

Software Engineering CSC 648/848 Spring 2021

Gator Connection

<https://www.gator-connection.com/>

A One Stop Website for SF State Gators

Team 05

Team Lead/Github Master: Alec Stephen Tenefrancia alectene@mail.sfsu.edu

Frontend Lead: Lakshita Chugh

Backend Lead: Angelo Gloria Reyes

Frontend member: Bikram Tamang

Backend member/Document Master: Benjamin Patrick Kao

Frontend member: Carmen Denisse Paisano

Backend member: Jiaxin Yu

Milestone 5

May 20, 2021

Milestone/Version	Date
M5V1	05/20/21
M4V2	05/14/21
M4V1	05/13/21
M3V2	04/25/21
M3V1	04/22/21
M2V2	04/24/21
M2V1	04/01/21
M1V2	03/09/21
M1V1	03/04/21

Table of Contents

Table of Contents	2
Product Summary	5
4) Itemized List of Priority 1 Requirements:	
Priority 1 Requirements:	5
Milestone V2 Documents (M1 -M4)	9
Milestone 1 V2	9
Table of Contents	10
Executive Summary	11
Main Use Cases	13
Main Data Items & Entities	26
List of Functional Requirements	28
List of Non-Functional Requirements	32
Competitive Analysis	35
High Level System Architecture and Technologies Used	38
Team Contributions	39
Team CheckList	42
Milestone 2 V2	
Software Engineering CSC 648/848 Spring 2021	43
Table of Contents	44
Data Definitions V2	45
Functional Requirements V2	48
Priority 1 Requirements:	48
Priority 2 Requirements:	51
Priority 3 Requirements:	52
UI Mockups and Storyboards	54
High Level Database Architecture and Organization	81
Business Rules	81
Entity, Attribute, Relationship, Domain Descriptions	83
Entity Relationship Diagram (ERD)	85
Database Model/Enhanced Entity Relationship (EER)	87
Database Management System (DBMS)	88
Media Storage	88

Search/Filter Architecture and Implementation	88
High Level APIs and Main Algorithms	89
High Level UML Diagrams	91
High Level Application Network and Deployment Diagrams	93
Project Risks	96
Project Management	98
List of Contributions	100
Milestone 3 V2	
Software Engineering CSC 648/848 Spring 2021	103
Table of Contents	104
Main Data Items & Entities V3	105
Functional Requirements	108
Priority 1 Requirements:	108
Priority 2 Requirements:	111
Priority 3 Requirements:	112
WireFrames	114
High Level Database Architecture and Organization V2	177
High Level Diagrams V2	178
High Level UML Diagram	178
High Level Application Network and Deployment Diagrams	181
List of Contributions	183
Milestone 4 V2	
Software Engineering CSC 648/848 Spring 2021	192
Table of Contents	193
Product Summary	194
Usability Test Plan	199
Test Objectives:	199
Test Description:	200
Usability Task Description:	201
Questionnaire:	209
QA Test Plan	215
Test Objectives:	215
QA Test Plan:	216
QA Test Plan #1:	217
QA Test Plan #2:	224
QA Test Plan #3:	232
Coding Style	238
Coding Style	238

Code Review	239
Self-Check on Best Practices for Security	252
Self-Check: Adherence to Original Non Functional Specs	265
List of Contributions	270
Screenshots of Superior feature	277
The below shows screenshots of our superior housing feature	277
Screenshots of Key DB Tables	284
Screenshots of Trello	287
Team Member Contributions	290
Post Analysis	304

Product Summary

1) Name of Product:

The name of our product is **Gator Connection**, a one stop website for SFSU students for housing, items, announcements, and restaurants near SFSU.

2) What is Unique about our product:

What is unique about our product compared to what other listing services have is our housing listings service. With our housing listings service, the one who posted the listing will have the final say in who he/she chooses to contact. We will do this by implementing a feature where buyers will not have access to any contact information about the listing. Instead they will only have access to the general location of the room, pictures of the room, and any other information about the room posted. Buyers can inquire that they are interested in the housing through a form that will contain information that they write about themselves; however, it will be up to the one who posted the listing to reply to the buyer that he/she is interested in selling them the room. By doing this, sellers of the room can rest easy knowing that they have the choice on who to contact. On top of this, our feature gives sellers the power to prevent their contact information from being shared, unlike other competitors such as Craigslist.

3) URL:

<https://www.gator-connection.com/>

4) Itemized List of Priority 1 Requirements:

Priority 1 Requirements:

- Guest User
 - A guest user shall be able to search for restaurants by title.
 - A guest user shall be able to navigate through announcements.
 - A guest user shall be able to create a student account using his/her unique SFSU email.
 - A guest user shall be able to create an admin account using his/her unique email.
 - A guest user shall be able to create a super user account using his/her unique email.
 - A guest user shall be able to navigate to a map of SFSU.

- A guest user shall be able to search for items on sale by preferred payment.
 - A guest user shall be able to search for items on sale by price.
 - A guest user shall be able to search for items on sale by title.
 - A guest user shall be able to search for housing on sale by preferred payment.
 - A guest user shall be able to search for housing on sale by price.
 - A guest user shall be able to search for housing on sale by title.
 - A guest user shall be able to search for housing on sale by location.
- Registered User
 - A registered user shall have all of the same permissions as guest users.
 - A registered user shall be able to log in to his/her account using his/her unique email.
 - A registered user shall be able to log in to his/her account using many devices.
 - A registered user shall be able to log out of the website.
 - A registered user shall be able to stay logged in if they have not logged out.
 - A registered user shall be able to verify his/her email.
 - A registered user shall be able to post reviews of restaurants/food.
 - A registered user shall be able to make purchase requests for items.
 - A registered user shall be able to see and search housing listings.
 - A registered user shall be able to make a request for a housing listing.
 - A registered user shall be able to edit a housing listing title he/she posted.
 - A registered user shall be able to edit a housing listing description he/she posted.
 - A registered user shall be able to edit a housing listing price he/she posted.
 - A registered user shall be able to change a housing listing image he/she posted.
 - A registered user shall be able to close a housing listing that he/she posted.
 - A registered user shall be able to edit a post about an item for sale that he/she posted.
 - A registered user shall be able to take down an item that he/she put up for sale.
 - A registered user shall be able to create a restaurant request to add a restaurant to the restaurants listing.
 - A registered user shall receive a notification about a house listing being closed if he/she had made a request to the house listing.
 - A registered user shall receive a notification about an approved restaurant that he/she requested to add.
 - A registered user shall receive a notification about a rejected restaurant that he/she requested to add.
 - A registered user shall have a unique registered id.
 - A registered user shall be able to fill out a form for a housing request.
 - A registered user shall receive a notification about purchase requests made on his/her account by email.

- A registered user shall receive a notification about purchase requests made on items he/she posted for sale by email.
 - A registered user shall receive a notification about a house listing request for a house listing that he/she posted that includes information about the person who requested, such as his/her name and email address.
 - A registered user with an approved admin account shall be able to post announcements.
 - A registered user with an approved admin account shall be able to remove an announcement he/she had posted.
- Student Account
 - A non-verified Student account shall prevent registered users from accessing item features.
 - A non-verified Student account shall prevent registered users from accessing housing features.
 - A non-verified Student account shall prevent registered users from accessing restaurant features.
- Admin Account
 - An admin account shall give registered users the ability to post announcements.
- Super User Account
 - A registered user with an approved super user account shall be able to approve newly created admin accounts.
 - A registered user with an approved super user account shall be able to reject newly created admin accounts.
 - A registered user with an approved super user account shall be able to approve newly created super user accounts.
 - A registered user with an approved super user account shall be able to reject newly created super user accounts.
- Sale Item
 - A sale item shall be posted by a registered user.
 - A sale item shall be requested by a registered user.
 - A sale item shall be taken down by the registered user that posted it for sale.
 - When a sale item is taken down, all registered users who made a request for the item shall be notified of the closure by email.
- Housing Listing
 - A housing listing shall be posted by a registered user.
 - A housing listing shall be taken down by the registered user that posted it.
 - A housing listing shall be requested by many registered users.
 - A housing listing shall have a form to fill out for interested users.
 - A housing listing shall have a situation(available/close).

- When a housing listing is taken down, all registered users who made a request shall be notified of the closure by email.
- Housing Listing Form
 - A housing listing form shall be filled out by a registered user.
 - A housing listing form shall fill out their full name.
 - A housing listing form shall fill out their school email.
 - A housing listing form shall fill out their phone number.
 - A housing listing form shall have a section dedicated to “an about” me.
 - A housing listing form shall have an expected day they want to move in.
- Restaurant
 - A restaurant shall have many restaurant reviews.
 - A restaurant shall show other related restaurants.
- Restaurant Request
 - A restaurant request shall be created by one registered user.
 - A restaurant request shall have the name of the restaurant.
 - A restaurant request shall have the location of the restaurant.
 - A restaurant request shall be confirmed/denied and closed by one and only one super user.
- Restaurant Review
 - A restaurant review shall be created by a registered user.
 - A restaurant review shall show the name of the registered user that posted it.
- Announcement
 - An announcement shall be posted by a registered user with an approved admin account.
 - An announcement shall be able to be viewed by one or many users.
 - An announcement shall be able to be taken down by the one who posted it.
 - An announcement shall be categorized by the type of admin account of the registered user that posted it.

Milestone V2 Documents (M1 -M4)

Milestone 1 V2

Software Engineering CSC 648/848 Spring 2021

Gator Connection

A One Stop Website for SF state Gators

Team 05

Team Lead/Github Master: Alec Stephen Tenefrancia alectene@mail.sfsu.edu

Frontend Lead: Lakshita Chugh

Backend Lead: Angelo Gloria Reyes

Document Master/Frontend member: Bikram Tamang

Backend member: Benjamin Patrick Kao

Frontend member: Carmen Denisse Paisano

Backend member: Jiaxin Yu

Milestone 1

Use Cases and High Level Requirements

February 27, 2021

Milestone/Version	Date
M1V2	05/09/21
M1V1	03/04/21

Table of Contents

Table of Contents	10
Executive Summary	13
Main Use Cases	15
Main Data Items & Entities	28
List of Functional Requirements	30
List of Non-Functional Requirements	34
Competitive Analysis	37
High Level System Architecture and Technologies Used	40
Team Contributions	41
Team CheckList	44

Executive Summary

“It’s so hard to find any information from the SFSU website, everything is so cluttered”. There is hardly any class in San Francisco State University where we have not heard these words. Whether it is coming from a STEM major or an Art major, junior, senior or even a professor, they all have something to complain about on the website. Having to visit ten other websites just to find about an event on the campus, getting lost at the front page, not finding the things you are looking for on a section or finding too many things in one, the list goes on. We think that it is time for a change. Therefore, we dared to create a website that addresses all the issues above and beyond. A single website you could visit that would have all the information about announcements and club events and more, including nearby SFSU restaurant information, items for sale, and nearby housing listings all in one place to look at. **Gator Connection**, a social announcement website exclusively for SFSU students.

Gator Connection was created with SFSU students in mind. The difference between **Gator Connection** and other social sites is that you must have a valid SFSU email address to register. By authenticating that the user has a valid SFSU email address by sending an email to that SFSU email address to confirm that they are an SFSU student, you can ensure that all the information posted will be from SFSU students and administrators. That is not to say non SFSU students cannot use **Gator Connection**, far from it. Individuals outside of SFSU can browse through posts made by other SFSU students by just going to the website. With that, they are able to see any upcoming events their friends may be in. The smaller base of members will ensure that all the posts you see online will be from SFSU students, guaranteeing you accurate information. The main thing that **Gator Connection** offers that other SFSU sites do not have is convenience and one stop info. For example, when looking through Facebook for SFSU clubs, students will find that info is scattered everywhere, therefore dissuading them from the task of searching for one that would interest them. On **Gator Connection**, since it is a one stop for all SFSU info, students will easily be able to look through all the many clubs and find one that may interest them. What’s more, they will also be able to see any important info related to that club, such as meeting times, what the club is about, etc. By eliminating unnecessary clutter that other social sites have, **Gator Connection** will be more user friendly for students to use.

On **Gator Connection**, we also connect students on things they may want to sell, whether it be school supplies or possible house listings. What separates **Gator Connection** from other marketing sites is that you can ensure all listings posts are from SFSU students, ensuring a reliable transaction with another student. Furthermore, students can arrange meetups when both parties would be on campus, eliminating hassles such as shipping or travel. This exclusivity for SFSU students also guarantees safety in purchasing a possible new room for sale.

Furthermore, on **Gator Connection**, we know students at SFSU like to go out for food and drinks on campus. With that in mind we want to provide personal SFSU reviews to restaurants students may want to try out. By going to our restaurants section, students can find this information in a place where their fellow students have experienced.

Gator Connection will be a high quality website that is the solution for students that will address the many issues that they face when dealing with decentralized and dispersed information from SFSU. By doing this, Gator Connection will help to improve many students' college experience.

Main Use Cases

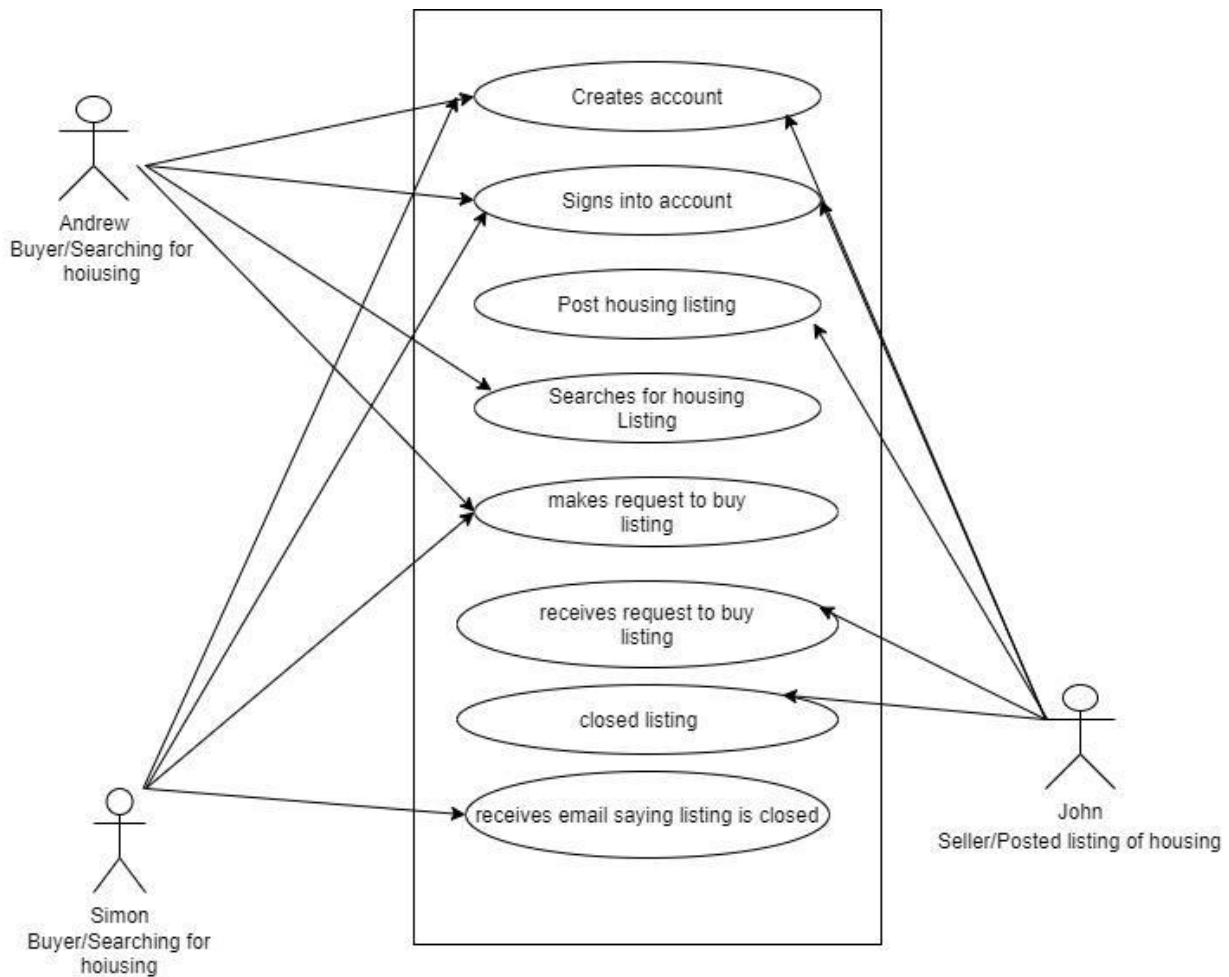
Case 1: Freshman Student looking for housing at SFSU

Actors: Andrew(Buyer, Searching for housing), Simon (Buyer, Searching for housing), John(Seller, Posted listing of housing)

Andrew is a recently admitted student who found out he got accepted for Fall 2021 at SFSU. Finding out that campus is going back to in person instruction, he is considering moving to SF for the college experience. If he can avoid it, he does not want to live in the dorms, as he believes it is over-priced. However, he wants to live with students from SFSU to have that experience of rooming with students and make friends much more easily in his first year.

Andrew knows he can check online on Facebook for house listings, but he has also heard that these listings were not reliable and did not guarantee he will live with other SFSU students. He wanted to find a more reliable way to filter out housing listings to know that any choice he makes will mean he'll be living with SFSU students. Andrew goes to **Gator Connection** to the housing section because he heard that only SFSU students can post housing listings, so it'll guarantee that he can live with other SFSU students. However, he gets blocked with the website informing him that he must be a registered user to have access to see housing listings. Andrew creates an account with his new SFSU email and logs in. He then had access to view the available housing listings. Andrew notices that many of the listings already have requests from other students wanting to buy. Andrew continues to look, and he finds an ideal housing situation, and he decides to make a request to buy. John is the SFSU student who posted the housing listing on **Gator Connection** that Andrew chose. After Andrew makes the buy request, John receives an email notification from **Gator Connection** that a registered user has made a request. The email informed John of Andrew's name and SFSU email so that they could get into contact with one another. Andrew and John start communicating through email, and after a while, they both believe that they will be a good fit as roommates. So, John goes online to **Gator Connection** and goes to his profile posts section. He finds his housing listing post and he closes the posting. When he goes back to the housing section, he sees that his house listing post had correctly shown that it was now closed. Simon, another SFSU student who made a request for John's housing listing, received an email from **Gator Connection** that the housing listing was now closed.

Case 1 Use Case Diagram

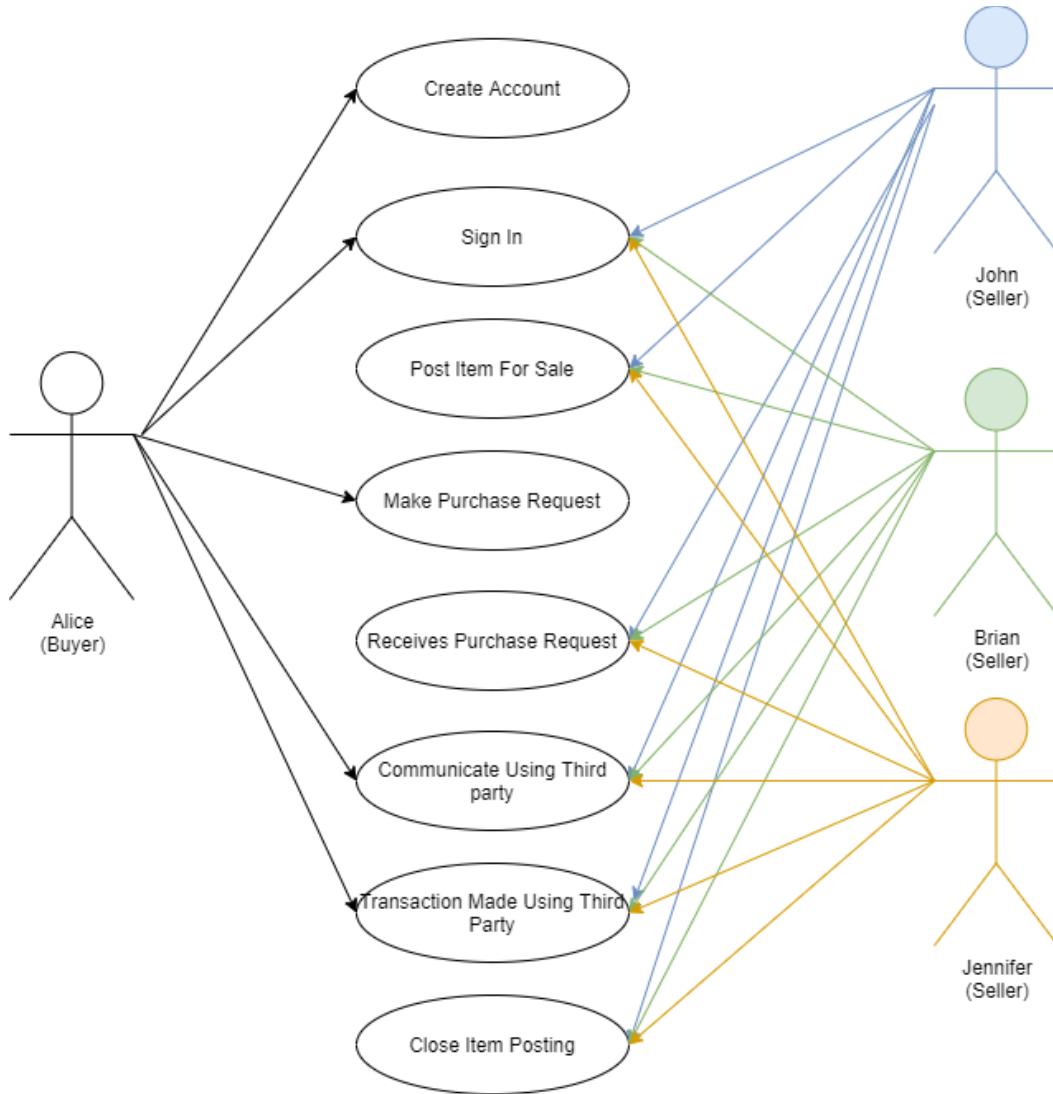


Case 2: Student at SFSU looking for cheaper textbooks

Actors: Alice(Buyer), John(Seller), Brian(Seller), Jennifer(Seller)

Alice is a student at SFSU. She is taking many classes this semester and all her classes require textbooks. Alice did not want to buy brand new textbooks because they would be too expensive for her, so she chose to look on Gator Connection for used, cheaper textbooks. She found one of the textbooks she needed and tried to make a purchase request. However, the website displayed a pop up error notifying her that she was not logged in and asked her to either log in to an existing account or create a new one. Because Alice did not have an account, she decided to create a new one. She had to enter her SFSU email, name, and password to create an account. After she created an account, she was able to make the purchase request. So, Alice continued to send purchase requests for all the textbooks she needed. Later, Alice checked her SFSU email, and received an email from Gator Connection of the request that was made on her account. John, Brian and Jennifer, all students at SFSU were the ones that posted the used textbooks for sale. After Alice placed her order, John, Brian, and Jennifer received an email from Gator Connection that a user wanted to buy their textbooks. The email informed them of Alice's name and email address and the textbooks that she requested to buy. They all got into contact with Alice through email and talked about their preferred payment methods. Finally, Alice paid each of them using the third party services they all chose. Then, John, Brian, and Jennifer all packed up the now sold textbooks and mailed them to Alice, and they all closed their item postings.

Case 2 Use Case Diagram



Case 3: Student at SFSU looking for food on campus, with his two friends outside SFSU accompanying him

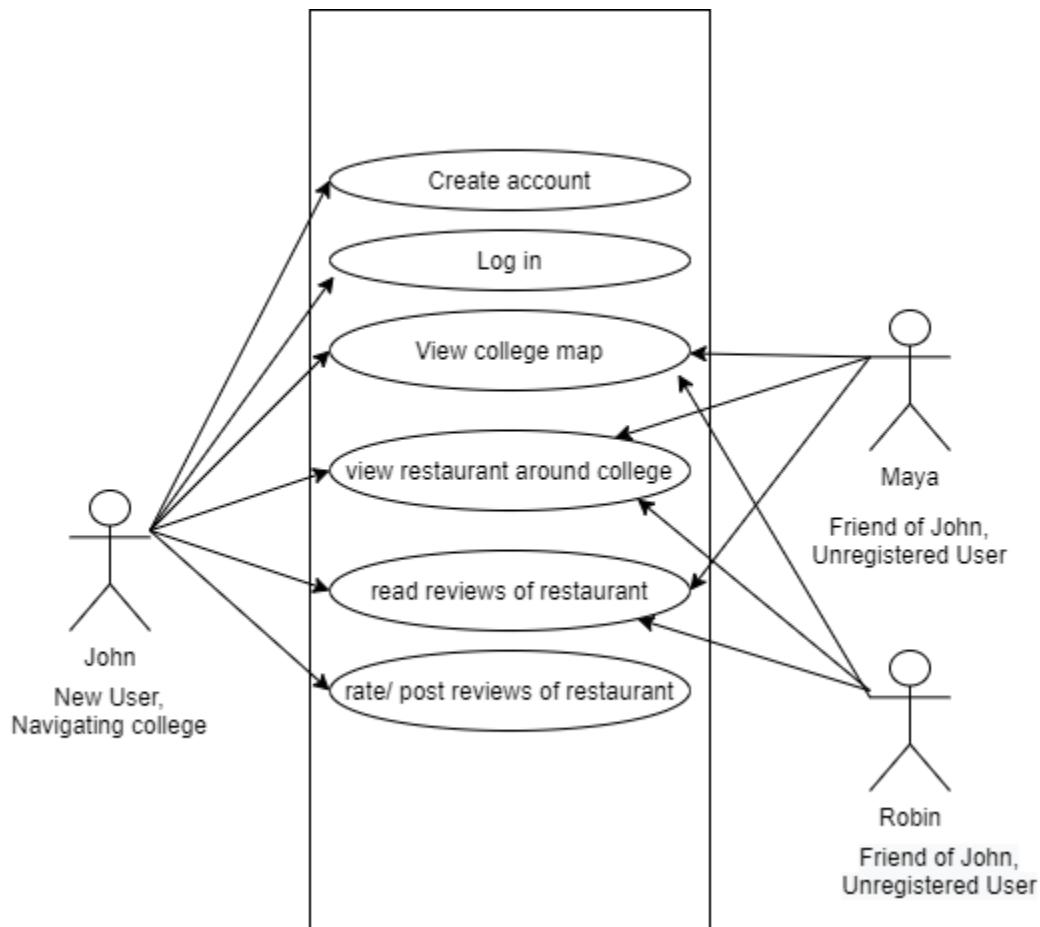
Actors: John(New User), Maya(Unregistered User), Robin(Unregistered User)

John is an incoming freshman and he visits SFSU. He wants to walk around the entire University to become familiar with the buildings, but he does not know any directions. John goes online to Gator Connection and navigates to the virtual map of the college. He is able to see all buildings located in the college and is able to walk around the campus. During this time, he gets really hungry, but he doesn't want to leave yet since he hasn't explored the entire campus yet.. So, he decides he wants to buy lunch at the university. However, he has no idea what kind of food is available or how good it is. He visits Gator Connection and navigates to the restaurant review section. He sorts the restaurants by highest-rated (rated by the students of SFSU) and he decides to go to the highest-rated restaurant on the list. He looks at the map on Gator Connection and he walks over to the restaurant. He bought and ate a sandwich in the restaurant. However, he did not like the sandwich, so he wanted to post a bad review about the restaurant on Gator Connection. When he tried to write a review about the restaurant on Gator Connection, Gator Connection threw an error informing John that he could not post a review without being a registered user. So, John creates an account using his new SFSU email account. Then he tries once again to write a review. Gator Connection allows him to and John posts the negative review on the website for everyone else to see.

Furthermore, two friends of John who go to different schools, Maya and Robin, want to surprise John for his first semester at SFSU. After hearing from him that he got a bad sandwich on campus, they decide to treat him to a restaurant on the SFSU campus, and find him something he may like. Maya and Robin visit Gator Connection and navigate to the restaurant's review section. After sorting it by the highest-rated, they take care to avoid the restaurant that John picked. They then pick the second highest rated restaurant and read the reviews there. Deciding that John may like something there, they decide that is where they want to treat John to. Maya and Robin then navigate to the virtual map of SFSU from Gator Connection to see where the restaurant is. After finding out the location, they then go to John's dorm. After telling John their plans, Maya and Robin take John to the restaurant that they had decided on and eat there.

After their meal, John decides that he enjoyed the meal at this restaurant a lot. He then decides to write a review for this restaurant. He logs into his Gator Connection account rates them highly, while also giving them a positive review.

Case 3 Use Case Diagram

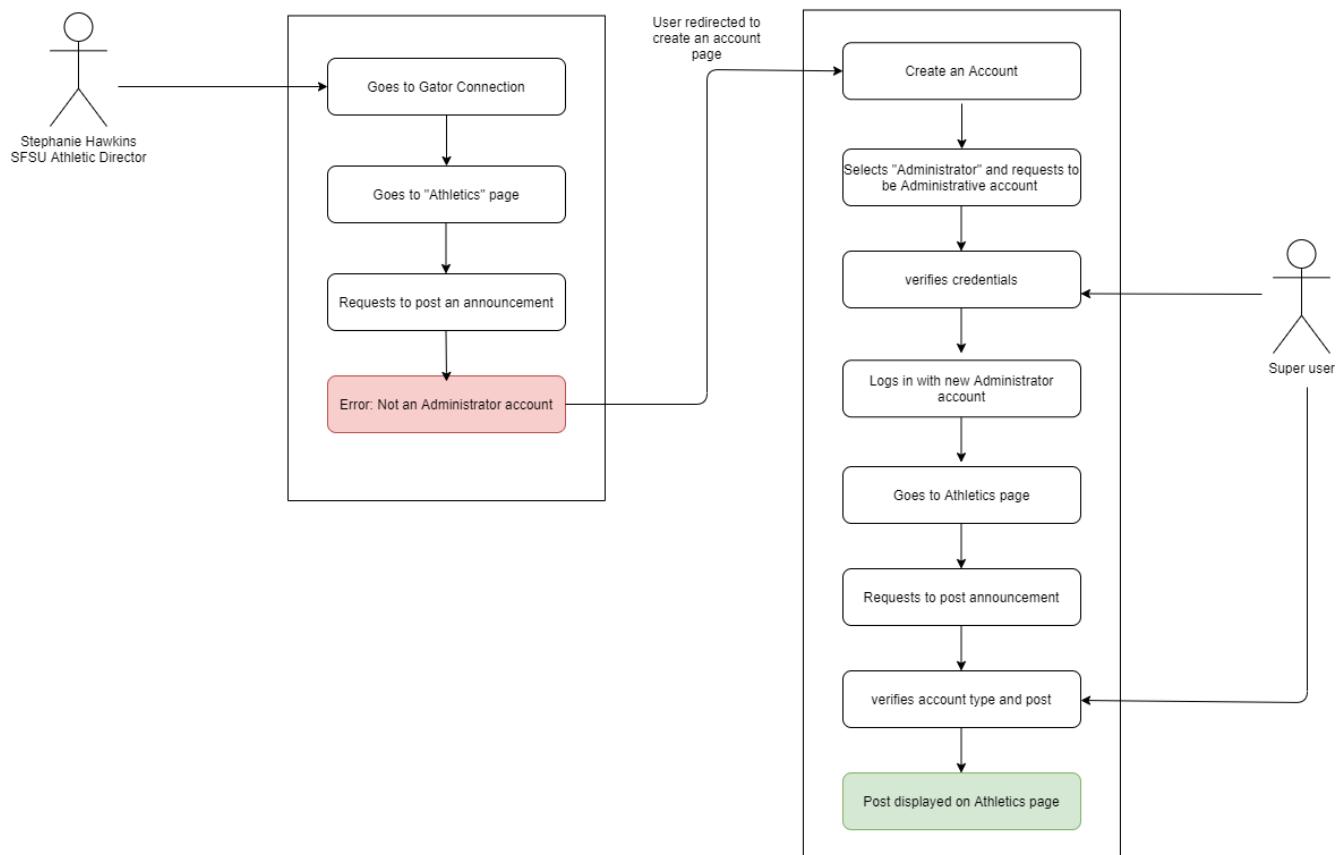


Case 4: Athletic Director at SFSU wants to announce upcoming events

Actors: Stephanie(User, Athletic Director)

Stephanie Hawkins is SFSU's athletic director and wants to announce to the students about an upcoming home basketball game. She heard about Gator Connection and realized that many students use Gator Connection as a hub for news and announcements about SFSU. She goes online to Gator Connection to the "Athletics" section and tries to post an announcement about the basketball game. However, Gator Connection throws an error informing Stephanie that only administrative users are allowed to post announcements on Gator Connection. So, Stephanie created an administrative account using her SFSU email and her position. After Stephanie created an administrative account, she then tried to post an announcement on the "Athletics" section once again and was given access. Once Stephanie finished writing her announcement about the event, she submitted the post. After Stephanie refreshed the "Athletics" section page, her announcement about the basketball game showed up for everyone to see.

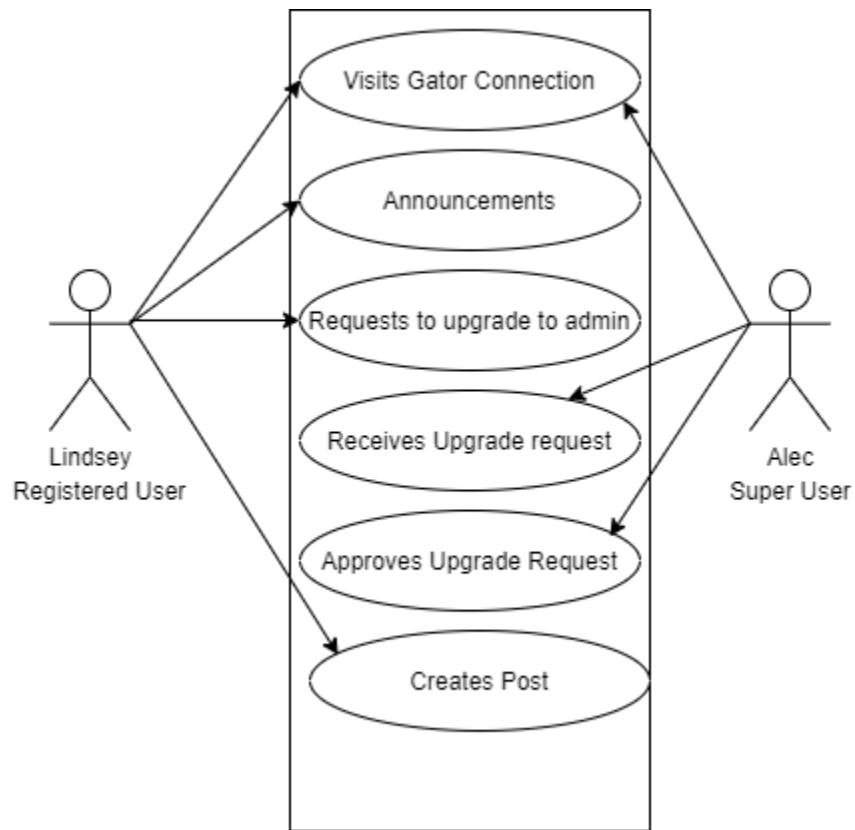
Case 4 Use Case Diagram



Case 5: President of SF Hacks wants to post about her upcoming events**Actors: Lindsey(Registered User), Alec(Super User)**

Lindsey is the new president of SF Hacks and wants to post about her organization's social media, announcements, and future events coming up. She always sees other organizations posting announcements on Gator Connection, so she decides she wants to do the same. She logs into her Gator Connection account, and she tries to post an announcement. However, Gator Connection gives her a pending post, making her realize she does not have full access to the announcements feature as only administrative users can post without having it be pending. She decides to upgrade her account to become an administrative user. She inputs the organization she is a part of, the role she has, how long she would be president, and her organization's email. However, she must wait to get official confirmation from Gator Connection that her account upgrade request was successful. Alec, a super user at Gator Connection, received an email that an account upgrade request had been made by Lindsey. Alec verified that Lindsey's provided information was correct and approved Lindsey's upgrade request. Lindsey gets an email from Gator Connection that her request was successful and her account has been upgraded to have administrator privileges until her role as president is over. So, Lindsey created a post announcing an upcoming SF Hackathon with the event details and submitted her post. Lindsey went to the "Organizations" section of Gator Connection and saw her announcement right away for everyone else to see.

Case 5 Use Case Diagram:

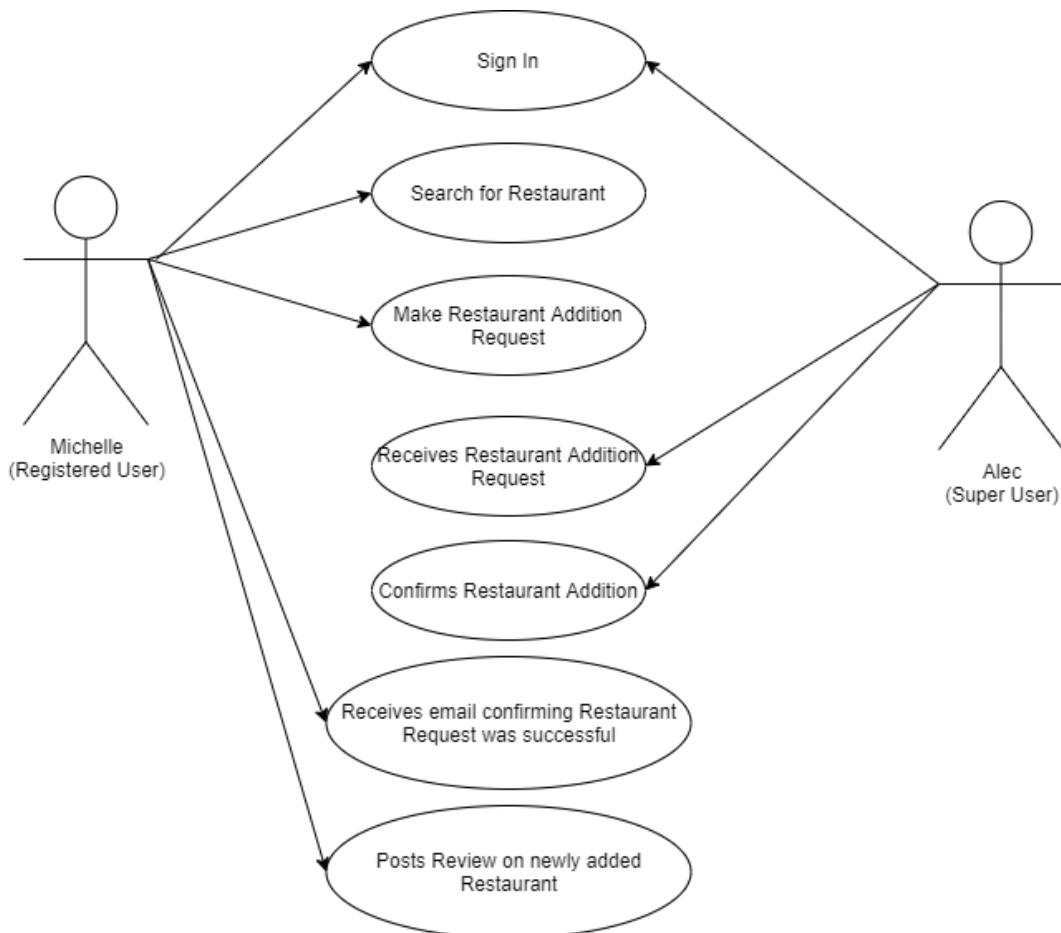


Case 6: Student at SFSU wants to write a review about a restaurant she liked

Actors: Michelle (Registered User), Alec (Super User)

A new restaurant opened up at Stonestown, and Michelle, a SFSU student and registered user, tried it out. Michelle absolutely loved the food and wanted to write a raving review about it on Gator Connection. However, when she went online and searched for the restaurant on Gator Connection, she could not find it. So, she made a request to Gator Connection to add the new restaurant to the restaurant section so that more people can find it and so she can post a review. She inputted the name of the restaurant and the location. Alec, a super user, received an email from Gator Connection that a restaurant addition request was made. He verified the information that Michelle provided and approved the addition of the restaurant. Michelle received an email informing her that her request had been approved. Michelle immediately went online and posted her raving review of the restaurant along with a picture she had taken of the food.

Case 6 Use Case Diagram:

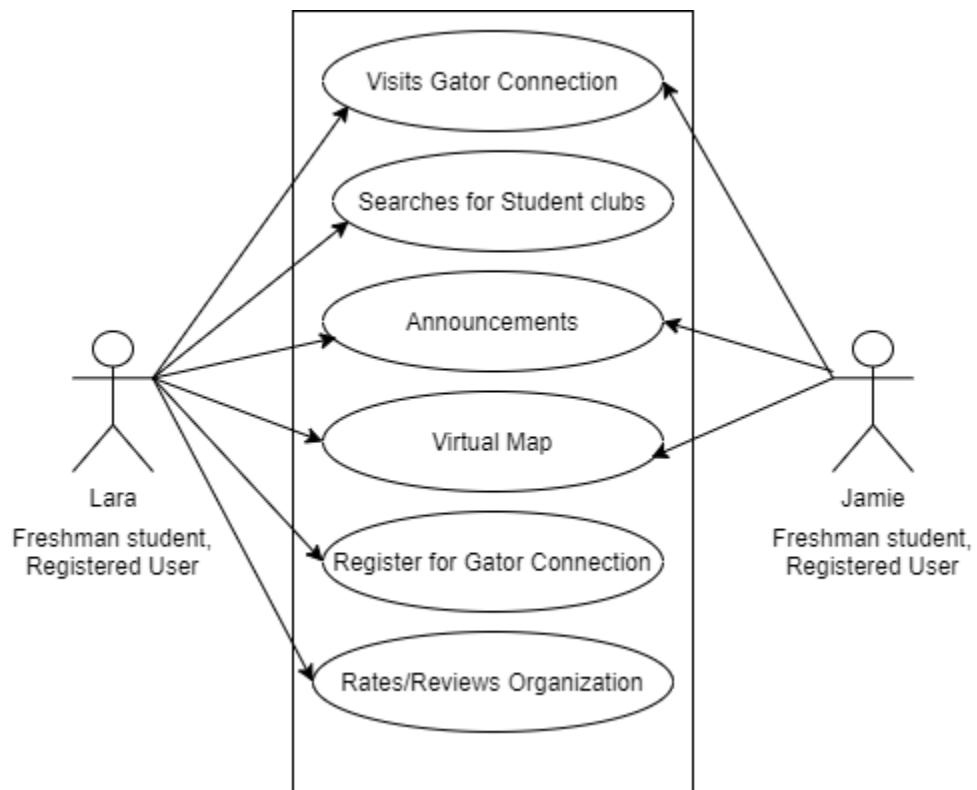


Case 7: Two Freshman at SFSU want to join a club but want more information**Actors: Lara(Registered User), Jamie(Registered User)**

Lara is a freshmen student at SFSU who is not sure about what clubs to join to network with people around her. While coming to campus, she heard people in her class mention about a new application called Gator Connection that helps students connect with one another. She decides to go to Gator connection to check if there might be any club announcements that may interest her. She finds one, and decides to attend their upcoming event. Lara's high school friend Jamie got into SFSU as well. Lara introduces the new event happening to Jamie so they could both make new friends together. Lara gave Jamie all the details verbally to Jamie, such as that the event would start at 11 am. Jamie eventually forgot due to school commitments, but she remembered about Gator Connection, where Lara said she got the information from. Jamie navigated to the application, specifically to the announcements section and found when the event would start. Jamie then noted down the details of the event so she would not forget again. Jamie and Lara then set out to the event.

However, the two realize they don't know where the location of the event is. With that, they realize that they saw a virtual map function on Gator Connection the last time that they were there. After visiting the virtual map section of Gator Connection, they then set out to where the event is. They both successfully attended their first event on campus. After attending the event, Lara randomly decided to checkout gator connection for other future events lined up. Surprisingly, when she went to the application again, event images and information about the next event was already underway. Lara then tries to make a post on the event announcement page, saying how they both had such a great time at the event. However, Gator Connection stops this, as only members of Gator Connection can post on the announcements section. Realizing she still has not signed up for Gator Connection, she then created an account. After this, Gator Connection then allows her to post her review of the event. This made her think about how efficient the club members would be. So, Lara went over to the "Organizations" section of Gator Connection, found the club and rated them the highest for others to know!

Case 7 Use Case Diagram:

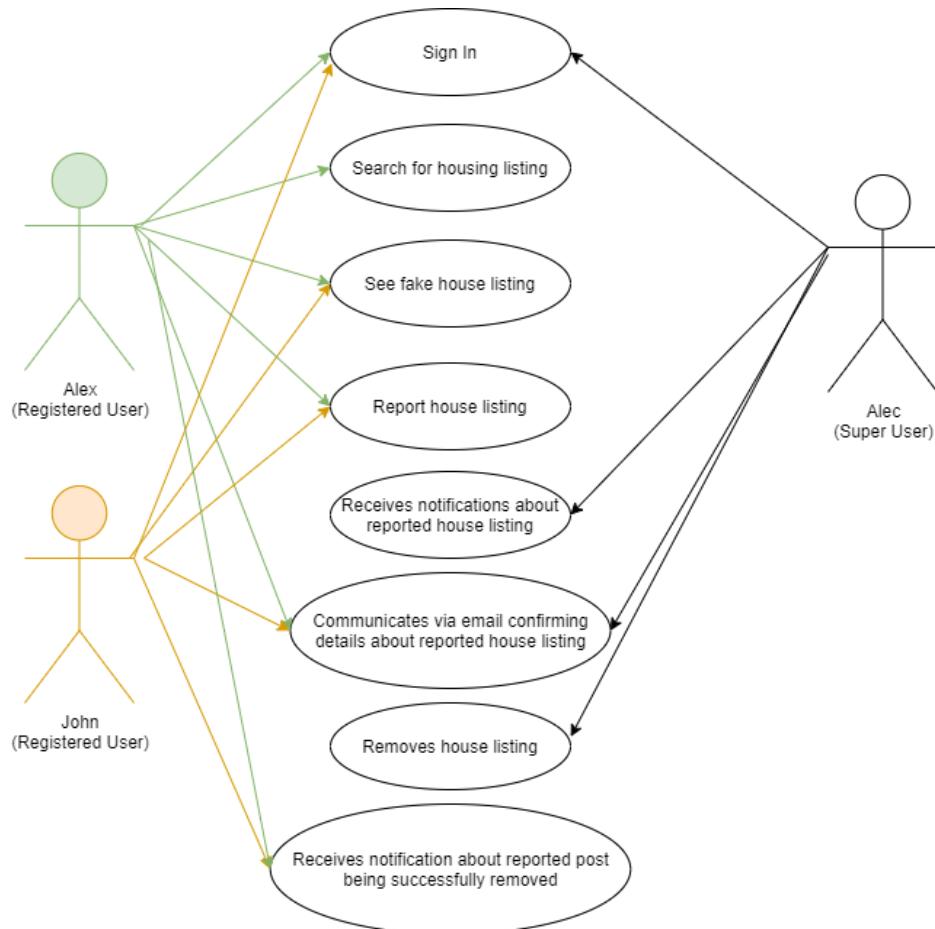


Case 8: A registered user reports a false housing listing

Actors: John (Registered User), Alex (Registered User), Alec (Super User)

John lives in an apartment near San Francisco State University. His friend, Alex, who also goes to SFSU is looking for a place to live for the new semester. Alex is a registered user, and he goes online to Gator Connection to look for housing listings. However, while Alex was scrolling through the listings, he noticed that John's apartment was listed as available. Alex texted John asking John if John was looking for a roommate. John said no, so Alex warned John about this fake house listing. Both Alex and John report the housing listing on Gator Connection indicating that the listing is fake with the reason that the owner of the apartment did not post the listing. Alec, a super user, received two report notifications about the housing listing that Alex and John reported. Alec emailed Alex and John about the report, and John informed Alec that John was the owner of the apartment and was not the one who posted the housing listing. So, Alec removed the housing listing from Gator Connection. Alex and John both received an email that a housing listing post that they had reported had successfully been dealt with and removed.

Case 8 Use Case Diagram:



Main Data Items & Entities

1. Unregistered User/ Guest

These are the people who aren't registered with the school system yet. They will be treated as guests, and would not be eligible for any of our student/staff services. This is an administrative task.

2. Registered User

A person who is registered with us using their email would have an account with us. He/She will follow the Create Account path. His/Her data is already registered with us, he/she will have the authority to create their password here. He/she shall have the ability to search housing listings, post restaurant reviews, and make purchase requests for posted items and housing listings.

3. Admin User

A person who is an administrative user has the power to approve and deny pending posts, whether they be from announcements or items for sale.

4. Super User

A person who is a super user has the power to approve restaurant additions, administrative account creation requests, and registered user account upgrade requests.

5. Notifications

Receiving notifications about housing, messages, announcements, submissions, etc.

6. Rate

This gives users the ability to rate various student organizations, restaurants, professors or even departments we have on campus. It'll prove to be a useful tool for fellow users.

7. Review

A written message often goes along with a rating. Users have an option to give these to a restaurant to describe any concerns they have with it.

8. Email

Being able to send emails to all of the users that are registered. This would be students and administration.

9. Announcements

Section designated to announcements that different departments make, along with the health centers, athletics, and organizations/clubs. These announcements are directly related to SFSU campus and only administrative users can post announcements.

10. Restaurants

Section dedicated to restaurants around campus. Users can view, rate, and write reviews for that specific restaurant. Unregistered users can view these but they cannot rate or review.

11. Restaurant Reviews

Posts shown under restaurants made by registered users and above. Unregistered users can view these but they cannot rate or review.

12. Ecommerce/Listing

The main section where users can view the selected items for sale or housing available and make purchase requests for them. Registered users can post things for sale. Unregistered guests cannot view these items.

13. Items

The item(s) listed for sale in the Ecommerce/Listing by a registered seller or for purchase by a registered buyer. Users can message the seller if they are interested in the item.

List of Functional Requirements

- **Guest User**

1. A guest user shall be able to search for restaurants/food.
2. A guest user shall be able to sort restaurant listings by rating.
3. A guest user shall be able to search/navigate through announcements.
4. A guest user shall be able to create an account using his/her unique SFSU email account.
5. A guest user shall be able to navigate to a map of SFSU.
6. A guest user shall be able to search for items on sale by category, price, etc.

- **Registered User**

7. A registered user shall have all of the same permissions as guest users.
8. A registered user shall be able to log in to his/her account using his/her SFSU email.
9. A registered user shall be able to log in to his/her account using many devices.
10. A registered user shall be able to log out of the website.
11. A registered user shall be able to stay logged in if they have not logged out.
12. A registered user shall be able to post reviews of restaurants/food.
13. A registered user shall be able to rate organizations.
14. A registered user shall be able to change their rating of the organization.
15. A registered user shall be able to post reviews under event announcements.
16. A registered user shall be able change any reviews they have under event announcements.
17. A registered user shall be able to make purchase requests for items.
18. A registered user shall be able to search for items by category, price, etc.
19. A registered user shall be able to sort and search for restaurants by ratings/reviews.
20. A registered user shall be able to see and search housing listings.
21. A registered user shall be able to make a request for a housing listing.
22. A registered user shall be able to rate a housing listing.
23. A registered user shall be able to post many pending housing listings with pictures.
24. A registered user shall receive a notification if their housing post has been approved.
25. A registered user shall receive a notification if their housing post has been denied.
26. A registered user shall be able to edit a housing listing he/she posted.
27. A registered user shall be able to close a housing listing that he/she posted.
28. A registered user shall be able to post many pending items for sale with pictures.
29. A registered user shall receive a notification if their post has been approved.
30. A registered user shall receive a notification if their post has been denied.

31. A registered user shall be able to rate any of the item listings on sale.
32. A registered user shall be able to change their rating of the item.
33. A registered user shall be able to edit a post about an item for sale that he/she posted.
34. A registered user shall be able to take down an item that he/she put up for sale.
35. A registered user shall be able to report housing listings and provide reasoning for the report.
36. A registered user shall be able to request to upgrade his/her account to an administrative account with a valid organization, role, and organization email.
37. A registered user shall be able to create a restaurant request to add a restaurant to the restaurants listing by providing the name of the restaurant and location.
38. A registered user shall be notified if their restaurant location has been approved or denied, with a message explaining why
39. A registered user shall receive a notification about purchase requests made on his/her account.
40. A registered user shall receive a notification about purchase requests made on items he/she posted for sale.
41. A registered user shall receive a notification about a house listing request for a house listing that he/she posted that includes information about the person who requested, such as his/her name and email address.
42. A registered user shall receive a notification when a post he/she reported is successfully removed or unsuccessfully denied.
43. A registered user shall receive a notification about a house listing being closed if he/she had made a request to the house lister.
44. A registered user who requested an account upgrade shall receive a notification when the request has been approved and the date of when their administration privileges will be taken away.
45. A registered user who requested an account upgrade shall receive a notification when the request has been denied.
46. An upgraded registered user shall receive a notification when their administration privileges are taken away and why.
47. A registered user shall have a unique registered id.
- **Administrative User**
 48. An administrative user shall have all of the same permissions as registered users.
 49. An administrative user shall be able to post announcements.
 50. An administrative user shall be able to remove an announcement he/she had posted.
 51. An administrative user shall have a unique admin id.

- **Super User**
 - 52. A super user shall be able to log into the website.
 - 53. A super user shall be able to log out of the website.
 - 54. A super user shall have all of the same permissions as administrative users.
 - 55. A super user shall be able to approve/deny and close account upgrade requests,
 - 56. A super user shall be able to approve/deny and close administrative account creation requests.
 - 57. A super user shall be able to approve/deny and close restaurant addition requests.
 - 58. A super user shall receive a notification about account upgrade requests, administrative account creation requests, restaurant requests, and reported posts.
 - 59. A super user shall be able to remove administrative privileges from an upgraded registered user and must give a reason.
 - 60. A super user shall be able to remove posts from item sales and housing listings.
 - 61. A super user shall be able to approve pending posts for restaurant reviews.
 - 62. A super user shall be able to reject pending posts for restaurant reviews.
 - 63. A super user shall be able to approve pending posts for items for sale.
 - 64. A super user shall be able to reject pending posts for items for sale.
 - 65. A super user shall be able to approve pending housing listings for sale.
 - 66. A super user shall be able to reject pending housing listings for sale.
- **Sale Item**
 - 67. A sale item shall be posted by a registered user.
 - 68. A sale item shall have a unique item id.
 - 69. A sale item shall have a title for the item.
 - 70. A sale item shall have a message describing the item.
 - 71. A sale item has the option to have a picture(s) attached to it.
 - 72. A sale item shall be purchased by a registered user.
 - 73. A sale item shall be taken down by the registered user that posted it for sale.
 - 74. A sale item shall be able to be approved by a super user.
 - 75. A sale item shall be able to be rejected by a super user.
 - 76. When a sale item is taken down, all registered users who made a request for the item shall be notified of the closure.
- **Housing Listing**
 - 77. A housing listing shall be posted by a registered user.
 - 78. A housing listing shall have a unique housing id.
 - 79. A housing listing shall have a title.
 - 80. A housing listing shall have a message describing the item.
 - 81. A housing listing shall have picture(s) attached to it.
 - 82. A housing listing shall be taken down by the registered user that posted it.
 - 83. A housing listing shall be requested by many registered users.
 - 84. A housing listing shall be able to be approved by a super user.

85. A housing listing shall be able to be rejected by a super user.
 86. When a housing listing is taken down, all registered users who made a request shall be notified of the closure.
- **Restaurant**
 87. A restaurant shall have a rating.
 88. A restaurant shall be able to have their rating changed.
 89. A restaurant shall have many reviews written by registered users.
 90. A restaurant shall have a name.
 91. A restaurant shall have a location.
 92. A restaurant shall have a unique restaurant id.
 - **Restaurant Request**
 93. A restaurant request shall be created by one and only one registered user and above.
 94. A restaurant request shall have the name of the restaurant.
 95. A restaurant request shall have the location of the restaurant.
 96. A restaurant request shall be confirmed/denied and closed by one and only one super user.
 97. When a restaurant request is confirmed/denied the registered user who made the request will be notified.
 - **Organization**
 98. An organization shall be able to be rated by many registered users.
 99. An organization shall be able to have their ratings changed.
 100. An organization shall be able to post announcements.
 - **Announcement**
 101. An announcement shall be posted by one and only one administrative user or organization.
 102. An announcement shall be able to be viewed by one or many users.
 103. An announcement shall be able to be reviewed.
 104. An announcement shall be able to be taken down by the one who posted it.
 105. An announcement shall be able to be rated.

List of Non-Functional Requirements

- **Performance:**
 1. The website should have a load time of no more than 4 seconds.
 2. Notifications to users shall be sent within 30 seconds of purchase requests to allow users to connect more quickly.
- **Marketing:**
 3. All pages shall have the company logo in the upper left hand corner.
- **Look and Feel:**
 4. A navbar shall be at the top of all website pages.
 5. Restaurant reviews shall have the user that posted, optional images, rating, and comments.
 6. Posted items shall have the user that posted, one required image and more optional images, and details about the item.
 7. Housing listings shall have the user that posted, many required images, and details about the housing.
- **Scalability:**
 8. The website shall be scalable and adapt optimally when the number of users and workloads increase.
 9. The website shall be scalable and maintainable as more features are added.
- **Recoverability:**
 10. In times where the website faces problems or failure, components shall be easy to navigate and fixed by the administrator to recover the website.
- **Functionality:**
 11. The website shall be developed and deployed using the tech stack approved by the class CTO.
 12. The website shall have UX/UI that is clear and helpful for users so they can navigate to other pages and find resources easily.
- **Security:**
 13. A registered user's with a Student Account shall have his/her email verified by the system to be an SFSU email account and unique in order to sign up.
 14. A verification email shall be sent to a registered user's email.
 15. A registered user with an Admin Account or Super User Account shall be approved by a super user before being allowed to use the features specific to each account type.
 16. A restaurant from a restaurant request shall be verified by a super user in order to be added to the restaurant catalog.
 17. A registered user shall be notified when a new device has logged into their account.
 18. Only administrative users shall be allowed to post announcements.

