# User Authentication System (UAS)

This is a comprehensive User Authentication System built with Django REST Framework that provides robust user management, role-based access control, and activity monitoring capabilities.

## Features

- \*\*User Authentication\*\*

- Custom User Model with email & username authentication

- Multi-Factor Authentication (MFA) support

- Password reset functionality

- User account locking mechanism

- JWT-based authentication using Simple JWT

- \*\*Role-Based Access Control (RBAC)\*\*

- Dynamic role management

- Permission-based access control

- View-level access control

- Role assignment to users

- \*\*Admin Dashboard\*\*

- User activity monitoring

- User management interface

- Role and permission management

- View access management

- \*\*Security Features\*\*

- Token blacklisting

- MFA support

- Account locking

- Activity logging

- IP tracking

## Project Structure

```

UAS/

├── accounts/ # Core authentication app

├── admin\_dashboard/ # Admin management interface

├── static/ # Static files (CSS, JS)

├── templates/ # HTML templates

├── utils/ # Utility functions

└── UAS/ # Project settings

```

### Key Components

1. \*\*accounts/\*\*

- Handles user authentication and management

- Custom User model with extended features

- OTP and MFA functionality

- Role and permission models

2. \*\*admin\_dashboard/\*\*

- User activity monitoring

- View access management

- Administrative functions

3. \*\*templates/\*\*

- Login/Register pages

- Password reset templates

- Admin panel templates

## Prerequisites

- Python 3.11+

- Django 5.1+

- Django REST Framework

- Additional requirements in requirements.txt

## Configuration

Key settings in `settings.py`:

- JWT configuration

- Authentication backends

- REST Framework settings

## API Endpoints

### Authentication

- POST /api/auth/register/ - User registration

- POST /api/auth/login/ - User login

- POST /api/auth/verify-otp/ - OTP verification

- POST /api/auth/reset-password/ - Password reset

- POST /api/auth/request-reset-password/ - Request password reset

### User Management

- GET /api/users/ - List users

- POST /api/users/ - Create user

- PUT /api/users/{id}/ - Update user

- DELETE /api/users/{id}/ - Delete user

### Role Management

- GET /api/roles/ - List roles

- POST /api/roles/ - Create role

- PUT /api/roles/{id}/ - Update role

- DELETE /api/roles/{id}/ - Delete role

## Security Considerations

1. \*\*Environment Variables\*\*

- Move sensitive data to environment variables

- Update SECRET\_KEY in production

- Configure allowed hosts

2. \*\*Production Settings\*\*

- Set DEBUG = False

- Configure HTTPS

- Set up proper CORS settings

3. \*\*Authentication Security\*\*

- Configure password validators

- Set up rate limiting

- Enable SSL/TLS

## Logging

- User activities are logged in `logs/user\_activity.log`

- Activities include:

- Login attempts

- Password changes

- Role modifications

- Access attempts