**E-COMMERCE APPLICATION ON IBM CLOUD**

**Team Member**

**SATHISH A - 310521104108**

**Phase 3 Document Submission**

**Creating a DB2 Database on IBM Cloud for Storing Data**

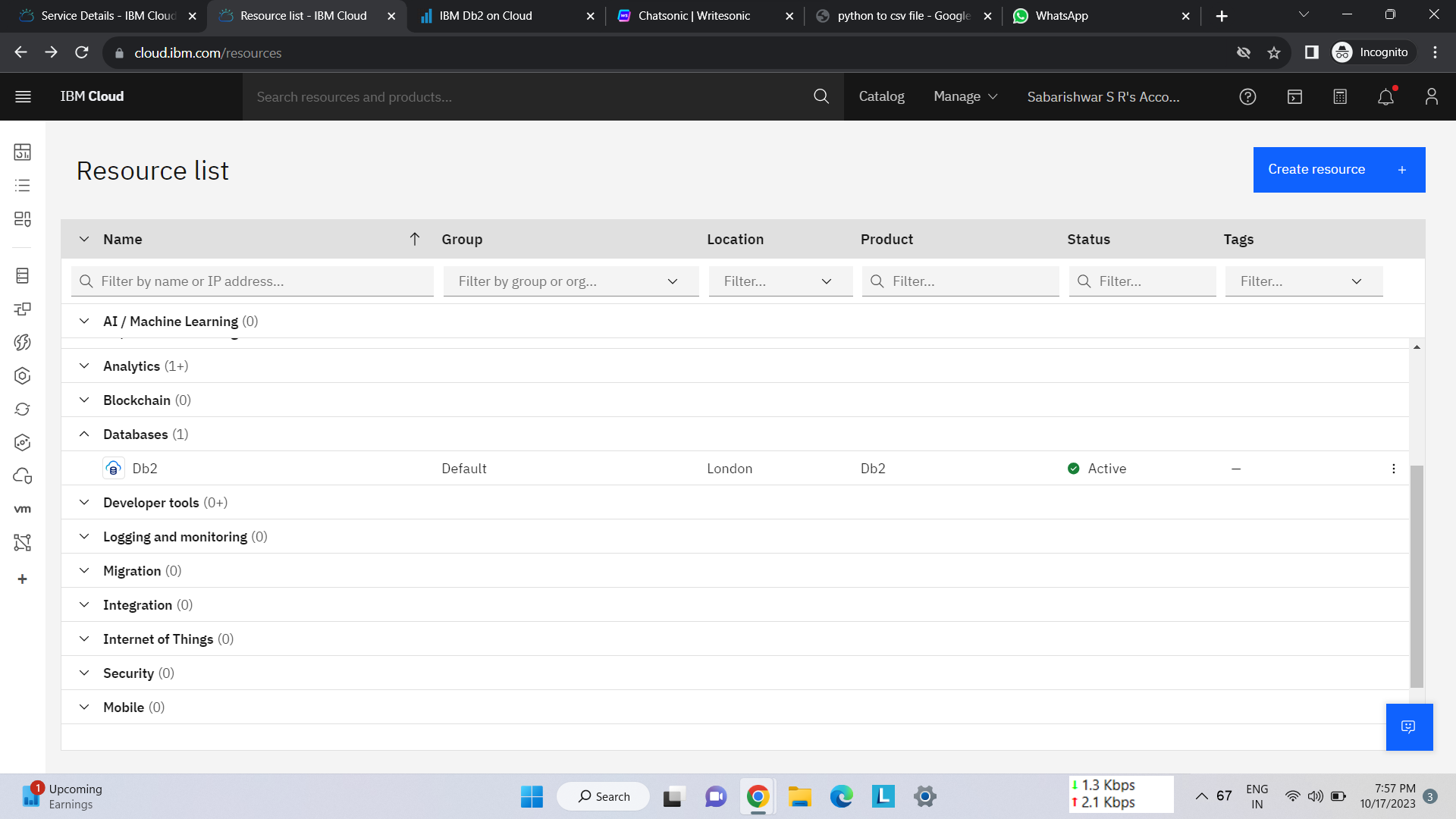
**Introduction:**

DB2 is a robust and reliable relational database management system (RDBMS) developed by IBM. IBM Cloud provides a platform for hosting DB2 databases, allowing users to store and manage their data securely. This document will guide you through the process of creating a DB2 database on IBM Cloud.

