PROJET 2SU

Mini Farmbot

AARAB Fadoua
HAMITOU Asmaa

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

Remerciement

Toute notre gratitude et profonde reconnaissance s’adressent à Monsieur Jeremy Briffaut de nous avoir garanti les meilleures conditions et facilités pour mener à bien ce projet et qui a su se rendre disponible pour nous aider quand le besoin s’en faisait ressentir.

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

Sommaire

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES CENTRE VAL DE LOIRE

1. *Introduction*

Le robot potager Farmbot est destiné à réintroduire la culture potagère dans nos modes de vie. Il s'agit d'un robot Open-Source conçu pour rendre un potager autonome. Les fonctionnalités techniques et le logiciel du robot permettent à l'utilisateur de programmer et de gérer son potager en optimisant l'espace et en réduisant l'apport en eau et en intrants.

Farmbot apporte donc une autre dimension à l'agriculture urbaine en permettant de concevoir des potagers ou des fermes connectés et autonomes.

2. *Présentation du projet*

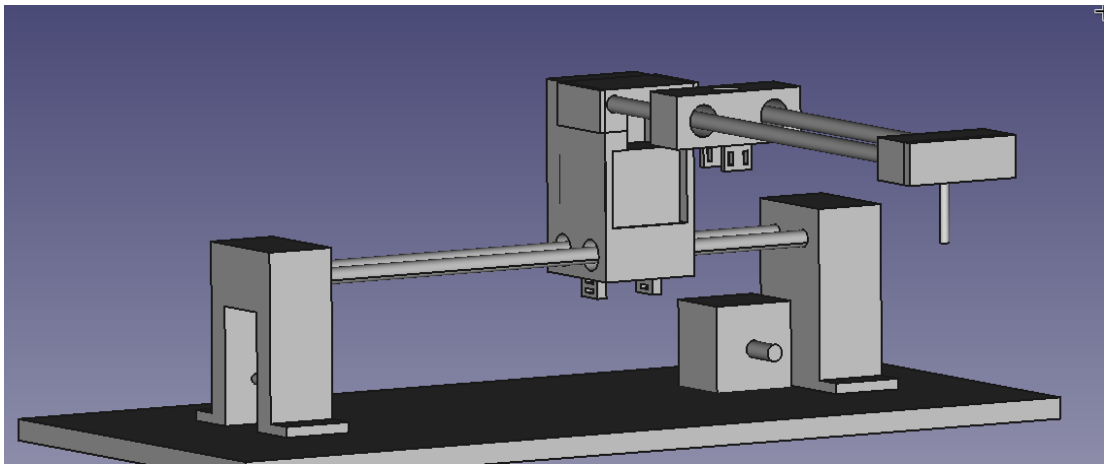
L'objectif du projet était de concevoir un système inspiré du Farmbot et d'en créer une version plus petite.
Contrairement au vrai Farmbot, le système qui est le sujet de ce projet permet seulement l'arrosage des plantes et la détection du niveau d'humidité.
Cela est possible grâce aux mouvements des moteurs selon les axes X et Y qui vont permettre de déplacer les pièces du système et donc le tuyau d'arrosage jusqu'aux plantes afin de les arroser et de déclencher l'arrosage en contrôlant une mini pompe à eau.

En tenant compte des circonstances dans lesquelles a été réalisé ce projet, le développement du système n'est pas abouti. Le système à ce niveau permet seulement le contrôle du moteur qui provoque le mouvement selon l'axe Y depuis un site web, la détection du niveau d'humidité et son affichage sur le site.

Le projet permet globalement à un utilisateur de contrôler le mini farmbot depuis une page web hébergée sur la raspberry pi qui va envoyer les commandes à l'arduino et l'arduino va ensuite les envoyer aux moteurs qui vont provoquer le mouvement du système afin d'arroser les plantes.

Pour réaliser le projet, un plan du système a été conçu et des pièces 3D ont été créées avec FreeCAD. L'assemblage est complété avec des roulements, des poulies et d'autre matériel qui sera détaillé plus bas dans le rapport.

Voici à quoi ressemble l'assemblage final du système :



Afin de contrôler les moteurs, deux solutions ont été testées.
Le contrôle des moteurs a bien été possible notamment avec une CNC, le firmware GRBL et le Universal Code Sender mais cette solution s'avère au final inappropriée puisque une interface plus complète devrait être développée. Le UGS ne permet que le contrôle des moteurs.

