FLASH ZONE

DESCRIPTION:

Flash Zone is an ecommerce application there we are going to create a responsive shopping user interact application. in flashzone users can see categories of products in homepage from those user can click whatever category he/she like and redirect related products page. from those he/she can click the product to reach single product page, up to here user can see the products without login if the user is not login the single product page shows to uses to login to begin buying if the user is already login the single product page shows the user add to cart button once user press the add to cart button user can redirect to profile page.in profile page users can see their profile and users can edit and update their profiles and user can see cart symbol and how many products he/she having in their carts in profile page. By clicking cart symbol user reaches to cart page there users can see how many products he/she added and product data if users don't want any product user can press
remove button to remove the entire product and users can see total cost of the product in cart page.

Admin can add the categories to the home page and admin can add the products to the respective categories.

Tech Stacks Used:

- HTML/css used for designing the front end part
- Bootstrap is used for form data and navigation bar
- Node.js is used for backend language
- Express.js is a frame work to create server and make Api’s
- Ejs is used for render the pages from the backend
- Bcrypt is used for password encryption and compare password while login

- Multer is used for storing the form data in memory.
- Base-64 is used for converting buffer data into encoded data.
- Cloudinary is used for storing images and from cloudinary to get the url of images.
- Cors is used for different origins to access our url
- Dotenv is used for storing the secret datas like mongodb url and cloudinary secrets and express session secrets.
- Ejs-mate is used for ejs layout.
- Express-flash is used for sending the messages to users through ejs while rendering.
- Express session is used for user authentication.
- Mongodb and mongoose are used for storing the users,cart,product data and retrieving the data.
- Morgan is used as a logger.
- Connect-mongo is used as a mongo store to store the session in badabase.

Routes used:

User:

- **get('/')-**used for rendering the home page from backend there users can see the categories.
- **get('/signup)-** used for rendering the signup page from backend.
- **post('/signup')**-by using this route to store the user data in database and store the user id in cart for identify the cart is belongs to user.here user can give unique email if user gives already existing email he/she will gets user is already exist once he/she gives valid details he/she redirects login page.
- **get('/login')**-used for rendering the login page from backend.

- **post(‘/login’)**-in this route user is entering the valid email and password if user enters valid details he/she redirect to profile page if the user gives invalid details he/she gets message as invalid username and password.in this route we are going to check the user email with database email and compare bcrypt password with user enter password if both are matches store emailID in express session.
- **get(‘/profile’)-** in this route users can see their profiles.check the user mail with express session mail if it is matches store the user data in profile and render the profile page with the help of user data. In this route initially set cart count by looping to entire cart items and rendering this count along with user data and displays to the user.
- **get(‘/editprofile’)-**in this route store the user data and render the editprofile page to the user with the userdata.
- **post(‘/editprofile’)-**in this route check if session is present if presents update the user data in database by using multer and cloudinary and redirect to the profile page.
- **get(‘logout’)-** by using this route to user can logout.

Category :

- **get(‘/addcategory’)-**by using this route to render the addcategory page. Admin only access to the page.
- **post(‘/addcategory’)-**by using this route admin can the categories.

Product:

- **get(‘/addproduct’)-**by using this route to render the add product page.
- **post(‘/addproduct’)-**by using this route admin can add the products once admin add the products whatever he/she added all are goes to respective category.

- **get(‘/cars’)-**by using this route store the user details in user variable by checking email is session email and if session is present check with cart ownerobect id with and user id it it is maches store the cart data in cart variable and run the loop entire cart and to store the total quantity in cart. The main thing is by using find method to check all the products with category of car id by using this to store all the cars in cars variable and while rendering the car page passes user details and total quantity in cart and total car data.
- **get(‘/bikes’)-**by using this route to store all the bikes data in bikes variable by using find method from all products with category of bikes object id.while rendering bikes pages along with bikes page send user data,total cart quantity and bikes data in bikes.ejs page run the loop to the entire bikes and each bike gives the link as bikes/<%=bikes[i]_.id%> to redirect to single product page.
- **get(‘/cars/:id’)-**by using this route to find each car by the method of find by id and passing this link to bikes page while running the loop of entire products to redirect to a single car page.
- **get(‘/bike/:id’)-**by using this route to find each bike by the method of find by id and passing this link to bikes page while running the loop of entire products to redirect to a single bike page. Here we render the bike data as a bike.ejs and display bike details likelbike.image, bike.name, bike.price and and create another form to post the bike details for storing each bike data in cart while clicking add to cart button here we gave condition if user is login he able to see the button as add to cart if user is not login user sees login to begin buying.
- **post(‘/addtocart’)-**in single page product we create a form to store the product data if user presses add to cart button if user presses add to cart button either bike or car data is saved in cart collection.by using cart.items.push method to the product data

into cart and and total price is also added with actual price to added product price.

- **get(‘/cart’)-** in this route by using cartmodel.findone method to check with user id to cart id if it matches populate the products in items in cart and render all products data to cart page in cart page run the loop to the entire products to display each product image price quantity and total cost.
- **post(‘/remove’)-**in cart page create another form while displaying each product in that form we mention the remove button while user press the remove button that product will be removed for that we created in postmethod find the user id with cart id if it matches by using cart.items.pull method to pull the product with the help of product id and total aslo deducted with total to price of the product.

FeatureScope:

In feature we are planning to add payment feature by using stripe, if user presses the pay button he/she redirected to payment page once user pay the payment he/she will receive notification as you payment is success by using socket.io and also add admin can approve the orders and user can see order status.