirements:

1) Platform: Android studio Arctic fox-2020.3.1 Patch 2

2) AVD Manger:

☆ Name: Pixel 2 API 30

☆ Resolution: 1080x2220; 440 dpi

☆ API: 30

☆ Target: Android 11.0 (Google play)

☆ CPU/ABI: x86

☆ Size on disk: 12GB

3) Minimum SDK: API 25: Android 7.1.1 (Nougat)

4) Language: Java

2.2 Design:

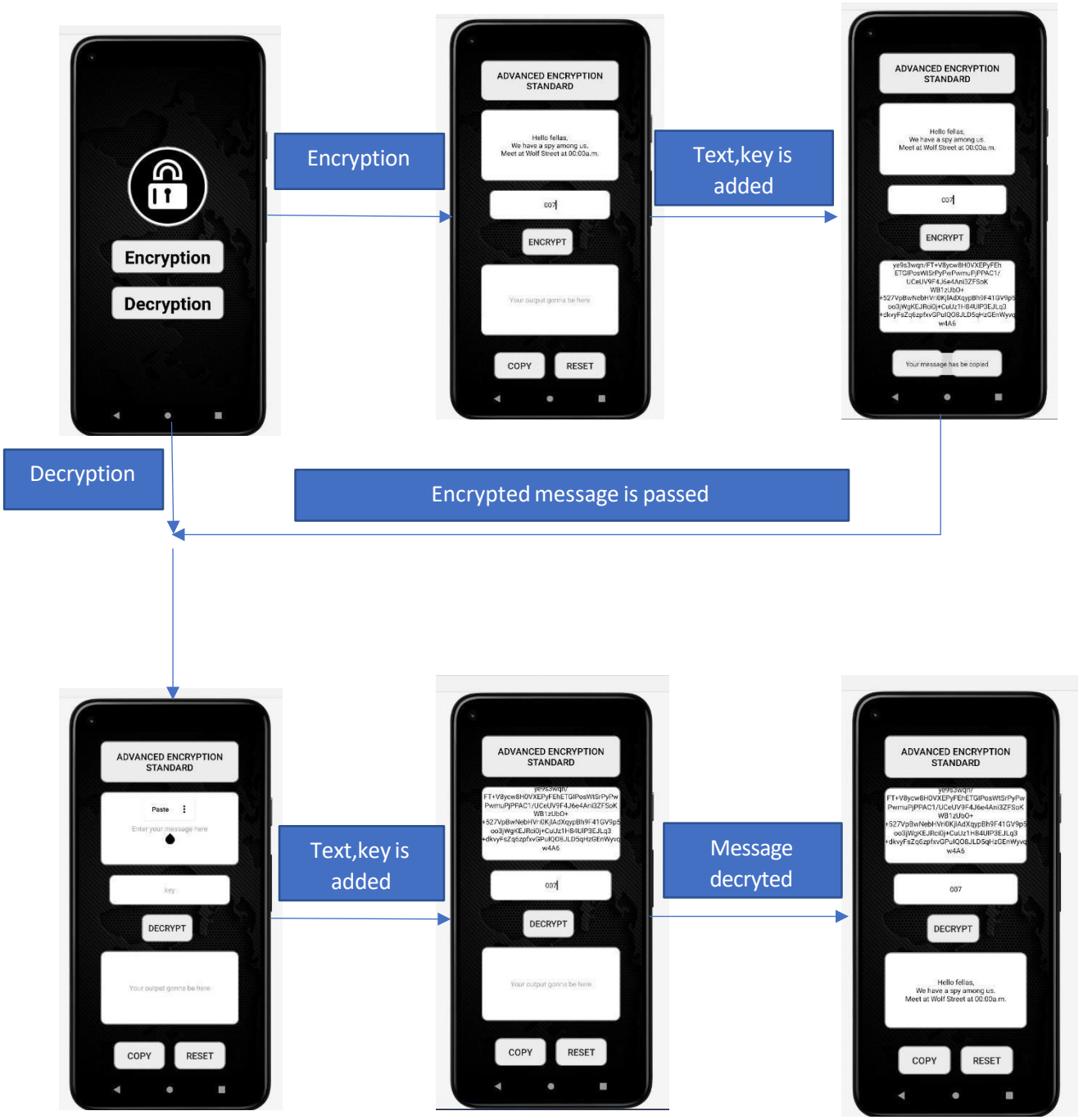

Fig 2.1 Blueprint of Data Encryption app

