PoolCon 1, Version 1, Apr 15, 2021

Design Document

Embedded Systems Design

Executive Summary

The objective of this document is to demonstrate that the PoolCon1 Team designed and created a product that provides solutions to all the requirements and constraints set forth for the Pool Controller Project. This document will outline the design process taken by the team that led to the final design. It will take into account the requirements, the available materials, the selected materials, and the implementation of the chosen materials that created the team's end product.

The final product is functional as outlined in the requirements. There are a number of sensors on the hardware side that all have different uses, and quite a few features the user can utilize on the software side. These were all designed with the express purpose of making the product more user-friendly, and to enhance user experience.

Not only will this document retrace the selection and design processes - it will also include proof that the system works. Toward the end of the document, there is a section outlining how the team implemented their design, and there is a section dedicated to testing and validation. Afterwards, there are some recommendations the team would give on improving the design. As a comprehensive document, this will give a good understanding of the team's process and ability to create a viable product.

This document will be accompanied by a user and installation guide that will assist the user in understanding and optimizing the use of their system. It will also help them install the system where they deem fit. While this document is more about the team and their process and proof of concept, the guide is more about the product and its application once it is in the user's home. The recommendations at the bottom of this document would be of use if and when another team or group would want to optimize an already existing project. This wouldn't be necessary but will allow for continuity past this team's creation, concept, prototypes, and project.

Table of Contents

Introduction

The pool controller designed by this team will be explained in depth. The team wanted to create a system that would be simple to use and implement that would be intuitive to the user. The user interface can be reached on any phone or computer and automatically adjusts to the size of the screen. The UI also has a light and dark mode for increased visibility in an array of environments. Environmental factors were taken into account for the requirements that dictated the trajectory of the end product.
Now, for more about the system itself. The system is composed of a Raspberry Pi that is being used for the computer in tandem with a GrovePi+. The sensors and relays attach directly to the GrovePi+, and communicate to the user through the user display, which is hosted on a Pi that is attached to an LCD. Specifications will be given more in-depth later in the document.
In the following pages, pictures will depict how the system operates, and diagrams will dictate how the internal mechanics fit together to create the Pool Control System. A User/Installation Guide will accompany this document in helping the end user understand the product and install it for use.

Requirements

Listed below are the requirements deemed necessary by both the customer and the team. The requirements are non-negotiable aspects that need to be met in order for the entire project to be deemed successful. In addition to the requirements, there are also environmental considerations that need to be taken into account, as well as agencies whose approval needs to be gained and additional resources needed to make an informed decision.

Requirements

1. The system shall be able to turn the filter on & off
    1. The system shall allow the user to manually perform this function
    2. The system shall allow the user to automatically perform this function
2. The system shall be able to turn the heater on & off
    1. The system shall allow the user to manually perform this function
    2. The system shall allow the user to automatically perform this function
3. The system shall control the water level of the pool
    1. The system shall track the water level
    2. The system shall turn water inlet on & off
4. The system shall control the water chemistry & pH
    1. The system shall track the water chemistry & pH
5. The system shall track the filter usage
6. The system shall track the pool water and ambient temperature
7. The system shall display the current time and date to the user
8. The system shall provide a web interface
    1. The system shall have all the system function available in the web interface
9. The system shall be powered by the typical home outlet (120V 60Hz)
10. The system shall cost no more than $200
11. The system shall be completed by the end of the spring semester.
12. The system shall adhere to all known standards regarding residential pools

Environmental Considerations:

1. Product must be able to account for changing pH due to rainfall and other forms of precipitation
2. Product must be able to account for varying water levels due to rainfall and other forms of precipitation
3. Product must be sturdy enough to handle inclement weather such as windstorms
4. Product must allow for varying cleaning cycles due to extra plant matter blown into the pool by strong winds and/or rains
5. Product must be environmentally friendly; e.g., must limit waste production and be savvy of external environmental matters
6. Product must account for small animals that may find their way into the pool

Agency Approvals

1. MAHC standards
2. IEEE - National Electrical Safety Code
3. NFPA - Swimming Pools, Hot Tubs and Spas
4. CDC - Center for Disease Control

Processor Selection

For the processor, the team chose from the three options shown below:

Computer	Raspberry Pi 4 Computer Model B 2GB V1.2	BananaPi - M5 4GB	Arduino Nano IoT

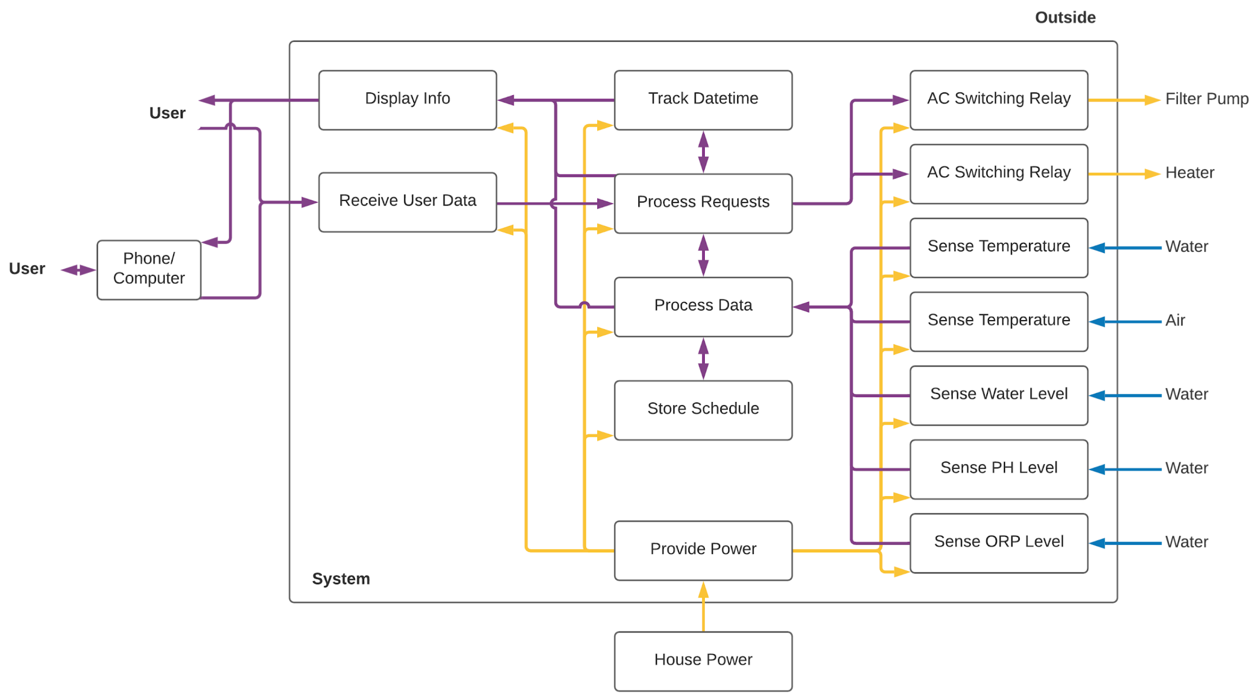
The processor used for the project is the Raspberry Pi, due to it's open source information, variety of models for different applications, and processor specifications. The other two processors are good options, but didn't fit our needs.

