Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

Senzorová stanice na bázi Raspberry Pi

Vít Všetečka

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Vít Všetečka

Informatika

Název práce

Senzorová stanice na bázi Raspberry Pi

Název anglicky

Sensor station based od Raspberry Pi

Cíle práce

Hlavním cílem práce je návrh měřícího zařízení využitelného pro monitoring environmentálních podmínek.
Dílčími cíli jsou:
- návrh zpracování získaných dat,
- analýza nástrojů zajišťujících reakci na naměřená data.

Metodika

Metodika řešení diplomové práce je založena na studiu a analýze dostupných odborných informačních zdrojů. Stěžejní pro vypracování diplomové práce bude aplikace získaných poznatků na modelovém monitorovacím zařízení. Dále pak jeho konfigurace pro použití webového rozhraní a implementace zařízení. Na základě získaných poznatků bude formulován závěr práce.

Doporučený rozsah práce

30 – 40 stran

Klíčová slova

Raspberry Pi, IoT, měření, čidlo, monitoring

Doporučené zdroje informací

BLUM, Richard. Linux command line and shell scripting bible, third edition. 3rd edition. Indianapolis, IN: John Wiley and Sons, 2015. ISBN 978-1-118-98384-3

GAJJAR, R. Raspberry Pi Sensors. Birmingham, UK: Packt Publishing Ltd, 2015. ISBN 978-1-78439-361-8

GONER, J. – UPTON, E. – HALFACREE, G. Raspberry Pi : uživatelská příručka. Brno: Computer Press, 2013. ISBN 978-80-251-4116-8.

NORRIS D. Raspberry Pi: Projekty. Computer Press, 2015. ISBN 978-80-251-4346-9

Předběžný termín obhajoby

2017/18 LS – PEF

Vedoucí práce

Ing. Alexandr Vasilenko, Ph.D.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 13. 11. 2017

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 13. 11. 2017

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 15. 03. 2018

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Senzorová stanice na bázi Raspberry Pi" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 15.3.2018 _____

Poděkování

Rád bych touto cestou poděkoval Ing. Alexandru Vasilenkovi, Ph.D., za rady, čas a odborné vedení mé bakalářské práce.

Senzorová stanice na bázi Raspberry Pi

Abstrakt

Tato bakalářská práce se zabývá návrhem měřícího zařízení, sloužícího pro monitoring environmentálních podmínek, dále pak návrhem zpracování získaných dat a analýzou nástrojů zajišťující reakci na naměřená data.

V teoretické části práce se nachází definice pojmu environmentální monitoring a jeho základní členění. Dále se zde nachází charakteristika základních environmentálních veličin. Teoretická část se také zabývá analýzou nástrojů, které zajišťují měření, zpracování a následnou reakci na naměřená data. Posledním bodem teoretické části je podrobný popis jednotlivých komunikačních nástrojů.

Praktická část práce se věnuje samotnému návrhu měřícího zařízení a jeho následné demonstraci na modelovém zařízení. V praktické části se blíže specifikují jednotlivé část návrhu a způsob jakým fungují. Další částí zobrazuje, jakým způsobem se využívá software, který by popsán v teoretické části. Nachází se zde je přesná konfigurace pro tento návrh a konkrétní modelové zařízení. Poslední část práce je podrobná analýza a vysvětlení jednotlivých funkcí programu, který sbírá měřená data a odesílá je k dalšímu zpracování.

Klíčová slova: Raspberry Pi, IoT, měření, čidlo, monitoring

Sensor station based on Raspberry Pi

Abstract

This bachelor thesis deals with the design of the measuring device, which serves for the monitoring of the environmental conditions, the design of the obtained data and the analysis of the instruments providing the response to the measured data.

In the theoretical part of the thesis is the definition of the concept of environmental monitoring and its basic division. There is also a characteristic of basic environmental variables. The theoretical part also deals with the analysis of tools that ensure the measurement, processing and subsequent reaction to the measured data. The last point of the theoretical part is a detailed description of the individual communication tools.

The practical part deals with the design of the measuring device itself and its subsequent demonstration on the model device. The practical part details the individual parts of the design and the way they work. The next section shows how software is used to describe the theoretical part. Here is the exact configuration for this design and specific model device. The last part of the thesis is a detailed analysis and explanation of individual program functions, which collects the measured data and sends it for further processing.

Keywords: Raspberry Pi, IoT, measurement, sensor, monitoring

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Monitoring enviromentálních podmínek může být nejen zajímavý, ale i člověku užitečný. Měření těchto podmínek na více místech, sběr jejich dat a zároveň využití možnosti zajistit reakci na jejich změnu je velmi mladý a moderní přístup k věci, která je oproti tomu velmi stará a částečné zanedbávaná. Stanic pro měření enviromentálních podmínek je velké množství. Ale stanic, které hodnoty měří, sbírají, vyhodnocují a dokáži na ně zajistit určitou reakci takové množství není. Jedná se o takzvaný „Internet věcí". V dnešní době je velký rozmach těchto zařízení, a výstup této bakalářské práce je jedna z možností, která tento pojem přesně vystihuje.

Cílem této práce je návrh měřícího zařízení využitelného pro monitoring environmentálních podmínek, včetně návrhu zpracování získaných dat a analýzy nástrojů zajišťujících reakci na tato naměřená data. Metodikou řešení této práce je studium a analýza dostupných odborných zdrojů, a stěžejním bodem této práce je aplikovat získané poznatky na modelovém zařízení. V neposlední řadě je konfigurace modelového zařízení pro použití webového rozhraní a jeho implementace. Nadstavbou tohoto zařízení je pak využití cloudové služby, která umožňuje kontrolu stanice z jakéhokoliv místa na zemi, kde je dostupné připojení k internetu.

2 Cíl práce a metodika

2.1 Cíl práce

Hlavním cílem práce je návrh měřícího zařízení využívaného pro monitoring environmentálních podmínek.
Dílčími cíli jsou:

- Návrh zpracování získaných dat
- Analýza nástrojů zajištujících reakci na naměřená data

2.2 Metodika práce

Metodika řešení bakalářské práce je založena na studiu a analýze dostupných odborných informačních zdrojů. Stěžejní pro vypracování bakalářské práce bude aplikace získaných poznatků na modelovém monitorovacím zařízení. Dále pak jeho konfigurace pro použití webového rozhraní a implementace zařízení. Na základě získaných poznatků bude formulován závěr práce.

3 Teoretická východiska

Teoretická část práce se věnuje definicím environmentální monitoring a enviromentálním podmínkám, popisu vybraných veličin. Dále se v tomto tématu analyzuje použitý hardware, použitý software, včetně jeho instalace. Dále se jedná o hlubší popis komunikační vrstvy mezi hardwarem a využívanými aplikacemi.

3.1 Environmentální monitoring

Z definice slova monitoring jasně vyplývá, že se jedná o sběr určitých dat, v tomto případě environmentálních, v prostoru a čase, za účelem vyhodnocování a následné porovnávání těchto dat. Monitoring neznamená tedy pouhé sledování určitých dat, ať už v čase, nebo jejich aktuální stavy, ale znamená to i následné zpracování a vyhodnocení získaných dat. Cílem monitoringu je získávání, shromažďování a zpracování dat, které jsou následně navzájem porovnatelné například v čase. Environmentální monitoring lze klasifikovat podle různých kritérií, mezi nejpoužívanější kritéria patří měřítko problému. To se dělí na tři základní skupiny. [1][2]

- **Globální monitoring**

  Globální, je monitoring celosvětový.
- **Regionální monitoring**

  Regionální, je monitoring velkých celků, například jednotlivých států.
- **Impaktní monitoring**

  Impaktní, je monitoring týkající se menších celků, například měst.

3.1.1 Environmentální podmínky

Pojem „environmentální podmínky“ je velmi široký a podle ČSN ISO normy to jsou takové podmínky, jejich stav, nebo charakteristika životního prostředí, jak jsou stanoveny v určitý okamžik. Stejně jako je pojem environmentální podmínky široký, tak stejně široké je i spektrum environmentálních veličiny, které je možně měřit. Environmentální podmínky jsou takové, které ovlivňují venkovní prostředí a ve velkém množství případů jedna veličina ovlivňuje veličiny jiné. Jak již bylo řečeno, environmentálních veličin je k měření velké množství. Některé jdou měřit přímo, jako je například teplota, vlhkost a podobně, a některé se získávají výpočtem jiných veličin. [3]

3.1.1.1 Teplota

Jedná se o základní fyzikální veličinou soustavy SI a její oficiální jednotkou je kelvin (K). Nejrozšířenější jednotkou je potom stupeň Celsia (°C). Pro převod mezi kelvinem a stupni Celsia existuje jednoduchý vzorec. Obě stupnice mají stejný rozdíl teplot 1 °C = 1 K, ale mají různý počátek. [4]

$$0\ K = 273{,}15\ ℃$$

$$0\ ℃ = -273{,}15\ K$$

Teplota může dosáhnout i teoretického minima, takzvané absolutní nuly, kdy se při -273,15 °C zastaví veškerý tepelný pohyb částic. Tomuto jevu se lze však pouze přiblížit, ale nikdy není možno této teploty dosáhnout. Teplota je klíčovou veličinou pro získáváních dalších environmentálních jevů. [5]

3.1.1.2 Vlhkost vzduchu

Vlhkost vzduchu představuje množství vodních par obsažených ve vzdušném prostoru. Vypařování vody a tím tedy její přeměna na vodní páry závisí hlavně na okolní teplotě. Existují dvě základní charakteristiky vlhkosti vzduchu. Existuje takzvaná absolutní vlhkost a relativní vlhkost.

Absolutní vlhkost vzduchu vyjadřuje hmotnost vodní páry v gramech, v jednom metru krychlovém vzduchu, a počítá se následujícím vzorcem, přičemž mH_2O je označení pro hmotnost vodní páry v gramech. V_{net} je označení pro objem směsi vzduchu a vodní páry.

$$AH = \frac{mH_2O}{V_{net}}$$

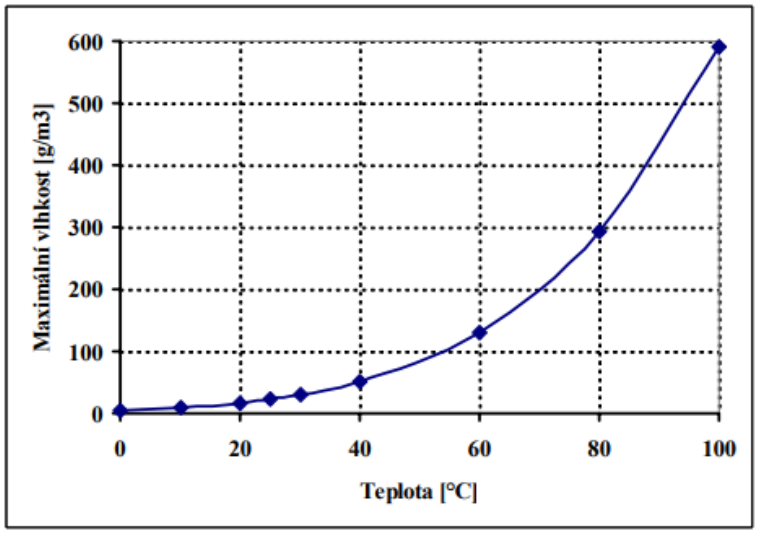
Relativní vlhkost vzduchu udává poměr absolutní vlhkosti a absolutní vlhkosti vzduchu, který by byl při stejné teplotě plně sytý vodními parami. Výpočet relativní vlhkosti se udává v procentech a jeho výpočet je následující. [6]

$$\phi = \frac{\phi_{max}}{\phi} * 100$$

3.1.1.3 Rosný bod

Rosný bod znázorňuje stav, kdy bylo dosaženo maximální nasycení vzduchu vodními parami (100% relativní vlhkosti vzduchu). A znamená to, že do prostředí, ve kterém bylo dosaženo rosného bodu, se již další vodní páry nemohou vypařit a případné nadbytky začnou

kondenzovat ve vodu. V grafu je vidět, jak s rostoucí teplotou roste nasycení vzduchu vodními parami (Obrázek 1). [7]



Obrázek 1 - Vliv teploty na nasycení vzduchu vodními parami

3.1.1.4 Ultrafialové záření

Ultrafialové záření je neviditelné záření, které má kratší vlnovou délku, než má světlo viditelné. Rozlišují se tři základní typy UV záření a jejich přirozeným zdrojem je Slunce. Všechna UV záření jsou před dopadem na Zemský povrch „filtrována“ ionosférou a atmosférou. Tři základní typy se rozlišují hlavně podle vlnové délky.

