Ecommerce Application on IBM Cloud Foundry

Introduction:

In this document, my project ecommerce application is for handmade products is explained in detail. Explains how this application works, how store the user data and product information in database table.

Abstraction:

Building an e-commerce application using IBM Cloud Foundry involves leveraging IBM's platform-as-a-service (PaaS) offering to deploy and run your application. In this project, implementing programming languages to build the ecommerce application. This ecommerce application is for handmade products.

Modules:

User module:

It is responsible for user authentication via email id and login also.Login page layout is designed by html, CSS. It uses the user information.

Product Management Module:

Displaying and listing the handmade products with price details and features. It also shows the available stock.

Cart Module:

In this application, shopping cart is available. If you want to buy later the products are added to the cart.

Order Module:

Implement the order details. Order and tracking details will be shown to the user or customer

Payment Module:

Implement payment processing in this application. User will pay the amount via online and also cash on delivery payment. Emi also available.

Inventory Module:

Implement product availability checks and stock updates.So user will know the availality of product.If the stock is coming notifies via email.

Review and Rating Module:

Implement the ability for users to leave reviews and ratings for the products.

Database Module:

Setting up a database to store user data, product information, cart contents, orders, and reviews.

Authentication and Authorization Module:

Ensure secure access to the application and modules by implementing proper authentication and authorization mechanisms.If the error coming in authentication does not enter into the application.

Integration Module:

Integrate with third-party services (if required) for payment processing, shipping, and other external functionalities.

Logging and Monitoring Module:

Implement logging and monitoring to track application performance and user activities.

Security Module:

Implement security measures such as data encryption, input validation, and protection against common web vulnerabilities.

Deployment Module:

Set up deployment scripts and configurations for IBM Cloud Foundry.

Documentation Module:

Document the codebase, APIs, and deployment procedures for future reference.

Resources:

Resources are used for this applications are:

- frontend Language:

  HTML, CSS

- Backend Languages:

  Php, My SQL

Frontend language is to build the frontend of an application by using HTML, CSS.

Backend language is to build the database for the application by using MySQL, php.

Design Thinking:

Design thinking is a human-centered approach to problem-solving and innovation that can be highly valuable in the context of e-commerce. It focuses on understanding user needs, ideating creative solutions, and iteratively testing and refining those solutions. Here's how you can apply design thinking to e-commerce:

Empathize:

Start by understanding your target audience, their needs, and pain points. Conduct user research, surveys, and interviews to gain insights into their shopping behaviors and preferences.

Define:

Based on your research, define specific problem statements and user personas. Clearly articulate the challenges and opportunities in your e-commerce business.

Ideate:

Brainstorm creative ideas to address the defined problems. Encourage a diverse team to generate a wide range of solutions. Consider new features, services, or product offerings that could enhance the shopping experience.

Prototype:

Create low-fidelity prototypes and wireframes to visualize your ideas. Test these prototypes with potential users to gather feedback. This iterative process helps in refining concepts.

Test:

Conduct usability testing and gather feedback on your prototypes. Use A/B testing and analytics to evaluate different solutions. This helps you identify what works best for your users.

Refine:

Based on user feedback and test results, refine your concepts. Iterate on your prototypes and designs to address any issues and improve the user experience.

Implement:

Once you have a well-defined and user-tested solution, proceed with the implementation. Develop the necessary features, user interfaces, and back-end functionality.

Launch:

Launch the updated e-commerce platform or feature to a select group of users or as a limited beta. Monitor its performance and gather real-world feedback.

Feedback and Iterate:

Continue to collect user feedback and iterate on your e-commerce solution. This may involve making small improvements or adding new features to meet evolving user needs.

Scale:

Once your solution proves successful with a smaller audience, scale it to a larger user base. Monitor its performance and make adjustments as necessary.

Customer Support and Engagement:

Provide excellent customer support and engage with your users. Address their concerns, respond to inquiries, and actively seek feedback to ensure a positive shopping experience.

Sustainability and Responsibility:

Consider the environmental and social impact of your e-commerce operations. Implement sustainable practices and ethical sourcing. This can be a significant selling point for many modern consumers.

