

Software Design – Lab 3

Name: Clay Carpenter

GitHub Link: [ClayCarp/SoftwareDesign-Lab3](https://github.com/ClayCarp/SoftwareDesign-Lab3)

Purpose Statement:

The lab aimed to explore PostgreSQL, Express, React, and Node.JS. The primary objective was to gain practical experience with specifically PostgreSQL and apply it to a web stack application. To achieve this, a basic web application was created utilizing version 17 of PostgreSQL (latest version). The lab focused on integrating a relational database (PostgreSQL) and applying it to a web application that featured a shopping cart experience.

Experiment Design:

To accomplish creating a web application that uses a shopping cart for the user I would of course need a database to keep track of products. The products had variables. For example, product number, price, product description, and quantity. The application was designed with a very simple directory only using what was needed to gain a simple understanding of PostgreSQL. From a visible standpoint the user will be able to input different things and get a total pricing of their cart. They will be able to select the product and will be able to select how much of the product that they want. They are able to add multiple products to the check out. The overall purpose was to use a very simple P.E.R.N stack to manipulate different features of PostgreSQL.

Resources Utilized:

- YouTube Tutorials
- PostgreSQL v17
- PostgreSQL Documentation

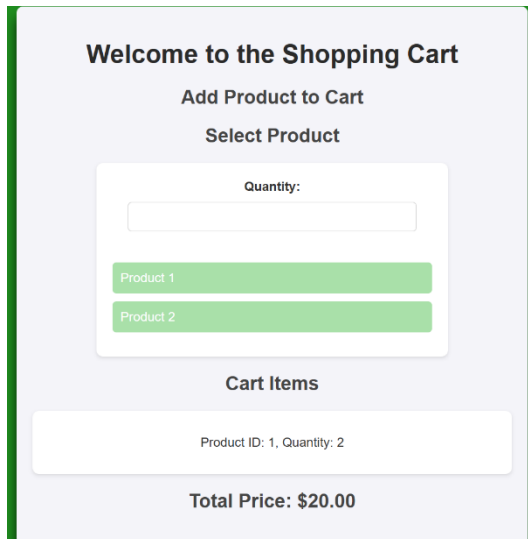
Time Estimate:

- 5 Hours

Experiment:

A simple web that the user can select a product and the quantity for a check out system. PostgreSQL kept track of the database elements used in the web application. For example, product table, user table, cart table, and cart_items table were declared to help keep track of things behind the scenes. The app incorporated a user interface with a simple container that held two product button options and a text field input for the user to enter the quantity.

The submit button was removed for simplicity. When the user selects the product the information will be displayed in the cart container. The image shows the simplicity of the web application. For instance, product one and product two are hardcoded for now just for



The screenshot shows a web application interface for a shopping cart. At the top, it says 'Welcome to the Shopping Cart'. Below that is a section titled 'Add Product to Cart' with a sub-header 'Select Product'. There is a 'Quantity:' label next to an input field. Below the input field are two green buttons labeled 'Product 1' and 'Product 2'. Underneath these buttons is a section titled 'Cart Items' which displays 'Product ID: 1, Quantity: 2'. At the bottom of the interface, it shows 'Total Price: \$20.00'.

experimentation. Product one costs \$10 and product two costs \$20 which is applied whenever the cart gets bigger it keeps track of the sum. The backend was more or less just connecting the relational database to the front end. More time was spent on constructing how the cart would manage updates.

Results:

This experiment provided a simple understanding of using a relational database in terms of a website application. Ultimately demonstrating a simple concept of adding items to a shopping cart. A lot of my time was spent trying to connect to the database and get it to integrate with the front end. After multiple errors I was able to get the app to run on the web in no time. After viewing my work in the allocated time of five hours I can vision what features I could add on to it.

Future Considerations:

1. Better user interface
2. Actual product items instead of just hardcoded product numbers
 - a. This could be changed with adding buttons that display images of the product itself
3. Could also add more than just one page to help the user with navigation.
4. Check out system for the user.

Instructions For Setup:

1. ? git clone <https://github.com/ClayCarp/SoftwareDesign-Lab3.git>
2. ? Install necessary dependencies in the terminal
3. Set up postgresql v17 and created the required tables
 - a. You can just copy and paste the tables I have in the SQL folder
4. In the terminal submit "npm start" this should run both the frontend and the backend concurrently.

