Name :- Sushant Kumar
Class :- TY CSE CSF
PRN :- 10322231329 Roll No :- 30
Batch :- A1

DOMS | Page No. 1
Date 6 / 08 / 2025

FSD Laboratory Assignment No. 1

- **Aim :-** Develop responsive web design using HTML5, 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 :-** i.) to understand HTML tags
ii.) to learn the styling of web pages using CSS
iii.) to learn Bootstrap Front End Framework

- **Theory :-**

Q1.) Define Responsive Web Design (RWD). What is its primary goal ?

Ans - Responsive Web design (RWD) is an approach to web design that makes web pages render well on a variety of devices and screen size. A responsive site fluidly changes its layout - resizing, hiding, shrinking or enlarging content - to adapt to the viewing environment. This ensures that whether a user is on mobile phone, tablet, laptop or large desktop monitor, the website is easy to read and navigate without excessive resizing, panning or scrolling.

The primary goal of RWD is to provide an optimal user experience (UX) for everyone regardless of device they use. It aims to make web content accessible and functional across the entire spectrum of digital devices, improving usability and user retention.

Q2.) Explain the role of <meta name = "viewport" ... > tag. Why is this tag essential for RWD ?

Ans → It an HTML element placed in the <head> section of a page that gives the browser instructions on how to control the page's dimensions and scaling. Its standard implementation is as -

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

where:
width = device-width, tells browser to set the width of viewport to the physical of the device's screen. &
initial-scale = 1.0, sets the initial zoom level of the page to 100% (when it first loaded).

This tag is foundational step for building a responsive site. mobile browsers will assume page is non-responsive, fixed-width desktop so display it, they render the page in a wide virtual viewport (eg. 980px) and it down to fit the small screen. This makes text and images illegible. tag prevents this behaviour, ensuring the browser uses actual device width allowing responsive CSS techniques to correctly apply styles based on the screen size.

93.) How does Bootstrap assist in creating a responsive layout? Discuss the concept grid system and how it adapts to different screen sizes.

Ans- Bootstrap is a popular front-end framework that significantly simplifies the of responsive websites. It provides a collection of pre-written HTML, CSS, javascript components, with its most powerful feature being responsive grid

Bootstrap grid system divides the horizontal space of page into 12 columns. Where we can place content inside these columns, we don't create grid ourselves use Bootstraps predefined CSS classes to tell our content how many of columns it should span.

Grid system is responsive because it has different classes for different "breakpoints", we can assign multiple classes to an element to define its on different screens. for eg. <div class="col-lg-4 col-md-6 on large (lg) screen, this div will take up 4 columns (one-third of on medium (md) screen, it will take up 6 columns (half of screen spac on small (sm) screen, it takes up 12 columns, causing columns to stack By these simple classes, developers quickly build complex layouts that reflow and adapt, ensuring content is well-organized and readable

94.) Different

Ans → Feature
- Syntax
- Scope
- Busab
- Specifi
- Prima

95.) Ex

Ans → i)

DOMS Page No. 3
Date / /

Q4.) Differentiate between Tag, Class and ID selectors.

Ans →

Feature	Tag Selector	Class selector	ID selector
• Syntax	elementName (eg h2)	.className (eg. .btn-primary)	#id name (eg. #main-nav)
• Scope	Selects all HTML elements of a certain type (eg. all h2 elements).	Selects all elements that share the same class attribute	Selects only one unique element that has a specific id attribute.
• Reusability	Affects all matching tags on a page	Can be used on many different element on page	must be unique and can only be used once per page.
• Specificity	Low : easily overridden	Medium : overrides tag	High : overrides both selectors
• Primary use	Applying general, base styles to all instances of an element	Styling a group of similar elements that should look same (most-common selector)	Targeting a single, specific structural element on a page - like header or footer

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

Ans → i) Inline CSS -

Method : Styles are written directly inside an HTML tag using the style attribute

eg : This para is styled inline .

scope : It is very specific and affects only one element, but generally avoided as it mixes content (HTML) with presentation (CSS) make code harder to read and maintain

ii) Internal css :-

- Method : CSS rules are placed inside a <style> element, which is typically placed within the <head> section of the HTML document.

eg: <head> <style>
body { background-colour : #f4f4f4; }
p { colour : #333; } </style> </head>

- Scope : keeps styles for a single page in one place useful for unique styles or styles unique to one specific page ; that won't affect any other

iii) External css :-

- Method : All css rules are placed in a separate file with a .css extension. This file is linked to one or more HTML pages using the <link> tag in the <head> section.
- Example : <link rel = "stylesheet" href = "css/style.css">
- Scope : Best practice, most widely used making code clean and easier to manage, can be used to style an entire website ensuring consistent look and feel.

? Problem Statement :- 3 (Roll Number 25 to 36)

Conclusion

- We learned about various HTML tags their usage, syntax and usage. Know about bootstrap, various methods of CSS styling and how to create a responsive web design using the concepts used.

12/8/25 Good

Shubhant Kumar
B.Tech CSE CSF
322/329 Roll No:-30

Gr. 5

ESD Laboratory Assignment - 02

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 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 characters in a password?

Ans→ Regular expressions (regex) are powerful tools for pattern matching and validation of text input. They define specific sequences and rules that input data must follow; this makes them ideal for checking data formats, such as phone numbers, email addresses or password strength.

They are suitable at :-
- enforcing exact patterns (eg. fixed number of digits, presence of symbols)
- check complex rules like "at least one uppercase letter, one digit and one special character within a specified length range".
- To quickly scan strings without writing verbose code.

for eg :-

To validate a US phone number like 123-456-7890, we can use regex as this :-

^\d{3}-\d{3}-\d{4}$

it means (^) start exactly with 3 digits (\d{3}),

then reading of the string ($).
for password validation requiring at least one uppercase, one digit
one special character we write regex as:-

^(?=.*[A-Z])(?=.*\d)(?=.*[!@#$%^&*])

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 are small pieces of data stored in the user's browser. They can store information but are mostly used to store a session ID. cookies are sent with every HTTP request sent to the server.

Sessions are server-side storage that holds user-data (eg:- login status, preferences) and is identified by a session ID.

How do they work together:-

1.) User logs in - server creates a session and stores user info on the server.
2.) Server sends a cookie containing session ID to the client.
3.) Client browser stores the cookie and sends it with each subsequent request.
4.) Server reads the session ID from the cookie and retrieves the session data to authenticate and personalize the user experience.

for eg:-

If cookie sessionId = abc123 is sent with a request, the server finds abc123 in its memory / database and knows the user is 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 verification could lead to a security vulnerability.

Client-side validation happens in the browser (using JavaScript). It provides fast feedback to users, reducing invalid data being sent to the server and improving UX.

Server-side validation happens on the server and is essential for security because users can bypass client-side checks by disabling JavaScript or tampering with requests.

Security risk example :-

If only client-side validation checks that a username does not contain SQL injection code, an attacker can disable JavaScript and submit a malicious payload like ' or '1'='1 directly to the server, potentially allowing unauthorized database access.

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.

```
<form id="loginForm">
<input type="text" id="username" placeholder="Username" required />
<button type="submit"> login </button>
</form>
<div id="welcomeMsg"> </div>

<script>
document.getElementById('loginForm').addEventListener('submit', function(event){
event.preventDefault();
const user = document.getElementById('username').value;
document.getElementById("welcomeMsg").textContent = 'Welcome, ${user}!';});
</script>
```

Here:

When the user submits the form, the script stops the page from refreshing, grabs the input value, and updates a with a personalized welcome message dynamically.

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

Ans:) Following are the steps for connectivity :-

i.) Frontend :- Create a form to collect user input using HTML / CSS / JS.

ii.) Send data to backend :- Use JavaScript's fetch API or API or AJAX to s
input data via HTTP POST / GET to a backend API.

iii.) Backend server :- A server (Node.js, PHP, Python, etc) receives the requ

iv.) Database connection :- The backend uses a MySQL client library (eg.
mysql package in Node.js) to connect to the MySQL database.

v.) Query execution :- Backend executes SQL queries to insert, update or f

vi.) Send response :- Backend sends a success / failure to the frontend

vii.) Frontend updates UI :- The frontend reacts to the response and upda
UI accordingly.

for eg :-

- Frontend sends username / password to POST / login.
- Backend receives, connects to MySQL, runs :-

Select * From users where username = 'input' and Password = 'input'

- Backend returns success or error message to frontend.

(FAQ's)

Qs1.) Write 3 reasons why form validations are important.

Ans:- Three Reasons why Form Validations are important are as follows:-

i.) Improves User Experience :- Provides immediate feedback on incorrect or invalid data entered by users, reducing frustration.

for eg :- <input type = "text" pattern = "[0-9]{10}" placeholder = "Enter" required>

ii.) Maintains Data Accuracy & Integrity :- It ensures that the user inputs are in the correct format. For eg :- an email field should contain a valid email like abc@gmail.com, instead of random text.

for eg :- <input type = "email" required>

iii.) Enhances Security :- Prevents harmful data from being submitted, reducing risks of attacks like SQL injection or Cross-site scripting (XSS).

for eg :- A password field can enforce a strong password (at least 8 characters with letters, numbers and symbols).

Qs2.) Give an example of how to modify an attribute value using DOM.

Ans:- The DOM (Document Object Model) allows us to change HTML elements dynamically using JavaScript. One common task is changing the attribute value of elements.

for eg :-
```
<!DOCTYPE html>
<html>
<body>
  <h2> Change Image using DOM </h2>
<img id = "myImage" src = "old.jpg" alt = "old Image" width = "200">
<button onclick = "changeImage()"> Change Image </button>
<script>
  function changeImage(){
    document.getElementById("myImage").setAttribute("src", "new.jpg"); }
</script>
</body> </html>
```

here;

initially the image source is old.jpg.
when the button is clicked, the javascript function changes the src attribute to new.jpg.

Q3) what are the different feature of javascript?

Ans→ javascript is a lightweight, dynamic, and versatile programming language used primarily to create interactive web applications.

Key feature:-

i) Lightweight and Interpreted

javascript is interpreted line by line in the browser, without the need for compilation.

for eg :- We can run a simple script directly inside <script> tags.

ii) Dynamic Typing

Variables do not need a fixed data type; the type is decided at runtime.

for eg :- let x = 10; x = "hello";

iii) Object-oriented

Supports objects, methods, and prototypes (though not class based like Java or C++).

for eg :-

```
let car = {
    brand : "Porsche 911",
    speed : 220,
    drive : function () {
return this.brand + "drives at" + this.speed + "Km/h";
    }
};
console.log(car.drive());
```

Date : 2/9/2025

Handwritten

Page No. : 11

iv.) Event - Driven Programming

JavaScript can respond to user actions such as click, form submissions, and key presses.

for eg :-

<button onclick = "alert ('Button Clicked!')"> Click Here </button>

v.) Cross - Platform

- Runs on all modern browsers (Chrome, Firefox, Safari, Edge, etc) and is supported on multiple operating systems.

vi.) Powerful for DOM manipulation

JavaScript can dynamically add, delete or modify HTML elements

for eg :-

document.getElementById("demo").innerHTML = "Content Changed!";

? Problem Statement :-

1.) Write a program to design student registration form by using HTML, CSS having following fields : username, email, password and write external javascript code to achieve following validations.

2.) Write a client-side script with JavaScript to access and manipulate Document Object Model (DOM) objects in an HTML web page. Develop a dynamic web page using javascript and DOM.

Conclusion

Sushant Kumar
TY B.Tech CSE CSF
2231329 Roll No:- 30

Handwritten

Date: / /

ISE Laboratory Assignment No 3

- Aim :- Design an interactive front end application using React by implementing templating using components, labels, state 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 interaction with events, ensuring a seamless user experience across various devices and screen size.

- Theory :-

Q: 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:- The purpose and difference of State and Props in React are described in point below :-

- State :- • It is internal, mutable data owned by the component (or by a hook).
  • It changes state (via setState / useState) it triggers re-render.
  • Its primary purpose is component's local data that affects rendering (user input, toggles, etc).
- Props :- • They are read-only inputs passed into a component from its parent.
  • used to configure a component or pass data and callbacks down.
  • A component must not mutate its own props.
  • Its primary purpose is to handle incoming data plus communication from parent to child components.

