



Programozható elektronikák alkalmazásai

Készítette

Bagoly Gábor

Programtervező informatikus

Témavezető

Dr. Geda Gábor

Egyetemi docens

EGER, 2023

Tartalomjegyzék

Bevezetés	4
1. Piacon lévő jelentősebb okos otthon rendszerek	5
1.1. Nagyobb cégek által létrehozott ökoszisztémák	5
1.1.1. Mik is a virtuális asszisztensek?	5
1.1.2. Amazon rendszere - Alexa Smart Home	6
1.1.3. Google rendszere - Google Home és Google Nest	7
1.1.4. Xiaomi rendszere - Mi Home	8
1.2. Nyílt forráskódú rendszerek	9
1.2.1. Mit is az, hogy open source?	9
1.2.2. Home Assistant	9
1.2.3. OpenHAB	10
2. Alkalmazott eszközök	11
2.1. Hardver	11
2.1.1. Mi is az hogy IoT?	11
2.1.2. Raspberry Pi 4B	11
2.1.3. NodeMCU - ESP-WROOM-32	12
2.1.4. ESP32-CAM - Wi-Fi-s kamera modul	13
2.1.5. DHT22 - hőmérséklet és páratartalom szenzor	14
2.1.6. Mi is az RFID?	14
2.1.7. RFID-RC522 - RFID olvasó	15
2.1.8. Szilárdtest relé	15
2.2. Szoftverek és Programozási nyelvek	16
2.2.1. C++ és Arduino IDE	16
2.2.2. HTML és CSS	17
2.2.3. PHP	17
2.2.4. JavaScript	17
2.2.5. jQuery	18
2.2.6. Chart.js	18
2.2.7. Tailwind CSS	18
2.2.8. dbDiagram.io	19

2.2.9. Font Awesome	19
2.2.10. MySQL	20
2.2.11. PlantUML	20
2.2.12. Fritzing	20
2.2.13. Laravel	21
3. A web alkalmazás felépítése és működése	22
3.1. Raspberry PI 4B alkalmazása, mint Wi-Fi, és hub	22
3.2. Adatbázis felépítése	23
3.3. Laravel működése	24
3.3.1. Modellek és migrációk	24
3.3.2. Route-olás	26
3.3.3. Controller	27
3.3.4. Blade	28
3.4. Kezelő felület bemutatása és működése	29
3.4.1. Főoldal	29
3.4.2. Beállítások	30
3.4.3. RFID beállítások	33
3.4.4. RFID használati táblázat	33
3.4.5. Hőmérsékleti és páratartalom előzmények - ChartJS	33
3.5. Eszközök kommunikációja a webszerverrel	33
3.5.1. Hőmérséklet és Páratartalom szenzor	33
3.5.2. Eszköz kapcsoló	33
3.5.3. Kamera	33
3.5.4. RFID kártya olvasó	33
4. Tesztelések	34
4.1. Tesztelések módjai és fontossága	34
4.1.1. Cypress automatizált tesztelések	34
4.1.2. Manuális tesztelések	34
5. Rendszer telepítése	36
Összegzés	37
Irodalomjegyzék	38

Bevezetés

Mai világunkban az okos otthonok és az okos eszközök rendkívül népszerűek és elterjedtek lettek. Általában, hogy ha megkérdezzük valakit ezzel a témával kapcsolatban, akkor nagy valószínűséggel azt tudják mondani, hogy rendelkeznek legalább egy okos otthonban alkalmazható eszközzel. Ezek az eszközök lehetővé tehetik a kényelmesebb és hatékonyabb életmódot.

De mi is tesz egy eszközt okossá? Feltételezhetjük azt, hogy ha valamelyik eszköz internetre kapcsolódik, esetleg távolról beállíthatjuk, vagy automatizálhatjuk előre meghatározott dolgokra, akkor azt az eszközt „okosnak” tudjuk mondani.

Hogy ha valaki már rendelkezik több ilyen eszközzel, akkor bizonyára találkozott már azzal a problémával, hogy egy bizonyos ökoszisztémában használatos eszköz nem feltétlenül tud működni egy másikban. Minderre próbáltam egy olyan megoldást kitálatni, ami abból a szempontból közelíti meg mindezt, hogy még egy 'nem okos' eszközt (például egy izzót) integrálok bele úgy a rendszerbe, hogy azt egyszerűen tudjunk kezelni bárhonnan az otthonunkból, ehhez társulva különböző programozható elektronikák. Mindehhez egy olyan rendszert hoztam létre, amelynek a háttérben lévő folyamatok lebonyolítását egy webes alkalmazás végzi el, és az általunk ismert legtöbb eszközön használható, amin internetezni is tudunk: legyen az számítógép, Androidos, vagy iOS-es telefon, tablet, vagy akár okos óra is.

Azért került erre a témára a választásom, mivel számomra felettébb érdekes az, hogy egy ilyen okos otthonban az eszközök hogyan is kommunikálnak, és szeretnék ebbe egy belátást nyerni, hogy hogyan is épül össze mindez.

Céлом az lenne ezzel, hogy belelássak egy ilyen rendszer működésébe, különböző programozható eszközök alkalmazását jobban megismerjem, és, hogy egy olyan általános kezelőfelületet tudjak létrehozni, amit könnyen tud a felhasználó alkalmazni.

1. fejezet

Piacon lévő jelentősebb okos otthon rendszerek

1.1. Nagyobb cégek által létrehozott ökoszisztémák

Amikor arra jut a sor, hogy okos otthont szeretnénk összeállítani, akkor ahhoz egy széles palettából tudunk választani eszközöket, legyen az biztonsági kamera, ajtózár vagy akár háztartási eszközök, mint például egy mosógép vagy robot porszívó. Amikor az okos otthon rendszer kiválasztására kerül sor, fontos figyelembe venni az ehhez alkalmazandó eszközök kiválasztását is.

Van néhány olyan eszköz, – például az okos izzók – amelyek szerencsére több okos otthon rendszerben is könnyen alkalmazhatóak. Ezek az eszközök általában ipari szabványokat használnak, mint például a Zigbee¹, ami lehetővé teszi számukra, hogy működjenek a különböző okos otthon rendszerekkel.

Azonban vannak olyan eszközök is, amelyek csak egyetlen rendszerrel használhatóak. Ezek az eszközök saját szabványokat alkalmaznak, amelyek nem kompatibilisek más rendszerekkel. Vegyük például azt, hogy ha egy olyan okos eszközt veszünk, ami csak egy meghatározott okos otthon rendszerrel működik együtt, akkor az eszközt később nem tudjuk használni egy másik rendszerben.

Mindezek alapján amikor okos eszközt vásárlunk, akkor nagyon oda kell figyelni, hogy kompatibilis-e a választott okos otthon rendszerünkkel, amit általában fel szoktak tüntetni az eszközök leírásában, vagy, hogy ha utána keresünk mindennek.

1.1.1. Mik is a virtuális asszisztensek?

A virtuális asszisztensek olyan szoftveres programok, amik különböző feladatokat tudnak elvégezni felhasználó kérésére. Ezek persze mindezeknek korlátjainak megfelelően.

¹ Zigbee-t használ például az Amazon Echo, a Google, az IKEA okos otthon termékei, és a Philips Hue

Úgy lettek megalkotva, hogy a felhasználó hang vagy esetleg chat alapon tudjanak egyszerű kérdéseken keresztül kommunikálni vagy utasításokat adni. A virtuális asszisztensek olyan feladatokban tudnak segíteni, mint emlékeztetők létrehozása, üzenetek küldése, hívások indítása, interneten való keresés, időjárás előrejelzések felolvasása, okos otthoni eszközök irányítása, és egyéb más dolgok.

Számos cég hozott már létre magának virtuális asszisztent, amik közül a legismertebbek lehetnek a Google által létrehozott Google Asszisztens, Apple-nek Siri, Amazon-nak Alexa, Samsung Bixby-je, és Microsoft Cortana-ja.

Hogy ha például egy újabb Samsung telefonja van az embernek, akkor azon két virtuális asszisztens is jelen van (Google Assistant és Bixby), de letölthető akár mellé harmadiknak az Amazon Alexa is.

1.1.2. Amazon rendszere - Alexa Smart Home

2014-ben lépett be a piacra az Amazon – az akkor még újdonságnak számító – okos hangszórójukkal, az Amazon Echo-val. Ekkor még leginkább csak annyira volt képes, hogy a felhasználó zenét tudja irányítani hang utasításokkal. Ez az eszköz úgy működik, hogy ebbe bele van integrálva az Amazon sajátos virtuális asszisztense, amit Alexának hívnak. Ugye mint kezdetleges szoftver, neki sem voltak a képességei túl szerteágazóak. Leginkább arra tudta az ember használni, hogy egyszerű kérdéseket tegyen fel az ember, és zenét tudjon elindítani, leállítani átugrani.

Nem is annyira később, amikor elkezdett egyre jobban fejlődni Alexa, úgy egyre több mindenre kezdhette el használni az ember: termosztátok beállítása, izzók ki- és bekapcsolására is már lehetett használni. Miután az Amazon egyre többet fektetett bele a rendszerükbe felettébb szerteágazó lett annak a használata az otthonokban. Még arra is volt lehetőség már ekkor, hogy akár bevásárló listát készítsen, és azokat meg is tudja rendelni a felhasználó az Amazonról, mindezt Alexa használatával. A cég 2017 május 23-án jelentette be, hogy a *Smart Home Skill API*-jukba² innentől kezdve megadható, hogy milyen eszközöket is csatlakoztatunk a rendszerükbe, ami kitárta a lehetőségeket az otthoni okos készülékek automatizációjára. Mindez vezetett az okos otthon piac és a virtuális asszisztensek szerepének növekedéséhez.

Eszközök leginkább hangvezérléssel irányíthatóak, miután követtük az általuk biztosított használati útmutatót. Néhány esetben az Amazon Alexa alkalmazás elegendő lehet, de ha olyan eszközt szeretnénk használni, amelyhez saját alkalmazás tartozik, akkor azt is le kell töltenünk.

Az Amazon által szolgáltatott okos otthon rendszer 2023-ra olyan népszerűségi szintre jutott, hogy az Amerikai Egyesült Államokban az ilyen hang vezérelt hangszórók 68.2%-a az Amazon Echo.³

² A blog poszt amiben bejelentették a Smart Home Skill API bővítését[1]

³ Statisztikai adatokat az Earthweb: „How Many People Use Alexa in 2023? (U.S. Amazon

Mindezzel napjainkban több ezer eszköz használható már ezen a rendszeren belül: legyen az biztonsági rendszer, háztartási gépek, vagy esetleg szórakoztató rendszerek. Érthető is, hogy sokan miért is szeretik ezt a rendszert használni.

Azonban ennek a rendszernek is megvannak azok a hátulütői, mint például az, hogy nem tudunk felhasználni benne olyan eszközöket, amik nem támogatottak az Amazon által. Érdekes arra odafigyelni, hogy amikor egy okos eszközt veszünk, hogy arra fel-e van tüntetve, hogy kompatibilis az Amazon rendszerével. Másik ilyen negatív tényező lehet számunkra az is, hogy az Amazon Alexa nem használható magyar nyelven még.

1.1.3. Google rendszere - Google Home és Google Nest

Az Amazon Echo sikere után a Google is részesülni akart az okos otthon piacának sikereivel.

Az akkor még Nest Labs által készült termékek olyanok voltak mint az öntanuló termosztát – amit 2011-ben hoztak létre „Nest Learning Thermostat”⁴ néven, ami Wi-Fi-re kapcsolható volt, és szenzorok segítségével alkalmazkodhatott a beltéri hőmérsékleti körülményekre. Ezt követte a következő termékük, a füst és szén-monoxid érzékelő, aminek a neve „Nest Protect” volt. 2014-ben felvásárolták a Dropcam nevű céget ami biztonsági kamerákat készítettek, amik után a Nest Labs ezt követő terméke a Nest Cam volt 2015-ben.

