HANNAH KIMURA

hannah.kimura@intersystems.com

EDUCATION

Wellesley College Class of 2024
B.A. in Computer Science *Cambridge, MA*
Relevant Coursework: Data Structures, Multivariable Calculus, Combinatorics and Graph Theory, Linear Algebra, Foundations of Computer Systems, Mobile App Development, Theory of Computation

Massachusetts Institute of Technology
Cross-Registered Student, Computer Science (Course 6-3) *Cambridge, MA*
Relevant Coursework: Fundamentals of Programming, Introduction to Algorithms, Introduction to Machine Learning, Software Construction, Software Design, Probability and Random Variables, Computer Language Engineering, Design and Analysis of Algorithms

EXPERIENCE

InterSystems Oct 2025 - Present
Systems Developer *Boston, MA*

- Implementing a Language Server Protocol (LSP) backend for InterSystems ObjectScript using Rust, tower-lsp, tokio, and Tree-sitter with incremental parsing for low latency editor updates.
- Designed and built a hierarchical scope and semantic model to power features like symbol resolution, diagnostics, and go-to definition features.
- Developed workspace indexing and project state management that asynchronously scans .cls/.mac/.inc files, builds Rope-based document representations, and maintains concurrent scope trees.

InterSystems July 2024 - Oct 2025
Cloud Engineer *Boston, MA*

- Developed an AI chatbot using a RAG framework trained on domain knowledge/schema, alongside direct LLM queries, to debug FHIR-to-OMOP transformation exceptions by generating SQL to inspect relevant tables and suggest resolution steps, cutting resolution time from weeks to minutes.
- Implemented a pod-based Rundeck runner on Kubernetes, integrated with PagerDuty Run Actions to automate running diagnostic runbooks when a relevant incident is triggered.
- Migrated all cloud infrastructure to Terraform to reduce human error and improve auditability.

PROJECTS

Tree-Sitter-ObjectScript

- Enabled real-time syntax parsing, intelligent highlighting, and structural editing for ObjectScript .cls, .mac, and .inc files in modern code editors.
- Designed polyglot-aware parsing for ObjectScript .cls files, seamlessly handling embedded SQL, HTML, Python, JavaScript, JSON, CSS, XML, and Markdown.
- Configured language bindings so the grammar can be used from Rust, C, Go, Node.js, Swift, and Python.

IrisList: $LIST in Rust

- Defined an IrisValue enum and built IrisList (Vec<IrisValue>) with functional parity to $LIST, including byte (de)serialization, validity checks, and access/mutation (get, insert, remove, push, list[i], list[i] = val).
- Supports both heap-allocated and buffer-based serialization paths for memory-safe and efficient binary output.
- Benchmarked core operations using the Criterion crate to validate performance and guide optimizations.