Date : __/__/____ Page No.: 13

Q 2.) What is React component ? Differentiate between a class component and Functional component, and discuss advantages of using a functional component with hooks like UseState.

Ans:- A React component is a reusable piece of UI logic that takes inputs (props) and returns UI (React elements). Components can compose other components and encapsulate presentation plus behavior. Snippets below highlight difference between Class component and functional

→ Class component :- import React from "react";
class Counter extends React.Component {
constructor(props) {
super(props);
this.state = { count: 0 }; }
increment = () => this.setState({ count: this.state
componentDidMount() { /* */ }
render() {
return <button onClick={this.increment}> {
state.count } </button>; } }

→ Functional component :- import React, { useState, useEffect } from
function Counter({ initial = 0 }) {
const [count, setCount] = useState(initial);
useEffect(() => {
document.title = `count: ${count}`;
[count]);
return <button onClick={() => setCount(c => c + 1)}>
</button>;
}

* Advantages of using functional component with hooks are discussed as :-

- Less boilerplate and cleaner syntax (no this, constructors).
- Easier to compose reusable logic via custom hooks.
- Better for testing and readability.
- Hooks encourage separation of concerns: related logic can be grouped by purpose.
- Functional components align with React development direction and usually result in smaller bundles and simpler mental model.

Q.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 In React, "Templating using components" help us to build user interfaces (UI's) by composing many small components (Button, Card, Header, ProductionList). Each component encapsulates markup (template), style and behaviour. The app so formed is then a tree of components.
→ Its superiority to monolithic HTML files are described in point below:-

- Reusability :- components can be reused across pages / projects.
- Encapsulation :- Each component manages its internal state / styles reducing global side effects.
- Maintainability :- Small focused components are easier to understand, test and refactor.
- Declarative update :- React describes what UI should look like for a given state; React handles DOM changes efficiently (less manual DOM manipulation).
- Composition :- Complex UIs are built by composing simple pieces, following the same mental model as functions.
- Performance :- virtual DOM diffs update only what's necessary.
- Developer ergonomics :- Hot reloading, clean props API, and hooks make evolving UI faster than editing huge HTML files.

Date: / /

Page No.

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

→ Event handlers are functions passed to JSX event props (camelCase) that can update state; code snippet is given below :-

```
import React, { useState } from "react";
function LikeButton() {
  const [liked, setLiked] = useState(false);
  function handleClick() {
    setLiked(prev => !prev);
  }
  return (
    <button onClick={handleClick}>{liked ? "Unlike" : "Like"}</button>
  );
}
export default LikeButton;
```

Q.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.

→ Responsive Web Design (RWD) ensures a website adapts to different screen sizes and devices like desktop, tablet, mobile, etc.

- It improves user experience across all platforms.
- Essential for accessibility and Search Engine optimization (SEO).
- Reduces the need for multiple device-specific versions.

Implementing Responsiveness in React:

→ CSS media queries :-

```
.container {
  width : 100%;
  padding : 200 px;
}
@ media (max-width : 768 px) {
  .container {
    padding : 10 px;
    font-size : 14 px;
  }
}
```

→ CSS in JS

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

→ UI libraries :- (Bootstrap, Material-UI, Tailwind CSS)
. These libraries provide us with built in responsive classes.

? Problem statement :- Q Roll 21 to 40] Design and develop a responsive React application for a User profile dashboard. The application should display user information, including a profile picture, name and a list of recent activities or posts.

Date : __/__/____

Page No.

Conclusion :- Thus learned and implemented some of the key f
React development. components make the UI
and reusable, state and props execute and
efficient data flow and dynamic updates, Event h
makes applications interactive; while RWD helps to
accessibility across ~~the~~ cross-platform or devices.

X

26/9/25

Date: / /

Page No.: 18

FSD Laboratory Assignment 4

- **Aim:** Enhance web page developed in earlier assignment by rendering lists and tables. Error Handling, routers and style with React CSS also make it a responsive design to work well across PC, Tablet and Mobile Phone.

