CS387 Project Design

Kartik Gokhale, Dhruv Rambhia, Advait Padhye, Dhairya Jhunjhunwala
200100083, 200050115, 200100014, 200010021

Spring 2023

1 Introduction

Our project title is "***E-Commerce Web Application***"

Our team members are

- Kartik Gokhale, 200100083
- Dhruv Rambhia, 200050115
- Advait Padhye, 200100014
- Dhairya Jhunjhunwala, 200010021

2 Problem Statement

Our goal is to create a web application that will serve as an e-commerce website. Users can sign up with personal details, such as name and address and payment methods. These will be used for the physical transfer of goods as well as authentication. Users can act as buyers and sellers. Buyers can access other user reviews to browse through the items they require as well as choose the best price. They will also receive recommendations based on their recent searches as well as the goods and sellers with the top reviews. Along the same lines, they can also provide reviews about sellers from whom they have made a purchase and about the items, which will serve as a feedback mechanism as well as an authenticity check on the seller. Meanwhile, users, as sellers, can put up their products on sale, advertise their products as well as provide customisation options as necessary. The sellers can use the calendar feature to set a delivery date for their product and other time-based features.

3 Users

This section covers the broad user roles to interact with the system. Broadly speaking, there are 2 user roles, the buyers and sellers. The intended users of this database system along with their corresponding use cases are the following

- Buyers: These will be the ones who join the website from the perspective of using it for buying goods. The corresponding use cases will be
    1. Signing up to use the system.
    2. User provides details to log in, which will be authenticated. On successfully logging in, the session storage will store the required details to keep the user logged in until logging out.
    3. User provides a search string that will do a 'grep' in the product name and displays all products containing the searched string.

4. User chooses a genre or category of objects amongst the ones available, which will restrict the results of the above use case to a particular category.
5. User visits product page where there are various details such as reviews of products and the seller. The user selects the options regarding payment and submits these as a form.
6. Buyer can add a review for any seller product they have purchased.
7. Buyer can visit the Recommendation Page where they will be recommended certain products based on their recent purchase history.
8. Buyer can view their previous purchases and also view the delivery dates of their purchased products using the Calendar feature.
9. Buyer can update their profile by providing their email address and residential address.

- Sellers: The ones who begin sales. The use cases are
  1. Sign up to use the system.
  2. Add sales with product details and price.
  3. Accept/Reject sale options.
  4. Set delivery dates for the products.
  5. Update profile for others to view.

4 Databasing Requirements

We will be using Postgresql as the database backbone of the application. The main databasing requirements are the following

- Store the personal details such as ID, username, email, hashed password, and address per buyer and seller.
- Store the sale details of every product on sales, such as (ID, price, dimensions, type, seller id).
- Store the transaction details recording the (ID, buyerID, sellerID, amount, date, objectID).
- Store details about transaction status such as (pending, transactionID)

These are the broad headings and ideas for the database tables. Each of these will have 'support' relations containing additional data. There will also be views created on the object details for buyers to view. Permissions will be appropriately granted based on the buyer and seller role created, and checks will be made to maintain integrity throughout the data.

The entities along with each of the attributes are shown in Figure 1. The relationship between each of the entities along with the primary keys is also shown. These are the basic entities that will cover the main features of the database.

5 Structure

The main structure of the application is that it is going to be a web application. The main interfaces will be

- Login & SignUp Page
- User Home Page: Welcome Page to the website
- User Profile Page: Contains personal details

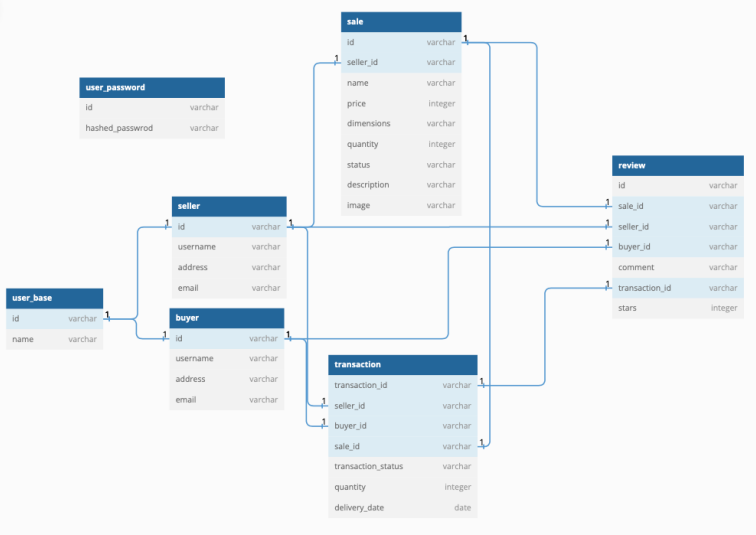

Figure 1: ER Diagram

- Update Profile Page : Update the personal details
- Search Page: For Buyers to look for goods
- Reviewing Page: A couple of pages to view objects, give reviews and access other user reviews.
- Create Sale Page: For sellers to put up new items for sale.
- Payment Page: Page to Complete transactions
- Calendar Page: For sellers to choose a delivery date.

