EX.NO: DATE:	**Write a Program to create a simple webpageusing HTML and CSS**

AIM:

To write a program to create a simple webpage using HTML and CSS.

ALGORITHM:

STEP 1: Start the program.

STEP 2: Define the structure of the code.

STEP 3: Insert the HTML code inside the <html> </html> tag.

STEP 4: Insert the Header, Footer and Main content for the webpage.

STEP 5: Insert the CSS code inside the <style></style> tag.

STEP 6: By using CSS code style the Header, Footer and Main content for the webpage

STEP 7: Stop the program

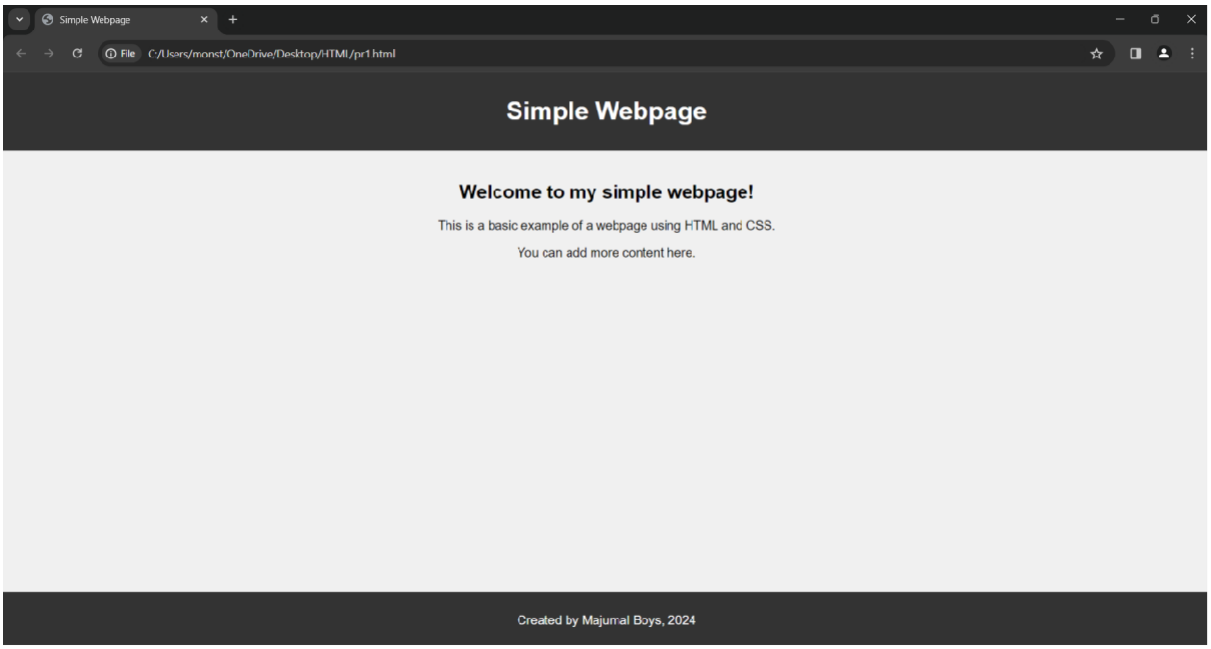
PROGRAM:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Simple Webpage</title>
    <style>
        /* CSS styles */
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f0f0f0;
        }
```

```
        header {
            background-color: #333;
            color: #fff;
            padding: 10px 0;
            text-align: center;
        }
        section {
            padding: 20px;
            text-align: center;
        }
        footer {
            background-color: #333;
            color: #fff;
            padding: 10px 0;
            text-align: center;
            position: fixed;
            bottom: 0;
            width: 100%;
        }
    </style>
</head>
<body>
    <!-- Header -->
    <header>
        <h1>Simple Webpage</h1>
    </header>
    <!-- Main Content -->
    <section>
        <h2>Welcome to my simple webpage!</h2>
        <p>This is a basic example of a webpage using HTML and CSS.</p>
        <p>You can add more content here.</p>
    </section>
```

```
    <!-- Footer -->
    <footer>
                <p>Created by Manjumal Boys, 2024</p>
    </footer>
</body>
</html>
```

OUTPUT:



RESULT:

EX.NO: DATE:	**Write a program to build a Chat module using HTML CSSand JavaScript**

AIM:

To write a program to build a Chat module using HTML CSS and JavaScript

ALGORITHM:

STEP 1: Start the program.

STEP 2: Open your preferred text editor (like Notepad, Sublime Text, Visual Studio Code, etc.) and create a new file.

STEP 3: HTML ,CSS and JavaScript are used to design the chat module.

STEP 4: Save the file with a meaningful name and **.html** extension. For example, you could name it **chat_module.html.**

STEP 5: Once saved, double-click the file to open it in your default web browser.

STEP 6: You'll see the chat module interface with an input field to type messages anda send button.

STEP 7: You can type a message, click the send button, and you'll see your message displayed in the chat box above.

STEP 8: Stop the program.

PROGRAM:

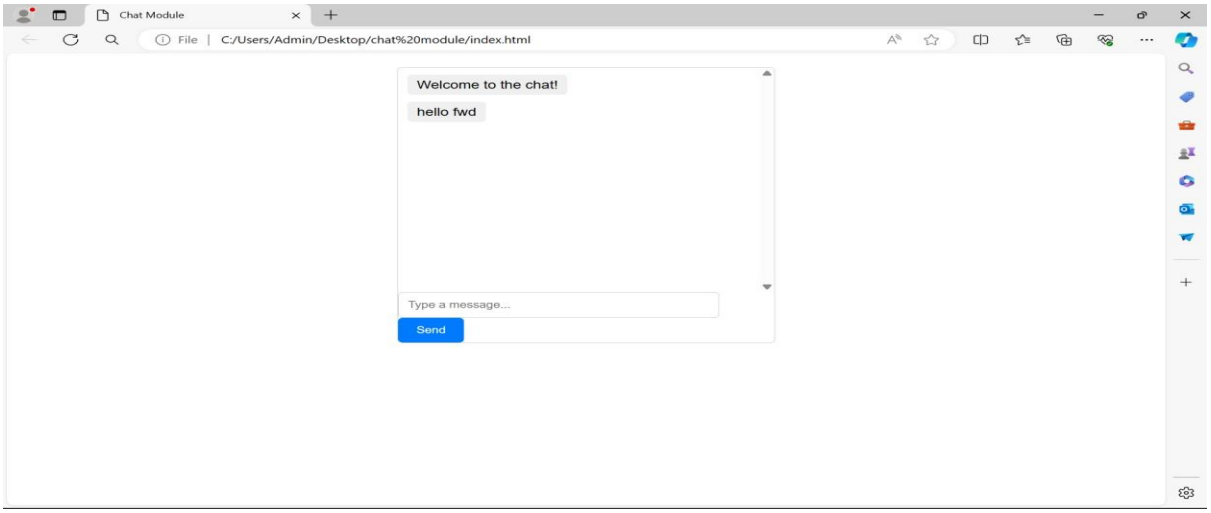
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Chat Module</title>
  <style>
    body {
```

```
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
}

.chat-container {
  max-width: 400px;
  margin: 20px auto;
  border: 1px solid #ccc;
  border-radius: 5px;
  overflow: hidden;
}

.chat-box {
  height: 300px;
  overflow-y: scroll;
  padding: 10px;
}

.chat-message {
  margin-bottom: 10px;
}

.chat-message .message {
  background-color: #f0f0f0;
  padding: 5px 10px;
  border-radius: 5px;
  display: inline-block;
}

#message-input {
  width: calc(100% - 80px);
  padding: 10px;
  border: 1px solid #ccc;
```

```
      border-radius: 5px;
    }

    #send-btn {
      width: 70px;
      padding: 10px;
      border: none;
      background-color: #007bff;
      color: #fff;
      border-radius: 5px;
      cursor: pointer;
    }

    #send-btn:hover {
      background-color: #0056b3;
    }
  </style>
</head>
<body>
  <div class="chat-container">
    <div class="chat-box" id="chat-box">
      <div class="chat-message">
        <span class="message">Welcome to the chat!</span>
      </div>
    </div>
    <input type="text" id="message-input" placeholder="Type a message...">
    <button id="send-btn">Send</button>
  </div>
  <script>
    document.addEventListener("DOMContentLoaded", function() {
      const messageInput = document.getElementById("message-input");
      const sendBtn = document.getElementById("send-btn");
      const chatBox = document.getElementById("chat-box");
```

```
      sendBtn.addEventListener("click", function() {
        const  message = messageInput.value.trim();
        if (message !== "") {
        appendMessage(message);
        messageInput.value = "";
        }
      });

      function appendMessage(message) {
        const messageElement = document.createElement("div");
        messageElement.classList.add("chat-message");
        messageElement.innerHTML = '<span class="message">' + message + '</span>';
        chatBox.appendChild(messageElement);
        chatBox.scrollTop = chatBox.scrollHeight;
      }
    });
  </script>
</body>
</html>
```

OUTPUT:



RESULT:

EX.NO: DATE:	**Write a program to create a simple calculator Application using React JS**

AIM:

To write a program to create a simple calculator Application using React JS

ALGORITHM:

STEP 1: Start the program.

STEP 2: Set up a new React project using Create React App or any other method youprefer.

STEP 3: Create a functional component called App, which will serve as the maincomponent for your calculator application

STEP 4: Use the useState hook to create state for managing the input value of thecalculator.

STEP 5: including the input field for displaying the value and buttons for numbers,operators, and special functions like clear and equal.

STEP 6: Define event handlers for button clicks to update the input value based onuser interactions.

STEP 7: Implement logic to evaluate the expression entered by the user when theequal button is clicked.

STEP 8: Stop the program.

PROGRAM:

MAIN.JSX

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

ReactDOM.createRoot(document.getElementById('root')).render(
```

```
  <React.StrictMode>
    <App />
  </React.StrictMode>,
)
```

APP.JSX

```
import { useState } from 'react'
import './App.css'

function App() {
  const [value,setValue] = useState('');


  return (
    <>
    <div className="container">
      <div className="calculator">
       <form action="">
        <div className='display'>
          <input type="text" value={value}/>
        </div>
        <div>
          <input type="button" value="AC" onClick={e => setValue('')}/>
          <input type="button" value="C" onClick={e => setValue(value.slice(0,-
1))}/>
          <input type="button" value="." onClick={e => setValue(value +
e.target.value)}/>
          <input type="button" value="/" onClick={e => setValue(value +e.target.value)}/>
        </div>
        <div>
          <input type="button" value="7" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="8" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="9" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="*" onClick={e => setValue(value +e.target.value)}/>
```

```
        </div>
        <div>
          <input type="button" value="4" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="5" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="6" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="+" onClick={e => setValue(value +e.target.value)}/>
        </div>
        <div>
          <input type="button" value="1" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="2" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="3" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="-" onClick={e => setValue(value +e.target.value)}/>
        </div>
        <div>
          <input type="button" value="0" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="00" onClick={e => setValue(value +e.target.value)}/>
          <input type="button" value="=" className='equal'onClick={e => setValue(eval(value))}/>
          </div>
        </form>
      </div>
    </div>
    </>
  )
}
export default App
```

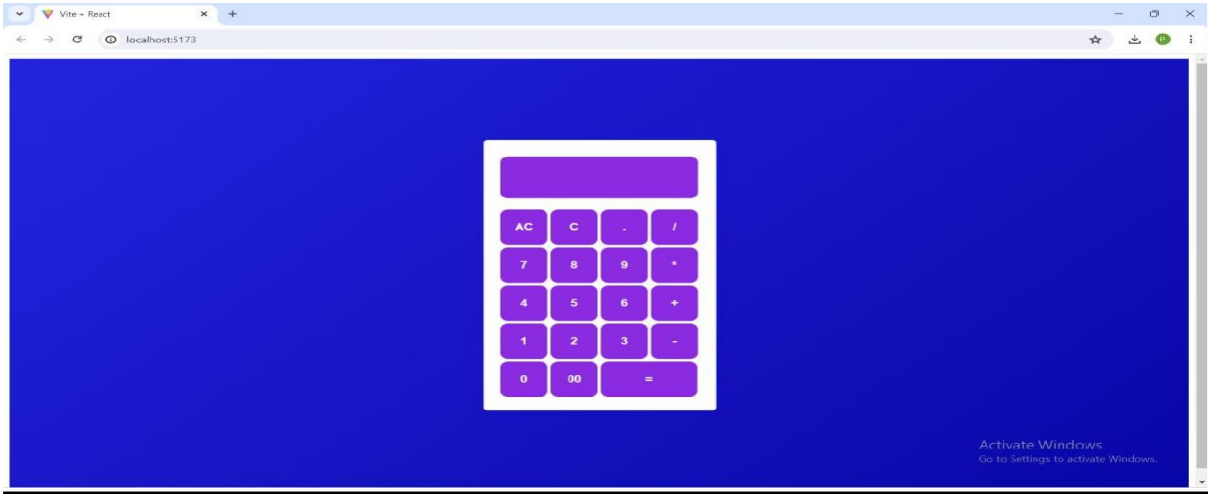
APP.CSS

```
.container {
  width: 100%;
  height: 100vh;
  display: flex;
  align-items: center;
```

```
  justify-content: center;
  background: linear-gradient(140deg,rgb(36,36,223),rgb(7, 7, 166));
}
.calculator{
  padding: 20px;
  border-radius: 5px;
  background-color: white;
}
form input{
  border: none;
  outline: 0;
  width: 60px;
  height: 60px;
  font-size: 16px;
  background-color: blueviolet;
  margin: 2px;
  border-radius: 10px;
  color: white;
  font-weight: bolder;
  cursor: pointer;
}
form input[type="button"]:hover{
  background-color:cadetblue;
}
form .display{
  display: flex;
  justify-content: flex-end;
  margin: 5px 0px 15px 0px;
}
form .display input{
  text-align: right;
```

```
  flex:1;
  font-size: 40px;
  padding: 5px 10px;
}
form input.equal{
  width: 123px;
}
```

OUTPUT:



RESULT:

EX.NO: DATE:	**Write a program to create weather application using React JS**

AIM:

To write a program to create weather application using React JS

ALGORITHM:

STEP 1: Start the program.

STEP 2: Initialize State Variables.

STEP 3: Define the 'searchLocation' Function.

STEP 4: Construct the API URL and Make the HTTP GET Request.

STEP 5: Process the API Response.

STEP 6: Define the UI Structure.

STEP 7: Export the Component.

STEP 8: Add CSS Styling.

STEP 9: Stop the program.

PROGRAM:

Main.jsx

```
import React from "react";
import ReactDOM from "react-
dom/client";import App from
"./App.jsx";

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

App.jsx:

```
import React, { useState } from "react"; import
"./App.css";
import axios from "axios";

function App() {
  const [data, setData] = useState({});
  const [location, setLocation] = useState("");

  const searchLocation = (event) => { if
    (event.key === "Enter") {
consturl=https://api.openweathermap.org/data/2.5/weather?q=${location},IN&appid=9
1349473260125a4ffd3f6169314079a;
      axios.get(url)
        .then((response) => {
          // Convert temperature from Kelvin to Celsius
          const celsiusTemp = Math.round(response.data.main.temp - 273.15);

          // Update state with converted temperaturesetData({
            ...response.data,
            main: {
              ...response.data.main,temp:
              celsiusTemp,
            },
          });

          console.log(response.data);
        })
        .catch((error) => {
          console.error("Error fetching weather data:", error);
        });
    }
  };
```

```
return (
  <>
    <div className="app">
      <div className="search">
        <input
          value={location}
          onChange={(event) => setLocation(event.target.value)}
          onKeyPress={searchLocation}
          placeholder="Enter the Location"
          type="text"
        />
      </div>
      <div className="container">
        <div className="top">
          <div className="location">
            <p>{data.name}</p>
          </div>
          <div className="temp">
            <h1>{data.main ? ${data.main.temp}°C : ""}</h1>
          </div>
          <div className="description">
            <p>{data.weather ? data.weather[0].description : ""}</p>
          </div>
        </div>
        <div className="bottom">
          <div className="feels">
            <p>{data.main ? ${data.main.feels_like}°C : ""}</p>
            <p>Feels Like</p>
          </div>
          <div className="humidity">
            <p className="bold">
```

```
            {data.main ? ${data.main.humidity}% : ""}
          </p>
          <p>Humidity</p>
        </div>
        <div className="wind">
          <p className="bold">
            {data.wind ? ${data.wind.speed} MPH : ""}
          </p>
          <p>Wind Speed</p>
        </div>
      </div>
    </div>
  </div>
 </>
);
}

export default App;
```

App.css:

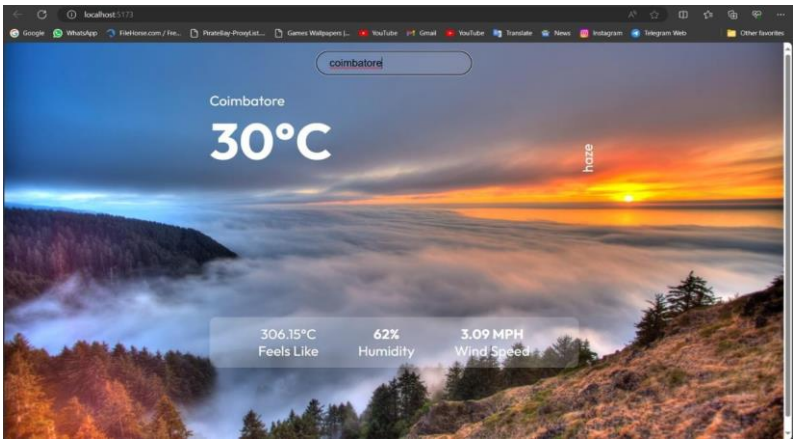
```
* {
  box-sizing: border-box;
  margin: 0;
  padding: 0;

}

body {
  font-family: "Outfit", "Segoe UI", "Roboto", "Oxygen", "Ubuntu",
    "Cantarell","Fira Sans", "Droid Sans", "Helvetica Neue", sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
```

```
  background-image: url("./assets/sunset.jpg"); /* Set background
  image */background-repeat: no-repeat;
  background-size: cover;
  background-position:
  center;
  background-attachment: fixed; /* Fixed background */
  background-color: #000; /* Added a background color for better readability */
 }
 p {
  font-size: 1.6rem;
 }
h1 {
  font-size: 6rem;
 }
 .app {
  width: 100%;
  min-height: 100vh; /* Change height to min-height to ensure it fills the viewport */
  position: relative;
  color: #fff;
 }
 .app .search {
  text-align:
  center;padding:
  1rem;
 }
 .app input {
  padding: 0.7rem
  1.5rem;font-size:
  1.2rem;
  border-radius: 25px;
  border: 1px solid rgba(255, 255, 255, 0.8);
  background: rgba(255, 255, 255, 0.1);
```

```
}
::placeholder
  { color:
  #f8f8f8;
}
.app .container {
  max-width:
700px;height:
  700px; margin:
  auto; padding: 0
  1rem; position:
  relative; top:
  10%;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
}
.app .top {
  width: 100%;
  margin: 1rem
  auto;
}
.app  .description
  {          position:
  relative;      left:
  100%;
  transform-origin: 0 0;
  transform: rotate(269deg);
}
.app .bottom
  {display:
  flex;
  justify-content: space-
```

```
 evenly;text-align: center;
 width: 100%;
 margin: 10rem
 auto;padding:
 1rem; border-
 radius: 12px;
 background-color: rgba(255, 255, 255, 0.2);
}
.bold {
 font-weight: 700;
}
```

OUTPUT:



RESULT:

EX.NO: DATE:	**Write a program to create and bulid a star rating system**

AIM:

To Write a program to create and bulid a star rating system

ALGORITHM:

STEP 1: Start the program.

STEP 2: Open your Visual Studio Code and create a new terminal.

STEP 3: Create a functional component named RatingStar with a prop noofStar.

STEP 4: Use useState hook to manage 'rating'.

STEP 5: Define handleClick(getCurrentId) to update 'rating' state to getCurrentId.

STEP 6: Import useState hook and RatingStar component into App.js.

STEP 7: Render RatingStar component within the App component.

STEP 8: Define .active class to style stars with yellow color into css.

STEP 9: Stop the program.

PROGRAM:

Rating Star.jsx

```
import { useState } from "react";
import { FaStar } from "react-icons/fa";
import "./style.css";

const RatingStar = ({ noofStar = 5 }) => {
  const [rating, setRating] = useState(0);
  const [hover, setHover] = useState(0);

  const handleClick = (getCurrentId) => {
    setRating(getCurrentId);
```

```
  };
  const handleMouseEnter = (getCurrentId) => {
    setHover(getCurrentId);
  };
  const handleMouseLeave = () => {
    setHover(rating);
  };

  return (
    <div>
      {[...Array(noofStar)].map((_, index) => {
        index += 1;
        return (
          <FaStar
            key={index}
            className={index <= (hover || rating) ? "active" : "inactive"}
            onClick={() => handleClick(index)}
            onMouseEnter={() => handleMouseEnter(index)}
            onMouseLeave={() => handleMouseLeave(index)}
          />
        );
      })}
    </div>
  );
};

export default RatingStar;
```

App.jsx:

```
import { useState } from "react";
import "./App.css";
import back from "./assets/image/back.png";
import laptop from "./assets/image/Laptop.png";
```

```
import RatingStar from "./component/RatinngStar";

function App() {
  const [count, setCount] = useState(0);

  return (
    <><div className="app">
        <h1>Which Product would you like More,Please Rate it</h1>
        <div className="image">
          <div className="image1">
            <img src={back} alt="This is Phone" />
            <br></br>
            <RatingStar />
          </div>
          <div className="image2">
            <img src={laptop} alt="This is Laptop" />
            <br></br>
            <RatingStar />
          </div>
        </div>
      </div>
    </>
  );
}
export default App;
```

Style.css

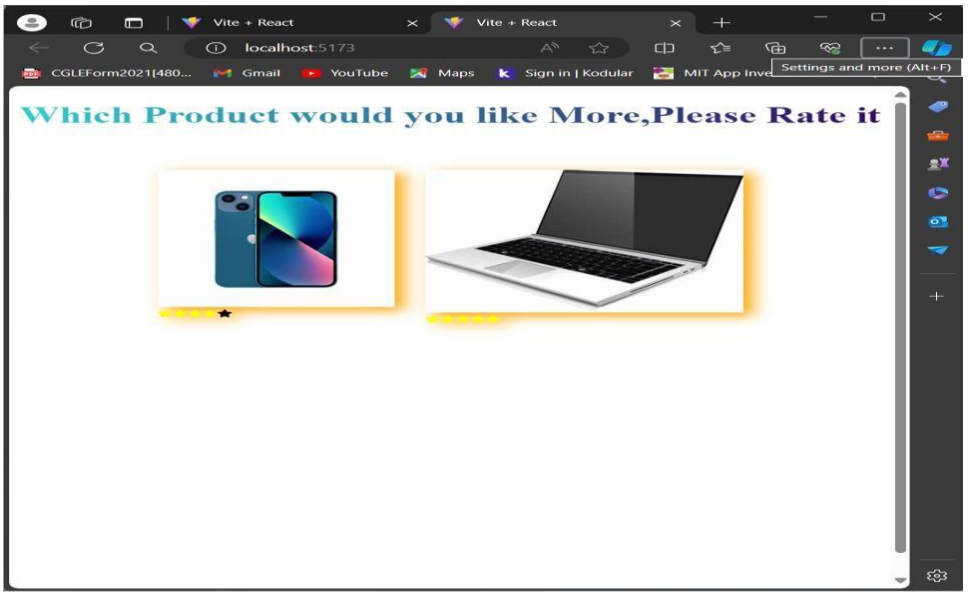
```
.active {
  color: yellow;
}
```

App.css:

```
app {
  width: 100%;
  height: 100vh;
```

```
}
h1 {
  text-align: center;
  background: linear-gradient(to right, #30cfd0 0%, #330867 100%);
  background-clip: text;
  color: transparent;
}
image {
  margin: 0 0 20px 0;
  display: flex;
  flex-wrap: wrap;
  justify-content: space-evenly;
  filter: drop-shadow(10px 7px 10px orange);
  transform: scale(0.75);
}
```

OUTPUT:



RESULT:

Ex.No: Date:	**Write a program to create and build a password strength Check using Reactjs**

AIM:

To write a Program to create and build a Password Strength Check using Reactjs

ALGORITHM:

STEP 1: Start the program.

STEP 2: Set up a new React project using Create React App or any other method preferred.

STEP 3: Create a functional component called PasswordStrengthIndicator, which will serve as the main component for your password strength indicator application.

STEP 4: Use the useState hook to create state for managing the password input value and the strength of the password.

STEP 5: Define a function called evaluatePasswordStrength(password) to evaluate the strength of the password based on certain criteria.

STEP 6: Inside the function evaluatePasswordStrength(password):

a.] Initialize a variable score to 0.
b.] Check if the password is not empty. If it's empty, return an empty string.
C.] Check the length of the password. If it's greater than 8 characters, increment thescore by 1.
d.] Check if the password contains lowercase letters. If it does, increment the scoreby 1.
e.] Check if the password contains uppercase letters. If it does, increment the scoreby 1.
f.] Check if the password contains numbers. If it does, increment the score by 1.
g.] Check if the password contains special characters. If it does, increment the scoreby 1.
h.] Based on the score, determine the strength of the password (Weak, Medium, orStrong) and return it.

STEP 7: In the PasswordStrengthIndicator component, return JSX elements:

a.] Render an input field of type password for users to enter their password.- Bind the input field value to the password state and update it using onChange event.
b.] Call the evaluatePasswordStrength function inside onChange event to update thestrength state based on the entered password.
c.] Render a strength indicator element to visually represent the strength of thepassword.

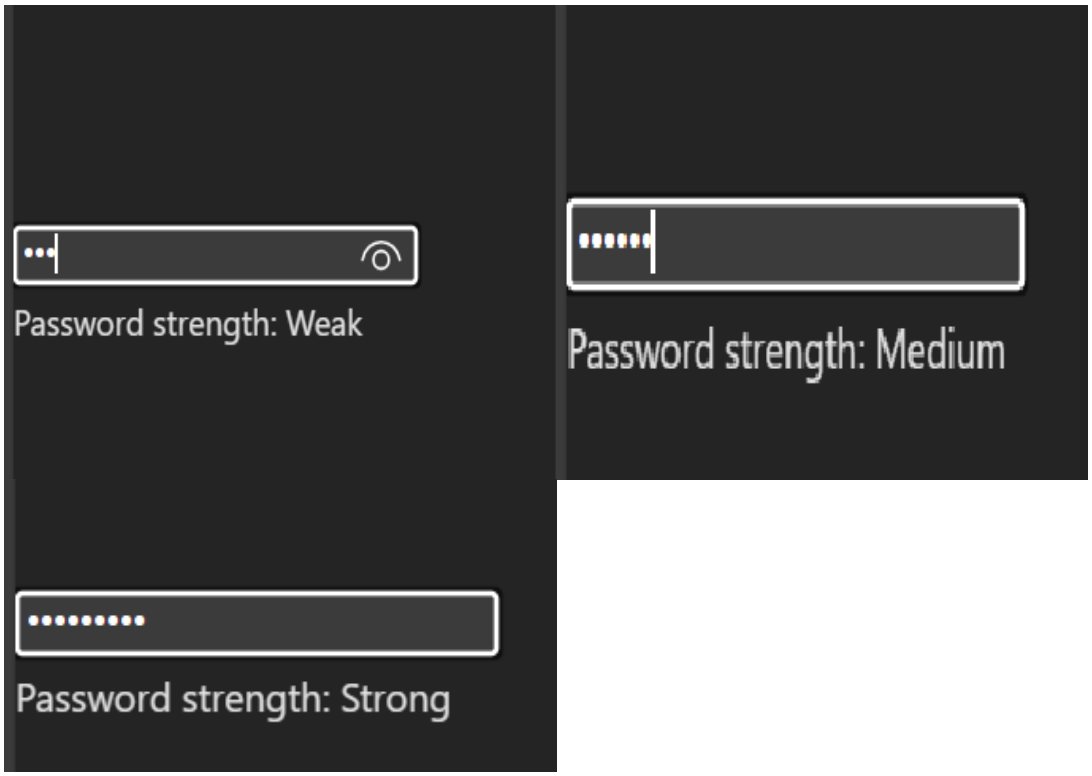
STEP 8: Stop the program.

PROGRAM:

```
 import React, { useState } from 'react';
// import './App.css'; // Import CSS file for styling

function PasswordStrengthIndicator() {
  const [password, setPassword] = us estate("");
  const [strength, setStrength] = useState("");

  // Function to evaluate password strength
  function evaluatePasswordStrength(password) {
  let score = 0;

    if (!password) return '';

    // Check password length
    if (password.length > 8) score += 1;
    // Contains lowercase
    if (/[a-z]/.test(password)) score += 1;
    // Contains uppercase
    if (/[A-Z]/.test(password)) score += 1;
    // Contains numbers
    if (/\d/.test(password)) score += 1;
    // Contains special characters
    if (/[^A-Za-z0-9]/.test(password)) score += 1;

    switch (score) {
      case 0:
      case 1:
      case 2:
        return "Weak";
      case 3:
        return "Medium";
      case 4:
      case 5:
        return "Strong";
      default:
        return "";
    }
  }

  return (
    <div className="password-container">
      <input
        type="password"
        placeholder="Enter your password"
        value={password}
        onChange={(event) => {
        setPassword(event.target.value);
          setStrength(evaluatePasswordStrength(event.target.value));
        }}
```

```
      />
      <div className="strength-indicator">
        <div className={`strength ${strength.toLowerCase()}`}></div>
        <small>Password strength: {strength}</small>
      </div>
    </div>
  );
}

export default PasswordStrengthIndicator;
```

OUTPUT:



RESULT:

Ex.No: Date:	**Write a program to create a simple Login form Application using React JS**

AIM:

The aim of this React application is to create a simple login form that takes an email and password input from the user .

ALGORITHM:

STEP 1: Start the program.

STEP 2: Set up a new React project using Create React App or any other method youprefer.

STEP 3: Create a functional component called App, which will serve as the maincomponent for your login form application.

STEP 4: Use the React useState hook to manage the state of the username andpassword input fields.

STEP 5: Style the login form in the App.css file to enhance its appearance and layout.

STEP 6: Apply the styles to the login form components in the App.jsx file.

STEP 7: Import the App.css file into the App.jsx file and apply the styles to the loginform components.

STEP 8: Stop the program.

PROGRAM:

App.jsx

```
import React, { useState } from 'react';import
'./App.css';
const App = () => {
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');const
   handleEmailChange = (e) => {
```

```
 setEmail(e.target.value);
};
const handlePasswordChange = (e) => {
  setPassword(e.target.value);
};
const handleSubmit = (e) => {
  e.preventDefault();
  console.log(`E-mail: ${email}, Password: ${password}`);
};
 return (
  <div className="container">
    <form onSubmit={handleSubmit} className="form">
      <h1>Login Form</h1>
      <label htmlFor="email">E-mail:</label>
      <input
        type="text"
        id="email"
        value={email}
        onChange={handleEmailChange}
        required
      />
    <label htmlFor="password">Password:</label>
      <input
        type="password"
        id="password"
        value={password}
        onChange={handlePasswordChange}required
      />
    <button type="submit">Submit</button>
    </form>
  </div>
```

```
  );
};

export default App;
```

App.Css

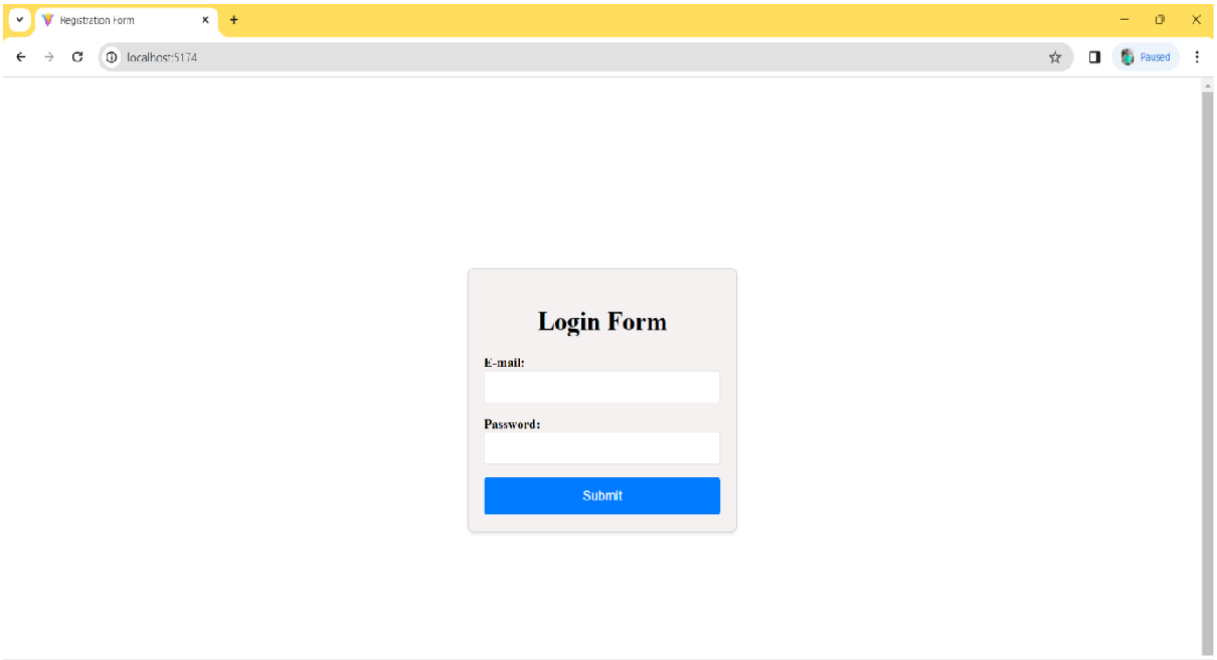
```
.container { display:
  flex;
  justify-content: center;align-
  items: center; height: 100vh;
}
.form {
  width: 300px;
  padding: 20px;
  border: 1px solid #ccc; border-
  radius: 8px;
  background-color: #f5f1f1;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

}
h1 {
  text-align: center; margin-
  bottom: 20px;
}

label {
  font-weight: bold;

}

input {
  width: 100%; padding: 10px;
  margin-bottom: 15px;
```

```
  border: 1px solid #ddd;border-
  radius: 4px;
  box-sizing: border-box;

}

button  { width:
  100%;
  padding: 12px; border:
  none; border-radius: 4px;
  background-color: #007bff;color:
  #fff;
  font-size: 16px;
  cursor: pointer;
}
button:hover {
  background-color: #0056b3;

}
```

main.jsx

```
import React from 'react'
import ReactDOM from 'react-dom/client'import
App from './App.jsx'
ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
)
```

OUTPUT:



RESULT:

Ex.No: Date:	**Write a program to Create a project on Grocery Delivery Application**

AIM:

To create a grocery delivery application using reactjs.

ALGORITHM:

STEP 1: Start the process.

STEP 2: Open a new directory and create two sub directories with the name server and client.

STEP 3: Open new terminal in server and create a file and enter server code.

STEP 4: Enter npm init -y and npm install express mongoose cors.

STEP 5: Start the server using the command node server.js

STEP 6: Open terminal in client directory and enter the command npx create-react-app client.

STEP 7: Enter the code in App.js, ItemContext.js, Header.js, ProductItem.js, ProductList.js and App.css files.

STEP 8: Save the code and run the code using npm start command.

STEP 9: The grocery delivery application is displayed.

PROGRAM:

// client/src/components/Header.js

```
import React, { useContext } from 'react';
import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
import { faCartShopping } from '@fortawesome/free-solid-svg-icons'
import { itemContext } from '../context/ItemContext';

const Header = () => {

	const { itemsInCart, totalPrice } = useContext(itemContext)

	return (
		<div className='header' >
			<h1 className='gfg'>
				Grocery application
			</h1>
			<h3 style={{ color: "green" }}>
				Total Price: {totalPrice}
			</h3>
			<div className='cart-num'>
				<div className='cart-items'>
					{itemsInCart}
```

```
                        </div>
                        <FontAwesomeIcon icon={faCartShopping} size="4x" />
                    </div>
                </div>
        );
};

export default Header;
```

// client/src/components/ProductItem.js

```
import React, { useContext } from 'react';
import { itemContext } from '../context/ItemContext';


const ProductItem = ({ product }) => {
        const { addToCart, removeFromCart } = useContext(itemContext)
        const handleAddToCart = (product) => {
                console.log(product)
                addToCart(product)

        };
        const handleRemoveToCart = (product) => {
                console.log("product removed", product)
                removeFromCart(product)

        };
        return (
                <div className="product-card">
                        <img className="product-image"
                                src={product.image}
                                alt={product.name} />
                        <div className="product-details">
                                <h3 style={{ fontWeight: "700" }}>
                                        {product.name}
                                </h3>
                                <p style={{ fontWeight: "300" }}>
                                        {product.description}
                                </p>
                                <p style={{ fontWeight: "500" }}>
                                        Price: {product.price} Rs/Kg
                                </p>
                                <button onClick={
                                        () => handleAddToCart(product)
                                }>
                                        Add to Cart
                                </button>
                                <button onClick={
                                        () =>
                                                handleRemoveToCart(product)
                                }>
```

```
                                        -
                                </button>
                        </div>
                </div>
        );
};

export default ProductItem;
```

// client/src/components/ProductList.js

```
import React, { useContext, useEffect, useState } from 'react';
import ProductItem from './ProductItem';
import { itemContext } from '../context/ItemContext';

const ProductList = () => {
        const { products } = useContext(itemContext);
        // Keep a local state for sorted products
        const [sortedProducts, setSortedProducts] =
                useState([...products]);
        const [minPrice, setMinPrice] = useState(0);
        const [maxPrice, setMaxPrice] = useState(3000);
        // 'all' represents no type filter
        const [selectedType, setSelectedType] = useState('all');

        useEffect(() => {
                setSortedProducts([...products])
        }, [products])

        const handleSortByPrice = () => {
                const sorted = [...sortedProducts]
                        .sort((a, b) => a.price - b.price);
                setSortedProducts(sorted);
        };

        const handleFilterByPriceRange = () => {
                const filtered =
                        products.filter(
                                (product) =>
                                        product.price >= minPrice &&
                                        product.price <= maxPrice);
                setSortedProducts(filtered);
        };

        const handleFilterByType = () => {
                if (selectedType === 'all') {
                        // Reset the type filter
                        setSortedProducts([...products]);
                } else {
                        const filtered =
                                products.filter(
```

```
                        (product) =>
                                product.type === selectedType);
                setSortedProducts(filtered);
        }
};

return (
        <div className='prdt-list'>
                <h2>Product List</h2>
                <div className='filter-btn'>
                        <button onClick={handleSortByPrice}>
                                Sort by Price
                        </button>
                        <label>
                                Min Price:
                                <input type='number' value={minPrice}
                                        onChange={
                                                (e) =>

setMinPrice(Number(e.target.value))
                                        } />
                        </label>
                        <label>
                                Max Price:
                                <input type='number' value={maxPrice}
                                        onChange={
                                                (e) =>

setMaxPrice(Number(e.target.value))
                                        } />
                        </label>
                        <button onClick={() => handleFilterByPriceRange()}>
                                Filter by Price Range
                        </button>
                        <label>
                                Filter by Type:
                                <select value={selectedType}
                                        onChange={
                                                (e) =>
                                                        setSelectedType(e.target.value)
                                        }>
                                        <option value='all'>
                                                All
                                        </option>
                                        <option value='Fruit'>Fruit</option>
                                        <option value='Vegetable'>Vegetable</option>
                                </select>
                        </label>

                        <button onClick={handleFilterByType}>
                                Filter by Type
```

```
                        </button>
                    </div>

                    <ul className='item-card'>
                        {sortedProducts.map((product) => (
                            <ProductItem key={product._id}
                                product={product} />
                        ))}
                    </ul>
                    <div className='buy-now-btn'>Buy Now</div>
                </div>
        );
};

export default ProductList;
```

//context/ItemContext.js

```
import {
        createContext,
        useEffect,
        useState
} from 'react';

const itemContext = createContext();

// creating custom provider
function CustomItemContext({ children }) {
        const [products, setProducts] = useState([]);
        const [cart, setCart] = useState([]);
        const [itemsInCart, setItemsInCart] = useState(0);
        const [totalPrice, setTotalPrice] = useState(0)

        // useEffect to load all the vegetables
        useEffect(() => {
                // Fetch products from the backend and dispatch 'SET_PRODUCTS' action
                const fetchData = async () => {
                        const response =
                            await fetch('http://localhost:5000/api/products');
                        const products = await response.json();
                        // console.log(products)
                        setProducts(products);
                };

                fetchData();
        }, []);

        const addToCart = (product) => {
                setTotalPrice(totalPrice + product.price)
                setCart([...cart, product]);
                setItemsInCart(itemsInCart + 1);
```

```
	};

	const removeFromCart = (product) => {
		const index =
			cart.findIndex(
				(prdt) =>
					prdt._id === product._id);
		console.log(index);

		if (index !== -1) {
			// Item found in the cart
			// Now you can remove it from the cart array
			const updatedCart = [...cart];
			updatedCart.splice(index, 1);
			setTotalPrice(totalPrice - cart[index].price);
			setCart(updatedCart);
			setItemsInCart(itemsInCart - 1);
		} else {
			console.log("Item not found in the cart");
		}
	};

	return (
		// default provider
		<itemContext.Provider value={
			{
				products, addToCart,
				removeFromCart,
				itemsInCart, totalPrice
			}}>
			{children}
		</itemContext.Provider>
	);
}

export { itemContext };
export default CustomItemContext;

// server.js
const express = require('express');
const mongoose = require('mongoose');
const app = express();
const PORT = process.env.PORT || 5000;
const cors = require('cors');

mongoose.connect('mongodb://localhost/fruitvegmarke',
{
	useNewUrlParser: true,
	useUnifiedTopology: true
}
);
```

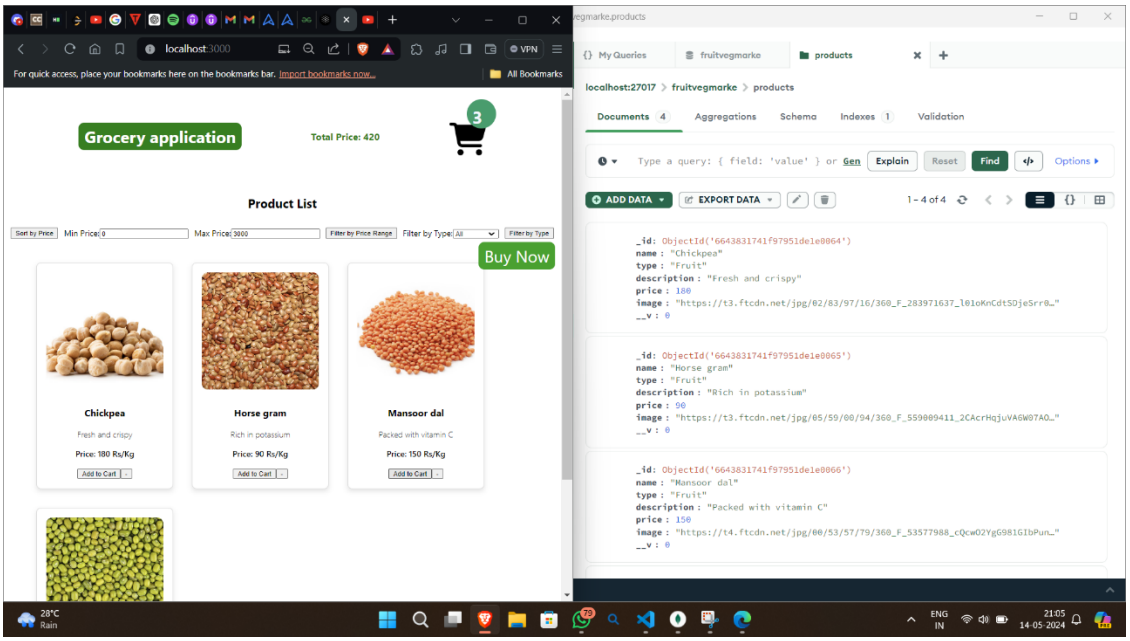
```
app.use(express.json());
app.use(cors()); // Use the cors middleware

const productSchema = new mongoose.Schema({
name: String,
type: String,
description: String,
price: Number,
image: String,
});

const Product = mongoose.model('Product', productSchema);

// Function to seed initial data into the database
const seedDatabase = async () => {
try {
	await Product.deleteMany(); // Clear existing data

	const products = [
	{
		name: 'Chickpea', type: 'Fruit',
		description: 'Fresh and crispy',
		price: 180,
		image:
'https://t3.ftcdn.net/jpg/02/83/97/16/360_F_283971637_l01oKnCdtSDjeSrr0HzsK35wQtx91CNc.jpg '
	},
	{
		name: 'Horse gram',
		type: 'Fruit',
		description: 'Rich in potassium',
		price: 90,
		image:
'https://t3.ftcdn.net/jpg/05/59/00/94/360_F_559009411_2CAcrHqjuVA6W07AOgxyEmpWelBBSeWR.jpg '
	},
	{
		name: 'Mansoor dal',
		type: 'Fruit',
		description: 'Packed with vitamin C',
		price: 150,
		image:
'https://t4.ftcdn.net/jpg/00/53/57/79/360_F_53577988_cQcwO2YgG981GIbPunnKYekK9WxjLWtn.jpg '
	},
	{
		name: 'Greengram',
		type: 'Vegetable',
		description: 'Healthy and crunchy',
		price: 80,
```

```
                image:
'https://t4.ftcdn.net/jpg/03/23/92/35/360_F_323923529_TkDiqEOJWQRTdJrGMMnxRb1Zq1I
bPc4j.jpg '
        },

        ];

        await Product.insertMany(products);
        console.log('Database seeded successfully');
} catch (error) {
        console.error('Error seeding database:', error);
}
};

// Seed the database on server startup
seedDatabase();

// Define API endpoint for fetching all products
app.get('/api/products', async (req, res) => {
try {
        // Fetch all products from the database
        const allProducts = await Product.find();

        // Send the entire products array as JSON response
        res.json(allProducts);
} catch (error) {
        console.error(error);
        res.status(500)
        .json({ error: 'Internal Server Error' });
}
});

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

/*App.css*/
.cart-items {
        border-radius: 50%;
        background-color: rgb(20, 158, 105);
        font-weight: 700;
        color: aliceblue;
        width: 30px;
        height: 30px;
        font-size: 30px;
        padding: 10px;
        top: 10px;
        position: relative;
        left: 30px;
```

```
}

.header {
	display: flex;
	justify-content: space-evenly;
	align-items: center;
	padding: 10px;
	border-bottom: 1px sold #ccc;

}


/* card */
/* client/src/components/ProductItem.css */
.product-card {
	border: 1px solid #ddd;
	border-radius: 8px;
	width: fit-content;
	padding: 16px;
	margin: 16px;
	box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
	background-color: #fff;
	display: flex;
	flex-direction: column;
	align-items: center;
}

.product-image {
	width: 200px;
	height: 200px;
	object-fit: cover;
	border-radius: 10px;
	margin-bottom: 12px;
	transition: transform 0.3s ease-in-out;
}

.product-image:hover {
	transform: scale(1.1);
	/* Enlarge the image on hover */
}

.product-details {
	text-align: center;
}


.item-card {
	display: flex;
	flex-wrap: wrap;
}
```

```
h2 {
	text-align: center;
}

.filter-btn {
	display: flex;
	flex-direction: row;
	padding: 10px;
	gap: 10px;
	justify-content: center;
}

.prdt-list {
	display: flex;
	flex-direction: column;
	justify-content: center;
}

.cart-num {
	margin-bottom: 40px;
	cursor: pointer;
}

.buy-now-btn {
	background-color: rgb(11, 162, 11);
	color: white;
	padding: 5px 10px;
	border-radius: 10px;
	font-size: 2rem;
	position: fixed;
	top: 30%;
	right: 10px;
	cursor: pointer;
}

.buy-now-btn:hover {
	background-color: rgb(113, 230, 113);
	color: brown;
}

.gfg {
	background-color: green;
	color: white;
	padding: 5px 10px;
	border-radius: 10px;
}
```

OUTPUT:



RESULT:

Ex.No: Date:	**Write a program to connect our TO-DO ReactJS Project with mongoDB**

AIM:

To write a program to create a TODO React js Project with mongodb.

ALGORITHM:

STEP 1:Start the process

STEP 2:Using NPM to set up the project as To do List.

STEP 3:Navigate to the Project Directory to the folder.

STEP 4:Using NPM to start the development server in the folder.

STEP 5:Create the TodoList Component which is responsible for managing the list of tasks and

handling task-related functionality.

STEP 6::To enhance the visual appeal of your Todo List, you can apply styling in it.

STEP 7:By using NPM to deploy the application to display it

STEP 8:Stop the process.

PROGRAM:

Home.jsx

```
import React, { useState, useEffect } from 'react';
import axios from 'axios'; // Import axios
import { BsCircleFill, BsFillCheckCircleFill, BsFillTrashFill } from 'react-icons/bs'; //
Assuming you've imported these icons
import Create from './Create';
import './App.css';
function Home() {
const [todos, setTodos] = useState([]);
useEffect(() => {
axios.get('http://localhost:3001/get')
.then(result => setTodos(result.data))
.catch(err => console.log(err));
```

```
}, []);
const handleEdit = (id) => { // Pass id as parameter
axios.put('http://localhost:3001/update/' + id)
.then(result => {
window.location.reload(); // Use window.location.reload() instead of location.reload()
})
.catch(err => console.log(err));
};
    const handleDelete = (id) => { // Pass id as parameter
        axios.delete('http://localhost:3001/delete/' + id)
            .then(result => {
                window.location.reload(); // Use window.location.reload() instead of
location.reload()
            })
            .catch(err => console.log(err));
    };

    return (
        <div className='home'>
            <h2>TO DO LIST</h2>
            <Create />
            <br />
            {
                todos.length > 0 // Check if todos is not empty before mapping
                    ? todos.map(todo => (
                        <div className='task' key={todo._id}>
                            <div className='checkbox' onClick={() => handleEdit(todo._id)}> {/*
Corrected onClick */}
                                {todo.done ? <BsFillCheckCircleFill /> : <BsCircleFill className='icon'
/>}
                                <p className={todo.done ? "line_through" : ""}>{todo.task}</p>
                            </div>
                            <div>
                                <span><BsFillTrashFill className='icon' onClick={() =>
```

```
handleDelete(todo._id)} /></span> {/* Corrected onClick */}
                    </div>
                </div>
            ))
            : <p>No todos found</p>
        }
    </div>
  );
}

export default Home;
```

MAIN.JSX

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

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

Create.JSX

```
import React,{ useState} from 'react'
import axios from 'axios'
import './App.css'
function Create() {
const [task, setTask] = useState()
const handleAdd = () => {
axios.post('http://localhost:3001/add', {task: task})
```

```
.then(result => {
location.reload()
})
.catch(err => console.log(err))
}
return (
<div className='create_form'>
<input type="text" name="" id="" placeholder='Enter yor Task' onChange={(e) =>
setTask(e.target.value)} />
<button type='button' onClick={handleAdd}>Add</button>
</div>
)
}
export default CreateApp.CSS
.home{
display: flex;
flex-direction: column;
align-items: center;
}
.create_form input{
width: 300px;
padding: 10px;
border-bottom: 2px solid;
outline: none;
}
.create_form button{
padding: 10px;
background-color: rgb(0, 0, 0);
color: rgb(255, 255, 255);
cursor: pointer;
}
.task{
display: flex;
align-items: center;
```

```
width: 345px;
justify-content: space-between;
background-color: rgb(218, 208, 208);
color: whitw;
padding: 2px 5px 2px 5px;
margin-top: 2px;
}
.checkbox .icon{
margin-right: 5px;
font-size: 15px;
}
.line_through{
text-decoration: line-through;
}
.checkbox{
display: flex;
align-items: center;
}
.task div span{
margin: 0px 5px 0px 4px
}
.task div .icon {
cursor: pointer;
}
```

SERVER/Todo.JS

```
const mongoose = require('mongoose')
const TodoSchema = new mongoose.Schema({
task: String,
done: {
type: Boolean,
default: false
}
})
const TodoModel = mongoose.model("todos", TodoSchema)
```

```
module.exports = TodoModel
```

SERVER/index.js

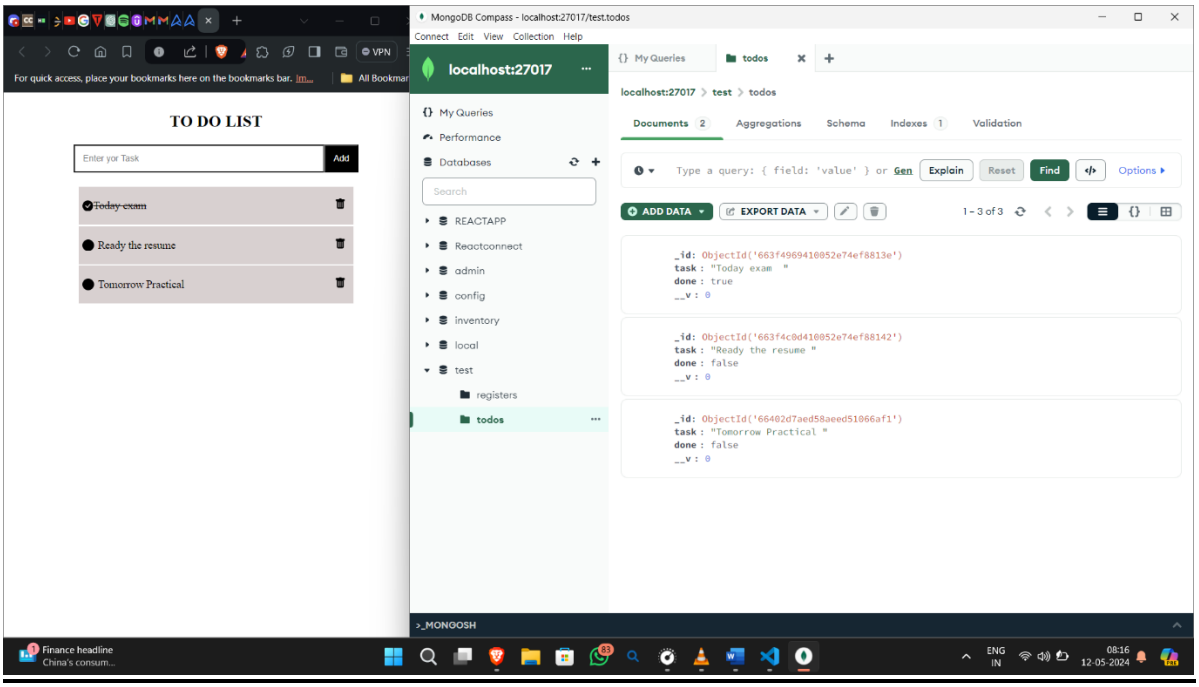
```
const express = require('express')
const mongoose = require('mongoose')
const cors = require('cors')
const TodoModel= require('./Models/Todo')
const app = express()
app.use(cors())
app.use(express.json())
mongoose.connect('mongodb://127.0.0.1:27017/test')
app.get('/get', (req,res) => {
TodoModel.find()
.then(result => res.json(result))
.catch(err => res.json(err))
})
app.put('/update/:id', (req,res) => {
const {id} = req.params;
TodoModel.findByIdAndUpdate({_id: id}, {done: true})
.then(esult => res.json(result))
.catch(err => res.json(err))
})
app.delete('/delete/:id', (req,res) => {
const {id} = req.params;
TodoModel.findByIdAndDelete({_id: id})
.then(esult => res.json(result))
.catch(err => res.json(err))
})
app.post('/add', (req,res) => {
const task = req.body.task;
TodoModel.create({
task: task
}).then(result => res.json(result))
.catch(err => res.json(err))
})
```

```
app.listen(3001, () => {
console.log("Server is running")
})
```

Package.json

```
{
"name": "server",
"version": "1.0.0",
"description": "",
"main": "index.js",
"scripts": {
"test": "echo \"Error: no test specified\" && exit 1",
"start": "nodemon index.js"
},
"keywords": [],
"author": "",
"license": "ISC",
"dependencies": {
"cors": "^2.8.5",
"express": "^4.19.2",
"mongoose": "^8.3.4",
"nodemon": "^3.1.0"
}
}
```

OUTPUT:



RESULT: