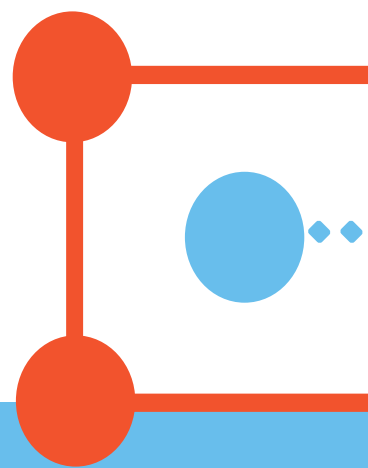


# NFC Reader project



# Acknowledgements

During this project we encountered several difficulties because of our lack of experience in some aspects of it. But we found help in our colleagues, they generously took time to allow us to move forward faster.

Thus we thank Valentin Lecommandeur for his help with the electronics, Ali Abid for his help concerning software, finally we want to thank Adrian Locke, Flannagan Noonan and Krishna Panduru for their involvement and advices for the design and 3D printing of the box.










# Summary

|                                 |   |
|---------------------------------|---|
| Acknowledgements.....           | 2 |
| 1 What did we have to do ?..... | 4 |
| 2 With what means ?.....        | 5 |
| 3 Project steps.....            | 8 |

# 1 What did we have to do ?

The Client wanted us to establish a link between the NFC scan and the saving of the card ID on a specific server. This specific server being a project of a former student which consisted in \_\_\_\_\_. To do so, we were asked to make a C program that allows, first to read from a config file the API ID and password of the server then to read the scanned card ID, prepare then send a POST request to the server and finally read and interpret the response. Concerning the interpretation, if there were no errors, the green LED should light up, otherwise the red LED would do so. Moreover, as we do prototyping, it was necessary to design a box able to contain the entire system while being as small as possible.

Below, here's the Gantt of our project :

|                                  |                        | Week 1<br>(20/04/2015 - 24/04/2015)  | Week 2<br>(25/04/2015 - 29/04/2015)   |
|----------------------------------|------------------------|--|---|
| Send POST Request                | $1 + \frac{1}{2}$ day  |     |   |
| Parse XML File                   | $\frac{1}{2}$ day      |     |   |
| Analyze libnfc                   | 1 day                  |    |   |
| Link NFC scan and POST request   | $2 + \frac{1}{2}$ days |  |   |
| Add LEDs to the circuit          | $\frac{1}{2}$ day      |  |  |
| Replace all XML by JSON          | $\frac{1}{2}$ day      |  |  |
| Design the box                   | 2 days                 |  |  |
| Boot the program at the RPi Boot | 2 days                 |  |  |
| Print the box                    | $\frac{1}{2}$ day      |  |  |

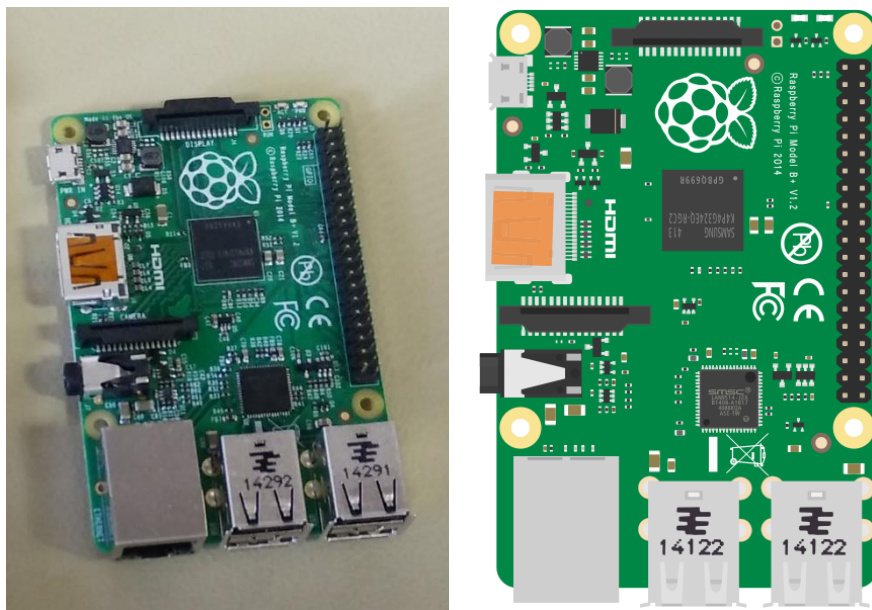
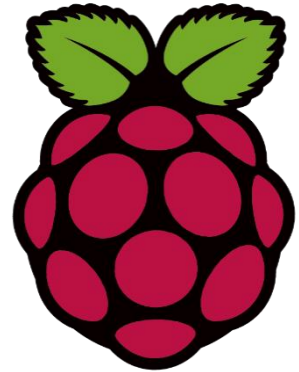
## 2 With what means ?

