CONGO

Team 16 - Sprint 2 Retrospective

Rohith Ravindranath, Kenny Varghese, Raman Kahlon, Spencer Kline, Ashwin Mahesh

What Went Well?

In general, we were able to extend upon the foundation we created in sprint one. In this sprint we did a much better job of planning out what functionality needed to be created first and how long it should take so we could stay on schedule and reduce the amount of down time in between each separate task. Since we had all of the clickable prototypes we were able to easily develop the frontend and backend concurrently and then hook up the two to have a working page.

User Story 1: As a congo merchant, I would like to edit products/tags

Task Number	Description	Time	Task Owner
1	User selects product to edit	2	Raman
2	User inputs the specific changes to product on UI	2.5	Raman
3	Input is updated to the database	2.5	Raman
4	Success message show on UI, that product information has been updated	2.5	Raman
5	Unit Tests - display dummy product, incorrect input, fetching from database	4	Raman

<u>**Completed:**</u>

For this story, we finished each of the tasks. The user, once logged in as the merchant, is able to view all of their uploaded products, and can edit each one by clicking on the edit field. All of the originally inputted fields are filled in with the pre-existing information, and the user can modify them as needed. The update, if successful, is reflected in the database. There is also UI to reflect the success.

User Story 2: As a congo merchant, I would like to remove products/tags

Task Number	Description	Time	Task Owner
1	Separate product and the tags associated to it	2.5	Kenny
2	Delete product from database	3	Kenny
3	Send confirmation message and email about product	3	Kenny
4	Unit Tests - Correctly remove products and make sure user can't search it	3	Kenny

Completed:

If the user is logged in as a merchant, they will be able to see all of the products which they have uploaded. Beneath each one is a remove button. We modified the functionality of this button. Instead of removing the product from the database, it simply inactivates the product, preventing customers from viewing and purchasing the product. This was done to prevent issues in the database with existing orders. All of this is fully functional. However, we did not implement functionality for sending email confirmations as we found it unnecessary.

User Story 3: As a user, I would like to get licensed for a merchant account

Task Number	Description	Time	Task Owner
1	Get user input	2	Ashwin
2	Validate user input	2.5	Ashwin
3	Encrypt password and merchant license	3	Ashwin
4	Admins approve of merchant registration on admin portal	2.5	Ashwin
5	Add to database	2	Ashwin

Completed:

All of the functionality for registering as a merchant is complete. If a user is logged in as a regular customer, they will be able to access the merchant registration page. This page takes in all of the required input, and validates it to make sure it matches our needs. Any errors are reflected in the database. If all input is valid, it is stored in the database. The password is encrypted. The merchants are given a license number, which we generate, to login as the merchant. However, we did not implement any site admin functionality to approve/reject merchants since it will be implemented in the admin page, so as of now, all merchants are automatically approved.

User Story 4: As a user, I would like to login into my merchant account

Task Number	Description	Time	Task Owner
1	Validate login	3	Raman
2	Store merchant login in session	3	Raman
3	Unit tests - incorrect password, attempt limits	3	Raman

Completed:

All of this functionality is complete. Merchants can login with their license number and a password. The password is encrypted the same way as with registration and is compared to the one stored in the database. If they match, and the licenses match, then the merchant who logged in is stored in local storage, to use for validation of other pages.

User Story 5: As a user, I would like to add products to my cart

Task Number	Description	Time	Task Owner
1	Product is added to the user's cart within database	3	Rohith
2	Display number of products remaining and subtract by one once added to cart	2.5	Rohith
3	Products in cart appear in Cart Page by calling the database	2.5	Rohith
4	Unit Test - Test to see if products are correctly in the database and extracting it	3	Rohith

Completed:

When the user clicks add to cart, the product is added to a Cart object allocated to that specific user, along with other necessary information about that specific item, like color, size, and quantity. On the cart page, we are fetching that cart successfully and all of the items it contains, by using the user's id, which is stored in local storage.

