**E-Commerce Application on IBM Cloud Foundry**

***Srividhya***

***912421104044***

**A cloud with icons and symbols

Description automatically generated with medium confidence**

**Phase-3: Development Part-1**

**Problem Statement:** Build an artisanal e-commerce platform using IBM Cloud Foundry. Connect skilled artisans with a global audience. Showcase handmade products, from exquisite jewelry to artistic home decor. Implement secure shopping carts, smooth payment gateways, and an intuitive checkout process. Nurture creativity and support small businesses through an artisan's dream marketplace!



**Development Part 1**

**1. Create an IBM Cloud Account:**

If you don't have an IBM Cloud account, sign up for one. You can do this by visiting the [IBM Cloud website] (**https://cloud.ibm.com/registration**) and following the registration process.

**2. Choose the Appropriate Database Service:**

Select the IBM Cloud Database service that best suits your project's needs. As mentioned earlier, you can choose between Db2 or MongoDB, depending on your dataset and requirements.

**3. Set Up a Database Instance:**

**For Db2:**

* Log in to your IBM Cloud account.
* From the IBM Cloud dashboard, click on the "Create Resource" button.
* In the catalog, select "Databases" and then "Db2."
* Follow the on-screen instructions to configure your Db2 database instance, including specifying the instance name, region, and other settings.
* Create the instance.

**Platform Layout:**

**1. User Interface (UI) Design:**

- Define the user interface layout, considering user experience and accessibility.

- Determine the structure of the platform, including the homepage, navigation menus, product pages, and search functionality.

**2. Homepage:**

- The homepage typically features a search bar, product categories, featured products, and promotional content.

**3. Product Listing Page:**

- Design a page to display a list of products with filtering and sorting options.

- Consider how products will be presented, including images, titles, descriptions, prices, and ratings.

**4. Product Detail Page:**

- Create a detailed view for individual products, showcasing additional information, images, specifications, and reviews.

**5. User Registration and Authentication:**

- Design pages for user registration, login, and profile management.

**6. Shopping Cart and Checkout:**

- Create a shopping cart for users to add products before checking out.

- Design checkout pages for users to provide shipping and payment information.

**7. Admin Dashboard:**

- If your platform will have administrators, design a dashboard for managing products, users, and orders.

**Database Design:**

For storing product information, you'll need a database. Here's an example of a simple database schema using SQL (Structured Query Language) for a product platform:

**1. Products Table:**

- Fields: product\_id (Primary Key), name, description, price, category\_id (Foreign Key), image\_url, stock\_quantity, manufacturer, ratings, etc.

**2. Categories Table:**

- Fields: category\_id (Primary Key), name, description, parent\_category\_id (for subcategories).

**3. Users Table (if you have user accounts):**

- Fields: user\_id (Primary Key), username, email, password (hashed), profile information.

4. \*\*Orders Table (for order history):\*\*

- Fields: order\_id (Primary Key), user\_id (Foreign Key), order\_date, total\_price, status, etc.

5. \*\*Order Details Table (for individual order items):\*\*

- Fields: order\_detail\_id (Primary Key), order\_id (Foreign Key), product\_id (Foreign Key), quantity, price, subtotal, etc.

**Development Technologies:**

**1. Front-End:**

- HTML, CSS, JavaScript

- Front-end frameworks like React, Angular, or Vue.js

**2. Back-End:**

- Server-side scripting languages like Python (Django/Flask), Ruby (Ruby on Rails), Node.js (Express), or PHP (Laravel).

- API design for communication between the front-end and database.

**3. Database Management:**

- Relational Database Management System (RDBMS) like MySQL, PostgreSQL, or SQL Server.

- Use SQL for creating and managing the database.

**4. Hosting and Deployment:**

- Choose a hosting provider (e.g., AWS, Azure, Heroku) and deploy your application.

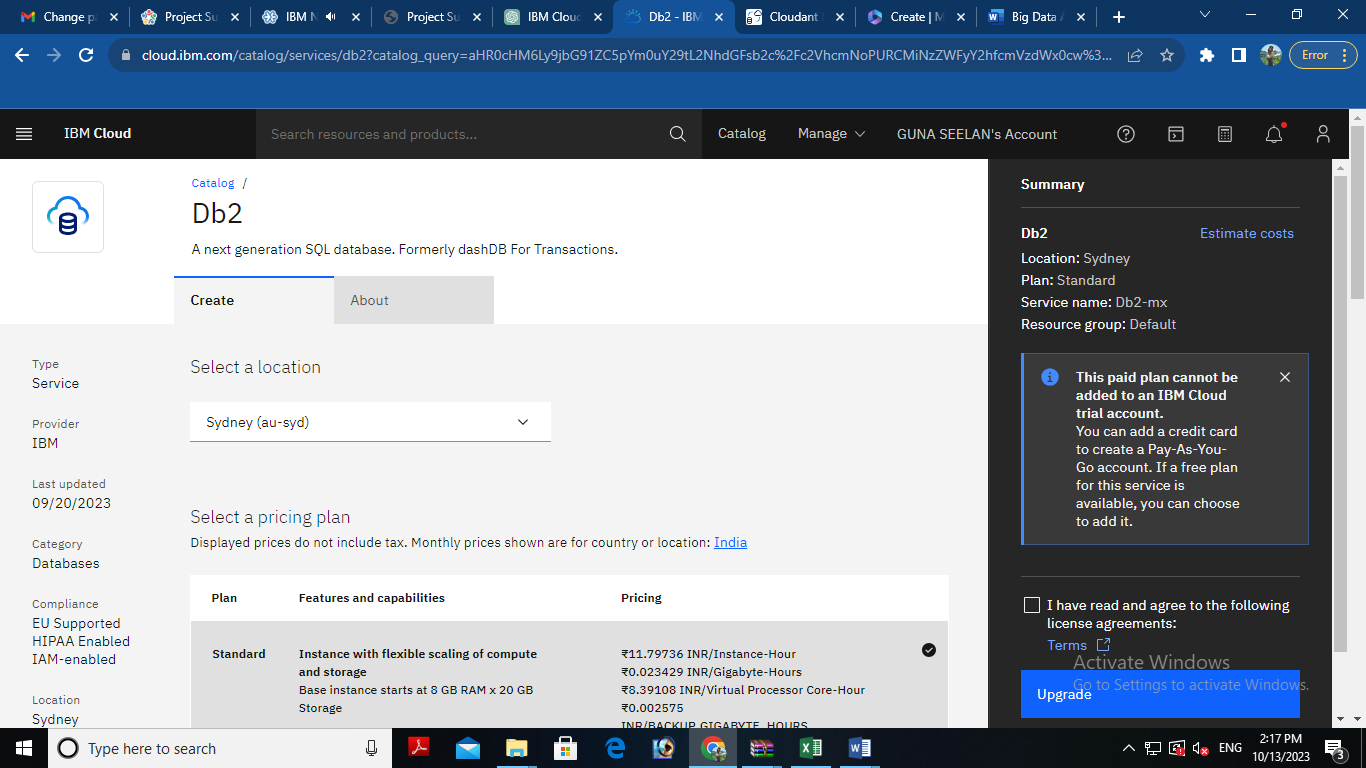
**5. Security:**

- Implement security measures, including data encryption, user authentication, and authorization.

**6. Scalability:**

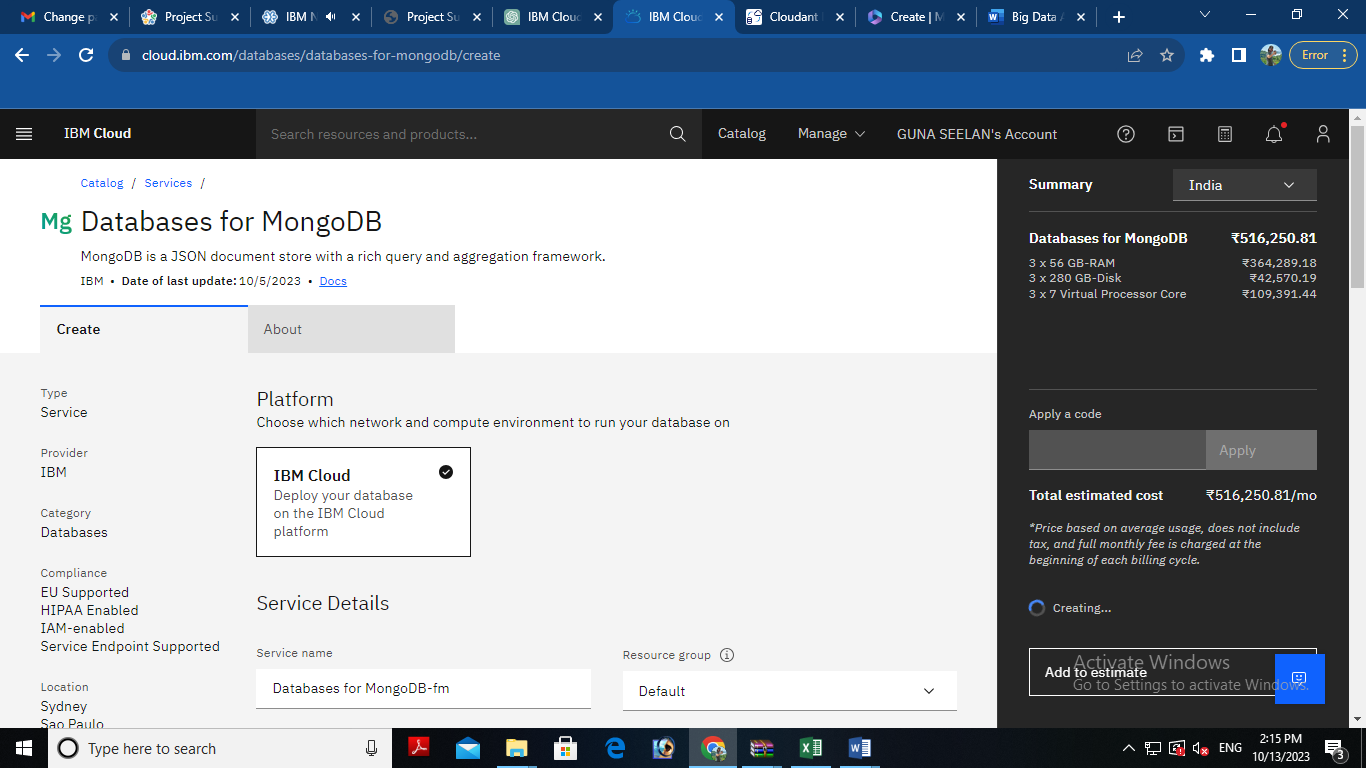
- Design the database and platform to scale as your user base grows.

Remember that designing and developing a platform is a complex process, and you may need to adapt the above suggestions to your specific needs and constraints. Additionally, you might consider using a web development framework or content management system to expedite the development process.



**For MongoDB:**

* Log in to your IBM Cloud account.
* From the IBM Cloud dashboard, click on the "Create Resource" button.
* In the catalog, select "Databases" and then "MongoDB."
* Follow the on-screen instructions to configure your MongoDB database instance, including specifying the instance name, region, and other settings.
* Create the instance.

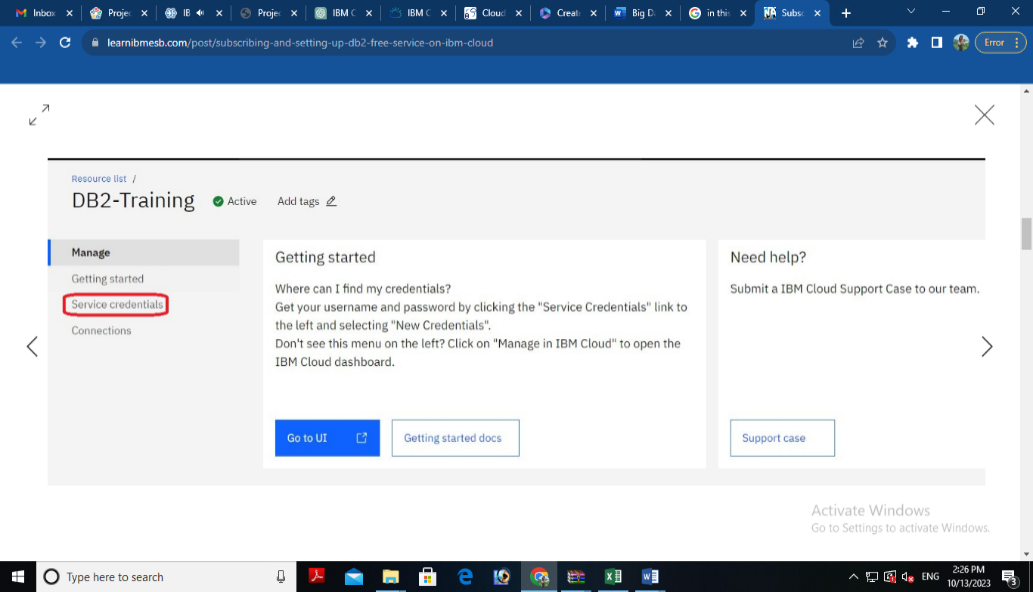


**4. Develop e commerce app:**

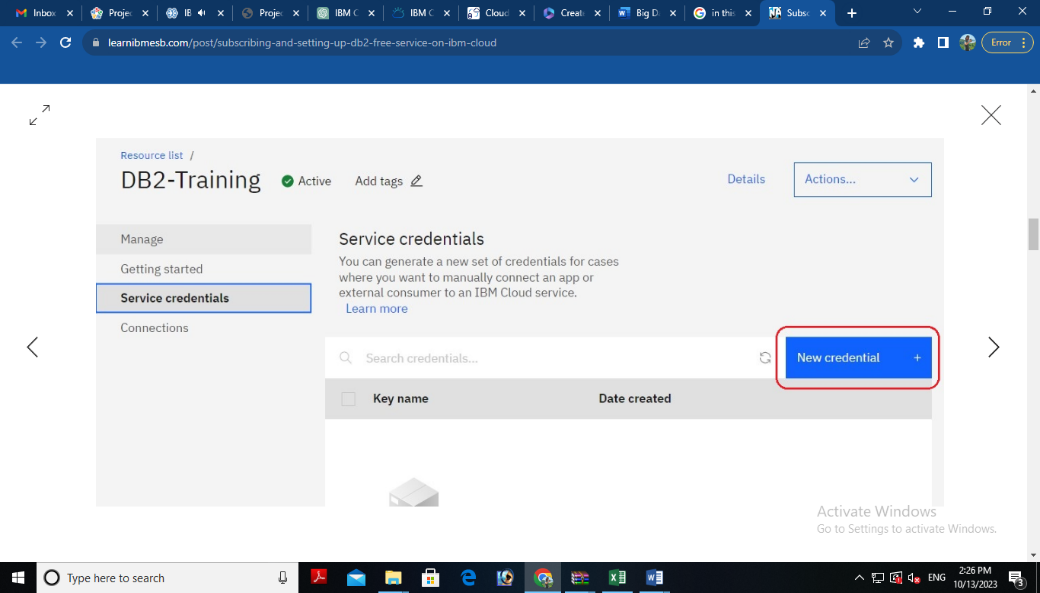
After setting up your database instance, you can start developing queries or scripts to explore and analyze your dataset. The type of queries and scripts you write will depend on the nature of your dataset and your analysis goals. You can use SQL for Db2 or MongoDB's query language for MongoDB.

