# Secure File Sharing System - Project Report

## 1. Project Overview

This project implements a secure file sharing system that allows users to upload and download files safely. Files are encrypted using AES before storage and decrypted upon retrieval. The system also includes basic key management to handle encryption keys securely.

## 2. Features

- File upload & download portal using Python Flask / Node.js
- AES encryption for securing files at rest and in transit
- Simple and user-friendly web interface for file handling
- Secure handling of encryption keys
- Tested for file integrity and security
- Well-documented architecture and security measures

## 3. Tools & Technologies

- Backend: Python Flask or Node.js (Express)
- Cryptography: PyCryptodome / Node.js Crypto
- Version Control: Git & GitHub
- Testing: Postman / curl
- Documentation & Deployment: GitHub

## 4. Skills Gained

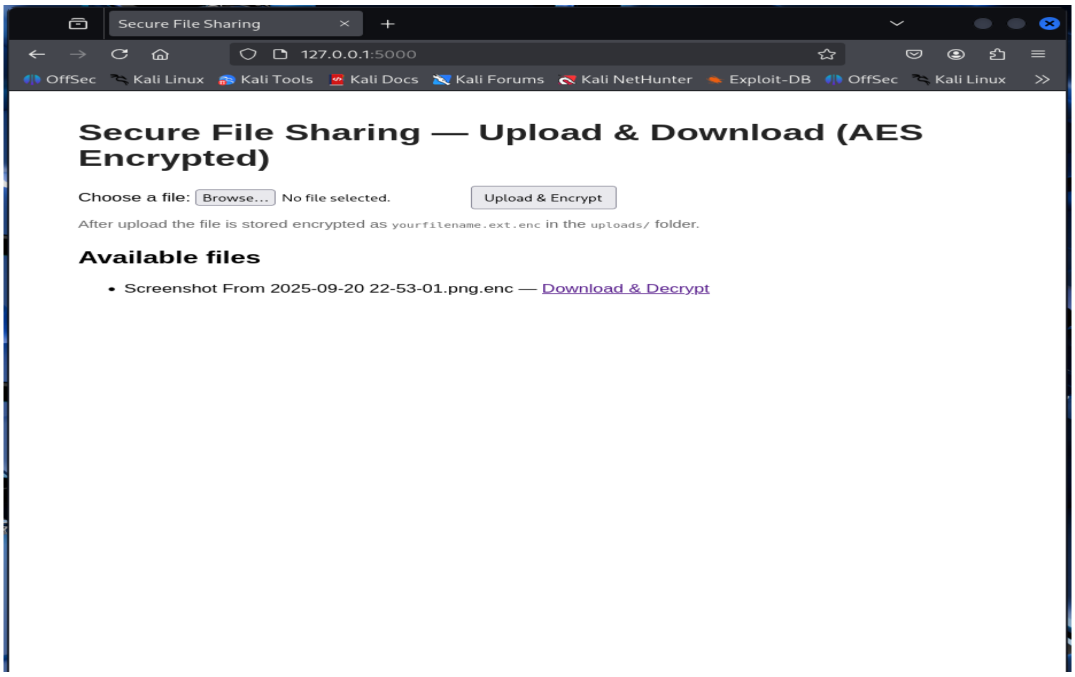
- Web development
- AES encryption implementation
- Secure file handling
- Key management
- System security testing
- Technical documentation

## 5. Deliverables

- GitHub Repository (source code + docs)
- Walkthrough video
- Security overview document

## 6. Project Screenshots

Below are screenshots from the project implementation:


Secure File Sharing

127.0.0.1:5000

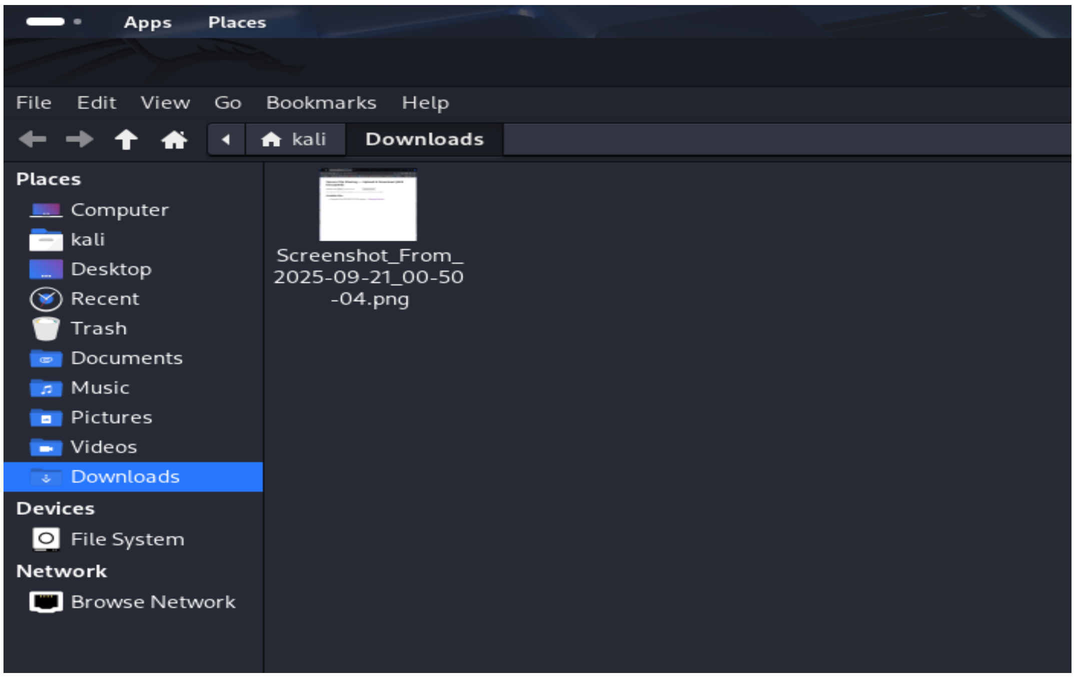
**Secure File Sharing — Upload & Download (AES Encrypted)**

