Members: Daniel Chen, Jess Cheng, Karim Rahal, William Hu, Zoe Zhou, Zuoning Zhang

CSCI-201

Dec 4, 2021

Mentor: Vartika Bhatia

Professor: Victor Adamchik

CSCI-201 Final Project Testing Plan

Black-Box Tests:

Module: Login

Scenario	Test Case	Pre-Condition	Test Steps	Expected Results
Inputting the username	Empty username	Username box should be empty	Do not write anything in the username box then click enter	Pop-up screen or alert that says that username is invalid
	Inputting username for a user that has not made an account yet	Username should not exist already in the database	Input a username that does not exist in the database	Pop-up screen or alert that says that username is invalid
	Correct username/password combo	Username and password should be already stored in our database	Input the correct username and password into the form and press login	Should be logged in to colorly and directed to the home page
Inputting the password	Empty password	Password box should be empty	Do not write anything in the username box then click enter	Pop-up screen or alert that says that password is invalid
	Wrong password	Password should be a wrong password and not exist in	Insert wrong password for a username	Pop-up or alert that says the password is incorrect

		database		
	Correct username/password combo	Username and password should be already stored in our database	Input the correct username and password into the form and press login	Should be logged in to colorly and directed to the home page

Module: Create Account

Scenario	Test Case	Pre-Condition	Test Steps	Expected Results
Inputting the username	Invalid characters in username	Username can only contain alphanumeric values or punctuation characters that are periods, underscores, and dashes	First insert invalid characters such as [!, @, #, ^] into the text box then click the login button	Pop-up screen or alert that says that username is invalid
	Length of username	Username does not already exist in the database	Inserting a username length that is out of the parameter or inside the parameters	Pop-up screen or alert that says that username is invalid if it is over the limit or allow the username to be accepted if in parameters
	Username that exists in the database	A user with a certain username has already created an account	Insert a user's credentials that does not exist yet in the database and then enter the password and click login	Pop-up screen or alert that says that user does not exist

	Empty username	The username box must be empty and have no text	Do not write anything in the username box then click enter	Pop-up screen or alert that says username is empty
Inputting the password	Empty password	The password box must be empty	Do not write anything in the password box then click enter	Pop-up screen or alert that says password is empty
	Length of password	Password box is not empty and is a valid password	Inserting a password length that is out of the parameter or inside the parameters	Pop-up screen or alert that says that password is invalid if it is over the limit or allow the password to be accepted if in parameters
Inputting the email	Invalid email	Email is in the incorrect format	Inputting an email that is not in the correct email format [email@website.domain]	Pop-up screen or alert that says email is invalid
	Length of email	Email does not already exist in the database	Inputting an email that is in the parameters or outside of the parameters in term of length	Pop-up screen or alert that says email is invalid or accept the email
	Empty email	Email box must be empty	Do not write anything in the email box	Pop-up screen or alert that says email is empty

Module: Community

Scenario	Test Case	Pre-Condition	Test Steps	Expected Result
Verify the like/dislike function	Like an outfit	Each user can only like/dislike an outfit once a time, and they cannot cancel it. The ranking would be updated immediately.	Click the Like Button	Both Like & Dislike Buttons disappear. Show message “You liked this”
	Dislike an outfit		Click the Dislike Button	Both Like & Dislike Buttons disappear. Show message “You disliked this”
View Outfit Details	View outfit details by clicking the image	User can view a public outfit by clicking on either the outfit image or the outfit name	Click the image	Jump to a new page with detailed outfit information
	View outfit details through clicking the outfit name		Click the outfit name	

Module: Home

Scenario	Test Case	Pre-Condition	Test Steps	Expected Result
Display the returned clothes from Backend	Click the Like	Each user can add a product to his cart only once	Click the Add to Cart button, Search for a term	Display results of products returned by backend

Module: MyCart

Scenario	Test Case	Pre-Condition	Test Steps	Expected Result
Display the returned clothes from Backend	Click the Like	Each user can add a product to his cart only once	Click the Add to Cart button, Search for a term	Display results of products returned by backend

Module: Ranking

Scenario	Test Case	Pre-Condition	Test Steps	Expected Result
Ranking updated	Like an outfit	The ranking should be immediately updated after a user like/dislike an outfit	Click the Like/Dislike Button of an outfit on the Community Page and refresh the Ranking Page	Ranking should be updated based on the new calculated score for each outfit
	Dislike an outfit			

Backend White-box Tests:

Our backend runs as a REST API which mostly performs CRUD operations on the database; therefore, our white-box tests will predominantly involve the interactions between the backend API and the database.

Backend Interaction With The Database:

- Testing if the corresponding items are presented for the OutfitCard and Cart tables before integrating with the front-end.
- Check the database table for the items in the cart after an item has been added to the cart of the user to see if it is reflected in the database.
- Check the database table for the items in the cart after an item has been deleted from the cart of the user to see if it is reflected in the database.
- Check the database after an outfit is created from the cart to see if it is correctly added
- Check the database after an outfit is deleted
- Check the database after an outfit is liked or disliked
- Check the database after a white-box user signup
- Check if the cart information is saved in the database after the user has logged out from their account and logged in again, see if it is reflected in the database and the website.
- Check if the value has been added correctly to the database. Check if the double variables are reflected as double, such as the itemPrice in the Item table.
- Check if the data mapping format from the front-end to the database is correct. We can check this by verifying with the front-end crew.
- Check whether our most popular outfits GET endpoint correctly returns the most liked outfits

Authorization

- Check whether our API correctly implements authorization in the following endpoints:

    `/cart`: are we returning the correct cart for the user?; are we deleting the correct cart for the user?

    - `/cart/items/<item_id>`: are we putting the items in the correct (current) user’s cart?; are we deleting the items in the correct (current) user’s cart?
    - `outfits/<outfit_id>`: is only the owner of the outfit able to delete the outfit?
    - `outfits/<outfit_id>/toggleLike`: are we toggling a like/dislike for the correct (currently-authenticated) user?

Utility Functions:

- Check whether our palette comparison algorithm returns expected results
    - Example:
        - Input palette 1: (204,219,220);(128,206,215);(99,199,178)
        - Input palette 2: (204,219,220);(128,206,215);(0,0,0)
        - Result: distance of 47.52893855326458 (not close)