In order to carry out the project, some hardware elements, which are explained below, were given to us.

To begin, a micro controller, the **Raspberry Pi B+ V1.2**, was provided to us. It allows to have all PC features in one little chip. Indeed, it has some RAM, input/output (HDMI, USB, etc, ...) but it has also a SD slot which allows it to have a Unix based operating system.

Features :

- Size : 85mm x 56mm x 17mm
- Processor : Broadcom SoC 700MHz
- RAM : 512MB
- 1 Ethernet port
- 4 USB ports
- 1 HDMI port
- 1 power outlet microUSB 5V/2A
- 1 GPIO connector with 40 pins
- 1 audio jack output 3.5mm
- 1 MicroSD card slot
- 4 fixation holes (drawn in beige on the diagram)



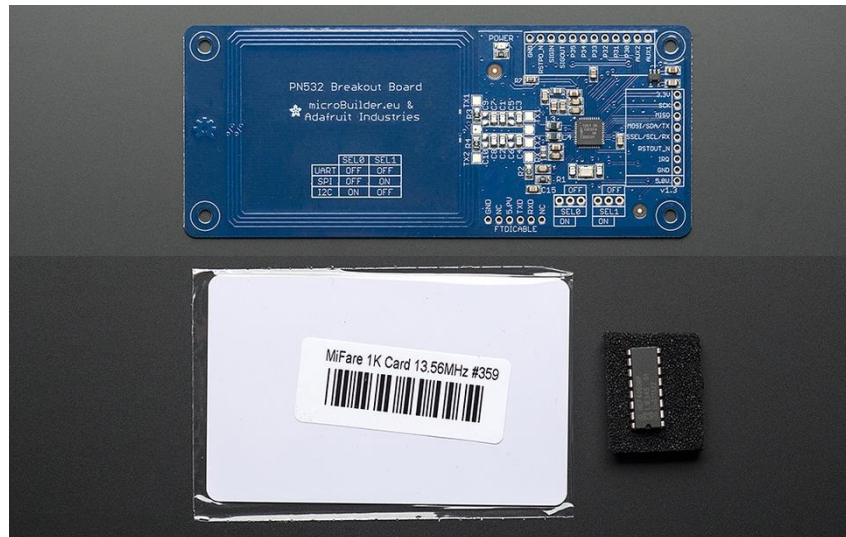
Picture and diagram of the Raspberry Pi B+ V1.2

Then, one **Adafruit PN532 Breakout Board** was provided to us. This card is the most popular card of the NFC cards and it is also this one which are in most of smartphone or devices containing NFC. Indeed, it permits to read and write data from a NFC card or even a NFC tag. Moreover this card is used by the libnfc library which allows to create programs that can interact with the card on every operating system (Windows, Linux, Mac).

Two NFC cards were given to us in order to test our program.