Des tentatives d'utiliser le code source et les fonctions de grbl ont été testées mais une solution indépendante d'un firmware a été jugée plus rapide à réaliser.

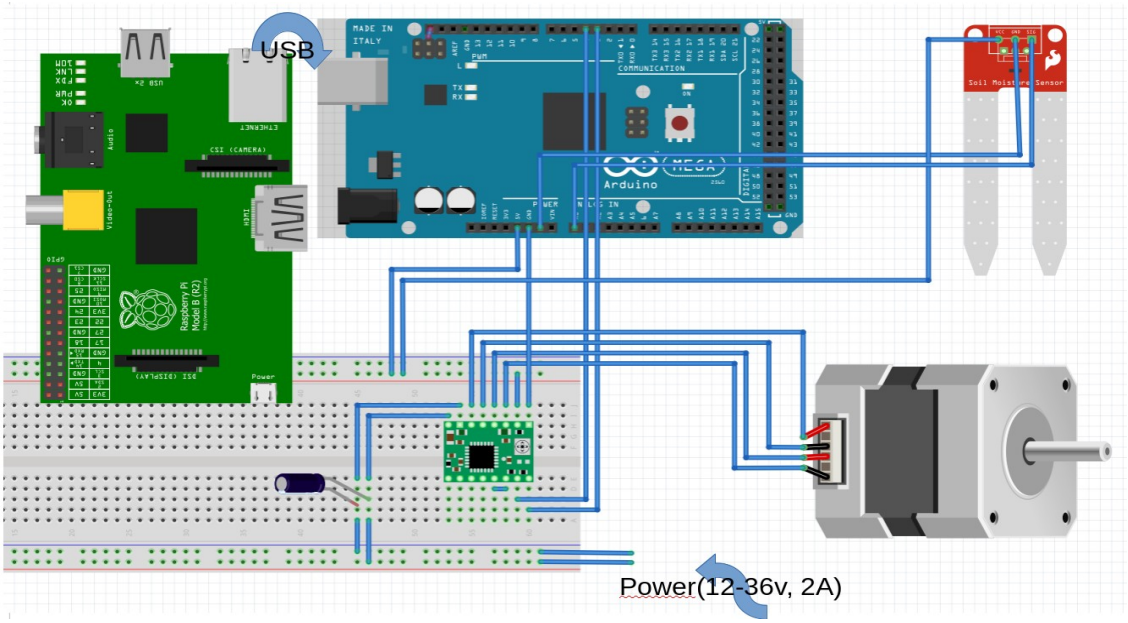
Par conséquent, une solution alternative a été mise en œuvre pour le contrôle des moteurs en utilisant le driver A4980 avec l'arduino.

L'arduino est aussi branchée à un capteur d'humidité. Ce capteur permet de calculer le taux d'humidité pour l'afficher mais éventuellement déclencher une suite de tâches selon sa valeur.

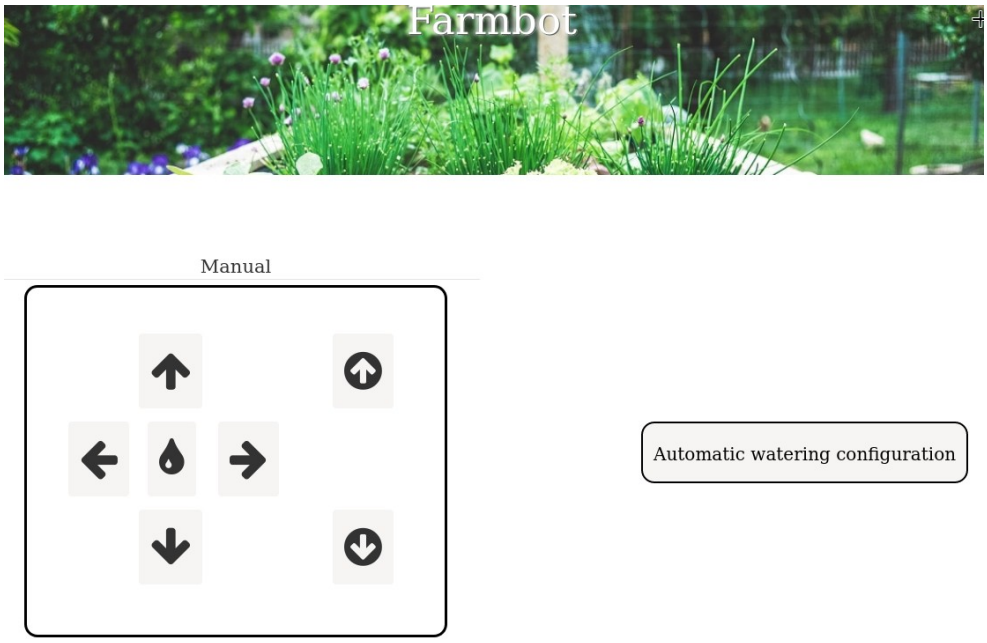
Le circuit avec le driver et le moteur est alimenté avec un adaptateur d'alimentation (19v, 2A).

