# Simple chat server 1.0.0

## Programming challenge

Design and implement a chat server (and supporting api's) which can handle multiple users at the same time.

Chat users should be able to do all of the following:

- sign in and sign out
- list existing chat rooms and create new chat rooms
- post messages to any chat room (there is no need to join the room)
- post messages to any other user
- list messages from any chatroom
- list messages posted to directly them
- subscribe for real time delivery of messages from any chat room
- subscribe for real time delivery of messages posted to directly them

## Requirements

- All the functionality described must be implemented
- **Note:** If you cannot implement some of the functionality, please provide a design for it
- You should write majority of the logic yourself. You can use some third party libraries to support it (e.g. ORM, json parsing, etc.).
- The service should be able to handle more than 10,000 users communicating to each other at the same time.
- Data must be persistent (you can use any database)
- You can use any authentication framework/mechanism
- You can use any real time delivery mechanism (e.g. websockets, long polling, server sent events, etc.)
- You can use publicly available frameworks with MIT or comparable licenses

**Note:** In cases that the project requirements are not clear, feel free to interpret them as seems reasonable to you. Please make sure to document those assumptions. You can stub components on your solution if necessary.

## What we are looking for

- We are looking into implementation design, security, scalability, maintainability and testability.
- Code should run and implement the functionality as described
- Design choices, things don't have to be perfect but you should be able to discuss the trade offs.
- You should be able to make minor changes on the fly
- You can use the API definitions below as an example/guide but feel free to add additional API to meet the above requirements

**rooms API to manage chat rooms (list rooms, create new room) GET /rooms Lists existing rooms**

Lists all existing chat rooms on the server, if no rooms exists an empty array must be returned

| Parameters | Description |
|---|---|
| *Authorization* *required<br><br>string (header) | Authorization for the currently signed in user |

**Responses**

| Code | Description |
|---|---|
| 200 | List Of Chat Rooms including their respective roomId's |
| 401 | User is not authenticated |

**POST /rooms Add a new chat room** Create a new chat room.

| Parameters | Description |
|---|---|
| *Authorization* *required<br><br>string (header) | Authorization for the currently signed in user |
| *Name* *required<br><br>string (body) | The new name – name must be unique |

**Responses**

| Code | Description |
|---|---|
| 201 | Chat Room Created<br><br>**Headers:**<br><br>Name: Location<br>Description: Absolute URI for the created chat room resource<br>Type: string |

| | |
|---|---|
| 400 | Bad Input |
| 401 | User is not authenticated |
| 409 | Room name already exists |

## Messages API to manage chat room messages (list messages, post a new message, subscribe for realtime message delivery)

### GET /room/{roomId}/subscribe Subscribe to receive new messages posted in the room

This method should allow a client to subscribe to receive **in real time** new messages posted on the room. You can use any mechanism of your preference (e.g. web sockets, server sent events, long polling, etc.)

| Parameters | Description |
|---|---|
| *Authorization* *required<br><br>string (header) | Authorization for the currently signed in user |
| RoomId *required<br><br>string (path) | ID of the room to subscribe |

**Responses**

| Code | Description |
|---|---|
| 202 | |
| 401 | User is not authenticated |
| 404 | Room does not exist |

### GET /room/{roomId}/messages Lists messages in a room

| Parameters | Description |
|---|---|
| *Authorization* *required<br><br>string (header) | Authorization for the currently signed in user |
| RoomId *required<br><br>string (path) | ID of the room to subscribe |

**Responses**

| Code | Description |
|---|---|
| 200 | Messages |
| 401 | User is not authenticated |

| | |
|---|---|
| 404 | Room does not exist |

**POST /room/{roomId}/messages Post message to a room**

| Parameters | Description |
|---|---|
| *Authorization* *required<br><br>string (header) | Authorization for the currently signed in user |
| RoomId *required<br><br>string (path) | ID of the room to subscribe |
| Body *required<br><br>(body) | Chat message object |

**Responses**

| Code | Description |
|---|---|
| 201 | Message posted |
| 401 | User is not authenticated |
| 404 | Room does not exist |