Full Stack Development
Assignment-1

- **Aim:** Develop responsive web designs using HTML5, containing a form. Style the pages using CSS. use of tag selector, class selector and id selectors. use inline, Internal and External CSS. Apply Bootstrap CSS.

- **Objectives:** i) To understand HTML tags.
  ii) To learn Styling of web pages using CSS.
  iii) To learn Bootstrap Front End Framework.

- **Theory:**

i) Responsive web design: (RWD)

- It is an approach to web design that makes web pages under well on a variety of ~~design~~ devices and window as screen size. This layout updates automatically to the screen size orientation and platform, ensuring usability & asthetics across desktop, tables and smartphones.
- Primary Goal: Provide an optimal meaning and interaction experience. easy navigation with minimal resizing, panning and scrolling - across wide range of devices.

ii) Role of <meta name = "view point">.

- <meta name = "view point"> tag tells the browser how to control the page dimensions & scaling.
- Role: it sets the visible area of a web page to match the devices screen, width and initial zoom level.

- Essential for RWD: Without this tag page, mobile Browser assume a default viewpoint width, which cause the page to appear zoomed out and not scale to small screen.

iii) Bootstrap & Grid System
- bootstrap helps create responsive website that adapt to devices (mobile, table, desktop).
- Uses a 12-column grid system where content is placed inside row and column element.
- the grid adapts using breakpoints with prefixes:
  - col- extra small (<576 px).
  - col-sm- small (≥576 px)
  - col-md- medium (≥ 768px).
  - col-lg- large (≥992 px)
  - col-xl- Extra large (≥1200 px).

iv) Difference between Tag, (lass & ID selectors)

	Selectors	Symbol	Purpose	Example.
→	Tag	none	Styles all <P> tags	P{Color = red}.
→	Class	.	Styles all elements with given class.	.highlight{background = green}.
→	ID	#	Styles one specific Element with an id.	#main{font Size = 20px}.

v) There ways to apply Css.
① Inline Css: written directly in HTML elements style
 Hello
② Internal Css = Written inside a <style> tag in (head)
(style)
p{color : green}.
(/style).

Name: Himanshu Kay Prakash.
PRN: 10322233853.
Roll No.: 25. A2 Batch

Assignment: 2

- **Aim:** Develop a web application using javascript to implement session, cookies, DOM. Perform validations such as checking for phone no, special character requirement for password, regular expression for certain format of the fields etc. Use the MYSQL database.

- **Objective:** (1.) To understand what form of validation
  (2) To learn basic functioning of DOM objectives.
  (3) To learn how to apply various techniques to implement it.

- **Theory:**

→ Regular Expressions are patterns used to match character combinations in strings. They provide a powerful & concise way to validate input formats by defining a set of rules.

- **Role:** They help check if an input string meets a specific pattern.

- **Stability:** They can enforce exact format rules like no. of digits, presence of certain characters,
  For Phone numbers, regex can ensure the no. only contain digits & ~~presence of certain charact~~ has specific length.

For password it can check for upper and lower case.

(02) Difference b/w session & cookie.

Aspect	Cookies	Session
Storage location	Stored on the client side	Stored on the side
Data size limit	limited	No strict size.
Life time	Can persist beyond browser session	Last unite log outs.
Security	less secure	More secure.

→ How they work together.

- When a user log in the server creats a session store user interface.
- The server sends this cookie back with session id
- The client sends this cookie.

(03) What are the different features of Javascript.

→
- Client side scripting. JS runs in the ~~features~~ browser. allowing web pages to be interactive by responding to user events like clicks, form submissions, & ~~move~~ movements.
- Event-Driven programming: JS supports events handling, so it can execute code in response to user actions or browser events.

(iii) External CSS: Stored in a CSS file and linked in <link rel = 'style sheet' hre / = style.CSS' >

- PROBLEM STATEMENT:
  Personal Portfolio website (Roll no. 13 to 24).

- CONCLUSION:

→ In this assignment, I learned how to build responsive web pages using HTML5, CSS and Bootstrap. I understand the use of different CSS selectors, the ways to apply CSS selectors, the ways to apply CSS, and how the viewpoint tag with bootstraps grid system ensure layouts adapt to all screen sizes.

✓

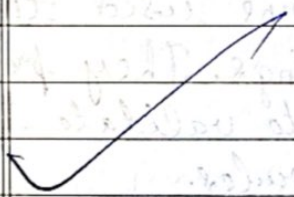
M
5/9/25
Late

(iii) External CSS: Stored in a CSS file and linked in <link rel = 'style sheet' hre/ = style.CSS' >

- PROBLEM STATEMENT:
Personal Portfolio website (Roll no. 13 to 24).

- CONCLUSION:

→ In this assignment, I learned how to build responsive web pages using HTML5, CSS and Bootstrap. I understand the use of different CSS selectors, the ways to apply CSS selectors, the ways to apply CSS, and how the viewpoint tag with bootstraps grid system ensure layouts adapt to all screen sizes.



CM 5/9/25 Late

- Dynamic Typing - Variables in JS can hold any type of data & their type can change at runtime, making the language flexible.
- Object Orient & functional : JS supports objects & prototype based inheritence, as well as functions as first class citizens.
- DOM manipulation: JS can access & modify HTML DOM, allowing dynamic content updates
- Asychronous Programming: Features like callback, promises, and async/await allow JS to perform tasks asynchronously, such as fetching data from a server without freezing the UI.

7/10/25

Page No.
Date

Full Stack Development
Assignment - 3

- **Aim:** Design an interactive front end app using React by implementing templating using components, states and props, class Events. It must be responsive to scale across different platforms.

