# FLUTTER APP FUNCTIONALITY EXPANSION

Open Acidification Tank Controller

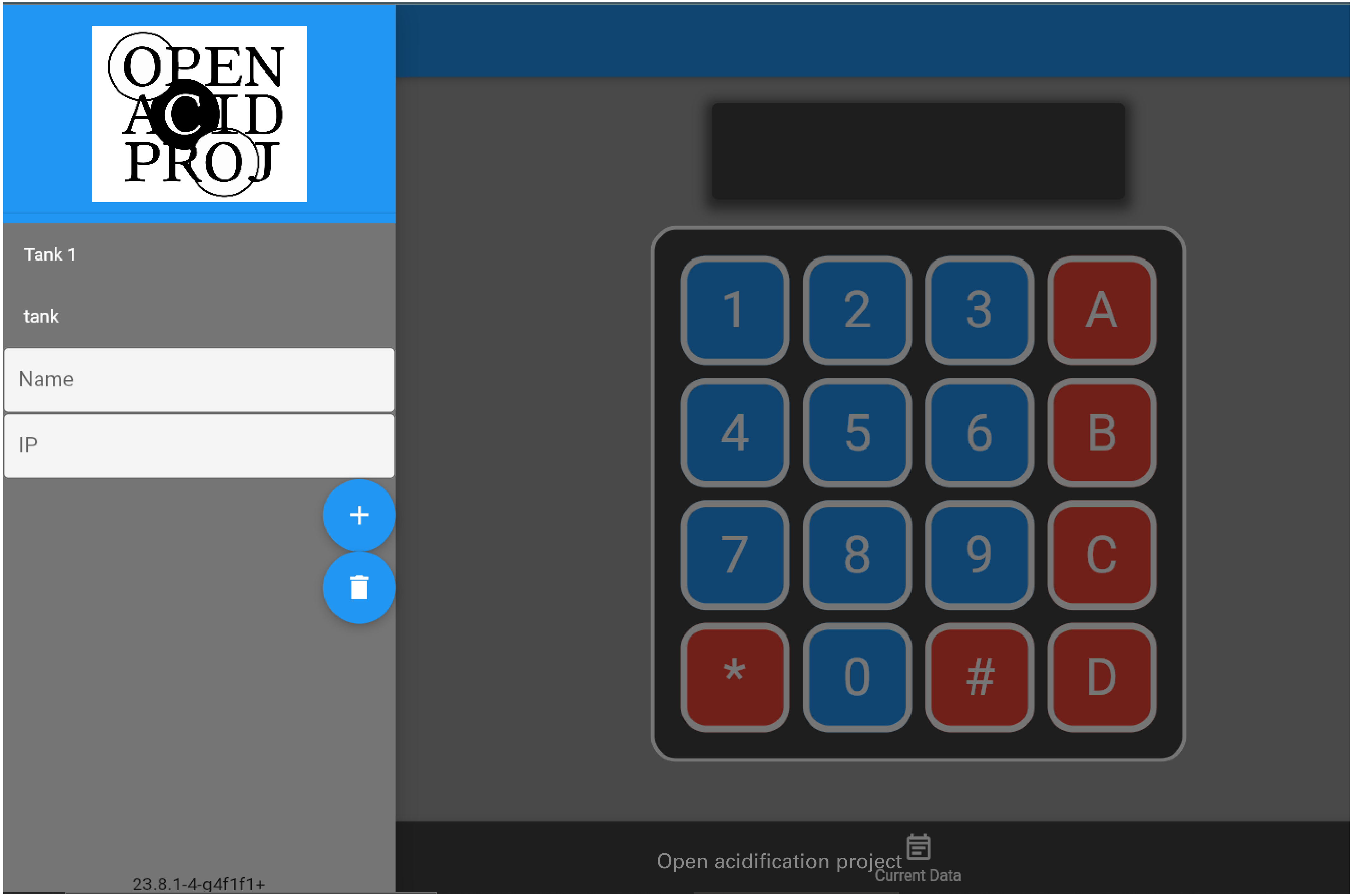

Lydia Thiel

# Background

This tank controller can measure and change temperature and pH of the water in tanks used to measure the effect of ocean acidification on sea animals, specifically octopuses.

Developed by Dr. Onthank, and now an expanding group of people, as the alternative is much more expensive.

Online Flutter app accesses the same things for convenience & to change from a distance.


OPEN ACID PROJ
Tank 1
tank
Name
IP
1 2 3 A
4 5 6 B
7 8 9 C
* 0 # D
23.8.1-4-g4f1f1+
Open acidification project
Current Data

# TOOLS USED

LANGUAGE: DART, C++

VERSION CONTROL: GITHUB

VISUAL STUDIO CODE

ARDUINO IDE, POSTMAN

# Original -> Final Goals

Focus narrowing-

- Temperature sine wave, mobile first design already completed
- Focus shifts to PUT Data values, active tests
- Handling PUT requests with different values to the server

TankController > src > wrappers > EthernetServer_TC.cpp > put()

```cpp
void EthernetServer_TC::put() {
    Therm_SinePeriodHours
  } var;
  if (memcmp_P(buffer + 4, F("/api/1/data?Kd="), 15) == 0) {
    var = Kd;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?Ki="), 15) == 0) {
    var = Ki;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?Kp="), 15) == 0) {
    var = Kp;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?PID="), 16) == 0) {
    var = PID;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?TankID="), 19) == 0) {
    var = TankID;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?Target_pH="), 22) == 0) {
    var = Target_pH;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?HeatOrChill="), 24) == 0) {
    var = HeatOrChill;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?pH_RampHours="), 25) == 0) {
    var = pH_RampHours;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?Therm_RampHours="), 28) == 0) {
    var = Therm_RampHours;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?pH_SineAmplitude="), 29) == 0) {
    var = pH_SineAmplitude;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?TargetTemperature="), 30) == 0) {
    var = TargetTemperature;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?pH_SinePeriodHours="), 31) == 0) {
    var = pH_SinePeriodHours;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?Therm_SineAmplitude="), 32) == 0) {
    var = Therm_SineAmplitude;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?GoogleSheetInterval="), 32) == 0) {
    var = GoogleSheetInterval;
  } else if (memcmp_P(buffer + 4, F("/api/1/data?Therm_SinePeriodHours="), 34) == 0) {
    var = Therm_SinePeriodHours;
  } else {
    serial(F("put \"%s\" not recognized!"), buffer + 5);
    sendResponse(HTTP_BAD_REQUEST);
```

