Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

Vzdálené měření teplot pomocí Raspberry Pi s výstupem do webového rozhraní

Petr Kruntorád

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Petr Kruntorád

Informatika

Název práce

Vzdálené měření teplot pomocí Raspberry Pi s výstupem do webového rozhraní

Název anglicky

Remote Measurement of Temperatures with Raspberry Pi with Output to the Web Administration

---

Cíle práce

Cílem práce je vytvořit webové rozhraní určené pro sběr teplot ze zařízení sloužícího k vzdálenému měření za využití Raspberry Pi.

Cílem teoretické části práce je vysvětlení základních pojmů týkajících se tvorby webového rozhraní, principu měření teplot a sběru dat. Mezi sbíraná data patří naměřené teploty z jednotlivých senzorů nebo MAC adresa sloužící pro identifikaci zařízení a jejich čidel. Dále bude provedena analýza již existujících řešení a technologií umožňujících vzdálené měření teplot založených jak na konceptu jednočipových počítačů, tak na konceptu využívajícím již speciálně předvytvořených zařízení.

Cílem praktické části práce je vytvoření webového rozhraní sloužícího pro výpis dat přijatých z Raspberry Pi, skriptu potřebného pro sběr dat ze senzorů a kompletace Raspberry Pi se senzory sloužícími pro měření teplot. Webové rozhraní bude postaveno na PHP frameworku Symfony a jeho účelem bude zobrazení naměřených teplot na jednom místě v uživatelsky přívětivé formě. Skript určený pro sběr dat bude vytvořen v jazyce Python. Kromě měření teplot a jejich odesílání na server provede server automaticky prvotní registraci zařízení do systému.

Metodika

Metodika řešené problematiky bakalářské práce je založena na studiu a analýze odborných informačních zdrojů týkajících se webového rozhraní, měření teplot a konfigurace Raspberry Pi. Dále bude provedena analýza stávajících řešení a technologií umožňujících měření teplot.

Na základě tohoto teoretického základu bude vytvořen vlastní návrh řešení webového rozhraní sloužícího pro výpis naměřených hodnot ze zařízení Raspberry Pi za využití senzorů určených pro měření teplot.

Doporučený rozsah práce

30-60 stran

Klíčová slova

měření teplot, webová aplikace, Raspberry Pi, Python, PHP framework, PHP

Doporučené zdroje informací

HONEK, Lukáš. HTML: Kapesní přehled. Brno: Computer Press, 2004. ISBN 80-722-6958-5.
SUMMERFIELD, Mark. Programming in Python 3: A Complete Introduction to the Python Language. 2. vydání. New Jersey: Addison-Wesley Professional, 2010. ISBN 978-0-321-68056-3.
UPTON, Eben, Gareth HALFACREE a Jakub GONER. Raspberry Pi: uživatelská příručka. 2. aktualizované vydání. Brno: Computer Press, 2016. ISBN 978-80-251-4819-8.
VRÁNA, Jakub. 1001 tipů a triků pro PHP. Brno: Computer Press, 2010. ISBN 978-80-251-2940-1.

Předběžný termín obhajoby

2021/22 LS – PEF

Vedoucí práce

Ing. Dana Vynikarová, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 1. 11. 2021

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 23. 11. 2021

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 08. 03. 2022

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Vzdálené měření teplot pomocí Raspberry Pi s výstupem do webového rozhraní" 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 15.03.2022 _____

Poděkování

Rád bych touto cestou poděkoval Ing. Daně Vynikarové, Ph.D. za odborné připomínky a rady, které byly cenné pro tvorbu této bakalářské práce.

Vzdálené měření teplot pomocí Raspberry Pi s výstupem do webového rozhraní

Abstrakt

Tato závěrečná práce se zabývá vzdáleným měřením teplot prostřednictvím počítače Raspberry Pi a zobrazením naměřených hodnot za využití webové aplikace v grafické formě. V teoretické části jsou vysvětleny základní pojmy, použité technologie, analyzovány existující řešení, specifikovány funkční a nefunkční požadavky. V analýze existujících řešení jsou zahrnuta i alternativní řešení, která lze také využít pro měření teplot, ale jejich sestavení nebo zapojení vyžaduje speciální dovednosti nebo vědomosti.

Praktická část je zaměřena na webovou administraci, která byla vytvořena za pomocí PHP frameworku Symfony, Python skriptů potřebných pro získávání teplot ze senzorů a připojení těchto senzorů k zařízení. Senzory využité v této práci se nazývají DS18B20 a jsou schopné měřit v rozsahu od -55 do +125 °C. Webová administrace obsahuje stránky pro přehled s počty o množství zařízení, uživatelů a oznámení pro aktuálního uživatele, správu oznámení, přehled zařízení, uživatelů a údajů aktuálního uživatele. Stránky určené pro výpis oznámení obsahují informace o mimořádných situací u zařízení (naměřená hodnota překročila limit, zařízení není aktivní a další).

U zařízení i webové administrace je vysvětlen způsob instalace softwarů a skriptů potřebných pro měření teplot na zařízení. Postup instalace specifikuje potřebný software a způsob, jak nainstalovat webovou administraci pro její využití.

Klíčová slova: měření teplot, webová aplikace, Raspberry Pi, Python, PHP framework, PHP

Remote Measurement of Temperatures with Raspberry Pi with Output to the Web Administration

Abstract

This bachelor thesis is focused on remote measurement of temperatures with the usage of the computer Raspberry Pi and displaying the measured values using a web application in a graphical form. The theoretical part explains the basic concepts, the technologies used, analyzes existing solutions, and specifies functional and non-functional requirements. In the analysis of existing solutions are included alternative solutions, which can be also used for temperature measurements, but require special skills or knowledge to set up or connect hardware together.

The practical part focuses on the web-based administration, which was created using the Symfony PHP framework, the Python scripts needed to obtain temperatures from the sensors and connect these sensors to the device. The sensors used in the thesis are called DS18B20 and are capable of measuring from -55 to +125 °C. The web administration includes pages to manage the notification, an overview of the number of devices, users, and notifications for the current user, an overview of devices, users, and profile details of the current user. The pages dedicated to listing notifications contain information on extraordinary situations for devices (measured value exceeded limit, device not active, etc.).

The installation of the software and scripts required for temperature measurements on the device are explained for both the device and the web administration. The installation procedure specifies the necessary software and how to install the web administration to use it.

Keywords: temperatures measurement, web app, Raspberry Pi, Python, PHP framework, PHP

Obsah

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

GPIO – General Purpose Input/Output
UART – Universal Asynchronous Receiver/Transmitter
I^2C – Inter-Integrated Circuit
SPI – Serial Perpheral Interface
ORM – Object-Relational Mapping
MOSI – Master Output, Slave Input
MISO – Master Input, Slave Output
SLCK – Serial Clock
SQL – Structured Query Language
API – Application Programming Interface
WiFi – Wireless Fidelity
PoE – Power over Ethernet
DQL – Doctrine Query Language
SŘBD – Systém řízení báze dat

Úvod

V současnosti jsou počítače stále větší součásti každodenního života a umožňují zautomatizovat činnosti, které by jinak uživatel musel vykonávat ručně. Stále více se začíná využívat takzvaný Internet věcí, pod který patří veškerá zařízení vybavená elektronikou s jejíž pomocí mohou mezi sebou komunikovat. Mezi tato zařízení mohou patřit moderní pračky, chytré osvětlení, televizory a další. Díky tomuto lze například kontrolovat správné vyhřátí domácnosti, bazénu nebo i nesprávnou teplotu zařízení jako je kotel. Toto, ale nejsou jediná možná využití pro zařízení spadající do internetu věcí. V praxi je možné uplatnění tam, kde je potřeba zautomatizovat nějakou činnost.

Toto téma bylo zvoleno, protože již existující zařízení byla mnohdy nevyhovující buď funkcemi nebo cenou. Z toho důvodu se jeví jednočipový počítač Raspberry Pi jako vhodná náhrada pro jeho rozšiřitelnost za využití různých hardwarových nebo softwarových modulů, cenu a jednoduchost. Další zásadní nevýhodou je, že k některým již existujícím řešením se uživatel pro získání dat musí připojit přímo, a proto bude pro měření teplot vyvinuto webové rozhraní.

Webové rozhraní bude sloužit pro správu veškerých zařízeních do ní napojených. Pod správu veškerých zařízení patří jejich prvotní registrace a následné schválení zápisu dat do systému, příjem, zpracování a následný výpis přijatých teplot z jednotlivých senzorů přidružených ke každému Raspberry Pi.

Práce je členěna na kapitoly Cíl práce a metodika, Teoretická Východiska, Vlastní práce, Seznam použitých zdrojů a Přílohy. V kapitole Teoretická východiska se nachází vymezení pojmů, použité technologie potřebné splnění cílů práce, analýza existujících řešení, funkční a nefunkční požadavky. Kapitola vlastní práce popisuje hardwarovou a softwarovou část spolu s ukázkou dílčích částí kódu, kompletace zařízení, instalaci skriptů na zařízení a webové administrace.

1 Cíl práce a metodika

1.1 Cíl práce

Cílem práce je vytvořit webové rozhraní určené pro sběr teplot ze zařízení sloužícího k vzdálenému měření za využití Raspberry Pi.

Cílem teoretické části práce je vysvětlení základních pojmů týkajících se tvorby webového rozhraní, principu měření teplot a sběru dat. Mezi sbíraná data patří naměřené teploty z jednotlivých senzorů nebo MAC adresa sloužící pro identifikaci zařízení a jejich čidel. Dále bude provedena analýza již existujících řešení a technologií umožňujících vzdálené měření teplot založených jak na konceptu jednočipových počítačů, tak na konceptu využívajícím již speciálně předvytvořených zařízení.

Cílem praktické části práce je vytvoření webového rozhraní sloužícího pro výpis dat přijatých z Raspberry Pi, skriptu potřebného pro sběr dat ze senzorů a kompletace Raspberry Pi se senzory sloužícími pro měření teplot. Webové rozhraní bude postaveno na PHP frameworku Symfony a jeho účelem bude zobrazení naměřených teplot na jednom místě v uživatelsky přívětivé formě. Skript určený pro sběr dat bude vytvořen v jazyce Python. Kromě měření teplot a jejich odesílání na server provede server automaticky prvotní registraci zařízení do systému.

1.2 Metodika

Metodika řešené problematiky bakalářské práce je založena na studiu a analýze odborných informačních zdrojů týkajících se webového rozhraní, měření teplot a konfigurace Raspberry Pi. Dále bude provedena analýza stávajících řešení a technologií umožňujících měření teplot.

Na základě tohoto teoretického základu bude vytvořen vlastní návrh řešení webového rozhraní sloužícího pro výpis naměřených hodnot ze zařízení Raspberry Pi za využití senzorů určených pro měření teplot.

2 Teoretická východiska

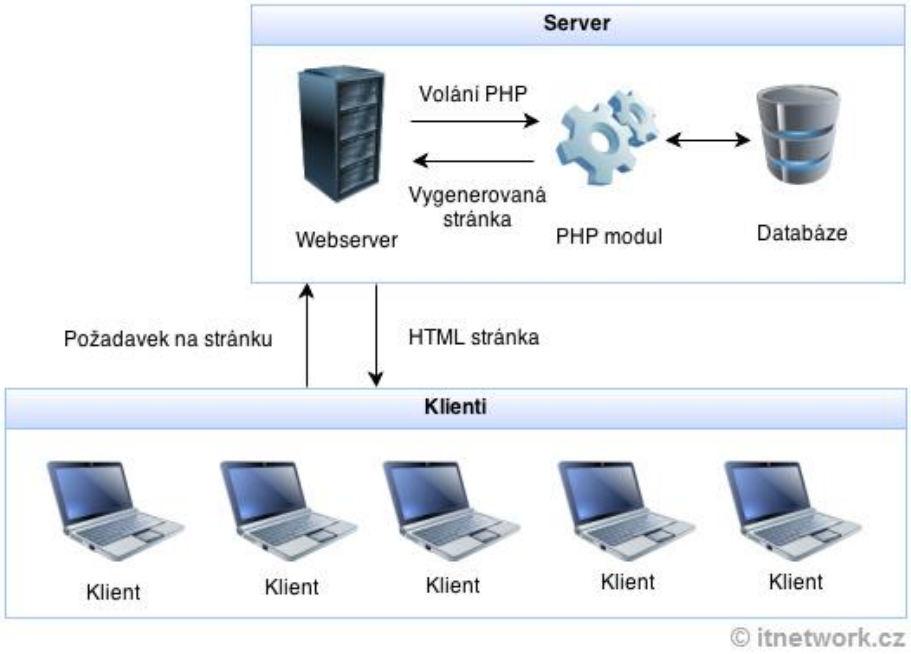
Kapitola teoretická východiska se zabývá vymezením pojmů důležitých pro splnění cílů práce, vysvětlením použitých technologií, analýzou existujících řešení a vytyčením funkčních a nefunkčních požadavků.

2.1 Vymezení pojmů

2.1.1 Webová aplikace

Webová aplikace je forma softwaru, která je na rozdíl od desktopových programů spuštěna na webovém serveru. Jedinou podmínkou pro využití webové aplikace je, aby měl uživatel nainstalován webový prohlížeč. Díky tomuto řešení, není tedy potřeba vyvíjet aplikaci pro všechny požadované platformy jako jsou Windows, Mac OS X, Android, iOS a další. Stačí vytvořit jednu aplikaci, která bude přístupná pro všechny platformy s kompatibilním webovým prohlížečem. [1]

S tímto řešením se však pojí i problémy spojené s tím, že webová aplikace nemusí být přístupná vždy a za jakýchkoliv situací. V případě, že uživatel nemá přístup k internetovému připojení nebo je toto připojení pomalé, ztrácí částečně nebo zcela možnost využití takové aplikace naplno. [2]



Obrázek 1 Webová aplikace – generování obsahu [3]

Oproti statickému webu je zásadní rozdíl v tom, že webový server neobsahuje již finální stránku, ale zpracuje klientské požadavky, vygeneruje data a případně načte data z databáze. Z toho všeho pak server složí hotovou stránku a vrátí ji klientovi do prohlížeče. [3]

2.1.2 Měření teplot

Měření teplot neboli termometrie slouží k objektivnímu zkoumání tepelného stavu látky. Teplota se vyjadřuje za využití několika základních teplotních stupnic a to Termodynamické (také Kelvinova), Celsiovi, Fahrenheitovi, Réamurovi a Rankineho. Z těchto stupnic se však nejčastěji používá stupnice Celsiova, Fahrenheitova a Termodynamická. Pro měření teplot je využíván takzvaný teploměr. Pro měření lze využít hned několik typů teploměrů, kdy každý typ využívá rozdílných principů a může být vhodný pro rozdílné využití nebo situaci. [4], [5]

2.1.2.1 Typy teploměrů

2.1.2.1.1 Kapalinové

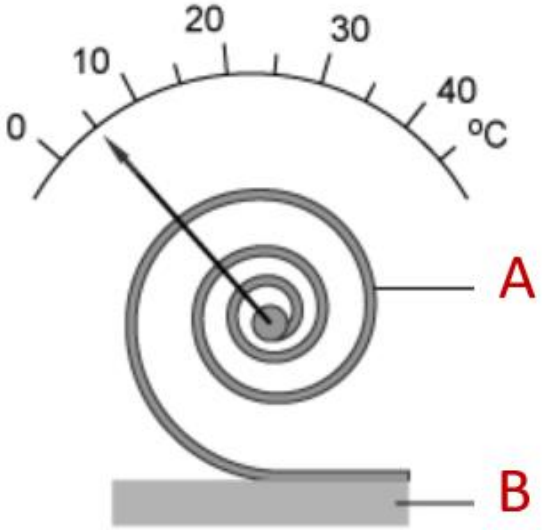
Tento typ teploměru je nejběžnější, se kterým se lze setkat. Nejčastěji se v tomto typu teploměru využívá rtuť, ale existují i teploměry využívající jinou kapalinu (například líh). Jsou složeny ze dvou hlavních částí, a to ze rtuťového rezervoáru s kapilárou a stupnice. [6]

Kapalinové teploměry se dále dělí na dva podtypy. Prvním podtypem je maximální teploměr, který je využíván k zaznamenání nejvyšší teploty a po dokončení měření je potřeba rtuť sklepat nazpátek do rezervoáru. Druhým podtypem je rychloběžka, která měří okamžitou teplotu. Citlivost obou těchto podtypů záleží na objemu rezervoáru a poloměru kapiláry, ale u obou platí přesnost na desetiny stupně. [6]

Obrázek 2 Rtuťové teploměry [6]

2.1.2.1.2 Bimetalové

Bimetalový teploměr funguje na principu rozdílných teplotních koeficientů kovů a teplot. Tento koeficient poukazuje na vztah roztažnosti kovu a teploty, která je pro danou změnu potřeba. Z toho lze tedy vyvodit, že v závislosti na tepelném koeficientu se daný kov rozpíná a smršťuje. [7]



Obrázek 3 Bimetalový teploměr [7]

V teploměru se nachází proužek složený ze 2 rozdílných kovů. Tento proužek je stočený do spirály, která se v závislosti na teplotách rozpíná nebo smršťuje. To způsobuje pohyb ukazatele, který následně ukazuje naměřené hodnoty. [7]

2.1.2.1.3 Odporové

Odporové teploměry pracují na principu změn elektrického odporu jednotlivých kovů a polokovů. Pro vyjádření závislosti lze použít vzorec $R_t = R_0 \cdot (1 + \alpha \cdot \Delta t)$. V tomto vzorci R0 zastupuje odpor při teplotě rovné nule, α je teplotní součinitel a Δt rozdíl teplot. [4]

Nejčastěji jsou vyrobeny z platiny, ale k jejich výrobě lze použít i jiné materiály, mezi které patří měď a nikl. V závislosti na zvoleném materiálu se odvíjí rozsah měření, stabilita odporu a přesnost. [4], [8]

2.1.2.1.4 Termoelektrické

Termoelektrické teploměry neboli termočlánky využívají termoelektrického jevu. Termoelektrický jev využívá principu, při kterém existují v uzavřeném elektrickém obvodu dva

vodiče vyrobené z různých kovů a každý z nich má rozdílenou teplotu. V případě jejich rozpojení lze měřit teplotní rozdíl mezi těmito spoji a definovat hodnoty termonapětí. [4]

