

**START YOUR  
FULL STACK  
DEVELOPER  
CAREER TODAY!!**

**CREDO SYSTEMZ**

**AI Powered MEAN Stack  
Program**

**India's 1st AI-Driven IT Training – Credo Systemz**

# Capstone Projects :

Real Time MEAN Stack Training Projects. Build the frontend and backend, add smart features using AI, and solve practical problems with your solution.



## Food Delivery App

Building an application to browse restaurants, order food, and track deliveries in real-time.



## Twitter/Instagram Clone

Developing a social media platform to post content, follow others, with likes and comments.



## Chat Application

Building a real-time chat app in which users can message each other in one-on-one or group chats.



## Event Booking App

You can see events and book tickets online. It sends email confirmation after booking.



## Learning Management System (LMS)

People can search and apply for jobs. Companies can post job openings.



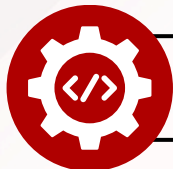
## Learning App

Users can order food online from restaurants. It shows live order updates and payment options.



# MEAN Stack

## Opportunities & Demand



**MEAN Stack Developer**



**Senior MEAN Developer**



**MEAN Stack Tech Lead**



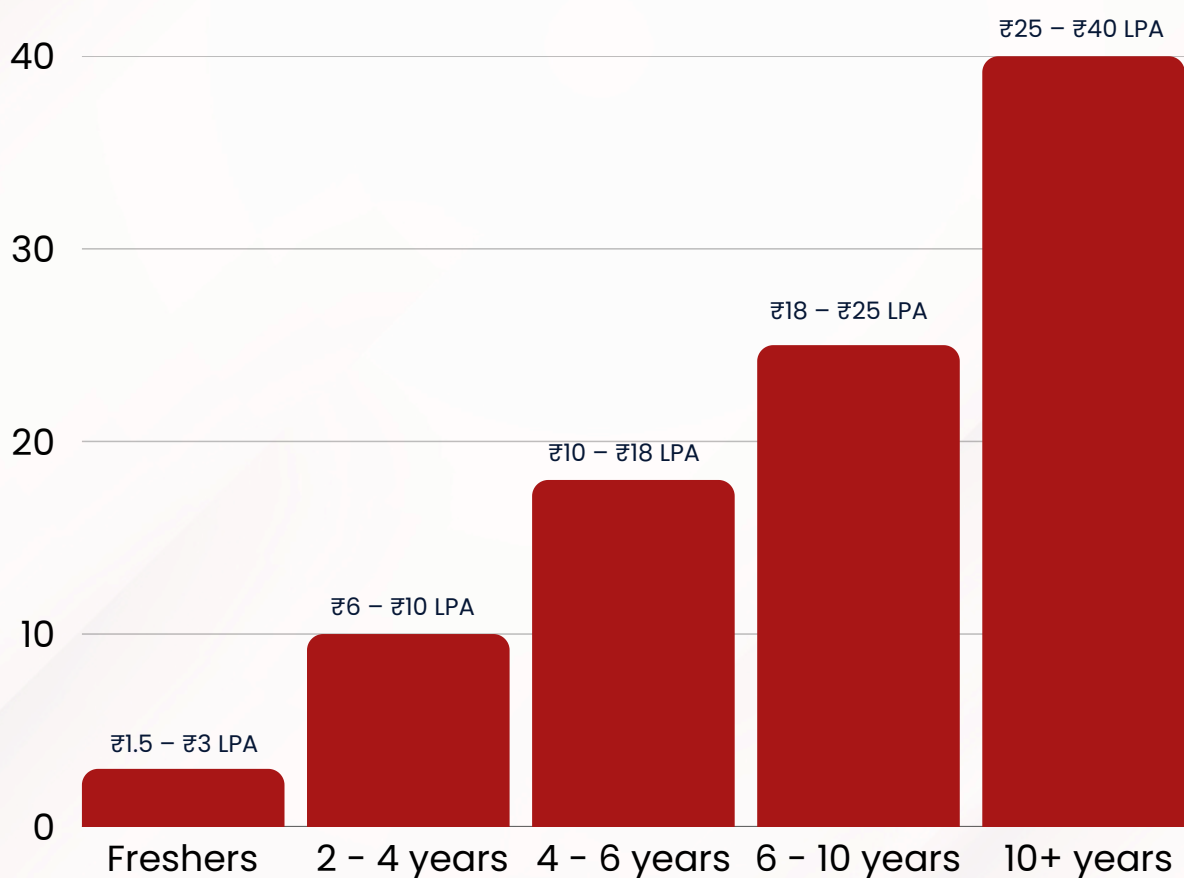
**MEAN Stack Architect**



**Junior MEAN Stack Developer**



**Full Stack Developer  
(MEAN Focused)**





# FULL STACK WITH AI COURSE SYLLABUS

ANGULAR + JAVA SPRING

Duration : 40 hrs

## Section 1 : Full Stack with AI

### Introduction

- What is Artificial Intelligence(AI)?
- Introduction to Artificial Intelligence
- Evolution of AI in the software industry.
- Key differences between AI, ML, and automation.
- How AI understands prompts and generates outputs in development.

### How Developers Use AI Tools Today

- Real-time code suggestions and auto-completion.
- Debugging and resolving errors.
- Auto-generating documentation and test cases.
- Enhancing productivity in frontend and backend development.
- Learning new technologies via AI-powered explanations.

### Overview of AI Tools (Copilot, ChatGPT, Codeium, Tabnine)

- GitHub Copilot – Code completion and generation inside VS Code.
- ChatGPT – Conversational AI for explaining concepts, solving bugs, and writing logic.
- Codeium – Lightweight open-source AI assistant for developers.
- Tabnine – Predictive code suggestions for multiple languages and editors.
- Amazon CodeWhisperer – AI coding tool optimized for cloud and AWS-based applications.
- Feature comparison and use cases.

