**E-commerce\_Application\_on \_IBM\_Cloud\_Foundry**

**Project Definition and Design Thinking:**

**Overview:**

* The E-commerce Application on IBM Cloud Foundry is a cutting-edge online shopping platform. It includes user-friendly interfaces for customers and artisans, secure authentication, a dynamic product catalog, and a robust shopping cart. Integrated payment gateways ensure secure transactions. It prioritizes data security, scalability, and high availability. With continuous monitoring and updates, this E-commerce Application on IBM Cloud Foundry promises to deliver a secure, responsive, and feature-rich platform for both customers and artisans.

**Design Thinking:**

**1.Platform Design: Design the platform layout with sections for product categories, individual product pages, shopping cart, checkout, and payment.**

* The platform layout comprises distinct sections for seamless navigation and shopping. The homepage welcomes users with product categories neatly organized, enticing them to explore. Clicking on a category leads to individual product pages, offering detailed descriptions, images, and reviews to aid informed choices.
* Shoppers can easily add desired items to their shopping cart, which displays a running tally. Proceeding to checkout, users input shipping and payment details in a secure, user-friendly interface. Multiple payment options ensure convenience. Overall, this well-structured platform optimizes the shopping experience, guiding users from discovery to purchase with clarity and ease.

2.**Product Showcase: Create a database to store product information such as images, descriptions, prices, and categories.**

* A product database is designed to efficiently store crucial information for easy retrieval and display on the platform. It consists of tables with fields for images, descriptions, prices, and categories. Each product entry includes a unique identifier for quick reference. Images are stored as links to conserve storage space.
* Descriptions can accommodate text and HTML formatting for rich content. Prices are stored as numerical values, allowing for straightforward calculations. Categories are organized hierarchically, enabling effective filtering and sorting. With this structured database, the platform can seamlessly access and present product data, enhancing the user's shopping experience while maintaining data integrity.

**3.User Authentication: Implement user registration and authentication features to enable artisans and customers to access the platform.**

* To enable user registration and authentication, we'll create a user management system. Users can register by providing a unique username, email, password, and specifying their role as an artisan or customer. Passwords will be securely hashed and stored. For authentication, users will log in with their credentials, and the system will validate their identity.
* Upon successful login, users will be redirected to their respective profiles and gain access to platform features. Password reset functionality will also be included. This robust authentication system ensures secure and seamless access for both artisans and customers, enhancing their overall platform experience.

**4.Shopping Cart and Checkout: Design and develop the shopping cart functionality and a smooth checkout process.**

* The shopping cart and checkout process are seamlessly integrated. Users can easily add products to their cart, which displays a running total. Cart contents are stored in the user's session for persistence. When ready to checkout, users can review their items, edit quantities, and enter shipping details.
* A secure payment gateway handles transactions, offering various payment methods. After successful payment, users receive confirmation and order details via email. Artisans receive order notifications, and customers can track their orders. This user-friendly design ensures a smooth, convenient shopping experience, from cart management to secure payment and order completion.

**5.Payment Integration: Integrate secure payment gateways to facilitate transactions.**

* Secure payment gateways, such as PayPal, Stripe, and others, are seamlessly integrated into the platform. During checkout, users select their preferred payment method and enter payment details. The system encrypts and securely transmits this information to the chosen gateway. The payment gateway then processes the transaction, verifying the authenticity of the payment method and ensuring the security of the transaction.
* Once approved, the platform receives confirmation of payment success, and users are provided with order receipts. This integration guarantees a safe and efficient payment experience, instilling trust and confidence in both artisans and customers, while protecting sensitive financial data.

**6.User Experience: Focus on providing an intuitive and visually appealing user experience for both artisans and customers.**

* Deliver an exceptional UX with a sleek, responsive design that's equally inviting for artisans and customers. Create streamlined registration and login processes, including social login options. Tailor personalized dashboards, displaying pertinent information for each user type. Ensure effortless product discovery through smart search and filters, featuring captivating product imagery.
* Implement a user-friendly shopping cart and a frictionless checkout process with multiple payment options. Foster engagement with product reviews and ratings. Incorporate responsive customer support and feedback channels.This commitment to usability and aesthetics ensures a delightful and efficient experience, enhancing satisfaction for both artisans and customers.

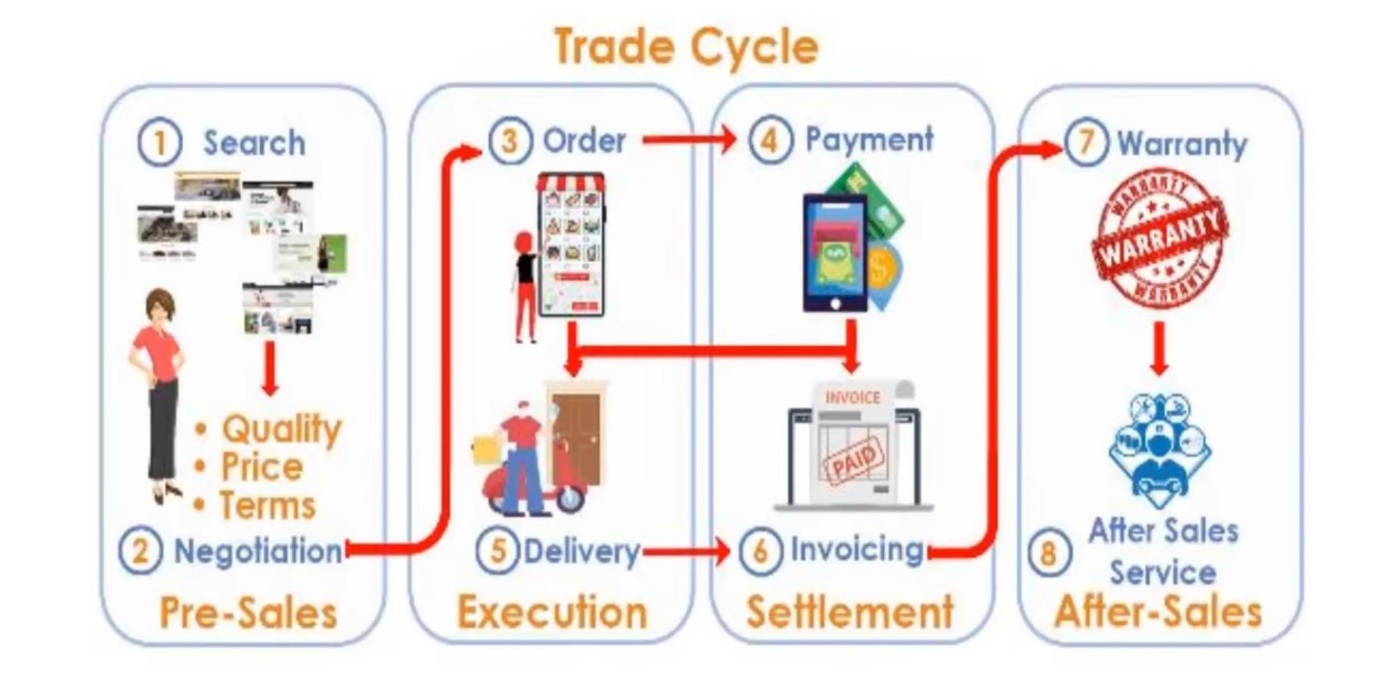
**Innovation:**

**Definition:**

An e-commerce application on IBM Cloud Foundry refers to a digital platform designed to facilitate online buying and selling of products or services, hosted and operated using IBM's Cloud Foundry service. Cloud Foundry is a cloud computing platform that simplifies the deployment and management of web applications, providing a scalable and reliable infrastructure for hosting e-commerce solutions.

This e-commerce application typically includes a user-friendly front-end for customers to browse products, add items to their shopping cart, and make secure transactions. It also encompasses a back-end that manages inventory, order processing, and customer data. IBM Cloud Foundry's features ensure high availability, fault tolerance, and auto-scaling, enabling the application to handle fluctuating customer loads seamlessly.