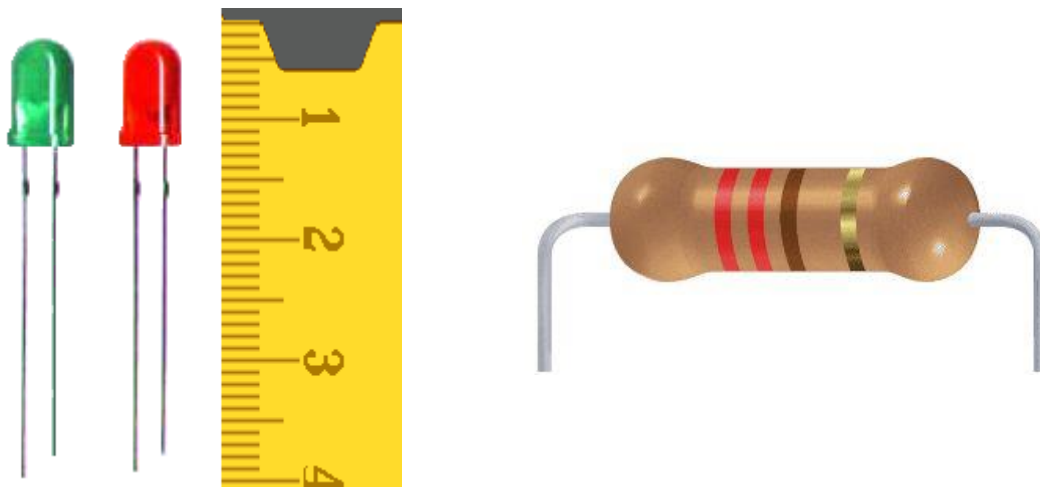
Features :

- Size : 51mm x 117.7mm x 1.1mm
- This board/chip uses I2C 7-bit address 0x48



Picture of the Adafruit PN532 card and of one NFC card

On the other hand to display a feedback to the user, two LEDs were provided to us, a green one and a red one were connected to the circuit. In order to prevent overload, it was necessary to connect a resistor before each LED. We were advised to use two electrical resistance of  $220\Omega \pm 5\%$  each.



Picture and scale of two LEDs (left side), 3D image of a  $220\Omega$  resistor (right side)

In order to link these two cards and LEDs, a breadboard and some electrical wires were given us. All of this was integrated in the following circuit :

