**FILE ENCRYPTION**

The

Project Report submitted to

Centurion University

For the partial fulfilment of the degree of

Bachelor of Technology

In

Computer Science Engineering

By

**MAMIDI VENU GOPAL**

**Regn. No 210101120102**

Under the guidance of

**MR. DEBENDRA MAHARANA**

**Head of Department, CSE**

CUTM, Paralakhemundi



Centurion University, Paralakhemundi

**Centurion University of Technology and Management**



**School of Engineering and Technology**

**CERTIFICATE**

Certified that this project report on “**FILE ENCRYPTION USING JAVA**” is the bonafide work of “**Mamidi Venu gopal** (210101120102)” of Btech CSE in **4th** semester has Successfully carried out the Project under your supervision.

**Signature of Student Signature of Faculty**

Mamidi Venu gopal Mr. Debendra Maharana

Btech CSE 4th semester Head of Department(CSE)

**ACKNOWLEDGMENT**

I **Mamidi Venu Gopal (210101120102)** from Btech computer Science Engineering of 4th semester wish to record my deep sense of gratitude and profound thanks to my supervisor **Mr. Debendra Maharana**, Associate Professor, School of Engineering and Technology of CSE department, Centurion University, Parlakhemundi for his keen interest, inspiring guidance, constant encouragement with my work during all stages, to bring this dissertation into fruition.

I thank my friends, fellow researchers and family members who have encouraged me in my research efforts and shouldered me in needy times.

**Abstract**

This project is aimed at developing a desktop-based application named File encryption is the process of converting a readable file into an unreadable format, which can only be accessed with the appropriate decryption key. This is done to protect sensitive data and prevent unauthorized access to it. File encryption can be used to protect files on a local device, in transit over a network, or stored on a remote server. File [encryption](https://www.webopedia.com/definitions/encryption/) protects individual files or file systems by encrypting them with a specific key, making them accessible only to the keyholder. The goal is to prevent malicious or unauthorized parties from accessing files that are stored on the disk. Support for file encryption can be built into an operating system or [file system](https://www.webopedia.com/definitions/file-management-system/). A decryption key allows access to the sensitive files. File encryption is helpful if a user needs to send individual files securely over the internet or store them on a removable device such as a USB stick. File encryption is an important security measure for protecting sensitive data from unauthorized access. It is widely used in industries such as healthcare, finance, and government, as well as by individuals who want to protect their personal data. There are many different encryption techniques available, each with its own level of complexity and strength. While encryption is a powerful tool for protecting sensitive data, it's important to remember that no system is completely fool proof, and there is always a risk of vulnerabilities being exploited.

**Objective of the Project**

**Primary objective**

The primary objectives of the project are mentioned below:

* To fulfil the requirement for achieving the Bachelor’s degree of Computer Science Engineering.
* To know the fundamentals of the .Net Technology and Visual Studio with the.Net Framework.

**Secondary objective**

The secondary objectives of this project are mentioned below:

* To Develop an application to protect sensitive information from unauthorized access.
* Encryption ensures the confidentiality and integrity of the data, preventing theft, tampering, or other security threats.
* It is commonly used for protecting personal information, financial data, trade secrets, and other confidential information.
* To make a secure way to share confidential data over unsecured networks, such as the internet.
* To prevent data breaches and cyber attacks, which can lead to reputational damage, financial loss, and legal consequences.
* Effective encryption requires strong encryption algorithms, secure storage of encryption keys, and proper implementation of encryption protocols.

**Scope of the Application**

The scope of file encryption is wide and varied, as it can be applied to protect sensitive data in various settings.

* Personal Data Protection
* Corporate Data Protection
* Cloud Security
* Government Data Protection

**Introduction**

The project File Encryption is a complete desktop-based application designed on .Net technology using Visual Studio Software. The main aim of the project is to

The "File Encryption" project is a Java program that allows users to encrypt and decrypt files using a simple encryption algorithm. The program uses a graphical user interface (GUI) implemented using the Swing library to provide a user-friendly interface for selecting files and entering encryption keys.