Pro měření teplot termočlánkem se jeden vodič umístí do prostředí, kde je referenční teplota a druhý vodič se umístí do místa určeného k měření teploty. Poté je změřeno termonapětí a za pomoci zařízení nazývaného Voltmetr lze určit teplotu na setiny stupně Celsia. [4]

2.1.2.1.5 Radiační

Radiační nebo také bezdotykové teploměry fungují na bázi měření vyzařovaného tepelného záření, které měřený objekt vyzařuje. Vlastnost každého objektu vyzařovat tepelné záření je definováno Stefan-Boltzmannovým zákonem, který platí pro černá tělesa, a proto je potřeba pro reálná tělesa zavést veličinu známou jako emisivita. Pro taková tělesa je nezbytné zohlednit jejich lesklost nebo průhlednost materiálu, které může tento způsob měření významně zkreslit. Pro snížení tohoto zkreslení, lze měřený povrch pokrýt materiálem, který je již tento typ teploměru schopný změřit (například samolepky u nižších teplot nebo vhodný nátěr u teplot vyšších). [6], [9]

Tento typ teploměru se skládá z optické soustavy (čočky, optická vlákna a spektrální filtry), zaměřovače (světelný nebo laserový), snímače, elektrických obvodů a displeje. Optická soustava se stará o to, aby paprsky odražené od měřeného objektu dopadly na snímač, ze kterého jsou následně získány naměřené hodnoty, které se poté zobrazí na displeji. [6], [9]

2.1.3 Framework

Framework je softwarové řešení sloužící pro zjednodušení a urychlení vývoje aplikací. Jeho výhoda spočívá zejména v sadě implementovaných knihoven, které obsahují řešení na nejběžnější úkony, které může programátor při tvorbě aplikace vyžadovat. Mezi tyto úkony se řadí různé typy validace uživatelských vstupů, propojení a práce s databází nebo velké množství jiných problémů, které byly již řešeny v minulosti. Programátor tedy nemusí dané řešení tvořit od začátku a pro požadovanou funkcionalitu stačí využít některou z již předpřipravených knihoven nebo komponent vhodných pro zvolený problém. [10]

2.1.4 Databáze

Databáze je roztříděná sbírka dat, tak aby se s nimi dalo efektivně pracovat (načítání, vyhledávání a další operace). Nejčastěji se pro efektivní roztřízení dat využívají tabulky, ve

kterých jsou data následně rozdělena do sloupců a řádků. O správu databáze se stará SŘBD. [11]

2.1.4.1 Databázové modely

2.1.4.1.1 Hierarchická

V Hierarchické databázi jsou data uspořádána ve stromové struktuře, každý prvek odpovídá jednomu uzlu. Jednotlivé uzly mají mezi sebou vztahy typu nadřazený (rodič) nebo podřazený (potomek). Každý uzel může mít více podřazených (vazba 1:N). Vztahy mezi takovými uzly jsou jednosměrné (od nejvyššího uzlu nebo otce po nejnižší). [12]

2.1.4.1.2 Síťová

Základní princip síťové databáze vychází z hierarchické, ale vztahy a vazby mezi jednotlivými prvky jsou pozměněny. Na rozdíl od hierarchické databáze mohou mít prvky mezi sebou libovolný počet obousměrných vztahů (vazba M:N). Jeden prvek tedy může mít větší množství nadřazených prvku (rodičů) a zároveň může být nadřazen sám sobě. [12]

2.1.4.1.3 Objektově-orientovaná

V objektově-orientované databázi jsou data ukládány ve formě objektů. Pro práci s tímto typem databáze se využívá objektově orientovaný jazyk, který je využit i pro aplikaci využívající tuto databázi. [12]

2.1.4.1.4 Relační

Relační databáze vychází z ukládaní dat do tabulek. V jedné tabulce se nemohou vyskytovat záznamy rozdílné povahy. To v praxi znamená, že například v tabulce uživatel se nemůže vyskytovat komentář. Data v tabulkách jsou rozděleny na řádky (záznamy) a sloupce (atributy). [13]

2.1.4.1.5 Objektově-relační

Objektově-relační vychází z relačních a objektově-orientovaných databází. Z objektově-orientované databáze přebírá objekty, třídy nebo dědění. Z relačních databázích se využívá tabulková struktura a datové typy. Protože se k ukládání dat využívá tabulková struktura, je potřeba k jejich správě využít dotazovací jazyk (například SQL). [14], [15]

Data z tabulek se převádí na objekty v aplikaci za využití rozhraní k tomu určenému. Toto rozhraní je schopné převést data na objekty nebo objekty na data. Za využití tohoto rozhraní je poté možno data v aplikaci načítat nebo upravovat. [15]

2.1.4.2 Systém řízení báze dat

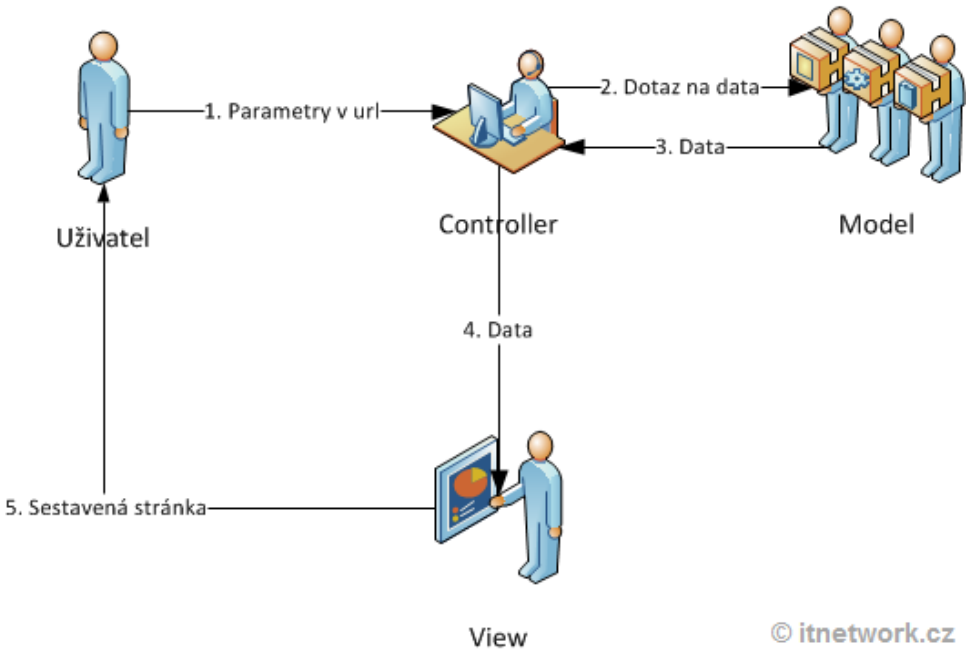
Systém řízení báze dat je software sloužící pro správu databází. Správou lze chápat vytváření databází nebo tabulek a úpravu jejich struktury nebo mazání. Zároveň však umožňuje se na takovou databázi nebo tabulku dotazovat a tím získávat data, která obsahuje. [11], [16]

Dalšími neméně důležitými úkoly SŘBD je definice schémat databází, umožnění přístupu k datům pro více uživatelů bez nežádoucího chování a narušení jejich integrity a zajištění odolnosti vůči chybám nebo výpadkům. [16]

2.1.5 MVC

MVC neboli Model-View-Controller je návrhový vzor jehož základní myšlenkou je rozdělení kódu na části, které slouží k vykonávání rozdílných úkolů (například zpracování různých operací a vykreslování webové stránky). [17]

Jak již název napovídá MVC je rozděleno do tří částí: model, pohled, řadič. Každá z těchto částí má svůj specifický účel, který je popsán v následujících kapitolách. [17]



Obrázek 4 MVC architektura [18]

2.1.5.1 Pohled (View)

Pohled se stará o vykreslení webu a zajišťuje, že se bude správně zobrazovat všem uživatelům. Pohled získává data od řadiče, ale ve své podstatě by neměl obsahovat žádnou logikou kromě základních podmínek, formátování nebo cyklů. [17]

2.1.5.2 Model

Model se stará o logiku a vše co s ní souvisí, ať už se jedná o výpočty nebo dotazy na databázi. Model nemá informace o tom, kdo nebo co zaslalo požadavek na data a v jakém formátu se má vrátit výstup. [18]

2.1.5.3 Řadič (Controller)

Řadič je ve své podstatě prostředník starající se o komunikaci s uživatelem, modelem a pohledem. Na rozdíl od dvou předchozích částí má hlubší znalost o funkcionalitě, protože se v něm nachází popisy procesů, které definují kroky potřebné pro dosažení jednotlivých výsledků. [18]

2.1.6 MAC adresa

Každé síťové rozhraní má od výrobce přiděleno MAC adresu, která by měla být celosvětově unikátní. Existují, ale jistá pravidla, které umožňuji výjimky, a proto se MAC adresy dělí na lokální/dočasné a adresy přidělené výrobcem. [19]

MAC adresy jsou 48bitové s rozdělením na 6 oktetů (například 50:E5:49:A2:80:61) a většinou využívají hexadecimální číslice. První polovina takovéto adresy je unikátní pro každého výrobce a je mu přidělena od sdružení IEEE, druhý polovina je unikátní v rámci daného výrobce, který si tuto unikátnost hlídá sám. [19]

2.2 Použité technologie

Kapitola použité technologie se zabývá vysvětlením technologií, které byly využity pro splnění vytyčených cílů práce.

2.2.1 Hyper Text Markup Language

Hyper Text Markup Language, který je v současnosti znám především zkráceně jako HTML je značkovací jazyk, který v roce 1991 vytvořen autorem Timem Bernersem-Lee, ale oficiálně byl zveřejněn až v roce 1995. Jeho hlavním účelem je formátovaní textu. Pro toto formátovaní se používají takzvané značky nebo také tagy, které existují jako párové nebo nepárové. [20], [21]

2.2.1.1 Struktura

Všechny dokumenty označené jako HTML musí dodržovat správnou strukturu, bez níž by nebyly validní. Základní struktura každého dokumentu by měla vypadat jako na následujícím obrázku. [22]

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Title of the document</title>
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

Obrázek 5 Vzorový HTML kód [23]

Z tohoto obrázku vyplývá, že by dokument měl obsahovat značky *<!DOCTYPE html>*, *<html>*, *<head>* a *<body>*. Značka *<!DOCTYPE html>* slouží pro definici typu dokumentu. V párové značce *<head>* se nachází základní informace o stránce jako jsou metadata, titulek stránky, znaková sada a další. V párové značce *<body>* se nachází samotný obsah webu, který se zobrazuje uživateli. Obě tyto značky musí být obaleny značkou *<html>*, která slouží jako obal pro všechny ostatní značky, kromě té definující typ dokumentu. Značka *<html>* by měla

vždy obsahovat atribut *lang*, který definuje jazyk obsahu stránky pro vyhledávače. [20], [23], [24]

2.2.1.2 Základní syntaxe

2.2.1.2.1 Značky

Syntaxe html se skládá značek nebo také z tagů, které lze identifikovat dle toho, že jsou obklopeny ostrými závorkami (<, >). Všechny značky lze rozdělit na párové a nepárové podle toho, zda mají uzavírací značku. Typickým příkladem párové značky je <*body*>, které se uzavírá pomocí </*body*>. Pro nepárovou značku je to poté tag <*img*>, který se neuzavírá žádnou značkou. [20]

2.2.1.2.2 Atributy

Vlastnosti a parametry značek se doplňují atributy, které se vyskytují v první značce. Formát atributů je vždy stejný a to: název="hodnota". [25], [26]

```
<img src="logo.gif" width="40" height="60" alt="Logo" class="img">
```

Obrázek 6 Vzorový element s atributy [25]

Na výše uvedeném obrázku se nachází značka <*img*> využívaná pro vykreslování obrázků. Dále se zde vyskytují atributy *src*, *width*, *height*, *alt* a *class*. Pro tento konkrétní příklad atribut *src* definuje obrázek/cestu k němu, *width* a *height* definují jaké bude mít po vykreslení rozměry, *alt* obsahuje text, který se vypíše v případě, že se nepodaří zobrazit obrázek a *class* obsahuje název třídy sloužící pro stylování obsahu. [25]

2.2.2 PHP

PHP nebo také Hypertext Preprocessor je skriptovací jazyk sloužící pro tvorbu webových stránek s dynamicky generovaným obsahem. Obsah je generován na serveru, oproti například populárnímu JavaScriptu a před vrácením odpovědi klientovi je převeden do formátu HTML. Z toho důvodu je možné do kódu umístit citlivé informace, mezi které mohou patřit například přístupové informace do databáze nebo k e-mailovému serveru, bez jakéhokoliv rizika, že je bude možné kýmkoliv zobrazit. Z toho plyne i nevýhoda tohoto skriptovacího jazyka a to, že v případě změny dat nedojde k jejich zobrazení do doby, než uživatel znovu načte danou stránku. [27], [28], [29]

Oproti již zmíněnému HTML a JavaScriptu není zajištěna funkčnost PHP na každém zařízení. Pro jeho správnou funkcionalitu je vyžadováno PHP na cílovém webovém serveru nejprve nainstalovat a následně nakonfigurovat. Pro tento účel se využívá webový server (například Apache a IIS) [29], [30]

2.2.2.1 Základní syntaxe

2.2.2.1.1 Proměnné

Proměnné ve skriptovacím jazyce PHP lze poznat, na základě toho, že začínají značkou dolaru (*$*) a následuje název, rovná se, hodnota a středník (*$promenna="hodnota";*). Způsob zápisu hodnoty se liší podle typu této hodnoty. Hodnoty textové povahy jsou zapisovány v uvozovkách, ale číselné nebo logické (*true*/*false*) jsou zapisovány bez uvozovek. [30]

2.2.2.1.2 Uzavírání kódu

Pro správné fungování PHP skriptu je potřeba použít správnou otevírací a v případě, že soubor není ukončen PHP kódem i uzavírací značku. Pro tento účel je možné využít běžný způsob zápisu (*<?php ?>*) nebo zkrácený (*<? ?>*). Se zkráceným způsobem zápisu je spojeno riziko, že není zajištěna správná funkcionalita na všech zařízeních, protože ve výchozím nastavení je zakázán, a proto je doporučeno využít běžný způsob zápisu. [30]

2.2.2.1.3 Ukončení příkazu

V PHP jsou jednotlivé příkazy ukončeny za pomoci středníku (*;*). Tento způsob ukončení příkazu je povinný až na výjimku, kterou je, že po příkazu následuje uzavírací značka, ale i přesto je doporučeno jej za příkaz umístit. [30]

2.2.2.1.4 Komentáře

Pro přehlednost kódu lze v PHP využít komentáře, které slouží primárně k popisu funkce jednotlivých části kódu, ale za použití „*TODO:*“ je lze použít i k označení částí, které je nezbytné dokončit nebo upravit. [30]

Pro komentování lze v PHP využít 4 způsoby. První dva způsoby jsou určeny komentování jednotlivých řádku, třetí umožňuje komentovat libovolný počet řádků a čtvrtý slouží jako dokumentační komentáře. Prvními dvěma způsoby jsou dvě lomítka (*//*) a hashtag (*#*). Pro komentování libovolného počtu řádků se používá lomítko a hvězdička (*/**) na začátku komentované části a hvězdička s lomítkem (**/*) při ukončení komentování dané části. Čtvrtý typ komentářů je velmi podobný třetímu s tím rozdílem, že začíná lomítkem a dvěma

hvězdičkami (/**) a všechny řádky začínající hvězdičkou. Kromě komentování kódu jej lze také využít ke generování dokumentace za použití nástroje phpDocumentor. V dokumentačních komentářích lze pro označení kódu se speciálním významem využít znak zavináče. [30]

2.2.3 Python

Python je programovací jazyk umožňující využití na velkém množství platforem bez jakéhokoliv omezení. Nezáleží tedy na tom, zda skript bude spouštěn na operačním systému Windows, Mac OS X nebo systémech založených na Unixovém jádře, všude bude spuštěn stejně. Je zde sice možnost vytvořit program určený pro specifickou platformu, ale jedná se spíše o vzácnost, protože je většina knihoven, včetně té základní vytvořena tak, aby byly zcela nebo z větší části multiplatformní. [31]

V základní konfiguraci Python obsahuje obsáhlou knihovnu, která umožňuje jeho širokou škálu využití nehledě na to, zda se jedná o práci se soubory nebo vytvoření webového serveru. Pro případ, že by však základní knihovna neobsahovalo potřebné knihovny lze využít knihovny třetích stran, které přidají požadovanou funkci a významně rozšíří možností využití tohoto programovacího jazyka. [31]

Programy využívající tento jazyk lze většinou poznat podle přípony **.py* nebo v případě, že je využito grafické rozhraní **.pyw*. [31]

2.2.3.1 Základní syntaxe

2.2.3.1.1 Proměnné

V jazyce Python je proměnná vytvořena až v okamžik nastavení její hodnoty za využití názvu proměnné, rovnítka a hodnoty (*promenna="hodnota"*). Oproti jiným programovacím jazykům není tedy potřeba proměnnou nejprve deklarovat a až poté přiřazovat hodnotu. [32]

Hodnota se proměnné přiřazuje pomocí znaménka rovná se. Nejprve se napíše název proměnné, poté znaménko rovná se a následně hodnota. Python také umožňuje přiřadit hodnotu několika proměnným ve stejný okamžik. Toho lze docílit napsáním názvu první proměnné, poté znaménka rovná se, název druhé proměnné, další znaménko rovná se a následně hodnotu, která se má těmto proměnným přiřadit (*a = b = c = "hodnota"*). [32]

2.2.3.1.2 Bloky kódu

Python na rozdíl od jiných programovacích jazyků nevyužívá závorky pro označování bloků, ale bloky jsou rozlišovány na základě odsazení. Pro správné fungování musí být tedy řádky, které jsou součástí jednoho bloku stejně odsazeny. [31]

```
if lines < 1000:
    print("small")
elif lines < 10000:
    print("medium")
else:
    print("large")
```

Obrázek 7 Příklad kódu s bloky [31]