## **Creating Service Credentials the IBM DB2 database**

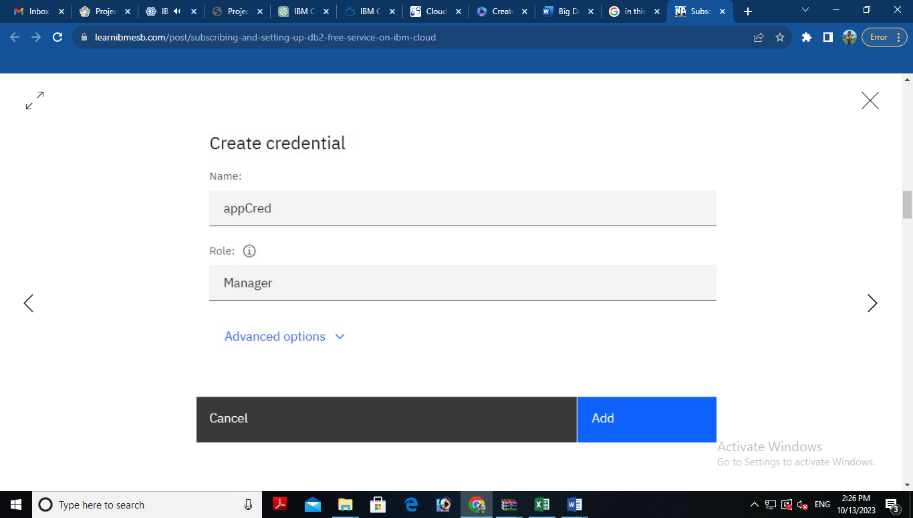
* In the resource list screen of IBM Cloud, click on the DB2 service (displayed under Services and software category) that you created
* From the service page, select the menu option "**Service Credentials**" to create / access the credentials of the db2 database



* Click on **New Credential** button in the Service Credential page to create a new credential

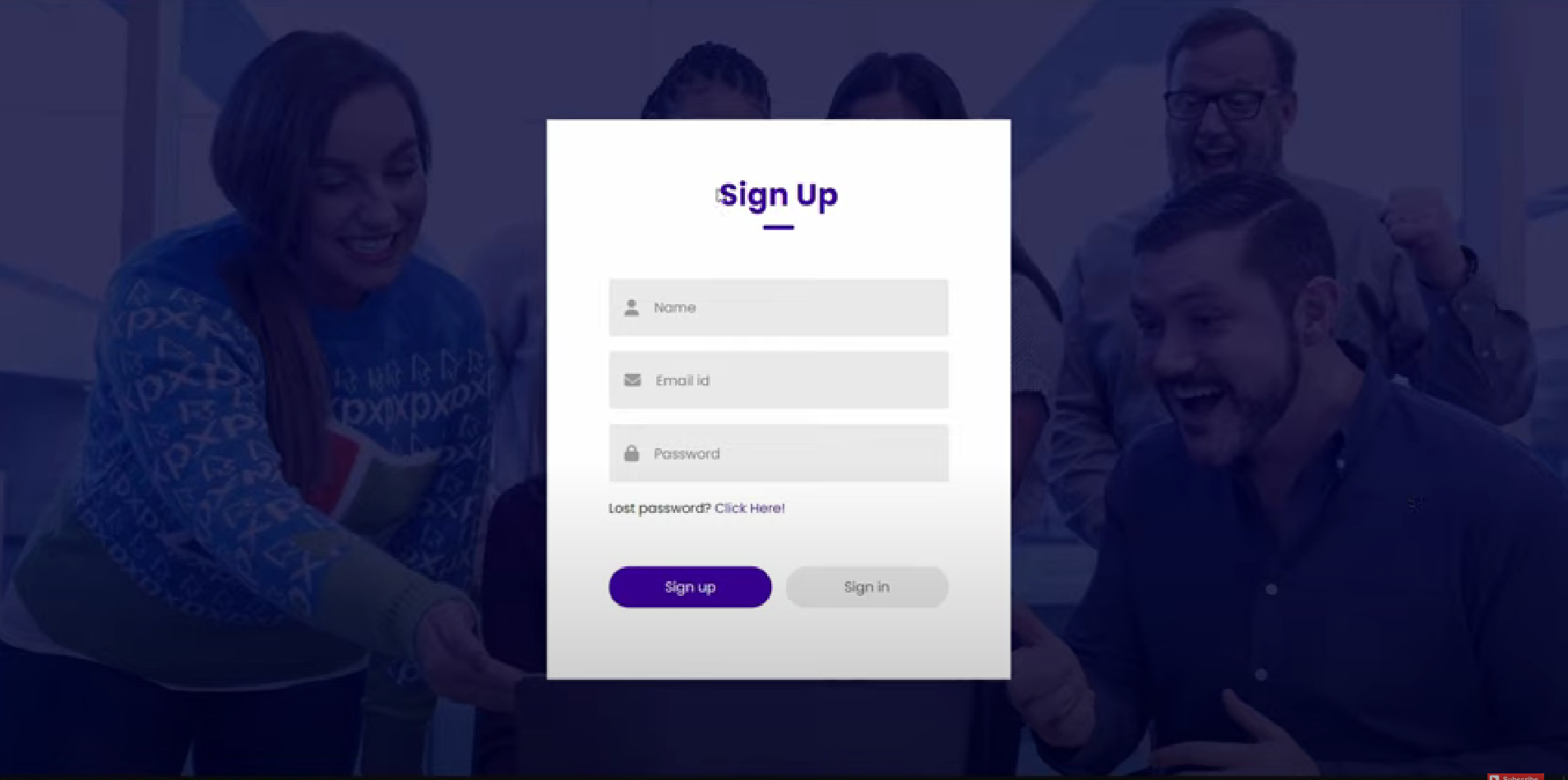


* Provide the any name for service credential (e.g. **appCred**) and click on **Add**