19. Only registered users shall be allowed to post items for sale.
 20. Only registered users shall be allowed to post housing listings.
 21. Only registered users shall be allowed to make a purchase request for items that are on sale.
 22. Only registered users shall be allowed to make a purchase request for housing listings.
 23. Only registered users and above shall be allowed to post restaurant reviews.
- **Privacy:**
 24. A registered user's password shall be encrypted and saved in the database.
 25. A registered user's name shall be saved in the database.
 26. A registered user's email shall be saved in the database.
 27. A registered user's name shall be passed around with purchase requests and restaurant reviews.
 28. A registered user's email shall be passed around with only purchase requests.
 - **System Requirements:**
 29. The website shall work up to Version 88.0.4324.182 of Google Chrome.
 30. The website shall work up to Version 86 of Mozilla Firefox.
 31. The website shall work up to Version 14.0.3 of Safari.
 32. The website shall be compatible for both Windows 10 and MacOS.
 - **Availability:**
 33. The website shall be fully functional at any time of the day whenever there is no maintenance.
 - **Capability/Expected Load:**
 34. The website shall support as many users as the AWS EC2 Instance can support.
 - **Legal:**
 35. A link to the terms and conditions shall be present at the bottom of all website pages.
 36. A link to the privacy notice shall be present at the bottom of all website pages.
 - **Coding Requirements:**
 37. All code shall be well documented.
 38. All code shall have relevant variable and function names.
 39. All code shall be well organized and have the same code style.
 40. All code shall be indented using spaces.
 - **Environmental:**
 41. Local environments on developer machines shall mirror the AWS EC2 Instance environment.
 42. All development shall be done on the development branch or lower.
 43. The development branch shall be approved by Team Lead before being merged into the master branch.
 44. All GitHub commits shall have extensive details about code added and changed.

45. All code shall be extensively tested locally before being deployed on the AWS server.

- **Storage:**

46. Data inputted by the user during sign up shall be stored in the database.

47. General users shall be stored in the database.

48. General users that leave the website shall be removed from the database.

49. All announcements shall be stored in the database.

50. All posts shall be stored in the database.

- **Fault Tolerance:**

51. The website shall be able to refresh whenever there is a lost connection.

Competitive Analysis

Websites with similar services	sfsu.edu	hunter.cuny.edu/students	SFSU Apartment roommate group on fb	sfstategators.com	craigslist.com
Strengths	<ul style="list-style-type: none"> -Officially supported by SFSU -First hand information directly from SFSU - Connects to multiple services and SFSU departments 	<ul style="list-style-type: none"> -UI is simple, organized and easy to navigate -Officially supported by Hunter College - Connects to multiple services 	<ul style="list-style-type: none"> - Specifically made for SFSU students, so can guarantee all listings are from SFSU - Very easy to use for someone who's has fb 	<ul style="list-style-type: none"> -provides first hand information from SFSU about sports and events -Covers all the athletics department that is available in the SFSU. 	<ul style="list-style-type: none"> -wide range of products to search for - you can search based on location -
Weaknesses	Very hard to navigate around UI, UI very cluttered	-Info not For SFSU students.	<ul style="list-style-type: none"> - Have to have a Facebook account - Have to be invited by someone who is already there - Possibility that posts that may be there is a scam 	<ul style="list-style-type: none"> -Unorganized Nav-bar, cluttered -Social media icon on desktop version doesn't redirect the user to the corresponding site. 	<ul style="list-style-type: none"> Really bad UI -Possible Scams
Pricing	Free	Free	Free to use	Free	-Free
Social Media	<ul style="list-style-type: none"> - Twitter - Facebook 	<ul style="list-style-type: none"> - Twitter - Facebook 	<ul style="list-style-type: none"> - Facebook 	<ul style="list-style-type: none"> -Twitter -Instagram 	None worth mentioning

	<ul style="list-style-type: none"> - Instagram - LinkedIn - Flickr - YouTube 	<ul style="list-style-type: none"> - Instagram - Flickr 		<ul style="list-style-type: none"> -Facebook 	
Onboarding	<ul style="list-style-type: none"> - Not required to create an account for general usage 	<ul style="list-style-type: none"> - Not required to create an account for general usage 	<ul style="list-style-type: none"> - Have to have a fb account 	<ul style="list-style-type: none"> -Account not required to use the services. 	<ul style="list-style-type: none"> -Account required to create a post

Features	sfsu.edu	hunter.cuny.edu/students	https://www.facebook.com/groups/223625011079774	sfstategators.com	craigslist.com	Gator Connection
UI Simplicity	-	+	-	+	-	++
Market	-	-	+	-	++	+
Search	+	+	++	+	+	+
Post and Request for	-	-	-	-	+	++

Housing Listings						
Restaurant Ratings/Reviews	-	-	-	-	-	++

+: Feature exists

++: Feature is superior

-: Feature does not exist

Summary of Competitive Analysis

Our main superior feature compared to what other listing services have is our housing listings service. With our housing listings service, the one who posted the listing will have the final say in who he/she chooses to contact. We will do this by implementing a feature where buyers will not have access to any contact information about the listing. Instead they will only have access to the general location of the room, pictures of the room, and any other information about the room posted. Buyers can inquire that they are interested in the housing through a form that will contain information that they write about themselves; however, it will be up to the one who posted the listing to reply to the buyer that he/she is interested in selling them the room. By doing this, sellers of the room can rest easy knowing that they have the choice on who to contact. On top of this, our feature gives sellers the power to prevent their contact information from being shared, unlike other competitors such as Craigslist.

High Level System Architecture and Technologies Used

Server Host:

Amazon AWS 1vCpus 1 gib ram not ebs optimized

Operating System:

Ubuntu 20.04

DataBase:

MySQL 8.0

Web Server:

NGINX 1.18

Server side Language(Back end):

Python 3.8

Additional Technologies:

a) Web Framework: Django 3.1

b) IDE: VS code, Pycharm

c) Gunicorn: 20.0.4

c) Supervisor

e) Bootstrap 4

Team Contributions

1) Executive Summary:

Alec wrote the first rough draft of the executive summary. During a team meeting, Angelo and Benjamin helped provide feedback and revise parts of it to ensure the message of centralizing all SFSU info was clear.

2) Use Cases:

After discussing Use Cases, everyone on the team was tasked to complete their own Use Cases. After receiving feedback from the CTO, we went over the Use Cases and how they could be improved. After Benjamin revised and provided feedback on each of our Use Cases to extract Data Items, Functional Requirements and Non-Functional Requirements, all of us then proceeded to create Use Case Diagrams for them.

3) List of Main Data Items and Entities:

Lakshita and Carmen were tasked to complete this part. After looking through this point during the main team meeting, the team, most notably Benjamin, helped provide feedback on the items that we should and should not keep.

4) Initial list of Functional requirements:

Jiaxin was tasked to complete this part. During the time that Jiaxin was working on this, Benjamin also helped and added his own points for Functional requirements. Furthermore, during the main team meeting, the team looked over the list to make sure everything was accounted for.

5) List of Non-Functional requirements:

Angelo was tasked to complete this part. During the time that he was working on this, the team, most notably Benjamin, helped Angelo by providing feedback to the points that he had listed. Furthermore, during the main team meeting, the team looked over to make sure everything was accounted for.

6) Competitive Analysis:

Benjamin and Bikram were tasked to complete this part. During the main team meeting, the team looked over the competitors that they had chosen to see if they each represent something our product could compare to. Furthermore, the team also decided on a new competitor to add to account for one of the services that we have on our App.

7) Tech Stack:

Alec reviewed the tech stack that he had used and sent to the Professor and checked if all of them were still in use. During the main team meeting, the team agreed that the tech stack that we had would be the one we would be using.

8) Contributions:

Alec filled out this part. After assigning the tasks to everyone, Alec looked over what everyone did to accurately fill out this part.

9) Checklist:

Alec filled out this part. During the main team meeting, Alec made sure to confirm that all the points in the checklist would have an answer to them, whether they be DONE, ON TRACK, and ISSUE

10) Formatting the project:

Bikram was tasked with compiling all the points that we have into one document. After he was finished with the document, the team, most notably Alec and Benjamin helped provide feedback and revise certain parts of the final document.

Team Members

- 1) Alec Tenefrancia (Team Lead, Github Master)
- 2) Angelo Reyes (Back End Lead, Github Admin)
- 3) Lakshita Chugh (Front End Lead, Github Admin)
- 4) Bikram Tamang (Front End team member, M1 Editor)
- 5) Carmen Paisonao(Front End team member)
- 6) Benjamin Kao(Back End team member)
- 7) Jiaxin Yu (Back End team member)

Team CheckList

- Team found a time slot to meet outside of the class
(DONE)
- Github master chosen
(DONE)
- Team decided and agreed together on using the listed SW tools and deployment server
(DONE)
- Team ready and able to use the chosen back and front end frameworks and those who need to learn are working on learning and practicing
(ON TRACK)
- Team lead ensured that all team members read the final M1 and agree/ understand it before submission
(DONE)
- Github organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.)
(DONE)

Milestone 2 V2

Software Engineering CSC 648/848 Spring 2021

Gator Connection

A One Stop Website for SF State Gators

Team 05

Team Lead/Github Master: Alec Stephen Tenefrancia alectene@mail.sfsu.edu

Frontend Lead/Document Master: Lakshita Chugh

Backend Lead: Angelo Gloria Reyes

Frontend member: Bikram Tamang

Backend member: Benjamin Patrick Kao

Frontend member: Carmen Denisse Paisano

Backend member: Jiaxin Yu

Milestone 2

Milestone/Version	Date
M2V2	04/24/21
M2V1	04/01/21
M1V2	03/09/21
M1V1	03/04/21

Table of Contents

Table of Contents	44
Data Definitions V2	47
Functional Requirements V2	50
Priority 1 Requirements:	50
Priority 2 Requirements:	53
Priority 3 Requirements:	54
UI Mockups and Storyboards	56
High Level Database Architecture and Organization	83
Business Rules	83
Entity, Attribute, Relationship, Domain Descriptions	85
Entity Relationship Diagram (ERD)	87
Database Model/Enhanced Entity Relationship (EER)	89
Database Management System (DBMS)	90
Media Storage	90
Search/Filter Architecture and Implementation	90
High Level APIs and Main Algorithms	91
High Level UML Diagrams	93
High Level Application Network and Deployment Diagrams	95
Project Risks	98
Project Management	100
List of Contributions	102

Data Definitions V2

1. Unregistered User/ Guest

These are the people who aren't registered with the school system yet. They will be treated as guests, and would not be eligible for any of our student/staff services. This is an administrative task.

2. Registered User

A person who is registered with us using their email would have an account with us. He/She will follow the Create Account path. His/Her data is already registered with us, he/she will have the authority to create their password here.

3. Account

An account contains identification data about the registered user that created the account. The data identifies what type of registered user the user is, as well as helps authentication registered users for restricted features such as posting announcements.

- a) **Student:** A student account gives registered users the ability to search housing listings, post restaurant reviews, and make purchase requests for posted items and housing listings.
- b) **Admin:** An admin account gives registered users the power to post public announcements. Admin accounts are split into more specific roles as to more easily group announcements together when filtering/searching.
 - i) **Sports/Athletics:** A registered user who has this type of account is in charge of a specific sport that he/she registered with.
 - ii) **Organization/Club:** A registered user who has this type of account is in charge of a specific organization that he/she registered with.
 - iii) **School Department:** A registered user who has this type of account is a part of SFSU staff.
- c) **Super User:** A super user account gives registered users the power to approve restaurant additions, administrative account creation requests, and registered user account upgrade requests.

4. Notifications

Receiving notifications about housing, submissions, etc.

- a) **Housing Notifications:** Registered users will receive email notifications about housing such as if they receive a request for a listing.
- b) **Submission Notifications:** Registered users will receive email notifications about submissions such as if their submission has been approved or denied

5. Rate

This gives users the ability to rate various student organizations, restaurants. It'll prove to be a useful tool for fellow users.

- a) **Student Organization Rating:** Ratings that users can give organizations from 1-10. The average of all ratings will be shown on the organization.
- b) **Restaurants Rating:** Ratings that users can give restaurants from 1-10. The average of all ratings will be shown on the restaurants.

6. Review

Posts shown under restaurants made by registered users and above. Unregistered users can view these but they cannot rate or post a review.

7. Email

Being able to send emails to all of the users that are registered. This would be students and administration.

- a) **Housing emails:** A User who posted a listing of a house for sale will receive emails from people who are interested to purchase the listing.
- b) **Registration Verification email:** A User who creates an account will receive an email to their SFSU email saying that their account has been created
- c) **Reset password email:** An email which a user can request to reset their password

8. Announcements

Section designated to announcements that different departments make, along with the health centers, athletics, and organizations/clubs. These announcements are directly related to SFSU campus and only administrative users can post announcements.

- a) **Athletic Announcements:** Announcements made from athletic directors
- b) **Organization Announcements:** Announcements made from organizations

- c) **School Announcements:** Announcements made from the school

9. Restaurants

Section dedicated to restaurants around campus. Users can view, rate, and write reviews for that specific restaurant. Unregistered users can view these but they cannot rate or review.

10. Ecommerce/Listing

The main section where users can view the selected items for sale or housing available and make purchase requests for them. Registered users can post things for sale. Unregistered guests cannot view these items.

a) Items:

The item(s) listed for sale in the Ecommerce/Listing by a registered seller or for purchase by a registered buyer. Users can message the seller if they are interested in the item.

b) Housing:

The listing(s) of housing listed for sale in the Housing section. Users interested in a listing can fill out a form which includes their name, school year, SFSU email, and a little information about themselves. After filling out the form, the system will send out an email to the person who posted the listing, and it will be up to them if they wish to respond back.

Functional Requirements V2

Priority 1 Requirements:

1. Guest User
 - 1.1. A guest user shall be able to search for restaurants by title.
 - 1.2. A guest user shall be able to sort restaurant listings by rating.
 - 1.3. A guest user shall be able to navigate through announcements.
 - 1.4. A guest user shall be able to create an account using his/her unique SFSU email account for a student account or a unique email account for an admin or super user account.
 - 1.5. A guest user shall be able to navigate to a map of SFSU.
 - 1.6. A guest user shall be able to search for items on sale by preferred payment.
 - 1.7. A guest user shall be able to search for items on sale by price.
 - 1.8. A guest user shall be able to search for items on sale by title.
2. Registered User
 - 2.1. A registered user shall have all of the same permissions as guest users.
 - 2.2. A registered user shall be able to log in to his/her account using his/her unique SFSU email.
 - 2.3. A registered user shall be able to log in to his/her account using many devices.
 - 2.4. A registered user shall be able to log out of the website.
 - 2.5. A registered user shall be able to stay logged in if they have not logged out.
 - 2.6. A registered user shall be able to post reviews of restaurants/food.
 - 2.7. A registered user shall be able to make purchase requests for items.
 - 2.8. A registered user shall be able to search for items by preferred payment, price, etc.
 - 2.9. A registered user shall be able to see and search housing listings.
 - 2.10. A registered user shall be able to make a request for a housing listing.
 - 2.11. A registered user shall be able to edit a housing listing title he/she posted.
 - 2.12. A registered user shall be able to edit a housing listing description he/she posted.
 - 2.13. A registered user shall be able to edit a housing listing price he/she posted.
 - 2.14. A registered user shall be able to change a housing listing image he/she posted.
 - 2.15. A registered user shall be able to close a housing listing that he/she posted.
 - 2.16. A registered user shall be able to post many pending items for sale with pictures.
 - 2.17. A registered user shall be able to edit a post about an item for sale that he/she posted.
 - 2.18. A registered user shall be able to take down an item that he/she put up for sale.

- 2.19. A registered user shall be able to create a restaurant request to add a restaurant to the restaurants listing by providing the name of the restaurant and location.
 - 2.20. A registered user shall be able to sort and search for restaurants by ratings/reviews.
 - 2.21. A registered user shall be notified if their restaurant location has been approved or denied, with a message explaining why.
 - 2.22. A registered user shall receive a notification about a house listing being closed if he/she had made a request to the house lister.
 - 2.23. An upgraded registered user shall receive a notification when their administration privileges are taken away and why.
 - 2.24. A registered user shall have a unique registered id.
 - 2.25. A registered user shall be able to fill out a form for a housing request.
 - 2.26. A registered user shall receive a notification if their housing post has been approved by email.
 - 2.27. A registered user shall receive a notification if their housing post has been denied by email.
 - 2.28. A registered user shall receive a notification if their post has been approved by email.
 - 2.29. A registered user shall receive a notification if their post has been denied by email.
 - 2.30. A registered user shall be able to request to upgrade his/her account to an administrative account with a valid organization, role, and organization by email.
 - 2.31. A registered user shall receive a notification about purchase requests made on his/her account by email.
 - 2.32. A registered user shall receive a notification about purchase requests made on items he/she posted for sale by email.
 - 2.33. A registered user shall receive a notification about a house listing request for a house listing that he/she posted that includes information about the person who requested, such as his/her name and email address.
 - 2.34. A registered user who requested an account upgrade shall receive a notification when the request has been approved and the date of when their administration privileges will be taken away by email.
 - 2.35. A registered user who requested an account upgrade shall receive a notification when the request has been denied by email.
3. Admin User
 - 3.1. An administrative user shall have all of the same permissions as registered users.
 - 3.2. An administrative user shall be able to post announcements.
 - 3.3. An administrative user shall be able to remove an announcement he/she had posted.
 - 3.4. An administrative user shall have a unique admin id.

4. Super User
 - 4.1. A super user shall be able to log into the website.
 - 4.2. A super user shall be able to log out of the website.
 - 4.3. A super user shall have all of the same permissions as administrative users.
 - 4.4. A super user shall be able to approve/deny and close account upgrade requests,
 - 4.5. A super user shall be able to approve/deny and close administrative account creation requests.
5. Sale Item
 - 5.1. A sale item shall be posted by a registered user.
 - 5.2. A sale item shall have a unique item id.
 - 5.3. A sale item shall have a title for the item.
 - 5.4. A sale item shall have a message describing the item.
 - 5.5. A sale item shall have one picture attached to it.
 - 5.6. A sale item shall be purchased by a registered user.
 - 5.7. A sale item shall be taken down by the registered user that posted it for sale.
 - 5.8. A sale item shall have a price.
 - 5.9. When a sale item is taken down, all registered users who made a request for the item shall be notified of the closure by email.
6. Housing Listing
 - 6.1. A housing listing shall be posted by a registered user.
 - 6.2. A housing listing shall have a unique housing id.
 - 6.3. A housing listing shall have a title.
 - 6.4. A housing listing shall have a message describing the item.
 - 6.5. A housing listing shall have a price.
 - 6.6. A housing listing shall have at least 1 picture attached to it.
 - 6.7. A housing listing shall be taken down by the registered user that posted it.
 - 6.8. A housing listing shall be requested by many registered users.
 - 6.9. A housing listing shall have a form to fill out for interested users.
 - 6.10. A housing listing shall have a situation(available/close).
 - 6.11. When a housing listing is taken down, all registered users who made a request shall be notified of the closure by email.
7. Housing Listing Form
 - 7.1. A housing listing form shall be filled out by a registered user.
 - 7.2. A housing listing form shall fill out their full name.
 - 7.3. A housing listing form shall fill out their school year.
 - 7.4. A housing listing form shall fill out their school email.
 - 7.5. A housing listing form shall fill out their phone number.
 - 7.6. A housing listing form shall have a section dedicated to “an about” me.
 - 7.7. A housing listing form shall have an expected day they want to move in.

8. Restaurant
 - 8.1. A restaurant shall have a name.
 - 8.2. A restaurant shall have a location.
 - 8.3. A restaurant shall have a unique restaurant id.
 - 8.4. A restaurant shall have a rating.
9. Restaurant Request
 - 9.1. A restaurant request shall be created by one and only one registered user and above.
 - 9.2. A restaurant request shall have the name of the restaurant.
 - 9.3. A restaurant request shall have the location of the restaurant.
 - 9.4. A restaurant request shall be confirmed/denied and closed by one and only one super user.
 - 9.5. When a restaurant request is confirmed/denied the registered user who made the request will be notified by email.
10. Organization
 - 10.1. An organization shall be able to post announcements.
11. Announcement
 - 11.1. An announcement shall be posted by one and only one administrative user or organization.
 - 11.2. An announcement shall be able to be viewed by one or many users.
 - 11.3. An announcement shall be able to be taken down by the one who posted it.

Priority 2 Requirements:

12. Guest User
 - 12.1. A guest user shall be able to make item purchase requests after filling out an identification form and completing a Captcha.
13. Registered User
 - 13.1. A registered user shall be able to rate organizations.
 - 13.2. A registered user shall be able to change their rating of the organization.
 - 13.3. A registered user shall be able to post reviews under event announcements.
 - 13.4. A registered user shall be able to change any reviews they have under event announcements.
 - 13.5. A registered user shall be able to rate a housing listing.
 - 13.6. A registered user shall be able to rate any of the item listings on sale.
 - 13.7. A registered user shall be able to change their rating of the item.
 - 13.8. A registered user shall be able to report housing listings and provide reasoning for the report.
 - 13.9. A registered user shall be able to report restaurant reviews.
 - 13.10. A registered user shall be able to report announcements
14. Admin User

- 14.1. An admin user shall be able to temporarily take down posts that he/she finds false and give a reason why.
15. Super User
 - 15.1. A super user shall receive notifications of posts that admin users take down.
16. Sale Item
 - 16.1. A sale item shall be able to be taken down by a super user.
17. Housing Listing
 - 17.1. A housing listing shall be able to be taken down by a super user.
18. Housing Listing Form
 - 18.1. A housing listing form shall be able to be saved by the creator in their notifications page.
19. Restaurant
 - 19.1. A restaurant shall be able to have their rating changed.
 - 19.2. A restaurant shall have many reviews written by registered users.
20. Restaurant Request
 - 20.1. A restaurant request shall also allow users to request to update incorrect information about a restaurant.
21. Organization
 - 21.1. An organization shall be able to be rated by many registered users.
22. Announcement
 - 22.1. An announcement shall be able to be taken down by a super user.
 - 22.2. An announcement shall be able to be rated.

Priority 3 Requirements:

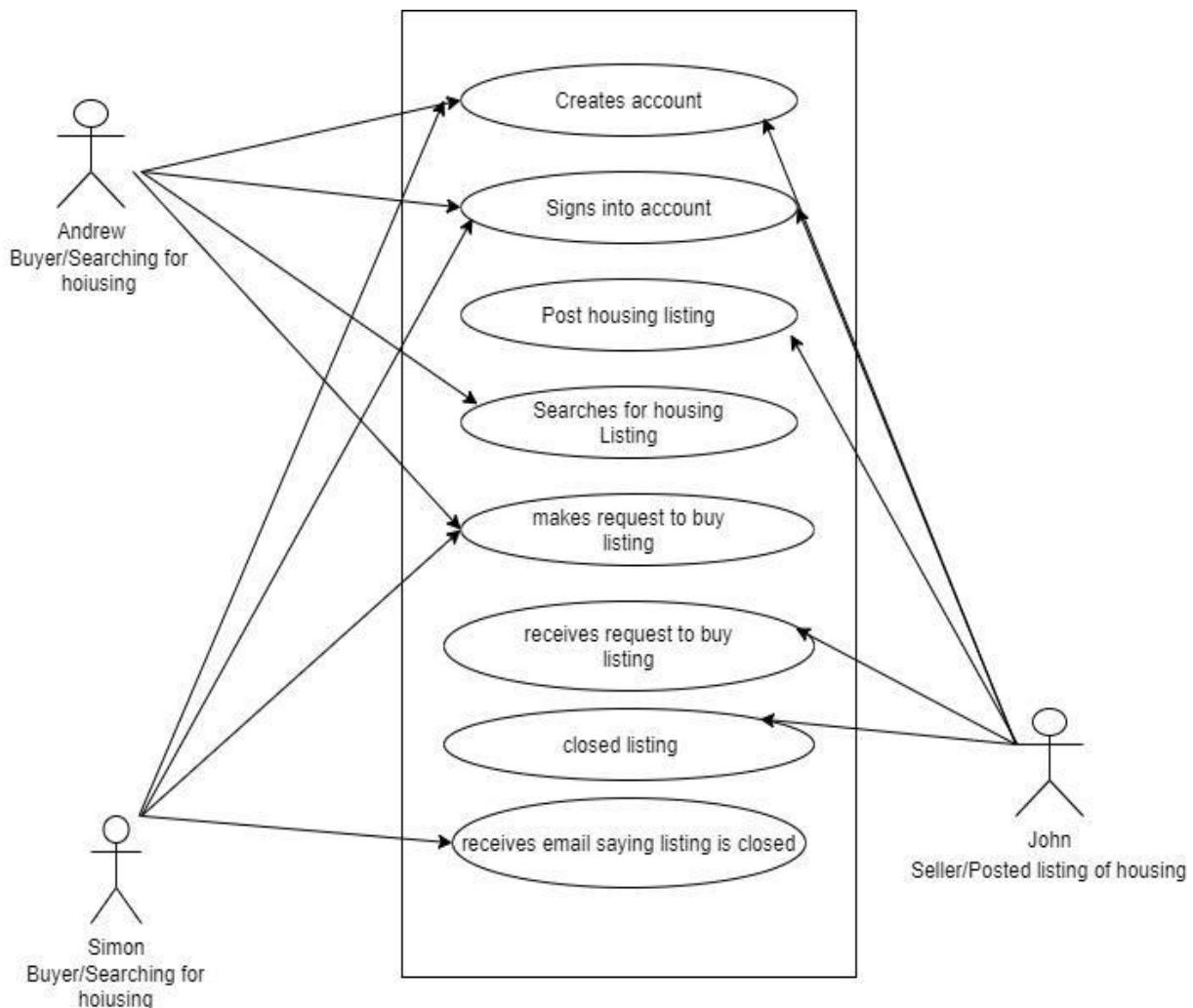
23. Guest User
 - 23.1. A guest user shall be able to make housing purchase requests after filling out an identification form and completing a Captcha.
24. Registered User
 - 24.1. A registered user shall be able to fill out a potential roommate form when posting a housing listing.
 - 24.2. A registered user shall be able to fill out a potential roommate form when searching through housing listing posts.
 - 24.3. A registered user shall be able to receive suggested housing listings where both the poster and searcher are ideal roommates.
 - 24.4. A registered user shall be able to subscribe to organizations and departments.
 - 24.5. A registered user shall be able to report item and housing listings for false/fake information.
 - 24.6. A registered user shall be able to be banned/restricted to only guest user privileges if he/she has a certain number of reports on posts.
25. Admin User

- 25.1. An admin user shall be able to be banned/restricted to only guest user privileges if he/she has a certain number of reports on announcements.
- 26. Super User
 - 26.1. A super user shall receive notifications of taken down posts such as announcements, housing listings, and items.
 - 26.2. A super user shall receive notifications of banned users.
 - 26.3. A super user shall be able to unban banned users.
- 27. Sale Item
 - 27.1. A sale item shall be automatically hidden from users when a certain number of reports is reached.
- 28. Housing Listing
 - 28.1. A housing listing shall be automatically hidden from users when a certain number of reports is reached.
 - 28.2. A housing listing shall be shown on a virtual map.
 - 28.3. A housing listing shall show the distance from SF State.
 - 28.4. A housing listing shall show how long it takes for a person to get to SF State.
- 29. Housing Listing Form
 - 29.1. A housing listing form shall be able to be customized for a housing listing post by registered users.
- 30. Restaurant
 - 30.1. A restaurant shall be shown on a virtual map.
 - 30.2. A restaurant shall show the distance from SF State.
- 31. Restaurant Request
 - 31.1. A restaurant request shall be able to be deleted by the one requested it.
- 32. Organization
 - 32.1. An organization shall be able to send emails to subscribed registered users about announcements.
- 33. Announcement
 - 33.1. An announcement shall be able to be mass sent to registered users.
 - 33.2. An announcement shall be automatically hidden from users when a certain number of reports is reached.

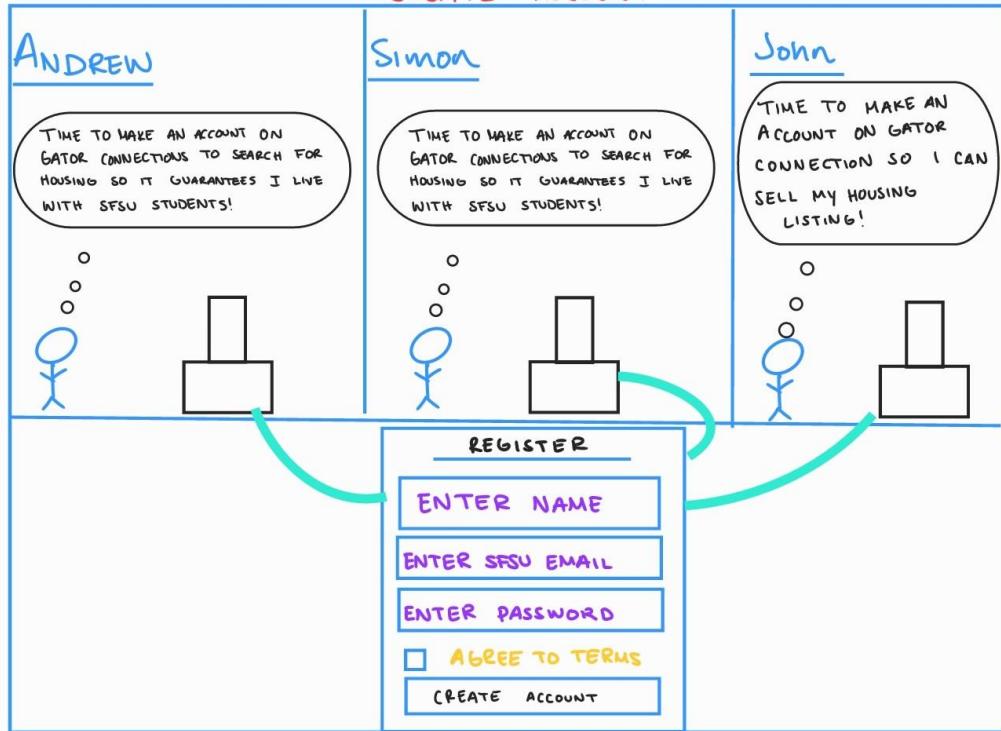
UI Mockups and Storyboards

Case 1: Freshman Student looking for housing at SFSU

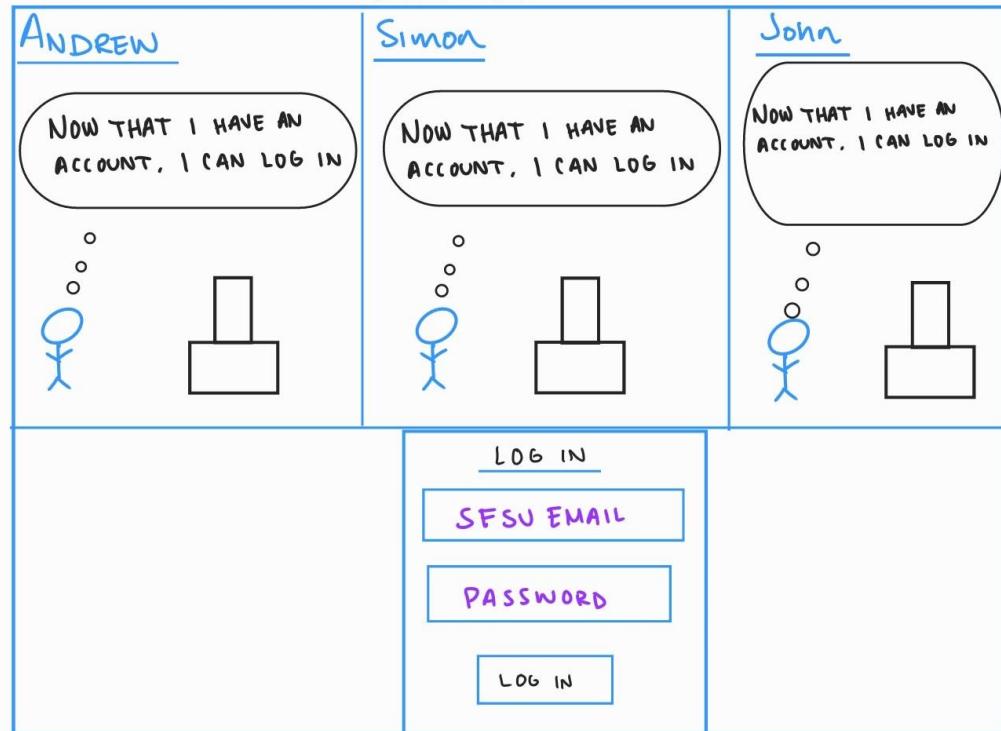
Actors: Andrew(Buyer, Searching for housing), Simon (Buyer, Searching for housing), John(Seller, Posted listing of housing)



CREATE ACCOUNT



LOG IN



POST HOUSE LISTING

John

TIME TO POST
MY HOUSING LISTING !

ENTER TITLE
ENTER CONDITION OF HOUSING
ENTER LOCATION
ENTER PRICING
ENTER CONTACT INFO
ENTER PREFERRED PAYMENT
UPLOAD PICTURE
POST

SEARCH FOR HOUSING LISTING

ANDREW

TIME TO SEARCH
FOR HOUSING

Simon

TIME TO SEARCH
FOR HOUSING

SEARCH FOR

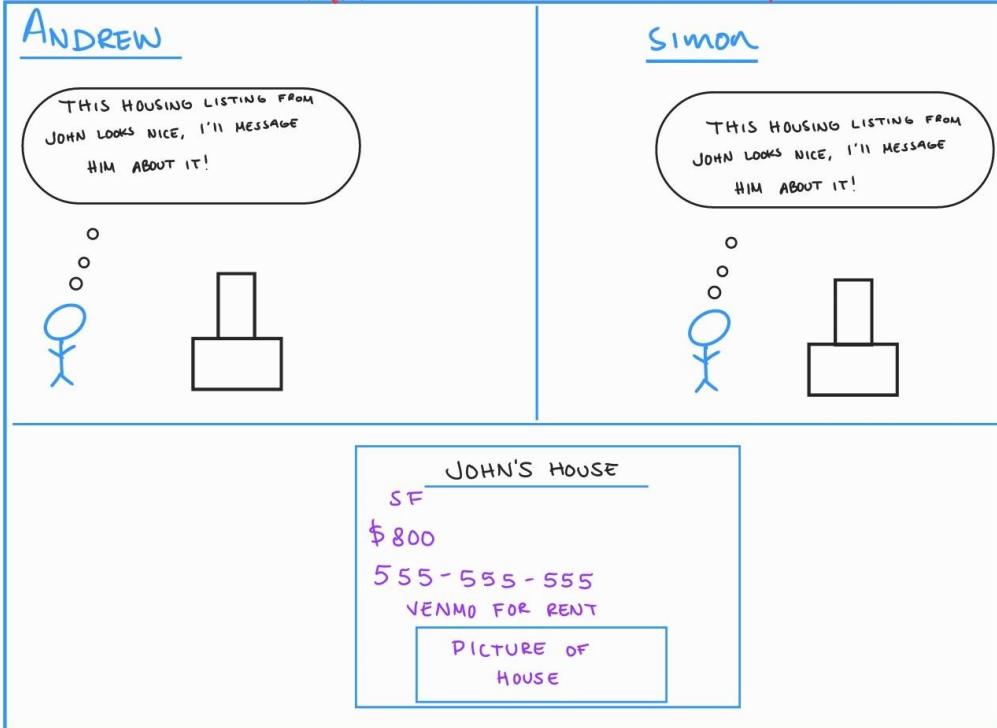
HOUSING 1

DROP DOWN BAR

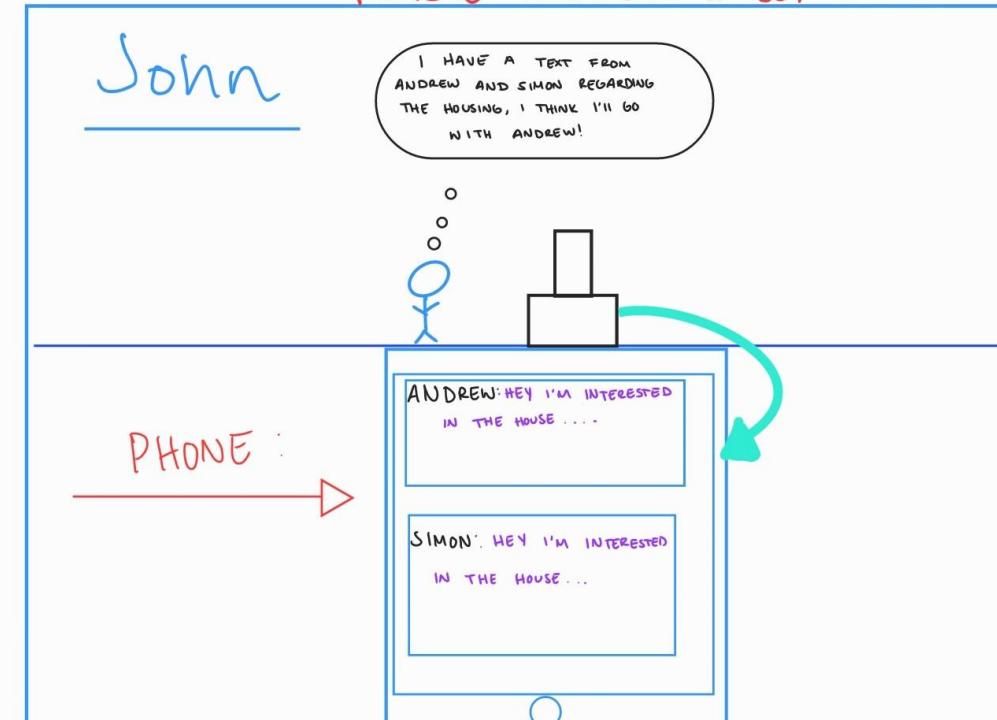
HOUSING 2

HOUSING 3

MAKE REQUEST TO BUY LISTING



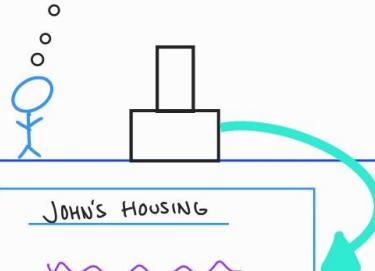
RECEIVE REQUEST TO BUY LISTING



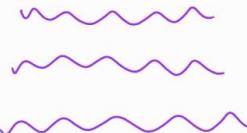
CLOSED LISTING

John

TIME TO CLOSE
THE LISTING SINCE I'M
GOING WITH ANDREW



JOHN'S HOUSING

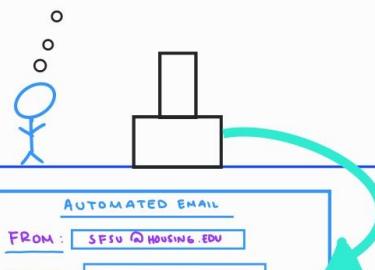


TAKE DOWN POSTING?

RECEIVES EMAIL SAYING LISTING IS CLOSED

Simon

THAT SUCKS THAT JOHNS LISTING
IS CLOSED, TIME TO SEARCH
FOR ANOTHER



AUTOMATED EMAIL

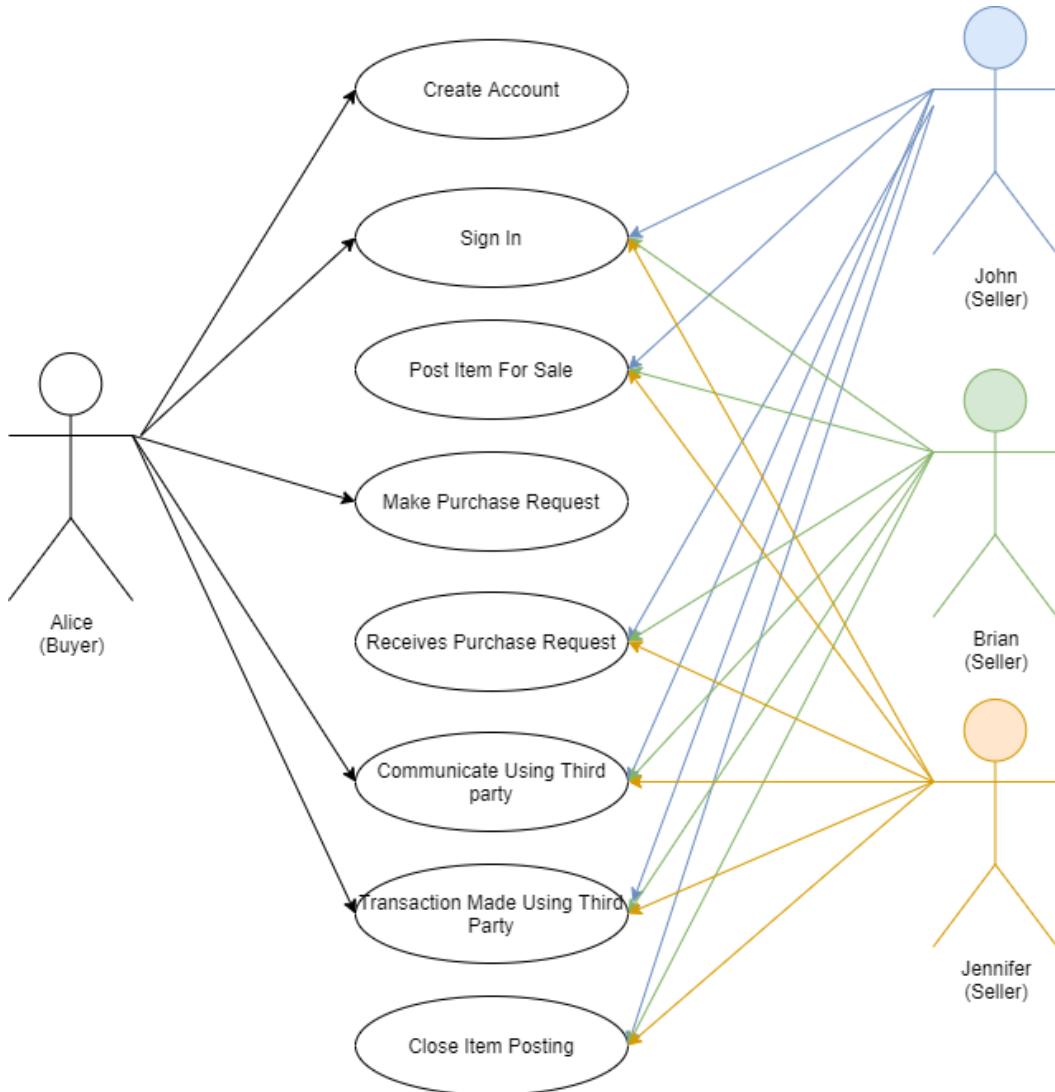
FROM: SFSU@HOUSING.EDU

SUBJECT: LISTING CLOSED

BODY: HELLO SIMON,
WE REGRET TO INFORM
YOU THAT JOHN'S LISTING
IS CLOSED.

Case 2: Student at SFSU looking for cheaper textbooks

Actors: Alice(Buyer), John(Seller), Brian(Seller), Jennifer(Seller)

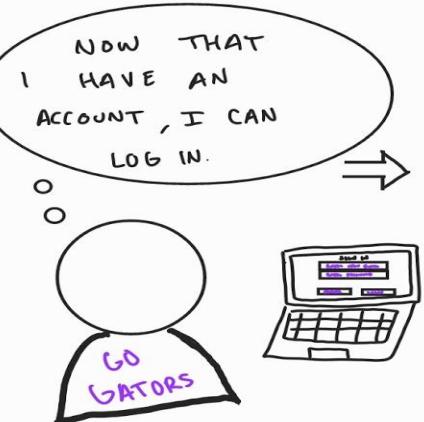


CREATE ACCOUNT - ALICE



A wireframe-style diagram of a registration form titled "REGISTER". It includes fields for "ENTER NAME", "ENTER SFSU EMAIL", "CREATE PASSWORD", and a checked checkbox labeled "AGREE TO TERMS OF PRIVACY AND SERVICE". A "CREATE ACCOUNT" button is at the bottom.

SIGN IN - ALICE



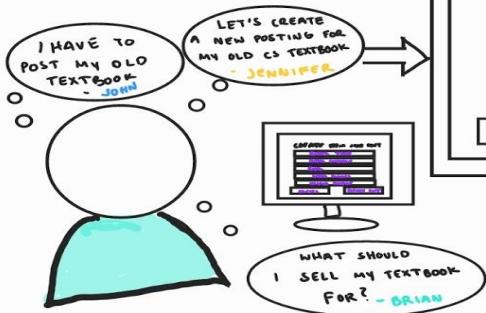
A wireframe-style diagram of a login form titled "SIGN IN". It includes fields for "ENTER SFSU EMAIL" and "ENTER PASSWORD", and buttons for "CANCEL" and "LOGIN".

SIGN IN - JOHN, BRIAN, JENNIFER



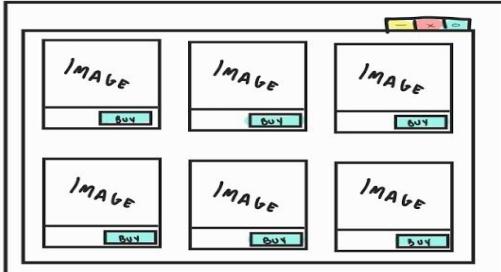
A wireframe-style diagram of a login form titled "SIGN IN". It includes fields for "ENTER SFSU EMAIL" and "ENTER PASSWORD", and buttons for "CANCEL" and "LOGIN".

POST ITEM - JOHN, BRIAN, JENNIFER



CREATE ITEM SALE POST	
ENTER TITLE	
ENTER CONDITION	
ENTER OTHER DETAILS	
ENTER PRICING	
UPLOAD PICTURE	
CANCEL	CREATE POST

MAKE PURCHASE REQUEST - ALICE



RECEIVING PURCHASE REQUEST - JOHN, BRIAN, JENNIFER

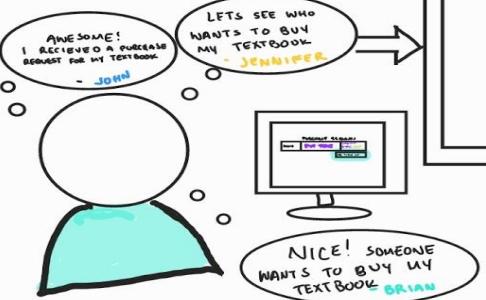
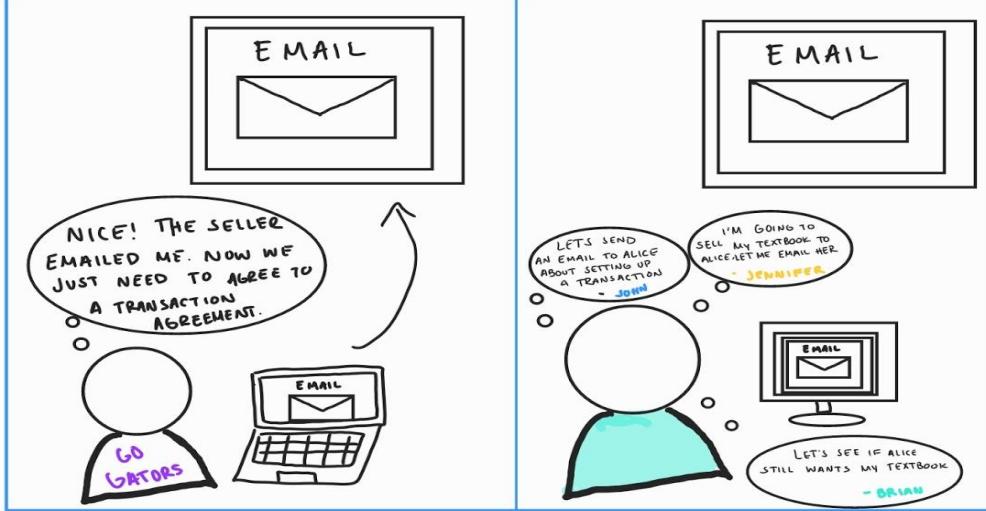
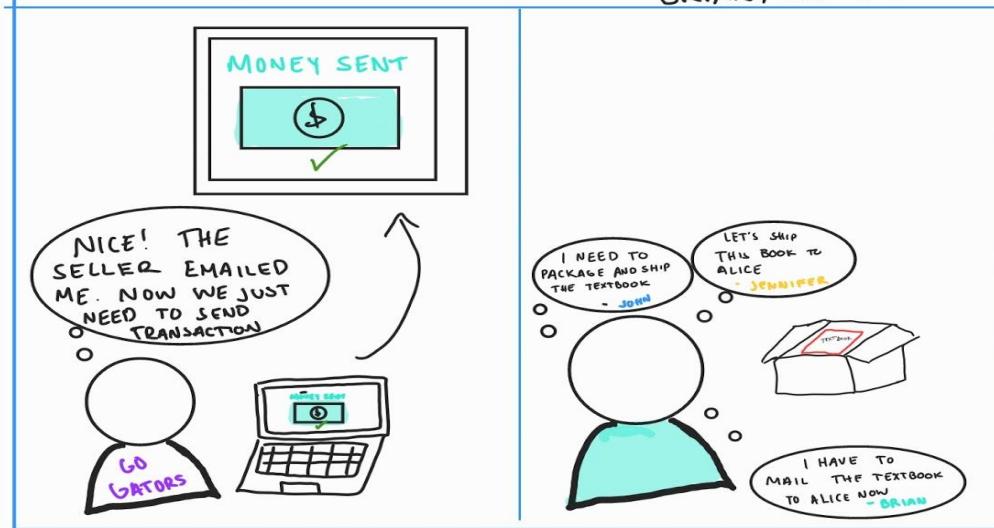


IMAGE	POST TITLE	REQUEST BUYER: ALICE
		SEE CONTACT INFO

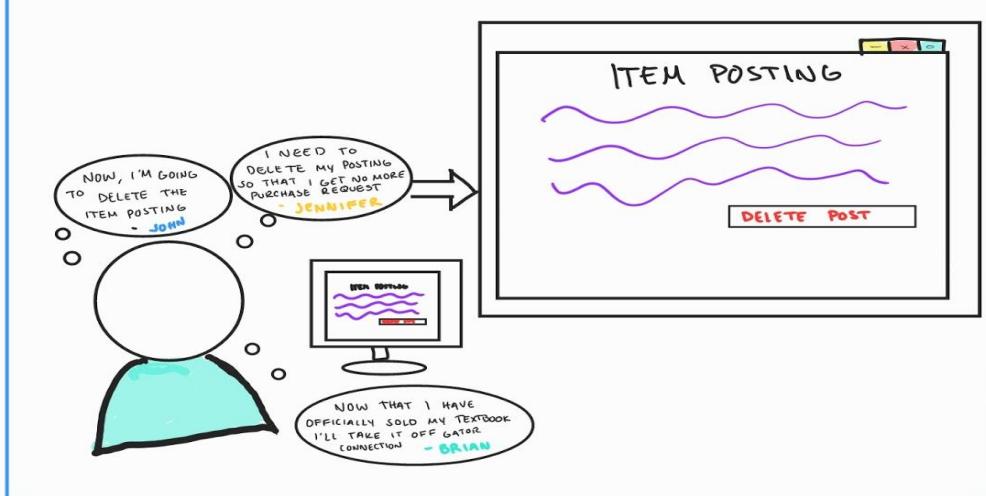
THIRD PARTY COMMUNICATION - ALICE, JOHN, BRIAN, JENNIFER



COMPLETE TRANSACTION - ALICE, JOHN, BRIAN, JENNIFER

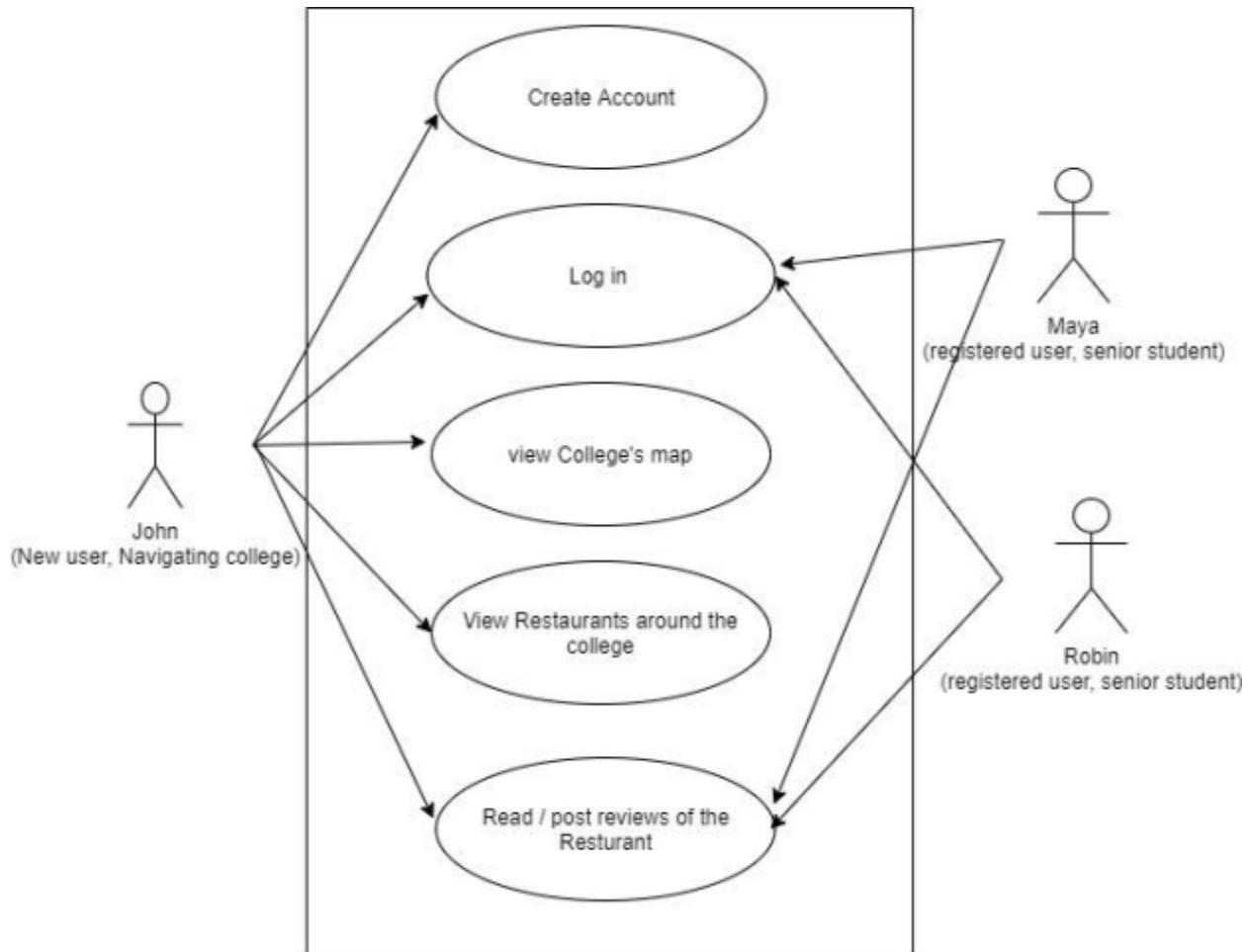


CLOSE POSTING - JOHN, BRIAN, JENNIFER



Use Case 3: Student at SFSU looking for food on campus, with his two friends outside SFSU accompanying him

Actors: John(New User), Maya(Unregistered User), Robin(Unregistered User)



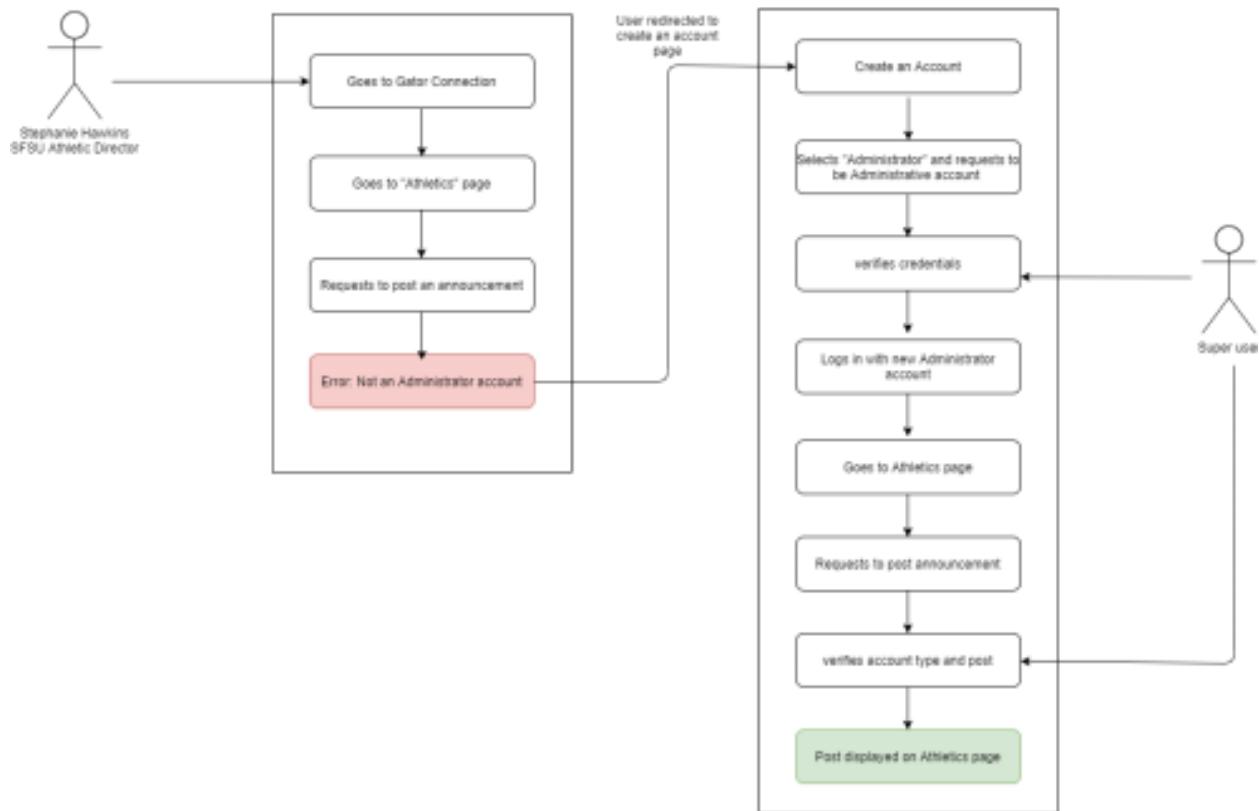
<p>JOHN, A FRESHMAN AT SFSU, WANTS TO BE FAMILIAR WITH THE CAMPUS & ITS SURROUNDINGS</p> <p>AFTER A LITTLE WHILE, JOHN GETS HUNGRY</p>	<p>THROUGH THE GATOR CONNECTION WEBSITE, JOHN CAN SEE ALL THE CAMPUS BUILDINGS AND ITS SURROUNDING</p> <p>GATOR HOME RESTURANTS MAPS About MAPS</p> <p>HE AGAIN VISITS GATOR CONNECTION WHERE HE CAN SEE RESTURANTS AROUND CAMPUS HOPEING TO FIND STH GOOD TO EAT</p>
--	--

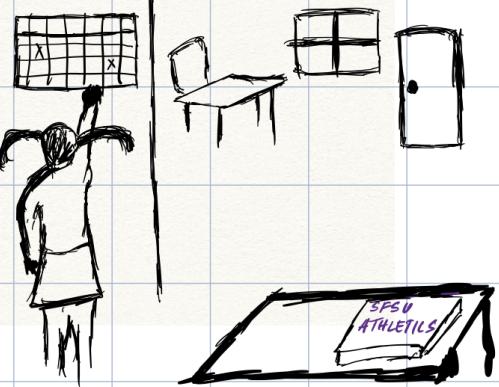
<p>AFTER A LITTLE WHILE, JOHN GETS HUNGRY</p>	<p>HE AGAIN VISITS GATOR CONNECTION WHERE HE CAN SEE RESTURANTS AROUND CAMPUS HOPEING TO FIND STH GOOD TO EAT</p> <table border="1"> <tr> <td>HOME</td><td>RESTURANTS</td><td>MAPS</td><td>HOUSINGS</td><td>ITEMS</td></tr> <tr> <td>KRABBY PATTY</td><td>★★★★★</td></tr> <tr> <td>ICHIRAKU RAMEN</td><td>★★★★★</td></tr> <tr> <td>CENTRAL PERK</td><td>★★★★★</td></tr> </table>	HOME	RESTURANTS	MAPS	HOUSINGS	ITEMS	KRABBY PATTY	★★★★★	ICHIRAKU RAMEN	★★★★★	CENTRAL PERK	★★★★★
HOME	RESTURANTS	MAPS	HOUSINGS	ITEMS								
KRABBY PATTY	★★★★★											
ICHIRAKU RAMEN	★★★★★											
CENTRAL PERK	★★★★★											

<p>APFTER LOOKING AT SOME REVIEWS POSTED BY SFSU STUDENTS JOHN GOES TO A RESTURANT LOOKING AT ITS HIGH RATINGS</p> <p>HOWEVER JOHN DID NOT LIKE THE FOOD & HE THOUGHT ABOUT LEAVING A REVIEW</p>	<p>- <input type="checkbox"/> X</p> <p>GATOR CONNECTION. Sorry, only registered users are able to post a review.</p> <p>Sign me up</p> <p>JOHN then Signs Up Using SFSU email and leaves a review to the restaurant.</p>
--	---

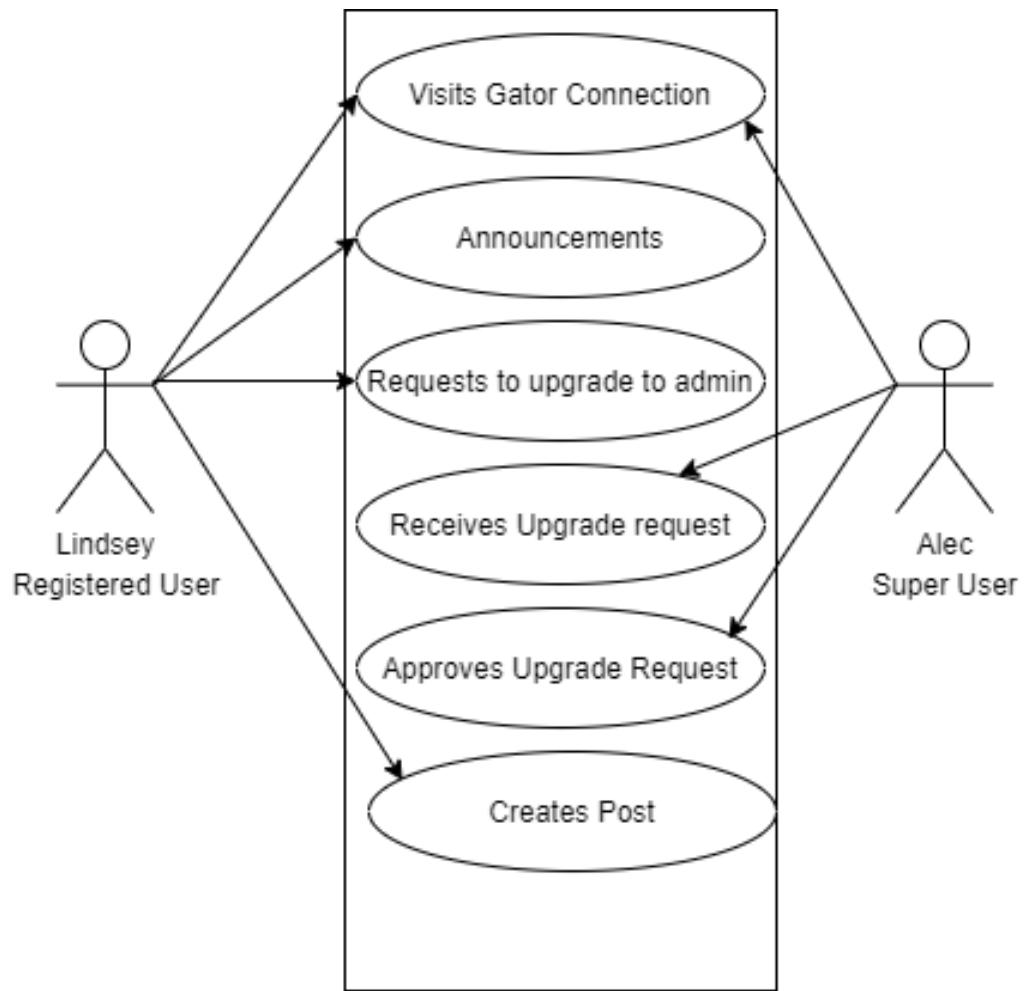
Case 4: Athletic Director at SFSU wants to announce upcoming events

Actors: Stephanie(User, Athletic Director)

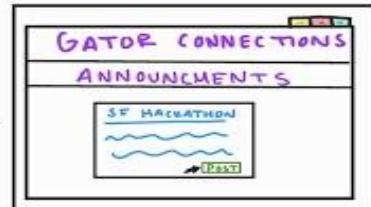


<p>Stephanie Hawkins is a athletics director in the SFSU</p>	<p>She realized Most of the students use GATOR connection & decided to post the announcement there.</p>
	<p>GATOR CONNECTION · ANNOUNCEMENTS</p> <ul style="list-style-type: none"> MAKE NEW ANNOUNCEMENT EDIT ANNOUNCEMENTS DELETE ANNOUNCEMENTS
<p>SHE WANTS TO ANNOUNCE the UPCOMING basketball GAME hosting at SFSU</p>	
<p>However, making announcements takes special privilege and she gets an error.</p>	<p>Stephanie successfully creates an administrative account and could successfully post an announcement so everyone can see it.</p>
<p>GATOR CONNECTION ANNOUNCEMENTS</p> <p>Sorry, but announcements can be made only with administrative account if you are a SFSU director please create a administrative account.</p> <p><u>Create a administrative account</u></p>	<p>GATOR CONNECTION HOME SHOP ANN</p> <p>ANNOUNCEMENTS ...</p> <p>ADVERTISEMENTS ...</p> <p>Stephanie's announcement is shown on the home page carousel.</p>

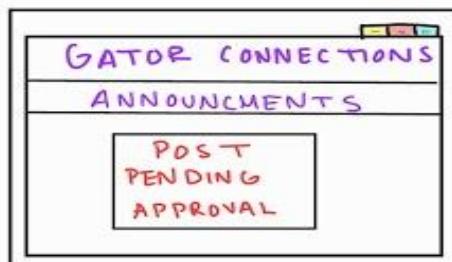
Case 5: President of SF Hacks wants to post about her upcoming events
Actors: Lindsey(Registered User), Alec(Super User)



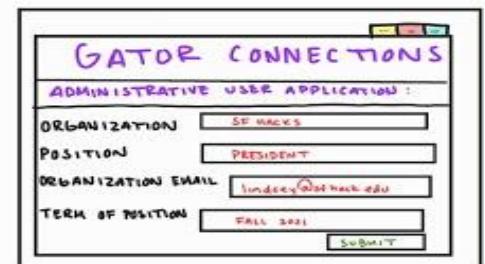
LINDSEY WANTS TO POST SF HACK'S SOCIAL MEDIA, ANNOUNCEMENTS, AND FUTURE EVENTS ON GATOR CONNECTIONS.



