Full Stack Development
Assignment : 1

Name : Jayaram Patel
Roll no. : 48
PRN : 10322325311
Batch : A2

Aim :-

Develop responsive web design using HTML5, containing a form. Style the pages using CSS, Use of tag selector, class selector and id selectors. Use inline, Internal & External CSS, Apply Bootstrap CSS.

Objectives :-

i) To understand HTML Tags.
ii) To learn the styling of web pages using CSS.
iii) To learn Bootstrap Front End development.

Theory :-

1. Define Responsive Web Design (RWD). What is its primary goal.

Definition: Responsive web design (RWD) is a design technique that allows website to automatically adjust their layout, images and text so that they look and function well on different devices (desktops, laptops, etc.)

Primary goal:

- Enhance usability.
- Eliminate the need for zooming and horizontal scrolling.
- Improve performance and user experience on s screen sizes.
- Provide a mobile-first experience since mo access websites via mobile.

Q2. Explain the role of <meta name = "viewport" ...
Ans2. Why is this tag essential for RWD?
By default, mobile browsers display webpages in virtual desktop viewport (usually ~980px wide) shrinking the content to fit. The <meta viewp tag overrides this behaviour and tells the bro to match the sites width with the devices ac screen width.

Why essential:

- Without this tag, responsive layouts won't displa correctly.
- It prevents text and elements from appearing small on mobiles.
- Ensures media queries work properly.

Q3. How does bootstrap assit in creating a responsive layout? Discuss the concept of a grid system an how it adapts to different screen sizes.
Ans 3. Bootstrap is a popular CSS framework that pro pre-built styles, components and a responsive grid

system to simplify web design.
- Bootstrap divides the page into 12 equal columns.
- You place content inside these columns and they automatically adjust based on screen sizes.
- Uses containers (fixed or fluid) → rows → columns.

Q4. Differentiate between tag, class and ID selectors.

	Syntax	Usage	Scope	Example
elector	tagname{}	Styles all elements of a given tag	Applies to all instances	p{color=blue;} →all turn blue
ector	classname{}	Styles elements with a specific class.	Can be reused multiple times.	highlight{ background: yellow;}
ctor	#idname{}	Styles one unique element	Should be unique for each page	#header{ font-size: 30px;}

Q5. Describe the three main ways to apply CSS to an HTML document.

ns 5 -

	CSS Type	Definition	Syntax eg	Use Case	Adv / Disadv
i)	Inline CSS	Applied directely inside an element's style attribute	Text	Quick fixes	✓ Fast for testing ✗ Hard to maintain
ii)	Internal CSS	Written in a <style> tag inside <head>	html <style>p {color:green}</style>	styling a single HTML page	✓ Easy for single page ✗ increases page size

External CSS	Written in a separate CSS file linked with (link)	Clink rel= "stylesheet" href = "style.css")	Large websites

Conclusion :-

Thus, implemented first laboratory assignment and le about responsive web design, page styling + inline, e and internal CSS and Bootstrap CSS.

✓

(8)
22/8/25

Full Stack Development
Assignment : 2

Aim :-

Develop a web application using javascript to implement sessions, cookies and DOM. Perform validations such as checking for emptiness, only numbers for phone number, special character requirement for password, regular expression for certain format of the fields etc. Use the mySQL database.

Objectives :-

i) To understand what form validation is.
ii) To learn basic functioning of DOM objects.
iii) To learn how to apply various techniques to implement it.

Theory :-

1) Explain the role of regular expressions. Why are they a suitable tool for validating data formats like a phone number or checking for the presence of specific character in a password?

Ans. i) Regular expressions (regex) define patterns to match text strings.
ii) They can check if input fits formats like phone numbers or passwords.
iii) Regex effectively verifies presence of digits, letters or special characters.

iv) This helps catch invalid input early, ensuring consistently.

v) Ideal for automating and speeding up data validation processes.

2) Explain the fundamental difference between a session and a cookie in the context of web application. How do they works together to maintain a user's state!

Ans.