### Installing GitHub Copilot in VS Code

- Prerequisites: GitHub account, VS Code setup.
- Installation steps: Search, install, and activate the Copilot extension.



- Verifying Copilot functionality within the editor.

## **Activating Copilot with GitHub Account**

- Connecting VS Code with GitHub credentials.
- Subscription setup (Free/Trial/Pro plans).
- Enabling Copilot across projects and environments.

## **Section 2 : HTML 5**

- Introduction - Web
- What are the different Types of Web Apps overview?
- Introduction to HTML
- Define HTML Structure
- Difference between Tag vs Element
- Difference between Semantic vs Non-semantic elements
- Difference between Block level elements vs Inline elements
- HTML Elements
- HTML Forms & its Attributes
- HTML Input Elements
- HTML Global Attributes
- HTML Element Specific Attributes

### **AI in Action:**

- Use ChatGPT to explain HTML tags, forms, and semantic elements.
- Generate HTML templates using GitHub Copilot or Codeium from simple prompts like "Create a contact form."
- Auto-suggest attributes and fix structure issues using AI linters or extensions in VS Code.

## **Section 3 : CSS & Bootstrap**

- What is CSS?
- Understanding the CSS Syntax.
- CSS Selectors
- How To Add CSS in HTML
- CSS Colors & Backgrounds
- CSS Borders, Margins, Padding, Height and Width
- Responsive Web Design – Media queries

- What is Bootstrap?
- Get Start with Bootstrap
- Containers
- Grid System
- Structure of a Bootstrap Grid
- Bootstrap Colors
- Bootstrap Tables
- Bootstrap Jumbotron
- Bootstrap Alerts, Buttons

#### **AI in Action:**

- Use Copilot to autocomplete style rules and responsive media queries.
- Generate custom CSS frameworks using AI prompt tools like Tabnine.
- Ask ChatGPT to refactor or explain complex CSS rules and layout bugs.
- Build Bootstrap grids or buttons by typing "Create 3-column responsive layout using Bootstrap."

## **Section 4: JavaScript**

- Introduction – Javascript
- JavaScript Events
- JavaScript Functions
- Inbuilt methods in JS
- Arrays in JavaScript
- Objects in JavaScript
- Conditional and loops in JavaScript
- HTML DOM Manipulation

#### **AI in Action:**

- Copilot can complete functions or suggest better loop structures.
- Use ChatGPT to explain callback vs promise vs async/await in your own words.
- Auto-generate ES6 classes, arrow functions,
- Destructuring examples using natural language.
- Debug JavaScript errors via ChatGpt Prompts
- Debug Logic mistakes via ChatGPT prompts.

