**Software Requirements and Design Document**

**For**

**Group <13>**

Version 2.0

**Authors**:

Jillian Bice

Matthew Bucataru

Brandon Specht

Brandon Cook

Libby Sauer

# Overview (5 points)

This project is a roommate matching app. You start in the app by making an account and profile. Part of the profile process is filling out a questionnaire that is about your roommate preferences. After the questionnaire is completed, you get a Tinder-style interface that allows an individual to swipe left (for no) and right (for yes) on certain roommates on your feed. If you and another person both swipe right on each other it creates a match. If you match with a possible roommate it allows both sides to chat and discuss specific information to determine whether or not the roommate match is the right match for you.

There will be a sign-in page, a dashboard, a swiping page, a message list page, chat page, and a profile editor page. The sign-in page will allow the user to enter a username and password. If the username and password is correct then it allows the user to continue to the dashboard, where they can access the swipe page. Once on the swipe page, the user can swipe left and right on possible roommates. The swipe page will have a navigation bar on the top that allows the user to access all the pages that were specified above.

# Functional Requirements (10 points)

1. The system must have a functional sign-up page that populates the database with the username and password. - High Priority
2. The sign-up page will also populate a separate table with roommate preferences, including: desired number of roommates, desired age range, apartment type. - Medium Priority
3. The login page will authenticate users and pass them to their dashboard.- High Priority
4. From the dashboard, the user will be able to open their messages, open the swiping page, and edit their roommate preferences set in the sign-up process. - Medium Priority
5. From the messages page, the user will be able to send and receive text messages to and from their matches. - Low Priority
6. From the swiping page, the user will be presented potential roommate matches sequentially. - High Priority
7. The user will be able to swipe left to pass, or swipe right to like a particular potential match. - High Priority
8. The user will be “matched” with a potential roommate when both people like each other by swiping right on each other. - Medium Priority
9. The user will continue swiping until there are no more possible likes, or until they return to their dashboard. - Medium Priority
10. Along the navigation bar, which will be on every page, there will be shortcuts to user information like logging out, managing settings, and the user’s home page. - Low Priority

# Non-functional Requirements (10 points)

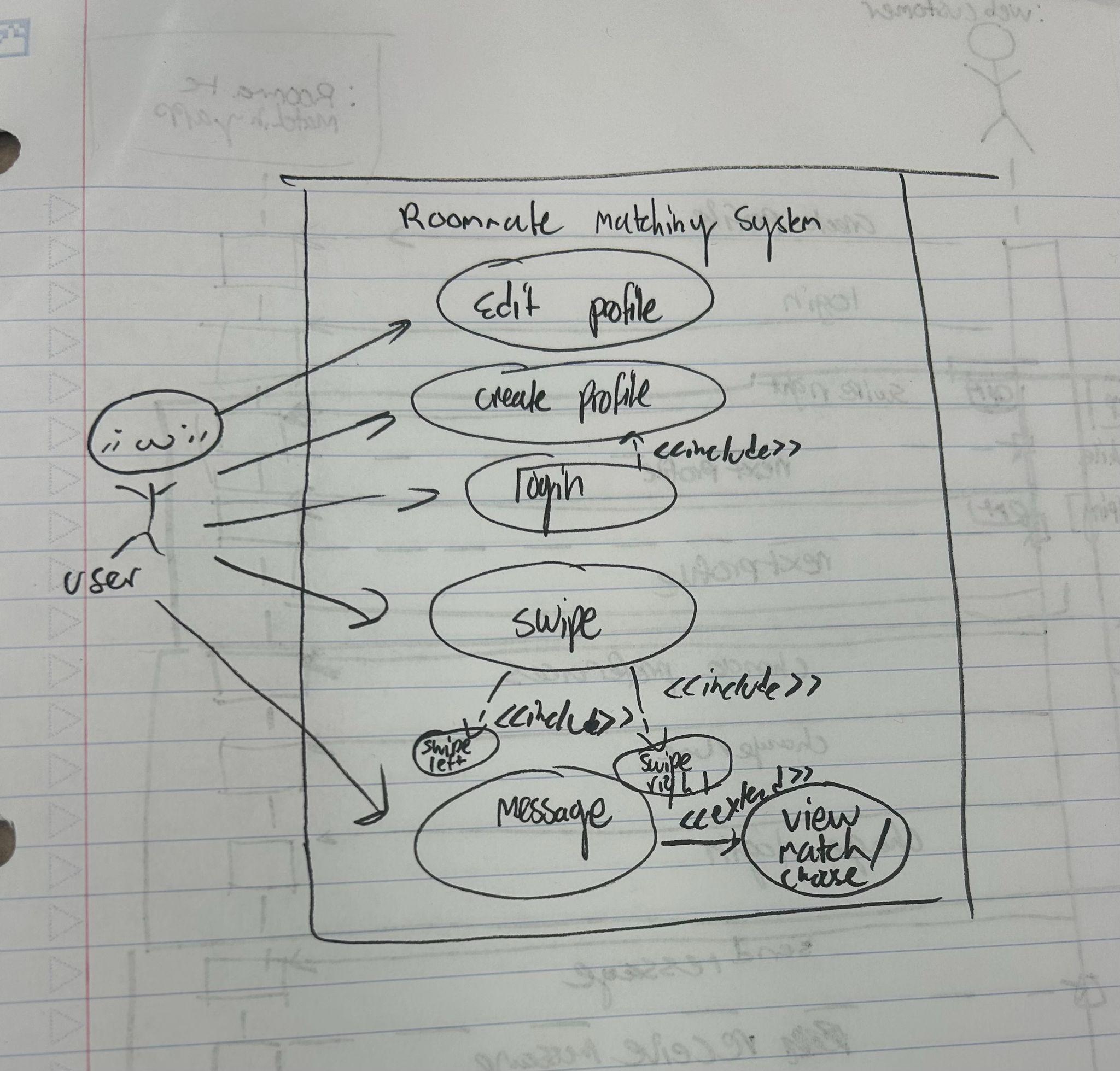
Security will be enforced with the standard username and password hashes.