- **UV-A**

UV-A záření má vlnovou délku od 315 do 400 nanometrů a je označováno jako dlouhovlnné. 99% veškerého UV záření, které dopadne na Zemi je právě dlouhovlnné.

- **UV-B**

UV-B záření má vlnovou délku od 280 do 315 nanometrů a je označováno jako středněvlnné. Toto záření dopadá na Zemi v menším množství, protože je filtrováno ozonovou vrstvou.

- **UV-C**

UV-C záření má vlnovou délku od 100 do 280 nanometrů a je označováno jako krátkovlnné. Toto záření je zdraví nejvíce škodlivé a je kompletně filtrováno ozonovou vrstvou a atmosférou. [8][9]

3.2 Hardware

Hardwarová část se věnuje charakteristice mikropočítače Raspberry Pi 3. generace, jeho hardwarovým specifikům, operačnímu systému Dále mikrokontrolérům, které se používá pro sběr dat. Také popisuje vybrané senzory, které je možné k těmto mikrokontrolérům připojit.

3.2.1 Jednodeskové počítače

Za jednodeskové počítače, známé pod zkratkou SBC (Single Board Computer) se označují takové počítače, které mají oproti klasickým stolním počítačům, či notebookům, všechny komponenty zabudované na jedné základní desce, a nemusí být propojovány žádnými kabely. Největší výhodou těchto zařízení je ta, že při nižších nárocích uživatele, mohou jednodeskové počítače nahradit počítače stolní, a to za přijatelnější cenu.

Jednodeskových počítačů je nepřeberné množství v různých provedeních. Mezi nejznámější řadíme Raspberry Pi, Arduino, Banana Pi, Chip, Intel Edison nebo BeagleBoard. Výrobce však nebývá jediné kritérium při výběru jednodeskového počítače. Variací je velké množství a vždy záleží na tom, co se s daným počítačem zamýšlí. Jednodeskové počítače se mohou pohybovat až v řádech od sta do desítek tisíc korun.

Z mnoha pohledů je brán jako nejpodstatnější kritérium typ a výkon procesoru. Na trhu jsou k dostání jednodeskové počítače s jedno až čtyř jádrovými procesory, při čemž společnost Intel, ve svém jednodeskovém počítači řady Edison, nabízí i dvoujádrový Intel Atom (Silvermont) taktovaný až na 2,6 GHz. Další z podstatných kritérií je velikost zabudované RAM paměti, kdy nejběžnější na trhu jsou počítače s 256 MB až 1 GB. Podstatné pro některé projekty je i množství USB konektorů, ethernet, GPIO piny pro možnost připojení externích zařízení apod. Velmi podstatnou roli při výběru by měla sehrát i uživatelská základna, dostupné literatury. [10]

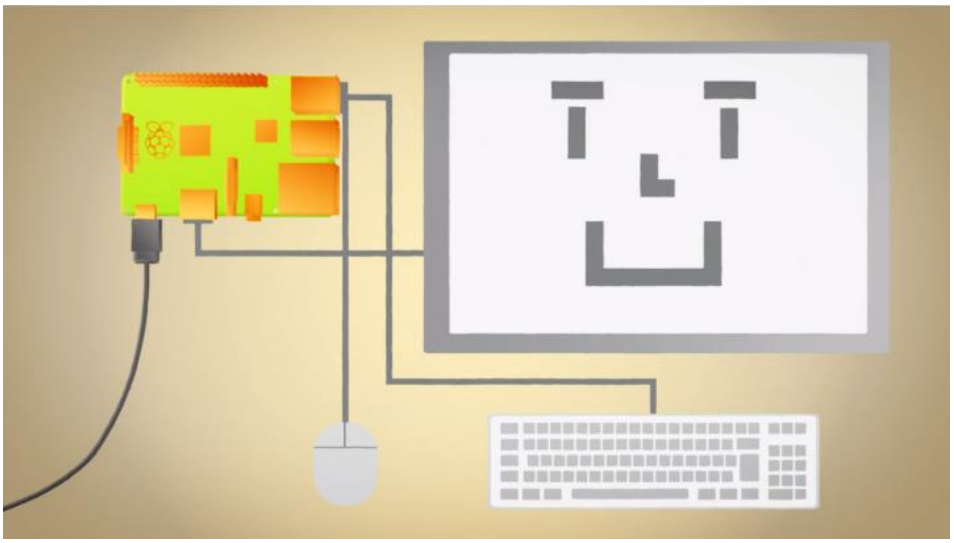
3.2.1.1 Historie jednodeskových počítačů

Historie sahá až do druhé poloviny 70. let, kdy byl společností E&L Instruments představen první jednodeskový počítač s názvem dyna-micro a poháněl ho dnes již legendární procesor Intel 8080A. Dalo by se ale říct, že opravdová historie jednodeskových počítačů, jak je známe dnes, se počítá někdy od roku 2004-2005, kdy se tým odborníků pokoušel vytvořit levný mikropočítač, který by uživatelé bez omezení znalostí a zkušeností mohli používat ve svých projektech. Tak vznikla značka Arduino, která je dnes jednou

z nejrozšířenějších společností prodávající jednodeskové počítače a kontroléry. V roce 2008 společnost BeagleBoard.org přinesla na trh stejnojmenný jednodeskový počítač, který se těšil velké oblibě u všech open source vývojářů. Dalším zlomovým milníkem pro jednodeskové počítače se stal rok 2012, kdy britská společnost Raspberry Pi Foundation vydala svůj první jednodeskový počítač Raspberry Pi model B.

3.2.2 **Raspberry Pi 3 model B**

Jedná se o nejpopulárnější jednodeskový počítač na světě. Raspberry Pi se dělí na čtyři základní modely. Nejedná se ovšem o finální řešení počítačů. Raspberry Pi je totiž možné konfigurovat podle vlastních potřeb. Je možné dokoupit nepřeberné množství modulů, které se dají připojit přímo na Raspberry, jako například až 7 palcový display, WiFi modul, USB periferie a kategorií samou pro sebe jsou takzvané GPIO moduly. Jedná se o moduly připojitelné na piny, které jsou osazené na Raspberry Pi.



Obrázek 2 - Raspberry Pi

3.2.2.1 Technická specifika

Jednodeskové počítače Raspberry Pi se rozlišují na čtyři základní verze. Seřadí-li se chronologicky podle roku výroby, jedná se o Raspberry Pi, které mělo 3 základní modely, a to v podobě modelů B, B+ a A+. Další verzí bylo Raspberry Pi 2, které mělo pouze označení Model B. Neposlední verzí bylo Raspberry Pi 3, které taktéž neslo označení Model B. (Pro)zatím poslední verzí je Raspberry Pi Zero, které má nejen základní model, ale také nový, rozšířený model, který nese označení „W“. Největší výhodou Raspberry Pi 3 oproti

ostatním modelům je skutečnost, že disponuje již vestavěnými WiFi a bluetooth moduly a silnějším procesorem.

Typ	**Raspberry Pi 3**
Model	Model B
Cena v ČR	1040,- Kč
Procesor	ARM Cortex-A53
Počet jader	Čtyř jádrový procesor
Frekvence	1.2 GHz
Grafický čip	Broadcom VideoCore IV @ 250 MHz
RAM	1 GB
Počet USB portů	4
Ethernet	Ano
Video výstup	HDMI 1080p30
Počet GPIO	20
Bluetooth	Ano
Zabudovaná WiFi	Ano

Tabulka 1 - Specifika Raspberry Pi 3

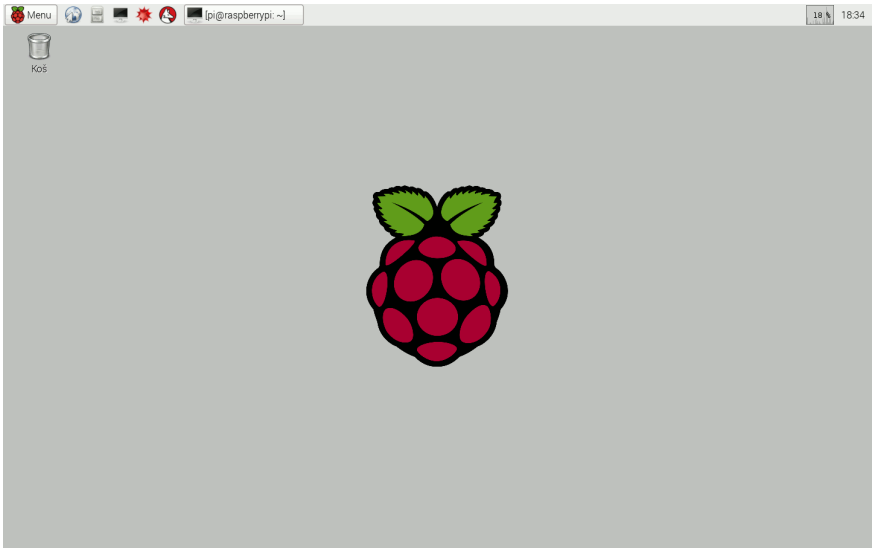
Cena modelu je orientační a je uváděny k datu 25. listopadu 2017, podle jediného oficiálního distributora Raspberry Pi v České Republice, konkrétně podle internetového obchodu rpishop.

3.2.2.1.1 Raspbian

Jedná se o modifikaci Linuxové distribuce Debian, která je upravená přímo pro Raspberry Pi a jeho hardware. Oficiálně se jedná o doporučený operační systém samotnými výrobci Raspberry Pi. Distribuce Raspbianu má dvě základní verze. Dá se rozlišit na takzvaný Raspbian *Stretch with Desktop* a Raspbian *Stretch Lite*. Stretch with Desktop nabízí klasický operační systém včetně grafického rozhraní, zatímco u Stretch with Lite se jedná pouze o příkazovou řádku. V této práci se bude využívat Raspbian Stretch Lite. Nejedná se však o jedinou distribuci Linuxu pro Raspberry Pi. Distribucí je velké množství, jednou z populárních distribucí je i upravená verze Fedory, takzvaný Pidora, která se dá naistalovat pomocí instalačního nástroje NOOBS.

Instalace systému lze provést buď přes instalační nástroj NOOBS, nebo pomocí programu *dd*, který se spouští v bashi a umožňuje nízkoúrovňové kopírování a následnou konverzi dat. Program *dd* využívá dvou základních parametrů *if* pro zadání vstupního souboru a parametr *of* pro výstupní soubor.

