**Dynamic Routes & Advanced Models**

Now we have to pass some data anusing them dynamically such as the product id so we can call that thing by their id or send by id or sending the data using url’s

So to add an id you have to add id by using Math.random() which will give random number

You have to pass id with this.id in the product class during the saving time

 save() {

**this.id = Math.random().toString();**

    getProductsFromFile(products => {

      products.push(this);

  <div class="card\_\_actions">

**<a class="btn" href="/products/<%= product.id  %>">Details</a>**

Then you have to pass the link in detail link so it opens the page with product id

* Dynamic routs in node.

Dynamic routs represented by /:someName it is used to save the the path after (/) in the :some name

Suppose you are passing some **id (/product/4586**) then :productId will take that 4586

router.get('/products**/:productId'**, shopController.getProduct);

so it will take any thing if you pass after /product/anything but notice if you pass /delete after it it will note open the delete file if you have any in case it will saves it as Dynamic

**if you have as such file present make shore you write it before and at last write the dynamic /:product because if you do that then only file read the delete other wise it will read /:product first because rout move top to bottom**

router.get('/products/delete', shopController.getProduct);

router.get('/products/:productId', shopController.getProduct);

and we can extract these productId in shopController.getproduct by use of params which can take the params of the path

exports.getProduct = (req,res,next) => {

**const proId =  req.params.productId;**

  console.log(proId);

  res.redirect('/');

}

So if we want to get particular element by id so we have to create that element in our product class in our model

So we create a function which takes the id and return the call back function containing particular element.

 static findById(id,cb) { // cb is call back which will get by the method which calling it

    getProductsFromFile(products => { // products is array getting from this function above

      const product = products.find(p => p.id === id); // find while find the particular element from array

      cb(product);

    });

  }

So we call cb and passing our particular product with that we are getting product as call back which we are console.log in screen

exports.getProduct = (req,res,next) => {

  const proId =  req.params.productId; // it stores the product id getting by /product/:productId

  Product.findById(proId, product => {

    console.log(product);

  })

  res.redirect('/');

}

So now we create our product detail page and pass a product array while render

exports.getProduct = (req,res,next) => {

  const proId =  req.params.productId; // it stores the product id getting by /product/:productId

  Product.findById(proId, product => {

    res.render('shop/product-detail.ejs',{

      product: product,

      pageTitle: product.title,

      path: '/product'

    });

  });

Now we create a form button add to cart so when ever we press we redirect to the cart page and latter on we can add the detail of the product init

<form action="/cart" method="POST"> <!-- we can pass the data using post request-->

    <button class="btn" type="submit">Add to cart</button>

    <input type="hidden" name="productId" value="<%= product.id %>">

    <!-- it will pass the value by post request in method-->

</form>

So we are using method post so the value in input can bet get by post method In the router by the key as name (productId)

exports.postCart = (req,res,next) => {

  // we get it from input value inside form method post

  const prodId = req.body.productId; //we can get the value during method post

  console.log(prodId);

  res.redirect('/cart');

}

router.post('/cart', shopController**.postCart**);

now we have to store the data of cart which we are going to save in **cart.json inside model**

**here we check the product if it is present then update the quantity only but if not present then update the cart by adding new product in** it

// we have to collect and save the cart detail so we hav e to create a model for that

const fs = require('fs');

const path = require('path');

const p = path.join( // file path where we want to read or write

    path.dirname(process.mainModule.filename),

    'data',

    'cart.json'

);

module.exports = class cart {

    static addProduct(id,productPrice) {

        //fetch the previous product

        fs.readFile(p, (err,fileContent) => {

            // creating cart so that can pass it in form of json

            let cart = {products: [], totalPrice: 0}

            if(!err) {

                cart = JSON.parse(fileContent); // add previous file content

            }

            // find the index of that id so if found we can update the quantity if not then update the json file

            const existingProductIndex = cart.products.findIndex(p => p.id === id);

            const existingProduct = cart.products[existingProductIndex];

            let updatedProduct;

            if(existingProduct) { // if not null then increase the quantity by one

               updatedProduct = {...existingProduct }

               updatedProduct.qty += 1;

               cart.products = {...cart.products}

               cart.products[existingProductIndex] = updatedProduct;

            }

            else { // if their is no product such like that present in data then update the data

                updatedProduct = {id: id, qty: 1}

                cart.products = [...cart.products , updatedProduct];

            }

            cart.totalPrice = cart.totalPrice + +productPrice; // increase the price in form of integer by + +

            fs.writeFile(p,JSON.stringify(cart), (err) => {

                console.log(err);

            })

        })

