

Need Based Sprinkler System Design

MSCS CE CSULB Spring 2015

Author: Joshua Baird 007700067

Advisor: John Tramel

**Abstract:**

This is a report documenting a sprinkler/watering system based on the need of the surrounding soil. The requirements for the system are described. This is hardware and software design of the system and the interconnected workings behind each component. A diagram of the entire system, details about the specific hardware, software diagrams, as well as a software listing are provided. Testing strategy and test results are also provided. A manual of how to use the system is included, as well as screen shots of the user interface. A conclusion with future enhancements and references can be found at the end.

Contents

Introduction: 3

Requirements: 3

Background: 4

Method: 4

System Design 4

Raspberry Pi, Model B+ 5

Capacitive Moisture Sensor by Catnip electronics 6

Output Signal 6

Wifi enabled router- or Network 6

A Desktop PC, Laptop or Cell phone 6

System Diagram Sprinkler System: 7

System Diagram in house plants System: 7

Software Design: 8

Software Diagram: 8

RESTFUL API 9

PYTHON CODE: 10

Testing Strategy: 11

Testing Results: 11

User Manual: 11

Initial Setup: 11

Modify Setup: 12

Daily Use: 12

Expand Use/Script Writing: 12

User Interface: 12

Conclusion: 18

References: 18

# Introduction:

Over the years the field of water conservation and water usage have become a greater and greater concern, as the populations grows and with recent droughts this has never been so evident. It had occurred to me that many people water their lawns with simple timers. Or even with standard mechanical systems without really knowing if there soil is in fact in need of water. There are solutions for determining at a greater level of “intelligence” if the sprinklers should turn on and off, some based even basing this on weather, moisture, and other factors. However all of these products can be very expensive and even require a homeowner or user to completely replace their current system. I wanted to take this further and explore expanding this to not just sprinkler systems but all types of water systems, with all types of sensors. This project is my solution to a flexible watering system that can be used with multiple sensors with multiple watering methods.

## Requirements:

Below is a list of initial requirements put on the design.

* The system must be responsive enough to changes in ground moisture that once the moisture level has been reach the sprinkler system is turned off within 5 seconds.
* The system must be running in conjunction with currently installed systems.
  + That is it must be easy to implement and adapt to any already operating sprinkler system.
  + Must be adaptable to be used in place where no system currently exists.
* The system must provide the user the ability to modify parameters for moisture level at each node.
* The system must be scalable for more than a single sensor. A good range is one to ten sensors. Two will be used for demonstration.
* It should not require any special training to setup and use.

# Background:

A user wishes to add moisture based sensing to his already installed sprinkler system. However he does not wish to go out and buy an entirely new sprinkler system.

With this system he could enjoy the benefits of a moisture sensing sprinkler system without the hassle of tearing out his sprinkler system and installing a whole new system in some cases.

The system can also be used for in house watering where a sprinkler system is not installed. For example a simple light or speaker can be used to notify the user that the plant requires water. This is exactly what the demo shows.

# Method:

The design allows for many methods of sprinkler systems and math methods of measurement.

This is achieved by creating a single REST API that allows different sensors and hardware to be controlled by it. The different sensors and hardware may require separate scripts, but these scripts can don’t require a recompile of the system.

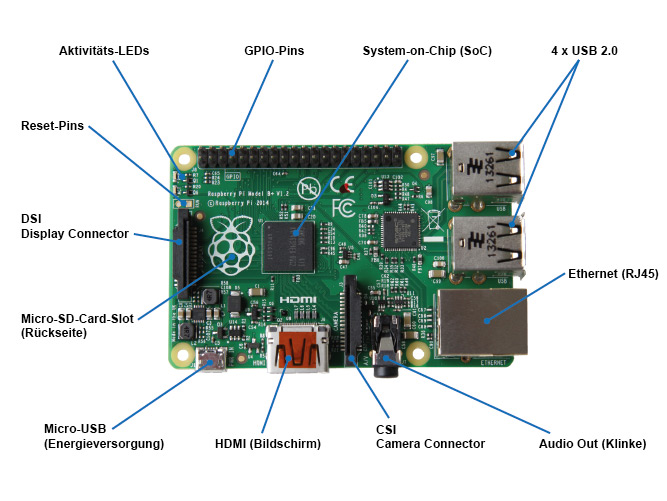
For a standard sprinkler system we can control it by simply redirecting the common grounds of the sprinkler valves to the raspberry pi and connecting moisture sensors to an I2C bus, with some setup on a web based setup page (see diagram on last page). The user can tie the moisture node to the common ground input and when a threshold is reached the common ground is bridged back to the common ground of the sprinkler system. So in an essence the raspberry pi is working as a switch with logic for when it is on and off built in.

For you indoor plans a simple LED light or other notification type device can be used that a simple low active signal can be used to drive. In fact most systems can be modified to work with a low active signal to determine when they are allowed to run.

