

# **Chapter 1: Introduction**

**Background, Objective, Purpose & Scope, Limitation**

## **1. Introduction:**

### **1.1 Background:**

The ***EncryptoPad*** project is developed to provide users with a simple yet effective file management application that incorporates text file operations and security features. The application allows users to create, edit, encrypt, decrypt, search, replace, and delete files securely. The growing need for data protection and easy file manipulation, this tool aims to bridge the gap in secure document handling.

### **1.2 Objective:**

The primary objective of the project is to design and implement a user-friendly file management system with robust encryption and decryption capabilities. Users should be able to manage their text files securely with options for creating, reading, editing, encrypting, decrypting, searching, replacing, and deleting files.

### **1.3 Purpose & Scope:**

The project focuses on providing a simple interface for secure text file manipulation. The scope includes developing a terminal-based application that enables:

- File creation and editing
- File encryption and decryption
- Search and replace functionality in files
- File deletion.

## 1.4 Limitation:

The primary limitation of the project is its focus on text files and the use of a simple encryption algorithm. While Caesar cipher is effective for demonstration purposes, it is not suitable for handling high-security encryption. Additionally, the application is terminal-based and may not be as user-friendly as GUI-based solutions.

# **Chapter 2: Feasibility Study**

**Economic Feasibility, Technical Feasibility, Behavioral Feasibility**

## **2.1 Economic Feasibility:**

From an economic perspective, the project is cost-effective because it relies on open-source tools and does not require expensive software or hardware infrastructure. The development and maintenance costs are minimal, making it a budget-friendly solution for individuals seeking basic file security.

## **2.2 Technical Feasibility:**

The application is technically feasible, using Python, a widely used programming language that is well-suited for handling file operations and encryption. The design of the project ensures that all features are implementable within the given scope, with no major technical obstacles anticipated during development.

## **2.3 Behavioral Feasibility:**

The project has a positive behavioral feasibility because it addresses the growing concern of file security. Users are likely to appreciate the convenience of managing their files securely with a simple, interactive interface.

# **Chapter 3: Requirement & Analysis**

Problem Definition, Planning & Scheduling , Hardware/Software Requirements

## **3.1 Problem Definition**

The problem that this application solves is the lack of simple, secure file management tools that allow users to easily create, edit, and encrypt files. Many people store sensitive information in text files, and this application provides a straightforward way to ensure the data's privacy while keeping the operations easy to use.

## **3.2 Planning & Scheduling (Flow Chart & Gantt Chart)**

The development was planned in stages, with a clear timeline established for each task. A flowchart and Gantt chart were used to visualize the project flow and assign deadlines to tasks. The stages included requirements gathering, system design, development, testing, documentation, and final review.

## **3.3 Software/Hardware Requirements**

- Software: Python 3.13.1, text editor (VS Code), terminal or command-line interface.
- Hardware: Any machine capable of running Python, basic computer or laptop.
- Additional Tools: Basic cryptographic tools like Caesar cipher (for encryption/decryption) and file manipulation libraries.

## **Chapter 4: Survey of Technology(Literature Review)**

Previous Work, Technology Used (Front End, Backend Database), IDE, API, Various Tools

### **4.1 Previous Work**

There are existing file management systems and encryption tools available in the market; however, many of these solutions are either too complex or not secure enough for personal use. This project fills the gap by offering a simplified, secure file management system without unnecessary features.

### **4.2 Technology Used (Front-End, Back-End, Database)**

- **Front-End:** Terminal-based interface.
- **Back-End:** Python programming language to handle file operations, encryption, and decryption.
- **Database:** No database is used as the application works with local files directly.

### **4.3 IDE, API, Various Tools**

- **IDE:** Visual Studio Code (VS Code) for writing and testing Python scripts.
- **APIs:** No external APIs are used; basic Python file handling functions and libraries are used for operations.
- **Tools:** Terminal for running the application.

## Chapter 5: Preliminary Module Description

---

This chapter includes an overview of the various modules within the EncryptoPad project, describing their functions and how they interact with each other:

- **File Creation Module:** Allows users to create a new text file.
- **File Reading Module:** Displays the contents of an existing file.
- **File Editing Module:** Appends new content to an existing file.
- **Encryption/Decryption Module:** Encrypts or decrypts the content of the file using a simple Caesar cipher.
- **Search and Replace Module:** Enables users to find and replace text within a file.
- **File Deletion Module:** Deletes a specified file from the system.

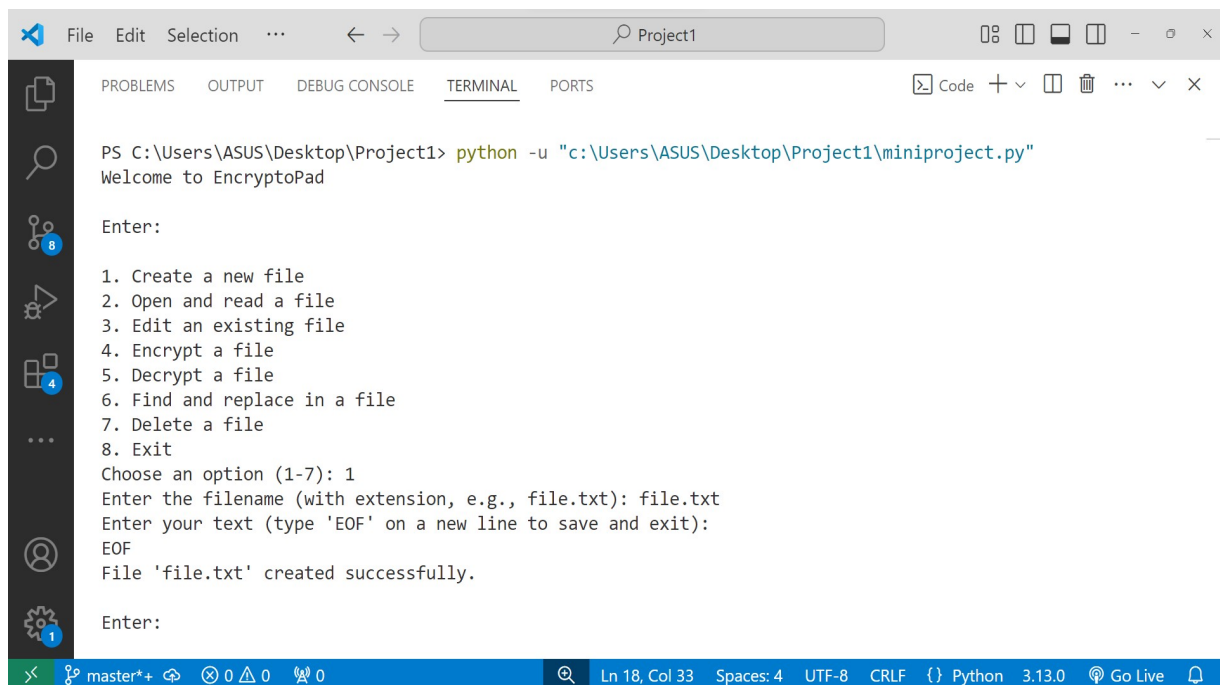
# Chapter 6: Detailed Design

Input / Output Screenshot of Project, Validation Checks

## 7.1 Input/Output Screenshot of Project

Screenshots of the input/output interface are included, demonstrating how users interact with the system through the terminal interface. These screenshots show file operations such as creating a file, reading content, encrypting text, and deleting a file.

