San Francisco State University

Final Project Report for Software Engineering
CSC 648-848 Summer 2020

Team 03
GatorHub

Megha Babariya (mbabariya@mail.sfsu.edu) (Team Lead)
Abraham Zepeda (Github Master)
Nathalia Sainez (Front-end Lead)
Raymond Kuang (Back-end Lead)
Yaxin Deng
Tania Nemeth
Harsh Saxena

URL of the Project : http://54.151.14.7/views/index.php

Date : 07th August, 2020

2) Product Summary :

Our product, GatorHub, is a custom web application built exclusively for SFSU students and faculty to help connect them to the best housing. The following is an itemized list of all major committed functions. The functions described in this list will be delivered in our final project.

1. Search: Search listings based on different parameters such as zip code, property type, and whether a user is looking to rent or buy.
2. Filter: Filter results based on attributes such as distance to SFSU, minimum Price etc.
3. Google Maps API: Showcases distance from the estate to SFSU campus.
4. Registration: Users shall be able to register for an account.
5. Login/Logout: Users shall be able to to login and logout of their account.
6. Compose a Listing: Users shall be able to post a listing.
7. Edit Listing: Users shall be able to edit or remove their listing.
8. Display Listing: When a user clicks a listing, it will direct the user to a full view of that listing.
9. Contact Listing Owner: Ability to contact seller via button on listing.
10. Admin: Admin approval is required when a user wants to add a listing.

    Unique Feature : Our web application was exclusively for SFSU students and faculties without any hustle.

    URL of the Project : http://54.151.14.7/views/index.php

3) Milestone Documents (ML1- ML4) :

Milestone Document 1 :

San Francisco State University

Software Engineering

CSC 648-848 SUMMER 2020

GatorHub

Milestone 01

Team 03

Megha Babariya (mbabariya@mail.sfsu.edu) (Team Lead)

Abraham Zepeda (Github Master)

Harsh Saxena (Back-end Lead)

Nathalia Sainez (Front-end Lead)

Raymond Kuang

Yaxin Deng

Tania Nemeth

History Table:

Version No.	Date	Comments
01	06/25/2020	Initial Document
02	06/29/2020	Comments (by Professor)

		implemented

1. Executive Summary

Our team's vision is to develop a web application exclusive for San Francisco State students and faculty called GatorHub to help them find places to rent . We aim to help set our users up for success by designing robust services specifically with their needs in mind. We understand how much an unsound living environment can affect an individual's mental health or ability to focus. Our intent is to streamline the renting process in hopes to alleviate the stresses that burden students and faculty. Our mission is to present practical information about the rooming arrangements to help them find housing that best accommodates their needs.

To make our application competitive and attractive for SFSU faculty and students we will showcase each listings' distance to the San Francisco State campus using the GoogleAPI. We want to enable our users with a solid understanding of the locations that they're considering with respect to the school and other resources. By limiting our tenants to only students and faculty members of San Francisco State, we are able to offer custom features to assist users in finding compatible roommates. Users can message landlords to ask questions and get more information and even leave reviews. Our goal is to ensure that users on our website are equipped with all the tools to feel empowered to find the best location.

Our team possesses useful insight necessary to craft an effective solution as we are composed of current students at San Francisco State University. Unlike our competitors, our application is designed with our users in mind from the beginning to the end. We know firsthand what the pain points are to get to and from the San Francisco State campus and we've learned how to overcome them. Our desire is to develop a simple, easy to use application with an exceptional user experience. GatorHub will not only impact individuals; it will also foster a greater sense of community at San Francisco State.

2. Personae and main Use Cases

2.1 Key Personas:

2.1.1 Tenants (Buyer) :

Eg: Eric

- Shifting to San Francisco from Texas
- Looking for affordable homes
- Concerned with price
- Willing to get a user-friendly website to quickly check for some good places to rent.

Most of the users for renting homes are SFSU students (tenants) or some professors. They have low to intermediate knowledge of web browsing due to advanced and fast developing technologies used by the websites. They are in search of homes at a good affordable range due to lack of income and high prices. So, the main task of the tenants is to find home at an inexpensive price and close to their workplace.

2.1.2 Landlord (Agent/Seller) :

Eg : Sarah

- Sells a home
- Willing to rent the place
- Wants a reliable site to get good tenants
- Very Busy
- Doesn't want to spend much time looking for tenants herself

As Landlords, they want to rent their houses asap. They may or may not be tech-savvy. Don't want to go about physically finding tenants or paying high fees to real estate management firms to get the tenants.

2.1.3 Admin :

Eg: John

- Needs to approve postings from the landlords
- Manages the database
- Handle issues
- Maintain updated information

An admin is responsible for secure maintenance and updating of all data. They have relevant skills to handle complex UI issues or database maintenance. They are also responsible for troubleshooting issues that the tenants or landlords face. All the supervising responsibilities are managed by the Admin department.

2.2 Use Cases:

2.2.1

Eric (Tenant)(Searching for home):

- Desires an affordable home in a desirable neighborhood, ideally within a 25 minute walk or commute to SFSU, with other students in the house. Eric wants to minimize use of his car, so he is seeking a rental that would allow him to walk or bike to the grocery store and some bars or restaurants. He is forced to look from his home in Texas rather than in person so he needs a tool that can give him information about location and the rental itself. He utilizes the search tool, filtering for his desired price range and move-in date. Eric finds the Google Maps API extremely

helpful, as it allows him to explore the areas around each property without being there. He can explore bus routes and MUNI stops near the home, as well as map to the nearest grocery store. He also quickly checks the questions and answers section of each rental, to see if there's any information he would find helpful, before messaging the landlord. He favorites each listing he likes so he can easily look back and keep track of potential homes. Whenever Eric favorites a home, if other SFSU students have expressed interest, he will be notified and have the option to view their profiles. If he thinks someone would make a good roommate/housemate, he has the option to reach out via direct messaging. Eric loves that he can quickly check for new listings daily and eliminate the ones he doesn't like.

2.2.2

Sarah (Landlord) (Renting out Properties):

- Just bought a new home and she's looking for potential tenants to lease the space. Sarah needs one integrated platform where she can easily list her home, filter out tenants, communicate with tenants, keep track of interest in her property and answer questions. She doesn't have a lot of free time and after doing some research, finds our site is reliable and easy to use. Sarah creates a rental listing for her property with all the information (availability, price range, location on Google Maps, floor space, parking, etc.) and waits for a message. She loves that she can easily edit the listing and answer questions asked by potential renters on the Q&A section. Sarah also loves that our platform provides a lot of transparency and helps her avoid going through a company to lease her home. She depends on the direct messaging aspect of the platform, as she hates getting Craigslist emails and text messages and wants to keep all communications with potential tenants in one area. An aggregated, easy to navigate platform is crucial, as it assists her with keeping track of who is interested, what they've offered and if their profile indicates they would be a good fit. Sarah can also easily track her communication between prospective tenants, without unnecessary stress or switching between multiple websites.

2.2.3

John (Admin) (Managing content, giving permissions and handling data) :

- John is a skilled worker who handles site maintenance for GatorHub and updates for the website. The admin functions built into the application allow him to monitor all content and postings on the site and take action if anything needs to be modified or removed. John frequently manages users and property listings to make sure all content has a place on the site and follows community guidelines. When John sees a user posting inappropriate comments or violating any rules, he decides whether to ban them/their ip address temporarily or permanently. John also takes down scam listings and suspect user profiles, as well a direct line of communication with registered users.

1. Administrator can manage Us
2. Administrator can manage properties
3. Agent (Seller) create property listing
4. Buyer send message to agent
5. User (Buyer/Agent/Anonymous User) can search and list properties (Available for Sell and Rent)
6. User (Buyer/Agent/Anonymous User) can Join/Register
7. User (Buyer/Agent/Anonymous User) can sign-in

3. List of Main Data Items and Entities :

The GatorHub Application is supported by MySQL and its database. MySQL Server provides a good response time of the data being stored making the search effective, convenient way for storing the photographs of the properties and storing the entire description and features of the Property Listings.

1. Registered User :
    1.1. Buyer : A Tenant or Purchaser
        - Will be able to view the property listings, but need to register for being able to see images of the apartment. This user can contact the Seller Agent to buy or rent a property.
    1.2. Agent : A person representing from Landlord to Rent or Sale properties
        - Will be posting property details and pricing, will need approval from Admin to post the properties on Rent or Sale.
    1.3. Admin : A person managing website, users, and its content
        - Responsible for providing permissions to Buyer, Agent, and Registered User. Would be handling all the accounts and listings of the property.

2. Unregistered User :
    2.1. It is an anonymous user who is visiting the website and didn't register and login.
    2.2. An unregistered user will not be able to contact the landlord.
    2.3. Unregistered users will again have the option to create an account as a Landlord, an Agent or a tenant.

3. Property:
    3.1. A house or office or apartment available for Rent or Sale. The property will have its features with image, prices, address, and posted by an Agent.
    3.2. The property will also show the distance from SFSU.

3.3. The property listing will also have the landlord ID for each Landlord.

4. Property Image:
   4.1. Each property will have zero or many images for its features such as Bed room, Kitchen, Bathroom etc.
   4.2. Each property image will be associated with the listing.

5. Message:
   5.1. The messages communication between buyer and agent regarding property sale or rent.
   5.2. It consists of a message id, body, timestamp value.

6. Favourite Listing :
   6.1. Tenants will be able to list the properties as favourite for future reference.

4. Initial List of Functional Requirements

Unregistered User Functions:

1. Search – Users shall be able to search for real estate.
2. Sort – Sort search results based on parameters such as distance and price.
3. Google Maps API – Users shall be able to view the listing on Google Maps within the webpage.
4. Social Media – Display buttons that users can click to post on their social media (Reddit).
5. Registration and Profiles – Users shall create an account to store data. Data includes personal information such as age, ethnicity, address, contact, and favorites..

Registered User Functions:

1. Login/Logout –Users shall be able to to login and logout of their account.
2. Posting – Users shall be able to post a listing.
3. Edit Posting – Users shall be able to edit or remove their listing.
4. Questions/Answers – Users shall have a section on each listing to post questions. Listing owners shall be able to answer user questions.
5. Favorites – Users shall be able to save listings for later access.
6. Logging – Display how many visits a listing has.
7. Report/Flag – Users shall have an option to report a listing on the page. Reasons include price gouging and false advertising.
8. Internal Direct Messaging – Users shall communicate with each other via direct (private) messaging.
9. Roommate Finder – Users shall be able to search for roommates based on multiple parameters.
10. Audit History – Changes to a listing shall be documented and accessible to users.
11. Ratings for listing and owner – Users shall be able to rate a listing and the owner.

Admin Functions:

1. Has permissions to remove listings, content on its page, and user accounts.
2. Administrator approval is required for listings before they can be visible to users.

5. List of Non-functional requirements

1. Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0 (some may be provided in the class, some may be chosen by the student team but all tools and servers have to be approved by class CTO).
2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers
3. Selected application functions must render well on mobile devices
4. Data shall be stored in the team's chosen database technology on the team's deployment server.
5. No more than 50 concurrent users shall be accessing the application at any time
6. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.
7. The language used shall be English (no localization needed)
8. Application shall be very easy to use and intuitive.
9. Google analytics shall be used
10. No email clients shall be allowed. Interested users can only message to sellers via in-site messaging. One round of messaging (from user to seller) is enough for this application
11. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.
12. Site security: basic best practices shall be applied (as covered in the class) for main data items
13. Media formats shall be standard as used in the market today
14. Modern SE processes and practices shall be used as specified in the class, including

collaborative and continuous SW development

15. The website shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Summer 2020. For Demonstration Only" at the top of the WWW page. (Important so as to not confuse this with a real application).

6. Competitive Analysis

Features	Zillow	Rent.com	Realtor	**GatorHub**
Search bar & Filters (Price, Location)	+	+	+	+
Map (& distance to SFSU)	+	-	+	++
Roommate Finder	-	-	-	+
Contact agent/owner information	++	+	+	+
Can be added to favorites	-	-	-	+
Reviews from users	+	+	+	+
UI design and ease of use	++	-	+	++

+ feature exists ++ superior feature - feature does not exist

Our product will provide SFSU students with an actual place to rent apartments seamlessly that will incorporate both **Distance** from the rental place to SFSU, as well as an option to search for other potential SFSU roommates through **Roommate Finder**. This is in contrast to other online rental places such as Zillo or Realtor, which both focus on the general public and do not provide distances to/from a specific place. The unique feature of our site, and what gives us competitive advantage over the others, is being able to tap into the SFSU niche market through Roommate Finder. Roommate Finder allows an SF State Student to rent a room with other SF State Students. By using Roommate Finder, a Student can even share the rooms with another student pursuing the same major in SFSU. Also, another feature giving us a good

advantage vs our competitors involves being able to **Internally Direct Message** the renter for further piece of mind when doing business. This messaging system is internal and no email server is necessary.

Instead of focusing on providing rental services for the general public like the rest of our competition, we only focus on SFSU students with @sfsu.edu verified emails. This is our niche market and what gives us competitive advantage and differentiation amongst the others. All the competitors referenced above provide countless types of filters for home searching such as number of bedrooms, bathrooms, by neighborhood, and many other criteria. But in our plan, the search result will be displayed on both map and in a list side by side by default. This will give our users a better overview of the area they are searching in. Like all of our competitors, we provide our registered users with the option to favorite the property they are interested in to narrow down their potential choices. Our biggest advantage is in the ease of use that will be achieved by keeping a clean and minimalistic look that focuses on the guest.

7. High-level system Architecture and Technologies used

The Real Estate Web Application is built using a layered architecture where the total functionality can be divided into layers having different functionalities. The main layers include the database access layer, business logic layer and the presentation layer. Thus, this application follows the 3-Tier Architecture.

High-level system Architecture Design:

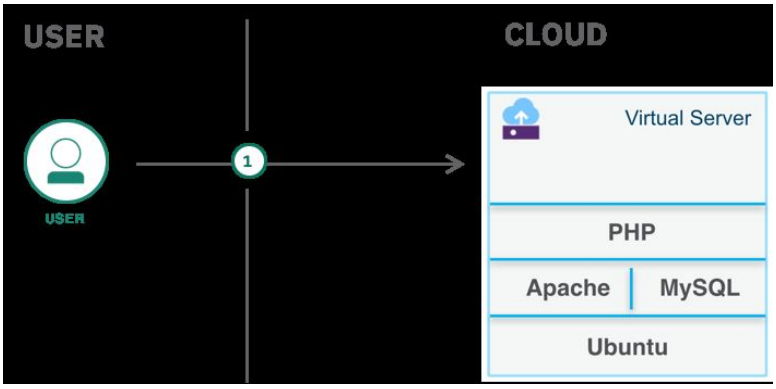


Image 1. High-level design

Architectural Design for the Real Estate Web Application which represents a three tier architecture comprising the Presentation Tier: list of web pages planned to be implemented in the application, the Middle Tier: class diagrams and description of each class and the Data Tier: comprising the ER diagram and database schema.

Tools and Technologies:

1. Server Host : Amazon Web Services (AWS-Free Tier EC2 Instance)
2. Operating System: Linux : Ubuntu 16.04 Server
3. Database : MySQL 5.6
4. Web Server: Apache HTTP WS
5. Server-Side Language: PHP 7.4
6. Frontend : HTML5, BootStrap, JQuery, CSS, JavaScript
7. IDE: PHPStorm (Community Edition),
8. SQL Editor: MySQL Workbench (Community Edition)
9. SSH : Putty/Terminal
10. FTP : FileZilla (Free Version)
11. Google Docs: Functional Requirement, Technical Design
12. Repository : Github

8. Team and Roles

1	Team Lead	Megha Babariya
2	Front End Lead	Nathalia
3	Back End Lead	Harsh
4	Github Master	Abraham
5	Front End Member	Yaxin
6	Back End Member	Tania
7	Back End Member	Raymond

9. Checklist

9.1	So far all team members are engaged and attending zoom sessions when required	On Track
9.2	Team found a time slot to meet outside of the class	Done
9.3	Github master chosen	Done
9.4	Team decided and agreed together on using the listed SW tools and deployment server	Done
9.5	Team ready and able to use the chosen back and frontend frameworks and those who need to learn are working on learning and practicing	On Track
9.6	Team Lead ensured that all team members read the final M1 and agree/understand it before submission	OK
9.7	Github organized as discussed in class(eg: master branch, development branch, folder for milestone documents etc)	Done

Milestone Document 2:

San Francisco State University

Software Engineering

CSC 648-848 SUMMER 2020

GatorHub

Milestone 02

Team 03

Megha Babariya (mbabariya@mail.sfsu.edu) (Team Lead)

Abraham Zepeda (Github Master)

Harsh Saxena (Back-end Lead)

Nathalia Sainez (Front-end Lead)

Raymond Kuang

Yaxin Deng

Tania Nemeth

History Table:

Version No.	Date	Comments
01	07/08/2020	Initial Document
02	07/10/2020	Updated document

1. Functional Requirements :

Priority 1 (Must-have):

Unregistered User Functions:

1. Search – Users shall be able to search for real estate.
2. Sort and filter – Sort search results based on parameters such as distance and price.
3. Registration and Profiles – Users shall create an account to store data. Data includes personal information such as age, ethnicity, address, contact, and favorites.

Registered User Functions:

1. Login/Logout –Users shall be able to login and logout of their account.
2. Posting – Users shall be able to post a listing via a unique sys id.
3. Edit Posting – Users shall be able to edit or remove their listing.
4. Contact Landlord - User shall be able to contact landlord by getting the Landlord's email id.

Admin Functions:

5. Has permissions to remove listings, content on its page, and user accounts.
6. Administrator approval is required for listings before they can be visible to users.

Unregistered User Functions.

1. Google Maps API – Users shall be able to view the listing on Google Maps within the webpage.

Priority 2 (Desired):

Registered User Functions:

1. Questions/Answers – Users shall have a section on each listing to post questions. Listing owners shall be able to answer user questions.

2. Favorites – Users shall be able to save listings for later access.

Priority 3 (Opportunistic):

Unregistered User Functions:

1. Social Media – Display buttons that users can click to post on their social media (Reddit).

Registered User Functions:

1. Roommate Finder – Users shall be able to search for roommates based on multiple parameters.
2. Audit History – Changes to a listing shall be documented and accessible to users.
3. Internal Direct Messaging – Users shall communicate with each other via direct (private) messaging.
4. Logging – Display how many visits a listing has.
5. Report/Flag – Users shall have an option to report a listing on the page. Reasons include price gouging and false advertising.
6. Ratings for listing and owner – Users shall be able to rate a listing and the owner.

1. List of Main Data Items and Entities :

Users

1. Unregistered User - Will be able to view listings but unable to make question posts on listings nor directly message other users.
2. Registered User – Will be able to view listings, flag listings, make posts, write direct messages, customize their profile and search for roommates.

Admin

1. Will be able to approve the listings.
2. Will be responsible to edit, delete listings and has all necessary permissions.

Listings

1. Listings will include a description, the main column for which users can initially search for, and be defined by attributes such as price, rooms, address, proximity to SF State.
2. Each Listing is uniquely defined by a 32-length integer primary key
3. The listing_images table, will contain a directory path to an image. Each row will contain a unique identifier, date created, and a foreign key pointing to its corresponding listing's primary key.
4. Listings will have other attributes like created_date, updated_date etc.

Messages

1. Messages will include a unique identifier, it will be linked for each property.
2. Messages will have a body, timestamp, and user_id pointing to the author's primary key.

Approvals

1. The administrator will be able to log into his account and in his admin dashboard see a list of potential listings that he can either approve or reject. When either action is taken, the landlord posting the listing will be notified of successful inclusion into the GarorHub listings database.

