Secure File Sharing System Report

Name: Kumari Tulsi

Task 3: Secure File Sharing System

Program: Future Interns Cybersecurity Internship

Date: September 2025

Technology Used: Python, Flask, HTML, CSS, JavaScript, AES Encryption

1. Introduction

The **Secure File Sharing System** is a web-based application that allows users to upload, download, and delete files securely.
All uploaded files are **encrypted using AES-256 encryption** before being stored, ensuring data confidentiality and security.
This project demonstrates the use of Flask for backend development, cryptography for encryption, and modern frontend technologies (HTML, CSS, JS) for a responsive UI.

2. Objectives

- To provide a **secure platform** for file sharing.
- To implement **AES-256 encryption** for file protection.
- To allow users to **upload, download, and delete** files.
- To design a **user-friendly web interface** with HTML, CSS, and JavaScript.

3. Tools & Technologies

- **Backend**: Python (Flask)
- **Frontend**: HTML5, CSS3, JavaScript
- **Security**: AES (via pycryptodome library)
- **Other tools**: Virtual Environment (venv), pip, Linux Terminal

4. System Requirements

- Python 3.10+
- Flask Framework
- PyCryptodome Library
- Browser (Firefox / Chrome)
- Operating System: Linux (Kali used for implementation)

5. Implementation Steps

1. **Environment Setup**
    - Created a project folder: secure-file-sharing
    - Initialized Python virtual environment:

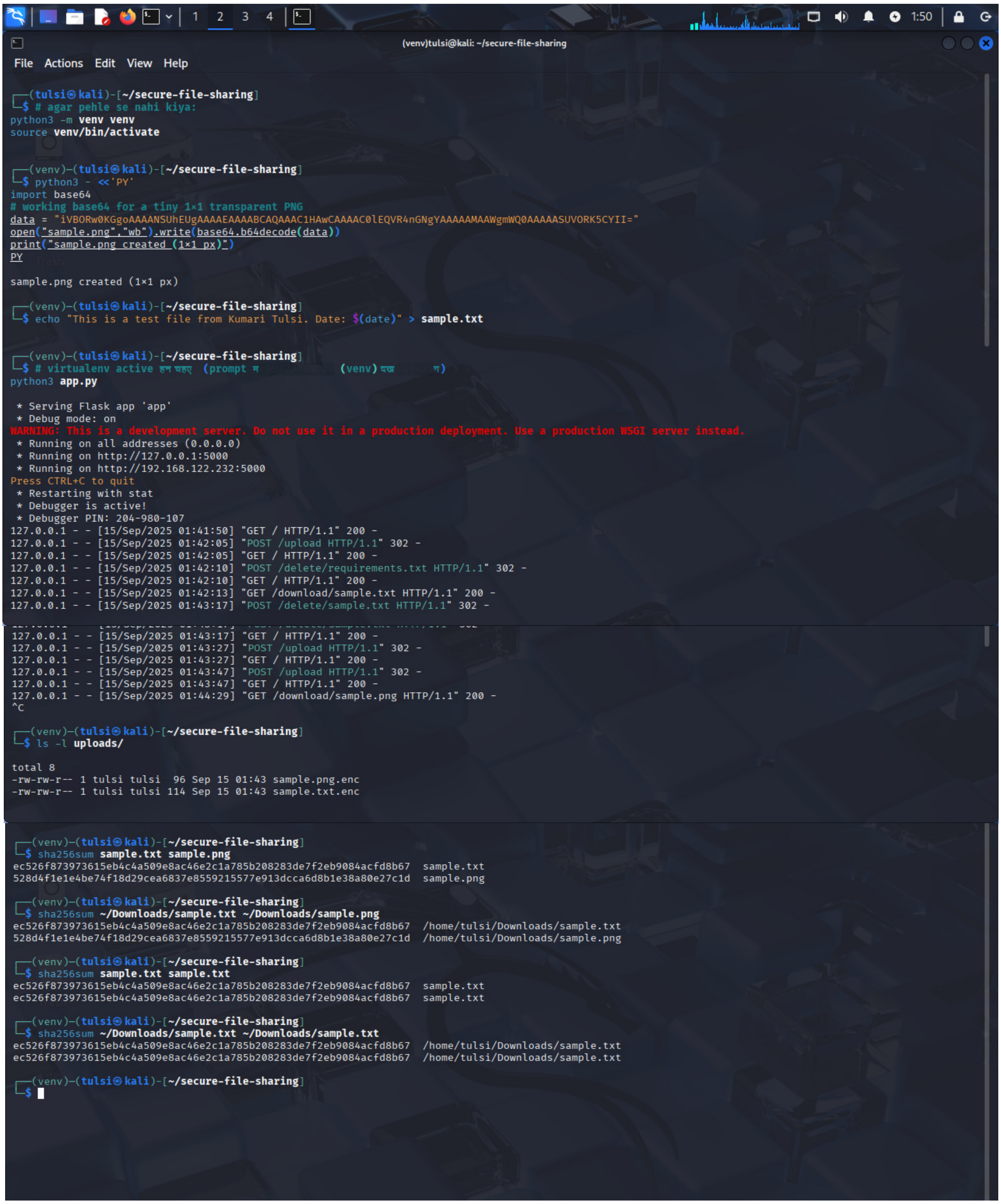
      ```
      python3 -m venv venv
      source venv/bin/activate
      ```

