API CreateTopic (topic, clientID)

Create topic API is used to create topic by calling client. Creating the client, system randomly assigns him an integer that will be his ID. After calling CreateTopic API, first we check if the given topic is equal to **exit**, it will allow user to quit that option and go back to main menu of APIs. Server check first if there is such topic in the MessageBuffer. If topic is not there, server add (ClientID, topic) as set to the messageBuffer dictionary **as key** and empty array as **value.** Since topic just has been created, there should be no subscriber to that topic, so we also add topic to the Subscription dicitonarty as **key:**topic and **value**: []

After completing we can back the acknowledgement that topic has been added or not.

API DeleteTopic (topic, clientID)

Similar to the Createtopic API, this one delete topic by calling client. In my code, when client asks to delete the topic, we ask for the topic name. If input is not **exit**, which is the key code to return to main menu, API send ClientID and topic name to the server. We check if there is such **key** as (topic,Client) is messageBuffer dictionary. If so, we delete that from message buffer and also from subscription table, as there is no more that topic exists. On top of that, we also delete all clients that have subscripted to the topic.

API Send (topic, ClientID, message)

We deserialize the incoming request and fetch the message. After that we check if the topic and the ClientID exist in the messageBuffer. If there is such topic in the buffer, we append that message to the messageBuffer[(topic, ClientID)]. We check for the empty messages as well, to make sure that we don't accept empty messages. In case if topic does not exist, we reply back that such topic does not exist for that client. Only client that is the creator of topic can send message to that topic.

API Subscribe (topic, clientID)

When client ask to subscribe to the topic, we check if that topic exist in the system. If so, we check if the client already subscribed or not. After that we add the client to the subscription dictionary and we initialize the readingIndex to 0. As the client just subscribed, he hasnot read any messages so we need to initialize it to 0.

API Pull (topic, clientID)

First we check if he topic exist. After we make sure that client is subscribed to that topic, so he is allowed to pull messages from there. If yes, we fetch his readIndex from the readHistory table to get the index to the message which he has read before. Server return the all messages starting from that index till end to the client. If there is something new, client will receive it or answer will be **empty.** After pulling we update his readIndex to the size of the messageBuffer of that topic.

After each pull we call garbageCollector so that to check if all subscribers have read the messages from that topic. If yes, all old message are removed from the messageBuffer and the indexes of subscribers are assigned to 0.