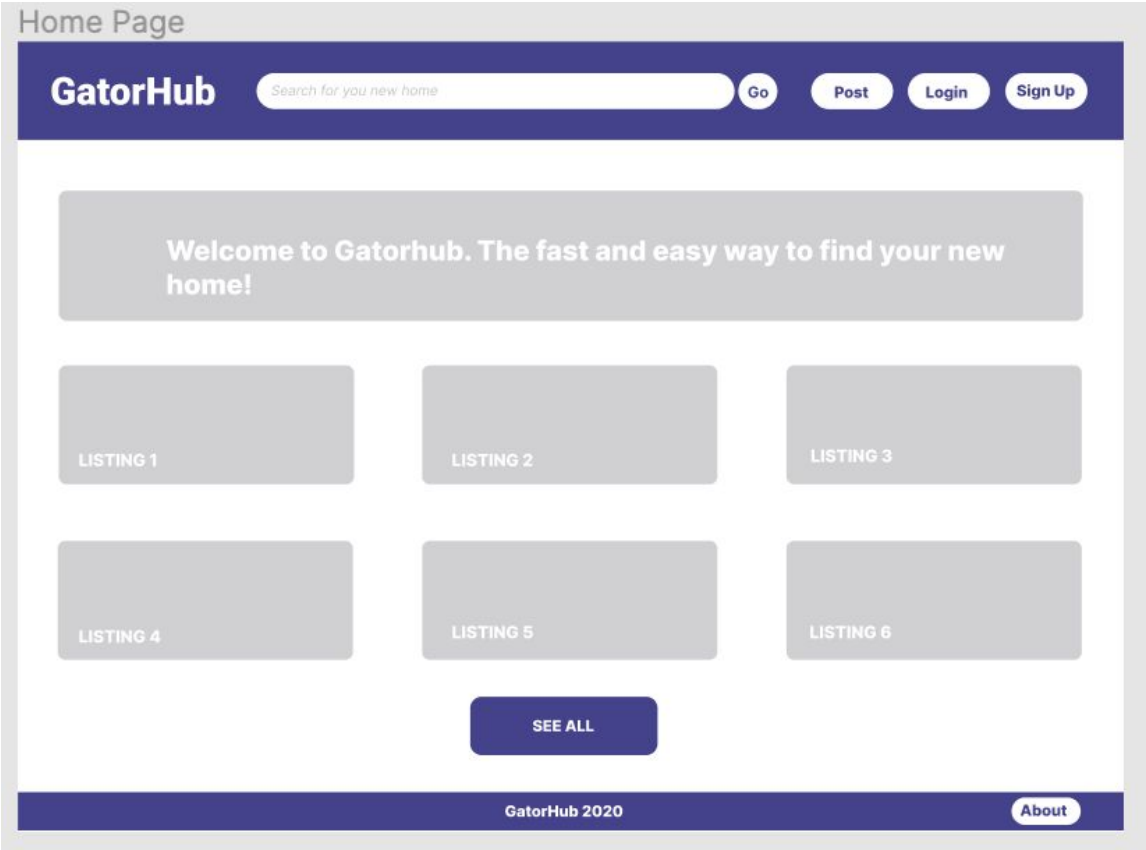
Roommate Finder

1. As a special feature that differentiates our product, a user will be able to register under certain tags denoting his gender, major, and interests, to help him potentially find similar roommates with similar assigned tags/characteristics. Only registered users who choose to be visible can be searched.

2. UI Mockups and Storyboards :

Home page

1. Upon reaching the home page, anybody can look at the six most recent listing thumbnails as well as use the search bar header feature to type keywords or tags that they are looking for in an apartment.
2. Using the search bar, or clicking “See All” will display a search results page with all the listings where filtering and sorting can be applied. See “Results Page” below.
3. The homepage will have our product logo on the left of the header, search bar, product information disclaimer, six recent postings, footer with “About” us link, as well as three buttons: “Post” apartment button, “Login” user button, and “Signup” button.
4. The footer’s “About” link goes to our team page profile that we created in M0.

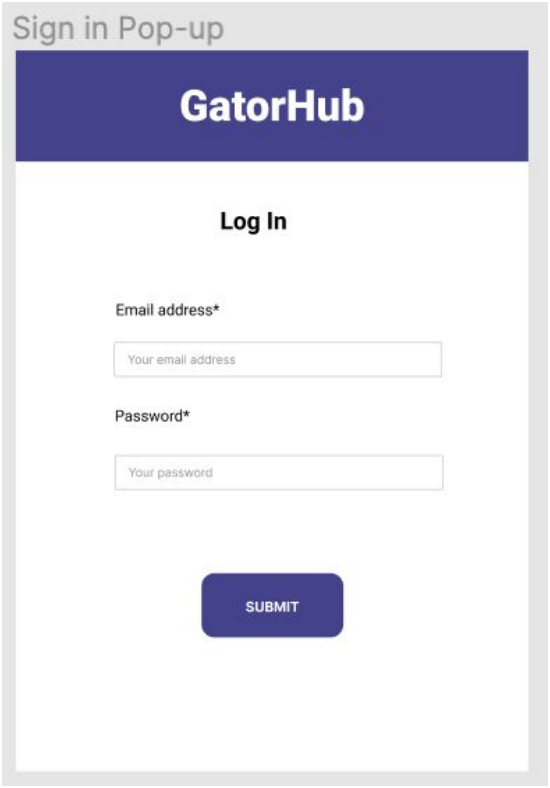


Post button

1. Clicking this button on the home page will launch the Login pop-up where Landlords can sign into their Dashboard to manage their current postings or put up a new property for rent.

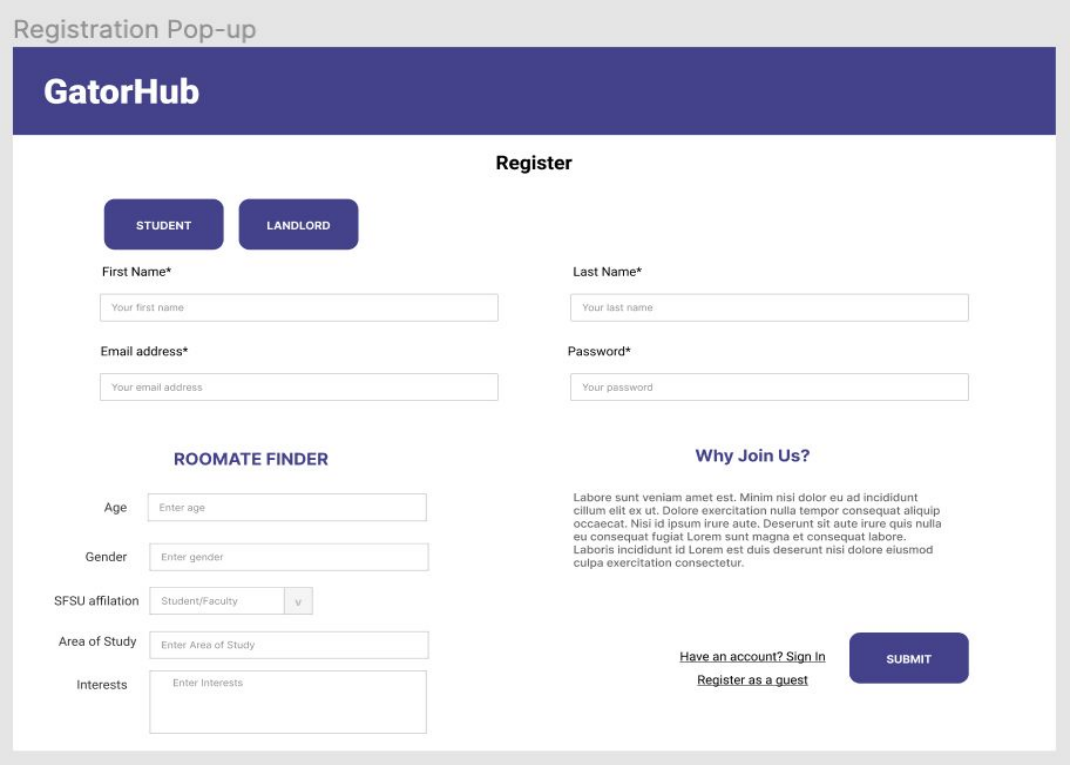
Login button

1. Clicking this button on the homepage will launch the Login pop-up where both Landlords and Students can sign into their accounts. If the user has previously registered as a landlord, his Dashboard will be displayed once he is signed up.



Sign Up button

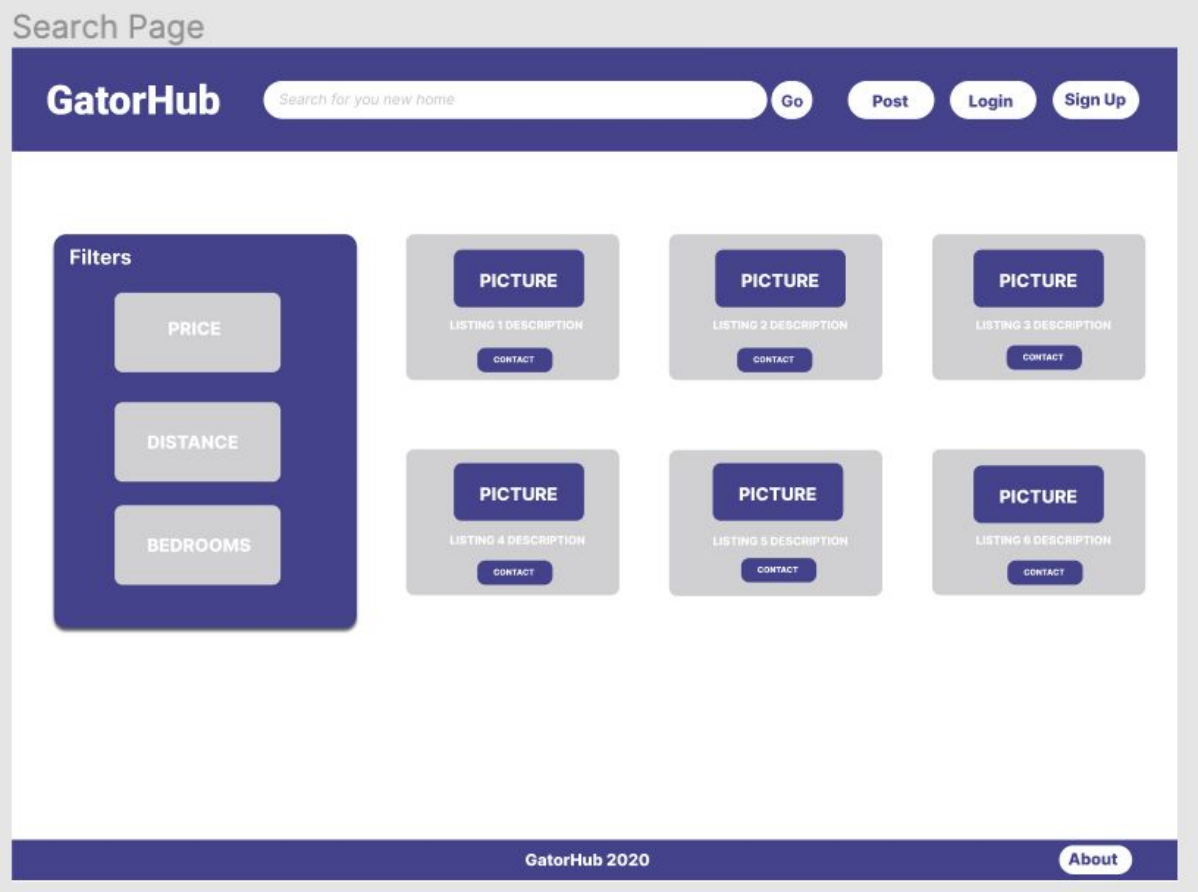
1. Clicking this button will launch the signup page where the user can register as a student, landlord, or guest. There will be something about why joining our site is so special as well as being able to go to the Login page if the user already has an account.
2. Here, the **GUEST** user will also have a change to make an account.
3. The student will also be able to use the **ROOMMATE FINDER** feature of the site by filling out his relevant details.



Search results page

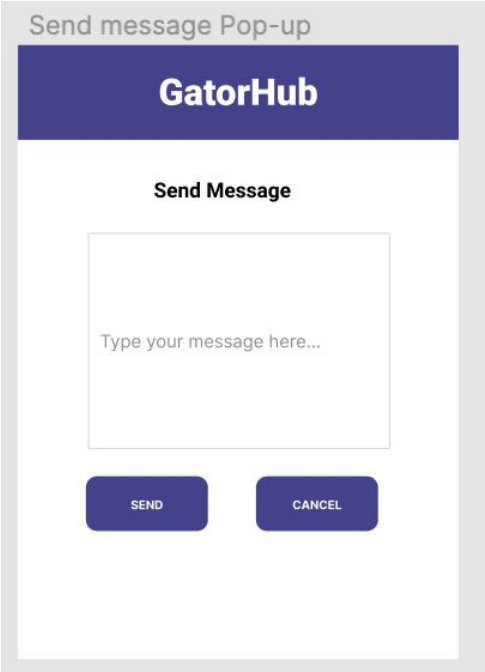
1. In the search results page, once the desired filters are applied, the thumbnails of each apartment will be displayed, but if the user clicks on the thumbnail, a prompt to login/register as a sfsu student or landlord (or guest) will be displayed. Otherwise the posting will not be able to be seen

in full detail. This site is for SFSU students and they need to have an @sfsu email to register as students.



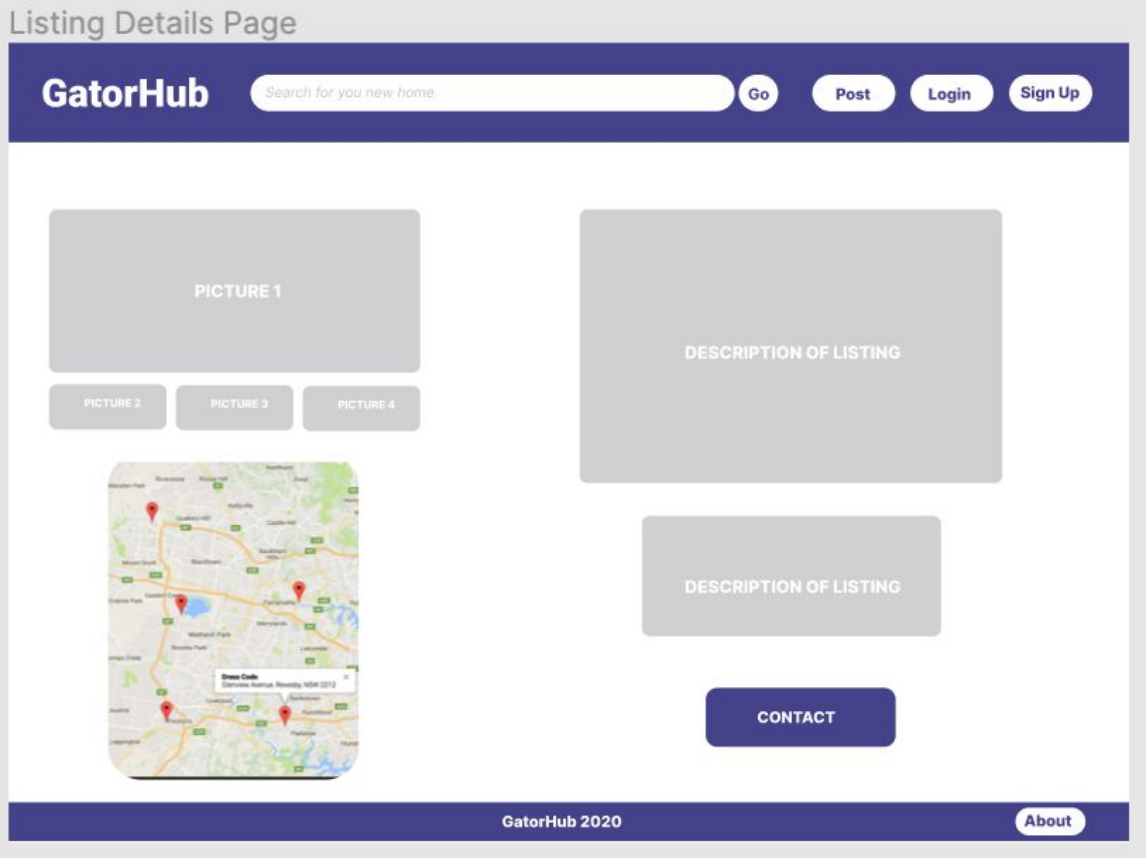
Send message pop-up

1. After clicking on the “Contact” button for the desired listing, a pop-up window will be displayed with an opportunity to message the landlord.



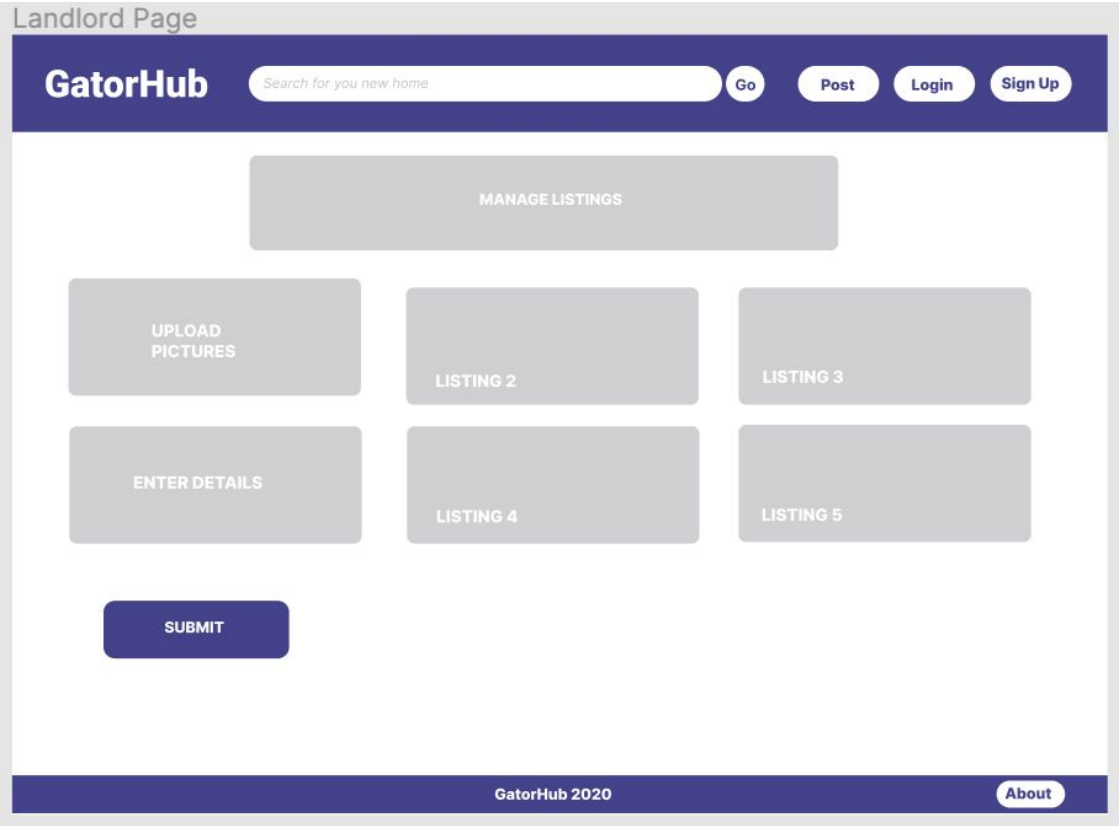
Listing Details page

1. Once the user clicks on a filtered listing, if he is logged in as an allowed user having a landlord account or an @sfsu email, he will be taken to the details of that posting.
2. There will be a main apartment picture with three smaller scrollable thumbnails below it.
3. There will be a Google map showing the driving, biking, and walking distance from the housing listing to the SFSU campus.
4. On the right side of the page, the details and tags of the listing will populate and there will be an opportunity to **DIRECT MESSAGE** the landlord through the “Contact” button.



