

LookLoom Project Report

Contributors

- Linh Tran (linhtran003)
- Olivia Zhu (oliviaazhu)
- Yash Thapliyal (Yash1hi)
- Brendan McCall (brendanmccall11)
- Thanh Dao (ThanhThanhDao)
- Sujay Potlapelly (supo3618)

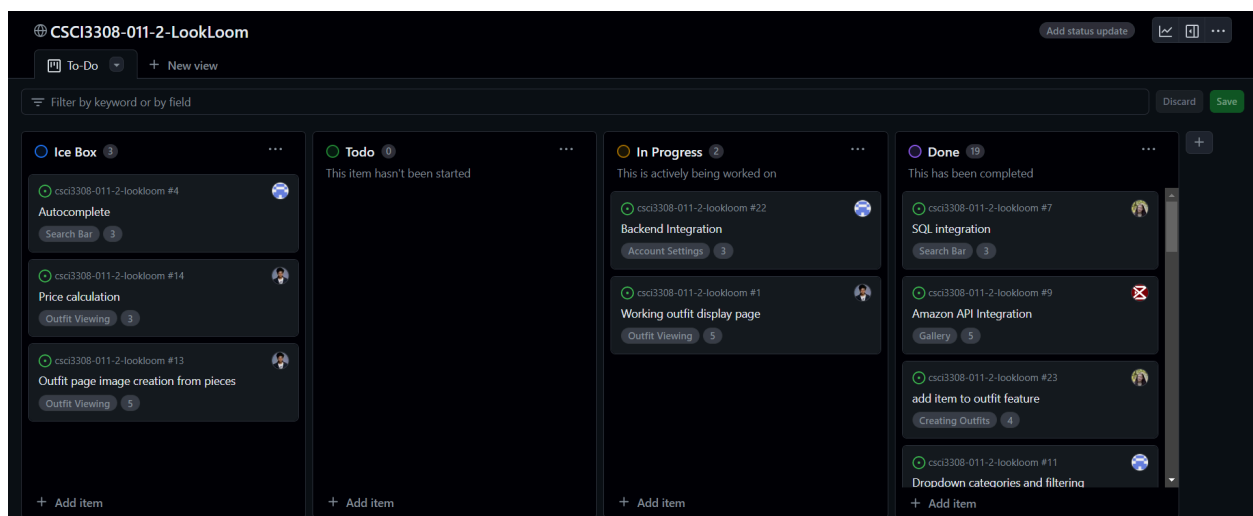
Project Description

For aspiring fashionistas, who are looking to bring their outfit ideas to life, LookLoom is a shopping and fashion website that allows users to bring together multiple vendors under a single site and mix and match clothing pieces, categorize outfits, and purchase items. LookLoom is a one stop shop for styling outfits.

The LookLoom gallery offers users a way to explore a variety of clothing pieces and search for items from all sorts of vendors. They are able to add these pieces to their closet. In their closet, users can see all the items they've added from various online shopping applications and import pictures and details of their own clothes. They are able to sort these items into different categories to make browsing easier. From there, the user can mix and match items to form outfits and save them to their closet. Users are able to view their assembled outfits as well as links to purchase each item.

Project Tracker (Github Project Board)

Link: <https://github.com/users/brendanmccall11/projects/1/views/1>



Video:

https://drive.google.com/file/d/117UUeinB5dnOA8vs0m_eOSFcjRB7t-F0/view?usp=sharing

VCS

Link: <https://github.com/brendanmccall11/csci3308-011-2-lookloom>

Contributions:

Olivia Zhu: I worked on the HTML/CSS and API routes for the login, logout, and register pages. Additionally, I developed the API and HTML/CSS for the ‘Add Your Own Item’ feature, which is a button that opens a modal that allows users to upload information about their own items. Similarly, I made a ‘Delete Item’ feature that lets users delete items from the closet and any outfits. I also worked on/fixed the API routes for the closet to only display information corresponding to the user, and wrote all of the automated test cases for the project.

Linh Tran: I worked on the HTML/CSS for all of the Handlebars partials for our website, focusing on the sidebar and navigation bar. Another main feature I worked on was the “Add to Outfit” feature, which allows a user to click on the “Add to Outfit” button on any item in their closet and a modal appears, allowing them to add the item to an existing outfit or create a new outfit. I did the HTML/CSS, API route, and SQL queries for this feature. Some other smaller tasks included creating the LookLoom logo and submitting release notes weekly.

