Tool for Mycelium Grain Spawn Production – UPDATE

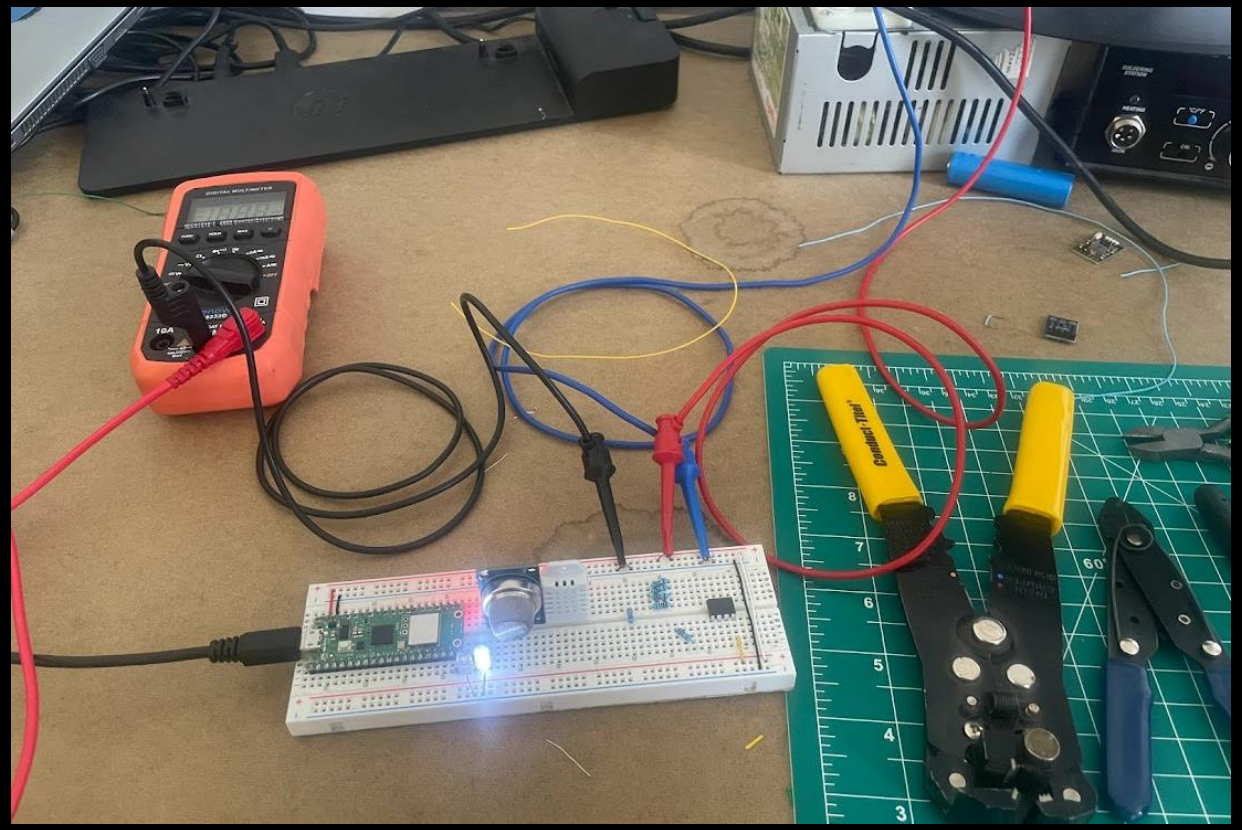
Isaac Peterson

Advisor: Dr. **Mitchell Neilsen**

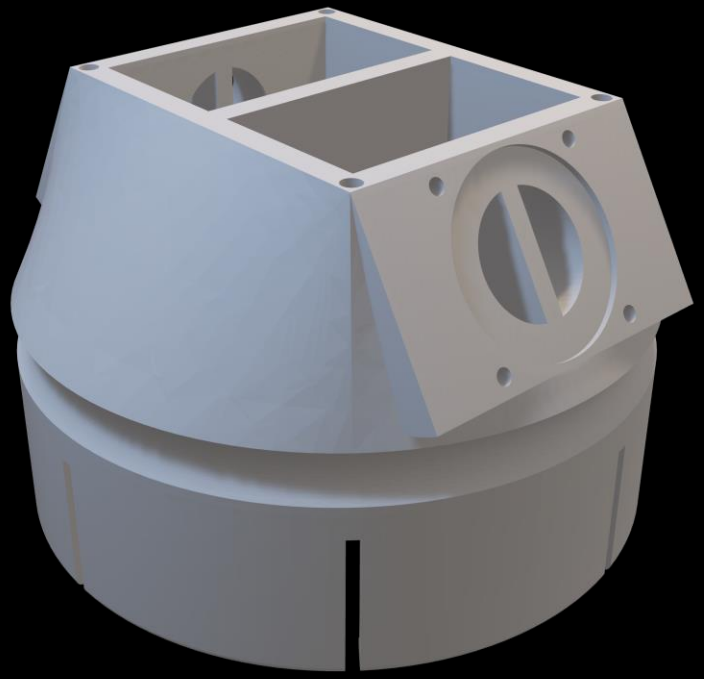
Minimum Viable Product (planned)

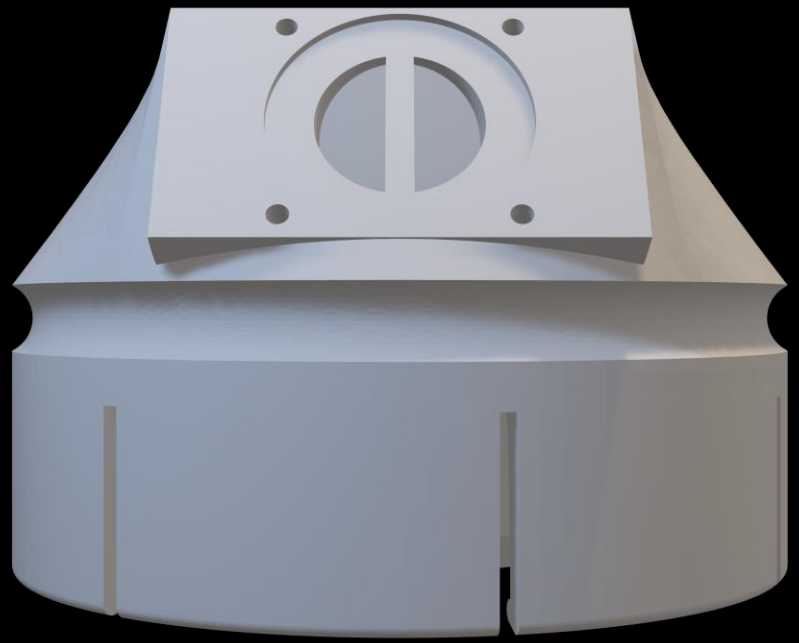
- Working hardware prototype
- FAE control based on sensor input
- Humidity control based on sensor input
- All sensor readings logged locally on microcontroller

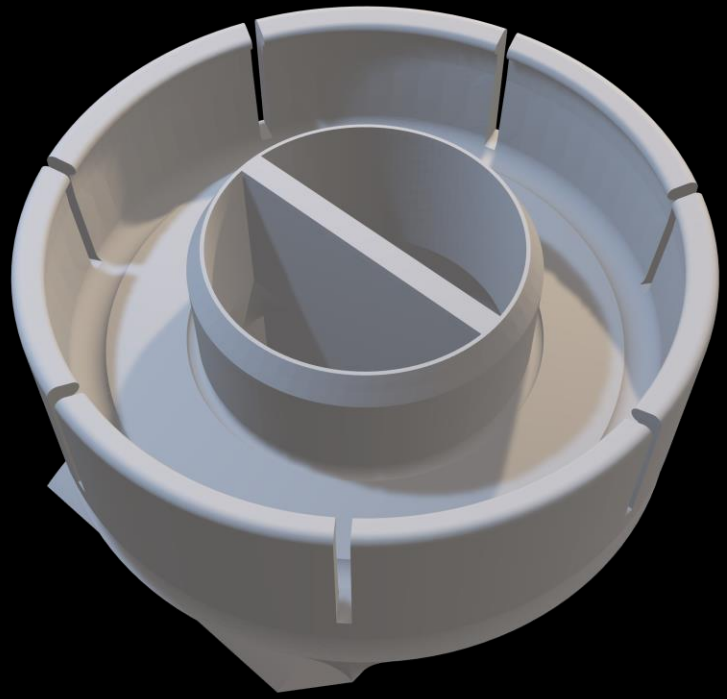
Hardware



Mechanical Design







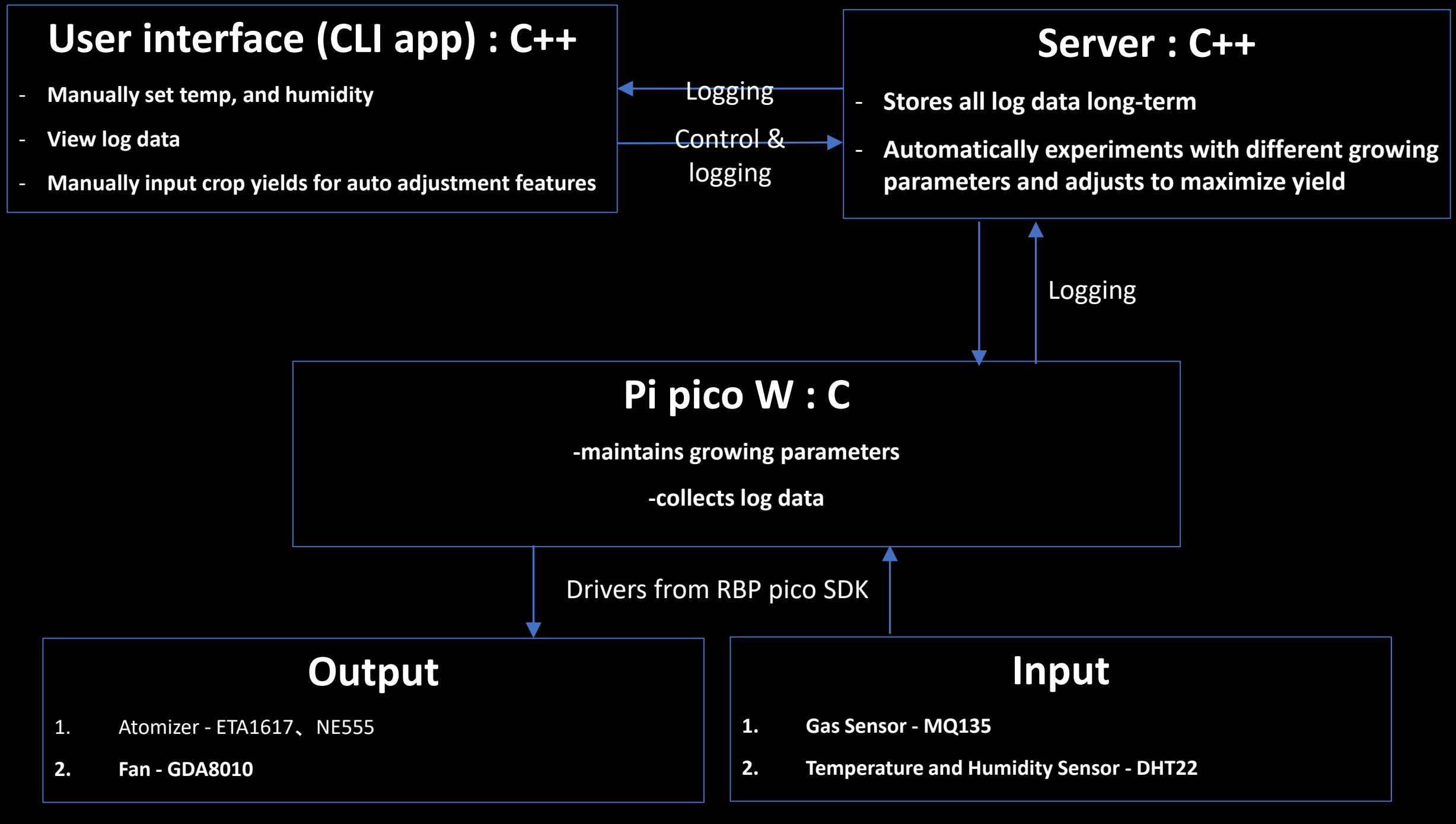
freecad.org

Software Design

- Raspberry Pi PICO Software Development Kit (SDK)
  - GCC
  - CMake
  - hardware_adc
  - Hardware_divider

```
161             if (err != ERR_OK) {
162                 DEBUG_printf("Failed to write data %d\n", err);
163                 return tcp_result(arg, -1);
164             }
165         }
166         return ERR_OK;
167     }
168
169     static bool tcp_client_open(void *arg) {
170         TCP_CLIENT_T *state = (TCP_CLIENT_T*)arg;
171         DEBUG_printf("Connecting to %s port %u\n", ip4addr_ntoa(&state->remote_addr), TCP_PORT);
172         state->tcp_pcb = tcp_new_ip_type(IP_GET_TYPE(&state->remote_addr));
173         if (!state->tcp_pcb) {
174             DEBUG_printf("failed to create pcb\n");
175             return false;
176         }
177
178         tcp_arg(state->tcp_pcb, state);
179         tcp_poll(state->tcp_pcb, tcp_client_poll, POLL_TIME_S * 2);
180         tcp_sent(state->tcp_pcb, tcp_client_sent);
181         tcp_recv(state->tcp_pcb, tcp_client_recv);
182         tcp_err(state->tcp_pcb, tcp_client_err);
183
184         state->buffer_len = 0;
185
186         // cyw43_arch_lwip_begin/end should be used around calls into lwIP to ensure correct locking.
187         // You can omit them if you are in a callback from lwIP. Note that when using pico_cyw_arch_poll
188         // these calls are a no-op and can be omitted, but it is a good practice to use them in
189         // case you switch the cyw43_arch type later.
190         cyw43_arch_lwip_begin();
191         err_t err = tcp_connect(state->tcp_pcb, &state->remote_addr, TCP_PORT, tcp_client_connected);
192         cyw43_arch_lwip_end();
193
194         return err == ERR_OK;
195     }
196
197     // Perform initialisation
198     static TCP_CLIENT_T* tcp_client_init(void) {
199         TCP_CLIENT_T *state = calloc(1, sizeof(TCP_CLIENT_T));
```

https://github.com/raspberrypi/pico-examples


User interface (CLI app) : C++
- Manually set temp, and humidity
- View log data
- Manually input crop yields for auto adjustment features
Logging
Control & logging
Server : C++
- Stores all log data long-term
- Automatically experiments with different growing parameters and adjusts to maximize yield
Logging
Pi pico W : C
-maintains growing parameters
-collects log data
Drivers from RBP pico SDK
Output
1. Atomizer - ETA1617、NE555
2. Fan - GDA8010
Input
1. Gas Sensor - MQ135
2. Temperature and Humidity Sensor - DHT22

[DEMO]

What’s next

- Wrap up hardware design
  - Sensor housing
  - Breadboard 😫 >>----> PCB 😎
- Linux host app
  - Wireless communication with multiple devices (TCP)

Questions?

- Hardware
- PICO SDK
- I/O
- Project Timeline