|  |  |  |
| --- | --- | --- |
|  | | |
| **Závěrečná studijní práce**  **dokumentace** | | |
| **Dálkové ovládání žaluzií** | | |
| Pavel Košnar | | |
|  | | |
|  | |  |
| **Obor:** | 18-20-M/01 INFORMAČNÍ TECHNOLOGIE  se zaměřením na počítačové sítě a programování | |
| **Třída:**  **Školní rok:** | IT4  2022/2023 | |

#### Poděkování

Chtěl bych poděkovat panu učiteli Ing. Petru Grussmannovi za pomoc s projektem a jeho náležité připomínky. Také děkuji svému otci za pomoc se složením hardwaru a sponzoring.

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité   
informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové   
a umělecké škole v Opavě, Praskova 399/8.

V Opavě 31. 12. 2022

*podpis autora práce*

**ANOTACE**

Práce slouží k bezdrátovému ovládání žaluzií pomocí ESP 8266. Vše funguje přes MQTT server Home Assistanta. V Home Assistantu se po úspěšném spojení objeví tlačítka k ovládání. Lze ovládat posun žaluzií nahoru či dolů a zastavit v jakoukoli chvíli. Je také možné žaluzie naklápět. Všechny informace o probíhajících procesích a stavu žaluzií lze pozorovat v MQTT záložce Home Assistanta, kde je také možné žaluzie ovládat pomocí příkazů. Pokud uživatel nemá k Home Assistantu přístup, může zařízení ovládat přes webovou stránku. Na webové stránce však není stav žaluzií.

**ANNOTATION**

The project is used to control blinds wirelessly using an ESP 8266. Everything works via the Home Assistant's MQTT server. Control buttons will appear in Home Assistant after a successful connection. The movement of the blinds can be controlled up or down and stopped at any moment. It is also possible to tilt the blinds. All information about ongoing processes and the state of the blinds can be observed in the MQTT tab of the Home Assistant, where it is also possible to control the blinds using commands. If the user does not have access to the Home Assistant, they can control the device via the web page. However, there is no status of the blinds on the website.

OBSAH

[Poděkování 2](#_Toc124555989)

[Úvod 5](#_Toc124555990)

[1 Teoretická a metodická východiska 6](#_Toc124555991)

[1.1 Home Assistant a MQTT server 6](#_Toc124555992)

[1.2 Knihovna ArduinoHA 7](#_Toc124555993)

[2 Využité technologie 8](#_Toc124555994)

[2.1 PlatformIO 8](#_Toc124555995)

[2.2 Home Assistant 8](#_Toc124555996)

[2.3 MQTT 8](#_Toc124555997)

[2.4 Little FS 8](#_Toc124555998)

[2.5 Hardware 9](#_Toc124555999)

[3 Způsoby řešení a použité postupy 10](#_Toc124556000)

[3.1 Založení projektu 10](#_Toc124556001)

[3.2 Webová stránka 10](#_Toc124556002)

[3.2.1 Vzhled stránky 10](#_Toc124556003)

[3.2.2 Knihovna ESPAsyncWebServer 11](#_Toc124556004)

[3.3 Funkce k ovládání 12](#_Toc124556005)

[3.3.1 Nahoru a dolů 12](#_Toc124556006)

[3.3.2 Zastavit 14](#_Toc124556007)

[3.3.3 Naklopení nahoru a dolů 14](#_Toc124556008)

[3.3.4 Naklopení o jeden stupeň 17](#_Toc124556009)

[4 Výsledky řešení, výstupy, uživatelský manuál 18](#_Toc124556010)

[4.1 Výsledek projektu 18](#_Toc124556011)

[4.2 Splněné a nesplněné cíle 18](#_Toc124556012)

[Závěr 19](#_Toc124556013)

[Seznam použitýCH INFORMAČNÍCH ZDROJů 20](#_Toc124556014)

Úvod

Mým cílem pro závěrečný projekt bylo posunout se v rámci informatiky zase o kousek dále. Jelikož jsem moc nerozuměl elektrotechnice, rozhodl jsem se rozvinout své znalosti právě v tomto oboru. Bezdrátové ovládání žaluzií se tedy jevilo jako nejrozumnější téma. Projekt totiž obsahuje nejen elektrotechniku, ale i programování v jazyce C++, komunikaci se serverem a logické přemýšlení.

Původně jsem pro svůj projekt chtěl vyvinout ještě vlastní aplikaci, ale z časových důvodů se tak nestalo.

Chtěl jsem se také naučit psát efektivnější funkce, což si myslím, že se mi postupem času povedlo. V kódu projektu je možné vidět postupné zlepšení, co se týče nejen efektivity, ale i rychlosti a jednoduchosti.

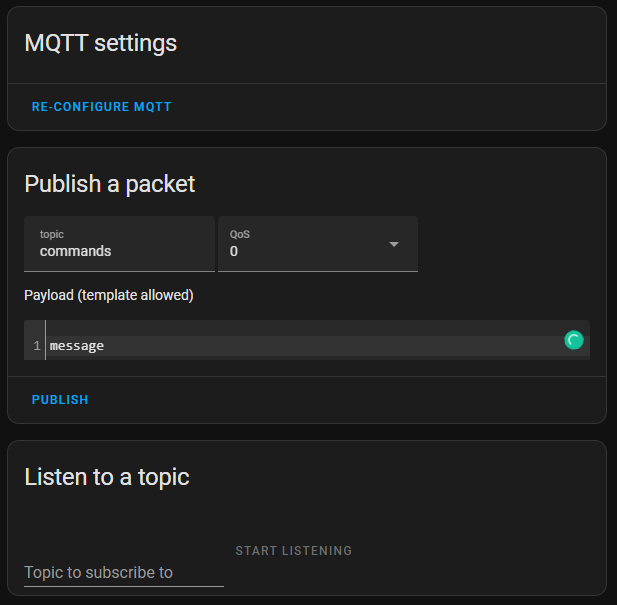
Dále v dokumentaci popisuji postupy a využité technologie.

# Teoretická a metodická východiska

## Home Assistant a MQTT server

Home Assistant poskytuje vhodné prostředí pro operace se zařízeními pro chytrou domácnost. Z ESP 8266 se navazuje spojení s Home Assistantem přes MQTT server, který je běžně dostupný v HA Add-ons. Přes Home Assistant lze také MQTT konfigurovat, posílat a přijímat zprávy.

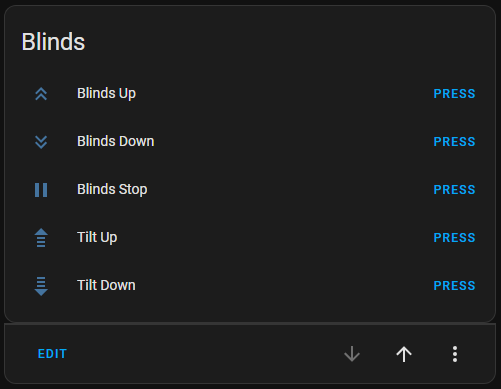
MQTT v Home Assistantu vypadá následovně:



Obrázek .: MQTT v HA

## Knihovna ArduinoHA

Pro připojení ESP 8266 k Home Assistantu je zapotřebí knihovna ArduinoHA. Není sice ani z daleka jediná, ovšem pro můj projekt se jevila jako nejvhodnější volba. Knihovna totiž kromě připojení k MQTT umožňuje také vytvářet vlastní tlačítka. Veškeré informace o tlačítkách posílá serveru v podobě JSON formátu. Home Assistant je poté schopen přijímané informace proměnit v tlačítka, které se zobrazují na Overview (také Dashboard) stránce.



Obrázek .: Tlačítka v Home Assistantu

# Využité technologie

## Hardware

Základním a nejdůležitějším hardwarem mého projektu je ESP 8266 – Wi-Fi mikročip se zabudovaným TCP/IP síťovým softwarem. Dále 80x50cm žaluzie s elektromotorem a dvě spínací relé.

## PlatformIO

Nejoblíbenější IDE řešení pro Microsoft Visual Studio Code. Uživatelsky přívětivé a rozšiřitelné integrované vývojové prostředí se sadou profesionálních vývojových nástrojů, které poskytuje moderní a výkonné funkce pro urychlení a zároveň zjednodušení vytváření a dodávání produktů.

Přes PlatformIO lze jednoduše komunikovat s ESP 8266 a to přímo ve Visual Studio Code. Byla to tedy nejpohodlnější volba pro můj projekt.

## Home Assistant

Home Assistant je bezplatný a open-source software pro domácí automatizaci navržený jako centrální řídicí systém pro inteligentní domácí zařízení se zaměřením na místní ovládání a soukromí. Lze k němu přistupovat prostřednictvím webového uživatelského rozhraní pomocí doprovodných aplikací pro Android a iOS nebo pomocí hlasových příkazů prostřednictvím podporovaného virtuálního asistenta, jako je Google Assistant nebo Amazon Alexa.