Yash Thapliyal: I worked on HTML/CSS for the outfits page of our website as well as the backend for that route, calling on the database to query for an outfit number and display associated information using the various relations. I also set up the initial skeleton for the website, setting up the docker container, necessary node modules, and GET routes for our main pages to give the team a starting point. The last thing I did was initially set up the PostgreSQL database, initializing relations and data tables to query out of.

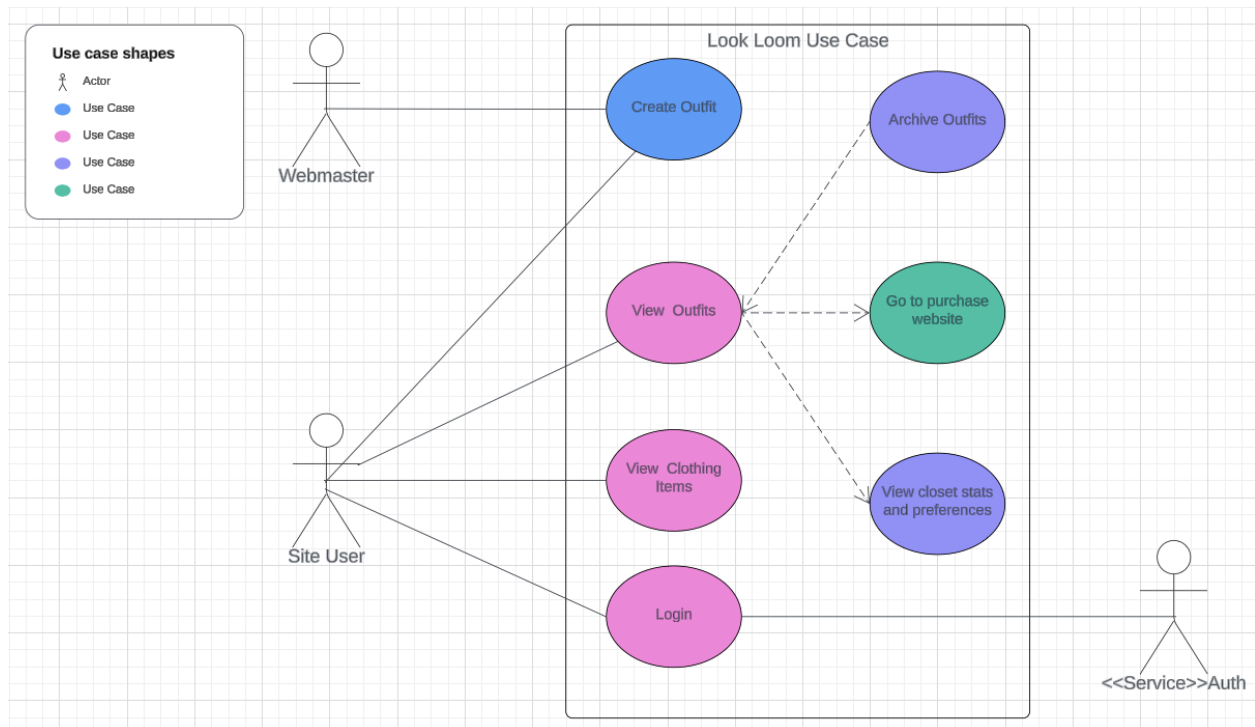
Brendan McCall: I worked on the HTML/CSS and API routes for the gallery page, allowing users to browse many different items. Additionally, I worked on the external API integration (Axesso) which pulled from Amazon to display items on the gallery page depending on what the user searched for using the search bar. I also worked on styling overall for the website, making sure that it was mostly continuous. The majority of this was for the clothing cards on the gallery and closet pages of the website.

