### ✅ **Technologies & Features You’ve Implemented**

#### 🔐 **User Authentication & Authorization**

* JWT-based login with token generation (jwtBearer, JwtToken)
* Password hashing using bcrypt
* Role-based access (ADMIN, USER) defined using an Enum
* Face authentication on login using face\_recognition library
* Image capture and storage via capture\_image function
* Image storage in MongoDB using GridFS

#### 🧍 **User Routes (**user\_router.py**)**

* POST /signup:
  + Validates input
  + Captures user image
  + Hashes password
  + Stores user with role and image in MongoDB
* POST /login:
  + Validates credentials
  + Captures live image and compares with stored one
  + Generates JWT token if matched
* PATCH /update-password:
  + Validates old password
  + Updates to new hashed password

### 🗂️ **Folder Management (**folder\_router.py**)**

* POST /folders:
  + Creates a folder with name and parent folder ID
* DELETE /folders/{id}:
  + Only admins can delete folders
* PUT /update-folder-name/{id}:
  + Update folder name with validation
* GET /folders/{id}:
  + Lists folder contents (subfolders & files)

### 📁 **File Management (**files\_router.py**)**

* File types: JPEG, PNG, PDF only
* Stored using GridFS
* File metadata saved in files\_metadata collection
* POST /upload:
  + Upload file to folder
  + Checks for duplicate names
* DELETE /delete-file/{id}:
  + Admin-only deletion
* PUT /update-file-name/{id}:
  + Update name with validation
* GET /files/download-by-id/{id}:
  + Downloads file with StreamingResponse

### 📦 **Backend Architecture**

* FastAPI for backend APIs
* MongoDB with pymongo + GridFS for file/image storage
* JWT tokens for secured routes and user sessions
* Folder-File hierarchy: Folders store child folders and files using parent\_folder\_id & folder\_id
* Face recognition authentication adds a biometric layer

### ▶️ **How to Run This Project Locally**

#### 🛠️ 1. **Requirements**

Make sure you have:

* Python 3.10+
* MongoDB running locally or MongoDB Atlas URI
* Webcam access (for face\_recognition)
* Install dependencies:

pip install fastapi uvicorn pymongo bcrypt python-multipart face\_recognition python-jose[cryptography] gridfs

**NOTE – No need of creating any database automatically when we run the project the databases and tables are created.**

#### 📁 2. **Project Structure (Example)**

project/

│

├── main.py

├── router/

│ ├── user\_router.py

│ ├── files\_router.py

│ └── folder\_router.py

├── configuration/

│ └── database.py

├── schema/

│ ├── user\_dto.py

│ └── folder\_dto.py

├── security/

│ ├── jwtToken.py

│ └── jwtConfig.py

├── utility/

│ └── common.py # (capture\_image function)

#### ⚙️ 3. **Run FastAPI Server**

uvicorn main:app --reload

#### 🌐 4. **Test with Swagger UI**

Navigate to:  
<http://127.0.0.1:8000/docs>