    - Installed dependencies:

      ```
      pip install Flask pycryptodome
      ```

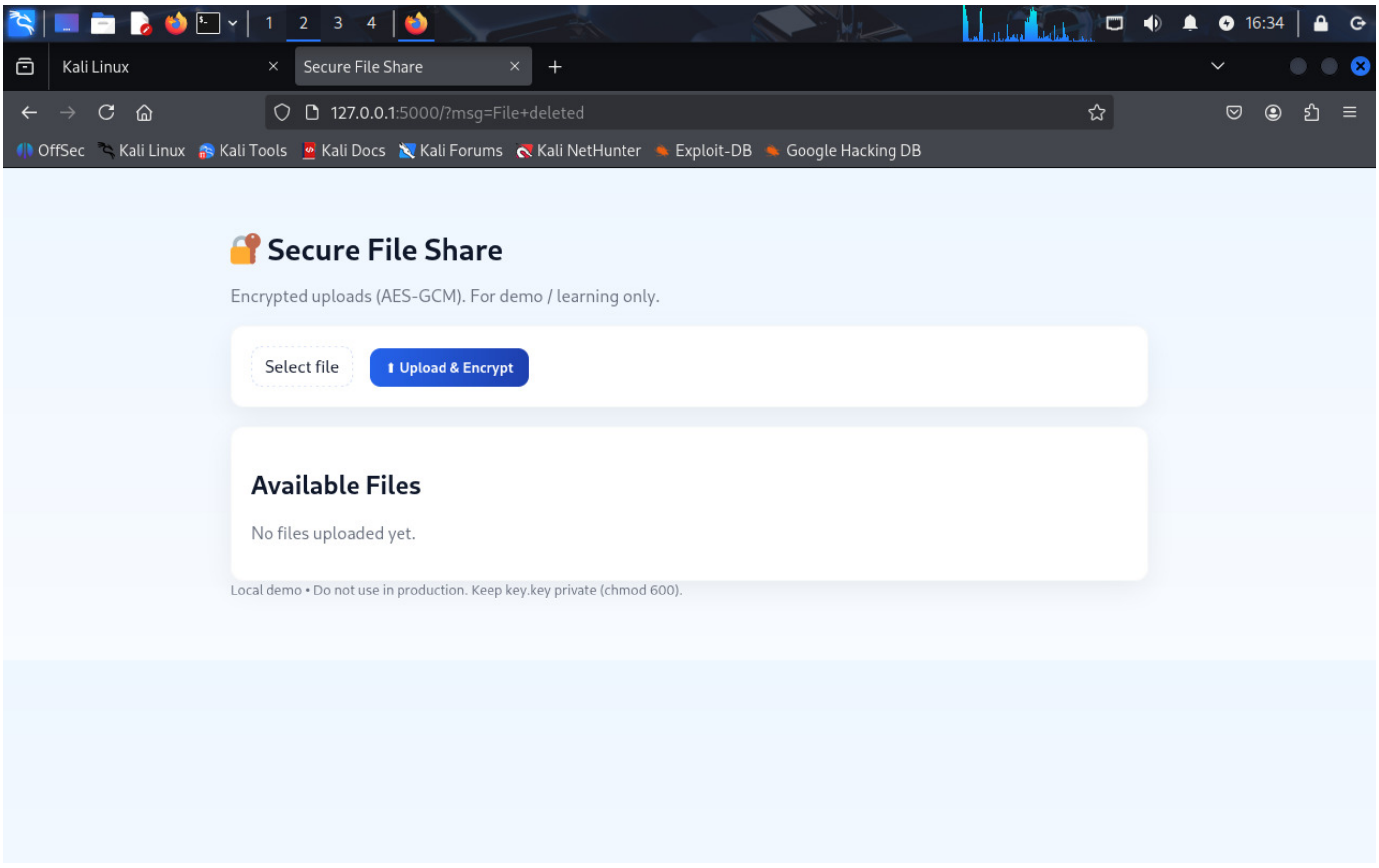
2. **AES Key Generation**
    - Generated a 32-byte AES key and saved as key.key.
3. **Backend (Flask App)**
    - Routes implemented:
        - / → Home page (list files)
        - /upload → Upload + Encrypt file
        - /download/<filename> → Decrypt + Download file
        - /delete/<filename> → Delete file
4. **Frontend (HTML, CSS, JS)**
    - Created responsive UI with:
        - Upload button
        - File listing section
        - Download/Delete options
        - Notification messages
5. **Encryption & Decryption**
    - Used **AES-GCM mode** for encryption (confidentiality + integrity).
    - Encrypted files stored with .enc extension.

