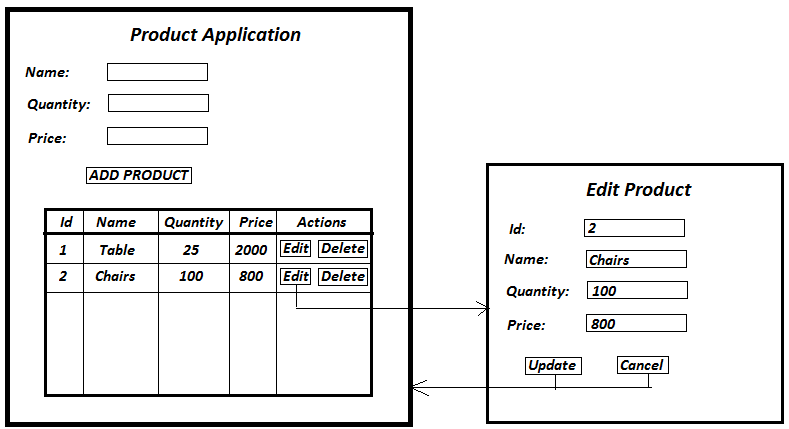
***Building Product Application with React***

We are going to build the following product application step by step:



**STEP 1:**

1. Create a React component called ’ProductForm’ with elements Name, Quantity, Price & ‘ADD PRODUCT’ button.

**STEP 2:**

1. Create a React component called ’ProductTable’ which renders product details table. Add static data inside the table.

**STEP 3:**

1. Write a parent component named ‘ProductApp’ that wraps ProductForm & ProductTable components. Thus, in app.js, you only render ProductApp & ProductApp’s render() method will internally render ProductForm & ProductTable.
2. Initialize the table rows data in ProductApp component using state in JSON format so that ProductTable component will use that data instead of static data. i.e.

this.state = { products : [{id: 1, name: 'Chair', quantity: 20, price: 800},

{id: 2, name: 'Bag', quantity: 110, price: 2400},

{id: 3, name: 'Laptop', quantity: 40000, price: 12}

]}

1. Now let us complete ‘ADD PRODUCT’ action in ProductForm component. When user clicks on ‘ADD PRODUCT’ button, you need to update the table data initialized inside ProductApp component. Make sure that the changed table data will reflect in ProductTable view as well.

**STEP 4:**

1. Define the action of deleting a product inside ProductTable component. This will reflect in making changes into table data of ProductApp component.
2. Create a new component named ‘EditProductForm’.
3. When user clicks on ‘EDIT’ button, you need to render this form with pre-populated values. Note that edit product is NOT a HTML frame.
4. Implement UPDATE & CANCEL actions on EditProductForm.

**STEP 5:**

1. Replace the initialized static table data from ProductApp component into REST call. Now you need to make a REST call in order to perform operations on Product entity. Here are the available REST calls:

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. | Description | URL | METHOD |
| 1. | Getting product list | http://localhost:8080/product | GET |
| 2. | Getting product by id | <http://localhost:8080/product/$id> | GET |
| 3. | Create new product  {name: ‘Chair’, quantity: 20, price: 800} | http:// localhost:8080/product | POST |
| 4. | Update an existing product  {name: ‘Chair’, quantity: 22, price: 950} | <http://localhost:8080/product/$id> | PUT |
| 5. | Delete product | <http://localhost:8080/product/$id> | DELETE |