Project Proposal: Raspberry Pi Smart Plant-Watering Controller

Name: Adam Marion

Email: adam.marion@example.com

Repository (GitHub): https://github.com/adam-marion/plants-love-rust

Project Title: “PiGrow: A Rust-Based Smart Irrigation Controller”

1. Project Topic Area

Embedded systems • IoT control • Instrumented simulation.
This project uses a single Raspberry Pi as both a hardware controller and a local web server. The system senses soil moisture with a probe, decides when to water, and drives a small pump through a relay — all implemented in Rust using safe concurrency and asynchronous I/O.

2. Project Vision and Goals

The goal is to create a self-contained, Wi-Fi-accessible irrigation controller that autonomously measures, decides, and acts while presenting a live simulation of soil hydration.

System Overview
Hardware Layer:
- Raspberry Pi 4 (or 3 B+)
- YL-69 soil-moisture sensor with MCP3008 ADC (https://www.instructables.com/Soil-Moisture-Sensor-Raspberry-Pi/)
- 5 V relay module switching a 12 V pump
- Pressure Sensor for determining water tank level height: (https://www.adafruit.com/product/3965)

Software Architecture (Rust):
1. pigrow-core – a Hardware Abstraction Layer (HAL) exposing trait-based APIs for sensor and pump access, with a simulated back-end for offline testing.
2. pigrow-server – a binary crate running an HTTP service (axum + tokio) that hosts a dashboard at http://<pi-ip>:8080.

Functional Features:
- Web Dashboard – live moisture readings, pump status, event log, and controls for manual or automatic watering. Alerts when tanks run low on water.
- Sensor-Driven Automation – periodic sampling and threshold logic trigger the pump; a simulated decay model visualizes drying soil between cycles.
- Safety and Instrumentation – watchdog timer caps pump runtime; JSON log records all actions.

- Course Theme Fit – the evolving moisture-simulation graph embodies the Instrumented Simulation requirement.

3. Expansion on Prior Embedded Rust Coursework

I would like to do this project to expand on previous embedded work in CS410 Embedded rust programming.

4. Technical Plan

Component | Description | Libraries / Tools
Web Server | REST API + dashboard | axum, tera, serde, tokio
HAL Layer | Traits for sensor and pump control | custom + rppal
ADC Interface | Read MCP3008 SPI data | rppal::spi
Data Storage | JSON or SQLite history | serde_json, rusqlite
Simulation | Async loop + moisture decay model | tokio::time
Testing / Docs | Unit tests + rustdoc + clippy | built-in

Milestones:
One – workspace + mock HAL simulation
Two – real ADC + relay
Three – web server + REST
Four – dashboard + logging
Five – testing + docs

5. Repository and Licensing

GitHub → https://github.com/adam-marion/pi-plant-server
Contains README.md, dual MIT/Apache licenses, pigrow-core and pigrow-server crates, and proposal.pdf for submission.

6. Conclusion

PiGrow unites embedded Rust discipline with system-level engineering on the Raspberry Pi. It controls real hardware, visualizes a live simulation, and demonstrates modular HAL design, concurrency, and instrumentation. This project is an opportunity to expand on previous course work, and build something functional for watering my indoor plants and emulate commercial kits on the market such as this: https://www.amazon.com/VIVOSUN-FlexFeed-Automatic-Irrigation-Professional/dp/B0B96R2RC3 . My first degree is in chemical engineering and I work as a manufacturing engineer, so I think this project is within reason for the term.