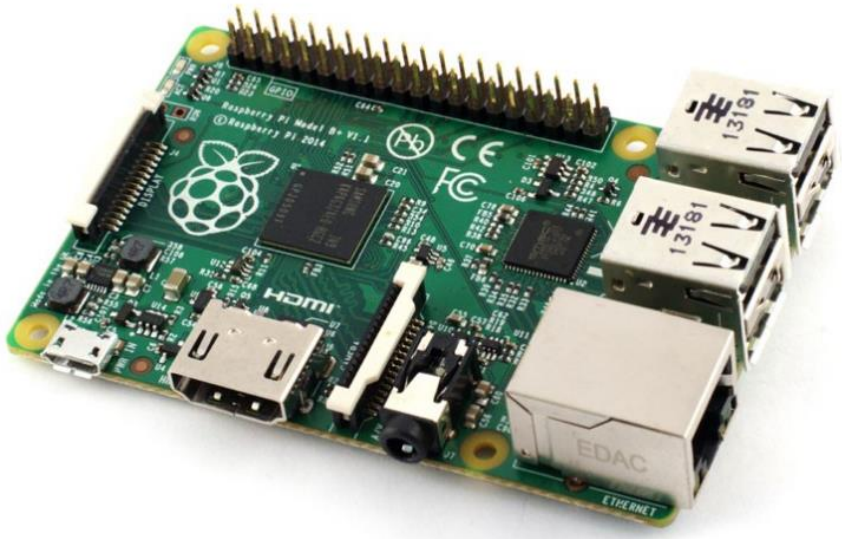
Na obrázku výše je znázorněno blokové rozdělení kódu a je zde patrné, že za podmínkami se nachází dvojtečky, které se například v programovacích jazycích Java nebo C++ nevyskytují. Dvojtečka je v Pythonu vkládána za podmínky, cykly a všechny části kódu, kde je vyžadována. [31]

2.2.3.1.3 Komentáře

Pro přidávání komentářů v kódu lze v jazyce Python využít hashtag, který je určen pro jednořádkové komentáře. Tedy vše, co se nachází v kódu za ním až do konce řádku. [31]

2.2.4 Raspberry Pi

Raspberry Pi je miniaturní počítač s možnostmi jeho využití ve velkém množství odvětví nehledě na to, zda se jedna o ovládání hardwaru pomocí programovacího jazyku Python nebo použití tohoto zařízení jako multimediálního centra. Oproti běžnému počítači výhoda Raspberry Pi spočívá v rozměrech, ceně a přítomnosti portů GPIO, které se využívají pro připojení přídavných zařízení. [33]

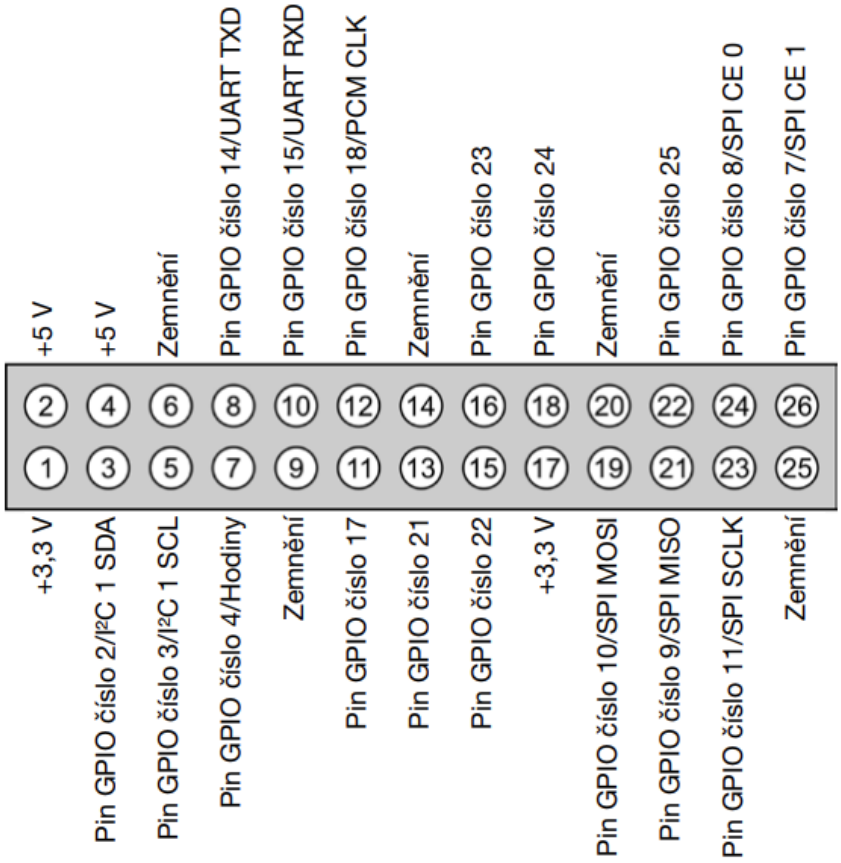


Obrázek 8 Raspberry Pi model B+ [33]

2.2.4.1 Rozšiřitelnost

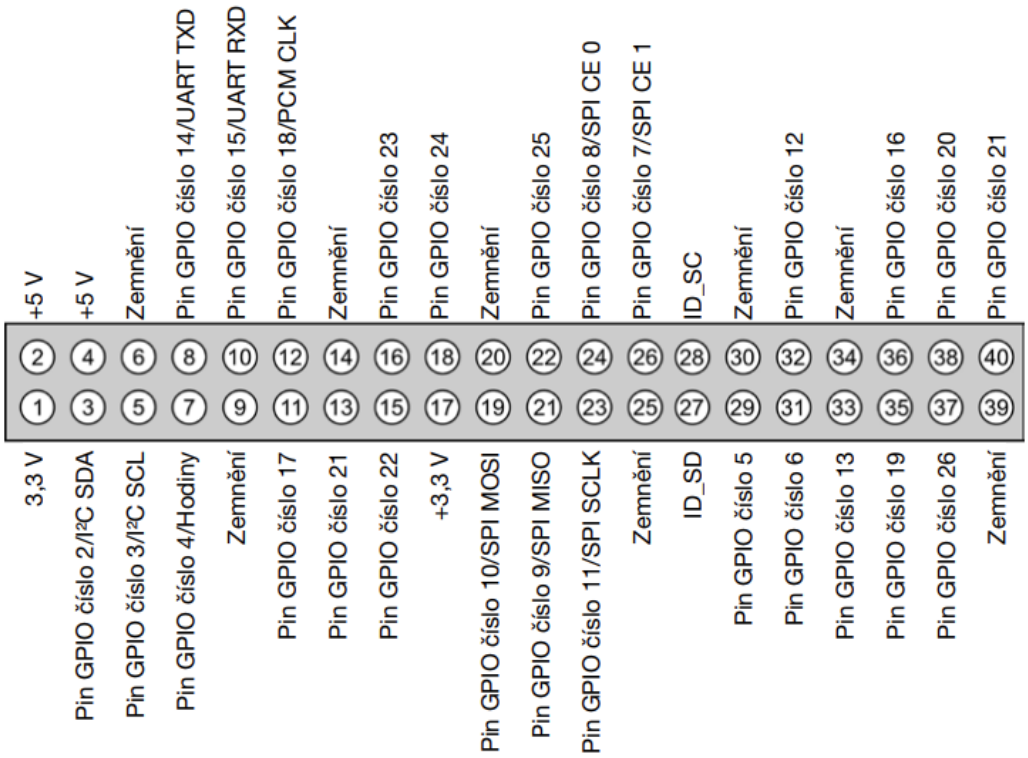
Jak již bylo zmíněno v kapitole 2.2.4, Raspberry Pi obsahuje port GPIO, který umožňuje k desce připojit přídavná zařízení. Pro správnou funkcionalitu přídavných zařízení je potřeba dbát na to, aby využívaly napětí 3,3 V, protože Raspberry Pi pro svoje interní komponenty využívá právě toto napětí. [33]

Množství rozšíření, které lze připojit závisí na síle napájení a počtu, které se odvíjí od modelu Raspberry Pi. [33]



Obrázek 9 26pinový GPIO konektor [33]

Starší model obsahovaly pouze 26pinový GPIO konektor, ale u novějších se vyskytuje 40pinový konektor se zachovaným schématem prvních 26 pinů. [33]



Obrázek 10 40pinový GPIO konektor [33]

Z obrázků 26pinový GPIO konektor [33] a 40pinový GPIO konektor [33] je patrné, že piny jsou řazeny do dvou řad a číslovány střídavě (v dolní řadě jsou piny s lichým číslem a v horní se sudým). Na obou schématech lze vidět, že konektor neobsahuje pouze univerzální GPIO piny, ale i piny označené jako UART, I^2C nebo SPI. [33]

2.2.4.1.1 Univerzální piny

Tento typ pinů umožňuje ovládat komponenty skrze ně připojené dvěma stavy, a to vysokým a nízkým. Vysoký odpovídá kladnému napětí 3,3 V a nízký oproti tomu odpovídá zemnění neboli napětí 0 V. Obě verze GPIO konektoru obsahují rozdílný počet pinů pro univerzální využití. Zatímco 26 pinový konektor obsahuje 8 pinů, 40 pinový jich obsahuje 17. [33]

2.2.4.1.2 UART

UART je sériová sběrnice sloužící pro zprávy. Pro tuto sběrnici slouží piny 8 a 10. Pin 8 slouží pro odesílání (transmit) a pin 10 slouží pro příjem (receive) signálu. Pokud je Raspberry Pi připojeno k zařízení se schopnosti tyto zprávy zobrazit, lze skrze něj sledovat zprávy jádra operačního systému Linux a použít jej jako diagnostický systém. Rychlost této sběrnice je běžně 115 200 bit/s, ale lze ji změnit v souboru cmdline.txt. [33]

2.2.4.1.3 I^2C

Sběrnice I^2C je určena k umožnění komunikace mezi integrovanými obvody. U Raspberry PI je jedním z obvodů procesor, který se stará o výpočetní operace tohoto zařízení. [33]

Tato sběrnice je dostupná na pinech s číslem 3 a 5. Pro signál SDA (Serial Data Line) slouží pin 3 a pro SCL (Serial Clock) signál slouží pin číslo 5. [33]

2.2.4.1.4 SPI

Tato synchronní sběrnice je schopna oproti I^2C dosáhnout vyšších rychlostí. Využívá čtyř vodičů a více linek výběru čipu díky čemuž je možné komunikovat s větším množstvím cílových zařízení. [33]

Ke sběrnici SPI se lze připojit prostřednictvím pinů 19 (SPI MOSI), 21 (SPI MISO), 23 (SLCK) a dvojici linek výběru čipu na pinech 24 a 26. Linky výběru čipu slouží pro volbu jednoho z maximálně dvou nezávislých vedlejších zařízení. [33]

2.2.5 Symfony

Symfony je PHP framework založený na architektuře MVC sloužící pro vývoj webových aplikací. Je složen ze skupiny PHP komponent, které rozšiřují jeho funkcionalitu a umožňují jednodušší vytváření webových aplikací. Nejznámější z těchto komponent je šablonovací systém Twig a ORM framework Doctrine. [34]

2.2.5.1 Doctrine

Doctrine je ORM framework starající se o mapování objektů na relační databázi. Dokáže tedy převést entitu (objekt nesoucí data) na strukturu databáze, ale zároveň ji lze využít i pro opačný směr, tedy převést strukturu databáze na entity. [35], [36]

2.2.5.1.1 Vrstvy

2.2.5.1.1.1 Common

V této vrstvě se nachází obecná rozhraní, třídy a knihovny (anotace, cache, události, práce s kolekcemi). Oproti DBAL a ORM je na ostatních vrstvách nezávislá, ale je jimi využívána. [35]

2.2.5.1.1.2 DataBase Abstraction Layer (DBAL)

DBAL se stará o abstrahování aplikace od konkrétního typu databáze. Stará se o integraci jazyka DQL. Tuto vrstvu nelze využít bez vrstvy Common z důvodu její závislosti, oproti tomu vrstva ORM není pro bezproblémový chod potřeba. [35]

2.2.5.1.1.3 Object-Relational Mapping (ORM)

Tato vrstva se stára o samotnou podstatu frameworku Doctrine, a to mapování entit na relační databázi a jejich následné načítaní a uchovávání. Jako jediné je ORM závislé na vrstvách Common i DBAL. [35]

2.2.5.1.2 Entita

Entita je objekt, jehož základním úkolem je uchovávání a předávání dat, ale měla by obsahovat validaci vstupu. Validací vstupu lze chápat například tak, že délka nebo datový typ odpovídá atributu, který se předává entitě. Dále může entita obsahovat jakékoliv metody, které mohou pracovat s daty, ale pouze s těmi, které dané entitě náleží. [36]

2.2.5.1.3 DQL

DQL neboli Doctrine Query Language využívá vlastní dotazovací jazyk, který je velmi podobný SQL až na to, že místo tabulek a sloupců pracuje s objekty a atributy, které obsahují. [36]

```
// takto by vypadal SQL dotaz do databáze
SELECT `name`, `email`, `ip`
FROM `user`
WHERE `id` > 10
ORDER BY `name` DESC

// a takto příkaz vypadá v DQL
SELECT u
FROM Jmenny\Prostor\User u
WHERE u.id > 10
ORDER BY u.name DESC
```

Obrázek 11 Rozdíl mezi SQL a DQL [36]

2.2.5.2 Twig

Twig je šablonovací systém využívaný v PHP frameworku Symfony, ale i v jiných. Jeho hlavním účelem je oddělení PHP a HTML bez ztráty možností, které tato kombinace přináší. Twig umožňuje i nadále využívat podmínky, cykly, proměnné, rozdělení souborů na více částí,

ale je čitelnější a umožňuje automatické nahrazování znaků se speciálním významem v obsahu proměnných za jiné odpovídající danému kontextu. [37], [38]

2.2.5.2.1 Základní syntaxe

Základní syntaxe šablonovacího systému Twig se skládá ze 3 syntaktických konstrukcí. Pro výpis hodnoty (například proměnné) slouží dvě složené závorky pro zahájení a dvě složené závorky pro uzavření (*{{ ... }}*). S veškerým obsahem nacházejícím se mezi těmito závorkami bude naloženo jako s proměnnou. Pro logiku (podmínky, cykly a další) se využívá následující zápis: *{% ... %}*, vše, co je mezi složenou závorkou a procentem bude systém chápat jako logickou operaci. Pro komentáře v Twigu existuje následující styl zápisu: *{# ... #}*, vše mezi složenou závorkou a hashtagem se bude považovat za komentář. [37]

2.2.6 MySQL

MySQL je takzvaný systém správy relačních databází, který vychází ze systému řízení báze dat a přidává další funkce. Server využívající MySQL zajišťuje přístup pro větší množství uživatelů s patřičnými oprávněními najednou a dohlíží na to, že mohou provádět pouze takové operace, které jim přidělená oprávnění umožňují. Pro práci s MySQL se využívá standardní dotazovací jazyk SQL. [39]

2.2.6.1 Třídění dat

Vzhledem k tomu, že se jedná o relační databázi, třídění dat je založeno na tabulkách a záznamech v nich (viz kapitola 2.1.4.1.4), ale k tomu ještě v MySQL přibývá typování. Typování znamená, že každý atribut musí mít pevně stanovený datový typ (číslo, pravda/nepravda, krátký text, dlouhý text, datum), který definuje, jaké hodnoty lze do něj uložit. U každého řádku by měl být definován unikátní identifikátor. [13]

2.2.6.2 SQL

SQL je jazyk sloužící pro práci především s relačními databázemi vyvinutý společností IBM. S využitím SQL je možné zapisovat data, dotazovat se na ně a upravovat je. Pro tyto operace existuje několik typu příkazů. [40]

2.2.6.2.1 SELECT

Příkaz SELECT slouží k výběru dat z databáze. Lze jej využít pro výpis z jedné, ale i většího množství tabulek, filtrovat vypsané hodnoty, vytvářet vypočtené sloupce, početní operace (průměr, součet, počet a další) a množinové operace. [40]

2.2.6.2.2 Data Modification Language

Příkazy spadající pod Data Modification Language slouží pro přidávání záznamů do tabulek v databázi a možnost jejich úprav nebo případného smazání. [40]

2.2.6.2.3 Transakce

Příkazy typu Transakce slouží pro ochranu dat v případě výskytu poruchy. V případě poruchy se musí provést celá skupina příkazu, bez možnosti přeskočení některých z příkazů nacházejících se v této skupině. [40]

2.2.6.2.4 Data Definition Language

Do typu Data Definition Language patří příkazy sloužící pro úpravu struktury databáze. Mezi tyto úpravy patří vytváření tabulek, jejich úprava, práce s indexy a práce s relacemi mezi tabulkami. [40]

2.2.6.2.5 Správa databáze

Tento typ příkazů slouží pro vytváření, mazání databází a jejich správu. Pod správu databáze se řadí nastavování různých parametrů nebo správa uživatelů a jejich oprávnění. [40]

2.2.7 Důvod pro zvolené technologie

Vzhledem k tomu, že vytvořené řešení má být uživatelsky přívětivé je nezbytné, aby i využití jednotlivých částí bylo co nejjednodušší. Z tohoto důvodu byl jako základ pro administraci zvolen skriptovací jazyk PHP, který je díky své popularitě podporovaný většinou poskytovatelů webových hostingů. [41]

Programovací jazyk Python byl pro tento projekt zvolen z důvodu podpory velkého množství platforem (Windows, Mac OS a Linux). Díky tomu je možné využívat programy napsané v tomto jazyce na různých platformách bez potřeby úpravy kódu. [31]

Počítač Raspberry Pi bylo zvoleno jako vhodné řešení této práce z důvodu rozměru a možnosti připojitelných zařízení skrze GPIO porty. [33]

PHP framework Symfony byl vybrán na základě popularity mezi vývojáři, možnosti rozšiřitelnosti a vhodnosti pro komplexní projekty. Alternativou je framework Laravel, ale vzhledem ke zkušenostem autora s frameworkem Symfony bylo zvoleno právě Symfony. [42]

2.3 Existující řešení

Kapitola existující řešení obsahuje alternativy ke zvolenému způsobu řešení, které lze zvolit v případě, že uživatel vyžaduje již kompletní produkt bez potřeby jeho kompletace nebo složité instalace. Tato zařízení byla vybrána na základě jejich možnosti využití pro vzdálené měření teplot.

