**UNITED INSTITUTE OF TECHNOLOGY**

**(College code:7145)**

**E -Commerce Application On IBM cloud foundry**

**Team Members**

**BHUVANESHWARI R NAVEEN KUMAR S**

**HARINI R SHONE V SAMUEL**

**MAHALAKSHMI K UMASHANKAR S**

**Mentor**

**Nishanthi - nishanthi@uit.ac.in**

**E -Commerce Application On IBM cloud foundry**

**Introduction :**

E-commerce applications on IBM Cloud Foundry offer businesses a robust and scalable platform to establish their online presence. Leveraging IBM’s cloud infrastructure, these applications enable companies to efficiently manage their digital storefronts, streamline customer experiences, and optimize their e-commerce operations. With the flexibility of Cloud Foundry, businesses can adapt and scale their e-commerce solutions as needed, ensuring high availability and reliability. This introduction sets the stage for a discussion of the various features, benefits, and capabilities of e-commerce applications on IBM Cloud Foundry.

**Innovation:**

Designing an innovative e-commerce application on Cloud Foundry involves several key considerations. Here’s a high-level overview of the process:

**Application Architecture:**

Choose a microservices architecture for flexibility and scalability.Decouple components such as product catalog, shopping cart, user management, and payment processing.

Utilize Cloud Foundry services for databases, messaging, and caching.

**User Experience:**

Implement a responsive and user-friendly design for both web and mobile platforms.

Personalization and recommendation engines to enhance user experience.

**Scalability:**

Leverage Cloud Foundry’s auto-scaling capabilities to handle traffic spikes.

Implement load balancing to distribute traffic across multiple instances.

**Security:**

Utilize HTTPS, data encryption, and secure authentication methods. Regularly update and patch your application to protect against vulnerabilities.

**Data Management:**

Choose a suitable database technology (e.g., relational, NoSQL) based on your data requirements.

Implement data redundancy and backups for fault tolerance.

**Payment Processing:**

Integrate with secure payment gateways and implement encryption for sensitive data.

**Inventory Management:**

Implement real-time inventory tracking to prevent overselling.

Utilize Cloud Foundry’s event-driven architecture for inventory updates.

**Analytics:**

Collect and analyze user data to improve the customer experience.

Use Cloud Foundry’s built-in monitoring and logging tools.

**DevOps and CI/CD:**

Implement continuous integration and continuous deployment (CI/CD) pipelines. Use Cloud Foundry’s automation features for smooth deployment and scaling.

**Cost Management:**

Regularly monitor resource usage to optimize costs. Utilize Cloud Foundry’s resource scaling and allocation features effectively.

**Third-Party Integrations:**

Integrate with external services such as shipping providers and social media platforms.

**Compliance:**

Ensure compliance with data protection regulations (e.g., GDPR) and industry-specific standards.

**Testing and Quality Assurance:**

Implement automated testing to maintain application quality. Utilize Cloud Foundry’s staging environments for testing.

**Feedback Loop:**

Collect user feedback to continuously improve the application.

**Documentation and Training:**

Document your application’s architecture, APIs, and deployment procedures. Provide training for your development and operations teams.

**Maintenance and Support:**

Set up a robust support system to address customer issues promptly. Perform regular maintenance and updates.

**CREATING A E- COMMERCE APPLICATIONS ON CLOUD FOUNDRY:**

**CREATING AN E-COMMERCE APPLICATION ON CLOUD FOUNDRY INVOLVES SEVERAL STEPS:**

**• SET UP CLOUD FOUNDRY:**

Ensure you have access to a Cloud Foundry environment. You can use a Platform like IBM Cloud, Pivotal Web Services, or SAP Cloud Platform, all of Which support Cloud Foundry.

**• SELECT A PROGRAMMING LANGUAGE:**

Choose a programming language and framework for your e-commerce Application. Popular choices include Java (using Spring Boot), Node.js, or Ruby on Rails.

**• DATABASE**

Decide on a database for your application. Cloud Foundry supports various Databases like PostgreSQL, MySQL, and MongoDB. You can choose the one that best fits your needs.

**• APPLICATION DEVELOPMENT:**

Develop your e-commerce application using the selected language and Framework. Ensure that it includes features like product listings, shopping Cart, user authentication, and payment processing.

**• CONTAINERIZATION :**

Containerize your application using Docker. This allows you to package Your application and its dependencies into a container image.

