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. <u>Functional Requirements</u> :

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

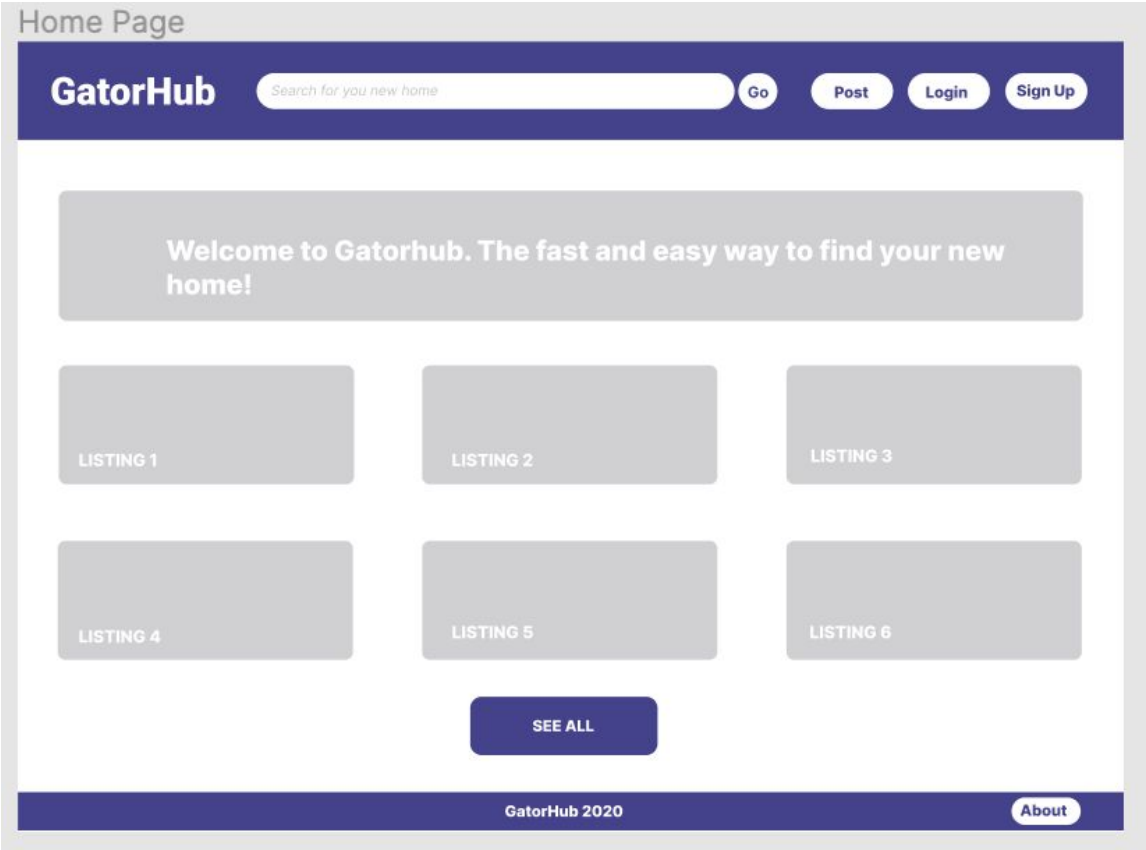
1. As a special feature that differentiates our product, a user will be able to register under certain tags denoting his gender, major, and interests, to help him potentially find similar roommates with

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.

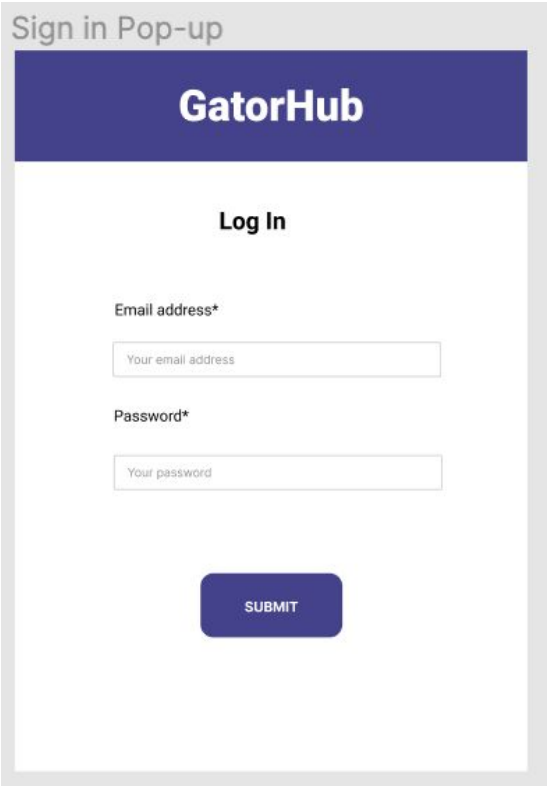


Post button

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.

