Security File Sharing System — Task 3 Report

About the Task:

In this hands-on project, I built a secure file sharing system as part of my task. The aim was to simulate a real-world scenario where users can upload and download files securely ; just like in healthcare, legal, or enterprise settings where data confidentiality is crucial.

To achieve this, I used Flask (Python) for the backend and AES encryption to protect files both at rest and during transfer.

Environment Setup

- Operating System: Kali Linux (via VMware)
- Framework: Python Flask
- Encryption Tool: PyCryptodome (for AES)
- IDE/Tools Used: Terminal, Nano Editor, Firefox browser

Steps Taken

Set Up Project Folder, Installed flash and pycryptodome

```
Sudo apt install python3-venv

source flaskenv/bin/activate

mkdir secure-file-share

cd secure-file-share

python3 -m venv flaskenv

pip install flask pycryptodome
```

Created Flask App Structure

Folders:

- `uploads/`: Where encrypted and decrypted files are stored
- `templates/`: HTML templates folder

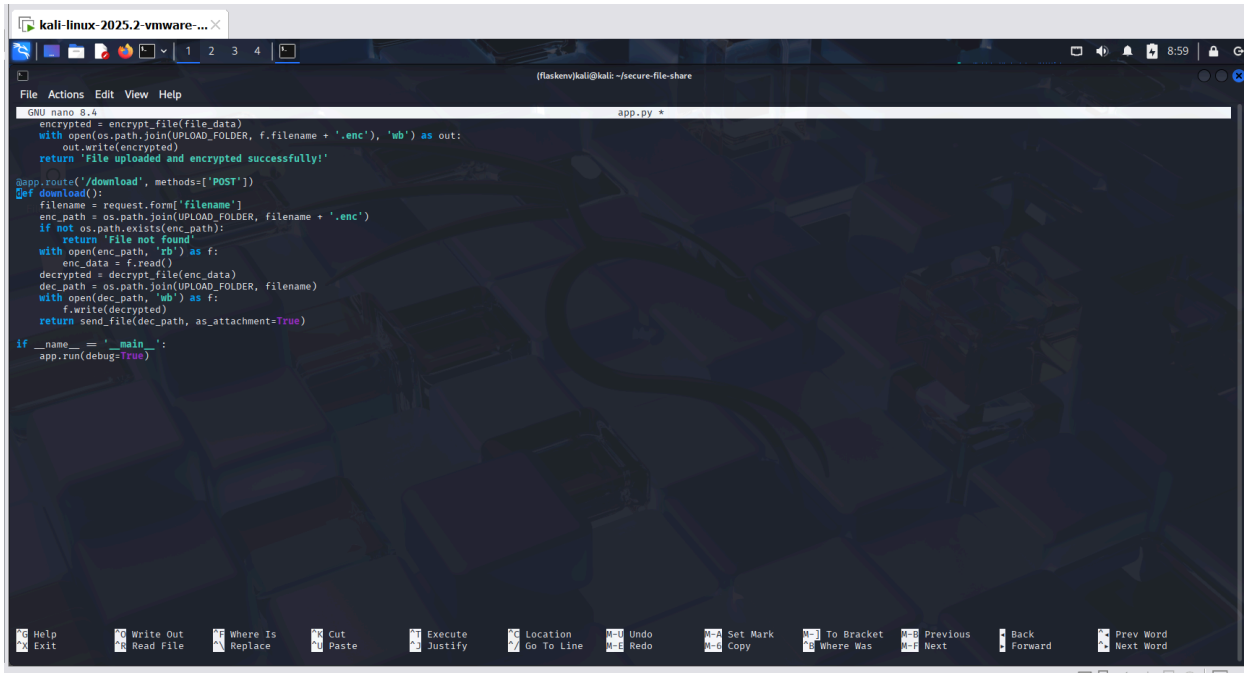
Files:

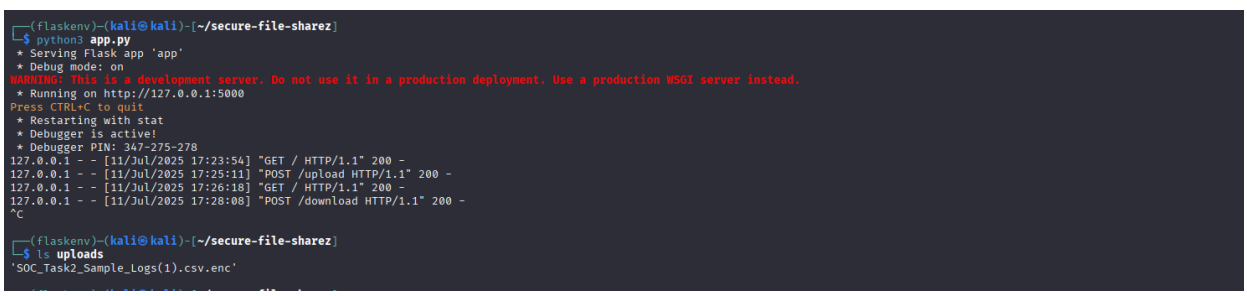
- `app.py`: Main Flask application
- `templates/index.html`: Web interface for file upload/download

```
(flaskenv)kali@kali: ~/secure-file-share
File  Actions  Edit  View  Help
Collecting markupsafe>=2.1.1 (from Flask)
  Downloading MarkupSafe-3.0.2-cp313-cp313-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (4.0 kB)
Collecting werkzeug>=3.1.0 (from Flask)
  Downloading werkzeug-3.1.3-py3-none-any.whl.metadata (3.7 kB)
Downloading flask-3.1.1-py3-none-any.whl (103 kB)
Downloading pycryptodome-3.23.0-cp37-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.3/2.3 MB 3.9 MB/s eta 0:00:00
Downloading blinker-1.9.0-py3-none-any.whl (8.5 kB)
Downloading click-8.2.1-py3-none-any.whl (102 kB)
Downloading itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Downloading jinja2-3.1.6-py3-none-any.whl (134 kB)
Downloading MarkupSafe-3.0.2-cp313-cp313-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (23 kB)
Downloading werkzeug-3.1.3-py3-none-any.whl (224 kB)
Installing collected packages: pycryptodome, markupsafe, itsdangerous, click, blinker, werkzeug, jinja2, Flask
Successfully installed Flask-3.1.1 blinker-1.9.0 click-8.2.1 itsdangerous-2.2.0 jinja2-3.1.6 markupsafe-3.0.2 pycryptodome-3.23.0 werkzeug-3.1.3

┌──(flaskenv)─(kali㉿kali)-[~]
└─$ mkdir secure-file-share

┌──(flaskenv)─(kali㉿kali)-[~]
└─$ cd secure-file-share

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-share]
└─$ touch app.py

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-share]
└─$ mkdir uploads

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-share]
└─$ touch templates/index.html
touch: cannot touch 'templates/index.html': No such file or directory

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-share]
└─$ touch templates/index.html
touch: cannot touch 'templates/index.html': No such file or directory

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-share]
└─$ mkdir templates

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-share]
└─$ touch templates/index.html

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-share]
└─$ nano templates/index.html

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-share]
└─$
```

```
kali-linux-2025.2-vmware-...
(flaskenv)kali@kali: ~/secure-file-sharez
File  Actions  Edit  View  Help
  GNU nano 8.4                          templates/index.html *
<!DOCTYPE html>
<html>
<head>
    <title>Secure File Upload</title>
</head>
<body>
    <h2>Upload a File</h2>
    <form action="/upload" method="post" enctype="multipart/form-data">
        <input type="file" name="file" required>
        <button type="submit">Upload</button>
    </form>

    <h2>Download a File</h2>
    <form action="/download" method="post">
        <input type="text" name="filename" placeholder="Enter filename" required>
        <button type="submit">Download</button>
    </form>
</body>
</html>

^G Help      ^O Write Out   ^W Where Is    ^K Cut       ^T Execute   ^C Location   M-U Undo      M-A Set Mark   M-] To Bracket  M-B Previous    ^← Back        ^← Prev Word
^X Exit      ^R Read File   ^\ Replace     ^U Paste     ^J Justify   ^/ Go To Line  M-E Redo      M-6 Copy       ^Q Where Was    M-F Next        ^→ Forward     ^→ Next Word
```

```
(flaskenv)kali@kali: ~/secure-file-sharez
File  Actions  Edit  View  Help
┌──(flaskenv)─(kali㉿kali)-[~]
└─$ mkdir secure-file-sharez

┌──(flaskenv)─(kali㉿kali)-[~]
└─$ cd secure-file-sharez

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-sharez]
└─$ touch app.py

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-sharez]
└─$ mkdir uploads

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-sharez]
└─$ mkdir templates

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-sharez]
└─$ touch templates/index.html

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-sharez]
└─$ nano template/index.html

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-sharez]
└─$ [200~nano template/index.html
zsh: bad pattern: [200~nano

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-sharez]
└─$ nano templates/index.html

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-sharez]
└─$ touch app.py

┌──(flaskenv)─(kali㉿kali)-[~/secure-file-sharez]
└─$ nano app.py
```





Built Secure Upload Logic

- Used `request.files` to grab the file
- Encrypted file using **AES (EAX Mode)**
- Stored `.enc` files in an `uploads/` folder

