MERN STACK

Assessments-

- Pre Assessment- Just to check the knowledge of the participants.
- Mid Assessment/Module Assessment
- Post Assessment

Mode of Delivery-

- Virtual Instructor Led Training- will be delivered using any virtual tool as per agreed by client.
- All trainings will be delivered using LMS.
- LMS access will be given to all Learners

Duration: - 22 Days

Training Contents –

Mongo DB, Express , REACT JS, NODE JS

Who this course is for?
- Front end Developers, Developers, Web Designers, Full stack developers
- Freshers/Graduates/ Software Developers
- Migrate using React for your application development.
- One who wants to be Mern stack developer?

NODE JS

- Introduction to Node JS
- Introduction
- What is Node JS?
- Advantages of Node JS
- Traditional Web Server Model
- Node.js Process Model

- Setup Dev Environment
- Install Node.js on Windows
- Installing in mac os
- Working in REPL
- Node JS Console

Event Loop
- Callback Concept
- Global Objects
- Streams
- Buffers
- Utility Modules

Node JS Modules
- Functions
- Buffer
- Module
- Module Types
- Core Modules
- Local Modules
- Module.Exports

Node Package Mananger
- What is NPM
- Installing Packages Locally
- Adding dependency in package.json
- Installing packages globally
- Updating packages

Creating Web server
- Creating web server
- Handling http requests
- Sending requests

- File System
- Fs.readFile
- Writing a File
- Writing a file asynchronously
- Opening a file
- Deleting a file
- Other IO Operations

Debugging Node JS Application

- Core Node JS debugger
- Debugging with Visual Studio

Events

- EventEmitter class
- Returning event emitter
- Inhering events

Getting Started with Node (Async patters, Even Loop, Worker Threads)

Modern JavaScript along with Modules and Concurrency

Using MongoDB with Node.js

Working on the HTTP Requests and responses making API Requests.

Understanding OAuth/SAML/JWT with Node.js

Connect mongodb with node js and Working with the Aggregation Pipelines

Data Retrieval and Modification Techniques

REST API's

Micro Front End using React, Node

REACT JS

React Introduction

- Overview of frameworks, libraries for client side Web applications
- React introduction
- Environment Setup for React Application
- Understanding NPM commands
- VS Code extensions for ES6, React

- **React Essential Features and Syntax**
  - React App Project Directory Structure
  - Overview of Webpack, Babel
  - React Component Basic
  - Create React Component
  - Understanding JSX

- Limitations of JSX
- Working with Components and Reusing Components

- **React Components, Props and State**
  - Understanding and using Props and State
  - Handling Events with methods
  - Manipulating the State
  - Two way data-binding
  - Functional (Stateless) VS Class (Stateful) Components
  - Parent – Child Communication
  - Dynamically rendering contents
  - Showing Lists, List and keys
- **Styling Components**
  - CSS Styling
  - Scoping Styles using Inline Styles
  - Limitations of inline styes
  - Inline Styles with Radium
  - Using Psuedo classes/media quries with inline styles
  - CSS Modules, importing css classes
  - Adding Bootstrap, Semantic UI to React apps
  - Using react-bootstrap, reactstrap packages
- **Debugging React Apps**
  - Understanding React Error Messages
  - Handling Logical Errors,
  - Debugging React apps using google developer tools and React DevTool
  - Understanding Error Boundaries
- **React Component life cycle**
  - Updating life cycle hooks
  - PureComponents
  - React's DOM Updating Strategy
  - Returning adjacent elements
  - Fragments
- **React Component in Details**
  - Higher Order Components
  - Passing unknown Props
  - Validating Props
  - Using References
  - React Context API
  - Updated LifeCycle hooks (16.3)
  - Best practices for React Projects
  - Demo apps