2.3.1 Hardwarová řešení

2.3.1.1 STE2 r2

STE2 r2 je internetový teploměr vyráběný firmou HW group s možností připojení k internetu prostřednictvím ethernetového kabelu nebo Wi-Fi. V případě, že je použit ethernetový kabel lze za předpokladu podpory technologie PoE na zařízení nacházejícím se na opačné straně kabelu využít napájení skrze tento kabel, v opačném případě se k napájení využívá 5 voltový napájecí adaptér. Zařízení obsahuje vlastní webový server, skrze který lze sledovat naměřené hodnoty a měnit nastavení. Avšak v případě potřeby, lze zařízení připojit do portálu HWg-cloud.com, který je neplacenou alternativou služby SensDesk od stejné společnosti nebo do jiného softwaru k tomu určeného. [43]

Obrázek 12 STE2 r2 [43]

2.3.1.1.1 Limity zařízení

Zařízení STE2 r2 umožňuje připojit v jeden okamžik až 5 senzorů skrze 1-Wire nebo 1-Wire UNI a maximálně 2 detektory skrze kontakty pro digitální vstup. Pro každý připojený

detektor či senzor lze nastavit hranice při kterých dojde k odeslání upozornění na specifikovaný cíl (e-mail, SMS). [43]

2.3.1.1.2 Připojitelná čidla a senzory

K STE2 r2 lze díky velkému množství dostupných konektorů připojit celá škála typu čidel a senzorů. Mezi tyto typy patří následující senzory a detektory: teplotní, relativní vlhkosti, CO2, VOC, světla, proudu AC/DC, střídavého napětí, 4-20 mA převodníky, zaplavení vody, proudění vzduchu, dveřního kontaktu, pohybu, úniku plynu, přítomnosti nebo vibrační. [43]

2.3.1.1.3 Cena zařízení

Výrobce na svých stránkách cenu nezveřejňuje, nicméně dle internetových obchodů se cena pohybuje okolo 5360 Kč bez DPH za zařízení s jedním teplotním čidlem a napájecím adaptérem. [44]

2.3.1.2 TME: Ethernetový teploměr

Ethernetový teploměr TME od společnosti Papouch jak již název napovídá je zařízení schopné měřit teploty. Zařízení je schopné měřit teploty v rozsahu od -55 °C do +125 °C za využití jednoho teplotního čidla. Zařízení obsahuje vlastní jednoduché webové rozhraní pro výpis teploty a správu zařízení chráněné jménem a heslem. Webové rozhraní, ale není jediná možnost pro výpis naměřených teplot. Naměřené hodnoty lze odesílat prostřednictvím různých protokolů do jiných služeb nebo webových aplikací. [45]

Obrázek 13 TME: Ethernetový teploměr [45]

2.3.1.2.1 Limity zařízení

Zásadním limitem tohoto zařízení je přítomnost maximálně jednoho teplotního čidla bez možnosti jakéhokoliv rozšíření nebo odpojení čidla. Pro toto čidlo lze nastavit odesílání

upozornění na mail v případě překročení specifikovaných mezí, které lze nastavit prostřednictvím webového rozhraní nacházejícího se na zařízení. [45]

2.3.1.2.2 Připojitelná čidla a senzory

Vzhledem k pevně přidělanému senzoru teploty nelze k zařízení připojit žádný jiný typ senzoru nebo čidla. [45]

2.3.1.2.3 Cena zařízení

Ethernetový teploměr TME stojí 2 990 Kč bez DPH a v ceně je zahrnuto teplotní čidlo, napájecí adaptér a 1 metr dlouhý nekřížený patch kabel. [45]

2.3.2 Softwarová řešení

2.3.2.1 TMEP

TMEP je webová aplikace sloužící pro výpis naměřených hodnot teploty a vlhkosti z čidel. Primárně je tato aplikace určena pro zařízení od společností MonkeyTech, Brrr.cz, Home Technologies a Papouch, ale lze ji využít i se zařízeními jiných firem. [46]

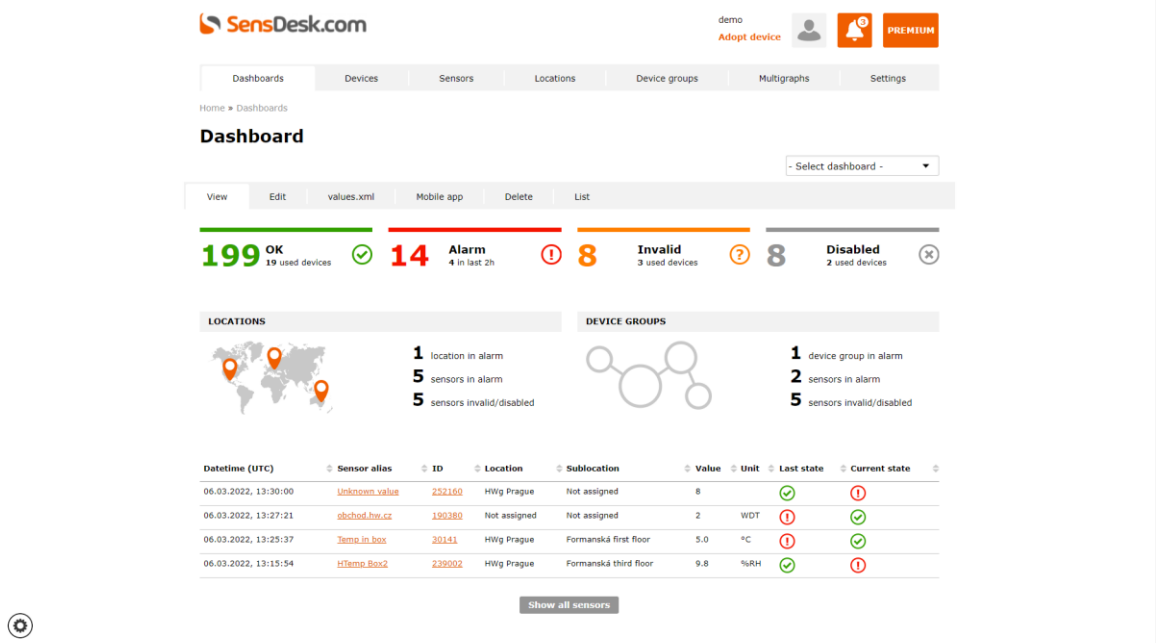
Obrázek 14 Dashboard aplikace TMEP [46]

2.3.2.1.1 Limity aplikace

Aplikaci lze využívat jako službu, ale je možné ji využívat na vlastním webovém serveru s databází. V případě, že je aplikace využívána jako služba lze využívat až 3 čidla pod jedním účtem při nekomerčním užití, ale lze kontaktovat autora a využít tak většího množství čidel i pro komerční účely. Pokud se však uživatel rozhodne tuto aplikaci využít na vlastním webovém serveru s databází, je nutné počítat s tím, že lze využít pouze 1 čidlo s jednoduchým výpisem bez jakékoliv administrace. Vzhledem k tomu, že je aplikace spustitelná na hostingu uživatele je možné ji upravit dle vlastních potřeb. [46]

2.3.2.2 SensDesk

SensDesk je softwarové řešení sloužící pro správu zařízení spadajících pod společnost HW group. V této službě lze zobrazit data ze všech zařízení připojených k danému účtu, odesílání upozornění a odesílání dat skrze Open API do jiných systémů. V administraci lze zobrazit teplotu, vlhkost, úniky vody, napětí, proud, spotřebu energie a další. v případě potřeby lze z administrace ovládat zařízení. [47], [48]



Obrázek 15 Dashboard ve službě SensDesk [49]

2.3.2.2.1 Limity aplikace

Jak již bylo zmíněno výše, službu SensDesk je možné využít pouze se zařízeními společnosti HW group a nelze tedy využít společně se zařízeními třetí strany. Dalším zásadním limitem je upravitelnost rozhraní, která vychází z toho, že je aplikace poskytována jako služba

a běží na serverech provozovatele. Administraci lze sice upravit, ale pouze některé základní části. [48]

2.3.2.2.2 Cena za využití

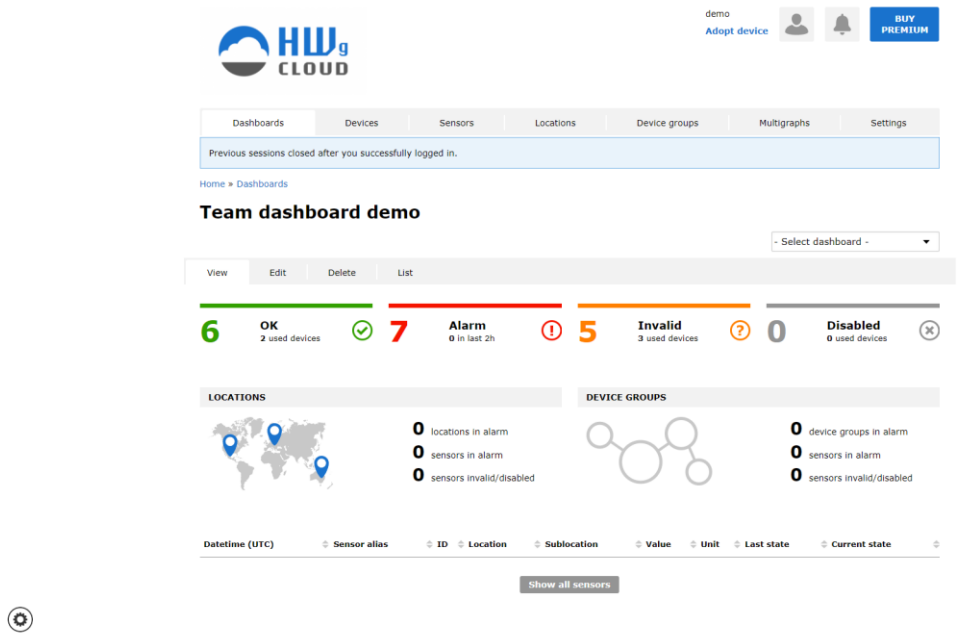
Služba SensDesk funguje na principu měsíčního a ročního předplatného. Cena tohoto předplatného se odvíjí od množství zařízení, uživatelů, množství upozornění za měsíc a délka uchování dat. Pro základní využití služby jsou připravena 3 předplatné s pevně stanovenými limity. Existuje však i čtvrté předplatné, které nemá pevně dáno limity a lze si jej sestavit na míru. [47], [48]

Tabulka 1 SensDesk – Předplatné [47]

Předplatné	5D	10D	25D
Počet zařízení	5	10	25
Limit uživatelů	1	2	3
SMS upozornění, mobilní hovory	40	75	100
Délka uchování dat	30	365	730
Cena/rok	144€	468€	1 188€

2.3.2.3 HWg cloud

HWg Cloud je bezplatnou obdobou služby SensDesk pro uživatele, kteří nechtějí nebo nepotřebují možnosti, které poskytuje placená verze. Tuto službu vytvořila také společnost HW group, a proto jsou tyto webové aplikace téměř totožné až na některé omezení, které placená verze neobsahuje. Se službou SensDesk má také aplikace společné omezení pouze na zařízení od společnosti HW group. [48], [50]



Obrázek 16 Dashboard služby HWg cloud [51]

2.3.2.3.1 Limity aplikace

Základní limity jsou oproti nejnižšímu plánu služby SensDesk výrazně lepší, protože umožňují do systému připojit až 20 zařízení, ale chybí podpora generování přehledů do PDF, mobilní aplikace nebo možnost využití Open API. Snížen je také limit pro upozornění na 2 cílové emaily a zachování dat maximálně na 10 dní zpětně. [48], [50]

2.3.2.3.2 Cena za využití

HWg cloud je zcela zdarma, ale toto využívání je podmíněno tím, že lze využít pouze se zařízeními od společnosti HW group. [50]

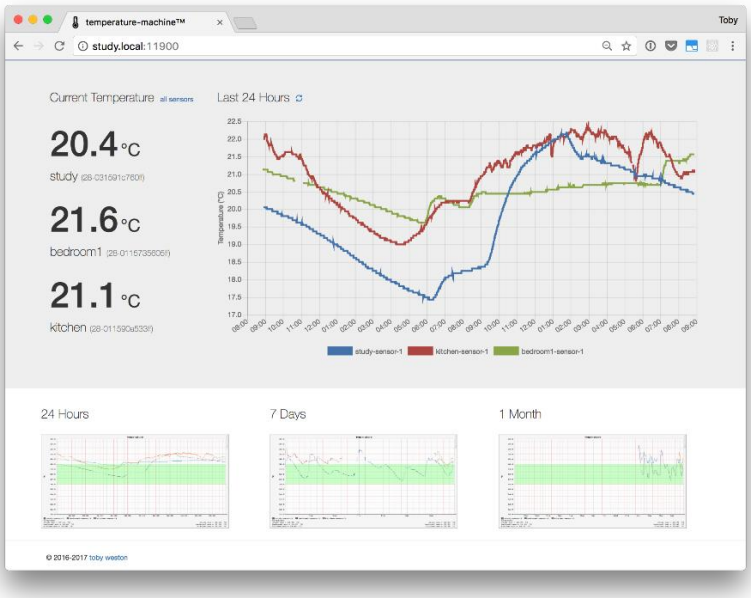
2.4 Alternativní řešení

Kapitola alternativní řešení obsahuje zařízení či software s jejichž využitím lze dosáhnout stejného nebo podobného cíle jako se zvoleným řešením, ale pro jejich využití je vyžadována určitá míra odbornosti nebo speciálních dovedností.

2.4.1 Temperature Machine

Temperature Machine je softwarové řešení sloužící pro měření teplot za využití Raspberry Pi se senzory DS18D20 a jejich následný výpis do webového rozhraní. Webové rozhraní umožňuje zobrazit teploty z většího množství zařízení. Ke každému zařízení lze připojit až 5 teplotních čidel zobrazovat z nich teploty samostatně nebo je zprůměrovat. [52]

Jednotlivá zařízení se rozlišují na server a klient. Zařízení sloužící jako server, ke kterému se připojují jednotliví klienti a odesílají na něj naměřené teploty. Server přijaté teploty uloží do databáze a umožní jejich vypsání ve webovém rozhraní. Oproti tomu klient naměřené teploty neukládá, ale pouze je odesílá. Oba typy umožňují připojení teplotních čidel, to znamená, že zařízení označené jako server je schopné teploty měřit a zároveň zpracovat data z ostatních zařízení. [52]



Obrázek 17 Temperature Machine Dashboard [52]

2.4.1.1 Limity aplikace

Limitujícími faktory Temperature Machine jsou přístupnost odkudkoliv a možnosti webového rozhraní. Vzhledem k tomu, že webové rozhraní je provozováno na zařízení, které je provozováno v režimu server, může být problematické přistupovat k naměřeným hodnotám odkudkoliv. [52]

Webové rozhraní kromě zobrazení teplot a zobrazení dat v grafech nic jiného neumí, a tak nelze využít ke správě připojených zařízení. [52]

2.4.1.2 Cena za využití

Temperature Machine je zcela zdarma, ale pro využívání tohoto softwaru je potřeba nakoupit hardware, který umožní měřit teploty. [52]

2.4.2 Arduino

Arduino je open-source platforma sloužící pro vývoj a návrh zařízení nejrůznějších zaměření. Nejvíce podobnosti lze hledat se zařízením Raspberry Pi, ale i přesto jsou mezi nimi zásadní rozdíly z důvodu zacílení těchto dvou platforem. Jak již bylo zmíněno, Arduino slouží spíše pro návrh a vývoj. Naproti tomu Raspberry Pi je počítač, který lze využít k běžným úkonům jako je prohlížení internetu a další. Kde se však tyto zařízení nejvíce podobají je možnost připojit různé moduly a rozšiřující desky, které umožňují nespočet využití a jediným omezujícím faktorem tak je množství pinů. [53], [54]

Obrázek 18 Arduino Uno [53]

2.4.2.1 Vývojové prostředí

Programy pro toto zařízení se vytvářejí v aplikaci nazvané Arduino IDE, které je kompatibilní s počítači využívající Windows, Linux nebo Mac OS. Tento software kromě vytváření programů umožňuje kompilaci kódu a jeho nahrání na Arduino. K vytváření programů se využívá programovací jazyk Wiring, ale lze použít i jazyky C a C++. [54], [55]

2.4.2.2 Limity zařízení

Jak již bylo zmíněno v úvodu, Arduino v možnostech hardwaru není limitováno téměř ničím z důvodu množství modulů a rozšiřujících desek umožňujících nespočet využití. [53], [54]

2.5 Vyhodnocení před vytvořených a alternativních řešení

Tabulka 2 porovnání analyzovaných řešení

Název řešení	Typ řešení	Proprietární	Modifikovatelnost	Počet senzorů	Cena
STE r2	Hardwarové i softwarové	Ano	Nelze modifikovat hardware, Software lze zvolit jiný	Až 5	5 360 bez DPH
TME	Hardwarové i softwarové	Ano	Nelze modifikovat hardware, Software lze zvolit jiný	1	2 990 bez DPH
TMEP	Softwarové	Ano	Ano, v případě provozu na vlastním hostingu	/	Není zmíněno
SensDesk	Softwarové	Ano	Ne	/	Záleží na zvoleném plánu
HWg cloud	Softwarové	Ano	Ne	/	Zdarma k zařízením od HW group
Temperature machine	Hardwarové i softwarové	Ne	Ano	Až 5	Cena záleží na zvoleném hardwaru

Z analýzy existujících a alternativních řešení vyplynulo, že existující řešení mají omezení ve formě množství připojitelných senzorů v jeden okamžik, míry upravitelnosti z důvodu proprietárního softwaru nebo hardwaru. Analyzovány byly i neproprietární řešení, která obsahují omezení ve formě možnosti přístupu k naměřeným hodnotám nebo množstvím funkcí.

Mezi alternativními řešení byla analyzována open-source platforma Arduino, která bez úprav, modulů nebo softwaru nedokáže oproti ostatním zanalyzovaným řešením teplotu měřit. Nicméně toto řešení lze zvolit jako alternativu k počítači Raspberry Pi, a proto bylo zahrnuto.

Na základě výsledků provedené analýzy existujících či alternativních řešení, bude v práci převzata možnost připojení více zařízení do administrace včetně možnosti připojení více než 1 teplotního čidla ke každému zařízení. Administrace bude umožňovat správu zařízení, senzorů, uživatelů a oznámení.