## MQTT

MQTT je standardní protokol pro zasílání zpráv OASIS pro internet věcí (Internet of Things). Je navržen jako extrémně lehký přenos zpráv pro publikování/odběr, který je ideální pro připojení vzdálených zařízení. MQTT se dnes používá v celé řadě průmyslových odvětví.

## Little FS

Little FS je vysoce integrovaný souborový systém v Mbed OS, optimalizován pro práci s omezeným množstvím RAM a ROM. Vyhýbá se rekurzi, omezuje dynamickou paměť na konfigurovatelné vyrovnávací paměti a v žádném okamžiku neukládá celý úložný blok do paměti RAM.

Little FS jsem využil pro práci s html, css a javascript soubory v paměti.

# Způsoby řešení a použité postupy

## Založení projektu

Než jsem mohl začít s psaním kódu projektu, bylo zapotřebí si stáhnout a nainstalovat mnoho programů. První a nejdůležitější z nich je Visual Studio Code. Ve VSCode jsem potřeboval různá rozšíření (např. PlatformIO, C/C++ a Python, ten jsem potřeboval stáhnout i přímo do počítače). Dále jsem si nainstaloval a nakonfiguroval Home Assistant přes Virtual Box. Byla také potřeba nainstalovat všechny knihovny, které jsem k projektu použil. To ovšem bylo díky PlatformIO velmi jednoduché, jelikož umožňuje automatické nainstalování knihoven po jejich připsání do „platformio.ini“ souboru. Po všech úspěšných instalacích jsem ještě napsal krátký kód pro rozsvícení LEDek, abych se ujistil, že vše funguje, jak má.

## Webová stránka

Webová stránka projektu slouží ke snadnému a rychlému přístupu k ovládání zařízení. Má proto velmi jednoduchý vzhled a obsahuje pouze pět tlačítek – nahoru, dolů, naklopení nahoru a dolů a zastavit.

### Vzhled stránky



Obrázek .: Vzhled webové stránky

### Knihovna ESPAsyncWebServer

Každé z tlačítek přesměruje uživatele na určitou stránku. Uživatel sice změnu nepozná, ale zařízení dostane potřebnou informaci k provedení určitého úkonu.

Ukázka kódu tlačítek pro přesměrování:

<div *class*="col-6">

<p><a *href*="/up"><button *class*="btn btn-success active btn-block"

*id*="up">UP</button></a></p>

<p><a *href*="/down"><button *class*="btn btn-success active btn-block" *id*="down">DOWN</button></a></p>

<p><a *href*="/stop"><button *class*="btn btn-danger active btn-block" *id*="stop">STOP</button></a></p>

<p><a *href*="/tilt-up"><button *class*="btn btn-primary active btn-block" *id*="tiltUp">TILT UP</button></a></p>

<p><a *href*="/tilt-down"><button *class*="btn btn-primary active btn-block" *id*="tiltDown">TILT DOWN</button></a></p>

</div>

Knihovna ESPAsyncWebServer dokáže poznat, zda se uživatel nachází v určitém adresáři stránky. Pro každý z jednotlivých odkazů je tedy přiřazena funkce, která se spustí ve chvíli, kdy uživatel klikne na tlačítko.

Ukázka kódu vyvolání funkce z jednotlivých odkazů:

server.**on**("/up", HTTP\_GET, [](**AsyncWebServerRequest** \*request){

**movement**(goUp, "up");

    request->**send**(LittleFS, "/index.html", **String**());

  });

  server.**on**("/down", HTTP\_GET, [](**AsyncWebServerRequest** \*request){

**movement**(goDown, "down");

    request->**send**(LittleFS, "/index.html", **String**());

  });

  server.**on**("/stop", HTTP\_GET, [](**AsyncWebServerRequest** \*request){

**stopMovement**("stop", true);

    request->**send**(LittleFS, "/index.html", **String**());

  });

## Funkce k ovládání

### Nahoru a dolů

Funkce nahoru nebo dolů nejprve zjistí, v jaké poloze se žaluzie nachází, poté do motoru posílá signál pro pohyb. Funkce také zjistí, jestli už žaluzie dorazily na konec – tedy úplně nahoru nebo úplně dolů. V tomto případě přestane signál posílat.