**Step 1:** Sign up for an IBM Cloud Account To get started, visit the IBM Cloud website and sign up for an account. Follow the registration process and provide the necessary information to create your account.

**Step 2:** Access the IBM Cloud Dashboard Once you have successfully signed up and logged in to your IBM Cloud account, you will be directed to the IBM Cloud Dashboard. This is the central hub for managing your cloud resources.

**Step 3:**  Provision a DB2 Instance In the IBM Cloud Dashboard, click on "Create Resource" to provision a new resource. Search for "DB2" in the catalog and select the DB2 service that suits your requirements.

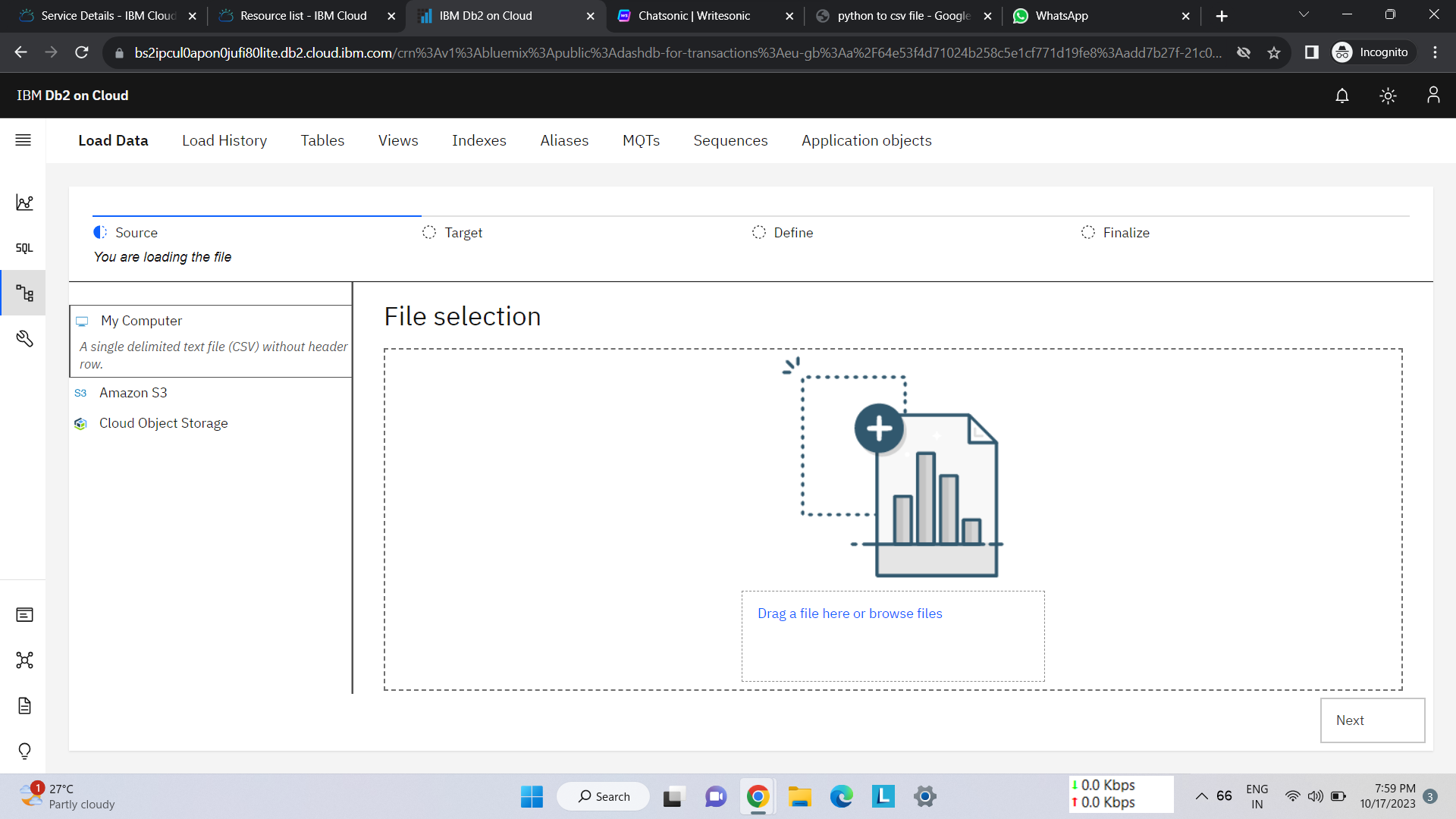


