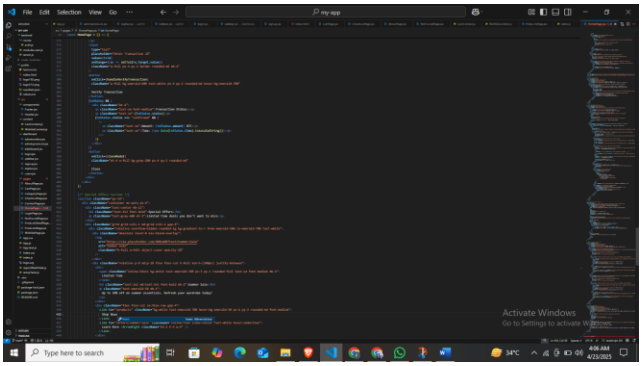
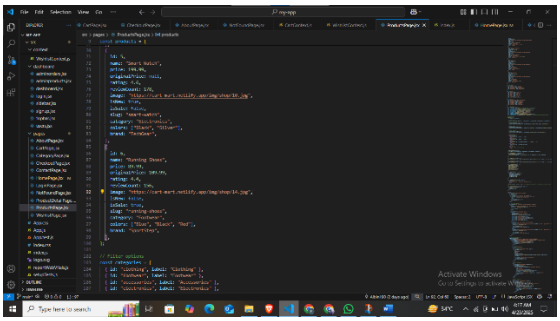
Homepage Project:



- people can go to those pages. There’s also a search bar and some icons for their account, wish list, and cart.
- **Special Offer Banner**: I added a banner that says you get free shipping if you spend over $50, and a code (WELCOME10) for 10% off your first order.
- **Main Section (Hero)**: This part says "Summer Collection 2025" with a little line about finding cool summer styles. It has buttons to shop or check out collections, plus some customer pictures and a summer sale tag with a code (SUMMER30).
- **Categories Part**: I made a grid showing different categories like Men's Fashion and Women's Fashion with pictures, names, and how many products are in each one. You can click to see more.
- **Featured Products Part**: This shows some cool products I picked, like t-shirts and jeans, with their price, rating, and reviews. You can add them to your cart or wish list, and there’s a button to check Bitcoin payments.
- **Bitcoin Check Pop-Up**: I added a pop-up where people can type a Bitcoin Transaction ID to see if their payment worked, using something called Blockchain API.
- **Special Offers Part**: This part has deals like a Summer Sale (up to 50% off) and New Arrivals with buttons to shop or learn more.
- **Newsletter Part**: I made a spot where people can type their email to get updates about new stuff and deals.

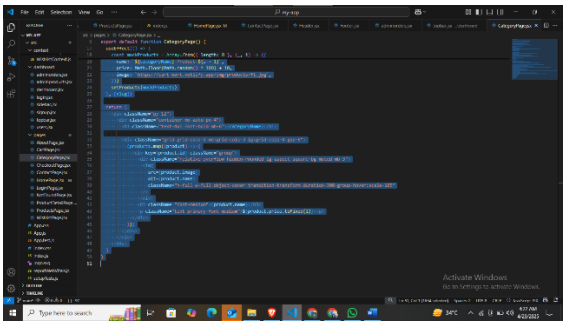
Products Page Project:



What I Added to the Page

- **Title and Breadcrumbs**: At the top, I put "All Products" as the title and a little path (Home / Products) so people know where they are.
- **Filter Section**: I made a sidebar (or a pop-up on mobile) where people can filter products by:
    - Price (they can slide to pick a price range between $0 and $200).
    - Categories like Clothing, Footwear, Accessories, and Electronics.
    - Colors like Black, White, Blue, and more.
    - Brands like Fashion Brand, DenimCo, and others.
- **Sorting Options**: People can sort products by Featured, Price (low to high or high to low), Newest, or Top Rated.
- **View Options**: They can see products in a grid (like cards) or a list (like rows). I added buttons to switch between these views.
- **Product Cards/Rows**: Each product shows its picture, name, price, rating, and reviews. If it's new or on sale, I added little tags. People can pick a color for the product and add it to their cart or wishlist.
- **Active Filters**: If someone picks filters, I show them at the top of the sidebar with a "Clear All" button to reset everything.
- **No Products Message**: If no products match the filters, I show a message saying "No products found" with a button to clear filters.
- **Pagination**: At the bottom, I added buttons to go to different pages of products (like page 1, 2, 3).