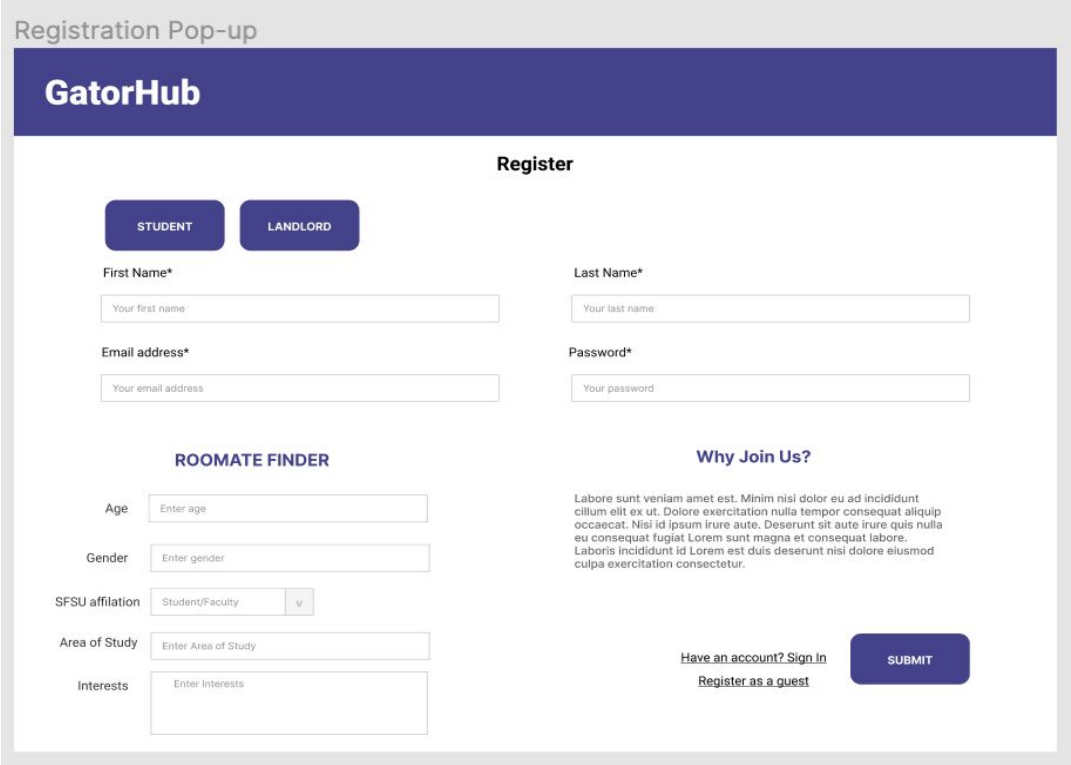
Login button

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.



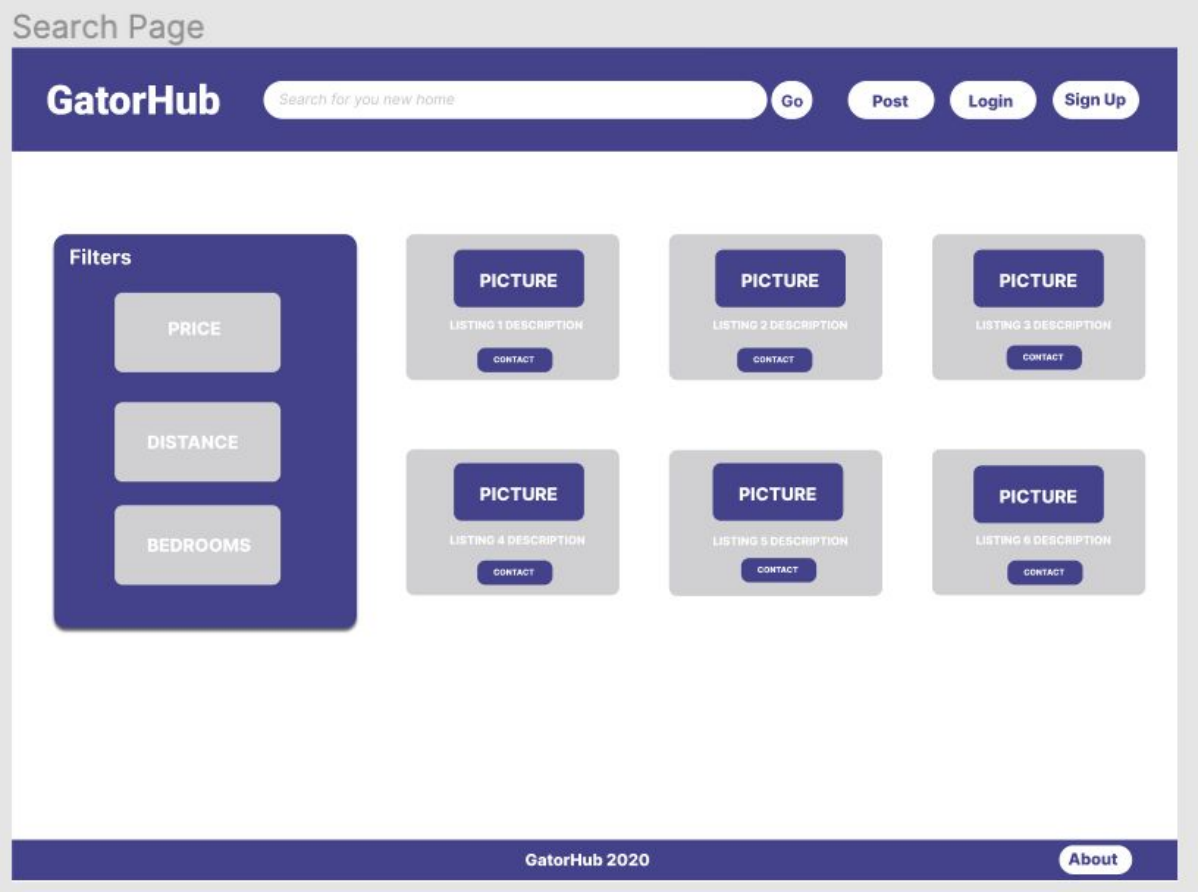
Sign Up button

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.



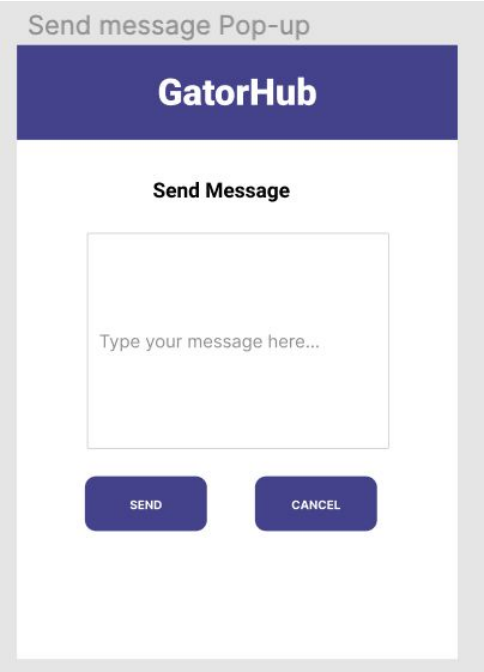
Search results page

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.



