Resume · Senior backend

Victor Razuvaev

Senior backend engineer. High-load PHP and Yii services, and the reliability libraries underneath them.

github.com/rasuvaeff

packagist.org/packages/rasuvaeff

t.me/rasuvaeff

rasuvaeff@gmail.com

rasuvaeff.github.io

Profile

Senior backend engineer with fifteen years of commercial PHP, ten of them in advertising and real-time bidding. I own services end to end: the domain model, the data stores, the failure behaviour under load, and the release. Where a problem repeats, I extract it into a library. The public record is 89 packages on Packagist with Psalm level 1, no suppressions, a mutation gate on every package (median 95%), SemVer and a backward-compatibility check before a major.

What I bring

High-load design

RTB and ad-serving paths where milliseconds and correctness are one problem. MySQL for operations, ClickHouse for impressions, Redis and Memcached in front. ClickHouse work is published as clickhouse-toolkit.

Failure behaviour

Deadlines through every layer, retries with backoff, circuit breakers, bulkheads, idempotency keys, a transactional outbox. Published as context, yii3-outbox, yii3-idempotency, resilience.

Architecture

Clean Architecture on Yii3 with layer boundaries enforced by Deptrac. Core plus swappable backends, with the DI rules written down and checked by a merge harness.

Testing discipline

Mutation testing as the gate, property-based tests with shrinking, doubles that are real calls. I built the tools: property-testing-core, property-testing-openapi, understudy, doc-exec.

Security-minded

SSRF-safe fetching with pinned-http-client, caller-scoped idempotency, fail-closed MCP endpoints, SHA-pinned CI audited with zizmor.

Models in the backend

The application is an MCP server: assistants call domain tools with per-user RBAC, rate and budget limits and an audit log. See yii3-mcp.

Selected systems

cska.in. A fan community rebuilt on a 20-year-old database and the same domain. Yii3, PHP 8.5, PostgreSQL, Redis, Temporal, Centrifugo. Live match calendar, a long archive of player and head-to-head statistics, stories, private messages, a Telegram bot and mini-app. An MCP door lets a personal agent use the same handlers as the site, with per-key scopes, quotas and an audit log. Runs on 31 of my packages. Write-up (in Russian).

Ad-tech, ten years. Advertising networks with RTB integrations, buying and selling traffic. MySQL for operational data, ClickHouse for impression analytics, Redis and Memcached for caching.

ecosystem-sandbox. A runnable Yii3 shop on 56 of the packages. Checkout and payment run outbox, idempotency, retries, circuit breaker, webhooks and telemetry together. Source.

Open source

The libraries are the portfolio. Thirteen have their own pages, among them understudy (test doubles as real calls), property-testing-openapi (contract fuzzing for PHP), doc-exec, context, yii3-outbox and yii3-mcp. The catalog has the rest. Every package ships an `llms.txt` reference, and forty ship an agent skill.

How the work is produced

Contract first. Layout, style, CI, docs and security rules come from a versioned rulebook, php-package-toolkit.

Write, then gate. An agent writes code, tests and docs, then the gate runs: Psalm level 1, mutation testing, a compatibility check, SHA-pinned CI, property tests where an invariant exists.

Judgment stays mine. I design the API, the boundaries and the release. The agent does not merge, tag or publish. The workflow page has the longer form.

Stack

Languages	PHP 8.3–8.5 · SQL · Bash
Frameworks	Yii3 · Yii2 · PSR standards
Data	PostgreSQL · MySQL · ClickHouse · Redis · Memcached
Messaging	Centrifugo · Temporal · Transactional outbox
Observability	OpenTelemetry · Jaeger · Prometheus · Grafana
Quality	Psalm · Testo · PHPUnit · Infection · Rector · Deptrac
Platform	Docker · Linux · Nginx · GitHub Actions · MCP

What I want to be building

Low-latency services where the tail latency and the correctness are the same problem.

Distributed patterns done as libraries: outbox, idempotency, retries with backoff, bulkheads.

A model inside the backend as a permissioned caller with an audit log, not as a chat box beside it.