

Fakulta riadenia a informatiky

Inteligentný skleník

Diplomová práca

Bc. Dominik Regec

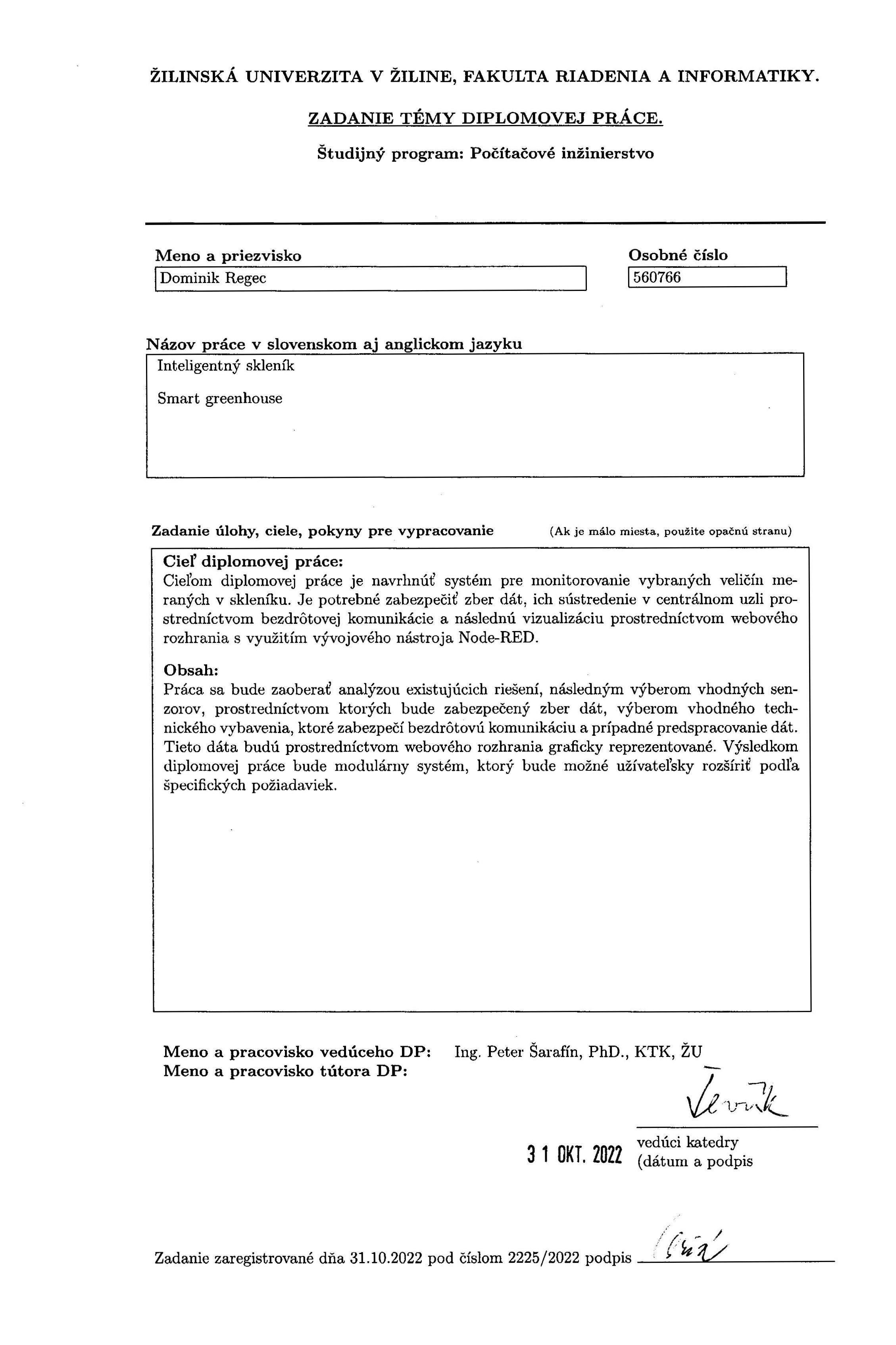
Študijný program: Počítačové inžinierstvo

Študijný odbor: Informatika

Školiace pracovisko: Žilinská univerzita v Žiline,

Vedúci diplomovej práce: Ing. Peter Šarafín, PhD.

Žilina 2023



Čestné vyhlásenie

Vyhlasujem, že som zadanú diplomovú prácu vypracoval samostatne, pod odborným vedením vedúceho práce/školiteľa a používal som len literatúru uvedenú v práci.

Žilina 3. apríla 2023

podpis

Poďakovanie

(Poďakovanie nie je povinná časť záverečnej práce.)

Abstrakt

REGEC, Dominik: Inteligentný skleník [Diplomová práca] **-** Žilinská univerzita v Žiline, Fakulta riadenia a informatiky, Katedra technickej kybernetiky. – Školiteľ: [Ing. Peter Šarafín, PhD.](mailto:Peter.Sarafin@fri.uniza.sk) ŽU v Žiline, 2023. – 68 s.

Diplomová práca je venovaná využitiu IoT systémov a bezdrôtových technológii pri návrhu modulárneho systému inteligentného skleníka. Cieľom práce je navrhnúť systém, ktorý bude obsahovať sadu senzorov a je na používateľovi a konfigurácii systému aké veličiny budú merane. Namerané veličiny budú následne distribuované do centrálneho uzla, kde sú spracované a odoslané na server. Server dáta spracuje a používateľovi je poskytnuté používateľské rozhranie slúžiace na monitoring inteligentného skleníka. Úvod práce je venovaný analýze požiadaviek systému a popis teoretických informácii. Kapitola 3 a 4 je venovaná návrhu a implementácii systému a záverom práce je testovanie systému. Významom práce je poskytnúť používateľovi systém, ktorý mu zabezpečí prostredie na pestovanie rastlín bez fyzického úsilia.

Kľúčové slová: IoT, Bluetooth, Wi-Fi, ESP, MQTT, Docker

Abstract

REGEC, Dominik: Smart greenhouse [The diploma thesis] – University of Žilina, Faculty of Management Science and Informatics, Department of Technical Cybernetic. – Consultant: [Ing. Peter Šarafín, PhD.](mailto:Peter.Sarafin@fri.uniza.sk) Žilina: FRI University of Žilina, 2023. – 68 p.

The diploma thesis is dedicated to the use of IoT systems and wireless technology in the design of a modular system of an intelligent greenhouse. The goal of the work is to design a system that will contain a set of sensors and it is up to the user and the system configuration what quantities what will be measured. The measured quantities will then be distributed to the central node, where they are processed and sent to the server. The server processes the data and the user is provided with a user interface for monitoring the smart greenhouse. The introduction of the work is devoted to the analysis of system requirements and the description of theoretical information. Chapters 3 and 4 are devoted to the design and implementation of the system, and the conclusion is the testing of the system. The meaning of the work is to provide the user with a system that will provide him with an environment for growing plants without physical effort.

**Keywords:** IoT, Bluetooth, Wi-Fi, ESP, MQTT, Docker

Obsah

[Úvod 20](#_Toc131867221)

[1 Cieľ práce 22](#_Toc131867222)

[2 Analýza 23](#_Toc131867223)

[2.1 Analýza bezdrôtových technológii 23](#_Toc131867224)

[2.1.1 Wi-Fi 23](#_Toc131867225)

[2.1.2 Bluetooth 24](#_Toc131867226)

[2.1.2.1 Bluetooth Classic a Bluetooth Low Energy 24](#_Toc131867227)

[2.1.2.2 Bluetooth rozsah 25](#_Toc131867228)

[2.1.2.3 Princíp fungovania Bluetooth technológie 26](#_Toc131867229)

[2.1.3 Porovnanie bezdrôtových technológii 26](#_Toc131867230)

[2.2 Analýza technických komponentov 27](#_Toc131867231)

[2.2.1 ESP-WROOM-32 27](#_Toc131867232)

[2.2.2 SHT41 27](#_Toc131867233)

[2.2.3 SCD41 27](#_Toc131867234)

[2.3 Databázové systémy 28](#_Toc131867235)

[2.3.1 SQL 28](#_Toc131867236)

[2.3.2 MySQL 28](#_Toc131867237)

[2.3.2.1 Výhody MySQL 29](#_Toc131867238)

[2.3.3 NoSQL 29](#_Toc131867239)

[2.3.3.1 História NoSQL databázy 29](#_Toc131867240)

[2.3.3.2 Typy NoSQL databázy 30](#_Toc131867241)

[2.3.4 NoSQL vs. SQL databázy 30](#_Toc131867242)

[3 Návrh systému 31](#_Toc131867243)

[3.1 Návrh klienta 32](#_Toc131867244)

[3.2 Návrh servera 33](#_Toc131867245)

[3.3 Princíp fungovania celého systému 34](#_Toc131867246)

[4 implementacia systemu 35](#_Toc131867247)

[4.1 ESP Klient 36](#_Toc131867248)

[4.2 Riadiaci člen ESP-WROOM-32 37](#_Toc131867249)

[4.2.1 Podsystém na riadenie zavlažovanie rastlín 37](#_Toc131867250)

[4.2.2 Podsystém na riadenie odvetrávania 38](#_Toc131867251)

[4.2.3 Softvérový modul na riadenie akčných členov 39](#_Toc131867252)

[4.3 Podsystém indikácie stavu 40](#_Toc131867253)

[4.4 Implementácia ESP klienta 41](#_Toc131867254)

[4.5 Implementácia riadiaceho člena 42](#_Toc131867255)

[4.6 Implementácia technológie Bluetooth 43](#_Toc131867256)

[4.6.1 Bluetooth GATT profil 43](#_Toc131867257)

[4.6.2 Bluetooth inicializácia BLE servera 44](#_Toc131867258)

[4.6.3 Princíp fungovanie Bluetooth objektov na BLE servery 45](#_Toc131867259)

[4.6.4 Bluetooth inicializácia BLE klienta 46](#_Toc131867260)

[4.6.5 Princíp fungovanie Bluetooth objektov na BLE klientovi 46](#_Toc131867261)

[4.6.6 Strata spojenia s BLE serverom 47](#_Toc131867262)

[4.7 Konfigurácia klienta 48](#_Toc131867263)

[4.7.1 Konfigurácia ESP klienta 48](#_Toc131867264)

[4.7.2 Konfigurácia riadiaceho člena 49](#_Toc131867265)

[4.8 Zber údajov 49](#_Toc131867266)

[4.9 Prenos dát do centrálneho uzla 50](#_Toc131867267)

[4.10 Spracovanie Bluetooth dát riadiacim členom 51](#_Toc131867268)

[4.11 Manažér synchronizácie času 52](#_Toc131867269)

[4.11.1 Architektúra protokolu NTP 53](#_Toc131867270)

[4.12 MQTT 53](#_Toc131867271)

[4.12.1 Protokol MQTT 53](#_Toc131867272)

[4.12.2 Model pub/sub 54](#_Toc131867273)

[4.12.3 Filtrácia správ 55](#_Toc131867274)

[4.12.4 MQTT Broker a MQTT Klient 55](#_Toc131867275)

[4.12.5 MQTT Pripojenie 56](#_Toc131867276)

[4.13 Implementácia MQTT klienta 57](#_Toc131867277)

[4.13.1 Vzdialené ovládanie akčných členov 57](#_Toc131867278)

[4.14 Implementácia servera 58](#_Toc131867279)

[4.14.1 Vytváranie sieti v rámci platformy Docker 59](#_Toc131867280)

[4.14.2 Komunikácia medzi kontajnermi a vonkajším svetom 60](#_Toc131867281)

[4.15 Sieťovanie na platforme Docker v navrhnutom systéme 60](#_Toc131867282)

[4.16 NTP server 62](#_Toc131867283)

[4.17 Používateľské webové rozhranie 62](#_Toc131867284)

[4.17.1 Používateľské webové rozhranie skleníka 63](#_Toc131867285)

[4.17.2 Archív nameraných dát 63](#_Toc131867286)

[4.17.2.1 Filtrácia archívnych dát 64](#_Toc131867287)

[4.18 Databázový systém 66](#_Toc131867288)

[5 Testovanie 67](#_Toc131867289)

[5.1 Testovanie akčných členov 67](#_Toc131867290)

[5.2 Manuálne testovanie softvérových modulov 68](#_Toc131867291)

[Záver 71](#_Toc131867292)

Zoznam obrázkov

Obr. 2.1 – Základný princíp systému, využívaný pri analýze systému 23

Obr. 3.1 – Bloková schéma navrhnutého systému 32

Obr. 3.2 – Sekvenčný diagram registrácie klienta 35

Obr. 4.1 – Schematické zapojenie senzorov 36

Obr. 4.2 – Schematické zapojenie podsystému na ovládanie závlahy 38

Obr. 4.3 – Schematické zapojenie ovládača ULN2003 [16] 38

Obr. 4.4 – UML diagram softvérového modulu na riadenia akčných členov 39

Obr. 4.5 – Schematické zapojenie RGB diódy 40

Obr. 4.6 – UML diagram softvérového komponentu StatusIndicator 40

Obr. 4.7 – Príklad GATT profilu [17] 44

Obr. 4.8 – UML diagram reprezentujúci vzťah medzi objektami 45

Obr. 4.9 – UML diagram modulu: ConnectionHolder 47

Obr. 4.10 – Záznam z konfigurácie ESP klienta 49

Obr. 4.11 – Princíp modelu pub/sub [21] 54

Obr. 4.12 – MQTT pripájacia sprava [22] 56

Obr. 4.13 – Sieťovanie v navrhnutom systéme 61

Obr. 4.14 – Používateľské rozhranie zobrazujúce hodnoty za posledných 30 minút 64

Obr. 4.15 – Používateľom definovaný rozsah zobrazenia údajov 66

Obr. 5.1 – 3D model systému na manipuláciu s oknom 68

Zoznam tabuliek

Tab. 2.1 – Popis základných rozdielov medzi verziami bluetooth technológie [3] 25

Tab. 2.2 – SQL vs. NoSQL [14] 31

Tab. 4.1 – Definícia farby RGB diódy 41

Tab. 4.2 –Dátová štruktúra bluetooth dát 50

Tab. 4.3 – ID Bajt 51

Tab. 4.4 – Bajt definujúci dynamickú časť 51

Tab. 4.5 – Význam stavového kódu 57

Tab. 4.6 – Zoznam kontajnerov 61

Tab. 4.7 – Popis funkcie tlačidiel v používateľskom rozhraní 65

Zoznam skratiek

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Skratka** | **Anglický význam** | | | **Slovenský význam** |
| ADC | Analogue to Digital Converter | | | Analógovo - digitálny prevodník |
| AP | Access point | | | Prístupový bod |
| BLE | Bluetooth Low Energy | | | Verzia Bluetooth s nízkou spotrebou energie |
| CO2 | Carbon dioxide | | | Oxid uhličitý |
| FHSS | Frequency-hopping spread spectrum | | | Metoda prenosu s preskakovaním frekvencie |
| GAP | Generic Access Profile | | | Profil s generickým prístupom |
| GATT | Generic Attribute Profile | | | Profil s generickým atribútom |
| GFSK | Gaussian Frequency Shift Keying | | | Gausova frekvenčná modulácia s posunom |
| *IoT* | Internet of things | | | Internet veci |
| *ISM* | Industrial, Scientific, Medical | | | Priemyselné, Vedecké, Zdravotnícke |
| *NAT* | Network address translation | | | Metoda prekladania sieťových adries |
| *NTP* | Network Time Protocol | | | Sieťový časový protokol |
| *SIG* | Špeciál Interest Group | | | Skupina so špeciálnym záujmom |
|  |  | | |  |
|  | |  |  | |
|  | |  |  | |

Úvod

Internet of things (IoT) v slovenčine nazývané ako internet veci je pojem, s ktorým sa v dnešnom svete stretávame čoraz častejšie. S IoT sa dnes už môžeme stretnúť takmer v každom jednom odvetí od stavebníctva, zdravotníctva až po nejaké zariadenia v domácnosti. Internet veci v informatike označuje systém, v ktorom je definovaná sieť prepojených zariadení, ktoré medzi sebou komunikujú. Spracovaná téma diplomovej práce je súčasťou tejto kategórie.

Spracovaná diplomová práca sa venuje návrhu modulárneho systému inteligentného skleníka. Hlavným cieľom je navrhnúť systém, ktorého úlohou je zber dát zo senzorov ich distribúcia a následná vizualizácia v používateľskom rozhraní. Špecifickou vlastnosťou systému je jeho variabilita pri meraní dát vo vnútri alebo vo vonkajšom prostredí skleníka na základe kritérií používateľa. Súčasťou práce je aj analýza bezdrôtových technológii, ktoré zohrávajú dôležitú úlohu pri distribúcii dát medzi zariadeniami, ako aj zasielanie dát na server k ich následnej vizuálnej reprezentácii.

Navrhnutý systém poskytuje používateľovi merať teplotu, vlhkosť vzduchu, vlhkosť pôdy a oxid uhličitý. Namerané dáta sú distribuované technológiou Bluetooth smerom ku centrálnemu uzlu, ktorý dáta spracuje a zasielala ich na server, kde sú graficky reprezentovane. Systém na základe nameraných dát taktiež ovláda závlahu inteligentného skleníka a odvetrávaci systém. Navrhnutý systém obsahuje používateľské rozhranie, ktoré poskytuje používateľovi prehľad o aktuálne nameraných hodnotách, ako aj prehľad o historických dátach.

Významom diplomovej práce je navrhnúť používateľovi systém, ktorý by mu poskytol prostredie na pestovanie rôznych druhov rastlín a zároveň ho odbremenil od povinnosti, ako je zavlažovanie týchto rastlín a podobné úkony, ktoré sú nevyhnutné pre správny rast rastliny. Nie menej dôležitým zmyslom práce je aj využitie IoT systémov a bezdrôtových technológii, ako je bluetooth a Wi-Fi na uľahčenie práce používateľa a poskytnutie mu rozhranie, vďaka ktorému všetky úkony spojené so skleníkom bude možné vykonávať vzdialene, bez fyzického kontaktu s monitoringom všetkých dostupných fyzikálnych veličín.

Prvá kapitola v rámci štruktúry práce je venovaná popisu hlavného cieľa diplomovej práce. Detailný opis cieľu ako aj popis jednotlivých čiastočných cieľov k dosiahnutiu hlavného. Druha kapitola je venovaná analýze a opisu technických komponentov. Analyzujú sa bezdrôtové technológie s cieľom vybrať najvhodnejšiu na prenos dát medzi zariadeniami v rámci skleníka. Časť kapitoly sa taktiež zaoberá aj opisom technických komponentov vhodných na využitie pri implementácii systému. Nasledujúca tretia kapitola už popisuje priamo návrh celkového systému. Návrh klienta, servera so záverom popisu funkčnosti kompletného systému. Kapitola implementácie je štvrtou kapitolou a zaoberá sa popisom softvérovej a hardvérovej implementácie navrhovaného systému. Taktiež obsahuje popis inicializácie a funkcie jednotlivých softvérových modulov v rámci systému. Záverom implementačnej kapitoly je opis používateľského rozhrania. Poslednou kapitolou v práci je kapitola zameraná na testovanie. Obsahom kapitoly je opis prvého zostrojeného prototypu určeného k testovaniu navrhnutého systému ako aj popis jednotlivých testovacích procesov.

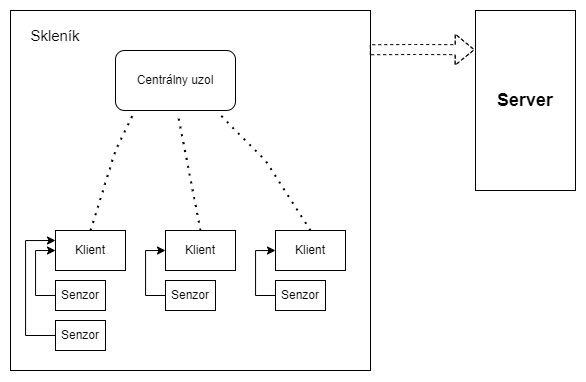
# Cieľ práce

Cieľom diplomovej práce je navrhnúť modulárny systém pre monitorovanie vybraných veličín meraných v skleníku. Používateľovi je potrebné zabezpečiť možnosť konfigurácie pri meraní rôznych veličín na základe jeho preferencii, z predom definovanej množiny. Ďalej je potrebné zabezpečiť zber dát a ich sústredenie v centrálnom uzly prostredníctvom bezdrôtovej komunikačnej technológie a ich následná vizualizácia prostredníctvom webového rozhrania s využitím vývojového nástroja Node-RED.

Čiastkové ciele pre splnenie hlavného cieľa:

* Analýza bezdrôtových technológii vhodných na komunikáciu s centrálnym uzlom
* Vyber senzorov na meranie veličín spoločne s výberom vhodného technického vybavenia na komunikáciu so senzormi
* Návrh systému pre klienta
* Návrh systému pre server
* Implementácia prvej verzie navrhnutého systému
* Sústredenie dát do centrálneho uzla s následným prenosom na server
* Vizualizácia dát
* Testovanie a rozbor výsledkov

# Analýza

Druha kapitola je venovaná analýze požiadaviek systému. Cieľom analýzy bezdrôtových technológii je výber najvhodnejšej na komunikáciu klientov s centrálnym uzlom. Analýza technických komponentov je zameraná na opis mikrokontroléra ESP-WROOM-32 a popis senzorov, ktoré sú neskôr využívané. Záver kapitoly je zameraný na rozbor databázových systémov, ktorý zohral kľúčovú úlohu pri výbere databázového systému v navrhovanom systéme.

Obr. 2.1 – Základný princíp systému, využívaný pri analýze systému

## Analýza bezdrôtových technológii

Kapitola Analýza bezdrôtových technológii obsahuje technické informácie o technológiách Wi-Fi a Bluetooth. Ich využiteľnosť v praxi a porovnanie jednotlivých technológii z pohľadu ich vlastnosti a využiteľnosti pre navrhovaný systém.

### Wi-Fi

Wi-Fi je bezdrôtová technológia, ktorá umožňuje zariadeniam pripojenie do internetu. Dôležitou súčasťou Wi-Fi siete je bezdrôtový smerovač. Smerovač zabezpečuje internetové pripojenie a zariadenie, ktoré pristupuje do Wi-Fi siete sa pripája k spomínanému smerovaču.

Štandard IEEE 802.11 definuje protokoly, ktoré umožňujú bezdrôtovú komunikáciu so súčasnými zariadeniami podporujúce technológiu Wi-Fi. Medzi zaradenia patria bezdrôtové smerovače a bezdrôtové prístupové body (AP), ktoré ale podporujú rôzne štandardy IEEE.

Postupom času boli jednotlivé štandardy ratifikované a jednotlivé štandardy sa od seba odlišujú poskytujúcou šírkou pásma, fungujúcou na rôznych frekvenciách a podporujúce rôzny počet kanálov. [1]

### Bluetooth

Bluetooth je vo všeobecnosti braná ako bezdrôtová komunikačná technológia krátkeho dosahu. Technológia umožňujúca elektronickým zariadeniam prenášať alebo prijímať dáta bezdrôtovo na krátke vzdialenosti. Viac o rozsahu technológie v podkapitole 2.1.2.2.

Konvencia mien pre technológiu Bluetooth sa od svojho prvého publikovania v roku 1999 zmenila a je potrebné spomenúť, že Bluetooth technológia pred rokom 2010 je referenciou na klasicky Bluetooth, keďže variant Bluetooth s nízkou spotrebou (s označením Bluetooth LE) ešte neexistoval. Skupina Bluetooth SIG (Bluetooth Special Interest Group) v roku 2010 publikovala Bluetooth 4.0 špecifikáciu, ktorá obsahovala klasicky Bluetooth spoločne s Bluetooth LE. [2]

#### Bluetooth Classic a Bluetooth Low Energy

**Bluetooth Classic** (klasicky Bluetooth), taktiež nazývaný aj ako Bluetooth Basic Rate/Enhanced Data Rate (BR/EDR), streamuje dáta v 2.4 GHz (ISM) nelicencovateľnom priemyselnom, vedeckom a zdravotníckom frekvenčnom rozsahu cez 79 kanálov. Bluetooth Classic je využívaný hlavne na streamovanie audia a stal sa štandardom pre bezdrôtové reproduktory.

Do systémov s požiadavkou nízkej spotreby bol navrhovaný **Bluetooth Low Energy** (Bluetooth LE). Prenos dát je možné vykonávať cez 40 kanálov v rovnakom 2.4 GHz frekvenčnom pásme, ako je to spomínané pri Bluetooth Classic. Bluetooth LE podporuje viacero komunikačných topológii a rozširuje o ďalšie topológie ako broadcast a taktiež topológiou mesh. Rozšírenie Bluetooth technológie o spomínané topológie poskytuje vývojárom vytvárať spoľahlivé a škálovateľné siete zariadení. Donedávna bol Bluetooth LE používaný hlavne kvôli jeho komunikačným schopnostiam, no dnes je široko využívaný aj ako technológia na detekciu zariadení. V súčasnosti Bluetooth LE disponuje funkciou, ktorá umožňuje zariadeniu určiť prítomnosť, vzdialenosť alebo smer iného zariadenia. Nasledujúca tabuľka porovnávaná Bluetooth Low Energy a Bluetooth Classic. [3]

Tab. .1 – Popis základných rozdielov medzi verziami bluetooth technológie [3]

|  |  |  |
| --- | --- | --- |
|  | Bluetooth Low Energy | Bluetooth Classic |
| Frekvenčný rozsah | 2.4 GHz ISM pásmo | 2.4 GHz ISM pásmo |
| Kanál | 40 kanálov s 2 MHz rozostupom | 79 kanálov s 1 MHz rozostupom |
| Využitie kanálov | FHSS | FHSS |
| Modulácia | GFSK | GFSK, π/4 DQPSK, 8DPSK |
| Prenosová rýchlosť | LE 2M PHY: 2 Mb/s LE 1M PHY: 1 Mb/s LE PHY (S=2): 500 Kb/s LE PHY (S=8): 125 Kb/s | EDR PHY (8DPSK): 3 Mb/s EDR PHY (π/4 DQPSK): 2 Mb/s BR PHY (GFSK): 1 Mb/s |
| Komunikačné topológie | Point-to-Point  Broadcast  Mesh | Point-to-Point |

#### Bluetooth rozsah

Na začiatku podkapitoly 2.1.2 je uvedená informácia, kde Bluetooth technológia je vo všeobecnosti braná ako technológia na krátku vzdialenosť, avšak tato technológia bola navrhnutá na širokú škálu dosiahnuteľných rozsahov. Flexibilita Bluetooth technológie poskytuje vývojárovi nastavenie rozsahu komunikácie medzi dvoma zariadeniami podľa predom definovaných kritérií. Nasledujúce faktory ovplyvňujú dosah spoľahlivého pripojenia:

* **Rádiové spektrum** – Rozsah rádiového spektra je od 30 Hz do 300 GHz a platí čím je frekvencia nižšia, tým je dosah dlhší. Frekvenčne pásmo Bluetooth technológie je 2.4 GHz, čo predstavuje rovnováhu medzi dosahom a priepustnosťou.
* **PHY (Physical layer)** – Fyzická vrstva bezdrôtovej technológii, ktorá určuje typ modulácie a ďalšie techniky využívané na odosielanie dát cez rádiofrekvenčné pásmo. V rámci toho je zahrnutý počet dostupných kanálov a ich efektívne využitie, korekcia chýb, ochranné zariadenia na zamedzenie rušenia a ďalšie.
* **Citlivosť prijímača** – Vyjadruje minimálnu hodnotu vstupného signálu na prijímači, pri ktorej je stále možné demodulovať signál a udržiavať spojenie. Bluetooth technológia špecifikuje, že v závislosti od použitej PHY je nevyhnutne, aby zariadenie bolo schopné dosiahnuť minimálnu citlivosť prijímača -70 dBm až -80 dBm.
* **Výkon vysielača** – Vyber výkonu vysielača je konštrukčným kompromisom medzi dosahom a spotrebou energie. Zvýšenie efektívneho dosahu je možné docieliť zvýšením výkonu vysielača no len na úkor vyššej spotreby zariadenia. Bluetooth technológia podporuje vysielacie výkony od -20 dBm (0.01 mW) do +20 dBm (100mW).
* **Zisk antény** – Primárnym cieľom antény je konverzia elektromagnetickej energie na elektrickú a naopak. Účinnosť antény pre vysielanie aj prijímanie signálu je možné ovplyvniť umiestením antény, dizajnom, ale aj jej rozmermi.
* **Utlm elektromagnetických vĺn** – Závisí od prostredia, v ktorom dochádza ku šíreniu signálu a jeho následnému zníženiu. [4]

#### Princíp fungovania Bluetooth technológie

Pri zariadeniach využívajúcich klasicky Bluetooth je potrebné aby boli dane zariadenia sparovane. Výsledok procesu párovania je dôvera medzi zariadeniami a umožnenie následnú výmenu kryptovaných dát. V prípade, keď sa 2 zariadenia dostanú do vzájomného kontaktu (dosahu), uskutočni sa overovacia konverzácia s výsledkom o dôvere. Konverzácia je uskutočňovaná automaticky a používateľ nemusí stláčať žiadne tlačidlo ani zadávať žiaden špecifický príkaz.

Zariadenia s Bluetooth LE môžu využívať techniku párovania opísanú v odseku vyššie no nie všetky produkty to vyžadujú. Zariadenie, ktorého cieľom je byt odhalené vysiela broadcastovú správu. Broadcastová správa obsahuje užitočné informácie o zariadení. Proces vysielania správy sa nazýva „advertising“. Ostatné zariadenie s rovnakou technológiou si skenovacím procesom dane zariadenie nájde a pripojí sa k nemu. [5]

### Porovnanie bezdrôtových technológii

Pri porovnávaní technológii z pohľadu využiteľnosti pre navrhovaný systém bolo brané do úvahy niekoľko faktorov, ktoré sú dôležité pri návrhu systému, ako aj pri jeho budúcich úpravách a vylepšeniach.

Zoznam faktorov:

* **Rýchlosť** – Prenosová rýchlosť technológie Wi-Fi je mnohonásobné vyššia (maximálna rýchlosť závisí od štandardu) v porovnaní s Bluetooth.
* **Množstvo prenášaných dát** – Bluetooth je často využívaná na posielanie malých objemov dát (numerické hodnoty zo senzorov) v porovnaní s Wi-Fi, ktorá je lepšia na prenos väčších dátových súborov.
* **Rozsah** – Bluetooth ma zvyčajne menší rozsah (viac o rozsahu Bluetooth a jeho možnostiach konfigurácie rozsahu v kapitole 2.1.2.2) v porovnaní s Wi-Fi.
* **Spotreba energie** – Bluetooth LE bol špeciálne navrhovaný na menšiu spotrebu ako Wi-Fi. [6]

## Analýza technických komponentov

Obsahom kapitoly je popis mikroprocesora ESP32-WROOM-32, popis senzorov na meranie veličín, ako je teplota, vlhkosť vzduchu a CO2.

### ESP-WROOM-32

ESP-WROOM-32 je výkonný mikrokontrolér podporujúci technológie Wi-Fi, Bluetooth a Bluetooth Low Energy so zameraním na širokú škálu rôznych aplikácii, od siete senzorov s požiadavkou na nízku spotrebu až po aplikácie s náročnejšími úlohami. Jadro mikrokontroléra je tvorené čipom ESP32-D0WDQ6, ktorý obsahuje dve jadra s možnosťou individuálneho riadenia. Ďalšou z funkcii mikrokontroléra je poskytnutie používateľovi nastaviť frekvenciu hodín od 80 MHz do 240 MHz. Používateľ taktiež dokáže vypnúť CPU a na monitorovanie periférii využiť koprocesor s nízkou spotrebou. Súčasťou mikrokontroléra je aj súbor rôznych periférii od Hallových senzorov, kapacitných senzorov, rozhrania SD karty až po komunikačné rozhrania SPI, UART a I2C. [7]

### SHT41

Senzor SHT41 je snímač na meranie relatívnej vlhkosti a teploty vzduchu s možnosťou merania v rôznych triedach presnosti. Komunikácia so senzorom je zabezpečená rozhraním I2C, kde je predom predkonfigurovaná I2C adresa. Senzor taktiež obsahuje interný výkonový ohrievač, ktorý dokáže pracovať v troch režimoch a senzoru umožňuje pracovať aj v náročnejších prostrediach. [8]

### SCD41

Senzor SCD41 je súčasťou ďalšej generácie miniatúrnych CO2 senzorov spoločnosti Sensirion. Princíp snímania je založený na fotoakustickej NDIR a technológiách PASens® a CMOSens® patentovaných spoločnosťou Sensirion, ktoré poskytujú vysokú presnosť. Súčasťou senzora je aj vstavaný senzor SHT41 na meranie relatívnej vlhkosti a teploty vzduchu. [9]

## Databázové systémy

Podkapitola Databázové systémy je venovaná analýze relačných a nerelačných databáz, obsahuje charakteristické vlastnosti databáz, ich výhody a nevýhody a záverom podkapitoly je ich vzájomné porovnanie.

### SQL

Pred samostatným opisom jednotlivých typov databáz, popisu ich vlastnosti spojené s výhodami a nevýhodami je nevyhnutné venovať jednu podkapitolu samotnému programovaciemu jazyku SQL.

**SQL** (angl. **S**tructured **Q**uery **L**anguage) je štandardný programovací jazyk používaný hlavné na extrakciu, správu a manipulovanie s dátami uloženými v relačnej databáze. Z tohto dôvodu je SQL často označovaný ako databázový jazyk určený na vykonávanie činnosti v databázach, ktoré obsahujú tabuľky vytvorené zo stĺpcov a riadkov.

Proces pracovania s údajmi v databáze funguje na báze dotazov. Po napísaní a spustení dotazu sa dotaz spracováva procesorom na kontrolu jazyka. Spracovanie dotazu prebieha v troch fázach:

* **Syntaktická analýza** – Proces, pri ktorom prebieha krížová kontrola syntaxi zadaného dotazu.
* **Overovanie sémantiky** – Overenie sémantiky dotazu pred jeho vykonaním.
* **Optimalizácia** – Poslednou fázou je optimalizácia, pri ktorom sa vygeneruje vykonávací plán. Primárnym cieľom je definovať najefektívnejší plán s minimálnym časom na jeho vykonanie. V konečnom dôsledku to znamená, čím je kratší čas odozvy na dotaz, tým sú výsledky vyhovujúcejšie. [10]

### MySQL

Všeobecnou definíciou MySQL je open-source[[1]](#footnote-1) relačný databázový systém s modelom klient – server. Databáza je základným dátovým úložiskom pre takmer všetky softvérové aplikácie a MySQL, ako relačná databáza ukladá údaje do samostatných tabuliek. Štruktúra databázy sa skladá z fyzických súborov, ktoré sú optimalizované pre rýchlosť.

Objekty ako dátové tabuľky, jednotlivé zobrazenia, stĺpce a riadky sú súčasťou logického dátového modelu databázy a poskytujú flexibilné programovacie prostredie, čo spoločné s riadiacimi pravidlami zabezpečí konzistentnosť dát.

Klient – server systém pozostáva z viacvláknového SQL servera, ktorý podporuje niekoľko rôznych klientskych programov a knižníc, administratívne nástroje a širokú škálu rozhraní pre programovanie aplikácii.

#### Výhody MySQL

Napriek faktu, že MySQL je ešte stále v priebehu vývoju, poskytuje používateľovi širokú škálu použiteľných funkcií a je definovaná ako rýchla, spoľahlivá, škálovateľná a jednoduchá na používanie. Prvotný vývoj bol sústredený na rýchlu prácu s veľkými databázami a už mnoho rokov sa využíva vo vysoko náročných prostrediach.