2.3 Implementation

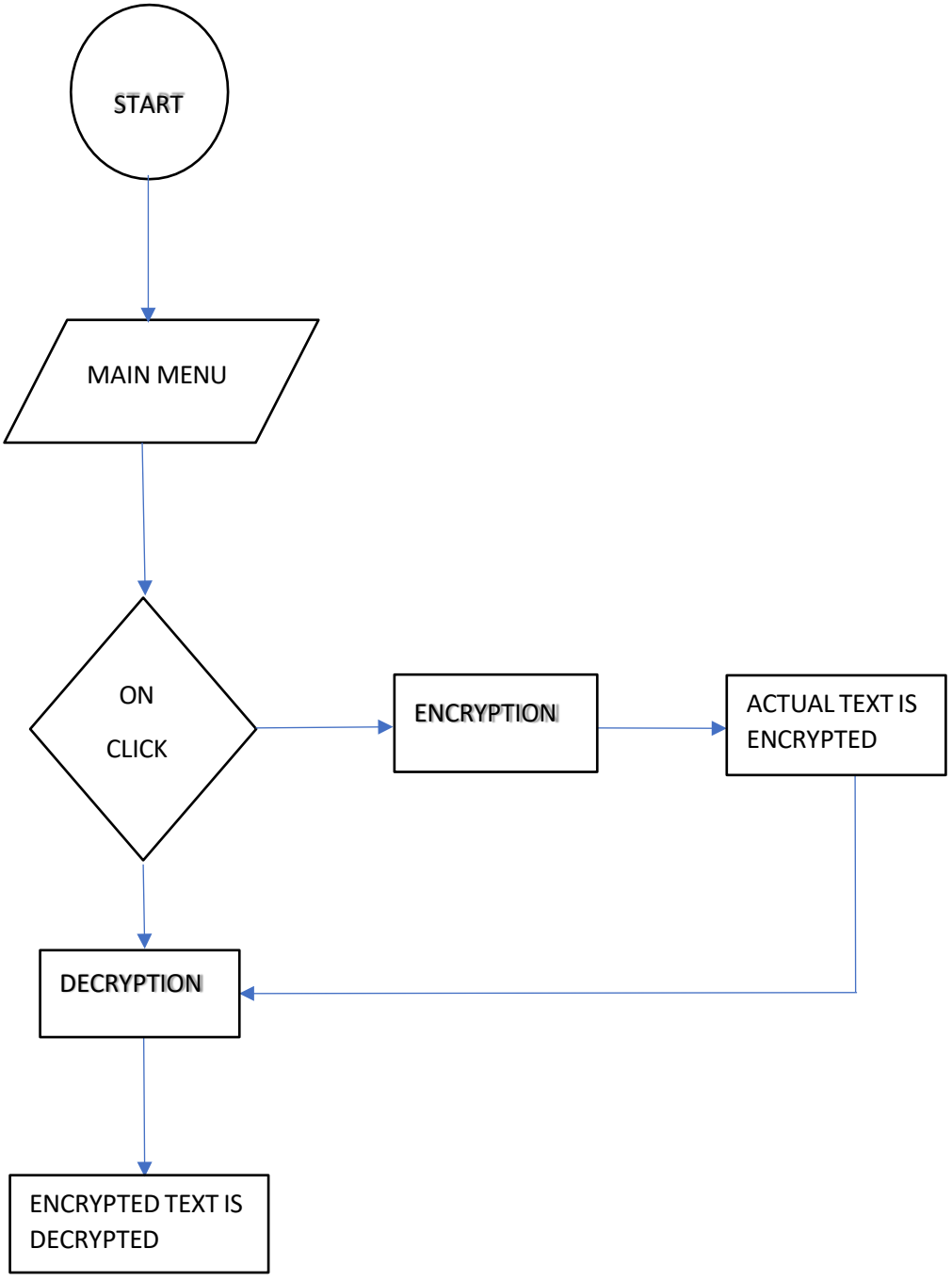

Fig 2.2 Flowchart of Data Encryption App

2.4 API used

1. Content.Intent: This class is used to represent an intent, which is a messaging object used to request an action from another component of the Android system or to communicate between components. In this code, an intent is used to start the `MainActivity` after the splash screen.

