SW Engineering CSC648/848 Section 2 Spring 2017

SFSU Marketplace

Team 12: Butterfree (Local)

Nicholas Hoffman: nicholas.camden.hoffman@gmail.com (SFSU)
Steven Soult (SFSU)
Vivian Lee (SFSU)
AJ Culanay (SFSU)
Matthew Serna (SFSU)
Jonas Vinter-Jensen (SFSU)

Milestone 4
May 1, 2017

Revision History
V1 5/1/17

1. Product Summary

Our product is named “SFSU Marketplace”. We are online, buy/sell website designed specifically for San Francisco State University students. We offer a quick, easy, and convenient way for students to sell their old textbooks, furniture, appliances, etc. to other students.

Anyone who visits this site is able to browse through all of our listings online, posted by other SFSU students. These listings can be accessed through an easy-to-use search bar. Additionally, you are able to narrow searches by categories (such as furniture, books, or appliances) or item condition (new, like new, used).

After registering an account with us by providing an SFSU email address, users are then able to message other users through our website to connect with different buyers and sellers. Also, a registered user is able to post their listings to our website, making it visible to anyone. Listings can contain a picture of the product you are selling, a quick description, title, relevant category, and a price you determine. SFSU Marketplace does not take any cut from your sales and charges nothing to host your listings. In addition to that, SFSU Marketplace makes easy suggestions of on-campus locations to meet between users and exchange goods.

On the backend, administrative personnel are able to assist users who have forgotten their passwords and can help them in resetting their password. On top of that, if it is found that a particular user is posting content that is deemed unsafe or exhibiting misconduct on the site can be banned by an admin.

Our website can be accessed by visiting sfsuse.com/~sp17g12 online.

2. Usability Test Plan

The test objective is to determine the efficiency and ease-of-use of the search feature. Are users able to easily understand the search function with all of it's parameters? Are items relevant to the search returned to the user?

Test Plan:

The user will get started by opening either Safari or Chrome and visiting the home page of our site sfsuse.com/~sp17g12. The goal is to have the user successfully complete a search on the website with adequate results returned. Users intended for this site are college students between the age of 18 and 25. The test is considered completed when the user is satisfied with their search results and had little to no difficulty operating the search function.

Test	Completed? Y/N	Errors	Comments
Search (Safari)	Yes	None	None
Search (Chrome)	Yes	None	None

1. 'Search' was easy to use. (circle one)

1	2	3	4	**5**
Totally Disagree	Disagree	Neutral	Agree	**Totally Agree**

2. I am satisfied with the search results. (circle one)

1	2	3	**4**	5
Totally Disagree	Disagree	Neutral	**Agree**	Totally Agree

3. I understand the individual search parameters in relation to the search function.

1	2	3	4	**5**
Totally Disagree	Disagree	Neutral	Agree	**Totally Agree**

3. QA Test Plan

Test Objectives: This test is designed to test the correctness and validity of the search feature. ‘Search’ should return accurate and relevant results related to the search input and access the correct sections of the database.

Hardware & Software Setup: Test is to be done from the user's laptop running either Windows or Mac from either Chrome or Safari internet browser.

Feature Tested: Search

Test Cases: Chrome

	Test Title	Test Description	Test Input	Expected Correct Output	Test Results (Pass/Fail)
1	Blank Search	Returns all items (17) from database with no filtering	Category: “All” Search: “”	Chair, Laptop, PS4...ASD	Pass
2	Category Search	Returns all items (5) from a specific category (electronics) while filtering out the rest	Category: “Electronics” Search: “”	Laptop, PS4, TV, Portable Game Device, Printer	Pass
3	Free Text Search	Returns items (1) relevant to search input across all categories	Category: “All” Search: “chair”	Chair	Pass
4	Category & Free Text Search	Returns items relevant to search input (2) within specified category	Category: “Books” Search: “textbook”	Textbook CSC600, Textbook CSC648	Pass

Test Cases: Safari

	Test Title	Test Description	Test Input	Expected Correct Output	Test Results (Pass/Fail)
1	Blank Search	Returns all items (20) from database with no filtering	Category: “All” Search: “”	Chair, Laptop, PS4...ASD	Pass
2	Category Search	Returns all items (5) from a specific category (electronics) while filtering out the rest	Category: “Electronics” Search: “”	Laptop, PS4, TV, Portable Game Device, Printer	Pass
3	Free Text Search	Returns items (1) relevant to search input across all categories	Category: “All” Search: “chair”	Chair	Pass
4	Category & Free Text Search	Returns items relevant to search input (2) within specified category	Category: “Books” Search: “textbook”	Textbook CSC600, Textbook CSC648	Pass

4. Code Review

a) Code Style

Group agreed style of conventions in regards to variable and function names, as well as indentation and placement of brackets.

