SAN FRANCISCO STATE UNIVERSITY

SW ENGINEERING CSC648/848 SECTION 02 SPRING 2017

GATORLIST – TEAM 11

HASAN NIFTIYEV
hasan.niftiyev@yahoo.com

JORDAN SCANDLYN
DARRYL ROBBIN
ANDRE SIMPELO
ANTHONY SEARLES
HIN VONG

MILESTONE 4
May 5, 2017

INITIAL DRAFT

1. Product Summary

Product Name: GATORLIST

Priority 1 functions:

1. Login
2. Browse through the website
3. Sort items by:
   NAME/PRICE/DESCRIPTION
4. Search items using a percent like algorithm that guarantees a return of any relevant product
5. Post sellable items
6. Contact Sellers if buyer is interested in an item
7. Upload an image as a visual representation of their product
8. Display posting date
9. Search Input stays persistent
10. Search form input validation i.e.:
    Error Message for invalid user input
    Validate user Input
11. Remove an item

Link to GATORLIST

Sfsuse.com/~sp17g11/GatorList

2. Usability test plan

The objective of this test is to determine the usability of the products search feature for finding specific items in the website database.

Purpose:

The purpose of this testing is figuring how customers view the current search system and improving upon this search in future iterations

Problem Statement:

To figure out how to best organize the search features on the site for the average user

Test Plan and objectives:

Subjects will be tasked with the three following objectives

1. Find the description of a specific book on the site "Soon I will be invincible"
2. Search all 'electronics'
3. Find the most expensive Book for sale

User profile:

The subjects will be San Francisco State University students who volunteer for the test.

Method and test Design:

Data will be collected via an observer watching the user as well as the survey collected after the test

Test environment and equipment:

The testing environment will be a single test subject per observer using one computer preloaded with the site.

Evaluation measures and data to be collected:

User Satisfaction judged by survey
Efficiency judged by time taken to complete objectives

Legal issues:

Tests will be voluntary, anonymous, and can be abandoned at any time
Report:

Final report will contain:

Subject test number
The objectives the subject was able to complete
Time taken to complete the objectives
User Survey

Survey:

The search bar was easy to find (please check one)

- ☐ Strongly disagree
- ☐ disagree
- ☐ neither agree nor disagree
- ☐ agree
- ☐ strongly agree

I felt that the search function was quick (please check one)

- ☐ Strongly disagree
- ☐ disagree
- ☐ neither agree nor disagree
- ☐ agree
- ☐ strongly agree

The search bar produced desired results (please check one)

- ☐ Strongly disagree
- ☐ disagree
- ☐ neither agree nor disagree
- ☐ agree
- ☐ strongly agree

3. QA Test Plan

Objectives:

The purpose of this test is to find any bugs and to make sure our product is working up to standards. This test focuses on the search function of our product. We want to make sure the search works correctly by showing the correct items that the user searches for.

Hardware Setup:

Testing will be done on a laptop running Ubuntu 16.04LTS.

Software Setup:

Testing will be done of the site using multiple browsers running the latest versions of Firefox and Chrome. The actual product is hosted on AWS server running a LAMP stack. Our current state of the product is in the early beta stage.

Feature to be Tested:

Our site utilizes a search function to show similar products that the user searches for. We will test if the product actually shows up and if all the categories are working correctly when chosen. The expected time of the test is about 15-30 minutes of running the different test cases.

Risks:

One risk is that the connection to our site might be interrupted during testing. A contingency we have in place for this is to have a copy of our site hosted locally so that the user can test without interruptions.

Test Cases:

1.) The user shall be able to search for a 'math book' and have it displayed in the results of the chosen category. The user is already logged in and on the homepage of the site. After the user searches, the user shall logout and search again for the same product. This is to determine whether the search function works consistently whether the user is logged in or out.

