# SYSC 3010
# Computer Systems Development Project
# Print Guard
# Detailed Design Document

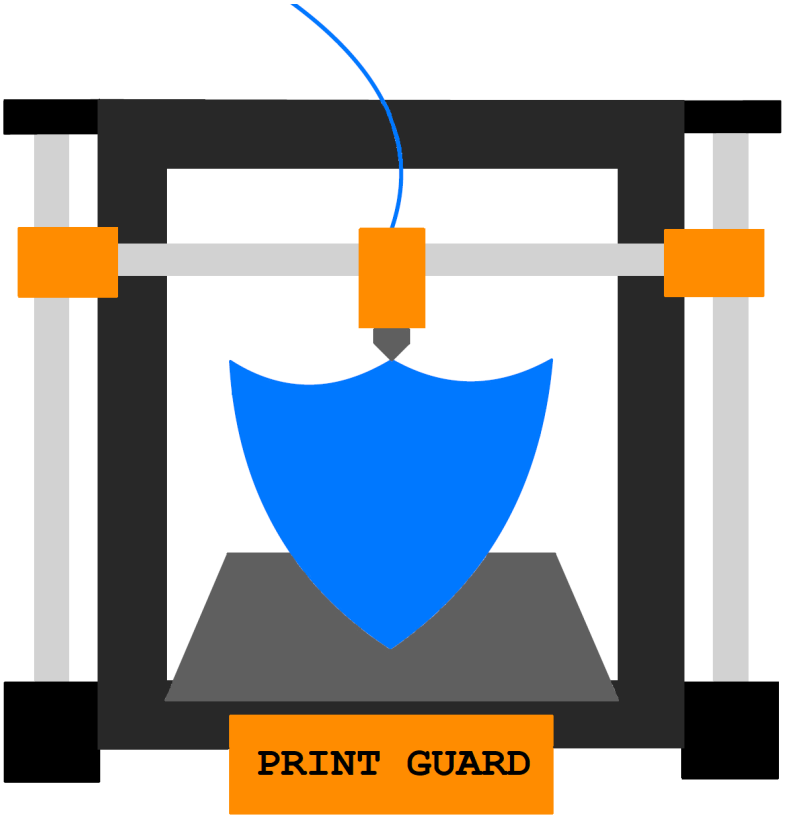



**Figure 1:** Print Guard Logo.

Group L2-G9
Joseph Vretenar - 101234613
Basma Aboushaer - 101186291
Cholen Premjeyanth - 101184247
Raunak Singh Soi - 101219648
Samual Mauricla - 101233500
TA: Ben Earle
April 10th 2024

# Table of Contents

# 1 Project Description

## 1.1 Motivation

The objective of the Print Guard system was to allow users to remotely monitor the status of their 3D printers, ensuring proper operation and to minimize wanted time, energy, and filament associated with print failures. This project can also be incorporated to any place that has a 3D printer, which would include anyone who uses a 3D printer. This versatility allows the system to be applicable for hobbyists, schools, small businesses, and even larger enterprises.

## 1.2 Problem

The problem with 3D printing is that there are many errors that can occur throughout the printing process. These errors can be quite detrimental to the throughput and use of the 3D printer. The main problems were a lack of temperature control around the print space, and no remote monitoring and control of the 3D printer. These problems would lead to errors in the print that can’t be fixed without being at the printer while it is printing, which can be harder especially with longer prints.

## 1.3 Overview of Solution

The solution to the problems mentioned above was the Print Guard system, which features an array of sensors, along with cameras for remote viewing, and a specially built enclosure to ensure good temperature control. The cameras allow for remote viewing of the printer throughout the printing process which is hosted on a website that links to the graphical user interface (GUI). The GUI contains the temperature and humidity sensors on the inside and outside of the enclosure, the fan speed which changes based on the internal temperature, and the printers sensors to monitor the printer directly.

# 2 Final Design Solution

## 2.1 Deployment Diagram

The two deployment diagrams below show the layout of the print guard system. The server with AI processing is a future goal to expand the Print Guard system. The main components in the deployment diagram are the two Arduinos, two Raspberry Pis and the 3D printer as computer devices. The sensors are two DHT22 sensors, the pi camera and the printer's sensors. The actuator in the system is the 120mm PWM fan.

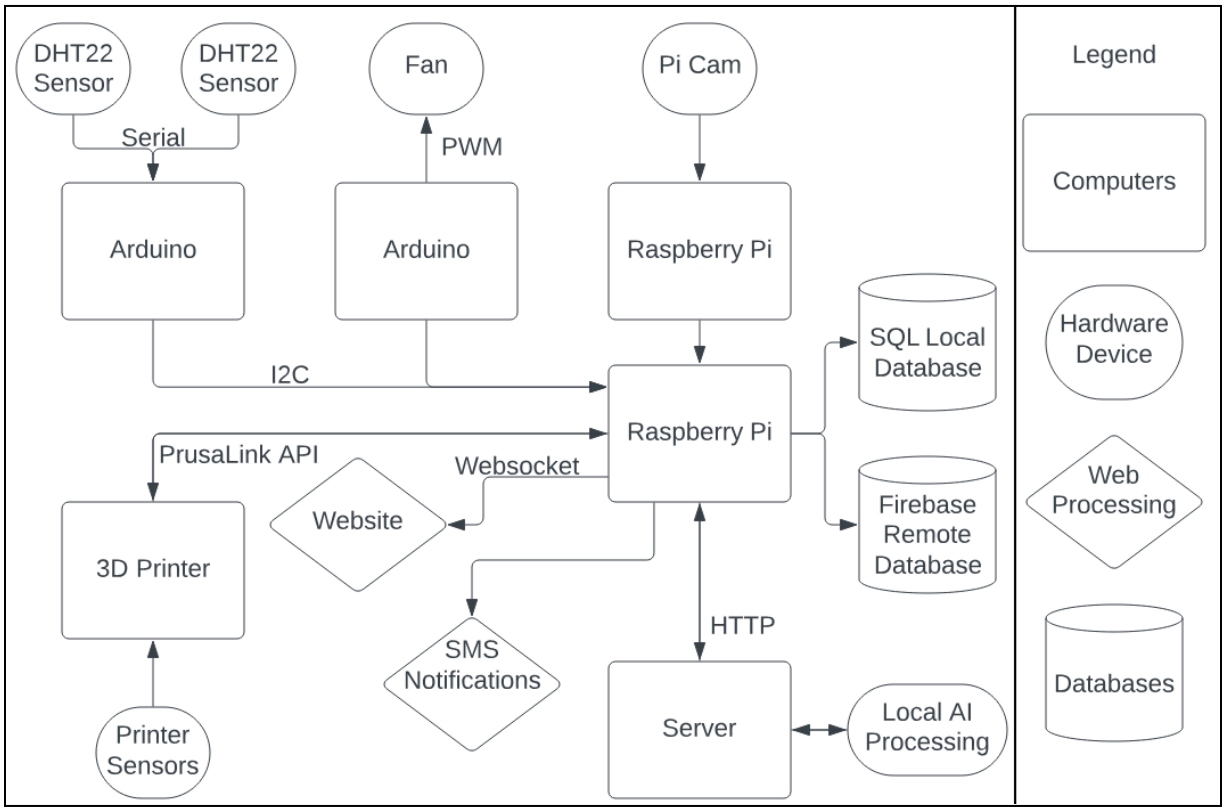



**Figure 2:** Simple Deployment Diagram

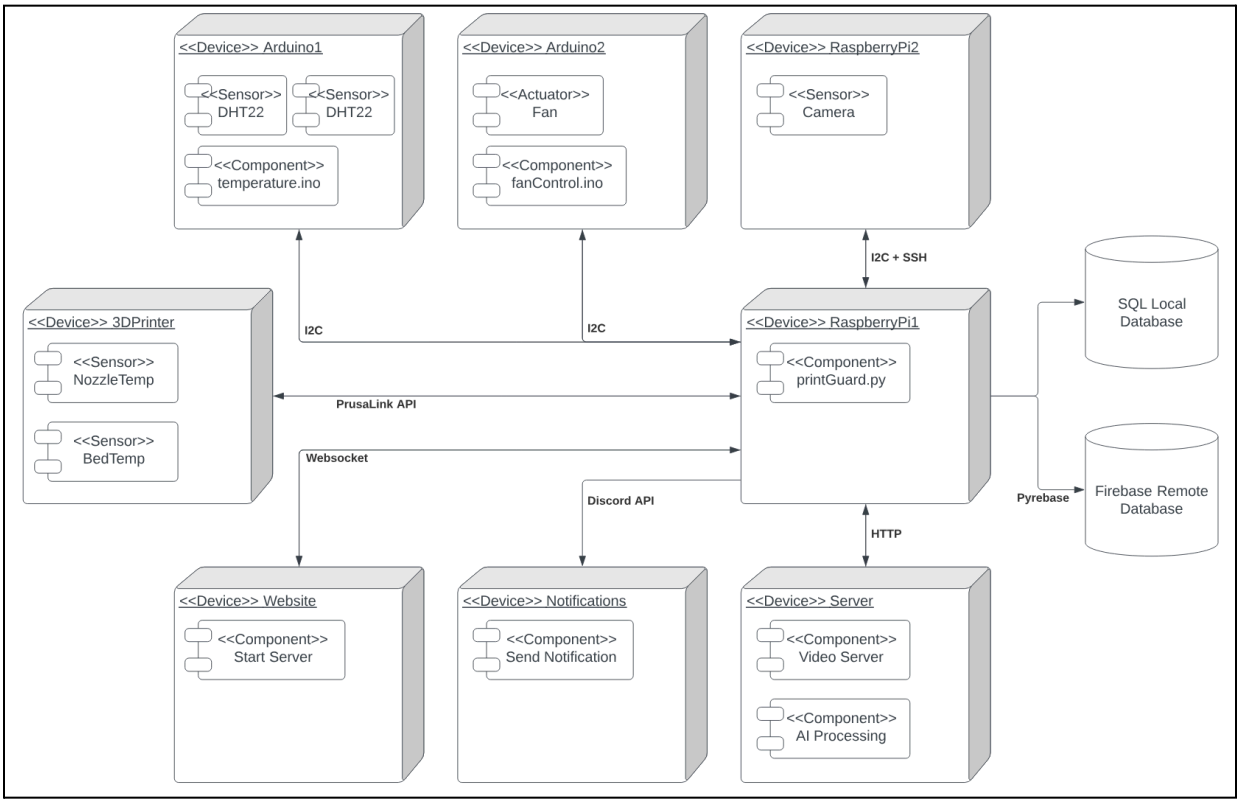



**Figure 3:** UML Deployment Diagram

## 2.2 Message Protocol Table

| Sender | Receiver | Message | Parameters | Return |
|---|---|---|---|---|
| Arduino 1 | DHT22 Internal | getInternal | `sensorName` | `temp, humidity` |
| Arduino 1 | DHT22 External | getExternal | `sensorName` | `temp, humidity` |
| Arduino 2 | Fan | changeSpeed | `PWM (0 - 255)` | `none` |
| RPi 1 | Arduino 1 | getTemps | `none` | `temp1, humidity1, temp2, humidity2` |
| RPi 1 | Arduino 2 | changeSpeed | `bool: 1(up), 0(down)` | `none` |
| RPi 2 | Pi Camera | takePhoto | `imageName, resolution` | `none` |
| RPi 2 | RPi 1 | sendPhoto | `imageName` | `none` |
| RPi 1 | 3D Printer | getTemps | `printer` | `nozzleTemp, bedTemp` |
| RPi 1 | 3D Printer | shutdown | `printer` | `none` |
| RPi 1 | 3D Printer | getRemaining | `printer` | `printTime, printPercentage` |
| RPi 1 | SQLite | storeDHT | `temp1, humidity1, temp2, humidity2, datetime` | `none` |
| RPi 1 | SQLite | storePrinter | `nozzleTemp, bedTemp, imageName, datetime` | `none` |
| RPi 1 | Firebase | storeDHT | `temp1, humidity1, temp2, humidity2, datetime` | log if connection cannot be established |

| | | | | |
|---|---|---|---|---|
| RPi 1 | Firebase | storePrinter | `nozzleTemp, bedTemp, imageName, datetime` | log if connection cannot be established |
| RPi 1 | Webserver | updateTable1 | `temp1, humidity1, temp2, humidity2, datetime` | `none` |
| RPi 1 | Webserver | updateTable2 | `nozzleTemp, bedTemp, imageName, datetime` | `none` |
| RPi 1 | Webserver | updateGraph | `none` | `none` |
| RPi 1 | Webserver | updatePrint | `printTime, printPercentage` | `none` |
| websocket | RPi 1 | getView | `password` | `cameraView, or error + log if authFailed` |
| websocket | RPi 1 | shutdown | `bool: 1(yes), 2(no)` | `Shutdown Message` |
| RPi 1 | Discord | highTemp | `none` | `none` |
| RPi 1 | Discord | crashDetected | `none` | `none` |

## 2.3 End-to-End Behavior Sequence Diagrams

### 2.3.1 Recording Sensor Data:

The Arduino collects sensor data every 15 seconds and sends it to Raspberry Pi 1. Pi 1 stores the data locally in SQLite and synchronizes it with Firebase. The GUI is updated with real-time sensor readings, and notifications are sent if temperature readings fall outside predefined ranges.

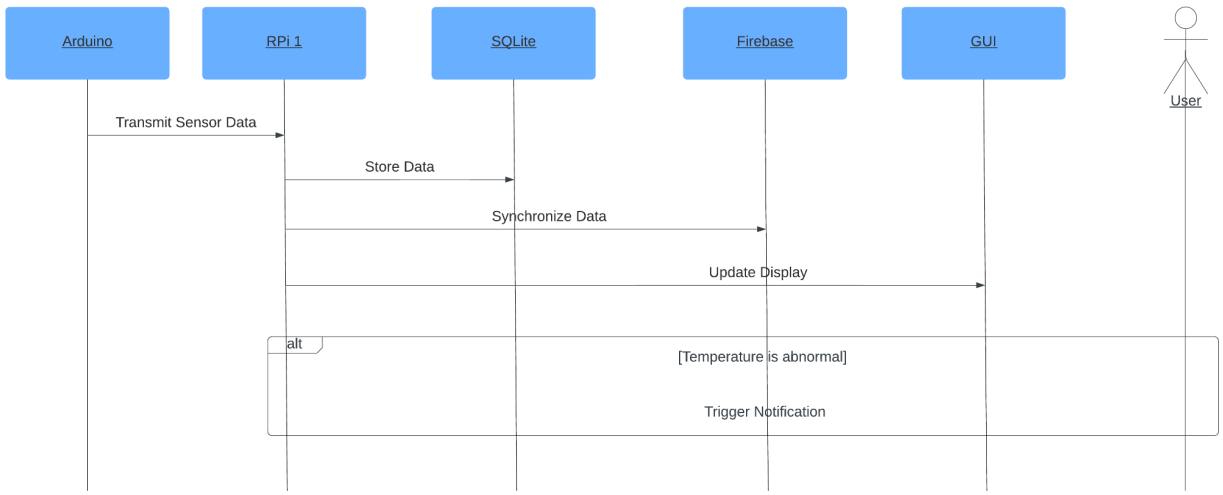



**Figure 4:** Message Sequence Diagram for Recording Sensor Data

## 2.3.2 Capturing Photos:

When a user prints an item, RPi 1 activates RPi 2 to capture photos. RPi 2 takes a photo and sends it back to RPi 1, which adds it to SQLite, Firebase, and updates the GUI.

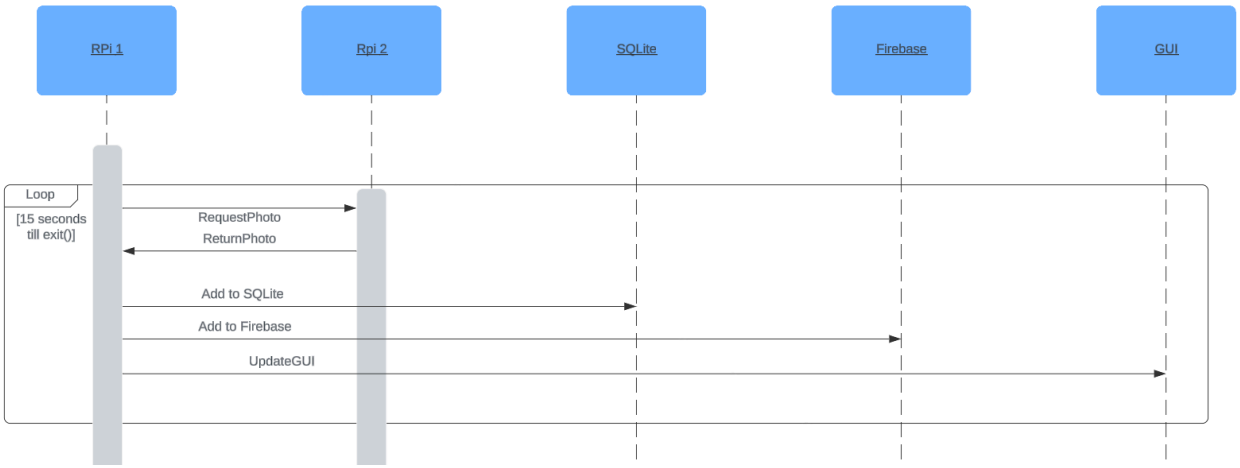



**Figure 5:** Message Sequence Diagram for Capturing Photos

### 2.3.3 3D Printer Control:

Users can remotely control the 3D printer through the GUI. Commands sent through the website are confirmed and executed by RPi 1, which then sends the command to the printer.

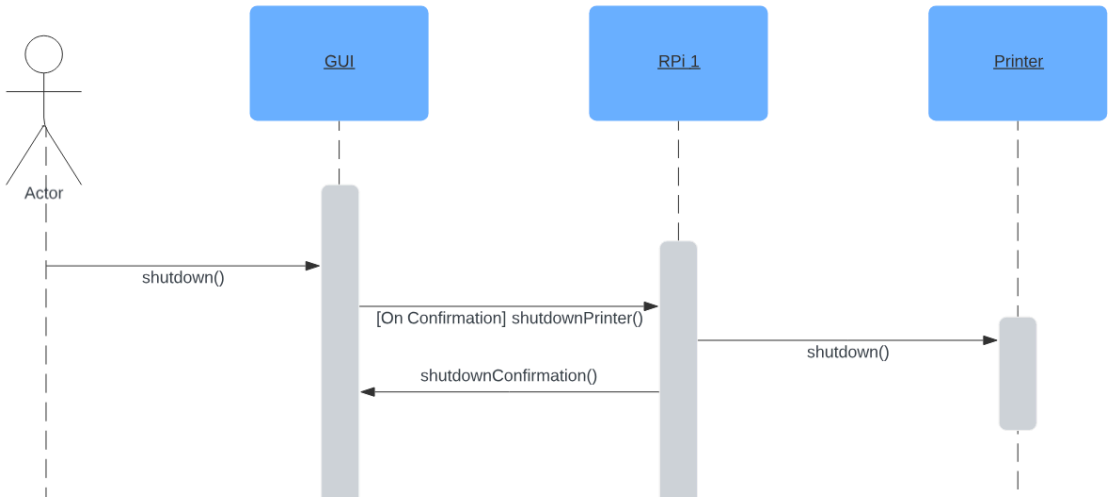



**Figure 6:** Message Sequence Diagram for 3D Printer Control

### 2.3.4 Notifying Users:

If errors occur, RPi 1 notifies RPi 2 to take a photo for analysis. Once the photo is taken, the user receives an error message via Discord. RPi 1 records error data in SQLite and Firebase for future reference.

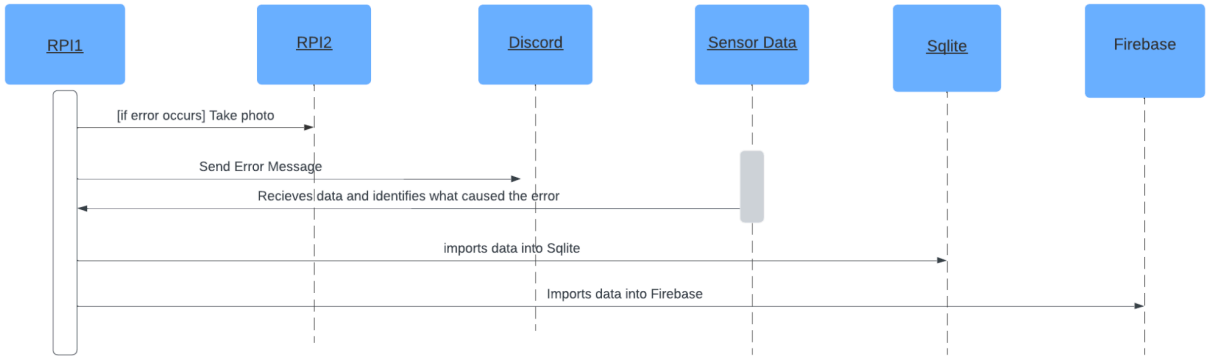



**Figure 7:** Message Sequence Diagram for Notifying Users

### 2.3.5 Live Camera Feed:

Users can view the live stream of the printing process through the GUI by entering a password for security. RPi 1 authorizes the stream, instructs RPi 2 to start the feed, and updates the GUI with the live view.

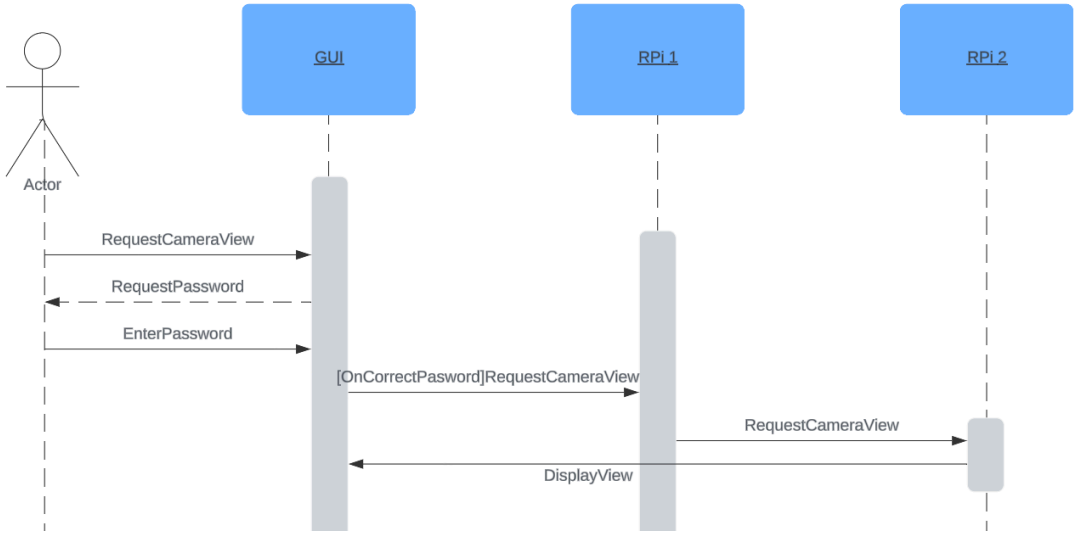



**Figure 8:** Message Sequence Diagram for Live Camera Feed

# 3 Discussion of Final Design

The final design of the Print Guard system completed all the main tasks that were set out. The final design differed slightly from the prototype as the system ended up having two Pi cameras instead of one due to streaming issues. The final version just added a second Pi camera to the first Raspberry Pi for taking photos. This was a simple solution to the problem we encountered with streaming, for future versions it would be great if this second Pi camera was removed. All the communication protocols stayed the same between devices with I2C being the main hardware communication protocol.

# 4 Contributions

## 4.1 Project Contributions

| | Project | Tests |
|---|---|---|
| Joseph Vretenar | RPi-Node1.py<br>fanDeployment.ino<br>GUITemplate.html<br>PrintGuardGUI.py<br>README.md | rpiCommTest.py<br>fanTest.ino<br>printerAPITest.py<br>printerUnitTest.py<br>sensorunittestrpi.py<br>unitTest_RaunakFan.ino |
| Basma Aboushaer | RPi-Node1.py<br>dhtDeployment.ino<br>README.md | sensorTest.ino<br>rpiCommTest.py<br>DHT22sensordemounittest.ino<br>sensorunittestrpi.py |
| Samuel Mauricla | RPi-Node1.py<br>README.md | firebaseTest.py<br>rpiCommTest.py<br>DbUnitTest.py |
| Raunak Singh Soi | PrintGuardGUI.py<br>GUITemplate.html | GUI_unitTest.html<br>PrintGuardGUI_Flask.py<br>unitTest_RaunakFan.ino |
| Cholen Premjeyanth | RPi-Node2.py | livestreamtest.py |

## 4.2 Document Contributions

| | Contribution |
|---|---|
| Joseph Vretenar | 1.1, 1.2, 1.3, 2.1, 2.3.2,3, 3, 4.1, 4.2, 5.1, 6 A |
| Basma Aboushaer | 2.2, 2.3, 2.3.1, 2.3.5, 4.1, 4.2, 5.2, 6 B, D, E |
| Samuel Mauricla | 2.3.4, 3, 4.1, 4.2, 5.3, 6 A, C |
| Raunak Singh Soi | 5.4 |
| Cholen Premjeyanth | 5.5 |

# 5 Reflections

## 5.1 Joseph Vretenar

Overall I felt as though the project went pretty well. We struggled greatly with quality contributions and equal collaboration throughout the project. We were able to overcome these challenges and learn from them for future work. This allowed us to finish the project strongly, creating a final system that we should all be proud of.

While creating this project, I learned a lot about the Raspberry Pi, the I2C communication protocol, and HTTP requests with APIs. I was able to create my own python library to control and read data from Prusa 3D printers from an API. The project taught me a lot to try and think how other people in the team think. This helped me to try and work towards people's strengths when we were assigning tasks where I could try to provide my insight towards documents and websites that help me.

With more time to work on the Print Guard system it would be great to implement AI and machine learning (ML) for crash detection through a camera. This would allow for faster reaction to problems with prints and enhanced protection. It would also be great to use AI and ML to help tune the system and produce better prints. Finally I would like to update the GUI to show more details that are essential to prints, as well as having the video be embedded.

## 5.2 Basma Aboushaer

Our journey with the print guard system was a rollercoaster of challenges and successes, ultimately leading to a successful outcome. Initially, our group faced the typical obstacles of getting to know each other and determining our individual strengths and preferences within the project. We encountered differences in the quality of work among group members, which required some of us to put in extra effort to improve certain aspects of the project. However, instead of letting these challenges slow down our progress, we embraced them as opportunities to grow and learn together.

Throughout the project, I gained new skills and knowledge. From coding for Arduino and Raspberry Pi to wiring sensors on a breadboard and implementing I2C communication protocols, each step provided valuable insights into the details of hardware and software integration. It was incredibly satisfying to see how our collective hard work and dedication culminated in a successful project outcome. The sense of achievement derived from overcoming obstacles and observing the results of our efforts was truly rewarding. Looking back, I realize the importance of effective teamwork and collaboration in achieving our goals. Despite our initial differences, we were able to rally together, support each other, and leverage our individual strengths to meet the project milestones and goals we had set. This experience not only enhanced my technical skills but also taught me valuable lessons in communication, problem-solving, and adaptability.

To further enhance the Print Guard's functionality and impact. I would explore opportunities to integrate additional sensors or features into the print guard system, such as smoke or gas sensors, to enhance safety measures. Additionally, leveraging data analytics to optimize printer performance and energy efficiency based on usage patterns could further enhance the system's efficiency and effectiveness. Overall, the print guard project was not just about building a functional system but also about the journey of collaboration, learning, and growth. It taught me valuable lessons that I will carry forward in future endeavors, both academically and professionally.

## 5.3 Samuel Mauricla

Overall as a group I felt the project had its ups and downs. In the beginning as a group we struggled to work together and get things done as a group. Some people had to work harder than the others but as the semester carried on, the group worked progressively better. In the beginning I did have struggles as I was new to raspberry pi but as the weeks went by, my understanding for using the raspberry pi became simpler. Having a background in python did make things a bit easier as I was able to code pretty well throughout the project. Overall the project towards the end of the semester went very well as a group we were able to get the job done.

While doing this project, I was able to learn quite a bit of new things that allowed me to implement what I learned in the project. One thing that I learned was to send notifications through discord. This allowed my group to receive notifications each time the Print Guard was

going to print something or if there was an error. Another thing I learned was using the raspberry pi to send data to the Firebase. This allowed us to send sensor data collected by the raspberry pi and store it in our Firebase. This was an essential part because it allowed us to collect data and see if there were any discrepancies.

Given additional time, I'd want to add all of the photos saved in Firebase and present them in the GUI. I mention this because, while the photographs are now saved in Firebase, they are not accessible because they are in bit form. Because they are in bit form, they are difficult to read, and we are unable to view them in Firebase. Given more time, I believe the group might have found a means to retrieve the photographs captured via the GUI.

## 5.4 Raunak Singh Soi

There was an initial struggle in the group when trying to kickstart the deployment of individual tasks, but the group showed great resilience and stuck through all challenges we faced. Joining the group late and catching up with the labs and discussions was initially challenging, but as we familiarized ourselves with group dynamics and communication channels, our integration improved. Towards the latter half, we found our rhythm and contributed effectively to the team's goals. The motivation increased also when we were able to see real-world implementations of our hard work.

As I've mentioned earlier in the project video, the project made me comfortable with collaboration in the realm of programming and effectively managing versions of the system on GitHub. From a personal perspective, I wanted to try my hand at new technology, hence why I chose the GUI, and have really honed some skills in HTML, CSS, and JS. My involvement with the fan cooling system also helped me research and learn a few new things about Arduino's I2C communication, etc. The fan cooling system helped us remove temperature as a node of possible failure. The GUI helped us visualize all different aspects of the system on one page while also maintaining additional features of printer control and some security measures.

Expansion-wise, I feel from a web or app development perspective it is very feasible to launch as a real-world product, as the actual application is very straightforward from a software perspective. Overall, I am very proud of what our team has achieved, and this project has surely nudged me to try new technologies within the scope of computers and software.

## 5.5 Cholen Premjeyanth

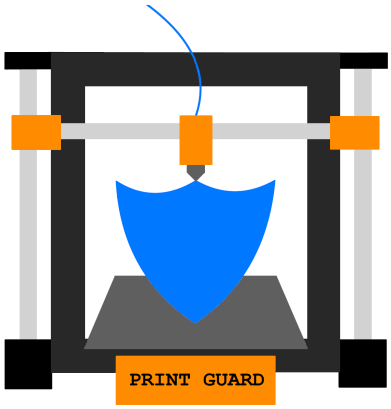
I felt the project went pretty well. On a personal basis, I struggled a little more with the learning curve that came with the Raspberry Pi, but my team members constantly supported me and made sure I had all the tools to succeed. As the project went on, I feel my contribution also increased, in all aspects. Overall, I learned a lot about myself and working with a team. From a project standpoint, I think it went very well. We were able to hit all our milestones. That, in itself, indicates how successful the project was.

Throughout this project, I learned various things, relating to technical aspects such as the Raspberry Pi, and also things related to working in a team. In terms of the Raspberry Pi, I think the most important thing I learned was how to host websites using Raspberry Pi. It taught me how to locally and publicly host websites, which is something I had never known how to do before. In terms of the team aspect, I learned how important good communication was.

If we were given the opportunity to work on this project for a longer period of time, a key attribute that we could include would be to have one system with multiple printers. This would allow a user to create many projects while having an overview of each project at once. Overall, I do believe that this project was great both for me and my group members. I had a lot of fun working with this group and am very grateful for the support and help they provided me with.

# 6 Appendix

## Appendix A: README




# Print Guard:

# Real-time 3D Printer Monitoring and Management System

## SYSC 3010 Computer Systems Development Project

**Group L2-G9 Members:**

Joseph Vretenar
Basma Aboushaer
Samuel Mauricla
Raunak Singh Soi
Cholen Premjeyanth

**TA:**

Ben Earle

# Table of Contents

# Project Overview

## Project Description

The Print Guard system will monitor an enclosure for the 3D printer as well as allow for remote control of the 3D printers. This will help to prevent crashes by minimizing changes around the printer. Currently the Print Guard system can control Prusa 3D printers. The system uses two DHT22 sensors to monitor the temperature and humidity both inside and outside of the enclosure. The inner sensor is used to adjust the speed of a 120mm fan so the enclosure can maintain a stable temperature. The system also uses a Raspberry Pi camera to take photos and stream a live feed to the GUI. The GUI will show the user all sensor data including printer data, as well as providing them control over the printer and access to the live feed.

## Project Structure

The Print Guard system is split among two Raspberry Pis and two Arduinos. The first Raspberry Pi is used for communication with the Arduinos, database storage, image capturing, and discord notifications. This Raspberry Pi will run `RPi-Node1.py` which can be seen in the `src` folder below. The Second Raspberry Pi is responsible for streaming and hosting the website, it will be running both `RPi-Node2.py` and `PrintGuardGUI.py` located below. One of the Arduinos will be running `fanDeployment.ino` and the other will be running `dhtDeployment.ino` and they should be wired up to the respective parts.

All test functions can be found in either the endToEndTests folder or in the unitTests folder. The end to end tests are used to test the communication protocols between devices, while the unit tests are used for testing the individual hardware and software components of the system.

### Project tree

📦src
┣ 📂PrusaLinkWrapper
┃ ┣ 📂**pycache**
┃ ┗ 📜`__init__.py`
┣ 📂static

```
|  └📜firebaseConfig.js
├📂templates
|  └📜GUITemplate.html
├📜dhtDeployment.ino
├📜fanDeployment.ino
├📜info.py
├📜PrintGuardGUI.py
├📜RPi-Node1.py
└📜RPi-Node2.py
```

# Creating your own Print Guard

## Installation Instructions

To install the Print Guard system first clone the repository on two Raspberry Pis, this will allow you to run both the Arduino connections as well as run the livestream and GUI. The next thing will be to flash the two Arduinos with their respective code files for the fan and DHT sensors. Once the Arduinos are flashed, the system should be wired up according to the picture and schematic located below.

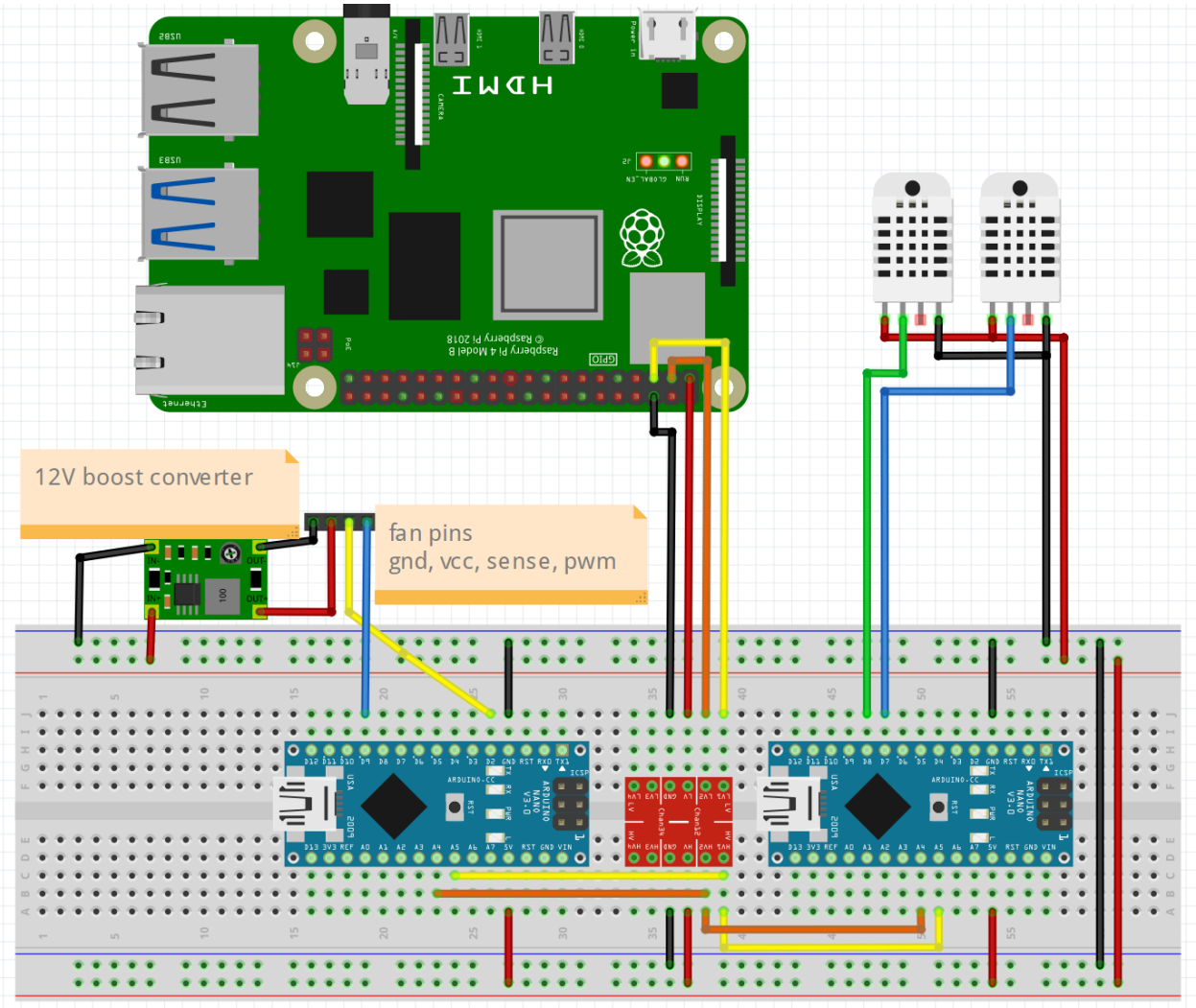
If using a Prusa printer (Mk4, Mini, or XL), you need to obtain the IP address located under `Settings > Network > IPv4 Address` and save it somewhere, you will also need to obtain the API key located under `Settings > Network > PrusaLink > Password` and save it as well. These will be used to setup the printer later on. If not using a Prusa printer, then as of right now printer control will not work, and the printer connect lines should be commented out.