Funkce je rozdělena na dvě části:

První nastaví proměnné na potřebné hodnoty pro pohyb požadovaným směrem a pošle zprávu MQTT serveru, že se žaluzie začnou posouvat určitým směrem.

Ukázka kódu funkce:

void **movement**(bool direction, const char\* topic) {

  if (direction == true) {

**stopMovement**("stop", true); //pokud už se žaluzie pohybují daným

směrem, pohyb zastaví

  } else {

    start\_time = **millis**();

    stop = false;

    if (topic == "up") {

      goUp = true;

      goDown = false;

**digitalWrite**(pinDown, **LOW**); //vypne směr dolů a zapne nahoru

**digitalWrite**(pinUp, **HIGH**);

      tiltFully **=** "up";

      mqtt.**publish**(movement\_topic, topic, true); //pošle zprávu mqtt serveru se směrem nahoru

    }

    else if (topic == "down") {

      goDown = true;

      goUp = false;

**digitalWrite**(pinUp, **LOW**); //vypne směr nahoru a zapne dolů

**digitalWrite**(pinDown, **HIGH**);

      tiltFully **=** "down";

      mqtt.**publish**(movement\_topic, topic, true); //pošle zprávu mqtt serveru se směrem dolů

    }

  }

}

Druhá běží v cyklu, kontroluje a aktualizuje polohu žaluzií. Každých 5 minut a při pohybu nahoru či dolů také posílá zprávu MQTT serveru s aktuální polohou v procentech.

Ukázka kódu funkce:

void **state**() {

  if (!stop) {

    if (goUp == true) {

      if (**millis**() > start\_time + 220) { //každých 220 milisekund

        current\_position += 20 / path\_length; //aktualizuje polohu žaluzií

        if (current\_position > 100) {

          current\_position = 100;

        }

        if (current\_position == 100) { //v případě, že žaluzie dojedou

**stopMovement**("top", true); //nahoru, pohyb zastaví

        } else {

          start\_time = **millis**();

        }

      }

    }

    else if (goDown == true) { //stejná jako minulá, akorát dolů

      if (**millis**() > start\_time + 200) { //aktualizuje každých 200ms,

        current\_position -= 20 / path\_length; //protože pohyb nahoru trvá déle

        if (current\_position < 0) {

          current\_position = 0;

        }

        if (current\_position == 0) {

**stopMovement**("bottom", true);

        } else {

          start\_time = **millis**();

        }

      }

    }

    if (goUp == true || goDown == true) {

      if (**millis**() > state\_time + 100) {

        state\_time = **millis**();

        state\_message = **itoa**(current\_position, buffer, 10);

        mqtt.**publish**(state\_topic, state\_message, true);

      } //pošle zprávu mqtt serveru s aktuální polohou

    } else {

      if (**millis**() > state\_time + 300000) {

        state\_time = **millis**();

        state\_message = **itoa**(current\_position, buffer, 10);

        mqtt.**publish**(state\_topic, state\_message, true);

      } //každých 5 minut pošle zprávu mqtt serveru s aktuální polohou

    }

  }

}

### Zastavit

Tato funkce je velmi jednoduchá a stručná. Zahrnuje pouze nastavení proměnných, přestane posílat signál do motoru a pošle zprávu MQTT serveru s důvodem zastavení. Důvody mohou být stisknutí tlačítka nebo dojezd žaluzií úplně nahoru či dolů.

Ukázka kódu funkce:

void **stopMovement**(const char\* reason, bool message) {

  goUp = false;

  goDown = false;

  stop = true;

  tiltDirection **=** "stop";

**digitalWrite**(pinUp, **LOW**);

**digitalWrite**(pinDown, **LOW**);

  if (message) {

    mqtt.**publish**(movement\_topic, reason, true);

  }

} //zastaví veškeré pohyby žaluzií

### Naklopení nahoru a dolů

Napsat vhodnou funkci pro naklápění žaluzií byl asi ten nejsložitější úkol, co se týče logiky. Nejen, že žaluzie nelze naklápět, zatímco jsou v pohybu, ale taky je před každým pohybem nahoru či dolů potřeba žaluzie naklopit do vhodné polohy pro pohyb daným směrem. Ještě je potřeba neustále kontrolovat úhel naklopení žaluzií, protože uživatel může proces kdykoliv přerušit tlačítkem „stop“.