6. Output Screenshots



```
┌──(tulsi㉿kali)-[~/secure-file-sharing]
└─$ # agar pehle se nahi kiya:
python3 -m venv venv
source venv/bin/activate


┌──(venv)─(tulsi㉿kali)-[~/secure-file-sharing]
└─$ python3 - <<'PY'
import base64
# working base64 for a tiny 1×1 transparent PNG
data = "iVBORw0KGgoAAAANSUhEUgAAAAEAAAABCAQAAAC1HAwCAAAAC0lEQVR4nGNgYAAAAAMAAWgmWQ0AAAAASUVORK5CYII="
open("sample.png","wb").write(base64.b64decode(data))
print("sample.png created (1×1 px)")
PY

sample.png created (1×1 px)

┌──(venv)─(tulsi㉿kali)-[~/secure-file-sharing]
└─$ echo "This is a test file from Kumari Tulsi. Date: $(date)" > sample.txt


┌──(venv)─(tulsi㉿kali)-[~/secure-file-sharing]
└─$ # virtualenv active हन चहए (prompt म             (venv) दख        न)
python3 app.py

 * 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 instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://192.168.122.232:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 204-980-107
127.0.0.1 - - [15/Sep/2025 01:41:50] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2025 01:42:05] "POST /upload HTTP/1.1" 302 -
127.0.0.1 - - [15/Sep/2025 01:42:05] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2025 01:42:10] "POST /delete/requirements.txt HTTP/1.1" 302 -
127.0.0.1 - - [15/Sep/2025 01:42:10] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2025 01:42:13] "GET /download/sample.txt HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2025 01:43:17] "POST /delete/sample.txt HTTP/1.1" 302 -
127.0.0.1 - - [15/Sep/2025 01:43:17] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2025 01:43:27] "POST /upload HTTP/1.1" 302 -
127.0.0.1 - - [15/Sep/2025 01:43:27] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2025 01:43:47] "POST /upload HTTP/1.1" 302 -
127.0.0.1 - - [15/Sep/2025 01:43:47] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2025 01:44:29] "GET /download/sample.png HTTP/1.1" 200 -
^C

┌──(venv)─(tulsi㉿kali)-[~/secure-file-sharing]
└─$ ls -l uploads/

total 8
-rw-rw-r-- 1 tulsi tulsi  96 Sep 15 01:43 sample.png.enc
-rw-rw-r-- 1 tulsi tulsi 114 Sep 15 01:43 sample.txt.enc


┌──(venv)─(tulsi㉿kali)-[~/secure-file-sharing]
└─$ sha256sum sample.txt sample.png
ec526f873973615eb4c4a509e8ac46e2c1a785b208283de7f2eb9084acfd8b67  sample.txt
528d4f1e1e4be74f18d29cea6837e8559215577e913dcca6d8b1e38a80e27c1d  sample.png

┌──(venv)─(tulsi㉿kali)-[~/secure-file-sharing]
└─$ sha256sum ~/Downloads/sample.txt ~/Downloads/sample.png
ec526f873973615eb4c4a509e8ac46e2c1a785b208283de7f2eb9084acfd8b67  /home/tulsi/Downloads/sample.txt
528d4f1e1e4be74f18d29cea6837e8559215577e913dcca6d8b1e38a80e27c1d  /home/tulsi/Downloads/sample.png

┌──(venv)─(tulsi㉿kali)-[~/secure-file-sharing]
└─$ sha256sum sample.txt sample.txt
ec526f873973615eb4c4a509e8ac46e2c1a785b208283de7f2eb9084acfd8b67  sample.txt
ec526f873973615eb4c4a509e8ac46e2c1a785b208283de7f2eb9084acfd8b67  sample.txt

┌──(venv)─(tulsi㉿kali)-[~/secure-file-sharing]
└─$ sha256sum ~/Downloads/sample.txt ~/Downloads/sample.txt
ec526f873973615eb4c4a509e8ac46e2c1a785b208283de7f2eb9084acfd8b67  /home/tulsi/Downloads/sample.txt
ec526f873973615eb4c4a509e8ac46e2c1a785b208283de7f2eb9084acfd8b67  /home/tulsi/Downloads/sample.txt

┌──(venv)─(tulsi㉿kali)-[~/secure-file-sharing]
└─$ 
```

