

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

CREDO SYSTEMZ

**AI Powered ASP.NET React
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.



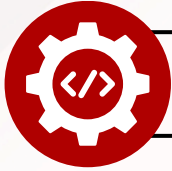
Learning App

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



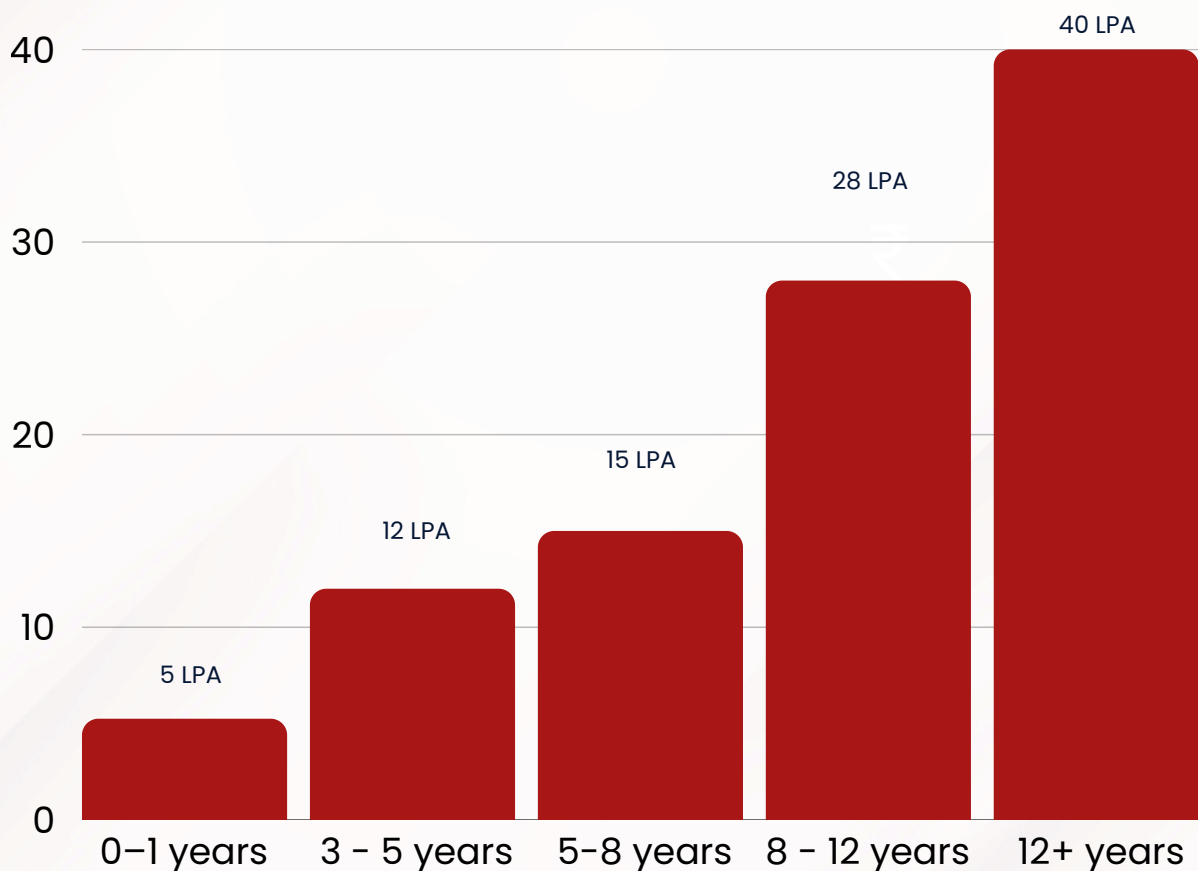
.NET Architect



Software Engineer



.Net Core Full Stack Developer



FULL STACK WITH AI COURSE SYLLABUS

REACT JS + JAVA SPRING

Duration : **100 hrs**

● Phase 1 – AI Introduction

● 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.

● **Phase 2 – Frontend Development**

● **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

Section 5: ES6 JavaScript

- History of JavaScript
- Features
- let & const and its example
- Arrow Functions
- Alternative
- Tips and Arrow Functions

- Exports and Imports
- Tips for exports and imports
- Classes
- Classes example
- Inheritance
- Spread and rest Parameter
- Destructing

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 6: Getting Started with React

- What is ReactJS?
- Installation or Setup
- Create React App
- Advantages Of React JS
- Workflow Of React JS
- Node Setup
- How to use NPM and its purpose
- ES6 Introduction
- JS vs JSX vs TS vs TSX

Section 7: Components

- Creating Components
- Basic Component
- Nesting Components
- Higher order components

Section 8: OVERVIEW OF JSX

- Introduction of Virtual DOM.

