Pet Health Monitor

SHAMEEMAH FUSEINI-CODJOE

Overview

For this project, my goal was to gain an understanding of how to build a digital ecosystem using a Photon 2, input sensors, local computation, cloud services, and various output devices. Given that I have no experience whatsoever with electronics, I chose to pursue this project at the Platypus level by pushing myself to explore a sensor outside of what was provided in our kits with minimal to no assistance.

Problem Statement

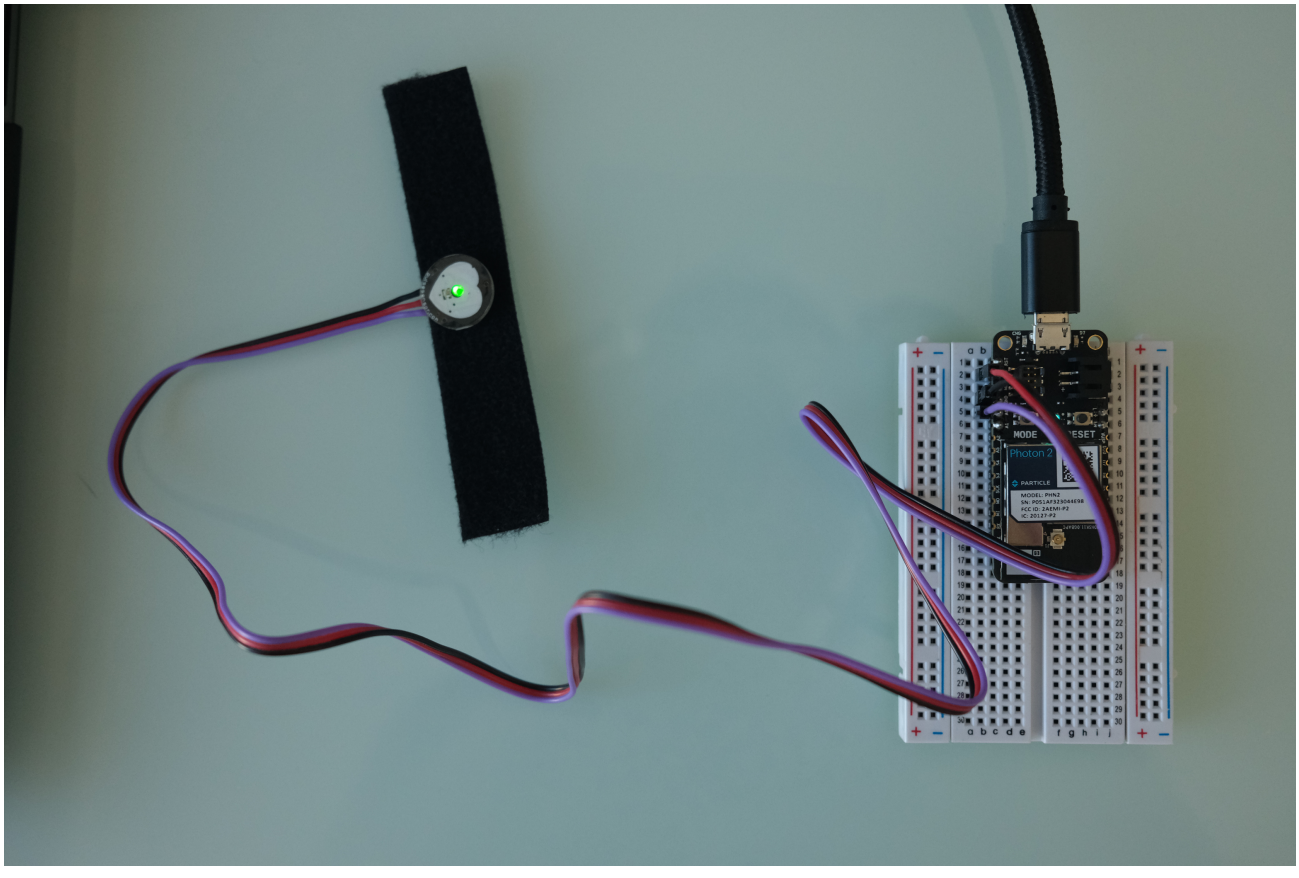
For many pet owners, it is difficult to tell whether your pet is in good health or not until they start exhibiting symptoms of illness, or until a routine vet visit reveals some underlying issues. What if there's a way that we can continuously monitor a pet 's health at home based on their vitals?

Contributions and Reflections

For this project, the goal was to use a pulse sensor to track a pet's pulse rate, calculate Beats Per Minute (BPM) based on the pulse rate, and to alert the owner if the BPM falls outside of the normal BPM for that animal. My contribution to the project was to perform all the exploration related to the pulse sensor, i.e., getting the pulse sensor to work, getting accurate readings from the sensor, calculating BPM, and finally, publishing the BPM to the Particle Device Cloud for consumption by the output devices.