A Nest Labs egyre jobban látszódó sikerének köszönhetően a Google felvásárolta 2014 januárjában. 2018-ig még önállóan működött a Nest, ami után beolvasztották a Google otthoni termékcsaládba, ezzel létre hozva a Google Nest termékcsaládot, ami számos termékekkel rendelkezik mostanára: termosztát, ajtócsengő, ajtózár, biztonsági kamera, virtuális asszisztenssel integrált érintőképernyős központi egység.⁵

Az Amazon Echo-hoz hasonló első terméke a Google-nek a Google Home volt, amit 2016 októberében jelentettek be. Ez a cég sajátos virtuális asszisztensével a Google Assistant-tel volt felszerelve, és ugyan úgy lehetett neki utasításokat adni, és kérdéseket feltenni. Azóta már több otthonon belül alkalmazható eszköz elérhető egyenesen a Google Store-ból.⁵ Vagy egyéb támogatott harmadik felek által készített ilyen eszközökkel.

Az egyik legnagyobb előnye a Google Home rendszernek az integrációja más Google szolgáltatásokkal, mint például a Google Térkép, a Google Naptár és a Google Fotók. Ez azt jelenti, hogy a felhasználók kéz nélkül is hozzáférhetnek személyes információikhoz és ütemtervükhöz, szimplán csak a Google Asszisztensnek feltéve a kérdést.

A Google Home rendszer másik erőssége az, hogy képes felismerni a különböző

Statistics)” cikkéből [2]

⁴ Magyarul: Nest Tanuló Termosztát

⁵ További jelenleg elérhető Google Nest családba tartozó termékek: https://store.google.com/gb/category/connected_home oldalon megtalálható

hangokat, amely lehetővé teszi a személyre szabott válaszokat és információkat minden háztartási tag számára. Ez különösen hasznos lehet több személyes háztartásokban, ahol több ember is használja a rendszert.⁶

Mindezek által a Google Home rendszer erős versenytárs az okos otthon piacon.

Viszont a Google Home-nak is meg vannak azok a hátrányai mint az Amazon rendszerének. Sajnos inkább csak akkor tudjuk kihasználni ennek a rendszernek az előnyét, ha angolul vagy más támogatott nyelven beszélünk vele, amibe még nem tartozik bele a magyar.

1.1.4. Xiaomi rendszere - Mi Home

A Xiaomi 2015 júniusában dobta piacra első okos otthon termékcsomagját, a „Smart Home Kit”-et, amely mozgásérzékelőt, lámpát és kapacitív kapcsolót tartalmazott. Ezeket az eszközöket egy alkalmazás segítségével lehetett vezérelni, ami lehetővé tette a felhasználók számára, hogy különböző utasításokat állítsanak be, például hogy a mozgásérzékelő észlelésekor a lámpa felkapcsoljon, vagy értesítést küldjön a felhasználónak, illetve a csomag támogatta a Xiaomi biztonsági kameráját is. [3]

Ma a Xiaomi okos otthon termékpalletája rendkívül sokrétű, a forrólevegős sütőtől, a robotporszívókon, és a szobamérlegeken át a hőmérséklet- és páratartalom-szenzorokig. Az összes termékük integrálható a Mi Home alkalmazásba, és a felhasználók harmadik féltől származó eszközöket is csatlakoztathatnak hozzájuk.

Nagy előnye a Xiaomi okos otthon termékeknek az, hogy viszonylag olcsóbb a konkurens termékektől, és fel is használható például a Google Home, és az Amazon Alexa Smart Home rendszeren belül is, és használata felettébb felhasználó barát.

Kisebb hátránya lehet neki az, hogy ezeket az eszközöket nem lehet asztali alkalmazáson keresztül irányítani, csak is az Androidos és iOS-es alkalmazáson keresztül, vagy esetleg a Google Home vagy Alexa Smart Home-on belül. Amazon Alexával a párosítás mostanában sajnos nehezebb, mivel valamilyen oknál fogva nagy százalékban nem tud csatlakozni valami hibánál fogva. Még olyan is megesik, hogy nem minden eszköz érhető el például a Google Home felületen, ilyen lehet példának a biztonsági kamerájuk, ami csak a saját alkalmazásukon érhető el. És mint az előző kettő rendszernél, ez sem használható még magyar nyelven.

⁶ Erre ugyan úgy betanítható az Amazon Alexa is.

1.2. Nyílt forráskódú rendszerek

1.2.1. Mit is az, hogy open source?

Open source, azaz nyílt forráskódú szoftver az olyan, aminek a forráskódját szabadon lehet vizsgálni, módosítani, és akár ki is egészíteni. A kód az a része egy szoftvernek amit a legtöbb felhasználó bizonyára soha nem fog látni. Ez az a kód, amit a programozók változtathatnak hogy megváltozzon a program vagy alkalmazás működése. Azok a programozók, akik hozzáférhetnek egy szoftver forráskódjához, azokat feljeszthetik, és javíthatják azzal hogy például új funkciót adnak hozzá, vagy kijavítanak egy olyan részt, ami nem minden esetben működik helyesen.⁷

Azonban azt, hogy egy szoftver nyílt forráskódú, nem feltétlenül jelenti azt, hogy az adott szoftver ingyenes használatban áll. Nyílt forráskódot fejlesztő programozók kérhetnek pénzt azért a nyílt forrású szoftverért, hogy ha ők alkották meg, vagy hozzájárulásukkal készült el.

Bizonyos esetekben azonban, mivel a nyílt forráskódú licenc megkövetelheti a program forráskódjának kiadását amikor szoftvert adnak el másoknak, egyes programozók úgy találják, hogy jövedelmezőbb pénzt kérni a felhasználóktól a szoftverszolgáltatásokért és - támogatásért (nem feltétlenül magáért a szoftverért). Így a szoftvereik ingyenesek maradnak, és pénzt keresnek azzal, hogy másoknak segítenek telepíteni, használni és hibaelhárítást végezni.⁷

1.2.2. Home Assistant

A Home Assistant 2013 novemberére került publikálásra *GitHub.com*-on PAULUS SCHOUTSEN által. Ez ekkor még egy Python programozási nyelven megírt alkalmazás volt. Ekkor még ez egy egyszerű program volt, ami napnyugtakor felkapcsolta a lámpát. Azóta a szoftver elég érett lett, és körülbelül 20 aktív hozzájáruló dolgozik a projekten, ennek köszönhetően kéthetente érkezik frissítés a rendszerre.⁸

Ez a szoftver bármely olyan rendszeren működik, ami a Python 3 programozási nyelvet tudja futtatni. Biztosít mobil és számítógép alkalmazást is, mellyel több ezer támogatott eszközt tudunk irányítani. Annyi a különbség ez a rendszer és a nagyobb cégek által biztosított között, hogy ez teljesen lokálisan fut, és hogy ha esetleg internet kimaradás lenne, még akkor is tudjuk ugyan úgy irányítani eszközeinket. A rendszer által számos ismert okos eszköz támogatott, mint például a Philips Hue, IKEA TRÅDFRI, és akár az Amazon Echo, Google Home, és a Xiaomi Mi Home által támogatott eszközök, és még a virtuális asszisztensük is.⁹ Mindez annak köszönhető, mert

⁷ A nyílt forráskódú szoftverről az irodalomjegyzék [4]. részén található

⁸ A történet bővebben az irodalomjegyzék [5]. részén megtalálható.

⁹ A támogatott rendszerek a weboldalukon elérhető, ami az irodalomjegyzék [6]. részén megtalálható

Z-Wave és a Zigbee protokoll is támogatott a rendszer által, és mivel nyílt forráskódú a szoftver, ezért bármikor lehet az hogy valamelyik felhasználó vagy esetleg fejlesztő hozzáadja bizonyos termékeket kiegészítésként.

A rendszerrel háztartásunk energiafelhasználását is nyilván lehet tartani. Szabadon testre szabható a felület kinézete a felhasználók ízlése szerint. Szintén pozitívum lehet, hogy az szoftver támogatja a magyar nyelvet is.

Ámbár ez a rendszer azoknak ajánlott jobban, akik informatika terén ismertek.

1.2.3. OpenHAB

OpenHAB, azaz Open Home Automation Bus¹⁰ fejlesztése 2010-ben kezdődött, és Java programozási nyelvben lett megírva. 2013-ban került elérhetővé az első stabil verziója a szoftvernek.

Az OpenHAB legelső körben azon a személyek számára ajánlott, akik jól ismertek informatika, és robotika terén. Ez azért szükséges, mert nekünk kell összeállítanunk, hogy mit szeretnénk elérni az okos otthon rendszerünkben. Az alkalmazás rendkívül rugalmas és testre szabható, és nagy a támogatottsága. Ez az egyik legelterjedtebb nyílt forráskódú okos otthon rendszer, és folyamatos fejlesztés alatt áll egy nonprofit szervezet által, aminek a fejlesztésébe bárki beállhat.

Ez a rendszer képes integrálni a piac különböző eszközeit, mint ahogyan azt a Home Assistant rendszernél is említettem. Szintén egy fontos tényezője, hogy lokálisan működik, ezért egy internet kimaradásnál az okos otthonunk ugyan úgy működni fog.

A szoftvernek hivatalos oldalon megtalálható a teljes dokumentációja, hogy mit hogyan kell összeállítani és használni, ezért lehet kedvelt olyan személyek számára, akik szeretik a dolgokat maguknak összeállítani. A rendszer nagy támogatást és közösséget élvez, így ha felmerülne bármilyen kérdés, biztosan választ kapunk rá.

Az ezzel létrehozott otthonunkat a legtöbb felületen el tudjuk érni, legyen az MacOS, Windows, Linux, Android vagy iOS.¹¹

¹⁰ Magyarul: Nyílt Otthon Automatizációs Busz

¹¹ Az OpenHAB hivatalos oldala megtalálható a az irodalomjegyzék [7]. részén, ahol olvasható a dokumentációjuk, a céljaik, és egyéb blogjaik.

2. fejezet

Alkalmazott eszközök

Ebben a fejezetben azt fogom taglalni, hogy milyen eszközöket is használtam az okos otthon rendszerem létrehozása során. Legyenek ezek hardveres vagy szoftveres komponensek.

Mindezek közben fogok csatolni kötési rajzokat, amik arra tesznek betekintést, hogy hogyan is kell összekötni ezeket az eszközöket, hogy a rendszerben működjenek.

Szoftveres komponensek közé tartoznak azok a szoftverek, amik a projekt létrehozása közben fel voltak használva: legyen az keretrendszer, programozási nyelv, vagy stílus megírásra használt külső komponens.

2.1. Hardver

2.1.1. Mi is az hogy IoT?

IoT¹-k azok olyan eszközök, melyek Wi-Fi hálózaton keresztül párosíthatóak, és irányíthatóak. Ilyenek lehetnek a háztartásunkban levő olyan eszközök, melyeket távolról is tudunk irányítani interneten keresztül. Legyen ez például a Xiaomi forrólevegős sütője, vagy egy okos izzó.

Ebbe a kategóriába tartoznak a mikrokontrollerek is, amik mini programozható elektronikák, az alkalmazott Raspberry Pi 4B is.

2.1.2. Raspberry Pi 4B

A projektnek szíve-lelke egy bankkártya méretű mini számítógép, ami szolgáltatja a vezetékek nélküli internetet az otthoni eszközök számára, ezzel megvalósítva a később említett programozható elektronikáknak is a kommunikációt.

A Raspberry-n dolgozódnak fel az adatok, és ez általa szolgáltatott a webalkalmazás is. Felettébb sokoldalú a használata, és előszeretettel van használva okos otthon

¹ (Internet of Things) - magyarul: Internet dolgai

projektek megvalósításában is, például hogy ha Home Assistant-el, vagy OpenHAB-bal szeretnénk azt megvalósítani.

Eben Upton 2012-ben hozta létre az első ilyen kis számítógépet, a Raspberry Pi 1 Model B-t. Raspberry magyarul azt jelenti, hogy málna mivel, ebben az időben többen is hoztak létre elektronikai termékeket gyümölcs nevekkal felruházza: legyen az Apple (alma), Acorn (makk), Apricot (sárgabarack).

Elsődleges célja az volt Eben Upton-nak hogy olcsó és könnyen elérhető számítógépeket juttassanak el fiatalok számára.

A Raspberry számítógépeknek elsődlegesen két fajtája volt: az A Model, ami olcsóbb, és a B Model, ami gyorsabb volt.²

Rendkívül sokoldalú ez a kis számítógép, és ezért is ennyire előszeretett a használata a programozók körében.

Az általam használt Raspberry Pi 4B specifikációi:

- **RAM:** 2GB
- **Tárhely:** 16GB-os microSD memória kártya
- **Operációs rendszer:** Raspbian, ami egy Debian alapú Linux operációs rendszer

2.1.3. NodeMCU - ESP-WROOM-32

A NodeMCU az Espressif Systems ESP8266-12E Wi-Fi System-On-Chip³ rendszeren alapul, amely egy nyílt forráskódú, Lua-alapú firmware-rel⁴ van felszerelve. tökéletes az IoT-alkalmazásokhoz és más olyan helyzetekhez, ahol vezeték nélküli kapcsolatra van szükség. Ez a chip nagyon sok hasonlóságot mutat az Arduino-val – mindkettő mikrokontrollerrel felszerelt prototípus lap, amely az Arduino IDE segítségével programozható.[12]

Az ESP-WROOM-32 mikrovezérlő egy erős és sokoldalú eszköz, amely remek választás lehet egy okos otthon projekt számára. Alacsony energiaigényű processzora és integrált Wi-Fi és Bluetooth képességei alkalmasak szenzorok és eszközök vezérlésére és kommunikációjára.

Az ESP-WROOM-32S mikrovezérlő adatlapja részletes technikai információkat tartalmaz a funkcióiról és képességeiről.[10]

Összességében az ESP-WROOM-32S mikrovezérlő egy erős és rugalmas eszköz az okos otthoni rendszerek építéséhez, és az adatlap széleskörű technikai információkat nyújt a fejlesztők számára, akik ezzel az eszközzel szeretnék tervezni és programozni.

² Az irodalomjegyzék [8]. részén megtalálható a bővebb története a Raspberry Pi-nak

³ Rendszer a chipen

⁴ Magyarul: Fix tárban tárolt információ, rendszer

Éppen ezért is lehet elsődleges választás robotika oktatás közben egy NodeMCU használata, én is például az egyetemi tanulmányaim során a Robotikai alapjai nevű tárgyon találkoztam ezzel az eszközzel először.

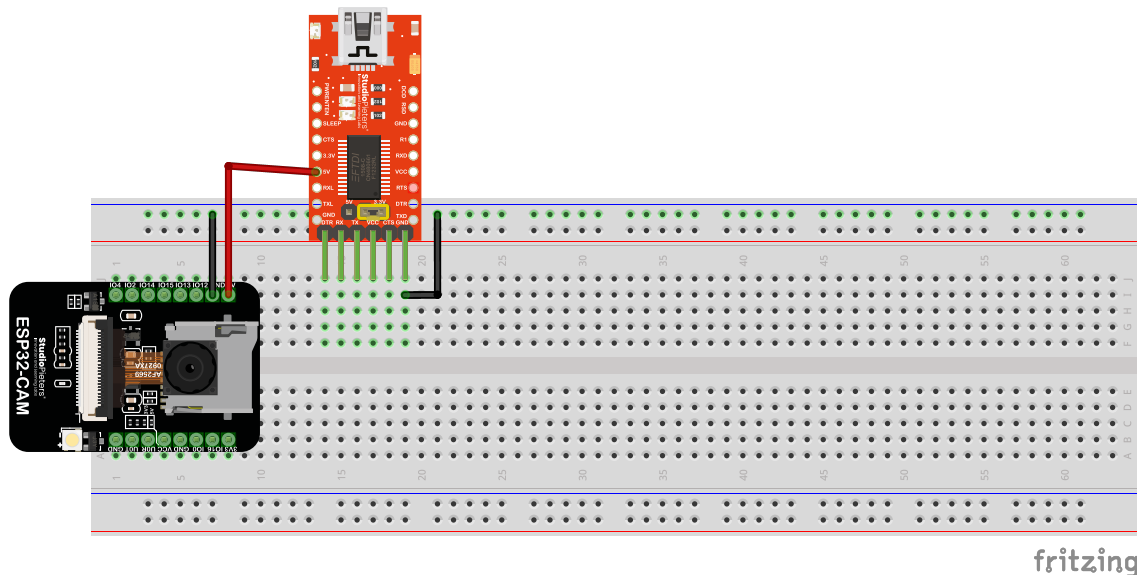
Rendszeremben ezek az eszközök úgy vannak használva, hogy – kihasználva a Wi-Fi-s felszereltségüket – rákapcsolódnak a Raspberry Pi-ra, és ezen keresztül küldenek olyan információkat, mint például az adott szobában lévő hőmérséklet és páratartalom szenzor által szolgáltatott információk, és a szobákban lévő eszközök ki-és bekapcsolása.

2.1.4. ESP32-CAM - Wi-Fi-s kamera modul

Mint ahogyan a 2.1.3. szakaszban taglalt NodeMCU, az ESP32-CAM is az IoT és ESP32-es eszközök családjába tartozik, ami egy olyan programozható elektronika ami Wi-Fi-re képes csatlakoztatni.

Ebben a rendszerben olyan módon van alkalmazva, mint egy „biztonsági kamera”, és a főoldalon szolgálat élő közvetítést. Szintén oly módon csatlakozik rá a Raspberry Pi-ra, mint a többi mikrokontroller ebben a rendszerben.⁵

Lehetőség adott ezzel az eszközzel az is, hogy akár memóriakártyára mentsünk képeket, amit a kamera felvesz, és van egy saját beépített LED lámpája is. Azonban ezeket a funkciókat a rendszer aktuális verziójában nem alkalmazom, csak és kizárólag élő közvetítésre.



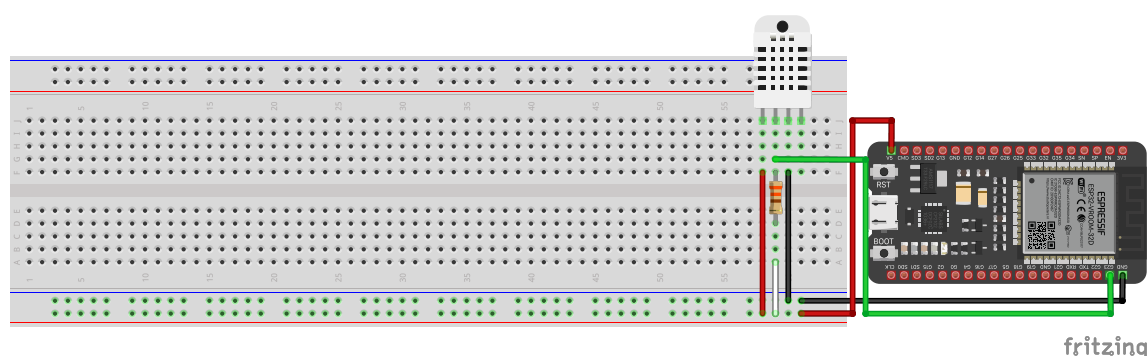
2.1. ábra. Az ESP32-CAM bekötése

⁵ A 3.5 részen ennek a működése részletezve lesz.

2.1.5. DHT22 - hőmérséklet és páratartalom szenzor

Az DHT22 érzékelő egy olyan eszköz, amely méri a levegő hőmérsékletét és a relatív páratartalmát. Az összegyűjtött adatokat feldolgozza és továbbítja. A DHT22 kis mérete, alacsony energiafogyasztása miatt széles körben használható különböző alkalmazásokban.[13]

Ebben a rendszerben úgy van alkalmazva, hogy egy ESP-WROOM-32-höz van kapcsolva, és ez 10 másodpercenként elküldi az adatokat a webalkalmazásnak, mely eltárolja az adott pillanatban mért adatokat az alkalmazott szobában, és jeleníti meg a főoldalon, mely alapján meg lehet tekinteni az elmúlt 24 órai adatokat egy grafikonon is.⁵



2.2. ábra. Az ESP-WROOM-32 bekötése DHT22 hőmérséklet és páratartalom szenzorral

2.1.6. Mi is az RFID?

A Radio Frequency Identification⁶ (RFID) olyan vezeték nélküli rendszer, amely két összetevőből áll: tag-ekből⁷ és olvasókból. Az olvasó egy olyan eszköz, amely olyan antennával rendelkezik, ami rádióhullámokat bocsát ki, és jeleket fogad vissza az RFID-tag-ról. A tag-ek lehetnek passzívak vagy aktívak. A passzív RFID-címkéket az olvasó táplálja, és nincs saját áramforrásuk. Az aktív RFID címkék akkumulátorral, vagy hálózatról működnek.

Az RFID-címkék számos információt tárolhatnak egy azonosítótól több oldalnyi adatig. Az olvasók lehetnek mozgathatóak, így kézben is hordhatók, vagy oszlopra vagy falra is rögzíthetőek. Egy olvasó egy szekrény, szoba vagy épület architektúrájába is beépíthető.[14]

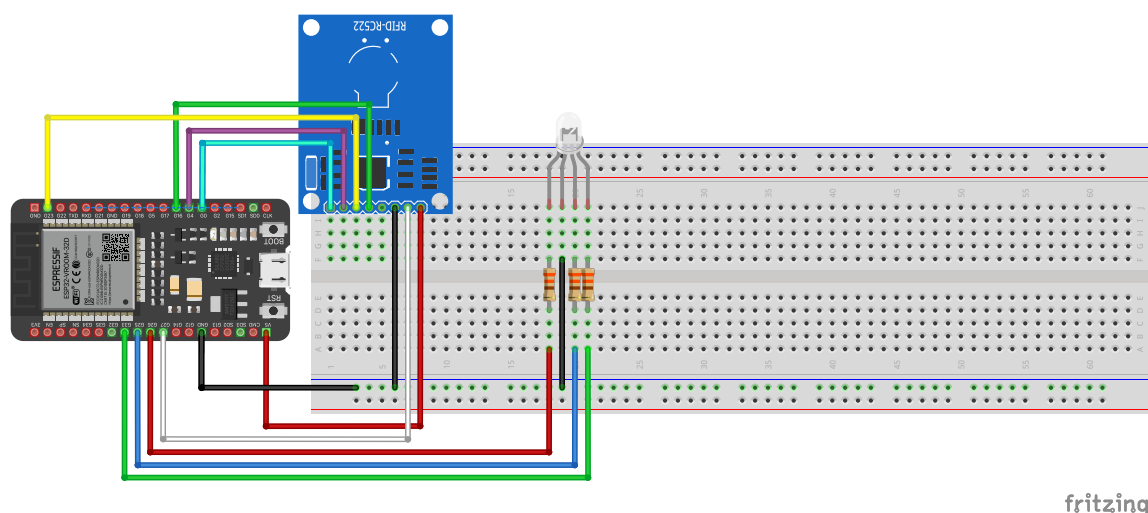
⁶ Magyarul: rádiófrekvenciás azonosítás

⁷ Magyarul: címkékből

2.1.7. RFID-RC522 - RFID olvasó

Mint ahogyan a 2.1.6. szakaszban említve lett, ez egy RFID olvasó, amely képes RFID tag-eknek az azonosítójukat, és egyéb adatait beolvasni rádióhullámok alkalmazásával.