**Step 4:** Configure the DB2 Instance After selecting the DB2 service, you will be prompted to configure the instance. Choose the desired region, resource group, and plan for your DB2 database. You can select the appropriate plan based on your storage and performance needs.

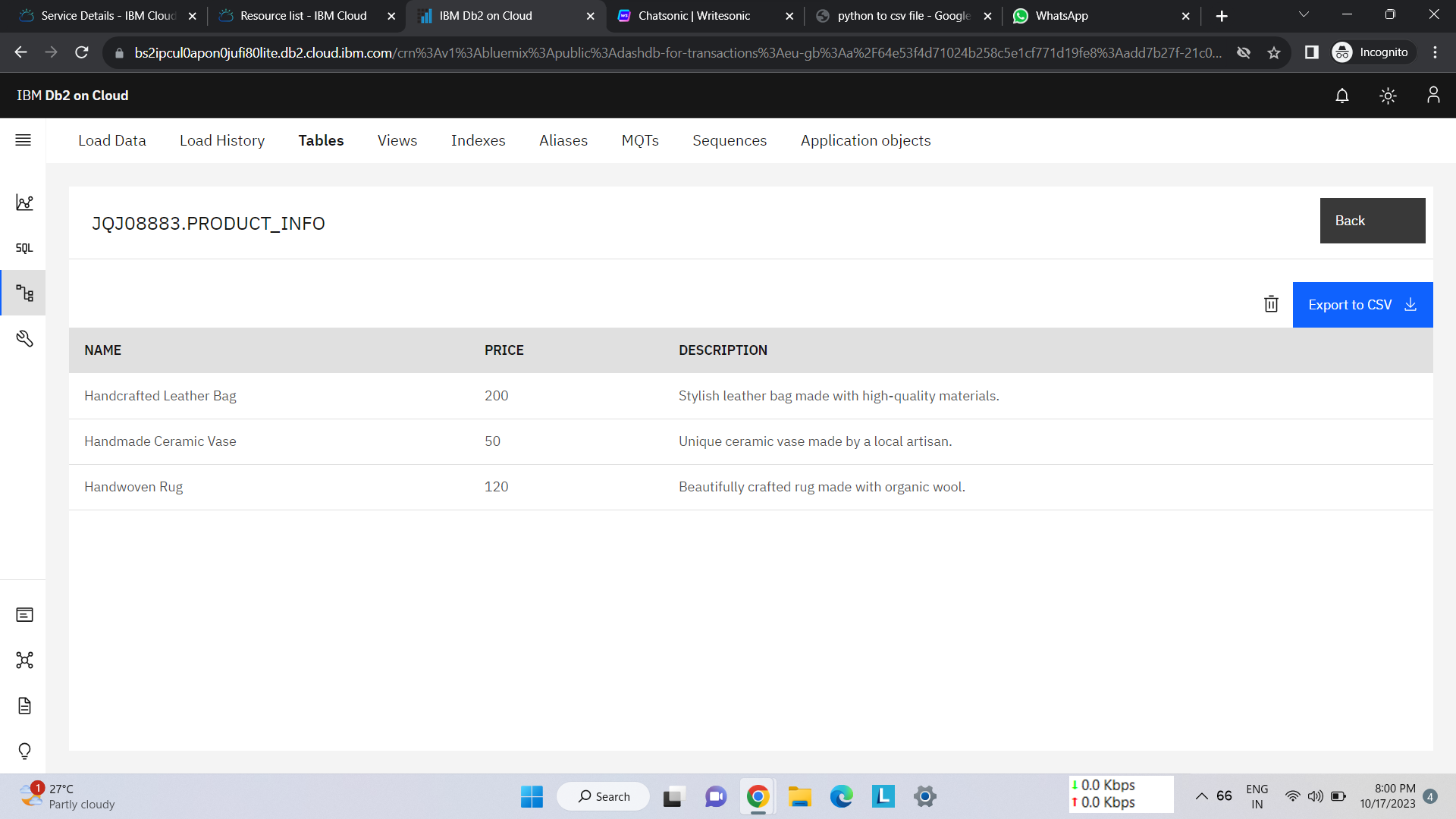
**Step 5:** Set Access and Security Options Next, configure the access and security options for your DB2 database. Set a unique name for your database, choose the authentication method (such as username/password or SSH key), and define access controls to restrict access to authorized users.

