**I propose the following layout:**

1. **UML Diagrams**
   1. **Use-Case diagrams**

**Explain the scenarios**

* + 1. **Scenario 1**
    2. **Scenario 2**
    3. **Scenario 3**
  1. **Activity diagrams**

**Explain the 3 main activities**

* + 1. **Scenario 1**
    2. **Scenario 2**
    3. **Scenario 3**

1. **Use-Case Diagrams**

Please introduce the 3 Use-Case scenarios here

**1.1 Login Use-Case scenario**

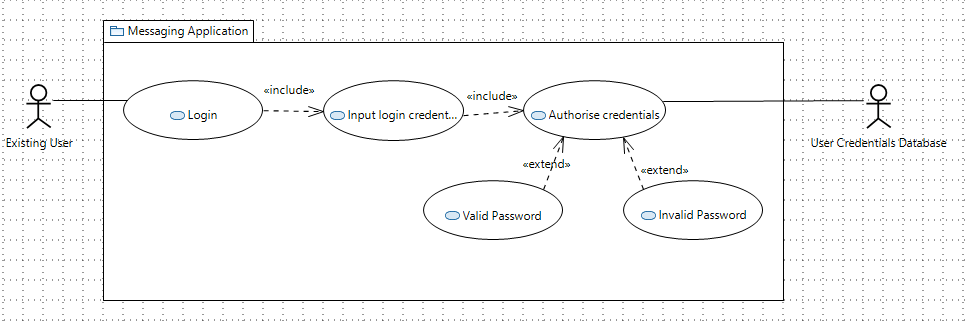


Figure 1: ….

Figure 1 shows the login process. The Use-case scenario assumes that the user is valid and that the user will use the credentials to login into the system. The credentials introduced by the user are then validated by the database.

**1.2. Send Message Use-Case scenario:**

A user attempts to send a message, the system will first need to validate their connection to ensure that the target user is online and connected to the broker:

* If the user is online:
  + The source message is sent, and the source chat history is updated
  + The target user history is also updated
  + The chat histories is updated in both ends within each databases
* If the user is offline:
  + Their message is stored within the database temporarily
  + Their connection is then repeatedly checked until they are confirmed to be online
  + Once the user is online, their message is then sent, and the following stages mentioned above occur

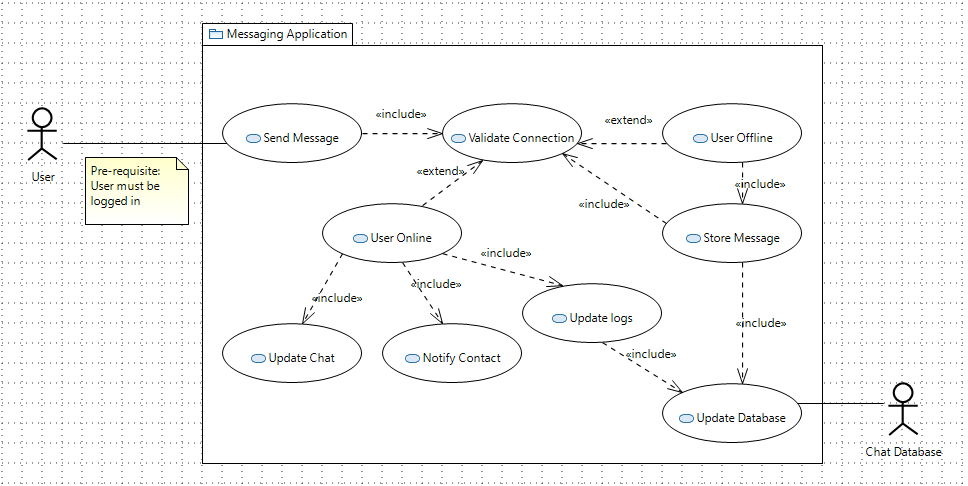


Figure2:…

**Login Activity Diagram:**

In this scenario, the User attemps to login into the application by inputting the User credentias (i.e username and password). The credentials are then authenticated against the existing credentials within the database.

* If the password is valid, they access their user space
* If the password is invalid, they are notified of their credentials being incorrect and are asked to try again.

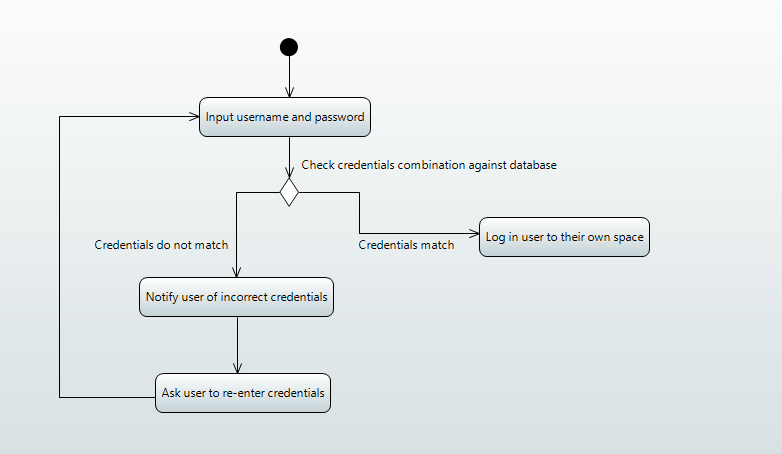


Figure 3: ...

### Send Message Activity Diagram:

The activity diagram coversthe flow of events required to send a message. The user triggers an event when sending a new message when using the application. The target User status is validated to ensure they are connected to the broker. If the target User is offline, then the message is temporarily stored on the database. The application keeps checking the status of the Users every **n** seconds.

Once the user's connection is established as being online, the system retrieves any messages that are stored in the database. A queue of messages to be sent is created and messages are ordered accordingly in a first in, first out fashion. A fork then occurs for several activities to run concurrently, such as notifying the receiver that a new message has been received, updating chat logs and updating the chat itself. Once these are all completed, they merge back to end the 'send message' activity.



Figure 4: