SWAMP

SW Engineering CSC648/848 Spring 2020

TEAM 04 Global - Milestone 04

May 18, 2020

Team Lead: Benjamin Lewis - email:
benjaminlewis984@gmail.com
Git Master: Weerachai Poorakkiat
Front-End Lead: Dang Le
Back-End Lead: William Lew
Front-End Dev: Onubulachi-Abigail Wami
Back-End Dev: Kevin Huynh

Table of Contents

-Product summary-

Swamp, an online marketplace for the SFSU community by the SFSU community.

Committed functions:

1. Unregistered users **shall**:
    I. Be able to register ***DONE***
    II. Be able to browse site content ***DONE***
2. Registered users **shall**:
    I. Be able to remove listing ***IN PROGRESS (BACKEND DONE)***
    II. Be able to post media for sale ***DONE***
    III. Be able to logout ***DONE***
    IV. Be able to login ***DONE***
    V. Be able to message media seller ***IN PROGRESS (BACKEND DONE)***
    VI. Be able to request to purchase ***IN PROGRESS (BACKEND DONE)***
    VII. Be able to approve purchase request ***IN PROGRESS (BACKEND DONE)***
    VIII. Be able to deny purchase request ***IN PROGRESS (BACKEND DONE)***
3. Administrative users **shall**:
    I. Be able to login ***DONE***
    II. Be able to logout ***DONE***
    III. Be required to approve posting ***DONE***
    IV. Be able to delete posting ***DONE***
    V. Be required to view user reports ***IN PROGRESS***
    VI. Be able to ban user ***DONE***
    VII. Be able to unban user ***DONE***

Our product contains a unique function that caters directly to SFSU students and faculty by means of utilizing a search feature that allows for site content to be searched by a class filter. This class filter allows users of Swamp to find/ post items based on a specific SFSU class (i.e CSC648).

URL to product - http://18.191.184.143:3000/

- Usability test plan -

- ***Test objectives:***

USE CASE: We are using Emma's use case (defined in earlier milestones) as a means of testing our user friendly site. We are trying to make sure that the site, upon entering, is easy to traverse so there is no confusion for the user. First when the site is loaded the user should be allowed to browse the page but to download an item the user should be taken to a registration page.

USER FRIENDLY: When creating our UI we made sure that our site was easy on the eyes (so the user can see and read everything with ease) so we picked out a color palate that would work and made sure all words on the website were a readable size font. This was a bit different than our UI mockups since those were a little crowded making it harder for a user to see what to do.

REGISTERING: Once signed in to the website the user will be registered and then taken to the login screen. Once logged in the user will be able to download content and contact the seller if the item they want to purchase is not free. Also the registered user will now have a user dashboard to view their items that they posted as well as user data like email, name, username. Also once the user is logged in the login and sign up button disappears from the nav-bar. Replacing login and sign up on the nav-bar is the user dashboard and a logout button.

SITE INFORMATION: Also throughout the site we display messages that allow the user to understand what is going on. For example, if the user tried to sign up with an email that isn't ending with @mail.sfsu.edu they will be prompted to put in a proper email. Also upon successfully logging in there will be a message that tells the user that they were logged in successfully.

Test background and setup -
When starting the creation of the Swamp e-commerce site the team got together to discuss things we all found to be of vital importance to our site. We had many SCRUM meetings that resulted in us creating use cases for inevitable reasons and actions people might do on our site most commonly.

separate paragraphs each covering: System setup, starting point, who are the intended users, URL of the system to be tested ad what is to be measured (for M4 focus only on user satisfaction evaluation e.g Likert tests). Up to 1 page

Usability Task description: When going through usability testing, our team simulated a new user entering the site in order to download an item of their choosing. Upon entering the site the user is brought to the main page which displays our site name and explains briefly what our site is about. From there our users went directly to the browsing page where they were able to look through our sites posted items. They would also use the search bar in order to narrow their search for finding the right digital content they wanted. They will need to register to our site if they want to download some of our site's content.

These are the instructions to be given to the tester: Write them in a separate paragraph in the format of instructions for the usability tester what to do e.g. describe the task testers do before filling out the Likert questionnaire and do the assessment, Then: - Say how would you measure effectiveness (in max 3 lines of text) - Say how you would measure efficiency (in max 3 lines of text) - Provide 3 Lickert scale questions getting user satisfaction after the above task has been performed, in a proper format and

grading answer scale as it is to be used by reviewer (check class slides) – 3/4 page

	Rating 1-5 (1=Strongly Disagree, 3=Neutral, 5=Strongly Agree)				
Statements	User Responses				
I found the signup process easy to follow.	5	5	5	3	5
It is simple to browse the website.	5	5	5	5	5
Uploading content to the website is a simple process.	3	1	5	3	3
Downloading content from the website is easy.	5	5	5	3	5
The website is easy to use.	5	3	5	3	5
Overall, the website is built with an effective user interface.	5	5	5	3	5
Your comments are greatly appreciated. Please tell us what you like and how to improve our product.					

