

.NET Full Stack Development

Course Duration ~110 Hours · 14 Weeks · Live Online · Build-Along · Deployed Enterprise-Style Applications

Curriculum reviewed: July 2026 — the .NET platform versions predictably; this syllabus tracks the current LTS line

C# & .NET

ASP.NET Core

EF Core

Blazor Render Modes

.NET Aspire

Legacy Migration

In short: One language, the whole product. Build enterprise-grade applications in C# — from the data model to an interactive browser UI — on a platform with LTS guarantees, first-class tooling and the deepest enterprise footprint in our catalogue. Not the trendiest choice; one of the most durable and best-rewarded: the release cadence is predictable and the skills date slowly, which matters when you're weighing a large time commitment. The syllabus teaches the platform as it works now: **Blazor render modes as a headline module** — a per-component decision (static server, streaming, interactive server, interactive WebAssembly, automatic), not the old whole-application binary, and the single biggest source of confusion and bugs in modern Blazor; minimal APIs first-class alongside controllers with honest guidance; **.NET Aspire** for cloud-native multi-service development — barely taught anywhere; modern C# and EF Core; and Native AOT judged on evidence. Architecture is taught honestly too: a well-designed modular monolith is the right answer for most teams, and microservices respond to organisational scale, not fashion. Across 12 modules you ship an enterprise CRUD application, a multi-render-mode Blazor app, a minimal API service, a secured multi-tenant app, SignalR real-time features, an Aspire-orchestrated solution, an EF Core optimisation workshop, a **.NET Framework legacy modernisation exercise** and a defended capstone. Choose .NET for enterprise engineering — banking, insurance, healthcare, government, large product organisations — or a Microsoft-centric team; our Django/FastAPI/Node and React/Next.js courses are the Python, JavaScript and frontend-first equivalents. The AI question, answered honestly: AI writes C# fluently and gets DI lifetimes, async correctness, EF Core query patterns and render-mode decisions wrong — the value is architecture, review, debugging and operational judgment.

Duration	14 weeks · ~110 hours · live + hands-on	Format	Live online · build-along · deployed enterprise-style applications
Prerequisite	Basic programming in any language. C# and web fundamentals taught from first principles. No prior .NET experience required.	Curriculum	12 modules, from the C# platform to deployment, legacy migration and capstone
Portfolio	8 deployed, enterprise-style projects plus a defended capstone	Ideal for	Fresh IT graduates targeting enterprise roles, and developers on .NET Framework or older .NET modernising

Why Learn .NET When AI Writes C#?

AI writes C# and scaffolds controllers fluently, and gets the things that matter wrong — dependency injection lifetimes, async correctness, EF Core query patterns that trigger N+1 problems or load an entire table into memory, and render-mode decisions in Blazor. Those surface in production, not in a demo. The value is architecture, review, debugging and operational judgment. You'll use AI tooling as an accelerator you review, and learn the failure patterns generated .NET code repeats — so you ship its speed without shipping its mistakes.

Course Highlights

- Every module leads with what you build and ship, not what you "learn about"
- One language end to end — C# from data model to browser UI
- Blazor render modes as a headline module — per-component decisions, prerendering, knowing where your code runs
- DI lifetimes taught as the most common source of subtle production bugs
- EF Core in depth — N+1 kills, bulk operations without loading entities, reading real SQL
- .NET Aspire for cloud-native development — barely taught anywhere, first-class here
- Architecture honesty: the modular monolith as the sensible default
- A legacy modernisation track — .NET Framework to modern .NET, planned properly
- AI tooling as an accelerator you review, never a substitute for understanding

Hands-on Projects & Portfolio

- **Enterprise CRUD Application** — layered architecture, EF Core migrations and full test coverage
- **Multi-Render-Mode Blazor App** — deliberately exercising multiple render modes, with a written rationale for each choice
- **Minimal API Service** — validation, OpenAPI docs, versioning and problem-details error handling
- **Secured Multi-Tenant App** — policy-based authorisation and external identity
- **Real-Time Feature with SignalR** — live updates, presence and reconnection handling
- **Aspire-Orchestrated Solution** — multi-service solution with telemetry and service discovery
- **Query Optimisation Workshop** — find and fix N+1 and memory-loading problems with profiler evidence
- **Legacy Modernisation Exercise** — assess an older .NET Framework codebase and produce a migration plan
- **Capstone** — production-grade full-stack application, deployed, monitored and defended

