LAB MANUAL

Course: Full Stack Development Lab

Course Code	22AIL54	Credits	1
L:T:P:S	0:0:2:0	CIE Marks: SEE Marks	50:50
Course Type	PCCL	Exam Hours	3

Introduction Full stack:

Full stack web developers have the ability to design complete web applications and websites. They work on the frontend, backend, database and debugging of web applications or websites.

- Four features of FULL STACK are:
  - Frontend
  - Backend
  - Database
  - Server

Introduction to Javascript

- JavaScript (JS) is a lightweight interpreted (or just-in-time compiled) programming language
- While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB
- The standards for JavaScript are the ECMAScript Language Specification (ECMA-262) and the ECMAScript Internationalization API specification (ECMA-402).
- It has dynamic typing and it is prototype-based object-oriented.

Introduction to React

- React, sometimes referred to as a frontend JavaScript framework, is a JavaScript library created by Facebook.
- React is a tool for building UI components.
- Instead of manipulating the browser's DOM directly, React creates a virtual DOM in memory, where it does all the necessary manipulating, before making the changes in the browser DOM.
- A React application is a tree of components.There is a root component, which is like a container to which all components are added.

Introduction to JSX

- JSX stands for JavaScript XML.
- JSX allows us to write HTML in React.
- JSX makes it easier to write and add HTML in React.

Introduction to MERN

- MERN is an acronym for the four technologies that form the stack: MongoDB, Express, React, and Node. js. Developers recognize the MERN stack as a simplified development platform for building highly scalable and interactive web applications.

React Installation

Step 1: Install node.js (LTS) it act as server and in new version no need to install babel separately
https://nodejs.org/en

Step 2: Use any editor here VScode used

Step 3: create folder in any drive e.g ReactFolder
Step 4: Open VScode open terminal and type following command
D:
Cd ReactFolder
To check node an npm type
node -v
npm -v
If version shows
Step -5: for new version u need this
npm install npm -g

Step 6: type npx create-react-app myapp
npm view react version

Press y
Step 7: cd myapp
Step 8: npm strat

o/p:

React page with localhost on port 3000

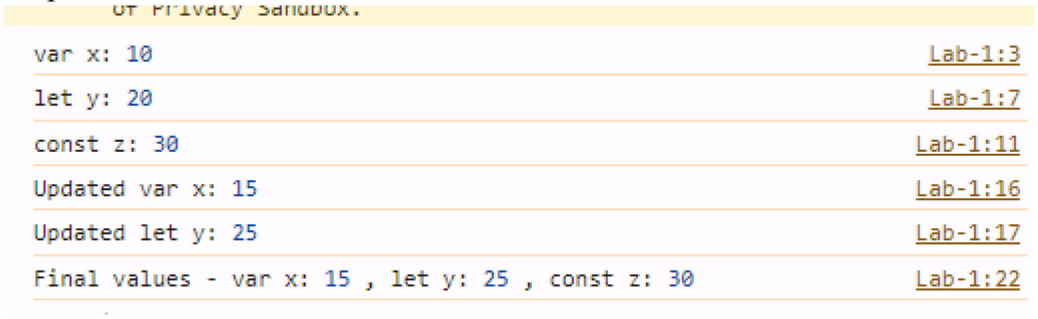
LAB PROGRAM

Lab Program 1: Write Program to Demonstrate type of variables.
Aim: Apply the concept of variables in javascript.

- Steps to write Javascript code in Console
  - Open Browser in Development mode(CTR+Shift+I for chrome)
  - Click source->snippet-> create new snippet and name it Lab1
  - Type below program and run

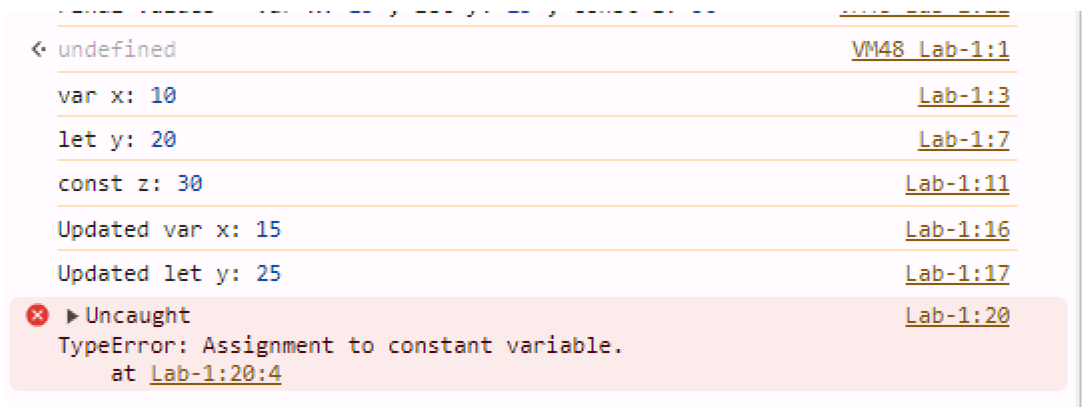
```
// Using var
var x = 10;
console.log("var x:", x); // Outputs: var x: 10

// Using let
let y = 20;
console.log("let y:", y); // Outputs: let y: 20

// Using const
const z = 30;
console.log("const z:", z); // Outputs: const z: 30

// Reassigning var and let
x = 15;
y = 25;
console.log("Updated var x:", x); // Outputs: Updated var x: 15
console.log("Updated let y:", y); // Outputs: Updated let y: 25

// Attempting to reassign const (will cause an error if uncommented)
// z = 35; // This will throw an error: Uncaught TypeError: Assignment to constant variable.

console.log("Final values - var x:", x, ", let y:", y, ", const z:", z);
```

Output:



```
   of Privacy Sandbox.
var x: 10                                              Lab-1:3
let y: 20                                              Lab-1:7
const z: 30                                            Lab-1:11
Updated var x: 15                                      Lab-1:16
Updated let y: 25                                      Lab-1:17
Final values - var x: 15 , let y: 25 , const z: 30     Lab-1:22
```