5.1 Backend

We are using NodeJS for the backend. We will be implementing the following API endpoints, which will help in the smooth functioning of our website. It is to be noted that in almost all API endpoints except User Registration and Authentication, we retrieve the User ID from the sessions. In addition to this, we can store other information in sessions, such as whether the current user is a buyer or seller.

- **User Registration** : When a new user registers on the website, the details, particularly the user's ID and the password that he has set during registration, will be stored on the database.
  Input: User ID and Password
  Output: Whether the user could be registered or not (Status of Registration)
- **User Login and Authentication**: When a user logins in on the website, we must verify the identity of the user, which is done with the help of the user password. Moreover, the passwords which are entered in the database must be hashed so that in case the database is hacked, the user credentials won't be compromised. For the purpose of password hashing and comparison, we will be using the bcrypt library.
  Input : User ID and Password
  Output : Authentication status of the user.
- **User Profile**: This page displays all the necessary information about the user.
  Input: User ID
  Output: Profile Details
- **Update Profile**: This endpoint enables the user to update his profile (email and address).

Input: User ID, Profile description
Output: Status Of Update

- **Product Page**: This page displays all the necessary information about the product.
  Input: Sales ID
  Output: Information about the product, Seller ID, Reviews for this product by other users.
- **Available Products**: When a buyer lands on its home page, it must be able to view all the products that are available for purchase.
  Input: Buyer ID
  Output: List of available products and their information.
- **Pending Transaction**:A user must be able to view all the products that are currently pending in Transaction.
  Input: User ID
  Output: List of Transactions that are pending.
- **Previous Purchases**: Buyers must be able to view all the products that they have purchased as well as view all the products that have been dispatched.
  Input: Buyer ID
  Output: List of previous purchases and List of dispatched products.
- **Category-based Products**: A buyer must be able to view all products related to a particular category or genre (Eg. Furniture, Toys, etc.)
  Input : Category Name/ID
  Output : List of products belonging to that category.
- **Purchase**: This endpoint will help in completing a transaction for a given product
  Input: Buyer ID, Sales ID, Seller ID
  Output: Status of Transaction
- **Reviews**: The customers should be able to post reviews of a particular product/seller that they have purchased
  Input: Buyer ID, Sales ID, Transaction ID, Review content
  Output: Status of whether the review was published or not
- **Items for Sale**: A seller must be able to add more products for sale.
  Input: Seller ID, Product Description
  Output: Status of whether the product was put up for sale or not
- **Inventory Management**: This endpoint will enable sellers to increase or decrease the number of items and delete certain products.
  Input: Seller ID, Sales ID
  Output: Final Quantity of Product available for purchase or Status of Deletion.
- **Transaction Status**: A seller must be able to accept/reject the offers for his products.
  Input: Seller ID, Transaction ID
  Output: Status of Transaction

In addition to these API endpoints, we will be using KNN Algorithm in the backend for implementing the recommendation system, which will help in recommending products to buyers. To get the recommendations, we spwan a new child process in the backend which runs the particular model and outputs the recommended products. Input: User ID, Purchase History of User, Sellers with Highest Ratings
Output: List of recommended products.

5.2 Frontend

We are using React for the frontend. e will be implementing the following routes to ensure a smooth and convenient user experience.

- **Landing Page**: Upon visiting the website, the first page will be the homepage comprising of the following buttons
  - **Button** : Link to Registration Page
  - **Button** : Link to Login Page
- **Dashboard**: On successful login, the user will be able to see the following buttons
  - **Button** : Link to Buyer Portal
  - **Button** : Link to Seller Portal
- **User - Profile Page**: The users can view the profiles of various other users
  - **Table**: Table containing the profile information about the user such as name, email address, residential address, etc.
- **User - Update Profile Page**: The users can update their profile which will be visible to the other users.
  - **Form**: Fill the updated user information such as email address or residential address
- **Buyer - Home Page** : When a user logins in on the website, and goes into the buyer portal, he will see the following
  - **Table**: Table containing each and every available product, with the columns containing seller_id, available inventory, and price.
  - **Table**: A second table consisting of pending transactions of the buyer.
  - **Button**: Each product will have a button to buy the product, redirecting the user to the payment page.
  - **Link**: Each product will be hyperlinked to its own product page where it will contain additional information about the product.
  - **Link**: Each seller name will be hyperlinked to its own profile page where it will contain additional information about the seller.
- **Buyer - Search**: The buyer portal will contain a search bar, which will link to a search result page.
  - **AutoComplete Search Bar**: We will implement a React Autocomplete Search Bar, which will suggest products as the user types it in, as well as provides the option to search all products with the given substring.
- **Buyer - Category-based Products**: The buyer portal will contain a menu to choose between subcategories of products
  - **SideBar**: We will implement a side menu to provide links to each category of products. There will be a miscellaneous category for the products not segmented in any defined category.
  - **Button**: Each option of the side menu will link to a page displaying available products of that category only.
- **Buyer - Add Reviews**: The buyer will be able to post reviews about the products that he has purchased in the past.
  - **Stars**: Five Stars will be displayed where the user will have the option to select the number of stars for the review.