- QA test plan -

Test objectives: - Search bar feature. Users shall be able to post items to the site.

HW and SW setup (including URL):

Feature to be tested - The search bar should be able to filter the site's content from the user's input.

QA Test plan:

				Browser Results		
#	**Description**	**Input**	**Expected output**	**Firefox**	**Chrome**	**Safari**
1	Test %like in search bar	Type "yee" in search bar	Get 1 result, with all having the word "yee" in the title	PASS	PASS	PASS
2	Test specific type of media, images	Type "grass" in search bar and select "images" in dropdown menu	Get 2 results, all of image type having the word apple in the title	PASS	PASS	PASS
3	Test whether item searched matches details	Type "grass" in search bar and select "images" in dropdown menu. Click the first image.	Get the details page of the item clicked. Title should match the card clicked in search. (Grass Field)	PASS	PASS	PASS

				Browser Results		
#	**Description**	**Input**	**Expected Output**	**Firefox**	**Chrome**	**Safari**
1	Test ability to upload to the website	Navigate to the Dashboard and Click the “+” icon to upload. Fill out the upload form.	Pending request for newly uploaded content will appear in Dashboard	PASS	PASS	PASS
2	Test ability to delete a post from the website (**IN PROGRESS)**	Navigate to the Dashboard view “Current Posts.” Click on the Delete button of the post that was uploaded in previous test	Current Posts field in Dashboard will be empty.	FAIL	FAIL	FAIL
3						

- Code Review-

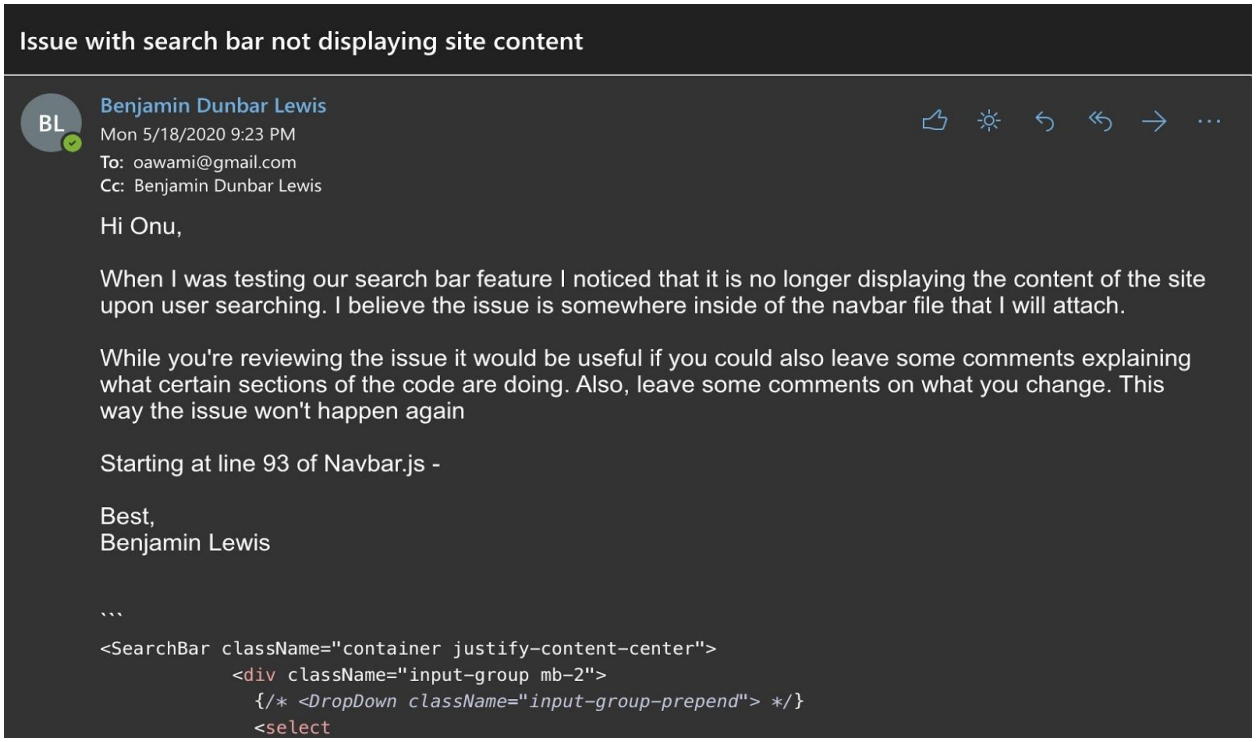
Coding Style:

Our style consists of a few different things:

1 - All of our variables, class names, function names, and file names are descriptive to what their purpose is. Our team follows an OOP (Object-Oriented Programming) style meaning each function, class, file, and variable is consistent to what its purpose is. Each function encapsulates what the purpose of the feature is to the corresponding classes, files and variables that pertain to that feature. For example, our login feature will have nothing to do with browsing the site though it will define variables that pertain to a wider scope. This OOP style makes it easy for our team to traverse our code so that the team is able to easily identify the purpose of certain sections. This allows the team to maximize the attempts to pinpoint areas the team works on like finding bugs, usability testing, adding features, etc.

