V

Backend Engineer — Distributed Systems, Compilers, FFI/Interop
ky5u7dse7@mozmail.com | github.com/rinz13r | Bangalore, India

SUMMARY

Backend engineer with 4+ years of production experience across Rust and C#/.NET, shipping high-throughput services on Azure (Substrate, ADLS) and building compilers, FFI bridges, and code-generation tooling. Comfortable owning systems end-to-end, from architecture and performance tuning to deployment infrastructure and cross-team rollout.

TECHNICAL SKILLS

Languages: Rust, C#, .NET, TypeScript, Python, Swift, OCaml, JavaScript, SQL

Cloud & Infrastructure: Azure (ADLS, Bicep, EV2), Substrate (M365), RBAC, CI/CD

Databases: MongoDB, ADLS (data lake)

Systems & Tooling: Compilers, FFI/Interop, P/Invoke, proc-macros, Microsoft Orleans, MSBuild, NuGet, dotnet SDK, Git

Concepts: Distributed Systems, Backend Engineering, Performance Optimization, Async Runtimes, Code Generation, Programming Language Design

EXPERIENCE

Software Engineer II — Microsoft, Viva Insights (EXP DCP Platform)

May 2025 – Present | Bangalore, India

- Owned the design and end-to-end implementation of the Weve Streaming Layer, a Substrate-hosted M365 service serving HR Data and Copilot Metrics scenario datasets to ~8M long-tail M365 tenants by streaming directly from ESS over async HTTP (REST) instead of staging through ADLS, removing the staging hop and its storage/compute cost.
- Part of a small group driving the workstream; ran the early POCs validating direct ESS to HTTP streaming from a Substrate Model B2 handler before the work was committed to.
- Owned the ADLS provisioning layer for DSS, a new (greenfield) EXP DCP storage abstraction; authored Bicep templates and EV2 deployment pipelines for repeatable, auditable per-region rollout, including RBAC for cross-app access.
- Helped with cross-team rollout: onboarded HR Data as the first end-to-end partner and established the onboarding pattern (contracts, scenario registration, validation) for subsequent teams including Copilot Metrics.

Software Engineer, Fullstack — Achieve.ai

Jul 2021 – Feb 2025 | Bangalore, India (Remote)

- Built a new Rust codebase from scratch and migrated the PrepDNA and EffortDNA user-metrics engines from C# (10k–100k+ attempt records per run), reducing new-version iteration time from a ~3-day release cycle to on-demand compute.
- Designed and implemented a bidirectional async FFI bridge between the existing C# application and the Rust engines, with safe memory handling and error propagation across the language boundary.
- Extended NScript (in-house C#-to-JavaScript compiler powering 100% of the frontend) to C# 7/8: lowered async/await and yield to native JS async and generators (the latter enabling a clean LINQ implementation), and added tuple literals and unpacking, pattern matching, and switch expressions.
- Replaced the legacy MSBuild custom-tool integration with first-class dotnet build support and published the SDK to NuGet (Mcqdb.NScript.Sdk, Mcqdb.NScript.Cs2Jsc).

Platform Engineering Intern — Titan, Directi

Jan 2021 – Jun 2021 | Bangalore, India (Remote)

- Shipped a general client-side email filtering system (read/unread and similar predicates) and bug fixes to Titan's production iOS app using Swift, UIKit, and CoreData.

SELECTED PROJECTS

Oxidizer — Rust to C# FFI Bindings Generator

github.com/rinz13r/oxidizer

- Language-agnostic Rust FFI bindings generator (C# today): annotate Rust types and functions with proc-macros and emit type-safe C# P/Invoke bindings, eliminating manual marshalling boilerplate.
- Supports async Rust functions (callback-based FFI exposed as C# Task<T>), opaque owned handles with automatic IDisposable cleanup, and multiple slice ownership patterns (OwnedSlice, FFISlice, SliceCallback).

MQB — Type-safe MongoDB Query Builder for Rust

github.com/achieveai/mqb

- Sole author of a type-safe MongoDB query builder used in production at Achieve.ai; proc-macro-generated keypaths turn field-name typos and incorrect value types into compile errors instead of silent runtime bugs.
- Integrates with serde, respecting rename attributes, custom serializers, and other field-name transformations.

PyOnBrowser — Python 3 to JavaScript Transpiler (GSoC 2019, JdeRobot)

github.com/JdeRobot/PyOnBrowser

- Built a Python 3 to JavaScript transpiler enabling robot programming in Python on top of WebSim, a browser-based simulator; implemented Python iteration via JS generators, a from-scratch import/module system, and a host-binding shim wiring user code to the simulator.
- Designed exception handling using JavaScript's native call stack plus a generated line-number map, so Python exceptions report locations in the original source.

EDUCATION

Integrated Master's in Computer Science — IIIT Bangalore

Jul 2016 – Jun 2021 | Bangalore, India