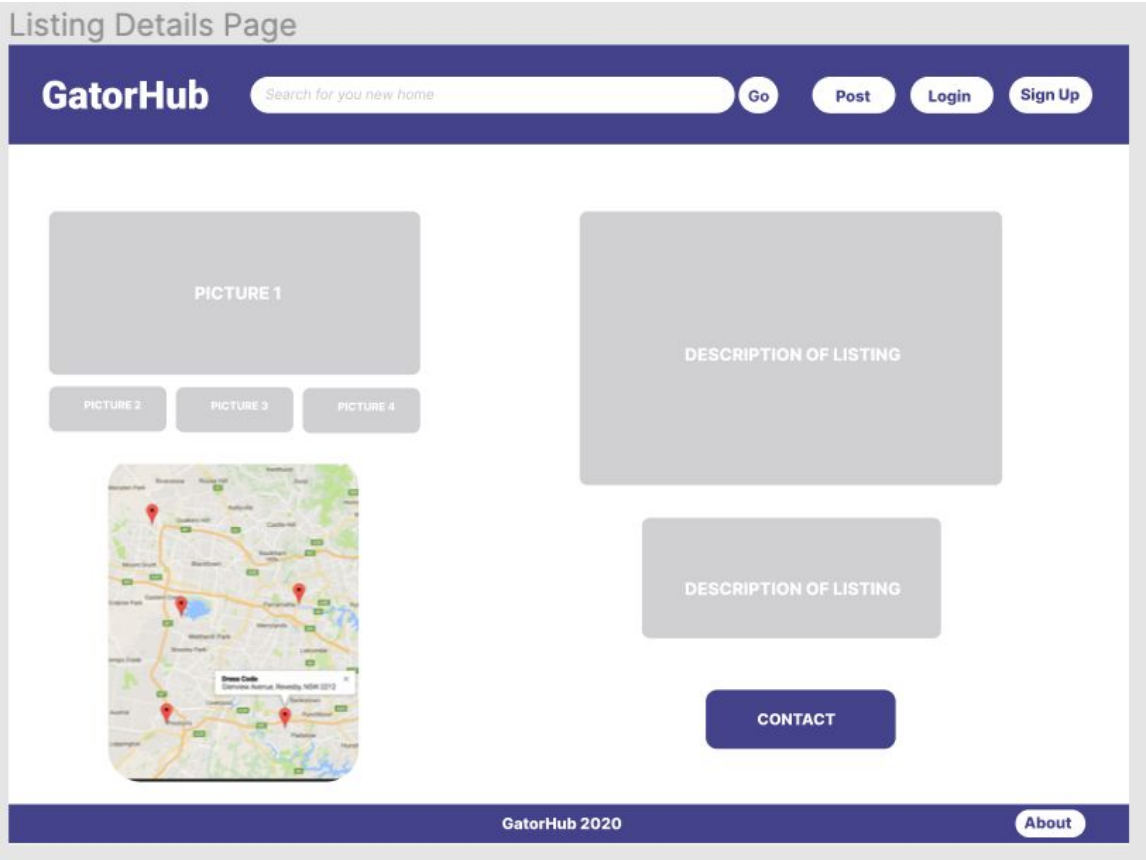
Send message pop-up

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.