2. Os.Bundle: This class is used to pass data between Android components. It is commonly used with intents to carry extra data. In this code, a `Bundle` is not directly used, but it is the parameter for the `onCreate` method.

3 Os.Handler: This class is used to schedule tasks to be executed at a later time. In this code, a `Handler` is used to delay the start of the `MainActivity` using the `postDelayed` method.

4. View.WindowManager: This class provides access to the system window manager. The code uses it to set flags on the window, such as making it fullscreen.

5. View.animation.Animation: This class represents an animation that can be applied to views. The code uses it to load and apply animations to the `LogoImg` ImageView.

6. Widget.ImageView: This class represents a view that displays an image. The code uses it to reference the `ImageView` for the application logo.

7. Widget.TextView: This class represents a view that displays text. The code imports it but doesn't use it in the provided code.

8. Appcompat.app.AppCompatActivity: This is the base class for activities that use the support library action bar features. The code extends this class to create the `SplashActivity` activity.

Chapter 3

RESULTS

Snapshots

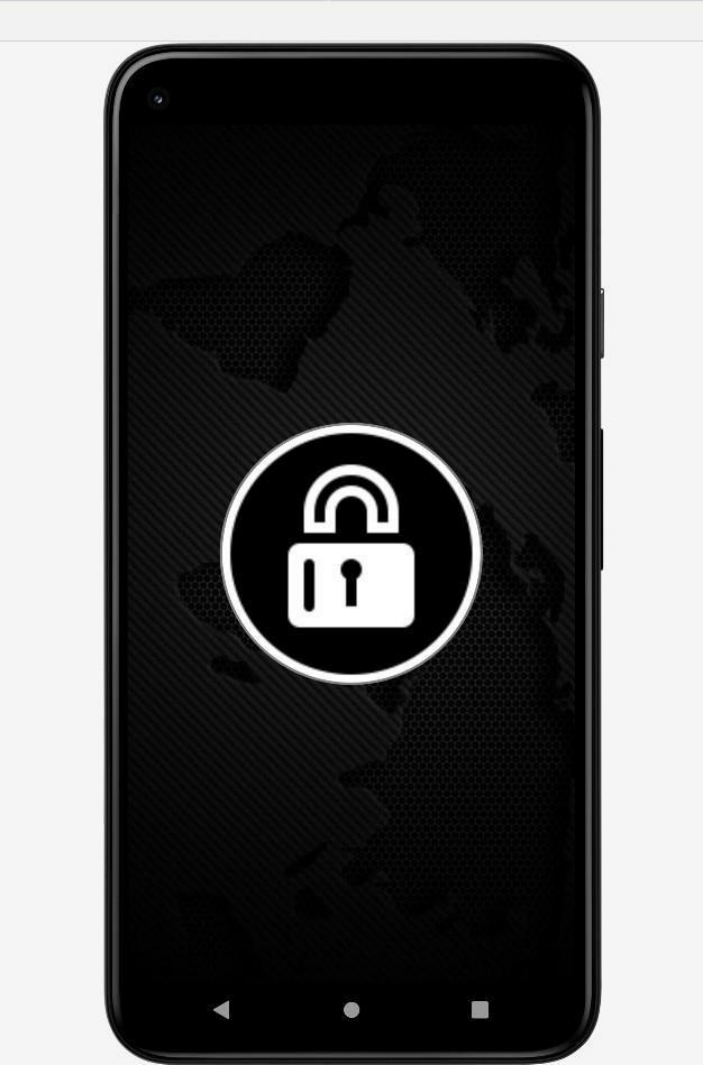

Fig 3.1 Splash screen.

Fig 3.1 is the splash screen which shows the symbol of our application

Fig 3.2 Menu screen.

Fig 3.2 is the menu screen which has two options for encryption and decryption.

Fig 3.3 Encryption screen.

Fig 3.3 is the page which shows the encryption screen.

Fig 3.4 Text copied.

Fig 3.4 is an example for copying the encrypted text upon clicking the copy button.

Fig 3.5 Main screen.