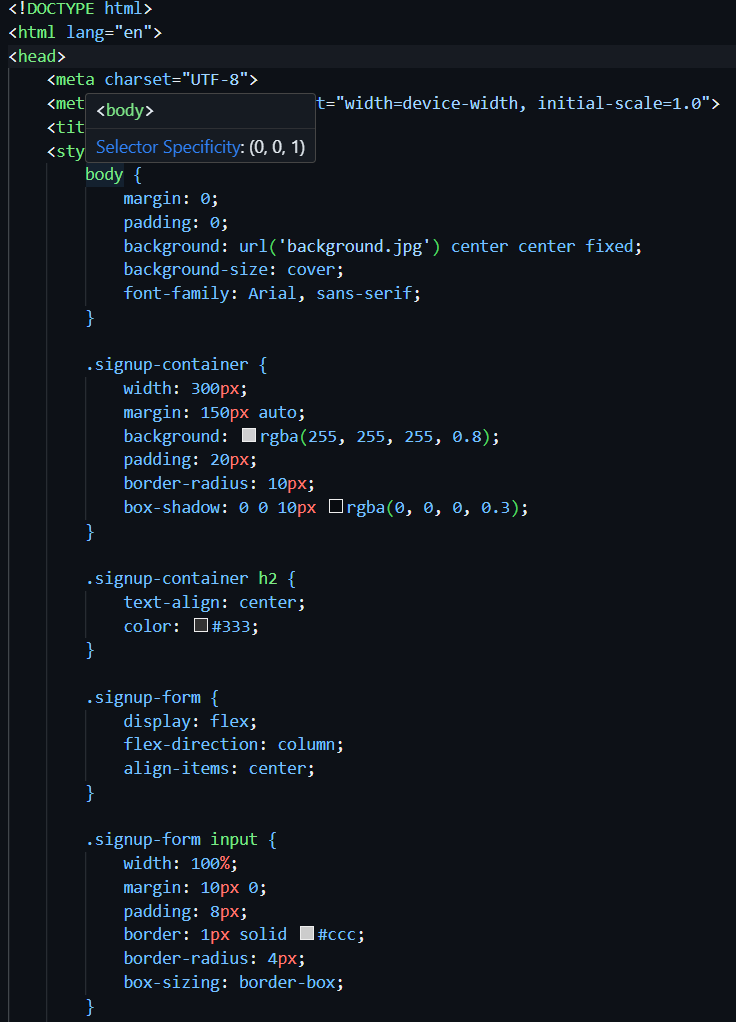


* New credential gets created and is displayed. Expand the newly to created credential to get the all the details that is required for client application to connect to the database. Note down the value for the following properties separately, which we will use it later to configure our application to connect to this database.

