

---






**UNIVERSITATEA „SAPIENTIA” DIN CLUJ-NAPOCA**  
**FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE,**  
**TÎRGU-MUREȘ**  
**SPECIALIZAREA CALCULATOARE**

**SISTEM INTELIGENT DE IRIGARE**  
**ȘI ILUMINARE**  
**PROIECT DE DIPLOMĂ**

**Coordonator științific:**  
**Conf. dr. ing. Bakó László**

**Absolvent:**  
**Soós Izabella**

**2023**

UNIVERSITATEA „SAPIENTIA” din CLUJ-NAPOCA Facultatea de Științe Tehnice și Umaniste din Târgu Mureș Specializarea: Calculatoare		Viza facultății: 
<b>LUCRARE DE DIPLOMĂ</b>		
Coordonator științific: Conf. dr. ing. Bakó László	Candidat: Soós Izabella Anul absolvirii: 2023	
<b>a) Tema lucrării de licență:</b> <b>SISTEM INTELIGENT DE IRIGARE ȘI ILUMINARE</b>		
<b>b) Problemele principale tratate:</b> <ul style="list-style-type: none"> <li>- Studiu bibliografic privind sisteme de control automat a îngrijirii plantelor</li> <li>- Achiziția de date de la senzorii de temperatură, umiditate și debitmetru</li> <li>- Comanda unei pompe de apă pentru realizarea irigației respectiv a unei matrice de LED-uri pentru implementarea iluminării controlate</li> <li>- Software de comandă a sistemului realizat pe microcontroler (de ex. Arduino sau ESP32)</li> <li>- Comunicație wireless prin WiFi cu un sistem de calcul care rulează o bază de date și interfață grafică accesibilă prin web.</li> </ul>		
<b>c) Desene obligatorii:</b> <ul style="list-style-type: none"> <li>- Schema bloc a sistemului hardware-software realizat</li> <li>- Scheme de conectare a componentelor hardware</li> <li>- Scheme UML ale programelor realizate</li> </ul>		
<b>d) Softuri obligatorii:</b> <ul style="list-style-type: none"> <li>- Software embedded pe microcontroler pentru comandă sistemului automat</li> <li>- Proiectarea și implementarea unei baze de date pentru stocarea profilurilor și a datelor măsurate</li> </ul>		
<b>e) Bibliografia recomandată:</b> <ol style="list-style-type: none"> <li>1. Mikroprocesszorok, mikroszámítógép elemek / Madarász László, Kecskeméti Főiskola Műszaki Főiskolai Kar, 1998, 004.3/MAD.</li> <li>2. A. Tanenbaum: Számítógép architektúrák. (Arhitectura calculatoarelor) Bp., Panem Könyvkiadó, 2004.</li> <li>3. Gamma, Erich, Helm, Richard, Johnson, Ralph, Vlissides, John, Programtervezési minták, Kiskapu 2004.</li> </ol>		
<b>f) Termene obligatorii de consultații: săptămânal</b>		
<b>g) Locul și durata practicii:</b> Universitatea „Sapientia” din Cluj-Napoca, Facultatea de Științe Tehnice și Umaniste din Târgu Mureș Primit tema la data de: 31.03.2022 Termen de predare: 28.06.2023		
Semnătura Director Departament 	Semnătura responsabilului programului de studiu 	
Semnătura coordonatorului 	Semnătura candidatului 	

---

## Declarație

Subsemnata/ul SOÓS IZABELLA, absolvent(ă) al/a specializării CALCULATOARE, promoția 2019-2023 cunoscând prevederile Legii Educației Naționale 1/2011 și a Codului de etică și deontologie profesională a Universității Sapientia cu privire la furt intelectual declar pe propria răspundere că prezenta lucrare de licență/proiect de diplomă/disertație se bazează pe activitatea personală, cercetarea/proiectarea este efectuată de mine, informațiile și datele preluate din literatura de specialitate sunt citate în mod corespunzător.

Localitatea, CORUNCA

Data: 28.06.2023

Absolvent

Semnătura



---

# Sistem inteligent de irigare și iluminare

## Extras

În viața omului, una dintre cele mai importante și valoroase lucruri este timpul. În lumea agitată de astăzi, evenimentele s-au accelerat atât de mult încât este o adevărată provocare să le menținem sub control și să ne gestionăm timpul eficient. Pentru a economisi timp, au apărut numeroase dispozitive și sisteme inteligente, care le ușurează viața oamenilor. Putem chiar spune că trăim în era sistemelor inteligente.

Tema tezei mele cuprinde un astfel de sistem. Acesta nu numai că ne ajută să economisim timp, dar ne și eliberează de o responsabilitate. Această soluție modernă și inovatoare ne permite să avem grijă de plantele noastre în cel mai bun mod posibil, minimizând în același timp sarcinile de îngrijire și maximizând sănătatea și frumusețea florilor noastre. Sistemul ajută la menținerea plantelor în condiții optime, mai ales atunci când nu suntem acasă sau nu avem suficient timp pentru îngrijirea lor. Senzorii și controlerele sistemului monitorizează parametrii de mediu al plantelor, cum ar fi temperatura, umiditatea, intensitatea luminii și necesitățile de udare. Sistemul de ghiveci inteligent reglează automat udarea, asigurându-se că plantele primesc întotdeauna cantitatea potrivită de apă la momentul potrivit. Nu mai trebuie să ne facem griji în legătură cu udarea excesivă sau insuficientă, deoarece sistemul se ocupă cu precizie de udarea florilor noastre.

Prin intermediul unei interfețe interactive pentru utilizator, putem seta cu ușurință programul de udare, monitoriza starea plantelor și urmări parametrii de mediu. Indiferent dacă suntem acasă sau plecați, sistemul permite accesul de la distanță, permițându-ne să rămânem conectați la florile noastre și să intervenim atunci când este necesar.

Pentru controlul sistemului, am utilizat microcontrolerul ESP32. ESP32 este o platformă extrem de versatilă și puternică, care permite controlul complet și monitorizarea sistemului meu de ghiveci inteligent.

**SAPIENTIA ERDÉLYI MAGYAR  
TUDOMÁNYEGYETEM  
MAROSVÁSÁRHELYI KAR  
SZÁMÍTÁSTECHNIKA SZAK**

**OKOS ÖNTÖZŐ ÉS VILÁGÍTÁSI  
RENDSZER**

**DIPLOMADOLGOZAT**

**Témavezető:**

**Conf. dr. ing. Bakó László**

**Végzős hallgató:**

**Soós**

**Izabella**

**2023**

---

# Kivonat

Az ember életében az egyik legértékesebb és legfontosabb dolog az idő. A mai rohanó világban annyira felgyorsultak az események, hogy valódi nehézséget jelent kézben tartani őket és jól menedzselni az időnket. Az időmegtakarítás céljából számos okos készülék és rendszer jelent meg, megkönnyítve az emberek életét. Akár úgy is fogalmazhatunk, hogy az okos rendszerek korát éljük.

A dolgozatom témája is egy ilyen rendszert foglal magába. Nemcsak időt spórol a felhasználóknak, hanem egy felelősségtől is megszabadítja őket. Ez a modern és innovatív megoldás lehetővé teszi, hogy gondoskodjunk növényeinkről a legmegfelelőbb módon, miközben minimalizáljuk a gondozási feladatokat és maximalizáljuk a virágok egészségét és szépségét. Segít, hogy a növények optimális állapotát fenntartsuk, különösen akkor, amikor távol vagyunk otthonunktól vagy nem tudunk elég időt fordítani a gondozásra. Az érzékelők figyelik a környezeti paramétereket, azaz a hőmérsékletet és nedvességtartalmat. Az okos virágtartó rendszer automatikusan szabályozza az öntözést és világítást, biztosítva a növények számára a megfelelő mennyiségű vizet és hőmérsékletet a megfelelő időben. Nem kell többé aggódnunk a túl vagy alulöntözés miatt, hiszen a rendszer precízen és pontosan gondoskodik a virágaink vízellátásáról.

A felhasználói felület segítségével könnyedén beállíthatjuk az öntözési ütemtervet, profilokat hozhatunk létre, nyomon követhetjük a növények állapotát és a környezeti paramétereket. Akár otthon vagyunk, akár távol, a rendszer távoli elérést biztosít, hogy mindig kapcsolatban maradhassunk a növényeinkkel és szükség esetén beavatkozhatunk.

A rendszer vezérléséhez az ESP32 mikrovezérlőt használtam, ami egy rendkívül sokoldalú és erőteljes platform és feladata a rendszer teljes körű vezérlése és felügyelete.

**Kulcsszavak:** felhasználói felület, okos rendszer, mikrovezérlő

---

# Abstract

In human life, one of the most important and valuable things is time. In today's fast-paced world, events have become so accelerated that it is truly challenging to keep them under control and manage our time effectively. To save time, numerous smart devices and systems have emerged, making people's lives easier. We can even say that we are living in the era of smart systems.

My thesis' topic encompasses such a system. It helps us not only to save time, but also relieves us from a responsibility. This modern and innovative solution allows us to take care of our plants in the best possible way while minimizing the tasks of plant care and maximizing the health and beauty of our flowers. It assists in maintaining the plants in optimal condition, especially when we are not at home or don't have enough time for their care. The system's sensors and controllers monitor the environmental parameters of the plants, such as temperature, humidity, light intensity, and their watering needs.

The smart flowerpot system automatically regulates watering, ensuring that the plants always receive the right amount of water at the right time. We no longer need to worry about overwatering or underwatering, as the system takes precise care of watering our flowers.

Through an interactive and user interface, we can easily set the watering schedule, monitor the plants' condition, and track the environmental parameters. Whether we are at home or away, the system allows remote access, enabling us to stay connected with our flowers and intervene when necessary.

For the control of the system, I have used an ESP32 microcontroller. The ESP32 is an extremely versatile and powerful platform that enables full control and monitoring of my smart flowerpot system.

**Keywords:** user interface, smart system, microprocessor

---

## Tartalomjegyzék

<b>1. Bevezető</b>	<b>12</b>
<b>2. Elméleti megalapozás és szakirodalmi tanulmány</b>	<b>13</b>
2.1. HTTP protokoll	14
2.2. JSON objektum	14
2.3. OneWire soros protokoll	14
2.4. Szoftver tervezés	15
2.5. ESP32 mikrovezérlő	15
2.6. Kapacitív nedvességmérő szenzor	17
2.7. DS18B20 hőmérséklet mérő szenzor	18
2.8. YF-S401 típusú hozammérő	19
2.9. Relé modul	19
2.10. Szivattyú	20
2.11. A növény megvilágításához használt LED csíkok	21
2.12. Nyák	22
2.13. Ismert hasonló alkalmazások	23
2.14. PHP	23
2.15. XAMPP	24
2.16. Laravel keretrendszer	25
2.17. MySQL	26
2.18. Arduino környezet	27
2.19. phpMyAdmin	28
<b>3. A rendszer specifikációi és architektúrája</b>	<b>28</b>
3.1. Rendszerkövetelmények	29
3.2. Felhasználói követelmények	30
3.2.1. Funkcionális követelmények	31
3.2.2. Nem funkcionális követelmények	32
<b>4. Részletes tervezés</b>	<b>32</b>
4.1. Hardver rész	33
4.2. Szoftver komponensek megvalósítása	34
4.2.1. Adatküldési intervallum	34
4.2.2. Adatbázis	35
4.2.3. Felhasznált könyvtárak	35
4.3. Biztonság	36
4.4. Felhasználói felület	38
4.5. Tervezési fázisok	49
4.6. Verziókövetés	53
<b>5. Üzembe helyezés és kísérleti eredmények - HIANYOS</b>	<b>54</b>
<b>5.1. Üzembe helyezés</b>	<b>54</b>
5.2. Felmerült problémák és megoldásaik	55
5.3. Kísérleti eredmények, mérések -FOLYAMATBAN	56
<b>6. A rendszer felhasználása</b>	<b>66</b>



---

<b>7. Következtetések</b>	<b>66</b>
7.1. Megvalósítások	66
7.2. Továbbfejlesztési lehetőségek	67
7.3. Költségvetés	68
<b>8. Irodalomjegyzék</b>	<b>69</b>
<b>9. Függelék</b>	<b>69</b>

---

## Ábrák jegyzéke

1. ábra Szoftver tervezési lépések	15
2. ábra ESP32 mikrovezérlő pinjei	16
3. ábra Kapacitív nedvességmérő szenzor	17
4. ábra DS18B20 hőmérséklet mérő szenzor	18
5. ábra YF- S401 típusú hozammérő	19
6. ábra Négy csatornás relé modul	20
7. ábra Pompa	22
8. ábra LED- ek	22
9. ábra 5x7- es nyák	23
10. ábra XAMPP felület	24
11. ábra Laravel fájl struktúra	26
12. ábra MySQL adatbázis	27
13. ábra phpMyAdmin felület	28
15. ábra Use case diagram	31
17. ábra POST kérés	34
18. ábra Adatbázis táblák	35
19. LED-ek ki-bekapcsolási logikája	38
20. Pompa ki-bekapcsolási logikája	38
21. ábra Home oldal	39
22. ábra Login oldal	40
23. ábra Jelszó visszaállítás	40
24. ábra Regisztrációs oldal	41
25. ábra Insert oldal	42
26. ábra Records oldal	42
27. Charts oldal - diagram 1	44
28. Charts oldal - diagram 2	44
29. ábra Sensor values oldal	45
30. ábra Export records oldal	46
31. ábra Excel export	46
32. ábra Fejléc	47
33. ábra Lábléc	48
34. ábra Fizikai design első prototípus	50
35. ábra A virágtartó hátulról - víztartály tervezése	50
36. ábra Végleges design	51
37. ábra Fizikai tartó	51
39. ábra A rendszer működés közben	53
40. ábra Hőmérséklet diagram	55
41. ábra Nedvesség diagram	56
42. ábra Tesztek - fájlrendszer	58
43. ábra Teszt 1	59
44. ábra Teszt 2	59

---

45. ábra Teszt 3	60
46. ábra Seeder teszt	60
47. ábra Email teszt	61
48. ábra Login teszt	61
49. ábra Dupla felhasználó teszt	62
50. ábra Egyedi e-mail teszt	63
51. ábra Törlés teszt	63
52. ábra Teszt - Felhasználó mentése	64
53. ábra Tesztek állapota	64
54. ábra JSON objektumok	69

### **Táblázatok jegyzéke**

1. táblázat A relé és az ESP32 csatlakoztatása	33
2. táblázat A nedvesség szenzor és az ESP32 csatlakoztatása	34
3. táblázat A hőmérséklet szenzor és az ESP32 csatlakoztatása	34
4. táblázat A hozammérő és az ESP32 csatlakoztatása	34
5. táblázat Költségvetés	67

---

## 1. Bevezető

Az okos virágtartó rendszerem pontosan behatárolt téma keretében terveztem és fejlesztettem, ahol egyértelmű célokat állítottam magam elé. Az volt a célom, hogy könnyebbé tegyem a növényeim gondozását és optimalizáljam az öntözésüket, miközben minimalizálom a manuális beavatkozás szükségességét.

Alaposan utánajártam minden egyes szenzornak és komponensnek, hogy biztosítsam a pontos és megbízható működést. Az ESP32 mikrovezérlő használata lehetővé teszi a rendszer teljes körű vezérlését és felügyeletét, így biztosítva a hatékony és precíz működést.

A rendszerben számos funkcionalitás található, amelyek segítenek az optimális növénygondozásban. Az érzékelők és vezérlők folyamatosan figyelik a növények környezeti paramétereit, mint például a hőmérséklet, páratartalom és fényerősség, valamint az öntözési szükségleteket. Ez lehetővé teszi, hogy a rendszer automatikusan szabályozza az öntözést, biztosítva, hogy a növények mindig megkapják a megfelelő mennyiségű vizet a megfelelő időben.

A felhasználói felület segítségével könnyedén beállíthatjuk az öntözési ütemtervet, nyomon követhetjük a növények állapotát és a környezeti paramétereket. Az ESP32 mikrovezérlő lehetővé teszi a távoli elérést, így akár otthon vagyunk, akár távol vagyunk, mindig kapcsolatban maradhatunk a virágainkkal és szükség esetén beavatkozhatunk.

A projekt célja az idő megtakarítása és a felelősség csökkentése a felhasználó számára. A rendszer egy modern és innovatív megoldás, ami lehetővé teszi számunkra, hogy gondoskodjunk növényeinkről a legjobb módon, miközben minimalizáljuk a gondozási feladatokat és maximalizáljuk a virágok egészségét és szépségét.

Az okos virágtartó rendszerem alaposan körbejárt és kutatott tervezési folyamat eredménye, amelynek célja a felhasználók életének megkönnyítése és a növénygondozás hatékonyságának növelése.

---

## 2. Elméleti megalapozás és szakirodalmi tanulmány

Az elméleti megalapozás és szakirodalmi tanulmány rendkívül fontos volt a rendszerfejlesztés szempontjából. Alapos kutatást végeztem a virágok gondozásával kapcsolatban, hogy megismerjem az öntözési igényeket és az optimális környezeti feltételeket. Továbbá információkat gyűjtöttem más rendszerekben alkalmazott eszközökről és technológiákról. Az általam végzett szakirodalom áttekintése segített megismerni a már létező kutatásokat és fejlesztéseket ezen a területen.

Az elméleti megalapozás és szakirodalmi tanulmány alapján megalapozott döntéseket hoztam a rendszer tervezése és megvalósítása során. A gyűjtött információk és az elméleti alapok felhasználása segített abban, hogy innovatív és hatékony megoldásokat találjak a virágtartó rendszerem számára. Az általam szerzett ismeretek segítségével biztosítottam, hogy a rendszer megbízható legyen, és kielégítse a felhasználói igényeket.

Alaposan utánanéztem minden egyes szenzornak és komponensnek, amelyeket a rendszeremhez használtam. Az szenzorokat és komponenseket gondosan válogattam ki az optimális működés és a felhasználói igények kielégítése érdekében.

A szenzorok esetében kutatást végeztem annak érdekében, hogy megismerjem a működési elveiket, a mért paraméterek pontosságát és a használatukat korábbi projektekben. Részletesen tanulmányoztam az adott szenzorok specifikációit, dokumentációit és a használati útmutatókat, hogy teljes mértékben megértsem a működésüket és a rendszerbe történő integrációjukat.

A komponensek kiválasztásakor alaposan áttekintettem a rendelkezésre álló lehetőségeket és összehasonlítottam azokat. Számos technikai adatlapot, felhasználói véleményt és tesztet tanulmányoztam, hogy megbizonyosodjak a komponensek minőségéről és megbízhatóságáról. Figyelembe vettem az egyes komponensek teljesítményét, interfészét, energiafogyasztását és kompatibilitását az ESP32 mikrovezérlővel.