Architecture Design

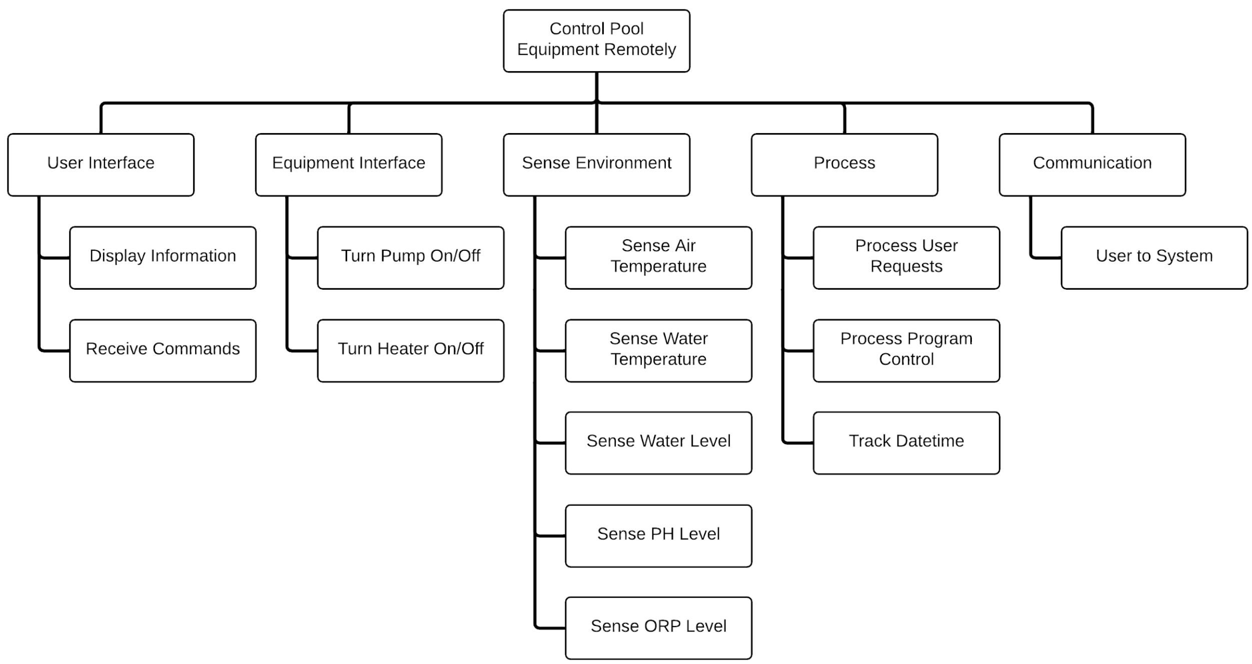
Concept

- Single CPU hosts web service and handles IO
- Physical touchscreen on head unit to interface with web service
- Web service can be accessed remotely by mobile phone or computer