Kľúčové benefity MySQL databázy sú:

* **Jednoduché používanie** – Vývojárovi stačí nainštalovať MySQL a v priebehu niekoľkých minútach je databáza pripravená na používanie.
* **Spoľahlivosť** – Testovanie databázy na širokej škále scenárov už viac ako 25 rokov deklaruje spoľahlivosť, ktorú potvrdzujú aj najväčšie svetové firmy.
* **Škálovateľnosť** – MySQL umožňuje skladovateľnosť tak, aby spĺňala požiadavky aplikácii.
* **Bezpečnosť** – Bezpečnosť uchovaných dát zahŕňa ochranu v súlade s normami Európskej Únie o ochrane údajov, štandard bezpečnosti údajov o platobných kartách a mnohé ďalšie. [11]

### NoSQL

NoSQL databázy sú netabuľkové databázy s rozdielnym ukladaním údajov v porovnaní s relačnými databázami. Existuje niekoľko rôznych typov NoSQL databázy, ktoré ale budú popísané nižšie.

#### História NoSQL databázy

História NoSQL databázy siaha do roku 2000, keď náklady na úložisko sa rapídne znížili. Pred rokom 2000 boli vytvárané zložité dátové modely, ktoré boli ale náročné na spravovanie, kde ich hlavným cieľom bolo vyhnúť sa duplicite dát. S klesajúcimi nákladmi sa zvyšovala kvantita dát, ktoré aplikácie potrebovali na archiváciu a dopytovanie. Prichádzajúce dáta sa odlišovali vo formáte a veľkosti, čo vyvrcholilo nakoniec do nemožnej úlohy, a to vopred definovať databázovú schému. Jednou z vlastnosti NoSQL databázy je možnosť uložiť obrovské množstvo neštruktúrovaných dát, čo poskytlo vývojárov flexibilitu pri vývoji aplikácii. [12]

#### Typy NoSQL databázy

NoSQL je termín definujúci všetky alternatívne systémy mimo SQL databáz a štyri hlavne type NoSQL databázy:

* **Databáza dokumentov –** Databáza ukladá dáta v dokumentoch formátu JSON, BSON alebo XML s možnosťou vnorených dokumentov. Ďalšou vlastnosťou takého typu databázy je pridanie indexovania na jednotlivé prvky pre rýchlejšie vyhľadanie. Ukladať a získavať dokumenty vo forme, ktorá je bližšie dátovým objektom uľahčuje používanie v aplikácii. Obľúbenosť databázy dokumentov tkvie vo flexibilite, ktorú poskytuje vývojárom pri formovaní dátovej štruktúry na základe podmienok aplikácie.
* **Databáza <kľúč - hodnota> –** Jedným z najjednoduchších typov NoSQL databázy je databáza na baze kľúča a hodnoty. Každý prvok v databáze je namapovaný ako par kľuč s príslušnou hodnotou. Takýto typ databázy ma istú podobnosť s relačnou databázou obsahujúcou iba dva stĺpce.
* **Stĺpcová databáza –** Typ databázy, kde sú údaje ukladané a organizované do stĺpcov, ktoré sú často rovnakého dátového typu. Pri takýchto stĺpcoch je využitá efektívnejšia kompresia, čim sa docieli rýchlejšie čítanie dát. Stĺpcová databáza je výrobná na analýzu dát, avšak obrovským nedostatkom je zápis dát, keďže zápisy do stĺpcov vyžadujú niekoľko operácii na disk.
* **Grafová databáza –** Pri grafovej databáze je kladený doraz na vzťah medzi dátovými prvkami. Uzol v grafe je reprezentáciou každého prvku. Značnou výhodou pri grafovej databáze je fakt, že spojenia medzi dátovými prvkami sú taktiež ukladane. Optimalizácia takéhoto typu databázy so spojeniami a samotnými prvkami nám redukuje réžiu, ktorá by bola potrebná pri spájaní viacerých tabuliek v SQL. [13]

### NoSQL vs. SQL databázy

Jednotlivé podkapitoly v sekcii opisujúce databázové systémy čiastočne opísali benefity SQL databáz ako aj výhody NoSQL, avšak súhrn základných rozdielov medzi týmito typmi databáz je popísaný v tabuľke nižšie. [14]

Tab. .2 – SQL vs. NoSQL [14]

|  |  |  |
| --- | --- | --- |
|  | SQL Databáza | NoSQL Databáza |
| Dátový model | Tabuľky s fixným počtom riadkov a stĺpcov | Dokument: JSON dokumenty,  <kľúč - hodnota>: par s kľúčom a priradenou hodnotou  Graf: dátové prvky a jednotlivé spojenia |
| Historicky vývoj | Vyvíjane v 70. rokoch s dôrazom na redukciu duplicity dát | Vyvíjane v začiatkoch roka 2000 s dôrazom na škálovateľnosť |
| Príklady | Oracle, MySQL, Microsoft SQL Server, PostgreSQL | Dokument: MongoDB, CouchDB  <kľúč - hodnota>: Redis, DynamoDB  Graf: Neo4j, Amazon Neptune |
| Primárne využitie | Všeobecné využitie | Dokument: všeobecné využitie  <kľúč - hodnota>: veľké množstvo dát s jednoduchým vyhľadávaním  Graf: analýza vzťahov medzi prepojenými dátami |
| Schéma | Pevná | Flexibilná |
| Škálovateľnosť | Vertikálna | Horizontálna |

# Návrh systému

Navrhnutý systém pozostáva z niekoľkých časti, ktoré budú popísané v tejto kapitole. Je založený na architektúre server – klient. Klienta reprezentuje v tomto kontexte skleník aj so všetkými jeho zložkami, ako je centrálny uzol, závlaha a pod. Nasledujúci obrázok zobrazuje jednotlivé časti celého systému, od ESP klientov, ktorí komunikujú so senzormi až po server, na ktorom je spustená platforma docker.

## Návrh klienta

Obr. 3.1 – Bloková schéma navrhnutého systému

Pevnou súčasťou klienta je jeden centrálny uzol (riadiaci člen). Na riadenie celého skleníka je navrhovaný mikrokontrolér ESP-WROOM-32 pravé kvôli jeho poskytovaným technológiám Wi-Fi a Bluetooth Low Energy. Technológia Wi-Fi poskytne riadiacemu členu konektivitu do internetu a umožni mu tým komunikáciu so serverom. Technológia Bluetooth Low Energy je využívaná v navrhnutom systéme ako rozhranie na bezdrôtovú komunikáciu s ESP klientmi. Ďalšou úlohou riadiaceho člena v systéme je riadenie akčných členov, zavlažovania a odvetrávacieho podsystému.

Nevyhnutou časťou celého systému sú aj ESP klienti, ktorí zabezpečujú komunikáciu so senzormi, spracovanie dát s následnou distribúciou do centrálneho uzla. Hardvérovým komponentom pre ESP klienta je navrhovaný mikrokontrolér ESP-WROOM-32. Mikrokontrolér poskytuje Bluetooth Low Energy, komunikačné rozhranie I2C a ADC prevodník. Spomenuté vlastnosti spĺňajú všetky kritéria na vhodného kandidáta s aplikovaním zariadenia do navrhovaného systému. Aplikované zariadenie využíva rozhranie I2C na komunikáciu so senzormi, ADC prevodník na prevod analógového signálu zo senzora vlhkosti pôdy a po ich spracovaní technológiou Bluetooth LE posiela dáta do centrálneho uzla. Systém poskytuje používateľovi umiestniť nelimitovaný počet klientov (vlastnosť bezdrôtovej technológie Bluetooth LE) do skleníka s možnosťou voľby či dané senzory budú merať veličiny vo vnútri skleníka alebo budú monitorovať veličiny v okolí skleníka. Do systému bola taktiež pridaná možnosť konfigurácie ESP klienta. Možnosť konfigurácie poskytujúca používateľovi možnosť zvoliť si o aké veličiny ma záujem pri meraní.

Každý z členov klienta obsahuje v navrhnutom systéme aj RGB diódu, ktorá vizuálne reprezentuje aktuálny stav člena počas jeho inicializácie ako aj počas neskoršieho behu systému.

## Návrh servera

Na základe požiadaviek systému by mal server disponovať vývojovým nástrojom Node-Red, ktorého úlohou je poskytnutie používateľovi webové rozhranie s vizualizáciou nameraných dát.

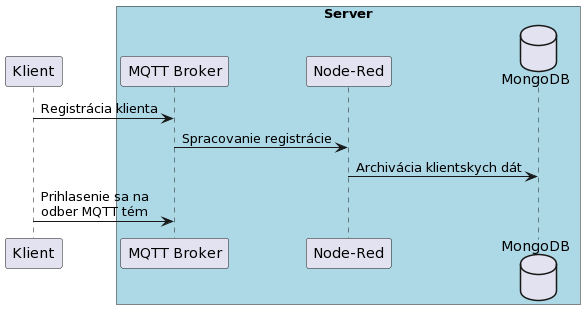
Pri návrhu servera bola braná do úvahy spomenutá požiadavka a keďže je nevyhnutné pri využívaní nastroja aj zabezpečiť automatické spustenie nástroja v prípade reštartu servera alebo zabezpečenie reštartu celého vývojového nástroja, preto do návrhu bola implementovaná platforma Docker. Platforma sa postará o správu aplikácie a používateľ bude mať vždy prístup k webovému rozhraniu v prípade spusteného servera. Ďalšie z aplikácii, ktoré bežia na platforme a sú súčasťou navrhovaného systému sú databázový systém MongoDB typu NoSQL, NTP server a MQTT Broker. NTP server v systéme zohráva dôležitú úlohu pri synchronizácii času v klientskej časti pri spracovaní nameraných dát.

Komunikácia medzi serverom a klientom je s využitím protokolu MQTT, kde spracovanie dát z klientskej časti sa vykonáva vo vývojom nastrojí Node-Red. Ďalšie z úloh nástroja je zaslanie dát na archiváciu a vytvorenie používateľského webového rozhrania. Rozhranie poskytuje používateľovi prehľad o jeho všetkých registrovaných skleníkoch s možnosťou výberu konkrétneho skleníka k zobrazeniu detailných informácii. Pri zobrazení detailných informácii je používateľ oboznámení o aktuálne nameraných hodnotách podľa jeho špecifických kritérií zvolených pri inštalácii systému. Rozhranie v navrhovanom systéme poskytuje používateľovi taktiež možnosť zobrazenia historických záznamov formou grafov.

## Princíp fungovania celého systému

Cely princíp fungovania systému je založený na meraní, zbere a distribúcie dát. Na základe nameraných dát sú vykonávané preddefinované akcie a zozbierané dáta sú poskytnuté používateľovi k vizualizácií.

O spustenie aplikácii na servery sa stará platforma Docker. Docker zabezpečí aby vývojový nástroj Node-Red, MongoDB, NTP server a MQTT Broker boli spustené a pripravené na používanie. Ďalším krokom je inicializácia klienta. Inicializácia riadiaceho člena a všetkých ESP klientov nastáva po pripojení k zdroju napájania. Nasledujúce dva odseky budú popisovať samostatne inicializáciu riadiaceho člena a inicializáciu ESP klienta.

Inicializácia riadiaceho člena začína spustením BLE servera. Spustenie BLE servera zabezpečí propagáciu (vysielanie broadcastovej správy, pozri kapitolu 2.1.2.3) riadiaceho člena. Ďalším krokom je pripojenie do siete s využitím technológie Wi-Fi. Po pripojení je v riadiacom člene inicializovaný MQTT klient, pomocou ktorého je zaslaná sprava o registrácii klienta do systému. Nasledujúci obrázok vizuálne popisuje proces registrácie klienta. Registrácia klienta je spracovaná vo vývojovom nastrojí Node-Red a údaje o klientovi sú uložené do databázy. Po procese inicializácie nastáva bežný chod programu, kde sú periodicky spracované dáta od ESP klientov, následná distribúcia na server, kde ich Node-Red spracuje a archivuje do databázy.

Obr. 3.2 – Sekvenčný diagram registrácie klienta

Inicializácia ESP klienta začína spustením technológie Bluetooth LE a skenovaním prostredia s cieľom nájsť riadiaci člen. Zber údajov je zabezpečený senzormi, ktoré poskytujú meranie teploty a vlhkosti vzduchu, CO2 a meranie vlhkosti v pôde. Primárnou úlohou ESP klientov v rámci skleníka je komunikácia so senzormi. V závislosti od typu senzora je následné aj zvolený typ komunikačného protokolu resp. priame analógové vyčítanie hodnoty. Zber údajov je vykonávaný v pravidelných intervaloch a záleží na konfigurácii aké veličiny budú merané a či sa bude využívať iba I2C zbernica alebo aj prevodník ADC, ktorý je potrebný pri meraní vlhkosti pôdy. Po úspešnom nameraní hodnôt sú dáta pripravené na odoslanie do centrálneho uzla s využitím technológie Bluetooth LE.

# implementacia systemu

Implementáciu systému je možné rozdeliť na dve časti, časť hardvérovú a časť softvérovú. V úvode kapitoly je popísané hardvérové vybavenie systému, ich schematické zapojenia (Príloha A) a popis softvérových modulov, za pomoci ktorých sú hardvérové komponenty riadenie. Zvyšok kapitoly je venovaný popisu softvérovej implementácii systému. Popis inicializácie ESP klientov ako aj riadiaceho člena v rámci skleníka, vytvorenie Bluetooth servera a pripájanie ESP klientov ku Bluetooth serveru. Súčasťou Bluetooth komunikácie je aj opis spracovania chybových stavov, nadviazanie opätovnej komunikácie v prípade náhlej straty spojenia a pod. V kapitole sú taktiež opísané softvérové moduly na nadviazanie spojenia so smerovačom, opätovné pripájanie v prípade prerušenia komunikácie so serverom a opis používateľského webového rozhrania.

## ESP Klient

Hlavný hardvérový prvok z pohľadu klientskej časti je mikrokontrolér ESP32 – WROOM-32. Primárnym cieľom je komunikácia so senzormi. Do navrhnutého systému bola pridaná vlastnosť na výber z ponúkaných senzorov podľa preferencií používateľa. Požívateľ ma možnosť si vybrať senzory SHT41 alebo SCD41 na meranie teploty a vlhkosti vzduchu a kapacitný senzor na meranie vlhkosti pôdy s analógovým výstupom. So senzorom SCD41 je taktiež používateľovi umožnené merať CO2. Senzory komunikujú s mikrokontrolérom pomocou komunikačnej zbernice I2C. Komunikačná zbernica I2C nám poskytuje možnosť v budúcnosti pridávať rôzne iné senzory, ktoré na komunikáciu využívajú komunikačné rozhranie I2C bez nutnosti meniť zapojenie ESP klienta.

Konfigurácia mikrokontroléra opísaná v návrhu systému na komunikáciu s vybranými senzormi je uskutočňovaná softvérovo a bude popísaná neskôr. Nasledujúci obrázok zobrazuje schematické zapojenie jednotlivých senzorov.

## Riadiaci člen ESP-WROOM-32

Obr. 4.1 – Schematické zapojenie senzorov

Riadiacim členom v rámci skleníka je využitý mikrokontrolér ESP-WROOM-32. Zabezpečuje komunikáciu s klientmi s využitím bezdrôtovej technológie Bluetooth LE a taktiež je prvkom, ktorý sa pripája do TCP/IP siete a zasiela namerané hodnoty na server. Riadenie akčných členov je taktiež nesmierne dôležitou súčasťou celého systému, čo ma v réžii taktiež riadiaci člen.

Súčasťou riadiaceho člena sú aj podsystémy na:

* Indikáciu stavu
* Riadenie zavlažovania rastlín
* Riadenie odvetrávania (manipulácia s oknom)

### Podsystém na riadenie zavlažovanie rastlín

Zavlažovanie rastlín v rámci skleníku je zabezpečované vodným potrubím, ktoré je napojené na ponorné čerpadlo. Pracovné napätie čerpadla je od 3V do 6 V. S maximálnym zaťažením čerpadla je spotreba a prietok tekutiny za hodinu dosahuje hodnotu 120 litrov. [15]

Riadiaci obvod zavlažovania obsahuje spínací prvok tranzistor na spínanie regulovaného napätia, keďže riadenie iba výstupným pinom mikrokontroléra nie je dostačujúce. Výstupná hodnota s využitím pinu mikrokontroléra zodpovedá hodnote 3.3V. Tlak pri hodnote 3.3 V nie je dostačujúci a nie je možné rozdistribuovať tekutinu do celého skleníka.

Nasledujúci obrázok zobrazuje schému riadiaceho podsystému na zavlažovanie rastlín. Spínacím prvkom je N-kanálový MOSFET tranzistor BS170. Obvod taktiež obsahuje rezistor R4, ktorý plní úlohu pull-down odporu, LED dióda s ochranným odporom R5 na signalizáciu zopnutého tranzistora a posledným prvkom je potenciometer na nastavenie požadovaného prietoku tekutiny.

### Podsystém na riadenie odvetrávania

Obr. 4.2 – Schematické zapojenie podsystému na ovládanie závlahy

Podsystém na riadenie odvetrávania skleníka spočíva v ovládaní krokového motora. Ovládanie krokového motora 28BYJ-48 je s využitím modulu, ktorý obsahuje ovládač ULN2003.

Ovládač ULN2003 poskytuje priame rozhranie medzi krokovým motorom a mikrokontrolérom. Modul poskytuje 4 vstupy, na ktorých je napojený mikrokontrolér a dva vstupy na napájanie celého modulu a krokového motora. Súčasťou modulu sú aj 4 LED diódy odzrkadľujúce aktuálne aktívne (zopnuté) cievky počas riadenia motora. [16]

### ULN2003 Stepper Driver - Micro RoboticsSoftvérový modul na riadenie akčných členov

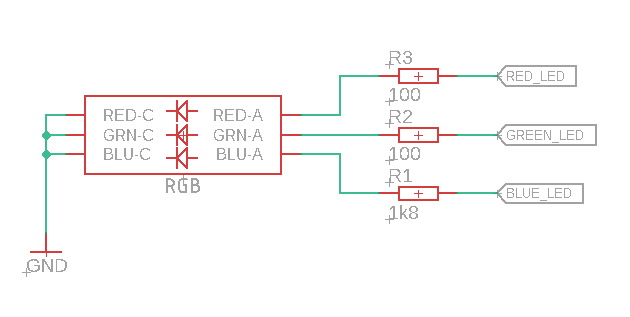
Obr. 4.3 – Schematické zapojenie ovládača ULN2003 [16]

Riadenie akčných členov v rámci systému je zabezpečené softvérovou implementáciou objektu s názvom **ComponentController**. Navrhnutý softvérový modul spĺňa vlastnosti návrhového vzoru **Singleton**. Singleton bol zvolený pravé pre jeho špecifické vlastnosti, ktoré sú potrebné pri riadení akčných členov. Úlohou kontroléra je komunikácia s ovládačmi na spínanie vodného čerpadla (závlahy) a ovládača na riadenie krokového motora (ventilácie). Využitím rozhrania objektu je taktiež možné zistiť aktuálny stav akčných členov.

Inicializácia objektu pozostáva z inicializácie ovládačov pre krokový motor a ponorné čerpadlo. Obidva ovládače vyžadujú konkrétne piny, na ktorých sú akčné členy fyzicky zapojené. Definícia týchto pinov je vykonávaná počas konfigurácie riadiaceho člena. Riadenie vodného čerpadla je vykonávané len s riadením logickej úrovne pinu mikrokontroléra. Pin je inicializovaný ako  výstupný s pull-down odporom a zakázaným prerušením. Pri riadení krokového motora je potrebné ovládať 4 piny mikrokontroléra, ktoré sú taktiež definované v procese konfigurácie. Ovládanie pinov je vykonávané v sekvencii polovičných krokov pre dosiahnutie čo najplynulejšieho pohybu motora.

## Obrázok, na ktorom je stôl Automaticky generovaný popisPodsystém indikácie stavu

Obr. 4.4 – UML diagram softvérového modulu na riadenia akčných členov

Podsystém na indikáciu stavu plní úlohu svetelnej indikácie aktuálneho stavu členov (riadiaci člen, ESP klienti) umiestnených v klientskej časti systému. Na svetelnú indikáciou je použitá RGB dióda. Nasledujúci obrázok zobrazuje schematické zapojenie RGB diódy aj s použitými odpormi.

Obr. 4.5 – Schematické zapojenie RGB diódy

Softvérové ovládanie podsystému je s využitím modulu s názvom **StatusIndicator**. Modul je súčasťou systému riadiaceho člena ako aj ESP klientov. Pri návrhu softvérového modulu bol využitý návrhový vzor Singleton. Nasledujúci obrázok zobrazuje UML schému modulu aj s prepojením na virtuálny RGB objekt reprezentujúci fyzickú RGB diódu.

Obrázok, na ktorom je stôl

Automaticky generovaný popisRozhranie modulu poskytuje dve metódy na manipulácia s diódou. Metódu na vymazanie indikácie aktuálneho stavu. Pri volaní metódy CleanState je dióda vypnutá a nereprezentuje aktuálny stav systému. Druhou metódou je nastavenie stavu RGB diódy podľa prijatého parametru, ktorý reprezentuje aktuálny stav zariadenia. Stav zariadenia sa v čase mení, a preto nasledujúca tabuľka vyjadruje správanie RGB diódy pri rôznych stavoch:

Obr. 4.6 – UML diagram softvérového komponentu StatusIndicator

Tab. 4.1 – Definícia farby RGB diódy

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stav zariadenia** | **Farba** | **Správanie** | **Počet opakovaní** | **Trvanie** |
| Inicializácia Bluetooth bola úspešná | Zelená | Blikanie | 3 | 200 |
| Inicializácia Bluetooth bola neúspešná | Červená | Svietenie | - | - |
| ESP klient nie je pripojený ku BLE serveru | Červená | Blikanie | 1 | 100 |
| ESP klient je pripojený ku BLE serveru | Cyanová | Blikanie | 1 | 100 |
| Riadiaci člen sa pripája do TCP/IP siete | Žltá | Svietenie | - | - |
| Pripájanie do TCP/IP siete zlyhalo | Červená | Svietenie | - | Do ďalšieho pokusu o pripojenie |
| Pripojenie do TCP/IP je vytvorené | Modrá | Blikanie | 2 | 200 |
| Pripojenie do TCP/IP nie je vytvorené | Červená | Blikanie | 2 | 200 |

## Implementácia ESP klienta

Implementovaného ESP klienta je možné rozdeliť na dve časti. Prvou časťou je inicializácia jednotlivých objektov. Druhá časť reprezentuje bežné fungovanie mikrokontroléra, kde v nekonečnej slučke sa periodicky vykonáva meranie veličín s následným zaslaním do centrálneho uzla.

Proces inicializácie ESP klienta sa začína inicializáciou NVS pamäte. Výsledok inicializácie NVS je kontrolovaný a v prípade neúspechu je vykonaný ďalší pokus. Úspešnosť inicializácie je znova prekontrolovaná, kde v prípade opätovného neúspechu je mikrokontrolér vyresetovaný a proces začína od začiatku. Za predpokladu úspechu je ďalším krokom získanie inštancie objektu GreenhouseManager. Objekt je softvérovým manažérom celého ESP klientskeho systému, stará sa o vytvorenie Bluetooth objektov, objektu na kontrolu konektivity s BLE serverom, inicializáciu ovládačov pre senzory, spracovanie nameraných dát a taktiež príprava pred odosielaním do centrálneho uzla.

Inštanciou manažéra je následné volaná metóda na spustenie technológie Bluetooth. V rámci metódy sú spúšťané procesy, ako je inicializácia Bluetooth kontroléra, registrácia callback-ov a ďalšie procesy, ktoré budú detailnejšie popísané v kapitole 4.6. Pri kontrole návratovej hodnoty metódy je nastavený príslušný stavový kód definujúci aktuálny stav klienta. Aktualizovaný stav klienta je následné vizuálne reprezentovaný využitím podsystému na indikáciu stavu. Úspešné spustenie Bluetooth technológie nám dovoľuje spustiť sledovač konektivity, ktorého hlavnou úlohou je iba vizuálna reprezentácia konektivity ESP klienta ku BLE serveru s využitím už spomenutého podsystému na indikáciu stavu.Indikácia stavu klienta s využitím sledovača je vykonávaná periodicky v intervale desiatich sekúnd. Každých 10 sekúnd RGB dióda vyšle svetelný signál používateľovi o aktuálnom stave pripojenia, kde červená farba reprezentuje prerušenie pripojenie a farba modrá stabilné spojenie s BLE serverom.

## Implementácia riadiaceho člena

Proces inicializácie riadiaceho člena sa vo veľkej miere podoba na inicializáciu ESP klienta popísanú vyššie. Prvý krok je totožný s ESP klientom, a to je inicializácia NVS pamäte s kontrolou úspešnosti. Nasleduje spustenie Bluetooth servera a všetkých inicializačných procesov, ktoré detailnejšie budú popísané v kapitole 4.6.2. Po úspešnej inicializácii BLE servera sa prechádza na ďalší krok a to je pripojenie do TCP/IP siete. Na pripojenie do siete je využívaná technológia Wi-Fi. Pripojenie ku Wi-Fi smerovaču je dôležitou časťou navrhnutého systému a proces pripájania sa periodicky opakuje až do momentu úspešného pripojenia s priradenou IP adresou.

Navrhnutý systém obsahuje implementáciu sieťového manažéra.Úlohou manažéra je inicializácia Wi-Fi ovládača, MQTT klienta, spracovanie dát zo siete a zasielanie dát do siete. Pri prvotnom volaní metódy na pripojenie do siete je vytvorená inštancia Wi-Fi ovládača. Pri vytvorení inštancie ovládača sa nastavuje konfiguračný súbor s typom autentifikácie, prihlasovacieho mena a hesla. Následne je vytvorená nekonečna slučka pre sieťového udalosti a taktiež prebehne registrácia callback-ov na spracovanie týchto udalosti. Proces pripájania začína so zaslaním prvej žiadosti o pripojenie. Celkovo je zaslaných týchto žiadosti 10 v časovom rozmedzí 100 ms. V prípade ak ani jeden z pokusov o pripojenie nie je úspešný sa proces postaví na jednu minútu. Po uplynutí minúty sa proces znova začína zaslaním žiadosti. Ako už bolo spomínané proces je vykonávaný až do momentu úspešného pripojenia. Po úspešnom pripojený a priradený IP adresy je užívateľ systému svetelným signálom informovaný o úspechu a mikrokontrolér pokračuje s iniciovaním ďalších sieťových komponentov dôležitých pre správnu funkčnosť celého navrhnutého systému.

Ďalšími sieťovými komponentami sú manažér na správu systémového času implementovaný na báze NTP protokolu a MQTT klient. Spomenuté komponenty budú detailnejšie popísané nižšie aj s opisom teoretických informácií využitých protokolov a úlohou v rámci riadiaceho člena.

## Implementácia technológie Bluetooth

Kapitola popisuje implementáciu technológie Bluetooth v navrhnutom systéme, inicializáciu Bluetooth servera a jeho klientov s popisom princípu funkčnosti implementovaných softvérových modulov. Súčasťou kapitoly je taktiež krátky teoreticky popis o Bluetooth GATT profile, ktorý bol využitý pri implementácii.

### Bluetooth GATT profil

Súčasťou Bluetooth technológie je podmnožina rôznych profilov, ktoré zohrávajú dôležitú súčasť pravé pri komunikácii medzi centrálnym zariadením (Bluetooth server) a perifériou (Bluetooth klient). Každý z profilov ma vlastne špecifikácie, ako aj výhody a nevydoby v závislosti s kritériami navrhovanej aplikácie. GATT je jeden z podmnožiny profilov.

GATT je akronymom pre generický atribútový profil (ang. **G**eneric **ATT**ribute Profile). GATT profil definuje spôsob prenášania údajov medzi dvoma Bluetooth LE zariadeniami. Profil je definovaný na koncepte servisov a charakteristík a využíva generický dátový protokol ATT (ang. **ATT**ribute Protocol), ktorý je využívaný na ukladanie servisov, charakteristík a údajov do tabuľky so 16-bitovým jedinečným identifikátorom.

Exkluzívnosť GATT spojení je výraznou vlastnosťou, ktorá bola zvažovaná pri návrhu systému. Exkluzívnosť GATT spojenia znamená, že Bluetooth LE klient (periféria) je schopný sa súčasne pripojiť iba k jednému centrálnemu bodu a po pripojení je samo-propagácia centrálneho uzla pozastavená a tým ho ostatné zariadenia nedokážu vidieť ani sa k nemu pripojiť. Spomínané spojenie je zároveň jedinou cestou, ako zabezpečiť obojsmernú komunikáciu medzi klientom a centrálnym zariadením.

Koncept funkčnosti je založený na vzťahu klienta a servera. Obsahom GATT servera sú ATT vyhľadávacie dáta a taktiež definície servisov a charakteristík. GATT klient po pripojení už len zasiela požiadavky na server, nazývané aj transakcie.

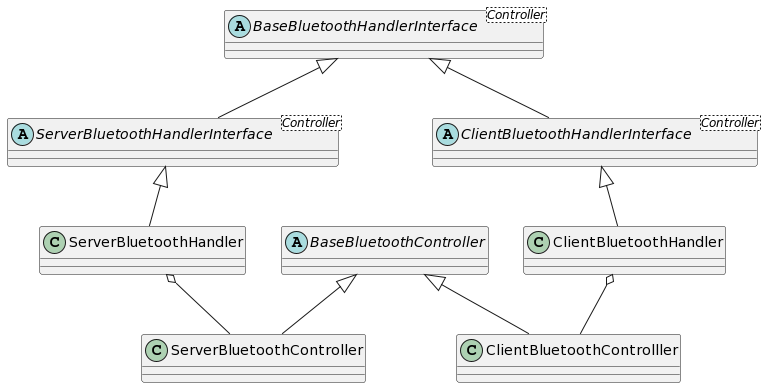
Obrázok, na ktorom je diagram

Automaticky generovaný popisGATT transakcie v Bluetooth LE sú založené na vnorených objektoch nazývané profily, servisy a charakteristiky. Nasledujúci obrázok graficky reprezentuje vnorene objekty pri GATT transakciách. [17]

Obr. 4.7 – Príklad GATT profilu [17]

### Bluetooth inicializácia BLE servera

Inicializácia servera pozostáva z niekoľkých krokov. Systém je prispôsobený na kontrolu každého z krokov počas inicializácie a v prípade neúspechu jedného z nich je využitý podsystém na indikáciu chybového stavu. Prvým krokom je inicializácia Bluetooth kontroléra. Po kontrole úspešnosti inicializácie kontroléra sa systém posúva na inicializáciu Bluetooth profilov. Záver inicializácie BLE servera pozostáva z registrácie callback-ov a registrácie profilu pre inteligentný skleník. Registrácia callback-ov pozostáva z registrácie **GAP** a **GATTs** callback-u. Navrhnutý systém obsahuje GreenhouseManager-a, ktorý vytvára objekty **ServerBluetoothHandler** a **ServerBluetoothController**. Obrázok nižšie graficky reprezentuje jednotlivé vzťahy medzi objektami.

Inicializácia kontroléra prebieha volaním funkcie s parametrom definujúcim bluetooth režim v rámci mikrokontroléra. V navrhnutom systéme bol použitý režim Bluetooth LE. V rámci volanej funkcie je v prvom rade uvoľnené miesto v pamäti pre nevyužitý klasicky Bluetooth s následnou inicializáciou kontroléra. Úspešnú inicializáciu zakončuje povolenie kontroléra k používaniu.

Obr. 4.8 – UML diagram reprezentujúci vzťah medzi objektami

Pri inicializácii profilov dochádza ku modifikácii dátovej štruktúry uloženej v objekte ServerBluetoothHandler. Pri tejto inicializácii je vložený záznam do dátovej štruktúry s ID Bluetooth profilu a callback-om pre spracovanie všetkých prichádzajúcich udalosti registrovaných pravé ku už spomínanému Bluetooth profilu. V navrhnutom systéme sa používa iba jeden profil. Profil s identifikátorom 0 reprezentujúci profil inteligentného skleníka. Obsahuje jeden servis s jednou charakteristikou na zápis údajov zo senzorov.

### Princíp fungovanie Bluetooth objektov na BLE servery

Primárnou úlohou objektu ServerBluetoothController je riadenie Bluetooth technológie vo vnútri ESP mikrokontroléra. Už počas inicializačného procesu bol používaný na registráciu callback-ov. Jeho ďalšie funkcie sú nastavovanie mena zariadenia, vytváranie servisu a charakteristík, ktoré sú pridávane do servisu, konfigurácia parametrov potrebných na propagáciu a taktiež samotné spúšťanie propagácie. ServerBluetoothHandler (ďalej len „handler“) je objektom slúžiacim na spracovanie všetkých druhov Bluetooth udalosti.

Komunikácia medzi objektami začína pri registrovaní profilu inteligentného skleníka. Handler pri spracovaní registračnej udalosti začne s nastavovaním mena zariadenia, nastavením konfiguračných dát na propagáciu, nastavením odpovedajúcich dát a ukončuje to vytvorením servisu. Pri každej z týchto operácii handler používa inštanciu triedy kontroléra a volá jeho funkcie. Každá z týchto operácii vyvoláva ďalšiu udalosť, ktorú spracováva handler. Pri prijatej udalosti o úspešnom nastavení konfiguračných dát a odosielajúcich dát sa začne proces so samo-propagáciou. Samozrejme aj o tejto udalosti je handler informovaný a je využitá ESP logovacia knižnica na zalogovanie úspešného resp. neúspešného pokusu o propagáciu. Po vytvorení servisu sú následné popridávané jednotlivé charakteristiky a servis je spustení.

Po všetkých týchto procedúrach ako bola inicializácia a následné vytváranie servisu a charakteristík je komunikácia medzi objektami sporadická a nastáva len vtedy, keď bluetooth klient posiela dáta do centrálneho uzla, vtedy udalosť o zápise dát do charakteristiky je spracovaná handler-om a s pomocou kontroléra je odoslaná odpoveď potvrdzujúca prijate dáta.

### Bluetooth inicializácia BLE klienta

Proces inicializácie BLE klienta je identicky s inicializáciou BLE servera. Inštanciou Greenhouse manažéra sú vytvorené objekty **ClientBluetoothController** a **ClientBluetoothHandler**. Volaním rodičovskej metódy kontroléra je kontrolér inicializovaný totožným spôsobom ako to bolo opísané v podkapitole na inicializáciou BLE servera, keďže ClientBluetoothController a ServerBluetoothController sú potomkovia spoločnej rodičovskej triedy BaseBluetoothController (pozri obrázok Obr. 4.8 – UML diagram reprezentujúci vzťah medzi objektami). Záver inicializácie uzatvára registrácia klientskych callback-ov s potrebným zaregistrovaním profilu s identifikátorom 0.

### Princíp fungovanie Bluetooth objektov na BLE klientovi

Podobne ako je to v prípade BLE servera, ESP klient obsahuje klientsku implementáciu kontroléra (ClientBluetoothController) a manažéra na spracovanie bluetooth udalosti (ClientBluetoothHandler). Klientska implementácia kontroléra sa vo veľkej miere odlišuje práve procesmi, ktoré vykonáva. Úlohy klientskeho kontroléra sú nastavovanie skenovacích parametrov, spustenie a zastavenie skenovania, vytvorenie spojenia s BLE serverom a samozrejme jeho ukončenie a mnohé ďalšie súvisiace s komunikáciou s BLE serverom.

Práca manažéra na spracovanie udalosti začína v momente zaregistrovania profilu. Po spracovaní registračnej udalosti manažér s využitím kontroléra nastavuje skenovacie parametre. Po úspešnom nastavení sa znovu použitím kontroléra spusti skenovanie bluetooth siete. Skenovanie je vykonávané v časovom horizonte 30 sekúnd. Počas tohto časového okna sú skenovacie výsledky spracovávané manažérom a v prípade ak výsledok skenovania je totožný s požadovaním názvom BLE servera tak skenovanie je pozastavené a vytvára sa spojenie.

Po vytvorení spojenia klient posiela na server navrhovanú maximálnu veľkosť jednej prenosovej jednotky. Paralelne s tým je manažérom spracovaná aj ďalšia udalosť. Udalosť indikuje, že na vzdialenom BLE servery boli objavené servisy. Manažér zas a znova s využitím kontroléra začne vyhľadávanie požadovaného servisu pre inteligentný skleník. Po úspešnom lokalizovaní požadovaného servisu systém spracuje dáta, získa charakteristiku servisu a ESP klient je pripravený na komunikáciu s centrálnym uzlom.

