SW Engineering CSC648/848 Fall 2021

RUMI

Section 02 Team 01

—

Team Lead ➔ Alex Shirazi
(Email: aredjaian@mail.sfsu.edu)
Github Lead ➔ Jasmine Kilani
Front-end Leads ➔ Nakulan Karthikeyan
Anmol Burmy
Front-end Team ➔ Jasmine Kilani
Rasul Imanov
Back-end Leads ➔ Joshua Miranda
Cheng-Yu Chuang
Back-end Team ➔ Rasul Imanov

—

Milestone 2

(October 5, 2021)

History Table

Revisions	Date
M2 - v1.0	October 5, 2021
M2 - v2.0	October 15, 2021

Table Of Contents

1 Functional Requirements - Prioritized

➔ Priority 1 (Must Have)

1. **Admin User**

   1.1. An admin shall ban any users from the website.
   1.2. An admin shall be able to delete posts.
   1.3. An admin shall be able to delete comments.

2. **Guest User**

   2.1. A guest user shall be able to see the available rooms and roommates on the website.
   2.2. A guest user shall be able to search any room and roommate available on the website.
   2.3. A guest user shall be able to view other user's profiles.
   2.4. A guest user shall be able to sign up and make a new account.

3. **Account**

   3.1. A guest user shall be able to make a new account.
   3.2. An account shall have a unique id for each user.
   3.3. An account shall require a username, which must pass the conditions of starting with a character (a-z or A-Z) and having the length of at least 3 characters.
   3.4. An account shall require a valid email.
   3.5. An account shall require a password, which must pass the conditions of having the length be at least 8 characters long and must contain at least 1 upper case letter and 1 number and 1 of the following special character (/ * - + ! @ # $ ^ &).
   3.6. An account shall require the age of the user.
   3.7. An account registration shall require the user to accept the Terms of Service and Privacy Rules of the website.

4. **Registered User**

   4.1. A registered user shall be able to login to the website.
   4.2. A registered user shall be able to see the available rooms and roommates on the website.

4.3. A registered user shall be able to search any room and roommate available on the website.

4.4. A registered user shall be able to create posts.

4.5. A registered user shall be able to delete their posted post.

4.6. A registered user shall be able to comment on the posts by other users.

4.7. A registered user shall be able to update their profile.

4.8. A registered user shall be able to see other users' profiles.

4.9. A registered user shall be able to copy link, share or save room and roommate's profile.

4.10. A registered user shall be able to set a preferred search for finding a room (location, price, amenities etc) and roommates (majors, school, smoker?, pets?, language, same interest, hobbies etc).

4.11. A registered user shall be able to get alerts for matching profiles that they are looking for.

5. Profile

5.1. All registered users shall have a profile.

5.2. A profile shall contain an about me and information about the user it belongs to.

5.3. A profile shall contain a profile picture of the user.

6. Posts

6.1. A Post shall be posted only by a registered user.

6.2. A Post shall be posted by a user looking for a roommate or a room.

6.3. A Post shall be posted by a user who wants to rent a room.

6.4. A Post shall contain a photo, a caption or a description.

6.5. A post shall have a post creation date and time.

7. Comments

7.1. Comments shall be posted by registered users under posts.

7.2. Comments shall contain a comment made on the post and the name of the user who made the comment.

7.3. Comments shall have a creation date and time.

8. **Messaging**

   8.1.   Messages shall be sent from a registered user to another registered user.
   8.2.   Messages shall have the message along with the time it was sent on.
   8.3.   Messages shall be replied to.

9. **Search**

   9.1.   Guest users shall be able to search any post on the website
   9.2.   Registered users shall be able to search any post on the website.
   9.3.   Search shall be filtered to suit the preferability of the user who is searching.
   9.4.   Search shall update the contents of the website as the search terms are entered.

➔ **Priority 2** (Desired)

1. **Admin User**

   1.1.   An admin shall receive reports about users.
   1.2.   An admin shall notify users about the status of their reports.

2. **Registered User**

   2.1.   A registered user shall be able to edit their posted post.
   2.2.   A registered user shall be able to edit their posted comments on any posts.

3. **Profile**

   3.1.   A profile shall contain all the posts posted by the user.

4. **Posts**

   4.1.   A post shall contain multiple photos.
   4.2.   A post shall be edited by the user who posted it.
   4.3.   A post shall be deleted by the user who posted it.

5. Comments

5.1. Comments shall be edited by the user who posted it.
5.2. Comments shall be deleted by the user who posted it.

6. Search

6.1. Search bar shall display the recent searches done by the user.

7. Reviews

7.1. A review shall be left by a registered user regarding living spaces and other registered users based on their experiences.
7.2. A review shall be based on stars. A 5 star being the highly recommended living space or a user, while a 0 star being the lowest recommended.
7.3. A review shall have a description of the review.
7.4. A review shall have a creation date.

→ Priority 3 (Opportunistic)

1. Registered User

1.1. A registered user shall be able to hide posts that they have posted.
1.2. A registered user shall be able to hide comments on their posts.
1.3. A registered user shall be able to view posts related to their previous and recent searches.

2. Profile

2.1. A profile shall display the recent activity (likes, shares and comments) of the user it belongs to.
2.2. A profile shall contain all the comments posted by the user.

