**NodeJS – Unit Testing -AWS**

This training talk to JavaScript developers how to create server-side applications with JavaScript and Node.js, culminating with an MVC application built on the Express framework that queries databases and calls back-end web services.

**Duration – 20 Days**

|  |  |
| --- | --- |
| **Duration** | **Modules** |
| Week 01 (05 Days) | JavaScript, Node, Express, MongoDB |
| (02 Days) | Practice, Code Challenge, Assignment |
| Week 02 (05 Days) | Authentication, Network Programming and GraphQL |
| (02 Days) | Practice: Creating POC using GraphQL Server |
| Week 03 (05 Days) | Unit Testing and AWS and Deployment |
| (Day 01) | Test & Deploy NodeJS App on AWS |

**NODE.JS TRAINING OBJECTIVES**

* Learn why server-side JavaScript is useful
* Install Node.js
* Create basic web applications with Node.js
* Build an HTTP server using the core modules in Node.js
* Use stream I/O to efficiently serve the web pages
* Create modules to organize the server
* Test the reliability of the application with unit tests
* Convert the application to an MVC framework using Express
* Interface to a MongoDB database and a web service

**NODE.JS TRAINING PREREQUISITES**

Attendees should have a fair knowledge of JavaScript as well as basic HTML and CSS.

**SOFTWARE NEEDED FOR EACH PC:**

* A recent version of Google Chrome or Mozilla Firefox
* A local installation of Node.js
* MongoDB Installer / Access to Mongo Cloud
* Admin privileges to install additional NPM features
* A JavaScript development IDE of your choice (Visual Studio Code)
* Open internet

**NODE.JS TRAINING OUTLINE**

**Week 01 (Day 01 to Day 05):**

* JavaScript Objects, Functions, and Events
  + How to use objects to work with data
  + How to use functions
  + How to handle events
* Closures & Callbacks
  + Introduction to closures
  + How to use closures
  + How to use callbacks
* ES6: New Features
  + Arrow functions
  + Classes
  + Modules
  + Destructuring
  + Named & Default Parameters
  + Spread & Rest Operators
* Debugging and Error Handling
  + Try … catch block
  + Browser Development Tools
  + Attaching Debugger to Node Process
  + Debugging in VS Code

Handle Asynchronous Programming using Promises / Async..await

* NodeJS : Introduction
  + Foundation
    - The Node.js Platform
    - Installing Node.js
    - Using Node.js to execute scripts
* Node Projects
  + The Node Package Manager
  + Creating a project
  + The package.json configuration file
  + Global vs. local package installation
  + Configuring Node for JavaScript
* HTTP
  + The HTTP protocol
  + Building an HTTP server
  + Rendering a response
  + Processing query strings
* File System
  + Synchronous vs. asynchronous I/O
  + Path and directory operations
  + \_\_dirname and \_\_filename
  + Asynchronous file reads and writes
* Modules
  + Modularization
  + Defining modules with exports
  + Modules are singletons
  + Creating a package
  + Module scope and construction

**Task : CLI Application – Notes App**

* Express
  + The model-view-controller (MVC) pattern
  + Building a front-end controller
  + Defining routes
  + Using REST – GET/POST/PATCH/DELETE
  + Adding middleware
  + Request / Response
* Data Sources
  + How Node.js connects to databases
  + RDBMS databases and NoSQL databases
  + Connecting to NoSQL databases (MongoDB)
  + Performing CRUD operations
  + Building client requests to web services

**Task : CRUD REST APIs for Todo App**

**(Day 06 and Day 07):**

**Practice:** Understanding and demonstrating the skills on ES6+ concepts

**Assignments:**

* Connect Mongo with Express App
* Creating REST endpoints
* Serving HTML / JSON

**Challenge –** Working with Two or more entities for data persistence. Cascading Operation. Technologies used – Express with Mongoose and MongoDB

**Week 02 (Day 08 to Day 12):**

* Creating Views :
  + View Engine (Handlebar / EJS)
* NodeJS Securities
  + Using JWT for API Authentication
  + Installing and using passport.js
    - Using Local Strategy
    - Using Auth0
* Network Programming
  + Understanding Bi-directional Stream
  + Using socket.io library
  + Creating Socket Server
  + Creating socket client
  + Communicating on Channel

**Task : Creating Chattr App to talk in real-time**

* **GraphQL : Introduction – What & Why ?**
* GraphQL Basics: Schemas and Queries
  + What is a Graph?
  + GraphQL Queries
  + Nested GraphQL Queries
  + Creating GraphQL API
  + GraphQL Scalar Types
  + Creating Custom Types
  + Working with Arrays
  + Relational Data: Basics
  + Relational Data: Arrays
* GraphQL Basics: Mutations
  + Creating Data with Mutations
  + The Input Type
  + Deleting Data with Mutations
  + Updating Data with Mutations

**(Day 13 and Day 14):**

**Practice:** Understanding and demonstrating the skills on generating templates dynamically.

**Assignments:**

* User authentication using JWT and Passport
* Creating Socket Server / bi-directional Continuous Communication

**Challenge –** Creating GraphQL Single endpoint. Cascading Operation. Technologies used – Express with MongoDB and GraphQL Server

**Week 03 (Day 15 to Day 19) :**

* Unit Testing :
  + Assert
  + Mocha
    - Installing Mocha
    - Running basic test
    - Mocha API
    - Hooks
    - Single Test run and exclusion
    - Testing promises
  + Using Chai
    - Additional Assertions
    - Chai plugins
    - BDD Style Assertions
* Deploy Node App on Server/Cloud

**AWS Fundamentals**

* Fundamental concepts of cloud computing and including storage, database, networking, virtualization, containers, and cloud architecture
* Create an AWS Free Tier account and launch your first virtual servers (Amazon EC2 instances) on the AWS Cloud
* Create and configure storage services and upload files and objects
* Launch a relational database on Amazon RDS and a NoSQL database using Amazon DynamoDB
* Create a Platform as a Service (PaaS) application on AWS Elastic Beanstalk
* How to use DevOps tools on AWS to automate a continuous integration and continuous delivery (CI/CD) pipeline
* AWS Lambda – What, Why, How? AWS Lambda vs. AWS EC2 vs. Elastic Beanstalk. Create a Lambda, Lambda Functions, Configure/Create Function, Test Lambda Function

**Day 20 :**

**Practice:** Creating pipeline and app deployment.