Propylon Document Manager Assessment

1. Project Overview

Purpose

This document management system was developed as part of the Propylon Engineering Assessment. It allows authenticated users to store, version, and retrieve files using virtual paths or content-addressable storage hashes.

Key Features

- Upload any file type
- Define virtual **url_path** on upload
- Automatic file versioning
- Retrieve via **url_path** + revision or **content_hash**
- Secure, token-based access
- Dynamic frontend (React)

2. System Architecture

Backend

- **Framework**: Django 5, Django REST Framework
- **Storage**: Files saved locally using **FileSystemStorage**
- **Authentication**: Token-based (**rest_framework.auth_token**)
- **Content Addressing**: SHA-256 computed and stored on each file upload
- **Key Models**:
    - **FileVersion**: represents each uploaded file version
    - **User**: custom user model with email-based login

Frontend

- **Framework**: React (create-react-app)
- **Key components**:
    - **FileVersions.js**: handles file listing/upload/download
    - **Login.js** & **Register.js**: for authentication
- **Token management**: Stored in **localStorage**

3. Configuration & Setup

Local Setup Instructions

1. Set up virtual environment

python -m venv venv

source venv/bin/activate *# or venv\Scripts\activate on Windows*

2. Install backend dependencies

pip install -r requirements/dev.txt

3. Run migrations

python manage.py makemigrations

python manage.py migrate

4. Create a superuser

python manage.py createsuperuser

5. Start the server

python manage.py runserver

Django will run at: ***http://localhost:8000/***

Frontend Setup

cd client/doc-manager

npm install

npm start

React app runs by default on ***http://localhost:3000/***.

Authentication

- Login: **POST /auth-token/** → returns token
- Use token in all API requests:
- Authorization: Token <token>

4. API Overview

Authentication

Endpoint	Method	Description
/auth-token/	POST	Login (with email/password)
/api/register/	POST	Register a new user

File Upload & Listing

Endpoint	Method	Description
/api/upload/	POST	Upload a new file version
/api/upload/	GET	List all uploaded versions (authenticated user only)

POST body (multipart/form-data):

Key	Example
file	**output.pdf**
file_name	**output.pdf**
url_path	**/documents/internal/output.pdf**

File Retrieval

Method	Endpoint Example
GET	**/api/documents/<path>?revision=n** → (Path + Version, internal only)
GET	**/api/cas/<content_hash>**

5. Frontend Functional Flow

Authentication:

- User lands on **/**
- If not logged in:
    - Shown login screen (**POST /auth-token/**)
    - Option to register (**POST /api/register/**)

File Upload:

- Fill form:
    - **file_name**
    - **url_path** (virtual path)
    - **file** upload field
- Submit triggers upload with auto-versioning

File Retrieval:

- Dropdown 1: Select **url_path**
- Dropdown 2: Select **version_number**
- Frontend finds matching **content_hash**
- Uses: **GET /api/cas/<hash>** to download

7. Testing Strategy

Unit Tests

Run: pytest

Tests include:

- File upload
- Duplicate version handling
- File list access
- Auth protection validation