1. User navigates to the search bar.
2. User types in 'math book' and selects ALL as a category
3. Site displays the expected output
4. User logs out of site and navigates to the home page
5. User repeats step 2
6. Site displays the expected output

2.) The user searches for glasses on multiple pages. This is to check if search is working on all pages of our site

1. User navigates to the home page and searches for glasses in 'ALL' categories
2. User then searches on the results page for the same product

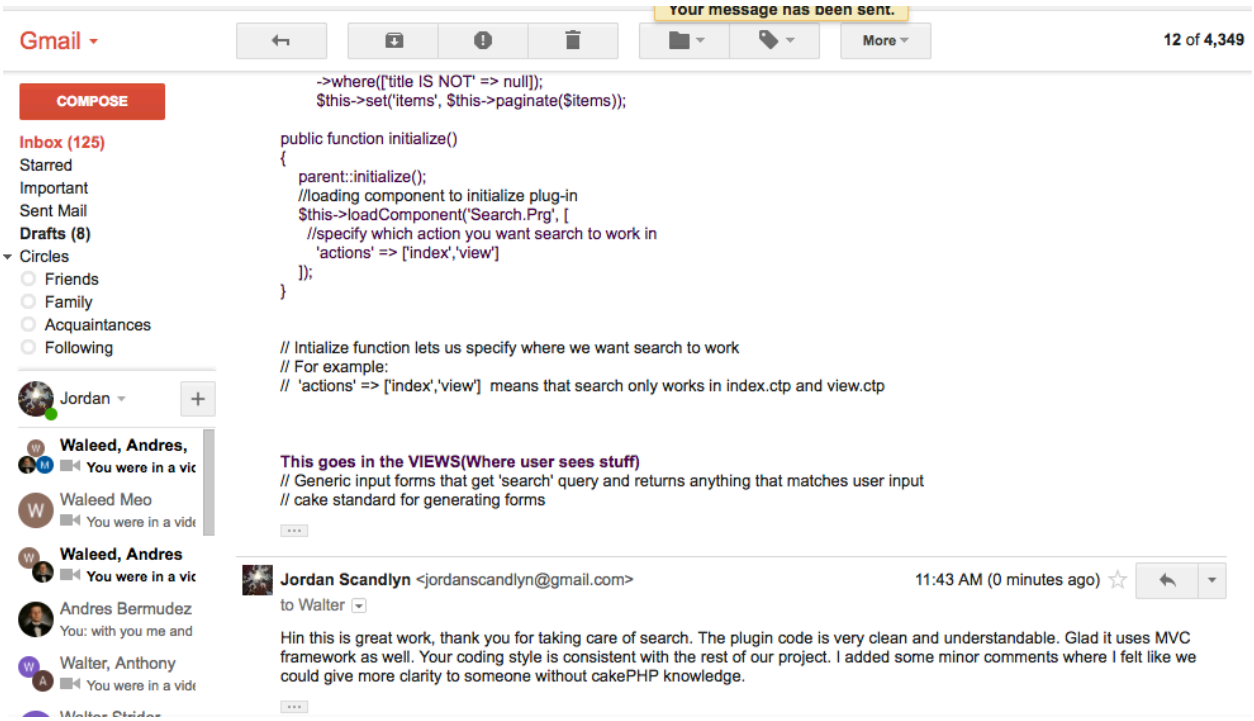
3.) This final test case is to check corner cases of our database. In our database with we have a 'math book', a 'math hat' and a 'math cup'. The expected results should be that the user is only shown the product of a specific category.

1. User navigates to the home page
2. User searches for 'math' in the books category. Expected Output: Only 'math book' is shown
3. User searches for 'math' in the furniture category. Expected Output: Only 'math cup' is shown
4. User searches for 'math' in the apparel category. Expected Output: Only 'math hat' is shown
5. User searches for 'math' in the 'ALL' category. Expected Output: 'math book', 'math cup', and 'math hat' is shown.

Test #	Description	Input	Expected Output	PASS/FAIL	NOTES
1	A logged in user searches for 'math book' in 'all' categories at different pages. User then logs out and searches for 'math book' in 'all' categories	'math book'	Only math books are shown Results should be consistent between a user that is logged in or out.	(PASS)	Meets "Expected Output" QA specification, but I expected the images to load. There seems to be a problem loading png format images. Searching functionality works as expected. Identical results while searching as a registered user vs guest. Tested on latest version of Firefox and Chrome (on Ubuntu 16.04LTS).
2	Logged in user searches for 'glasses' in homepage. User then searches for 'glasses' on the results page.	'glasses'	Results are consistent on each page by showing the products for 'glasses'	(FAIL)	Glasses were successfully listed after the search from the home page, as well as the results page. But directs me to a "Database Error" page while searching from the "Sell" page. Error persists in the latest versions of Firefox and Chrome (on Ubuntu 16.04LTS).
3	In our database we have a 'math book', a 'math hat' and a 'math cup'. The expected results should be that the user is only shown the product of a specific category and not any other category.	'math'	User searches for 'math' in the books category. Expected Output: Only 'math book' is shown User searches for 'math' in the furniture category. Expected Output: Only 'math cup' is shown User searches for 'math' in the apparel category. Expected Output: Only 'math hat' is shown User searches for 'math' in the 'ALL' category. Expected Output: 'math book', 'math cup', and 'math hat' is shown.	(FAIL)	Obtained expected results for searching "math" in "Books" and "Furniture" from the home page, but failed to find a category option from the dropdown list for "Apparel", which fails step (4). The "All" category works as expected. Tested on latest version of Firefox and Chrome (on Ubuntu 16.04LTS).

4. Code Review

- Comments in black written in email are from a peer review of the code.
- Any text that is purple came from original sender expecting feedback.
- Our team followed the MVC framework.
- Concatenated words for functions are camelCase.
- Variables with two words concatenated are separated by an_underscore.



- Feedback response to teammate regarding their work.

```
This goes in Items Model
// we have a category table to reference categories by their category_id
   $this->searchManager()
         ->value('category_id')
        //'search' is the name we give to the form bar
          //we use %LIKE% to search our table for any matching input from user
          // specify the fieldMode and comparison operators in our case OR, LIKE
          // we use camel case for any two concatenated words unless we are referencing database columns
          ->add('search', 'Search.Like', [
              'before' => true,
              'after' => true,
              'fieldMode' => 'OR',
              'comparison' => 'LIKE',
              'wildcardAny' => '*',
              'wildcardOne' => '?',
              'field' => ['title', 'description'] //we want search to work for user input in
                                    //title and description
          ]);

// we specify that we are using %LIKE% and that we want search title and description
```

- Example including more comments in project code.

This goes in Items Controller

```
    //limit results to 200
    $categorys = $this->Items->Categorys->find('list', ['limit' => 200]);
    $this->set(compact('item', 'categorys'));

    //paginate items even if no search yet.
     $items = $this->paginate($this->Items);

     $items = $this->Items
    //get 'search' query from MODEL
     ->find('search', ['search' => $this->request->query])

    //paginate returned items
     ->where(['title IS NOT' => null]);
     $this->set('items', $this->paginate($items));

public function initialize()
{
   parent::initialize();
   //loading component to initialize plug-in
   $this->loadComponent('Search.Prg', [
     //specify which action you want search to work in
      'actions' => ['index','view']
   ]);
}


// Intialize function lets us specify where we want search to work
// For example:
//  'actions' => ['index','view']  means that search only works in index.ctp and view.ctp
```

5. Self-check on best practices for security

- List major assets you are protecting
- Confirm that you encrypt PW in the DB
- Confirm Input data validation (list what is being checked and how) – we recommend you validate:

    a) search bar input;

    b) e-mail in registration to ensure it is sfsu e-mail by checking the string to contain “sfsu.edu”

Major Assets: • Username • Password • Email Address • Integrity of Website (Only logged in users can post)

Password is encrypted

Data Validation: ON TRACK

- Will check @mail.sfsu.edu
- Will check for invalid user input

6. High-level non-functional specifications (HOW the app is delivered and other constraints) that MUST be adhered to

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. **DONE**
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. **ON TRACK**
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 **ON TRACK**
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**