Uncommented z



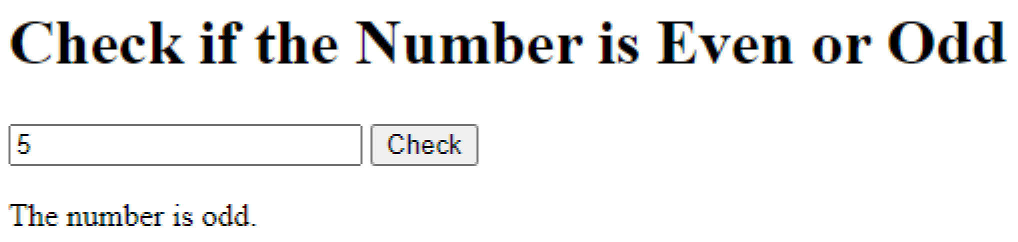
```
<· undefined                                    VM48 Lab-1:1
   var x: 10                                         Lab-1:3
   let y: 20                                         Lab-1:7
   const z: 30                                      Lab-1:11
   Updated var x: 15                                Lab-1:16
   Updated let y: 25                                Lab-1:17
⊗ ▶Uncaught                                         Lab-1:20
   TypeError: Assignment to constant variable.
       at Lab-1:20:4
```

Lab Program 2:
Aim: Apply Conditional Statements and Loop in JavaScript
Part A: Write Program to check number is even or odd

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Even or Odd Checker</title>
</head>
<body>
   <h1>Check if the Number is Even or Odd</h1>

   <input type="number" id="userInput" placeholder="Enter a number">
   <button id="checkButton">Check</button>

   <p id="result"></p>

   <script>
      // Adding event listener to the button
      document.getElementById("checkButton").addEventListener("click", function() {
         let number = document.getElementById("userInput").value;
         let result = "";

         if (number % 2 === 0) {
            result = "The number is even.";
         } else {
            result = "The number is odd.";
         }

         document.getElementById("result").textContent = result;
```

```
        });
    </script>
</body>
</html>
```

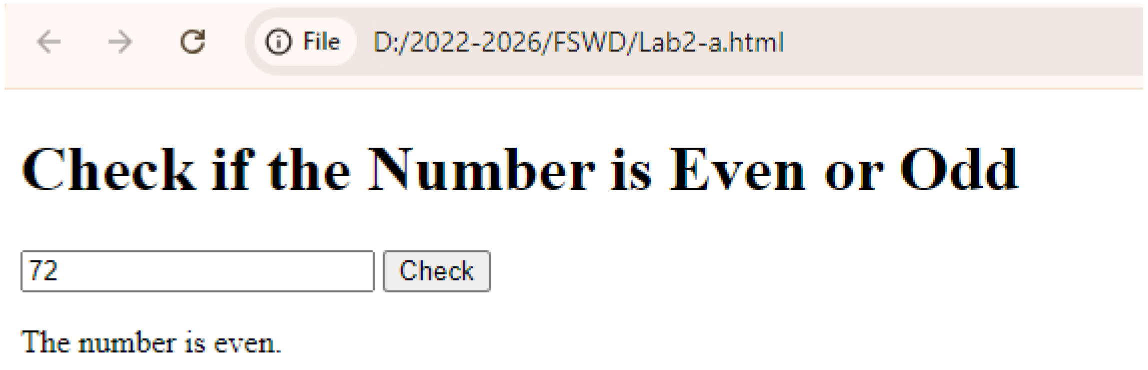
Output:



Check if the Number is Even or Odd

5 Check

The number is odd.



D:/2022-2026/FSWD/Lab2-a.html

Check if the Number is Even or Odd

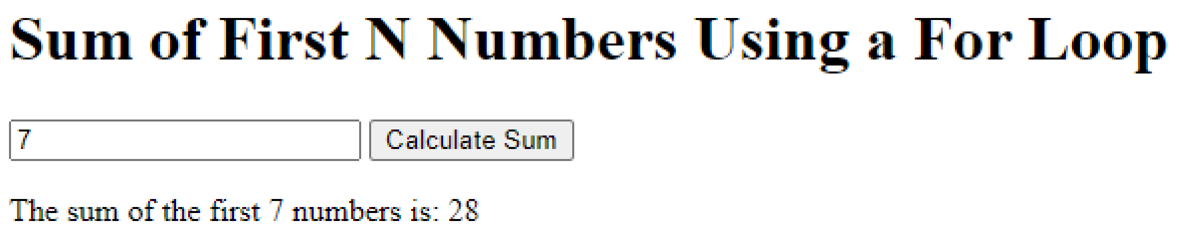
72 Check

The number is even.

Part B: Write Program to print sum of 1 to n

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sum of N Numbers - For Loop</title>
</head>
<body>
    <h1>Sum of First N Numbers Using a For Loop</h1>

    <input type="number" id="userInput" placeholder="Enter a number">
    <button id="calculateButton">Calculate Sum</button>

    <p id="result"></p>

    <script>
        document.getElementById("calculateButton").addEventListener("click", function() {
            let n = parseInt(document.getElementById("userInput").value);
            let sum = 0;

            for (let i = 1; i <= n; i++) {
                sum += i;
            }

            document.getElementById("result").textContent = "The sum of the first " + n + "
numbers is: " + sum;
        });
    </script>
</body>
</html>
```

Output:



```
Sum of First N Numbers Using a For Loop

7   Calculate Sum

The sum of the first 7 numbers is: 28
```

Lab Program 3: Write Program to calculate Area and Perimeter of Rectangle using javascript and CSS.
Aim: Apply Functions in java script

Solution:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Rectangle Calculator</title>
    <!-- Link to external CSS file -->
    <link rel="stylesheet" href="Lab3.css">
</head>
<body>
```

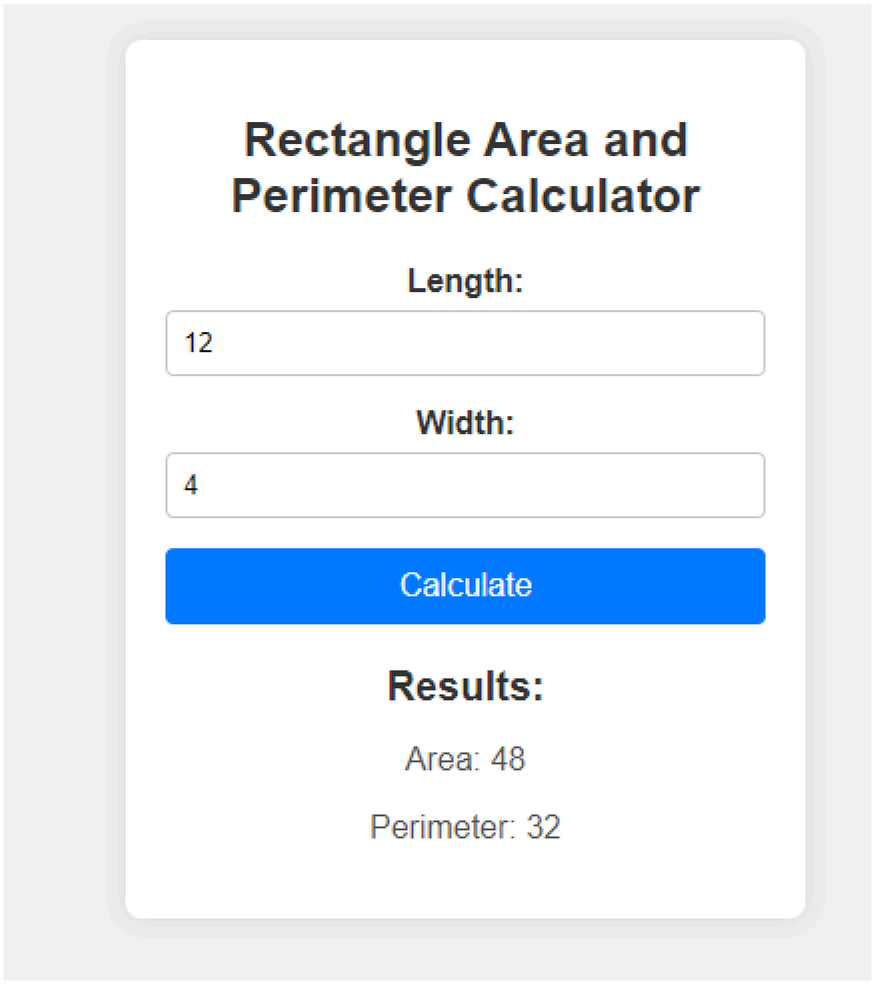
```html
    <div class="container">
        <h1>Rectangle Area and Perimeter Calculator</h1>

        <label for="length">Length:</label>
        <input type="number" id="length" required>
        <br>

        <label for="width">Width:</label>
        <input type="number" id="width" required>
        <br>

        <button id="calculateBtn">Calculate</button>

        <h2>Results:</h2>
        <p id="area">Area: </p>
        <p id="perimeter">Perimeter: </p>
    </div>

    <!-- Link to external JavaScript file -->
    <script src="Lab3.js"></script>
</body>
</html>
```

```javascript
// Function to calculate the area and perimeter of a rectangle
function calculateRectangle(length, width) {
    const area = length * width;
    const perimeter = 2 * (length + width);
    return {
        area: area,
        perimeter: perimeter
    };
}

// Function to handle the calculation and updating the results
function handleCalculate() {
    // Get the input values
    const length = parseFloat(document.getElementById('length').value);
    const width = parseFloat(document.getElementById('width').value);

    // Check if the inputs are valid numbers
    if (isNaN(length) || isNaN(width) || length <= 0 || width <= 0) {
        alert('Please enter valid positive numbers for length and width.');
        return;
    }
    // Calculate the results
    const result = calculateRectangle(length, width);

    // Update the HTML with the results
    document.getElementById('area').textContent = 'Area: ' + result.area;
    document.getElementById('perimeter').textContent = 'Perimeter: ' + result.perimeter;
}

// Add event listener to the button
document.getElementById('calculateBtn').addEventListener('click', handleCalculate);
```

```
body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
    margin: 0;
    padding: 0;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
}

.container {
    background-color: #ffffff;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    padding: 20px;
    width: 300px;
    text-align: center;
}

h1 {
    font-size: 24px;
    color: #333333;
    margin-bottom: 20px;
}

label {
    display: block;
    font-weight: bold;
    color: #333333;
    margin-bottom: 5px;
}
```

```
input[type="number"] {
   width: 100%;
   padding: 8px;
   margin-bottom: 15px;
   border: 1px solid #cccccc;
   border-radius: 4px;
   box-sizing: border-box;
}

button {
   width: 100%;
   padding: 10px;
   background-color: #007bff;
   color: white;
   border: none;
   border-radius: 4px;
   cursor: pointer;
   font-size: 16px;
}

button:hover {
   background-color: #0056b3;
}

h2 {
   font-size: 20px;
   color: #333333;
   margin-top: 20px;
   margin-bottom: 10px;
}

p {
   font-size: 16px;
   color: #555555;
}
```

Output:


Rectangle Area and Perimeter Calculator

Length:

12

Width:

4

Results:

Area: 48

Perimeter: 32

Lab Program 4 : Write A Program for creating form and apply validation for form, like all fields required, Password minimum length ,email id etc.
Aim: Apply jQuery for Form Validation

Sol:

1. **Create JavaScript file valid.js**

```
$().ready(function () {

        $("#signupForm").validate({

                // In 'rules' user have to specify all the
                // constraints for respective fields
                rules: {
                        firstname: "required",
                        lastname: "required",
                        username: {
```

```
                    required: true,
                    minlength: 2 // For length of lastname
                },
                password: {
                    required: true,
                    minlength: 5
                },
                confirm_password: {
                    required: true,
                    minlength: 5,

                    // For checking both passwords are same or not
                    equalTo: "#password"
                },
                email: {
                    required: true,
                    email: true
                },
                agree: "required"
            },
            // In 'messages' user have to specify message as per rules
            messages: {
                firstname: " Please enter your firstname",
                lastname: " Please enter your lastname",
                username: {
                    required: " Please enter a username",
                    minlength:
                        " Your username must consist of at least 2 characters"
                },
                password: {
                    required: " Please enter a password",
                    minlength:
                        " Your password must be consist of at least 5
characters"
                },
                confirm_password: {
                    required: " Please enter a password",
                    minlength:
                        " Your password must be consist of at least 5
characters",
                    equalTo: " Please enter the same password as above"
                },
                agree: "Please accept our policy"
            }
        });
});
```

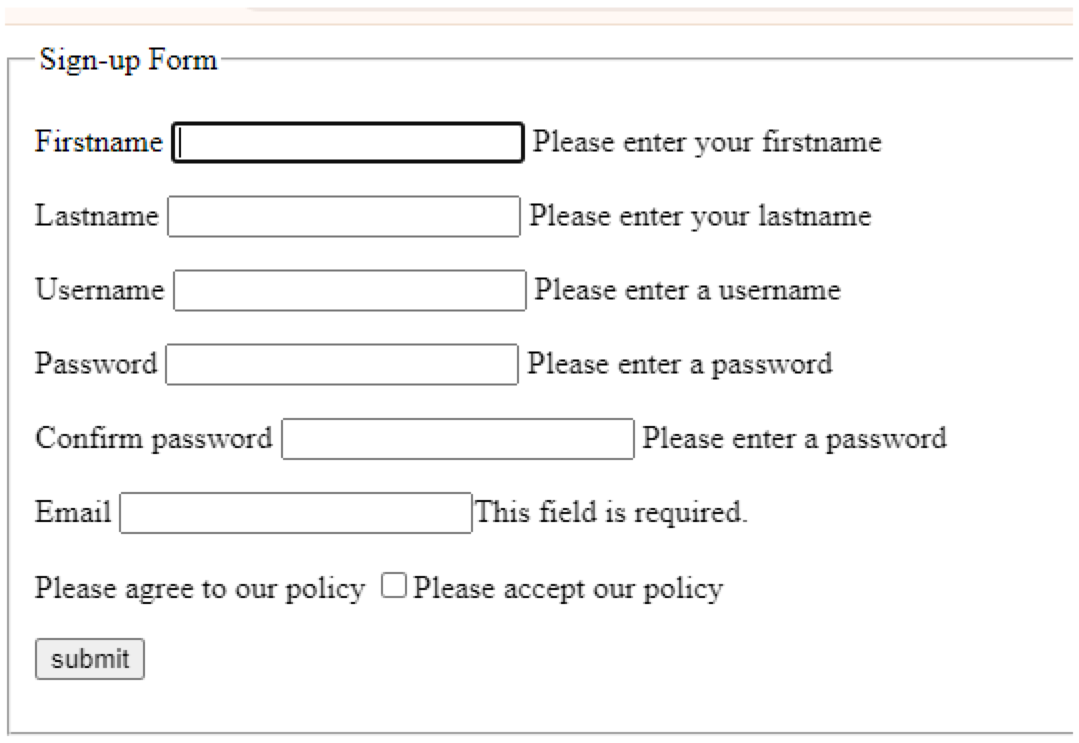
2. Create HTML file Lab4.html

```
<!doctype html>
<html>

<head>
          <meta charset="utf-8">

          <!-- below we are including the jQuery
                    and jQuery plugin .js files -->
          <script src=
"https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
          </script>
          <script src=
"https://cdn.jsdelivr.net/jquery.validation/1.16.0/jquery.validate.min.js">
          </script>
          <script src=
"https://cdn.jsdelivr.net/jquery.validation/1.16.0/additional-methods.min.js">
          </script>
</head>

<body>
<script src="valid.js"></script>
          <form
                    id="signupForm"
                    method="get"
                    action="anim.html"
                    autocomplete="off">
                    <fieldset>
                              <legend>Sign-up Form</legend>

                              <p>
                                        <label for="firstname">
                                                  Firstname
                                        </label>
                                        <input id="firstname"
                                                  name="firstname"
                                                  type="text">
                              </p>
                              <p>
                                        <label for="lastname">
                                                  Lastname
                                        </label>
                                        <input id="lastname"
                                                  name="lastname"
                                                  type="text">
```

```
</p>
<p>
        <label for="username">
                Username
        </label>
        <input id="username"
                name="username"
                type="text">
</p>
<p>
        <label for="password">
                Password
        </label>
        <input id="password"
                name="password"
                type="password">
</p>
<p>
        <label for="confirm_password">
        Confirm password
        </label>
        <input id="confirm_password"
                name="confirm_password"
                type="password">
</p>
<p>
        <label for="email">
        Email
        </label>
        <input id="email"
                name="email"
                type="email">
</p>
<p>
        <label for="agree">
        Please agree to our policy
        </label>
        <input id="agree"
                name="agree"
                type="checkbox">
</p>
<p>
        <input class="submit"
                type="submit"
                value="submit">
</p>
```

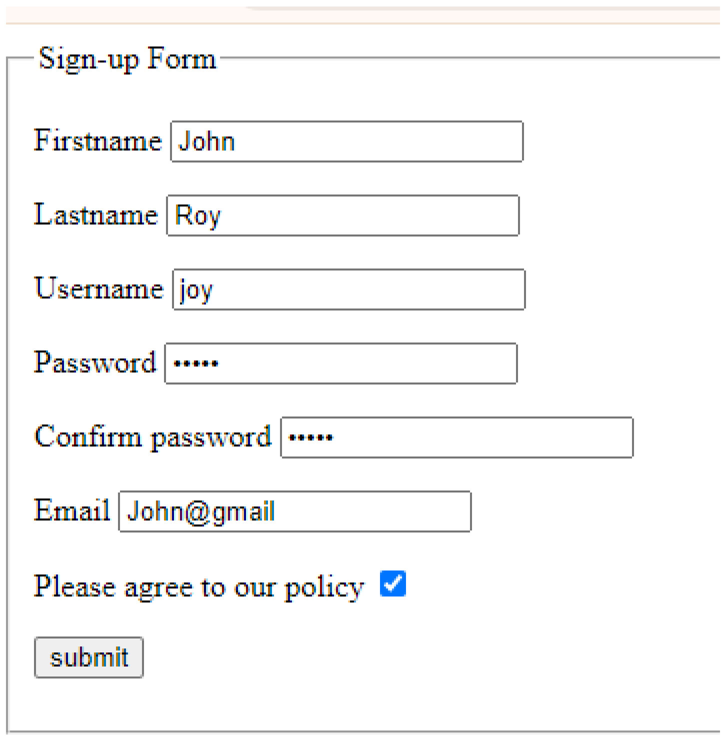
```
                    </fieldset>
          </form>
</body>

</html>
```

Output:



```
Sign-up Form

Firstname [          ] Please enter your firstname

Lastname [          ] Please enter your lastname

Username [          ] Please enter a username

Password [          ] Please enter a password

Confirm password [          ] Please enter a password

Email [          ]This field is required.

Please agree to our policy ☐ Please accept our policy

submit
```

After validation


Sign-up Form

Firstname John

Lastname Roy

Username joy

Password •••••

Confirm password •••••

Email John@gmail

Please agree to our policy ✓

Output:

Program 5: Write a program to create Match operation ADD,SUB,MUL in React.
Aim: Apply React concept in Web Development.
Step1: Create component as Calculator.js in component folder
Step 2: create cal.css in Style folder

- File Structure is as below

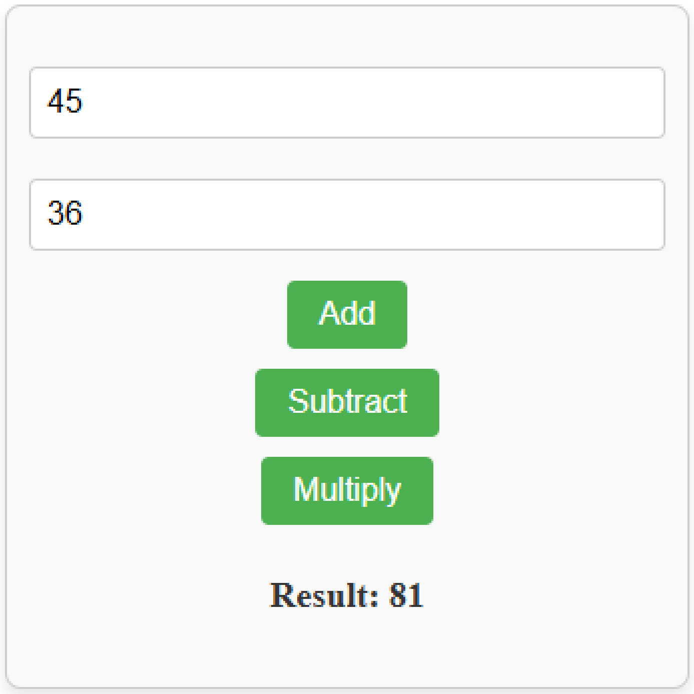
Step3: Add Calculator component in index.js

```
import React, { useState } from 'react';
import '../style/cal.css';
function Calculator() {
  const [num1, setNum1] = useState(0);
  const [num2, setNum2] = useState(0);
  const [result, setResult] = useState(0);

  const Addition = () => setResult(Number(num1) + Number(num2));
  const Subtraction = () => setResult(Number(num1) - Number(num2));
  const Multiplication = () => setResult(Number(num1) * Number(num2));

  return (
    <div className="calculator">
      <input type="number" value={num1} onChange={(e) => setNum1(e.target.value)} />
      <input type="number" value={num2} onChange={(e) => setNum2(e.target.value)} />
      <button onClick={Addition}>Add</button>
      <button onClick={Subtraction}>Subtract</button>
      <button onClick={Multiplication}>Multiply</button>
      <h2>Result: {result}</h2>
    </div>
  );
}

export default Calculator;
```

```
.calculator {
    display: flex;
    flex-direction: column;
    align-items: center;
    padding: 20px;
    border: 1px solid #ccc;
    border-radius: 8px;
    width: 300px;
    margin: 50px auto;
    background-color: #f9f9f9;
    box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
  }

  .calculator input {
    margin: 10px 0;
    padding: 8px;
    width: 100%;
    border: 1px solid #ccc;
    border-radius: 4px;
    font-size: 16px;
  }
```

```css
 .calculator button {
   margin: 5px;
   padding: 8px 16px;
   font-size: 16px;
   cursor: pointer;
   border: none;
   border-radius: 4px;
   background-color: #4CAF50;
   color: white;
   transition: background-color 0.3s;
 }

 .calculator button:hover {
   background-color: #45a049;
 }

 .calculator h2 {
   margin-top: 20px;
   font-size: 18px;
   color: #333;
 }
```

```javascript
import React from 'react';
import ReactDOM from 'react-dom/client';
import reportWebVitals from './reportWebVitals';
import Calculator from './component/LABCal.js';
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <Calculator/>
  </React.StrictMode>
);

reportWebVitals();
```



Program 6: Write a program in react to create Login Page
Aim: Apply React concept in Web Development.

Sol:
Step1: Create LoginForm.js
Step 2: Create login.css
Step 3: call LoginForm in App.js

```
import React, { useState } from 'react';
import'../style/Login.css'
function LoginForm() {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');
  const [error, setError] = useState('');

  const handleLogin = (e) => {
    e.preventDefault();
    if (username === 'admin' && password === 'password') {
      alert('Login successful');
      setUsername('');
      setPassword('');
      setError('');
    } else {
      setError('Invalid username or password');
    }
  };

  return (
    <div className="login">
      <h2>Login</h2>
      <form onSubmit={handleLogin}>
        <div>
          <label>Username</label>
          <input
            type="text"
            value={username}
            onChange={(e) => setUsername(e.target.value)}
            required
          />
        </div>
        <div>
          <label>Password</label>
          <input
            type="password"
            value={password}
            onChange={(e) => setPassword(e.target.value)}
            required
          />
        </div>
        {error && <p style={{ color: 'red' }}>{error}</p>}
        <button type="submit">Login</button>
      </form>
    </div>
  );
}

export default LoginForm;
```

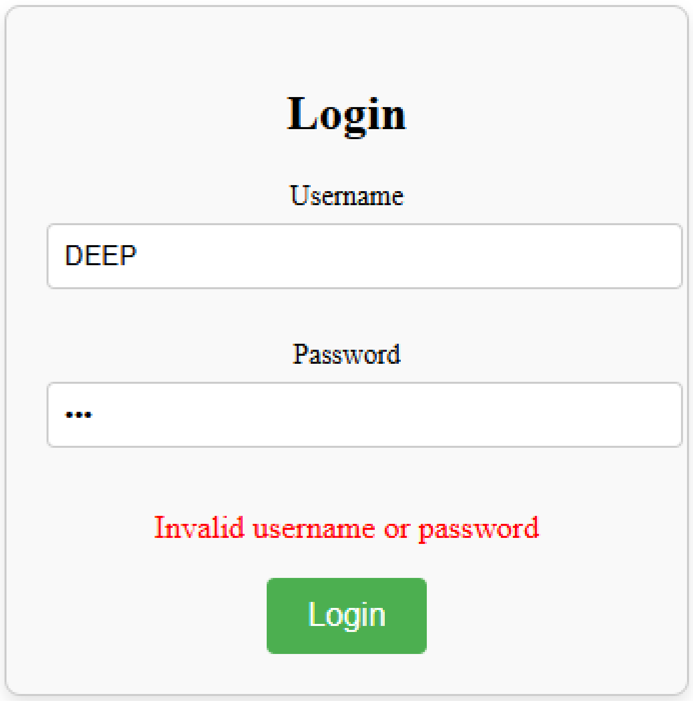
```
.login {
    max-width: 300px;
    margin: 50px auto;
    padding: 20px;
    border: 1px solid #ccc;
    border-radius: 8px;
    box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
    background-color: #f9f9f9;
    text-align: center;
  }
```

```
 .login h2 {
   margin-bottom: 20px;
 }

 .login label {
   display: block;
   margin: 10px 0 5px;
   font-size: 14px;
 }

 .login input {
   width: 100%;
   padding: 8px;
   margin-bottom: 15px;
   border: 1px solid #ccc;
   border-radius: 4px;
 }

 .login button {
   padding: 10px 20px;
   background-color: #4CAF50;
   color: white;
   border: none;
   border-radius: 4px;
   cursor: pointer;
   font-size: 16px;
 }

 .login button:hover {
   background-color: #45a049;
 }
```

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import reportWebVitals from './reportWebVitals';
import LoginForm from './component/LoginForm.js';
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
   <LoginForm/>
  </React.StrictMode>
);
```

```
reportWebVitals();
```

Output:



Login

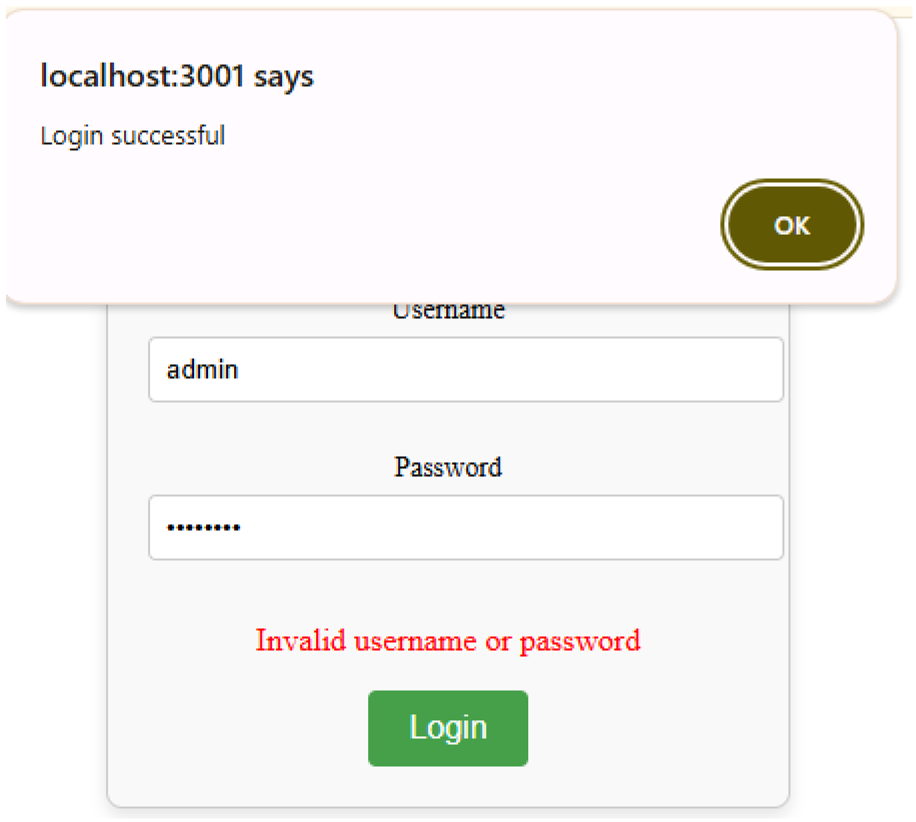
Username

DEEP

Password

•••

Invalid username or password


localhost:3001 says

Login successful

Program 7: Write React Program to create task List in React
Aim: Analyze React concept in Web Development.

a. Create List of task in react using state
b. Style the elements using css
C. Add task in above list
D. Delete Task

```
//create css

li {
   color: darkblue;
   margin: 5px;
   list-style-type: none;
}
.myDiv {
   border: 5px outset red;
   background-color: lightblue;
   text-align: center;
  }
  .button1
  {
   border-radius: 50%;
   background-color: #04AA6D; /* Green */
   border: none;
   color: white;
   padding: 20px;
   text-align: center;
   text-decoration: none;
   display: inline-block;
   font-size: 16px;
   margin: 4px 2px;
   cursor: pointer;
}
.stock::before
{
  background-color: #4224ca;
  border: none;
  color: white;
  padding: 20px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 20px;
  margin: 4px 2px;
  content:"In Stock"
}
.unavilable::before
{
  background-color: #c2155a;
  border: none;
  color: white;
  padding: 20px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 20px;
  margin: 4px 2px;
  content:"Un avilable";
}


//Edit App.js
```

```
import './App.css';
import ListTask from './component/ListTask';
import Addtask from './component/Addtask';
import { useState } from "react";
function App()
{
  const[tasks,setTasks]=useState([]);
  return (
    <div>
        <Addtask tasks={tasks} setTasks={setTasks}/>
        <ListTask tasks={tasks} setTasks={setTasks}/>

    </div>
  );
}

export default App;
```

//create new file AddATsk.js

```
import  '../CSS/mystyle.css'
import React from 'react';
import { useState } from "react";

 const  Addtask=({tasks,setTasks})=>
{
   const [taskValue,setTaskValue]=useState("");
   const handleChange=(event)=>{
       setTaskValue(event.target.value);
```

```
    }
    const handleSubmit=(event)=>{
        event.preventDefault()
        //object task
        const task={
            id:Math.floor(Math.random()*10000),
            name:taskValue,
            completed:false
        }
        setTasks([...tasks,task]);
        console.log(task);
    }
    return(
            <form onSubmit={handleSubmit}>
                <label htmlfor="">Task Name:</label>
                <input type="text" name="task" id="task" onChange={handleChange}/>
                <button className="button1" type="Submit"  >Add Task</button>
            </form>
    );
}

export default Addtask
```

//create new file ListTask.js

```
import  '../CSS/mystyle.css'

const ListTask=({tasks,setTasks})=>
{
    function DeleteItem(id)
    {
        setTasks(tasks.filter(task=>id!==task.id));
    }
    return(
      <div className="myDiv">
          <h1>Task List</h1>

      <ul>
        {tasks.map((task)=>(
        <li key={task.id}>
          <span>{task.id}-{task.name}</span>
          <button onClick={()=>DeleteItem(task.id)} className="button1">Del</button>

        </li>
        ))}
      </ul>
      </div>

    );
}
export default ListTask;
```

//create new file index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App/>
  </React.StrictMode>
);
```

Program 8: Create different links like Home,About US ,Contact Us,Details using react-Router and Navlink
Aim: Design Navigation using React concept in Web Development

React Router

Routing is a process in which a user is directed to different pages based on their action or request. ReactJS Router is mainly used for developing Single Page Web Applications. React Router is used to define multiple routes in the application. When a user types a specific URL into the browser, and if this URL path matches any 'route' inside the router file, the user will be redirected to that particular route.
React Router is a standard library system built on top of the React and used to create routing in the React application using React Router Package. It provides the synchronous URL on the browser with data that will be displayed on the web page

Need of React Router

React Router plays an important role to display multiple views in a single page application. Without React Router, it is not possible to display multiple views in React applications. Most of the social media websites like Facebook, Instagram uses React Router for rendering multiple views.

1. **react-router-native:** It is used for mobile applications.
2. **react-router-dom:** It is used for web applications design.

What is Route?

It is used to define and render component based on the specified path. It will accept components and render to define what should be rendered.

Step 1:
Install React Router if you haven't already:

```
npm install react-router-dom
```

Step 2: Create Components like Home.js,About.js etc

```
function Home() {
  return (
    <div>
      <h2>Home Page</h2>
      <p>Welcome to the Home page!</p>
    </div>
  );
}

export default Home;


function About() {
  return (
    <div>
      <h2>About Us</h2>
      <p>This is the About page. We are a company dedicated to providing the best service.</p>
    </div>
  );
}

export default About;
```

Step 3: Edit App.js

Program 9: Write A Program to create JSON API to fetch data
Aim: Learn to create JSON API and fetch data using React

1. Create data.json

```
{
   "Fruits":[
      {
         "id":1,
         "name":"Apple",
         "price":500,
         "stock":true
      },
      {
         "id":2,
         "name":"Mango",
         "price":800,
         "stock":true


      },
      {
         "id":3,
         "name":"Orange",
         "price":300,
         "stock":true
      },
      {
         "id":4,
         "name":"Apple",
         "price":500,
         "stock":true
      },
      {
         "id":5,
         "name":"Mango",
         "price":800,
         "stock":true


      },
      {
         "id":6,
         "name":"Orange",
         "price":300,
         "stock":true
      }
```

```
            ]

        }
```

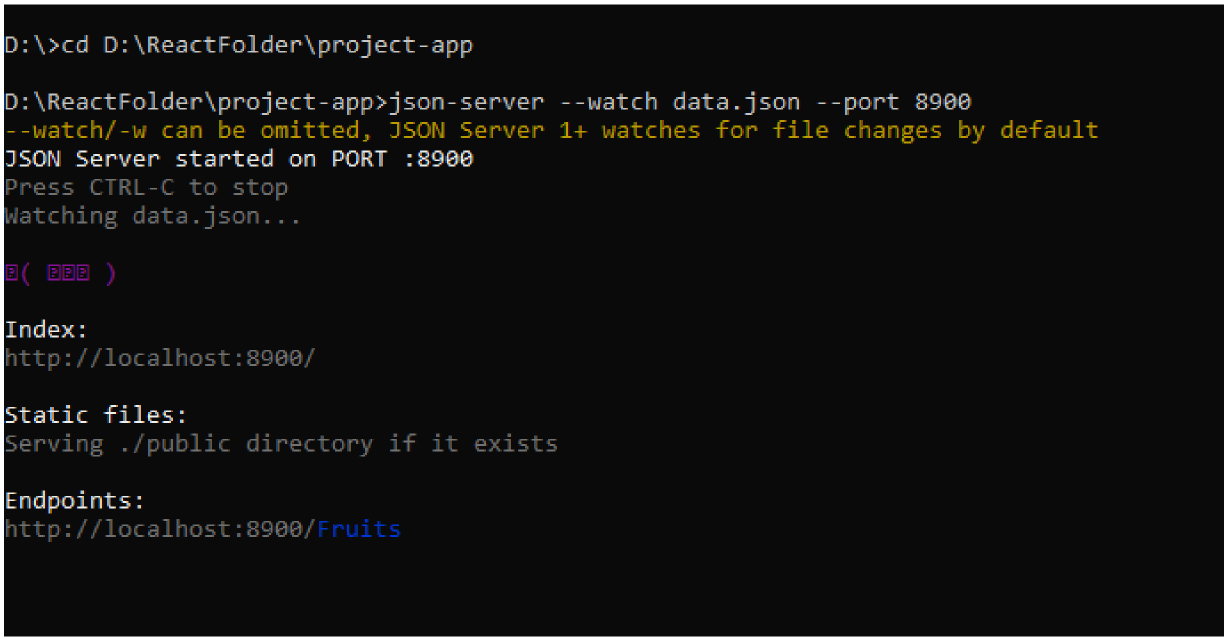
2. Create CSS mystyle.css

```
li {
   color: darkblue;
   margin: 5px;
   list-style-type: none;
}
.myDiv {
   border: 5px outset red;
   background-color: lightblue;
   text-align: center;
 }
 .button1
 {
   border-radius: 50%;
   background-color: #04AA6D; /* Green */
   border: none;
   color: white;
   padding: 20px;
   text-align: center;
   text-decoration: none;
   display: inline-block;
   font-size: 16px;
   margin: 4px 2px;
   cursor: pointer;
}
.stock::before
{
  background-color: #4224ca;
  border: none;
  color: white;
  padding: 20px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 20px;
  margin: 4px 2px;
  content:"In Stock"
}
.unavilable::before
{
  background-color: #c2155a;
  border: none;
  color: white;
  padding: 20px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 20px;
  margin: 4px 2px;
  content:"Un avilable";
}
```

3. Step Open Cmd and create API by using command
    a. Open cmd and type command to install json server

       ```
       Install json-server
       ```
    b. Change directory where React app is installed and type following

```
json-server --watch data.json --port 8900
```



```
D:\>cd D:\ReactFolder\project-app

D:\ReactFolder\project-app>json-server --watch data.json --port 8900
--watch/-w can be omitted, JSON Server 1+ watches for file changes by default
JSON Server started on PORT :8900
Press CTRL-C to stop
Watching data.json...

?( ??? )

Index:
http://localhost:8900/

Static files:
Serving ./public directory if it exists

Endpoints:
http://localhost:8900/Fruits
```

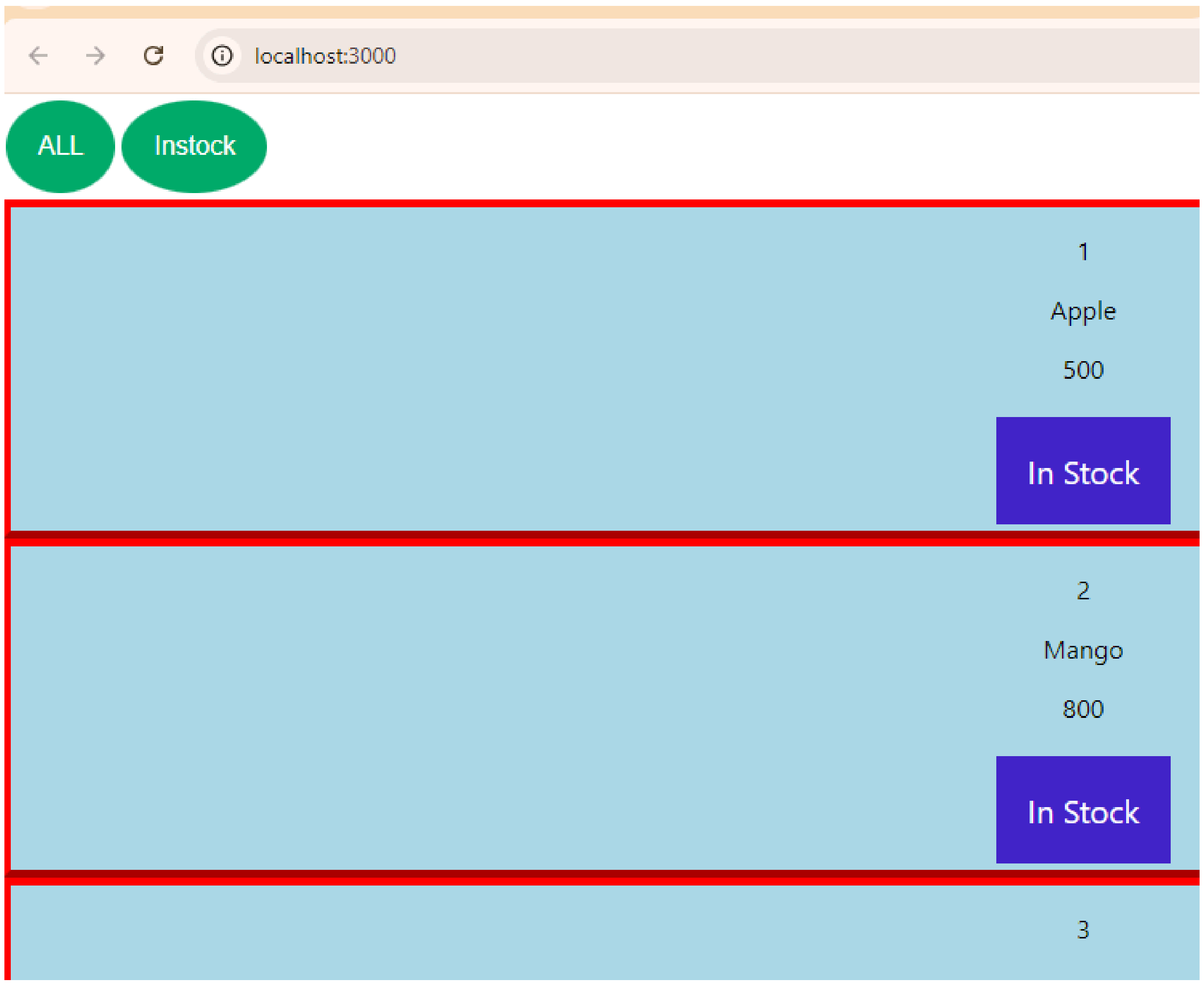
4. Create file ProductList.js

```
import './mystyle.css'
import {useState,useEffect} from "react"
const ProductList=()=>
{
   const [products,setProducts]=useState([]);
   const [url,setUrl]=useState("http://localhost:8900/Fruits");
   useEffect(()=>{
   fetch(url)
   .then(resposnse=>resposnse.json())
   .then(data=>setProducts(data));
   },[url]);
   return(
       <section>
       <button className='button1'
onClick={()=>setUrl("http://localhost:8900/Fruits")}>ALL</button>
       <button className="button1"
onClick={()=>setUrl("http://localhost:8900/Fruits?stock=true")}>Instock</button>
```

```
        {
        products.map((mydata)=>(//products is from array line 4
<div className='myDiv' key={mydata.id}>
    <p>{mydata.id}</p>
    <p>{mydata.name}</p>
    <p>{mydata.price}</p>
    <span className={mydata.stock?"stock":"unavilable"}>{mydata.stock}</span>
        </div>
    ))}
    </section>
    )//return closing
}
export default ProductList;
```

5. Type in VS code terminal npm start

//Output



Program 10: Write A Program to insert Data in MongoDB Database
Aim: Learn how to use Database with Front end

Step 1: Create Folder Myserver
Step 2: Install mongoose,and express with cors
npm install express mongoose cors

Step 3: Create file server.js

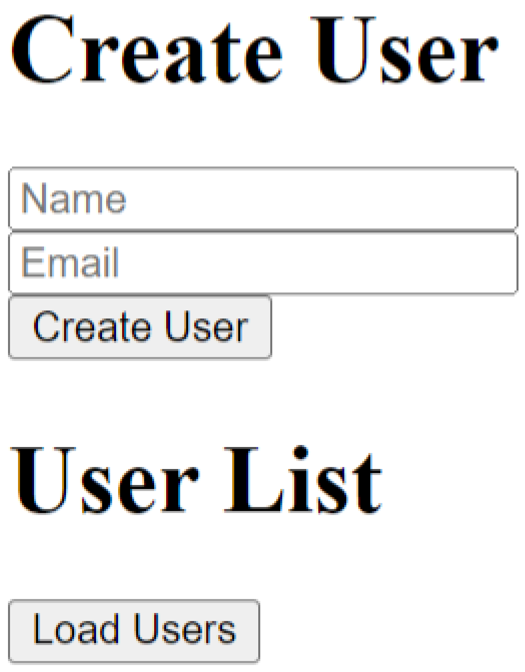
```
const express = require('express');

const mongoose = require('mongoose');


const app = express();

const port = 8000;
```

```
mongoose.connect('mongodb://localhost:27017/newCollection', { useNewUrlParser: true,
useUnifiedTopology: true })
  .then(() => {
    console.log('Connected to MongoDB');
  })
  .catch((error) => {
    console.error('Error connecting to MongoDB:', error);
  });

const userSchema = new mongoose.Schema({
  name: String,
  email: String
});

const User = mongoose.model('User', userSchema);
const cors = require("cors");
app.use(cors());
app.use(express.json());

app.post('/api/users', async (req, res) => {
  try {
    const { name, email } = req.body;

    if (!name || !email) {
      return res.status(400).send('Name and email are required');
    }

    const user = new User({ name, email });
    await user.save();

    res.send('User created successfully');
  } catch (error) {
    console.error('Error creating user:', error);
    res.status(500).send('Internal server error');
  }
});

app.listen(port, () => {
  console.log(`Server is running on port ${port}`);
});
```

Step 4: create MernDemo.js

```
import React, { useState } from 'react';
import axios from 'axios';
import { useEffect } from 'react';
function MernDemo() {
  const [name, setName] = useState('');
  const [email, setEmail] = useState('');
  const [users, setUsers] = useState([]);

//to display we will use useEffect

  const handleSubmit = async (e) => {
    e.preventDefault();

    try {
      await axios.post('http://localhost:8000/api/users', { name, email });
      alert('User created successfully');
    } catch (error) {
      console.error('Error creating user:', error);
      alert('Failed to create user');
    }
  };

  return (
    <div>
      <h1>Create User</h1>
      <form onSubmit={handleSubmit}>
```

```
            <input type="text" placeholder="Name" value={name} onChange={(e) =>
setName(e.target.value)} required /><br />
            <input type="email" placeholder="Email" value={email} onChange={(e) =>
setEmail(e.target.value)} required /><br />
            <button type="submit">Create User</button>
          </form>



        </div>
    );
}

export default MernDemo;
//Output
```


Create User
Name
Email
Create User
User List
Load Users