

# 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 4

Milestone/Version	Date
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

---

<b>Product Summary</b>	3
<b>Usability Test Plan</b>	8
Test Objectives:	8
Test Description:	9
Usability Task Description:	10
Questionnaire:	18
<b>QA Test Plan</b>	24
Test Objectives:	24
QA Test Plan:	25
QA Test Plan #1:	26
QA Test Plan #2:	33
QA Test Plan #3:	41
<b>Coding Style</b>	47
Coding Style	47
Code Review	48
<b>Self-Check on Best Practices for Security</b>	60
<b>Self-Check: Adherence to Original Non Functional Specs</b>	73
<b>List of Contributions</b>	78

# 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:

- 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.
- 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.
- Student Account
  - A Student account shall have a unique student id.
  - A Student account shall have a unique SFSU email.
  - A Student account shall have the graduation date of the registered user.
- Admin Account
  - An admin account shall have a unique admin id.
- 
- Super User Account
  - A super user account shall have a unique super user id.
- Sale Item

- A sale item shall be posted by a registered user.
- A sale item shall have a unique item id.
- A sale item shall have a title for the item.
- A sale item shall have a message describing the item.
- A sale item shall have one picture attached to it.
- A sale item shall be purchased by a registered user.
- A sale item shall be taken down by the registered user that posted it for sale.
- A sale item shall have a price.
- 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 have a unique housing id.
  - A housing listing shall have a title.
  - A housing listing shall have a message describing the item.
  - A housing listing shall have a price.
  - A housing listing shall have at least 1 picture attached to it.
  - 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 year.
  - 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 a name.
  - A restaurant shall have a location.
  - A restaurant shall have a unique restaurant id.
  - A restaurant shall have a rating.
  - 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

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

<b>Task</b>	<b>Description</b>
<b>Task</b>	Search for a housing listing
<b>Machine state</b>	Housing page(already logged in/registered/verified)
<b>Successful completion task</b>	Housing posts searched by search criteria
<b>Benchmark</b>	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
<b>Post a Housing Listing</b>	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"><li>- Form validation should have stopped user before submitting, indicating that price was too much.</li><li>- 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.</li><li>- User suggested instead of having a set price, have a price range that a user is willing to sell for.</li><li>- User also believed that form fields are not good when posting an apartment.</li></ul>	2 minutes and 30 seconds
<b>Delete a Housing Listing</b>	100%	None	<p>Very easy.</p> <p>NOTE: Did not test delete housing notification for interested users.</p>	5 seconds

<b>Request a Housing Listing</b>	50%	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.	Is it necessary to have graduation date? What if someone was just looking for a house?  NOTE: Did not test request housing notification for user that posted.	
<b>Edit a Housing Listing</b>	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
<b>Search for a Housing Listing</b>	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

			<p>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.</p> <p>User also suggested adding a number of bathrooms/bedrooms indicator.</p>	
--	--	--	--	--

**ii) Participant 2:**

<b>Test/Use Case</b>	<b>%Completed</b>	<b>errors</b>	<b>comments</b>	<b>Average time</b>
<b>Post a Housing Listing</b>	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
<b>Delete a Housing Listing</b>	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

<p><b>Request a Housing Listing</b></p>	<p>75%</p>	<p>System crashed. <i>IntegrityError at /housing_request/21</i>  <i>(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>
---	------------	--	---	---



<b>Edit a Housing Listing</b>	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
<b>Search for a Housing Listing</b>	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:

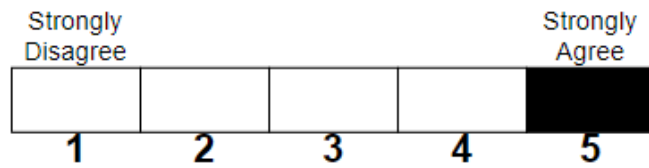
The below Test Tables were tested with 2 participants, outlined as **Participant 1** and **Participant 2**. (Participants are the same from the Test Tables)