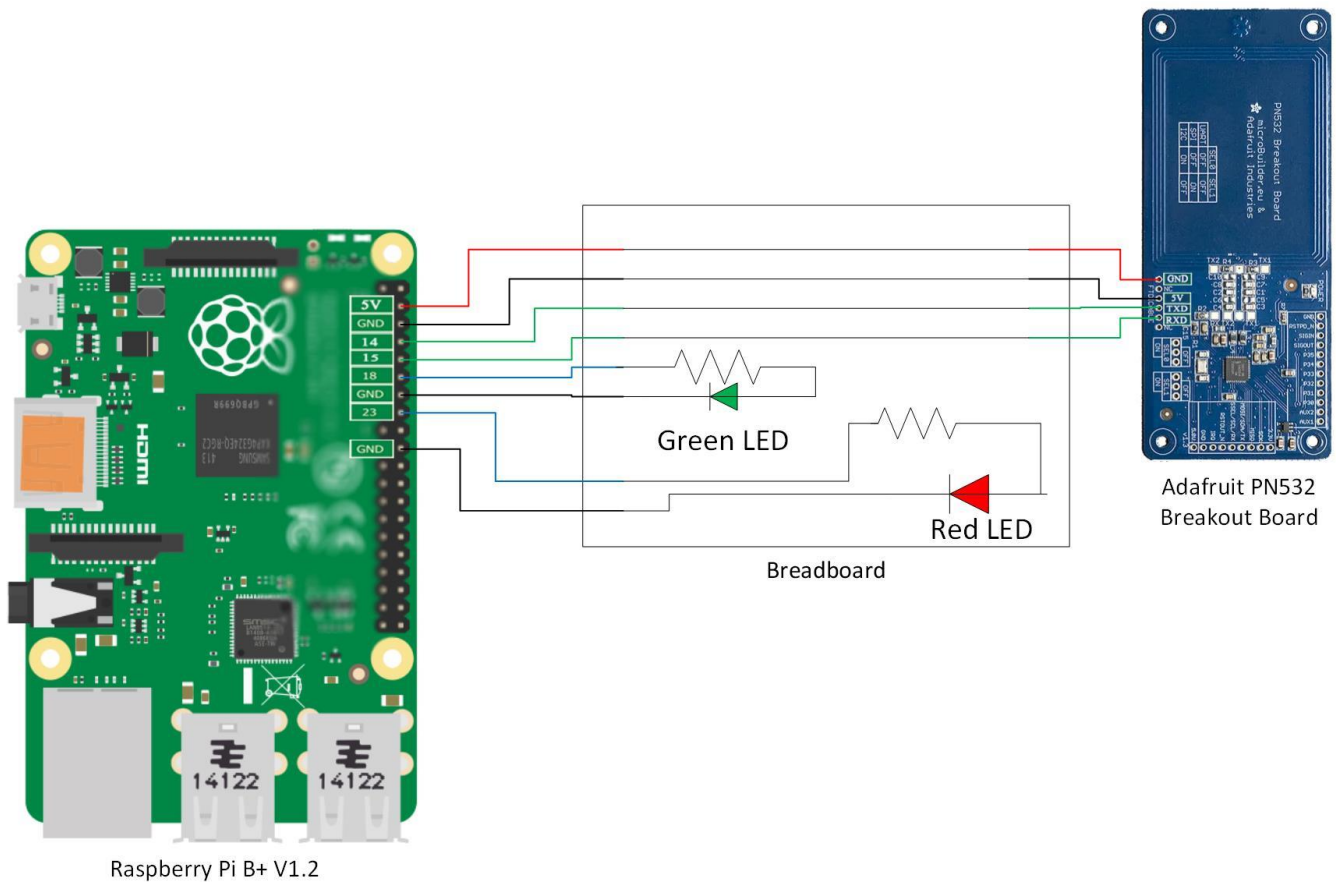


Diagram of the circuit which joins the two cards and the two LEDs with the breadboard

It was asked us to work with C language and the libnfc library which provide methods that allows to interact with the NFC reader.

For this project, we decided, with Bilal El Yassem, to use curl library, which makes network exchanges easier, and json parsing libraries, which permits to parse JSON files easily.

### 3 Project steps

The first aspect of the project we chose to work on was the **HTTP POST request** as it only depended on software. While looking for a simple way to send HTTP requests with C code we found **libcurl** and decided to use it for its simplicity. We then created a training class to practice and used the **Requestbin** website to test the requests.

The next step was to use an **XML parser** in order to read the response from the server. We chose to use **ezxml** which seemed to be the easiest way to parse an XML file with C code. We trained with this library like with libcurl then moved on to the next step.

**LibNFC**, the library developed by Adafruit to use the NFC shield was obviously on the list of the technologies we had to master. Using the documentation and source code of the library we managed to extract from the shield the relevant information for what we needed and tested it in a training class.

Then we had to begin to assemble the codes, first we linked the action of reading a card ID with the NFC shield and sending a request to the server. At this moment of the project the server was not available so we continued to work using **Requestbin**.

We also found out that the server was not sending XML replies but JSON replies, so we tried to modify the server to send XML answers but it was not possible. Our solution was then to **modify** our previous work to **parse JSON instead of XML**. But when we managed to do it we realized that the LEDs were not working because the circuit on the breadboard was not properly done, we then dismantled the breadboard installation provided and assembled it in the right way.