## ● Section 5: TypeScript

- TypeScript Introduction
- TypeScript Environment Setups
- Variables in TS
- Datatypes in TS
- OOPS in TypeScript
- Features in TS

### **AI in Action:**

- Get type suggestions using Copilot, auto-generate interfaces/classes, and refactor TS code with AI tools.

## ● Section 6: Angular Introduction

- Angular- Definition
- Difference between Framework & Library?
- History of Angular and its versions.
- Why Angular?
- What are the Features of Angular
- Define Single Page Application
- What is the Difference between SPA & Traditional Application?
- Define MVC
- How MVC works in Client & Server sides?

## ● Section 7: Angular Environment setups

- What is Angular CLI?
- What is the Purpose of the CLI?
- Angular CLI installation.
- CLI vs Without CLI Overview.
- Create an Angular App by using CLI.
- Compiling the Angular App & Open it in a browser.
- Angular app Bootstrapping process
- About Angular libraries
- Brief explanation about the structure of the Angular App.

## ● Section 8: Main Building Blocks of Angular

- An Overview of the below Main Building blocks of Angular



- Modules
- Components
- Decorator
- Metadata
- Templates
- Data binding – Directives
- Services
- Dependency Injection.

## ● **Section 9: Angular Modules**

- Angular Module Overview.
- Define the Importance of the Module?
- Why Modules?
- Root Module, Core Module, Feature Module and Shared Module – Overview.
- How to create Angular Modules?
- @NgModule Decorator & its Meta data properties – Overview.

## ● **Section 10: Components**

- Angular Component – Overview.
- @Component decorator & its Meta data properties.
- Component's Structure overview.
- What are the ways to render a Component in the view?
- Component Lifecycle Hooks.
- Nested or Parent & Child Component – Overview

## ● **Section 11: Data Binding, Property Binding, Event Binding & 2-way Data Binding**

- Data Binding Introduction
- Define String Interpolation.
- Property Binding – Overview.
- what is Custom Property Binding?
- Overview of Event Binding
- String Interpolation VS Property Binding
- Two-way Data Binding
- Implementing the 2-way Data Binding

## ● Section 12: Services

- Introduction – Service.
- Importance of Service.
- How to create Services in Angular?
- What are the ways to Provide Services in Angular?
- Dependency Injection – Overview.
- How to use Dependency Injection?
- What is @Injectable()?

## ● Section 13: Directives

- Directives – Introduction.
- Component VS Directives
- What are the Different kind of Directives available in Angular?
- Difference between Structural & Attribute Directives.
- Overview of All Structural & Attribute Directives.
- @Input decorator and its methods.
- What is ElementRef and its purpose?

## ● Section 14: Components Communication

- Overview of Components Interaction.
- Component Interaction from Parent to Child.
- Component Interaction from Child to Parent.
- @ViewChild decorator overview
- @Input & @Output decorator overview

## ● Section 15: Reactive Form in Angular

- What is Reactive Form?
- Difference between Template Drive & Reactive Form
- What is form group & form control?
- How to sync view & Reactive form TS?
- What is Patch Values & Set Values?
- How to get Reactive from Values?

## ● **Section 16: Service and Dependency Injection in Angular**

- Service in Angular
- Create & configure Service in Angular.
- How to do Dependency Injection in Angular?
- Define Singleton Object

## ● **Section 17: Routing in Angular**

- What is Routing?
- How Routing makes our App into SPA?
- How to configure Routing in an Application?
- Load our components dynamically based on url path.
- How to create Child Route?
- Navigating to other links programmatically.
- Passing Parameter to the Routes.
- Client-side authorization using Route Guard

## ● **Section 18: HTTP & Observable in Angular**

- HTTP Client in Angular.
- REST API – Overview
- How to establish HTTP request to Server side.
- How HTTP Mechanism works?
- What are the methods available in HTTP?
- Define Observable & Observer
- What are the call back methods available in Observable?
- Creation of a Custom Observable
- Define next(), error() and complete()
- How to send Query Params & Custom Headers?
- How to connect any backend & APIs?