LINDSEY TRIES TO POST AN ANNOUNCEMENT BUT SHE REALIZES SHE DOESN'T HAVE FULL ACCESS AS ADMINISTRATIVE USER, THEREFORE HER POST IS PENDING.



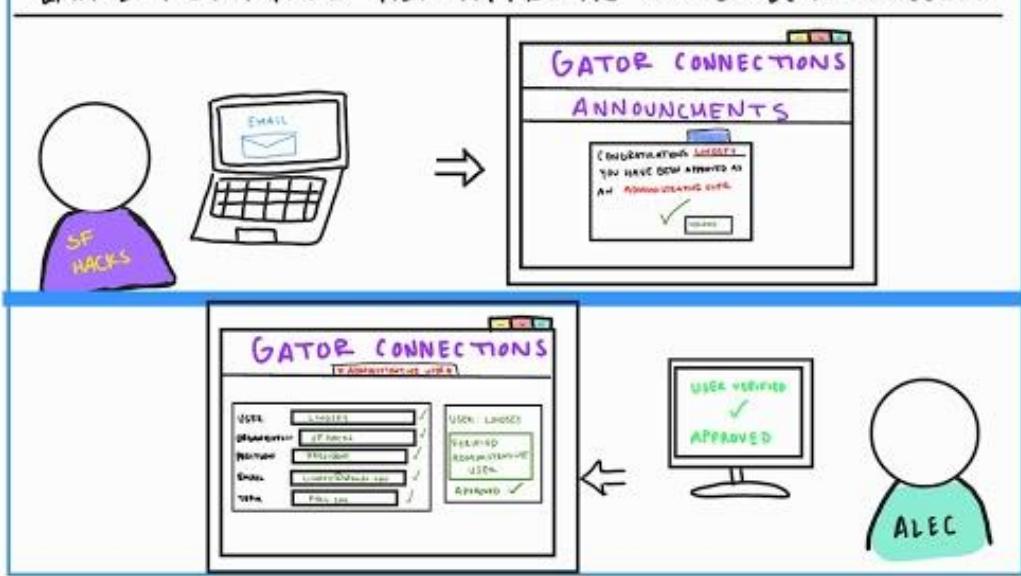
LINDSEY DECIDES THAT SINCE SHE IS PRESIDENT OF SF HACKS SHE CAN APPLY AS AN ADMINISTRATIVE USER SO HER POSTS DON'T HAVE TO GO THROUGH A PENDING PROCESS



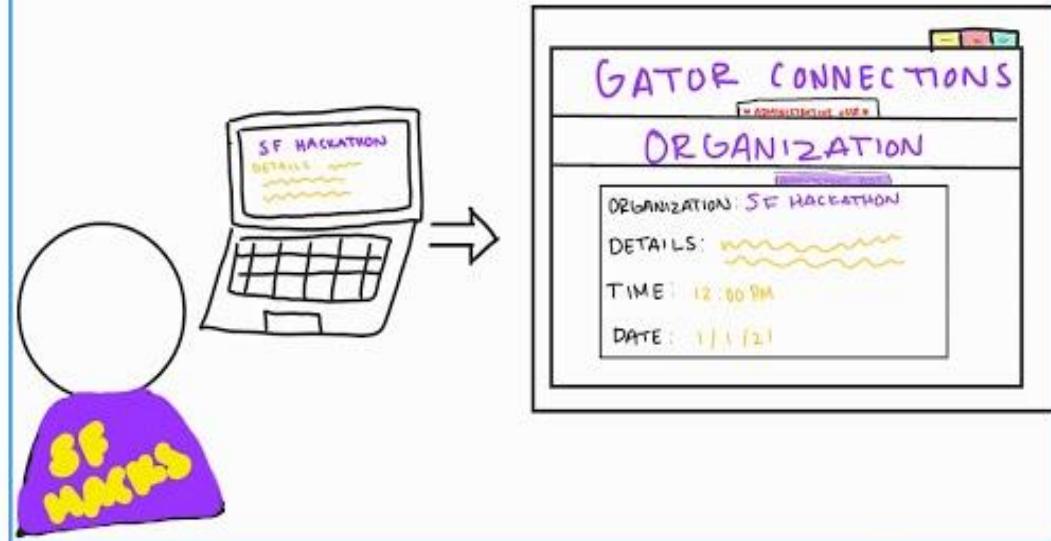
WHILE LINDSEY IS WAITING FOR OFFICIAL CONFIRMATION, SUPER USER ALEC HAS RECEIVED HER REQUEST.



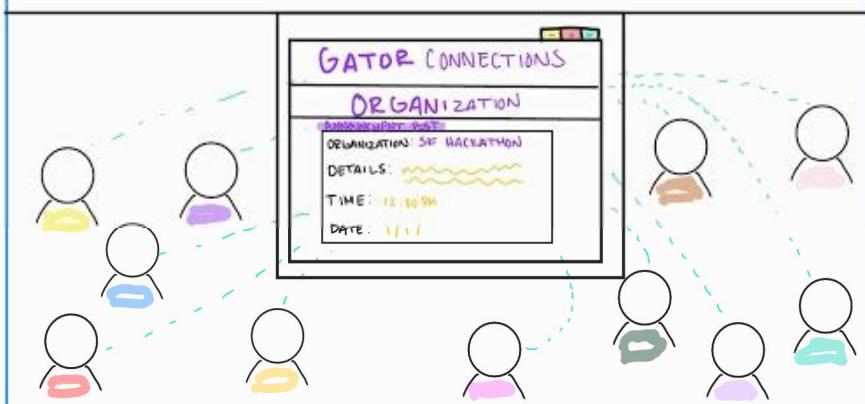
ALEC VERIFIES LINDSEY'S INFORMATION AND APPROVES HER REQUEST AND LINDSEY RECIEVES AN EMAIL NOTIFYING HER APPROVAL TO UPGRADE HER ACCOUNT.



NOW THAT LINDSEY HAS ADMINISTRATIVE PRIVILEGES AS PRESIDENT SHE POSTS AN UPCOMING SF HACKATHON EVENT WITH DETAILS.

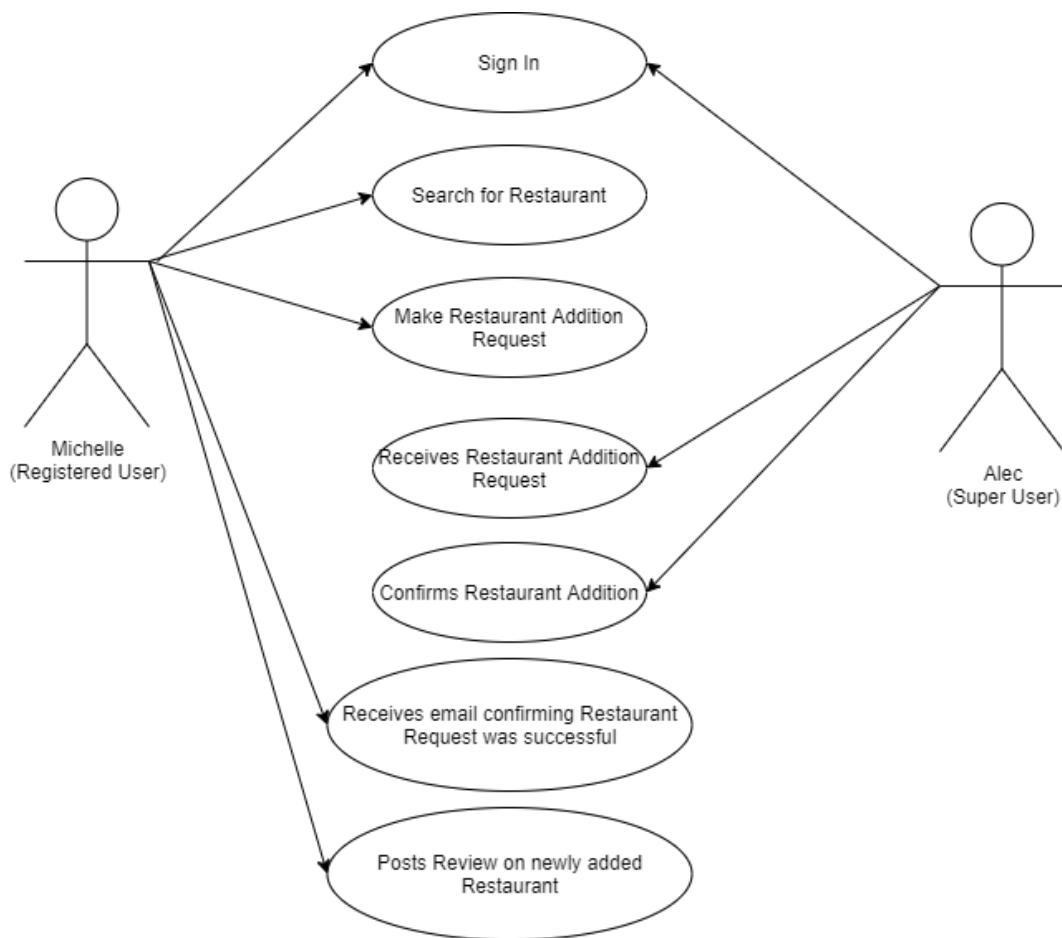


AFTER POSTING HER POST, LINDSEY WENT TO THE "ORGANIZATIONS" SECTION OF GATOR CONNECTIONS AND HER ANNOUNCEMENT WAS AVAILABLE FOR EVERYONE TO SEE RIGHT AWAY.

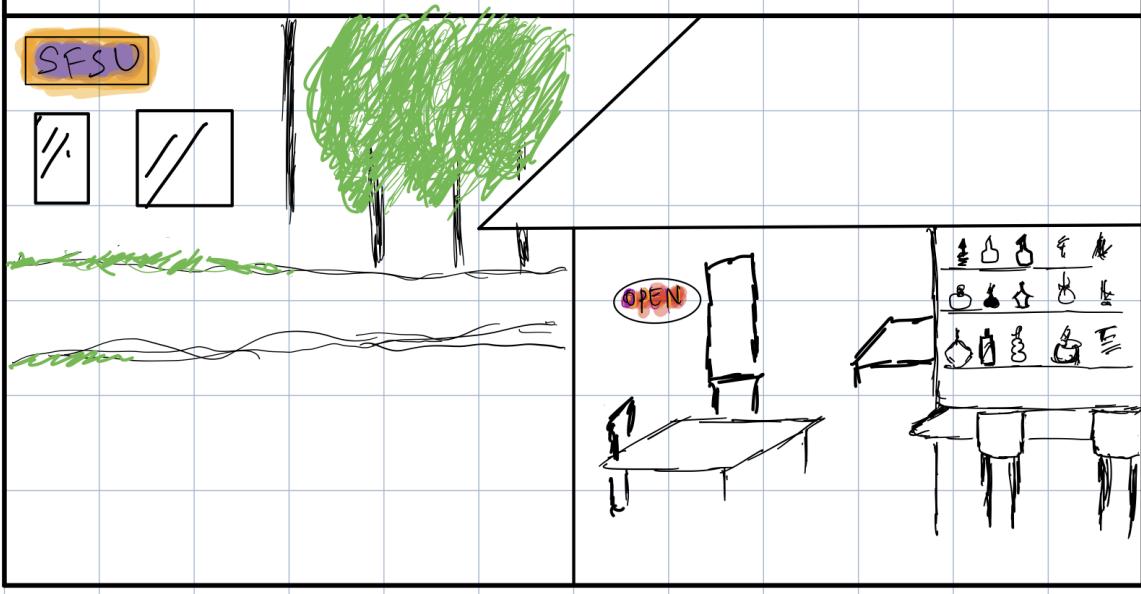


Case 6: Student at SFSU wants to write a review about a restaurant she liked

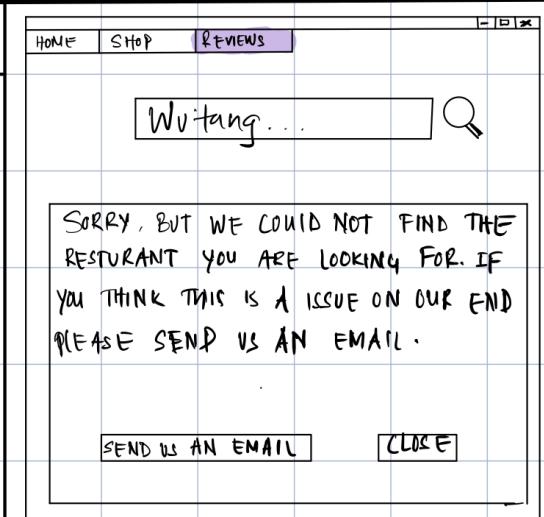
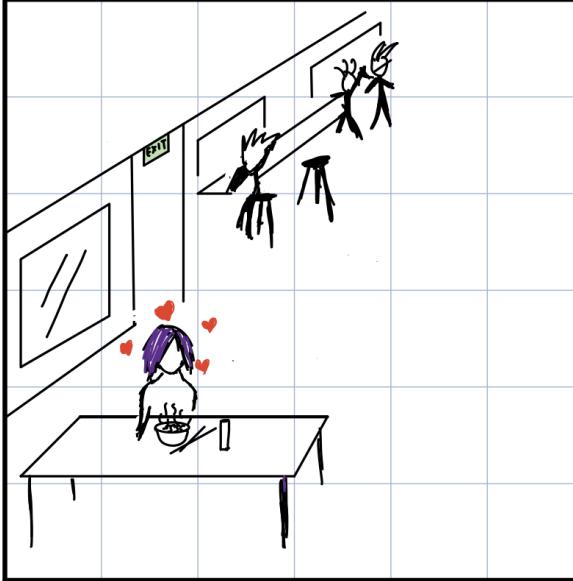
Actors: Michelle (Registered User), Alec (Super User)



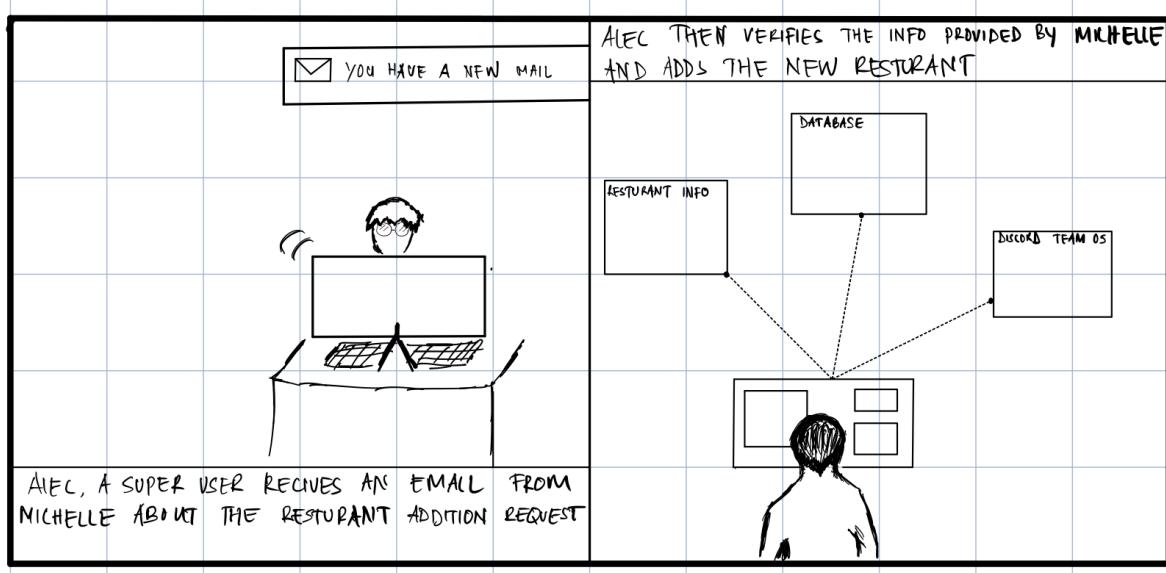
A NEW RESTURANT NEAR SFSU WAS OPENED RECENTLY



Michelle, a SFSU student loved the food & thought about leaving a positive review in Gator Connection.



MICHELLE SENDS THE ADMIN AN EMAIL WITH THE NAME & ADDRESS OF THE RESTURANT.



MICHELLE GETS AN EMAIL SAYING HER REQUEST HAS BEEN APPROVED.

HOME | SHOP | RESTURANTS | REVIEWS

WUTANG ★★★★★

NO REVIEWS FOUND

ADD A REVIEW

★★★★★

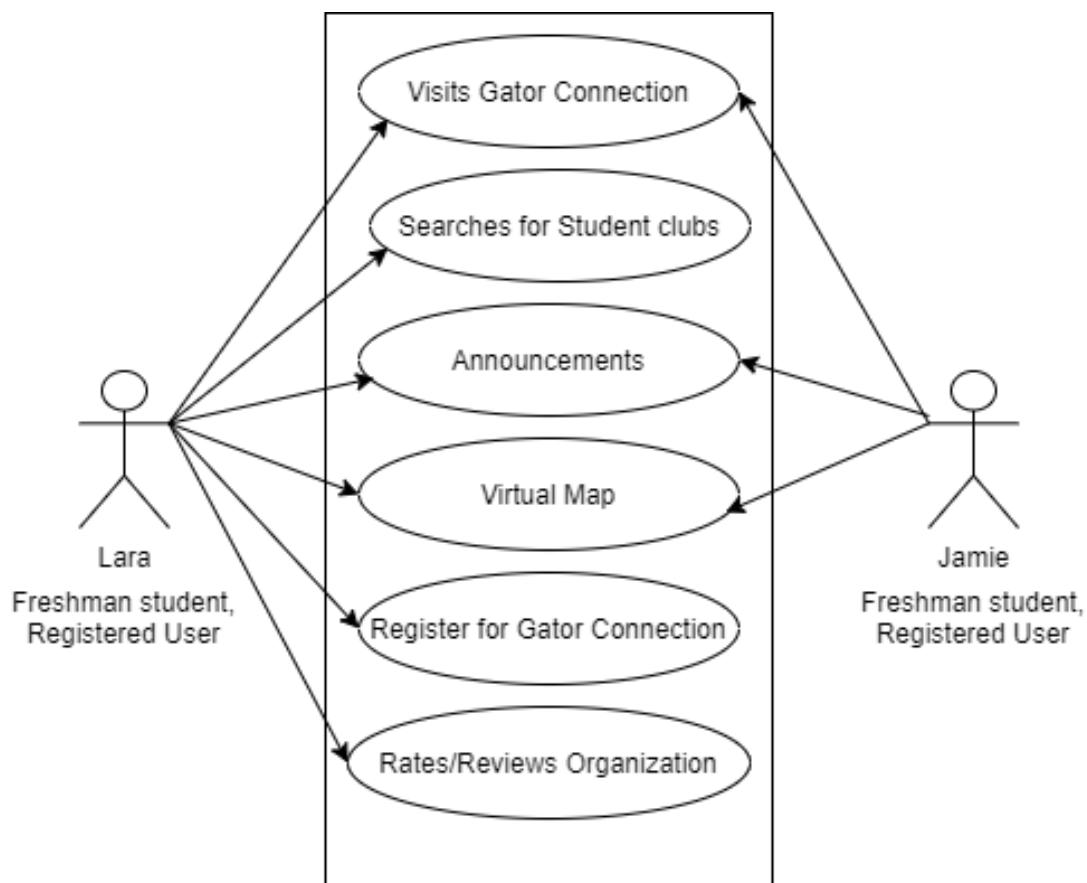
MUST TRY YUM YUM !!!

SHE CHECKS GATOR CONNECTION & FINDS THE NEW RESTURANT LISTED.

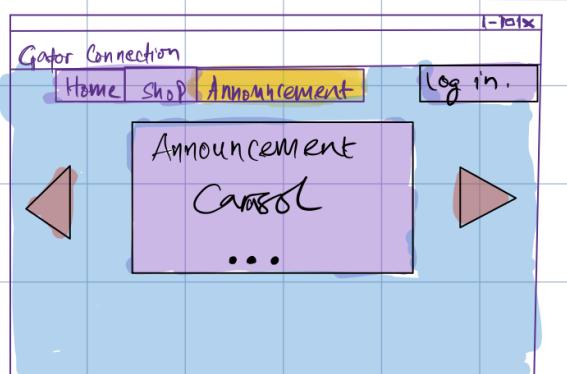
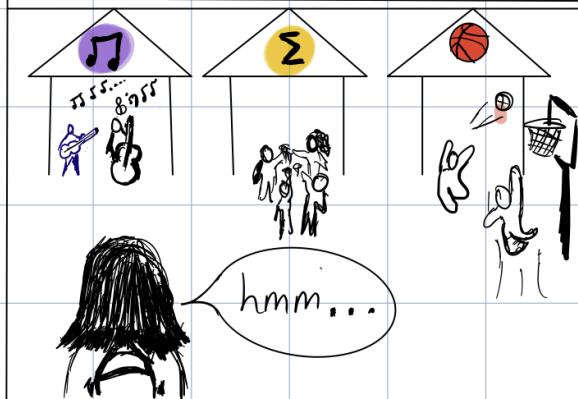
MICHELLE SUCESSFULLY POSTS A RAVING REVIEW WITH A PICTURE

Case 7: Two Freshman at SFSU want to join a club but want more information

Actors: Lara(Registered User), Jamie(Registered User)

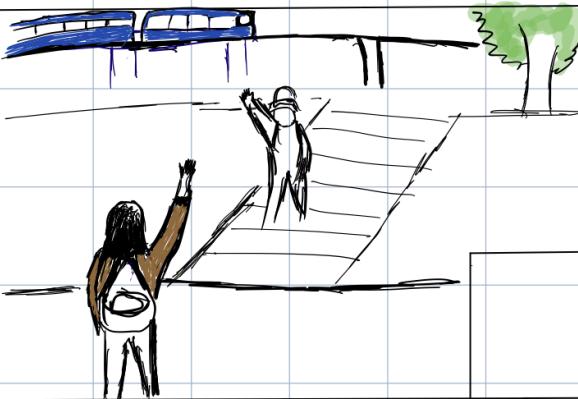


LARA, a Freshman at SFSU is deciding which club to join at the campus.

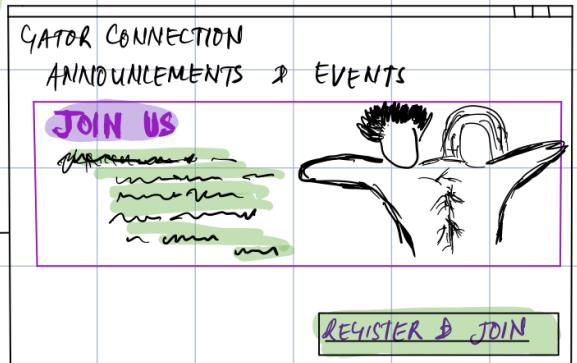


LARA Checks the GATOR CONNECTION ANNOUNCEMENT to see if there is any event that may peak her interest.

LARA MEETS HER HIGH SCHOOL FRIEND JAIME WHO IS ALSO ATTENDING SFSU

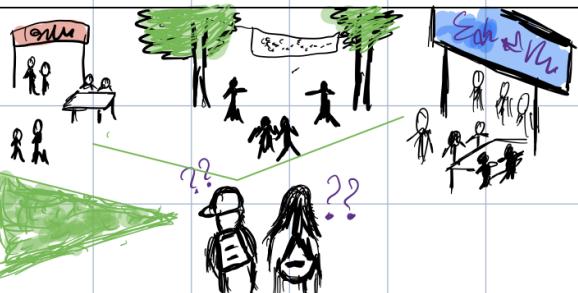


JAYME ALSO VISITS GATOR CONNECTION AND THEY BOTH REGISTER FOR AN EVENT THEY LIKE



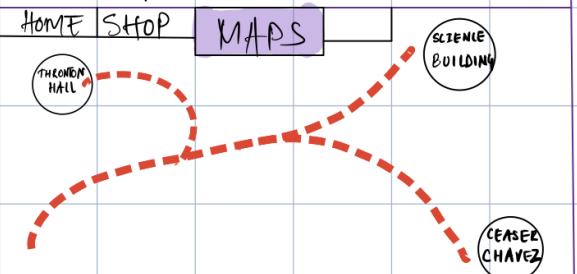
LARA tells JAIME ABOUT THE EVENTS GOING AND THEY BOTH DECIDE TO JOIN A EVENT TOGETHER

AFTER REGISTERING AND JOINING THE EVENT BOTH GET EMAIL CONFIRMATION



HOWEVER THEY DON'T KNOW WHERE THE EVENT IS HOSTED.

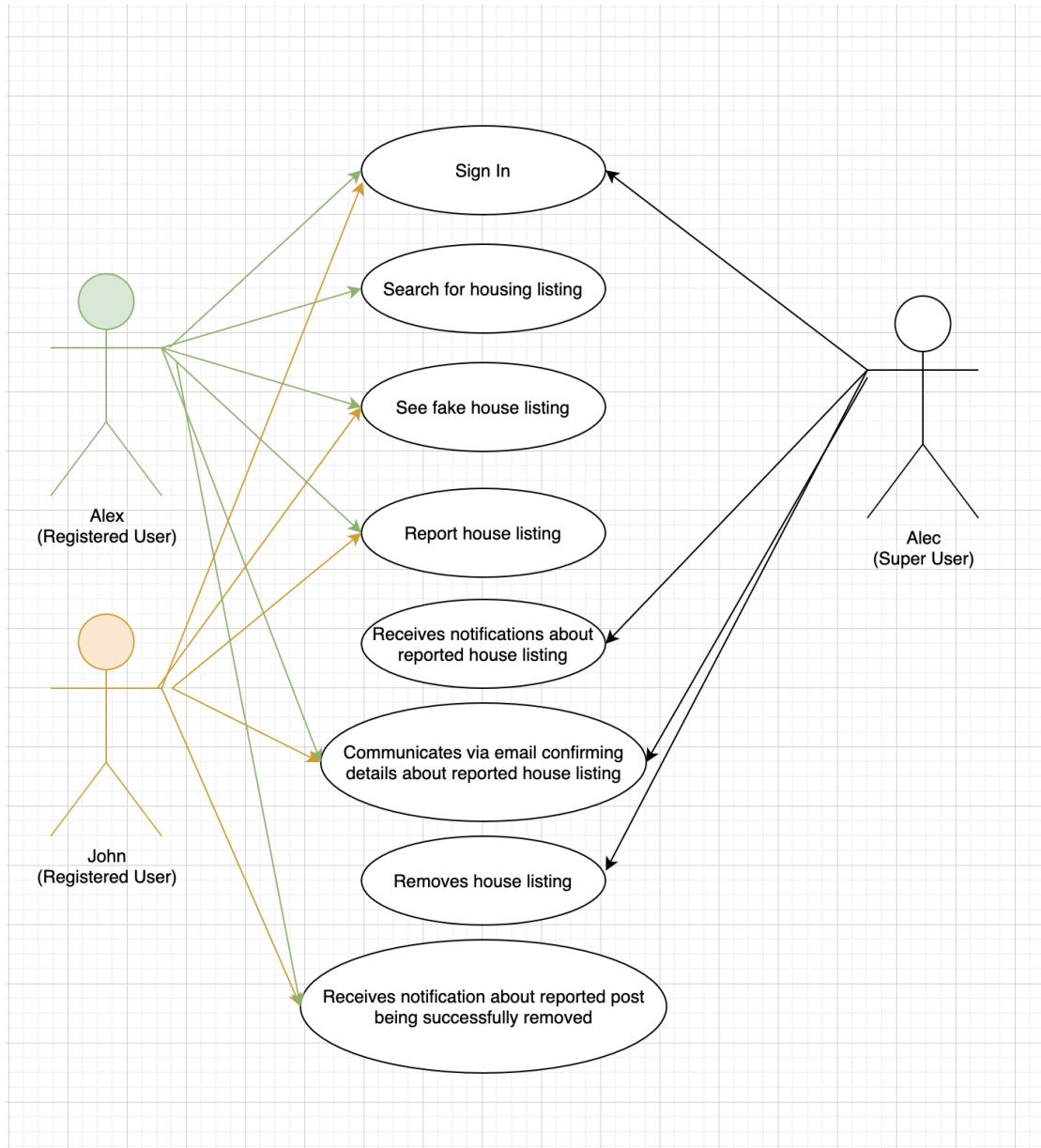
THEY GO TO GATOR CONNECTION ONCE MORE WHERE THEY CAN SEE THE MAP OF THE CAMPUS



BOTH SUCESSFULLY ATTENDS THE EVENT

Case 8: A registered user reports a false housing listing

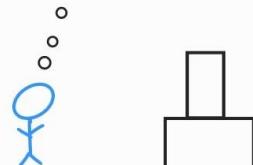
Actors: John (Registered User), Alex (Registered User), Alec (Super User)



LOG IN

ALEX

IN THE SEMESTER I
WANT TO GET A NEW
PLACE TO LIVE



LOG IN

SFSU EMAIL

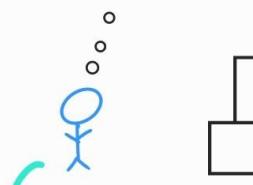
PASSWORD

LOG IN

SEARCH FOR HOUSING

ALEX

LET'S GO SEARCH
FOR HOUSING



HOUSING

SEARCH FOR

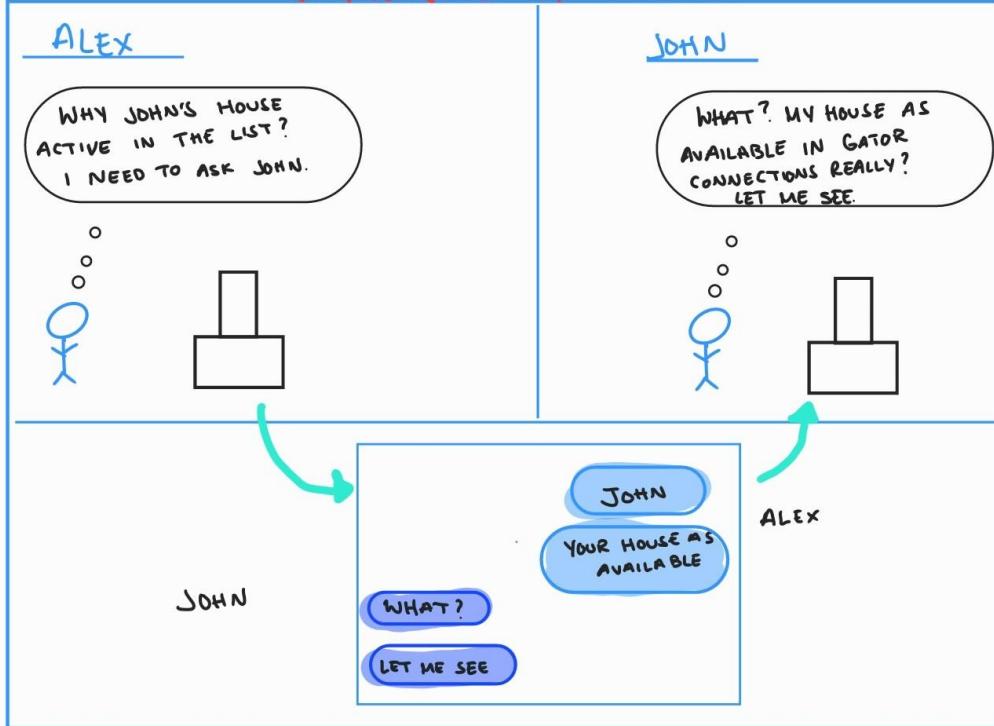
DROP DOWN BAR

HOUSING 1
JOHN

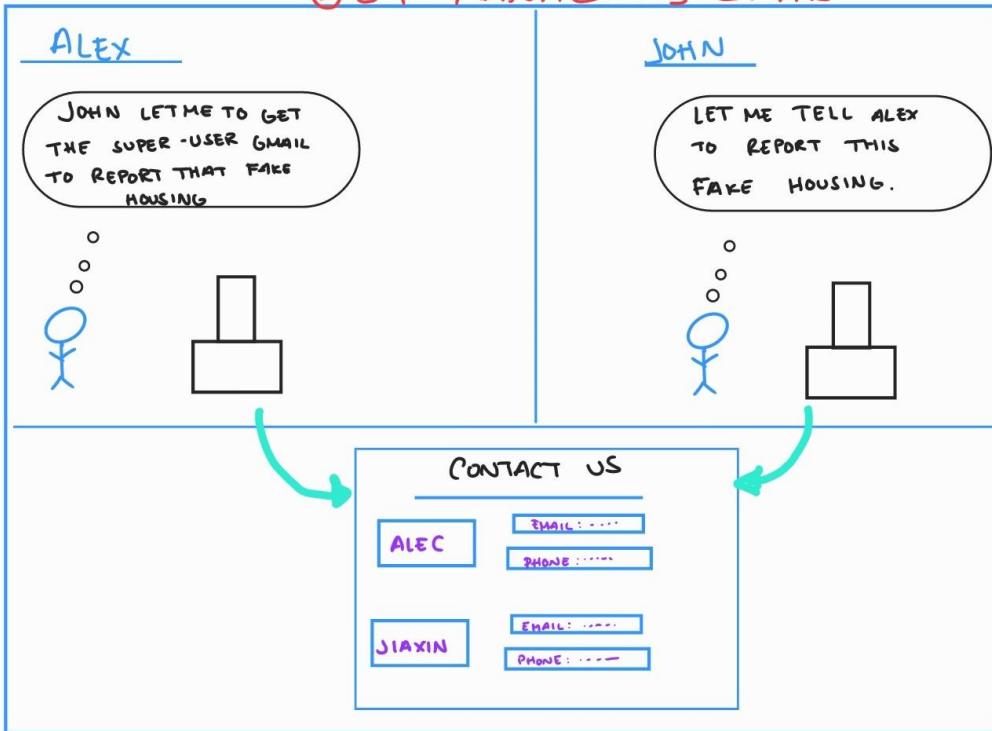
HOUSING 2

HOUSING 3

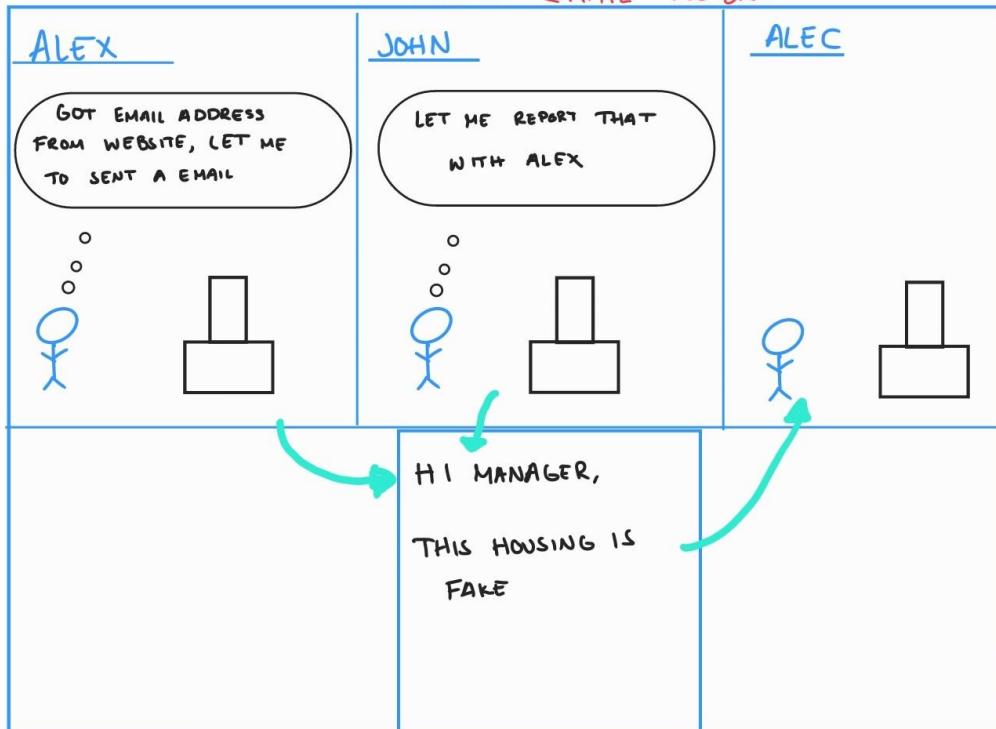
COMMUNICATE BY MESSAGE



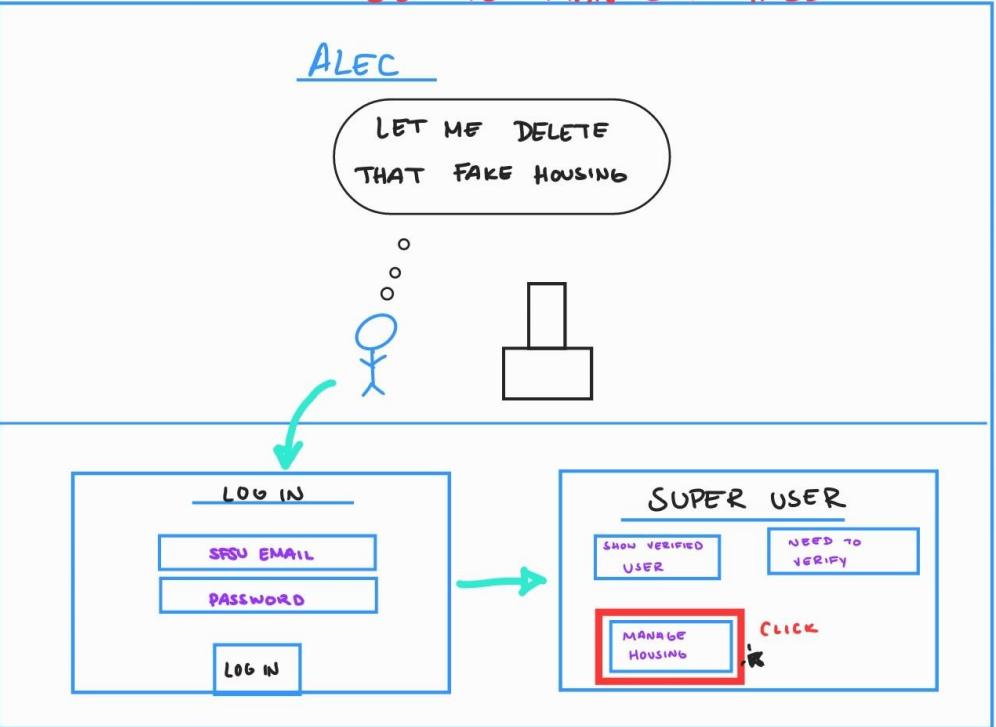
GET MANAGER'S EMAIL



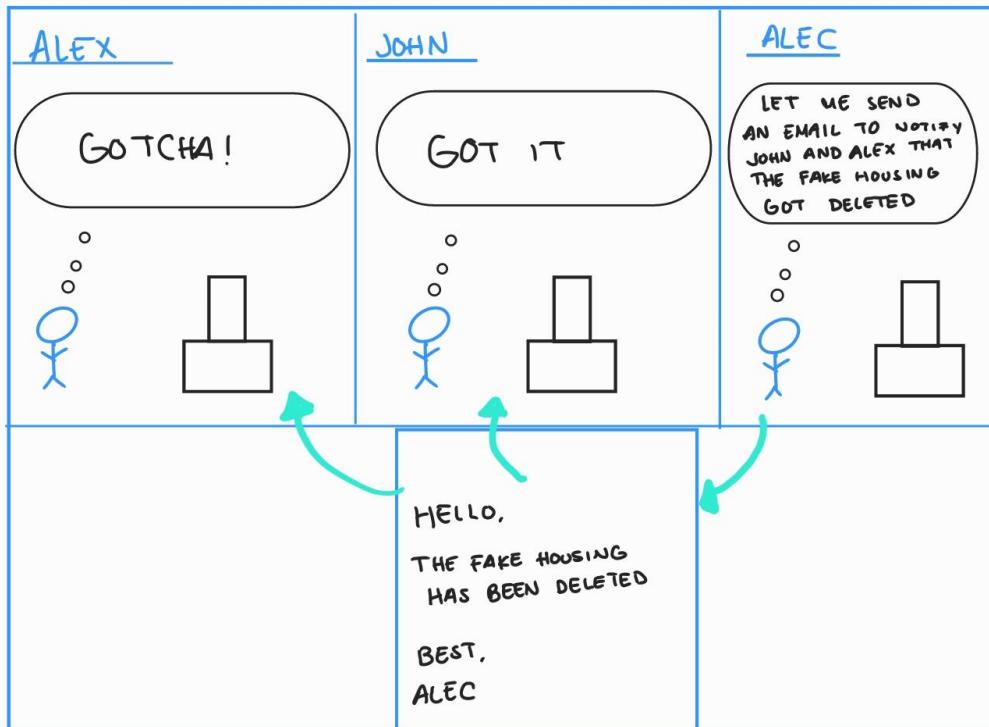
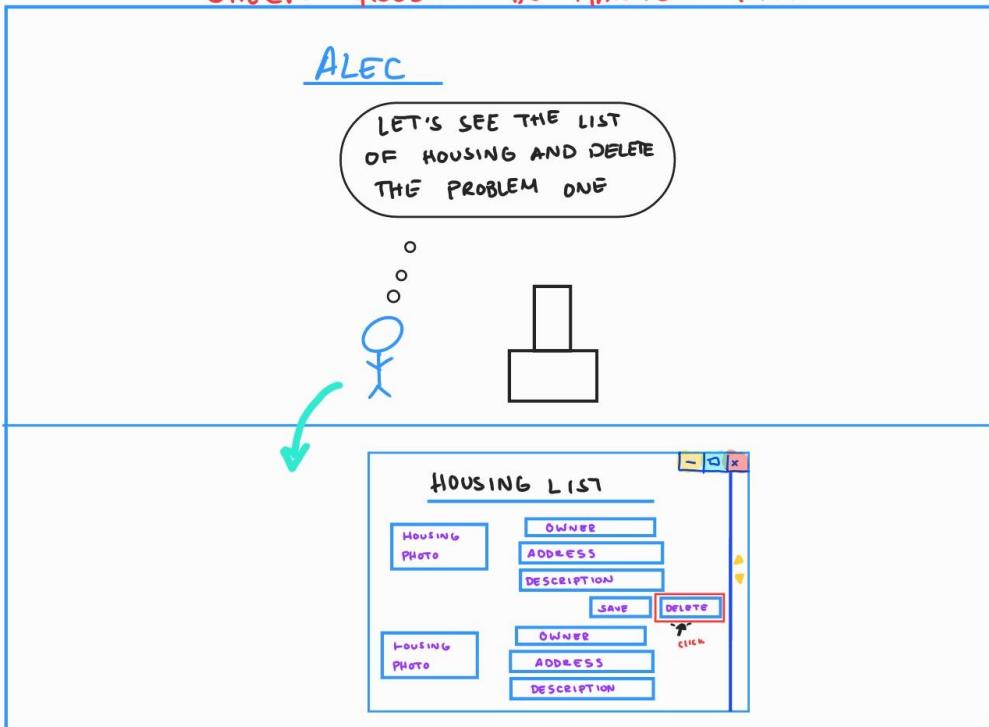
SEND EMAIL REPORT



GO TO MANAGER PAGE



CHECK HOUSING IN MANAGER PAGE



High Level Database Architecture and Organization

Business Rules

1. Registered User
 - a. A registered user shall be able to create zero or many housing listing requests.
 - b. A registered user shall have zero or many housing listing request notifications.
 - c. A registered user shall be able to fill out one housing listing form for one housing listing request.
2. Housing Listing Request
 - a. A housing listing request shall be created by one and only one registered user.
 - b. A housing listing request shall be for one housing listing post.
 - c. A housing listing request shall create one housing listing request notification.
 - d. A housing listing request shall have one housing listing form.
 - e. A housing listing request shall have a date of when the request was made.
3. Housing Listing Request Notification
 - a. A housing listing request notification shall have one housing listing form.
 - b. A housing listing request notification shall notify one and only one registered user.
 - c. A housing listing request notification shall have one and only one housing listing request.
 - d. A housing listing request notification shall have one and only one unique notification id.
 - e. A housing listing request notification shall have a date of when the notification was created.
4. Housing Listing Post
 - a. A housing listing post shall be posted by one and only one registered user.
 - b. A housing listing post shall have a unique housing id.
 - c. A housing listing post shall have a title.
 - d. A housing listing post shall have an item description.
 - e. A housing listing post shall have a price.
 - f. A housing listing post shall have many images.
 - g. A housing listing post shall be able to be removed by one and only one registered user that posted the housing listing.
 - h. A housing listing post shall be requested by one or many registered users.
 - i. A housing listing post shall have one housing listing form.

5. Housing Listing Form

- a. A housing listing form shall be for one housing listing post.
- b. A housing listing form shall be filled out by one and only one registered user.
- c. A housing listing form shall have the registered user's full name.
- d. A housing listing form shall have an about me about the registered user.
- e. A housing listing form shall have an expected day they want to move in.

6. Image

- a. An image shall belong to one and only one post.
- b. An image shall have a file path to the image.