Design thinking in e-commerce helps you create user-friendly, innovative, and customer-centric experiences. By focusing on empathy and continuous improvement, you can adapt to changing market dynamics and meet the ever-evolving needs of your customers

Development Phase:

There are three development phases:

- First Build the layout or template using html, css
- Then Create the database using MySQL for storing
- Lastly, the registration page, login page and shopping cart page function are functioning and functions the payment process also.

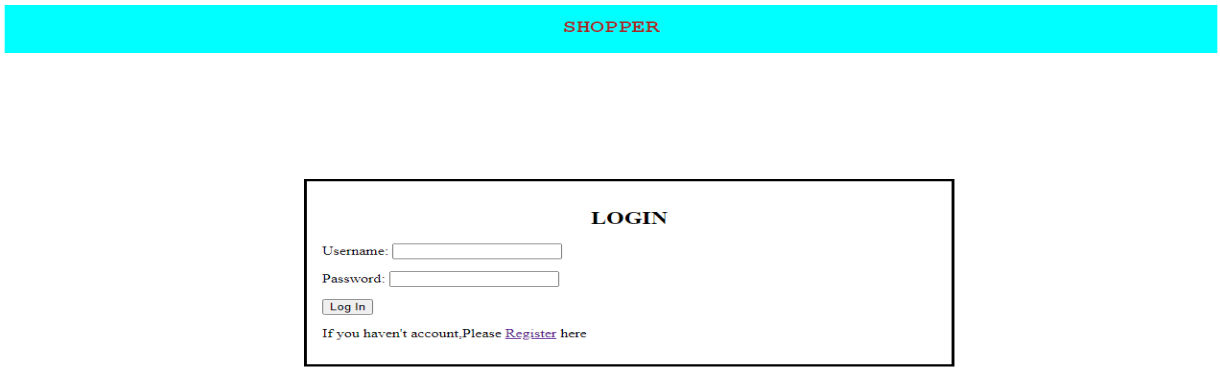
Innovations:

- You will have received notification msg one day before delivery.
- You will also track your order.
- All types of payment methods.
- Support every type of device and system.

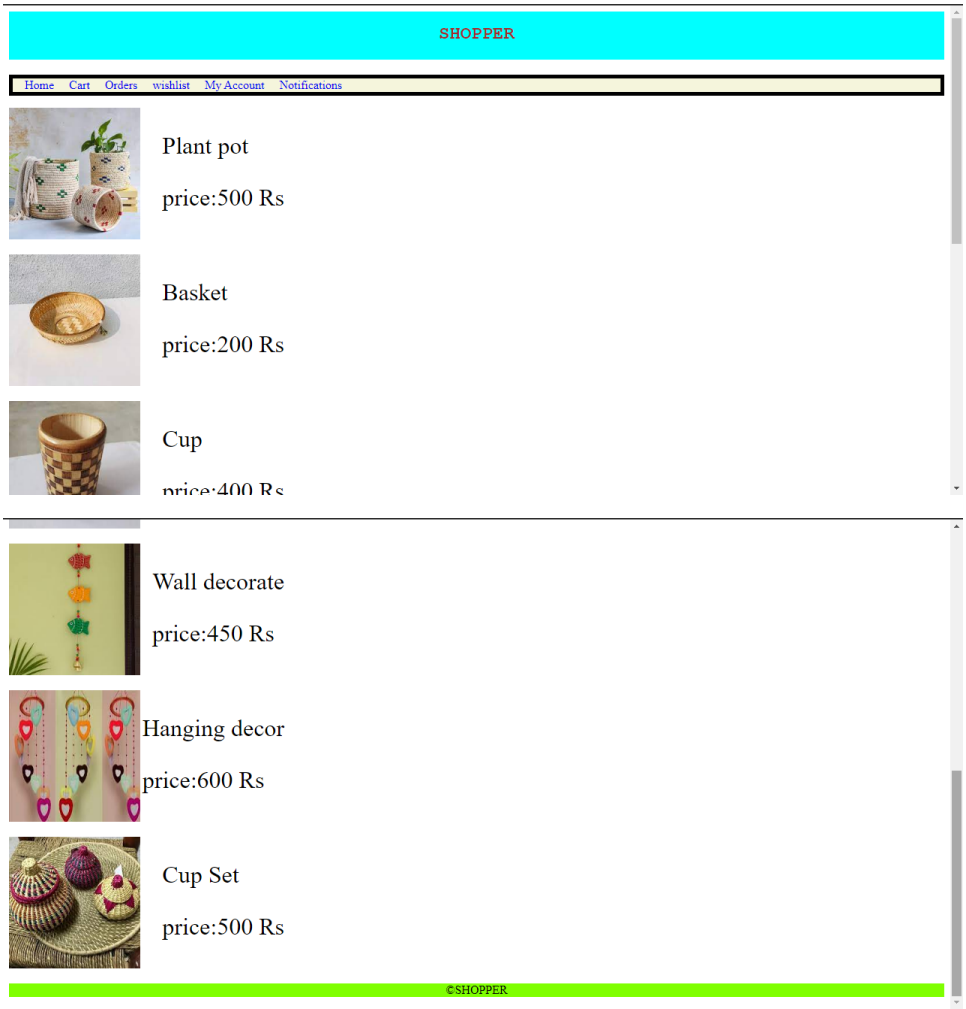
Templates:

Templates for this application are created by using html, css. Template means user interface of an application. Many templates are there. The templates are attached below as image,

Register Template



Login Template



Home Page Template

SHOPPER

Home Cart Orders wishlist My Account Notifications

Shopping Cart

Plant pot

price:500 Rs

©SHOPPER

Shopping Cart Template

SHOPPER

Home Cart Orders wishlist My Account Notifications

Your Order

Plant pot

price:500 Rs

©SHOPPER

Order Template

SHOPPER

Home Cart Orders wishlist My Account Notifications

Your Profile

Username:Rohith

Phonenumber:890-456-1234

Email-id:Championbravo@gmail.com

No:67,Gopal Street,Vyasarpadi,Chennai-600018

Log Out

©SHOPPER

Profile Template

SHOPPER

Home Cart Orders wishlist My Account Notifications

Notifications

©SHOPPER

Notification Template

SHOPPER

Home Cart Orders wishlist My Account Notifications

Your Wishlist

Cup

price:400 Rs

©SHOPPER

Wishlist Template

SHOPPER

Press F11 to exit full screen

Home Cart Orders wishlist My Account Notifications

Plant pot

price:500 Rs

Features

Material:Wood+coconut fabric

uses:Plant Growth

Buy

Add To Cart

Product Template

Payment Template

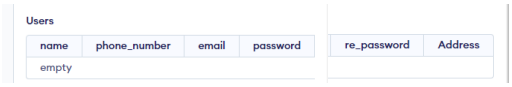
Database:

Database is used for storing the information on database table. For this application, a database is used for storing the information of user and product on database table.

Code:

```
CREATE TABLE users (
  name VARCHAR(100) NOT NULL,
  phone_number VARCHAR(15) NOT NULL,
  email VARCHAR(100) NOT NULL,
  password VARCHAR(255) NOT NULL,
  re_password VARCHAR(255) NOT NULL,
  Address varchar(255) NOT NULL
);
```

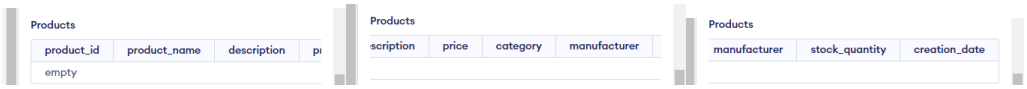
Output:



Code:

```
CREATE TABLE products (
    product_id INT PRIMARY KEY AUTO_INCREMENT,
    product_name VARCHAR(100) NOT NULL,
    description TEXT,
    price DECIMAL(10, 2) NOT NULL,
    category VARCHAR(50),
    manufacturer VARCHAR(100),
    stock_quantity INT NOT NULL,
    creation_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Output:



Storing:

Storing means store the data into the database table such as user information, product information. Php is used for storing the data into database table through submitting in the registration form of html template.

Code:

```
<?php


$name=$_POST['name'];
```

```
$number=$_POST['number'];
$mail=$_POST['mail'];
$password=$_POST['password'];
$repassword=$_POST['repassword'];
$address=$_POST['address'];

if(!empty($name)||!empty($number)||!empty($mail)||!empty($password)||!empty($repassword)||!empty($address))
{
$host="localhost";
$dbusername="root";
$dbpassword="";
$dbname="project";
$conn=new mysqli($host,$dbusername,$dbpassword,$dbname);

if(mysqli_connect_error())
{
die('connect error('.mysqli_connect_error().')'.mysqli_connect_error());
}
else
{
$SELECT="SELECT mail From register where mail= ? Limit 1";
$INSERT="INSERT Into register(name,number,mail,password,repassword,address)values(?,?,?,?,?,?)";
```