The next part is setting up the firebase for real time database collection. This can be setup through the firebase console, the only change needed is to set the read and write rules to both be true. This is something that can be changed in the future to protect your data.

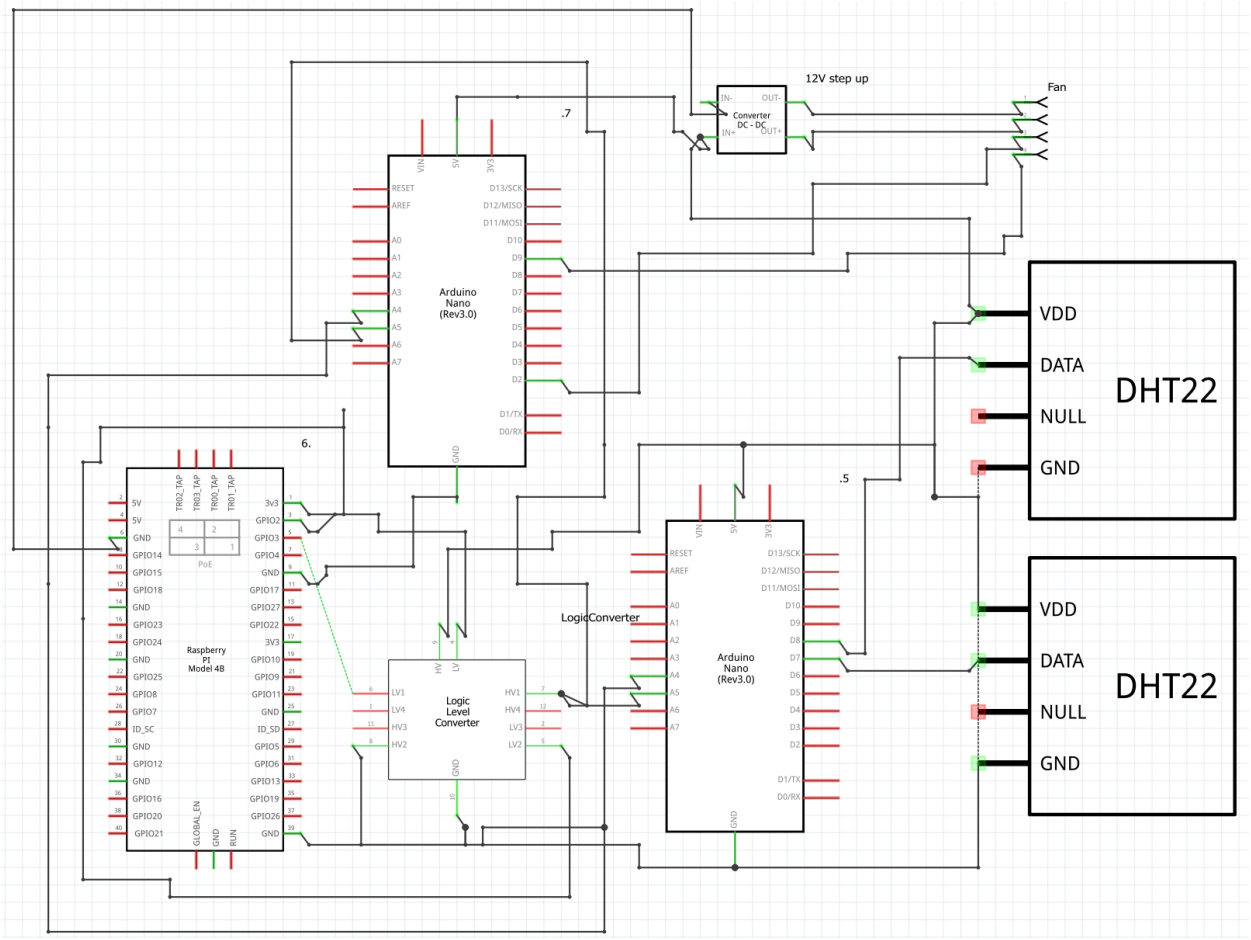
Finally the discord webhook has to be setup, to do you first need a server to setup the notifications on. On this server go to `Server settings > Integrations > Webhooks > New Webhook` you can then change the name and select the channel for that bot to send messages to. The Webhook can be copied here and will need to be saved for later.

### Wiring Diagrams

This is the breadboard view for the Print Guard wiring. Note that the Sensors can either be DHT11 or DHT22 which can be changed in the deployment code, and that the fan requires a 5V to 12V boost converter.



This is the schematic view of the above breadboard, it shows all the same connections with the same requirements.




## Hidden Files

The file structure located above shows two files that are not on the github page, which are `info.py` and `firebaseConfig.js`, this is because they have data specific to you. These files will have to be created manually, with `info.py` on both Raspberry Pis and `firebaseConfig.js` on the Raspberry Pi running the GUI.

`Info.py` should look like this:

printerIP = "PRINTER IPv4 ADDRESS"
printerKey = "PRINTER PASSWORD"
firebaseAPI = "YOUR API KEY"
firebaseAuth = "YOUR_PROJECT_ID.firebaseapp.com"
firebaseURL = "YOUR_DATABASE_URL"

```
firebaseStorage = "YOUR_PROJECT_ID.appspot.com"
DISCORD_WEBHOOK_URL = "YOUR DISCORD WEBHOOK"
```

`firebaseConfig.js` should look like this:

```
var firebaseConfig = {
   apiKey: "YOUR API KEY",
   authDomain: "YOUR_PROJECT_ID.firebaseapp.com",
   databaseURL: "YOUR_DATABASE_URL",
   storageBucket: "YOUR_PROJECT_ID.appspot.com"
};
```

# Run Instructions

Once everything is downloaded, flashed, wired up, and setup. The system can be run. The files can be run in any order but the best way to set it up would be to run:

1. `RPiNode1.py`
2. `RPiNode2.py`
3. `PrintGuardGUI.py`

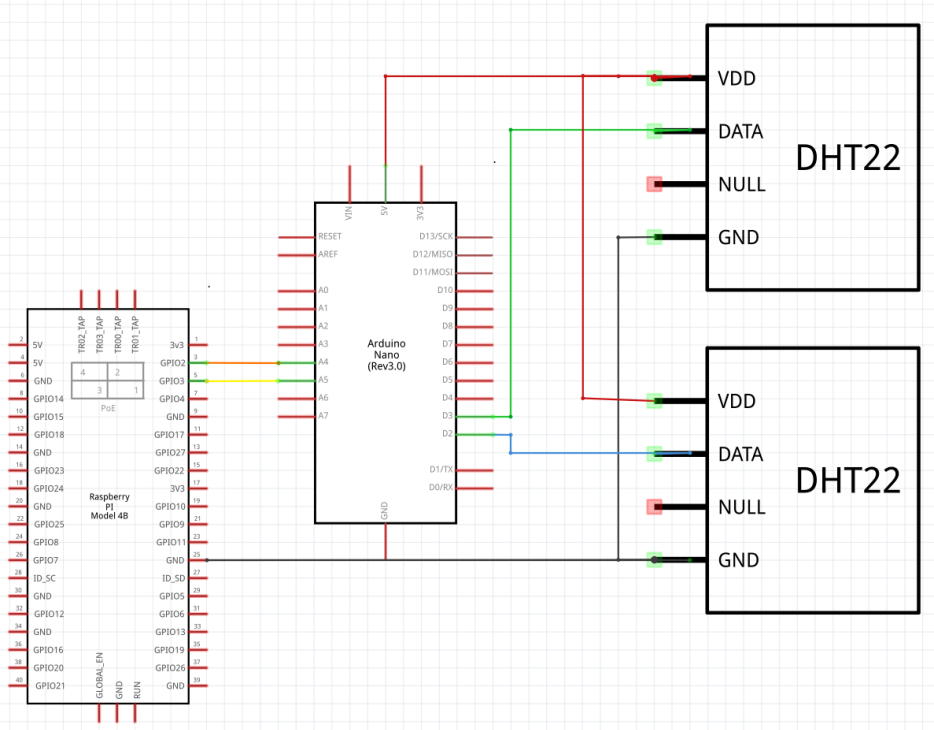
The GUI can be connected to locally on the Raspberry Pi by going to `http://127.0.0.1:5000` or remotely by going to `http://RASPBERRYPI-IP:5000` where RASPBERRYPI-IP is the ip address of teh Raspberry Pi running the GUI.

# Expected Results

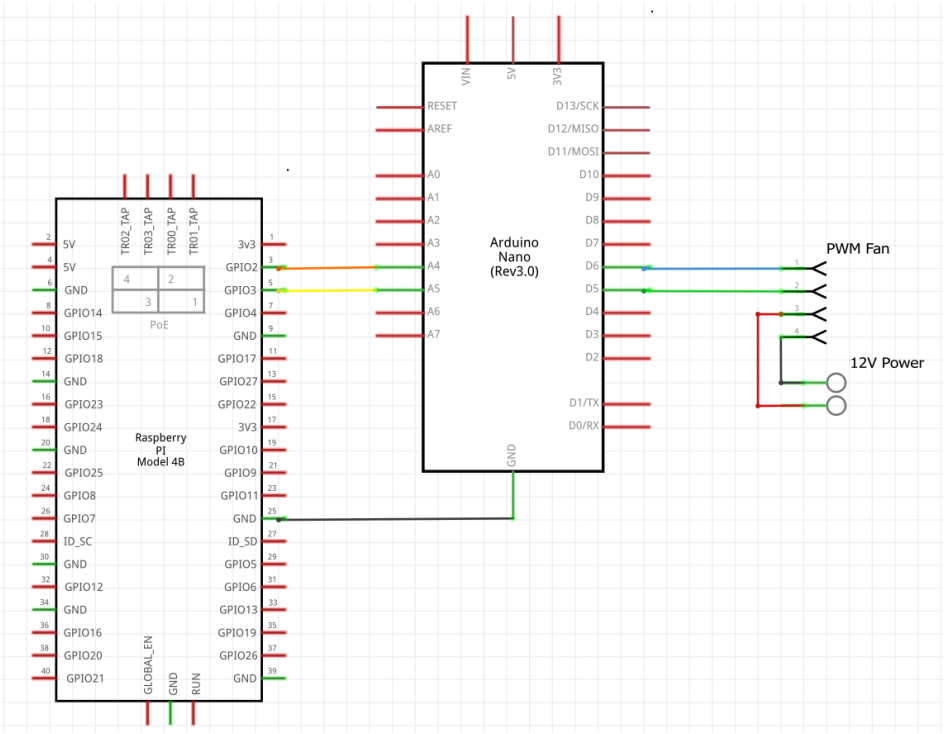
Once everything is running, the first result should be a discord notification informing you that the system is operational. The GUI is expected to show the sensor data for the printer nozzle and bed, the internal temperature and humidity, the external temperature and humidity, and the fan speed. It should also allow you to view the camera by clicking the camera icon. The default password for admin mode is L2-G9 which can be changed on line 137 in `GUITemplate.html`. Finally you should be able to stop and pause your print with the buttons on the GUI, this will only work if a print is running.

# Appendix B: Schematics

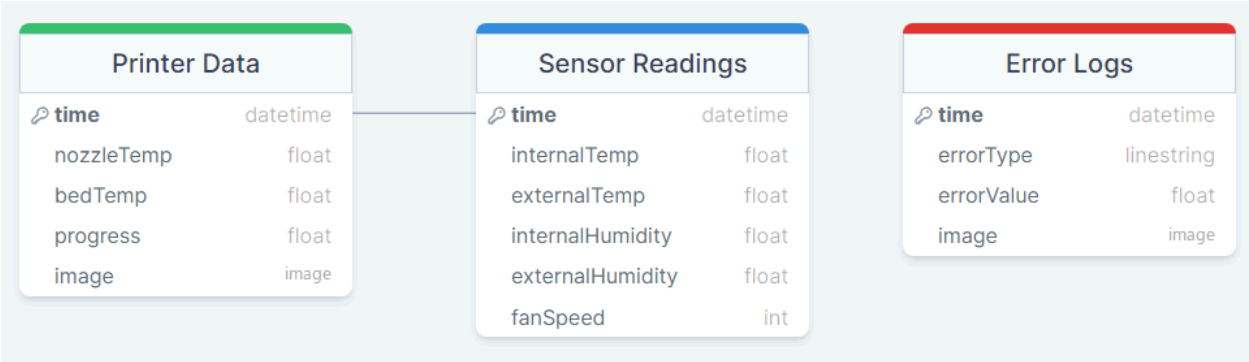
The figure below shows the DHT22 wiring with the Arduino and Raspberry pi




The figure below shows the schematic for the fan connected to the Arduino and Raspberry Pi