Thanh Dao: I developed the HTML/CSS and API routes for the closet page, enabling users to explore their wardrobe and subcategories such as tops, bottoms, dresses, shoes, and accessories. I

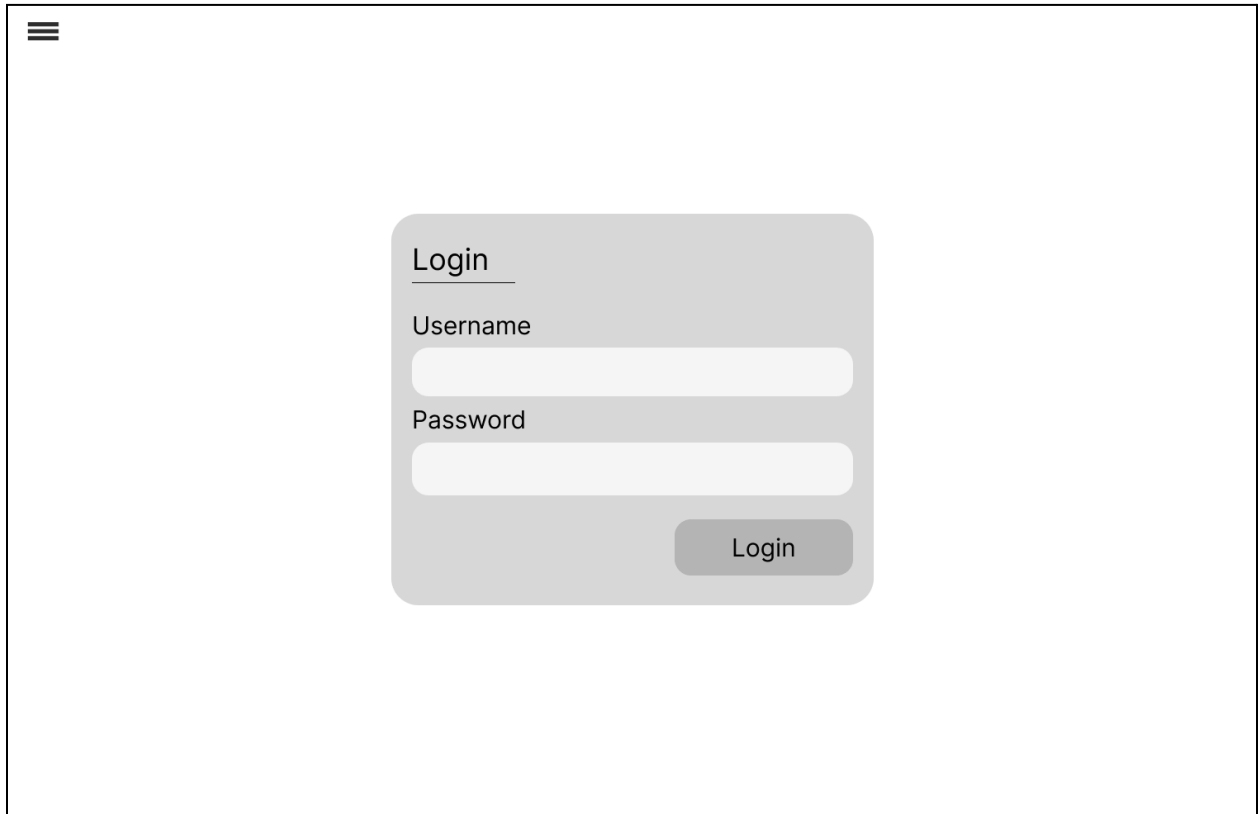
also implemented the API for the 'add to closet' feature, which allows users to include static gallery items in their closet. Additionally, I debugged the webpage to ensure full visibility and functionality of item cards and their associated buttons of the gallery and closet page.

Sujay Potlapelly: I worked on the HTML/CSS and API routes for the account details page. This included allowing users to change their passwords. The account details display the users' number of items and outfits. The user can also update their preference on the type of closet they want to have. Users can also choose between public and private closets. All of the changes made on the account details/profile page interact with the database.

Use Case Diagram



Wireframes



A wireframe of a login form. In the top-left corner of the page is a hamburger menu icon (three horizontal lines). Centered on the page is a light gray rounded rectangle containing the login form. The form has a title 'Login' with a horizontal line underneath. Below the title are two text labels, 'Username' and 'Password', each followed by a white rounded rectangular input field. At the bottom right of the form is a dark gray rounded rectangular button with the text 'Login' in white.