Tools & Stack

C# · .NET · ASP.NET Core · Blazor · Entity Framework Core · SignalR · .NET Aspire · xUnit · Moq · Playwright · Testcontainers · SQL Server · PostgreSQL · Redis · Docker · Kubernetes · Azure · OpenTelemetry · Serilog · Visual Studio · Git · GitHub Actions

Career Outcomes

.NET has the deepest enterprise footprint of any platform in our catalogue — banking, insurance, healthcare, government, manufacturing and large product organisations run on it — which makes demand steady rather than cyclical, and heavily represented in India's enterprise services and global capability centres. A very large volume of existing .NET Framework code still needs modernising, which is durable, well-rewarded work — and the predictable LTS cadence means these skills date far more slowly than frontend framework skills. This course targets:

- .NET Developer · Full Stack Developer (.NET) · Backend Engineer · Software Engineer in enterprise product teams
- Solution Architect (with experience) · Cloud Engineer (Azure-centric) · Technical Lead

Who This Course Is For

- Final-year students and fresh IT graduates targeting enterprise engineering roles
- **Developers already working on .NET Framework or older .NET who need the modern platform** — a very large audience; the legacy track was built for you
- Java, Python, PHP and Node developers moving into the Microsoft ecosystem
- Frontend developers who want backend and full-stack depth in a single language
- Support, QA and services engineers moving into development in enterprise organisations
- Teams standardising on Azure and Microsoft tooling

Module 1: C# and the Modern .NET Platform

10 Hours

YOU SHIP

- A working C# toolkit of console and library projects — records, pattern matching, LINQ and async throughout — built with the SDK and CLI, not just the IDE

TOPICS

- The unified platform and its release cadence; the SDK and CLI; project and solution structure
- The type system; records and pattern matching; nullable reference types; LINQ properly
- async/await and the task model; the recent language features that shorten everyday code

Assessment: Quiz + Lab

Module 2: Object-Oriented & Clean Code Foundations

8 Hours

YOU SHIP

- A refactor of a tangled codebase into C# a reviewer doesn't have to untangle — abstractions that earn their keep, composition over inheritance, exceptions designed rather than sprinkled

TOPICS

- Interfaces and abstraction that earn their keep; SOLID applied rather than recited
- Composition over inheritance; generics and collections
- Exception design; reviewable C#

Assessment: Quiz + Lab**Module 3: ASP.NET Core Fundamentals**

9 Hours

YOU SHIP

- A configured web application with custom middleware, a hosted background service — and a lab reproducing the classic DI-lifetime bug (a scoped service captured by a singleton) before fixing it

TOPICS

- The hosting model and request pipeline; middleware
- Dependency injection and service lifetimes — the most common source of subtle production bugs
- Configuration and options; environments; logging; hosted background services

Assessment: Quiz + Lab**Module 4: Building APIs**

10 Hours

YOU SHIP

- A minimal API service with validation, OpenAPI docs, versioning and problem-details errors — plus a SignalR real-time feature with presence and reconnection — and a written minimal-vs-controllers choice

TOPICS

- Minimal APIs and controller-based APIs side by side — guidance on choosing; model binding and validation
- REST design; versioning; OpenAPI documentation; error handling and problem details; pagination and filtering; file uploads
- Server-sent events; real-time communication with SignalR

Assessment: Quiz + Lab**Module 5: Data Access with EF Core**

10 Hours

YOU SHIP

- The query optimisation workshop: N+1 and whole-table-in-memory problems found and fixed with profiler evidence — plus a zero-downtime production migration pattern executed end to end

