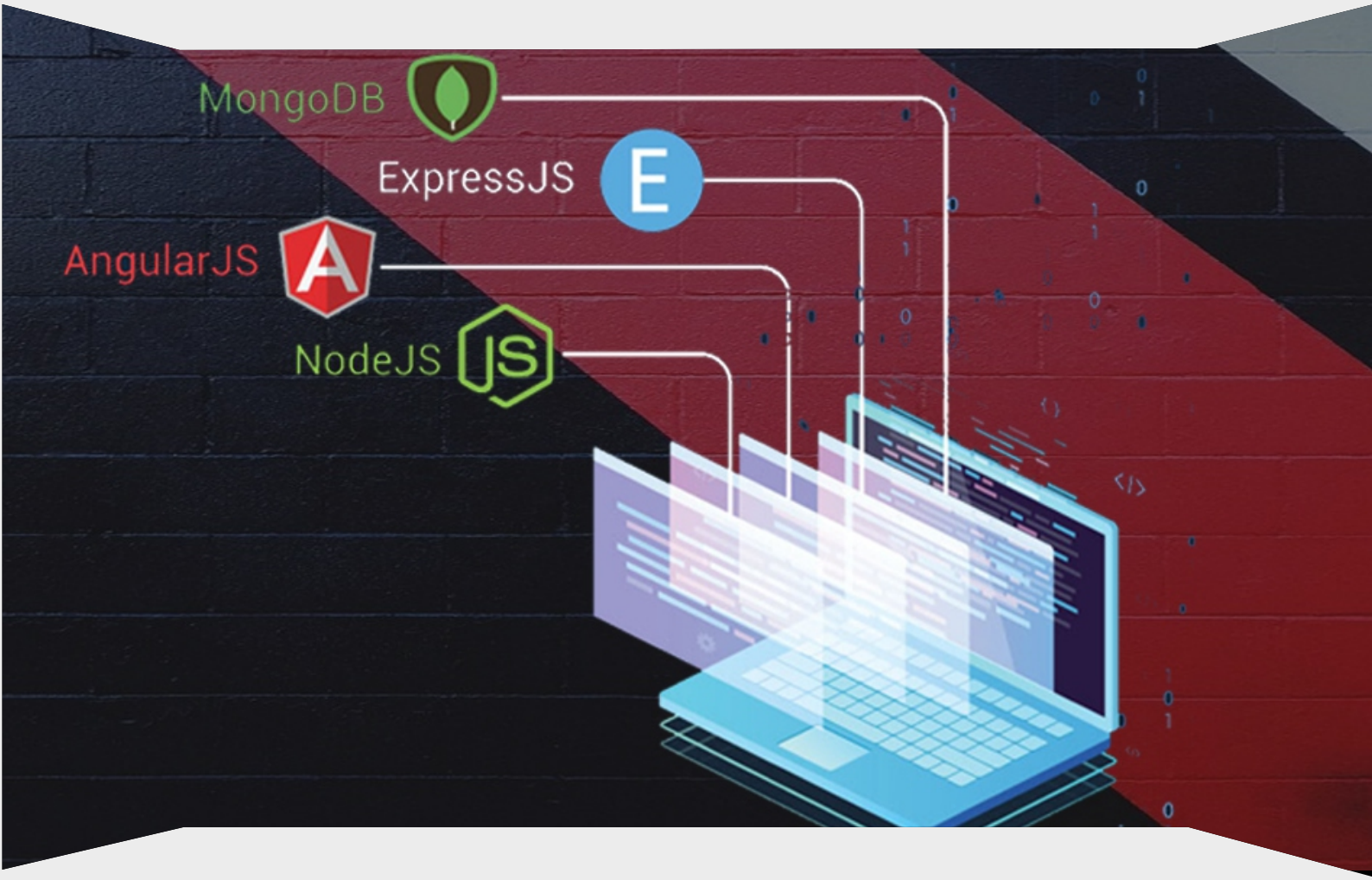
MEAN Stack



SevenMentor
PVT.LTD

MEANSTACK

MongoDB
Express JS
AngularJS
Node.js

Introduction and Environment Set Up.

What is Typescript and ECMAScript?, with Features,
AOT, JIT, TSC (Transpilation) Work and Config, code difference in JS and TS
What is Angular?
What is the framework?
The need for MVC, MVVM, MVW and MV* Architecture in Web Application
What is Components-Based Web Development? And the benefits.
AngularJs (vs) Angular
Setup for the local development environment
Angular files and folder structure with JSON configuration
Role of Node JS and NPM in Angular
What is CLI? Angular CLI
(Command Line Interface) Commands
Introduction Of Example Project

Start First Angular S. P. A. from basic

Execution of angular.json and package.json
Linking between all project files in MVC architecture
Creating our first element and typescript
Selector: customize or ignore tag(s)
Decorators and Metadata
Import and imports array,
Component and @Component, NgModule and @NgModule,
bootstrap, Selector and template,
Backticks and coding std in the template (ES2015 feature)
templateUrl, styles array, and the styleUrls array
Declaration array, class,
BrowserModule and bootsrap Module etc.
Launching the application.
Role of the Module and Components.
Splitting of Module and Component.
Exporting in Angular

Apart from this, we will learn how Angular reacts for
invalid structure, we will understand the concept of the framework
with MVVM, MV*, MVW or MVC architecture.

