2023

Django

User Authentication

Amin Jandaghian

Let’s not make a mountain out of a molehill in understanding and coding authentication in django. **Authentication**, as we all know, is a basic necessity required for the users’ to pass through, before they encounter the actual treasure or the application in general. It’s a protective gear to be worn by an application to ensure secure access to it. Here, I’m going to walk you through 5 simple steps that shall help you in integrating authentication with your django application.

To start a django project -

```
django-admin startproject auth_django .if windows,
django-admin.exe startproject auth_django .
```

Step 1: Handling settings in settings.py

Assuming we've a project set up, let's quickly define a few authentication URLs in the *settings.py* file in the project (*auth_django*) folder (an app isn't being considered due to the simplicity of the application).

```
# Login & Logout URLs
LOGIN_URL = '/login/'
LOGIN_REDIRECT_URL = '/home/'
LOGOUT_REDIRECT_URL = '/login/'
```

Also, initialize the path to *templates/* folder in *'DIRS'*.

```
TEMPLATES = [
{
  'DIRS': [os.path.join(BASE_DIR, 'templates/')],
  ...
]
```

Note: Create a folder named *templates/* in parent *auth_django/* directory.

Step 2: Defining Views in views.py

Django Views is where the logic resides. Create views.py file in *auth_django/auth_django/* path. We define two views — one to help the user register, and the other, to display the home page. Login and Logout views are taken from the predefined auth_views (to be seen in Step 3).

The *home* view renders a success page. *@login_required* is to make sure that home isn't accessible to unauthenticated users'. The *register* view initially displays a *UserCreationForm* that prompts the user for username, password and confirm password. When a user submits the form, the details are captured and validated. If validation is successful, details are saved in the default database (SQLite) and the user is taken to the home page via authentication.

Step 3: Initializing URLs in urls.py

Authentication URLs are all associated with views in *urls.py*. *home* and *register* URLs are mapped with the views in *views.py*. *login* and *logout* are mapped with the inbuilt *auth_views*. *as_view()* is a method in the class *LoginView/LogoutView* which returns a callable view.

Step 4: Creating Templates

Users understand the application through templates. *login* view by default uses *registration/login.html in templates/* folder. Thus, we define all the templates in *registration/* folder.

1. base.html — Encloses external libraries
2. login.html — Login Form
3. register.html — Registration Form
4. success.html — Home page

Project Directory Structure

```
auth_django
|-- auth_django
    |-- pycache/
    |-- __init__.py
    |-- asgi.py
    |-- settings.py
    |-- urls.py
    |-- views.py
    |-- wsgi.py
|-- templates
    |-- registration
        |-- base.html
        |-- login.html
        |-- register.html
        |-- success.html
|-- db.sqlite3
|-- manage.py
```

Step 5: Code Harvest!

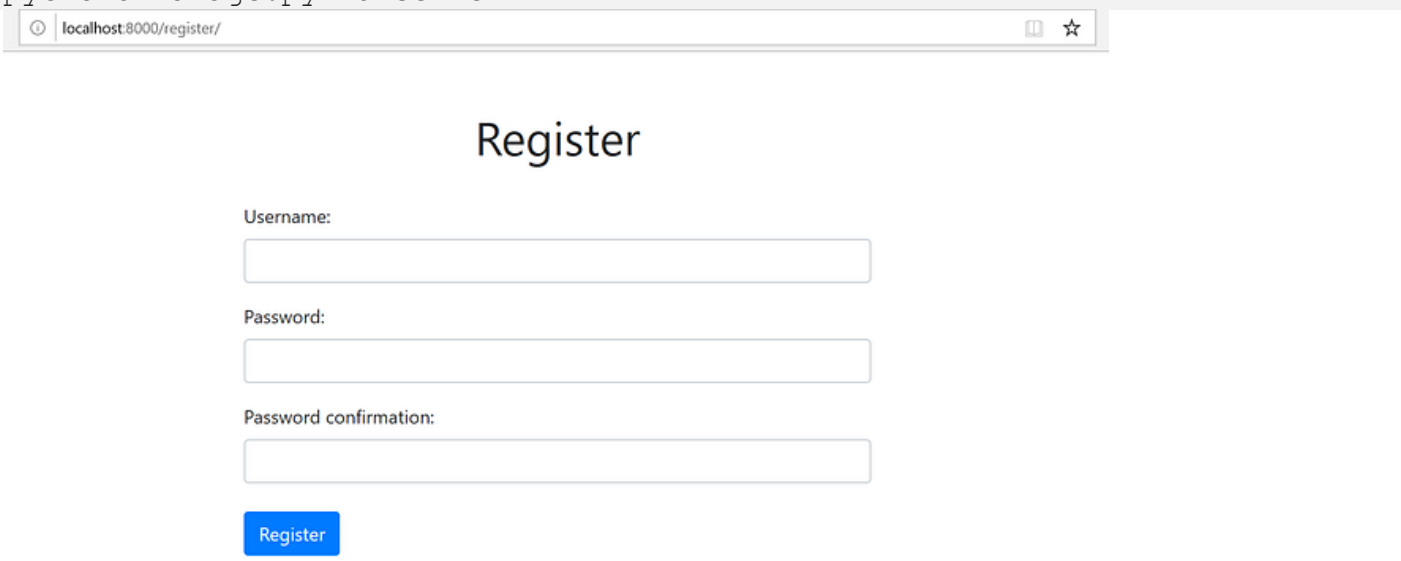
Handy commands:

To migrate the database -

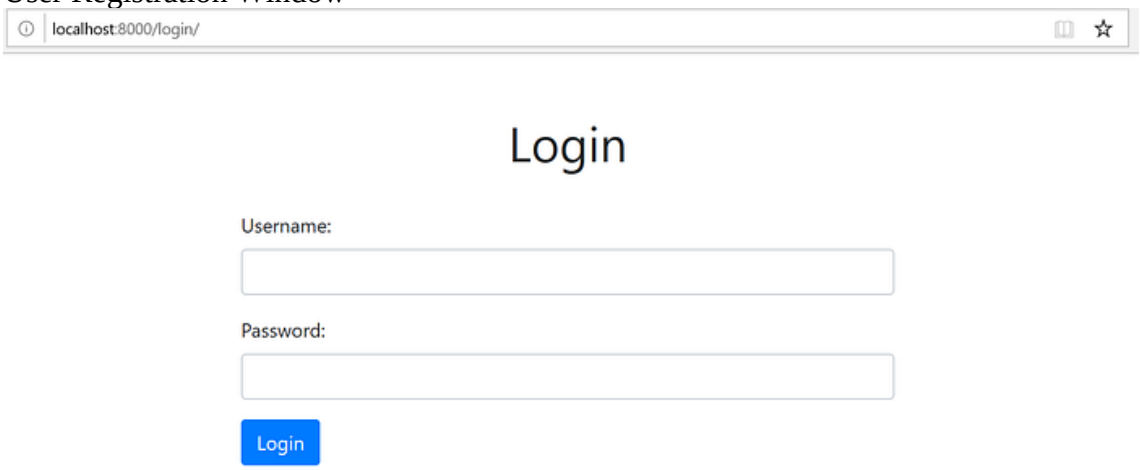
```
python3 manage.py migrate
```

To run server -

```
python3 manage.py runserver
```



User Registration Window



User Login Window