

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

CREDO SYSTEMZ

**AI Powered ASP .NET Angular
Program**

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

Capstone Projects :

Real Time Full Stack Developer 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.



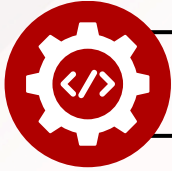
Learning App

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



ASP.NET

Opportunities & Demand



Full Stack .Net Developer



Microsoft Full Stack Engineer



.NET Software Engineer



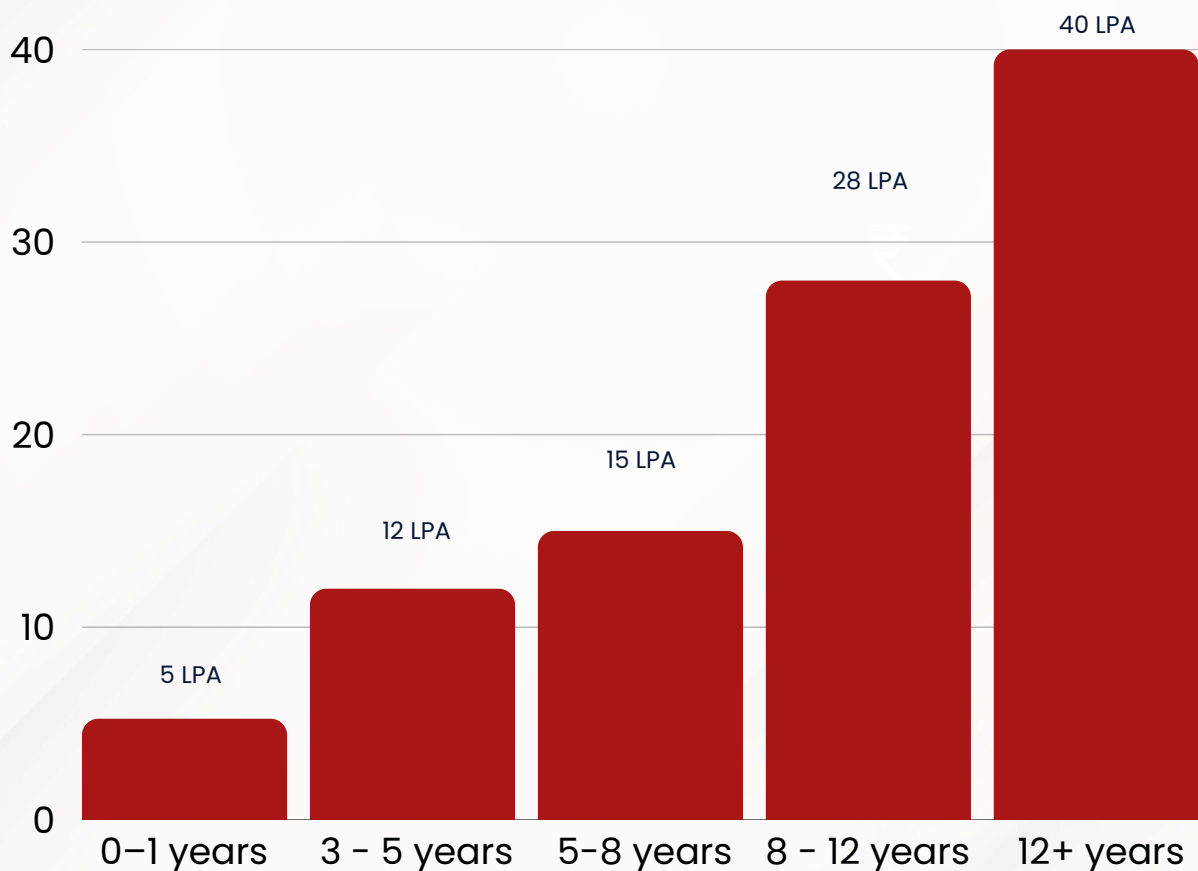
.NET Architect



Software Engineer



.Net Core Full Stack Developer



FULL STACK WITH AI COURSE SYLLABUS

ANGULAR + ASP.NET

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: 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 21: C# Fundamentals

- Introduction to .NET in C#
- Variables and Data Types
- Operators and Expressions
- Control Structures (if, switch)
- Loops (for, while, foreach, do-while)
- Error Handling with Try-Catch
- Methods and Parameters
- Namespaces and Assemblies

● Section 22: Object-Oriented Programming (OOP)

- Classes and Objects

- Constructors and Destructors
- Encapsulation, Abstraction
- Inheritance and Polymorphism
- Interfaces vs Abstract Classes
- Access Modifiers (public, private, etc.)
- Static vs Instance Members

● **Section 23: C# Advanced Topics**

- Collections (List, Dictionary, Queue, Stack)
- Generics
- Lambda Expressions
- LINQ (Language Integrated Query)
- Exception Handling (Advanced)
- File I/O (Reading/Writing files)
- Nullable Types and Null Safety