Directives

Structural directives
    Built-in Directives
        ngIF, ngFor, ngSwitch
    Style and Class Directives
        ngClass, [class.clsName]
        ngStyle, [style.stlName]
Attribute directives
    Customise Directive
Component: Way to Create, Split and reuse it.
Host Listener and Host Binding

SevenMentor
PVT.LTD

Data Binding

Interpolation
Property binding
Event binding
Two-way Binding
Class binding
Style binding
Methods

Components

create a dynamic component (without a separate component file) using @Component.
What are the components?
Understanding Components lifecycle hooks
Creating a component with CLI
Split an Angular application using components to make Angular application lightweight and high performance.

Modules

Root App module
Ahead-Of-Time(AOT) Compilation
Feature modules
Getting more Object Oriented:
Create a Model for data (validating data)
Classes – Properties, Methods, Constructors, Inheritance
Exporting a model
Mock data model (as the Angular team prefers)

View

Implementing style:
inline style,
internal style, and
external style file
Splitting view files
CSS style Scope

Forms

Forms in Angular
Template Driven Forms
Reactive

Pipes

Why pipes are useful?
Built-in pipes
Parameterizing pipes
Custom pipes

Services & Dependency injection

Creating Service
$http Service
Introduction to Injectors (Dependency Injection)
Providers: use and implementation.

Routing

parameterized routing.
Introduction
Configuring & Navigating
Parameterized routes

Operations Using Http Service

requests using HTTP service.
Creating Services
Creating Components
Creating Routings
Configuring NgModule
Working with JSON Data file
Run the application

Deployment of an optimize app product

Deploy on FTP web server
Deployment on Google firebase web hosting service
Build an application as a product with a specific location
Build an application as a product in an optimized way

API implementation in Angular Application.

What is API(s)? Use and Benefits of using API(s).
Way to configure and implement it.

Angular Material

What is Google's Material Design?
Use and benefits using Angular Material.
How to add and configure a new Module with an existing angular root module.
Way to convert and implement Materialize Designs in Angular Framework.

Implementation of Bootstrap Framework in Angular Framework with dependent JQuery library(es).

What are Bootstrap and ngb?
Way to implement for development and testing environment.

New CLI(s)

Understanding new and Deprecated CLI(s) in Angular.
Start with the development build

SevenMentor
PVT.LTD

Node.js:

Node.js is a development framework based on Google's V8 JavaScript engine.Node.js code is written in JavaScript and then compiled into machine code by V8 to be executed. Nice thing about Node.js is that it is all just JavaScript, so you can easily take functionality from a client-side script and place it in a server-side script. Following are the reason why Node.js is a great framework to start from:

I. JavaScript end-to-end: One of the biggest advantages to Node.js is that it allows to write both server-side and client-side scripts in JavaScript.

II. Event-driven scalability: Node.js applies a different logic to handling web requests. Rather than having multiple threads waiting to process web requests, they are processed on the same thread using a basic event model.

III. Extensibility: Node.js has a great following and an active development community.

IV. Time: Nodejs is super easy to set up and develop in. In only a few minutes, you can install Node.js and have a working web server.

Introduction
Environment Setup
First Application
REPL Terminal
Package Manager (NPM)
Callbacks Concept
Event Loop
Event Emitter
Buffers
Streams
File System
Global Objects
Utility Modules
Web Module
Express Framework
RESTFul API

SevenMentor
PVT.LTD

MongoDB:

MongoDB is an agile and scalable NOSQL Database. The name Mongo DB comes from “humongous”. It is based on the NoSQL document store model, meaning that data is stored in the database as a form of JSON objects rather than the traditional columns and rows of a relational database. Following are some reasons that MongoDB really fits in the Node.js stack well:

I. High performance: MongoDB is one of the highest performing databases available. Especially today when more and more people interact with websites, it is important to have a backend that can support heavy traffic.

II. High Availability: MongoDB’s replication model makes it easy to maintain scalability while keeping high performance.

III. High Scalability: MongoDB’s structure makes it easy to scale horizontally by sharing the data across multiple servers.

IV. NO SQL Injection: MongoDB is susceptible to SQL injection because objects are stored as objects, not using SQL strings.

Overview
Advantages
Environment
Data Modeling
Create Database
Drop Database
Create Collection
Drop Collection
Data Types
Insert Document
Query Document
Update Document
Delete Document

Express:

The Express Module acts as the webserver in the Node.js-to-Angular stack. The fact that it is running in Node.js makes it easy to configure, implement, and control. The Express module is an extension of Node.js for handling several web requests. This allows you to implement a running web server in Node.js with only a few lines of code. Features of Express are:

I. Route management: Express makes it easy to define routes that tie directly to Node.js script functionality on the server.
II. Error Handling: Express provides built-in error handling for documents not found and other errors.
III. Easy integration: An Express server can easily be implemented behind an existing reverse proxy system such as Nginx or Varnish. This allows it to be easily integrated into your existing secured system.
IV. Cookies: Express provides easy cookie management.
V. Session and cache management: Express also enables session management and cache management.

Overview
Environment
Hello World
Routing
HTTP Methods
URL Building
Middleware
Templating
Static Files
Form Data
Database

Project:
CRUD operation with MEAN from scratch.