4th Year IoT Project  
Aquarium Monitoring System: HCFF

Julien Rose, Achille Bayart, Nicolas Rigaudy

Table of contents

[Introduction and context 3](#_Toc43373630)

[Completed work 3](#_Toc43373631)

[Hardware 3](#_Toc43373632)

[Raspberry Pi 3](#_Toc43373633)

[Temperature 3](#_Toc43373634)

[Brightness 3](#_Toc43373635)

[Image Recognition 3](#_Toc43373636)

[Initial Idea 3](#_Toc43373637)

[OpenCv 3](#_Toc43373638)

[Android App 4](#_Toc43373639)

[Figma 4](#_Toc43373640)

[Android Studio 4](#_Toc43373641)

[Data collection and linking 4](#_Toc43373642)

[Rest Client 4](#_Toc43373643)

[Project perspective 4](#_Toc43373644)

[Planned additions 4](#_Toc43373645)

[Challenges 4](#_Toc43373646)

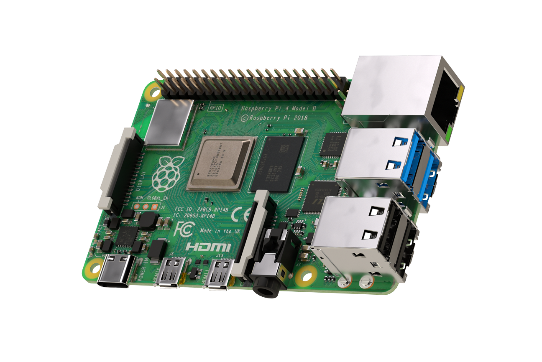
[Conclusion 4](#_Toc43373647)

# Introduction and context

# Completed work

## Hardware

### Raspberry Pi

The Raspberry Pi is a nanocomputer.

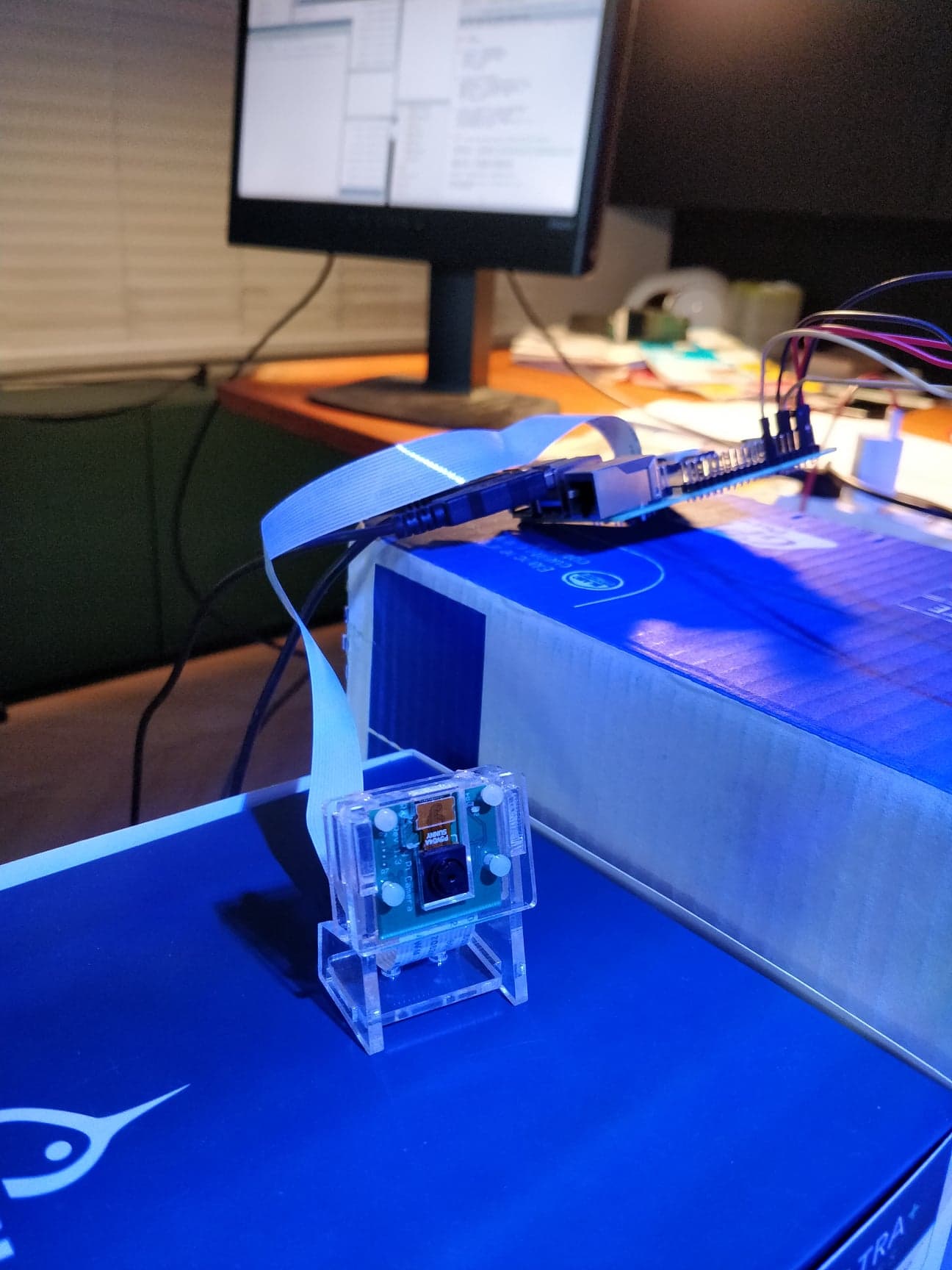
We decided to choose the RPi 4 Model B because this is the last model of Raspberry Pi and it fits perfectly with what we want to do. A random-access memory of 2 GB is good enought that's why we didn't choose more.

