GONZAGA UNIVERSITY
School of Engineering and Applied Science
Center for Engineering Design and Entrepreneurship

Smart Watering System with IoT

Project Overview
Plan Section 01

Release:
Draft v0.1

PROJECT PLAN DRAFT STAGE DOCUMENT
9/10/2024

Irrigators

Caleb Lefcort
Matt Nguyen
Arvand Marandi
Madison Spink

1 Project Overview

1.1 Project Summary

Water conservation is a critical challenge in agriculture. Traditional irrigation methods often rely on fixed schedules or manual intervention, which can result in inefficient water distribution. This inefficiency can easily lead to wasted resources and potentially harm the crops. By leveraging technology, there is an opportunity to ensure that crops receive the exact amount of water they need at the right time, requiring minimal human involvement. Our project aims to address the problem of water overuse and improve irrigation systems' efficiency and ease of use.

To accomplish this, we are developing a smart watering system that combines IoT sensors and MathWorks tooling to enable real-time data monitoring and analysis of crops. The IoT devices will capture key environmental conditions such as soil moisture and temperature, while ThingSpeak and MATLAB process this data. A state machine will then determine if the crops require watering. If they do, the system will send a notification, allowing the user to react using actions built into the application, which may incorporate other industry-relevant technologies in addition to MathWorks tools. This solution aims to reduce water consumption, improve crop health, and optimize growth conditions around Gonzaga.

1.2 Project Objectives

Of the desired outcomes, the primary business objective of this project is to demonstrate the utility of Mathworks' various tools in developing an IoT-enabled smart watering system. In application, this will be done using Mathworks' ThingSpeak plugin to facilitate communication between IoT devices and MATLAB to process the produced data. Conclusions will then be drawn from this data using a state machine.

In conjunction with the latter, another objective of this project is to optimize the system. Given that the project will be hosted in a gardening setting, this means our system will produce the most crops it can (sensory data being the limiting factor) while using a minimal amount of water.

1.3 Project Stakeholders

Irrigators Development Team
The Irrigators aim to gain technical and industry experience and develop project planning and management skills.

MathWorks
MathWorks, through liaison Roberto Valenti, aim to promote MathWorks' ThingSpeak and Simulink while solving industry leading problems such as efficient water usage in agriculture.

Project Advisors

The project advisor’s goals are to promote the success of the project and facilitate the learning process within the Irrigators team, as well as provide extra support in communication with the sponsor.

DAB Members
The members of the Design Advisory board are here to facilitate the design process and execution of the project.

Target Users
Our target users for this product are gardeners and small farmers aiming to optimize water usage by automating irrigation. We are focusing primarily on outdoor crops with sprinkler-based irrigation systems, what may receive more water than needed with an automatic sprinkler system that cannot respond to relevant elements like soil water retention and weather conditions when on a timer system.

1.4 Project Deliverables

Smart Watering System Application
The main deliverable is the smart watering system that uses a state machine model to manage plant irrigation. Initially, the system will use mock sensor outputs, simulating soil moisture data based on data from Gonzaga's Plant Services. The sensors will transmit this data to ThingSpeak for processing, and the system will determine an appropriate watering strategy using the state machine. The user can choose to execute the strategy based on real-time notifications. We will deliver this software as a cloud application with a user interface.

Mock Sensor Data
This deliverable will simulate the garden's soil moisture conditions in software, allowing the team to test different watering strategies without relying on physical sensors and budget.

State Machine
This deliverable will house the logic determining when and how to water the crops based on the mocked sensor data. The state machine will ultimately decide the watering strategy.

ThingSpeak Integration
This deliverable will involve setting up and configuring ThingSpeak to collect, process, and analyze environmental data from the mocked sensors. ThingSpeak will communicate with the system's state machine to determine a watering strategy based on the processed data.

Responses
The deliverable will define how the system responds to the state machine's output. For example, one possible action is to send notifications to users based on the state machine's decision for watering.

Documentation
The project will include user and developer documentation. The user documentation will detail how to set up and operate the system, while the developer manual will explain the technical architecture, state machine logic, and integration with ThingSpeak and other technologies used. The team will provide both documents in PDF format in the project repository.

1.5 Project Scope

Much of this project will be handled in-house as most of the components fall within the scope of the project. While we will be using MathWorks tools to build this project (Matlab/Simulink,) and collected data will be sent to and processed on MathWorks's ThinkSpeak we are not considering this a separate service because tools for data aggregation, analytics and visualizations will be developed in house on ThingSpeak.

