VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Belagavi – 590018

A MOBILE APPLICATION DEVELOPMENT MINI PROJECT REPORT

On

Android Text Encryption

Submitted in the partial fulfilment for the requirements for the award of the Degree

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

Submitted By

Gagan R	**1ST20CS039**
Hari Prasad BP	**1ST20CS043**

Under the Guidance of:

Prof. Anuradha U
Assistant Professor

Department of CSE,
SaIT

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SAMBHRAM INSTITUTE OF TECHNOLOGY

M. S. Palya, Bengaluru – 560097

2022-2023

SAMBHRAM INSTITUTE OF TECHNOLOGY

M. S. Palya, Bengaluru – 560097

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the mini project work entitled **"Android Text Encryption"** carried out by **Gagan R** **(1ST20CS039) &** **Hari Prasad BP** **(1ST20CS043),** are bonafide students of **SAMBHRAM INSTITUTE OF TECHNOLOGY** in partial fulfilment for the award of **BACHELOR OF ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING** of **VISVESVARAYA TECHNOLOGICAL UNIVERSITY**, Belagavi during the year **2022-2023**. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report deposited. The Project report has been approved as it satisfiesthe academic requirements in respect of mini project work prescribed for the said degree.

Guide
Prof. Anuradha U
Assistant Professor
Dept. of CSE,
SaIT, Bengaluru

HOD
Dr. T John Peter
Dept. of CSE,
SaIT, Bengaluru

Signature of Examiner

1. ____________________________________

2. ____________________________________

ABSTRACT

Text Encryption deals with the development and evaluation of a mobile application for text encryption. The objective was to design and implement a user-friendly mobile application that allows users to encrypt their text messages for enhanced security. The experiment involved creating an Android application using Java programming language and integrating various encryption algorithms, including AES and RSA. The application provided options for users to choose the desired encryption algorithm, enter their text message, and encrypt it with a unique key or passphrase. The encrypted message could then be sent securely to the intended recipient. The evaluation involved testing the application's functionality, usability, and security features.The results indicated that the mobile application successfully encrypted text messages using the selected encryption algorithms, and the user interface was intuitive and easy to navigate. The report discusses the significance of text encryption in mobile Communications and highlights the potential benefits and challenges of using encryption in mobile applications. Overall,this lab experiment demonstrates the feasibility and effectiveness of incorporating text encryption into mobile applications to ensure secure communication.

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped us in carrying out this mini project work.We would like to take this opportunity to thank them all.

We would like to express our heartfelt thanks to **Dr. H. G. Chandrakanth, Principal, Sambhram Institute of Technology**, whose valuable guidance has been the one that helped usto complete the project.

We would like to express our profound gratitude to **Dr. T John Peter, HOD, Department of CSE, Sambhram Institute of Technology**, for his suggestions and his instructions have servedas the major contribution towards the completion of the mini project.

We would like to extend our impassioned thanks and admiration to our **guide, Prof. Anuradha U , Assistant Professor, Department of CSE, Sambhram Institute of Technology**, for her able guidance, regular source of encouragement and assistance throughout this mini project.

We would like to thank all the teaching and non-teaching staff members of the Computer Science Department, who have helped us directly or indirectly for the successful completion of the mini project.

Finally, we would like to thank our Parents and Friends who have helped us with their valuable suggestions and guidance for the completion of our mini project.

Gagan R
(1ST20CS039)
Hari Prasad BP
(1ST20CS043)

TABLE OF CONTENTS

TABLE OF FIGURES

CHAPTER 1

INTRODUCTION

In today's digital age, the need for secure communication and data protection has become paramount. Text encryption plays a crucial role in safeguarding sensitive information, ensuring confidentiality, and preventing unauthorized access. With the widespread use of mobile devices, the development of mobile applications that provide text encryption functionalities has become increasingly important.

The objective of this lab experiment is to design and implement a mobile application that allows users to encrypt their text messages for enhanced security. The mobile application will provide a user-friendly interface, enabling users to easily encrypt their messages before sending them. By incorporating encryption algorithms such as Advanced Encryption Standard (AES) and Rivest-Shamir-Adleman (RSA), the application aims to provide strong encryption capabilities to protect user communications.

