Automata öntöző rendszer fejlesztői dokumentációja

Tervezési fázis

Alapvetően nem volt ötletünk, hogy milyen szerkezetet kellene megvalósítani, de aztán az egyik ‘Kertészeti alapismeretek’ óráról kilépve arra jöttünk rá, hogy milyen menő lenne egy automata locsoló rendszer. És ez milyen jó ötlet volt. Kinek van kedve hajolgatni és vizet hordani minden egyes nap, mert van egy élőlény(parazita) aki meghal a segítségünk nélkül. A modulok beszerzése bonyolult volt elsőre, mivel az Arduino ami nekünk kellett (WIFI modulos) az hiányban volt, de a további modulok, amik szükségesek voltak az öntözéshez (pumpa, cső, relé stb.), azok már könnyebben beszerezhetőek voltak.

Futtatási környezet

Arduino IDE 2.0.1 futtatására alkalmas operációs rendszer:

- Windows:
    - Windows 10, vagy újabb
- Linux:
    - AppImage 64 bits (X86-64)
- MacOS:
    - 10.14: “Mojave” vagy újabb, 64 bit

Periféria követelmények

- Billentyűzet
- Egér
- Monitor

A fejlesztői környezet

Én az Arduino Nano RP2040 Connect felprogramozásához az Arduino IDE 2.0.1 nevű programozási környezetet használtuk, mely C++ programozási nyelvet használ, és én Windows 11-et használtam, mivel az volt telepítve a számítógépemre. Továbbá, az felhasznált Arduino modul hátrányaiból kimenően, nem voltunk képesek CSV fájlokat tárolni az apró meghajtóján, szóval ez miatt az Arduino Cloud – IOT Cloud szolgáltatását vettük igénybe, ami lehetővé tette azt nekünk, hogy adatokat tároljunk az nedvességérzékelő

állapotáról, továbbá telefonon is meg lehet tekinteni egy alkalmazáson keresztül, amit az Arduino fejlesztett.

Arduino Cloud – IOT Cloud

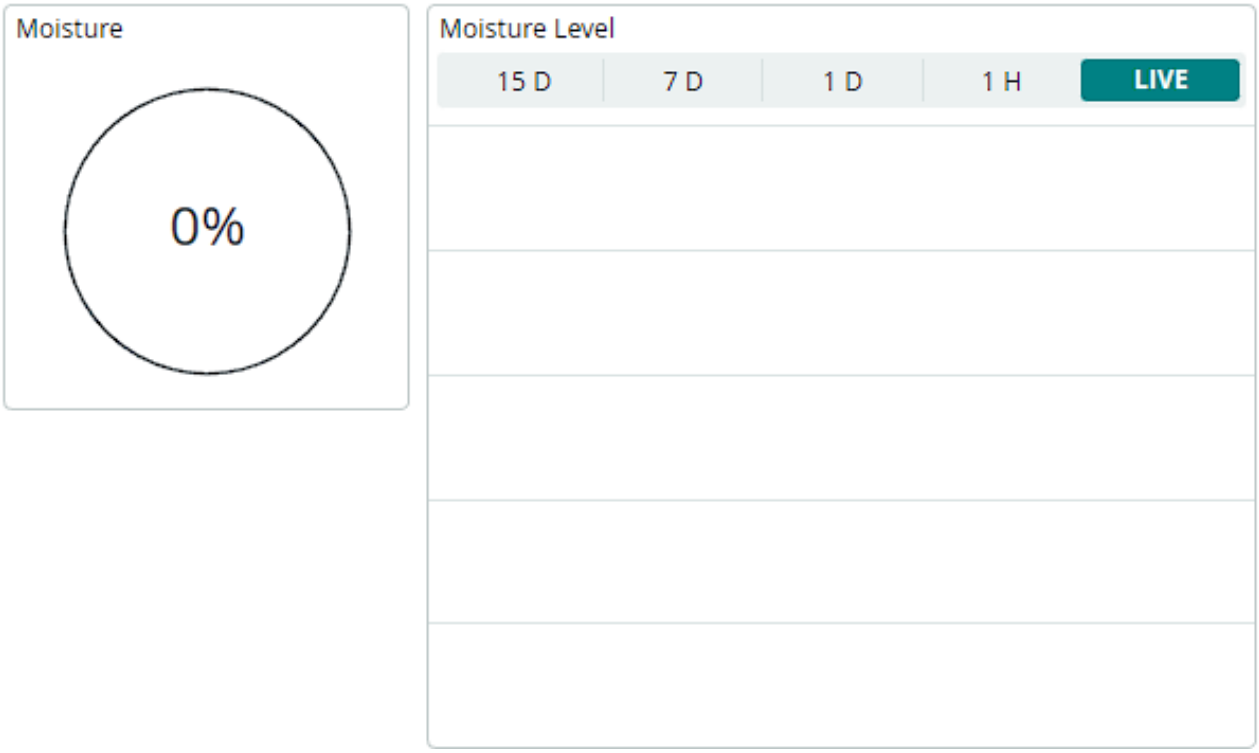
Az Arduino IoT Cloud egy online platform, amely megkönnyíti az IoT-projektek létrehozását, telepítését és figyelését.

Az IOT Cloudban létrehozott változók a következők voltak:

	Name ↓	Last Value	Last Update
☐	**current_Moisture** `int current_Moisture;`	0	27 Dec 2022 13:11:54 ⋮
☐	**pump_Status** `bool pump_Status;`	true	27 Dec 2022 13:11:54 ⋮
☐	**trigger_Level** `int trigger_Level;`	30	27 Dec 2022 13:11:14 ⋮

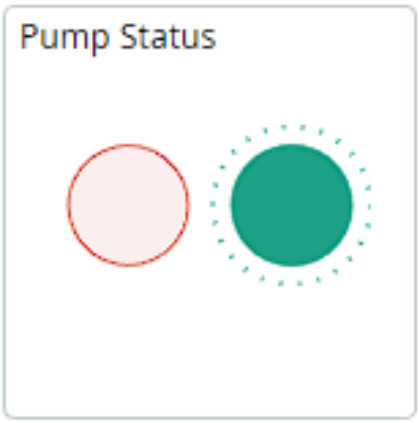
current_Moisture:

A nedvességérzékelő jelenlegi állapotát tárolja (százalékban értelmezve), és ezt visszadja a felhasználónak, és ebből majd az IOT Cloud elmenti egy CSV fájlba a szerverükön.



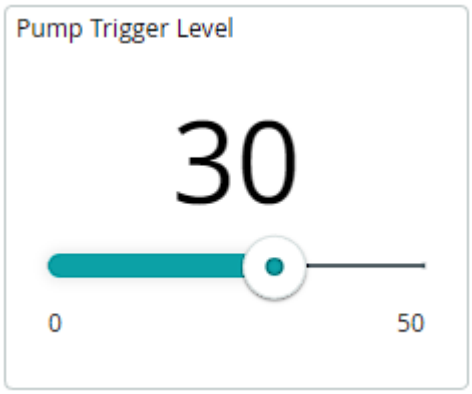
pump_Status

A pumpa állapotát tárolja bool formában. Ennek az értékét egy grafikán kirajzolja a felület.



trigger_Level

Ez egy olyan integer, ami azt a kritikus értéket százalékban tárolja, mikor a pumpa bekapcsol. Alapból 30 az alapérték.



