Mousa Amiri Motlagh

Junior .NET Backend Developer

Email: mousaamiri.code@gmail.com | GitHub: github.com/mousaamiri
Portfolio: mousaamiri.ir | Availability: **Remote / Freelance only**

Professional Summary

Self-taught junior backend developer focused on C# and ASP.NET Core, with no formal professional experience in software. I've built my skills through several complete personal projects — not scattered tutorials — including an online course marketplace with a layered architecture and 170+ handlers, a published desktop application with 164 automated tests, and a live portfolio website with CI/CD. I take documentation, testing, and disciplined Git workflow seriously on every project, and I'm now looking for the opportunity to work on a real team, get feedback from more experienced developers, and grow faster.

Technical Skills

Languages & Frameworks

- C# — Proficient
- .NET 8/10 — Proficient
- ASP.NET Core Web API — Proficient
- ASP.NET Core MVC/Razor — Familiar, hands-on
- WPF — Proficient

Databases & ORM

- SQL Server — Proficient
- Entity Framework Core — Proficient
- PostgreSQL — Familiar, hands-on

Architecture & Patterns

- Repository Pattern / Unit of Work — Proficient
- Clean Architecture — Familiar, hands-on
- MVVM — Familiar, hands-on
- CQRS with MediatR — Familiar, hands-on
- Outbox Pattern with Quartz — Basic familiarity

Testing

- xUnit, FluentAssertions, Moq — Familiar, hands-on

Security

- JWT Bearer Authentication, Rate Limiting, User Secrets, secure/HttpOnly cookies — Familiar, hands-on

Caching & Performance

- IMemoryCache, Cache-Aside — Familiar, hands-on
- Redis / IDistributedCache, BenchmarkDotNet, Task vs ValueTask — Basic familiarity

Tools & DevOps

- Git/GitHub — Proficient
- Docker — Familiar, hands-on
- GitHub Actions (CI/CD) — Familiar, hands-on
- Docker Compose — Basic familiarity

Basic Frontend & Language

- HTML, CSS, JavaScript — Basic familiarity
- English — Basic level (reading technical docs, writing commits)

Projects

Vitastic — Online Course Marketplace

Personal Project — Work in Progress (WIP)
github.com/mousaamiri/Vitastic

- Designed and developed an online course marketplace with user registration, course browsing, shopping cart, orders, discounts, wallet, payment, and purchased-course access flows.
- Structured the solution into five Domain, Application, Infrastructure, API, and Web projects using Clean Architecture and dependency inversion; the MVC interface communicates with the REST API over HTTP.
- Implemented 12 business areas with 116 Commands, 68 Queries, and 171 Handlers in the Application layer using MediatR and pipeline behaviors for validation, logging, and Unit of Work handling.
- Applied DDD in practice using Aggregate Roots, Entities, Strongly Typed IDs, Value Objects, and 57 domain-event types across domains such as courses, orders, carts, payments, and wallets.
- Implemented the Outbox Pattern with a SaveChangesInterceptor to persist domain events, then processed pending messages through a Quartz job running every 10 seconds (up to 20 messages per run) and published them with MediatR.
- Set up Docker Compose for PostgreSQL and the project's services.

C# · .NET 10 · ASP.NET Core Web API/MVC · PostgreSQL · EF Core · MediatR · Quartz · Docker

Wazhechin — Desktop Library Lending Management System

Personal Project — Stable v1.0.0 release
github.com/mousaamiri/Wazhechin

- Designed and developed a Windows desktop application for managing books, authors, categories, members, addresses, and the book lending/return workflow, with a statistical dashboard.
- Released a stable v1.0.0 build as a downloadable ZIP package; the application automatically seeds an initial administrator account on first run.
- Structured the solution into eight projects using Layered Architecture and MVVM, separating the UI, ViewModel, business logic, and data-access layers with Repository, Unit of Work, and DbContextFactory patterns.
- Built a custom wrapper system around ModelWrapper<T> for change tracking and validation, along with ChangeTrackingCollection<T> to synchronize added/removed/modified items with the underlying model.
- Investigated and handled a real circular-reference challenge in nested wrappers, such as the Member–MemberAddress relationship with navigation properties and lazy loading.
- Wrote 164 xUnit tests across 29 test files covering wrappers, change tracking, validation, navigation, and dashboard behavior.

C# · .NET 10 · WPF · EF Core · SQL Server · MVVM · MahApps.Metro · xUnit

Portfolio — Personal Portfolio Website (live at mousaamiri.ir)

Personal Project — Live and in active use
github.com/mousaamiri/Portfolio · mousaamiri.ir

- Designed and developed a bilingual (Persian/English) portfolio website with Clean Architecture across five Domain, Application, Infrastructure, API, and Web projects; the frontend consumes data from the API over HTTP instead of accessing the database directly.
- Built an admin panel for full CRUD over site content, with separate public and admin API areas (34 controllers, 100 HTTP actions).
- Implemented server-side language detection with cookie persistence and full RTL support for Persian.
- Implemented JWT Bearer authentication for the API, secure HttpOnly cookies for the Web admin panel, and rate limiting on the login and contact forms.
- Ran automatic EF Core migrations and idempotent content seeding at startup, keeping secrets out of source control via User Secrets.
- Wrote 415 automated tests across 5 test projects using xUnit, FluentAssertions, and Moq.
- Set up CI/CD with GitHub Actions to publish and separately deploy the API and Web to a Plesk host over FTPS, with retry handling and app_offline.htm.

.NET 10 · C# · ASP.NET Core Web API · ASP.NET Core MVC/Razor · EF Core · SQL Server · JWT · xUnit · GitHub Actions

DotNet Performance Lab — .NET Performance Experiments

Supplementary lab project
github.com/mousaamiri/DotNet-Performance-Lab

- Implemented Cache-Aside with IMemoryCache and Redis/IDistributedCache, including expiration and cache invalidation.
- Simulated Cache Stampede and implemented per-key double-checked locking with SemaphoreSlim.
- Compared sync-over-async with proper await using a custom C# load-testing tool; in the first measured run with 250 concurrent requests, the proper-async version reduced runtime from about 26.7 seconds to about 1.8 seconds (dependent on cold thread-pool/cache conditions).
- Benchmarked Task vs ValueTask with BenchmarkDotNet: in the ready-data scenario, ValueTask took about 5.85 ns with no allocation versus about 12.14 ns and 144 bytes of allocation for Task.
- Explored response caching and Gzip/Brotli response compression.

.NET 10 · C# · ASP.NET Core Minimal API · Redis · Docker · BenchmarkDotNet

Continuous Learning

- Self-directed, project-based learning of C#/.NET, focused on building complete projects rather than scattered courses.
- Ongoing practice with HTML, CSS, and JavaScript to better understand the UI layer that connects to backend APIs.
- Daily use of Git/GitHub for version control, with a dedicated README documenting each project.

Why Me

I entered this field with no formal work experience in software, but by building several complete projects — not just exercises — I've shown I can learn quickly, document my work, write tests, and see a project through to the end. I'm now looking for a real team where code review and feedback can help me measure my engineering discipline against industry standards and work on real problems.