# **CROPMON**

|  |  |
| --- | --- |
| Student Name | **Tadhg Ó Conghaile** |
| Student ID | **20091418** |

This project is a plant monitoring and watering system. It will use sensors to collect data and pump/valve devices. A web application will also allow a user to read soil moisture levels and trigger the delivery of water to a plant. It will also allow a user to set up rules for automatic watering based on soil moisture levels.

# **Tools, Technologies and Equipment**

The project will consist of two different device setups monitoring two separate plants. The idea behind having multiple hardware setups is an exercise in ensuring that the system is sufficiently abstracted so that it can be scaled and adapted to different devices and that all interfaces are agnostic to hardware, OS and physical variance. A specific example is by having a dive pump on one device and a gravity valve on the other device. This will mean that the water flow parameters will have to be fully configurable per device in the application. The communication between devices might be different, but the UI for controlling and reporting on the device will be the same.

## **Devices**

These are the two device setups that I will build:

**Device 1**

An Arduino board powered by an AA battery pack. This will take readings of Soil moisture, Humidity and Temperature from sensors. The board will control a dive pump submersed in a body of water. This Arduino board will transfer data using a Wi-Fi Bee (via XBee socket). Most of the hardware for this device will come from the EcoDuino Planting kit (https://www.dfrobot.com/product-641.html)

**Device 2**

A Raspberry Pi powered by socket. This device will take Soil Moisture readings from a sensor and will control a solenoid valve through GPIO. The water source will be suspended allowing gravity-watering when the valve is open. The Raspberry Pi will transfer data via WiFi using the built-in adapter.

## **System Architecture**

I am considering two different approaches for the implementation:

**Approach 1 (Node.js)**

This approach would consist of a Node.js application and front-end built using Express or React. Messaging architecture would be built using a Node.js library such as PubNub.

**Approach 2 (Azure)**

This approach would involve using Azure IOT Hub for the device messaging and a web App hosted on Azure. The stack of the web app will be Node.js.

# **Project Repository**

https://github.com/Tadhg-io/cropmon