TOPICS

- Modelling and relationships; migrations including production and zero-downtime patterns; change tracking and why it surprises people
- Eager versus lazy loading and eliminating N+1 queries; projections; bulk operations without loading entities; raw SQL escape hatches
- Transactions; concurrency handling; reading generated SQL

Assessment: Quiz + Lab**Module 6: Blazor & the Frontend**

10 Hours

YOU SHIP

- A Blazor application deliberately exercising multiple render modes — with a written rationale per component, state persisted across the prerender boundary, and a diagnosis lab on where the code actually executes

TOPICS

- Razor components, parameters, lifecycle and rendering; forms and validation; state management; JavaScript interop kept minimal; component libraries
- Render modes as a per-component decision: static server, streaming, interactive server, interactive WebAssembly, automatic
- Prerendering and state persistence across the boundary; diagnosing where your code is actually executing

Assessment: Quiz + Lab

Module 7: Authentication, Authorisation & Security**9 Hours****YOU SHIP**

- A secured multi-tenant application: Identity plus an external provider, policy-based authorisation, modern credentials, hardened headers — checked against the OWASP risks with .NET-specific fixes

TOPICS

- ASP.NET Core Identity; cookie and token authentication; OAuth2 and OpenID Connect; external identity providers and Entra ID; passkeys and modern credentials
- Role and policy-based authorisation; multi-tenancy
- Secrets management; HTTPS and headers; the OWASP risks with their .NET-specific fixes

Assessment: Quiz + Lab

Module 8: Application Architecture**9 Hours****YOU SHIP**

- A modular monolith with clean module boundaries and a vertical-slice feature — plus a written when-microservices-actually-win analysis for the same system

TOPICS

- Layering, clean architecture and vertical slices; the modular monolith as the sensible default; domain modelling
- CQRS and mediator patterns used judiciously; repository and unit-of-work debates settled pragmatically
- When microservices genuinely become the right answer — organisational scale, not fashion

Assessment: Quiz + Lab

Module 9: Testing & Quality**8 Hours****YOU SHIP**

- A fully tested feature slice — xUnit units, integration tests on the in-memory host, database tests in containers, Blazor component tests, and Playwright end-to-end — gated in CI

TOPICS

- Unit testing with xUnit; mocking and test doubles; integration testing with the in-memory test host
- Database testing with containers; testing Blazor components; end-to-end coverage with Playwright
- Coverage that means something; static analysis and linting in CI

Assessment: Quiz + Lab

Module 10: Cloud-Native .NET**10 Hours****YOU SHIP**

- A multi-service solution orchestrated with .NET Aspire — service discovery, configuration, health checks, resilience policies and distributed tracing across the whole thing

TOPICS

- Containerising .NET properly; container images without hand-written Dockerfiles
- .NET Aspire for orchestration, service discovery, configuration and telemetry across a multi-service solution
- Health checks; resilience and retry policies; caching; message-based communication; distributed tracing with OpenTelemetry

Assessment: Quiz + Lab

Module 11: Performance & Operations

8 Hours

YOU SHIP

- A slow endpoint diagnosed and fixed with profiler and allocation evidence — plus a Native AOT build compared honestly on startup, memory and container size against its constraints

TOPICS

- Profiling and diagnostics; memory and allocation; async pitfalls and deadlocks
- Response caching and output caching; Native AOT and startup cost — when ahead-of-time compilation is worth its constraints
- Load testing; structured logging; error tracking; diagnosing with evidence rather than guesswork

Assessment: Quiz + Lab**Module 12: Deployment, Legacy Migration & Capstone**

9 Hours

YOU SHIP

- The legacy modernisation exercise — an older .NET Framework codebase assessed with a migration plan produced — and your capstone: one production-grade full-stack application, deployed, monitored and defended

TOPICS

- CI/CD pipelines; deployment to cloud platforms and containers; configuration and secrets across environments; database migration strategy under real traffic; blue/green and canary releases
- The legacy modernisation track: .NET Framework to modern .NET, and how to plan it
- Capstone design, build, deploy and defense

Assessment: Capstone Project + Review