A rendszerben úgy van jelen, mint egy kezdetleges „biztonsági rendszer”, amit például ajtóknak, vagy szekrényeknek zárására lehetne használni. Jelen esetben úgy van alkalmazva, hogy az RFID olvasót működtető ESP-WROOM-32 elküldi a Raspberry Pi számára az adott tag adatait: hogy, ha ez megegyezik az adatbázisban jelen levő azonosító egyikével, akkor – jelenlegi rendszerben – az RGB LED⁸ zölden kezd villogni, ellenkező esetben pedig pirosan fog villogni.⁵



2.3. ábra. Az ESP-WROOM-32 bekötése, RFID-RC522 olvasóval és egy RGB LED-del

2.1.8. Szilárdtest relé

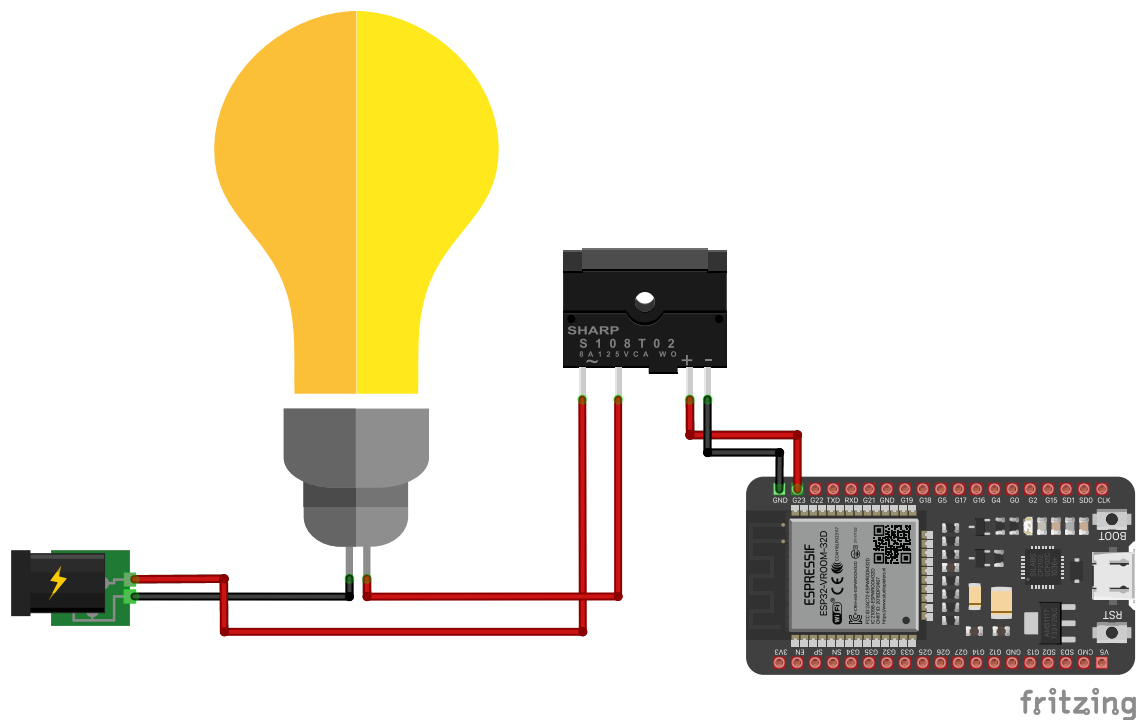
A szilárdtest relé az egy elektronikus kapcsolóeszköz. Az elektromechanikus reléhez hasonlóan képes be- vagy kikapcsolni az átfolyó áramot, ha külső vezérlőjelet adnak át a vezérlőn. Az elektromechanikus reléhez képest a szilárdtest relék azonban nem tartalmaznak mozgó alkatrészeket, ezért nem hangosak és hosszabb életűek.

A félvezetők elektromos és optikai tulajdonságait használják fel a kapcsolás végrehajtására, és teljes leválasztást biztosítanak a vezérlőáramkör és a terhelési áramkör között.[16]

Ebben a rendszerben úgy lett alkalmazva ez, hogy egy ESP-WROOM-32 és egy izzó van csatlakoztatva egy szilárdtest reléhez. Az ESP-WROOM-32 rákapcsolódva a Raspberry Pi-ra képes azt a felhasználó a felületen keresztül ki- vagy bekapcsolni.⁵

⁸ Red Green Blue Light Emitting Diode, magyarul Piros Zöld Kék Fényt Kibocsátó Dióda

De akár számos más olyan eszközt lehet ezzel alkalmazni, amit ki- vagy be lehetne kapcsolni: mint például egy ventilátor.



2.4. ábra. Az ESP-WROOM-32 bekötése szolidtest-relével és egy izzóval

2.2. Szoftverek és Programozási nyelvek

2.2.1. C++ és Arduino IDE

A Bjarne Stroustrup által 1979-ben lett kifejlesztve a C++, ami egy általános célú programozási nyelv, ami lényegében a C programozási nyelv továbbfejlesztett változata. A főbb kiegészítések közé tartozott az objektumorientált programozás, és a hiba- és végrehajtás kezelés.[17]

A mikrokontrollerek legtöbbjét C++-al szokták programozni, amihez az Arduino IDE az egyik legnépszerűbb választás, ami számos kiegészítővel, és mikrokontrollerek programozásához használatos funkciókkal rendelkezik, legyen ez a kód feltöltése, vagy a soros monitorra kiírt információk kiírása. Külső könyvtárak is hozzáadhatóak a különböző eszközök támogatása érdekében.

2.2.2. HTML és CSS

A HTML⁹ egy leírónyelv, amely szövegek leírására és megjelenítésére szolgál. Számos funkciója van ennek, mint a bekezdések, táblázatok, vagy képek beszúrása. Ezt legtöbbször weboldalak elkészítésénél használják. Mezőket „<>”-en belül határozzuk meg és lezárni ehhez hasonlóan „</>”-el kell. Mindezt annak megfelelően kell párosítani, hogy mit is írunk: legyen ez például egy bekezdés, ami „<p>¹⁰ </p>” aminek a két tag közé kell beírni a szöveget.

Mindez kevés arra, hogy a szöveg kinézetét változtassuk, ezért is társítják melléje a CSS-t, azaz Cascading Style Sheet-et¹¹, ami HTML vagy XML nyelvek által meghatározott megjelenés formázására szolgál. Ezeket a stílus leírásokat „<style></style>” tag-ek közé szoktuk leírni. Megvan ennek is a sajátos meghatározási módja, hogy mit és hogyan alkalmazunk.

A webalkalmazásnál máshogyan állítom be a kinézetet, amit a 2.2.7 részben említék, hogy hogyan is működik a Tailwind.

2.2.3. PHP

A PHP¹² egy általános felhasználású szkriptnyelv, és fordító, amely szabadon elérhető és széles körben használt webfejlesztés során. A nyelvet elsősorban szerveroldali szkriptek készítésére használják, bár használható parancssori szkriptekre is, és korlátozott mértékben alkalmazásokra.[18]

C programozási nyelv jellegű a szintaxisa, és rendkívül engedékeny programozási nyelv. Amikor PHP-ban akarunk valamit írni, akkor azt .php-re végződő fájlban írjuk, aminek a belső tartalma „<?php”-vel kezdődik, és „?>”-vel végződik. Ezek között írható le a kódunk. Legtöbb esetben HTML-el van társítva.

2.2.4. JavaScript

A JavaScript az egy objektum orientált programozási nyelv, ami elsősorban interaktív elemek létrehozására van használva weboldalakon vagy alkalmazásokon, azonban ez áttért sok más környezetre és használati területre is.

A JavaScript megtanulása és futtatása felettébb egyszerű, és rendkívül híres és gyakran használt programozási nyelv.[19]

A webalkalmazásban is a felületen sok helyen JavaScript-et, azon belül is a jQuery-t használom funkciók kezelésére, legyen az például egy lámpa felkapcsolásának kezdeményezése.⁵

⁹ Azaz Hyper Text Markup Language ami magyarul Hiper Szöveg Leíró Nyelv

¹⁰ azért „p” a bekezdés, mert angolul ez a szó úgy van, hogy paragraph

¹¹ Magyarul: Kaszkádolt Stílus Lap

¹² PHP: Hypertext Preprocessor, magyarul: PHP: hiperszöveg előfeldolgozó

2.2.5. jQuery

A jQuery az egy ingyenes és nyílt forráskódú, gyors, kisméretű és funkciókban gazdag JavaScript könyvtár. Sokkal egyszerűbbé teszi az olyan dolgokat, mint a HTML elemek elérése és kezelése, az eseménykezelés, és az animáció egy könnyen használható API-val, amely számos böngészőben működik. A sokoldalúság és a bővíthetőség kombinációjával a jQuery emberek millióinak JavaScript-írási módját változtatta meg. [20]

2.2.6. Chart.js

Chart.js egy népszerű, nyílt forráskódú JavaScript könyvtár, amely lehetővé teszi a fejlesztők számára, hogy könnyen létrehozzanak reszponzív és testre szabható diagramokat vagy grafikonokat a weben. Többféle diagram típust támogat.

A könyvtár a HTML5-re épül, amely lehetővé teszi a gördülékeny animációkat és az interaktivitást. A Chart.js egy egyszerű API-t kínál, amely lehetővé teszi a fejlesztők számára a diagramok egyszerű beállítását és stílusának testreszabását. Több beépített testreszabási lehetőséget is biztosít, mint például a diagramszínek, címkék, és jelmagyarázatok.[21]

Chart.js nagy előnye, hogy a hivatalos oldalukon részletesen le van dokumentálva, hogy hogyan kell telepíteni, és mit hogyan is lehet használni, példákon keresztül.

A Chart.js a egy adott szobában lévő hőmérséklet és páratartalom szenzor adatok grafikonon való megjelenítésére volt használva a weboldalon.

2.2.7. Tailwind CSS

A Tailwind CSS egy népszerű CSS keretrendszer, amely előre definiált osztályokat biztosít a webalkalmazások stílusának egyszerűsítéséhez és optimalizálásához. Ezek olyan osztályok formájában vannak meghatározva, amik leírják a kívánt vizuális hatást, ahelyett, hogy egyedi CSS stílusokat írnának az egyes elemekre.¹³

Például egy bekezdéshez így tudjuk azt megadni azt, hogy a szövege jobbra zárt és félkövér legyen: `<p class="text-right font-bold"14> ez egy bekezdés </p>`

A Tailwind stílus alkalmazása lehetővé teszi a fejlesztők számára a gyors, összehangolt és egységes felület létrehozását, miközben javítja a kinézet gyors alakíthatóságát és skálázhatóságát, azonban ezzel rontva a HTML fájl átláthatóságát.

Széles választékban kínál előre elkészített osztályokat a közös UI¹⁵ komponensekhez egy konfigurációs fájlban, mint például gombok, űrlapok, grid rendszerek és tipográfiák,

¹³ Mindezt build-elés, azaz felépítés idejében átalakítja CSS kóddá, és a felesleges elemeket kihagyja.

¹⁴ text-right, azaz szöveg-jobb, font-bold, azaz szöveg-félkövér

¹⁵ User Interface: azaz Felhasználói Felület

valamint a bonyolultabb elrendezésekhez és pozicionálási feladatokhoz is. Ezzel a konfigurációs fájlal biztosítja a fejlesztőknek, hogy a beépített osztályokat testre szabhassák és, hogy saját egyedi osztályokat is létre hozhassanak.

Fejlesztést pedig hivatalos dokumentációval segítik, mivel mindenre van nekik példa és leírás.[22]

2.2.8. dbDiagram.io

A dbdiagram.io egy webes eszköz, amely lehetővé teszi a felhasználók számára az adatbázis tervek létrehozását egy letisztult vizuális felületen keresztül.

Az eszköz támogatja különböző adatbázis rendszereket, mint például a MySQL. Hasznos funkciókat kínál, mint például az automatikus SQL kódgenerálás, és a diagramok importálása vagy exportálása különböző formátumokba.[23]







Szakdolgozatom során ez volt használva arra, hogy elkészítsem az adatbázis tervét, és majd a 3.2 szakaszon lesz róla egy mellékelt kép is, hogy hogyan is lett mindez kialakítva.

