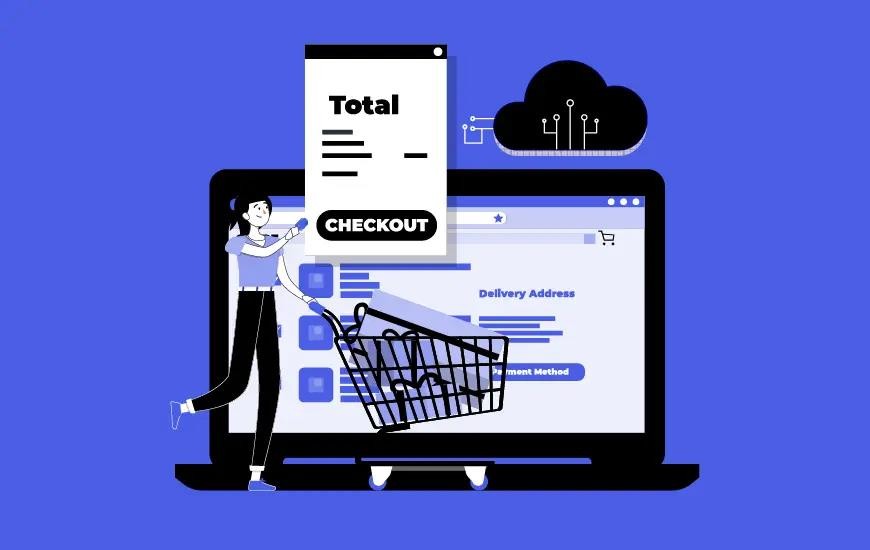
E-commerce Platform on Cloud Foundry



# Project Overview

***Submitted by***

1. *SANDIP MONDAL*
2. *SRINITHESH*
3. *SURYA*
4. *SADURDEVA*

The objective of this project is to develop a feature-rich e-commerce platform on Cloud Foundry that offers user authentication, shopping cart functionality, and a smooth checkout process. This platform will enable users to register, browse products, add items to their cart, calculate the total, and complete the purchase.

# Project Phases

* User Registration and Authentication

## Objective:

Implement user registration and authentication features to secure the platform.

## Implementation Steps:

* Choose a backend technology stack (e.g., Node.js, Python).
* Set up a user database to store user information securely.
* Create registration and login interfaces for users.
* Implement server-side authentication logic.

Code Sample (Node.js with Express - User Registration and Authentication): ‘’ JavaScript

// User registration route

app.post('/register', (req, res) => {

// Validate user input

// Hash and store the user's password securely

// Save user data to the database

// Send a confirmation email

});

// User login route

app.post('/login', (req, res) => {

// Authenticate the user

// Generate and send an access token

});

‘‘

# Shopping Cart Functionality

## Objective:

Create a shopping cart feature to enable users to add and manage items they want to purchase.

## Implementation Steps:

* Design and develop a shopping cart component in the frontend.
* Implement cart management logic in the backend.
* Enable users to add, update, and remove items in the cart.

Code Sample (Frontend - JavaScript for Shopping Cart):

``JavaScript

// Sample JavaScript code for adding an item to the cart function addToCart(item) {

// Add the item to the cart object

// Update the cart display

}

``

# Checkout Process

## Objective:

Implement a smooth and secure checkout process.

## Implementation Steps:

* Create a checkout page in the frontend.
* Calculate the total cost of items in the cart.
* Collect user shipping and payment information.
* Implement payment processing (e.g., using a payment gateway API).
* Send order confirmation emails to users.

Code Sample (Frontend - JavaScript for Calculating Total):

``JavaScript

// Sample JavaScript code for calculating the total cost function calculateTotal(cart) {

// Iterate through cart items and calculate the total

// Display the total on the checkout page

}

``

# Cloud Foundry Deployment

## Objective:

Deploy the e-commerce platform on Cloud Foundry to make it accessible online.

## Implementation Steps:

* Create a Cloud Foundry account.
* Prepare your application for deployment (e.g., create a manifest file).
* Use the Cloud Foundry CLI to push your application to the cloud.
* Monitor the application's performance and availability.

Code Sample (Cloud Foundry Manifest File - `manifest.yml`):

‘ yaml ‘

applications:

- name: your-ecommerce-app memory: 512M

buildpack: nodejs\_buildpack

**Back end integrations:**

-> we have used laravel for the back end.

//Logic to add products to cart:

function AddToCart($id) {

$product = Product::find($id);

$user\_id = Auth::user()->id;

Cart::create([

'user\_id' => $user\_id,

'product\_id'=> $product->id,

'product\_name' => $product->name,

'product\_price' => $product->price,

]);

return back();

}

//Logic to checkout

function CheckOut() {

$user\_id = Auth::user()->id;

$carts = Cart::where('user\_id', Auth::user()->id)->select('user\_id', 'product\_id', 'product\_name', 'product\_price')->get();

$array = json\_decode($carts, true);

DB::table('orders')->insert($array);

Cart::where('user\_id', Auth::user()->id)->delete();

return redirect('/home');

}

//Simple logic to display products from database:

@php

$data = \App\Models\Product::all();

@endphp

@foreach($data as $item)

<h6>{{ $item->name }}</h6>

<h5>{{ $item->price }}</h5>

@endforeach

//Logic to authenticate users

class RegisterController extends Controller

{

use RegistersUsers;

@var string

protected $redirectTo = RouteServiceProvider::HOME;

@return void

public function \_\_construct()

{

$this->middleware('guest');

}

@param array $data

@return \Illuminate\Contracts\Validation\Validator

protected function validator(array $data)

{

return Validator::make($data, [

'name' => ['required', 'string', 'max:255'],

'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],

'password' => ['required', 'string', 'min:8', 'confirmed'],

]);

}

/\*\*

\* Create a new user instance after a valid registration.

\*

\* @param array $data

\* @return \App\Models\User

\*/

protected function create(array $data)

{

return User::create([

'name' => $data['name'],

'email' => $data['email'],

'mobile' => $data['mobile'],

'address' => $data['address'],

'password' => Hash::make($data['password']),

]);

}

}

class LoginController extends Controller

{

use AuthenticatesUsers;

@var string

protected $redirectTo = RouteServiceProvider::HOME;

@return void

public function \_\_construct()

{

$this->middleware('guest')->except('logout');

}

}

# Conclusion

Successfully implementing user registration and authentication, shopping cart functionality, and a smooth checkout process, the e-commerce platform is ready to offer users a secure and convenient shopping experience. Users can register, add items to their cart, calculate the total, and complete purchases with confidence.