# System Design

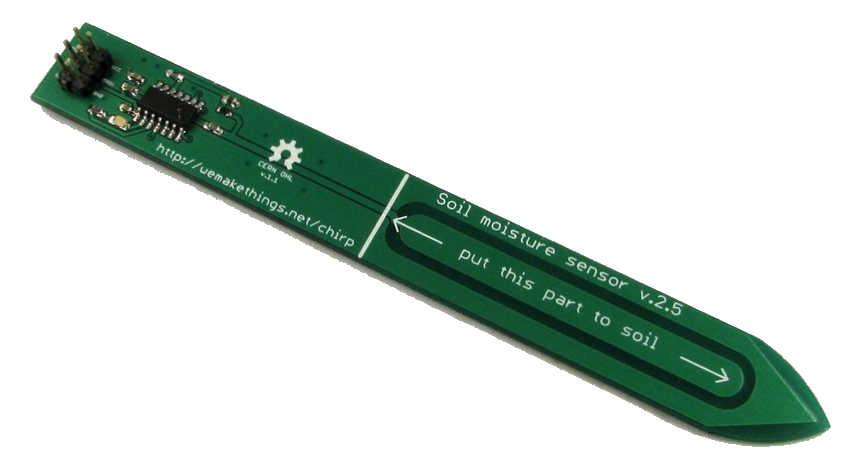
The system will contain the following main components:

## Raspberry Pi, Model B+



* + See <https://www.raspberrypi.org/products/model-b-plus/> for detailed specs
  + To act as a receiver for the moisture sensors on the I2C BUS
  + To act as a switch like controller for the common wires between the sprinkler valve and a standard sprinkler controller.
  + To act as a server to display a web page setup for the “need” of each moisture sensor.

## Capacitive Moisture Sensor by Catnip electronics



* + See <https://www.tindie.com/products/miceuz/i2c-soil-moisture-sensor/> for detailed specs
  + Senses moisture and provides the localized moisture level on a I2C BUS

## Output Signal

* + Standard Sprinkler system Or LED’s, something to display or turn on when water is needed.
  + To interface with using the raspberry pi.

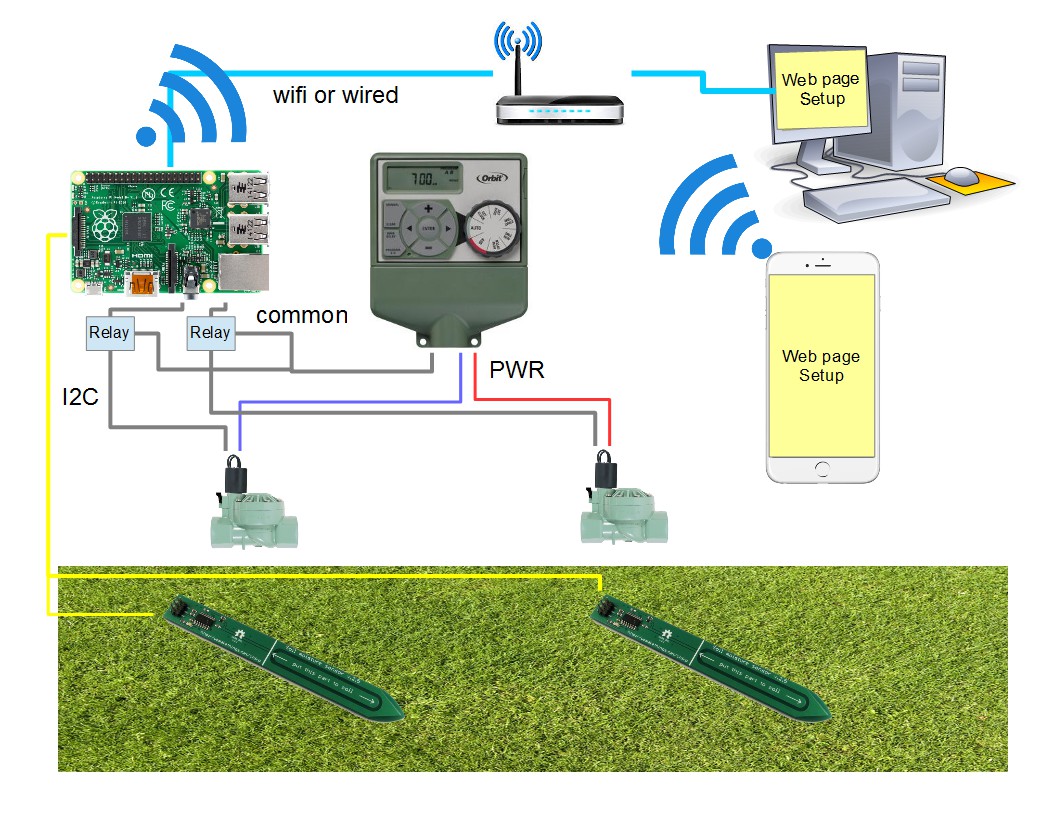
## Wifi enabled router- or Network

* + Used to connect devices to the raspberry pi’s web page to setup the levels for turning on and off an individual sprinkler

## A Desktop PC, Laptop or Cell phone

* + Capable of connecting to the network and opening a web page.

