

9. Dynamic Routes & Advanced Models

* Chapter 111: Dynamic Routes & Advanced Models



What's In This Module?

Passing Route Params

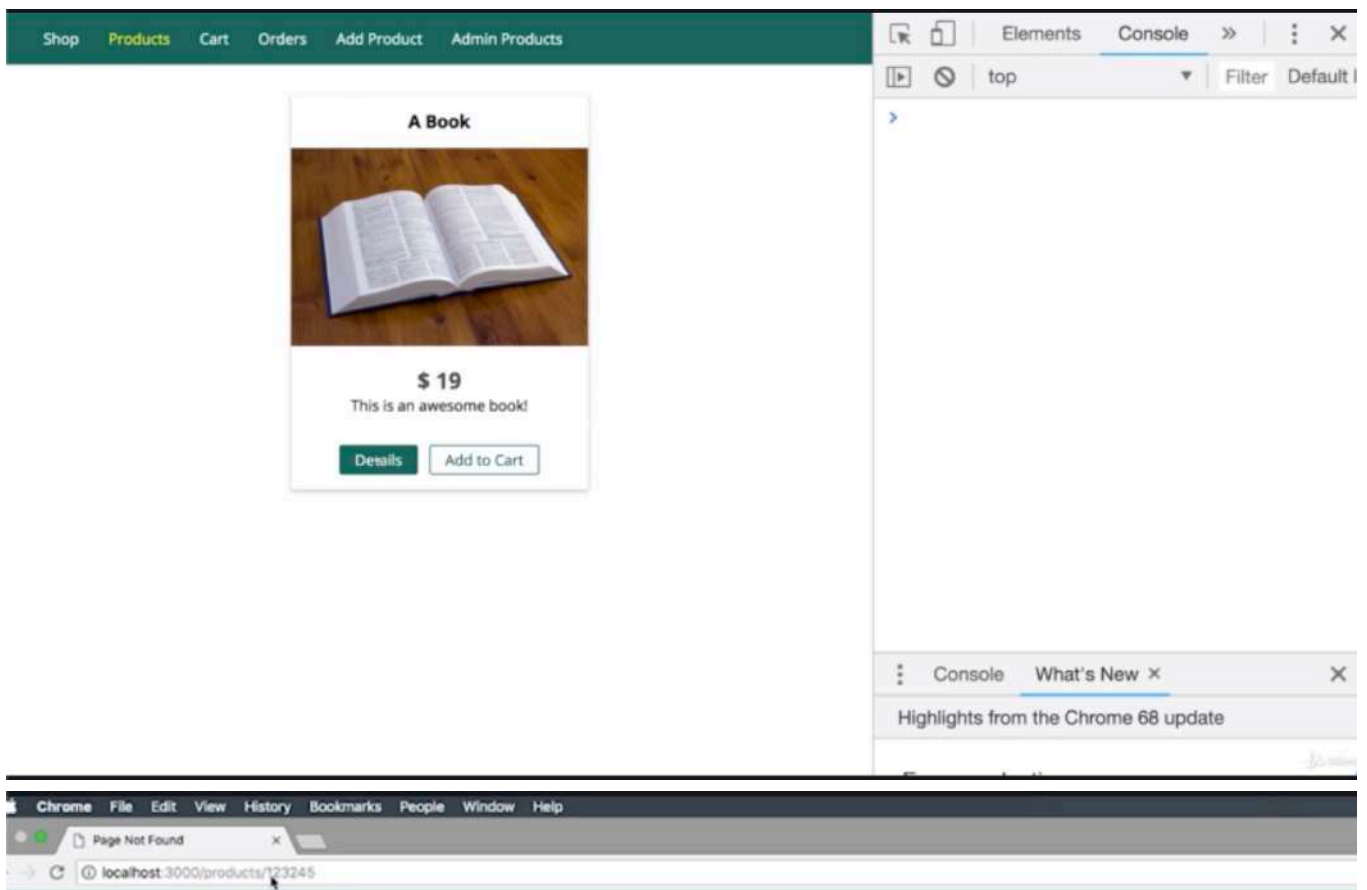
Using Query Params

Enhance our Models

Academind

* Chapter 114: Adding The Product ID To The Path

- 1. update
 - ./views/shop/product-list.ejs
 - ./models/product.js
 - ./data/products.json



Page Not Found!

```
1 <!--./views/shop/product-list.ejs-->
2
3 <%= include('../includes/head.ejs') %>
4   <link rel="stylesheet" href="/css/product.css">
5   </head>
6
7   <body>
8     <%= include('../includes/navigation.ejs') %>
9
10    <main>
11      <%= if (prods.length > 0) { %>
12        <div class="grid">
13          <%= for (let product of prods) { %>
```

```

14 <article class="card product-item">
15   <header class="card_header">
16     <h1 class="product_title">
17       <%= product.title %>
18     </h1>
19   </header>
20   <div class="card_image">
21     ">
22   </div>
23   <div class="card_content">
24     <h2 class="product_price">$
25       <%= product.price %>
26     </h2>
27     <p class="product_description">
28       <%= product.description %>
29     </p>
30   </div>
31   <div class="card_actions">
32     <!--/product/0.43432 would be one possible path
33     now we wanna make sure that we are able to handle that
34     and then extract that unique ID from the path in our
routes file.
35     so that in the controller, we can load the correct
product and how the details for it.
36
37     we send some information as part of the path.
38     so that we can extract all the data we need for the
product from the controller or inside of the controller
39     because we can't really send the entire product as part
of the URL
40     but we can send this key information.
41     -->
42     <a href="/products/<%= product.id %>"
class="btn">Details</a>
43     <form action="/add-to-cart" method="POST">
44       <button class="btn">Add to Cart</button>
45     </form>
46   </div>
47 </article>
48 <% } %>
49 </div>
50 <% } else { %>
51   <h1>No Products Found!</h1>
52 <% } %>
53 </main>
54 <%= include('../includes/end.ejs') %>

```

```

1 //./models/product.js
2
3 const fs = require('fs');
4 const path = require('path');
5
6 const p = path.join(
7   path.dirname(process.mainModule.filename),
8   'data',
9   'products.json'

```

```

10 );
11
12 const getProductsFromFile = cb => {
13   fs.readFile(p, (err, fileContent) => {
14     if (err) {
15       cb([]);
16     } else {
17       cb(JSON.parse(fileContent));
18     }
19   });
20 };
21
22 module.exports = class Product {
23   constructor(title, imageUrl, description, price) {
24     this.title = title;
25     this.imageUrl = imageUrl;
26     this.description = description;
27     this.price = price;
28   }
29
30   save() {
31     this.id = Math.random().toString();
32     getProductsFromFile(products => {
33       products.push(this);
34       fs.writeFile(p, JSON.stringify(products), err => {
35         console.log(err);
36       });
37     });
38   }
39
40   static fetchAll(cb) {
41     getProductsFromFile(cb);
42   }
43 };
44

```

```

1 //./data/products.json
2

```

```

3 [{"id": "123456", "title": "A
Book", "imageUrl": "https://www.publicdomainpictures.net/pictures/10000/velka/1-
1210009435EGmE.jpg", "description": "This is an awesome book!", "price": "19"}]

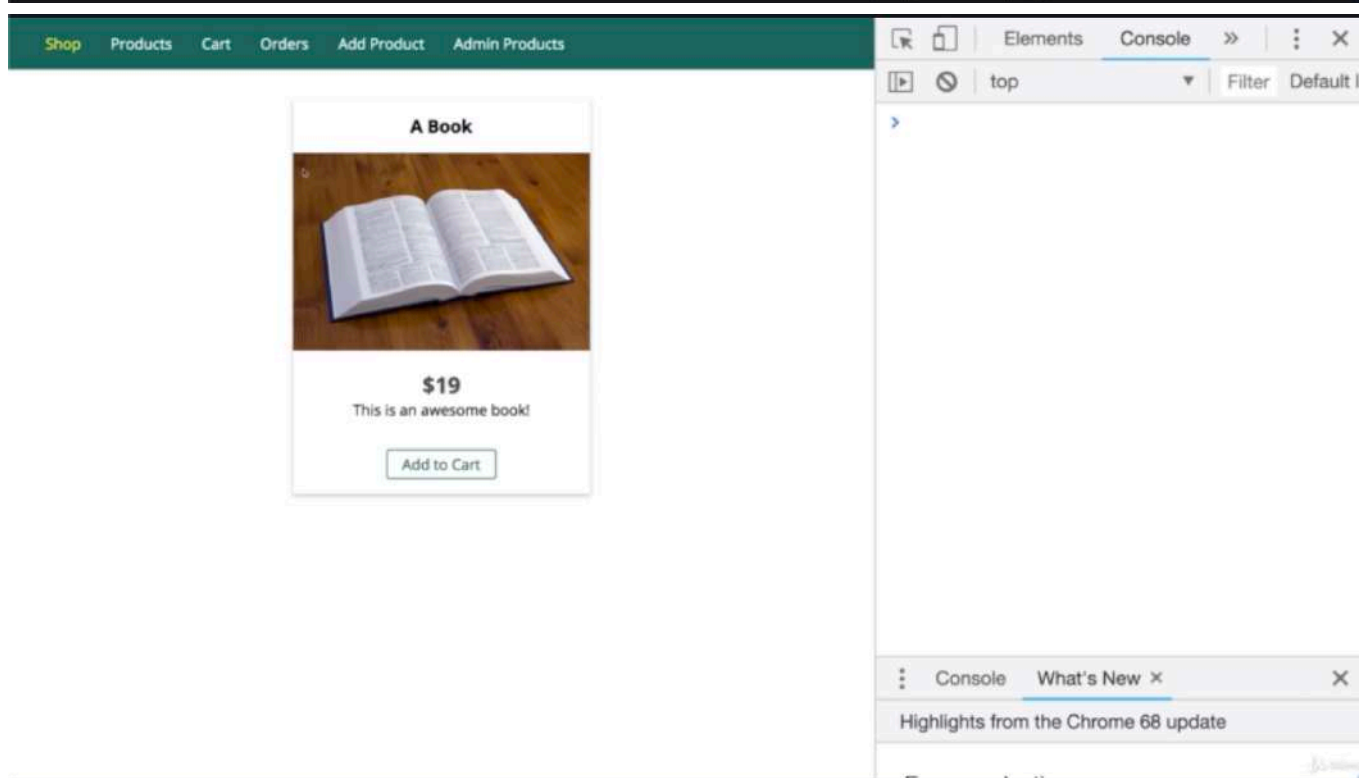
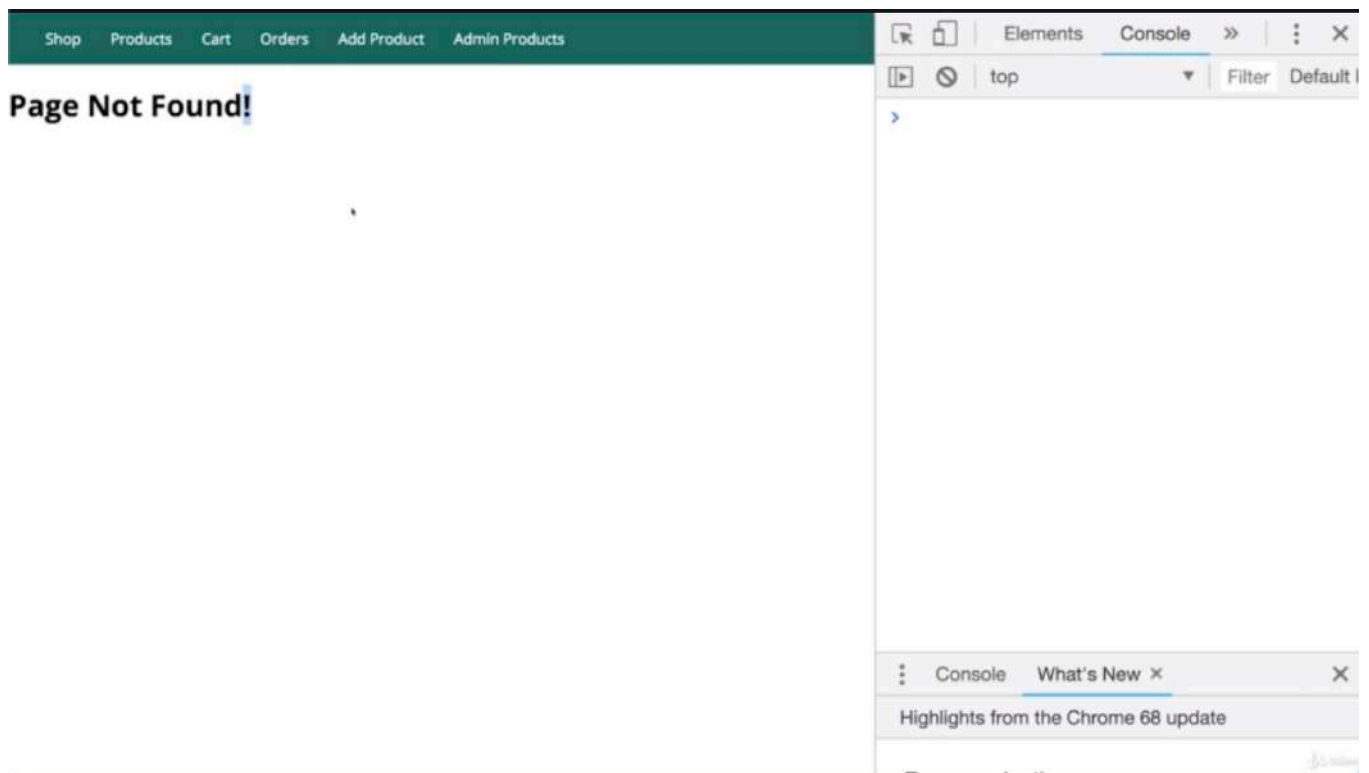
```

* Chapter 115: Extracting Dynamic Params

1. update

- ./controllers/shop.js

- ./routes/shop.js



- if i reload this, i'm getting redirected which means we handle this. we don't get the 404 page anymore.

The screenshot shows the VS Code interface. The Explorer on the left shows a project structure with folders like controllers, data, models, and routes. The main editor shows the file `shop.js routes` with the following code:

```
1 const path = require('path');
2
3 const express = require('express');
4
5 const shopController = require('../controllers/shop');
6
7 const router = express.Router();
8
9 router.get('/', shopController.getIndex);
10
11 router.get('/products', shopController.getProducts);
12
13 router.get('/products/:productId', shopController.getProduct);
14
15 router.get('/cart', shopController.getCart);
16
17 router.get('/orders', shopController.getOrders);
18
19 router.get('/checkout', shopController.getCheckout);
```

The Terminal at the bottom shows the output of a nodemon command, indicating that the application is restarting due to changes and starting successfully.

```
1 //../controllers/shop.js
2
3 const Product = require('../models/product');
4
5 exports.getProducts = (req, res, next) => {
6   Product.fetchAll(products => {
7     res.render('shop/product-list', {
8       prods: products,
9       pageTitle: 'All Products',
10      path: '/products'
11    });
12  });
13 };
14
15 exports.getProduct = (req, res, next) => {
16   /**we can get access to it by accessing our request
17    * and then express.js already gives us a 'params' object on our request
18    * and then on that params object,
19    * we can access our productId
20    * and we can access productId here
21    * because we use productId in our ./route/shop.js file as a name after the colon ':'
22    *
23    */
24   const prodId = req.params.productId
25   console.log(prodId);
26   res.redirect('/')
27 }
28
29 exports.getIndex = (req, res, next) => {
30   Product.fetchAll(products => {
31     res.render('shop/index', {
32       prods: products,
33       pageTitle: 'Shop',
34       path: '/'
```

```

35     });
36   });
37 };
38
39 exports.getCart = (req, res, next) => {
40   res.render('shop/cart', {
41     path: '/cart',
42     pageTitle: 'Your Cart'
43   });
44 };
45
46 exports.getOrders = (req, res, next) => {
47   res.render('shop/orders', {
48     path: '/orders',
49     pageTitle: 'Your Orders'
50   });
51 };
52
53 exports.getCheckout = (req, res, next) => {
54   res.render('shop/checkout', {
55     path: '/checkout',
56     pageTitle: 'Checkout'
57   });
58 };
59

```

```

1 // ./routes/shop.js
2
3 /**The order would matter.
4 *
5 *   router.get('/products/:productId')
6 *   router.get('/products/delete')
7 *
8 * keep in mind that your code is parsed from top to bottom
9 * and the request goes through that from top to bottom.
10 * if you order it like this, you would never reach that route
11 * because if you had a route like '/products/delete',
12 * express.js would already fire at '/products/:productId'
13 * or would already handle it in '/products/:productId'
14 * because delete would basically be treated as the dynamic segment.
15 *
16 *   router.get('/products/delete')
17 *   router.get('/products/:productId')
18 *
19 * so if you had a dynamic segment and a specific route,
20 * you would have to put the more specific route first.
21 * so that for '/products/delete', this handles the request
22 * and thereafter it will not continue its journey
23 * because you don't fire next.
24 * but if you then have something else which doesn't match '/products/delete',
25 * then you would go into that dynamic routes
26 */
27 const path = require('path');
28
29 const express = require('express');
30
31 const shopController = require('../controllers/shop');

```

```

32
33 const router = express.Router();
34
35 router.get('/', shopController.getIndex);
36
37 router.get('/products', shopController.getProducts);
38
39 /**we can tell the express router that there will be some variable segment by adding a colon
40  * and then any name of our choice like 'productId'
41  * later we will be able to extract that information by that name here.
42  *
43  * the important part is the colon ':'
44  * this signals to express that it should not look for a route
45  * like '/products/:productId' but only this part ':productId'
46  *
47  */
48 router.get('/products/:productId', shopController.getProducts);
49
50 router.get('/cart', shopController.getCart);
51
52 router.get('/orders', shopController.getOrders);
53
54 router.get('/checkout', shopController.getCheckout);
55
56 module.exports = router;
57


```

* Chapter 116: Loading Product Detail Data

1. update
 - ./models/product.js
 - ./controllers/shop.js
 - ./data/products.json

ShopProductsCartOrdersAdd ProductAdmin Products

A Book



\$ 19

This is an awesome book!

DetailsAdd to Cart

ElementsConsole>>⋮×

topFilterDefault


>

⋮ ConsoleWhat's New ××

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

A Book



\$19

This is an awesome book!

Add to Cart

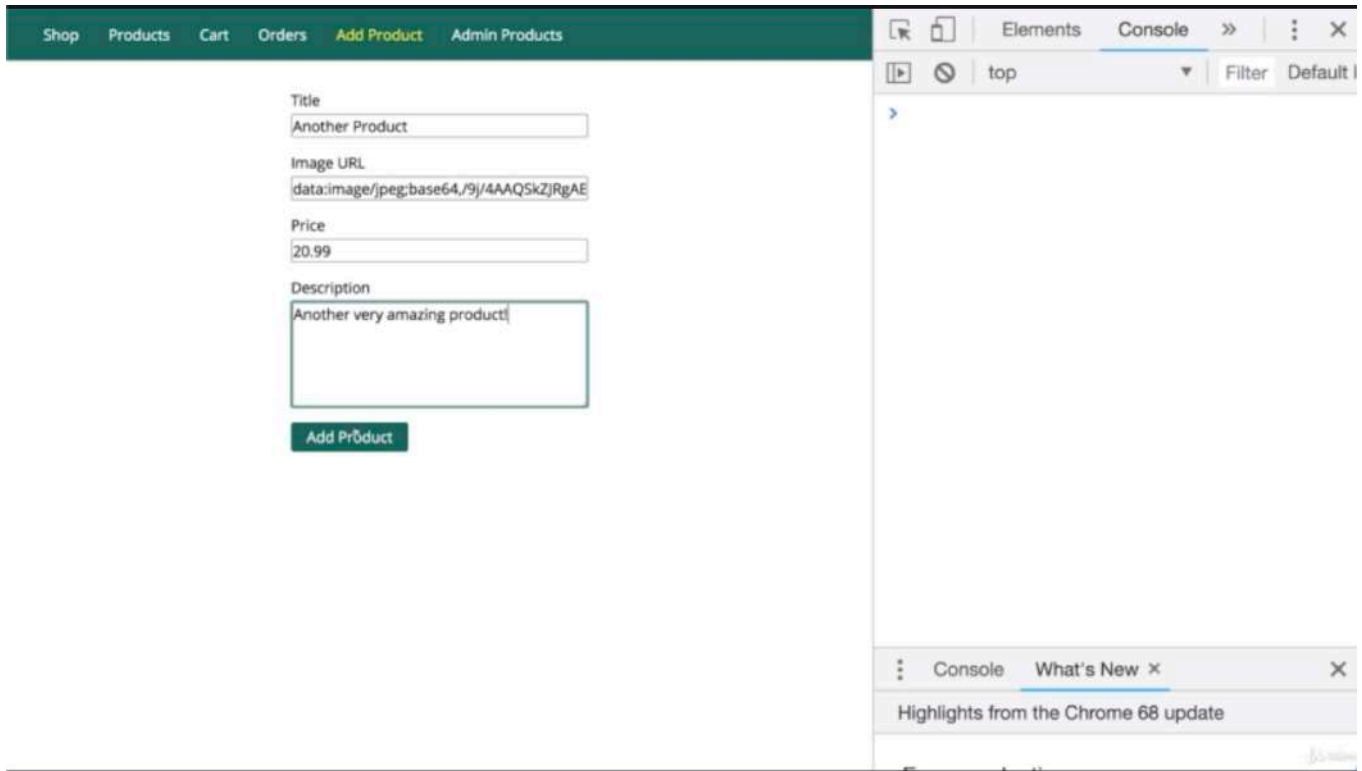
ElementsConsole>>⋮×

topFilterDefault

>


⋮ ConsoleWhat's New ××

Highlights from the Chrome 68 update



[Shop](#) [Products](#) [Cart](#) [Orders](#) [Add Product](#) [Admin Products](#)

A Book




\$19

This is an awesome book!

Add to Cart

Another Product



\$20.99

Another very amazing product!

Add to Cart

Elements

Console

top

Filter

Default

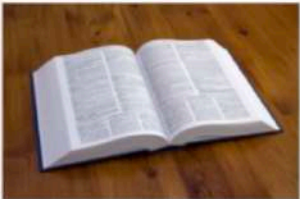
Console

What's New

Highlights from the Chrome 68 update

[Shop](#) [Products](#) [Cart](#) [Orders](#) [Add Product](#) [Admin Products](#)

A Book




\$ 19

This is an awesome book!

Details

Add to Cart

Another Product



\$ 20.99

Another very amazing product!