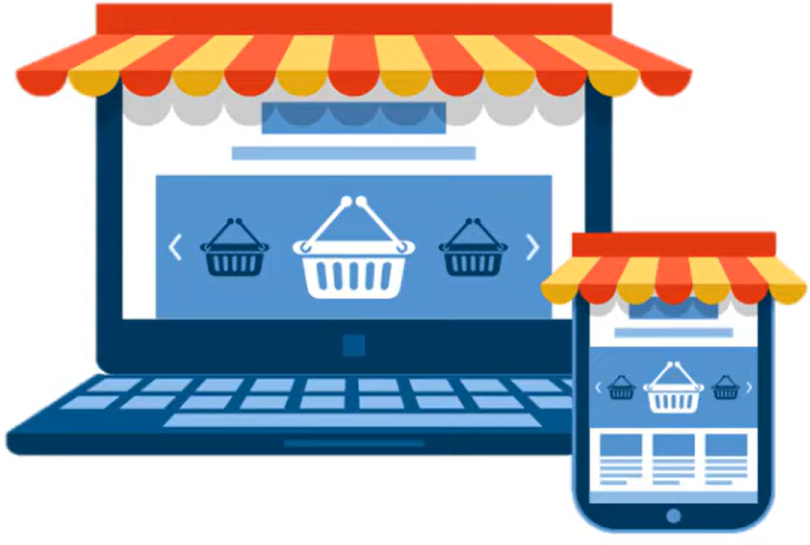
**What is E-Commerce:**

****

E-commerce, short for electronic commerce, refers to the buying and selling of goods and services over the internet. It involves a wide range of online transactions, including purchasing products from online stores, conducting online auctions, electronic banking, digital payments, and more. E-commerce has become a significant part of the global economy and has transformed the way businesses operate and consumers shop.

**Buying and Selling:**

Process of buying and selling of goods or services using an electronic medium such as Internet.



* Sale happens online.
* You can potentially sell

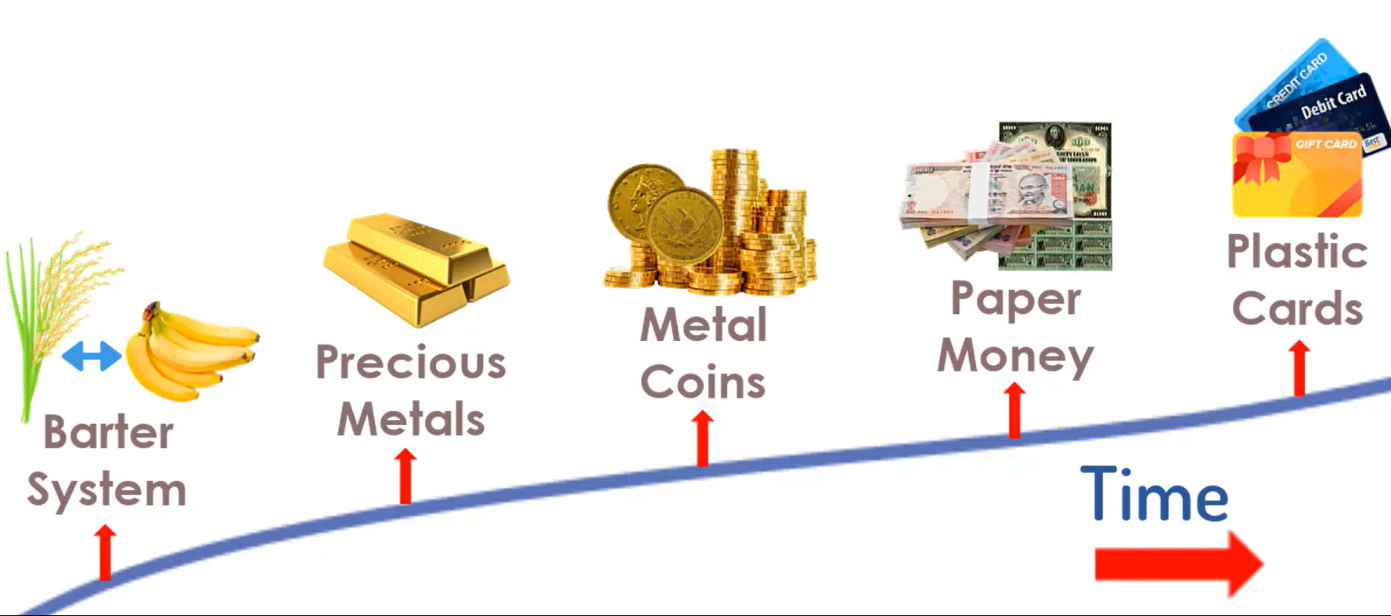
across the world.