2.6 Funkční a nefunkční požadavky

2.6.1 Funkční požadavky

Pro splnění cílů práce a na základě analýzy existujících řešení by webová administrace měla splňovat následující funkční požadavky:

- Správa zařízení
    - Přehled zařízení – výpis zařízení se základními údaji
    - Přidávání zařízení
    - Mazání zařízení
    - Editace zařízení – editace základních údajů o zařízení
    - Základní změna nastavení zařízení – úprava nastavení pro funkce systému
    - Detail zařízení – výpis naměřených teplot, všechny informace o zařízení
    - Výpis hodnot získaných ze zařízení – aktualizující se grafy
    - Stažení instalačních souborů pro zařízení
- Správa oznámení
    - Přehled oznámení – výpis všech oznámení
    - Mazání oznámení
    - Změna stavu oznámení – potvrzení oznámení, zrušení potvrzení
- Přehled oznámení podle uživatele – výpis oznámení pro jednotlivé uživatele na základě cílového příjemce oznámení, přístupné pro roli uživatel
- Správa uživatelů
    - Přehled uživatelů – výpis všech uživatelů
    - Přidávání uživatelů
    - Editace uživatelů
    - Mazání uživatelů
- Správa vlastního uživatelské účtu
    - Změna hesla
    - Změna e-mailové adresy nebo uživatelského jména
- Přihlášení

- Odhlášení
- Prvotní uživatelská registrace při instalaci webové administrace – umožňuje registraci při prvním připojení do administrace

Pro skripty nacházející si na zařízení měřící teploty jsou na základě cílů práce vytyčeny tyto funkční požadavky:

- Odesílání naměřených teplot
- Odesílání dat ze zařízení (například MAC adresa)
- Aktualizace konfiguračního souboru na zařízení

2.6.2 Nefunkční požadavky

Webová administrace a skripty nacházející se na zařízení by měly splňovat následující nefunkční požadavky:

- Zabezpečená komunikace
- Dostupnost naměřených hodnot až 90 kroků zpětně
- Senzory budou nadále odesílat teploty v případě výpadku některého z ostatních senzorů
- Uživatelsky přívětivé rozhraní
- Obnovení odesílání dat v případě výpadku elektřiny
- Jednoduchost instalace skriptů na zařízení

3 Vlastní práce

3.1 Softwarové požadavky

Pro správný chod a instalaci je vyžadováno softwarové vybavení ve specifických verzích. V případě použití jiných verzí není zaručena správná funkcionalita webové administrace a skriptů nacházejících se na zařízení.

Webová administrace vyžaduje pro instalaci správce balíků a závislostí s názvem Composer ve verzi alespoň 1.11.99.4 (testováno na verzi 2.1.12) a skriptovací jazyk PHP alespoň ve verzi 7.2.5 (testováno na verzi 7.4.7).

Skripty nacházející se na zařízení vyžadují programovací jazyk Python 3 (testováno na verzi 3.9.2).

3.2 Webová Administrace

Tato kapitola se bude zaměřovat na vybrané části kódu, webové administrace a její instalaci. Z důvodu rozsáhlosti nebude v této práci popsán veškerý kód, ale pouze vybrané části. Zbylé části kódu, které nebudou popsány v této kapitole jsou zdokumentovány komentáři přímo v kódu.

Zmíněnou webovou administraci je možné otestovat na URL adrese https://bp.petrkruntorad.cz/ a lze se do ní přihlásit pomocí testovacího účtu s uživatelským jménem „*bpTestUser*“ a heslem „*J9T4CCzUzGv1Gk7yVZz8re*“ bez uvozovek. Tento účet má administrátorská práva a může tedy v systému vykonávat vše, co systém umožňuje.

Kromě frameworku Symfony, který se stará o všechny operace na pozadí byla na popředí využita šablona AdminLTE [56], díky které je uživatelské rozhraní jednoduché a uživatelsky přívětivé.

Vytvořený software se všemi nezbytnými skripty a soubory je možné stáhnout z platformy GitHub na adrese https://github.com/petrkruntorad/bachelors-thesis-kruntorad.

3.2.1 Uživatelská role

V administraci se nachází 2 uživatelské role, a to uživatel a administrátor. Účet, který má roli uživatel je oprávněn systém využívat k zobrazení senzoru, zobrazení naměřených hodnot ze zařízení, zobrazení a schválení oznámení pro senzory, u kterých je označen jako cílový uživatel pro příjem oznámení a správě svého profilu. Oproti tomu účet, který má roli

administrátor může vše, co účet s rolí uživatele, ale navíc může spravovat jednotlivá zařízení, uživatele a oznámení.

Možnost spravovat zařízení znamená, že uživatel může přidávat nová zařízení, stahovat instalační balíčky, schvalovat i zrušit schválení pro zařízení, měnit nastavení i detaily zařízení a mazat senzory nebo jednotlivá zařízení. Pod správu uživatelů je zařazeno vytváření nových uživatelů, editace a mazání již existujících. Správa oznámení umožňuje mazat, potvrzovat nebo rušit potvrzení oznámení, kde daný uživatel není označen jako příjemce pro notifikace.

3.2.2 Řadiče

Pro přehlednost a snadnou orientaci v kódu je funkcionalita rozdělena na větší množství řadičů s názvy umožňující pochopení jejich obsahu a určení.

3.2.2.1 AdminController

Řadič *AdminController* obsahuje metody sloužící pro základní funkcionality administrace. Nachází se zde například metoda *index*, která je určena pro výpis stránky přehled.

3.2.2.2 DeviceController

V tomto řadiči se nachází funkcionality potřebné pro vše související se zařízeními. Takovou funkcionalitou je například výpis zařízení, zobrazení detailu, přidávání zařízení a další.

3.2.2.3 NotificationController

V řadiči *NotificationController* se nachází metody potřebné pro výpis, potvrzení, zrušení potvrzení nebo mazání oznámení.

3.2.2.4 ProfileController

V tomto řadiči se nachází metody sloužící pro správu uživatelského profilu. Mezi takové metody zařazena změna hesla nebo údajů aktuálního uživatele.

3.2.2.5 SecurityController

Tento řadič je určen pro funkcionalitu spojenou s bezpečností a ověřováním uživatelů. Mezi takovou funkcionalitu lze zařadit přihlašování nebo prvotní registraci do systému.

3.2.2.6 SensorController

SensorController je řadič sloužící pro funkcionalitu související se zařízeními. Za takovou funkcionalitu lze označit například smazáni senzoru.

3.2.2.7 SensorDataController

Řadič *SensorDataController* je určen pro metody sloužící k získávání dat pro senzory vypsané na stránce detailu jednotlivých zařízení.

3.2.2.8 UserController

V řadiči *UserController* jsou obsaženy metody vyhrazené pro správu uživatelů. Oproti řadiči *ProfileController* jsou metody v tomto řadiči přístupné pouze uživatelům s rolí administrátor.

3.2.3 Služby

Pro eliminaci duplicitního kódu jsou v práci využity takzvané služby, do kterých je často používaný kód rozdělen a díky tomu je případná editace tohoto kódu jednoduchá a není potřeba jej upravovat u všech výskytů.

3.2.3.1 DeviceService

Služba *DeviceService* obsahuje metody určené pro práci se zařízeními. Mezi takové metody je zařazeno generování konfiguračního souboru, ověření existence konfigurace a další metody spojené se zařízeními.

3.2.3.2 MailerService

Služba *MailerService* obsahuje metody sloužící pro odesílání specifických mailu odkudkoliv z aplikace.

3.2.3.3 NotificationService

V této službě se nachází metody využívané pro práci s oznámeními. Mezi takové patří načítaní počtu oznámení, které je využito pro výpis čísla vedle položky oznámení v menu nebo vytváření samotných oznámení.

3.2.3.4 SensorService

Služba *SensorService* obsahuje metody využívané při práci se senzory. Díky této službě lze ověřit aktivitu senzorů nebo celého zařízení na základě senzorů.

3.2.3.5 UserService

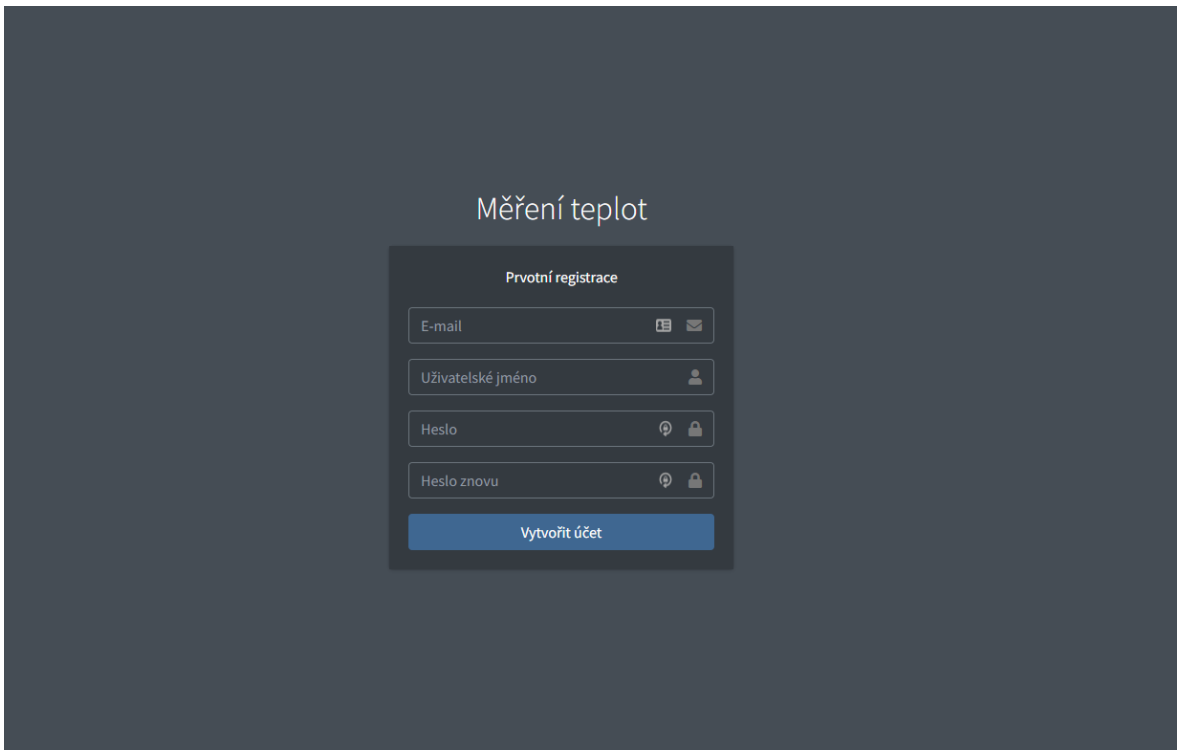
Služba *UserService* je určena pro metody využívající při práci s uživateli. V této službě se nachází metoda pro generování hesla.

3.2.4 Stránky

Kapitola stránky je zaměřena na jednotlivé stránky nacházející se v administraci.

3.2.4.1 Prvotní registrace

Tato stránka je zobrazena v případě, že se v databázi nenachází žádní registrovaní uživatelé. Účelem této stránky je registrace prvního účtu do systému s rolí administrátor, který umožní další práci s administrací.

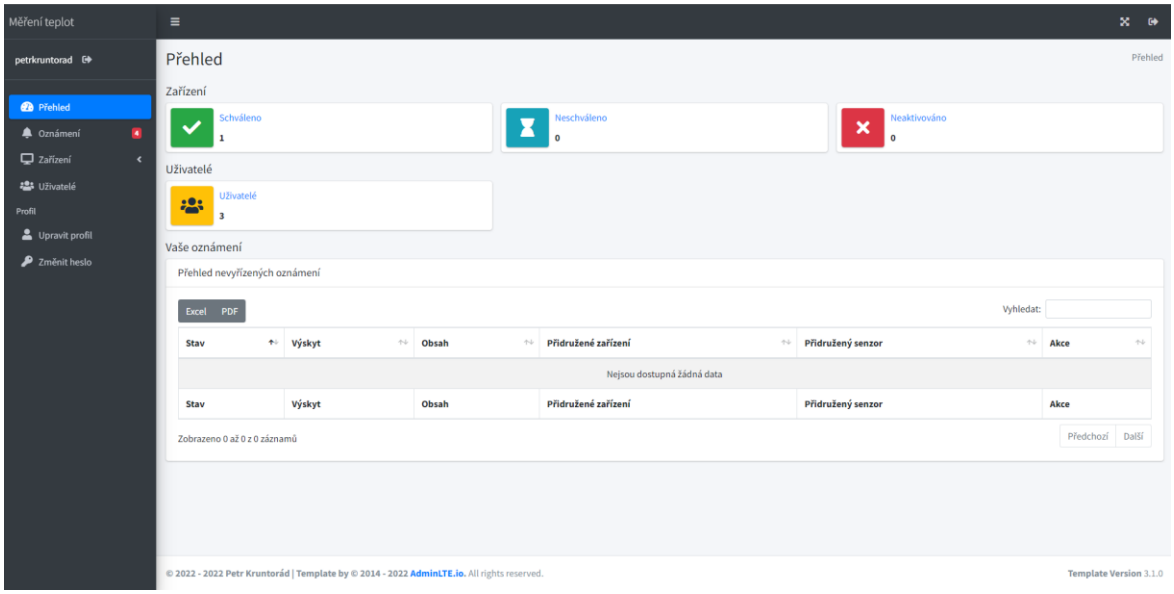


Obrázek 19 Webová administrace - Stránka prvotní registrace

3.2.4.2 Přehled

Přehled je využit jako úvodní stránka po přihlášení, na níž jsou zobrazeny informace o počtu zařízení v jednotlivých kategoriích (schváleno, neschváleno a neaktivováno), počet

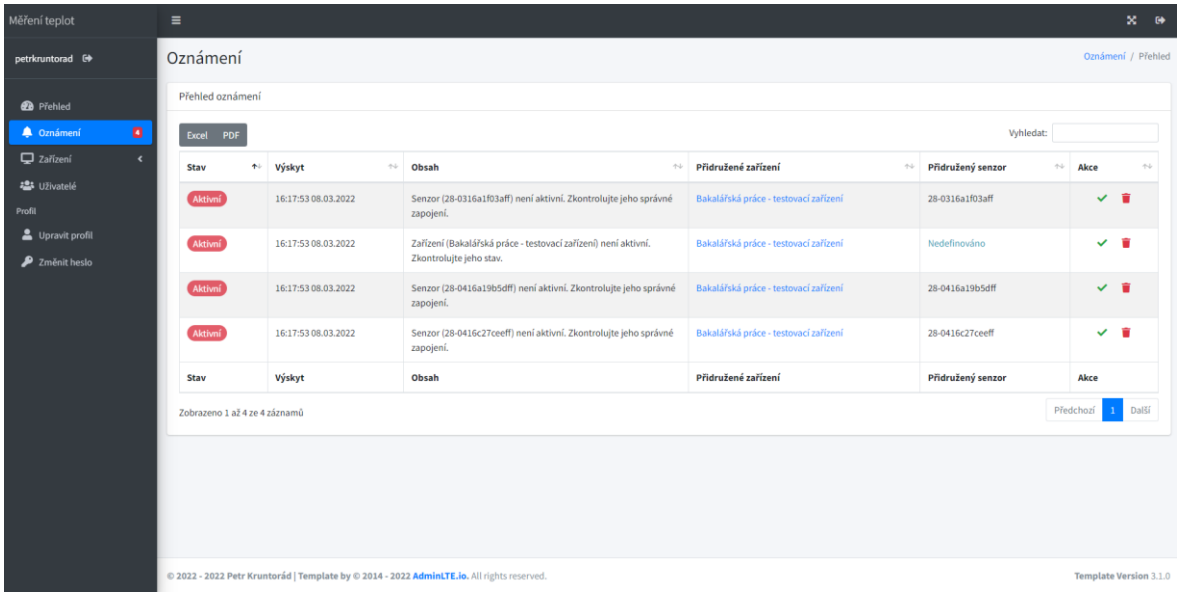
uživatelů v případě, že přihlášený uživatel má roli administrátor a přehled oznámení přiřazených k danému účtu s možností jejich potvrzení.



Obrázek 20 Webová administrace - Přehled

3.2.4.3 Oznámení

Stránka oznámení je přístupná aktuálnímu uživateli pouze v případě, že má v systému přidělenou roli administrátor. Je určena pro výpis, potvrzení, zrušení potvrzení a mazání oznámení, bez ohledu na to, zda je aktuálně přihlášený uživatel označen jako příjemce oznámení pro senzor vztahující se k jednotlivým notifikacím.



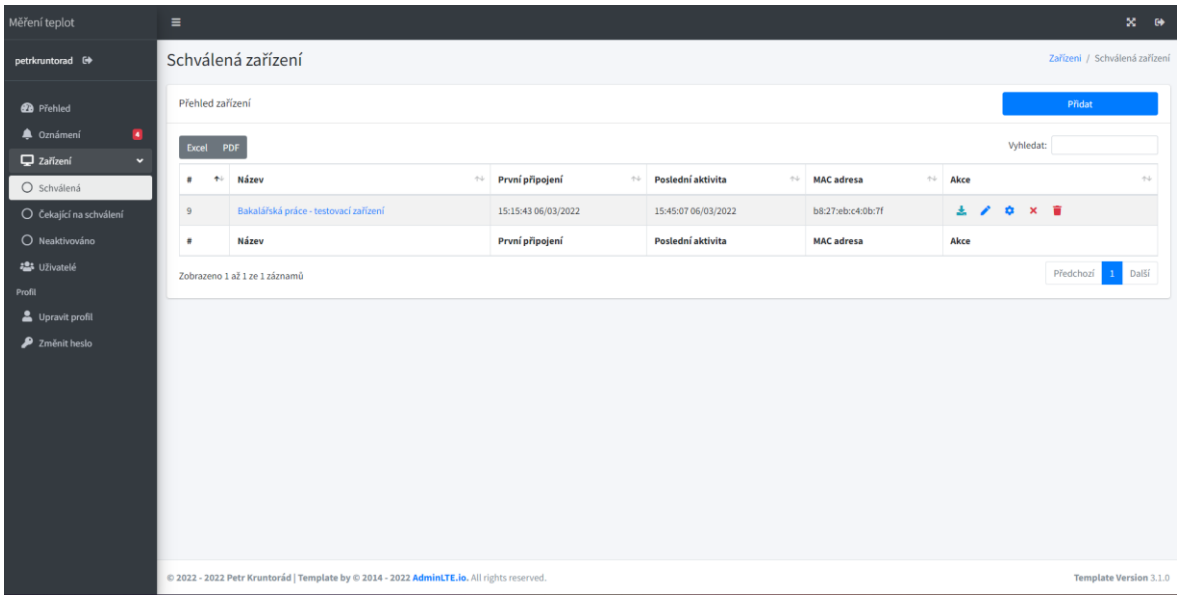
Obrázek 21 Webová administrace - Přehled Oznámení

3.2.4.4 Zařízení

3.2.4.4.1 Schválená zařízení

Na stránce schválená zařízení jsou vypsána zařízení, která úspěšně dokončila prvotní připojení a byla schválena uživatelem s rolí administrátor. Zařízení vypsaná na této stránce jsou schopna zapisovat data do systému, je možné je filtrovat dle sloupců a jejich export ve formátu *.xlsx* nebo *.pdf*. Po kliknutí na název zařízení v sloupci název je uživatel přesměrován na stránku detail.