Sessions	Cookies
i) Stored on the server side.	Stored on the client
ii) Holds user-specific data securely.	Holds small pieces of data (eg: session ID)
iii) More secure since data is not exposed.	Sent with every HTTP request to the server.
iv) Used to track user login state.	Identifies the session on the server.
v) Can store complex data.	Limited in size (~4KB)

3) What is the purpose of performing both client-side and server side validation? Describe a scenario where relying solely on client side validation could lead to a security vulnerability.

ns. i) Client-side validation gives instant feedback, improving UX and saving server resources.
ii) Server-side validation securely checks data to prevent manipulation or attacks.
iii) Relying only on client-side can be dangerous as it's easy to bypass.
iv) Attackers can send harmful inputs directly to the server without validation.
v) Server-side validation is essential for protecting the application's integrity.

4) Provide a simple example of how a JavaScript script can interact with the DOM to dynamically change the content of a web page after a user action, such as a form submission.

ns:
i) JavaScript listens to/for user actions, like submitting a form.
ii) It prevents the page from reloading with event.preventDefault()
iii) Reads input values ~~to~~ from form fields dynamically.
iv) Updates page elements (like paragraphs or divs) with new content.
v) This creates interactive, responsive web pages without refreshing.

~~Steps to Give the steps for connect~~

```
<!DOCTYPE html>
<head>
  <title> DOM Interaction Example </title>
</head>
<body>
<form id = "myForm">
  <input type = "text" id = "nameInput" placeholder = "E
    your name" required>
  <button type = "submit"> Submit </button>
</form>
<p id = "greeting"></p>

<script>
  document.getElementById('myForm').addEventListen
  'submit', function (event) {
      event.preventDefault();

      const name = document.getElementById('name
      .value;
      document.getElementById('greeting').textContent
      'Hello, ' + name + '!';
    });
</script>
</body>
</html>
```

5) Give the steps for connectivity from Frontend using HT
CSS JS to mySQL.

Ans: i) Frontend collects user data via forms or UI elements.

ii) JavaScript sends data to the backend server using

AJAX or fetch API.

iii) Backend processes the request and communicates with MySQL using drivers or ORM.

iv) Database returns results which backend sends back to Frontend.

v) Frontend updates the UI based on the response, showing results or status.

FAQs:-

1. Write 3 reasons why Form validations are important.

ans 1:- i) Prevent submission of incorrect or incomplete data.

ii) Enhance user experience by providing immediate feedback.

iii) Protect the backend from harmful or malicious inputs.

Q2. Give an example of how to modify an attribute value using DOM.

ans 2: document.getElementById('myImage').setAttribute('src', 'new-image.jpg');

This changes the "src" attribute of the image with ID "myImage" dynamically.

Q3. What are the different features of JavaScript?

ans 3.

i) Dynamic typing: Variables can hold any data type.

ii) Event-driven: Responds to user actions like clicks and key presses.

iii) Prototype-based inheritance: Supports object-oriented

programming via prototypes.

iv) Asynchronous programming: Supports callbacks, prom
and async/wait.

v) DOM manipulation: Can dynamically change HTML
CSS content.

vi) First-class functions: Functions are treated as
and can be passed around.

✱ Conclusion:-

Thus, we learnt that by implementing form validati
using JavaScript, sessions and cookies, we can en
both user experience and security by ensuring data
Combining client-side and server-side validations with
database connectivity is essential for robust web applica
development.

30/5/25

Assignment: 3

Name : Jayaram Patel
Roll no. : 48
PRN : 10322325 31
Batch : A2
TY - CSF

✱ Aim :-

Design an interactive front end application using React by implementing templating using components, States and Props, Class, Events. It must be reponsive to scale across different platforms.

✱ Objectives :-

To develop a responsive, iterative front-end application using React.js that effectively demonstrates the fundamental concepts of component-based architecture, state management, and event handling. The application will serve as a practical exercise in building a scalable user interface by implementing templating with components, managing dynamic ~~data~~ with states & props and handling user interactions with events, ensuring a seamless user experience across various devices & screen sizes.

Theory :-

Q1. Explain the role of State & Props in React. How they differ, and what is the primary purpose of in managing data flow within a component based application?

Ans 1. State :

State is the internal data of a component that can over time. It is mutable and managed within the comp using hooks like useState. When state changes, React automatically re-renders the component to reflect new data.

Props :

Props (short for Properties) are read-only values from a parent to a child component. They enable sharing and make components reusable.

Difference :

	State	Props
i)	Mutable (can change)	Immutable (read-
ii)	Managed by the component itself.	Ownership is passe from parent to c
iii)	Purpose is to handle internal, dynamic data.	Purpose is to pass between components.