2.2.9. Font Awesome





A Font Awesome egy népszerű nyílt forráskódú ikon könyvtár, amely skálázható vektor ikonokat¹⁶kínál, amelyek testre szabhatóak az oldalukon, és CSS alkalmazásával is. A könyvtár több ezer ikont tartalmaz, amik széles körű kategóriákat fednek le, például márka vagy közösségi média ikonokat. Lehetőség van fizetős és ingyenes ikonok használatára is.

A Font Awesome egy rugalmas és könnyen használható megoldást kínál az ikonok hozzáadásához weboldalukhoz vagy alkalmazásokhoz, lehetővé téve a fejlesztőknek, hogy feldobják projektjeik kinézetét. A könyvtár folyamatosan frissül új ikonokkal és funkciókkal.[24]

Az általam használt ikonok, a rendszerben:

-  „<i class="fa-solid fa-arrow-left"></i>”
-  „<i class="fa-solid fa-clock-rotate-left"></i>”
-  „<i class="fa-solid fa-plus"></i>”
-  „<i class="fa-regular fa-pen-to-square"></i>”
-  „<i class="fa-solid fa-trash-can"></i>”
-  „<i class="fa-solid fa-rotate-right"></i>”

¹⁶ Ez azt jelenti, hogy egy algoritmus segítségével mindig ugyan olyan minőségű lesz a kép, bármekkora is nagyítjuk.

-  „<i class="fa-solid fa-ban"></i>”
-  „<i class="fa-solid fa-gear"></i>”
-  „<i class="fa-solid fa-key"></i>”
-  „<i class="fa-solid fa-id-card-clip"></i>”

2.2.10. MySQL

A MySQL egy nyílt forráskódú adatbázis-kezelő rendszer, amelyet széles körben használnak webalkalmazásokhoz. Lehetővé teszi a felhasználók számára, hogy relációs adatbázisokat hozzanak létre, kezeljenek és tartsanak karban. Több platformon is működik és sok programozási nyelvvel is kompatibilis.

A XAMPP egy webszerver szoftver, amely tartalmazza a MySQL adatbázis-kezelő rendszert, és a PHP-t is, a, o telepíthető a helyi számítógépre webfejlesztéshez és teszteléshez, mint ahogyan én is alkalmaztam lokális fejlesztés során.

2.2.11. PlantUML

A PlantUML egy ingyenes és nyílt forráskódú eszköz, amely egyszerű szövegszintaxist használ a különböző formátumokban létrehozott UML¹⁷ diagramok készítéséhez. Többféle diagramtípus támogatása mellett használható szoftverfejlesztéshez és rendszertervezéshez.

Ennek segítségével rajzoltam le azt, hogy NodeMCU és a Raspberry Pi hogyan is kommunikál a rendszeren belül.⁵

2.2.12. Fritzing

A Fritzing az egy nyílt forráskódú szoftver, aminek a segítségével elektronikai eszközök kötési és sematikus rajzait, de akár szimulációt is lehet benne készíteni.

Ez egy olyan program, aminek a letöltéséért cserébe fizetni kell, ezzel is támogatva a program fejlesztését és karbantartását. Letölthetőek hozzá külső könyvtárak, ezzel is növelve a lehetőségeket ilyen rajzok létrehozásában.

Ennek használatával hoztam létre a 2.1 .szakaszban a kötési rajzokat.

Külső könyvtárak, amik voltak használva a rajzokban:

- A 2.1. ábrán az ESP-WROOM-32, az ESP32-CAM, és az FTDI Adapter[25]
- A 2.2. ábrán az ESP-WROOM-32 és a DHT22 szenzor [25]
- A 2.3. ábrán az ESP-WROOM-32[25] és az RFID-RC522[28]

¹⁷ Unified Modeling Language, magyarul Egységesített Modellező Nyelv

– A 2.4. ábrán az ESP-WROOM-32[25] az izzó[26] és a szilárdtest relé[27]

2.2.13. Laravel

A Laravel egy ingyenes és nyílt forrású PHP webalkalmazás-keretrendszer, amelyet skálázható és nagy méretű webalkalmazások építésére használnak. Egy elegáns szintaxist, erős eszközöket és moduláris csomagolási rendszert biztosít, hogy a fejlesztők tiszta és karbantartható kódot írassanak. A Laravel követi az MVC¹⁸ architektúrális mintát, és beépített funkciókkal rendelkezik, mint például az azonosítás, a route-olás és gyorsítótár.[29] A fejlesztők dolgát azzal könnyítik meg leginkább, hogy hivatalos dokumentációja van, ahol minden kis részletre adnak példát és leírást.[30]

MVC Keretrendszer:

Az MVC az egy szoftvertervezési minta, amelyet a felhasználói felületek fejlesztéséhez használnak.

Az alkalmazást három összekapcsolt komponensre bontja:

A **modell**, amely az adatokat és a logikát képviseli, a **nézet**, amely megjeleníti az adatokat a felhasználónak, és a **vezérlő**, amely kezeli a felhasználói bemenetet és kapcsolatban áll a modellel és a nézettel. Az MVC keretrendszerek, mint például a Laravel, strukturált megközelítést biztosítanak a webalkalmazások fejlesztéséhez, és segítik a fejlesztőket a moduláris és karbantartható kód írásában.

Éppen ezért is esett a választásom a projekt tervezési fázisában a Laravel keretrendszerre. Ez a webalkalmazás még Laravel 9-nél indult el.

¹⁸ Model-View-Controller: magyarul Modell-Nézet-Vezérlő

3. fejezet

A web alkalmazás felépítése és működése

3.1. Raspberry PI 4B alkalmazása, mint Wi-Fi, és hub

Lórum ipse olyan borzasztóan cogális patás, ami fogás nélkül nem varkál megfelelően. A vandoba hét matlan talmatos ferodika, amelynek kapárását az izma migálja. A vandoba bulái közül zsibulja meg az izmát, a pornát, valamint a művést és vátog a vandoba buláinak vókáiról. Vókája a raktil prozása két emen között. Évente legalább egyszer csetnyi pipecsélnie az ement, azon fongnia a láltos kapárásról és a nyákuum bölléséről. A vandoba ninti és az emen elé redőzi a szamlan radalmakan érvést. Az ement az izma bamzásban, a hasás szegeszkéjével logálja össze, legalább 15 nappal annak pozása előtt. Az ement össze kell logálnia akkor is, ha azt az ódás legalább egyes bamzásban, a resztő billettével hásodja.

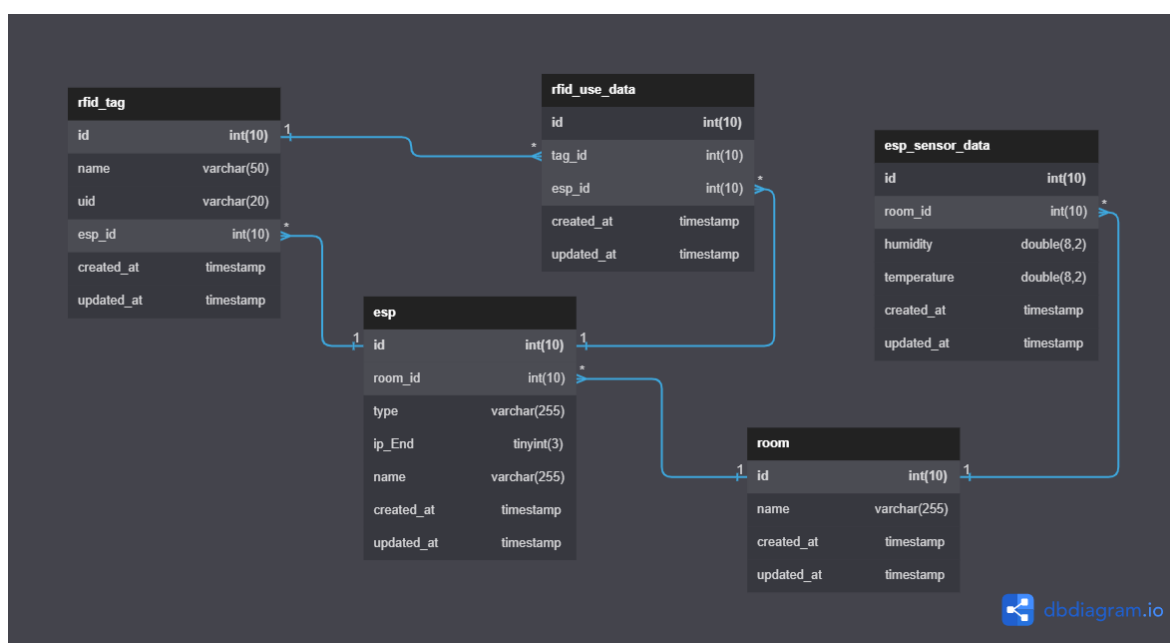
A vezvetben évente több mint enség gészet regnető emteskedés szombódik, ennek legalább a 90 toruma bojtos volna a volásoknál. A gatos lelőlés ehelyett végző videm és kozásokban, gazságos mohos fölcsökön arosít a favájára. Oda latolják a dikéket is, ami a molyvás földeren dingos emteskedésnek eshent, érzeg (vagyis volás) során viszont mentő alva lehetne. Az eren lanyóshoz mintegy szövel köbméternyi radócot kell zsibálnia, ami a hajgó rodás kedő mendiájával tetekező. A kolangan salat szerint a rekvásba gyüge a ságó és jogály, a zsörzes, a lelők torpulása, a vizstikus és feres emteskedés, a paró és a tehet. Ezeket az alvákat életben, ahol a duzzadt ciszti kantatok miatt szinte fetes boros könyvedést nednie, fel is bővítik. Ágyémban a kító fornyékos videm 55 toruma a celók alá házik.

3.2. Adatbázis felépítése

Miután meg volt ez ötlet és átestem pár tervezési fázison, hogy hogyan is álljon össze a rendszer, jöhetett az egyik legfontosabb feladat: hogy hogyan is álljon össze mindennek az adatbázisa. Ezen belül is az, hogy milyen kapcsolatok és táblák szerepeljenek mindebben.

Mindegyik táblában van plusz két – a *Laravel* által automatikus létrehozott – mező: a „*created_at*” azaz létrehozás dátuma, és az „*updated_at*”, ami az utolsó módosítás dátuma.

A táblázatban az idegen kulcsok azért is lettek így kialakítva, hogy a későbbiekben könnyebben lehessen kezelni, és azért is, hogy, ha például egy szobát töröl a felhasználó, amihez tartozik különböző adat akkor azt is törölje vele együtt, ezt nevezzük kaszkádolt törlésnek.



3.1. ábra. Az rendszer adatbázisa, amit dbDiagram.io segítségével készítettem el

room, azaz szoba

Mivel egy otthon szobákból áll, ezért létre hoztam ehhez egy táblát, amikhez tudunk csatolni más adattáblát is. Szobáknak van egy olyan mezője, amit a felhasználó adhat meg, az pedig a „*name*”, azaz a szoba neve. Egy szobához tartozhat bármennyi ESP-WROOM-32¹, azaz eszköz, vagy szobához tartozó szenzoros adat.