*Raspberry Pi 4 Model B*

We installed Raspbian (Raspberry Pi OS) which is a free operating system based on Debian.

To have the best experience as possible, we connected a lot of things to the Raspberry Pi as :

* **A power supply USB type-C**, it’s inevitable if we want the raspberry pi to work.
* **A microSD card**, the nanocomputer needs an SD card to store all its files and the Raspbian operating system. We need a microSD card with a capacity of at least 8 GB, we selected a microSD of 64 GB to be safe.
* **A computer screen**, to view the Raspbian desktop environment, we need a screen and a cable to link the screen and the pi. (We use a standard HDMI to HDMI cable plus a micro HDMI to HDMI adapter, to connect Raspbeery Pi 4 to the screen)
* **An USB keyboard and an USB mouse.**



Moreover, we bought a camera module B01 especially for the Raspberry Pi. In fact, we absolutely need it for fishes recognition with OpenCV.

We had some problems to connect cameras to the Raspberry Pi. Indeed, we had the possibility to use a Withings or Samsung camera but it was impossible to connect them. My father uses these cameras to film the aquarium when he's away, so, they are connected to an application. They are maybe too secure to be used as we wish.

That’s why we decided to buy a camera module especially for the Raspberry Pi, it’s useful and not very expensive.

*Camera module B01*

### Temperature

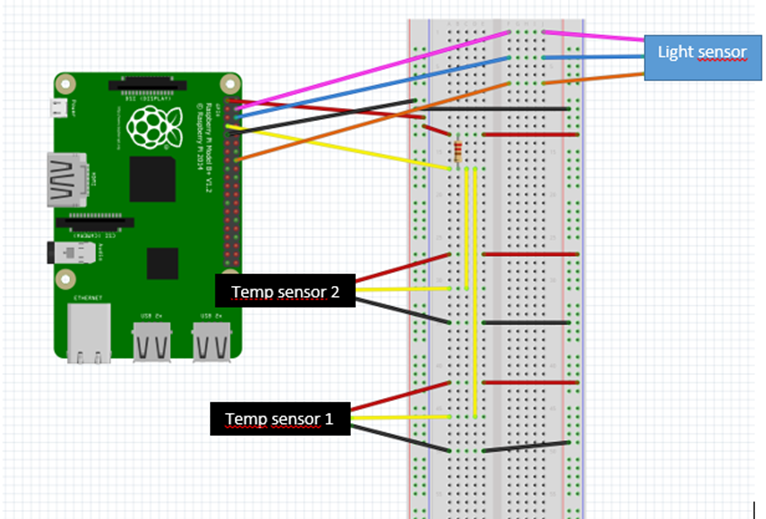
We wanted to measure the aquarium’s temperaure thanks to sensors.

We decided to choose the DS18B20 sensor which is very common. This sensor has a measurement range from -55°C to +125°C and it is waterproof.

We started our test with only one temperature sensor. After that, we wanted to save the temperature values in a text file. We managed it without too much trouble.