Implemented Secure Download Logic

- User enters the original filename
- The app finds the encrypted `.enc` version
- Decrypts it and returns the original file

Created User Interface

- A simple HTML form to:
  - Upload files
  - Download files

Tested the System

- Uploaded `.csv`, `.txt`, and other files
- Verified successful encryption and decryption
- Monitored file changes in the `uploads/` directory

AES Encryption Overview

- Used `PyCryptodome`'s AES in **EAX mode**
- Key was static for demo: `b'ThisIsASecretKey'`

  Encrypted file stored as:

  `nonce + tag + ciphertext`
- During download:

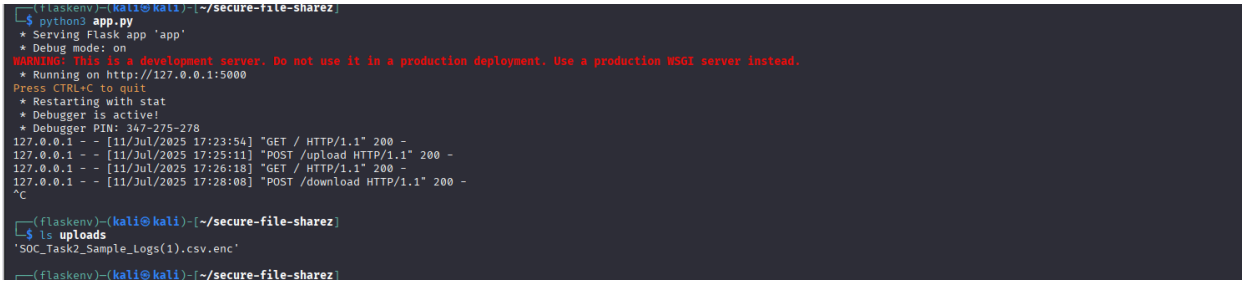
- Nonce and tag verified
- Decryption returns the original file

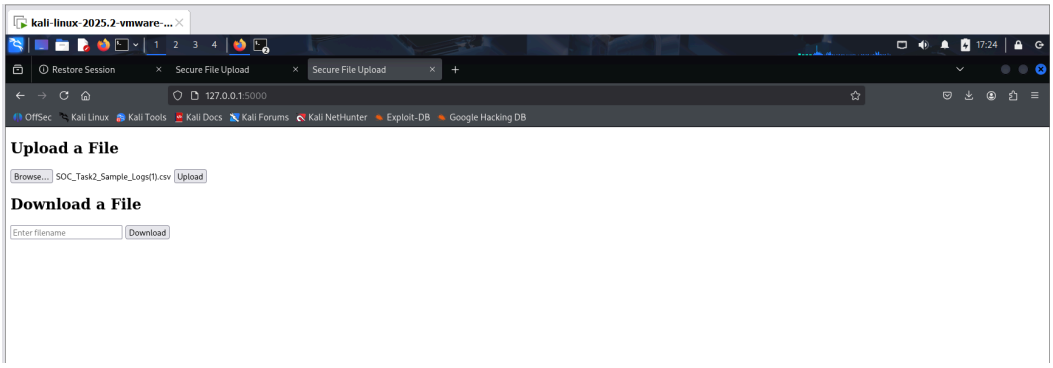
Project Structure

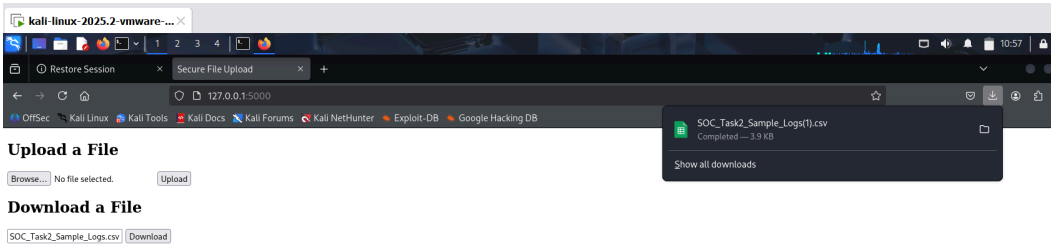
```
secure-file-share/
├── app.py
├── uploads/
│   └── filename.enc
├── templates/
│   └── index.html
```

Visual Evidence

- Upload form success message
- Encrypted file inside `uploads/`
- Download form decrypted successfully
- Flask running at `127.0.0.1:500`







Key Takeaways

- Learned Flask backend and routing
- Understood AES symmetric encryption workflow
- Built a fully working secure file transfer system
- Practiced file I/O, data security, and Python logic
- Improved confidence in secure app development