SW Engineering CSC648/848 Fall 2018

Gator Exchange

Team 11

Matthew Ho

Jason Chow

Dian Zhu

Riza Shrestha

Justin Shee

Ghulam Khan

Juan Valdez

"Milestone 4"

December 02 2018

Revision	Comment	Date
1		

Product Summary:

Gator Exchange is a website that serves as a buy and sell hub that caters towards students at San Francisco State University. Our website is unique in its ability to allow students to find what they need and sell what they want with ease of use. Postings on *Gator Exchange* are a great way to showcase items by student sellers, and provide an easy way of contact for potential buyers.

I. **All Users**
   A. All users are able to view recent postings
   B. All users are able to register to create an account which is used for contacting sellers and making new item posts.
   C. All users are able to search through items by category or name.

II. **Registered Users**
   A. Registered users are allowed to contact sellers to inquire about item postings
   B. Registered users are presented with a new posting form which allows them to create a new listing that includes a description, title, image, and price of the item.
   C. Registered users have access to a seller dashboard which contains a list of postings they've made and comments that have been made on their posts.
   D. Registered users can delete or modify their item listings

III. **Administrators**
   A. Administrators can browse the website and have the same functionality as a registered user.
   B. Administrators have their own admin dashboard which contains a list of postings that are pending for approval.
   C. Administrators can approve or disapprove postings.

URL: https://gator-exchange.netlify.com/

Usability Test Plan: Search

Test Objectives

Our test objective is to find out how usable is the search function through usability tests. These tests will identify if the search function is effective, efficient, and satisfactory. These three keys will be used as a metric to understand the usability of the search function. The function of search is to populate the page with similar to close results as the keywords inputted on the search bar. First, the search function must be able to consistently provide accurate results. Second, the search function must be efficient in time, effort, and has an easy to understand interface. Time is calculated as duration from start to finish of using the search function. Effort is calculated by ease of accessibility. Finally, satisfaction will be defined by the user on how they feel about the overall usability of the search function. The objective is to define where our search function feature is lacking and how we can improve on it with the information we gather from usability testing.

Test Plan

URL of system to be tested: https://gator-exchange.netlify.com/

System Startup

Our search function is usable on most operating systems and web browsers. To test simply use the URL above to be directed to gator exchange. Our search bar is prominently on the top of the website. It is indicated by a placeholder text “Search for your item”.

Starting point

From that point, users may enter an item they are looking for. For example, Books, Clothes, White Shirt, batteries, etc. After entering the information, the user is looking for, press enter or click on the search button. This will generate the search function and respond with populating the page with results of the user inputted text.

Intended User

The intended user of our search function is mostly SFSU students visiting our website, Gator Exchange. However, any user visiting the website is allowed to use our search function. This allows for users to actively search for items they would like to buy. Making an easy and effective way to search our extensive catalog of items.

Task to be accomplished

Task	**Description**
Search input	item or category in entered in search bar
Search	Search function running
Successful completion criteria	Results displayed on page
Benchmark	Completed in 30 seconds

Questionnaire

Place an X on one choice per question, Provide optional comments about your usability.

Question	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
It was easy to find the search bar					

Comments:

Question	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
It was easy to search for an item					

Comments:

Question	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The results of the search were accurate					

Comments:

3.) QA test plan

Test Objectives

Our test objective for QA testing is to be able to be able to test our search function performs to the specs of our software, or in this case our website. The idea is that we want to be able to test whether our search function to do certain things given specific parameters. First thing to test for is if our search results yield results despite the search parameter being empty as we wouldn't want people to be given no result in order to persuade them to take a look at some products. The second thing we want to test is that when we type in parts of what is being searched for we get some similar results that way if a word is misspelled the function doesn't give us a blank screen once again. Lastly, we want to be able to check if the search result is yielding proper results, we wouldn't want something along the lines of searching for shirt and having the first few searches come up as pants. This would be a rather large bug, but this could happen and if not tested for would make the website unusable. Events that are not tested for through the search function will be how the results are displayed, despite making use of the search function this is another function in itself. Another event not tested would be the checking if the category works as that in itself is another function which is combined with the search function in order to display results. In this test we want to make sure search works fine and without major bugs that could make the the website not perform properly.

HW & SW setup

URL of system to be tested: https://gator-exchange.netlify.com/

Hardware

Windows Laptop 8GB RAM, intel core i5

Mac Laptop 8GB RAM, intel core i5

Software

Using Latest 2 version of Firefox and Chrome

The website frontend will be run through Netlify

The website backend will be run through Heroku

Coded through VS Code through the use of React & Javascript

Feature to be tested: Search function

Test Plan

Test DB

Item 1: category = "clothes"; "title": "Bape Hoodie",

Item 2: category= "books"; "title": "CSC 648 Book",

Item 3: category= "books"; "title": "CSC 600 Book",

Item 4: category="clothes"; "title": "Plain White Shirt",

Item 5: category="miscellaneous"; search = "title": "4 AA Batteries",

Test

Input: enter “”
Output: Check that all five items from the database are displayed

Input: enter “light”
Output: Check that results consist exactly the following
- “new light blue handkerchief”
- “Lamp without lightbulb”

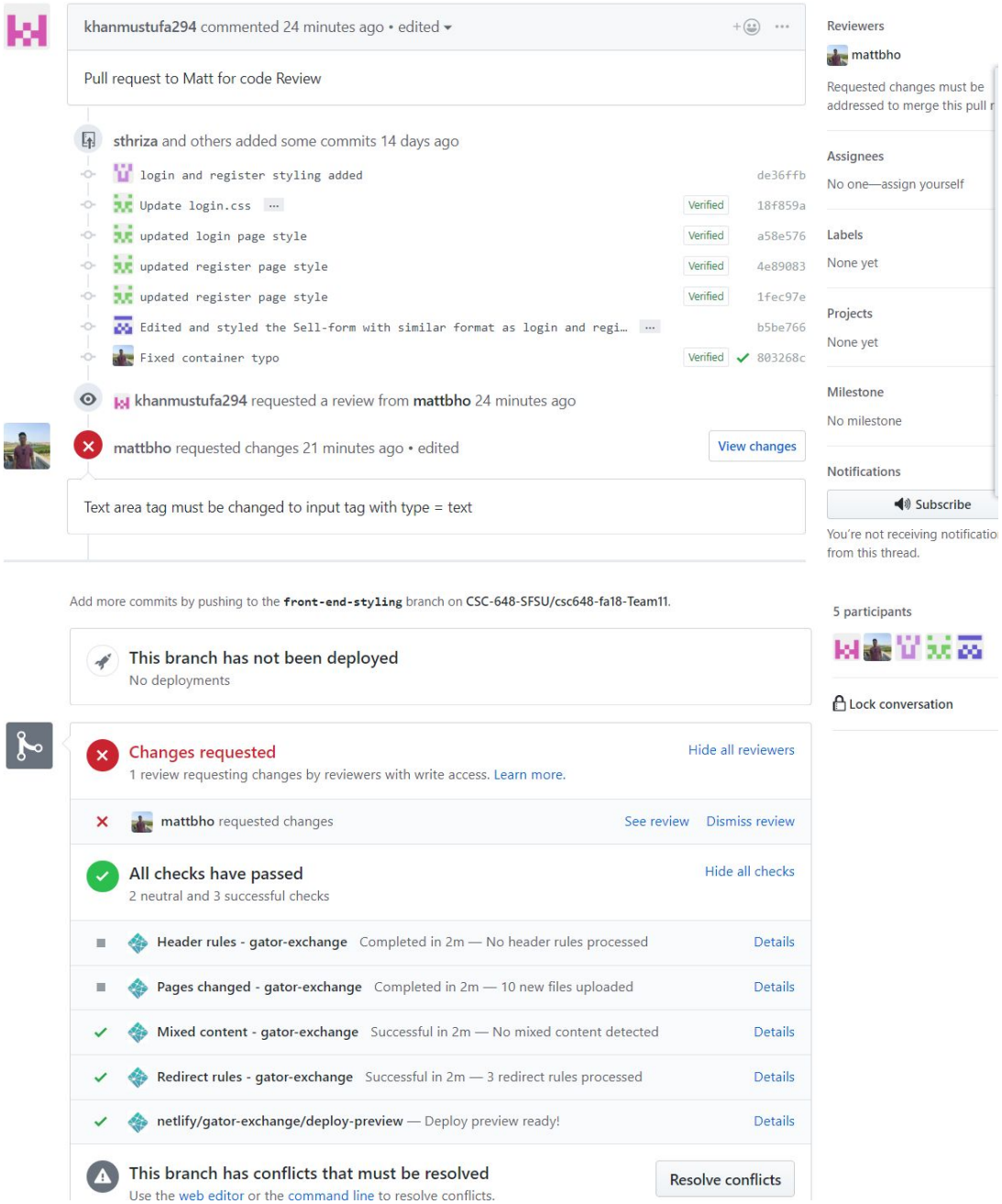
Input: enter “shir”
Output: Check that the result “plain white shirts” is displayed

Number	Title	Description	Test Input	Expected Output	Pass/Fail
1		Test empty string in search to check if all items will display	“”	Get all 5 results from the test DB	Pass
2		Test that % like will work by putting in a string that belongs in multiple posts and check if both results appear	“book”	Get the 2 results with books in the title namely: CSC 648 Book and CSC 600 Book	Pass
3		Another test for %like to check if someone doesn’t complete the word will their result still come up or if autofill exists	“shir”	Bring up the result of the plain white shirt	Pass

4) Code Review

a) The coding style we chose is ES6 Javascript following Google's coding convention on the front and backend.

b) We used Github for our peer review and we communicated code over Discord and Slack.



6 client/src/App.js

Copy path | View file

```
@@ -6,6 +6,10 @@ import "./styling/navbar.css";
  import "./styling/errorpage.css";
  import "./styling/homepage.css";
  import "./styling/details.css";
+ import "./styling/register.css";
+ import "./styling/login.css";
+
+

  import NavBar from "./pages/components/navbar";
  import SearchBar from "./pages/components/searchbar";
@@ -17,7 +21,7 @@ import PostDetails from "./pages/PostDetails";
  import SearchResults from "./pages/components/search-results";
  import SellForm from "./pages/components/sell-form";
  import Login from "./pages/components/login";
- import Register from "./pages/components/register"
+ import Register from "./pages/components/register";

  class App extends Component {
    render() {
```

5) Self-check on best practices for security :

1. Buyer account information
2. Seller account data
3. Server security
4. Database protection
5. Secure internet protocols

Best Practices:

Password encryption in database is done by hashing with bcrypt. Users are able to register an account or login if they have existing account, which information gets stored in the database.

For registration, combination of frontend html code and passport is being used to validate the data. User emails are being checked by using Javascript to ensure it matched the correct email format. For password, user needs to confirm the password so that there is no error.

Search bar input is also being validated using frontend html codes. The search bar input is also being checked by % like algorithm which compares whether the input information matches the database or not.

6) Self-check: Adherence to original Non-functional specs

High Level specifications:

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).**-On Track**
2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of all major browsers: Mozilla, Safari, Chrome. **-On Track**
3. Selected application functions must render well on mobile devices **-On Track**
4. Data shall be stored in the team's chosen database technology on the team's deployment server.**-On Track**
5. No more than 50 concurrent users shall be accessing the application at any time-**On Track**
6. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.**-On track**
7. The language used shall be English.**-Done**
8. Application shall be very easy to use and intuitive.**-On track**
9. Google analytics shall be added **-On Track**
10. No email clients shall be allowed-**Done**
11. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated.**-Done**
12. Site security: basic best practices shall be applied (as covered in the class)**-On Track**
13. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development**-On Track**
14. The website shall prominently display the following exact text on all pages *"SFSU-Fulda Software Engineering Project CSC 648-848, Fall 2018. For Demonstration Only"* at the top of the WWW page. (Important so as to not confuse this with a real application).**-On Track**