### Input Screenshot:



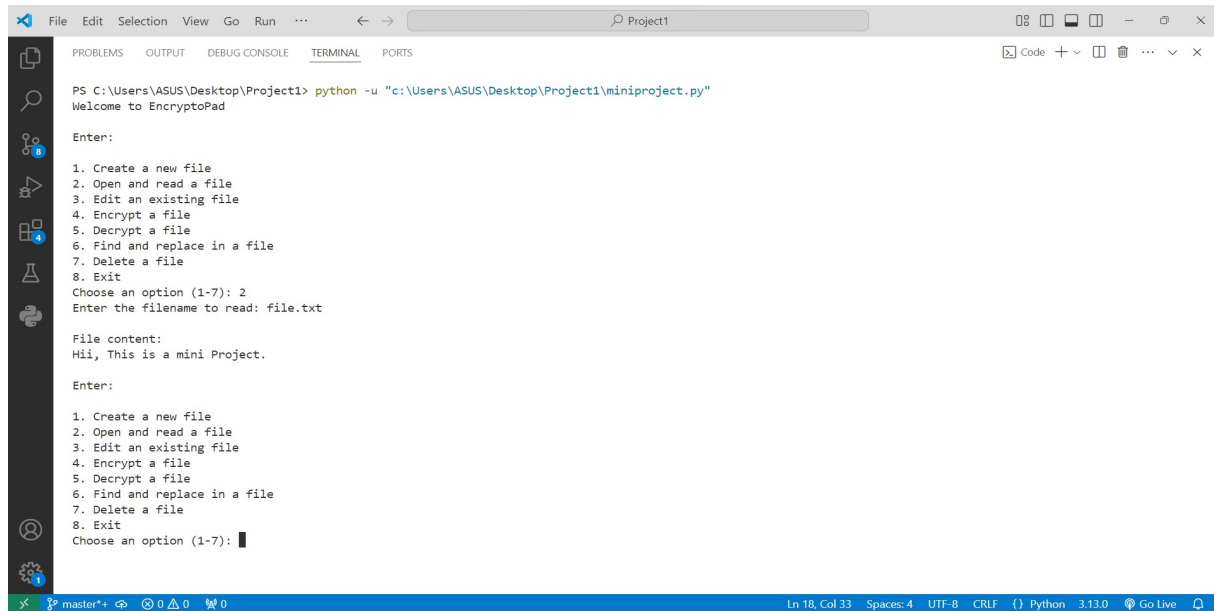
```
File Edit Selection ... < > Project1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ASUS\Desktop\Project1> python -u "c:\Users\ASUS\Desktop\Project1\miniproject.py"
Welcome to EncryptoPad

Enter:

1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 1
Enter the filename (with extension, e.g., file.txt): file.txt
Enter your text (type 'EOF' on a new line to save and exit):
EOF
File 'file.txt' created successfully.

Enter:
```

### OutPut Screenshot:



```
PS C:\Users\ASUS\Desktop\Project1> python -u "c:\Users\ASUS\Desktop\Project1\miniproject.py"
Welcome to EncryptoPad

Enter:

1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 2
Enter the filename to read: file.txt

File content:
Hi, This is a mini Project.

Enter:

1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): █
```

## 7.2 Validation Checks

The system validates input at each stage:

- Ensures files exist before attempting operations (reading, editing, etc.).
- Prompts the user to confirm actions like file deletion.
- Verifies that encrypted files can only be decrypted with the correct key.



## Chapter 7: Testing Techniques

---

- **Unit Testing:** For individual modules like file creation, encryption, and decryption.
- **Integration Testing:** To check that all modules work together smoothly.
- **System Testing:** To ensure the entire application functions as expected when handling all operations.

Testing was conducted manually by simulating user interactions with the system.

## **Chapter 8: Future Scope & Enhancements**

---

### **Limitations**

- The system supports only text files and lacks advanced encryption algorithms.
- The terminal interface is not as user-friendly as a graphical one.

### **Future Scope & Enhancements**

- **Future Scope:** Expanding support to handle other file formats (e.g., PDFs, images) and implementing stronger encryption algorithms (e.g., AES).
- **Enhancements:** Adding a GUI, integrating cloud storage options, and expanding the search functionality to support regular expressions.

