**E-commerce Application on IBM cloud Foundry**

**Phase 5: project documentation and Submission**

Build an artisanal e-commerce platform using IBM Cloud Foundry. Connect Skilled artisans with the global audience. Showcase handmade products , from exquisite jewellery to artistic home decor. Implement secure shopping carts, Smooth payment gateways, and an intuitive checkout process. Nuture creativity and support small business throw an artisan’s dream marketplace!

**Problem Definition:**

* Building an artisanal e-commerce platform using cloud foundry will help the skilled artisans connect with the global audience. The handmade products and handcrafts are not well-known by many people and the job of the artisans are vanishing day by day as people are unaware of the handmade products.
* Inorder to solve the problem, the handcraft products must be marketed for which a e-commerce platform must be created to showcase the products so that people can easily access the products. The objective is to leverage IBM cloud's infrastructure and services to create a secure, scalable and user-friendly online marketplace.
* a web-based portal designed keeping in mind the dedication and hardwork of individual artisans and craftsmen. Aim is to incorporate modern technology to provide artisans with a platform to showcase their skills of crafts and cater to a wider range of audience. This approach reduces the cost of acquiring a middle-man and also provides an opportunity for a greater profit margin for the sellers. Sellers can directly register on the portal and showcase their skills to the world

**Design Thinking:**

Building an artisanal e-commerce platform on IBM Cloud Foundry involves several steps. Below is a high-level outline of the process. Please note that this is a simplified overview, and you might need to adapt it based on your specific requirements and the tools available on IBM Cloud Foundry.

Platform design:

To design the platform layout with sections for product categories, individual project pages, shopping cart, checkout and payment, front end is being used and back end is used to ensure the accessibility.

**Product Showcase**:

Mongo database is used to store product information such as images, description, prices, and categories which is used to create the ecommerce.

**User Authentication:**

User registration and authentication features are implemented . Once this has been confirmed, authorization is then used to enable artisans and customers to access the platform.

**Shopping cart and Checkout:** Back end isresponsible for storing and organising data, and ensuring everything on the client-side actually worksto develop the shopping cart functionality and smooth checkout process.

**Payment Integration:** A payment processor is chosen and security features are implemented ,that is integrated to facilitate transactions.

**User Experience:** Focus on providing an intuitive and visually appealing user experience for both artisans and customers ,products which are usable, credible, valuable, accessible and desirable are much more likely to succeed.

TECHNOLOGY:

MERN Stack

* MERN stack is an acronym for four popular technologies - MongoDB, Express, React, and Node. MERN stack is widely used for developing web applications and provides a robust and scalable environment for building web applications.
* MongoDB is a NoSQL database that is designed to store data in JSON-like documents. MongoDB is widely used for building scalable and flexible databases, making it ideal for web applications.
* Express is a popular web application framework for Node.js that provides a robust set of features for building web applications. Express simplifies the development of web applications by providing a variety of tools and features, such as middleware, routing, and templating engines.
* React is a JavaScript library for building user interfaces. React provides a component-based architecture that makes it easy to build reusable UI components, which can be used across different parts of the application.
* Node JS is a JavaScript runtime built on the Chrome V8 JavaScript engine.. Node JS provides a variety of features that simplify the development of web applications, such as a built-in HTTP server and a module system for managing dependencies.

Advantages of MERN

1) Full-stack java script

2) Fast development

3) Scalability

4) Community support

SYSTEM ARCHITECTURE

The System Architecture is divided into 3 sections: Backend, Database and Frontend.

1) Backend: For backend part we have used NodeJS a JavaScript runtime environment with Express framework which provide APIs.

2) Database: For the database we have used the MongoDB store the user details, product details and order details. MongoDB is a popular NoSQL document-oriented database that stores data in a flexible, JSON-like format called BSON.

3) Frontend : For the Frontend we have used ReactJS. For designing part html, CSS, JavaScript are used.

a) HTML: HTML stands for Hyper Text Markup Language. It is used to create webpages.Html has elements which tells the browser how to display the content.

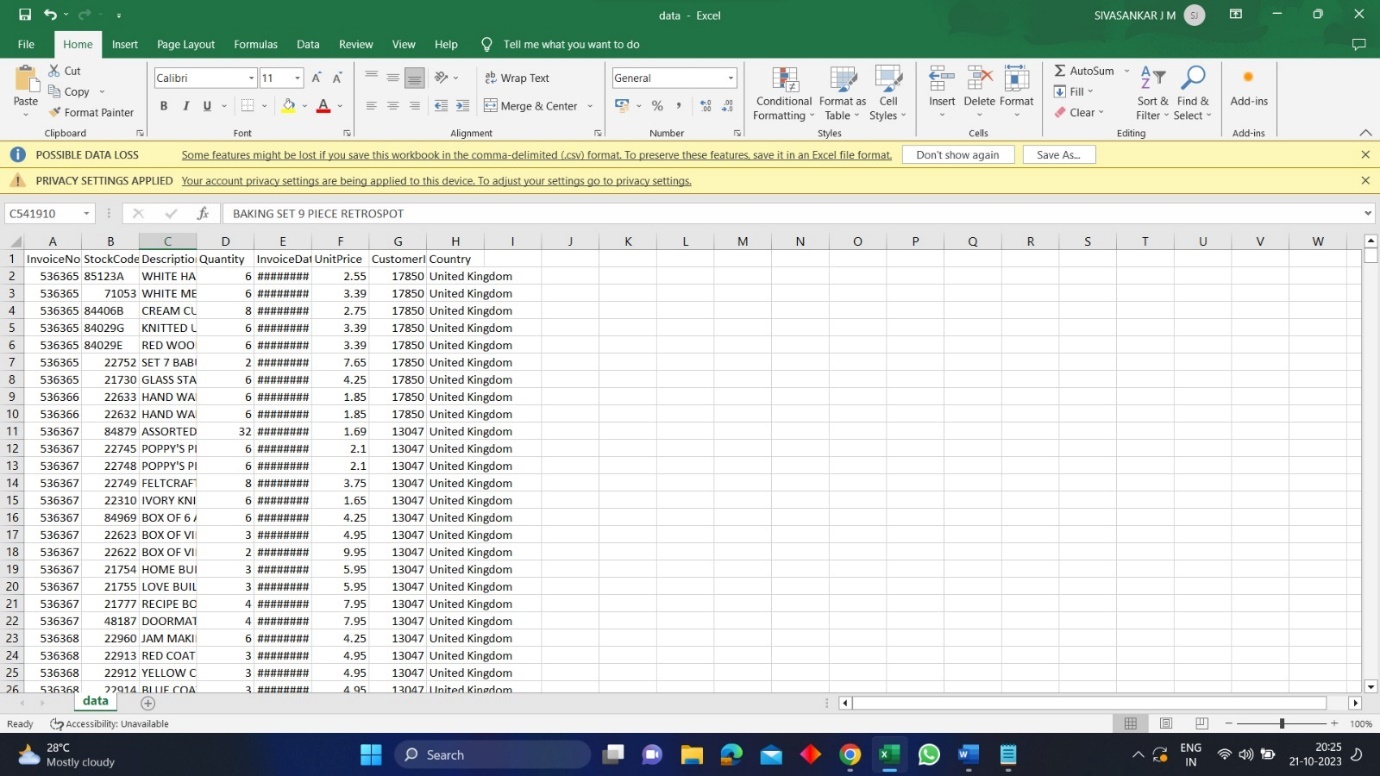
b) CSS: CSS is the language we use to style an HTML document.CSS describes how the html elements should be displayed

c) JavaScript: Is used to program the behaviour of web pages. For the database we have used the MongoDB store the user details, product details and order details .

The instances that comprise the dataset :

* **Invoice No:** Invoice number. Nominal, a 6-digit integral number uniquely assigned to each transaction. If this code starts with letter 'c', it indicates a cancellation.
* **Stock Code:** Product (item) code. Nominal, a 5-digit integral number uniquely assigned to each distinct product.
* **Description:** Product (item) name. Nominal.
* **Quantity:** The quantities of each product (item) per transaction. Numeric.
* **Invoice Date:** Invoice Date and time. Numeric, the day and time when each transaction was generated.
* **Unit Price:** Unit price. Numeric, Product price per unit in sterling.
* **CustomerID:** Customer number. Nominal, a 5-digit integral number uniquely assigned to each customer.
* **Country:** Country name. Nominal, the name of the country where each customer resides.

**SAMPLE DATASET COLLECTION :**



**Explanation:**

* The Categories table stores information about product categories.
* The Products table contains details about individual products, including the category they belong to.
* The Customers table stores customer information.
* The Orders table tracks orders placed by customers.
* The OrderDetails table contains details about the products included in each order.

This is a basic schema, and you may need to expand it based on additional features such as user authentication, reviews, ratings, and more. Also, consider indexing certain columns to improve query performance, and implement data validation and constraints to maintain data integrity.

**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:**

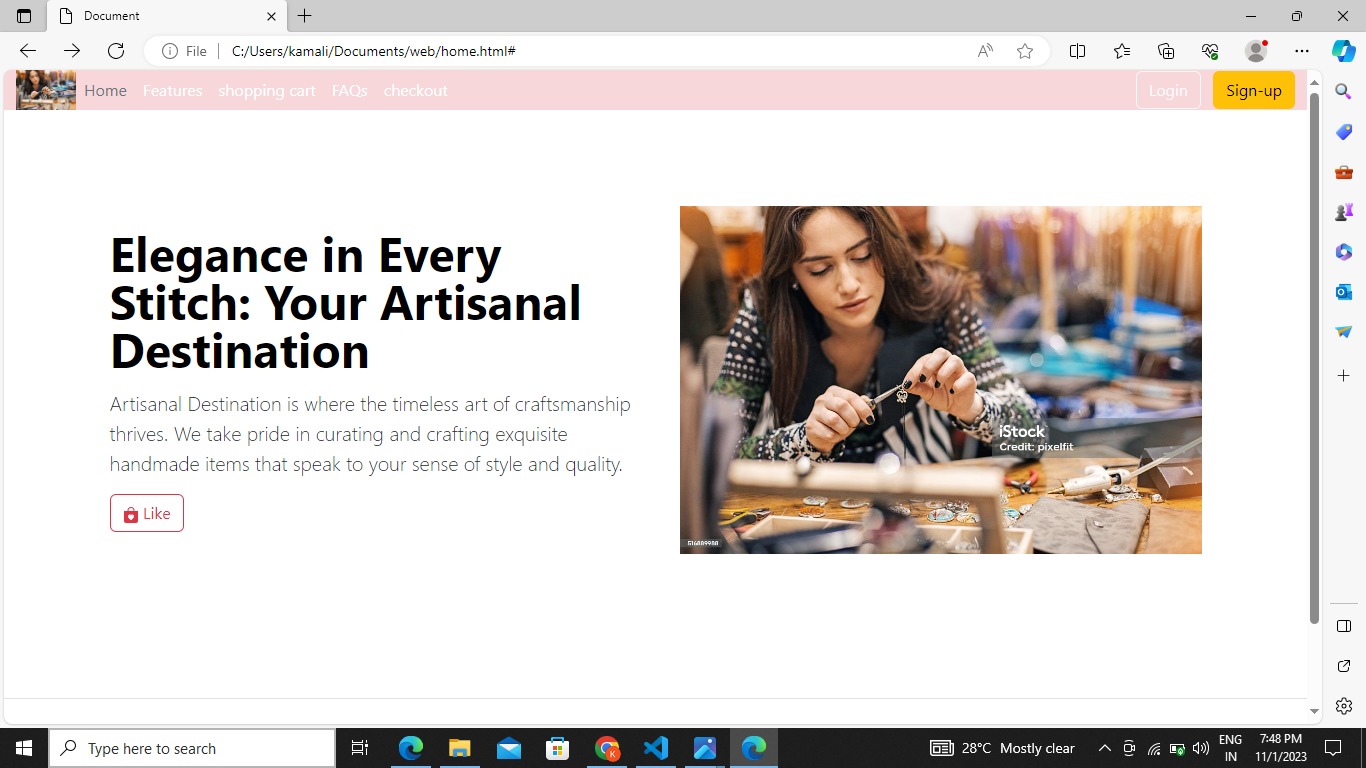
* Implement a smooth and intuitive checkout process that guides users through the final steps of their purchase.
* Integrate with a secure payment gateway to handle transactions, such as Stripe or PayPal.
* Store order details, including the items purchased and customer information, for order fulfilment and tracking.

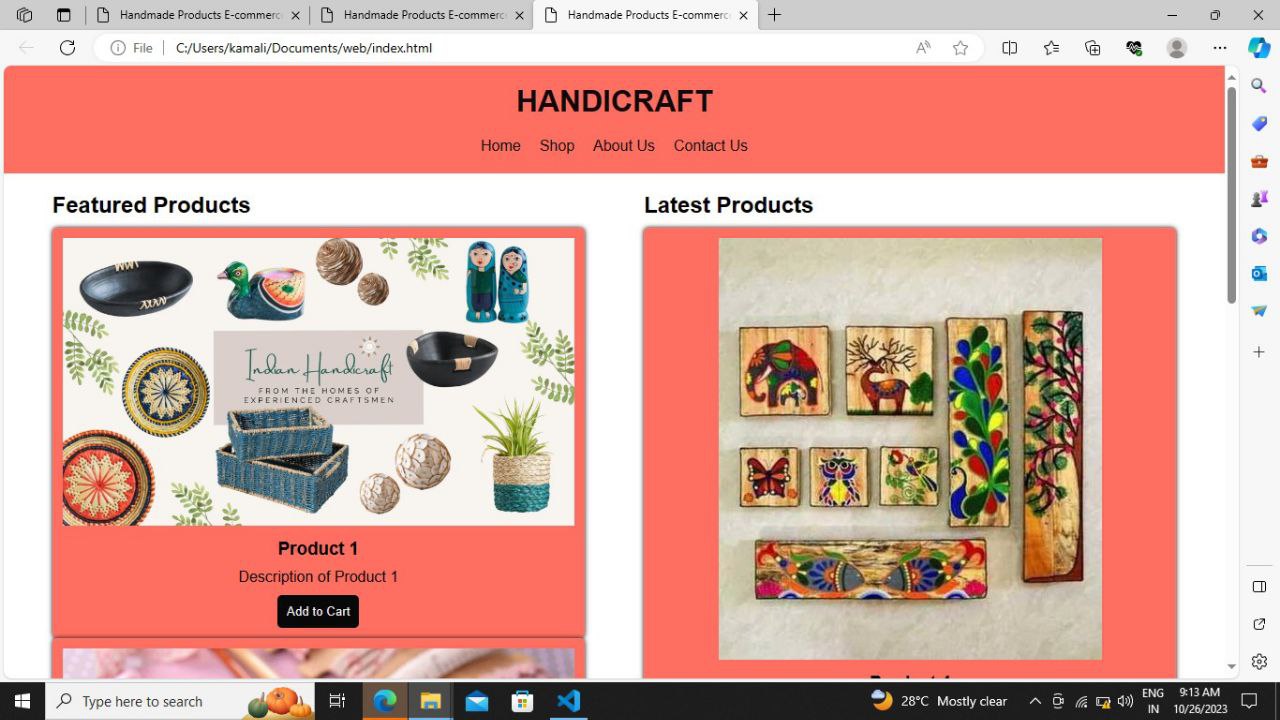
**Security and Reliability**:

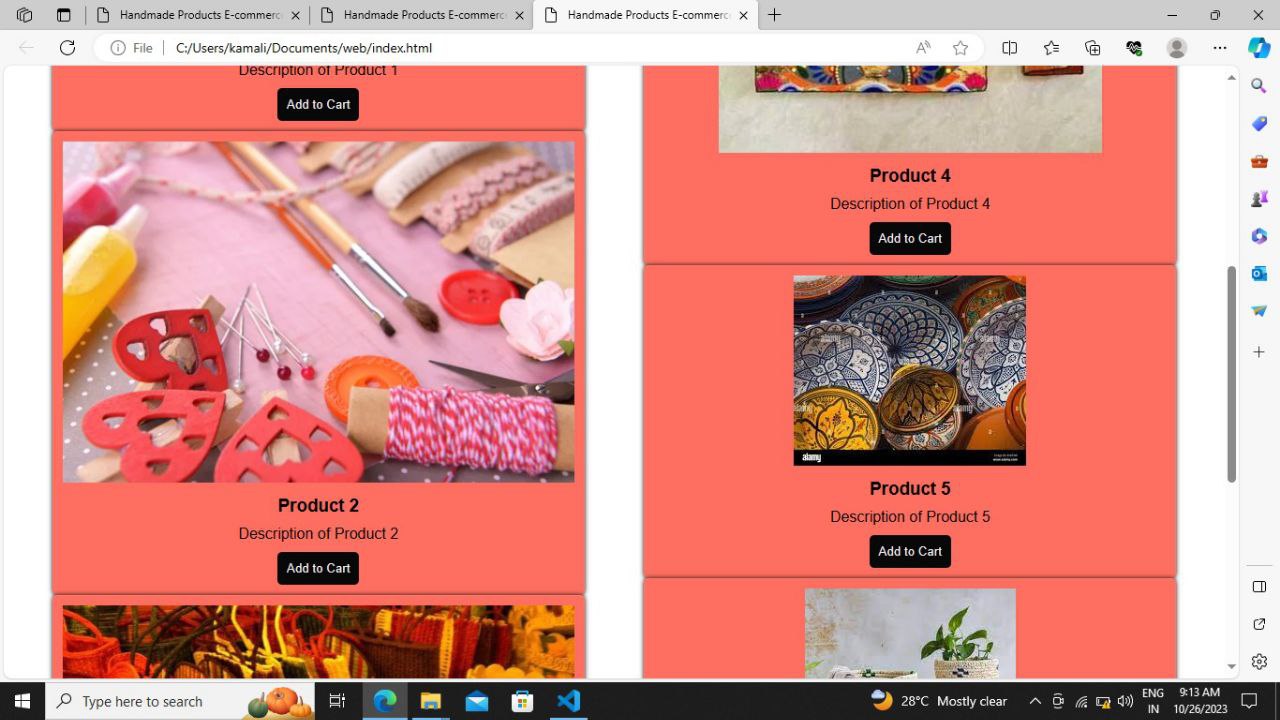
* Ensure that all user data, including personal information and payment details, is stored and transmitted securely.
* Implement appropriate security measures to protect against common web application vulnerabilities, such as SQL injection and cross-site scripting (XSS).

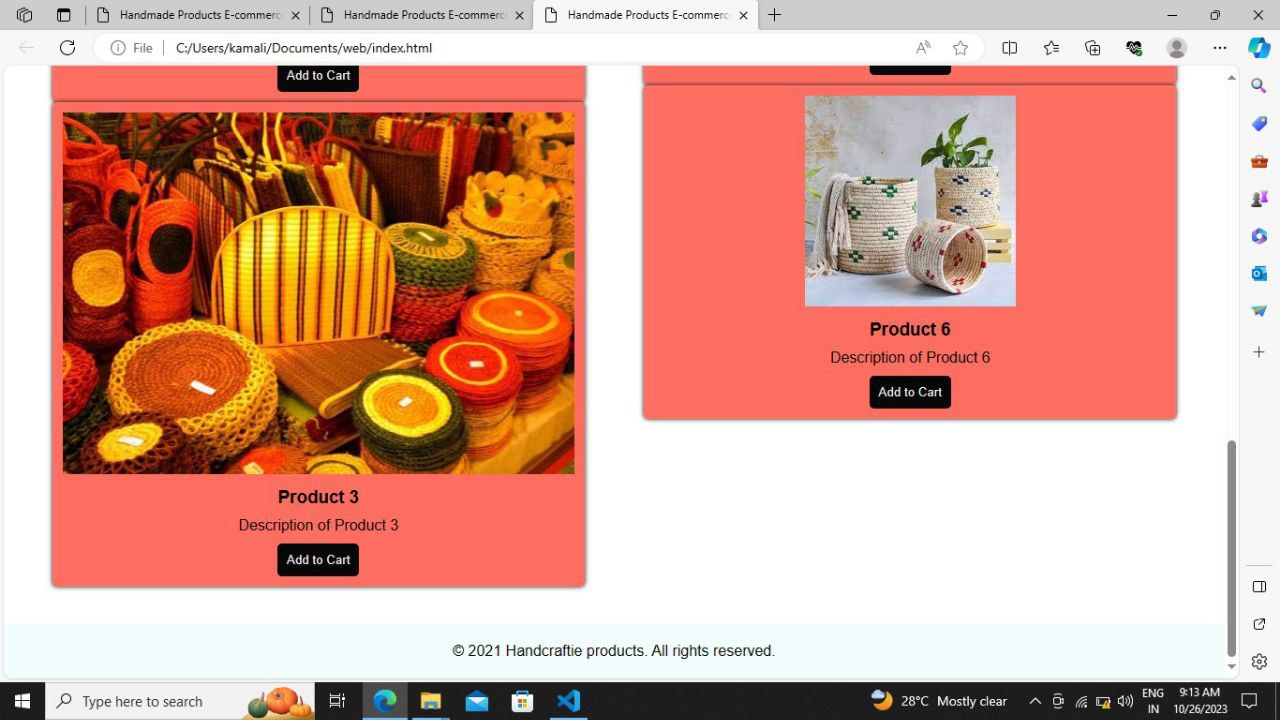
Implement HTTPS to encrypt data during transmission, enhancing data security.

**Sample home page:**

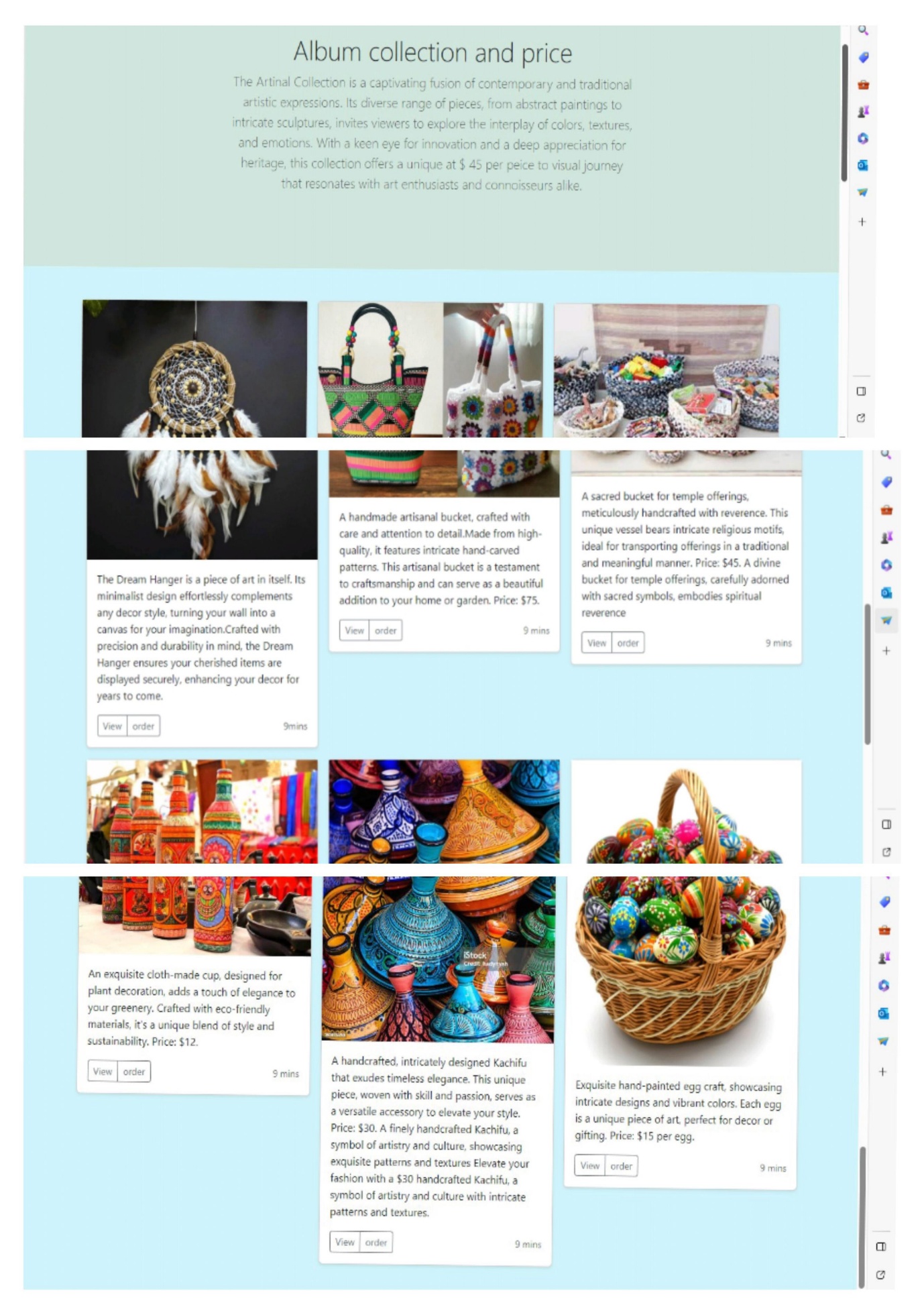


**Sample Feature Output:**

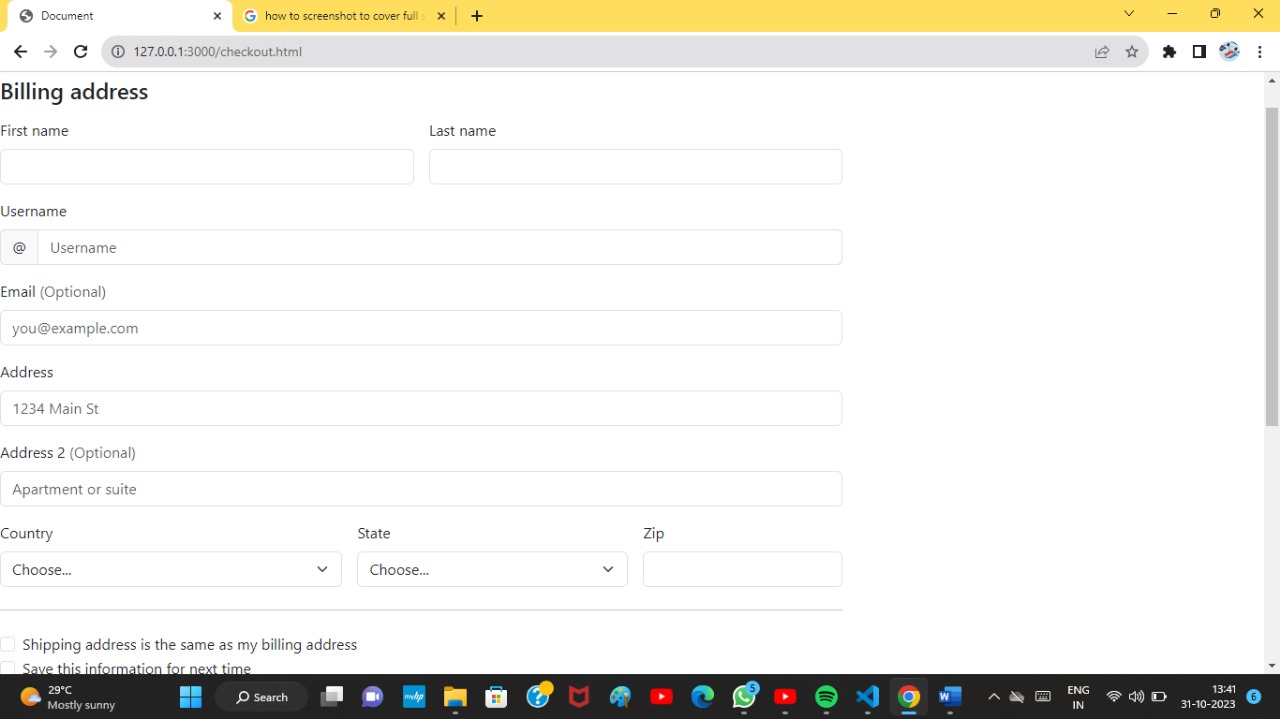
****

****

Shopping Cart Page:



**Checkout Page:**



**CONCLUSION:**

In conclusion, our e-commerce application's integration with IBM Cloud Foundry has yielded cost-effective and highly available infrastructure. The platform's ease of use and automation features have streamlined our operations, allowing us to focus more on innovation and customer experience. With a strong foundation in the Cloud Foundry environment, our application is well-positioned for future growth and adaptability in the ever-evolving e-commerce landscape. This strategic move ensures that our business remains competitive and agile in the digital marketplace while maintaining a high level of customer satisfaction.