Felhasznált modulok

- Arduino Nano RP2040 Connect
- RELC-1CH-5V-UNI
- AD20P-1230C
- WTRH-TR-811
- RC-40-20/MF
- SA-27
- OBO A11

A fontosabb modulok leírása

Arduino Nano RP2040 Connect

Az Arduino Nano RP2040 Connect az az Arduino Nano családba tartozó WIFI és Bluetooth modullal felszerelt mikrovezérlő. Képes az Internetre csatlakozni (ami szükséges) és továbbá, ha szükség lenne rá most a növény locsoló rendszerhez, akkor Bluetooth segítségével a közelében lévő Bluetooth eszközöket is tudnánk vezérelni. Egy a mikroszámítógép továbbá támogatja az Arduino Cloud fejlesztői felületet, és az integrálását.

A mikrovezérlőnek van 3.3 és 5V outputja (az utóbbi forrasztás után érhető el), amik képesek feszültséggel táplálni a ‘RELC-1CH-5V-UNI' relét, és a ‘SA-27’ vízérzékelőt.

RELC-1CH-5V-UNI

Ez egy 1 csatornás 5 Voltos opcionális vezérlésű relé optocsatolóval. Ez a legjobb relé, amit találtunk és úgy gondoljuk, hogy teszi a dolgát. Ahogy az elején le lett írva, 5V feszültséggel

lehet kapcsolni és alacsony, illetve magas vezérlő jel opcióban is lehet vezérelni, de most a mi projektünknél magas feszültség beállításban lesz.

A relé képes átkapcsolni 30V feszültséget egyenáramon, és 250V feszültséget váltakozó áramon, ami egy picit erős a mi kis 12V feszültségű vízpumpánkhoz, de a túlzás sose árt.

Mérete apró: 26x50x20 mm, és ez miatt könnyen belefér a 'OBO A11' kis tároló dobozban a többi központi modullal együtt.

Hozzáadott bónusz az, hogy led-del van felszerelve, és így lehet látni, ha be van kapcsolva, vagy ha ki van kapcsolva.

SA-27

Ez lesz a vízérzékelő modul a szerkezetben, és egyben a (szerintünk) második legfontosabb komponens a mikrovezérlő után. Ez egy olcsó rezisztív talajnedvesség érzékelő modul, ami azt jelenti, hogy nem olyan strapabíró, mint a kapacitív testvére, de könnyű előállítani, kisebb, és bónuszként esetlegesen fém mérgezi a talajt hosszas használat után, mivel a feszültség hatására korrózál az érzékelő fejben lévő fém.

AD20P-1230C

Ez a vízpumpa, amely a vizet fogja majd pumpálni egy tartájból. 12V tápfeszültség szükséges neki, 350mA mellett. Meglehetősen erős, szóval figyelembe kellett venni azt az algoritmus tervezése során, hogy el ne árassza a növényt a locsolás során.

Az algoritmusok és a kódok

Könyvtárak:

ArduinoMqttClient

ArduinoECCX08

Arduino_DebugUtils

Arduino_ConnectionHandler

ArduinoIoTCloud

WiFiNINA=

src.ino teljes kódja

```
#include "thingProperties.h"

int Relaypin = 2;
int sensorPin = A0;

int sensorValue;
int soilMoistureValue;
int soilMoisturePercent;

const int DryValue = 1000;  // a setupCode-ban látható módon szereztük meg
const int WetValue = 400;   // ugyan így
int pump_trigger = 30;
String pump_status_text = "OFF";



int periodShort = 1500;  // 1,5 másodperc
int periodLong = 10000;  // 10 másodperc
int periodUpdate = 5000;  // 5 másodperc
unsigned long time_now = 0;
unsigned long time_nowUpdate = 0;

void setup() {
  Serial.begin(9600);
  delay(1500);

  initProperties();



  ArduinoCloud.begin(ArduinoIoTPreferredConnection, false);  // Az arduino cloud
szolgáltatását vettük ígénybe
```

```
  pinMode(Relaypin, OUTPUT);
  digitalWrite(Relaypin, LOW);
  pump_Status = false;

  pinMode(sensorPin, INPUT);

  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

void loop() {

  moist();

  if (soilMoisturePercent <= pump_trigger) {
  pumpOn();
  time_now = millis();

  while(millis() < time_now + periodShort)
  {
    moist();
    delay(1000);
    if(millis() >= time_nowUpdate + periodUpdate)
    {
       time_nowUpdate += periodUpdate;
       ArduinoCloud.update();
    }
  }
```

```
  pumpOff();
  time_now = millis();

  while(millis() < time_now + periodLong)
  {
    moist();
    delay(1000);
    if(millis() >= time_nowUpdate + periodUpdate)
    {
       time_nowUpdate += periodUpdate;
       ArduinoCloud.update();
    }
  }

  }
  else
  {
    delay(1000);
  }


  if(millis() >= time_nowUpdate + periodUpdate){
       time_nowUpdate += periodUpdate;
       ArduinoCloud.update();
  }


}

void pumpOn() {
  digitalWrite(Relaypin, HIGH);
```

```
  pump_status_text = "ON";
  pump_Status = true;
  moist();
  ArduinoCloud.update();

}

void pumpOff() {
  digitalWrite(Relaypin, LOW);
  pump_status_text = "OFF";
  pump_Status = false;
  moist();
  ArduinoCloud.update();
}

void moist() {
  soilMoistureValue = analogRead(sensorPin);
  soilMoisturePercent = map(soilMoistureValue, DryValue, WetValue, 0, 100);
// "map-oljuk" a %-ra
  soilMoisturePercent = constrain(soilMoisturePercent, 0, 100);
// hibakezelés 0-nál alacsonyabb illteve 100-nál magasabb % érték kiküszöbölése
  current_Moisture = soilMoisturePercent;
  Serial.println(soilMoistureValue);
}
void onTriggerLevelChange() {
  pump_trigger = trigger_Level;
}
```

Ez a src.ino teljes kódja, és ez lesz, ami ténylegesen tartalmazza a főbb dolgokat, melyeket a későbbiekben részletezünk majd.

void pumpOn() függyvény

```
void pumpOn() {
  digitalWrite(Relaypin, HIGH);
  pump_status_text = "ON";
  pump_Status = true;
  moist();
  ArduinoCloud.update();

}
```

Ez a függvény azt csinálja, hogy a relé portjára először is egy magas jelet küld, ami megnyitja a pumpát. Ezután a két sorban a pumpa led állapotát állítja ON-ra, ezután mér egyet a moist() függyvénnyel, aztán küld egy updatet a(z) Arduino Cloud - nak.

void pumpOff() függyvény

```
void pumpOff() {
  digitalWrite(Relaypin, LOW);
  pump_status_text = "OFF";
  pump_Status = false;
  moist();
  ArduinoCloud.update();
}
```

Ez a függvény azt csinálja, hogy a relé portjára először is egy alacsony jelet küld, ami lekapcsolja a pumpát. Ezután a két sorban a pumpa led állapotát állítja OFF-ra, ezután mér egyet a moist() függyvénnyel, aztán küld egy updatet a(z) IOT Cloud felületnek - nak.

void moist() függvény

```
void moist() {
  soilMoistureValue = analogRead(sensorPin);
  soilMoisturePercent = map(soilMoistureValue, DryValue, WetValue, 0, 100);  // "map-
oljuk" a %-ra
```

```
  soilMoisturePercent = constrain(soilMoisturePercent, 0, 100);          // hibakezelés 0-nál
alacsonyabb illteve 100-nál magasabb % érték kiküszöbölése
  current_Moisture = soilMoisturePercent;
  Serial.println(soilMoistureValue);
}
```