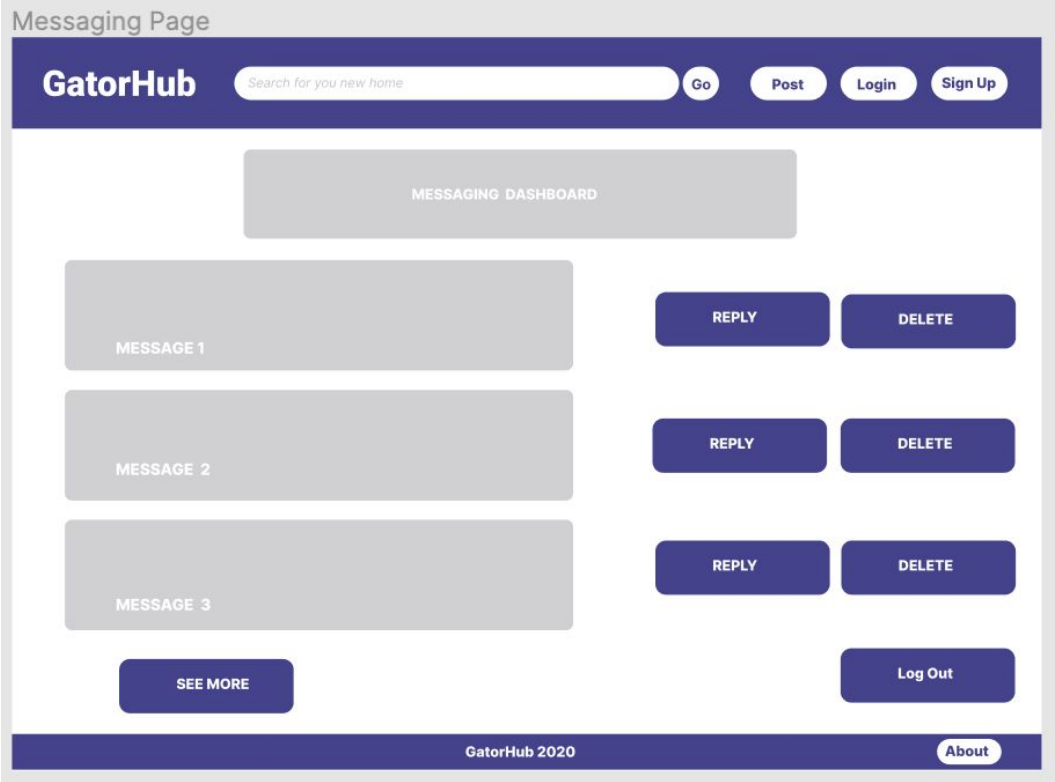
Landlord page

1. If the user is a landlord, his dashboard with his current listings as well as new listing opportunities will be displayed.



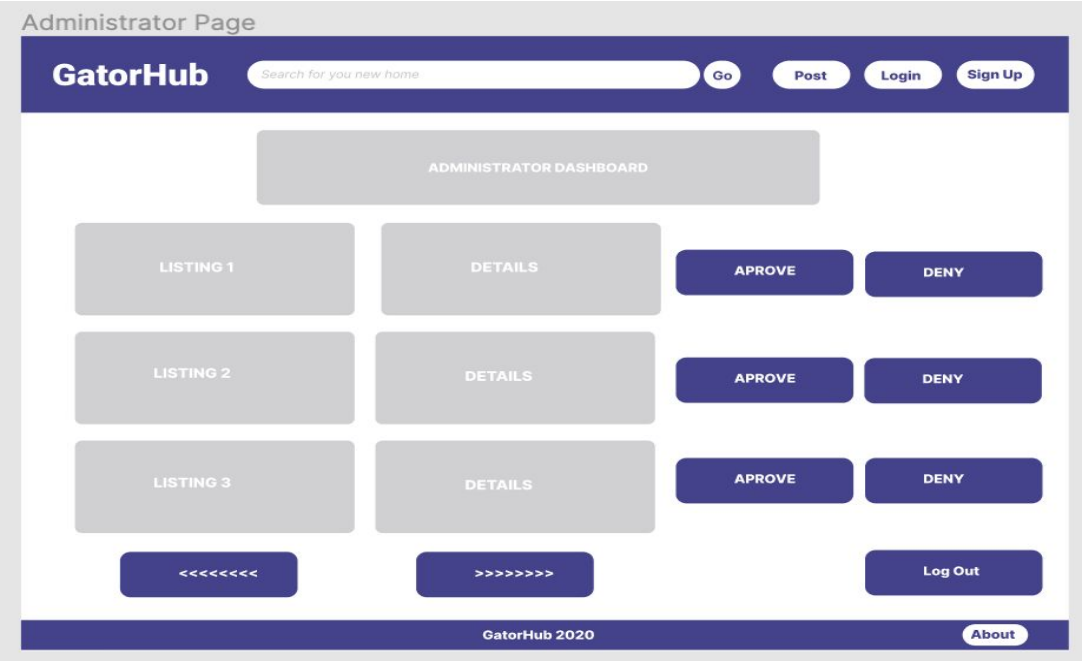
Messaging page

1. The messaging dashboard can be accessed to read and reply to internal messages



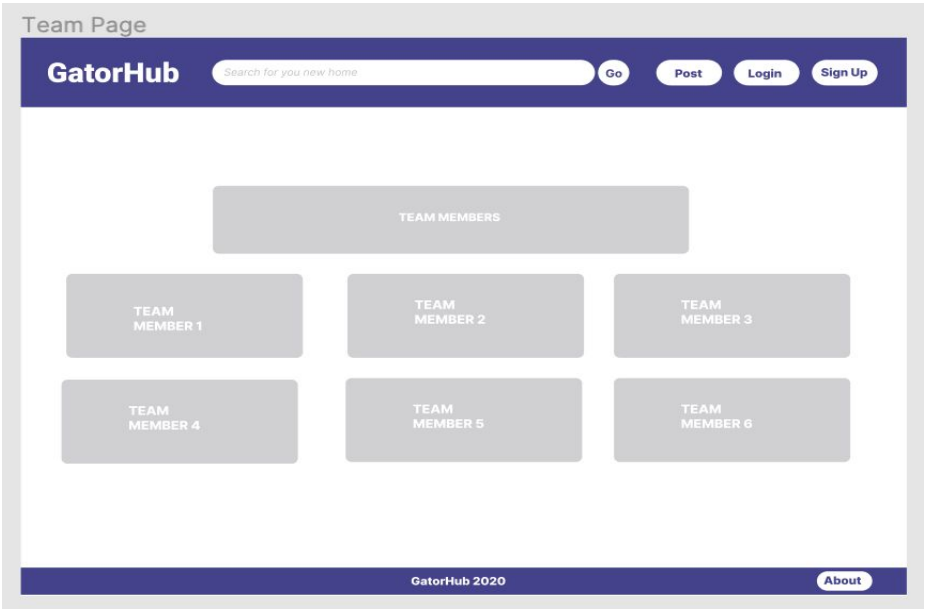
Administrator page

1. If the user is the administrator, upon login in through the normal method, his admin dashboard will be displayed with the listings that he either needs to approve or reject.



About Us page

1. When somebody clicks on the “About” button on the site’s footer, the Team page will be displayed, showing the team members’s biographies and interests that we created in M0.



3. High Level Architecture, Database Organization :

Database organization

listing table	datatype	description
sys_id	VARCHAR(32)	unique 32-length integer identifier
active	TINYINT(1)	inactive = 0, active = 1 inactive when listing is sold
street_address	VARCHAR(45)	location of listing,
city	VARCHAR(45)	location of listing,
state	VARCHAR(45)	location of listing,
zip_code	VARCHAR(45)	location of listing,
created	DATETIME	populated with current time when record is created
expired	DATETIME	when listing expires
price	INT	price of listing, used together with payment_type
payment_type	ENUM	describes the type of sale (monthly rent, buy)

property_type	ENUM	describes property as either an apartment, studio, condominium, or house
utilities	SET	utilities that the property covers, useful when user wants to filter search
parking	TINYINT(1)	boolean, has or has not parking space
image	VARCHAR(45)	path to image of listing's thumbnail/main image
description	VARCHAR(256)	description of property, max length 256 characters
rooms	INT	how many rooms in property
floor_size	INT	floor size measured in square feet
pets	TINYINT(1)	boolean, allow pets or not
user_id	VARCHAR(32)	foreign key referencing user.sys_id
category	VARCHAR(32)	Type : apartment, room, house

listing_images table	**datatype**	**description**

sys_id	VARCHAR(32)	unique 32-length integer identifier
path	VARCHAR(256)	path to image
created	VARCHAR(45)	created date
listing_id	VARCHAR(32)	foreign key referencing listing.sys_id

user table	**datatype**	**description**
sys_id	VARCHAR(32)	unique 32-length integer identifier
active	TINYINT(1)	inactive = 0, active = 1 inactive
admin	TINYINT(1)	identifies user as admin or not
first_name	VARCHAR(45)	user's first name
last_name	VARCHAR(45)	user's last name
name	VARCHAR(45)	concatenated using first and last name

username	VARCHAR(12)	user’s display name
password	VARCHAR(20)	user’s account password
street_address	VARCHAR(45)	MIGHT NOT NEED THIS
city	VARCHAR(45)	MIGHT NOT NEED THIS
state	VARCHAR(45)	MIGHT NOT NEED THIS
zip_code	INT	MIGHT NOT NEED THIS
email	VARCHAR(45)	used to login
major	VARCHAR(45)	used for roommate finder, user’s area of study
age	INT	used for roommate finder
gender	ENUM	used for roommate finder
visible	TINYINT(1)	determines if this user will show up on roommate finder
description	VARCHAR(256)	used for roommate finder, personal biography

user_images table	**datatype**	**description**
sys_id	VARCHAR(32)	unique 32-length integer identifier
path	VARCHAR(256)	path to image
created	VARCHAR(45)	created date
listing_id	VARCHAR(32)	foreign key referencing listing.sys_id

approvals table	**datatype**	**description**
sys_id	VARCHAR(32)	unique 32-length integer identifier
active	VARCHAR(256)	path to image
approval_state	ENUM	populated with current time when record is created
created	VARCHAR(45)	created date

user_id	VARCHAR(32)	foreign key referencing user.sys_id
listing_id	VARCHAR(32)	foreign key referencing listing.sys_id

messages table	**datatype**	**description**
sys_id	VARCHAR(32)	unique 32-length integer identifier
body	VARCHAR(2000)	user's message
timestamp	VARCHAR(45)	created date
user_id	VARCHAR(32)	foreign key point to the message's author

Media storage

Images will be stored in a folder. Its table record will contain the image's relative path.

API

An interface will require child classes to implement methods for inserting, updating, and cycling through retrieved rows. The parent class contains methods for database connection as well as retrieving, ordering, limiting, and deleting rows. Each child class will inherit from the parent class and implement the interface along with any additional methods necessary. The parent class contains commonly used variables while the child classes contain variables pertaining to its table columns.

Search/filter architecture implementation

API classes include methods to build query strings for INSERT, UPDATE, DELETE, and SELECT queries.

Example of building a query string:

Initial string will be `SELECT * FROM listing WHERE` and addQuery() will append

`` `$column` $condition '$value' AND `` or `` `$column` LIKE '%$value%' AND ``

Before the query is executed, substr() will remove extras from the end of the string to create:

e.g `SELECT * FROM listing WHERE` `` `price` < '1000' AND `rooms` > 3 ``

Supported operators are =, !=, <, <=, >, >=, and LIKE (%value%). When a user finishes selecting his/her filter condition and updates the search results, a string query will be constructed and executed. SELECT queries will support ordering by column and limiting results. While the API supports queries based on any column, frontend code will place limits on what can be searched.

4. Identify actual Key risks for your project at this time:

1. **Skills Risks:**

   The team is split with experienced and less experienced developers.

   The backend team is working on php and mysql. Those with less knowledge on these skills are doing good to complete the tasks assigned by learning through different tutorials and with the use of stack overflow for issues.

   The frontend team is good to go with bootstrap and trying to work on the pages quickly.

   We are using Figma for Fidelity diagrams wherein all the members are learning the tool and implementing it.

2. **Schedule Risks:**

   We haven't faced any schedule risks till now and everyone is communicating with each other that will finally help us complete our project on scheduled time.

3. **Technical Risks:**

   No technical issues till now.

4. **Teamwork Risks:**

   We have frequent calls to discuss the work. The people who have enough knowledge for the issues arising help to solve it quickly.

5. **Legal/ Content Risks:**

   No copyright risks that we are aware of.

5. Project Management :

Our Team is using a Trello board for Project Management. This makes it easy to divide the tasks between the people. Each category has a Todo, Doing, and Done area to make everyone in the team aware of the status of the work.

Milestone 3 :

San Francisco State University

Software Engineering

CSC 648-848 SUMMER 2020

GatorHub

Milestone 03 Review Summary and Plans

Team 03

Megha Babariya (mbabariya@mail.sfsu.edu) (Team Lead)
Abraham Zepeda (Github Master)
Nathalia Sainez (Front-end Lead)
Raymond Kuang (Back-end Lead)
Yaxin Deng
Tania Nemeth
Harsh Saxena

Date of Milestone 3 Review : 07/27/2020

1. **Date of Milestone 3 review**: 07/27/2020

2. **Date of this summary and plan document:** 07/27/2020

3. **Summary of feedback and tasks to do :**

1. Search Bar on the top of the website
2. Make Large Logo.
3. Search by using Zip code.
4. Create a Listing button on Navbar
5. Guide the user where to sign up and create an account
6. Make changes to Font. (Big Size)
7. Show mandatory fields in Register.php
8. Make navigation bar bigger
9. Add I accept in register.php as mandatory one.
10. Make descriptions bigger(management) BIGGER IS BETTER.
11. Center the form structure(management page).Make less white space.
12. Having both options (send/update)->(admin). Can not have the same option to update and the option to send in the same page. Make sure to separate both options to clearly show which boxes need to be updated and which boxes are used for send.
13. Put the bars in the middle by following the UI mockup(admin page)

14. For Searching use Zip Code, Property Type and Rent/Buy.

15. Encrypt passwords.

16. Add Create Listing Page

17. Add User Dashboard Page.

18. Drop P3 features and focus on P1 features.

Final P1 items :

Unregistered User Functions:

1. Search – Users shall be able to search for real estate.
2. Sort and filter – Sort search results based on parameters such as distance and price.
3. Registration and Profiles – Users shall create an account to store data. Data includes personal information such as age, ethnicity, address, contact, and favorites.
4. Google Maps API – Users shall be able to view the listing on Google Maps within the webpage.

Registered User Functions:

1. Login/Logout –Users shall be able to login and logout of their account.
2. Posting – Users shall be able to post a listing via a unique sys id.
3. Edit Posting – Users shall be able to edit or remove their listing.
4. Contact Landlord - User shall be able to contact landlord by getting the Landlord's email id.
5. Personal User Dashboard Page : Users will be able to see personal listings.
6. Google Maps API – Users shall be able to view the listing on Google Maps within the webpage.

Admin Functions:

1. Has permissions to remove listings, content on its page, and user accounts.
2. Administrator approval is required for listings before they can be visible to users.

Milestone 4 :

San Francisco State University

Software Engineering

CSC 648-848 SUMMER 2020

GatorHub

Milestone 04

Team 03

Megha Babariya (mbabariya@mail.sfsu.edu) (Team Lead)
Abraham Zepeda (Github Master)
Nathalia Sainez (Front-end Lead)
Raymond Kuang (Back-end Lead)
Yaxin Deng
Tania Nemeth
Harsh Saxena

History Table:

Version No.	Date	Comments
01	08/03/2020	Initial Document

02	08/07/2020	Updated Document

1. Product Summary :

Our product, GatorHub, is a custom web application built exclusively for SFSU students and faculty to help connect them to the best housing. The following is an itemized list of all major committed functions. The functions described in this list will be delivered in our final project.

1. Search: Search listings based on different parameters such as zip code, property type, and whether a user is looking to rent or buy.
2. Filter: Filter results based on attributes such as distance to SFSU, minimum Price etc.
3. Google Maps API: Showcases distance from the estate to SFSU campus.
4. Registration: Users shall be able to register for an account.
5. Login/Logout: Users shall be able to to login and logout of their account.
6. Compose a Listing: Users shall be able to post a listing.
7. Edit Listing: Users shall be able to edit or remove their listing.
8. Display Listing: When a user clicks a listing, it will direct the user to a full view of that listing.
9. Contact Listing Owner: Ability to contact seller via button on listing.
10. Admin: Admin approval is required when a user wants to add a listing.

    Unique Feature : Our web application was exclusively for SFSU students and faculties without any hustle.

    URL of the Project : http://54.151.14.7/views/index.php

2. Usability Test Plan:

2.1 Usability Test Plan for Search bar - Test Objectives

Effectiveness of the search functionality shall provide San Francisco State students with the tools to filter through the listings so that they can find the apartments, condos, homes, or studios that they need and want to live in. The search feature is the most important feature that we offer as it gives a quick result of listings that the students are looking for. Search should be simple and intuitive to use, and the task of searching for a place to call home shall be reliable and fast, with the whole search process requiring very little effort and having high user efficiency.

Searching cuts browsing time significantly by allowing users to get the results that they want and the website also allows search by category of living space which further narrows the scope of a given search. In addition to this, the search and category fields must be persistent for ease of use. Correctly working on search and results are vital to the project and must be thoroughly tested.

Here, we will be testing usability, not functionality, and as such, the major function that we have selected to be tested for usability is the search function. We will mainly perform a “Validation” usability test to ensure that the product is usable. The test objectives are to thoroughly test the search function on GatorHub to get results for apartment listings and ensure that the search function is easy and intuitive to use without any further explanation. The placement of the search bar and input field placeholder text should be enough to guide the user through this process without guidance, it should be intuitive. During the usability test, we hope that the user is able to expose any usability flaws that impede the completion of the tested search functionality.

2.2 Usability Test Plan for 'Search' - Test Plan (background and setup)

System Setup: GatorHub website supports Google Chrome and Mozilla Firefox browsers and one of our teammates will use either browser to test the system by inputting various specific inputs in the search bar and making sure that the correct results are displayed. The tester will use edge cases and input only those fields that are intended to work in the search bar. These will be based on the functionality of the search bar and we will determine that the results agree with our expectations. For example, searching by zip work must work and display listings with that zip code.

Starting Point: The search bar functionality is present in the navbar at the top of every webpage. In addition, it functions the same regardless of which page the user is searching from. Home page of the site is reachable at http://54.151.14.7/views/index.php

Intended Users: The search bar is implemented for all users looking to find a place to live. The user base for this website is focused towards students at SFSU ages 18-30. Expected computer skills are average but should not be technically demanding. Searching is intended to be the primary way for users to find the living spaces that they need or want. Users (landlords) looking to post a living space can also search for similar items to use as an example when creating their own listings.

Evaluation: Efficiency in effort and design will be tested in this usability plan. Task should be done with minimal clicks and instructions. User satisfaction will be surveyed after the usability tests to measure ease of use and website clarity. San Francisco State students will be tested on comfort and acceptability.

The search input will be tested by using different use cases : Can be Zip code etc.

Property (category dropdown) will be composed of 4 items, which are: All (Property) by default, Apartments, Studios, Condos, and Houses.

What are we measuring: We will be measuring how the user feels through the process of searching for a specific listing and how does s/he feels with the general search procedure, and also how intuitive the search procedure is.

2.3 Usability Test Plan for 'Search' - Usability Task Description

Open a Google Chrome or Firefox browser and navigate to the project page:

http://54.151.14.7/views/index.php

Perform the following Tasks:

Here we evaluate if the user can understand and interpret the results for the listings, having different options to narrow down the results by having only apartment listings, property type Rent and property type Sale.

Effectiveness - would be measured by the amount of time taken and confusion caused by the usability task. This can be a part of Satisfaction or Effectiveness. A large amount of time taken would indicate that the interface for searching is unclear for first-time users to perform the given searches. If any one of those tasks takes more time than the others, we can pinpoint which functionality in our search needs to be reviewed.