Všechny verze Raspbianu lze zdarma stáhnout na oficiálních stránkách Raspberry Pi. Na stránkách se vždy nachází poslední stabilní verze v provedení Desktop i Lite. Verze Raspbianu Stretch with Desktop je vidět na obrázku (Obrázek 4). [11]



Obrázek 3 - Raspbian Stretch with Desktop

3.2.3 **Mikrokontroléry**

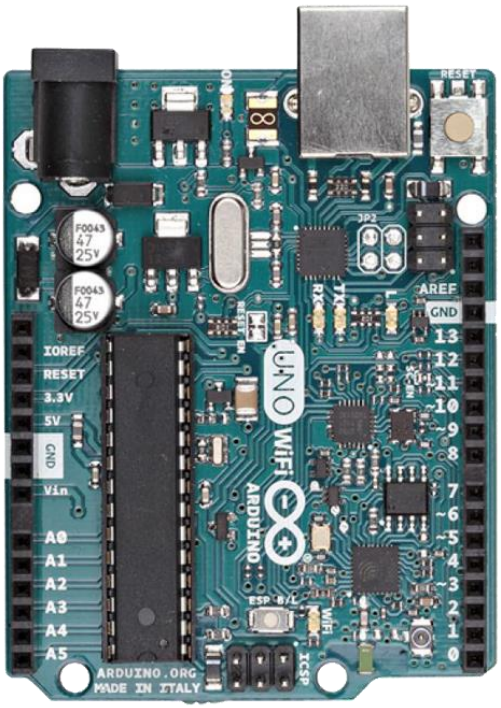
Jedná se o integrovaný obvod, který se dá programovat. Mikrokontroléry standardně disponují procesorem, programovatelnou pamětí, datovou pamětí a GPIO piny. Oproti běžným počítačům, či jednodeskovým počítačům jako je například Raspberry Pi, mikrokontrolér však nedisponuje žádným operačním systémem, a ani nenabízí možnost instalace operačního systému. Na mikrokontrolér se přes takzvaný programátor nahraje program, který se drží v paměti a stále se opakuje. Mikrokontroléry využívají Harvardskou architekturu, aby byla paměť pro data a pro samotný program oddělena.

3.2.3.1 Arduino

Arduino je označení pro open source mikrokontrolér. V informatice se z označení Arduino stalo synonymum pro téměř veškeré, s okolním světem interaktivní mikrokontroléry, využívané převážně pro vývoj a prototypování. Těmto neoficiálním deskám se říká Arduino klony a je jich značné množství.

Originální desky Arduina jsou vždy postaveny na procesoru od firmy Atmel, který je obklopen dalšími elektronickými komponenty. Velkou výhodou originálních desek Arduina, jsou takzvané Shieldy. Jedná se o fyzické rozšíření Arduino desky, které se dá dokoupit. Shieldy mohou od originální desky obsahovat například WiFi modul, síťový konektor a podobně. Shield se poté jen zasune do předpřipravených zdířek na Arduinu. Je ale nutné dávat pozor na to, aby deska Arduino byla kompatibilní s daným Shieldem. Stejně jako samotných desek je velké množství také klonových shieldů.

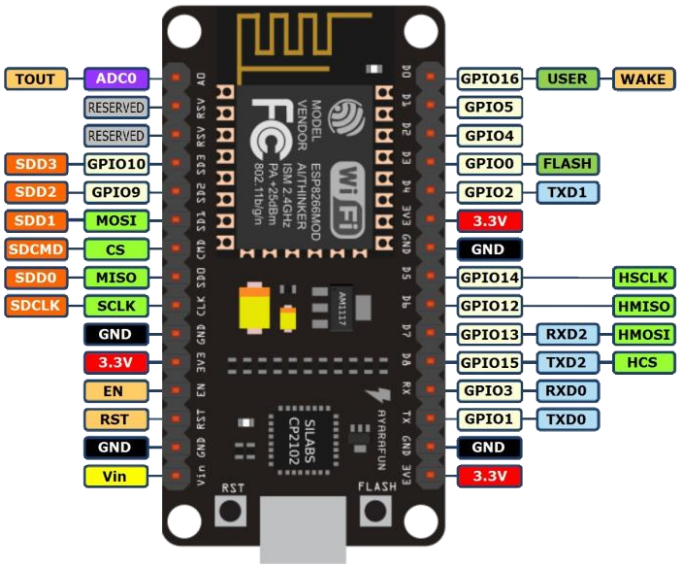
Mateřská společnost Arduino AG se kromě návrhu samotného Arduina věnuje také vývoji softwaru, který je díky možnosti doinstalování knihoven kompatibilní se všemi Arduiny od jiných výrobců a využívá se k programování těchto zařízení. Tento software nese označení Arduino IDE. [11][18]



Obrázek 4 - Arduino Uno WiFi

3.2.3.2 NodeMCU ESP8266 v3

NodeMCU je open source platformou pro čip ESP8266, což je mikrokontrolér s WiFi modulem, fungující na TCP/IP protokolu. To vše je umístěno na hardwaru, který funguje na základě ESP-12 a disponuje dvěma řadami GPIO piny, kdy na každé straně se nachází 15 těchto pinů. Celý node se programuje přes nástroj Arduino IDE. Node disponuje 32 bitovým Xtensa LX procesorem, pamětí o velikosti 1 MB pro zkompilovaný kód a 4 MB flash pamětí. Průměrná cena se pohybuje okolo 200 Kč. NodeMCU je nejčastěji porovnáváno s oficiálním modelem od společnosti Arduino, konkrétně Arduino Uno WiFi (Obrázek 4), které je ovšem několikanásobně větší, ale zároveň má mnohonásobně menší kapacitou pamětí. Rozložení GPIO pinů je vidět na obrázku (Obrázek 5). [13][20]



Obrázek 5 - Diagram GPIO pinů na NodeMCU

3.2.4 **Senzory**

Výběr senzorů vždy záleží na tom, jaké hodnoty se mají měřit a kolik máme mikrokontrolérů nebo jiných zařízení k připojení.

3.2.4.1 Teplotní a vlhkostní senzor DHT22

Jedná se o senzor snímající teplo a vlhkost, který je připojený na jednoduchém mikrokontroleru. Výhodou modelu 22 oproti modelu 11, který se nacházel v původním návrhu je, že model 22 má vyšší přesnost měření. Měření teploty je zde po 0,5 stupních Celsia oproti modelu 11, který měřil pouze celé stupně. DHT22 umožňuje měřit teplotu v rozsahu od -40°C až do 80°C. Měřit vlhkost umožňuje v plném rozsahu, to je od 0 do 100%

Celým výstupem z toho senzoru jsou tři hodnoty. Teplota, vlhkost a pocitová teplota. Celý tento senzor se dá snadno připojit k nodu pomocí tří kabelů, konkrétně 3V3, GND a například D5. Pro nastavení senzoru je třeba naimportovat v Arduino IDE knihovnu DHT. [15]

3.2.4.2 Senzor plynů MQ-135

Jedná se o senzor, který dokáže rozeznat více skupin plynů, které ovlivňují kvalitu ovzduší v jeho okolí. Tento senzor dokáže reagovat na Amoniak (NH_3), oxidy dusíku (NOx), benzeny a oxid uhličitý, který se produkuje při hoření (CO_2). Celý senzor funguje na principu aktivního prvku, který mění svůj odpor v důsledku na koncentraci zmíněných

plynů. V tomto případě je aktivním prvkem Oxid Číničitý (SnO_2). Pro připojení tohoto senzoru jsou potřeba čtyři vodiče. 5V, GND a piny D2 a A2, které se připojují na senzor k pinům D0 a A0. Stejně jako u senzoru DHT22, je potřeba naimportovat knihovnu, která nese název MQ135.[16]

3.2.4.3 Senzor UV-A / UV-B záření ML8511

Jedná se o senzor, který je schopen ve svém okolí měřit intenzitu UV záření. Podle výrobce je senzor nejcitlivější v rozmezí od 280-390 nanometrů, což odpovídá téměř celému pásmu UV-A záření a UV-B záření. Senzor je nutné připojit na napětí 3 volty, jinak dojde k jeho zničení. Pro připojení použijeme čtyři piny. 3V3, GND D0 a D1. [17]

3.3 Software

V kapitole software se nachází charakteristika aplikací, které se využívají v samotném měřícím zařízená. Je zde charakterizováno vývojové prostředí Arduino IDE, aplikace openHAB, která slouží pro webovou prezentaci nasbíraných dat, jeho cloudová nadstavba myopenHAB, databázový systém InfluxDB a aplikace třetí strany IFTTT, která reaguje na naměřené hodnoty.

3.3.1 **Arduino IDE**

Jak už bylo popsáno, jedná se o oficiální nástroj od společnosti Arduino AG. Arduino IDE je naprogramovaný v Javě. Programovat v Arduino IDE lze buď v jazyce C nebo C++, je však doporučené využívat knihovnu Wiring, která je v současnosti již natolik rozšířená, že je o ní možno hovořit jako o samostatném programovacím jazyce. Arduino IDE má možnost doinstalování knihoven, které jsou potřeba pro práci s určitými senzory, Arduiny, nebo například pro využívání komunikačního protokolu MQTT. Základním předpokladem pro používání nástroje Arduino IDE je mít desku, která umožňuje sériovou komunikaci. Převodník, bývá buď napevno připájený na desce, nebo jej má čip v sobě. Další možností je použít převodník externí a USB kabel, kterým desku připojíme k počítači, abychom mohli nahrávat programy do Arduina. [13]

3.3.1.1 Sériová komunikace

Sériová komunikace funguje k odesílání či přijímání dat, však největší využití nachází při ladění kódu. Jedná se o jakousi formu kontroly, protože tímto způsobem je možné

pozorovat výstupy jednotlivých podmínek v textovém poli sériové komunikace. Tato kontrola se nazývá *Serial* monitor a nachází se v panelu IDE, jedná se ikonu s lupou. Další, velmi podstatnou funkcí je využívání sériové komunikace s dalšími zařízeními. Všechny funkce v kódu, které využívají sériovou komunikaci obsahují slovo *Serial*.

```
void setup() {
      Serial.begin(9600);
}
```

Funkce Serial.begin se využívá pro zahájení sériové komunikace. V závorkách se poté nachází parametr rychlosti sériové komunikace, který odpovídá počtu přenosů za sekundu. Při komunikaci s PC se nejčastěji využívá 9600 a celá funkce se volá v část setup.

```
void loop() {
      Serial.print(test);
      Serial.println(test2);
}
```

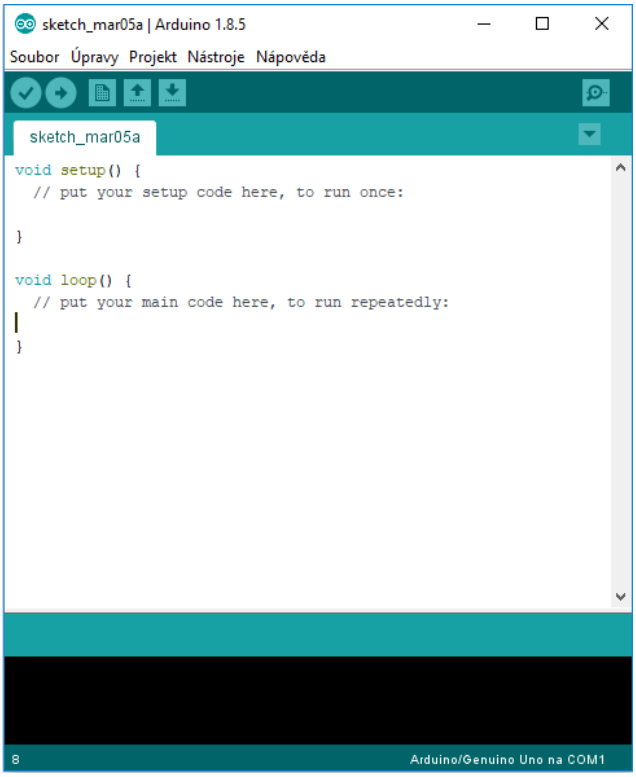
Funkce *Serial.print()* a *Serial.println()* slouží k zasílání hodnot v závorkách do PC, nebo jiného zařízení. Rozdíl mezi nimi je v tom, že funkce *Serial.print()* posílá data stále za sebou, umístěných na jeden řádek. Zatímco funkce *Serial.println()* vždy na konci odeslání dat přidá znak pro zalomení řádku.

