**Building a Real-Time Chat Application with Django, Channels, and User Authentication**

[[Chetan Sureka](https://chetansureka.medium.com/?source=post_page-----df9fb59a40c8--------------------------------)](https://chetansureka.medium.com/?source=post_page-----df9fb59a40c8--------------------------------)

[[AWS Tip](https://awstip.com/?source=post_page-----df9fb59a40c8--------------------------------)](https://awstip.com/?source=post_page-----df9fb59a40c8--------------------------------)

[Chetan Sureka](https://chetansureka.medium.com/?source=post_page-----df9fb59a40c8--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F1b42b816ac66&operation=register&redirect=https%3A%2F%2Fawstip.com%2Fbuilding-a-real-time-chat-application-with-django-channels-and-user-authentication-df9fb59a40c8&user=Chetan+Sureka&userId=1b42b816ac66&source=post_page-1b42b816ac66----df9fb59a40c8---------------------post_header-----------)

Published in

[AWS Tip](https://awstip.com/?source=post_page-----df9fb59a40c8--------------------------------)

·

2 min read

·

Dec 31, 2023

5

1

In this tutorial, we will enhance our real-time chat application by incorporating Django’s built-in authentication system. This will add a layer of security and enable personalized interactions within the chat. We will use Django, Channels, and Web Socket for real-time communication.



Photo by [Caspar Camille Rubin](https://unsplash.com/@casparrubin?utm_source=medium&utm_medium=referral) on [Unsplash](https://unsplash.com/?utm_source=medium&utm_medium=referral" \t "_blank)

**Prerequisites**

Before we begin, ensure you have the following installed:

* Python (3.6 or later)
* Django
* Channels

Install Channels using:

pip install channels

**Setting Up a Django Project**

Create a new Django project and an app within it:

django-admin startproject chat\_project  
cd chat\_project  
python manage.py startapp chat

Include the necessary apps in your project’s settings:

# chat\_project/settings.py  
INSTALLED\_APPS = [  
'channels',  
'chat',  
'django.contrib.auth',  
'django.contrib.contenttypes',  
#...  
]

**Creating a Model for Messages with User Authentication**

Extend the `Message` model in `chat/models.py` to include a foreign key to the user model:

# chat/models.py  
from django.db import models  
from django.contrib.auth.models import User  
class Message(models.Model):  
 author = models.ForeignKey(User, on\_delete=models.CASCADE)  
 content = models.TextField()  
 timestamp = models.DateTimeField(auto\_now\_add=True)  
 def \_\_str\_\_(self):  
 return f'{self.author.username}: {self.content}'

Run migrations to apply the changes:

python manage.py makemigrations  
python manage.py migrate

**Implementing Web Socket with Channels for User Authentication**

Update your consumer in `chat/consumers.py` to handle authenticated users:

# chat/consumers.py  
import json  
from channels.generic.websocket import AsyncWebsocketConsumer  
from channels.db import database\_sync\_to\_async  
  
class ChatConsumer(AsyncWebsocketConsumer):  
 async def connect(self):  
 await self.accept()  
  
 async def disconnect(self, close\_code):  
 pass  
  
 async def receive(self, text\_data):  
 data = json.loads(text\_data)  
 message = data['message']  
  
 user = self.scope['user']  
 if user.is\_authenticated:  
 await self.save\_message(user, message)  
 await self.send\_chat\_message(user.username, message)  
  
 @database\_sync\_to\_async  
 def save\_message(self, user, message):  
 Message.objects.create(author=user, content=message)  
  
 async def send\_chat\_message(self, username, message):  
 await self.send(text\_data=json.dumps({  
 'message': f'{username}: {message}'  
 }))

**Updating the HTML Template for User Authentication**

Modify your HTML template in `chat/templates/chat/index.html` to include user authentication and display the username:

<!-- chat/templates/chat/index.html -->  
<!DOCTYPE html>  
<html>  
<head>  
 <title>Real-Time Chat</title>  
</head>  
<body>  
 {% if user.is\_authenticated %}  
 <p>Logged in as: {{ user.username }}</p>  
 <div id="chat">  
 <input type="text" id="messageInput" placeholder="Type your message..." />  
 <button onclick="sendMessage()">Send</button>  
 <ul id="chatMessages"></ul>  
 </div>  
 <script>  
 const chatSocket = new WebSocket('ws://' + window.location.host + '/ws/chat/');  
  
 chatSocket.onmessage = function(event) {  
 const message = JSON.parse(event.data).message;  
 const ul = document.getElementById('chatMessages');  
 const li = document.createElement('li');  
 li.appendChild(document.createTextNode(message));  
 ul.appendChild(li);  
 };  
  
 function sendMessage() {  
 const messageInput = document.getElementById('messageInput');  
 const message = messageInput.value;  
 chatSocket.send(JSON.stringify({'message': message}));  
 messageInput.value = '';  
 }  
 </script>  
 {% else %}  
 <p>Please log in to use the chat.</p>  
 {% endif %}  
</body>  
</html>

**Updating `views.py` for User Authentication**

Ensure that your views are rendering the template with the correct context, including the `user`:

# chat/views.py  
from django.shortcuts import render  
  
def index(request):  
 return render(request, 'chat/index.html', {'user': request.user})

After implementing these changes, your real-time chat application now requires users to log in before participating in the chat. This adds a layer of security and personalization to your real-time chat functionality. Explore further by adding features like message persistence, user authentication handling, and a more polished user interface to enhance your real-time chat experience.

[Django](https://medium.com/tag/django?source=post_page-----df9fb59a40c8---------------django-----------------)

[Chat App Development](https://medium.com/tag/chat-app-development?source=post_page-----df9fb59a40c8---------------chat_app_development-----------------)

[Python](https://medium.com/tag/python?source=post_page-----df9fb59a40c8---------------python-----------------)

[Web Development](https://medium.com/tag/web-development?source=post_page-----df9fb59a40c8---------------web_development-----------------)

[Tutorial](https://medium.com/tag/tutorial?source=post_page-----df9fb59a40c8---------------tutorial-----------------)