Milestone 4 Beta Launch, QA and Usability Testing and Final Commitment for Product Features (P1 list)

Software Engineering CSC648/848 Fall 2020

The Gator Store

Team #2

Giovann Fox (Team lead/ Backend Lead) - gfox@mail.sfsu.edu

Ramy Fekry (Github Master)

Beatriz Ribeiro (Frontend Lead)

Jessica Serrano (Frontend Member)

Ikenna Eke (Backend Member)

December 2, 2020

History Table

Date Submitted: December 8th, 2020	Date Revised:

Product Summary

Title of Product: The Gator Store

Priority 1:

- ➔ Unregistered users
  - Unregistered users can browse the home page
  - Unregistered users can register for an account
  - Unregistered users can view the product description
  - Unregistered users can see public posts
  - Unregistered users can search for the product by name, class number, Professor’s name, category, title, and/or description (unique to our product)

- ➔ Registered users
  - Can do everything an unregistered user can do
  - Registered users can login
  - Registered users can sign out
  - Registered users will be able to use their dashboard to edit, see their posts, and see messages
  - Registered users can contact the seller of the product they’re interested in

- ➔ Admin
  - Admin is required to approve all posts before they go live
  - Admin is able to delete inappropriate posts
  - Admin has a dashboard to approve and report posts.

URL to Product: http://35.163.149.5:3000/home.html

Usability Test Plan

Test Objectives

For this evaluation we will be testing our search function on The Gator Store website. The purpose of this test plan is to assess how well our function performs. By the end of this test we will know whether our product is usable, user friendly, and efficient. Our test objectives for the usability test plan will test the user's ability to search products within our website. The user should also be able to filter through products by searching the products description, title, professor and course number. We do not expect the website to be perfectly functioning and we will use this usability test plan to improve it. We look forward to feedback from our users in order to further enhance our product.

Test Background and Setup

Our test is designed around the Gator Store website and will determine the users ability to search products they're interested in buying. There are many products in the Gator Store and the user should be able to filter through them in order to find the product they like. The website has to be functional on multiple browsers as well. The browsers we tested on are Google Chrome and Mozilla Firefox. Students, faculty and staff will be able to search our website.

To start the usability testing we will use the Google Chrome browser. The user will begin at the Gator Store's home page on the website. Next we assign the tasks for the user to search for different products. We will not assist the user, they should be able to figure it out on their own if we designed the product well. If not, then we would need to work on it more to make it user friendly. When the user is done with the tasks we will use the criterias in the usability task description table and the Likert test to grade whether our function passes the usability test.

The user will test the website at the Gator Store's URL: http://35.163.149.5:3000/home.html. They should immediately see the search bar at the top, common with most websites like amazon for example. The user can check the entire stock of products by searching with the "All" category filter. Or the user can filter through the six categories. Furthermore, the user can search by title, description, professor and course number. The usability test will be complete if the user can successfully navigate to certain products we give them. An example task we might give them is to search for all books in the store and then to search for professor Fox's book. The test would be complete if they are able to filter through the search and find professor Fox's textbook.

Usability Task Description

Task	Description
Task	Search posts in the Book category
Machine state	Search posts loaded
Successful completion of criteria	Shows related search results depending on what was queried
Benchmark	Completed in 15 seconds

Effectiveness

We would measure effectiveness by the amount of people who completed the task and people who did not complete the tasks. If more people completed the tasks than the people who didn't then we will determine our product to be effective. We want this percentage to be above 90% so we know the majority of people passed the usability test.

Efficiency

We would measure efficiency by measuring the time it took for them to complete the tasks. If it took them over 15 seconds to search for posts in the Gator Store, we would have to shorten that time to enhance our product. Besides measuring the time, we would also measure the amount of errors our users had. We would like the amount of errors to be as small as possible in order to have an efficient website.

Likert Scale

Mark “X” for any applicable columns

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
I was able to find search bar easily					
I was able to search for tasked postings					
I found the related search results					
I found results in a reasonable amount of time					
I encountered no bugs while searching					

This is subjective testing and therefore has no right or wrong answers. It will only be used to provide an opinion on the Gator Store.