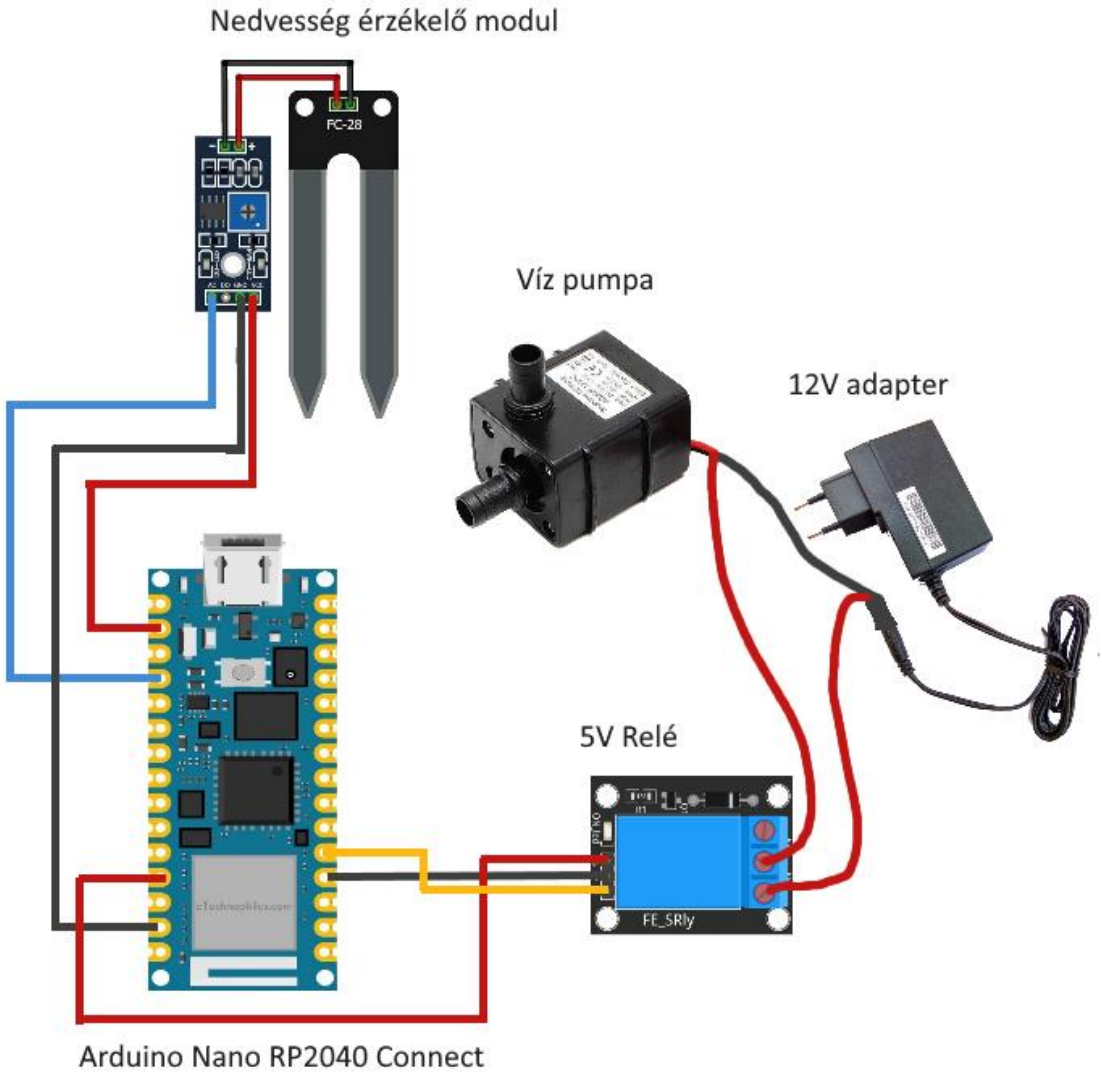
Ez a függvény az első sorában a nedvességérzékelőről olvassa le az értéket 400 és 1000 között (analogRead), ezután ezt 0 és 100 közötti értékbe fogja a következő két sorban, ezután az utolsó sorban az IOT Cloud felülettel kommunikál.