### i) Participant 1:

<b>Gender: Female</b>	<b>Age Range: 18-24yrs old</b>												
<b>1) Post Housing:</b>													
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>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>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>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									
<b>2) Delete Housing:</b>													
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>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>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>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:

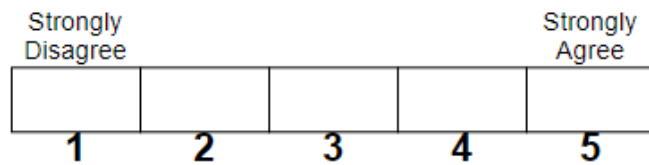
i) The process to send a housing request was simple and quick:



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

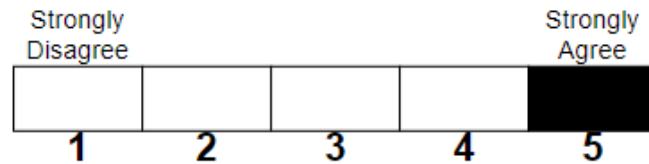


iii) The housing request email was easy to understand for the receiver:

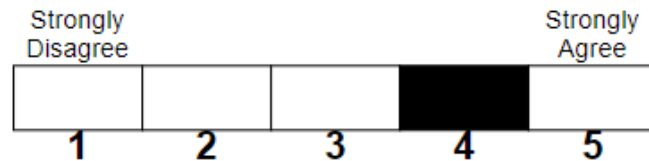


### 4) Edit Housing:

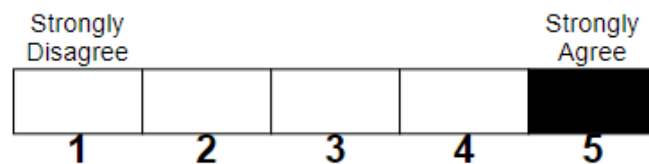
i) It was easy to edit my housing listing post:



ii) Adding more images to the listing was helpful:



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



## 5) Search Housing:

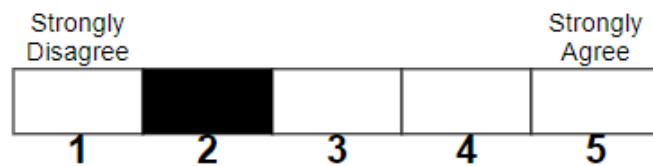
i) The search bar was easy to use:



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



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



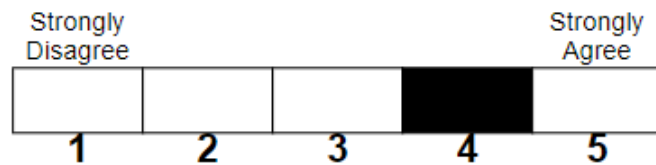
ii) Participant 2:

**Gender:** Female

**Age Range:** 45-54yrs old

**1) Post Housing:**

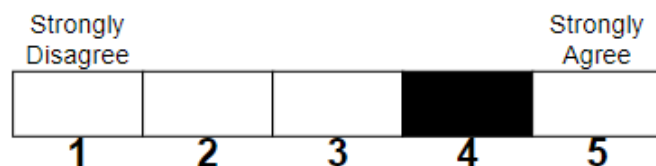
i) It was easy to input all of the necessary information to post a housing listing:



ii) It was easy to post a housing listing:



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

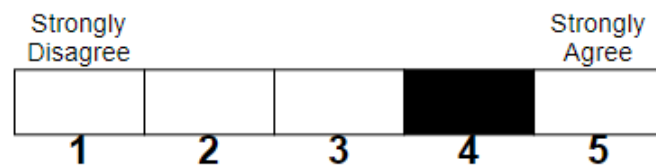


**2) Delete Housing:**

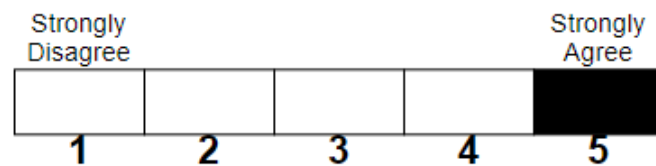
i) It was easy to delete my housing listing:



ii) The delete housing notification was a useful confirmation:

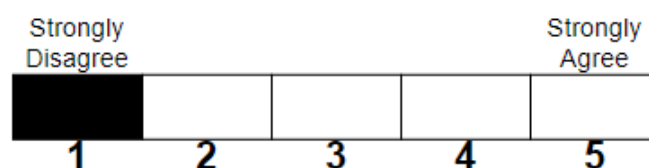


iii) The delete housing notification/email was necessary:

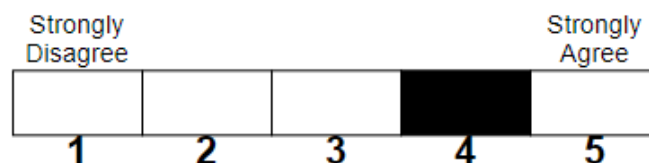


### 3) Request Housing:

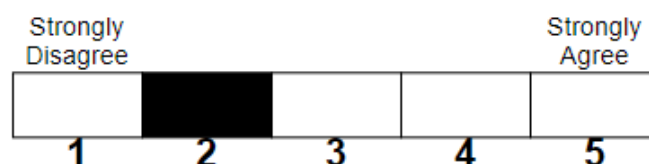
i) The process to send a housing request was simple and quick:



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

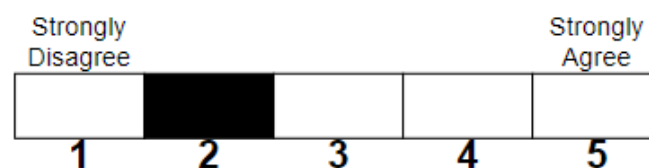


iii) The housing request email was easy to understand for the receiver:

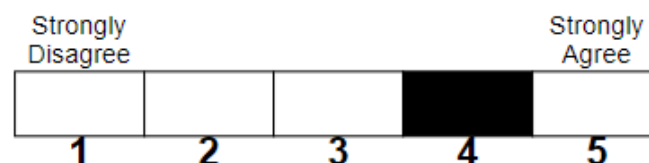


### 4) Edit Housing:

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



ii) Adding more images to the listing was helpful:

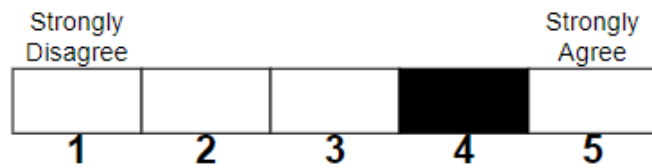


iii) It was helpful having my previous information filled out:

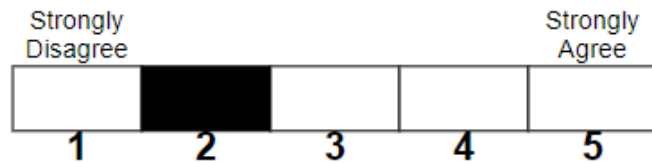


## 5) Search Housing:

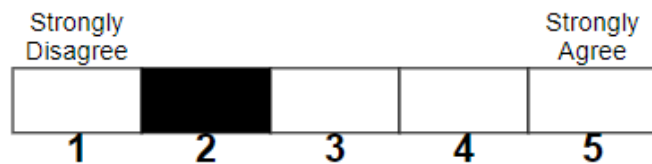
i) The search bar was easy to use:



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.

		purchase request or in this case, a housing listing interested request.	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.	
--	--	---	--	--

QA Test Case #2:

<b>Tas k</b>	<b>Description</b>	<b>Test Input</b>	<b>Expected Output</b>	<b>PASS/FAIL</b>
<b>1</b>	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
<b>2</b>	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.
<b>3</b>	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.	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>  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. 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.