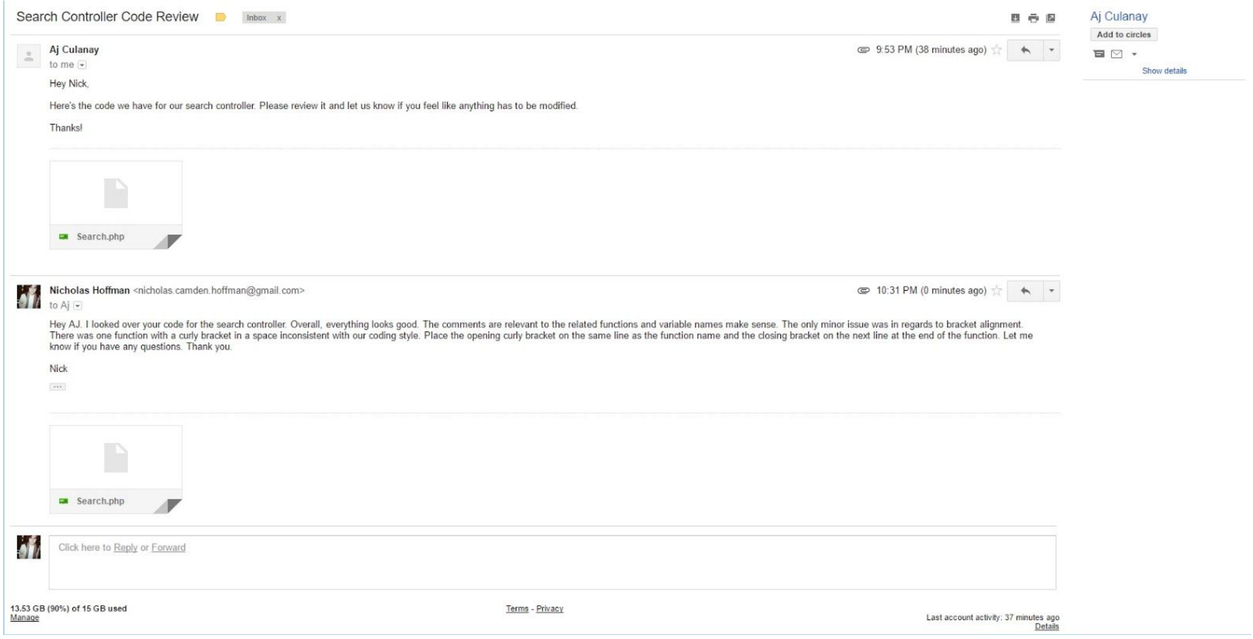
- Code is clean, well formatted (appropriate whitespace and indentation)
- Classes, methods, and variables are meaningfully named and are in camelcase
- Classes serve a single purpose, code attempts to reduce coupling
- Methods are small and serve a single purpose
- Code is well organized into a meaningful file structure, making use of the MVC design pattern and Codeigniter file structure

b) Example of Code

```php
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class Search extends CI_Controller{
    public function __construct()
    {
        parent::__construct();
        /** loads 'search_model' so we can call get_items()
         *  and form helper
         *  Url helper is needed for bootstrap. */
        $this->load->model('search_model');
        $this->load->helper('form');
        $this->load->helper('url');
    }
    /** For the functions below, the real page is located in the search_view
file. The header and footer are
     *  placed on each side, sandwiching the content in the middle. header.php
will load bootstrap, and takes the
     *  title of the page as an argument. footer.php includes the legal stuff
at the bottom of the page. */
    public function index(){
        //loads seach_view.php
        $title = array(
            'title' => 'Vertical Prototype');
        $this->load->view('header', $title);
        $this->load->view('search_view');
        $this->load->view('footer');
    }
    public function execute_search()
    {
        // receives the input from the form
        $keyword = $this->input->post('searchquery');
        $category = $this->input->post('Category');
        /* passes the search keyword to get_items()
        and stores the data in an array named 'results' */
        $data['category'] = $category;
        $data['results'] = $this->search_model->get_items($keyword, $category);
        // loads the search_view page, passing it data from get_items()
        $title = array(
            'title' => 'Vertical Prototype');
        $this->load->view('header', $title);
        $this->load->view('listing_view', $data);
        $this->load->view('footer');
    }

    public function load_description(){
        //loads the view for the item description
        $title = array(
            'title' => 'Description');
        $this->load->view('header', $title);
        $this->load->view('listingPage_view');
        $this->load->view('footer');
    }
    public function category_books() {
        $data['results'] = $this->search_model->get_items('', 'books');
        $this->load->view('header');
        $this->load->view('listing_view', $data);
        $this->load->view('footer');
    }
    public function category_furniture() {
        $data['results'] = $this->search_model->get_items('', 'furniture');
        $this->load->view('header');
        $this->load->view('listing_view', $data);
        $this->load->view('footer');
    }
    public function category_electronics() {
        $data['results'] = $this->search_model->get_items('', 'electronics');
        $this->load->view('header');
        $this->load->view('listing_view', $data);
        $this->load->view('footer');
    }
}
```

Peer Review Between AJ Culunay and Nick Hoffman.



Search Controller Code Review

Aj Culanay
to me

9:53 PM (38 minutes ago)

Hey Nick,

Here's the code we have for our search controller. Please review it and let us know if you feel like anything has to be modified.

Thanks!

Nicholas Hoffman <nicholas.camden.hoffman@gmail.com>
to Aj

10:31 PM (0 minutes ago)

Hey AJ. I looked over your code for the search controller. Overall, everything looks good. The comments are relevant to the related functions and variable names make sense. The only minor issue was in regards to bracket alignment. There was one function with a curly bracket in a space inconsistent with our coding style. Place the opening curly bracket on the same line as the function name and the closing bracket on the next line at the end of the function. Let me know if you have any questions. Thank you.

Nick

Search.php
Click here to Reply or Forward
13.53 GB (90%) of 15 GB used
Manage
Terms - Privacy
Last account activity: 37 minutes ago
Details
Aj Culanay
Add to circles
Show details

5. Self Check

Assets being protected are user information such as usernames and passwords. We are encrypting passwords by hashing them into our database. Upon registration, data input is being validated and checked. Emails are checked to be either “@mail.sfsu.edu” or “@sfsu.edu” to ensure users are students or staff at San Francisco State University.

6. Non-functional Specs

1. Application shall be developed using class provided LAMP stack

DONE

2. Application shall be developed using pre-approved set of SW development and collaborative tools provided in the class. Any other tools or frameworks must be be explicitly approved by Anthony Souza on a case by case basis.

DONE

3. Application shall be hosted and deployed on Amazon Web Services as specified in the class

DONE

4. Application shall be optimized for standard desktop/laptop browsers, and must render correctly on the two latest versions of all major browsers: Mozilla, Safari, Chrome.

ON TRACK

5. Application shall have responsive UI code so it can be adequately rendered on mobile devices but no mobile native app is to be developed

ON TRACK

6. Data shall be stored in the MySQL database on the class server in the team's account

DONE

7. Application shall be served from the team's account

DONE

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

ON TRACK

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

DONE

10. The language used shall be English.

DONE

11. Application shall be very easy to use and intuitive. No prior training shall be required to use the website.

DONE

12. Google analytics shall be added

ON TRACK

13. Messaging between users shall be done only by class approved methods to avoid issues of security with e-mail services.

DONE

14. Pay functionality (how to pay for goods and services) shall not be implemented.

DONE

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

DONE

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

DONE

17. The website shall prominently display the following text on all pages "SFSU Software Engineering Project, Spring 2017. For Demonstration Only". (Important so as to not confuse this with a real application).

DONE