Calypso

Iteration 2

Start date: 2023-06-28
End date: 2023-07-06

Process

Since all mandatory front-end pages were completed in sprint 1, our team dedicated sprint 2 to figuring out which databases would be best for our application. After everyone's research and team discussions, we decided it would be best to use MongoDB for user authentication and messages while using Neo4j for everything else. This sprint was used to establish connections to both databases and to send data to them. Further, this sprint was used to add two pages, messaging page, error page and creation/connection of MongoDB and Neo4j Databases

Changes from previous iteration

1. Previously, we decided to work with MVC architecture, however, during our discussions, we noticed that Microservices architecture better suited our project. This was because everyone works on different parts of the application meaning that we can all work on separate services. This way it makes the app able to be developed in parallel and also it makes it so that we all know who to contact if we want to connect the services/pages with one another. Finally, we know that this approach for our process is successful because the pull requests have been really smooth with minimal/no conflicts and everybody has been able to work on their own tasks without depending on the development of prerequisite features from other members.

2. Changes were made to branch names and commit messages since we weren't previously including the ticket number of our user stories to every commit message and branch.

3. Started adding subtasks to main user stories since we weren't previously away subtasks did not have to be in the same format as user story

4. Only 1 reviewer is requested for pull requests [due to GitHub only allowing one request] and this reviewer must check out that branch and test the new code

Roles & responsibilities

Every team member will be doing both front end and back-end development for their assigned user stories. The different services and persons assigned as follows:

- Willam & Maaneth: Products & Media [Neo4j]
- Austin & Shadman: Transactions [Neo4j]
- Tasif: User Data [Neo4j]
- Cassandra: Messaging [MongoDB]
- Arielle: Users Authentication [MongoDB]

Events

Meetings will be held online on discord, to discuss our next steps and any difficulties. Regular standup meetings will occur 5 times during this sprint, all at 8pm via discord. Two code review meetings will be held where we discuss what we did and explain the code we wrote to others / help other members if needed.

Artifacts

- We will use JIRA to keep track of our progress on stories and which stories need to get done
- Jira will also be used to list any new subtasks that must be completed in order to complete the user story
- The highest priority tasks would be the ones which are the foundation of the app. We prioritize the functionality of our app. We would like to always create the minimum viable project for each sprint.
- Each team member chooses which user stories they are assigned to. If two or more developers want the same task, we will randomize the choice.

Git / GitHub workflow

Since everyone is working on different parts of the project, there has not been any conflicts within our codebase. For this sprint, even though multiple persons were working with products or media, they were working on different user stories, so there wasn't conflict. We will continue to separate the work items in a similar way.

The workflow is as follows:

1. Pull the latest version of main from GitHub to local version and create a branch
2. Name the branch as the user story number or even subtask number, along with a short description if needed. Publish the branch.
3. Work on code. Commit as needed throughout your process, to this branch.
4. If someone pushes and merges new changes while working on this branch, ensure to pull those changes to branch
5. When all code is completed, ensure all commits are made, and create a pull request and request the review of a fellow team member.
6. Reviewers must check out the newly published branch, and review all changes made by peer.
7. After changes / suggestions are made (if any), the reviewer will approve of the pull request. The team member that created the pull request is responsible for merging their new code to main.
8. Notify the team that your changes are merged so that others can pull to their branch.

Product

Goals and tasks

Goal:

The main goal of this sprint is to ensure connection to databases. We would need to connect to MongoDB and Neo4j. This way, we can work on sending and retrieving data for sprint 3.

Tasks:

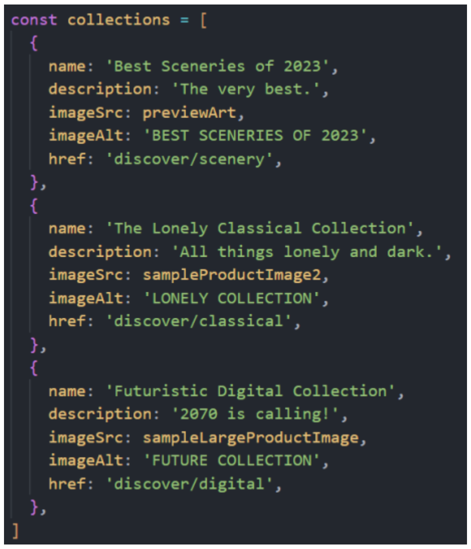
- Connect to MongoDB [cal 64]
- Connect to Neo4j [cal ?]
- User registration details collected and sent to MongoDB [cal 66]
- First iteration on messaging feature [cal 22 ?]
- Recommendation Algorithm [cal 15]
- Create different databases:
    1. Users
    2. Products
    3. Messages / Notifications
    4. Transactions
- User should be able to edit their profile
- Products should be linked to backend
- First iteration of basic messaging feature