  - **Form** : Fill the review content for the particular purchase.

- **Buyer - Previous Purchases**: The buyer will be able to view the delivery dates of the purchases products with the help of a Calendar.
  - **Calendar**: A Calendar to view all the delivery dates for various products purchased by the user.
  - **Table**: Table containing the previous purchases of the user along with the information of each purchase.
  - **Table**: Table containing the products purchased by the user that have been dispatched.
  - **Button**: There will be a button to add a review which will allow the user to add a review if they have purchased the product in the past, and give an error otherwise.
- **Buyer - Payment Page**: To complete the payment of a product, the users must fill some information like the quantity of goods that they would like to purchase.
  - **Form**: Fill the necessary details to complete the payment.
  - **Button**: There will be a button to confirm the payment for the given product.
- **Buyer - Recommendation Page**: Using the KNN Algorithm, the buyer will be recommended certain products based on their purchase history.
  - **Table**: Table containing the recommended products and the seller of that product
  - **Link**: Each recommended product will be hyperlinked to its own product page where it will contain additional information about the product.
  - **Link**: Each seller name will be hyperlinked to its own profile page where it will contain additional information about the seller.
- **Product Page**: Each product will have its page, where we have product information as well as user reviews.
  - **Table**: Table containing the information about the product such as price, available quantity, dimensions, item description, etc.
  - **Image** : Display an image of the product provided by the seller
  - **Table**: Table containing reviews of past buyers of the product. It will contain the reviewer name (hyperlinked to its profile page), product rating, and review body.
- **Seller - Home Page** : The seller portal will contain the following details
  - **Table**: Table containing every product listed by the seller, with the columns containing inventory remaining, and price.
  - **Button**: Each product will have a column with a button to increase or decrease inventory, depending on availability.
  - **Table**: Table containing all the pending transactions for the products put for sale by the user.
  - **Button**: Each product will have a column with a button to either accept or reject the transaction.
  - **Table**: Table containing all the previous sales of the products listed by the seller
- **Seller - Add New Item** : The seller can add new products for sale
  - **Form**: Fill the necessary details of the product such as price, quantity, dimensions and item description that will be displayed on the product page.
  - **Button** : Seller must upload an image of the product which will be stored in the backend.

6 Future Work

- As suggested by the Professor, spwaning a new child process each time when a buyer requests for recommendation is extremely slow. To overcome this, we plan to use pre-trained models and instead of child process, use a service.
- Add a new user role called Moderator. This particular role will be responsible for the following actions.
  - Manage the provided reviews and censor them if necessary, using the delete review method
  - Manage users with consistently bad reviews using the delete account method
  - Remove products and sales that are dangerous or illegal by using the delete sale method.
- Currently our portal allows only one way communication (Buyer to Seller). We plan to integrate a messaging system where the seller can reply back to the reviews.

7 Our Learnings

This project was very helpful for getting used to web development using React and Node JS. During the development of this project, we used several support libraries provided by React to implement certain features such as Calendar and AutoComplete Search. This helped us in getting acquainted to the powerful libraries developed by the React Community.

A major challenge that we faced during the development of the project was to implement the image upload functionality where the seller can upload the image of a product. For implementing this functionality, we had to use multipart/form-data that is used for uploading files. In Node, we used Multer which is a middleware for handling multipart/form-data requests. Developing this API endpoint was a new learning experience for us since we had always used application/json header in the requests.

The most important feature of the website is the recommendation system. Implementing this functionality consisted of following two major tasks.

- Implementing a ML algorithm for the recommendation system
- Generating Data for the recommendation system since there was not any pre-existing database which matched with our schema.

We came across several approaches for implementing the recommendation system such as content based systems, collaborative filtering systems, etc. Finally, we decided upon using User based Collaborative Filtering since it required the review information which we had incorporated in our schema.

For training the ML model, we need huge amount of data which we obtained from Amazon Reviews Dataset. However, this dataset didn't exactly match with our database schema. Therefore, to maintain database consistency, we needed to generate some random data. During data generation, we needed to take care of several functional dependencies present in our database such as `ItemID` $\longrightarrow$ `SellerID`. After identifying all the functional dependencies, we generated a master csv file which consisted of all the necessary columns from all the relations. After generating random data in the master file, we split it into the original relations present in our schema. In this way, we generated data for training our recommendation system.

Developing the recommendation system helped us to learn various ML algorithms and their use cases. During the data generation phase, we learnt about identifying several functional dependencies present in our schema and incorporating them while designing the final database.