

## COMP2004 – Project 1 – Week 7

Due date: Tuesday, Oct. 29<sup>th</sup>. At 11:59 p.m.

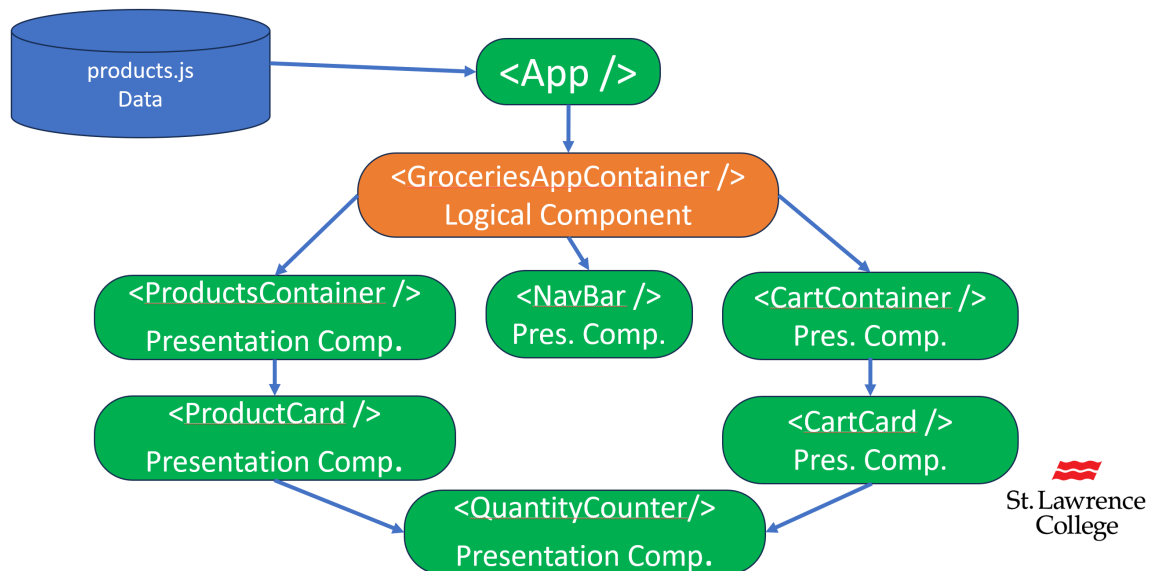
Grade: 15%

### Project Overview:

This project entails building a simple yet functional grocery shopping app, utilizing most of the core programming concepts covered. As part of the task, students will create an interactive app where users can select grocery items from an inventory, view their cart contents, and check out.

A summary of how the app flow works is as follows:

## Groceries App Gameplan



The app consists of the following components.

1. **GroceriesAppContainer** is the primary logical component. This component will handle all the states and functions needed to get this app working. This component will import the data from products.js and pass it to the inventory component.
2. **NavBar** is the top presentation container that displays the app's title, a username greeter, and a cart icon that indicates whether the cart is full, shown by the red dot over the cart icon or empty.
3. **ProductsContainer** is a presentation component that renders all the product cards from the data passed from the parent component.
4. **ProductCard** is a presentation component that displays a single card with all the required elements. It also contains a **QuantityCounter** component if the user wants to add several of the same products to the cart. The component has an "Add to Cart" button; when the user clicks on it, the product and its chosen quantities should be added to the cart. The **CartCard** should be updated with the added amount if the product exists. For example, if the Coca-Cola product has two quantities in the cart and you added another Coca-Cola product with three quantities, the cart should update to five on

the Coca-Cola **CartCard**. If the user tries to add a product with zero items, an alert should be displayed, asking the user to add a quantity before adding it to the cart.

5. **CartContainer** is a presentation component. This component will render out the list of items chosen by the user. If the cart is empty, the component will show “No items in the cart.” Otherwise, it will show the number of items in the cart, and each item will be represented as a **CartCard** component. There are also two buttons at the end of the list:
  - a. Empty the cart button. When clicked, the cart empties.
  - b. Buy button that shows the total price that has no functionality.
6. **CartCard** is a presentational component that displays an image of the product, the price, the quantity, and the sub-total of the products. It also has the **QuantityCounter** component, which allows users to edit the number of the same product they put in the cart, and a remove item button that will remove the item from the cart.
7. **Quantity Counter** is a presentational component that displays the number of items for each product. It has a button to increase the amount and another to decrease the number of products.  
**Note:** Products cannot be less than zero in the **ProductCard** and cannot be less than 1 in the **CardCard**. Please reuse the same component for both **ProductCard** and **CartCard** components.

**Please fork the repo linked on the assignment page, then clone the forked repo to your machine. The repo you will clone will contain the starter code, so there is no need to make a new Vite project. It will also contain the example’s style sheets if you want to use my styling.**

#### **Submission:**

Each student should submit a GitHub link to the project on the submission page on Blackboard.

**Grading rubric:**

Key Concept	Extensive Evidence	Convincing Evidence	Limited Evidence	No Evidence
Functioning Code	The code functions with no errors and passes all test cases. (5%)	The code functions without errors, but not all test cases pass. (3-4%)	The code functions with errors, and some of the tests pass (1 - 2%)	The code is not functional. (0%)
Program Logic and Correctness	The program's output is as expected. (5%)	The program's output produces minor differences than expected. (3-4%)	The program's output produces significant differences than expected. (1 - 2%)	The program does not output. (0%)
Commenting and annotations	The code contains extensive comments and annotations describing the code functions. (3%)	The code contains comments and annotations describing the code functions. (2%)	The code contains some comments and annotations describing the code functions. (1%)	The code is without comments or annotations (0%)
Variable and function naming	The variables and functions' naming are descriptive. (2%)	The variables and functions' naming are somehow descriptive. (1%)		The variables and functions' naming are not descriptive (0%)

**Challenge yourself: (No bonus marks)**

*If you have finished the project and want to challenge yourself with more tasks, you can try the following:*

- 1. Make the cart toggle its appearance when you press on the shopping cart icon. If the cart disappears, the products should fill the app.*
- 2. Refactor the logical handler so that if the user brings down the counter on the **CartCard** to zero, the product will be removed from the cart.*
- 3. For task 2, if you want better usability, warn the user that the product will be removed and ask whether to accept it. If accepted, the product will be removed; otherwise, the product will remain, and the counter will be up to 1.*
- 4. Make the checkout button usable—your choice of presentation of the next steps.*