- **Homepage after server start**



127.0.0.1:5000/?msg=File+deleted

Secure File Share

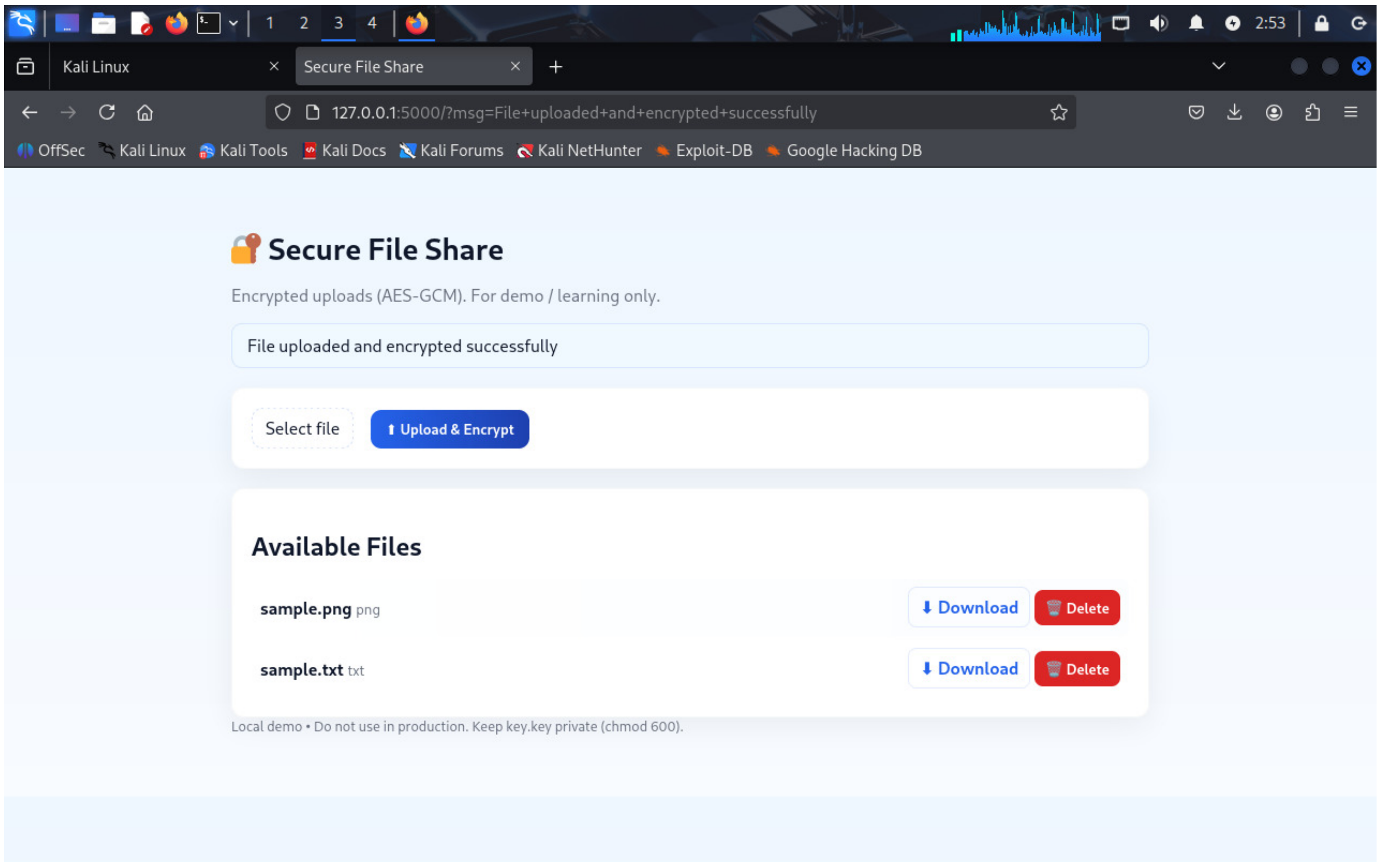
Encrypted uploads (AES-GCM). For demo / learning only.

Available Files

No files uploaded yet.

Local demo • Do not use in production. Keep key.key private (chmod 600).

- **File upload success message**



127.0.0.1:5000/?msg=File+uploaded+and+encrypted+successfully

Secure File Share

Encrypted uploads (AES-GCM). For demo / learning only.

