

# Job Ready AI Powered Cohort 2.0: Complete Web Development + DSA + Aptitude & Reasoning + Web3 + GenAI + DevOps + AWS

---

 [Projects Exercises](#) - [See all the exercises that matters](#)

---

## Episode 1 - Code

### 1. How the Internet Works:

- History of Web (Web 1.0 to Web 3.0).
- How computer communicate with each other.
- How computer send data all over the world.
- What is Domain Name, IP & MAC Addresses and Routing.
- How ISP and DNS work together to deliver data.

### 2. Client-Server Architecture:

- What is Client-Server Model.
- Difference between Client (browser) and Server (the computer hosting your website).

- How HTTP request and response cycle works (how browser talk to server).
- What happens when you visit a website.
- Difference between Front-end and Back-end (Front-end vs Back-end).
- What are Static Websites and Dynamic Websites.
- What is web hosting and how it works.

### 3. Internet Protocols:

- What is TCP protocol and why is widely used
- How Connection is established using TCP (3 Way handshake)
- What is UDP and why its used for fast communication
- How UPD establishes connection
- Difference between TCP and UPD

### 4. Understanding HTTP & HTTPS

- What is HTTP and its different version
- HTTP status code for responses
- What is HTTPS and why its better than HTTP
- How HTTPS provides a secure connection
- What is SSL/TLS Encryption
- What are Proxy and Reverse Proxy
- How VPN works and helps accessing restricted content

### 5. Preparing Your Machine

- Installing & Setting up VS Code
- Installing helpful extensions
- Setting up your browser for development
- What are file and folders and how to create them
- Testing our environment via serving a webpage - " Namaste Duniya "

---

# Episode 2 - Stage

## 1. Starting with HTML

- Understanding HTML and its use Cases.
- Creating first HTML page in VS Code
- Understand HTML Structure
- Understanding Tags and building simple HTML page - `doctype` , `html` , `head` , `title` , `body`
- Working with text elements - `h tags` , `p tag` , `br tag` , `a tag` , `span` , `code` , `pre`
- Working with HTML Lists(Ordered & Unordered lists) - `ol` , `ul` , `li`
- Understanding Concept of nested elements in HTML
- Working with Media Tags - `img` , `video` , `audio`
- HTML attributes - `href` , `target` , `alt` , `src` , `width` , `height` ,
- Navigating between pages

## 2. More on HTML

- Understanding semantic tags - `article` , `section` , `main` , `aside` , `form` , `footer` , `header` , `details` , `figure`
- Differentiating between block and inline elements
- Text formatting tags in HTML - `b` , `strong` , `i` , `small` , `ins` , `sub` , `sup` , `del` , `mark`
- Working with HTML tables - `table` , `td` , `tr` , `th`

## 3. HTML Forms and Inputs

- What is Form and why its important
- Creating a simple Form with tags - `form` , `input` , `textarea` , `select` , `button` , `label`
- Types of input fields - `checkbox` , `text` , `color` , `file` , `tel` , `date` , `number` , `radio` , `submit` , `range`

- Attributes of Form Elements - `method` , `actions` , `target` , `novalidate` , `enctype` , `name` , required, placeholder

## 4. Media Tags in HTML

- Understanding with audio and video Tags
- Attributes if media tags - `src` , `width` , `height` , `alt` , `muted` , `loop` , `autoplay` , `controls` , `media`
- Using source element for alternative media files

## 5. Basics of CSS (Cascading Style Sheet)

- Introduction to CSS and Why it is important
- Understanding Syntax, Selectors and comments in CSS
- Adding CSS to HTML Page - `Inline` , `Internal` , `External`
- Understanding difference between selectors - `class` , `id` , `element`
- Understanding precedence of selectors
- How to style text using CSS - `font family` , `font style` , `font weight` , `line-height` , `text-decoration` , `text-align` , `text-transform` , `letter-spacing` , `word-spacing` , `text-shadow`

## 6. Styling With CSS

- Working with colors in CSS - `name` , `rgb` , etc.
- Working with css units - `%` , `px` , `rem` , `em` , `vw` , `vh` , `min` , `max`
- Working with borders and border styling
- Working with box properties - `margin` , `padding` , `box-sizing` , `height` , `width`
- Understanding Background properties - `background-size` , `background-attachment` , `background-image` , `background-repeat` , `background-position` , `linear-gradient`
- Implementing shadow property.

## 7. More about CSS

- Applying display properties - `inline` , `grid` , `flex` , `none` , `inline-block` , etc.

- Introduction to FlexBox for aligning and structure - `flex-direction` , `order` , `flex-wrap` , `flex-grow` , `flex-shrink` , `justify-content` , `align-items` , `align-content` , `align-self` , `flex-basis` , `shorthand properties of flex`
- Understanding Flex Grid for making grids using CSS.
- Working with positional properties - `absolute` , `relative` , `static` , `sticky` , `fixed` .
- Understanding Overflow - `visible` , `hidden` , `scroll`.
- Working with Grouping Selectors.
- Why we use Nested Selectors.

## 8. Interesting things about CSS 🙌

- Applying pseudo classes and Pseudo Elements [ `hover` , `focus` , `after` , `before` , `active` ] .
- Learning CSS Transitions ( `properties` , `duration` , `timing functions` , `delays` ).
- Creating with `Transform` ( `translate` , `rotate` , `scale` , `skew` , `transform` , `rotate` ).
- Working with `3D Transform` ( `translate3d()` , `translateZ()` , `scale3d()` , `scaleZ()` , `rotate3d()` , `rotateZ()` ) .
- Understanding `CSS Animation` ( `@keyframes` ).

## 9. Responsive with CSS 🖥️

- Difference Between Mobile-first and Desktop first Website(mobile-first vs desktop first).
- Measurement units for Responsive Design - `px(pixel)` , `in(inch)` , `mm(millimetre)` , `%` , `rem`
- Using Viewport meta element for Responsive.
- Setting up `Images` and `Typography` for Responsiveness.
- What are Media queries [ `@media` , `max-width` , `min-width` ].
- Using Different function of CSS [ `clamp` , `max` , `min` ].
- Understand HTML structure for Responsive Design.

## 10 Working With SASS (SASSY) my favorite 🏆

- What is SASS? `Variables` , `Nesting` , `Mixins` , `Functions and Operators` .
- Setting up environment for `SCSS` .
- SCSS or SASS? and Setting Up `SCSS` .

## 11. Basics of Javascript with ES6+ Features 🚀

- Introduction to JavaScript, Why it is Important! and What can it do for you?
- How to link javascript files using `script-tag` .
- Running JavaScript in the Browser Console .
- Variables and Keywords in Javascript [ `var` , `let` , `const` ].
- Logging with javascript - [ `console.log()` , `console.info()` , `console.warn()` , `prompt` , `alert` ]
- Working with String in JS and there -[ `splice` , `slice` , `template string` , `split` , `replace` , `includes` ]
- What are Statement and Semicolons in JS
- How to add Comments in JavaScript
- What are Expression in Js and difference between expression and statement
- JavaScript Data Types - [ `float` , `number` , `string` , `boolean` , `null` , `array` , `object` , `Symbol` , `Undefined` ]
- Some Important Values - [ `undefined` , `null` , `NaN` , `Infinity` ]
- Relative and Primitive Data Type in JavaScript
- Basic Operators(Arithmetic, Assignment, Increment, Decrement, Comparison, Logical, Bitwise) - [ `+` , `,` , `/` , `++` , `-` , `==` , `===` , `!=` , `and more` ]
- Variable hoisting in JavaScript

## 12 . Loops & Conditionals in Javascript

- Understanding Condition Operator in Javascript - [ `if` , `else` , `if-else` , `else-if` , `Ternary Operator` , `switch` ]
- `for` Loop in JavaScript

- `while` Loop in JavaScript
- `do...while` in JavaScript
- `forEach` in JavaScript
- `for in` Loop in JavaScript
- `for of` Loop in JavaScript
- Recursion in JavaScript
- Loop control statements - [ `break` , `continue` ]

### 13. Functions in JavaScript

- Understanding Function in JavaScript and why its widely used - [ `parameters` , `arguments` , `rest parameters` , `hoisting` , `Variable Hoisting` , `Function Hoisting` ]
- Parameters in JavaScript - [ `required` , `destructured` , `rest` , `default` ]
- Arguments in JavaScript - [ `positional` , `default` , `spread` ]
- `Classic Function` , `Nested Function` (function within function), `Scope Chain` in Javascript.
- Understanding Immediately Invoked Function Expression(IIFE).
- More Functions in JavaScript - [ `Arrow Function` , `Fat Arrow` , `Anonymous` , `Higher Order` , `Callback` , `First Class` , `Pure Function` , `Impure Function` ]
- Understanding Scoping in JS - [ `Global scope` , `Function scope` ]
- Understanding `Closures` , `Scoping Rule` .

### 14. Arrays and Objects in JavaScript

- What are Arrays in JavaScript and how to Create an Array.
- Understand How to Accessing Elements in Array.
- Functions on Arrays - [ `push` , `pop` , `shift` , `unshift` , `indexOf` , `array destructuring` , `filter` , `some` , `map` , `reduce` , `spread operator` , `slice` , `reverse` , `sort` , `join` , `toString` ]
- Iterating Over Arrays using - [ `For Loop` , `forEach` ]
- Understanding What are Objects in JavaScript - [ `key-value pair` ]

- Creating Objects, Accessing Properties, Deleting Property and Nested Objects.
- Recognise How Objects Are Stored, Traverse Keys of an Object, Array as Object.
- Timing Events - `setTimeout()` , `setInterval()` , `clearTimeout()` , `clearInterval()`
- Operation in Objects - [ `freeze` , `seal` , `destructuring` , `object methods` , `this keyword` ]

## 15. Document Object Model Manipulation

- Introduction to DOM in JavaScript
- Understanding DOM Structure and Tree - [ `nodes` , `elements` , `document` ]
- Fetching Elements in DOM - [ `document.getElementById` , `document.getElementsByTagName` , `document.getElementsByClassName` , `document.querySelectorAll` , `document.querySelector` ]
- DOM Tree Traversal - [ `parentNode` , `childNodes` , `firstChild` , `nextSibling` ]
- Manipulating DOM Element in JavaScript - [ `innerHTML` , `textContent` , `setAttribute` , `getAttribute` , `style property` , `classList` ]
- Create and Removing DOM Elements - [ `createElement()` , `appendChild()` , `insertBefore()` , `removeChild()` ]

## 16. Event Handling in JavaScript

- Event Handling in JavaScript - [ `addEventListener()` , `event bubbling` , `event.target` ]
- Understanding Scroll Events, Mouse Events, Key Events and Strict Mode.
- Working with Forms and Input Elements [ `Accessing Form Data` , `Validating Forms` , `preventDefault()` , `onsubmit` , `onchange` ]
- Working with Classes \*\*\*\*Adding, Removing , Toggling (classList methods)
- Browser Events - [ `DOMContentLoaded` , `load` , `resize` , `scroll` ]

## 17. Using Browser Functionalities in JavaScript

- Browser Object Model - [ `window` , `navigator` , `history` , `location` , `document` ]
- Window Object - [ `window.location` , `window.history` ]



- Working with Storage - [ `Local Storage` , `Session Storage` , `Cookies` ]
- Web APIs in DOM - [ `Fetch API` ]

## 18. Object Oriented Concepts in JavaScripts

- Introduction to OOPS in JavaScript
- Understanding `classes` and `objects` in JavaScript
- Understanding `Constructor` and `Prototypes` - [ `this keyword` , `call` , `apply` , `bind` ]
- More Topics in OOPS - [ `class expression` , `hoisting` , `inheritance` , `getter & setter` ]

## 19. Asynchronous Programming JavaScript

- Introduction to Asynchrony in JavaScript.
- Introduction to `callbacks` and Problems in Callbacks
- Understanding `promises` - `pending` , `resolved` , `rejected`
- How to prevent callback hell using `async` & `await` .
- `setInterval` & `setTimeout` in JavaScript

## 20. Error Handling in JavaScript

- Introduction to Error Handling
- Common types of errors in JavaScript - [ `Syntax errors` , `Runtime errors` , `Logical errors` ]
- Understanding the Error object - [ `message` , `name` , `stack` ]
- Handling exceptions using `try-catch` , `try-catch-finally`
- How to Throw Errors in JavaScript
- How to create custom error in JavaScript
- Error Handling in Asynchronous Code

## 21. Kuch Baatein Advance JavaScript Pr

- Throttling and Debouncing uses in JavaScript
- JSON Handling and JavaScript - [ `JSON.parse()` , `JSON.stringify()` ]

## 22. Git and Github

- What is Git and Github?
  - Concepts - `Git commits` , `Understanding branches` , `Making branches` , `merging branches` , `conflict in branches` , `understanding workflow` , `pushing to GitHub` .
  - How to use GitHub with team members, forking, PR(pull requests) open source contribution, workflow with large teams.
- 

## Episode 3 - Commit

### 1. Introduction of React

- What is React, and Why Use It?
- What are Components and types of Components - `class component` , `function components`
- Understanding Single Page Applications (SPAs), Single Page Applications Vs Multi-Page Applications.
- Difference between `Real DOM` and `Virtual DOM`
- `NPM` Basics | Installing `Packages` .
- How does updates work in React? and More `ES6+` features like `Import & Exports` ,
- Difference Between React and Other Frameworks ( `Angular` , `Vue` ).
- Learning Some Basic Terminal Commands - `pwd` , `ls` , `cd` , `clear`
- Setting Up React Environment with `nodejs` .
- Install `React-Vite` Boilerplate and Installing React Developer Tools.
- Understanding `JSX or JavaScript XML` and Its Importance - `Fragments` , `Components Naming` .
- Creating and Understanding best practices for `Components` in React.

### 2. Styling in React

- Different Styling Approaches.

- Importance of component-based styling. `Inline Styles` , `CSS Modules`
- Dynamic Styling Based on Props or State.
- Responsive Design in React
- `Media queries` with CSS and `styled-components`.

### 3. Animations

- Animation and Transitions Using libraries like `framer-motion` or `gsap` for advanced animations.

### 3. React Basics

- Create Components with `functions` .
- Importing css file/stylesheets in react and Adding a CSS Modules Stylesheet - `Styled Components` , `Dynamic styling with styled-components` .
- Creating a state and Manage State using `setState` - `What is State?` , `setState` , `useState` .
- Creating `Parameterised Function Components` in React.
- `React Props` : Passing Data to Components.
- Function chaining in React and Conditional Rendering - Rendering Array Data via `map` , Eliminating Array Data via `filter` .

### 4. More on React

- `Higher Order Components` in React.
- Reusing Components, Lists and Keys in React.
- Sharing Data with child components : `Props Drilling` .
- Rendering a List, Mapping and Component Lifecycle - `Mounting` , `Updating` , `Unmounting` .
- Understanding React Component `Lifecycle` .

- Different Lifecycle Methods like `componentDidMount` .

## 5. Useful Hooks in React

- Understanding React Hooks
- Rules of hooks.
- Commonly Used Hooks:
  - `useState`
  - `useEffect`
  - `useContext`
  - `useRef`
  - `useCallback`
  - `useMemo`
- `Custom Hooks` : When and How to Create Them
- Understanding and Applying `Context API` .

## 6. Navigation in the React with React Router

- Introduction to React Router.
- Setting Up and Configuring React Router setup of `react-router-dom` .
- Navigating Between Pages with .
- Passing Data while Navigating
- Dynamic Routing
- URL Parameters and Query Strings
- Nested Routes
- Programmatic Navigation Using `useNavigate` .
- Handling `404` Pages : fallback route for unmatched paths, Customizing the "Page Not Found" experience.

## 7. State Management Using Redux.

- Introduction to `Redux` , What is redux?, When and Why use redux?
- Understand Principles of Redux and Redux Flow.
- Understanding State Management in React using Redux.
- Why Use `State Management` Libraries?
- Why Redux need reducers to be `pure functions` .
- Redux Basics: `Actions` , `Reducers` , `Store` , `Currying` , `Middleware` , `Async Actions: Thunk`
- Connecting Redux to React Components with `react-redux` .
- Introduction to `Redux Toolkit`.
- Alternatives: Recoil, Zustand, or MobX.

## 8. Form controls in the React : Building Dynamic Forms

- Introduction to Forms in React.
- Building Basic Forms.
- Creating form elements like `input` , `textarea` , `select` , etc.
- Two way binding with react [ `input` , `textarea` ].
- Handling Form Events [ `onChange` , `onSubmit` , `event.preventDefault()` ].
- Validation in React Forms : `client-side form validation`.
- Integrating Forms with APIs.
- Sending form data to a backend using `fetch` or `axios` .
- Handling loading states and success/error feedback.

## 9. Performance Optimization

- Code Splitting with `React Lazy` and Suspense
- Avoids redundant calculations by caching Using Memoization Techniques:
  - `React.memo`
  - `useMemo`
  - `useCallback`

- Avoiding Re-Renders using `useState` ,
- Optimizing Component Structure
- Performance Profiling Tools using `Chrome DevTools` , `Lighthouse` , `Web Vitals` , Largest Contentful Paint (LCP), First Input Delay (FID)

## 10. Deploying React projects

- Preparing a React App for `Production` .
- Building React Applications.
- Environment Variables in React.
- Deployment Platforms: `Netlify` , `Vercel` , `GitHub Pages` ,

## 11. Real-World Project with React

- Building a Complete React Project
- Combining All Concepts ( `Routing` , `State Management` , `API` , etc.)
- `Styling` and `Responsiveness` ,
- `Optimizing` and `Deploying` the Project.

## 12. Basic SEO Principles

- On-Page Optimization in SEO.
- Guide to SEO Meta Tags.
- Image SEO Best Practices.
- Internal Link Building SEO.
- Create An SEO Sitemap For a Website.

## 13. Three.js and React Three-Fiber

- Understanding what is `Scene` .
- Using `3d models` for animation.
- Controlling view with `Orbit controls.`
- Applying `Lights` inside the scene.

- Understanding different types of `Cameras` .
  - Animating the `mesh` with `GSAP` or `Framer motion` .
  - Different types `Geometries` .
  - Using different `Materials` for animation.
- 

## Episode 4 - Push

### 1. Starting with Node.js - The Beginning 🏁

- Introduction to Node.js and Getting Our Tools - `Node.js LTS` , `Postman` , `Editor`
- Setting up the Tools for our Environments
- Running `script` with `nodejs` - `"Namaste Duniya"`
- `NPM` Basics | Installing `Packages` .
- Creating and Managing `package.json` .

### 2. Creating Server - Writing Our First Server 📱

- What is Server and how it works?
- Setting Up Our First `Node.js Server` using `HTTP`
- Serving A Response to the Browser and Understanding Responses.
- `Routing` in HTTP Servers.
- Understanding Status Code - `1XX` , `2XX` , `3XX` , `404 - Not Found` , `200 - success` , `500 - Internal Server error` , `422 - Invalid Input` , `403 - the client does not have access rights to the content` , etc.
- Installing `Nodemon` for Automatic Server Restarts.

### 3. Some talk on Different Architectures 🏰

- Different Architectures in backend like `MVC` and `SOA` .
- Understanding MVC Architecture `Model` , `View` , `Control` .
- MVC in the context of `REST APIs` .

## 4. Web Framework - Express.js 🚀

- what is `Express.js` and why to use it.
- Setting Up `Express Server` .
- Returning Response from the server.
- Using `Query Parameters` and `URL` Parameters.
- HTTP Request - `Some Important part of requests` , Different Types of Requests - `Get` , `Post` , `PUT` , `Patch` , `Delete` .
- Serving Static Files with `express.static()` .

## 5. Template Engine - EJS 🚧

- What is Template Engine and What is the use of Template Engine.
- Template Engine Option - `Handlebars` , `EJS` , `Pug` , `jade` but We'll use `EJS` .
- Setting Up Template Engine - `Installed EJS template engine` .
- Rendering Our First Page using `EJS` and Some important syntax - `<%= %>` , `<% %>` , `<%- %>` .
- Loop statement, Conditional statement and Locals in views - `EJS` .
- Accessing the Static Files Inside `EJS` file.

## 6. Middleware in Express.js (one of my favorite) 🐒

- Understanding the `middleware` in express.
- Implementing `middleware` with express.
- Different types of middleware : `builtin middleware` , `third-party middleware` , `custom middleware` .
- Different level of middleware : `Application-Level` , `Router-Level` .
- Handling Errors and Security with middleware : `Error-Handling` , `Helmet` , `CORS` .

## 7. Handling file with Express 📁

- Understand `Multer` and its usecase?



- Uploading file with multer.
- Understanding `Memory` and `Disk` Storage.
- Accessing uploaded file `req.file` .
- Working with `express.static` .
- Using `Cloudinary` or `Imagekit` for Real-time media processing APIs and Digital Asset Management.

## 8. Beginning of Database Basics ( Bohot km theory )

- Relational and non-relational Databases : `mongodb` & `mysql` .
- What is `MongoDB` ? Why Use It?
- Installing Compass and Understand how to access DB using terminal.
- Setting Up MongoDB `Locally` and in the `Cloud` .
- Understanding `Datatypes` `Collections` and `Documents` .
- Connecting MongoDB to Node.js with `Mongoose` .
- Database Relations - `One to One` , `One to Many OR Many to One` , `Many to Many` , `Polymorphic` .
- Handling Relationships with Mongoose ( `populate` ).

## 9. API Development(REST)

- What is a REST API?
- Versioning in RESTful APIs - `/v1/`
- Using `Postman` for API Testing and developing - `Send Requests` , `Save Collections` , `Write Tests` .
- Understanding and Working With `Status code` , `2xx (Success)` , `4xx (Client Errors)` , `5xx (Server Errors)` .
- Validating API Inputs Using libraries like `express-validator` or `Sanitization` .
- Security Handling - Rate Limiting with `express-rate-limit` , `XSS Attack` , `CSRF Attack` , `DOS Attack` .

## 10. Database Optimization for Fast response

- Indexing for Performance with MongoDB :- Single-Field Indexes , Compound Indexes , Text Indexes , Wildcard Indexes .
- Best practice with Indexing explain() .
- Learning MongoDB Aggregation .
- Comparison Operators - [ \$eq , \$ne , \$lt , \$gt , \$lte , \$gte , \$in , \$nin ]
- Logical Operators - [ \$not , \$and , \$or and \$nor ]
- Array[ \$pop , \$pull , \$push and \$addToSet ]
- Stages in Aggregation pipeline :- \$match , \$group , \$project , \$sort , \$lookup .
- Creating Database on Local and Atlas
- Creating parallel pipeline with \$facet .
- Learning MongoDB Operators .
- Understanding Different types of Operators :- Comparison , Regex , Update , Aggregation .

## 11. Logging Backend : Express.js

- Why is Logging Important?
- Setting Up Logging with Libraries winstone , Pino , Morgan .
- Different mode of morgan , dev , short , tiny .
- Error Handling and Logging.

## 12. Production Wala Project Structure and Configuration

- Understanding the Basic Structure of application.
- Learning File Naming Conventions, Git Configuration,
- Understanding Important Folders :- src/ , config/ , routes/ , utils/ .
- Role of package.json , ENV and .gitignore .
- Production Environment - PM2 , Error & Response Handling Configuration , CORS Configuration , async-handler.js .

- Using and Configuring `ESLint` and `Prettier` for code formatting.
- Testing APIs using `Postman`.

### 13. Authentication and Authorization

- Difference Between Authentication & Authorization
- Working with `Passwords` and `Authentication` - `Cookie Authentication` , `OAuth Authentication`
- Understanding Session and Token Authentication.
- Implementing JWT Authentication :- `jsonwebtoken` `JWT_SECRET` .
- Securing user password with `bcrypt` `hashing` `salt` .
- Role-Based Access Control ( `RBAC` ).
- Authenticating user with `Express middleware` .
- Understanding `Passport.js` and its usecase?
- Glancing through and Installing Passport.js
- Setting up Passport.js - `passport-local` , `local-strategy` , `google-OAuth`
- `express-sessions` and using passport for authentication.

### 14. Working Real time communication : WebSockets and socket.io



- Understanding `WebSockets` protocol for realtime applications?
- Learning `handshake` , `Persistent connection` , `Bidirectional communication` , `HTTP polling` .
- Understanding difference between WebSocket Vs Socket.io.
- Working with `socket.io` for realtime applications.
- Understanding usage of `Rooms` in Socket.io.
- Understanding `Middleware` in Socket.io.

### 15. Working With Caching - Local and Redis

- What is Caching and How to cache data locally?
- What is `Redis` ?

- Why Use Redis for `Caching` ?
- Implementing `Redis Caching` in `Node.js` .
- Advanced Redis Features `TTL` , `Complex Data Structures` , `Pub/Sub` .

## 16. Error handling in express

- Basic Error Handling in Express `next()` .
- Catching Specific Errors `try` & `catch` .
- Creating Util Class for Error Handling.

## 17. Testing Tools

- Understanding Unit-Testing With Jest.
- Cross Browser Testing and Why Is It Performed?
- What Is Web Testing? and How to Test a Website.

# Episode 5 - Merge

## 1. Generative AI and Applications

- Overview of `Generative AI` : Understanding its core concepts and potential.
- Building an Authentication System with `Generative AI` .
- Exploring `Social Media Automation` and `Content Generation` Projects.
- Introduction to `LangChain` : Features and Practical Uses.
- Developing Real-World Applications: `AI-powered Resume Reviewer` and `Virtual Interview Assistant` using tools like `ChatGPT` or `Gemini` .
- `Agentic-ai application`
- Working with `multi agent system`
- `MCP` server

## 2. Progressive Web App (PWA) Development.

- Overview of `Progressive Web Apps` and their benefits.
- Understanding `Service Workers` and their role in PWA.
- Lifecycle of a Service Worker ( `Install` , `Activate` , `Fetch` ).
- Understanding the `Manifest` File.
- Creating a Manifest.json File.
- `Key Properties` (name, short\_name, icons, start\_url, theme\_color, background\_color)
- Browser `DevTools` for `PWA Debugging` .
- Implementing `Lazy Loading` and `Code Splitting` for improved performance.
- Exploring various `testing techniques` for PWAs.
- Optimizing performance with `advanced caching` strategies.

### 3. DevOps Fundamentals - Docker

- Understanding `DevOps` and its importance in modern software development.
- Learning about Continuous Integration and Continuous Deployment (CI/CD) `pipelines` .
- Introduction to `Docker` and the basics of `containerization` .

### 4. Building Microservices with Node.js

- What are `Microservices` ? Why Use Them?
- `Monolithic` vs `Microservices` Architecture.
- `Challenges` of Microservices.
- Creating a `Node.js` Microservice.
- Designing a Microservice Architecture for a sample application.
- Role of `package.json` in Each Microservice.
- What is `Inter-Service` Communication?

- Communication Patterns ( `Synchronous` vs `Asynchronous` ).
- Role of an `API Gateway` in Microservices.
- Setting Up an API Gateway with `Express.js` .
- Microservices and `Proxying Requests` .
- `Rate Limiting` and Authentication in API Gateway.
- `REST APIs` for Communication
- Understanding `Message Brokers` (e.g., Redis `Pub/Sub` ).
- `Event-Driven` Communication with `Redis` or `RabbitMQ` .
- OverView of `Docker` and `Kubernetes` .
- Using Docker for microservice.

## 5. Nextjs

- Next.js Fundamentals
- File-based routing
- Static assets & Image optimization
- Dynamic routes ([id].js)
- Rendering & Data Fetching
- Styling in Next.js
- Deployment

## 6. Web3 Basics. ₿ (Introductory)

- Understanding the concept and potential of `Web3` .
- Fundamentals of `Blockchain` technology and how it powers Web3.
- Exploring Decentralized Applications ( `DApps` ) and their use cases.
- Introduction to `Smart Contracts` : How they work and their applications.
- Overview of `Cryptocurrencies` and their role in the `Web3` ecosystem.

## 7. Deployment

- We will be deploying the project on the cloud.
  - Easy and Smart - We'll `DigitalOcean App Platform` (in-built load-balancer, scalable, containers) for Deploying our app.
  - Service providers give us a machine-like cloud [ AWS, GCP, Heroku, Azure ] but we'll use `AWS`.
  - Launching Our First Machine using `EC2`.
  - Setting up the Machine - `SSH`.
  - Pulling the code and clone the repository of the code to the main server.
  - Configuring the `NGINX`.
  - Masking the `Domain` On Our `IP` (We are now going to buy a new domain and Link it with cloud AWS).
- 

# DSA with JavaScript

## 1. Conditional Statements

- Understanding Conditional Statements
- Types of Conditional Statements `if` , `if-else` , `if-else if` , `switch`
- Making decisions in a program based on inputs or variables.
- Validating user data or input forms.
- Creating interactive menus or options in applications.

## 2. Loops, Nested Loops, Pattern Programming

- Understanding the use of Loops.
- `for` loop.
- `while` loop.
- `do-while` loop.
- Understanding the Use of Nested Loops.

- Learning Pattern Programming - `Pyramid patterns` , `right-angled triangles` , and `inverted triangles` .
- Understanding Control Flow statement `break` and `continue`
- Learning how to set correct conditions to avoid getting stuck in infinite loops.
- Understand how to optimize nested loops for better performance and reduced time complexity.

### 3. Array

- Understanding the use of Arrays.
- Basic Manipulations - `insertion` , `deletion` , `updatation`
- Accessing Elements in Arrays .
- Traversing Elements in Arrays .
- Array Algorithms - `Two Pointer Algorithm` , `Rotation Algorithms` , `Kadane's Algorithm` , etc

### 4. Object-Oriented Programming (OOP) in JavaScript

- Understanding Object-Oriented Programming
- Learn how to define a `class` for creating objects.
- Understand how to instantiate `objects` from a class
- Learn how the `constructor()` function initializes an object when it's created.
- Understand how `this` refers to the current object in the context.
- Use `this` to access properties and methods within the same object.

### 5. Strings in JavaScript

- Understanding Strings in JavaScript
- Learning String Manipulation Methods - `concat()` , `slice()` , `substring()` , `replace()` , `replaceAll()`
- Learning String Search and Check Operations - `indexOf()` , `lastIndexOf()` , `includes()` , `startsWith()` , `endsWith()`
- Learning String Transformations - `toUpperCase()` , `toLowerCase()` , `trim()`



- Learning String Splitting and Joining: - `split()` , `join()`
- Embed variables and expressions in strings using backticks `()``
- Learning Escape Characters - `\n` , `\t` , `\'`
- Algorithms on Strings - `Reverse a String` , `Check for Palindrome` , `Find Longest Common Prefix` , `Character Frequency Count` , `Anagram Check`

## 6. Time and Space Complexity

- Understanding Time Complexity
- Understanding the `Big-O` Notation.
- Constant Time - `O(1)`
- Logarithmic Time - `O(log n)`
- Linear Time - `O(n)`
- Linearithmic Time - `O(n log n)`
- Quadratic Time - `O(n2)`
- Exponential Time - `O(2n)`
- Factorial Time - `O(n!)`
- Key Factors That Affect Complexity - `Algorithm Design` , `Data Structure Choice` , `Problem Constraints`
- Tips to Reduce Time Complexity - `Avoid Nested Loops` , `Efficient Data Structures` , `Optimize Recursion` , `Divide and Conquer`
- Understanding what is Recursion and its use case

## 7. Math Problems and Algorithms

- Understanding Mathematical Operations and Their Applications
- Mathematical operations like `(pow)` `(sqrt)` and greatest common divisor (HCF) are essential in various problem-solving scenarios.

## 8. Advanced Problems on Array

- Understanding Advanced Array Concepts

- Learning `two-pointer` approach ,
- Learning prefix sums
- Solving complex problems efficiently.
- `Multi-Dimensional` Arrays in JavaScript
- Working with Multi-Dimensional Arrays
- Key Operations on Multi-Dimensional Arrays
- Algorithms Using Multi-Dimensional Arrays
- Multi-Dimensional Arrays in Real-World Scenarios

## 9. Sorting Algorithms ,Time complexity and their application

- Learning `Selection Sort`
- Learning `Insertion Sort`
- Learning `Merge Sort`
- Learning `Quick Sort`
- Learning `Cyclic Sort`

## 10. Binary Search and Its Algorithms

- Binary Search on Sorted Arrays
- Variations of Binary Search
- Binary Search on Infinite Arrays
- Binary Search in Rotated Sorted Array
- Binary Search on 2D Matrix
- Real-World Use Cases of Binary Search

## 11. Hashing (Set and Map) in JavaScript

- Understanding Hashing in JavaScript - `Set` , `Map` \*
- Working with `Set` in JavaScript
- Methods in Set - `add(value)` , `delete(value)` , `has(value)` , `clear()` , `size`

- Working with `Map` in JavaScript
- Methods in Map - `set(key, value)` , `get(key)` , `delete(key)` , `has(key)` , `clear()` , `size`
- Learning Algorithms Using `Set` & `map`

## 12. Linked List in JavaScript

- Understanding Linked List - `Data` , `Pointer`
- `Singly Linked List.`
- `Doubly Linked List.`
- `Circular Linked List.`
- Creating a Node in Linked List:
- Building a Linked List:
- Traversing a Linked List:
- Operations on Linked Lists - `Insertion` , `Deletion` , `Searching`
- Algorithms Using Linked Lists

## 13. Queue in JavaScript

- Implementation of Queue by Linked List and Array
- Working with Queues - `Basic Queue` , `Circular Queue`
- Operations on Queues - `Enqueue` , `Dequeue` , `Peek` , `IsEmpty` , `Size`
- Algorithms Using Queues
- Applications of Queues

## 14. Stack in JavaScript

- Understanding Stacks in JavaScript
- Implementation of Stack by Linked List and Array
- Working with Stacks
- Operations on Stacks - `Push` , `Pop` , `Peek` , `IsEmpty` , `Size`
- Algorithms Using Stacks

- Applications of Stacks

## 15. Advanced Problems on Recursion and Backtracking

- Understanding Advanced Recursion and Backtracking
- Key Problems and Algorithms like [N-Queens Problem](#) , [Sudoku Solver](#) , [Subset Sum](#) , [Word Search](#)
- Optimizing Recursive Solutions with Backtracking
- Challenges with Recursion and Backtracking
- Applications of Recursion and Backtracking

## 16. Tree

- Understanding Binary Trees
- Types of Binary Trees - [Full Binary Tree](#) , [Complete Binary Tree](#) , [Perfect Binary Tree](#)
- Key Terminology in Binary Trees - [Node](#) , [Root](#) , [Leaf](#) , [Height of a Tree](#) , [Depth of a Node](#) , [Level of a Node](#)
- Binary Tree Operations - [Insertion](#) , [Deletion](#) , [Traversal](#) , [Searching](#)
- Binary Tree Algorithms - [Height](#) , [Diameter](#) , [LCA](#) , [Symmetry Check](#)
- Applications of Binary Trees

## 17. Binary Search Tree (BST):

- Understanding Binary Search Tree
- Properties of Binary Search Tree
- BST Operations -
- Binary Search Tree Algorithms
- Applications of Binary Search Tree
- Advantages of Binary Search Tree

---

# Aptitude and Reasoning

# Classic Chapters

## 1. Percentage

- Learn **tips and tricks** for percentages.
- Solve **basic** , **medium** , and **advanced** questions.
- Practice **MCQs** to master percentages.

## 2. Profit and Loss

- Concepts of Profit and loss
- Relationship between **cost price** , **selling price** , and **mark-up price** .
- Solve practical scenarios involving **discounts** , **successive transactions** .
- Sharpen your skills with **MCQs** to prepare for competitive exams.

## 3. Simple Interest

- Master the **formula** for calculating simple interest.
- Differentiate between **principal** , **interest rate** , and **time period** .
- Solve **case-based problems** related to borrowing and lending.
- Practice **MCQs** for thorough preparation

## 4. Compound Interest

- Understand the **growth** of investments and savings.
- Differentiate between **simple interest** and **compound interest** .
- Solve problems with **annual** , **semi-annual** , and **quarterly** compounding.
- Practice **MCQs** for preparation.

## 5. Ratio and Proportion

- Grasp the basics of **ratios** .
- Solve problems on **proportional relationships** .
- Analyze scenarios involving **scaling** , **sharing** , and **dividing** quantities.