3. Posts

3.1. Posts shall be sorted by the users on the basis of reviews.

4. **Comments**

    4.1.   Comments shall have stickers and emojis.

    4.2.   Comments shall have GIFs.

5. **Messaging**

    5.1.   A message shall be deleted by the user who sent it.

    5.2.   Registered users shall message a photo or a video to other registered users.

    5.3.   A message shall have stickers and emojis.

    5.4.   A message shall have GIFs.

    5.5.   A message can be sent to a group of users.

6. **Search**

    6.1.   The search shall display the option to view most popular searches done by other users, when clicked on the search bar.

2 UI Mockups and Storyboards

➔ **UI Mockups:**

Registration page:

Login page:

Post Room:

Post Roomate:

Navigation bar: Rumi Logo | Find Room | Find Roomamate | Post | Profile | About | Login/logout

Post

Type [Room] [Roomphate]

First Name [____]
Last Name [____]
age [____]
School [____]
Willing to pay monthly [____]

Description [____]

Flags [____]

[Submit]   [Import from Profile ?]

Find Roommate:

Find a Room:

Rumi Logo | Find Room | Find Roommate | Post | Profile | About | Login/logout

Find a Room

Filter | [search bar] | [search icon]

Recommended:

Room Post Title

Cover Photo

Details:

Room Post Title

Cover Photo

Details:

Profile Page:

About Page:

Software Engineering SFSU
Fall 2021
Section 02
Team 01

Alex Shirazi	**Nakulan Karthikeyan**	**Jasmine Kilani**
Joshua Miranda	**Anmol Burmy**	**Cheng-Yu(Alan) Chuang**
	Rasul Imanov	

➔ Storyboards:

➔ Use Case 1: A Student Finding a Room

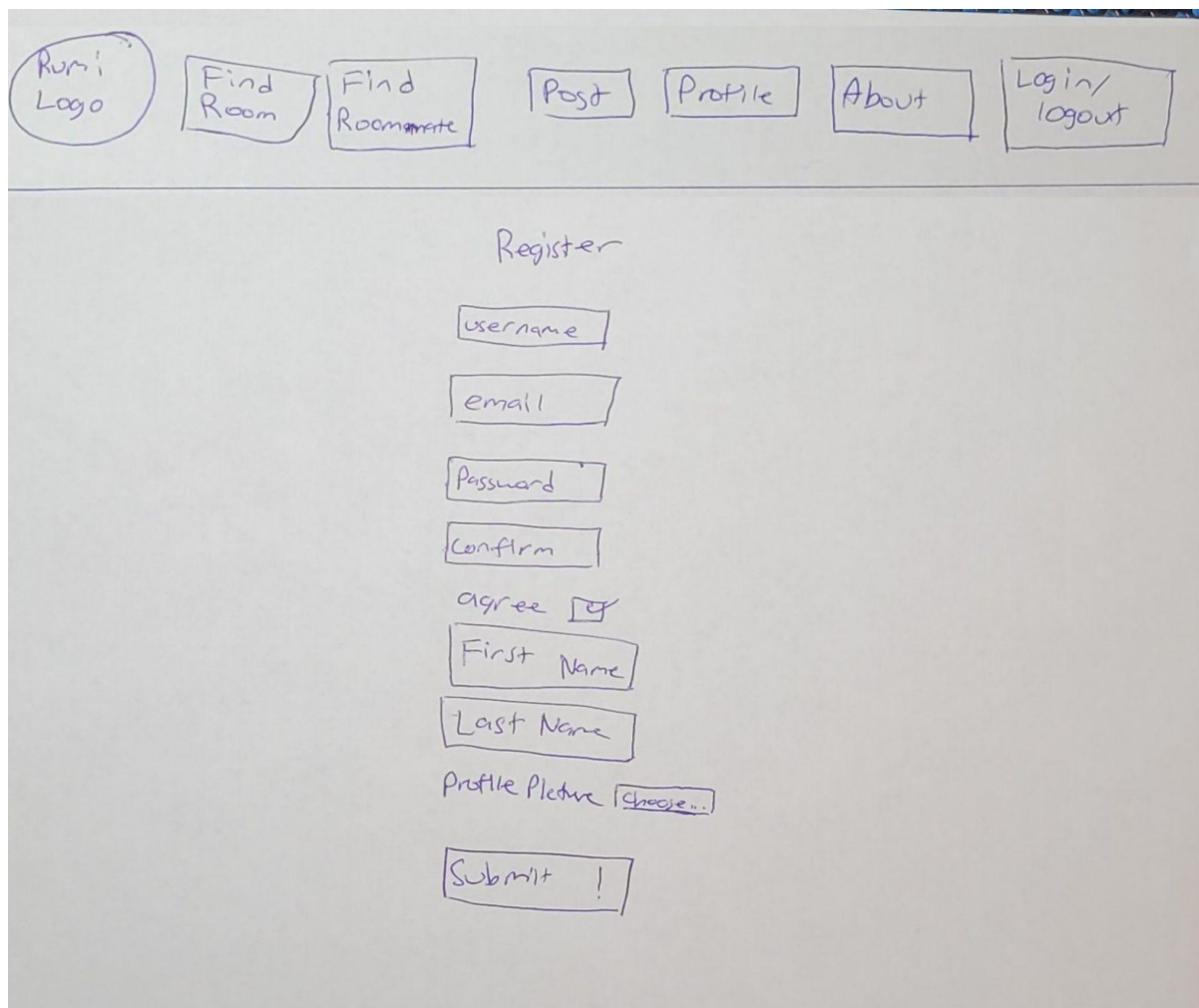The student first navigates to the login page to be able to find a room. If the students do not have an account, they can register directly through the login page registration redirection.
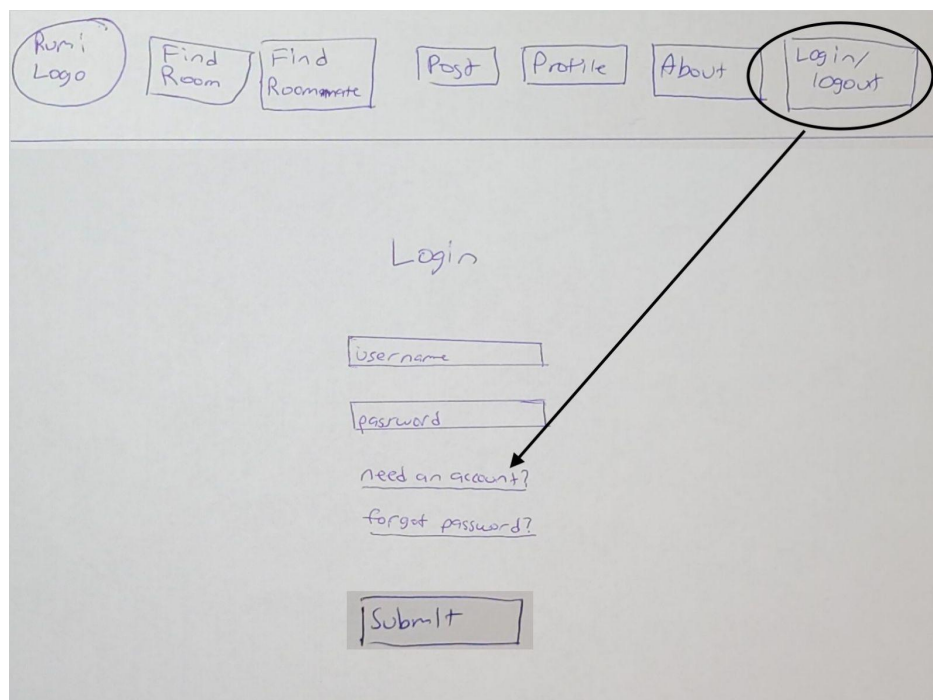
From the registration page, the student can now successfully create an account.