File uploaded and encrypted successfully

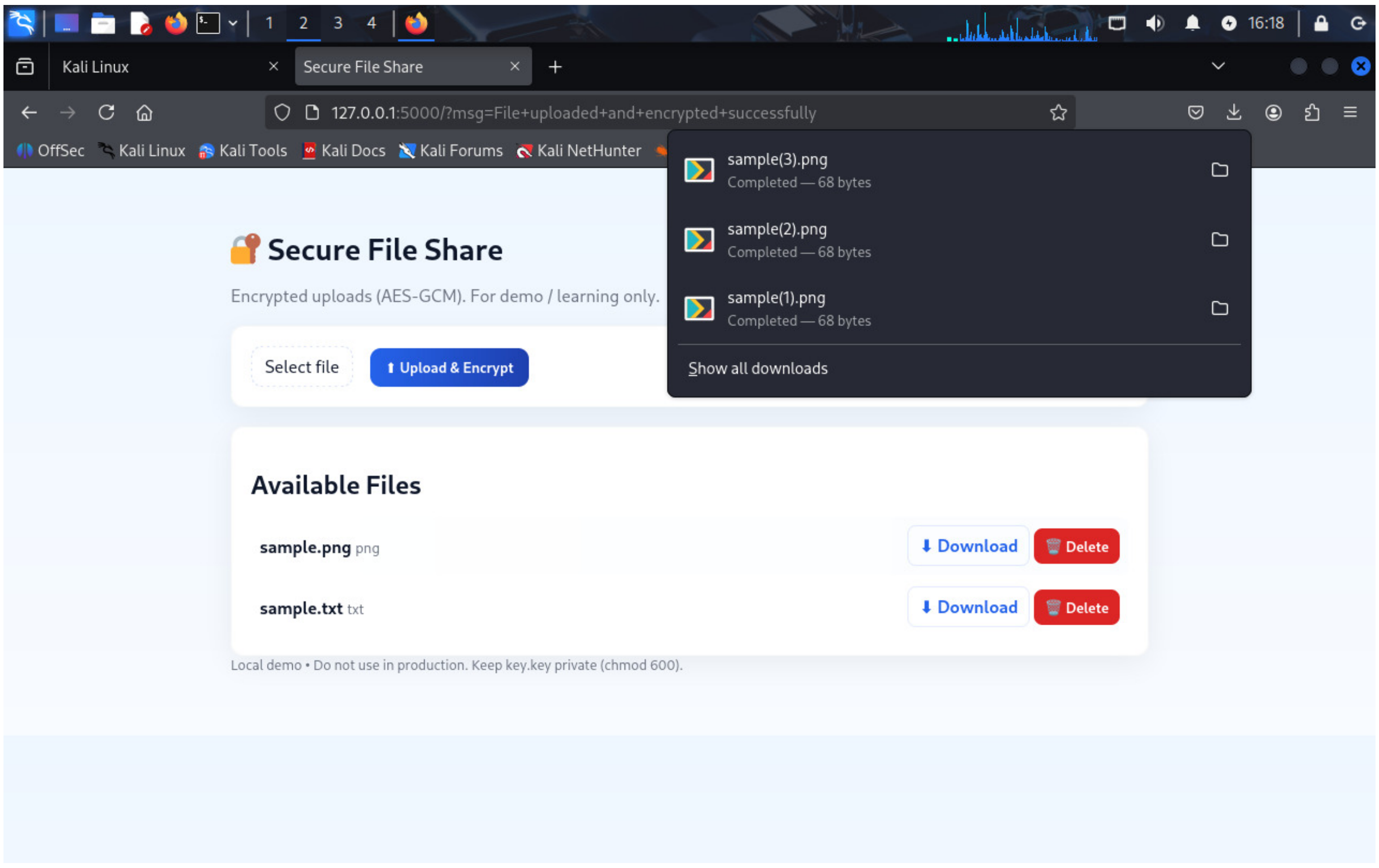
Available Files

sample.png png

sample.txt txt

Local demo • Do not use in production. Keep key.key private (chmod 600).

- **Downloaded decrypted file**



127.0.0.1:5000/?msg=File+uploaded+and+encrypted+successfully

sample(3).png
Completed — 68 bytes

sample(2).png
Completed — 68 bytes

sample(1).png
Completed — 68 bytes

Show all downloads

Secure File Share

Encrypted uploads (AES-GCM). For demo / learning only.