Comments field:

QA Test Plan

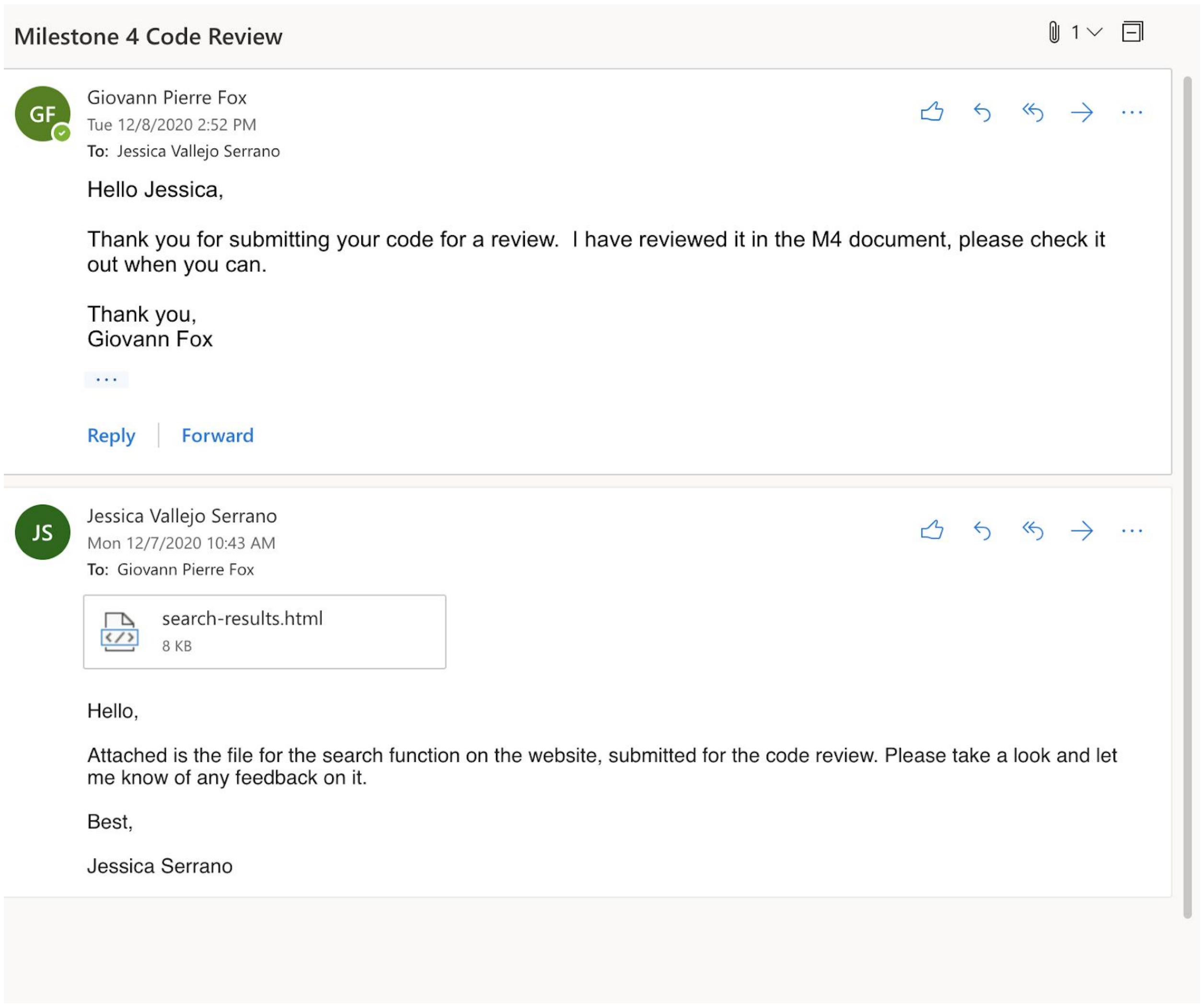
- Test objectives
    - We will test the Gator Store's search feature for users as well as the accuracy of the search results.

- Hardware and Software setup
    - Hardware:
        - MacOS Catonlina
    - System setup:
        - The setup consists of using Google Chrome to run the website where we will start on the home page. The home page as with the other pages has the search bar at the top for convenience. Next we will text on a Mozilla Firefox browser where we will again begin on the homepage of the website. Once the website runs we check if the search bar and search button are present and if the search results exist.
    - The intended users are the students, faculty and staff of San Francisco State University.
    - URL to the Gator Store: http://35.163.149.5:3000/home.html
- Feature to be tested:
    - Search function

- QA test plan
    - When performing the test, we followed the table and tested the search function on Google Chrome and Mozilla Firefox.

