Przemysław Alexander Kamiński (aka xlii & exlee)
przemyslaw@kaminski.se, + 48 728967163
Warsaw, Poland

https://xlii.space
https://github.com/exlee
https://codeberg.com/exlee
https://linkedin.com/in/axkaminski

Summary

Staff-level generalist, 20 years. Currently focused on Rust systems programming. Background in distributed systems, high-availability backends, and performance tooling. Independent engineering work interspersed with medical leave (2025). Remote, Warsaw.

Selected Experience

- Independent Systems Engineering (05.2025 — Present)
  Independent Developer — Rust, Go, Haskell, C
- Bluecode AG (05.2023 — 04.2025) — Payment Processor — Distributed Systems, HA
  Sr. Backend Engineer — Elixir, Rust, Go, OTEL, Distributed Systems
- Subscribe+ (07.2022 — 02.2023) — Alternative Fund Investment Processor
  Sr. Fullstack Engineer — Rule-based data processing, Ruby on Rails, React
- SameSystem LLC (05.2021 — 11.2021) — Retail Workforce Management
  Sr. Backend Engineer — Integration, Ruby on Rails, React
- Owens Corning (2019 — 2021) — Main website development
  Sr. Fullstack Engineer — Ruby on Rails, React
- Entropy-1 sp. z o.o. (2016 — 2022) — Contracting LLC
  CEO/CTO — React, Ruby on Rails, Unity
- WeWantToKnow AS (02.2013 — 10.2017) — Edu Game Development Company
  Sr. Backend Engineer — Unity, Ruby on Rails, Ember
- Voxcommerce sp. z o.o. (07.2008 — 08.2012) — Realestate Software Provider
  Staff Engineer — PHP 4/5, Django, JavaScript

Technology Experience

- 10+ years: JavaScript, PostgreSQL, Shell-scripting
- 5-10 years: React, Ruby on Rails
- 2-5 years: Rust, Python, Elixir, Go
- 1-2 years: TypeScript, Cuelang
- 6-12 months: TLA+, Prolog, Perl, Clojure, Common Lisp, Erlang, C#
- 1-6 months: C, Haskell, ObjectiveC, Java, OCaml

Other information

- Master’s Degree in Software Engineering (major: Distributed Systems & Database Engineering)
- Bachelor’s Degree in IT Project Management
- Certified Educational Trainer by TenStep Poland
- Generalist with strong pull toward backend systems, performance engineering, and distributed architecture

Selected Projects

Independent Systems Engineering (05.2025 – Present)

- **Emacs Core**: Authored and upstreamed line-spacing patch (2–3 months review cycle) and macOS performance fixes. Patch shipped in mainline Emacs.
- **kak-univ-search**: Kakoune search framework in Rust — async FIFO-based communication layer enabling concurrent search across buffers, file listings, and content without blocking the editor.
- **ssort**: Stream sorting utility in Go with concurrent pipeline architecture.
- **hn-jobs-evaluator**: Rust/egui desktop tool that ingests HN "Who's Hiring" threads, scores each listing via LLM against a resume and requirements set, and ranks by alignment. Built to solve a real problem: filtering 300+ listings took hours manually.
- **pikchr.pl Suite**: Diagramming ecosystem — DiagramIDE (multi-window, workspace-based Rust/egui IDE), pikchr.pl (live-preview editor with event-driven UI and async rendering pipeline), PikchrPro (Prolog-to-Pikchr SVG compiler). Async used throughout to avoid blocking the immediate-mode GUI.
- **tmplr**: Single-binary Cabal-style template system: one template file expands to a full project file tree.
- **vibe-coded**: Rust CLI that clones a Git repo and runs heuristic rules to detect AI-generated codebases.
- **codelooker**: Generates annotated, syntax-highlighted HTML reports from file-line specifications.

Bluecode AG — Selected Work (2023–2025)

Bluecode is a European payment processor with strict HA requirements and a complex multi-party settlement architecture.

- **Dev Environment Tooling (Go)**: The existing stack took 1–2 weeks to onboard; CI ran >1.5 hours. Built a Go-based orchestration tool that containerized the full stack with full dependency parallelization — reduced full test suite to minutes on a developer machine. Added OTEL injection, chaos testing, and network-partition simulation.
- **Settlement Reporting**: Designed and implemented settlement reporting pipeline for external integrators — cross-system reconciliation across distributed, non-contiguous event flows.
- **Cross-crate Type Unification (Rust)**: Identified and consolidated duplicated domain types (Currency, Money, etc.) spread across a multi-crate workspace. Established canonical shared crate with migration path.
- **Partner Scheme Integration (Rust/Elixir)**: Designed and partially implemented full integration with a payment scheme partner — protocol alignment, API integration, event handling across async boundaries.
- **Telemetry (OTEL)**: Implemented and debugged distributed tracing across a non-contiguous service mesh. Significant troubleshooting of trace propagation gaps in async Elixir/Rust boundaries.
- **Event System Upgrades**: Implemented new and upgraded event types resulting from payment process changes; maintained backward compatibility across distributed consumers.
- **Internal Talks**: Delivered sessions on supply chain attacks, macro security in Rust, and safe macro patterns.

Subscribe+ — Selected Work (2022–2023)

Subscribe+ processed alternative fund investments through complex rule-based document pipelines.

- **Rule-based Document Processing**: Implemented multi-stage document transformation pipeline with branching rule evaluation across investment instrument types.
- **eSignature Debugging**: Diagnosed a broken eSignature integration using MITM proxy — traced failure to undocumented API behavior and patched the integration.
- **JS Stack Upgrade**: Inherited a several-years-stale Node/npm stack. Devised a safe upgrade strategy by resolving each dependency against its latest version at a specific historical date rather than blindly upgrading to current latest — coordinated the entire dependency graph forward in steps without breakage.

NPM Time Machine

A dependency upgrade technique for stale npm trees: instead of resolving to current latest (which produces incompatible combinations across an old graph), resolve each package to its latest version as of a specific past date — then advance the date in increments. Implemented during the Subscribe+ stack upgrade; generalises to any project with years of accumulated drift. Initial implementation in Ruby, rewritten in Rust and open sourced after engagement.

OwensCorning — Selected Work (2019–2021)

Site Performance & Infrastructure

- **Postgres Optimization**: Profiled and restructured data loading on owenscorning.com — reduced full page load from ~400ms to under 100ms through query restructuring and eager loading.
- **Stack Upgrade**: Led seasonal theme speed-up project — modernized the pipeline to cut time-to-ship for seasonal site changes.

Roof Builder

- Led the 3D Virtual Roof Builder end-to-end: architecture, data store, rule-based state transitions, and implementation (Ruby on Rails + React).
- Designed the contractor embedding API — allowed third parties to embed the configurator with customization hooks for branding and product filtering.
- Volunteered a parallel implementation during an uncertain agency engagement; internal version chosen over agency's — richer in features and more stable.
- Built without prior 3D experience.

Selected Articles

- Upgrading Node Stack (2023)
  https://xlii.space/eng/upgrading-node
- Emacs: the macOS Bug (07.2025)
  https://xlii.space/eng/emacs-the-macos-bug
- State of Opinion - Programming Languages - Autumn 2025 Edition (10.2025)
  https://xlii.space/eng/opinion-languages-autumn2025/
- The Four Language Waltz: A Tale of Allocators and Regret (01.2026)
  https://xlii.space/eng/the-four-language-waltz-a-tale-of-allocators-and-regret/
- I Hate GitHub Actions with Passion (01.2026)
  https://xlii.space/eng/i-hate-github-actions
- Cuelang: Does It All But Can It Literate? (01.2026)
  https://xlii.space/cue/cue-does-it-all-but-can-it-literate/
- Approved, Unread, Shipped (03.2026)
  https://xlii.space/blog/approved_unread_shipped/

Fun facts

- For fun I game, play piano, engage in photography and practice boxing
- I tend to do software engineering off the work hours, because I still find it fun
- This resume is a Typst text file :)