**Step 6:** Review and Create the DB2 Database Review the configuration details you have provided and ensure they are accurate. Once you are satisfied, click on "Create" to create your DB2 database on IBM Cloud.

**Step 7:** Connect to the DB2 Database After the database is successfully created, you can connect to it using various methods. IBM Cloud provides a web-based console, command-line tools, and APIs for managing and accessing your DB2 database. Choose the method that suits your preference and connect to your DB2 database.



**Step 8:** Create Tables and Store Data Once connected to your DB2 database, you can start creating tables and storing data. Use SQL statements or graphical tools to define the structure of your tables and insert data into them. You can also import existing data into your DB2 database as CSV Files if needed. We Now now Importing CSV file.



**Step 9:** Monitor and Optimize Performance IBM Cloud provides monitoring and performance optimization tools for your DB2 database. Monitor resource utilization, query performance, and database health to identify and resolve any performance issues. Optimize your database configuration and queries to ensure efficient data storage and retrieval.

**Conclusion:**

Creating a DB2 database on IBM Cloud is a straightforward process that allows you to store and manage your data securely. By following the steps outlined in this document, you can provision a DB2 instance, configure access and security options, and start storing your data in the cloud. Leverage the power of DB2 on IBM Cloud to build robust and scalable data solutions for your business needs

**Creating an E-commerce Application Using Flask :**

**Flask Application :**

A Flask application is a lightweight and versatile web framework for Python. It is known for its simplicity and minimalistic design, making it an excellent choice for building web applications and APIs. Flask allows developers to create web services quickly by providing essential tools and libraries, but it doesn't impose a strict structure, giving developers the freedom to design applications the way they see fit. Its microframework nature means you can add extensions as needed for features like authentication, databases, and more. Flask's flexibility, combined with its active community and extensive documentation, makes it a popular choice for web development, especially for smaller projects and prototyping.

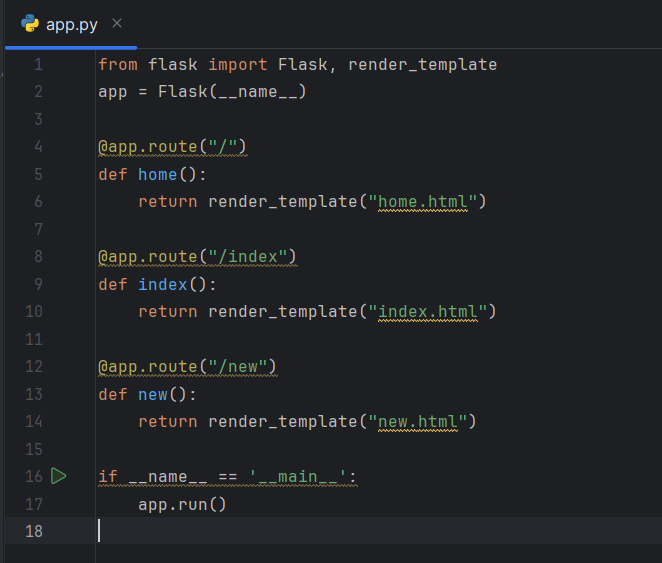
**Creating a Flask Application :**

Creating a Flask application is a straightforward process that empowers developers to build web applications with Python. It begins by installing Flask, a micro web framework, and setting up a virtual environment to manage dependencies. The heart of the application is the creation of routes, which map URLs to specific Python functions, enabling the handling of HTTP requests and responses. Flask's simplicity allows for the easy integration of HTML templates and dynamic content using Jinja2, while its lightweight nature enables the addition of custom functionalities through extensions. Finally, by running the Flask development server, one can see their web application come to life, ready to serve web pages and interact with users through a straightforward and efficient framework.