The lab experiment will evaluate the functionality, usability, and security features of the developed mobile application. Functionality testing will ensure that the encryption process works correctly, and the encrypted messages can be decrypted using the appropriate keys or passphrases. Usability testing will assess the ease of use and intuitiveness of the application's interface. Security testing will focus on evaluating the strength of the encryption algorithms and assessing the application's resistance against potential attacks.

By developing and evaluating a text encryption mobile application, this lab experiment aims to highlight the significance of encryption in mobile communications. It emphasizes the need for secure communication channels and demonstrates the practical implementation of encryption algorithms in mobile applications. The results of this experiment will provide insights into the feasibility and effectiveness of text encryption mobile applications, paving the way for further research and development in the field of mobile security and data protection.

CHAPTER 2

OVERVIEW OF THE PROJECT

The aim of this project is to design, develop, and evaluate a text encryption mobile application. The mobile application will provide users with the ability to encrypt their text messages for enhanced security and privacy. The project will involve several key phases, including requirements gathering, design and development, testing, and evaluation.

- Requirements Gathering:

  In this phase, the project team will gather requirements by understanding the needs and expectations of potential users. This will involve conducting surveys, interviews, or user studies to identify the desired features and functionalities of the text encryption mobile application. The requirements will include factors such as encryption algorithms, user interface design, key or passphrase management, and compatibility with different mobile platforms.

- Design and Development:

  Based on the gathered requirements, the project team will proceed with the design and development phase. This phase will involve designing an intuitive and user-friendly interface for the mobile application. The design should focus on providing a seamless user experience, allowing users to enter their text messages, select the encryption algorithm, and input or generate encryption keys or passphrases. The application will be developed using appropriate mobile app development frameworks and programming languages.

- Encryption Algorithm Integration:

  The core functionality of the text encryption mobile application lies in its ability to incorporate encryption algorithms. Common encryption algorithms like Advanced Encryption Standard (AES) and Rivest-Shamir-Adleman (RSA) will be integrated into the application. This integration will ensure that the user's text messages are encrypted using strong and reliable cryptographic techniques.

- Testing:

The developed mobile application will undergo rigorous testing to ensure its functionality, usability, and security. Functionality testing will focus on verifying that the encryption process works correctly, and encrypted messages can be decrypted using the appropriate keys or passphrases. Usability testing will evaluate the ease of use and intuitiveness of the application's interface. Security testing will assess the strength of the encryption algorithms and evaluate the application's resilience against potential attacks.

- Evaluation:

Once the testing phase is completed, the project team will evaluate the overall performance of the text encryption mobile application. This evaluation will involve gathering feedback from users and analyzing their experience with the application. The feedback will be used to identify any potential areas for improvement and further enhance the application's functionality and usability.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 Functional Requirements:

Text Encryption:

- User should be able to enter text messages that they want to encrypt.
- The applications should provide options to select the desired encryption algorithm.
- User should be able to choose the encryption key for encrypting their messages.
- The application should encrypt the entered text message using selected algorithm and key.

Text Decryption:

- Users should be able to enter or retrieve the encrypted text message they want to decrypt.
- The application should determine encryption algorithm used for the encrypted message.
- User should be able to provide correct decryption key for decrypting the message.
- The application should decrypt the encrypted text message and display the original plaintext.

User Interface and Experience:

- The application should have an intuitive and user friendly interface.
- Users should be able to navigate easily through the application's screen.
- Clear instructions or prompts must be provided to guide users through the encryption and decryption processes.
- Proper error handling and validation should be implemented to ensure data integrity and prevent unauthorized access.
- Accessibility: The app should be designed to be accessible to users with disabilities, incorporating features such as screen reader compatibility, adjustable font sizes, and color contrast options.

Non-Functional Requirements:

Compatibility and Platform Support:

- The application should be compatible with Android devices, supporting various screen sizes and resolutions.
- It should be designed and developed following best practices for Android App Development.
- The application should be tested for compatibility with different versions of the Android Operating Systems.