## ● **Section 19: Authorization in Angular & JWT**

- Client side Authorization vs Server side Authorization.
- Server side Authorization by using JWT Token.
- Set JWT Key Expiry time.
- HTTP interceptors – Overview.
- How to configure HTTP interceptors?



## ● Section 20: Real-time Project in Angular

- During the course we will take one Real-time E-commerce application and apply all the above sections into the project. In the project Front-end will be in Angular and Back-end will be in Node JS. On top of the Node JS we will write Express JS as a REST Api. For Database, we choose MongoDB for CRUD Operations.

## ● Section 21: Angular – With AI Integration

### Use Copilot to

- Create components, services, and modules quickly
- Write routing setup and navigation logic
- Build forms with validation rules
- Add Dependency Injection in services
- Auto-complete commonly used Angular code

### ChatGPT:

- Explore Angular topics like data binding and component communication
- Solve errors in templates or logic
- Lifecycle hooks in easy terms
- Generate examples for forms, routing, and API calls
- Answer “how-to” questions while learning

### Use Codeium or Tabnine:

- Suggest HTML and binding code in templates
- Auto-fill \*ngIf, \*ngFor, and form inputs
- Style/class bindings
- Generate reusable code blocks
- Speed up writing component and service logic

## ● Section 22: Getting Started with Node Js

- Welcome Preview
- What is Node Preview
- Node Architecture Preview

- How Node Works Preview
- Installing Node Preview
- Your First Node Program

## ● **Section 23: Understanding Node Module System**

- Introduction
- Creating a Module
- Loading a Module
- Modules and modularity
- Global Object
- OS Module
- Path Module
- File System Module
- Events Module
- HTTP Module

## ● **Section 24: Node Package Manager**

- Introduction to NPM
- Installing a Node Package
- NPM Packages and Source Control
- Semantic Versioning
- Registry Info for a Package
- How to install specific version of the package
- Introduction to NPM
- Installing a Node Package
- NPM Packages and Source Control
- Semantic Versioning
- Registry Info for a Package
- How to install specific version of the package
- How to update Local Packages
- Uninstall the packages
- Dev Dependencies
- Publishing Package

## ● **Section 25: Asynchronous Programming**

- What is Asynchronous Programming in JavaScript?

- Synchronous vs Asynchronous
- JavaScript EventLoop
- Callbacks
- Callback Hell
- Promises in JavaScript
- Promise Chaining
- Async Await

## ● **Section 26: Rest API & HTTP**

- What is HTTP
- Understand how HTTP works?
- List of HTTP response codes
- HTTP Module in Node
- HTTP Methods – GET, POST, PUT & DELETE
- Processing Form Data
- Sending response back to Server

## ● **Section 27: Understanding Express JS**

- What is Express JS?
- Installing Express JS
- Creating HTTP Server
- Nodemon
- Environment Variables
- Routing
- Route Parameters
- Handling Multiple Routes
- Input Validation
- Handling HTTP Method – Get, Post, Put & Delete using Express

## ● **Section 28: Node.js, Express – With AI Integration**

### **Use Copilot:**

- Scaffold Express routers and middleware using short comments.
- Write API routes with GET/POST/PUT/DELETE handlers instantly.



## ChatGPT:

- Help debug routing issues or token verification.
- Explain REST patterns, middleware chaining, or error handling.
- Use AI tools to generate boilerplate code for services, validation.

## ● Section 28: MongoDB & Mongoose

- What is MongoDB
- MongoDB Advantages
- Installing MongoDB
- Mongoose ODM
- Schemas
- Models
- Documents
- Saving a Document
- Querying Documents
- Logical & Comparison Query Operators
- Regular Expressions
- Counting
- Pagination
- CRUD Operations in MongoDB using Mongoose and Express

## ● Section 29: Data Validation in MongoDB

- Understanding Validation in MongoDB
- Built-in Validators
- Custom Validators
- Validation Errors

## ● Section 30: Authentication and Authorization

- Introduction
- User Model
- User Registration
- Hashing Passwords
- Authenticating Users
- JSON Web Token

- Generating Authentication Tokens
- Setting Response Headers
- Auth Protect Middleware
- Protecting Routes
- Role-based Authorization

## ● **Section 31: Handling Errors**

- Introduction
- Express Async Errors
- Logging Errors
- Preparing the App for Production
- Adding the Code to a Git Repository
- Setting Environment Variables

