# The Secure Photo Vault

December 20, 2024

# Contents

# Introduction

- **Project Goal:** Develop a secure cloud-based web application for managing photos with advanced encryption and authentication techniques.
- Photos are stored in an encrypted format to ensure confidentiality and integrity.
- Integration of Two-Factor Authentication (2FA) for enhanced user security.
- Utilizes End-to-End Encryption (E2EE) to protect data transmitted over the network.

# Scope

- Privacy Protection: Addresses critical privacy concerns by safeguarding sensitive data against unauthorized access.
- Data Security: Ensures the confidentiality and integrity of user photos through advanced encryption techniques.
- Enhanced Authentication: Utilizes Two-Factor Authentication (2FA) for robust user security.
- Secure Communication: Implements End-to-End Encryption for protecting data transmission.
- Efficient Photo Management: Facilitates easy image management and retrieval with automatic tagging and a comprehensive search and filter systems.

# Relevance

- Increased Privacy Concerns:Users are more worried about data breaches and privacy issues.
- Cloud Storage and Data Security:As cloud storage use grows, securing data and access controls are crucial.
- Increased Risk of Exploitation:Leaked images can be exploited for malicious purposes, such as blackmail or harassment.
- Increasing Threats of Data Leaks:Frequent data breaches highlight the need for strong security.
- Privacy Violations:Unauthorized exposure of personal images can lead to significant breaches of privacy and personal safety.

# Requirement Analysis

- **Existing System:**

- Basic Photo Management with limited functionalities.
- Lacks advanced encryption and robust authentication.
- Often does not use End-to-End Encryption.
- Requires time-consuming manual tagging.
- Insufficient measures to safeguard sensitive data against unauthorized access.

- **Proposed System:**

- Advanced Encryption: Ensures photo confidentiality and integrity.
- Enhanced Authentication: Uses Two-Factor Authentication.
- Implements End-to-End Encrypted communication.
- Automatic tagging and organizing photos using ML.

# Requirement Analysis

**Hardware :**

- Processor: Modern multi-core CPU
- RAM: Minimum 4 GB
- Storage: 50GB
- Display: High-resolution display for optimal interface experience
- Camera: Optional for photo capture

**Software :**

- Operating System: Windows 10 or 11
- Web Browser:Latest versions of Chrome, Firefox or Edge
- Database:MySQL or PostgreSQL

# Development Methodology

- Approach: Use Agile with Scrum for iterative development.
- Front-End Development: HTML, CSS, JavaScript
- Back-End Development: Using Python and Django.
- Database: PostgreSQL or MySQL
- Security: 2FA (TOTP) E2E Encryption
- VS Code IDE for development, testing, and deployment.