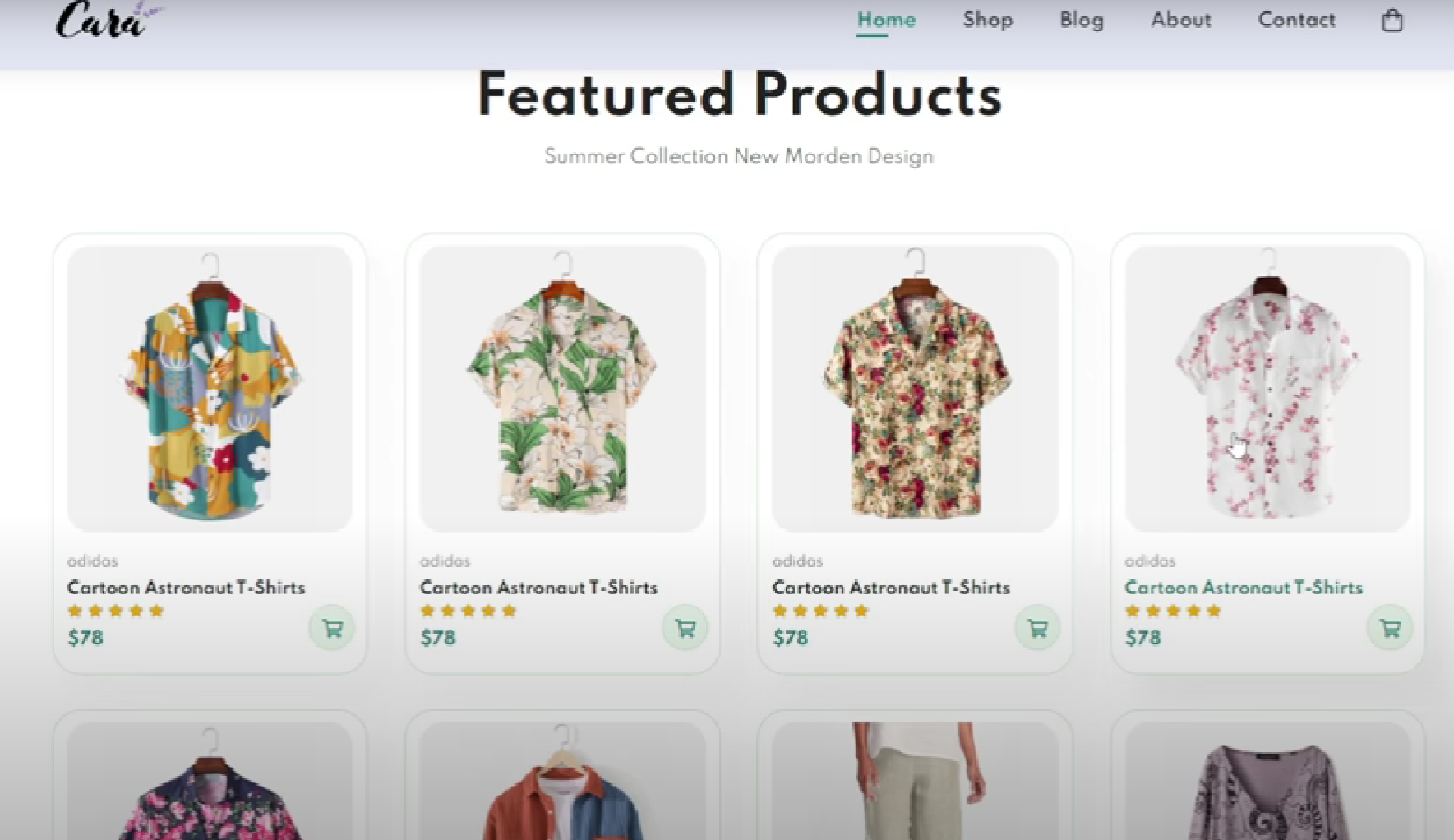
**Creating authentication page:**

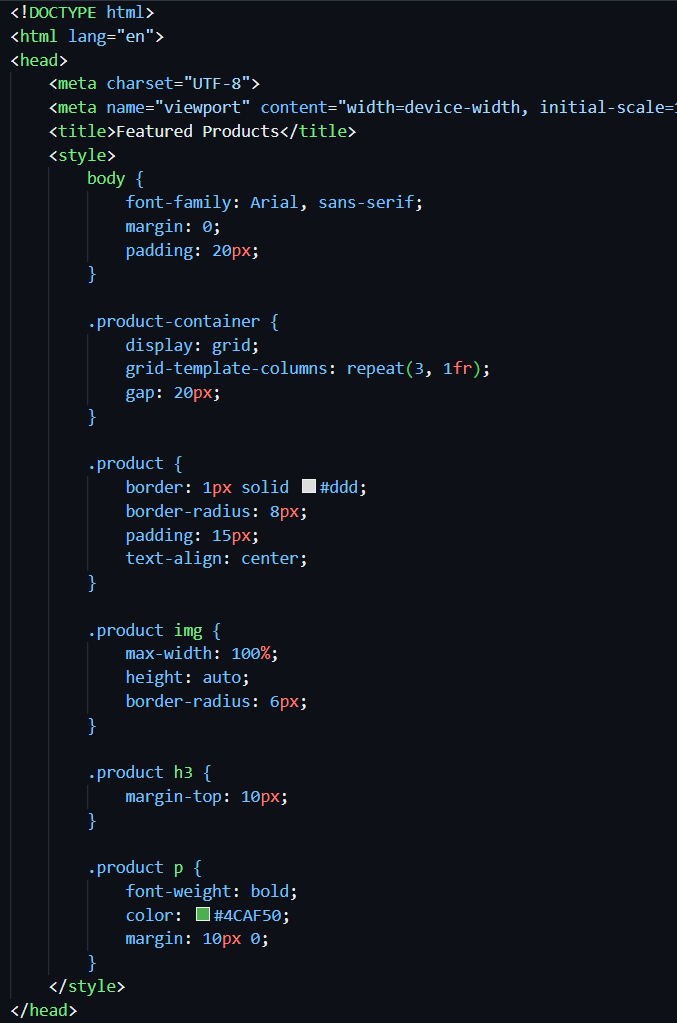
****

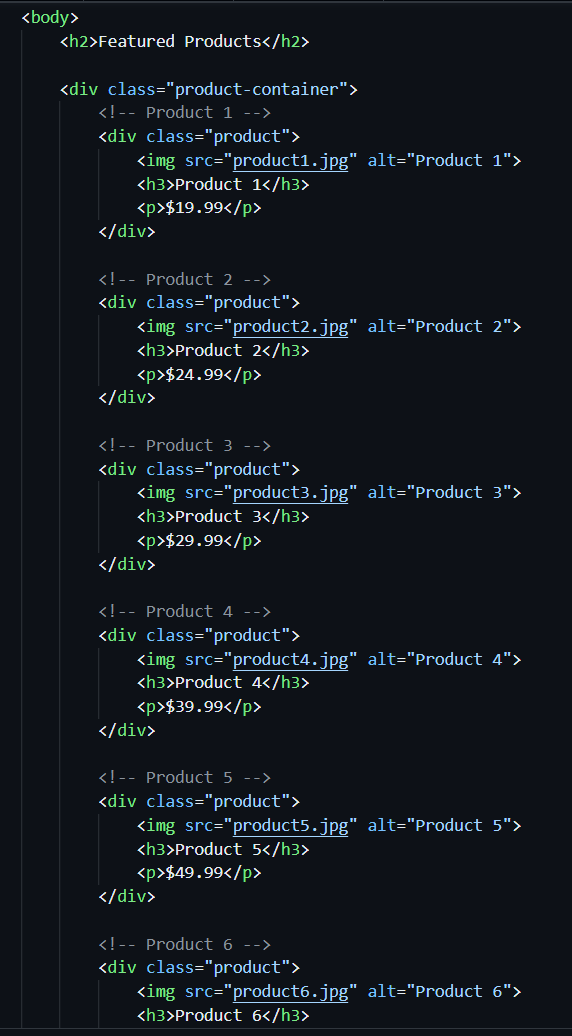
Html code is given below


**Creating catalog for products:**

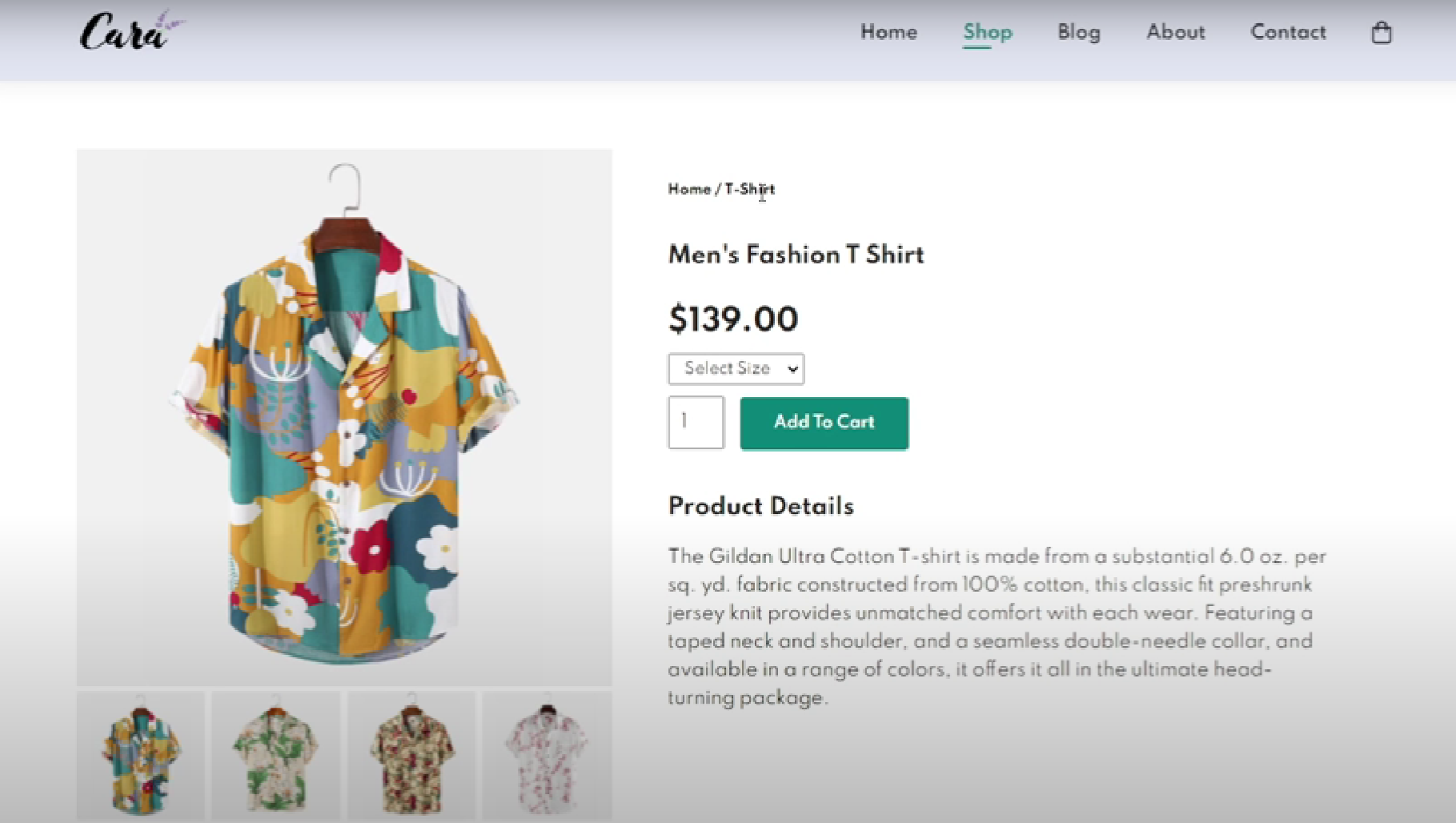
****

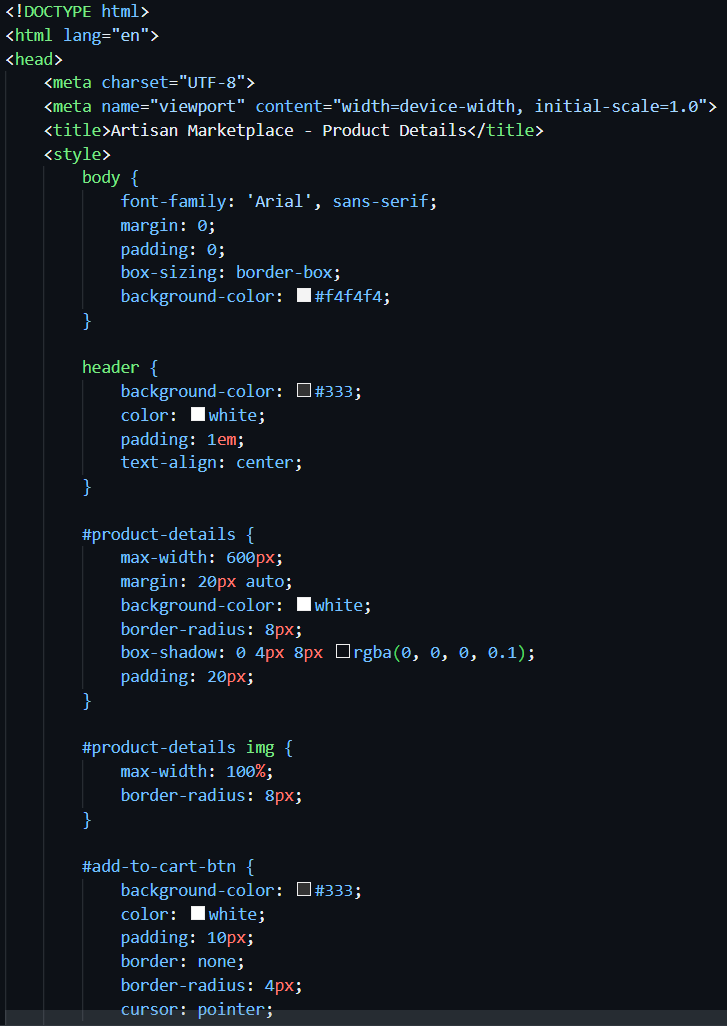
html code for featured products 

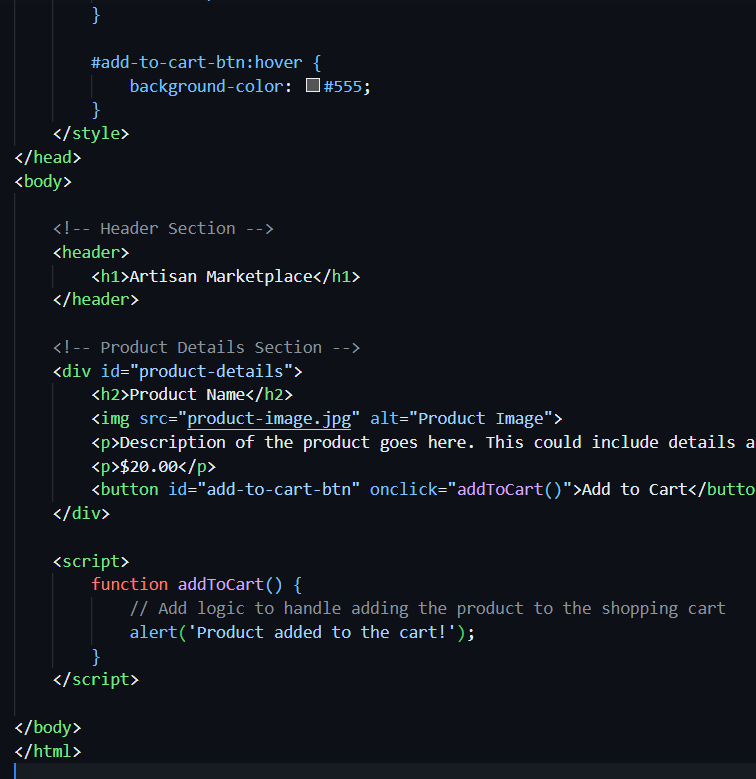




**Creating single product description:**

****

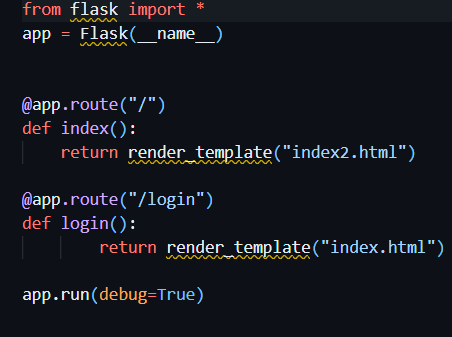
**Html code for given **

****

**Connecting database:**

As part of your data storing you may need to perform data cleaning and transformation. This can involve database

Remember that I can provide guidance, answer questions, and help with SQL queries or MongoDB queries if you encounter specific issues during your project. Feel free to ask for assistance with any part of your project, and I'll do my best to help you successfully complete it.

