* Cover sheet (Project title, Group members' names and SID.)
* Problem Definition
* Project Objectives
* Stakeholders List
* Success /Acceptance Criteria for each Stakeholder
* Use case diagram(s)
* Selected Use case Descriptions {only two descriptions}
* Sequence diagram(s) {for the selected use case for descriptions}
* System Architecture
* Detailed Class diagram(s)
* State-machine diagram {for the whole system}
* ER – Diagram (Data modeling)
* GitHub link to your project source code
* ACTUAL APP LOL
* Conclusion (lesson learned)
* Project Work Breakdown Structure (as an appendix of your report)
* Task Assignment Matrix (as an appendix of your report)
* Sample of commits on the selected version control system (in appendix)
* Powerpoint slides
* Video Presentation

**Problem Definition**

Boston University needs a safe and reliable messaging platform for its students to use for communication purposes which will help increase student well-being and school spirit. The messaging platform should be efficient, have a sleek user-interface and user-experience, and support encrypted communications for everyone at BU.

**Problem Objectives**

With a team of 3 skillful student engineers from CS 411, develop a reliable and effective chat application that Boston University students can use to communicate with other students in classes and across the institution in *five clicks or less*. The registration and login process should be swift using BU identification. Reduce communication time and hassle for students with effective chat and group forming.

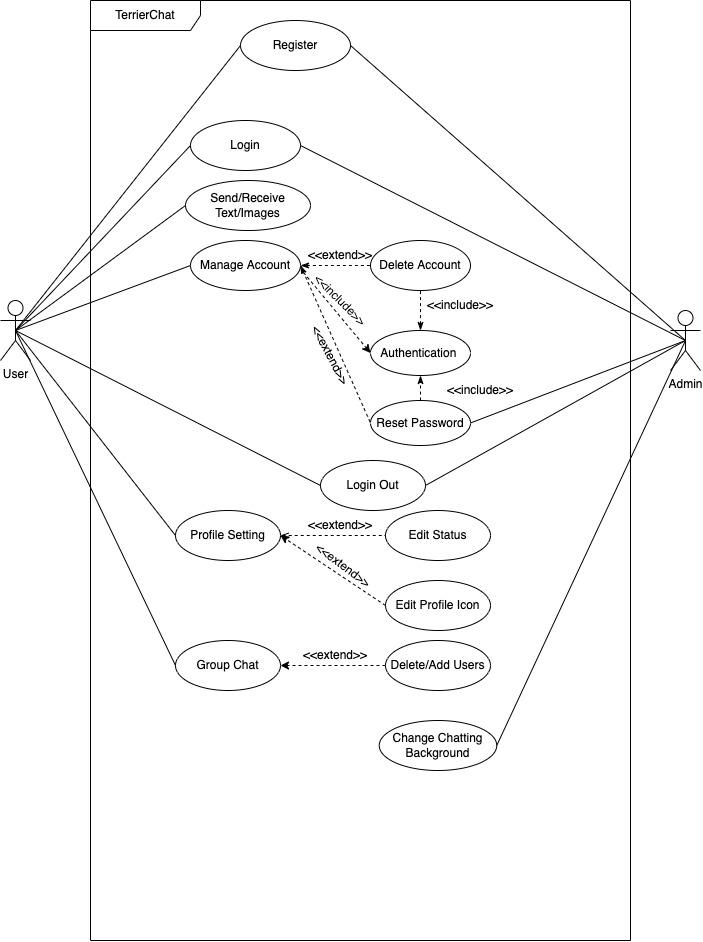
**Stakeholders List**

| **Stakeholder** | **Criteria** |
| --- | --- |
| BU Administration | Chatting systems should be easy to maintain with clear documentation. Chatting server should provide security for each user account allowing for password reset, delete account, and changing icons. |
| Students | Chatting systems should highly respect the privacy of each user and protect each student account’s security. The messaging system should also have a minimal UI/UX to minimize student distraction and maximize efficiency as well as usability. |
| Professors and TAs | Accounts of professors and TAs can see students associated with their courses. Professors and TAs can send announcements to their relevant students. |

**Functional Requirements**

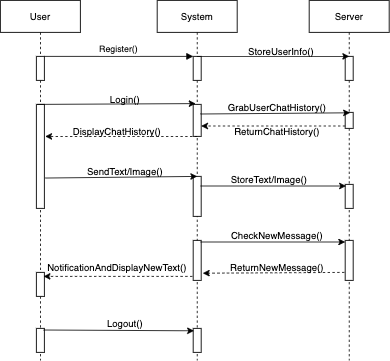
| Requirements | Descriptions |
| --- | --- |
| Login/Signup | Users should be able to login or sign up for their own accounts. The system should store these information in a secured data structure |
| Send/receive messages | Sending and receiving messages is the most essential function of this chat application |
| Manage account | The users should be able to manage accounts. This includes resetting passwords and deleting an account; both requirements authentication which is a security question that the user set up when they created the account |
| Profile setting | The users should be able to change their profile setting, which includes giving themselves a unique avatar and setting a status that shows whether they are online or not |
| Group chat | The users can create a group chat with their friends. The admin of the group is able to delete a user from the group chat |