Once the **LEDs** were properly working we decided to **include** them in the program using the **GPIO API** to light them from the C program. The idea was to use them to provide feedback to the user, once a card was read and the request sent to the server the red LED would light on, if a positive answer was to come the red light would be set off and the green one would light on for a few seconds. We also used the LEDs to show different codes to signal the different possible failures of the system.

Our next goal was then to **launch the program on startup** and to loop it as long as the device would be on. We did it using shell scripts that we integrated to the boot directories of the operating system of the Pi.



Then we **assembled** the **whole circuit** and **soldered** it using a project board. This allowed us to take the measurements we needed to begin the design of the box.  
Here is, below, the last version of the circuit :

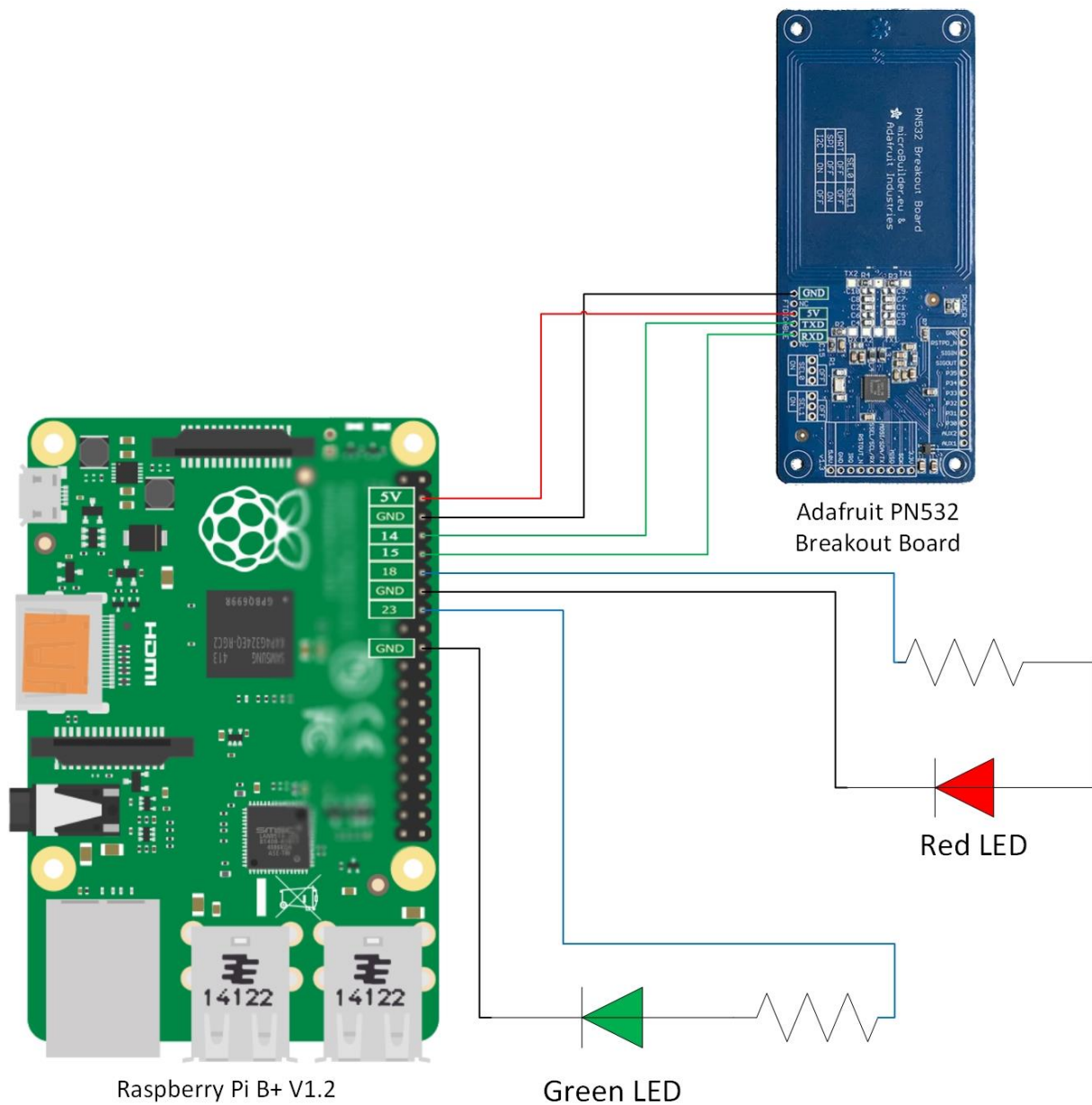
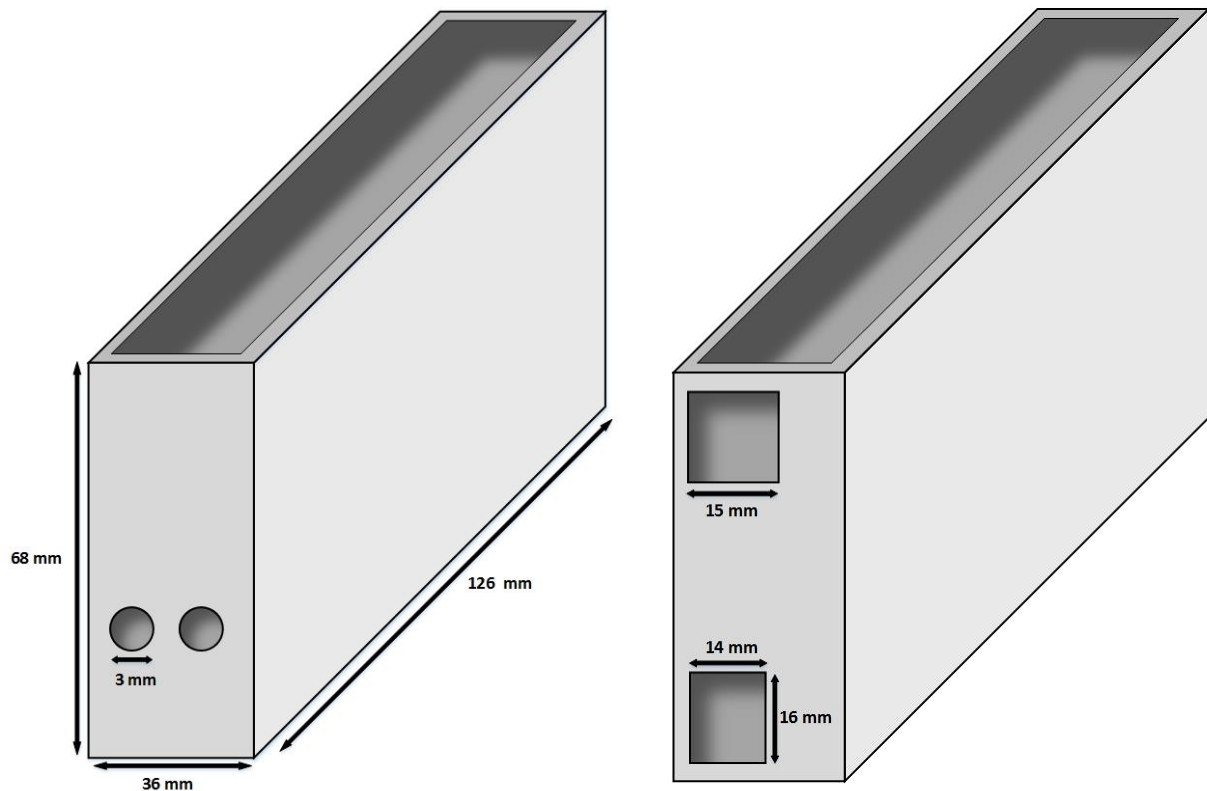
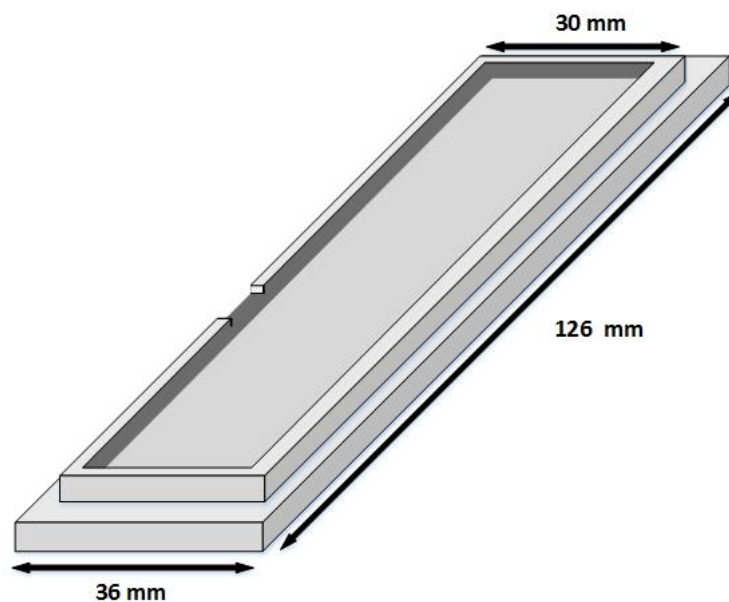


Diagram of the final circuit which joins the two cards and the two LEDs

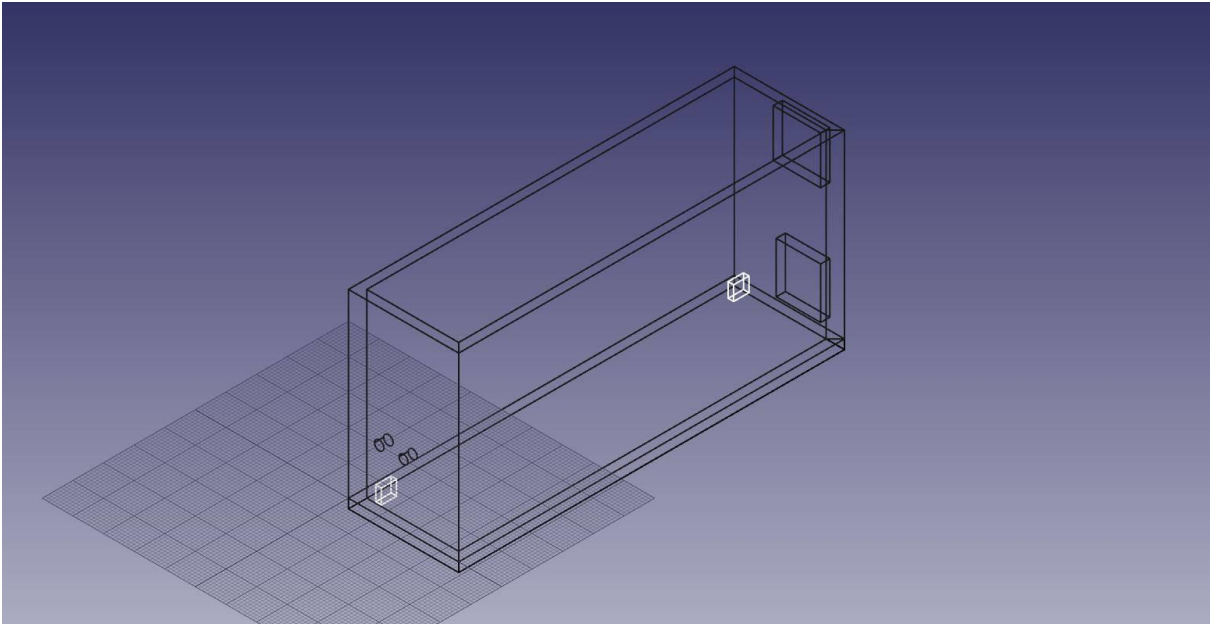
To **design** the **box** we chose **FreeCAD**, a simple 3D design software and created the box and its lid using the measurements we made during the previous step. Here is, below, the plan of the box :



