Add to cart functionality

Step 1: Update the Products Listing View

1. **Open the Products Listing File:**

Open the file views/products/index.ejs in your code editor.

2. **Locate the Card Footer Section:**

Find the part of the code where the product details (such as the “View Details” button) are rendered. It will look similar to this:

```
<div class="card-footer">
  <small class="text-muted">Price: $<%= product.price.toFixed(2) %>
</small>
  <a href="/products/<%= product._id %>" class="btn btn-primary btn-sm
float-end">View Details</a>
</div>
```

3. **Insert the “Add to Cart” Form:**

Modify the card footer to include a form that submits a POST request to the /cart/add endpoint. For example, update the footer section as follows:

```
<div class="card-footer">
  <small class="text-muted">Price: $<%= product.price.toFixed(2) %>
</small>
  <!-- Add to Cart Form -->
  <form action="/cart/add" method="POST" class="d-inline me-2">
    <input type="hidden" name="productId" value="<%= product._id %>">
    <!-- Default quantity is 1; you can also provide a field if you wish -
->
    <input type="hidden" name="quantity" value="1">
    <button type="submit" class="btn btn-success btn-sm">Add to
Cart</button>
  </form>
  <a href="/products/<%= product._id %>" class="btn btn-primary btn-sm
float-end">View Details</a>
</div>
```

Explanation:

• The form uses the POST method to submit data to /cart/add.

• Hidden input fields pass the product ID and a default quantity (set to 1).

• The btn btn-success btn-sm classes from Bootstrap style the “Add to Cart” button.

4. **Save Your Changes:**

Save the updated index.ejs file.

Step 2: Verify the Cart Endpoint Receives Data

1. **Confirm Cart Route Functionality:**

Ensure your cart route (routes/cart.js) is correctly set up to handle POST requests at /cart/add:

```
// POST /cart/add – Add a product to the cart
router.post('/add', async (req, res) => {
  initCart(req);
  const { productId, quantity } = req.body;
  try {
    const product = await Product.findById(productId);
    if (!product) {
      return res.status(404).send('Product not found');
    }
    const existingItem = req.session.cart.find(item =>
item.product._id.toString() === productId);
    if (existingItem) {
      existingItem.quantity += Number(quantity) || 1;
    } else {
      req.session.cart.push({ product, quantity: Number(quantity) || 1 });
    }
    res.redirect('/cart');
  } catch (err) {
    console.error(err);
    res.status(500).send('Server error');
  }
});
```

This route finds the product by its ID, then either updates the quantity of an existing item in the cart or adds a new item.

2. **Test the Functionality:**

• Start or restart your server:

```
node app.js
```

• Navigate to http://localhost:3000/products in your browser.

• Click on the “Add to Cart” button for a product.

• After submitting, you should be redirected to http://localhost:3000/cart and see the product listed in the cart.

Step 3: Troubleshoot (if needed)

• **If the Cart Remains Empty:**

• Ensure that the form submission is triggering a POST request by checking your browser’s developer tools (Network tab).

• Confirm that the product ID is being correctly set in the hidden input field.

• Make sure your cart route in routes/cart.js is mounted properly in app.js:

```
app.use('/cart', cartRouter);
```

• **Optional:**

If you’d like users to choose a quantity before adding a product, replace the hidden quantity input with a number input:

```
<input type="number" name="quantity" value="1" min="1" class="form-control
d-inline" style="width: 70px;">
```

]