**Use Case Diagram**



**Sequence Diagram**

(Send/Receive Text Use Case)

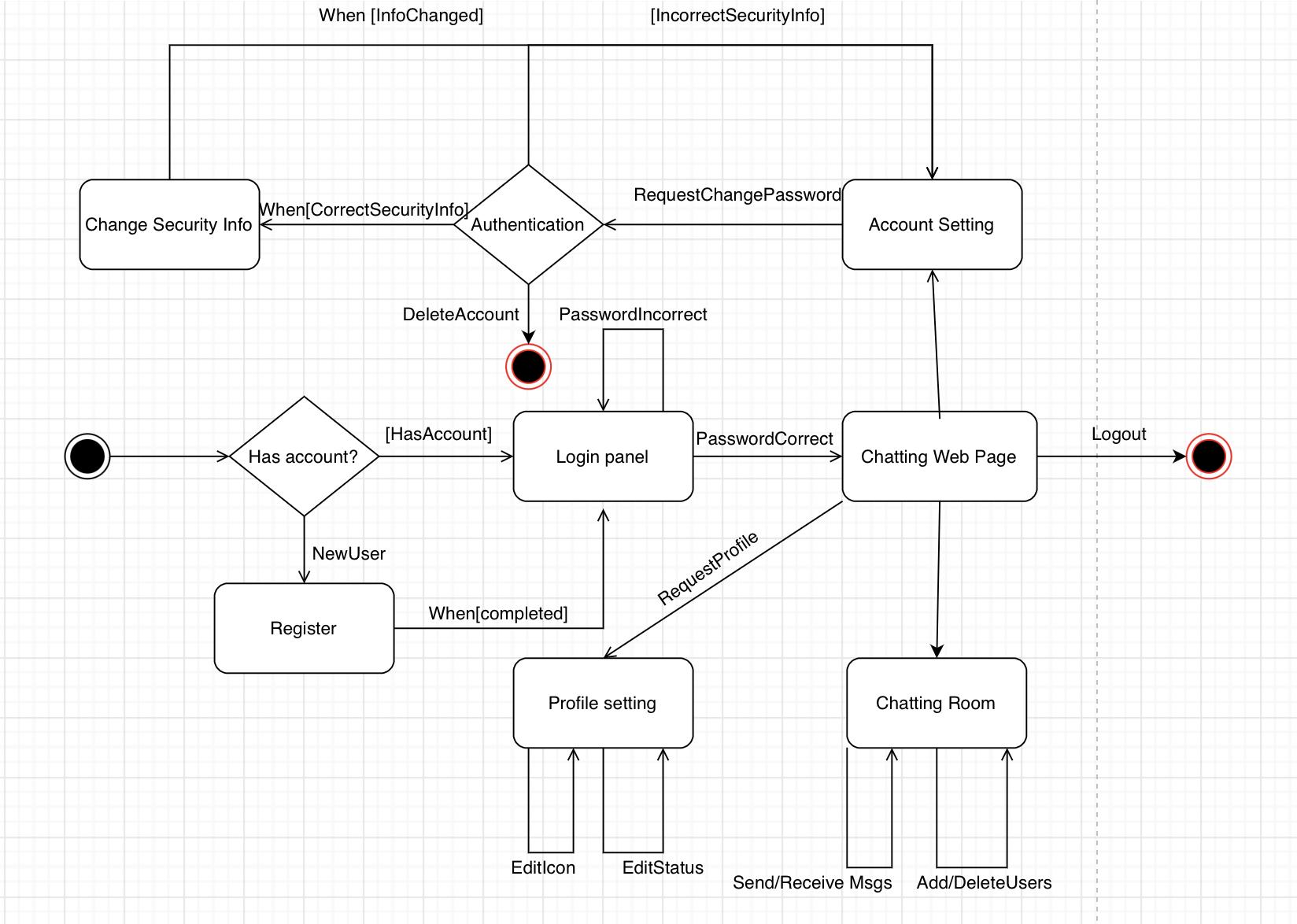


**Use Case Descriptions**

| Use case name: Group chat |
| --- |
| Actors: Users, admin |
| Stakeholder:  BU students: The BU students as the main user group should be able to make changes to their account. |
| Triggering Event: The user request to create a group chat |
| Preconditions:   1. User must be logged in |
| Postconditions:   1. System should store chat history in a secured data structure |
| Steps performed:   1. The user attempts to create a group chat 2. System displays the page for group chat where group chat admin can add/delete other users |

| Use Case Name: Manage accounts |
| --- |
| Actors: Users |
| Stakeholder:  BU students: The BU students should be able to manage their accounts |
| Triggering Event: The user request to access to their account information |
| Preconditions:   1. User must be logged in 2. There is a secured data structure that stores user information and allows changes |
| Postconditions:   1. User must be authenticated before given permission to make any changes to the account 2. The database must store correct information after any changes that the user has made |
| Steps performed:   1. The user attempts to access his/her account management page 2. System displays the account management page 3. The users attempt to reset password/delete the account 4. System requires authentication from the user 5. The users type in correct information that matches with information stored in databases |

**State-Machine Diagram**



**Appendix**

**Project Work Breakdown Structure**