3.3.1.2 Základní funkce

V Arduino IDE jsou dvě základní funkce *setup* a *loop*.

```
void setup() {
}
void loop() {
}
```

Do funkce setup se spustí jednou a přejde na další krok, který bývá funkce loop. Funkce loop oproti tomu je taková, která se spouští stále dokola a funguje například pro opakované sbírání dat ze senzorů. Nástroj Arduino IDE lze stáhnout na oficiálních stránkách vývojářů Vývojové prostředí je na obrázku (Obrázek 6). [13]

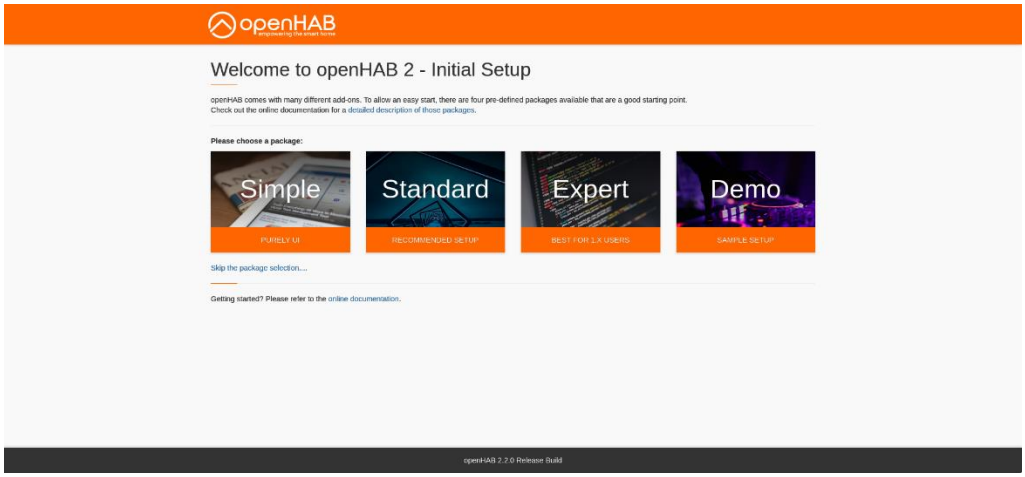


Obrázek 6 - Vývojové prostředí Arduino IDE

3.3.2 **openHAB**

openHAB je open source software, který slouží především k řízení inteligentních domů a domácností. OpenHAB sám o sobě neposkytuje žádné služby pro ovládání jednotlivých IoT, ale díky různým doplňkům je možné jej nakonfigurovat například pro odesílání varovných emailů, pokud je nainstalovaný SNTP server. Zasílání zpráv na různé komunikační kanály, vkládání příspěvků na Twitter a podobně.

OpenHAB je naprogramovaný v Javě, což je jeho výhoda, protože dokáže fungovat všude tam, kde lze Javu nainstalovat, včetně mobilních zařízení. Proto je potřeba nejdříve Javu nainstalovat. OpenHAB nepodporuje OpenJDK, proto je nutné nainstalovat OracleJDK, konkrétně verzi 8. Pro samotnou instalaci programu openHAB je třeba nejdříve stáhnout správné repozitáře a GPG klíč, který zaručuje vyšší spolehlivost zdroje a autenticitu balíčků.



Obrázek 7 - Úvodní stránka openHAB

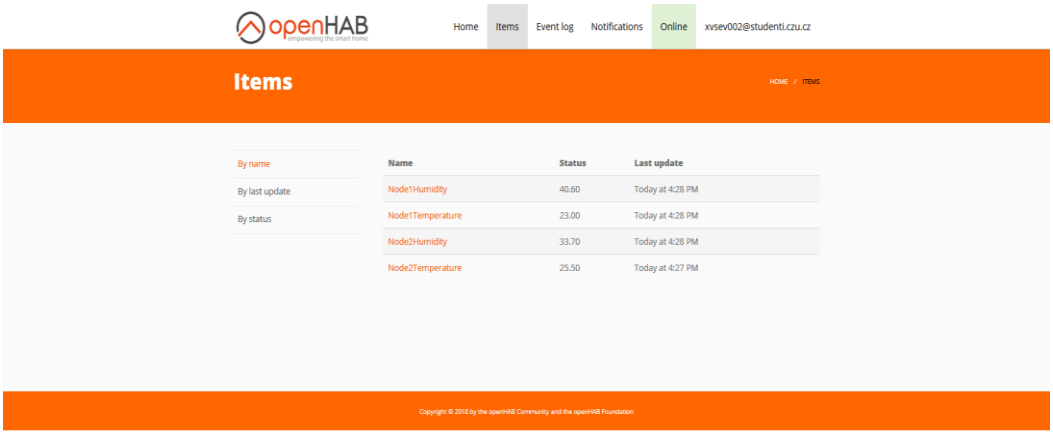
3.3.2.1 myopenHAB

Jedná se o cloudovou službu, která je nabízena zcela zdarma a umožňuje připojení se k openHABu odkudkoliv. Jedná se o nepovinnou nadstavbu aplikace openHAB. Její nastavení zabere pár minut a zpřístupňuje tak aplikaci openHAB odkudkoliv, a to i z jiné sítě, než ve které se nachází. Tato službu umožňuje nahlížení do záznamů událostí, do aktuálních stavu zařízení, která jsou do openHABu připojena a mnoho dalších akcí. Služba myopenHAB se dá propojit se službou IFTTT, která je schopna zajistit reakci na naměřená data. Stránka cloudové služby myopenHAB je vidět na obrázku (Obrázek 7).

Pří registraci je nutné propojit tuto službu s nainstalovaným openHABem a k tomuto kroku je potřeba znát *uuid* a *secret*. Jedná se o unikátní identifikační kód dané instalace a veřejný klíč. Každá instalace openHABu má tyto identifikátory jedinečné.

```
cat /var/lib/openHAB2/uuid && echo
cat /var/lib/openHAB2/openHABcloud/secret && echo
```

Aby tato cloudová služba byla kompatibilní s aplikací openHAB, je nutné mít na Raspberry Pi nainstalovanou Javu 8.



Obrázek 8 - Cloudová služba myopenHAB

3.3.3 InfluxDB

Jedná se o takzvaný „Time Series Database“ open-source databázový systém, vyvíjený společností InfluxData. Označení Time Series znamená, že se zde pracuje s daty, která mají takzvané časové razítko. A proto, se jedná o databázový systém, který je vhodný pro práci s IoT, a senzory obecně. Data do InfluxDB je možné ukládat například přes protokoly HTTP a HTTPS

I zde je před instalací samotné služby nutno stáhnout správné repozitáře a přidat GPG klíč. Influx jako takový nemá vlastní konfigurační soubor, protože celé nastavení probíhá přímo v databázi systému, kde je nejdříve nutné vytvořit uživatele, databázi a přiřadit potřebná práva k operacím.

```
Influx
Connected to http://localhost:8086 version 1.5.0
InfluxDB shell version 1.5.0

>CREATE DATABASE openHAB_db
>CREATE USER admin WITH PASSWORD 'Vseteck4-' WITH ALL PRIVILEGES
>CREATE USER openHAB WITH PASSWORD 'bpvsetecka'
>GRANT ALL ON openHAB_db TO openHAB
>Exit
```

V databázovém systému je také třeba nastavit strategii uchovávání dat. To se provádí v souboru */etc/openahab2/persistence/influxdb.persist*, kde se definuje strategie uchovávání a předměty, které mají tuto strategii využívat. V položce Items se dá použít speciální znak „*“ a značí všechny předměty, které služba openHAB má definované.

```
Strategies {
everyMinute : "0 * * * * ?"
everyHour   : "0 0 * * * ?"
everyDay    : "0 0 0 * * ?"
}
```

```
Items {
*       : strategy = everyUpdate, everyMinute
}
```

3.3.4 **IFTTT**

IFTTT je zkratka názvu „If This Then That“ a jedná se internetovou službu pro propojování a automatizaci zařízení, služeb a různých platforem, které na sebe reagují v podmínkách. IFTTT funguje se třemi základními pojmy, a to úkoly, kanály a applety. IFTTT má velmi snadné ovládání a propojuje se ve velkém množství případů například se službou G-Suite od Google, sociálními sítěmi, nebo s aplikací openHAB. IFTTT umožňuje velké množství akcí, které také podporuje samotný openHAB, jen v jednodušším provedení. Příkladem může být odesílání emailů bez nutnosti instalovat na zprostředkovateli SNTP server. Služba IFTTT se zavede do provozu pouhou registrací a následným nastavením udělení práv ve službě openHAB. Pro aktivování reakcí, je třeba nastavit služby, které k tomu budou potřeba.

3.4 **Komunikace**

Raspberry Pi má několik základních komunikačních kanálů a model, který se využívá ve této práci, stejně jako model Zero W, má v základní verzi ještě o WiFi modul navíc. Mezi základní komunikačním kanály, které se ale nebudou využívat, patří například klasické USB 2.0 porty, Bluetooth a například HDMI. Podstatným komunikačním kanálem, kterým disponuje téměř každé Raspberry Pi jsou GPIO piny a v neposledním řadě pak WiFi modul, kterým v základu disponuje pouze Raspberry Pi 3 model B, nebo Raspberry Pi Zero W.