Efficiency - would be measured by the amount of time, clicks, pages the tester takes to perform each search in the task. The percent of the users able to complete the task in a given time. By reviewing the click count of search, we can examine how efficient each action is in our search functionality.

Satisfaction - would be measured by deploying the following survey after the task is completed where we provide 3 Lickert scale questions getting user satisfaction after the above task has been performed, in a proper format as it is to be used by the reviewer:

Question	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Search bar and drop down menu were easy to use (good GUI design)					
Search results were informative, easy to interpret, and clearly displayed (reliable results)					
Searching was prefered to browsing all listings (convenient)					

There will also be a **comments section** along with the survey, where the user can put down how he felt about the whole search procedure. This survey results will signal to us the overall satisfaction of the product, specifically the search bar, and give us more insight into both its usability, marketing and future success among SFSU users that are looking for a place to live.

3. QA Test Plan:

3.1 Test objectives - What is being tested

We are testing the functionality of using the website's search, and testing it for bugs as well as to see if it performs as per specs such as validation. Thus, we will test input validation where it should only take up to 40 characters. The search and dropdown listing category selected after a search should also remain persistent. Search and category menu should be fully functional from any page in the website. Only correct and relevant results should be displayed for each tested search. Empty searches should return all found listings.

The search input will be tested for functionality. By functionality, we mean that the user will be able to search items, in this case, apartments close to campus by using the following: Property (category dropdown), search by zip code, city, or simply using the search bar to look for a specific field in the listing.

Property (category dropdown) will be composed of 4 items, which are: All (Property) by default, Apartments, Studios, Condos, and Houses. These items will filter listings based on listing- type. Second, the search bar will filter the listing by zip code, based on the user input.

3.2 HW and SW setup (http://54.151.14.7/views/index.php)

Our website should function on any machine connected to the internet that can run:

1. Google Chrome or 2. Firefox browsers.

We must also support the latest two versions of Google Chrome and Firefox. Browsers must be updated to ensure this. No further software required for testing.

3.3 Feature to be tested

This QA test will test the functionality of the search function in all of the major pages: home page, search result page, rent out page, listing detail page and user account page. The test will test for the following expected Functionality:

1. The search result page shall show all listings that match the search query (the search query being either ZIP CODE (5 digit number), PROPERTY (apartment, studio, condominium, or house), TYPE (rent/buy), as well as the number of found listings.
2. All results should be readable and openable to the listings detail page.
3. The search query shall not disappear from the search box after searching.
4. All listings shall include a listing title, listing details, listing price, ZIP code, number of bedrooms, pets allowed, and distance from the apartment to SFSU along with a map showing the Origin 'O' (listing address) and destination 'D' (SFSU address) pins.
5. All results shall be the same across both browsers.

3.4 Test Cases

3.4.1 Test 1 – Searching by ZIP CODE: Search for the ZIP code "94122" from all of the Testing Pages. Validate that every Expected Functionality is met. If all of the Expected Functionality is met, the test result shall be PASS. Perform the search on both browsers (Google Chrome, Firefox).

3.4.2 Test 2 – Searching by PROPERTY: Search for the property “House” from all of the Testing Pages. Validate that every Expected Functionality is met. If all of the Expected Functionality is met, the test result shall be PASS. Perform the search on both browsers.

3.4.3 Test 3 – Searching by TYPE: Search for the option to “Rent” from all of the Testing Pages. Validate that every Expected Functionality is met. If all of the Expected Functionality is met, the test result shall be PASS. Perform the search on both browsers.

3.5 Test Results

Test #	Test Field Search	Test Description	Test Input	Expected Output	Test Results
1	Zip Code	Search listings in a specific zip code	Enter in this search field “94122”	12 Listings in 94122 that are apartments, studios, condos, or houses for rent and sale	Pass/Pass
2	Property	Search listings that are Houses	Enter in this search field "House"	6 Listings that are Houses for rent and sale in any Zip code	Pass/Pass
3	Type	Search listings that are for Rent	Enter in this search field "Rent"	23 Listings for rent that are apartments, studios, condos, or houses in any Zip code	Pass/Pass

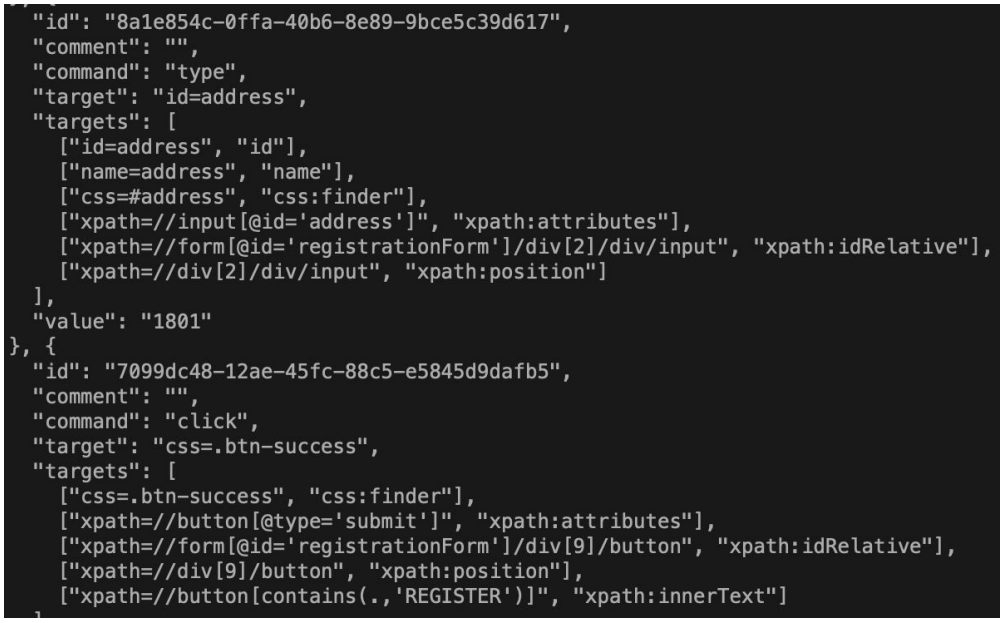
3.6 Selenium Automated Testing

In order to further conduct more testing, we decided to use Selenium and went through various activities, here is the registering/updating an account activity:

Firefox and Chrome WebDrivers:

We initiated a driver in both Firefox and Chrome browsers by creating a reference variable for the respective WebDriver interfaces and then instantiated them using a FirefoxDriver or ChromeDriver class. We defined a reference variable (driver) whose type was an interface, where the objects that we assigned to it were an instance of class FireFoxDriver or ChromeDriver drivers that implement that interface.

A default profile was launched in both web browsers so that no extensions and plugins would be loaded with the instance.



```
  "id": "8a1e854c-0ffa-40b6-8e89-9bce5c39d617",
  "comment": "",
  "command": "type",
  "target": "id=address",
  "targets": [
    ["id=address", "id"],
    ["name=address", "name"],
    ["css=#address", "css:finder"],
    ["xpath=//input[@id='address']", "xpath:attributes"],
    ["xpath=//form[@id='registrationForm']/div[2]/div/input", "xpath:idRelative"],
    ["xpath=//div[2]/div/input", "xpath:position"]
  ],
  "value": "1801"
}, {
  "id": "7099dc48-12ae-45fc-88c5-e5845d9dafb5",
  "comment": "",
  "command": "click",
  "target": "css=.btn-success",
  "targets": [
    ["css=.btn-success", "css:finder"],
    ["xpath=//button[@type='submit']", "xpath:attributes"],
    ["xpath=//form[@id='registrationForm']/div[9]/button", "xpath:idRelative"],
    ["xpath=//div[9]/button", "xpath:position"],
    ["xpath=//button[contains(.,'REGISTER')]", "xpath:innerText"]
```

Excerpts of Sign-Up Selenium tests:


```
{
  "id": "6ab3712c-3f4c-487a-8401-2bd597d69c38",
  "version": "2.0",
  "name": "CSC648",
  "url": "http://fooridise.com",
  "tests": [{
    "id": "c3fbe9cd-247a-4e45-9d92-abff8de9f80c",
    "name": "test1",
```

```
"commands": [{
  "id": "89e713f0-3298-4373-bcee-93cabe429ea6",
  "comment": "",
  "command": "open",
  "target": "/",
  "targets": [],
  "value": ""
}, {
  "id": "cbd41053-b6cb-4065-af17-615cf81af699",
  "comment": "",
  "command": "setWindowSize",
  "target": "1440x821",
  "targets": [],
  "value": ""
}, {
  "id": "fa396264-21ea-41c8-b4b5-0042a8d10e60",
  "comment": "",
  "command": "click",
  "target": "linkText=Sign Up",
  "targets": [
    ["linkText=Sign Up", "linkText"],
    ["css=#navbarSupportedContent a", "css:finder"],
    ["xpath=//a[contains(text(),'Sign Up')]", "xpath:link"],
    ["xpath=//div[@id='navbarSupportedContent']/button[2]/a", "xpath:idRelative"],
    ["xpath=//a[contains(@href, '/register')]", "xpath:href"],
    ["xpath=//button[2]/a", "xpath:position"],
    ["xpath=//a[contains(.,'Sign Up')]", "xpath:innerText"]
  ],
  "value": ""
}, , {
  "id": "61342e71-ef19-435a-98c4-d931092a39a0",
  "comment": "",
  "command": "click",
  "target": "id=email",
  "targets": [
    ["id=email", "id"],
    ["name=email", "name"],
    ["css=#email", "css:finder"],
    ["xpath=//input[@id='email']", "xpath:attributes"],
    ["xpath=//form[@id='registrationForm']/div[4]/div[2]/input", "xpath:idRelative"],
    ["xpath=//div[4]/div[2]/input", "xpath:position"]
  ],
  "value": ""
}, {
  "id": "6a1cc498-ade0-4685-8320-3976d0b8b17b",
  "comment": "",
  "command": "type",
  "target": "id=email",
  "targets": [
    ["id=email", "id"],
    ["name=email", "name"],
    ["css=#email", "css:finder"],
    ["xpath=//input[@id='email']", "xpath:attributes"],
    ["xpath=//form[@id='registrationForm']/div[4]/div[2]/input", "xpath:idRelative"],
    ["xpath=//div[4]/div[2]/input", "xpath:position"]
  ],
  "value": "azepeda@sfsu.edu"
```

```
    }, {
      "id": "7099dc48-12ae-45fc-88c5-e5845d9dafb5",
      "comment": "",
      "command": "click",
      "target": "css=.btn-success",
      "targets": [
        ["css=.btn-success", "css:finder"],
        ["xpath=//button[@type='submit']", "xpath:attributes"],
        ["xpath=//form[@id='registrationForm']/div[9]/button", "xpath:idRelative"],
        ["xpath=//div[9]/button", "xpath:position"],
        ["xpath=//button[contains(.,'REGISTER')]", "xpath:innerText"]
      ],
      "value": ""
    }]
  }],
  "suites": [{
    "id": "8ebb9aed-d166-4725-a387-5be4543a6fbd",
    "name": "Default Suite",
    "persistSession": false,
    "parallel": false,
    "timeout": 300,
    "tests": ["c3fbe9cd-247a-4e45-9d92-abff8de9f80c"]
  }],
  "urls": ["http://fooridise.com/"],
  "plugins": []
}
```


Results

We conducted various tests, including account sign-ups and customer orders and passed them all. Here are some excerpts from them.

```
        Running ‘Sign-up test”
07:28:27

        1.
        open on / OK
        07:28:28
        2.
        setWindowSize on 1440x821 OK
        07:28:29
        3.
        click on linkText=Sign Up OK
        07:28:30
        4.
        click on css=.container-fluid:nth-child(3) .btn > h3 OK
        07:28:32
        5.
        click on id=firstName OK
        07:28:34
        6.
        type on id=firstName with value Abraham OK
        07:28:36
```

7.
type on id=lastName with value Zepeda OK
07:28:38
8.
type on id=address with value 1801 14th St, Oakland, CA 94607 OK
07:28:40
9.
click on id=education OK
07:28:42
10.
select on id=education with value label=Yes OK
07:28:44
11.
click on id=phone OK
07:28:46
12.
type on id=phone with value 4152339878 OK
07:28:46
13.
click on css=.col-md-7:nth-child(2) OK
07:28:46
14.
click on id=email OK
07:28:46
15.
type on id=email with value azepeda@sfsu.edu OK
07:28:46
16.
click on id=password OK
07:28:46
17.
type on id=password with value hello OK
07:28:47
18.
type on id=passwordConfirmation with value hello OK
07:28:47
19.
click on id=terms OK
07:28:47
20.
click on css=.btn-success OK
07:28:47
21.
click on css=.row:nth-child(2) OK
07:28:47
22.
type on id=address with value 1801 OK
07:28:47
23.
click on css=.btn-success OK
07:28:47
'Sign-up test' completed successfully

Comparison between Expected and Actual Values:

We used a basic if-else structure to compare the actual condition with the expected one.

```
if (actualCondition.contentEquals (expectedCondition)) {
System.out.printIn("Success");
}
else {System.out.printIn("Fail");
}
```

We were able to gather all successes in all of our Selenium testing.

4. Code Review:

Coding Style: Careful indentation, consistent use of spaces over tabs, clear and concise commenting in the style of "/* text */" or "// text" and starting brackets on the same line as function call (Egyptian braces).

Peer review was conducted on the search feature mentioned in usability testing section 2. The snippet from index.php is included in the document and feedback from index as well as listing_search.php (file is 250+ lines so did not paste it).

Snippet from Index.php prior to peer review:

```
<!-- Search -->
<form method="POST" action="/views/listings/result.php">
  <div class="row">
    <div class="col-lg-12">
      <div class="row">

        <!-- Zip Code -->
        <div class="col-lg-6 col-md-3 col-sm-12 p-0 m-1">
          <input type="text" name="zip_code_listing" class="form-control search-slt"
                 placeholder="search by zip code">
        </div>
```

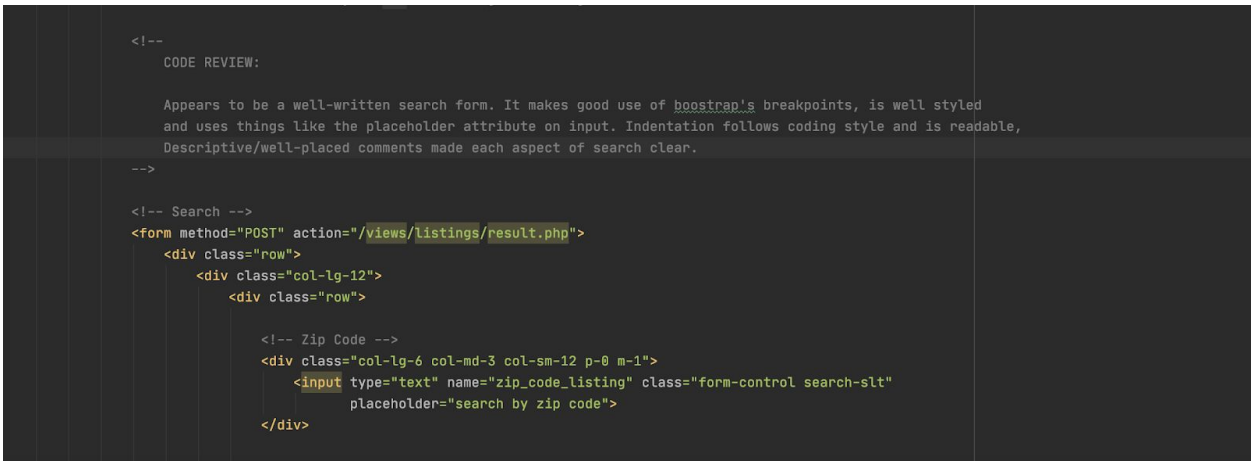
```
            <!-- Property -->
            <div class="col-lg-2 col-md-3 col-sm-12 p-0 m-1">
                <select class="form-control search-slt" id="property_type"
                    name="property_type_listing">
                    <option value="">Property</option>
                    <option value="apartment">Apartment</option>
                    <option value="studio">Studio</option>
                    <option value="condonimium">Condominium</option>
                    <option value="house">House</option>
                </select>
            </div>

            <!-- Type -->
            <div class="col-lg-2 col-md-3 col-sm-12 p-0 m-1">
                <select class="form-control search-slt" id="type"
                    name="payment_type_listing">
                    <option value="">Type</option>
                    <option value="rent">Rent</option>
                    <option value="buy">Buy</option>
                </select>
            </div>

            <!-- Submit btn -->
            <div class="col-lg-1 col-md-5 col-sm-5 p-0 m-1">
                <input type="submit" class="btn btn-dark">
            </div>
        </div>
    </div>
  </div>

</form>
```

Feedback for index.php:



```
<!--
    CODE REVIEW:

    Appears to be a well-written search form. It makes good use of boostrap's breakpoints, is well styled
    and uses things like the placeholder attribute on input. Indentation follows coding style and is readable,
    Descriptive/well-placed comments made each aspect of search clear.
-->

<!-- Search -->
<form method="POST" action="/views/listings/result.php">
    <div class="row">
        <div class="col-lg-12">
            <div class="row">

                <!-- Zip Code -->
                <div class="col-lg-6 col-md-3 col-sm-12 p-0 m-1">
                    <input type="text" name="zip_code_listing" class="form-control search-slt"
                           placeholder="search by zip code">
                </div>
```

Feedback for listing_search.php:

```
/*
    CODE REVIEW:
    Repetitiveness could be minimized, maybe put all of the search property names in an
    array and loop through that? There's some additional complexity added there by booolean arguments
    (since they need to be converted from true/false to 1/0), but that can be addressed with some type inspection.

    There is a nice use of console.log in a script tag, makes debugging this module easier. The use of $_SERVER['document_root']
    is clever as it prevents relative imports (PHP has strange import semantics). Implementing this prevents difficult-to-debug
    issues further down the line.

    The comment at the top that mentions that this module does something on load which is useful & makes it clear that stuff
    is being done when the module is imported.Overall could use slightly more commenting but seems to generally following
    coding style and is clean/readable.

*/

// retrieve form values
if (isset($_POST['title_listing'])) {
    $title = $_POST['title_listing'];
}

if (isset($_POST['city_listing'])) {
    $city = $_POST['city_listing'];
}
```

5. Self-Check on best practices for security:

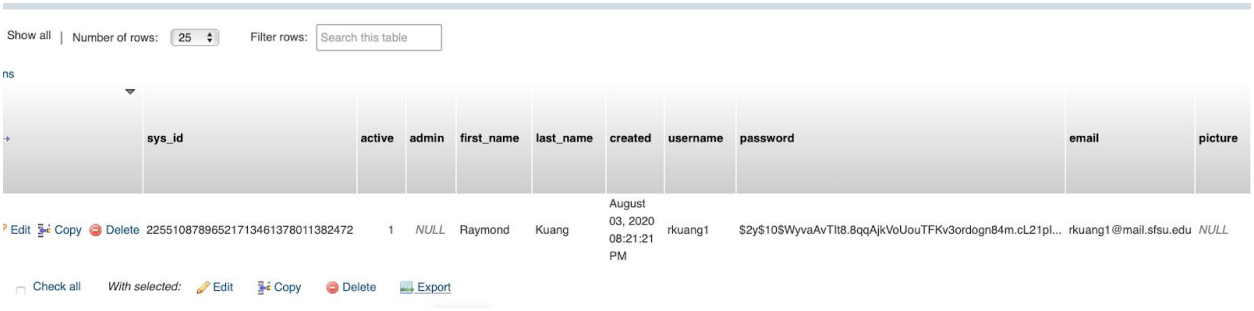
1. **List of major assets you are protecting**
    - User personal information
    - Images on server
    - Database integrity

2. **List of major threats for each asset above**
    - User personal information
    - Images on server
    - Database integrity

