Week 3: Important concepts of React. js

a. Write a React program to implement a counter button using React use State hook

Program:

```
import React, { useState } from "react";
function Counter() {
const [count, setCount] = useState(0); // Initialize count state to 0
return (
<div style={{ textAlign: "center", marginTop: "50px" }}>
<h2>Counter Example</h2>
<p>Current Count: {count}</p>
<button onClick={() => setCount(count + 1)}>Increase</button>
<button onClick={() => setCount(count - 1)} style={{ marginLeft: "10px" }}>
Decrease
</button>
</div>
);
}
export default Counter;
```

Output:

b. Write a React program to fetch the data from an API using React use Effect hook

Program:

```
import React, { useEffect, useState } from "react";
function FetchData() {
const [users, setUsers] = useState([]);
```

```
useEffect(() => {
// Fetch dummy data
fetch("https://jsonplaceholder.typicode.com/users")
.then((response) => response.json())
.then((data) => setUsers(data));
}, []);
return (
<div style={{ textAlign: "center", marginTop: "30px" }}>
<h2>Users List (Fetched from API)</h2>
<ul style={{ listStyle: "none" }}>
{users.map((user) => (
<li key={user.id}>
{user.name} ({user.email})
</li>
))}
</ul>
</div>
);
}
export default FetchData;
```

Output:

c. Write a React program with two react components sharing data using Props.

Output:

```
import React from "react";
function ChildComponent({ message }) {
return <h3>Message from Parent: {message}</h3>;
}
function ParentComponent() {
const text = "Hello from Parent!";
return (
<div style={{ textAlign: "center", marginTop: "50px" }}>
<h2>Parent and Child Example</h2>
<ChildComponent message={text} />
</div>
);
}
export default ParentComponent;
```

Output:

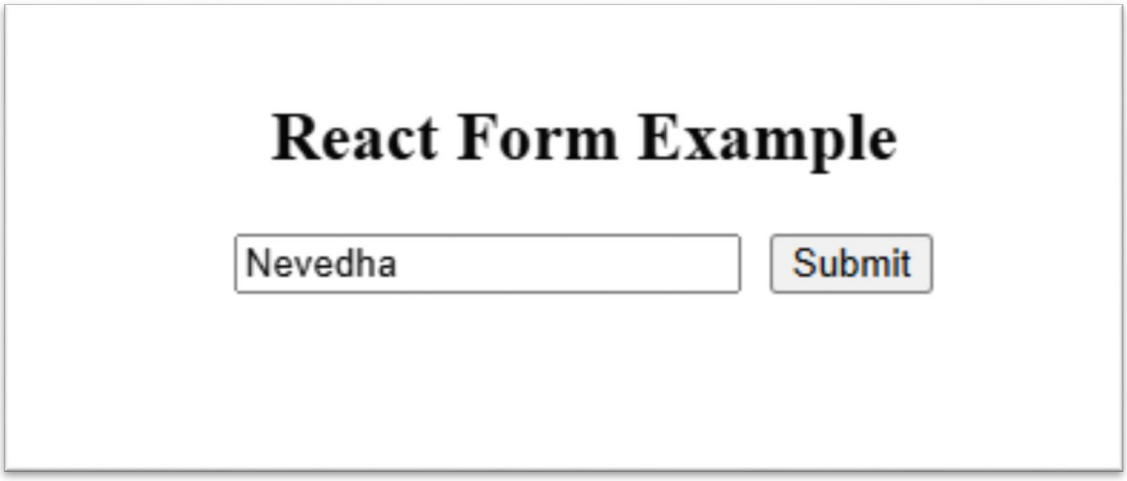
d. Write a React program to implement the forms in react

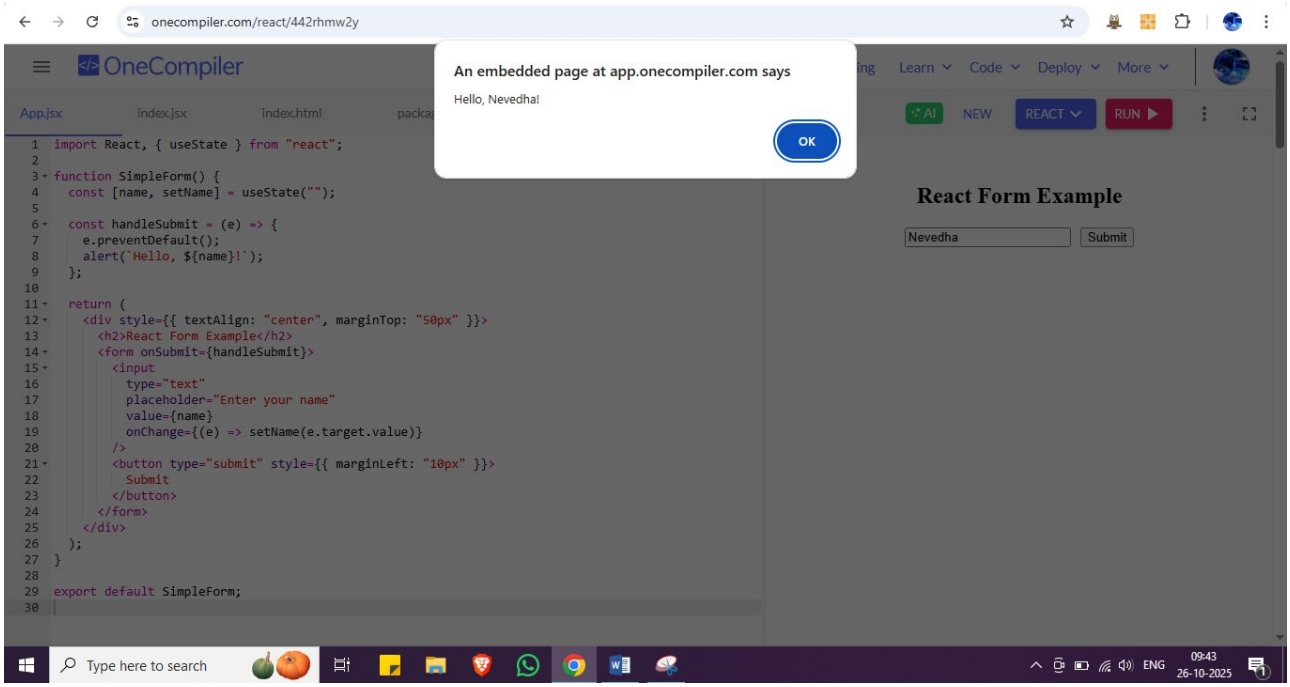
Program:

```
import React, { useState } from "react";
function SimpleForm() {
const [name, setName] = useState("");
const handleSubmit = (e) => {
e.preventDefault();
```

```
alert(`Hello, ${name}!`);
};
return (
<div style={{ textAlign: "center", marginTop: "50px" }}>
<h2>React Form Example</h2>
<form onSubmit={handleSubmit}>
<input
type="text"
placeholder="Enter your name"
value={name}
onChange={(e) => setName(e.target.value)}
/>
<button type="submit" style={{ marginLeft: "10px" }}>
Submit
</button>
</form>
</div>
);
}
export default SimpleForm;
```

Output:





e. Write a React program to implement the iterative rendering using map() function.

Program:

```
import React from "react";
function ItemList() {
const fruits = ["Apple", "Banana", "Cherry", "Mango"];
return (
<div style={{ textAlign: "center", marginTop: "40px" }}>
<h2>Fruit List (Rendered using map)</h2>
<ul style={{ listStyle: "none" }}>
{fruits.map((fruit, index) => (
<li key={index}>{fruit}</li>
))}
</ul>
</div>
);
}
export default ItemList;
```

Output:

Week 6:Introduction to Node. js and Express. Js

a. Write a program to print the 'hello world' in the browser console using Express. Js

Program:

```
// Import express
const express = require("express");
// Create an express app
const app = express();
// Define a simple route
app.get("/", (req, res) => {
console.log("Hello World"); // This prints in your console
res.send("Check your console! 'Hello World' printed there.");
});
// Start the server
app.listen(3000, () => {
console.log("Server running at http://localhost:3000");
});
```

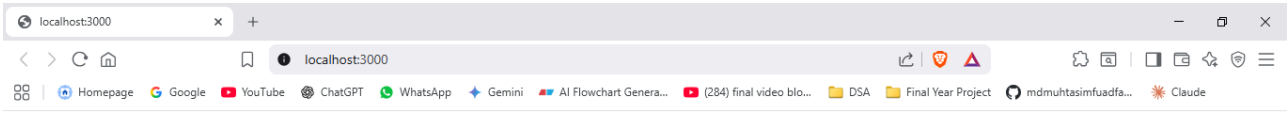
Output:

npm init –y

npm install express

node app.js

Server running at http://localhost:3000



b. Write a program to implement the CRUD operations using Express. Js

Program:

(crud.js)

```
const express = require("express");
const app = express();
app.use(express.json());
// Root route
app.get("/", (req, res) => {
res.sendFile(__dirname + "/index.html"); // Serve the HTML page
});
// Initial users
let users = [
{ id: 1, name: "Nevedha" },
{ id: 2, name: "Nikitha" },
];
// API routes
app.get("/users", (req, res) => res.json(users));
app.post("/users", (req, res) => {
const newUser = { id: users.length + 1, name: req.body.name };
users.push(newUser);
```

```
res.json({ message: "User added", user: newUser });
});
app.put("/users/:id", (req, res) => {
const id = parseInt(req.params.id);
const user = users.find(u => u.id === id);
if (!user) return res.status(404).json({ message: "User not found" });
user.name = req.body.name;
res.json({ message: "User updated", user });
});
app.delete("/users/:id", (req, res) => {
const id = parseInt(req.params.id);
users = users.filter(u => u.id !== id);
res.json({ message: "User deleted" });
});
// Start server
app.listen(3000, () => console.log("Server running at http://localhost:3000"));
```

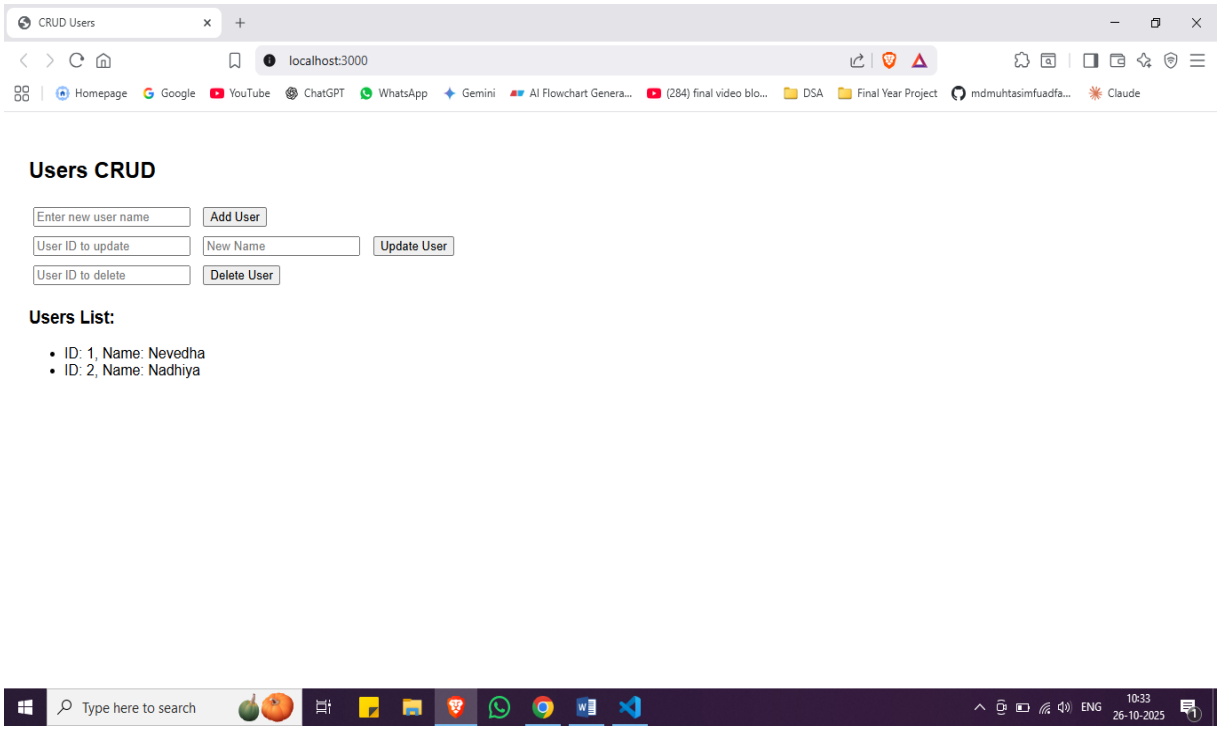
(index.html)

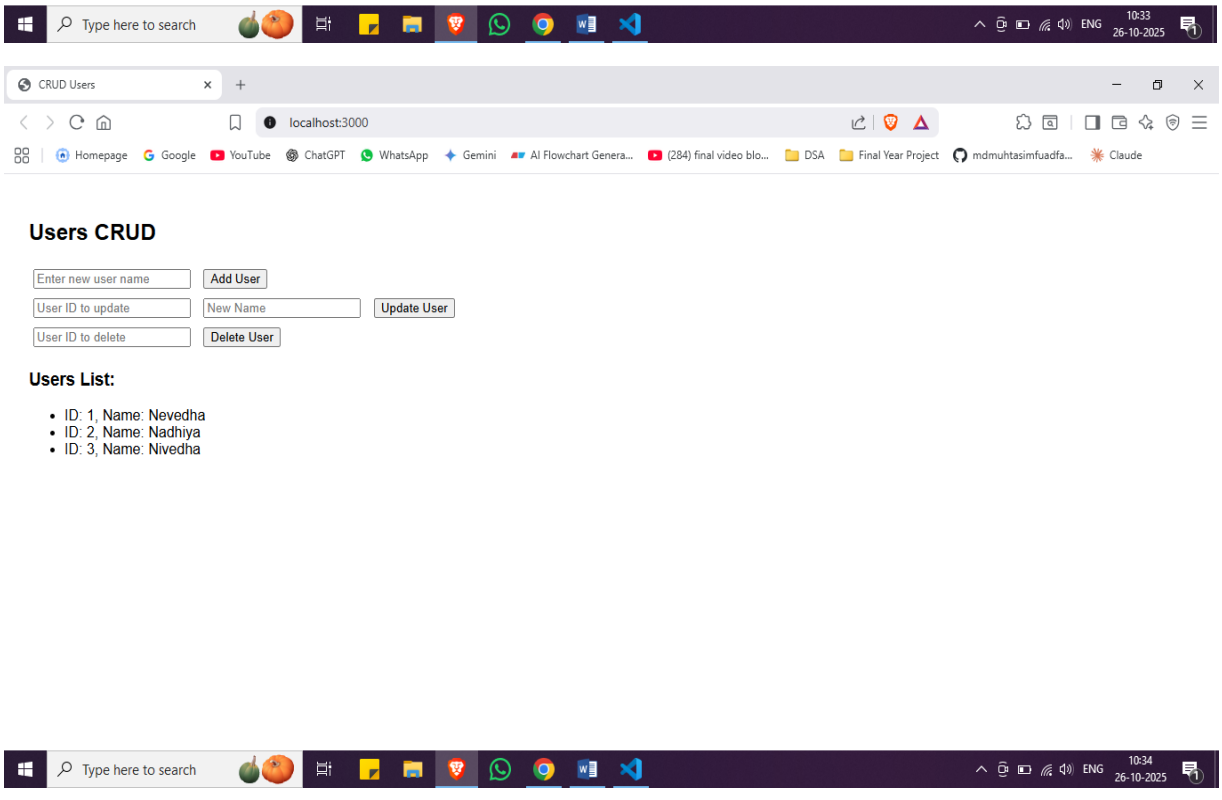
```
<!DOCTYPE html>
<html>
<head>
<title>CRUD Users</title>
<style>
body { font-family: Arial; padding: 20px; }
input, button { margin: 5px; }
</style>
</head>
<body>
<h2>Users CRUD</h2>
<div>
<input type="text" id="newUserName" placeholder="Enter new user name">
<button onclick="addUser()">Add User</button>
</div>
```

```
<div>
<input type="number" id="updateId" placeholder="User ID to update">
<input type="text" id="updateName" placeholder="New Name">
<button onclick="updateUser()">Update User</button>
</div>
<div>
<input type="number" id="deleteId" placeholder="User ID to delete">
<button onclick="deleteUser()">Delete User</button>
</div>
<h3>Users List:</h3>
<ul id="userList"></ul>
<script>
// Fetch and display users
async function fetchUsers() {
const res = await fetch("/users");
const users = await res.json();
const list = document.getElementById("userList");
list.innerHTML = "";
users.forEach(u => {
const li = document.createElement("li");
li.textContent = `ID: ${u.id}, Name: ${u.name}`;
list.appendChild(li);
});
}
async function addUser() {
const name = document.getElementById("newUserName").value;
if (!name) return alert("Enter a name");
await fetch("/users", {
method: "POST",
headers: { "Content-Type": "application/json" },
body: JSON.stringify({ name })
```

```
});
document.getElementById("newUserName").value = "";
fetchUsers();
}
async function updateUser() {
const id = document.getElementById("updateId").value;
const name = document.getElementById("updateName").value;
if (!id || !name) return alert("Enter ID and new name");
await fetch(`/users/${id}`, {
method: "PUT",
headers: { "Content-Type": "application/json" },
body: JSON.stringify({ name })
});
document.getElementById("updateId").value = "";
document.getElementById("updateName").value = "";
fetchUsers();
}
async function deleteUser() {
const id = document.getElementById("deleteId").value;
if (!id) return alert("Enter ID to delete");
await fetch(`/users/${id}`, { method: "DELETE" });
document.getElementById("deleteId").value = "";
fetchUsers();
}
// Initial fetch
fetchUsers();
</script>
</body>
</html>
```

Output:





Week 7: Introduction to My SQL

a. Write a My SQL queries to create table, and insert the data, update the data in the table

Program:

```
CREATE TABLE users (
id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(50) NOT NULL
```

```
);
INSERT INTO users (name) VALUES ('Nevedha'), ('Nikitha');
SELECT * FROM users;
UPDATE users SET name = 'NevedhaUpdated' WHERE id = 1;
SELECT * FROM users;
DELETE FROM users WHERE id = 2;
SELECT * FROM users;
```

Output:

id	name
	Nevedha
	Nikitha
	Nevedha
	Nikitha