L'interface avec l'utilisateur est possible grâce à un site web hébergé sur une Raspberry pi qui communique en série avec l'arduino.

Voici le schéma du circuit final :



Le site web a été développé avec flask et bootstrap pour permettre à l'utilisateur de contrôler les mouvements du système, sa configuration et l'affichage de données récupérées de ce dernier.
Voici une image d'une page du site :



Le projet développé est donc une application web qui va permettre, par le biais d'une Raspberry Pi, de communiquer avec un système d'arrosage embarqué dans une arduino.

2.1. Description de l'arborescence des dossiers et fichiers du projet

Les fichiers utilisés dans ce projets sont disponible sur github : https://github.com/AsmaaAmina/MiniFarmbotV1.1

Le dossier ***3D*** contient les plans 3D des pièces et une prévision d'ensemble du système (seulement les pièces à imprimé et deux moteurs (il faudra rajouter les roulements, courroies, vis, etc) il y a un dossier avec les versions stl et un autre en fcstd.
Le dossier ***FarmBotServer*** contient un dossier avec les codes arduinos à téléverser dans l'arduino et un autre dossier qui contient les fichiers utiles pour le serveur web.
Le dossier ***images and circuits*** contient des images des pièces et le schéma du circuit.
Le fichier ***ReadMe.txt*** contient des consignes pour les installations et quelques remarques sur les étapes de la réalisation du projet.

3. *CAO et assemblage*

3.1. Software

FreeCAD et A2plus Workbench.

3.2. Matériel pour l'assemblage

L’ensemble de matériel dont on aura besoin pour ce projet:

- 4 roulements de diamètre 15 mm : 2 courts et 2 longs
- 4 tiges de diamètre 8 mm et de longueur 30 cm
- 4 poulies
- Courroies
- Vis
- Moteurs

4. *Conception des pièces 3D*

Afin de réaliser notre assemblage nous avons conçu six pièces:

1. La première pièce se déplacera selon l’axe x. Elle sert dans un premier temps à fixer les tiges permettant le mouvement sur l’axe y, mais aussi comme support pour le moteur
2. La deuxième pièce qui se déplacera sur l’axe y a pour but de maintenir le tuyau d’eau
3. La 3eme pièce sert comme support pour le moteur.
4. La 4eme pièce sert tout d’abord à fixer les tiges de l’axe y, mais également à maintenir une poulie entraînant le déplacement de la pièce 2 grâce à une courroie et au moteur supérieur.
5. Les pièces 5 et 6 servent à fixer les tiges de l’axe x, en plus la pièce 6 permet de maintenir la poulie entraînant le mouvement de la pièce 1.

