

Secure file sharing system  
Cybersecurity internship Task\_03  
Future interns

**Title :** secure file sharing system

**Intern name :** ODAI WAHIB

**Date :** 28-08-2025

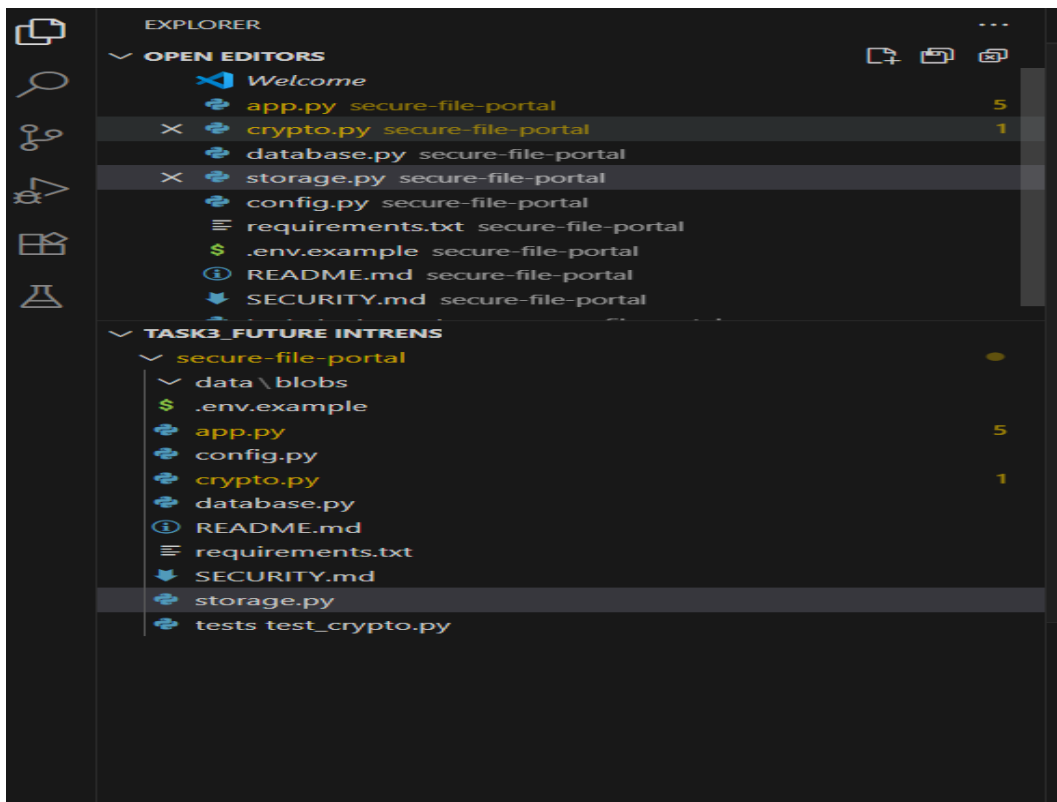
## Secure File Portal - Project Report

### Introduction

The Secure File Portal is a Flask-based web application that provides a secure way to upload and download files. The project ensures data confidentiality by using AES-GCM encryption with an envelope encryption model. Each file is encrypted with a Data Encryption Key (DEK), which is further wrapped with a Key Encryption Key (KEK).

### Features

- Upload files securely through the web interface or using curl commands.
- Files are encrypted with AES-GCM before storage.
- Encrypted file metadata (nonce, tag, wrapped DEK) stored in database.
- Files can be downloaded and automatically decrypted.
- Uses Flask framework with SQLAlchemy for database handling.
- Simple and secure design for confidentiality.