The primary external sources of data will be the user, who we expect to input relevant crop and location data, and data from external sources, like weather stations for forecasts. The primary output of the smart watering system will be a command to turn on or off a sprinkler system either to the system itself or to a user operating the sprinkler system. We also plan to use ThingSpeak's data visualization tools to provide reports of the analyzed sensor data to the user.

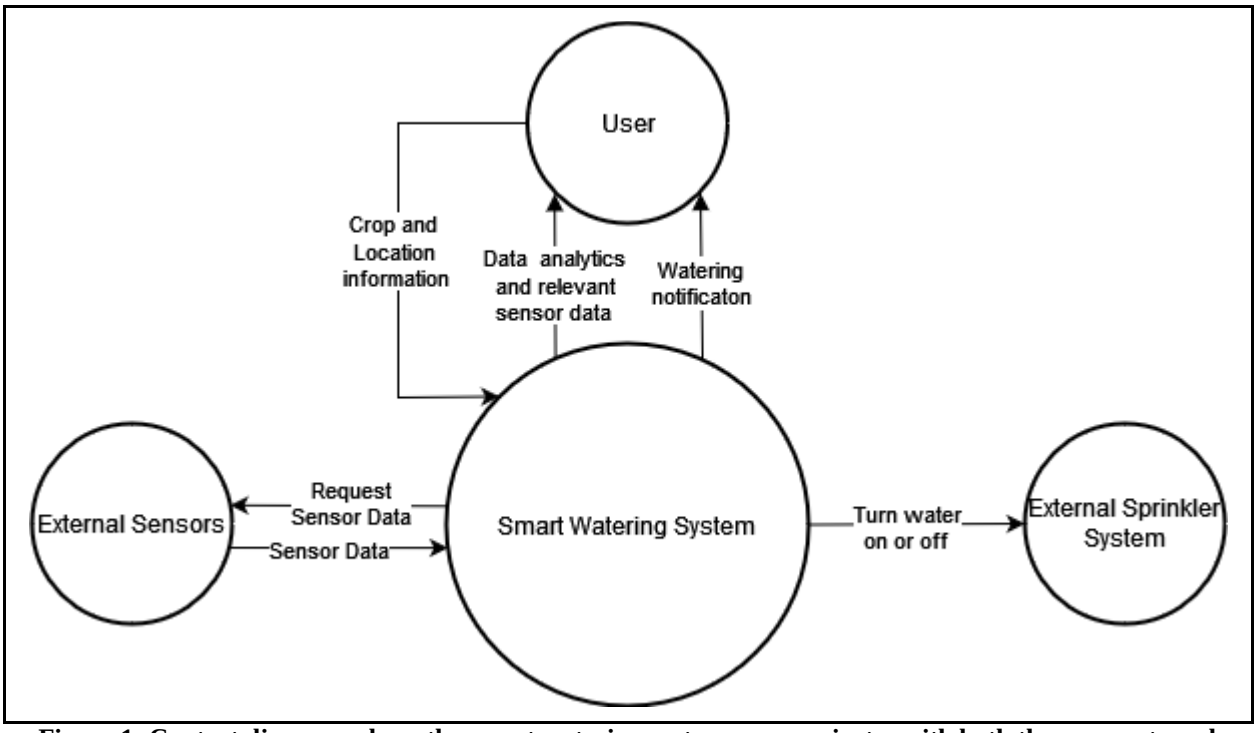

Figure 1: Context diagram where the smart watering system communicates with both the user, external sensors and external sprinkler system. The Smart watering system requests and receives data from the external sensors and the user and sends data and commands to the user and the sprinkler system.

1.6 Related Work

There have been multiple implementations of smart irrigation projects. Some of these include a smart watering system for strawberries in a greenhouse[1], a Nigerian low-cost watering system and information

relay to farmers[2], and an automated watering system for green walls[3]. Many of these projects take a similar approach to our design first sensor data is collected then sent to the central system via various channels (Wi-Fi, GSM, LoRa, MQTI). The central system then aggregates the data and decides based on the information received. Some projects also include weather data when informing their decisions. Once the decision is made the system either messages the user that the crops need watering (via Email or SMS), or the system sends a message to a pump system initiating a watering cycle.

The most similar approach to our design was conducted by a group of researchers at CVR College of Engineering[4]. They collected moisture, temperature, and humidity information. This was then sent to a NodeMCU microcontroller unit via Wi-Fi and a decision is made based on certain thresholds. The data is also sent to a mobile app for monitoring sensor data. Finally, the decision is relayed to a pump system that waters the crop. The general design of this project is similar to our own with the main difference being the use of a mobile app for relaying information. Our project will likely use a web interface instead. The other major difference is our use of the MathWorks ecosystem of tools. Much of our project will be built with these tools so the underlying technologies will be different.