Nashua Community College
Nashua, New Hampshire

Electronic Engineering Technology

Template updated by Professor Marcotte 4/24/2021

ELET274N: Capstone Final Project Report

Fire and Flood Detection and Mitigation System

Course Instructor: Prof Hughes

Performed by:	[redacted]
Date submitted:	04/25/2022
Date report due:	05/04/2022

Grading Rubric:

Final Project complete and demonstrated (video) to instructor/+ evaluators	(30): ___
Project Complexity	(10): ___
Final Project write-up is well written, thorough, and clear.	(20): ___
Oral presentation evaluation from reviewers	(30): ___
Uploaded correctly (to Canvas)	(10): ___
Grade	____

No late submission accepted.

Introduction:

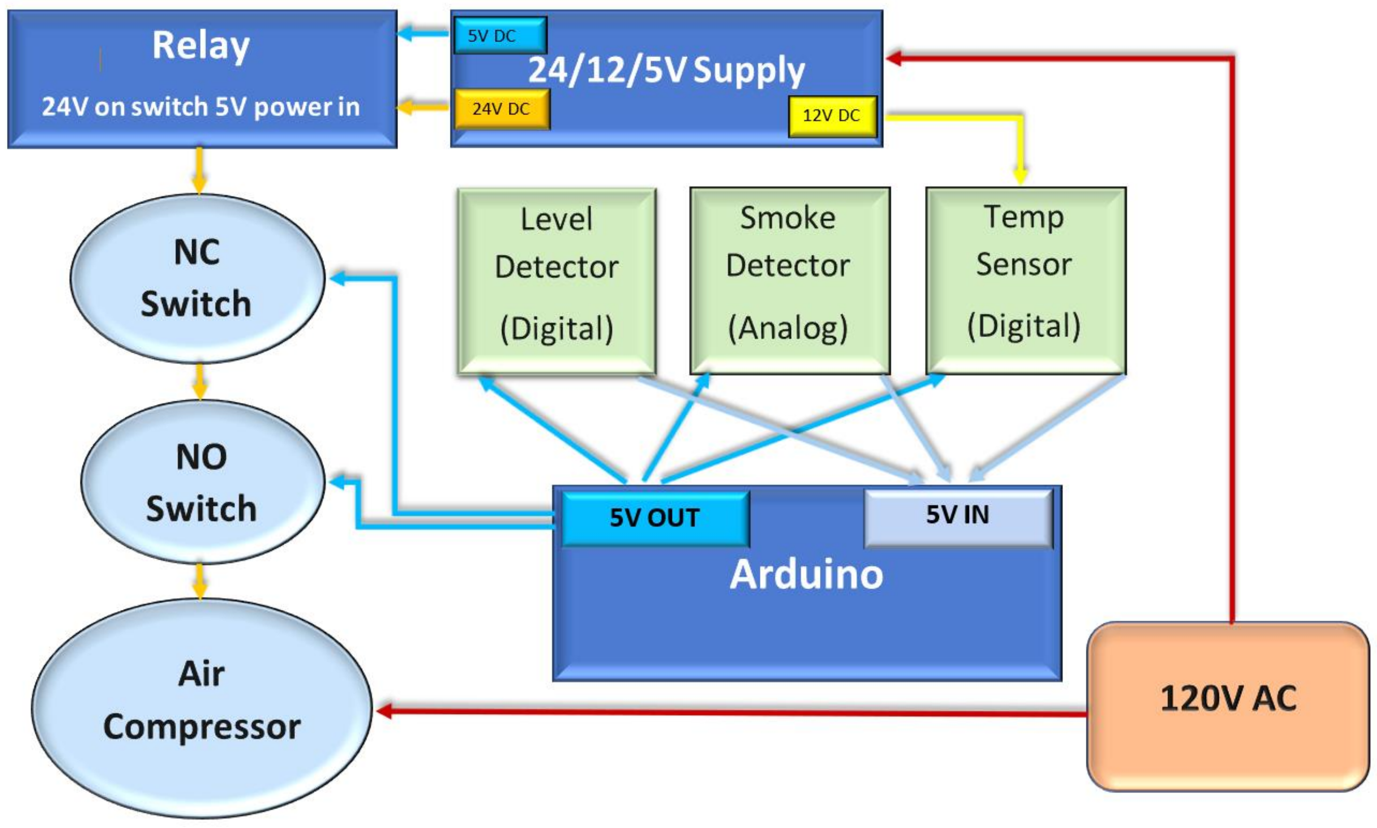
The fire and flood detection and mitigation system is designed to control the flow of water through valves in response to the presence of water and or the presence of heat and smoke. For safety in a class environment, pneumatics substituted water, an alcohol sensor substitutes a smoke detector, and cold temperatures substitute excessive heat. A normally open shutoff solenoid valve is electronically activated when the presence of water is detected via a float switch. A normally closed solenoid is activated in the presence of both cold (hot) temperature and alcohol vapor (smoke). The temperature is registered via a thermistor on a digital control board. Alcohol vapor is detected via an analog alcohol sensor. An Arduino takes the inputs from all sensors and controls the outputs to the solenoid valves. Because both solenoids are attached to the same line; the sprinkler system solenoid takes priority over the shutoff solenoid.