Működési vázlat szimuláció online:

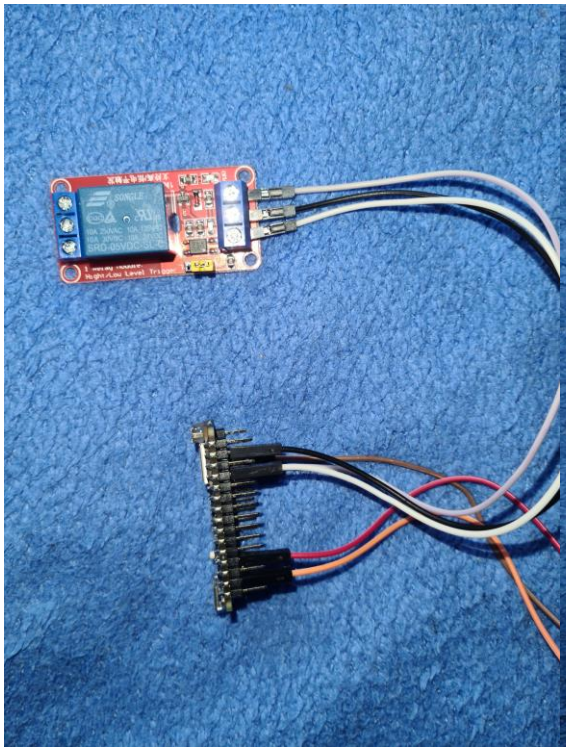
A működési vázlatok nagy részét online raktuk össze a circuit.io nevű oldal segítségével, mely nagyban megkönnyítette a munkánkat már a tervezési folyamatoktól egészen a hibakezelésig az utolsó simításoknál. Mivel majdnem minden alkatrész megtalálható volt ezen az oldalon ezért csak ajánlani tudjuk.

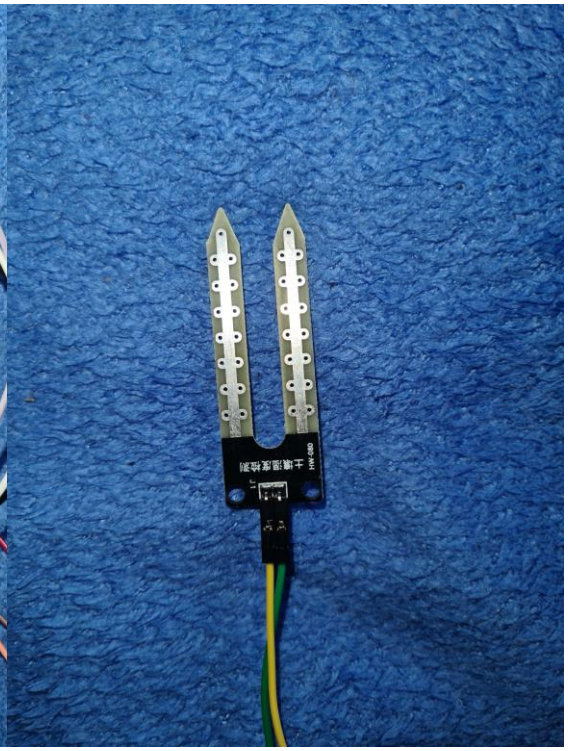
Az ábrát azonban kézzel lett elkészítve, mivel a circuit.io oldal “breadboard” felhasználásával rakta össze az áramkört, de mi annak a használata nélkül valósítottuk meg.