3.4.1 **GPIO**

Za nejnižší úroveň komunikace se na Raspberry Pi považují takzvané GPIO piny. Jedná se o řadu vstupně/výstupních pinů, které se dají softwarově programovat a umožňují tak základní komunikaci s hardwarem. Raspberry Pi 3 Model B, který se v této práci používá, obsahuje dvě řady po 20 GPIO pinech. Celá Raspberry Pi GPIO logika je dimenzována na 3,3 voltech. Při zapojení 5 voltů by mohlo dojít k nevratnému poškození Raspberry Pi. Na osmi pinech nalezneme uzemnění, které slouží k uzemnění připojeného zařízení. Ve své podstatě je jedno, který uzemňovací pin se použije, jen musí být ve stejné řadě jako připojené zařízení. Nejvyužívanější a největší zastoupení mají klasické GPIO piny, kterých je na desce 17 a slouží pro základní komunikací s připojeným zařízením. Na

Raspberry Pi nalezneme devět dalších pinů, které slouží pro sériovou komunikaci, a to za použití protokolu I^2C, UART a největší zastoupení mají piny podporující SPI zařízení. Mezi piny dále nalezneme dva DNC piny, tyto piny se objevili na Raspberry Pi model B+ a vyšších generacích, a jsou zkratkou pro „Do Not Connect“, a budou vždy neobsazené. Přesné rozložení pinů je na obrázku (Obrázek 9). [22]

3.4.1.1 Sběrnice I^2C

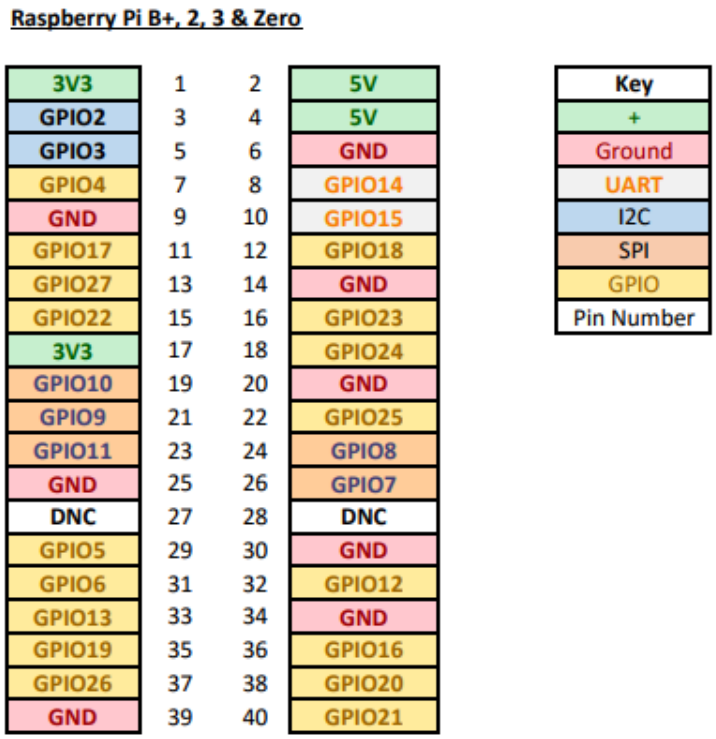
Jedná se o sériovou sběrnici, ke které se lze připojit pomocí pinů 3 a 5 (viz. Obrázek 8), kde pin 3 zajišťuje datový kanál SDA a pin číslo 5 hodinový signál SCL. Sběrnice na těchto pinech je ve skutečnosti pouze jednou I^2C sběrnicí, kterými Raspberry Pi disponuje, ale druhá není uživateli k dispozici. Tato sběrnice má zajistit komunikaci mezi více integrovanými obvody. V případě Raspberry Pi se mezi tyto integrované obvody řadí například procesor Broadcom, který tvoří jádro celého systému.

3.4.1.2 Sběrnice UART

Jedná se o sériovou sběrnici se dvěma vodiči, která je dostupná na pinech číslo 8 a 10 (viz. Obrázek 8), přičemž pin 8 přenáší signál odesílání, takzvaný transmit a pin 10 signál příjem, takzvaný recieve. Pokud se v souboru /boot/cmdline.txt, který se chová jako standartní zavaděč, do něhož lze zapsat téměř jakoukoliv funkci jádra, kterou operační Linux povoluje, nastaví sériový port, používá se poté jako port pro vypisování zpráv právě tato sběrnice. Pokud se připojí přes sériovou sběrnici UART zařízení, které zobrazuje data, je možné zle sledovat zprávy jádra systému. Tento nástroj se využívá jako diagnostický nástroj při problémech s bootováním systému.

3.4.1.3 Sběrnice SPI

Sběrnice SPI má mezi sériovými sběrnicemi největší zastoupení a je k nalezení na pinech 19, 21,23 a piny umožňují výběr čipu24 a 26 (viz. Obrázek 8). Pin 19 poskytuje signál SPI MOSI, pin 21 oproti tomu signál SPI MISO a pin 23 hodinový signál, díky kterému je umožněno synchronizovat komunikaci. Piny číslo 24 a 26 vedou signály výběr čipu, aby bylo možné zvolit jedno z nejvýše dvou nezávislých vedlejších zařízení. SPI je takzvaná synchronní sériová sběrnice, která je určena primárně pro interní systémové programování mikrokontrolérů a jiných zařízení. Oproti sběrnicím I^2C a UART se jedná o sběrnici, která umožňuje komunikovat s více než jedním cílovým zařízením. [11]



Obrázek 9 - Diagram GPIO pinů na Raspberry Pi

3.4.2 **WiFi**

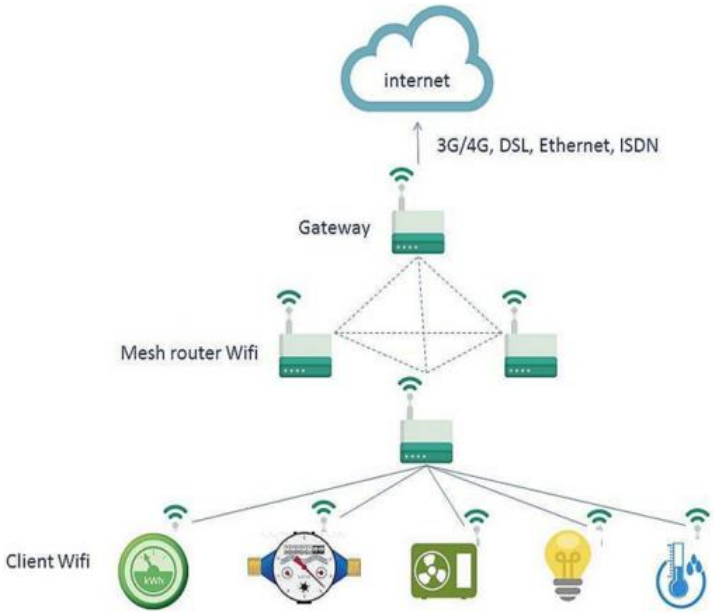
Jedná se o další komunikační vrstvu, která je pro tuto práci zapotřebí. Tento komunikační kanál se využívá při připojování jednotlivých nodů k Raspberry Pi, které sbírají data v jiných oblastech. WiFi zde funguje na standardizovaném ovladači 802.11n, který vysílá na frekvenci 2,4 GHz a disponuje teoretickou, přenosovou rychlostí až 600 Mbit/s s tím, že reálná přenosová rychlost je do 70 Mbit/s.

3.4.2.1 WiFi Access Point

Jednou z největších výhod Raspberry Pi je zabudovaný, nebo dokoupený WiFi modul, díky kterému je možné vytvořit Access Point, na který se budou ostatní nody připojovat a nemusí být připojeny přes GPIO, díky čemuž se dá získat měření z více míst. Celá konfigurace Access Pointu se odehrává v souboru */etc/hostapd/hostpad.conf*.

3.4.2.2 WiFi Mesh Networking

WiFi Mesh Networking je síťovou topologií, do které jsou připojené jednotlivé uzly, ať už aktivní (routery, access pointy, switche), nebo pasivní (počítače, telefony, tablety). Zásadní myšlenkou této topologie je fakt, že oproti klasickým sítím využívající aktivní uzly není nutné propojovat dvě sítě pomocí routingu, ale mesh aktivní prvek si tuto činnost obstará sám a celá síť se potom tváří jako jedna. Největší výhodou Mesh Networkingu je zastupitelnost. Ztratí-li totiž jeden z aktivních prvků signál, nebo se porouchá, nemusí to znamenat odříznutí části sítě, ale aktivní prvky si jednoduše zvolí cestu jinou, jiný aktivní prvek, ke kterému se bude připojovat. Mesh networking je ideální pro používání IoT, které díky tomu mohou být rozmístěny v různých lokacích a vzdálenostech, kam mesh aktivní prvky dosáhnou. Pro připojení k internetu je možné použít i jeden z pasivních prvků. Například mobilní telefon, který bude disponovat předplacenou SIM kartou s připojením k mobilní síti, odkud se budou jednou za pět minut odesílat data MQTT brokeru. Na obrázku (Obrázek 10), je názorné zobrazení WiFi Mesh Networking topologie. [18]



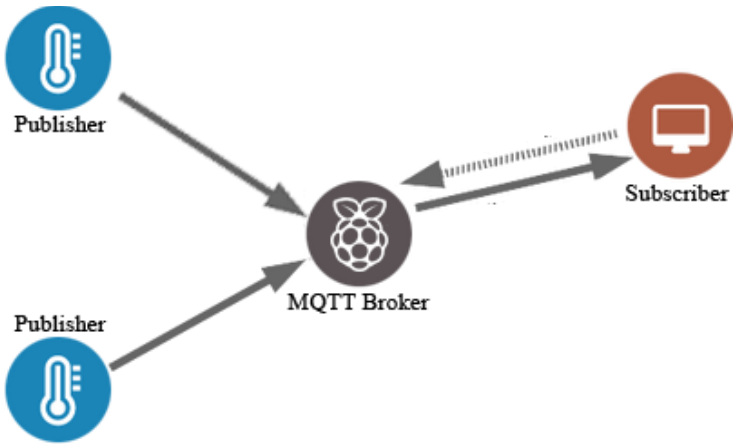
Obrázek 10 - WiFi Mesh Networking topologie

3.4.3 MQTT

Jedná se o jednoduchý protokol, umožňující připojení velkého množství IoT zařízení a následnou komunikaci mezi nimi. V této komunikaci musí vždy existovat jeden zprostředkovatel, takzvaný „Broker“, který se stará o celou komunikaci. MQTT využívá návrhového vzoru Publisher-Subscriber, a tento přenos funguje na protokolu TCP. Výměna

zpráv funguje na bázi témat, nebol-li „Topic“ a zařízení, takzvaný vydavatel, může zprávy buďto publikovat, tím pádem odesílá data brockeru, který je ukládá a distribuuje dál. Nebo je zařízení přihlášeno k odběru tématu, toto zařízení nese označení odběratel, a brocker poté všechny zprávy s daným tématem odesílá právě do tohoto odběratele. Jednotlivá zařízení mohou být přihlášená k odběru více témat. Jednotlivá téma jsou řazena hierarchicky a jsou oddělovány lomítky, například: *Praha/byt/pokoj1/topení*.

Díky této hierarchii lze využívat i speciální znaky, „+“, nebo „#“. V hierarchii poté tyto speciální znaky znamenají vždy něco jiného, přičemž znak „+“ nahrazuje jednu úroveň, což znamená, že při následujícím použití, by zprostředkovatel přijímal všechny zprávy ze všech pokojů a jejich aktuální teplotu u topení *Praha/byt/+/topení*. Speciální znak „#“ nahrazuje jednu či více úrovní hierarchie a musí být vždy poslední. V následujícím použití by to znamenalo, že by zprostředkovatel přijímal zprávy o všech měřených hodnotách v pokoji1 *Praha/byt/pokoj1/#*. Schéma navržení MQTT protokolu je vidět na obrázku (Obrázek 11).