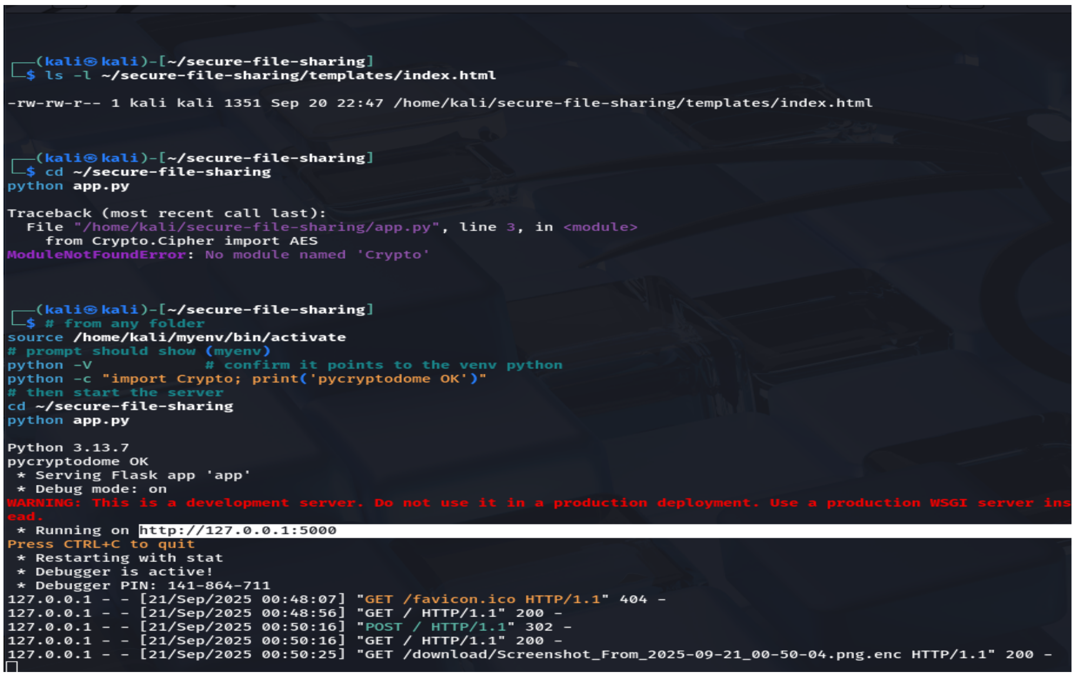
Choose a file: Browse… No file selected. Upload & Encrypt

After upload the file is stored encrypted as `yourfilename.ext.enc` in the `uploads/` folder.