Security Measures:

The application should implement secure storage mechanisms to protect user data and encryption keys or passphrases.

Appropriate encryption algorithms and cryptographic techniques should be used to ensure the security and integrity of the encrypted message.

The application should implement secure communication protocols for transmitting data over

3.2 Details of the Software:

The Android 11 SDK includes changes that are not compatible with some older versions of Android Studio. For the best development experience with the Android 11 SDK, use Android Studio 4.2 or higher.

Software and Hardware Requirements

Software Requirements:

- Android Studio (latest version) for development.
- Java or Kotlin programming language.
- Android SDK (Software Development Kit) for Android platform-specific libraries and tools.

Hardware Requirements:

- Computer system with sufficient processing power and memory to run Android Studio.
- Android device or emulator for testing the application.
- Internet connectivity for downloading dependencies and testing remote functionality (if applicable).

CHAPTER 4

IMPLEMENTATION

4.1 XML:

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/ConstraintLayout"
    android:background="#FFFFFF"
    tools:context="Main.">
    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scaleType="centerCrop"
        android:src="@drawable/grjml"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

```
<Button
   android:id="@+id/Swtich"
   android:layout_width="300dp"
   android:layout_height="90dp"
   android:layout_marginTop="30dp"
   android:onClick="HashButtonClick"
   android:text="MD5"
   android:textColor="#000000"
   android:background="@drawable/buttonshape"
   android:textSize="20sp"
   app:layout_constraintEnd_toEndOf="parent"
   app:layout_constraintHorizontal_bias="0.5"
   app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toTopOf="parent" />
<EditText
   android:id="@+id/TextArea"
   android:layout_width="300dp"
   android:layout_height="160dp"
   android:layout_marginTop="20dp"
   android:background="@drawable/shape"
   android:gravity="center"
   android:hint="@string/enter_your_message_here"
   android:inputType="textMultiLine"
   android:textColor="#000000"
   android:textSize="15sp"
```

```
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.504"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Swtich" />
<EditText
    android:id="@+id/salt"
    android:layout_width="208dp"
    android:layout_height="63dp"
    android:layout_marginTop="8dp"
    android:background="@drawable/shape"
    android:gravity="center"
    android:hint="Salt"
    android:maxLength="10"
    android:paddingTop="5dp"
    android:paddingBottom="5dp"
    android:textColor="#000000"
    android:textSize="15sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.497"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/TextArea" />
<TextView
    android:id="@+id/Answer"
    android:layout_width="300dp"
    android:layout_height="160dp"
```

```
  android:layout_marginTop="12dp"

  android:background="@drawable/shape"

  android:gravity="center"

  android:hint="@string/your_output_gonna_be_here"

  android:inputType="textMultiLine"

  android:textColor="#000000"

  android:textSize="15sp"

  app:layout_constraintEnd_toEndOf="parent"

  app:layout_constraintHorizontal_bias="0.495"

  app:layout_constraintStart_toStartOf="parent"

  app:layout_constraintTop_toBottomOf="@+id/hash_Buuton"

  tools:ignore="TextViewEdits" />

<Button

  android:id="@+id/hash_Buuton"

  android:layout_width="111dp"

  android:layout_height="60dp"

  android:layout_marginTop="20dp"

  android:background="@drawable/buttonshape"

  android:onClick="HashButtonClick"

  android:text="@string/hash"

  android:textSize="19sp"

  app:layout_constraintEnd_toEndOf="parent"

  app:layout_constraintStart_toStartOf="parent"

  app:layout_constraintTop_toBottomOf="@+id/salt" />
```

```
<Button
    android:id="@+id/copy_button"
    android:layout_width="111dp"
    android:layout_height="60dp"
    android:background="@drawable/buttonshape"
    android:onClick="encryptionButtonClick"
    android:text="@string/copy"
    android:textSize="20sp"
    app:layout_constraintBottom_toTopOf="@+id/Matrix"
    app:layout_constraintEnd_toStartOf="@+id/reset_button"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Answer" />
<Button
    android:id="@+id/reset_button"
    android:layout_width="111dp"
    android:layout_height="60dp"
    android:background="@drawable/buttonshape"
    android:onClick="encryptionButtonClick"
    android:text="@string/reset"
    android:textSize="20sp"
    app:layout_constraintBottom_toTopOf="@+id/Matrix"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toEndOf="@+id/copy_button"
    app:layout_constraintTop_toBottomOf="@+id/Answer" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

4.2 JAVA:

MainActivity.java