**Analysis and Design**

**I. Analysis**

Requirements Analysis

The program should provide the ability to encrypt and decrypt files.

The program should use a simple encryption algorithm.

The program should have a user-friendly graphical interface.

Use Cases and Scenarios

Use Case 1: Encrypting a File

Scenario 1: User selects a file to encrypt.

Scenario 2: User enters a PIN to use as the encryption key.

Scenario 3: Program encrypts the file using the addition algorithm.

Scenario 4: Program displays a message indicating that the file has been encrypted.

Use Case 2: Decrypting a File

Scenario 1: User selects an encrypted file to decrypt.

Scenario 2: User enters the same PIN used to encrypt the file.

Scenario 3: Program decrypts the file using the subtraction algorithm.

Scenario 4: Program displays a message indicating that the file has been decrypted.

**II. Design**

Overall Architecture

The program uses a Model-View-Controller (MVC) architecture.

The program uses Swing components for the graphical interface.

The program uses input and output streams to read and write files.

The program uses the addition and subtraction encryption algorithm to encrypt and decrypt files.

The program uses a byte array to hold the file data.

The program uses JTextArea components to display messages to the user.

Algorithms

The program uses the addition and subtraction encryption algorithms to encrypt and decrypt files.

The addition algorithm takes each byte of data in the file and performs a bitwise addition operation with the encryption key to create the encrypted data. To decrypt the file, the same key is used to perform a bitwise subtraction operation on the encrypted data.

User Interface Design

The user interface includes a label to prompt the user for the encryption key.

The user interface includes a text field to enter the encryption key.

The user interface includes two buttons: one to select a file for encryption, and one to select a file for decryption.

The user interface includes a JTextArea component to display messages to the user.

**Features**

User-friendly GUI: The application has a graphical user interface (GUI) that is easy to use and intuitive. Users can easily select the file they want to encrypt or decrypt, and enter the encryption key.

File selection: The application provides a file chooser dialog that enables users to select the file they want to encrypt or decrypt. This eliminates the need for users to remember the exact path of the file they want to operate on.

Encryption/decryption: The application provides two main functions: encryption and decryption. Users can use the encryption function to encrypt their files, and use the decryption function to decrypt their encrypted files.

Encryption key: The application allows users to enter an encryption key, which is used to encrypt or decrypt the file. Users can enter any integer value as the key.

Security: The application provides a basic level of security by using a simple encryption algorithm to encrypt and decrypt files. This algorithm ensures that the file is not easily readable by unauthorized persons.

Information message: The application displays an information message after the encryption or decryption process is complete. This message informs users whether the operation was successful or not.

Note: The application includes a note in the GUI that informs users that encrypted files get multiple encryption if encrypted multiple times, and decryption acts like encryption when a non-encrypted file is decrypted.

**Implementation of code**

import javax.swing.\*;

import java.awt.\*;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

public class FileEncryption {

public static void operate(int key){

JFileChooser fileChooser=new JFileChooser();

fileChooser.showOpenDialog(null);

File file = fileChooser.getSelectedFile();

try{

FileInputStream fis=new FileInputStream(file);;

byte []data=new byte[fis.available()];

fis.read(data);

int i=0;

for(byte d:data){

data[i]=(byte)(d-key);

i++;

}

FileOutputStream fos=new FileOutputStream(file);

fos.write(data);

fos.close();

fis.close();

JOptionPane.showMessageDialog(null,"Encrypted");

}

catch(Exception e){

e.printStackTrace();

}

}

public static void de\_operate(int key){

JFileChooser fileChooser=new JFileChooser();

fileChooser.showOpenDialog(null);

File file = fileChooser.getSelectedFile();

try{

FileInputStream fis=new FileInputStream(file);;

byte []data=new byte[fis.available()];

fis.read(data);

int i=0;

for(byte d:data){

data[i]=(byte)(d+key);

i++;

