SW Engineering CSC648/848 Fall 2018

Gator Saver

Team 9

Jake Carter
Gordon Su
Alvin Lee
Wagner Ayllon
Tina Nguyen
Martin Lee
Gary Deng

Team lead email: jcarter4@mail.sfsu.edu

Milestone 4

12-12-18

Initial Submission: 12-2-18
Revision: 12-12-18

1) Product Summary

Our product, the Gator Saver, will provide a platform for San Francisco State University students to buy and sell to each other. This will allow students to save money when buying textbooks or other school related materials. Additionally, students will be able to recuperate the costs of old school materials by selling them to other students. When released, our product will have the following features:

- Login/Registration:
  - Users shall be able to create accounts and log into them, and remain logged in during the duration of their session
  - Stored passwords will be hashed, and registration page will have a captcha
- Search with Results:
  - Users shall be able to search by text or by category, and be brought to a results page
- Post details:
  - Each post shall have a details page with a title, image, price, and details about item posted
- Messaging:
  - Users shall be able to send messages to other users about posts and respond to them
- Seller Dashboard:
  - Users shall have a dashboard to see the status of their posted items
- Admin:
  - Administrators shall be able to approve/deny posts before they are viewable on the website

Link to website: http://35.235.67.248

2) Search Function Usability Test Plan

Test Objectives:

1. The Unregistered user shall go to our website and search for any item.
2. They shall try searching nothing.
3. They shall try searching using “All categories”.
4. They shall try searching an item in “All categories”
5. They shall try searching nothing in another category.
6. They shall try searching an item in another category.
7. They shall try to filter their search by price or date.
8. They shall try to advance to the next page in their search results.

Test Plan:

Website URL: http://35.235.67.248/

Intended user: Registered and Unregistered users (The SFSU students who wants to buy)

- Test if the search function is usable.
  - Does the search work?
  - Do the search results work with any input?
  - Does the search result show anything with no input?
  - Is it possible to sort the results by category, price, date, or by alphabetical order?

- Test if the search function is effective.
  - Is it easy to locate the search bar?
  - How many clicks did it take to find the result you were searching for?
  - How many screens did you see to get to the result?

- Test if the search function is efficient.
  - Is it easy to locate the search bar?
  - Were you able to use the search function on any web page?
  - Is it easy to search another item after finding the previous item?
  - Does it have a long wait time after clicking search?

Questionnaire:

1. The search page responded fast.
    a. Strongly Agree
    b. Agree
    c. Neither agree or disagree
    d. Disagree
    e. Strongly Disagree

2. It was easy to use Advanced Search
    a. Strongly Agree
    b. Agree
    c. Neither agree or disagree
    d. Disagree
    e. Strongly Disagree

3. I found the following aspects of the search UI easy to use (Fill in the blank)

    ____

4. It was easy to find what I was looking for
    a. Strongly Agree
    b. Agree
    c. Neither agree or disagree
    d. Disagree
    e. Strongly Disagree

5. It was easy to browse the search results.
    a. Strongly Agree
    b. Agree
    c. Neither agree or disagree
    d. Disagree
    e. Strongly Disagree

6. I found the search function easy to use overall
    a. Strongly disagree
    b. Disagree
    c. Neither agree or disagree
    d. Agree
    e. Strongly agree

3) Search Function QA Test Plan

Test Objectives: Test the search function of the website using specific test sequences to make sure there are no serious bugs. Unit and smoke tests are done to see how many bugs there are needed to be fixed in the program. Testers are told specific instructions what to do so that they know if the end result passes or fails.

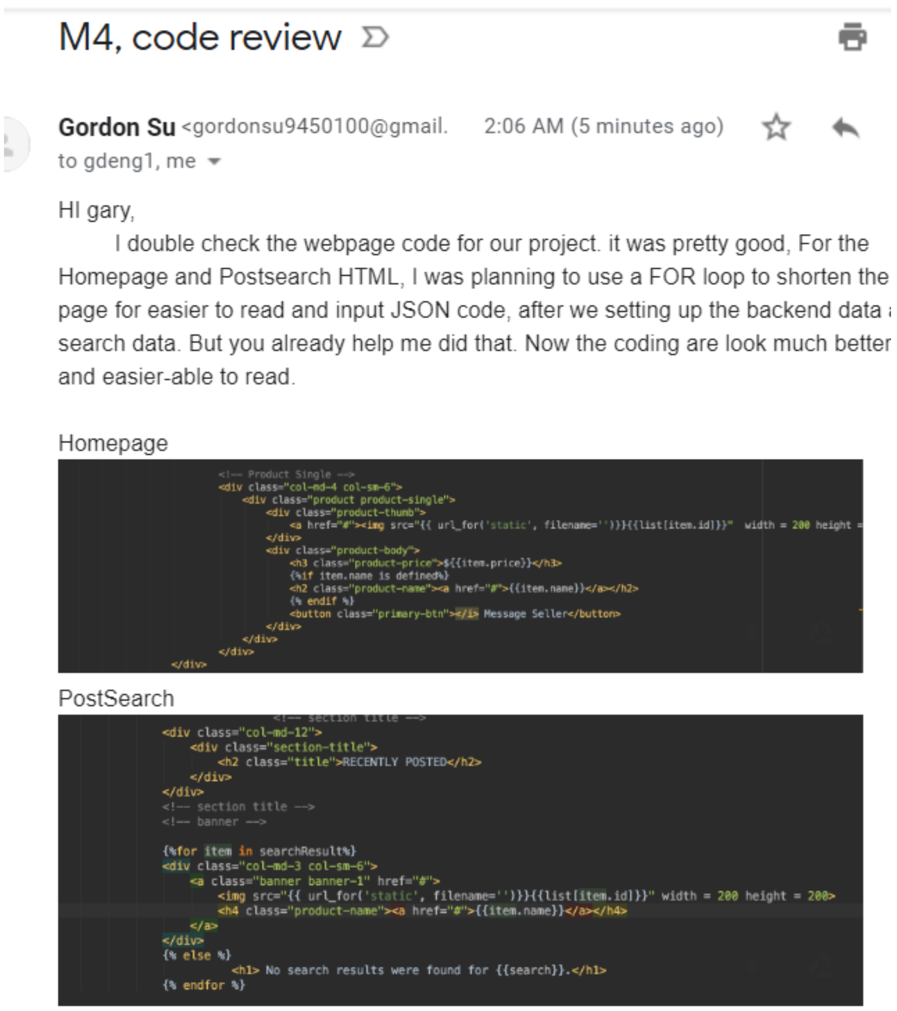
HW and SW setup: Operating system version and database used to run the website: Ubuntu 14.0 and sqlite. Hardware for testing: a mobile device and desktop/laptop. The website will support the latest versions of Google Chrome and Mozilla Firefox, as well as mobile. To setup, go to either one of the browsers, and go to http://35.235.67.248/ . This will bring you to the homepage, where you can use the search bar for testing.

Feature to be tested: Search function

Test Plan:
Use the following chart to test the search function using the latest two versions of both Google Chrome and Mozilla Firefox, and then use the same test plan for mobile devices.

Number	Description	Test Input	Expected Output	Pass/Fail
1	Test % like in search for name field	Enter "book" into the search bar input field	Gets 5 results, all of them have "book" in the name field	Pass
2	Make sure search results always return something	No input, press search button	Most recent items should be returned; A calculator, then laptop, then 5 books	Pass
3	Make sure input is a valid input	Enter "123???>[}/" into the search bar input field	User will be unable to submit the search, "Please match the requested format" text should be displayed	Pass

4) Code Review



M4, code review

Gordon Su <gordonsu9450100@gmail.   2:06 AM (5 minutes ago)
to gdeng1, me

HI gary,
I double check the webpage code for our project. it was pretty good, For the Homepage and Postsearch HTML, I was planning to use a FOR loop to shorten the page for easier to read and input JSON code, after we setting up the backend data a search data. But you already help me did that. Now the coding are look much better and easier-able to read.

Homepage

```
<!-- Product Single -->
<div class="col-md-4 col-sm-6">
    <div class="product product-single">
        <div class="product-thumb">
            <a href="#"><img src="{{ url_for('static', filename='')}}{{list[item.id]}}"  width = 200 height =
        </div>
        <div class="product-body">
            <h3 class="product-price">${{item.price}}</h3>
            {%if item.name is defined%}
            <h2 class="product-name"><a href="#">{{item.name}}</a></h2>
            {% endif %}
            <button class="primary-btn"></i> Message Seller</button>
        </div>
    </div>
</div>
</div>
```

PostSearch

```
        <!-- section title -->
<div class="col-md-12">
    <div class="section-title">
        <h2 class="title">RECENTLY POSTED</h2>
    </div>
</div>
<!-- section title -->
<!-- banner -->

{%for item in searchResult%}
<div class="col-md-3 col-sm-6">
    <a class="banner banner-1" href="#">
        <img src="{{ url_for('static', filename='')}}{{list[item.id]}}" width = 200 height = 200>
        <h4 class="product-name"><a href="#">{{item.name}}</a></h4>
    </a>
</div>
{% else %}
          <h1> No search results were found for {{search}}.</h1>
{% endfor %}
```

5) Best practices for security

Assets being protected:

- Username and password – Since many people use the same username and password for many websites, it is vital that the password is protected so it can't be accessed. Currently, the password is hashed before it is stored in the database and can not be reversed into the password.
- Posted items and images – Because user created posts contain images posted by the user, the images are protected in our database and can not be edited. Additionally, users can only see approved posts, so that any posts containing explicit content will not be approved and not shown to users. Finally, users will only be able to access their own posts on the user dashboard, and not posts created by other users.

Input Validation:

- Both the search function and the title input field for creating posts require users to enter only alphanumeric characters; the fields require a pattern and users will not be able to submit their forms if they input non-alphanumeric characters.

6) Adherence to original Non-functional specs

1. Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0
    - DONE
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.
    - IN PROGRESS
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. MySql 14.14 shall be the database that is used for storage. Apache 2.4.7 shall be the web server, hosted on Google Compute Engine 1vCPU 3.75 memory.
    - DONE – Switched from MySql to sqlite
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.
    - IN PROGRESS
7. The language used shall be English.
    - DONE
8. Application shall be very easy to use and intuitive.
    - IN PROGRESS
9. Google analytics shall be added.
    - DONE
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.

    - DONE

12. Site security: basic best practices shall be applied (as covered in the class).

    - IN PROGRESS

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-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).

    - DONE