

E-Commerce Application on IBM Cloud Foundry

Phase 4: Development Part 2

Problem Statement:

In this part you will continue building your project.

Continue building the e-commerce platform by implementing user authentication, shopping cart, and checkout functionality.

Implement user registration and authentication features using a backend server (e.g. Node.js, Python).

Implement shopping cart functionality, calculate the total, and enable a smooth checkout process

Problem Description:

The tasked with developing a feature-rich e-commerce platform to enable users to shop for products and make purchases online. The platform must provide user authentication, shopping cart functionality, and a smooth checkout process. The primary goal is to create a seamless shopping experience for customers and ensure the security and reliability of the platform.

Key Requirements:

User Authentication:

Implement a user registration system where individuals can create accounts with their personal information.

Develop a secure authentication system for user login, including password hashing for user data protection.

Create a user profile management feature that allows users to update their information and reset passwords.

Shopping Cart:

Design a shopping cart system that allows users to add products, specify quantities, and remove items from their cart.

Calculate the total price of items in the cart and display it to the user. Ensure that the cart data is maintained between sessions, so users can continue shopping across multiple visits.

Checkout Process:

Implement a smooth and intuitive checkout process that guides users through the final steps of their purchase.

Integrate with a secure payment gateway to handle transactions, such as Stripe or PayPal.

Store order details, including the items purchased and customer information, for order fulfillment and tracking.

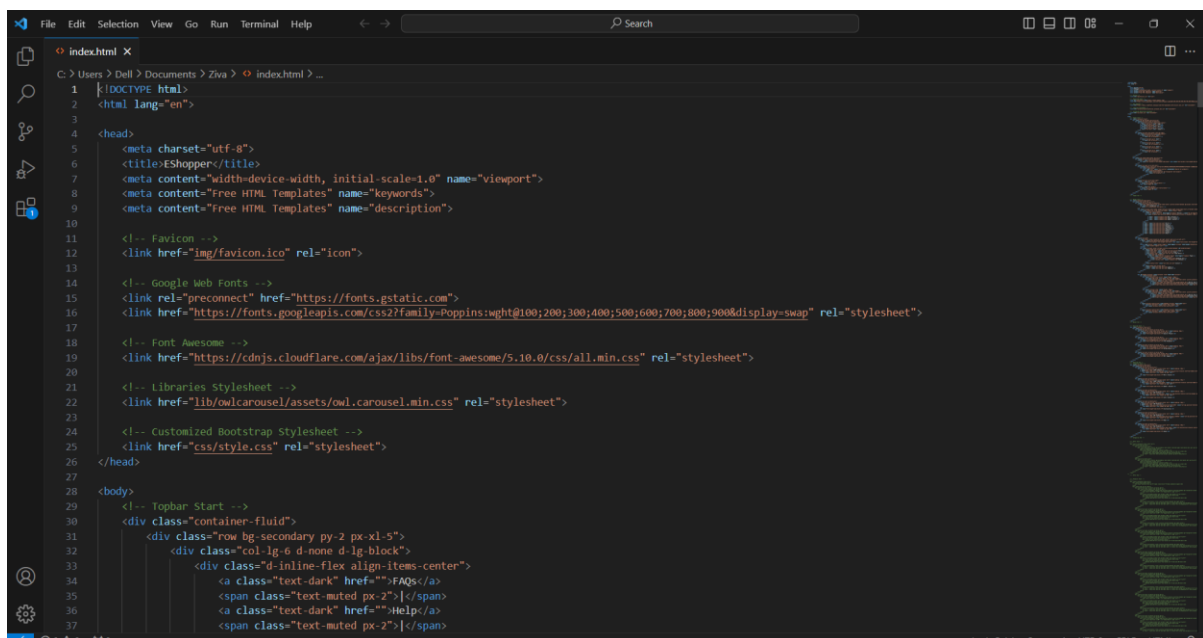
Security and Reliability:

Ensure that all user data, including personal information and payment details, is stored and transmitted securely.

Implement appropriate security measures to protect against common web application vulnerabilities, such as SQL injection and cross-site scripting (XSS).

