

**Software Engineering 648/848 Spring 2022**  
**GatorMart**

Team 2

Shane Waxler - Team Lead - [swaxler@mail.sfsu.edu](mailto:swaxler@mail.sfsu.edu)

Chuting Yan - Front-End Lead

Robert Garcia - Back-End Lead

Minggu Ma - GitHub Master

Melissa Ho - Front-End

Xiaoqing Yao - Front-End

Joe Guan - Back-End

**Milestone 4**

**5/14/2022**

## I. Product Summary

GatorMart is an online marketplace for the members of the San Francisco State students, professors, and faculty. It allows users with a validated SFSU email to buy and sell items and services. Once registered, validated, and logged in, a user may view product details, make offers on listings, as well as post their own. Interest shown in a user's listings will come in the form of messages, which is viewed on the user's profile page.. Users of the website can be assured that they are interacting with other people from the SFSU community due to the site administration validation.

GatorMart allows users to

- View the current items in the marketplace
- Search for items
- Filter Listings
- Register for an account with SFSU email

Registered users can

- Login to their account
- Upload data to post their own listings
- Message other sellers to request purchase
- Accept or decline offers on their current listings

Administrators can

- Remove listings
- Approve or deny listings
- Approve or deny new user registration, or delete user accounts

GatorMart's unique feature is to allow posters to choose between several on campus options to meet up. These locations are main hubs in busy areas of the campus.

<http://ec2-54-237-46-1.compute-1.amazonaws.com/>

## II. Usability Test Plan (max 2 pages)

### Test Objectives - Uploading and Posting Data

The objective of our usability test is to test user satisfaction on the ease of posting data to GatorMart. GatorMart is an online marketplace, where registered users are expected to post multiple items or services to the database to be displayed to other members. The test will help us determine future areas of improvement to optimize the user experience.

### Test Background and Setup

#### System setup

A browser on a PC, laptop, or other mobile device with internet access.

#### Starting point

A user that is registered with GatorMart but not currently logged in.

#### Intended Users

San Francisco State University students, faculty, and staff. Those with basic knowledge of how to navigate a browser on the PC or a mobile device.

URL <http://ec2-54-237-46-1.compute-1.amazonaws.com/item/post>

#### User Satisfaction

For our user satisfaction survey, we would present the focus group with this survey:

Task	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
The process of locating the link to the post/uploading data page was simple	1				
Process of navigating to the "POST" page was easy	1				
The form to post and upload data was intuitive and aesthetically pleasing	1				
The mandatory sections were clearly marked	1				
The clickable sections were not too small	1				
I was able to complete the task in a reasonable amount of time	1				
The process of clearing all the data was simple and intuitive	1				
The buttons and fields were clearly marked so that there were no ambiguities on what to fill in	1				

## Usability Task Description

Please log in to your GatorMart account. Then, proceed to the post page and fill out the required fields on the form. You may also add a description for further details of your post. Afterwards, please click the submit button to submit your listing.

## Evaluation of Effectiveness

To evaluate the effectiveness of posting a listing, we would measure the percentage of users that completed these following tasks successfully, as well as note the errors that users could encounter in the process. We would also give users an option to leave a comment on how effectively they were able to complete the task in a reasonable amount of time.

To evaluate effectiveness of this function, we would adhere to the following example chart:

Test/Use case	%completed	errors	comments
Login	100		
Navigation to /item/post page	100		
Filling out the forms	100		
Uploading a picture	100		
Clearing the form	100		
Submitting the form	100		

## Evaluation of Efficiency

To measure efficiency of the feature, we would check the following tasks: Login, navigation to page, filling out the form, upload of pictures, clearing the form, and submitting the form. We would test for how long a user takes to complete each task, the number of clicks, and the number of errors a user encounters.

