.NET Full Stack Developer SYLLABUS

Duration 100+ hrs

Project Covered 3-Live Project

1) Frontend (HTML + CSS + JS + Bootstrap + React JS)

2) Database (MS-SQL)

3) C#, .NET CORE MVC

4) WEB API

5) INTEGRATION REACT JS

HTML, CSS, JavaScript, and Databases Course Content

- **Topics**:
  1. **HTML**:
     - Basic Structure: Tags, Elements, Attributes
     - Common Tags: , <input>, <form>, <table>
     - All HTML TAGS
     - Container Tags
  2. **CSS**:
     - Styling Basics: Selectors, Properties, Values
     - Layout Concepts: Box Model, Flexbox
     - Bootstrap, Tailwind
  3. **JavaScript**:
     - Syntax: Variables, Functions, Events
     - DOM Manipulation: Accessing and Modifying Elements
     - ES6 Features
  4. **MS-SQL Databases**:
     - Relational Database Concepts: Tables, Rows, Columns
     - Basic SQL: SELECT, INSERT, UPDATE, DELETE
     - Salesforce Context: Objects as Tables, Fields as Columns

REACT Content

1. **Components & JSX**
   - The building blocks of React. You'll learn how to create reusable UI pieces using functional or class components and write HTML-like JSX syntax.
2. **State & Props**
   - State manages data that changes (e.g., user input), while props pass data between components.
3. **Hooks (e.g., useState, useEffect)**
   - Hooks let you "hook into" React features like state and lifecycle methods in functional components. useState handles state, useEffect manages side effects (e.g., fetching data).
4. **Event Handling**
   - How to respond to user actions like clicks or form submissions.
5. **Routing (React Router)**
   - Adding navigation to create multi-page-like experiences in a single-page app.
6. **State Management (e.g., Context API, Redux)**
   - Tools to manage app-wide data (beyond local state). Context API is built-in; Redux is popular for complex apps.
7. **Fetching Data & APIs**
   - Connecting React to external APIs (e.g., using fetch or axios) to display real-time data.

C# Course Content

Introduction of C#

- C# Features, How to create Envionment of C#, how to create first program of C# using visual studio

Variables, Data Types & Type Safety

- Primitive types (int, double, bool), strings, and complex types (arrays, lists). C# is strongly typed, so understanding type safety and conversions (e.g., casting) is crucial.

Control Structures

- Loops (for, while), conditionals (if, switch),

Methods & Parameters

- Writing reusable functions, passing arguments (by value vs. reference), and using optional or named parameters..

Object-Oriented Programming (OOP) Concepts

- **Classes & Objects**: The backbone of C#. You define blueprints (classes) and create instances (objects).
- **Inheritance**: Reuse and extend code by deriving classes from others.
- **Polymorphism**: Use methods in different ways (e.g., method overriding).
- **Encapsulation**: Hide data with access modifiers (public, private, etc.).

Async/Await & Multithreading

- Writing asynchronous code to handle tasks like I/O operations without blocking the app.

Collections (List, Dictionary, etc.)

- Managing groups of data with dynamic structures like List<T> or key-value pairs in Dictionary<TKey, TValue>.
- *Type of Collection, Generic and Non Generic Collection*

Exception Handling & File Handling

- Using try-catch-finally to gracefully handle errors and keep your app running.

Delegate & Events

- What is delegate , type of delegate
- *What is Event*

Reflection and Extension Class

- What is reflection , how to use it?
- *What is Extension class and Extension Method*

ASP.NET CORE MVC CONTENT

Module 1: Introduction to ASP.NET Core

- **Topics**:
  1. What is ASP.NET Core? (Cross-platform, open-source, vs. ASP.NET Framework)
  2. Setting Up the Environment (Visual Studio, VS Code, .NET CLI)
  3. Creating Your First ASP.NET Core App (Project structure: Program.cs, appsettings.json)
  4. Running and Debugging Basics
- **Hands-On**: Build a "Hello World" web app.

Module 2: Mastering the Middleware Pipeline

- **Topics**:
  1. What is the Middleware Pipeline? (Request delegate concept)
  2. Built-In Middleware (e.g., UseRouting, UseStaticFiles, UseAuthentication)
  3. Configuring Middleware in Program.cs
  4. Middleware Ordering and Short-Circuiting
  5. Writing Custom Middleware (e.g., logging, request modification)
- **Hands-On**: Add custom middleware to log requests and handle errors.

Module 3: MVC Architecture in ASP.NET Core

- **Topics**:
  1. MVC Overview (Model, View, Controller roles)
  2. Controllers and Actions (Routing, HTTP methods: GET/POST)
  3. Views with Razor Syntax (Dynamic HTML, layouts, partials)
  4. Models and Data Binding (Strongly-typed models, validation)
  5. Routing Basics (Conventional vs. attribute routing)
- **Hands-On**: Create a simple CRUD app (e.g., a to-do list).

Module 4: Dependency Injection (DI)

- **Topics**:
  1. What is Dependency Injection? (IoC principle)
  2. DI in ASP.NET Core (Service lifetimes: Transient, Scoped, Singleton)
  3. Registering and Injecting Services
  4. Using DI in Controllers and Middleware
- **Hands-On**: Refactor the CRUD app to use DI for a service layer.

Module 5: Working with Data (Entity Framework Core)

- **Topics**:
  1. Introduction to EF Core (ORM basics)

2. Setting Up EF Core (DbContext, migrations)
3. CRUD Operations with EF Core
4. Relationships (One-to-Many, Many-to-Many)
5. Querying with LINQ

- **Hands-On**: Extend the CRUD app with a database (e.g., SQLite or SQL Server).

Module 6: Building RESTful APIs

- **Topics**:
  1. What are RESTful APIs? (HTTP methods, status codes)
  2. Creating API Controllers
  3. Serialization and JSON Responses
  4. API Versioning and Swagger/OpenAPI Integration
  5. Securing APIs (Intro to authentication)
- **Hands-On**: Build a REST API for the CRUD app (e.g., /api/todos).

Module 7: Authentication and Authorization

- **Topics**:
  1. Authentication Basics (Cookies, JWT)
  2. Setting Up Identity in ASP.NET Core
  3. Authorization (Roles, policies)
  4. Protecting Routes and APIs
- **Hands-On**: Add login/logout to the CRUD app with Identity.

Module 8: Advanced Features and Performance

- **Topics**:
  1. Razor Pages (Alternative to MVC)
  2. SignalR for Real-Time Features (e.g., chat)
  3. Caching (In-memory, distributed)
  4. Performance Tips (Minimal APIs, response compression)
  5. Deployment Basics (Azure, Docker, or IIS)

API Integration in React JS APP

- **Topics**:
  1. Create API using ASP.NET Core WEB API
  2. Test API using Postman or Swagger Tool
  3. Use API under React Project
  4. Create Login and Registration Module
  5. Creating Admin Dashboard using React API with CRUD Operation

course designer

I am a teacher and entrepreneur for the past 12 years, I am managing the Software development company Kangaroo Software PVT LTD and the best Software Education Institute SHIVA CONCEPT SOLUTION.I am very much passionate about teaching and helping students.

shiva gautam

Connect with us.

Email
shivaconceptsolution@gmail.com

Social Media
@shivaconceptsolution

Call us
7805063968