**Exp 01 - create static webpage using html & css**

Index.html. <!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Home Page</title>

<style>

body { font-family: sans-serif; text-align: center; padding: 50px; background-color: #e6f7ff; }

.content { background-color: white; padding: 20px; border-radius: 5px; box-shadow: 0 2px 4px rgba(0,0,0,0.1); }

a { color: #007bff; text-decoration: none; font-weight: bold; }

a:hover { text-decoration: underline; }

</style>

</head>

<body>

<div class="content">

<h1>Welcome to Page 1 🏠</h1>

<p>This is the main or \*Home Page\*.</p>

<p>Go to the <a href="page2.html">Second Page</a>.</p>

</div>

</body>

</html>

Page2.html <!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Second Page</title>

<style>

body { font-family: sans-serif; text-align: center; padding: 50px; background-color: #fff0e6; }

.content { background-color: white; padding: 20px; border-radius: 5px; box-shadow: 0 2px 4px rgba(0,0,0,0.1); }

a { color: #ff6347; text-decoration: none; font-weight: bold; }

a:hover { text-decoration: underline; }

</style>

</head>

<body>

<div class="content">

<h1>You are on Page 2 🚀</h1>

<p>This is the \*Linked Page\*.</p>

<p>Go back to the <a href="index.html">Home Page</a>.</p>

</div>

</body>

</html>

**Exp 02 - JS validation on registration**

<!DOCTYPE html>

<html lang="en">

<head>

<title>Form Validation</title>

<link rel="stylesheet" href="style.css">

<script src="Validation.js"></script>

</head>

<body>

<h1 style="text-align: center;">REGISTRATION FORM</h1>

<form name="regform" onsubmit="return Validation()" method="post">

<p>Name: <input type="text" size="50" name="Name" /></p><br>

<p>Address: <input type="text" size="50" name="Address" /></p><br>

<p>Email: <input type="text" size="50" name="Email" /></p><br>

<p>Password: <input type="password" size="50" name="Password" /></p><br>

<p>Telephone No: <input type="text" size="50" name="Telephone" /></p><br>

<p>Select Your Courses:

<select type="text" name="subject">

<option value="">--Select--</option>

<option >B-TECH</option>

<option >B.E</option>

<option >BCA</option>

<option >BBA</option>

<option >B.com</option>

<option >LLB</option>

</select>

</p><br>

<p>Comments:<textarea cols="55" name="comment"></textarea></p>

<p>

<input type="submit" value="Send" name="submit" />

<input type="reset" value="Reset" name="reset" />

</p>

</form>

</body>

</html>

**Javascript code:**

function Validation()

{

var name=document.forms.regform.Name.value;

var email=document.forms.regform.Email.value;var phone=document.forms.regform.Telephone.value;

var what=document.forms.regform.subject.value;

var password=document.forms.regform.Password.value;

var address=document.forms.regform.Address.value;

var regEmail=/^\w+([\.-]?\w+)\*@\w+([\.-]?\w+)\*(\.\w{2,3})+$/g;

var regPhone=/^\d{10}$/;

var regName = /\d+$/g;

if (name == "" || regName.test(name)) {

window.alert("Please enter your name properly.");

name.focus();

return false;

}

if (address == "") {

window.alert("please enter your address.");

address.focus();

return false;

}

if (email == "" || !regEmail.test(email)) {

window.alert("Please enter a valid e-mail address.");

email.focus();

return false;

}

if (password == "" ) {

alert("Please enter your password");

password.focus();

return false;

}

if(password.length <6) {

alert("Password should be atleast 6 character long");

password.focus();

return false;

}

if (phone == "" || !regPhone.test(phone)) {

alert("Please enter valid phone number.");

phone.focus();

return false;

}

if (what.selectedIndex == -1) {

alert("Please enter your course.");

subject.focus();

return false;