- **Objectives:** i) Enhance user interface and experience
  ii) Improve application Robustness and Navigation

- **Theory:**

Q3.2) How do list & keys work in React?

Ans: When rendering dynamic array of elements. React lets us use .map() function to produce a list of component / elements.

code snippet:-

```
const fruits = ["Apple", "Banana", "Mango"];
return (
  <ul>
    {fruits.map(fruit => <li key={fruit}>{fruit}</li>)}
  </ul>
);
```

While keys are special prop used to help React identify which items have changed, added or removed. Keys should be unique, stable and predictable (to avoid using array indexes if data can reorder) without keys, React may unnecessarily re-render / reuse the wrong DOM nodes.

Date : __/__/____

Page No. 12

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

Ans - A React Portal allows rendering children into a different part of the DOM tree (outside the parent component's hierarchy), while still being logically part of the React component tree.

→ import reactDOM from "react-dom";
function Modal ({ children }) {
return ReactDOM.createPortal (
 {children} ,
document.getElementById("modal-root")
); }

- Modals, tooltips, dropdowns, popovers - elements that need to visually escape of parent containers (eg - outside overflow: hidden or stacking context) but still controlled by React.

Q 3.) Discuss the importance of Error Boundaries in React.

Ans - Error Boundaries in React components that catch JavaScript errors in their child component tree and prevent the whole app from crashing. It can display a fallback UI ("like something went wrong") and only catch rendering errors, lifecycle errors and constructor errors but not event handler or async errors.

eg - class ErrorBoundary extends React.Component {
state = { hasError: false };

static getDerivedStateFromError () {
return { hasError: true };
}

componentDidCatch (error, info) {
  console.error("Error caught:", error, info);
}
render() {
  if (this.state.hasError) return <h2>Something went wrong.</h2>;
  return this.props.children;
}
}

In production, it prevents a single broken component from bringing down the entire app.

Q.6) How does React Router enable Single Page Application (SPA) functionality?

Ans- SPA (Single Page Application) : loads a single HTML page, dynamically update content without full page reload ; thus improves speed and user experience like a desktop application.

, React Router intercepts browser navigation events (clicking links, using back/forward), updates the URL and renders the matching component without reloading page. Uses <BrowserRouter>, <Routes>, <Route> and <Link> for declarative routing.

eg- import { BrowserRouter, Routes, Route, Link } from "react-router-dom";
function App() {
  return (<BrowserRouter>
<nav>
<Link to = "/" > Home </Link> <Link to ="/about"> About
</Link> </nav> <Routes>
<Route path = "/" element = {<h1> Home </h1>}/>
<Route path = "/about" element = {<h1> About </h1>}/>
</Route>
</Routes>
</BrowserRouter>
);
}

Date : / /

</BrowserRouter>); }

Q:3) Explain the different ways to style a React application.

i) Plain CSS :- Import .css file or use CSS modules for scoped

→ import "./App.css";
import styles from "./App.module.css";
 Click

ii) Inline styles :- Use JS objects for styles

→ <div style={{ color: "blue", padding: "10px" }}>

iii) CSS in JS :- Write CSS directly inside JS, scoped to comp

→ import styled from "styled-components";
const Button = styled.button`
background: blue;
color: white;
&:hover { background: darkblue; }`;

iv.) Utility-first CSS :- use prebuilt utility classes

→ <button className="bg-blue-500 text-white">Click</button>

7. Problem statement :- Expand the e-commerce product gallery to use responsive model for displaying product details (Roll

C.P. 26/9 Conclusion :- Thus implemented and understood the concept of Portals, error boundaries, React Router and styling techniques in React. These enhancements collectively improve user experience and application robustness.