**• CLOUD FOUNDRY MANIFEST:**

Create a Cloud Foundry manifest file that specifies details about your Application, including the runtime, memory, and services it requires.

**• PUSH TO CLOUD FOUNDRY:**

Use the Cloud Foundry command-line interface (cf CLI) to push your Application to the Cloud Foundry platform. This will make your application Accessible on the cloud.

**• SERVICE INTEGRATION:**

Integrate any necessary services like payment gateways, caching, or CDN Services. You can use Cloud Foundry’s marketplace to add and bind these Services to your app.

**SCALING:**

Configure auto-scaling and load balancing to ensure your e-commerce Application can handle varying levels of traffic.

**• MONITORING AND LOGGING :**

Set up monitoring and logging to keep an eye on the health and Performance of your application. Cloud Foundry often provides tools and Integrations for this purpose.

**• SECURITY:**

Implement security best practices, including data encryption, secure Authentication, and protection against common web application Vulnerabilities like cross-site scripting (XSS) and SQL injection.

**• CONTINUOUS INTEGRATION AND DEPLOYMENT(CI/CD):**

Implement CI/CD pipelines to automate the deployment process and Ensure that updates are rolled out seamlessly.

**•TESTING:**

Thoroughly test your e-commerce application to ensure it functions Correctly and securely.

**• BACKUP AND DISASTER RECOVERY:**

Implement backup and disasterrecovery solutions to protect your data And ensure business continuity.

**• TECHNOLOGY STACK:**

Choose a technology stack that supports real-time features. This might Include a combination of are us follows:

**• BACKEND FRAMEWORK:**

Node.js with frameworks like Express.js or Nest.js, or other real-time Capable languages and frameworks.

**• FRONTEND FRAMEWORK:**

A modern JavaScript framework like React, Angular, Or Vue.js.

**• DOMAIN SETUP:**

Configure your domain and DNS settings to point to your e-commerce Application on Cloud Foundry

**• LUNCH AND MONITOR:**

Finally, launch your e-commerce application and continuously monitor its Performance, making necessary optimizations as needed.

**• SOURCE CODE:**

**<!DOCTYPE html> <html>**

**<head>**

**<title>E-Commerce Store</title> </head>**

**<body>**

**<h1>Welcome to our E-Commerce Store</h1>**

**<!—Product Listing 🡪 <div class=”product-list”>**

**<div class=”product”>**

**<img src=”product1.jpg” alt=”Product1”> <h2>Product1</h2>**

**<p>Price: $10.00</p>**

**<button onclick=”addToCart(1)”>Add to Cart</button> </div>**

**<div class=”product”>**

**<img src=”product2.jpg” alt=”Produc**

**2”>**

**<h2>Product 2</h2> <p>Price: $20.00</p>**

**<button onclick=”addToCart(2)”>Add to Cart</button> </div>**

**<!—Add more product listings here 🡪 </div>**

**<script>**

**Function addToCart(productId) {**

**// Implement JavaScript logic to add the selected product to the shopping**

**Cart. // You may want to use AJAX to update the cartin real-time.**

**} </script>**

**Output:**

**The output of this HTML code,when opened in a web browser, will display A webpage with the following elements:**

**The title of the webpage is “E-Commerce Store.”**

**A heading: “Welcome to our E-Commerce Store.”**

**A product listing section with two product entries. Each entry includes:**

**An image of the product.**

**The product name (e.g., “Product1” and “Product 2”). The product price**

**(e.g., “$10.00” and “$20.00”).**

**An “Add to Cart” button that, when clicked, triggers the JavaScript Function “addToCart(productId)” with the respective product’s ID as an Argument.**

**An high-level overview of how to implement user Authentication, shopping cart, and checkout functionality:**

**User Authentication:**

Choose a backend technology for your server. Implement user registration, including storing user data Securely in a database.Create an authentication system that Issues and verifies user tokens.

**Shopping Cart:**

Design a data structure to store the contents of a user’s Shopping cart, typically linked to their session or user ID.Calculate the total price of items in the cart, considering Quantities and prices.

**Checkout Process:**

Create a secure endpoint for initiating the checkout Process.Validate and process payment securely using a Payment gateway.Deduct the amount from the user’s Payment method upon successful payment.Clear the user’s Shopping cart after a successful purchase.