The user looks for the room they want to find. They could either view listings that the app recommends, or manually search for specific listings
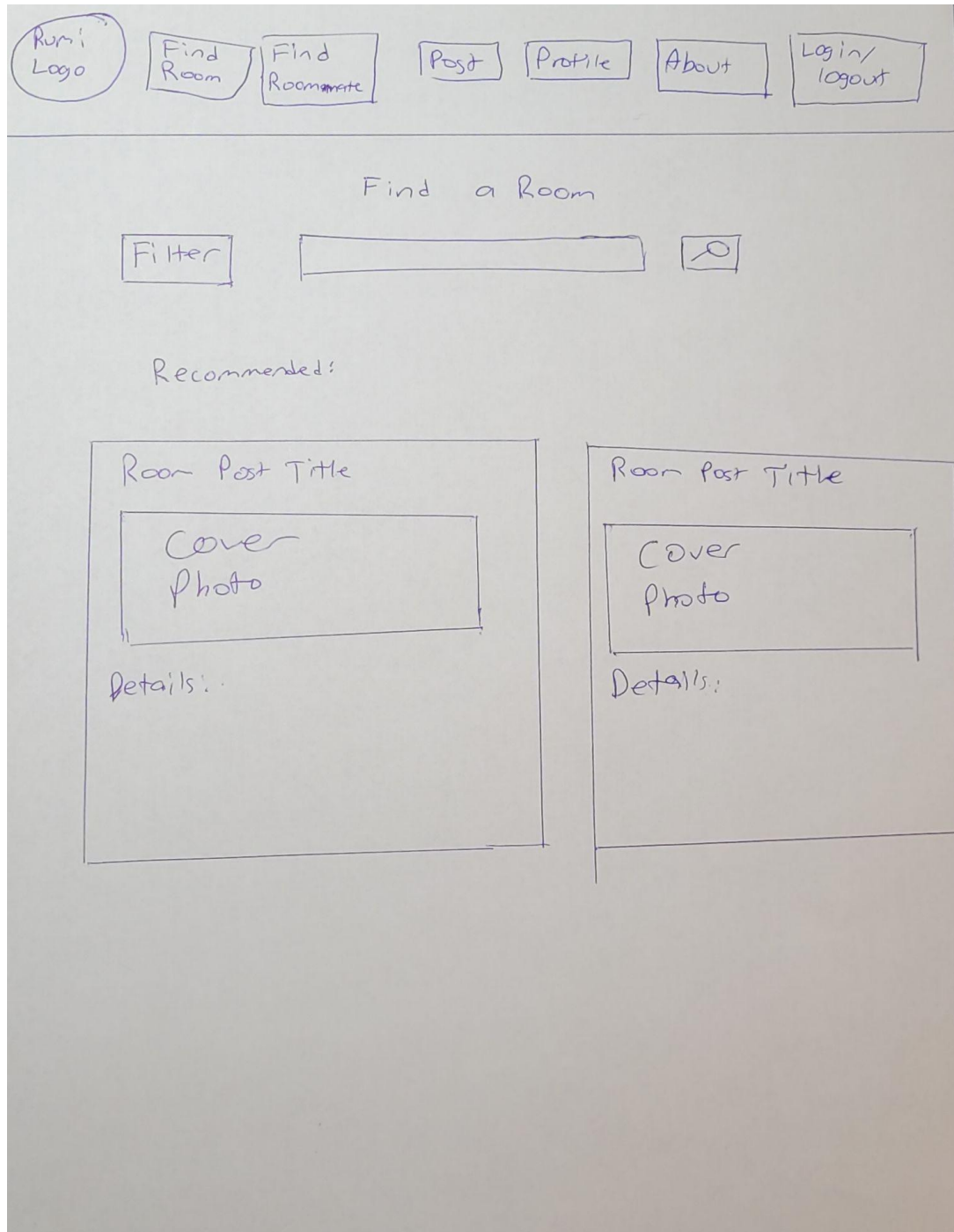
➔ Use Case 2: A Student posts a room to split rent cost

