SW Engineering CSC648/848 Fall 2018
Gator Trader - Milestone 4
Team 14 local
Jack Cole (Team Lead) jcole2@mail.sfsu.edu, Christian Caeg, Achin Kalia
XiaoQian Huang, Juan Ledezma, Anthony Carrasco, Ryan Jin
Oct 18, 2018

Edit History

Nov 27, 2018 - Document created
Nov 1, 2018 - Added and filled all current sections

Product summary

Name: Gator Trader
Functions:

- Search, Details, and Sort
- Messaging
- Login and Registration
- User Dashboard
- Create post
- Working homepage
- One Image per Post

URL: http://sfsu-list.us.to

Usability Test Plan

This Usability Test Plan will test the Create Post functionality.

Test Objectives

Tested

- Can the user create a post within 300 seconds?
- Does the user understand how to fill out the proper information to create a post?
- Is the user frustrated with the experience?
- Does the user understand that the post will take up to 24 hours to be approved
- Does the user wish to be able to upload more than one photo?
- Does the website perform well?

Not tested

- Searching for posts
- Sending messages
- Viewing their account page
- Viewing individual posts

Test Plan

Be sure to have a device that has access to an internet connection and web browser. An example of qualifying devices would be a desktop, laptop, tablet, or smartphone. Make sure that your device is connected to a stable internet connection. To ensure the best experience on our website, your device should have either Safari or Google Chrome installed. Other browsers may work but may not be fully supported. Open your browser and navigate to the url bar. At the url bar input the website "http://sfsu-list.us.to". Press enter and wait for the web page to load.

The home page of Gator Trader (http://sfsu-list.us.to). If on a desktop/laptop, the user will have the window maximized to their screen.

The intended users for this Useability Test are SFSU Students. SFSU Students are mostly tech savvy 18-25 year olds with a very diverse range of majors who read, write, and speak English.

From the front page, the user should be able to easily navigate to the create post page. Once on the create post page, the user is expected to understand how to fill out the required information for creating a post. After the user has completed filling out the required information for creating a post, the user should be able to navigate to the create post button. Before submitting the post, the user should also understand that the post will take up to 24 hours to be accepted by an admin. Navigating and creating a post should be a quick task and is expected to be completed within 300 seconds.

The URL to be tested on is http://sfsu-list.us.to/

Questionnaire

I knew exactly what the forms on the Create Post page were asking for				
○ Strongly Agree	○ Agree	○ Neither Agree nor Disagree	○ Disagree	○ Strongly Disagree

I knew exactly what the forms on the Registration page were asking for				
○ Strongly Agree	○ Agree	○ Neither Agree nor Disagree	○ Disagree	○ Strongly Disagree

I knew what happened to my Post after submitting it				
○ Strongly Agree	○ Agree	○ Neither Agree nor Disagree	○ Disagree	○ Strongly Disagree

One photo was all that was necessary for my Post				
○ Strongly Agree	○ Agree	○ Neither Agree nor Disagree	○ Disagree	○ Strongly Disagree

Every page on the site loaded quickly				
○ Strongly Agree	○ Agree	○ Neither Agree nor Disagree	○ Disagree	○ Strongly Disagree

Comments

QA test plan

Test objectives

- The API of create Post,
-

HW and SW setup

Device: Windows laptop or desktop, Apple Laptop, Apple iPhone, Android phone
Operating System: Windows 10, OSX, iOS, and Android
Web Browser: Two Latest Versions of Google Chrome and Safari

Feature to be tested

Creating a Post, while logged in, logged out, and not having an account created.

Test Plan

#	title	description	input	expected correct output	result
1	Create Post while logged in	Click Login, then fill out the email and password fields with the given input, then click login button below the password field. After logging in, click Create Post button at the top right, then fill out the forms with the given information. Upload the provided image, and click the Submit button at the bottom.	Email: test1@test.com Password: test9000 Title: Test Title 1 Price: 105.99 Category: Phones Condition: New Description: “Test Description 1“ Upload: image1.jpg	A page should load displaying the message “Your post has been received and is awaiting approval by admins. Please wait up to 24 hours for approval”	
2	Create Post then login	Click Create Post button at the top right, then fill out the forms with the given information. Upload the provided image, and click the Submit button at the bottom.	Title: Test Title 2 Price: 105.99 Category: Phones Condition: New	A page should load displaying the message “Your post has been received and is awaiting approval	

		On the login page that loads, fill out the email and password fields with the given input, then click login button below the password field.	Description: “Test Description 2“ Upload: image1.jpg Email: test1@test.com Password: test9000	by admins. Please wait up to 24 hours for approval”	
3	Create Post and then registe r	Click Create Post button at the top right, then fill out the forms with the given information. Upload the provided image, and click the Submit button at the bottom. On the login page that loads, click the Register Button. On the register page, fill out the forms with the input given, then click the register button at the bottom.	Title: Test Title 3 Price: 105.99 Category: Phones Condition: New Description: “Test Description 3“ Upload: image1.jpg Email: test2@test.com Password: test9000 First Name: Tester Last Name: Agent	A page should load displaying the message “Your post has been received and is awaiting approval by admins. Please wait up to 24 hours for approval”	

Code Review

Coding Style

All methods and functions shall contain Javadocs style comments, with a description, arguments, and authors.
Groups of methods separated by multi-line comments with the group name at the top, e.g. “API Methods” in the controller.
Use TODO comments for features that have been intentionally left out, that shall be added in the future.
Code review comments will be prepended with “REVIEW”
Github commits shall be descriptive, with focus on being over descriptive rather than under descriptive.

Sample Code

```
// REVIEW: Description not descriptive enough. Also, include your
email in the author.
/**
 * @description Creates a post
 * @author Juan Ledezma
 */
app.post('/api/post/create', async function(req,res){
    let dateTime = new Date().toISOString().slice(0, 19).replace('T',
' ')
    var newPost={
        "user_id":req.body.user_id,
        "category_id":req.body.category_id,
        "post_title":req.body.post_title,
        "post_description":req.body.post_description,
        "post_status":"pending",
        "price":req.body.price,
        "price_is_negotiable":req.body.price_is_negotiable,
        "last_revised":dateTime,
        "create_date":dateTime,
        "number_of_images":req.body.number_of_images
    }
   // REVIEW: Needs a catch method to handle errors.
    let post = await Business.createPost(newPost)
    res.json(post)
});
```

```
// Review: Put your email in the author field
/**
 * @description Create a post Page, returns createpost.njk
 * @author Ryan Jin
 */
app.get('/createpost', function(req, res){
    res.render('createpost');
})

// Review: Put your email in the author field
/**
 * @description Creates a new post, returns confirmation
 * @param newPost All details for a new post
 * @returns {Post}
 * @author Ryan Jin
 */
static async createPost(newPost){
    // TODO: validation? user exists in db
   // REVIEW: Having await called here and again in the controller is
redundant and will slow down the program. Make only the controller do
await.
    let post = await
Post.insertNewRecord(newPost).catch(function(err) {
        console.error(`Business.createPost() error: ${err}`)
    })

    return post
}
// Review: Put your email in the author field
    /**
     * @description Creates a post
     * @param callback {function} The function to be called after
results are found
     * @author Juan Ledezma
     */
    static createPost(callback){
        // REVIEW: Needs to have arguments which have post
information, and the type changed from GET to POST.
        let url = '/api/post/create'
        return $.get(url, callback)
    }
```

Self-check on best practices for security

Protected assets:

- [ ] Email
- [ ] Passwords are hashed
- [ ] Full name (First name is displayed to the public but not the last name)
- [ ] Messages
- [ ] Post history
- [ ] Input Sanitation (Sanitized with built in ExpressJS to catch symbols)
  - Search bar input
  - Log in
  - Sign Up
  - Create post

Adherence to 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 the latest 2 versions of the Safari and Google Chrome desktop browsers. **ON TRACK**
3. Application functions shall render appropriately on mobile devices. **ON TRACK**
4. Data shall be stored using MySQL. **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 shall 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. **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. **DONE**