    }

}

Which we calling this from our shop controller from shop.js inside controller folder

exports.postCart = (req,res,next) => {

  // we get it from input value inside form method post

  const prodId = req.body.productId; //we can get the value during method post

  Product.findById(prodId, (product) => {

    cart.addProduct(prodId, product.price); //it will run once we get product from callBack

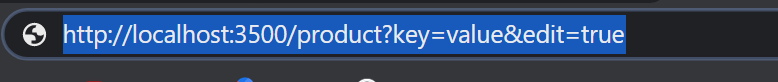
  })

  res.redirect('/cart');

}

* **Query Parameters in node**

Query parameters are the key value pair which can be used after ? in our url



<http://localhost:3500/product?key=value&edit=true>

here we can add key value using dynamic content in url by example => /:product

an we can pass it in our ejs file so we pass some key value pair also which can be used when we need some dynamic data



if admin wants to edit the page the admin press edit button so after pressing the edit button admin should get the old product detailed filled up on the page

and then he can edit the data and when press update button the data should be update on the memeory can do it by get nd post the data by admin.js

// when admin wants to edit/update the product admin will get the detail of that product in the page

exports.getEditProduct = (req, res, next) => {

  const editMode = req.query.edit;

  if (!editMode) {

    return res.redirect('/');

  }

  const prodId = req.params.productId;

  Product.findById(prodId, product => {

    if (!product) {

      return res.redirect('/');

    }

    res.render('admin/edit-product', {

      pageTitle: 'Edit Product',

      path: '/admin/edit-product',

      editing: editMode,

      product: product

    });

  });

};

// when admin change something from get detail content it will save in same index with update

exports.postEditProduct = (req, res, next) => {

  const prodId = req.body.productId; // id get from edit-product.ejs from hidden input

  const updatedTitle = req.body.title;

  const updatedPrice = req.body.price;

  const updatedImageUrl = req.body.imageUrl;

  const updatedDesc = req.body.description;

  const updatedProduct = new Product(

    prodId,

    updatedTitle,

    updatedImageUrl,

    updatedDesc,

    updatedPrice

  );

  updatedProduct.save(); // update the the new data of product

  res.redirect('/admin/products');

};

So to save file then data check for the product is updated or the product is new one so in our save function if we the id is null that means the data is new otherwise data is updated

save() { // here it is check the product is new or you are editing the previous

    getProductsFromFile(products => {

      if (this.id) { // if id present that means the product is old one and you want to edit

        const existingProductIndex = products.findIndex(

          prod => prod.id === this.id

        );

        const updatedProducts = [...products];

        updatedProducts[existingProductIndex] = this; // passing updated this value in th updated index

        fs.writeFile(p, JSON.stringify(updatedProducts), err => {

          console.log(err);

        });

      } else { // if the id is null therefore creating the random id

        this.id = Math.random().toString();

        products.push(this);

        fs.writeFile(p, JSON.stringify(products), err => {

          console.log(err);

        });

      }

    });

Now we have to show our items in cart with the help of cart data we can add the element in our cart but first we have to create our cart page

  <body>

        <%- include('../includes/navigation.ejs') %>

        <main>

            <% if (products.length > 0) { %>

                <ul class="cart\_\_item-list">

                    <% products.forEach(p => { %>

                        <li class="cart\_\_item">

                            <h1><%= p.productData.title %></h1>

                            <h2>Quantity: <%= p.qty %></h2>

                            <form action="/cart-delete-item" method="POST">

                                <input type="hidden" value="<%= p.productData.id %>" name="productId">

                                <button class="btn danger" type="submit">Delete</button>

                            </form>

                        </li>

                    <% }) %>

                </ul>

            <% } else { %>

                <h1>No Products in Cart!</h1>

            <% } %>

In the cart page we run for loop in the product array which is present in cart.JSON and we get our data from there and and display inside for loop

And to show the data in our ejs file we have to pass our data by the use of model to controller so that controller can send that data to the view

// it is used to pass the cart.json data into shop.js controller

  static getCart(cb) {

    fs.readFile(p, (err, fileContent) => {

      const cart = JSON.parse(fileContent);

      if (err) {

        cb(null); // means cart.json is empty

      } else {

        cb(cart);

      }

    });

  }

Now we call that get cart in our controller shop.js

exports.getCart = (req, res, next) => {

  Cart.getCart(cart => {

    Product.fetchAll(products => { // fetching the particular product to cart

      const cartProducts = [];

      for (product of products) {

        const cartProductData = cart.products.find(

          prod => prod.id === product.id

        );

        if (cartProductData) {

          cartProducts.push({ productData: product, qty: cartProductData.qty });

        }

      }

      res.render(**'shop/cart'**, {

        path: '/cart',

        pageTitle: 'Your Cart',

        products: cartProducts // contains the cart data which will shown in cart.ejs page

      });

    });

  });

};

So the data will render throw **shop/cart.ejs**