# Design

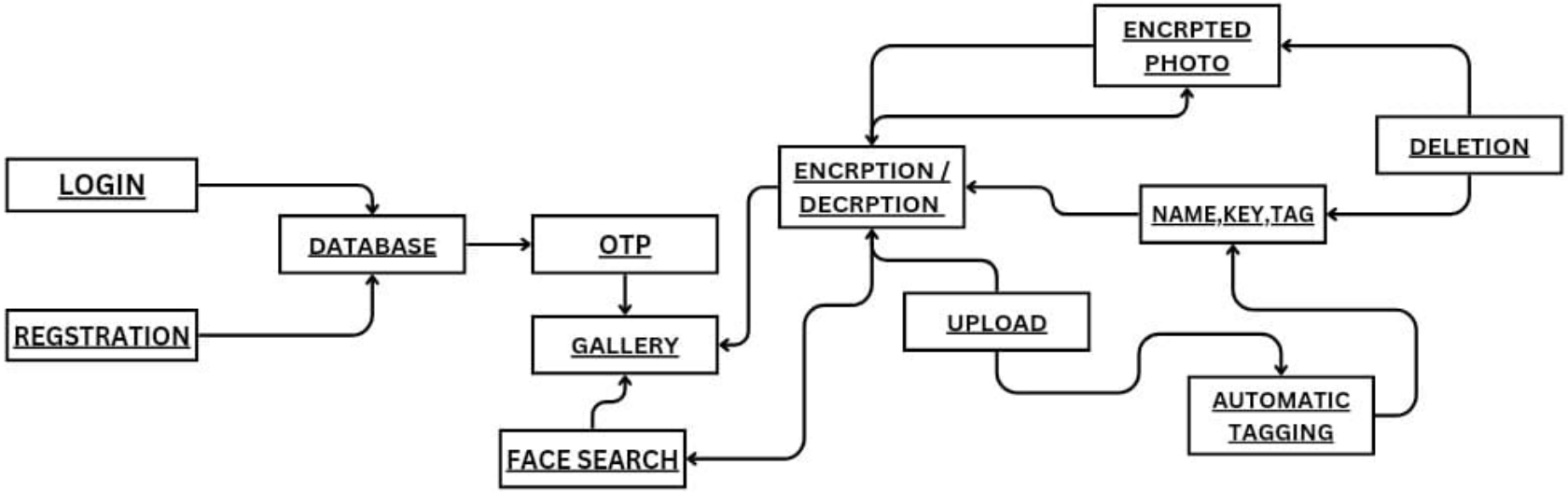



Fig.1. Block Diagram

# Design

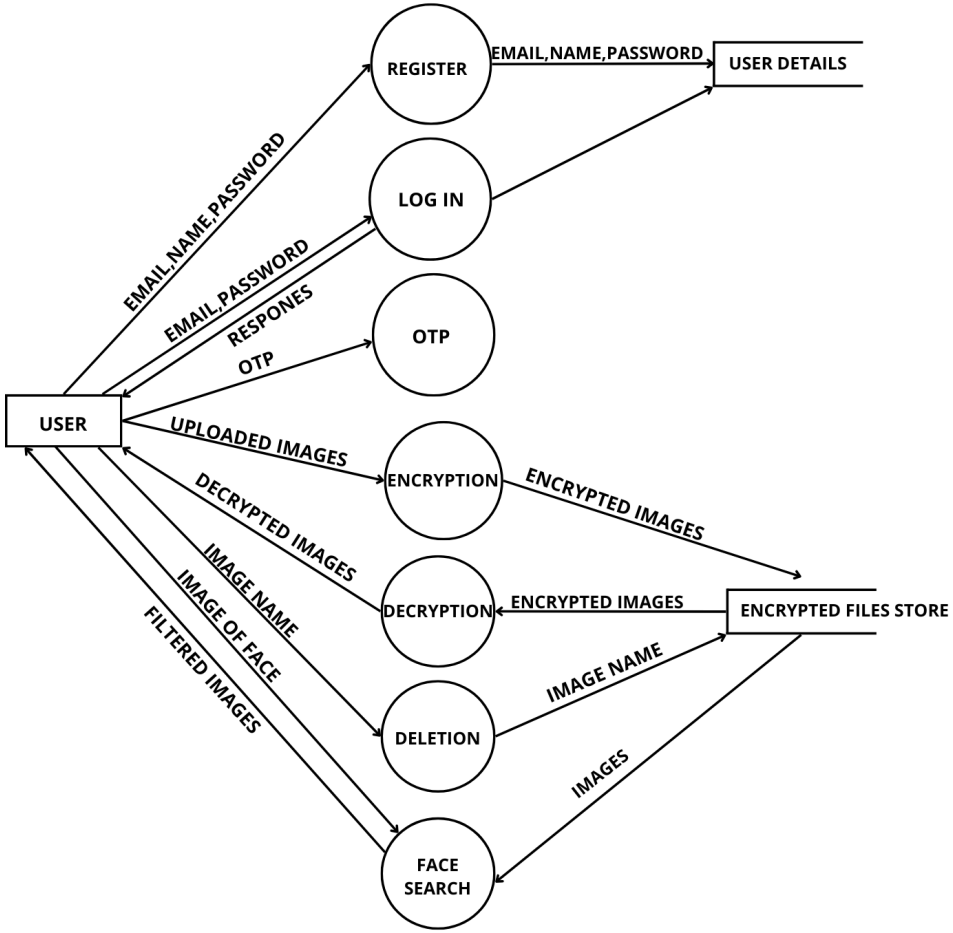



Fig.2. Level 1 DFD of User

# Implementation Details

- Encryption and decryption implemented with Elliptic Curve Cryptography.
- Frontend implemented using (HTML,CSS and Javascript).
- Backend implemented (Django and PostgreSQL).
- Two factor authentication usig OTP.
- Automatic tagging using a fine-tuned VGG16 model.
- face-recognition-based sorting of images.

# Results

- Successfully encrypted images using ECC and AES techniques.
- Utilized two-factor authentication (OTP-based) for securing user accounts.
- Implemented a fine-tuned VGG16 model for automatic image tagging.
- Face recognition-based sorting was implemented by comparing face embeddings.

# Result

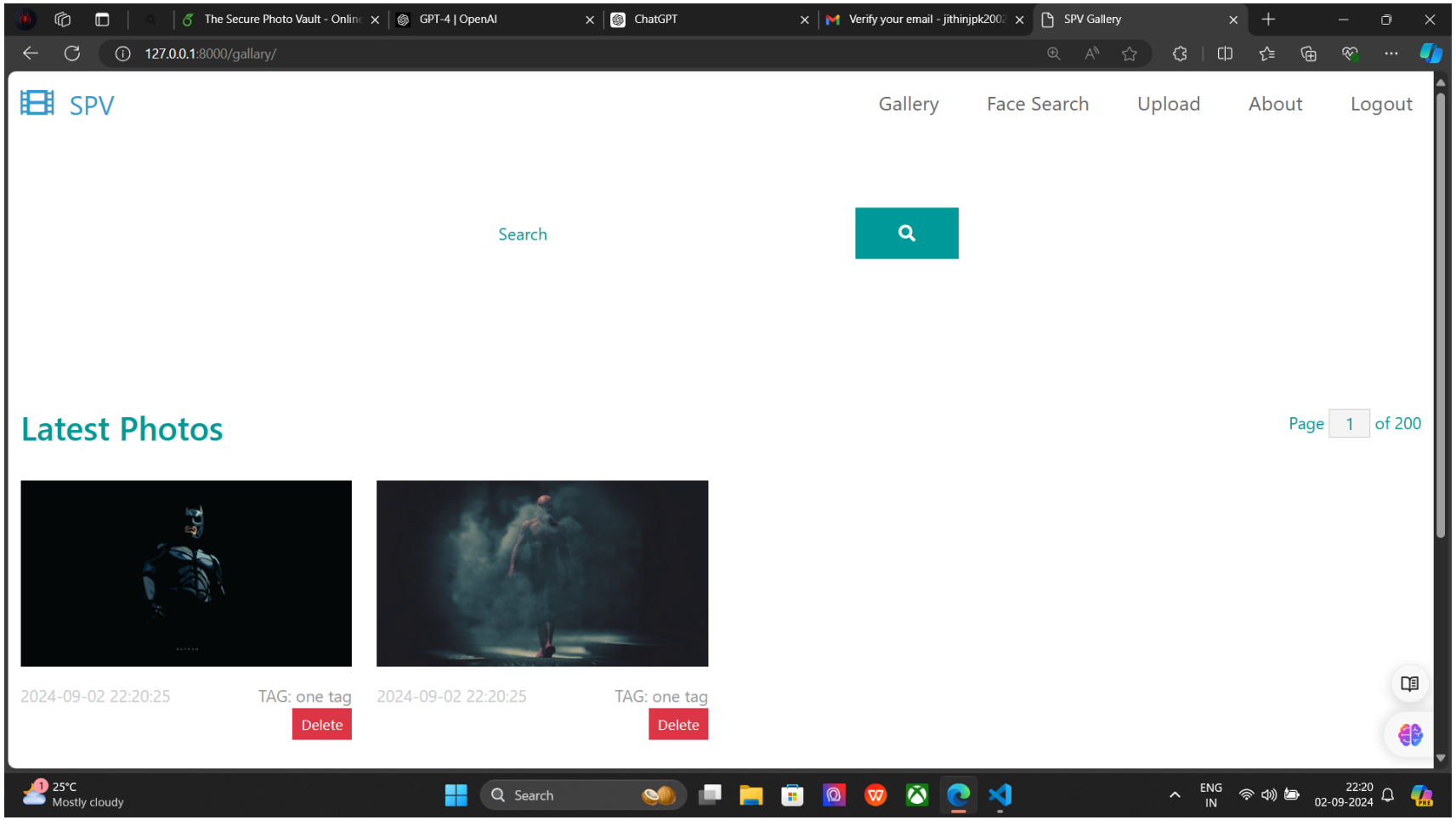