### Strata spojenia s BLE serverom

Obrázok, na ktorom je text

Automaticky generovaný popisStrata spojenia s BLE serverom je obrovským problémom navrhnutého systému. Keďže v navrhnutom systéme prakticky neexistujú žiadne iné redundantné cesty na zasielanie dát na BLE server pre ich ďalšie spracovanie, a preto je potrebné aby ESP klienti, ktorí sú zodpovední za zber údajov a následnú distribúciu mali stabilné spojenie s BLE serverom. Za týmto účelom bolo do klientskeho systému implementovaný modul na kontrolu a udržiavanie spojenia s BLE serverom. Definícia modulu pozostáva iba z konštruktora a deštruktora. Dôležitou súčasnou implementovaného objektu sú práve parametre konštruktora, smerník na Bluetooth kontrolér, referencia na premennú, ktorá uchováva aktuálny status ESP klienta z pohľadu bluetooth konektivity a posledným parametrom je hodnota, ktorá definuje periódu kontrolovania spojenia. Na kontrolu konektivity je využívaný ESP časovač, ktorý v definovanej perióde generuje prerušenia. Hlavnou ideou opísanej inicializácie objektu len s využitím konštruktora bez ďalších metód na kontrolu časovača je redukcia ďalších potrebných operácii na spustenie kontroly stavu. S definovanou periódou sú generované prerušenia, kde je kontrolované spojenie s BLE serverom. V prípade ak spojenie s BLE serverom neexistuje s využitím Bluetooth kontroléra sa spúšťa proces skenovania na 30 sekúnd ako to bolo opísane vyššie.

Obr. 4.9 – UML diagram modulu: ConnectionHolder

## Konfigurácia klienta

Konfigurácia klienta pozostáva z konfigurácie riadiaceho člena a ESP klientov. Konfigurácia je ďalšou časťou v procese implementácie a je nevyhnutné ju vykonať pred procesmi buildovania a nahrávania softvéru do mikrokontroléra. Nevyhnutnosť tohto kroku spočíva v tom, že konfigurácia projektu a úprava samotného kódu je vykonávaná pri proces kompilácie. Konfigurácia zohráva obrovskú úlohu pravé k uspokojeniu kritérií používateľa a prispôsobenie systému na základe použitých senzorov a iných dôležitých hodnotách.

Na konfiguráciu je využívaný ESP-IDF SDK konfiguračný editor. V konfiguračnom editore je využívaný esp-idf-kconfig, ktorý je založený na knižnici **kconfiglib**. Samotná knižnica je rozšírením systému **Kconfig** založený na Pythone. Mechanizmus na konfiguráciu projektu v čase kompilácie nám poskytuje priamo systém Kconfig, ktorý podporuje niekoľko dátových typov: celé číslo, reťazec znakov a dátový typ boolean. Špecifikácia systému Kconfig nám poskytuje možnosti na definovanie predvolených hodnôt, závislosťami medzi konfiguračnými hodnotami, zoskupenia týchto hodnôt a ďalšie. Po ukončení konfigurácie v editore sú dáta uložené do súboru sdkconfig, ktorý je súčasťou koreňového adresa aplikácie. Zo súboru sdkconfig je pri procese buildovania vytvorený hlavičkový súbor sdkconfig.h. Obsahom hlavičkového súboru sú vygenerované direktíva, ktoré sú následné využívané pri implementácii systému. [18]

### Konfigurácia ESP klienta

V navrhnutom systéme je konfigurácia ESP klienta rozdelená do troch kategórii. Prvá všeobecná kategória zahŕňa nastavenie klientskeho identifikátora a jeho pozíciu v rámci skleníka. Pri konfigurácii klientskeho ID je nevyhnutné aby osoba konfigurujúca klienta zabezpečila jedinečnosť poskytnutej hodnoty v rámci skleníka. Pri konfigurácii pozície je možnosť výberu z 2 možnosti. ESP klient môže byt umiestnený vo vnútri skleníka alebo mimo neho. Druhá kategória pokrýva konfiguráciu bluetooth s nastavením bluetooth názvu vzdialeného BLE servera. Poskytnutá hodnota je následné porovnávaná s výsledkami počas procesu skenovania. Posledná kategória obsahuje zoznam dostupných veličín. Pri výbere veličín je taktiež potrebné sparovať veličiny so správnym senzorov a daný senzor aj fyzicky pripojiť ku mikrokontroléru. Príkladom nesprávnej konfigurácie môže byt práve požiadavka od používateľa merať CO2 no k mikrokontroléra sa pripojí senzor SHT41, ktorý je limitovaný na meranie iba teploty a vlhkosti vzduchu.

### Konfigurácia riadiaceho člena

Obr. 4.10 – Záznam z konfigurácie ESP klienta

Konfiguráciu riadiaceho člena je taktiež možné rozdeliť do niekoľkých kategórii. Do všeobecnej kategórii, kde sa konfiguruje identifikátor klienta. Pri konfigurácii identifikátora je potrebné zabezpečiť jedinečnosť v rámci celého systému a je nevyhnutné použiť unikátne ID pre každý registrovaný skleník. Ďalšou konfiguračnou časťou je nastavovanie sieťových prvkov. Prvým krokom je konfigurácia údajov na prístup do siete s využitím technológie Wi-Fi. Nasleduje konfigurácia IP adresy servera, na ktorom sú spustené služby ako MQTT Broker, NTP server a pod. Služby nemusia byt umiestnené na jednom servery, a preto bola pridaná možnosť individuálne nakonfigurovať IP adresy prislúchajúcej službe. Poslednými konfiguračnými hodnotami k nastaveniu je časová zóna využívaná v NTP servery a autentifikačné údaje na pripojenie ku MQTT Broker-u.

## Zber údajov

Najpodstatnejšou úlohou ESP klienta je komunikácia so senzormi na zber dát. Pri senzoroch SHT41 a SCD41 je na komunikáciu využívaná zbernica I2C. Meranie vlhkosti pôdy je zabezpečené kapacitným senzorom, ktorého výstupom je analógová hodnota. Pri meraní veľkosti pôdy je využívaný v rámci mikrokontroléra analógovo - digitálny prevodník so šírkou 12 bitov. Výstupná hodnota z prevodníka je následné poskytnutá ako vstup do vzorca na výpočet finálnej hodnoty vyjadrujúca vlhkosť pôdy v percentách.

 [%]

* Data – vstupné dáta zo senzora
* SENSOR\_MIN – minimálna vstupná hodnota zo senzora
* SENSOR\_MAX– maximálna vstupná hodnota zo senzora

Softvérová implementácia v navrhnutom systéme je prispôsobená na inicializáciu zbernice I2C zbernice len za predpokladu, že počas konfigurácie ESP klienta bola zvolená aspoň jedna z možnosti, či už to bolo meranie teploty, vlhkosti vzduchu alebo meranie CO2. Variácia jednotlivých možnosti taktiež vo veľkej miere ovplyvňuje typ inicializovaného ovládača pre konkrétny senzor. V prípade ak používateľovi postačuje merať iba teplotu a vlhkosť vzduchu je inicializovaný ovládač pre senzor SHT41. Pri zvolenej možnosti CO2 je nepodstatné aké ďalšie možnosti sú zvolené, automaticky sa inicializuje ovládač na senzor SCD41, keďže jediný poskytuje meranie tejto veličiny. Integrácia senzora SHT41 do navrhnutého systému, napriek tomu, že je súčasťou senzora SCD41 bola z dôvodu nevyužiteľnosti plného potenciálu senzora SCD41 v spomínanom príklade, kedy používateľ nepožaduje meranie CO2. Samostatný senzor SHT41 je taktiež lacnejšou alternatívou za predpokladu merania iba teploty a vlhkosti vzduchu.

## Prenos dát do centrálneho uzla

Proces zasielania dát je vykonávané s využitím bluetooth kontroléra, no ešte pred samotným zaslaním údajov je vykonané meranie predkonfigurovaných veličín na dosiahnutie čo najaktuálnejších hodnôt reflektujúce realitu. Namerané dáta sa následné spracujú a vytvorí sa dynamická štruktúra obsahujúca základne informácie o klientovi, bajt definujúci obsah dynamickej štruktúry a nasledujú namerané dáta. Tabuľka nižšie graficky reprezentuje obsah dynamickej štruktúry. Každá bunka tabuľky reprezentuje 8 bitov v rámci štruktúry.

Tab. .2 –Dátová štruktúra bluetooth dát

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | <2 … N> | | | | | | | |
| ID/P | D\_CON | TEM\_H | TEM\_L | HUM\_H | HUM\_L | CO2\_H | CO2\_L | Soil\_H | Soil\_L |

Popis jednotlivých bajtov:

* **ID/P** - Bajt na 0 pozícii vždy obsahuje klientsky identifikátor a pozíciu v rámci skleníka. Tieto údaje sú v rámci bajtu rozdelené na 2 časti, kde bity 0 a 1 vyjadrujú pozíciu.

Tab. .3 – ID Bajt

|  |  |
| --- | --- |
| 6 bitov | 2 bity |
| Klientsky identifikátor | Pozícia klienta |

* **D\_CON** – Popis dynamickej časti štruktúry je definovaný práve v tomto bajte. Každej veličine je priradený jeden bit, kde hodnota bitu vyjadruje či veličina je súčasťou dynamickej štruktúry. Bit s hodnotou 1 vyjadruje, že daná veličina je zahrnutá. Pri bite s hodnotou 0 je význam opačný.

Tab. .4 – Bajt definujúci dynamickú časť

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Teplota | Vlhkosť vzduchu | Oxid uhličitý | Vlhkosť pôdy | Rezervovane | | | |

* **Dynamická časť** – Pri vytváraní dynamickej časti je nevyhnutné dodržať predefinované poradie a dáta zapisovať v poradí ako je to v tabuľke Tab. 4.2 –Dátová štruktúra bluetooth dát. Pri návrhu systému bola definovaná veľkosť dát na 2 bajty pre každú jednu veličinu. Pri veličinách, ktoré môžu dosahovať hodnoty s presnosťou na desatiny je horných 8 bitov rezervovaných pre celú časť a dolných 8 bitov pre desatinnú časť. V tabuľke Tab. 4.2 sú horné bity veličín definované ako VELICINA\_H a dolné bity sú VELICINA\_D. Dynamická časť štruktúry bola navrhnutá do systému na redukciu veľkosti prenášaných dát s cieľom efektívnejšej komunikácie s BLE serverom.

## Spracovanie Bluetooth dát riadiacim členom

Proces spracovanie klientskych dát z rozhrania bluetooth začína spracovaním bluetooth udalosti objektom **ServerBluetoothHandler.** Ide o udalosť kedy ESP klient sa snaží zapísať dáta do charakteristiky servisu. Pri prijatí dát je s využitím kontroléra zaslaná odpoveď o úspešnom prijatí dát a nasleduje parsovanie dát. Proces parsovania dát je opačný ku procesu vytvárania dynamickej dátovej štruktúry. Prvým krokom je rozparsovanie klientskeho ID a jeho pozície a následné na základe hodnoty bitu na pozícii 1 je rozparsovaná dynamická časť dát.

Proces distribúcie rozparsovanych dát v rámci systému zohrával dôležitú úlohu z hľadiska zasielania dát na server, ako aj na budúce úpravy softvéru či už implementácia archivácie dát v mikrokontroléry, spracovanie dát a ich následná distribúcia do iných časti systému a pod. Z týchto dôvodov bol do systému implementovaný návrhový vzor **Observer**. Vlastnosti a výhody tohto návrhového vzoru nám dopomohli ku distribúcii dát do všetkých častiach systému, ktoré o dáta prejavili záujem.

Princíp implementácie návrhového vzoru je postavený na modely pub/sub. Objekty, ktoré majú záujem o dáta zo špecifickej udalosti sa prihlásia k odberu danej udalosti. Spracovanie registrácie odberu, publikovanie dát naviazané na špecifickú udalosť a distribúcia dát konkrétnym záujemcom je v réžii objektu EventManager.

Distribúcia rozparovaných dát objektom ServerBluetoothHandler je zabezpečovaná EventManager-om. Vytvorí sa dátová štruktúra, ktorá obsahuje základné informácie ohľadom klienta a jeho poskytnuté namerané dáta. Po vytvorení štruktúry sú informovaný všetci záujemcovia o prijatí najnovších dát od ESP klienta. Aktuálna implementácia systému obsahuje iba jedného záujemcu, a to objekt **BluetoothDataObserver.** Pozorovateľ je súčasťou sieťového manažéra a po spracovaní dát sa s využitím jeho funkcionality zasielajú dáta na server. Dáta sú zasielané na server vo formáte JSON a obsahujú identifikátor klienta, pozíciu a namerané dáta.

## Manažér synchronizácie času

Primárnou úlohou manažéra je synchronizácia času, ako je možné vydedukovať z názvu kapitoly. Cieľom implementácie manažéra do systému je synchronizácia času na báze protokolu NTP medzi všetkými skleníkmi, ktoré majú vytvorené spojenie k jednému spoločnému serveru a taktiež poskytnutie aktuálneho času ostatným komponentom, či už pri logovaní alebo priraďovaní aktuálneho času ku dátam zo senzorov.

NTP (ang. **N**etwork **T**ime **P**rotocol) je sieťový protokol z 80. rokov minulého storočia. Účelom protokolu bola synchronizácia systémových hodín počítačov, ktoré boli prepojené v sieťach založených na prepínaní paketov. Čas pri implementácii NTP protokolu na synchronizovaných zariadeniach musí byt totožný alebo len trocha rozdielny. Dôležitosť takejto zásady zohráva obrovskú úlohu pravé v systémoch, kde je presný čas súčasťou nejakej analýzy.

Vlastnosti protokolu NTP:

* NTP servery využívajú atómové a GPS hodiny, ktoré sú jedny z najpresnejších na svete. Čas z týchto hodín je používaný ako referenčný a využíva sa na synchronizáciu medzi klientmi. Časovou jednotkou NTP serverov je UTP. V UTP sa nezohľadňujú časove pásma no dôvod využitia UTP v NTP serveroch je relatívna presnosť pri zisťovaní zmien.
* Ďalšou vlastnosťou protokolu je zabraňovanie zlomkových chýb a zraniteľnosti pri výmene informácii medzi klientskou a serverovou časťou. NTP disponuje technikami, ktoré vyriešia chybu bez ohľadu na veľkosť danej chyby. [19]

V aktuálnej verzii navrhnutého systému sa manažér synchronizácie času využíva výhradné na získanie aktuálneho času pri inicializácii dátovej štruktúry, ktorá obsahuje namerané dáta prijaté rozhraním bluetooth.

### Architektúra protokolu NTP

Typ architektúry protokolu NTP je definovaný ako hierarchicky. Každá úroveň v rámci hierarchie je nazývaná vrstvou. Architektúra hierarchického systému môže obsahovať maximálne 16 vrstiev a NTP server na vrstve 1 je najpresnejším časovým serverom. Najpresnejší je z dôvodu priamej referencie na autoritatívny zdroj času, ako sú atómové hodiny.

Hierarchicky systém definuje NTP servery na úrovni 1, ako primárne servery a NTP servery na úrovni 2 synchronizujú svoj čas so servermi o vrstvu vyššie. Rovnaký spôsob synchronizácie je uplatnený aj v ostatných vrstvách, kde daná vrstva berie ako referenciu synchronizácie času NTP servery z úrovne vyššie.

Synchronizácia času s využitím NTP pozostáva z výmeny niekoľkých paketov. Každá výmena paketov obsahuje pár požiadavky a jej odpovede. Pri úvodnom zaslaní požiadavky klientsky packet obsahuje svoj vlastný klientsky čas. Na strane servera je packet prijatý a spracovaný s následným pridaním času servera a packet sa vracia späť ku klientovi. Po prijatí packetu klientom sa zaznamenáva znova klientsky čas a prebiehajú výpočty na určenie celkového oneskorenia. Celkové oneskorenie v podstate vyjadruje čas cesty packetov sieťou a čas spracovania na servery.

Nastavenie nového času však nenastáva ihneď po prvej výmene opísaného páru packetov. K synchronizácii času je potrebné splniť podmienky, ktoré sú definované v špecifikácii protokolu: výmena niekoľkých párov paketov, pakety musia prejsť súborom kontrolných algoritmov a samotný NTP server musí by považovaný za validný NTP server. [19]

## MQTT

Ďalším zo sieťových komponentov v rámci navrhnutého systému je MQTT klient. MQTT klient je implementovaný na báze protokolu MQTT slúžiaci hlavné na obojsmernú komunikáciu a výmenu údajov medzi serverom a riadiacim členom skleníka.

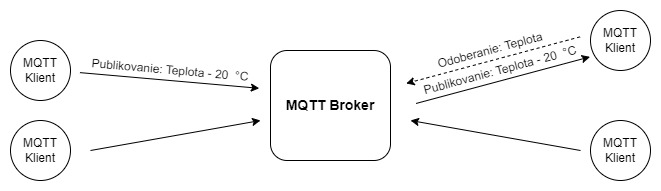
### Protokol MQTT

Protokol MQTT je odľahčený binárny transportný protokol, založený na architektúre server – klient. Je navrhnutý na báze publikovania a odoberania správ s vlastnosťami ideálnymi na využitie v systémoch IoT alebo v prostrediach M2M (ang. **M**achine to **M**achine).

Protokol MQTT navrhli v roku 1999 Andy Stanford-Clark a Arlen Nipper. Andy a Arlen v tom období potrebovali protokol, ktorý by splnil požiadavky na využitie minimálnej šírky pásma pri spojení ropovodu a satelitu s minimálnou spotrebou pri komunikácii. Následné na to bolo zadefinovaných niekoľko požiadaviek budúceho protokolu:

* Jednoduchá implementácia
* Odľahčený a efektívny na šírku pásma
* Poskytovať údaje o kvalite služby
* Protokol by mal byt dáta agnosticky [20]

### Model pub/sub

Model pub/sub je alternatívou architektúry server – klient, ktorý funguje na báze publikovania a odoberania správ. Architektúra server – klient poskytuje používateľovi (klient) priamu komunikáciu s koncovým bodom (server). Model pub/sub separuje klienta, ktorý publikuje správu od ostatných klientov, ktorí správu odoberajú. Špecifikácia protokolu nedovoľuje, aby sa jednotliví klienti navzájom priamo kontaktovali. Spojenie a komunikácia medzi klientami je sprostredkovaná tretím potrebným členom: Broker-om. Úlohou broker-a je filtrácia prichádzajúcich sprav a redistribúcia sprav klientom, ktorí majú o danú správu záujem. Nasledujúci obrázok popisuje základný princíp fungovania modelu pub/sub.