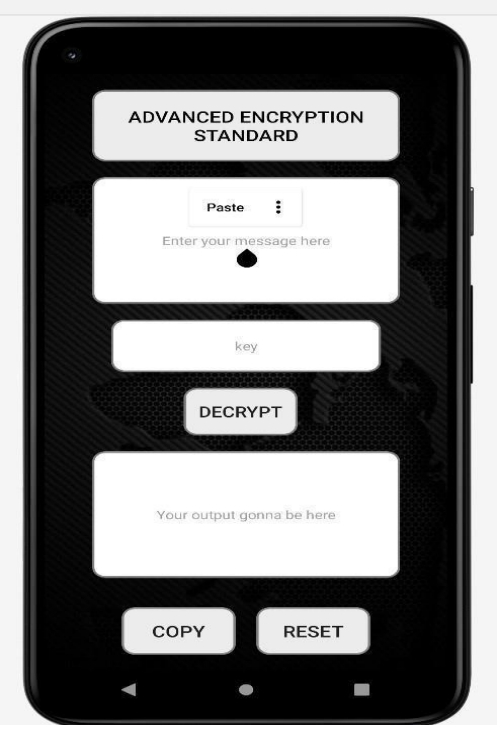

Fig 3.6 Pasting text.

Fig 3.6 is the page which shows the pasting of the text for decryption

Fig 3.7 Encryption text key.

Fig 3.7 is the page which shows entering of the key for decryption.

Fig 3.8 Decrypted message.

Fig 3.8 is the page which shows the decrypted message

Fig 3.9 Wrong key.

Fig 3.9 is the page which shows toast message that the entered key is wrong.

Fig 3.10 Before reset.

Fig 3.10 is the page that shows all the text written in the textbox before resetting.

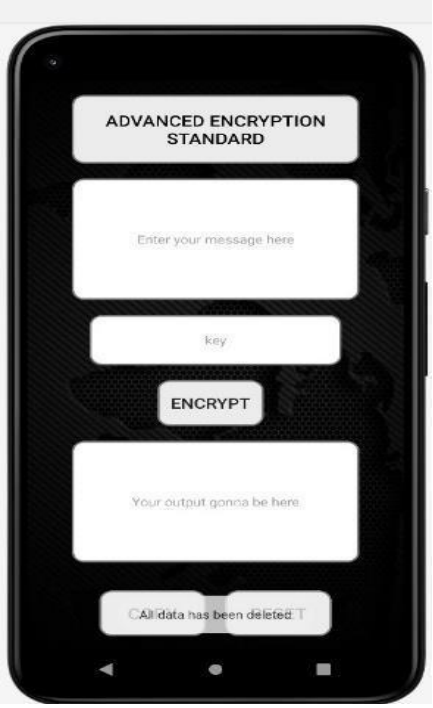

Fig 3.11 After reset.

Fig 3.11 is the page that shows all the empty textboxes after resetting the text present.

Chapter 4

CONCLUSIONS AND FUTURESCOPE

4.1 Conclusions

- The whole system has been evaluated with simple data. It is absolutely a menu driven system. The present system is free from all sorts' problem and drawbacks of existing system. This also feasible. This software was developed with grater user friendliness, because entire program are menu driven. So a new user can use it easily. The efficiency of the system can be improved by applying some more modification. It would be necessary to make few corrections in the programs has on the changes in the system and users advanced need. This soft provides facility for future needs. Password have to be shared which can be hacked and used. Only small length of text can be sent like hardly 2-3 lines. Requires active internet connection. This system can be used by everyone who wants to send some confidential text via social media.

4.2 Limitations of the System

This project has an assumption that is both the sender and receiver must have shared some secret information before imprisonment. Pure cryptography means that there is none prior information shared by two communication parties. The problem encountered here is searching information about computer security through Data Encryption and Key Algorithm and another problem is since the secret key has to be sending to the receiver of the encrypted data, it is hard to securely pass the key over the network to the receiver.

4.3 Future scope

The project “Advanced Encryption system” is designed for many future additions so that any user requirements can be made easy. Though the system is working on various assumptions it can be modified easily to a kind of requirements. Future enhancements are possible even in specific modules as entire systems are computerized and modifiable approach. The system is flexible enough to incorporate new database to existing one. Since the entire system is developed in a modular approach, modification if necessary can be done on specific module without distributing the system.

References

[1] Google Developer Training, "Android Developer Fundamentals Course– Concept Reference”, Google Developer Training Team, 2017.

[2] Erik Hellman, “Android Programming– Pushing the Limits”, 1st Edition, Wiley India Pvt Ltd, 2014.

[3] stackoverflow.com

[4] geeksforgeeks.com

[5] https://en.wikipedia.org/wiki/Playfair_cipher