2 - Syntax: All of our variables, files, functions and classes follow a camel-case syntax (exampleVariable = true).

-Code review Sample-



Issue with search bar not displaying site content

Benjamin Dunbar Lewis
Mon 5/18/2020 9:23 PM
To: oawami@gmail.com
Cc: Benjamin Dunbar Lewis

Hi Onu,

When I was testing our search bar feature I noticed that it is no longer displaying the content of the site upon user searching. I believe the issue is somewhere inside of the navbar file that I will attach.

While you're reviewing the issue it would be useful if you could also leave some comments explaining what certain sections of the code are doing. Also, leave some comments on what you change. This way the issue won't happen again

Starting at line 93 of Navbar.js -

Best,
Benjamin Lewis

```
<SearchBar className="container justify-content-center">
            <div className="input-group mb-2">
              {/* <DropDown className="input-group-prepend"> */}
              <select
```

```
            id="category"
            onChange={(e) => setCategory(e.target.value)}>
            <option value="all" selected>All</option>
            <option value="document">Documents</option>
            <option value="image">Images</option>
            <option value="audio">Audio</option>
            <option value="video">Video</option>
          </select>
          {/* </DropDown> */}

          <input
            id="userInput"
            class="form-control"
            type="text"
            aria-label="Text input with dropdown button"
            placeholder="Search by title.."
            value={query}
            onChange={(e) => setQuery(e.target.value)}
          ></input>

          <button
            className="btn btn-dark"
            onClick={() => {
              value.setProducts(category, query);
              history.push("/result");
            }}
          >
            <i class="fas fa-search"></i>
          </button>
        </div>
      </SearchBar>
```

Issue with search bar not displaying site content

OW

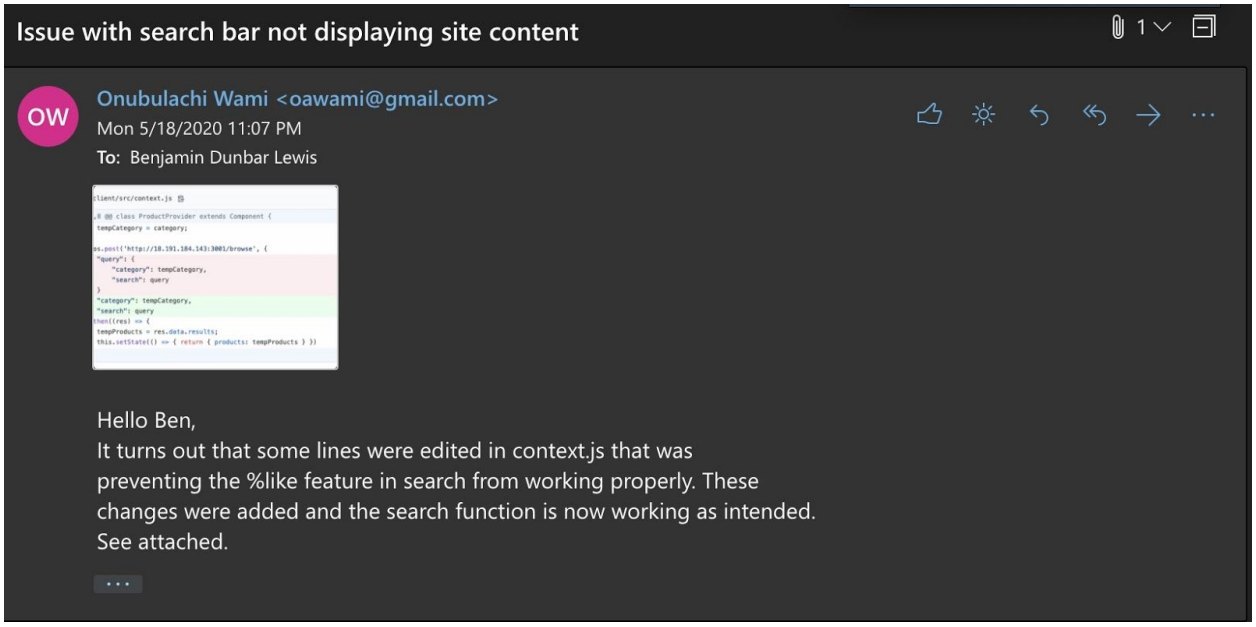
Onubulachi Wami <oawami@gmail.com>
Mon 5/18/2020 9:45 PM
To: Benjamin Dunbar Lewis