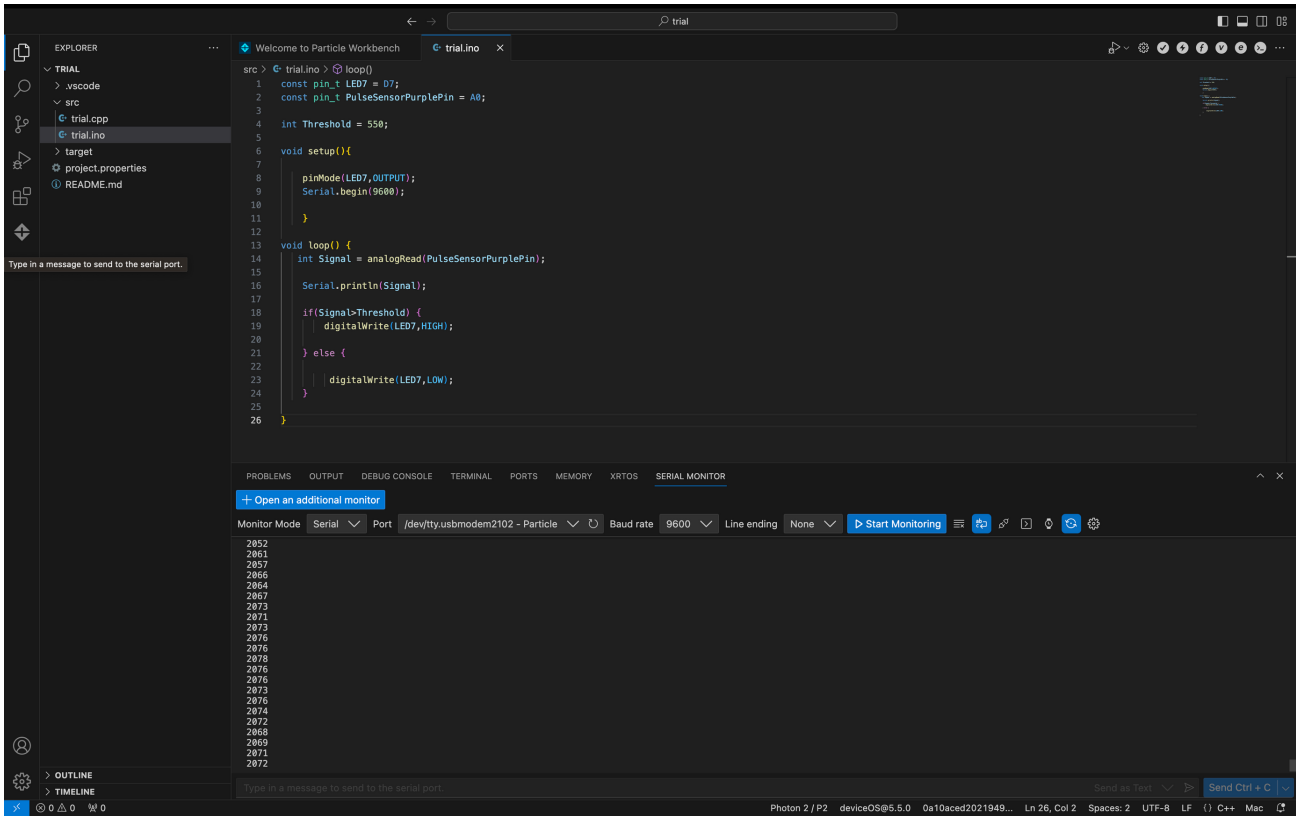
I started off by connecting the pulse sensor to my Photon. The circuit I used for the pulse sensor was incredibly simple. I connected my Photon 2 to a breadboard and made the following connections:

- Black wire from the pulse sensor to the Photon ground pin
- Red wire from pulse sensor to the Photon 3.3v pin
- Purple wire from pulse sensor to the Photon A0 pin



Pulse sensor successfully connected

Following the successful connection, I wrote some code to blink the built-in D7 LED on the Photon 2 to the rhythm of my pulse. Due to the sensitivity of the sensor, I had to play around with the threshold value in order to get the best reading for my pulse. This required some research into understanding how the resolution of the ADC inputs affects the resting signal of the pulse sensor. After several iterations, I managed to get this working successfully and started to get some accurate readings from the sensor.