The user who is looking for another roommate to join them in this room fills out all information about the room. When they submit the post, users looking for a room can find this post under the 'find room' tab.

➔ Use Case 3: A Student posts a roommate request find a room
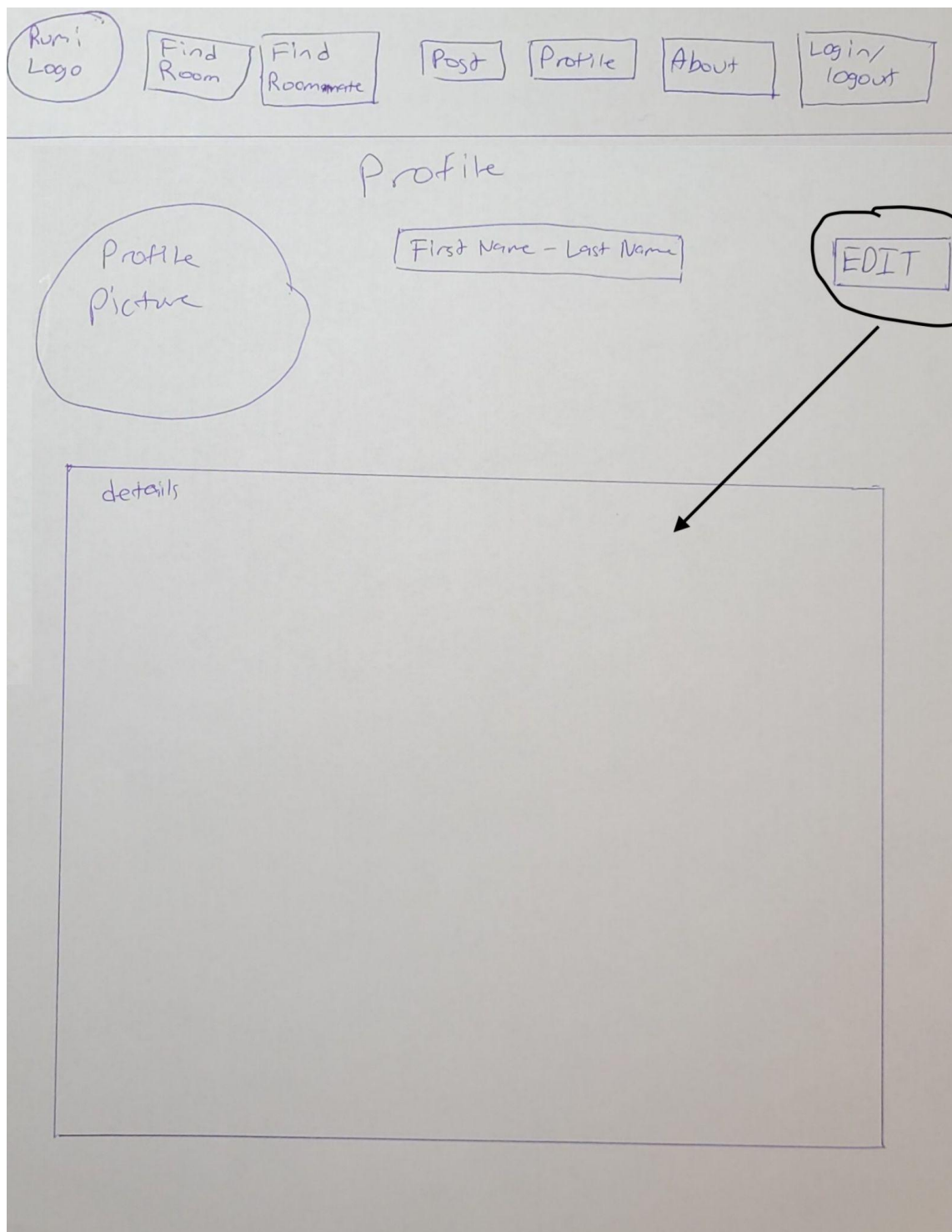
The user navigates to the profile page, clicks edit and fills out their profile page

The user makes a roommate post and can import details from their profile to speed up the posting process. When they submit, people looking for a roommate can find this post under the 'Find Roommate' tab.