Problém jsem vyřešil rozdělením do tří funkcí:

První je opět jednoduchá – zase jde pouze o nastavení potřebných proměnných a poslání signálu do motoru.

Ukázka kódu funkce:

void **startTilting**(**String** direction) {

  tiltDirection **=** direction;

  stop = true;

  tiltTime = **millis**() + 400;

  pauseTime = tiltTime + 800;

  if (direction **==** "up") {

**digitalWrite**(pinUp, **HIGH**);

  }

  else if (direction **==** "down") {

**digitalWrite**(pinDown, **HIGH**);

  }

} //nastaví potřebné proměnné k naklápění

Druhá funkce dohlíží na správný chod naklápění – 400ms posílat signál do motoru, 800ms pauza. Vždy při pauze pošle zprávu MQTT serveru s aktuální polohou naklopení v procentech.

Ukázka kódu funkce:

void **tiltMovement**(**String** tilt) {

  if (tilt **!=** "stop" && stop) {

    if (tiltTime < **millis**() && pauseTime > **millis**()) {

      tiltTime = pauseTime + 400; //zastaví naklápění po 300ms

      if (tilt **==** "up") {

**digitalWrite**(pinUp, **LOW**);

        tilt\_position += 10;

      }

      else if (tilt **==** "down") { //aktualizuje polohu naklopení

**digitalWrite**(pinDown, **LOW**);

        tilt\_position -= 10;

      }

      if (tilt\_position >= 100) {

        tiltDirection **=** "stop";

        tilt\_position = 100;

      }

      else if (tilt\_position <= 0) {

        tiltDirection **=** "stop";

        tilt\_position = 0;

      }

      tilt\_message = **itoa**(tilt\_position, buffer, 10);

      mqtt.**publish**(tilt\_topic, tilt\_message, true);

    }

    else if (pauseTime < **millis**()) { //zapne naklápění po 600ms

      pauseTime = tiltTime + 800;

      if (tilt **==** "up") {

**digitalWrite**(pinUp, **HIGH**);

      }

      else if (tilt **==** "down") {

**digitalWrite**(pinDown, **HIGH**);

      }

    }

  }

}

Třetí funkce pracuje pouze v případě, že byl zadán příkaz posunout žaluzie nahoru či dolů. Má za úkol naklopit žaluzie do správné polohy pro pohyb daným směrem. Při započetí i při dokončení naklápění informuje MQTT server.

Ukázka kódu funkce:

void **fullTilt**() { //rozhodne, na jakou stranu naklopit a poté akci provede

  if (tiltFully **==** "up" && current\_position >= 100) {

    tiltFully **=** "no";

  }

  else if (tiltFully **==** "down" && current\_position <= 0) {

    tiltFully **=** "no";

  }

  if (tiltFully **==** "up") {

    if (!stop && tilt\_position < 100) {

      stop = true;

      fullTiltTime = **millis**();

      mqtt.**publish**(movement\_topic, "Tilting up...", true);

    }

  }

  else if (tiltFully **==** "down") {

    if (!stop && tilt\_position > 0) {

      stop = true;

      fullTiltTime = **millis**();

      mqtt.**publish**(movement\_topic, "Tilting down...", true);

    }

  }

  if (stop) { //aktualizuje stav naklopení

    if (tiltFully **==** "up" && fullTiltTime + 400 < **millis**()) {

      tilt\_position += 10;

      if (tilt\_position > 100) {

        tilt\_position = 100;

      }

      fullTiltTime = **millis**();

    }

    else if (tiltFully **==** "down" && fullTiltTime + 400 < **millis**()) {

      tilt\_position -= 10;

      if (tilt\_position < 0) {

        tilt\_position = 0;

      }

      fullTiltTime = **millis**();

    }

    if (tilt\_position >= 100 && tiltFully **==** "up" || tilt\_position <= 0 && tiltFully **==** "down") {

      stop = false;

      tiltFully **=** "no";

      mqtt.**publish**(movement\_topic, "Done", true);

    }

  }

}

### Naklopení o jeden stupeň

Velmi stručná a jednoduchá funkce. Funguje podobně jako funkce naklápění, ale posune se pouze o kousek a zůstane stát.

Tvořena ze dvou funkcí, kdy první opět jen nastavuje proměnné, druhá žaluzie naklopí a poté zastaví.

Ukázka kódu obou funkcí:

void **startStepTilt**(**String** direction) { //nastavení potřebných proměnných

  stepTiltDirection **=** direction;

  stepTiltTime = **millis**() + 800;

  if (direction **==** "up") { //zapne naklápění na správnou stranu

**digitalWrite**(pinUp, **HIGH**);

  }

  else if (direction **==** "down") {

**digitalWrite**(pinDown, **HIGH**);

  }

}

void **stepTilt**(**String** direction) { //po 400ms naklápění vypne

  if (direction **!=** "none") {

    if (stepTiltTime < **millis**()) {

      stepTiltDirection **=** "none";

      if (direction **==** "up") {

        tilt\_position += 20;

        if (tilt\_position > 100) {

          tilt\_position = 100;

        }

**digitalWrite**(pinUp, **LOW**);

      }

      if (direction **==** "down") {

        tilt\_position -= 20;

        if (tilt\_position < 0) {

          tilt\_position = 0;

        }

**digitalWrite**(pinDown, **LOW**);

      }

      tilt\_message = **itoa**(tilt\_position, buffer, 10);

      mqtt.**publish**(tilt\_topic, tilt\_message, true);

    }

  }

}

# Výsledky řešení, výstupy, uživatelský manuál

## Výsledek projektu

Projekt jsem vymyslel tak, aby uživatel mohl žaluzie sám zprovoznit a spojit s Home Assistantem. Na GitHubu mého projektu je proto návod, jak Home Assistant nainstalovat, spustit a nakonfigurovat MQTT server. Dále jak si stáhnout Visual Studio Code a nainstalovat PlatformIO. Nakonec jak ESP 8266 správně nastavit a spustit.

## Splněné a nesplněné cíle

Hlavním cílem projektu pro mě bylo více porozumět elektrotechnice, programovacímu jazyku C++ a komunikaci se serverem. K tomu všemu jsem chtěl ještě vytvořit aplikaci pro ovládání, nejspíš v programovacím jazyce Kotlin.

Své cíle jsem z většiny splnil – žaluzie jsem spojil s ESP 8266 a vše funguje, jak jsem předpokládal. Celý kód je napsaný v jazyce C++, tudíž jsem si jej dost procvičil a komunikovat se serverem jsem se také naučil.

Jediný cíl, který jsem nesplnil, byla aplikace. Nečekal jsem totiž, že všechny ostatní cíle budou tolik časově náročné, a tak už na aplikaci nezbyl čas.

Seznam cílů:

* Posunout se ve znalostech elektrotechniky
* Více porozumět jazyku C++
* Naučit se komunikovat se serverem
* Vytvořit aplikaci k ovládání
* Návod k použití (návod nejspíš není snadný pro běžné uživatele)

# Závěr

Cílem projektu bylo vytvořit pro uživatele vhodné rozhraní, ze kterého může žaluzie různě ovládat. Toto rozhraní poskytuje Home Assistant nebo má webová stránka. V Home Assistantu běží MQTT server, kde může uživatel pohyb a stav žaluzií sledovat.

V budoucnu bych chtěl pro uživatele zjednodušit spuštění zařízení. Aby například nemusel psát údaje své Wi-Fi sítě přímo v kódu. Pro uživatele by taky bylo určitě jednodušší, kdyby vůbec nebylo zapotřebí stahovat Visual Studio Code a PlatformIO.

Praktické řešení: https://github.com/pavelkosnar/zaverecny-projekt

Seznam použitýCH INFORMAČNÍCH ZDROJů

[1] Dawid Chyrzynski. ArduinoHA Documentation [online] [cit. 2022-12-30]. <<https://dawidchyrzynski.github.io/arduino-home-assistant>>.

[2] RandomNerdTutorials. ESP 8266 Web Server using *SPIFFS* [online] [cit. 2022-12-30]. <<https://randomnerdtutorials.com/esp8266-web-server-spiffs-nodemcu>>.

[3] RandomNerdTutorials. ESP 8266 NodeMCU Async Web Server [online] [cit. 2022-12-30]. <https://randomnerdtutorials.com/esp8266-nodemcu-async-web-server-espasyncwebserver-library>.

[4] W3Schools. Bootstrap 4 Tutorial [online] [cit. 2022-12-30]. <https://www.w3schools.com/bootstrap4/default.asp>.

[5] The Hook Up. Home Assistant Beginners Guide [online], YouTube. Jan 29, 2020 [cit. 2022-12-30]. <https://www.youtube.com/watch?v=sVqyDtEjudk&t=611s&ab\_channel=TheHookUp>.