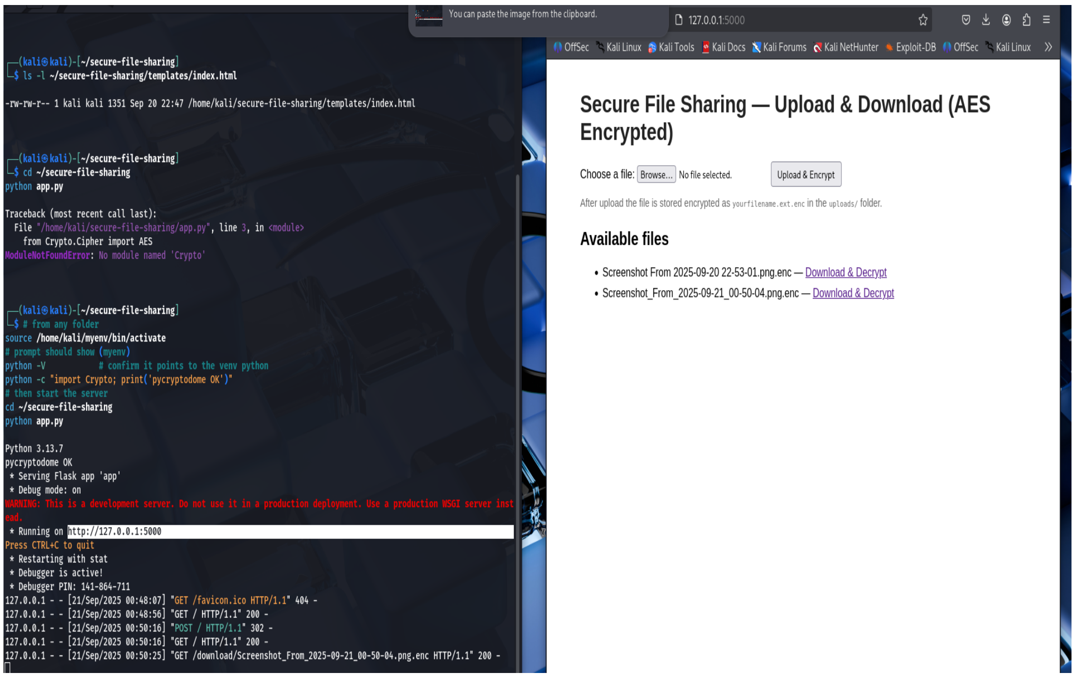
**Available files**

- Screenshot From 2025-09-20 22-53-01.png.enc — Download & Decrypt





```
┌──(kali㉿kali)-[~/secure-file-sharing]
└─$ ls -l ~/secure-file-sharing/templates/index.html

-rw-rw-r-- 1 kali kali 1351 Sep 20 22:47 /home/kali/secure-file-sharing/templates/index.html


┌──(kali㉿kali)-[~/secure-file-sharing]
└─$ cd ~/secure-file-sharing
python app.py

Traceback (most recent call last):
  File "/home/kali/secure-file-sharing/app.py", line 3, in <module>
    from Crypto.Cipher import AES
ModuleNotFoundError: No module named 'Crypto'


┌──(kali㉿kali)-[~/secure-file-sharing]
└─$ # from any folder
source /home/kali/myenv/bin/activate
# prompt should show (myenv)
python -V            # confirm it points to the venv python
python -c "import Crypto; print('pycryptodome OK')"
# then start the server
cd ~/secure-file-sharing
python app.py

Python 3.13.7
pycryptodome OK
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server ins
ead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 141-864-711
127.0.0.1 - - [21/Sep/2025 00:48:07] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [21/Sep/2025 00:48:56] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [21/Sep/2025 00:50:16] "POST / HTTP/1.1" 302 -
127.0.0.1 - - [21/Sep/2025 00:50:16] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [21/Sep/2025 00:50:25] "GET /download/Screenshot_From_2025-09-21_00-50-04.png.enc HTTP/1.1" 200 -
```



```
┌──(kali㉿kali)-[~/secure-file-sharing]
└─$ ls -l ~/secure-file-sharing/templates/index.html

-rw-rw-r-- 1 kali kali 1351 Sep 20 22:47 /home/kali/secure-file-sharing/templates/index.html


┌──(kali㉿kali)-[~/secure-file-sharing]
└─$ cd ~/secure-file-sharing
python app.py

Traceback (most recent call last):
  File "/home/kali/secure-file-sharing/app.py", line 3, in <module>
    from Crypto.Cipher import AES
ModuleNotFoundError: No module named 'Crypto'


┌──(kali㉿kali)-[~/secure-file-sharing]
└─$ # from any folder
source /home/kali/myenv/bin/activate
# prompt should show (myenv)
python -V            # confirm it points to the venv python
python -c "import Crypto; print('pycryptodome OK')"
# then start the server
cd ~/secure-file-sharing
python app.py

Python 3.13.7
pycryptodome OK
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server inst
ead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 141-864-711
127.0.0.1 - - [21/Sep/2025 00:48:07] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [21/Sep/2025 00:48:56] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [21/Sep/2025 00:50:16] "POST / HTTP/1.1" 302 -
127.0.0.1 - - [21/Sep/2025 00:50:16] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [21/Sep/2025 00:50:25] "GET /download/Screenshot_From_2025-09-21_00-50-04.png.enc HTTP/1.1" 200 -
```

You can paste the image from the clipboard.

127.0.0.1:5000

Secure File Sharing — Upload & Download (AES Encrypted)

Choose a file: Browse... No file selected. Upload & Encrypt

After upload the file is stored encrypted as `yourfilename.ext.enc` in the `uploads/` folder.

Available files

- Screenshot From 2025-09-20 22-53-01.png.enc — Download & Decrypt
- Screenshot_From_2025-09-21_00-50-04.png.enc — Download & Decrypt

## 7. Security Measures

- AES symmetric encryption (256-bit key size)
- Secure key management (environment variables)
- File integrity validation
- Encrypted storage and secure transfer protocols

## 8. How to Run the Project

Steps to set up and run the project locally:

- Clone the repository: git clone https://github.com/your-username/secure-file-sharing.git
- Install dependencies: pip install flask pycryptodome (or npm install express crypto)
- Run the server: python app.py (or node server.js)
- Open the web portal at http://127.0.0.1:5000
- Upload files to encrypt and download to decrypt

## 9. Author

Developed by: [Your Name] Connect on LinkedIn: https://linkedin.com/in/yourprofile