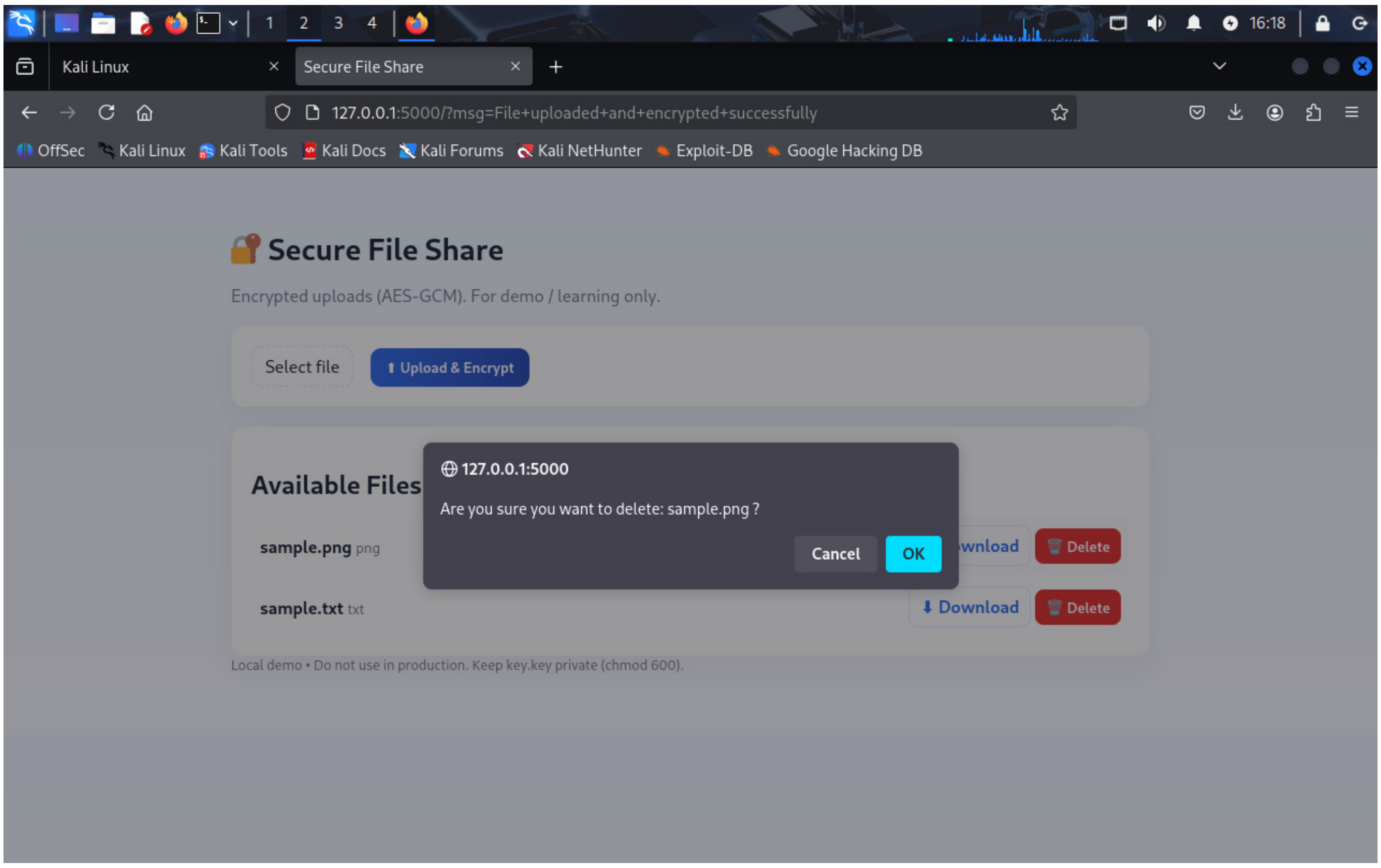
Available Files

sample.png png

sample.txt txt

Local demo • Do not use in production. Keep key.key private (chmod 600).

- **Delete confirmation**



127.0.0.1:5000/?msg=File+uploaded+and+encrypted+successfully

Secure File Share

Encrypted uploads (AES-GCM). For demo / learning only.

Available Files

127.0.0.1:5000

Are you sure you want to delete: sample.png ?

sample.png png

sample.txt txt

Local demo • Do not use in production. Keep key.key private (chmod 600).

7. Results

- Successfully uploaded and encrypted files.
- Files downloaded in decrypted form.
- Files deleted securely.
- UI is simple, user-friendly, and responsive.

8. Conclusion

The project **successfully implements a secure file sharing system** with AES encryption.
This ensures that sensitive data is protected during storage and transfer.
It demonstrates the use of Python Flask for backend, and HTML/CSS/JS for frontend design.

9. Future Enhancements

- Add **user authentication** (Login/Signup).
- Implement **file size limits**.
- Support **cloud storage (AWS/GCP/Azure)**.
- Add **audit logs** for file activity tracking.

10. References

- Flask Documentation: https://flask.palletsprojects.com/
- PyCryptodome Docs: https://pycryptodome.readthedocs.io/
- Python Official Docs: https://docs.python.org/