**Security:**

Implement HTTPS to secure data transmission.Sanitize andValidate user input to prevent security vulnerabilities.Keep Your software dependencies and frameworks up to date toAddress security vulnerabilities.

**Testing and Deployment:**

Thoroughly test your application, including unit tests, Integration tests, and security testing.Deploy your application On a reliable server or cloud platform .Monitor and log Application behavior to identify and resolve issues.

**User Experience:**

Design a user-friendly and responsive front-end interface for Users to interact with your e-commerce platform.Ensure a Smooth and intuitive shopping experience, with features like Search, and product details.

**E-Commerce applications In user authentication Steps:**

**User Registration:**

When a user wants to create an account on an e-commerce Platform, they provide their personal information, such as Name, email, and a password. User authentication is used to Validate the user’s identity during the registration process.

**Login and Account Access**:

After registration, users can log in to their accounts by Providing their username/email and password Successful Authentication allows users to access their account, view Order history, manage payment methods, and perform Various actions.

**Password Reset and Recover**:

Users may forget their passwords or need to reset them.This Often involves sending a password reset link to the user’s Email, which expires after a certain period and requires the User to establish a new password using mechanisms like CAPTCHA.

**Security and Compliance:**

In e-commerce, complying with security and privacy Regulations is crucial. User authentication is integral to Providing a secure and personalized shopping experience in e-commerce applications. It safeguards user accounts and data, builds trust is a fundamental component in ensuring the confidentiality and integrity of sensitive information.

**E-Commerce applications In shopping steps**:

**Product Selection:**

Users browse through the product catalog and select items they Wish to purchase. Each product listing includes details such as price, Description, and images.

**Quantity Adjustment:**

Users can adjust the quantity of each product in the cart. They can Increase or decrease the quantity of a specific item they want to Buy.

**Cross-sell and Up-sell**:

E-commerce platforms often use the shopping cart as an Opportunity to recommend additional products (cross-selling) or Encourage users to upgrade items (up-selling).

**Inventory Management:**

The e-commerce platform updates the inventory to reflect the Purchased items, reducing stock counts accordingly.

**E-Commerce applications In security steps**:

**Data Encryption:**

Use HTTPS (SSL/TLS) to encrypt data transmitted between The user’s browser and your server. This protects sensitive Information, such as login credentials and payment details.

**Data Storage Security:**

Store user data (e.g., passwords, personal information) Securely using encryption and hashing. Protect databases From unauthorized access.

**DDoS Protection:**

Implement DDoS (Distributed Denial of Service) protection to Guard against malicious traffic floods that can disrupt your service.

**Monitoring and Logging:**

Implement robust monitoring and logging mechanisms to Detect and respond to security incidents. Analyze logs to Identify unusual activity.

**Third-Party Security:**

Assess the security practices of third-party services and Plugins used in your application, as vulnerabilities in these components can impact your application.

**Implementation of authentication and registration using Backend server:**

**Create API Endpoints:**

Define routes and API endpoints for user Registration and authentication. You can use a Framework like Express.js to handle routing.

**Database Integration:**

Choose a database (e.g., MongoDB, PostgreSQL) And set up a connection to store user data. Use a Library like Mongoose for MongoDB or Sequelize for PostgreSQL.

**User Registration:**

Implement a registration endpoint that accepts user data (e.g., username, email, password), validates it, and stores the User’s information in the database after hashing the Password.

**User Authentication:**

Create an authentication endpoint where users can provide Their credentials (e.g., email and password). Validate the Credentials, compare the hashed password with the stored Hash, and generate a JSON Web Token (JWT) for Authentication.

**JWT Handling :**

Use a JWT library (e.g., jsonwebtoken) to handle token Creation, validation, and expiration. Certainly, here’s an outline for documenting and preparing your e-commerce platform project for submission:



**Challengeas faced** :

Inventory Management: Maintaining accurate product listings, stock levels, and real-time updates can be technically demanding.

**Search and Filtering:**

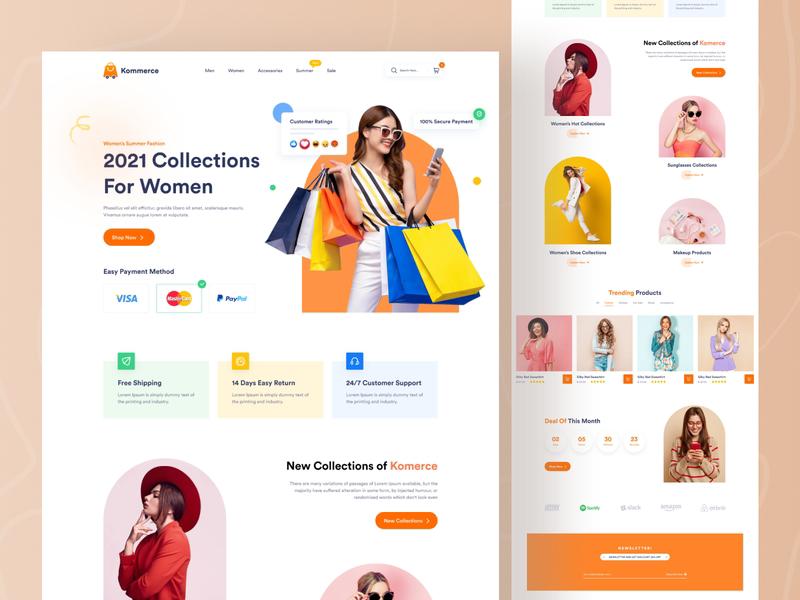
Implementing effective search and filtering mechanisms to help users find products easily requires robust algorithms and data structures.

**Order Processing:**

Handling order fulfillment, shipping, and tracking while optimizing for speed and accuracy can be a technical challenge.

**SEO and Performance:**

Optimizing the platform for search engines (SEO) and ensuring fast page load times are crucial for attracting and retaining customers.



**Trouble shooting issues**:

**Website Performance:**

Slow Page Load Times: Optimize code, compress images, and utilize content delivery networks (CDNs) to improve loading speed.

**Downtime:**

Monitor server uptime, set up alerts for outages, and have a backup plan in place.

**Payment Issues:**

Failed Transactions: Investigate the reasons for failed payments and provide helpful error messages to users.

**Payment Gateway Problems:**

Check for issues with the payment gateway integration, and contact the service provider for support.

**Mobile Responsiveness:**

**Cross-Browser and Device Testing**:

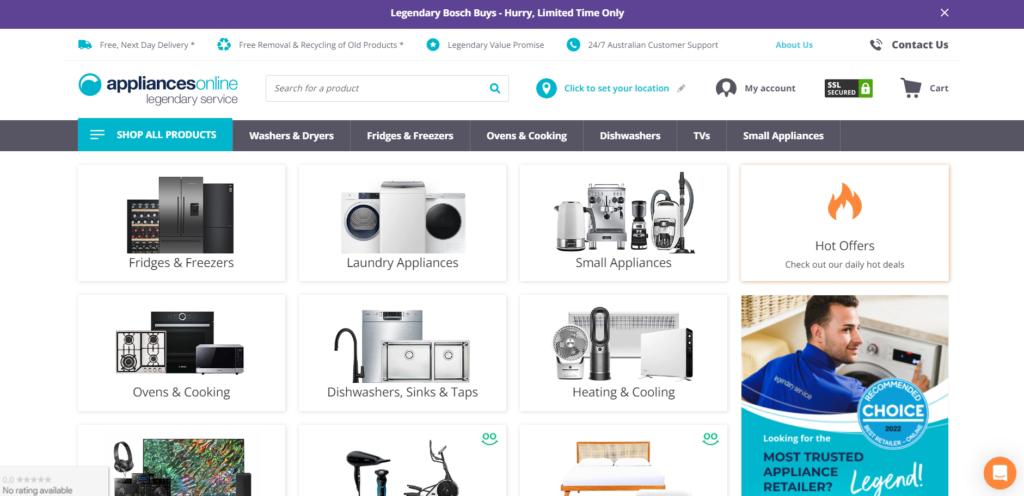
Ensure the website works well on various browsers and devices, and address any compatibility issues.

**Customer Support:**

Be responsive to customer complaints, and have clear channels for support.

Communication: Keep customers informed about any ongoing issues and estimated resolution times.

**Website design:**

****

**Conclusion:**

In summary, e-commerce applications on IBM Cloud Foundry provide a robust and flexible platform for businesses to establish and maintain their online stores. This platform ensures reliability, security, and scalability, enabling businesses to focus on delivering a seamless shopping experience to their customers.

Creating an e-commerce application on the cloud is a Strategic and efficient way to establish and operate an online store. It Offers several advantages, including scalability, cost-efficiency, and flexibility.