- **HTTP Requests/Ajax Calls**
  - HTTP Requests in React
  - Introduction of Axios package
  - HTTP GET Request, fetching & transforming data
  - HTTP POST, DELETE, UPDATE
  - Handing Errors
  - Adding/Removing Interceptors
  - Creating/Using Axios intances
- **React Routing**
  - Routing and SPAs
  - Setting Up the Router Package
  - react-router vs react-router-dom
  - Preparing the Project For Routing
  - Switching Between Pages, Routing-Related Props
  - The "withRouter" HOC & Route Props
  - Passing & extracting route/query parameters
  - Using Switch to Load a Single Route
  - Navigating Programmatically
- **React Forms and Form Validation**
  - Creating a Custom Dynamic Input Component
  - Setting Up a JS Config for the Form
  - Dynamically Create Inputs based on JS Config
  - Adding a Dropdown Component
  - Handling User Input
  - Handling Form Submission
  - Adding Custom Form Validation
  - Fixing a Common Validation
  - Adding Validation Feedback
  - Showing Error Messages
  - Handling Overall Form Validity
- **Deploying React App to the Web**
- **Testing React apps with JEST**

Configuring and working with Typescript in React application

Declaring component using Props (Using Hooks) Typescript

Understanding Server Rendering React Components

Optimize Performance for React

Implementing Forms and Styling React Components

TeamLease EDTECH
Making India Employable

Calling APIs with React and Managing Large Datasets

React Security: Best Practices

Express js

Introduction of ExpressJs

- o What is ExpressJS
- o How Express.js works
- o Installation of Express.js
- o Basic Example

- **Templating Engines**
  - o Introduction
  - o pug Templating Engine
  - o Working with Tags in pug
  - o Working with id and classes in pug
  - o Attributes and Nesting Tags in pug
  - o Using if & unless in pug
  - o Using for & each in pug
  - o Using case & mixins in pug
  - o Include and Extend in pug
  - o EJS Templating engine
  - o Express Handlebars
- **Working with Express.js**
  - o Introduction
  - o Introduction to Express.js
  - o Connect Module
  - o Express.js Installation
  - o app.js
  - o Steps for creating Express.js Application
  - o application, request, response object properties & methods
- **Request/Response in Express.js**
  - o Request-params,body,files,route,header,get
  - o Response-render,locals,status,json,redirect
- **Using middleware**
  - o Types of middleware
  - o Application level middleware
  - o Express-json,session,logger,compress
  - o Router level middleware
  - o Built-in middleware

- Third party middleware
- Express 4.0 Router

MONGO DB

MongoDB – Overview

- Understand what is NOSQL
- Describe CRUD
- State the types of NOSQL
- Explain what is Aggregation
- Describe Replication & Sharding

- **CRUD Operations**
  - Understand what are Crud Operations
  - Explain what is Upsert
  - Describe Query Interface
  - List the Comparison Operators and Logical Operators
  - State what are Wrapped Queries and Query Operators
- **Basic Operations**
  - Crud Operations
  - Basic Operations With Mongo Shell
  - Data Model
  - JSON
  - BSON
  - MongoDB – Datatypes
  - BSON Types
  - The _id Field
  - Document
  - Document Store
  - Blog: A Bad Design
  - Blog: A Better Design
- **Aggregations**
  - Types of Aggregations
  - What is Aggregation?
  - The Aggregate() Method
  - Pipeline Concept
  - Pipelines
  - Pipeline Flow
  - Pipeline Operators
  - $match, $unwind

- o $group, $project
- o $skip, $limit
- o $sort, $first
- o $last, $sum

- **Indexing**
  - o Understand about Indexes
  - o Understand different types of Indexes
  - o Understand properties of Indexes
  - o Explain Plan in MongoDB
  - o Mongostat
  - o Mongotop
  - o Logging Slow queries
  - o Profiling
- **Replication and Sharding**
  - o Understand about Replication
  - o Purpose of Replication
  - o Understand Replica Set
  - o Sharding
  - o Sharding Mechanics
  - o GridFS

During the training we will also cover below points

Classification: Internal

Docker overview

CI CD process of Deployments on AWS/AZURE

*Query Session, Assessment & Feedback