*DS18B20 sensor*

We added another temperature sensor to the breadboard as you can see:



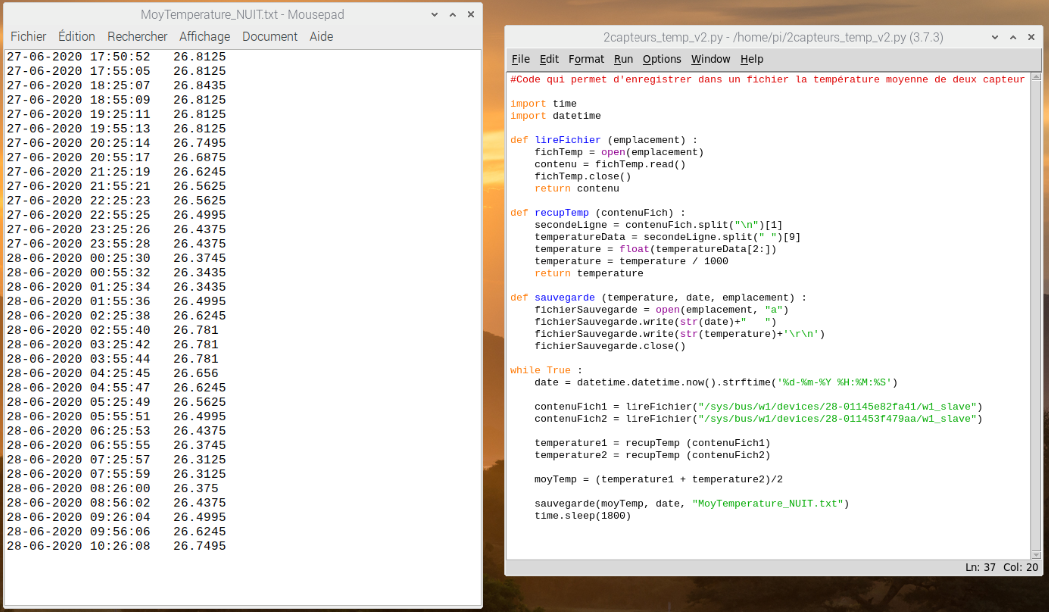
*The connection diagram made on Fritzing*

We used a resistor of 4,7 kΩ in this connection and you have to connect:

* The red wire to the pin 1 (3,3V power)
* The yellow wire to the pin 7 (GPIO 4)
* The black wire to the pin 9 (Ground)

That installation permited us to have the average temperature of 2 sensors. So, the temperature is more precise.

The python code which saves the average temperature in a text file is below:



*Python code which saves the average temperature in the text file « MoyTemperature\_NUIT »*

The average temperature is saved every 30 minutes (1800 seconds). As you can see on the previous picture, we let the sensors in the aquarium during one night. We have a difference of 0,5°C and the temperature is around 26,5°C. This is excatly what we expected.



*Taking temperature from the aquarium during the night*

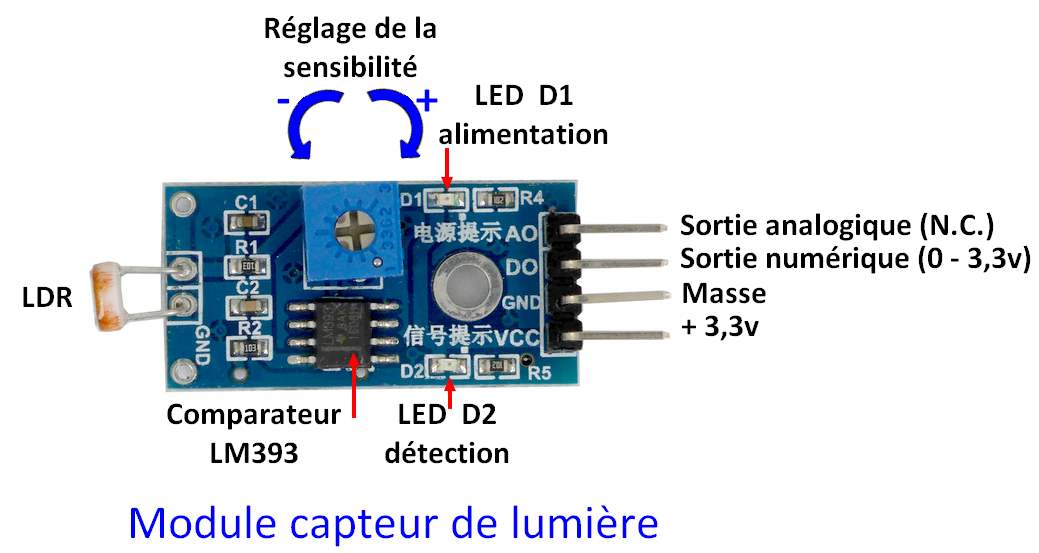
### Brightness

We decided to add another sensor to the Raspberry Pi : the LDR light detector module. (LDR = Light Dependant Resistor)

