CSC 648-848 01 Fall 2019

Gator Trader

Team 05

Milestone 04:

Overview & QA Test Plans

- Grayson Mical - grayson.mical@gmail.com
- Hayato Waki
- Samjot Singh
- Anmol Gondara
- Daniel Gutierrez
- Anton Abramson

Date	Description
12 / 11 / 2019	First Draft

Product Summary

Welcome to **Gator Trader**, A website designed by students, for students; so you know the product is something the demographic will want and need. This website offers a safe and reliable way for students at San Francisco State University to both buy and sell items that are essential to being a student in 2020. Below are our priority one functions:

- Users shall be able to search for desired products
- Users shall be able to view the results
- Users shall be able to view specific products for details
- Users shall be able to contact the seller
- Users shall be given access to the Userdashboard once an account is created
    - Users shall be able to view messages
    - Users shall be able to view personal items posted for sale
- Users shall be able to post items for sale once an account is created
- Users shall be able to filter by price on search results

Our product separates itself from other sales platforms for two main reasons; we are only available to SFSU students and each transaction will be done in a public location marked by our platform, known as Blue Pole locations. Blue pole locations are a unique safety feature designed to keep students safe and near others while conducting sales and trades. Our product url is http://13.52.75.46/ . Please note this url does change from time to time, please check our github for the most up-to-date URL. https://github.com/CSC-648-SFSU/csc648-fall2019-Team05

Usability Test Plan

Test Objectives

Our group will be testing our search feature. This is the feature we wish to test because it demonstrates our backend and frontend design working as one. There are several features of search to inspect; we will show a successful search, a category specific search, the results of a blank search, and so on. We are measuring effectiveness by our likert scale prompts-

Test Background and Setup

System Setup

Team 05 must ensure our server is up and running, as well as launching from the correct branch on our repo. We will be demonstrating on any system as our team designed our project to function properly on any device, any system.

Starting Point

Our design is meant to maximize intuitive behavior. As such, the moment our website is accessed the feature is available to test. Moreover, you are able to access the search feature from any of the primary listing pages.

Intended Users

Our users are students at SFSU. This covers a wide range of people, but by in large the students at SFSU are tech savvy enough to navigate Amazon.com, and therefore should be able to easily navigate our website.

URL

As mentioned in the product summary, our product url is http://13.52.75.46/ . Please note this url does change from time to time, please check our github for the most up-to-date URL. https://github.com/CSC-648-SFSU/csc648-fall2019-Team05

To Be Measured

For our feature we want to test usability and expected result comparison for search. Our prompts for the Likert tests will be:

The search feature gave the result(s) I expected
The search feature is easy to use
The search by category yielded correct results
A blank search yielded results for you to continue browsing

Usability Task Description

Please,

- Head to Gator Trader via the provided URL (http://13.52.75.46/)
- Locate the search bar, use it to search for a computer
- From wherever you'd like, use the search bar, filter by Furniture category, and find a couch
- Search for something not likely to be on the site
- Search for something not likely to be in a specific category using the category search

Lickert Subjective Test

1. The search feature gave the result(s) I expected
2. The search feature is easy to use
3. The search by category yielded correct results
4. A blank search yielded results for you to continue browsing

We will measure effectiveness and efficiency by taking the average of these tests, as well as the standard deviation.

QA Test Plan

The QA test plan for the search feature will be played out as follows.

TEST #	Title	Description	Input	Expected	Results
1	Specific Item	Use search bar at top of page to search for ‘computer’ with categories filter unchanged	‘computer’	An item listing page with computers as result	Pass/Pass
2	Blank Search	Without adding anything to the search bar, search. Describe results	None		Pass/Pass
3	Category Search	At the top of the page to the left of the search bar is the categories filter. Click the filter and select furniture. Search for couch	‘couch’	An item listings page with couches for sale on our site.	Pass/Pass
4	Incorrect Search	Search for ‘zzzzzzzz’ , an item that does not exist for sale. Describe the results	‘zzzzzzzz’	An item listings page populated with items.	Pass/Pass

