

**College of Engineering**

**CMPS 350: Web Development Fundamentals**

**Spring 2024**

**Course Project Phase 1**

**E-Commerce Platform - Digitalogy**

**GitHub Link:**

<https://github.com/tamader-ta2102604/Digitalogy>

**Student Names:**

Muneera Alnaimi 202102530

Tamader Aldosari 202102604

Aisha Alassiri 201901690

**Introduction:**

"Digitalogy," our e-commerce platform that is developed specifically for digital assets and products. Our platform is designed to help the user to easily access, buy, and sell digital assets, with an emphasis on digital products. A user can be either a customer who can buy from the featured digital products, a seller who can sell a specific product or an admin who manages the platform. The design of our e-commerce platform combines simplicity and integrates with the digital theme, in addition to the design our platform prioritizes functionality, ensuring that users can easily search and purchase the products.

The localStorage object is used for storing data in our web browser's storage. It allows data to be saved and retrieved across browser sessions. In this code, localStorage is used to store and retrieve all the data related to customers, admins, and sellers.

Class diagram:

A diagram of a customer

Description automatically generated

**Use case 1: Login**

**A screenshot of a computer

Description automatically generated**

From the home page, the user can login with the username and the password and all users are retreieved from the json file, and once the user is logged in the website, he will be redirected back to the home page and can proceed with the provided functionalities depending on his role (admin, seller or customer). The login function takes the username and password the user inputs and uses the .find method to check whether this user exists by comparing the users username and password.

**Use case 2: search available items**

**A computer screen shot of a computer

Description automatically generated**

Within the main page, we added a search bar that helps the user to search for a specific item that is available within our products. As shown above, when we type laptop we get the item. The search method was implemented to get the user input and goes through the products (cards) and if any of these cards include the given input then it is displayed.

**Use case 3: purchase an item**

**A computer screen shot of a computer

Description automatically generated**

As a pre-condition of purchasing an item the user should be logged in and verified as a valid customer, so in this screenshot the user is not logged in and he can not purchase as the purchase button is not showing but all the item details including the price is shown.

A computer screen shot of a computer

Description automatically generated

Once the user is logged in as a customer he can purchase any of his desired items and the purchase button is now shown to him and then he is redirected to this page where he can proceed by choosing the item quantity and the shipping address and the quantity and address are retrieved and stored as entities from form fields.

A screenshot of a computer

Description automatically generated

**Use case 4: view the purchase history**

A computer screen shot of a computer

Description automatically generated

In the purchase history, the transaction id, item name, quantity of each product, total price and the buyers address is all shown. In this case, the user choose 2 items and all their details are displayed. The transaction id is generated through a function that is called generateTransactionId() and it generates a 6-digit random ID number. Then in the purchaseItem(itemId, quantity, address) function that handles all the purchases, once the purchase is done it updates each of the sellers sale history and the buyers purchase history by setting the attributes, and finally updates the item quantity by reducing it with the number of the quantities purchased.

This table shown above is generated through the generatePurchaseHistoryTable() function, it first clears out any existing content and creates a table element and appends whatever purchase is done.

**Use case 5: view available items on sale and sale history**

**A screenshot of a computer

Description automatically generated**

As a seller, the seller can view the sales history. In the displaySellerSaleHistory() it first creates a div element and a card for each item of the sales items and it then adds it using the .innerHtml with the name, image, quantity and price and displays all the products to the seller.

**Use case 6: upload an item to be sold**

**A screenshot of a computer

Description automatically generated**

First of all, when a seller is logged in and wants to upload a product to be sold, he is forced to add all these products details like the name of the product, price of the product, available quantity and a picture to be displayed in the website. In the displayProducts() function when the role of the user is a seller, we can create an element whenever the seller wants to upload a product then all the inputs are injected using .innerHtml and appended to be displayed. And the uploadItem() function sets all the fields with the input got from the seller.

**A screenshot of a computer

Description automatically generated**

The website shows this message when everything is right and all fields are not empty.

A screenshot of a computer

Description automatically generated

The website shows this message whenever one of the fields are missing.

**A computer screen shot of a white charger

Description automatically generated**

Finally, when everything goes right the product is uploaded within all other products and the customers can easily view and purchase it.