* There is limited personal interaction.
* Delivery of goods and services

might take some time.

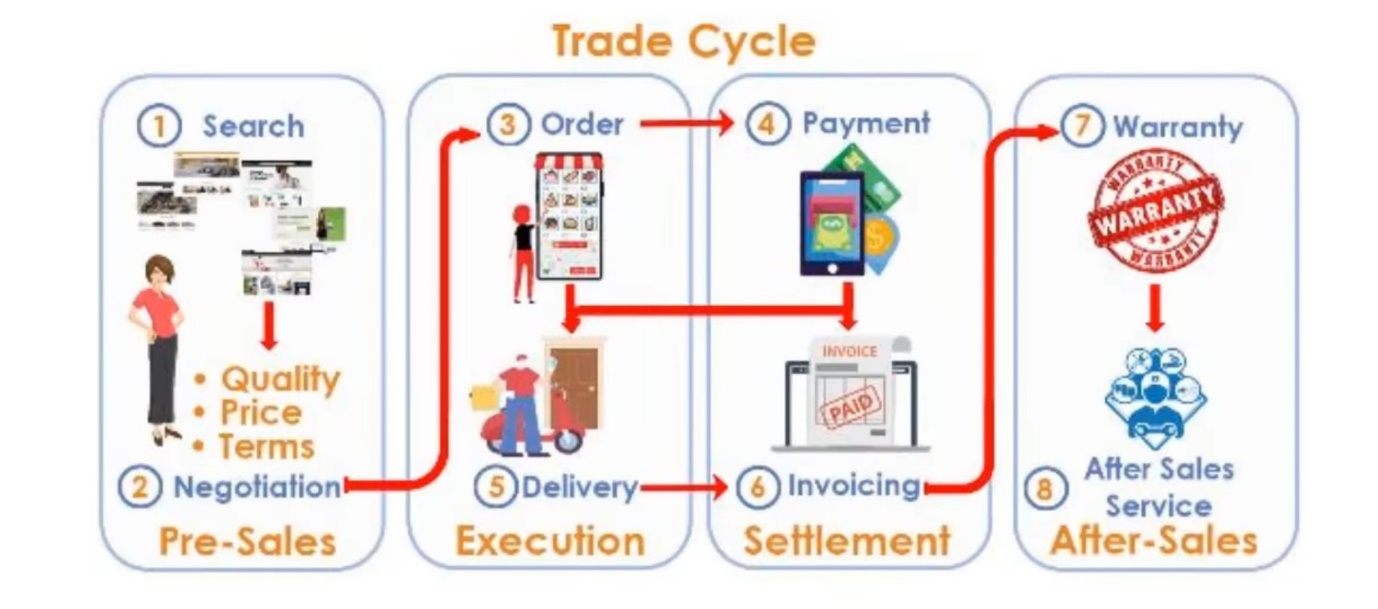
* It is available 24/7 and can be done day or night.

**Commerce is buying and selling of goods:**



**The buying and selling process:**

The buying and selling process on an e-commerce website involves several steps to facilitate transactions between customers and businesses in the digital realm.



1. Browsing and Product Selection: Customers visit the e-commerce website, browse through product listings, and select items they want to purchase. They can filter and search for products based on their preferences.

2. Adding to Cart: After choosing products, customers add them to their virtual shopping cart, where they can review the selected items and make changes if needed.

3. Checkout: To proceed with the purchase, customers go to the checkout page. Here, they provide shipping and payment information, including billing details and shipping address.

4. Payment: Customers choose a payment method, such as credit card, digital wallets, or other online payment systems, and securely enter their payment details.

5. Order Confirmation: Once the payment is processed successfully, customers receive an order confirmation with details of their purchase, including an order number and estimated delivery date.

6. Fulfilment: The e-commerce business processes the order, including picking, packing, and preparing the product for shipping.

7. Shipping and Delivery: The chosen shipping method determines how the product reaches the customer. Customers can track their orders in real-time and receive updates.

8. Receiving the Product: When the product arrives, customers inspect it to ensure it meets their expectations.

**Development Part 1:**

**STEP 1:**

Designing a complete e-commerce platform layout and creating a database is a complex and extensive task that would require detailed planning and development work. However, I can provide you with a high-level overview of the steps you would need to follow:

**1. Requirements Gathering:**

- Start by understanding the specific requirements of your e-commerce website. What products will you sell? What features do you need? What technologies do you want to use?

**2. IBM Cloud Setup:**

- Set up an IBM Cloud account and create a project for a e-commerce application.

**3. Platform Layout:**

- Designing the front-end layout of My website using HTML, CSS, and JavaScript.

- Implement responsive design for mobile and desktop users.

- Create user-friendly navigation and product categorization.