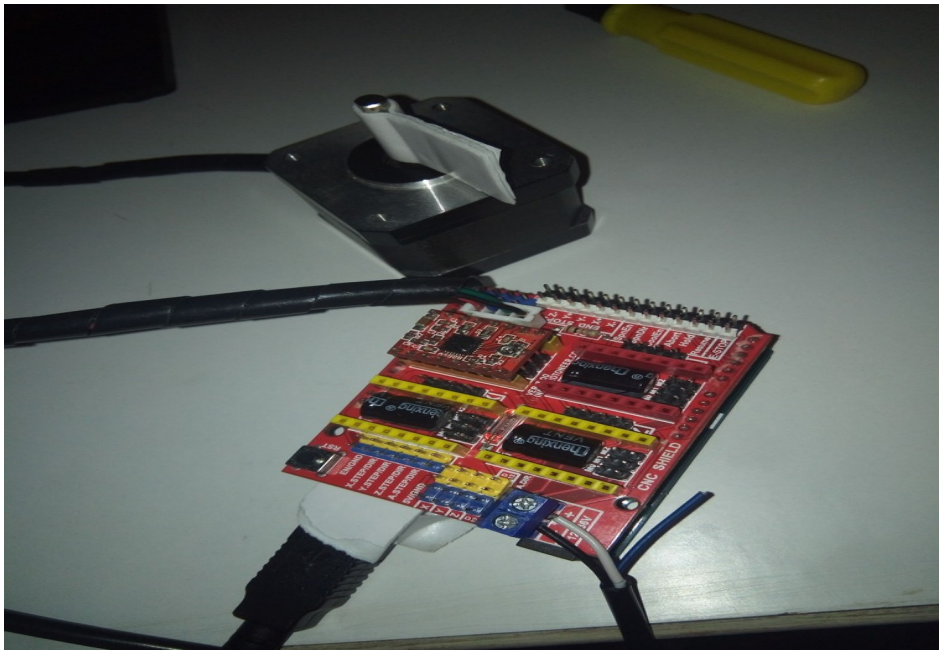
5. *Tests avec le GRBL*

5.1. Matériel électroniques

- Raspberry pi 3B
- Arduino Uno
- Drivers A4980
- CNC shield et un adaptateur d'alimentation (entre 12 et 36v , ~2A).

5.2. Circuit

- On branche les drivers avec la CNC et la CNC à l'arduino.
- On branche l'alimentation avec la CNC.
- On branche le(s) moteur(s) avec la CNC :



5.3. Arduino

On ajoute tout d'abord le grbl au librairies de l'IDE Arduino. On téléverse le grbl (GrblUploading.ino) dans l'arduino et on utilise UniversalGcodeSender pour envoyer les commandes à la CNC.

6. *Solution avec la page web*

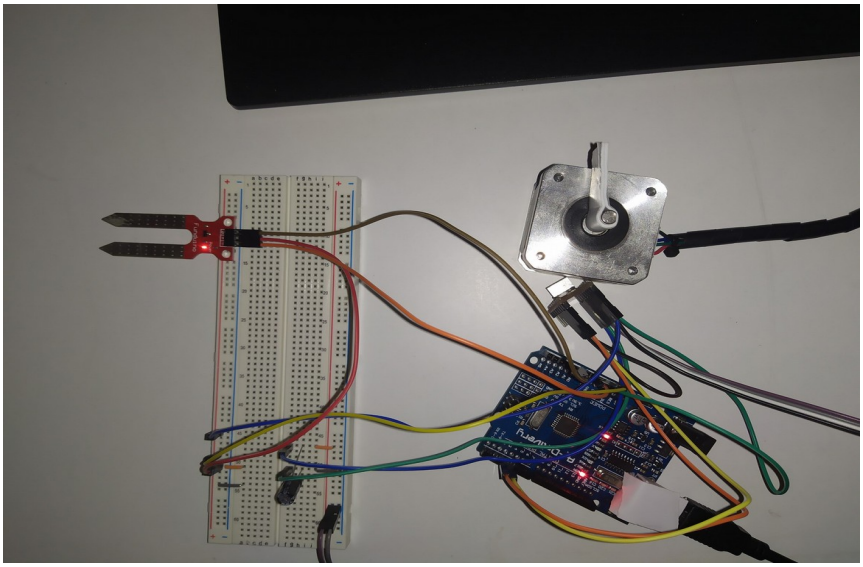
6.1. Matériel électroniques

- Raspberry pi 3B, Arduino Uno, drivers A4980.
- Condensateur (on ajuste la bonne valeur d'intensité avec le potentiomètre du driver pour protéger les drivers et l'arduino).
- Fils
- Adaptateur d'alimentation.
- Funduino humidity sensor.