Obrázek 11 - Schéma MQTT

3.4.3.1 Mosquitto

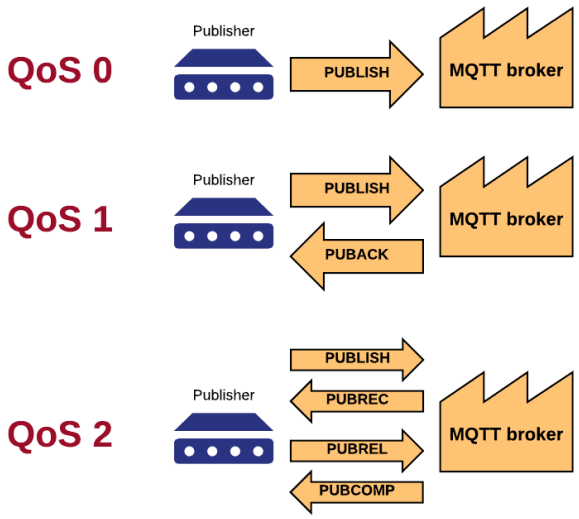
MQTT server, neboli zprostředkovatel, musí mít pro svou správnou funkci nainstalovaný software Mosquitto, který ho automaticky nakonfiguruje jako zprostředkovatele a řídí celou výměnu dat a správu témat. Pro jeho instalaci je zapotřebí nejdříve stáhnout repozitář a GPG klíč, který je následně potřeba přidat.

Základní konfigurace mosquitta probíhá v soboru */etc/mosquitto/mosquitto.conf* a je třeba mu v něm nastavit statickou IP adresu zprostředkovateli.

3.4.3.2 MQTT QoS

MQTT využívá tři úrovně QoS (Quality of Service), a jedná se o potvrzování komunikace mezi vydavatelem a zprostředkovatelem.

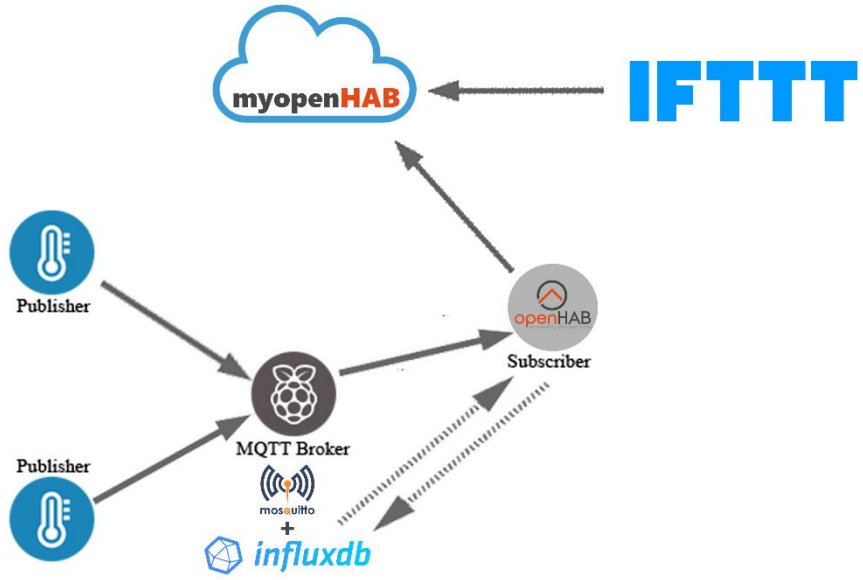
Na úrovni 0, pošle vydavatel zprávu zprostředkovateli a dál se nestará o přijetí zprávy. Zprostředkovatel stejným způsobem odesílá zprávu odběratelům. Na úrovni 1, posílá vydavatel zprávu zprostředkovateli a čeká, až zprostředkovatel zpráv příjme, a odešle ji dále odběrateli. Jakmile odběratel potvrdí přijetí zprávy zprostředkovateli, ten odešle zprávu o tom, že odběratel zprávu obdržel zpět vydavateli a ten, na základě této informace zprávu smaže. Na úrovni 2, posílá vydavatel zprávu zprostředkovateli, ten ji přijme, odešle dál odběratelům, a jako u předchozí úrovně čeká na odpověď. Pokud obdrží zprávu od odběratele o přijetí, pošle vydavateli zpět zprávu o přijetí, vydavatel dá pokyn zprostředkovateli, že je možné zprávu smazat, ten zprávu smaže a odešle vydavateli zprávu o smazání zprávy. Funkce QoS komunikace je vyobrazena na obrázku (Obrázek 12). [23]



Obrázek 12 - QoS komunikace

4 Vlastní práce

Kapitola pojednává o vlastním návrhu měřícího zařízení pro sběr environmentálních podmínek na více místech najednou. Tato kapitola se především věnuje detailnímu popisu jednotlivých procesů výměny dat a také konkrétní konfiguraci softwaru právě pro tuto stanici a modelové zařízení, které demonstruje celý návrh.



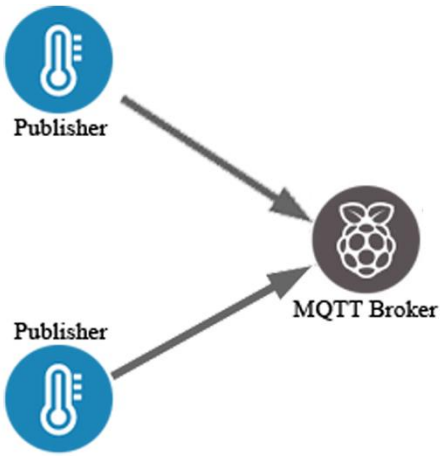
Obrázek 13 - Návrh měřícího zařízení

4.1 Návrh měřícího zařízení

Jedná se o popis komunikace jednotlivých služeb s použitým hardwarem a zároveň popis komunikace, který probíhá mezi službami samotnými.

4.1.1 Odesílání měřených hodnot

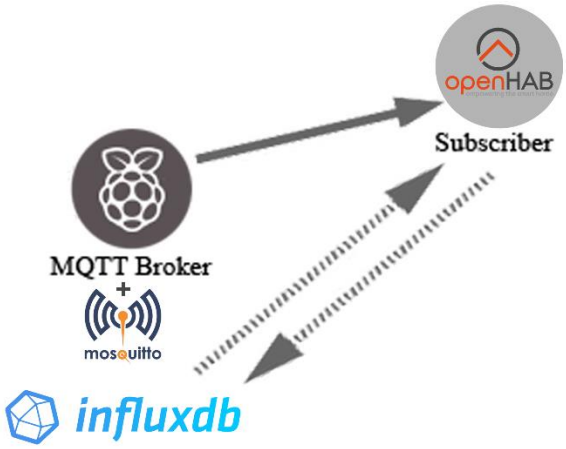
Měření hodnot se probíhá na jednotlivých nodech, ke kterým jsou připojeny senzory na měření environmentálních veličil. Modelové zařízení využívá senzor DHT22 pro měření teploty a relativní vlhkosti. Prvním krokem je načíst hodnoty ze senzoru, o tento proces se starají funkce *dht.getTemperature()* a funkce *dht.getHumidity()* programu, který je nahraný v mikrokontroléru. Dalším krokem je odeslat naměřená data na MQTT server. O tento proces se stará funkce *mqttSend(void)*. Všechny tři funkce jsou dostupné díky doimportovaným knihovnám DHTesp.h do programu Arduino IDE. [12]



Obrázek 14 - Komunikace mezi nody a Raspberry Pi

4.1.2 Zpracování dat

Zpracování dat se provádí na mikropočítači Raspberry Pi, který funguje jako MQTT server. Na něm je nainstalovaný software Mosquitto, Influxdb a openHAB. Raspberry Pi přijatá data převezme, odešle zpět mikrokontrolérům zprávu o přijetí dat a ty následně odešle odběrateli, kterým je služba openHAB. Jak je v aplikaci nastaveno, služba openHAB ukládá všechna data do databáze openhab_db a zároveň z ní sám čerpá, aby mohl vykreslit grafy. Veškerá komunikace spojená se službou openHAB probíhá na přes HTTP port 8080, nebo přes HTTPS port 8443, pokud je tak nadefinovaný.

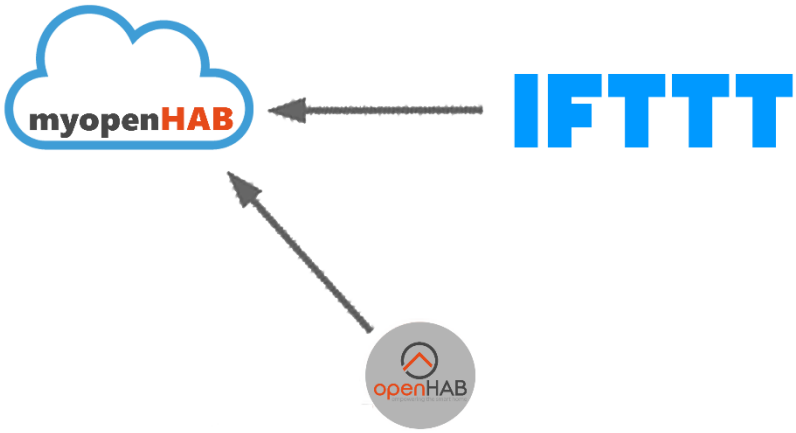


Obrázek 15 - Model zpracování dat

4.1.3 Zobrazení dat a reakce na ně

O vyobrazení dat přes webové rozhraní se stará služba openHAB. Cloudová nadstavba myopenHAB, která je taktéž schopna vyobrazovat data, je propojena s instalací openHABu pomocí uuid a secret klíče. Komunikace mezi těmito dvěma službami probíhá pomocí HTTPS TLS protokolu. Služba třetí strany, jakou je právě IFTTT, které reaguje na změnu

dat ve službě myopenHAB, komunikuje s touto službou pomocí OAuth2 protokolu, který umožňuje přistupovat k HTTPS službám. Služba openHAB odesílá data na cloudovou nadstavbu, kam následně přistupuje IFTTT a kontroluje, zdali některá z hodnot nesplňuje nastavenou podmínku.



Obrázek 16 - Princip zobrazování dat

4.2 Použitý software

Tato část práce se věnuje konkrétní konfiguraci použitého softwaru, který je popsán v rešerši a bude nainstalován na Raspberry Pi.

Dnsmasq

Jedná se o malý dhcp server, který se stará o rozdělování IP adres připojeným zařízením, Jeho konfigurace probíhá v souboru */etc/dnsmasq.conf* a nastavuje se rozsah dhcp server a doba, jakou si server má uchovávat záznamy o zařízeních

```
interface=wlan0
wlan0
     dhcp-range=10.19.95.100,10.19.95.200,255.255.255.0,168h
```

Hostapd

Jedná se o program, který vytváří na Raspberry Pi Access Point a jeho konfigurace probíhá v souboru */etc/hostapd/hostapd.conf*.