## System Requirements

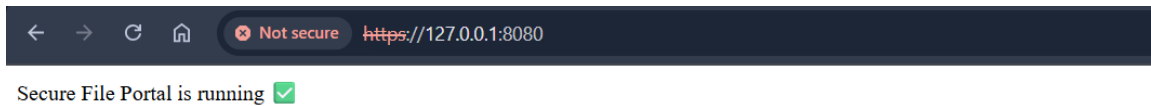
- Python 3.9 or above
- Flask
- SQLAlchemy
- cryptography library
- curl (for testing uploads/downloads)

```
Downloading Jinja2-3.1.6-py3-none-any.whl.metadata (2.9 kB)
Collecting itsdangerous>=2.1.2 (from Flask==3.0.3->requirements.txt (line 1))
  Downloading itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)
Collecting click>=8.1.3 (from Flask==3.0.3->requirements.txt (line 1))
  Downloading click-8.2.1-py3-none-any.whl.metadata (2.5 kB)
Collecting blinker>=1.6.2 (from Flask==3.0.3->requirements.txt (line 1))
  Downloading blinker-1.9.0-py3-none-any.whl.metadata (1.6 kB)
Collecting typing_extensions>=4.6.0 (from SQLAlchemy==2.0.36->requirements.txt (line 4))
  Downloading typing_extensions-4.15.0-py3-none-any.whl.metadata (3.3 kB)
Collecting MarkupSafe>=2.1.1 (from Werkzeug==3.0.3->requirements.txt (line 5))
  Downloading MarkupSafe-3.0.2-cp313-cp313-win_amd64.whl.metadata (4.1 kB)
Collecting colorama (from click>=8.1.3->Flask==3.0.3->requirements.txt (line 1))
  Downloading colorama-0.4.6-py2.py3-none-any.whl.metadata (17 kB)
Downloading flask-3.0.3-py3-none-any.whl (101 kB)
Downloading python_dotenv-1.0.1-py3-none-any.whl (19 kB)
Downloading pycryptodome-3.20.0-cp35-abi3-win_amd64.whl (1.8 MB)
 1.8/1.8 MB 11.2 MB/s 0:00:00
Downloading SQLAlchemy-2.0.36-cp313-cp313-win_amd64.whl (2.1 MB)
 2.1/2.1 MB 10.6 MB/s 0:00:00
Downloading werkzeug-3.0.3-py3-none-any.whl (227 kB)
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-win_amd64.whl (15 kB)
Downloading typing_extensions-4.15.0-py3-none-any.whl (44 kB)
Downloading colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Installing collected packages: typing_extensions, python-dotenv, PyCryptodome, MarkupSafe, itsdangerous, colorama, blinker, Werkzeug, SQLAlchemy, Jinja2, click, Flask
Successfully installed Flask-3.0.3 Jinja2-3.1.6 MarkupSafe-3.0.2 PyCryptodome-3.20.0 SQLAlchemy-2.0.36 Werkzeug-3.0.3 blinker-1.9.0 click-8.2.1 colorama-0.4.6 itsdangerous-2.2.0 python-dotenv-1.0.1 typing_extensions-4.15.0
```

## Project Structure

The project consists of the following main files:

- app.py : Flask application entry point.
- crypto.py : Encryption and decryption logic.
- storage.py : File storage and retrieval logic using database.
- database.py : Database models and migration setup.
- config.py : Configuration file (database URL, KEK, etc.).



## Workflow

1. Start the Flask server by running: `python app.py`

2. Upload a file using curl:

```
curl.exe -k -F "file=@test.txt" https://127.0.0.1:8080/upload
```

Response will return a unique `file_id`.

3. Download the file using:

```
curl.exe -k -o downloaded.txt https://127.0.0.1:8080/download/<file_id>
```

4. The file is decrypted automatically upon download.

```
PS D:\task3_future_intrens\secure-file-portal> python app.py
* Serving Flask app 'app'
* Debug mode: off
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 https://127.0.0.1:8080
* Running on https://172.20.10.2:8080
Press CTRL+C to quit
172.20.10.2 - - [28/Aug/2025 02:02:24] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [28/Aug/2025 02:02:39] "POST /upload HTTP/1.1" 200 -
127.0.0.1 - - [28/Aug/2025 02:03:46] "GET /download/729d14ad-05f2-4540-b364-45069762cc72 HTTP/1.1" 200 -
```

## Conclusion

The Secure File Portal successfully demonstrates a secure system for handling files using Flask and cryptographic best practices. With AES-GCM encryption and KEK/DEK envelope encryption, it ensures that uploaded files remain confidential and protected.