}

FileOutputStream fos=new FileOutputStream(file);

fos.write(data);

fos.close();

fis.close();

JOptionPane.showMessageDialog(null,"Decrypted");

}

catch(Exception e){

e.printStackTrace();

}

}

private final static String newLine="\n";

public static void main(String[] args) {

JFrame f=new JFrame();

f.setTitle("File Encryption");w

f.setSize(350,300);

f.setLocationRelativeTo(null);

f.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

Font font=new Font("Roboto",Font.BOLD,15);

Font font2=new Font("Times new Roman",Font.ITALIC,13);

JLabel label=new JLabel("Enter your PIN here:");

label.setSize(20,10);

JTextArea jpm =new JTextArea(6,26);//"Note: Encrypted file gets multiple encryption if encrypted multiple times.\n Decryption acts like encryption when a non-encrypted file is decrypted.")

jpm.setEditable(false);

jpm.append("Note: Encrypted file gets multiple encryption\nif encrypted multiple times.\nDecryption acts like encryption \nwhen a non-encrypted file is decrypted."+newLine);

JScrollPane scrollPAne=new JScrollPane(jpm);

jpm.setFont(font2);

JTextField textField = new JTextField(10);

textField.setFont(font);

JButton button=new JButton();

JButton button2=new JButton();

button2.setText("Select file to decrypt");

button2.setFont(font);

button.setText("Select file to encrypt");

button.setFont(font);

button.setSize(30,15);

button2.setSize(30,15);

button.addActionListener(e->{

String text= textField.getText();

int temp=Integer.parseInt(text);

operate(temp);

});

button2.addActionListener(e->{

String text= textField.getText();

int temp=Integer.parseInt(text);

de\_operate(temp);

});

f.setLayout(new FlowLayout());

f.add(label);

f.add(textField);

f.add(button);

f.add(button2);

f.add(scrollPAne);

f.setVisible(true);

