Name- Yashika Raj
PRN - 1032232524
Roll No. - 47, Batch A2
TY Btech CSF

FSD

Assignment - 01

Page No.
Date

Aim:

Develop responsive web design using HTML 5, containing a form. Style the pages using CSS, use of tag selector, class selector and id selector use Inline, Internal and External CSS, Apply Bootstrap CSS.

Objectives:

1. To understand HTML tags
2. To learn the Styling of web Pages using CSS.
3. To learn Bootstrap Front End Framework.

Theory:

1. Define Responsive web Design (RWD). what is its Primary goal?

Ans- Responsive web design (RWD) is a web development approach that creates websites able to adapt and respond to different screen sizes and devices (like phones, tablets and desktops).

Primary goal:

To ensure a seamless user experience by making the layout, content and images, adjust automatically to fit the screen, without requiring users to zoom or scroll horizontally.

2. Explain the role of the <meta name = "viewport" ...> tag. why is this tag essential for RWD?

Ans. The <meta name = "viewport"> tag tells the browser how to control the page's dimensions and scaling.

For example:

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

Without this tag, mobile browsers display web pages with a desktop width viewport, making the content too small to read. This tag enables proper scaling and layout adjustment for mobile devices, which is a core requirement for responsive design

Page No.
Date

3. How does Bootstrap assist in creating a responsive layout ?

Ans- Bootstrap is a popular front end framework that makes it easy to build responsive websites using predesigned components, and flexible grid system.

Grid system, Explanation:

Bootstrap's grid system divides the page into 12 columns. You can combine these columns in different ways using classes like col-, col-sm-, col-md-, col-lg- and col-xl- to control how much space each element takes at various screen sizes.

- Col-sm-6 means the element takes 6 columns (half width) on small screens and above.
- The grid automatically stacks or resizes columns to fit different screens, ensuring responsiveness.

4. Differentiate between Tag, class and ID selectors.

Ans. Tag selector: Targets all elements of a specific type.

Example: p { color: blue; } - changes colour of all <p> tags.

Class selector: Targets elements that share the same class attribute

Example: .box { padding: 10px; } - applies styles to any element with class = "box".

ID selector: Targets a specific element with a unique ID.

Example: #header { background: gray; } - applies styles to the element with id = "header".

5. Describe the three main ways to Apply CSS to an HTML document.

Ans. Inline CSS:

Applied directly to an element using the style attribute.

Internal CSS:

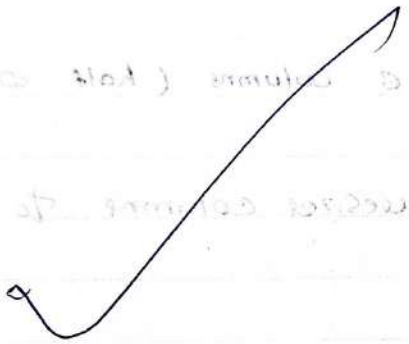
Written inside a <style> tag in the <head> section of HTML

External CSS:

Linked using a <link> tag to a separate .css file.

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 and how the view port tag with bootstrap grid system layout adapt to all screen sizes.



24/8/22

Name- Yashika Raj
PRN - 1032232524
Roll No. - 47
TY Btech CSF, Batch A2

FSD

Assignment - 02

Page No.
Date

Aim:

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

Objectives:

1. To understand what form validation is
2. To learn basic functioning of DOM objects.
3. To learn how to apply various techniques to implement it.

Theory:

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

Ans- Role of Regular Expression:

- Regex are patterns used to match and validate text
- Suitable for checking data formats (e.g. phone numbers must be 10 digits)
- useful in password validation (uppercase, digit, special characters).

Therefore, regular expressions are suitable for validation tasks because they:

- Define precise rule for input formats
- Quickly detect invalid data before processing
- Improve reliability and consistency of user inputs.

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

Ans- Cookies:

- Small data stored on the client's browser

- Holds user preferences, IDs, tokens.
- Can persist after browser is closed

Sessions:

- Data stored on the server (user-specific info).
- A unique session ID is generated and stored in a cookie on the client side.
- Ends when the user logs out or the session expires.

How they work together:

- When you log in → Server creates a session (with user data).
- The session ID is sent back to the browser → stored in a cookie.
- On the next requests, browser sends the cookie → server verifies session → user stays logged in

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.

Ans. Purpose of client-side and server side validation:

▶ Client-side validation (JavaScript in browser):

- Improves user experience by catching errors early
- Reduces unnecessary server requests.

▶ Server-side validation (PHP, Node.js, Python, etc.):

- Ensures security and data integrity.
- Cannot be bypassed since it runs on the server.

Scenario of relying only on client side validation (vulnerability):

Suppose you only check password strength in JavaScript.

- A malicious user can disable JavaScript or use developer tools to bypass validation → sending weak or malicious data to the server.
- That's why server-side validation is mandatory.

```
static getDerivedStateFromError() {
    return { hasError: true };
}
render() {
if (this.state.hasError) return <h2>Something went
wrong!</h2>;
return this.props.children;
}
}
```

4. How does React Router enable Single Page Application (SPA) functionality?

Ans. React Router allows navigation between pages without refreshing the browser.

- It turns a react app into a single page application (SPA) by dynamically updating the view when the URL changes.

Benefits:

faster navigation

Smooth user experience

Route based rendering

Example: import { BrowserRouter, Route, Routes }
from "react-router-dom";

```
<BrowserRouter>
<Routes>
<Route path="/" element={<Home />} />
<Route path="/about" element={<About />} />
</Routes>
</BrowserRouter>.
```

5. Explain the different ways to style a react application.

Ans. ⓐ CSS Stylesheets
Traditional CSS files imported into components

ⓑ Inline styles
Directly style elements using JavaScript objects

ⓒ CSS modules
Scoped CSS that avoids naming conflicts

ⓓ CSS in JS libraries
Styled components, Emotion (Styles written in JS)

ⓔ UI frameworks
BootStrap, material-UI, Tailwind CSS for pre built responsive designs

Conclusion:

By using lists and keys, portals, error Boundaries, React Router and different styling techniques, React applications become:

- more dynamic (lists)
- more flexible (portals)
- more user-friendly (SPA navigation)
- more reliable (error boundaries)
- more visually appealing and responsive (styling methods)

These enhancements collectively improve both user experience and application robustness.

26/9/25

Name - Yashvee Roy
PRN - 1032232524
Roll no - 47
TY B.tech CSE
Batch - A2

FSD

Assignment - 03

Aim

Design an interactive front-end application 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 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 and props and handling user interactions with events, ensuring a seamless user experience across various devices and screen sizes

Theory

1. Explain the role of State and props in React. How do they differ and what is the primary purpose of each in managing data flow within a component-based application.

Ans - State: State represents mutable data that is managed inside a component

- It allows components to create and update their own data over time in response to user actions, network responses or other events.
- State updates cause the component to re-render and reflect the changes in the UI.

Example - A counter value that increases when the user click a button.

Props: Props (short for properties) are immutable data passed from a parent component to a child component.

- They are used for data flow and make components reusable by supplying external values.
- Example - passing a username from a parent component to a child component for display.

- Difference:

Aspect	State	Props
Ownership	Managed inside a component	passed from parent to child
Mutability	Mutable (can be updated)	Immutable (read-only)
Purpose	Handles dynamic, local data	Passes data / config between components

2. What is React component ? Differentiate between a class Component and a functional Component and discuss the advantages of using a functional component with hooks like useState and useEffect over a class component.

Ans- A React component is a building block of a React application that represents part of the UI.

Components are reusable, independent and modular class component.

- Written using ES6 classes
- Uses this.state and this.setState() for state
- Uses lifecycle methods like componentDidMount() management.

```
Class Welcome extends React.component {
  Constructor (props) {
    Super (props);
    this.state = {count : 0};
  }
  Hendey () {
```

```
        return <h1> Hello, {this.props.name} ! </h1>;
    }
}
```

Functional Component:

- written as simple JavaScript functions.
- use Hooks (like useState, useEffect) for state and lifecycle management.

```
function Welcome ({name}) {
    return <h1> Hello, {name} ! </h1>;
}
```

Advantages of functional components with Hooks:

- Cleaner and more concise syntax.
- No need for this keyword.
- Better performance optimization
- Hooks provide powerful features like state (useState) and side effects (useEffect).

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

Ans- Concept: Instead of writing one long HTML file, React breaks the UI into small reusable components (e.g. Navbar, footer, Profile Card).

- Components act as templates that can accept props and display dynamic data.

Why its superior to traditional HTML:

- Promotes reusability and reduces code duplication
- Easier maintenance since UI is modular
- Encourages scalability for large application.
- facilities dynamic updates, with state management.

4. How do you handle user events in React (eg. a button click)? provide a simple code snipper to demonstrate how an event handler is defined in a component and how it can be used to update the component's state?

Ans- Events in React are similar to JavaScript DOM events but use camel case syntax.

- we can attach an event handler function to an element and update state.

Example: Button click counter.

```
import React, { use State } from "react";
function counter () {
Const [ Count , Set Count ] = use State (0);
Const handle click = () => {
    Set count ( Count + 1 );
};
Return (
<div>
<p> you clicked { Count } times </p>
<button on click = { handle click } > click me </button>
</div>
);
}
Export default Counter;
```

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 JS library.

Definition:

Responsive web design (RWD) ensures a website adapts to different

Screen sizes and devices (desktop, tablet, mobile).

Importance:
Improves user experience across all platforms.
Essential for accessibility and SEO.
Reduces the need for multiple device-specific versions.
Implementing Responsiveness in React.

① CSS media queries.

```
Container {
  width: 100%;
  padding: 20px;
}
@ media (max-width: 768 px) {
  . Container {
    padding: 10 px;
    front size: 14px;
  }
}
```

② CSS in JS (Styled components / Emotion)

```
import Styled from "Styled-components";
Const Box = Styled. div
  width: 100%;
  padding: 20px;
@ media (max-width: 768 px) {
    padding: 10 px;
    front size: 14px;
  }
;
```

③ UI libraries (Bootstrap, material-ui, Tailwind CSS)
these libraries provide build in responsive classes.

Conclusion:
This exercise demonstrates the core principles of React development
Components make the UI modular and reusable
State and props enable efficient data flow and dynamic updates
Event Handling makes applications interactive
Responsive Design ensures accessibility across devices.

26/9/25

Name - Yashika Raj
PRN - 1032232524
Roll No. - 47
TY Btech CSF
Batch - A2

FSD

Assignment - 04

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:

1. How do lists and Keys work in React?

Ans- Lists:

In React, lists are used to render multiple items dynamically from an array using the map() function.

Keys:

Each list item must have a unique key prop, which helps React identify which items have changed, been added or removed.

Purpose: Keys improve rendering performance and avoids bugs during Re-renders.

Example:

```
Const items = ["Apple", "Banana", "Cherry"];
<ul>
{items.map(fruit, index) => (
<li key = {index}> {fruit} </li>
))}
</ul>
```

2. What is a React portal and when could you use one?

Ans. A React portal allows you to render a child component into a DOM node outside the main parent hierarchy (root).

Use cases:

- modals / popups
- Tooltips
- floating elements that need to visually "escape" container boundaries.

Example:

```
React DOM. create Portal (
  < modal content />,
  document. get Element By Id ("modal- root")
);
```

3. Discuss the importance of Error Boundaries in React.

Ans. Definition:

Error Boundaries are React Components that catch Javascript errors in their child components and display a fallback UI, instead of crashing the whole app.

Importance:

Improves robustness by preventing app wide crashes

provides better user experience with custom error messages.

Example:

```
class Error Boundary extends React. Component {
    Construction ( props) {
    Super (props);
    this. State = { has Error : false };
  }
```

Page No.
Date

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.

