LAB EXERCISE:

Task 1:

• Create a React component that fetches data from a public API (e.g., a list of users) and displays it in a table format.

• Create a React app with Json-server and use Get , Post , Put , Delete & patch method on Json-server API.

API Example.jsx:

```jsx
import React, { useEffect, useState } from 'react'

const API_Example = () => {
    const [data, setData] = useState([]);
    useEffect(()=>{
        fetch('https://fakestoreapi.com/products')
        .then(response => response.json())
        .then(ab => setData(ab));
    },[])
  return (
    <div>
        <table border={2}>
            <thead>
                <tr>
                    <th>ID</th>
                    <th>Image</th>
                    <th>product</th>
                    <th>Category</th>
                    <th>Price</th>
                </tr>
            </thead>
            <tbody>
                {
                    data.map((i)=>{
                        return(
                            <tr>
                                <td>{i.id}</td>
                                <td><img src={i.image} height={'50px'} width={'50px'}/></td>
                                <td>{i.title}</td>
                                <td>{i.category}</td>
                                <td>{i.price}</td>
                            </tr>
                        )
                    })
                }
            </tbody>
        </table>
    </div>
  )
}

export default API_Example
```

App.jsx:

```jsx
import './App.css'
import API_Example from './API_Example'

function App() {

  return (
    <>
      <API_Example />
    </>
  )
}

export default App
```

App2.jsx:

```jsx
import './App.css'
import First from './First'

function App() {

  return (
    <>
      <First />
    </>
  )
}

export default App
```

First.jsx:

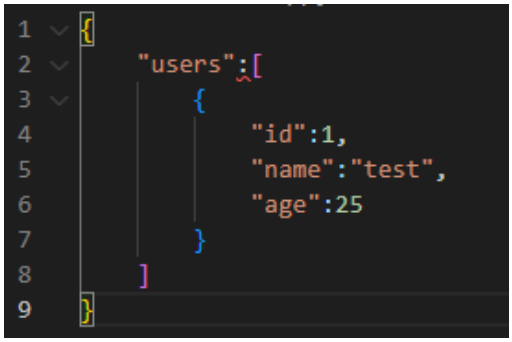
```jsx
import React, { useState } from 'react'

const First = () => {
    const [data,setData] = useState({
        uname:'',
        age:""
    })
    const [id,setId] = useState('')
    const [alldata,setAlldata]= useState([])
    const handleChange = (e)=>{
        const {name,value} = e.target
        setData({
            ...data,
            [name]:value
        })
    }
    const saveData = (e)=>{
        e.preventDefault()
        // setAlldata([
        //     ...alldata,
        //     data
        // ])
        axios.post("http://localhost:3000/users",data)
    }
    const delData = (id)=>{
        let res = alldata.filter((i,index)=>{
            return index != id
        })
        setAlldata(res)
    }
    const editData = (id)=>{
        let res = alldata.find((i,index)=>{
            return index == id
        })
        setData(res)
        setId(id)
    }
  return (
    <div>
      <form action="#" method='post' onSubmit={saveData}>
        Name:
        <input type="text" name="uname" id="uname" value={data.uname} onChange={handleChange}/><br /><br />
        Age:
        <input type="number" name="age" id="age" value={data.age} onChange={handleChange}/><br /><br />
        <input type="submit" value='Save'/><br /><br />
      </form>
      <br /><br />
```

```jsx
      <table>
        <thead>
            <tr>
                <th>Id</th>
                <th>Name</th>
                <th>Age</th>
                <th>Action</th>
            </tr>
        </thead>
        <tbody>
            {
                alldata.map((i,index)=>{
                    return (
                        <tr>
                            <td>{index+1}</td>
                            <td>{i.uname}</td>
                            <td>{i.age}</td>
                            <td>
                            <button onClick={()=>editData(index)}>Edit</button>
                                <button onClick={()=>delData(index)}>Delete</button>
                            </td>
                        </tr>
                    )
                })
            }
        </tbody>
      </table>
    </div>
  )
}

export default First
```

Db.json:



```
{
    "users":[
        {
            "id":1,
            "name":"test",
            "age":25
        }
    ]
}
```

Task 2:

• Create a React app crud and Authentication with firebase API.

• Implement google Authentication with firebase API.

App.jsx:

```
import React, { useState, useEffect } from 'react';
import { auth, googleProvider, signInWithPopup, signOut } from './firebase';
import { getDocs, collection } from 'firebase/firestore';

function App() {
  const [user, setUser] = useState(null);
  const [users, setUsers] = useState([]);
  const userCollection = collection(db, 'users');

  useEffect(() => {
    const unsubscribe = auth.onAuthStateChanged(setUser);
    return () => unsubscribe();
  }, []);

  const handleGoogleLogin = async () => {
    try {
      const result = await signInWithPopup(auth, googleProvider);
      setUser(result.user);
      alert('Logged in with Google');
    } catch (error) {
      alert(error.message);
    }
  };

  const handleLogout = async () => {
    await signOut(auth);
    setUser(null);
    alert('Logged out');
  };

  const fetchUsers = async () => {
    const data = await getDocs(userCollection);
    setUsers(data.docs.map(doc => doc.data()));
  };

  useEffect(() => {
    fetchUsers();
  }, []);

  return (
    <div>
      <h2>Firebase Google Authentication</h2>
```

```
      {user ? (
        <>
          <p>Welcome, {user.displayName}</p>
          <button onClick={handleLogout}>Logout</button>
          <hr />
          <h3>Users</h3>
          <ul>
            {users.map((user, index) => (
              <li key={index}>{user.name}</li>
            ))}
          </ul>
        </>
      ) : (
        <button onClick={handleGoogleLogin}>Login with Google</button>
      )}
    </div>
  );
}

export default App;
```

Firebase.jsx:

```
import { initializeApp } from 'firebase/app';
import { getAuth, GoogleAuthProvider, signInWithPopup, signOut } from 'firebase/auth';
import { getFirestore } from 'firebase/firestore';

const firebaseConfig = {
  apiKey: "YOUR_API_KEY",
  authDomain: "YOUR_PROJECT.firebaseapp.com",
  projectId: "YOUR_PROJECT_ID",
  storageBucket: "YOUR_PROJECT.appspot.com",
  messagingSenderId: "SENDER_ID",
  appId: "APP_ID"
};

const app = initializeApp(firebaseConfig);

export const auth = getAuth(app);
export const db = getFirestore(app);
export const googleProvider = new GoogleAuthProvider();
export { signInWithPopup, signOut };
```

Task 3:

• Implement error handling and loading states for the API call. Display a loading spinner while the data is being fetched.

```jsx
import React, { useState, useEffect } from 'react';
import { auth, googleProvider, signInWithPopup, signOut, db } from './firebase';
import { collection, getDocs, addDoc, updateDoc, deleteDoc, doc } from 'firebase/firestore';

function App() {
  const [user, setUser] = useState(null);
  const [loading, setLoading] = useState(false);
  const [error, setError] = useState(null);
  const [users, setUsers] = useState([]);
  const [input, setInput] = useState('');
  const userCollection = collection(db, 'users');

  // Fetch data from Firestore with error handling and loading states
  const fetchUsers = async () => {
    setLoading(true);
    setError(null);
    try {
      const data = await getDocs(userCollection);
      setUsers(data.docs.map(doc => ({ ...doc.data(), id: doc.id })));
    } catch (err) {
      setError('Error fetching data. Please try again.');
    } finally {
      setLoading(false);
    }
  };

  // Google login handler
  const handleGoogleLogin = async () => {
    setLoading(true);
    setError(null);
    try {
      const result = await signInWithPopup(auth, googleProvider);
      setUser(result.user);
    } catch (error) {
      setError('Error during login. Please try again.');
    } finally {
      setLoading(false);
    }
  };

  // Logout handler
  const handleLogout = async () => {
    setLoading(true);
    setError(null);
    try {
      await signOut(auth);
      setUser(null);
```

```
    } catch (error) {
      setError('Error logging out. Please try again.');
    } finally {
      setLoading(false);
    }
  };

  // Add a user to Firestore
  const addUser = async () => {
    if (input) {
      setLoading(true);
      setError(null);
      try {
        await addDoc(userCollection, { name: input });
        setInput('');
        fetchUsers();
      } catch (err) {
        setError('Error adding user. Please try again.');
      } finally {
        setLoading(false);
      }
    }
  };

  // Update user data
  const updateUser = async (id) => {
    const userDoc = doc(db, 'users', id);
    const newName = prompt('Enter new name');
    if (newName) {
      setLoading(true);
      setError(null);
      try {
        await updateDoc(userDoc, { name: newName });
        fetchUsers();
      } catch (err) {
        setError('Error updating user. Please try again.');
      } finally {
        setLoading(false);
      }
    }
  };

```

```jsx
  // Delete a user from Firestore
  const deleteUser = async (id) => {
    const userDoc = doc(db, 'users', id);
    setLoading(true);
    setError(null);
    try {
      await deleteDoc(userDoc);
      fetchUsers();
    } catch (err) {
      setError('Error deleting user. Please try again.');
    } finally {
      setLoading(false);
    }
  };

  useEffect(() => {
    fetchUsers();
  }, []);

  return (
    <div>
      <h2>Firebase Google Authentication & CRUD</h2>

      {/* Show loading spinner or text if loading */}
      {loading && <div>Loading...</div>}

      {/* Show error message if there was an issue */}
      {error && <div style={{ color: 'red' }}>{error}</div>}

      {/* Show login or user info */}
      {user ? (
        <>
          <p>Welcome, {user.displayName}</p>
          <button onClick={handleLogout}>Logout</button>
          <hr />
          <h3>Users</h3>
          <input
            placeholder="Enter name"
            value={input}
            onChange={(e) => setInput(e.target.value)}
          />
          <button onClick={addUser}>Add User</button>
          <ul>
            {users.map((user) => (
              <li key={user.id}>
                {user.name}
                <button onClick={() => updateUser(user.id)}>Edit</button>
                <button onClick={() => deleteUser(user.id)}>Delete</button>
              </li>
            ))}
          </ul>
        </>
      ) : (
        <button onClick={handleGoogleLogin}>Login with Google</button>
      )}
    </div>
  );
}

export default App;
```