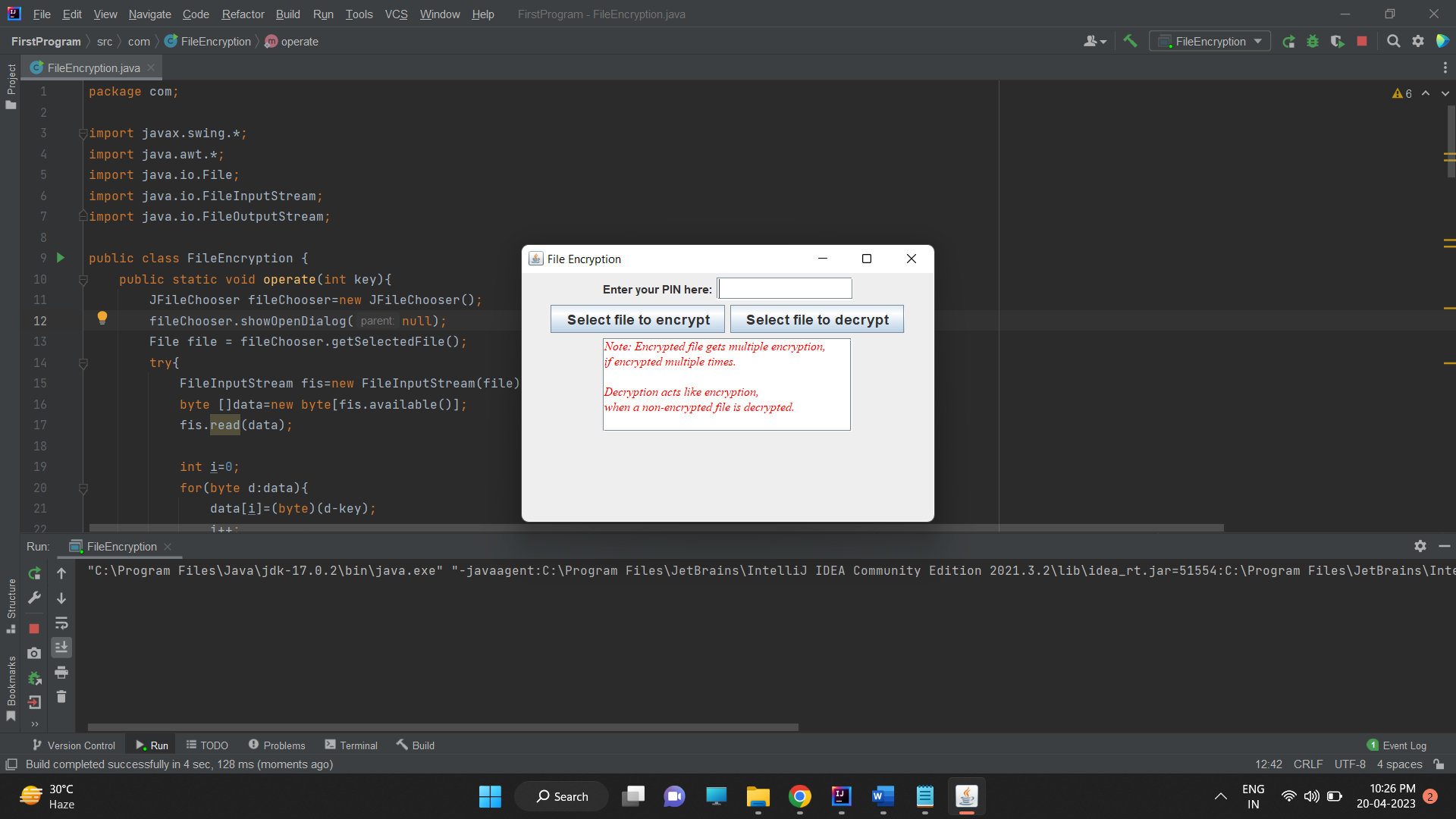
}

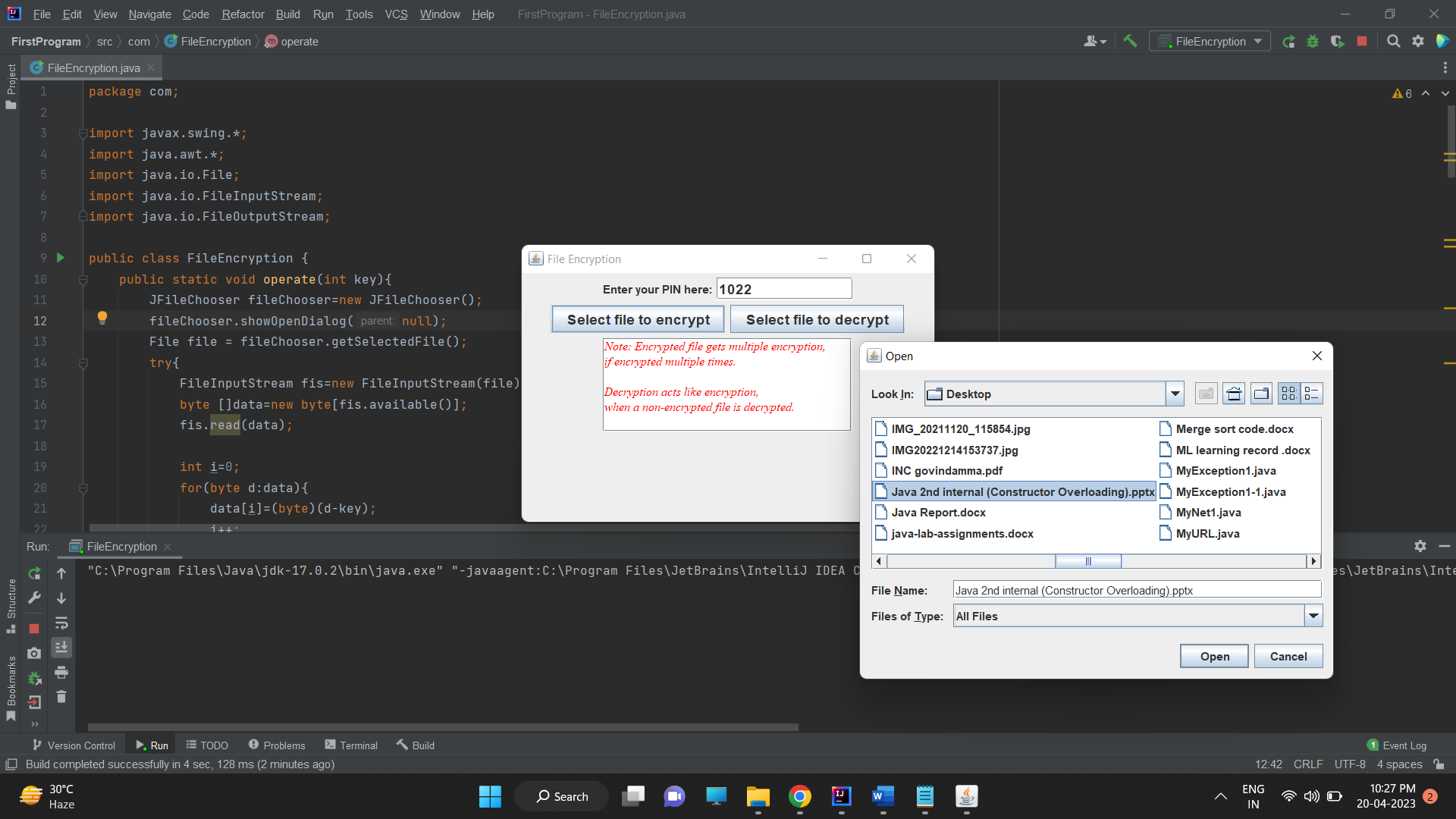
}

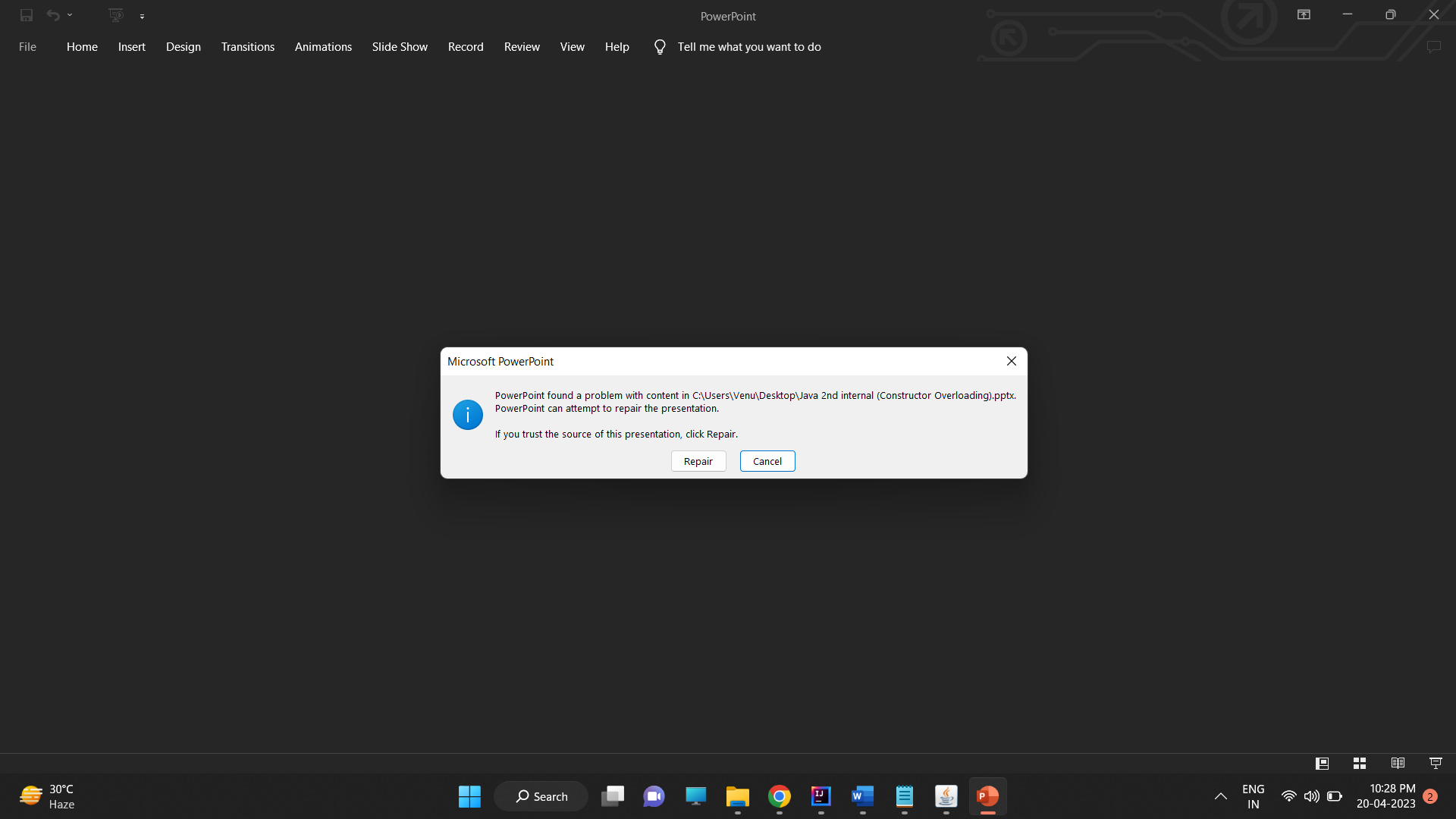
**Flow of code**

1. The program displays a GUI window with a label, a text field, two buttons, and a text area.
2. The user enters a PIN in the text field.
3. The user clicks the "Select file to encrypt" button to choose a file to encrypt. The program opens a file chooser dialog to allow the user to select a file.
4. The program reads the content of the selected file using a FileInputStream.
5. The program encrypts the file content by subtracting the PIN value from each byte.
6. The program writes the encrypted content back to the same file using a FileOutputStream.
7. The program displays a message dialog to indicate that the file has been encrypted.
8. The user clicks the "Select file to decrypt" button to choose a file to decrypt. The program opens a file chooser dialog to allow the user to select a file.
9. The program reads the content of the selected file using a FileInputStream.
10. The program decrypts the file content by adding the PIN value to each byte.
11. The program writes the decrypted content back to the same file using a FileOutputStream.
12. The program displays a message dialog to indicate that the file has been decrypted.

**Results and Conclusion:**







This project is a good introduction to file encryption and decryption, and provides a solid foundation for further exploration of encryption algorithms and techniques. However, it should be noted that the Bitwise Addition and subtraction-based encryption used in this program is not suitable for secure applications, and should not be relied upon for protecting sensitive data. More advanced encryption algorithms, such as AES or RSA, should be used instead.

**Bibliography**

Oracle. (2021). How to Use File Choosers. Retrieved from <https://docs.oracle.com/javase/tutorial/uiswing/components/filechooser.html>

Jelena Vukovic. (2019). Encryption and Decryption with Java. Retrieved from <https://www.baeldung.com/java-file-encryption-decryption>

StackOverflow: How to use the swing class for GUI. Retrieved from

<https://stackoverflow.com/questions/15363706/how-to-program-this-gui-in-java-swing>

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***