Détails

Add to Cart

Elements

Console

top

Filter

Default

Console

What's New

Highlights from the Chrome 68 update

```

1  const Product = require('../models/product');
2
3  exports.getProducts = (req, res, next) => {
4    Product.fetchAll(products => {
5      res.render('shop/product-list', {
6        prods: products,
7        pageTitle: 'All Products',
8        path: '/products'
9      });
10   });
11 };
12
13 exports.getProduct = (req, res, next) => {
14   const prodId = req.params.productId;
15   Product.findById(prodId, product => {
16     console.log(product);
17   });
18   res.redirect('/');
19 };

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: node

```

Xwh0ZK0ryUr283VsSNxk8ypNay46uVt74tPtTK0ld1XUcatFw3lp5UezPnLg0dZUWqezJgqGeE6TWI1Itrqyk/Yz502fYVfFI8omu1l2H2Nm+w0B
A+zlsJcbXG813FajHY6kW/ox6Uv2Y5ZFQayb2E3R9V0pQUfnKkoxpry2sy4RMX6u8i01cV56lvQbfX0a3d+onn9pxNx5LcLP7PLn683VrNeU8Ygv
opLYl3L3vaS0bXhcF0DkoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAARqwo1fnKcJ43a0E2uD6iSAKS45L2s/Nkv1tb3T1seogT5E0PMk4/qtfwSib
UC7RqP/AAUuq4qL9e4/qnqPIuPXxm+Mq/BAVNsA/VGSR5F2/n5k/wBWL/j11wt+TdpD9Frek21+z5PuLcDaMdKjGKxGK1luSSSXqMgBAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAH/9k=
description: 'Another very amazing product!',
price: '20.99',
id: '0.558673316244021' }

```

Ln 16, Col 26 Spaces: 2 UTF-8 LF JavaScript Prettier: ✓

```

1  //./models/product.js
2
3  const fs = require('fs');
4  const path = require('path');
5
6  const p = path.join(
7    path.dirname(process.mainModule.filename),
8    'data',
9    'products.json'
10 );
11
12 const getProductsFromFile = cb => {
13   fs.readFile(p, (err, fileContent) => {
14     if (err) {
15       cb([]);
16     } else {
17       cb(JSON.parse(fileContent));
18     }
19   });
20 };
21
22 module.exports = class Product {
23   constructor(title, imageUrl, description, price) {
24     this.title = title;
25     this.imageUrl = imageUrl;
26     this.description = description;
27     this.price = price;
28   }
29
30   save() {
31     this.id = Math.random().toString();
32     getProductsFromFile(products => {
33       products.push(this);
34       fs.writeFile(p, JSON.stringify(products), err => {

```

```

35     console.log(err);
36   });
37   });
38 }
39
40 static fetchAll(cb) {
41   getProductsFromFile(cb);
42 }
43
44 static findById(id, cb){
45   getProductsFromFile(products => {
46     /**'find()' will execute a function we pass to find on every element in the array
47      * and we will return the element for which this function we pass returns true.
48      *
49      * there is a short arrow function syntax where you can omit the curly braces
50      * if you only got 1 line in there and you also return the result of this 1 line,
51      * so there is a implicit return statement in front of that code,
52      * i will now write.
53      */
54     const product = products.find(p => p.id === id)
55     cb(product)
56   })
57 }
58 };

```

```

1  //./controllers/shop.js
2
3  const Product = require('../models/product');
4
5  exports.getProducts = (req, res, next) => {
6    Product.fetchAll(products => {
7      res.render('shop/product-list', {
8        prods: products,
9        pageTitle: 'All Products',
10       path: '/products'
11     });
12   });
13 };
14
15 exports.getProduct = (req, res, next) => {
16   const prodId = req.params.productId
17   Product.findById(prodId, product => {
18     console.log(product)
19   })
20   res.redirect('/')
21 }
22
23 exports.getIndex = (req, res, next) => {
24   Product.fetchAll(products => {
25     res.render('shop/index', {
26       prods: products,
27       pageTitle: 'Shop',
28       path: '/'
29     });
30   });
31 };
32

```

```

33 exports.getCart = (req, res, next) => {
34   res.render('shop/cart', {
35     path: '/cart',
36     pageTitle: 'Your Cart'
37   });
38 };
39
40 exports.getOrders = (req, res, next) => {
41   res.render('shop/orders', {
42     path: '/orders',
43     pageTitle: 'Your Orders'
44   });
45 };
46
47 exports.getCheckout = (req, res, next) => {
48   res.render('shop/checkout', {
49     path: '/checkout',
50     pageTitle: 'Checkout'
51   });
52 };
53

```

```

1 [{"id":"123245","title":"A
Book","imageUrl":"https://www.publicdomainpictures.net/pictures/10000/velka/1-
1210009435EGmE.jpg","description":"This is an awesome book!","price":"19"},{"title":"Another
Product","imageUrl":"data:image/jpeg;base64,/9j/4AAQSkZJRgABAQAAQABAAD/2wCEAAkGBw8QDxUQEAP
EhUPFRAVEBAWDQ8WFRAQFhUWFhURFRYaHiggGBoLHRUVITEhJSkrLi4uFx8zODMtNygtLisBCgoKDQ00FxAAPFy0ZFRkr
LS0rKy0tKy0rLS0rKzcrLS0rNy0rLTctKysrLS03Kys3LSsrLSs4KysrKys3KystK//AABEIALcBFAMBIgACEQEDEQH/
xAAcAAEAAgMBAQEAAAAAAAAAABAUDBgCAQj/xABJEAAQAMABQCHBwkGBwAAAAAAAAAQIDBBEFEiExcQYTIkFRYbEy
QnKBkaHBiZNSU9Lh8AckQ4KS1LLR0xQVRLPC8RY0YoSFk6L/xAAXAQEBAQEAAAAAAAAAAAAAAAAAQID/8QAGhEBAQEB
AQEBAAAAAAAAAAAAAAAAAAERITECEv/aAAwDAQACEQMRAD8A7iAAAAAAAAAAAAAAAAAGKtcQhjXlG0diy0svsAygwK7pN6qqQbe
5a8cvgivr6foxeHpS29Bzpwls7ptFwW4K2npGcv0cFxuKfwe5aQkmlze/rU00uIwTwRFePsh66uPgeLdPsj/7F/IYJ
II0rtJZazjqUk36l1niWkYqKk4VNqTwoptdzim3kYJgIn940tXW10u6XRl7JYZ4paXt5PHORT7JbPEZROB4p1Yy8mUXj
fhp4PZAAAAAAAAAAAAAAAAAAAAAAAAAAACK5WVSpQ5yjkSqUHRwipNKoltlTmlslFpbmXZ8k0ltxjr4Fgo6ejba5
oRrUYRpSqKMoZjFJqSetqzS8pKS2p9hrHLXSM62jmsSVahW5utG0cxmk9qxtw1hp95c6P0iqCrTowq1rdzzCUdTEZvP0
qLb6UcqlYs7ZSXUaryiv3/aVdQ1qcK6ar03GTUo01rRrbFsaWsnnuN/M6lado+vXTf0c41h7Ja/xPs79rzvA69yaq89R
UsvDWzvXBnvSnJy2rL5SnTk+10qefbjJbekcQu9L1I+TNR1lLlLdp7KzXt0kaa5CUNrp9H2NexlRbcInGW2NtNf9VnQ
l4xElXY1SlylvX/iJe0z10UN3q/8xL/5/kdGt0T1HHS7F/+Pt/smWroWgt1taLhZ26/0kvzSfUQvy0tXX0TuFCrKm+i
5Qg8bu7v0o3dGjQn1IU9WCb6UI4il4H0LStK2nGdFRjqvbGMYxTXWmlsLm709Tv6tGzjKVNVCzr0UWLm5R+RjJ7G3ly
2fRRL8prYNALW5ysoKEKjSpLVScqcVjXfc+ruXew55hFJJYSSSYl1HoxVAAAAAAAAAAAAAAAAAAAAAAAAAAAAhaY
VN0Jqq5KGFruLaerlZWtw9zx1MmlZyiadtVjLzD0bS62lJL969pZ6NXvNMU7hxo28oc1DCUI4WxblhbkZLezq3c1TqU
XTToXSk8p1VLpmsdaeFltYxlbdb080qNJ52b0stbTTLxTjInXWjjqWwPZk7Wc4zHYaNF00I4SXUQryt2M5bV5XXy3X
dRcVTfiivu0WF+v8Snxo0/gjEiuk3M8keCjnq0W1eXN+v0PjS+88R/KDfLzbV8aMvtGtR2a2jHG9Hm4pR7Uclp/LHvf
q7L92b8ZGdflDvWvKto8L0n8chW9XtFb0yE6CnHUKnjKakt8JLd0L7V9z2M0i65X3099y13Ro0I+ESlnfVqtRc5Wqzy1
sdSTW/s3E1cFpfRVxzLGMtbXeEpS1cZkt7x1Eso+RscWVPgy80d9AAEAAAAAAAAAAAAAAAAAAAAAAAAAAACv01YwrUmp
eapPdF52PK27uKwywMN383P0ZeDA4vcaItZYkp160erKcVwT1m/ajGtAJ7IXec7ta3x71Nv3FpWp51T1e4TS4Y09Za3c
8nLlbqvtLu/OE/8ALKu40Be9UIPhVS/iSN8gtmTXrlZk1395jWmo19BXv1Uf3ih9owf3HefVQ/erb7Zh0xVqKtNRqTS1
ntU5rBdndVXj5Wrv+tL4DRBU+T14/wBFT/e7X7ZLp8mrz6FJf9xB/wAOTDofnJLpSLd57b/AJm32tBamfFsuI16HJ27
f1K4yr/0zJR0DqVoQrXMIzm0uU4Qk9Z+nh6vriBtG0wobyknpCk35uGtuMPK+DZnetWcde5I6Mlb0FrVHNzSe1t46uvg
tyS7kXpG0csUYeiSZqAAIAAAAAAAAAAAAAAAAAAAAAAAAAAGG8+bn6M/BmYw3nzc/Rn4MDmNaGnUX/LJc0ottJ22J
LHVj3YKu9XTR3ZeZqVqeo1yulrPq7dvXNjj5BrLzHPt7DFac/wBK556fX0nj7y0oZeM/yJulIvnZ4T8p7tifduI8Eu1r
btXb2AbBoeCcd2cY61tNysqXyaNV0NDZjhxSN2s4dBGkZcFHfLF9TX0tnVu3/A2LBS38Pz6g/wAbpGY1fHZbL5qHox8D
0YbRfJx9GPgZjmgAAAAAAAAAAAAAAAAAAAAAAAAAG0v5EvRl4GQx3HkS4PwA0m/WZFLfLpewur3GsVF2ul7DsY8J
dA125XH2Gyyjima9XjtIrRdJU3zkvSb6/vPFCk0/X2E+8pPXl6T61uPLcj7uLEgu9D2+7CNvpR2JFDoajhZ4GxwWxGh
9ZWXCm3NB98l7mWkiLKGasH9GUX4r4mVdWoLoR4R8DIeafkrgj0cgAAAAAAAAAAAAAAAAAAAAAAAAAAADFc+RLg/Ay
mK68iXB+Ag0i7fSIFxHmibcvpESstu47svlaPR9R9xT2mwVfSkmvAit0ua0ZvZ1vbgYw1ttLGpbLWfEzU6KTKJ1hDES
3hHYQLWJYw3AfJnyzjrVeU0S/Gwy6Mwq4oyrpcdyPp8R90QAAAAAAAAAAAAAAAAAAAAAAAAAAABiuvm5cGZTDefNy

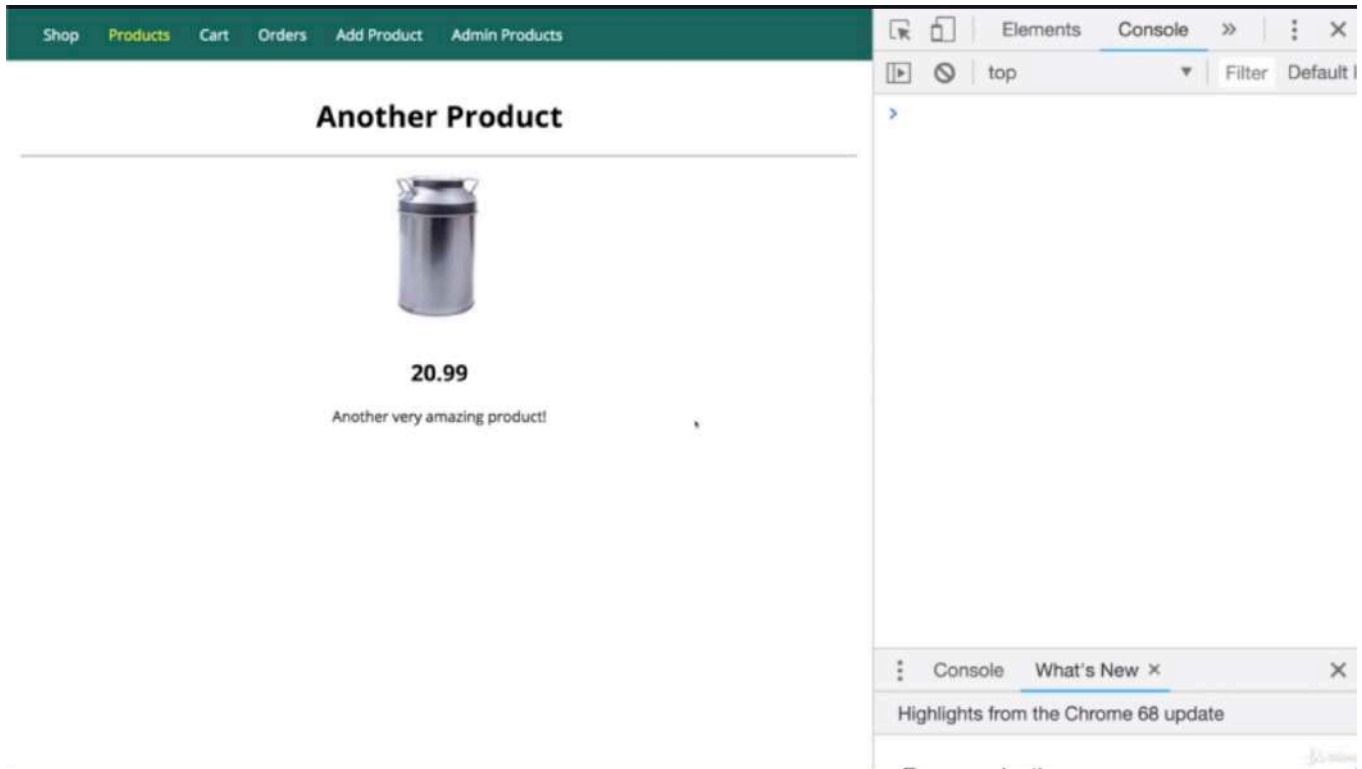
```

```
4MQaPceV7SNU3meu+kY2dmWNrYV9WJYyWwiVYBVPUp7WeYwX4RKlT27vA+au3/cqM9sidBcSJR3ZJiIrHWJGhVmtHiIN
WZK0C/l498kQdIABYUAAAAAAAAAAAAAAAAAAAAAAAAAAAAACpB/JT9FkgiaTfyM/RZZ6NHnPaew/wAYMeeke3I7Mvkt
xGrErqI9QggSXx2nj00z103JiSWfxuKJFFdxIRgpoytkViuJEjk+/wA4h6UfEhXEiZycTdeHpR8SDpqPp8ifTkoAAAAA
AAAAAAAAAAAAAAAAAAAAAFZygguFvKSTeMZX9H033FmeK1NSi4tZTWGu1Fg5vDEulFqSlTtXWh0ZK0ryUr283Vs5N
xk8ypNay46uVt74tPtTK0ld1XUcatFww3lp5UezPnLg0dZUWqezJgqGeE6TWI1Itrqyk/Yz502fYVfFI8omu1l2H2Nm+
wDBA+zlsJcbXG8i3FajHY6kW/ox6Uv2Y5ZFQayb2E3R9V0pQUfnKkoxprry2sy4RWX6u8i01cV56lvQbfX0a3d+onn9p
xNx5LcLP7PLn683VrNeU8YgvopLYl3L3vaS0bXHcfQDkoAAAAAAAAAAAAAAAAAAAAAAAAAAAAARqwo1fnKcJ43a0
E2uD6iSAKS45L2s/Nkv1tb3T1seogT5E0PMk4/qtfwSibUC7RqP/AAUuq4qL9e4/qnqPIuPXXm+Mq/8AVNsA/VGsR5F2
/nSk/wBWL/j1iwt+TdpD9Frek21+z5PuLcDaMdKjGKxGKiluSSSXqMgBAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAH/9k=", "description": "Another very amazing
product!", "price": "20.99", "id": "0.558673316244021"}]}
```

* Chapter 117: Rendering The Product Detail View

1. update

- ./views/shop/product-detail.ejs
- ./public/css/main.css
- ./controllers/shop.js



```
1 <!--./views/shop/product-detail.ejs-->
2
3 <%= include('../includes/head.ejs') %>
4   </head>
5
6   <body>
7     <%= include('../includes/navigation.ejs') %>
8     <main class="centered">
9       <h1><%= product.title %></h1>
10      <hr>
11      <div>
12        ">
```



```

13     </div>
14     <h2><%= product.price %></h2>
15     <p><%= product.description %></p>
16     <form action="/cart" method="post">
17         <button class="btn" type="submit">Add to Cart</button>
18     </form>
19 </main>
20 <%- include('../includes/end.ejs') %>

```

```

1 /*./public/css/main.css*/
2
3 @import url('https://fonts.googleapis.com/css?family=Open+Sans:400,700');
4
5 * {
6     box-sizing: border-box;
7 }
8
9 body {
10     padding: 0;
11     margin: 0;
12     font-family: 'Open Sans', sans-serif;
13 }
14
15 main {
16     padding: 1rem;
17     margin: auto;
18 }
19
20 form {
21     display: inline;
22 }
23
24 .centered {
25     text-align: center;
26 }
27
28 .main-header {
29     width: 100%;
30     height: 3.5rem;
31     background-color: #00695c;
32     padding: 0 1.5rem;
33     display: flex;
34     align-items: center;
35 }
36
37 .main-header__nav {
38     height: 100%;
39     display: none;
40     align-items: center;
41 }
42
43 .main-header__item-list {
44     list-style: none;
45     margin: 0;
46     padding: 0;
47     display: flex;
48 }

```



```
49
50 .main-header__item {
51   margin: 0 1rem;
52   padding: 0;
53 }
54
55 .main-header__item a {
56   text-decoration: none;
57   color: white;
58 }
59
60 .main-header__item a:hover,
61 .main-header__item a:active,
62 .main-header__item a.active {
63   color: #ffeb3b;
64 }
65
66 .mobile-nav {
67   width: 30rem;
68   height: 100vh;
69   max-width: 90%;
70   position: fixed;
71   left: 0;
72   top: 0;
73   background: white;
74   z-index: 10;
75   padding: 2rem 1rem 1rem 2rem;
76   transform: translateX(-100%);
77   transition: transform 0.3s ease-out;
78 }
79
80 .mobile-nav.open {
81   transform: translateX(0);
82 }
83
84 .mobile-nav__item-list {
85   list-style: none;
86   display: flex;
87   flex-direction: column;
88   margin: 0;
89   padding: 0;
90 }
91
92 .mobile-nav__item {
93   margin: 1rem;
94   padding: 0;
95 }
96
97 .mobile-nav__item a {
98   text-decoration: none;
99   color: black;
100   font-size: 1.5rem;
101   padding: 0.5rem 2rem;
102 }
103
104 .mobile-nav__item a:active,
```

```
105 .mobile-nav__item a:hover,
106 .mobile-nav__item a.active {
107     background: #00695c;
108     color: white;
109     border-radius: 3px;
110 }
111
112 #side-menu-toggle {
113     border: 1px solid white;
114     font: inherit;
115     padding: 0.5rem;
116     display: block;
117     background: transparent;
118     color: white;
119     cursor: pointer;
120 }
121
122 #side-menu-toggle:focus {
123     outline: none;
124 }
125
126 #side-menu-toggle:active,
127 #side-menu-toggle:hover {
128     color: #ffeb3b;
129     border-color: #ffeb3b;
130 }
131
132 .backdrop {
133     position: fixed;
134     top: 0;
135     left: 0;
136     width: 100%;
137     height: 100vh;
138     background: rgba(0, 0, 0, 0.5);
139     z-index: 5;
140     display: none;
141 }
142
143 .grid {
144     display: flex;
145     flex-wrap: wrap;
146     justify-content: space-around;
147     align-items: stretch;
148 }
149
150 .card {
151     box-shadow: 0 2px 8px rgba(0, 0, 0, 0.26);
152 }
153
154 .card__header,
155 .card__content {
156     padding: 1rem;
157 }
158
159 .card__header h1,
160 .card__content h1,
```

```

161 .card__content h2,
162 .card__content p {
163   margin: 0;
164 }
165
166 .card__image {
167   width: 100%;
168 }
169
170 .card__image img {
171   width: 100%;
172 }
173
174 .card__actions {
175   padding: 1rem;
176   text-align: center;
177 }
178
179 .card__actions button,
180 .card__actions a {
181   margin: 0 0.25rem;
182 }
183
184 .btn {
185   display: inline-block;
186   padding: 0.25rem 1rem;
187   text-decoration: none;
188   font: inherit;
189   border: 1px solid #00695c;
190   color: #00695c;
191   background: white;
192   border-radius: 3px;
193   cursor: pointer;
194 }
195
196 .btn:hover,
197 .btn:active {
198   background-color: #00695c;
199   color: white;
200 }
201
202 @media (min-width: 768px) {
203   .main-header__nav {
204     display: flex;
205   }
206
207   #side-menu-toggle {
208     display: none;
209   }
210 }
211

```

```

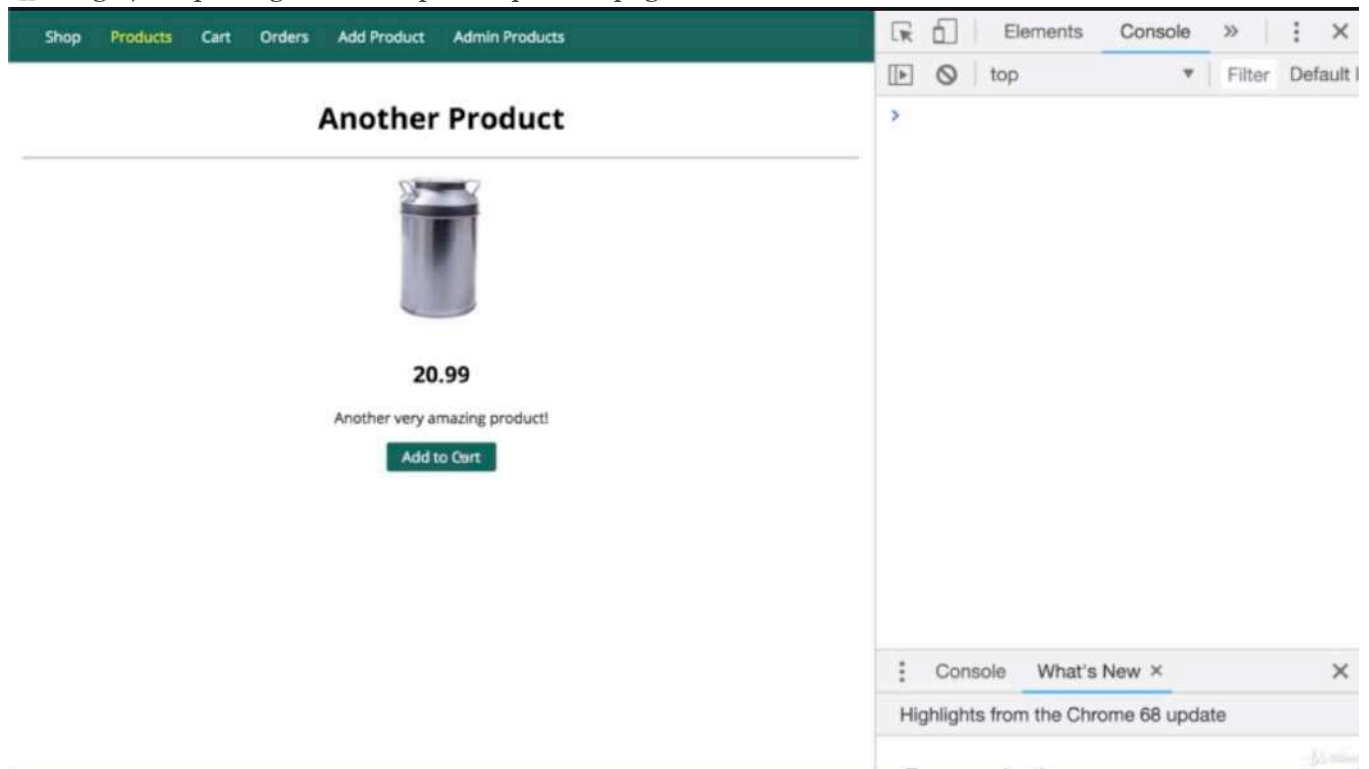
1 //../controllers/shop.js
2
3 const Product = require('../models/product');
4
5 exports.getProducts = (req, res, next) => {