`interface=wlan0`	Rozhraní, na kterém má Access point fungovat
`driver=nl80211`	Ovladač, který se má použít
`ssid=rpi`	Název sítě
`hw_mode=g`	WiFi mód

channel=7	Kanál
wmm_enabled=0	Podpora QoS
macaddr_acl=0	Filtrování MAC adres
auth_algs=1	Určuje druh zabezpečení (1=WPA)
ignore_broadcast_ssid=1	SSID WiFi nebude viditelné
wpa=2	Jaký druh WPA se bude používat
wpa_passphrase=bpvsetecka	Heslo k WiFi síti
wpa_key_mgmt=WPA-PSK	Jedná se o doporučené nastavení týkající se zabezpečení sítě
wpa_pairwise=TKIP	
rsn_pairwise=CCMP	

Tabulka 2 - Konfiguračního souboru Access Pointu s vysvětlivkami

openHAB

Jedná se o nástroj, který bude mít na starost interpretaci dat přes webové rozhraní. Pro tyto účely se nejvíce hodí prostředí BasicUI, ale pro potřebnou konfiguraci se používá PaperUI, ve kterém celá konfigurace probíhá

Mosquitto

Služba mosquitto nastavuje Raspberry Pi jako zprostředkovatele a stará se o veškerou výměnu dat, mezi vydavateli, zprostředkovatelem a odběrateli. Základní nastavení bylo již zmíněno, nyní je potřeba celkové nastavení mosquitta. Nastavení probíhá v souboru */etc/openHAB2/service/mqtt.cfg*. V tomto konfiguračním souboru stačí nastavit čtyři základní parametry. Jelikož se již dříve celá síť nastavila jako „neviditelná“, a využívá odlišnou WLAN, do které se budou připojovat pouze měřící zařízení, není nutné zde řešit uživatele a uživatelské heslo.

mosquitto.url=tcp://10.19.95.1:1883	Nastavuje adresu zprostředkovatele včetně portu
mosquitto.qos=1	Quality of Service
mosquitto.retain=true	Nastavuje uchování přijatých zpráv
mosquitto.async=false	Nastavuje asynchronní komunikaci

Tabulka 3 - Konfigurační souboru MQTT s vysvětlivkami

InfluxDB

InfluxDB se stará o persistenci dat, aby bylo možné data zobrazovat historicky v grafech, je třeba držet jejich persistenci. InfluxDB má automatické mazání starých záznamů, aby nedocházelo k zaplňování SD karty.

Konfigurací databáze Influx je myšleno její připojení ke službě openHAB, a to se provádí v souboru */etc/openHAB2/services/influxdb.cfg*. Zde je třeba přidat na konec souboru pět řádků a tím se vyřeší připojení.

`url=http://localhost:8086`	Nastavuje adresu databáze včetně portu
`user=openHAB`	Nastavuje uživatele databáze
`password=bpvsetecka`	Heslo tohoto uživatele
`db=openHAB_db`	Název databáze
`retentionPolicy=autogen`	Zachování politik

Tabulka 4 - Nastavení propojení openHABu s databázovým systémem InfluxDB

IFTTT – Vytvoření appletu

V položce My Applets je možné vytvořit, upravit nebo smazat Applety, je možné k jednomu předmětu vytvořit Appletů více. Applet bude vždy reagovat na změnu, který se projeví v aplikaci openHAB, proto se za položku *+this* vždy vybere openHAB. V položce *+that* bude reakční prvek.

myopenHAB

Pro nastavení služby myopenHAB je nutné, se nejdřív registrovat na webové stránce a ve službě openHAB nastavit předměty, které budou v cloudové nadstavbě zobrazovány.

4.3 Program pro sběr dat

Pro načítání hodnot ze senzorů je nutné nahrát do paměti nodu program, který bude zařizovat načítání hodnot a jejich následné zpracování, ve smyslu komunikace s ostatními komponenty. Psaní celého programu probíhá v aplikaci Arduino IDE. Pro napsání programu je potřeba stáhnout tři základní knihovny. Pomocí #include do programu načteme požadované knihovny.

```
#include "ESP8266WiFi.h"
#include "DHTesp.h"
#include "PubSubClient.h"
```

ESP8266WiFi.h	Umožňuje využívat WiFi modul
DHTesp.h	Umožňuje práci se senzory DHT11/22
PubSubClient.h	Umožňuje využívat MQTT protokolu

Tabulka 5 - Popis instalovaných knihoven

Po načtení knihoven do Arduino IDE je třeba samotnou nodu připojí k WiFi. K tomu je třeba znát a mít nadefinované globální proměnné *ssid* a *password*.

```
const char* ssid = "rpi";
const char* password = "bpvsetecka";
```

Protože celá komunikace probíhá pomocí MQTT protokolu, je nutné jej v programu nastavit.

```
const char* server = "10.19.95.1";
const char* clientName = "node*";
const char* topicTemperature = "node*/temperature";
const char* topicHumidity = "node*/humidity";
```

server	Definuje, na který server (zprostředkovatel) se připojí
clientName	Název zařízení
topicTemperature	Název vytvořeného tématu pro měření teploty
topicHumidity	Název vytvořeného tématu pro měření vlhkosti

Tabulka 6 - Popis jednotlivých proměnných

```
void connectWifi(void)

{
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
     delay(500);
     Serial.println("...connecting...");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println(WiFi.localIP());
}
```

Touto funkcí se nod připojí k WiFi. Výstup z toho se odesílá na sériový port, který je možné sledovat přímo v aplikace Arduino IDE, reálný výstup vypadá tak, že se na sériovém portu zobrazí „WiFi connected“ a IP adresa zařízení.

```
void mqttSend(void) {
  if (client.connected()) {
    client.publish(topic,messageMQTT);
  } else {
    mqttReConnect();
  }
}
```

```
void mqttReConnect(void) {
  while (!client.connected()) {
    if (client.connect(clientName))
      else {
      delay(5000);
    }
  }
}
```

Tyto dvě funkce zajišťují komunikaci pomocí MQTT protokolu. Přesněji řečeno, první funkce se stará o zasílání dat na MQTT server. Pokud je klient připojen k MQTT serveru, odešle na něho data, pokud není, spustí druhá funkce. Druhá funkce se stará o komunikaci z hlediska času, ztratí-li klient spojení s MQTT serverem, počká pět vteřin a pokusí se spojení opět navázat.

```
void setup(void)
{
  Serial.begin(9600);
  connectWifi();
  client.setServer(server, 1883);
  dht.setup(14);
}
```

Funkce *setup*, je taková funkce, která se spustí při startu, proběhne a v kódu se pokračuje dále. Tato proměnná má na starost připojení se k server včetně portu, na kterém komunikuje, a sběr dat z DHT senzoru.

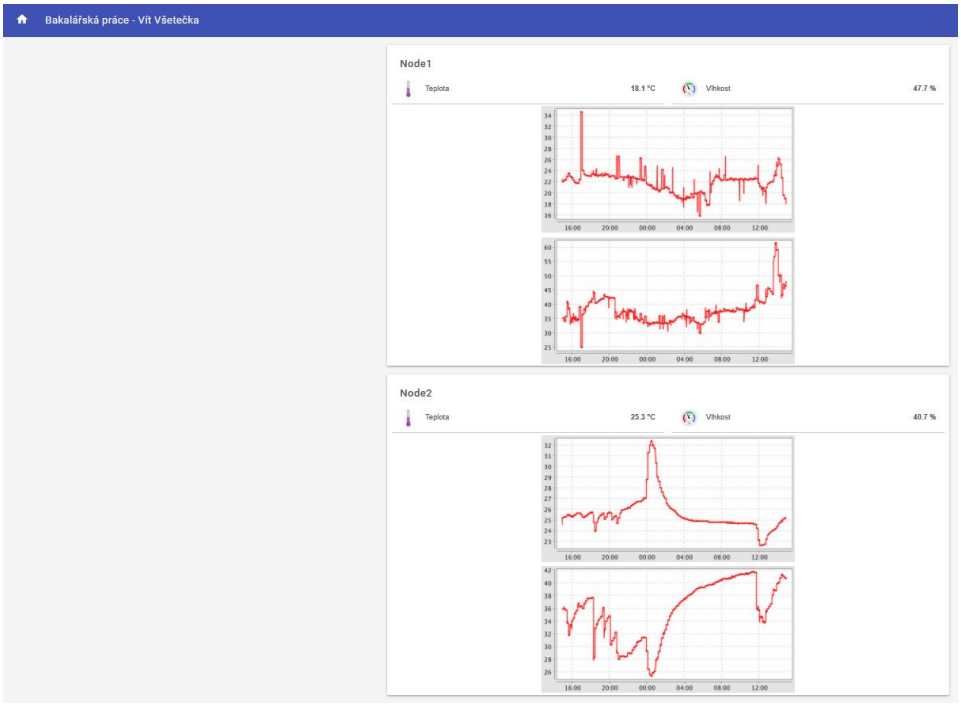
```
void loop() {
    delay(3*dht.getMinimumSamplingPeriod());
  temperature = dht.getTemperature();
  message = String(temperature);
  messageMQTT = message.c_str();
  topic = topicTemperature;  mqttSend();


    delay(3*dht.getMinimumSamplingPeriod());
  humidity = dht.getHumidity();
  message = String(humidity);
  messageMQTT = message.c_str();
  topic = topicHumidity;  mqttSend();
   delay(1000);
  serialDebug();
}
```

Funkce *loop*, se oproti funkci *setup* nespustí pouze jednou, ale automaticky se spouští stále dokola, to zapříčiňuje stéle aktuální naměřená data. *Delay* na začátku zpožďuje minimální dobu sběru dat na senzoru. První část sbírá data o teplotě a odesílá je do tématu pojmenovaného jako *topicTemperature*. To samé se děje ve druhé části kódu, akorát to platí

pro hodnoty vlhkosti. Při nahrávání programu na nod, program vždy projde kompilací. Po této kompilaci je dobré spustit výstup sériového portu a sledovat, zda vše probíhá, jak má.

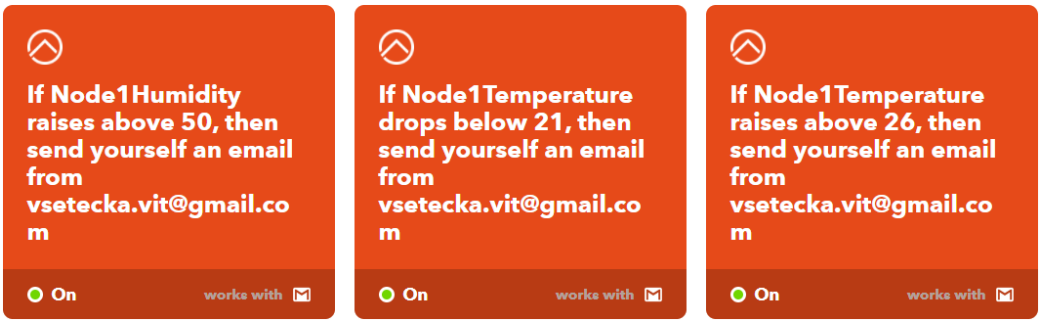
5 Výsledek

Výsledkem této bakalářské práce je plně funkční návrh měřícího zařízení, které slouží pro monitoring environmentálních podmínek. Tento návrh je aplikován na modelovém zařízení, které je schopné získávat data na více místech zároveň. To vše se provádí za pomoci dvou mikrokontrolérů, ke kterým jsou pro demonstraci připojeny senzory DHT22. Tyto senzory sbírají data o teplotě a relativní vlhkosti. Mikrokontroléry jsou naprogramovány na odesílání těchto hodnot za pomoci MQTT protokolu do databáze, která s daty pracuje dále. Díky persistenci těchto dat, kterou tato databáze umožňuje, je možné zobrazit data v podobě grafů. Tyto grafy jsou nastaveny na vyobrazování hodnot v časovém úseku 24 hodin, ale tento úsek jde přenastavit (Obrázek 17).



Obrázek 17 - Výsledný openHAB

Celé modelové zařízení má i vlastní cloudovou službu, ne kterou je možné se připojit odkudkoliv na světě, kde je připojení k internetu. Tato služba se jmenuje myopenHAB a je možné v ní sledovat aktuální stav hodnot, které se promítají díky propojení této služby konkrétní instalací openHABu. Ke cloudové nadstavbě je připojena aplikace takzvaně třetí strany, jedná se o službu IFTTT. Tato služba zajišťuje odeslání varovného emailu, pokud dojde k výrazné změně teploty, či vlhkosti v jedné z destinací, kde se provádí měření.