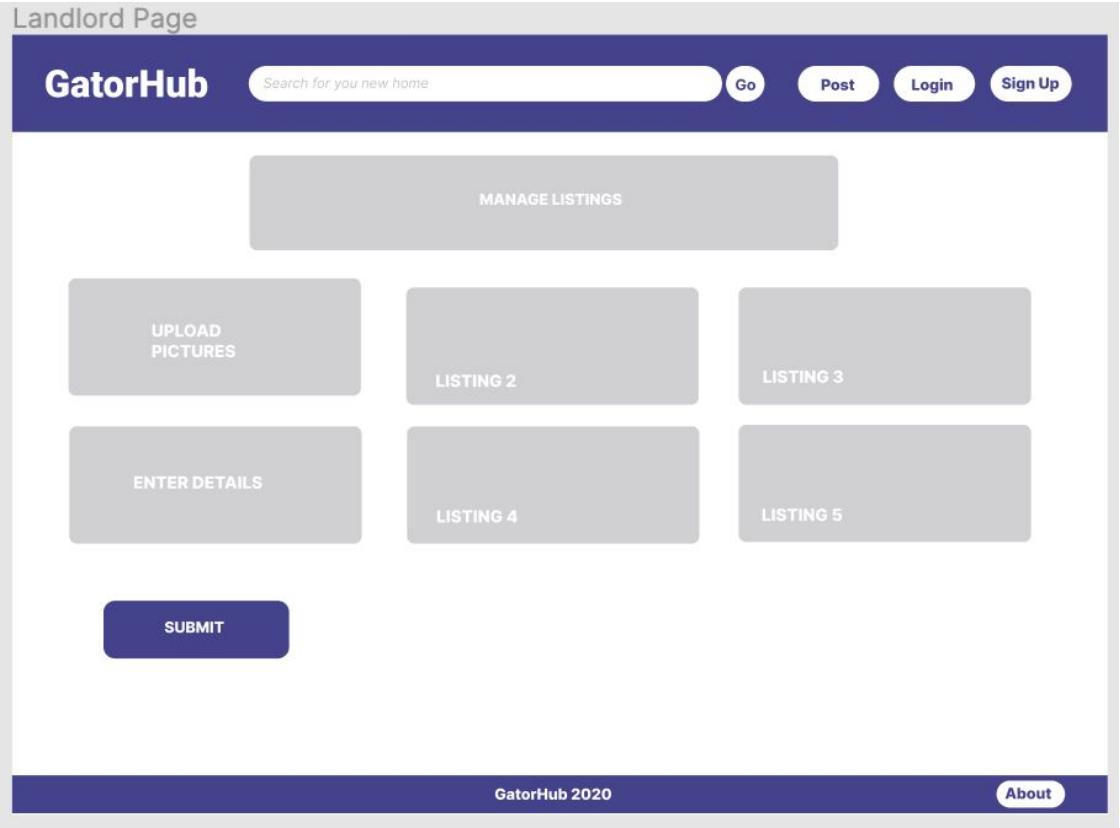
Listing Details page

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.