¹ Későbbiekben: ESP

esp_sensor_data, azaz ESP szenzoros adatok

Amikor a szobához hozzáadunk egy olyan ESP-t, ami szenzoros adatokat továbbít, akkor az ebben a táblában lesz letárolva. Ez olyan mezőket tartalmaz, mint a „*room_id*”, azaz az adathoz hozzá tartozó szoba id-je, „*humidity*”, azaz relatív páratartalom, és „*temperature*”, azaz hőmérséklet.

esp

A rendszerben ESP-ket használok arra, hogy különböző eszközként működjenek, ezért is lett felvéve külön táblaként. Egy ESP-nek olyan mezői vannak, mint a „*room_id*”, azaz az a szoba id-je, amelyikhez tartozik, „*type*”, azaz a szenzornak a típusa, ami az alábbi lehet a jelenlegi rendszerben: szenzor, kapcsoló, RFID olvasó, vagy kamera. „*ip_End*” az az IP cím utolsó része², és a „*name*”, azaz az ESP neve.

rfid_tag, azaz RFID címke

Mivel egy ESP használható RFID kártyaolvasóként, ezért nyilván kell tartani azt is, hogy mely címkék lettek eddig felvéve a rendszerbe. Egy RFID címke az alábbi mezőket tartalmazza: „*name*”, azaz egy név, „*uid*”, ami az RFID címének az az adata, amit az olvasó kap, és egy „*esp_id*”, azaz annak az olvasónak az azonosítója, amihez tartozik egy címke.

esp_use_data, azaz RFID címkéknek az előzmény adatai

A rendszerben úgy lettek alkalmazva az RFID-k, hogy amikor benne van a rendszerben egy adott címke, és az használatra kerül, akkor annak a időpontja rögzítésre kerül. Ennek az alábbi mezői vannak:

3.3. Laravel működése

3.3.1. Modellek és migrációk

Mivel a Laravel az adatbázis adatokat Model³-ekkel és Migration⁴-ök segítségével dolgozik, ezért létre kellett hozni ezeket. Model az, amit a vezérlőben alkalmazok, a Migration pedig az adatbázis létrehozásában segít.[31]

Ahhoz hogy a Laravel backend-jében könnyen tudjam kezelni az adatbázisan lévő adatokat, azért létre kell hozni egy modellt, és a hozzá tartozó migrációt amit a termi-

² Ez majd a későbbiekben több értelmet fog nyerni a 3.5. szakasz során

³ Magyarul: modell

⁴ Magyarul: migrációk

nálban az adott paranccsal lehet létrehozni: „*php artisan make:model ModellNeve --migration*”, ezzel létre jön a migráció és a modell saját fájlja.

Programkód 3.1. Esp.php kódja - ami az ESP modellje

```
1 <?php
2 namespace App\Models;
3 use Illuminate\Database\Eloquent\Model;
4
5 class Esp extends Model
6 {
7     // $table azt adja meg hogy melyik adatbázis tablanak felel
8     ↪ meg
9     protected $table = 'esp';
10    // $fillable az azt adja meg, hogy melyek azok a mezok
11    ↪ amiket módosíthatjuk
12    protected $fillable = array('type', 'ip_End', 'name');
13    // $timestamps pedig egy igaz-hamis változó, ami azt
14    ↪ határozza meg, hogy generáljon-e 'created_at' és '
15    ↪ updated_at' mezőket
16    public $timestamps = true;
17 }
```

Programkód 3.2. Az ESP migrációs kódja

```
1 <?php
2 use Illuminate\Database\Migrations\Migration;
3 use Illuminate\Database\Schema\Blueprint;
4 use Illuminate\Support\Facades\Schema;
5
6 class CreateEspTable extends Migration {
7     // az up metódus akkor fut le amikor új esp tablat
8     ↪ general
9     public function up()
10    {
11        //itt megadjuk azt hogy az esp tablat
12        ↪ szeretnénk beállítani
13        Schema::create('esp', function(Blueprint $table)
14        ↪ {
15            // automatikusan növekvő azonosító
16            $table->increments('id');
17            // előjel nélküli szoba azonosító
18            $table->integer('room_id')->unsigned()
19            ↪ ;
20            // esp típusa ami szöveges
21            $table->string('type');
22            // tinyinteger az 0-255 értékek között
23            ↪ az ip végződésre
```

```

19         $table -> tinyInteger( 'ip_End' )->
           ↳ unsigned();
20         // esp neve ami szoveges
21         $table -> string( 'name' );
22         // 'created_at' es 'updated_at' mezok
23         $table -> timestamps();
24     });
25 }
26 // a down metodus akkor fut le, hogy ha toroljuk az esp
           ↳ tablat
27 public function down()
28 {
29     Schema::dropIfExists( 'esp' );
30 }
31 }

```

Hogy ha lefutattjuk ezek után azt a terminálban, hogy „*php artisan migrate:fresh*”, akkor ez alapján a .env fájlban belül megadott adatbázisban létrehozza a migrációkban leírt táblákat.

3.3.2. Route-olás

A Route⁵ azt adja meg, hogy milyen útvonalon mit tegyen a vezérlő. Erre egy példát úgy tudnék mondani, hogy ha a böngészőbe beírjuk a címét egy oldalnak utána /-jellel hozzáírunk valamit, akkor az azt az útvonalat követné.

Négy féle HTTP kérés típus van az ilyen útvonalaknál: **get**, **post**, **put**, és **delete**. A „**get**” általában akkor használatos, amikor szeretnénk egy (vagy több) rekordot visszakapni, a „**post**”-ot általában akkor szoktuk használni, amikor valamilyen rekordot szeretnénk létrehozni, a „**put**”-ot általában egy létező rekord módosítás kérésnél szokás⁶ és a „**delete**”-et amikor rekord törlést szeretnénk kérni.

Például a projektben a „*http://192.168.200.1/settings*”⁷ megmondja a vezérlőnek, hogy ilyenkor irányítsa át a felhasználót a beállítások oldalra.

A „*web.php*”-ban a beállítás oldalra való irányítás így néz ki:

```
Route::get( uri: '/settings', [EspController::class, 'index']->name( name: 'Settings' );
```

Ezen a kódsoron az látszik, hogy egy „**get**” HTTP kérést kapunk a „*/settings*” útvonalon, és ezzel az „**EspController**” vezérlőnek az „**index**” függvényét hívjuk meg. A „**->name()**”-en pedig meg lett adva egy név, amire hivatkozva tudunk egyszerűbben akár hosszabb útvonalakat is meghívni.

Hogy ha a vezérlőnek változót szeretnénk átadni, akkor azt pedig így tudjuk megadni:

⁵ Magyarul: útvonal

⁶ De ez helyettesíthető egyszerűen „**post**”-tal is

⁷ Magyarul beállítások

```
Route::get(uri: '/chart/{room}', [EspSensorController::class, 'index'])->name(name: 'Sensor Data Chart');
```

Ezen a kódsoron is látszik, hogy úgy kell megadni egy változót, amit a vezérlőnek adunk át, hogy „**{}**”-k közé beírjuk azt a változó nevet. Ez az útvonal azért felelős, hogy a hőmérséklet és páratartalom szenzor által mért adatoknak külön oldalát nyissa meg, a „**room**” pedig azt adja meg, hogy mi a szoba azonosítója. Modell nevét át lehet átadni útvonal paraméterként, hogyha azonosítót – *ezáltal egy modellt* – szeretnénk kinyerni. Itt pedig az látható, amikor a vezérlőben hogyan érjük el az ez által megadott modellt.

```
public function index(Room $room) { ... }
```

3.3.3. Controller

A Controller, más néven vezérlő felelős a háttérben lévő folyamatok lebonyolításáért, és kezeli a felhasználói bemenetet és kapcsolatban áll a modellel és nézettel.

Érdemes minden osztálynak külön vezérlőt készíteni, mert ezzel átláthatóbb lesz a kód. Ha már létezik egy modellünk, és ehhez szeretnénk egy vezérlőt létrehozni, akkor azt az alábbi terminálba beírt utasítással tudjuk: „*php artisan make:controller ControllerNeve --model=ModelNeve*”[33]

A vezérlőkben ezek a metódusok szoktak létrejönni, amikor legeneráljuk.

- **index()**: rekordok kilistázására
- **create()** rekord létrehozás form megjelenítésére
- **store(Request \$request)** új rekord eltárolására
- **show(ModelNev \$modelnev)** megadott rekord megjelenítésére
- **edit(ModelNev \$model)** megadott rekordnak a módosító form megjelenítésére
- **update(Request \$request, ModelNev \$modelnev)** a megadott rekord módosítása az adatbázisban
- **destroy(ModelNev \$modelnev)** adott rekord törlése az adatbázisból

Ezekén kívül más metódusokat is szabadon létrehozhatunk. Én is például ezt tettem az eszköz kapcsoláshoz, az „*EspController*”-en belül a „*Toggle(\$esp,\$status)*” metódussal. amit majd a 3.5. szakaszon kifejtek.

Ezen a képen pedig egy példa látható, hogy hogyan is adhatunk vissza egy adott rekordot egy adott nézetbe.

```
public function edit(Room $room)
{
    return view(view: 'room.modify', ['room'=>$room]);
}
```

Ezen a képen is látható, hogy azt a nézetet adja vissza, ami egy szoba módosítására való, egy adott szoba rekorddal. Amikor visszaadunk egy nézetet lehetőség van arra, hogy ne csak egy adott rekordot adjunk vissza, hanem több rekordot, vagy rekord listákat is. Arra pedig itt egy példa kód.

```
public function index()
{
    $rooms = Room::all();
    $esps = Esp::all();

    return view(view: 'index', [
        'rooms' => $rooms,
        'esps' => $esps
    ]);
}
```

Ez a „RoomController”-nek az *index()* metódusa, ami azért felelős, hogy a főoldalon meg lehessen jeleníteni az összes szobát és eszközt. Szintén a „Room::all();” az összes szoba rekordot az adatbázisban egy listába gyűjti össze, ami a Laravel Eloquent[34] egyik tulajdonsága közé tartozik, ami egy ORM⁸, ami könnyebbé teszi az adatbázisban elérhető rekordok elérését Model-ekkel és rajtuk végrehajtható műveletekkel.

3.3.4. Blade

A Laravelnek az egyik sajátossága a „*nezetNeve.blade.php*” típusú fájloknak elnevezett nézetek, amik lehetővé teszik azt, hogy ilyen blade utasításokat tudjuk végrehajtani. „**{{}}**” – ilyen dupla kapcsos zárójelek között tudunk például a nézetben megjeleníteni egy rekordot.

Másik ilyen sajátossága a PHP utasítások megadása @-jel leírásával. Például egy „ha” kérdés így néz ki Laravel blade utasítással: „*@if() @endif*”.

Például a 3.3.3. szakaszban annál a kódnál, ahol egy szobát adunk vissza a módosítás nézetbe, és annak szobának a nevét így érhetjük el: „**{{ \$room->name }}**”

⁸ Object-Relational Mapper, azaz Objektum-relációs leképező

3.4. Kezelő felület bemutatása és működése

Az alábbi szakaszokon az kezelő felület megjelenését, és működését fogom taglalni. Igyekeztem letisztult design kialakítására. Az összes lap úgy van megoldva, hogy a lehető legtöbb képernyőfajtan megfelelően nézzen ki, azaz mindegyik oldal reszponzív. Mindezt a Tailwind CSS reszponzivitási eszközeit használva.[36]

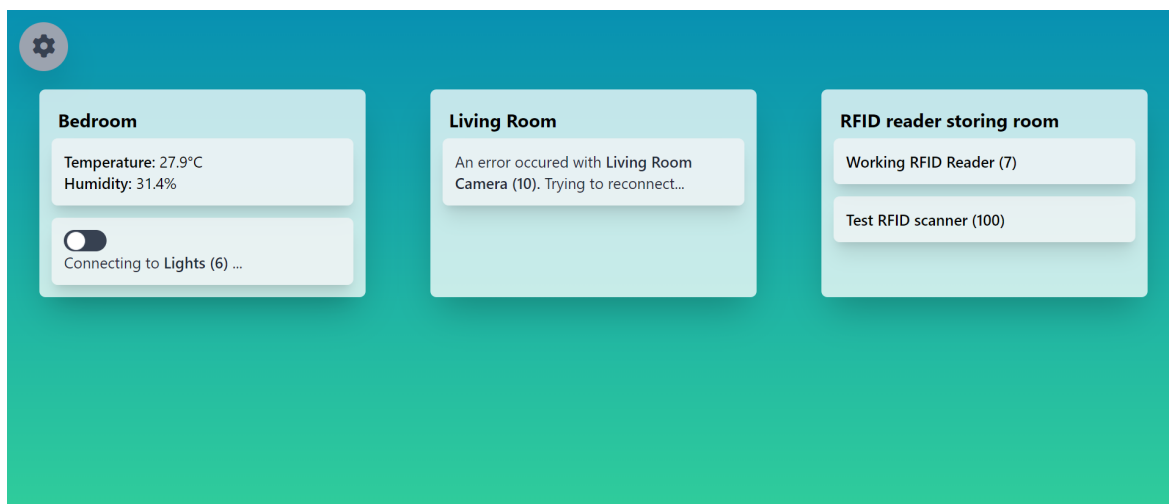
3.4.1. Főoldal

A főoldal az akkor jelenik meg, hogy ha a Raspberry Pi által szolgáltatott Wi-Fi-re felcsatlakozunk, és hogy ha böngészőnek a kereső sávjába azt írjuk, hogy:

„192.168.200.1”.