# Chapter 9: References & Bibliography

## References, Bibliography

---

### 11.1 References

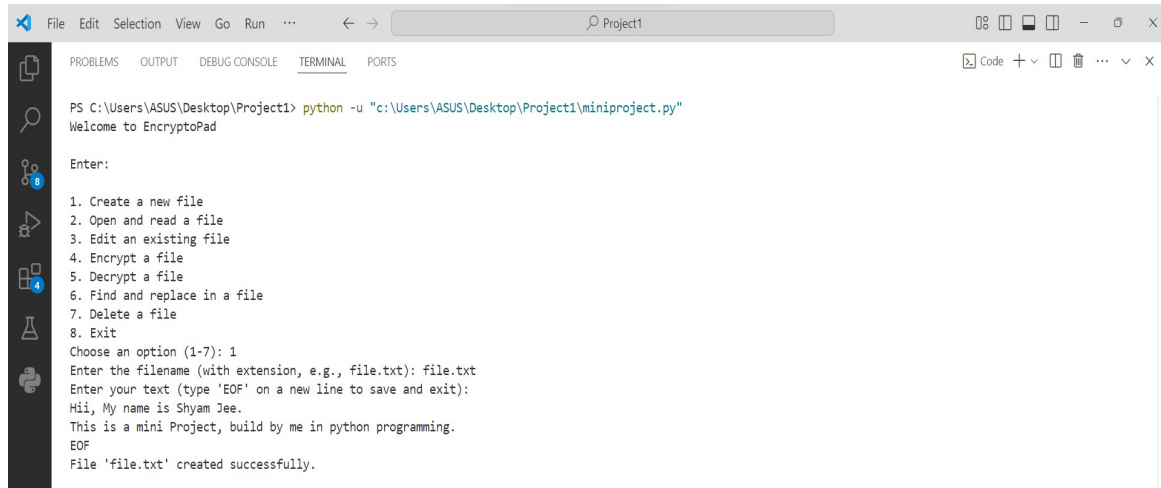
- Python documentation (<https://docs.python.org/>)
- Tutorials on Caesar cipher and file handling in Python.

### 11.2 Bibliography

1. online articles on file management systems, file security and Python programming.

# Appendix in Report

## 1. Create a new file:

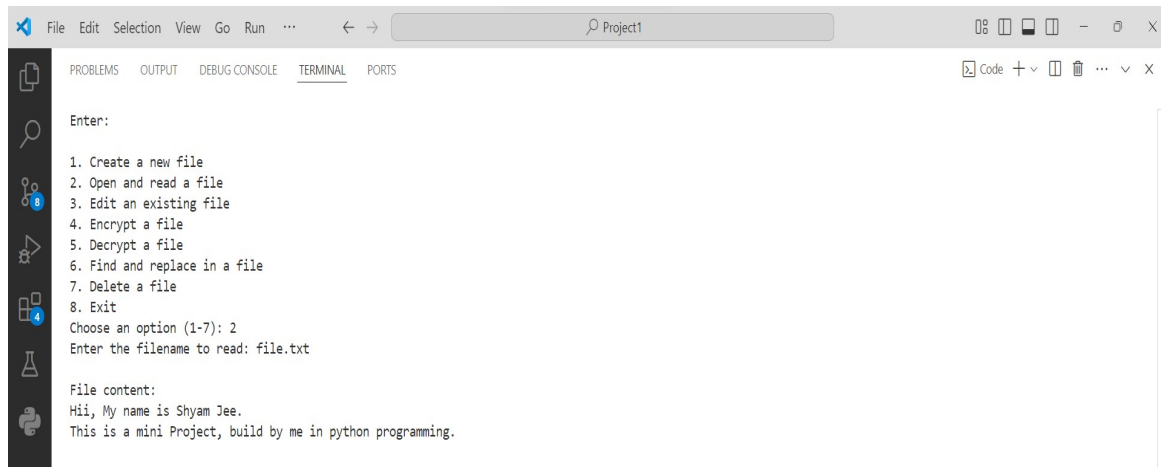


```
PS C:\Users\ASUS\Desktop\Project1> python -u "c:\Users\ASUS\Desktop\Project1\miniproject.py"
Welcome to EncryptoPad

Enter:

1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 1
Enter the filename (with extension, e.g., file.txt): file.txt
Enter your text (type 'EOF' on a new line to save and exit):
Hii, My name is Shyam Jee.
This is a mini Project, build by me in python programming.
EOF
File 'file.txt' created successfully.
```