## System Diagram Sprinkler System:



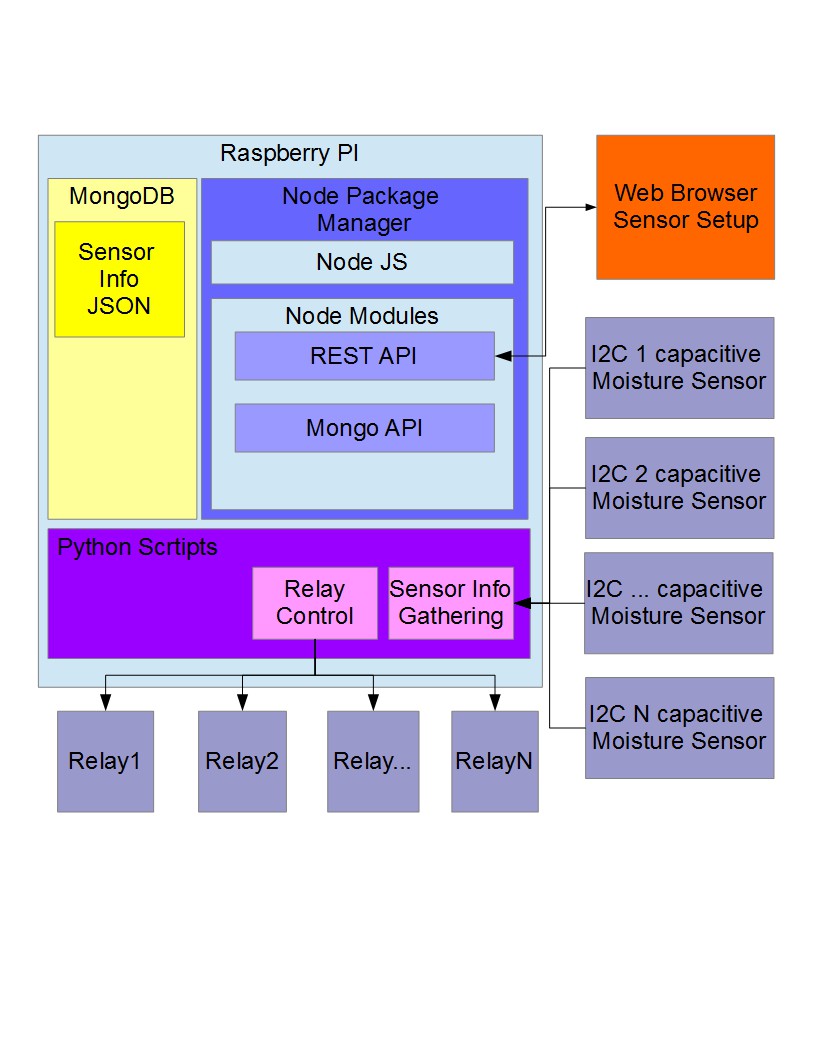
## System Diagram in house plants System:

<<TODO insert Diagram>>

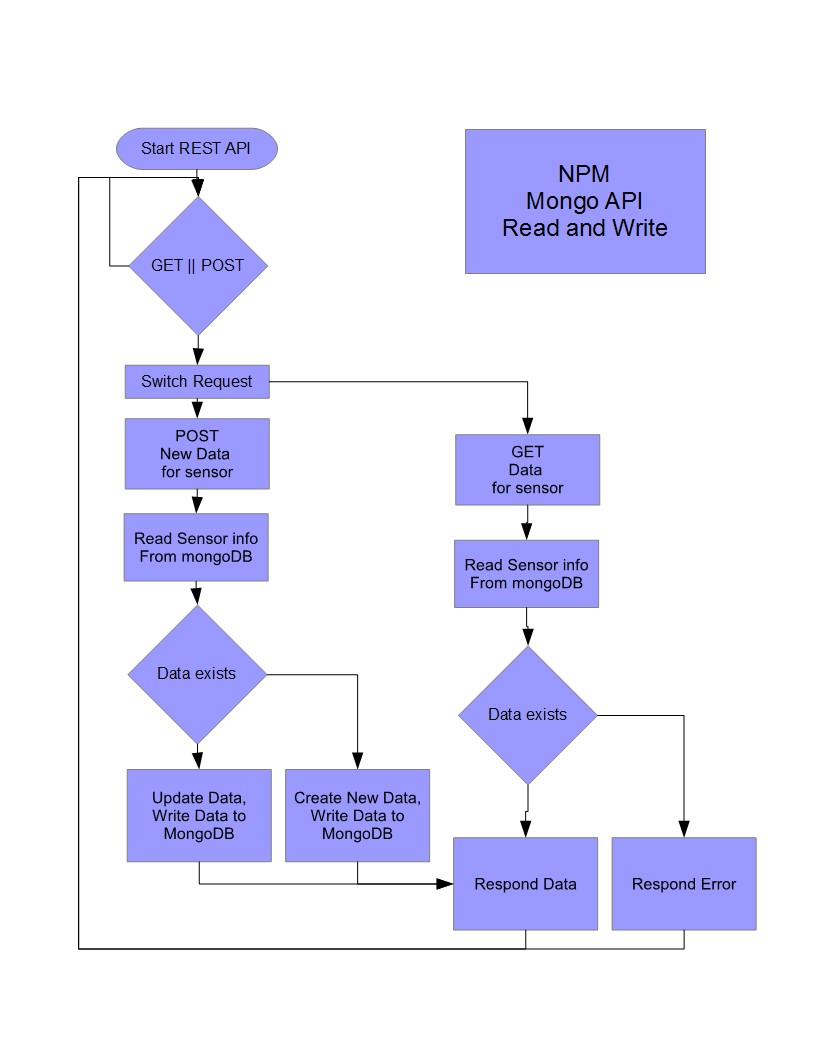
# Software Design:

All Software can be found on github: https://github.com/joshbaird/MoistureSensingSprinkler/

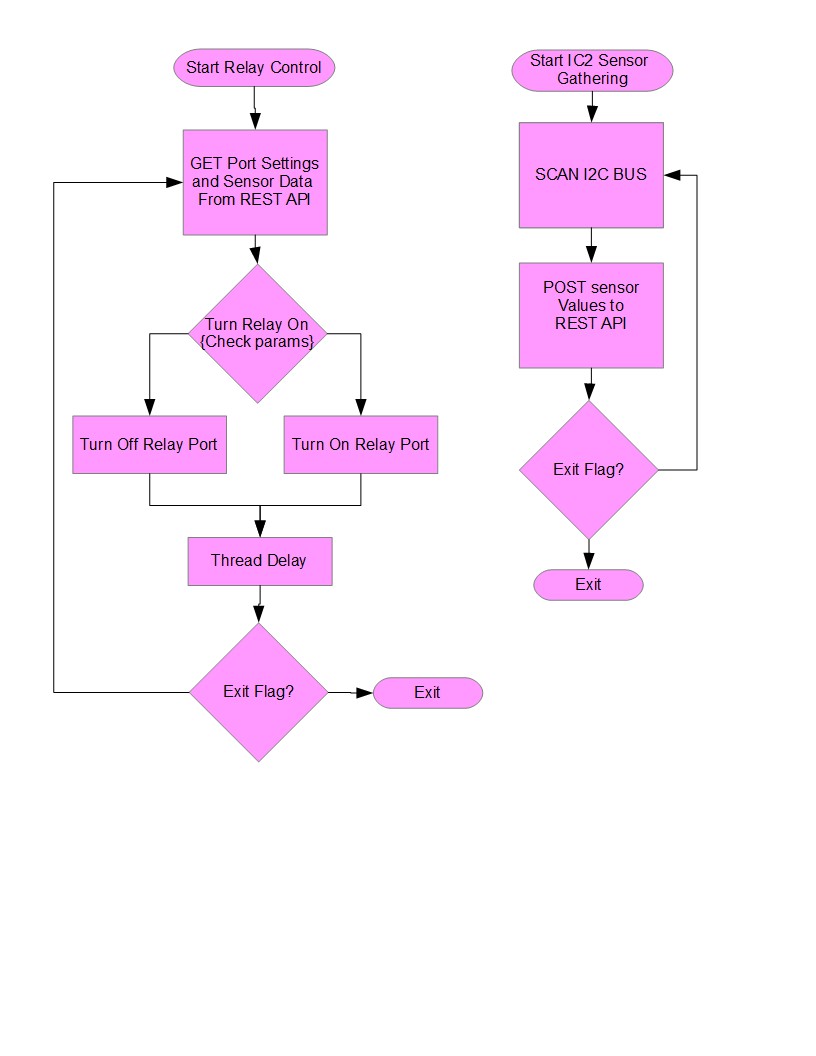
## Software Diagram:



## RESTFUL API



## PYTHON CODE:



# Testing Strategy:

The test for this product will be conducted using the system as a whole minus the sprinkler heads as they are difficult to remove and move. Instead Lights will simulate the operation of the sprinkler and a more reliable hand delivery of water will be used. The test will involve replacing the sprinkler heads with Lights that turn on when water is demanded and off when not. Simple LED lights will be used in conjunction with grounding them to the Raspberry PI. Two simple planters pot, with initially dry soil will be used to test. Two moisture sensors will be used for the demo. They will be placed in different buckets to show different readings. The test sequence is as follows:

1. Setup System.
   1. Follow system diagram below for details on connections.
   2. Actual pin numbers and connections will follow
2. Connect to system using laptop and configure sensors.
   1. Set the initial state of the moisture sensors to just above turning off the lights.
3. Pour water onto the sensor area until the light turns off.
   1. Allow the soil to soak up the water. And give it some time before pouring more.
4. Demonstrate that you can adjust the sensors on and off level in real time on the laptop.
5. Repeat with different levels of moisture for each sensor.

# Testing Results:

The system tested as expected.

<<TODO DISCUSS MORE HERE>>

# User Manual:

<<TODO INCLUDE INSTRUCTIONS>>

## Initial Setup:

<<TODO Add instructions for setting up PI and connecting>>

## Modify Setup:

<<TODO Add instructions for changing the intial settings in Config, and Adding new sensors>>

## Daily Use:

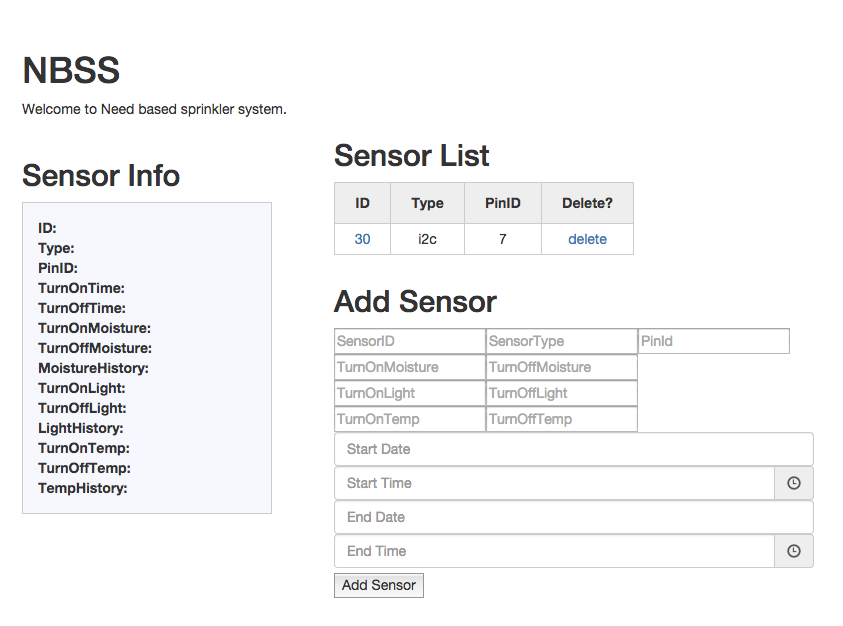
<< TODO adjusting sensors are needed>>

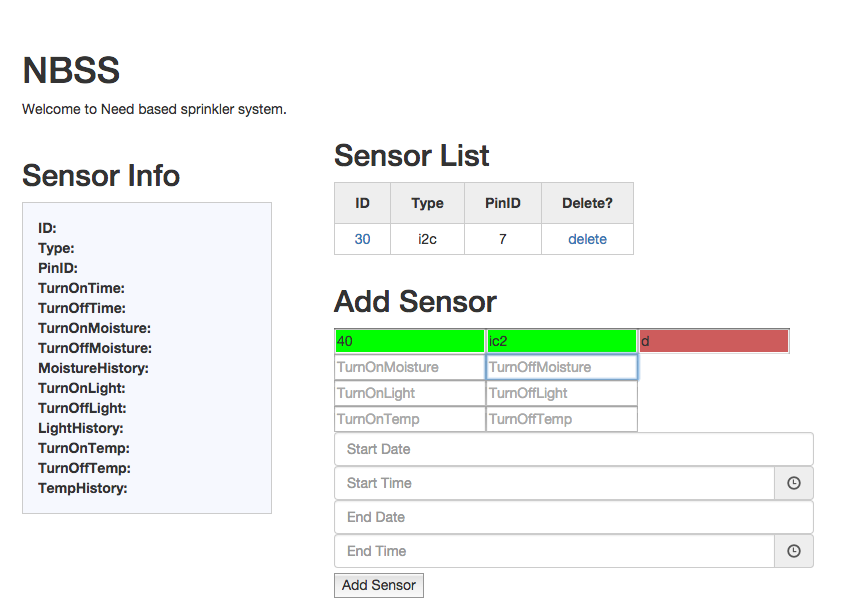
## Expand Use/Script Writing:

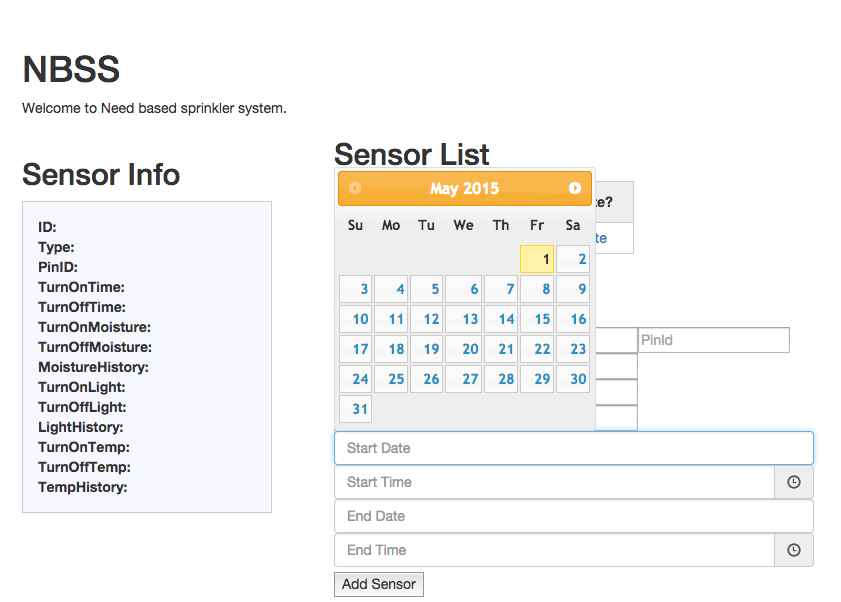
<<TODO simple guide on writing scripts, example in python, however any language that can do GETs, and POST can be used, Javascript, Python, Java, C#, C, C++, pretty much any language…>>

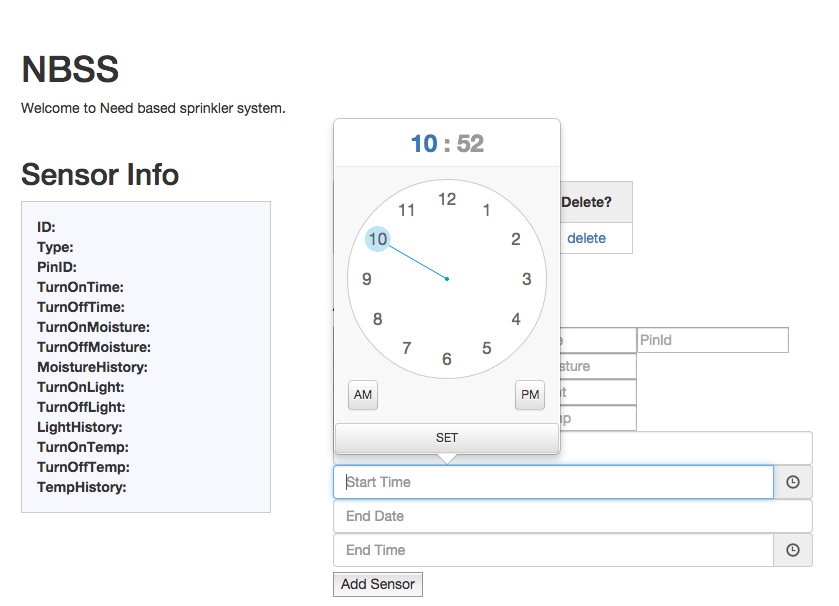
# User Interface:

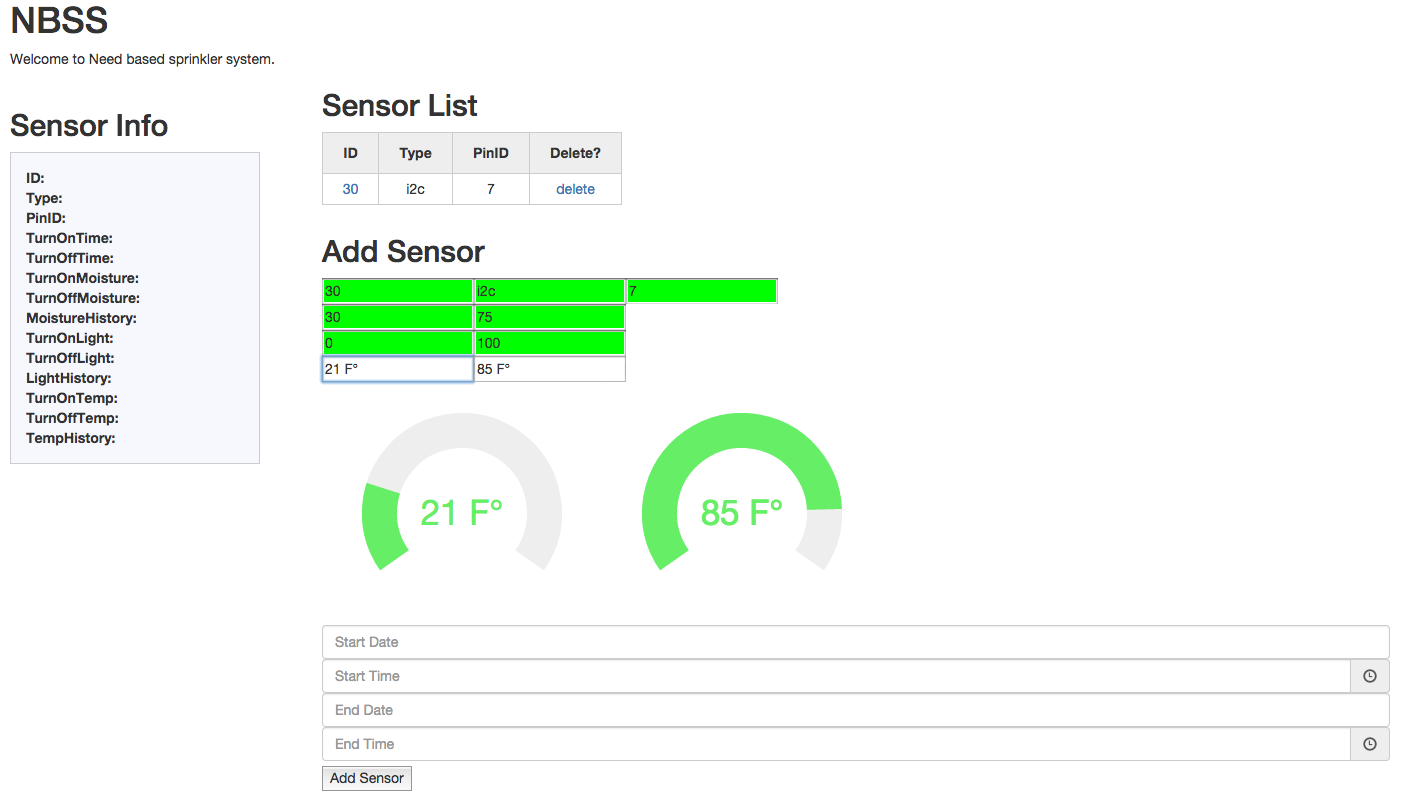
<<TODO DISCUSS IMAGES>>

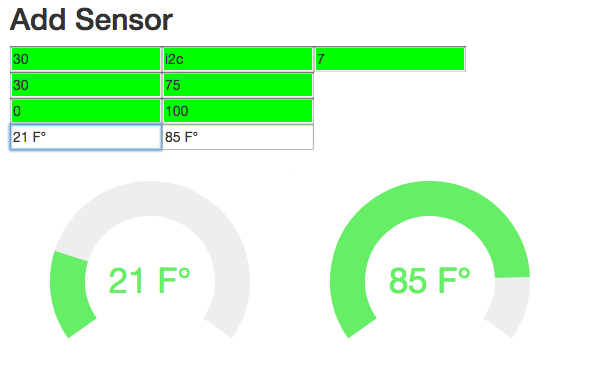


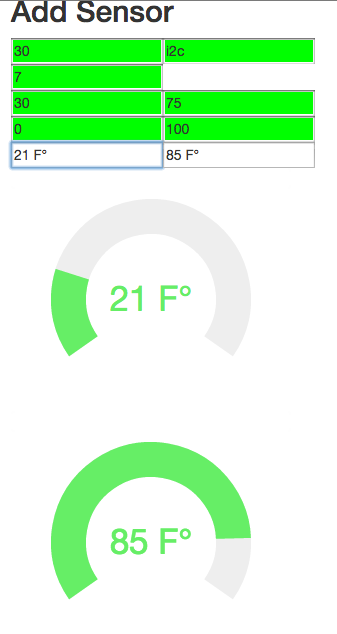


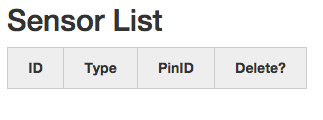
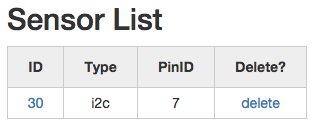