V případě, že uživatel má roli administrátor, nachází se na této stránce tlačítko *přidat* určené pro přidávání zařízení a u každého zařízení jsou akce pro stažení konfiguračních souboru, editace, nastavení, zrušení potvrzení a smazání.

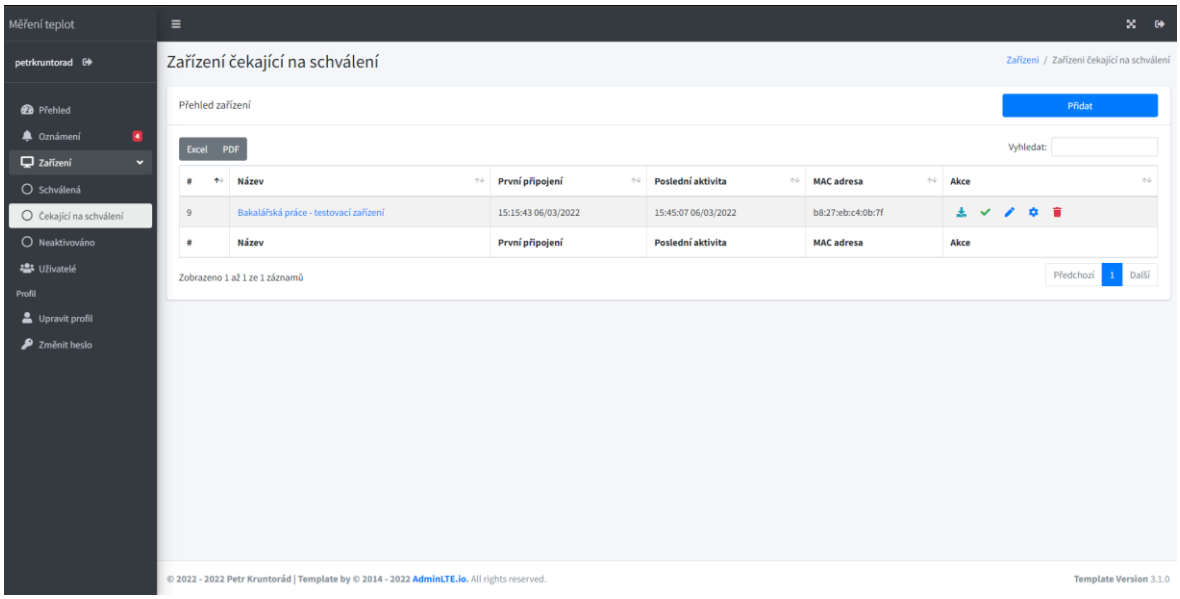


Obrázek 22 Webová administrace - Schválená zařízení

3.2.4.4.2 Zařízení čekající na schválení

Stránka zařízení čekající na schválení nebo také čekající na schválení je využita k výpisu zařízení, která úspěšně dokončila prvotní připojení do administrace, ale doposud nebyla schválena nebo jejich schválení bylo zrušeno. Stejně jako na stránce schválena zařízení je vypsaná zařízení možné filtrovat, exportovat nebo po kliknutí na název zařízení zobrazit detail.

V případě, že uživatel má roli administrátor je na této stránce zobrazeno tlačítko *přidat* určené pro přidávání zařízení a u každého zařízení jsou stejné akce jako na stránce schválená zařízení až na tlačítko zrušit potvrzení, které je nahrazeno tlačítkem potvrdit zařízení.

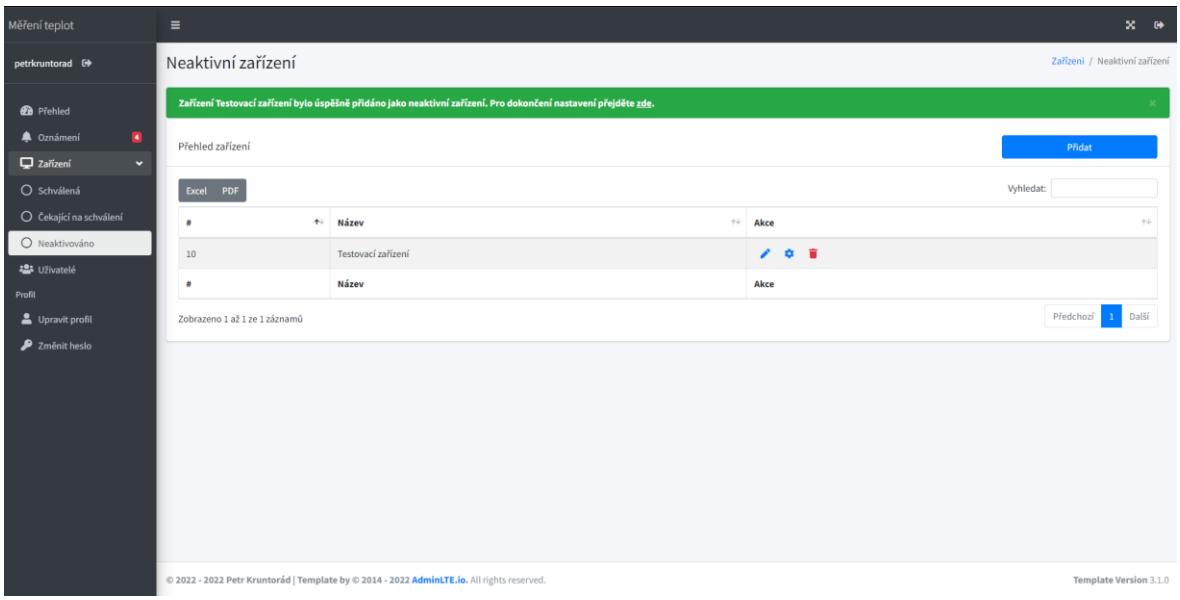


Obrázek 23 Webová administrace - Zařízení čekající na schválení

3.2.4.4.3 Neaktivní zařízení

Na této stránce jsou vypsaná zařízení, která doposud nedokončila prvotní připojení, a tedy je nelze schválit pro zápis do systému. Zařízení lze stejně jako na stránkách schválená zařízení a zařízení čekající na schválení filtrovat a exportovat nebo po kliknutí na název zařízení zobrazit detail.

Pokud má aktuálně přihlášený uživatel roli administrátor, je na stránce zobrazeno tlačítko *přidat* pro přidání nového zařízení a každého zařízení jsou akce pro editaci, nastavení, smazání a v případě, že bylo zařízení úspěšně nastaveno i tlačítko pro stažení konfiguračních souborů.

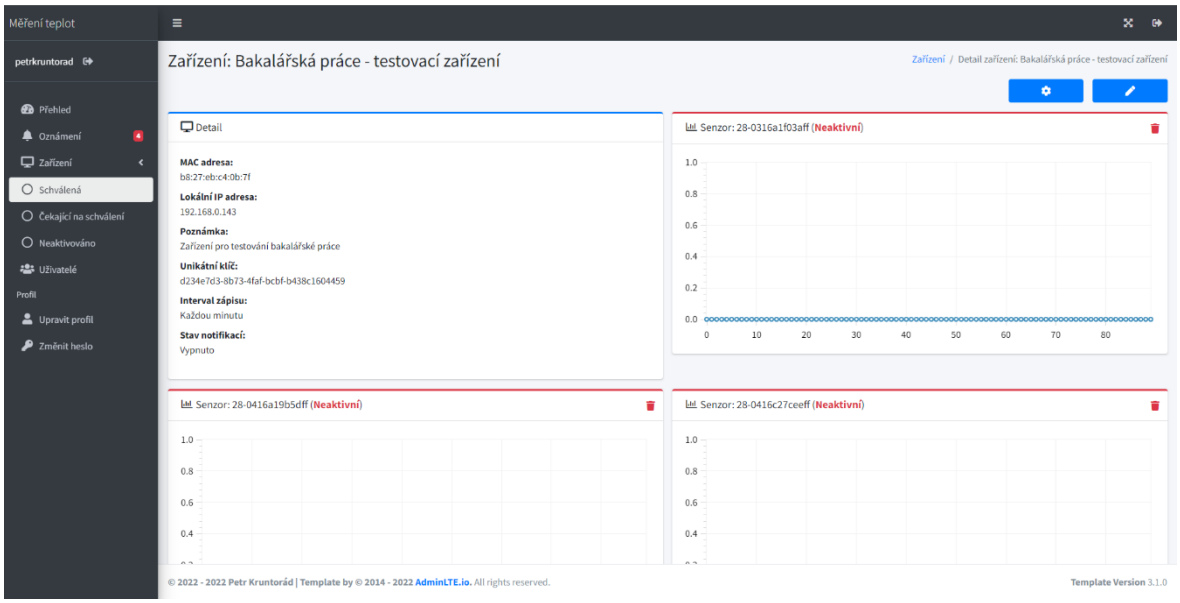


Obrázek 24 Webová administrace - Neaktivní zařízení

3.2.4.4.4 Detail zařízení

Stránka detail zařízení obsahuje detailní informace o zařízení, jakou jsou MAC adresa, lokální IP adresa, poznámka, interval zápisu, stav notifikací, příjemce notifikací a v případě, že aktuální uživatel má roli administrátor i unikátní klíč. V případě, že je uživatel administrátor je možné z této stránky zařízení editovat nebo nastavovat.

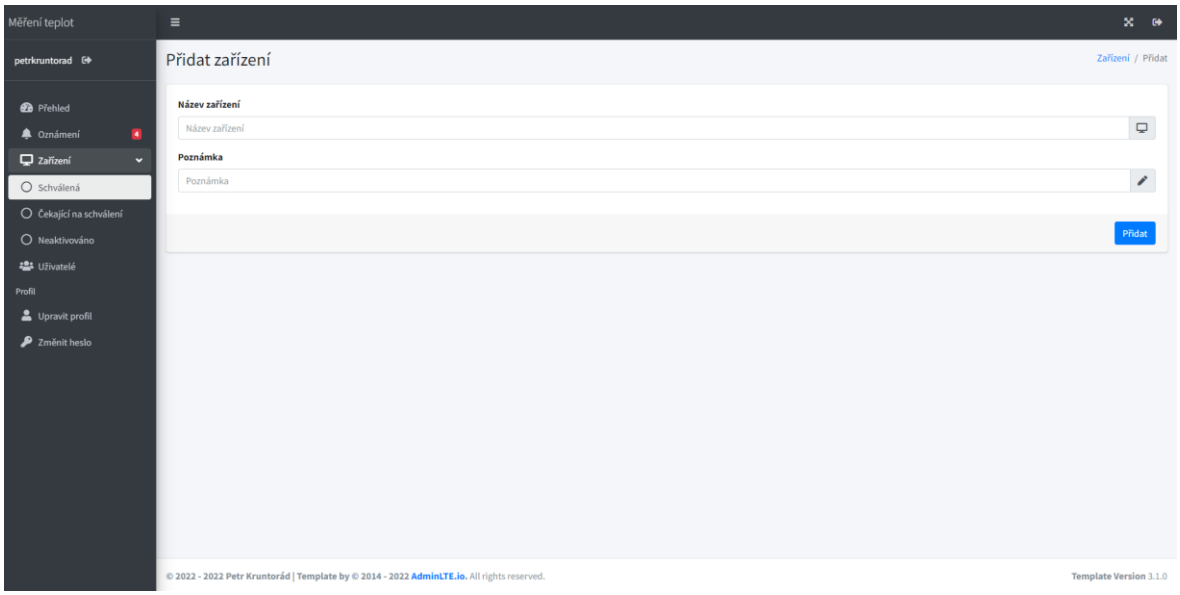
Dále jsou na této stránce zobrazeny živé grafy pro jednotlivé senzory přidružené k danému zařízení s možností odebrání jednotlivých senzorů (pouze administrátor). Tyto grafy vždy zobrazují 90 hodnot a rozestup těchto hodnot odpovídá intervalu zápisu. Aktualizace grafů je závislá na intervalu zápisu.



Obrázek 25 Webová administrace - Detail zařízení

3.2.4.4.5 Přidat zařízení

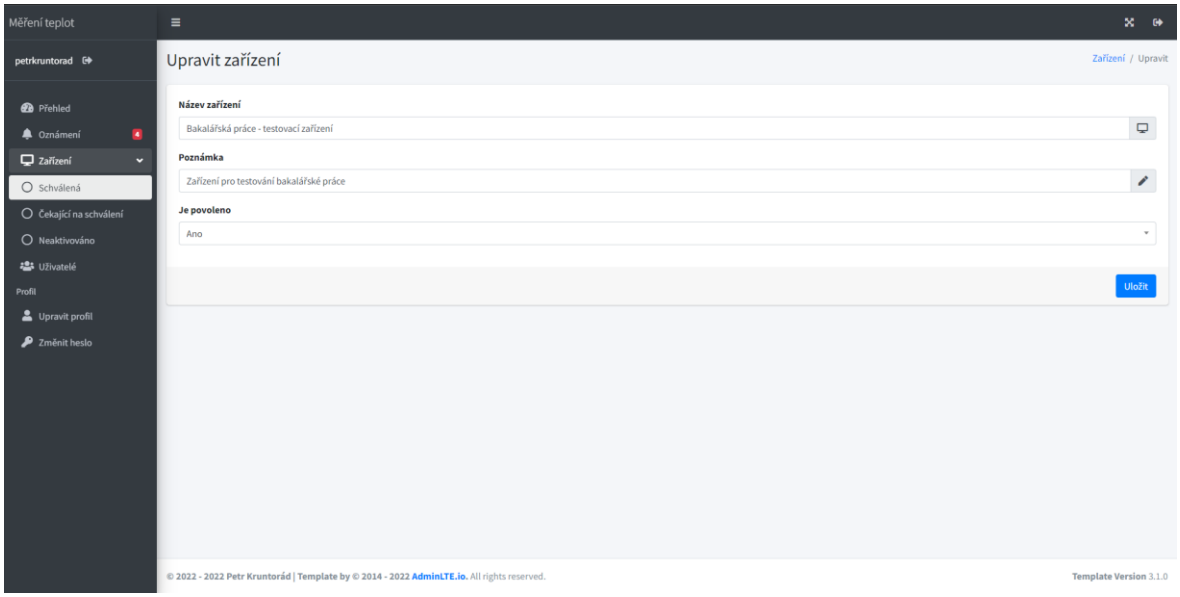
Na stránce přidat zařízení se nachází formulář pro přidání nových zařízení do systému. Tento formulář obsahuje pouze vstup pro název zařízení a poznámku. Po přidání nového zařízení je uživatel vyzván k dokončení nastavení tohoto zařízení. Tato stránka je přístupná pouze pro uživatele s rolí administrátor.



Obrázek 26 Webová administrace - Přidat zařízení

3.2.4.4.6 Upravit zařízení

Stránka upravit zařízení je vyhrazena pro změny základních informací o zařízení. Mezi tyto informace je zařazen název zařízení, poznámka a zda je zařízení povoleno. Tato stránka je přístupná pouze pro uživatele s rolí administrátor.

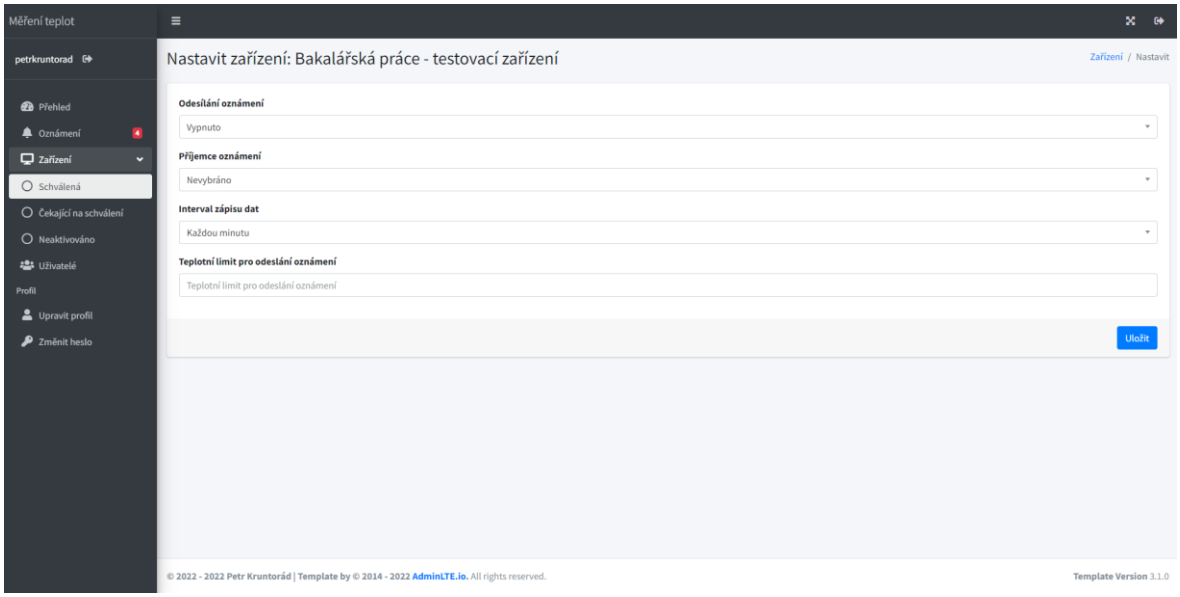


Obrázek 27 Webová administrace - Upravit zařízení

3.2.4.4.7 Nastavit zařízení

Na stránce Nastavit se nachází formulář se vstupy pro aktivaci odesílání oznámení, definování příjemce nastavení, změna intervalu zápisu dat a specifikování teplotního limitu pro odeslání oznámení.