		Open: 11:00 AM Close: 09:00 PM Takeout Available: Yes Description: aaaaaa Upload Image: 9931.jpg <u>Super User Input:</u> Click "Reject"		
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 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: Check Pets Allowed: No Upload Images: 99310.jpg	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	.
<b>5</b>	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	User tests registering for a student account. <u>Input:</u> First Name: Alec Last Name: Tenefrancia SFSU Email: alectene@mail.sfsu.edu 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.	Pass
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 "Accept"	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>  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>  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	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>	Student that is logged in can see the button request for the housing or item.	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: alectenefranca2@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	<p>User tests to try posting a listing on housing page</p> <p><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</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>  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	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	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	inputted information the exact same.	
4	Test that only registered users shall be allowed to post restaurant reviews.	A registered user with a Student account goes to any restaurant detail page and clicks on the "Write a Review" button.  <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.	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.	<b>Pass</b>
5	Test that only registered users shall be allowed to make a purchase request	Logged Out user navigates to the Housing section, finds any Housing Listing Post, and tries to make a request. Logged Out user navigates to the Item section, finds any Item Listing Post, and tries to make a request.  <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	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:

<b>Tas k</b>	<b>Description</b>	<b>Test Input</b>	<b>Expected Output</b>	<b>PASS/FAIL</b>
<b>1</b>	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
<b>2</b>	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.
<b>3</b>	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</u> <u>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>  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. 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 Tenebrancia

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

---





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





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](#)



