1. **Sign Up for IBM Cloud:**

If you haven’t already, sign up for an IBM Cloud account at [https://cloud.ibm.com/.](https://cloud.ibm.com/)

1. Install Required Tools:

Install the IBM Cloud CLI (Command Line Interface) on your local machine. You can download it from [https://cloud.ibm.com/docs/cli.](https://cloud.ibm.com/docs/cli)

1. Create a New Cloud Foundry Application:

Log in to your IBM Cloud account using the CLI:

Ibmcloud login

Target the appropriate resource group and region:

Ibmcloud target -g <resource-group> -r <region> Create a new Cloud Foundry application:

Ibmcloud cf create-service cloudantnosqldb Lite my-cloudant

Ibmcloud cf push my-artisanal-app

This example uses a Lite Cloud ant NoSQL database service. You can choose other database services or services based on your requirements.

1. Develop Your E-Commerce Application:

• Begin developing your artisanal e-commerce platform. You can use popular web development frameworks and tools like Node’s, Python, Ruby, or others.

5. Connect to the Cloud ant Database:

* You can use the IBM Cloud SDKs to connect your application to the Cloud ant NoSQL database. Install the SDK and include it in your code.
* Use environment variables to securely store and access credentials and connection information.

6. Implement E-Commerce Features:

• Implement features like product listing, user registration, shopping cart, and payment processing in your application.

7. Set Up Security:

• Implement proper security measures, such as user authentication, encryption, and protection against common web vulnerabilities.

8. Testing:

• Thoroughly test your application to ensure it functions as expected.

9. Deployment:

• Deploy your application to IBM Cloud

Ibmcloud cf push my-artisanal-app

10. Scale Your Application:

• IBM Cloud Foundry allows you to scale your application based on demand. You can configure auto-scaling rules to handle increased traffic.

11. Monitoring and Maintenance:

* Set up monitoring and logging for your application to track its performance.
* Regularly update and maintain your platform, addressing any bugs and adding new features.

12. Domain and SSL Configuration:

• Configure custom domain and SSL certificates for your ecommerce platform if needed.

13. Backup and Disaster Recovery:

• Implement backup and disaster recovery strategies for your database and application data.

14. Compliance and Legal Considerations:

• Ensure that your e-commerce platform complies with legal and regulatory requirements for online retail businesses.

Remember that the specific steps and technologies used may vary based on your chosen development language, framework, and requirements. Be sure to consult the IBM Cloud documentation and seek expert advice if needed.

Additionally, consider implementing continuous integration and continuous deployment (CI/CD) pipelines for efficient development and deployment processes.

1. Requirements Analysis:

* Define the scope of your e-commerce platform.
* Identify the types of products you want to sell.
* Determine the features and functionality you want to offer (e.g., product search, shopping cart, user profiles, reviews).

2. System Architecture:

* Choose the technology stack for your platform (e.g., webbased technologies like HTML, CSS, JavaScript, and backend technologies like Node’s or Python).
* Select a database management system (e.g., MySQL, PostgreSQL, MongoDB).

3. Database Design:

* Create an Entity-Relationship Diagram (ERD) to define your database structure. Here’s a simplified example:
* Product table: ID, Name, Description, Price, Category, Stock, etc.
* User table: ID, Username, Email, Password, etc.
* Order table: ID, User ID (foreign key), Total Price, Date, etc.

Order Item table: ID, Order ID (foreign key), Product ID (foreign key), Quantity, Price, etc.

* Define relationships between tables using foreign keys.

4. Front-End Design:

* Create wireframes or mockups of your website’s layout and user interface. Tools like Figma, Adobe XD, or even pen and paper can be useful.
* Design a user-friendly and responsive UI, keeping in mind the user experience.

5. Back-End Development:

* Develop the server-side logic for your platform using your chosen technology stack.
* Implement APIs for user registration, product listing, cart management, order processing, etc.
* Connect your server to the database.

6. Front-End Development:

* Implement the user interface based on your design.
* Integrate with the back-end using AJAX, REST APIs, or other relevant methods.
* Implement features like product search, user registration, shopping cart, and payment processing.

7. Testing:

• Test the platform thoroughly to ensure functionality, security, and performance.

Implement security measures like user authentication, encryption, and protection against common vulnerabilities (e.g., SQL injection, cross-site scripting).

8. Deployment:

* Choose a hosting environment (e.g., cloud-based hosting like AWS, Azure, or a dedicated server).
* Deploy your front-end and back-end to the chosen hosting platform.

9. Maintenance and Scaling:

* Regularly update and maintain your platform, addressing bugs and improving features.
* As your platform grows, consider strategies for scaling, such as load balancing and database optimization.

10. Marketing and Promotion:

* Create a marketing strategy to attract users to your ecommerce platform.

Use digital marketing channels and SEO to improve visibility.

* This is a high-level overview, and each of these steps involves many sub-tasks. Depending on your expertise, you may need the assistance of developers, designers, and database administrators. It’s also important to keep up with best practices in e-commerce and web development to ensure your platform’s success.
* Program

<input type=”password” placeholder=”Enter Password” < name=”password” required>

<button type=”submit”>Login</button>

<input type=”checkbox” checked=”checked”>

Output

Login

USERNAME: Enter your name

PASSWORD: Enter your password

THANK YOU……