Entity, Attribute, Relationship, Domain Descriptions

1. General User (Strong)
 - a. user_id: key, numeric
2. Registered User (Strong)
 - a. reg_user_id: key, numeric
3. Image (Strong)
 - a. image_id: key, numeric
 - b. post_id: key, numeric
 - c. image_path: alphanumeric
4. Registered User Device (Weak)
 - a. reg_user_device_id: key, numeric
 - b. reg_user_id: key, numeric
 - c. device_id: key, numeric
 - d. createdAt: DateTime
5. Device Session (Weak)
 - a. device_session_id: key, numeric
 - b. device_id: key, numeric
 - c. account_id: key, numeric
 - d. createdAt: DateTime
6. Account (Weak)
 - a. account_id: key, numeric
 - b. full_name: composite, alphanumeric
 - c. password: alphanumeric
7. Student Account (Weak)
 - a. student_id: key, numeric
 - b. sfsu_email: key, numeric
 - c. graduation_year: numeric
8. Housing Listing Request (Weak)
 - a. housing_request_id: key, numeric
 - b. request_id: key, numeric
 - c. housing_id: key, numeric
 - d. createdAt: DateTime
9. Housing Listing Post (Weak)
 - a. post_id, key, numeric
 - b. housing_id: key, numeric
 - c. title: alphanumeric
 - d. description: alphanumeric
 - e. price: numeric

10. Housing Listing Form (Weak)

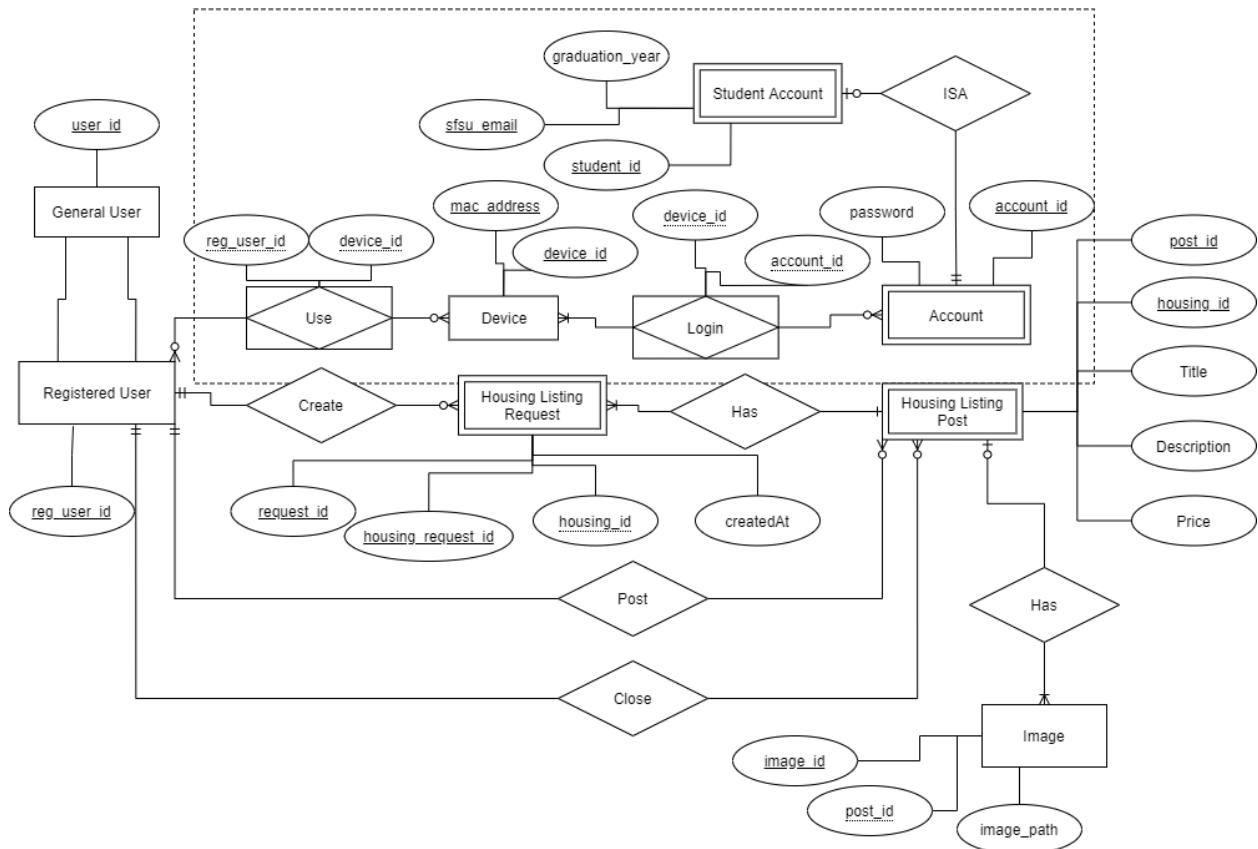
- a. form_id, key, numeric
- b. request_id, key, numeric
- c. housing_id: key, numeric
- d. full_name: composite, alphanumeric
- e. about_me: alphanumeric
- f. move_in: Date

11. Housing Listing Request Notification (Weak)

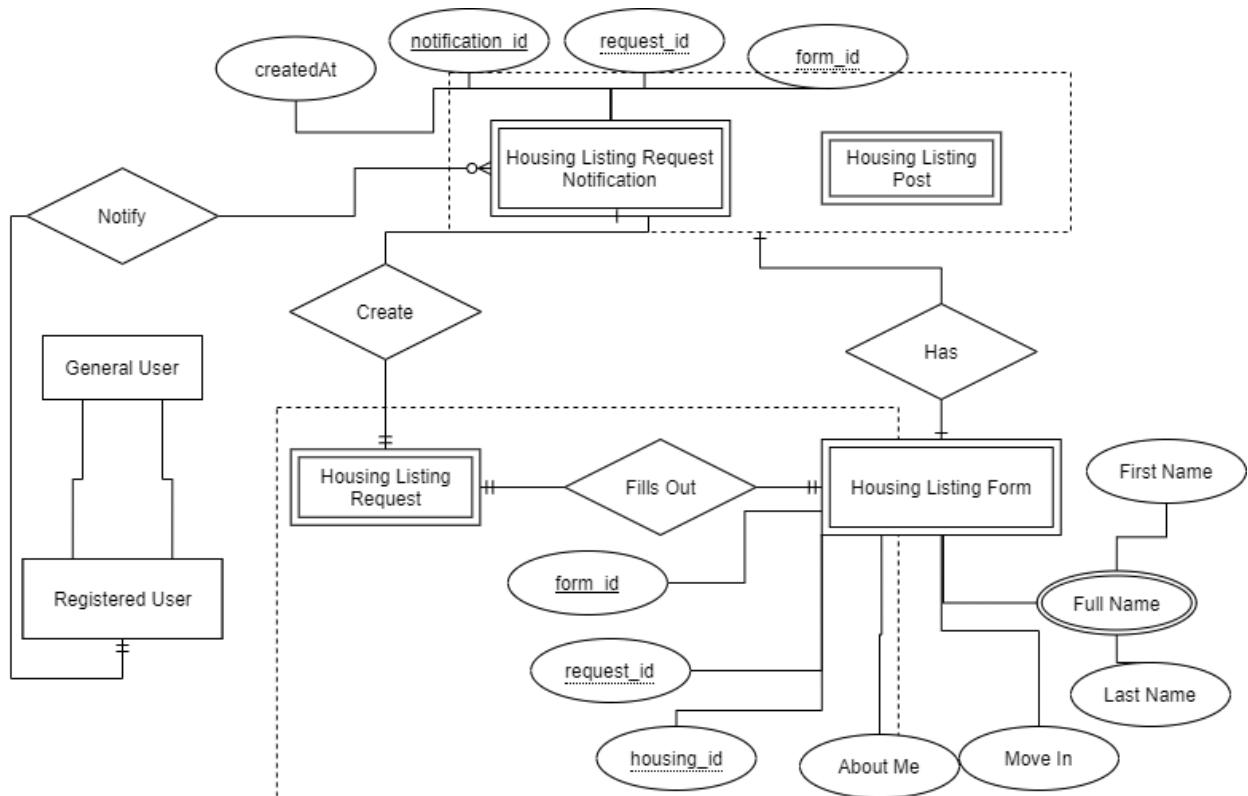
- a. notification_id: key, numeric
- b. form_id: key, numeric
- c. request_id: key, numeric
- d. createdAt: DateTime

Entity Relationship Diagram (ERD)

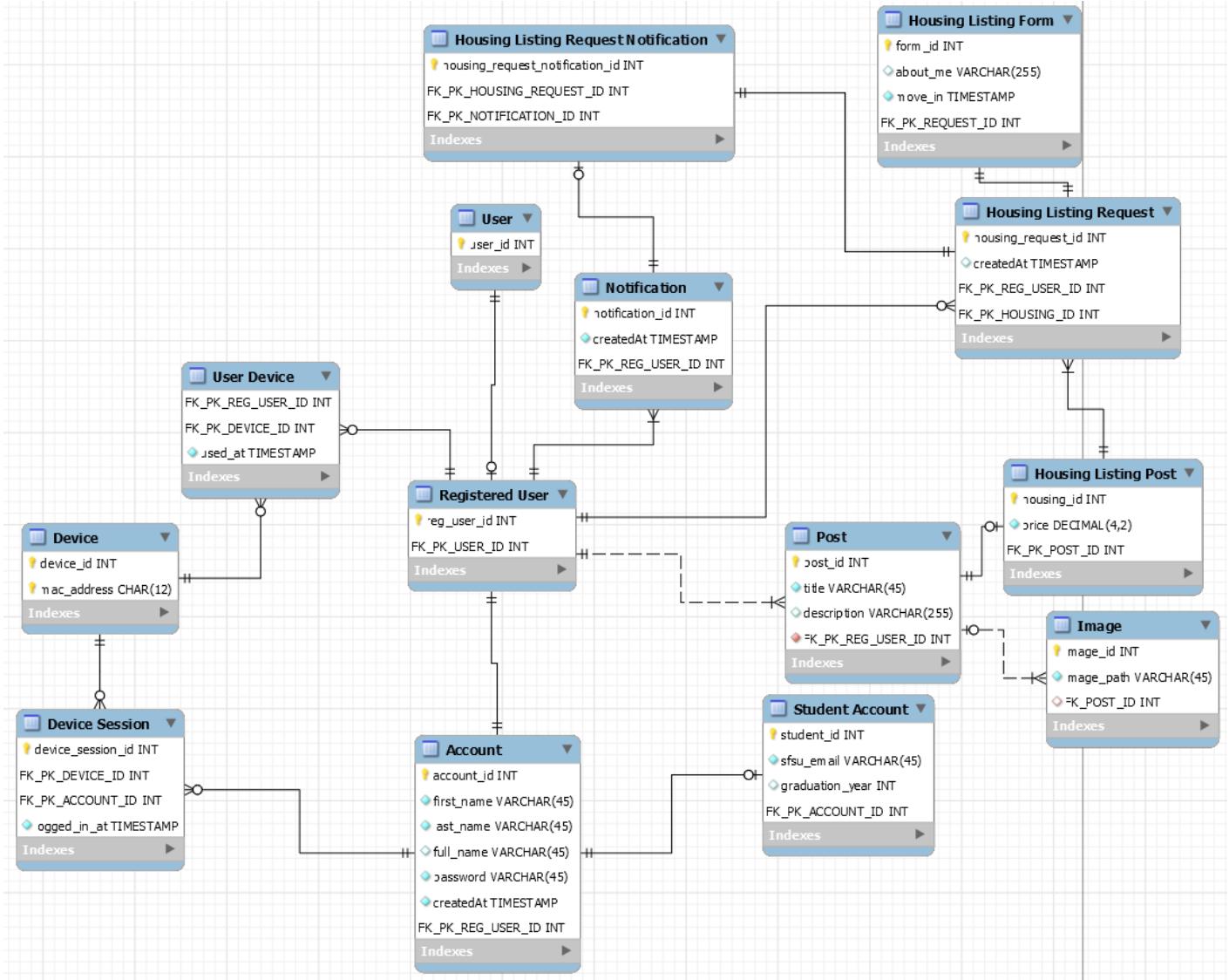
Shows login portion of ERD and relationships between Registered User, Housing Listing Request, and Housing Listing Post



Shows notification portion of ERD and relationships between notifications, Housing Listing Request, Housing Listing Form, and Housing Listing Post.



Database Model/Enhanced Entity Relationship (EER)



Database Management System (DBMS)

The DBMS that we are using to create the database is MySQL because we are creating a web application that mainly only deals in data transactions and simple queries. Another option we were considering was PostgreSQL. However, we concluded that MySQL was suitable for our application in the long run, and PostgreSQL, even though it supports more complex features, does not provide any significant advantages or extra utility for our web application needs.

Media Storage

Images will be kept in the file system in the EC2 instance because storing the images as blobs in our database does not provide any significant performance advantages. On top of this, storing the images in the EC2 instance keeps our database size small.

Search/Filter Architecture and Implementation

For our basic search functionality, we will be taking advantage of MySQL's InnoDB engine Full Text Search feature. In our database tables such as items, housing, restaurants, etc., we will create full text indexes based off of the columns in the table we want searched. Full text search allows us to search multiple columns in tables for relevant text strings due to word stemming which means our search functionality will be flexible in the results it finds. Also, we can have MySQL automatically sort search results by most relevant text-wise, which SQL's LIKE operator does not do. On top of this, based on our research, Full Text Search, on average, is faster than using SQL's LIKE operator. Therefore, our main search implementation method will be using MySQL's InnoDB engine Full Text Search. However, we will still look into SQL's LIKE operator and compare the performance and results of individual queries. Therefore, some sections of our search implementation will be decided on a case to case basis. We will create full text indexes for the combinations of table columns that we need. These full text indexes also have the added benefit of taking care of most of our filtering.

High Level APIs and Main Algorithms

AUTHENTICATION

We are creating our own authentication API instead of using similar APIs such as Django's built-in authentication API. We chose to create our own API because from our research, Django's authentication API only supports one type of user with one type of account. However, for our application, we have many different types of accounts such as Student, Admin, and Super User. This means Django's API will not be compatible with our application when it comes to dealing with user permissions, creating/registering different types of accounts, and logging into different types of accounts. By creating our own authentication API, we will be able to support multiple types of accounts and deal with user permissions as well. Our authentication API will handle registering new users by checking if users have a SFSU email. Login will be done by searching for the email that the user enters and comparing the password that the user enters with the encrypted password stored in our database.

- 1) Password Encryption:** We will be using the popular password encryption algorithm, BCrypt to encrypt passwords in our database. We chose to use BCrypt instead of creating our own password encryption algorithm because the encryption is much more secure than what we can provide in our current timeframe. On top of this, the development costs of creating a new password encryption algorithm from scratch would outweigh the very minimal benefits we might get from creating our own algorithm. BCrypt's algorithm works by adding randomized data to the end of a given password, and encrypting the combined result many times.
- 2) Email Verification:** We will be creating our own email verification API because all of Django's premade email verification modules use Django's authentication user model. Because we will be creating our own authentication API since Django's authentication user model is not compatible with our application, we must also create our own email verification API. However, we will be using Django's built-in email backend as the foundation for our email verification API. When a user creates an account, the account will be created in our database, but this account will have an extra piece of information indicating whether or not the user has verified his/her account and a unique code. Also, a verification email will be sent to the user's provided email. This email will contain a link with the unique code appended to the end of the link and when the user clicks the link, his/her account will automatically be verified in our database.
- 3) Email Verification Link Token Generator:** We will be using Python's built-in secrets library to create URL safe tokens that will be appended to the unique link found in the verification email. Python's secrets URL safe tokens algorithm works by taking random

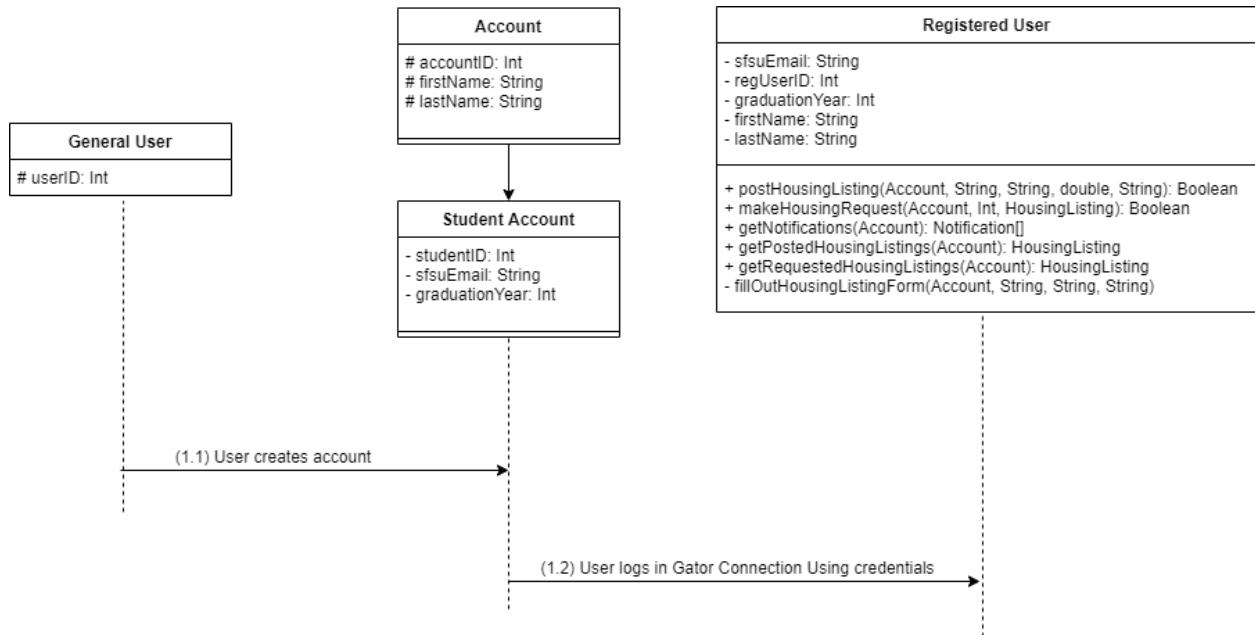
data and encoding it. Then the algorithm returns this encoded random data as the unique token.

NOTIFICATIONS

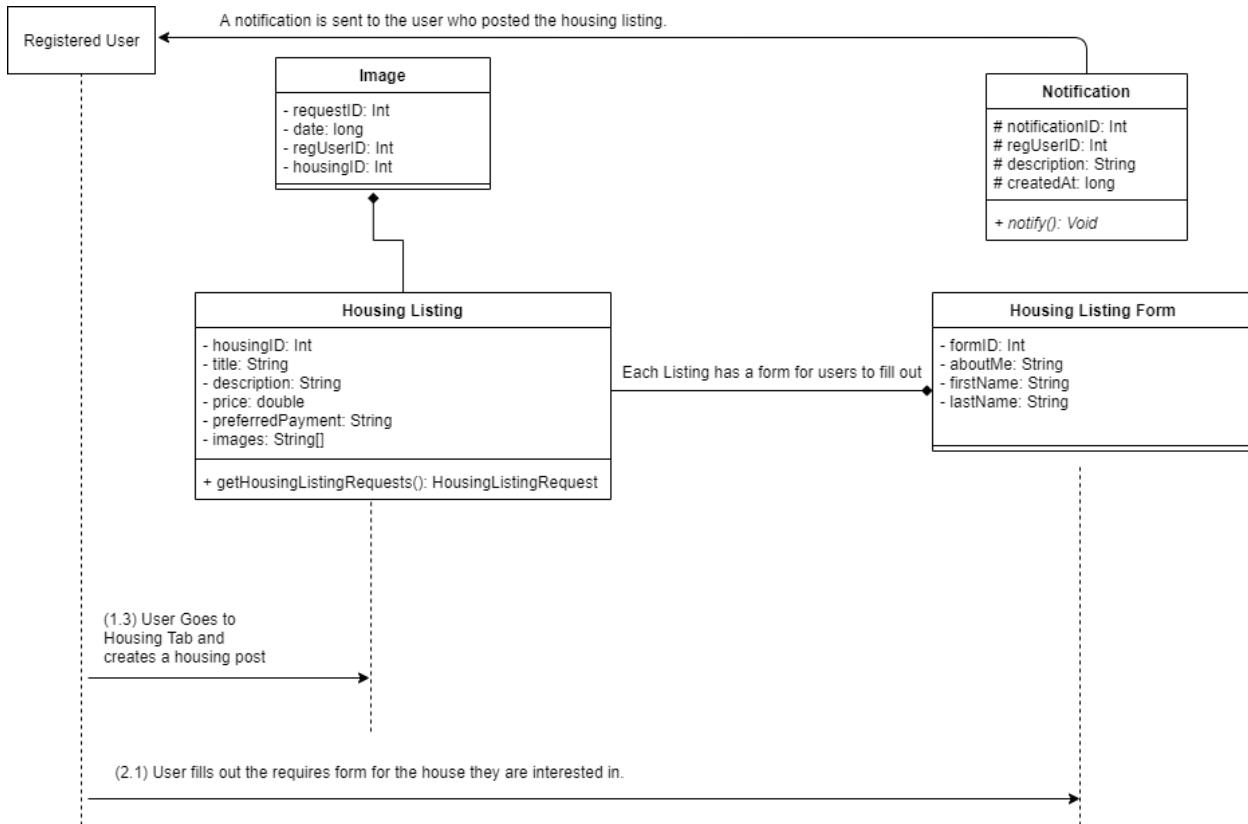
We will be creating our own notification API because we could not find an equivalent Django notification module. However, similar to our verification email API, we will be using Django's email backend as the foundation of our notification API. Our API will be designed so that whenever a user does an action that should notify others such as closing a posting or sending a buy request, we will create notifications for each user that must be notified in the database. Also, we will send notification emails of the event that occurred that created the notification. Because this email notification API is similar to the email verification API we will be creating, creating the notification API will not affect the development cost of our product greatly.

High Level UML Diagrams

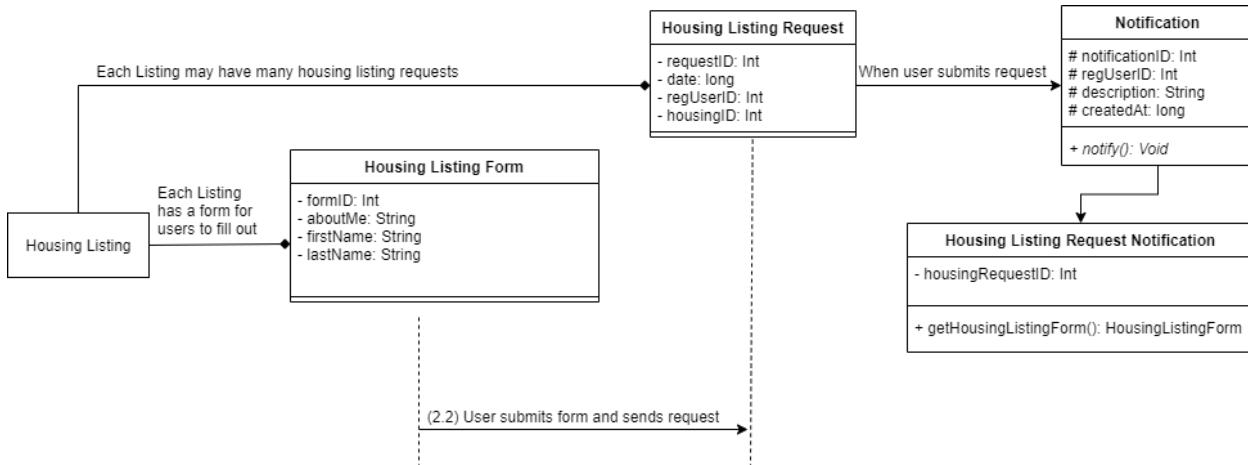
Association and Relationships between General User, Registered User, Account, and Student Account.



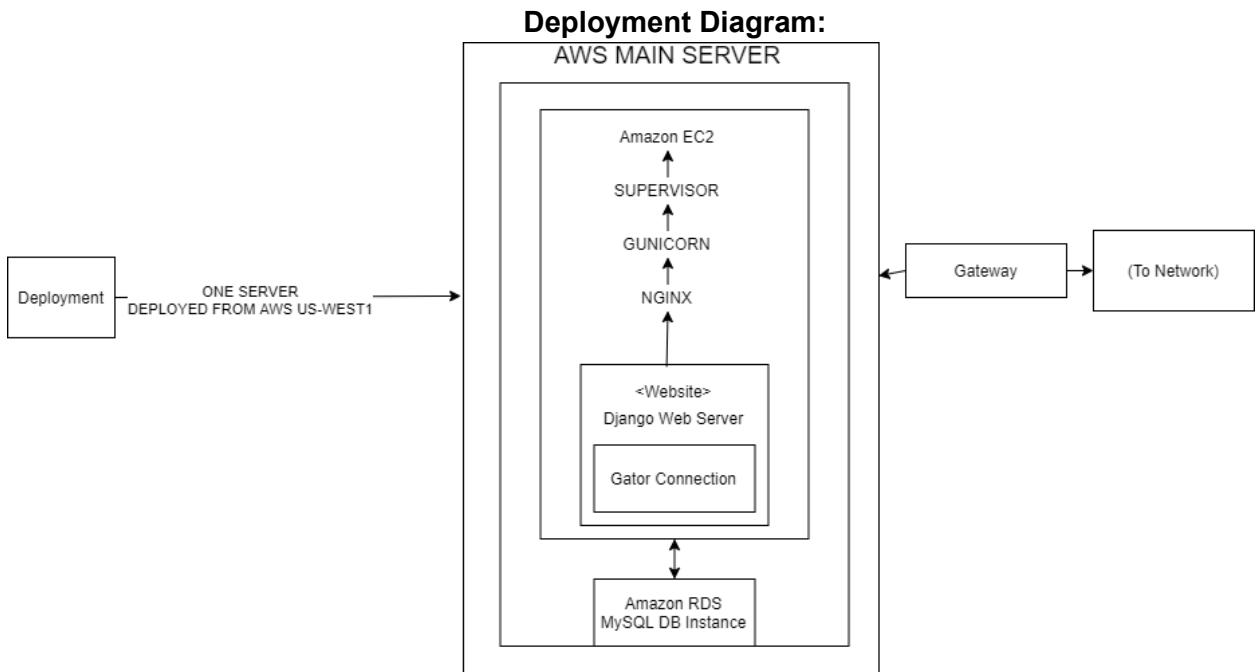
Associations and Relationships between Registered User, which was already defined above, Housing Listing, Image, Housing Listing Form, and Notification.



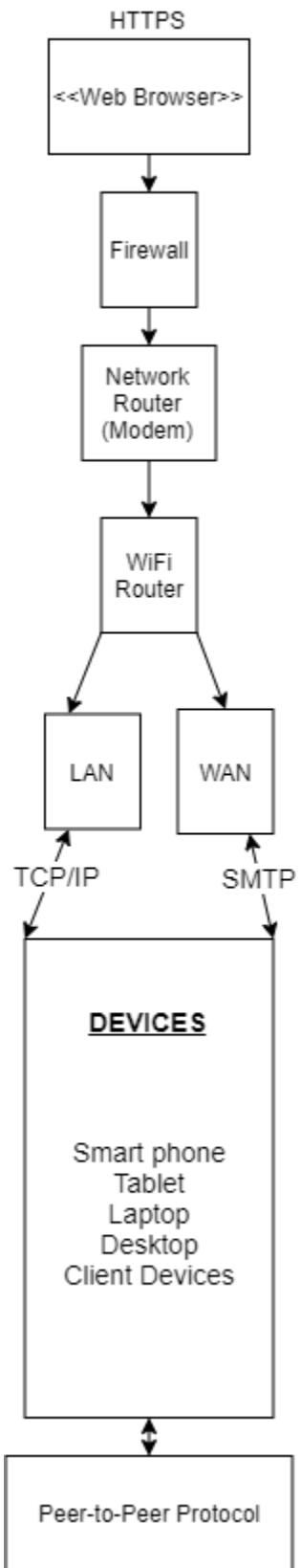
Associations and relationships between Housing Listing, which was already defined above, Housing Listing Form, Housing Listing Request, Notification, and Housing Listing Request Notification.



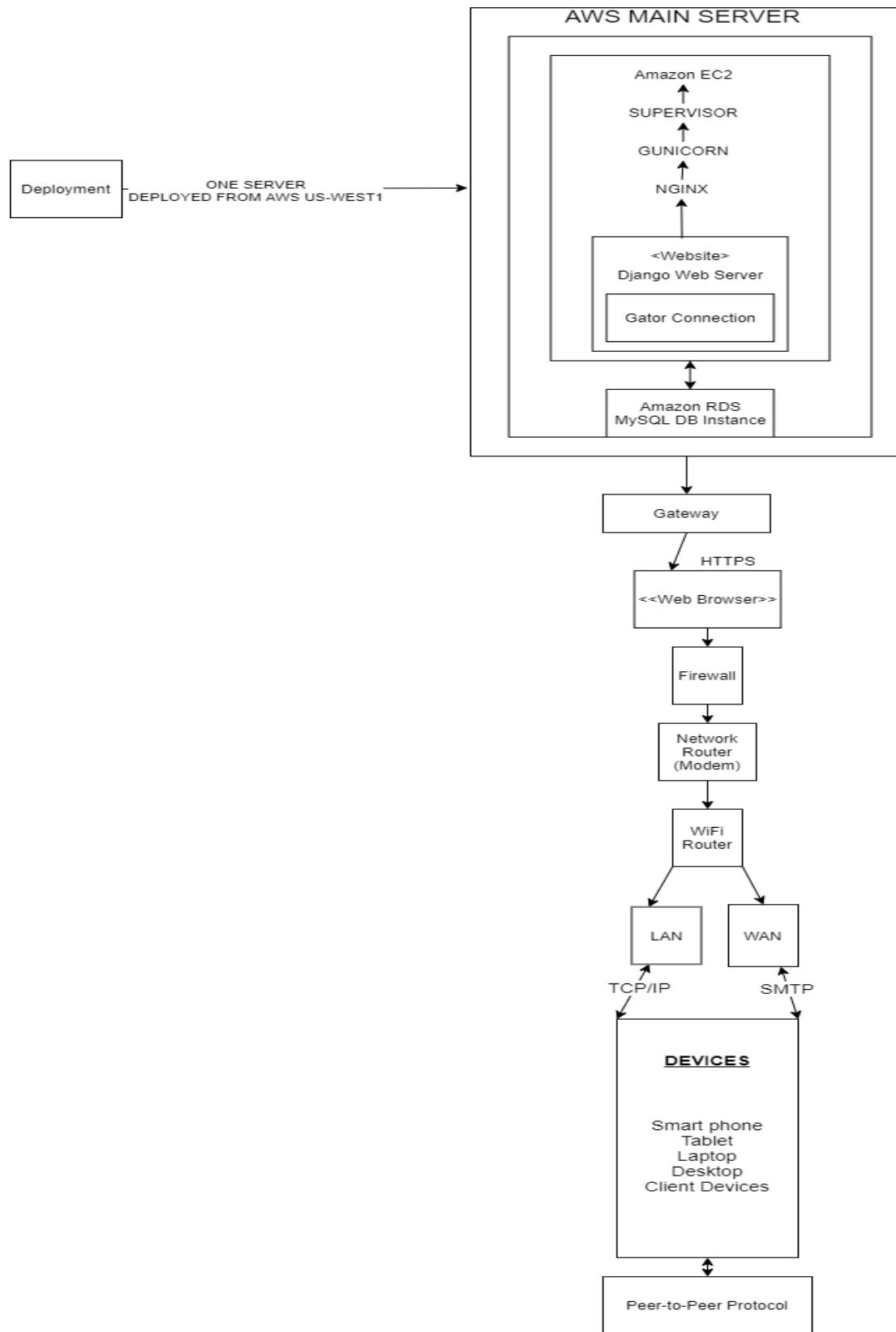
High Level Application Network and Deployment Diagrams



Application Network Diagram:



Application Network Diagram with Application Network Diagram:



Project Risks

Schedule risk

Due to everyone's schedules, it's hard to get a meeting time for everyone to be present. However, with having a trello account set up for project management for all team members, it has been easier to convey task distribution. Furthermore, we have been able to meet by separating our meetings into front-end and back-end meetings, and recapping the team members who can not attend.

Skill risk

Our team has development experience, however, none of us had experience designing a website. Furthermore, while some of us were familiar with the django web framework, it was unfamiliar to most of the team, with most of us having to learn it on the fly. In addition, Python is also unfamiliar to some of the team, so some of us are still in the process of understanding it. Lastly, most of us have never done front end design at this large of a scale, so everyone on the team is learning it as we go along as well.

Technical risk

One technical risk was associated with our github repo. What was at risk was our git branches, specifically with push/pull and merge conflicts. How we keep track of merge conflicts is with the multiple branches we have for different tasks and having a Github master who handles the merge conflicts. Another technical risk was the issue of our MySQL database because we wanted the team to connect to the database locally so we would all work with the same data. Connecting to the database was challenging for the team, but we had managed to all do it. However, one thing was that we had to do was make our database public, which could expose information. However, we fixed this problem by creating security rules that allowed only the team's and EC2 instance's IP addresses to access the database. One more technical risk was how everyone has different operating systems, such as Windows, Mac, etc. In the end, we decided to use Ubuntu for our project, which is generally easier for people with Apple products. For the team members who had a Windows Operating system, they had to learn Linux commands to get used to the environment for our project.

Teamwork risk

One major teamwork risk is having good communication between the backend and frontend teams due to the team's scheduling issues. However, with our project management being handled by Trello and the team lead posting updates on the discord channel and attending every meeting to be the middleman between the front and backend, everyone is up to date on what needs to be done and what each team is working on.

Legal/Content risk

Our legal risks may vary depending on the end product of the deployment. On our server, we will have control and ownership of images on the site. With that, one risk associated with us is that we may encounter people who upload images that do not belong to them. Another risk is that we will have external links to outside resources for services we do not provide. This means that if anything happens to the user after he/she clicks on an external link, we might be held responsible. To resolve these risks, when a user registers their account, they must agree to a terms and conditions form. These terms and conditions would ensure that everything they upload belongs to them and that the user understands that we are not responsible for anything that occurs after the user exits our site via any external links. We will also warn users when they click on an external link that they are leaving our site.

Project Management

In our first couple of team meetings for M2, we addressed the sections that we believed had to be done with everyone together such as setting the priorities of our functional requirements, enhancing our data definitions, and examples of how we wanted to format our UI mockups and storyboards. Afterwards, our next team meeting, we set up Trello as our main project management tool with tasks split up into frontend tasks, backend tasks, team tasks, and completed tasks. By keeping track with Trello, backend and frontend team members would create rough drafts of the sections assigned to them. Once those team members completed the rough drafts and notified the team on Trello and Discord, either the frontend lead or backend lead would give feedback depending on which team was in charge of which section and if needed, set up meetings with the team members that created the rough drafts and edited the sections together. Otherwise, the team members would incorporate the feedback and continue the cycle of getting feedback from a frontend lead or backend lead. Once the frontend lead or backend lead deemed the section was completed, the team lead did the final quality check. The team lead also made sure to check in and help with all sections that team members were working on and gave intermittent feedback as well. For future tasks, we will continue to use Trello and Discord to keep track of tasks and our progress. Most importantly, we will continue to continue the cycle we were practicing of creating rough drafts, receiving feedback from a team lead member, and incorporating the feedback.

Gator Connection Tasks

Board | Team visible | LC AR AY AT BK +2 | Invite

Milestone 2 Tasks

- Create Database model
- Finish m2 doc (Mar 30)
- Search method, filter student's email. (AR, BK)
- + Add another card

Team Lead Tasks

- + Add a card

Front-end Tasks

- Shop-Items(development branch) (Apr 6)
- Home Page updated (Mar 25) (LC)
- Create less whitespace bw picture and welcome to gator connection string
- Add carasoul under mission statement
- Have searching underneath the carasoul
- Updated Home Page (Mar 26) (LC)
- + Add another card

Back-end Tasks

- Send email to verification (AY)
- Display stored images on items page. (BK)
- Create tables for signup, login, material post. (Mar 14) (BK)
- Create DB models on MySQL workbench. (AR)
- Send email to reset password (AR)
- Verify only SFSU emails are registered. (AR)
- Start on MySQL tables (Due TBA)
- Function if user tries to find all.
- + Add another card

Everyone Tasks

- + Add a card

COMPLETED TASKS

- Create new Django app folder: gator_connection for new path. (Mar 6) (AR, BK)
- Create ERD by Monday (Mar 8) (AR, BK)
- Create html for login for signup (Mar 16)
- NavBar (Mar 18) (1 comment)
- Home Page (Mar 18)
- EnableHttps for main instance
- New version of ERD (Mar 13) (BK)
- Work on search bar function on home page. M2 branch (Mar 21) (AY)
- API Calls
- + Add another card

List of Contributions

1) Data Definitions V2:

During a team meeting, the team looked at the data definitions we had, and decided on which terms we needed to expand upon. After deciding, Alec added the extra terms to explain certain subsections, with the team revising what should be said to explain them.

2) Functional Requirements V2:

During a team meeting, the team looked at the functional requirements we had. During this time, with suggestions from other members, namely Angelo and Benjamin, we were able group up our functional requirements into the specified three priority groups, with Priority 1 being the most important, to Priority 3 being opportunistic.

3) UI Mockups and StoryBoards:

During a team meeting, Alec tasked the group to do a rough draft of a UI mockup of the Use Case they had made. After receiving the rough draft, Alec looked over the styles everyone had submitted and decided to have the Front End members, namely Carmen and Bikram to revise everyone's rough draft into a final draft version.

4) High level database architecture and organization:

During a team meeting, Angelo started the rough draft of the entity relationship diagram (ERD) for our special feature. After feedback from Jiaxin and Benjamin, the ERD was revised to be more accurate of the special feature we had chosen for our project and to follow more closely along the lines of the business rules. From the business rules, Angelo was able to extract the entities and describe them at a deeper level. On the database model, Benjamin designed the database model that we would be using for the special feature based off of the ERD. On this part, we also decided that we would use MySQL for our Database Management System(DBMS) and our images would be kept in the file systems of our EC2 instance, with Benjamin filling this part out. Furthermore, on another team meeting, Benjamin showed the team the Full Text Search feature, which we would decide to use for our search/filter architecture.

5) High Level APIs and Main Algorithms:

Alec listed down certain API's that the team was going to eventually use for the project, such as POST calls for logging in and registering, GET calls for getting user information, etc. After feedback from the team, the API's were more fleshed out to describe the functionality.

6) High Level UML Diagrams:

Benjamin was tasked to complete the UML diagram for the special feature. After a meeting and some feedback for the first version from Angelo, Benjamin was able to revise the UML diagram to fit the standards of our special feature.

7) High Level Application Network and Deployment Diagrams:

Alec started a rough draft of the Application Network diagram. After a class lecture, the team, most notably Benjamin, helped revise the Application Network diagram to portray more info of what the professor had explained about the diagram. After clarification from the professor, Alec found out that the High Level Application Network and Deployment Diagrams could be combined into one, and revised it as such.

8) Identify actual key risks for your project at this time:

During a team meeting, the team helped suggest things that were considered as risks for the project. After jotting them down, Alec organized them into 5 sections, which were schedule, skill, technical, teamwork, and legal/content risks.

9) Project management:

During a team meeting, Angelo set up Trello and invited everyone on the team so we could start assigning tasks. On Trello, Alec delegated tasks to the team, such as having UI mockups done by a certain date.

10) Detailed list of contributions:

Alec filled out this part. After assigning the tasks to everyone, Alec looked over what everyone did to accurately fill out this section.

11) Milestone2 Editor:

After volunteering to edit the document, Lakshita organized the Milestone2 document to make it ready to be turned in.

Vertical Prototype Contributions

Backend:

For the backend portion of our vertical prototype, Angelo helped set up basic API features, such as registering and logging in for users. After a revision from Benjamin for the login portion, the registering and logging in features was complete. After the team learned how to use the login feature, which would allow us to restrict certain actions from someone if they were not logged in, Jiaxin was tasked to start implementing the search function. With that, Angelo expanded upon the search feature which would then allow users to search for a user based by their email. Alec then helped on the searching feature, so users would then be able to filter by sfsu email, and by the first or last name associated with that email. Benjamin then improved upon Alec's filtering feature by having the feature use Full Text Search to improve upon the searching feature. After improving upon the filtering feature, Benjamin then reorganized our backend code structure to make everything easier to find. Angelo then set up the DB table "student" on MySQL workbench. When a user would register, their info would be stored there. Furthermore, on the registering and logging in functions, Alec added error handling to make sure that the program would continue in case of an error.

Frontend:

The front end team was in charge of designing the vertical requirement home page. Lakshita, Carmen, and Bikram added features such as having a mission statement at the top of the screen, and a carousel slide at the bottom to show SFSU pictures. Other features implemented were having certain features only appear if a user is logged in, having messages appear to notify a user that once they have completed a certain action, such as logging in or registering. Furthermore, these messages have replies for actions that fail or succeed. In the case of an action succeeding, they will be greeted with a green box saying something like "Thank you for registering". On the flip side, if they fail, they will be greeted with something like "Please try again". Another feature added was structuring how the data coming from the database would look like on the page. On the navbar features, Alec designed the rough prototype, then had Bikram implement features on the navbar such as the navbar would highlight a specific section when it was on that page. Furthermore, Carmen designed our logo which would show on the top left of the page. On the design of the login and registering boxes, Alec designed the rough prototype for them, and after feedback from the team, he was able to revise it to make it look much cleaner than it was before.

Milestone 3 V2

Software Engineering CSC 648/848 Spring 2021

Gator Connection

A One Stop Website for SF State Gators

Team 05

Team Lead/Github Master: Alec Stephen Tenefrancia alectene@mail.sfsu.edu

Frontend Lead/Document Master: Lakshita Chugh

Backend Lead: Angelo Gloria Reyes

Frontend member: Bikram Tamang

Backend member: Benjamin Patrick Kao

Frontend member: Carmen Denisse Paisano

Backend member: Jiaxin Yu

Milestone 3

Milestone/Version	Date
M3V2	05/18/21
M3V1	04/22/21
M2V2	04/24/21
M2V1	04/01/21
M1V2	03/09/21
M1V1	03/04/21

Table of Contents

Table of Contents	104
Main Data Items & Entities V3	107
Functional Requirements	110
Priority 1 Requirements:	110
Priority 2 Requirements:	113
Priority 3 Requirements:	114
WireFrames	116
High Level Database Architecture and Organization V2	179
High Level Diagrams V2	180
High Level UML Diagram	180
High Level Application Network and Deployment Diagrams	183
List of Contributions	185

Main Data Items & Entities V3

1. Unregistered User/ Guest

These are the people who aren't registered with the school system yet. They will be treated as guests, and would not be eligible for any of our student/staff services. This is an administrative task.

2. Registered User

A person who is registered with us using their email would have an account with us. They will follow the Create Account path. Their data is already registered with us, they will have the authority to create their password here.

3. Account

An account contains identification data about the registered user that created the account. The data identifies what type of registered user the user is, as well as helps authentication registered users for restricted features such as posting announcements.

- a) **Student:** A student account gives registered users the ability to search housing listings, post restaurant reviews, and make purchase requests for posted items and housing listings.
- b) **Admin:** An admin account gives registered users the power to post public announcements. Admin accounts are split into more specific roles as to more easily group announcements together when filtering/searching.
 - i) **Sports/Athletics:** A registered user who has this type of account is in charge of a specific sport that he/she registered with.
 - ii) **Organization/Club:** A registered user who has this type of account is in charge of a specific organization that he/she registered with.
 - iii) **School Department:** A registered user who has this type of account is a part of SFSU staff.
- c) **Super User:** A super user account gives registered users the power to approve restaurant additions, administrative account creation requests, and registered user account upgrade requests.

4. Notifications

Receiving notifications about housing, submissions, etc.

- a) **Housing Notifications:** Registered users will receive email notifications about housing such as if they receive a request for a listing.

- b) Submission Notifications:** Registered users will receive email notifications about submissions such as if their account requests and restaurant requests have been approved or denied

5. Rate

This gives users the ability to rate various student organizations, restaurants. It'll prove to be a useful tool for fellow users.

- a) Restaurants Rating:** Ratings that users can give restaurants from 1-10. The average of all ratings will be shown on the restaurants.

6. Review

Posts shown under restaurants made by registered users and above. Unregistered users can view these but they cannot rate or post a review.

7. Email

Being able to send emails to all of the users that are registered. This would be students and administration.

- a) Housing emails:** A User who posted a listing of a house for sale will receive emails from people who are interested to purchase the listing.
- b) Registration Verification email:** A User who creates an account will receive an email to their SFSU email saying that their account has been created
- c) Reset password email:** An email which a user can request to reset their password

8. Announcements

Section designated to announcements that different departments make, along with the health centers, athletics, and organizations/clubs. These announcements are directly related to SFSU campus and only administrative users can post announcements.

- a) Athletic Announcements:** Announcements made from athletic directors
- b) Organization Announcements:** Announcements made from organizations
- c) School Announcements:** Announcements made from the school

9. Restaurants

Section dedicated to restaurants around campus. Users can view, rate, and write reviews for that specific restaurant. Unregistered users can view these but they cannot rate or review.

10. Ecommerce/Listing

The main section where users can view the selected items for sale or housing available and make purchase requests for them. Registered users can post things for sale. Unregistered guests cannot view these items.

a) Items:

The item(s) listed for sale in the Ecommerce/Listing by a registered seller or for purchase by a registered buyer. Users can message the seller if they are interested in the item.

b) Housing:

The listing(s) of housing listed for sale in the Housing section. Users interested in a listing can fill out a form which includes their name, school year, SFSU email, and a little information about themselves. After filling out the form, the system will send out an email to the person who posted the listing, and it will be up to them if they wish to respond back.

Functional Requirements

Priority 1 Requirements:

1. Guest User
 - 1.1. A guest user shall be able to search for restaurants by title.
 - 1.2. A guest user shall be able to sort restaurant listings by rating.
 - 1.3. A guest user shall be able to navigate through announcements.
 - 1.4. A guest user shall be able to create an account using his/her unique SFSU email account for a student account or a unique email account for an admin or super user account.
 - 1.5. A guest user shall be able to navigate to a map of SFSU.
 - 1.6. A guest user shall be able to search for items on sale by preferred payment.
 - 1.7. A guest user shall be able to search for items on sale by price.
 - 1.8. A guest user shall be able to search for items on sale by title.
2. Registered User
 - 2.1. A registered user shall have all of the same permissions as guest users.
 - 2.2. A registered user shall be able to log in to his/her account using his/her unique SFSU email.
 - 2.3. A registered user shall be able to log in to his/her account using many devices.
 - 2.4. A registered user shall be able to log out of the website.
 - 2.5. A registered user shall be able to stay logged in if they have not logged out.
 - 2.6. A registered user shall be able to post reviews of restaurants/food.
 - 2.7. A registered user shall be able to make purchase requests for items.
 - 2.8. A registered user shall be able to search for items by preferred payment, price, etc.
 - 2.9. A registered user shall be able to see and search housing listings.
 - 2.10. A registered user shall be able to make a request for a housing listing.
 - 2.11. A registered user shall be able to edit a housing listing title he/she posted.
 - 2.12. A registered user shall be able to edit a housing listing description he/she posted.
 - 2.13. A registered user shall be able to edit a housing listing price he/she posted.
 - 2.14. A registered user shall be able to change a housing listing image he/she posted.
 - 2.15. A registered user shall be able to close a housing listing that he/she posted.
 - 2.16. A registered user shall be able to post many pending items for sale with pictures.
 - 2.17. A registered user shall be able to edit a post about an item for sale that he/she posted.
 - 2.18. A registered user shall be able to take down an item that he/she put up for sale.

- 2.19. A registered user shall be able to create a restaurant request to add a restaurant to the restaurants listing by providing the name of the restaurant and location.
 - 2.20. A registered user shall be able to sort and search for restaurants by ratings/reviews.
 - 2.21. A registered user shall be notified if their restaurant location has been approved or denied, with a message explaining why.
 - 2.22. A registered user shall receive a notification about a house listing being closed if he/she had made a request to the house lister.
 - 2.23. An upgraded registered user shall receive a notification when their administration privileges are taken away and why.
 - 2.24. A registered user shall have a unique registered id.
 - 2.25. A registered user shall be able to fill out a form for a housing request.
 - 2.26. A registered user shall receive a notification if their housing post has been approved by email.
 - 2.27. A registered user shall receive a notification if their housing post has been denied by email.
 - 2.28. A registered user shall receive a notification if their post has been approved by email.
 - 2.29. A registered user shall receive a notification if their post has been denied by email.
 - 2.30. A registered user shall be able to request to upgrade his/her account to an administrative account with a valid organization, role, and organization by email.
 - 2.31. A registered user shall receive a notification about purchase requests made on his/her account by email.
 - 2.32. A registered user shall receive a notification about purchase requests made on items he/she posted for sale by email.
 - 2.33. A registered user shall receive a notification about a house listing request for a house listing that he/she posted that includes information about the person who requested, such as his/her name and email address.
 - 2.34. A registered user who requested an account upgrade shall receive a notification when the request has been approved and the date of when their administration privileges will be taken away by email.
 - 2.35. A registered user who requested an account upgrade shall receive a notification when the request has been denied by email.
3. Admin User
 - 3.1. An administrative user shall have all of the same permissions as registered users.
 - 3.2. An administrative user shall be able to post announcements.
 - 3.3. An administrative user shall be able to remove an announcement he/she had posted.
 - 3.4. An administrative user shall have a unique admin id.

4. Super User
 - 4.1. A super user shall be able to log into the website.
 - 4.2. A super user shall be able to log out of the website.
 - 4.3. A super user shall have all of the same permissions as administrative users.
 - 4.4. A super user shall be able to approve/deny and close account upgrade requests,
 - 4.5. A super user shall be able to approve/deny and close administrative account creation requests.
5. Sale Item
 - 5.1. A sale item shall be posted by a registered user.
 - 5.2. A sale item shall have a unique item id.
 - 5.3. A sale item shall have a title for the item.
 - 5.4. A sale item shall have a message describing the item.
 - 5.5. A sale item shall have one picture attached to it.
 - 5.6. A sale item shall be purchased by a registered user.
 - 5.7. A sale item shall be taken down by the registered user that posted it for sale.
 - 5.8. A sale item shall have a price.
 - 5.9. When a sale item is taken down, all registered users who made a request for the item shall be notified of the closure by email.
6. Housing Listing
 - 6.1. A housing listing shall be posted by a registered user.
 - 6.2. A housing listing shall have a unique housing id.
 - 6.3. A housing listing shall have a title.
 - 6.4. A housing listing shall have a message describing the item.
 - 6.5. A housing listing shall have a price.
 - 6.6. A housing listing shall have at least 1 picture attached to it.
 - 6.7. A housing listing shall be taken down by the registered user that posted it.
 - 6.8. A housing listing shall be requested by many registered users.
 - 6.9. A housing listing shall have a form to fill out for interested users.
 - 6.10. A housing listing shall have a situation(available/close).
 - 6.11. When a housing listing is taken down, all registered users who made a request shall be notified of the closure by email.
7. Housing Listing Form
 - 7.1. A housing listing form shall be filled out by a registered user.
 - 7.2. A housing listing form shall fill out their full name.
 - 7.3. A housing listing form shall fill out their school year.
 - 7.4. A housing listing form shall fill out their school email.
 - 7.5. A housing listing form shall fill out their phone number.
 - 7.6. A housing listing form shall have a section dedicated to “an about” me.
 - 7.7. A housing listing form shall have an expected day they want to move in.

8. Restaurant
 - 8.1. A restaurant shall have a name.
 - 8.2. A restaurant shall have a location.
 - 8.3. A restaurant shall have a unique restaurant id.
 - 8.4. A restaurant shall have a rating.
9. Restaurant Request
 - 9.1. A restaurant request shall be created by one and only one registered user and above.
 - 9.2. A restaurant request shall have the name of the restaurant.
 - 9.3. A restaurant request shall have the location of the restaurant.
 - 9.4. A restaurant request shall be confirmed/denied and closed by one and only one super user.
 - 9.5. When a restaurant request is confirmed/denied the registered user who made the request will be notified by email.
10. Organization
 - 10.1. An organization shall be able to post announcements.
11. Announcement
 - 11.1. An announcement shall be posted by one and only one administrative user or organization.
 - 11.2. An announcement shall be able to be viewed by one or many users.
 - 11.3. An announcement shall be able to be taken down by the one who posted it.

Priority 2 Requirements:

12. Guest User
 - 12.1. A guest user shall be able to make item purchase requests after filling out an identification form and completing a Captcha.
13. Registered User
 - 13.1. A registered user shall be able to rate organizations.
 - 13.2. A registered user shall be able to change their rating of the organization.
 - 13.3. A registered user shall be able to post reviews under event announcements.
 - 13.4. A registered user shall be able to change any reviews they have under event announcements.
 - 13.5. A registered user shall be able to rate a housing listing.
 - 13.6. A registered user shall be able to rate any of the item listings on sale.
 - 13.7. A registered user shall be able to change their rating of the item.
 - 13.8. A registered user shall be able to report housing listings and provide reasoning for the report.
 - 13.9. A registered user shall be able to report restaurant reviews.
 - 13.10. A registered user shall be able to report announcements
14. Admin User

- 14.1. An admin user shall be able to temporarily take down posts that he/she finds false and give a reason why.
15. Super User
 - 15.1. A super user shall receive notifications of posts that admin users take down.
16. Sale Item
 - 16.1. A sale item shall be able to be taken down by a super user.
17. Housing Listing
 - 17.1. A housing listing shall be able to be taken down by a super user.
18. Housing Listing Form
 - 18.1. A housing listing form shall be able to be saved by the creator in their notifications page.
19. Restaurant
 - 19.1. A restaurant shall be able to have their rating changed.
 - 19.2. A restaurant shall have many reviews written by registered users.
20. Restaurant Request
 - 20.1. A restaurant request shall also allow users to request to update incorrect information about a restaurant.
21. Organization
 - 21.1. An organization shall be able to be rated by many registered users.
22. Announcement
 - 22.1. An announcement shall be able to be taken down by a super user.
 - 22.2. An announcement shall be able to be rated.

Priority 3 Requirements:

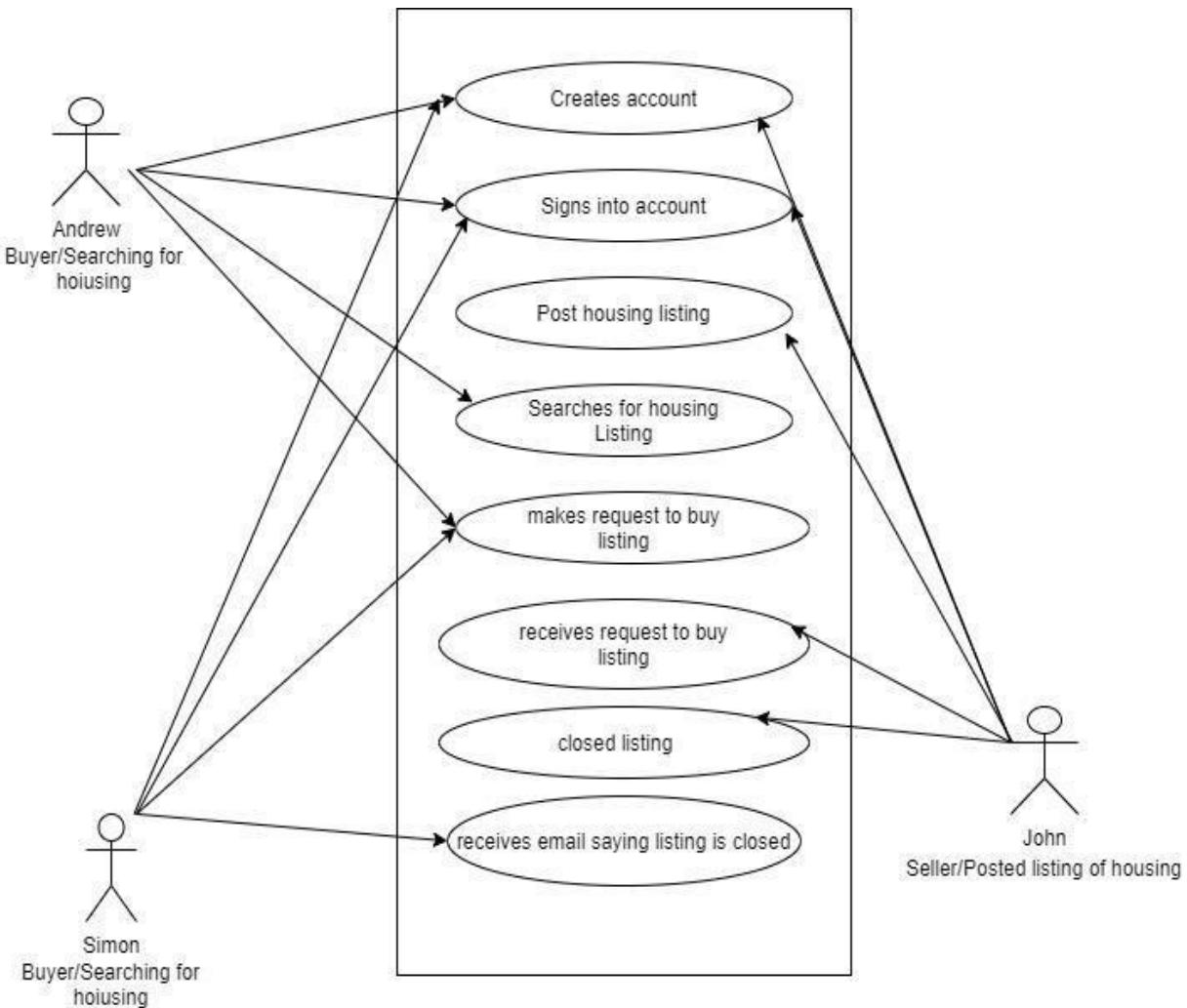
23. Guest User
 - 23.1. A guest user shall be able to make housing purchase requests after filling out an identification form and completing a Captcha.
24. Registered User
 - 24.1. A registered user shall be able to fill out a potential roommate form when posting a housing listing.
 - 24.2. A registered user shall be able to fill out a potential roommate form when searching through housing listing posts.
 - 24.3. A registered user shall be able to receive suggested housing listings where both the poster and searcher are ideal roommates.
 - 24.4. A registered user shall be able to subscribe to organizations and departments.
 - 24.5. A registered user shall be able to report item and housing listings for false/fake information.
 - 24.6. A registered user shall be able to be banned/restricted to only guest user privileges if he/she has a certain number of reports on posts.
25. Admin User

- 25.1. An admin user shall be able to be banned/restricted to only guest user privileges if he/she has a certain number of reports on announcements.
26. Super User
 - 26.1. A super user shall receive notifications of taken down posts such as announcements, housing listings, and items.
 - 26.2. A super user shall receive notifications of banned users.
 - 26.3. A super user shall be able to unban banned users.
27. Sale Item
 - 27.1. A sale item shall be automatically hidden from users when a certain number of reports is reached.
28. Housing Listing
 - 28.1. A housing listing shall be automatically hidden from users when a certain number of reports is reached.
 - 28.2. A housing listing shall be shown on a virtual map.
 - 28.3. A housing listing shall show the distance from SF State.
 - 28.4. A housing listing shall show how long it takes for a person to get to SF State.
29. Housing Listing Form
 - 29.1. A housing listing form shall be able to be customized for a housing listing post by registered users.
30. Restaurant
 - 30.1. A restaurant shall be shown on a virtual map.
 - 30.2. A restaurant shall show the distance from SF State.
31. Restaurant Request
 - 31.1. A restaurant request shall be able to be deleted by the one requested it.
32. Organization
 - 32.1. An organization shall be able to send emails to subscribed registered users about announcements.
33. Announcement
 - 33.1. An announcement shall be able to be mass sent to registered users.
 - 33.2. An announcement shall be automatically hidden from users when a certain number of reports is reached.

WireFrames

Case 1: Freshman Student looking for housing at SFSU

Actors: Andrew(Buyer, Searching for housing), Simon (Buyer, Searching for housing), John(Seller, Posted listing of housing)



Action 1 and 2: Creating account/Signing into Account (Andrew, Simon, John)

Creating Account/Signing into Account

The screenshot shows the Gator Connection website with a navigation bar at the top. The 'Register' button in the top right is highlighted with a large arrow pointing to it from the left. A callout box on the right contains the following text:

Simon, Andrew, and John proceed to enter their first name, last name, SFSU email, graduation year, password into the respective boxes. They then click register to finish making their account. They then go to their email and verify their account to use our features

Register

Student (On student) Admin Super User

By registering, you agree to Gator Connection's Terms and conditions.

First Name
Last name
SFSU Email
Graduation Year
Password

Register

Already have an account? Click here to sign in

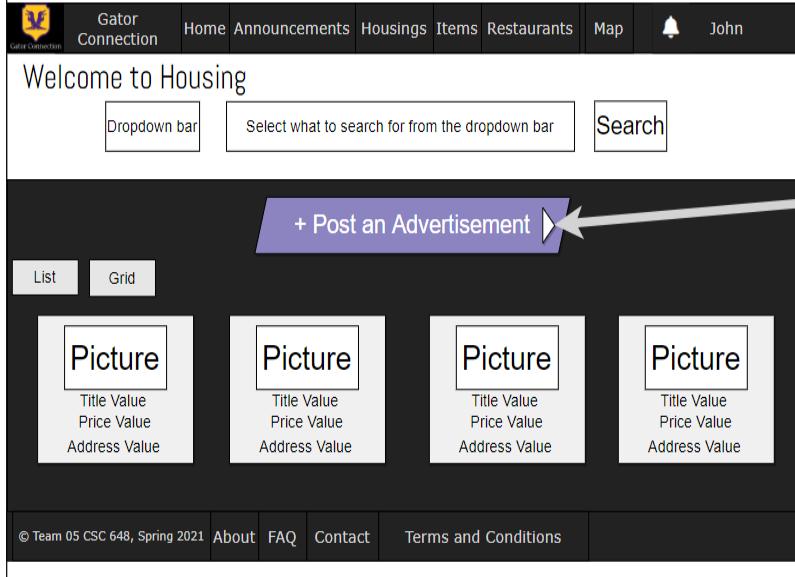
Close

Action 3: Post Housing Listing (Wireframe shown as John)

Sequence 1: On home page going to Housing

The wireframe shows the Gator Connection homepage. At the top, there is a navigation bar with links for Home, Announcements, **Housings**, Items, Restaurants, and Map. A user profile for "John" is also present. Below the navigation bar is the Gator Connection logo featuring a stylized gator head. A search bar with placeholder text "Search for items, housing, or restaurants" and a "Search" button are located below the logo. A banner below the search bar reads "A one Stop For all your Gator Needs." At the bottom of the page are four category buttons: HOUSING, ITEMS, RESTAURANTS, and ANNOUNCEMENTS. A large banner below these buttons contains a "Picture" on the left and a "Banner Description" on the right. A callout box with the text "Click on housing to go to housing" has arrows pointing from it to the **Housings** link in the navigation bar and to the **HOUSING** button at the bottom.