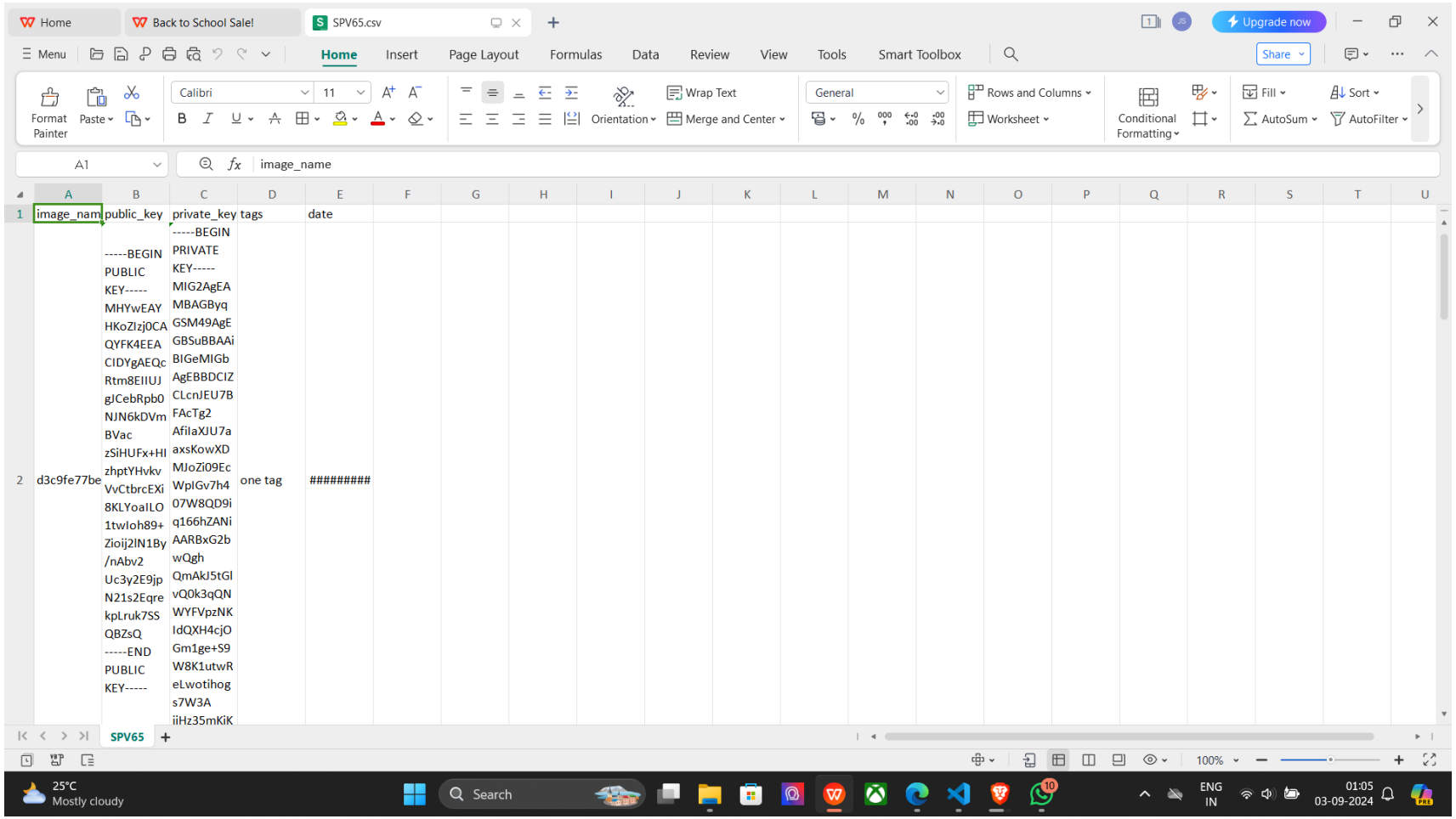



Fig.4. Gallary

# Result



| image_name | public_key | private_key | tags | date |
|---|---|---|---|---|
| d3c9fe77be | -----BEGIN PUBLIC KEY----- MHYwEAYHKoZIzj0CAQYFK4EEACIDYgAEQcRtm8EIIUJgJCebRpb0NJN6kDVmBVac zSiHUFx+HIzhptYHvkvVvCtbrcEXi8KLYoaILO1twIoh89+Zioij2lN1By/nAbv2Uc3y2E9jpN21s2Eqrekp LRuk7SSQBZsQ -----END PUBLIC KEY----- | -----BEGIN PRIVATE KEY----- MIG2AgEAMBAGByqGSM49AgEGBSuBBAAiBIGeMIGbAgEBBDCIZCLcnJEU7BFAcTg2AfiIaXJU7aaxsKowXDMJoZi09EcWplGv7h407W8QD9iq166hZANiAARBxG2bwQghQmAkJ5tGlvQ0k3qQNWYFVpzNKIdQXH4cjOGm1ge+S9W8K1utwReLwotihogs7W3AiiHz35mKiK | one tag | ######### |

Fig.5. Image Meta Data

# Result

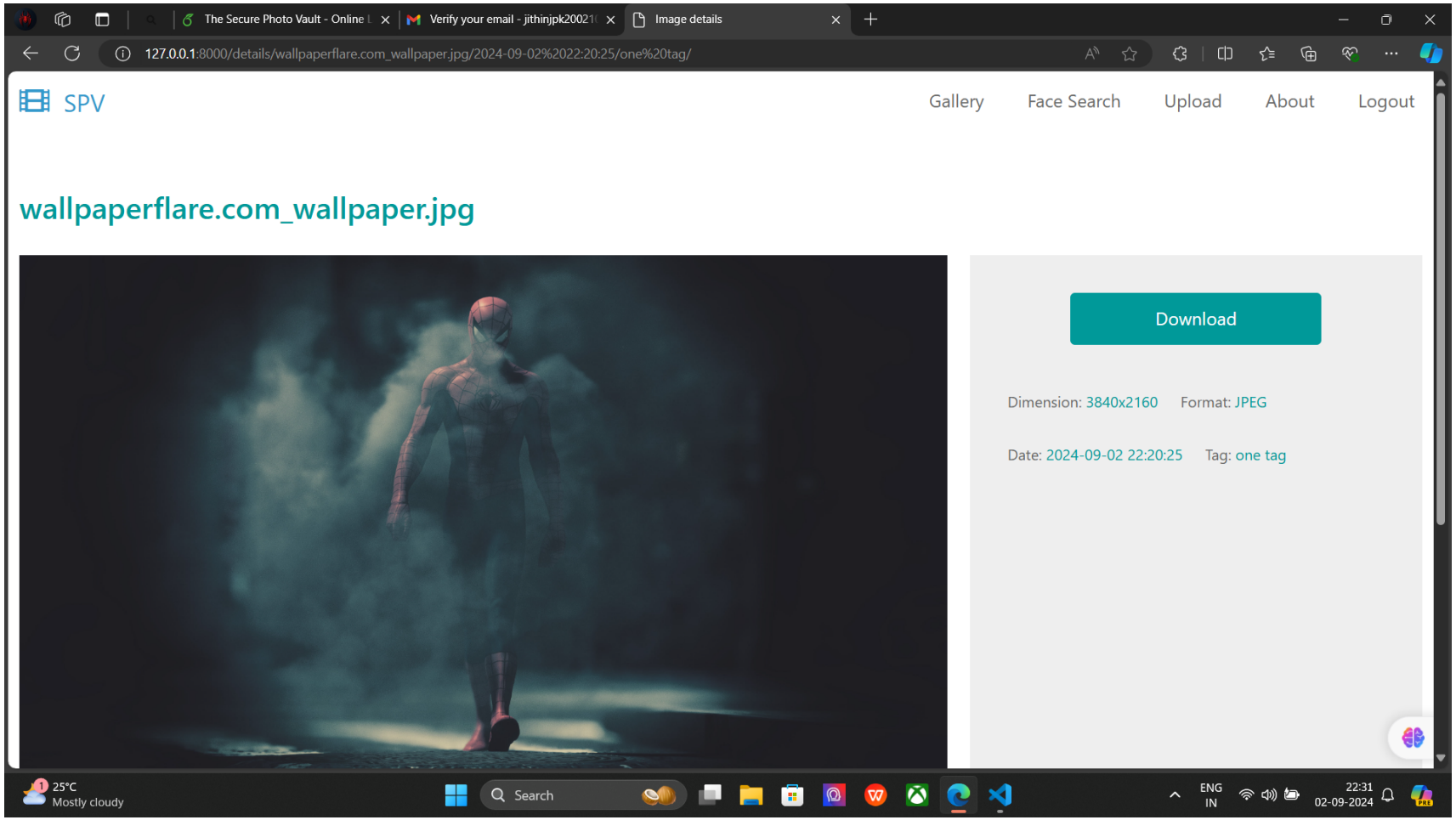



Fig. 6. Image Details

# Result

| Name | Date modified | Type | Size |
|---|---|---|---|
| d3c9fe77be89db14.jpg | 02-09-2024 11:47 | BIN File | 381 KB |
| d6cfb19aa348001d.jpg | 02-09-2024 11:47 | BIN File | 657 KB |
| d93c44ab770c8a8e.jpg | 02-09-2024 11:47 | BIN File | 51 KB |
| d605da7159eed840.jpg | 02-09-2024 11:47 | BIN File | 110 KB |
| d686a08f0473b54b.jpg | 02-09-2024 11:47 | BIN File | 208 KB |

Fig. 7. Encrpted Image Files

# Results Analysis

- The system effectively encrypts images regardless of their file format.
- Effective automatic tagging of any images.
- Effective face recognition-based sorting of images.

# Steps to Encrypt an Image Using Elliptic Curve Cryptography

- Generate Keys: Create a pair of keys using elliptic curve cryptography (ECC) – one public and one private.
- Create Shared Key: Use these keys to generate a shared secret that will be used to create a symmetric key.
- Derive Symmetric Key: Convert the shared secret into a strong encryption key using a process called PBKDF2HMAC.
- Encrypt Data: Use the derived key to encrypt the image data, adding random elements to ensure unique encrypted results.
- Store: Save the encrypted image along with metadata, such as the encryption keys, in a secure format.

# Conclusion

- The proposed system ensures "Secure Photo Vault" uses AES, ECC, E2EE, and 2FA for robust privacy and protection.
- Face recognition and searching mechanism for photo management.
- Built with agile methodology and technologies like HTML, CSS, JavaScript, Python, and Django, the application offers a scalable, secure, and responsive interface for managing photos.

# Future Scope

- Explore biometric authentication for added security layers.
- Extend support for other media formats like video, enabling secure storage and management beyond just images.
- Refine encryption, face detection, and sorting processes to improve speed and reduce processing time..

# Git History

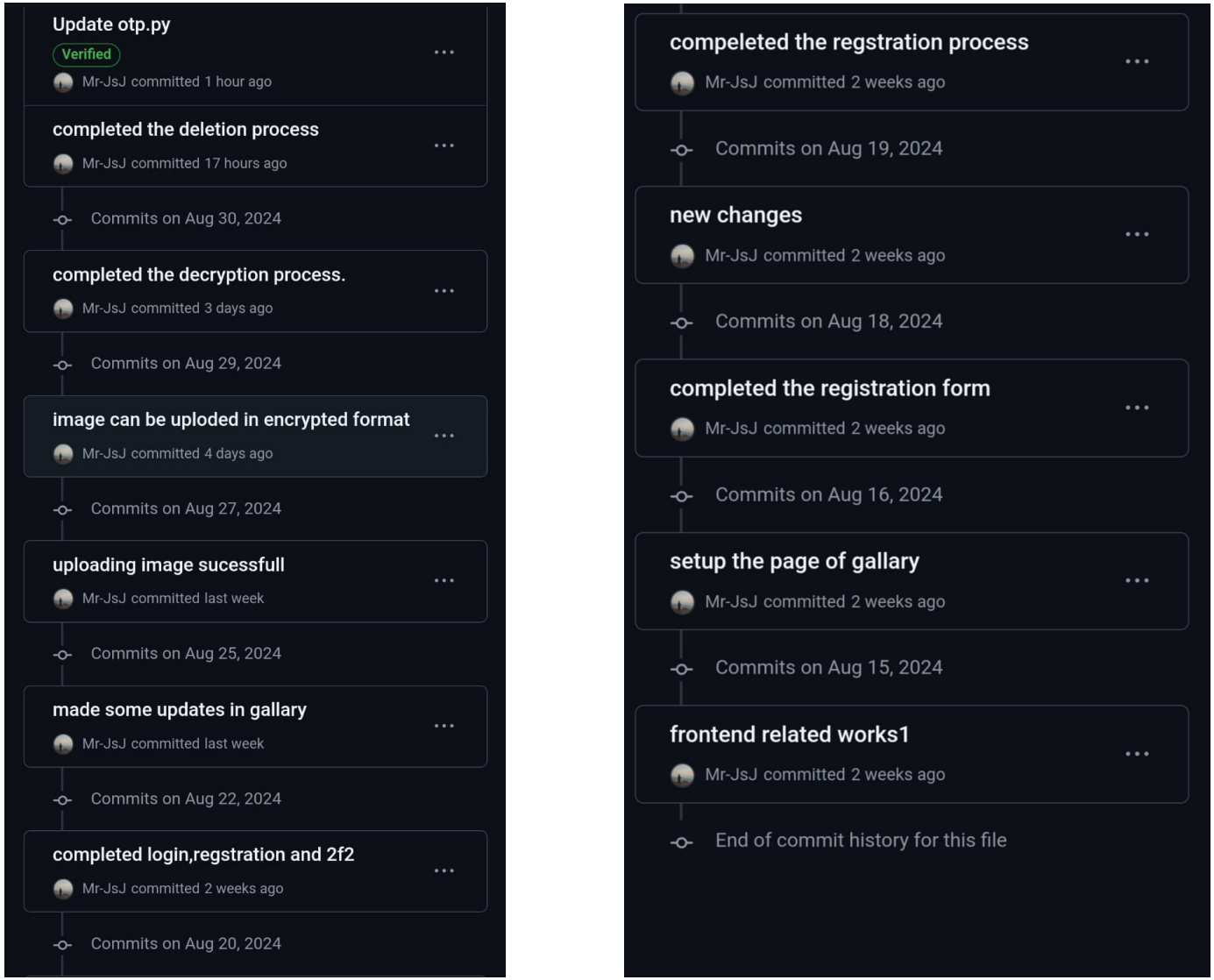



Fig.8. Git History Screenshot

# Bibliography


- Image-encryption-using-elliptic-curve-cryptography-with-data-security-IJERTV11IS050068.pdf
- Open CV documentation
- Django documentation

# Thank you!