Übung: SignalR Chatter

Program a chatter with Angular and .Net Core using SignalR as the main communication channel.

A user can send messages to others. These messages are distributed by the hub to all others.

Frontend

The frontend shall more or less look like the screenshot above. The various buttons shall invoke actions on the hub described below in detail.

Backend

The backend shall consist of one hub providing data with SignalR.Additionally a controller for Admins shall be programmed.

ClientRepository

Create a service ClientRepository that stores - Username, - RegisterTime and - LastMessageTime for every connected client in a dictionary (key is the ConnectionId).

This service will be used by both the hub and the controller (see below).

Hub

Create a hub with the endpoint "/hub/chat" in Program.cs. Use a typed hub, i.e. create a file an interface IChatHub which holds both interfaces with the required methods to be called on the client as well as

the ones to be called on the server:

```
public class ChatHub(ClientRepository clientRepository)
  : Hub<IChatServerToClient>, IChatClientToServer
```

The interfaces should look like the following:

```
public interface IChatServerToClient
{
  Task NewMessage(string name, string message, string timestamp);
  Task ClientConnected(string name);
  Task ClientDisconnected(string name);
  Task AdminNotification(string message);
  Task NrClientsChanged(int nr);
}

public interface IChatClientToServer
{
  int GetNrClients();
  void SendMessage(string name, string message, string topic = "");
  void RegisterTopicsOfInterest(List<string> topicsOfInterest);
  bool SignIn(string username, string password);
  void SignOut();
}
```

Some notifications will only be sent to the Admin group (see below).

The "topic"-part is not required for the first version (see chapter "Extension" below).

Controller

Write a controller with two endpoints. The main purpose is to exercise, how a hub-message can be distributed from a controller.

"AllUsers" ... get some information about the connected users. Supply the timestamps in a readable format:

Additionally inform all administrators, whenever someone calls a controller method.

"Broadcast" ... Send a message to all users via a HttpPost.

Actions

SignIn / SignOut

With the methods `bool SignIn(string username, string pwd)` and `void SignOut()` a client can sign in to the Chatter.

Take care of the following:

- If the length of the password is less than 5 throw an exception. This exception must be displayed on the client.
- For simplicity the user is treated as an administrator if the username starts with "Admin".
- The return value of SignIn is true, if the user is an administrator.
- The user is added to/removed from the ClientRepository.
- A ClientConnected/ClientDisconnected message is sent to all other clients.
- If the client disconnects without explicitly calling SignOut, it will be disconnected anyway.

Message

A user can send messages to others with `SendMessage(...)`. These messages are distributed by the hub to all others by invoking `NewMessage(...)`. Don't forget to update LastMessageTime in the repository.

Administrators

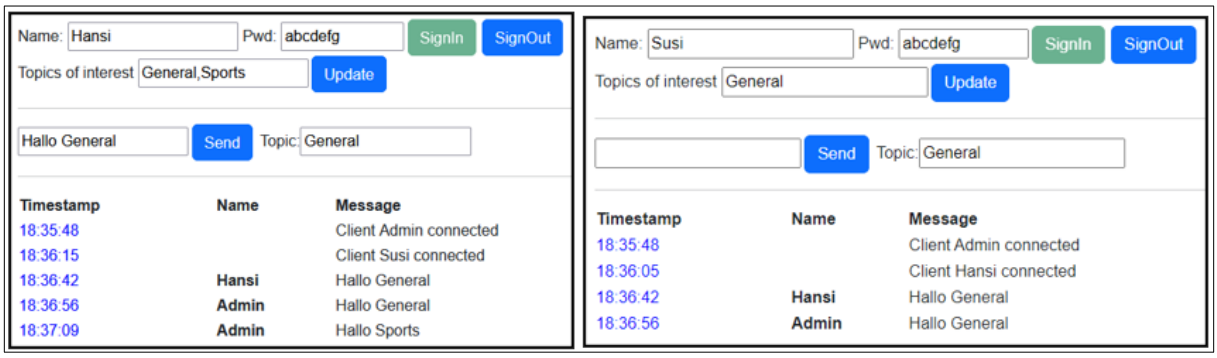
Administrators will get additional information from the backend:

- New number of clients, if a client signs in or out
- Anytime someone request all connected clients from the controller (see below).
- On the frontend an administrator has to request the current number of clients after signing in.

Extension

Every user can specify a list of topics she is interested in. This will be stored in the ClientRepository.

When sending a message, additionally a topic can be added. If a topic is given, only those clients are informed that are interested in that topic.




Content:
[
{
"name": "Admin",
"registeredString": "18:35:48",
"lastMessageTimeString": "18:37:09",
"topicsOfInterest": [
"General",
"Sports"
]
},
{
"name": "Hansi",
"registeredString": "18:36:05",
"lastMessageTimeString": "18:36:42",
"topicsOfInterest": [
"General",
"Sports"
]
},
{
"name": "Susi",
"registeredString": "18:36:15",
"lastMessageTimeString": "00:00:00",
"topicsOfInterest": [
"General"
]
}
]