X

Sushant Kumar
TY B.Tech CSE CSF
Roll No.- 30

Date: __/__/____

FSD Laboratory Assignment No. 5

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

Objectives :- i.) Develop a Full-Stack Web Application
ii.) Demonstrate Backend Development and Deployment Proficiency

Theory :-

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

Ans- Node.js provides a runtime environment for executing Javascript on the server side. however, building a full fledged web application directly with Node.js can be time consuming because you would have to handle things like routing, HTTP requests, middleware and responses manually. Express.js comes into the picture as a minimalist, fast and flexible web framework for Node.Js that simplifies these tasks. Some key roles of Express.js are as follows:-

- Routing :- Easily define application states and routes (eg. Get /users, Post /products) for different parts of your App.
- Middleware support :- Helps process incoming request before sending response (eg. authentication, logging, error handling)
- Template engines :- Integrates with templating engines like EJS, Pug or Handlebars to render dynamic HTML
- Scalability :- Supports building RESTful APIs and complex web applications.
- Integration :- Works smoothly with databases like MongoDB, MySQL or PostgreSQL

eg.

```
const express = require ('express');
const app = express ( );
app. get ('/', (req, res ) => {
    res. send (" Welcome to my JS App ! "); });
app. listen ( 3000, ( ) => {
    console. log ( 'Server running on port 3000' ); });
```

Date : __/__/____

Page No.:

Q522 Explain the concept of CRUD operations in context of web application.

Ans- CRUD stands for Create, Read, Update and Delete ; the four basic functions for interacting with data in any web application.

Operation	HTTP method	Description	Example in a Web App
Create	POST	Add new data to database	User signs up → data
Read	~~GET~~ GET	Retrieve data from database	View list of products →
Update	PUT / PATCH	Modify existing data in database	Edit user profile →
Delete	DELETE	Remove data from database	Delete product from

for eg :-

```
app.post('/user', async (req, res) => {
    const user = new user(req.body);
    await user.save();
    res.send(user);
});
```

Create

```
app.get('/user', async (req, res) => {
    const users = await User.find();
    res.send(users);
});
```

Read

```
app.put('/users/:id', async (req, res) => {
    const user = await user.findByIdAndUpdate(req.params.id
        { new: true });
    res.send(user);
});
app.delete('/users/:id', async (req, res) => {
    await User.findByIdAndDelete(req.params.id);
    res.send({ message: "user deleted" });
});
```

Q.23) Why is MongoDB suitable choice for this project ?

Ans MongoDB is noSQL, document-oriented database that stores data in flexible, JSON like documents rather than rigid rows and column like traditional SQL databases.

Reasons for its suitability are :-

i) Schema-less Structure :- No fixed schema ; thus easy to modify data structures as the application evolves.

ii) Scalable :- Handles large amount of data efficiently, support horizontal scaling (via sharding).

iii) JSON compatibility :- Stores data in BSON, perfect for JS applications

iv) High Performance :- Optimised for read/write operations, making CRUD very fast.

v) Integration with Node.js :- Libraries like Mongoose simplify communication between Node.Js and MongoDB.

vi) Cloud Support :- Easily Deployable on MongoDB Atlas for production use.

Q.24) What steps are involved in in deploying a Node.js and Express application

Ans Deploying means making our web application available on the internet for users. A common approach is using cloud platforms like Heroku, Render, AWS or DigitalOcean.

Typical steps :-

1) Prepare the Application :-

- Ensure package.json has all dependencies
- use environment variables for sensitive data (eg. DB cred
- Add a start script in package.json → "start" : "node server.js".

Date : __/__/____

Page No.:

2) Set up Production database :
- use MongoDB Atlas or other cloudbase services
- Update your connection string in .env file.

2) Push code to GitHub :
create a repository → commit and push your code

3) Deploy on hosting platform :
- for heroku , →
heroku creat
git push heroku main
heroku config : set MongoDB URL = your-db-connection
This app gets a live URL after deployment.

5) Testing & monitoring :-
- Check logs using heroku logs --tail or equivalent tool
- set up monitoring and error tracking (with PM2, log

? Problem statement :- [Roll 21 to 40] E-commerce Product Catalog Design
build a responsive web page that acts as a product
The Express backend handles all product data, frontend displays all pro
in grid, admin panel adds new products, view all products, mo
a product detail and remove product. The product information will
stored in a MongoDB collection.

- Conclusion :- Express.js plays a crucial role in simplifying server
development, while CRUD operations form the foundation of
dynamic applications. MongoDB is well suited due to its flexible
No SQL, schemaless, BSON design. Thus learned and implemented about
tools that together enable the creation of a robust, responsive and
fully deployed full-stack application.

26/9/25

×