Sequence 2: On Housing Page



Click here to
bring up
Modal

Sequence 3: Inside Modal Box

Post a Listing X

Title: _____
Enter Title for Housing

Pricing: \$ _____
Ex: 350.00

Zip Code: _____

Number: _____

Street: _____

Description: _____
Enter Description of Housing here

Preferred method of Payment: Dropdown bar

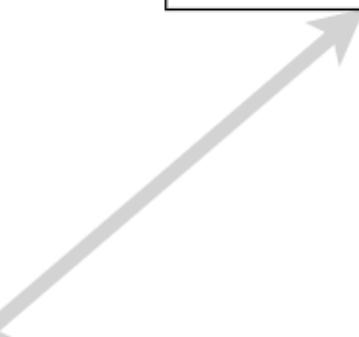
Pets Allowed: yes no

Upload Images: choose file no file chosen

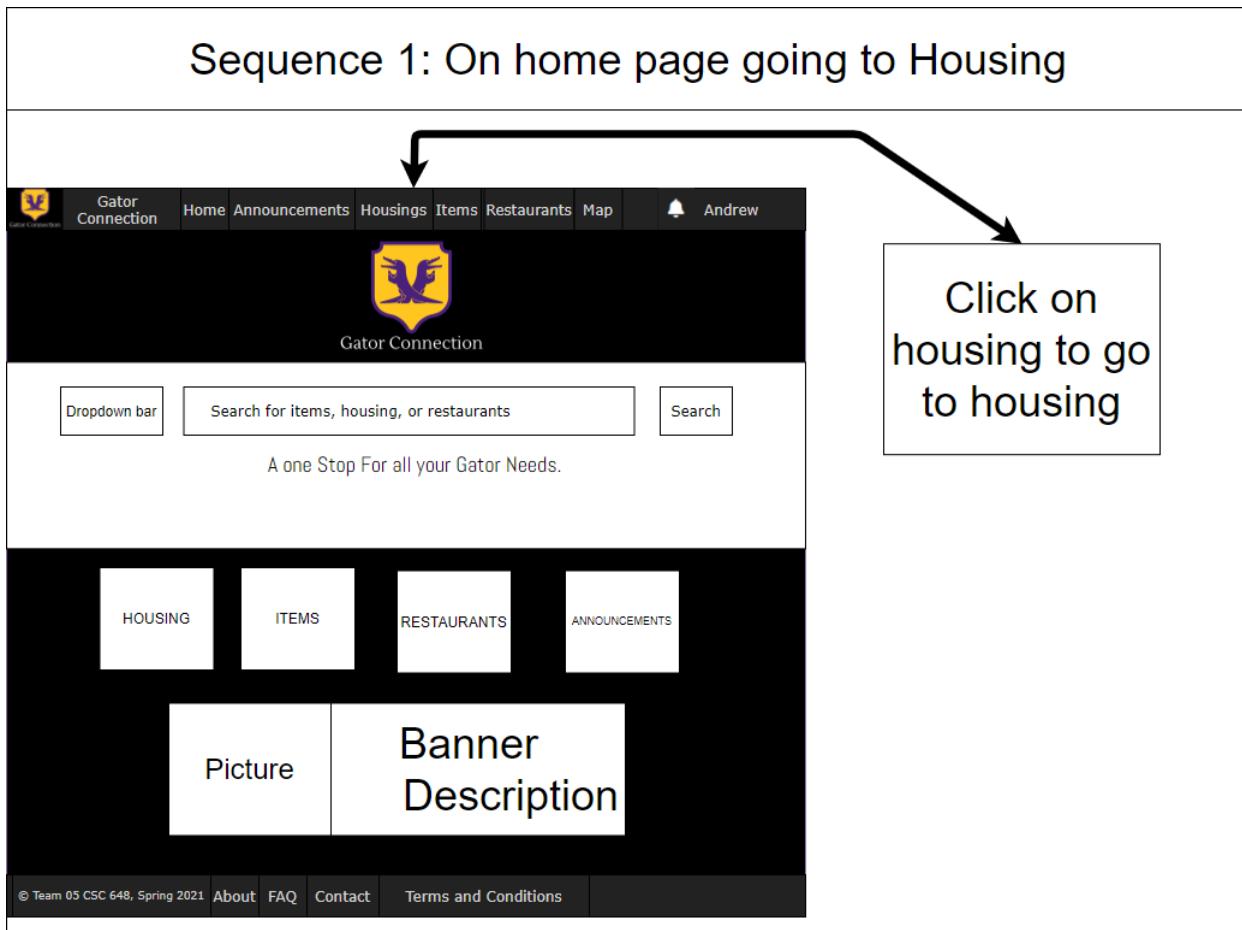
Submit

Close

Click here submits the housing post



Action 3 and 4: Search for Housing Listing/Make Request to buy Listing (Wireframe shown as Andrew)



Sequence 2: On Housing Page

The screenshot shows the 'Housing' section of the Gator Connection website. At the top, there is a navigation bar with links for Home, Announcements, Housings, Items, Restaurants, and Map. A user profile for 'Andrew' is also visible. Below the navigation bar, a search interface includes a dropdown bar, a search input field with placeholder text 'Select what to search for from the dropdown bar', and a 'Search' button. A large purple button labeled '+ Post an Advertisement' is prominently displayed. Below this, there are two tabs: 'List' (which is selected) and 'Grid'. Four advertisement cards are listed, each featuring a placeholder 'Picture' and three data fields: 'Title Value', 'Price Value', and 'Address Value'. At the bottom of the page, there is a footer with links for About, FAQ, Contact, and Terms and Conditions.

Click on picture
to bring up
Listing

Sequence 3: On John's Listing Page

The screenshot shows a listing page for a property. At the top, there is a navigation bar with links for Home, Announcements, Housings, Items, Restaurants, and Map. A user profile for Andrew is also visible. The main content area displays a title and price, followed by a description section containing several bullet points. Below the description is a purple button labeled "+ Interested?". To the right of the listing, a large rectangular box contains the text "Click here to bring up Modal". Two arrows point from this text box towards the "+ Interested?" button.

Pictures here,
placed into a
carousoul for
pictures

Title Value Here
Price Value Here

- Description:
DESCRIPTION OF HOUSING HERE
- Preferred Payment Type:
PREFERRED PAYMENT HERE
- Pets Allowed:
PETS ALLOWED VALUE HERE
- Address:
ADDRESS HERE

+ Interested?

Click here to
bring up Modal

© Team 05 CSC 648, Spring 2021 | About | FAQ | Contact | Terms and Conditions

Sequence 4: On Modal

Tell us about yourself! x

enter first name

enter last name

enter sfsu email

Phone Number:

Move in Date:

Tell us about yourself!

Submit

Close

Our system sends email to owner of listing, so in this case, John

Action 5: Receive Request to buy Listing (Wireframe shown as John)

Receive request to buy Listing

The wireframe shows a top navigation bar with 'Gator Connection' logo, 'Home', 'Announcements', 'Housings', 'Items', 'Restaurants', 'Map', a bell icon labeled 'John', and a 'Notifications' section. Below is a 'Requests' section. A callout box points to the bell icon with the text: 'Click on bell icon to get to the notifications page'. Another callout box points to the 'Requests' section with the text: 'This box represents John copying Andrew's contact info and replying to him through a third party email'.

Gator Connection Home Announcements Housings Items Restaurants Map John

Notifications Requests

You have a notification from housing

TIME VALUE HERE

Notification from Simon as well

Andrew is interested in your housing
Here is his contact info: xxxxxxxx

TIME VALUE HERE CREATOR VALUE HERE

Notification from Simon as well

© Team 05 CSC 648, Spring 2021 About FAQ Contact Terms and Conditions

Click on bell icon to get to the notifications page

This box represents John copying Andrew's contact info and replying to him through a third party email

Action 6: Take down listing

(Wireframe shown as John, on John's Listing Page that he made)

Deleting John's Listing Page as John

The wireframe illustrates the process of deleting a listing. It starts with a main listing page for 'John' which includes a photo carousel, title, price, and a list of details. Below these are two buttons: 'Edit Housing Listing Post' and 'Delete your Posting?'. A large grey arrow points from the 'Delete your Posting?' button to a confirmation dialog box. This dialog box contains the text 'Delete this Listing' at the top, a message 'Are you sure you want to delete this listing?', and two buttons at the bottom: 'Delete' (red) and 'Close'.

Pictures here,
placed into a
carousoul for
pictures

Title Value Here
Price Value Here

- Description:
DESCRIPTION OF HOUSING HERE
- Preferred Payment Type:
PREFERRED PAYMENT HERE
- Pets Allowed:
PETS ALLOWED VALUE HERE
- Address:
ADDRESS HERE

Edit Housing Listing Post

Delete your Posting?

Clicking here deletes listing

Delete this Listing X

Are you sure you want to delete this listing?

Delete Close

**Action 7: Receive email saying Listing is Closed
(Wireframe shown as Simon, going to his Notifications Page)**

Receive notification/email that listing is closed

Gator Connection Home Announcements Housings Items Restaurants Map Simon

Notifications Requests

You have a notification from housing

TIME VALUE HERE

Andrew Has closed his Listing

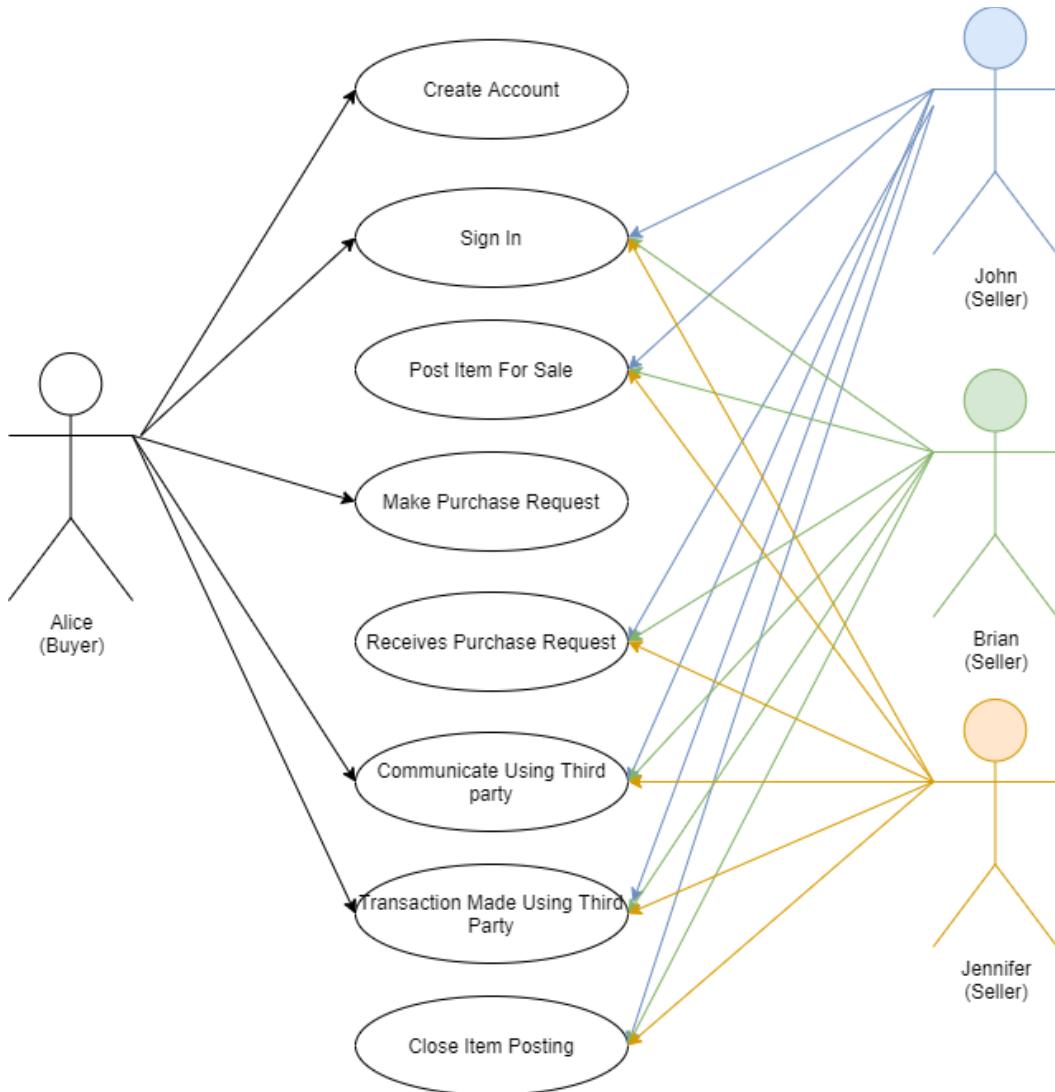
TIME VALUE HERE CREATOR VALUE HERE

© Team 05 CSC 648, Spring 2021 About FAQ Contact Terms and Conditions

This box represents Simon also receiving this message through email

Case 2: Student at SFSU looking for cheaper textbooks

Actors: Alice(Buyer), John(Seller), Brian(Seller), Jennifer(Seller)



Action 1 and 2: Creating Account/Signing into Account (Alice, John, Brian, Jennifer)

Creating account/Signing into Account

The screenshot shows the Gator Connection website with a navigation bar at the top. The navigation bar includes a logo, the text "Gator Connection", and links for "Home", "Announcements", "Housings", "Items", "Restaurants", and "Map". On the right side of the bar are "Login" and "Register" buttons. A large callout box on the right side of the page contains descriptive text about the registration process. A modal dialog box titled "Register" is overlaid on the page. The "Register" button in the modal is highlighted with a purple rectangle.

Alice proceeds to enter her first name, last name, SFSU email, graduation year, password into the respective boxes. They then click register to finish making their account. She then go to their email and verify their account to use our features

Register X

Student (On student) Admin Super User

By registering, you agree to Gator Connection's Terms and conditions.

First Name
Last name
SFSU Email
Graduation Year
Password

Register

Already have an account? Click here to sign in

Close

Action 3: Post Item Listing (Wireframe shown as John)

Sequence1: On Home Page

Click on Items to go to Items

The wireframe shows the Gator Connection homepage. At the top, there is a navigation bar with links for Home, Announcements, Housing, Items, Restaurants, and Map. A user profile for 'John' is also present. Below the navigation bar is a large yellow shield logo with a purple alligator. The main content area features a search bar with placeholder text 'Search for items, housing, or restaurants' and a 'Search' button. A descriptive tagline 'A one Stop For all your Gator Needs.' is centered below the search bar. At the bottom of the page, there is a footer with links for HOUSING, ITEMS, RESTAURANTS, and ANNOUNCEMENTS. A banner section contains a placeholder 'Picture' and a 'Banner Description'. The footer also includes links for © Team 05 CSC 648, Spring 2021, About, FAQ, Contact, and Terms and Conditions.

Sequence2: On Items Page

The screenshot shows the Gator Connection website's Items page. At the top, there is a navigation bar with links for Home, Announcements, Housings, Items, Restaurants, Map, and a user profile for John. Below the navigation bar, a welcome message "Welcome to Items" is displayed, followed by three input fields: "Dropdown bar", "Select what to search for from the dropdown bar", and a "Search" button. A large purple button labeled "+ Add an Item for Sale" is positioned prominently. To the right of this button, a callout box contains the text "Click here to open up modal". Below the main content area, there are four items listed, each with a "Picture" placeholder and three value fields: Title Value, Price Value, and Address Value. At the bottom of the page, there are links for © Team 05 CSC 648, Spring 2021, About, FAQ, Contact, and Terms and Conditions.

dropdown bar

Select what to search for from the dropdown bar

Search

+ Add an Item for Sale

List Grid

Picture

Title Value
Price Value
Address Value

Click here to open up modal

© Team 05 CSC 648, Spring 2021 About FAQ Contact Terms and Conditions

Sequence 3: Inside Post Item Modal Box

Post an Item for sale X

Title: _____
Enter Title for Item

Pricing: \$ _____
Ex: 350.00

Description: _____
Enter Description of Item here

Preferred method of Payment: Dropdown bar

Condition: Good Used

Upload Images: choose file no file chosen

Submit

Close

John enters info about his item that he is trying to sell.

**Action 4 and 5: Search for Item Listing/Make Request to buy Listing
(Wireframe shown as Alice)**

Sequence1: On Home Page

Click on Items to go to Items

The wireframe shows the Gator Connection homepage. At the top, there is a navigation bar with links for Home, Announcements, Housing, Items, Restaurants, and Map. A bell icon and the name 'Alice' are also present. Below the navigation bar is a large yellow shield logo with a purple alligator. The main content area features a search bar with placeholder text 'Search for items, housing, or restaurants' and a 'Search' button. Below the search bar is a tagline 'A one Stop For all your Gator Needs.' Underneath this is a row of four white rectangular buttons labeled 'HOUSING', 'ITEMS', 'RESTAURANTS', and 'ANNOUNCEMENTS'. At the bottom of the page is a footer bar with links for 'About', 'FAQ', 'Contact', and 'Terms and Conditions'. An arrow points from the text 'Click on Items to go to Items' down to the 'ITEMS' button in the footer.

Sequence2: On Items Page

The screenshot shows the 'Items' page of the Gator Connection website. At the top, there's a navigation bar with links for Home, Announcements, Housings, Items, Restaurants, Map, a bell icon for notifications, and a user profile for Alice. Below the navigation is a search bar with a dropdown bar placeholder 'Select what to search for from the dropdown bar' and a 'Search' button. A large purple button labeled '+ Add an Item for Sale' with a right-pointing arrow is centered above the item grid. The grid itself displays four items, each with a placeholder 'Picture' and three descriptive fields: 'Title Value', 'Price Value', and 'Address Value'. Below the grid is a footer with links for © Team 05 CSC 648, Spring 2021, About, FAQ, Contact, and Terms and Conditions.

Click on
picture to
open up
John's
Listing

Action 6: Receive Request to buy Item Listing (Wireframe shown as John)

Sequence3: On John's Listing, and replying to John

The wireframe shows a listing page for 'John'. At the top, there is a navigation bar with the Gator Connection logo, Home, Announcements, Housings, Items, Restaurants, Map, a bell icon, and the name 'Alice'. The main content area displays a listing with a title, price, and a list of details. A purple button labeled '+ Interested?' is highlighted with a large arrow pointing to a callout box at the bottom. The callout box contains the text: 'Click on interested to bring up Modal'.

Pictures here,
placed into a
carasoul for
pictures

Title Value Here
Price Value Here

- Description:
DESCRIPTION OF HOUSING HERE
- Preferred Payment Type:
PREFERRED PAYMENT HERE
- Pets Allowed:
PETS ALLOWED VALUE HERE
- Address:
ADDRESS HERE

+ Interested?

Click on interested to bring up Modal

© Team 05 CSC 648, Spring 2021 | About | FAQ | Contact | Terms and Conditions

Sequence 4: Sending Purchase Request to John



Click "Yes" to submit a request to buy to the owner.

Our system sends an email to the owner of the listing, so in this case, John

Receive request to buy Listing

The screenshot shows a mobile application interface with a dark theme. At the top, there is a navigation bar with the "Gator Connection" logo, menu items ("Home", "Announcements", "Housings", "Items", "Restaurants", "Map"), a bell icon labeled "John", and a user profile icon. Below the navigation bar, there are two main sections: "Notifications" on the left and "Requests" on the right. The "Notifications" section contains a card with the message "You have a notification from items" and placeholder text "TIME VALUE HERE". The "Requests" section contains a card with the message "Alice is interested in your item. Here is her contact info: xxxxxxx" and placeholder text "TIME VALUE HERE" and "CREATOR VALUE HERE". At the bottom of the screen, there is a footer with links: "© Team 05 CSC 648, Spring 2021", "About", "FAQ", "Contact", and "Terms and Conditions".

Click on bell icon to get to the notifications page

This box represents John receiving Alice's contact info, replying to her through a third party email, and them making a transaction using a third party app

Action 7: Take down Item Listing (Wireframe shown as John, on the detailed Item Listing page for the Item Listing John made)

Sequence1: Deleting John's Listing Page as John

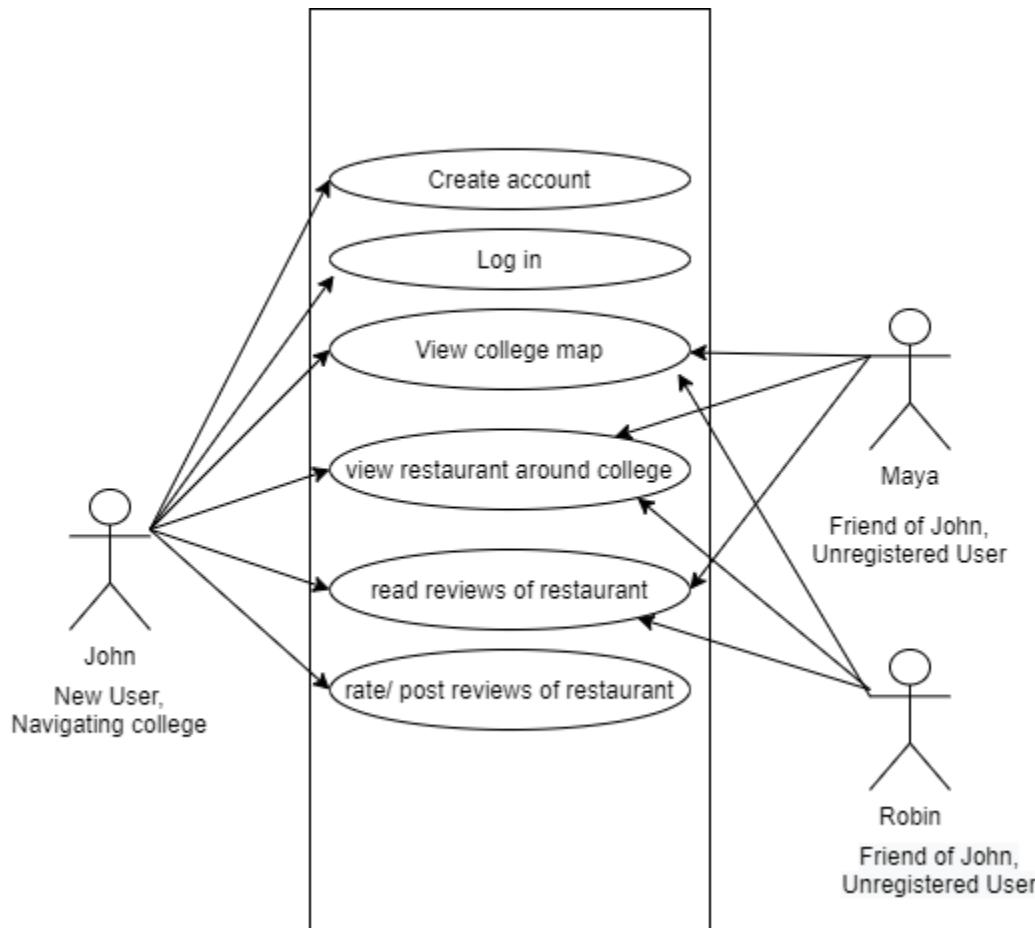
The wireframe illustrates the steps to delete a listing:

- Initial Listing Page:** Shows a listing for "Title Value Here" with a price of "Price Value Here". The listing includes a description, preferred payment type, pets allowed, address, and a carousal for pictures.
- Edit and Delete Buttons:** Below the listing are buttons for "Edit Housing Listing Post" and "Delete your Posting?".
- Delete Confirmation Dialog:** A modal dialog box asks "Delete this Listing?" and "Are you sure you want to delete this listing?". It has "Delete" and "Close" buttons.
- Final Step:** An annotation box with an arrow points to the "Delete" button in the confirmation dialog, stating "Clicking here deletes the listing".

Case 3

Student at SFSU looking for food on campus, with his two friends outside SFSU accompanying him

Actors: John(New User), Maya(Unregistered User),
Robin(Unregistered User)



Action 1 and 2: Creating/Signing into Account (John, Maya, Robin)

Sequence 1: Creating/Signing into Account

The diagram illustrates the 'Register' process on the Gator Connection website. On the left, a screenshot of the website's header shows the 'Gator Connection' logo, 'Home', 'Announcements', 'Housings', 'Items', 'Restaurants', 'Map', 'Login', and 'Register' buttons. Below the header, a modal window titled 'Register' is displayed. The modal has tabs for 'Student (On student)', 'Admin', and 'Super User'. It contains fields for 'First Name', 'Last name', 'SFSU Email', 'Graduation Year', and 'Password', each with a placeholder text. A 'Register' button is at the bottom, and a 'Close' button is at the bottom right. A large callout box on the right side contains the following text:

John, Maya, and Robin proceed to enter their first name, last name, SFSU email, graduation year, password into the respective boxes. They then click register to finish making their account. They then go to their email and verify their account to use our features

Action 3: View Virtual Map of College (Wireframe shown as John)

Sequence 1: On Home Page

The wireframe shows the Gator Connection home page. At the top is a navigation bar with the Gator Connection logo, "Gator Connection", and links for "Home", "Announcements", "Housings", "Items", "Restaurants", "Map", a bell icon, and "John". Below the navigation bar is a dark banner with the Gator Connection logo and the text "Gator Connection". A grey arrow points from the "Map" link in the navigation bar down to the "Map" button in the main content area. The main content area contains a "Dropdown bar", a search bar with the placeholder "Search for items, housing, or restaurants", and a "Search" button. Below these is a tagline "A one Stop For all your Gator Needs.". In the center of the page is a large white box divided into four sections labeled "HOUSING", "ITEMS", "RESTAURANTS", and "ANNOUNCEMENTS". To the left of this box is a "Picture" placeholder, and to the right is a "Banner Description" placeholder. At the bottom is a footer bar with links for "© Team 05 CSC 648, Spring 2021", "About", "FAQ", "Contact", and "Terms and Conditions".

Click on map to go to map

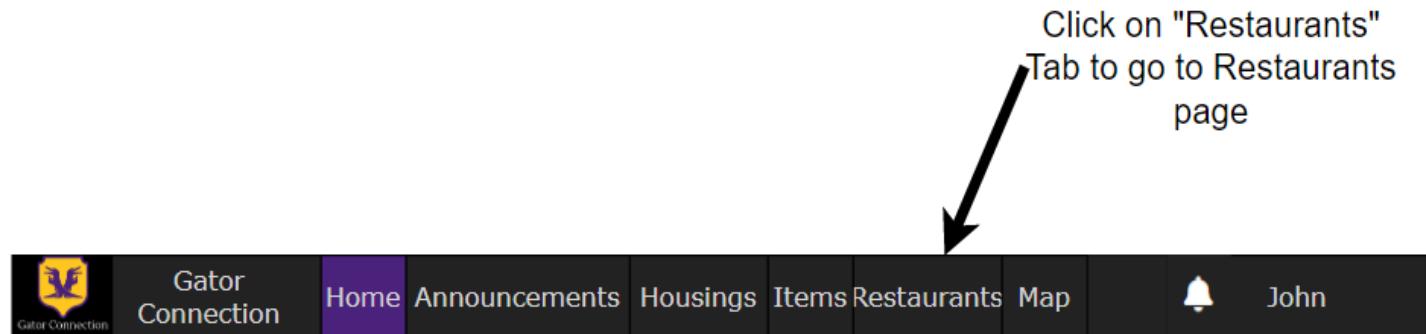
Sequence 2: On Virtual Map Page

The screenshot shows the Gator Connection website interface. At the top, there is a navigation bar with links for Home, Announcements, Housings, Items, Restaurants, and Map. To the right of the navigation bar is a user profile for 'John' with a notification bell icon. Below the navigation bar is a search bar labeled 'Search Address'. The main content area is titled 'Virtual Map' and contains a large, empty map placeholder. At the bottom of the page is a footer bar with links for © Team 05 CSC 648, Spring 2021, About, FAQ, Contact, and Terms and Conditions.

John types the address of NIZARIO'S PIZZA(SFSU) in the search bar to find the way to the restaurant

Action 4: View Restaurants Page (Wireframe shown as John)

Sequence 1: On Home Page



Sequence 2: On Restaurants Page

The screenshot shows the Gator Connection website's Restaurants page. At the top, there is a navigation bar with links for Home, Announcements, Housings, Items, Restaurants, and Map. A user profile for "John" is also visible. Below the navigation bar, a welcome message "Welcome to Restaurants" is displayed. To the right of this message are three boxes: a "Dropdown bar" containing the text "Select what to search for from the dropdown bar", a "Search" button, and a "PICTURE HERE" placeholder. To the right of the "PICTURE HERE" box is a callout box with the following text: "Find the restaurant that you want to see and click anywhere on the card to go to the details page of that specific restaurant." An arrow points from the "Search" button towards the "PICTURE HERE" box. At the bottom of the page, there is a footer with links for About, FAQ, Contact, and Terms and Conditions.

dropdown bar

Select what to search for from the dropdown bar

Search

PICTURE
HERE

Title of Restaurant

Location Value
Time Value
Take-out value
Description Value

Dont see what you're looking for? Click here to add a restaurant

© Team 05 CSC 648, Spring 2021 About FAQ Contact Terms and Conditions

Find the restaurant that you want to see and click anywhere on the card to go to the details page of that specific restaurant.

Action 5: Read/Post Reviews of Restaurant (Wireframe shown as John)

Sequence 1: On Restaurant Detail Page

The wireframe shows a dark-themed website layout. At the top is a navigation bar with a logo, "Gator Connection", and links for "Home", "Announcements", "Housings", "Items", "Restaurants", "Map", a bell icon, and "John". The main content area has a title "Restaurant Name" and a placeholder "Restaurant Picture". Below this is a "Reviews" section containing "Review Title", "Review Creator", and a large "Review Description" input field. To the right of the reviews is a sidebar titled "Related Restaurants" with four entries, each showing a "Picture" and "Restaurant Title/Description". A grey arrow points from a callout box at the bottom left towards the "Review Description" input field.

Location Value
Time Value
Take-out value
Description Value

Write a Review

Reviews

Review Title Rating

Review Creator

Review Description

© Team 05 CSC 648, Spring 2021 About FAQ Contact Terms and Conditions

Scroll through/read reviews
of the restaurant posted by
SF State students/personnel

Sequence 2: Writing a Review

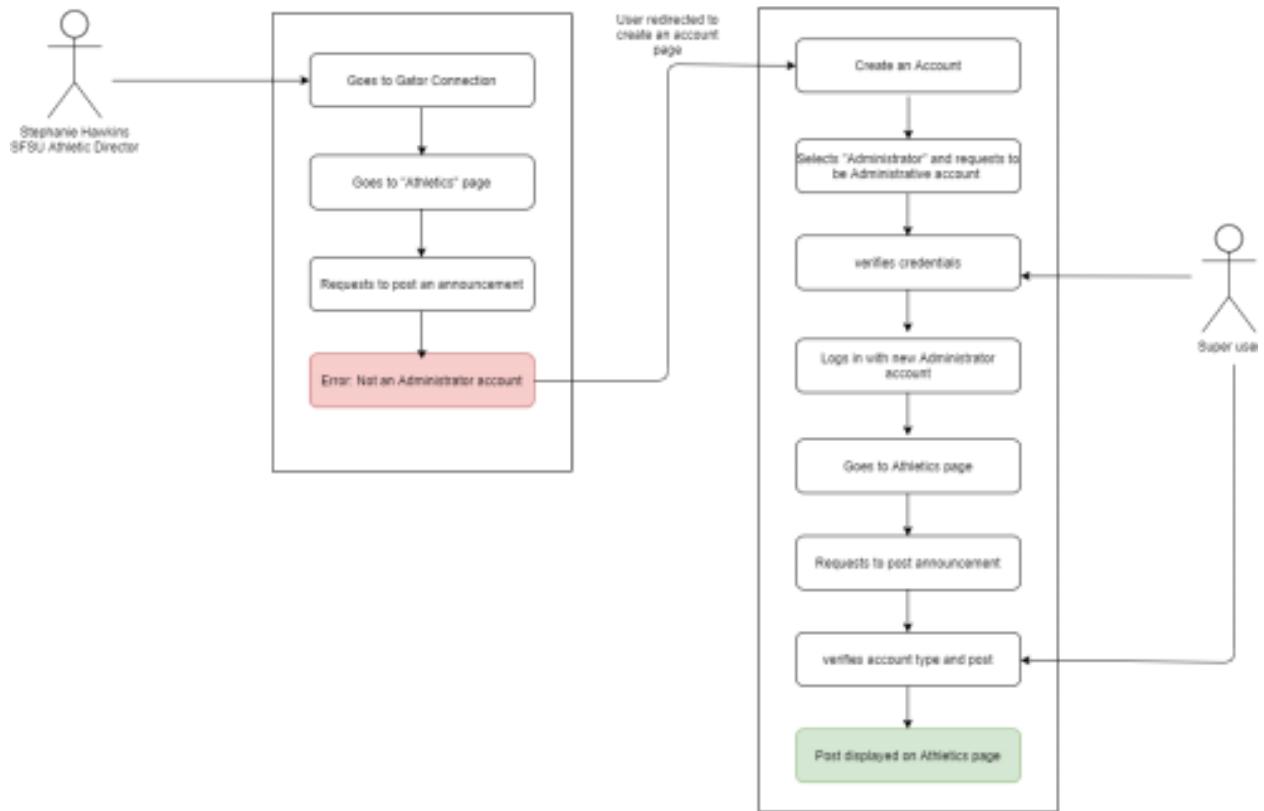
The screenshot shows a web application interface for a restaurant review site. At the top, there is a navigation bar with links for Gator Connection, Home, Announcements, Housings, Items, Restaurants, and Map. A user profile for 'John' is also visible. Below the navigation bar, the main content area displays a restaurant's information: 'Restaurant Name' (placeholder text), 'Restaurant Picture' (placeholder image), and a 'Reviews' section. The reviews section includes fields for 'Review Title', 'Rating', 'Review Creator', and a large 'Review Description' text area. To the right of the reviews section, there is a sidebar titled 'Related Restaurants' containing four entries, each with a 'Picture' placeholder and 'Restaurant Title' and 'Restaurant Description' fields. A purple button labeled 'Write a Review' is located at the bottom right of the reviews section. A callout bubble with an arrow points from the 'Write a Review' button to a text box on the right side of the screen. The text box contains the instruction: 'Click on the "Write a Review" button to open up the modal to submit a review.'

Sequence 3: In Restaurant Review Modal

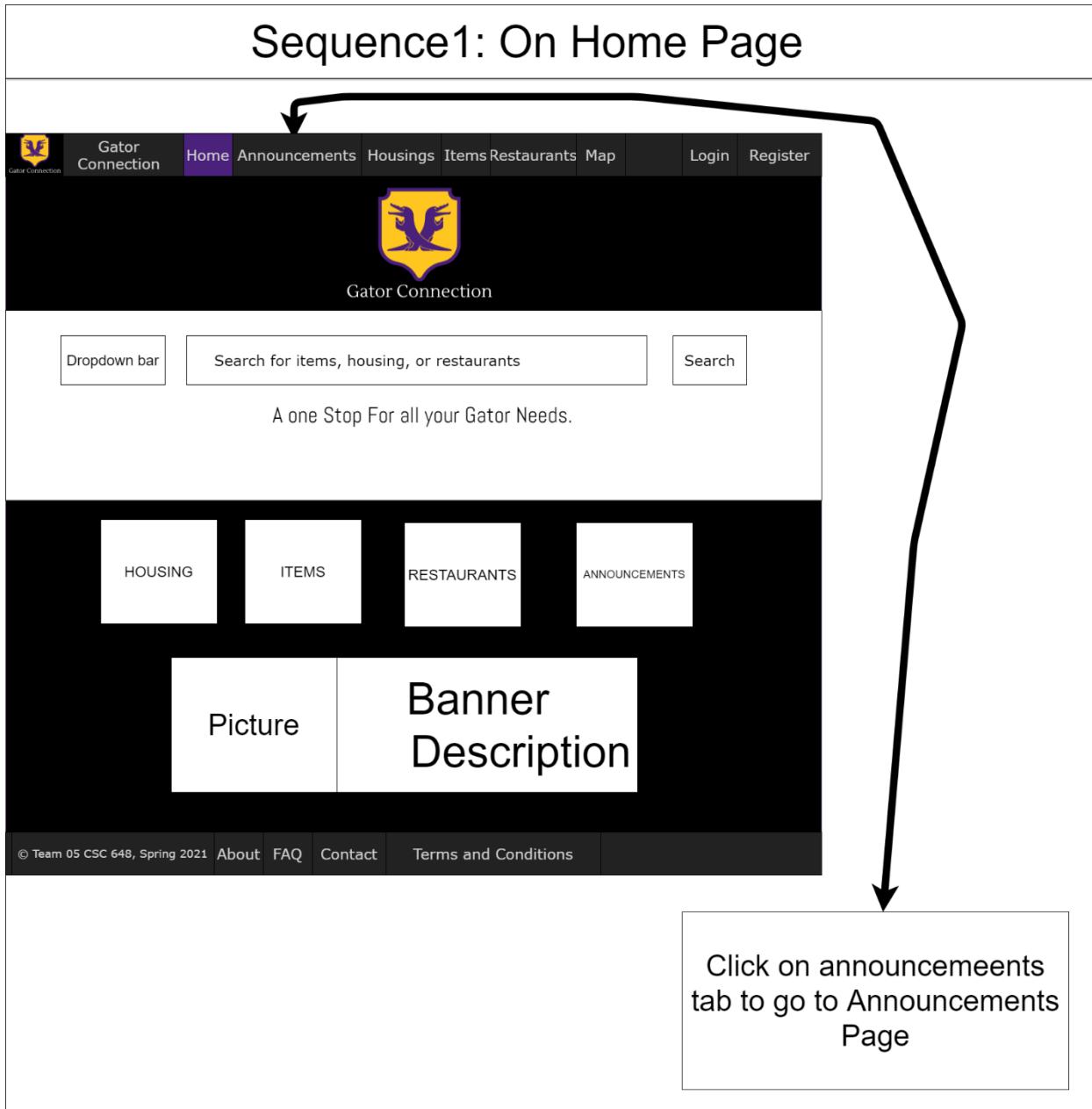
Write a Review	
<p>Title: _____ Enter Title for Review</p> <p>Select your rating: </p> <p>Write your review here</p> <p>*Note: Your name will be stored with this review.</p>	<p>Fill in all of the necessary information of the review and rate the restaurant.</p> <p>Press "Submit" to post the review</p>
<p>Submit</p> <p>Close</p>	

Case 4: Athletic Director at SFSU wants to announce upcoming events

Actors: Stephanie(User, Athletic Director)

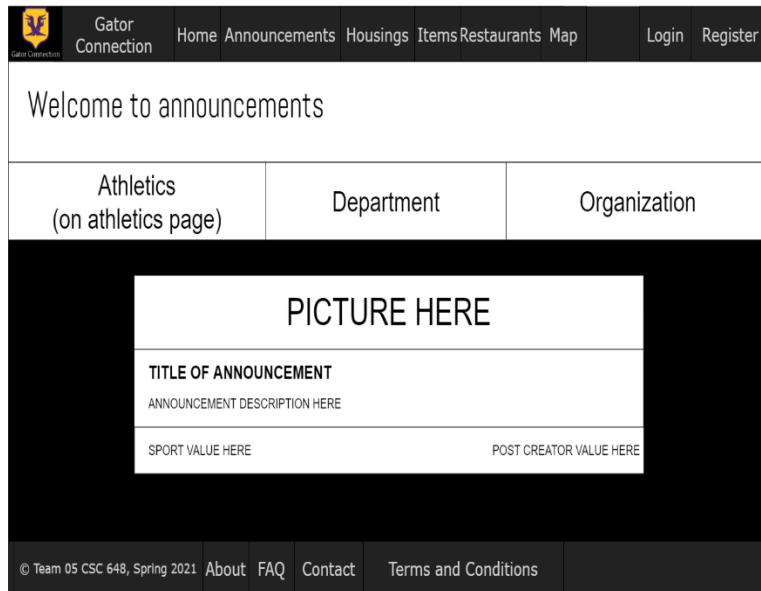


**Action 1: Goes to Announcements Page
(Stephanie Hawkins)**



**Action 2: Tries to Post Athletic announcement
(Stephanie Hawkins)**

Sequence2: On Announcement Page



can't post anything because not logged in as admin athletic account

**Action 3 and 4: Creates an athletic admin account, and signs in
(Stephanie Hawkins)**

Sequence1: Creating Admin account/Signing into Admin Account

The screenshot shows the Gator Connection website's navigation bar with links for Home, Announcements, Housings, Items, Restaurants, Map, Login, and Register. A modal window titled "Register" is open. The modal has tabs for Student, Admin (On Admin), and Super User, with "Admin (On Admin)" selected. It contains fields for First Name, Last name, Email, Sport, Position, and Password, along with a "Register" button. A note at the bottom says "By registering, you agree to Gator Connection's Terms and conditions." and a link to Athletics. A "Close" button is at the bottom right. Three arrows point from the text annotations to specific parts of the interface: one arrow points to the "Register" tab in the navigation bar; another arrow points to the "Admin (On Admin)" tab in the modal; and a third arrow points to the "Register" button in the modal.

Click On Register tab to bring up register modal then switch to admin tab

Stephanie submits her credentials, then a super user has to verify them, shown in the following next boxes. She also goes into her email and clicks on the verification link.

Sequence2: Creating Super User account/Signing into Super User Account

The screenshot shows the Gator Connection website's navigation bar at the top, featuring links for Home, Announcements, Housings, Items, Restaurants, and Map, along with Login and Register buttons. Below the navigation bar is a modal window titled "Register". The modal has tabs for Student, Admin, and Super User (On Super User). The "Super User (On Super User)" tab is highlighted. Inside the modal, there are fields for First Name, Last name, Email, and Password, followed by a "Register" button and a link to sign in if already registered. A "Close" button is at the bottom right. A callout box points to the "Super User (On Super User)" tab with the instruction: "Click on Register to bring up Register modal, and click on Super User tab". Another callout box points to the "Register" button with the instruction: "Super User inputs info to register, then goes to email to click on the verification link. Someone then verifies this account, shown in the next step".

Click on Register to bring up Register modal, and click on Super User tab

Super User inputs info to register, then goes to email to click on the verification link. Someone then verifies this account, shown in the next step

Verify Admin/Super User account

The screenshot shows a dark-themed web application interface. At the top, there is a navigation bar with links for Home, Announcements, Housings, Items, Restaurants, and Map. On the far right of the top bar is a bell icon labeled "Super User". Below the navigation bar, there are two main sections: "Notifications" on the left and "Requests" on the right. The "Notifications" section contains a message: "You have a notification from an admin creation" with a placeholder "TIME VALUE HERE". The "Requests" section contains a message: "Stephanie has requested to make an admin account" with placeholders "TIME VALUE HERE" and "CREATOR VALUE HERE", followed by two buttons: "accept?" and "reject?". A large gray arrow points from the text "Click on bell to get to this page" on the right side to the "Super User" icon at the top right of the header.

Super User clicks here to verify account, works for both admin and super users

**Action 5: Goes back to announcements and makes an athletic announcement
(Stephanie Hawkins)**

Sequence 1: Back to Announcements

The screenshot shows the Gator Connection website interface. At the top, there is a navigation bar with links for Home, Announcements (which is highlighted in purple), Housings, Items, Restaurants, Map, and a user profile for Stephanie. Below the navigation bar, a welcome message says "Welcome to announcements". Underneath, there are three categories: Athletics (on athletics page), Department, and Organization. A prominent purple button labeled "Create an Athletics Post!" is visible. Below this button is a placeholder area labeled "PICTURE HERE". Further down, there are fields for "TITLE OF ANNOUNCEMENT" and "ANNOUNCEMENT DESCRIPTION HERE", followed by "SPORT VALUE HERE" and "POST CREATOR VALUE HERE". At the bottom of the page, there is a footer with links for About, FAQ, Contact, and Terms and Conditions.

Now since Stephanie is logged in as an admin for athletics, she sees the button and clicks it to open a modal

Sequence 2: Makes an athletic announcement

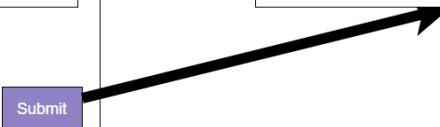
Create an Athletics Post! X

Title:

Description:

Upload Images:
 no file chosen

Stephanie enters the title, description, and uploads a photo for her announcement



Action 6: See's her post on athletics
(Stephanie Hawkins)

Stephanie see's her post on the feed

The screenshot shows a website interface. At the top is a navigation bar with the Gator Connection logo, "Gator Connection", "Home", "Announcements" (which is highlighted in purple), "Housings", "Items", "Restaurants", "Map", a bell icon, and "Stephanie". Below the navigation bar is a section titled "Welcome to announcements". Underneath this, there are three categories: "Athletics (on athletics page)", "Department", and "Organization". A purple button labeled "Create an Athletics Post!" is positioned above a white rectangular box containing the text "Stephanie's Post Here". A grey arrow points from this white box down to a grey annotation box at the bottom left, which contains the text "Stephanie now see's her post on the feed". The bottom of the page features a footer with links for "© Team 05 CSC 648, Spring 2021", "About", "FAQ", "Contact", and "Terms and Conditions".

Stephanie see's her post on the feed

Gator Connection Home Announcements Housings Items Restaurants Map Bell Stephanie

Welcome to announcements

Athletics (on athletics page) Department Organization

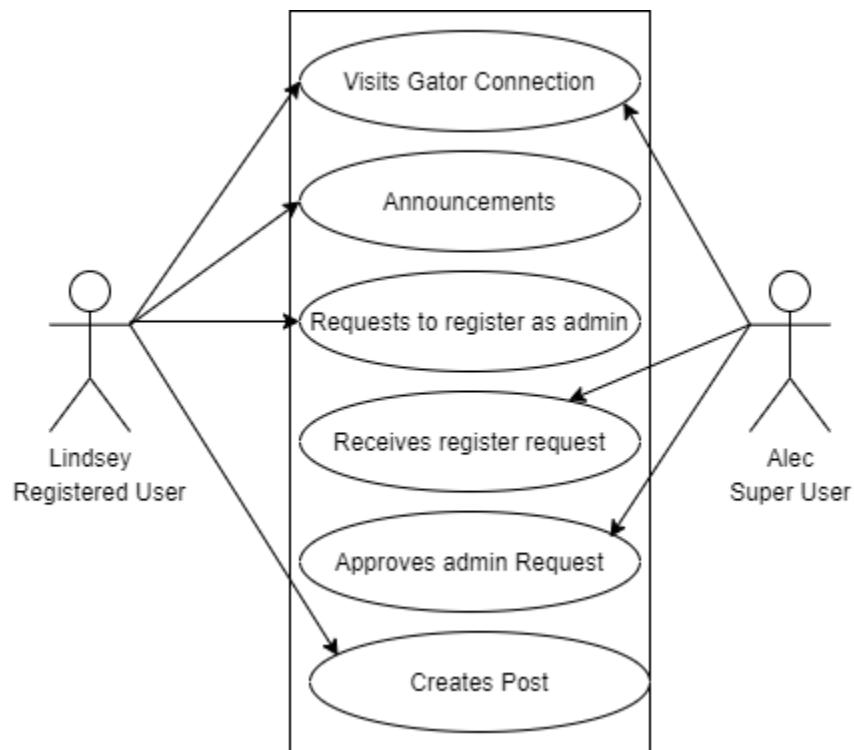
Create an Athletics Post!

Stephanie's Post Here

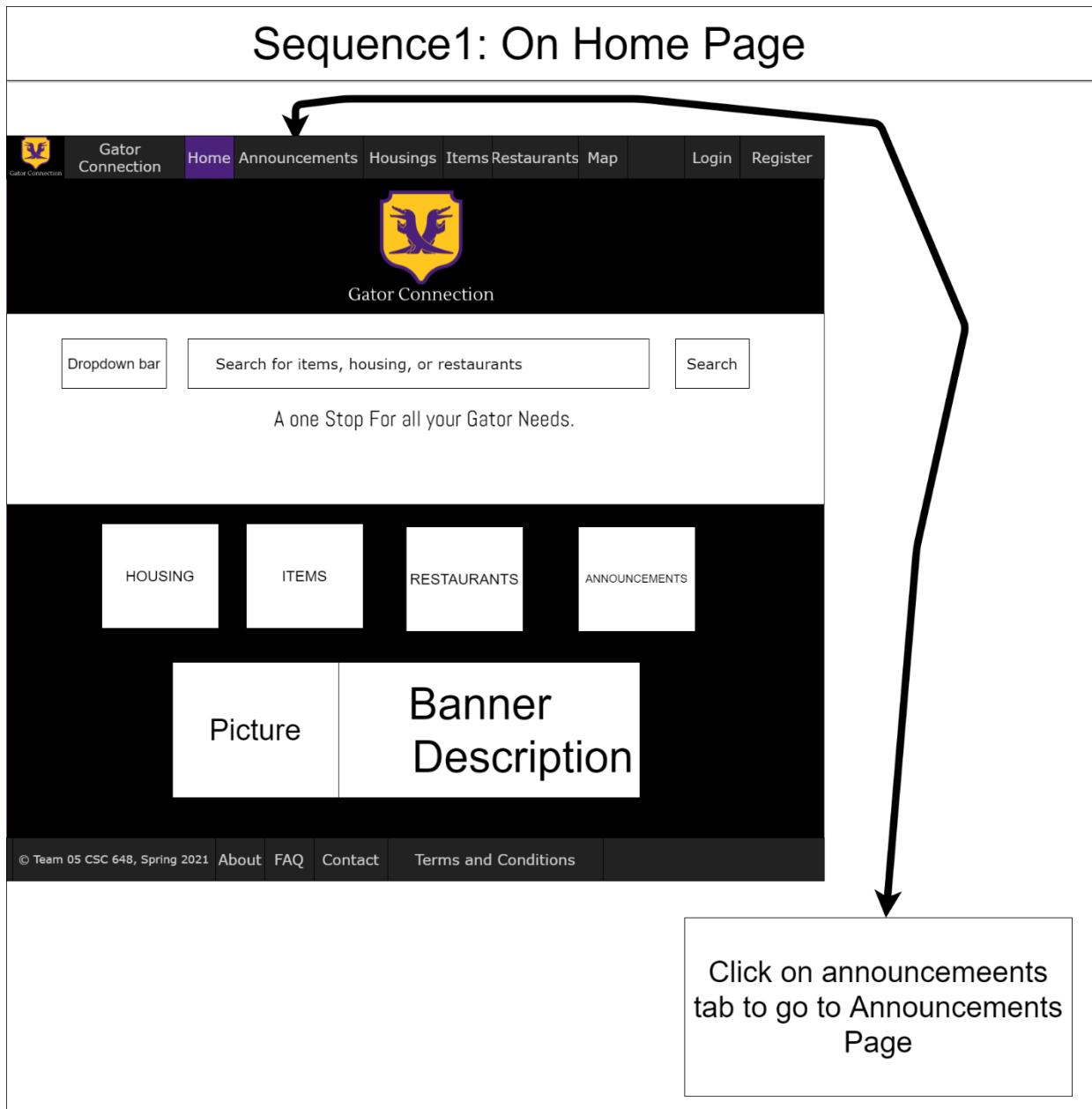
Stephanie now see's her post on the feed

Case 5: President of SF Hacks wants to post about her upcoming events

Actors: Lindsey(Registered User), Alec(Super User)



**Action 1 and 2: Goes to Announcements and wants to post a department announcement
(Lindsay)**



Sequence2: On Announcement Page

The screenshot shows the Gator Connection website. At the top, there is a navigation bar with links: Gator Connection (with a logo), Home, Announcements (which is highlighted in purple), Housing, Items, Restaurants, Map, Login, and Register. Below the navigation bar, a welcome message "Welcome to announcements" is displayed. Underneath this, there are three categories: Athletics, Department (on department page), and Organization. A large black rectangular area contains placeholder text: "PICTURE HERE", "TITLE OF ANNOUNCEMENT", "ANNOUNCEMENT DESCRIPTION HERE", "SPORT VALUE HERE", and "POST CREATOR VALUE HERE". At the bottom of the page, there is a footer with links: © Team 05 CSC 648, Spring 2021, About, FAQ, Contact, Terms and Conditions.

can't post
anything because
not logged in as
admin
department
account

Action 3 and 4: Decides to request to make an admin department account, then signs in (Lindsay)

Sequence1: Creating Admin account/Signing into Admin Account

The screenshot shows the Gator Connection website's navigation bar at the top, featuring a logo, the text "Gator Connection", and links for Home, Announcements, Housings, Items, Restaurants, and Map. Below the navigation bar is a "Login" and "Register" button. A modal window titled "Register" is open in the center. The modal has tabs for Student, Admin (On Admin), and Super User, with "Admin (On Admin)" selected. It contains fields for First Name, Last name, Email, Sport, Position, and Password, along with a "Register" button. A note below the fields states: "By registering, you agree to Gator Connection's Terms and conditions." There is also a link for "Athletics". At the bottom of the modal are "Close" and "Already have an account? Click here to sign in" links.

Click On Register tab to bring up register modal then switch to admin tab

Lindsay submits her credentials, then a super user has to verify them, shown in the following next boxes. She also goes into her email and clicks on the verification link.

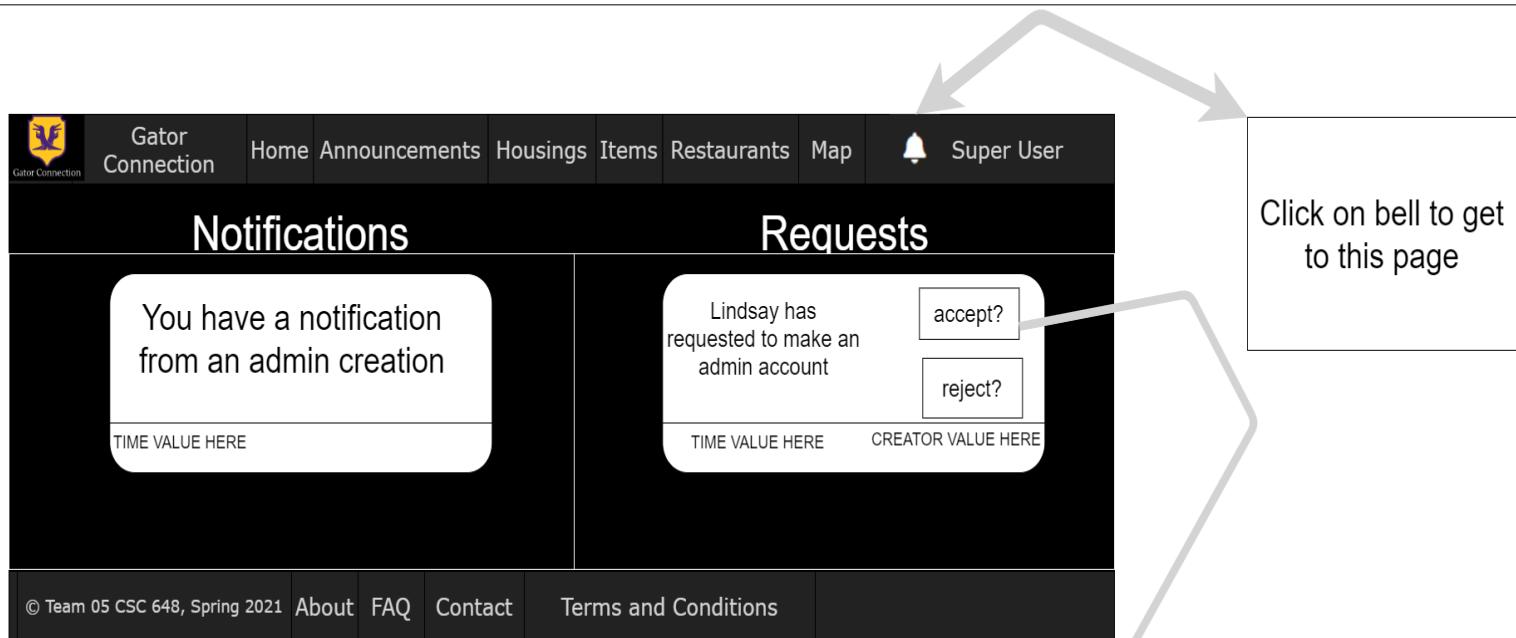
Sequence2: Creating Super User account/Signing into Super User Account

The screenshot shows the Gator Connection website's navigation bar at the top, featuring links for Home, Announcements, Housings, Items, Restaurants, and Map, along with Login and Register buttons. Below the navigation bar is a modal window titled "Register". The modal has tabs for Student, Admin, and Super User (On Super User). The "Super User (On Super User)" tab is highlighted. Inside the modal, there are fields for First Name, Last name, Email, and Password, followed by a "Register" button and a link to sign in if already registered. A "Close" button is at the bottom right. A callout box points to the "Register" button with the instruction: "Click on Register to bring up Register modal, and click on Super User tab". Another callout box points to the "Super User (On Super User)" tab with the instruction: "Super User inputs info to register, then goes to email to click on the verification link. Someone then verifies this account, shown in the next step".