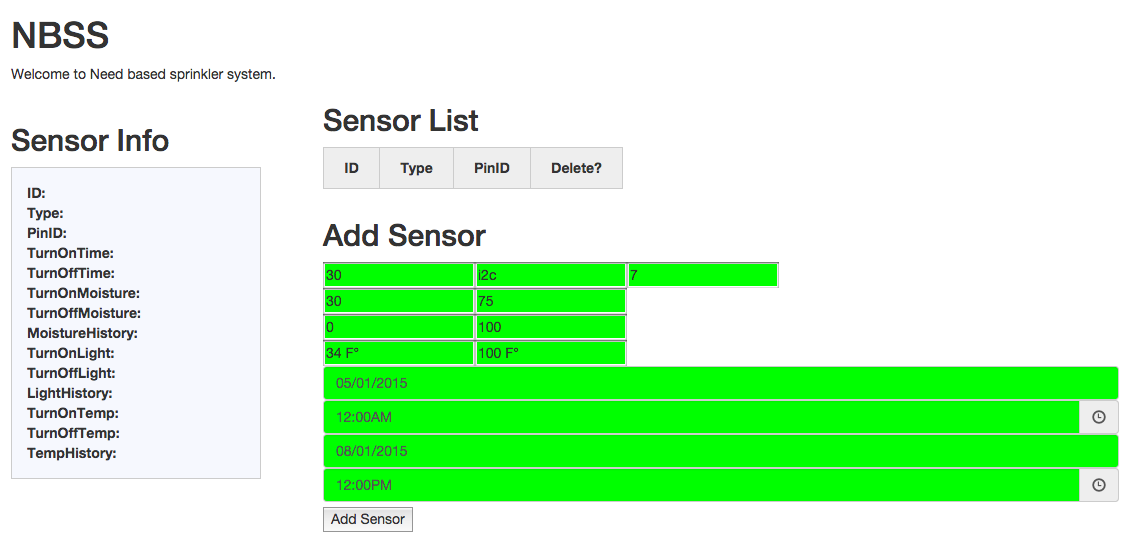


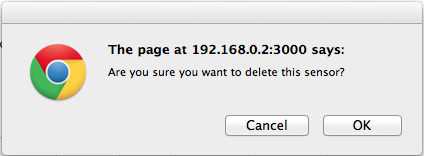


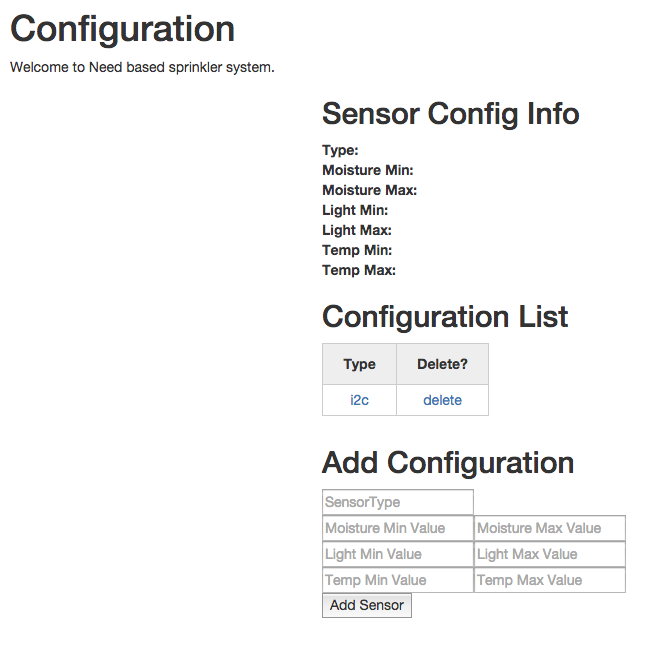


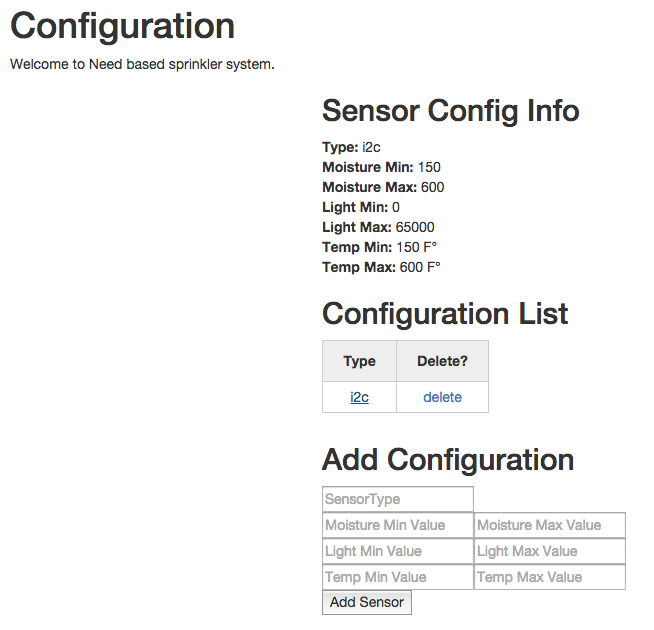












# Conclusion:

<<TODO INSERT CONCLUSIONS>>

# References:

<<TODO INSERT REFERENCES>>