## 2. Open and read a file:



```
Enter:

1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 2
Enter the filename to read: file.txt

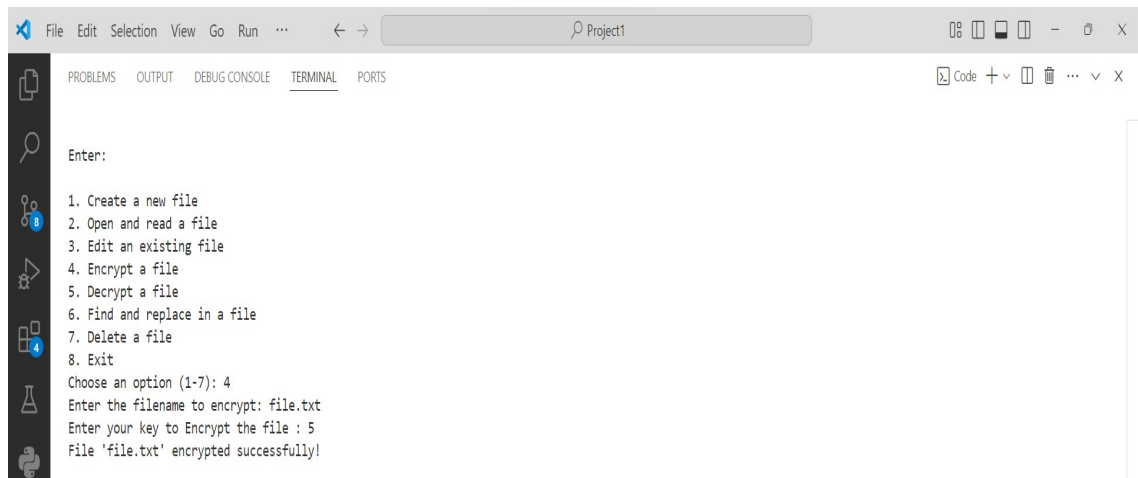
File content:
Hii, My name is Shyam Jee.
This is a mini Project, build by me in python programming.
```

## 3. Edit an existing file:



```
File Edit Selection View Go Run ... Project1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 3
Enter the filename to edit: file.txt
Enter text to append (type 'EOF' on a new line to save and exit):
Thankyou
EOF
Changes saved to 'file.txt'.
```

#### 4. Encrypt a file:



```
File Edit Selection View Go Run ... Project1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Enter:
1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 4
Enter the filename to encrypt: file.txt
Enter your key to Encrypt the file : 5
File 'file.txt' encrypted successfully!
```

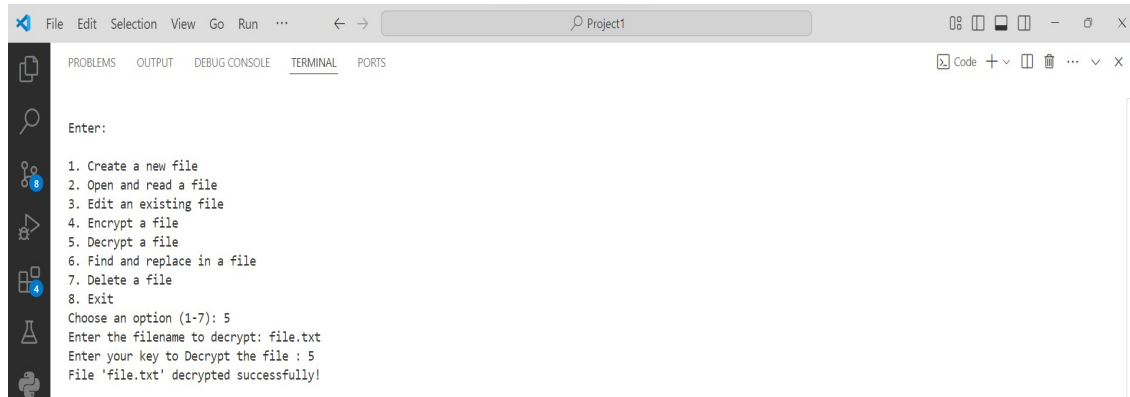
#### 5. After Encryption the content:



```
File Edit Selection View Go Run ... Project1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Enter:
1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 2
Enter the filename to read: file.txt

File content:
Mnn, Rd sfrj nx Xmdfr Ojj.
Ymnx nx f rnsn Uwtjhy, gznqi gd nj ns udymts uwtlwfrnsl.
Ymfspdtz
```