The code will be easily readable and simple to follow.

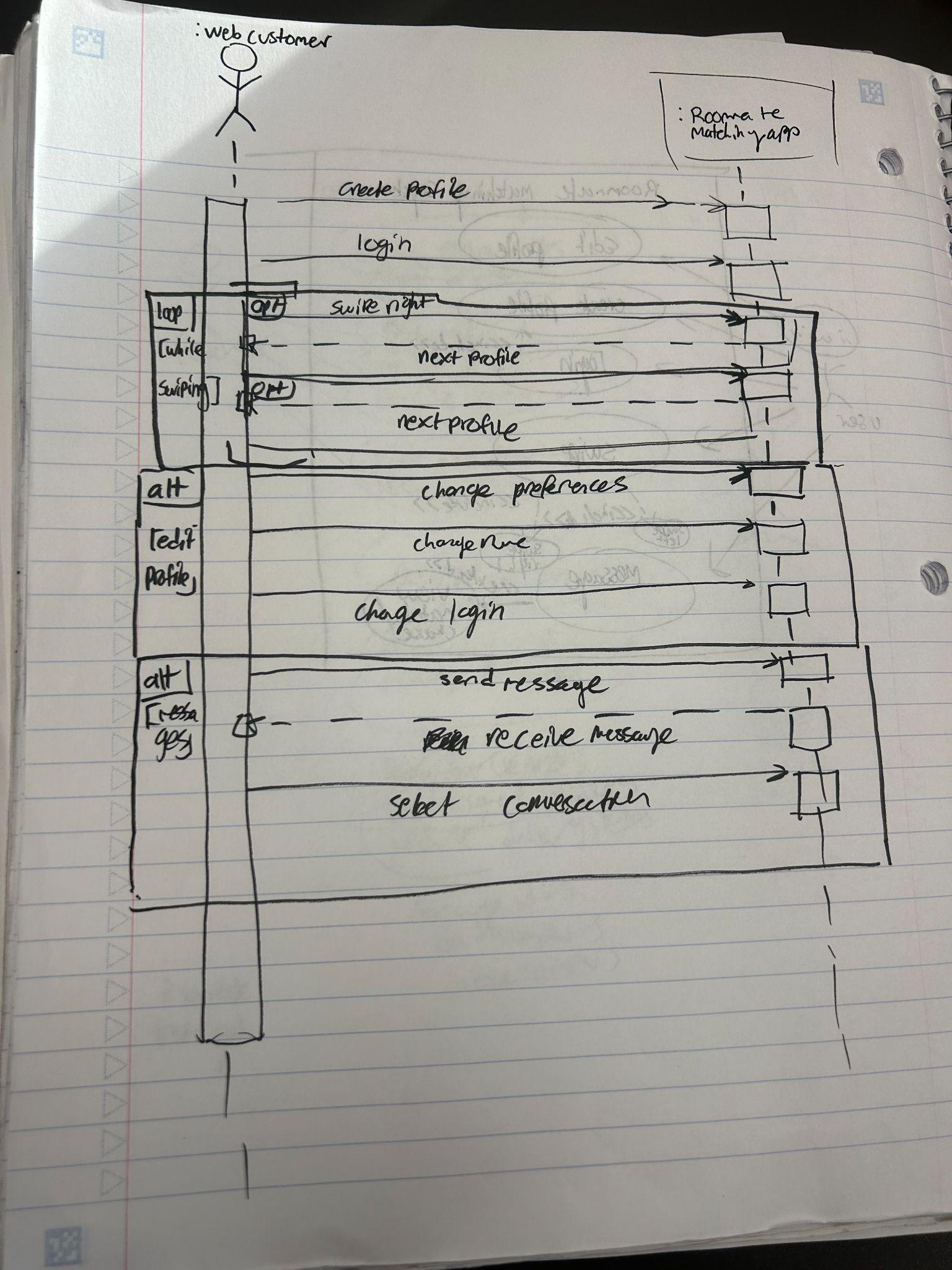
HTTPS will be used as opposed to HTTP for security reasons.

The system will be able to hold enough data to accommodate all users while still performing most tasks in under 1 second.

# Use Case Diagram (10 points)

**

# Class Diagram and/or Sequence Diagrams (15 points)

**

# Operating Environment (5 points)

The environment should be able to operate on most operating systems that support a web browser, as well as mobile devices including iPhone and Android. We do not intend on creating a mobile application, but the website should display correctly on mobile screens in browsers such as Safari, Google Chrome, and Firefox.

# Assumptions and Dependencies (5 points)

While we do not intend on reusing any software components from another project, we are going to be relying on a third-party for a cloud-based server to run the web app on. The database is running on Amazon RDS.

We are also relying on a fair bit of NPM packages, which will add to our dependencies.

Issues regarding the operating environment should primarily center around getting the front end and the back end to talk to each other, but most of the problems are going to be within our own code and fetching data from the database.