3. **For each asset say how you are protecting it**
    - **User personal information**: password encryption using php's password_hash() function
    - **Images on server**: file name encryption using php's password_hash() function
    - **Database integrity**: using PDO which allows for positional and named placeholders to prevent SQL injections, also using php's addslashes() to escape characters to further protect against SQL injections

4. Confirm that you encrypt PW in the DB

- User passwords are encrypted before being sent out to the database. Here you can see a screenshot showing the encrypted user passwords in the database:



Show all | Number of rows: 25 Filter rows: Search this table

	sys_id	active	admin	first_name	last_name	created	username	password	email	picture
Edit Copy Delete	225510878965217134613780113824 72	1	NULL	Raymond	Kuang	August 03, 2020 08:21:21 PM	rkuang1	$2y$10$WyvaAvTlt8.8qqAjkVoUouTFKv3ordogn84m.cL21pl...	rkuang1@mail.sfsu.edu	NULL

Check all With selected: Edit Copy Delete Export

Passwords are hashed in the API before inserting or updating a record

```php
function insert()
{
    // has the user's password before inserting the new record
    $this->password = password_hash($this->password, algo: PASSWORD_DEFAULT);
```

```php
function update()
{
    // has the user's password before updating the current record
    $this->password = password_hash($this->password, algo: PASSWORD_DEFAULT);
```

5. Confirm Input validation (list what is being validated and what code you used

- User's email and confirm email must match
- Email must be an official SFSU one
- User's password and confirm password must match
- Search is restricted to 40 alphanumeric characters

We have implemented a listener and once the submit button on the form is clicked, the following code is used for input validation of the registration form:

```
function validate(e) {

    //validate email is SFSU
    if (document.getElementById( elementId: "email_register").value.split('@')[1] !== 'mail.sfsu.edu') {
        alert('you must register with an SFSU email address');
        e.preventDefault();
        return false;
    }

    // validate emails match
    if (document.getElementById( elementId: "email_register").value !== document.getElementById( elementId: "email_confirm_register").value) {
        alert('emails do not match, please reverify your email');
        e.preventDefault();
        return false;
    }


    // validate passwords match
    if (document.getElementById( elementId: "password_register").value !== document.getElementById( elementId: "password_confirm_register").value) {
        alert('passwords do not match, please re-enter your password');
        e.preventDefault();
        return false;
    }
}
```

Snippet from index.php that restricts search entry to 40 characters or less:

```
<div class="col-lg-6 col-md-3 col-sm-12 p-0 m-1">
    <input type="text" name="zip_code_listing" class="form-control search-slt" maxlength="40"
           placeholder="search by zip code">
</div>
```

Snippet from result.php that restricts search entry to 40 characters or less:

```
<div class="col-lg-6 col-md-3 col-sm-12 p-0 m-1">
    <input type="text" name="zip_code_listing" class="form-control search-slt" maxlength="40"
           placeholder="search by zip code">
</div>
```

6. Self- Check: Adherence to original non-functional specs:

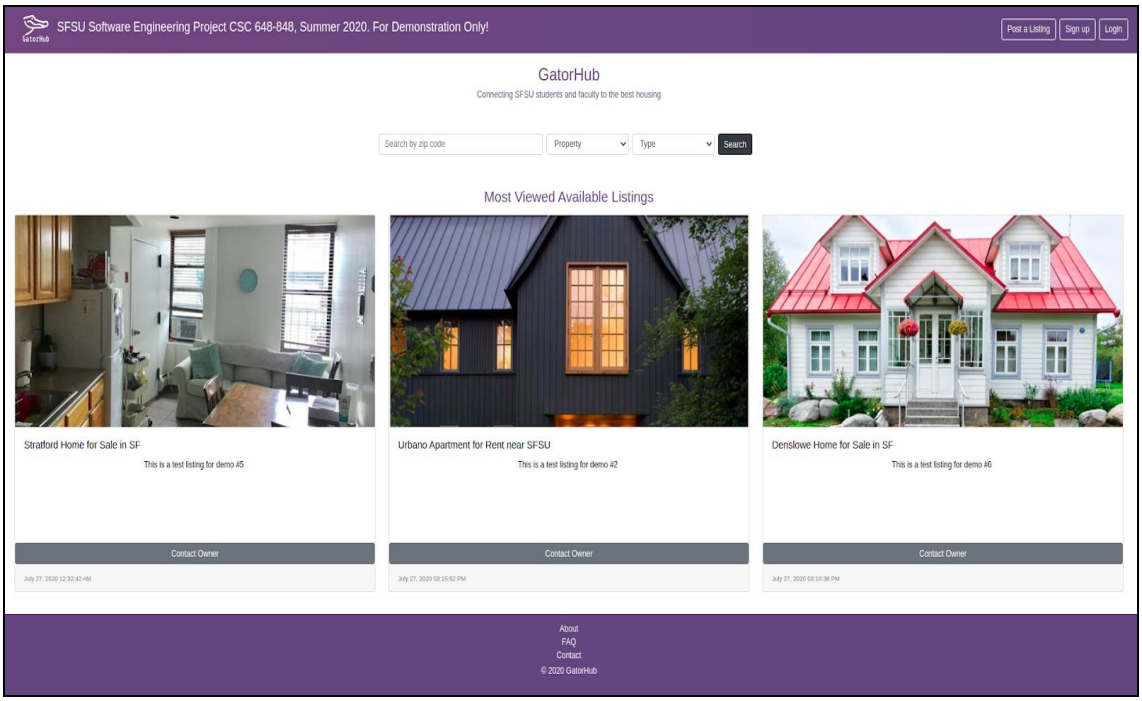
Non-functional Specs	Done/ Not Done
1. Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0.	Done
2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers	Done
3. Selected application functions must render well on mobile devices	Done
4. Data shall be stored in the team's chosen database technology on the team's deployment server.	Done
5. No more than 50 concurrent users shall be accessing the application at any time.	Done

6. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.	Done
7. The language used shall be English (no localization needed)	Done
8. Application shall be very easy to use and intuitive.	Done
9. Google analytics shall be used	Done
10. No email clients shall be allowed. Interested users can only message to sellers via in-site messaging. One round of messaging (from user to seller) is enough for this application	In Progress
11. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.	Done

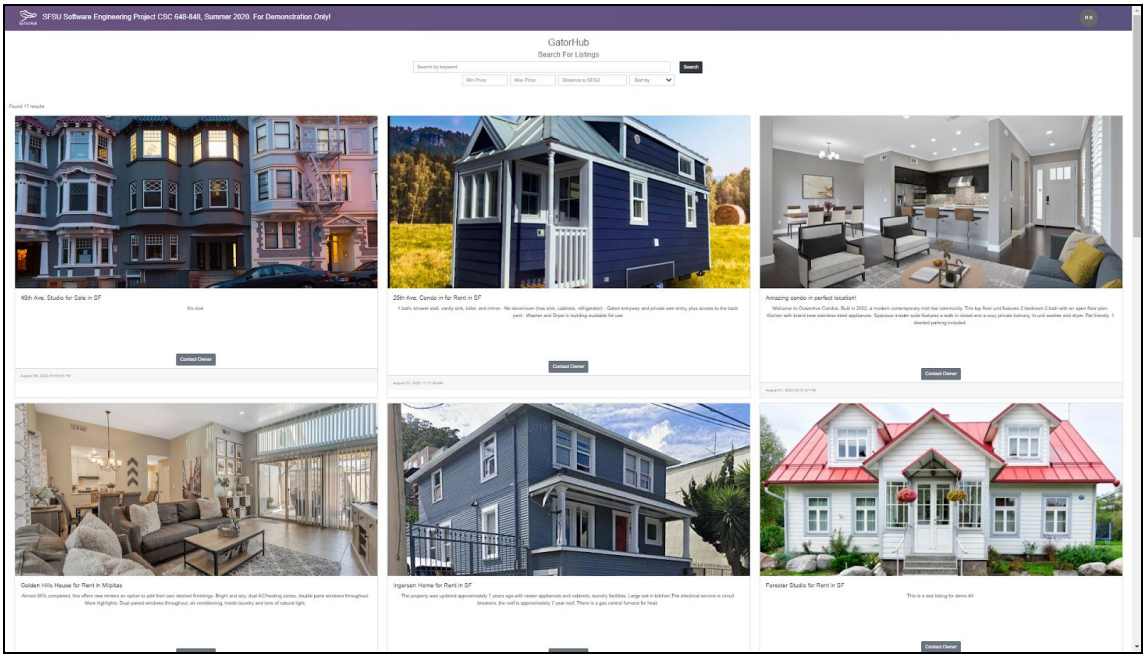
12. Site security: basic best practices shall be applied (as covered in the class) for main data items.	Done
13. Media formats shall be standard as used in the market today.	Done
14. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development.	Done
15. The website shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Summer 2020. For Demonstration Only" at the top of the WWW page. (Important so as to not confuse this with a real application).	Done

4) Product Screenshots:

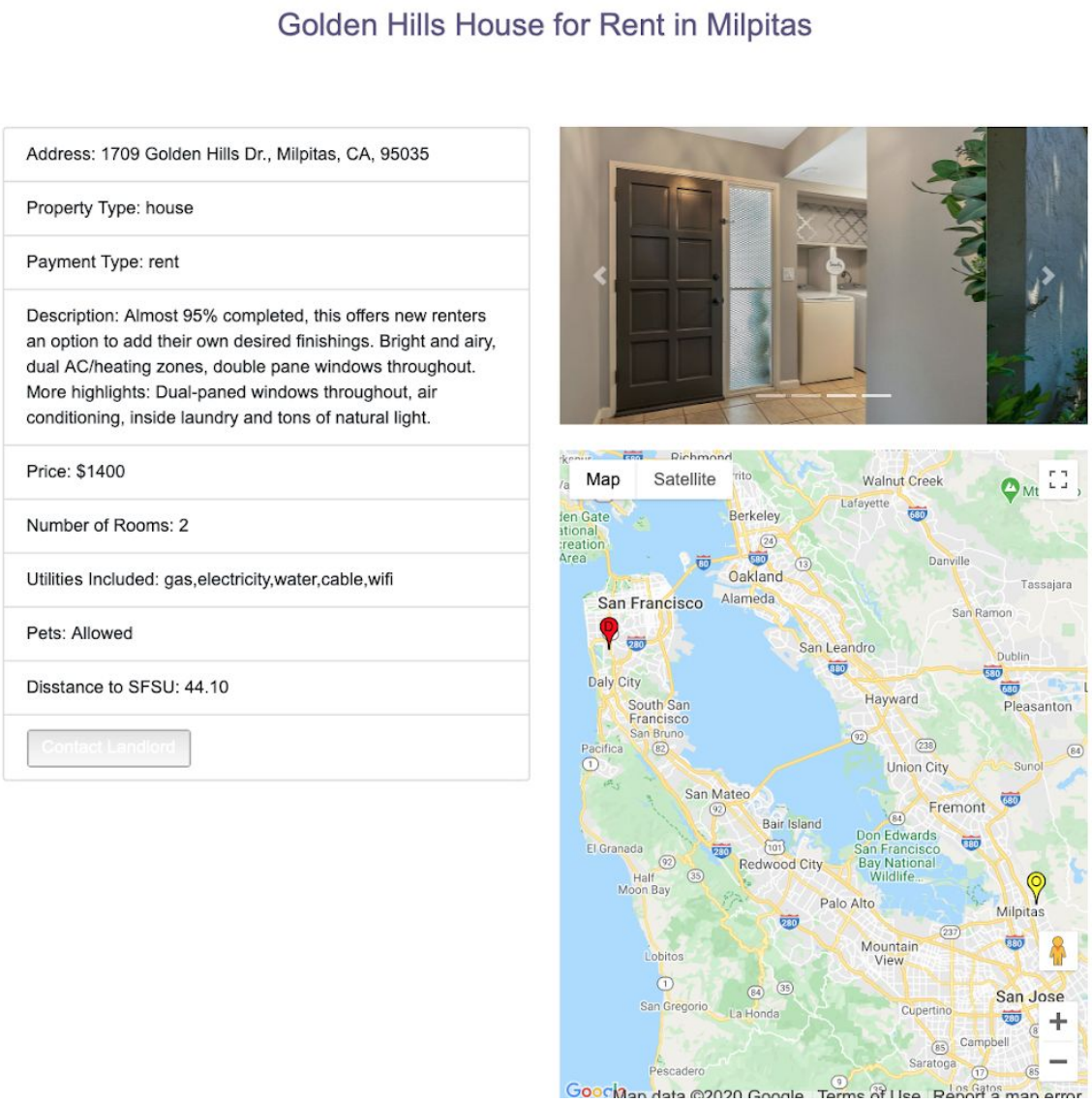
- **Homepage :** landing page when a new customer arrives to the site.



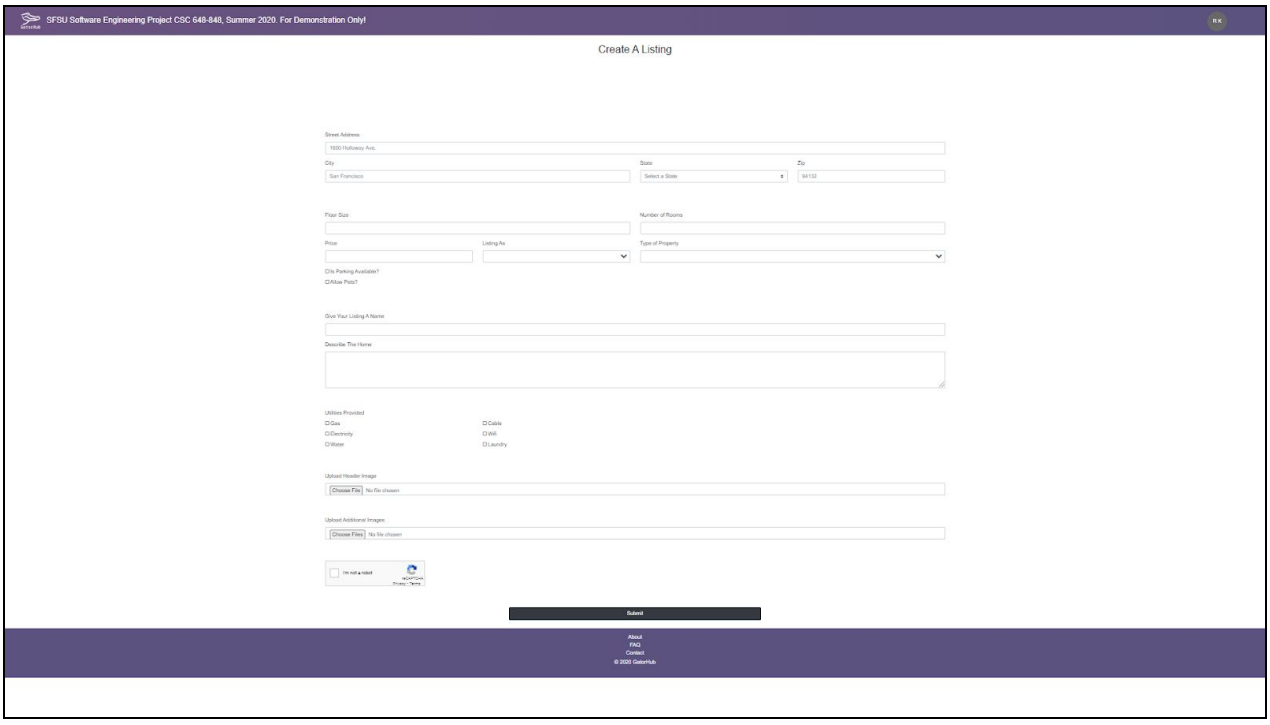
- **Result Page :** page after a customer searches for something via the nav bar



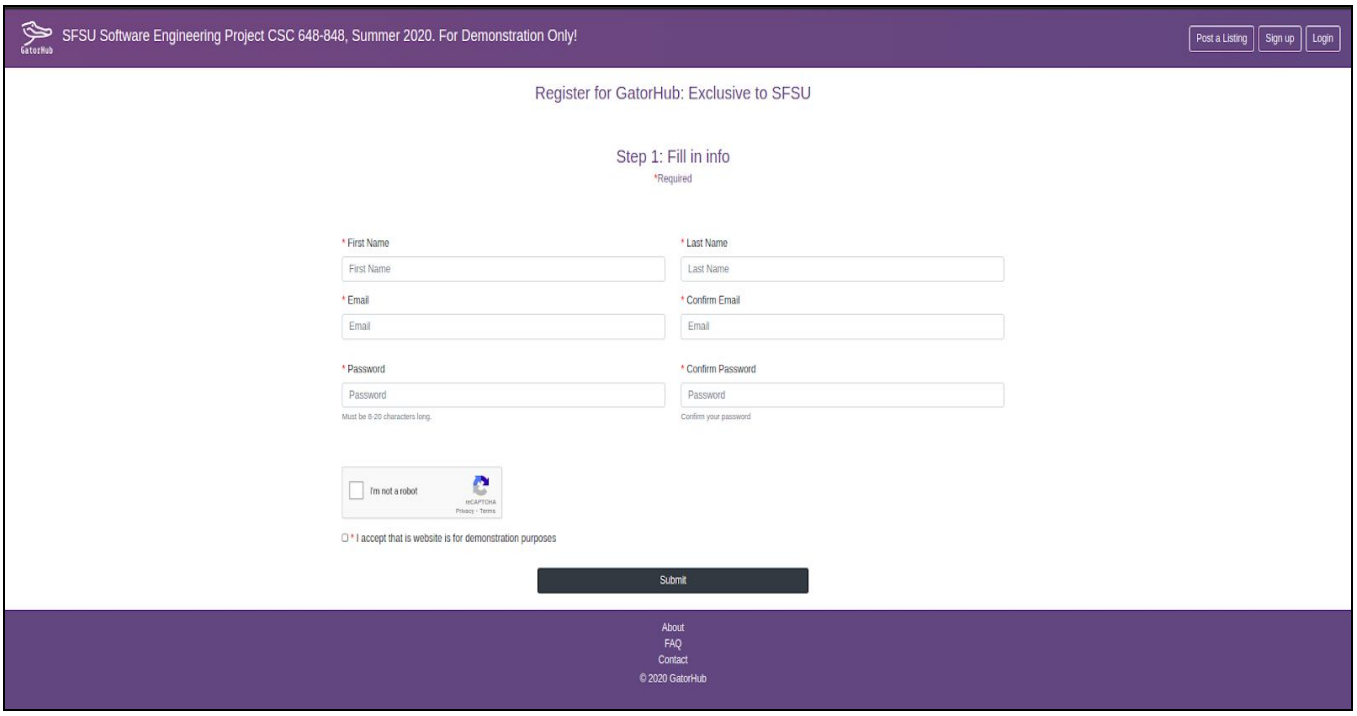
- **<u>Detailed listing</u> :** detailed information of a listing