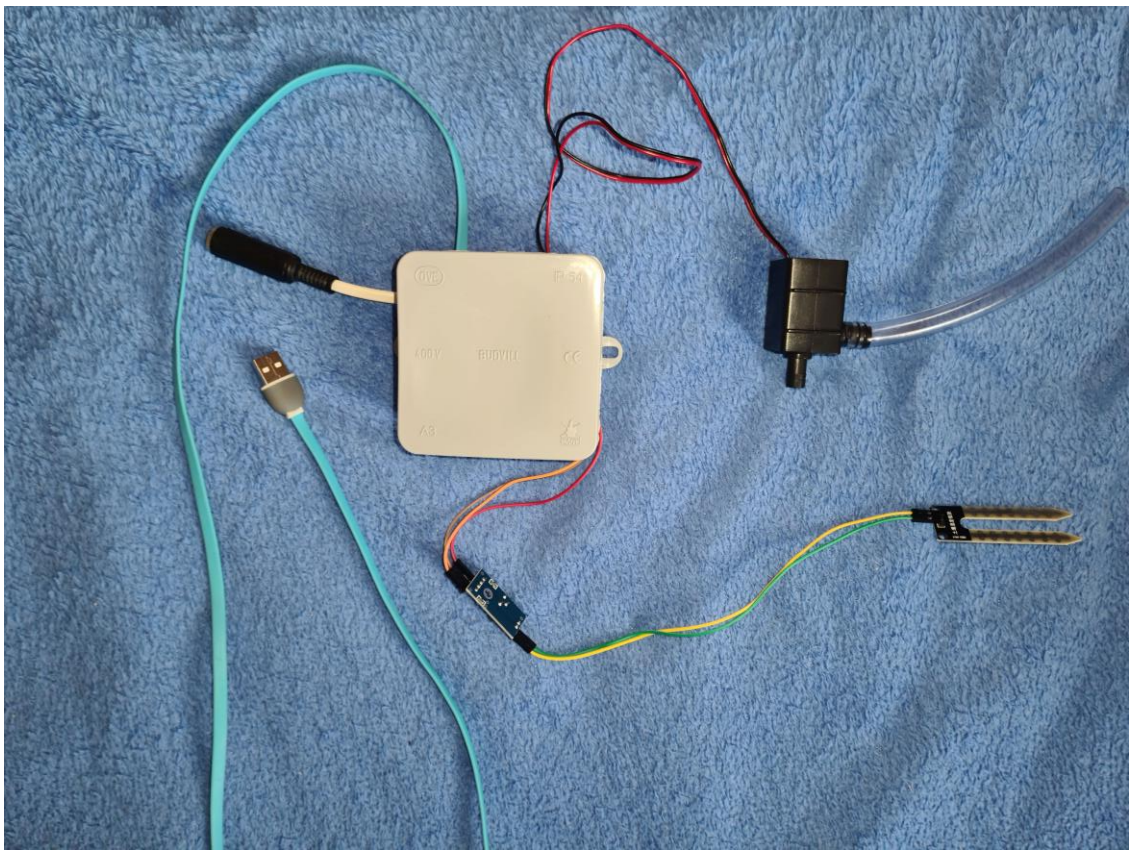
Kapcsolási rajz



Képek az eszközről







Fejlesztési lehetőségek

A most elkészült szerkezet egy prototípus megoldás, mely otthoni szakszerű használat mellett megfelelő nem túl vízigényes növényekhez.

Megfordult még a fejünkben:

- Vízállóság javítása, jelen állapotában nagyon oda kell figyelni, hogy az eszköz, érzékelőn kívüli részeihez ne érjen egy csepp víz sem.
- Amellett, hogy figyeljük a talajnedvességét, azt nem figyeljük, hogy a tartályunkban mennyi még a víz ezt a problémát most úgy küszöböltük ki, hogy elég nagy edényt alkalmazunk több locsolás alkalmára. Arra, hogy ezt mikor tölti újra a felhasználónak kell figyelnie. Egy köztes megoldás még az lehetne, hogy letiltjuk a pumpát, ha nincs mit tolnia és jelzést küldenénk a felhasználónak.
- Egyéb talajtípusok, bizonyos talajtípusoknál megfigyeltük, hogy a víz gyorsan elfolyik így az érzékelőnktől másmilyen működés lenne elvárt.

Készítette:

Jakab Kristóf Márk - BL2NDD

Kiss Vince Gergely – A2UWSD

Tantárgy kódja: GKNB_INTM020

Neve: Mikroelektromechanikai rendszerek