- Practice **MCQs** for preparation.
- 

## Number Related Topics

### 1. Number System

- Understand the classification of **natural numbers** , **whole numbers** , **integers** , **rational numbers** , and **irrational numbers** .
- Master **divisibility rules** , **factors** , **multiples** , and **place value** .
- Practice **MCQs** to improve understanding and problem-solving speed.

### 2. HCF and LCM

- Learn techniques to find **HCF** and **LCM** .
- Understand their applications in **scheduling** and **resource sharing** .
- Solve word problems involving **time** , **distance** , and **recurring patterns** .
- Practice **MCQs** for competitive exam preparation.

### 3. Average

- Understand **averages** and their significance.
  - Solve problems on **weighted averages** , **missing numbers** , and **group data** .
  - Apply averages in **performance analysis** and **time management** .
  - Practice **MCQs** to enhance speed and accuracy.
- 

## Speed Work and Time Related Topics

### 1. Work and Time

- Understand the relationship between **work** , **time** , and **efficiency** .
- Solve problems involving **individuals** or **groups** working together.
- Analyze scenarios like **alternating work schedules** and **work completion rates** .
- Practice **MCQs** problems.

## 2. Pipes and Cisterns

- Understand the analogy between **pipes** and **work-time**.
- Solve problems with **multiple pipes** working together or alternately.
- Address challenges like **leaks** or partial closure.
- Practice **MCQs** to improve your skills.

## 3. Speed, Distance, and Time

- Master the formula: **Speed = Distance / Time**.
- Solve problems on **relative speed**, **average speed**, and **varying speeds**.
- Practice **MCQs** questions.

## 4. Problems on Trains

- Calculate the time for a train to cross **poles**, **platforms**, or other trains.
- Apply **relative speed** in train-related problems.
- Solve problems with trains of different **lengths** and **speeds**.
- Practice **MCQs** questions.

## 5. Boats and Streams

- Understand the impact of **stream direction** (upstream, downstream) on speed.
- Solve problems on **relative speed** and **effective speed** in flowing water.
- Analyze scenarios like **rowing competitions** or **river crossings**.
- Practice **MCQs** to test your understanding.

---

## Probability and Combinations

### 1. Permutations and Combinations

- Understand the difference between **permutations** (arrangement) and **combinations** (selection).
- Learn key **formulas** and techniques for calculating arrangements and selections.

- Solve problems with **factorials** , **repetition** , and **circular permutations** .
- Practice **MCQs** to improve problem-solving skills.

## 2. Probability

- Understand **probability** as a measure of likelihood.
  - Learn **formulas** for calculating probability in events.
  - Practice **MCQs** to improve proficiency.
- 

## Progressions

### 1. Arithmetic Progression (AP)

- Understand **Arithmetic Progression** with a constant difference.
- Derive formulas for **general term (an)** and **sum of n terms (Sn)** .
- Apply **AP** in real-life problem solving.
- Solve problems on **missing terms** , **specific terms** , and **sum of series** .
- Practice **MCQs** and concept-based questions.

### 2. Geometric Progression (GP)

- Understand **Geometric Progression** with a constant ratio.
  - Solve problems on **missing terms** , **specific terms** , and **sum of series** .
- 

## Miscellaneous Topics

### 1. Calendar

- Understand **days** , **months** , **leap years** , and **century years** .
- Learn **Odd Days** concept and calculation for day of the week.
- Use key **formulas** to find the day for any given date.
- Solve problems on **repeating calendar years** and calendar-based tricks.
- Practice **MCQs** and scenario-based questions.



## 2. Clocks

- Understand **clock structure** , **minute hand** , **hour hand** , and their movements.
  - Solve **angle problems** between clock hands.
  - Solve problems on **overlaps** , **right angles** , and **opposite directions** .
  - Practice **clock puzzles** and time calculation problems.
  - Practice **MCQs** and puzzle-based questions.
- 

# Logical Reasoning

## 1. Direction Sense

- Understand **directions** (North, South, East, West) and final direction after movements.
- Track **movements** and **turns** (right/left) to find final position.
- Solve problems with **multiple directions** and movement patterns.
- Practice **MCQs** for speed and accuracy.

## 2. Blood Relation

- Identify relationships like **father** , **mother** , **brother** , **sister** .
- Analyze clues to trace **family connections** .
- Solve problems with **family trees** and complex relationships.
- Practice **MCQs** to improve deduction skills.

## 3. Syllogism

- Understand **logical reasoning** and conclusion deduction.
- Break down **premises** to check conclusions.
- Work with **All** , **Some** , **No** premises.
- Solve **MCQs** to identify valid/invalid conclusions.

## 4. Arrangements

- Learn to arrange **people** or **objects** based on conditions.
- Apply **constraints** like sitting together or specific positions.
- Solve problems with multiple **arrangement conditions**.
- Practice **MCQs** to strengthen understanding.

## 5. Series

- Understand **number sequences** and identify next terms.
  - Recognize patterns like **arithmetic progressions**, **geometric progressions**.
  - Solve problems with varying **series types** and difficulty.
  - Practice **MCQs** to improve pattern recognition.
- 

# Verbal Reasoning

## 1. Sentence Ordering

- Practice **MCQs** to improve sentence **ordering** skills.

## 2. Error Identification

- Practice **MCQs** to sharpen error **spotting** and correction.

## 3. Sentence Improvement

- Practice **MCQs** to improve sentence **quality**.
-