## ● **Section 32: Deployment**

- Preparing the App for Production
- Adding the Code to a Git Repository
- Setting Environment Variables

## ● **Section 33: Real Time Projects**

- 5+ Real-time projects (Full stack coverage)
- Setup Git for local repository
- Create a GitHub account
- Sync the code base with GitHub repo
- Firebase Hosting Overview
- Setting up Firebase in local
- Host your project in Firebase

## ● **Section 34: MongoDB – With AI Integration**

### **AI in Action:**

- Generate Mongoose models using AI prompts like:
- Create a Mongoose schema for a blog with title, content, author, and timestamps
- Use ChatGPT to explain population, virtuals, and query chaining.

- Auto-create Mongo queries, pagination logic, and custom validations with Copilot.

## ● **Section 35: Placements**

- Build your Professional Resume
- Update LinkedIn Profile
- Interview based Training – GD, Tech round, HR panel
- Minimum 5 Mock interviews before Real interview
- 100% Placement Guaranteed
- Join your Dream Job



# SKILLS AND TOOLS

## Skills Covered

HTML



CSS



Mongo DB



Node JS



Typescript



Bootstrap



API



Ajax



Javascript



Database



Angular



Typescript



## Tools Covered

VS Code



Postman



Git



Node Package  
Manager



PM2



Bcrypt.js



Helmet.js



MongoDB Compass





# PLACEMENT SUCCESS STORIES

Designation ..... Company ..... Package .....



Rajiv

**Full Stack Developer**



**7.8 LPA**



YogaLakshmi

**Front End Developer**



**8.5 LPA**



Madhumathi

**Software Developer**



**7.8 LPA**



Suja

**Cloud Engineer**



**8.6 LPA**



Sanakarapandiyan

**Full Stack Developer**



**8.0 LPA**



Asai Thambi

**Full Stack Developer**



**11.0 LPA**



# OUR HIRING PARTNERS



# Earn your Full Stack Course Completion Certificate

Credo Systemz's certificate is highly recognized by  
30K Global companies around the world.



# WHAT OUR TRAINEE SAYS?



**Libin Charan**

**4.7** ★★★★★

Credo Systemz's Full Stack course is very practical and job-oriented. The trainers explained every topic clearly with real-time examples. Placement support was strong. I got placed after training. Totally worth the investment!



**Mukesh Babu**

**4.2** ★★★★★

The Full Stack training at Credo Systemz covered front-end and back-end thoroughly. Live projects helped a lot. Mentors were always helpful. Thanks to their placement team, I'm now working in an IT company.



**Padmesh**

**5.0** ★★★★★

I joined Credo Systemz for Full Stack training and it was the best decision. Great trainer, hands-on projects, and mock interviews helped me switch to IT. Strongly recommend for career changers!



**Vasmitha**

**4.9** ★★★★★

Credo Systemz Full Stack program gave me confidence to attend interviews. I learned MERN stack with real-time examples. The support from admin and placement team was great. Got placed right after course completion.



**Jason Isreal**

**4.0** ★★★★★

As a fresher, Credo Systemz's Full Stack course gave me all the knowledge I needed. The trainer's guidance and regular assessments helped me grow. I got placed within weeks of completing the course!



**Subritha**

**4.5** ★★★★★

I had no coding background but the Full Stack course at Credo Systemz was easy to follow. Trainers were supportive and explained clearly. Placement team constantly followed up until I got placed. Thanks!






# CHENNAI


## VELACHERY

New # 30, Old # 16A, Third Main Road, Rajalakshmi Nagar, Velachery, (Opp. to Murugan Kalyana Mandapam), Chennai – 600 042.

 +91 98844 12301

## OMR

Plot No.8, Vinayaga Avenue, Rajiv Gandhi Salai, (OMR), Okkiampettai, (Behind Okkiyampet Bus Stop) Chennai – 600 097.

 +91 96001 12302

# OVERSEAS

## USA

Houchin Drive, Franklin, TN -37064. Tennessee

## UAE

Sima Electronic Building, LLH Opposite, Electra Street – Abu Dhabi

**India's 1st AI-Driven IT Training**  
**Credo Systemz**