Mo van Eeden

vaneeden.monja@gmail.com

480 255 7140

New Castle, WA

Embedded & Systems-Oriented Software Engineer with hands-on experience building real-time telemetry pipelines in C++/Qt, integrating with vehicle hardware over serial (OBD-II), and designing testable, fault-tolerant modules. Strong focus on reliability, modular architecture, and automated testing; excited to work on high-impact mission-critical systems like Starlink flight and ground software.

Skills

Languages: C++ (Qt), Python, C#, Java
Embedded & Systems: Serial communication (OBD-II / ELM327), event-driven telemetry, real-time style polling, hardware integration, basic electronics familiarity
Testing & Reliability: QtTest, custom mocks, integration tests, failure-mode handling, regression test mindset
Tools & Platforms: Qt, Git/GitHub, Linux/WSL, VS Code/Qt Creator
Concepts: Fault-tolerant design, modular architecture (SOLID, C4 diagrams), concurrency basics, networking fundamentals, data modeling

Education:

Bachelors of Science - Computer Science
|Bellevue College|
Expected Graduation: Fall 2026

Associate of Art and Science - Computer Science
|Bellevue College|
Completed May 2023

Key Engineering Work:

Real-Time Vehicle Telemetry Dashboard (C++/Qt, OBD-II)
Bellevue College – Software Engineering Project

- Designed and implemented a real-time telemetry pipeline in C++/Qt to read OBD-II data via a USB ELM327 interface, including serial communication management, PID polling scheduler, and hex frame decoding into structured vehicle metrics (speed, RPM, coolant temperature).
- Applied SOLID and modular architecture principles to separate concerns between transport, decoding, PID registry, and UI, making it easy to extend with new signals and to reason about failures.
- Built a custom mock serial interface and QtTest-based unit and integration tests to validate behavior under noisy data, timeouts, and disconnections—mirroring hardware-in-the-loop style regression testing.
- Implemented a simple fault-tolerance strategy: connection monitoring, automatic reconnection, and graceful degradation of UI when telemetry is unavailable, improving robustness in real vehicle tests.
- Collaborated in an Agile team, owning the telemetry subsystem end-to-end—from design diagrams (C4, sequence diagrams) through implementation, test automation, and live demos in a running Chevy Blazer.

Campus | Park & Path – Smart Navigation & Parking System (Capstone)
Full-stack, sensor-integrated web system

- *Co-led architecture for a mobile-friendly web app and on-campus kiosk system that provides live-ish parking occupancy and navigation for a large college campus.*
- *Designed backend modules for sensor ingestion and camera-based lot monitoring, focusing on modular boundaries, event flow, and scalability as additional lots and sensors are added.*
- *Defined reliability and privacy constraints for edge-processed images (local comparison of lot snapshots, deletion policies) and per-spot sensors, translating ambiguous real-world requirements into concrete technical trade-offs.*
- *Created C4 architecture diagrams and sequence diagrams to document knowledge flow, components, and failure modes, aligning with distributed and fault-tolerant system design principles.*