```
package Main;
import android.os.Bundle;
import android.view.View;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentTransaction;
import com.example.Algorithms.R;
import Encryption.EncryptionMain;
import Hash.HashMain;
public class MainActivity extends AppCompatActivity {
EncryptionMain encryptionMain;
HashMain hashMain;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
Fragment fragment = new MainFragment();
FragmentManager fragmentManager = getSupportFragmentManager();
fragmentManager.beginTransaction().replace(R.id.container, fragment).commit();
```

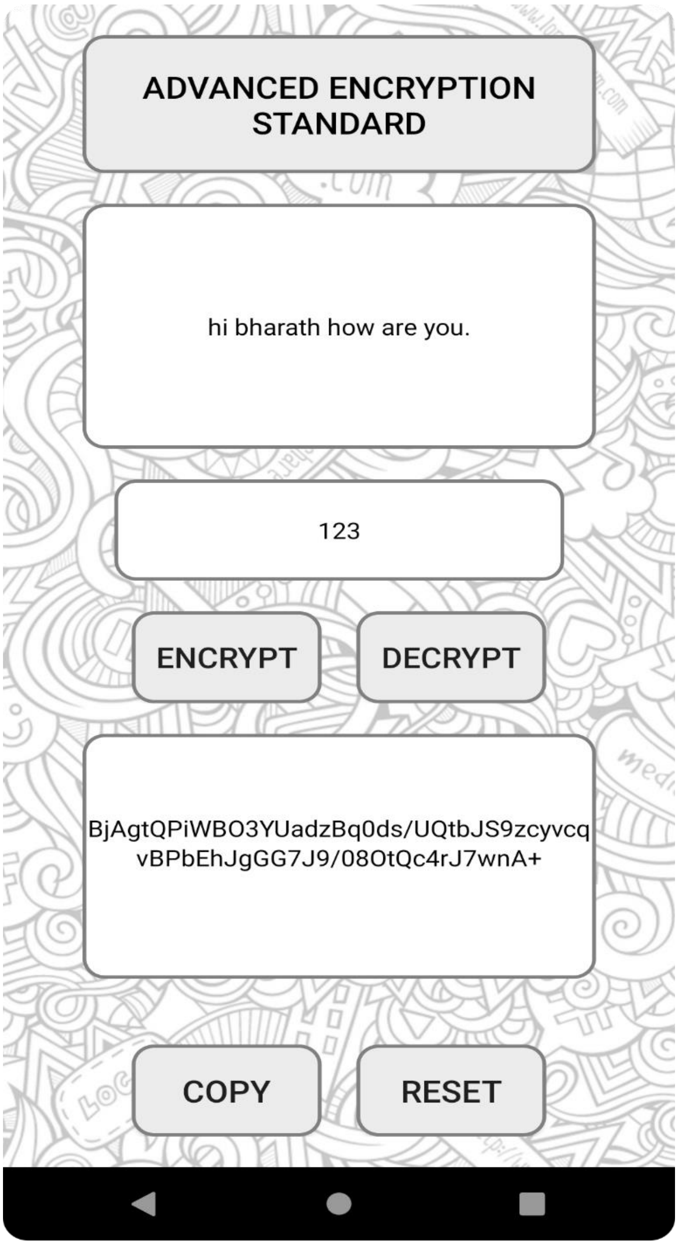
```
   }
public void goToEncryption(View view)
{
encryptionMain = new EncryptionMain();
FragmentManager manager = getSupportFragmentManager();
FragmentTransaction transaction = manager.beginTransaction();
transaction.setCustomAnimations(android.R.anim.fade_in,android.R.anim.fade_out,
android.R.anim.fade_in, android.R.anim.fade_out);
transaction.replace(R.id.container, encryptionMain);
transaction.addToBackStack(null);
transaction.commit();
   }
public void goToHash(View view) {
hashMain = new HashMain();
FragmentManager manager = getSupportFragmentManager();
FragmentTransaction transaction = manager.beginTransaction();
transaction.setCustomAnimations(android.R.anim.fade_in,android.R.anim.fade_out,
android.R.anim.fade_in, android.R.anim.fade_out);
transaction.replace(R.id.container, hashMain);
transaction.addToBackStack(null);
transaction.commit();
   }
public void encryptionButtonClick(View view) {
try {
switch (view.getId()) {
case R.id.Swtich:
```

