## **Project Overview: Django Authentication API**

## This Django-based backend project includes a modular app called LoginUtility that handles user registration and login functionality. The authentication system is implemented using JWT (JSON Web Token) via the djangorestframework-simplejwt package.

## **App: LoginUtility**

## **1. User Registration API**

## - Allows new users to register. - Accepts fields like: • username • email • password • first\_name • role - Data validation and user creation are handled through a custom serializer.

## **2. User Login API**

## - Authenticates users using the provided username and password. - If credentials are valid, it returns: • Access token (expires in 3 minutes) • Refresh token (used to get a new access token)

## **3. JWT Authentication**

## - Upon successful login, the frontend (e.g., Angular) will: • Store the access token and refresh token • Send the access token in the Authorization header for each API request: Authorization: Bearer <access\_token> - Token-based authentication is stateless and scalable.

## **4. Token Expiry**

## - Access Token: 3 minutes

## **Key Technologies Used**

## - Django - Django REST Framework - djangorestframework-simplejwt - Angular (Frontend) - JWT Authentication

## **Test Coverage**

## - User registration success and failure cases - Login success and failure cases - Token retrieval and validation - Authenticated and unauthenticated user info access

## **Media App**

## The **Media** app is responsible for handling file uploads via POST requests. When a file (e.g., image) is uploaded:

## The API receives the file and extracts the filename.

## The file is uploaded to the **Cloudinary** cloud platform for storage.

## The app then stores relevant details in the database, including:

## The file URL returned by Cloudinary

## The user ID who uploaded the file

## The user’s role

## Created date and other metadata