To evaluate the efficiency of our function, we would adhere to the following example chart:

Test/Use case	Time of completion	Number of clicks	Number of errors
Login			
Navigation to /item/post page			
Filling out the forms			
Uploading a picture			
Clearing the form			
Submitting the form			

### III. QA Test Plan

#### Test Objectives

To test the reliability of the posting function on GatorMart when a registered and logged in user attempts to create a posting. We are testing the success rates of when a user attempts to post and upload new data to GatorMart's marketplace.

#### Hardware and Software Setup (include URL)

Hardware setup - A PC, laptop, or mobile device with internet connection

Software setup - Chrome or Firefox browser

<http://ec2-54-237-46-1.compute-1.amazonaws.com/>

#### Feature to be tested

Priority 1 function "Post/Sell". We are testing to see if filling out the form, filling it out incompletely or incorrectly, and clearing the form are successful.

#### QA test plan example chart:

Test	Title	Description	Test Input	Expected Correct Output	Results (P/F) Firefox	Results (P/F) Chrome
1	Full inputs on the form	When a user completes all the fields on the form, they click submit to upload the data to the marketplace	1. Completely and correctly fill all the fields on the entire post form. 2. Press Submit	Redirected to user's profile page, where the item shows up on the user's profile under "current listings" section. When searching for the item, the item appears in the marketplace search function.		
2	Only one field	When a user only completes one field on the form and attempts to click the submit button.	1. Fill out the name of the listing only 2. Press Submit	A message appears reminding the user that all fields marked with a red asterisk are mandatory. No redirect. Submission is unsuccessful.		
3	Too many words in the description box	The user writes too many words in the description box, which states that the maximum is 250 characters	1. Write 251 characters in the description box 2. Press Submit	A message appears reminding that the user shall not write more than 250 letters for the description of the item. There is no redirect, submission is unsuccessful.		
4	Clearing the fields	The user fills out one or more fields on the form, but	1. Completely and correctly fill all the fields on the entire post form.	All of the fields on the post form are cleared, the page is redirected to the same page as before the user has filled out		

		changes mind about posting at that moment.	2. Press “clear” button.	anything.		
--	--	--	--------------------------	-----------	--	--

## IV. Code Review

↩ Reply

↩ Reply all

→ Forward

⚙

📁 Archive

🗑 Delete

🚩 Set flag

⋮

RE: Code Review Request

MH

Melissa Ho <mho10@mail.sfsu.edu>

9:36 PM

To: Chuting Yan

Hello Chuting,

I've reviewed your code for the post page. Here is the summary:

M4 Code Review - post.hbs. Reviewer - Melissa Ho. The code is very organized in terms of layers, divs, and TODO actions. The way Chuting set up her comments really helps for when merge happens. The back end team will clearly be able to see where the JS elements are needed to complete the page. Overall, really great job. The github commit comments she included are succinct, but descriptive enough for me to understand how changes were implemented and in what order. I suggested some minor detail changes on where certain text fields are positioned on the page so that the user has an easier time reading down the page. Great job, Chuting!

I've also made comments in the code itself via GitHub, as well as uploaded the summary on the commit.

Have a great weekend!

Best Regards,

Melissa

From: [Chuting Yan](#)

Sent: Thursday, May 12, 2022 9:12 PM

To: [Melissa Ho](#)

Subject: Code Review Request

Hi Melissa,

Hope you are doing well.

Can you please review the post page?

Thanks,

Chuting

The screenshot shows a GitHub pull request interface. At the top, the title is "M4 Code Review" with a "Browse files" button. The description states: "M4 Code Review - post.Nbs. Reviewer - Melissa Ho. The code is very organized in terms of layers, divs, and TODO actions. The way Chuting set up her comments really helps for when merge happens. The back end team will clearly be able to see where the 35 elements are needed to complete the page. Overall, really great job. The github commit comments are succinct, but descriptive enough for me to understand how changes were implemented and in what order. I suggested some minor detail changes on where certain text fields are positioned on the page so that the user has an easier time reading down the page. Great job, Chuting!". The pull request is from user "melissah717" committed 5 minutes ago. It shows "1 parent 3ee787" and "commit 459586bf359839e872d784fac240309e2516". Below this, it says "Showing 1 changed file with 36 additions and 1 deletion." and "Split Unified" view options. The file "application/views/post.Nbs" is selected, showing a diff. The diff highlights changes in the HTML structure, including the addition of a "CODE REVIEW" comment and the restructuring of the "post-item" container to include a "form-box" for the form and a "banner" at the bottom. The code uses Bootstrap classes like "grid-container" and "form-box".