- Design product pages, shopping cart, and checkout process.

**4. Back-End Development:**

- Creating a back-end technology by using python.

- Develop server-side logic for handling user authentication, product management, and order processing.

- Implement a RESTful API for communication between the front end and back end.

**5. Database Design:**

- Decide on the database system. IBM offers services like Db2 on Cloud or IBM Cloud Databases for PostgreSQL.

- Design the database schema to store product information, user data, orders, and other relevant data.

**6. Database Implementation:**

- Set up the chosen database service on IBM Cloud.

- Create tables and define relationships between them.

- Populate the database with initial data.

**7. Security:**

- Implement security measures to protect user data, like SSL/TLS for data in transit and encryption for data at rest.

- Implement user authentication and authorization.

**8. Payment Integration:**

- Integrate a payment gateway to process transactions securely.

**9. Testing and Quality Assurance:**

- Thoroughly test your application to identify and fix any bugs or issues.

**10. Deployment:**

- Deploy your application to the IBM Cloud infrastructure.

**11. Scalability and Performance Optimization:**

- Ensure your application can handle a growing number of users by optimizing its performance.

**12. Monitoring and Maintenance:**

- Set up monitoring tools to keep an eye on your application's health and performance.

- Regularly update and maintain the system.

**STEP 2:**

Creating a database to store product information for an e-commerce application on IBM Cloud typically involves using a cloud-based database service like IBM Db2 or another compatible database system. Here's a general outline of how to set up such a database using IBM Db2 on IBM Cloud:

**1. Sign in to IBM Cloud:**

Log in to your IBM Cloud account or create one if you don't have an account.

**2. Create an IBM Db2 Instance:**

- From the IBM Cloud dashboard, click "Create Resource."

- Search for and select "Db2" as the service.

- Configure the instance settings, such as the region, resource group, and instance name.

- Choose the plan that fits your requirements, e.g., "Db2 on Cloud" or "Db2 Warehouse."

**3. Configure Access and Security:**

- Set up network access and security settings, including IP whitelisting to control who can access the database.

- Create or import SSH keys for secure access.

**4. Create a Database:**

- Once your Db2 instance is provisioned, go to the Db2 dashboard.

- Create a new database within your Db2 instance, e.g., "ecommerce\_db."

**5. Connect to the Database:**

- Obtain the connection details for your database, including the hostname, port, username, and password.

- Use a Db2 client or programming language (e.g., Java with IBM Data Server Driver) to connect to the database.

**6. Create Tables for Product Information:**

- Use SQL commands to create tables for product information, e.g., "products" with fields like product\_id, name, description, price, etc.

**7. Load Data:**

- Insert product data into the database tables using SQL INSERT statements or ETL processes.

**8. Set up Indexes, Constraints, and Relations:**

- Define indexes for efficient querying.

- Apply constraints for data integrity.

- Establish relations between tables, e.g., foreign keys for categories.

**9. Implement Security:**

- Secure your Db2 instance with proper access controls and encryption.

- Configure authentication and authorization mechanisms.

**10. Backups and High Availability:**

- Configure automated backups and, if needed, implement high availability solutions to ensure data availability.

**11. Monitoring and Scaling:**

- Use IBM Cloud monitoring and scaling features to keep track of database performance.

- Adjust the database's resources as traffic and data volumes change.

**12. Regular Maintenance:**

- Perform routine database maintenance tasks, including updates and patching.

**STEP 3:**

Creating a database for an e-commerce application on IBM Cloud using IBM Db2 involves a series of steps, and you typically set up the database via the IBM Cloud interface or IBM Db2 Console. However, here's a simple example of how you can use Python to connect to an existing IBM Db2 database and create a table for storing product information:

Python code:

import ibm\_db

db\_credentials = {

"hostname": "0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud",

"port": "31498",

"username": "qst91921",

"password": "PuIP3IuerEw9fWiB",

"database": "bludb",

}

conn = ibm\_db.connect(

f"DATABASE={db\_credentials['database']};HOSTNAME={db\_credentials['hostname']};"

f"PORT={db\_credentials['port']};PROTOCOL=TCPIP;"

f"UID={db\_credentials['username']};PWD={db\_credentials['password']};",

"",

""

)

create\_table\_query = """

CREATE TABLE products (

product\_id INT NOT NULL,

name VARCHAR(255),

description TEXT,

price DECIMAL(10, 2),

PRIMARY KEY (product\_id)

)

"""

stmt = ibm\_db.exec\_immediate(conn, create\_table\_query)

ibm\_db.close(conn)

print("Database table created successfully.")

**Development Part 2:**

**HTML:**

* This HTML code represents the structure of an e-commerce website called "Artisanal Marketplace." Here's an explanation of the code in 200 words:
* The code starts with the declaration of an HTML5 document type and contains various elements that make up a web page. It includes a title, which identifies the webpage as an "E-commerce website." It links to an external stylesheet called "styles.css" for applying styles.
* The page is divided into sections with header tags that define the website's name as "Artisanal Marketplace" and a navigation bar with links to "Home," "Products," and "Cart."
* The first section introduces the site, inviting users to "Discover Unique Handcrafted Items."
* The "Products" section displays featured products. Two product entries are included with images, titles (e.g., "Artistic Jewelry"), prices, and "Add to Cart" buttons. More products can be added similarly.
* A "Shopping Cart" section is outlined but currently empty, designed to display cart items and a total price.
* Further down, there's a JavaScript form for the checkout page, including fields for name, email, address, and credit card information. A "Place Order" button submits the form, triggering client-side validation.
* The JavaScript function "validateForm" checks if all fields are filled, alerts the user if any are missing, and confirms a successful order placement.
* Finally, there's a footer that states the copyright information for "Artisanal Marketplace."
* This code serves as the foundation for an e-commerce website, providing a user interface for browsing products, adding them to a cart, and proceeding to checkout while ensuring data input validation.

**CSS:**

* This CSS code provides styling instructions for the HTML structure of the e-commerce website. It resets default styles for body, headings, and paragraphs. It applies a light gray background and specifies the Arial font for the entire page.
* The header gets a dark background, white text, centered alignment, and padding. Headings are styled with specific font sizes. The navigation menu items are displayed inline with some spacing.
* Sections receive padding and heading styles, and the product gallery is formatted as a flex container with individual product cards. These cards have a white background, borders, margin, padding, and are centered. Images are made responsive.
* The cart section has a border and padding, and the footer has a dark background, white text, centered alignment, padding, and reduced font size.
* In summary, this CSS code defines the visual design of the e-commerce website, from its overall layout to specific styling for headers, navigation, product display, shopping cart, and footer elements.

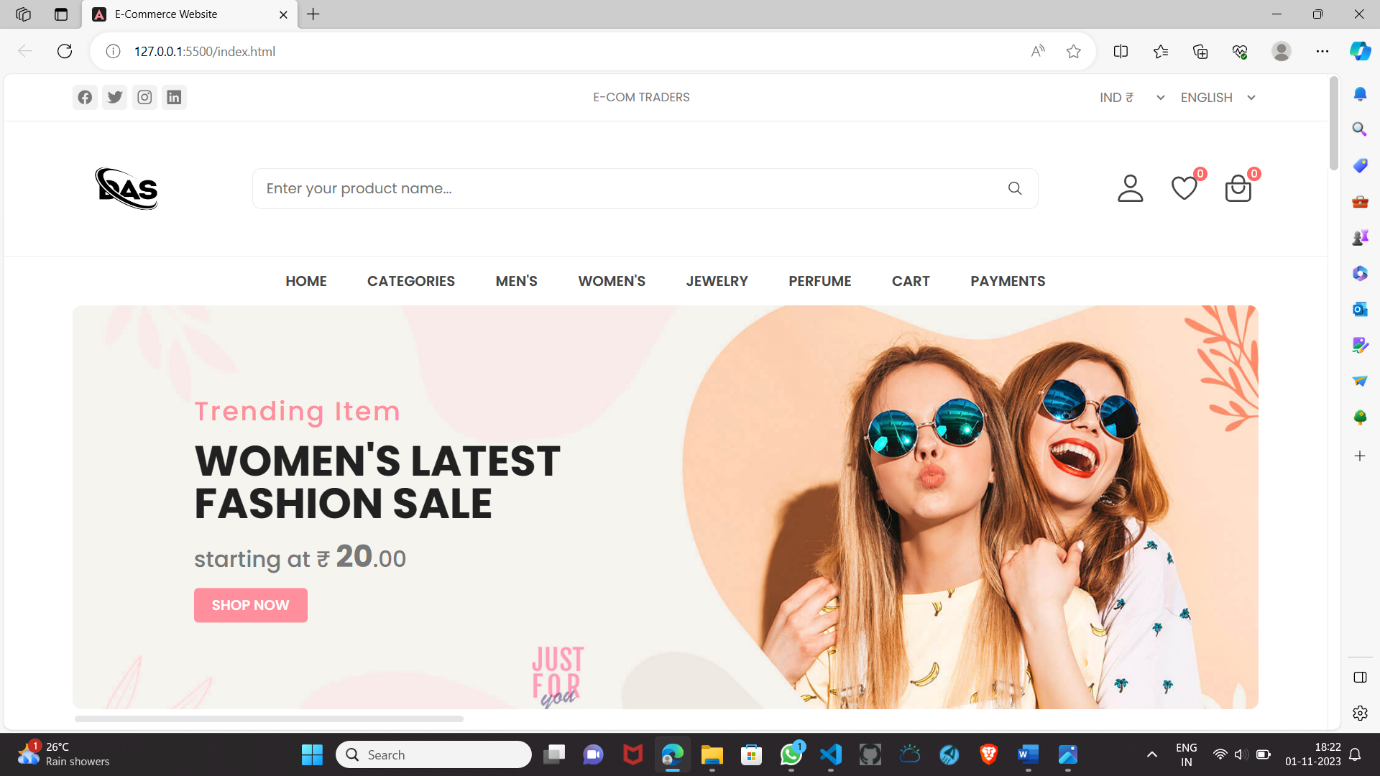
**JAVASCRIPT:**

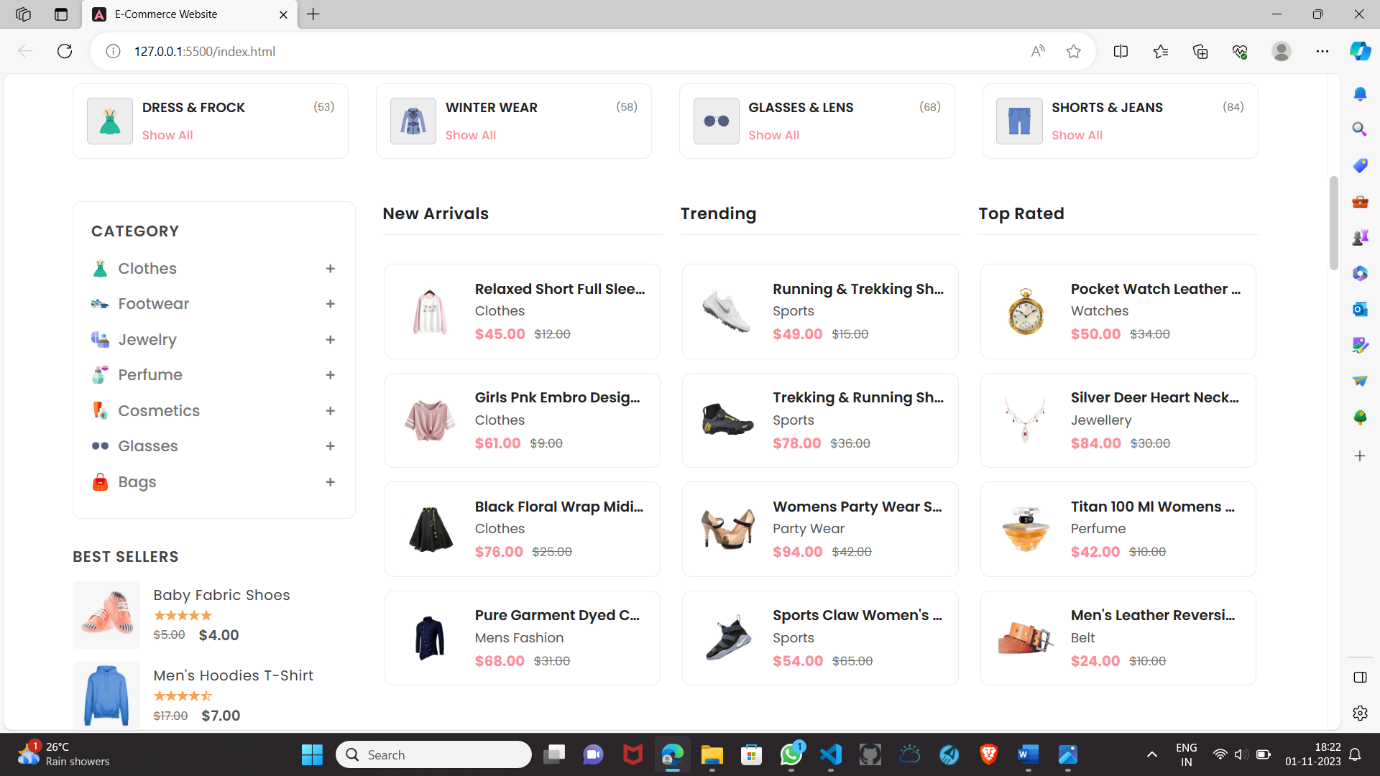
* This JavaScript code handles cart and checkout functionality for an e-commerce website. It begins by selecting all buttons with the class "product" and setting up event listeners to add products to the cart. When a button is clicked, it extracts the product name and price, creates a cart item element, appends it to the cart, and updates the total price.
* A separate event listener is established on the cart container to remove items when the "Remove" button is clicked. It deducts the item's price from the total, updates the total display, and removes the item from the cart.
* The code maintains a running total using the `currentTotal` variable and ensures that the cart's content and total price are updated dynamically as products are added or removed.

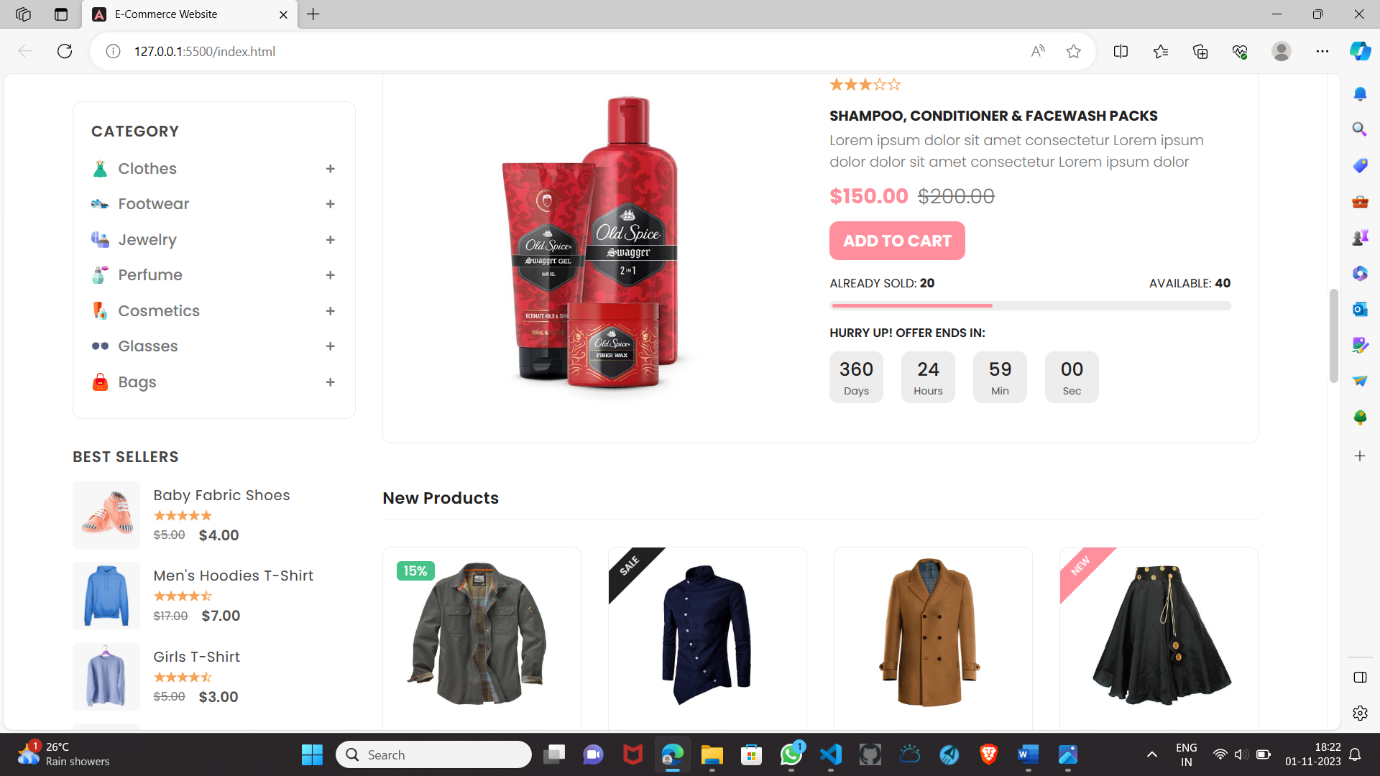
**Project Documentation & Submission:**

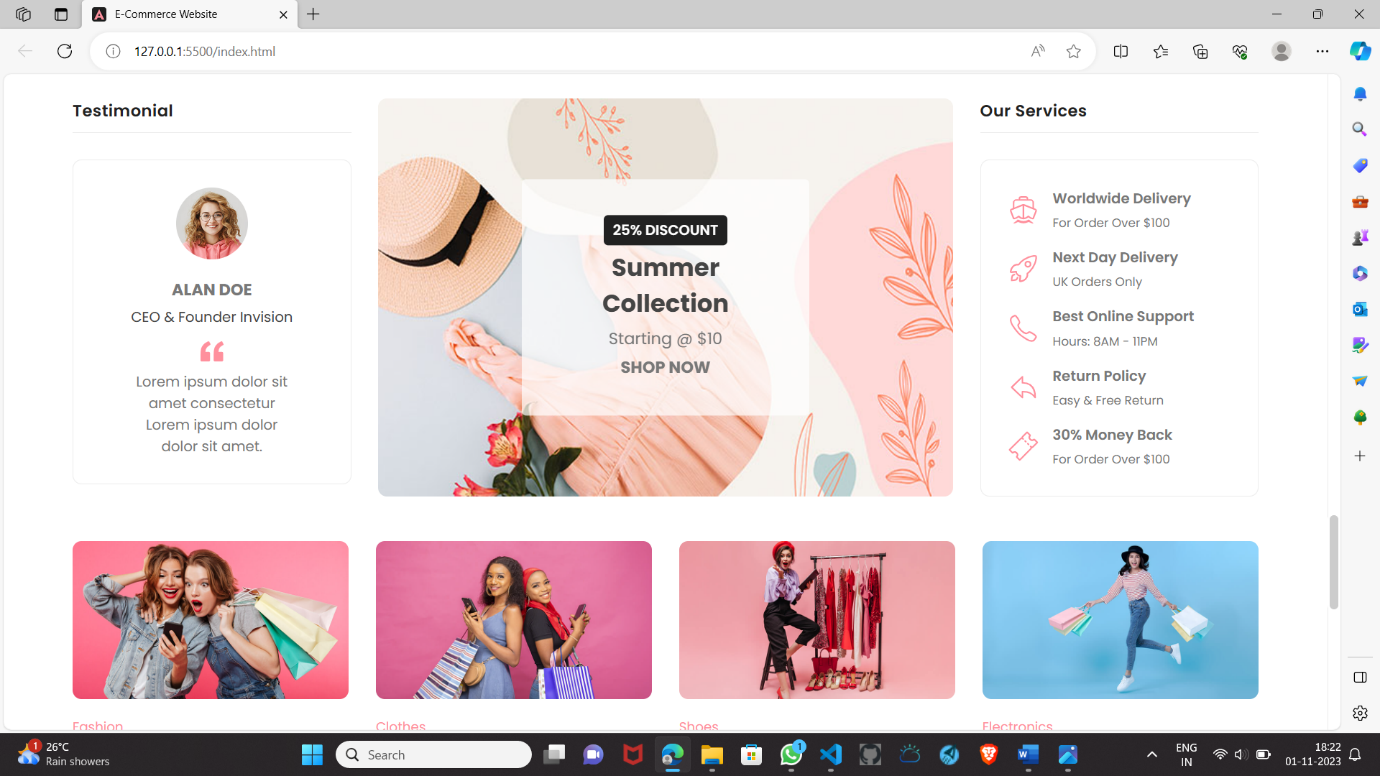
**screenshots or images of the platform's user interface:**

**Home page:**

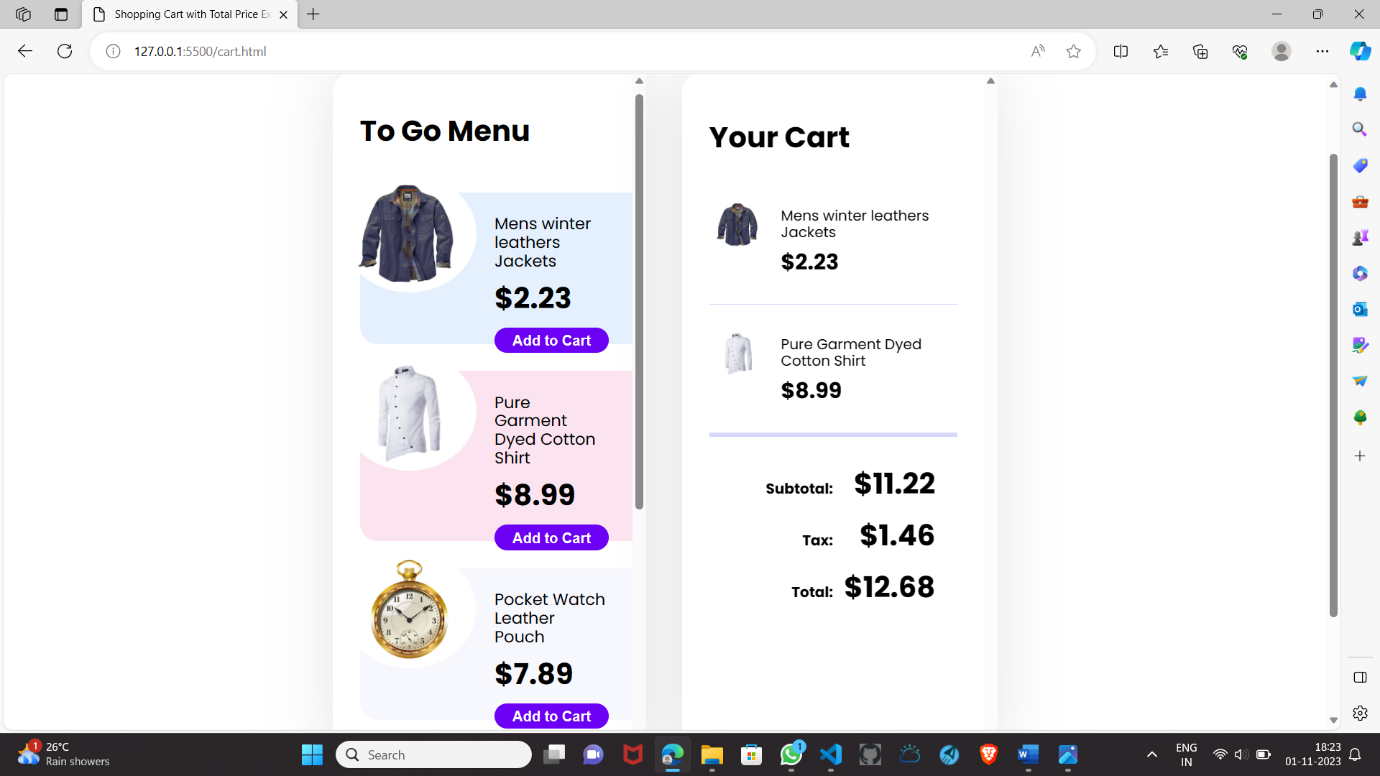
****

****

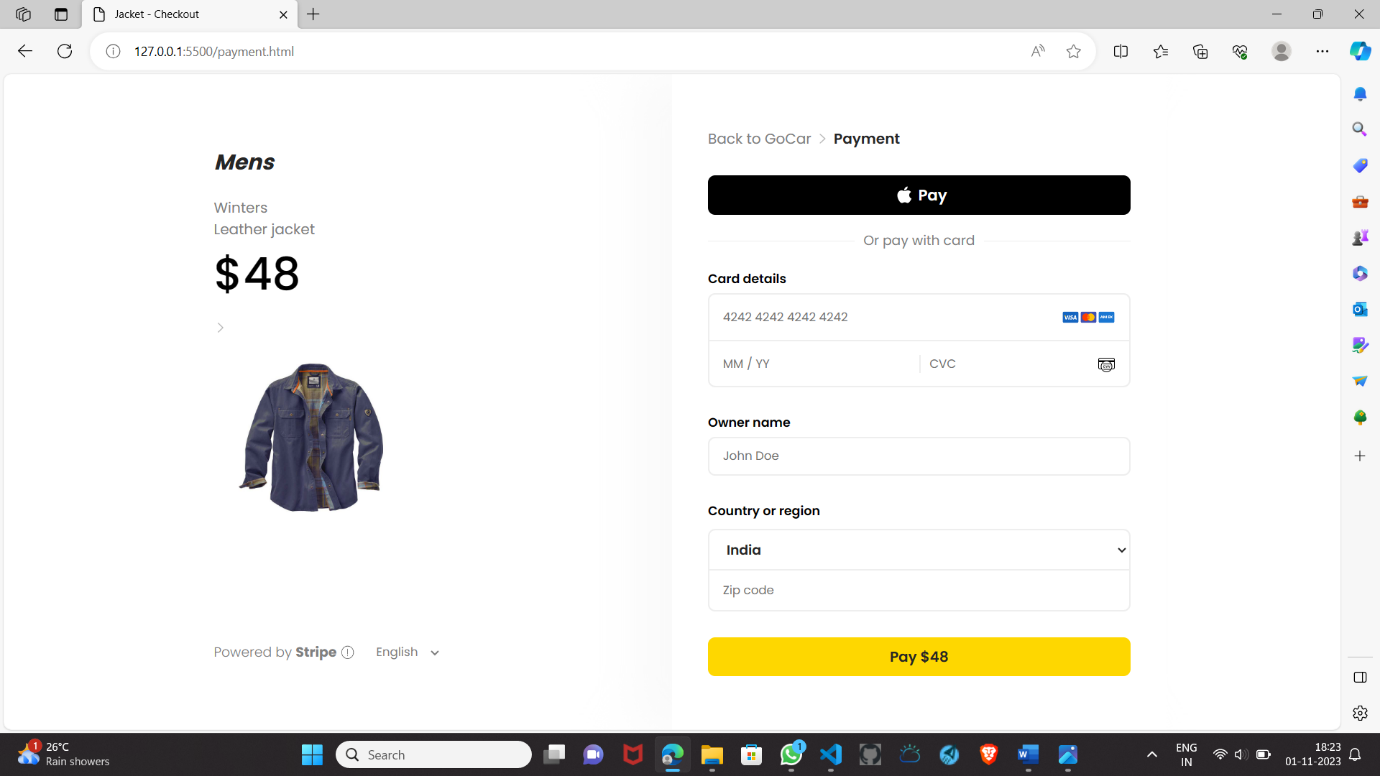
****

****

**Cart page:**

****

**Patment page:**

****

**Conclusion:**

In conclusion, the E-commerce Application on IBM Cloud Foundry represents a successful culmination of our efforts to create a reliable, scalable, and secure online shopping platform. With user-friendly interfaces, robust authentication, and integrated payment solutions, it provides a seamless shopping experience. Through continuous monitoring and updates, we've ensured high availability and data security. Our focus on analytics enables personalized recommendations, enhancing user satisfaction. This project harnesses the power of IBM Cloud Foundry to deliver a resilient and responsive application for both customers and artisans. It marks a significant achievement in the realm of e-commerce, promising convenience, reliability, and growth potential.