```
encryptionMain.switchAlgho(view);
break;
case R.id.Encrypt_Buuton:
encryptionMain.encrypt(view);
break;
case R.id.Decrypt_Buuton:
encryptionMain.decrypt(view);
break;
case R.id.copy_button:
encryptionMain.copyToClipboard(view);
break;
case R.id.reset_button:
encryptionMain.reset(view);
break;
        }
     }
catch (Exception e){
e.printStackTrace();
   }
   }
 public void HashButtonClick(View view) {
try {
switch (view.getId()) {
case R.id.Swtich:
hashMain.switchAlgho(view);
```

```
 break;

 case R.id.hash_Buuton:

hashMain.hash(view);

 break;

 case R.id.copy_button:

 hashMain.copyToClipboard(view);

 break;

 case R.id.reset_button:

 hashMain.reset(view);

 break;

 }

 }

 catch (Exception e){

 e.printStackTrace();

      }

   }

 }
```

CHAPTER 5

SNAPSHOTS

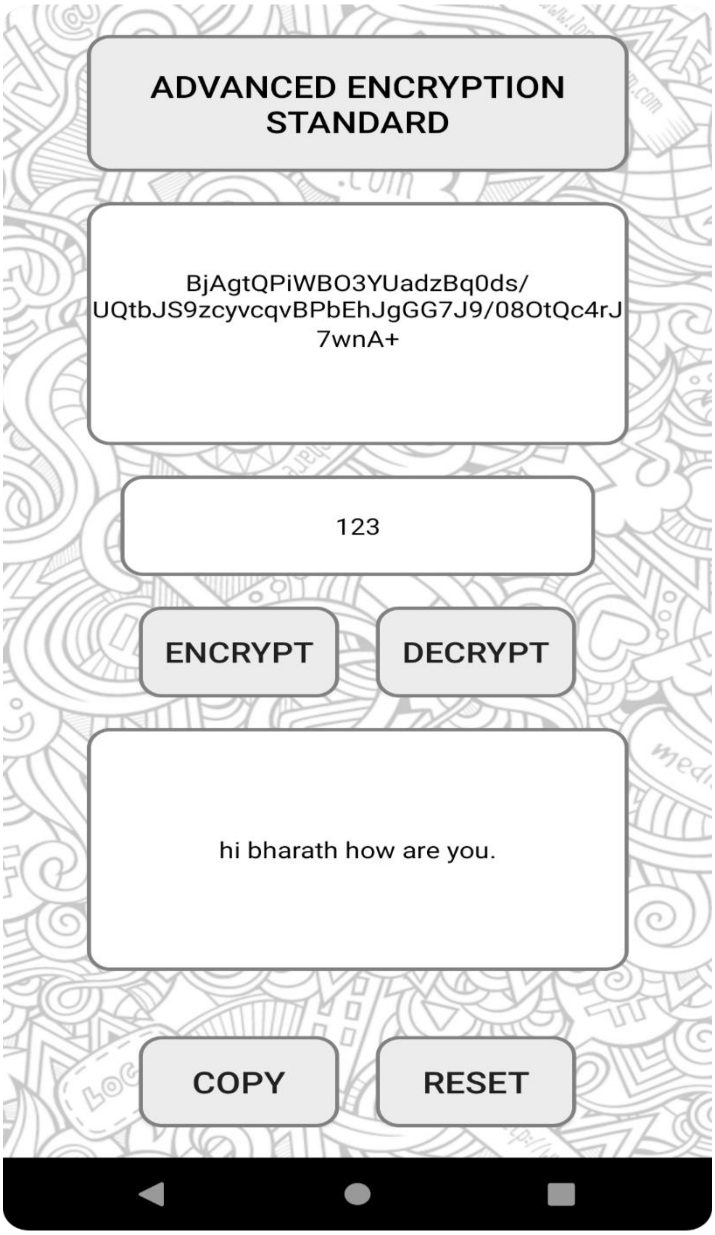


hi bharath how are you.

123

BjAgtQPiWBO3YUadzBq0ds/UQtbJS9zcyvcqvBPbEhJgGG7J9/08OtQc4rJ7wnA+

Fig 5.1 Encrypt



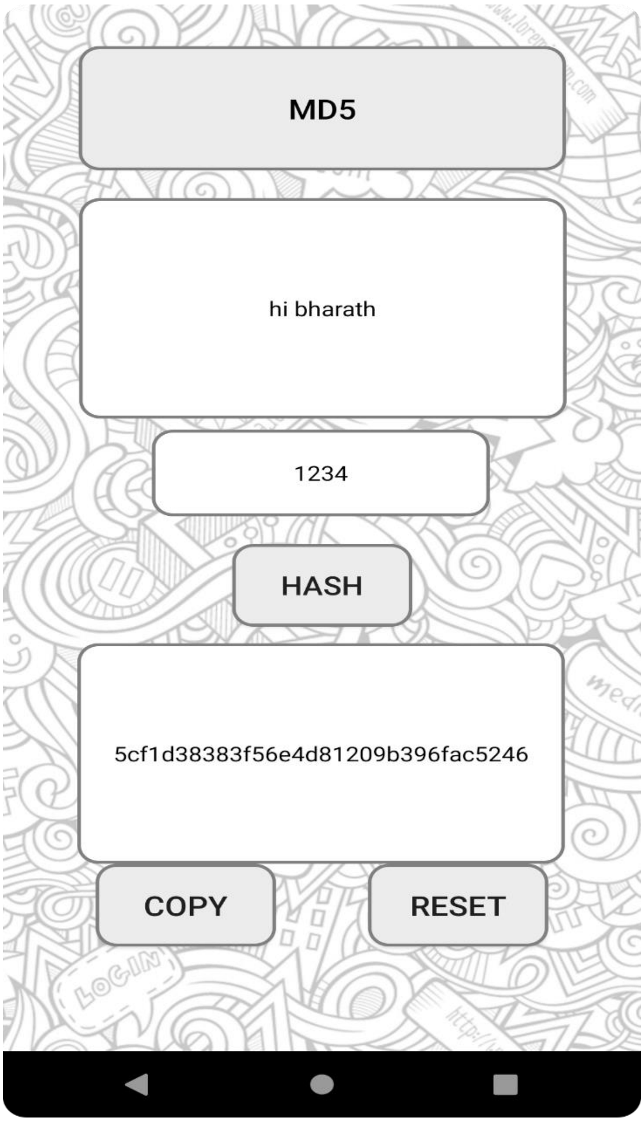
BjAgtQPiWBO3YUadzBq0ds/UQtbJS9zcyvcqvBPbEhJgGG7J9/08OtQc4rJ7wnA+

123

hi bharath how are you.

Fig 5.2 Decrypt

Fig 5.3 Hashing using MD5

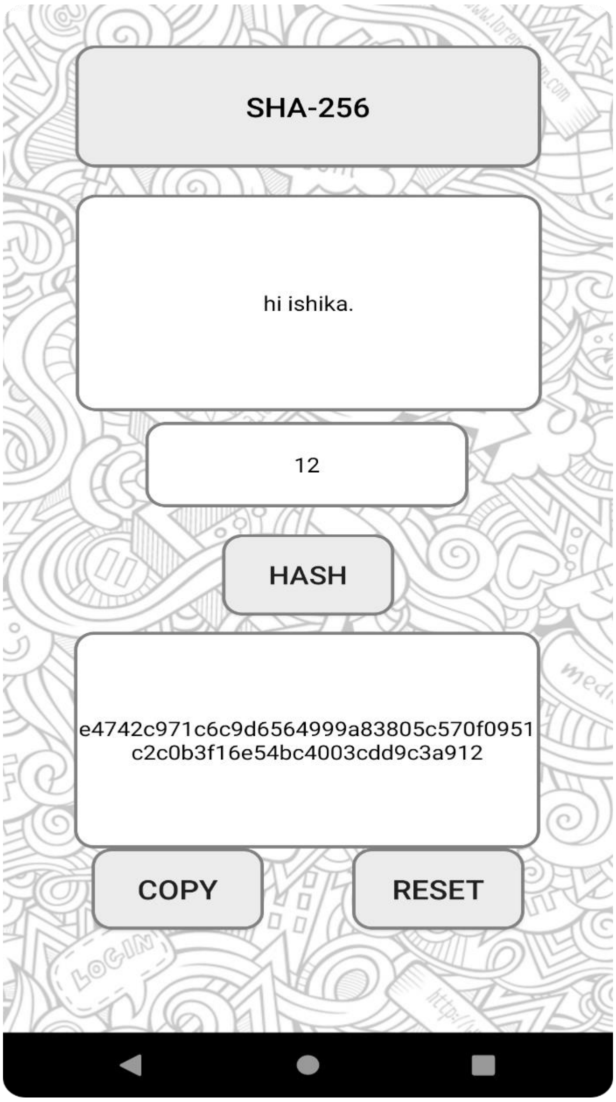


hi bharath

1234

5cf1d38383f56e4d81209b396fac5246

Fig 5.6 Hashing using SHA-256



hi ishika.

12

e4742c971c6c9d6564999a83805c570f0951
c2c0b3f16e54bc4003cdd9c3a912

CONCLUSION