User Story 6: As a user, I would like to checkout products in my cart.

Task Number	Description	Time	Task Owner
1	Getting user shipping information from form	3	Spencer
2	Validating Information	3	Spencer
3	Unit Test - provide valid and invalid information, assert checkout when correct	4	Spencer

Completed:

When a user checks out, we first get all of the shipping information from the user. Once these fields have been validated, we process the creation of that order. A new order item is created in the database with all of the purchased items, and is allocated to that user, to allow for later fetching. The cart is then cleared of all items. All of this is fully functional.

User Story 8: As a user, I would like to browse via categories.

Task Number	Description	Time	Task Owner
1	Show dropdown of categories on click	2.5	Kenny
2	Display all product data based on selected category	3	Kenny
3	Allow user to filter category search	3	Kenny
4	Unit Test - Display dummy product and sort by given category, assert categories properly	4	Kenny

Completed:

We created a predefined set of categories that the Congo merchants can choose when they are creating their product. On the homepage header, whenever anyone wants to search a product, they can toggle the categories button and select one of the predefined options. The page will be redirected to the search page, displaying only products that are tagged with that specific category option and the matched regex to what the user wanted to search.

User Story 9: As a user, I would like to browse featured products

Task Number	Description	Time	Task Owner
1	Retrieve featured products from database	3	Rohith
3	Display featured products on home page, in correct place	3	Rohith
4	Unit Test - Display dummy features, fetching from database	4	Rohith

Completed:

We implemented browsing featured products by first creating functionality for verified merchants to promote their products as either a big banner, small banner, or a featured product. Once that was implemented we created a backend endpoint to return the required amount of each type of promoted product. On the frontend we retrieve those promoted products from the backend and display each of the different types in their respective place on the homepage.

User Story 10: As a user, I would like to view my purchase history

Task Number	Description	Time (hours)	Task Owner
1	Create clickable prototype	4	Ashwin
2	Retrieve previously purchased products from database	3	Ashwin
3	Display purchased products on purchase history page	3	Ashwin
4	Unit test - Display correct purchase history of user	4	Ashwin

Completed:

The frontend was set up with the clickable prototype to allow for easier implementation of the functionality of the orders history page. Once the page is loaded the previous orders for that user are fetched from the database. When they are returned each order is displayed with critical information about each of the orders.

What did not go well?

For this sprint, there was just one user story that we weren't able to complete. The story was payment with Stripe. Due to time constraints, we were forced to make decisions on how to move forward. We decided to push this story to the next sprint because all of the other functionality could be accomplished without this feature.

User Story 7: As a user, I would like to complete the payment necessary to the products in my cart.

Task Number	Description	Time	Task Owner
1	Get payment info from user	2.5	Raman
2	Send info to Stripe	2.5	Raman
3	If valid, complete and record transaction in database	2.5	Raman
4	Else, report error to user	2.5	Raman
5	Unit Test - Correctly submit transaction to Stripe and retrieve correct response code	3	Raman

Not Completed:

We did not finish much of this for this sprint. We set up our account with Stripe, however we did not implement any of it yet in our site. All of the necessary input fields are set up, however we still need to send that information to the Stripe API, and perform the necessary actions after that.

How Should We Improve?

In this second sprint, we greatly improved the rate at which we accomplished activities. We implemented three weekly meetings where we all met up and coded for 3-4 hours, bouncing ideas back and forth, and helping each other out. We found that this greatly boosted productivity.

One issue we had however, sometimes one person's code clashed with another person's. On certain pages, we had multiple people working on them. On some of these pages, we had to rewrite and clean up due to the two chunks of code not working properly together. We can fix this with communication, and commenting of the code.

Due to the now very large size of our project, comments are essential because there is so much code to read. Currently, there is very little commenting and reading code that someone else wrote takes more time than it should. For the next sprint, we will go back and comment all of the code that we wrote, so we can easily understand each others implementation. This will also make it easier to clean up things that are unnecessary, or to modify what's already there.