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
heo 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 tha 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 commnet 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))
```

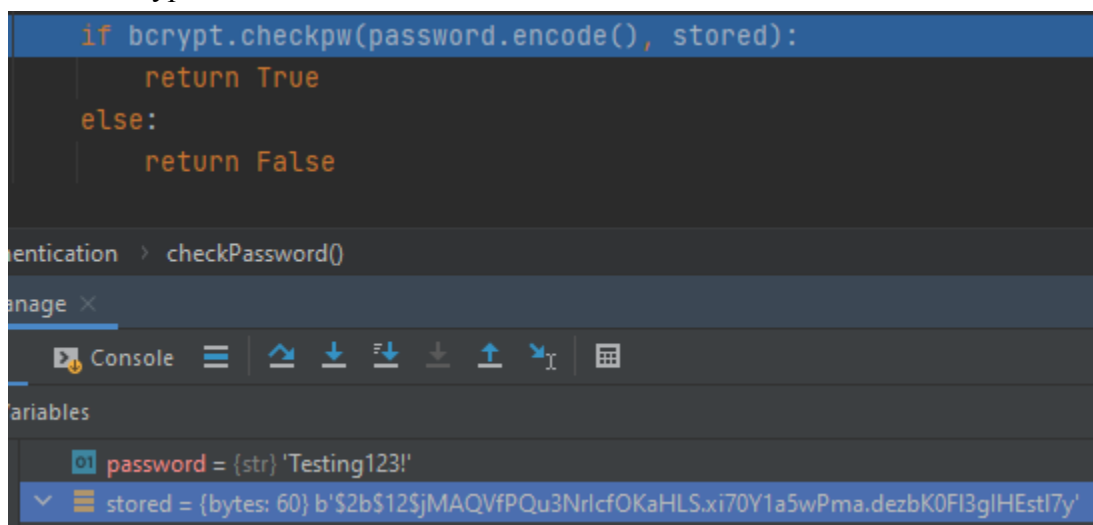
### MySQL Workbench Account Table Data:

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	alectenefranicaa2@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

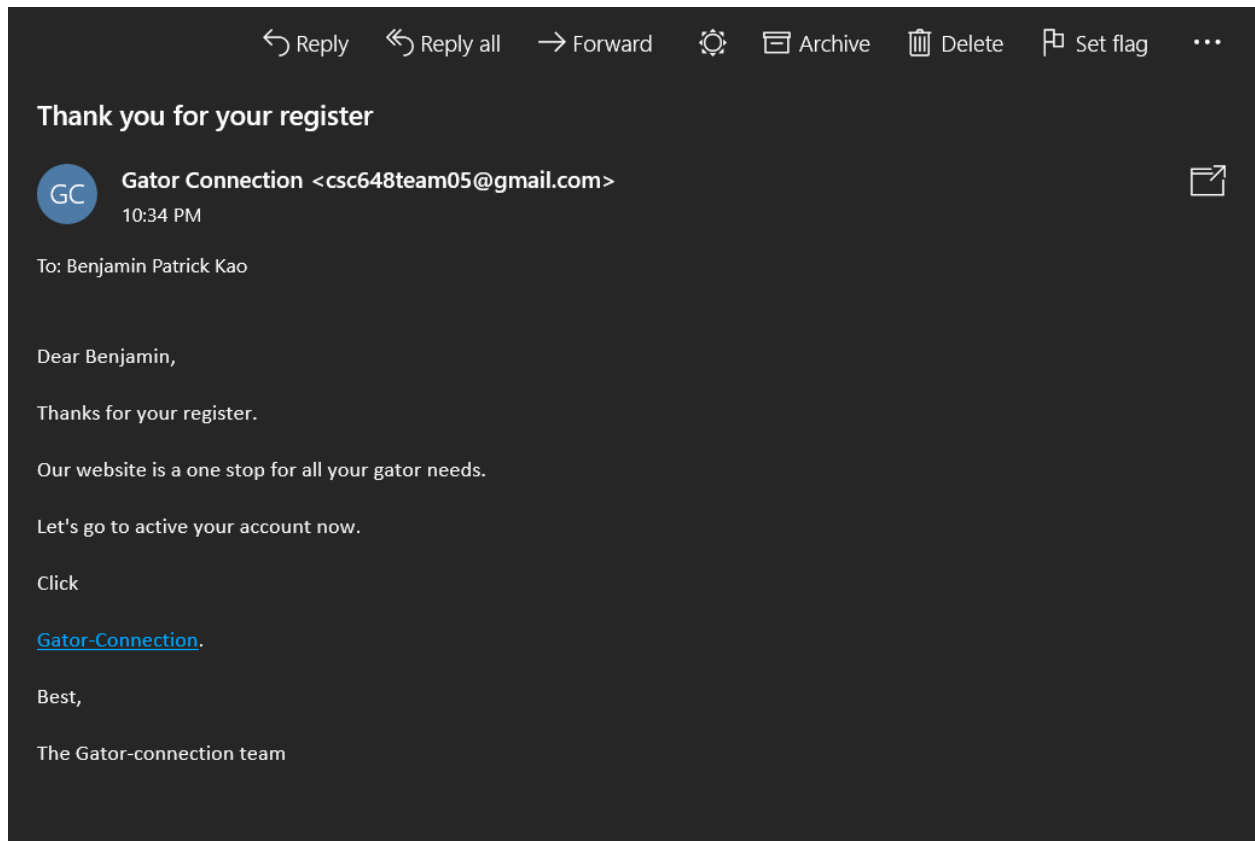
### Debugger Password:

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

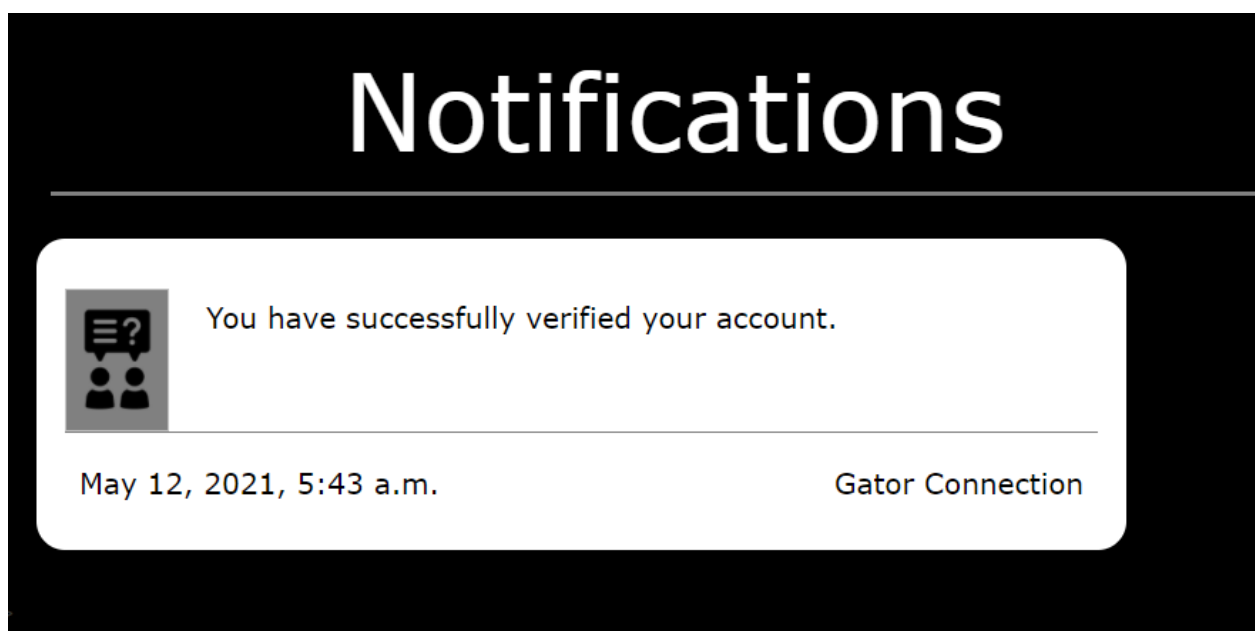