Obr. 4.11 – Princíp modelu pub/sub [21]

Najvýraznejšou vlastnosťou modelu pub/sub je oddelenie klientov. Definícia protokolu však poskytuje oddeľovanie klientov v troch rôznych častiach. Prvou časťou je priestorové oddelenie, ktoré už bolo popísané a funguje s využitím tretieho člena. Ďalšou časť

ou je oddelenie časové, čo znamená, že odoberajúci klient nemusí byt spustený v rovnakom čase ako publikujúci klient, ktorý posiela správu cez broker-a. Posledným možným oddelením je synchronizačné. Synchronizačné oddelenie poskytuje klientovi záruku, že počas publikovania alebo prijímania správy nebude prerušený.

### Filtrácia správ

Z opisu modelu pub/sub je jasné, že MQTT broker je nevyhnutným členom v rámci architektúry a je v jeho réžii filtrácia a redistribúcia správ. Broker ma niekoľko možnosti ako filtrovať prichádzajúce správy.

* **Filtrovanie na základe predmetu –** Filtrácia správ na základe predmetu alebo témy, ktorá je súčasťou každej správy. Klient, ktorý ma záujem o špecifickú tému zašle do MQTT broker-a registračný podnet na odber danej témy. Následné je povinnosťou MQTT broker-a aby zabezpečil, že klient, ktorý sa zaujíma o tému obdržal všetky správy publikované do danej témy. Téma je definovaná ako pole znakov s hierarchickou štruktúrou.
* **Filtrovanie na základe obsahu –** Pri filtrácii na základe obsahu je MQTT broker nakonfigurovaný na špecificky obsahový filter. Následné jednotliví klienti sa registrujú s požiadavkou záujmu o špecificky obsah. Obrovskou nevýhodou takejto filtrácie je predom definovaný obsah, ktorý nie je možné ani zašifrovať ani modifikovať.
* **Filtrovanie na základe typu** - Filtrácia na základe typu je zvyčajne využívaná pri objektovo-orientovaných jazykoch, kde klient odoberá istý druh správ na základe dátového typu alebo triedy. [21]

### MQTT Broker a MQTT Klient

MQTT klient je akékoľvek zariadenie, od mikrokontroléra až po plnohodnotný server, ktoré využíva MQTT knižnicu a pripája sa ku MQTT Broker-u. MQTT klient môže byt súčasné využívaný na publikáciu správ ako aj na prijímanie. Výbavou klienta je štítok, ktorý definuje klienta či je aktuálne publikujúci alebo práve odoberá správu.

MQTT broker je zariadenie, ktoré tvorí jadro celého pub/sub modelu. Je zodpovedný za príjem každej správy s následnou filtráciou a rozdistribuovaním požadovaným klientom. Jedna z pod úloh MQTT broker-a je aj tvorba záznamovom o reláciách všetkých klientov, záznamy o prihlásených klientov a príslušných témach, ktoré odoberajú. Jednotlivé záznamy sú využívané práve pri distribúcii prijatých sprav konkrétnym klientom. Ďalšou obrovskou zodpovednosťou je autentifikácia a autorizácia klientov. Avšak závisí od konfigurácii broker-a a systém je možné integrovať aj bez autentifikácie a autorizácie.

### MQTT Pripojenie

MQTT pripojenie je založené z definície protokolu na TCP/IP s tým, že spojenie je vždy vytvorené medzi MQTT klientom a MQTT broker-om. Na vytvorenie spojenia je potreba iniciatíva zo strany klienta. Klient posiela tzv. **CONNECT** správu a očakáva správu **CONNACK** posielanú serverom.

V prípade ak pri inicializácii spojenia klienta s broker-om je zaslaná chybná sprava CONNECT, v tom prípade je spojenie MQTT brokerom odmietnuté. Ďalším z prípadov kedy spojenie je prerušené na strane broker-a je v prípade ak časový rozdiel medzi otvorením webového socketu a zaslaním pripojovacej správy prekročí definovanú hodnotu.

Obrázok, na ktorom je text

Automaticky generovaný popis**ClientId** – Identifikátor MQTT klienta. Je potrebné zabezpečiť jedinečnosť v rámci klienta aj MQTT broker-a. Vo verzii MQTT 3.1.1 je možné zaslať prázdny identifikátor klienta, ak nie je za potreby, aby broker vlastnil stav klienta.

Obr. 4.12 – MQTT pripájacia sprava [22]

**CleanSession** – Príznakový bit, ktorého úlohou je informovať broker-a či daný klient ma záujem o trvalú reláciu. Pri trvalej relácii je nevyhnutné mať nastavenú kvalitu služieb 1 alebo 2 a vtedy všetky klientske odbery ako aj zmeškané správy sú archivované.

**Username/Password –** Polia na autorizáciu klienta. Ak implementácia neposkytuje kryptovací algoritmus alebo dáta nie sú šifrované na úrovni TLS tak v tom prípade údaje o mene a hesle sú zaslané ako obyčajný text.

Sprava, zasielaná MQTT broker-om, **CONNACK** obsahuje stavový kód. Nasledujúca tabuľka obsahuje stavové kódy v správe od MQTT broker-a. [22]

Tab. .5 – Význam stavového kódu

|  |  |
| --- | --- |
| Stavový kód | Význam |
| 0 | Spojenie akceptované |
| 1 | Spojenie odmietnuté. Neakceptovaná verzia protokolu |
| 2 | Spojenie odmietnuté, odmietnutý identifikátor |
| 3 | Spojenie odmietnuté, server nie je dostupný |
| 4 | Spojenie odmietnuté, nesprávne meno alebo heslo |
| 5 | Spojenie odmietnuté, neautorizovaný |

## Implementácia MQTT klienta

Inicializácia MQTT klienta v rámci navrhnutého systému pozostáva z vytvorenia dátovej štruktúry, ktorá obsahuje všetky potrebné informácie na vytvorenie korektnej správy CONNECT a následné úspešné prihlásenie. Vypĺňaný je identifikátor klienta, IP adresa a port MQTT broker-a a poslednými údajmi sú aj meno a heslo na autentifikáciu. Po úspešnej inicializácii klienta je s využitím sieťového manažéra zaregistrovaný callback na spracovanie MQTT udalosti a je volaná metóda na spustenie klienta. Spustenie klienta v podstate znamená vytvorenie správy **CONNECT** a vytvorenie spojenia s MQTT broker-om.

O úspešnosti pripojenia je systém oboznámený v registrovanom callback-u cez spracovanie dát z MQTT udalosti. Navrhnutý systém spracováva päť typov udalosti. Spracovanie udalosti, ako je publikovanie dát , registrácia požadovaných tém (odber) alebo odpojenie klienta pozostáva s využitia iba ESP logovacej knižnice a vytvorenie logovacieho záznamu. Pri spracovaný udalosti o pripojení klienta ku MQTT broker-ovi sú vykonané ďalšie procedúry. Prvou je odoslanie registračných údajov skleníka. Štruktúra a dôvod odosielania týchto údajov je momentálne nepodstatná pri popise MQTT klienta. Významnosť údajov bude popísaná pri popise navrhnutého riešenia na servery s využitím databázových systémov. Druhou procedúrou je registrácia odberových tém. Klient odoberá témy úzko prepojené so závlahou a ventiláciou skleníka. Poslednou udalosťou, ktorú sieťový manažér spracovaná je prijímanie dát z tém, ktoré boli zaregistrované k odberu.

### Vzdialené ovládanie akčných členov

Vzdialené ovládanie akčných členov je dosiahnuté v navrhnutom systéme práve kombináciou ComponentController-a a MQTT klienta. Odoberané témy v rámci MQTT klienta sú využité na vzdialené nastavenie požadovanej hodnoty akčného člena.

Odoberané témy:

* Greenhouse/window – riadenie závlahy
* Greenhouse/window/<Jedinečný identifikátor skleníka>
* Greenhouse/irrigation – riadenie odvetrávania
* Greenhouse/irrigation/<Jedinečný identifikátor skleníka>

Odoberane témy bez využitia jedinečného identifikátora skleníka sú implementované do systému k budúcemu využitiu, kde používateľ vo webovom rozhraní bez udania špecifického skleníka bude schopný spustiť závlahu alebo ventiláciu vo všetkých zaregistrovaných skleníkoch jedným klikom. Témy so špecifikovaním identifikátora sú určené priamo konkrétnemu skleníku a publikované dáta sú spracované sieťovým manažérom. Publikované dáta sú vo formáte JSON s jediným elementom. Element definujúci požadovanú hodnotu {"requested":1}. Sieťový manažér zabezpečuje korektné spracovanie publikovaných dát a podľa obsahu JSON-u je s využitím ComponentController-a spustená príslušná akcia.

Pri všetkých odoberaných témach je prijatá dátová štruktúra rovnaká. Hodnotou „1“ je požadované aktivovať akčný člen a pri hodnote „0“ to je deaktivácia. Aktivácia v prípade odvetrávania znamená otvorenie okna a pri závlahe to je spustenie vodného čerpadla. Deaktivácia je opačným úkonom pri už popísaných dejoch. Do systému bol taktiež implementovaný kontrolný algoritmus a v prípade ak požadovaná hodnota koreluje s aktuálnym stavom akčného člena, požiadavka je ignorovaná.

## Implementácia servera

Celá implementácia pozostáva z využitia platformy Docker, kde sú spustené štyri kontajnery.  Každý z kontajnerov zohráva v navrhnutom systéme dôležitú úlohu no skôr ako budú detailnejšie popísané je vhodné najskôr opísať využitú platformu.

Platforma **Docker** je platforma s verejne dostupným zdrojovým kódom, ktorá poskytuje vývojárom správu kontajnerov. Pod správou sú zahrnuté operácie ako vytváranie, nasadzovanie, spúšťanie a aktualizácia kontajnerov. Kontajner je komponentom, ktorý kombinuje operačný systém a jeho knižnice so zdrojovým kódom aplikácie a všetkými ďalšími závislosťami, ktoré sú nevyhnutné na spustenie aplikácie v akomkoľvek prostredí.

Využitie kontajnerovej technológie poskytuje vývojárovi možnosti na prideľovanie zdrojov medzi procesmi, obmedzovanie zdrojov pre špecificky proces a taktiež mu umožňuje zdieľať zdroj inštancie hostiteľského operačného systému podobným spôsobom, ako je to umožňované hypervízorom vo virtuálnych strojoch, kde je zdieľaná CPU, pamäť a ďalšie hardvérové komponenty. Tato technológia ponuka všetky funkcie a výhody virtuálnych strojov, ako je izolácia aplikácie, efektívna škálovateľnosť pri nízkych nákladoch a ďalšie:

* Nižšie zaťaženie systému v porovnaní virtuálnych mašín, keďže kontajnery zahŕňajú procesy v operačnom systéme v závislosti so spustením kódu aplikácie.
* Lepšie využitie kapacity hardvéru, keďže veľkosť kontajnerov je často meraná iba v megabajtoch, čo ma za následok aj rýchlejšie spúšťanie.
* Zvyšovanie produktivity vývojárom, keďže aplikácie v kontajnery treba napísať iba raz a vlastnosti kontajnera poskytujú ľahšiu distribúciu.
* Efektivita pri využití hardvérových zdrojov, využitie kontajnera umožňuje spustenie viacerých kópii aplikácie na rovnakom hardvéri v porovnaní s virtuálnymi strojmi.

Docker v navrhnutom systéme obsahuje NTP server na synchronizáciu času s riadiacim členom, kontajner s vývojovým nástrojom Node-Red, kontajner s NoSQL databázou MongoDB na archiváciu nameraných dát a posledným kontajnerom je MQTT broker. [23]

### Vytváranie sieti v rámci platformy Docker

Vytváranie sieti v rámci platformy Docker je využívané hlavne na nadviazanie komunikácie medzi dockrovymi kontajnermi a vonkajším svetom. Zariadenie, na ktorom je spustený docker je využívané ako prostriedok na komunikáciu s vonkajším svetom. Docker podporuje rôzne sieťové ovládače, ktoré sú zamerané na rôzne prípady využitia podľa špecifikácie systému. Najbežnejšie typy sieťových ovládačov na platforme Docker sú:

* **bridge –** Jedna z najbežnejších typov sieťových ovládačov. Využívanie takejto siete je obmedzené na dockrové kontajnery v rámci jedného hostiteľa. Pri konfigurácii kontajnera s integráciou do siete **bridge** je nevyhnutné vykonať mapovanie portov. Mapovanie portov umožni komunikáciu s vonkajším svetom.
* **overlay –** Pri vytváraní sieťovej vrstvy je využívaná softvérová virtualizácia s cieľom vytvoriť ďalšie abstraktné sieťové vrstvy fungujúce nad fyzickou sieťou. Využitie takéhoto typu siete je možné nájsť pri komunikácii s viacerými hostiteľmi, ktorí využívajú sieťový ovládač **overlay.** Technológia Virtual Extensible LAN (VXLAN), je technológiou, ktorá poskytuje prenositeľnosť medzi cloudovými, lokálnymi a virtuálnymi prostrediami.
* **macvlan –** Posledným typom z množiny najbežnejších je **macvlan,** ktorý sa využíva predovšetkým na pripojenie kontajnera priamo ku rozhraniu hostiteľského zariadenia. Pri využití takéhoto sieťového ovládača nie je potrebné mapovať porty, vytvárať iné vrstvy alebo využiť techniku NAT s prekladaním IP adries ale ovládač poskytuje priradenie verejne IP adresy priamo na rozhranie, ktoré je prepojené s kontajnerom. Výhodou je nízka latencia z dôvodu priameho prepojenia.

### Komunikácia medzi kontajnermi a vonkajším svetom

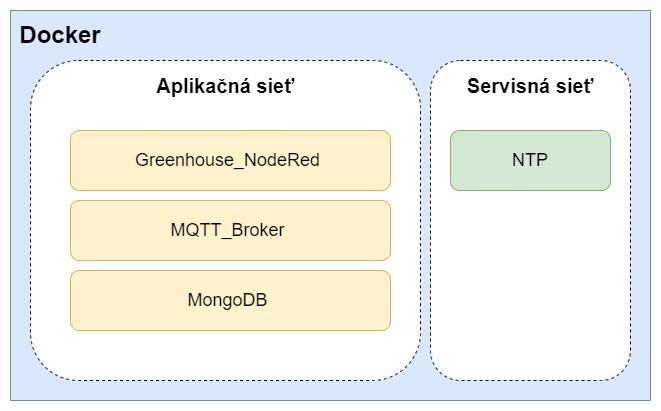
Komunikácia medzi kontajnermi závisí od typu využitej siete so sieťovými ovládačmi. Na základe typu siete platforma Docker umožňuje komunikáciu medzi kontajnermi s využitím IP adries alebo využitím mien kontajnerov. Pri vytváraní kontajnera Docker automaticky priradí názov a sieťové meno kontajneru a priradí ho do preddefinovanej siete **docker0.** Každá hodnota z týchto atribútov je konfigurovateľná podľa požiadaviek vývojára. Po vytvorení kontajnera a nastavení požadovaných atribútov Docker uchováva priradený názov a sieťové meno a prebieha proces mapovania týchto atribútov voči priradenej IP adrese kontajnera. Mapovanie atribútov voči IP adresám umožňuje komunikáciu medzi kontajnermi na základe mena bez explicitného používania IP adresy.

Ďalšou možnosťou na komunikáciu medzi kontajnermi je priame prepojenie. Priame prepojenie je možné zadefinovať pri spúšťaní kontajnera a umožňuje to vzájomné objavenie kontajnerov s možnosťou prenášať bezpečne informácie medzi kontajnermi.

Ako to bolo pri komunikácii medzi kontajnermi aj v prípade komunikácii kontajnerov s vonkajším svetom existuje viacero možností. Prvým spôsobom je odhalenie špecifického portu v sieti a nasmerovanie žiadosti z vonkajšieho sveta na definovaný port. Pri tomto spôsobe je doležíte namapovanie portu aplikácie na port, kde budú nasmerované všetky žiadosti z vonkajšieho sveta. Ďalším spôsobom je vytvorenie viacerých sieti. Pri vytváraní kontajnerov bude každý priradený do predom definovanej siete a následné budú tieto siete prepojené. [24]

## Sieťovanie na platforme Docker v navrhnutom systéme

Navrhnutý systém obsahuje dve siete s využitím sieťového ovládača **bridge**. Sieť s názvom **GreenhouseNetwork** (aplikačná sieť) a **ServiceNetwork** (servisná sieť)**.** Aplikačná sieť obsahuje Node-Red, NoSQL databázu a kontajner s MQTT Broker-om. Servisná sieť obsahuje iba inštanciu NTP servera. Rozdelenie siete je možné vidieť na nasledujúcom obrázku.

V obidvoch sieťach bolo využité mapovanie portov na komunikáciu s vonkajším svetom. Priradene IP adresy a mapovanie portov je rozpísane v tabuľke nižšie. Zoskupenie kontajnera Node-Red, NoSQL databázy a MQTT Broker-a bolo využité hlavne kvôli komunikácii medzi kontajnermi. S poskytovanými vlastnosťami platformy Docker a jeho sieťovania sme schopný pri zoskupený týchto kontajnerov využiť mapovanie sieťových mien ku IP adresám na prenos dát v back-end-ovej časti vývojového nástroja Node-Red a databázy. Obrovskú výhodu to prináša aj pri využívaní MQTT klientov u ktorých je potrebné zaregistrovať odoberanú tému a pri tomto návrhu je to bez nutnosti využiť IP adresu MQTT Broker-a.