Hello Ben,
Thank you for notifying me about the issue with the search bar. I will take a look into it as it is a critical issue. The issue was most likely produced when redesigning the navigation bar, and moving it out of the home page. I went ahead and added some comments to the code. Please let me know of any other issues you find.

```
{/* Search bar */}
{/* Search bar is using bootstrap so the size will shrink when the browser shrinks in order to accommodate mobile devices (flex-shrink-1). The search bar also follows common practice in websites by always being present in the navbar and being centered. The dropdown menu allows the user to select a specific category for searching content on the website. */}
{ navSearch ?
<ProductConsumer className="flex-shrink-1 align-content-center">
{(value) => (
<SearchBar className="container justify-content-center">
<div className="input-group mb-2">
<select
id="category"
onChange={(e) => setCategory(e.target.value)}>
<option value="all" selected>All</option>
<option value="document">Documents</option>
<option value="image">Images</option>
<option value="audio">Audio</option>
<option value="video">Video</option>
</select>

<input
id="userInput"
class="form-control"
type="text"
aria-label="Text input with dropdown button"
placeholder="Search by title.."
value={query}
onChange={(e) => setQuery(e.target.value)}
></input>
{/* Search results are generated through setProducts which is located in src/context.js. T
his takes the query from the searchbar and generates an array of items related to the query
using %like on the title of the items posted to the website. The generated array will display
```

```
on the /results page when redirected.*/}
<button
className="btn btn-dark"
onClick={() => {
value.setProducts(category, query);
history.push("/result");
}}
>
<i class="fas fa-search"></i>
</button>
</div>
</SearchBar>```
```



-Security-

Password encryption: In order to make sure our users are protected we used bcryptjs in our backend. Bcrypt is a tool for allowing credentials like passwords to be secured in our database. It turns the user's password into an encrypted hash value that is essentially generated bytecode that can only be reversed through its decryption. The hash value created is sort of like a virtual

fingerprint that is unique so even if two passwords are the same the hash value will be different.

- **Threats**: Just there are people who create code for the good of others there are always those who try to exploit it. The hackers have many methods of exploiting protected things like passwords called attacks. Our encryption algorithm will protect against attacks like rainbow table attack, brute force, Man in the Middle, Drive-by attack, Cross-site scripting, etc. The list goes on and on but with encryption our team is able to keep even well equipped hackers at bay.

Named Placeholders: To prevent our database from being the victim of a malicious attack and putting our users' information at risk, we've adopted the usage of named placeholders in our sql queries. Using named placeholders is akin to replacing all the parameters in the brackets with their respective positions in the query string and escaping them.

Search bar validation: Inside off our search bar we only allow users to be able to type up to 40 characters. Among these 40 characters only alphanumeric characters are allowed to be entered. This ensures that only properly formed data is entering the search bar. This will prevent malformed data from entering our database that could potentially give our site malfunctions. Only allowing alphanumeric data to enter our site is a form of syntactic validation which forces inputs from the user to enter correct syntax. We correct any data before it gets too deep into our code so that the site can spot errors early before causing damage.

-Non-functional specs-

I. 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). ***ON TRACK***

II. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers. ***ON TRACK***
III. Selected application functions must render well on mobile devices. ***ISSUE***
IV. Data shall be stored in the team's chosen database technology on the team's deployment server. ***DONE***
V. Full resolution free media shall be downloadable directly, and full resolution media for selling shall be obtained after contacting the seller/owner. ***ON TRACK***
VI. No more than 50 concurrent users shall be accessing the application at any time. ***ON TRACK***
VII. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users. ***DONE***
VIII. The language used shall be English (no localization needed). ***DONE***
IX. Application shall be very easy to use and intuitive. ***DONE***
X. Google analytics shall be used. ***ISSUE***
XI. No e-mail clients shall be allowed. ***DONE***
XII. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI. ***ON TRACK***
XIII. Site security: basic best practices shall be applied (as covered in the class) for main data items. ***DONE***
XIV. Media formats shall be standard as used in the market today ***ISSUE***
XV. Media material shall be either free or for sale, as determined by media owner. ***ON TRACK***
XVI. Each media material shall have its license info as one of the following: a) free use and modification; b) free but only allowed for SFSU related projects; c) for sale. ***ON TRACK***
XVII. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development. ***DONE***
XVIII. The website shall <u>prominently</u> display the following <u>exact</u> text on all pages *"SFSU Software Engineering Project CSC 648-848, Spring 2020. For Demonstration Only"* at the top of the WWW page. (Important so as to not confuse this with a real application). ***DONE***