The project spanned from late January 2022 to the end of April. The beginning consisted to theoretical design and proposal. Within a couple weeks all parts were ordered and tested individually to confirm their functionality. Some items did not function as anticipated and there was additional circuitry needed to properly integrate singular parts into the whole of the project. Additional parts and mounting hardware were purchased as necessary. Once all parts could theoretically function together the software was written on the Arduino and the inputs and outputs tested without hardware physically attached. Proceeding this the parts were integrated into the whole one by one with the Arduino running software to test input and output ports to confirm they received the correct signals. After all parts worked as a whole unit the wires were cleaned up, everything was hard mounted, and the project was tested again for functionality. In the final weeks small revisions were made to the design to clean up the appearance of physical components and the software was consolidated and described in detail in the code.

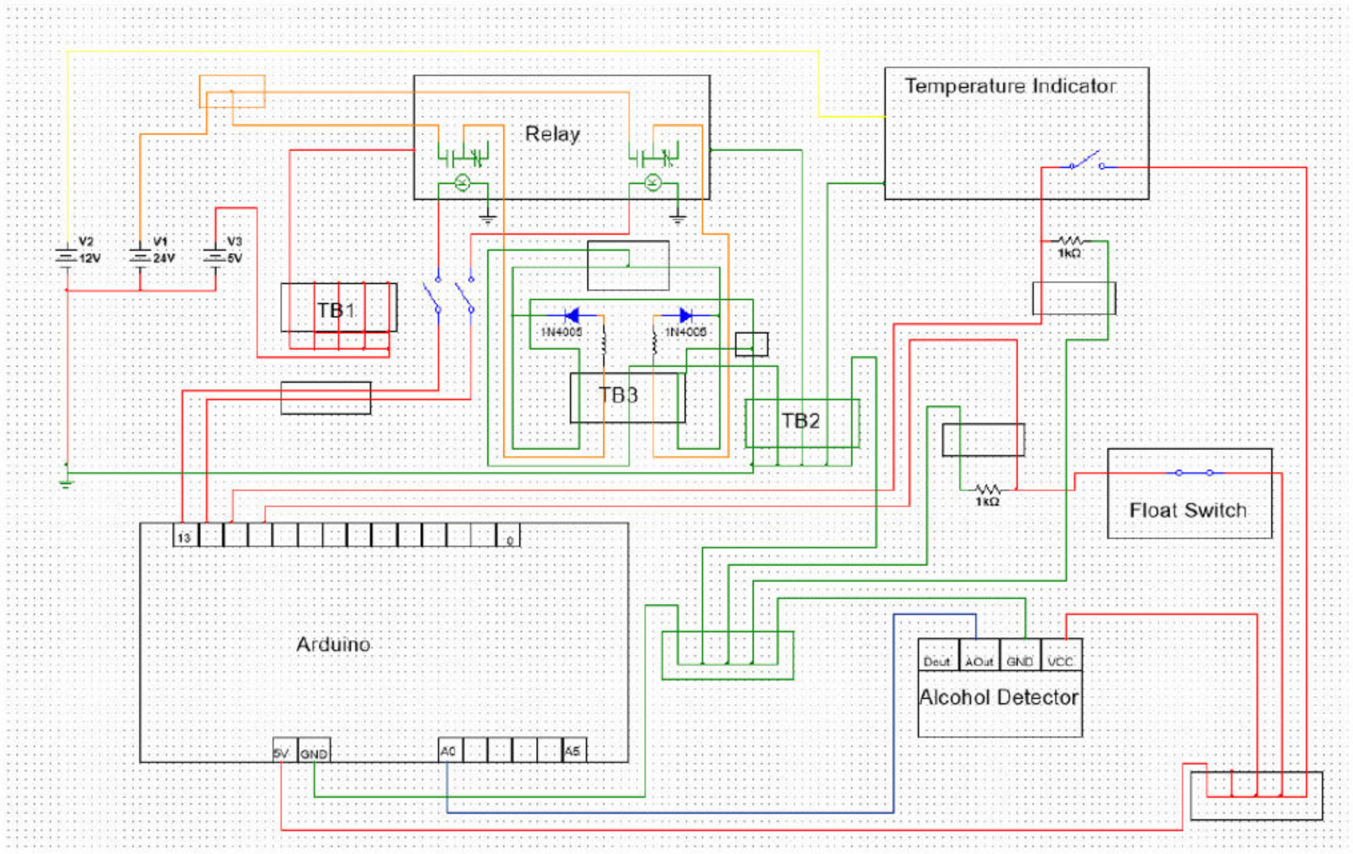
Overall Goal:

The goal of this project was to create a system that shutoff a water supply in the event of flooding and opened a sprinkler system if both heat and smoke were detected. In doing so I wanted to integrate pieces of each class taken leading up to this and combine it into a final working project. This consisted of circuit design, analog circuitry, digital circuitry, and programming. The temperature sensor contained analog and digital circuitry, the float sensor was digital, the smoke detector is analog. The Arduino integrates analog, digital, and software programming. The entire assembly, integration of wiring, resistors, and diodes are representative of circuit design. Together it works as an integrated, real life implementation representative of my education in electronics.

Hardware Block Diagram:



Schematic:



Obtaining Parts (include final BOM):

BOM	
Pancake Air Compressor	$ 60.00
Fittings	$ 38.00
Miscellaneous Electrical Hardware	$ 98.00
Valves	$ 202.00
Water Level Switch	$ 13.00
5V Relay	$ 12.00
Temperature relay	$ 12.00
Gas Detectors	$ 14.00
Electrical Standoffs	$ 15.00
24V/12V/5V power supply	$ 87.00
Plastic Electronics Box	$ 32.00
Cable Glands	$ 16.00
Arduino	$ 27.00
GFCI Outlet/Wiring	$ 53.00
1N4005 Diode (2)	$ -
1KΩ Resistor (2)	$ -
Switches	$ 16.00
Total	$ 695.00

Final Test Plan:

To properly test all aspects of the project testing is broken down into three subcategories. (1) Testing individual components, (2) software functionality, and (3) integrated testing one component at a time. Hardware and software non integral to the functionality of the device is necessary to demonstrate the proper operation of the functional components. The design is composed of components that detect the presence of smoke, heat, and water a control device, a control system, solenoids activated in accordance with what is detected, and a power supply for the electrical components. The following outlines how each component is individually tested out of the box, how software is tested, and how each piece is tested as it is added to the sum of the final project.

Testing Individual components:

Each functional component must be tested individually before being combined into the final product. The relay is powered with 5V DC. Each trigger is individually energized with 5V DC and each relay is confirmed to switch between normally closed continuity at a zero-energy state to continuity to the normally open switch when energized. The alcohol detector should be energized with 5V DC. While monitoring the analog output with a DMM, alcohol should be brough within an inch of the device. An increase in the output voltage from approximately 0V DC to greater than 5V DC should be measured. The float switch should be checked for continuity. In its normal resting state, the input should have continuity to the output and when the float is activated there should be no continuity. The temperature indicator should be energized with 12V DC, and heating mode selected from mode button. Temperature trigger set to 12 degrees Celsius and a hysteresis of 1 degree C. With these set touch the probe to an ice cube, confirm the relay is open and closes when the temperature reaches 12 degrees. Confirm relay opens again once the temperature reaches 13 degrees. The power supply is tested by first confirming it is set to 120V AC, and not 220 AC. Attach three clips from power cord to line, neutral, and ground. Turn on the supply and measure the 5V, 12V, and 24V DC respectively to confirm they are within a 10% tolerance. The 5V can be adjusted with ADJ knob on power supply if out of tolerance, but the others cannot. To test the solenoids, tie one of the line wires to 24V DC and the other side and ground to common. Energize with 24V and listen for valve to click open or closed.

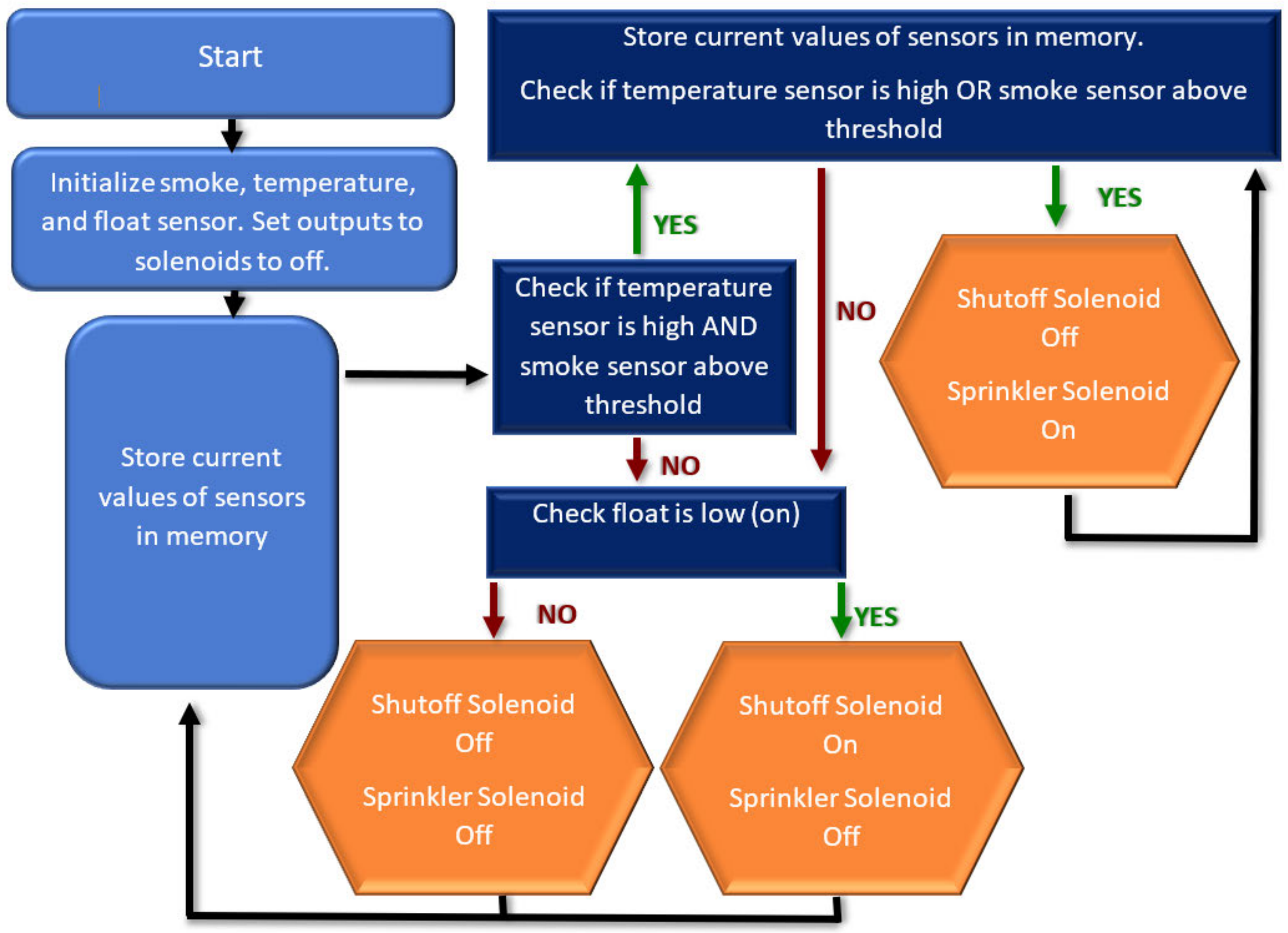
Testing Software:

The second stage is software functionality testing. Here the software is written to the Arduino and the inputs and outputs are tested and measured to verify signals will be processed correctly when integrated with the remaining hardware. The program starts, initializes the sensors, and then monitors the sensors for changes to control the output signal to two solenoid valves. To properly test this a low signal is sent to the pin the float sensor would be connected to and the output pin for the shutoff solenoid is monitored to ensure it goes high. Next the temperature sensor input pin is triggered and then the smoke independently to make sure the shutoff valve stays on, but the sprinkler solenoid pin is not activated. Then all sensors are triggered and the output on the sprinkler solenoid pin is measured to confirm it is high and the shutoff pin goes low. The program is then reset, and the temperature sensor pin, smoke sensor pin, and float signals pin are all set high and the output to the sprinkler system pin is measured to make sure it is high, and the shutoff is low. Next the smoke signal pin is set to low and the outputs to the solenoids should be measured to confirm they do not change. The smoke sensor pin is then set high, and the temperature pin set to low; again, confirming no change to the solenoids. Finally, the temperature and smoke sensor are set to low and the output to both solenoids should be low. During each step while the sprinkler solenoid is high the float signal is cycled, and the shutoff solenoid monitored to ensure it stays low.

Testing Integrated System:

First the power supply is mounted, wired to 120V AC, powered on and all outputs confirmed to be within specifications. Next the Arduino is mounted with the ground linked to the common on the power supply. Following this each remaining item is mounted one by one and the functionality is tested. Once all items are secured the functionality is tested to ensure it functions in accordance with design specifications. Use schematic drawing as a reference for voltages and wiring. The recommended order for attaching hardware is as follows: Solenoids, relay, temperature sensor. At this point the integrated functionality of the shutoff valve can be tested. After this the temperature sensor and then the smoke sensor can be attached, and the integrated control can be tested. This is done in the same manner as described in the software functionality section, but the observed reactions of the components are observed. The shutoff switches are tested by turning on the shutoff solenoid with the float, then turning the switch to off and confirming the valve opens again. The float is then actuated again to ensure while the switch is off, the solenoid cannot be activated. For the sprinkler solenoid both the temperature sensor and smoke detector are set to high, the switch turned to off, and the sprinkler solenoid is to deactivate. Turn both sensor signals to low, then back to high and confirm the sprinkler solenoid cannot be activated. Turn both manual switches off and repeat the steps up to testing the manual switches to confirm all components are still operating correctly.

Flow Chart:


Start
Initialize smoke, temperature, and float sensor. Set outputs to solenoids to off.
Store current values of sensors in memory
Check if temperature sensor is high AND smoke sensor above threshold
YES
NO
Store current values of sensors in memory.
Check if temperature sensor is high OR smoke sensor above threshold
NO
YES
Shutoff Solenoid Off
Sprinkler Solenoid On
Check float is low (on)
NO
YES
Shutoff Solenoid Off
Sprinkler Solenoid Off
Shutoff Solenoid On
Sprinkler Solenoid Off

Final Code:

```
/*******

James Murphy
EET Capstone
Spring 2022

*******/

//Initializing pin values
int Shutoff_Solenoid = 13;                // Shutoff Solenoid connected to pin 13
int Sprinkler_Solenoid = 12;             /// Sprinkler Solenoid connected to pin 12
int Temp_Switch = 11;                   //// Temperature Switch connected to pin 11
int Float_Switch = 10;                 ///// Float Switch connected to pin 10
int Smoke = A0;                       ////// Analog Smoke Sensor connected to pin A0
int sensorThres = 400;               /////// Threshhold value to compare analog smoke readout against
```

```
//Initialization setup for pins
void setup() {
  pinMode(Shutoff_Solenoid, OUTPUT);          //Shutoff Solonoid is an output
  pinMode(Sprinkler_Solenoid, OUTPUT);       ///Sprinkler Solenoid is an output
  pinMode(Temp_Switch, INPUT_PULLUP);       ////Temperature Switch is an input on an internal pullup resistor
  pinMode(Float_Switch, INPUT_PULLUP);     /////Float Switch is an input on an internal pullup resistor
  pinMode(Smoke, INPUT);                  //////Smoke sensor is an analog input

  Serial.begin(9600);                        //Initialize serial port with 9600 baud rate
  digitalWrite(Sprinkler_Solenoid, LOW);    ///Set output to Sprinkler Solenoid to low (Off)
  digitalWrite(Shutoff_Solenoid, LOW);     ////Set output to Shutoff Solenoid to low (Off)
}


//Beginning of program loop
void loop() {

  Serial.print("Pin A0: ");                   //Text printed to serial port
  Serial.println(Smoke);                     ///Print the value read from pin A0, smoke

//Read each sensor and store values
  Val_Temp = digitalRead (Temp_Switch);
  Val_Float= digitalRead(Float_Switch);
  Val_Smoke=analogRead(Smoke);

 // Checks if smoke reached threshold value and temperature sensor is high
  while (Val_Smoke > sensorThres && Val_Temp == HIGH)
    {

     //Checks to see if smoke sensor is above threshold value or temperature is high. Both loops drop only when both sensors are low
     while (Val_Smoke > sensorThres or Val_Temp == HIGH)
     {
     //Read each sensor and store values
    Val_Temp = digitalRead (Temp_Switch);
    Val_Float= digitalRead(Float_Switch);
    Val_Smoke=analogRead(Smoke);

    digitalWrite(Sprinkler_Solenoid, HIGH);   //Sprinkler Solenoid turned on
    digitalWrite(Shutoff_Solenoid, LOW);     ///Shutoff Solenoid turned off
    delay(2000);                            ////2 second delay
      };
    };


//Checks to see if float has risen, resulting in a low signal. Turns on shutoff and turns sprinker off(redundent)
  if  (Val_Float == LOW)
    {
     digitalWrite(Shutoff_Solenoid, HIGH);
     digitalWrite(Sprinkler_Solenoid, LOW);
    }
//If the float is still down, both solenoids are set to off (redundent)
   else
    {
      digitalWrite(Shutoff_Solenoid, LOW);
      digitalWrite(Sprinkler_Solenoid, LOW);
    }
//2 second delay
  delay(2000);
}
```

Summary:

Overall, the project worked as expected and originally designed. With time to spare I was able to add manual shutoff switches to both the solenoids but was unable to complete the more complex task of integrating a Bluetooth transmitter into the project and building a phone-based application for control. This could be accomplished if this was a two-semester project instead of one. The Arduino's software was tested completely by mocking input signals and measuring output signals to ensure all software troubleshooting was complete before final assembly. By ordering parts early on, looking at multiple suppliers to reduce supply chain related delays, and making minor substitutions when necessary; all the primary components were ordered early, and I was able to test them within the first four weeks. This allowed me to order replacement parts for broken ones, and order additional small parts to improve the overall functionality early on. The largest setback was a power supply that took several weeks to arrive and did not function according to specification. This led to the relay and temperature indicator being destroyed. This almost put the project behind schedule, but significant time was allotted to the full project assembly as soon as replacement parts arrived. This setback led me to purchase additional backup relays, float sensor, and an Arduino on the off chance any other component was damaged in the final integration. Withing two days after the new power supply arrived, the rough wiring was complete, and everything worked correctly. With approximately a month before final presentation I used this time to add the manual shutoff switches, order additional mounting hardware, modified the code for better functionality with less lines of code, and cleaned up the overall appearance of the project.

There is a market for this project design in both a home and industrial setting. Personally installed and monitored home security systems have become increasingly more popular. Most components consist of cameras, door and window sensors, and motion sensors. This brings a higher level as shutoff valves in one's home would require a professional plumber to install. What is great is once installed; the homeowner has full control over the system and general maintenance should primarily consist of changing the battery on the valve. There are motors on the market that can be installed on ball valves to turn them on and off for around $65 and fully integrated systems for around $350. Using an electronic ball valve and control system, when bought in bulk can be done for around $65. This is as good as the $350 product and more reliable than the $65 project at an intermediate price. The sprinkler system again is targeted as a newer design at a mid-price range. It is more expensive than a traditional sprinkler system but could be custom programmed to reduce overall damage to a facility in the case of a fire as multiple sprinklers could be triggered off one going off. This is unlike a traditional system where fire needs to spread to other areas and trigger those sprinklers before they start going off. Too often sprinklers are bumped and broken during construction, causing flooding. This design would allow for easy shutoff and minimize flooding damage. It is a mid-level design between traditional sprinklers, and full automation of foaming agents and oxygen depletion systems sometimes used to protect high value products from fire.

I learned several things about electronic project design from this I can take into the future. Amazon is not a great place to acquire electronics as the prices are typically higher than if they were ordered straight from the supplier, the parts are generally low to mid quality, and they may be coming from sitting on a shelf for years. There are generally many suppliers for the same or similar components. To reduce lead times, it is best to look at two to three suppliers for similar components. If an electrical component does not have a specification sheet, or it is poorly written; it is highly probable the component will not function as expected so it is best to look for alternatives. Although I've been attaching proper grounding to components for years, I never truly understood the rationale behind different grounding systems. The feedback I received from online forums was mixed. As such sometimes it is best to consult with an industry professional before proceeding with a design as we all have limitations, and it is best to do something right even if you feel it is an unintelligent question to ask. In line with the grounding, I needed to overcome the back voltage cause by the electromagnetic field of the solenoid's inductor breaking down. There was a good chance without the diodes to correct for this the project would work right but I wanted to look at every component as if I knew nothing, thoroughly research it, and see if I could find ways to improve them for personal safety and longevity of the components. Overall, I combined the knowledge from electronics and programming into a final integrated design that works as intended.