≡

Login

Username


Password

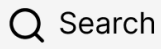
Login



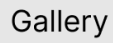
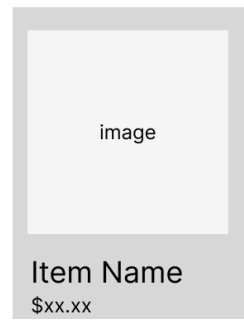
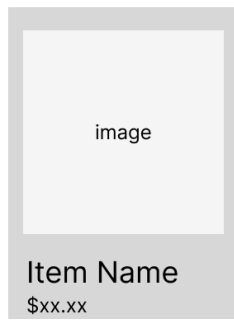
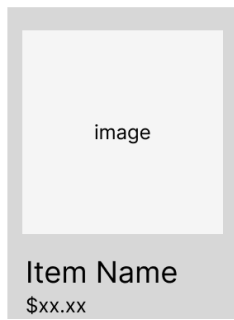
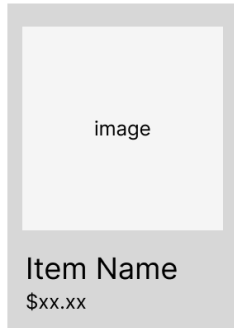
Closet

- Hats
- Jewelry
- Tops
- Pants
- Skirts
- Shoes

 Account Settings



LT



Outfits

Hats

Jewelry

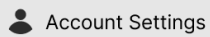
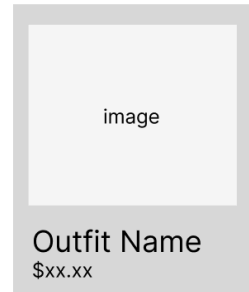
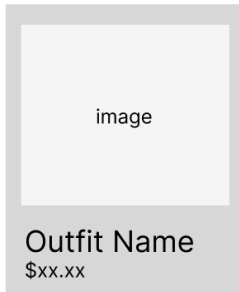
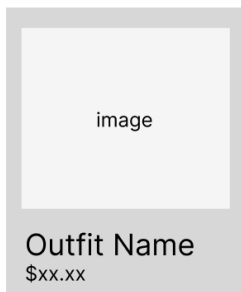
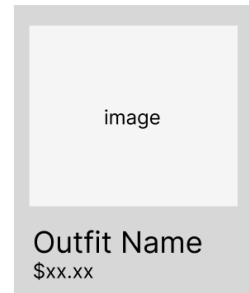
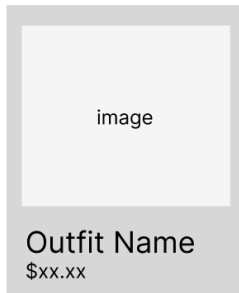
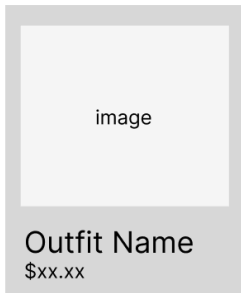
Tops