Click on Register to bring up Register modal, and click on Super User tab

Super User inputs info to register, then goes to email to click on the verification link. Someone then verifies this account, shown in the next step

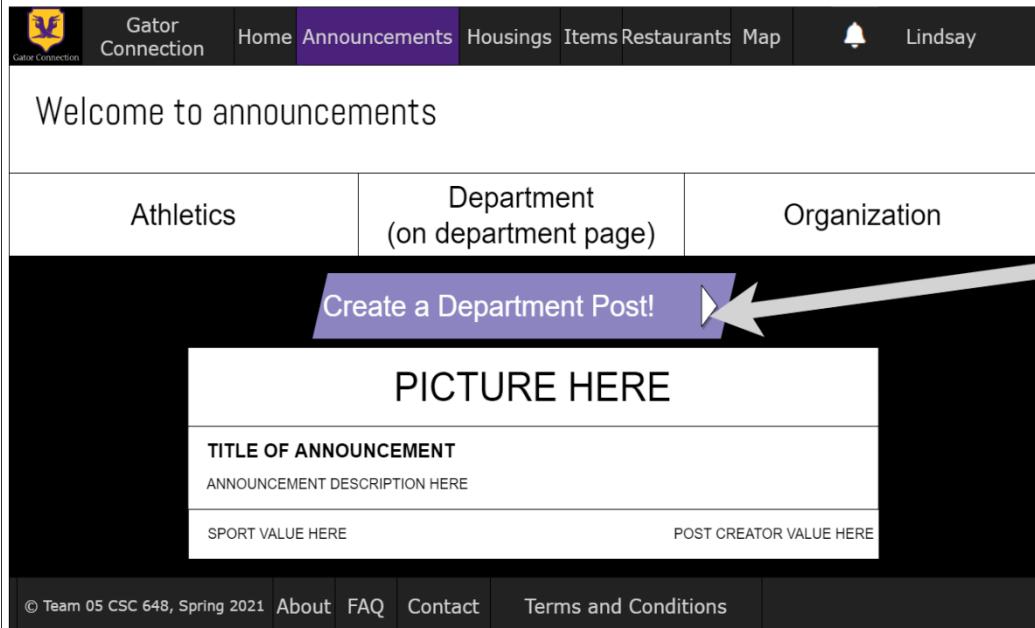
Verify Admin/Super User account



Super User clicks here to verify account, works for both admin and super users

Action 5: Makes an announcement Department Post: (Lindsay)

Sequence 1: Back to Announcements



Now since Lindsay is logged in as an admin for department, she see's the button and clicks it to open a modal

Sequence 2: Makes a Department announcement

Create an Athletics Post! X

Title:

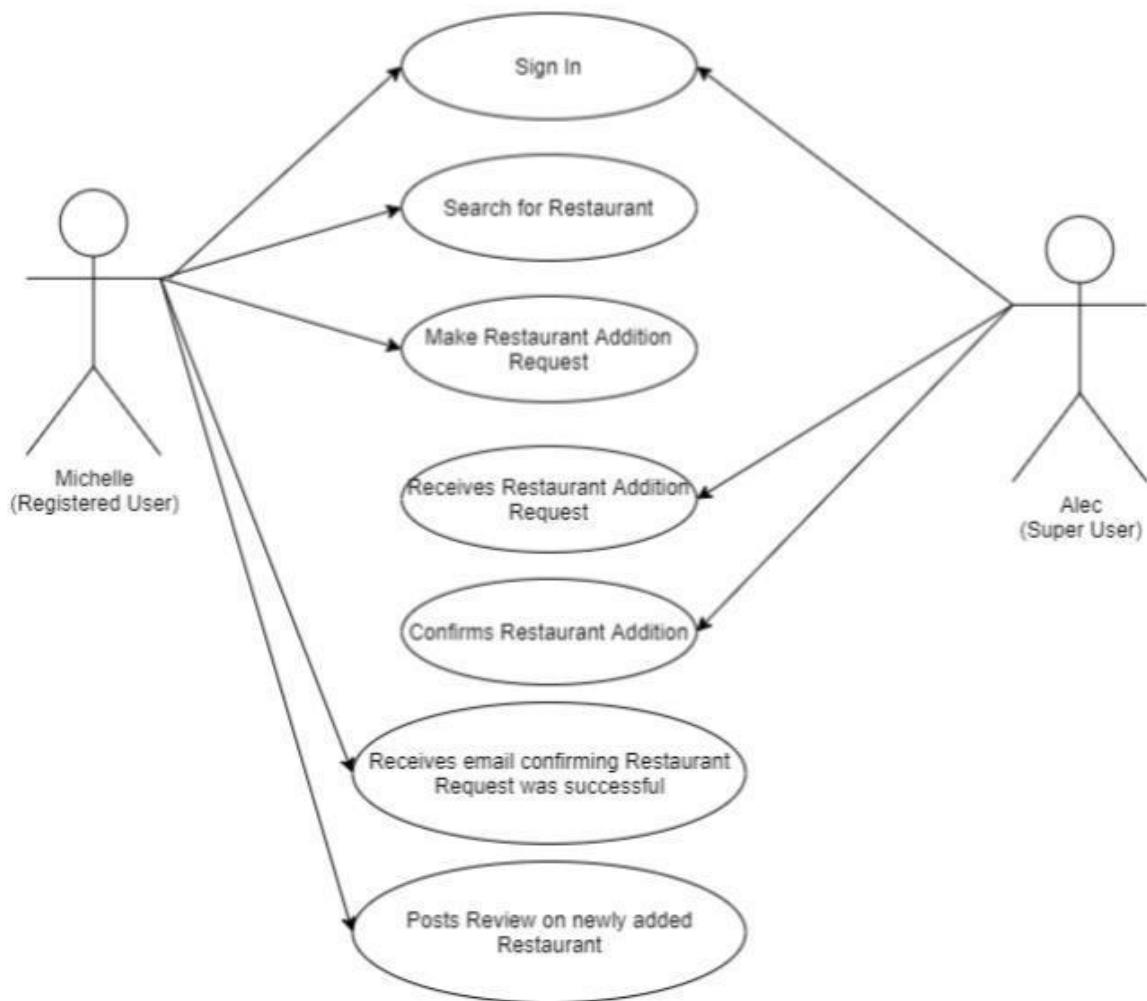
Description:

Upload Images:
 no file chosen

Lindsay enters the title, description, and uploads a photo for her announcement

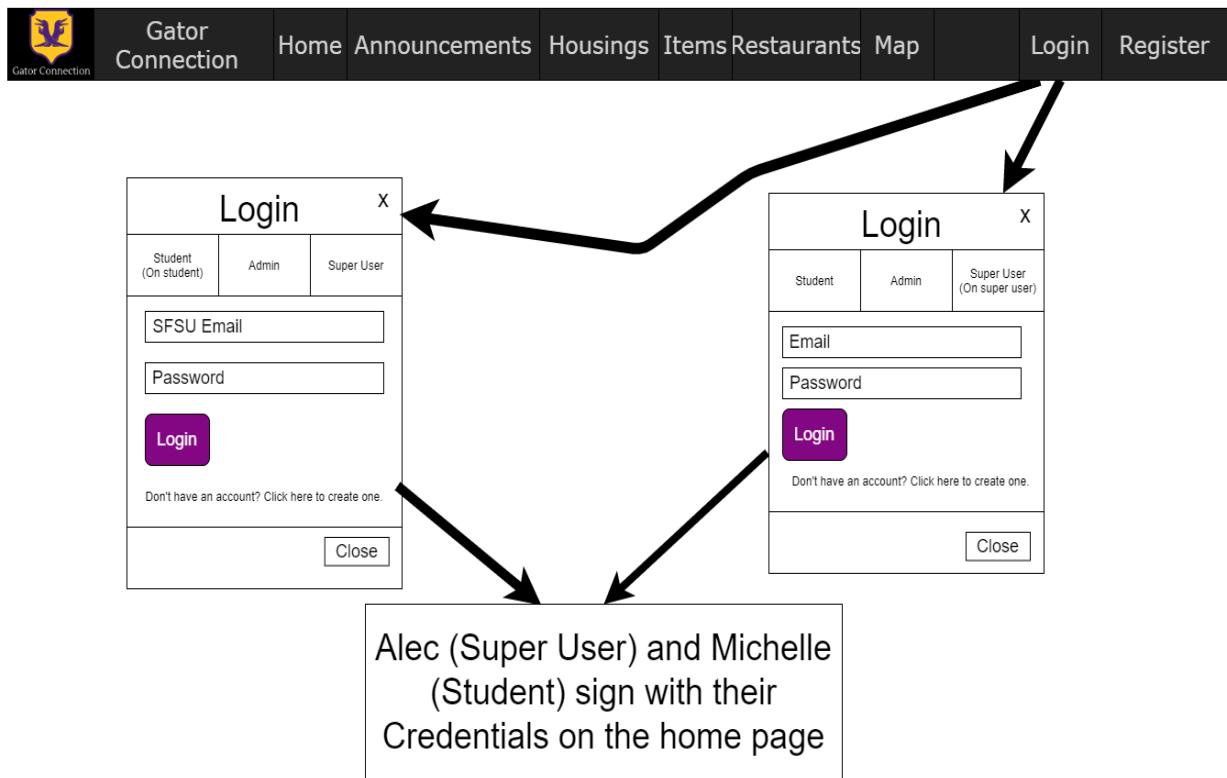


Case 6: Student at SFSU wants to add a Restaurant
Actors: Michelle (Registered User), Alec (Super User)



Action 1: Sign In (Michelle, Alec)

On Home Page in Login Modals



Action 2: Search for Restaurant (Michelle)

Sequence 1: On Home Page

Click on "Restaurants" tab to go to Restaurants page

The screenshot shows the Gator Connection website homepage. At the top, there is a navigation bar with tabs: Home (highlighted in purple), Announcements, Housings, Items, Restaurants (highlighted in orange), Map, and a user profile for Michelle. Below the navigation bar is a large yellow shield logo with a purple alligator. The text "Gator Connection" is displayed below the logo. In the center of the page is a search bar with the placeholder text "Search for items, housing, or restaurants". To the left of the search bar is a "Dropdown bar" button. To the right is a "Search" button. Below the search bar, the text "A one Stop For all your Gator Needs." is displayed. At the bottom of the page, there are four white rectangular buttons labeled "HOUSING", "ITEMS", "RESTAURANTS", and "ANNOUNCEMENTS". The "RESTAURANTS" button is highlighted in orange. Below these buttons is a banner with a picture on the left and the text "Banner Description" on the right. At the very bottom of the page is a footer bar with links: © Team 05 CSC 648, Spring 2021, About, FAQ, Contact, and Terms and Conditions.

Sequence 2: On Restaurants Page

The screenshot shows the Gator Connection website interface. At the top, there is a navigation bar with links for Home, Announcements, Housing, Items, Restaurants (which is highlighted in purple), and Map. To the right of the navigation bar is a user profile for Michelle, featuring a bell icon and the name "Michelle". Below the navigation bar, a large header says "Welcome to Restaurants". Underneath the header is a search bar with three input fields: "Dropdown bar", "Select what to search for from the dropdown bar", and a "Search" button. The main content area is black and displays the message "No Restaurants Found". Below this message is a purple button with the text "Don't see what you're looking for? Click here to add a restaurant". At the bottom of the page is a footer with links for "About", "FAQ", "Contact", and "Terms and Conditions". A callout box in the bottom right corner contains the text "Michelle searches for the restaurant she wants but cannot find it even though she knows it exists".

Gator Connection

Home Announcements Housing Items Restaurants Map

Michelle

Welcome to Restaurants

Dropdown bar Select what to search for from the dropdown bar Search

No Restaurants Found

Don't see what you're looking for? Click here to add a restaurant

© Team 05 CSC 648, Spring 2021 About FAQ Contact Terms and Conditions

Michelle searches for the restaurant she wants but cannot find it even though she knows it exists

Action 3: Make Restaurant Addition Request (Michelle)

Sequence 1: On Restaurants Page

The screenshot shows the Gator Connection website's "Restaurants" page. At the top, there is a navigation bar with links for Home, Announcements, Housings, Items, Restaurants (which is highlighted in purple), and Map. There is also a user profile for Michelle with a bell icon. Below the navigation bar, a search bar has three fields: "Dropdown bar", "Select what to search for from the dropdown bar", and "Search". The main content area displays the message "Welcome to Restaurants" and "No Restaurants Found". In the footer, there is a link to add a restaurant: "Dont see what you're looking for? Click here to add a restaurant". A large grey arrow points from this link to a callout box in the center of the page containing the text: "Click on this button to open up the \"Request a Restaurant\" modal.". The footer also includes links for About, FAQ, Contact, and Terms and Conditions.

Sequence 1: On Restaurants Page

Gator Connection Home Announcements Housings Items Restaurants Map Michelle

Welcome to Restaurants

Dropdown bar Select what to search for from the dropdown bar Search

No Restaurants Found

Dont see what you're looking for? Click here to add a restaurant

© Team 05 CSC 648, Spring 2021 About FAQ Contact Terms and Conditions

Click on this button to open up the "Request a Restaurant" modal.

Sequence 2: In "Request a Restaurant" modal

Request a restaurant x

Name: _____
Restaurant Name _____

Zip Code: _____ Number: _____
Zipcode _____ Number _____

Street: _____
Street _____

City: _____
City _____

Business Hours:

Open: _____ Close: _____

Takeout Available: yes no

Description: _____
Enter Description of Restaurant here

Upload Images:
 choose file no file chosen

Submit

Close

Fill in all of the respective information about the restaurant you want to add.

Press the "Submit" button to submit the restaurant addition request



Action 4: Verify Restaurant Request (Super User Alec)

The screenshot shows a web application interface titled "Verify Restaurant Request". At the top, there is a navigation bar with links: Gator Connection, Home, Announcements, Housings, Items, Restaurants, Map, and a bell icon labeled "Alec". A grey arrow points from the "Alec" label to a callout box on the right containing the text "Click on bell to get to this page". Below the navigation bar, the main content area has two sections: "Notifications" and "Requests". The "Notifications" section contains a message: "You have a notification from restaurants" and a placeholder "TIME VALUE HERE". The "Requests" section contains a card with "Restaurant Info: XXXX" and two buttons: "Accept" (green) and "Reject" (red). A grey arrow points from the "Accept" button to a callout box below it containing the text "Alec clicks accept to accept the restaurant addition". At the bottom of the page, there is a footer with links: © Team 05 CSC 648, Spring 2021, About, FAQ, Contact, Terms and Conditions.

Verify Restaurant Request

Gator Connection Home Announcements Housings Items Restaurants Map Alec

Click on bell to get to this page

Notifications Requests

You have a notification from restaurants

TIME VALUE HERE

Restaurant Info: XXXX

Accept

Reject

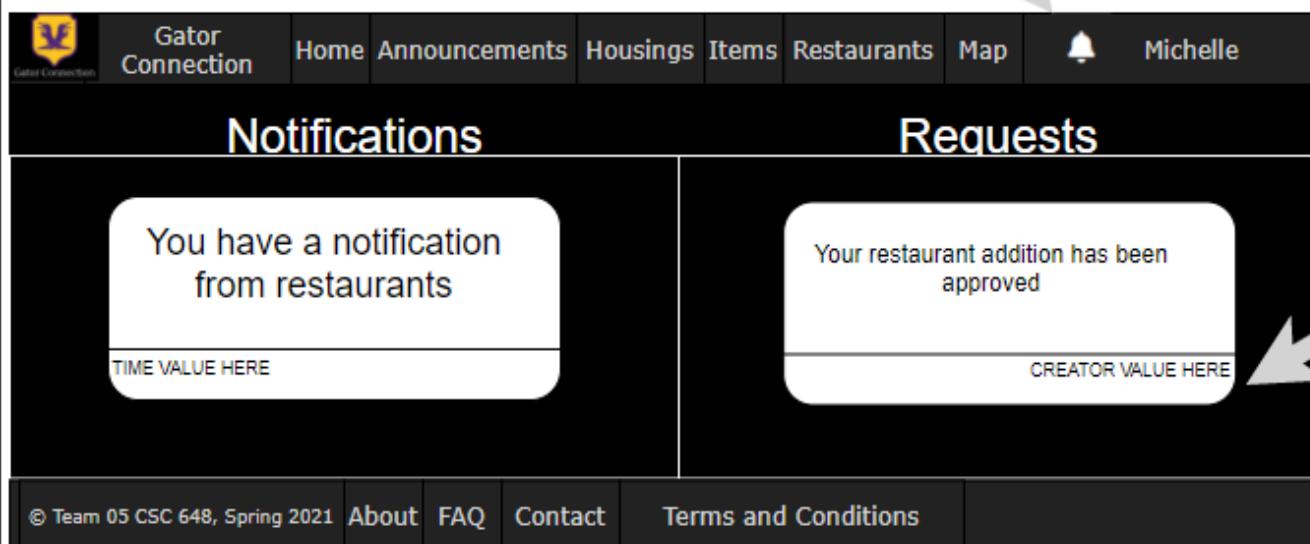
CREATOR VALUE HERE

Alec clicks accept to accept the restaurant addition

© Team 05 CSC 648, Spring 2021 About FAQ Contact Terms and Conditions

Action 5: Notified of Approved Restaurant Addition through email and notifications

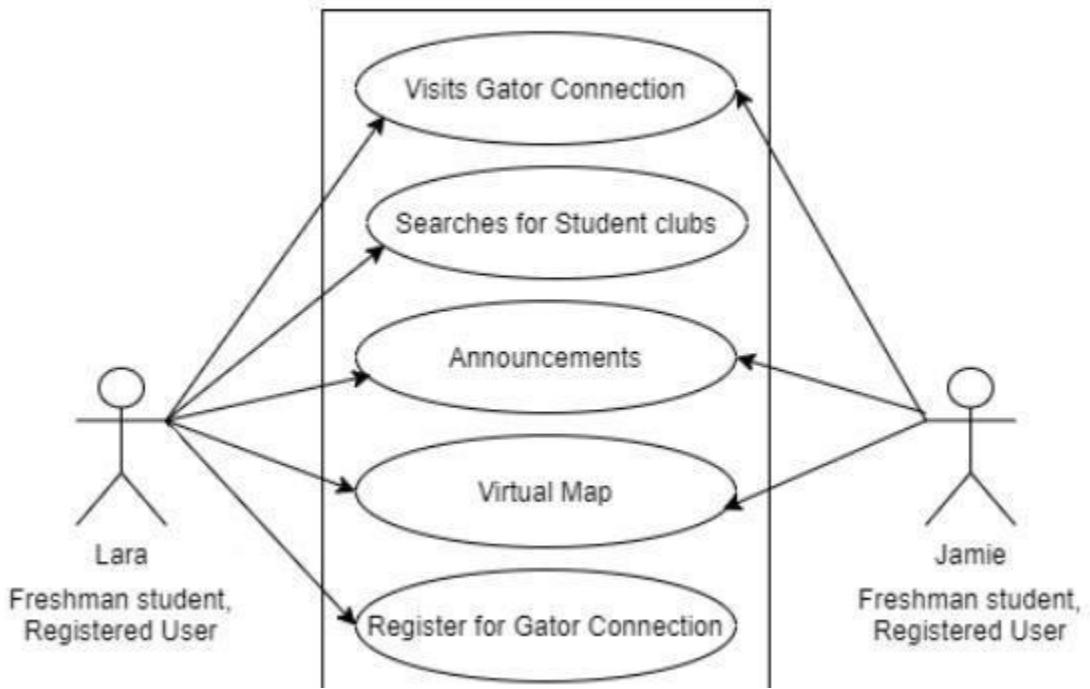
Michelle receives notification of addition of restaurant through notification and email



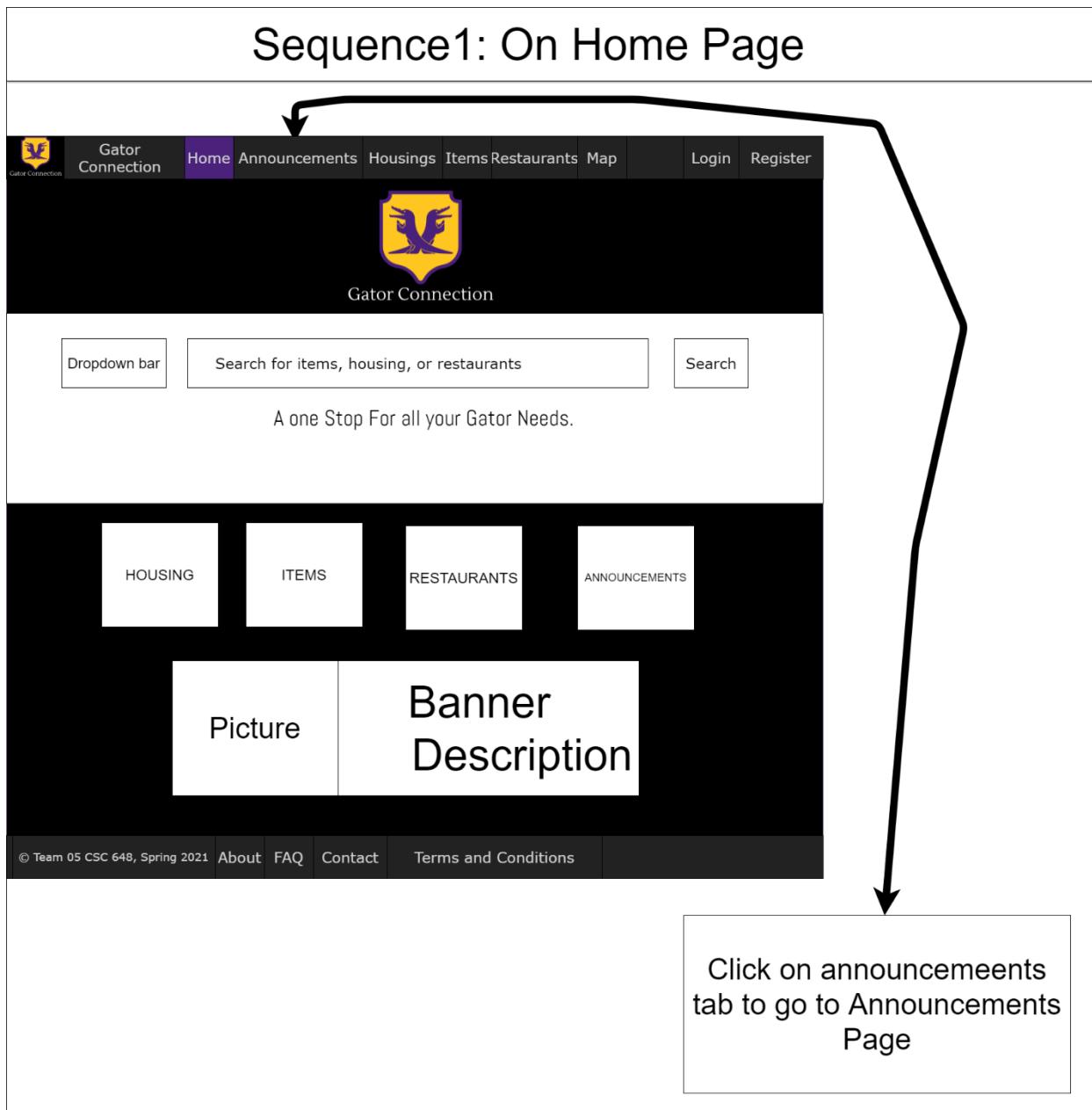
Click on bell to get to this page

Michelle receives a notification and email that her restaurant addition request has been accepted

Case 7: Two Freshman at SFSU want to join a club but want more information
Actors: Lara (Registered User), Jamie (Registered User)



Action 1 and 2: Goes to Announcements and wants to find a club to join on Organization (Lara)



Sequence2: On Announcement Page

The screenshot shows a website layout for "Gator Connection". At the top, there's a navigation bar with links for Home, Announcements (which is highlighted in purple), Housings, Items, Restaurants, Map, Login, and Register. Below the navigation bar, a welcome message says "Welcome to announcements". Underneath this, there are three tabs: Athletics, Department, and Organization (On Organization Tab). A large black rectangular area contains a placeholder "PICTURE HERE". Below this, there's a white box containing the "TITLE OF ANNOUNCEMENT" and "ANNOUNCEMENT DESCRIPTION HERE". At the bottom of this white box, there are two smaller boxes: one labeled "SPORT VALUE HERE" and another labeled "POST CREATOR VALUE HERE". At the very bottom of the page, there's a footer with links for © Team 05 CSC 648, Spring 2021, About, FAQ, Contact, Terms and Conditions.

Click on picture
to open up Post
page

Action 3: Goes to Announcements and wants to find a club to join on Organization (Lara)

On Post Page

PHOTO WITH ANNOUNCEMENT HERE

TITLE OF ANNOUNCEMENT
POST CREATOR VALUE HERE
TIME POST CREATED HERE
The event will be held at:
CESAR CHAVEZ STUDENT CENTER SAN FRANCISCO STATE UNIVERSITY

© Team 05 CSC 648, Spring 2021 | About | FAQ | Contact | Terms and Conditions

Lara and Jamie copy the address of the event

Action 4: Goes to to Virtual Map after getting the address
(Lara)

On Map Page

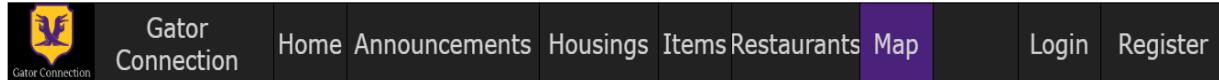
Click on Map page to go to Map

Lara and Jamie type the address: CESAR CHAVEZ
STUDENT CENTER SAN FRANCISCO STATE
UNIVERSITY
in the search bar and proceeds to look at the map here
to find their way to the event

© Team 05 CSC 648, Spring 2021 | About | FAQ | Contact | Terms and Conditions

**Action 5: Registers for Gator Connection after having a fun time at the event
(Lara)**

Creating account/Signing into Account

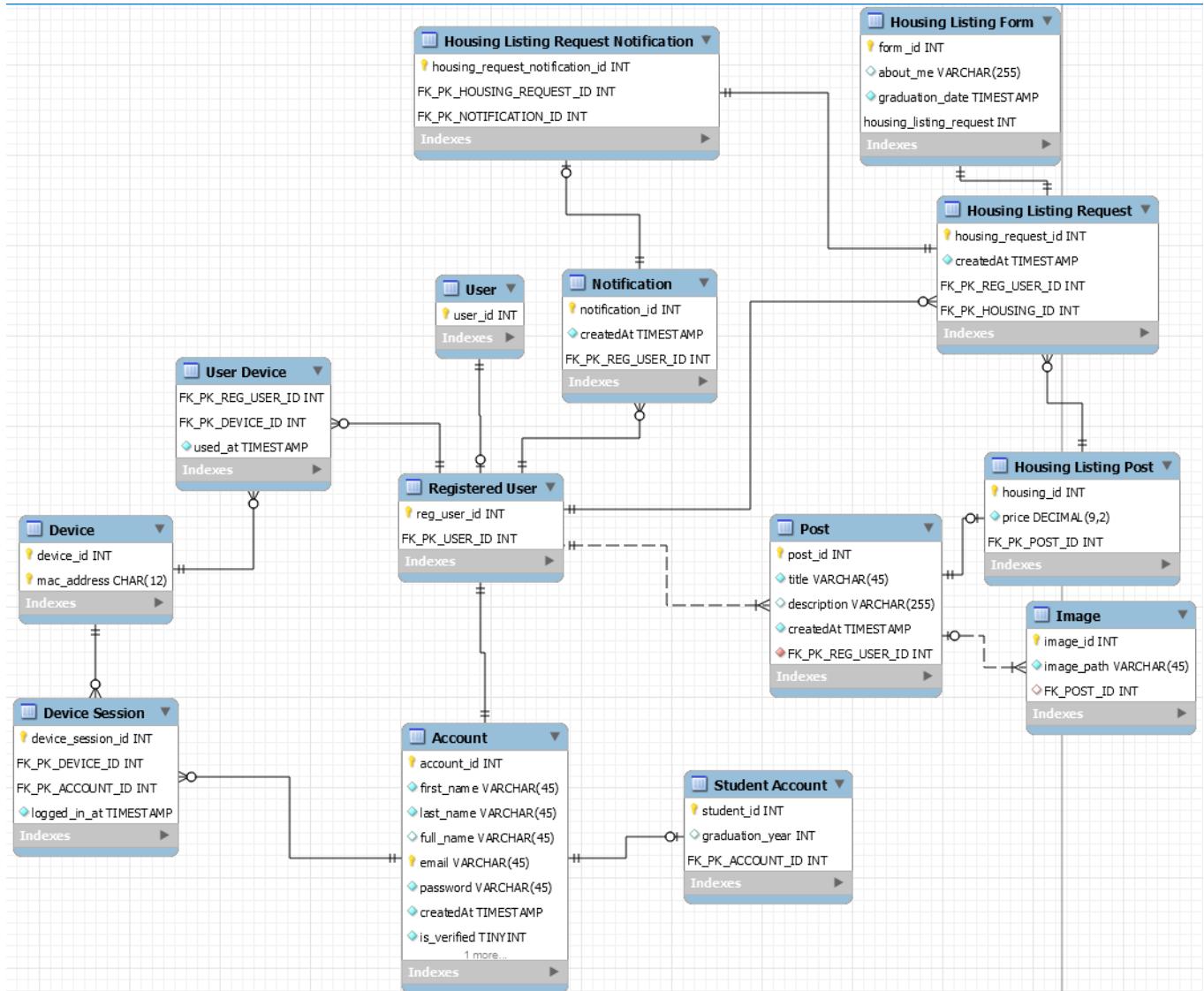


Click on Register to bring up Register Modal

Register			X
Student (On student)	Admin	Super User	
By registering, you agree to Gator Connection's Terms and conditions.			
First Name			
Last name			
SFSU Email			
Graduation Year			
Password			
<input type="button" value="Register"/>			
Already have an account? Click here to sign in			
<input type="button" value="Close"/>			

Lara proceeds to enter her first name, last name, SFSU email, graduation year, password into the respective boxes. She then clicks register to finish making her account. She then clicks on the verification link in her email to use the features

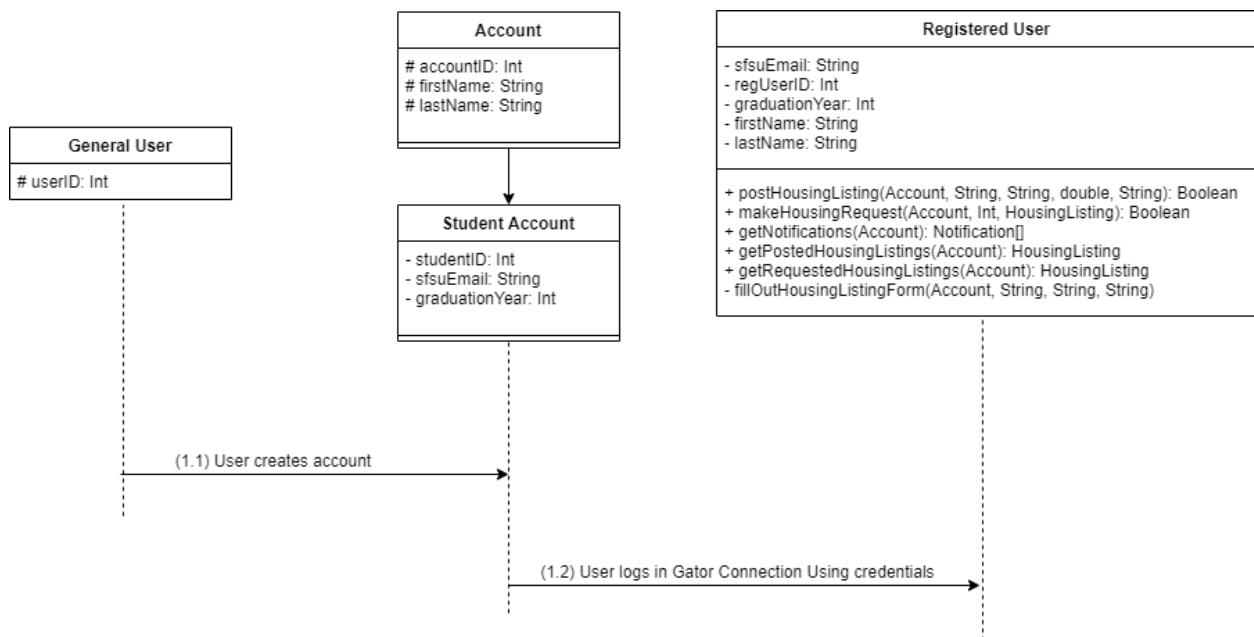
High Level Database Architecture and Organization V2



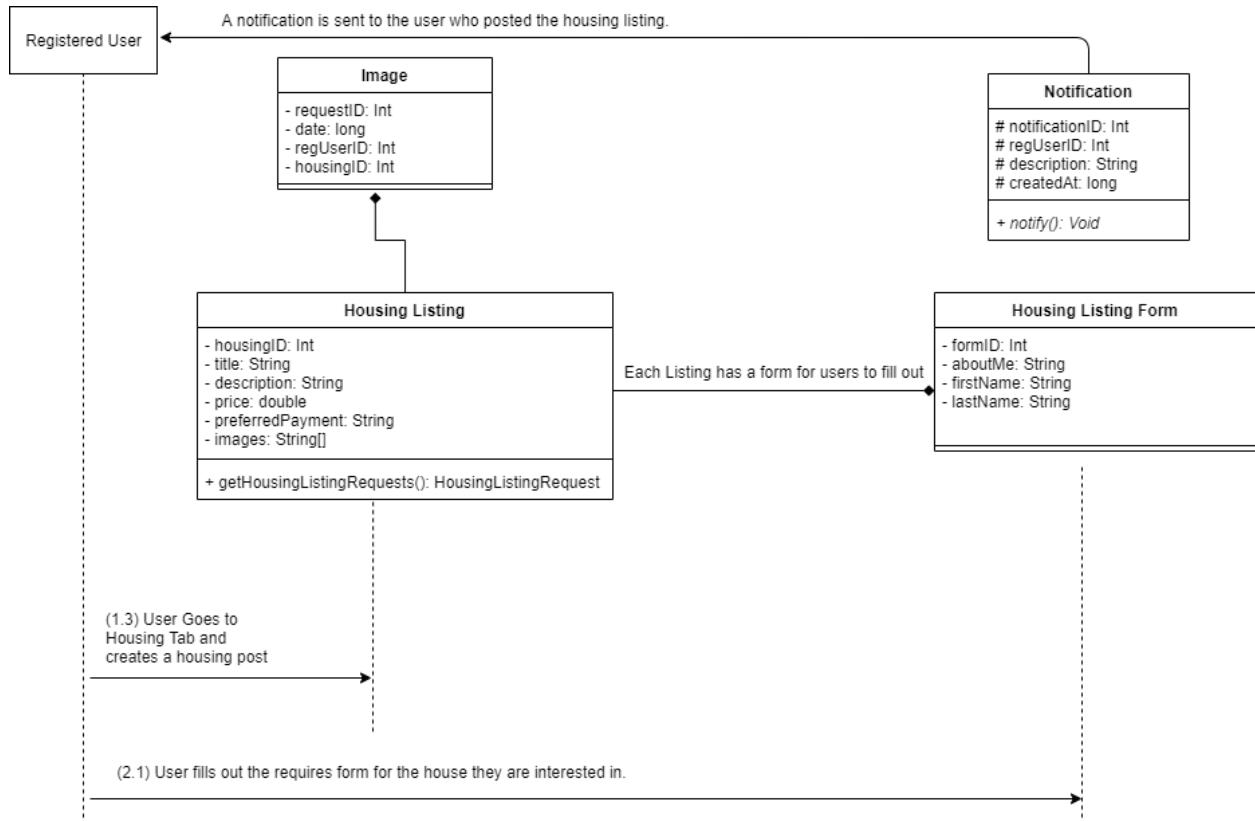
High Level Diagrams V2

High Level UML Diagram

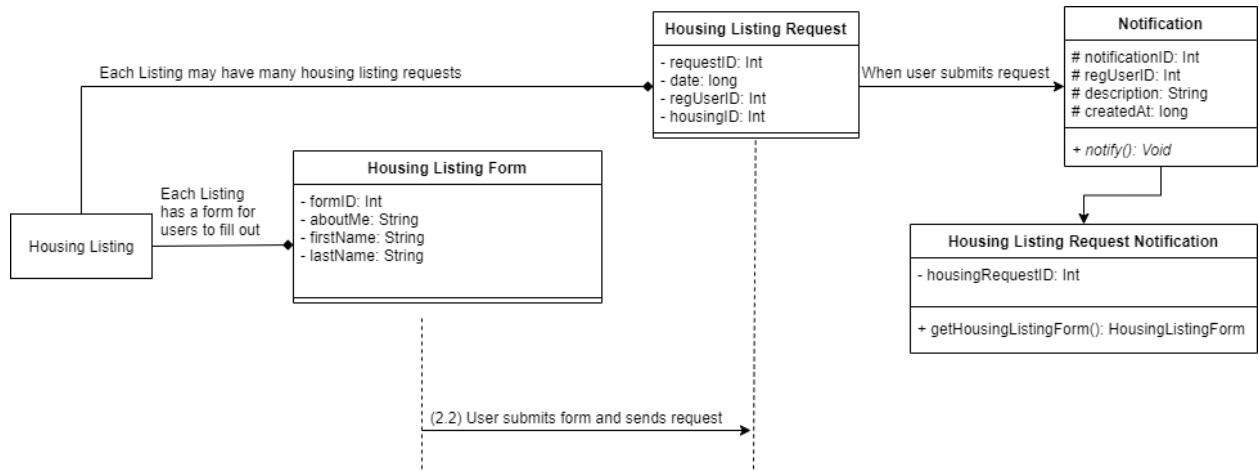
- ## *1. Association and Relationships between the General User, Registered User, Account, and Student Account.*



2. *Associations and Relationships between Registered User, which was already defined above, Housing Listing, Image, Housing Listing Form, and Notification.*

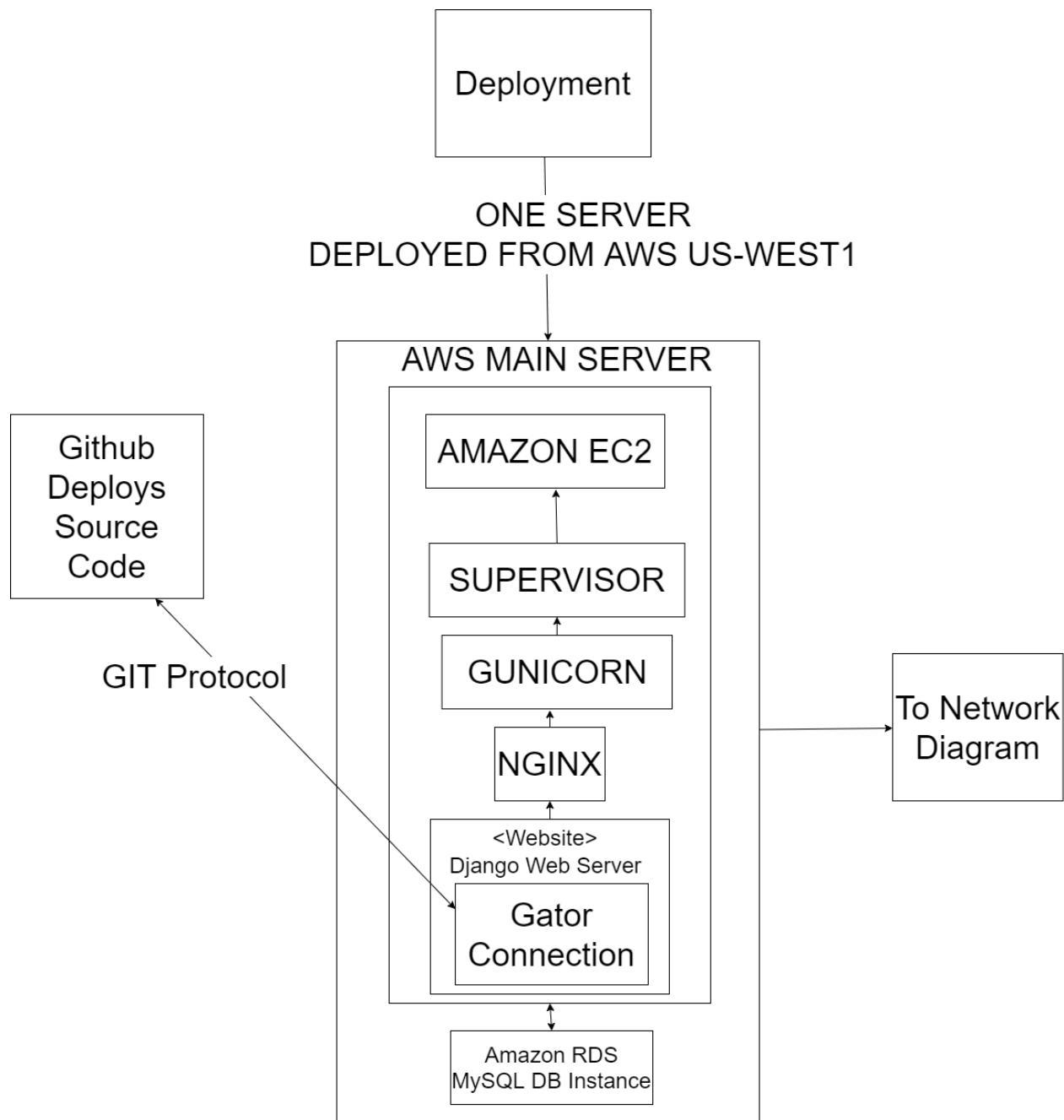


3. *Associations and relationships between Housing Listing, which was already defined above, Housing Listing Form, Housing Listing Request, Notification, and Housing Listing Request Notification.*

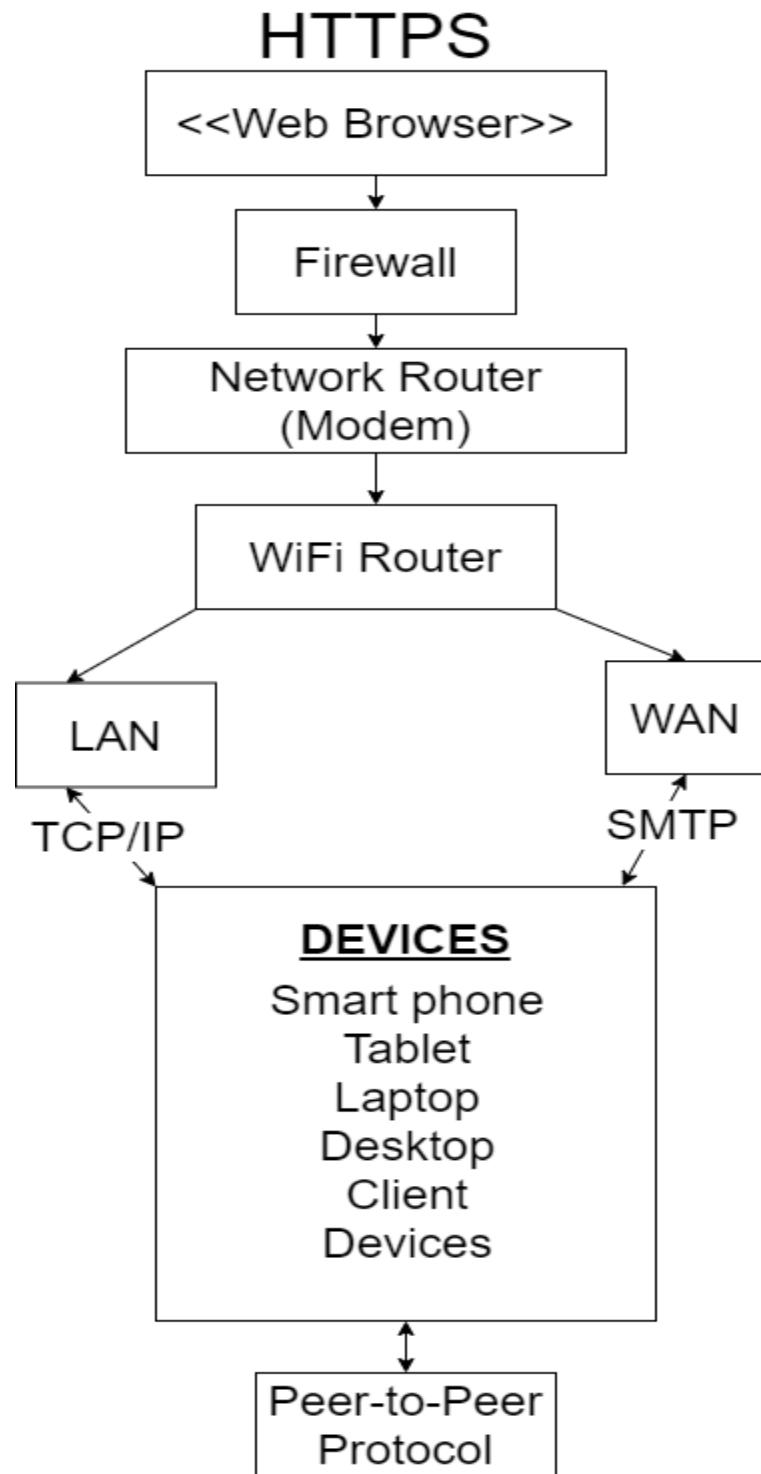


High Level Application Network and Deployment Diagrams

Deployment Diagram:



Application Network Diagram:



List of Contributions

Jiaxin:

a) Backend:

Backend features that Jiaxin was tasked to work on was the email feature, specifically:

i) email_helper.py file:

In the email_helper.py file, it is used to send out emails after certain actions, such as registering, requesting a housing listing, requesting an item, etc. We did manage to get an email to be sent out after a user registers but decided to disable it for this current horizontal prototype.

ii) item.py file:

Jiaxin set up the basic framework for the items endpoint, such as being able to post an item, render an item post, etc.

b) Frontend:

Frontend features that Jiaxin worked on was:

i) items.html

In the items.html file, Jiaxin provided the basic rendering to show the items coming from the database.

Carmen:

a) Frontend:

Frontend features that Carmen worked on was setting up the basic layout of the grid system that we have in:

- i) items.html**
- ii) housing.html**

In these files, Carmen set up the grid system that we have in there. There a user can select a “list” option, so the listing objects can be presented in a vertical fashion. By selecting grid, a user can set these objects at most 4 across, where it will make another row.

Another Frontend feature that Carmen worked on was designing the logo that we have at the top left and aligning it to make sure it matches the orientation on the navbar.

Furthermore, Carmen added a placeholder value for our searchable map, so it would always go to SFSU.

b) documentation:

Things for the M3 document that Carmen worked on specifically was getting the rough draft for our wireframes ready, specifically for Wireframes 5 and 6.

Bikram:

a) Frontend:

Frontend features that Bikram worked on was setting up dummy pages, specifically:

- i) **restaurant_post(1-3).html**
- ii) **announcements_post(1-3).html**
- iii) **housing/item_post1.html**
- iv) **notification.html**

In these files, Bikram set up dummy pages. By having these dummy pages, when it came time to have our data from the database all styled up, we could easily copy the html, switch out the variables and everything would be rendered up to our specifications. Furthermore, a dummy page was set up to some extent in **restaurant.html**, where after it was done, we were able to copy dynamic data from the database and apply css on it.

Furthermore, Bikram also worked on setting up the **maps.html** page we have, including the searching option that is currently there as well.

b) Documentation:

Things for the M3 document that Bikram worked on specifically was getting the rough draft for our wireframes ready, specifically for Wireframes 7 and 8.

Lakshita:

a) Frontend:

Frontend features that Lakshita worked on was setting up the button format shown below:



After this button was created and applied to our **housing.html**, **item.html**, and **restaurants_home.html** page by Lakshita, Alec was able to easily copy the html and css for it and apply it to our **announcements_home.html**, and all the **_detail.html** pages from items, housing, and announcements.

Furthermore, another frontend feature Lakshita worked on was expanding the grid system we have in our **housing.html** and **item.html** pages. She expanded upon it in a way so that it was cleaned up previously, and now implements the “shadow” card system shown there, similarly to Craigslist.

Additionally, Lakshita also provided the “banner” photos and the welcome text in the **housing.html**, **item.html**, **restaurants_home.html**, and **announcements_home.html** files. These banners show a photo representing the page. She also implemented new ttf fonts for the headings.

b) Documentation:

Things for the M3 document that Lakshita worked on specifically was getting the rough draft for our wireframes ready, specifically for Wireframes 3 and 4.

c) Scriber and Editor

Lakshita volunteered to be the scribe for notes during our project presentation for its implementation in our upcoming milestones.

Lakshita worked on editing the documentation for all individual sections to be submitted as a final M3 pdf.

Angelo:

a) Backend:

Backend features that Angelo worked on was making the logic work for passing in database data into the respective html page, specifically:

i) **housing.py file:**

In the housing.py file, Angelo worked on getting the logic right to pass the data into the respective **housing.html** file. By doing this, we were able to render text from the database into the html page, which we could style as necessary. Furthermore, after getting the logic done, we were able to copy the logic for the housing endpoint and apply it to other endpoints, such as those on **item.py** and **announcement.py**. By having the logic done, we were able to easily apply it to other endpoints. Going off rendering from the database, Angelo collaborated with Benjamin to create a way to render images without storing it in the database, rather, in the project itself. By applying the logic for this, we were able to successfully have dynamic images render from what the user would submit.

b) Frontend:

Frontend features that Angelo worked on was getting the logic right to render database data on:

i) **housing.html file**

By getting the logic right, we are able to apply this to all the pages that fetched data from the database, which are basically all the main pages for items, announcements, restaurants, etc. We were able to restrict users from inputting invalid housing prices. (i.e with 3 decimal places)

ii) **item.html file**

We also were able to restrict users from inputting invalid item prices. (i.e with 3 decimal places)

c) Documentation:

i) **Database Architecture:**

Things for the M3 document that Angelo helped on was working with Benjamin to get everything from our database architecture read. Furthermore, he helped Benjamin revise our Database model to account for the changes we had.

Benjamin:

a) Backend:

Everything that Benjamin did for the backend includes:

i) The entirety of the “database” folder in gator_connection.

Here, Benjamin set up the logic for the endpoints for our announcements, housing, items, and authentication section. This means that Benjamin set up the logic of us being able to post to the database. Here, by setting up the logic for us being able to post to the database, we were able to have dynamic data. Furthermore, Benjamin also set up the entirety of our authentication system, which includes the logging in and registering of accounts, registering for different accounts(student, admin, super user), and showing which data would render on a page depending on what user was logged in as. For example, someone who was not logged in would not see buttons to write a review for restaurants. Benjamin also set up our search system for this milestone. The search feature works across the home page, housing page, and item page. Benjamin also set up custom exception classes, for when we would run into exceptions, such as when a user gets their login wrong, or if they register with an email that's already in the system.

ii) The entirety of our MySQL DataBase and Schema.

Here, Benjamin set up the entirety of our schema in MySQL. He named the schema “**gator_connection**”, and set up all of the tables inside the schema as well, with all the properties that would come with it, such as a table that has a row that would only accept unique values.

b) Frontend:

i) The Restaurant Detail Page

Benjamin revamped Bikram’s restaurant_post(1-3).html page into restaurant_detail.html so that it showed data from the database and added a related restaurant column to increase the usefulness of the page for users.

ii) Cards for the Announcement and Restaurant Pages

Benjamin created the cards that layout each announcement post and restaurant post.

iii) Added Mission Statements to each Main Section Page

Benjamin formatted and added mission statements to the sections: Announcements, Housing, Items, and Restaurants.

iv) Created Terms and Conditions & Privacy Policy

Benjamin used a terms and conditions generator and a privacy policy generator to create our website’s Terms and Conditions and Privacy Policy. He also edited the wording of these documents so they more accurately represented our website.

v) The Notifications Page

Benjamin revamped Bikram's notification.html dummy page to have a "Requests" section where users can see requests such as housing listing requests and item requests. Super Users can see account requests and restaurant requests.

c) Documentation:

i) Database Architecture:

Benjamin revised our Database model to account for our schema that he set up.

Alec:

a) FrontEnd:

FrontEnd pages that Alec implemented features on are:

i) announcements.html

On announcements, features that were implemented were the tab system to switch between athletics, organization, and department.

ii) navbar.html

Switched the navbar background color to what it is now.

iii) item_detail.html, housing_detail.html

Alec designed the layout for all the “shop” detail pages

Features that were implemented by Alec were:

i) basic modal structure shown on all the pages.

After setting up the basic modal layout on housing, Alec tasked Bikram and Benjamin mostly to apply these modals to the other pages.

ii) revamped button

After Lakshita finished her button design, Alec designed a smaller button from her design for all “edit”, “interested”, “delete” buttons across all other pages.



iii) Password Validation

Alec implemented a password validation for the registering boxes, so that a user can only register if their password is 7 or more characters.

b) Documentation:

i) WireFrames:

Alec tasked out the front end team and himself to do the rough drafts of the wireframes. After those rough drafts were done, Alec designed how the overall website would look by setting up diagrams to show which pages would lead to what. After that, Alec revised the rough draft wireframes to what they are now to account for the website.

ii) Functional Requirements v3:

During a team meeting, Alec had the team decide which ones were priority 1 requirements to account from the feedback from the professor.

iii) Data Def v3:

Alec saw that there were no changes to this section, so we left it as is.

iv) High Level Diagrams v2:

Alec revamped the entire Application Network and Deployment Diagram to match the professors feedback. Alec also decided that the UML diagram was fine as it.

v) List of Contributions:

Alec filled out this section, detailing what people did for this milestone, whether it be for the document or for the horizontal prototype.

Milestone 4 V2

Software Engineering CSC 648/848 Spring 2021

Gator Connection

A One Stop Website for SF State Gators

Team 05

Team Lead/Github Master: Alec Stephen Tenefrancia alectene@mail.sfsu.edu

Frontend Lead: Lakshita Chugh

Backend Lead: Angelo Gloria Reyes

Frontend member: Bikram Tamang

Backend member/Document Master: Benjamin Patrick Kao

Frontend member: Carmen Denisse Paisano

Backend member: Jiaxin Yu

Milestone

Milestone/Version	Date
M4V2	05/14/21
M4V1	05/13/21
M3V2	04/25/21
M3V1	04/22/21
M2V2	04/24/21
M2V1	04/01/21
M1V2	03/09/21
M1V1	03/04/21

Table of Contents

Table of Contents	193
Product Summary	196
Usability Test Plan	201
Test Objectives:	201
Test Description:	202
Usability Task Description:	203
Questionnaire:	211
QA Test Plan	217
Test Objectives:	217
QA Test Plan:	218
QA Test Plan #1:	219
QA Test Plan #2:	226
QA Test Plan #3:	234
Coding Style	240
Coding Style	240
Code Review	241
Self-Check on Best Practices for Security	254
Self-Check: Adherence to Original Non Functional Specs	267
List of Contributions	272

Product Summary

1) Name of Product:

The name of our product is **Gator Connection**, a one stop website for SFSU students for housing, items, announcements, and restaurants near SFSU.

2) What is Unique about our product:

What is unique about our product compared to what other listing services have is our housing listings service. With our housing listings service, the one who posted the listing will have the final say in who he/she chooses to contact. We will do this by implementing a feature where buyers will not have access to any contact information about the listing. Instead they will only have access to the general location of the room, pictures of the room, and any other information about the room posted. Buyers can inquire that they are interested in the housing through a form that will contain information that they write about themselves; however, it will be up to the one who posted the listing to reply to the buyer that he/she is interested in selling them the room. By doing this, sellers of the room can rest easy knowing that they have the choice on who to contact. On top of this, our feature gives sellers the power to prevent their contact information from being shared, unlike other competitors such as Craigslist.

3) URL:

<https://www.gator-connection.com/>

4) Itemized List of Priority 1 Requirements:

Priority 1 Requirements:

- Guest User
 - A guest user shall be able to search for restaurants by title.
 - A guest user shall be able to navigate through announcements.
 - A guest user shall be able to create a student account using his/her unique SFSU email.
 - A guest user shall be able to create an admin account using his/her unique email.
 - A guest user shall be able to create a super user account using his/her unique email.
 - A guest user shall be able to navigate to a map of SFSU.
 - A guest user shall be able to search for items on sale by preferred payment.
 - A guest user shall be able to search for items on sale by price.
 - A guest user shall be able to search for items on sale by title.
 - A guest user shall be able to search for housing on sale by preferred payment.
 - A guest user shall be able to search for housing on sale by price.
 - A guest user shall be able to search for housing on sale by title.
 - A guest user shall be able to search for housing on sale by location.
- Registered User
 - A registered user shall have all of the same permissions as guest users.
 - A registered user shall be able to log in to his/her account using his/her unique email.
 - A registered user shall be able to log in to his/her account using many devices.
 - A registered user shall be able to log out of the website.
 - A registered user shall be able to stay logged in if they have not logged out.
 - A registered user shall be able to verify his/her email.
 - A registered user shall be able to post reviews of restaurants/food.
 - A registered user shall be able to make purchase requests for items.
 - A registered user shall be able to see and search housing listings.
 - A registered user shall be able to make a request for a housing listing.
 - A registered user shall be able to edit a housing listing title he/she posted.
 - A registered user shall be able to edit a housing listing description he/she posted.
 - A registered user shall be able to edit a housing listing price he/she posted.
 - A registered user shall be able to change a housing listing image he/she posted.
 - A registered user shall be able to close a housing listing that he/she posted.
 - A registered user shall be able to edit a post about an item for sale that he/she posted.
 - A registered user shall be able to take down an item that he/she put up for sale.

- A registered user shall be able to create a restaurant request to add a restaurant to the restaurants listing.
 - A registered user shall receive a notification about a house listing being closed if he/she had made a request to the house listing.
 - A registered user shall receive a notification about an approved restaurant that he/she requested to add.
 - A registered user shall receive a notification about a rejected restaurant that he/she requested to add.
 - A registered user shall have a unique registered id.
 - A registered user shall be able to fill out a form for a housing request.
 - A registered user shall receive a notification about purchase requests made on his/her account by email.
 - A registered user shall receive a notification about purchase requests made on items he/she posted for sale by email.
 - A registered user shall receive a notification about a house listing request for a house listing that he/she posted that includes information about the person who requested, such as his/her name and email address.
 - A registered user with an approved admin account shall be able to post announcements.
 - A registered user with an approved admin account shall be able to remove an announcement he/she had posted.
- Student Account
 - A non-verified Student account shall prevent registered users from accessing item features.
 - A non-verified Student account shall prevent registered users from accessing housing features.
 - A non-verified Student account shall prevent registered users from accessing restaurant features.
 - Admin Account
 - An admin account shall give registered users the ability to post announcements.
 - Super User Account
 - A registered user with an approved super user account shall be able to approve newly created admin accounts.
 - A registered user with an approved super user account shall be able to reject newly created admin accounts.
 - A registered user with an approved super user account shall be able to approve newly created super user accounts.
 - A registered user with an approved super user account shall be able to reject newly created super user accounts.

- Sale Item
 - A sale item shall be posted by a registered user.
 - A sale item shall be requested by a registered user.
 - A sale item shall be taken down by the registered user that posted it for sale.
 - When a sale item is taken down, all registered users who made a request for the item shall be notified of the closure by email.
- Housing Listing
 - A housing listing shall be posted by a registered user.
 - A housing listing shall be taken down by the registered user that posted it.
 - A housing listing shall be requested by many registered users.
 - A housing listing shall have a form to fill out for interested users.
 - A housing listing shall have a situation(available/close).
 - When a housing listing is taken down, all registered users who made a request shall be notified of the closure by email.
- Housing Listing Form
 - A housing listing form shall be filled out by a registered user.
 - A housing listing form shall fill out their full name.
 - A housing listing form shall fill out their school email.
 - A housing listing form shall fill out their phone number.
 - A housing listing form shall have a section dedicated to “an about” me.
 - A housing listing form shall have an expected day they want to move in.
- Restaurant
 - A restaurant shall have many restaurant reviews.
 - A restaurant shall show other related restaurants.
- Restaurant Request
 - A restaurant request shall be created by one registered user.
 - A restaurant request shall have the name of the restaurant.
 - A restaurant request shall have the location of the restaurant.
 - A restaurant request shall be confirmed/denied and closed by one and only one super user.
- Restaurant Review
 - A restaurant review shall be created by a registered user.
 - A restaurant review shall show the name of the registered user that posted it.
- Announcement
 - An announcement shall be posted by a registered user with an approved admin account.
 - An announcement shall be able to be viewed by one or many users.