}

return true;}

**Exp 03 - program for js function**

1.Arrow fuction with default parameter

const verifyAgeWithDefault = (age = 18) => age >= 18 ? "Adult" : "Minor";

console.log("Age 25: " + verifyAgeWithDefault(25));

console.log("No Age: " + verifyAgeWithDefault());

2.Generator fuction

function\* ageChecker(ages) {

for (const age of ages) {

let status = age >= 18 ? "Adult" : "Minor";

yield Age ${age}: ${status};

}

}

let ageList = [16, 21, 35, 12];

const check = ageChecker(ageList);

console.log(check.next().value);

console.log(check.next().value);

console.log(check.next().value);

3.Rest parameter

function logAgeVerification(...ages) {

console.log(Checking ${ages.length} users:);

for (const age of ages) {

let status = age >= 18 ? "Adult" : "Minor";

console.log(- Age ${age}: ${status});

}

}

logAgeVerification(17, 20, 15, 45);

4.Anonymous fuction

const checkUserAge = function(age) {

return age >= 18 ? "Adult" : "Minor";

};

console.log("User status: " + checkUserAge(15));

**Exp 04 - implement javascript class and object**

class Person {

constructor(name, age) {

this.name = name;

this.age = age;

}

greet() {

console.log("My name is: " + this.name + " i am " + this.age + " years old");

}

}

let person1 = new Person("ram", 20);

let person2 = new Person("ravan", 20);

person1.greet();

person2.greet();

**Exp 05 - implement javascript inheritance concept**

1.Single inheritance

class Animal {

move() {

console.log("Animal moves.");

}

}

class Dog extends Animal {}

const buddy = new Dog();

buddy.move();

// Output: Animal moves.

2.Multilevel Inheritance

class Animal {

eat() { console.log("Animal is eating."); }

}

class Dog extends Animal {

bark() { console.log("Dog is barking."); }

}

class Puppy extends Dog {}

const pup = new Puppy();

pup.eat();

// Output: Animal is eating.

3.Hierarchical Inheritance

class Vehicle {

start() { console.log("Vehicle started."); }

}

class Car extends Vehicle {}

class Truck extends Vehicle {}

const myCar = new Car();

myCar.start();

// Output: Vehicle started.

4.Multiple Inheritance

// Mixin uses a regular function expression

const CanFly = function(Base) { return class extends Base { fly() { console.log("Flying high!"); } }; };

class Mammal {}

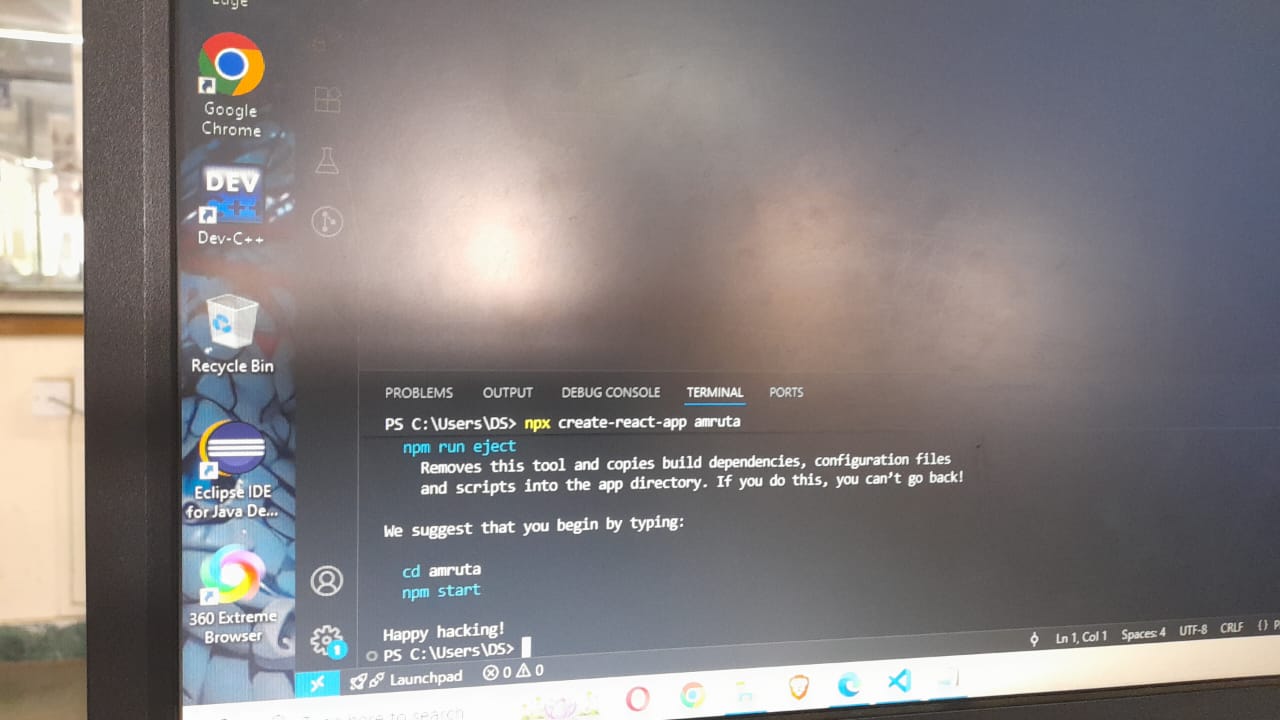
class Bat extends CanFly(Mammal) {}

const batman = new Bat();

batman.fly();

// Output: Flying high!

**Exp 06 - implement program for props and state**

****

[app.js](http://app.js)

import React from 'react';

import './App.css';

const App = (props) => {

return (

<div className="person">

<h2>Name: {props.name}</h2>

<h2>Age: {props.age}</h2>

<h2>College Name: {props.clgname}</h2>

<h2>Department: {props.dept}</h2>

</div>

);

}

export default App;

[index.js](http://index.js)

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<App

name="Danish Naik"

age={20}

clgname="Saraswati College of Engineering"

dept="Information Technology"

/>

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

State

[app.js](http://app.js)

import React from 'react';

import './App.css';

class Home extends React.Component {

constructor(props) {

super(props);

this.props = props;

this.state = {

Clgname: "Saraswati College of Engineering",

location: "Kharghar"

};

}

render() {

return (

<div>

<h1>This is home page.</h1>

<i>

College Name is : <b>{this.state.Clgname}</b> and location is : <b>{this.state.location}</b>

</i>

</div>

);

}

}

export default Home;

**Exp 07 - implement react router**

**Home.js**

import react from 'react';

function Home(){

return <h1>Welcome to the world of React Router!!</h1>

}

export default Home;

**Aboutus.js**

import react from 'react';

function Aboutus(){

return<div>

<h2> It's a About Us Page!!</h2>

read more about us:

<a href='https://www.google.com/'>

</a>

</div>

}

export default Aboutus;

**Contactus.js**

import React, {Component} from "react";

class Contactus extends Component{

render(){

return(

<div> <h2> GOT QUESTIONS??</h2>

<p> The easiest thing to do is post on our

<a href="http://forum.kirupa.com">formus</a>

</p>

</div>

);

}}

export default Contactus;

**index.js**

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

import FinalRoute from './FinalRoute';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<FinalRoute></FinalRoute>

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals(); **FinalRoute.js**

import React, { Component } from "react";

import { BrowserRouter as Router,Routes,Route,Link} from 'react-router-dom';

import Home from "./home";

import Aboutus from "./Aboutus";

import Contactus from "./Contactus";

import './App.css';

class FinalRoute extends Component{

render(){

return(

<Router>

<div className="App">

<ul className="App-header">

<li>

<Link to ="/">Home</Link>

</li>

<li>

<Link to ="/Aboutus">About us</Link>

</li>

<li>

<Link to ="/Contactus">Contact us</Link>

</li>

</ul>

<Routes>

<Route exact path="/" element={<Home/>}></Route> <Route exact path="/Aboutus" element={<Aboutus/>}></Route>

<Route exact path="/Contactus" element={<Contactus/>}></Route>

</Routes>

</div>

</Router>

);

}}

export default FinalRoute;

**Exp 08 - implement hooks in react js**

Index.js

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import reportWebVitals from './reportWebVitals';

import ParentComponent from './ParentComponent';

import StyledComponent from './StyledComponent';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

{/\* You can swap between these components to test them \*/}

{/\* <ParentComponent /> \*/}

<StyledComponent />

</React.StrictMode>

);

reportWebVitals();

ParentComponent.js

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

function ChildComponent({ onClick }) {

console.log('ChildComponent rendered'); // Check console to see when it re-renders

return (

<button onClick={onClick}>Click me</button>

);

}

function ParentComponent() {

const [count, setCount] = useState(0);

const [text, setText] = useState('');

const increment = useCallback(() => {

setCount(prevCount => prevCount + 1);

}, []); // Empty dependency array means the function is stable

return (

<div>

<h1>useCallback Example</h1>

<p>Count: {count}</p>

{/\* The ChildComponent only re-renders when 'count' or 'text' changes,

NOT when the 'increment' prop changes (because it doesn't) \*/}

<ChildComponent onClick={increment} />

<input

type="text"

value={text}

onChange={(e) => setText(e.target.value)}

placeholder="Type something..."

/>

<p>Input: {text}</p>

</div>

);

}

export default ParentComponent;

**Exp 09 - implement use insertion effect in react**

import React, { useInsertionEffect } from 'react';

// Define the style rule we want to inject

const styleRule = `

.dynamic-text {

color: darkgreen;

font-weight: bold;

}

`;

function StyledComponent() {

// useInsertionEffect is ideal for injecting styles before the browser reads the layout.

useInsertionEffect(() => {

// 1. Check if the style tag already exists

let style = document.getElementById('dynamic-style');

if (!style) {

// 2. If it doesn't exist, create it

style = document.createElement('style');

style.id = 'dynamic-style';

document.head.appendChild(style);

}

// 3. Insert the CSS rule

style.textContent = styleRule;

// Return a cleanup function (optional)

return () => {

// Clean up if the component unmounts (optional for global styles)

};

}, []); // Dependency array ensures it only runs once

return (

<div>

<h1>useInsertionEffect Example</h1>

{/\* This text will use the dynamically injected style \*/}

<p className="dynamic-text">This text is styled by useInsertionEffect.</p>

</div>

);

}

export default StyledComponent;

**Exp 10 - useRef & useEffect in react**

import React, { useState, useRef, useEffect } from 'react';

function CountPrev() {

const [count, setCount] = useState(0);

const prevCountRef = useRef();

// 1. Update the ref AFTER the render, whenever 'count' changes.

// This stores the 'current' count for the next render.

useEffect(() => {

prevCountRef.current = count;

}, [count]);

// The component always renders the count value before the useEffect runs.

// On the first render, prevCountRef.current is undefined.

// On the second render, prevCountRef.current holds the count from the first render (0).

const previousCount = prevCountRef.current;

return (

<div className="flex flex-col items-center justify-center min-h-screen bg-gray-50 p-4">

<div className="bg-white p-8 rounded-xl shadow-2xl w-full max-w-sm text-center">

<h1 className="text-2xl font-extrabold text-indigo-600 mb-6">

useRef for Previous State

</h1>

<p className="text-xl mb-2 font-medium text-gray-800">

Current Count: <span className="text-3xl font-bold text-indigo-500">{count}</span>

</p>

<p className="text-lg mb-6 text-gray-500">

Previous Count: <span className="text-xl font-semibold text-red-500">{previousCount === undefined ? 'N/A' : previousCount}</span>

</p>

<button

onClick={() => setCount(count + 1)}

className="w-full py-3 px-4 bg-indigo-500 text-white font-semibold rounded-lg shadow-md hover:bg-indigo-600 transition duration-200 focus:outline-none focus:ring-2 focus:ring-indigo-500 focus:ring-opacity-50"

>

Increment

</button>

</div>

</div>

);

}

export default CountPrev;

**Exp 11 - login credentials using hooks in react**

import React, { useState } from "react";

const SUCCESS\_MESSAGE\_TIMEOUT = 2000;

export default function LoginEx() {

const [email, setEmail] = useState("");

const [password, setPassword] = useState("");

const [success, setSuccess] = useState(false);

const [error, setError] = useState(false);

const handleSubmit = (e) => {

e.preventDefault();

// Reset flags on new submission

setSuccess(false);

setError(false);

const isSuccess = email === "dev@gmail.com" && password === "password";

if (isSuccess) {

setSuccess(true);

setTimeout(() => setSuccess(false), SUCCESS\_MESSAGE\_TIMEOUT);

// Optionally clear inputs on successful login

// setEmail('');

// setPassword('');

} else {

setError(true);

setTimeout(() => setError(false), SUCCESS\_MESSAGE\_TIMEOUT);

}

}

// Determine the message type and content

let message = null;

let messageClass = "";

if (success) {

message = "Welcome back!";

messageClass = "bg-green-100 text-green-700 border-green-400";

} else if (error) {

message = "Are you a hacker or just forgetful? Try again.";

messageClass = "bg-red-100 text-red-700 border-red-400";

}

return (

<div className="flex items-center justify-center min-h-screen bg-gray-100 p-4">

<div className="w-full max-w-md bg-white p-8 rounded-xl shadow-2xl">

<h1 className="text-3xl font-bold text-center text-gray-800 mb-8">Secure Login</h1>

{/\* Message Display Area \*/}

{message && (

<div

className={p-3 mb-4 rounded-lg border text-sm font-medium transition-opacity duration-300 ${messageClass}}

role="alert"

>

{message}

</div>

)}

<form onSubmit={handleSubmit} className="space-y-6">

<div>

<input

type="email"

placeholder="Enter email (dev@gmail.com)"

value={email}

onChange={(e) => setEmail(e.target.value)}

required

className="w-full px-4 py-3 border border-gray-300 rounded-lg focus:outline-none focus:ring-2 focus:ring-indigo-500 transition duration-150"

/>

</div>

<div>

<input

type="password"

placeholder="Enter password (password)"

value={password}

onChange={(e) => setPassword(e.target.value)}

required

className="w-full px-4 py-3 border border-gray-300 rounded-lg focus:outline-none focus:ring-2 focus:ring-indigo-500 transition duration-150"

/>

</div>

<button

type="submit"

className="w-full py-3 bg-indigo-600 text-white font-semibold rounded-lg shadow-lg hover:bg-indigo-700 transition duration-200 focus:outline-none focus:ring-4 focus:ring-indigo-500 focus:ring-opacity-50"

>

Log in

</button>

</form>

<p className="mt-6 text-center text-sm text-gray-500">

Use Email: <code className="font-mono text-indigo-600">dev@gmail.com</code> and Password: <code className="font-mono text-indigo-600">password</code>

</p>

</div>

</div>

);

}

**Exp 12 - express using node js**

const express = require('express');

const app = express();

const port = 3000;

app.get('/', (req, res) => {

res.send('Hello from Express.js!');

});

app.get('/users/:name', (req, res) => {

const userName = req.params.name; // Access the route parameter

res.send(`Hello, ${userName}! Welcome to the user page.`);

});

app.listen(port, () => {

console.log(`Express app listening at http://localhost:${port}`);

});