Secure Image Encryption - Final Project Report

1. Introduction

Secure Image Encryption is a web-based application developed using Python and Flask that provides advanced encryption for digital images through a combination of pixel shuffling and color transformation. This solution ensures that both the structure and color of the image are altered based on a user-provided key, making it highly secure.

2. System Overview

The project offers:

- Direct image upload
- Secure encryption/decryption using text-based keys
- In-memory processing (no files saved)
- Direct image download after processing

Objectives:

- Implement pixel shuffling + color transformation
- Use SHA-256 for key-based seeding
- Build responsive frontend with Flask
- Avoid persistent storage for enhanced security

3. Technical Architecture

Libraries Used:

- Flask: Web application backend
- OpenCV: Image reading, resizing, and pixel-level access
- NumPy: Matrix operations and array reshaping
- hashlib: SHA-256 key hashing
- base64: In-memory image transmission
- Pillow: Image format conversion and buffer handling

Techniques:

- Pixel Shuffling: Randomizes pixel positions based on hashed key seed
- Color Transformation: Alters RGB values using key-derived values
- SHA-256 Hashing: Converts key into a strong numeric seed
- Base64 Encoding: Encodes image as string for transfer without saving

4. User Interface

Features:

- Modern CSS with glass-morphism
- Responsive layout
- Loading spinner
- Key-based encryption/decryption toggle
- Download after processing
- Input field for key
- Same UI, enhanced logic underneath

5. Working Process

1. User uploads an image and enters a key
2. Flask backend hashes the key using SHA-256
3. Image is converted to a NumPy array
4. RGB channels are separated
5. Pixel shuffling is performed using a permutation based on the key
6. Each color channel is also modified by SHA-based transformation
7. Channels are merged and image is returned via base64
8. User downloads the encrypted or decrypted result

6. Key Strengths

- Dual-level protection (structure + color)
- Strong key hashing with SHA-256
- No disk I/O for better privacy

- Access from any browser
- User-friendly UI
- Efficient and fast execution

7. Limitations & Future Scope

- Large image support limited by memory
- Currently supports JPG/PNG only
- No preview before encryption

Future Improvements:

- Dark/light mode
- WebAssembly client-side processing
- Slider comparison of encrypted vs original
- Add support for other formats (TIFF, WebP)

8. Security Visualization

- Original image: clear
- Encrypted: completely scrambled + color-shifted
- Decrypted (correct key): identical to original
- Wrong key: image remains distorted
- Uses SHA-256 to secure the key logic

9. Conclusion

The Secure Image Encryption project delivers a powerful, user-accessible tool for privacy-focused image protection. With its use of both pixel position and color transformation, it significantly improves over basic shuffling techniques, and all without requiring any image to be saved on the server.