<u>Test #</u>	<u>Test Title</u>	<u>Test Description</u>	<u>Test Input</u>	<u>Expected Correct output</u>	<u>Chrome</u>	<u>Firefox</u>
1	Search Bar Exists	The search bar should be visible and on the page at all times	N/A	A visible search bar	PASS	PASS
2	Search Button Works	The search button should update the main page when content is passed through it with a category	Search “Biology” in the search bar and press enter or search icon	An updated search results page with a Biology book showing	PASS	PASS
3	Search Results are Relevant	The search results page should be on par with given search input	Search “jacket” and press search button	Jacket should pop up in search results page (1 post)	PASS	PASS

Code Review



Milestone 4 Code Review

Giovann Pierre Fox
Tue 12/8/2020 2:52 PM
To: Jessica Vallejo Serrano

Hello Jessica,

Thank you for submitting your code for a review. I have reviewed it in the M4 document, please check it out when you can.

Thank you,
Giovann Fox

Jessica Vallejo Serrano
Mon 12/7/2020 10:43 AM
To: Giovann Pierre Fox

search-results.html
8 KB

Hello,

Attached is the file for the search function on the website, submitted for the code review. Please take a look and let me know of any feedback on it.

Best,

Jessica Serrano

Review comments in red

```
<!DOCTYPE html>
// Great job Jessica! Code is very well done and just a few things
need improvement.
// Make sure to include basic headers and in-line comments.
// Header comments should include author of code, input/output, and
what the code is for
// Good use of in-line comments
<html lang="en">
   <head>
       <title>Search Results</title>
       <meta charset="UTF-8">

// Good use of inline comments for the different bootstraps
        <!-- Bootstrap JS -->
       <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
       <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
       <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
       <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>
       <!-- bootstrap-select CDN -->
       <script
src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-select/1.13.1/js/bootstrap-selec
t.min.js"></script>

       <link rel="stylesheet" type="text/css" href="assets/style.css">

       <!-- Bootstrap CSS -->
       <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
       <!-- bootstrap-select stylesheet -->
       <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-select/1.13.1/css/bootstrap-sel
ect.min.css">

       <script>
// Can use an inline comment here explaining what this section
does
         function letternumber(e){

         var key;
         var keychar;
```

```
          if (window.event)
             key = window.event.keyCode;
         else if (e)
             key = e.which;
         else
             return true;
         keychar = String.fromCharCode(key);
         keychar = keychar.toLowerCase();
// Good use of inline comments, tells what each section of code
is
          // control keys
         if ((key==null) || (key==0) || (key==8) ||
             (key==9) || (key==13) || (key==27) )
             return true;
          // alphas and numbers
         else if ((("abcdefghijklmnopqrstuvwxyz0123456789").indexOf(keychar) > -1))
             return true;
         else
             return false;
         }
     </script>
   </head>

   <body>
     <!-- fixed top navbar -->
     <nav class="navbar navbar-expand-lg fixed-top navbar-dark bg-dark">
       <a class="navbar-brand" href="/home.html"><img src="assets/logo2.png"
width="120" alt="logo"></a>
       <div class="title" >
           <h4 style="text-align:center; color:white; font-size:10px;">SFSU Sofware
Engineering Project - CSC648-03 Team 02</h4>
       </div>
       <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent"
aria-expanded="false" aria-label="Toggle navigation">
       <span class="navbar-toggler-icon"></span>
       </button>
       <form class="form-inline my-2 my-lg-0 navbar-form" method="get" name="search"
action="/search-results.html">
          <div class="input-group input-group-search mx-auto">
             <div class="input-group-prepend">
```

```
                <select class="form-control selectWidth"  id="selectionType"
name="categoryKey">
                   <option value="">All</option>
                   <option value="books">Books</option>
                   <option value="electronics">Electronics</option>
                   <option value="furniture">Furniture</option>
                   <option value="clothings">Clothing</option>
                   <option value="supplies">Supplies</option>
                   <option value="household">Household</option>
                   <!-- <div class="title" >
                   </br>
                       <h4 style="text-align:center; color:white; font-size:20px;
margin-top:20%;"> CSC648-03 Team 02</h4>
                   </div>  -->
                </select>
                <input type="search" id="search_bar" class="form-control"
placeholder="Search..." aria-label="Search" aria-describedby="search-button-addon"
maxlength="40"  onKeyPress="return letternumber(event)" name="searchKey">
             </div>
             <div class="input-group-append">
                <button class=" btn-grey" type="submit"
id="search-button-addon">&#128269;</button>
             </div>
          </div>
       </form>
       <div class="collapse navbar-collapse" id="navbarSupportedContent">
          <ul class="navbar-nav">
             <li class="nav-item">
                <a class="nav-link text-light" href="/about/about.html">About</a>
             </li>
             <li class="nav-item">
                <a class="nav-link text-light" href="post.html">Post</a>
             </li>
             <li class="nav-item">
                <a class="nav-link text-light" href="login.html">Login</a>
             </li>
             <li class="nav-item">
               <a class="nav-link text-light" href="signup.html">SignUp</a>
            </li>
          </ul>
       </div>
    </nav>
```

```
     <h2 id="key" class="pt-5 text-center"></h2>
     <div id="count" class="col m-3"></div>
// Nice job with inline comments here
     <!-- display results of search -->
     <div id="result" class="row row-cols-1 row-cols-md-2 p-2"></div>

     <!-- bottom navbar -->
     <nav class="navbar bottom navbar-expand-lg navbar-light bg-light
justify-content-center">
       <ul class="navbar-nav justify-content-center">
           <li class="nav-item">
               <a class="nav-link" href="#">Contact</a> <!-- dummy link-->
           </li>
           <li class="nav-item">
               <a class="nav-link" href="#">Terms</a> <!-- dummy link-->
           </li>
           <li class="nav-item">
               <a class="nav-link" href="#">Privacy</a> <!-- dummy link-->
           </li>
       </ul>
     </nav>
       <script>

         $("div[id='result'").ready((e) => {
           handleSubmit(e)
         });

         async function handleSubmit(e) {
               // e.preventDefault()
               const url = new URL(window.location.href);
               console.log(url.searchParams.get("searchKey"));
               $('#search').val(url.searchParams.get("searchKey"));


               let response = await
fetch('http://localhost:3000/search/results?categoryKey='+url.searchParams.get("catego
ryKey")+'&searchKey='+ url.searchParams.get("searchKey"));

               //  let response = await
fetch('http://35.163.149.5:3000/search/results?categoryKey='+url.searchParams.get("cat
egoryKey")+'&searchKey='+ url.searchParams.get("searchKey"));
```

```
             const searchResult = await response.json(); // read response body as text

             const searchKey = url.searchParams.get("searchKey");
             document.getElementById("key").innerHTML = 'Search Results for "' +
searchKey + '"';
//Can use inline comment here explaining that this is what gets
the search results
             var count = 0;
             searchResult.results.forEach(item => {
               count++;
               document.getElementById('result').innerHTML +=
                 '<div class="col mb-4">' +
                   '<div class="card" style="width: 18rem;">' +
                     '<img class="card-img-top" src="' + item.image + '" alt="' +
item.title + '" style="width: 18rem; height: 18rem;">' +
                     '<div class="card-body">' +
                       '<h3 class="card-title">' + item.title + '</h3>' +
                       '<p class="card-text">Instructor: ' + item.instructor +
'<br>Course: ' + item.course + '<br>$' + item.price + '</p>' +
                       '<a href="/item-details.html?id=' + item.id + '"
target="_blank" class="btn btn-primary">See more details</a>' +
                     '</div>' +
                   '</div>' +
                 '</div>'
             });

             document.getElementById('count').innerHTML +=
               'Number of results: ' + count;

         }
       </script>

   </body>

</html>
```