6.2. Software et frameworks

- Flask
- Bootstrap
- Arduino IDE

6.3. Circuit



6.4. Partie Arduino

Le code téléversé dans l'arduino pour cette version est dans le fichier farmbotArduino.ino:

```
#define dirPin 3
#define stepPin 4
#define stepsPerRevolution 200
#define SensorPin A0
float sensorValue = 0;
void setup() {
  // Declare pins as output:
  pinMode(stepPin, OUTPUT);
  pinMode(dirPin, OUTPUT);
  Serial.begin(9600);  // start serial communication at 9600 baud
}

void loop() {
  // Set the spinning direction clockwise:
  // Read and execute commands from serial port
   if (Serial.available()) {  // check for incoming serial data
      String command = Serial.readString();  // read command from serial port
       if (command == "yu") {
         digitalWrite(dirPin, HIGH);// Spin the stepper motor 1 revolution slowly
         for (int i = 0; i < stepsPerRevolution; i++) {
        // These four lines result in 1 step:
         digitalWrite(stepPin, HIGH);
         delayMicroseconds(1000);
         digitalWrite(stepPin, LOW);
         delayMicroseconds(1000);
         }
       }
       else if (command == "yd"){
         digitalWrite(dirPin, LOW);

  // Spin the stepper motor 1 revolution quickly:
        for (int i = 0; i < stepsPerRevolution; i++) {
       // These four lines result in 1 step:
         digitalWrite(stepPin, HIGH);
         delayMicroseconds(1000);
         digitalWrite(stepPin, LOW);
         delayMicroseconds(1000);
     }

       }
       else if (command == "h") {
         // read and send A0 analog value
            Serial.println(analogRead(SensorPin));
       }
```

On ouvre une connexion USB avec la raspberry et on attend les commandes. Selon la commande reçue, on fait tourner les moteurs dans la bonne direction ou on détecte et on envoie la valeur de l'humidité à la raspberry.

6.5. Partie Raspberry pi

Du côté de la raspberry pi, nous avons en back-end un serveur Flask.
On vérifie la bonne connexion de l'USB pour soit afficher une page d'erreur ou la page pour les fonctionnalités du farmbot.
Selon les valeurs des champs d'un formulaire (boutons en front-end), on envoie les bonnes commandes à l'arduino.

- **Back-end**

```
from flask import Flask, render_template, request, jsonify
USB_PORT1 = "/dev/ttyUSB0"
import serial
import time
import random
app = Flask(__name__)

@app.route("/", methods = ['POST', 'GET'])
def template_test():
    try:
        usb = serial.Serial(USB_PORT1, 9600, timeout=2)
    except:
        print("ERROR - Could not open USB serial port.  Please check your port name and permissions.")
         #print("Exiting program.")
         #exit()
        return render_template('error.html')
```

```
    elif  request.form.get('upbtn') == 'upbtn':
        # pass # do something else
        usb.write(b'yu')
        print("Yup")
    elif  request.form.get('downbtn') == 'downbtn':
        # send
        usb.write(b'yd')
        print("Ydown")
    elif  request.form.get('Zup') == 'Zup':
        # pass # do something else
        print("Zup")
    elif  request.form.get('Zdown') == 'Zdown':
        # pass # do something else
        print("Zdown")
    elif request.form.get('Humidity') == 'Humidity':
            usb.write(b'h')  # send command to Arduino
            line = usb.readline()  # read input from Arduino
            line = line.decode()  # convert type from bytes to string
            line = line.strip()  # strip extra whitespace charactersv
            f = open("humidity.txt", "w") #write humidity in file
            f.write(line)
            f.close()
```

La valeur de l'humidité reçue de l'arduino est stockée dans le fichier humidity.txt pour ensuite être lu et affichée en front.

- **Front-end**

6.6. Description de la page web

On a développé un site web avec flask et bootstrap, afin de permettre à l'utilisateur de contrôler les mouvements du système, sa configuration ainsi que l'affichage de données récupérées de ce dernier.

Sur le partie gauche on retrouve un manuel qui permet le mouvement selon l'axe x et y et z (actuellement on ne peut faire que les mouvements selon x et y, celui selon l'axe z n'est pas encore implémenté vu qu'on disposé que de deux moteurs)
A droite on a un bouton qui devra configurer automatiquement l'arrosage, mais qui n'est pas encore implémenté sur cette version.
Et enfin on a l'affichage de l'humidité, actuellement on ne peut afficher l'humidité qu'au moment où on rafraîchit la page, mais l'idéal sera de l'afficher en temps réel.