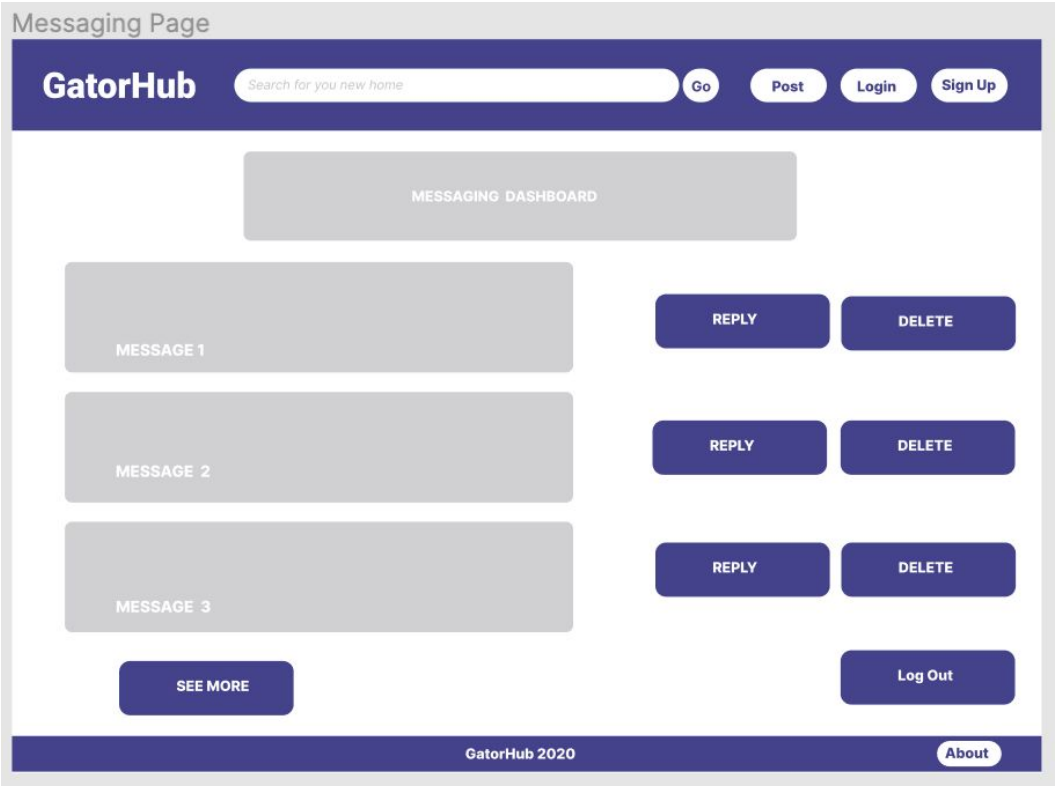
Landlord page

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



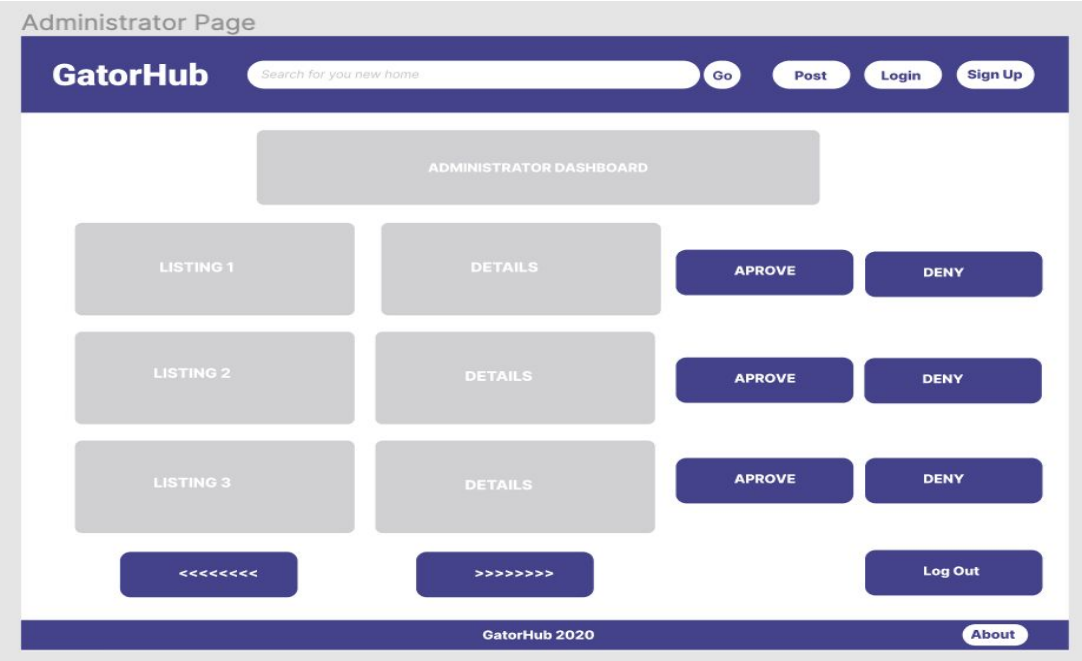
Messaging page

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



Administrator page

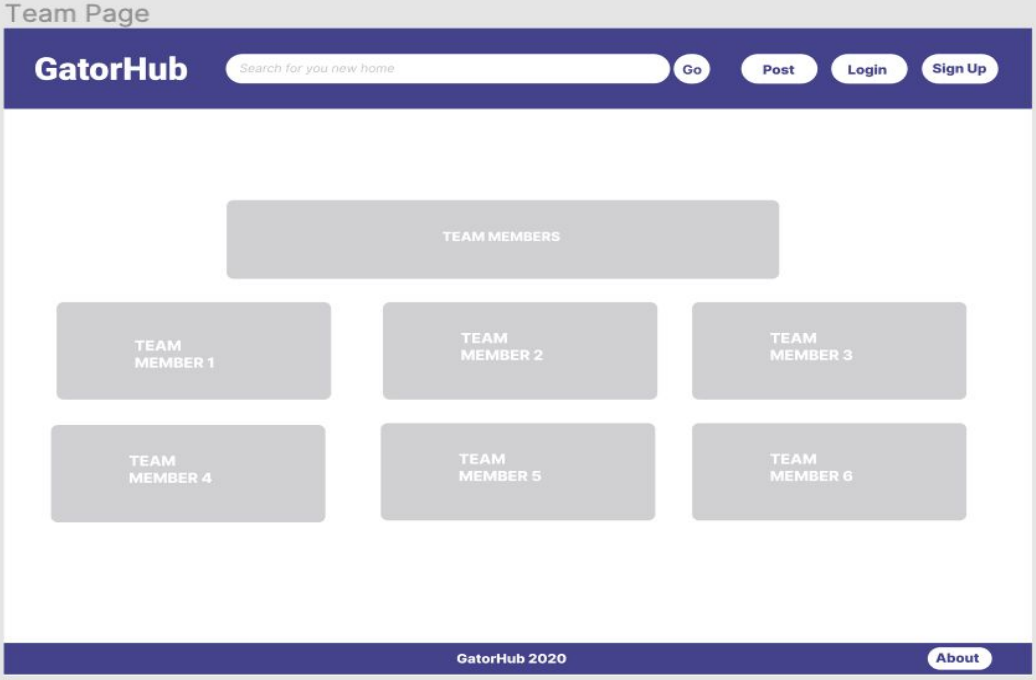
1. If the user is the administrator, upon login in through the normal method, his admin dashboard will be displayed with the listings that he either needs to approve or reject.



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.

Team Page


GatorHub
Search for you new home
Go
Post
Login
Sign Up
TEAM MEMBERS
TEAM MEMBER 1
TEAM MEMBER 2
TEAM MEMBER 3
TEAM MEMBER 4
TEAM MEMBER 5
TEAM MEMBER 6
GatorHub 2020
About

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.