## 6. Decrypt a file:

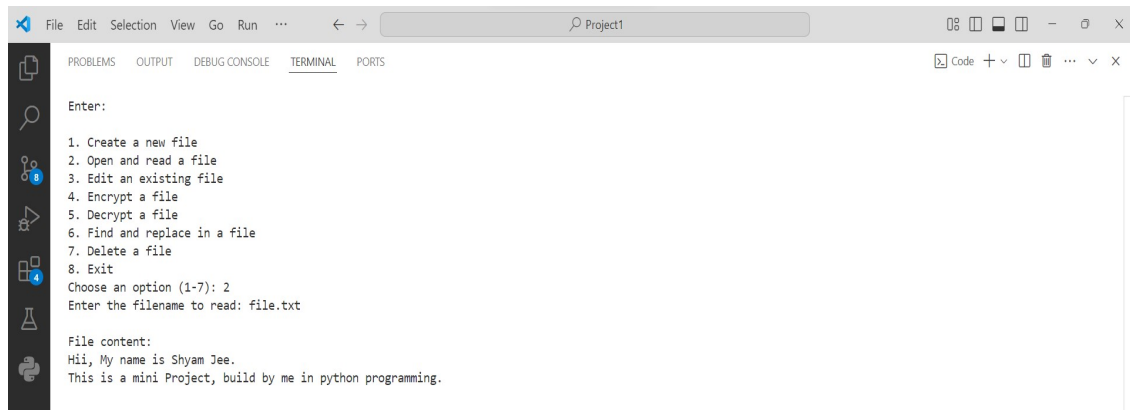


```
File Edit Selection View Go Run ... < -> Project1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - - - - X

Enter:

1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 5
Enter the filename to decrypt: file.txt
Enter your key to Decrypt the file : 5
File 'file.txt' decrypted successfully!
```

## 7. After Decryption the content of file:



```
File Edit Selection View Go Run ... < -> Project1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - - - - X

Enter:

1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 2
Enter the filename to read: file.txt

File content:
Hii, My name is Shyam Jee.
This is a mini Project, build by me in python programming.
```

## 8. Find an replace in a file:

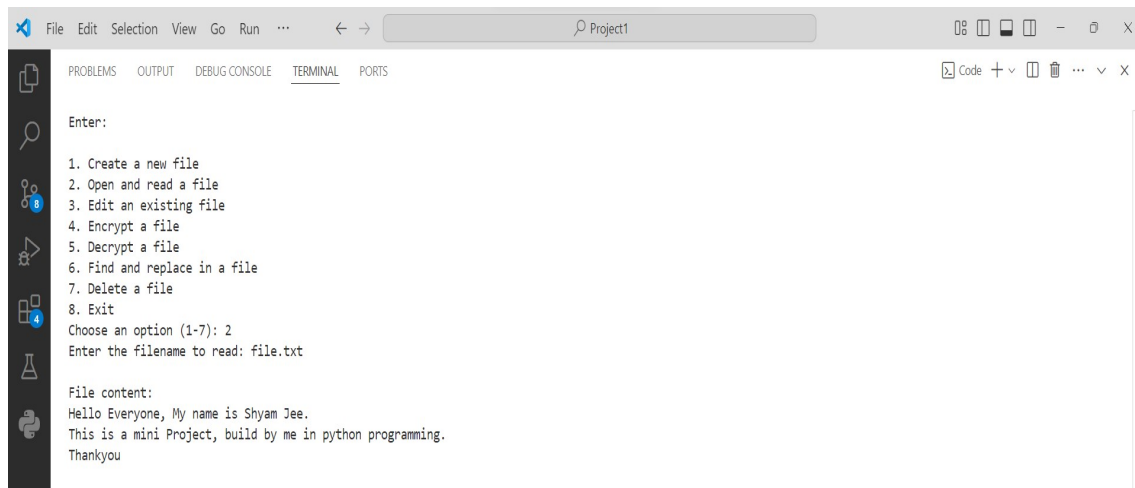


```
File Edit Selection View Go Run ... < -> Project1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - - - - X

Enter:

1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 6
Enter the filename to find and replace text in: file.txt
Enter the text to find: Hii
Enter the text to replace it with: Hello Everyone
All occurrences of 'Hii' have been replaced with 'Hello Everyone' in 'file.txt'.
```