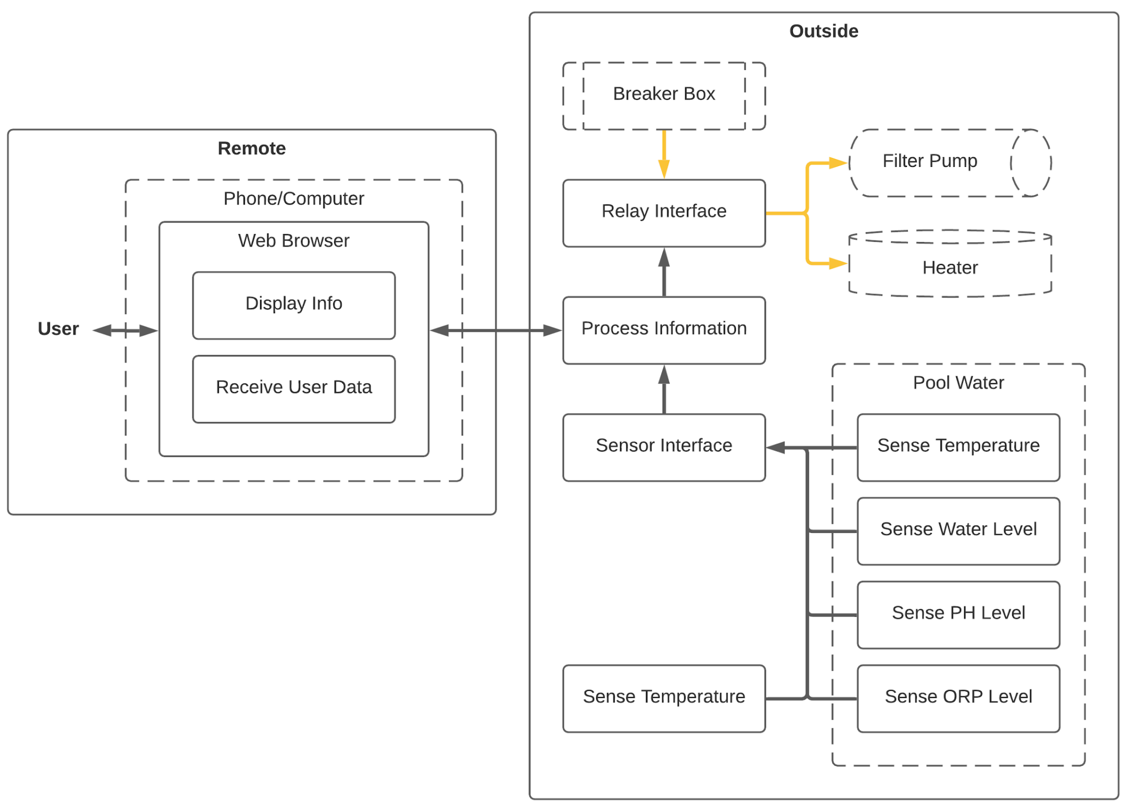
FFBD



Hierarchical



Physical

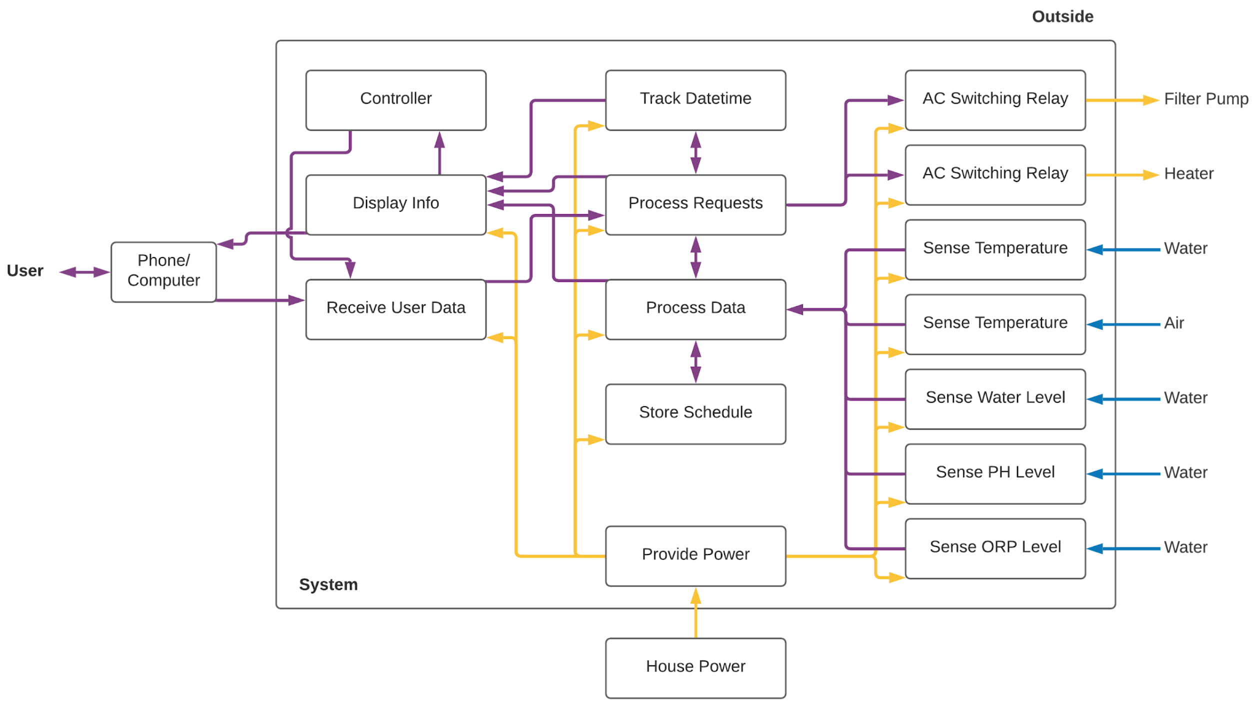


Hardware Design

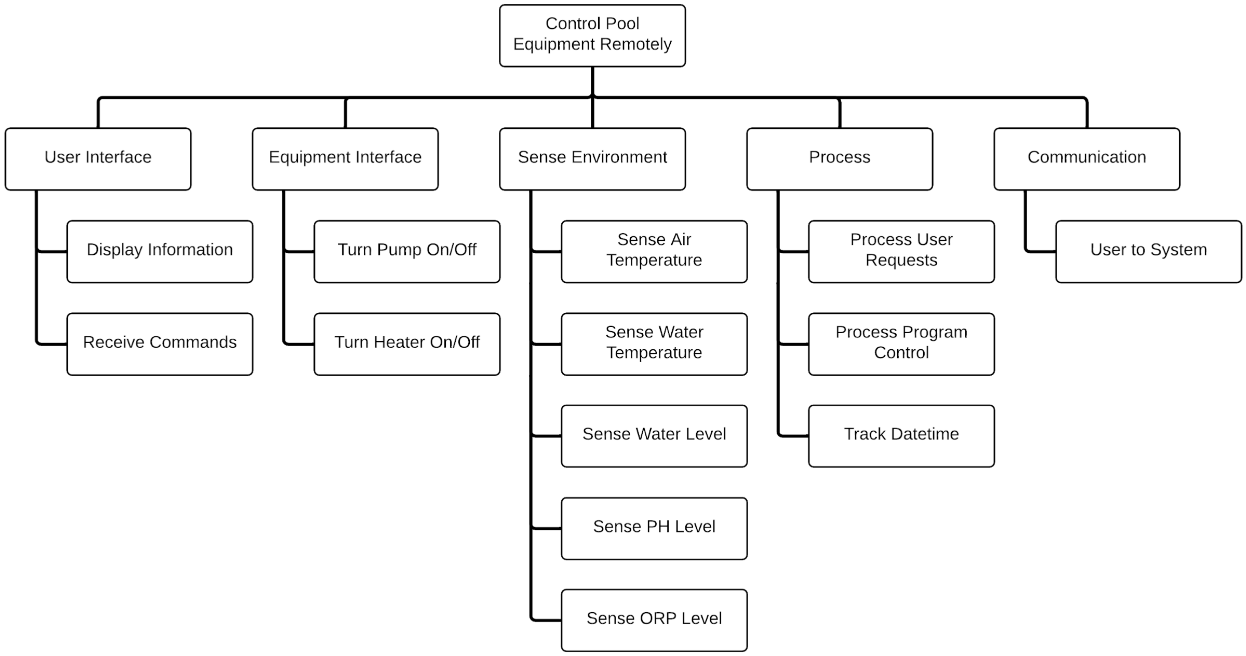
Concept

- CPU hosts web service and handles IO outside
- CPU with touchscreen interface with web service inside
- Web service can be accessed remotely by mobile phone or computer

