Objective:

Build a simple file upload service using **Node.js** or **Express.js** that allows users to upload files, store them on the server, and save file metadata in a database.

Requirements:

1. File Upload API

- **Goal**: Implement an API endpoint to handle file uploads.
- **Instructions**:
  - Use **Multer** (or another file-handling middleware) to manage file uploads in **Express.js**.
  - Accept common file types like images (`JPEG`, `PNG`), PDFs, and others as needed.
  - Restrict the maximum file size to **5MB**.

2. Metadata Storage

- **Goal**: Store file-related metadata in a database (e.g., **MongoDB**, **PostgreSQL**, etc.).
- **Metadata to Store**:
  - **Filename**.
  - **File size**.
  - **File type** (MIME type).
  - **Upload timestamp**.
  - **Uploader information** (only if authentication is implemented).
- **Database**: Use a database like **MongoDB** (with **Mongoose** for ODM) or **PostgreSQL** to store metadata.

3. Security

- **File Type Restrictions**:
  - Only allow specific file types such as images (JPEG, PNG) and PDFs.
- **Error Handling**:
  - Gracefully handle errors like unsupported file types, oversized files, etc.

4. Download API

- **Goal**: Provide an API endpoint to download uploaded files.
- **Instructions**:
  - Create an endpoint that allows users to download files using their unique ID or filename.
  - Ensure the proper headers, such as `Content-Disposition`, are included so the file is downloaded with its original filename.

5. File Storage

- **Goal**: Store files either on the local filesystem or in the cloud.
- **Instructions**:
    - By default, files should be stored locally on the server's filesystem.
    - Optionally, implement cloud storage using services like **AWS S3** or **Google Cloud Storage** for storing files.

6. File Deletion API

- **Goal**: Provide an API to delete uploaded files.
- **Instructions**:
    - Implement an API endpoint to delete files from both the server and the database (i.e., delete the file and its associated metadata).