****

8

Create a database to store product information such as images, descriptions, prices and categories.

Creating tables : **Products Table:**

**Attributes:**ProductID (Primary Key, Auto-incremented),ProductName (String),Description (Text),Price (Decimal),CategoryID (Foreign Key),ImageURL (String)

**Categories Table:**

**Attributes:**CategoryID (Primary Key, Auto-incremented),CategoryName (String)

**Use Cases From Backend**

**Product Listing**: Retrieve a list of products with their names, prices, and images for displaying on the website.

**Sql query*:*** *SELECT ProductName, Price, ImageURL FROM Products;*

**Category Filtering:** Allow users to filter products by category by querying the "Products" table based on the CategoryID field.

**Sql query:** *SELECT ProductName, Price, ImageURL*

*FROM Products*

*WHERE CategoryID = category\_id;*

**Product Details:** Display the detailed product description and price when a user selects a specific product.

**Sql query:** *SELECT ProductName, Description, Price, ImageURL FROM Products WHERE ProductID = product\_id;*

**Adding New Products:** Insert new products into the "Products" table, specifying the product's name, description, price, category, and image URL.

**Sql query:***INSERT INTO Products (ProductName, Description, Price, CategoryID, ImageURL)*

*VALUES ('New Product Name', 'Product Description', 19.99, 1, 'image\_url.jpg');*

**Updating Product Information:** Modify product details such as price, description, or image URL when necessary.

**Sql query:***UPDATE Products*

*SET Price = 24.99, Description = 'Updated Description'*

*WHERE ProductID = product\_id;*

**Deleting Products:** Remove products from the database when they are no longer available.

**Sql query:***DELETE FROM Products WHERE ProductID = product\_id;*