Obr. 4.13 – Sieťovanie v navrhnutom systéme

Tab. .6 – Zoznam kontajnerov

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Názov kontajnera | Sieť | Adresa siete | Ip adresa | Port aplikácie | Priradený externý port |
| Greenhouse\_NodeRed | aplikačná | 172.20.0.0/24 | 172.20.0.2 | 1880 | 26001 |
| MQTT\_Broker | aplikačná | 172.20.0.0/24 | 172.20.0.3 | 1883 | 26002 |
| MongoDB | aplikačná | 172.20.0.0/24 | 172.20.0.4 | 27017 | 26003 |
| NTP | servisná | 172.21.0.0/24 | 172.21.0.2 | 123 | 123 |

## NTP server

Konfigurácia NTP servera pozostáva z inicializácie kontajnera, ktorý bol vytvorený z docker obrazu cturra/ntp. Obraz obsahuje implementáciu chrony na alpine linux jadre. Chrony je implementácia NTP servera, ktorá poskytuje synchronizáciu systémového času s inými NTP servermi. Presnosť synchronizácie je meraná v milisekúnd s tým, že v prípade sieti LAN je presnosť v desiatkach mikrosekúnd. Súčasťou chrony je chronyd démon, ktorý je spúšťaný pri štarte kontajnera a chronyc. Chronyc je rozhranie príkazového riadku, ktorý poskytuje rozhranie na monitorovanie chronyd-u a modifikáciu parametrov počas behu programu.

Kontajner NTP je súčasťou servisnej siete s priradenou IP adresou 172.21.0.2. Služba počúva na lokálnom porte 123 s využitým mapovaním na externý port 123.

V hierarchickej štruktúre NTP serverov je umiestený vo vrstve 4. Synchronizácia prebieha s preddefinovaným NTP serverom „time.cloudflare.com“ umiestneným na úrovni 3.

## Používateľské webové rozhranie

Používateľské rozhranie je vytvorené s využitím vývojového nástroja Node-Red. Rozhranie používateľovi umožňuje zobrazenie aktuálneho zoznamu registrovaných skleníkov k jeho poskytnutým používateľským menom. Po zvolení jedného skleníka zo zoznamu je používateľovi zobrazená webová stránka zobrazujúca aktuálne hodnoty v rámci skleníka ako aj jeho vonkajšieho prostredia. Zvolením skleníka v zozname je používateľovi taktiež umožnený prístup ku archívnym údajov a s využitím grafov je možné sledovať historicky vývoj veličín v skleníku a mimo neho (Príloha B).

Registráciu inteligentného skleníka ku používateľovi je možné vykonať iba za podmienky prístupu k databáze a následnej modifikácii zoznamu skleníkov. Tato modifikácia je vykonávaná iba administrátorom spomínanej databázy. Po modifikácii databázy je používateľovi umožnený prístup ku skleníku na získanie prehľadu o stave skleníka. Používateľské meno je poskytnuté v rámci prístupu ku zoznamu skleníkov ako URL parameter s názvom **username.**

Používateľské rozhranie taktiež obsahuje panel umiestnený v ľavej časti obrazovky, ktorý slúži na prepínanie medzi jednotlivými kartami. Panel môže obsahovať maximálne 3 ikony. Ikona s domčekom reprezentujúca domovskú kartu s registrovanými skleníkmi, ikonu s kvetom a posledná ikona reprezentuje hodiny so šípkou vracajúcou sa v čase. Ikona s kvetom zobrazuje kartu s aktuálnymi hodnotami skleníka, ikona s hodinami zobrazí kartu, ktorá obsahuje grafy s možnosťou zobrazenia archívnych údajov.

Po načítaní domovskej stránky s registrovanými skleníkmi je panel s ikonami modifikovaný a používateľ nemá možnosť prejsť na inú kartu a jeho jedinou možnosťou je výber jedného z aktuálneho zobrazených a registrovaných skleníkov. Po zvolení konkrétneho skleníka je používateľovi umožnené prepínať medzi všetkými kartami. Avšak v prípade ak používateľ sa znova vráti späť na domovskú stránku so všetkými jeho skleníkmi možnosť prepínania medzi kartami mu je zablokovaná.

### Používateľské webové rozhranie skleníka

Po vybratí konkrétneho skleníka je používateľovi automaticky načítaná karta so základnými údajmi o skleníku. Používateľovi sú poskytnuté základné informácie o skleníku ako jedinečný identifikátor skleníka, typ mikrokontroléra, ktorý je riadiacim členom vybraného skleníka a dátum registrácie skleníka. Ďalšie údaje, ktoré webové rozhranie poskytuje sú najaktuálnejšie namerané hodnoty. Namerané hodnoty sú rozdelené do dvoch skupín. Prvou sú namerané údaje z vnútra skleníka. Druhá skupina pokrýva záznamy o hodnotách z vonkajšieho prostredia skleníka. Každá zo skupín môže zobrazovať údaje o teplote, vlhkosti vzduchu, vlhkosti pôdy a nameraných hodnotách oxidu uhličitého. Na grafickú reprezentáciu údajov sú využité grafy (Príloha B). V prípade ak niektoré z údajov chýbajú, graf je prázdny.

Poslednou funkciou, ktorá je používateľovi poskytovaná v rámci tejto karty je manuálne riadenie akčných členov s využitím webové rozhrania.  Na ovládanie sú využívané dva prepínače na oddelenie riadenia odvetrávania a zavlažovania skleníka. Princíp vzdialeného riadenia akčných členov je postavený na využití MQTT klienta. Pri zmene stavu prepínača je vytvorená dátová štruktúra vo formáte JSON, ktorej obsah pozostáva z kľúča s názvom *„requested“* a hodnota kľúča je aktuálna hodnota prepínača. Vytvorený JSON je následné publikovaný do príslušnej témy v závislosti, s ktorým z prepínačov bolo manipulované. V prípade skleníka s identifikátorom 0 pri manipulácii s prepínačom s názvom Window, JSON bude publikovaný do *„Greenhouse/window/0“.* Pri prepínači s názvom Irrigation je publikovaná téma *„Greenhouse/irrigation/0“.*

### Archív nameraných dát

Náhľad do archívu nameraných dát nám poskytuje tretia a zároveň posledná karta s ikonou hodín. Po kliknutí na ikonu je používateľovi načítaný obsah so štyrmi grafmi, ktoré reprezentujú každú z podporovaných veličín. Každý z grafov poskytuje zobrazovanie archívnych záznamov aj z vnútra, aj z vonkajšieho prostredia skleníka. Tieto záznamy pre konkrétnu veličinu sú umiestnené do jedného grafu s možnosťou filtrácie zobrazenia dát v závislosti od pozície merania. Opísaný filter je zároveň legendou grafu a je umiestnený v hornej časti každého grafu. Hodnoty s označením „Inside“ podfarbené zelenou farbou reprezentujú archívne zábery veličiny namerané vo vnútri skleníka. Druhou sériou hodnôt sú dáta s označením „Outside“ podfarbené farbou červenou. Tieto dáta graficky znázorňujú priebeh konkrétnej veličiny zozbierane mimo vnútra skleníka.

#### Obrázok, na ktorom je tabuľka Automaticky generovaný popisFiltrácia archívnych dát

Obr. 4.14 – Používateľské rozhranie zobrazujúce hodnoty za posledných 30 minút

Filtrovanie a následné grafické zobrazenie nameraných dát je možné vykonať s využitím filtrovaniach tlačidiel, ktoré sú umiestnené v hornej časti. Používateľ ma na výber zo štyroch tlačidiel. Posledné stlačené tlačidlo s významom aktivovania príslušného filtra je podfarbené zelenou farbou. Tlačidla „MINUTES“, „HOURS“, „DAYS“ filtrujú dáta v rozsahu, kde dolná hranica intervalu zodpovedá hodnote aktuálneho času mínus hodnota z posúvača v závislosti od zvoleného tlačidla. Nasledujúca tabuľka zobrazuje hodnoty, ktoré je možné docieliť s využitím tlačidiel.

Tab. .7 – Popis funkcie tlačidiel v používateľskom rozhraní

|  |  |  |  |
| --- | --- | --- | --- |
| Tlačidlo | Krok | Minimum | Maximum |
| MINUTES | 5 [minút] | 10 [minút] | 60 [minút] |
| HOURS | 1 [hodina] | 1 [hodina] | 24 [hodín] |
| DAYS | 1 [deň] | 1 [deň] | 7 [dni] |

Príklad využitia tlačidiel môže nastať ak používateľ požaduje zobrazenie nameraných hodnôt za poslednú pol hodinu. V tom prípade používateľ použije tlačidlo „MINUTES“ a za pomoci posúvača, s krokom 5 minút, nastaví hodnotu 30 min. Po nastavení týchto hodnôt back-end aplikácie vo vývojovom prostredí Node-Red spracuje vstupné dáta a vytvorí sa požiadavka na databázu. Požiadavka je vytvorená so špecifickými parametrami definujúci rozsah požadovaných hodnôt. Odpoveď z databázy sa následné spracováva a pripravujú sa dáta pre grafy, ktoré po spracovaní sú zobrazené používateľovi podľa jeho definovaných kritérií. Rovnaký princíp funkčnosti je použitý aj pri ostatných tlačidlách, ako je napríklad zobrazenie nameraných hodnôt za posledné 2 dni s využitím tlačidla „DAYS.

Posledné tlačidlo k filtrovaniu dát, ktoré je ponúkané používateľovi je tlačidlo „CUSTOM“. Tlačidlo umožňuje používateľovi filtrovať dáta na základe zvolených dni. Po stlačení tohto tlačidla sú používateľovi poskytnuté dva vstupné polia na nastavenie rozsahu, od ktorého dátumu požaduje zobraziť dáta a druhým poľom, v ktorom je definovaný deň, po ktorom sa dáta už zobrazovať nebudú. Systém zároveň ale nedovoľuje používateľovi zvoliť dátum, ktorý ešte nenastal. V prípade ak sa používateľ pokúsi o nastavenie neplatného dátumu je vo vstupnom poli nastavený na aktuálny a používateľ je informovaný dialógovou správou o invalidnom dátume. Po poskytnutí hraničných dátumov používateľom je potrebné zo strany používateľa aby zvolené dátumy potvrdil stlačením tlačidla „CONFIRM“. Po stlačení sa vstupné dáta spracujú a proces vytvorenia požiadavky na databázu a prípravy dát pre grafické zobrazenie je totožný ako bol opísaný v odseku vyššie.

## Obrázok, na ktorom je text, vnútri Automaticky generovaný popisDatabázový systém

Obr. 4.15 – Používateľom definovaný rozsah zobrazenia údajov

Súčasťou navrhnutého systému je databáza typu NoSQL MongoDB, ktorá už bola niekoľko krát spomenutá. Prevádzkovaná je na platforme docker a pri inicializácii bol využitý dokrový obraz mongo. Kontajneru bola priradená IP adresa 172.20.0.4 a je súčasťou aplikačnej siete s využitím mapovaním lokálneho portu 27017 na port 26003.

Databáza v navrhnutom systéme disponuje dvoma kolekciami. Kolekcia s názvom **Accounts** a kolekcia **Greenhouses.** Accounts kolekcia slúži na evidenciu používateľov.  Pri evidencii používateľov každý zaevidovaný používateľ musí mať priradené používateľské meno, heslo a prázdny zoznam identifikátor reprezentujúci registrované skleníky. Aktuálna verzia implementácie využíva z tejto kolekcie iba používateľské meno a zoznam registrovaných skleníkov. Priradené heslo ku používateľskému menu je do systému implementované pre budúce potreby autentifikácie používateľa. Pri opise prístupu ku používateľskému rozhraniu bolo spomínané, že pri prístupe je potrebné zadať ako parameter URL meno používateľa. Z poskytnutého mena sa vytvorí požiadavka na databázu a vyhľadá sa používateľ s jeho registrovanými skleníkmi. V odpovedi databáze na požiadavku sa nachádza následné zoznam, ktorý je spracovaný a graficky reprezentovaný na prvej karte používateľského rozhrania.

Kolekcia **Greenhouses** obsahuje všetky základné údaje aj namerané dáta všetkých registrovaných inteligentných skleníkov. Každý registrovaný skleník disponuje základnými údajmi a zoznamami nameraných dát. Jedinečný identifikátor, typ riadiaceho člena a deň registrácie sú základné údaje a sú dopytované pri načítavaní používateľského rozhrania na karte 1, ktorá zobrazuje údaje skleníka ako aj aktuálne namerané hodnoty. Štruktúra dokumentov taktiež obsahuje aj zoznam dokumentov, kde sú uložené záznamy z merania. Pri dokumentoch zo zoznamu sa štruktúra mení v závislosti od zaregistrovaného skleníka, preferencii používateľa a meraných dát. V prípade ak registrovaný skleník meria hodnoty vo vnútri aj vo svojom okolí tak sú vytvorené dva zoznamy oddeľujúce namerané hodnoty v závislosti od pozície snímačov. Následné každý zo zoznamov obsahuje jednotlivé záznamy meraní, ktoré obsahujú informáciu o čase merania a hodnoty, ktoré sú špecifické pre skleník ako teplota, vlhkosť pôdy, vlhkosť vzduchu a CO2.

# Testovanie

Pre účely testovania bol vytvorený prvý prototyp skleníka. Skleník bol postavený z dreveného materiálu s rozmerný 400 x 300 mm. Výška skleníka vrátane strechy je 290 mm. Skleník obsahuje dvoch ESP klientov, ktorí merajú vlhkosť pôdy, teplotu, vlhkosť vzduchu a jeden z klientov aj oxid uhličitý. Okolie skleníka monitoruje jeden ESP klient, ktorý meria teplotu a vlhkosť vzduchu. Pri zariadení je taktiež umiestnený centrálny uzol.

## Testovanie akčných členov

Na testovanie akčných členov bolo do zariadenie nainštalované potrubie napojené na ponorné čerpadlo a taktiež bol vytvorený systém na manipuláciu s oknom. Systém na manipuláciu s oknom bol namodelovaný a vytlačený za pomoci 3D tlačiarne. Model je možné vidieť na obrázku nižšie. Princíp systému spočíva vo vysunutí plastovej tyče počas rotácie motora. Spätne zatvorenie okna zabezpečil magnet umiestnený na okne a druhy magnet umiestnený na pevnej časti skleníka.

Bolo vykonané manuálne testovanie kedy z používateľského rozhrania bol zaslaný požiadavka na spustenie odvetrávania, čo spôsobilo spustenie motora a otvorenie okna. Úspešné bolo aj testovanie závlahy kedy po zaslaní požiadavky z používateľského prostredia bol zopnutý obvod pre ponorné čerpadlo. Testovanie bolo úspešné vykonané aj s opačným zámerom vypnúť závlahu a zavrieť okno.

## Manuálne testovanie softvérových modulov

Obr. 5.1 – 3D model systému na manipuláciu s oknom

Kapitola popisuje postup pri manuálnom testovaní softvérových komponentov. Opisuje sa postup testovania, očakávané správanie softvérového komponentu a záverom je výsledok testu.

Test – ESP klient a pripojenie ku BLE serveru

**Účel testovania:** Účelom testovania je overenie správnej funkčnosti mechanizmu, ktorý zabezpečuje znovu pripojenie ESP klienta ku BLE serveru v prípade straty konektivity počas bežného fungovania systému.

**Postup testovania:** Centrálny uzol bude spustený a bude bežať BLE server. Centrálny uzol bude zároveň vysielať svoje bluetooth dáta a je pripravený nadviazať komunikáciu s ESP klientom. ESP klient sa následne pripojí ku zdroju napájania. Po uplynutí 10 sekúnd bude centrálny uzol odpojení z napájania. Uplynutím ďalších 10 sekúnd sa centrálny uzol pripojí znova k napájaniu a spusti sa BLE server.

**Očakávané správanie:** Očakávané správanie klienta po pripojení k napájaniu je inicializácia spojenia s BLE server (indikácia RGB diódou 3x bliknutia zelenou farbou). Počas 10 sekúnd pred odpojením centrálneho uzla ESP klient bude indikovať modrou farbou, že je pripojený ku BLE serveru. Následné po vyradení centrálneho uzla z prevádzky sa farba diódy zmení na červenú a počas nedostupnosti BLE servera, ESP klient bude indikovať stav pripojenia červenou farbou. Po znovu pripojení centrálneho uzla sa ESP klient znova pripojí a RGB dióda bude svietiť modrou farbou.

**Výsledok testu:** Správanie ESP klienta bolo totožné s očakávaním. Po pripojení napájania sa pripojil ku BLE serveru. Po strate spojenia signalizoval červeným svetlom stratu spojenia a po znovu spustení BLE servera sa pripojenie obnovilo.

Test – Centrálny uzol a prístup do Wi-Fi siete

**Účel testovania:** Účelom testovania je potvrdiť korektnú implementáciu správania systému pri nedostupnosti Wi-Fi siete alebo náhlej straty spojenia a jeho znovu nadviazania.

**Postup testovania:** Prvým krokom je vytvorenie Wi-Fi prístupového bodu na lokálnom počítači. Dôvod vytvorenia na PC je práve pre jeho jednoduchšiu manipuláciu počas doby testovania. Následné sa pripojí klient ku zdroju napájania. Spustenie vytvoreného Wi-Fi prístupového bodu sa vykoná až po 45 sekundách od pripojenia klienta ku zdroju napájania. Spustením prístupového bodu sa prechádza do druhej fázy testovania, a to náhla strata spojenia. Systém sa najprv ponechá v normálnom režime po dobu 5 minút. Po uplynutí 5 minút sa Wi-Fi prístupový bod vypne a po 30 sekundách sa opäť zapne.

**Očakávané správanie:** Popripojení klienta k napájaniu sa spusti BLE server a nasleduje proces pripájania do siete. Signalizácia žltým svetlom na RGB dióde. Po neúspešnom pripojení sa zmení farba diódy na červenú a systém sa na jednu minútu uspí. Po prebudení sa pokúsi znova pripojiť. Za predpokladu, že Wi-Fi už bude k dispozícii, klient sa pripojí a úspešné pripojenie bude signalizovať modrým svetlom. Počas doby v normálnom režime klient bude signalizovať 2 krát bliknutím modrým svetlom, že je pripojený. Po strate spojenia sa systém bude znova pokúšať pripojiť k sieti a počas stavu, kedy klient nie je pripojený do siete bude RGB dióda signalizovať stav červenou farbou.