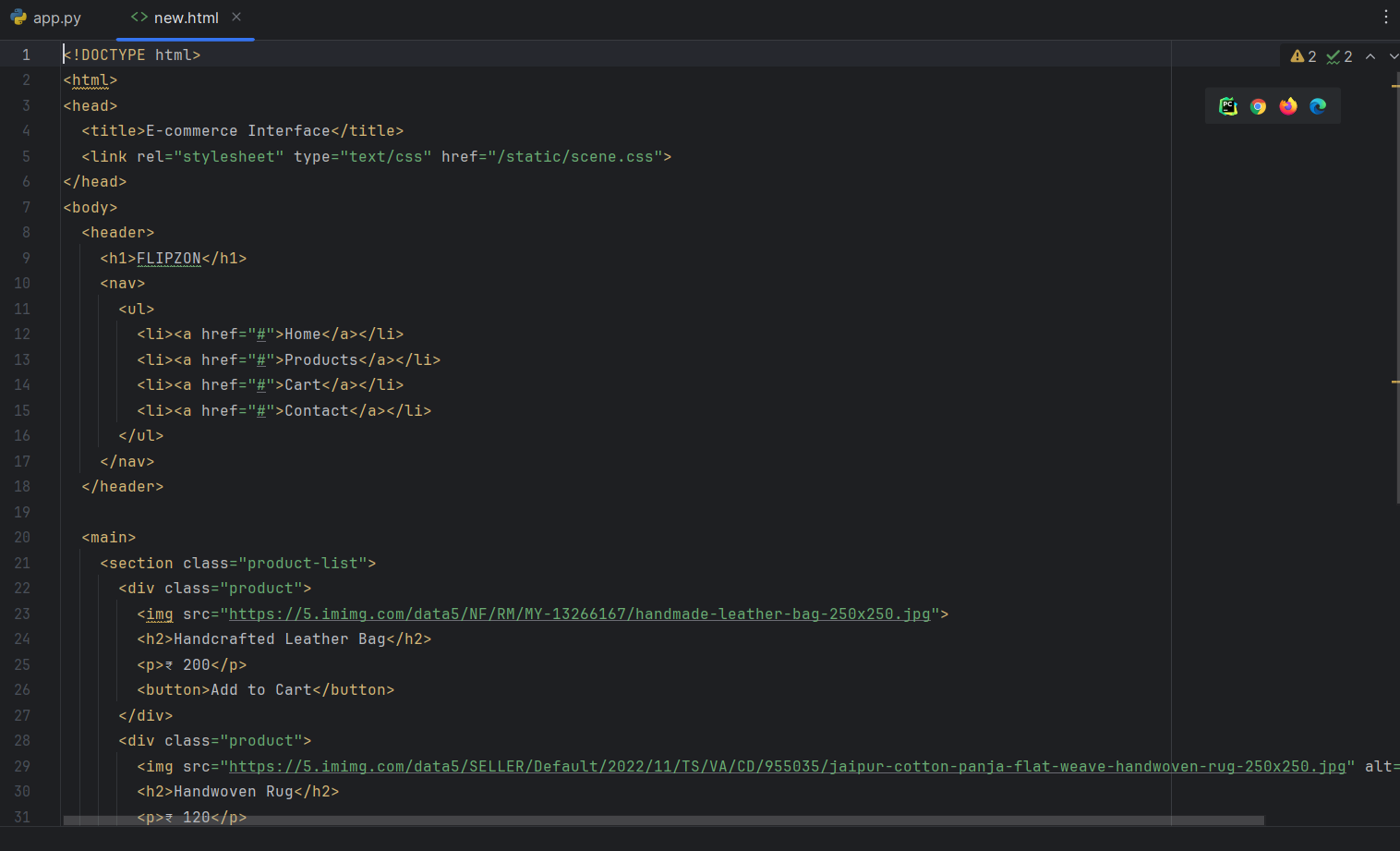
**Platform Layout of an E-Commerce Application :**