Artifacts

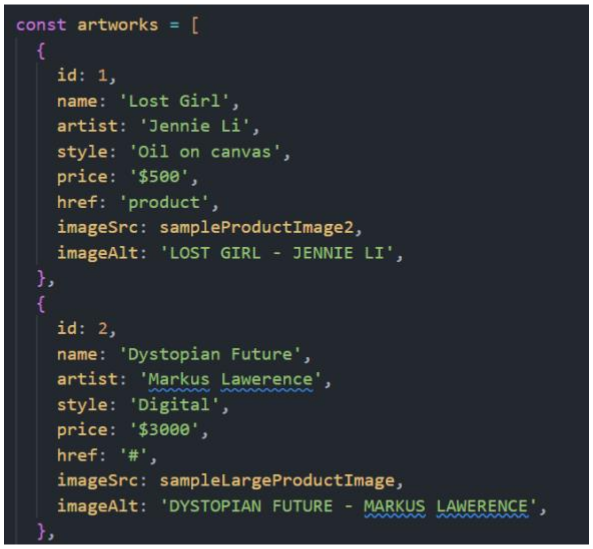
Artifact 1:

Create mock data and add it to the various databases

- Pages should be using the database to pull data to display instead of it being hard-coded.
- Users can upload products that will be stored on the database



```
const collections = [
  {
    name: 'Best Sceneries of 2023',
    description: 'The very best.',
    imageSrc: previewArt,
    imageAlt: 'BEST SCENERIES OF 2023',
    href: 'discover/scenery',
  },
  {
    name: 'The Lonely Classical Collection',
    description: 'All things lonely and dark.',
    imageSrc: sampleProductImage2,
    imageAlt: 'LONELY COLLECTION',
    href: 'discover/classical',
  },
  {
    name: 'Futuristic Digital Collection',
    description: '2070 is calling!',
    imageSrc: sampleLargeProductImage,
    imageAlt: 'FUTURE COLLECTION',
    href: 'discover/digital',
  },
]
```



```
const artworks = [
  {
    id: 1,
    name: 'Lost Girl',
    artist: 'Jennie Li',
    style: 'Oil on canvas',
    price: '$500',
    href: 'product',
    imageSrc: sampleProductImage2,
    imageAlt: 'LOST GIRL - JENNIE LI',
  },
  {
    id: 2,
    name: 'Dystopian Future',
    artist: 'Markus Lawerence',
    style: 'Digital',
    price: '$3000',
    href: '#',
    imageSrc: sampleLargeProductImage,
    imageAlt: 'DYSTOPIAN FUTURE - MARKUS LAWERENCE',
  },
```

Sample images of database structure

Artifact 2:

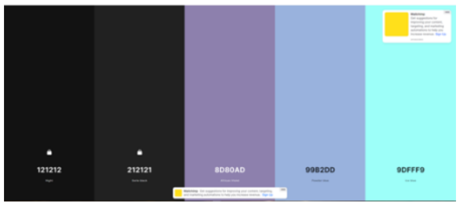
Have most pages be functional

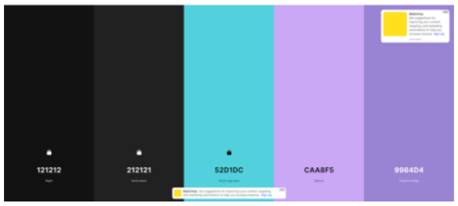
- Transaction page should be functional (transactions made with mock data for now)
- The intended function of each page should be completed at least to a bare-bones standard
- Register page now accepts user information and posts to MongoDB

Artifact 3:

Make the different pages adhere to the same styles in terms of colors, fonts, buttons.

- All the pages should use the same colour palette and the same font palette


121212
212121
8D80AD
99B2DD
9DFFF9


121212
212121
52D1DC
CAA8F5
9984D4