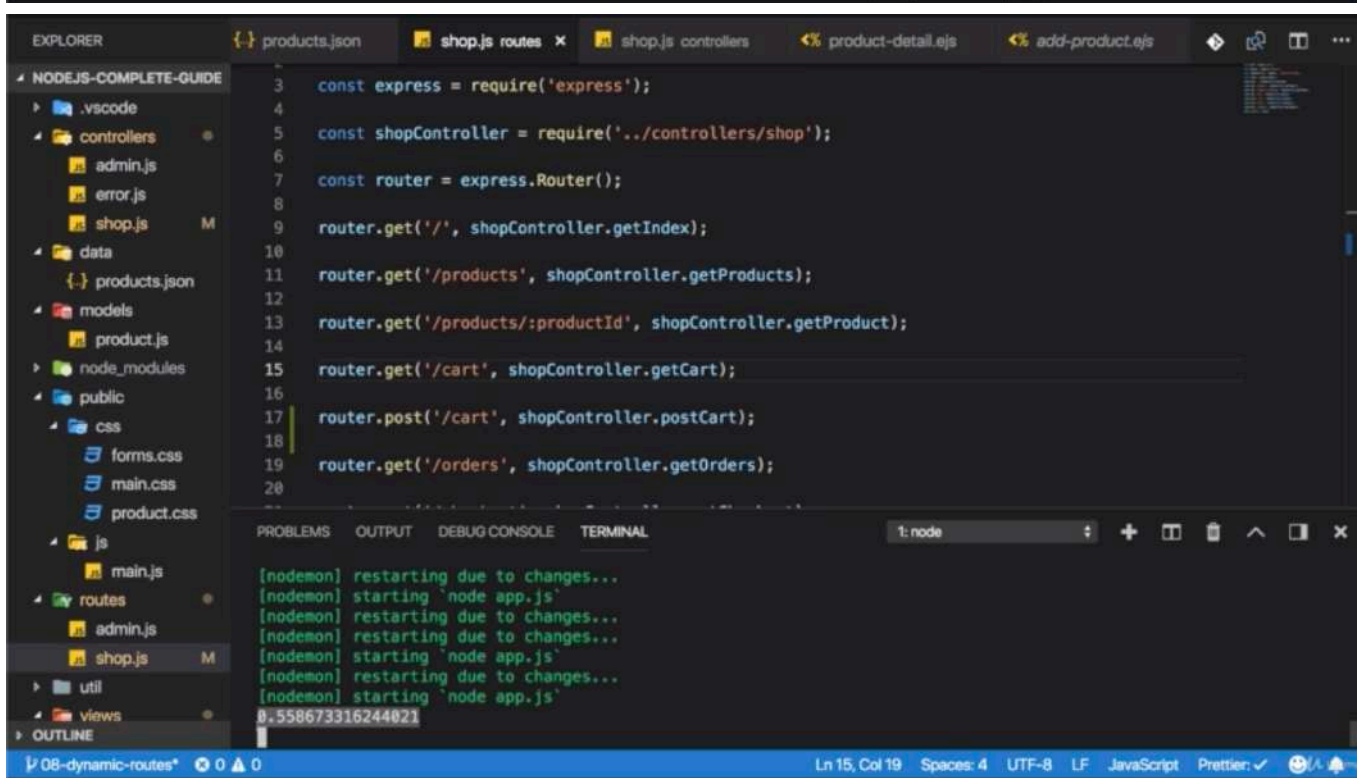
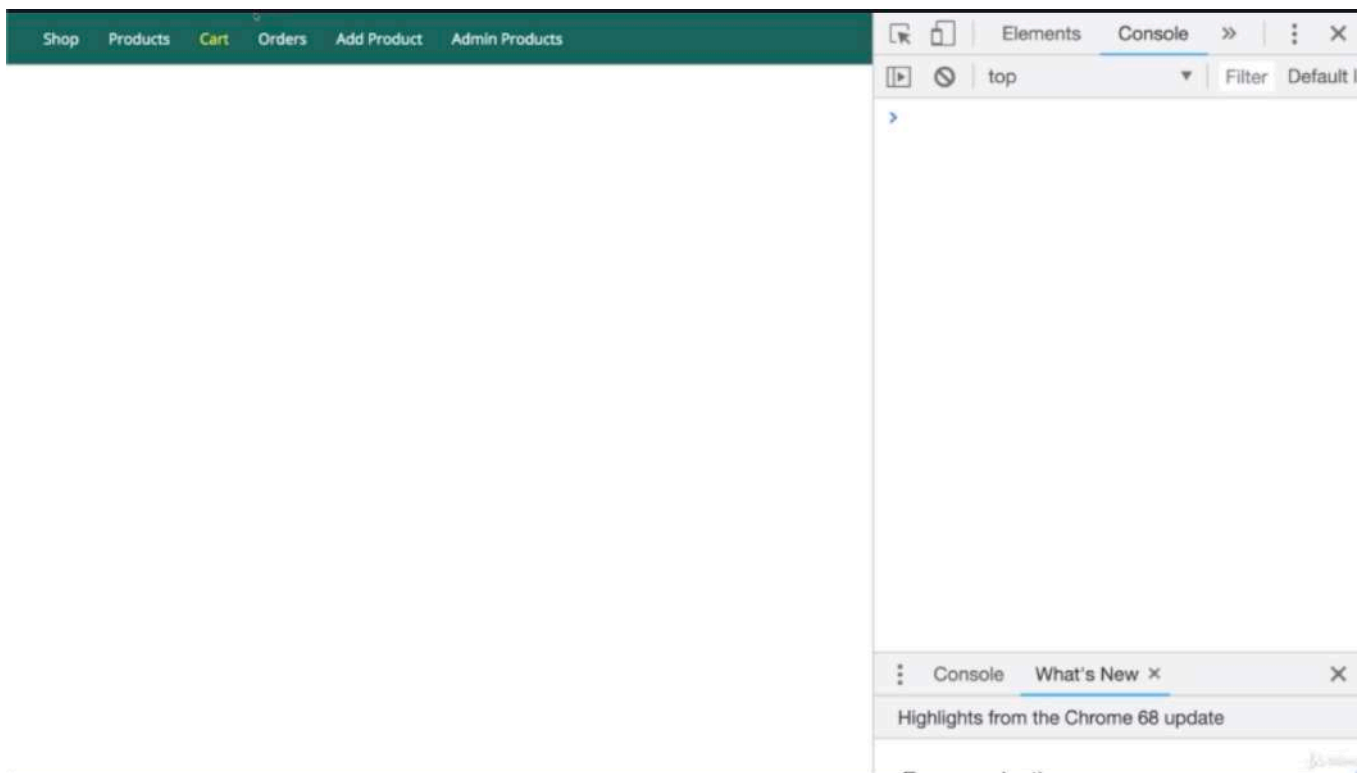
```

```
6   Product.fetchAll(products => {
7     res.render('shop/product-list', {
8       prods: products,
9       pageTitle: 'All Products',
10      path: '/products'
11    });
12  });
13 };
14
15 exports.getProduct = (req, res, next) => {
16   const prodId = req.params.productId;
17   Product.findById(prodId, product => {
18     res.render('shop/product-detail', {
19       /**we now need to pass in a product property
20        * because we are accessing 'product' in the 'product-detail.ejs' file in the view
21        *
22        * 'product' of the left is the key by which we will be able to access it in the view
23        * 'product' of the right is 'product' in 'Product.findById(prodId, //product// =>{})'
24        *
25        */
26     product: product,
27     pageTitle: product.title,
28     path: '/products'
29   })
30 });
31 };
32
33 exports.getIndex = (req, res, next) => {
34   Product.fetchAll(products => {
35     res.render('shop/index', {
36       prods: products,
37       pageTitle: 'Shop',
38       path: '/'
39     });
40   });
41 };
42
43 exports.getCart = (req, res, next) => {
44   res.render('shop/cart', {
45     path: '/cart',
46     pageTitle: 'Your Cart'
47   });
48 };
49
50 exports.getOrders = (req, res, next) => {
51   res.render('shop/orders', {
52     path: '/orders',
53     pageTitle: 'Your Orders'
54   });
55 };
56
57 exports.getCheckout = (req, res, next) => {
58   res.render('shop/checkout', {
59     path: '/checkout',
60     pageTitle: 'Checkout'
61   });
62 };
```

* Chapter 118: Passing Data With POST Requests


1. update
 - ./views/shop/product-detail.ejs
 - ./views/admin/add-product.ejs
 - ./routes/shop.js
 - ./controllers/shop.js
 - ./views/shop/index.ejs
 - ./views/shop/product-list.ejs
 - ./views/includes/add-to-cart.ejs





ShopProductsCartOrdersAdd ProductAdmin Products

A Book




\$ 19

This is an awesome book!

DetailsAdd to Cart

Another Product



\$ 20.99

Another very amazing product!

DetailsAdd to Cart

ElementsConsole>>⋮×

topFilterDefault


>

⋮ConsoleWhat's New ××

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

Another Product



20.99

Another very amazing product!

Add to Cart

ElementsConsole>>⋮×

topFilterDefault


>

⋮ConsoleWhat's New ××

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

A Book




\$ 19

This is an awesome book!

DetailsAdd to Cart

Another Product



\$ 20.99

Another very amazing product!

DetailsAdd to Cart

ElementsConsole>>⋮×

topFilterDefault

>

ConsoleWhat's New ××

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

ElementsConsole>>⋮×

topFilterDefault

>

ConsoleWhat's New ××

Highlights from the Chrome 68 update

The screenshot shows a VS Code editor with the file explorer on the left displaying a project structure for 'NODEJS-COMPLETE-GUIDE'. The main editor shows the 'product-detail.ejs' file with the following content:

```
1 </head>
2
3 <body>
4   <%- include('../includes/navigation.ejs') %>
5   <main class="centered">
6     <h1><%= product.title %></h1>
7     <hr>
8     <div>
9       ">
10    </div>
11    <h2><%= product.price %></h2>
12    <p><%= product.description %></p>
13    <%- include('../includes/add-to-cart.ejs') %>
14  </main>
15  <%- include('../includes/end.ejs') %>
16
```

The terminal window at the bottom shows an error message:

```
de/node_modules/ejs/lib/ejs.js:482:10)
    at View.render (/Users/mschwarzmueller/development/teaching/udemy/nodejs-complete-guide/node_modules/express/lib/view.js:135:8)
    at tryRender (/Users/mschwarzmueller/development/teaching/udemy/nodejs-complete-guide/node_modules/express/lib/application.js:640:10)
    at Function.render (/Users/mschwarzmueller/development/teaching/udemy/nodejs-complete-guide/node_modules/express/lib/application.js:592:3)
    0.558673316244021
```

```
1 <!--../views/shop/product-detail.ejs-->
2
3 <%- include('../includes/head.ejs') %>
4   </head>
5
6   <body>
7     <%- include('../includes/navigation.ejs') %>
8     <main class="centered">
9       <h1><%= product.title %></h1>
10      <hr>
11      <div>
12        ">
13      </div>
14      <h2><%= product.price %></h2>
15      <p><%= product.description %></p>
16      <!--we can grab the entire <form></form>
17          and make sure we also use that form in all the place
18          where we also have added to cart button.
19      -->
20      <%- include('../includes/add-to-cart.ejs') %>
21      <%- include('../includes/end.ejs') %>
```

```
1 <!--../views/admin/add-product.ejs-->
2
3 <%- include('../includes/head.ejs') %>
4   <link rel="stylesheet" href="/css/forms.css">
5   <link rel="stylesheet" href="/css/product.css">
6 </head>
7
8 <body>
9   <%- include('../includes/navigation.ejs') %>
10
11   <main>
12     <!--For adding a product, we have a form
13     and as i mentioned there when we added this the first time in the course,
```

```

14      this automatically gives us a request which puts all the input data and so on
    into its body
15      but this("method="POST"") only works for posting data
16      this is not available for getting data but for posting data,
17      we therefore don't need to add anything to the URL
18      because we can put the information in the POST body.
19      so therefore what we should do on product-detail in this form is that
20      we should simply pass some important information as part of the POST request
21      -->
22      <form class="product-form" action="/admin/add-product" method="POST">
23      <div class="form-control">
24          <label for="title">Title</label>
25          <input type="text" name="title" id="title">
26      </div>
27      <div class="form-control">
28          <label for="imageUrl">Image URL</label>
29          <input type="text" name="imageUrl" id="imageUrl">
30      </div>
31      <div class="form-control">
32          <label for="price">Price</label>
33          <input type="number" name="price" id="price" step="0.01">
34      </div>
35      <div class="form-control">
36          <label for="description">Description</label>
37          <textarea name="description" id="description" rows="5"></textarea>
38      </div>
39
40      <button class="btn" type="submit">Add Product</button>
41  </form>
42  </main>
43  <%- include('../includes/end.ejs') %>

```

```

1  // ./routes/shop.js
2
3  const path = require('path');
4
5  const express = require('express');
6
7  const shopController = require('../controllers/shop');
8
9  const router = express.Router();
10
11  router.get('/', shopController.getIndex);
12
13  router.get('/products', shopController.getProducts);
14
15  router.get('/products/:productId', shopController.getProduct);
16
17  router.get('/cart', shopController.getCart);
18
19  router.post('/cart', shopController.postCart);
20
21  router.get('/orders', shopController.getOrders);
22
23  router.get('/checkout', shopController.getCheckout);
24
25  module.exports = router;

```

```

1  ../controllers/shop.js
2
3  const Product = require('../models/product');
4
5  exports.getProducts = (req, res, next) => {
6    Product.fetchAll(products => {
7      res.render('shop/product-list', {
8        prods: products,
9        pageTitle: 'All Products',
10       path: '/products'
11     });
12   });
13 };
14
15 exports.getProduct = (req, res, next) => {
16   const prodId = req.params.productId;
17   Product.findById(prodId, product => {
18     res.render('shop/product-detail', {
19       /**we now need to pass in a product property
20        * because we are accessing 'product' in the 'product-detail.ejs' file in the view
21        *
22        * 'product' of the left is the key by which we will be able to access it in the view
23        * 'product' of the right is 'product' in 'Product.findById(prodId, //product// =>{})'
24        *
25        */
26       product: product,
27       pageTitle: product.title,
28       path: '/products'
29     })
30   });
31 };
32
33 exports.getIndex = (req, res, next) => {
34   Product.fetchAll(products => {
35     res.render('shop/index', {
36       prods: products,
37       pageTitle: 'Shop',
38       path: '/'
39     });
40   });
41 };
42
43 exports.getCart = (req, res, next) => {
44   res.render('shop/cart', {
45     path: '/cart',
46     pageTitle: 'Your Cart'
47   });
48 };
49
50 exports.postCart = (req, res, next) => {
51   /**we will now have to retrieve the productId from the incoming request
52    * and then also fetch that product in our database so in our file
53    * and add it to our cart.
54    *
55    * 'productId' is from '<input type="hidden" name="productId" value="<%= product.id %>">'
56    in ./views/shop/product-detail.ejs

```

```

56  */
57  const prodId = req.body.productId;
58  console.log(prodId)
59  res.redirect('/cart')
60 }
61
62 exports.getOrders = (req, res, next) => {
63   res.render('shop/orders', {
64     path: '/orders',
65     pageTitle: 'Your Orders'
66   });
67 };
68
69 exports.getCheckout = (req, res, next) => {
70   res.render('shop/checkout', {
71     path: '/checkout',
72     pageTitle: 'Checkout'
73   });
74 };

```

```

1  <!--./views/shop/index.ejs-->
2
3  <%- include('../includes/head.ejs') %>
4    <link rel="stylesheet" href="/css/product.css">
5  </head>
6
7  <body>
8    <%- include('../includes/navigation.ejs') %>
9
10   <main>
11     <% if (prods.length > 0) { %>
12       <div class="grid">
13         <% for (let product of prods) { %>
14           <article class="card product-item">
15             <header class="card_header">
16               <h1 class="product_title"><%= product.title %></h1>
17             </header>
18             <div class="card_image">
19               "
21             </div>
22             <div class="card_content">
23               <h2 class="product_price">$<%= product.price %></h2>
24               <p class="product_description"><%= product.description %></p>
25             </div>
26             <div class="card_actions">
27               <%- include('../includes/add-to-cart.ejs', {product: product})
28             </div>
29           </article>
30         <% } %>
31       </div>
32     <% } else { %>
33       <h1>No Products Found!</h1>
34     <% } %>
35   </main>
36   <%- include('../includes/end.ejs') %>

```

```

1 <!--./views/shop/product-list.ejs-->
2
3 <%- include('../includes/head.ejs') %>
4   <link rel="stylesheet" href="/css/product.css">
5   </head>
6
7   <body>
8     <%- include('../includes/navigation.ejs') %>
9
10    <main>
11      <% if (prods.length > 0) { %>
12        <div class="grid">
13          <% for (let product of prods) { %>
14            <article class="card product-item">
15              <header class="card_header">
16                <h1 class="product_title">
17                  <%= product.title %>
18                </h1>
19              </header>
20              <div class="card_image">
21                ">
22              </div>
23              <div class="card_content">
24                <h2 class="product_price">$
25                  <%= product.price %>
26                </h2>
27                <p class="product_description">
28                  <%= product.description %>
29                </p>
30              </div>
31              <div class="card_actions">
32                <!--/product/0.43432 would be one possible path
33                now we wanna make sure that we are able to handle that
34                and then extract that unique ID from the path in our
35                routes file.
36                so that in the controller, we can load the correct
37                product and how the details for it.
38                we send some information as part of the path.
39                so that we can extract all the data we need for the
40                product from the controller or inside of the controller
41                because we can't really send the entire product as part
42                of the URL
43                but we can send this key information.
44                -->
45                <!--we are looping through all the products
46                and then product is a local variable available in that
47                loop only,
48                then in 'include', included in the loop doesn't get that
49                variable by default
50                but you can pass it to that 'include'
51                you can do so by adding a second argument to the include
52                function
53                where you pass an object

```

```

49 and simply add that variable again.
50
51 'include'
52 and 'product' of the right is what you include or you
add the value you available in this file,
53 so product in the loop is now passed to product in the
include.
54 this needs to be done for all our include so also in
index.ejs.
55 but not one in the 'product-detail' because it's not
inside a loop
56 so it's available globally and therefore works well.
57 -->
58 <a href="/products/<%= product.id %>"
class="btn">Details</a>
59 <%- include('../includes/add-to-cart.ejs', {product:
product}) %> </div>
60 </article>
61 <% } %>
62 </div>
63 <% } else { %>
64 <h1>No Products Found!</h1>
65 <% } %>
66 </main>
67 <%- include('../includes/end.ejs') %>

```

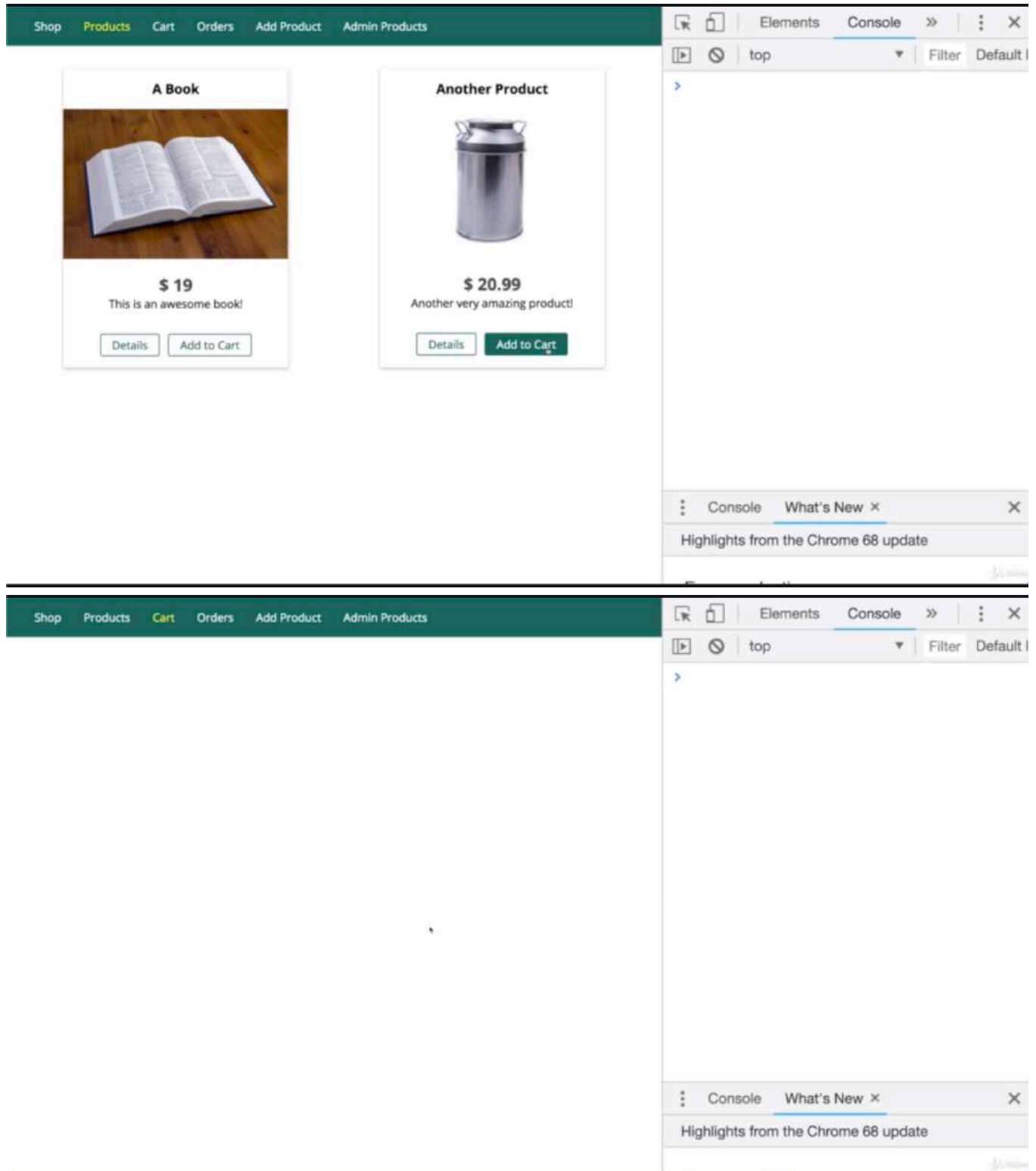
```

1 <!--../views/includes/add-to-cart.ejs-->
2
3 <form action="/cart" method="post">
4 <!--'Add to Cart' sends a POST request
5 and this has one important implication
6 we can pass data in the request body
7 this was not possible for a GET request
8 but for POST request, you typically use the request body
9 this is also what we use when adding a product.
10
11 so therefore what we should do on product-detail in this form is that
12 we should simply pass some important information as part of the POST request
13 -->
14 <button class="btn" type="submit">Add to Cart</button>
15 <!--i will make it of type 'hidden'
16 so that it doesn't really get displayed on the page
17 but still encoded in the request that is getting sent
18 and i will give this a name of productId
19
20 we also need to store a value that will be the value
21 which is added to the request for that productId.
22
23 i will use ejs again in the value to output productId
24 because i still wanna pass that ID to my node express app
25 but i don't wanna use the URL
26 because i do use a POST request so there's no need to use it.
27 -->
28 <input type="hidden" name="productId" value="<%= product.id %>">
29 </form>
30

```

* Chapter 119: Adding A Cart Model

- 1. update
- ./models/cart.js
- ./controllers/shop.js



```
33 };
34
35 exports.getCart = (req, res, next) => {
36   res.render('shop/cart', {
37     path: '/cart',
38     pageTitle: 'Your Cart'
39   });
40 };
41
42 exports.postCart = (req, res, next) => {
43   const prodId = req.body.productId;
44   Product.findById(prodId, product => {
45     Cart.addProduct(prodId, product.price);
46   });
47   res.redirect('/cart');
48 };
49
50 exports.getOrders = (req, res, next) => {
51   res.render('shop/orders', {
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: node

```
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
[nodemon] restarting due to changes...
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
null
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
```

Ln 45, Col 44 Spaces: 2 UTF-8 LF JavaScript Prettier: ✓

- 'null' means there was no error. so good sign.

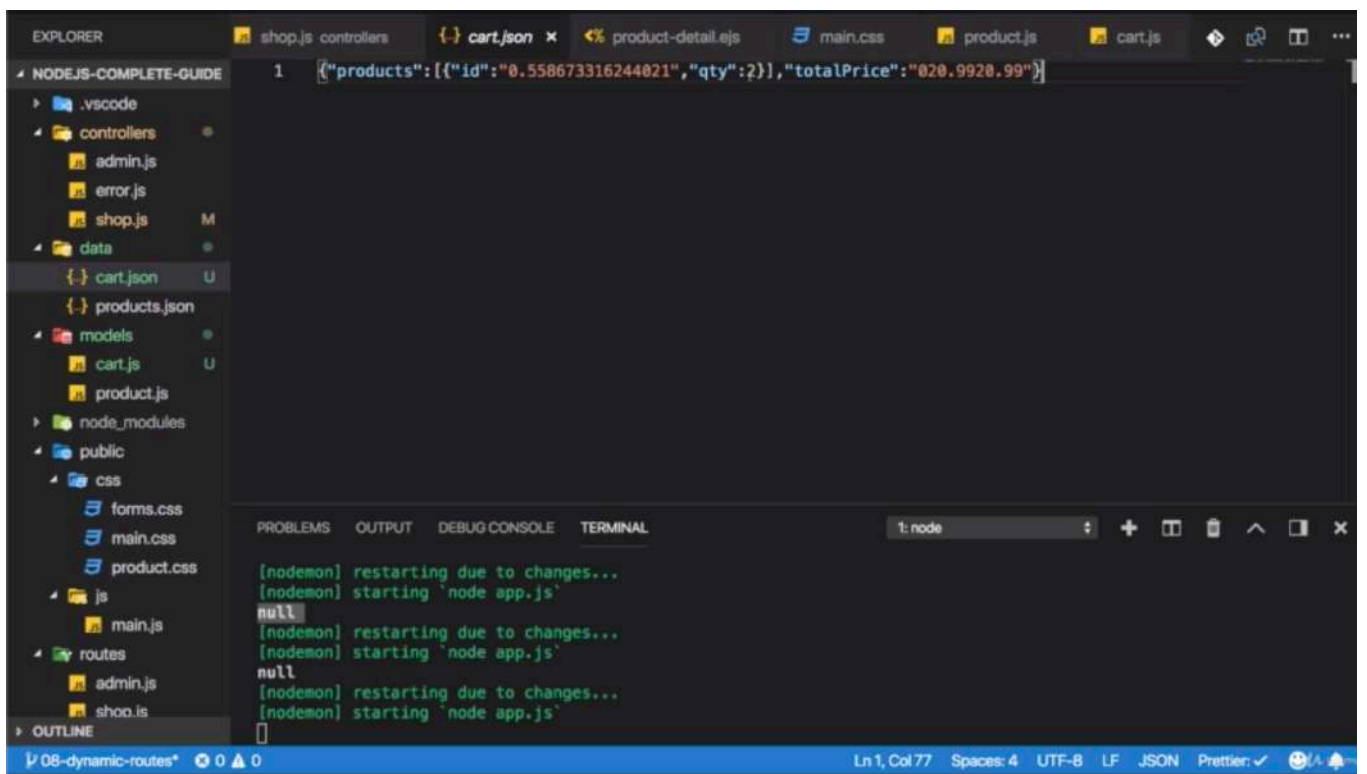
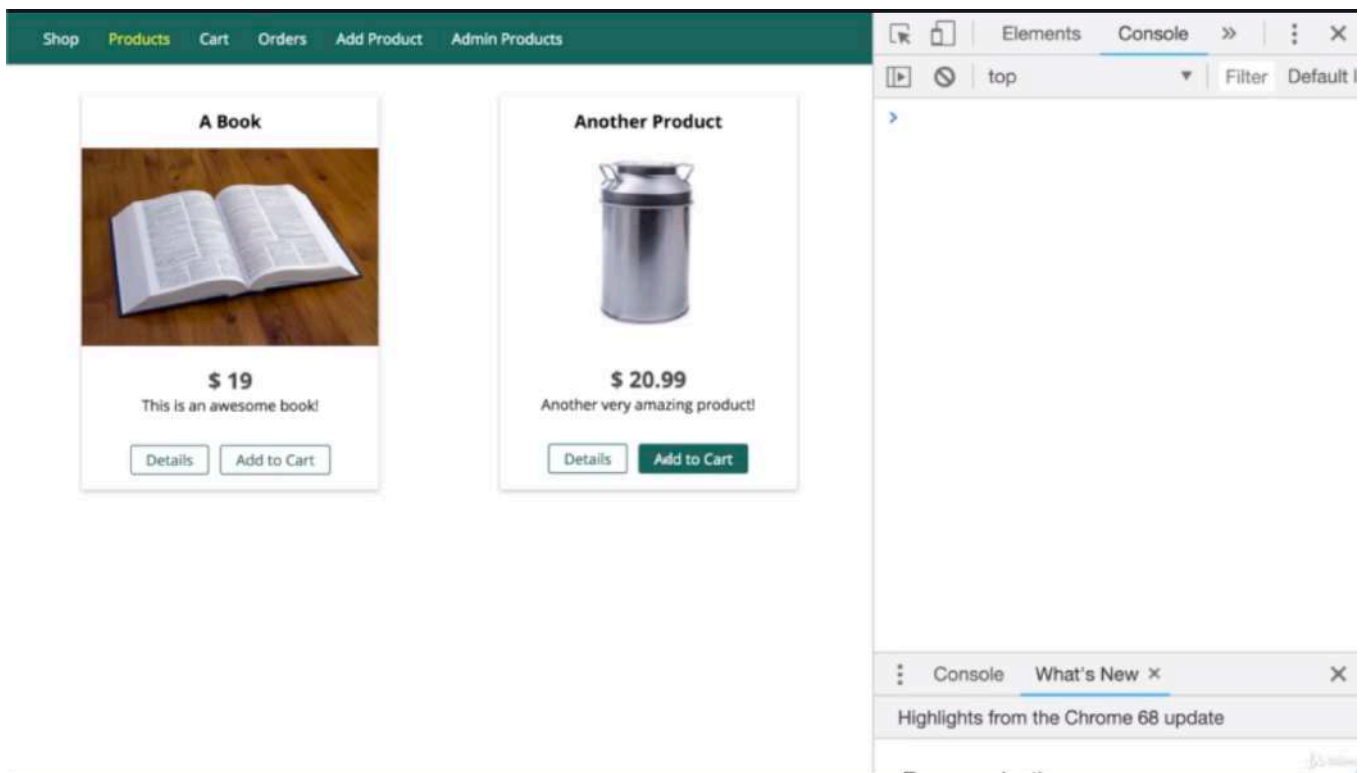

```
1 {"products":[{"id":"0.558673316244021","qty":1},"totalPrice":"020.99"}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: node

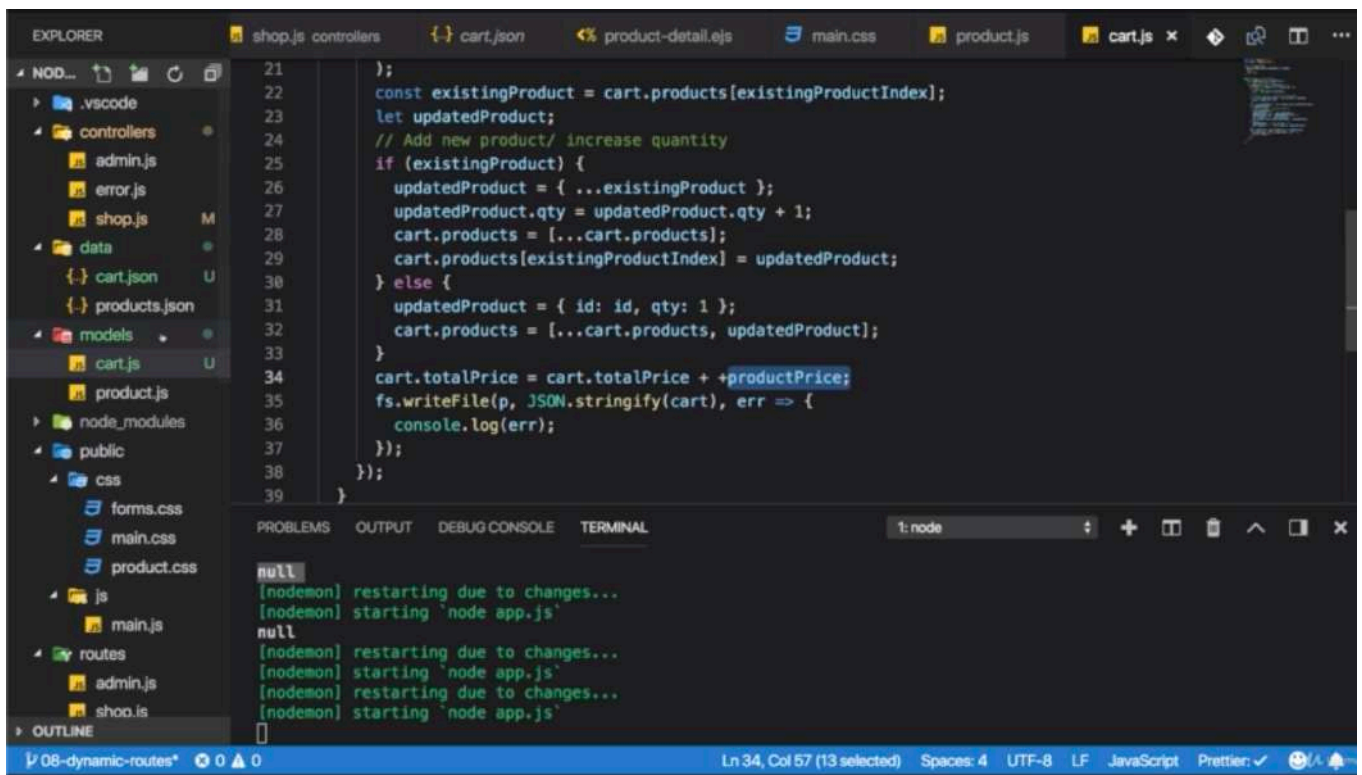
```
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
[nodemon] restarting due to changes...
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
null
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
```

Ln 1, Col 46 (5 selected) Spaces: 4 UTF-8 LF JSON Prettier: ✓



- the quantity is increased. but the 'totalPrice' is misbehaving because it's stored as a string and therefore concatenated. so we will have to do something about that.

- in cart.js, the price we are extracting, it's stored as a string in our product model.



```
21  };
22  const existingProduct = cart.products[existingProductIndex];
23  let updatedProduct;
24  // Add new product/ increase quantity
25  if (existingProduct) {
26    updatedProduct = { ...existingProduct };
27    updatedProduct.qty = updatedProduct.qty + 1;
28    cart.products = [...cart.products];
29    cart.products[existingProductIndex] = updatedProduct;
30  } else {
31    updatedProduct = { id: id, qty: 1 };
32    cart.products = [...cart.products, updatedProduct];
33  }
34  cart.totalPrice = cart.totalPrice + +productPrice;
35  fs.writeFile(p, JSON.stringify(cart), err => {
36    console.log(err);
37  });
38  });
39  }
```

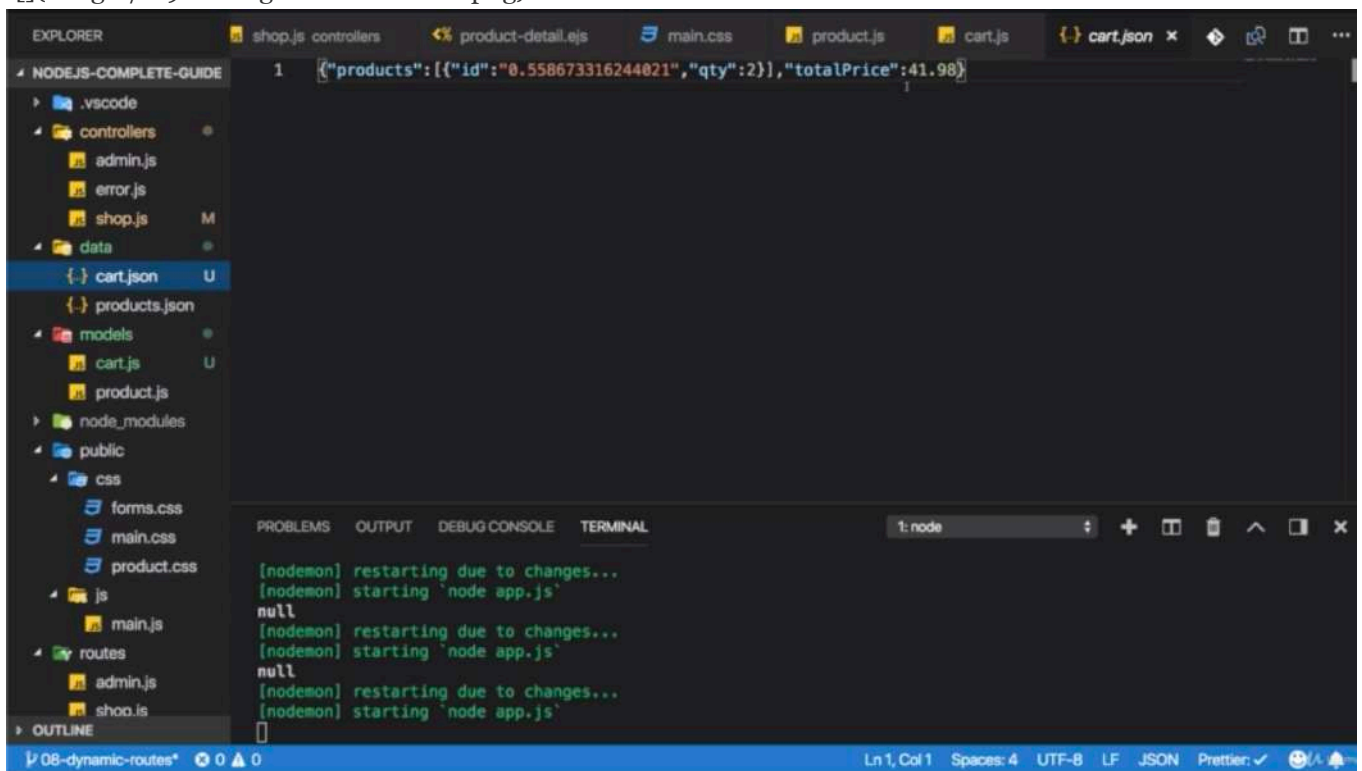
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: node

```
null
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
null
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
```

Ln 34, Col 57 (13 selected) Spaces: 4 UTF-8 LF JavaScript Prettier: ✓

- so what we have to do is when we work on the price, i have to add a plus in front of productPrice to convert that string to a number



```
1  {"products":[{"id":"0.558673316244021","qty":2}],"totalPrice":41.98}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: node

```
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
null
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
null
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
```

Ln 1, Col 1 Spaces: 4 UTF-8 LF JSON Prettier: ✓

```
1  //./models/cart.js
2
3  const fs = require('fs');
4  const path = require('path');
5
6  const p = path.join(
7    path.dirname(process.mainModule.filename),
8    'data',
9    'cart.json'
10 );
```

```

11
12 module.exports = class Cart {
13     /**what we need on this cart is a way to add and remove our products
14     * now the problem is that the cart itself is not really an object we will constantly
    recreate,
15     * not for every new product that we add we wanna have a new cart,
16     * instead there always will be cart in our application
17     * and we wanna manage the products in there.
18     * so the approach i wanna take will be a different one.
19     *
20     * so i will add a 'static' method
21     */
22     static addProduct(id, productPrice) {
23         // Fetch the previous cart
24         fs.readFile(p, (err, fileContent) => {
25             let cart = { products: [], totalPrice: 0 };
26             if (!err) {
27                 cart = JSON.parse(fileContent);
28             }
29             // Analyze the cart => Find existing product
30             /**so you will have the existing product index */
31             const existingProductIndex = cart.products.findIndex(
32             /**'[existingProductIndex]' allows me to use that index to replace the item in our
    cart products.*/
33                 prod => prod.id === id
34             );
35             const existingProduct = cart.products[existingProductIndex];
36             let updatedProduct;
37             // Add new product/ increase quantity
38             if (existingProduct) {
39                 updatedProduct = { ...existingProduct };
40                 updatedProduct.qty = updatedProduct.qty + 1;
41                 cart.products = [...cart.products];
42                 /**i will not add 'updatedProduct'
43                 * instead i will set 'cart.products'
44                 * and overwrite 'existingProductIndex'
45                 */
46                 cart.products[existingProductIndex] = updatedProduct;
47             } else {
48                 /**if we are creating a product for the first time,
49                 * so if i'm in this 'else' block,
50                 * then i simply will add the updated product as a new additional product.
51                 * however if i got an existing product here,
52                 * i don't wanna add a new product.
53                 * instead i wanna replace the old one
54                 * and to do that, i need to find out where in my old products existingProduct
    is located in.
55                 * so to do that, i will get the index instead of the product.
56                 */
57                 updatedProduct = { id: id, qty: 1 };
58                 cart.products = [...cart.products, updatedProduct];
59             }
60             /**i have to add a plus in front of productPrice to convert that string to a number */
61             cart.totalPrice = cart.totalPrice + +productPrice;
62             fs.writeFile(p, JSON.stringify(cart), err => {
63                 console.log(err);

```

```
64     });
65   });
66 }
67 };
68
```

```
1  //./controllers/shop.js
2
3  const Product = require('../models/product');
4  const Cart = require('../models/cart');
5
6  exports.getProducts = (req, res, next) => {
7    Product.fetchAll(products => {
8      res.render('shop/product-list', {
9        prods: products,
10       pageTitle: 'All Products',
11       path: '/products'
12     });
13   });
14 };
15
16 exports.getProduct = (req, res, next) => {
17   const prodId = req.params.productId;
18   Product.findById(prodId, product => {
19     res.render('shop/product-detail', {
20       product: product,
21       pageTitle: product.title,
22       path: '/products'
23     })
24   });
25 };
26
27 exports.getIndex = (req, res, next) => {
28   Product.fetchAll(products => {
29     res.render('shop/index', {
30       prods: products,
31       pageTitle: 'Shop',
32       path: '/'
33     });
34   });
35 };
36
37 exports.getCart = (req, res, next) => {
38   res.render('shop/cart', {
39     path: '/cart',
40     pageTitle: 'Your Cart'
41   });
42 };
43
44 exports.postCart = (req, res, next) => {
45   const prodId = req.body.productId;
46   Product.findById(prodId, product => {
47     Cart.addProduct(prodId, product.price)
48   })
49   res.redirect('/cart')
50 }
51
```

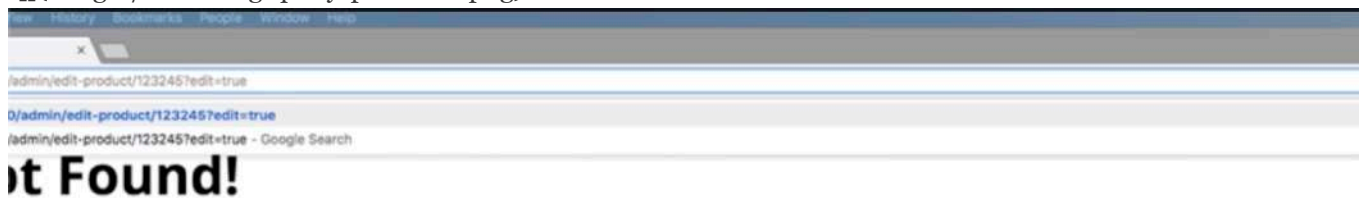
```

52 exports.getOrders = (req, res, next) => {
53   res.render('shop/orders', {
54     path: '/orders',
55     pageTitle: 'Your Orders'
56   });
57 };
58
59 exports.getCheckout = (req, res, next) => {
60   res.render('shop/checkout', {
61     path: '/checkout',
62     pageTitle: 'Checkout'
63   });
64 };

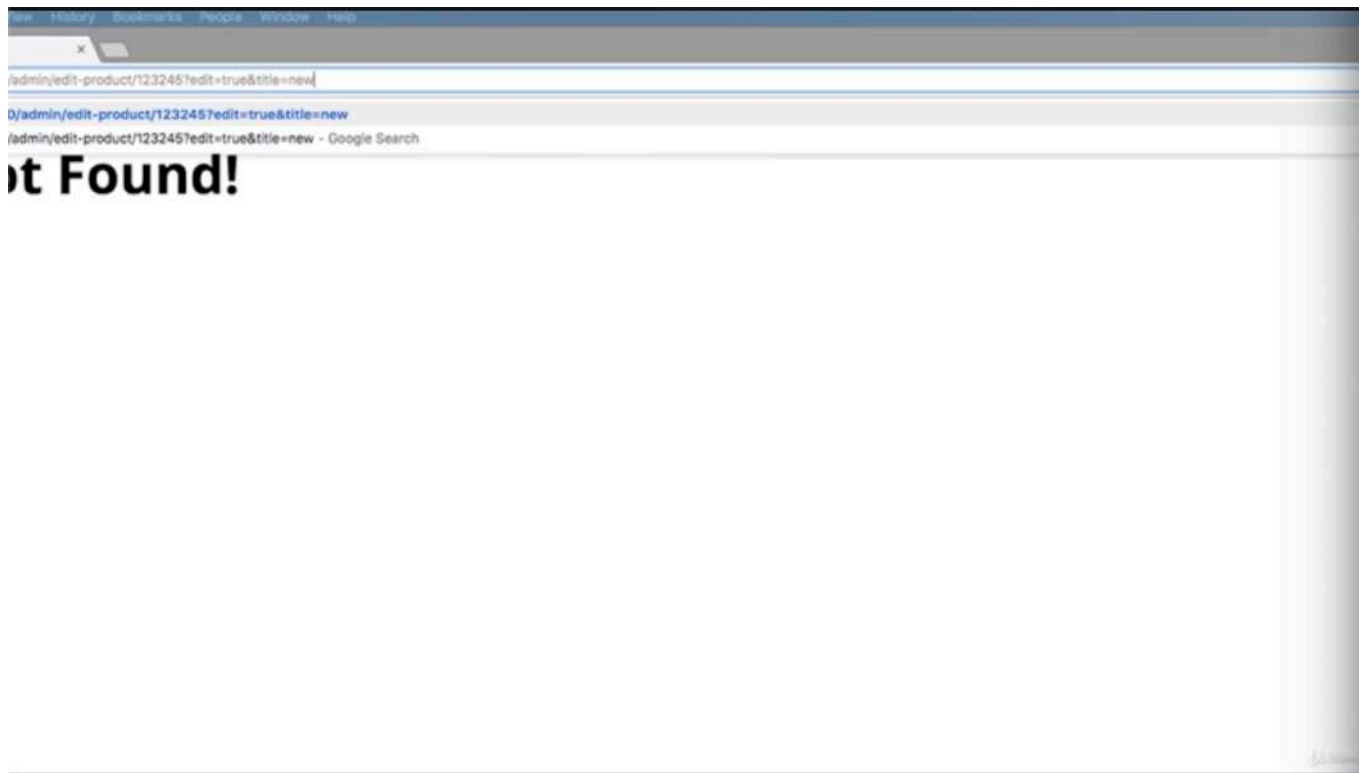
```

* Chapter 120: Using Query Params

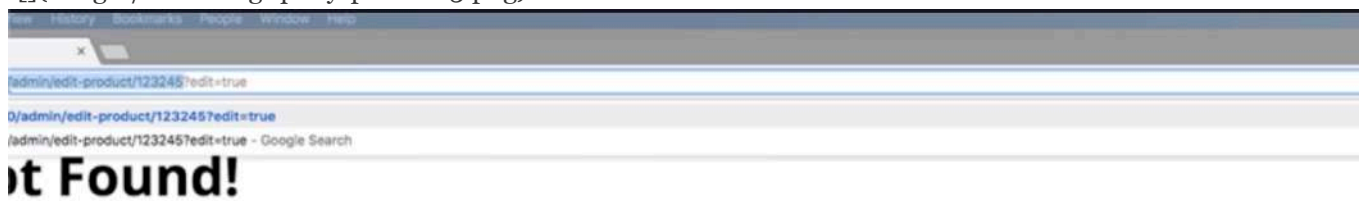
1. update
 - ./views/admin/edit-product.ejs
 - ./controllers/admin.js
 - ./routes/admin.js
 - ./routes/shop.js



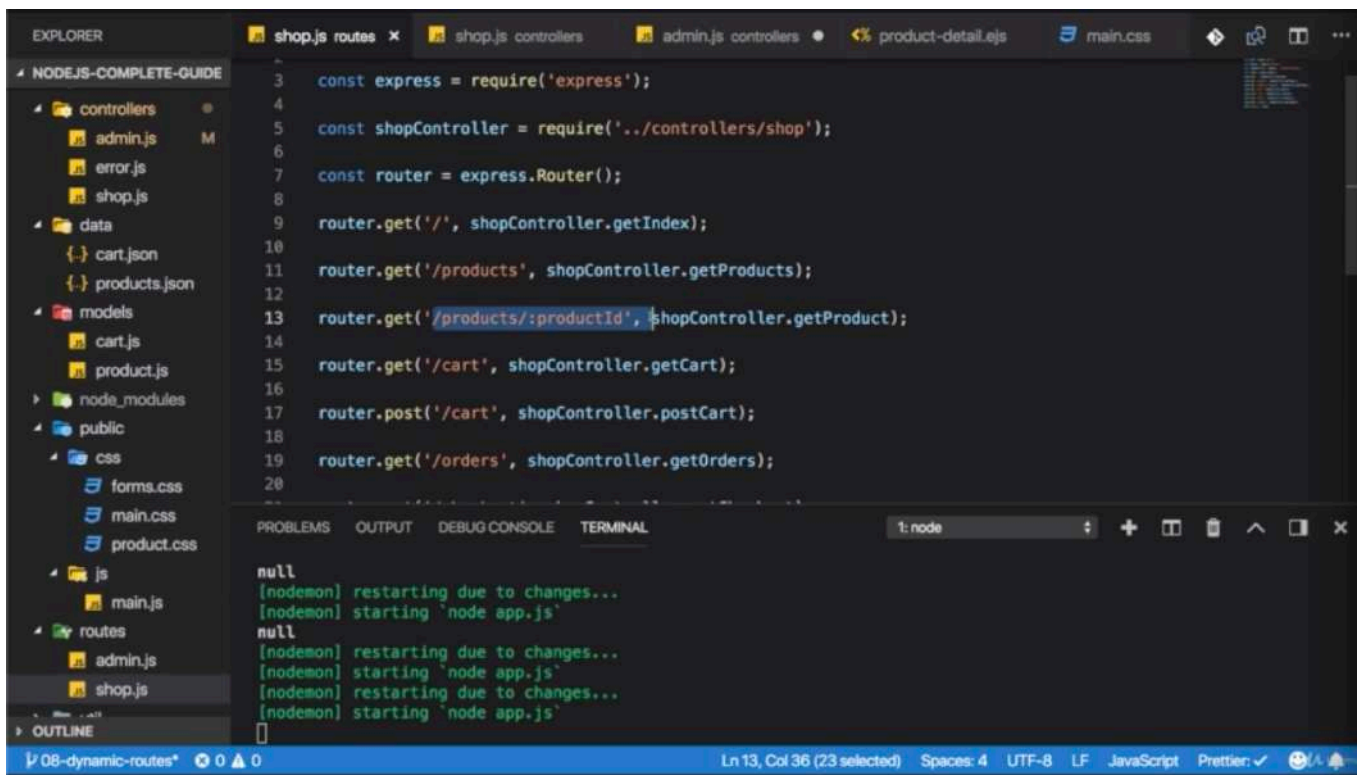
- we wanna get an additional confirmation by ensuring that people have to pass us a so-called 'query parameter' in the URL.
- A query parameter can be added to any URL by adding a question mark and then a key-value pair separated by an equal sign



- you can have multiple query parameters by separating and them with ampersand(&)
- so this is so-called optional data.



- the path, the route which get reached is determined by the part up to the question mark.



```
3 const express = require('express');
4
5 const shopController = require('../controllers/shop');
6
7 const router = express.Router();
8
9 router.get('/', shopController.getIndex);
10
11 router.get('/products', shopController.getProducts);
12
13 router.get('/products/:productId', shopController.getProduct);
14
15 router.get('/cart', shopController.getCart);
16
17 router.post('/cart', shopController.postCart);
18
19 router.get('/orders', shopController.getOrders);
20
```

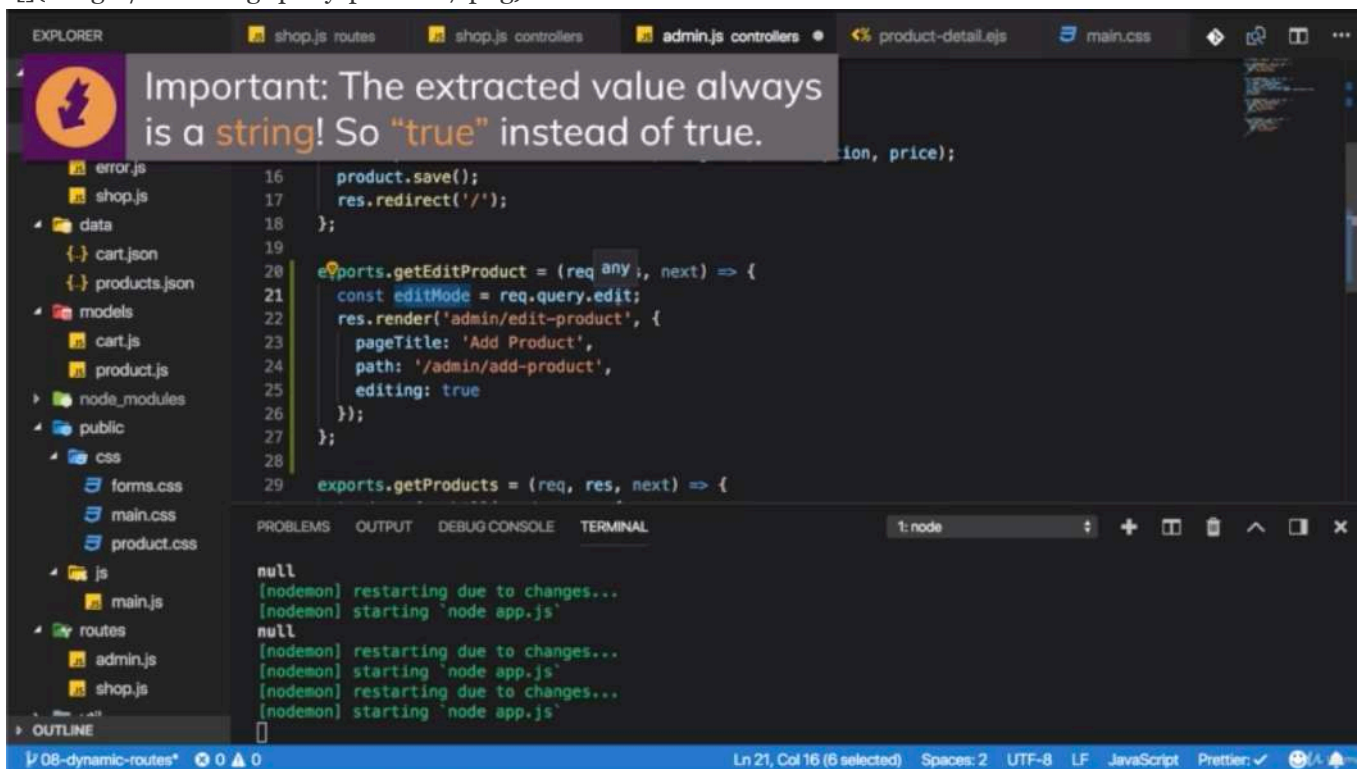
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: node

```
null
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
null
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
```

Ln 13, Col 36 (23 selected) Spaces: 4 UTF-8 LF JavaScript Prettier ✓

so you don't need to add any information about query parameters you might get to your routes file. these paths are not affected but you can always check for query parameters in your controller.



Important: The extracted value always is a **string**! So "true" instead of true.

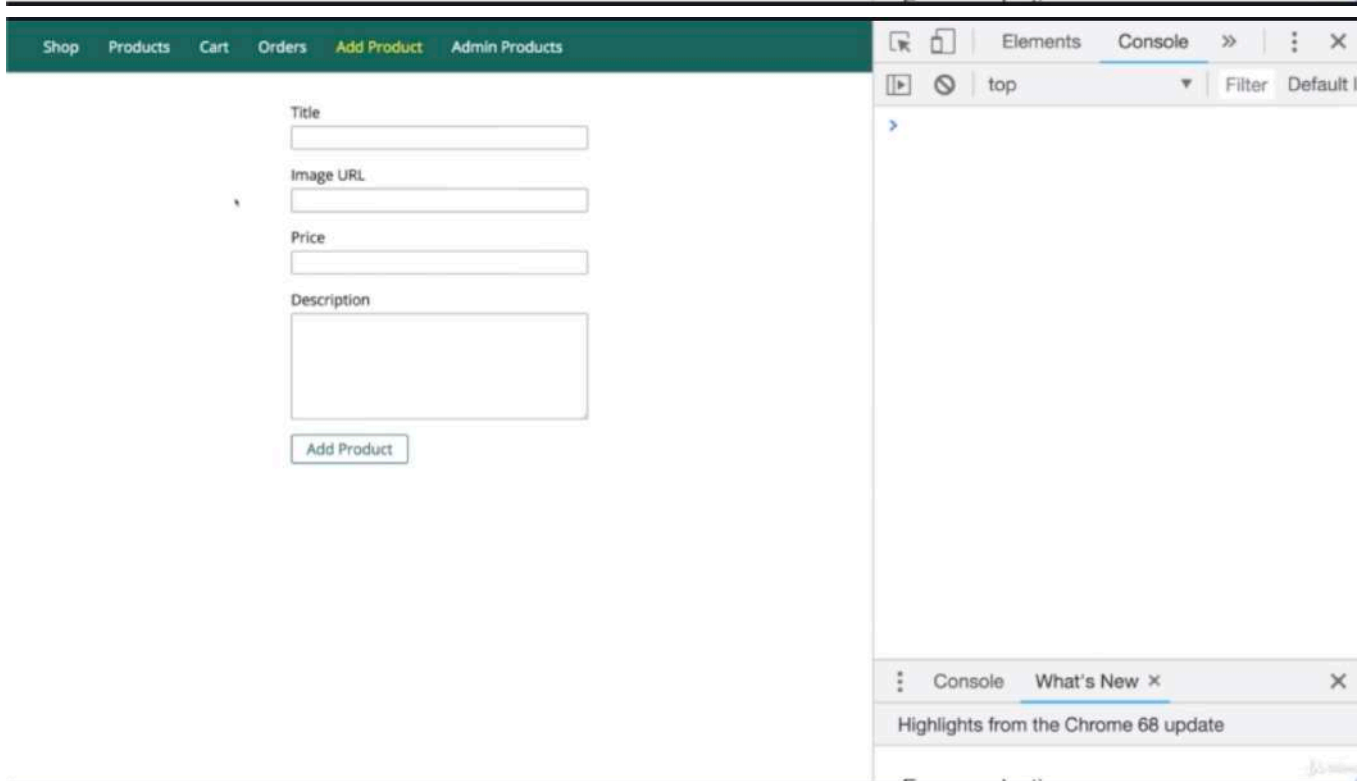
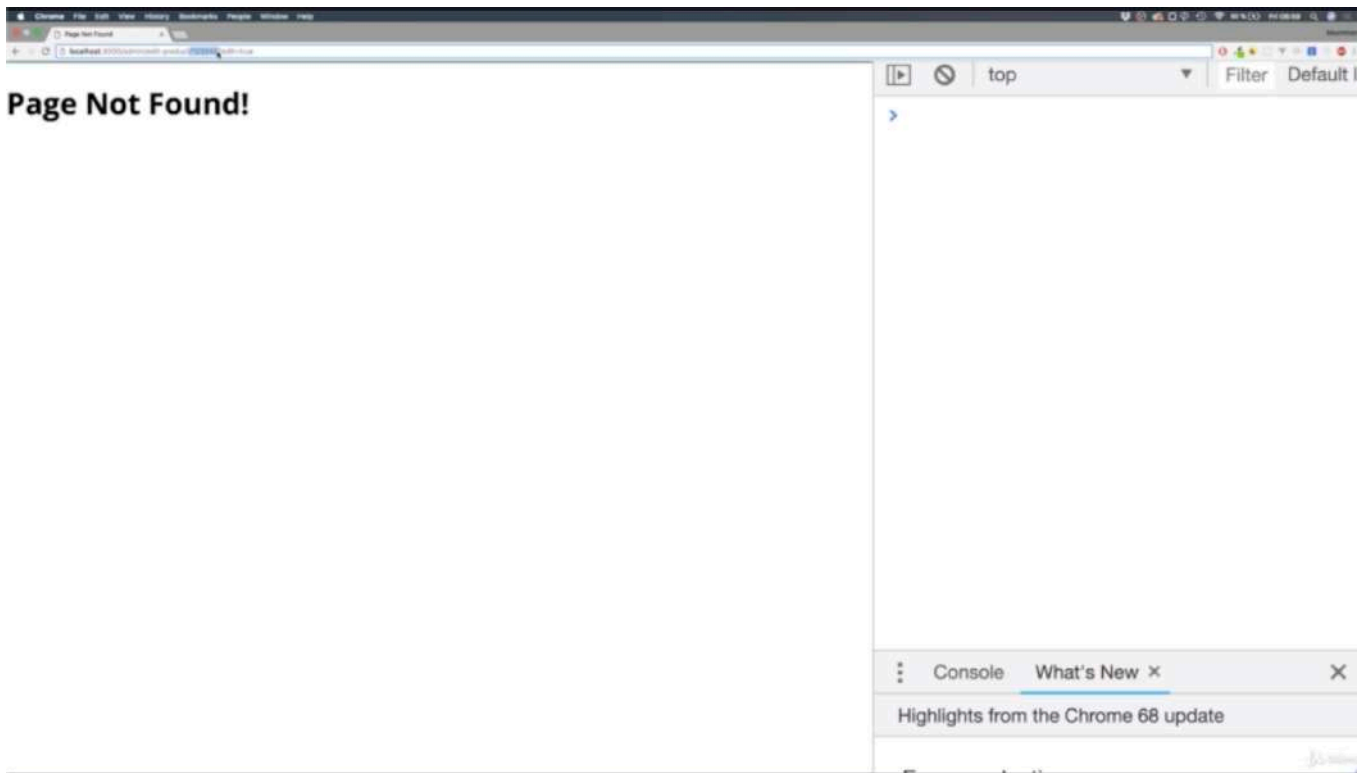
```
16 product.save();
17 res.redirect('/');
18 };
19
20 exports.getEditProduct = (req, res, next) => {
21   const editMode = req.query.edit;
22   res.render('admin/edit-product', {
23     pageTitle: 'Add Product',
24     path: '/admin/add-product',
25     editing: true
26   });
27 };
28
29 exports.getProducts = (req, res, next) => {
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: node

```
null
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
null
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
```

Ln 21, Col 16 (6 selected) Spaces: 2 UTF-8 LF JavaScript Prettier ✓



```
1 <!--./views/admin/edit-product.ejs-->
2
3 <%- include('../includes/head.ejs') %>
4   <link rel="stylesheet" href="/css/forms.css">
5   <link rel="stylesheet" href="/css/product.css">
6 </head>
7
8 <body>
9   <%- include('../includes/navigation.ejs') %>
10
11   <main>
12     <form class="product-form" action="/admin/add-product" method="POST">
13       <div class="form-control">
```



```

14     <label for="title">Title</label>
15     <input type="text" name="title" id="title">
16 </div>
17 <div class="form-control">
18     <label for="imageUrl">Image URL</label>
19     <input type="text" name="imageUrl" id="imageUrl">
20 </div>
21 <div class="form-control">
22     <label for="price">Price</label>
23     <input type="number" name="price" id="price" step="0.01">
24 </div>
25 <div class="form-control">
26     <label for="description">Description</label>
27     <textarea name="description" id="description" rows="5"></textarea>
28 </div>
29
30     <button class="btn" type="submit">Add Product</button>
31 </form>
32 </main>
33 <%- include('../includes/end.ejs') %>

```

```

1 // ./controllers/admin.js
2
3 const Product = require('../models/product');
4
5 exports.getAddProduct = (req, res, next) => {
6   res.render('admin/edit-product', {
7     pageTitle: 'Add Product',
8     /**this can be stated as 'add-product'
9     * because that is what we use for highlighting a certain navigation item.
10    *
11    */
12     path: '/admin/add-product',
13   });
14 };
15
16 exports.postAddProduct = (req, res, next) => {
17   const title = req.body.title;
18   const imageUrl = req.body.imageUrl;
19   const price = req.body.price;
20   const description = req.body.description;
21   const product = new Product(title, imageUrl, description, price);
22   product.save();
23   res.redirect('/');
24 };
25
26 exports.getEditProduct = (req, res, next) => {
27   /**you don't need to add any information
28   * about query parameters you might get to your routes file.
29   * these paths are not affected
30   * but you can always check for query parameters in your controller.
31   *
32   * so here we can check if this get request, and there will be a 'query' object
33   * this is also created and managed by express.js
34   * and you can access your data by trying to access the keys you got in your query
35   parameters.
36   *

```

```

36 * http://localhost:3000/admin/edit-product/123245?edit=true&title=new
37 *
38 * and only if 'edit' is set to somewhere in the list of query parameter which can be
separated with ampersand,
39 * if this is set, then i will get the value which i set
40 * so now i would have true in 'editMode'
41 * if i don't have it, if i don't find it, i will get 'undefined' which is treated as
false in boolean check
42 */
43
44 /**'edit' in 'req.query.edit' is 'edit' in 'http://localhost:3000/admin/edit-
product/123245?edit=true&title=new' */
45 const editMode = req.query.edit
46 if(!editMode){
47     return res.redirect('/');
48 }
49 res.render('admin/edit-product', {
50     pageTitle: 'Edit Product',
51     path: '/admin/edit-product',
52     /**we can pass an additional information field to our view editing
53     * and set this to true so that we can check this with an if condition to find out.
54     * for example, if upon clicking that save button,
55     * we should try to add the product
56     * and send a request to that route
57     * or try to edit it and send it to a different route.
58     */
59     editing: editMode
60 });
61 };
62
63
64 exports.getProducts = (req, res, next) => {
65     Product.fetchAll(products => {
66         res.render('admin/products', {
67             prods: products,
68             pageTitle: 'Admin Products',
69             path: '/admin/products'
70         });
71     });
72 };
73

```

```

1 // ./routes/admin.js
2
3 const path = require('path');
4
5 const express = require('express');
6
7 const adminController = require('../controllers/admin');
8
9 const router = express.Router();
10
11 // /admin/add-product => GET
12 router.get('/add-product', adminController.getAddProduct);
13
14 // /admin/products => GET
15 router.get('/products', adminController.getProducts);

```

```

16
17 // /admin/add-product => POST
18 router.post('/add-product', adminController.postAddProduct);
19
20 router.get('/edit-product/:productId', adminController.getEditProduct);
21
22 module.exports = router;

1 // ./routes/shop.js
2
3 const path = require('path');
4
5 const express = require('express');
6
7 const shopController = require('../controllers/shop');
8
9 const router = express.Router();
10
11 router.get('/', shopController.getIndex);
12
13 router.get('/products', shopController.getProducts);
14 /**you don't need to add any information
15  * about query parameters you might get to your routes file.
16  * these paths are not affected
17  * but you can always check for query parameters in your controller. */
18 router.get('/products/:productId', shopController.getProduct);
19
20 router.get('/cart', shopController.getCart);
21
22 router.post('/cart', shopController.postCart)
23
24 router.get('/orders', shopController.getOrders);
25
26 router.get('/checkout', shopController.getCheckout);
27
28 module.exports = router;

```

* Chapter 121: Pre-Populating The Edit Product Page With Data

1. update
 - ./controllers/admin.js
 - ./views/admin/edit-product.ejs

ShopProductsCartOrdersAdd ProductAdmin Products

Title

Image URL

Price

Description

Update Product

ElementsConsole>>⋮×

▶🔍top▼FilterDefault

>

⋮ConsoleWhat's New ××

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

Title

Image URL

Price

Description

Update Product

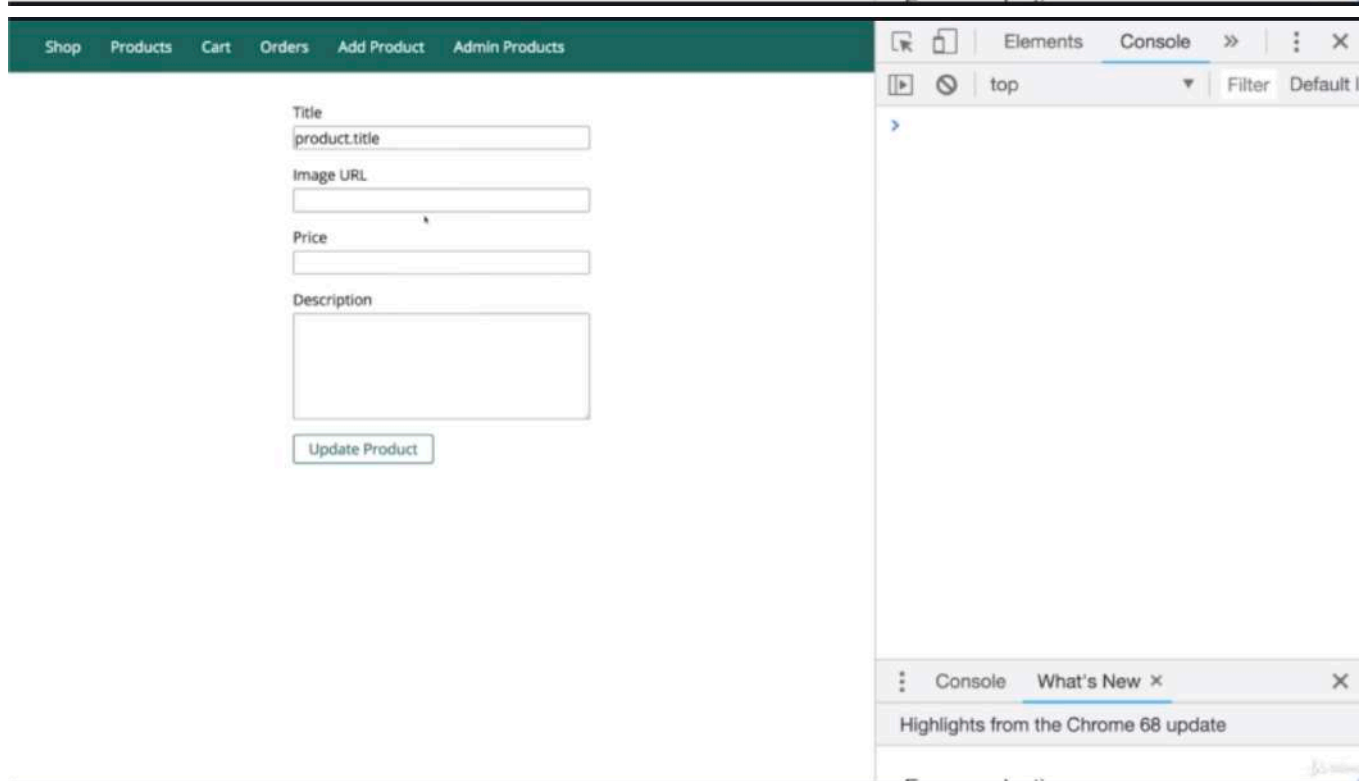
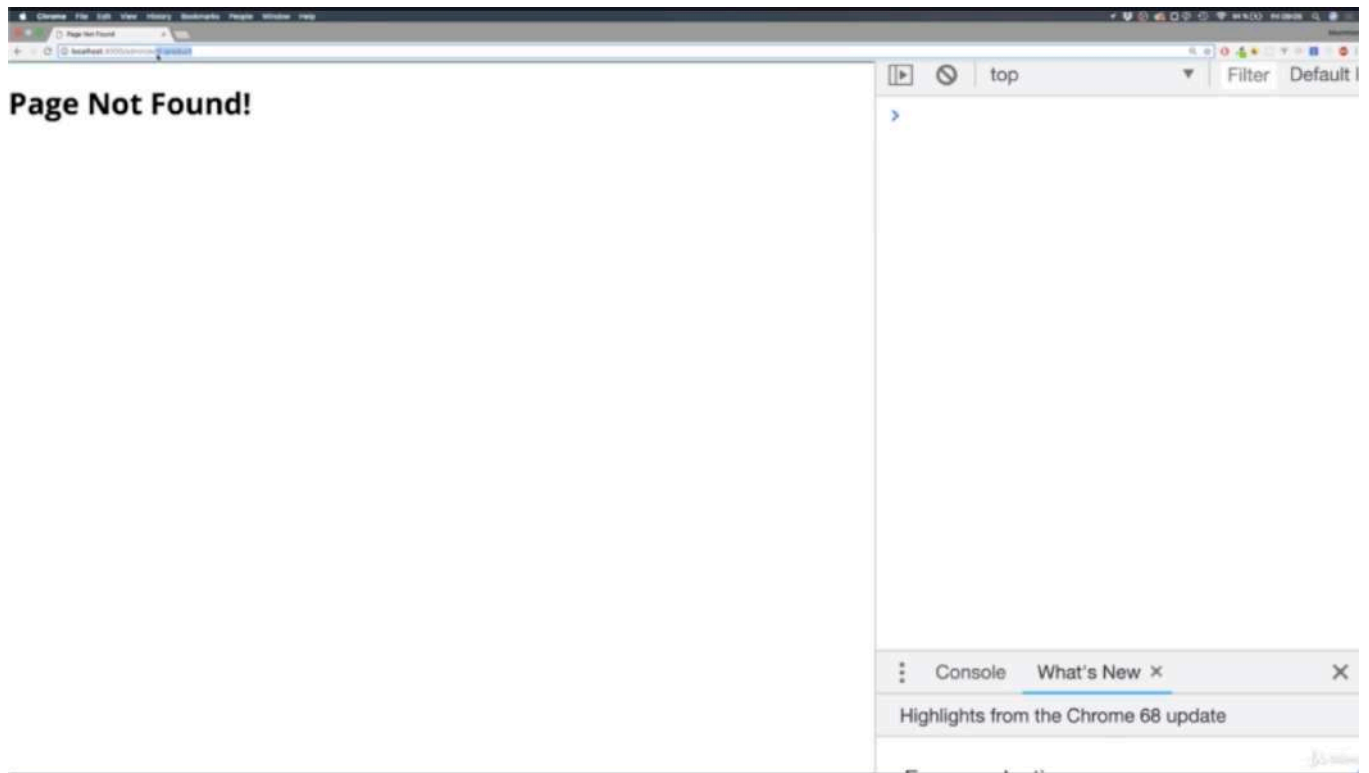
ElementsConsole>>⋮×

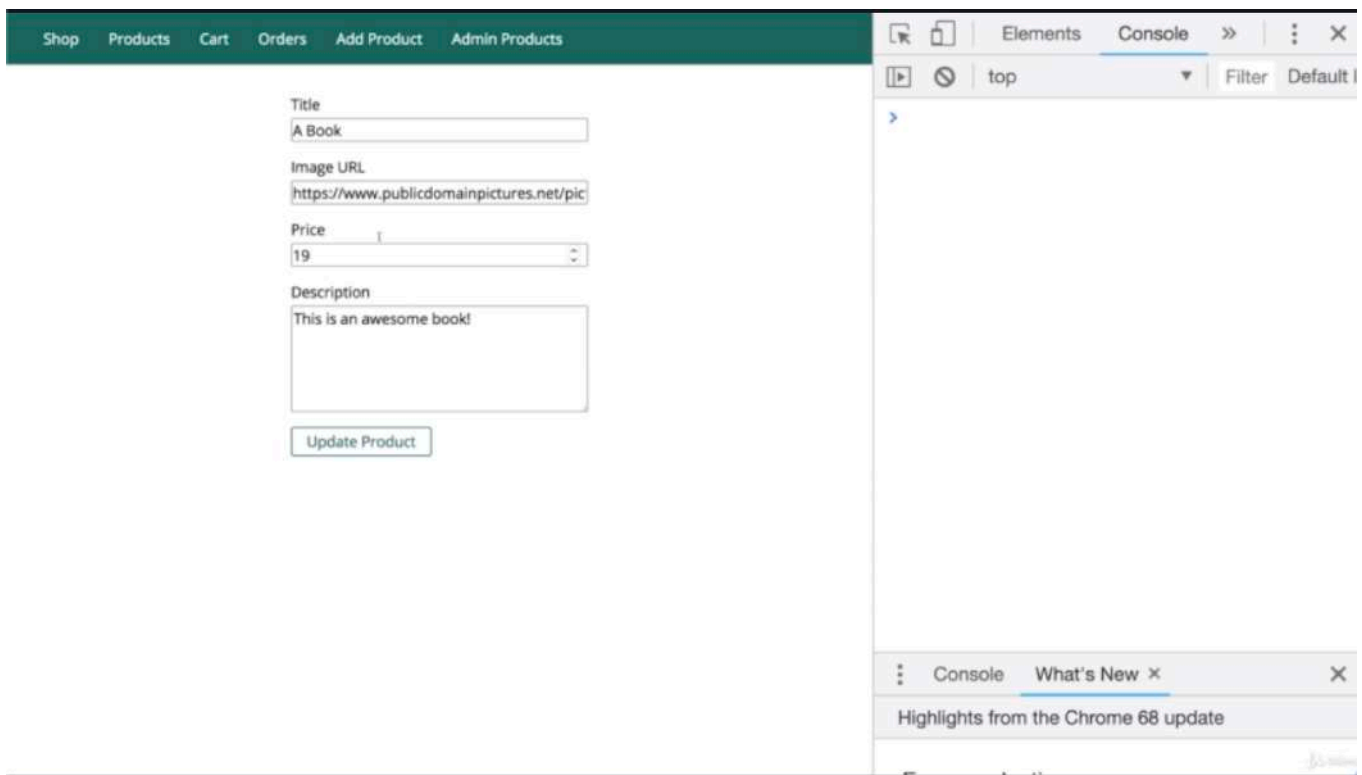
▶🔍top▼FilterDefault

>

⋮ConsoleWhat's New ××

Highlights from the Chrome 68 update





```

1 <!--./views/admin/edit-product.ejs-->
2
3 <%- include('../includes/head.ejs') %>
4   <link rel="stylesheet" href="/css/forms.css">
5   <link rel="stylesheet" href="/css/product.css">
6 </head>
7
8 <body>
9   <%- include('../includes/navigation.ejs') %>
10
11   <main>
12     <form class="product-form" action="/admin/<% if (editing) { %>edit-product<% } else
13 { %>add-product<% } %>" method="POST">
14       <div class="form-control">
15         <label for="title">Title</label>
16         <input type="text" name="title" id="title" value="<% if (editing) { %><%=
17 product.title %><% } %>">
18       </div>
19       <div class="form-control">
20         <label for="imageUrl">Image URL</label>
21         <input type="text" name="imageUrl" id="imageUrl" value="<% if (editing) { %>
22 <%= product.imageUrl %><% } %>">
23       </div>
24       <div class="form-control">
25         <label for="price">Price</label>
26         <input type="number" name="price" id="price" step="0.01" value="<% if
27 (editing) { %><%= product.price %><% } %>">
28       </div>
29       <div class="form-control">
30         <label for="description">Description</label>
31         <textarea name="description" id="description" rows="5"><% if (editing) {
32 %>Update Product<% } else { %>Add Product<% } %></textarea>
33       </div>
34     </form>
35   </main>
36 </body>
37 </html>

```

```

30         <button class="btn" type="submit"><% if (editing) { %>Update Product<% } else {
    %>Add Product<% } %></button>
31     </form>
32 </main>
33 <%- include('../includes/end.ejs') %>

```

```

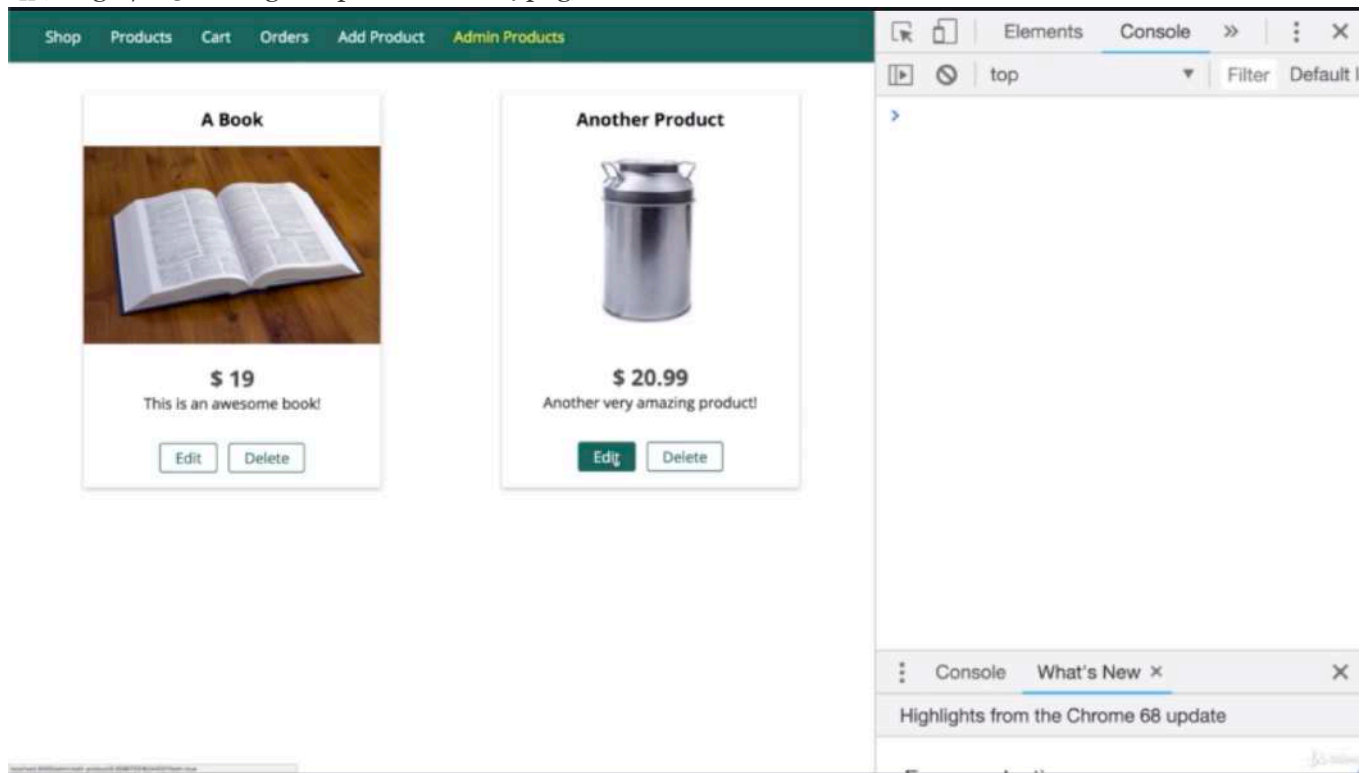
1 // ./controllers/admin.js
2
3 const Product = require('../models/product');
4
5 exports.getAddProduct = (req, res, next) => {
6     res.render('admin/edit-product', {
7         pageTitle: 'Add Product',
8         path: '/admin/add-product',
9         editing: false
10    });
11 };
12
13 exports.postAddProduct = (req, res, next) => {
14     const title = req.body.title;
15     const imageUrl = req.body.imageUrl;
16     const price = req.body.price;
17     const description = req.body.description;
18     const product = new Product(title, imageUrl, description, price);
19     product.save();
20     res.redirect('/');
21 };
22
23 exports.getEditProduct = (req, res, next) => {
24     const editMode = req.query.edit
25     if(!editMode){
26         return res.redirect('/');
27     }
28     const prodId = req.params.productId;
29     Product.findById(prodId, product => {
30         if(!product){
31             return res.redirect('/');
32         }
33         res.render('admin/edit-product', {
34             pageTitle: 'Edit Product',
35             path: '/admin/edit-product',
36             editing: editMode,
37             product: product
38         });
39     })
40 };
41
42
43 exports.getProducts = (req, res, next) => {
44     Product.fetchAll(products => {
45         res.render('admin/products', {
46             prods: products,
47             pageTitle: 'Admin Products',
48             path: '/admin/products'
49         });
50     });
51 };

```

* Chapter 123: Editing The Product Data

1. update

- ./views/admin/product.ejs
- ./views/admin/edit-product.ejs
- ./routes/admin.js
- ./controllers/admin.js
- ./models/product.js



ShopProductsCartOrdersAdd ProductAdmin Products

Title

Another Product

Image URL

data:image/jpeg;base64,/9j/4AAQSkZJRgAE

Price

20.99

Description

Another very amazing product!

Update Product

ElementsConsole>>⋮×

topFilterDefault

>

⋮ ConsoleWhat's New ××

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

Title

Another Product!!!

Image URL

data:image/jpeg;base64,/9j/4AAQSkZJRgAE

Price

30.95

Description

Another very amazing product! It's really great!

Update Product

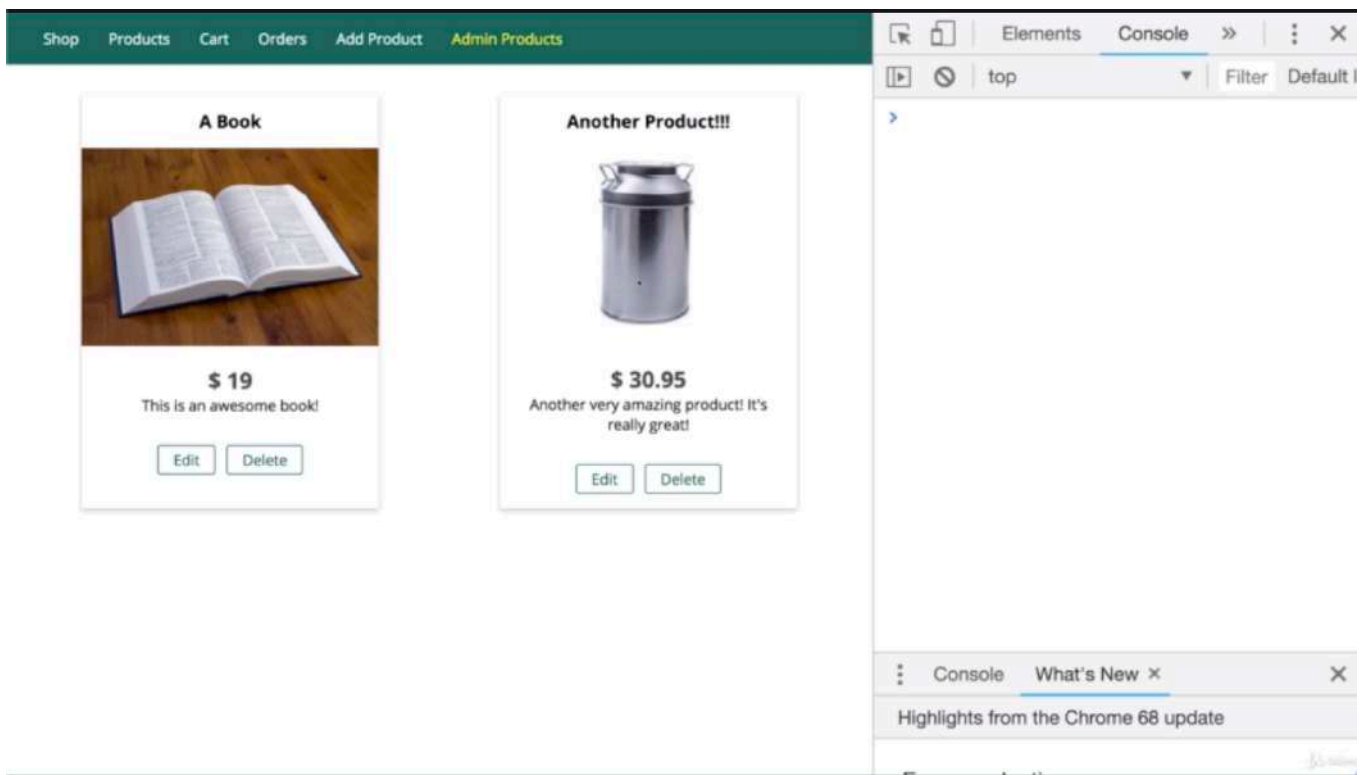
ElementsConsole>>⋮×

topFilterDefault

>

⋮ ConsoleWhat's New ××

Highlights from the Chrome 68 update



```

1 <!--./views/admin/products.ejs-->
2
3 <%- include('../includes/head.ejs') %>
4   <link rel="stylesheet" href="/css/product.css">
5   </head>
6
7   <body>
8     <%- include('../includes/navigation.ejs') %>
9
10    <main>
11      <% if (prods.length > 0) { %>
12        <div class="grid">
13          <% for (let product of prods) { %>
14            <article class="card product-item">
15              <header class="card_header">
16                <h1 class="product_title">
17                  <%= product.title %>
18                </h1>
19              </header>
20              <div class="card_image">
21                ">
22              </div>
23              <div class="card_content">
24                <h2 class="product_price">$
25                  <%= product.price %>
26                </h2>
27                <p class="product_description">
28                  <%= product.description %>
29                </p>
30              </div>
31              <div class="card_actions">
32                <a href="/admin/edit-product/<%= product.id %>?
edit=true" class="btn">Edit</a>

```

```

33     <form action="/admin/delete-product" method="POST">
34         <button class="btn" type="submit">Delete</button>
35     </form>
36
37     </div>
38 </article>
39 <% } %>
40 </div>
41 <% } else { %>
42     <h1>No Products Found!</h1>
43     <% } %>
44 </main>
45 <%- include('../includes/end.ejs') %>

```

```

1 <!--../views/admin/edit-product.ejs-->
2
3 <%- include('../includes/head.ejs') %>
4     <link rel="stylesheet" href="/css/forms.css">
5     <link rel="stylesheet" href="/css/product.css">
6 </head>
7
8 <body>
9     <%- include('../includes/navigation.ejs') %>
10
11     <main>
12         <form class="product-form" action="/admin/<% if (editing) { %>edit-product<% } else
13 { %>add-product<% } %>" method="POST">
14             <div class="form-control">
15                 <label for="title">Title</label>
16                 <input type="text" name="title" id="title" value="<% if (editing) { %><%=
17 product.title %><% } %>">
18             </div>
19             <div class="form-control">
20                 <label for="imageUrl">Image URL</label>
21                 <input type="text" name="imageUrl" id="imageUrl" value="<% if (editing) { %>
22 <%= product.imageUrl %><% } %>">
23             </div>
24             <div class="form-control">
25                 <label for="price">Price</label>
26                 <input type="number" name="price" id="price" step="0.01" value="<% if
27 (editing) { %><%= product.price %><% } %>">
28             </div>
29             <div class="form-control">
30                 <label for="description">Description</label>
31                 <textarea name="description" id="description" rows="5"><% if (editing) { %>
32 <%= product.description %><% } %></textarea>
33             </div>
34             <% if (editing) { %>
35                 <!--now i can extract it by that name in the incoming request in my
36 controller
37 so req.body.productId in ./controllers/admin.js
38 -->
39                 <input type="hidden" value="<%= product.id %>" name="productId">
40                 <% } %>
41
42                 <button class="btn" type="submit"><% if (editing) { %>Update Product<% } else {
43 %>Add Product<% } %></button>

```

```
37     </form>
38 </main>
39 <%- include('../includes/end.ejs') %>
```

```
1 // ./routes/admin.js
2
3 const path = require('path');
4
5 const express = require('express');
6
7 const adminController = require('../controllers/admin');
8
9 const router = express.Router();
10
11 // /admin/add-product => GET
12 router.get('/add-product', adminController.getAddProduct);
13
14 // /admin/products => GET
15 router.get('/products', adminController.getProducts);
16
17 // /admin/add-product => POST
18 router.post('/add-product', adminController.postAddProduct);
19
20 router.get('/edit-product/:productId', adminController.getEditProduct);
21
22 router.post('/edit-product', adminController.postEditProduct);
23
24 module.exports = router;
```

```
1 // ./controllers/admin.js
2
3 const Product = require('../models/product');
4
5 exports.getAddProduct = (req, res, next) => {
6   res.render('admin/edit-product', {
7     pageTitle: 'Add Product',
8     path: '/admin/add-product',
9     editing: false
10  });
11 };
12
13 exports.postAddProduct = (req, res, next) => {
14   const title = req.body.title;
15   const imageUrl = req.body.imageUrl;
16   const price = req.body.price;
17   const description = req.body.description;
18   /**when adding a new product in 'postAddProduct',
19    * we also need to set null as an ID as a first argument here on our product constructor
20    * because we just added this as an additional argument in the constructor in
21    ./models/product.js
22    * 'constructor(id, title, imageUrl, description, price){}'
23    * so if 'id' is null,
24    * then this check will fail
25    * and we will therefore make it into the new product created mode which is what we want.
26    */
27   const product = new Product(null, title, imageUrl, description, price);
28   product.save();
29 }
```

```

28   res.redirect('/');
29 };
30
31 exports.getEditProduct = (req, res, next) => {
32   const editMode = req.query.edit;
33   if (!editMode) {
34     return res.redirect('/');
35   }
36   const prodId = req.params.productId;
37   Product.findById(prodId, product => {
38     if (!product) {
39       return res.redirect('/');
40     }
41     res.render('admin/edit-product', {
42       pageTitle: 'Edit Product',
43       path: '/admin/edit-product',
44       editing: editMode,
45       product: product
46     });
47   });
48 };
49
50 exports.postEditProduct = (req, res, next) => {
51   /**we wanna construct a new product
52    * and replace the existing one with this product
53    * this means that we have to do some work on the product model.
54    */
55
56   /**these keys(productId, title, price, imageUrl)
57    * have to match the names you have on your inputs in your added product view
58    */
59   const prodId = req.body.productId;
60   const updatedTitle = req.body.title;
61   const updatedPrice = req.body.price;
62   const updatedImageUrl = req.body.imageUrl;
63   const updatedDesc = req.body.description;
64   /**instantiate a new product
65    * therefore and i do pass my existing 'prodId' as first argument
66    * and this will ensure that in the product model in ./models/product.js file
67    * in this check, we find a valid ID
68    * and therefore we go into this updating mode(if) instead of the add mode(else)
69    */
70   const updatedProduct = new Product(
71     prodId,
72     updatedTitle,
73     updatedImageUrl,
74     updatedDesc,
75     updatedPrice
76   );
77   updatedProduct.save();
78   res.redirect('/admin/products');
79 };
80
81 exports.getProducts = (req, res, next) => {
82   Product.fetchAll(products => {
83     res.render('admin/products', {

```

```

84     prods: products,
85     pageTitle: 'Admin Products',
86     path: '/admin/products'
87   });
88 });
89 };
90

```

```

1  //./models/product.js
2
3  const fs = require('fs');
4  const path = require('path');
5
6  const p = path.join(
7    path.dirname(process.mainModule.filename),
8    'data',
9    'products.json'
10 );
11
12 const getProductsFromFile = cb => {
13   fs.readFile(p, (err, fileContent) => {
14     if (err) {
15       cb([]);
16     } else {
17       cb(JSON.parse(fileContent));
18     }
19   });
20 };
21
22 module.exports = class Product {
23   constructor(id, title, imageUrl, description, price) {
24     /**when creating a new product,
25     * we should accept an ID too
26     * and then set this ID equal to ID
27     * but we will simply pass 'null' for a brand new product,
28     * so that we can still create products that don't have ID yet,
29     * then the ID will be assigned to 'this.id = Math.random().toString()'
30     * but if we are editing one,
31     * we do have the ID already, so we can simply assign it ID here.
32     */
33     this.id = id;
34     this.title = title;
35     this.imageUrl = imageUrl;
36     this.description = description;
37     this.price = price;
38   }
39   /**now we have a 'save()'function
40   * that we can use both for adding new products
41   * or editing existing products. */
42   save() {
43     getProductsFromFile(products => {
44       if (this.id) {
45         const existingProductIndex = products.findIndex(
46           prod => prod.id === this.id
47         );
48         const updatedProducts = [...products];
49         /**on that array updatedProducts,

```

```

50      * i will replace my existingProductIndex with 'this'
51      * because 'this' inside of class is the updatedProduct
52      * because you have to imagine that i create a new product instance
53      * i will populate it with information about my existingProduct
54      * and then i just call save
55      * and i will find out that i already have this product
56      * and therefore i just replace it in the array
57      * which is stored in the file with the newly created product i'm in.
58      *
59      * so with that being saved,
60      * i have to write that information to the file
61      * so fs.writeFile is what i need to execute.
62      */
63      updatedProducts[existingProductIndex] = this;
64      fs.writeFile(p, JSON.stringify(updatedProducts), err => {
65          console.log(err);
66      });
67  } else {
68      this.id = Math.random().toString();
69      products.push(this);
70      fs.writeFile(p, JSON.stringify(products), err => {
71          console.log(err);
72      });
73  }
74  });
75  }
76
77  static fetchAll(cb) {
78      getProductsFromFile(cb);
79  }
80
81  static findById(id, cb) {
82      getProductsFromFile(products => {
83          const product = products.find(p => p.id === id);
84          cb(product);
85      });
86  }
87  };

```

* Chapter 125: Deleting Cart Items

1. update

- ./routes/admin.js
- ./views/admin/product.ejs
- ./controllers/admin.js
- ./models/product.js
- ./models/cart.js
- ./data/cart.json

EXPLORER controllers admin.js controllers product-detail.ejs main.css product.js cart.js

This will be fixed later - "cart" doesn't exist here, we'll need to parse that from fileContent.

```
41 static deleteProduct(id, productPrice) {
42   fs.readFile(p, (err, fileContent) => {
43     if (err) {
44       return;
45     }
46     const updatedCart = { ...cart };
47   });
48 }
49 };
50
```

shop.js data models cart.js product.js node_modules public routes admin.js shop.js util views admin edit-product.ejs products... M includes add-to-cart.ejs end.ejs head.ejs

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: node

```
null
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
null
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
```

08-dynamic-routes* 0 0 Ln 46, Col 39 Spaces: 4 UTF-8 LF JavaScript Prettier ✓

Shop Products Cart Orders Add Product Admin Products

Title
Test

Image URL
data:image/jpeg;base64,/9j/4AAQSkZJRgAE

Price
20.99

Description
Test

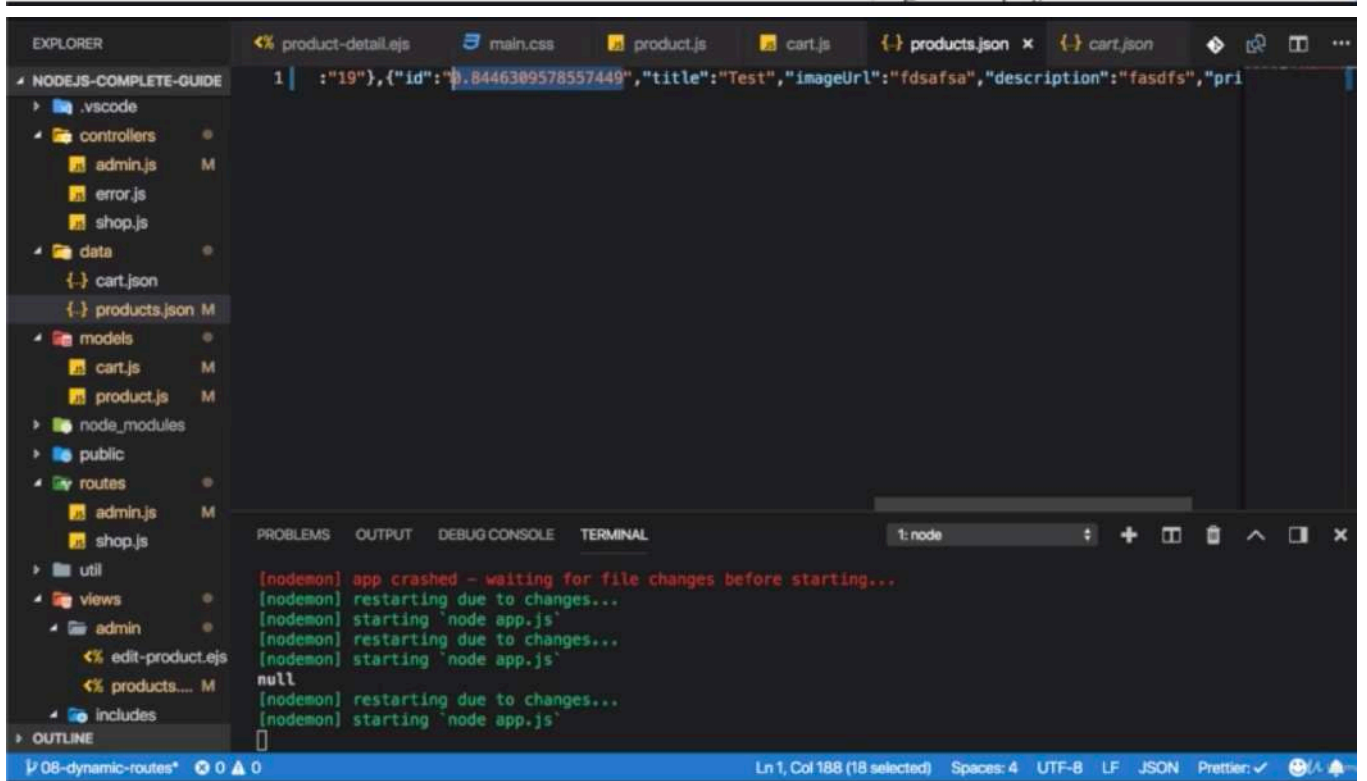
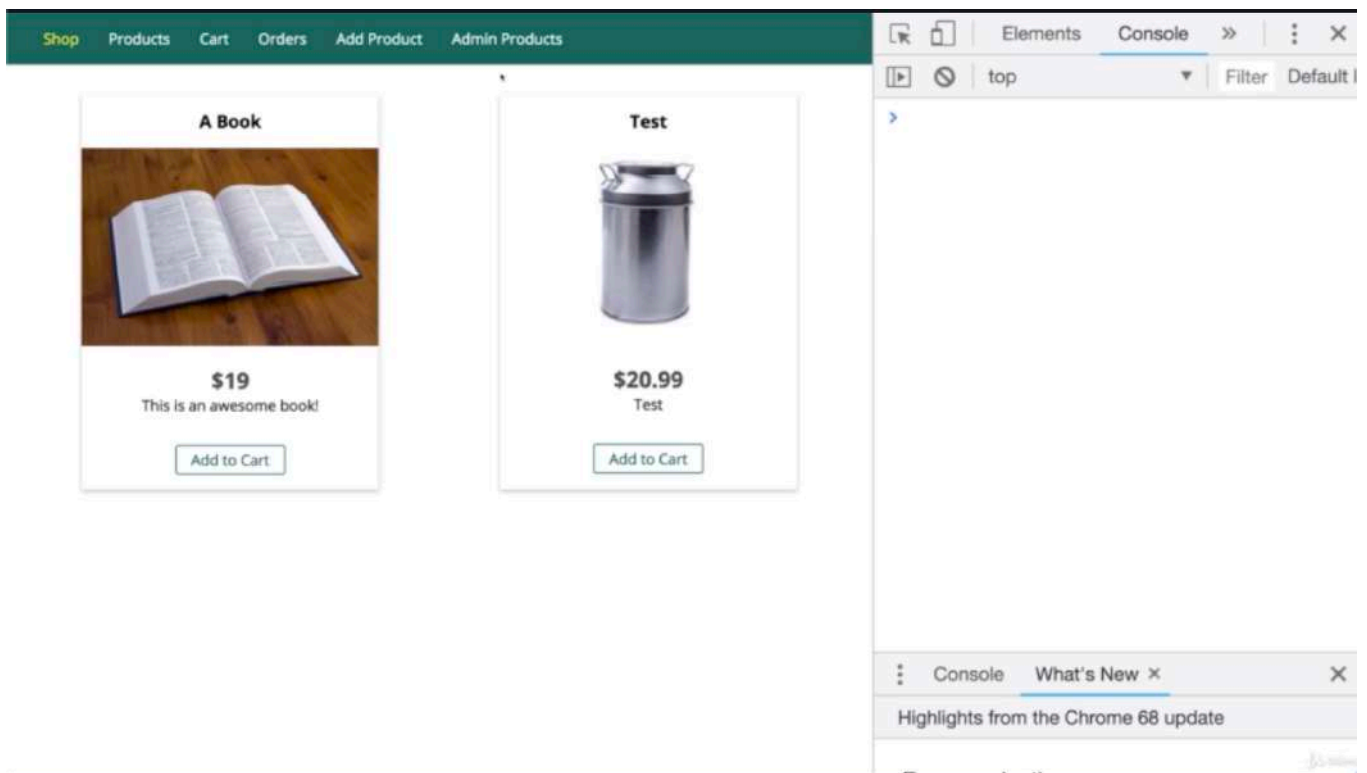
Add Product

Elements Console >> Filter Default

top


Console What's New x

Highlights from the Chrome 68 update



ShopProductsCartOrdersAdd ProductAdmin Products

A Book



\$ 19

This is an awesome book!

EditDelete

Test

\$ 20.99

fasdfs

EditDelete

Console

top

GET http://localhost:3000/products/1060/admin/fdsafsa 404 (Not Found)


Console

What's New

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

A Book



\$ 19

This is an awesome book!

EditDelete

Elements

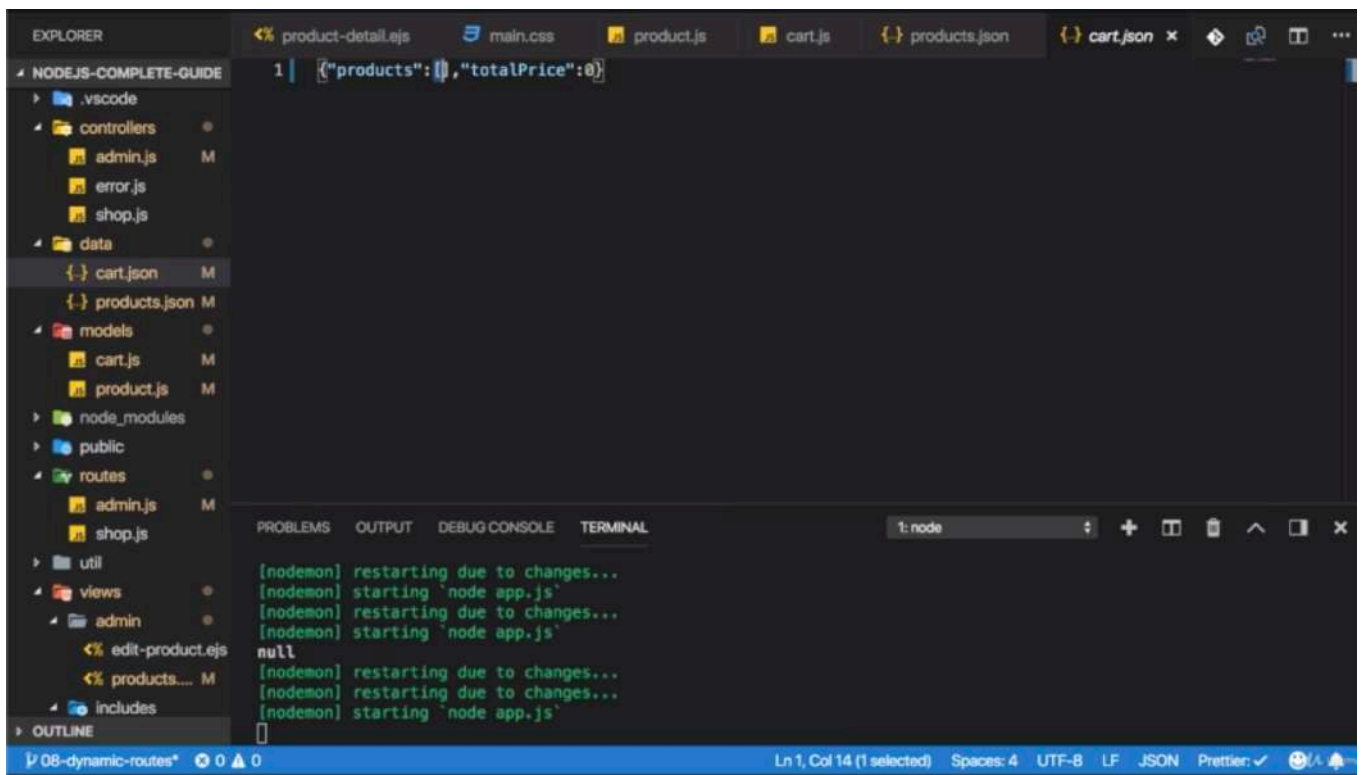
Console

top

Console

What's New

Highlights from the Chrome 68 update



```
1 // ./routes/admin.js
2
3 const path = require('path');
4
5 const express = require('express');
6
7 const adminController = require('../controllers/admin');
8
9 const router = express.Router();
10
11 // /admin/add-product => GET
12 router.get('/add-product', adminController.getAddProduct);
13
14 // /admin/products => GET
15 router.get('/products', adminController.getProducts);
16
17 // /admin/add-product => POST
18 router.post('/add-product', adminController.postAddProduct);
19
20 router.get('/edit-product/:productId', adminController.getEditProduct);
21
22 router.post('/edit-product', adminController.postEditProduct);
23
24 router.post('/delete-product', adminController.postDeleteProduct);
25
26 module.exports = router;
```

```
1 <!--./views/admin/products.ejs-->
2
3 <%- include('../includes/head.ejs') %>
4   <link rel="stylesheet" href="/css/product.css">
5   </head>
6
7   <body>
8     <%- include('../includes/navigation.ejs') %>
```

```

9
10 <main>
11 <% if (prods.length > 0) { %>
12 <div class="grid">
13 <% for (let product of prods) { %>
14 <article class="card product-item">
15 <header class="card_header">
16 <h1 class="product_title">
17 <%= product.title %>
18 </h1>
19 </header>
20 <div class="card_image">
21 ">
22 </div>
23 <div class="card_content">
24 <h2 class="product_price">$
25 <%= product.price %>
26 </h2>
27 <p class="product_description">
28 <%= product.description %>
29 </p>
30 </div>
31 <div class="card_actions">
32 <a href="/admin/edit-product/<%= product.id %>?
edit=true" class="btn">Edit</a>
33 <form action="/admin/delete-product" method="POST">
34 <!--set the value to productId using ejs templating
syntax and the name to productId
35 so that we can extract that information by that
name.
36 -->
37 <input type="hidden" value="<%= product.id %>"
name="productId">
38 <button class="btn" type="submit">Delete</button>
39 </form>
40
41 </div>
42 </article>
43 <% } %>
44 </div>
45 <% } else { %>
46 <h1>No Products Found!</h1>
47 <% } %>
48 </main>
49 <%- include('../includes/end.ejs') %>

```

```

1 // ../controllers/admin.js
2
3 const Product = require('../models/product');
4
5 exports.getAddProduct = (req, res, next) => {
6   res.render('admin/edit-product', {
7     pageTitle: 'Add Product',
8     path: '/admin/add-product',
9     editing: false
10  });

```

```

11 };
12
13 exports.postAddProduct = (req, res, next) => {
14   const title = req.body.title;
15   const imageUrl = req.body.imageUrl;
16   const price = req.body.price;
17   const description = req.body.description;
18   const product = new Product(null, title, imageUrl, description, price);
19   product.save();
20   res.redirect('/');
21 };
22
23 exports.getEditProduct = (req, res, next) => {
24   const editMode = req.query.edit;
25   if (!editMode) {
26     return res.redirect('/');
27   }
28   const prodId = req.params.productId;
29   Product.findById(prodId, product => {
30     if (!product) {
31       return res.redirect('/');
32     }
33     res.render('admin/edit-product', {
34       pageTitle: 'Edit Product',
35       path: '/admin/edit-product',
36       editing: editMode,
37       product: product
38     });
39   });
40 };
41
42 exports.postEditProduct = (req, res, next) => {
43   const prodId = req.body.productId;
44   const updatedTitle = req.body.title;
45   const updatedPrice = req.body.price;
46   const updatedImageUrl = req.body.imageUrl;
47   const updatedDesc = req.body.description;
48   const updatedProduct = new Product(
49     prodId,
50     updatedTitle,
51     updatedImageUrl,
52     updatedDesc,
53     updatedPrice
54   );
55   /**the callback in 'save()' would be best for redirecting */
56   updatedProduct.save();
57   res.redirect('/admin/products');
58 };
59
60 exports.getProducts = (req, res, next) => {
61   Product.fetchAll(products => {
62     res.render('admin/products', {
63       prods: products,
64       pageTitle: 'Admin Products',
65       path: '/admin/products'
66     });

```

```

67   });
68 };
69
70 exports.postDeleteProduct = (req, res, next) => {
71   const prodId = req.body.productId;
72   /**it would be best if we have a callback in deleteById
73    * so that only redirect once we are done
74    * the same is also true for updating.
75    */
76   Product.deleteById(prodId);
77   res.redirect('/admin/products');
78 };
79

```

```

1  //./models/product.js
2
3  const fs = require('fs');
4  const path = require('path');
5
6  const Cart = require('./cart');
7
8  const p = path.join(
9    path.dirname(process.mainModule.filename),
10   'data',
11   'products.json'
12 );
13
14 const getProductsFromFile = cb => {
15   fs.readFile(p, (err, fileContent) => {
16     if (err) {
17       cb([]);
18     } else {
19       cb(JSON.parse(fileContent));
20     }
21   });
22 };
23
24 module.exports = class Product {
25   constructor(id, title, imageUrl, description, price) {
26     this.id = id;
27     this.title = title;
28     this.imageUrl = imageUrl;
29     this.description = description;
30     this.price = price;
31   }
32
33   save() {
34     getProductsFromFile(products => {
35       if (this.id) {
36         const existingProductIndex = products.findIndex(
37           prod => prod.id === this.id
38         );
39         const updatedProducts = [...products];
40         updatedProducts[existingProductIndex] = this;
41         fs.writeFile(p, JSON.stringify(updatedProducts), err => {
42           console.log(err);
43         });

```

```

44     } else {
45         this.id = Math.random().toString();
46         products.push(this);
47         fs.writeFile(p, JSON.stringify(products), err => {
48             console.log(err);
49         });
50     }
51 });
52 }
53
54 static deleteById(id) {
55     getProductsFromFile(products => {
56         const product = products.find(prod => prod.id === id);
57         /**'filter' method also take an anonymous function
58          * and will return me all elements as part of a new array
59          * that do match the criteria, my function returns.
60          * so if this returns true, the element is kept.
61          *
62          * now i wanna keep all elements where the ID of the element is not equal to the ID
        i'm trying to delete
63         * because all elements where the ID is not equal should be kept around,
64         * should be part of the new array which will be the array i save back to my file
65         *
66         * i wanna keep the item
67         * so this will then return true if the ID are not equal,
68         * therefore the item is kept
69         * and only for that single product i'm looking for.
70         * this will be false.
71         */
72         const updatedProducts = products.filter(prod => prod.id !== id);
73         fs.writeFile(p, JSON.stringify(updatedProducts), err => {
74             if (!err) {
75                 Cart.deleteProduct(id, product.price);
76             }
77         });
78     });
79 }
80
81 static fetchAll(cb) {
82     getProductsFromFile(cb);
83 }
84
85 static findById(id, cb) {
86     getProductsFromFile(products => {
87         const product = products.find(p => p.id === id);
88         cb(product);
89     });
90 }
91 };
92

```

```

1  //./models/cart.js
2
3  const fs = require('fs');
4  const path = require('path');
5
6  const p = path.join(

```

```

7  path.dirname(process.mainModule.filename),
8  'data',
9  'cart.json'
10 );
11
12 module.exports = class Cart {
13   static addProduct(id, productPrice) {
14     // Fetch the previous cart
15     fs.readFile(p, (err, fileContent) => {
16       let cart = { products: [], totalPrice: 0 };
17       if (!err) {
18         cart = JSON.parse(fileContent);
19       }
20       // Analyze the cart => Find existing product
21       const existingProductIndex = cart.products.findIndex(
22         prod => prod.id === id
23       );
24       const existingProduct = cart.products[existingProductIndex];
25       let updatedProduct;
26       // Add new product/ increase quantity
27       if (existingProduct) {
28         updatedProduct = { ...existingProduct };
29         updatedProduct.qty = updatedProduct.qty + 1;
30         cart.products = [...cart.products];
31         cart.products[existingProductIndex] = updatedProduct;
32       } else {
33         updatedProduct = { id: id, qty: 1 };
34         cart.products = [...cart.products, updatedProduct];
35       }
36       cart.totalPrice = cart.totalPrice + +productPrice;
37       fs.writeFile(p, JSON.stringify(cart), err => {
38         console.log(err);
39       });
40     });
41   }
42   /**we got 'static addProduct()' method
43    * now i need a new static method which is deleteProduct
44    *
45    */
46   static deleteProduct(id, productPrice) {
47     fs.readFile(p, (err, fileContent) => {
48       if (err) {
49         /**if i got an error,
50          * i can already return
51          * because that simply means somehow i didn't find a cart,
52          * so there is nothing to delete
53          * so i can just ignore this.
54          */
55         return;
56       }
57       const updatedCart = { ...JSON.parse(fileContent) };
58       const product = updatedCart.products.find(prod => prod.id === id);
59       const productQty = product.qty;
60       updatedCart.products = updatedCart.products.filter(
61         prod => prod.id !== id
62       );

```



```

63     updatedCart.totalPrice =
64         updatedCart.totalPrice - productPrice * productQty;
65
66     fs.writeFile(p, JSON.stringify(updatedCart), err => {
67         console.log(err);
68     });
69 });
70 }
71
72 static getCart(cb) {
73     fs.readFile(p, (err, fileContent) => {
74         const cart = JSON.parse(fileContent);
75         if (err) {
76             cb(null);
77         } else {
78             cb(cart);
79         }
80     });
81 }
82 };
83

```

```

1  ../data/cart.json
2
3  {"products":[],"totalPrice":0}

```

* Chapter 126: Displaying Cart Items On The Cart Page

1. update
 - ./views/shop/cart.ejs
 - ./models/cart.js
 - ./controllers/shop.js

[Shop](#) [Products](#) [Cart](#) [Orders](#) [Add Product](#) [Admin Products](#)

No Products in Cart!

Elements Console >> ⋮ ✕

top Filter Default


>

⋮ Console What's New ✕

Highlights from the Chrome 68 update

[Shop](#) [Products](#) [Cart](#) [Orders](#) [Add Product](#) [Admin Products](#)

A Book



\$ 19

This is an awesome book!

[Details](#) [Add to Cart](#)

Elements Console >> ⋮ ✕

top Filter Default

>

⋮ Console What's New ✕

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

A Book (1)

ElementsConsole>>⋮✕

topFilterDefault


>

ConsoleWhat's New ✕✕

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

A Book



\$ 19

This is an awesome book!

DetailsAdd to Cart

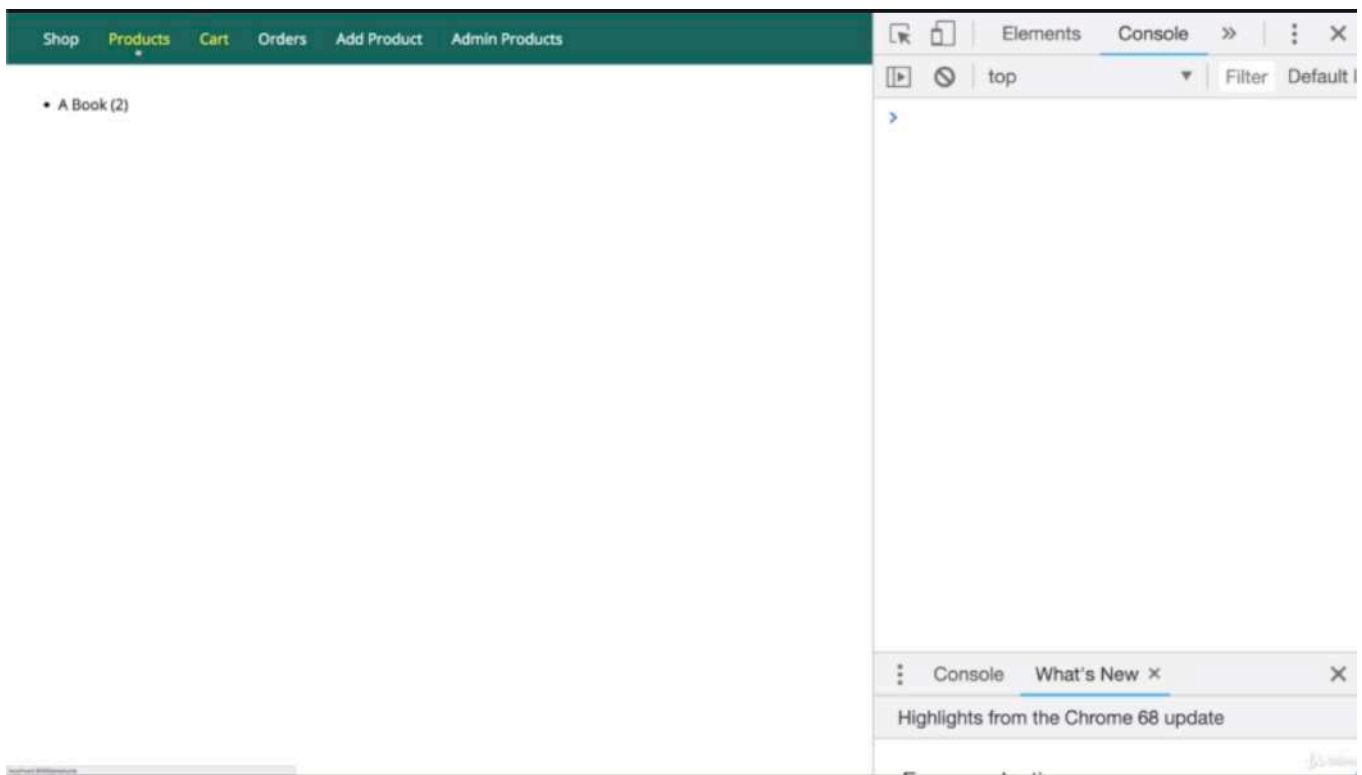
ElementsConsole>>⋮✕

topFilterDefault

>

ConsoleWhat's New ✕✕

Highlights from the Chrome 68 update



```

1 <!--./views/shop/cart.ejs-->
2
3 <%- include('../includes/head.ejs') %>
4   </head>
5
6   <body>
7     <%- include('../includes/navigation.ejs') %>
8     <main>
9       <!--'products' in here is 'products' in the 'getCart()'
./controllers/shop.js
10       -->
11       <% if (products.length > 0) { %>
12         <ul>
13           <!--
14             'p' in the 'forEach()' does hold or does refer to a
product with a productData field and quantity field
15             and the productData field is the 'product' i'm
interested in.
16             but i also need the quantity
17             -->
18             <% products.forEach(p => { %>
19               <li><%= p.productData.title %> (<%= p.qty %>)</li>
20             <% })%>
21           </ul>
22         <% } else { %>
23           <h1>No Products in Cart!</h1>
24         <% } %>
25       </main>
26       <%- include('../includes/end.ejs') %>

```

```

1 //./controllers/shop.js
2
3 const Product = require('../models/product');
4 const Cart = require('../models/cart');
5

```

```

6 exports.getProducts = (req, res, next) => {
7   Product.fetchAll(products => {
8     res.render('shop/product-list', {
9       prods: products,
10      pageTitle: 'All Products',
11      path: '/products'
12    });
13  });
14 };
15
16 exports.getProduct = (req, res, next) => {
17   const prodId = req.params.productId;
18   Product.findById(prodId, product => {
19     res.render('shop/product-detail', {
20       product: product,
21       pageTitle: product.title,
22       path: '/products'
23     })
24   });
25 };
26
27 exports.getIndex = (req, res, next) => {
28   Product.fetchAll(products => {
29     res.render('shop/index', {
30       prods: products,
31       pageTitle: 'Shop',
32       path: '/'
33     });
34   });
35 };
36
37 exports.getCart = (req, res, next) => {
38   Cart.getCart(cart => {
39     Product.fetchAll(products => {
40       const cartProducts = []
41       for(product of products){
42         const cartProductData = cart.products.find(
43           prod => prod.id === product.id
44         )
45         if(cartProductData){
46           /**this means that after the loop is done,
47            * i will have an array of cartProducts that contains all the products
48            * which are indeed part of the cart.
49            */
50           cartProducts.push({productData: product, qty: cartProductData.qty});
51         }
52       }
53       res.render('shop/cart', {
54         path: '/cart',
55         pageTitle: 'Your Cart',
56         products: cartProducts
57       });
58     })
59   })
60 };
61

```

```

62 exports.postCart = (req, res, next) => {
63   const prodId = req.body.productId;
64   Product.findById(prodId, product => {
65     Cart.addProduct(prodId, product.price)
66   })
67   res.redirect('/cart')
68 }
69
70 exports.getOrders = (req, res, next) => {
71   res.render('shop/orders', {
72     path: '/orders',
73     pageTitle: 'Your Orders'
74   });
75 };
76
77 exports.getCheckout = (req, res, next) => {
78   res.render('shop/checkout', {
79     path: '/checkout',
80     pageTitle: 'Checkout'
81   });
82 };

```

