**The JavaScript Induction Course for Developers**

**Duration 23 Days (8 Hrs/Day)**

**Course Description**

This course is designed for the entry level professionals to provide them Conceptual and Hands-on training on JavaScript full-stack application development. This course targets to the end-to-end application development using JavaScript technologies and databases servers. The course covers following technologies

* JavaScript
* ES 6
* Node.js
* React
* Redux
* Relational database programming using MySQL
* NoSQL Database with DynamoDB
* Microservices development using AWS Lambda
* Application Management with GitHub and Jenkins

**Course Coverage**

The course will be completed with following steps for each topic

* Conceptual explanation
* Concept demonstration with syntax
* Application development with Hands-on Lab

**Course Takeaway**

After completing the course attendees will be ready to work on JavaScript full-stack application development.

**Software and Hardware requirements**

**Software**

* OS: Windows 8, 10, Linux 18.0+, Mac OS
* IDE: Visual Studio Code. Download it from <https://code.visualstudio.com>
* Runtime: Node.js. Download it from <https://www.nodejs.org>
* Database: MySQL
* Valid AWS Account

**Hardware**

* Core i7 + 64 bit Machine
* 8 GB Minimum RAM
* 50 GB Free Hard Disk

**Section 1: The JavaScript Programming with Basic Front-End Concept**

This section covers the JavaScript and ES 6 programming concepts for front-end application development. After the completion of this section, attendees will be ready with JavaScript programming concepts so that they can use this knowledge for JavaScript full-stack application development.

**Day 1: JavaScript a Dynamic Programing Language**

|  |
| --- |
| * What is JavaScript and why need it?   + Modern Consumer-Oriented Application   + Trends of modern application development   + Browser Runtime   + Cross-Browser Application Requirements * Understanding JavaScript Object Model (Hands-on-Lab)   + The **document** object   + The **window** object   + Creating JavaScript Code file   + Loading JavaScript Code file in browser * JavaScript basics (Hands-on-labs)   + Variable declarations   + Scopes of the variable   + JavaScript Operators   + Data and value equality   + Conditional statements   + Loops   + Number methods   + String methods   + JavaScript Objects     - Object equality     - Object assign * JavaScript functions (Hands-on-labs)   + Calling functions |

**Day 2: JavaScript Programing**

|  |
| --- |
| * Using JavaScript with html (Hands-on-labs)   + Understanding events   + DOM Elements   + DOM Events   + Subscribing to events and Eventing concept of Browser   + The event callback * Accepting data from Html DOM and processing it with JavaScript (Hands-on-labs) * JavaScript Forms (Hands-on-labs)   + Validations   + DOM attributes * JavaScript Immediately Invocable Function Expressions (Hands-On-Labs) * Reference Functions for Defining objects |

**Day 3: Using HTML 5 and CSS 3 for Modern UI Application Development**

|  |
| --- |
| * Understanding the UI Programming with HTML 5 and CSS3   + What is HTML 5 and why is it needed?   + HTML 5 Form-Elements   + Using HTML 5 Browser APIs (Hands-on-labs)     - LocalStorage     - SessionStorage     - IndexedDB     - Drag-Drop * Working with CSS3 (Hands-on-labs)   + Understanding CSS 3   + Using CSS Classes     - Selectors     - Filters   + Inline and External CSS   + Applying CSS Dynamically using code   + Base and Derive CSS classes |

**Day 4: Implementing Rich UX using CSS Frameworks**

|  |
| --- |
| * What is an advantage of CSS Framework? * Using Bootstrap (Hands-on-Lab)   + Installing and Configuring bootstrap   + Using Standard classes e.g. Container, table, button, etc.   + Media queries * The concept of Less and SCSS (Hands-on-Labs)   + Creating rich and interactive Styles using less   + Converting less into css * Knowing about Flexbox layouts for Interactive (Hands-on-Lab)   + Containers   + Wrapping   + Alignments |