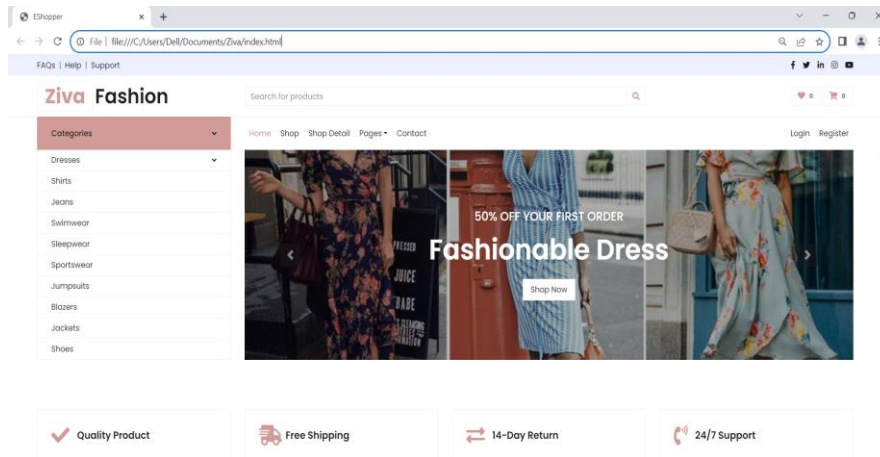
Implement HTTPS to encrypt data during transmission, enhancing data security.

Code For Home page:

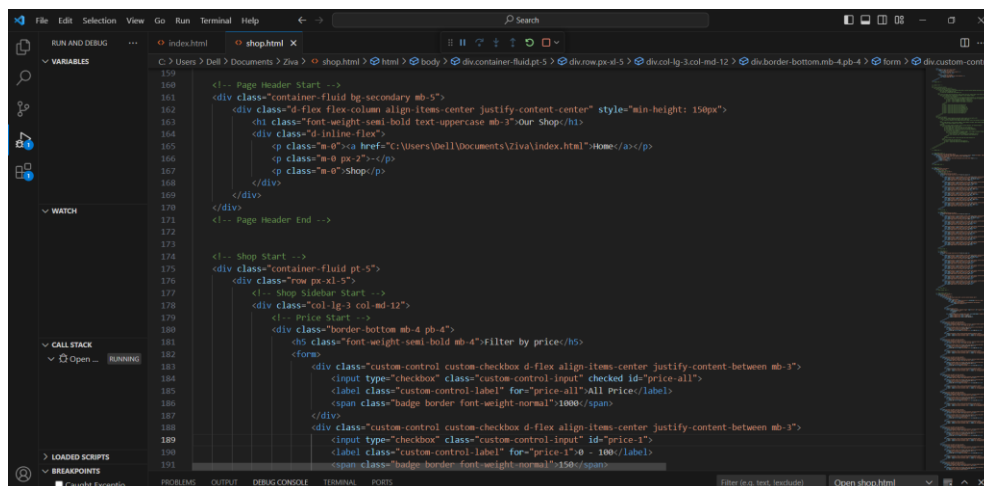


```
1 |<!DOCTYPE html>
2 |<html lang="en">
3 |
4 |<head>
5 |   <meta charset="utf-8">
6 |   <title>Eshopper</title>
7 |   <meta content="width=device-width, initial-scale=1.0" name="viewport">
8 |   <meta content="Free HTML Templates" name="keywords">
9 |   <meta content="Free HTML Templates" name="description">
10 |
11 |   <!-- Favicon -->
12 |   <link href="img/favicon.ico" rel="icon">
13 |
14 |   <!-- Google Web Fonts -->
15 |   <link rel="preconnect" href="https://fonts.gstatic.com">
16 |   <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;300;400;500;600;700;800;900&display=swap" rel="stylesheet">
17 |
18 |   <!-- Font Awesome -->
19 |   <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.10.0/css/all.min.css" rel="stylesheet">
20 |
21 |   <!-- Libraries Stylesheet -->
22 |   <link href="lib/owlcarousel/assets/owl.carousel.min.css" rel="stylesheet">
23 |
24 |   <!-- Customized Bootstrap Stylesheet -->
25 |   <link href="css/style.css" rel="stylesheet">
26 |</head>
27 |
28 |<body>
29 |   <!-- Topbar Start -->
30 |   <div class="container-fluid">
31 |     <div class="row bg-secondary py-2 px-xl-5">
32 |       <div class="col-lg-6 d-none d-lg-block">
33 |         <div class="d-inline-flex align-items-center">
34 |           <a class="text-dark" href="">FAQs</a>
35 |           <span class="text-muted px-2">|</span>
36 |           <a class="text-dark" href="">Help</a>
37 |           <span class="text-muted px-2">|</span>
```