**Výsledok testu:** Správanie klienta bolo rovnaké ako sa očakávalo. Počas nedostupnej siete sa klient pokúšal pripojiť a signalizoval to žltým svetlom. Po neúspechu, RGB dióda svietila na červeno. Po definovanom čase sa spustil Wi-Fi prístupový bod a zariadenie sa úspešne pripojilo. Po piatich minútach pri strate spojenia a znova spustením Wi-Fi sa podarilo klientovi podarilo obnoviť pripojenie do siete.

Test – Prenos dát zo senzorov až po vizualizáciu v používateľskom rozhraní

**Účel testovania:** Účelom testovania je potvrdiť úspešný prenos dát zo senzorov až po ich vizualizáciu v používateľskom rozhraní.

**Postup testovania:** Vymazanie všetkej doterajšej histórie daného skleníka (klienta) ako aj jeho úplne vymazanie z kolekcie Greenhouses. Dôvod takéhoto kroku je docielenie prvotnej registrácie klienta a dáta v databáze budú zodpovedať všetkým nameraným dátam z testu. Systém bude spustený po dobu 120 minút a následne dáta budú vizuálne zobrazené v používateľskom rozhraní.

**Očakávané správanie:** Pri tomto teste sa očakáva kompletne zobrazenie všetkých dát v používateľskom rozhraní po dobu spusteného systému.

**Výsledok testu:** Výsledkom testu je grafické zobrazenie aktuálne nameraných hodnôt v skleníku a taktiež všetky namerané dáta je možné získať na karte zobrazujúce grafy nameraných hodnôt v používateľskom rozhraní.

Záver

Primárnym cieľom celej diplomovej práce bolo navrhnúť modulárny systém. Obsah kapitoly, ktorá sa venuje implementácií navrhnutého systému a kapitola testovanie dokazujú korektné fungovanie navrhnutého systému. Systém poskytuje používateľovi možnosť konfigurácie meraných hodnôt z predom definovanej množiny, zasiela dáta centrálnemu uzlu a poskytuje používateľovi rozhranie na grafickú reprezentáciu zosnímaných dát.

Testovanie systému preukázalo správnosť výberu bezdrôtovej technológie pri distribúcii dát z ESP klientov ku centrálnemu uzlu. Vybraná technológia systému taktiež zvýšila bezpečnosť a znemožnila potencionálnym útočníkom prístup z vonkajšej siete za predpokladu využitia technológie Wi-Fi. Technológia Wi-Fi bola využitá iba pri centrálnom uzly, ktorý ale nemá priamy prístup ku senzorom v rámci skleníka. Testovaním bola taktiež potvrdená funkčnosť používateľského rozhrania s možnosťou prepínania medzi viacerými skleníkmi, prehľadom aktuálnych dát a grafickým reprezentovaním všetkých doposiaľ nameraných dát.

Žiadny z navrhovaných systémov nie je perfektný pri svojom vzniku a tento navrhnutý systém nie je žiadnou výnimkou. Pri testovaní bol objavený jeden z najväčších nedostatkov systémom a to v prípade resetu systému nemožno zistiť aktuálny stav akčných členov. Nevyhnutnosťou ďalšej verzie návrhu by boli hardvérové komponenty, ktoré by umožňovali detegovať aktuálny stav členov.

Účelom práce bolo poskytnúť prostredie na pestovanie byliniek s využitím IoT systémov a bezdrôtových technológiou a táto práca je dôkazovom toho, že sa to podarilo. S využitím zopár senzorov a niekoľkých mikrokontrolérov bol vytvorený systém pre používateľa na pestovanie byliniek bez námahy a výsledkom je funkčný systém, ktorý zároveň aj poukázal na miesta, kde by sa dal vylepšiť a stal sa ešte inteligentnejším.

Zoznam použitej literatúry

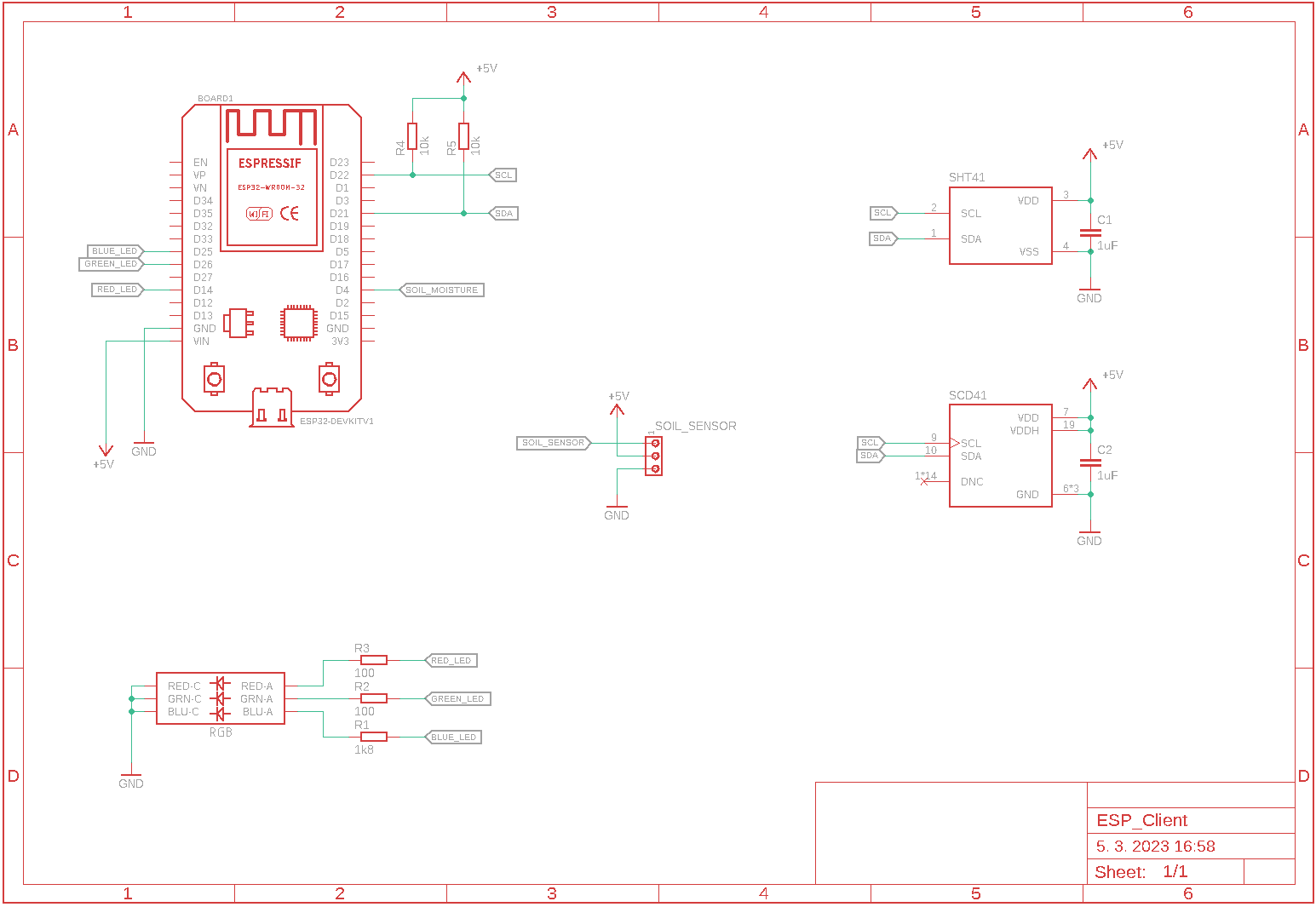
1. **Cisco.** (n.d.). What Is Wi-Fi?. [online]. [cit. 2023-02-26]. Dostupné na: <https://www.cisco.com/c/en/us/products/wireless/what-is-wifi.html>
2. **Symmetry Electronics Team.** 2019. The Difference Between Classic Bluetooth and Bluetooth Low Energy. [online]. [cit. 2023-02-25]. Dostupné na: <https://www.symmetryelectronics.com/blog/the-difference-between-classic-bluetooth-and-bluetooth-low-energy/>
3. **Bluetooth SIG.** (n.d.). Bluetooth Technology Overview. [online]. [cit. 2023-02-25]. Dostupné na: <https://www.bluetooth.com/learn-about-bluetooth/tech-overview/>
4. **Bluetooth SIG.** (n.d.). Understanding Bluetooth Range. [online]. [cit. 2023-02-25]. Dostupné na: <https://www.bluetooth.com/learn-about-bluetooth/key-attributes/range/>
5. **Curt Franklin, Chris Pollette.** 2021. How Bluetooth Works. [online]. [cit. 2023-02-25]. Dostupné na: <https://electronics.howstuffworks.com/bluetooth.htm>
6. **Jacob Arellano.** 2019. Bluetooth vs. Wi-Fi for IoT: Which is Better?. [online]. [cit. 2023-02-28]. Dostupné na: <https://www.verytechnology.com/iot-insights/bluetooth-vs-wi-fi-for-iot-which-is-better>
7. **Espressif Inc.** 2018. ESP32-WROOM-32 (ESP-WROOM-32) Datasheet. [online]. [cit. 2023-03-01]. Dostupné na: <https://www.mouser.com/datasheet/2/891/esp-wroom-32\_datasheet\_en-1223836.pdf>
8. **SENSIRION.** 2023. Datasheet – SHT4x. [online]. [cit. 2023-04-03].<https://sensirion.com/media/documents/33FD6951/640B22DB/Datasheet\_SHT4x.pdf>
9. **SENSIRION.** 2022. SCD4x. [online]. [cit. 2023-04-03]. <https://sensirion.com/media/documents/E0F04247/631EF271/CD\_DS\_SCD40\_SCD41\_Datasheet\_D1.pdf>
10. **Vijay Kanade.** 2022. What Is SQL? Definition, Elements, Examples, and Uses in 2022. [online]. [cit. 2023-02-26]. Dostupné na: <https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-sql/>
11. **Oracle Cloud Infrastructure.** (n.d.). What is MySQL?. [online]. [cit. 2023-02-26]. Dostupné na: <https://www.oracle.com/mysql/what-is-mysql/>
12. **Lauren Schaefer.** (n.d.). What is NoSQL?. [online]. [cit. 2023-02-28]. Dostupné na: <https://www.mongodb.com/nosql-explained>
13. **MongoDB.** (n.d.). Understanding the Different Types of NoSQL Databases. [online]. [cit. 2023-02-28]. Dostupné na: <https://www.mongodb.com/scale/types-of-nosql-databases>
14. **Lauren Schaefer.** (n.d.). NoSQL vs. SQL Databases. [online]. [cit. 2023-02-28]. <https://www.mongodb.com/nosql-explained/nosql-vs-sql>
15. **Rajguru Electronics.** (n.d.). DC Mini Submersible Water Pump. [online]. [cit. 2023-03-05]. Dostupné na: <https://module-center.com/administrator/files/UploadFile/dc-mini-submersible-water-pump.pd f>
16. **Kiatronics.** (n.d.). 4 Phase ULN2003 Stepper Motor Driver PCB. [online]. [cit. 2023-03-05]. Dostupné na: <https://www.electronicoscaldas.com/datasheet/ULN2003A-PCB.pdf>
17. **Kevin Townsend.** 2014. GATT. [online]. [cit. 2023-03-23]. Dostupné na: <https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gatt?fbclid=IwAR2IyxDAuAi4\_Zq0eKyTTEm-o8ZYag8l2ofXPCbd314aN3P91pLxJVU52Vw>
18. **Espressif.** (n.d.). Project Configuration. [online]. [cit. 2023-03-23]. Dostupné na: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/kconfig.html?fbclid=IwAR05aADqLKcXFpr1oNvUsAxpmHlAIITdqc5DYJXEVGUK1tWdCs-YYkX7dT8>
19. **Chiradeep BasuMallick.** 2022. What Is Network Time Protocol (NTP)? Meaning, Working, Benefits, and Challenges. [online]. [cit. 2023-03-23]. Dostupné na: <https://www.spiceworks.com/tech/networking/articles/what-is-network-time-protocol/?fbclid=IwAR2uu-vhxY3onIhYTWjd4gOU2G4ni1-NE00YqW4fKzCqXNY9E9ISgAp68HQ>
20. **The HiveMQ Team.** 2015. Introducing the MQTT Protocol - MQTT Essentials: Part 1. [online]. [cit. 2023-03-23]. Dostupné na: <https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/>
21. **The HiveMQ Team.** 2015. Publish & Subscribe - MQTT Essentials: Part 2. [online]. [cit. 2023-03-23]. Dostupné na: <https://www.hivemq.com/blog/mqtt-essentials-part2-publish-subscribe/>
22. **The HiveMQ Team.** 2019. MQTT Client and Broker and MQTT Server Connection Establishment Explained - MQTT Essentials: Part 3. [online]. [cit. 2023-03-23]. Dostupné na: <https://www.hivemq.com/blog/mqtt-essentials-part-3-client-broker-connection-establishment/>
23. **IBM.** (n.d.). What is Docker?. [online]. [cit. 2023-03-25]. Dostupné na: <https://www.ibm.com/topics/docker?fbclid=IwAR2DTkqxBiZ-5iVoqkLn2LSL5MzM-M8c\_CMmebTmq5j6nmkOiqJBLENsLlc>
24. **aqua.** (n.d.). Docket Networking. [online]. [cit. 2023-03-25]. Dostupné na: <https://www.aquasec.com/cloud-native-academy/docker-container/docker-networking/?fbclid=IwAR36JRNDCSt875khfgGlneuDOT-Jp9sV1jKdsxJh6CWvird9ChUx582bIk0>

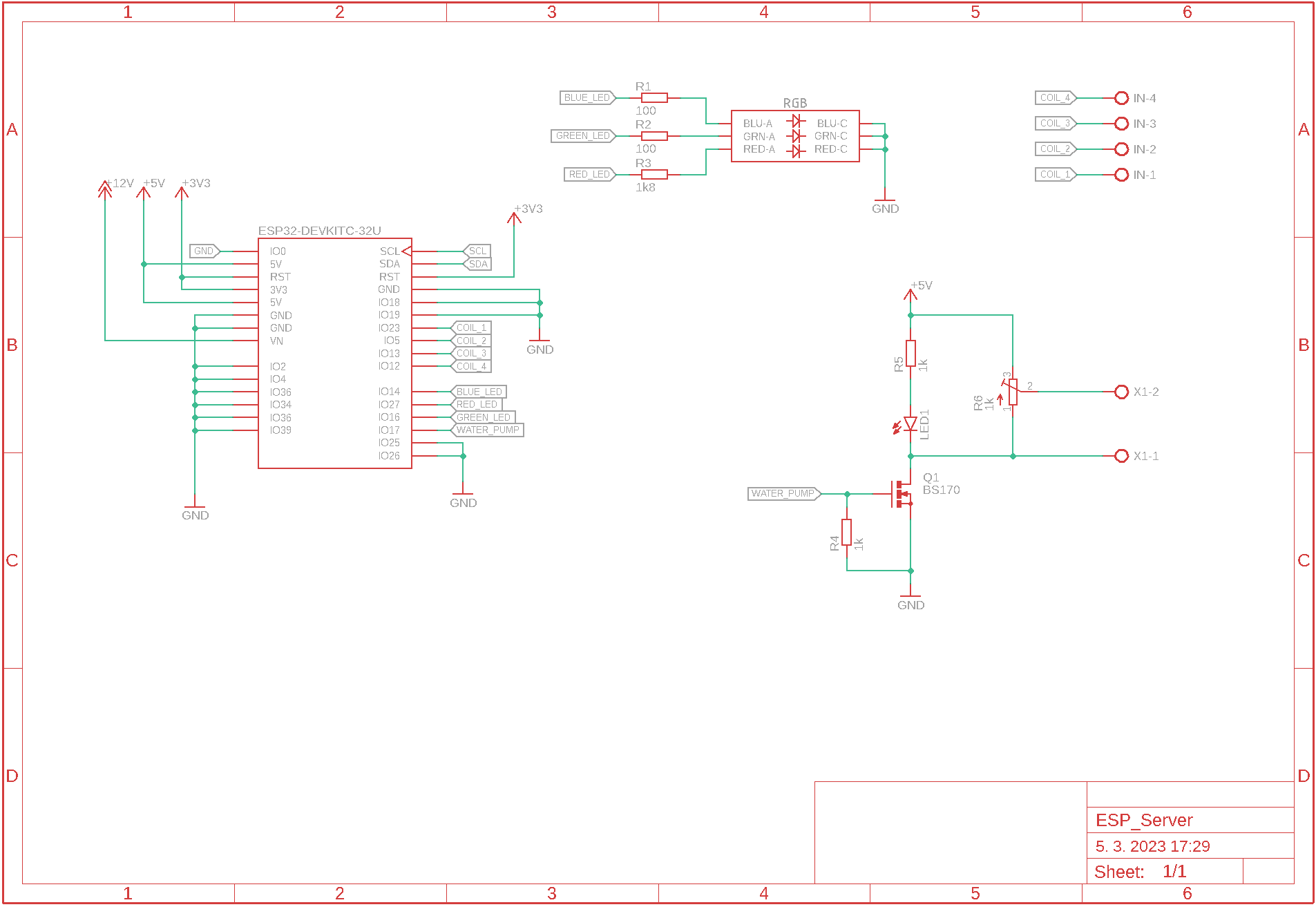
**Prílohy**

Zoznam príloh

[Príloha A | Schematické zapojenia 77](#_Toc131266230)

[Príloha B | Karty v používateľskom rozhraní 78](#_Toc131266231)

1. Schematické zapojenia



1. Obrázok, na ktorom je text, drevo, zelenina

   Automaticky generovaný popisKarty v používateľskom rozhraní

1. Počítačový softvér, ktorého zdrojový kód je prístupný a s licenciou na jeho distribúciu, študovanie, úpravu a ďalšie operácie spojené s využitím a vylepšením. [↑](#footnote-ref-1)