3D Plans of the box on each side

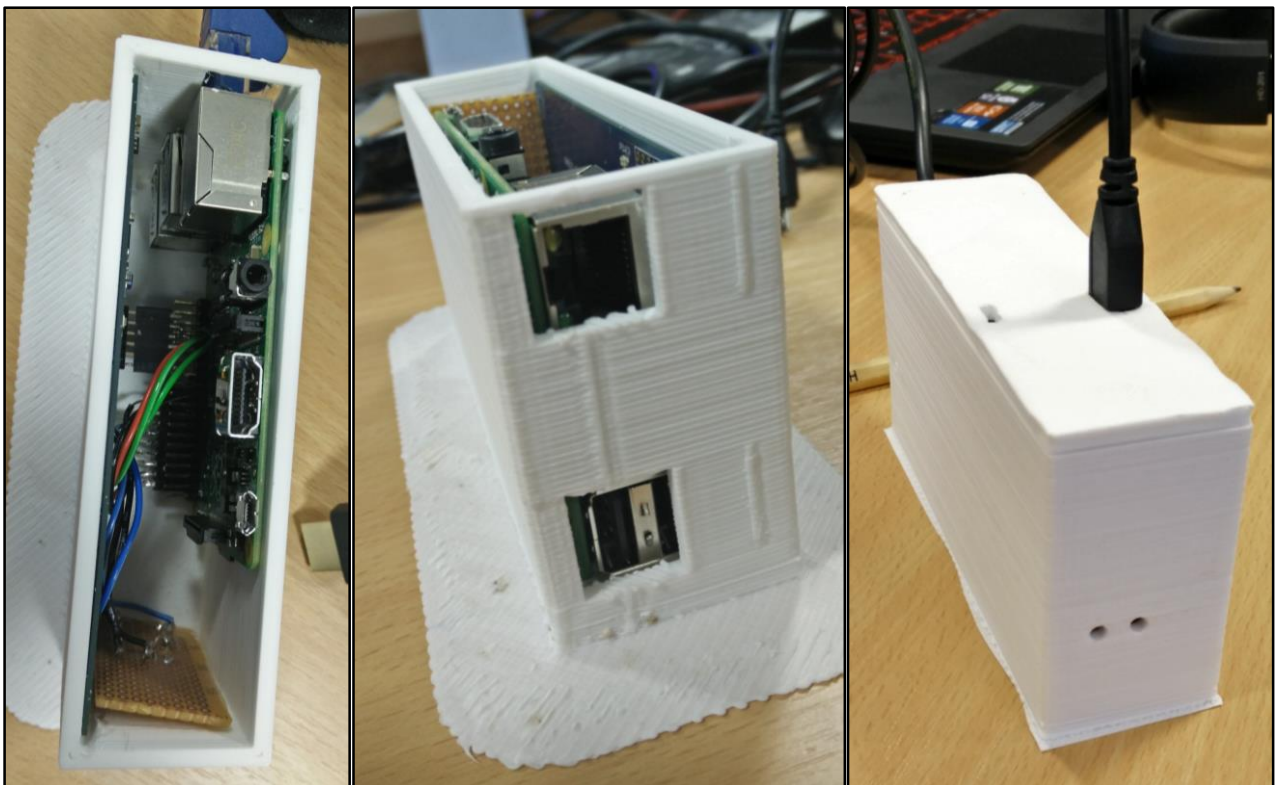


3D Plan of the box lid



Box wireframe on FreeCAD

Finally we **printed the box** and even if there were a lot of design problems with it we chose to adapt it without printing another one to avoid a waste of resources.



Picture of the printed box