- An announcement shall be able to be taken down by the one who posted it.
- An announcement shall be categorized by the type of admin account of the registered user that posted it.

Usability Test Plan

Test Objectives:

What is being tested is our special feature which is our housing feature, and the functions in it that we are testing are:

- a)** Posting a Housing listing
- b)** Deleting a Housing listing
- c)** Requesting a Housing listing
- d)** Editing a Housing listing
- e)** Searching for a Housing listing

We are testing these five major functions because they are all related to our special feature for housing. The reason these five functions were selected were because these are all the main processes one would go through when going through a housing listing site. Furthermore, these five functions represent what someone would select when going through the process of trying to find housing.

Test Description:

a) System Setup:

How the tests were set up for our participants was that they were placed in the same room as the observer, running the tests on the observers computer. Before the observer started the test for the participant, the observer had the participants read the task tables featured in the **Usability Tasks, Task Table** section. While the participants would take the test, the observer recorded any feedback they had for the task and put them in the **Usability Tasks, Usability Test Table** section.

b) Starting point:

The starting point for all of the above tests is the Housing Page in our site. By having the starting point at the housing page, we will not have to account for the time used to register and login their account, and will make the time used to complete their tasks more accurate.

c) Intended Users:

The Intended Users of our website are SFSU students, as they would need a SFSU email to register for an account, which would then allow them to use most of our major functions in our product.

d) URL of the system to be tested:

The URL of the systems to be tested is:

<https://www.gator-connection.com/housing>

The above URL relates to our Housing Page, which is our special feature.

e) Measuring User Satisfaction:

What we are measuring is the user satisfaction for our five functions that we have chosen. How we are measuring them is using the Lickert scale questions shown at the **Questionnaire** shown further in this section.

Usability Task Description:

a) Task Tables:

The below tables show the task tables for the five functions of posting, deleting, requesting, editing, and searching a housing listing.

Task	Description
Task	Post a housing listing
Machine state	Housing page(already logged in/registered/verified)
Successful completion task	Housing post listed
Benchmark	Completed in 30 sec

Task	Description
Task	Delete a housing listing
Machine state	Housing page(already logged in/registered/verified)
Successful completion task	Housing post is deleted
Benchmark	Completed in 30 sec

Task	Description
Task	Request a housing listing
Machine state	Housing page(already logged in/registered/verified)
Successful completion task	Housing post is successfully requested
Benchmark	Completed in 30 sec

Task	Description
Task	Edit a housing listing
Machine state	Housing page(already logged in/registered/verified)
Successful completion task	Housing post is successfully edited
Benchmark	Completed in 30 sec

Task	Description
Task	Search for a housing listing
Machine state	Housing page(already logged in/registered/verified)
Successful completion task	Housing posts searched by search criteria
Benchmark	Completed in 30 sec

b) Usability Test Tables:

The below Test Tables were tested with 2 participants, outlined as **Participant 1** and **Participant 2**.

i) Participant 1:

Test/Use Case	%Completed	errors	comments	Average time
Post a Housing Listing	100%	First Attempt: Set housing price to be too expensive, but frontend validation did not notify user. After post housing listing failed, user was confused about why post had failed and did not notice the error message at the top of the screen.	<ul style="list-style-type: none"> - Form validation should have stopped user before submitting, indicating that price was too much. - On post fail, user complained that the post housing form modal was not reopened with the values that he/she had already inputted saved. On fail, modal should be opened with values already inputted. - User suggested instead of having a set price, have a price range that a user is willing to sell for. - User also believed that form fields are not good when posting an apartment. 	2 minutes and 30 seconds
Delete a Housing Listing	100%	None	Very easy. NOTE: Did not test delete housing notification for interested users.	5 seconds

Request a Housing Listing	50%	<p>Send Housing Request Failed Because Housing Listing Post that was requested did not have a corresponding account. Because the housing listing post was not deleted automatically when the account was deleted.</p>	<p>Is it necessary to have graduation date? What if someone was just looking for a house?</p> <p>NOTE: Did not test request housing notification for user that posted.</p>	
Edit a Housing Listing	100%	None	Everything was straightforward. User believed it was useful. User believes a last edited timestamp would be useful to other users who are searching for housing listings.	30 seconds
Search for a Housing Listing	100%	None	User was confused by Home Page Housing filter. Did not know that he/she could change which category to filter by within the Housing Section. After being redirected from the home page main search bar which only had “housing” as the filter, no categories. User also believed that all of the information such as “Pets Allowed”, “Preferred Payment	15 seconds

			Type”, and “Address” should all be shown on the housing cards so that users do not have to click to see the details for every housing post. User also suggested adding a number of bathrooms/bedrooms indicator.	
--	--	--	---	--

ii) Participant 2:

Test/Use Case	%Completed	errors	comments	Average time
Post a Housing Listing	100%	None	<p>User complained that the “Close” form button was too close to the “Submit” button so he/she accidentally closed the form without submitting and got really confused.</p> <p>User complained that format for price was not necessary to force him/her to add the decimal places because housing prices are always rounded to the nearest dollar.</p> <p>User had difficulty figuring out how to get to the housing section page from the home page. User instead got lucky with home page search bar because the filter is automatically set to housing so it redirected him/her to housing page luckily.</p>	4 minutes and 40 seconds
Delete a Housing Listing	100%	None	<p>User thought email notification notifying him/her that a housing request he/she was interested in was closed was satisfactory. The user wants to have a dedicated page where all of the user’s posts are held in like a user’s profile, so that he/she can easily find his/her post and delete it without needing to search for it in the housing section.</p>	2 minutes

Request a Housing Listing	75%	<p>System crashed.</p> <p><i>IntegrityError at /housing_request/21 (1062, "Duplicate entry '50-21' for key 'Housing Listing Request.PRIMARY'"")</i></p> <p>However, notification email was still received for some reason for the user who posted the housing listing.</p>	<p>User was able to use the main home page search bar and found the housing listing post that she wanted.</p> <p>User also played the role of the user that posted the housing listing post being requested. User had a hard time navigating to the notification page. Firstly, the email did not have a link to the notification page which could have made the process a lot faster. User went to the housing listing post that he/she posted and did not see any buttons indicating that a request was made. In the navbar, there was no red notification indicator showing how many notifications the user had so it was hard to even tell that there were notifications without the email notification. User was also confused on how he/she could contact the person that was interested in the ad. The email notification did not direct the user to email the interested user so the user was confused. The email also didn't include the email of the interested user, so the user had to go back to the website and go to the notifications page where he/she could find the email.</p>	<p>Request Housing: 2 minutes and 45 seconds.</p> <p>Request Housing Notification: 3 minute and 30 seconds.</p>
----------------------------------	-----	--	---	---

Edit a Housing Listing	100%	None	<p>User was able to find the housing page through the navbar this time. Was able to find his/her housing listing post that he/she just posted, but because there aren't many posts, he/she did not need to use search feature.</p> <p>The user wants to have a dedicated page where all of the user's posts are held in like a user's profile, so that he/she can easily find his/her post and delete it without needing to search for it in the housing section.</p>	45 seconds
Search for a Housing Listing	100%	None	<p>User complained that the housing cards did not show the address of the housing so user had to click through every post to find a place with a suitable address.</p> <p>User also suggested to look at realtor.com to see how they show housing posts.</p>	30 seconds

Questionnaire:

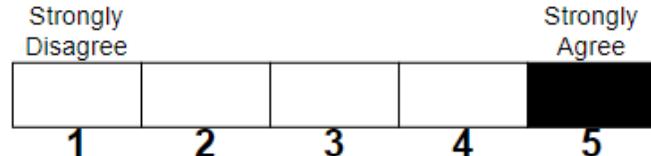
The below Test Tables were tested with 2 participants, outlined as **Participant 1** and **Participant 2**. (Participants are the same from the Test Tables)

i) Participant 1:

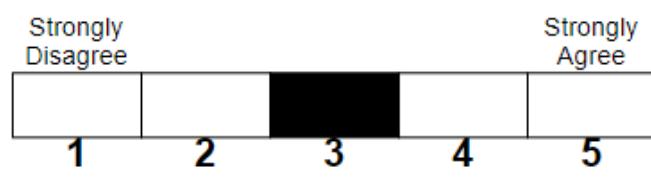
Gender: Female	Age Range: 18-24yrs old																		
1) Post Housing:																			
i) It was easy to input all of the necessary information to post a housing listing:	<table><tr><td>Strongly Disagree</td><td></td><td></td><td></td><td></td><td>Strongly Agree</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td></td></tr></table>	Strongly Disagree					Strongly Agree							1	2	3	4	5	
Strongly Disagree					Strongly Agree														
1	2	3	4	5															
ii) It was easy to post a housing listing:	<table><tr><td>Strongly Disagree</td><td></td><td></td><td></td><td></td><td>Strongly Agree</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td></td></tr></table>	Strongly Disagree					Strongly Agree							1	2	3	4	5	
Strongly Disagree					Strongly Agree														
1	2	3	4	5															
iii) The form provided allowed me to put all necessary information to sell a listing:	<table><tr><td>Strongly Disagree</td><td></td><td></td><td></td><td></td><td>Strongly Agree</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td></td></tr></table>	Strongly Disagree					Strongly Agree							1	2	3	4	5	
Strongly Disagree					Strongly Agree														
1	2	3	4	5															
2) Delete Housing:																			
i) It was easy to delete my housing listing:	<table><tr><td>Strongly Disagree</td><td></td><td></td><td></td><td></td><td>Strongly Agree</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td></td></tr></table>	Strongly Disagree					Strongly Agree							1	2	3	4	5	
Strongly Disagree					Strongly Agree														
1	2	3	4	5															
ii) The delete housing notification was a useful confirmation:	<table><tr><td>Strongly Disagree</td><td></td><td></td><td></td><td></td><td>Strongly Agree</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td></td></tr></table>	Strongly Disagree					Strongly Agree							1	2	3	4	5	
Strongly Disagree					Strongly Agree														
1	2	3	4	5															
iii) The delete housing notification/email was necessary:	<table><tr><td>Strongly Disagree</td><td></td><td></td><td></td><td></td><td>Strongly Agree</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td></td></tr></table>	Strongly Disagree					Strongly Agree							1	2	3	4	5	
Strongly Disagree					Strongly Agree														
1	2	3	4	5															

3) Request Housing:

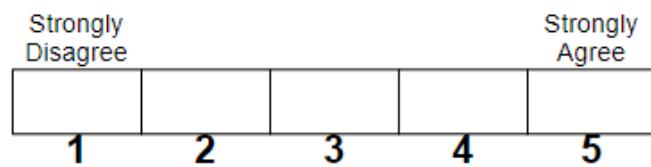
i) The process to send a housing request was simple and quick:



ii) The housing request form that I filled out was easy to fill out:

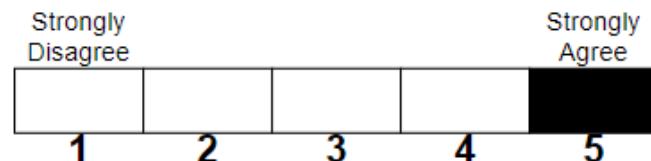


iii) The housing request email was easy to understand for the receiver:

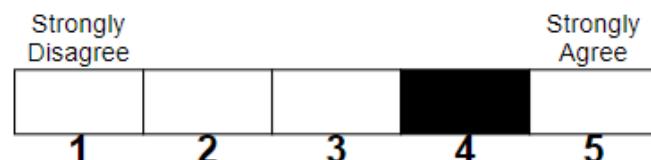


4) Edit Housing:

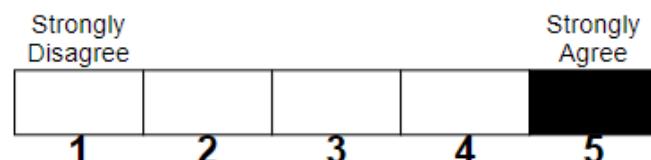
i) It was easy to edit my housing listing post:



ii) Adding more images to the listing was helpful:

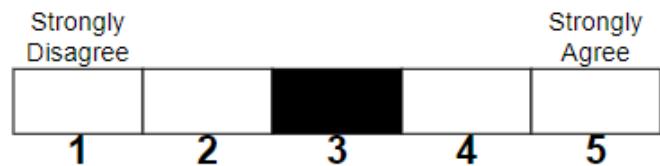


iii) It was helpful having my previous information filled out:



5) Search Housing:

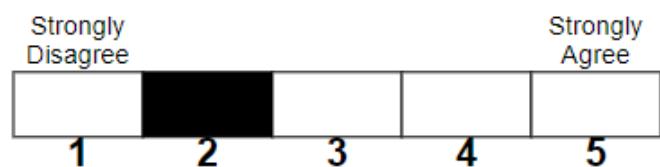
i) The search bar was easy to use:



ii) I was able to find the housing listing I wanted:



iii) The information provided on the housing cards was useful:



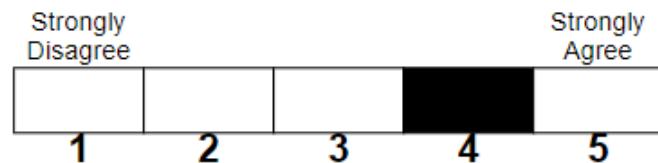
ii) Participant 2:

Gender: Female

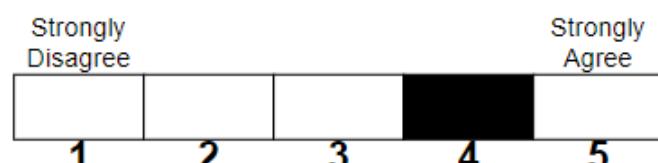
Age Range: 45-54yrs old

1) Post Housing:

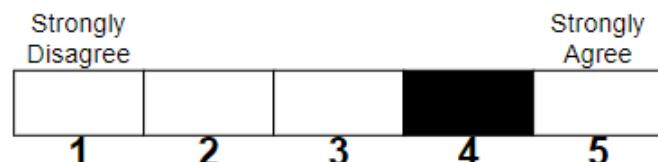
- i) It was easy to input all of the necessary information to post a housing listing:



- ii) It was easy to post a housing listing:



- iii) The form provided allowed me to put all necessary information to sell a listing:

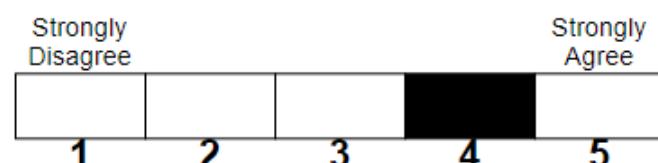


2) Delete Housing:

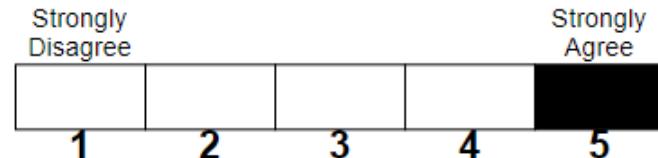
- i) It was easy to delete my housing listing:



- ii) The delete housing notification was a useful confirmation:

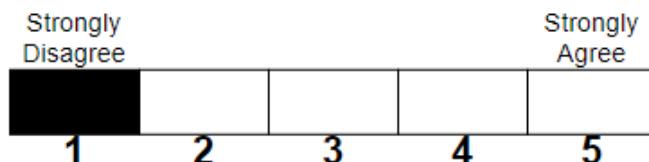


- iii) The delete housing notification/email was necessary:

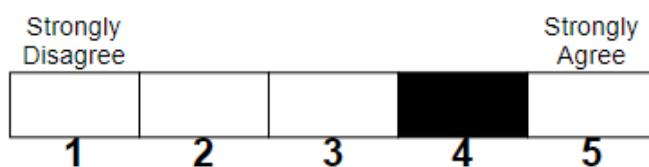


3) Request Housing:

i) The process to send a housing request was simple and quick:



ii) The housing request form that I filled out was easy to fill out:



iii) The housing request email was easy to understand for the receiver:

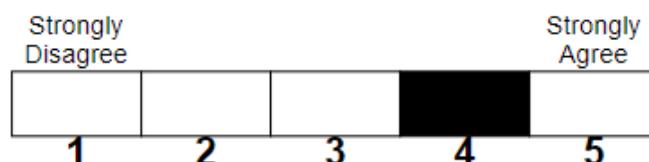


4) Edit Housing:

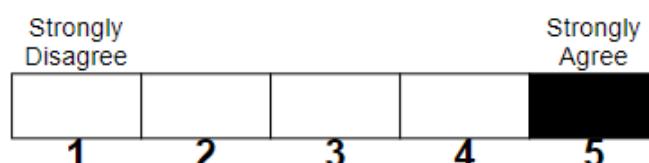
i) It was easy to edit my housing listing post:



ii) Adding more images to the listing was helpful:

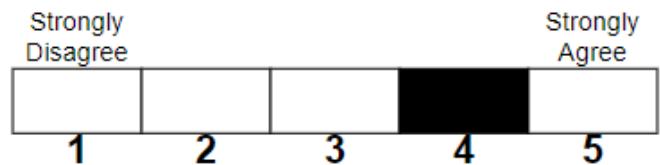


iii) It was helpful having my previous information filled out:

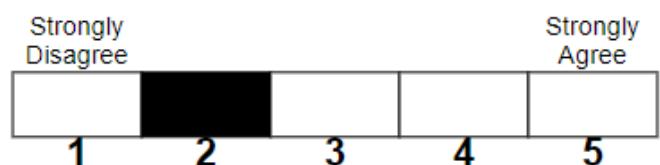


5) Search Housing:

i) The search bar was easy to use:



ii) I was able to find the housing listing I wanted:



iii) The information provided on the housing cards was useful:



QA Test Plan

Test Objectives:

In our QA tests, we will be testing these 5 nonfunctional requirements:

- A. A verification email shall be sent to a registered user's email.
- B. A restaurant from a restaurant request shall be verified by a super user in order to be added to the restaurant catalog.
- C. Only registered users shall be allowed to post housing listings.
- D. Only registered users shall be allowed to post restaurant reviews.
- E. Only registered users shall be allowed to make a purchase request for housing listings.

Features to be tested:

The features that are being tested are as follows:

1. Email Verification Feature
2. Restaurant Request Feature
3. Housing Post Feature
4. Restaurant Review Feature
5. Housing Request Feature

Below, we have listed how these features relate to the nonfunctional requirements we are testing:

- The **Email Verification Feature** is related to **Test Objective A**.
- The **Restaurant Request Feature** is related to **Test Objective B**.
- The **Housing Posting Feature** is related to **Test Objective C**.
- The **Restaurant Review Feature** is related to **Test Objective D**.
- The **Housing Request Feature** is related to **Test Objective E**.

QA Test Plan:

The below shows the table used for the QA Test Plan:

Task	Description	Test Input	Expected Output	PASS/FAIL
1	Test having verification email sent to registered user's email	User tests registering for an account	Verification email sent to their account	Google Chrome: Firefox: Safari:
2	Test having a restaurant request verified by a super user	User tests requests a restaurant, with a super user approving it	Restaurant appears after super user verifies it	Google Chrome: Firefox: Safari:
3	Test that only registered users can post house listings	User tests to try posting a listing on housing page	Their listing page appears on the housing page	Google Chrome: Firefox: Safari:
4	Test that only registered users can post restaurant reviews.	User tests to try posting a restaurant review for a restaurant.	Their review appears on the restaurant detail page.	Google Chrome: Firefox: Safari:
5	Test that only registered users shall be allowed to make a purchase request	User tries to make a purchase request	User is allowed to make a purchase request	Google Chrome: Firefox: Safari:

QA Test Plan #1:

Hardware:

Laptop: Dell XPS 15 9570

CPU: i7-8570H

GPU: Nvidia Geforce GTX 1050 TI Max-Q

Software:

OS: Windows 10 Home

Browser: Firefox Version 88.0.1

QA Test Case #1:

Task	Description	Test Input	Expected Output	PASS/FAIL
1	Test having verification email sent to registered user's email	<u>Input:</u> First Name: Benjamin Last Name: Kao SFSU Email: bkao1@mail.sfsu.edu Graduation Date: 05-22-2021 Password: Testing123!	Verification email sent to user's email account that he/she registered with that contains a link to Gator Connection that will verify the user's email.	Pass
2	Test having a restaurant request verified by a super user	<u>Request Restaurant Input:</u> Restaurant Name: Marugame Udon Zip Code: 94132 Number: 3251 Street: 20th Ave Space 184 City: San Francisco Open: 11:00 AM Close: 09:00 PM Takeout Available: Yes Description: Hip spot where an array of udon dishes are prepared with noodles hand-pulled in an open kitchen. Upload Image: marugame.jpg <u>Super User Input:</u> Click "Accept"	Restaurant appears with all of the inputted information from the Request Restaurant Form in the Restaurant section after the Super User verifies it.	Failed, the image is not being rendered correctly, but all of the other expected output passes.
3	Test that only	<u>Input:</u>	User's housing listing post	Pass

	registered users can post house listings	<p>Title: QA Test Plan 1 Price: 20.00 Zip Code: 94132 Number: 200 Street: Main Street City: San Francisco Description: This is the first QA Test Plan for Milestone 4. This test plan is for Firefox. Preferred Method of Payment: Check Pets Allowed: No Upload Images: unocards.png</p>	appears on the housing page with all of the inputted information the exact same.	
4	Test that only registered users shall be allowed to post restaurant reviews.	<p>A registered user with a Student account goes to any restaurant detail page and clicks on the “Write a Review” button.</p> <p><u>Restaurant Review Form Inputs:</u> Title: QA Test Case #3 Test 4 Rating: 4 Review: This is the review description for a restaurant review which we are testing in QA Test Case #3 Test 4.</p>	The user shall have the restaurant detail page reloaded with the review he/she had just written showing in the “Reviews” section of the restaurant detail page, along with their name. Also, the overall rating shall be updated accordingly to take into account the new rating that was just added with the review. Also, an alert message notifying the user that his/her restaurant review had been successfully posted should pop up at the top of the page.	Pass
5	Test that only registered users shall be allowed to make a purchase request	<p>Student goes to the Housing Page section, finds any Housing Listing Post that isn't his/her own and makes a request.</p> <p><u>Housing Request Form Inputs:</u> First Name: Benjamin Last Name: Kao Graduation Date: 05/22/2021 Tell us about yourself: This is QA test case #2, test 5 about testing if registered users can make a</p>	Student can see “Interested” button in the Housing Listing Detail and Item Listing Detail page. The student is allowed to make a purchase request for both housing and item listings. After the student submits the purchase request, they are redirected back to the detailed post page they were on with an	Fail, all of the expected output passes, but in addition to the expected output, an email notification was sent to the user that sent the request which is not supposed to happen.

		<p>purchase request or in this case, a housing listing interested request.</p>	<p>alert message, alerting them that their request was successfully sent. Also, an email notification and notification is sent to the owners of the posts notifying them that a user is interested.</p>	
--	--	--	---	--

QA Test Case #2:

Task	Description	Test Input	Expected Output	PASS/FAIL
1	Test having verification email sent to registered user's email	User tests registering for a student account. <u>Input:</u> First Name: Benjamin Last Name: Kao SFSU Email: benjamin.kao00@gmail.com Graduation Date: 05-22-2021 Password: 1234567	The user cannot submit the registration form because the email inputted is not a valid SFSU email.	Pass
2	Test having a restaurant request verified by a super user	<u>Restaurant Request Form</u> <u>Inputs:</u> Restaurant Name: Los Kuyas Zip Code: 94132 Number: 3251 Street: 20th Ave Space 184 City: San Francisco Open: 11:00 AM Close: 09:00 PM Takeout Available: Yes Description: aaaaaa Upload Image: 9931.jpg <u>Super User Input:</u> Click "Fail"	Restaurant does not appear in the restaurant section. The registered user who made the restaurant request should be given a notification alerting him/her that his/her restaurant request was rejected.	Failed. The restaurant does not appear in the restaurant section, but no notification was sent to the registered user that made the restaurant request.
3	Test that only registered users can post house listings	<u>Housing Listing Form:</u> Title: QA Test Plan 2 Price: 500000000000.00 Zip Code: 94116 Number: 200 Street: Main Street City: San Francisco Description: This is the first QA Test Plan for Milestone 4. This test plan is for Google Chrome Preferred Method of Payment:	Registered User is redirected back to the housing page with an alert message notifying the user that the price of the housing listing is too much and must be lowered.	Pass

		Check Pets Allowed: No Upload Images: 99310.jpg		
4	Test that only registered users shall be allowed to post restaurant reviews.	User tries to post a restaurant review on the restaurant section, while not logged in.	The user does not see the “Write a Review” button on the restaurant detail page. Instead sees text saying “You must be logged in to post a review” in place of the button	Pass
5	Test that only registered users shall be allowed to make a purchase request	Student tries to make a purchase request, while not logged in	Student that isn’t logged in cannot see the “Interested” request button, and therefore, cannot make a purchase request. Instead, there is text alerting the user: “Please log in to request this listing”	Pass

QA Test Case #3:

Task	Description	Test Input	Expected Output	PASS/FAIL
1	Test having verification email sent to registered user's email	<u>Input:</u> First Name: Benjamin Last Name: Kao SFSU Email: bkao1@mail.sfsu.edu Graduation Date: 05-22-2021 Password: Testing123!	The user should be redirected to whatever page he/she was on with an error message alerting the user that the email that he/she tried registered has already been taken and to try another email.	Pass
2	Test having a restaurant request verified by a super user	A super user who has not been approved goes to the notification/request page.	The super user should be able to get to the notification/request page, but should be alerted that he/she must be approved by another super user before being able to see Account Requests and Restaurant Requests. As a result, the restaurant requested does not show up in the restaurant section.	Failed. A super user that has not been approved by another super user was able to see account and restaurant requests, as well as, accept and reject them.
3	Test that only registered users can post house listings	An admin goes to the housing page. Presses "Post an Advertisement". <u>Housing Listing Form Inputs:</u> Title: QA Test Case 2 Price: 2020.00 Zip Code: 94116 Number: 200 Street: Main Street City: San Francisco Description: This is the second QA Test Case for Milestone 4. Preferred Method of Payment: Check	The user should be redirected to the housing page with an error message alerting the user that the file uploaded was not an image and the post housing listing failed.	Fail. The post housing listing succeeded.

		Pets Allowed: No Upload Images: test.txt		
4	Test that only registered users shall be allowed to post restaurant reviews.	<p>A super user shall go to any restaurant detail page and click the “Write a Review” button.</p> <p><u>Restaurant Review Form Inputs:</u></p> <p>Title: QA Test Case #3 Test 4</p> <p>Rating: 4</p> <p>Review: This is the review description for a restaurant review which we are testing in QA Test Case #3 Test 4.</p>	The user shall have the restaurant detail page reloaded with the review he/she had just written showing in the “Reviews” section of the restaurant detail page. Also, the overall rating shall be updated accordingly to take into account the new rating that was just added with the review. Also, an alert message notifying the user that his/her restaurant review had been successfully posted should pop up at the top of the page.	Pass
5	Test that only registered users shall be allowed to make a purchase request	<p>Student goes to the Housing Listing Page, finds his/her own post and goes to the detail page to make a purchase request.</p> <p>Student goes to the ItemListing Page, finds his/her own post and goes to the detail page to make a purchase request.</p>	For each case here, the student should not be able to see the “Interested” request button. Instead, the student should see two buttons, one that allows him/her to edit his/her post, and a “Delete” post button.	Pass

QA Test Plan #2:

Hardware:

CPU:AMD Ryzen 7 3700X

GPU: Nvidia Geforce RTX 2070 TI Super

Software:

OS: Windows 10 Home

Browser: Google Chrome version 90.0.4430.93

QA Test Case #1:

Task	Description	Test Input	Expected Output	Pass/Fail
1	Test having verification email sent to registered user's email	User tests registering for a student account. <u>Input:</u> First Name: Alec Last Name: Tenefrancia SFSU Email: alectenefranica2@gmail.com Graduation Date: 05-22-2021 Password: 1234567	Verification email sent to user's email account that he/she registered with that contains a link to Gator Connection that will verify the user's email.	Fail Failed because the email was not an sfsu email, which disabled the register button.
2	Test having a restaurant request verified by a super user	User tests requests a restaurant, with a super user approving it. Restaurant Name: Los Kuyas Zip Code: 94132 Number: 3251 Street: 20th Ave Space 184 City: San Francisco	Restaurant appears after super user verifies it	Fail Failed because even though it passed all verification checks, the Super User clicked "Reject" which rejected the request, thus not making it show up on the restaurants page.

		<p>Open: 11:00 AM Close: 09:00 PM Takeout Available: Yes Description: aaaaaa Upload Image: 9931.jpg <u>Super User Input:</u> Click “Reject”</p>		
3	Test that only registered users can post house listings	<p>User tests to try posting a listing on housing page</p> <p><u>Input:</u></p> <p>Title: QA Test Plan 2 Price: 500000000000.00 Zip Code: 94116 Number: 200 Street: Main Street City: San Francisco Description: This is the first QA Test Plan for Milestone 4. This test plan is for Google Chrome Preferred Method of Payment: Check Pets Allowed: No Upload Images: 99310.jpg</p>	The housing listing was not posted, and the user was redirected back to the housing page with an alert notifying the user that the price of the housing listing was too much.	Pass
4	Test that only registered	User tries to post a restaurant review on the restaurant	The user does not see the “Write a Review” button	Pass

	users shall be able to post restaurant reviews	section, while not logged in.	on the restaurant detail page. Instead sees text saying “You must be logged in to post a review” in place of the button	.
5	Test that only registered users shall be allowed to make a purchase request	Student tries to make a purchase request, while not logged in	Student that isn't logged in cannot see the “Interested” request button, and therefore, cannot make a purchase request. Instead, there is text alerting the user: “Please log in to request this listing”	Pass

QA Test Case #2:

Task	Description	Test Input	Expected Output	Pass/Fail
1	Test having verification email sent to registered user's email	<p>User tests registering for a student account.</p> <p><u>Input:</u></p> <p>First Name: Alec Last Name: Tenefrancia SFSU Email: alectene@mail.sfsu.edu Graduation Date: 05-22-2021 Password: 1234567</p>	Verification email sent to user's email account that he/she registered with that contains a link to Gator Connection that will verify the user's email.	Pass
2	Test having a restaurant request verified by a super user	<p>User tests requests a restaurant, with a super user approving it.</p> <p>Restaurant Name: Los Kuyas Zip Code: 94132 Number: 3251 Street: 20th Ave Space 184 City: San Francisco Open: 11:00 AM Close: 09:00 PM Takeout Available: Yes Description: aaaaaa Upload Image: 9931.jpg <u>Super User Input:</u> Click "Accept"</p>	Restaurant appears after super user verifies it	Pass

3	Test that only registered users can post house listings	<p>User tests to try posting a listing on housing page</p> <p><u>Input:</u></p> <p>Title: QA Test Plan 2 Price: 500.00 Zip Code: 94116 Number: 200 Street: Main Street City: San Francisco Description: This is the second QA Test Plan for Milestone 4. This test plan is for Google Chrome Preferred Method of Payment: Check Pets Allowed: No Upload Images: 99310.jpg</p>	The housing listing was posted on the housing page.	Pass
4	Test that only registered users shall be able to post restaurant reviews	<p><u>Restaurant Review Form</u></p> <p><u>Inputs:</u></p> <p>Title: Restaurants 1 Rating: 4 Review: This is the review description for a restaurant review which we are testing in restaurants.</p>	The user sees the restaurant review posted on the restaurant page.	Pass

5	<p>Test that only registered users shall be allowed to make a purchase request</p>	<p>Student goes to the Housing Listing Page, finds his/her own post and goes to the detail page to make a purchase request.</p> <p>Student goes to the ItemListing Page, finds his/her own post and goes to the detail page to make a purchase request.</p>	<p>Student that is logged in can see the button request for the housing or item.</p>	Pass
---	--	---	--	------

QA Test Case #3:

Task	Description	Test Input	Expected Output	Pass/Fail
1	Test having verification email sent to registered user's email	User tests registering for a student account. <u>Input:</u> First Name: Alec Last Name: Tenefrancia SFSU Email: alectenefranica2@gmail.com Graduation Date: 05-22-2021 Password: 1234567	Verification email sent to user's email account that he/she registered with that contains a link to Gator Connection that will verify the user's email.	Fail Failed because the email was not an sfsu email, which disabled the register button.
2	Test having a restaurant request verified by a super user	User tests requests a restaurant, with a super user approving it. Restaurant Name: Los Kuyas Zip Code: 94132 Number: 3251 Street: 20th Ave Space 184 City: San Francisco Open: 11:00 AM Close: 09:00 PM Takeout Available: Yes Description: aaaaaa Upload Image: 9931.jpg <u>Super User Input:</u> Click "Reject"	Restaurant appears after super user verifies it	Fail Failed because even though it passed all verification checks, the Super User clicked "Reject" which rejected the request, thus not making it show up on the restaurants page.

3	Test that only registered users can post house listings	User tests to try posting a listing on housing page <u>Input:</u> Title: QA Test Plan 4 Price: 5000.00 Zip Code: 94116 Number: 200 Street: Main Street City: San Francisco Description: This is the third QA Test Plan for Milestone 4. This test plan is for Google Chrome Preferred Method of Payment: Check Pets Allowed: No Upload Images: 99310.jpg	The housing listing was posted on the housing page.	Pass
4	Test that only registered users shall be able to post restaurant reviews	<u>Restaurant Review Form</u> <u>Inputs:</u> Title: Restaurants 1 Rating: 4 Review: This is the review description for a restaurant review which we are testing in restaurants.	The user sees the restaurant review posted on the restaurant page.	Pass
5	Test that only registered users shall be allowed to make a purchase request	Student tries to make a purchase request, while not logged in	Student that isn't logged in cannot see the "Interested" request button, and therefore, cannot make a purchase request. Instead, there is text alerting the user: "Please log in to request this listing"	Pass

QA Test Plan #3:

Hardware:

CPU: Intel Core i5-5350U

GPU: Intel HD Graphics 6000

Software:

OS: macOS Catalina 10.15.4

Browser: Safari V14.0.1

QA Test Case #1:

Task	Description	Test Input	Expected Output	PASS/FAIL
1	Test having verification email sent to registered user's email	<u>Input:</u> First Name: Angelo Last Name: Reyes SFSU Email: areyes24@mail.sfsu.edu Graduation Date: 05-22-2021 Password: March-99	Verification email sent to user's email account that he/she registered with that contains a link to Gator Connection that will verify the user's email.	Pass
2	Test having a restaurant request verified by a super user	<u>Request Restaurant Input:</u> Restaurant Name: Test Zip Code: 94132 Number: 3251 Street: 20th Ave Space 184 City: San Francisco Open: 11:00 AM Close: 09:00 PM Takeout Available: Yes Description: Hip spot where an array of udon dishes are prepared with noodles hand-pulled in an open kitchen. Upload Image: marugame.jpg <u>Super User Input:</u> Click "Accept"	Restaurant appears with all of the inputted information from the Request Restaurant Form in the Restaurant section after the Super User verifies it.	Failed, the image is not being rendered correctly, but all of the other expected output passes.
3	Test that only registered users can post	<u>Input:</u> Title: QA Test House 3 Price: 1400.00	User's housing listing post appears on the housing page with all of the	Pass

	house listings	<p>Zip Code: 98090 Number: 123 Street: Test Street City: San Francisco Description: This is for the QA test Preferred Method of Payment: Venmo Pets Allowed: Yes Upload Images: IMG_3188.jpeg</p>	inputted information the exact same.	
4	Test that only registered users shall be allowed to post restaurant reviews.	<p>A registered user with a Student account goes to any restaurant detail page and clicks on the “Write a Review” button.</p> <p><u>Restaurant Review Form Inputs:</u> Title: QA Test Case #1 Test 4 Test Plan #3 Rating: 4 Review: This is the review description for a restaurant review which we are testing in QA Test Case #1 Test 4.</p>	The user shall have the restaurant detail page reloaded with the review he/she had just written showing in the “Reviews” section of the restaurant detail page, along with their name. Also, the overall rating shall be updated accordingly to take into account the new rating that was just added with the review. Also, an alert message notifying the user that his/her restaurant review had been successfully posted should pop up at the top of the page.	Pass
5	Test that only registered users shall be allowed to make a purchase request	<p>Logged Out user navigates to the Housing section, finds any Housing Listing Post, and tries to make a request.</p> <p>Logged Out user navigates to the Item section, finds any Item Listing Post, and tries to make a request.</p> <p><u>Housing Request Form Input:</u> First Name: Angelo Last Name: Reyes Graduation Date: 5/14/2021 Description: This is for my interest in our house. QA test</p>	Logged Out user cannot find “Interested” button. Instead, website shows text notifying user to “Please log in to request this housing listing/item listing”	Fail, all of the expected output passes, but in addition to the expected output, an email notification was sent to the user that sent the request which is not supposed to happen.

QA Test Case #2:

Task	Description	Test Input	Expected Output	PASS/FAIL
1	Test having verification email sent to registered user's email	User tests registering for a student account. <u>Input:</u> First Name: Angelo Last Name: Reyes SFSU Email: angelo.reyes@gmail.com Graduation Date: 05-22-2021 Password: 1234567	The user cannot submit the registration form because the email inputted is not a valid SFSU email.	Pass
2	Test having a restaurant request verified by a super user	<u>Restaurant Request Form</u> <u>Inputs:</u> Restaurant Name: Los Kuyas Zip Code: 94132 Number: 3251 Street: 20th Ave Space 184 City: San Francisco Open: 11:00 AM Close: 09:00 PM Takeout Available: Yes Description: aaaaaa Upload Image: 9931.jpg <u>Super User Input:</u> Click "Fail"	Restaurant does not appear in the restaurant section. The registered user who made the restaurant request should be given a notification alerting him/her that his/her restaurant request was rejected.	Failed. The restaurant does not appear in the restaurant section, but no notification was sent to the registered user that made the restaurant request.
3	Test that only registered users can post house listings	<u>Housing Listing Form:</u> Title: QA Test Plan 2 Price: 50000000000.00 Zip Code: 94116 Number: 200 Street: Main Street City: San Francisco Description: This is the first QA Test Plan for Milestone 4. This test plan is for Google Chrome Preferred Method of Payment:	Registered User is redirected back to the housing page with an alert message notifying the user that the price of the housing listing is too much and must be lowered.	Pass

		Check Pets Allowed: No Upload Images: 99310.jpg		
4	Test that only registered users shall be allowed to post restaurant reviews.	User tries to post a restaurant review on the restaurant section, while not logged in.	User does not see the “Write a Review” button.	Pass
5	Test that only registered users shall be allowed to make a purchase request	<p>Student goes to the Housing Page section, finds any Housing Listing Post that isn't his/her own and makes a request.</p> <p><u>Housing Request Form Inputs:</u> First Name: Angelo Last Name: Reyes Graduation Date: 05/22/2021 Tell us about yourself: This is QA test case #2, test 5 about testing if registered users can make a purchase request or in this case, a housing listing interested request.</p>	<p>Student can see “Interested” button in the Housing Listing Detail and Item Listing Detail page. The student is allowed to make a purchase request for both housing and item listings. After the student submits the purchase request, they are redirected back to the detailed post page they were on with an alert message, alerting them that their request was successfully sent. Also, an email notification and notification is sent to the owners of the posts notifying them that a user is interested.</p>	Pass

QA Test Case #3:

Task	Description	Test Input	Expected Output	PASS/FAIL
1	Test having verification email sent to registered user's email	<u>Input:</u> First Name: Angelo Last Name: Reyes SFSU Email: areyes24@mail.sfsu.edu Graduation Date: 05-22-2021 Password: Testing123!	The user should be redirected to whatever page he/she was on with an error message alerting the user that the email that he/she tried registered has already been taken and to try another email.	Pass
2	Test having a restaurant request verified by a super user	A super user who has not been approved goes to the notification/request page.	The super user should be able to get to the notification/request page, but should be alerted that he/she must be approved by another super user before being able to see Account Requests and Restaurant Requests. As a result, the restaurant requested does not show up in the restaurant section.	Failed. A super user that has not been approved by another super user was able to see account and restaurant requests, as well as, accept and reject them
3	Test that only registered users can post house listings	An admin goes to the housing page. Presses "Post an Advertisement". <u>Housing Listing Form Inputs:</u> Title: QA Test Price: 2020.00 Zip Code: 94116 Number: 200 Street: Main Street City: San Francisco Description: This is the second QA Test Case for Milestone 4. Preferred Method of	The user should be redirected to the housing page with an error message alerting the user that the file uploaded was not an image and the post housing listing failed.	Fail. The post housing listing succeeded.

		Payment: Check Pets Allowed: No Upload Images: test.txt		
4	Test that only registered users shall be allowed to post restaurant reviews.	<p>A super user shall go to any restaurant detail page and click the “Write a Review” button.</p> <p><u>Restaurant Review Form Inputs:</u></p> <p>Title: QA Test Case #3 Test 4</p> <p>Rating: 4</p> <p>Review: This is the review description for a restaurant review which we are testing in QA Test Case #3 Test 4.</p>	The user shall have the restaurant detail page reloaded with the review he/she had just written showing in the “Reviews” section of the restaurant detail page. Also, the overall rating shall be updated accordingly to take into account the new rating that was just added with the review. Also, an alert message notifying the user that his/her restaurant review had been successfully posted should pop up at the top of the page.	Pass
5	Test that only registered users shall be allowed to make a purchase request	<p>Student goes to the Housing Listing Page, finds his/her own post and goes to the detail page to make a purchase request.</p> <p>Student goes to the ItemListing Page, finds his/her own post and goes to the detail page to make a purchase request.</p>	For each case here, the student should not be able to see the “Interested” request button. Instead, the student should see two buttons, one that allows him/her to edit his/her post, and a “Delete” post button.	Pass

Coding Style

Coding Style

General Coding Style:

- We are using 4 spaces as indents, not tabs.
- We are making sure to add inline comments wherever code is not easily readable or understandable.

Front End Specific Coding Style:

- For JavaScript code, we are using camelcase for variables and methods.
- For HTML and CSS code, we are using lower case names with multi-worded names being separated with “-”.

Back End Specific Coding Style:

- For variables, we are using snakecase.
- For methods, we are using camelcase.
- We are making sure that variable and method names are descriptive of their function and the data they are holding.
- We are documenting methods with the expected data types for each parameter following Python’s reStructured Text documentation style.

Code Review

Internal Code Review:

For our internal code review, we decided that our front end team would review our back end team's submitted code, and our back end team would review our front end team's submitted code. The submitted code for internal review was for front end, our **housing.html** and **housing.css** and for back end, our **housing.py** file.

Front end:

The files that were reviewed for the front end section between our team are our **housing.html** and **housing.css** files. The below shows our conversations between our team members discussing reviewing the file.



Bikram Tamang

Thu 5/6/2021 2:25 PM

To: Alec Stephen Tenefrancia

Cc: Benjamin Patrick Kao; Jiaxin Yu; Angelo Gloria Reyes



housing.html

12 KB

Here is a code from the housing section of the Gator connection. Please provide feedback.

Thank you,

Bikram Tamang

BK

Benjamin Patrick Kao

Thu 5/6/2021 5:19 PM



To: Bikram Tamang; Alec Stephen Tenefrancia
Cc: Jiaxin Yu; Angelo Gloria Reyes



housingPeerReview.html

14 KB

Thank you for the housing.html file. I have updated the file to include my comments. You can find the general comments at the top of the file, while the specific comments are scattered around the file. Can you please send me any CSS or JavaScript files associated with this HTML file so I can review those as well too?

Thanks,

Benjamin Kao

...

BT

Bikram Tamang
Thu 5/6/2021 11:16 PM

To: Benjamin Patrick Kao
Cc: Angelo Gloria Reyes; Jiaxin Yu; Alec Stephen Tenefrancia



housing.css
16 KB

Hey Benjamin, Thank you for the feedback on housing.html, here is the housing.css file

...

[Reply](#) | [Reply all](#) | [Forward](#)

BK

Benjamin Patrick Kao
Fri 5/7/2021 12:44 AM

To: Bikram Tamang
Cc: Alec Stephen Tenefrancia; Lakshita Chugh; Carmen Denisse Paisano



housingPeerReview.css
19 KB

Thanks Bikram, I have also reviewed the housing.css file you just sent as well.

Thanks,

Benjamin Kao

...

[Reply](#) | [Reply all](#) | [Forward](#)

The below shows the summary of the reviewed code by our backend team:

housing.html

```
<!--  
M4 Code Review: housing.html  
  
Peer Reviewed By: Benjamin Kao  
Date: 05/06/21
```

General Comments and Review:

- You guys are missing the file header explaining what this file is for and what logic is contained in this file. This will help everyone who wants to find something to look at the top of the file and get a quick summary of what this file offers.
- The naming conventions for classes and IDs are sometimes inconsistent with some using "-"s and some using "_"s to separate multi-word names. However, the names themselves are descriptive of what they are for which is great.
- I personally think there should be more inline comments separating/explaining what the HTML code is supposed to render on the page for organization. And this will make it easier to scan through this document and easily find what you need to fix when there are bugs. Also just more inline comments in general.
- Speaking of organization, I really like how you guys organized the HTML code so that it is in order of how everything will be rendered on the page.
- I honestly prefer to have Javascript extracted into its own separate Javascript file. I think this way it is more sustainable and scalable, and it just organizes everything better.
- Because you guys are using Django templating logic, I think you should explain what you are doing with this logic when you use it so that you can come back later and easily understand what you are doing.

Overall, besides the missing file header and the slightly low amount of comments, I think this HTML file is good.

Specific Comments/Review are added using inline comments.

-->

housing.css

```
/*
M4 Code Review: housing.css
Peer Reviewed By: Benjamin Kao
Date: 05/06/21
```

General Comments and Review:

- You guys are missing the file header explaining what is in this file.
- I think there is a lack of inline comments, however there are some sections where there is a decent amount of comments. I think there needs to be more comments in general especially given how difficult CSS is to debug.
- I think there should be more comments that section off the CSS and the CSS should be organized into sections such that each section refers to a specific branch in the DOM.

For example, if you have a chat sidebar, any CSS that styles anything in this chat sidebar should be sectioned off using comments and labeled.

- There are some class names that are kind of confusing in what they are referring to, and it seems like there is some CSS that may be old because it does not refer to any

elements in the housing.html file, so I think that should be deleted and cleaned up.

- A lot of the generalized styling in tags such as * and body, I think should be extracted and put in the styles.css file because I think the styling done in these tags should be consistent across all web pages.

- One feature of CSS that I highly suggest is custom properties. These are basically CSS variables that you can access in descendants of any rules/tags that you define them in.

For example, if you create a color scheme using custom properties and place it in the * rule in your styles.css file, this can be accessed in any CSS files that are loaded after the styles.css file. This means you can make all web pages consistent and it means you only need to change values in one place and see how it affects the entire website.

Specific Comments/Feedback is done below in inline comments.

```
*/
```

Back end:

The files that were reviewed for the back end section between our team is our **housing.py** file. The below shows our conversations between our team members discussing reviewing the file.

 Benjamin Patrick Kao
Thu 5/6/2021 12:42 PM

To: Alec Stephen Tenefrancia
Cc: Bikram Tamang; Lakshita Chugh; Carmen Denisse Paisano

 housing.zip
3 KB

Hello Frontend Team,

Here is the Backend Team's housing.py Python code that contains the Housing feature. Please review the code and provide feedback and comments for us in the code file on things we should improve upon.

Thanks,

Benjamin Kao

 Bikram Tamang
Thu 5/6/2021 11:38 PM

To: Benjamin Patrick Kao
Cc: Angelo Gloria Reyes; Jiaxin Yu; Alec Stephen Tenefrancia

 reviewed_housing.py.zip
3 KB

Hey Backend Team, here is a reviewed version of the housing.py

Thanks,
Bikram Tamang

The below shows the summary of the reviewed code by our Front end team:

housing.py

```
# Milestone 4, Code Review:  
# Overall Review:  
    # Overall a very well written code. Almost all the  
    sections has comments explaining  
        # what it does following with meaningful variables.  
Since it is python, the code are  
    # very well indented and looks organized.  
    # the only thing i can suggests is to have some  
    comments on the imports telling what it  
        # is importing so it would be easier to know what some  
functions are doing.
```

Code Review with other teams:

How we did our code review with other teams was that a member of our back end team sent our **housing.py** file to **Team04** for review, and a member of our front end team sent our **housing.css** and **housing.html** files to **Team07** for review.

Front end:

The files that were reviewed for the front end section with **Team07** were our **housing.html** and **housing.css** files. The below shows the conversations between our team members and **Team07** discussing reviewing the file.

From: Bikram Tamang <btamang@mail.sfsu.edu>

Sent: Saturday, May 8, 2021 8:47 PM

To: Janson Jun Jie Leong <jleong6@mail.sfsu.edu>

Subject: Re: CSC648-04 Team 7 Code Review

Hello Janson, can you send me the file so i can leave comments like the professor asked.

Thanks,

Bikram

From: Bikram Tamang <btamang@mail.sfsu.edu>

Sent: Saturday, May 8, 2021 9:27 PM

To: Janson Jun Jie Leong <jleong6@mail.sfsu.edu>

Subject: Re: CSC648-04 Team 7 Code Review

here is the review. Janson. and i have also attached my front end code. please take a look and leave some review.

Thanks, Bikram

From: Janson Jun Jie Leong <jleong6@mail.sfsu.edu>

Sent: Sunday, May 9, 2021 3:05 PM

To: Bikram Tamang <btamang@mail.sfsu.edu>

Subject: Re: CSC648-04 Team 7 Code Review

Hello, I also forgot to ask which part you want code reviewed, as the professor stated that the code review should be one function/method.

From: Bikram Tamang <btamang@mail.sfsu.edu>
Sent: Sunday, May 9, 2021 3:06 PM
To: Janson Jun Jie Leong <jleong6@mail.sfsu.edu>
Subject: Re: CSC648-04 Team 7 Code Review

Hey Janson, whichever function you think we need to improve upon is fine.

From: Rodrigo Gallardo <rgallardo@mail.sfsu.edu>
Sent: Wednesday, May 12, 2021 6:51 PM
To: Bikram Tamang <btamang@mail.sfsu.edu>
Subject: Code Review

Hello Bikram,

I'm part of team 07 and I attacked the code review for the code you have provided.

Thanks for reviewing our code.

The below shows the summary of the reviewed code by **Team07** for our **housing.html** and **housing.css** file:

//code Reviewed by : **Rodrigo Gallardo (Front end Member - Team 07)**
//General Comment: **Code is organized and good overall.**
//Some Suggestions,
//provide some comments so it could be helpful for other programmers to understand the code better

Back end:

The files that were reviewed for the back end section with **Team04** is our **housing.py** file. The below shows our conversations between our team members and **Team04** discussing reviewing the file.

 Benjamin Patrick Kao Sat 5/8/2021 11:24 PM    

To: YANGESH KC; Zaid Saleh Alkhatib
Cc: Alec Stephen Tenefrancia; Angelo Gloria Reyes

 housing.zip
4 KB

Hello Team 04 Team Lead Yangesh and Backend Lead Zaid,

My name is Benjamin Kao, and I am a backend member for Team 05. I would like to request a code review for one of my team's backend code files. We have already completed an internal code review of this backend code file of our housing feature, and we would love to hear any feedback/criticisms you can give so we can improve. My apologies that this file is zipped up, we are coding with Python and sending non-zipped Python files is not supported through email.

Thanks,

Benjamin Kao

From: [YANGESH KC](#)

Sent: Sunday, May 9, 2021 11:42 PM

To: [Benjamin Patrick Kao](#); [Zaid Saleh Alkhatib](#); [Danish Hassan Siddiqui](#); [Mark Zulueta Jovero](#)

Subject: Re: CSC 648-04 Team 05: M4 Code Review Request

Hello Team 05,

Thank-you for your feedback. Our team has reviewed the code you provided. I have attached the file via this email. Feel free to reach out if you have any questions.

Thanks,

Yangesh KC

Team-04

The below shows the summary of the reviewed code by **Team04**:

housing.py

```
# Code Reviewed By: Danish Siddiqui - TEAM04 - FrontEnd Lead
""" Good detailed description on each function. I'm assuming
that your team categorized the database model and all exceptions
into their own files and are importing them here for the easy
access and for more scalability. Even though, i have personally
never worked with Python as a backend language, but this file's
syntax is easier to read and the logic is easier to follow
through. The only suggestion I would wanna give if it applies to
the Python way of programming is to maybe split get request
functions to their own file and keep housing data manipulation
functions to this file so as to make this file a little shorter
and precise to one request service. In any case, having all
requests, related to housing, together is good too.
"""

#Code Reviewed By: Mark and KC May 09
```

```
"We think that you need to add more inline comments to the
code. Overall the code is good for the housing part.
```

Backend Lead: Zaid

```
"""

Amazing comments and description for each function. I would
agree with Denish's comment about splitting the code into
smaller files to make it more readable, other than that,
everything looks perfect. Good job!
"""


```

Self-Check on Best Practices for Security

1. Major Protected Assets

- Emails

We make sure that users' emails are only sent to other users when the user makes an item or housing listing request for another user's post. Also, for Admin Accounts and Super User Accounts, these accounts have their email sent with their account requests in addition to the item and housing listing requests. Other than these exceptions, emails are not accessible by anyone.

- Passwords

We make sure that all passwords upon registration are first encrypted using BCrypt and then inserted into our database. These passwords are being stored as binary blobs as well and not plain text.

- Item Requests

We make sure that a user's item request will only be available to the user who posted the item.

- Housing Listing Requests

We make sure that a user's housing listing request will only be available to the user who posted the housing listing given how sensitive the "About Me" data from the Housing Form is.

2. Password Encryption (Provide Screenshots of MySQL Workbench and also of the debugger showing the encrypted passwords because we cannot see the data in MySQL Workbench because they are stored as BINARY BLOBS)

We have made sure to encrypt passwords before storing them on our MySQL database. We are using the popular password encryption algorithm, BCrypt, to encrypt the passwords in our database. After a guest user registers for one of our accounts after frontend password input validation, we take the password data and use Python's BCrypt module's built-in functions to encrypt this password data. After the BCrypt function finishes encryption, we store this encrypted password in the database. Below, we have provided screenshots of our code and MySQL Workbench showing how we encrypt passwords and the final result of how passwords are stored in our MySQL Workbench.

Password Encryption Code:

```
@staticmethod  
def encryptPassword(password):  
    return bcrypt.hashpw(password.encode(), bcrypt.gensalt(12))
```

MySQL Workbench Account Table Data:

Because the data type for passwords in our Account Table is BINARY(60), MySQL Workbench only shows passwords as blobs.

account_id	email	password	createdAt	email_verified	user
30	cpaisano@mail.sfsu.edu	BLOB	2021-05-06 18:35:29	0	119
31	alectenefranica2@gmail.com	BLOB	2021-05-06 21:58:43	1	125
32	sad@mail.sfsu.edu	BLOB	2021-05-07 00:36:03	0	130
35	alectenefranica2@gmail.com	BLOB	2021-05-07 02:19:31	0	134
36	bkao1@mail.sfsu.edu	BLOB	2021-05-07 19:40:23	1	96
39	alectene@mail.sfsu.edu	BLOB	2021-05-07 21:27:18	1	155
40	ldorje1995@mail.sfsu.edu	BLOB	2021-05-07 22:20:24	0	168
46	benjamin.kao00@gmail.com	BLOB	2021-05-07 23:40:00	1	174
47	bikram@mail.sfsu.edu	BLOB	2021-05-08 05:17:38	0	184
48	btamang@mail.sfsu.edu	BLOB	2021-05-08 05:27:33	1	185

Debugger Password:

Here is a screenshot of a debugger that shows that the password retrieved from our database is indeed encrypted.