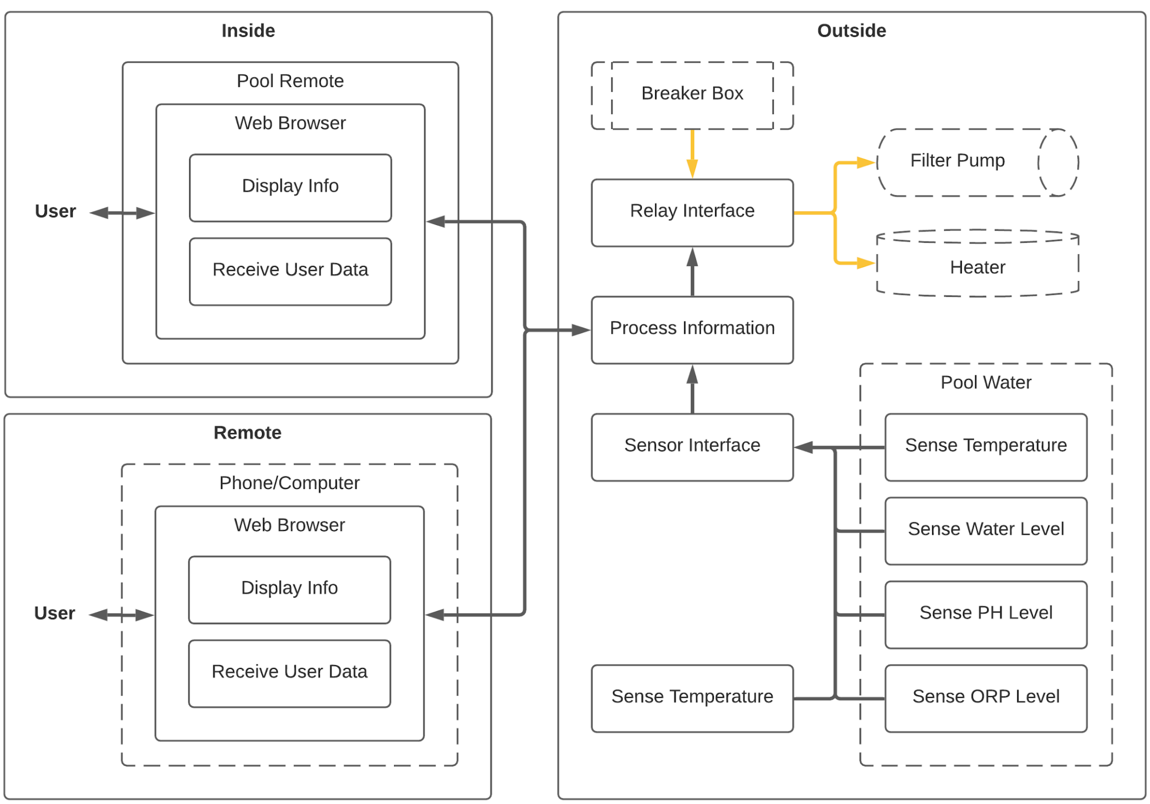
FFBD



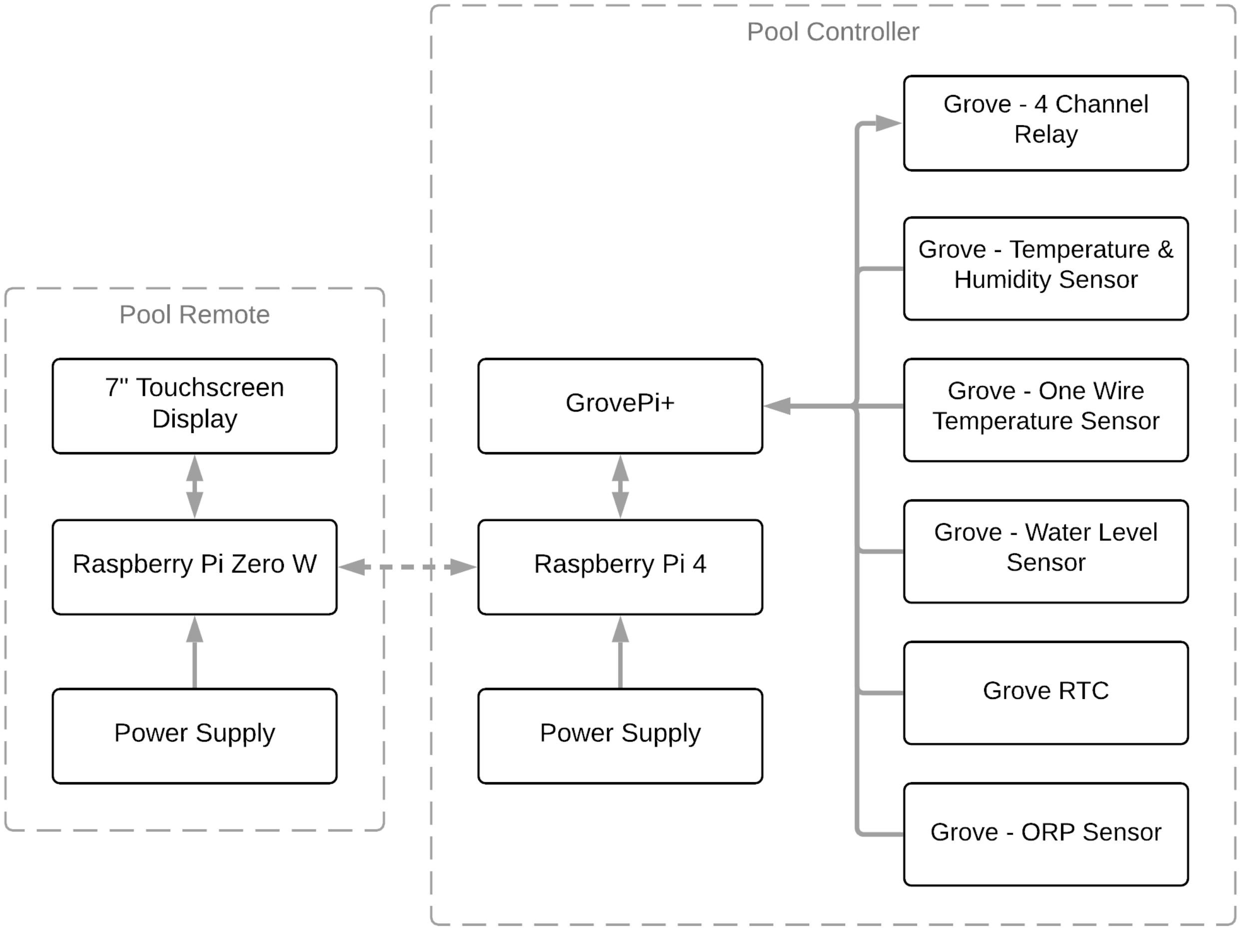
Hierarchical



Wireless Connection Plan



Connection Details



Component Details

Hardware Function	Part
Filter & Heater Relay	Grove - Relay
Air Temperature Sensor	Grove - Temperature & Humidity Sensor (DHT11)
Water Temperature Sensor	One Wire Temperature Sensor - DS18B20
Water Level Sensor	Grove - Water Level Sensor
PH Level Sensor	Grove - PH Sensor Kit (E-201C-Blue)
Real-Time Clock	Grove RTC DS1307
Sensor Interface	GrovePi+
Computer	Raspberry Pi 4 Computer Model B 2GB V1.2
Controller microSD Card	micro SD Card with Card Reader-32GB(Class 10)
Computer Power Supply	Official Raspberry Pi Power Supply 5.1V 3A with USB C
Controller Display	5-Inch 800x480 DSI Interface LCD Capacitive Touchscreen
Controller Computer	Raspberry Pi Zero
Controller Power Supply	Micro USB Cable - 48cm
Controller microSD Card	micro SD Card with Card Reader-32GB(Class 10)
Chemistry Sensor	Grove - ORP Sensor Kit (501Z)

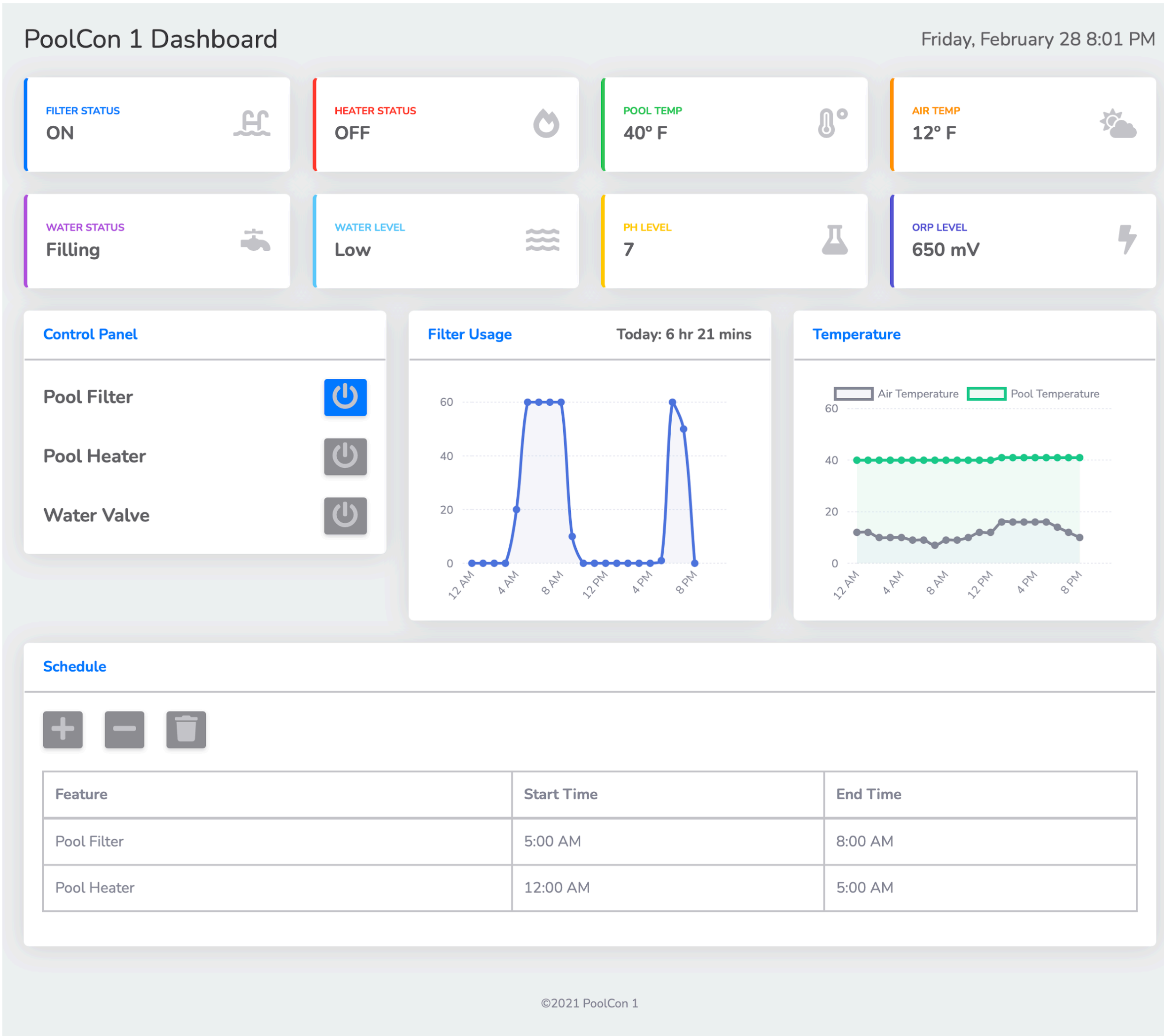
Information Details

The MCU being utilized is a 328p, and is being controlled by our raspberry pi via a controller computer. The information will be displayed in the ways outlined below in the interface section, and the user input method will be via buttons on said display screen. The temperature sensors will be either submerged in water or available to the outside air and will process and report the temperatures real time where they will be displayed to the user.

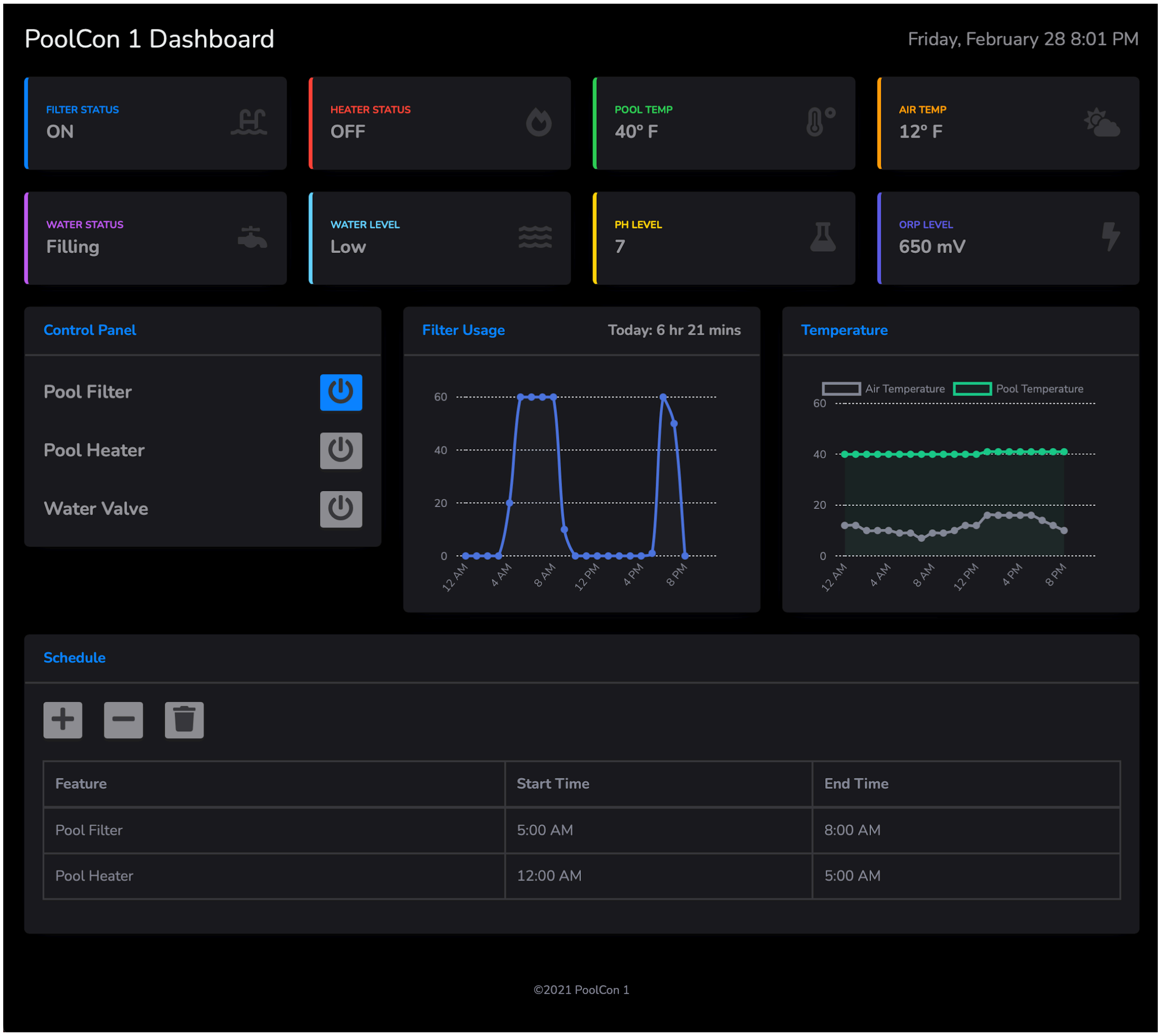
Interface

The unit on the interior as well as the mobile interface will share the same web service, as it can adapt to screen size and light/dark mode for user convenience. The mobile interface can be conducted on a mobile phone or laptop. The inside unit will be an LCD that can either be mounted on a wall or propped on a surface such as a table or nightstand, depending on the user's preference. Below is the interface that our team has designed, in the light mode on an LCD. The web service updates in real time and shows updates between each interface simultaneously. Tasks can be scheduled to control the sensors, and sensor data updates on the UI, while trends are tracked.

Light Mode Dashboard:



Dark Mode Desktop:



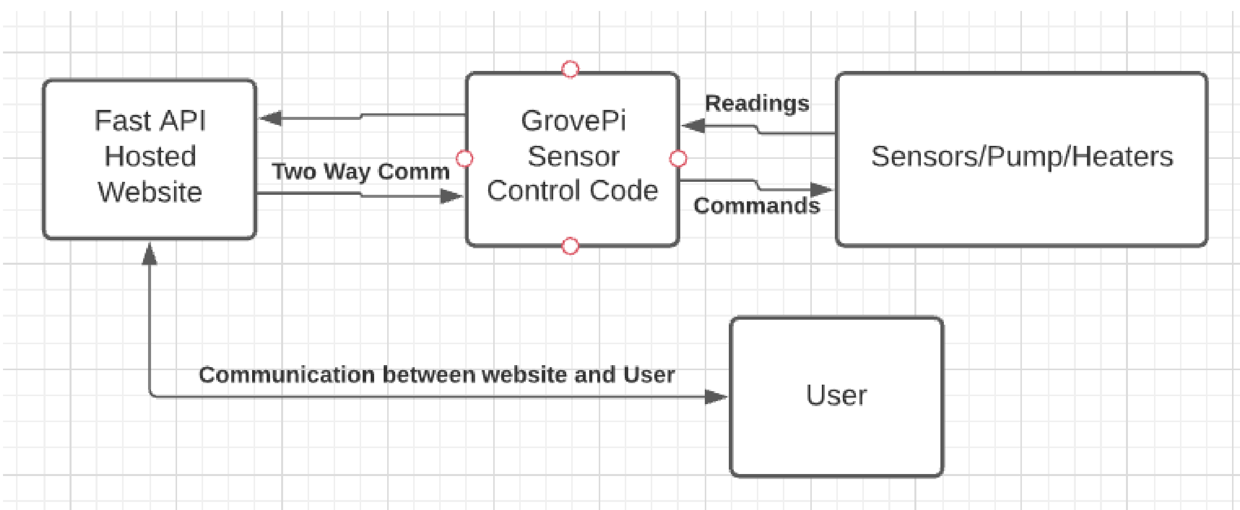
Software Design

Software Description

The software will boot up a webservice via a FastAPI server. Once the server is up and running it will first take real time polls of the sensors connected to the pi. It will take these inputs and upload the information to the server to be displayed to the user via the website. From there the software will take the previously mentioned real time inputs and manual inputs from the user, including preset schedules, to send appropriate signals to the filter, heater (both relays), PH control, etc.

The software so far is designed to switch three relays on and off, with real-time interaction with the user's choice of an LCD interactive display, or a remote control that is accessed via phone or laptop. It is able to do this by establishing wireless communication between the remote and the pi (which the sensors are connected to). The software also reads and displays the water and air temperatures.

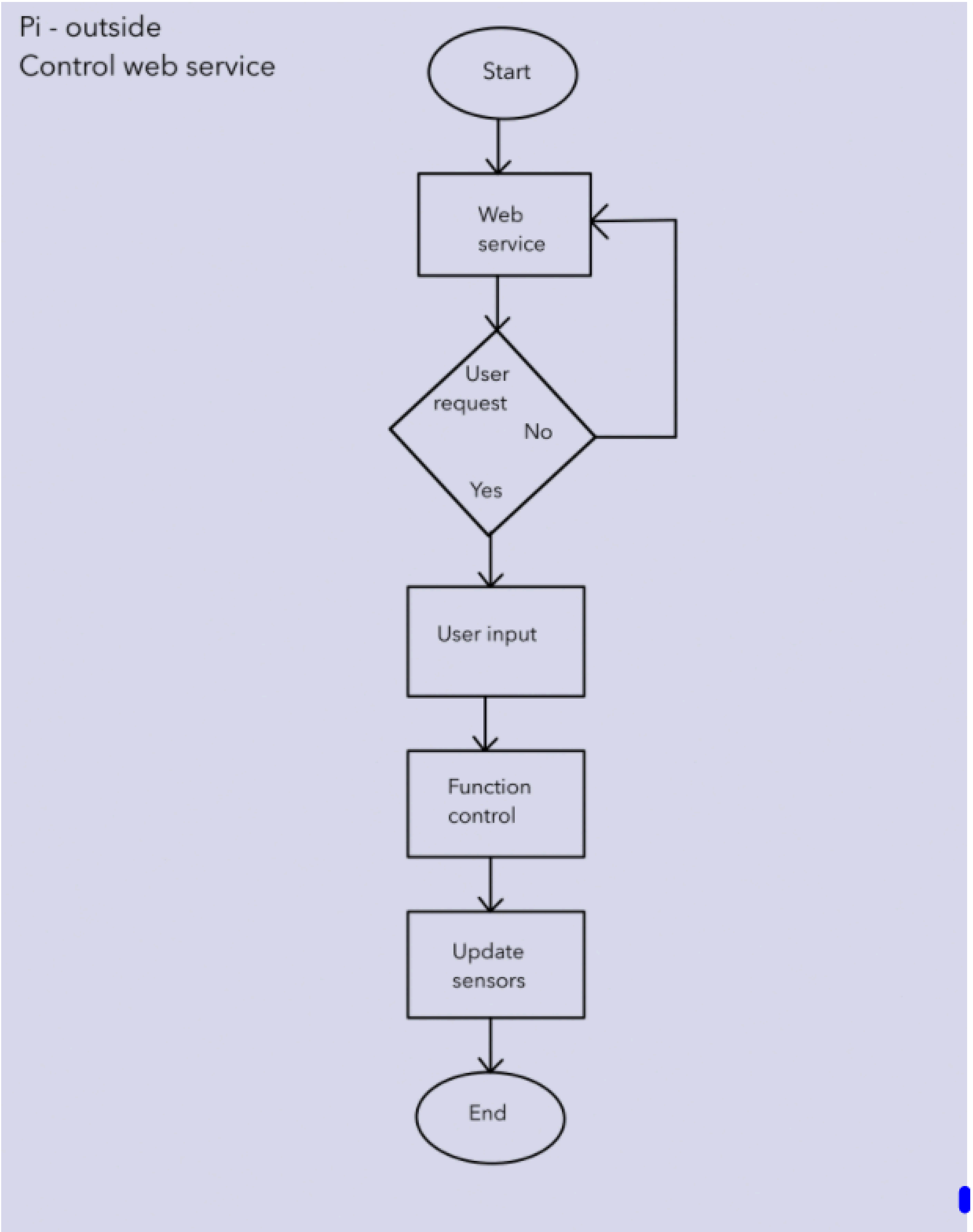
Software Architecture




```
import dependencies
initialize GPIO
define system globals
start web_service app
establish db connection
start apscheduler

define Web service Endpoints
```

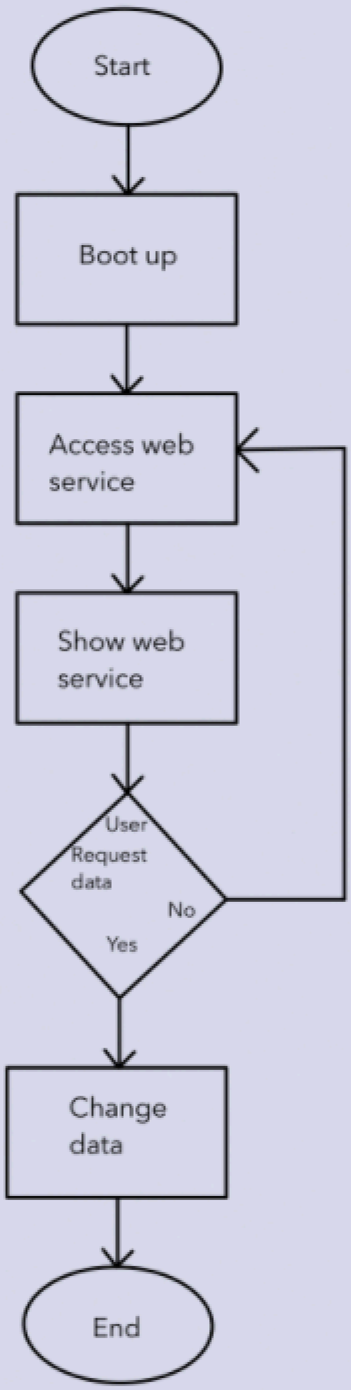
```
  GET home -> dashboard.html

  POST toggle_filter -> toggle_filter_relay
  POST toggle_heater -> toggle_heater_relay
  POST toggle_water -> toggle_water_relay

  POST add_event -> schedule_event(form_data)
  POST remove_event -> remove_scheduled_event(form_data)
  POST delete_event -> delete_all_events

  web socket -> update_sensor_status()

def toggle_filter_relay():
  log_time()
  toggle_relay()

def update_sensor_status(): -> JSON data
  get_time()
  get_water_temperature()
  get_air_temperature()
  get_water_level()
  get_ORP_level()
  get_PH_level()

  return JSON data

def schedule_event(form_data):
  apscheduler.add_job(form_data)

def remove_scheduled_event(form_data):
  apscheduler.remove_job(form_data)

def delete_all_events():
  for job in schedule:
    apscheduler.remove_job()
```

