# Lab 3 [Getting started with ReactJS]

|  |  |
| --- | --- |
| Lab Title: | Getting started with ReactJS |
| **Expected duration** (hours): | **1 hour 45 mins** |

|  |
| --- |
| Objectives |
| Understanding how to create a basic react app |
| Learn how to add components |
| Learn how to create navigation using react-router |

|  |
| --- |
| Requirements |
| A laptop or a desktop with Windows or Mac as an operating system |
| Visual studio code or something similar as a text editor |
| A modern web browser like chrome |

## Lab Instructions:

# Understanding how to create a basic react app

This lab will help you to get familiar with basic layout and setup for a react js app, you will add multiple components to list product, create product and you will also transform the create product page interactive. None of these pages will use anything but HTML, CSS and JavaScript. The purpose of this lab is to getting students familiar with a frontend library like ReactJS and show the power of it.

### Create a react app using Vite

1. Set Up Your Environment
   1. Open your preferred code editor (such as VS Code, Sublime Text, or Notepad++).
   2. First, ensure you have Node.js installed. Then, you can create a new Vite project.
2. Open your terminal and run the below command -
   1. npm create vite@latest my-product-app --template react
3. This will set up a new Vite project with React. Navigate into your project directory:
   1. cd my-product-app
4. Install the necessary dependencies: npm install
5. Create a Product List Component
   1. Create a new file ProductList.jsx in the src directory with the following content:

import React from 'react';

const products = [

{ id: 1, name: 'Product 1', price: '$10' },

{ id: 2, name: 'Product 2', price: '$20' },

{ id: 3, name: 'Product 3', price: '$30' },

];

const ProductList = () => {

return (

<div>

<h1>Product List</h1>

<ul>

{products.map(product => (

<li key={product.id}>

{product.name} - {product.price}

</li>

))}

</ul>

</div>

);

};

export default ProductList;

1. Update App Component
   1. Modify the App.jsx file in the src directory to include the ProductList component:

import React from 'react';

import ProductList from './ProductList';

import './App.css';

const App = () => {

return (

<div className="App">

<ProductList />

</div>

);

};

export default App;

1. Run the Application - Start the development server to see your application in action:

npm run dev

1. Access Your App - Open your browser and go to http://localhost:5173. You should see a simple product list displayed on the page.

### Create New Products

To add a new page for creating new products, you'll need to set up routing in your React app. We can use react-router-dom to handle routing between the product list page and the new product creation page. Here's how you can do it:

1. Install react-router-dom

npm install react-router-dom

1. Create the NewProduct Component
   1. Create a new file NewProduct.jsx in the src directory with the following content:

// src/NewProduct.jsx

import React, { useState } from 'react';

const NewProduct = ({ addProduct }) => {

const [name, setName] = useState('');

const [description, setDescription] = useState('');

const [price, setPrice] = useState('');

const handleSubmit = (e) => {

e.preventDefault();

const newProduct = { id: Date.now(), name, description, price };

addProduct(newProduct);

setName('');

setDescription('');

setPrice('');

};

return (

<div className="container">

<h1>Create New Product</h1>

<form onSubmit={handleSubmit}>

<div>

<label>Product Name:</label>

<input

type="text"

value={name}

onChange={(e) => setName(e.target.value)}

required

/>

</div>

<div>

<label>Description:</label>

<input

type="text"

value={description}

onChange={(e) => setDescription(e.target.value)}

required

/>

</div>

<div>

<label>Price:</label>

<input

type="text"

value={price}

onChange={(e) => setPrice(e.target.value)}

required

/>

</div>

<button type="submit">Add Product</button>

</form>

</div>

);

};

export default NewProduct;

1. Update the App Component
   1. Modify the App.jsx file to include routing:

// src/App.jsx

import React, { useState } from 'react';

import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';

import ProductList from './ProductList';

import NewProduct from './NewProduct';

import productsData from './data';

import './styles.css';

const App = () => {

const [products, setProducts] = useState(productsData);

const addProduct = (product) => {

setProducts([...products, product]);

};

return (

<Router>

<div className="App">

<nav>

<ul>

<li>

<Link to="/">Product List</Link>

</li>

<li>

<Link to="/new">Create New Product</Link>

</li>

</ul>

</nav>

<Routes>

<Route path="/" element={<ProductList products={products} />} />

<Route path="/new" element={<NewProduct addProduct={addProduct} />} />

</Routes>

</div>

</Router>

);

};

export default App;

1. Update the ProductList Component
   1. Modify the ProductList.jsx file to accept products as props:

// src/ProductList.jsx

import React from 'react';

const ProductList = ({ products }) => {

return (

<div className="container">

<h1>Available Products</h1>

{products.map(product => (

<div className="product" key={product.id}>

<h2>{product.name}</h2>

<p>{product.description}</p>

<p className="price">{product.price}</p>

</div>

))}

</div>

);

};

export default ProductList;

1. Add CSS for Form
   1. Update your styles.css file to add some basic styling for the form:

/\* src/styles.css \*/

.container {

max-width: 800px;

margin: 0 auto;

padding: 20px;

}

h1 {

text-align: center;

margin-bottom: 20px;

}

.product {

border: 1px solid #ccc;

padding: 15px;

margin-bottom: 20px;

border-radius: 5px;

}

.product h2 {

margin-top: 0;

}

.product .price {

font-weight: bold;

color: green;

}

form {

display: flex;

flex-direction: column;

}

form > div {

margin-bottom: 10px;

}

label {

font-weight: bold;

}

input {

padding: 5px;

font-size: 16px;

}

button {

padding: 10px;

font-size: 16px;

background-color: green;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

}

1. Run the Application
   1. Start the development server to see the updated application:

npm run dev