## 9. After replacement the content of file:

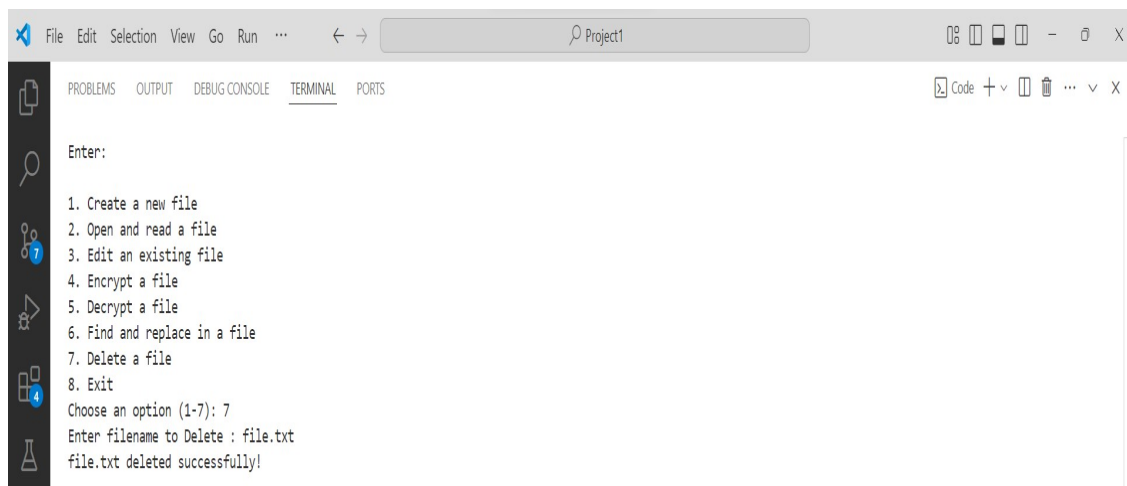


The screenshot shows the EncryptoPad application interface. The terminal window displays the following text:

```
Enter:
1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 2
Enter the filename to read: file.txt

File content:
Hello Everyone, My name is Shyam Jee.
This is a mini Project, build by me in python programming.
Thankyou
```

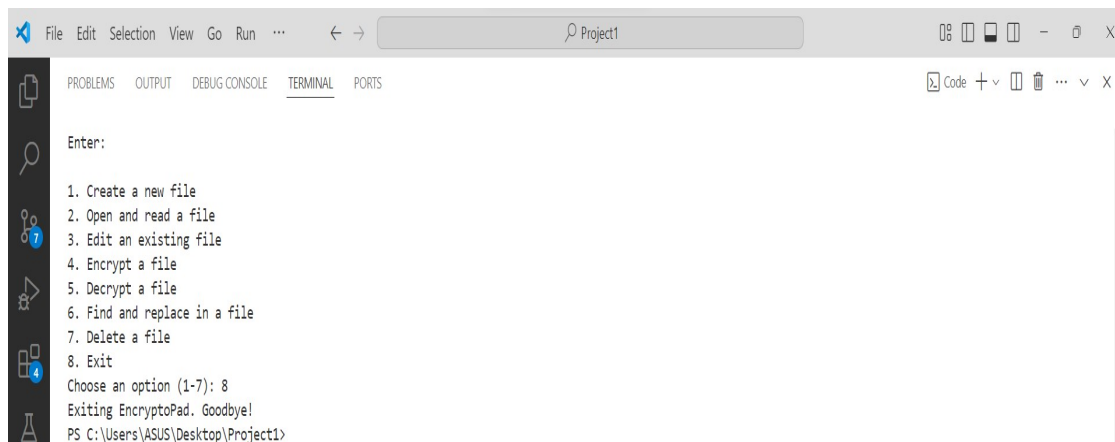
## 10.Delete a file:



The screenshot shows the EncryptoPad application interface. The terminal window displays the following text:

```
Enter:
1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 7
Enter filename to Delete : file.txt
file.txt deleted successfully!
```

## 11.Exiting from the project:



```
File Edit Selection View Go Run ... ← → Project1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Enter:
1. Create a new file
2. Open and read a file
3. Edit an existing file
4. Encrypt a file
5. Decrypt a file
6. Find and replace in a file
7. Delete a file
8. Exit
Choose an option (1-7): 8
Exiting EncryptoPad. Goodbye!
PS C:\Users\ASUS\Desktop\Project1>
```