# FeedBack -HOC-App Case Study

## Project Structure

```
feedback-hoc-app/
 ├── src/
│   ├── components/
│   │   ├── FeedbackForm.jsx
│   │   ├── withValidation.jsx
│   │   ├── withAuthentication.jsx
│   │   ├── withLogging.jsx
│   ├── App.jsx
│   ├── index.jsx
└── package.json
```

---

**withValidation.jsx**

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

const withValidation = (WrappedComponent) => {
  return function ValidatedComponent(props) {
    const [error, setError] = useState("");

    const validate = (fields) => {
```

```
    for (const key in fields) {
      if (!fields[key]) {
        setError(`Please fill out the ${key} field`);
        return false;
      }
    }
    setError("");
    return true;
  };

  return (
    <div>
      {error && <p style={{ color: "red" }}>{error}</p>}
      <WrappedComponent {...props} validate={validate} />
    </div>
  );
 };
};

export default withValidation;
```

---

**withAuthentication.jsx**

```
import React from "react";
import { Navigate } from "react-router-dom";

const withAuthentication = (WrappedComponent) => {
 return function AuthenticatedComponent(props) {
    const isLoggedIn = localStorage.getItem("employee"); // Mock login

    if (!isLoggedIn) {
      return <Navigate to="/login" />;
    }

    return <WrappedComponent {...props} />;
  };
};

export default withAuthentication;
```

---

**withLogging.jsx**

```
import React, { useEffect } from "react";

const withLogging = (WrappedComponent) => {
  return function LoggingComponent(props) {
    useEffect(() => {
      console.log("Form Opened");
    }, []);

    const logAction = (message) => {
      console.log("LOG:", message);
    };

    return <WrappedComponent {...props} logAction={logAction} />;
  };
};

export default withLogging;
```

---

**FeedbackForm.jsx**

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

const FeedbackForm = ({ validate, logAction }) => {
  const [formData, setFormData] = useState({
    milestone: "",
    comments: "",
  });

  const handleChange = (e) => {
    setFormData({ ...formData, [e.target.name]: e.target.value });
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    if (validate(formData)) {
      logAction(`Feedback Submitted: ${JSON.stringify(formData)}`);
      alert("Feedback Submitted Successfully!");
    } else {
      logAction("Validation Error Occurred");
    }
  };
  return (
    <form onSubmit={handleSubmit}>
      <label>
        Milestone 2 Completed?:
        <select name="milestone" value={formData.milestone}
onChange={handleChange}>
          <option value="">--Select--</option>
          <option value="Yes">Yes</option>
          <option value="No">No</option>
        </select>
      </label>
      <br />
      <label>
        Comments:
        <input
          type="text"
          name="comments"
          value={formData.comments}
          onChange={handleChange}
        />
      </label>
      <br />
      <button type="submit">Submit Feedback</button>
    </form>
  );
};
```

```
export default FeedbackForm;
```

---

**Wrapping FeedbackForm with HOCs in App.jsx**

```
import React from "react";
import FeedbackForm from "./components/FeedbackForm";
import withValidation from "./components/withValidation";
import withAuthentication from "./components/withAuthentication";
import withLogging from "./components/withLogging";

const EnhancedFeedbackForm = withAuthentication(
  withValidation(
    withLogging(FeedbackForm)
  )
);

const App = () => {
  return (
    <div>
      <h2>Employee Feedback Portal</h2>
      <EnhancedFeedbackForm />
    </div>
  );
};

export default App;
```

---