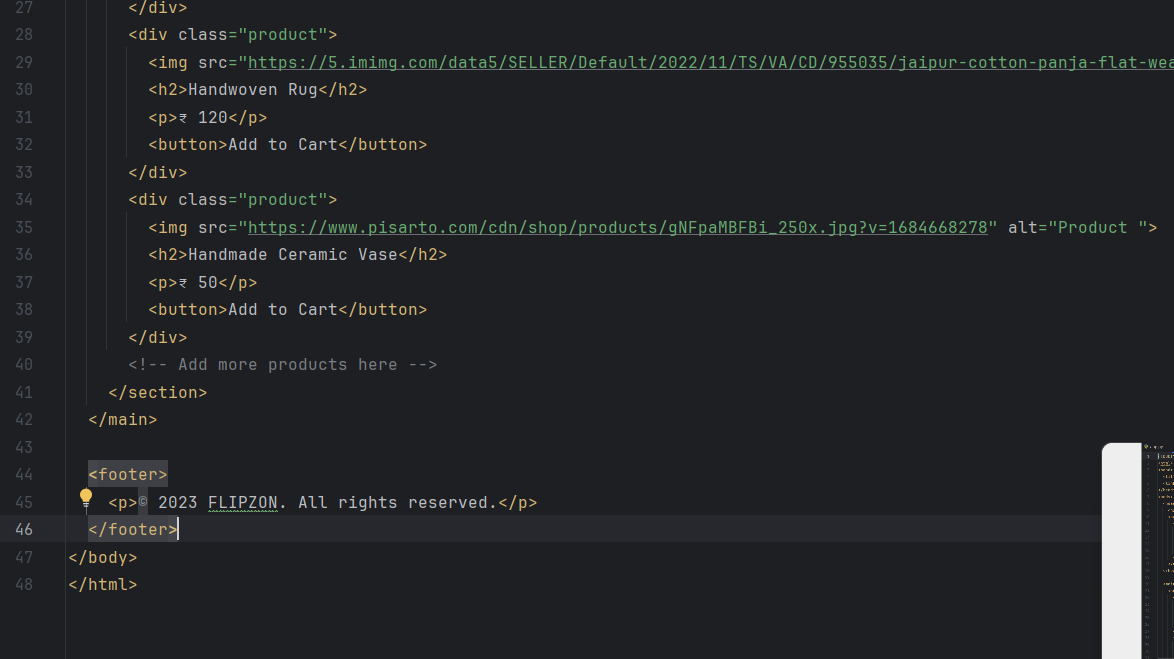
The platform layout of an e-commerce application plays a critical role in shaping the user experience. It typically consists of several key components. At the core is the homepage, which serves as the digital storefront, showcasing featured products and promotions. Product listings are organized into categories and displayed in a user-friendly manner, facilitating easy navigation. Search functionality allows users to find specific items quickly. Individual product pages provide detailed information, images, and customer reviews. The shopping cart and checkout process ensure a seamless transaction, while user profiles enable personalized experiences. Additionally, a customer support and contact section enhances communication. The layout's responsiveness across various devices and an intuitive user interface are vital to ensure a pleasant and efficient shopping experience for consumers.

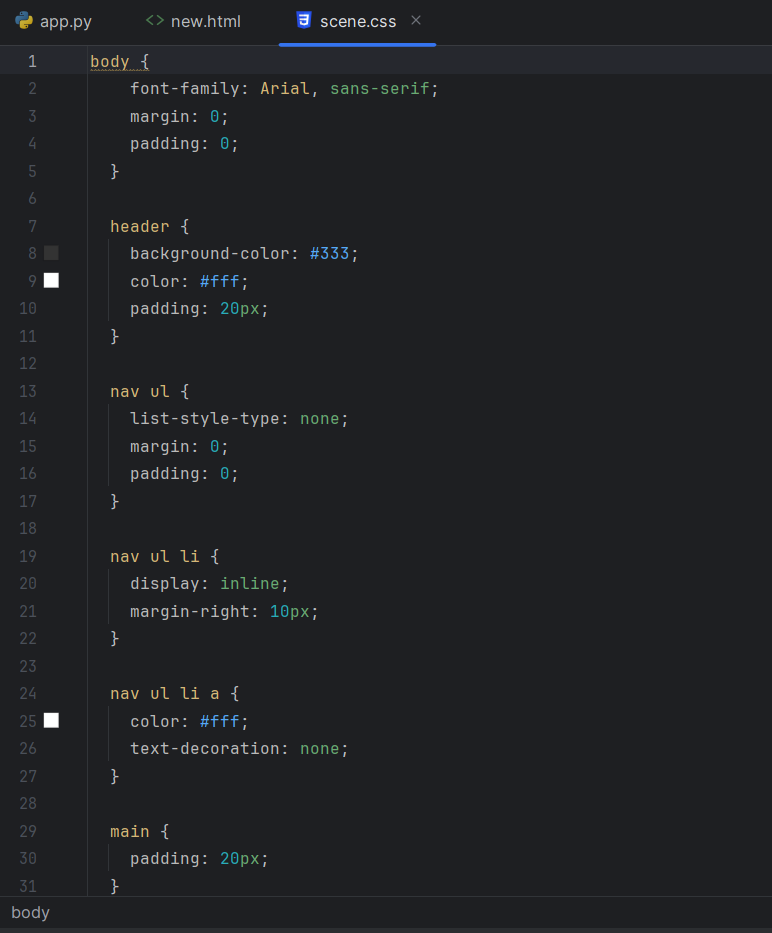
**Python Code :**



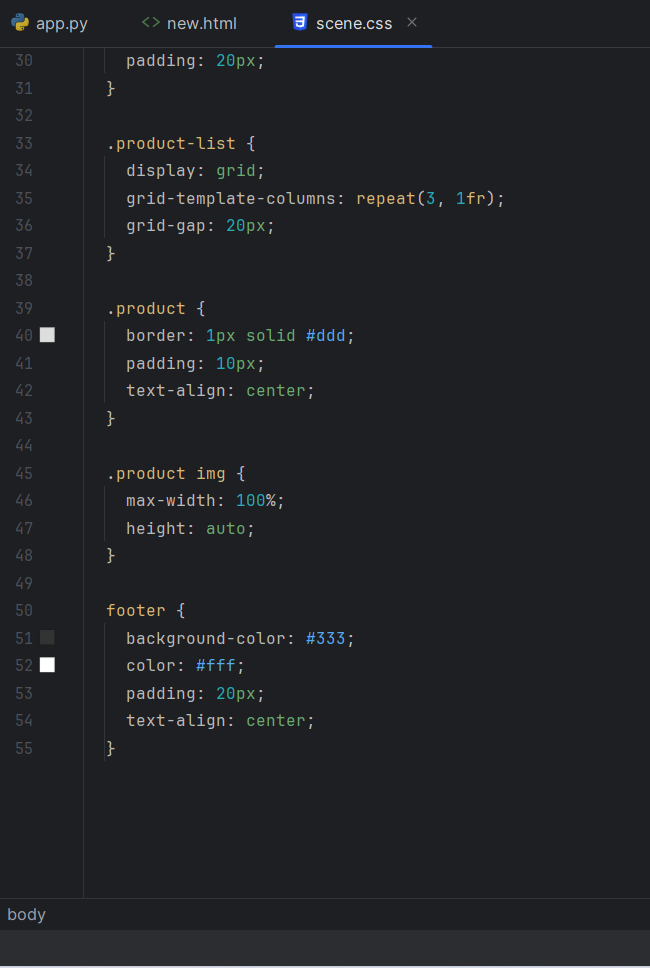
**HTML Code :**







**CSS Code :**



**Output :**