**Day 5: Using Modern JavaScript for Application Development (ES 6)**

|  |
| --- |
| * Why need ES 6? * Using ES 6 Transpiler   + Configuring ES 6 project   + Understanding babel   + Transpiling the ES 6 into ES 3 * Understanding ES 6 Scopes using **let** (Hands-on-Labs)   + ES 6 Type System * Understanding ES 6 Class (Hands-on-Labs)   + Array methods     - Filter     - Foreach     - Sort     - Reverse     - Find   + String Class     - ES 6 String methods   + Date class   + Math Object * Working with ES 6 Collections   + Set, WeakSet   + Map, WeakMap |

**Day 6: Using Modern JavaScript for Application Development (ES 6)**

|  |
| --- |
| * Understanding Object Oriented Programming with ES 6 (Hands-on-Labs)   + Creating classes   + Constructor   + Methods   + Instances * Template String * Arrow Functions * Destructuring * Iterations * Rest parameters * Object.keys * Testing and Debugging Js Applications using Jasmine |

**Day 7: JavaScript ES 6 Advance programming**

|  |
| --- |
| * ES 6 Generators   + Functions   + Control Flow   + Methods * Symbols * Proxying * Reflection * Promises   + Async and Await Programming   + Http Operations using Fetch * JavaScript Modules   + Code Splitting   + Browserfy   + Export and Import |

**Section 2: The Database programming concepts with relational database programming using MSQL and NoSQL with DynamoDB**

This section focuses on the server-side programming where attendees will be able to work on creating database and table using MySQL database. In MySQL, the focus will be on database design, creating tables and performing DDL and DML operations along with indexing, sorting etc. In DynamoDB, the focus on NoSQL database along with collections will be explained. (Note: This section is kept limited only on database creations and read/write operations advance concepts are not covered here. In case of need of advanced concepts, the duration will be increased.).

**Day 8: Understanding the MySQL**

|  |
| --- |
| * Understanding Relational Database with MySQL   + MySQL Features   + Creating Database (Hands-on-labs)     - Database Design   + Datatypes   + Creating Tables     - Columns     - Constraints     - Relationships     - Normalizations * Working with Queries (Hands-on-Labs)   + DML   + DDL   + Join   + Sorting   + Indexes   + Transactions   + Resexps * Optimizing Database Queries (Hands-on-Labs) |

**Day 9: NoSQL Database with DynamoDB**

|  |
| --- |
| * What is NoSQL Database and why is it needed? * Difference between Relational and NoSQL database * Creating DynamoDB in AWA (hands-On-Labs)   + Datatypes     - Scalar     - Document     - Set   + Creating Table   + Load Table   + Create, Read, Update, Delete Items   + Batch writing   + Batch Retrieve   + Indexing   + Best Practices |

**Section 3: Server-Side JavaScript Application development using Node.js**

This is the most important section of on the training program. In modern enterprise web applications, the server-side development plays a very important role. The Node.js is the server-side JavaScript application development runtime. This is a cross-platform runtime where we can run JavaScript application workflows in asynchronous execution mode. Node.js is package rich server-side runtime. One of such packages is Express.js. Express.js is a web application development framework used on Node.js. This framework ca be used for Single-Page and Multi-Page applications. We can also use Express.js for creating REST APIs. Node.js also have a support for Object-Relational-Mapping (ORM) using Sequlize package. We use this package for working with database programming.

**Day 10: Understanding Node.js**

|  |
| --- |
| 1. What is Node.js and why to use it?    1. Modern Web App    2. Need of Asynchronous Applications    3. World of JavaScript application development    4. Application scenarios for Node.js applications 2. Node.js Architecture    1. Event Loop    2. Internal Worker threads    3. Asynchronous Execution    4. Single-Thread Processing Model    5. Concurrency 3. Creating Node.js application (HOL)    1. Node.js Project structure    2. Installing Require NPM Packages    3. Package.json    4. ES 6 Coding with Node.js    5. Transpiling the Code    6. Executing and Debugging the application 4. Understanding Node.js Modules (HOL)    1. http    2. path    3. fs    4. creating Node.js web server       1. Managing Static files in web server       2. Handling Simple Routes    5. creating custom modules 5. Making Asynchronous calls to externally hosted services (HOL)    1. Understanding **Promises** comparing it with Callback hell    2. Using Q for Promises    3. Using Async and Await calls |

**Day 11: Creating REST APIs using Express.js**

|  |
| --- |
| 1. Using Express.js for Web Application Development (HOL)    1. What is Express.js    2. Using Express.js for creating REST APIs    3. Connecting to Relational Database from REST APIs       1. Using MySql          1. Using ORM (Sequlize/node-orm2)          2. Code-First approach for generating tables using models             1. Relations             2. Constraints          3. Database first             1. Generating Models based on database          4. Performing CRUD Operations       2. API Documentation with Swagger |

**Day 12: Creating REST APIs using Express.js and AWS DynamoDB and Session with Security**

|  |
| --- |
| 1. Accessing DynamoDB using REST APIs for Node.js Application    * 1. Managing Connections      2. Performing CRUD operations 2. Session Management using Node.js Applications (HOL)    1. Using Expression Sessions    2. Managing User State on Server-Side for Accessing Server objects 3. Securing REST APIs using JWT Authentication (HOL)    1. Configuring JWT with Secret    2. Creating and Validating Tokens    3. Accessing secure APIs from client applications |

**Section 4: Node.js Microservices with AWS Lambda**

The modern enterprise apps have one of the frequently followed architecture as Microservices. It is an architecture which helps to address and build most complex requirements of the server-side applications by segregating server-side workflows in separate services. This section focuses on Microservices application development using Node.js.

**Day 13: Microservices in Node.js**

|  |
| --- |
| * What are Microservices?   + Need of Microservices   + Application Architecture     - 2-Tier/3-Tier Application development   + Monolithic Architecture   + Advantages of Microservices   + Limitations of Microservices * Creating Microservices using Node.js (Hands-on-Labs)   + Understanding Docker   + Configuring Dockers for the Node.js application   + Dockerizing Node.js application * Communication across Microservices using RabbitMQ (Hands-on-labs)   + Docker-Compose for deployment |

**Day 14: Microservices on AWS**

|  |
| --- |
| * Deploying Microservices in AWS with Docker (Hands-on-Labs)   + Creating EKS clusters for managing Microservices * Creating Serverless Microservices on AWS using Node.js (Hands-on-Labs)   + What is AWS Lambda   + Managing functions   + Invoking functions   + API for Lambda Functions   + API Gateway * API Caching with Redis Cache and Node.js * Application Level Caching * Database level Caching * Load Balancer * Elastic Search integration |

**Section 5: Working with JavaScript practices and Testing Node.js applications**

This section provides more information of implementing the JavaScript applications with patterns and practices of coding so that the applications are developed by considering Test-Driven-Development (TDD) approach.

**Day 15: JavaScript Advanced Programming**

|  |
| --- |
| * Understanding JavaScript Design Patterns   + Why to learn JavaScript Design Patterns?   + Module Pattern     - CommonJS   + Singleton Pattern   + Observer Pattern   + Mediator Pattern   + Prototype Pattern   + Command Pattern   + Factory Pattern   + Façade Pattern |

**Day 16: Testing Node.js applications**

|  |
| --- |
| * Testing Node.js Applications (Hands-on-Labs)   + Unit Testing using Jasmine   + Mocha and Chai   + REST API Testing * Node.js Coding Practices   + Scripts in package.json   + Npmrc file   + Environment variables   + Async and await   + Handle errors   + Auto restart apps   + Logging   + Monitor |

**Section 6: JavaScript Front-End application development using React.js and Application State management using Redux**

The React is a View Library by Facebook for Virtual DOM and uses Redux framework to manage store, Event dispatcher, Subscription. React.js is view implementation for building composable UI applications. The main part of the React application is Components. The component is an HTML abstraction with data and events for building reactive UI applications. As the front-end applications becomes more complex, we need to use application state management using Redux. Redux library provides a mechanism of Store that stores data in browser’s process. This helps to resolve complex data sharing use case across multiple React components.