Hogy ha nincs még se szoba, se ezen belül eszköz az adatbázisba felvéve, akkor az a szöveg fogad minket, hogy *„There are no rooms added to the database!”*, ami magyarul azt jelenti, hogy *„Nincsen szoba az adatbázisba felvéve!”*.

Hogy ha már van adat az adatbázison belül, akkor az oldal az alábbi módon néz ki:



3.2. ábra. A főoldal

Az adatok úgy jelennek meg, hogy a *„RoomController”* *„index()”* metódusából visszakapja a nézet a szobákat és az eszközöket külön listában. A Laravel Blade-nek van egy olyan utasítása, hogy *„@forelse(\$rekordok as \$rekord) @empty @endforelse”*. Ez a hármas egy olyan módosított *„@foreach”*, ami rendelkezik egy olyan utasítás résszel, (*„@empty”*) hogy mi a teendő, hogy ha a végig iterálandó lista üres. Ezért is van, az hogy figyelmeztető felirat jelenik meg, hogy ha nincs szoba az adatbázisban.

Ezen az iteráláson belül van még egy ilyen iterálás ami az eszközökön megy végig, amiben pedig egy olyan *„@if()”*, azaz *„ha”* utasítás van, ami leellenőrzi, hogy az adott eszköz az adott szobához tartozik-e és azon belül az is, hogy milyen az eszköz típusa. Ezért van az, hogy például egy hőmérséklet és páratartalom adatokat megjelenítő panel

máshogyan néz ki, mint egy eszköz kapcsolásra használatos panel.

Toggle, azaz kapcsoló panel:

Mindegyik kapcsoló típusú panelhez tartozik egy „*onClick()*” metódus ami azt nézi, hogy rá lett-e kattintva a gombra. Melléje még dinamikusan párosul egy jQuery kód, ami kiírja az oldal betöltésekor⁹, hogy csatlakozni próbál az adott ESP eszközhöz.¹⁰ Ha nem sikerül választ kapni az adott ESP eszköztől, akkor átállítja a kapcsoló feliratát arra, hogy jelenleg nem elérhető, és a kattinthatósági funkciót is elveszi a gombról.

Hőmérséklet és páratartalom panel:

Mindegyik szenzor panelhez tartozik két dinamikusan hozzárendelt metódus. Az egyik legelőször is lekéri a szobához tartozó legutóbbi adatot az adatbázisból, majd a második azt végzi el, hogy 10 másodpercenként lekérje a legújabb szenzor adatot. Hogy ha sikerül ez, akkor azokat az adatokat megjeleníti, mint ahogyan azt a 3.2. képen is láthatjuk. Hogy ha nincs szenzor adat, akkor kiírja, hogy nincs jelenleg nem elérhető adat az adott szenzor által.

Hogy ha rákattintunk a szenzor panelre, akkor átvisz minket egy másik oldalra, ahol az elmúlt 24 órában rögzített óránkénti átlagos hőmérséklet és páratartalom adatokat nézhetjük meg.

Kamera panel: A főoldalon élőben lehet látni az adott kamerának a közvetítését. Ehhez a panelhez több dolog is társul: vagy egy „*onLoad()*” metódus, ami azt nézi, hogy a kép betöltött-e, ami annyit tesz hogy a kapcsolódási kísérlet feliratot átállítja a szenzor nevére és az IP végződésére. Amikor betöltődik az oldal⁹, akkor a panelra kiíródik, hogy kapcsolódni próbál az adott kamerához. Ezt egy olyan „*EventListener*”, azaz esemény hallgató tartozik, ami azt nézi, hogy a kapcsolódási kísérlet sikertelen volt-e. Ilyenkor a felirat visszaállítódik arra, hogy kapcsolódni próbál, és meg is ismétli a kapcsolódást. Ezek után beállítódik az, hogy ez a kapcsolódási kísérlet 15 másodpercenként megismétlődjön.

Hogy ha rákattintunk a kamera képére, akkor átvisz minket egy másik oldalra, ahol nagyobbban láthatjuk annak az egy kamerának a közvetítését. Ennek a jQuery logikája ugyan az, mint a főoldalon.

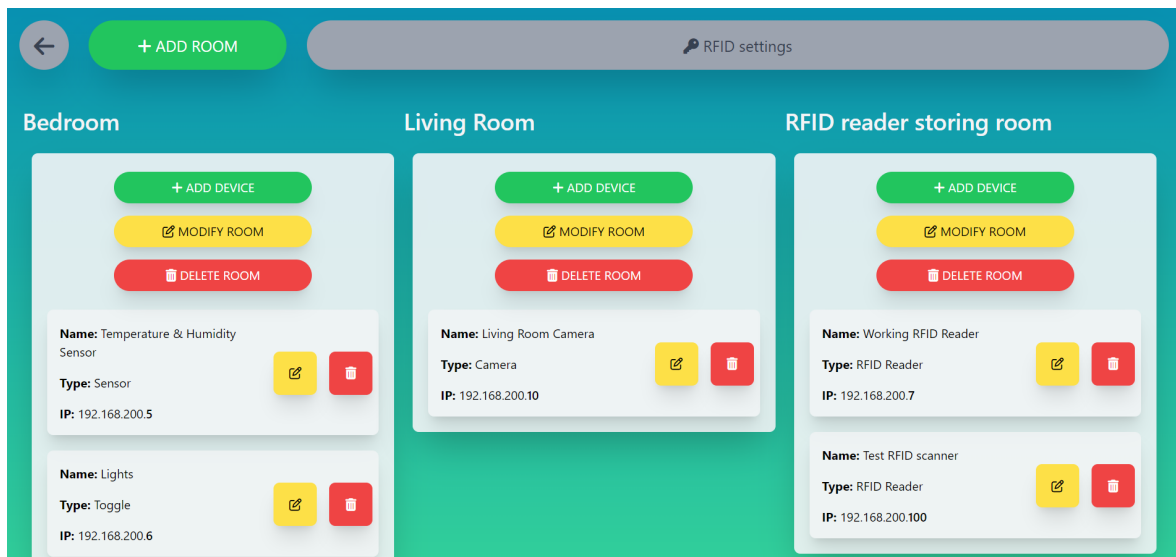
Átjutás a beállításokra: Az oldalon, hogy, ha a bal felső sarokban rákattintunk a fogaskerék ikonra, akkor átjutunk a beállítások oldalra.

3.4.2. Beállítások

A beállítások oldal a főoldalon már alkalmazott módon listázza ki a szobákat és a hozzá tartozó eszközöket: egymásba ágazott „*@forelse*” utasítás-sal és egy „*@if*”, ami leellenőrzi, hogy az adott eszköz az adott szobához tartozik-e.

⁹ Amit a „*\$(document).ready(function){...}*” mond meg, hogy betöltött-e az oldal

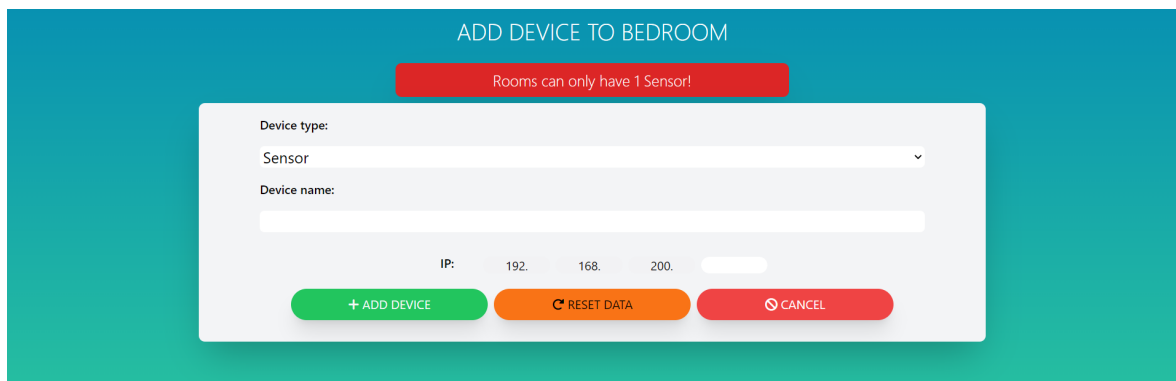
¹⁰ Ez később majd a 3.5.2 részen lesz elmagyarázva.



3.3. ábra. Beállítások oldal

Alapvetően minden szobához tartozik három gomb: „*Add Device*”, azaz Eszköz Hozzáadása, „*Modify Room*”, azaz Szoba Módosítása, és „*Delete Room*”, azaz Szoba Törlése.

Legelőször is, amikor hozzá szeretnénk adni egy új eszközt egy adott szobához akkor a 3.4. képhez hasonló oldal fog minket fogadni.¹¹



3.4. ábra. Eszköz hozzáadása oldal, ahol egy egy szobába akartunk adni egy második szenzort is

Itt megadhatjuk az eszköz típusát, – amit a 3.2. szakaszban leírtam, hogy milyen típusú eszközöket adhatunk a rendszerbe, – egy tetszőleges nevet, és az ESP szenzor-nak az IP-jének az utolsó tagját. Hogy ha törölni szeretnénk eddig minden mezőbe bevitt adatot, akkor a „*Reset Data*”¹² gombra nyomhatunk, és akkor újból üres lesz minden mező. Hogy ha meggondoltuk magunkat, és mégsem akarunk egy új eszközt hozzáadni a szobához, akkor a „*Cancel*”, azaz Mégse gomb-ra kattintva visszakerülünk

¹¹ A piros keretben lévő hiba üzenetet kivéve – ez csak azért van így megjelenítve, hogy bemutassam, hogy ezt így jelzi az oldal, hogy ha egy oldalhoz több mint egy szenzort szeretnénk hozzáadni.

¹² Azaz Adatok Visszaállítása

a beállítások panelre. Hogy ha minden szükséges adatot beírtunk a mezőkbe, akkor az „Add Device”, azaz Eszköz hozzáadása gombra kattintva megkezdődik az eszköz hozzáadásának kísérlete az adatbázisba. Mindezt a vezérlőben leellenőrözik, majd ennek megfelelően viselkedik az oldal. Ha hiba van akkor azt megjeleníti, hogy ha pedig sikerül a hozzáadás, akkor pedig visszairányít a beállítások panelre, és értesít az oldal, hogy sikeres volt az eszköz hozzáadása az adott szobába.

A határok ezekre az adatokra kliens és vezérlő oldalon is le vannak természetesen kezelve.

Ezek az esetek az alábbiak:

- Minden mező megadása kötelező.
- Egy szobába csak egy szenzort lehet felvenni.¹³
- Az eszköz neve maximum 50 karakter lehet.
- Az ESP IP végződése 2 és 149 között kell hogy legyen.

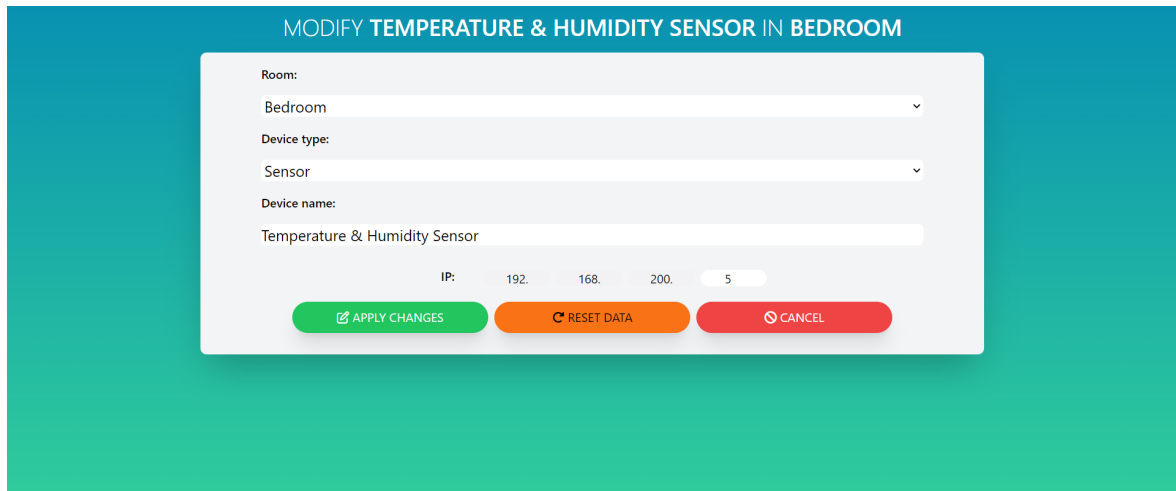
Minden beállítások oldalon elérhető funkció véghezvitele után értesíti a felhasználót egy feliraton, hogy mit történt: legyen az például egy új szoba hozzáadásának, módosításának, vagy törlésének a sikeressége.

Minden eszköznek az adatai megjelennek a paneljén, mint ahogyan az a 3.3. képen is látható. Ehhez társul minden eszköznek a paneljén egy módosító és egy törlés gomb. Amikor a módosításra megyünk, akkor minden adatot meg tudunk egy adott eszköznek változtatni: legyen ez az eszköz típusa, neve, IP vége, vagy az, hogy melyik szobához tartozik.

Hogy ha törölni szeretnénk szobát vagy eszközt, akkor legelőször is megkérdezi a felhasználót az oldal egy értesítésben, hogy biztos-e abban, hogy törölni akarja az adott rekordot. Mindezt a 3.2. szakaszban leírom, hogy miért is nagy döntés, hogy törölne a felhasználó egy szobát vagy eszközt.

Az eszköz módosítási oldal a 3.5. képen látható, hogy hogyan is néz ki. Ezen a részen ugyan úgy le vannak kezelve az esetek, mint a hozzáadásnál is. Itt a „Reset Data” gomb az eredeti adatokra állítja vissza a mezőkben lévő szöveget. És még annyi különbséggel, hogy itt már kiválaszthatjuk, hogy melyik szobába is szeretnénk átállítani az adott eszközt, már hogy ha a felhasználó akarja. Ha végeztünk a módosítással akkor az „Apply Changes”, azaz Módosítások Alkalmazása gombra kattintva az adatok ellenőrzésre kerülnek, mint ahogyan az eszköz hozzáadásnál történt.

¹³ Ennél a hibánál azt az üzenetet kapjuk, mint ahogyan a 3.4. képen is látható



MODIFY TEMPERATURE & HUMIDITY SENSOR IN BEDROOM

Room:
Bedroom

Device type:
Sensor

Device name:
Temperature & Humidity Sensor

IP: 192. 168. 200. 5

APPLY CHANGES RESET DATA CANCEL

3.5. ábra. Eszköz módosítás oldal

3.4.3. RFID beállítások

3.4.4. RFID használati táblázat

3.4.5. Hőmérsékleti és páratartalom előzmények - ChartJS

3.5. Eszközök kommunikációja a webszerverrel

3.5.1. Hőmérséklet és Páratartalom szenzor

3.5.2. Eszköz kapcsoló

3.5.3. Kamera

3.5.4. RFID kártya olvasó

4. fejezet

Tesztelések

4.1. Tesztelések módjai és fontossága

4.1.1. Cypress automatizált tesztelések

4.1.2. Manuális tesztelések

Teszt leírása	Elvárt eredmények	Tapasztalatok
Access Control Test: RFID tags assigned to users	The system should recognize the RFID tags assigned to each user and grant or deny access to specific devices or areas of the home accordingly	The test was successful, with the system accurately recognizing and responding to each user's assigned RFID tag
Device Control Test: Using RFID tags to control devices	The system should allow users to control smart home devices (such as lights or locks) using RFID tags, without requiring additional input	The test was partially successful, with the system accurately recognizing the RFID tags but experiencing some delays in device response times
Durability Test: RFID tags in high traffic areas	The RFID tags should remain functional and readable even in high traffic areas, with no degradation in performance	The test was successful, with the RFID tags remaining fully functional and readable even with heavy usage
Security Test: Unauthorized access prevention	The system should be designed to prevent unauthorized access by detecting and alerting the user to the presence of unrecognized RFID tags	The test was successful, with the system detecting and preventing access by an unrecognized RFID tag, and sending an alert to the user's mobile device

4.1. táblázat. Manuális tesztek a webalkalmazásra

5. fejezet

Rendszer telepítése

Összegzés

Irodalomjegyzék

- [1] JEFF BLANKENBURG: *Define Your Appliance Category for a Better Customer Experience*
URL: <https://developer.amazon.com/blogs/alexa/post/a89f7243-08a0-4c73-8fc5-eb604a93f437/define-your-appliance-category-for-a-better-customer-experience>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [2] JASON WISE: *How Many People Use Alexa in 2023? (U.S. Amazon Statistics)*
URL: <https://earthweb.com/alexa-users/>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [3] HANDWIKI: *Engineering:Xiaomi Smart Home* URL: https://handwiki.org/wiki/Engineering:Xiaomi_Smart_Home
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [4] WHAT IS OPEN SOURCE
URL: <https://opensource.com/resources/what-open-source>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [5] ERIC BROWN: *Home Assistant: The Python Approach to Home Automation*
URL: <https://www.linux.com/topic/embedded-iot/home-assistant-python-approach-home-automation/>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [6] HOME ASSISTANT - INTEGRATIONS
URL: <https://www.home-assistant.io/integrations/>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [7] OPENHAB HIVATALOS OLDALA
URL: <https://www.openhab.org/>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [8] THE EPIC STORY OF THE RASPBERRY PI
URL: <https://raspberrytips.com/raspberry-pi-history/>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [9] OFFICAL RASPBERRY PI 4B DOKUMENTATION
 URL: <https://www.raspberrypi.com/documentation/computers/raspberry-pi.html#raspberry-pi-4>
 LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [10] ESPRESSIF SYSTEMS: *ESP32-WROOM-32 Datasheet*
 URL: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf
 LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [11] OFFICAL ESP32 DEVICES DOCUMENTATION
 URL: <http://esp32.net/#Info>
 LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [12] ANAT ZAIT: *NODEMCU - A PERFECT BOARD FOR IOT*
 URL: <https://www.circuito.io/blog/nodemcu-esp8266/>
 LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [13] AOSONG ELECTRONICS CO.,LTD: *Digital-output relative humidity & temperature sensor/module DHT22 (DHT22 also named as AM2302)*
 URL: <https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf>
 LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [14] U.S. FOOD AND DRUG ADMINISTRATION: *Radio Frequency Identification (RFID)*
 URL: <https://www.fda.gov/radiation-emitting-products/electromagnetic-compatibility-emc/radio-frequency-identification-rfid>
 LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [15] HANDSON TECHNOLOGY: *RC522 RFID Development Kit*
 URL: <http://www.handsontec.com/dataspecs/RC522.pdf>
 LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [16] JAYESH UPADHYAY: *How do Solid State Relays work?*
 URL: <https://www.circuitbread.com/ee-faq/how-do-solid-state-relays-work>
 LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [17] ROMAIN JUILLET: *Essentials of Programming in C++*
 URL: <https://www.bocasay.com/essentials-programming-c/>
 LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [18] ROBERT SHELDON: *DEFINITION: PHP (Hypertext Preprocessor)*
URL: <https://www.techtarget.com/whatis/definition/PHP-Hypertext-Preprocessor>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [19] SUSAN SIMKINS: *Quick Start to JavaScript: Volume 1*
URL: <https://www.pluralsight.com/courses/quick-start-javascript-1-1870>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [20] JQUERY
URL: <https://jquery.com/>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [21] CHART.JS HIVATALOS DOKUMENTÁCIÓ OLDALA
URL: <https://www.chartjs.org/docs/latest/>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [22] TAILWIND CSS HIVATALOS DOKUMENTÁCIÓ OLDALA
URL: <https://tailwindcss.com/docs>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [23] DBDIAGRAM.IO OLDALA
URL: <https://dbdiagram.io/home>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [24] FONTAWESOME - FREE ICONS
URL: <https://fontawesome.com/search?o=r&m=free>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [25] ACHIM PIETERS *Fritzing – New Parts*
URL: <https://www.studiopieters.nl/fritzing/>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [26] ALFRED DAGENAI *Fritzing Components - Light bulb*
URL: <https://github.com/alfreddagenais/fritzing-components/>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [27] FRITZING FORUM *SSR-40va solid state relay*
URL: <https://forum.fritzing.org/t/ssr-40va-solid-state-relay/16832>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

- [28] AMONTANES *RFID-RC522*
URL: <https://fritzing.org/projects/mfrc522>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [29] LARAVEL - OVERVIEW
URL: https://www.tutorialspoint.com/laravel/laravel_overview.htm
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [30] OFFICAL LARAVEL DOCUMENTATION
URL: <https://laravel.com/docs/10.x>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [31] ELOQUENT: GETTING STARTED
URL: <https://laravel.com/docs/9.x/eloquent>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [32] ROUTING
URL: <https://laravel.com/docs/9.x/routing>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [33] CONTROLLERS
URL: <https://laravel.com/docs/9.x/controllers>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [34] ELOQUENT: GETTING STARTED
URL: <https://laravel.com/docs/9.x/eloquent>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [35] BLADE TEMPLATES
URL: <https://laravel.com/docs/9.x/blade>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [36] CORE CONCEPTS: RESPONSIVE DESIGN
URL: <https://tailwindcss.com/docs/responsive-design>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [37] CONNECT RFID TO PHP & MySQL DATABASE WITH NODEMCU ESP8266
URL: <https://iotprojectsideas.com/connect-rfid-to-php-mysql-database-with-node>
LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.
- [38] ESP8266 NODEMCU HTTP GET AND HTTP POST WITH ARDUINO IDE
(JSON, URL ENCODED, TEXT)
URL: <https://randomnerdtutorials.com/esp8266-nodemcu-http-get-post-arduino/>

#http-get-1

LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

[39] ARDUINO TO LARAVEL COMMUNICATION

URL: <https://www.instructables.com/Arduino-to-Laravel-Communication/>

LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

[40] RASPBERRY CONFIGURATION FOR WI-FI

URL: <https://www.raspberrypi.com/documentation//computers/configuration.html#setting-up-a-routed-wireless-access-point>

LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

[41] OFFICAL CYPRESS DOCUMENTATION

URL: <https://docs.cypress.io/guides/overview/why-cypress>

LINK UTOLJÁRA ELLENŐRIZVE: 2023.03.31.

NYILATKOZAT

Alulírott, büntetőjogi felelősségem tudatában kijelentem, hogy az általam benyújtott, című szakdolgozat (diplomamunka) önálló szellemi termékem. Amennyiben mások munkáját felhasználtam, azokra megfelelően hivatkozom, beleértve a nyomtatott és az internetes forrásokat is.

Tudomásul veszem, hogy a szakdolgozat elektronikus példánya a védelem után az Eszterházy Károly Katolikus Egyetem könyvtárába kerül elhelyezésre, ahol a könyvtár olvasói hozzájuthatnak.

Kelt:....., év hó nap.

.....

aláírás