EMBEDDED LINUX PROJECT PLAN
Spring 2017

AquaPi: A Self-Watering Plant System

Student name	Student Major	Grade Grade Received
Otto Vogrincic	CS	
Mario Mena	CS	
Dhanvinder Singh	CE	

PROJECT DESCRIPTION:

It is predicted that by 2020, Internet of Things (IoT) based devices will reach a staggering 50 billion objects. This averages to roughly six and a half devices per person. While IoT usually catches the interests of those in technology fields, it is clearly an influence on everyone's daily lives. Most users may not even recognize they are using a device that could be classified as IoT.

In this project, we create a self-watering plant system, AquaPi, that uses a Raspberry Pi 3 to automate the watering of plants. Through the use of sensors, we gather measurements of moisture, temperature, and humidity to activate a mini water pump. The brain of the operation is the Raspberry Pi which initiates the mechanism once the moisture sensor shows low levels of moisture in the soil. This causes the pump to activate and the plant receives the needed water. The Raspberry Pi also acts as a web server and controls a web application that can be used to access the system and the corresponding information.

Table of Contents

1. Project Goals:

In order for AquaPi to be completed, the following goals were established.

A. Configure a moisture sensor to read moisture levels from the soil in a plant pot.

B. Configure a humidity/temperature sensor to read temperature and humidity around a plant.

C. Configure a mini water pump to pump water onto plant according to soil moisture levels.

D. Bring together all sensor components through a python application run on the Raspberry Pi 3.

E. Build a web server on the Raspberry Pi 3.

F. Build a web application to on the Raspberry Pi that allows access to the AquaPi program and displays the gathered data.

G. Have a functioning system that waters plants when soil moisture levels are low, displays the data, and is accessed through a web application.

2. Project Implementation Details:

AquaPi has three main components:

A. Hardware:

Soil sensor, humidity/temperature sensor, water pump.

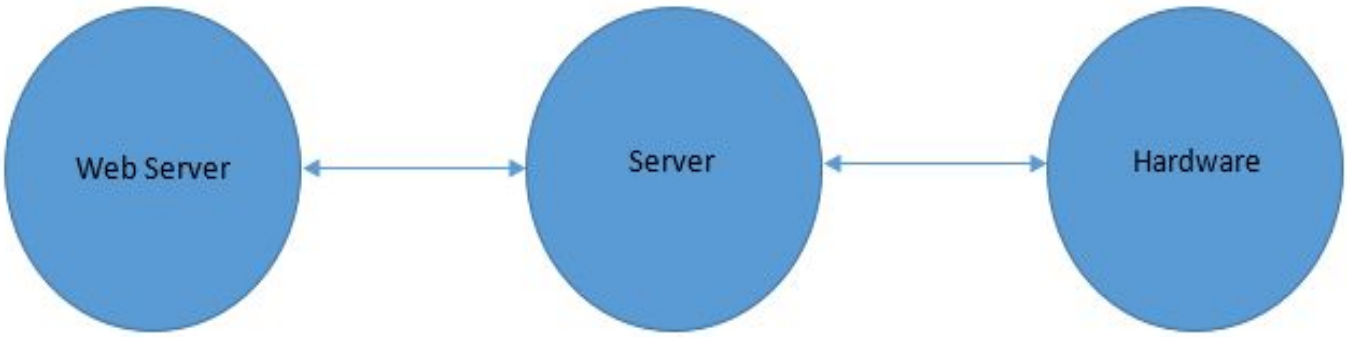
B. Web Server:

Flask microframework.

C. Physical server. Raspberry Pi 3 used as a physical server acting as the intermediary between the hardware and the web server.

All the code and detailed documentation can be found on github:

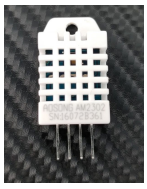
https://github.com/ovogrin/ELSpring2017



3. Project Breakdown:

There were multiple hardware components to this project that makes it work properly. The hardware components are explained in details below:

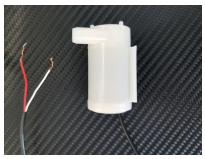
- **Temp/hum sensor:**



The DHT22 is a basic digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin. In our case, the sensor is used to

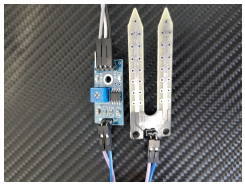
measure both the temperature and humidity around the plant. The data collected from this sensor is saved into a database.

- **Submersible water pump:**



This Mini Water Pump is submersible and takes a voltage between 6 and 12 Volts. It has a motor inside it which pumps water in through the hole at the bottom and pumps it out the spout on the side (plastic tubing can be attached to the spout to reroute the liquid). A common task such as watering plants is easily done with this water pump into the soil when the moisture levels are low.

- **Moisture sensor:**

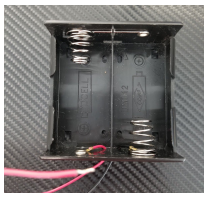


YL-69 moisture sensor. The soil moisture sensor or the hygrometer is usually used to detect the humidity of the soil. It was a perfect sensor to build an automatic watering system or to monitor the soil moisture of your plants. The sensor is set up by two pieces: the electronic board (at the left), and the probe with two pads, that detects the water content (at the right). The sensor electronic board has components such as potentiometer, power LED, and a digital output LED.

- **Wooden Box:**

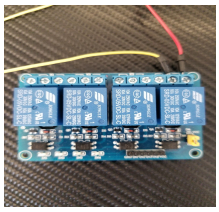
The wooden box was created from scratch to house the the hardware components, such as the relay switch, battery holder, Raspberry Pi, and most importantly the wiring for the project. To create the box, we measured each component and saw how much room they took. The width of the box is 6½ inches, the height is 3½ inches and the length is 5 inches. The two 6½ by 3½ inch pieces were connected together with screws to create a rectangle. Then we added the bottom to the rectangle with lid on the top with a hinge to open the box. To have the an easy access to the to wires we made 2 inch and 1 inch holes on the side.

- **Battery Holder:**



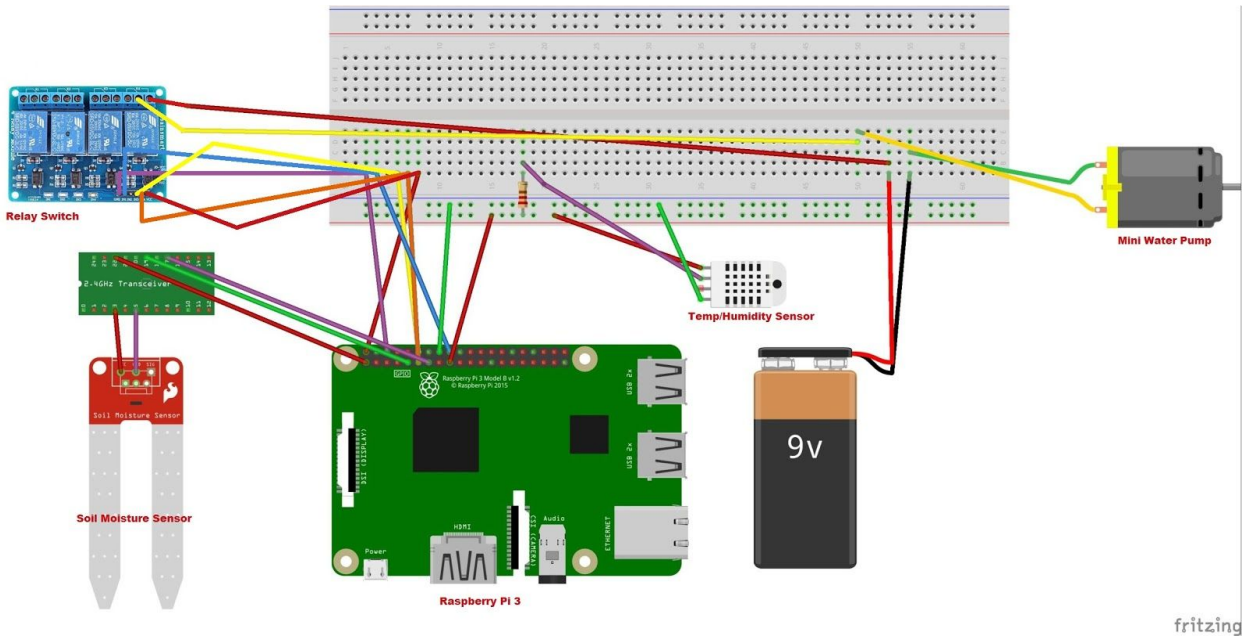
D batteries are used to power the water pump. This wire connector battery holder holds two "D" batteries and has color-coded wire leads. The two colored wires are hot and neutral, hot being red and black being neutral.