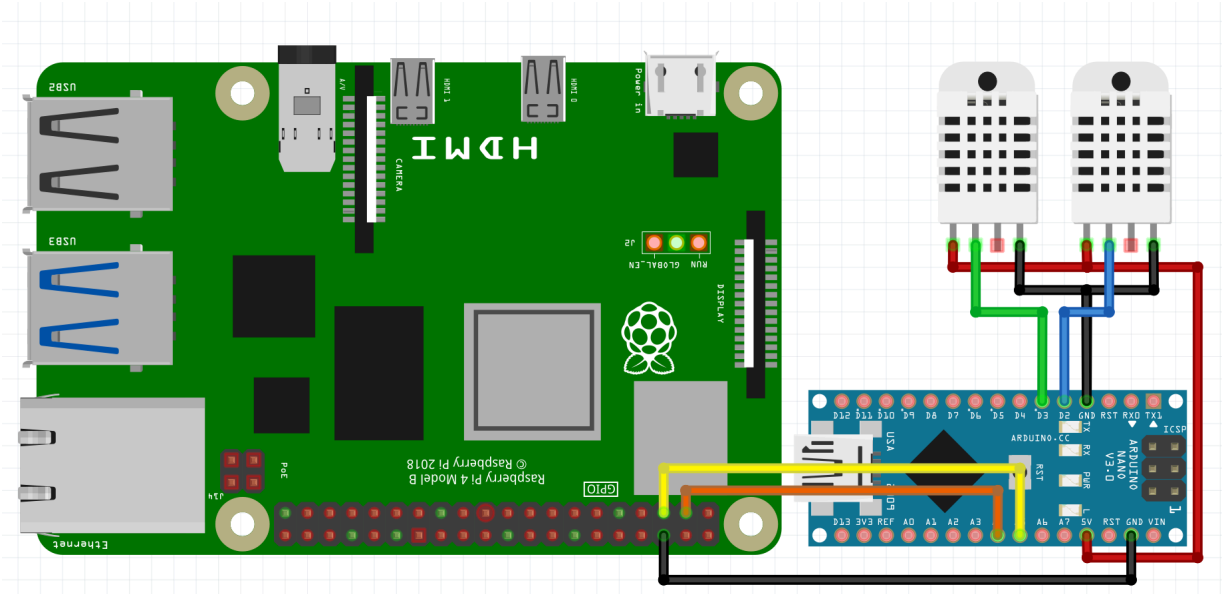



The figure below shows the schematic/design of the database table:

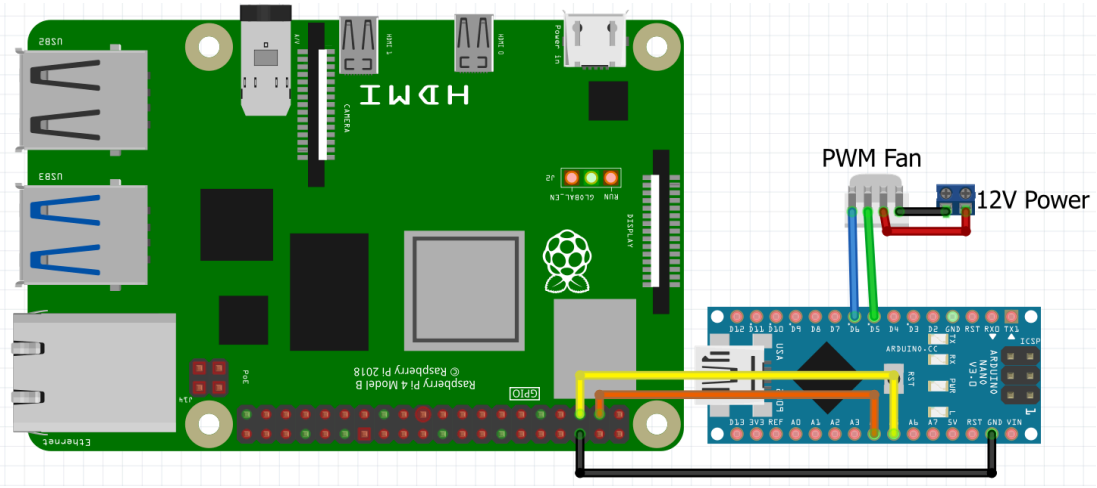



# Appendix C: Wiring Diagrams

The figure below shows the wiring diagram for the DHT22 temperature sensors connected to the Arduino and Raspberry Pi:

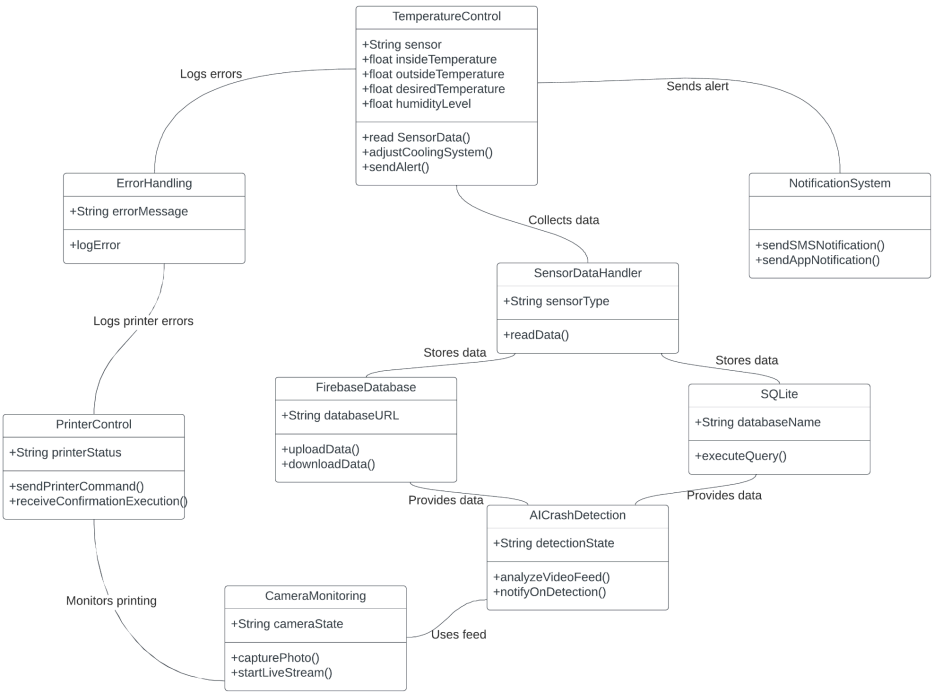


The figure below shows the Wiring diagram for the Fan connected to the Arduino and Raspberry Pi:




## Appendix D: Class Diagram

The UML diagram shown below visualizes a Print Guard system's modular design, illustrating the relationships and data flow between various components responsible for overseeing and managing printing operations.




# Appendix E: GUI Design Layout

The figure below shows an example of our GUI Design:


| Time | Nozzle Temp | Bed Temp | Internal Temp | External Temp |
|---|---|---|---|---|
| 12:05:00 | 111 | 30 | 22 | 21 |
| 12:06:15 | 184 | 34 | 23 | 21 |
| 12:08:00 | 186 | 37 | 25 | 21 |
| 12:05:45 | 187 | 41 | 25 | 21 |
| 12:06:00 | 200 | 43 | 26 | 21 |
| 12:06:15 | 215 | 46 | 25 | 21 |
| 12:06:30 | 217 | 50 | 26 | 21 |
| 12:06:45 | 214 | 53 | 27 | 22 |
| 12:07:00 | 218 | 57 | 29 | 22 |
| 12:07:15 | 215 | 59 | 30 | 22 |
| 12:07:30 | 216 | 58 | 31 | 22 |
| 12:07:45 | 214 | 60 | 33 | 22 |
| 12:08:00 | 218 | 61 | 34 | 23 |
| 12:08:15 | 213 | 60 | 33 | 21 |
| 12:08:30 | 215 | 60 | 32 | 21 |
| 12:08:45 | 215 | 59 | 31 | 21 |
| 12:09:00 | 216 | 60 | 31 | 21 |
| 12:09:15 | 214 | 60 | 32 | 21 |
| 12:09:30 | 215 | 60 | 33 | 21 |