CPSC 313: Distributed and Cloud Computing
Spring 2022

Lab 5: Message based chat – MVP3

We now have a good basis upon which we can build more features/functionality. In this MVP, we are adding a few additional features: Here is a quick list of reasonable next features:

- Required for this MVP:
    - Pay down debt
        - More testing
            - Add simple scale testing on get_messages
        - Better commenting
        - Factoring
    - Authentication
        - All API's must be authenticated
    - Reading messages in sequence and sorting
        - We'll want to move the message sequence_num out of mess_props and into the main ChatMessage class to make it easy to sort
        - Each call to the DB, should include a sort on sequence num
        - We then have to insert any new messages in the right place in the deque
            - This must be efficient, so use binary search to find the right place
    - Blocking users
        - If an alias is blocked, you don't see any of their messages in your rooms
        - The blacklist (list of blocked users) is kept in the ChatUser class
        -
    - Registering rooms, users
    - Deleting rooms, users
        - For each of these, have a flag (Boolean var) in the appropriate classes called "removed" that is set when an instance is removed.
        - Add options for relevant methods within the classes to handle removed instances
            - Don't return rooms that are removed
            - Don't return or accept users that are removed
            - It should be easy to undo – simply change the flag
    - Editing/deleting messages sent
        - Search for messages
            - By user pip
        - Remove messages
            - By user
    - Simple metrics
        - Latency for each get_messages and send_message call
        - Counts of messages returned from get_messages
- Backlog:
    - Sending personal messages within a room
    - Search for messages
        - By keyword(s)

- Remove messages
  - By Keywords
- Simple UI

•

Requirements and notes:

- Both unittest and pytest are good test frameworks for python. I tend to use both. You are required to use either of these. Your tests must be in a class that inherits the test framework's TestClient class.
- I strongly recommend you use VSCode to run your tests. It's easy to integrate either pytest or unittest (or both) into VSCode to run your tests.
- You should be getting used to using postman (postman.com) to help you manually test your API's