Reading pulse rate values from the pulse sensor

Next, I moved on to the computation of the BPM. To do this, I wrote some code to count the number of beats received from the pulse sensor in 10 seconds, and then multiplied this value by 6 to get the BPM reading.

src > PulseSensor.cpp > counted

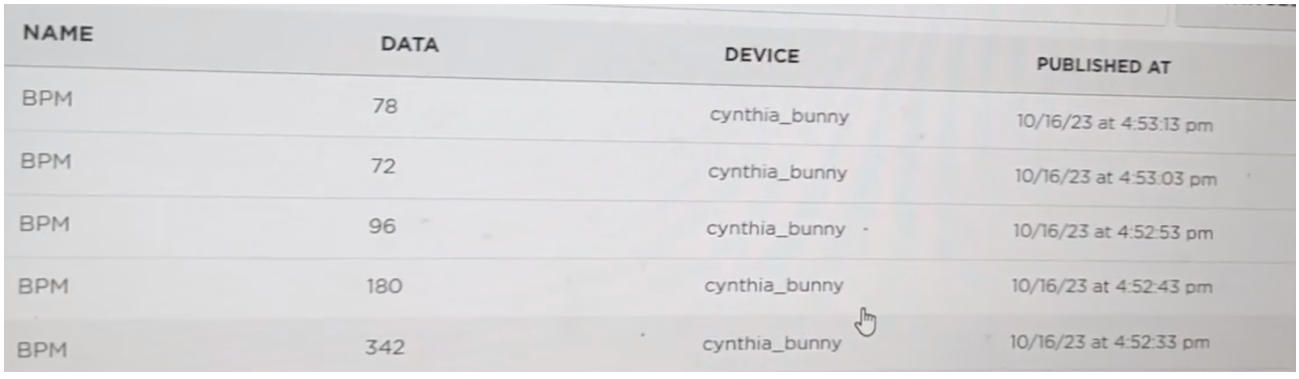
```
19
20  void setup(){
21
22      pinMode(LED7,OUTPUT);
23      Serial.begin(9600);
24
25      }
26
27  void loop() {
28
29    startTime = millis();
30
31    while (millis()<startTime+10000)
32    {
33      sensorValue = analogRead(PulseSensorPurplePin);
34
35      if (sensorValue > threshold && counted == false)
36      {
37        count++;
38
39        Serial.print ("count = ");
40        Serial.println (count);
41
42        digitalWrite (LED7,HIGH);
43        delay (50);
44        digitalWrite (LED7, LOW);
45
46        counted = true;
47      } else if (sensorValue < threshold)
48      {
49        counted = false;
50        digitalWrite (LED7, LOW);
51      }
52
53
54    }
55
56    heartRate = count*6;                          // Multiply the count by 6 to get beats per minute
57
58    Serial.println ();
59    Serial.print ("BPM = ");
60    Serial.println (heartRate);                   // Display BPM in the Serial Monitor
61    Serial.println ();
62
63    count = 0;
64
65  }
```

Calculating BPM using pulse rate from pulse sensor

Finally, I passed the BPM value to the Particle Device Cloud using the publish() function, and the BPM was available for consumption by our output devices!

```
37      }                                           // Multiply the count
38      heartRate = count*6;
39      Serial.println ();
40      Serial.print ("BPM = ");                    // Display BPM in the
41      Serial.println (heartRate);
42      Serial.println ();
43      count = 0;
44      Particle.publish("BPM", String::format(String(heartRate)));
45  }
```

Publishing BPM to Particle Device Cloud using publish() function



NAME	DATA	DEVICE	PUBLISHED AT
BPM	78	cynthia_bunny	10/16/23 at 4:53:13 pm
BPM	72	cynthia_bunny	10/16/23 at 4:53:03 pm
BPM	96	cynthia_bunny	10/16/23 at 4:52:53 pm
BPM	180	cynthia_bunny	10/16/23 at 4:52:43 pm
BPM	342	cynthia_bunny	10/16/23 at 4:52:33 pm

BPM showing up in Particle Device Cloud

Through this exploration, I learned:

- how to connect a pulse sensor to the Photon 2.
- how to read the signals from a pulse sensor.
- how to use the data received from an input sensor to derive further information through computation.
- how to get two Photons to communicate with each other.
- how to publish data to the Particle Device Cloud.
- how to develop in VS Code for the Photon.
- how to work virtually with a team using collaborative tools such as Slack and Figma.

Speculations

As we move into the era of AI-powered systems, I believe that integrating Machine Learning with the digital ecosystem could be incredibly powerful. For this specific project, I imagine that we could use the pulse sensor to collect a ton of data from different animal species in varying degrees of health. With this data, we could then train a machine learning model to be able to predict a pet's health status based on their live pulse rate. We could also incorporate a dashboard that allows owners to see their pet's pulse data and track their health status based on the predictions of the model.

Conclusions and Next Steps

This project granted me the opportunity to develop an understanding of electronics and the digital ecosystem, and how I can incorporate these technologies into my design practice. Given the time constraints of this project, I did not have the ability to delve as deeply as I would have liked. For next steps, I am hoping to test our existing project on a pet, perform some further explorations that allow me to experiment with other types of sensors and output devices, create more complex circuits, and integrate machine learning into this project.

Appendix

- Pet Health Monitor YouTube Video.
- Pet Health Monitor pulse sensor project files.

Thank you!