Software System Flow Charts



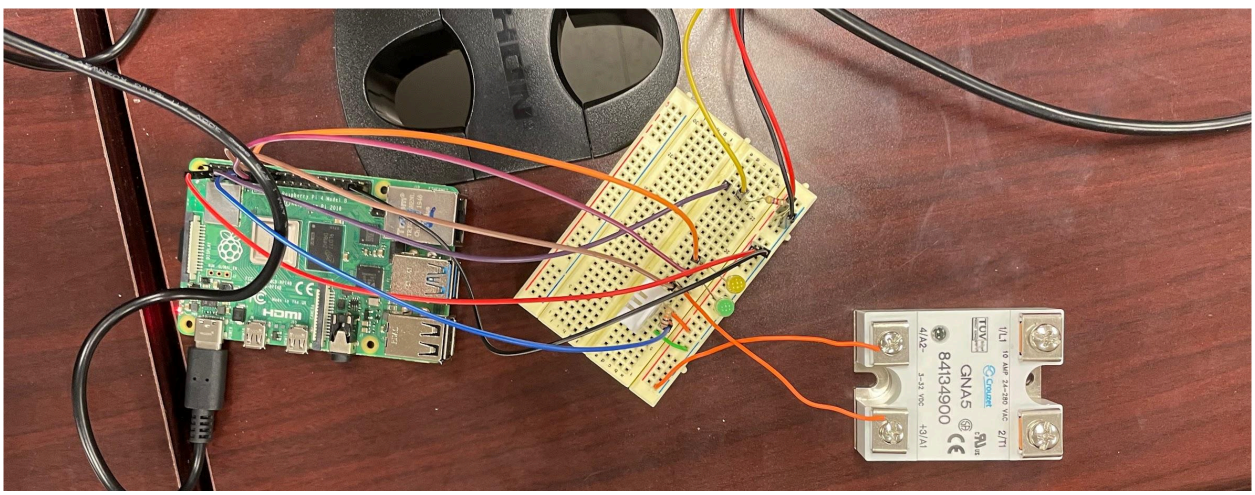
Pi - Touchscreen controller

Access web service

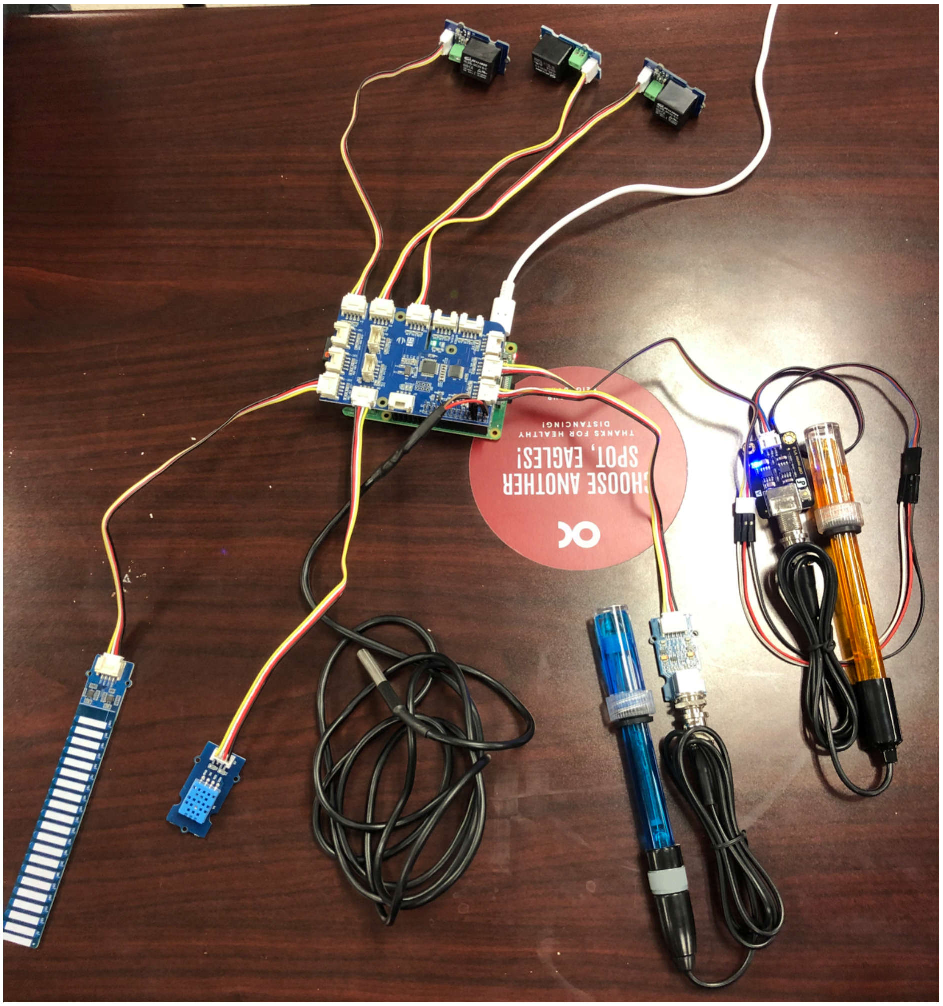


Design Implementation

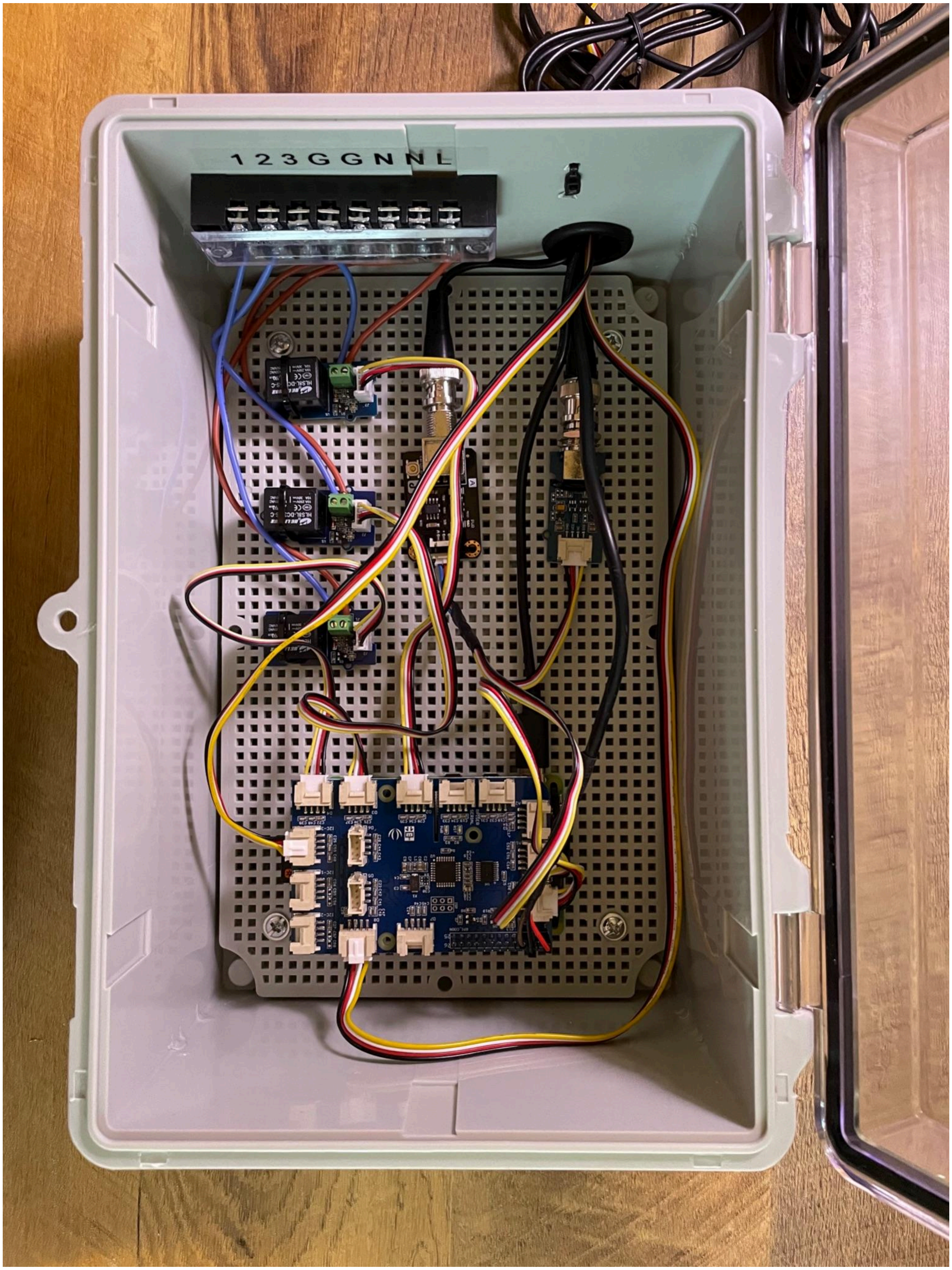
The design was implemented in two stages. First, placeholder components were used because the hardware the team ordered didn't come in until midway through the month of April. The first picture below displays the system, with the wires not going to the Pi connected to sensors.
The second phase was when the team got in the proper components and were able to package them, which are the next three photos. In implementing the design, the team followed the design outlines for hardware and software that are discussed above. Hardware was attached as it is very straightforward, and then code was written and implemented that would allow the hardware to function as it was intended. Then, documentation was written on the operation of the system and the end result is this document and the system that will be turned in after demonstration to the Professor.



Raspberry Pi with Relay and Temperature Sensors Attached and Functional



Raspberry Pi with GrovePi+ and components connected



Pool Controller Unit completed inside waterproof enclosure

Pool Controller Unit sensors for pool and power output

Design Verification/Validation

Below are the tests that were run on each of the sensors and the server. The tests are labelled and explain step by step how to test each sensor (or the server). The last step for each test explains what condition must be met in order for the test to be deemed a success. We conducted all of these tests on the system and have concluded that the design is verifiable and validated and the system is a success.

Server Test

1. Ensure the homebox is plugged in
2. Turn on the home device and get it running
3. Go to the website on any device
4. If the website pulls up than it is working properly

Relay Test

(requires server functionality)

1. Ensure the Relay is plugged into the correct spot of the home device box.
2. Turn on the home device and get it running
3. Plug a high powered light into the output of the relay
4. Turn on the relay via the home device
5. If the light turns on, the relay is working properly

Water Level Test

(requires server functionality)

1. Ensure the water level sensor is plugged into the correct spot of the home device box.
2. Turn on the home device and get it running
3. Place the water level sensor in a cup of room temperature water
4. Look on the home devices UI to see if the water level tab has a reading displayed, if so the water level subsystem is working properly

Temperature sensors

(requires server functionality)

