



Maksim Harbacheuski

Senior Backend Engineer | .NET | Distributed Systems | FinTech

📍 Minsk, Belarus ✉ gorbachewski.m@gmail.com [in](#) LinkedIn [lastuser42](#) [Habr](#)

I'm a software engineer with 5 years of experience building backend systems for FinTech and enterprise applications. Specialized in .NET, distributed systems, database performance and high-volume data processing. Experienced with ASP.NET Core, PostgreSQL, SQL Server, Redis, RabbitMQ and Kubernetes, with a strong infrastructure background spanning Linux, networking and self-hosted systems.

TECHNICAL SKILLS

Backend: C#, .NET, ASP.NET Core, Web API, EF Core, REST, GraphQL

Architecture: Distributed Systems, Microservices, Modular Monolith,DDD, CQRS, EDD

Databases: PostgreSQL, Microsoft SQL Server, MongoDB, SQL Optimization, Query Execution Plans

Messaging & Caching: RabbitMQ, Kafka, Redis, Multi-Level Caching

Testing: NUnit, Integration Testing, TestContainers, Apache JMeter

DevOps & Infrastructure: Docker, Kubernetes, GitLab CI/CD, Jenkins, ArgoCD, Linux

Networking: MikroTik, VLAN, Routing, Firewalling, VPN, WireGuard

EXPERIENCE

FXOpen — .NET Developer

Nov 2023 - Present

- Large-scale financial platform with a modular monolith and standalone services, **covering ~200 business modules and databases** with tables containing up to billions of records
- Designed financial workflows for deposits, withdrawals and transaction processing. Built asynchronous communication with payment providers and trading platforms
- Designed high-performance reporting and data export services. Reduced critical trading report generation time **from ~15 seconds to under 1 second**
- Designed payment processing services and integrations with Dukascopy, bePaid, Oplati and other payment providers
- Designed trading platform (TP) integrations using DLLs, APIs and socket-based communication with MT4/MT5, TT and other external trading systems
- Optimized SQL Server **workloads at billion-row scale**. Worked with execution plans, indexing, isolation levels and performance-critical RAW SQL queries
- Designed an S3-based caching mechanism for large report datasets with **Zstandard (ZSTD)** compression. Built a reusable approach for reducing storage, transfer overhead and response times
- Designed a multi-transaction refund mechanism that could distribute a partial refund across multiple source transactions while preserving financial consistency
- Reliable payment processing with webhooks, callbacks, retries, idempotency, duplicate transaction protection and transaction state management
- Protection against SMS provider abuse and **SMS flooding** through system-wide rate limiting, CAPTCHA, Telegram verification codes and cooldowns
- Implemented **KYC/AML** and security features including Google 2FA, rate limiting, per-user IP whitelists/blacklists and granular role-based access control

- Migration from ASP.NET MVC / .NET Framework to ASP.NET Core Web API, including REST API standards, versioning, authentication, authorization, validation and reusable models
- Conducted code reviews and participated in technical planning and estimation. Proposed architectural improvements and drove their adoption across the development team

Skills: ASP.NET Core | ASP.NET Framework | MSSQL | LLBLGen Pro | EF Core | MongoDB | REST | RAW SQL to JSON | S3 | ZSTD | Modular Monolith | Unity DI | Repository | Database First | JWT | GitLab CI/CD | Azure | TFS | FluentValidation | Nginx | IIS | 2FA | Telegram Codes

Scheduling & Placement Management System — .NET Developer

Nov 2021 - Nov 2023

- **Multi-level caching** with in-memory cache and Redis, including TTL, **tag-based invalidation** and adaptive cache lifetimes based on data volatility
- **Request coalescing (single-flight)** for expensive read operations, allowing concurrent requests for the same data to share a single in-flight **Task**
- DTO-based read models for read-heavy scenarios, **making query results cacheable** and reducing database load and memory consumption
- Production **memory optimization from ~80 GB to ~12 GB** peak usage. The issue was caused by large **SplitQuery/AsNoTracking** workloads and excessive materialization of records; DTO projections, caching and request coalescing addressed the underlying bottlenecks
- Optimized database performance through query analysis and index design, improving performance of complex read and reporting workloads
- **Shared row-level caching** across the reporting subsystem, allowing different reports to reuse the same cached data instead of maintaining separate per-report caches
- Common reporting architecture and data-access patterns across multiple reports, reducing duplicated implementations and improving cache reuse
- Refactoring of calculation-heavy business logic, consolidating duplicated mathematical and domain calculations and reducing code duplication by approximately **30%**
- **Genetic algorithm** for automated placement optimization with configurable constraints and custom penalty functions
- Redesigned time interval modeling to eliminate boundary-related calculation errors.
- Amazon S3 integration replacing MinIO-specific APIs with the standard AWS SDK and **removing application-level vendor coupling**
- Production Git Flow, application versioning, commit tagging and changelog management; Jenkins CI/CD with SonarQube, Docker image builds and Kubernetes deployments through ArgoCD
- Unit testing with NUnit for complex domain calculations, business logic and optimization workflows

Skills: ASP.NET Core | PostgreSQL | EF Core | Redis | GraphQL | DDD | CQRS/ES | MediatR | Microservices | Kafka | Unit Of Work | S3 | GeneticSharp | NUnit | TestContainers | Git Flow | Jenkins | ArgoCD | SonarQube | Kubernetes

EDUCATION

Belarusian State University of Informatics and Radioelectronics

Computers, Systems and Networks

Sep 2021 - Jul 2025

LANGUAGE

Russian

Native

English

Professional working