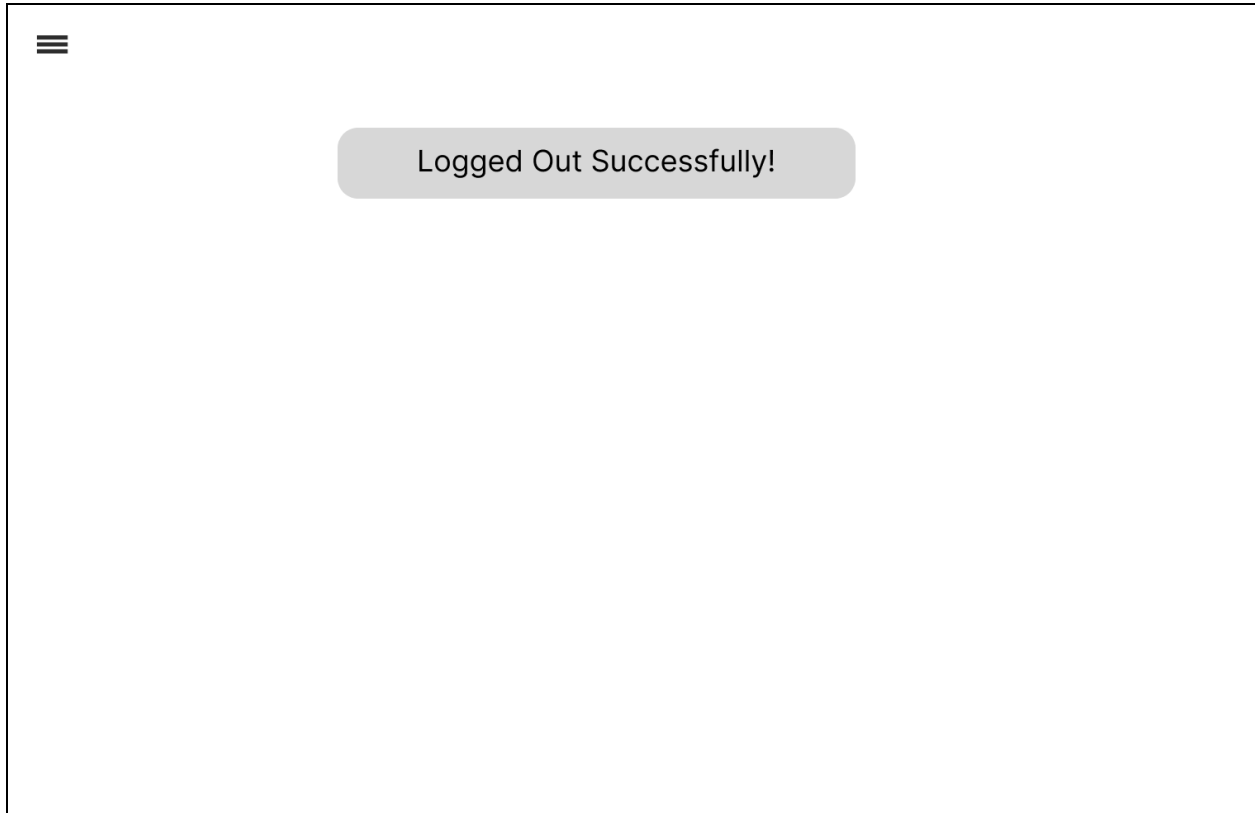
Pants

Skirts

Shoes

LT





Test Results:

1. Login

The user is logging in with the provided test data, so they can access the features of the application. They type in the provided data, and fill out the correct fields, before hitting the login button. Their behavior is consistent with the use case, as when they put in the provided login details, they are logged in and taken to the gallery page. When they change the username to an unregistered account/change the password to be wrong, they are unable to login and the error message shows. Additionally, when they click on the “Register” button on the bottom of the login interface, it takes them to the registration page if they don’t have an account. There were not any deviations from any of the expected results, so we ultimately did not need to make changes to our application.

2. Create an outfit

The user is navigating through their closet, so they can decide what clothes would make a good outfit. When they see a piece of clothing they like, they click the ‘Add to Outfit’ button, which opens a module and allows them to either add it to an existing outfit, or make a new one. In this case, the user makes a new outfit called “Casual” and they also add a random description, because there are currently no other outfits. After they add it to the outfit, the modal is closed. They continue to browse and add other items to the outfit. When they go to the outfit page, the

outfit is displayed, and they click on it to view the details. Again, there are not any user actions that deviated from what we expected. One thing to note is that our application does not pull up a whole new ‘Create Outfit’ page like our test case describes. It just pulls up a modal, but the functionality is still the same, and it still works, so we did not make any changes.

3. Searching for items

On the gallery page, the user navigates to the search bar. They type in their query. In this case, it is “hat.” The website takes a second to load before pulling up a bunch of hats that can be found on Amazon. Each hat has a description that shows its price and its rating. The user tries to click on one of them, and it takes them to the respective Amazon page. Each item also has an ‘Add to Closet’ button that they can click. There is a slight deviation from the expected actions because the API takes a while to load, so the user almost tries to reload the page. However, it loads just in time, so they stay. Ideally, we would be able to fix this, but it is more on our API’s side than our side, so for now, no changes were made.

Deployment:

Link to repository: <https://github.com/brendanmccall11/csci3308-011-2-lookloom>

If you are trying to run this locally, clone the repository and make sure you have the correct .env file. Go to the ProjectSourceCode folder and run docker compose up in the terminal. Navigate to <http://localhost:3000/> to try out the website. Otherwise, if there is a working link, please use the link: <http://recitation-11-team-2.eastus.cloudapp.azure.com:3000/>