3 High level Architecture, Database Organization

➜ DB organization

◆ *Users*

id	char(36)	PK, Unique, not null
username	char(32)	Unique, not null
password	char(36)	Not null
last_name	char(256)	
first_name	char(256)	
email	char(256)	Unique, not null
phone	char(36)	
birthday	date	Not null
school	char(256)	
major	char(256)	
smoker	bit	1:true/0:false
pets	bit	1:true/0:false
language	varchar(max)	
interest	varchar(max)	
hobbies	varchar(max)	
created_date	date	Generate automatically
deleted	bit	1:true/0:false
deleted_date	date	
admin	bit	1:true/0:false
activated	bit	1:true/0:false, admin shall ban any user

◆ *Post*

id	char(36)	PK, Unique, not null
caption	char(256)	Not null
description	text	
photo	varchar(max)	File path
thumbnail	varchar(max)	File path
location	int	Related to location table
price	int	USD
creator_id	char(36)	FK, Not null
created_date	date	Generate automatically
deleted	bit	1:true/0:false
deleted_date	date	

◆ *Location*

id	int	PK, Sequence number, not null
location	char(36)	Not null

◆ *Comment*

id	char(36)	PK, Unique, Not null
text	varchar(max)	Not null
post_id	char(36)	FK, Not null
creator_id	char(36)	FK, Not null
created_date	date	Generate automatically

deleted	bit	1:true/0:false
deleted_date	date	

◆ *Message*

id	char(36)	PK, Unique, Not null
text	varchar(max)	Not null
from_id	char(36)	FK, Not null
to_id	char(36)	FK, Not null
creator_id	char(36)	FK, Not null
created_date	date	Generate automatically
deleted	bit	1:true/0:false
deleted_date	date	

◆ *Favorite (priority 2)*

id	char(36)	PK, Unique, not null
post_id	char(36)	FK, Not null
like	bit	1:true/0:false
saved	bit	1:true/0:false
match	bit	1:true/0:false
creator_id	char(36)	FK, Not null
created_date	date	Generate automatically
deleted	bit	1:true/0:false
deleted_date	date	

◆ *Review (priority 2)*

id	char(36)	PK, Unique, not null
text	varchar(max)	Not null
star	int	0~5, not null
post_id	char(36)	FK, Not null
creator_id	char(36)	FK, Not null
created_date	date	Generate automatically
deleted	bit	1:true/0:false
deleted_date	date	

◆ *Notification (priority 2)*

id	char(36)	PK, Unique, not null
text	varchar(max)	Not null
from	char(256)	
from_id	char(36)	FK
to_id	char(36)	FK
created_date	date	Generate automatically
deleted	bit	1:true/0:false
deleted_date	date	

◆ *SearchText (priority 3)*

id	char(36)	PK, Unique, not null
text	char(256)	Not null

frequency	int	Not null
created_date	date	Generate automatically
deleted	bit	1:true/0:false
deleted_date	date	

→ Schema EER diagram
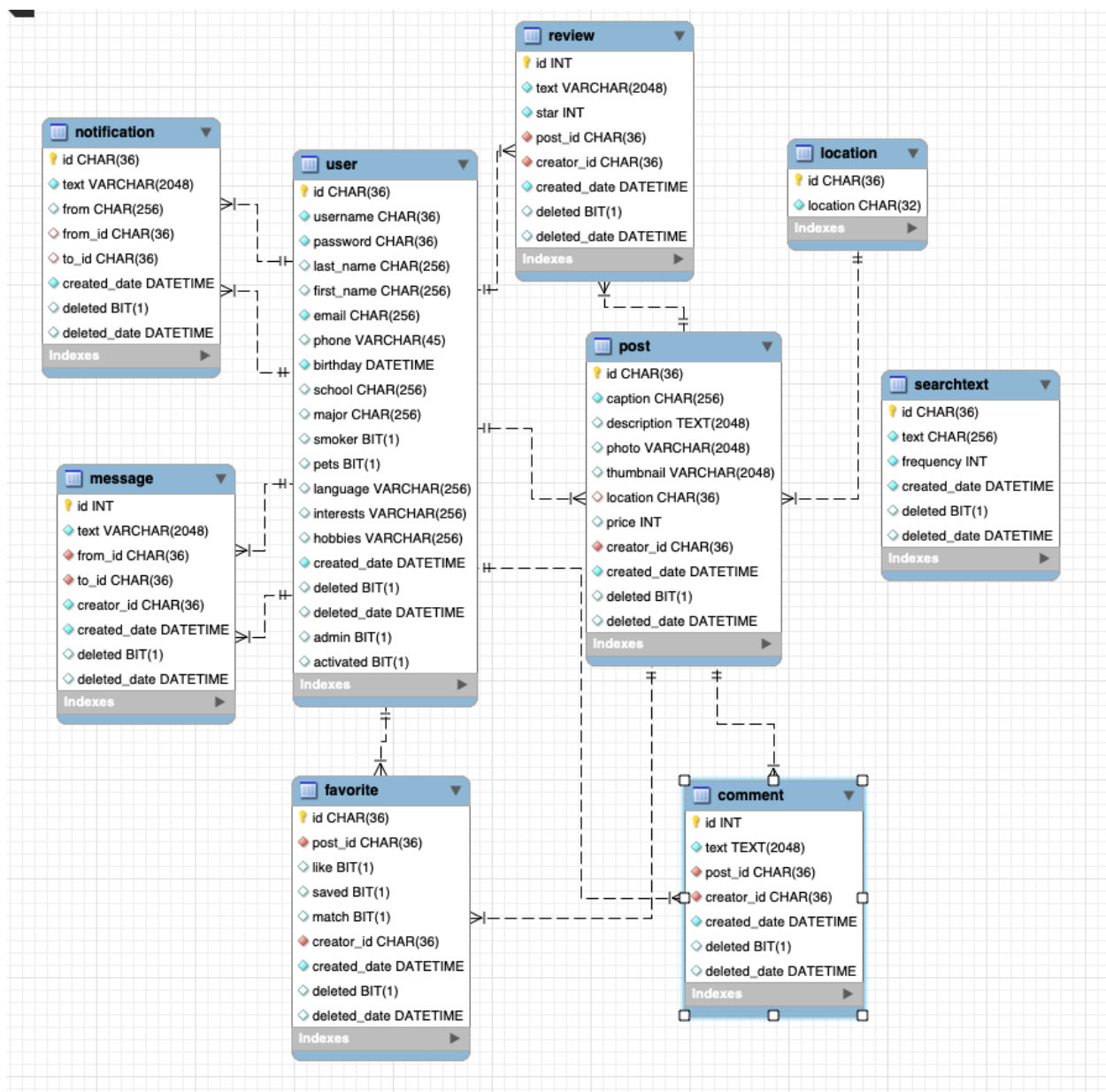
➜ Media Storage

Images and video/audio will be kept in file systems
Images: JPEG-size is no more than 3 Mb. GIF-size is no more than 10Mb
Video: MP4, size is no more than 10 Mb
No Audio in the website

➜ Search/filter architecture and implementation

Split search text and use %like to do fuzzy search, use filter to search

For example:
Search text: "California kitchen", filter: price < 1000, location: SF(front-end will request SF's id in location table), order by: newest
Sql SELECT:
SELECT * FROM post
WHERE CONCAT(caption, ' ', description) LIKE '%California%'
AND   CONCAT(caption, ' ', description) LIKE '%kitchen%'
AND   price < 1000
AND   location = SF's id
ORDER BY created_date DESC

➜ APIs

Our website will contain our own API with the path "/home" or "/" as the default. This path will direct the user to the home page which acts as a hub containing a navigation bar which will include several hyperlinks to different paths within our website. For example, one of the links will lead to the path "/about" which includes more information about the team behind the website. This navigation bar will be found at the top of each page in the website to allow for easy functionality throughout the site. There will also be a search bar created in order to allow the user to be able to search for certain keywords that they will need to find easier, such as name, city, and price.

➜ On-Trivial Algorithm or Process

Our team plans to hopefully include a sorting process which allows listings to be displayed and prioritized in a certain way. For example, if the user wishes

to view the listings of potential roommates with the best reviews or a profile with the most comments, the user can select either sorting methods through a dropbox menu.

➜ SW Tools and Frameworks

Server-side Technologies:

- ◆ Server Provider - AWS (RAM 1gb, vCPU 1 2.5 GHz)
- ◆ OS - Linux Ubuntu Server 20.04 LTS
- ◆ Database - MySQL 8.0
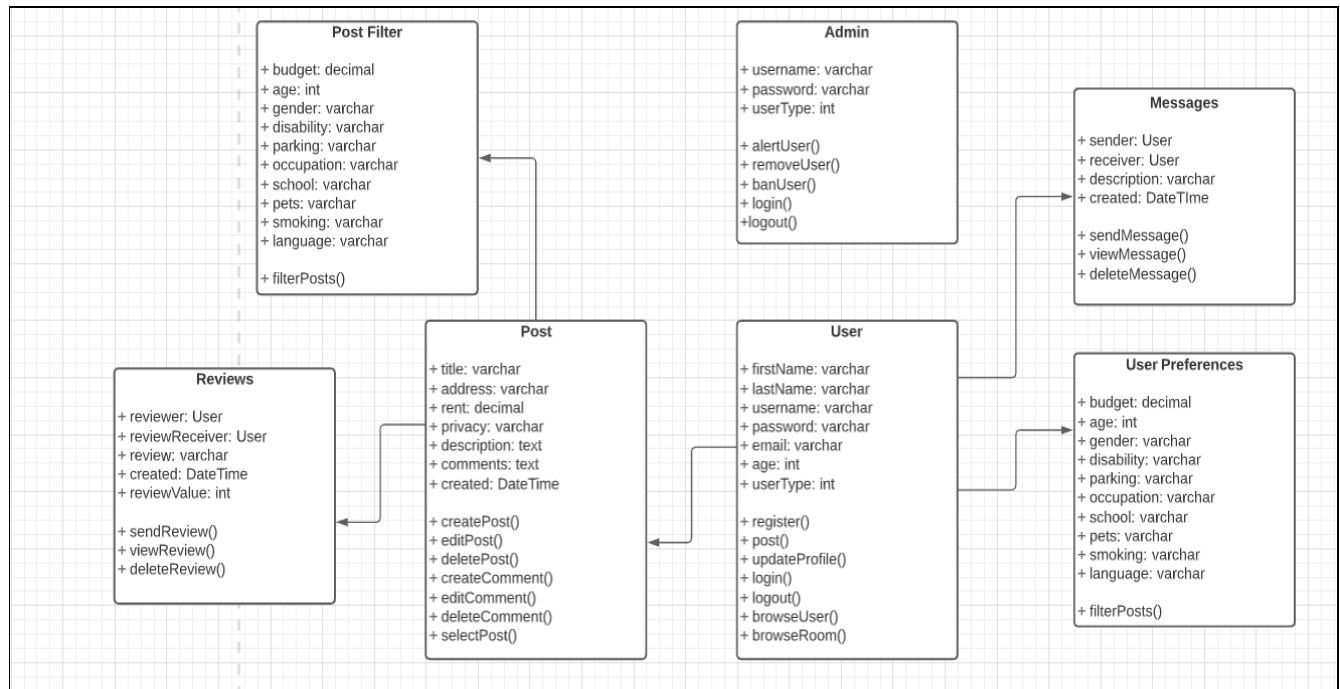
Additional Tools:

- ◆ Frontend - HTML, CSS, Javascript ES2015
- ◆ Backend - Javascript ES2015
- ◆ IDE - Visual Studio Code
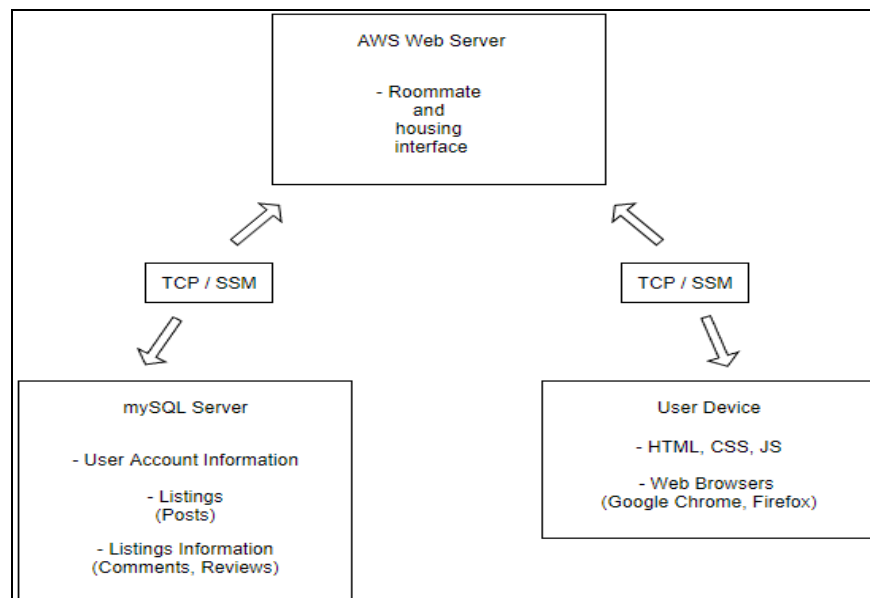- ◆ Javascript Library - React.js, Node.js

Targeted Browsers:

- ◆ Google Chrome
- ◆ Mozilla Firefox
- ◆ Internet Explorer

4 High Level UML Diagrams

➔ High-level UML Class Diagram

➔ UML Component and Deployment Diagrams

5   Key Risks For The Project

➜ Skills risks

An obvious skill risk that we believe is present not just in our team, but in most teams, is the difference in level of skill each member holds. Due to the diverse set of skills that the team members possess, the risk is failing to work efficiently due to varying knowledge in certain technologies that our project requires and being burdensome in the team's progress.

How we plan on working on resolving this risk is by focusing on each team member's strength and resolving problems and issues faced regarding the project according to those strengths. This way, we are able to better coordinate the team and work more efficiently. This coordination should be done by being transparent with team lead and team in general about skills that each team member possesses, and assigning tasks accordingly.

➜ Schedule risks

With the current conditions surrounding the world, with COVID-19 still being very apparent in our communities and many colleagues choosing to study from overseas in different time zones, timing conflicts are sure to be a risk in any project that requires group work. It is no different in our case with team members studying remotely and a member being overseas. So it is important to create a schedule where meaningful progress can be made to keep up with the time frame of a given assignment.

We plan to tackle this by breaking down the tasks for each subgroup (front-end, back-end) and having them schedule meetings that create no conflict with their time and align with the group's overall time schedule. That way, different parts of a problem that need to be solved, get solved as a joint effort from all subteams. We also agree on a specific time at least 2 days of the week to gather together and provide updates on our progress, keeping the rest of the subgroups up to date with the latest changes.

➜ Technical risks

Without a doubt, one major technical risk that may arise throughout the development of the project is incompatibility of code amongst the different subteams or members. As members work through the high level

specification our project requires, they may implement a functionality in the code that is not compatible with another member's work, therefore this type of risk would cause long delays in the development and rise conflicts in the overall functionality of the product. Another technical risk that needs to be solved is keeping the product up to date with the latest significant changes to the code.

First risk can be addressed by having frequently occurring meetings where each team member and subteam collectively updates the rest of the team on the updates they have done to the code and explains the significance and nature of the updates so the team is able to coordinate accordingly. The second risk can be resolved by making sure that the active website is displaying the latest version of the code found on team github.

➜ Teamwork risks

Teamwork risks in our case mainly tie in with scheduling risks due to the current climate of the world. Being that the entire team is working remotely at the moment, some risks that may arise relating to teamwork are team members not being able to collaborate comfortably in person and therefore resulting in mixed online meeting availability. The collaboration being fully remote imposes a risk that may lead to the team not collaborating efficiently therefore, also arising the risk of not meeting deadline requirements.

We plan to work around this issue by improving all other risk categories so that there are very little hiccups along the way of the development cycle. Avoiding the other shortcomings that may arise, we decrease the chances of teamwork risks occurring. To directly combat any teamwork risks, we update the team of any problems that an individual team member might have so we are able to combat these risks before it imposes a serious threat to the completion of the tasks in hand.

➜ Legal/content risks

One major content risk that we may face in the future of the development of our product is content posted by users on the platform. Although our product is directed to the niche of housing, we cannot control what user's post. The content the users post onto the platform can pose a risk in terms of copyrighted or illegal content. In terms of software and content that we

need to properly develop the product, at the current moment, we face little to no risk as most of the tools we will be using are free to use and will not be used for any monetary gains.

What we plan to include to combat the scenarios of users posting harmful/copyrighted/illegal content on our platform is to have disclaimer on post screen about content that is not allowed on the platform and a filtering system where a team member will be able to act as admin and approve/disapprove of content that is deemed illegal/copyrighted/harmful. Such a system will provide a safety net for any occurrences where the team might face a legal/content risk.

6 Project Management

From the very start, RUMIs development team has taken an organized "divide and conquer" approach to designing, developing, and maintaining documentation of the RUMI app. Our team has identified two key areas in the development plan for RUMI and allocated members of our team to the correct sub team based on skill sets and levels of experience. The development team has divided itself into front & back-end sub teams allowing for the engineers to develop software aligning best with their skill sets and experience To optimize our time and produce the best results, we've also assigned a lead to each sub team, a GitHub lead, documentation expert, and a team lead to manage our overall project development cycles over JIRA.
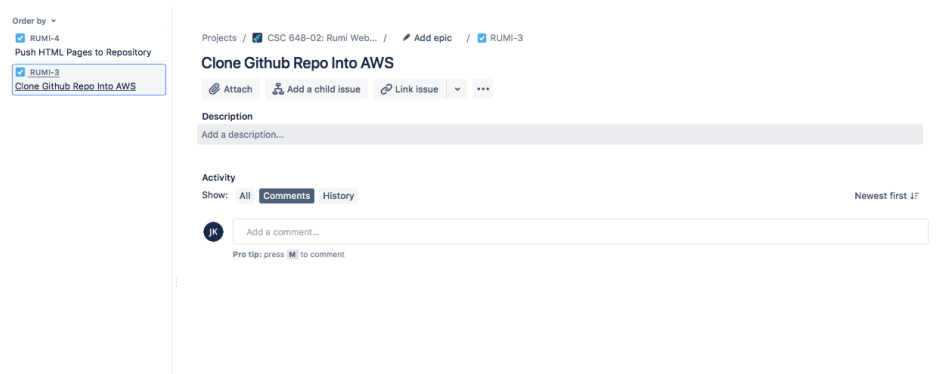
JIRA has allowed our team to identify key tasks to be completed and assign each to the respective teams and/or individual. Each JIRA ticket can be viewed within the project dashboard with an assigned start and due date, individuals assigned, assignees, and priority level. In the following screenshot, we can see an example of one of the earliest tasks assigned to members of our team, pushing our individual HTML pages to the github repo with the aforementioned details.
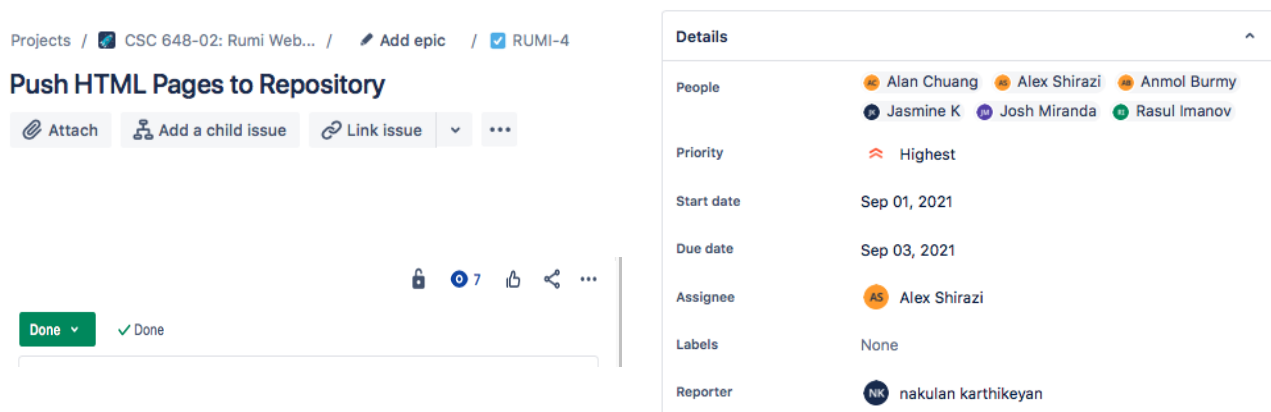
Our team has also taken a divide and conquer approach to M2 by allocating each section documentation to a team member and consolidating ideas during our hands-on meeting to resolve any and all consistency issues. Moving forward, we plan to continue using JIRA to assign tasks to members based on their technical and organizational role within. The dashboards functionality on JIRA allows us to monitor our progress respective to each sub team while the calendar functionality allows us to monitor progress respective to each deadline resembling a Gantt chart for project scheduling and increased diligence to ship all features on time.