Ex. No.: 08	Making The Application use right database
Date : 11/09/2024	

Aim:

To create a React application that allows users to enter their name and email, and submit this information to a backend API. The application demonstrates the use of form handling, state management, and Axios for HTTP requests in React.

Procedure:

1. Setup Environment:

- Ensure Node.js and npm are installed on your system.
- Create a new React application using Create React App:

  bash

  ```
  npx create-react-app user-info-submission
  cd user-info-submission
  ```

- Install Axios for handling HTTP requests:

  bash

  ```
  npm install axios
  ```

2. Set up the Backend:

- Ensure you have a backend server running on http://localhost:5500 with an endpoint POST /api/submit to handle data submission.
- The server should be able to receive JSON data and handle requests from your frontend.

3. Create the App Component:

- Replace the code in src/App.js with the provided code for the user submission form.

4. Define State Variables:

- Use the useState hook to create two state variables:
- name for storing the user's name.
- email for storing the user's email address.
- Each state variable is initialized with an empty string ('').

5. Handle Form Submission with Axios:

- Define an asynchronous handleSubmit function:

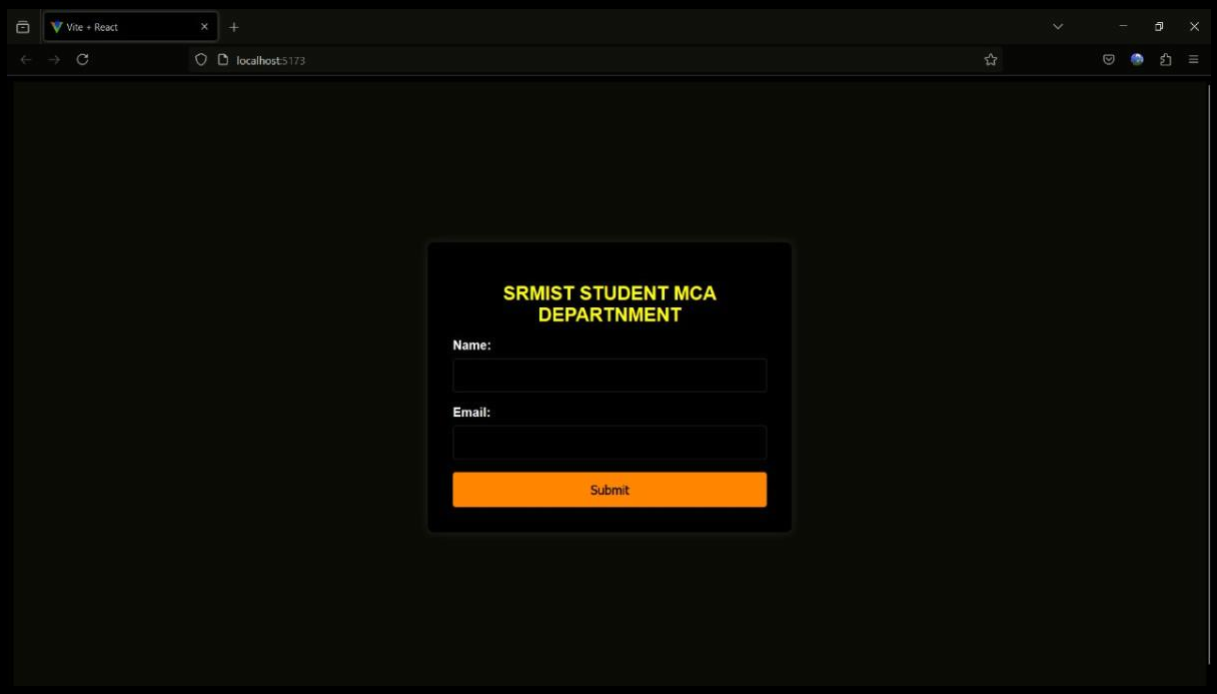
- Prevent the page reload using e.preventDefault().
- Use Axios to make a POST request to the backend at http://localhost:5500/api/submit.
- Send the name and email in the request body.
- Handle successful responses by displaying a success message and logging the response data.
- Handle errors by logging the error to the console and displaying an alert for failure.

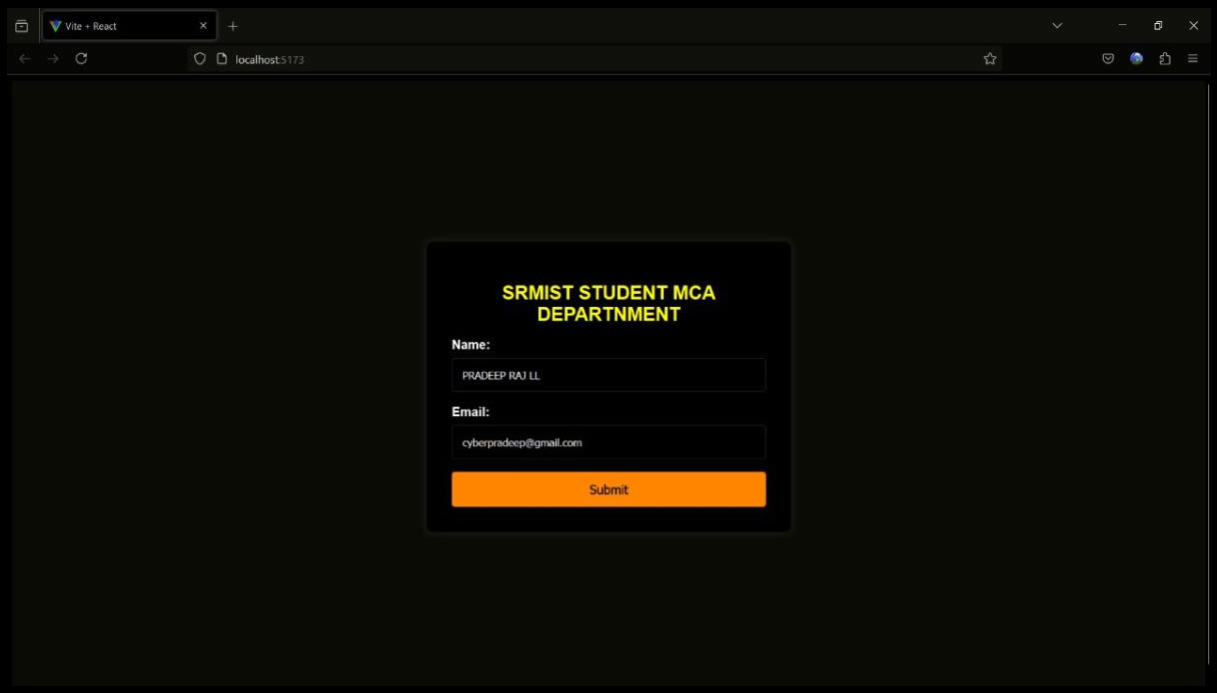
SOURCE CODE:

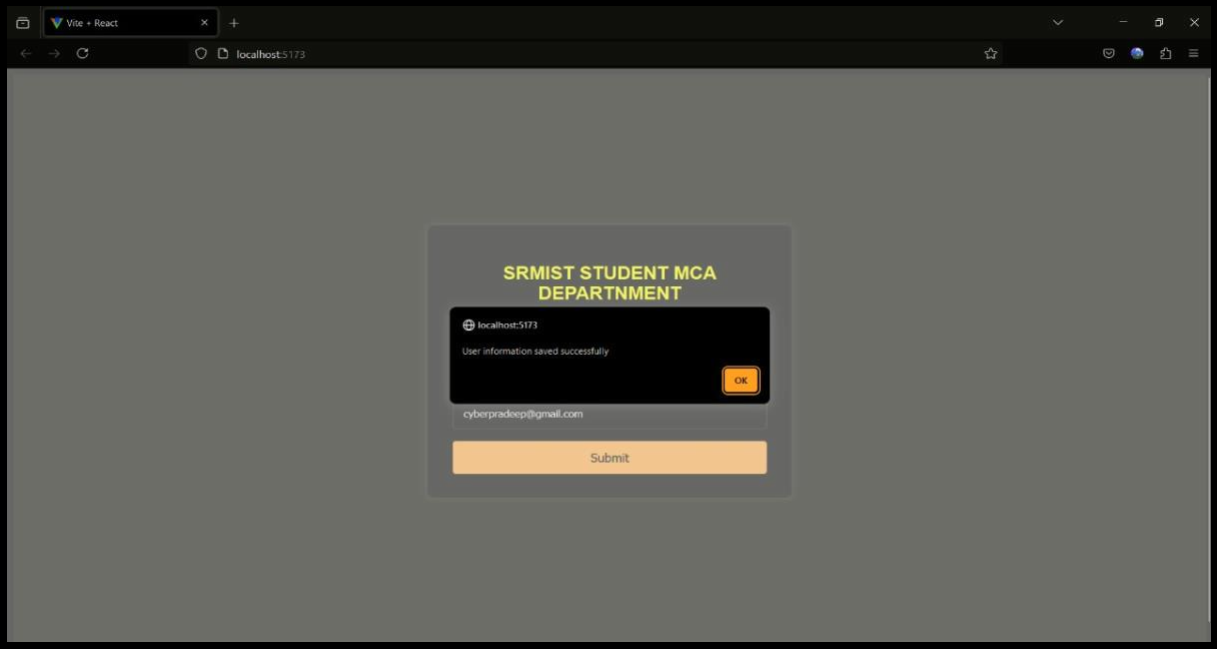
```
import { useState } from 'react';
import axios from 'axios';
import './App.css';

function App() {
  const [name, setName] = useState('');
  const [email, setEmail] = useState('');

  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      const response = await axios.post('http://localhost:5500/api/submit', {
        name,
        email,
      });
      console.log(response.data);
      alert('User information saved successfully');
    } catch (error) {
      console.error('There was an error!', error);
      alert('Failed to save user information');
    }
  };

  return (
    <div className="App">
      <div className="form-container">
```

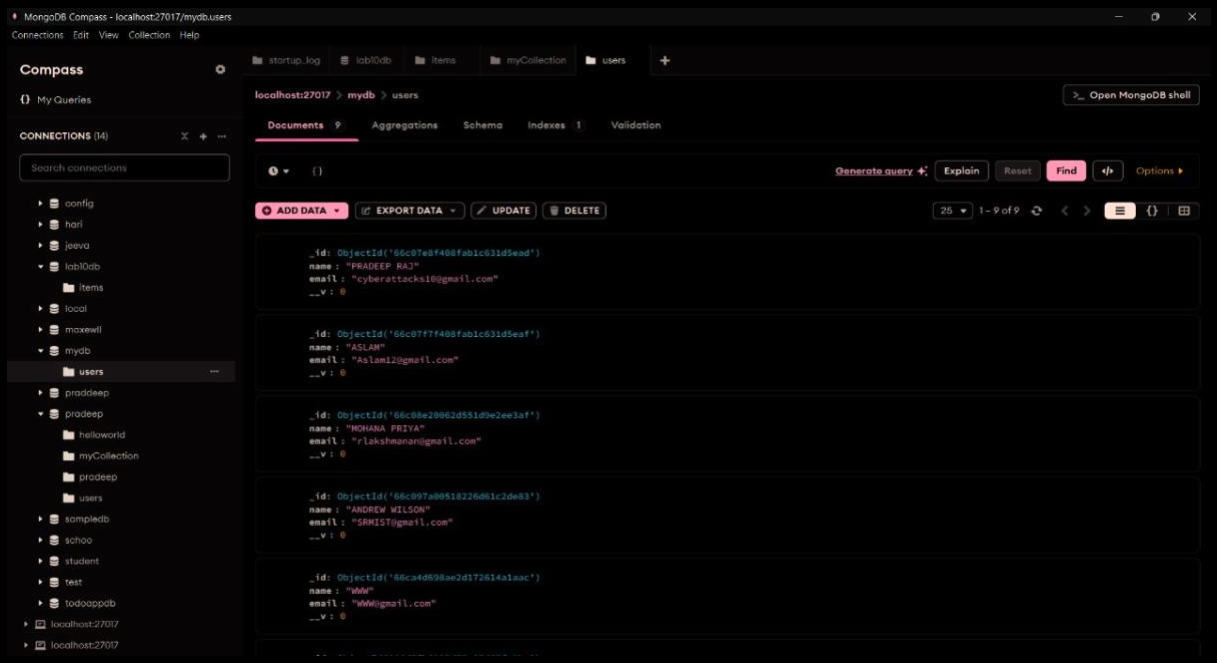
```
      <h2 style={{color:'blue'}}>SRMIST STUDENT MCA DEPARTNMENT</h2>
      <form onSubmit={handleSubmit}>
        <div className="form-group">
          <label>Name:</label>
          <input
            type="text"
            value={name}
            onChange={(e) => setName(e.target.value)}
            required
          />
        </div>
        <div className="form-group">
          <label>Email:</label>
          <input
            type="email"
            value={email}
            onChange={(e) => setEmail(e.target.value)}
            required
          />
        </div>
        <button type="submit">Submit</button>
      </form>
    </div>
  </div>
  );
}

export default App;
```

OUTPUT:


Vite + React
localhost:5173
SRMIST STUDENT MCA DEPARTNMENT
Name:
Email:
Submit


Vite + React
localhost:5173
SRMIST STUDENT MCA DEPARTNMENT
Name:
PRADEEP RAJ LL
Email:
cyberpradeep@gmail.com
Submit





RESULT:

User Information Submission Application collects and submits the user's name and email address to the backend. Upon submitting the form:

- The application sends a POST request to the backend API with the collected information.
- If the request is successful, a confirmation message is displayed, and the console logs the server response.
- If the request fails, an error message is displayed, indicating that the submission was unsuccessful.