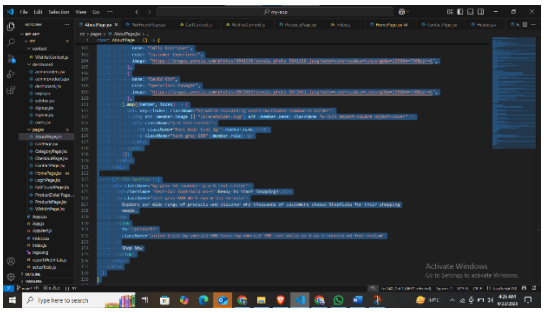
Category Page Project:



What I Added to the Page

- **Category Title**: I made the page show the category name at the top in big, bold text. For example, if the link is "mens-fashion," it shows as "Mens Fashion."
- **Product Grid**: I added a grid to show products in the category. Right now, I'm using fake products (8 of them), and each one has:
    - A picture (I used the same image for all of them for now).
    - A name like "Mens Fashion Product 1" (it changes based on the category).
    - A random price between $10 and $110.
- **Hover Effect**: When you hover over a product picture, it zooms in a little to look cool.

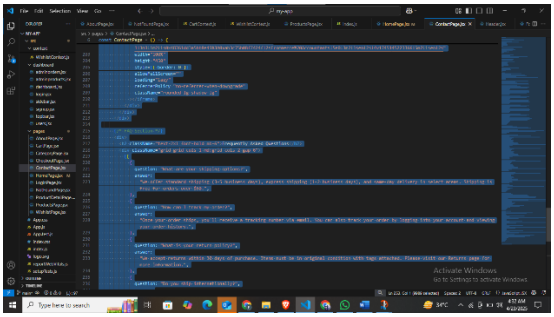
About Page Project:



What I Added to the Page

- Header Section: I put a big title "About ShopVista" at the top with a short line saying we want to give good products at nice prices and make shopping awesome.
- Our Story Section: I wrote a little story about how ShopVista started in 2020 with the idea to focus on customers. It says we grew from a small store to a big one with lots of products, and we care about quality, low prices, and good service.
- Values Section: I made a grid with four boxes to show what we care about:
    - Customer First: We always think about our customers and try to make them happy.
    - Quality: We pick products that are really good and last a long time.
    - Efficiency: We work fast so customers don't have to wait for their orders or help.
    - Community: We like to help the places where our customers and team live.
      Each box has a little icon and some text to explain.
- Team Section: I added a grid with four team members (with fake names and pictures for now). Each one has their name, role (like Founder & CEO), and a picture.
- Shop Now Section: At the bottom, I made a section that asks if they're ready to shop. It has a button that says "Shop Now" and takes them to the products page.

Contact Page Project:

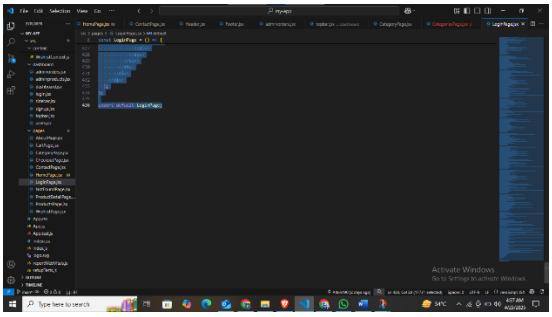


What I Added to the Page

- **Header Section**: I put a big title "Contact Us" at the top with a line that says we'd love to hear from them and our team is here to help.
- **Contact Info Section**: On one side, I added a box with our info:
    - **Location**: I wrote a fake address (123 Commerce Street, Shopville, SV 12345, United States) with a map icon.
    - **Phone**: I added a fake phone number (+1 (555) 123-4567) and hours (Mon-Fri, 9am-6pm EST) with a phone icon.
    - **Email**: I put two fake emails (support@shopvista.com and sales@shopvista.com) with a mail icon.
    - **Hours**: I listed our business hours (Mon-Fri: 9am-6pm, Sat: 10am-4pm, Sun: Closed) with a clock icon.
- **Contact Form**: On the other side, I made a form where people can send us a message. They need to fill in:
    - Their name
    - Their email
    - A subject
    - Their message
      When they click "Send Message," it shows a green box saying "Thank you for your message! We'll get back to you soon." (Right now, it just logs the message in the console since there's no real backend.)

- **Map Section**: I added a section with a Google Map showing a fake location (I used an iframe to embed the map).
- **FAQ Section**: I made a grid with four common questions and answers:
    - Shipping options (standard, express, and same-day, free over $50).
    - How to track an order (with a tracking number or in their account).
    - Return policy (within 30 days, items must be in original condition).
    - International shipping (yes, to select countries, check the Shipping page).

Login Page Project:



What I Added to the Page

- **Tabs for Login and Register**: At the top, I made two tabs: "Login" and "Register." People can click to switch between them.
- **Login Form**: In the Login tab, people can:
    - Type their email and password.
    - Click a button to show or hide their password (with an eye icon).
    - Check a box to "Remember me."
    - Click "Forgot password?" to go to a reset page (not built yet).
    - Click "Sign In" to log in.
- **Register Form**: In the Register tab, people can:
    - Type their full name, email, password, and confirm their password.
    - Click buttons to show or hide their password and confirm password (with eye icons).
    - Check a box to agree to the Terms of Service and Privacy Policy (with links to those pages).
    - Click "Create Account" to sign up.
- **Social Login Buttons**: I added buttons for signing in with Facebook, Google, and GitHub (they don’t work yet, just for looks).
- **Loading and Alerts**: When someone logs in or signs up, I show a "Loading" message on the button. I used a tool called SweetAlert2 to show a success message if it works, or an

error message if something goes wrong (like if passwords don't match or the backend isn't running).

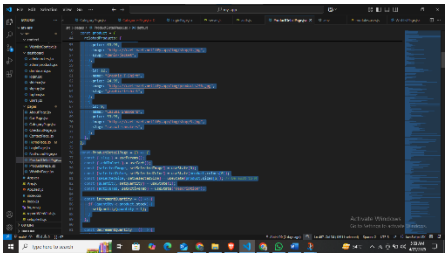
- **Backend Connection**: I made the form talk to my backend (Node.js and MongoDB) to check the login details or save new users. If login works, it saves a token and user info in the browser and takes them to the Home page. If signup works, it switches to the Login tab so they can log in.

What I Did in the Backend

I made an authentication system using Node.js and MongoDB. It has two parts:

- **Login**: It checks if the email and password are correct in the MongoDB database and sends back a token if they match.
- **Signup**: It saves the new user's name, email, and password in MongoDB and creates a token for them.

Product Detail Page:



Here’s what the page includes and does:

1. Product Information:

- Shows the product name, price, and any discounts (like a sale price).

- Displays a rating (stars) and the number of reviews.

- Has a "New" or "Sale" badge if the product is new or on sale.

2. Product Images:

- Shows a big main image of the product.

- Has smaller thumbnail images below that users can click to change the main image.

3. Customization Options:

- Lets users choose the product color from a list (like Black, White, etc.).

- Allows picking a size (like XS, S, M, etc.).

- Has a quantity selector to choose how many items to buy (with plus and minus buttons).

4. Add to Cart & Wishlist:

- Users can click a button to add the product to their cart with the selected color, size, and quantity.

- There’s also a button to add the product to a wishlist.

5. Extra Details:

- Shows benefits like free shipping, free returns, and secure payments.

- Has a "Share" button for sharing the product.

6. Tabs for More Info:

- Description Tab: Explains what the product is and lists its key features (like "100% organic cotton").

- Details Tab: Shows specific info like material, fit, care instructions, and where it's made.

- Reviews Tab: Displays customer reviews (or a message if there are none) and a button to write a review.

7. Related Products:

- At the bottom, there's a section showing other products users might like, with images, names, prices, and an "Add to Cart" button.

8. Navigation:

- Has a breadcrumb trail (like Home > Products > Product Name) so users know where they are on the site.

- Links to a size guide and a "View All" button for more products.

Why I made it this way?

I wanted the page to be:

- User-friendly: Easy to navigate with clear buttons and options.

- Informative: Gives all the important details about the product.

- Interactive: Lets users customize their choices and see related items.

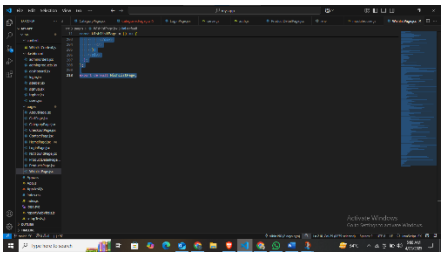
- **Attractive**: Looks clean and modern with images and badges.

This page is part of a bigger e-commerce website, and I designed it to help customers shop easily while making the website look professional.

How to use it?

- Open the page to see the product details.
- Click on thumbnail images to change the main image.
- Choose a color and size, and adjust the quantity.
- Click "Add to Cart" to add the product or "Add to Wishlist" to save it.
- Use the tabs to read more about the product or check reviews.
- Scroll down to see related products and add them to the cart if you want.

Wishlist Page:



Here's what the page includes and does:

1. Empty Wishlist View:

- If the wishlist is empty, it shows a message saying, "Your wishlist is empty."
- Displays a heart icon and a suggestion to start shopping.
- Has a "Start Shopping" button that takes users to the products page.

2. Wishlist Items Display:

- If there are items in the wishlist, it shows a list of them in a clean, organized layout.
- Each item includes:
  - A checkbox to select the item.
  - A small image of the product.
  - The product name (clickable to go to its detail page).
  - Details like color, size, and price (shows original price with a strikethrough if there's a discount).
  - A stock status (like "In Stock" or "Out of Stock").
  - An "Add to Cart" button (disabled if the item is out of stock).
  - A remove button (an "X") to delete the item from the wishlist.

3. Selection Features:

- Users can select items using checkboxes next to each product.

- A “Select All” checkbox at the top selects or deselects all items.

- The page shows how many items are in the wishlist (e.g., “Select All (3 items)”).

4. Manage Wishlist:

- Remove Selected: If users select some items, they can click a “Remove Selected” button to delete them all at once.

- **Clear Wishlist**: A “Clear Wishlist” button removes everything from the wishlist.

- When items are removed or the wishlist is cleared, a small notification pops up to confirm the action.

5. Add to Cart:

- Users can click “Add to Cart” for any in-stock item, and it gets added to their cart with the chosen color and size.

- A notification confirms the item was added to the cart.

6. Navigation Buttons:

- A “Continue Shopping” button takes users back to the products page.

- A “View Cart” button takes users to their cart to check out.

How to use it?

- Open the page to see your wishlist.

- If it’s empty, click “Start Shopping” to browse products.

- If there are items:

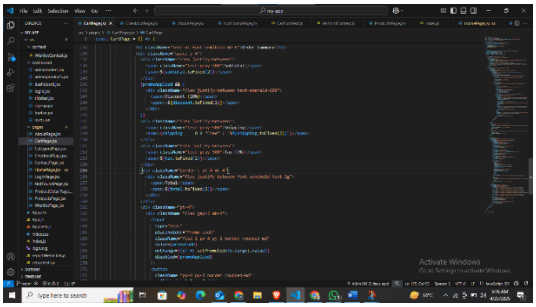
- Check the boxes to select items you want to manage.

- Click “Add to Cart” to add an item to your cart (if it’s in stock).

- Click the “X” to remove a single item or use “Remove Selected” for multiple items.

- Click “Clear Wishlist” to delete everything.

Shopping Cart Page:



Features

- Cart Items List: Shows each item with its image, name, color, size, price, and quantity. You can:
    - Increase or decrease the quantity using plus/minus buttons.
    - Remove items with a delete button.
    - Click the item name to go to its product page.
- Order Summary: Displays:
    - Subtotal of all items.
    - Discount (10% if the promo code "DISCOUNT10" is applied).
    - Shipping cost (free for orders over $100, otherwise $4.99).
    - Tax (7% of the subtotal after discount).
    - Total cost.
- Promo Code: Users can enter a promo code ("DISCOUNT10") to get a 10% discount.
- Checkout Button: Takes users to the checkout page.
- Empty Cart Message: If the cart is empty, it shows a friendly message with a "Continue Shopping" button.
- Payment Info: Shows accepted payment methods (Visa, Mastercard, Amex, PayPal) and mentions free shipping on orders over $100 and secure payments.

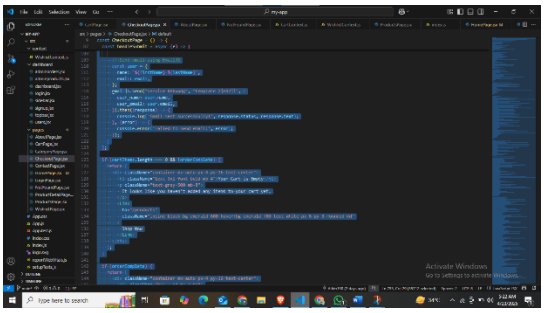
How it Works

- The page uses React and the CartContext to manage the cart items (like adding, updating, or removing items).
- It uses React Router for navigation (e.g., linking to product or checkout pages).
- The Lucide React library provides icons for buttons and visuals.
- The layout is responsive, meaning it looks good on both mobile and desktop screens.
- The design uses Tailwind CSS for styling, making it clean and modern.

How to Use It

1. Add items to the cart (this page assumes items are already added via the CartContext).
2. View your items, update quantities, or remove them.
3. Enter the promo code "DISCOUNT10" to get a discount.
4. Check the order summary for the total cost.
5. Click "Proceed to Checkout" to continue.

Checkout Page:



Features

- Multi-Step Checkout Process: The checkout is divided into three steps:

1. Shipping: Collects user details like name, email, address, and phone number.

2. Payment: Offers multiple payment options, including credit/debit card, PayPal, Apple Pay, and cryptocurrency (Ethereum via MetaMask).

3. Review: Shows a summary of shipping info, payment method, and cart items for final confirmation.

- Order Summary: Displays the subtotal, shipping cost (free over $100), tax (7%), and total.

- Cryptocurrency Payment: Users can pay with Ethereum using MetaMask. The page connects to the user's wallet and processes payments securely.

- Email Confirmation: After placing the order, an email is sent to the user using EmailJS.

- Order Confirmation Page: Once the order is complete, users see a confirmation page with order details, including an order number, date, total, and payment method. For crypto payments, it shows the transaction hash with a link to Etherscan.

- Empty Cart Handling: If the cart is empty, users are shown a message with a "Shop Now" button.

- Responsive Design: Works well on both mobile and desktop screens.

- Security Features: Includes icons and text to reassure users about secure checkout, free shipping over $100, and multiple payment options.

How to Use It

1. Add items to your cart (assumed to be done via CartContext).

2. Go to the checkout page from the cart.

3. Shipping Step: Enter your email, name, address, city, state, ZIP code, country, and phone number. Optionally save the info for next time.

4. Payment Step: Choose a payment method:

- For credit card, enter card details (name, number, expiry, CVC).
- For PayPal or Apple Pay, select the option (assumes integration).
- For crypto, connect MetaMask, then pay the total in ETH.

5. Review Step: Check your shipping info, payment method, and cart items. Edit if needed.

6. Place the order to complete the purchase.

7. View the confirmation page with order details and receive an email confirmation.

Notes

- Replace `"YOUR_ETHEREUM_WALLET_ADDRESS"` with your actual Ethereum wallet address for crypto payments.

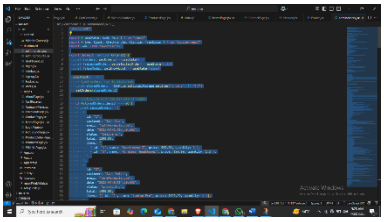
- Replace `"YOUR_EMAILJS_PUBLIC_KEY"` with your EmailJS public key, and ensure the service (`service_88hvpdg`) and template (`template_2je42ji`) IDs match your EmailJS setup.

- The crypto payment assumes an ETH price of $2000 for conversion. Adjust the rate as needed.

- The credit card form is a placeholder; you'd need a payment gateway (e.g., Stripe) for real transactions.

- PayPal and Apple Pay options are included but require additional setup for full functionality.

Orders Page:



Features

- Order List: Shows a table of orders with columns for:
    - Order ID
    - Customer name and email
    - Order date and time
    - Status (with color-coded badges)
    - Total amount
    - Action button to view details
- Order Details Modal: Clicking the "View" button opens a modal with:
    - Customer info (name, email)
    - Order date and status
    - List of items (product name, price, quantity, subtotal)
    - Total order amount
    - Buttons to update the order status
- Status Updates: Users can change an order’s status to:
    - Processing
    - Shipped
    - Delivered
    - Cancelled
    - Updates are saved to localStorage and shown with a success popup using SweetAlert2.

- Status Badges: Each status has a unique color for easy recognition:
    - Pending: Yellow
    - Processing: Blue
    - Shipped: Purple
    - Delivered: Green
    - Cancelled: Red
- Sample Data: If no orders exist, the page loads sample orders to demonstrate functionality.
- Responsive Design: The table and modal look good on both mobile and desktop screens.
- SweetAlert2 Notifications: Shows a popup when the order status is updated.

How it Works

- The page uses React with useState and useEffect to manage orders and load data from localStorage.
- localStorage stores the orders, so data persists between page refreshes.
- If localStorage is empty, it loads sample orders with predefined data (e.g., customers, items, statuses).
- SweetAlert2 provides a clean popup for status update confirmations.
- Lucide React icons enhance the visuals (e.g., eye for view, clock for processing, truck for shipped).
- The design uses Tailwind CSS for a modern, clean look.
- The modal is scrollable and responsive, ensuring it works well for long item lists.

How to Use It

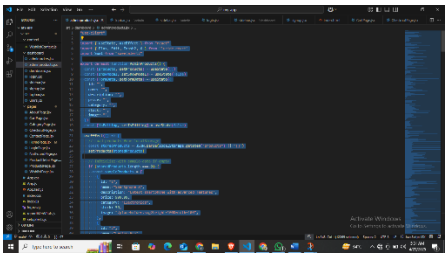
1. Visit the Orders page to see the list of orders.
2. If no orders exist, sample orders (e.g., for John Doe, Jane Smith) are loaded automatically.
3. Click the "View" (eye icon) button next to an order to open the details modal.
4. In the modal, review the customer info, order date, status, and items.

5. Update the order status by clicking one of the status buttons (Processing, Shipped, Delivered, Cancelled).
6. A SweetAlert2 popup confirms the status change.
7. Close the modal by clicking the "Close" button or the X icon.
8. Changes are saved to localStorage, so they persist when you revisit the page.

Notes

- The page uses localStorage for simplicity. In a real application, you'd connect to a backend API to store and fetch orders.
- The sample orders are loaded only if localStorage is empty. You can modify the sample data in the code to suit your needs.
- The status badges and colors are customizable by editing the getStatusBadge function.
- The SweetAlert2 popup auto-closes after 1.5 seconds, but you can adjust the timer or add a confirmation button if needed.

Admin Products Page:



Features

- Product List: Displays a table of products with columns for:
    - Product (name, description preview, and image)
    - Category
    - Price
    - Stock
    - Actions (edit or delete)
- Add/Edit Product Modal: A form in a modal to add new products or edit existing ones, capturing:
    - Product name
    - Description
    - Price
    - Category
    - Stock
    - Image URL (defaults to a placeholder if not provided)
- Delete Product: Admins can delete products with a confirmation prompt to prevent accidental deletions.
- SweetAlert2 Notifications: Shows popups for successful actions (add, update, delete) and confirmation for deletions.

- Sample Data: If no products exist, the page loads sample products (e.g., Smartphone X, Laptop Pro) to demonstrate functionality.
- LocalStorage Persistence: Product data is saved in localStorage, so it persists between page refreshes.
- Responsive Design: The table and modal work well on both mobile and desktop screens.
- Lucide React Icons: Used for actions (plus for add, edit, trash for delete, X for closing modal).

How it Works

- The page uses React with useState and useEffect to manage product data and load from localStorage.
- localStorage stores products, ensuring data persists across sessions.
- If localStorage is empty, sample products are loaded automatically.
- SweetAlert2 provides clean popups for success messages and delete confirmations.
- The modal form handles both adding and editing products:
    - Adding creates a new product with a unique ID (based on timestamp).
    - Editing updates the existing product based on its ID.
- Tailwind CSS is used for a modern, clean design.
- The delete action requires confirmation to avoid mistakes, using a SweetAlert2 prompt.

How to Use It

1. Visit the Admin Products page to see the list of products.
2. If no products exist, sample products (e.g., Smartphone X, Laptop Pro) are loaded automatically.
3. To add a product:
    - Click the “Add Product” button to open the modal.
    - Fill in the form (name, description, price, category, stock, and optional image URL).
    - Click “Add Product” to save. A success popup confirms the addition.
4. To edit a product:

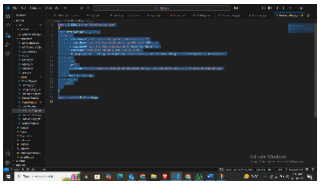
- Click the “Edit” (pencil icon) button next to a product.
- Update the form fields in the modal.
- Click “Update Product” to save changes. A success popup confirms the update.

5. To delete a product:
    - Click the “Delete” (trash icon) button.
    - Confirm the deletion in the SweetAlert2 prompt.
    - A success popup confirms the product was deleted.
6. Close the modal by clicking “Cancel” or the X icon.
7. All changes are saved to localStorage and persist when you revisit the page.

Notes

- The page uses localStorage for simplicity. In a real application, you’d connect to a backend API to manage products.
- The sample products are loaded only if localStorage is empty. You can modify the sample data in the code to fit your needs.
- The image field uses a placeholder URL by default. In a real app, you’d integrate a file upload system or image hosting service.
- The SweetAlert2 popups auto-close after 1.5 seconds for success messages, but the delete confirmation requires user input.
- The form validates required fields (name, description, price, category, stock) to ensure complete data.

Not Found Page:



What is this?

This is a 404 Not Found Page I built for a website using React. It appears when a user tries to access a page that doesn't exist or is unavailable. The page has a simple, user-friendly design with a clear message and a button to return to the homepage.