Guide for future devs

In this file let’s talk about important parts of the project:

```python
from pymongo import MongoClient
import certifi
def get_mongo_collection():
    client = MongoClient('mongodb+srv://dbakalov:GW40QU
    db = client['Chatbot']
    collection = db['Accounts']
    return collection
```

This python code just returns the collection of accounts, so you can manipulate the accounts of the users

```python
from django.shortcuts import render
def add_user(request,insert_data, collection):
    collection.insert_one({"first_name":insert_data[0], "last_name":insert_data[1], "email":insert_data[2], "selected_role":insert_data[3
    return render(request,'success.html', {'email':insert_data[2]})
```

This code creates new object in accounts based on the insert_data parameter

```python
from django.shortcuts import render
def login_user(request,insert_data, collection):
    result=collection.find_one({"email":insert_data[0], "password":insert_data[1]})
    if result:
        return render(request,'success.html', {'email':insert_data[0]})
    else:
        return render(request,'Loginfail.html')
```

This function serves for logging the user , if it can’t find user in db, it just returns fail page

```python
from django.shortcuts import render
def display_all(request, collection,email):
    query = {"email": {"$ne": email}}
    documents = collection.find(query)
    documents_list = [doc for doc in documents]
    return render(request, 'chatList.html', {'items': documents_list, 'email':email})
```

This code is responsible for getting all available for chatting users except the user who accesses the chat, so you couldn’t chat with yourself

```python
from django.shortcuts import render
def display_user(request, collection,email):
    query = {"email":  email}
    document = collection.find_one(query)
    return render(request, 'UserProfile.html', {'item': document, 'email': email})
```

This code just gets user based on his email and and gives all the data about him

```python
def update_user(request,data, collection,first_email):
    search={'email':first_email}
    updated = {"$set": data}
    collection.update_one(search, updated)
    return render(request, 'UpdateSuccess.html')
```

This function updates the user based on the data parameter

```python
class YourConsumer(AsyncConsumer):
    async def websocket_connect(self, event):
        await self.send({"type": "websocket.accept"})
        await self.send({
            "type": "websocket.send",
            "text": "You have joined the chat."
        })

    async def websocket_receive(self, text_data):
        data = json.loads(text_data['text'])
        message = data['text']
        room = data['room']
        await self.send_group(room, message)

    async def websocket_disconnect(self, event):
        user = self.scope['user']
        room_name = f"chat_{user.id}"
        await self.channel_layer.group_discard(room_name, self.channel_name)



    async def chat_message(self, event):
        print(event)
        message = event['text']
        print(message)
        await self.send({
            "type": "websocket.send",
            "text": message
        })


    async def send_group(self, group_name, message):
        await self.channel_layer.group_add(group_name, self.channel_name)
        await self.channel_layer.group_send(
```

Here is the consumer for this project .
As we can see it when it receives text it finds message and room info to broadcast it to other users

```
const user = document.getElementById('name').textContent;
const friend = document.getElementById('friend').textContent;
const socket = new WebSocket("ws://127.0.0.1:8000/ws");
socket.onopen = (event) => {
    console.log("WebSocket connection opened:", event);
};

socket.onmessage = (event) => {
    console.log('event', event)
    const message = event.data;
    displayMessage(message);
};

socket.onclose = (event) => {
    console.log("WebSocket connection closed:", event);
};
function displayMessage(message) {
    const chatMessages = document.getElementById("chat-messages");
    const messageElement = document.createElement("div");
    messageElement.textContent = message;
    chatMessages.appendChild(messageElement);
}

function generateRoomName(user1, user2) {
    let cleanedUser1 = user1.replace(/[@.]/g, '1');
    let cleanedUser2 = user2.replace(/[@.]/g, '1');
    return [cleanedUser1, cleanedUser2].sort().join('');
}
let currentChatRoom = generateRoomName(user,friend);
console.log('room', currentChatRoom);
updateChatUI();
```

```
  function updateChatUI() {
      console.log(`Joined chat room: ${currentChatRoom}`);
  }

  function sendMessage() {
      const messageInput = document.getElementById("message-input");
      const message = messageInput.value;
      if (message.trim() !== "") {
          const data = { text: user+":"+message, room: currentChatRoom };
          console.log(data)
          socket.send(JSON.stringify(data));
          messageInput.value = "";
      }
  }
```

This is the js code which is responsible for sending messages to the consumer so it could broadcast them further.