SW Engineering CSC648/848 Fall 2018

GatorTrader

Milestone 4: Beta Launch
Team 01
December 2, 2018

Team Lead - Marcus Mertilien
Email: marcusmertilien@gmail.com

Front-End Lead - Alex Ha
Athena Javier
Michael Phan

Back-End Lead - Raul Serrano
Daniel Martinez
Albert Shevchuk

Revision Date	Notes

1. Product Summary

Tired of dealing with strangers? Use Gator Trader!

Are you a busy San Francisco State University student? Then we have an awesome new app that is going to help you purchase and sell goods online to nearby students just like you! Gator Trader was specifically designed by an outstanding crew of engineers to help SFSU students with their needs. Our application is user friendly, sophisticated, and super convenient. Only SFSU students now have the advantage to post, view, and message each other via our website making transactions simple so you can kiss that moving truck goodbye. Dealing with local students doesn't just make our product unique, but also an accomodation to you. Visit our page at

http://gatortrader.herokuapp.com/

Thanks for looking :)
1.1 Functional Commitments

1. Guest users shall be able to browse through the website.

2. Guest users shall be able to browse by categories.

3. Guest users shall be able to search on the search engine.

4. All users who wants to purchase an item shall create an account.

5. Registered users shall be able to contact any sellers.

6. All users shall be presented the Policy while they're creating their account.

7. Registered users shall include contact information.

8. Registered users shall be able to look at the lists of the items they posted for sell.

9. Registered users shall able to manage their account. Shall able to remove or edit postings.

10. Guest users shall be able to use sorted feature: Low to high prices, High to low prices, Alphabetical.

2. Usability Test Plan

2.1 Test Objectives:

- Post ad with Tagged Image File format (.tif) image using Windows 10 on the latest Internet Explorer Browser.
- Post ad with Graphics Interchange Format (.gif) image using Windows 10 on the latest Mozilla Firefox Browser.
- Post ad with Joint Photographic Experts Group (.jpg) image using Windows 10 on the latest Opera Browser.
- Post ad with Portable Network Graphics Image (.png) image using Windows 10 on the latest Google Chrome Browser.
- Post ad with Tagged Image File format (.tif) image using Mac OS High Sierra on the latest Google Chrome Browser.
- Post ad with Graphics Interchange Format (.gif) image using Mac OS High Sierra on the latest Opera Browser.
- Post ad with Joint Photographic Experts Group (.jpg) image using Mac OS High Sierra on the latest Mozilla Firefox Browser.
- Post ad with Portable Network Graphics Image (.png) image using Mac OS High Sierra the latest Internet Explorer Browser.

2.2 Test Plan

For the useability test portion we chose the Upload Image function to be thoroughly tested. The plan it to try two operating systems: Windows 10 version 1809 and Mac OS High Sierra Version 10.13.6. In conjunction 4 different popular web browsers will be used at a random order desired by the technician which are stated to be Firefox Version 70.0 Nightly, Google Chrome Version 70.0.3538.110, Opera Version 57.0.3098.76. Internet Explorer Version 11.

The starting point will be http://gatortrader.herokuapp.com/ which is the home page. The intended user is a qualified Q&A personnel who is also a student. They shall navigate to the top left corner where the gray "Sell" button is and click it. Assuming they have not created an account the user will be prompted to register. User will have to input their First Name, Last Name, SFSU

Student ID, Username, and Password. Then click on the Green Box below named "Register" and they will be redirected to the posting page. Here the user will input their Username, the Item they are selling, a Description, the Price, the Category of the item being sold, and a upload box at the very bottom that will allow the user to browse their device and select an image to go along with the post. The user will create 4 ads with a Windows Machine and 4 ads with a Macintosh. The purpose of this is to test the limits of common image file formats to see if they are compatible with the application being tested. The three most common formats widely used that's expected to pass will be Joint Photographic Experts Group (.jpg), Graphics Interchange Format (.gif) and Portable Network Graphics Image (.png). While white elephant in the room that's expected to fail will be Tagged Image File Format (.tif), this format is widely used but its not as common. The user will create the 4 different ads with the 4 different file formats then switch the operating system and go to the same home webpage but this time just navigate to the "Login | Reg" tab to the left of the "Sell" tab where the user can simply log on with their existing username and password on the left hand side of the page and continue posting. For testing purposes user will be responsible for acquiring the proper format images with the proper licenses via google search. It it solely up to the user to follow the criteria to test the individual web browsers in any order they wish to.

2.3 Questionnaire

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
The test plan wes simple to read and follow					
The application was simple to navigate					
This was a through Test Plan					
How could you improve this test plan?					

3. QA Test Plan

3.1 Test Objectives

The objective is to test thoroughly the upload function of GatorTrader in order to

ensure that upload results are fully functional and display the image on the posting

page. Detailed information referred from section 2.2.

3.2 System Setup

Processor: Intel Quad Core i7
Ram: 16GB Ram
Operating System: Mac OS X High Sierra
Browser 1: Google Chrome
Browser 2: Mozilla Firefox
Browser 3: Internet Explorer
Browser 4: Opera
URL: http://gatortrader.herokuapp.com/

3.3 Feature to be Tested

This QA test will prove the functionality of the upload function from GatorTrader's

major page: Sell. The test will account for the following expected functionality:

- The upload button shall integrate with most web browsers and operating systems
  to be functional in a way to retrieve images files stored on the device.
- At least 3 image file extensions shall be able to upload to the database.
- All images uploaded shall be shown on the ad posted listing.

3.4 Test Cases

The test cases are as followed: 4 different web browsers, 2 different operating

systems, but most importantly are the 4 individual image format files. More detailed

information seen on section 2.2 .

3.5 Test Results

The test results are as followed: All 4 browsers along with both operating systems loaded the pages and uploaded the images accordingly. All expected image format files succeeded the upload and were seen on the individual ads posted. The only format that failed was Tagged Image File format (.tif) that succeeded the upload but failed to be seen on the ad posting with the error message "Card Image Cap". This was expected and is completely normal for the operation conducted. Thus concluding these results and moving foward.

3.6 QA TABLE

	.jpg	.png		.gif	.tif	
Macintosh OS	✔		**Opera**	✔		**Windows 10**
Macintosh OS	✔		**Mozilla Firefox**	✔		**Windows 10**
Macintosh OS		✔	**Internet Explorer**		**X**	**Windows 10**
Macintosh OS		✔	**Google Chrome**		**X**	**Windows 10**

✔ = Pass on Windows 10

✔ = Pass on Macintosh OS

X = Fail on Windows 10

X = Fail on Macintosh OS
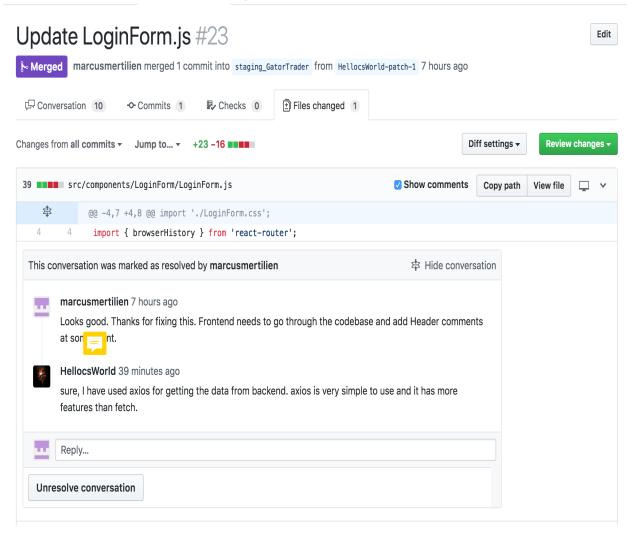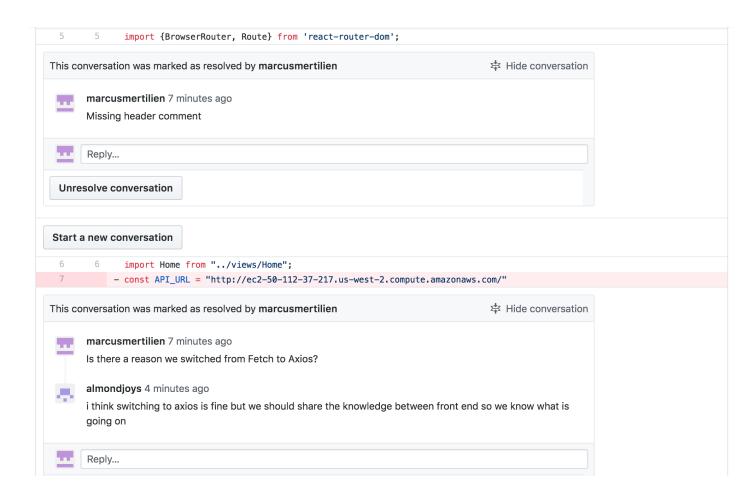
4. Code Review:

4.1 Coding Style

We opted to use several standards while developing this application. For our naming conventions, we decided to use meaningful names for our identifiers, and classes. Since our project is written completely in Javascript we are implementing the ES6 standard, enforced by ESLint.

4.2 Review

This code review is a sample of our internal code review process. The code reviewed was the back-end of the login function. Below are the conversation exchange through GitHub and code snippet relevant to the review.

4.2.1 GitHub Conversation Exchange

```
5    5    import {BrowserRouter, Route} from 'react-router-dom';
```

This conversation was marked as resolved by **marcusmertilien**   ⚙ Hide conversation

marcusmertilien 7 minutes ago

Missing header comment

Reply...

Unresolve conversation

Start a new conversation

```
6    6    import Home from "../views/Home";
     7  - const API_URL = "http://ec2-50-112-37-217.us-west-2.compute.amazonaws.com/"
```

This conversation was marked as resolved by **marcusmertilien**   ⚙ Hide conversation

marcusmertilien 7 minutes ago

Is there a reason we switched from Fetch to Axios?

almondjoys 4 minutes ago

i think switching to axios is fine but we should share the knowledge between front end so we know what is going on

Reply...

Reviewed Code Snippet

```
  7 | + const axios = require('axios')
  8 | +
8 9 |   class LoginForm extends Component {
9 10 |
10 11 |
```

```
@@ -14,7 +15,6 @@ class LoginForm extends Component {
14 15 |       this.state = {
15 16 |         username: "",
16 17 |         password: "",
17    | -       submittedPassword: ""
18 18 |     }
19 19 |   }
20 20 |
```

```
@@ -25,23 +25,24 @@ class LoginForm extends Component {
25 25 |     }
26 26 |     UpdatePassword = (password) => {
27 27 |       this.setState({
28    | -       submittedPassword: password.target.value
   28 | +       password: password.target.value
29 29 |       })
30 30 |     }
31 31 |     GetUser = (e) => {
32 32 |       e.preventDefault()
33    | -     if(this.state.username !== ""){
34    | -     fetch(API_URL + 'login/' + this.state.username)
35    | -     //take response turn into json
36    | -     .then(response => { return response.json() } )
```

```
   44 | +           console.log(valueBack)
   45 | +           if(valueBack == 'true'){
45 46 |           browserHistory.push({
46 47 |             pathname: '/',
47 48 |             state: {
```

```
@@ -52,10 +53,16 @@ class LoginForm extends Component {
52 53 |           })
53 54 |         }
54 55 |         else{
   56 | +         /**
   57 | +          * TODO - create an error like "Email or password incorrect please check your info and try again"
   58 | +          */
   59 | +
   60 | +
   61 | +
55 62 |         }
56 63 |       })
57 64 |     }
58    | -   }
   65 | +   }
59 66 |   }
60 67 |     render(){
61 68 |       return (
```

```
@@ -69,7 +76,7 @@ class LoginForm extends Component {
69 76 |             </FormGroup>
70 77 |             <FormGroup id="FormElement">
71 78 |               <Label id="Label" for="Password">Password</Label>
72    | -             <Input type="password" name="password" id="password" minlength = "4" maxlength="20" value = {this.state.su
   79 | +             <Input type="password" name="password" id="password" minlength = "4" maxlength="20" value = {this.state.pa
73 80 |             </FormGroup>
```
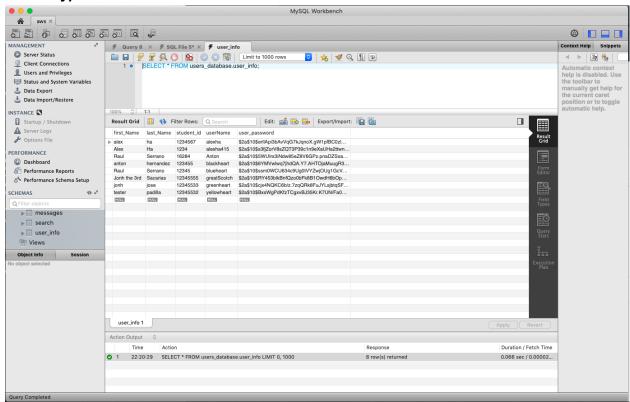
5) Self-check on best practices for security

5.1 Major Assets Being Protected
The following user information is being protected:

- Name
- Password
- Email address
- Listing address

5.2 Encryption of Passwords in the Database

5.3 Input Data Validation
The following user input is checked for validation on the front-end by built in html validation and bootstrap for registration, login and posting.

Listing details:
- Price
- Description
- Name of the item

Search query input:
- For example, if a user attempts to search "jgdfuhdkghdlfghlikhsgkhgkjehrkfghierkhekrhgerhtkjerhjerbjerjthbjerbtjerb", the field will only take up to 40 characters.

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

	Non-Functional Spec	Status
1.	Application shall be developed using React, Heroku, Express, MySQL.	DONE
2.	Application shall be designed for desktop and laptops. The browsers that support it are Chrome, Mozilla, and Safari.	DONE
3.	Application shall render well on mobile	ON TRACK
4.	Data will be stored on MySQL.	DONE
5.	No more than 50 users can access the website at a 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.	DONE
9.	Google analytics shall be added	ON TRACK
10.	No E-mail clients shall be allowed	DONE
11.	No pay functionality only messages.	DONE
11.	The website will display "SFSU-Fulda Software Engineering Project CSC 648-848, Fall 2018. For Demonstration Only" at the top of the page.	ON TRACK