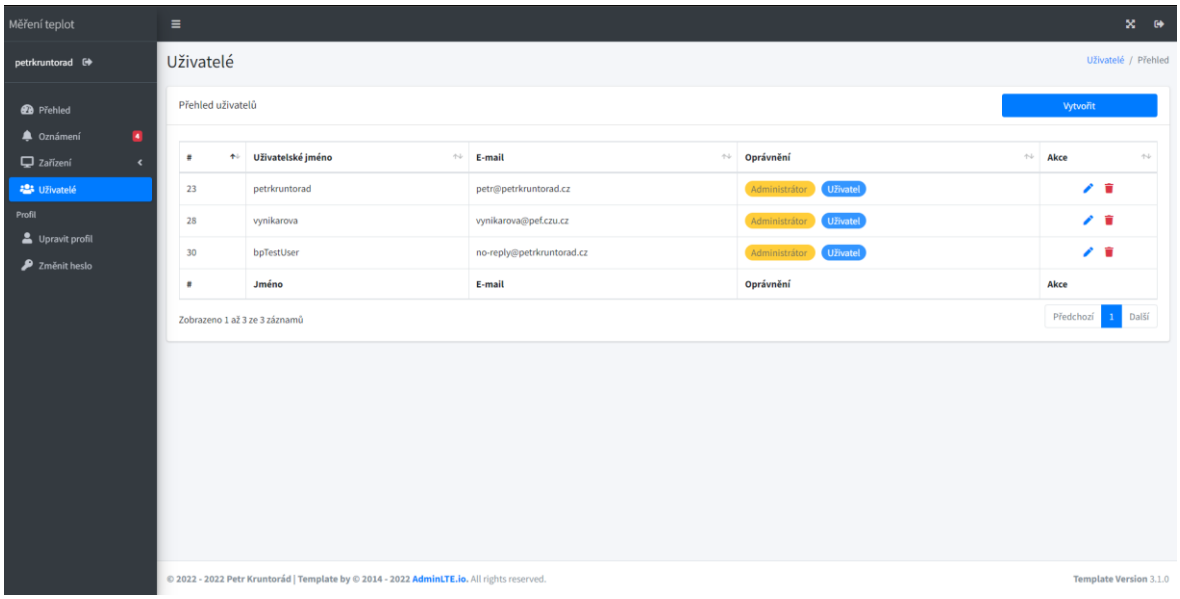
Data z tohoto formuláře jsou využita pro generování konfiguračního souboru a interní funkce webové administrace. Tato stránka je přístupná pouze pro uživatele s rolí administrátor.



Obrázek 28 Webová administrace - Nastavit zařízení

3.2.4.5 Uživatelé

Stránka uživatelé obsahuje tlačítko pro přidání uživatele a výpis všech uživatelů, kteří jsou přidání do systému. Aby mohl uživatel tuto stránku zobrazit je vyžadována role administrátor. Ve výpisu je zobrazeno uživatelské jméno, e-mail a oprávnění. Každého uživatele lze upravit nebo smazat.

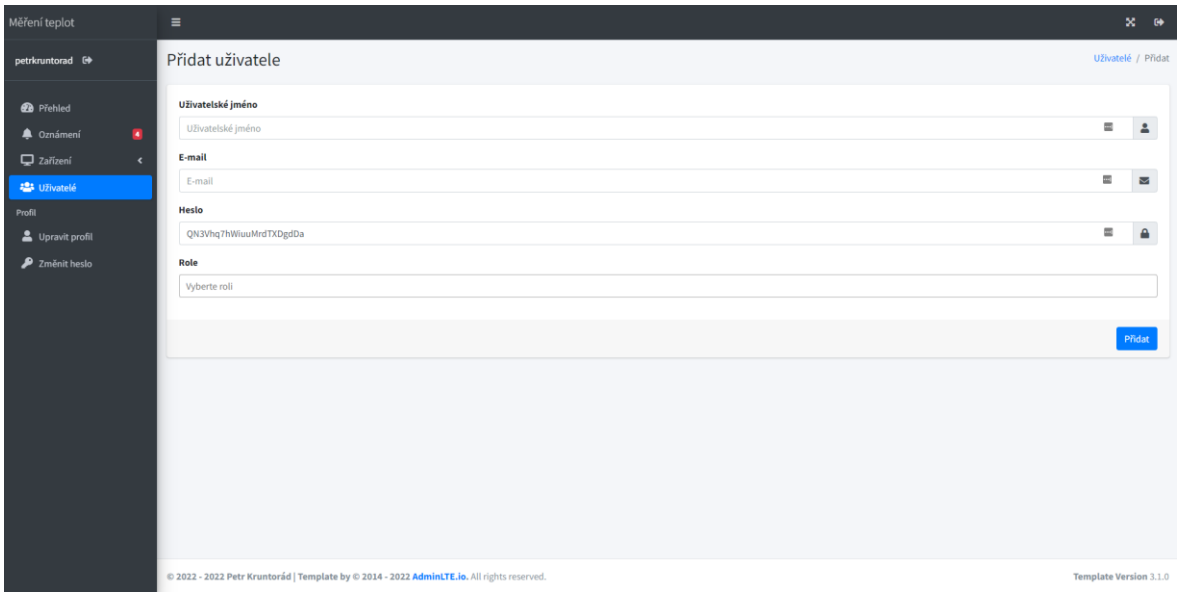


Obrázek 29 Webová administrace - Uživatelé

3.2.4.5.1 Přidat uživatele

Stránka přidat uživatele je přístupná pouze pro uživatele s rolí administrátor a obsahuje formulář se vstupy uživatelské jméno, e-mail, heslo s předem vygenerovaným heslem

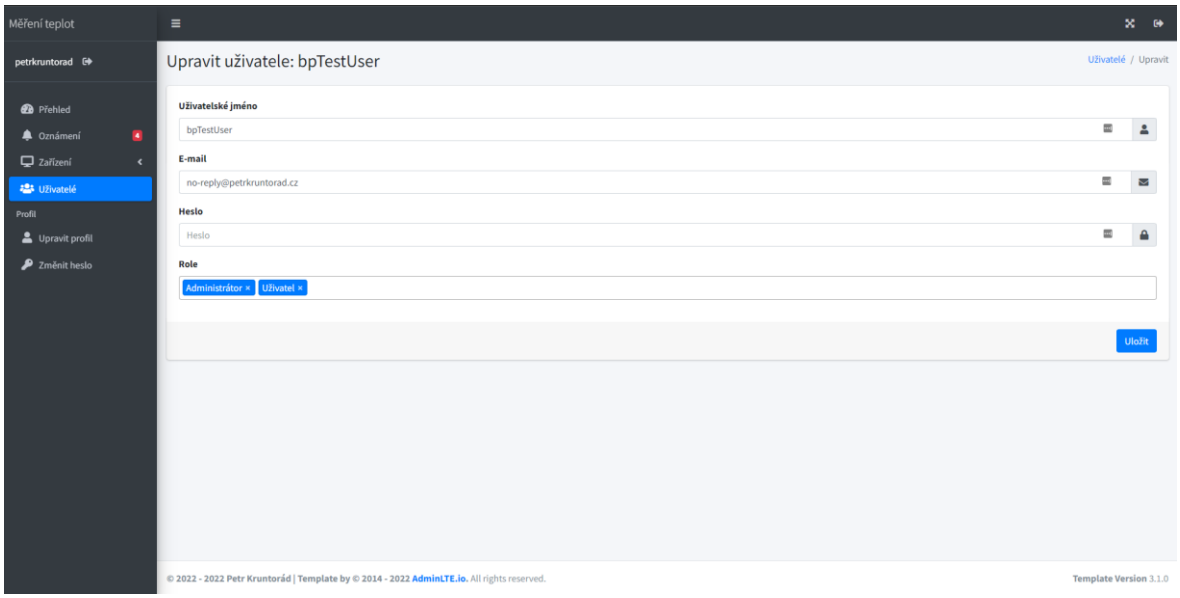
a možnost vybrání role. Po přidání uživatele se odešle na zadanou e-mailovou adresu e-mail s údaji potřebnými pro přihlášení.



Obrázek 30 Webová administrace - Přidat uživatele

3.2.4.5.2 Upravit uživatele

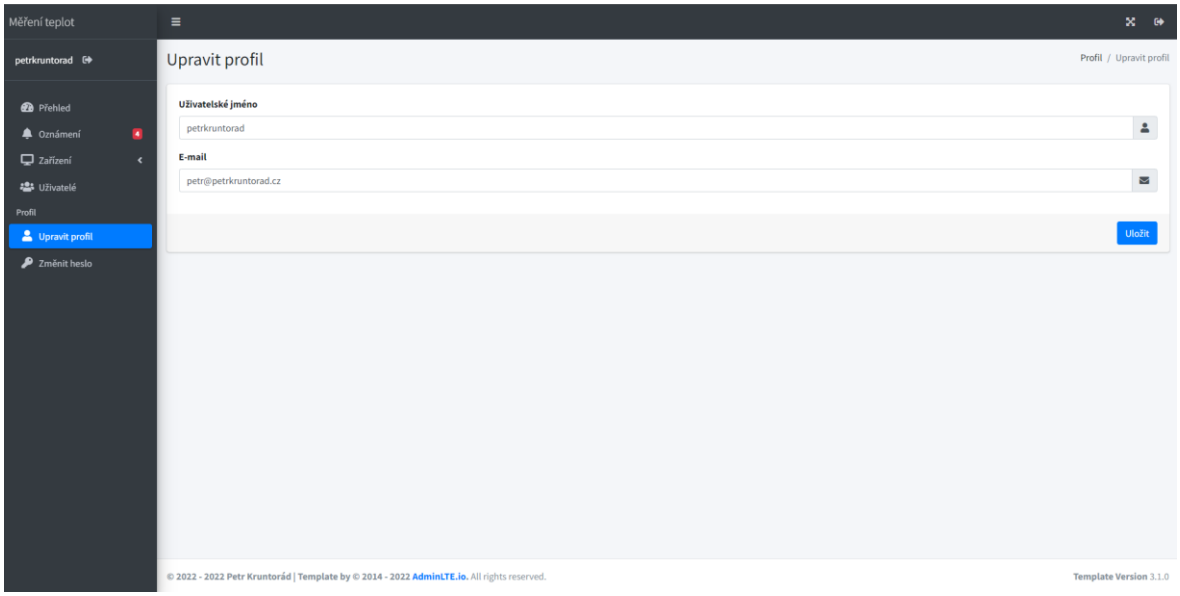
Tato stránka je přístupná pouze pro uživatele s rolí administrátor a je určena k editaci jednotlivých uživatelů. Formulář je shodný se stránkou přidat uživatele s tím rozdílem, že obsahuje vyplněné údaje pro editovaného uživatele. Vyplněné není pouze heslo, které se v případě, že není vyplněno nezmění. Pokud dojde k vyplnění vstupu pro heslo, dojde k jeho změně v databázi a odeslání e-mailu uživateli u kterého bylo heslo změněno s informacemi o novém heslu.



Obrázek 31 Webová administrace - Upravit uživatele

3.2.4.6 Upravit profil

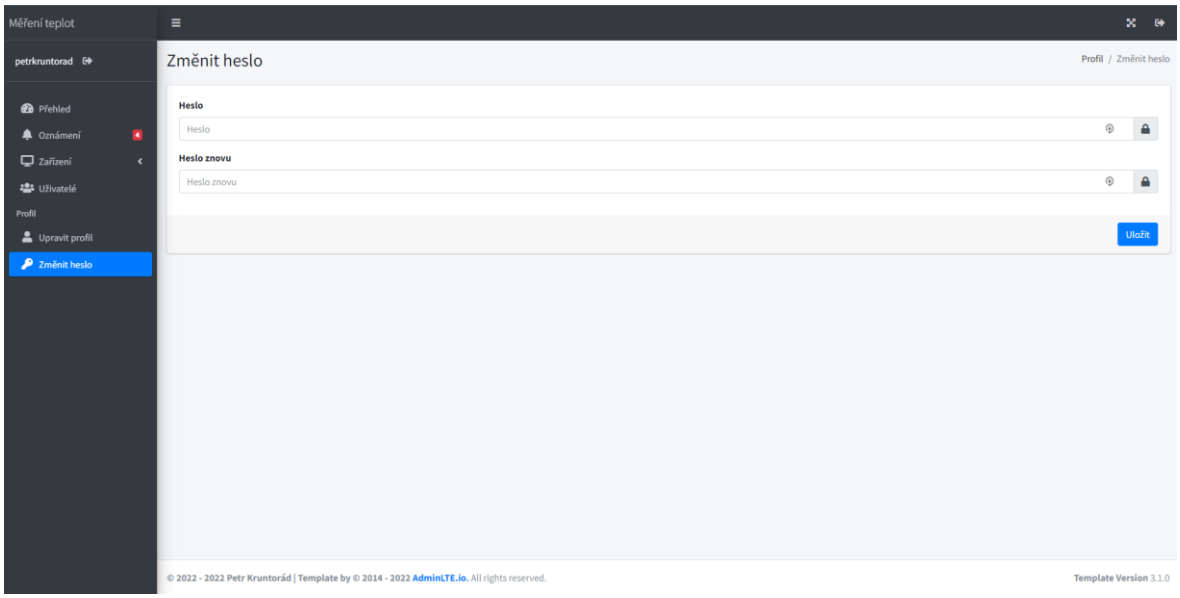
Na stránce Upravit profil je možné změnit uživatelské jméno a e-mail aktuálně přihlášeného uživatele. Tuto stránku není možné využít pro změnu údajů jiného uživatele nacházejícího se v databázi.



Obrázek 32 Webová administrace - Upravit profil

3.2.4.6.1 Změnit heslo

Stránka změnit heslo je určena pro změnu hesla aktuálně přihlášeného uživatele a nelze ji využít pro změnu jiného uživatele v databázi. Pro změnu hesla se musí shodovat zadané heslo ve vstupu Heslo a Heslo znovu.



Obrázek 33 Webová administrace - Změnit heslo

3.2.5 Vybrané části kódu

3.2.5.1 Metoda getUpdates

Metoda *getUpdates* se nachází v řadiči *DeviceController*. Tato metoda je využívána pro získání aktuálního konfiguračního souboru pro jednotlivá zařízení. K metodě lze přistupovat prostřednictvím URL adresy, která je složena z domény, na které je administrace hostována a cesty */device/get-updates*. Tuto adresu využívají všechna zařízení získání konfiguračních souborů.

Ve chvíli, kdy je zavolána tato metoda, dojde k ověření, zda byly v metodě POST poskytnuty hodnoty *uniqueHash* a *macAddress*. V případě, že nebyly poskytnuty tyto hodnoty je vrácena chybová hláška a soubor vygenerován není. Pokud jsou tyto údaje poskytnuty, systém se pokusí o nalezení zařízení v databázi se zadanými hodnotami a pokud je nalezeno dojde k vygenerování souboru *config.json* a jeho vrácení v odpovědi klientovi.

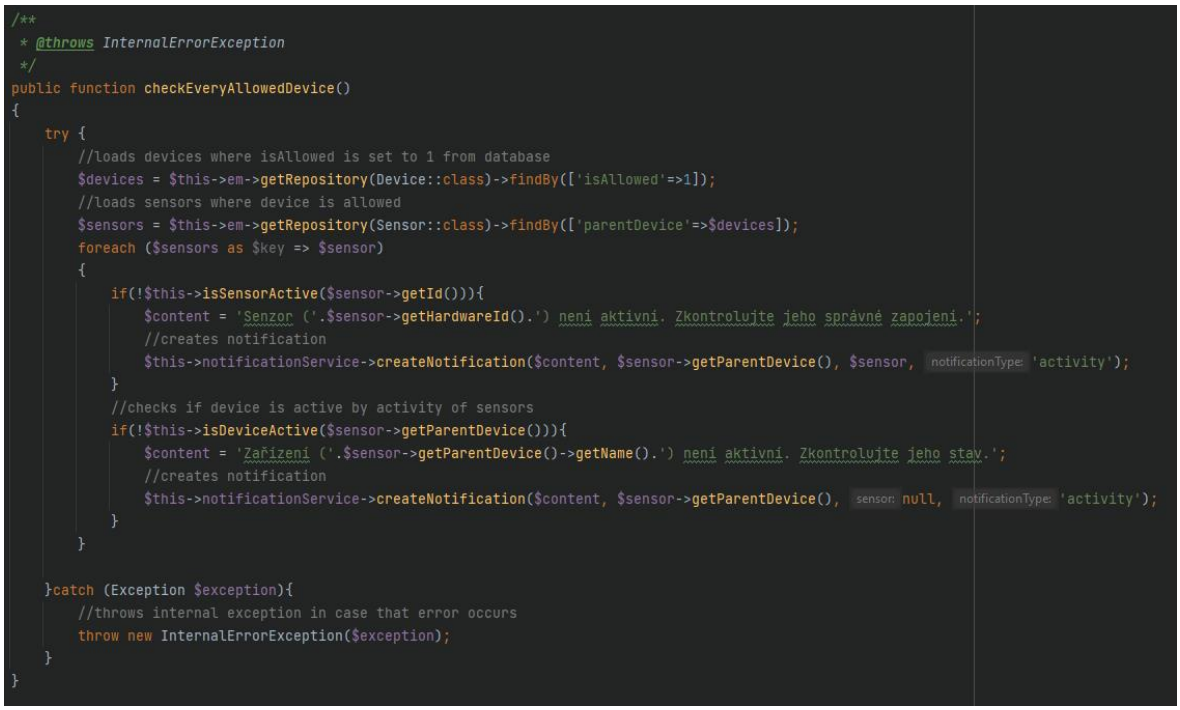
```php
/**
 * @Route ("/device/get-updates", name="device_getUpdates")
 * @throws Exception
 */
public function getUpdates()
{
    try {
        //checks if uniqueHash is defined
        if (!$_POST['uniqueHash'])
        {
            //throws an exception in case that unique hash is missing
            throw new Exception( message: "Unique hash is missing.");
        }
        //assigns uniqueHash to variable
        $uniqueHash = strval($_POST['uniqueHash']);

        //checks if macAddress is defined
        if (!$_POST['macAddress'])
        {
            //throws an exception in case that unique hash is missing
            throw new Exception( message: "MAC address is missing.");
        }
        //assigns macAddress to variable
        $macAddress = strval($_POST['macAddress']);

        //checks if device with specified uniqueHash is in database
        $device = $this->em->getRepository(Device::class)->findOneBy(array('uniqueHash'=>$uniqueHash,'macAddress'=>$macAddress));

        //if device is not defined throws exception
        if (!$device)
        {
            throw new Exception( message: "No such device with a specified unique hash.");
        }

        //inits the response with generated json
        $response = new Response($this->ds->generateConfigFile($device));

        //inits the disposition of file
        $disposition = HeaderUtils::makeDisposition(
             disposition: HeaderUtils::DISPOSITION_ATTACHMENT,
             filename: 'config.json'
        );
        //sets header
        $response->headers->set( key: 'Content-Disposition', $disposition);

        //returns response
        return $response;
    }
    catch (Exception $exception)
    {
        //throws an exception in case some error occurs
        throw new Exception( message: "Something went wrong: ".$exception);
    }
}
```

Obrázek 34 DeviceController – getUpdates

3.2.5.2 Metoda checkEveryAllowedDevice

Tato metoda se nachází ve službě *SensorService* a je určena ke kontrole všech povolených zařízení, zda jsou aktivní. V případě, že zařízení nebo senzor není aktivní, systém vytvoří oznámení a pokud je nastaven příjemce notifikací pro dané zařízení odešle na poskytnutou e-mailovou adresu zprávu o neaktivitě senzoru nebo zařízení.