In conclusion, the text encryption and decryption mini project has been successfully implemented and achieved its objectives. The project involved developing an algorithm or a set of functions that can encrypt plain text messages to make them unreadable and then decrypt them back to their original form. The encryption process used a specific key or a series of steps to transform the text, making it secure and unintelligible to unauthorized users.

During the project, various encryption techniques were explored, such as substitution ciphers, transposition ciphers, and more advanced algorithms like RSA or AES. The chosen encryption method depended on the level of security required and the resources available. Decryption was performed by reversing the encryption process using the same key or algorithm.

Future Enhancements:

User Interface: One potential enhancement could be the development of a user-friendly interface for the encryption and decryption process. This could include a graphical interface where users can input their text, select the encryption method, and easily retrieve the encrypted or decrypted result.

Multiple Encryption Algorithms: Currently, the mini project may have implemented a few encryption techniques. In the future, additional encryption algorithms could be incorporated, allowing users to choose from a wider range of options based on their specific needs. Key Management: Enhancing the key management system would be beneficial.

This could involve implementing a secure key generation mechanism, key exchange protocols, and key storage solutions. Additionally, the project could explore the use of public-key cryptography for secure communication between parties.

Cryptanalysis Techniques: Another interesting future enhancement would be to implement cryptanalysis techniques within the project. This would involve developing algorithms or methods to break or decipher encrypted messages without knowledge of the encryption key. By studying various cryptographic attacks, such as frequency analysis or brute force methods, the project could provide a better understanding of encryption vulnerabilities and their countermeasures

Integration with Other Applications: The mini project could be extended to integrate with other applications or platforms, such as messaging apps, email clients, or file encryption tools. This would provide users with seamless encryption and decryption functionalities within their existing workflows.

Performance Optimization: As the project grows and incorporates more advanced encryption algorithms, there may be a need for performance optimization. Implementing techniques such as parallel processing, algorithmic improvements, or hardware acceleration can enhance the speed and efficiency of encryption and decryption operations.

By incorporating these future enhancements, the text encryption and decryption mini project can continue to evolve and provide users with more secure and robust encryption solutions.

REFERENCES

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

[2].https://developer.android.com/docs

[3].https://stackoverflow.com/questions/tagged/android

[4]. https://github.com/android/