Course Duration For Angular Training: 2 Months

Objectives For Angular Training:

- Get familiar with Angular framework and how Angular works.
- Learn about the changes in Angular 1, 2, 4 and the new updates in Angular5.
- Get up to speed with TypeScript
- Implement single-page applications
- Use various Angular features like directives, components and services
- Implement one-way as well as Two-Way Data Binding
- Handle Angular forms using both Template-driven as well as Reactive Approach
- Use Angular modules and optimize apps
- Learn & Implement Dependency Injection
- Change pages with Routing
- Understand & use Observables
- Learn about Authentication & how it works in SPAs Implement a functional web application using Angular Running tests with CLI and testing dependencies

Eligibility For Angular Training:

Knowledge of HTML, CSS, Javascript or any programming language.

Angular 4/5syllabus:

HTML Fundamentals

- HTML Introduction How Web Works?
- What is a Web Page?
- HTML Basic HTML Fundamentals
- HTML Attributes
- HTML Styles HTML Controls
- HTML Formatting
- HTML Forms

CSS Overview

- CSS Syntax
- CSS Id & Class
- CSS Styling
- CSS Box Model
- CSS Margins, Dimensions, Display
- CSS Navigation Bar

Typescript

- Typescript
- What is Typescript

- First Typescript Example
- Basic Data Types & Variables Using Types
- Variables in Typescript
- Type Assertion
- Operators & their types
- Decision making constructs
- Loops
- Using Arrays
- Functions
- Writing & Using
- Classes Constructor Method
- Inheritance of Classes
- Type Assertion
- Abstract Class
- Working With Interfaces
- All About Generics

Introduction to Angular

- What is Angular?
- Angular 5 v/s 4 v/s 2 v/s AngularJS
- Angular CLI
- NodeJS Introduction(NPM)
- Setup of NodeJs and Angular
- What is Typescript?
- How does Angular get started?
- First Angular App

Components Overview

- Introduction to Components
- Creating components
- Role of AppModule & Component Declaration
- Registering Components
- Using Registered Components
- Creating Components with CLI
- Multiple components & passing data
- Nesting Components
- Working with Component templates
- Working with Component Styles
- Understanding Component Selector

Components & Databinding

- Introduction to Modules & Databinding
- Splitting Apps into Components
- Property & Event binding overview
- Binding to Custom Properties
- Assigning an Alias to Custom Properties
- Binding to Custom Events
- Assigning an Alias to Custom events
- Custom Property and Event Binding Summary
- Understanding View Encapsulation
- Using Local References in Templates
- Getting Access to the Template & DOM with @ViewChild
- Projecting Content into Components with ng-content
- Understanding the Component
- Lifecycle Hooks in Action

- Lifecycle Hooks & Template Access
- Accessing ng-content with @ContentChild

Directives

- Understanding Directives
- Using ngIf to Output Data Conditionally
- Enhancing ngIf with an Else Condition
- Output Lists with ngFor
- Styling Elements Dynamically with ngStyle
- Applying CSS Classes Dynamically with ngClass
- Creating Basic Attribute Directive
- Using the Renderer to build Better Attribute Directive
- More about Renderer
- Listen to Host Events using HostListener
- Bind to Host Properties using Host
- Binding Binding to Directive Properties
- Behind the scenes of Structural Directives
- What is ngSwitch?

Services & Dependency Injection

- Introduction to Dependency Injection
- Why do we need Services ?
- Creating a Logging Service
- Injecting the Logging Service into Components
- Creating a Data Service
- Understanding Hierarchical Injector
- How many Instances of Service?
- Injecting Services into Services

- Using Service for Cross-Component Communication

Transport Output using Pipes

- Introduction to Pipes
- Why are Pipes useful?
- Using Pipes
- Parameterizing Pipes
- Chaining Multiple Pipes
- Creating a Custom Pipe
- Parameterizing a Custom Pipe
- Creating a Filter Pipe
- Pure & Impure Pipes
- Understanding 'async' Pipes