- Introduction of Virtual DOM.
- Difference between JS and JSX.
- Why use JSX?
- JSX Attributes
- JSX Styling

● **Section 9: Props in React**

- Introduction
- Default props
- PropTypes
- Passing down props using spread operator
- Props.children and component composition
- Detecting the type of Children components
- Props Drilling

● **Section 10: State in React**

- Basic State
- Common Antipattern
- setState()
- State, Events And Managed Controls

● **Section 11: React Component Lifecycle**

- Component componentWillMount
- Component UnMount
- Component Update
- Lifecycle method call in different states
- React Component Container

● **Section 12: Handling Events**

- Event handling in React
- Binding event handlers
- Arrow functions vs. regular functions
- Primary Events
- Sharing the events between components

● **Section 13: Communicate Between Components**

- Child to Parent Components
- Parent to Child Components
- Not-related Components

● **Section 14: Conditional Rendering**

- Conditional rendering with if statements
- Conditional Rendering using logical && in JSX
- Conditional Rendering using ternary operator
- Preventing Component from Rendering

● **Section 15: Lists and Keys**

- Steps to Create and Traverse React JS Lists
- Rendering lists inside Components
- Key in React List

● **Section 16: React JS Keys**

- What is a key in React ?
- Assigning keys to the list
- Difference between keys and props in React

● **Section 17: React Hooks**

- useState()
- useEffect()
- useContext()
- useRef()
- useReducer()
- Custom React Hooks
- useDispatch(), useSelector, useMemo, useLayoutEffect()

● **Section 18: React Hooks**

- useEffect()

- useEffect()
- useContext()
- useRef()
- useReducer()
- Custom React Hooks
- useDispatch(), useSelector, useMemo, useLayoutEffect()

● **Section 19: Styling in React**

- CSS in React
- Different approaches for styling (CSS, CSS-in-JS, CSS Modules)
- Inline styles
- Styling Libraries
- Popular CSS frameworks (e.g., Bootstrap, Material-UI)

● **Section 20: Router**

- Introduction to React Router
- Setting up React Router
- Creating routes
- Navigating with React Router
- Using Link and NavLink
- Nested Routes and Dynamic Routing
- Nested routes
- Passing parameters to routes
- Query String, accessing current URL

● **Section 21: React JS forms**

- React Forms
- Lists of Form components
- Control Input elements.
- Controlled and Uncontrolled form components
- Adding Forms in React
- Handling React Forms
- Submitting React Forms
- Multiple Input Fields
- React JS Form validations
- Custom Validations

● Section 22: State Management with Redux

- Introduction to Redux
- Understanding the need for state management
- Basic concepts: actions, reducers, store
- Setting Up Redux
- Installing Redux and setting up a store
- Creating actions and reducers
- Connecting React with Redux
- Using connect to connect components to the store
- Dispatching actions

● Section 23: Asynchronous Programming and API Integration

- AJAX and Fetch API
- Making HTTP requests in React
- Fetching data from an API
- Async/Await and Promises

● Section 24: Handling errors in React application

- Error Handling and Debugging
- Debugging React apps
- Performance Optimization
- Memoization
- React.memo and PureComponent

● Section 25: React JS Virtual DOM

- What is DOM ?
- Disadvantages of real DOM
- Virtual DOM
- How does virtual DOM actually make things faster?
- How virtual DOM Helps React?
- Virtual DOM Key Concepts
- Differences between Virtual DOM and Real DOM

- **Section 26: Unit Testing Overview**
- **Section 27: Lazy Loading**
- **Section 28: Code Splitting**
- **Section 29: Server-Side Rendering**
- **Section 30: Micro FrontEnd Overview**
- **Section 31: Deploying a React application**
- **Section 32: React JS – With AI Integration**

Use Copilot

- Scaffold React components with useState, useEffect, or useReducer
- Write router configs, validation logic, and Redux action templates

ChatGPT:

- Explain prop drilling or React context
- Help fix state update bugs
- Create optimized component hierarchies
- Generate form validation rules, lifecycles, hooks using AI prompts.
- Use Codeium or Tabnine to auto-suggest JSX templates and inline styles.

● **Phase 3 – Backend Development**

● **Section 33: 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 34: 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 35: 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 36: 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 37: 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 38: .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

● **Phase 4 – Database Development**

● **Section 39: 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 40: 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 41: 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