- **Relay Switch:**



A 5 volt four channel relay switch was used as a power interrupter for the mini water pump. A relay is an electromagnetic switch operated by a relatively small electric current that can turn on or off a much larger electric current. The main component to the relay is an electromagnet, which is a coil of wire that becomes a temporary magnet when electricity flows through. Although a four channel relay was used for this project, only one channel was effectively used. A one-channel relay would suffice for this project.

Wiring Diagram



Project code can be found at:

https://github.com/N02959575/ELSpring2017/tree/master/waterServer

4. Challenges:

There were a series of challenges faced during the project. The first challenge was to get used to the project environment. Python is the programming language most commonly used with Raspberry Pi, but none of us were familiar with it so we had to learned it by following tutorials and trial and error. We found out that Python programing could be used to program different devices just like C programing language. We also learned that Python source code has a very specific structure using indentation.

Another challenging part of the project was the moisture sensor. We had trouble calibrating the sensor and having it give accurate readings. We realized that if the sensor was not fully submerged, it would give conflicting readings.

For the water pump we adapted a python script that was used to turn on an LED. At the beginning this did not work. Then, we realized that the Raspberry Pi did not have enough voltage to turn the pump on. A relay switch was used to control external power gathered from double D batteries.

Using the DHT22 sensor was an easy part of the project due to the ample amount of code available online already for it. Both adafruit and pigpio have libraries for it. There was also detailed explanation on how to hook it up using wires, resistor, and breadboard. This sensor is widely available compared to the the water pump and the moisture sensor, which made it easy to add the web interfacing. Once the code for logging the data on a database was done, we were pretty much done worrying about this sensor. All we had to do next was add the information it was storing into our website.

Just like with python we were all unfamiliar with web servers. So we took a flask tutorial to create a server and webpages used in our project. The tutorial was fairly simple and straightforward to follow, what presented a challenge was to create our own html files that would display the information we wanted. Adding a table showing all the information from the sensor took a while to figure out, but eventually we were able to added to the website. However later on we decided to make the webpage prettier. This presented a problem because we would have to change the way we configured the web page for the table. In addition to these new changes we also decided we wanted to add a chart to the website using google charts. So that was yet another

new thing we had to learn how to do. Eventually we managed to add the table back into the website using a little bit of a work around on the code. After we managed to display the table we tackled the chart. We started by adding the information from the table to the chart and seeing if it would display right on a separate web page. Once we were able to fill up the information for the chart and it displayed it, we started working on adding it to its respective tab on the website. The solution for this was very similar to the solution for displaying the table.

5. Project Stakeholder's and Communication Plan

Table 1. Team Member Roles and Responsibilities.

Team Member	Role	Responsibility
Otto	Water Pump	Water pump code, integration of the pump code into the moisture sensor code, web server, web page development.
Dhanvinder	Moisture Sensor, box/container.	Moisture sensor code, helped with the integration of the pump code into the moisture sensor code. Build the wooden box to house the system.
Mario	Temp/Hum	Temp/hum code, created a database to store the readings of the DHT22 sensor, helped with the integration of the pump code into the moisture sensor code. Added the table and chart to the webpage.

All Together	Web Server	Work on web app, The Wooden Box, Written Report .

Each member had separate responsibilities which are outlined in table 1. However, each member was not limited to development in the area stated by the table. For this project to deliver a working product, each component relies on the functionality of the rest. Each team member had ensured that one piece does not excel past the others in terms of development. Resources were allocated on a weekly basis when progression is measured at class meetings. The stakeholders in the project or members not a part of the development team that have invested interest in the outcome of the project are our advising team, Professors Easwaran.

LESSONS LEARNED:

There were many lessons learned with project for as planning the project goes. Project plans such as this paper have been developed and timelines have been established. Resource allocation and weekly meetups with just the group members and with the professor were essential to the success of this project. As a group, we learned how to manage stress and divide up the work among three of us. We also learned to communicate our problems to each other efficiently and eloquently so that they could be solved.

For the server, we learned how to use a terminal based language, which was Linux. We learned how to work with Python, how to set up a simple web server, and how to make hardware work through networks and code. The most important lesson we learned, was the ability to teach ourselves.

REFERENCES/RESOURCES

Chirakkal Easwaran, Embedded Linux Course, Coursework Notes, Diagrams, and Guides.

Flask Tutorial - Calico Jake, Youtube.

https://www.youtube.com/playlist?list=PL0DA14EB3618A3507

DHT22 Sensor Tutorial and Logger Guide

http://www.instructables.com/id/Raspberry-PI-and-DHT22-temperature-and-humidity-lo/

Relay Switch Connection Guide

https://www.raspberry-solutions.com/connect-relay-to-raspberry-pi/