1. Ensure the air and water temperature sensors are plugged into their respective spots on the home device
2. Turn on the home device and get it running
3. Put the water temperature sensor in the water
4. Open up the website and check the temperature
5. If the temperatures are reasonable than they are working properly

Chemical sensor

(requires server functionality)

1. Ensure the Chemical sensor is plugged into the home device properly
2. Turn on the home device and get it running
3. Look at the PH level section of the UI on the website via the home device or other device.
4. Observe the ph level, then put some chemicals on the sensor.
5. If there is a change than the chemical sensor is working properly, if there is not or there is no value there is an issue with the sensor itself or the connection

Recommendations

As far as recommendations go, we do not have an excessive amount, and the ones we do are largely cosmetic. We feel quite good about our software and wouldn't change the decisions we made there unless we needed to add several more sensors, and then we would add more processing power and ensure real-time processing.
On the hardware side, there are changes that we would like to make. If this were a longer class, we would have looked at custom packaging options, like 3D printed housing or potentially different box sizes. We think that the weather proof package chosen is a very viable option, but it isn't the sleekest look in the long-term. Otherwise we are happy with the design decisions we made, as our sensors are all the same brand and communicate in a similar way to the Raspberry Pi.

Appendix

References/Related Documents

1. Embedded Systems Design PPT - The Semester Projects
2. Embedded Systems Design PPT - Requirements Definition
3. Python Web Service (https://fastapi.tiangolo.com)
4. Similar Application (https://create.arduino.cc/projecthub/mmackes/pool-controller-8dfa69)
5. IEEE - National Electrical Safety Code
6. NFPA - Swimming Pools, Hot Tubs and Spas
7. https://www.nsf.org/testing/water/pool-spa/pool-spa-equipment
8. https://www.3mediaweb.com/blog/web-design-standards-guidelines-for-consistency/
9. https://www.calbo.org/sites/main/files/file-attachments/swimming_pool_and_spa_checklist_and_regulations.pdf?1525201941
10. https://www.poolweb.com/oklahoma-pool-regulations
11. https://pooltroopers.com/blog/swimming-pool-laws/
12. https://www.cdc.gov/healthywater/swimming/aquatics-professionals/regulation-inspection.html

Attachments

1. MAHC - model aquatic health code
2. IEEE - National Electrical Safety Code (https://ieee.app.box.com/v/NESC-Value)
3. NFPA - Swimming Pools, Hot Tubs and Spas (https://www.nfpa.org/NEC/electrical-news-and-resources/Hot-topics/Electric-Shock-Drowning/Pools)