**Web and Software Development 6 Months Course Outline**

|  |  |  |  |
| --- | --- | --- | --- |
| **TIME** | **DETAILS** | **OUTCOMES** | **SIGNATURE** |
| **Week 1** | Introduction to Web Development: Basics of HTML, CSS, and JavaScript. Setting up environment. | Understand the structure of a web page and basic web technologies. |  |
|  | Install Node.js, npm, and a code editor (e.g., VSCode). | Ability to set up and navigate a development environment. |  |
| **Week 2** | JavaScript Deep Dive: Variables, Data Types, Functions, and Loops. | Strong foundation in JavaScript syntax and programming concepts. |  |
|  | Introduction to ES6+ features like let/const, arrow functions, template literals, etc. | Familiarity with modern JavaScript features. |  |
| **Week 3** | DOM Manipulation: Working with DOM, Event Listeners, and Basic Interactions. | Ability to create dynamic web pages using JavaScript. |  |
|  | Introduction to Git and GitHub: Creating repositories and version control basics. | Basic version control knowledge for collaborative development. |  |
| **Week 4** | Basics of HTTP and REST APIs: Understanding requests, responses, and status codes. | Understanding of how web applications communicate over the internet. |  |
|  | Fetch API and basic data handling in JavaScript. | Ability to make API calls and process responses. |  |
| **Week 5** | Introduction to React: Functional components, JSX, and Props. | Build basic React components and structure simple applications. |  |
|  | Setting up a React project using Create React App (CRA) and Vite. | Hands-on experience setting up and running React applications. |  |
| **Week 6** | State Management in React: useState and useEffect hooks. | Manage component state and handle side effects in React applications. |  |
|  | Styling in React: CSS Modules, Styled Components, and inline styles. | Create visually appealing and responsive React applications. |  |
| **Week 7** | JavaScript in Depth: Advanced concepts like closures, promises, and async/await. | Write efficient and asynchronous JavaScript code. |  |
|  | Working with modules and bundlers (e.g., Webpack, Vite). | Understand the JavaScript module system and optimization techniques. |  |
| **Week 8** | Responsive Design: Using Flexbox, Grid, and Media Queries. | Build websites that adapt seamlessly to different screen sizes. |  |
|  | Accessibility Best Practices: ARIA roles, semantic HTML. | Design accessible web applications for diverse users. |  |
| **Week 9** | Advanced React: Context API, useReducer for state management, and React Router. | Handle complex state and routing in React applications. |  |
|  | Custom hooks: Creating and using custom reusable logic. | Enhance React development with reusable hooks. |  |
| **Week 10** | Introduction to Node.js: Event loop, modules, and building a basic server. | Build simple server-side applications using Node.js. |  |
|  | Express.js Basics: Setting up routes, middleware, and controllers. | Structure backend applications effectively. |  |
| **Week 11** | MongoDB Basics: Introduction to NoSQL databases, CRUD operations. | Understand and work with MongoDB for data storage and retrieval. |  |
|  | Mongoose ODM: Schema creation, validations, and relationships. | Manage MongoDB collections effectively using Mongoose. |  |
| **Week 12** | Connecting Frontend and Backend: Setting up APIs for React frontend to consume. | Build a basic full-stack application integrating React and Express. |  |
|  | Implementing RESTful API standards. | Design and consume RESTful APIs in projects. |  |
| **Week 13** | Authentication: Implementing JWT-based authentication and authorization. | Secure applications with user authentication and role-based access control. |  |
|  | Storing and validating tokens on the frontend. | Implement robust user login/logout flows. |  |
| **Week 14** | Advanced Express.js: File uploads, error handling, and optimization techniques. | Build production-ready backend services. |  |
|  | Working with external APIs: Consuming third-party services. | Integrate external functionalities into applications. |  |
| **Week 15** | Advanced React Concepts: Server-side rendering (SSR) and Next.js basics. | Optimize React apps for performance and SEO. |  |
|  | State management with Redux Toolkit. | Manage complex global state effectively in large applications. |  |
| **Week 16** | Testing: Unit testing with Jest and React Testing Library. | Write test cases to ensure the reliability of frontend code. |  |
|  | Backend testing with Mocha, Chai, and Supertest. | Test backend logic and API endpoints. |  |
| **Week 17** | Real-time Communication: WebSockets and Socket.io. | Implement real-time features like chat and notifications. |  |
|  | Building scalable server-side applications. | Optimize applications for real-time use cases. |  |
| **Week 18** | Performance Optimization: Code splitting, lazy loading, and caching. | Improve application performance and user experience. |  |
|  | Advanced deployment strategies (e.g., CI/CD pipelines). | Automate and streamline application deployment workflows. |  |
| **Week 19** | Advanced MongoDB: Aggregation, indexing, and replication. | Work with complex data operations and enhance database performance. |  |
|  | Database security and backup strategies. | Ensure secure and reliable database management. |  |
| **Week 20** | Microservices Architecture: Introduction and building microservices with Node.js. | Design scalable and modular backend systems. |  |
|  | API Gateway and service orchestration. | Manage and coordinate multiple microservices effectively. |  |
| **Week 21** | Advanced React: Animations with libraries like Framer Motion. | Enhance UI/UX with smooth and dynamic interactions. |  |
|  | Optimizing React performance (e.g., memoization, React Profiler). | Build highly efficient and responsive React applications. |  |
| **Week 22** | GraphQL Basics: Schema design, queries, and mutations. | Work with GraphQL APIs for flexible data querying. |  |
|  | Integrating GraphQL with React and Apollo Client. | Build full-stack applications with GraphQL. |  |
| **Week 23** | Security Best Practices: Preventing XSS, CSRF, and other vulnerabilities. | Implement secure coding practices across the stack. |  |
|  | Advanced logging and monitoring strategies. | Maintain and debug production-ready applications. |  |
| **Week 24** | Capstone Project: Building a production-grade MERN stack application. | Apply all learned concepts to create and deploy a full-stack application. |  |
|  | Presentation and code review. | Demonstrate the ability to develop and deliver a complete software solution. |  |