npm install @reduxjs/toolkit react-redux
npm i axios

src/redux/store.js

```
import { configureStore } from "@reduxjs/toolkit";
import userReducer from "./slices/userSlice";
export const store = configureStore({
  reducer: {
    users: userReducer,
  },
});
```

src/redux/slices/userSlice.js

```
import { createSlice, createAsyncThunk } from "@reduxjs/toolkit";
import axios from "axios";
const BASE_URL = "http://localhost:5001/users";
const BASE_URL = import.meta.env.VITE_BASE_URL;
// Fetch users
export const fetchUsers = createAsyncThunk("users/fetch", async () => {
  const res = await axios.get(BASE_URL, {
    headers: {
      Authorization: `Bearer ${localStorage.getItem("token")}`,
    },
  });
  return res.data;
});
// Create user
export const createUser = createAsyncThunk("users/create", async (data) => {
  const res = await axios.post(BASE_URL, data, {
    headers: {
      Authorization: `Bearer ${localStorage.getItem("token")}`,
    },
  });
  return res.data;
});
// Update user
export const updateUser = createAsyncThunk("users/update", async (data) => {
  const res = await axios.put(`${BASE_URL}/${data.id}`, data, {
    headers: {
      Authorization: `Bearer ${localStorage.getItem("token")}`,
    },
  });
  return res.data;
});
// Delete user
export const deleteUser = createAsyncThunk("users/delete", async (id) => {
  await axios.delete(`${BASE_URL}/${id}`, {
    headers: {
      Authorization: `Bearer ${localStorage.getItem("token")}`,
    },
  });
  return id;
});
const userSlice = createSlice({
  name: "users",
  initialState: {
    list: [],
    loading: false,
    error: null,
  },
```

```
  extraReducers: (builder) => {
    builder
      // Fetch
      .addCase(fetchUsers.pending, (state) => {
        state.loading = true;
      })
      .addCase(fetchUsers.fulfilled, (state, action) => {
        state.loading = false;
        state.list = action.payload;
      })
      // Create
      .addCase(createUser.fulfilled, (state, action) => {
        state.list.push(action.payload);
      })
      // Update
      .addCase(updateUser.fulfilled, (state, action) => {
        const index = state.list.findIndex((u) => u.id === action.payload.id);
        state.list[index] = action.payload;
      })
      // Delete
      .addCase(deleteUser.fulfilled, (state, action) => {
        state.list = state.list.filter((u) => u.id !== action.payload);
      });
  },
});
export default userSlice.reducer;
```

Src/main.jsx

import { StrictMode } from 'react'

```
import { createRoot } from 'react-dom/client'
import './index.css'
import App from './App.jsx'
import { Provider } from "react-redux";
import { store } from "./redux/store";
createRoot(document.getElementById('root')).render(
  <StrictMode>
    <Provider store={store}>
      <App />
    </Provider>
  </StrictMode>,
)
```

UserForm.jsx

```
import { useDispatch } from "react-redux";
import { createUser } from "../redux/slices/userSlice";
const dispatch = useDispatch();
const handleSubmit = (e) => {
    e.preventDefault();
    dispatch(
      createUser(form)
    );
    setForm({ name: "", email: "", password: "" });
  };
const handleChange = (e) => {
    setForm({ ...form, [e.target.name]: e.target.value });
  };
return (
    <form onSubmit={handleSubmit} className="form">
      <input name="name" placeholder="Name" value={form.name}
onChange={handleChange} />
```

```
      <input name="email" placeholder="Email" value={form.email}
onChange={handleChange} />
      <input name="age" placeholder="Age" value={form.age}
onChange={handleChange} />
      <button type="submit">Add User</button>
    </form>
  );
```

UserList.jsx

```
import { useEffect } from "react";
import { useDispatch, useSelector } from "react-redux";
import { fetchUsers, deleteUser } from "../redux/slices/userSlice";
export default function UserList() {
  const dispatch = useDispatch();
  const { list, loading } = useSelector((state) => state.users);
  useEffect(() => {
    dispatch(fetchUsers());
  }, []);
  if (loading) return <p>Loading...</p>;
  return (
    <div>
      <h2>User List</h2>
      {list.map((u) => (
        <div key={u.id}>
          {u.name} - {u.email} ({u.age})
          <button onClick={() => dispatch(deleteUser(u.id))}>Delete</button>
        </div>
      ))}
    </div>
  );
}
```