FELADATKIÍRÁS

A feladatkiírást a **tanszék saját előírása szerint** vagy a tanszéki adminisztrációban lehet átvenni, és a tanszéki pecséttel ellátott, a tanszékvezető által aláírt lapot kell belefűzni a leadott munkába, vagy a tanszékvezető által elektronikusan jóváhagyott feladatkiírást kell a Diplomaterv Portálról letölteni és a leadott munkába belefűzni (ezen oldal HELYETT, ez az oldal csak útmutatás). Az elektronikusan feltöltött dolgozatban már nem kell megismételni a feladatkiírást.



Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Stahorszki Péter Bence

Otthoni növénygondozási rendszer tervezése

Konzulens

BUDAPEST, 2017

Tartalomjegyzék

[Összefoglaló 5](#_Toc497913699)

[Abstract 6](#_Toc497913700)

[1 Bevezetés 7](#_Toc497913701)

[2 Irodalomkutatás 8](#_Toc497913702)

[2.1 Létező megoldások 8](#_Toc497913703)

[2.1.1 Edyn 8](#_Toc497913704)

[2.1.2 GreenIQ 10](#_Toc497913705)

[2.1.3 PlantLink 11](#_Toc497913706)

[2.1.4 Grove Smart Plant Care Kit 11](#_Toc497913707)

[2.1.5 GardenBot 12](#_Toc497913708)

[2.2 Hardver Technológiák 12](#_Toc497913709)

[2.2.1 Raspberry Pi 12](#_Toc497913710)

[2.2.2 Arduino 14](#_Toc497913711)

[2.3 Kommunikációs technológiák 15](#_Toc497913712)

[2.3.1 HTTP, HTTP/2 15](#_Toc497913713)

[2.3.2 WebSocket 16](#_Toc497913714)

[2.3.3 MQTT 16](#_Toc497913715)

[3 Tervezés 18](#_Toc497913716)

[4 Implementáció 19](#_Toc497913717)

[5 Tesztelés 20](#_Toc497913718)

[Irodalomjegyzék 21](#_Toc497913719)

[Függelék 22](#_Toc497913720)

Hallgatói nyilatkozat

Alulírott **Stahorszki Péter Bence**, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy hitelesített felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2017. 11. 15.

...…………………………………………….

Stahorszki Péter Bence

Összefoglaló

Ide jön a ½-1 oldalas magyar nyelvű összefoglaló, melynek szövege a Diplomaterv Portálra külön is feltöltésre kerül.

Abstract

Ide jön a ½-1 oldalas angol nyelvű összefoglaló, amelynek szövege a Diplomaterv Portálra külön is feltöltésre kerül.

* Bevezetés 1-2 oldal
* Irodalomkutatás 10-15 oldal
* Tervezés 10-15 oldal
* Implementáció 5-10 oldal
* Tesztelés 5 oldal
* Összefoglalás 1-2 oldal

# Bevezetés

Az információ technológia fejlődésével egyre mélyebben szivárog bele az emberek életébe a számítástechnika. Ahogy ezek az eszközök egyre több mindenre képesek, kényelmesebb a használatuk, az embereknek is megnövekedett az ilyen irányú igényük. Először a számítógép terjedt el a háztartásokban, amit leginkább szórakozásra használtunk, közben az járműgyártásába is beivódott, ahol rengeteg olyan kényelmi és biztonsági funkciót valósítottak meg vele, amik nélkül ma már igazán el se lehetne adni autót. Az ezredfordulú környékén pedig egyre több helyen vált elérhetővé az internet, és az internetre kapcsolódni képes eszközök, egy új fogalom jelent meg, az **Internet of Things (IoT)**,magyarul a dolgok internete.

Az IoT azon eszközök összessége, amelyek a bennük megtalálható szoftverek, szenzorok, elektronikák segítségével képesek adatokat gyűjteni, amelyeket a hálózaton keresztül képesek továbbítani. Az IoT technológia lehetővé teszi a világunk leképzését számítógépes rendszerbe, ezáltal rengeteg folyamat esetén csökkenthető az emberi beavatkozásnak a szükségessége, ezáltal az esetleges hibák száma is redukálható.

Az IoT egy nagyon érdekes területe az okosotthon, amelynek az otthoni folyamatok automatizálásáról szól, legyen szó szűkebb értelemben a fűtés, világítás, takarító-robot, szellőztetés szabályozásáról, tágabb értelemben pedig az adott háztartásra jellemző, nem mindenhol felmerülő feladatokról, akár egy állat rendszeres etetése, vagy mint esetemben, egy mobil növénygondozó rendszer.

Egy olyan rendszer tervezését, és építését választottam feladatul, ami képes lehet több növény egyidejű gondozására, emberi beavatkozás nélkül, emellett a növény környezetének állapotát rendszeresen elmenti, és egy webes felületen is megjeleníti. Természetesen akkor is működnie kell, amikor nincs internetes kapcsolat.

# Irodalomkutatás

Az interneten számos hasonló megoldásról lehet olvasni, nem egy ilyen termék megvásárolható, de rendelkezésre áll számos nyilvános forráskód is, amik segítségével az érdeklődő informatikus hobby kertész könnyebben elindulhat az automatizálás útján. Néhány terméket a közösségi finanszírozás[[1]](#footnote-1) útján akarták piacra dobni, a pénzt sikerült is megszerezni hozzá – szóval a vásárlói igény meg van - viszont ezek közül sok kudarcba fulladt. Ezekről rendszerekről olvashatunk az következő oldalakon.

## Létező megoldások

### Edyn[1]

Az Edyn egy moduláris rendszer, kerti alkalmazáshoz. Három részből áll, a **Garden Sensorból**, ami egy összetett egység, képes a fény, talajnedvesség, páratartalom és a talaj tápanyagtartalmának mérésére. Beépített WiFi modullal rendelkezik, ennek segítségével tud kapcsolatot tartani a rendszer többi részével. A **Water Valve** egy okos szelep, amit rá kell kötni a vízhálózatra, és a **Garden Sensor**, valamint az időjárási adataira alapozva megállapítja, hogy mennyi vizet juttasson a növényhez. A kert állapotát egy telefonos alkalmazáson keresztül lehet figyelemmel kísérni, a növény adatbázisban 5000 fajta jellemzői vannak tárolva, a felhasználó ezek közül választhatja ki a sajátját. Ezen a felületen megjelennek a helyszín időjárási adatai, valamint megtekinthetők a fény és nedvességi viszonyok változása napi, heti, havi és évi lebontásban. Az adatbázisban található információk, és a szenzorból nyert adatok alapján a rendszer megpróbálja meghatározni, hogy az adott növények milyen fejlődési stádiumban van. Ha fontos változás lép fel a kert, vagy egy növény állapotában, a felhasználó erről értesítést kap, ami vonatkozhat az esetleges kedvezőtlen körülményekre, vagy arról, hogy új milyen új növekedési fázisba lépett a növény. A rendszer arra is fel van készítve arra az esetre is, ha elveszítené az internetkapcsolatot. Folyamatosan tölti be előre a következő napok időjárási előrejelzések alapján a várhatóan szükséges öntözések időzítését. Működéséhez nem kell külső tápellátás, a beépített napelemnek és akkumulátornak köszönhetően ideális körülmények között akár teljesen töltés nélkül is képes operálni. Egy szenzorhoz több növényt is be lehet regisztrálni, és 25m2 nagyjából az a terület, amit egy egységgel le lehet fedni. Értelemszerűen, ha a kertünkbe többféle fényviszonnyal rendelkező terület van, vagy esetleg másfajta vízigényű növényeket szeretnénk egyszerre nevelni, akkor több szenzorra van szükség, és ha ezekhez szeretnénk külön öntözést, akkor annyi helyre kell venni egyet-egyet a szelepből is. Ez elég költséges lehet, tekintve hogy a Garden Sensor fogyasztói ára $100 dollár, a Water Valve-ért pedig $60 dollárt kell otthagynia a vásárlónak. Ez nem kevés pénz, viszont ezért egy elég összetett és jól átgondolt megoldást szállítanak. Ami szerintem még hiányzik ahhoz, hogy teljes legyen a rendszer, az a kiegészítő világítás arra az esetre, ha nem lenne elegendő a természetes fény, valamint az alkalmazás oldalán olvasott vélemények alapján a növényi adatbázis növelésére is szükség lenne. /\*Mindent összevetve egy nagyon elegáns megoldás. \*/



1. ábra: Garden Sensor



2. ábra: Water Valve

### GreenIQ [2]

A GreenIQ más irányból közelíti meg a kert automatizálását. Nem annyira a különböző növények specifikus igényeit veszi figyelembe, hanem a vízfelhasználás optimális szinten tartását. Célja az, hogy átvegye a szerepét a régi fajta beépített öntöző berendezéseknek, amik a felhasználók által lettek beprogramozva, valószínűleg nem elég hatékonyan. Egy az egyben kiváltja a már meglévő vezérlőt, és ha már a kert be van építve öntözőrendszerrel, nincs is szükség másra, mint felkonfiguráljuk, és csatlakoztassuk a régi egység helyére. WiFi-n kapcsolódik az internetre, és nem rendelkezik beépített akkumulátorral, így áramkimaradás esetén értelemszerűen szünetel a működés. Több öntözési zónát képes kezelni, valamint az internetről letöltött adatok alapján vezérelheti a kerti világítást is.

Az IoT koncepciót magas szinten valósítja meg. Létezik kliense a nagyobb platformokra, mint IOS, Android, és webböngészőn keresztül is használhatjuk. Nem csak egyszerű szenzorok (talajnedvesség, vízfolyás érzékelő) jeleit használhatja a legkedvezőbb öntözés meghatározásához, a helyi legközelebbi, vagy az esetleges saját időjárás állomás adatait is képes lekérdezni. Ezeken felül irányítható Amazon Echo, és Google Home segítségével is, ehhez az **IFTTT** technológiát használja. Ez a **If This Than That** rövidítése, ami egy web-alapú technológia, ha valami (this) történik, akkor végrehajtja az utasítást (that). Összekapcsolható okosotthon vezérlő alkalmazásokkal, mint a **Gideon[[2]](#footnote-2)** vagy a **Muzzley**, amik célja az, hogy a legtöbb okosotthon rendszernek nyújtsanak egy összpontosított felületet, ezzel könnyítve a felhasználó dolgát. Azon felül, hogy a kerti öntözést sokkal kényelmesebbé teszi, biztonsági funkciót is betölthet a GreenIQ. Ha a **Nest** otthonfigyelő rendszer füst érzékelője túl magas értéket mér, beindít egy öntözési ciklust az összes zónában, ezáltal nedvesen tartja a ház környékét, ezáltal segítve a tűz esetleges terjedésének a megakadályozását.



3. ábra: GreenIQ

### PlantLink [3]

A PlantLink kínálja a legkisebb funkcionalitást, cserébe a kedvezőbb árért, és a használat egyszerűségéért. Két részből áll, a központi egységből és a szenzorból. A központi egységre az otthoni routerre csatlakozik egy Ethernet hálózaton, hozzá pedig 64 szenzor csatlakozhat egyszerre, ezek az alacsony frekvenciás rádiójelnek köszönhetően elég távol, akár 100 méterre is lehetnek az állomástól. Egy szenzor felkonfigurálásához szükség van a talaj és növény típusára. Egyedül a talajnedvesség mérésére képes, ez alapján tesz ajánlást az öntözések időpontjára, viszont mennyiségére nem, ezért inkább tapasztaltabb kertészek számára lehet ideális. A kezdő csomag $79-be kerül, ami tartalmaz egy állomást és egy szenzort, ezen felül minden újabb szenzor plusz $35.



4. ábra: PlantLink

### Grove Smart Plant Care Kit

A következő példa inkább azoknak szól, akik szívesen barkácsolnak. A Grove Smart Plant Care Kit egy kevés részből álló készlet, amit a meglévő Arduino mikrokontrolleres fejlesztőkártyánkhoz vásárolhatunk meg. Azoknak ajánlott, akik most ismerkednek az Arduino világával. Aki megvásárolja, kézhez kap egy csomagot, ami tartalmazza a komponenseket, ami egy egyszerű öntöző és állapotmegfigyelő rendszerhez szükséges.

Nyílt forráskódú projekt, így tökéletes lehet olyan emberek számára, akik szeretnék megérteni egy ilyen rendszer működését, hogyan működik együtt az Arduino a szenzorokkal és beavatkozókkal, hogyan kell rá programot írni.



5. ábra: Grove Smart Plant Care Kit

## Hardver Technológiák

A fentebb felsorolt termékek már olyan célhardvert használtak, amit kifejezetten ezekre az alkalmazásokra fejlesztett a gyártó. Ha nagy mennyiségű termelés a cél, ennek a specifikus hardvernek a megalkotása fontos, mivel azzal, hogy csak a legkisebb szükséges halmazra redukálja a képességeit, nagyban csökkenti az előállítási költséget. Egy egységen a féldollárnyi költség megtakarítása elsőre nem tűnik olyan soknak, viszont ha ebből sikerül eladni mondjuk egy millió darabot, akkor máris fél millió dollárt spórolt a gyártó, amit sok fontos dologra költhet, fejlesztésre, terjeszkedésre, reklámra.

Számomra és a többi hobbista, építgetni szerető emberek számára más szempontok az érdekesek. Fontos általában, hogy az irányító egység sokoldalú legyen, mivel a projekteknél sok a próbálkozás, sokszor ki kell próbálni különböző megoldásokat. Emiatt, a hobbiépítők egyik legjobb barátai a nem régen megjelent kicsi, egyetlen áramköri lapra integrált számítógépek. Rendelkeznek processzorral, memóriával, háttértárral, szóval mindennel, amit egy teljesen funkcionális számítógépnek tartalmaznia kell. Ezek általában nagyjából bankkártya méretű kis teljesítményű gépek, amelyeken teljesértékű számítógépes operációs rendszerek futnak, ami általában Linuxnak valamely változata, valamint változatos IO-val szerelik őket. A funkcionalitásukhoz képest olcsók, a kis méret miatt könnyen kezelhetők, sok helyre beépíthetők, a számítógépes OS miatt pedig a felhasználási lehetőségeinek csak a programozójának képességei szabnak határt.

### Raspberry Pi

A legelterjedtebb ilyen eszköz a Raspberry Pi. Ezt a kis komputert eredetileg oktatási célból fejlesztették ki, hogy bevezessék az iskolásokat a számítástechnika világába és, hogy olcsóságával a fejlődő országokban is alternatíva legyen az egyszerűbb, számítógépet igénylő feladatok megoldásához. A termék sokkal sikeresebb lett, mint várták, mivel képességei nem csak oktatási célra teszik alkalmassá, hanem akár robotok, beágyazott okos rendszerek alapjának is. 2012-ben adták ki az első Pi-t, és öt éves pályafutása alatt több mint 12 milliót adtak el belőle, ezzel elnyerve a legtöbbet eladott brit számítógép, és a világ harmadik legtöbbet eladott általános célú számítógép címet. Jellemzően a Linuxnak valamelyik variánsa fut rajta, de az újabbakra már a Windows 10 IoT Core operációs rendszert is feltelepíthetjük.

Többfajta kiadásban találkozhatunk a Raspberry Pivel. Az általánosabb felhasználásra szánt típus a kezdetekben Model A, majd Model B ma a harmadik iterációnál jár. Nagyjából bankkártya méretű, és rendelkezik a legtöbb olyan input/output lehetőséggel, amit megszoktunk egy PC vagy laptop esetében.

Az első generációs modellt egy magos, 700 MHz-es processzorral, 256 MB RAM-mal, USB, video és audio csatlakozókkal szerelték, kezdetben hálózati port nélkül. Emellett rendelkezett GPIO, azaz általános célú input/output tüskékkel, amelyeknek a viselkedését a felhasználó futási időben vezérelheti, megfigyelheti. Ezen tüskék között találhatóak tápellátáshoz szükséges csatlakozók, így alacsonyabb energiaigényű eszközöket könnyen irányíthatunk segítségükkel. Az operációs rendszer, és háttértár egy cserélhető SD kártyán foglal helyet.



6. ábra: Raspberry Pi 3

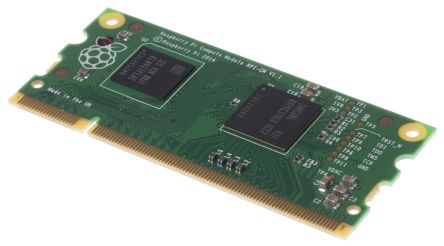
A harmadik generációs Raspberry Pi már négy magos, 1.2 GHz-es processzorral, 1GB RAM-mal, beépített WiFi modullal rendelkezik. Fogyasztói ára $35, ennyi pénzért egy olyan számítógépet kapunk, melynek hardveres képességei lehetővé teszik, hogy akár fullHD médiakiszolgálóként, dedikált játékgépként használjuk, de volt, aki felhő szervert csinált belőle.

Létezik egy **Zero** nevű típus is, ami még kisebb, mint a Model A/B család, ezáltal kisebb teljesítményű, kevesebb porttal rendelkezik, kis mérete és súlya viszont alkalmassá teszi arra, hogy olyan helyeken alkalmazzuk, ahol szűkek a méret vagy súly korlátok, például okos drónok irányítására. Alapára $5 dollár, $10 dollár a beépített WiFi modullal rendelkező verzióé, szóval az okos projektjeink vezérlését olcsón megoldhatjuk.



7. ábra: Raspberry Pi Zero

Ezek mellett még forgalmaznak egy **Compute Module** nevű típust, ami formára megegyezik a SO-DIMM memóriákkal, viszont a lábkiosztása értelemszerűen teljesen más. 4GB beépített tárhellyel rendelkezik, és megvásárlása esetén kapunk mellé egy IO Board eszközt, aminek segítségével fel tudjuk programozni. Ezt azok számára fejlesztették ki, akik saját nyomtatott áramkört terveznek, így biztosan a lehetséges legkisebb helyet foglalja el belőle a Pi.

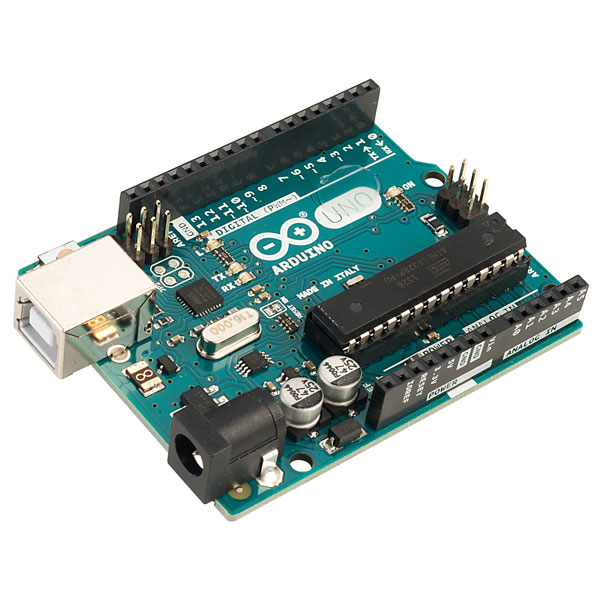


8. ábra: Raspberry Pi Compute Module

### Arduino

Sok olyan projekt van, amikor nincs szükség akkora számítási kapacitásra, mint amivel például a Raspberry Pi rendelkezik, szeretnénk valami egyszerűbben programozhatóbb vezérlő egységet, kevesebb a hely a vezérlőnek, valós időben beavatkozásra képes alkalmazást szeretnénk írni, vagy nagyobb mennyiségű GPIO tűre van szükségünk, esetleges analóg bemenetekre, amivel a Pi nem rendelkezik. Ilyenkor kerülnek képbe a mikrokontrollerek.

Általában olyan helyeken alkalmazzák, amikor a környezetben belépő változásra gyorsan kell reagálni. Általában nem operációs rendszer, hanem az adott feladatra készített program fut rajtuk. Programozásuk történhet gépi kódon, saját programnyelv, vagy magas szintű programnyelven keresztül, ez attól függ, hogy milyen fejlesztőrendszert építettek az adott mikrokontroller köré.



9. ábra: Arduino Uno

A ma leginkább elterjedt mikrokontrolleres fejlesztőkártya az Arduino. Ezek az Atmel 8-bites mikrokontrollerei köré épülnek. Egyszerűen programozható az Arduino IDE segítségével, ami felismeri a számítógéphez csatlakoztatott Arduino eszközöket, a kód validálása után pedig fel is tölthetjük a programunkat rajta keresztül.

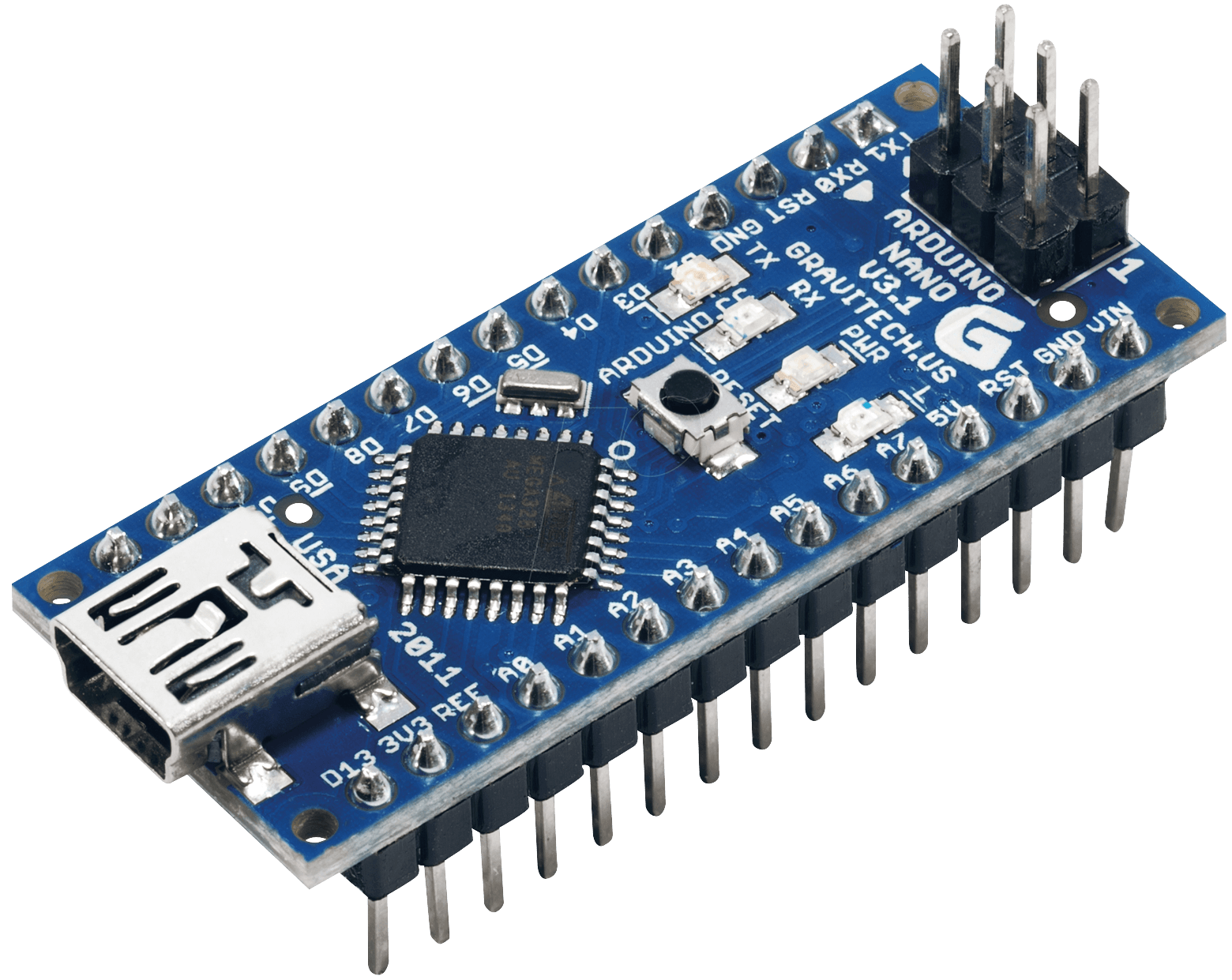
Népszerűségét több dolognak is köszönhet. Egyik ilyen, hogy nem kell hozzá külön programozó áramkör. A bootloader egy olyan program, ami minden bekapcsolás után lefut, egy bizonyos byteszekvenciát vár a soros porton. Ha megérkezik ez a bytesorozat, akkor programozza a mikrokontrollert, ha ezt nem érkezik meg, akkor betölti a már rátelepített programot. Ezáltal könnyebb a fejlesztés, viszont érdemes figyelembe venni, hogy ez minden indításnál késleltetést jelent, ami valós idejű alkalmazások esetén fontos lehet.

Egy Arduino program minimum két függvényből áll, ez a setup() és a loop(). A setup() egyszer fut le, a program indulásakor, itt konfiguráljuk a mikrokontroller lábainak viselkedését. A loop() függvény tartalmazza a végrehajtandó parancsokat, folyamatosan fut.

Rengetegen használják, emiatt nagyon sok open-source library található hozzá mindenféle részfeladatra, amik segítségével néhány sor kód segítségével egészen komplex rendszereket lehet megvalósítani.

Ha kevésnek találnánk az Arduino képességeit, könnyen bővíthetjük az úgynevezett Shieldek segítségével. A Shieldek egyszerű interfacet adnak az Arduino számára komplex hardveres technológiák használatához, mint például Bluetooth, WiFi, LCD kijelző. Olyan áramköri lapok, amik méretre és lábkiosztásra megegyeznek az Arduinoval, így csak rá kell helyezni, és akár több ilyen Shieldet is használhatunk egyszerre.

A Raspberryhez hasonlóan az Arduino is megkapható több formában, eltérő specifikációkkal. Az alap Uno mellett létezik például a Nano, ami sokkal kisebb helyet foglal, a Mega, ami nagyobb, és sokkal több lábkivezetést tartalmaz, vagy az Ethernet, ami beépített internetes interfésszel rendelkezik. Az eredeti Arduino is elég olcsó, €20 körül kapható, viszont mivel nyilvánosak a hardveres tervrajzai, sokan gyártanak belőle másolatokat, amik ennek akár az ötödébe is kerülhetnek.



0. ábra: Arduino Nano

### Szenzorok

asdasd

## Kommunikációs technológiák

Amikor olyan eszközöket készítünk, amivel interneten keresztül szeretnénk kommunikálni gondolnunk kell arra, hogy mi a legmegfelelőbb technológia erre a feladatra.

/\*

<https://www.linkedin.com/pulse/internet-things-http-vs-websockets-mqtt-ronak-singh-cspo>

<https://hu.wikipedia.org/wiki/HTTP>

\*/

### HTTP, HTTP/2

A HyperText Transfer Protocol egy kérés-válasz alapú kommunikációs protokoll kliens és szerver között. Első verzióját 1991-ben alkották meg, a legfrissebb a 2015-ben kiadott HTTP/2. A kliens egy kérést küld a szerver felé, a végrehajtani kívánt metódussal és paraméterekkel. Ezt a kérést a szerver feldolgozza, majd az eredményt válaszüzenetben visszaküldi a kliensnek. Ez az üzenet tartalmazza a státuszt a kérés teljesítéséről, valamint a kért tartalmat, és egyéb információkat. Egy megbízható szállítási protokollt igényel, ez a legtöbb esetben a TCP, így a QoS (Quality of Service) felelősséget nem vállal.

A régebbi HTTP protokollok szöveges üzenetekkel, és soros üzenetfeldolgozással működtek, viszont az új HTTP/2 már bináris adatot használ, és a kéréseket multiplexálással dolgozza fel. Ezáltal csökken az üzenet mérete, és a feldolgozás ideje. Újdonság még a HTTP/2-ben a szerver oldali „push” művelet, aminek segítségével lehetőség van arra, hogy akkor is küldjünk adatot a kliensnek, amikor nem érkezett kérés.

Bár a HTTP/1 is használható volt az IoT eszközök esetében, az új HTTP/2 verzió alkalmasabb a feladatra, tekintve, hogy sokkal kisebb az erőforrásigénye, ami fontos, ha egy kis teljesítményű hardverrel, vagy akkumulátorral szerelt eszközről van szó.

### WebSocket

A WebSocket protokoll a kliens és a szerver között létesít kétirányú kapcsolatot, ezáltal ideálissá teszi, ha egy rendszer állapotát szeretnénk folyamatosan megfigyelni, vagy ha egy kliensnek rendszeresen, és gyors frissítésekre van szüksége. Ahhoz hogy létrejöjjön egy ilyen kapcsolat, a kliens egy „kézfogás” üzenetet küld a szervernek HTTP-n keresztül, a szerver válaszával pedig felépül a közvetlen kapcsolat a kliensalkalmazás és a szerver között.

2008-ban kezdték a használatát, a fő motiváció a létrejöttére az volt, hogy, a HTTP/1-ből hiányzott a kétirányú kapcsolat. Bár a HTTP/2-ben ez bevezetésre került, a WebSocket így sem válik szükségtelenné, mivel bár a HTTP/2 push művelete eljuttatja a kliensgépre az üzenetet, de nem a kliensalkalmazáshoz, a WebSockettel ellentétben, így még egy lépésre szükség van.

### MQTT

A Message Queue Telemetry Transport egy kis erőforrásigényű protokoll, amit az IBM fejlesztett ki a gép-gép közötti kommunikáció megkönnyítésére, azokra az esetekre, amikor fontos, hogy a kapcsolat fenntartásához a lehető legkisebb kódot használjuk, vagy a sávszélesség erősen korlátozott. Működéséhez szükség van egy harmadik, köztes szereplőre, az üzenetközvetítőre. Az ő szerepe az, hogy a küldő által alkalmazott üzenetküldési protokollról átfordítsa az üzenetet a fogadó protokolljára.

Háromszintű QoS-t definiál, arra vonatkozóan, hogy az üzenetközvetítő/kliens mennyi felelősséget vállal az üzenet kézbesítésére.

1. A közvetítő/kliens egyszer küldi el az üzenetet, nem vár megerősítésre.
2. A közvetítő/kliens legalább egyszer elküldi, és megerősítésre vár.
3. A közvetítő/kliens egyszer küldi el az üzenetet, négy-utas[[3]](#footnote-3) kézfogást alkalmazva a biztos átvitel érdekében.

Az üzeneteket bármilyen szinten lehet küldeni, és a kliensek akármilyen szintű „témára” feliratkozhatnak. Ez azt jelenti, hogy ha a kliens QoS 0 szinten iratkozott fel, de QoS 2 szintű az üzenet, azt QoS 0 szinten fogja megkapni.

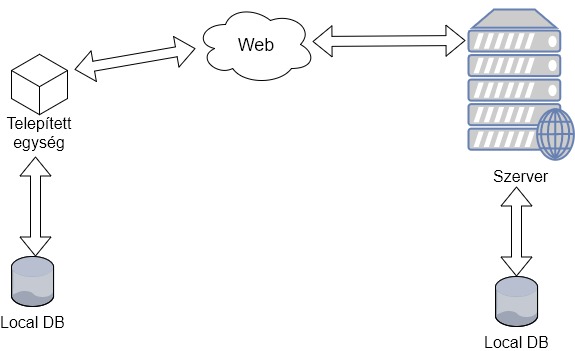
Jelenleg ez a legkisebb erőforrásigényű a széles körben elterjedtek protokollok között, így ha erősen korlátoltak az eszköz hardveres képességei, esetleg akkumulátorról működik, vagy gyenge, megbízhatatlan az adatkapcsolat érdemes számításba venni.

# Tervezés

A célom ezzel a projekttel egyrészt az volt, hogy megismerkedjek az IoT világával, és kipróbáljak pár olyan technológiát, amivel eddig még nem volt szerencsém foglalkozni, ezért lehet, hogy a megoldásom nem lesz optimális.

A feladat egy olyan rendszer megtervezése, ami önállóan képes több növény felügyelésére és gondozására. Ezen felül legyen lehetőség a növények állapotát távolról megfigyelni, beavatkozni.

Ezek szerint két részből kell állni a rendszernek, egy egységből, ami a növények mellett van, és egy webszerverből. A növény mellé telepített egység kötelességei a növény állapotának folyamatos rögzítése, ezek alapján a gondozási teendők elvégzése, az adatok továbbítása a webszerver felé. A webszerver feladata az adatok tárolása, megjelenítése könnyen értelmezhető formában, és felhasználói tevékenységek kezelése.



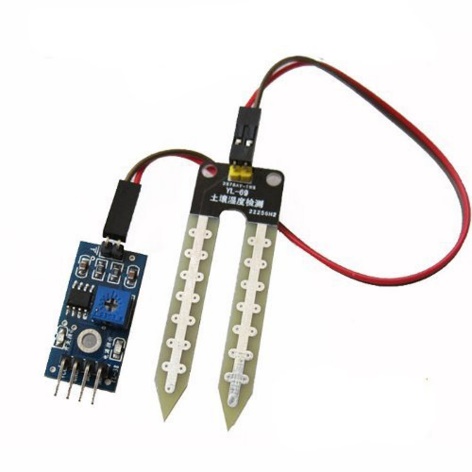
11. ábra: Kommunikációs diagram

## A növények gondozása

A két legfontosabb dolog a növény számára a víz, és a fény, ezeket kell egy optimális szinten tartani, ahhoz, hogy megfelelő legyen a növekedés. Ahhoz, hogy megállapítsuk a környezet alapján a gondozási teendőket, szükség lesz érzékelőkre, egy talajnedvességmérőre, és egy fényszenzorra növényenként, valamint hőmérséklet mérőre megfigyelés szempontjából.

Ezekből van olyan, aminek csak bináris kimenettel rendelkezik, és van, amelynek van analóg kimenete is. Nekem az utóbbira lesz szükségem, a bináris ugyanis csak 0 és 1 értékeket ad vissza. A potenciométerrel meg lehet adni azt a szintet, ami felett 1-es jelet ad, így jó lehet olyan alkalmazások esetén, amikor ez a szint nem változik. Az én rendszeremet úgy tervezem, hogy többféle növény gondozására is képes legyen, ezért az analóg szenzort választom, analóg jelet ad vissza, amit a fogadó eszköz ADC-je alakít át digitális jellé Minél több bites az ADC, annál pontosabb a mérés, 10 bites esetén 0-1023 az értékkészlet, 16 bites esetén 0-65535.

A talajnedvesség nem változik hirtelen, ezért úgy gondolom, hogy ha 10 percenként mérem meg, az elegendő.



12. ábra: Talajnedvesség érzékelő

A fénymérés már nem ilyen egyszerű. A növény állapotának tárolásához egy táblázatot tervezek használni, viszont ha a fényt csupán 10 percenként mérem meg, akkor könnyen lehet, hogy az adatok fals eredményhez vezetnének. A nap általi megvilágítás a felhők vonulása miatt gyakrabban változhat, mint a talajnedvesség, azaz a mért jel frekvenciája nagyobb, így nagyobb mintavételi frekvenciára van szükség. Ezért úgy terveztem, hogy a fényt sokkal sűrűbben mérem, mint a talajnedvességet, 10 másodpercenként, majd 10 percenként megnézem hány százaléka van a mintáknak azon érték felett, ami már megfelelő a növény számára. Ez a százalék elég pontosan visszaadja, hogy a 10 perc alatt hány percig sütött a nap megfelelő mértékben, mivel 10 másodperc alatt ritkán változnak drasztikusan a fényviszonyok. Ehhez a 10 másodperces mérésekhez egy külön táblázatot hozok létre, amiből 10 percenként törlöm az értékeket.



13. ábra: Fényérzékelő

Az öntözéshez a vízellátást először úgy terveztem, hogy egy keret tetején lesz a tartály, amiből egy elektromos szelep segítségével engedem ki a vizet, viszont a megvalósítás során rájöttem, hogy ez túlságosan körülményes, bekorlátozza a növény magasságát, és az se mindig egyértelmű, hogy mennyi vizet ereszt át. Ezért inkább a növény mellé helyezem a tartályt, és egy kisméretű szivattyúval szívom fel belőle a vizet. Ahhoz, hogy a felhasználó tudja, mikor kell tölteni a tartályba, egy ultrahangos távolságmérőt helyezek a tetejére, ami a kibocsát egy hullámot, és a visszaverődő hullám beérkezési ideje alapján kiszámítja a távolságot, emiatt ügyelni kell arra, hogy merőleges legyen a vízfelszínre az elhelyezése.



14. ábra: Ultrahangos távolságérzékelő

Arra az esetre, ha nem volt elegendő a napi fénymennyiség, a hiányzó részt egy növény nevelő lámpával biztosítanám. A növényhez tárolva lesz a kívánt a napsütéses órák száma, így az adatokból könnyen kiszámítható, mennyi fényre van még szüksége.

A fotoszintézisnek két szakasza van, a fényszakasz és a sötétszakasz, a sötétszakasz a fényszakaszban megtermelt anyagok felhasználásával állítja elő a glükózt, ami a növény tápanyaga. Ahhoz, hogy ez a leghatékonyabban működjön, arra kell törekedni, hogy az egyes szakaszok ne váltogassák egymást túl sűrűn, azaz, ha a növényről elkerül a fény, mondjuk 19:00-kor, ne 23:00-kor kapcsoljuk be a világítást, ezzel kirángatva a növényt a sötét szakaszból. Ezt úgy oldom meg, hogy minden növényhez be lehet állítani, mikor van a napfelkelte, mikor a naplemente, és ezek közötti számolja az aznapi fénymennyiséget. A naplemente időpontjában összegzem a fénymennyiséget, és ha kell, bekapcsolom a lámpát, így nagyobb az esély, hogy a növény folytatni tudja a fényszakaszt. Azért nem éjféltől kezdem az összegzést, mivel ha mondjuk egyik nap egyáltalán nem sütött a nap, és este 19:00-kor elindít egy 8 órás világítást, akkor az már átcsúszik a következő napra, így az összegbe is beleszámítana.

## Telepített egység

A telepített egység iránt támasztott követelmények a következők. Képesnek kell lennie kapcsolódni az internetre. Viszonylag nagy tárhellyel kell, hogy rendelkezzen, hogy tárolni tudja az adatokat. Az adatokat valamilyen adatbázisban kell tárolni, hogy könnyű legyen vele dolgozni, és megfelelően legyen strukturálva. Tudni kell adatokat gyűjteni a növényről, ehhez érzékelőkre lesz szükség. A növénygondozási feladatokhoz szükség lesz pumpára, vagy szelepre, valamint a megvilágításhoz lámpára, ezeket tudni kell ki-be kapcsolni.

# Implementáció

Implementáció

# Tesztelés

Tettelés

Miután elkészültünk a dokumentációval, ne felejtsük el a következő lépéseket:

* Kereszthivatkozások frissítése: miután kijelöltük a teljes szöveget (Ctrl+A), nyomjuk meg az F9 billentyűt, és a Word frissíti az összes kereszthivatkozást. Ilyenkor ellenőrizzük, hogy nem jelent-e meg valahol a "Hiba! A könyvjelző nem létezik." szöveg.
* Dokumentum tulajdonságok megadása: a dokumentumhoz tartozó meta adatok kitöltése (szerző, cím, kulcsszavak stb.). Erre való a Dokumentum tulajdonságai panel, mely a Fájl / Információ / Tulajdonságok / Dokumentumpanel megjelenítése úton érhető el.
* Kinézet ellenőrzése PDF-ben: a legjobb teszt a végén, ha PDF-et készítünk a dokumentumból, és azt leellenőrizzük.

Irodalomjegyzék

1. Edyn: <https://www.edyn.com/>
2. GreenIQ: <http://greeniq.co/>
3. Plantlink: <http://myplantlink.com/>
4. Gideon: <http://www.gideon.ai/>
5. Muzzley: <https://smarthome.muzzley.com/>
6. Grove Smart Plant Care Kit: <https://www.seeedstudio.com/Grove-Smart-Plant-Care-Kit-for-Arduino-p-2528.html>
7. Wikipedia: Evaluation strategy, <http://en.wikipedia.org/wiki/Evaluation_strategy> (revision 18:11, 31 July 2012)

Függelék

1. Közösségi finanszírozásról beszélünk, amikor egy projektre úgy gyűjtik össze a szükséges pénzt, hogy sok ember járul hozzá kisebb összeggel, tipikusan interneten keresztül. [↑](#footnote-ref-1)
2. Gideon [4] és Muzzley [5] mindkettő olyan alkalmazás, amin keresztül az otthonunkban található okos eszközöket irányíthatjuk. [↑](#footnote-ref-2)
3. A négy-utas kézfogás során egy üzenet küldése, amikor a fogadó egy PUBLISH üzenetet kap, visszaküld egy PUBREC (publish redeived) üzenetet, és eltárolja az üzenetben kapott csomag azonosítóját, hogy elkerülje az esetleges dupla feldolgozást, ha többször kapná meg az üzenetet. A küldő a PUBREC üzenetre egy PUBREL (publish release) üzenettel válaszol, amint ezt a fogadó megkapja, törölheti a tárolt állapotokat, és visszaküldi a PUBCOMP (publish completed) üzenetet. Amikor a küldő megkapja, ezt az üzenetet, ő is törölheti az üzenet adatait. [↑](#footnote-ref-3)