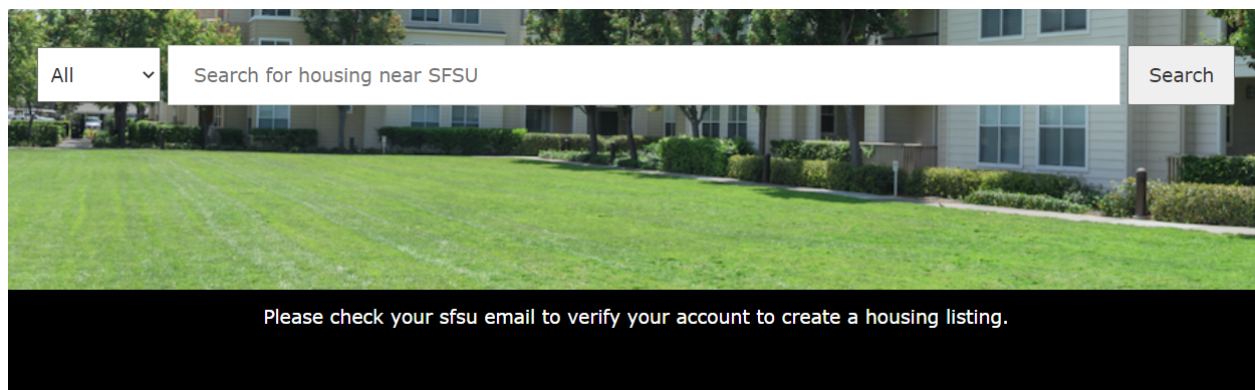


### 3. Email Verification

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

Login

Student Admin Super User

alectene@mail.sfsu.edu

This is a valid SFSU email!

.....

This is a valid password length!

Login

Don't have an account? Click here to create one.

Close

## 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

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: 05/01/2021

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

Choose File

IMG\_3398.jpg

submit

## 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

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.