The screenshot shows a debugger interface with a code editor and a variables panel. The code editor displays a Python function:

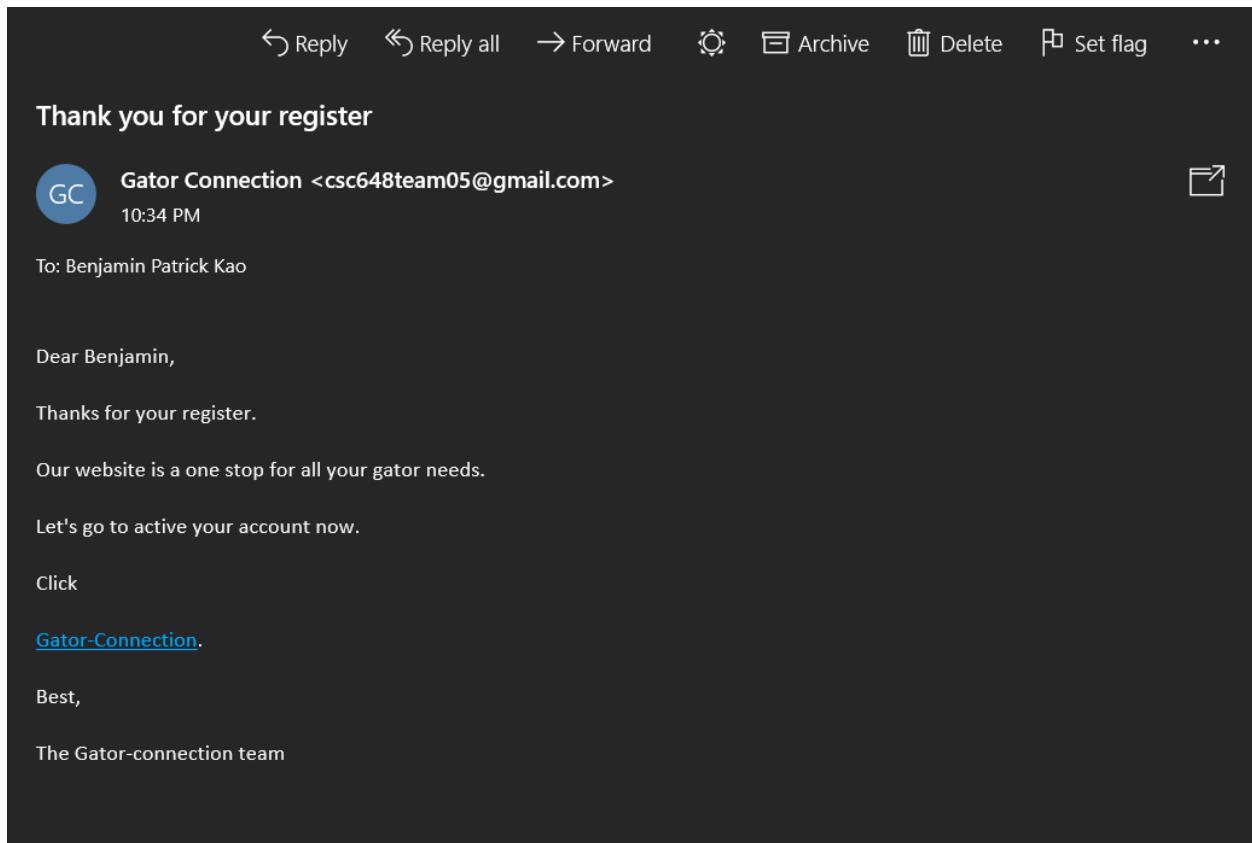
```
if bcrypt.checkpw(password.encode(), stored):  
    return True  
else:  
    return False
```

The variables panel shows two entries:

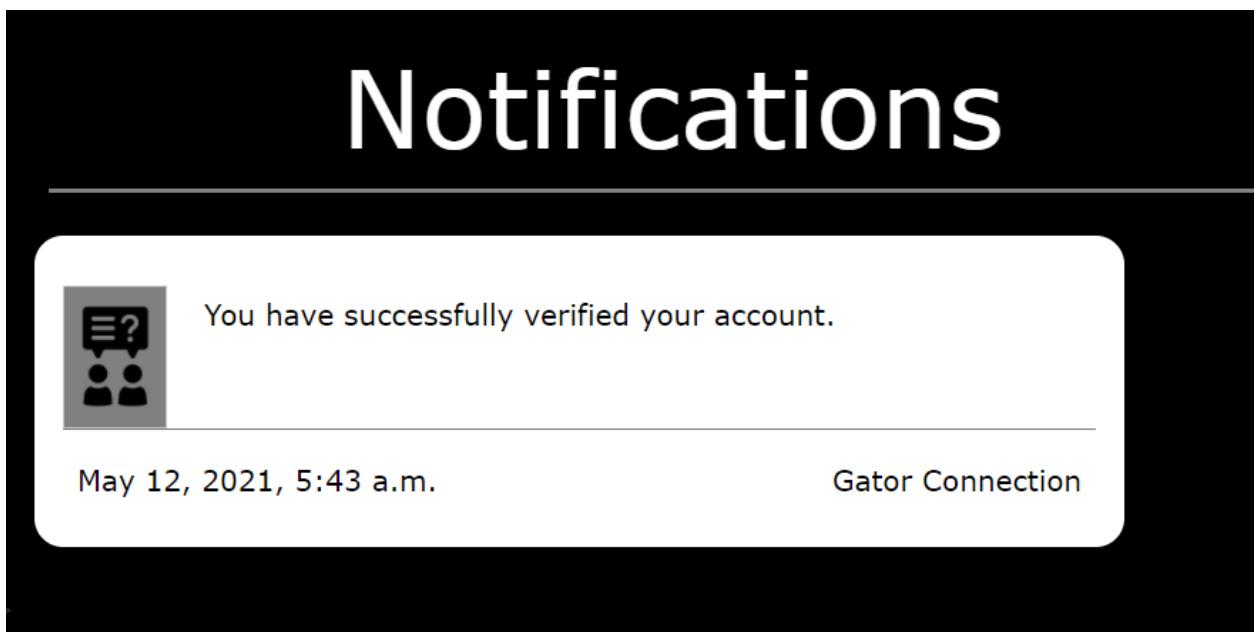
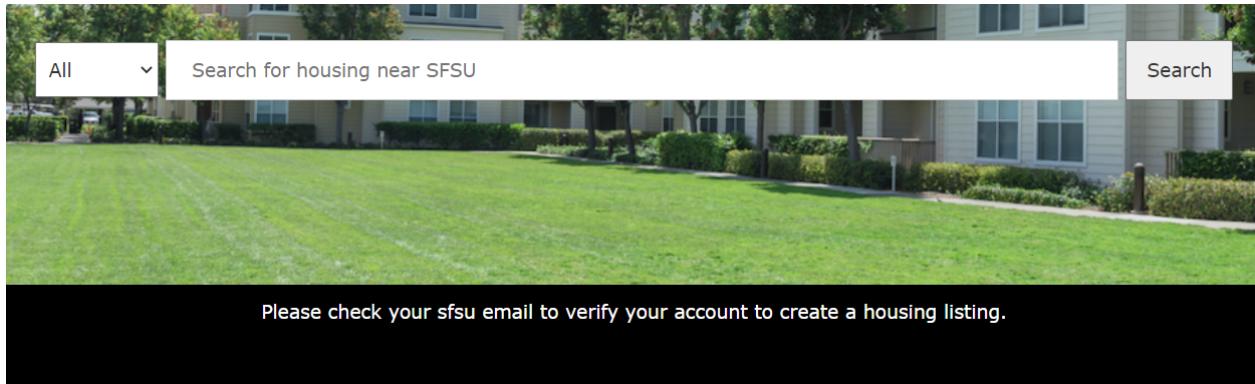
- password = {str} 'Testing123!'
- stored = {bytes: 60} b'\$2b\$12\$jMAQVfPQu3NrIcfOKaHLS.xi70Y1a5wPma.dezbK0Fl3glHEstl7y'

3. Email Verification

Our product's main target audience is San Francisco State University students and personnel. In order to verify that our users are who they say they are to protect the rest of our users, we send a verification email to every user that registers for an account. In addition to this, we prohibit registered users who have not verified their emails from using our services in order to protect the rest of our users. Finally, for users who have a Student Account, we require that they register with an SFSU email account. This registration restriction, complemented with our email verification, ensures that our Student Account registered users are students from SFSU. As a result, this adds an extra layer of security for our users.



Example of Non Verified Account:



4. Admin and Super User Account Verification By Super User

Although our product's main target audience is San Francisco State University students and personnel, some of our features such as posting announcements, are created for users who are a part of organizations/clubs at SFSU. Super User accounts are for those who are part of our team and are moderating our product. However, these organizations/clubs and Super Users are not always provided with SFSU emails. Therefore, we cannot enforce the SFSU email registration restriction onto Admin and Super User accounts. Thus, to counteract this possible security hole, in addition to the email verification security layer we have, we also only allow Admin and Super User accounts verified by a Super User to use the features that the accounts have.

Super User Account Request

May 11, 2021, 11:30 p.m.

Accept

Reject

cpaisano5@mail.sfsu.edu

CARMEN PAISANO

Admin Account Request

May 11, 2021, 11:09 p.m.

Accept

Reject

cpaisano1@mail.sfsu.edu

carmen paisano

Admin Account Request

May 11, 2021, 11:19 p.m.

Accept

Reject

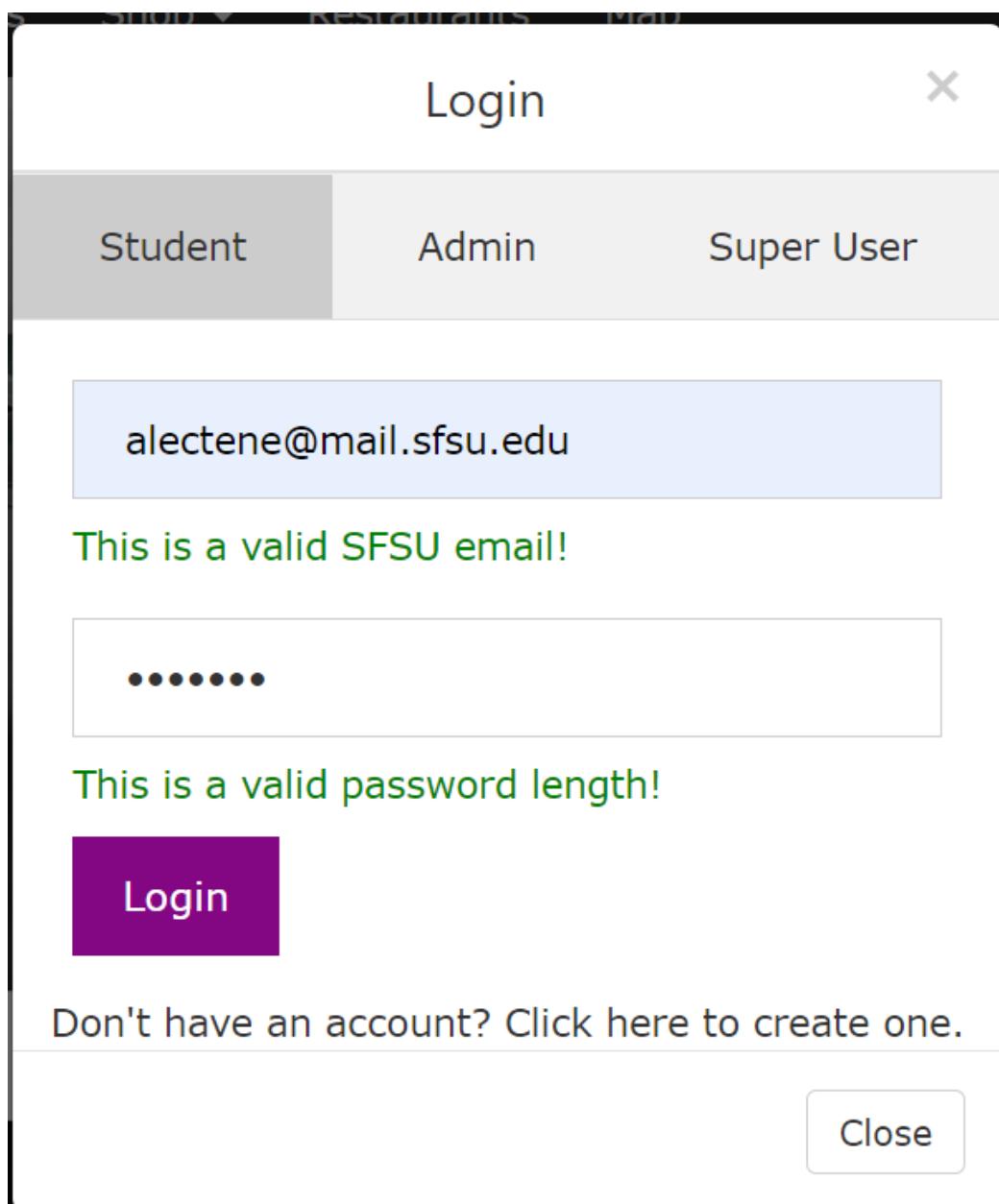
cpaisano2@mail.sfsu.edu

carmen paisano

5. Frontend Input Data Validation For Login/Register Forms

Login Form

- Email
- Password



Register Form

- First Name
- Last Name
- Email (SFSU Email for Student Accounts Only)
- Password
- Sport (Athletics Accounts Only)
- Organization Name (Departments Accounts Only)
- Department Name (Organizations Accounts Only)

Register X

Student	Admin	Super User
---------	-------	------------

By registering, you agree to Gator Connection's Terms and conditions.

Alec

Tenefrancia

alectene@mail.sfsu.edu

This is a valid SFSU email!

Graduation Date:

Thank you for putting in your graduation date!

.....

This password is great!

Register

Already have an account? [Click here to sign in.](#)

Close

Post Housing Listing Form

- Title
- Price
- Zip Code
- Street Number

Post a Listing

Title:

Title accepted

Alec's test house

Pricing (Format ex: 12.00): \$

350.00

Zip Code:

94116

Valid Zip-code

Number:

19

Valid number

Street:

valid address

1925 17th Ave

City:

City validated

San Francisco

Description:

Description looks good! 

This is a test description for housing

Preferred Method of Payment

Venmo 

Pets Allowed

 yes  no

Upload Images

IMG_3398.jpg

Post Item Listing Form

- Title
- Price

Post an Item for sale ×

Title:
Title accepted ✓

This is a test item

Pricing: (Format ex: 12.00): \$
✓

350.00

Description:
Description looks good! ✓

Hello, this is a test item

Preferred Method of Payment Venmo

Condition: Good Used

Upload Image Choose File IMG_3398.jpg

submit

Close

Example of Frontend Input Data Validation Code:

```
/* STUDENT REGISTER CHECK'S BEGIN */

function validate_sfsu_email()
{
    /*
        Function here validates for sfsu email by checking if the input includes
        string "@mail.sfsu.edu", if it does it returns true, if not false.
        Also, to check if people go out of order on the registering modal, there
        is also a check that checks if the other values from the fields are
        correct. If they are
        it will allow them to register.
    */

    if(document.getElementById("sfsu-email").value.includes("@mail.sfsu.edu") ||
    |document.getElementById("sfsu-email").value.includes("@sfsu.edu"))
    {
        document.getElementById("student_email_message").innerHTML = "<span
            style='color: green;'>This is a valid SFSU email!</span>";
        if((document.getElementById("sfsu-email").value.includes("@mail.sfsu.
            edu") ||document.getElementById("sfsu-email").value.includes("@sfsu.
            edu")) && document.getElementById("graduation-date").value.includes
            ("-") && document.getElementById("student_password").value.length >=
            6)
        {
            document.getElementById("student_register_button").disabled =
            false;
            return true;
        }

        return true;
    }
    else if(document.getElementById("sfsu-email").value.length >= 45)
    {
        document.getElementById("student_email_message").innerHTML = "<span
            style='color: red;'>Your input is too long, please reduce the amount
            </span>";
    }
}
```

6. Backend Input Data Validation For Login/Register Forms

Login Form

- Email
- Password

Register Form

- First Name
- Last Name
- Email (SFSU Email for Student Accounts Only)
- Password
- Sport (Athletics Accounts Only)
- Organization Name (Departments Accounts Only)
- Department Name (Organizations Accounts Only)

Post Housing Listing Form

- Title
- Price
- Zip Code
- Street Number

Post Item Listing Form

- Title
- Price

Examples of Backend Data Input Validation Code:

```
@staticmethod
def checkGeneralLoginFormValidation(email, password):
    email_length = len(email)
    password_length = len(password)

    if email_length > 45:
        raise InvalidFormError(email,
                               f"should be less than 45 characters. Current Length: {email_length}")
    if password_length > 45:
        raise InvalidFormError(password,
                               f"should be less than 45 characters. Current Length: {password_length}")

    if not (Authentication.checkEmailFormat(email)):
        raise InvalidFormError(email, "is not an email.")
```

```
@staticmethod
def checkEmailFormat(email):
    regex = r"\b[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,}\b"

    if re.search(regex, email, re.IGNORECASE):
        return True
    return False

@staticmethod
def checkUniqueEmail(email):
    # Check if email is unique
    try:
        emailCheck = Account.objects.get(email=email)

        if emailCheck:
            # Email already exists, throw unique email error
            raise UniqueEmailError(email)
    except Account.DoesNotExist:
        pass
```

Self-Check: Adherence to Original Non Functional Specs

- **Performance:**

1. The website should have a load time of no more than 4 seconds. **(DONE)**
2. Notifications to users shall be sent within 30 seconds of purchase requests to allow users to connect more quickly. **(ON TRACK)**

- **Marketing:**

3. All pages shall have the company logo in the upper left hand corner. **(DONE)**

- **Look and Feel:**

4. A navbar shall be at the top of all website pages. **(DONE)**
5. Restaurant reviews shall have the user that posted, optional images, rating, and comments. **(ON TRACK)**
6. Posted items shall have the user that posted, one required image and more optional images, and details about the item. **(ON TRACK)**
7. Housing listings shall have the user that posted, many required images, and details about the housing. **(ON TRACK)**

- **Scalability:**

8. The website shall be scalable and adapt optimally when the number of users and workloads increase. **(DONE)**
9. The website shall be scalable and maintainable as more features are added. **(ON TRACK)**

- **Recoverability:**

10. In times where the website faces problems or failure, components shall be easy to navigate and fixed by the administrator to recover the website. **(ON TRACK)**

- **Functionality:**

11. The website shall be developed and deployed using the tech stack approved by the class CTO. **(DONE)**

12. The website shall have UX/UI that is clear and helpful for users so they can navigate to other pages and find resources easily. **(ON TRACK)**

- **Security:**

13. A registered user's with a Student Account shall have his/her email verified by the system to be an SFSU email account and unique in order to sign up. **(DONE)**

14. A verification email shall be sent to a registered user's email. **(DONE)**

15. A registered user with an Admin Account or Super User Account shall be approved by a super user before being allowed to use the features specific to each account type. **(DONE)**

16. A restaurant from a restaurant request shall be verified by a super user in order to be added to the restaurant catalog. **(DONE)**

17. A registered user shall be notified when a new device has logged into their account. **(ISSUE)**

At first, we believed we could use a device's MAC address to implement this feature. However, after much research, we discovered that accessing a device's MAC address is not allowed. That being said, we have come up with a possible solution where we instead give a device a UUID to store locally.

Whenever a device logs in, we check if the device already has a corresponding UUID or not. Although this means users will receive false positives whenever a device deletes its locally stored UUID given by us.

18. Only administrative users shall be allowed to post announcements. **(DONE)**
19. Only registered users shall be allowed to post items for sale. **(DONE)**
20. Only registered users shall be allowed to post housing listings. **(DONE)**
21. Only registered users shall be allowed to make a purchase request for items that are on sale. **(DONE)**
22. Only registered users shall be allowed to make a purchase request for housing listings. **(DONE)**
23. Only registered users shall be allowed to post restaurant reviews. **(DONE)**

- **Privacy:**

24. A registered user's password shall be encrypted and saved in the database. **(DONE)**
25. A registered user's name shall be saved in the database. **(DONE)**
26. A registered user's email shall be saved in the database. **(DONE)**
27. A registered user's name shall be passed around with purchase requests and restaurant reviews. **(DONE)**
28. A registered user's email shall be passed around with only purchase requests. **(DONE)**

- **System Requirements:**

29. The website shall work up to Version 88.0.4324.182 of Google Chrome. **(ON TRACK)**

30. The website shall work up to Version 86 of Mozilla Firefox. **(ON TRACK)**
 31. The website shall work up to Version 14.0.3 of Safari. **(ON TRACK)**
 32. The website shall be compatible for both Windows 10 and MacOS. **(ON TRACK)**
- **Availability:**
 33. The website shall be fully functional at any time of the day whenever there is no maintenance. **(ON TRACK)**
 - **Capability/Expected Load:**
 34. The website shall support as many users as the AWS EC2 Instance can support. **(ON TRACK)**
 - **Legal:**
 35. A link to the terms and conditions shall be present at the bottom of all website pages. **(DONE)**
 36. A link to the privacy notice shall be present at the bottom of all website pages. **(ON TRACK)**
 - **Coding Requirements:**
 37. All code shall be well documented. **(ON TRACK)**
 38. All code shall have relevant variable and function names. **(ON TRACK)**
 39. All code shall be well organized and have the same code style. **(ON TRACK)**
 40. All code shall be indented using spaces. **(ON TRACK)**
 - **Environmental:**
 41. Local environments on developer machines shall mirror the AWS EC2 Instance environment. **(DONE)**

42. All development shall be done on the development branch or lower. **(DONE)**
43. The development branch shall be approved by Team Lead before being merged into the master branch. **(DONE)**
44. All GitHub commits shall have extensive details about code added and changed. **(ON TRACK)**
45. All code shall be extensively tested locally before being deployed on the AWS server. **(ON TRACK)**

- **Storage:**

46. Data inputted by the user during sign up shall be stored in the database. **(DONE)**
47. General users shall be stored in the database. **(DONE)**
48. General users that leave the website shall be removed from the database. **(ON TRACK)**
49. All announcements shall be stored in the database. **(DONE)**
50. All posts shall be stored in the database. **(DONE)**

- **Fault Tolerance:**

51. The website shall be able to refresh whenever there is a lost connection. **(DONE)**

List of Contributions

Jiaxin:

a) Backend:

Backend features that Jiaxin was tasked to work on was improving the email feature, specifically:

i) email_helper.py file:

In the email_helper.py file, Jiaxin implemented the email sending for whenever someone requests for an item, deletes their item, and posts their listing.

Furthermore, Jiaxin implemented the email verification feature, as when someone registers for an account, a verification email gets sent out to the email they registered with. Unless they go into the email and click the link, their account won't be verified, and they won't be able to use any of the functions of the website.

b) documentation:

Things for the M4 document that Jiaxin worked on specifically was helping the team decide for point 6) Self-check: Adherence to original Non-Functional Specs, which of our non-functional requirements are **done**, **on track**, or **issue**.

Carmen:

a) Frontend:

Frontend features that Carmen worked on was setting up the input validation for the modals that we have in :

- i) restaurants folder**
- ii) announcements folder**

In these files, Carmen set up the input validation for modals. For the modals, if the input was wrong, such as if it was too long of an input, or too short, then it would pop out a message saying something like “your input is wrong, please enter the correct value”. When the input would be correct, it would say “this input is valid” or something along the like. Furthermore, if any of the inputs would be wrong, it would not let them post with the modal.

b) documentation:

Things for the M4 document that Carmen worked on specifically was recording info for the test table for the usability test notes and having her actors fill out the questionnaire as well.

Bikram:**a) Frontend:**

Frontend features that Bikram worked on was setting up input validation for the modals that we have in :

- i) housing.html**
- ii) housing_detail.html**
- iii) item.html**
- iv) item_detail.html**

In these files, Bikram set up the input validation for modals. For the modals, if the input was wrong, such as if it was too long of an input, or too short, then it would pop out a message saying something like “your input is wrong, please enter the correct value”. When the input would be correct, it would say “this input is valid” or something along the like. Furthermore, if any of the inputs would be wrong, it would not let them post with the modal.

Furthermore, Bikram also worked on improving the **maps.html** page we had, by making the search bar bigger.

b) Documentation:

Things for the M4 document that Bikram worked on specifically was recording info for the test table for the usability test notes and having her actors fill out the questionnaire as well.

Furthermore, Bikram also helped on the code review section as he sent our front end code to another team and to our backend team to review it, as well as reviewing our backend code that our backend team sent.

Angelo:

a) Backend:

Backend Features that Angelo worked on was having the housing request features set up, such as sending the email out if someone requested a housing or deleted a housing, which is present in the files:

- i) housing.py**
- ii) housing_request.py**

Furthermore, Angelo also collaborated with Benjamin to have our “notifications” feature set up, which is a collection of notifications all saved to the user, which they can view, which is in the file:

- i) notifications.py**

b) Frontend:

Frontend Features that Angelo worked on specifically was collaborating with Benjamin to have the our notifications data render on the notifications page, which is found on the file:

- i) notifications.html**

c) Documentation:

Things for the M4 document that Angelo worked on specifically was recording info for the test table for the usability test notes and having his actors fill out the questionnaire as well. Also, Angelo also contributed on the QA test plan section by taking one of the browser tests.

Furthermore, Angelo also helped the team decide for point 6) Self-check: Adherence to original Non-Functional Specs, which of our non-functional requirements are **done**, **on track**, or **issue**.

Lakshita:

a) Frontend:

Frontend Features that Lakshita worked on specifically was redesigning our home page from the ground up. Furthermore, Lakshita also made an about page, which showed all of us in it. The below shows the files that she specifically worked on:

- i) home.html**
- ii) about.html**

Benjamin:**a) Backend:**

Backend features that Benjamin worked on was collaborating with Jiaxin on the email verification feature for users registering, and collaborating with Angelo on the notifications feature, specifically on the files and folders:

i) notification.py**ii) authentication folder**

Furthermore, Benjamin completed the verification for Super Users and Admin accounts, which were shown in the **notification.py** file and the **authentication folder**.

b) Frontend:

Frontend features that Benjamin worked on were revamping our **notifications.html** page to actually render notifications from the database, such as if someone requested an item from someone, or requested housing from someone. Furthermore, after Lakshita updated the home page with new fonts and coloring, Benjamin was able to find those variables and place them in our **style.css** file, which allowed an easier way to change the layouts of our website.

c) Documentation:

Things for the M4 document that Benjamin worked on specifically was recording info for the test table for the usability test notes and having his actors fill out the questionnaire as well. Also, Benjamin helped set up the necessary tables and tests that we had in the QA test Plan, while also taking one of the browser tests. Furthermore, Benjamin also helped the team decide for point 6) Self-check: Adherence to original Non-Functional Specs, which of our non-functional requirements are **done, on track, or issue**. Furthermore, Benjamin also helped on the code review section as he sent our back end code to another team and to our front end team to review it, as well as reviewing our front end code that our front end team sent. Benjamin also filled out point 5) Self-check on best practices, as he was most familiar with how we were storing things in our database for added security.

Alec:

a) Frontend:

Frontend features that Alec worked on was providing feedback for the updated homepage design, such as exporting the theme from the **home.html** to all the other pages as well.

b) Documentation:

Things for the M4 document that Alec worked on was setting up the formats for most of the points that we had, which included:

i) Product Summary:

Alec helped set up the page layout for the product summary.

ii) Usability Test Plan:

Alec set up the layout for this section of the milestone, while tasking out others to take the test. After getting the results, Alec organized the section into its specific part.

iii) QA Test Plan:

Alec set up the layout for this section of the milestone, while also taking one of the browser tests. After getting the results, Alec organized the section into its specific part.

iv) Code Review:

Alec tasked out others to do code review, within the team and also collaborating with other teams to review our code. Furthermore, Alec also got the results from the code review and organized it.

v) Self-Check on best practices for Security:

Alec tasked Benjamin to do this part.

vi) Self-Check: Adherence to Original Non-Functional Specs:

Alec led a team meeting with the backend team, and went over the Non-Functional Specs, and applied the necessary points to the specs.

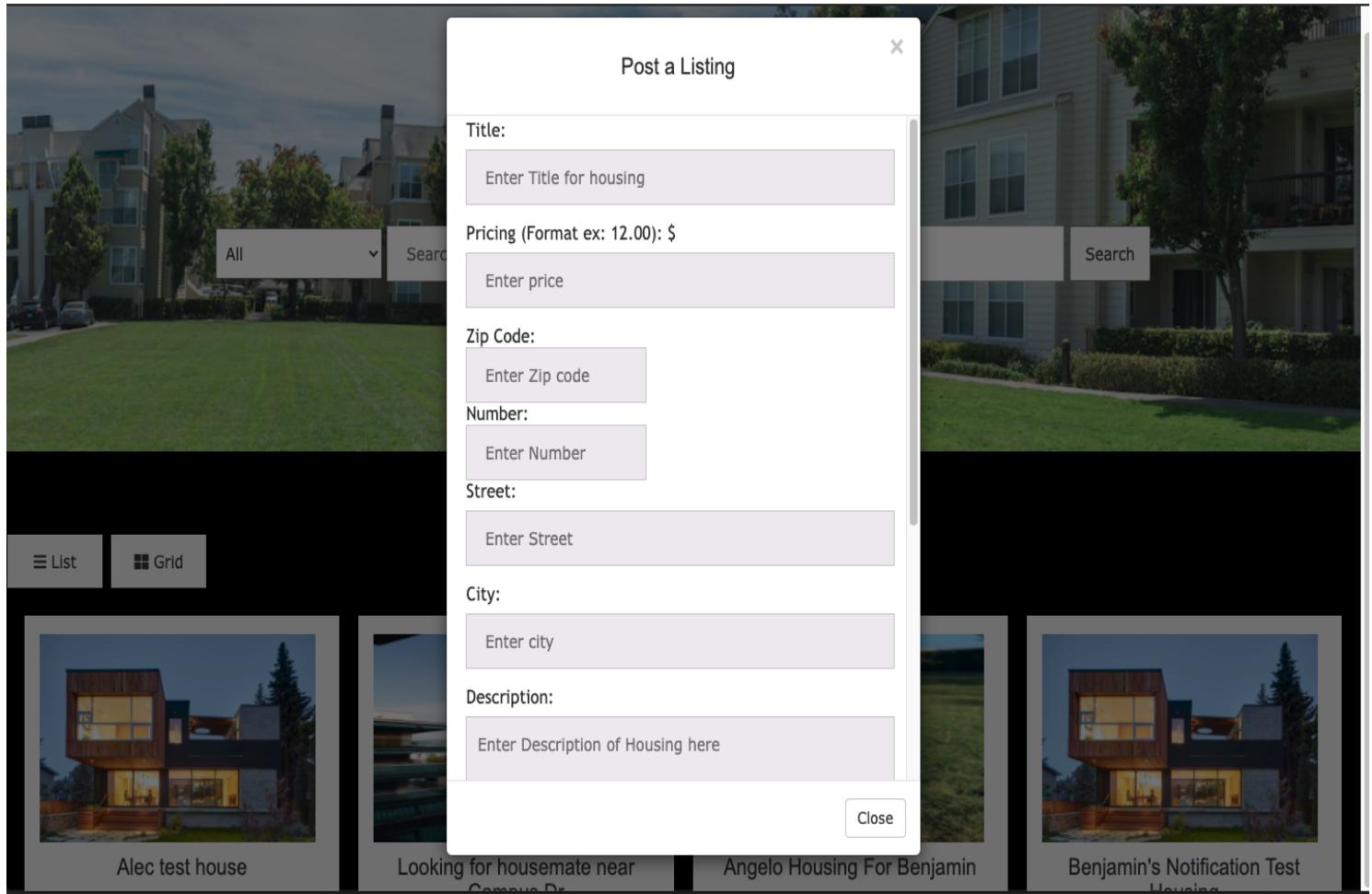
vii) List of Contributions:

Alec filled out this part.

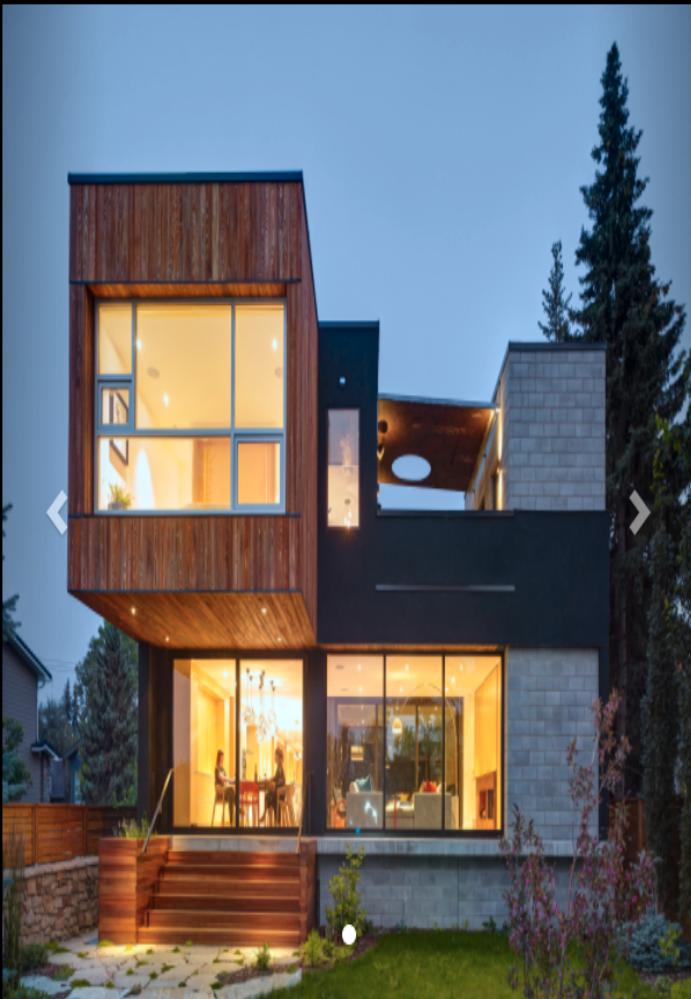
Screenshots of Superior feature

The below shows screenshots of our superior housing feature

Posting a housing listing



Housing Listing Posted



QA Test Plan 1

\$20.00 per month

• **Description:**

This is the first QA Test Plan for Milestone 4. This test plan is for Firefox.

• **Preferred Payment Type:**

Check

• **Pets Allowed:**

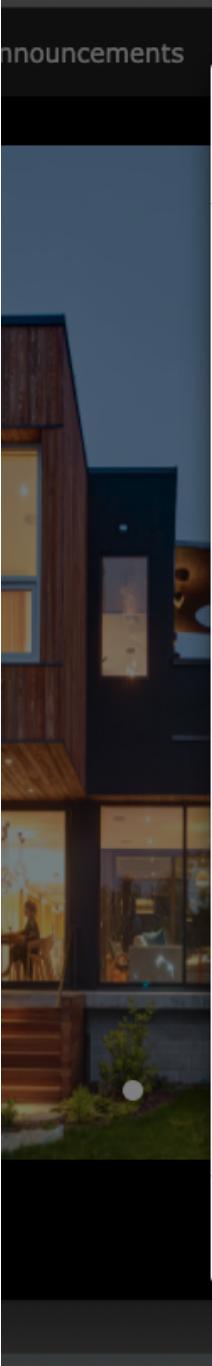
No

• **Address:**

200 Main Street San Francisco CA 94132

+ Interested?

Requesting a Housing Listing



A screenshot of a mobile application interface for requesting a housing listing. At the top, there is a navigation bar with tabs for "Announcements", "Housings", "Items", "Restaurants", and "Map". Below the navigation bar, a modal window is displayed with the title "Tell us about yourself!". Inside the modal, there are four input fields containing the following information:

- Bikram
- Tamang
- btamang@mail.sfsu.edu
- Phone Number: Please Enter in this format: # #-# #-# #-# #-#

Below these fields, there is a "Move in Date:" field with a placeholder "mm/dd/yyyy" and a calendar icon. A large text area labeled "Tell us about yourself!" is present, though it appears empty. At the bottom right of the modal, there is a purple "submit" button and a white "Close" button.

Announcements Housings Items Restaurants Map

Tell us about yourself!

Bikram

Tamang

btamang@mail.sfsu.edu

Phone Number:
Please Enter in this format: # #-# #-# #-# #-# #-#

Move in Date: mm/dd/yyyy

Tell us about yourself!

submit

Close

Getting a housing request in email

Housing Listing Interest!

D Do Not Reply | Gator-connection <csc648team05@gmail.com>
Wed 5/19/2021 9:33 PM
To: Bikram Tamang

Dear Bikram,

Alec Tenefrancia is interested in your housing listing. Here is his/her information:

His/Her Email: alectene@mail.sfsu.edu

His/Her Phone Number: 650-387-9238

Preferred Move In Date: 2021-05-26

About Him/Her:
sasasasasasasasasasasasas

If you feel that he/she is a good match, please get in contact with him/her.

Best,

Gator Connection

[Reply](#) | [Forward](#)

Requesting a housing listing (getting notification on email)

Housing Listing Interest!

D

Do Not Reply | Gator-connection <csc648team05@gmail.com>

Wed 5/19/2021 9:01 PM

To: Bikram Tamang



You have made a request for Christian's housing listing. If the post owner feels that you are a good match, he/she will email you.

Best,

Gator-Connection

[Reply](#)

[Forward](#)

Requesting a housing listing (getting notification on notifications page)

Alec Tenefrancia is interested in your housing listing post.



May 10, 2021, 6:29 a.m.

Gator Connection

Angelo Reyes is interested in your housing listing post.



May 11, 2021, 4:04 p.m.

Gator Connection

Angelo Reyes is interested in your housing listing post.



May 11, 2021, 4:11 p.m.

Gator Connection

Housing Listing Request

May 11, 2021, 4:11 p.m.

sdfad

areyes24@mail.sfsu.edu

Angelo Reyes

Housing Listing Request

May 17, 2021, 4:03 a.m.

afafasdfsdfsfasfasf

areyes24@mail.sfsu.edu

Angelo Reyes

Editing a Housing Listing Post

 Edit Housing Listing Post X

Title:
Looking for roommate near Elliot drive

Pricing: \$
1200.00

Zip Code:
94534

Number:
224

Street:
Elliot drive

City:
san francisco

Description:
roommate (male) needed in a 2 bedroom apartment
near Elliot drive. no smoking. Shared bathroom and

FAQ Close

Screenshots of Key DB Tables

Registered User - Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
reg_user_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
user	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
first_name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
last_name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
full_name	VARCHAR(90)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	CONCAT(first_name, ' ', las...

Account - Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
account_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
email	VARCHAR(45)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
password	BINARY(60)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
createdAt	TIMESTAMP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
email_verified	TINYINT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	0
user	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Student Account - Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
student_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
graduation_month	TIMESTAMP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
account	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Admin Account - Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
admin_account_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
account	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
position	VARCHAR(45)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
is_approved	TINYINT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	0

Super User Account - Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
super_user_id	SMALLINT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
account	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
is_approved	TINYINT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	0

Housing Listing Post - Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
housing_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
price	DECIMAL(10,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
post	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
are_pets_allowed	VARCHAR(5)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
preferred_pay_type	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
address	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Housing Listing Request - Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
housing_request_id	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
createdAt	TIMESTAMP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
registered_user	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
housing_listing	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Housing Listing Form - Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
form_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
about_me	TEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
move_in	TIMESTAMP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
phone_number	VARCHAR(20)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
housing_listing_request	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
registered_user	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

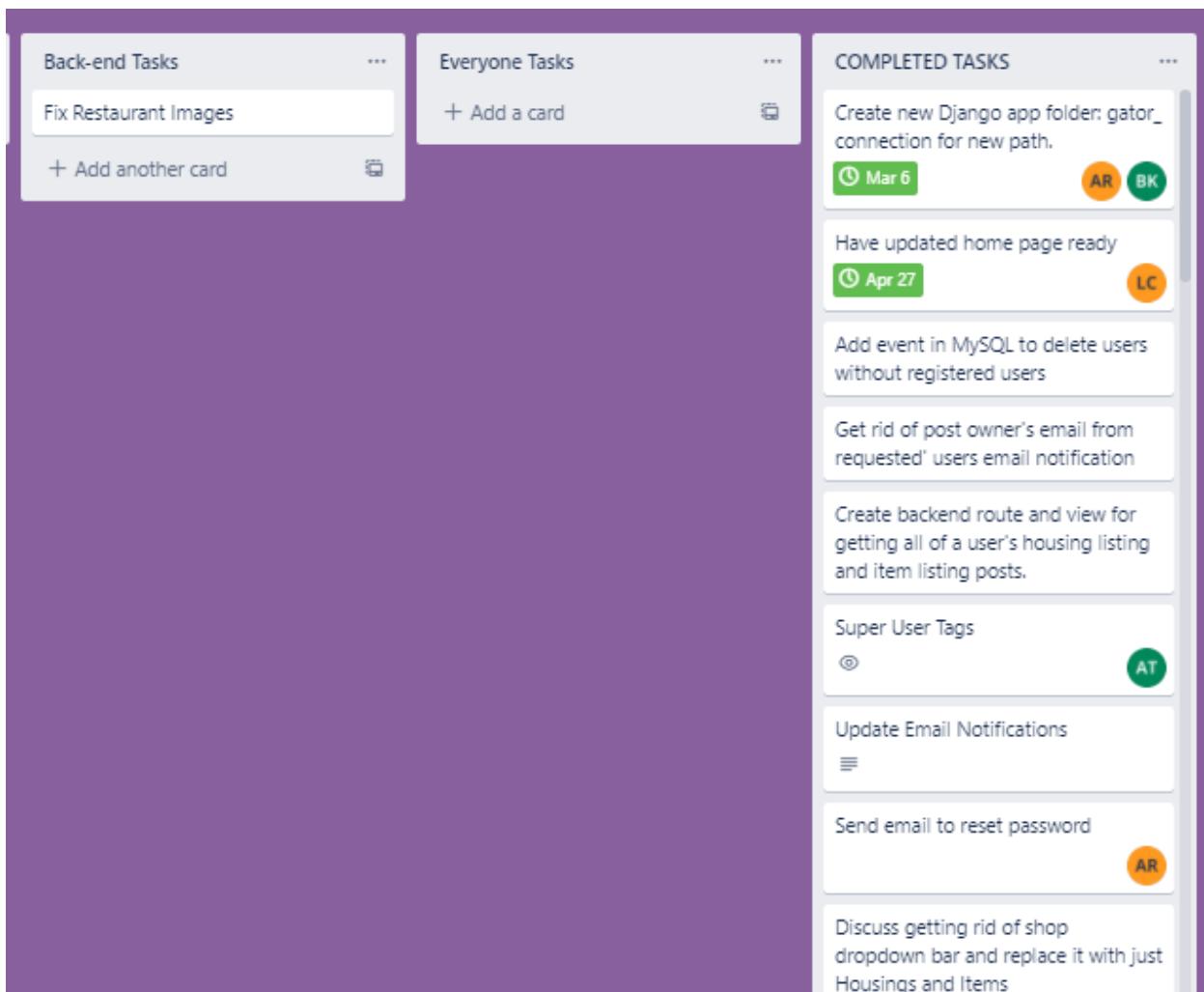
Notification - Table X

Table Name:		Notification	Schema: gator_connection									
Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression		
notification_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>			
createdAt	TIMESTAMP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>			
registered_user	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>			
message	TEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL		
read	TINYINT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'		

Screenshots of Trello

The below shows our tasks for the project. It shows how we kept track of the project, and the tasks we needed to complete. Trello allowed us to split tasks separately to the backend and frontend team. By doing this, we were able to keep track of the tasks we were assigned to. How the dashboard was organized is that it is split into 6 categories, Milestone 5 tasks, Team Lead tasks, Front-End tasks, Back-end Tasks, Everyone Tasks, and COMPLETED TASKS.

Furthermore, we added more tables to accommodate for Milestone 5, such as a table called Milestone 5 tasks and Milestone Revisions. By doing this, we were able to keep track which task was assigned to which team, which tasks we had completed, and which tasks were assigned to who.



COMPLETED TASKS ...

2/2

M1V2

Update dummy announcement pages we wish to keep
② AT

Housing Request Modal
② AT

Change cards for Announcements, Items, housing, restaurants on home page
⌚ May 9 LC

Code Review
⌚ May 9

reset password - backend
⌚ May 15

Usability test plan
⌚ May 9

Home Page
⌚ May 9 ≡ LC

Add form validation for all modals - frontend
⌚ May 7 ≡

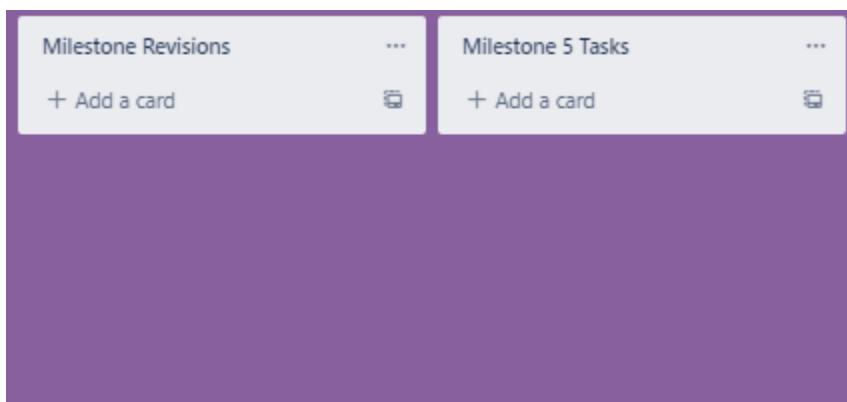
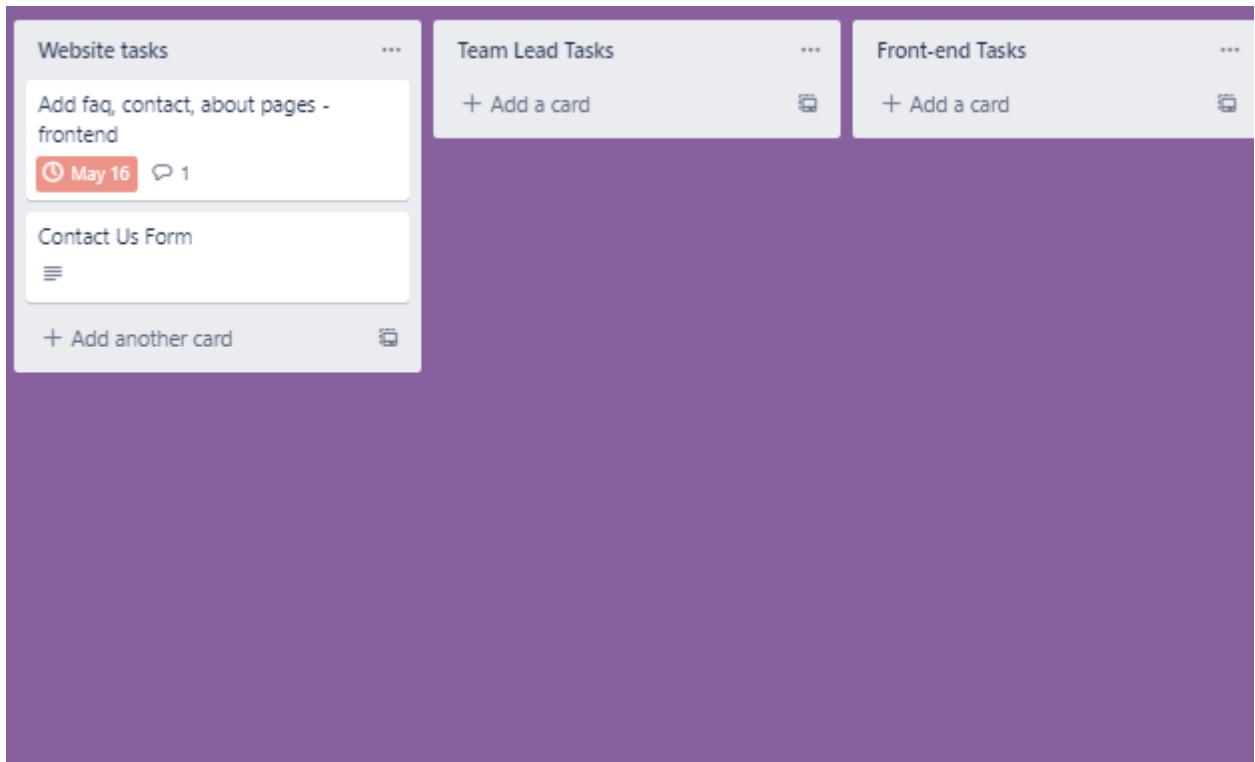
Create ERD by Monday
⌚ Mar 8 AR BK

Restaurant carousels thumbnails and fix
BT

Add address to housing Listing cards on the housing main page

is verified for housing checks
② ⌚ May 7 AT

redesign other pages to match theme of home page-frontend
≡
+ Add another card



Team Member Contributions

Alec

AT

Alec Stephen Tenefrancia

Sat 5/15/2021 11:10 AM



To: Angelo Gloria Reyes; Lakshita Chugh; Benjamin Patrick Kao; Bikram Tamang; Carmen Denisse Paisano; Jiaxin Yu

Hello Team05,

This is the start of the collection of emails for point **7) Team Member Contributions** in M5. When you are ready, please **reply to this email and send it out to everyone**, with the following information:

- a) His/her contributions to team project and teamwork (technical and any other) in no more than half a page – point format is OK.
- b) Number of submissions he/she made to Github team dev. branch

Thank you again,

Alec Tenefrancia

AT

Alec Stephen Tenefrancia

Thu 5/20/2021 9:50 AM



To: Lakshita Chugh; Jiaxin Yu; Benjamin Patrick Kao
Cc: Angelo Gloria Reyes; Bikram Tamang; Carmen Denisse Paisano

Hello Team,

It was great working with everyone here! Here are my contributions.

Alec

- Scheduled all the meetings for both backend and frontend teams.
- Provided meeting recaps at the end for each meeting to catch up others who could not make it.
- Set up the google drive for our team for all the necessary information we would need for the milestones and other important info.
- Designed the final drafts for the wireframes from other team members after getting their rough drafts.
- Managed Trello with all the tasks we would need for the Project.
- Turned in all milestone related information to the Professor.
- Finalized all Milestone documents, while checking over everyone's work in the respective milestones.
- Set up the EC2 instance for the website.
- Set up the MySQL database in the cloud.
- Set up the Navbar for the website.
- Designed the login and register modals.
- Implemented JavaScript Validation for the login and register modals.
- After getting the logic for the JavaScript Validation, tasked out other front-end members to do it for the other modals.

As team lead, I was responsible for scheduling all the meetings, providing recaps for everyone, setting up Milestone drafts, assigning others to complete tasks through Trello, and finalizing all the Milestone Drafts. Furthermore, I was also responsible for checking all features when they were pushed to GitHub. After I found that everything was good, I would push it then to the EC2 instance in the cloud, which I also maintained and made sure was working.

I pushed 199 commits to the GitHub team development.

...

[Reply](#) | [Reply all](#) | [Forward](#)

Angelo

AR

Angelo Gloria Reyes

Wed 5/19/2021 8:22 PM



To: Alec Stephen Tenefrancia; Lakshita Chugh; Benjamin Patrick Kao; Bikram Tamang +2 others

Angelo Reyes

- Attended every backend and whole team meetings
- Assigned and organized tasks on Trello specific to backend team so that work was distributed between the team
- Implemented backend logic for registration and login. The code was later revised with help of Benjamin, Jiaxin, and Alec
- Implemented email notification for Housing Listing Request, Post and Delete.
- Implemented frontend and backend logic for Housing Listing Post/Item when a user posted a housing listing or item for sale. Code was later revised by Benjamin
- Implemented backend logic to save notifications on Notifications table
- Edited and gave feedback on database schema and models on models.py
- Edited M4 document, edited and worked on QA test plan as well as other sections in other milestones.

I pushed 54 commits to the team Github development.

...

Alec replying to Angelo

AT

Alec Stephen Tenefrancia

Thu 5/20/2021 10:27 AM

To: Angelo Gloria Reyes



Thank you Angelo! I agree with this.

Regards,

Alec Tenefrancia

...

[Reply](#) | [Forward](#)

Lakshita



Lakshita Chugh

Thu 5/20/2021 8:50 AM



To: Jiaxin Yu; Benjamin Patrick Kao

Cc: Alec Stephen Tenefrancia; Angelo Gloria Reyes; Bikram Tamang; Carmen Denisse Paisano

Hi Team,

It was amazing working with everyone. Here are my contributions:

- Attended every front-end meeting and the whole team meetings as well.
- After Milestone 2 suggestions, completely transformed our homepage to look it more professional
 - Changed the theme to dark mode for all the web pages.
 - Added elements including animated sequential cards, column containers below cards with text, and animation to the subtitle, search bar, and logo.
- Selected all the images for our homepage cards, Announcements background, Items background, and Restaurants background. Designed the background section for all.
- Created animated buttons for all the web pages.
- Added new TTF fonts to the headings, other than the regular ones available.
- Designed the square cards for each of our housing listings on the housing webpage, and listing of items on our items webpage with the list and grid viewability.
- Created the Meet our team about page, and designed the columns for each team member by adding their images along with a transparent logo at the top.
- Created and formatted the faq webpage with assistance from Carmen.
- Organized and edited the Milestone 2 and Milestone 3 documentation for the final turn-in.

I pushed 51 commits to the Github team development.

Our front-end meetings with the entire team used to happen every Tuesday at 5:30 pm. As a lead, I coordinated with the front-end members, Bikram and Carmen, outside hours to assist each other with almost everything ranging from understanding the git better, connecting MySQL, AWS server to finishing up all the front-end tasks and polishing it at every stage. I worked with them on Wednesdays at 8 pm after our classes got over to brainstorm new potential features, bug detection, its fixing, and then worked on our decided font-end tasks later at our own paces and as per scheduled deadlines. We took each other's suggestions during this session to make sure the aesthetics looked good and did a trial of some new UI features together.

Best regards,
Lakshita Chugh

...

Alec replying to Lakshita

AT

Alec Stephen Tenefrancia

Thu 5/20/2021 10:26 AM

To: Lakshita Chugh



Thank you Lakshita! I agree with this.

Regards,

Alec Tenefrancia

...

Benjamin

BK

Benjamin Patrick Kao

Wed 5/19/2021 10:57 PM



To: Alec Stephen Tenefrancia; Angelo Gloria Reyes; Lakshita Chugh; Bikram Tamang +2 others

Hello Everyone,

It was great working with all of you. Here are my contributions to this project:

- Attended every backend and whole team meeting, as well as, most of the frontend meetings.
- Created the Authentication API
- Implemented all of the search algorithms
- Implemented the frontend logic for the Restaurants page and Restaurant Detail page, and the backend logic for it.
- Implemented the frontend and backend logic for the Notifications page, including the Super User Tools
- Implemented the frontend and backend logic for the User's profile page.
- Implemented the Announcements feature, Post/Edit/Delete Housing and Items feature, notification feature, and a lot of the backend routing for the views.
- Created the database EER and schema and created the models for the entire database in models.py
- Added all of the backend form validation logic and created some custom ORM objects so that we could use Full Text Search for some of the search features.
- Created the custom Gator Connection exceptions
- Edited M4 document, completed the security sections of the milestone documents and any other database sections in the milestones.

I pushed 122 commits to the GitHub team development.

Thanks,

Benjamin Kao

...

Alec replying to Benjamin



Alec Stephen Tenefrancia

Thu 5/20/2021 10:27 AM

To: Benjamin Patrick Kao



Thank you Benjamin! I agree with this.

Regards,

Alec Tenefrancia

...

Jiaxin

JY

Jiaxin Yu
Wed 5/19/2021 11:46 PM
To: Benjamin Patrick Kao
Cc: Alec Stephen Tenefrancia; Angelo Gloria Reyes; Lakshita Chugh; Bikram Tamang; Carmen Denisse Paisano



Jiaxin Yu

- Attended every backend and whole team meeting as well.
- Created the Email and Search API
- Created and edited the database table based on the development of API.
- Implemented backend logic for notification check and set.
- Implemented email notification for Item Listing Request, Post and Delete.
- Implemented verification logic and feature.
- Implemented frontend and backend logic for Item Listing Post by referencing the code that Benjamin and Angelo code.
- Wrote the documentation for files of notification, restaurants, super user, views and the database subdirectories.
- Edited M4 document, edited and self-check document.

I pushed 23 commits to the Github team development.

Alec replying to Jixin



Alec Stephen Tenefrancia

Thu 5/20/2021 10:27 AM

To: Jixin Yu



Thank you Jixin! I agree with this.

Regards,

Alec Tenefrancia

...

Bikram

BT

Bikram Tamang

Wed 5/19/2021 11:00 PM



To: Angelo Gloria Reyes; Alec Stephen Tenefrancia; Benjamin Patrick Kao; Jiaxin Yu +2 others

Hello, Team my contributions are as follows:

Bikram Tamang

- Attended all front end meetings and all whole team meetings
- Edited milestone 1 version 1
- Collaborated with Benjamin to establish the foundation for competitive analysis.
- Collaborated with Carmen to make the UI mockups consistent with all team members.
- Created all the base HTML pages on the front-end which help act as a bridge for the back-end to test and deploy data.
- Collaborated with Benjamin, Lakshita, and Carmen for major CSS and layout changes for the front-end.
- With major help from Benjamin, remodeled the restaurant page.
- Created a 5-star rating system on the front-end which later was fully implemented by the back-end team.
- Implemented a Google Maps API on the front-end with searchable functionality.
- Worked with all the team members to create a consistent wireframe for the horizontal prototype.
- Collaborated with Carmen to implement Javascript validation on all the modals in the front-end.
- Specifically implemented input validation of housing and items pages to improve UX
- Collaborated with other teams and Benjamin for code review.
- Collaborated with front end and team lead for product presentation.
- GitHub commits: 61 commits

...

Alec replying to Bikram



Alec Stephen Tenefrancia

Thu 5/20/2021 10:27 AM

To: Bikram Tamang



Thank you Bikram! I agree with this.

Regards,

Alec Tenefrancia

...

Carmen

CP

Carmen Denisse Paisano

Wed 5/19/2021 11:04 PM



To: Bikram Tamang; Angelo Gloria Reyes; Alec Stephen Tenefrancia; Benjamin Patrick Kao; Jixin Yu +1 other

Hi, team here are my contributions:

Carmen Paisano

- Attended front end meetings and whole team meetings.
- Collaborate with Bikram to make the UI mockups consistent with all team members
- Created Logo for our website and added to our Navbar.
 - Logo had to be updated and design was changed to match the aesthetic of our site.
- Collaborated with Bikram and Lakshita for layout changes for front-end
 - This included brainstorming, bug detecting, and editing with Lakshita and Bikram.
- Implemented the base grid system for Housing and Items page as well as adding button options for list or grid view.
- Revised the Map feature by making a placeholder that would directly start at SFSU.
- For Milestone 3 worked on the rough drafts for wireframes 5 and 6.
- Collaborated with Bikram to implement JavaScript validation on all the modals in the front-end.
 - Specifically implemented input validation of Restaurants and Announcements pages
- Implemented JavaScript validation for Contact page along
 - Specifically implemented input validation of emails and description
- Collaborated with Lakshita to make FAQ questions

I pushed 22 commits to GitHub team development

...

Alec replying to Carmen

AT

Alec Stephen Tenefrancia

Thu 5/20/2021 10:27 AM

To: Carmen Denisse Paisano



Thank you Carmen! I agree with this.

Regards,

Alec Tenefrancia

...

Post Analysis

a) Main Challenges:

The challenges that our team came across implementing our project was:

i) Scheduling Challenges:

None of us had consistent schedules for meetings, so it was hard to schedule meetings, until we found out it would be easier to split meetings between front and back end meetings. Although we had meeting recaps, it was still hard to communicate until we met during the main meetings on Thursdays.

ii) Communication Challenges:

While we had split meetings between both teams, and meeting recaps for those meetings as well, it was still hard to communicate between our team. As such, there were times when information was lost between us communication, further increasing setbacks.

iii) Technical Challenges:

While finalizing our Tech Stack, we found out that most of the team had no experience with Django or Python. Furthermore, for the members who had experience with Python, Django was completely new territory, as it dealt with both front end and back end technologies, which most of us were unfamiliar with.

b) What to do better next time to address those challenges:

What we would do better next time for implementing the project and to address these challenges are:

i) Scheduling Improvement:

What we would do to improve on our scheduling is to make sure that we found two days where everyone could meet, making our meeting count up to four. For this, we only had one meeting where everyone would meet, and two separate meetings

ii) Communication Improvement:

What we would do to improve our communication, is that after each meeting to provide a more detailed meeting recap. While initially we did do meeting recaps, they were not as detailed as we hoped they would be. Furthermore, we would now make sure that everyone read it by asking them to acknowledge it by liking the post, or replying back to the email saying that they read it.

iii) Technical Improvement:

What we would do to improve our technology stack is to make sure that everyone in the team is kept up with their technologies. For example, the tech stack we chose was Django, which is primarily python based. What we would do to make sure is that everyone would be able to understand python, so they could contribute to the project. Furthermore, also along the lines of Communication Improvements as well, but we would make sure to write even more detailed documentation, so that everyone would know what function, feature, etc. does, so to make sure that everyone is kept up to speed.