Changing Pages with Routing

- What is Routing?
- Why do we need a Router?
- Setting up and Loading Routes
- Navigating with Router Links
- Understanding Navigation Paths
- Styling Active Router Links
- Navigating Programmatically
- Using Relative Paths in Programmatic Navigation
- Passing Parameters to Routes
- Fetching Route Parameters
- Fetching Route Parameters Reactively
- Route Observables
- Passing Query Paramters and Fragments
- Retrieving Query Parameters & Fragments

- Setting up Child(Nested) Routes
- Configuring the Handling of Query Parameters
- Redirection & Wildcard Routes
- Outsourcing the Route Configuration
- Introduction to Route Guards
- Protecting Routes with canActivate
- Controlling Navigation with canDeactivate
- Passing static data to a Route
- Resolving Dynamic Data with the resolve Guard
- Understanding Location strategies
- Understanding Observables

Template Driven Forms

- Introduction to handling forms
- Why do we need Angular's help?
- Template Driven(TD) v/s Reactive Approach
- Creating Template driven Forms & Registering Controls
- Submitting & Using the Form
- Understanding Form State
- Accessing the Form with @ViewChild
- Adding Validation to check User Input
- Built-in Validators & Using HTML5 Validation
- Using the Form State
- Outputting Validation Error Messages
- Set Default Values with ngModel Property Binding
- Using ngModel with Two-Way-Binding
- Grouping Form Controls

- Handling Radio Buttons
- Setting & Patching form values
- Using Template Driven Form Data
- Resetting Template Driven Forms

Reactive Forms

- Introduction to Reactive Approach
- Creating a Reactive Form in Code
- Syncing HTML and Form
- Submitting Reactive Forms
- Adding Validation to Reactive Forms
- Getting Access to Controls
- Grouping Controls
- Arrays of Form Controls
- Creating Custom Validators
- Using Error Codes with Reative Forms
- Creating Custom Async Validator
- Reacting to Status or Value Changes

Making HTTP Requests

- Introduction to Http Requests
- How HttpRequests Work in SPAs
- Sending Requests
- Adjusting Request Headers
- Sending GET Requests
- Sending a PUT Request

- Observable Operators
- Using Returned Data
- Catching Http Errors
- Using 'async' Pipe with Http Requests

Authentication & Route Protection

- Introduction to Authentication
- How Authentication works in SPAs
- Aout Jason Web Tokens
- Creating Signup Page & Route
- Setting up Firebase SDK
- Signing Users Up
- Signing Users In
- Requiring & Sending Token
- checking & Using Authentication Status
- Adding Logout Button
- Route Protection & Redirection Example

Angular Modules & Optimizing Apps

- The idea behind Modules
- Understanding App Module
- Understanding Feature Modules
- Creating a Feature Module
- Registering Routes in Feature Modules
- Understanding Shared Modules

- Creating a Shared Module
- Creating the Auth Feature module
- Understanding Lazy Loading
- How Modules & Services Work Together
- Understanding the Core Module
- Creating a Basic Core Module
- Using Ahead-of-time compilation
- Using AoT Compilation with CLI

HttpClient

- Introduction to HttpClient
- Unlocking the HttpClient
- Request Configuration & Response
- Requesting Events
- Setting Headers
- Http Parameters
- Progress
- Interceptors
- Modifying Requests in Interceptors
- Multiple interceptors

Angular Animations

- Introduction to Angular
- Animations Animation Triggers & State
- Switching between States
- Transitions
- Advanced
- Transitions Transition Phases

- The "void" State
- Using Keyframes for Animations
- Grouping Transitions
- Using Animation Callbacks

Unit Testing in Angular Apps

- Introduction
- Why Unit Tests?
- Analyzying the CLI Testing Setup
- Running Tests with CLI
- Adding Components and Some Tests
- Testing Dependencies: Components & Services
- Simulating Async Tasks
- Isolated v/s Non-isolated Tests

Project

Students will create a dynamic web site using various Angular components like controllers, directives, filters, Angular forms and modules. This project will be completed under the careful guidance of an experienced faculty. Working on this project will give students all the clarity they require in order to develop Angular applications in the software world.