**User Authentication System Using Django Admin Application**

[[Kazi Mushfiqur Rahman](https://medium.com/@KaziMushfiq1234?source=post_page-----e164f4fd264c--------------------------------)](https://medium.com/@KaziMushfiq1234?source=post_page-----e164f4fd264c--------------------------------)

[Kazi Mushfiqur Rahman](https://medium.com/@KaziMushfiq1234?source=post_page-----e164f4fd264c--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F16ea6542ad64&operation=register&redirect=https%3A%2F%2Fmedium.com%2F%40KaziMushfiq1234%2Fuser-authentication-system-using-django-admin-application-e164f4fd264c&user=Kazi+Mushfiqur+Rahman&userId=16ea6542ad64&source=post_page-16ea6542ad64----e164f4fd264c---------------------post_header-----------)

4 min read

·

1 day ago



Photo by [NordWood Themes](https://unsplash.com/@nordwood?utm_source=medium&utm_medium=referral" \t "_blank) on [Unsplash](https://unsplash.com/?utm_source=medium&utm_medium=referral" \t "_blank)

**User authentication**

Django Authentication System: Django comes with a built-in user authentication system that handles user-related functionalities such as user accounts, groups, permissions, and cookie-based user sessions. This entire package of authentication and authorization is commonly referred to as the “authentication system” in Django.

Default Configuration: When you start a new Django project using django-admin startproject, some default configuration is automatically set up in your settings.py file.

* INSTALLED\_APPS:
* django.contrib.auth: This includes the core of the authentication framework, providing default models for user authentication.
* django.contrib.contenttypes: This part of the Django content type system allows you to associate permissions with the models you create in your project.
* MIDDLEWARE:
* SessionMiddleware: Manages sessions across requests. Sessions allow you to persist data across multiple requests from the same user.
* AuthenticationMiddleware: Associates users with requests using sessions. This middleware ties authenticated users to their corresponding sessions

In simpler terms:

* The django.contrib.auth app provides the fundamental components for user authentication (like login, logout, and user management).
* The django.contrib.contenttypes app allows you to associate permissions with your own models.
* The SessionMiddleware helps in managing user sessions, which are crucial for maintaining state across multiple requests.
* The AuthenticationMiddleware connects authenticated users to their respective sessions, ensuring that the system knows who is making a request.

Together, these components make up the Django authentication system, providing a secure and customizable way to handle user authentication and authorization in your web applications.

**Django.contrib.auth**

In your Django project’s models.py file, you might have a class called User. This class is a fundamental part of the authentication system in Django.

1. Purpose of the User Class: The User class represents individuals interacting with your website. It serves as a way to manage and organize information about users, allowing you to perform various tasks such as restricting access, registering user profiles, and associating content with creators.
2. Single User Class: In Django’s authentication framework, there’s only one class of user. Whether someone is a superuser (admin) or a staff user, they are all instances of the same User class. The difference lies in the attributes or properties set for these user objects, not in having distinct classes.
3. Special Attributes for Different Users: Superusers (admin) and staff users are essentially instances of the User class, but they have special attributes set. These attributes grant them specific privileges or roles within the system. For example:

* Superusers have elevated permissions, often allowing them to manage all aspects of the application.
* Staff users may have additional permissions compared to regular users.

In simpler terms:

* The User class is like a blueprint for representing people on your website.
* All users, whether they are regular users, superusers (admin), or staff, are instances of this single User class.
* The differences between these users lie in the attributes or properties assigned to them, determining their roles and permissions within the application.

Understanding and managing the User class is essential for implementing authentication features and controlling access in your Django project.

**Show the location of auth model in Django GitHub**

**[django/django/contrib/auth/models.py at main · django/django](https://github.com/django/django/blob/main/django/contrib/auth/models.py?source=post_page-----e164f4fd264c--------------------------------" \t "_blank)**

[The Web framework for perfectionists with deadlines. - django/django/contrib/auth/models.py at main · django/django](https://github.com/django/django/blob/main/django/contrib/auth/models.py?source=post_page-----e164f4fd264c--------------------------------" \t "_blank)

[github.com](https://github.com/django/django/blob/main/django/contrib/auth/models.py?source=post_page-----e164f4fd264c--------------------------------" \t "_blank)