pH_mobile* ⊗ 6 ⚠ 1 ⓘ 5 0 Debug my code Ln 172, Col 29

0

14

Submit new Target_pH value

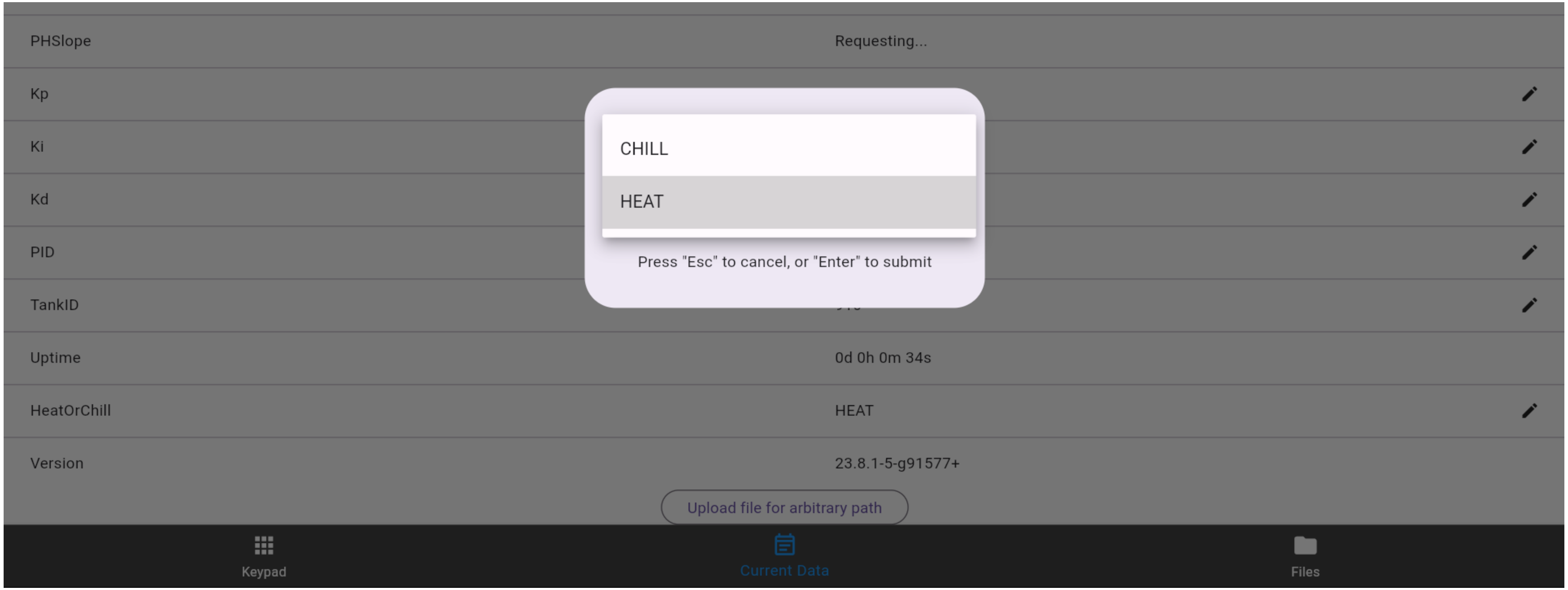
7

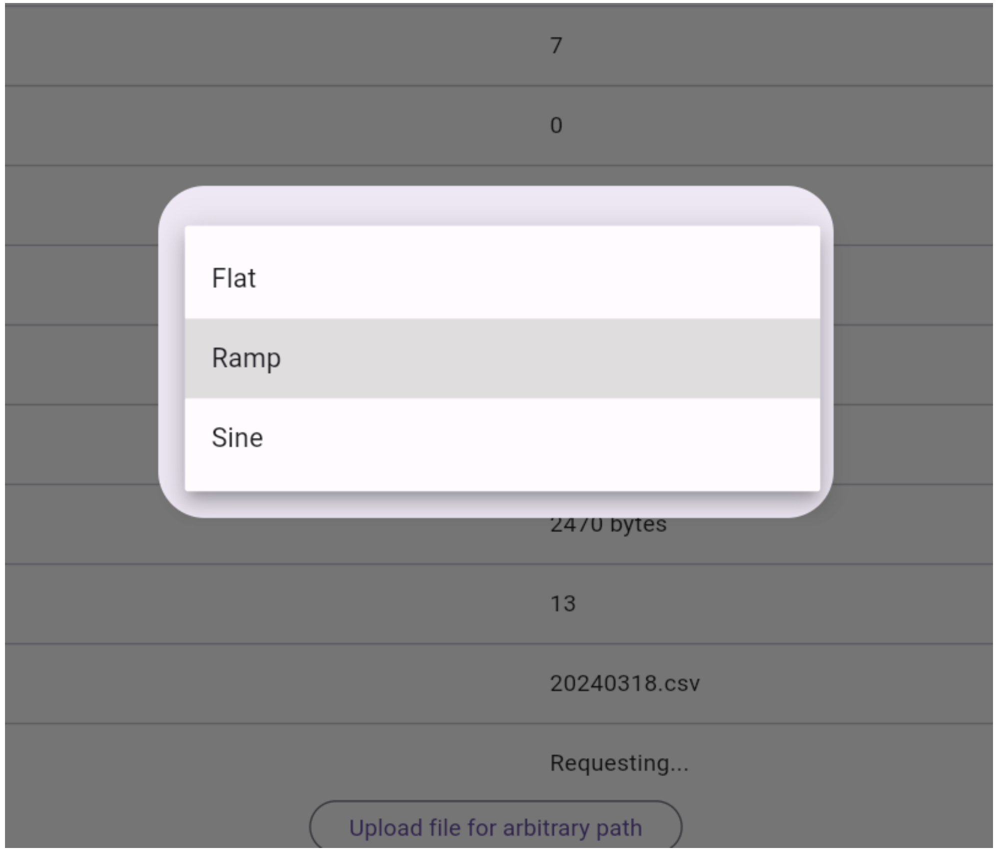
Press "Esc" to cancel, or "Enter" to submit

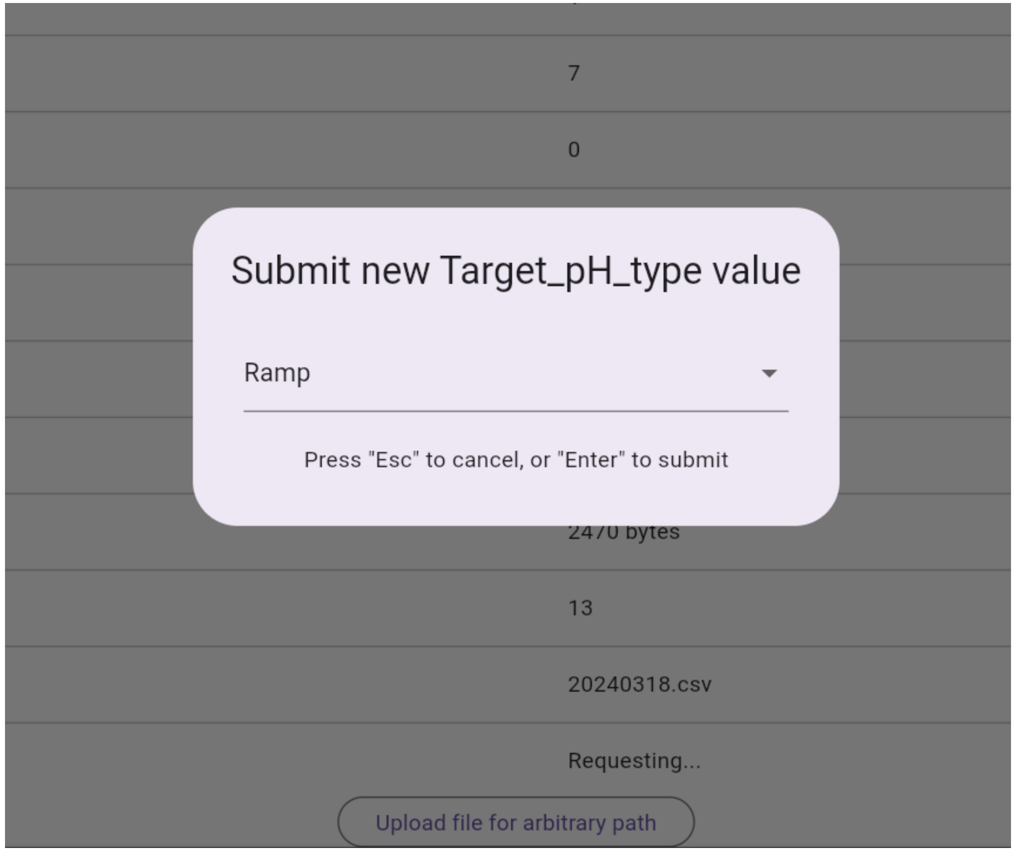
20240318.csv

Requesting...

90


PHSlope
Requesting...
Kp
Ki
Kd
PID
TankID
Uptime
0d 0h 0m 34s
HeatOrChill
HEAT
Version
23.8.1-5-g91577+
CHILL
HEAT
Press "Esc" to cancel, or "Enter" to submit
Upload file for arbitrary path
Keypad
Current Data
Files




# WINTER QUARTER PROPOSED METHOD

Tank Manager: Tank 1

| | |
|---|---|
| pH | 0 |
| Temperature | 0 |
| pH_FunctionType | RAMP |
| Target_pH | 3 |
| pH_RampHours | 8 |
| pH_SinePeriodHours | 0 |
| pH_SineAmplitude | 0 |
| Therm_FunctionType | SINE |
| TargetTemperature | 18 |
| Therm_RampHours | 0 |
| Therm_SinePeriodHours | 9 |
| Therm_SineAmplitude | 6 |

Upload file for arbitrary path

Keypad Current Data Files

TankController > src > model > JSONBuilder.cpp > buildCurrentValues()

```
int JSONBuilder::buildCurrentValues() {
  bytes = snprintf_P(buffer, BUFFER_SIZE,
                     (PGM_P)F("{"
                              "\"pH\":%i.%i,"
                              "\"Temperature\":%i.%i,"
                              "\"pH_FunctionType\":\"%s\","
                              "\"Target_pH\":%i.%i,"
                              "\"pH_RampHours\":%i.%i,"
                              "\"pH_SinePeriodHours\":%i.%i,"
                              "\"pH_SineAmplitude\":%i.%i,"
                              "\"Therm_FunctionType\":\"%s\","
                              "\"TargetTemperature\":%i.%i,"
                              "\"Therm_RampHours\":%i.%i,"
                              "\"Therm_SinePeriodHours\":%i.%i,"
                              "\"Therm_SineAmplitude\":%i.%i,"
                              "\"IPAddress\":\"%d.%d.%d.%d\","
                              "\"MAC\":\"%02X:%02X:%02X:%02X:%02X:%02X\","
                              "\"FreeMemory\":\"%i bytes\","
                              "\"GoogleSheetInterval\":%i,"
                              "\"LogFile\":\"%s\","
                              "\"PHSlope\":\"%s\","
                              "\"Kp\":%i.%i,"
                              "\"Ki\":%i.%i,"
                              "\"Kd\":%i.%i,"
                              "\"PID\":\"%s\","
                              "\"TankID\":%i,"
                              "\"Uptime\":\"%id %ih %im %is\","
                              "\"HeatOrChill\":\"%s\","
                              "\"Version\":\"%s\","
                              "\"EditableFields\":["  // List of editable fields follows
                              "\"Target_pH\","
                              "\"TargetTemperature\","
                              "\"GoogleSheetInterval\","
                              "\"Kp\","
                              "\"Ki\","
                              "\"Kd\","
```

pH_mobile* ⊗ 7 ⚠ 1 ⓘ 5 0 Debug my code

```
unittest(PUT_Heat_Mode) {
  assertEqual(0, EEPROM_TC::instance()->getHeat());

  EthernetServer_TC* server = EthernetServer_TC::instance();
  server->setHasClientCalling(true);
  delay(1);
  server->loop();
  EthernetClient_CI client = server->getClient();
  const char request[] =
      "PUT /api/1/data?HeatOrChill=HEAT HTTP/1.1\r\n"
      "Host: localhost:80\r\n"
      "Accept: text/plain;charset=UTF-8\r\n"
      "Accept-Encoding: identity\r\n"
      "Accept-Language: en-US\r\n"
      "\r\n";
  client.pushToReadBuffer(request);
  server->loop();
  deque<uint8_t>* pBuffer = client.writeBuffer();
  assertEqual(81, pBuffer->size());
  String response;
  while (!pBuffer->empty()) {
    response.concat(pBuffer->front());
    pBuffer->pop_front();
  }
  const char expectedResponse[] =
      "HTTP/1.1 303 See Other\r\n"
      "Location: /api/1/data\r\n"
      "Access-Control-Allow-Origin: *\r\n"
      "\r\n";
  assertEqual(expectedResponse, response);
```

# LOGISTICS

- Creating new tests for new functionality (server side tests)
- Keeping old tests up to date with new changes (dart testing)
- Familiarization with code base, running tests, saving and downloading new versions of code via Arduino IDE
- Debugging, use of Postman to test HTTP PUT requests

# MY TAKEAWAY

- Working with existing project
- Broad implementation, working with many different functionalities
- Experience with HTTP requests, Arduino, Flutter, C++
- Debugging skills and habits

# THANK YOU

Lydia Thiel