Obrázek 18 - Nastavené Applety ve službě IFTT

5.1 Celková náklady měřícího zařízení

V této kapitole bude probíhat kalkulace jednotlivých částí, které byly použity na modelovém zařízení. Vyobrazené ceny nemusí být aktuální, ceny se datují k listopadu 2017 a uvádí se včetně DPH. Do ceny se zahrnují pouze položky, které bylo potřeba koupit, není zde započítán například montážní materiál (pájení, spojovací materiály).

Položka	Množství	Poštovné v Kč	Cena za kus v Kč
Raspberry Pi 3 model B	1x	49,-	1040,-
Napájecí zdroj 2,5 A	1x		343,-
Krabička transparentní	1x		169,-
SD karta 16 GB	1x		169,-
Senzor DHT22	2x	-	79,-
Node MCU ESP8266	2x	-	230,-
Síťový kabel CAT5 1m	1x	49,-	29,-
Suma		**98,-**	**2368,-**
Celkem			**2466,-**

Tabulka 7 - Celkové náklady na měřící zařízení

6 Závěr

Hlavním cílem této bakalářské práce byl návrh měřícího zařízení využitelného pro monitoring environmentálních podmínek. Tento cíl byl splněn, včetně demonstrování tohoto návrhu na modelovém zařízení, které je plně funkční a je založeno na jednodeskovém počítači Raspberry Pi 3 model B.

Prvním z dílčích cílů byl návrh zpracování získaných dat. Zpracování dat se nakonec provádí za pomoci aplikací Mosquitto, openHAB a databázového systému InfluxDB, která se stará o persistenci dat. Druhým dílčím cílem byla analýza nástrojů zajišťující reakci na naměřená data. Toto modelové zařízení využívá webové služby IFTTT, která umožňuje velké množství reakčních procesů. Modelové zařízení, které demonstruje celý návrh a využívá reakční proces, v podobě zaslání varovného emailu.

V teoretické části se definoval pojem environmentální monitoring a uvedly se některé ze základních environmentálních podmínek. Dále se tato část práce zabývala hardwarem, který se v práci využívá a detailně se zde analyzovalo Raspberry Pi 3 model B, které se využívá jako centrální bod celého zařízení. Teoretické část také obsahovala popis senzorů, které by bylo možné využít pro sběr dat. Dalším bodem této práce byl popis softwaru, který je využívaný v samotné práci. Posledním, ale podstatným bodem teoretických východisek byl popis nástrojů, které se využívají pro komunikaci.

V praktické části práce se definuje samotný návrh zařízení, včetně schématu, které fungovalo jako předloha pro samotné modelové zařízení. V této části se také řešila celková konfigurace softwarové části a jednotlivých aplikací, které byly použity. Posledním bodem této části byl podrobný popis programu, který zajišťoval sběr dat ze senzorů a jejich následné odesílání dalším částem zařízení.

7 Seznam použitých zdrojů

[1]. HŘEBÍČEK, Jiří; KUBÁSEK, Miroslav. Environmentální informační systémy. Akademické nakladatelství CERM, 2011.

[2]. BAMODU, Oluleke; XIA, Liang; TANG, Llewellyn. An indoor environment monitoring system using low-cost sensor network. Energy Procedia, 2017, 141: 660-666.

[3]. ČSN EN ISO 14004. Informační systém úvadění výrobků na trh. [online]. [cit: 22. 12. 2017]. Dostupné na: https://www.nlfnorm.cz/terminologicky-slovnik/70859

[4]. Thomson W., Mathematical and Physical Papers, vol. I, Cambridge 1882, vol. II, London 1890

[5]. ISO 80000-5:2007, item 5-2a

[6]. Wyer, S.S., "A treatise on producer-gas and gas-producers", (1906) The Engineering and Mining Journal, London, p.23

[7]. ČERMÁK, Jan. SNÍMAČ VLHKOSTI VZDUCHU. 2008.

[8]. RAMEŠ, J. Bencko V. Ultrafialové záření a jeho biologické účinky. Cas Lek Cesk, 1993, 132: 129-33.

[9]. BENCKO, Vladimír, et al. Hygiena : Učební texty k seminářům a praktickým cvičením. 2. přepracované a doplněné vydání vydání. Praha : Karolinum, 2002. 205 s. s. 126

[10]. ROSCH, Winn L. Winn L. Rosch hardware bible. Que Publishing, 2003.

[11]. UPTON, Eben; HALFACREE, Gareth. Raspberry Pi-uživatelská příručka. Computer Press, Albatros Media as, 2017.

[12]. PERUMAL, Venkat Subramanian Arumuga; BASKARAN, Krishnamoorthy; RAI, Suleman Khalid. IMPLEMENTATION OF EFFECTIVE AND LOW-COST BUILDING MONITORING SYSTEM (BMS) USING RASPBERRY PI. Energy Procedia, 2017, 143: 179-185.

[13]. VODA, Zbyšek; KITCHEN, H. W. Průvodce světem Arduina. Martin Stříž, 2015.

[14]. Kumar, Abhijeet, and Apoorva Sharma. "Internet of Life (IOL)." (2015). ISBN 978-93-5156-328-0

[15]. Spark Fun. Digital-output relative humidity & temperature sensor/module. [online]. [cit: 15. 2. 2018]. Dostupné na: https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf

[16]. Olimex - MQ135. MQ135 GAS SENSOR. [online]. [cit: 15. 2. 2018]. Dostupné na: https://www.olimex.com/Products/Components/Sensors/SNS-MQ135/resources/SNS-MQ135.pdf

[17]. Mathematics and Computer Science. ML8511 UV Sensor with Voltage output. [online]. [cit: 15. 2. 2018]. Dostupné na: https://www.mcs.anl.gov/research/projects/waggle/downloads/datasheets/lightsense/ml8511.pdf

[18]. MUHENDRA, Rifki, et al. Development of WiFi Mesh Infrastructure for Internet of Things Applications. Procedia engineering, 2017, 170: 332-337.

[19]. Make us of. Muo Linux Raspbian Jessie gui. [online]. [cit: 12. 13. 2018]. Dostupné na: https://cdn.makeuseof.com/wp-content/uploads/2015/10/muo-linux-raspbian-jessie-gui.png

[20]. Kiwi electronics. Arduino Uno WiFi. [online]. [cit: 13. 3. 2018]. Dostupné na: https://www.kiwi-electronics.nl/image/cache/data/products/arduino/boards/ARD-A000133-1-1000x667.jpg

[21]. IoT bytes. NodeMCU pinout. [online]. [cit: 28. 1. 2018]. Dostupné na: https://pradeepsinghblog.files.wordpress.com/2016/04/nodemcu_pins.png?w=616

[22]. Pi My Life Up. Raspberry Pi pinout. [online]. [cit: 28. 1. 2018]. Dostupné na: https://cdn.pimylifeup.com/wp-content/uploads/2015/09/Raspberry-Pi-GPIO-pinout-diagram.jpg

[23]. Root. Protokol MQTT: Komunikační standart pro IoT. [online]. [cit: 25. 2. 2018]. Dostupné na: https://i.iinfo.cz/images/116/bigclown-2-3.png

8 Přílohy

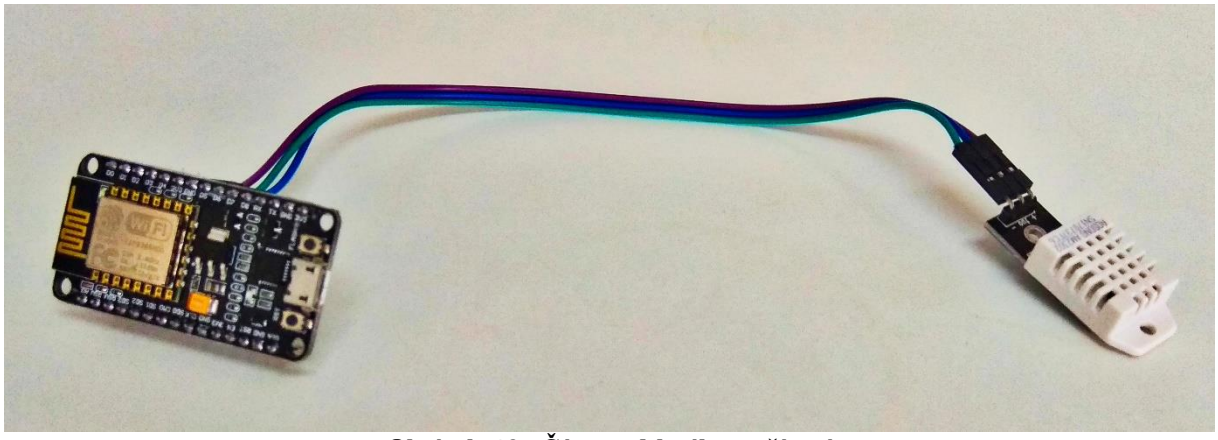
8.1 Kód používaný mikrokontrolérem NodeMCU

```
#include "ESP8266WiFi.h"
#include "DHTesp.h"
#include "PubSubClient.h"

////////////////////////////////////// WiFi parameters
const char* ssid = "rpi";
const char* password = "bpvsetecka";

////////////////////////////////////// MQTT parameters
const char* server = "10.19.95.1";
const char* clientName = "node1";
const char* topicTemperature = "node1/temperature";
const char* topicHumidity = "node1/humidity";

////////////////////////////////////// Global vars
DHTesp dht;
float temperature;
float humidity;
String message;
const char* messageMQTT = "";
const char* topic = "";
WiFiClient wifiClient;
PubSubClient client(wifiClient);

////////////////////////////////////// Custom functions
void connectWifi(void)
{
 WiFi.begin(ssid, password);
 while (WiFi.status() != WL_CONNECTED) {
   delay(500);
   Serial.println("...connecting...");
 }
 Serial.println("");
 Serial.println("WiFi connected");
 Serial.println(WiFi.localIP());
}

void mqttReConnect(void) {
 while (!client.connected()) {
  if (client.connect(clientName)) {
   client.subscribe(topic);
  } else {
   delay(5000);
```

```
    }
  }
}

void mqttSend(void) {
  if (client.connected()) {
    client.publish(topic,messageMQTT);
  } else {
    mqttReConnect();
  }
}

void serialDebug(void){
  Serial.print(dht.getStatusString());
  Serial.print("\t");
  Serial.print(humidity, 1);
  Serial.print("\t\t");
  Serial.print(temperature, 1);
  Serial.print("\t\t");
  Serial.print(dht.computeHeatIndex(temperature, humidity, false), 1);
  Serial.println("");
  delay(dht.getMinimumSamplingPeriod());
  delay(dht.getMinimumSamplingPeriod());
}

////////////////////////////////////// Default functions
void setup(void)
{
  Serial.begin(9600);
  connectWifi();
  client.setServer(server, 1883);
  dht.setup(14);
}
void loop() {

  delay(3*dht.getMinimumSamplingPeriod());
  temperature = dht.getTemperature();
  message = String(temperature);
  messageMQTT = message.c_str();
  topic = topicTemperature;
  mqttSend();

  delay(3*dht.getMinimumSamplingPeriod());
  humidity = dht.getHumidity();
  message = String(humidity);
  messageMQTT = message.c_str();
  topic = topicHumidity;
  mqttSend();

  delay(1000);
```

```
  serialDebug();
}
```

8.2 Mikrokontrolér NodeMCU se senzorem DHT22



Obrázek 19 - Část modelového zařízení