**Create virtual environment**

python -m venv myenv

**Activate virtual environment**

myenv\Scripts\activate

Install Django

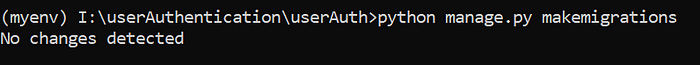
pip install django

Start a Django project

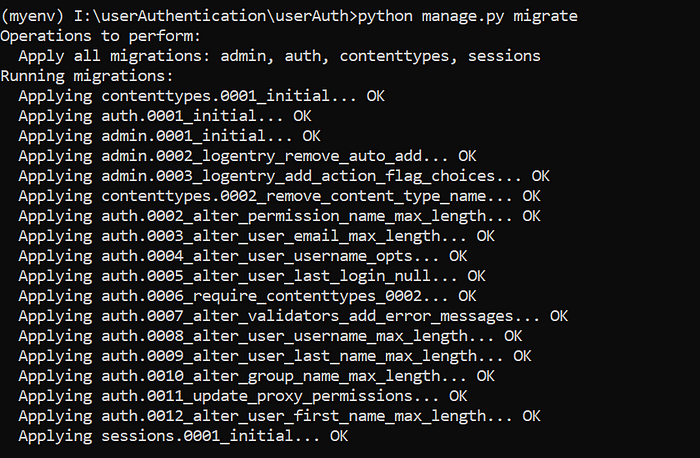
django-admin startproject uaerAuth

Perform migration

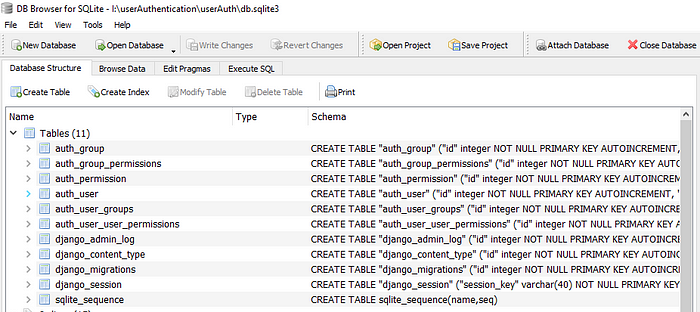
python manage.py makemigrations



python manage.py migrate

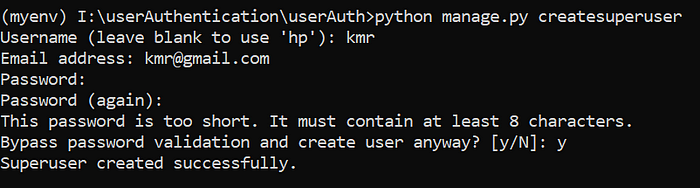


Show the created table in DB Browser for sqlite

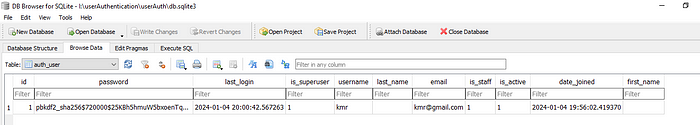


Create super user

python manage.py createsuperuser

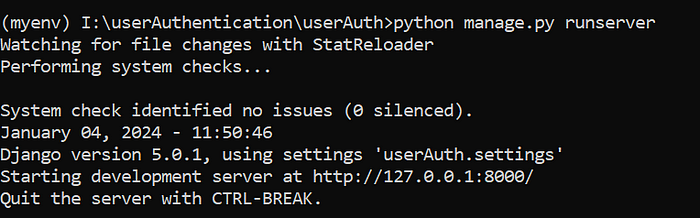


created user is stored in the table auth



Run server

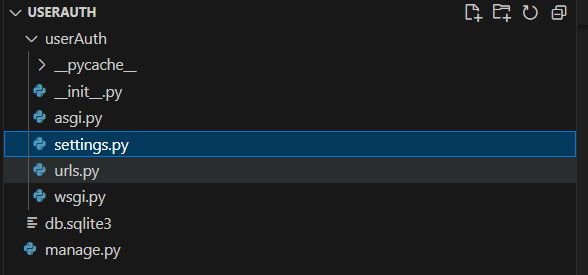
python manage.py runserver



Copy the url and paste it to the browser

http://127.0.0.1:8000/admin

Project directory



***userAuth/settings.py***

# Application definition  
  
INSTALLED\_APPS = [  
 'django.contrib.admin',  
 'django.contrib.auth',  
 'django.contrib.contenttypes',  
 'django.contrib.sessions',  
 'django.contrib.messages',  
 'django.contrib.staticfiles',  
]

Here, 2nd and 3rd line of codes are responsible for performing authentication

***userAuth/settings.py***

MIDDLEWARE = [  
 'django.middleware.security.SecurityMiddleware',  
 'django.contrib.sessions.middleware.SessionMiddleware',  
 'django.middleware.common.CommonMiddleware',  
 'django.middleware.csrf.CsrfViewMiddleware',  
 'django.contrib.auth.middleware.AuthenticationMiddleware',  
 'django.contrib.messages.middleware.MessageMiddleware',  
 'django.middleware.clickjacking.XFrameOptionsMiddleware',  
]

Here, 5th and 6th lines of codes are responsible for performing authentication