|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Start Date** | 14-Aug | **End Date** | 18-Sep |  |  |
| **August** | | | **September** | | |
| **Week 1** | 14-Aug | **Prerequisite**   * HTML, CSS & JS * JS VS JSX * JSX Style Component * JSX Properties * Arrays (Add, Replace & Remove) * Import & Exports * Objects * Inputs Types (Date, Text etc.) * Handling Inputs * Integer, String, Boolean etc. | **Week 4** | 2-Sep | **Testing**   * Testing in JavaScript * Assertions, test runners, mocking in javascript, mocking network responses. * End-to-end testing libraries like cypress, playwright etc.   **State Management**   * Actions, reducers, hook, redux, immutability helper libraries. |
| 15-Aug | **JSX**   * IF/Else & Operators in JSX * Repeating Elements with key property. * Adding event handlers in JSX. * JSX Fragments * Security Attacks * Developer Environment (Formatter : prettier, linter: eslint, developing with node.js & npm, development server) * Production build & deployment. * React Developer Tools Browser Plugins * Project File Structure, Common Approaches. | 3-Sep | **Advanced Concepts**   * Portals * Errors boundaries * Higher-order components. * Render Props.   **Other Topics**   * User Authentication * Internalization * Pre-rendering & next.js * Redux in detail. * React & GraphQL / Apollo * React Native * PSAs |
| 16-Aug | **TypeScript**   * Type Script Basics, its benefits. * Type declarations for libraries * Type inference * Declaring types of variables * Primitive types (Int, String, Boolean) * Array Types * Object Types * Exporting & Importing in Type Script * Function types * Type assertions * Union types | 4-Sep | **Node.js Fundamentals**   * Introduction to Node.js, Node.js Runtime & architecture, Event-Driven architecture, Non-blocking I/O. * Node.js vs bowser JavaScript, Commom JS modules in Node.js. * ES6 module in Node. Require & module.exports * Import & Export. |
| **Week 2** | 19-Aug | **Components**   * Function components & Class Components * Components as function * Sharing data between components. * Existing Component Libraries * Props in custom components * Events in custom components * Passing content to components * Wrapping existing elements in components (Button, Styled Button) | 5-Sep | **JavaScript Basics for Node.js**   * JavaScript syntax & operators. * Functions & closures. * Asynchronous JavaScript (Callback, promises, async/await) * Error handling in JavaScript.   **JavaScript Core Modules**   * File system fs * Path manipulation path * HTTP module * HTTPS module * Events module * Stream API module * URL module * OS module   Process module |
| 20-Aug | **Hooks**   * What are hooks? * Rules of hooks * Overview of built-in hooks from libraries * Creating custom hooks. | 6-Sep | **Package Management**   * Introduction to NPM * Package.json file, scripts, dependencies & metadata. * Installing & updating packages. * Semantic versioning * Understanding Node\_Modules & .gitignore.   **Asynchronous Programming**   * Understanding the event loop. * Callbacks vs promises vs async/await * Error handling in asynchronous code. * Handling multiple asynchronous operations. |
| 21-Aug | **React basics**   * Component Libraries, API queries, routing, styling tools, from handling, state management. * Network requests in JavaScript (Fetch, async/await) * API-query libraries for React. * useEffect | **Week 5** | 9-Sep | **Web Development**   * Building basic HTTP server * Using express.js routing, middleware & error handling. * Parsing request (body-parser) * Serving static files.   **API & Databases**   * Creating RESTful API’s * Introduction to GraphQL * Working with databases (PostgreSQL, NoSQL) e.g mongoDB * Querying databases. * Handling database migrations. |
| 22-Aug | **react-router**   * Routing Basics (client-side routing. * defining routes, using links, nested routes. * route parameters, navigation from JS, active link style) | 10-Sep | **Authentication & Authorization**   * User authentication strategies (JWT, OAuth) * Securing API’s & web application * Handling user sessions & cookies.   **Error Handling & Debugging**   * Debugging Node.js Applications. * Logging best practices. Error handling strategies.   **Testing**   * Unit Testing with frameworks like Mocha, Jasmine or Jest. * Integration testing. * End-to-end testing with tools like cypress or puppeteer. * Mocking & stubbing. |
| 23-Aug | **Styling Tools**   * Class names package * CSS Modules * SCSS * CSS in JS * Inline styles & Styled Components * Animations (Appearance & disappearance of elements) | 11-Sep | **Performance Optimization**   * Profiling & monitoring * Optimizing performance with caching * Memory management * Scaling Node.js applications * Load Balancing.   **Security**   * Understanding common security vulnerabilities (XSS, CSRF) * Securing HTTP Headers. * Using HTTPs and SLL/TLS * Input validation & sanitization. |
| **Week 3** | 26-Aug | **Form Validation**   * Form Validation strategies * Libraries (react-hook-form, formik)   **State Management**   * Libraries Zustand, redux , mobX etc. | 12-Sep | **Deployment Operations**   * Deployment Strategies * Continuous Integration * Hosting Options * Environments variables & configurations management.   **Advanced Topics**   * Understanding & implementing microservices architecture. * Using worker threads and child processes. * Creating custom Node.js module. * Implementing web sockets for real-time communication. |
| 27-Aug | **Advance React**   * State Hook detail, state transformer functions, outdated state. * State initialization function * Defining Components as classes * Props in class Components * State in class components * Potential problems & side-effects. * Cleaning up side effects. * Strict mode | 13-Sep | **Additional Libraries & Tools**   * Using template engine. * Task runners & build tools. * File Upload & processing libraries. * Email Sending libraries. * Web scraping libraries. |
| 28-Aug | **Context**   * Pure Providers & stateful providers * Stateful context with constate * Context with vanilla React | **Week 6** | 16-Sep | **Best Practice & Design Patterns**   * Writing clean & maintainable code. * Adhering to coding standards & conventions.   Using design patterns like singleton, factory & observer in Node.js |
| 29-Aug | **Refs**   * Refs for storing objects that don’t influence the rendering. * Using refs for avoiding outdated states. * The ref property for accessing DOM elements. * The ref property in custom components. | 17-Sep |  |
| 30-Aug | **Performance Optimization**   * React dev Tools & performance. * Memorization * Memorization of expensive calculations. * Skipping unneeded renders. * Virtual DOM. * Lazy-loading components. * Reducing bundle size. | 18-Sep |  |