AES (Advanced Encryption Standard) and Post-Quantum Cryptography Working Together in Your Platform

In your platform, **AES** and **post-quantum cryptography (PQC)** are combined to provide a hybrid encryption solution, addressing both **current security threats** and **future quantum threats**. Here's a detailed breakdown of how these cryptographic systems work together:

---

1. AES (Advanced Encryption Standard)

AES is a **symmetric encryption algorithm**, meaning the same key is used for both encryption and decryption. AES is widely used today because it's fast and secure for encrypting large amounts of data like files and documents.

In your platform:

- When a user uploads a document, the system **generates an AES key** (symmetric key).
- This AES key is then used to **encrypt the document**. The encryption process involves breaking the document into blocks of data (typically 128 bits) and scrambling it using the AES key in multiple rounds of encryption.
- The **encrypted document** is then stored in the system (in the database or file system).

AES Pros in Your Platform:

- **Efficiency**: AES is fast, allowing large documents to be encrypted and decrypted quickly, which enhances performance.
- **Security**: AES-256, for example, is considered extremely secure against classical (non-quantum) computers.

However, AES is **vulnerable to quantum attacks** in the long run. This is where **PQC** comes in.

---

2. Post-Quantum Cryptography (PQC)

Post-quantum cryptography refers to cryptographic algorithms that are resistant to attacks from **quantum computers**. These algorithms are based on hard mathematical problems that even quantum computers cannot solve efficiently.

In your platform, PQC (e.g., **Kyber**) is used to **secure the AES key**.

How it works:

- After the document is encrypted using AES, the system still needs to securely share or store the AES key used in the encryption.

- To protect this AES key from quantum computers, you use a **PQC algorithm** (like Kyber) to **encrypt the AES key**.
- The PQC-encrypted AES key is then stored along with the AES-encrypted document in the system.

PQC Pros in Your Platform:

- **Quantum Resistance**: Algorithms like Kyber are designed to be resistant to quantum computers, ensuring that the AES key remains secure even in the future when quantum computers become powerful enough to break classical encryption algorithms like RSA.
- **Long-term Security**: While AES can secure the document now, PQC ensures that even in a quantum future, the AES key is still safe from attacks.

Step-by-Step Process:

Encryption Process (When a Document is Uploaded):

1. **User Uploads Document**:
    - A user uploads a document through the platform's front-end (Next.js). This request is sent to the back-end (Django).
2. **Generate AES Key**:
    - The Django back-end generates a random **AES key** for encrypting the document. The key length is typically 256 bits for maximum security (AES-256).
3. **Encrypt the Document with AES**:
    - The document is encrypted using the AES key. The original document is transformed into an encrypted version that can only be decrypted by the same AES key.
4. **Encrypt the AES Key with Post-Quantum Cryptography**:
    - The AES key itself is then **encrypted** using a PQC algorithm like **Kyber**. This ensures that even if quantum computers become capable of breaking AES key transmission methods, they will not be able to decrypt the AES key.
5. **Store Encrypted Data**:
    - The **AES-encrypted document** and the **PQC-encrypted AES key** are both stored securely in the database or file storage.

Decryption Process (When a Document is Downloaded):

1. **User Requests to Download Document**:
    - A user wants to download the document. They make a request through the Next.js front-end to the Django back-end.
2. **Retrieve Encrypted Data**:
    - Django retrieves the **AES-encrypted document** and the **PQC-encrypted AES key** from the database.
3. **Decrypt AES Key**:

    - The back-end uses the **PQC algorithm (Kyber)** to **decrypt the AES key** that was previously secured with PQC encryption.
4. **Decrypt the Document with AES**:
    - The decrypted AES key is then used to **decrypt the document**, turning the encrypted file back into its original readable form.
5. **Serve Decrypted Document**:
    - The decrypted document is sent back to the user through the Next.js front-end.

How AES and PQC Ensure Document Security:

- **AES for Fast and Efficient Document Encryption**: AES is incredibly fast and secure for encrypting the actual document content. However, if a quantum computer were available, it could break traditional methods of key exchange (e.g., RSA or ECC), making it possible to recover the AES key and decrypt the document.
- **PQC for Future-Proof Security**: To ensure long-term security, the AES key is protected by post-quantum cryptography. Quantum computers would not be able to break PQC algorithms, meaning the AES key remains secure even if quantum computers advance. This **combination** ensures that your document-sharing platform is secure both now and in the future.

Key Technologies and Algorithms in Your Platform:

- **AES (Advanced Encryption Standard)**:
    - **Algorithm Type**: Symmetric Encryption (uses one key for encryption and decryption).
    - **Use Case**: Encrypting the document for fast and efficient encryption.
    - **Strength**: Secure against classical attacks.
- **Kyber (Post-Quantum Cryptography Algorithm)**:
    - **Algorithm Type**: Public-Key Encryption.
    - **Use Case**: Encrypting the AES key to resist quantum computer attacks.
    - **Strength**: Resistant to attacks from quantum computers, ensuring long-term security.

Summary:

- **AES** ensures the fast encryption of documents, making it suitable for large files.
- **Post-Quantum Cryptography (Kyber)** ensures the security of the AES key against future quantum attacks.

- Together, they provide a **hybrid encryption solution** that addresses both current and future security needs, keeping your document-sharing platform secure from today's classical attacks and tomorrow's quantum threats.

This setup offers **optimal security** by balancing **performance (AES)** and **future-proof quantum resistance (PQC)**