Explanatory document for next student

This document will explain the important aspects needed to quickly understand and continue working on the G-Mill robot. It is a supplement to the report written and goes through the more technical parts that were not explained in the report.

1 Code structure

The code is divided into two: one part for the arduino in C++ and another part for the raspberry pi in python. The file for rapsberry pi's code is named `G-Mill_Marianne.py` and the code for the arduino is placed in the folder GMill_sw-main. The folder structure is the following:

```
. ─┬─ G-Mill_Marianne.py
   └─ GMill_sw-Main ─┬─ include
                     ├─ lib
                     ├─ test
                     ├─ platformio.ini
                     └─ src ─┬─ bwsencoders.cpp
                             ├─ bwsencoders.h
                             ├─ communication.cpp
                             ├─ communication.h
                             ├─ endswitches.cpp
                             ├─ endswitches.h
                             ├─ main.cpp
                             ├─ manualencoder.cpp
                             ├─ manualencoder.h
                             ├─ motorencoders.cpp
                             ├─ motorencoders.h
                             ├─ parameters.cpp
                             ├─ parameters.h
                             ├─ treadmills.cpp
                             ├─ treadmills.h
                             ├─ weightsupports.cpp
                             └─ weightsupports.h
```

The code for the arduino is placed in the src folder. Everything is called in the `main.cpp` file and the rest is coded in the other files like a normal c++ architecture.
To run the code, the file `G-Mill_Marianne.py` needs to be executed on the raspberry pi. Once this is done, since the arduino and raspberry pi are connected via serial communication (with a cable), they will communicate and the arduino's code will work depending on the information received by the raspberry pi.

2 Flashing code onto the arduino

To flash the code onto the arduino, this will be done through the raspberry pi. Indeed, the raspberry pi is connected to the screen where the raspberry IDE is located. The raspberry IDE can be used as a computer, from there, all of the files can be accessed. What I usually did to get the new code onto it, was connecting the raspberry pi to internet via ethernet cable and using git to upload the new code onto it.

Once this is done, the arduino's code can be flashed onto it. To do so, the software platformio is very useful as it allows to flash the code directly from vsCode and without needing to install the arduino IDE. The procedure is the following:

1. On the raspberry's IDE, open the GMill_sw-Main folder which also contains the platformio.ini file in vsCode.
2. This directly initiates the platformio software.
3. Press on the ✓symbol to build the code and check for errors.
4. Press on the → symbol to upload the code onto the arduino.
5. You should get a success message meaning that it worked properly.

Once this is done, you can directly run the raspberry's code (`G-Mill_Marianne.py`) and the changes on the arduino will have been applied.

3 Git

As mentioned before, I used git to upload code onto the raspberry-pi's IDE. To get the last version of my git here is the procedure:

1. Open a terminal window
2. Go to /Desktop/G-mill
3. Perform a git pull
4. All of the new files should arrive
5. If there is an error with the git, simply remove it and add it again or use the files that I provided with my report.

4 Automatic GUI interface popup at startup

In order to automatically get the GUI interface when starting the robot, such that the user doesn't need to go through the files and manually run the raspberry's code, I added a file to run during the reboot program.
Indeed, I created the file `g_mill_start.desktop` placed in

/home/.G-Mill/.config/autostart/g_mill_start.desktop

which calls the `G-Mill_Marianne.py` file when the robot starts. If the file raspberry's file were to change location or name, the `g_mill_start.desktop` will also need to be modified and take into account the change.

5 Switches numbering

The switches on the robot were numbered in the code. I wrote down their positions and respective numbers such that they could be easily found.
There are two types of switches, internal switches and external switches (see figures 1, 2). The internal switches are placed inside the back of the treadmill whereas the external ones are placed outside, near the rodent supports.

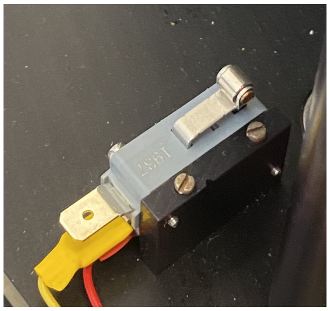

Figure 1: Internal switch

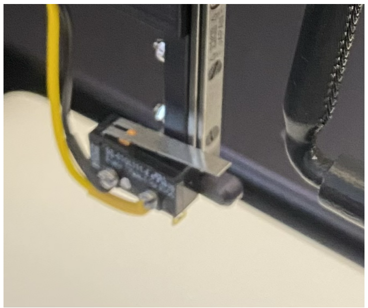

Figure 2: External switch

The numbers correspond to the following:

	Mice's side	**Rats' side**
Internal	1	2
External	3	4

Table 1: Numbering of the switches and their meaning

They are then separated into up or down to differentiate the top or bottom switch.

6 Species numbering

A small side note regarding the species numbering. In the raspberry pi's code, they are numbered as 0 and 1 to represent the mice and rats respectively. Whereas in the arduino, they are numbered as 1 and 2 for the mice and rats respectively. This could be changed to improve the code's lisibility.