First of all, it is possible to adjust the sensitivity of this sensor. It works as follows:

* Light intensity < limit: output = 0
* Light intensity > limit: output = 1

**Sensitivity adjustment**



**Analog output**

**Digital output**

**Ground**

**+5V**

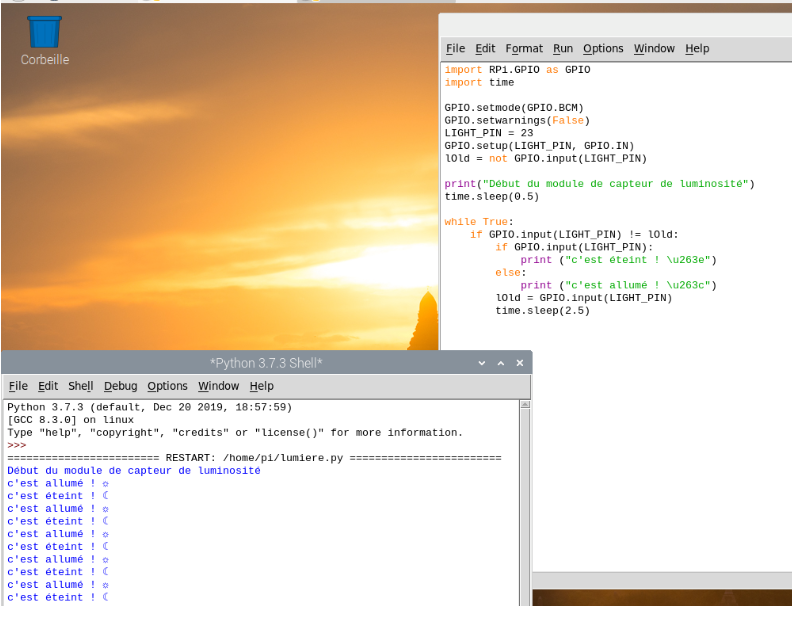
**Comparator LM393**

*Light sensor module*

As you can see in the connection diagram (temperature part), to use this sensor, you have to connect :

* The pink wire to the pin 4 (5V power)
* The blue wire to the pin 6 (Ground)
* The orange wire to the pin 16 (GPIO 23)

The python code of the light sensor is below:



*Brightness sensor python code*

A new line is created when the brightness change. For example, when the light intensity < fixed limit, the code says « it’s off » and when the light intensity > fixed limit, the code says « it’s on ».

## Image Recognition

### Initial Idea

One feature we wanted to work on was image recognition. This was first intended as a way to count and detect the different fish and show them according to their species, also allowing to analyze in detail their behavior to alert of any signs that could warn the user of potential health problems.  
However, to be able to recognize the fish this accurately, we would have needed to use TensorFlow, a very detailed and thus quite complex machine learning platform and use a custom library of data (a large amount of images of the fish with boundary boxes and labelled according to their species). This path was very interesting but could have been a complete AI project, and we did not have the time or resources to go through with this idea in its entirety.

OpenCv  
We thus opted for simpler image recognition, using a Python library called OpenCv, which is a basic yet powerful too that can be used in a variety of ways with images and videos. It enabled us to detect fish movement in short videos and have an approximate count of the number of fish visible on camera during that video.

#TODO add Screenshots

## Android App

### Figma

### Android Studio

## Data collection and linking

### Rest Client

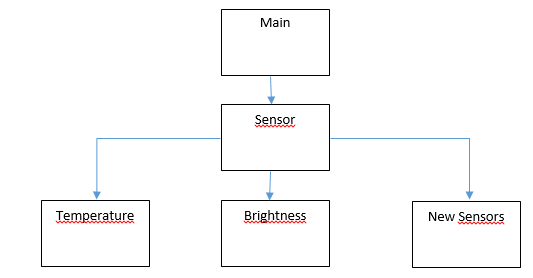
#### Android

#### Raspberry Pi

# Project perspective

## Planned additions

We would like to improve the Python codes and create classes to make the code more functional.



## Challenges

# Conclusion