```

1  //./models/cart.js
2
3  const fs = require('fs');
4  const path = require('path');
5
6  const p = path.join(
7    path.dirname(process.mainModule.filename),
8    'data',
9    'cart.json'
10 );
11
12 module.exports = class Cart {
13   static addProduct(id, productPrice) {
14     // Fetch the previous cart
15     fs.readFile(p, (err, fileContent) => {
16       let cart = { products: [], totalPrice: 0 };
17       if (!err) {
18         cart = JSON.parse(fileContent);
19       }
20       // Analyze the cart => Find existing product
21       const existingProductIndex = cart.products.findIndex(
22         prod => prod.id === id
23       );
24       const existingProduct = cart.products[existingProductIndex];
25       let updatedProduct;
26       // Add new product/ increase quantity
27       if (existingProduct) {
28         updatedProduct = { ...existingProduct };
29         updatedProduct.qty = updatedProduct.qty + 1;
30         cart.products = [...cart.products];
31         cart.products[existingProductIndex] = updatedProduct;
32       } else {
33         updatedProduct = { id: id, qty: 1 };
34         cart.products = [...cart.products, updatedProduct];
35       }

```

```

36     cart.totalPrice = cart.totalPrice + +productPrice;
37     fs.writeFile(p, JSON.stringify(cart), err => {
38         console.log(err);
39     });
40 });
41 }
42 /**we got 'static addProduct()' method
43  * now i need a new static method which is deleteProduct
44  *
45  */
46 static deleteProduct(id, productPrice) {
47     fs.readFile(p, (err, fileContent) => {
48         if (err) {
49             /**if i got an error,
50              * i can already return
51              * because that simply means somehow i didn't find a cart,
52              * so there is nothing to delete
53              * so i can just ignore this.
54              */
55             return;
56         }
57         const updatedCart = { ...JSON.parse(fileContent) };
58         const product = updatedCart.products.find(prod => prod.id === id);
59         const productQty = product.qty;
60         updatedCart.products = updatedCart.products.filter(
61             prod => prod.id !== id
62         );
63         updatedCart.totalPrice =
64             updatedCart.totalPrice - productPrice * productQty;
65
66         fs.writeFile(p, JSON.stringify(updatedCart), err => {
67             console.log(err);
68         });
69     });
70 }
71
72 static getCart(cb) {
73     fs.readFile(p, (err, fileContent) => {
74         const cart = JSON.parse(fileContent);
75         if (err) {
76             cb(null);
77         } else {
78             cb(cart);
79         }
80     });
81 }
82 /**i need to get a callback
83  * therefore which i can call once i got the products
84  * and then i can use this function to read the file
85  */
86 static getCart(cb){
87     fs.readFile(p, (err, fileContent) => {
88         const cart = JSON.parse(fileContent)
89         if(err){
90             cb(null)
91         } else {

```

```
92         cb(cart)
93     }
94 }
95 }
96 };
97
```

* Chapter 128: Fixing A Delete Product Bug

- 1. update
- ./views/shop/cart.ejs
- ./routes/shop.js
- ./controllers/shop.js
- ./models/cart.js

ShopProductsCartOrdersAdd ProductAdmin Products

A Book (3)

Delete

Console

top

[Intervention] Slow network is detected. See <https://www.chromestatus.com/feature/5636954674692896> for more details. Fallback font will be used while loading: <https://fonts.gstatic.com/s/opensans/v15/mem8YaGs126MiZpBA-UfVZ@bf8pkAg.woff2>

ConsoleWhat's New

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

No Products in Cart!

Console

top

[Intervention] Slow network is detected. See <https://www.chromestatus.com/feature/5636954674692896> for more details. Fallback font will be used while loading: <https://fonts.gstatic.com/s/opensans/v15/mem8YaGs126MiZpBA-UfVZ@bf8pkAg.woff2>

[Intervention] Slow network is detected. See <https://www.chromestatus.com/feature/5636954674692896> for more details. Fallback font will be used while loading: <https://fonts.gstatic.com/s/opensans/v15/mem5YaGs126MiZpBA-UN7rg0UuhpKKSTjw.woff2>

ConsoleWhat's New

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

Title

fadsfdsa

Image URL

dsafs

Price

122

Description

dsffsd

Add Product

ElementsConsole»⋮×

top▼FilterDefault

✖ GET http://localhost:3000 localhost/:980/dsafs 404 (Not Found)


>

⋮ ConsoleWhat's New ××

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

A Book




\$19

This is an awesome book!

Add to Cart

fadsfdsa



\$122

dsffsd

Add to Cart

Console»⋮×

top▼FilterDefault

✖ GET http://localhost:3000 localhost/:980/dsafs 404 (Not Found)

✖ [Intervention] Slow network is detected. See https://www.chromestatus.com/feature/5636954674692096 for more details. Fallback font will be used while loading: https://fonts.gstatic.com/s/opensans/v15/mem8YaGs126MiZpBA-UfVZ@bf8pkAg.woff2

✖ [Intervention] Slow network is detected. See https://www.chromestatus.com/feature/5636954674692096 for more details. Fallback font will be used while loading: https://fonts.gstatic.com/s/opensans/v15/mem5YaGs126MiZpBA-UN7rgOUhpkKSTjw.woff2


>

⋮ ConsoleWhat's New ××

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

A Book




\$19

This is an awesome book!

Add to Cart

fadsfsda



\$122

dsffsd

Add to Cart

Console

top

GET http://localhost:3000 localhost/:980/dsafs 404 (Not Found)

[Intervention] Slow network is detected. See https://www.chromestatus.com/feature/5636954674692096 for more details. Fallback font will be used while loading: https://fonts.gstatic.com/s/opensans/v15/mem8YaGs126MiZpBA-UFVZ0bf8pkAg.woff2

[Intervention] Slow network is detected. See https://www.chromestatus.com/feature/5636954674692096 for more details. Fallback font will be used while loading: https://fonts.gstatic.com/s/opensans/v15/mem5YaGs126MiZpBA-UN7rg0UuhpKKSTjw.woff2

ConsoleWhat's New

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

A Book (1)

Delete

Console

top


[Intervention] Slow network is detected. See https://www.chromestatus.com/feature/5636954674692096 for more details. Fallback font will be used while loading: https://fonts.gstatic.com/s/opensans/v15/mem8YaGs126MiZpBA-UFVZ0bf8pkAg.woff2

ConsoleWhat's New

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

A Book




\$ 19

This is an awesome book!

DetailsAdd to Cart

fadsfdsa



\$ 122

dsffsd

DetailsAdd to Cart

Console

top

GET http://localhost:3000/products:1050/dsafs 404 (Not Found)

[Intervention] Slow network is detected. See https://www.chromestatus.com/feature/5636954674692096 for more details. Fallback font will be used while loading: https://fonts.gstatic.com/s/opensans/v15/mem8YagS126MiZpBA-UFVZ0bf8pkAg.woff2

[Intervention] Slow network is detected. See https://www.chromestatus.com/feature/5636954674692096 for more details. Fallback font will be used while loading: https://fonts.gstatic.com/s/opensans/v15/mem5YagS126MiZpBA-UN7rg0UuhpKKSTjw.woff2

ConsoleWhat's New

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

A Book (1)

Delete

fadsfdsa (1)

Delete

Console

top

[Intervention] Slow network is detected. See https://www.chromestatus.com/feature/5636954674692096 for more details. Fallback font will be used while loading: https://fonts.gstatic.com/s/opensans/v15/mem8YagS126MiZpBA-UFVZ0bf8pkAg.woff2

ConsoleWhat's New

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

- A Book (1)

Delete
- fadsfdsa (1)

Delete

ElementsConsole»⋮×

topFilterDefault

>

ConsoleWhat's New ××

Highlights from the Chrome 68 update

ShopProductsCartOrdersAdd ProductAdmin Products

- fadsfdsa (1)

Delete

Console»⋮×

topFilterDefault

[Intervention] Slow network is detected. See <https://www.chromestatus.com/feature/5636954674692896> for more details. Fallback font will be used while loading: <https://fonts.gstatic.com/s/opensans/v15/mem8YagS126MiZpBA-UfVZ@bf8pkAg.woff2>

>

ConsoleWhat's New ××

Highlights from the Chrome 68 update

EXPLORER

product-detail.ejs main.css product.js cart.js cart.json x products.json

1 | {"products":[{"id":"0.20178551512777343","qty":1},"totalPrice":122]}

NODEJS-COMPLETE-GUIDE

- .vscode
- controllers
 - admin.js
 - error.js
 - shop.js M
- data
 - cart.json M
 - products.json M
- models
 - cart.js
 - product.js
- node_modules
- public
- routes
 - admin.js
 - shop.js M
- util
- views
 - admin
 - edit-product.ejs
 - products.ejs
 - includes

OUTLINE

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: node

```
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
null
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
null
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
```

Ln 1, Col 3 (12 selected) Spaces: 4 UTF-8 LF JSON Prettier: ✓

Shop Products Cart Orders Add Product Admin Products

• fadsfdsa (1)

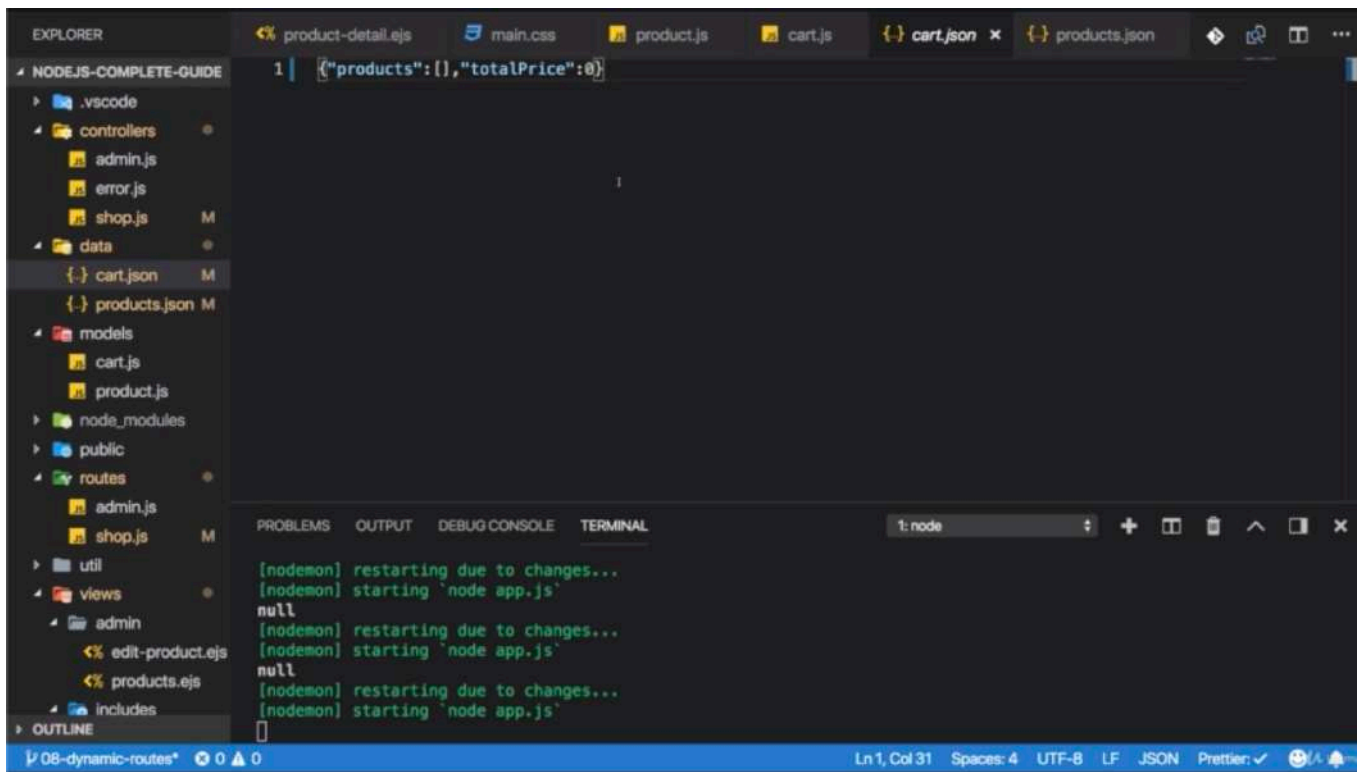
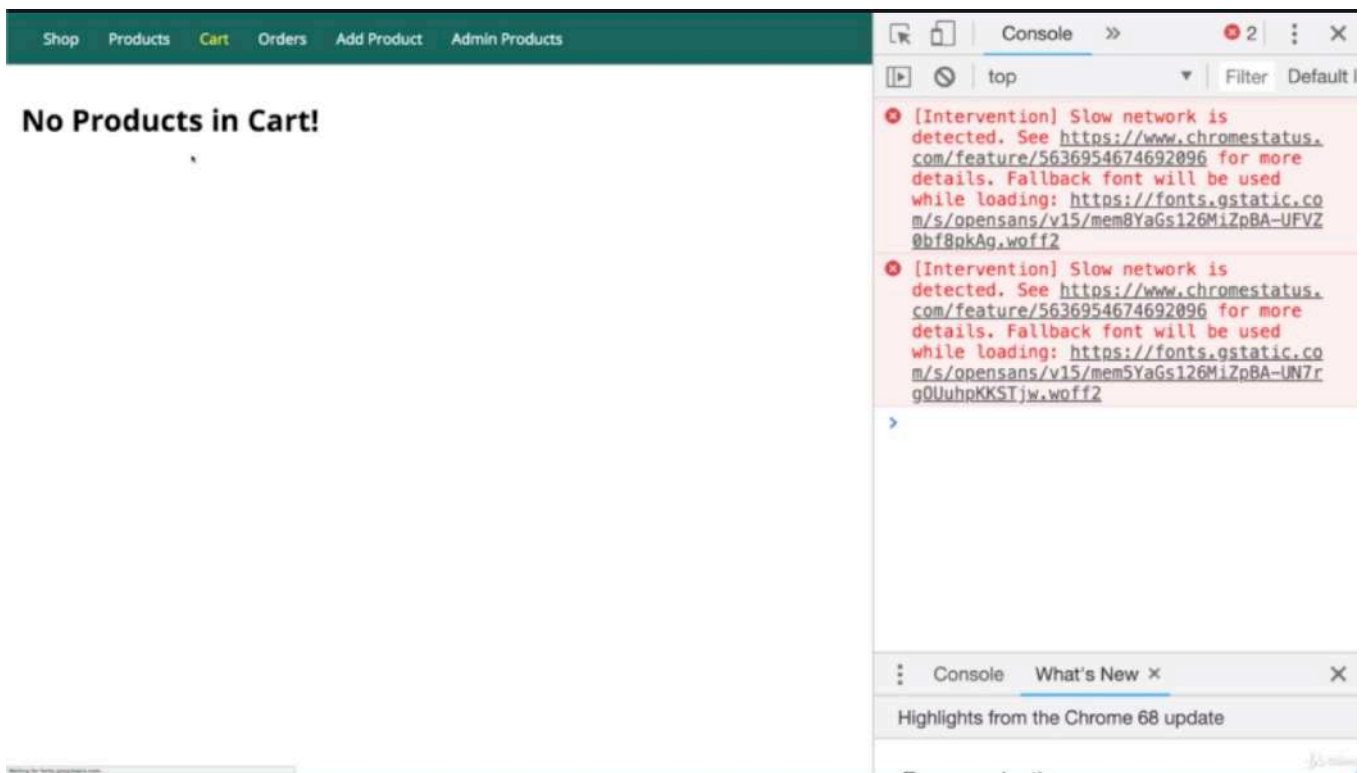
Delete

Elements Console >> Filter Default

top

Console What's New x

Highlights from the Chrome 68 update



ShopProductsCartOrdersAdd ProductAdmin Products

Title
fasdf

Image URL
fasdf

Price
12

Description
fadsf

Add Product

<!doctype html>
<html lang="en">
 <head>...</head>
 <body cz-shortcut-listen="true">
 <div class="backdrop">
 </div>
 <header class="main-header">...</header>
 <nav class="mobile-nav">
 ...</nav>
 <main>
 <form class="product-form" action="/admin/add-product" method="POST"> == \$0
 </main>
 </body>
</html>

... main form.product-form

StylesComputed>>

Filter: :hov .cls +


.product-form { product.css:1

What's New x

Highlights from the Chrome 6...

ShopProductsCartOrdersAdd ProductAdmin Products


A Book



\$19
This is an awesome book!

Add to Cart

fasdf



\$12
fasdf

Add to Cart

<!doctype html>
<html lang="en">
 <head>...</head>
 <body cz-shortcut-listen="true"> == \$0
 <div class="backdrop">
 </div>
 <header class="main-header">...</header>
 <nav class="mobile-nav">
 ...</nav>
 <main>...</main>
 <script src="/js/main.js"></script>
 </body>
</html>

html body

StylesComputed>>

Filter: :hov .cls +


body { main.css:7

What's New x

Highlights from the Chrome 6...

Shop
Products
Cart
Orders
Add Product
Admin Products


A Book



\$ 19
This is an awesome book!

Edit Delete

fasdf



\$ 12
fasdf

Edit Delete

<!doctype html>
<html lang="en">
<head>...</head>
<body cz-shortcut-listen="true"> == \$0
<div class="backdrop">
</div>
<header class="main-header">...</header>
<nav class="mobile-nav">...</nav>
<main>...</main>
<script src="/js/main.js"></script>
</body>
</html>

html
body


Styles
Computed
>>

Filter
:hov .cls +

body {
main.css:7
What's New x
Highlights from the Chrome 6...

Shop
Products
Cart
Orders
Add Product
Admin Products

A Book



\$ 19
This is an awesome book!

Edit Delete

<!doctype html>
<html lang="en">
<head>...</head>
<body cz-shortcut-listen="true"> == \$0
<div class="backdrop">
</div>
<header class="main-header">...</header>
<nav class="mobile-nav">...</nav>
<main>...</main>
<script src="/js/main.js"></script>
</body>
</html>

html
body

Styles
Computed
>>

Filter
:hov .cls +

body {
main.css:7
What's New x
Highlights from the Chrome 6...

```

1 <!--./views/shop/cart.ejs-->
2
3 <%= include('../includes/head.ejs') %>
4   </head>
5
6   <body>
7     <%= include('../includes/navigation.ejs') %>
8     <main>
9       <%= if (products.length > 0) { %>
10         <ul>
11           <%= products.forEach(p => { %>
12             <li>
13               <p><%= p.productData.title %> (<%= p.qty %>)

```

```

14 </p>
15 <form action="/cart-delete-item"
method="POST">
16 <input type="hidden" value="<%=
p.productData.id %>" name="productId">
17 <button class="btn"
type="submit">Delete</button>
18 </form>
19 </li>
20 <% }>%>
21 </ul>
22 <% } else { %>
23 <h1>No Products in Cart!</h1>
24 <% } %>
25 </main>
<%- include('../includes/end.ejs') %>

```

```

1 // ./routes/shop.js
2
3 const path = require('path');
4
5 const express = require('express');
6
7 const shopController = require('../controllers/shop');
8
9 const router = express.Router();
10
11 router.get('/', shopController.getIndex);
12
13 router.get('/products', shopController.getProducts);
14
15 router.get('/products/:productId', shopController.getProduct);
16
17 router.get('/cart', shopController.getCart);
18
19 router.post('/cart', shopController.postCart)
20
21 router.post('/cart-delete-item', shopController.postCartDeleteProduct)
22
23 router.get('/orders', shopController.getOrders);
24
25 router.get('/checkout', shopController.getCheckout);
26
27 module.exports = router;
28

```

```

1 //./controllers/shop.js
2
3 const Product = require('../models/product');
4 const Cart = require('../models/cart');
5
6 exports.getProducts = (req, res, next) => {
7   Product.fetchAll(products => {
8     res.render('shop/product-list', {
9       prods: products,
10      pageTitle: 'All Products',
11      path: '/products'

```

```

12     });
13   });
14 };
15
16 exports.getProduct = (req, res, next) => {
17   const prodId = req.params.productId;
18   Product.findById(prodId, product => {
19     res.render('shop/product-detail', {
20       product: product,
21       pageTitle: product.title,
22       path: '/products'
23     });
24   });
25 };
26
27 exports.getIndex = (req, res, next) => {
28   Product.fetchAll(products => {
29     res.render('shop/index', {
30       prods: products,
31       pageTitle: 'Shop',
32       path: '/'
33     });
34   });
35 };
36
37 exports.getCart = (req, res, next) => {
38   Cart.getCart(cart => {
39     Product.fetchAll(products => {
40       const cartProducts = [];
41       for (product of products) {
42         const cartProductData = cart.products.find(
43           prod => prod.id === product.id
44         );
45         if (cartProductData) {
46           cartProducts.push({ productData: product, qty: cartProductData.qty });
47         }
48       }
49       res.render('shop/cart', {
50         path: '/cart',
51         pageTitle: 'Your Cart',
52         products: cartProducts
53       });
54     });
55   });
56 };
57
58 exports.postCart = (req, res, next) => {
59   const prodId = req.body.productId;
60   Product.findById(prodId, product => {
61     Cart.addProduct(prodId, product.price);
62   });
63   res.redirect('/cart');
64 };
65
66 exports.postCartDeleteProduct = (req, res, next) => {
67   const prodId = req.body.productId;

```

```

68 Product.findById(prodId, product => {
69   Cart.deleteProduct(prodId, product.price);
70   res.redirect('/cart');
71 });
72 };
73
74 exports.getOrders = (req, res, next) => {
75   res.render('shop/orders', {
76     path: '/orders',
77     pageTitle: 'Your Orders'
78   });
79 };
80
81 exports.getCheckout = (req, res, next) => {
82   res.render('shop/checkout', {
83     path: '/checkout',
84     pageTitle: 'Checkout'
85   });
86 };
87

```

```

1  //./models/cart.js
2
3  const fs = require('fs');
4  const path = require('path');
5
6  const p = path.join(
7    path.dirname(process.mainModule.filename),
8    'data',
9    'cart.json'
10 );
11
12 module.exports = class Cart {
13   static addProduct(id, productPrice) {
14     // Fetch the previous cart
15     fs.readFile(p, (err, fileContent) => {
16       let cart = { products: [], totalPrice: 0 };
17       if (!err) {
18         cart = JSON.parse(fileContent);
19       }
20       // Analyze the cart => Find existing product
21       const existingProductIndex = cart.products.findIndex(
22         prod => prod.id === id
23       );
24       const existingProduct = cart.products[existingProductIndex];
25       let updatedProduct;
26       // Add new product/ increase quantity
27       if (existingProduct) {
28         updatedProduct = { ...existingProduct };
29         updatedProduct.qty = updatedProduct.qty + 1;
30         cart.products = [...cart.products];
31         cart.products[existingProductIndex] = updatedProduct;
32       } else {
33         updatedProduct = { id: id, qty: 1 };
34         cart.products = [...cart.products, updatedProduct];
35       }
36       cart.totalPrice = cart.totalPrice + +productPrice;

```

```

37     fs.writeFile(p, JSON.stringify(cart), err => {
38         console.log(err);
39     });
40 });
41 }
42 static deleteProduct(id, productPrice) {
43     fs.readFile(p, (err, fileContent) => {
44         if (err) {
45             return;
46         }
47         const updatedCart = { ...JSON.parse(fileContent) };
48         const product = updatedCart.products.find(prod => prod.id === id);
49         if (!product) {
50             return;
51         }
52         const productQty = product.qty;
53         updatedCart.products = updatedCart.products.filter(
54             prod => prod.id !== id
55         );
56         updatedCart.totalPrice =
57             updatedCart.totalPrice - productPrice * productQty;
58
59         fs.writeFile(p, JSON.stringify(updatedCart), err => {
60             console.log(err);
61         });
62     });
63 }
64
65 static getCart(cb) {
66     fs.readFile(p, (err, fileContent) => {
67         const cart = JSON.parse(fileContent);
68         if (err) {
69             cb(null);
70         } else {
71             cb(cart);
72         }
73     });
74 }
75 static getCart(cb){
76     fs.readFile(p, (err, fileContent) => {
77         const cart = JSON.parse(fileContent)
78         if(err){
79             cb(null)
80         } else {
81             cb(cart)
82         }
83     })
84 }
85 };

```

* Chapter 129: Wrap Up

Module Summary

Dynamic Routing

- You can pass dynamic path segments by adding a ":" to the Express router path
- The name you add after ":" is the name by which you can extract the data on `req.params`
- Optional (query) parameters can also be passed (`?param=value&b=2`) and extracted (`req.query.myParam`)

More on Models

- A Cart model was added – it holds static methods only
- You can interact between models (e.g. delete cart item if a product is deleted)
- Working with files for data storage is suboptimal for bigger amounts of data