Self Check on Security

Major Assets

- Posting Information - This is a top priority. Any threats of confidentiality of our system could be devastating without preliminary measures. We are protecting our Databases with an encrypted password that one cannot easily guess.
- User information - This is another top priority. To protect user information we are using bcrypt to hash users passwords and protect their vital personal information.

Input data validation is being taken into account and we validate search bar input for up to 40 alphanumeric characters. We also tell users they must include “sfsu.edu” in their email for a correct registration. Only students, faculty or staff should be able to register with their school email.

Self Check on Non-Functional Specs

1. Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0 (some may be provided in the class, some may be chosen by the student team but all tools and servers have to be approved by class CTO).

   DONE

2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers

   DONE

3. All or selected application functions must render well on mobile devices

   DONE

4. Data shall be stored in the database on the team's deployment server.

   ON TRACK

5. No more than 50 concurrent users shall be accessing the application at any time

   DONE

6. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.

   DONE

7. The language used shall be English (no localization needed)

   DONE

8. Application shall be very easy to use and intuitive

   DONE

9. Application should follow established architecture patterns

DONE

10. Application code and its repository shall be easy to inspect and maintain

    DONE

11. Google analytics shall be used

    DONE

12. 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

    ON TRACK

13. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.

    DONE

14. Site security: basic best practices shall be applied (as covered in the class) for main data items

    DONE

15. Media formats shall be standard as used in the market today

    DONE

16. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development

    DONE

17. The application UI (WWW and mobile) shall <u>prominently</u> display the following <u>exact</u> text on all pages *"SFSU Software Engineering Project CSC 648-848, Fall 2020. For Demonstration Only"* at the top of the WWW page. (Important so as to not confuse this with a real application).

    DONE