29	*
30	* <!--HELLO-->
31	* I really like how this portion of the code is set up.
32	* It's easy to read and understand. However, the text in front of the
33	* banner is a little hard to read, I recommend changing the color or
34	* font size.
35	* -->
36	*
37	</div class="tu-postitem">Post item for sale or list jobs</div>
38	</div>
39	*
40	* <!--HELLO-->
41	* A suggestion I have is to put the labels denoted with * are mandatory
42	* Line underneath the header instead of by itself on the right. That way, it's
43	* easier for the user to read as he or she browses down the page.
44	* -->
45	*
46	</div class="item float-right">
47	* Labels denoted with cspans=2/3spans= are mandatory.
48	</div>
49	*
50	* <!--HELLO-->
51	* In my opinion, the labels for the forms should be a little larger to look like headings.
52	* -->
53	*
54	</div class="title">
55	<label for="name">Item/Job Name cspans=2/3spans=1</label>
56	<input id="name" type="text" name="name" required />
57	*
58	<label for="instructions">Description cspans=2/3spans=1</label>
59	<textarea id="instructions" rows="3" required/>Enter
60	</div>
61	* <!--HELLO-->
62	* I like the feature that limits users to a certain number of characters for the description.
63	* It's a good security implementation. However, I feel as if the character is too little. Maybe you could
64	* change it to 256 characters, to keep it short but still allow the user to say what he or she needs.
65	* -->
66	<p><small>100 characters</small></p>
67	</div class="btn-block">
68	<button type="submit" class="certain" href="/profile/submit/>Submit
69	</button>
70	</div>
71	</div>
72	<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.min.js"
73	integrity="sha384-1lZ7CklRt6VnIt1wJ2RK7oquIdU7hK9EOYga6hjFwpqvgzHXWgQOhvm8xrs3q8" crossorigin="anonymous">



## V. Self-check on best practices for security

Asset to be protected	Types of possible/expected attacks	Your strategy to mitigate/protect the asset
Password	Unsecured accounts	Encryption, track system usage, two factor authentication. Using session handling.
Database	SQL injection	Input validation and queries using prepared statements.
User Accounts	Spam or robot users.	A valid SFSU email is required to register.
Postings	Dangerous postings, threats, scams	Administrator's validation before each posting, giving a window of 24 hours.
Confidential data	Data leak	Using HTTPS because it is a secure protocol.
Search		Limit search bar input to 40 alphanumeric characters

## VI. Self-check of the adherence to original Non-functional specs

Non Functional Specifications	Status	Comments
Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0	On track	
Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers	Done	
All or selected application functions must render well on mobile devices	Done	
Data shall be stored in the database on the team's deployment server.	Done	
No more than 50 concurrent users shall be accessing the application at any time	On track	
Privacy of users shall be protected	On track	
The language used shall be English (no localization needed)	Done	
Application shall be very easy to use and intuitive	Done	
Application should follow established architecture patterns	Done	
Application code and its repository shall be easy to inspect and maintain	Done	
Google analytics shall be	On track	

used		
No e-mail clients shall be allowed. Interested users can only message to sellers via in-site messaging. One round of messaging (from user to seller) is enough for this application	Done	
Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.	Done	
Site security: basic best practices shall be applied (as covered in the class) for main data items	On track	
Media formats shall be standard as used in the market today	Done	
Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development	Done	
The application UI (WWW and mobile) shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Spring 2022. For Demonstration Only" at the top of the WWW page nav bar. (Important so as to not confuse this with a real application).	Done	