Lab 05-01 Grade 100/100

horizontal line

# Overview: React Shopping Cart

In this assignment we will be implementing a simple shopping cart together with React.

# Step 1. Create a new repository

As we have done with previousassignments, you need to create a new GitHub repository for this project.

Details are listed below:

* Name the repository **"react-shopping-cart"**
* Make the repository **"private"**
* Add a **README** file
* Add a **.gitignore** file, using the **"Node"** template **(Don't forget this!)**
* Add a **LICENSE** file, using the **"MIT License"**
* Add **evangudmestad** as a **Collaborator/Admin**

# Step 2. Initialize the project

1. Clone the repository to your computer.
2. Open the repository folder in Visual Studio Code.
3. Run the following command in the Terminal window:

**npm create vite@latest**

**Project Name: shopping\_cart**

**Framework: React**

**Variant: JavaScript**

**cd shopping\_cart**

**npm install**

1. Start the project by running the following command in the Terminal window:

**npm run dev**

1. You should now see a new browser window/tab open up, with the spinning React logo in it.

# 

# Step 3. Remove unused example code

Delete the following css files and remove all import statements for them:

* index.css
* app.css

Replace the **App.js** with the following code:

|  |
| --- |
| function App() {  return <div>Hello World!</div>; }  export default App; |

# Step 4. Install Bootstrap

Install bootstrap by running the following command in the terminal:

**npm install bootstrap**

Add the following import statement to **index.js**

|  |
| --- |
| import 'bootstrap/dist/css/bootstrap.min.css'; |

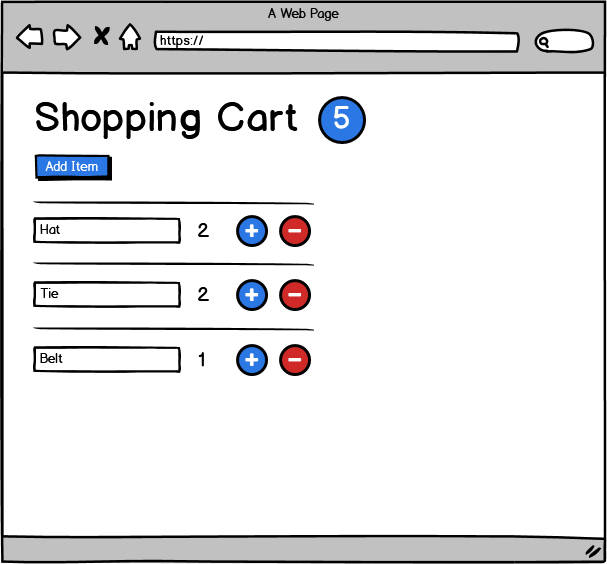
# Step 5. Install other dependencies

Install the following dependencies as well:

* **nanoid**
* **lodash**

# 

# Step 5. Screen mockups

Shown below is a screen mockup for the shopping cart that we are creating.

# 

# Step 6. Requirements

Our shopping cart must implement the following requirements. We will go over how to implement this functionality in class together.

1. Users can edit the name of an item.
   1. *This must be kept in sync with the items array, by handling the onChange event, and hoisting state into the CartList component.*
   2. Give the name field the classes "CartItem-name form-control"
   3. If the name is blank, give the name field the class "is-invalid"
   4. If the name is not blank, give the name field the class "is-valid"
2. Users can increase and decrease the quantity of the item.
   1. *This must be kept in sync with the items array, by handling the onClick event, and hoisting state into the CartList component.*
   2. Give the plus button the classes "CartItem-add btn btn-primary m-1"
   3. Give the minus button the classes "CartItem-remove btn btn-danger m-1"
   4. Do not allow the quantity to go above 10.
   5. Do not allow the quantity to go below zero.
   6. If the quantity is currently 10 or greater, disable the plus button.
   7. If the quantity is currently zero or less, disable the button.
3. Display the total number of items in the shopping cart.
   1. Display this at the top of the page, as shown in the mockup.
   2. Style it with the classes "badge bg-primary rounded-pill fs-6 ms-2"
   3. Hide the count when there are zero items in the cart.
4. Implement a button to add new items to the cart.
   1. When the "Add Item" button is clicked add a new item to the items list, with the following properties: **id = nanoid(), name = "", quantity = 1**
5. Implement a way to remove items from the cart.
   1. When the user decrements the quantity to zero, instead of decreasing the quantity to zero, remove the item from the items list.
6. Show an error message when there are no items in the shopping cart.
   1. When there are no items in the cart, display the following message: "Your cart is empty! Add some items to it."
7. Use CSS to improve the look and feel of the app.