**MEAN Stack Development Course**

**Learn full-stack web development with Node JS, Express, Mongo DB, & Angular modules.**

**Duration: 15 Full Days [or 25 Days @ 5 Hrs Per Day ]**

**What you’ll learn**

* Build Mean Stack Application
* Build and route a basic SPAs
* Build RESTful API
* Develop Module
* Secure your app
* Model-driven forms

**Who Should Attend**

* Web Developers
* Technical Project Managers
* Technical Leads

**Prerequisites**

**An individual should be familiar with:**

* Basic knowledge of developing web applications
* HTML and CSS web languages
* JavaScript, IDE or Text Editor, Command Line Interface (CLI)
* Server-side development with any programming language.

**Course Content:**

1. **Introduction to Node**

* Installing Node.js
* Node’s Event Loop
* Writing asynchronous code

1. **Advance JS primer**

* Callbacks
* Closures
* Promises
* Async/await

1. **Modularizing code**

* Understanding built-in module
* Techniques for modularizing JavaScript code
* Using require() to modularize application code
* Using npm for third-party modules
* Handling Exceptions

1. **Events and Streams**

* EventEmitter class
* Understanding Streams
* Reading and writing streams
* Using pipe()

1. **Accessing local resources**

* Process Object
* Manipulating File System
* Understanding Buffers

1. **Nodejs and the web**

* Handling web requests
* Building a web server

1. **Working with Express JS**

* Installing Express.js
* Routing
* Parameters and queries in routing
* Building views using view engine of choice (ejs / jade / handlebars)
* Using blocks for layout
* Displaying data
* Working with forms
* Serving files
* Error Handling

1. **Securing web apis with passport**

* Authentication & Authorization using local user
* Protecting API with JWT

1. **Real time communication with socket.io**

* Understand the need of websockets
* Install and setup socket.io
* Develop real time dashboard

1. **Scaling node apps**

* The Child process model
* Exec, spawn, and fork functions
* Using the Cluster module

1. **Unit testing with mocha & chai**

* What is unit testing
* Install and setup mocha and chai
* Write unit tests to test api

1. **Introduction to MongoDB**

* Installing MongoDB
* The current SQL/NoSQL landscape
* Document-oriented vs. other types of storage
* Mongo's featureset
* Common use-cases
* MongoDB databases
* MongoDB Collections
* MongoDB Documents

1. **CRUD Operations in MongoDB**

* Creating documents
* insert()
* update()
* save()
* Querying documents
* find()
* Working with equality
* Query operators
* Building complex queries
* Updating documents
* Deleting documents

1. **Introduction to Mongoose**

* Word on ORM/ODM
* Installing Mongoose
* Connecting to MongoDB from Mongoose

1. **Core concepts of Mongoose**

* Understanding Mongoose schemas and datatypes
* Working with Models
* Using modifiers in schema
* Using virtual fields
* CRUD operations with Mongoose

1. **Extending Models**

* Working with hooks
* Validation of model data
* Creating custom static methods
* Creating custom instance methods

1. **Introduction to Angular**
2. **Why Angular?**

* User Experience similar to a Desktop Application
* Productivity and Tooling
* Performance
* Community
* Full-featured Framework
* Platform for Targeting Native Mobile not just Web Browsers

1. **Understanding Angular Versions**

* AngularJS (Angular 1.x)
* Angular 2
* Angular 4
* Angular 5
* Angular 6
* Angular 7
* Angular 8

1. **Typescript & ES6**

* What is Typescript
* Why Typescript
* Setup and installation
* IDE support
* Scoping using Let and Const Keywords ( ES6 )
* Template Literals ( ES6 )
* Spread Syntax and Rest Parameters ( ES6 )
* Destructuring ( ES6 )

1. **Power of Types**

* Type inference
* Type Annotations
* Number
* Boolean
* String
* Array
* Tuple
* Enum
* Any
* Void
* Null and Undefined
* Never (Typescript 2)

1. **Explore Functions**

* Using types in functions
* Function as types
* Optional and default parameters
* Arrow functions

1. **Classes**

* Inheritance
* Access modifiers
* Getters and setters
* Readonly & static

1. **Interfaces**

* Optional properties and methods
* Strict structural contract
* Extending interface
* Implementing interface

1. **Modules**

* Introduction
* Import
* Export
* Default

1. **Decorators (Typescript Aspect Oriented Programming)**
2. **Working with Angular CLI**

* Angular CLI
* Anatomy of the project
* Debugging Angular apps
* Working with Augury

1. **8 Main Building blocks of Angular (Birds Eye View Of Angular )**

* Modules
* Components
* Templates
* Metadata
* Data binding
* Directives
* Services
* Dependency injection

1. **Angular modules**

* Why modules
* How to create modules
* Built in modules
* Root Module
* feature module

1. **Components**

* @Component decorator
* Component configuration object
* Custom components
* Component with templates
* Inline
* External
* Component with Styles
* Inline
* External

1. **Templating**

* HTML as template
* Data binding
* Interpolation
* Property Binding
* Event Binding
* Two way binding
* Template expressions
* Template syntax
* Attribute, class and style bindings
* @Input()
* @Output
* Template reference variables
* Safe navigation operator

1. **Directives**

* Built-in directives
* Structural directive
* NgIf
* NgFor
* NgSwitch
* Attribute directive
* NgClass
* NgStyle
* NgModel

1. **Pipes**

* Built-in pipes
* @Pipe decorator

1. **Forms**

* @angular/forms library
* Template driven forms
* Form and field validation
* Validation check with ng-pristine,ng-dirty, ng-touched, ng-untouched, ng-valid, ng-invali
* Show and hide validation error messages
* Form submission with ngSubmit
* Reactive/ Model drive forms
* ReactiveFormsModule
* FormGroup, FormControl classes
* FormBuilder for easy form building
* Validations using Validators
* Setting form model using setValue and patchValue
* Use FormArray to build repeated from controls or form groups

1. **Dependency Injection**

* Why DI
* @Injectable decorator
* Custom service development
* Registering the service with NgModule using providers key
* Provider Types
* Class
* Factory
* Value

1. **Routing And Navigation**

* @angular/router library
* Configure routes
* RouterModule.forRoot and RouterModule.forChild
* RouterOutlet, RouterLink, RouterLinkActive
* Nested Routes
* Parametrized routes
* Route guards

1. **RxJs Primer**

* Why RxJs
* Observable interface
* Streams
* Operators
* Subscription
* Subject
* Schedulers

1. **HTTP Deprecated & HttpClient**

* Setup installing the module
* Making a request for JSON data
* Typechecking the response
* Error handling
* Sending data to the server
* Making a POST request
* Configuring other parts of the request

1. **New Features in Angular 8**

* New features in angular cli
* ng update
* ng add
* Angular Elements
* Updates to @angular/material
* CLI workspaces
* Building libraries
* Tree shakable providers
* RxJs 6 Support
* How to upgrade to Angular 6

1. [**Testing**](https://www.knowledgehut.com/web-development/mean-stack-development-training#collapse-curriculum-140)

* Testing in Typescript
* Testing Component
* Testing Service/Provider
* Testing Pipe

1. **Deployment**

* Manually
* Using the Angular CLI with Ahead-Of-Time (AOT) Compilation and Tree-Shaking (removing unused library code)