```
$stmt=$conn->prepare($SELECT);
$stmt->bind_param("s",$mail);
$stmt->execute();
$stmt->bind_result($mail);
$stmt->store_result();
$rnum=$stmt->num_rows;
if($rnum==0)
{
$stmt->close();
$stmt=$conn->prepare($INSERT);
$stmt->bind_param("sissss",$name,$number,$mail,$password,$repassword,$address);
$stmt->execute();
echo "New record inserted successfully";
}
else{
echo "someone already register using this email";
}
$stmt->close();
$conn->close();
}
}
else{
echo "All field are required";
```

```
die();
}
?>
```

The above code stores the user details in user database table.

Code:

```
<?php
session_start();
include('db_connection.php');

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['username'];
    $password = $_POST['password'];

    $stmt = $conn->prepare("SELECT password FROM register WHERE username = ?");
    $stmt->bind_param("s", $username);
    $stmt->execute();
    $stmt->bind_result($hash);

    if ($stmt->fetch() && password_verify($password, $hash)) {
        $_SESSION['username'] = $username; // Store username in the session
        header("Location: home.html"); // Redirect to the home page
```

```
        exit();
    } else {
        echo "Login failed.";
    }

    $stmt->close();
}

$conn->close();
?>
```

The above php code checks the password in user database table with password entered in login page. If matches, redirecting to the home page otherwise does not enter into home page.

Shopping Cart:

```
<?php
// Start or resume a session
session_start();

// Initialize the shopping cart as an empty array
if (!isset($_SESSION['cart'])) {
    $_SESSION['cart'] = array();
}

// Function to add a product to the cart
```

```
function addToCart($product_id, $quantity) {
    // Check if the product is already in the cart
    if (array_key_exists($product_id, $_SESSION['cart'])) {
        // If it exists, update the quantity
        $_SESSION['cart'][$product_id] += $quantity;
    } else {
        // If it doesn't exist, add it to the cart
        $_SESSION['cart'][$product_id] = $quantity;
    }
}

// Function to remove a product from the cart
function removeFromCart($product_id) {
    // Check if the product is in the cart
    if (array_key_exists($product_id, $_SESSION['cart'])) {
        // If it exists, remove it from the cart
        unset($_SESSION['cart'][$product_id]);
    }
}

// Example of adding a product to the cart
addToCart(1, 2); // Add 2 units of product with ID 1 to the cart

// Example of removing a product from the cart
removeFromCart(1); // Remove product with ID 1 from the cart
```

```
// Display the cart contents
echo "Cart Contents: ";
print_r($_SESSION['cart']);
?>
```

The above code is for adding and deleting the product into cart and from cart.

Payment:

```
<?php
// Include the Stripe PHP library
require 'vendor/autoload.php'; // Path to the Stripe PHP library

\Stripe\Stripe::setApiKey('YOUR_SECRET_KEY');

// Replace these variables with actual data
$token = 'tok_1234567890'; // Replace with the token generated by your payment form
$amount = 1000; // Replace with the actual amount in cents (e.g., $10.00)

try {
    // Create a charge
    $charge = \Stripe\Charge::create([
        'amount' => $amount,
        'currency' => 'usd',
```

```
        'description' => 'Example charge',
        'source' => $token,
    ]);

    // If the charge is successful, you can save the order and provide a success message to the user
    // Replace this with your order processing logic

    echo 'Payment successful. Thank you for your purchase!';
} catch (\Stripe\Error\Card $e) {
    // The card has been declined
    echo 'Payment failed: ' . $e->getMessage();
}
```

The above code is responsible for payment.

Conclusion:

This application is created and developed very well. This application will give nice user experience to the users. This application has various features like amazon, Flipkart.... This application is the first ecommerce application for the handmade products such as furniture, home decors. In this ecommerce application, various new features are technically implemented but these new features are not in other ecommerce

application.