Dr. M.G.R.
EDUCATIONAL AND RESEARCH INSTITUTE
DEEMED TO BE UNIVERSITY
University with Graded Autonomy Status
(An ISO 21001 : 2018 Certified Institution)
Periyar E.V.R. High Road, Maduravoyal, Chennai-95. Tamilnadu, India.

PROJECT REPORT

JAVA PROGRAMMING FUNDAMENTALS (EBDS22ET2)

2024-2025(EVEN SEMESTER)

DEPARTMENT OF B.Tech DS & AI (E&T)

COURSE : B-TECH CSE-DS (AI)

YEAR/SEM/SEC : I st year/ II nd Sem/ AC1

PROJECT TITLE : FILE ENCRYPTION AND DECRYPTION

Dr. M.G.R.
EDUCATIONAL AND RESEARCH INSTITUTE
DEEMED TO BE UNIVERSITY
University with Graded Autonomy Status
(An ISO 21001 : 2018 Certified Institution)
Periyar E.V.R. High Road, Maduravoyal, Chennai-95. Tamilnadu, India.

BONAFIDE CERTIFICATE

JAVA PROGRAMMING FUNDAMENTALS

DEPARTMENT OF B.Tech DS & AI (E&T)

Certified that this project report **" File Encryption and DECRYPTION Tool"** is confirmed work of **NAME : A.R.Sarah Nausheen(AC1-18),B .DHARSHINI(AC1-03),Abinaya Sriveni(AC1-31)** I-year B-Tech CSE-DS(AI) in **JAVA PROGRAMMING FUNDAMENTALS (EBDS22ET2)** who carried out the project work under the supervision

Signature of Lab-in-Charge

Signature of Head of Dept

Submitted for the Practical Examination held on_____

Internal Examiner

External Examiner

ABSTRACT

In today's digital era, data security and confidentiality are critical. This project presents a File Encryption-Decryption Tool developed in Java using the AES (Advanced Encryption Standard) algorithm. The tool ensures that sensitive data stored in files remains secure from unauthorized access. It encrypts a plain text file into an unreadable format and allows decryption back to the original form using the same secret key. AES is a widely accepted symmetric encryption algorithm known for its speed and security. This project demonstrates secure key generation, file handling, and cryptographic transformations, offering a simple yet powerful way to safeguard files.

INTRODUCTION TO JAVA:

Java is a high-level, object-oriented programming language developed by Sun Microsystems (now owned by Oracle). It was designed with the philosophy of "Write Once, Run Anywhere", meaning compiled Java code can run on any platform with a Java Virtual Machine (JVM). Java is widely used in enterprise applications, Android development, web applications, and security-based software due to its robustness, portability, and security features.

Key features of Java include:

Platform Independence via JVM

Object-Oriented Programming (OOP)

Rich Standard Library including APIs for networking, file I/O, and security

Built-in Security Features, making it ideal for cryptography and encryption tasks

Automatic Memory Management (Garbage Collection)

In this project, Java is used to build a file-based encryption-decryption system by leveraging its powerful javax.crypto library, showcasing its capability in implementing real-world security applications.

FILE ENCRYPTION AND DECRYPTION

AIM:

To develop a Java-based tool that can encrypt and decrypt files using the AES (Advanced Encryption

Standard) algorithm, ensuring data security and confidentiality.

ALGORITHMS:

Encryption:

1. Generate or use an existing AES secret key.

2. Read the original file contents.

3. Initialize AES Cipher in ENCRYPT_MODE.

4. Encrypt the file bytes using the cipher.

5. Write the encrypted bytes to a new file.

Decryption:

1. Use the same AES secret key.

2. Read the encrypted file bytes.

3. Initialize AES Cipher in DECRYPT_MODE.

4. Decrypt the file bytes using the cipher.

5. Write the decrypted bytes to a new file.

SOURCE CODE:

```
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import java.io.*;
import java.security.SecureRandom;
import java.util.Base64;

public class AESFileEncryptDecrypt {
```

```
    // Generate AES key
    public static SecretKey generateKey() throws Exception {
        KeyGenerator keyGen = KeyGenerator.getInstance("AES");
        keyGen.init(128); // AES-128
        return keyGen.generateKey();
    }

    // Save key to file
    public static void saveKey(SecretKey key, String path) throws IOException {
        byte[] keyBytes = key.getEncoded();
        FileOutputStream fos = new FileOutputStream(path);
        fos.write(keyBytes);
        fos.close();
    }

    // Load key from file
    public static SecretKey loadKey(String path) throws IOException {
        byte[] keyBytes = new byte[16];
        FileInputStream fis = new FileInputStream(path);
        fis.read(keyBytes);
        fis.close();
        return new SecretKeySpec(keyBytes, "AES");
    }

    // Encrypt file
    public static void encryptFile(File inputFile, File outputFile, SecretKey key) throws Exception {
        Cipher cipher = Cipher.getInstance("AES");
        cipher.init(Cipher.ENCRYPT_MODE, key);

        byte[] inputBytes = readFileBytes(inputFile);
        byte[] outputBytes = cipher.doFinal(inputBytes);

        writeFileBytes(outputFile, outputBytes);
    }

    // Decrypt file
    public static void decryptFile(File inputFile, File outputFile, SecretKey key) throws Exception {
        Cipher cipher = Cipher.getInstance("AES");
        cipher.init(Cipher.DECRYPT_MODE, key);

        byte[] inputBytes = readFileBytes(inputFile);
        byte[] outputBytes = cipher.doFinal(inputBytes);

        writeFileBytes(outputFile, outputBytes);
    }

// Read file as bytes
    private static byte[] readFileBytes(File file) throws IOException {
```

```
        FileInputStream fis = new FileInputStream(file);
        byte[] bytes = fis.readAllBytes();
        fis.close();
        return bytes;
    }

    // Write byte data to file
    private static void writeFileBytes(File file, byte[] data) throws IOException {
        FileOutputStream fos = new FileOutputStream(file);
        fos.write(data);
        fos.close();
    }

    public static void main(String[] args) {
        try {
            // Generate and save AES key
            SecretKey secretKey = generateKey();
            saveKey(secretKey, "aes.key");

            // Load key
            SecretKey key = loadKey("aes.key");

            // Input & output files
            File original = new File("plain.txt");
            File encrypted = new File("encrypted.dat");
            File decrypted = new File("decrypted.txt");

            // Encrypt
            encryptFile(original, encrypted, key);
            System.out.println("File Encrypted Successfully!");

            // Decrypt
            decryptFile(encrypted, decrypted, key);
            System.out.println("File Decrypted Successfully!");

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

OUTPUT:
Suppose plain.txt contains:

This is a top secret message.

Console Output:

File Encrypted Successfully!

File Decrypted Successfully!

decrypted.txt Content:

This is a top secret message.

encrypted.dat: Contains unreadable binary data (encrypted).

RESULT:

-Successfully developed a Java program to encrypt and decrypt files using AES.

- Demonstrated secure key generation, file handling, and transformation.

- Ensures file confidentiality by converting readable data into an unreadable format and back