Day 17: Understanding and Using ReactJS

|  |
| --- |
| * What is Reactjs and why to use it? * Project creation using React CLI * The ES lint for code Quality Checks * React Js Features (HOL)   + Component     - The component class and its properties     - Creating Component Using ReactJs APIs   + Define HTML Element   + Render Method   + Defining Component Properties     - State and Props     - Applying Properties for the component from Root Render method * Understanding JSX (HOL)   + Creating Elements in the Component Class   + Defining Properties for the Element.   + Integrating third-party JavaScript / CSS library with React.js * React Events (HOL)   + Event Binding * Creating reusable components for UI Composition   + Data sharing across components   + Context |

Day 18: Using React Forms

|  |
| --- |
| * Using React.js Forms (HOL)   + Using Form Element   + Managing Submit   + Controlled Components   + Component Lifecycle     - * Methods invoked while loading a component       * Methods invoked on re-rendering a component       * Methods invoked while unloading a component   + Un-Controlled Components     - * Using Refs   + Validations * Higher Order Components (HoC) (HOL)   + Creating a Pure Function for Hoc.   + Managing Data Flow   + Instantiating Component |

Day 19: React Routing and Ajax calls

|  |
| --- |
| * Developing Single Page Application using React (HOL)   + React Router   + Path Matching   + Routing with Multiple Components   + Parameterized Routing * Managing Ajax Calls   + Making Secure Calls to Ajax   + Understanding and using axios library   + User Based Authenticated Navigation   + Managing Tokens locally in React Client Application |

Day 20: React Hooks and Application Development

|  |
| --- |
| * Understanding and using React Hooks (Hands-on-labs)   + Functional components   + useState   + useContext   + useEffects   + useReducer   + Hooks rules   + Building Custom hooks   + Best practices of using react hooks * Code splitting * Accessibility * Profilers * Portals * Error Boundaries |

Day 21: Working with Redux

|  |
| --- |
| * Understanding Redux (HOL)   + Store for managing Application State and Event Broadcasting   + Action for Creating Action, Dispatching Action, Action Binders   + Using Redux hooks for Action Dispatching   + Using Selectors for reading data from Store   + Reducer for Creating and combining reducers   + Injecting Redux Store using Providers * Selectors and Middlewares   + Using Saga   + GET/POST calls with generators |

Day 22: React.js Server-Side Rendering and Testing applications

|  |
| --- |
| * Understanding Server-Side Rendering using Next.js (HOL)   + Installing Next.js   + Navigate between Pages   + Shared Components   + Dynamic Pages   + Clean Urls and Server-Side Support   + Fetching Data for Pages * React.js performance considerations (HOL) * Testing React Applications using Jest and Enzyme (HOL)   + Dom Testing   + Event Testing |

**Section 7: Understanding CI/CD pipelines for JavaScript application development and Git integration**

The JavaScript enterprise application development that includes multiple teams with Micro Module development must be streamlined using an appropriate application management. This section provides an application development guideline for Code versioning management, check-in checkout policies of JavaScript application code with GIT and Jenkins

**Day 23: Working with Gitlab CI/CD Pipeline**

|  |
| --- |
| * Code-Documentation for JavaScript Application   + React-Docgen   + Tooling +Script   + Create/ Fork a project   + Branch Creation   + Add File and Rebase * Configuration DevOps with JavaScript Full Stack Applications   + User and Group Management   + CI/CD Pipeline   + Integrating with Git   + Managing Extensions * Issue Tracker   + Create Issue   + Merge request   + Referencing issue * Source Control with GitHub   + Versioning   + Check-in   + Check-out |

**Day 24: The Final Day for Project evaluation and Application Deployment**

|  |
| --- |
| * Project Evaluation   + Code Quality Check   + Code Review   + Coding Styles   + Output Validations   + Personal Quiz * The Application Deployment Scenarios (Hands-on-Labs)   + Standalone application using Docker   + Using API Apps with Amplify with Git Integration |

**Important Note: The course topics are designed as per the discussion which I had with experts from customer’s side. For this batch if you want to add more topics then the duration will be increased.**

**Some of the topics which are taken as overview for the induction are as follows**

1. **Node.js ORM with Sequlize and the token-based authentication concepts are included on application development purpose. If concept like ORM is to be covered in detail, then 1 day needs to be added in the course.**
2. **The course covers all topics. In case of more topics required. Then the duration will be increased.**
3. **The day-wise breakup is for the reference purpose and will be followed, but in-case of queries from attendees may extend a topic to the next day, in that case as a trainer I will use my liberty to shuffle topics, but I will make sure that all topics will be covered in detail.**
4. **The code sharing to the attendees will be done using Git repository.**
5. **Attendees have to share the Hands-on labs using their Git links.**