16.mailregisterapp

Objectives

- **Explain React Forms validation**
- **Identify the differences between React Form and HTML Form**
- **Explain about controlled components**
- **Identify various React Form input controls**
- **Explain how to handle React Forms**
- **Explain about submitting forms in React**

In this hands-on lab, you will learn how to:

- **Implement React forms validation**
- **Use various input controls like textbox, button and textarea**

Prerequisites

The following is required to complete this hands-on lab:

- **Node.js**
- **NPM**
- **Visual Studio Code**

Notes

Estimated time to complete this lab: 60 minutes.

Create a React App named “mailregisterapp” which will have a component named “register.js”. Create a form which accepts the name, email and password and validate the fields as per the following:

1. **Name should have atleast 5 characters**
2. **Email should have @ and .**
3. **Password should have atleast 8 characters.**

Ensure that validations are implemented through eventhandle and eventsubmit of a form.

CODE:

App.js

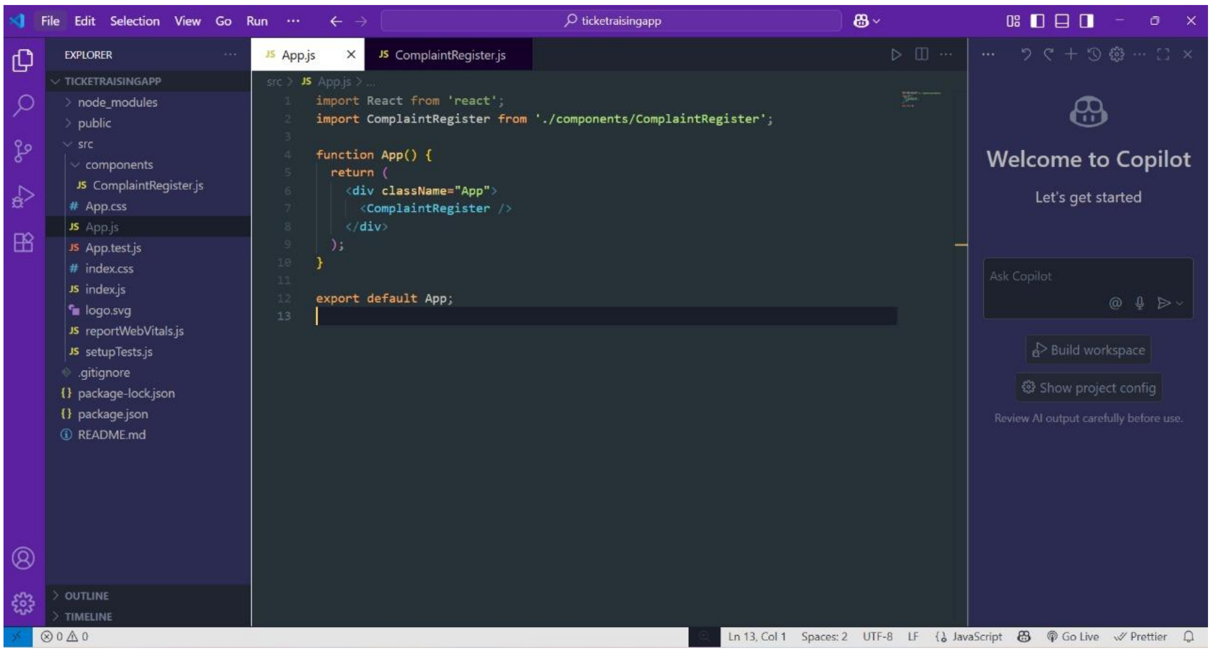
```
import React from 'react';
import Register from './components/Register';

function App() {
  return (
    <div className="App">
      <Register />
    </div>
  );
}

export default App;
```

Register.js

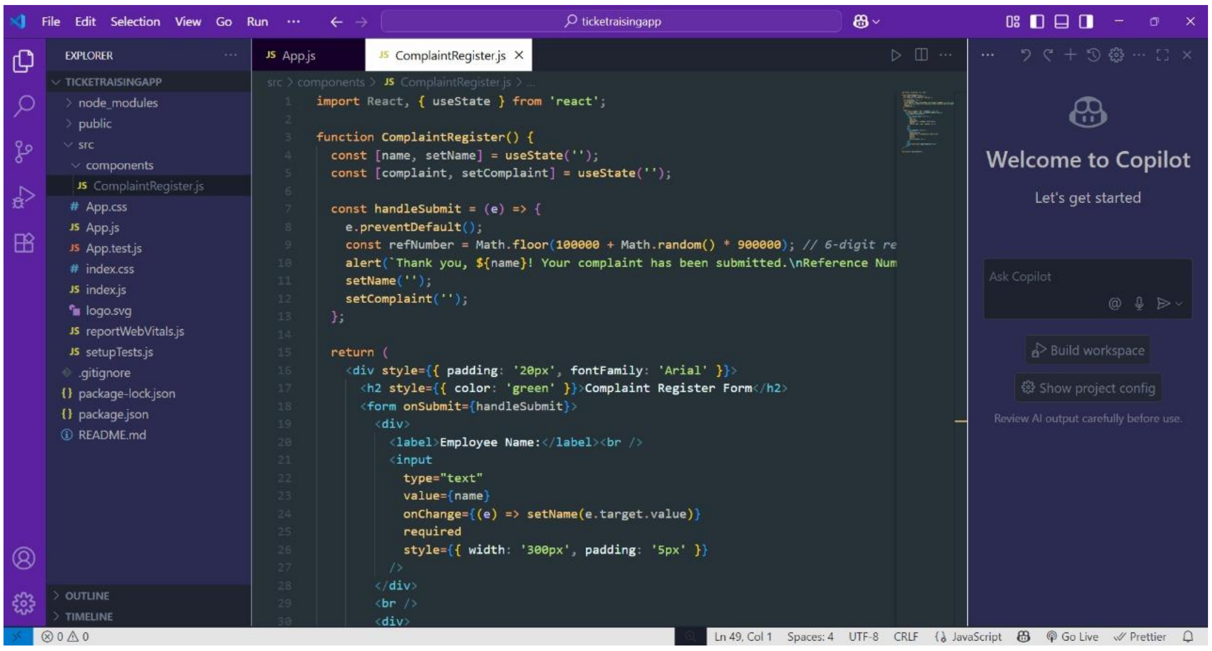
```
import React, { useState } from 'react';

function Register() {
  const [formData, setFormData] = useState({
    name: '',
    email: '',
    password: '',
  });

  const [errors, setErrors] = useState({});

  const validate = () => {
    const tempErrors = {};

    if (formData.name.length < 5) {
      tempErrors.name = 'Name should be at least 5 characters long.';
    }

    if (!formData.email.includes('@') || !formData.email.includes('.')) {
      tempErrors.email = 'Email must contain "@" and ".";
    }

    if (formData.password.length < 8) {
      tempErrors.password = 'Password should be at least 8 characters long.';
    }

    setErrors(tempErrors);
    return Object.keys(tempErrors).length === 0;
  };

  const handleChange = (e) => {
    const { name, value } = e.target;

    setFormData({
      ...formData,
      [name]: value
    });
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    if (validate()) {
      alert(`Registration successful!\nName: ${formData.name}\nEmail:
${formData.email}`);
      setFormData({ name: '', email: '', password: '' });
      setErrors({});
    }
  };
```

```
return (
 <div style={{ padding: '20px', fontFamily: 'Arial' }}>
  <h2 style={{ color: 'navy' }}>Mail Register Form</h2>
  <form onSubmit={handleSubmit}>
   <div>
    <label>Name:</label><br />
    <input
     type="text"
     name="name"
     value={formData.name}
     onChange={handleChange}
     required
    /><br />
    <span style={{ color: 'red' }}>{errors.name}</span>
   </div>
   <br />

   <div>
    <label>Email:</label><br />
    <input
     type="text"
     name="email"
     value={formData.email}
     onChange={handleChange}
     required
```

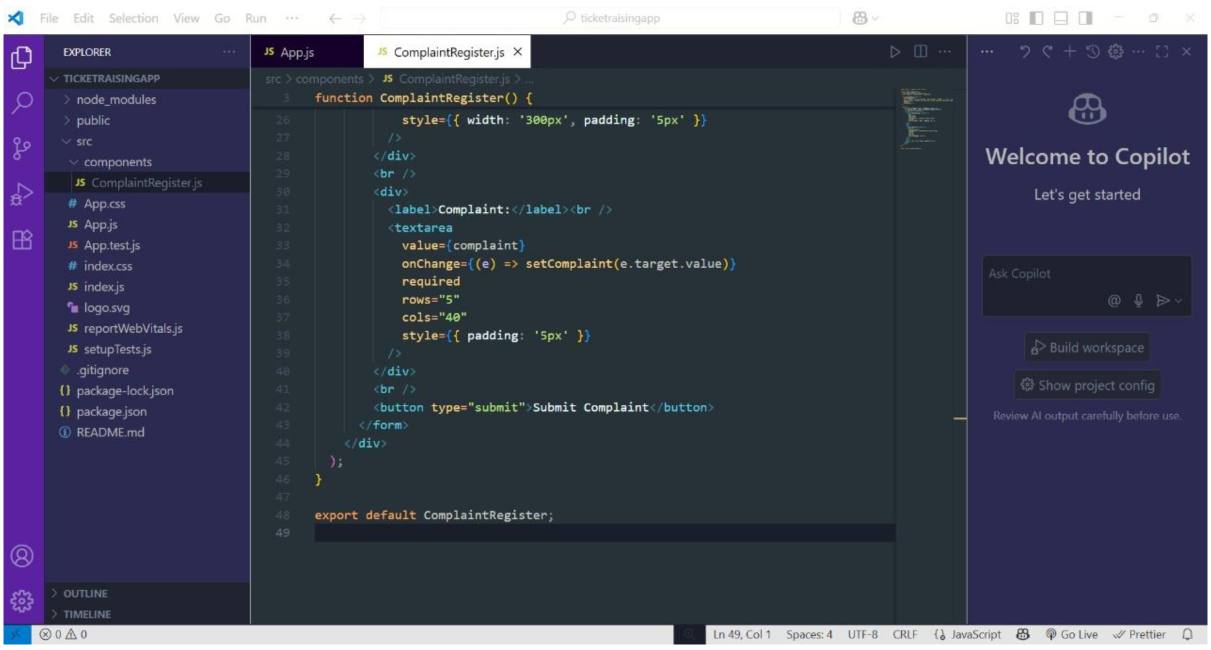
```
          /><br />
          <span style={{ color: 'red' }}>{errors.email}</span>
        </div>
        <br />

        <div>
          <label>Password:</label><br />
          <input
            type="password"
            name="password"
            value={formData.password}
            onChange={handleChange}
            required
          /><br />
          <span style={{ color: 'red' }}>{errors.password}</span>
        </div>
        <br />

        <button type="submit">Register</button>
      </form>
    </div>
  );
}

export default Register;
```



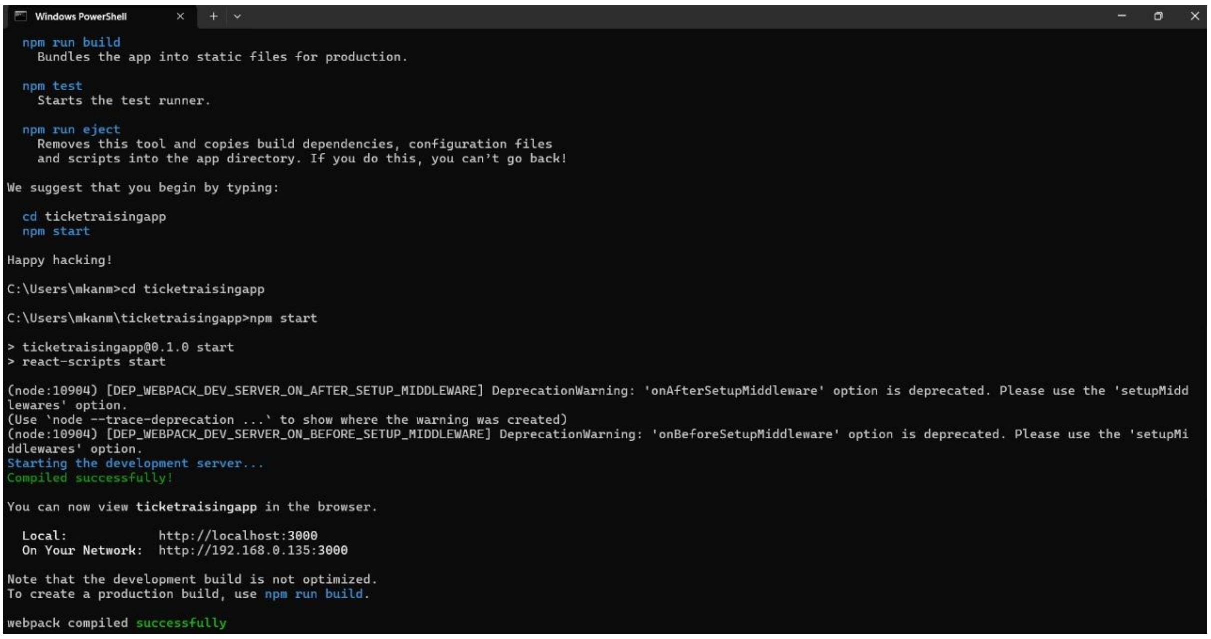
```
import React from 'react';
import ComplaintRegister from './components/ComplaintRegister';

function App() {
  return (
    <div className="App">
      <ComplaintRegister />
    </div>
  );
}

export default App;

```



```
import React, { useState } from 'react';

function ComplaintRegister() {
  const [name, setName] = useState('');
  const [complaint, setComplaint] = useState('');

  const handleSubmit = (e) => {
    e.preventDefault();
    const refNumber = Math.floor(100000 + Math.random() * 900000); // 6-digit re
    alert(`Thank you, ${name}! Your complaint has been submitted.\nReference Num
    setName('');
    setComplaint('');
  };

  return (
    <div style={{ padding: '20px', fontFamily: 'Arial' }}>
      <h2 style={{ color: 'green' }}>Complaint Register Form</h2>
      <form onSubmit={handleSubmit}>
        <div>
          <label>Employee Name:</label><br />
          <input
            type="text"
            value={name}
            onChange={(e) => setName(e.target.value)}
            required
            style={{ width: '300px', padding: '5px' }}
          />
        </div>
        <br />
        <div>
```

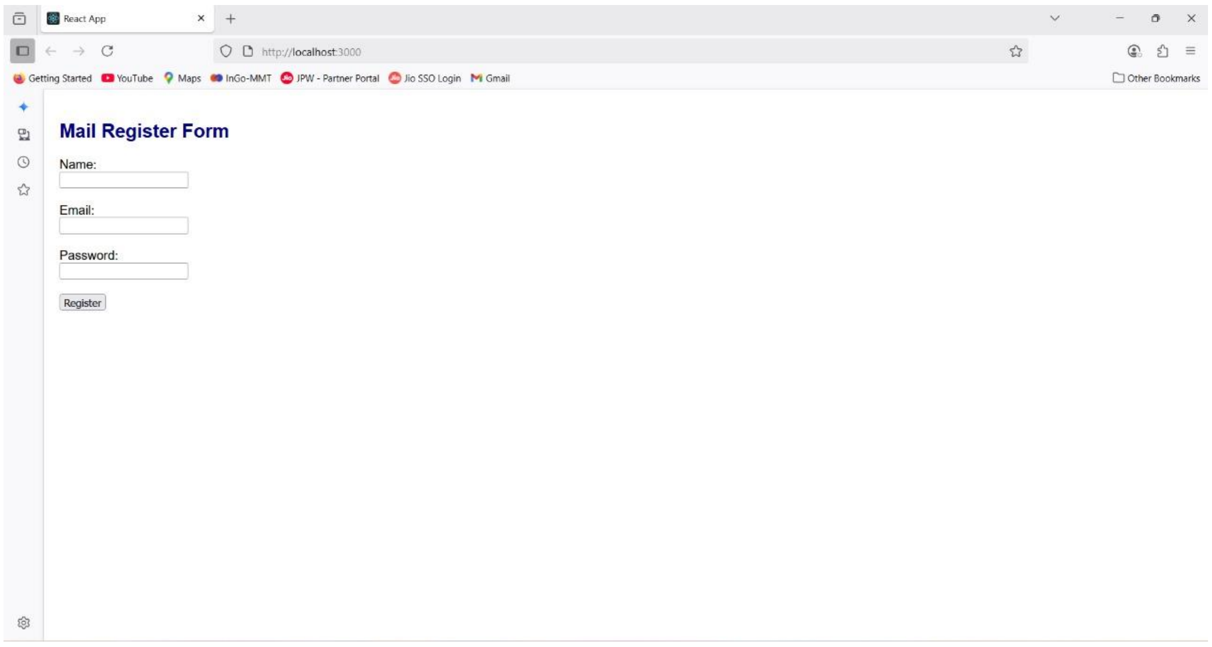


```
function ComplaintRegister() {
          style={{ width: '300px', padding: '5px' }}
        />
      </div>
      <br />
      <div>
        <label>Complaint:</label><br />
        <textarea
          value={complaint}
          onChange={(e) => setComplaint(e.target.value)}
          required
          rows="5"
          cols="40"
          style={{ padding: '5px' }}
        />
      </div>
      <br />
      <button type="submit">Submit Complaint</button>
    </form>
  </div>
);
}

export default ComplaintRegister;
```



```
  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd ticketraisingapp
  npm start

Happy hacking!

C:\Users\mkanm>cd ticketraisingapp

C:\Users\mkanm\ticketraisingapp>npm start

> ticketraisingapp@0.1.0 start
> react-scripts start

(node:10904) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the 'setupMidd
lewares' option.
(Use `node --trace-deprecation ...` to show where the warning was created)
(node:10904) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the 'setupMi
ddlewares' option.
Starting the development server...
Compiled successfully!

You can now view ticketraisingapp in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.0.135:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

Output :


React App
http://localhost:3000
Getting Started YouTube Maps InGo-MMT JPW - Partner Portal Jio SSO Login Gmail
Other Bookmarks

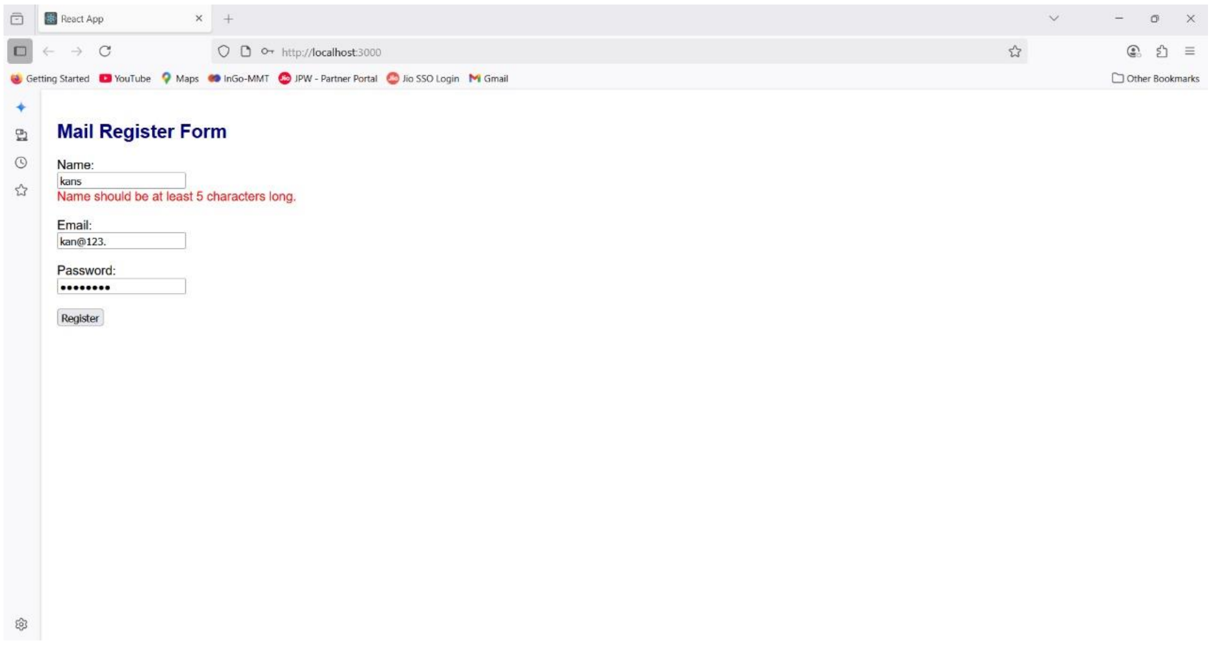
Mail Register Form

Name:

Email:

Password:

Register


React App
http://localhost:3000
Getting Started YouTube Maps InGo-MMT JPW - Partner Portal Jio SSO Login Gmail
Other Bookmarks

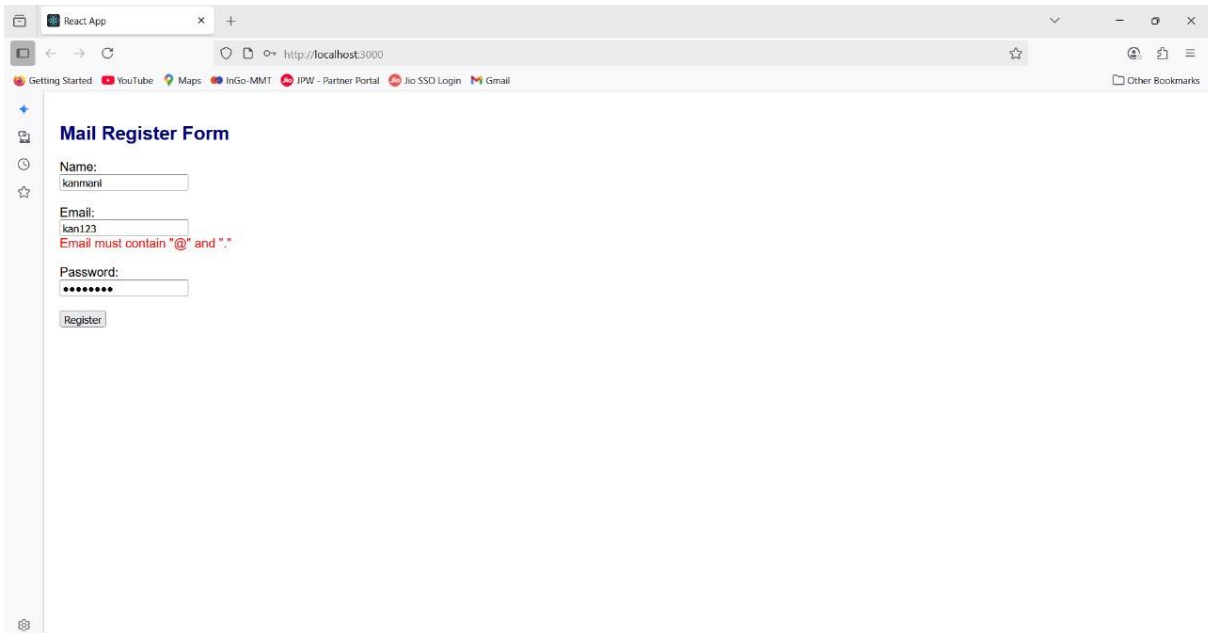
Mail Register Form

Name:
kans
Name should be at least 5 characters long.

Email:
kan@123.

Password:
••••••••

Register



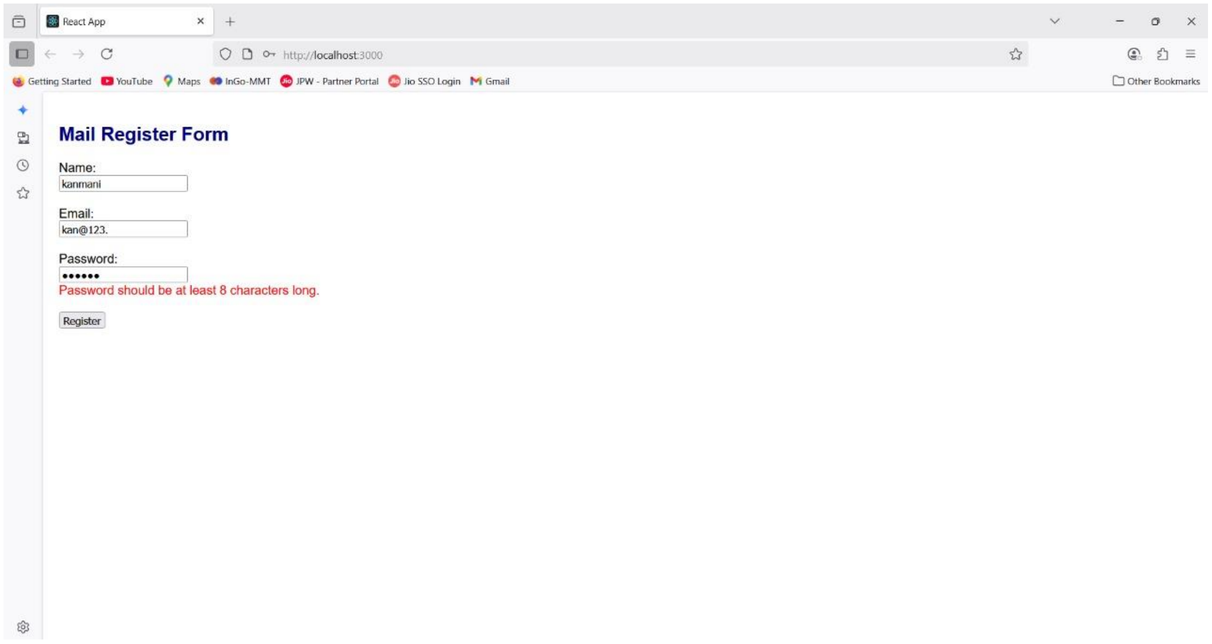
Mail Register Form

Name:
kanmani

Email:
kan123
Email must contain "@" and "."

Password:
••••••••

Register



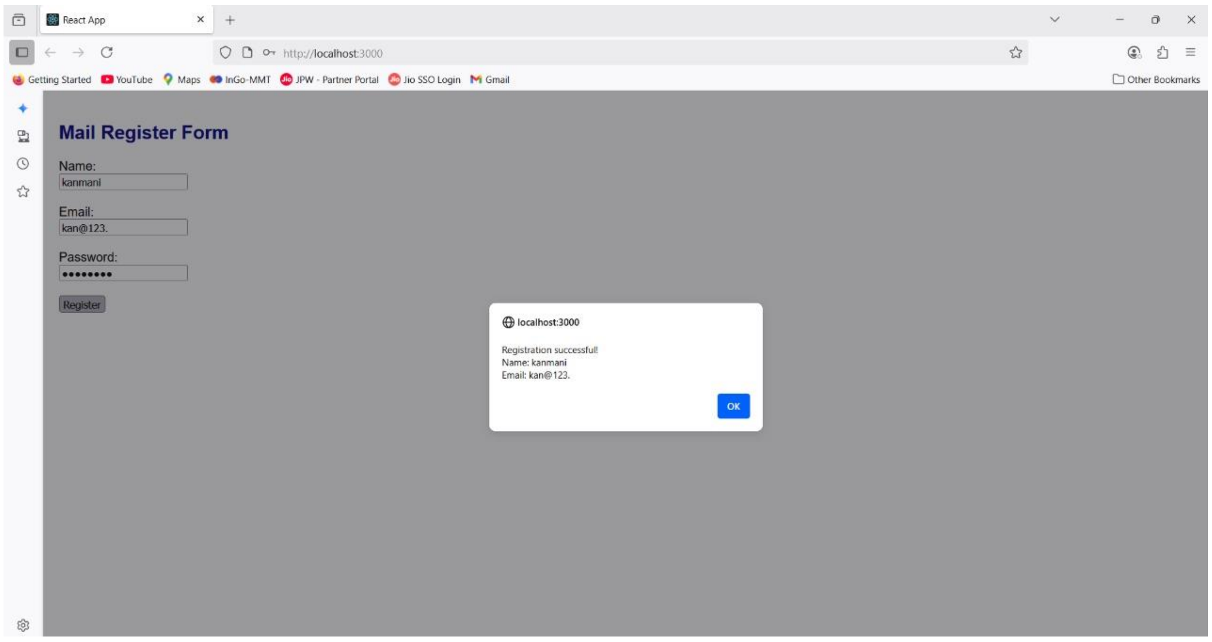
Mail Register Form

Name:
kanmani

Email:
kan@123.

Password:
••••••
Password should be at least 8 characters long.

Register


Mail Register Form

Name:
kanmani

Email:
kan@123.

Password:
••••••••

localhost:3000

Registration successful!
Name: kanmani
Email: kan@123.

KANMANI MURUGHAIYAN

SUPERSET ID: 6407636