---

Az alapos utánajárás és a részletes elemzés eredményeként olyan szenzorokat és komponenseket választottam ki, amelyek megfelelnek a rendszerem funkcionalitásának és teljesítményigényeinek. Ez biztosítja, hogy a rendszerem megbízhatóan működjön és a felhasználók elégedettek legyenek a funkcionalitással és a teljesítménnyel.

A dokumentálódás során számos ismert és kevésbé ismert fogalommal találkoztam:

## **2.1. HTTP protokoll**

A HTTP protokoll egy üzenetváltást, adatcserét valósít meg a kliens és a szerver között. A kliens általában egy böngésző szokott lenni.

Működési elve: a kliens küld egy kérést, amit HTTP request-nek nevezünk a szerver felé, amely válaszol (HTTP response). A kérésben meghatározottak lehetnek például az átvitel típusa, az adott weboldal címe (URL), kérés metódus (GET, POST, stb.) vagy egyéb adatok.

## **2.2. JSON objektum**

A JSON objektum egy könnyen olvasható és írható adatformátum, kulcs-érték párokból épül fel. Széles körben használják a webes kommunikációban, különösen a RESTful API-knál. A könnyű olvashatóság, az egyszerű struktúra és a platformfüggetlen jelleg miatt a JSON népszerű adatsere formátum a fejlesztők és az alkalmazások között.

## **2.3. OneWire soros protokoll**

Egyszerű és hatékony adatátviteli protokoll, amelyet a szenzorok és egyéb perifériákkal való kommunikáció megvalósításához használnak.

Egyetlen vezetéken történik az adatátvitel és az energiaellátás. Az adatok bitenként továbbítódnak és minden bit időzítéssel van ellátva.

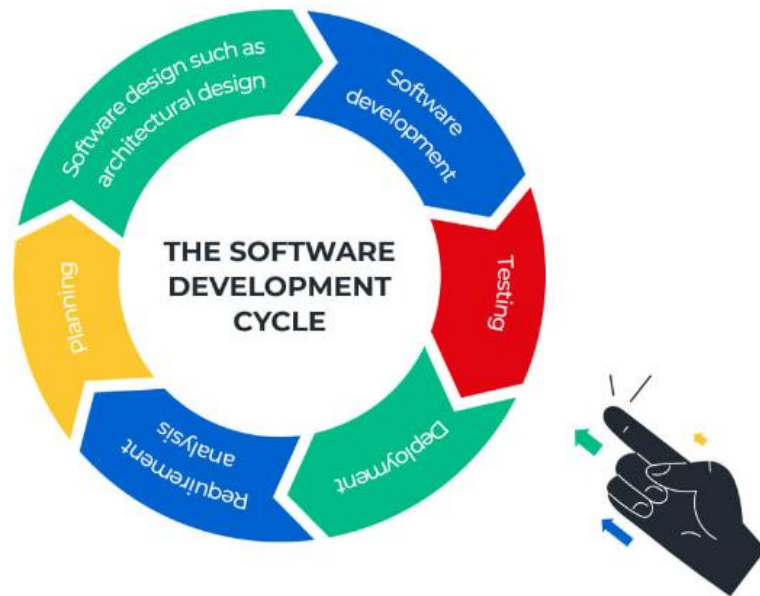
## **2.4. Szoftver tervezés**

Ahhoz, hogy egy rendszer hatékonyan és megbízhatóan működjön, illetve megfeleljen a felhasználói és rendszerkövetelményeknek, elengedhetetlen, hogy a rendszert alaposan megtervezzük és strukturáljuk.

Az szoftvertervezés folyamata az alábbi lépéseket foglalja magába:

1. Követelményelemzés

- 
2. Architektúra kialakítása
  3. Adatmodell létrehozása
  4. Funkcionális tervezés
  5. Tesztelés



*1.ábra Szoftver tervezési lépések*

Kép forrása: [link](#).

## 2.5. ESP32 mikrovezérlő

Az ESP32 mikrovezérlőt az Espressif Systems kínai cég hozta létre és fejlesztette ki. Elődje az ESP8266 volt, ami 2014- ben került piacra.

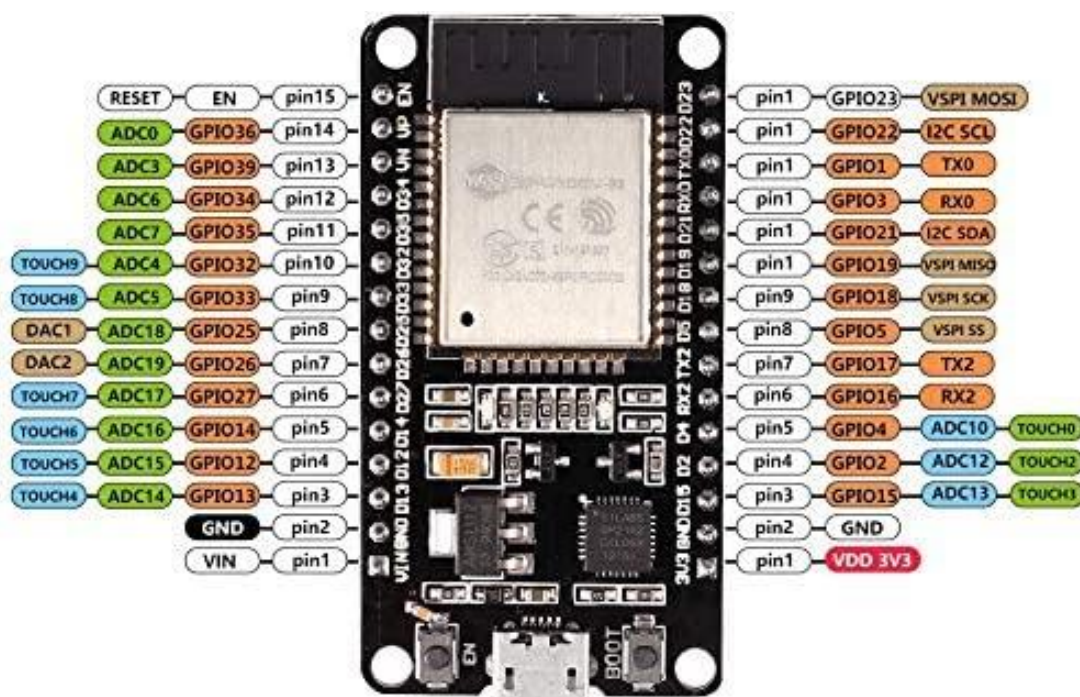
Nagyon népszerű, rugalmas és teljesítőképes mikrovezérlő fejlesztő lap. Számos IoT (Internet of Things) alkalmazásokban, okos otthon rendszerekben, ipari automatizációban stb használatos.

Néhány fontosabb tulajdonság, amivel rendelkezik:

1. **Kétmagos processzor:** két Tensilica Xtensa LX6 maggal rendelkezik, amelyek 32 bites RISC architektúrán alapulnak. A párhuzamos végrehajtás miatt sokkal nagyobb teljesítményt tud biztosítani.
2. **WiFi és Bluetooth modul:** beépített WiFi és Bluetooth modullal rendelkezik, amelynek köszönhetően vezeték nélküli kommunikációt tud biztosítani más eszközökkel.

3. **Alacsony energiafogyasztás:** az energia optimalizálási funkcióknak köszönhetően alacsony energiafogyasztással rendelkezik, így az energiahatékony alkalmazásoknál igen hasznos tud lenni.
4. **Kiterjesztett I/O interfészek:** számos I/O interfésszel rendelkezik, lehetővé téve a szenzorok, perifériák és kijelzők csatlakoztatását. Pl: analóg és digitális bemenetek, PWM kimenetek, SPI, I2C illetve UART kommunikációs interfészek.

Az alábbi ábrán láthatóak az ESP32 mikrovezérlő pinjei:



2. ábra ESP32 mikrovezérlő pinjei

Kép forrása: [link](#)

Bár az ESP8266 alacsonyabb energiafogyasztással rendelkezik, mint az ESP32 a választásom mégis az ESP32-re esett. A rendszerem szempontjából nagyon fontos volt, hogy egy olyan mikrovezérlőt használjak, amely gazdag I/O interfésszel rendelkezik. Az ESP32 kiterjesztett I/O interfész választékának köszönhetően számos külső eszközt tudtam csatlakoztatni hozzá minden probléma nélkül. Ugyanakkor az ESP32-nek több memóriája van, így komplexebb alkalmazások is futtathatóak rajta.



---

## 2.6. Kapacitív nedvességmérő szenzor

A nedvességmérő szenzorokat széles körben alkalmazzák. Két fő csoportra oszthatjuk: kapacitív és rezisztív nedvességmérő szenzorok. Mindkét típusnak megvannak a maga előnyei és hátrányai. A dolgozatomhoz én a kapacitív változatot választottam.



3. ábra Kapacitív nedvességmérő szenzor

Kép forrása: [link](#)

Néhány érv, ami miatt erre a típusra esett a választásom:

1. **Korrózióállóság:** a kapacitív szenzorok sokkal ellenállóbb a korrózióval és kopással szemben, így sokkal hosszabb "élettartammal" rendelkezhetnek.
2. **Érzékenység:** jobban reagálnak a kisebb nedvesség változásokra, pontosabb eredményt biztosítva. A rezisztív szenzorokkal ellentétben sokkal érzékenyebbek, ami igen fontos szempont, ha a növénynek a megfelelő vízmennyiséget szeretnénk biztosítani.
3. **Nincs elektromos vezetés:** ez azt jelenti, hogy a kapacitív szenzorok nem igényelnek közvetlen elektromos vezetést, elkerülve az elektromos interferenciát illetve a korábban említett korróziós problémát.
4. **Nincs kontakt a talajjal:** a nedvességet a szenzor és a közeli anyag közötti kapacitás változása alapján mérik.

---

## 2.7. DS18B20 hőmérséklet mérő szenzor

A DS18B20 egy digitális hőmérséklet szenzor, amelyet széles körben alkalmaznak különböző területeken: otomatisztikus rendszerekben, hőmérsékletérzékelésben, meteorológiai állomásokban, klímaberendezésekben stb. Minden olyan területen használatos, ahol fontos szempont a pontosság illetve digitális mérésre van szükség.

Széles mérési tartománnyal rendelkezik, akár  $-55\text{ }^{\circ}\text{C}$  és  $+125\text{ }^{\circ}\text{C}$  közötti hőmérsékletet is képes mérni.



4. ábra DS18B20 hőmérséklet mérő szenzor

Kép forrása: [link](#)

A DS18B20 fontosabb tulajdonságai:

1. **Pontosság:** nagyon pontosan méréseket végez.  $\pm 0.5\text{ }^{\circ}\text{C}$  pontossággal mér, viszont beállítással még nagyobbra növelhetjük a pontosságot.
2. **Digitális kommunikáció:** One Wire protokollt használ, amely egy digitális kommunikációs protokoll. Csak egyetlen egy vezeték szükséges a kommunikáció megvalósításához és az adatküldéshez, illetve természetesen tápfeszültségre is szükség van.
3. **Egyedi 64 bites azonosítóval** rendelkezik. Ennek köszönhetően több szenzort is használhatunk ugyanazon a One Wire buszon.

Szenzor adatlap: [link \[14\]](#)

## 2.8. YF-S401 típusú hozammérő

A hozammérőt általában a folyadékok áramlásának mérésére használják. A hozam azt az anyagmennyiséget jelenti, amely időegységenként áramlik. PVC testből, vízrotorból és Hall-

---

effektus szenzorból áll. Amikor a víz átfolyik a rotoron, mozgásba hozza azt és sebessége különböző áramlási sebességgel változik.

Méret, forma és működési elv függvényében számos változata létezik illetve széles körben alkalmazzák őket ipari alkalmazásokban.

Én a dolgozatomhoz az YF-S401 típusú hozammérőt használtam. Azért esett erre a választásom, mivel kis mérete van és alkalmas 0,3 és 6 liter/perc közötti áramlási sebességekhez. Az is fontos szempont volt, hogy ugyanazzal az átmérővel rendelkezik, mint a pompa, így a csövet, amin keresztül áramlik a víz, nem kellett átalakítani, tökéletesen illett rá.



5. ábra YF- S401 típusú hozammérő

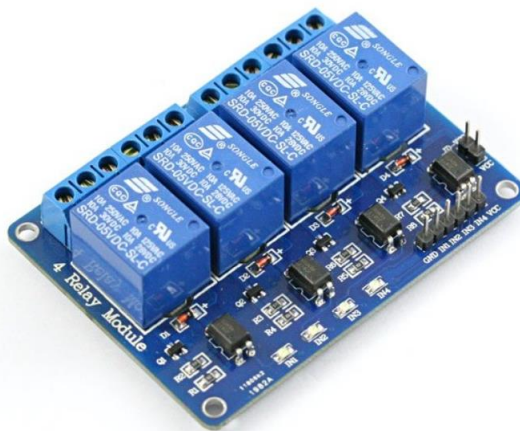
Kép forrása: [link](#)

## 2.9. Relé modul

A relé modul segítségével távolról irányíthatjuk a mikrovezérlő által vezérelt elektromos eszközöket.

A modul négy csatornája különállóan vezérelhető. A projekthez jelenleg két csatornát használok fel belőle, de a későbbiekben nem kizárt, hogy szükség lesz a másik két csatornára is, ezért döntöttem amellett, hogy ezt a típusú relét fogom használni.

Jóval nagyobb a mérete, mint például a két csatornás relének, de a dolgozatom esetében ez nem okozott gondot, hiszen a nyák is 5x7 cm-es mérettel rendelkezik és a tartó, amibe került pont akkora, hogy tökéletesen ráillett a relé és csavarokkal hozzá tudtam erősíteni.



6. ábra Négy csatornás relé modul

Kép forrása: [link](#)

Néhány fontos információ a 4 csatornás relé modulokról:

1. **Relék:** a relék elektromos kapcsolóként funkcionálnak és lehetővé teszik a nagy áramú vagy magas feszültségű áramkörök vezérlését kis vezérlő árammal.
2. **Csatornák:** a csatornák különálló vezérlésének köszönhetően, különböző eszközök vezérlését teszi lehetővé egyetlen egy modul használatával.
3. **Vezérlési mód:** A PC- től vagy mikrovezérlőtől kapott vezérlési jel alapján be- vagy kikapcsolja a relét. A jelet kaphatja digitális jel vagy impulzus formájában is.
4. **Bemeneti feszültség:** külön tápellátással rendelkezik, így megfelelő feszültséget tud biztosítani.

Nagy népszerűségnek örvend, számos területen használják őket, főleg olyan rendszereknél, ahol különböző eszközöket kell vezérelni.

## 2.10. Szivattyú

Ez a rendkívül kompakt méretű, víz alá merülő szivattyú kiválóan alkalmas otthoni felhasználásra. A szivattyú könnyedén integrálható fejlesztőlapokkal, modulokkal és egyéb szenzorokkal, lehetővé téve különféle feladatok megoldását.

Néhány fontosabb technikai jellemző:

- 
- A szivattyú működéséhez DC 3-5V feszültségre van szükség.
  - Az áramerősség tartománya 100-200mA között helyezkedik el.
  - Az áramlási sebesség 1,2-1,6 liter/perc.
  - A szivattyú rendkívül könnyű, mindössze 28 gramm.
  - A műanyagból készült burkolat biztosítja a tartósságot és a könnyű kezelhetőséget.
  - A szivattyú cső kimenetének külső átmérője 7,5 mm, míg a belső átmérője 4,7 mm.
  - A szivattyú átmérője körülbelül 24 mm, hossza pedig mintegy 45 mm.
  - Az összmagasság körülbelül 33 mm.

Fontos megjegyezni, hogy a szivattyú folyamatos üzemideje ne haladja meg az ajánlott 500 órát.

Forrás: [link](#)

Ez a szivattyú kiváló választás, ha egy hatékony és megbízható vízmozgató eszközt keresünk otthoni felhasználásra. A kompakt mérete, alacsony energiafogyasztása és könnyű kezelhetősége nagyszerű tulajdonságok, amelyek hozzájárulnak a rendszerünk hatékony működéséhez és kényelmes használatához.



7. ábra Pompa

Kép forrása: [link](#)

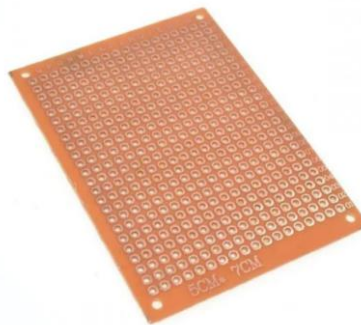
## 2.11. A növény megvilágításához használt LED csíkok



8. ábra LED- ek

## 2.12. Nyák

A nyák (vagy nyomtatott áramkör, PCB - Printed Circuit Board) egy alaplap, amelyen elektronikus komponensek és vezetékek vannak elhelyezve. A nyomtatott áramkörök olyan eszközök, amelyek lehetővé teszik a különböző elektronikus alkatrészek (mint például ellenállások, kondenzátorok, tranzisztorok stb.) összekapcsolását és szerelését.



9. ábra 5x7- es nyák

## 2.13. Ismert hasonló alkalmazások

Az okos virágtartók széles körben elérhetők a piacon, és számos vállalat kínálja ezeket a termékeket. Az alkalmazások és platformok, amelyeket ezek az eszközök támogatnak, lehetővé teszik a felhasználók számára, hogy több virágtartót is kezeljenek és integrálják a növény gondozást a modern életstílusba.

---

Rengeteg hasonló projekt van jelen az interneten, amiből tanulságokat lehet levonni. A rendszerem a megvalósított projektek nagy százalékához képest komplexebb, hiszen sokkal több funkciót kínál a felhasználók számára. Egyedi felhasználói felülettel rendelkezik, amely maximális kényelmet nyújt a felhasználóknak.

A szenzorok kiválasztásánál fontos szempont volt, hogy minél pontosabban mérjenek, növelve a rendszer megbízhatóságát. Számos projektnél DHT11 hőmérséklet szenzort használnak, viszont a DS18B20 szenzor sokkal pontosabban mér.

## 2.14. PHP

PHP (Hypertext Preprocessor) egy szerveroldali, szkriptelésre használt programozási nyelv, amelyet kifejezetten webfejlesztésre terveztek.

