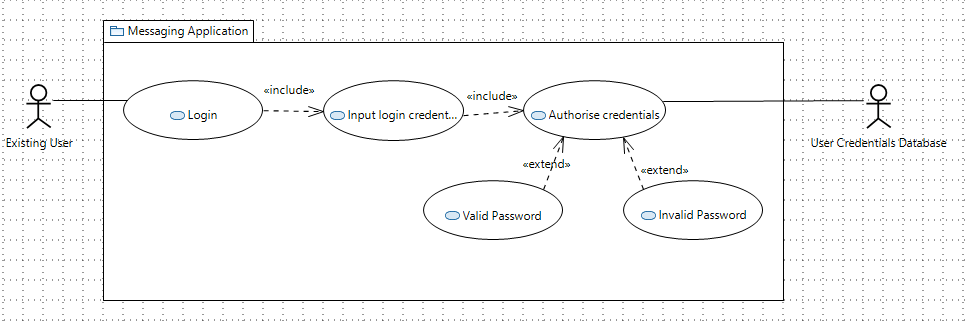
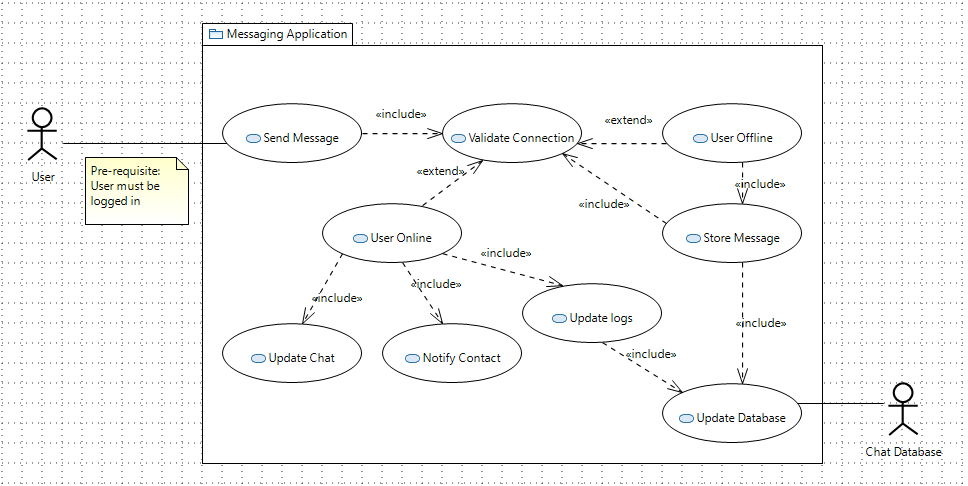
**Login Use-Case:**

This diagram looks into the process of a user attempting to login into their account on the application. Assuming that the user is an existing user, they can use their credentials to log in. Their credentials are then authenticated against the data stored within the credentials database.

**Send Message Use-Case:**

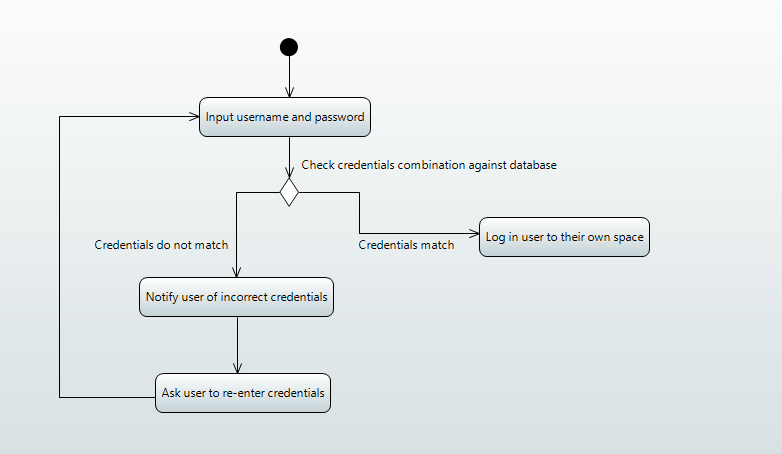
When a user attempts to send a message, the system will first need to validate their connection to ensure that they are online and connected to the server:

* If the user is online:
  + Their message is sent, and their chat is updated
  + The contact they are talking to is notified
  + Their chat history is updated within the database
* 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

****

**Login Activity Diagram:**

The diagram first begins with the user attempting to login into the application by inputting their username and password. These 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.

### Send Message Activity Diagram:

This diagram looks at the flow of activities involved in sending a message. The activity first begins when the user sends a message through the application. Their status is then checked to ensure they are connected to the server. In the case that they are offline, their message is temporarily stored on the database. The process then loops back to the check to see whether they are online yet.

Once the user's connection is established as being online, the system retrieves any messages that are stored away 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.

