BRANDON ZWIER

Unit Testing

Student ID: 1597151
University of California, Santa Cruz
CMPS115 - Software Engineering Project I

Contents

EDITARTISTPROFILE.JS

EditArtistProfile.js has been tested in an un-automated environment several ways. The container holds several interactable fields.

- Four text fields for user handles. (Instagram, Twitter, Tumblr, Facebook)
- Two Buttons (Update Profile Information, Cancel)

The four text fields are able to take in a string, and then the **Update Profile Information** button will POST them to the database. The page will then redirect to the **Artist Dashboard**.

For the following un-automated tests, input will be referenced in the order presented in the itemized section above. (Instagram, Twitter, Tumblr, Facebook)

Test 1:

Input: @BMoneyzmoney, @Nothing, @Test, https://facebook.com/brandon.zwier
Expected Output: @BMoneyzmoney, @Nothing, @Test, https://facebook.com/brandon.zwier

Upon entering the data in these fields, and hitting the **Update Profile Information** button, the page redirected me to the dashboard as expected.

Actual Output: @BMoneyzmoney, @Nothing, @Test, https://facebook.com/brandon.zwier

Test 1 was successful.

Test 2:

Paramater *null* represents a null or empty string.
Input: *null*, 123, @Test, https://facebook.com/
Expected Output: Error: Instagram field cannot be left blank.

Upon entering the data in these fields, and hitting the **Update Profile Information** button, the page threw an error stating I cannot enter a null string into the data field.

Actual Output: Error: Instagram field cannot be left blank.

Test 2 was successful.

Test 3:

This test doesn't take in user input, but relys on the function to generate blank handle URL's on new account creation. As an example, a new accounts Instagram field will not be blank, but will be given as **https://www.instagram.com/** to prompt the user to finish the URL. Therefore there is no provided input.
Expected Output: @BMoneyzmoney, @Nothing, @Test, https://www.facebook.com/brandon.zwier

Upon creating a new account, and choosing the **My Account** button to direct me to updating my profile information, the four fields were filled in with the correct handle information.

Actual Output: https://www.instagram.com/, https://www.twitter.com/, https://www.tumblr.com/, https://www.facebook.com/

Test 3 was successful.

EDITBUSINESSPROFILE.JS

EditBusinessProfile.js has been tested in an un-automated environment several ways. The container holds several interactable fields.

- Three text fields with multi-row functionality for business account information (About, Worth Knowing, Additional Notes)
- Four text fields for user handles. (Instagram, Twitter, Tumblr, Facebook)
- Three Buttons
  - Update Profile Information
  - Cancel
  - Upload Image (For uploading an account avatar)

The first three text fields are able to take in a considerably long string, and will expand to a maximum of 10 rows to accommodate the string visually for the user. The four handle text fields are able to take in a string, and then the **Update Profile Information** button will POST them to the database. The page will then redirect to the **Business Submissions Dashboard**. The page also displays a **Business Card** which is a visual representation output of the information a business enters on this page.

For the following un-automated tests, input will be referenced in the order presented in the itemized section above. (Avatar, About, Worth Knowing, Additional Notes, Instagram, Twitter, Tumblr, Facebook)

Test 1:

This test uses *Image* to denote that the **Upload Image** button has been used and an image file has been selected from the file system.
Input: *Image*, "Lorem Ipsum", "Lorem", "Ipsum", @BMoneyzmoney, @Nothing, @Test, https://facebook.com/brandon.zwier
Expected Output: *Image*, "Lorem Ipsum", "Lorem", "Ipsum", @BMoneyzmoney, @Nothing, @Test, https://facebook.com/brandon.zwier

Upon entering the data in these fields, and hitting the **Update Profile Information** button, the page redirected me to the **Business Submissions Dashboard** as expected. The information was properly displayed on the **Business Card**.

Actual Output: *Image*, "Lorem Ipsum", "Lorem", "Ipsum", @BMoneyzmoney, @Nothing, @Test, https://facebook.com/brandon.zwier

Test 1 was successful.

Test 2:

This test uses *Image* to denote that the **Upload Image** button has been used and an image file has been selected from the file system. Paramater *null* represents a null or empty string.
Input: *Image*, *null*, "Lorem", "Ipsum", "@BMoneyzmoney", "@Nothing", "@Test", "http://"
Expected Output: Error, about field cannot be left empty.

Upon entering the data in these fields, and hitting the **Update Profile Information** button, the page threw the error: Error, about field cannot be left empty.

Actual Output: Error, about field cannot be left empty.

Test 2 was successful.

SENDTOBUSINESS.JS

SendToBusiness.js has been tested in an un-automated environment several ways. The container utilizes **BusinessCardMedia.js** which is documented in the next section. **SendToBusiness.js** contains several functions with the following outputs.

- An **Axios** function which GET's all business's and their respective information from the database.
- The **Render** function will display the actual page.

The **Axios** function will GET the business information from the database and fill an array called **info[]** that is stored in the state of the class. The array is filled with Objects which contains each business's respective information. The Render function displays the actual page and maps the data from **Info[]** into the input fields of **BusinessCardMedia**. The page will then display a list of **Business Cards** with each business's information.

For the following un-automated tests, input will be provided directly into the state of the class in order to show expected output.

Test 1:

For this test, *url* represents and imgur.com url fed into the function to show an avatar. The input is provided as follows: Avatar, name, creation date, about, worth knowing, additional notes.
Input: *url*, bBrandon, November 29, 2018, Lorem Ipsum, Lorem, Ipsum.
Expected Output: *url*, bBrandon, November 29, 2018, Lorem Ipsum, Lorem, Ipsum.

Upon creating these indirect inputs and feeding them into the state, the page displated the following output.

Actual Output: *url*, bBrandon, November 29, 2018, Lorem Ipsum, Lorem, Ipsum.

Test 1 was successful.

BUSINESSCARDMEDIA.JS

BusinessCardMedia.js is a class that creates a business card taht can be displayed in various parts of the site. It takes inputs of Avatar, Name, Creation Date, About, Worth Knowing, and Additional Notes. It displays these fields in a scalable card. **BusinessCardMedia.js** has been tested in an un-automated environment several ways.

For the following un-automated tests, input will be provided directly into the state of the class in order to show expected output.

Test 1:

For this test, *url* represents and imgur.com url fed into the function to show an avatar. The input is provided as follows: Avatar, name, creation date, about, worth knowing, additional notes.
Input: *url*, bBrandon, November 29, 2018, Lorem Ipsum, Lorem, Ipsum.
Expected Output: *url*, bBrandon, November 29, 2018, Lorem Ipsum, Lorem, Ipsum.

Upon creating these indirect inputs and feeding them into the state, the page displated the following output.

Actual Output: *url*, bBrandon, November 29, 2018, Lorem Ipsum, Lorem, Ipsum.