Néhány jellemző tulajdonsága és információ a PHP-ről:

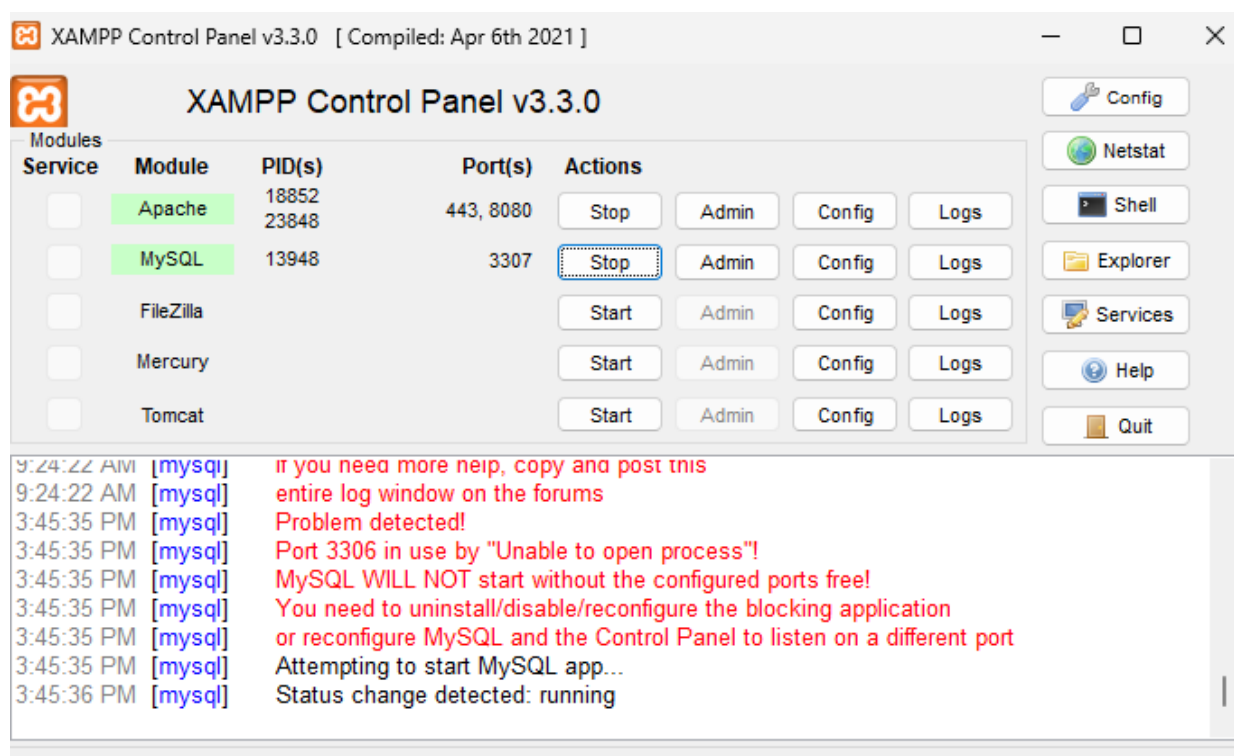
- 1. Szerveroldali programozás :** PHP egy szerveroldali nyelv, ami azt jelenti, hogy a szerveren fut, és a generált HTML-t küldi vissza a böngészőnek. Ez lehetővé teszi a dinamikus weboldalak és webalkalmazások fejlesztését, ahol a szerver oldalon feldolgozás, adatbázis-kapcsolatok és más logikai műveletek történnek.
- 2. Adatbázis kezelés :** a PHP erőteljes adatbázis-kezelési funkciókkal rendelkezik. A nyelv különféle adatbáziskezelő rendszerekkel, mint például a MySQL, PostgreSQL, SQLite stb. történő kapcsolatot támogatja, és lehetővé teszi az adatok lekérdezését, beillesztését, frissítését és törlését az adatbázisból.
- 3. Platformfüggetlenség :** a PHP platformfüggetlen nyelv, ami azt jelenti, hogy futtatható különböző operációs rendszereken, mint például a Windows, Linux, macOS stb. Ez nagy rugalmasságot és hordozhatóságot jelent a PHP-alapú projektek számára.

## 2.15. XAMPP

Az XAMPP egy népszerű és széles körben használt szoftvercsomag, amelyet az adatbázisok és webalkalmazások fejlesztéséhez és teszteléséhez használtam. Az XAMPP az Apache webkiszolgáló, a MySQL adatbázis kezelő, a PHP programozási nyelv és más komponensek kombinációját tartalmazza.

Az XAMPP rendkívül könnyen telepíthető és konfigurálható, így gyorsan beállítottam az adatbázis környezetet a projekthez.

A MySQL adatbázis kezelő nagyszerű eszköz az adatok tárolásához és lekérdezéséhez. Az XAMPP segítségével könnyedén hozhatunk létre adatbázisokat, táblákat és végrehajthatók SQL lekérdezések is. Az adatbázis kezelése és manipulálása egyszerű és hatékony módon történik. Összességében az XAMPP megbízható és kényelmes megoldást kínál az adatbázisok webalkalmazások fejlesztéséhez.



10. ábra XAMPP felület

## 2.16. Laravel keretrendszer

A Laravel egy nyílt forráskódú, PHP alapú keretrendszer. Számos funkcióval és egyszerűsítéssel rendelkezik, megkönnyítve a fejlesztők dolgát. Hatékonyan és gyorsan lehet web alkalmazásokat fejleszteni a használatával.

Néhány fontosabb információ, ami a Laravel mellett szól:

1. **Könnyedség:** parancssorból könnyen lehet létrehozni új projekteket, konfiguráció kezelésre is alkalmas, adatbázis migrálást lehet végezni illetve sok más egyéb hasznos funkciót is lehet létrehozni vele.



---

Példa néhány parancsra:

**composer create-project --prefer-dist laravel/laravel new\_project**

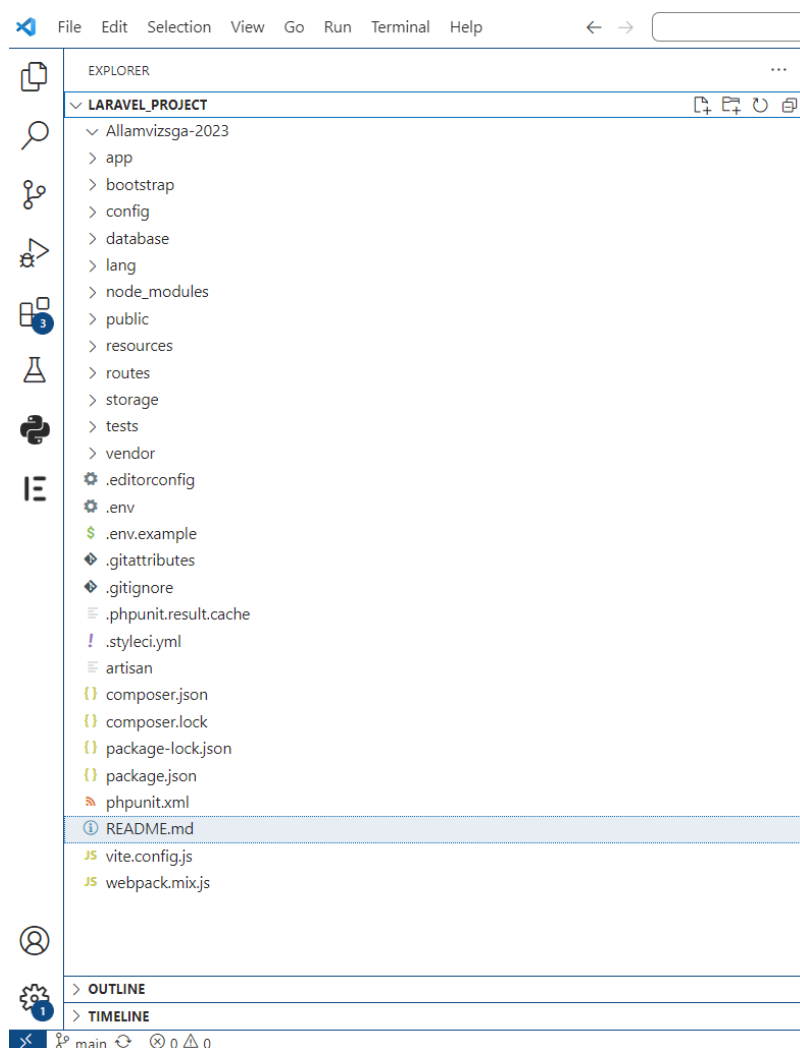
**php artisan make:migration create\_new\_table**

**php artisan migrate**

**php artisan tinker**

Lásd : <https://laravel.com/docs/10.x/artisan>

2. **MVC architektúra:** a kód szétválasztása érdekében a Laravel támogatja a Model- View- Controller mintát. Ennek köszönhetően a kód sokkal karbanthatóbbá és könnyebben bővíthetővé válik.
3. **Object- Relational Mapping:** PHP objektumok segítségével lehetővé teszi az adatbázis műveletek és az adatok egyszerű és intuitív kezelését.
4. **Egyszerű útvonalak:** az útvonalakat vagy ún. rout- okat könnyen lehet definiálni.



## 11. ábra Laravel fájl struktúra

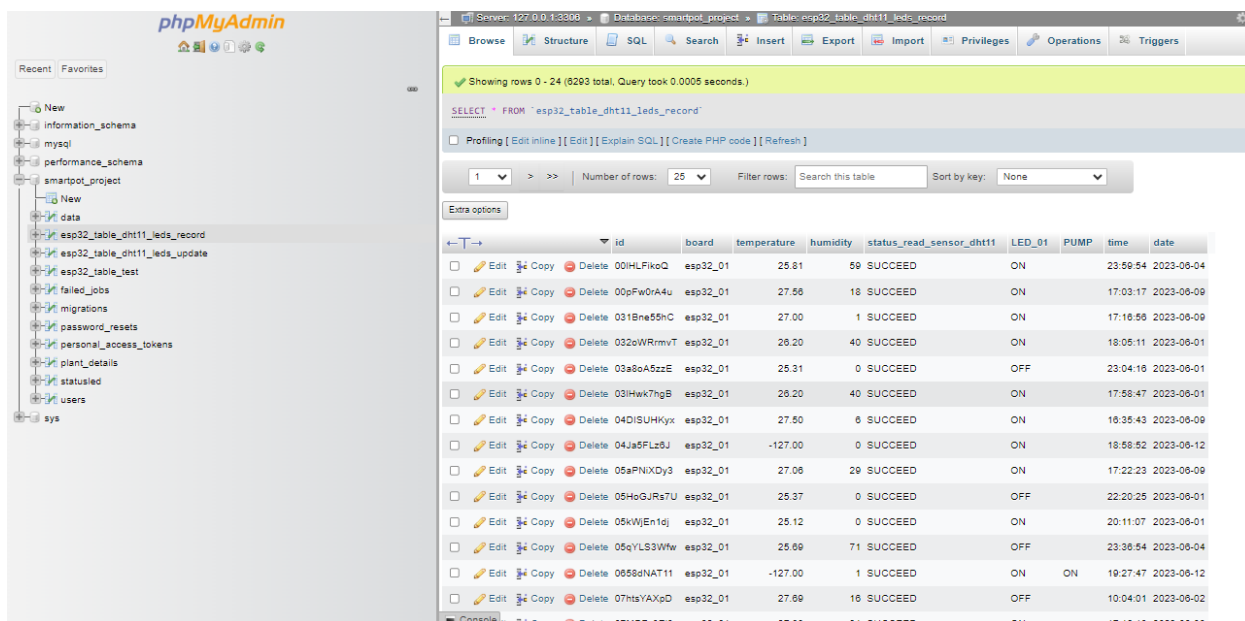
### 2.17. MySQL

A MySQL egy népszerű és hatékony relációs adatbázis-kezelő rendszer, amelyet széles körben alkalmaznak a webalkalmazások és szoftverrendszerek fejlesztésében. A MySQL erőteljes, skálázható és megbízható megoldást nyújt az adatok tárolására, lekérdezésére és kezelésére.

A MySQL kiváló teljesítményt és gyors adatfeldolgozást biztosít. Az adatbázis számos optimalizációs technikát alkalmaz, például indexelést és cache-elést, hogy gyors hozzáférést és hatékony adatmanipulációt tegyen lehetővé. Emellett a MySQL támogatja a többfelhasználós működést és a konkurens hozzáférést, így több alkalmazás vagy felhasználó egyidejűleg dolgozhat az adatbázison.

A MySQL adatbázis-kezelő rendszert választva biztosítottam a rendszerem számára egy megbízható és hatékony adatbázis-kezelést. Ez lehetővé tette, hogy az adatokat strukturáltan tároljam, hatékonyan manipuláljam és könnyen integráljam más rendszerekkel és alkalmazásokkal.

A MySQL adatbázis-kezelő rendszerének skálázhatósága és megbízhatósága lehetővé tette, hogy a rendszeremet a jövőben bővíthessem és kezelhessem egyre növekvő adatmennyiséggel. Emellett a MySQL támogatja a többfelhasználós működést, így egyszerre több felhasználó tudja használni és módosítani az adatokat.



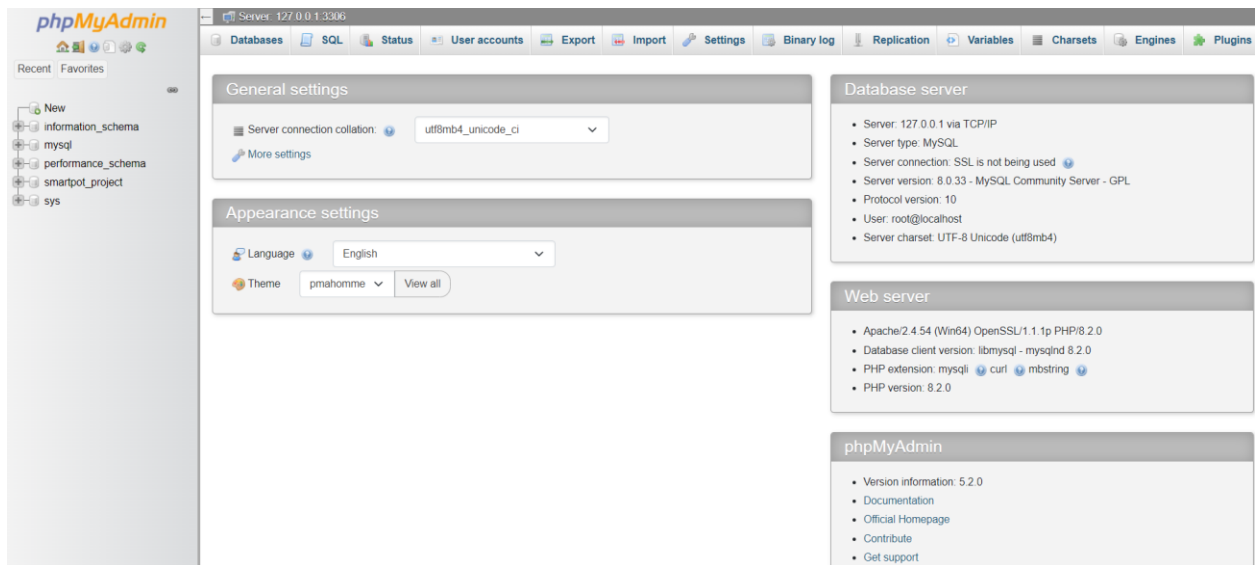
	id	board	temperature	humidity	status_read_sensor_dht11	LED_01	PUMP	time	date
<input type="checkbox"/>	001HLFIkoQ	esp32_01	25.81	59	SUCCESS	ON		23:59:54	2023-06-04
<input type="checkbox"/>	00pFW0rA4u	esp32_01	27.56	18	SUCCESS	ON		17:03:17	2023-06-09
<input type="checkbox"/>	031Bne55nC	esp32_01	27.00	1	SUCCESS	ON		17:16:56	2023-06-09
<input type="checkbox"/>	032oWRmvT	esp32_01	26.20	40	SUCCESS	ON		18:05:11	2023-06-01
<input type="checkbox"/>	03a8oA5zeE	esp32_01	25.31	0	SUCCESS	OFF		23:04:16	2023-06-01
<input type="checkbox"/>	031Hwk7ngB	esp32_01	26.20	40	SUCCESS	ON		17:58:47	2023-06-01
<input type="checkbox"/>	04D1SUHKyx	esp32_01	27.50	6	SUCCESS	ON		18:35:43	2023-06-09
<input type="checkbox"/>	04Ja5FLz2J	esp32_01	-127.00	0	SUCCESS	ON		18:58:52	2023-06-12
<input type="checkbox"/>	05aPNIXDy3	esp32_01	27.08	29	SUCCESS	ON		17:22:23	2023-06-09
<input type="checkbox"/>	05HoGURs7U	esp32_01	25.37	0	SUCCESS	OFF		22:20:25	2023-06-01
<input type="checkbox"/>	05kWiEn1dJ	esp32_01	25.12	0	SUCCESS	ON		20:11:07	2023-06-01
<input type="checkbox"/>	05qYLS3Ww	esp32_01	25.69	71	SUCCESS	OFF		23:38:54	2023-06-04
<input type="checkbox"/>	0658dNAT11	esp32_01	-127.00	1	SUCCESS	ON	ON	19:27:47	2023-06-12
<input type="checkbox"/>	07htsYAXpD	esp32_01	27.69	16	SUCCESS	OFF		10:04:01	2023-06-02
<input type="checkbox"/>	07MR5Gp3R	esp32_01	27.00	31	SUCCESS	ON		17:16:10	2023-06-09

## 2.18. Arduino környezet

Az Arduino környezet egy nyílt forráskódú fejlesztői platform és programozási környezet, amelyet kifejezetten mikrovezérlő alapú projektek fejlesztésére terveztek. Az Arduino platform népszerűvé vált a hobbisták, diákok és profi fejlesztők körében is, mert könnyen használható és rugalmasan alkalmazható.

Az Arduino környezetben a fejlesztők különböző szenzorokat, aktuátorokat és más hardveres modulokat tudnak kezelni és vezérelni. Az Arduino környezet támogatja a széles körben elérhető könyvtárakat és modulokat, amelyek megkönnyítik a különböző funkciók implementálását.

## 2.19. phpMyAdmin



13. ábra phpMyAdmin felület

## 3. A rendszer specifikációi és architektúrája

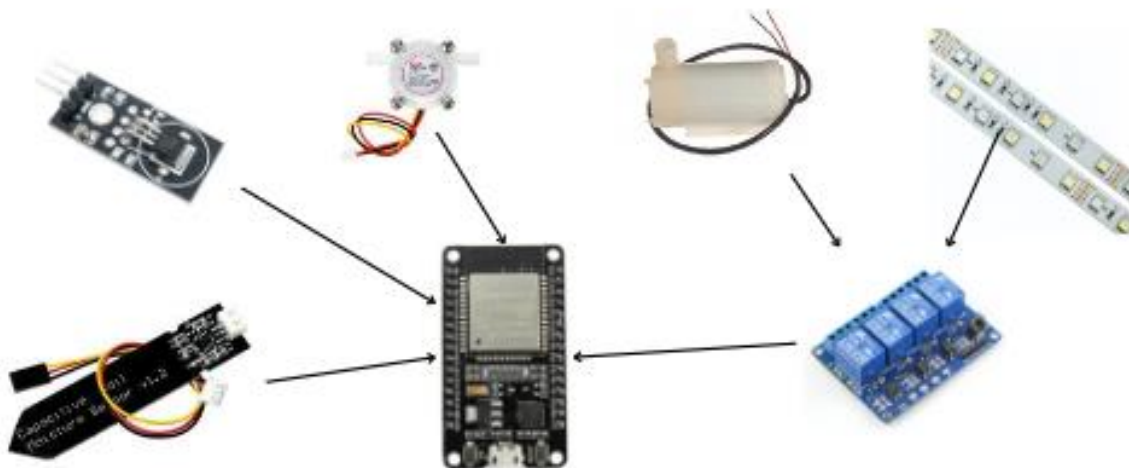
A rendszer architektúrája egy jól strukturált rendszert foglal magába, biztosítva a felhasználói igényeinek kielégítését és a hatékony működést. A rendszer alapját a szenzorok jelentik, amelyek szerepe, hogy folyamatosan érzékeljék a környezeti paramétereket. Ezek az adatok továbbításra kerülnek az ESP32 mikrovezérlő felé. A vezérlőegység felel a rendszer

---

vezérléséért, szabályozza az öntözést és világítást a felhasználói felületről kapott értékek és a mért paraméterek alapján.

A felhasználói felület lehetővé teszi a felhasználók számára, hogy interaktív módon kommunikáljanak a rendszerrel. A felhasználók növény profilokat hozhatnak létre, beállíthatják az öntözési ütemtervet, beavatkozhatnak a rendszer működésébe, nyomon követhetik a növények állapotát és nem utolsósorban távolról is kapcsolatban maradhatnak a rendszerrel. A mikrovezérlő és a felhasználói felület közötti adatcseréért a TCP/IP kommunikációs protokoll felelős.

Az okos virágtartó rendszer architektúráját a szenzorok, vezérlőegység, felhasználói felület, kommunikációs protokollok és az adatbázis együttes működése és kommunikációja alkotja. Célja, hogy lehetővé tegye a hatékony növénygondozást, a felhasználói igények kielégítését és a rendszer zavartalan működését.



14. ábra A megvalósított rendszer architektúrája

### 3.1. Rendszerkövetelmények

---

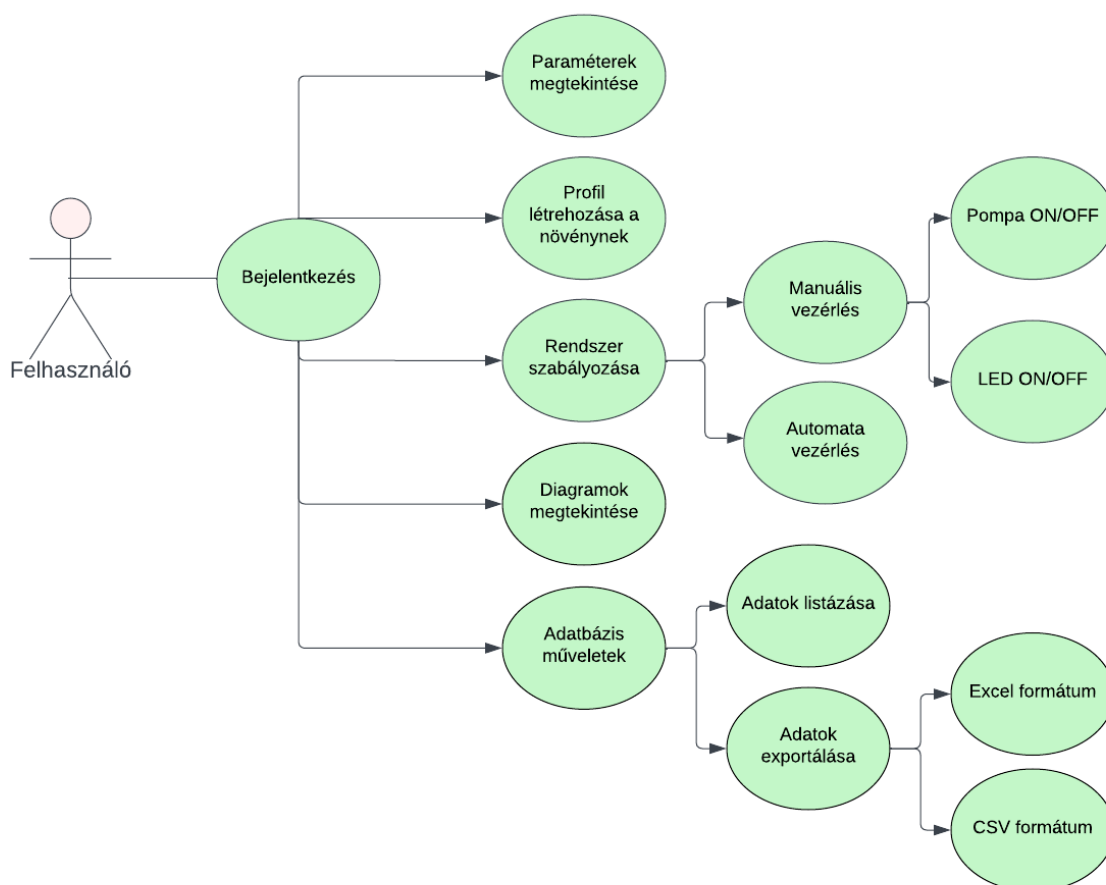
A következőkben röviden bemutatom a rendszerkövetelményeket, amelyeknek egy ilyen rendszernek meg kell felelnie. Olyan szempontokat foglal magába, amelyek biztosítják a rendszer hatékonyságát, megbízhatóságát és felhasználóbarát működését.

1. Ahhoz, hogy a rendszer megfelelően működjön, szükség van egy megbízható mikrovezérlőre, mint például az ESP32, ami képes megfelelően kezelni és irányítani az összes szükséges funkciót illetve kommunikálni az érzékelőkkel és egyéb külső csatlakozókkal. Nagyon fontos, hogy a mikrovezérlő elegendő erőforrással és memóriával rendelkezzen a program futtatásához és az adatok kezeléséhez.
2. Szükség van különböző érzékelőkre. Ilyenek például a hőmérséklet- és nedvesség érzékelők, amelyek képesek pontosan mérni a növények környezeti paramétereit. Az érzékelők minél megbízhatóbbak és pontosabbak kell legyenek, hiszen ezek alapján hoz a rendszer döntéseket és befolyással vannak a növények állapotára.
3. Szükség van egy megbízható és hatékony öntözési rendszerre, ami egy pompát és csövet foglal magába, így a rendszer biztosítani tudja a megfelelő mennyiségű vizet a növényeknek.
4. A rendszernek rendelkeznie kell egy interaktív felhasználói felülettel, amely lehetővé teszi a felhasználók számára a növények állapotának és környezeti paramétereinek nyomon követését illetve egyéb funkciókat is el tudnak érni és kezelni tudnak a felületről. Ez lehet webes felület, mobilalkalmazás vagy más felhasználóbarát interfész. A projektem megvalósításakor én egy webes felületet készítettem, amelyet bármilyen eszközről elérhetünk WiFi- n keresztül a megfelelő elérési útvonal, azaz az URL segítségével.
5. Végül, a rendszernek biztosítani kell a távoli hozzáférést és a rendszer vezérlését. A felhasználók bárholonnan hozzáférhetnek a rendszerhez és ellenőrizhetik a növények állapotát, valamint beavatkozhatnak a működésbe, amennyiben szeretnének élni ezzel a lehetőséggel vagy megkívánja a helyzet. Ez a funkció lehetővé teszi a rugalmas kezelést és felügyeletet, még akkor is, ha a felhasználó nem tartózkodik a rendszer közelében.

Ezek a rendszerkövetelmények biztosítják az okos öntöző és világító rendszer hatékony és megbízható működését, illetve a felhasználói élményt is fokozza és biztosítja a növények megfelelő gondozását.

### 3.2. Felhasználói követelmények

A felhasználói követelmények azon elvárásoknak és igényeknek az összesége, amelyeket a felhasználók állítanak fel rendszerekkel kapcsolatban. Ezek az elvárások fontosak a felhasználók számára a rendszer használata során, hiszen hozzájárulnak a felhasználói élmény és elégedettség biztosításához.



15. ábra Use case diagram

---

### 3.2.1. Funkcionális követelmények

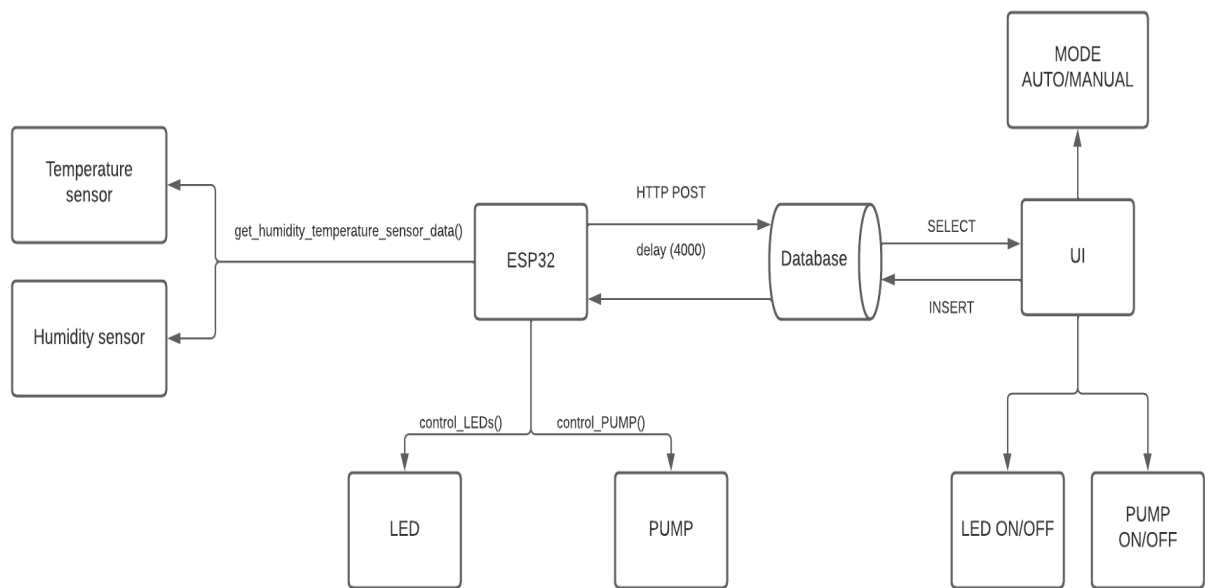
1. Beállítások testreszabása: A felhasználók kell tudják irányítani az öntözést és világítást illetve növény profilokat kell tudjanak létrehozni a növény egyedi igényeihez igazodva.
2. Könnyű kezelhetőség: A felhasználói felület minél egyszerűbb kell legyen, hogy a felhasználók könnyen navigálhassanak a rendszerben és könnyen tudjanak kezelni minden egyes funkciót.
3. Távoli elérés és vezérlés: A rendszernek lehetőséget kell biztosítania a felhasználók számára, hogy bármikor és bárhol hozzáférhessenek a rendszerhez, és ellenőrizhessék a növények állapotát és különböző műveleteket hajthassanak végre.

### 3.2.2. Nem funkcionális követelmények

1. Teljesítmény: A rendszernek hatékonyan és megbízhatóan kell működnie, biztosítva a megfelelő öntözési mennyiséget és időzítést.
2. Biztonság: A rendszernek megfelelő biztonsági intézkedésekkel kell rendelkeznie, például jelszóvédelemmel vagy titkosítással a távoli hozzáféréshez.
3. Megebízhatóság: A rendszernek stabilnak és megbízhatónak kell lennie, minimalizálva a hibákat és a meghibásodásokat.
4. Skálázhatóság: A rendszernek lehetőséget kell biztosítania a bővítésre és új funkciók hozzáadására a jövőben, ha szükséges.

## 4. Részletes tervezés

Az államvizsga témám 2 nagyobb részből tevődik össze: a szoftver illetve hardver részből. A virágtartó működése és vezérlése hardver oldalon történik, de elengedhetetlen kritérium volt, hogy a felhasználó számára egy szép és könnyen kezelhető UI álljon a rendelkezésére, amin keresztül nyomon tudja követni a virág állapotát a mért adatok segítségével. Ugyanakkor parancsokat is tud küldeni, befolyásolva a ledet kapcsolását és az öntözést.



16. ábra Szekvencia diagram

## 4.1. Hardver rész

Az alábbi táblázatban a relé modul és az ESP32 kapcsolása látható. Az IN1 signal a ledet képviseli, illetve az IN2 signal a pompát.

Az ESP32 A Vin lábán 5V- ot tud szolgáltatni, ezért a relé VCC pinjét a megfelelő működés érdekében a Vin - re csatlakoztattam.

RELÉ	ESP32
GND	GND
IN1	D13
IN2	D12
VCC	VIN

[1. táblázat](#) A relé és az ESP32 csatlakoztatása

NEDVESSÉG SZENZOR	ESP32
GND	GND



VCC	3V3
AOUT	D35

2. [táblázat](#) A nedvesség szenzor és az ESP32 csatlakoztatása

HŐMÉRSÉKLET SENZOR	ESP32
GND	GND
VCC	3V3
AOUT	D4

3. [táblázat](#) A hőmérséklet szenzor és az ESP32 csatlakoztatása

HOZAMMÉRŐ	ESP32
GND	GND
VCC	3V3
AOUT	D27

4. [táblázat](#) A hozammérő és az ESP32 csatlakoztatása

## 4.2. Szoftver komponensek megvalósítása

A mikrovezérlő felprogramozása Arduino környezeten keresztül történt.

A program, amely az ESP-t irányítja tartalmaz különböző könyvtárakat, amelyek segítségével a mikrovezérlő képes kommunikálni a WiFi-hálózattal, HTTP kéréseket indítani, adatokat érzékelni, és vezérelni a kimeneti eszközöket.

A kód megírásában a következő tutoriál volt a segítségemre, amit majd a saját rendszerem igényei szerint formáltam : [link](#) .

---

A `setup()` függvény inicializálja a soros kommunikációt, beállítja a pinek módját, csatlakozik a WiFi-hálózathoz, és inicializálja a hőmérséklet-érzékelőt.

Ezt követően, két vezérlő függvényt hívunk meg: `control_LEDs()` és `control_PUMP()`. Ezek a függvények a bejövő JSON adatok alapján vezérlik az LED-eket és a pumpát. Például ha az üzemmód "AUTO" és a hőmérséklet alacsonyabb, mint az alsó határ, vagy az üzemmód "MANUAL" és az "LED\_01" be van kapcsolva, akkor a LED-et bekapcsoljuk. Ugyanez vonatkozik a pumpára is a páratartalom alapján.

Ezt követően, összeállítunk egy adatokat tartalmazó POST kérést, amelyet elküldünk a szervernek.

```
http.begin("http://192.168.1.6:8000/esp_communication/update_recordtable");  
http.addHeader("Content-Type", "application/x-www-form-urlencoded");
```

*17. ábra POST kérés*

#### **4.2.1 Adatküldési intervallum**

Az adatküldési időköz a `loop()` függvényben található `delay(4000)`; sorral van beállítva. Az ESP32 időközönként, minden 4 másodpercben leolvassa a hőmérsékletet és nedvességet a szenzorokról, majd elküldi ezeket az adatokat a szerverre.

---

## 4.2.2 Adatbázis

<b>smartpot_project esp32_table_record</b> id : varchar(255) board : varchar(255) temperature : float(10,2) humidity : int status_read : varchar(255) LED_01 : varchar(255) PUMP : varchar(255) time : time date : date	<b>smartpot_project plant_details</b> id : bigint unsigned plant_name : varchar(255) plant_family : varchar(255) temp_low : float temp_high : float hum_low : float hum_high : float water_level : varchar(255) light_level : varchar(255)	<b>smartpot_project statusled</b> ID : int Stat : varchar(10)  <b>smartpot_project data</b> id : bigint unsigned humidity : double(8,2) temperature : double(8,2) created_at : timestamp updated_at : timestamp	<b>smartpot_project users</b> id : bigint unsigned name : varchar(255) email : varchar(255) email_verified_at : timestamp password : varchar(255) remember_token : varchar(100) created_at : timestamp updated_at : timestamp
<b>smartpot_project esp32_table_led_update</b> id : varchar(255) temperature : float(10,2) humidity : int status_read : varchar(255) LED_01 : varchar(255) PUMP : varchar(255) MODE : varchar(255) time : time date : date	<b>smartpot_project migrations</b> id : int unsigned migration : varchar(255) batch : int  <b>smartpot_project password_resets</b> email : varchar(255) token : varchar(255) created_at : timestamp	<b>smartpot_project personal_access_tokens</b> id : bigint unsigned tokenable_type : varchar(255) tokenable_id : bigint unsigned name : varchar(255) token : varchar(64) abilities : text last_used_at : timestamp created_at : timestamp updated_at : timestamp	

18. ábra Adatbázis táblák

## 4.2.3 Felhasznált könyvtárak

**OneWire:** Ez a könyvtár lehetővé teszi a OneWire protokoll alapú eszközök (például hőmérséklet-érzékelők) kommunikációját az ESP32 mikrovezérlővel. A OneWire protokoll egyszerű, egy dróton történő kommunikációt biztosít, amely lehetővé teszi az adatok küldését és fogadását.

**DallasTemperature:** Ez a könyvtár egy olyan kiterjesztés a OneWire könyvtárhoz, amely lehetővé teszi a Dallas Semiconductor (most Maxim Integrated) hőmérséklet-érzékelőinek használatát. Segítségével könnyedén kommunikálhatunk és olvashatjuk a DS18B20 típusú hőmérséklet-érzékelő adatait.

**WiFi:** Ez a könyvtár lehetővé teszi a WiFi-hálózatokhoz való csatlakozást, kapcsolódást és kommunikációt az ESP32 mikrovezérlővel. Segítségével a program kapcsolódhat egy meglévő WiFi-hálózathoz, és adatokat küldhet vagy fogadhat a hálózaton keresztül.

---

**HTTPClient:** Ez a könyvtár lehetővé teszi HTTP kérések küldését és fogadását. Segítségével a program például HTTP GET vagy POST kéréseket indíthat egy szerver felé, adatokat küldhet vagy fogadhat.

### 4.3. Biztonság

A biztonság kiemelten fontos szerepet játszik minden rendszer esetében. Fontos, hogy a rendszer megfelelő védelmet biztosítson az adatok tárolása és továbbítása során illetve a működése során.

A hálózati biztonság kiemelten fontos. Az okos virágtartó rendszer egy hálózathoz kapcsolódik, ami sebezhetővé teheti a külső támadásokkal szemben. Éppen ezért javasolt erősített biztonsági intézkedéseket alkalmazni, például erős jelszavakat használni és korlátozni a hozzáférést a rendszerhez.

A hardver komponensek biztonsága is roppant fontos. A felhasználóknak tisztában kell lenniük a rendszer működésével a lehetséges veszélyekkel és a biztonsági intézkedésekkel egyaránt.

A rendszerem kialakítása során alkalmaztam néhány biztonsági lépést, ugyanakkor ezeket még tovább lehet fejleszteni.

Néhány fontosabb biztonsági lépés, ami jellemzi a rendszert:

- A felhasználói felülethez regisztrálni kell, majd pedig bejelentkezni. Ezen lépések után a felhasználó hozzáférhet az adatokhoz illetve kezelni tudja a beállításokat.  
A bejelentkezést és regisztrációt a Laravel keretrendszer segítségével valósítottam meg, ami lehetővé teszi a felhasználók jelszavának biztonságos tárolását a megfelelő kriptográfiai algoritmusok segítségével. Alapértelmezetten a Laravel bcrypt algoritmust használja, amely erős jelszó-hash-elést valósít meg.
- A Laravel biztosítja a biztonságos munkamenet-kezelést, amely segít a felhasználók azonosításában és azonos munkameneten belüli tevékenységeik követésében. Ez segít megvédeni a felhasználói adatokat és csökkenti a biztonsági kockázatokat.

- 
- Az űrlap validáció is egy fontos szempont, hiszen megakadályozza a rossz adatok beküldését és segít elkerülni a biztonsági sebezhetőségeket, például az SQL-injekciót vagy a cross-site scripting-et (XSS).

#### 4.3.1 Mi történik a rendszerrel áramszünet esetében?

Áramszünet esetében az ESP32 mikrovezérlő elveszíti a tápellátását és leáll, a kód futása megszakad. Minden egyes tevékenység és kommunikáció félbeszakad. A rendszer vezérlése leáll és a LED-ek valamint a szivattyú nem működnek.

#### 4.3.2 Hiszterézis

“A hiszterézis (általában fizikai) rendszerek azon tulajdonsága, hogy érzéketlenek – nem azonnal reagálnak a rájuk ható erőkre, hanem késleltetéssel, vagy pedig nem térnek teljesen vissza az eredeti állapotukba: ezeknek a rendszereknek az állapota függ az előéletüktől.” Forrás : [Wikipédia](#) .

A rendszerem során is szükség volt alkalmazni a hiszterézist, megakadályozva, hogy a ledék és a pompa folyamatosan be- és kikapcsoljon, ha a hőmérséklet és nedvesség netalán ingadozna.

Az alábbi kódrészletek a ledék és a pompa ki- és bekapcsolási a logikáját valósítják meg.

Az első if azt ellenőrzi, hogy a "MODE" érték egyezik-e az "AUTO" stringgel (`strcmp(sensors["MODE"], "AUTO") == 0`) és a mért hőmérséklet (`tempC`) kisebb-e, mint a "tempLow" változó értéke vagy a "MODE" érték egyezik-e az "MANUAL" stringgel (`strcmp(sensors["MODE"], "MANUAL") == 0`) és a LED\_01 értéke ON (`strcmp(sensors["LED_01"], "ON") == 0`) . Ha a két feltétel közül az egyik igaz, akkor a LED- et bekapcsoljuk és kiírjuk a serial monitorra a hőmérséklet értékét illetve a LED állapotát.

A második if ág ugyanezt ellenőrzi, csak fordított esetben a felső határt figyelve. Ennek a logikának köszönhetően, a LED és a pompa nem fog folyamatosan ki - be kapcsolni, amennyiben a paraméterek ingadoznának. A LED bekapcsol, ha a hőmérséklet a minimális küszöbérték alá csökken és csak akkor kapcsol ki, ha elérte a maximális küszöbértéket, ami egy nagyobb intervallumot jelent.

---

```
if((strcmp(sensors["MODE"], "AUTO") == 0 && tempC < tempLow) || (strcmp(sensors["MODE"], "MANUAL") == 0 && strcmp(sensors["LED_01"], "ON") == 0 ) )
{
    digitalWrite(LED_01, LOW);
    ledState=true;
    Serial.printf("Homerseklet : %.2f °C\n", tempC);
    Serial.println("LED ON");
}
if((strcmp(sensors["MODE"], "AUTO") == 0 && tempC > tempHigh) || (strcmp(sensors["MODE"], "MANUAL") == 0 && strcmp(sensors["LED_01"], "OFF") == 0 ) )
{
    digitalWrite(LED_01,HIGH);
    ledState=false;
    Serial.printf("Homerseklet : %.2f °C\n", tempC);
    Serial.println("LED OFF");
}
```

### 19. LED-ek ki-bekapcsolási logikája

Az alábbi kódrészlet ugyanazt a logikát valósítja meg, akárcsak a fenti kód. Az egyetlen különbség az, hogy itt a pompát irányítjuk.

```
if((strcmp(sensors["MODE"], "AUTO") == 0 && humidity < humLow) || (strcmp(sensors["MODE"], "MANUAL") == 0 && strcmp(sensors["PUMP"], "ON") == 0 ) )
{
    digitalWrite(PUMP,LOW);
    Serial.printf("Nedvesseg : %d %%\n",humidity);
    Serial.println("PUMP ON");
}
if((strcmp(sensors["MODE"], "AUTO") == 0 && humidity > humHigh) || (strcmp(sensors["MODE"], "MANUAL") == 0 && strcmp(sensors["PUMP"], "OFF") == 0 ) )
{
    Serial.printf("Nedvesseg : %d %%\n",humidity);
    digitalWrite(PUMP, HIGH);
    Serial.println("PUMP OFF");
}
```

### 20. Pompa ki-bekapcsolási logikája

## 4.4. Felhasználói felület

A felhasználói felülethez Laravel keretrendszert használtam, amely segítségével sokkal könnyebben és időt spórolva tudtam megvalósítani az eltervezett design és a funkciókat. Fontos volt, hogy minél átláthatóbb és könnyen kezelhető legyen, megkönnyítve a rendszer használatát.

- Home oldal

A Home oldal, azaz kezdőoldal egy bemutatkozó oldal, ahol a felhasználó egy kis ízelítőt kaphat az egyetemről, rólam és a projektről. A képekre kattintva a jobb oldali szöveg változik.



21. ábra Home oldal

- Login oldal

A Bejelentkezés oldalon a felhasználó be tud jelentkezni a rendszerbe azokkal az adatokkal, amelyekkel regisztrált. A sikeres bejelentkezéshez meg kell adnia az e-mail címét és a jelszavát. Azt is kiválaszthatja, hogy a rendszer megjegyezze-e vagy sem.

A Laravel segítségével parancssorból létrehozhatjuk a bejelentkezés és regisztrációs oldalakat. Minden ehhez szükséges beállításról a keretrendszer gondoskodik.

Az alábbi sorokra van szükségünk:

**composer require laravel/ui:^2.4**

**php artisan ui vue --auth**

Lásd : <https://laravel.com/docs/7.x/authentication>

## BELÉPÉS

Login

Email Address

Password

☐ Remember Me

LOGIN

[Forgot Your Password?](#)

22. ábra Login oldal

THINK SMART  
LIVE SMART

LOG IN

REGISTER



Reset Password

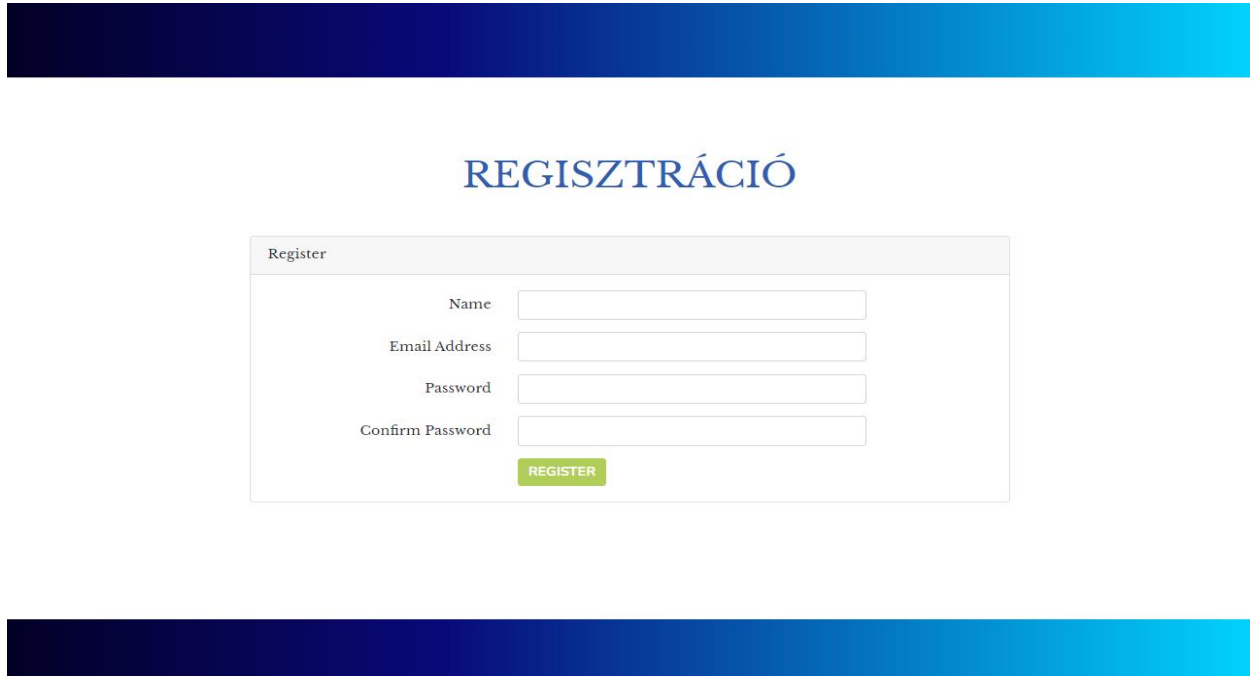
Email Address

Send Password Reset Link

23. ábra Jelszó visszaállítás



- 
- Register oldal



The image shows a registration form titled "REGISZTRÁCIÓ" (Registration). The form is contained within a light gray box with the title "Register" at the top left. It features four input fields: "Name", "Email Address", "Password", and "Confirm Password". Below these fields is a green "REGISTER" button. The form is set against a background with a blue-to-cyan gradient bar at the top and bottom.

*24. ábra Regisztrációs oldal*

- Insert oldal

Az Insert nevű oldalon az űrlap kitöltésével profilokat tudunk létrehozni a növényünk számára. Amennyiben automata módon szeretnénk működtetni a rendszerünket, az itt megadott küszöbértékek alapján végzi az öntözést és a világítást.

Az űrlap kitöltése során meg kell adnunk a növény nevét, hogy milyen családba tartozik, maximális és minimális hőmérsékletet illetve nedvességtartalmat. A mentés gombra kattintva az adatok elmentődnek az adatbázisba.

## NÖVÉNY HOZÁADÁS

Növény neve:

Növény család:

Minimális hőmérséklet:

Maximális hőmérséklet:

Minimális nedvességtartalom:

Maximális nedvességtartalom:

Mentés

25. ábra Insert oldal

- Records oldal

A Records oldalon az adatbázisba elmentett növény profilekat tudjuk megtekinteni.

## NÖVÉNYEK

NÖVÉNY NEVE	NÖVÉNY CSALÁD	MAXIMÁLIS HŐMÉRSÉKLET	MINIMÁLIS HŐMÉRSÉKLET	MAXIMÁLIS NEDVESSÉGTARTALOM	MINIMÁLIS NEDVESSÉGTARTALOM	VÍZ IGÉNY	FÉNY IGÉNY
Rozsa	virag	0	0	0	0	normal	normal
Paprika	zoldseg	18	25	20	80	normal	normal
Eper	gyumolcs	28	35	50	90	normal	normal

26. ábra Records oldal

- Charts oldal

A Charts oldalon a nedvesség és hőmérséklet adatok alapján ábrázolt diagramokat láthatjuk.

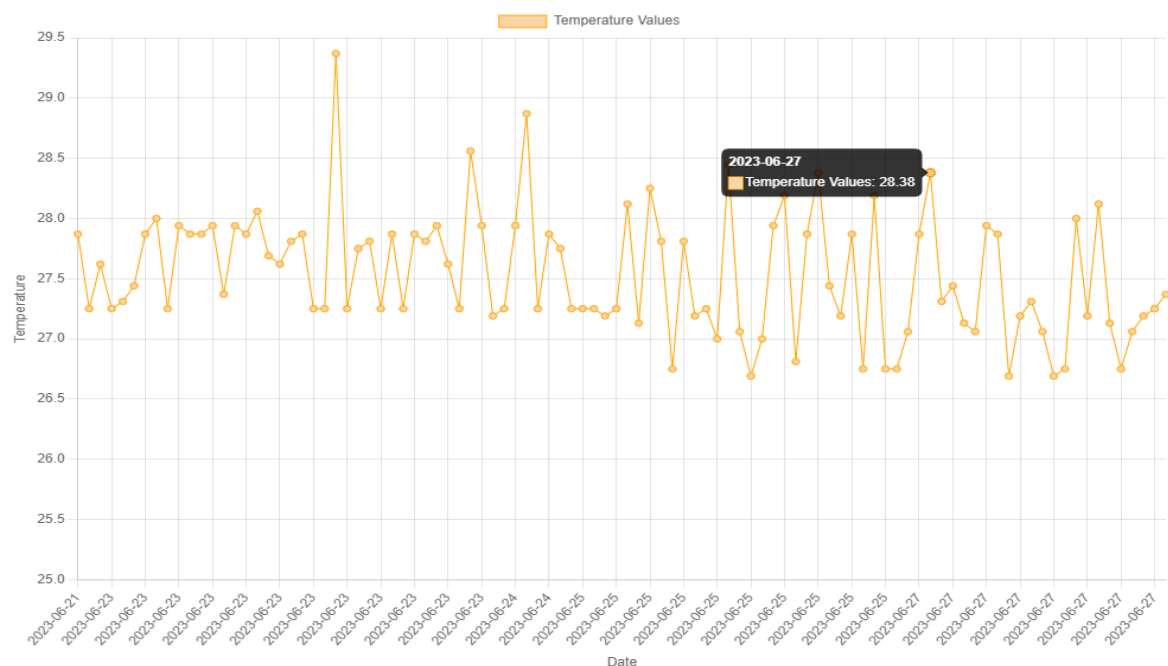
A diagramok kirajzolásához a Chart.js ingyenes és nyílt forráskódú JavaScript könyvtárat használtam, amely lehetővé teszi interaktív és testreszabható diagramok létrehozását a webes alkalmazásokban. A Chart.js diagramok könnyen beilleszthetők weboldalakba vagy webes alkalmazásokba, és egyszerűen kezelhetők a dokumentáció által biztosított API segítségével.

A megvalósításról bővebben:

<https://www.itsolutionstuff.com/post/how-to-add-charts-in-laravel-5-using-chart-js-example.html>

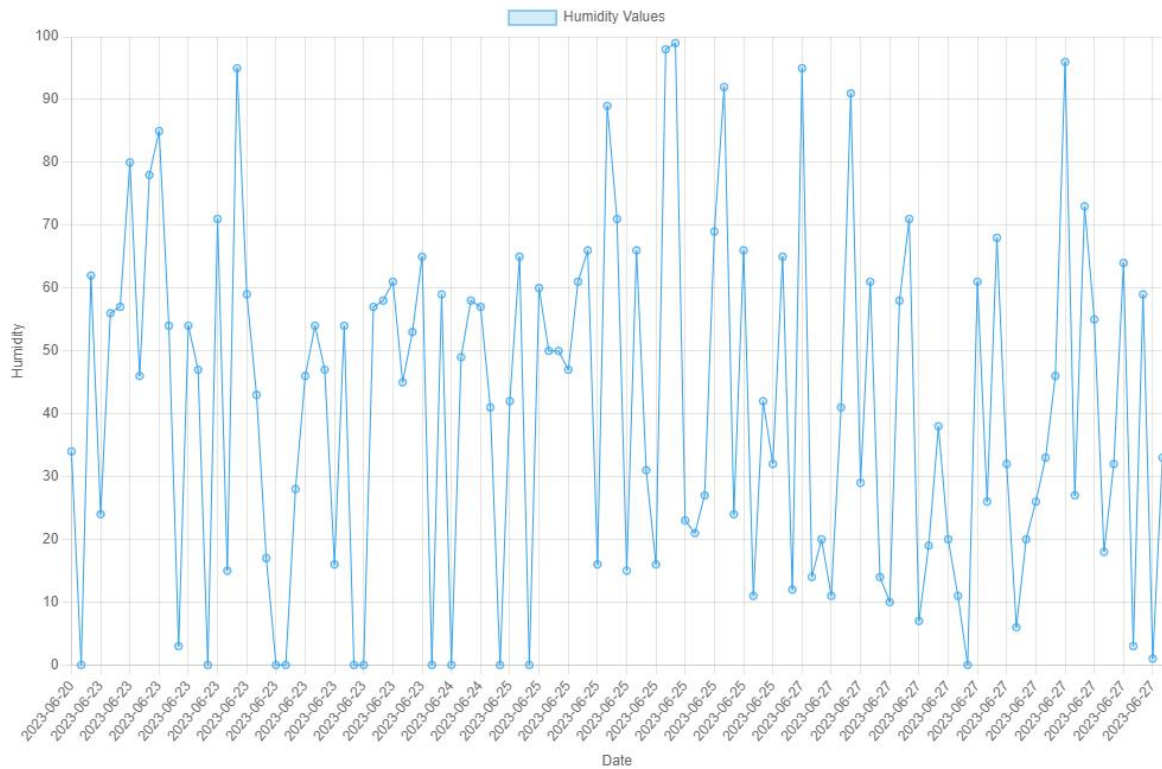


## Temperature



27. Charts oldal - diagram 1

## Humidity

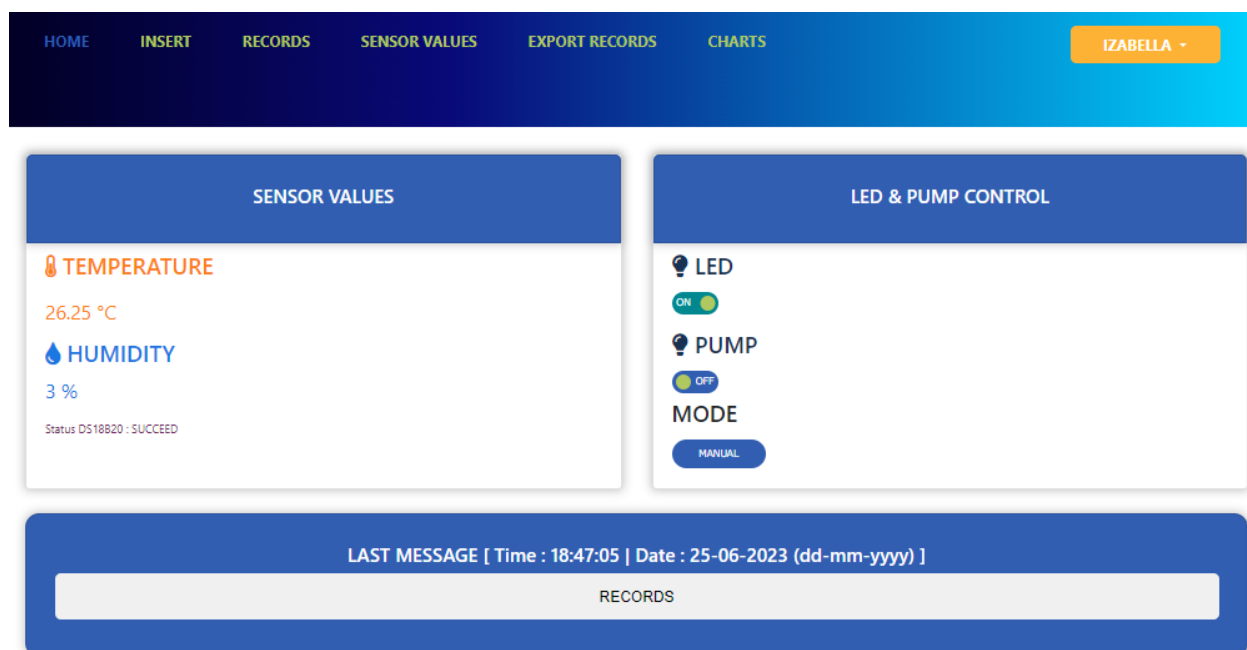


28. Charts oldal - diagram 2

- Sensor values oldal

A sensor values oldalon nyomon tudjuk követni a szenzorok által mért környezeti paramétereket (hőmérséklet és nedvesség) illetve hogy sikeres volt-e a DS18B20 szenzor olvasása. A jobb oldali doboz pedig a pompa és a led irányítását teszi lehetővé. Módunkban áll kiválasztani azt is, hogy automata vagy manuális módon szeretnénk vezérelni a rendszert. Amennyiben automata módra van állítva, a növény profil létrehozása során megadott küszöbértékeknek megfelelően fogja a rendszer kielégíteni a növény szükségleteit. Azt is láthatjuk, hogy mikor kaptunk utoljára adatot az ESP32 mikrovezérlőtől.

A Records gombra kattintva megtekinthetjük az esp32\_table\_record táblába elmentett összes adatot .



29. ábra Sensor values oldal

- Export records oldal

Az Export records oldalon, táblázat formájában kilistázva vannak az esp32\_table\_record tábla tartalma illetve lehetőségünk van az adatokat letölteni .xlsx és csv formátumban.



## ADATOK EXPORTÁLÁSA

CSV Export Excel Export

NO	ID	BOARD	TEMPERATURE (°C)	HUMIDITY (%)	STATUS READ SENSOR DS18B20	LED	PUMP	TIME	DATE
1	mOTkhqHv9W	esp32_01	22.6	46	SUCCEED	OFF		14:26:07	11-05-2023
2	C5jqUEaKeE	esp32_01	22.6	46	SUCCEED	OFF		14:26:12	11-05-2023
3	H5OOwpF9kw	esp32_01	22.6	46	SUCCEED	OFF		14:26:17	11-05-2023
4	URyA4WlaLM	esp32_01	22.6	46	SUCCEED	OFF		14:26:23	11-05-2023
5	X4guOEhFDK	esp32_01	22.6	46	SUCCEED	OFF		14:26:28	11-05-2023
6	NBsWvIGJQm	esp32_01	22.6	46	SUCCEED	ON		14:26:34	11-05-2023
7	999JEISDIT	esp32_01	22.6	46	SUCCEED	OFF		14:26:39	11-05-2023
8	91WEEIngvD	esp32_01	22.6	46	SUCCEED	OFF		14:26:45	11-05-2023
9	ZbUdXls8D6	esp32_01	22.6	46	SUCCEED	OFF		14:26:50	11-05-2023
10	C3ACmeKkdn	esp32_01	22.6	46	SUCCEED	OFF		14:26:55	11-05-2023

Prev Next Table : 1/941 (Total Number of Rows = 9410) | Number of Rows : 10 Apply

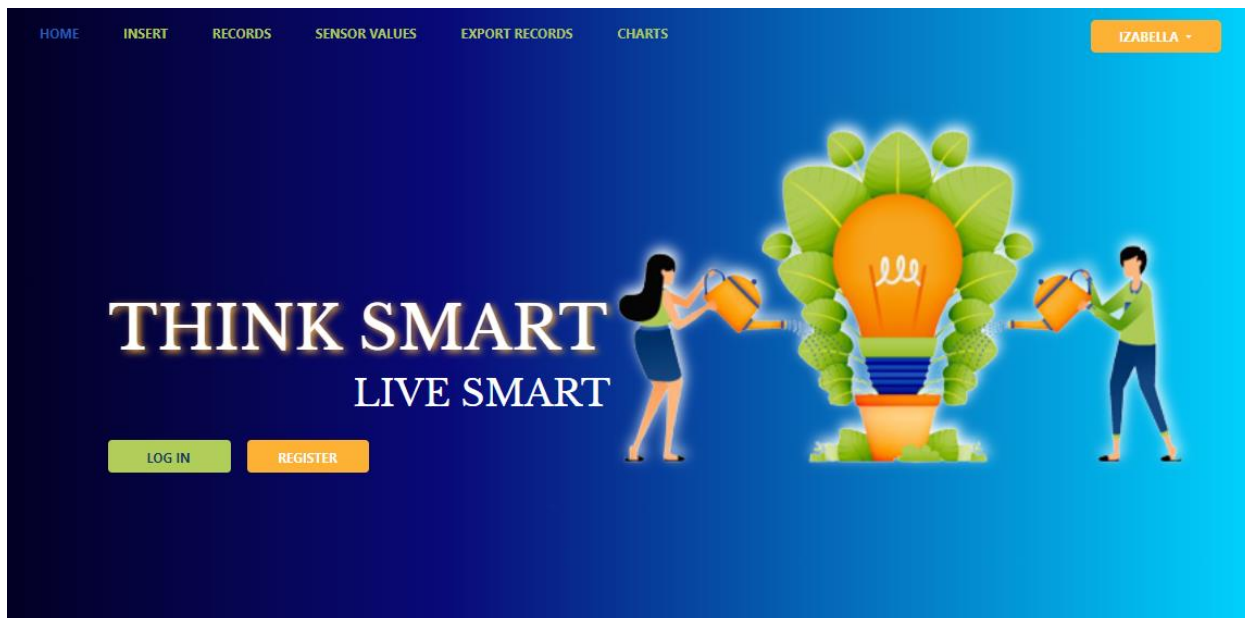
30. ábra Export records oldal

Az alábbi képen látható az excel táblázat, ami tartalmazza az adatbázis táblába elmentett adatokat :

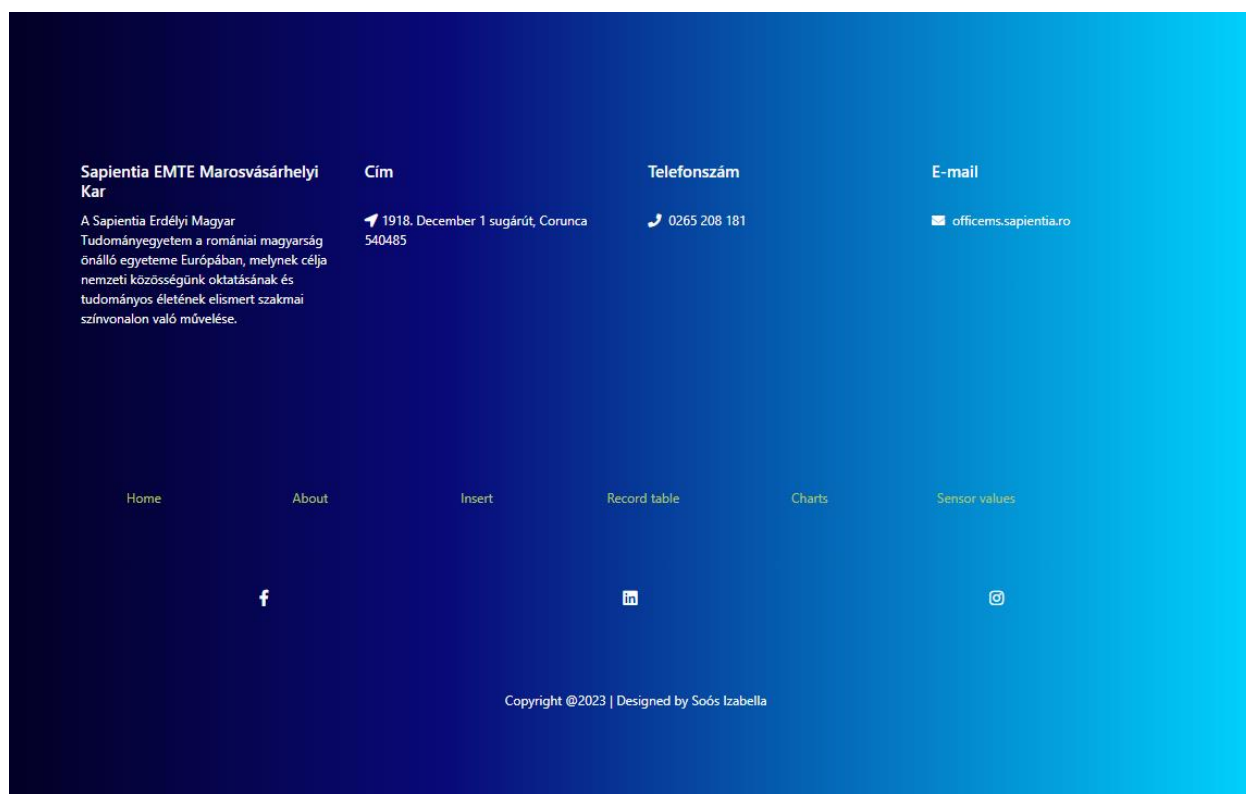
ID	Device ID	Value	Status	Date
25	esp32_01	25.06	SUCCEED ON	20:12:29 2023-06-01
26	esp32_01	27.13	67 SUCCEED ON	16:56:32 2023-06-09
27	esp32_01	27.13	16 SUCCEED ON	21:52:19 2023-06-23
28	esp32_01	28.25	89 SUCCEED ON	18:39:52 2023-06-23
29	esp32_01	26.94	7 SUCCEED ON	23:39:59 2023-06-24
30	esp32_01	24.8	38 SUCCEED ON	14:30:44 2023-06-01
31	esp32_01	25.31	SUCCEED OFF	22:21:00 2023-06-01
32	esp32_01	27.81	76 SUCCEED OFF	09:39:09 2023-06-02
33	esp32_01	23.87	SUCCEED OFF	21:14:16 2023-06-01
34	esp32_01	26.75	11 SUCCEED ON	00:17:05 2023-06-25
35	esp32_01	27.81	71 SUCCEED ON	19:11:48 2023-06-23
36	esp32_01	25.56	60 SUCCEED OFF	16:36:56 2023-06-18
37	esp32_01	25.25	SUCCEED OFF	22:13:08 2023-06-01
38	esp32_01	27.62	SUCCEED ON	16:14:16 2023-06-09
39	esp32_01	27.13	SUCCEED ON	17:10:44 2023-06-09
40	esp32_01	22.94	84 SUCCEED OFF	19:53:31 2023-06-18
41	esp32_01	26.75	9 SUCCEED ON	00:52:01 2023-06-25
42	esp32_01	26.81	90 SUCCEED ON	01:41:42 2023-06-25
43	esp32_01	27.19	32 SUCCEED ON	17:08:13 2023-06-09
44	esp32_01	26.5	SUCCEED ON	21:33:54 2023-06-07
45	esp32_01	26.75	15 SUCCEED ON	22:35:20 2023-06-23
46	esp32_01	27.81	74 SUCCEED OFF	09:38:57 2023-06-02
47	esp32_01	23	44 SUCCEED ON	14:56:59 2023-05-11
48	esp32_01	24.75	SUCCEED ON	20:35:02 2023-06-01
49	esp32_01	25.75	70 SUCCEED ON	23:48:51 2023-06-04
50	esp32_01	27.81	66 SUCCEED ON	19:10:08 2023-06-23

31. ábra Excel export

A weboldal tartalmaz egy fejléc és lábléc részt is. A fejlécről elérhetjük a Login és Register oldalakat is, illetve amennyiben a felhasználó be van jelentkezve, a jobb oldali gombra kattintva, (ami a bejelentkezett felhasználó nevét is megjeleníti) ki tud jelentkezni.



32. ábra Fejléc



33. ábra Lábléc

## 4.5. Tervezési fázisok

A tervezési fázis minden projekt alapköve, amely biztosítja a szilárd alapot a megvalósítás számára. A következőkben bemutatom a tervezési folyamatot, amelyet a rendszerem megvalósítása során alkalmaztam. A tervezés célja a projekt részletezése, az elemzés, a követelmények definiálása és a megfelelő erőforrások és stratégiák kiválasztása.

### 4.5.1 Követelmények elemzése

Az első lépés a projekt céljainak és követelményeinek megértése volt. Ez magába foglalta az igények és elvárások felmérését és a hasonló rendszerek kutatását. Az ezen fázis során készült elemzés segített abban, hogy meghatározzam a projekt határait és a tervezési döntéseket.

### 4.5.2 Funkcionális tervezés

A funkcionális tervezés során a projekt céljai alapján részletesen meghatároztam a funkcionalitást. Ez magában foglalja a főbb modulok és alrendszerek azonosítását, a felhasználói interakciók tervezését és a funkciók hierarchiájának meghatározását. A folyamat során részletesen vizsgáltam a funkcionalitás jellemzőit és követelményeit.

---

#### **4.5.3 Szerkezeti tervezés**

A szerkezeti tervezés során meghatároztam a rendszer architektúráját és az alkalmazott technológiákat. Ez magában foglalta az adatbázisom tervezését, a rendszerkomponensek és interfészek meghatározását, valamint a rendszer architekturális diagramjainak elkészítését.

#### **4.5.4 Adatbázis tervezés**

Az adatbázis tervezés a projekt adatmodelljének és adatbázis sémájának kidolgozását jelentette. Ez magában foglalta az adatbázis kiválasztását illetve az entitások, attribútumok és kapcsolatok meghatározását, valamint az adatbázis lekérdezések és táblák tervezését.

#### **4.5.5 Felhasználói felület tervezése**

A felhasználói interfész tervezése során kiválasztottam a keretrendszert, aminek segítségével könnyen megvalósítottam a felhasználói felületet. Miután eldöntöttem, hogy a felhasználói felület egy weboldal lesz, részletesen megterveztem a különböző elemek, ikonok, menük és navigációs rendszerek logikáját és designját. Fontos volt, hogy minél egyszerűbb és átláthatóbb legyen.

#### **4.5.6 Tesztelési tervezés**

A tesztelési tervezés során meghatároztam a tesztelési stratégiát és módszereket. Rögzítettem a tesztelési eseteket, a tesztadatokat és az elvárt eredményeket, annak érdekében, hogy minél kedvezőbb eredményeket kapjak.

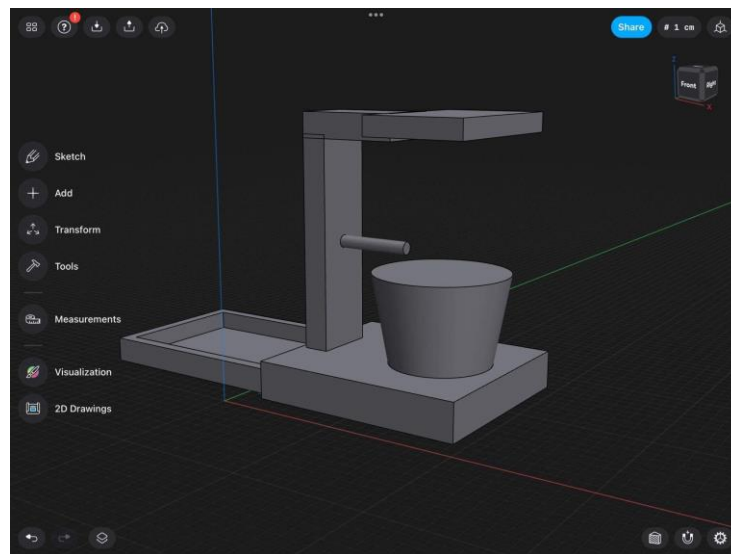
#### **4.5.7 Fizikai tartó**

Szerettem volna, ha az államvizsga dolgozatom egy szép design- al is rendelkezne, ezért a 3D nyomtatás mellett döntöttem. Részletes tervezés után és filament keresés után végre elkészült a 3D terv majd ezt követte a nyomtatás megkezdése.

A legelső fázisban még nem tudtam a pontos méreteket, így csak egy lehetséges változata készült el a virágtartóról.

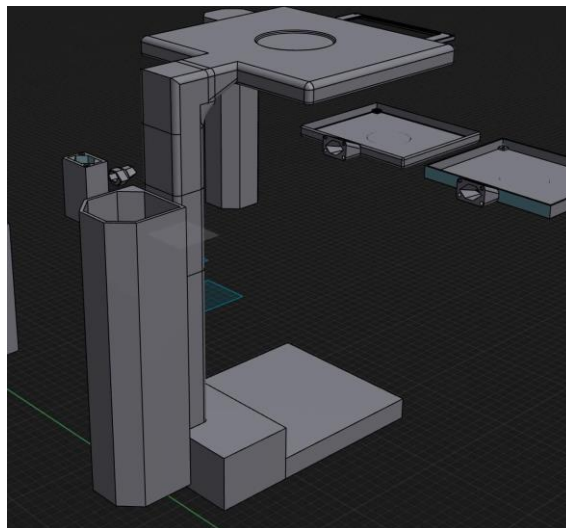
Az alábbi ábrán látható a legelső terv:



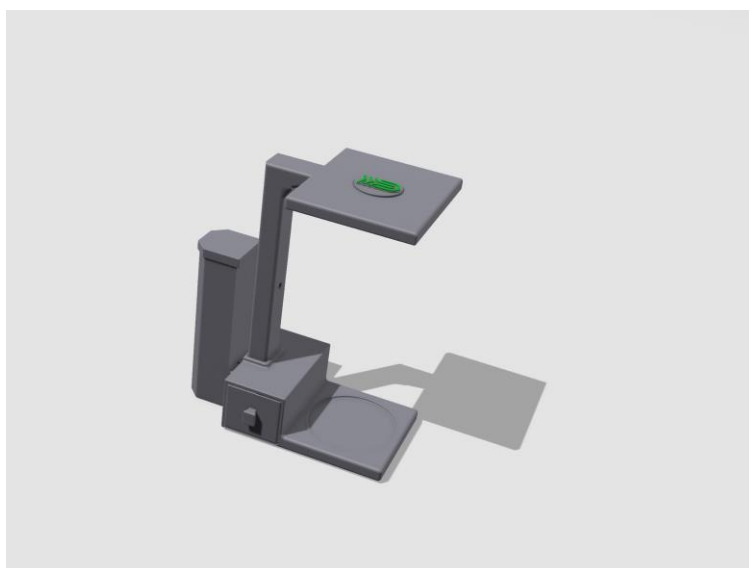


*34. ábra Fizikai design első prototípus*

A víztartály úgy volt megtervezve, hogy egy AQUA- s palack elférjen benne. Ebbe kerül be a pompa és természetesen innen jön ki a cső, amin keresztül áramlik a víz először a hozammérő fele majd a növényhez. A tartály belső falán van egy rés, amin keresztül a csövet ki lehet vezetni.



*35. ábra A virágtartó hátulról - víztartály tervezése*



*36. ábra Végleges design*

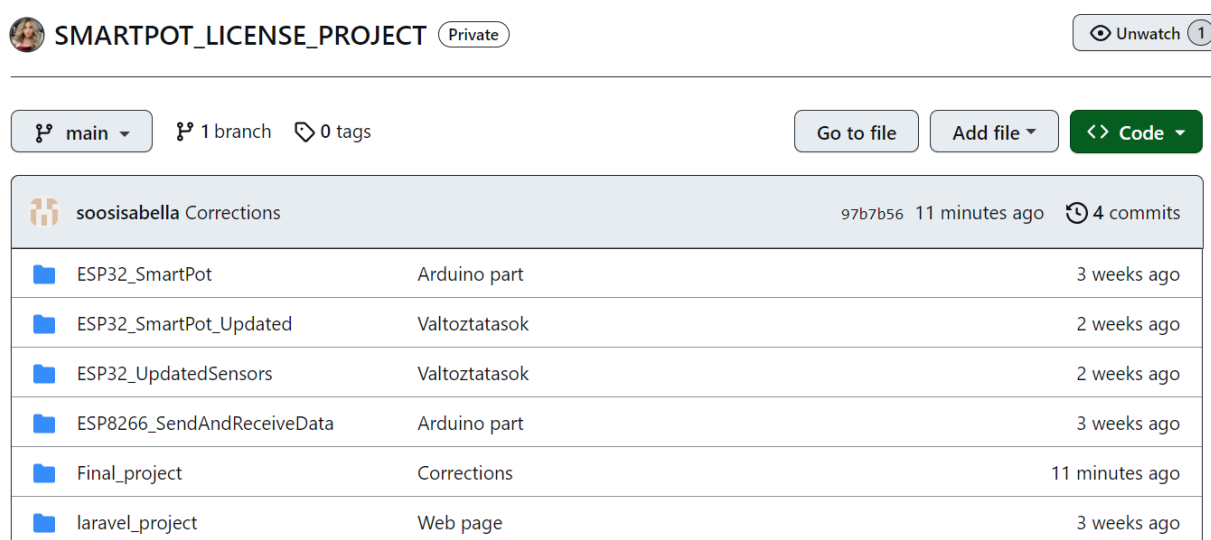


*37. ábra Fizikai tartó*

## 4.6. Verziókövetés

A verziókövetés illetve biztonság végett Github verziókövetőt használtam.

Az alábbi ábrán látható a SMARTPOT\_LICENSE\_PROJECT néven létrehozott repository, ahova commitoltam a kódokat nagyobb változtatások után.



38. ábra Github repository

## 5. Üzembe helyezés és kísérleti eredmények

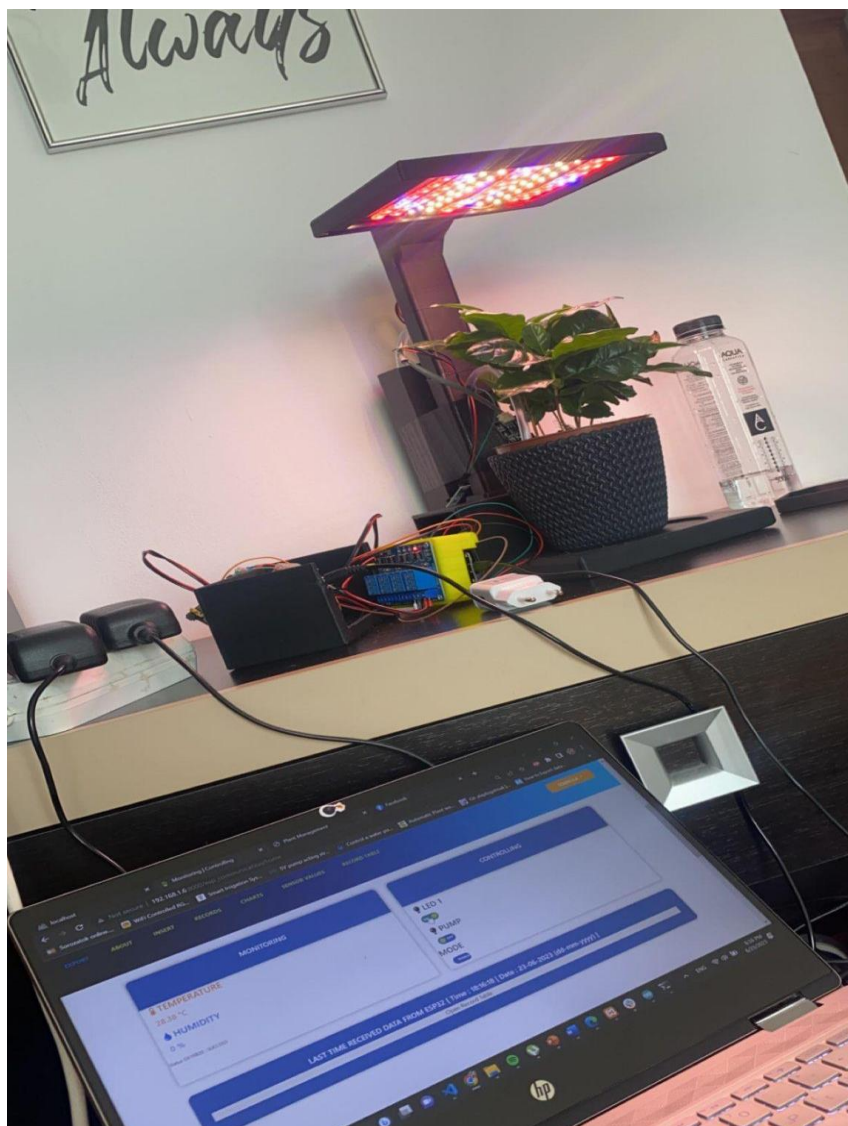
### 5.1. Üzembe helyezés

Egy rendszer üzembe helyezése a tervezési folyamat legvégén történik. Az üzembe helyezés célja, hogy a fejlesztett rendszert bevezessük a valós környezetbe és működőképességét biztosítsuk. Az én esetemben az üzembe helyezés folyamata a következő lépéseket foglalta magába:

1. Környezet előkészítése
2. Telepítés és konfiguráció
3. Nyák elkészítése
4. Alkatrészek összekapcsolása
5. Rendszer elindítása

---

Az alábbi ábrán látható a rendszer működés közben:



39. ábra A rendszer működés közben

## 5.2. Felmerült problémák és megoldásaik

Bármilyen rendszer megtervezése és megépítése közben problémák merülhetnek fel. Nem volt ez másképp az én esetemben sem. Hiába tervezünk előre, vannak bizonyos problémák, amelyek időközben merülnek fel és megoldásra várnak, azonban minden egyes probléma egy tanulság volt és úgy érzem sokat segítettek a fejlődésben.

Néhány probléma, amivel szembesültem:

- 
1. Amikor a mikrovezérlőről próbáltam adatot továbbítani az adatbázisba, a következő hibaüzenetet kaptam: “ **Access denied for user 'root@localhost' (using password:NO)** ”. Rengeteg fórumot felkerestem ezzel kapcsolatban, próbáltam parancssorból is jelszót beállítani, sikertelenül.  
A megoldás az volt, hogy újratelepítettem a MySQL- t és telepítés során új jelszót állítottam be.
  2. A weboldalamat Laravel keretrendszer segítségével fejlesztettem, és eredetileg csak a helyi gépen, a 127.0.0.1 IP-címen volt elérhető. Hogy a weboldalhoz hozzáférjünk a mikrovezérlőn keresztül is, IPv4 címet kellett használnjak.  
A megoldás az volt, hogy a Laravelben a rendszert a következő paranccsal indítottam el: **php artisan serve --host 0.0.0.0**.  
Ez a parancs a Laravel beépített fejlesztői szerverét indítja el, és a --host 0.0.0.0 opció beállítása azt jelenti, hogy a szerver az összes elérhető hálózati interfészen figyel.
  3. Eredetileg a hozammérő, amit vettem YF-S201 modell volt, ami 1-30 L vizet képes érzékelni. Ha a pompát 5V- al tápláltam a hozammérő jól mért, viszont a víz túl gyorsan folyt, ami nem volt ideális. Ha a pompát 3V- al tápláltam, a hozammérő nem érzékelte a rajta keresztül átfolyó vizet.  
Megoldásként lecseréltem a hozammérőt egy kisebb verzióra, aminek a mérési tartománya 0.6-6 L.

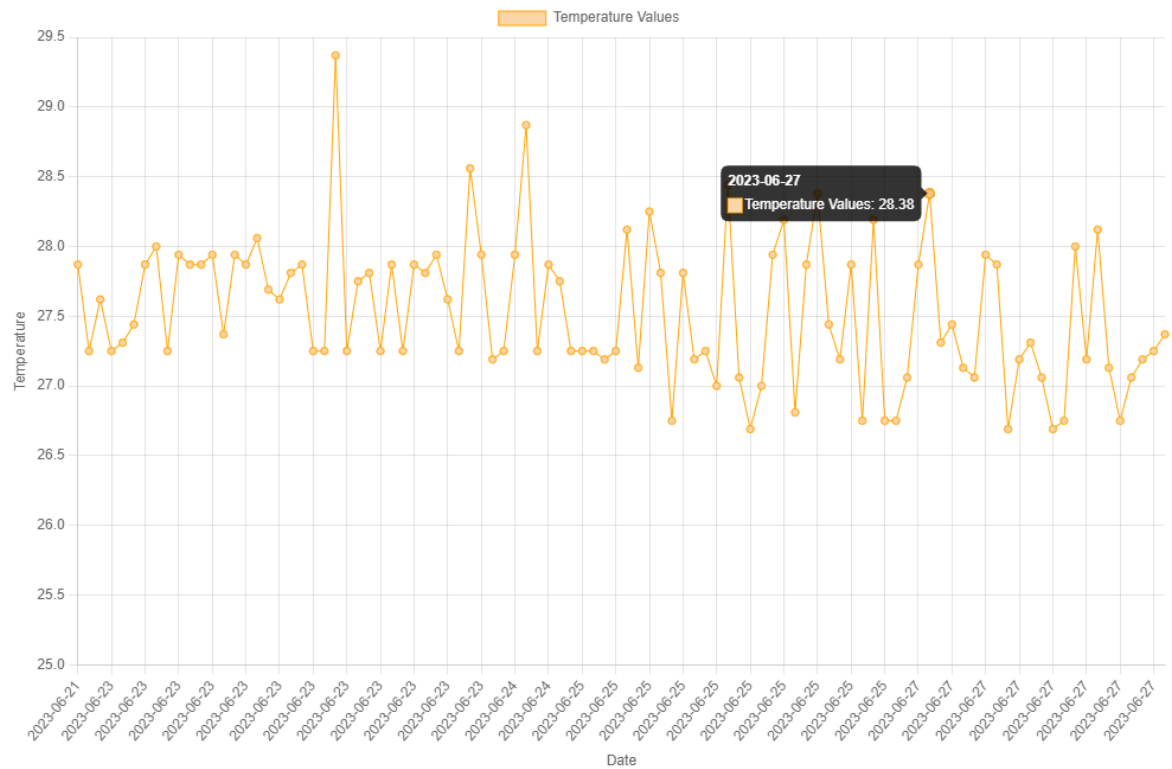
### 5.3. Kísérleti eredmények, mérések

A mérések és kísérleti eredmények a tesztelés fontos részét képezik. Ahhoz, hogy megbizonyosodjak a rendszer megfelelő működéséről, bevezettem egy új növényt, aminek megadtam küszöbértékeit és figyeltem a rendszer reakcióját.

Amint a küszöbérték alá vagy fölé ért a hőmérséklet illetve nedvességtartalom, a rendszer megfelelően reagált rá, eleget téve az elvárásoknak. A manuális vezérlés is megfelelően működik.

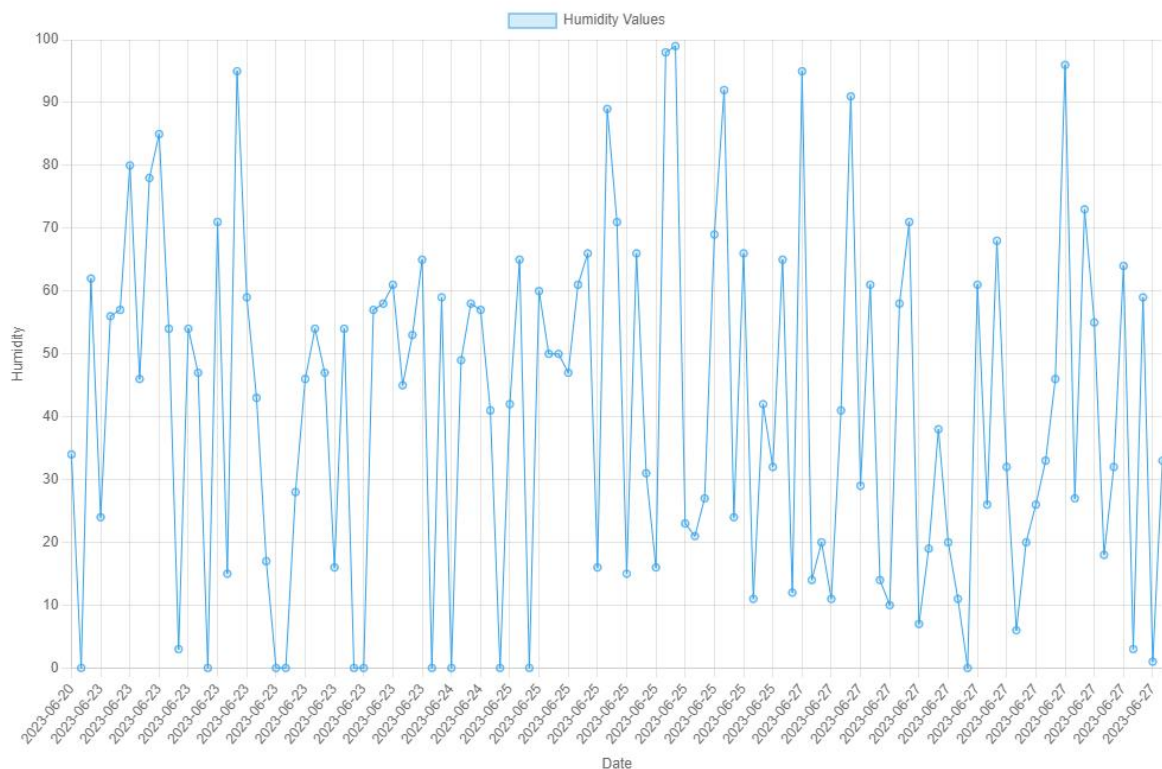
Ahhoz, hogy minél hatékonyabban nyomon tudjuk követni a paraméterek változásait, diagramokat rajzoltam ki, ezáltal is megkönnyítve a rendszer megértését.

# Temperature



40. ábra Hőmérséklet diagram

## Humidity



41. ábra Nedvesség diagram

### 5.4. Tesztelés

A szoftvertesztelés során a szoftvertervezők és fejlesztők tesztelik a szoftver alkalmazásokat, annak érdekében, hogy biztosítsák a minőséget és a megfelelő működést a felhasználók számára.

A szoftvertesztelés során különböző tesztek futtatásával ellenőrizhető a szoftver megfelelő működése, az alkalmazás által biztosított funkciók teljesítménye, a biztonság és az adatvédelem, valamint a felhasználói élmény. A tesztek lehetnek manuálisak, amelyek során egy személy teszteli az alkalmazást, vagy automaták, amelyeket szoftverek hajtanak végre.



---

Számos előnye van a szoftvertesztelésnek, mint például a hibák és problémák korai azonosítása, a biztonságos és hatékony alkalmazások kialakítása, a felhasználói elégedettség növelése és a költségek csökkentése.

Mivel a UI részhez Laravel keretrendszert használtam, ezért ehhez kellett alkalmazkodnom és ezen belül kellett végezni a teszteléseket.

A Laravel egy PHP alapú webalkalmazás-keretrendszer, amelynek része a beépített tesztelési támogatás is. A Laravel tesztelési támogatása lehetővé teszi a tesztelést az alkalmazás különböző részein, beleértve a vezérlőket, a modell osztályokat, a nézeteket és az adatbázis műveleteket is.

A Laravel tesztelési támogatása a PHPUnit tesztkeretrendszerre épül, ami lehetővé teszi úgy a funkcionális akár csak az egységteszt írását is.

A PHPUnit segítségével a Laravel alkalmazás tesztelése könnyedén és hatékonyan végezhető el. A funkcionális tesztekkel a teljes alkalmazás működését tudjuk tesztelni, beleértve a HTTP kérések feldolgozását és a válaszok ellenőrzését is. Az egységteszt a különböző alkalmazási részek, például az adatbázis műveletek, a vezérlők, a modell osztályok és a nézetek tesztelésére szolgálnak.

A tesztek írása segíthet a hibák és problémák korai azonosításában, a kódminőség javításában, valamint a biztonságos és megbízható alkalmazások kialakításában.

A tesztek nagyon egyszerűen létre tudjuk hozni parancssorból : **php artisan make:test <Name>** . Itt a <Name> helyére a teszt osztály neve kerül. **php artisan make:test <Name> - - unit** (unit test létrehozása)

Az előző lépésben létrehozott parancs létrehozza a teszt osztályunkat a tests/Feature vagy tests/Unit mappában, attól függően, hogy milyen típusú tesztről van szó. Az egységteszt az Unit mappában találhatóak, míg a funkcionális tesztek a Feature mappában.

Lásd bővebben itt : <https://laravel.com/docs/8.x/testing> ,

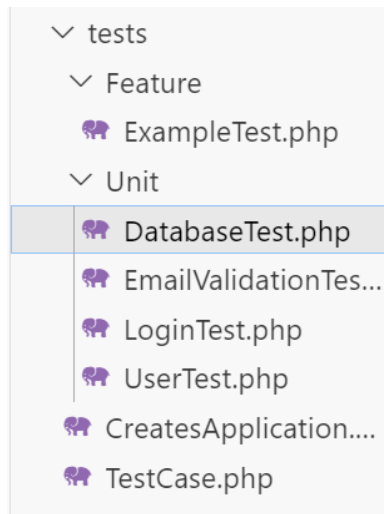
<https://laravel-news.com/how-to-start-testing> .

A tesztek futtatni lehet parancssorból a következő parancsokkal :



---

**php artisan test** vagy **.vendor/bin/phpunit** . Ez a parancs futtatja az összes tesztet ami a “tests/” mappában található fájlokban van. Az alábbi ábrán látható a “tests” mappa :



42. ábra Tesztek - fájlrendszer

Hogyha csak egy adott tesztet szeretnénk futtatni, akkor azt az alábbi paranccsal tehetjük meg, ahol a <test> helyére a teszt nevét kell beírni: **php artisan test --filter <test>** .

Az eredményeket a terminálban érhetjük el. A sikeres tesztek zöld, míg a hibás tesztek piros színnel jelennek meg. Az eredmények mellett a tesztek futtatási ideje is fel van tüntetve.

A tesztek futtatása után, törölhetjük a tesztek által létrehozott adatokat az adatbázisból az alábbi paranccsal: **php artisan migrate:fresh** .

Tesztek :

## DatabaseTest.php

```
WARN Tests\Unit\DatabaseTest
✓ database connection
✓ database
✓ plant exists
! seeders → This test did not perform any assertions C:\Users\gallm\laravel_project\tests\Unit\DatabaseTest.php:41

public function testDatabaseConnection()
{
    $this->assertTrue(\DB::connection()->getDatabaseName() === env('DB_DATABASE'));
}
```

---

---

#### 43. ábra Teszt 1

A fenti kódrészlet célja, hogy ellenőrizze, hogy a Laravel alkalmazás adatbázis kapcsolata helyesen van beállítva-e vagy sem.

Megpróbál kapcsolódni az adatbázishoz a `\DB::connection()` metódus segítségével, majd próbálja megszerezni az adatbázis nevét a `getDatabaseName()` metódussal és ellenőrzi, hogy az adatbázis neve megegyezik-e az `env('DB_DATABASE')` konfigurációs változó értékével.

Ha az adatbázis kapcsolódás helyes, akkor a teszt sikeresen lefut, és a `assertTrue()` metódus `true` értéket ad vissza. Ha sikertelen vagy a neve nem azonos, akkor a teszt hibát fog dobni és a PHPUnit jelzi, hogy a teszt sikertelen.

```
public function test_database(){  
    $this->assertDatabaseHas('users',[  
        'name'=>'Pinter3'  
    ]);  
}
```

#### 44. ábra Teszt 2

Ezzel a teszttel az adatbázis tartalmát tudjuk ellenőrizni. A teszt akkor sikeres, ha az adatbázis tartalmaz egy olyan rekordot, amely eleget tesz a teszt feltételeinek. Ha az adatbázis nem tartalmazza a rekordot, akkor a teszt hibás lesz.

Az `assertDatabaseHas()` metódus segítségével ellenőrizni tudjuk, hogy a `users` táblában található-e olyan rekord, amelynek a `name`, azaz név mezője `Pinter3`.

```
public function test_plant_exists(){  
    $this->assertDatabaseHas('plant_details',[  
        'plant_name'=>'Rozsa'  
    ]);  
}
```

#### 45. ábra Teszt 3

Ez a teszt is szintén azt ellenőrzi, hogy az adatbázisban létezik-e egy bizonyos rekord. A teszt akkor sikeres, ha az adatbázis tartalmazza az adott rekordot, ellenkező esetben a teszt hibát dob.

---

Az `assertDatabaseHas()` metódus segítségével ellenőrizni tudjuk, hogy a `plant_details` táblában van-e olyan rekord, aminek a `plant_name` mezője Rozsa.

```
public function test_seeders(){  
    $this->seed(); //Seedeli az osszes seedert ,olyan mint a db:seed parancs  
}
```

46. ábra Seeder teszt

A fenti teszt az adatbázis seederek megfelelő működését teszteli. A `seed()` metódus segítségével inicializálja az adatbázist az összes seedert tartalmazó adatbázis seeder segítségével.

### EmailValidationTest.php

**PASS** Tests\Unit>EmailValidationTest  
✓ email field must be a valid email

```
public function test_email_field_must_be_a_valid_email()  
{  
    $response = $this->post('/register', [  
        'name' => 'Test User',  
        'email' => 'invalid_email',  
        'password' => 'password',  
        'password_confirmation' => 'password',  
    ]);  
  
    $response->assertStatus(302);  
    $response->assertSessionHasErrors(['email']);  
}
```

47. ábra Email teszt

Ez a kódrészlet az űrlap validációját ellenőrzi, amikor egy új felhasználót regisztrálunk. Ellenőrzi, hogy a megadott e-mail cím érvényes-e vagy sem.

Az `assertStatus()` metódus segítségével ellenőrizni tudjuk, hogy az űrlap beküldése után a válasz státuszkódja 302 (Found, átirányítás) vagy nem. Sikertelen validáció esetén a Laravel a felhasználót visszairányítja az űrlapra, ezért a válasznak 302 státuszkóddal kell rendelkeznie.

Ha az űrlap beküldése során az e-mail mező érvénytelen értéket tartalmaz, akkor a teszt sikeres lesz. Ha az e-mail mező érvényes értéket tartalmaz, akkor a teszt hibát fog dobni.

---

## LoginTest.php

```
public function test_create_model()
{
    // $user = factory('App\User')->create();
    $user = new User();
    $user->name = "Alma";
    $user->email = "alma@gmail.com";
    $user->password = "12345";
    $user->save();

    $userModelExists = $user ? true : false;

    $this->assertTrue($userModelExists);

    //beloginolas es homera valo atiranyitas
    $response = $this->actingAs($user)->get('/home');

    $response->assertStatus(200);
}
```

48. ábra Login teszt

Ez a teszt az adatbázisba való mentést és bejelentkezést ellenőrzi. Létrehoz egy új User modellt, kitölti a szükséges mezőket és elmenti az adatbázisba. Ezek után ellenőrzi, hogy a modell sikeresen létrejött-e vagy sem. Megpróbál bejelentkezni a felhasználói modellel, majd lekérdezi a /home útvonalat. Végül pedig ellenőrzi, hogy az útvonal válasza 200-as státuszkódot tartalmaz-e.

Ha a teszt sikeresen lefut, akkor azt jelenti, hogy az új User modell el lett mentve az adatbázisba és hogy a bejelentkezés megfelelően működik. A teszt hibát fog adni abban az esetben, ha valamelyik lépést nem sikerül végrehajtani.

## UserTest.php

- ✓ user duplications
- ✓ create user with unique email
- ✓ delete user
- ✓ it stores new users

---

```
public function test_user_duplications()
{
    $user1 = User::make([
        'name' => "Pinter Bela",
        'email' => "pinterbela@gmail.com"
    ]);
    $user2 = User::make([
        'name' => "Pinter Bela",
        'email' => "pinterbela@gmail.com"
    ]);

    $this->assertTrue($user1->name == $user2->name);
}
```

*49. ábra Dupla felhasználó teszt*

Ez a teszt két felhasználót hoz létre, amelyeknek azonos a nevük és az e-mail címük. Azután azt ellenőrzi, hogy a két felhasználó neve azonos-e. Ha igen, akkor a teszt sikeresnek tekinti a műveletet.

```
public function test_create_user_with_unique_email()
{
    $user1 = User::factory()->create([
        'email' => 'pinteranna@gmail.com',
    ]);

    $response = $this->post('/register', [
        'name' => 'Kovacs Peter',
        'email' => 'pinteranna@gmail.com',
        'password' => '12345',
        'password_confirmation' => '12345',
    ]);

    $response->assertSessionHasErrors(['email']);
}
```

*50. ábra Egyedi e-mail teszt*

Ebben a tesztben először létrehozunk egy felhasználót a "pinteranna@gmail.com" email címmel. Ezután próbálunk létrehozni egy új felhasználót ugyanezzel az email címmel a /register útvonalon keresztül. Mivel az email cím már használatban van, a regisztrációs űrlap feldolgozása során hibaüzenetet kell kapnunk. Ezt az üzenetet az assertSessionHasErrors metódus segítségével ellenőrizzük.

---

```

public function test_delete_user()
{
    $user = User::factory()->create();
    // Ellenőrizzük, hogy a felhasználó valóban létezik az adatbázisban
    $this->assertDatabaseHas('users', ['id' => $user->id]);

    $user->delete();

    // Ellenőrizzük, hogy a felhasználó már nem létezik az adatbázisban
    $this->assertDatabaseMissing('users', ['id' => $user->id]);
}

```

### 51. ábra Törlés teszt

A teszt az `assertDatabaseHas` és `assertDatabaseMissing` függvényeket felhasználva ellenőrzi, hogy az adatbázisban szerepel-e az adott adat. A `$user` változó létrehozásához a `User::factory()->create()` metódust használja, amely egy új felhasználó modellt hoz létre majd elmenti az adatbázisba.

```

public function test_it_stores_new_users(){
    $response=$this->post('/register',[
        'name'=>'Pinter3',
        'email'=>'pinter3@gmail.com',
        'password'=>'pinter1234567',
        'password_confirmation'=>'pinter1234567'
    ]);
    $response->assertRedirect('/home');
}

```

### 52. ábra Teszt - Felhasználó mentése

A teszt célja, hogy ellenőrizze, hogy a felhasználók sikeresen létrejönnek-e. Első körben a `register` útvonalhoz `POST` kérést küld, majd ezután ellenőrzi, hogy a válasz átirányítja-e a felhasználót a `home` oldalra. Azt is ellenőrzi, hogy a felhasználók a megfelelő adatokat adták-e meg, például helyes e-mail formátumot, a jelszavak megegyeznek-e stb. Az adatok ellenőrzése után létre kell jönnie az új felhasználónak, és a `home`-ra való átirányítás jelzi, hogy az új felhasználó sikeresen létrejött.

Itt láthatjuk, hogy a tesztek sikeresen lefutottak, valamint a seeder tesztet kockázatosnak ítélte meg.

**Tests: 1 risky, 10 passed**  
**Time: 0.74s**

---

### 53. ábra Tesztek állapota

A fenti tesztek a Laravel által nyújtott alap tesztelési funkciókat mutatják , amelyek igen hasznosak lehetnek, ha bizonyos működéseket szeretnénk tesztelni, mint például a modellek létrehozása, az egyedi azonosítóknak az ellenőrzését, a sikeres regisztrációt és bejelentkezést, valamint az adatbázis műveleteket.

A tesztek által garantálni lehet a rendszer megfelelő működését, az adatvédelmi és biztonsági szabályok betartását, és segítenek felderíteni a hibákat, amelyek akkor is előfordulhatnak, ha a rendszer ellenőrizve volt. Nagyon fontos, hogy a hibákat időben azonosítsuk és kijavítsuk, hiszen komoly károkat okozhatnak.

A kapott eredmények alapján javítani lehet a rendszeren.

## 6. A rendszer felhasználása

Az okos öntöző és világító rendszer felhasználható otthoni környezetben, kinti és benti növények gondozásához egyaránt. Ideális azok számára, akik szeretnék biztosítani növényeik optimális körülményeit és gondozását anélkül, hogy folyamatosan jelen lennének. Alkalmas lehet azon hobbi kertészek és növénykedvelők számára is, akik hatékonyabb módszert keresnek a növényeik gondozásához és szeretnék minimalizálni a manuális beavatkozás szükségességét. A felhasználói felület segítségével könnyen kezelhető a rendszer, így akár kezdőként is könnyedén használható, nem igényel semmilyen komoly tudást ahhoz, hogy a növények optimális állapotát garantálni tudjuk.

## 7. Következtetések

### 7.1. Megvalósítások

Noha számos továbbfejlesztési lehetőség maradt hátra, a rendszer legfontosabb tulajdonságait és funkcióit sikeresen megvalósítottam.

Ezek a következők :

#### **A környezeti paraméterek mérése szenzorok segítségével**

A rendszer a DS18B20 hőmérséklet szenzor és kapacitív nedvesség mérő szenzor segítségével nagy pontossággal méri a környezeti paramétereket.

#### **Felhasználói felület létrehozása**

---

Fontos volt, hogy a felhasználó nyomon tudja követni a növény állapotát és kezelni tudja a beállításokat.

### **Kapcsolat a mikrovezérlő - adatbázis - felhasználói felület között**

A mikrovezérlő kommunikál az adatbázissal, ahol eltároljuk az adatokat és minden változás ide kerül elmentve. A felhasználói felületről történő vezérlés adatai is az adatbázisba íródnak be, innen pedig az információ továbbítódik a mikrovezérlő felé. Az adatbázis "kapcsolatot" biztosít a mikrovezérlő és a felhasználói felület között.

### **Öntözés és világítás**

A növények optimális növekedését elősegítve a rendszer a környezeti paramétereket figyelembe véve öntözi és fvilágítja a növényeket.

### **Profilok bevezetése**

A növény profilok bevezetésével minden növény a saját igényeinek megfelelő víz és fény mennyiséget kap.

### **Automata és manuális mód**

A felhasználó manuálisan is kezelni tudja az öntözés és világítás funkciókat illetve automata módon a rendszer saját maga adja ki a parancsokat a bevezetett küszöbértékek alapján.

## **7.2. Továbbfejlesztési lehetőségek**

### **Több növény integrált követése és gondozása:**

Ez a továbbfejlesztési lehetőség azt foglalja magába, hogy a rendszer több növény állapotát és gondozását tudja nyomon követni, több növényt illetve virágtartót lehet a rendszerhez csatlakoztatni.

### **WiFi csatlakozás felhasználóbarát módon:**

Jelenleg a hálózathoz való csatlakozáshoz szükséges adatok hard codeolva vannak a mikrovezérlőbe, ezért sokkal hatékonyabbá tenné a rendszert az a megvalósítás, hogy ezeket az



---

adatokat felhasználói felületről adjuk meg. A mikrovezérlő létrehoz egy acces pointot, amire rácsatlakozunk például telefonról és a router nevét illetve jelszavát elküldjük a mikrovezérlőnek.

### Vízszint szabályozás:

A vízszint szabályozás továbbfejlesztése elősegítheti a vízfogyasztás hatékonyabb kezelését és a vízszint stabilizálását a kívánt értéken. Ezáltal biztosítható a vízellátás megbízhatósága.

A rendszer figyelné a tényleges vízfogyasztást a tartály űrtartalmát figyelembe véve és amennyiben túl sok illetve túl kevés víz lenne a tartályban, figyelmeztető üzenetet küldene.

## 7.3. Költségvetés

ALKATRÉSZ	ÁR	PÉNZNEM	LINK
ESP32	51,17	RON	<a href="#">link</a>
DS18B20	14,13	RON	<a href="#">link</a>
Kapacitív nedvességmérő szenzor	19,07	RON	<a href="#">link</a>
YF-S401 hozammérő	24,00	RON	<a href="#">link</a>
Pompa	11,38	RON	<a href="#">link</a>
Relé modul	20,50	RON	<a href="#">link</a>
Ledek		RON	
Nyák	2,60	RON	<a href="#">link</a>
Jumper wire anya-anya	24,95	RON	<a href="#">link</a>
Jumper wire anya-apa	24,95	RON	<a href="#">link</a>
Jumper wire apa-apa	19,99	RON	<a href="#">link</a>
Filament-1kg	109	RON	<a href="#">link</a>
Cső 1m	12,72	RON	<a href="#">link</a>

<b>VÉGÖSSZEG</b>	<b>334,46</b>	<b>RON</b>	
------------------	---------------	------------	--

#### 5. táblázat Költségvetés

A fenti táblázatban megadott árak és a végösszeg is változhat. Ide kell számolni a nyomtatási , szállítási költségeket is illetve szükség van speciális felszerelésre, amivel a kábelezést és forrasztást végezhetjük.

Ezek ellenére is megéri házilag elkészíteni egy ilyen rendszert, hiszen sokkal olcsóbb a piaci árakhoz képest és több funkcióval is rendelkezik.

## 8. Irodalomjegyzék

- [1] Mikroprocesszorok, mikroszámítógép elemek / Madarász László, Kecskeméti Főiskola Műszaki Főiskolai Kar, 1998, 004.3/MAD.
- [2] A. Tanenbaum: Számítógép architektúrák. (Arhitectura calculatoarelor) Bp., Panem Könyvkiadó, 2004.
- [3] Gamma, Erich, Helm, Richard, Johnson, Ralph, Vlissides, John, Programtervezési minták, Kiskapu 2004.
- [4] Vedat Ozan Oner, Developing IoT Projects with ESP32: Automate your home or business with inexpensive Wi-Fi devices
- [5] "Internet of Things: Principles and Paradigms" - Rajkumar Buyya, Amir Vahid Dastjerdi
- [6] Internet of Things Projects with ESP32 : Build Exciting and Powerful IoT Projects Using the All-new Espressif ESP32 - Agus Kurniawan
- [7] IoT and Edge Computing for Architects: Implementing edge and IoT systems from sensors to clouds with communication systems, analytics, and security - Perry Lea
- [8] IoT Security: Advances in Authentication - Madhusanka Liyanage
- [9] How the Internet Works & the Web Development Process - Micheal Full
- [10] Http Developer's Handbook - Chris Shiflett
- [11] Learning HTTP/2 - Stephen Ludin & Javier Garza
- [12] <https://laravel.com/docs/10.x>
- [13] [How to start testing](#)
- [14] <https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf>

---

## 9. Függelék

A sensor és details JSON objektumok létezésének ellenőrzése:

```
if(myObject.hasOwnProperty("sensors") && myObject.hasOwnProperty("details")){
    JSONVar sensors = myObject["sensors"];
    JSONVar details = myObject["details"];

    if (JSON.typeof(sensors) == "undefined") {
        Serial.println("SENSORS: Hiba a JSON objektumban!");
        return;
    }
    if (JSON.typeof(details) == "undefined") {
        Serial.println("DETAILS: Hiba a JSON objektumban!");
        return;
    }
}
```

54. ábra JSON objektumok

Videó a rendszer működéséről :

[https://drive.google.com/file/d/1SDVSikg\\_JSN\\_FHxVjC\\_iE18TOM5F4cON/view](https://drive.google.com/file/d/1SDVSikg_JSN_FHxVjC_iE18TOM5F4cON/view)

---

UNIVERSITATEA SAPIENTIA DIN CLUJ-NAPOCA  
FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE, TÎRGU-MUREȘ  
SPECIALIZAREA CALCULATOARE

Vizat decan  
Conf. dr. ing. Domokos József

Vizat director departament  
Ș.l. dr. ing. Szabó László Zsolt