Q2. What is a React component? Differentiate between a C component and a Functional component, and discuss the advantages of using a functional component with hooks lik useState and useEffect over a class component.

A React component is a reusable piece of UI that can manage its own logic and rendering.

- Class component: Defined using ES6 classes, manages state with this.state, and uses lifecycle methods like componentDidMount.

- Functional component: Defined as a simple Javascript function and uses Hooks (like useState, useEffect) for state and lifecycle management.

- Advantages of Functional Components:
i) Simpler and more readable syntax.
ii) No need for "this" keyword.
iii) Better performance and easier testing.
iv) Hooks provide flexible and reusable logic.

3. Describe the concept of "templating using components" in React. Why is this approach considered superior to traditional web development methods that rely on ~~static~~ monolithic HTML files?

In React, templating means building the UI using reusable components ~~that~~ represent different parts of the interface - such as buttons, forms or cards. Instead of writing one large HTML file, React applications are composed of modular and dynamic templates.

Advantages over Traditional HTML development:

i) Reusability: Components can be reused across multiple pages

or projects, reducing redundancy.

ii) Maintainability: Changes made in one component propagate throughout the app.

iii) Separation of Concerns: Each component handles its logic and UI, leading to cleaner architecture.

iv) Dynamic Rendering: Components can display data dyn using props and state.

v) Scalability: Large applications can be built effective composing smaller, testable components.

Q4. How do you handle events in React (eg: a button Provide a sample code snippet to demonstrate how an event handler is defined in a component and how can be used to update the component's state.

Ans4. In React, user events (like clicks or input change are handled using event handlers written in camelCa (eg: "onClick"). These handlers are functions that update the component's state using hooks like "useState eg:

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

function Counter() {
  const [count, setCount] = useState(0);
  const handleClick = () => setCount(count+1);

  return (
    <div>
      <h2> Count : {count} </h2>
      <button onClick = {handleClick}> Increase </b
```

(3) lollipops

```
    </div>
  );
}
export default Counter;
```

- Explanation:
→ useState (0) initializes count with 0.
→ The handleClick function updates the state when the button is clicked.
→ React re-renders the component to show the new count.

5. What is responsive web design, and why is it crucial for modern applications? Describe how you would implement a responsive design in a React application using CSS media queries or a CSS-in-JSS library?

Responsive Web Design ensures a website automatically adjusts its layout to look good on different screen sizes (mobile, tablet, desktop).

- Importance:
i) Improves user experience on all devices.
ii) Enhances accessibility and SEO.
iii) Reduces the need for separate mobile and desktop versions

- Implementation methods:

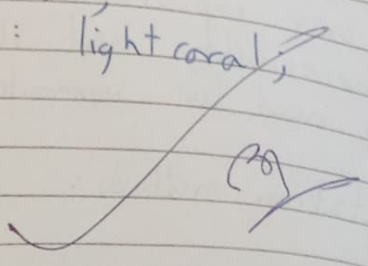
(i) Using CSS Media Queries

ON NEXT PAGE

```
.container {
    padding : 20px;
}

@media (max-width : 600px){
    .container{
        padding : 10px;
        font-size : 14px;
    }
}
```

(ii) Using CSS-in-JS

```
import styled from 'styled-components';

const Box = styled.div`
    width : 80%;
    margin : auto;
    background : lightblue;