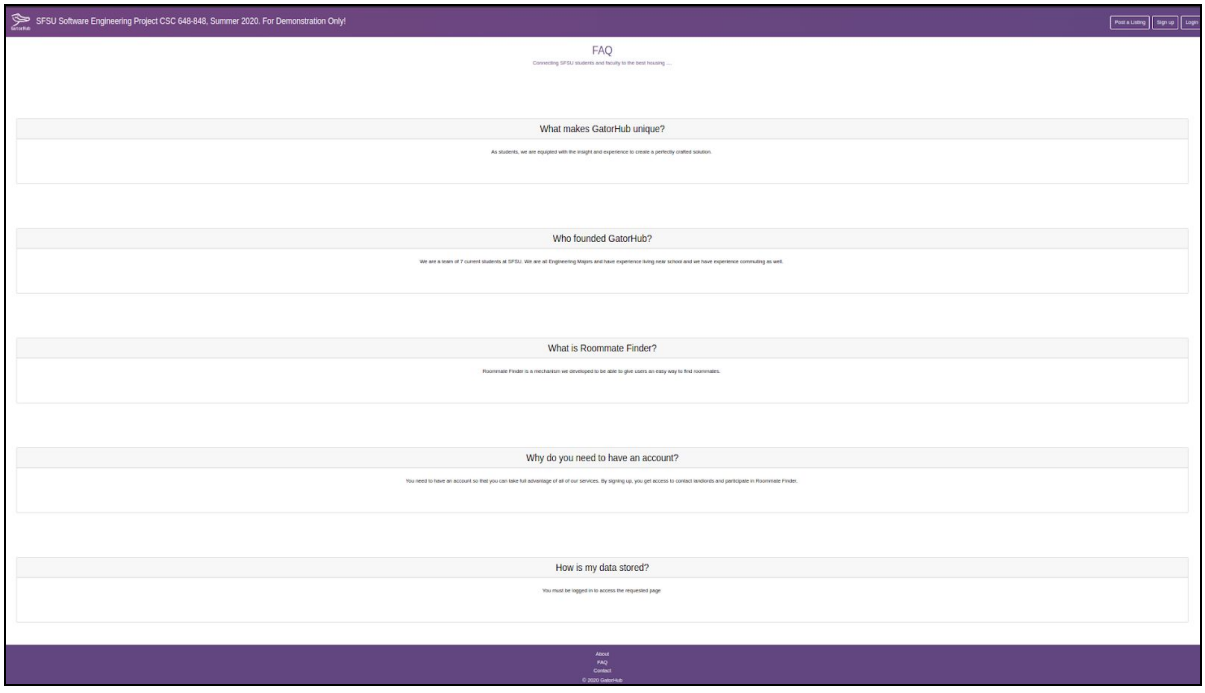
- **<u>Create a Listing Page</u> :** signed-in customer can create a listing



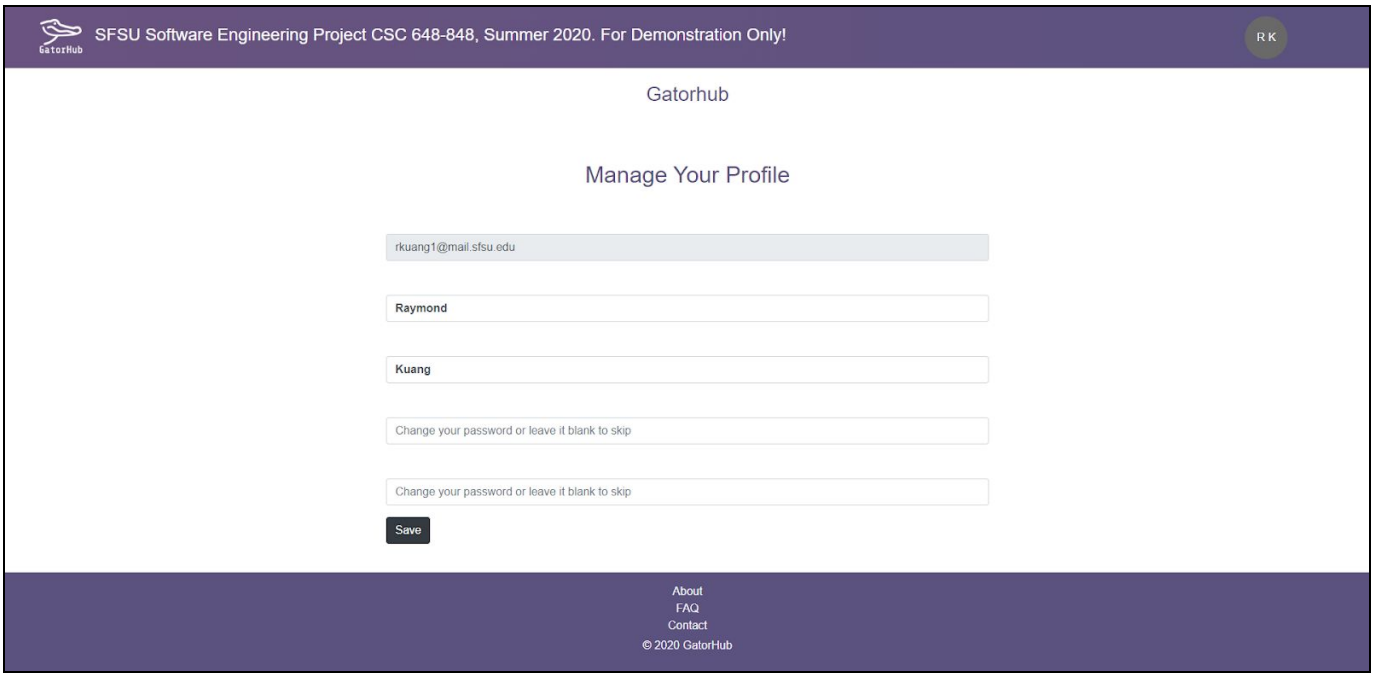
- **<u>Sign Up Page</u> :** new customer can choose to create account so that he can post and save listings



- **<u>FAQ Page</u> :** terms and conditions as well as who we are and why our site is different

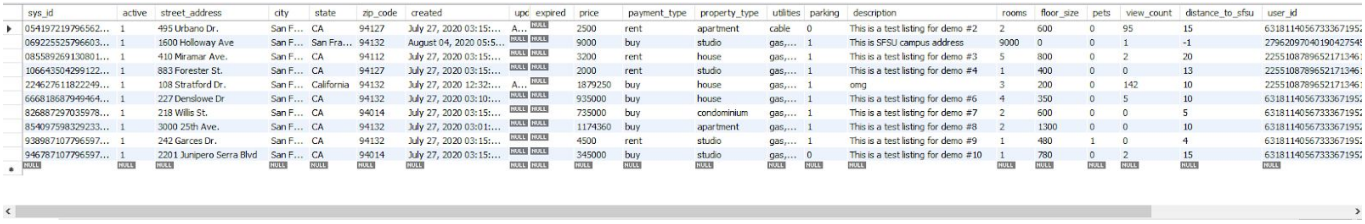


- **<u>Profile Page</u> :** the logged-in user can edit his/her profile



5) Database Organization :

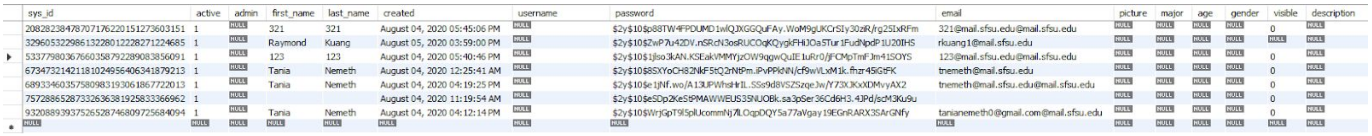
Listing Table



sys_id	active	street_address	city	state	zip_code	created	upd	expired	price	payment_type	property_type	utilities	parking	description	rooms	floor_size	pets	view_count	distance_to_sfsu	user_id
0541972197965620...	1	495 Urbano Dr.	San F...	CA	94127	July 27, 2020 03:15:...	A...	NULL	2500	rent	apartment	cable	0	This is a test listing for demo #2	2	600	0	95	15	6318114056733367195
0692255257966030...	1	1600 Holloway Ave	San F...	San Fra...	94132	August 04, 2020 05:5...	NULL	NULL	9000	buy	studio	gas,...	1	This is SFSU campus address	9000	0	0	1	-1	2796209704019042754
0855892691308010...	1	410 Miramar Ave.	San F...	CA	94112	July 27, 2020 03:15:...	NULL	NULL	3200	rent	house	gas,...	1	This is a test listing for demo #3	5	800	0	2	20	2255108789652171346
1066435042991220...	1	883 Forester St.	San F...	CA	94127	July 27, 2020 03:15:...	NULL	NULL	2000	rent	studio	gas,...	1	This is a test listing for demo #4	1	400	0	0	13	2255108789652171346
2246276118222490...	1	108 Stratford Dr.	San F...	California	94132	July 27, 2020 12:32:...	A...	NULL	1879250	buy	house	gas,...	1	omg	3	200	0	142	10	2255108789652171346
6668186879494640...	1	227 Denslowe Dr	San F...	CA	94132	July 27, 2020 03:10:...	NULL	NULL	935000	buy	house	gas,...	1	This is a test listing for demo #6	4	350	0	5	10	6318114056733367195
8268872970359780...	1	218 Willis St.	San F...	CA	94014	July 27, 2020 03:15:...	NULL	NULL	735000	buy	condominium	gas,...	1	This is a test listing for demo #7	2	600	0	0	5	6318114056733367195
8540975983293330...	1	3000 25th Ave.	San F...	CA	94132	July 27, 2020 03:01:...	NULL	NULL	1174360	buy	apartment	gas,...	1	This is a test listing for demo #8	2	1300	0	0	10	6318114056733367195
9389871077965970...	1	242 Garces Dr.	San F...	CA	94132	July 27, 2020 03:15:...	NULL	NULL	4500	rent	studio	gas,...	1	This is a test listing for demo #9	1	480	1	0	4	6318114056733367195
9467871077965970...	1	2201 Junipero Serra Blvd	San F...	CA	94014	July 27, 2020 03:15:...	NULL	NULL	345000	buy	studio	gas,...	0	This is a test listing for demo #10	1	780	0	2	15	6318114056733367195
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
sys_id	VARCHAR(32)	☑	☑	☑	☐	☐	☐	☐	☐	
active	TINYINT(1)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
street_address	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
city	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
state	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
zip_code	INT	☐	☐	☐	☐	☐	☐	☐	☐	NULL
created	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
updated	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
expired	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
price	INT	☐	☐	☐	☐	☐	☐	☐	☐	NULL
payment_type	ENUM('rent', 'buy')	☐	☐	☐	☐	☐	☐	☐	☐	NULL
property_type	ENUM('apartment', 's...	☐	☐	☐	☐	☐	☐	☐	☐	NULL
utilities	SET('gas', 'electricity'...	☐	☐	☐	☐	☐	☐	☐	☐	NULL
parking	TINYINT(1)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
description	MEDIUMTEXT	☐	☐	☐	☐	☐	☐	☐	☐	NULL
rooms	INT	☐	☐	☐	☐	☐	☐	☐	☐	NULL
floor_size	INT	☐	☐	☐	☐	☐	☐	☐	☐	NULL
pets	TINYINT(1)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
view_count	INT	☐	☐	☐	☐	☐	☐	☐	☐	NULL
distance_to_sfsu	INT	☐	☐	☐	☐	☐	☐	☐	☐	NULL
user_id	VARCHAR(32)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
title	VARCHAR(512)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
image	VARCHAR(512)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

User Table



sys_id	active	admin	first_name	last_name	created	username	password	email	picture	major	age	gender	visible	description
2082823847870717622015127360315	1	NULL	321	321	August 04, 2020 05:45:06 PM	NULL	$2y$10$p88TW4FPDUMD1wlQJXGGQuFAy.WoM9gUKCrSIy30zR/rg251xRFm	321@mail.sfsu.edu@mail.sfsu.edu	NULL	NULL	NULL	NULL	0	NULL
3296053229861322801222827122468 5	1	NULL	Raymond	Kuang	August 05, 2020 03:59:00 PM	NULL	$2y$10$ZwP7u42DV.nSRch3osRUCOqKQygkFHiJOa5Tur1FudNpdP1U20IHS	rkuang1@mail.sfsu.edu	NULL	NULL	NULL	NULL	NULL	NULL
5337798036766035879228908385609 1	1	NULL	123	123	August 04, 2020 05:40:46 PM	NULL	$2y$10$1jlso3kAN.KSEakVMMYjzOW9qqwQuIE1uRr0/jFCMpTmFJm41SOYS	123@mail.sfsu.edu@mail.sfsu.edu	NULL	NULL	NULL	NULL	0	NULL
6734732142118102495640634187921 3	1	NULL	Tania	Nemeth	August 04, 2020 12:25:41 AM	NULL	$2y$10$8SXYoCH82NkF5tQ2rNtPm.iPvPPkNN/cf9wVLxM1k.fhzr45iGtFK	tnemeth@mail.sfsu.edu	NULL	NULL	NULL	NULL	0	NULL
6893346035758098319306186772201 3	1	NULL	Tania	Nemeth	August 04, 2020 04:19:25 PM	NULL	$2y$10$e1jNf.wo/A13UPWhsHrIL.SSs9d8VSZSzqeJw/Y73XJKxXDMvyAX2	tnemeth@mail.sfsu.edu@mail.sfsu.edu	NULL	NULL	NULL	NULL	0	NULL
7572886528733263638192583336696 2	1	NULL			August 04, 2020 11:19:54 AM	NULL	$2y$10$eSDp2KeSt9MAWWEUS3SNUOBk.sa3pSer36Cd6H3.4JPd/scM3Ku9u		NULL	NULL	NULL	NULL	0	NULL
9320889393752652874680972568409 4	1	NULL	Tania	Nemeth	August 04, 2020 04:12:14 PM	NULL	$2y$10$WrjGpT9l5pIUcommNj7LOcpDQY5a77aVgay19EGnRARX3SArGNfy	tanianemeth0@gmail.com@mail.sfsu.edu	NULL	NULL	NULL	NULL	0	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
sys_id	VARCHAR(32)	☑	☑	☑	☐	☐	☐	☐	☐	
active	TINYINT(1)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
admin	TINYINT(1)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
first_name	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
last_name	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
created	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
username	VARCHAR(50)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
password	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
email	VARCHAR(45)	☐	☐	☑	☐	☐	☐	☐	☐	NULL
picture	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
major	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
age	INT	☐	☐	☐	☐	☐	☐	☐	☐	NULL
gender	ENUM('male', 'female...	☐	☐	☐	☐	☐	☐	☐	☐	NULL
visible	TINYINT(1)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
description	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
recovery_question_school	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
recovery_question_name	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
recovery_question_city	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
recovery_question_book	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

Message Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
sys_id	VARCHAR(32)	☑	☑	☑	☐	☐	☐	☐	☐	
sender_id	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
receiver_id	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
content	MEDIUMTEXT	☐	☑	☐	☐	☐	☐	☐	☐	
created	VARCHAR(45)	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

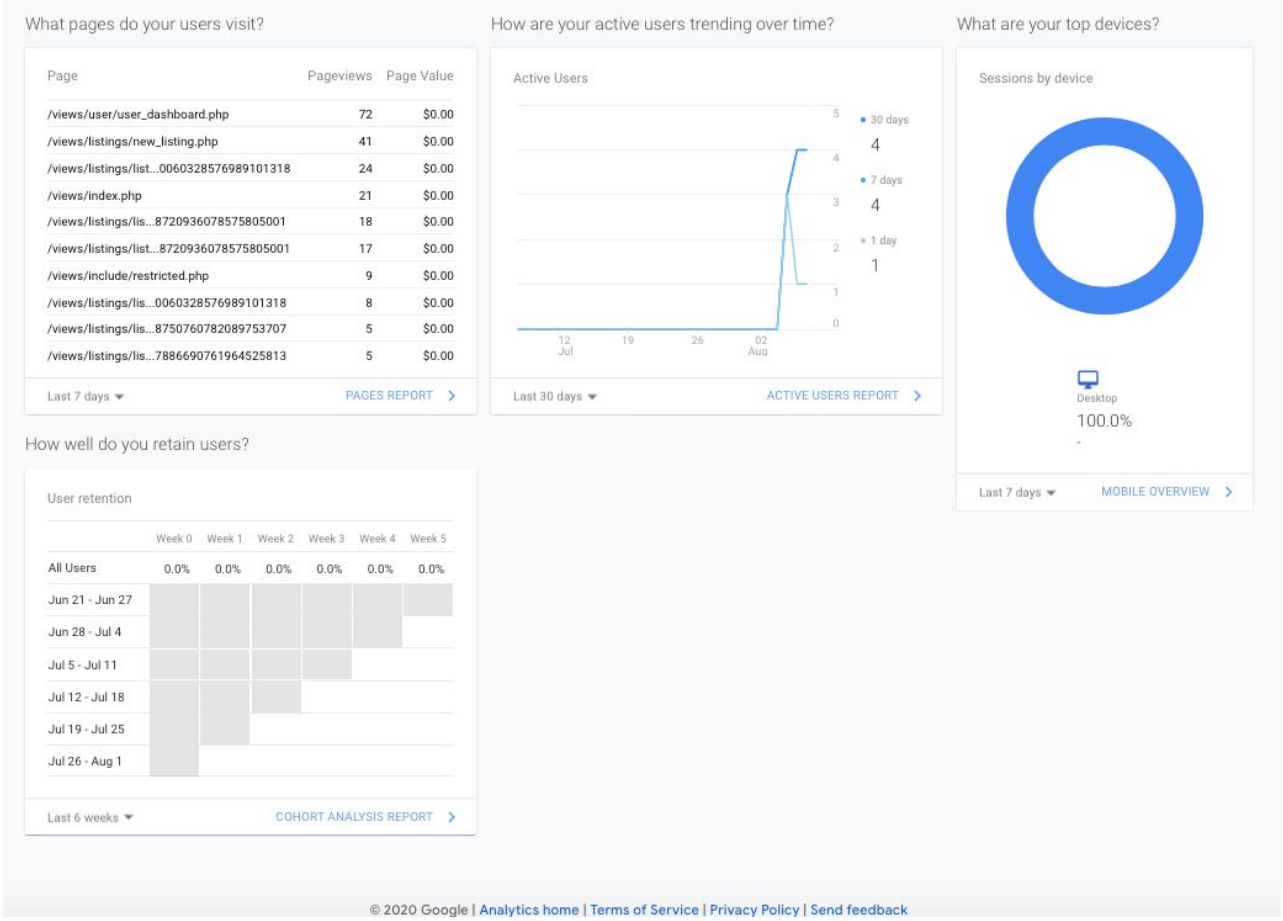
Approvals Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
sys_id	VARCHAR(32)	☑	☑	☑	☐	☐	☐	☐	☐	
active	TINYINT(1)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
approval_state	ENUM('requested','approved')	☐	☐	☐	☐	☐	☐	☐	☐	NULL
created	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
user_id	VARCHAR(32)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
listing_id	VARCHAR(32)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
approver_id	VARCHAR(32)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

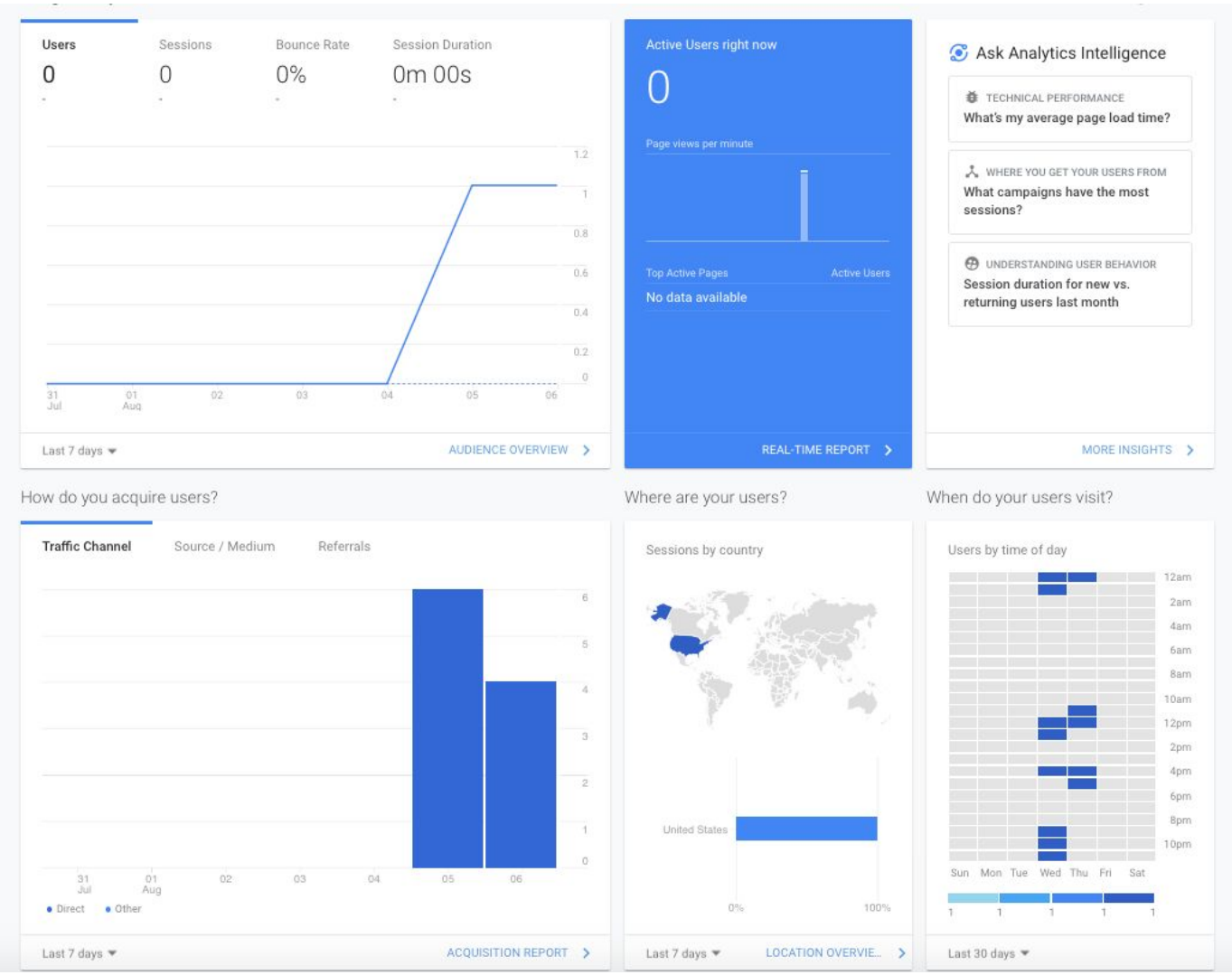
6) Google Analytics plot for WWW site:

Google Analytics allowed us to do a digital analysis of qualitative and quantitative data from our housing listing website GatorHub, a business which we would like to actually create in the future. This in turn can help us drive a continuous improvement of the online experience that current and potential customers have, and translate that desired outcome for our business in attracting a niche market of students and local landlords.

