Last updated in Aug 2026

Ryland Donovan

Full-Stack Developer
Toronto, ON · RylandDonovan@gmail.com · www.rylanddonovan.com · github.com/RrylandD

Summary

Full-stack developer, 6 years at Hatch. Took an internal engineering platform from prototype to production and own it end to end: Angular on the front end, ASP.NET Core APIs, and event-driven services on Kubernetes.

Experience

Full-Stack Developer *May 2019 – present*
Hatch
Digital Project Delivery, the internal software group at a global engineering and consulting firm

- Lead developer on the Digital Engineering Platform and the services behind it, from prototype to production. Own architecture, code review, and releases. Used by hundreds of engineers across dozens of live projects.
- Own the CI/CD pipelines in GitHub Actions and Azure DevOps: automated tests, container builds, and deploys.
- Migrated 250 repositories and 200 developers from Azure DevOps to GitHub with full history and LFS artifacts intact, then set the org-wide branching and pull request standards.
- Built GitHub Enterprise spend reporting mapped to internal cost codes, surfacing seats billed to the wrong organization and licence tier.
- Mentor interns and junior developers, enforcing development standards and best practices.

Education

University of Guelph *Sept 2015 – June 2020*
Bachelor of Computing in Computer Science, Minor in Business

Projects

Digital Engineering Platform | Angular, ASP.NET Core, Azure
Web app for reviewing 3D models of industrial projects and assigning engineering data to model elements.

- Built the filtering layer that compiles UI filter state into ECSQL queries against the model database, letting users select and colour 3D elements by any engineering property.
- Integrated Bentley iTwin for model viewing with the enterprise systems that own the source engineering data.

Model Sync Pipeline | TypeScript, Kubernetes, KEDA, Azure Service Bus
Two background services: one writes engineering data into 3D models, the other precomputes model descriptors to Blob Storage for the app to read.

- Azure Functions publish model change events onto Service Bus; Kubernetes workers consume the queue and scale on queue depth with KEDA.
- Explicit failure handling: exponential backoff, dead-lettering for permanent failures, and per-session locking so concurrent jobs cannot collide on one model.
- Pulls engineering data from REST APIs and writes it into models as native elements, so project data is visible against the design.

Technical Skills

Languages: TypeScript, C#, SQL

Frontend: Angular, RxJS, HTML/CSS

Backend & Data: ASP.NET Core, REST APIs, Azure Functions, Entity Framework, SQL Server

Cloud & Infrastructure: Azure (AKS, Service Bus, Event Grid, Blob Storage), Kubernetes, Docker, KEDA

Practices: CI/CD, Automated testing, Code review, Production support