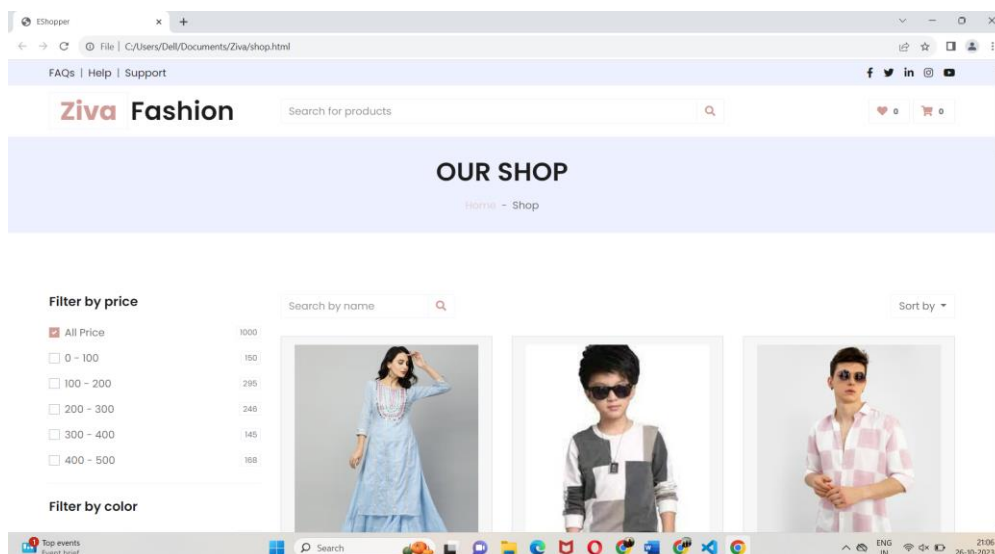
Sample page:



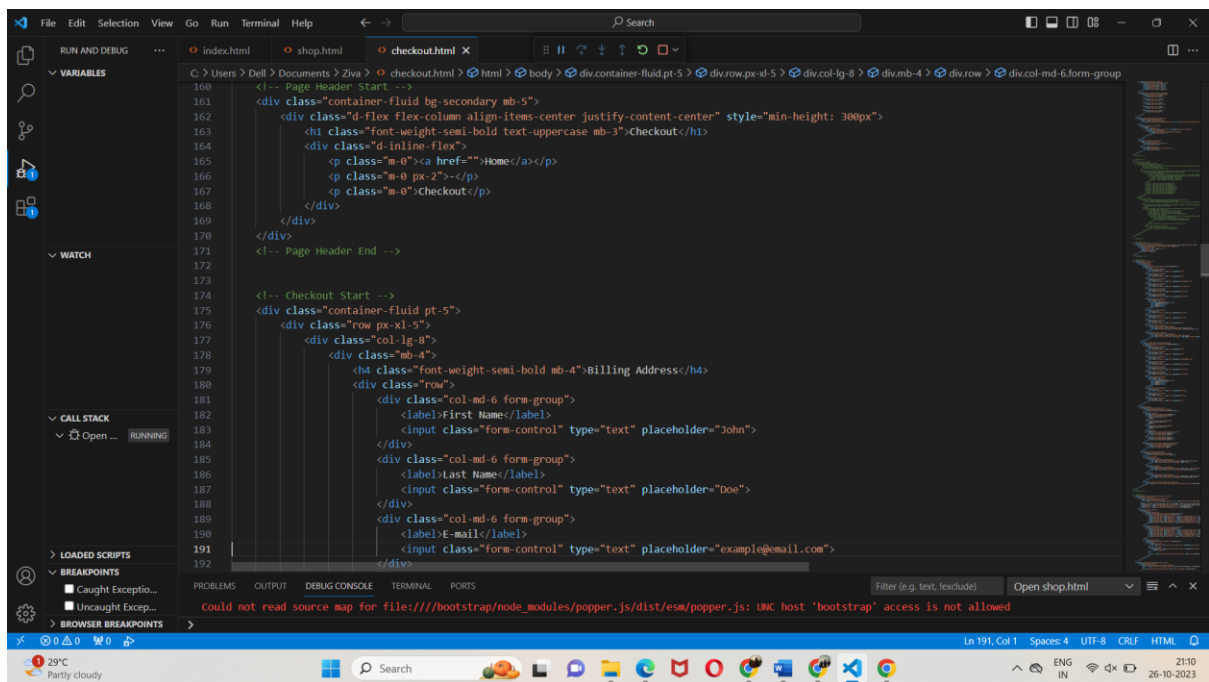
Code for Shopping Cart Page:



Sample page:



Code for Checkout Page:



Sample Page:

