

Lesson 07 Demo 04

React Libraries

Objective: To develop a React application that demonstrates React libraries

Tools Required: Node Terminal, React App, and Visual Studio Code

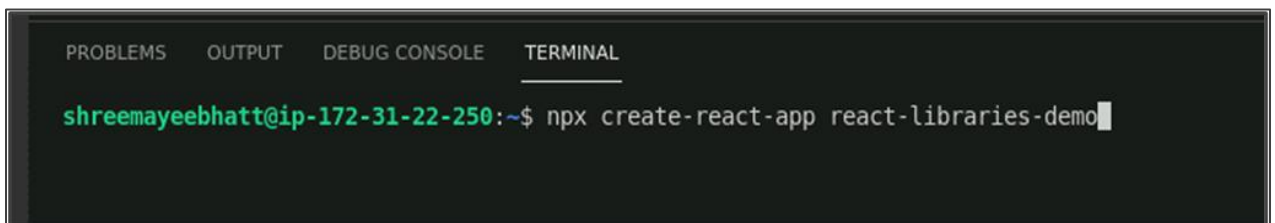
Prerequisites: Knowledge of creating a React app and understanding of the folder structure

Steps to be followed:

1. Create a new React app
2. Install the React libraries and import them into **App.js**
3. Run the app

Step 1: Create a new React app

1.1 Open the terminal and run the command **npx create-react-app react-libraries-demo**



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
shreemayeebhatt@ip-172-31-22-250:~$ npx create-react-app react-libraries-demo
```

This command will create a new React app with the name **react-libraries-demo**.

1.2 Move to the newly created directory by running the command **cd react-libraries-demo** in the terminal

Step 2: Install the React libraries and import them into App.js

These libraries are **axios** for remote data fetching and **formik** for building performant forms.

- 2.1 In the terminal, inside the project directory, run the command **npm install axios formik** to install the **axios** and **formik** libraries

```
happy hacking:
shreemayeebhatt@ip-172-31-22-250:~$ cd react-libraries-demo/
shreemayeebhatt@ip-172-31-22-250:~/react-libraries-demo$ npm install axios formik
```

You can create a simple form that uses **formik** to manage its state and validation and **axios** to submit data to a remote server.

- 2.2 Open your React project with Visual Studio Code, navigate into the **src** folder, and open the **App.js** file

- 2.3 Import **React**, **Formik**, **Form**, **Field**, **ErrorMessage** from **formik** and **axios**

```
import React from 'react';
import { Formik, Form, Field, ErrorMessage } from 'formik';
import axios from 'axios';
```

- 2.4 Inside the **App** component, define the **handleSubmit** function that will be called when the form is submitted

- 2.5 Use **axios.post** to send a **POST** request to the specified URL, **https://jsonplaceholder.typicode.com/posts** with the form values as the data payload

- 2.6 Use **.then** and **.catch** to handle the response and error, respectively

- 2.7 Within the **handleSubmit** function, **setSubmitting** to false to indicate that the submission process is complete

```
function App() {  
  const handleSubmit = (values, { setSubmitting }) => {  
    axios.post('https://jsonplaceholder.typicode.com/posts', values)  
      .then(response => {  
        console.log(response);  
        setSubmitting(false);  
      })  
      .catch(error => {  
        console.log(error);  
        setSubmitting(false);  
      });  
  };  
}
```

- 2.8 Return a **JSX** expression that represents the structure and content of the app. Use the **div** element as the root container and inside it, use the **Formik** component to wrap the form and provide state management and validation capabilities
- 2.9 Set the **initialValues** prop to an object with empty values for the **title** and **body**
- 2.10 Define the **validate** function to check if the **title** and **body** are empty and return an object with the corresponding error messages if validation fails
- 2.11 Inside the **Formik** component, use the **Form** element to define the form structure, and use the **label** element and the **Field** component for each form input, specifying the type and name attributes
- 2.12 Use the **ErrorMessage** component to display any validation errors
- 2.13 Include a **button** element with the type **submit** to submit the form, and disable it when **isSubmitting** is true
- 2.14 Export the **App** component as the default export of the file

```

return (
  <div>
    <h1>React Libraries Demo</h1>
    <Formik
      initialValues={{ title: '', body: '' }}
      validate={values => {
        const errors = {};
        if (!values.title) {
          errors.title = 'Title is required';
        }
        if (!values.body) {
          errors.body = 'Body is required';
        }
        return errors;
      }}
      onSubmit={handleSubmit}
    >
      {({ isSubmitting }) => (
        <Form>
          <label htmlFor="title">Title</label>
          <Field type="text" name="title" />
          <ErrorMessage name="title" component="div" />

          <label htmlFor="body">Body</label>
          <Field as="textarea" name="body" />
          <ErrorMessage name="body" component="div" />

          <button type="submit" disabled={isSubmitting}>
            Submit
          </button>
        </Form>
      )}
    </Formik>
  </div>
);
}

export default App;

```

Refer to the following code to configure the **App.js** file:

```
import React from 'react';
import { Formik, Form, Field, ErrorMessage } from 'formik';
import axios from 'axios';

function App() {
  const handleSubmit = (values, { setSubmitting }) => {

    axios.post('https://jsonplaceholder.typicode.com/posts', values)

    .then(response => {
      console.log(response);
      setSubmitting(false);
    })
    .catch(error => {
      console.log(error);
      setSubmitting(false);
    });
  };

  return (
    <div>
      <h1>React Libraries Demo</h1>
      <Formik
        initialValues={{ title: '', body: '' }}
        validate={values => {
          const errors = {};
          if (!values.title) {
            errors.title = 'Title is required';
          }
          if (!values.body) {
            errors.body = 'Body is required';
          }
          return errors;
        }}
        onSubmit={handleSubmit}
      >
```

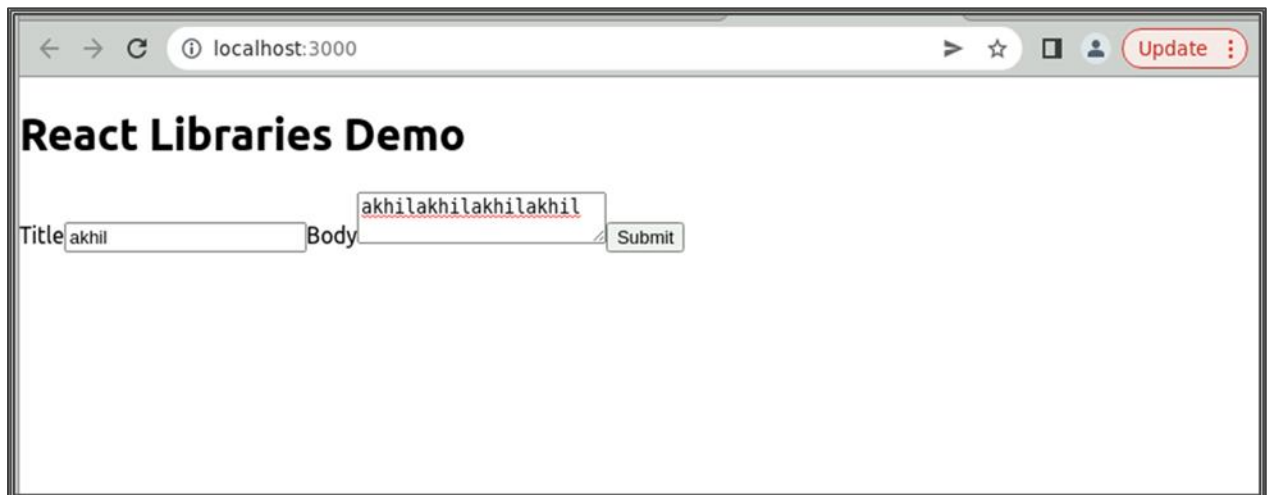
```
{{ isSubmitting }} => (  
  <Form>  
    <label htmlFor="title">Title</label>  
  
    <Field type="text" name="title" />  
    <ErrorMessage name="title" component="div" />  
  
    <label htmlFor="body">Body</label>  
  
    <Field as="textarea" name="body" />  
    <ErrorMessage name="body" component="div" />  
  
    <button type="submit" disabled={isSubmitting}>  
      Submit  
    </button>  
  </Form>  
)}  
</Formik>  
</div>  
);  
}
```

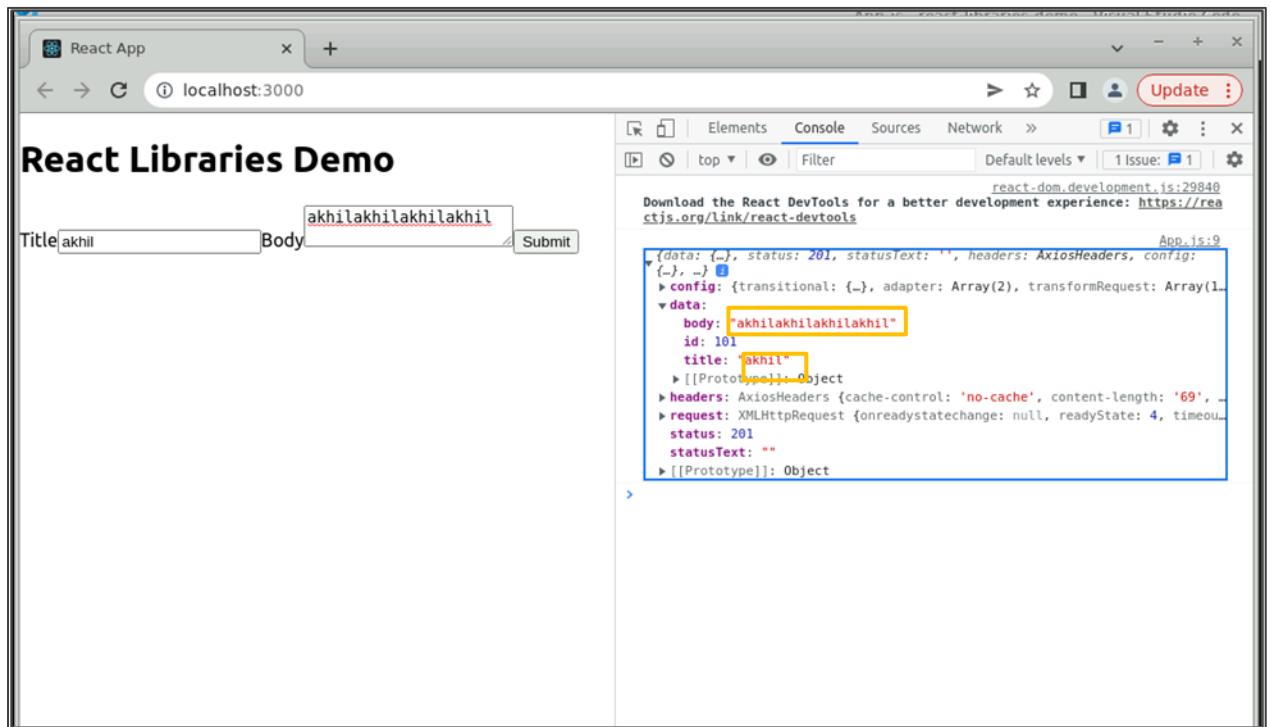
export default App;

Step 3: Run the app

3.1 In the terminal, navigate to the project directory and run the command **npm start** to start the development server

3.2 Open your browser and navigate to **http://localhost:3000**





You should see a simple form that allows you to submit a new post to a remote server using **axios**, with form validation and state management provided by **formik**.

With this, you have successfully created a React application that demonstrates the working of the React libraries – **axios** and **formik**.