    @media (max-width : 600px){
        width : 100%;
        background : lightcoral;
    }
`;
```



Assignment : 4

Aim :-

Enhance web page developed in earlier assignment by rendering Lists and Portals, Error Handling, Routers and style with React CSS also make it a responsive design to scale well across PC, tablet and Mobile Phone.

Objectives :-

Enhance User Interface and Experience.
Improve Application Robustness and Navigation.

Theory :-

How do Lists and Keys work in React?

1. In React, lists are created by iterating over arrays using the .map() function to render multiple components dynamically. Each element in a list must have a unique key prop to help React identify which items have changed, been added or removed. This improves rendering effeciency and performance.

2. What is a React Portal and when would you use one?

2. A React Portal allows components to render children into a different part of the DOM outside the main component hierarchy. It's mainly used for rendering

modals, techniques, tooltips or pop-ups where
need to visually break out of their parent cont

Q3. Discuss the importance of Error Boundaries in Re
Ans 3. Error Boundaries are React components that catch
JavaScript errors in their child components and di
a fallback UI instead of crashing the entire app.
They enhance application stability and user experien
by preventing unexpected blank screens or crashes.

Q4. How does React Router enable Single Page Applicat
(SPA) functionality?
Ans 4. React Router enables navigation between different vie
of a React app without reloading the page. It ma
URL changes and renders components dynamically,
giving the feel of multiple pages while still being a
single-page application.

Q5. Explain the different ways to style a React App
Ans 5. React applications can be styled using CSS style
inline styles, CSS Modules, Styled Components or
CSS frameworks like Bootstrap and Tailwind CSS.
Each method offers different levels of scoping, reus
and customization for consistent UI design.

☆ Conclusion:-

Through this assignment, we implemented React fe
to enhance functionality and responsiveness, improvi
user experience.

For example :

- Inline styling :

```
<h1 style = {{ color : "blue", fontSize : "24px" }}
> Hello World </h1>
```

This method is simple and component - specific.

- CSS Stylesheet :

You can create a file like App.css and import it in your component.

```
import './App.css';
```

- CSS Modules :

Provides locally scoped styles to avoid name conflicts.

```
import styles from './Button.module.css';
<button className = {style.primaryBtn} >Click Me</button>
```

- Styled Components :

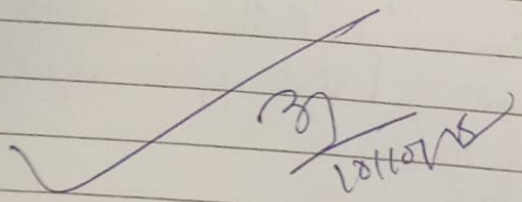
Uses JavaScript to write CSS, allowing dynamic styling based on props.

```
const Button = styled.button `background-color : ${props
=> props.primary ? "blue" : "gray"};`;
```

Frameworks like Bootstrap or Tailwind CSS help in faster development by providing pre-defined utility classes for responsive and attractive layouts.

Conclusion :-

Through this assignment, we implemented React features to enhance functionality and responsiveness improving user experience.



Assignment : 5

Aim :-

Develop a responsive web design using Express Framework to perform CRUD operations and deploy with Node JS. Use Mongo DB.

Objectives :-

- Develop a Full-Stack Web Application
- Demonstrate Backend Development and Development Proficiency.

Theory :-

1. What is the role of Express.js as a web framework for Node.js?

Ans1. Express.js is a lightweight and flexible web framework built on top of Node.js that simplifies server-side development. It provides features for routing, middleware handling, and request/response management. Express helps developers build scalable and maintainable web applications quickly and efficiently.

2. Explain the concept of CRUD operations in the context of a web application.

Ans2. CRUD stands for Create, Read, Update, Delete, which are the basic operations performed on data in a

web application. These operations allow users to manage resources effectively, such as adding, view editing and removing data entries stored in a da

Q3. Why is MongoDB a suitable choice for this projec

Ans3. MongoDB is a NoSQL database that stores data flexible JSON-like documents, making it ideal for dynamic web applications. It offers high performa easy scalability and seamless integration with Nod and Express, making it well-suited for full stack development.

Q4. What steps are involved in deploying a Node.js an Express application?

Ans4. Development typically involves preparing the project f production, setting environment variables, connecting a database, and hosting on a platform like Rend Vercel or Heroku. You also need to ensure that server listens on the correct port and handles incoming requests securely.

Conclusion:-

In this experiment, we developed a full-stack app using Node.js, Express and MongoDB. This helped understand backend development, database integration deployment for responsive, data-driven web system

4th Answer continued :

Additional steps include :

i) Installing dependencies using npm install to ensure all required packages are available.

ii) Building the frontend if you are using frameworks like React - for example, running npm run build to create a production build.

iii) Setting environment variables for sensitive data such as database URLs or API keys.

iv) Connecting to a database like MongoDB or MySQL using libraries such as Mongoose or Sequelize.

v) Testing locally to confirm that both backend and frontend work as expected before deployment.

vi) Deploying on cloud platforms such as Render, Vercel, Heroku, or AWS EC2, where you push your code using Git.

vii) Monitoring performance and using logging tools like PM2 or Winston to manage uptime and detect errors.