- **Objectives:** To develop a responsive, interactive front end application using React Js that effectively demonstrates the fundamental concepts of concepts of Components ~~of Co~~ 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 and props, and handling user interactions with events, ensuring a seamless user experience across various devices and screen sizes.

- **Theory:**

⇒ Role of state and props in React state:

- State is an object managed within a component.

- It represents dynamic, mutable data that can change over time.
- When the states changes, reacts automatically re-renders the component to reflect the new data.
- Eg. Tracking whether a button is clicked or not.

⇒ Props:
These are inputs to a component they are immutable & passed from parent to child.

⇒ Difference:
State: Managed internally by components
Props: passed external from parent.

⇒ React component class v/s functional
• React Component
- A reusable building block in react that defines part of the UI.
- Each component can accept props, manages state, and return JSX to describe how the UI should look.

• Class component
- Define using ES6 classes.
- Uses this state and lifecycle methods.
- More verbose and harder to manage in large projects

• functional component.
- Defined as JS component that return JSX.
- Initially stateless. But with hooks they can manage state & side effects.
- Shorter, cleaner syntax.

→ Templating using components in React.

• Concept
- Instead of writing one large HTML file, React allows developer to break the UI into smaller reusable components
- Each components acts like "template".

• Why superior.
- Reusability: components can be reused across the application.
- Modularity: Each components manages it's own logic & styling, making the code easier to maintain.
- Dynamic Rendering: Components can accept prop & state, making them highly flexible.

⇒ Handling user Events in React.
- Events are handled using ~~and~~ and by passing functions as events handler

```
import React, {use state} from 'react';
function Counter() {
    Const [count, set Ecount] = Use. state(0);
    Const handClick = () => {
        . Set Count (count + 1);
    };
    return (
     <div>
      <P> Current Count = {count} </P>.
      {button on click={handle click}> Increase
      </div>.                                </button}
    );
}.
enport default Counter;
```

⇒ Responsive web Design.
Responsive design web design ensures that web apps adjust their layout and appearance to different screen sizes.

- Why crucial:
  - Improves user enprience.
  - makes application mobile - friendly.
  - Reduces the need to build seperate apps for different devices.

Marks

Full Stack Development

Lab Assignment: 04

- **Aim:** Enhance web pages developed in earlier assignment by rendering lists & portals, Error Handling, routers & style with react CSS also make it a responsive design to scale well across PC, tablet & mobile phones.

- **Objectives:** Enhances user Interface & Experience Improve Application Robustness & navigation.

- **Theory:**

1. How do lists & keys work in React?

→ List render multiple items dynamically using map().

→ Keys are unique identifiers for ~~first~~ list items, helping react efficiently update & rerender only changed elements.

→ Without proper Keys, performance drops & unexpected UI bugs may occur.

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

→ A react Portal allows rendering a child component into a DOM node outside its parent hierarchy.

→ useful for modals, tooltips, popups & overlays where you need separation from parent styles or positioning.

3. Discuss the Importance of Error Boundaries in React.

→ Error Boundaries catch JavaScript Errors in components during rendering, lifecycle methods, or constructors.

→ Prevents the Entire app from crashing by displaying fallback UI.

→ Crucial for improving user Experience & debugging in production.

4.) How does router ~~enable~~ enable SPA functionality?

→ React-Router maps URLs to components without full-page reloads.

→ Uses client-side routing with history API for ~~sensors~~ smooth navigation.

→ Provides <route>, <Link>, <switch> for seamless SPA transition.

5) Explain the different ways to style a React application.

→ CSS style sheets - Import standard CSS files.

→ Inline styles - use style={{}} with javascript objects.

→ CSS modules - Scoped CSS with automatic class name mapping.

→ Styled-component (CSS in JS) - Write CSS inside JS using tagged templates.

→ Frameworks - Utility first CSS framework for rapid UI design.

- Problem statement: Create a dynamic user dashboard with multiple sections that can be navigated using tabs.

* Conclusion: This assignment concludes with an Enhanced react application using hists portals, error handling, routers & responsive CSS, delivering a robust, user-friendly & scalable Interface across PC, table & mobile devices.

7/11/25

Full Stack Development
Lab Assignment : 05

- **Aim:** Develop a responsive web design using Express framework to perform CRUD operations & develop with node JS. use MongoDB.

- **Objectives:** Develop a full stack web application.

→ Demonstrate backend Development & Development Proficiency.

Theory:

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

→ Express.js is a lightweight, flexible web framework built on Node.js.

→ It simplifies handling HTTP requests, responsive & routing.

→ Provides middleware support for authentication, sessions & Error handling.

→ Speeds up API & web app development with minimal boiler plate code.

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

→ CRUD stands for Create, Read, Update & Delete.

(i.) Create – Add new records.

(ii.) Read – Fetch/display data.

(iii.) Update – modify existing data.

(iv.) Delete – remove data.

- CRUD Ensures complete life cycle management of application data.

3) Why is Mongo DB a suitable choice for this project?

→ Mongo DB is a NOSQL, document-oriented db.

→ Stores data in flexible JSON-like documents, making it deal for dynamic applications.

→ Scales easily for large datasets & distributed systems.

→ integrates seamlessly with Node.js & Express Mongoose.

→ Schema flexibility supports rapid development & iteration.

4) What steps are involved in deploying a node.js & Express application?

1.) Prepare Application

2.) Choose hosting Platform

3.) Setup server

4.) Upload Code

5.) Install Dependencies

6.) Configure Process Manager.

7.) Reverse Proxy.

8.) Monitor & Scale.

- Problem statement: A recipe sharing platform.

- Conclusion: This assignment concludes with a full-stack web application using Express, node.js & mongoDB, successfully performing CRUD operations & deployment, ensuring responsive design, backend proficiency & seamless functionality across devices.