Code Review

Our code review is done by in large by our team leads and Github manager. As we broke our code into the MVC model it set our group back in time but forward in project clarity. We also transitioned into some standard naming conventions for our variables, functions, and file names. After our second milestone we adapted our prefered language (JavaScript, CSS/html, bootstrapped, node JS), which helped us all begin to contribute. Below is a snippet of our code used for the search bar. The search bar is found in the header of each page, and our project structure allows us to reuse that header on each page. The path to the header file is csc648-fall2019-Team05/application/gatortrader/view/partials/header.ejs
Partials is the location of the parts of our web page, an example of our coding style.

```
<!--
Navb
ar
-->
        <nav class="navbar navbar-default">
           <div class="container">
               <!-- Home logo button -->
               <div class="navbar-header">
                   <button type="button" class="navbar-toggle" data-toggle="collapse"
        data-target="#myNavbar">
                       <span class="icon-bar"></span>
                       <span class="icon-bar"></span>
                       <span class="icon-bar"></span>
                   </button>
                   <a class="navbar-brand" href="/">Gator Trader</a>
```

```
        </div>


        <!-- Collapsible menu -->
        <div class="collapse navbar-collapse" id="myNavbar">
            <ul class="nav navbar-nav navbar-right">
                <li><a href="/new">SELL</a></li>
                <li><a href="/about">ABOUT US</a></li>
                <li><a href="/signIn"><span class="glyphicon
glyphicon-user"></span>ACCOUNT</a></li>
            </ul>
        </div>


        <!-- Search form -->
        <div class="header-bar">


        <form action="/search" method="get" onsubmit="return formValidation()">


            <!--dropdown button for search-->
            <div class="input-group-prepend">
                <select name="category" class="custom-select"
id="inputGroupSelect01">
                    <option value="">Categories</option>
                    <% categories.forEach(function (category) { %>
                    <option value="<%=(typeof category != "undefined" ?
category["category_id"] : "")%>"><%= category["category_name"] %></option>
                    <% }) %>
                </select>
            </div>


            <div class="input-group mb-3">
```

```
                <!--Actual search bar-->
                <input id="search" class="form-control" type="text" name="search"
value="<%= (typeof searchTerm != "undefined" ? searchTerm : "") %>"
placeholder="Search..." maxlength="40">


                <!--Searchbar search button-->
                <div class="input-group-btn">
                    <button class="btn btn-default" type="submit"
onclick="this.form.submit()">
                        <i class="glyphicon glyphicon-search"></i>
                    </button>
                </div>
            </div>


        </form>
        </div>
    </div>
</nav>
```

Each section is labeled clearly so visitors can easily tell what they are looking at. Below are screenshots of our work together on Github branches.

Can see page view after creating new items on sell pa #76

Closed **wakkihaya** wants to merge 6 commits into `development_daniel_local` from `development_revise_daniel`

Conversation 0 | Commits 6 | Checks 0 | Files changed 13

wakkihaya commented 5 days ago

- Can see page view after creating new items on sell page
- Show the images on main page

wakkihaya and others added 6 commits 5 days ago

 `Can see page view after creating new items on sell page` 9cb2a24

 `show the images` 7a9165a

`Connected routes for messages and your_listings` ... ac45ea4

`Fixed meet-up points img route and cleaned up modal window` ... 7903506

`Implemented css to categories drop-down to align it with search bar.` ... 1802852

`Implemented a new header for users who have logged in and can navigat…` ... 38b2251

 **anmol2727** closed this 2 days ago

Sell page new items registered in DB #64

Merged **anmol2727** merged 1 commit into `development` from `feature/issue63` 23 days ago

Conversation 0 | Commits 1 | Checks 0 | Files changed 6

wakkihaya commented 23 days ago

issue

issue 63

what I did

- Alter the type of columns in database
- Save the new items on sell page to the database
- Show the new items on the main index.ejs

What I didn't do yet

- **image send button is not working now. **
  - if we choose the image, the image of the item on main page is showing nothing.
  - if we don't choose the image and post, the image of that is showing GT_LOGO.png

 `Sell page new items registered in DB` `e10392a`

 **anmol2727** merged commit `041a69f` into `development` 23 days ago Revert

Security Self-Check

Our major assets are low on account of no credit information being stored. It comes down to just three items to protect.

- User Passwords
- User login ID
- User email

We intend to secure the above information by encryption in the DB, however we have yet to implement this as we are still ironing out some of our P1 items. Seeing as this website is for demonstration purposes, no authentic personal information will be stored, and therefore encrypting the data is P1.5. We will be doing field validation to prevent JS injection on our site.

Non-Functional Specs Self-Check

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. Selected application functions must render well on mobile devices
DONE

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

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.
DELAY - as this is a demo site we instruct viewers to not provide any true personal details. As such we have other functional specs to complete before we get to this priority. Our final product may still have encryption.

7. The language used shall be English.
DONE

8. Application shall be very easy to use and intuitive.
DONE

9. Google analytics shall be added
On Track

10. No e-mail clients shall be allowed
DONE

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

DONE

12. Site security: basic best practices shall be applied (as covered in the class)
On Track, see (6)

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

14. The website shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Fall 2019. For Demonstration Only” at the top of the WWW page. (Important so as to not confuse this with a real application).
On Track