🔒 Security Overview Document

📄 Project Title:

Secure File Sharing Web Application

👩‍💻 Author:

Tiya Mehta

📅 Date:

03 July 2025

Github:

https://github.com/Tiya0409/FUTURE_CS_03

🔷 Objective

This document provides an overview of the security mechanisms implemented in the **Secure File Sharing Web Application**.
It highlights the encryption algorithms, file security process, and key management strategy.

🔷 Encryption Methods

✅ Algorithm: AES-256

- Uses the Advanced Encryption Standard (AES) — a trusted symmetric encryption standard.
- 256-bit key ensures a high level of security.
- AES provides **confidentiality** of user data.

✅ Mode: EAX

- AES runs in **EAX mode**, which provides:
    - Confidentiality (encryption)
    - Integrity (authentication tag)
- Random **nonce** is generated for every encryption operation to enhance security.
- EAX mode prevents tampering and ensures data integrity.

🔷 How Files Are Secured

🔐 On Upload:

- User uploads a file → the app:
  - Generates a random nonce.
  - Encrypts the file using AES-256-EAX.
  - Stores the ciphertext + nonce + tag in a .enc file in the uploads/ folder.
- Original plaintext file never touches the disk.

🔐 On Download:

- User requests a file → the app:
  - Reads the encrypted .enc file.
  - Uses the nonce & tag to decrypt and verify the file's integrity.
  - Returns the decrypted file to the user.
- If integrity check fails → download is rejected.

🔷 Key Management

🔑 Key Generation:

- A secure, random 256-bit (32-byte) key is generated and stored in .env.

🔑 Key Storage:

- .env file contains the key in Base64 encoding:
- AES_KEY=*******
- .env is ignored from version control using .gitignore to prevent accidental exposure.

🔑 Best Practices:

- In production: Use a secret management service (like AWS KMS, Vault).
- Rotate the key periodically.
- Never hard-code keys into codebase.

🔷 Additional Security Measures

✅ Only encrypted files are stored.
✅ .env and uploads/ are excluded from version control.
✅ AES-EAX ensures both confidentiality and integrity.
✅ User never sees or handles the encryption key.

🔷 Recommendations for Production

- Use HTTPS for secure file transfer.
- Add user authentication & access control.
- Rotate keys periodically and re-encrypt files when keys change.
- Implement monitoring for access to the uploads folder and .env.

🔷 Summary Table

Component	Details
Encryption Algorithm	AES-256
Mode	EAX (authenticated encryption)
Key Length	256 bits (32 bytes)
Key Storage	.env file (Base64 encoded)
Integrity Protection	EAX authentication tag
Confidentiality & Integrity	Provided by AES-EAX
Key Visibility	Never exposed in code or Git

🔷 Conclusion

The Secure File Sharing Web Application applies robust encryption using AES-256-EAX mode and secure key management practices to ensure confidentiality and integrity of user data at rest.
Following recommended practices and periodic review of key management can further enhance its security in production environments.

📌 **Prepared by:**

Tiya Mehta
Internship Project — Secure File Sharing App