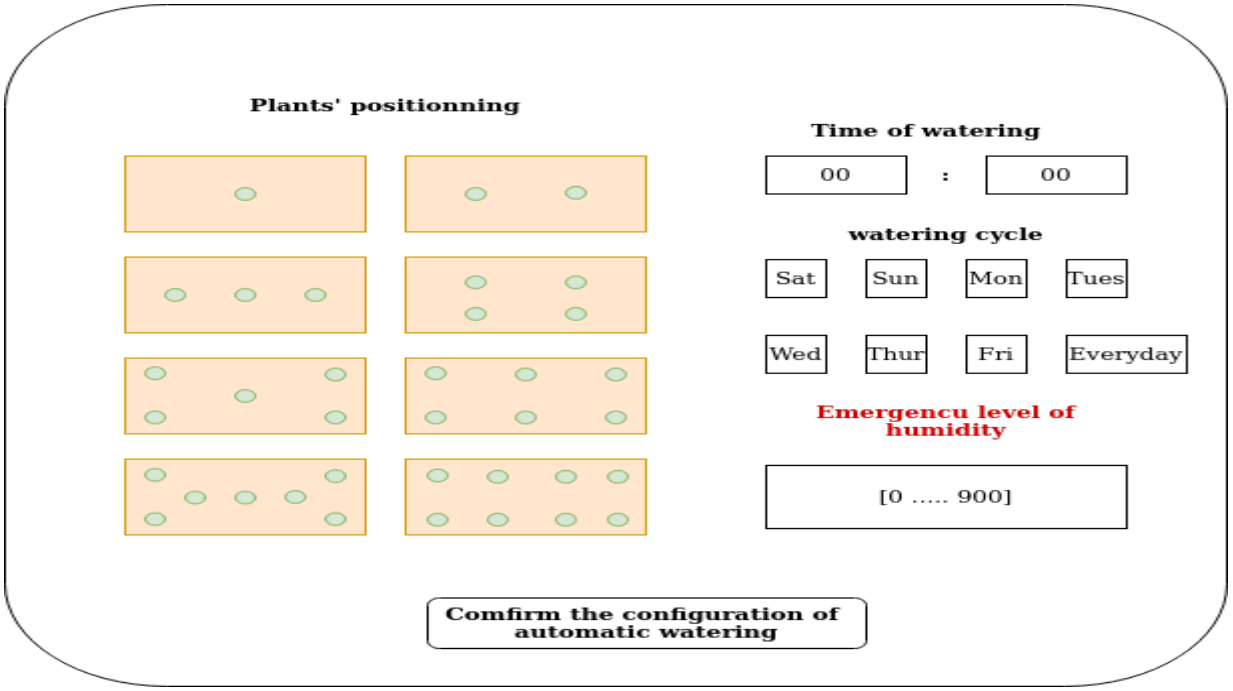
Le projet développé est donc une application web qui va permettre, par le biais d'une Raspbery Pi, de communiquer avec un système d'arrosage embarqué dans une arduino.

6.7. Tests

Les vidéos des tests et le reste des fichiers sont disponibles sur le github : https://github.com/AsmaaAmina/MiniFarmbotV1.1

7. *Améliorations*

Le bouton **automatic watering configuration** est censé nous rediriger vers un formulaire pour configurer un cycle d'arrosage automatique.
Cette page n'est pas implémentée mais elle ressemblerait à la maquette suivante:



L'utilisateur pourrait choisir le positionnement de ces plantes, le temps de l'arrosage et le cycle d'arrosage. Un cycle d'arrosage sera déclenché à l'heure précisée aux jours choisis.
On peut aussi définir une valeur d'humidité qui lorsque détectée déclenche un arrosage d'urgence.

Autres améliorations

- Introduire la partie de l'arrosage avec le contrôle de la mini pompe à eau.
- Introduire les limiteurs de mouvement sur les axes.
- Affichage de l'humidité sur le site en temps réel.
- Introduire le mouvement z avec une autre pièce et un troisième moteur.
- Implémenter le formulaire de configuration de l'arrosage automatique et la partie back + le code arduino.

8. *Sources*

- gnea/grbl: An open source, embedded, high performance g-code-parser and CNC milling controller written in optimized C that will run on a straight Arduino
- winder/Universal-G-Code-Sender: A cross-platform G-Code sender for GRBL, Smoothieware, TinyG and G2core.
- Installation — Flask Documentation (1.1.x)