Ryan Kugler

kuglerryan@gmail.com +1 (647)-527-7357 ryan-kugler ryankugler

Skills

Languages: Python, TypeScript, JavaScript, C++, SQL, Java
Backend: Node.js, Express, REST APIs, PostgreSQL
Frontend: React, TailwindCSS, MaterialUI
Tools: Git, Docker, AWS, Vercel, Codex **Testing:** Python, JavaScript, C++, SQL, Java
Frameworks & Libraries: React, Node.js, Express, TailwindCSS, JUnit, Pytest
Tools & Platforms: Git, Docker, GitHub Actions, PostgreSQL, AWS, Vercel

Experience

TD Bank **Feb 2025 – Present**
Vulnerability Security Engineer *Toronto, ON*
- Developed Python-based automation scripts and internal dashboards to track remediation lifecycles across large asset inventories, reducing mean-time-to-remediate by **35%**
- Engineered REST API integrations between vulnerability scanners and **JIRA** to automate ticket generation, routing, and lifecycle synchronization for developer teams
- Performed triage and remediation analysis of high-impact vulnerabilities using **Qualys**, **Veracode**, and **Snyk** across enterprise environments
- Led proof-of-concept evaluations of static and dynamic analysis tooling, expanding scanning coverage and improving developer adoption through IDE-native integrations

Automation Accelerator **Sep 2024 – Dec 2024**
Software Developer (Contract) *Toronto, ON*
- Architected client automation workflows integrating third-party APIs (**Zapier**, **n8n**, **Google Workspace**) to synchronize operational data and improve process efficiency by up to **60%**
- Designed and deployed production **React** web applications on **Vercel**, enabling intuitive user interaction with automation pipelines and monitoring tools
- Collaborated directly with stakeholders to translate business requirements into scalable API-driven technical solutions

Studica Robotics **Sep 2023 – Dec 2023**
Software Engineering Co-op *Mississauga, ON*
- Refactored the flagship application backend from **C#** to **C++**, improving overall system performance by **30%**
- Implemented real-time sensor communication using **Boost.Asio** and COM port data streams, increasing gyro precision by **15%**
- Re-architected the frontend into a responsive **React** application, reducing load times by **25%**
- Packaged the application with **Electron** for cross-platform deployment on Windows and macOS

TD Bank **Jan 2023 – Apr 2023**
Full-Stack Software Developer Co-op *Toronto, ON*
- Designed and implemented backend business-logic services in **Java** and REST APIs consumed by a **React** frontend, automating credit-card eligibility workflows and reducing approval time by **65%**
- Developed reusable, component-driven UI architecture using **TailwindCSS** and **MaterialUI**, improving maintainability and decreasing new feature development time by **20%**
- Wrote and maintained **60+** automated **JUnit** test suites with **Mockito**, increasing legacy defect detection coverage by **80%** and enabling safer refactoring
- Collaborated in an Agile development environment using Git-based workflows, code reviews, and CI-driven validation to ensure production-ready releases

Projects

2D Bin Packing with Ant Colony Optimization | *Python, NumPy, Matplotlib*
- Implemented a metaheuristic **ant colony optimization** algorithm to solve the 2D bin-packing problem, minimizing waste and maximizing placement efficiency
- Designed pheromone-based exploration and heuristic weighting strategies to improve convergence and solution quality across generations
- Built real-time visualization of packing states using **Matplotlib** to support debugging and performance analysis

Real-Time Operating System (RTOS) | *C, C++, Keil MCB1700 IDE, NXP LPC1768 Microcontroller*
- Designed and implemented an RTOS in **C** supporting priority scheduling, preemption, memory management, and console I/O
- Developed an earliest-deadline-first scheduler to efficiently allocate CPU time across real-time tasks
- Programmed interrupt handling and UART-based task creation for interactive system control

LL(1) Compiler | *Java, VHDL, JUnit*
- Built multiple compilers for **LL(1)** grammars using recursive-descent parsing and parser-generator techniques in **Java**
- Generated SVG visualizations from abstract syntax trees to improve debugging and enable common sub-expression elimination
- Optimized compilation using fixed-point iteration and De Morgan's transformations, reducing compile time by **40%**

Education

University of Waterloo **Apr 2024**
*BASc. in **Computer Engineering** – Graduated with Distinction* *Waterloo, ON*
- Coursework: Data Structures & Algorithms, Real-Time OS, Compilers, Software Architecture, Databases, Concurrency