● **Section 24: Working with APIs in C#**

- Understanding RESTful APIs (GET, POST, PUT, DELETE)
- Using HttpClient for making API calls
- Sending and Receiving JSON
- Deserialization with System.Text.Json or Newtonsoft.Json
- Handling API Authentication (Bearer Tokens, API Keys)
- Calling External APIs and Consuming Responses
- Creating Unit Tests for API Calls

● **Section 25: Building APIs with ASP. Net Core**

- Introduction to ASP. Net Core
- Project Structure (Controllers, Models, Program.cs, etc.)
- Creating Controllers and Endpoints
- Routing and Attributes
- Model Binding and Validation
- Dependency Injection (DI)
- Middleware and Request Pipeline
- Swagger / OpenAPI Integration
- Error Handling and Logging in APIs

● Section 26: .NET with AI Integration

Use Copilot

- Quickly create controllers, models, and services
- Write API methods (GET, POST, etc.) faster
- Set up Dependency Injection easily
- Build CRUD operations with Entity Framework
- Add error handling and async code

ChatGPT:

- Explain C# and ASP.NET Core topics in simple terms
- Help fix code errors or bugs
- Give examples for APIs, LINQ, and database queries
- Guide you in setting up JWT authentication
- Answer “how-to” questions as you learn

Use Codeium or Tabnine:

- Suggest code for controller methods and classes
- Auto-complete LINQ queries and database logic
- Write test methods and mock data
- Fill in config files like appsettings.json
- Speed up writing SQL queries and helper functions

● Section 27: Working with Databases and SQL

- Introduction to Relational Databases and SQL Basics
- CRUD Operations with SQL (SELECT, INSERT, UPDATE, DELETE)
- Joins and Query Optimization
- Entity Framework Core (EF Core)
- Code-First and Database-First Approaches
- DbContext and DbSet
- LINQ to SQL
- Migrations
- Connecting C# to SQL Server
- Handling Relationships (One-to-Many, Many-to-Many)

● Section 28: Advanced

- Asynchronous Programming with async / await
- Working with Azure/AWS for Deployment
- JWT Authentication in APIs
- Role-Based Authorization
- Versioning in APIs

AI in Action:

- Convert Prompts to SQL – using ChatGPT, Azure OpenAI
- Write Queries Faster – with GitHub Copilot, Codeium
- Improve Query Performance – get tips from ChatGPT
- Clean Data with SQL – use ChatGPT, AI tools
- Fix SQL Errors – using ChatGPT, Copilot
- Learn SQL Smarter – with AI-powered tutors like ChatGPT
- Create Reports Easily – using Power BI Copilot, Excel + AI
- Explain SQL Code – with help from ChatGPT

● SECTION 29: 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

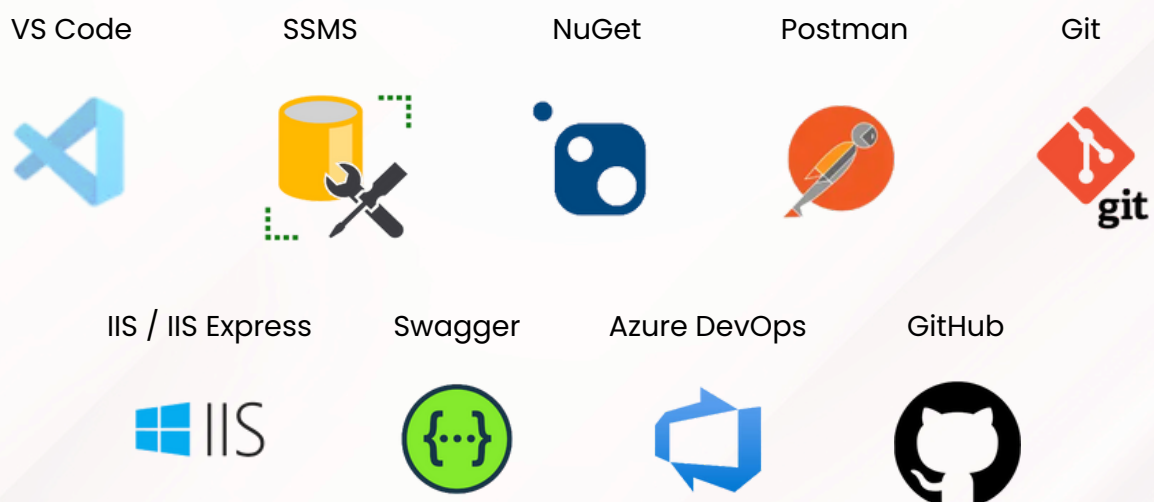


SKILLS AND TOOLS

Skills Covered



Tools Covered



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