Parent Child Communication Lab

Title: Managing a Shopping Cart

Objective: Implement a simple shopping cart application where users can add products to the cart. Practice parent-child communication to update the cart component with the selected products.

Requirements:

1. Create a parent component called `ProductList` to display a list of products.
2. Create a child component called `Product` to represent each individual product.
3. Each product should have a button to add it to the cart.
4. Create another component called `Cart` to display the list of products added to the cart.
5. Implement communication between the `Product` and `Cart` components using `@Output` and `EventEmitter`.
6. The `Cart` component should update dynamically as products are added to it.

Steps:

1. Set up a new Angular project using the Angular CLI:

```
ng new shopping-cart-app
```

2. Generate the `ProductList`, `Product`, and `Cart` components:

```
ng generate component ProductList

ng generate component Product

ng generate component Cart
```

3. Define a `Product` model to represent each product:

```
// product.model.ts
export interface Product {
  id: number;
  name: string;
  price: number;
}
```

10. Implement the `ProductList` component to display a list of products:

    - Fetch a list of products from a service or hard-code them in the component.
    - Use the `Product` component to display each product.
    - Implement an event handler to receive the `add to cart` events from the `Product` component.

11. Implement the `Product` component:

    - Receive a product as an input and display its details.
    - Emit an event when the `add to cart` button is clicked.

12. Implement the `Cart` component:

    - Receive an array of products as an input and display them in a list.
    - Update the list dynamically when new products are added.

13. Set up parent-child communication between the `Product` and `Cart` components using `@Output` and `EventEmitter`.

    - Emit an event from the `Product` component when the `add to cart` button is clicked.
    - Listen for the event in the `ProductList` component and update the cart accordingly.

14. Test the application:

    - Add products to the cart and verify that they appear in the `Cart` component.
    - Remove products from the cart and verify that the cart updates accordingly.

Extensions:

- Implement functionality to remove products from the cart.
- Add a quantity selector for each product in the cart.
- Persist the cart data using local storage or a backend service.

Note: Don't forget to add styles and improve the UI/UX as needed. This exercise provides a foundation for practicing parent-child communication in Angular while building a practical application.