Ans- JavaScript Example - Interacting with DOM

Here's a simple example where a user submits a form and the page updates dynamically:

```
<!DOCTYPE html>
<html>
<head>
  <title> DOM Example </title>
</head>
<body>
  <input type = "text" id = "nameInput" placeholder = "Enter your name">
  <button onclick = "updateMessage()"> Submit </button>

  <p id = "message"></p>

  <Script>
    function updateMessage() {
      let name = document.getElementById("nameInput").value;
      document.getElementById("message").innerText = "Hello, " + name + "!";
    }
  </Script>
</body>
</html>
```

When the user enters a name and clicks "Submit", the <p> content updates dynamically.

5) Give the steps for connectivity from front end using HTML CSS JS to mysql.

Ans:- a. User fills form (HTML / JS).

b. Javascript sends data to backend (via fetch / AJAX).

c. Backend (PHP / Node.js / Python) connects to MYSQL.
d. SQL Queries run on database.
e. Backend sends response → JS updates webpage.

FAQs:

1) Write 3 reasons why Form Validations are important.

Ans- a. Accuracy:
Ensures users enter correct and properly formatted data.
b. Security:
Prevents malicious inputs like SQL injection or XSS.
c. User Experience:
Gives instant feedback and reduces errors in submission.

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

Ans-
```
<img id = "myImg" src = "old.jpg">

<Script>
    document.getElementById("myImg").SetAttribute("src","new.jpg");
</Script>
```
This changes the image source from old.jpg to new.jpg using DOM.

3) What are the different features of JavaScript?

Ans- Light-weight and Interpreted, Event-driven and interactive, Object-oriented, Cross-platform, Rich-libraries / API.

Conclusion:

In Full Stack development, Validations ensure secure data, Sessions and Cookies manage user state, and Javascript with DOM makes web pages dynamic, while backend connectivity with databases completes the application.

Name- Yashika Roy
PRN - 1032232524
Roll No. - 47
Btech - CSF - TY
Batch - A2

FSD

Assignment - 05

Aim:
Develop a responsive web design using Express framework to perform CRUD operations and deploy with Node.Js use MongoDB

Objectives:
Develop a full stack web Application.
Demonstrate Backend Development and Deployment proficiency.

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

Ans. Express.Js is a minimal and flexible web framework built on top of Node.Js. It provides powerful features such as routing, middleware support, request and response handling and template rendering which makes backend development faster and easier instead of writing complex server code with just Node.js, developers use Express.js to build RESTful APIs, full stack applications and scalable web services with less efforts and more structure.

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

Ans- CRUD stands for Create, Read, Update and Delete which are the four basic operations needed to manage data in any application.

- Create → Add new data
- Read → Retrieve existing data
- Update → Modify existing data.
- Delete → Remove data.

In a web application CRUD is usually implemented through API endpoints that interact with the database. These operations form the backbone of most dynamic application.

3. Why is MongoDB suitable choice for this project?

Ans- MongoDB is a document-oriented NoSQL database that stores data is JSON like format. This makes it very natural to use with JavaScript and Node.js. It's flexible schema allows developers to store different types of data without needing a fixed structure which speeds up development. MongoDB also supports scalability, high performance and easy integration with Express.js through libraries like MongoDB. This makes it a perfect choice for modern web applications where data requirements may change frequently.

4. What steps are involved in deploying a Node.js and express application?

Ans- Deploying a Node.js and Express application typically involves.

1) Develop locally →
Build your app with express routes and MongoDB integration

2) Version control →
Push your code to Github or another repository.

3) Choose hosting →
use platforms like heroku, render Aws or Digital ocean

4) Install Dependencies →
Run npm install on the server

5) Configure Environment Variables →
Set up MongoDB connection string, port, etc

6) Run App →
Start using node server.js or a process manager like PM2.

7) Test Deployment →
Ensure routes and database connections work correctly online.

Conclusion:
Express.js plays a crucial role in simplifying server-side development, while CRUD operations form the foundation of dynamic applications MongoDB is well-being suited due to its flexibility. Json Based storage and scalability Deploying an Express app involves preparing the code, hosting it on server, configuring the database and running it efficiently together these components enable the creation of a robust, responsive and fully deployable full stack application.

25/9/25