The analytics system of Google Analytics has four components: collection, processing, configuration, and reporting. We collected user interaction data from different pages in our websites and transformed that raw data into useful data. We were able to apply such filters as to what pages were the most popular and for how long each person stayed in each page as seen below:



In another report below, we are able to track e-commerce activity and performance for the GatorHub website and use our customized data collection to guide webpage development. We can see the engagement through session duration and since the site launched not too long ago, this number is still very small with high hopes of exponential growth as time passes on.



However, we did face some limits with Google Analytics as they limit the amount of goals that we could create. But one feature that we did find extremely helpful however, was seeing the day-time of engagement seen above so that we could work on our server during non peak times for best user experience.

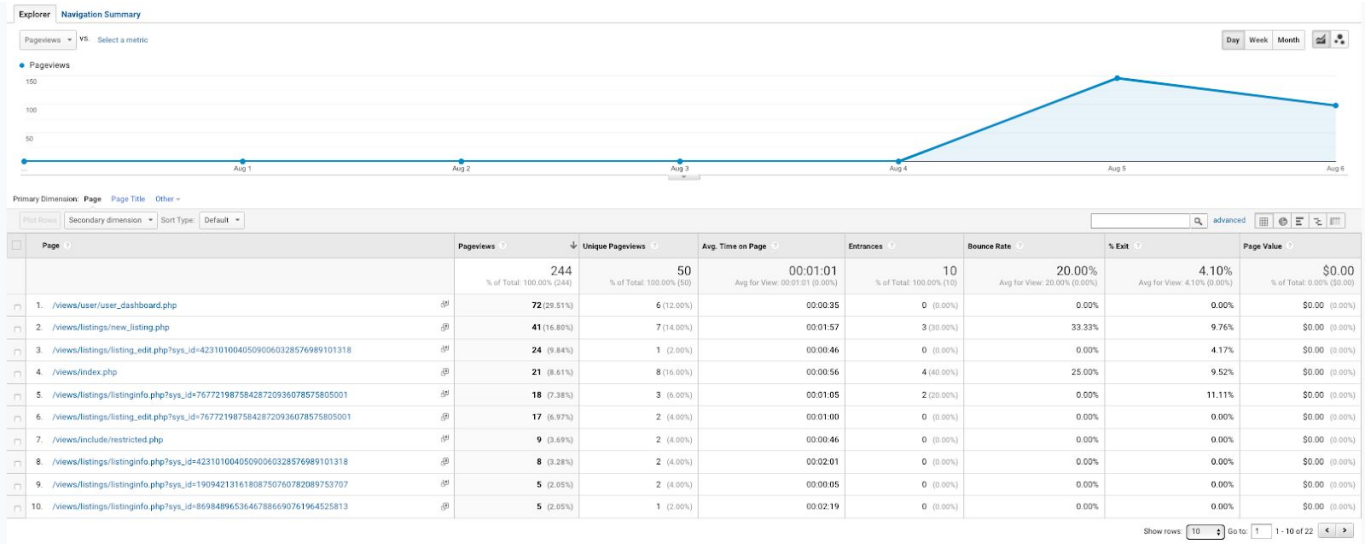
We also came up with a list of best metrics that we could track such as:

1. the bounce rate
2. average session duration
3. page views per visit

Bounce rate, which is the percentage of visitors that only viewed one page of the digital environment, can allow us in the future to analyze what we have been doing wrong, and what we can do in terms of usability to persuade visitors to move past the landing page. This was widely talked about in our CSC-648 class lectures where the professor mentioned that as a web developer, we need good UI/UX design because customers are not willing to take the time to stay too long in any one web page if it is difficult to follow and not interesting or engaging to them.

The average session duration is extremely significant as well, because it can let us track the length of customer interaction within each item in the webpage and this can lead to usability improvements or making the site more like a certain part that is being preferred or utilized more by visitors.

Page views is also beneficial to find out the amount of current or potential customers that are visiting the website or mobile app and where they are coming from in the world.



7) Project Management:

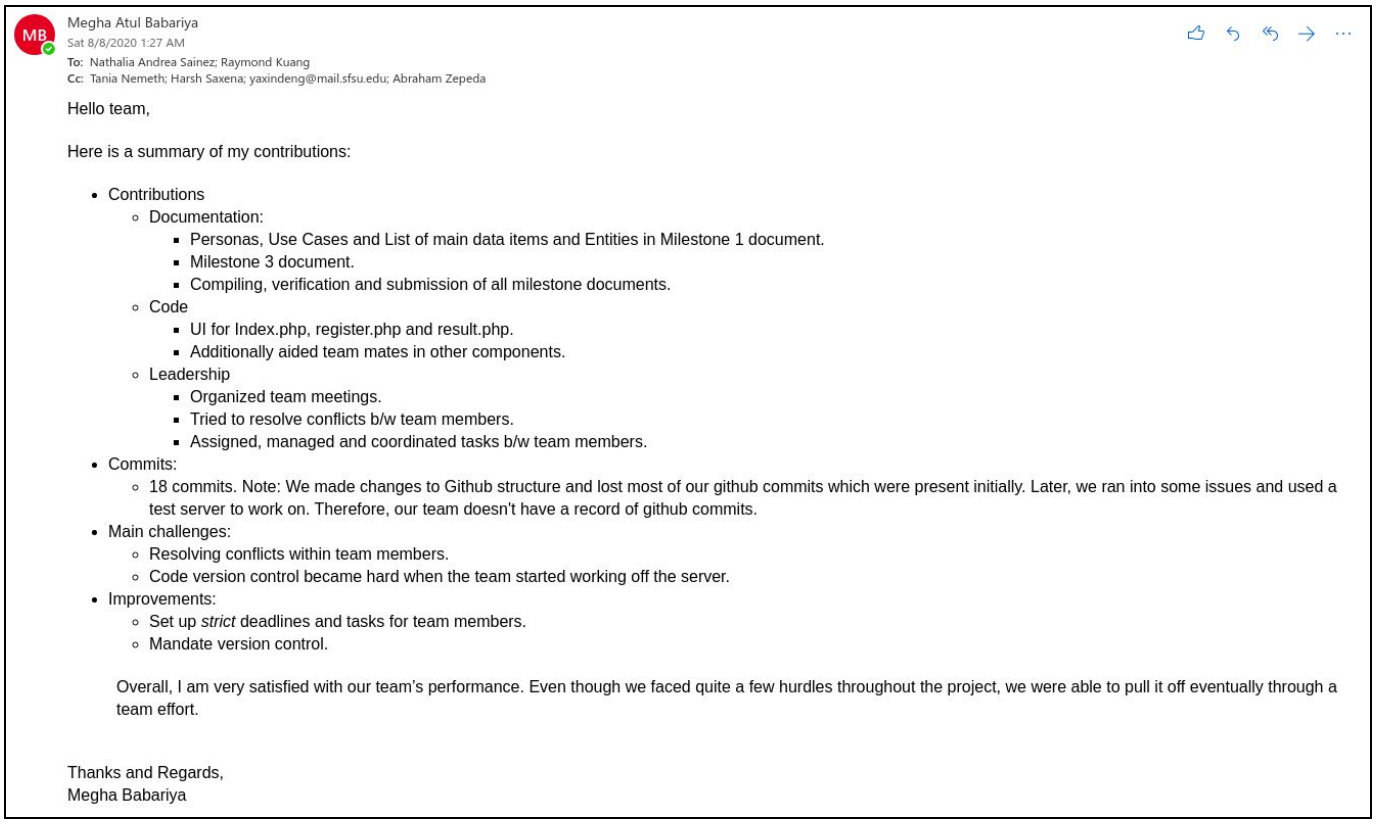
Trello required you to login every single time and navigate their cumbersome UI. We went with Google Sheets to track deployments, manage tasks, and organize the file hierarchy of the site.

	Webpage Dev	Status	Controller D	Status	File Name	Notes
Home	**Megha**	Completed	**Raymond**	Completed	index.php	
Listing - search	**Megha**	Completed	**Raymond**	Completed	listing_search.php	need to convert hard coded elements to be created
Listing - display 1 listing in detail	**Abraham**	Completed	**Raymond**	Completed	listing_detailed.php	
Listing - insert	**Abraham**	Completed	**Raymond**	Completed	listing_insert.php	
Listing - manage own listing	**Yaxin**	Completed	**Raymond**	Completed	listing_manage.php	
Listing - update own listing	**Abraham**	Completed	**Raymond**	Completed	listing_update_onLoad.php, listing_update_onSubmit.php	
Listing - delete	**Abraham**	Completed	**Raymond**	Completed	listing_delete.php	
Listing Approval (Administrator Dashboard)					done via workbench or phpMyAdmin	
Sign In	**Nathalia**	Completed	**Raymond**	Completed	login.php, logout.php	
Registration	**Raymond**	Completed	**Raymond**	Completed	register.php, /controller/register.php	
Messaging- inbox	**Nathalia**	Completed	**Tania**	Completed	GatorMessage.php, messaging.php (small php part)	
Messaging - view message/mark as opened						
Send Message Popup			**Tania**	Work In Progress		
About Team	**Nathalia**	Completed		Completed	/views/about	
Profile	**Yaxin**	Completed	**Raymond**	Completed	profile_onSubmit.php, profile_onLoad.php	
Header Login Button - display name after login	**Tania**	Completed	**Tania**	Completed	header_registered.php, header_guest.php	
Must be logged in page	**Nathalia**	Completed				
Roommate Finder - search	**Nathalia**	Work In Progress	**Raymond**	Not Started		
Roommate Finder - results			**Raymond**	Not Started		
Images: upload header image in create listing	**Tania**	Completed	**Tania**	Completed	listing_insert.php, GatorImages.php, new_listing.php	
Images: uploaded additional images in create listing	**Tania**	Completed	**Tania**	Completed	listing_insert.php, GatorImages.php	
Images: replace header image in edit_listing	**Tania**	Completed	**Tania**	Completed	listing_update.php, GatorImages.php, listing_edit.php,	
Detailed listing view: Display distance to sfsu	**Abraham**	Completed	**Tania**	Completed	GatorListing.php (in insert function)	

Deployment	Feature owner(s)	Feature	File	Type	Comments
	Nathalia	Log in UI Modal	index.php	Add	created a simple pop up that allows users to login into GatorHub
	Nathalia	Footer UI	footer.php	Add	created a simple footer that contains links to contact, about, and FAQ.
	Nathalia	Contact UI	contact.php	Add	created a simple form in which users can submit feedback/comments/questions to admin
	Megha	Home Page	index.php	Add	created a sample home page with hard coded listing cards
	Nathalia	FAQ UI	faq.php	Add	created a simple static page which displays generic questions that most users may have
	Nathalia	About Us UI	about.php	Add	created a simple about page which displays our mission statement and all our team members with links to their respective pages
	Nathalia	Header UI/Logo	header.php	Add	created a GatorHub logo and a header which contains links to post, sign up, and login
	Megha	Form Validation	register.php	Modify	added javascript form validation
	Yaxin	Profile	profile.php	Add	new profile page
	Abraham	Profile	profile.php	Add	added information to profile
	Raymond, Nathalia	views folder	/views	Modify	grouped all files by functionality into folders, need to fix all redirects and includes
	Abraham	Carrousel and Maps	listinginfo.php	Add	added initial file with carousel of images and google map integration
	Abraham	Create/edit/delete listing	createlisgintinfo.php	Add	added initial code being able to create a new listing using a pop-up modal, editing a listing via sys_id, and deleting via sys_id
	Megha	Listing Cards UI	index.php results.php	Modify	new css for listing cards, resize card images
	Abraham	Google API Distance Mat	testmap.php	Add	added file to test API integration of the distance matrix for googe DM
	Raymond	User Registration	register.php	Modify	relabeled name attributes for form fields, disabled email validation due to bug, removed address fields
	Raymond	controller for registration	insertUser.php	Add	registers new user to the database, auto login the user (enumerates session varaibles), redirect to index.php
	Abraham	Google API Finalized	testmap.php	Modify	finalized code via test data to be used in display of maps with Origin and Destination pins in listinginfo.php
	Raymond	controller for login	login.php	Modify	removed address session variable setting, set redirecting to index.php after successful script run
	Nathalia	Profile	profile.php	Modify	reworked profile page into a simple form, made email read-only
	Tania, Raymond	Header File logic	header.php	Modify	added session_start() so every page can access session variables, logo will redirect to index.php
	Abraham	Image retrieval/storage	createlisgintinfo.php	Add	added javascript for direct image retrieval/uploading so that images could be seen in the carousel in listinginfo.php
	Tania, Raymond	Header File logic	header_loggedin.php	Modify	instead of displaying user's name, will show 'Profile' which redirects to profile.php
	Tania, Raymond	Header File logic	header_loggedout.php	Modify	added comments to make file easier to understand
	Abraham	Google Maps distance	MapDistance.php	Add	Google Maps integration for distance calculations
	Abraham	New listing	new_listing.php	Modify	modified new_listing.php by adding more validation fields, drop down states, and fixed problem that it was writing "city" to "state"
	Megha	Search Filter Bar	within results.php	Modify	an addition to the search bar for filtering functionality
	Raymond	Search Filter Bar	within results.php	Modify	controller for filtering and sorting
	Abraham	Image Controller Code	listing_insert.php, listin	Modify	Found bug in image controlled code and fixed it so that distance_to_sfsu would work as "city" to "state" issue was resolved
	Raymond	Support for OR Querying	GatorHub.php	Modify	added new back-end to allow OR queries in addition to the standard AND
	Tania	Form Validation	register.php	Modify	remove hard coded @mail.sfsu.edu and fixed form validation
	Tania	Image Controller Code	listing_insert.php, listin	Modify	Controller for uploading images when you create a listing or when you edit an existing one
	Tania	Image Upload logic	GatorImages.php, new	Modify	Allows user to upload images to their listing when they create one, handles uploads for header and multiple images (up to 20).
	Abraham	Listing information age	listinginfo.php	Modify	Modified listing info one last time to remove modal carousel, remove extra js code to retrieve images, and used bootstrap better
	Abraham	Filled Database with data	sql database	Modify	Went to zillow and retrieved many addresses and images to fill out own db with, and further test code.
	Abraham	Selenium Testing	various actions	Add	Completed extensive selenium testing of searches, user registrations, and listing creations/editions/deletions.

8) Team Member self assessment and Contributions:

1) Megha Babariya (Team-Lead):



Megha Atul Babariya
Sat 8/8/2020 1:27 AM
To: Nathalia Andrea Sainez; Raymond Kuang
Cc: Tania Nemeth; Harsh Saxena; yaxindeng@mail.sfsu.edu; Abraham Zepeda

Hello team,

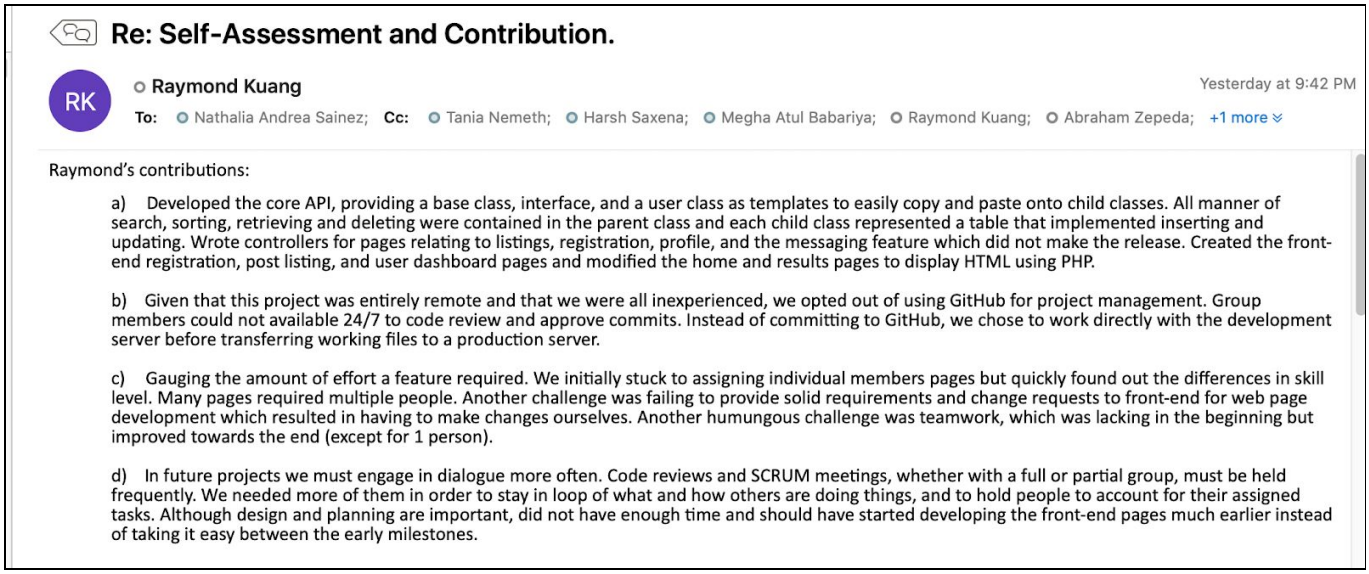
Here is a summary of my contributions:

- Contributions
  - Documentation:
    - Personas, Use Cases and List of main data items and Entities in Milestone 1 document.
    - Milestone 3 document.
    - Compiling, verification and submission of all milestone documents.
  - Code
    - UI for Index.php, register.php and result.php.
    - Additionally aided team mates in other components.
  - Leadership
    - Organized team meetings.
    - Tried to resolve conflicts b/w team members.
    - Assigned, managed and coordinated tasks b/w team members.
- Commits:
  - 18 commits. Note: We made changes to Github structure and lost most of our github commits which were present initially. Later, we ran into some issues and used a test server to work on. Therefore, our team doesn't have a record of github commits.
- Main challenges:
  - Resolving conflicts within team members.
  - Code version control became hard when the team started working off the server.
- Improvements:
  - Set up *strict* deadlines and tasks for team members.
  - Mandate version control.

Overall, I am very satisfied with our team's performance. Even though we faced quite a few hurdles throughout the project, we were able to pull it off eventually through a team effort.

Thanks and Regards,
Megha Babariya

2) Raymond Kuang (Backend-Lead)



Re: Self-Assessment and Contribution.

RK **Raymond Kuang** Yesterday at 9:42 PM

To: Nathalia Andrea Sainez; **Cc:** Tania Nemeth; Harsh Saxena; Megha Atul Babariya; Raymond Kuang; Abraham Zepeda; +1 more

Raymond's contributions:

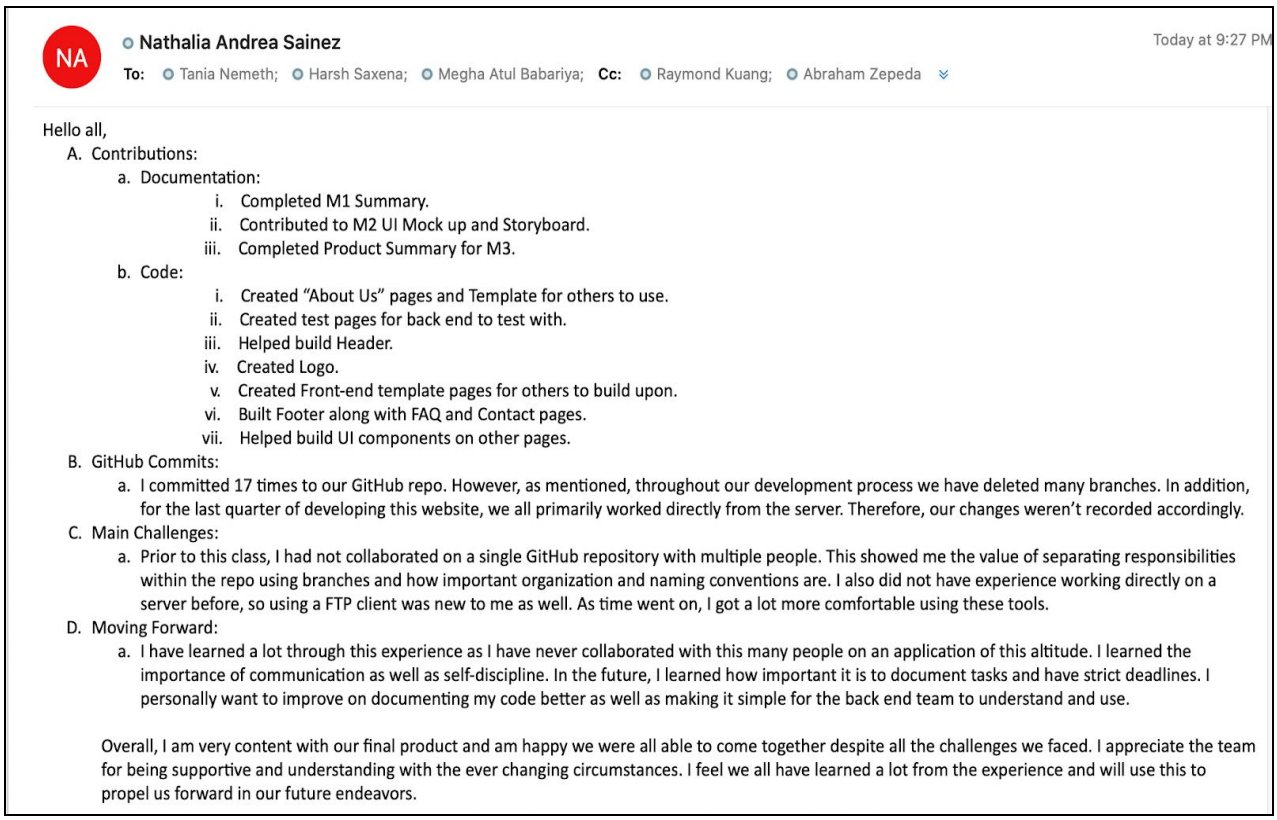
a) Developed the core API, providing a base class, interface, and a user class as templates to easily copy and paste onto child classes. All manner of search, sorting, retrieving and deleting were contained in the parent class and each child class represented a table that implemented inserting and updating. Wrote controllers for pages relating to listings, registration, profile, and the messaging feature which did not make the release. Created the front-end registration, post listing, and user dashboard pages and modified the home and results pages to display HTML using PHP.

b) Given that this project was entirely remote and that we were all inexperienced, we opted out of using GitHub for project management. Group members could not available 24/7 to code review and approve commits. Instead of committing to GitHub, we chose to work directly with the development server before transferring working files to a production server.

c) Gauging the amount of effort a feature required. We initially stuck to assigning individual members pages but quickly found out the differences in skill level. Many pages required multiple people. Another challenge was failing to provide solid requirements and change requests to front-end for web page development which resulted in having to make changes ourselves. Another humungous challenge was teamwork, which was lacking in the beginning but improved towards the end (except for 1 person).

d) In future projects we must engage in dialogue more often. Code reviews and SCRUM meetings, whether with a full or partial group, must be held frequently. We needed more of them in order to stay in loop of what and how others are doing things, and to hold people to account for their assigned tasks. Although design and planning are important, did not have enough time and should have started developing the front-end pages much earlier instead of taking it easy between the early milestones.

3) Nathalia Sainez (Frontend-Lead):



NA **Nathalia Andrea Sainez** Today at 9:27 PM

To: Tania Nemeth; Harsh Saxena; Megha Atul Babariya; **Cc:** Raymond Kuang; Abraham Zepeda

Hello all,

A. Contributions:
   a. Documentation:
      i. Completed M1 Summary.
      ii. Contributed to M2 UI Mock up and Storyboard.
      iii. Completed Product Summary for M3.
   b. Code:
      i. Created "About Us" pages and Template for others to use.
      ii. Created test pages for back end to test with.
      iii. Helped build Header.
      iv. Created Logo.
      v. Created Front-end template pages for others to build upon.
      vi. Built Footer along with FAQ and Contact pages.
      vii. Helped build UI components on other pages.

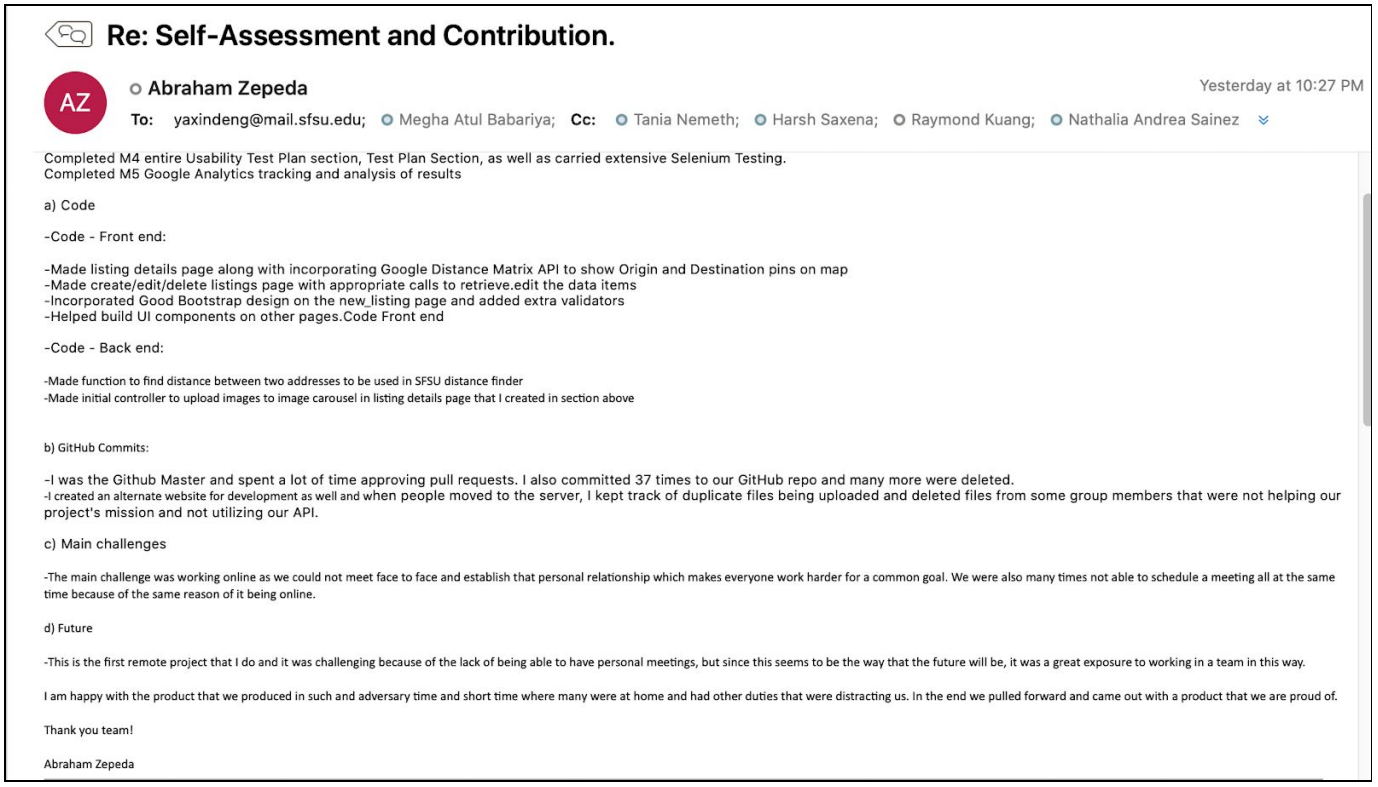
B. GitHub Commits:
   a. I committed 17 times to our GitHub repo. However, as mentioned, throughout our development process we have deleted many branches. In addition, for the last quarter of developing this website, we all primarily worked directly from the server. Therefore, our changes weren't recorded accordingly.

C. Main Challenges:
   a. Prior to this class, I had not collaborated on a single GitHub repository with multiple people. This showed me the value of separating responsibilities within the repo using branches and how important organization and naming conventions are. I also did not have experience working directly on a server before, so using a FTP client was new to me as well. As time went on, I got a lot more comfortable using these tools.

D. Moving Forward:
   a. I have learned a lot through this experience as I have never collaborated with this many people on an application of this altitude. I learned the importance of communication as well as self-discipline. In the future, I learned how important it is to document tasks and have strict deadlines. I personally want to improve on documenting my code better as well as making it simple for the back end team to understand and use.

Overall, I am very content with our final product and am happy we were all able to come together despite all the challenges we faced. I appreciate the team for being supportive and understanding with the ever changing circumstances. I feel we all have learned a lot from the experience and will use this to propel us forward in our future endeavors.

4) Abraham Zepeda (Github-Master):



Re: Self-Assessment and Contribution.

AZ **Abraham Zepeda** Yesterday at 10:27 PM

To: yaxindeng@mail.sfsu.edu; Megha Atul Babariya; **Cc:** Tania Nemeth; Harsh Saxena; Raymond Kuang; Nathalia Andrea Sainez

Completed M4 entire Usability Test Plan section, Test Plan Section, as well as carried extensive Selenium Testing.
Completed M5 Google Analytics tracking and analysis of results

a) Code

-Code - Front end:

-Made listing details page along with incorporating Google Distance Matrix API to show Origin and Destination pins on map
-Made create/edit/delete listings page with appropriate calls to retrieve.edit the data items
-Incorporated Good Bootstrap design on the new_listing page and added extra validators
-Helped build UI components on other pages.Code Front end

-Code - Back end:

-Made function to find distance between two addresses to be used in SFSU distance finder
-Made initial controller to upload images to image carousel in listing details page that I created in section above

b) GitHub Commits:

-I was the Github Master and spent a lot of time approving pull requests. I also committed 37 times to our GitHub repo and many more were deleted.
-I created an alternate website for development as well and when people moved to the server, I kept track of duplicate files being uploaded and deleted files from some group members that were not helping our project's mission and not utilizing our API.

c) Main challenges

-The main challenge was working online as we could not meet face to face and establish that personal relationship which makes everyone work harder for a common goal. We were also many times not able to schedule a meeting all at the same time because of the same reason of it being online.

d) Future

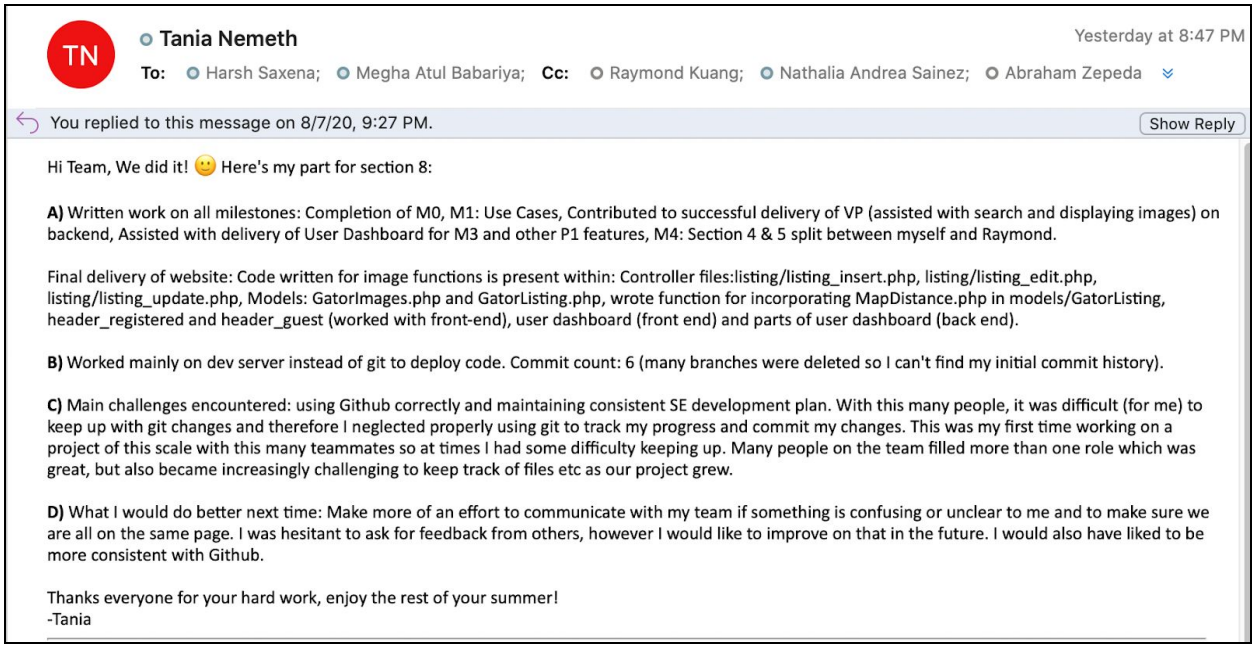
-This is the first remote project that I do and it was challenging because of the lack of being able to have personal meetings, but since this seems to be the way that the future will be, it was a great exposure to working in a team in this way.

I am happy with the product that we produced in such and adversary time and short time where many were at home and had other duties that were distracting us. In the end we pulled forward and came out with a product that we are proud of.

Thank you team!

Abraham Zepeda

5) Tania Nemeth (Backend Member) :



TN **Tania Nemeth** Yesterday at 8:47 PM

To: Harsh Saxena; Megha Atul Babariya; **Cc:** Raymond Kuang; Nathalia Andrea Sainez; Abraham Zepeda

You replied to this message on 8/7/20, 9:27 PM. Show Reply

Hi Team, We did it! 🙂 Here's my part for section 8:

A) Written work on all milestones: Completion of M0, M1: Use Cases, Contributed to successful delivery of VP (assisted with search and displaying images) on backend, Assisted with delivery of User Dashboard for M3 and other P1 features, M4: Section 4 & 5 split between myself and Raymond.

Final delivery of website: Code written for image functions is present within: Controller files:listing/listing_insert.php, listing/listing_edit.php, listing/listing_update.php, Models: GatorImages.php and GatorListing.php, wrote function for incorporating MapDistance.php in models/GatorListing, header_registered and header_guest (worked with front-end), user dashboard (front end) and parts of user dashboard (back end).

B) Worked mainly on dev server instead of git to deploy code. Commit count: 6 (many branches were deleted so I can't find my initial commit history).

C) Main challenges encountered: using Github correctly and maintaining consistent SE development plan. With this many people, it was difficult (for me) to keep up with git changes and therefore I neglected properly using git to track my progress and commit my changes. This was my first time working on a project of this scale with this many teammates so at times I had some difficulty keeping up. Many people on the team filled more than one role which was great, but also became increasingly challenging to keep track of files etc as our project grew.

D) What I would do better next time: Make more of an effort to communicate with my team if something is confusing or unclear to me and to make sure we are all on the same page. I was hesitant to ask for feedback from others, however I would like to improve on that in the future. I would also have liked to be more consistent with Github.

Thanks everyone for your hard work, enjoy the rest of your summer!
-Tania

6) Harsh Saxena (Backend Member):

From: Harsh Saxena <hsaxena@mail.sfsu.edu>
Sent: Friday, August 7, 2020 4:40 PM
To: Megha Atul Babariya <mbabariya@mail.sfsu.edu>
Cc: Raymond Kuang <rkuang1@mail.sfsu.edu>; Tania Nemeth <tnemeth@mail.sfsu.edu>; Nathalia Andrea Sainez <nsainez@mail.sfsu.edu>; Abraham Zepeda <azepeda@mail.sfsu.edu>
Subject: Self-Assessment and Contribution.

Hi Team,

a) Initially, I had a lot of contributions to the team project specifically to the following feature development including 'logic':

a. About Us Template Page
b. Initial design of Database and Tables
c. Login/Logout backend PHP coding
d. Registration backend PHP coding
e. Property Search coded in PHP
f. Property Listing coded in PHP

Later on, after the code style change, I coded the below feature.

g. Session-based page redirect coding in PHP

b) Number of submissions I made to GitHub team Dev. branch
I made 18 GitHub commits and help out merging the code to master
Additionally, in the new code repository structure, I made 2 commits.

c) There was no e-mail communication among team members. All the communication can be taken from Discord.

Thanks
Harsh Saxena
SFSU SID#: 917461368
Phone Number: (510)-980-3111
San Francisco State University
Bachelor of Science In Computer Science(Student)

7) Yaxin Deng (Frontend Member):

YD

Yaxin Deng
Sat 8/8/2020 1:46 AM

To: Megha Atul Babariya
Cc: Harsh Saxena; Tania Nemeth; Raymond Kuang; Nathalia Andrea Sainez

a) His/her contributions to team project and teamwork (technical and any other) in no more than half a page –list item format is OK.
Wrote some front-end webpages, wrote my personal profile page, finished some parts of milestones, make some new listings.
b) Number of submissions he/she made to GitHub team Dev. Branch
I wasn't too familiar with how to use GitHub, so made some mistake earlier. Submitted to the wrong branch, with teammates' helps, I was finally able to use it in the right way.
c)One brief paragraph on main challenges he/she encountered in team project.
I was a bit unfamiliar with the front end and back end service at first, never had this kind of software engineering project experience before, and worked as a team. There were technical issues as well, and I wasn't able to access the service, got confused many times. Even though there were plenty of puzzles, with teammates' helps, they were solved eventually and I learned a lot in this meaningful experience, which I appreciate it.
d)One brief paragraph on what would he/she do better next time based on what was learned in the class about SE management and processes.
I learned a lot in this team project, and learned how to communicate with teammates. There were many confusions and misunderstandings. This was the first time that I worked in a software engineering project with both back-end and front-end, and worked with a team. So next time I will be more familiar with the entire process and know what should I do in each step and milestone, with the knowledge I gained in this class, I learned what should I do in each role of project management, quality management, people Management, cost Management, process management and improvement, and I learned more about QA test and how to test it which I was always interested in working this area.

...