Při zavolání této metody se nejprve načtou všechna povolená zařízení a ta se následně použijí k načtení senzorů k nim přiřazeným. Vrácené pole objektů se poté projde *for* cyklem a pokud je senzor nebo nadřazené zařízení neaktivní vygeneruje se oznámení.



```php
/**
 * @throws InternalErrorException
 */
public function checkEveryAllowedDevice()
{
    try {
        //loads devices where isAllowed is set to 1 from database
        $devices = $this->em->getRepository(Device::class)->findBy(['isAllowed'=>1]);
        //loads sensors where device is allowed
        $sensors = $this->em->getRepository(Sensor::class)->findBy(['parentDevice'=>$devices]);
        foreach ($sensors as $key => $sensor)
        {
            if(!$this->isSensorActive($sensor->getId())){
                $content = 'Senzor ('.$sensor->getHardwareId().') není aktivní. Zkontrolujte jeho správné zapojení.';
                //creates notification
                $this->notificationService->createNotification($content, $sensor->getParentDevice(), $sensor, notificationType: 'activity');
            }
            //checks if device is active by activity of sensors
            if(!$this->isDeviceActive($sensor->getParentDevice())){
                $content = 'Zařízení ('.$sensor->getParentDevice()->getName().') není aktivní. Zkontrolujte jeho stav.';
                //creates notification
                $this->notificationService->createNotification($content, $sensor->getParentDevice(), sensor: null, notificationType: 'activity');
            }
        }

    }catch (Exception $exception){
        //throws internal exception in case that error occurs
        throw new InternalErrorException($exception);
    }
}
```

Obrázek 35 SensorService - checkEveryAllowedDevice

3.2.5.3 Metoda generateConfigFile

Metoda *generateConfigFile* nacházející se ve službě *DeviceService* je v rámci aplikace využívána pro generování souboru *config.json* pro jednotlivá zařízení. Tento soubor je využíván skripty nacházejícími se na zařízení. Při zavolání této metody je vyžadován parametr *$device* datového typu odpovídajícího třídě *Device*.

V okamžik, kdy je metoda zavolána je vytvořeno prázdné pole *$fileContent* a parametr *$device* využit pro získání nastavení pro specifikované zařízení, která jsou uloženy v databázi. Tato nastavení jsou z databáze vrácena jako objekt. Ze získaného objektu jsou data přiřazena se specifickými klíči do vytvořeného pole. Pro klíče *writeUrl*, *updateUrl* a *touchUrl* jsou vygenerovány absolutní URL adresy na základě názvů jednotlivých metod v řadičích. Poté je pole je převedeno na JSON objekt a vráceno do původní metody odkud byla metoda *generateConfigFile* zavolána.

```php
/**
 * @throws Exception
 */
public function generateConfigFile(Device $device)
{
    $fileContent = [];
    try {
        //loads options for specified device
        $deviceOptions = $this->em->getRepository(DeviceOptions::class)->findOneBy(array('parentDevice'=>$device));

        //sets values with specified keys to array
        $fileContent['uniqueHash'] = $device->getUniqueHash();
        $fileContent['writeInterval'] = $deviceOptions->getWriteInterval();
        $fileContent['writeUrl'] = $this->router->generate( name: 'device_write_data', array(), referenceType: UrlGeneratorInterface::ABSOLUTE_URL);
        $fileContent['updateUrl'] = $this->router->generate( name: 'device_getUpdates', array(), referenceType: UrlGeneratorInterface::ABSOLUTE_URL);
        $fileContent['touchUrl'] = $this->router->generate( name: 'device_touch', array(), referenceType: UrlGeneratorInterface::ABSOLUTE_URL);

        //returns json with correct formatting
        return json_encode($fileContent, flags: JSON_UNESCAPED_UNICODE);
    } catch (Exception $exception) {
        //throws exception if error occurs
        throw new Exception($exception);
    }

}
```

Obrázek 36 DeviceService – generateConfigFile

3.2.5.4 Metoda getData

Metoda *getData* se nachází v řadiči *SensorDataController* a je využívána v administraci pro načítání dat do jednotlivých grafů. Při zavolání této metody jsou vyžadovány parametry *$device*, *$hardwareId* a *$push*. Hodnoty těchto parametrů jsou získávány z parametrů URL adresy. Parametr *$device* není v URL adrese přítomen přímo, ale data jsou získána za využití parametru id z databáze. Avšak toto načtení dat je podmíněno existencí patřičného záznamu v databázi. Parametr *$hardwareId* je textového typu a obsahuje sériové číslo senzoru. Poslední parametr *$push* obsahuje celočíselné hodnoty, jeho výchozí hodnota je 0. Tato hodnota je určena pro identifikaci toho, zda se jedná o první zavolání této metody, a tedy načtení všech 90 hodnot nebo aktualizaci grafu a načtení pouze 1 hodnoty.

```php
/**
 * @Route("/device/sensor/get-data/{id}/{hardwareId}/{push}", name="sensor_getData")
 * @IsGranted("ROLE_USER")
 */
public function getData(Device $device, string $hardwareId, $push=0){
    //inits empty array
```

Obrázek 37 metoda getData – parametry

S využitím poskytnutých parametrů jsou z databáze načtena nastavení pro zařízení a údaje o senzoru, které jsou použity pro nastavení sekund, rozestupu mezi záznamy ve vteřinách a celkový počet vteřin pro získání 90 hodnot. Poté už je jen nastavena hodnota proměnné *$currentTimestamp*, která obsahuje rozdíl vteřin mezi datem 1.1.1970 a aktuálním časem.

```php
//inits empty array
$data = [];

//loads options for device
$deviceOptions = $this->em->getRepository(DeviceOptions::class)->findOneBy(array('parentDevice'=>$device->getId()));

//loads sensor by hardware id and parent device
$sensor = $this->em->getRepository(Sensor::class)->findOneBy(array('hardwareId'=>$hardwareId, 'parentDevice'=>$device->getId()));

//sets default temperature
$defaultTemperature = 0;

//checks if write interval is set
if($this->ds->getWriteParametersForCron($deviceOptions->getWriteInterval()))
{
    //loads steps from device service
    $steps = $this->ds->getWriteParametersForCron($deviceOptions->getWriteInterval())['secondsSteps'];
}else{
    $steps = 60;
}
//sets seconds and total seconds by steps
$seconds = $steps;
$totalSeconds = $steps*90;
//current timestamp converted to seconds from 1.1.1970
$currentTimestamp = strtotime(date( format: "Y-m-d H:i:s"));
```

Obrázek 38 metoda getData - získání dat a nastavení proměnných

V případě, že parametr *$push* je roven 0, dojde k načtení posledních 90 hodnot pro specifikovaný senzor a vytvoření pole s 90 nulami. Toto pole je postupně procházeno *for* cyklem a jednotlivé hodnoty jsou v případě, že existují nahrazovány podle času zápisu naměřenými teplotami. Číslo aktuální iterace *$i* je využito pro vypočítání aktuální vteřinové pozice. To znamená, že při první iteraci, musí být čas od zápisu u vhodné hodnoty menší než interval zápisu jednotlivých hodnot nastavený pro zařízení nadřazené senzoru.

V každé iteraci tohoto *for* cyklu je spuštěn cyklus *foreach*, který prochází načtené hodnoty nacházející se v proměnné *$sensorData* a kontroluje, zda některá z hodnot nesplňuje podmínky pro nahrazení v poli na základě vteřin od zápisu. Pokud je taková hodnota nalezena, je přidána do pole *$temperaturesInit* a odstraněna z původního pole nacházejícího se v proměnné *$sensorData*. V okamžik, kdy je *for* cyklus dokončen, je pole *$temperaturesInit*

převráceno (to znamená, že první hodnota je poslední a naopak) a přidáno do pole *$data* pro index *temperatures*.

```php
//checks if current push is init when page is loading or not
if ($push == 0){
    //loads data from database by write time, limit 90
    $sensorData = $this->em->getRepository(SensorData::class)->getNumberOfTemperatures($sensor->getId(), limit: 90);

    //creates array filled with zeros to limit 90 records
    $temperaturesInit = array_fill( start_index: 0,  count: 90,  value: 0);

    //for each key in array
    for($i = 0; $i <= count($temperaturesInit)-1;)
    {
        //sets current position in seconds by steps and number of iteration
        $currentSecondsPosition = ($i+1) * $steps;
        //iterates every sensor data
        foreach ($sensorData as $key => $row) {
            //checks if write timestamp is not null
            if ($row->getWriteTimestamp() != NULL){
                //loads write timestamp for current iteration and converts it to second from 1.1.1970
                $writeTime = strtotime($row->getWriteTimestamp()->format('Y-m-d H:i:s'));
                //removes writetime from currentTimestamp
                $secondsFromWrite = $currentTimestamp - $writeTime;
                //checks if seconds from write are lower or equal to total seconds
                if ($secondsFromWrite <= $totalSeconds){
                    //checks if current seconds position is higher than seconds from write of current value
                    if($currentSecondsPosition >= $secondsFromWrite){
                        //sets value to array
                        $temperaturesInit[$i] = $row->getSensorData();
                        //remove used value from array
                        unset($sensorData[$key]);
                    }
                }
            }
        }
        $i++;
    }
    //assigns array with values to $data aray and reverse the order
    $data['temperatures'] = array_reverse($temperaturesInit);
}else{
```

Obrázek 39 metoda getData - načtení 90 hodnot

Pokud je hodnota v parametru *$push* větší než nula, je načtena z databáze pouze poslední hodnota a stejným principem jako hodnoty v předchozím *for* cyklu je zkontrolována, zda byla zapsána před intervalem zápisu. Tato hodnota může být například 60 vteřin, v případě že jsou data zapisována každou minutu. Pokud hodnota tuto podmínku nesplňuje je nastavena hodnota 0 v poli *$data* pro index *temperatures*.

```
}else{
    //loads single last value from sensordata
    $sensorData = $this->em->getRepository(SensorData::class)->findOneBy(array('parentSensor'=>$sensor->getId()),array('id'=>'DESC'));
    //checks if sensorData exists
    if($sensorData){
        //check if date is not null
        if ($sensorData->getWriteTimestamp() != NULL) {
            //loads write timestamp for current iteration and converts it to second from 1.1.1970
            $writeTime = strtotime($sensorData->getWriteTimestamp()->format('Y-m-d H:i:s'));
            //removes writetime from currentTimestamp
            $secondsFromWrite = $currentTimestamp - $writeTime;
            //check if the value is not older than specified amount of seconds
            if ($secondsFromWrite <= $seconds) {
                $defaultTemperature = $sensorData->getSensorData();
            }
        }
    }
    //assign the default value to $data array
    $data['temperatures'] = $defaultTemperature;
}
```

Obrázek 40 metoda getData - načtení 1 hodnoty

Jakmile jsou nastaveny všechny potřebné hodnoty, bez ohledu na to, zda se jedná o 90 hodnot nebo pouze 1, je vracená hodnota převedena na JSON objekt a vrácena tam, odkud byla metoda zavolána.

```
}
//response creation
//encodes array to json
$finalResponse = json_encode($data);
$response = new Response($finalResponse);
//sets header
$response->headers->set( key: 'Content-Type', values: 'application/json');
//return the response
return $response;
```

Obrázek 41 metoda getData - vrácení hodnot

3.2.6 Databáze

Tato kapitola se zaměřuje na strukturu tabulek v databázi a význam jednotlivých sloupců. Ze seznamu sloupců bude vynechán sloupec *id*, protože se nachází ve všech tabulkách a jeho účel je všude stejný, a to jednoznačný identifikátor každého záznamu. Díky tomuto jednoznačnému identifikátoru lze s daty manipulovat (měnit, mazat) nebo na ně odkazovat prostřednictvím relací z jiných tabulek.

3.2.6.1 Tabulka device

Tabulka *device* je využívána k ukládání základních dat o zařízeních, která jsou nezbytná pro základní funkce administrace. Tyto základní data jsou poté rozšířena relacemi na tabulky *device_notifications* a *device_options*.

Tabulka 3 tabulka device

Název sloupce	Datový typ	Popis
name	Varchar	Sloupec *name* obsahuje název zařízení
note	Tinyint	Obsahuje libovolnou poznámku u zařízení
is_allowed	Varchar	Obsahuje hodnoty 0 (zakázáno) nebo 1 (povoleno), je využit k určení, zda je zařízení povoleno nebo zakázáno
first_connection	Datetime	Obsahuje datum a čas, kdy se zařízení poprvé připojilo do systému
mac_address	Varchar	Obsahuje MAC adresu
unique_hash	Varchar	Obsahuje unikátní řetězec, který se používá pro ověření zařízení při komunikaci se serverem
local_ip_address	Varchar	Obsahuje lokální IP adresu pro ulehčení připojení k zařízení
last_connection	Datetime	Obsahuje datum a čas posledního připojení zařízení k serveru

3.2.6.2 Tabulka device_notifications

Tabulka *device_notifications* je využívána pro ukládání oznámení související se zařízeními nebo senzory.

Tabulka 4 tabulka device_notifications

Název sloupce	Datový typ	Popis
parent_device_id	Integer	Tento sloupec obsahuje cizí klíč z tabulky *device*, je určen pro identifikaci zařízení, ke kterému se oznámení vztahuje
sensor_id	Integer	Tento sloupec obsahuje cizí klíč z tabulky *sensor*, je určen pro identifikaci senzoru, ke kterému se oznámení vztahuje
notification_content	Varchar	Obsah oznámení
state	Tinyint	Stav oznámení, specifikuje, zda je oznámení aktivní (0) nebo vyřešené (1)
occurrence	Datetime	Obsahuje datum a čas výskytu oznámení
notificationType	Varchar	Obsahuje typ oznámení pro interní operace

3.2.6.3 Tabulka device_options

Tabulka *device_options* obsahuje doplňující nastavení k jednotlivým zařízením, která jsou potřebná instalaci nebo využití dalších funkcí v systému.

Tabulka 5 tabulka device_options

Název sloupce	Datový typ	Popis
parent_device_id	Integer	Tento sloupec obsahuje cizí klíč z tabulky *device*, je určen pro identifikaci zařízení, ke kterému se nastavení vztahuje
notifications_status	Tinyint	Tento sloupec specifikuje, zda jsou oznámení pro zařízení zapnuta (1) nebo vypnuta (0)
write_interval	Varchar	Obsahuje interval zápisu pro interní účely
notifications_target_user_id	Integer	Tento sloupec obsahuje cizí klíč z tabulky user, využívá se k identifikaci, na kterou e-mailovou adresu se mají oznámení zasílat
temperature_limit	Integer	Tento sloupec obsahuje limitní teplotu, při které se odešle oznámení z důvodu překročení teplotního limitu

3.2.6.4 Tabulka *sensor*

Tabulka *sensor* obsahuje informace o jednotlivých senzorech. Obsahuje hardwarové id senzoru a zařízení, ke kterému daný senzor přidělen.

Tabulka 6 tabulka sensor

Název sloupce	Datový typ	Popis
parent_device_id	Integer	Tento sloupec obsahuje cizí klíč z tabulky *device*, je určen pro identifikaci zařízení, ke kterému je senzor přidělen
hardware_id	Varchar	Obsahuje hardwarové id senzoru

3.2.6.5 Tabulka sensor_data

Tabulka *sensor_data* obsahuje získaná data z jednotlivých senzorů a všechny potřebné údaje k určení, kdy byla data zapsána a k jakému senzoru jsou přiřazena.

Tabulka 7 tabulka sensor_data

Název sloupce	Datový typ	Popis
parent_sensor_id	Integer	Tento sloupec obsahuje cizí klíč z tabulky sensor, je určen pro identifikaci senzoru, ke kterému patří data
sensor_data	Double	Obsahuje data přijatá ze senzoru
write_timestamp	Datetime	Obsahuje datum a čas, kdy byl záznam přidán

3.2.6.6 Tabulka *user*

Tabulka *user* je určena k ukládání uživatelů. Uživatelská data nacházející se v této tabulce jsou využívána pro přihlašování a odesílání oznámení.

Tabulka 8 tabulka user

Název sloupce	Datový typ	Datový typ
email	Varchar	Obsahuje e-mailovou adresu, která je využívá pro odesílání oznámení
roles	Longtext	Obsahuje role každého uživatele
password	Varchar	Obsahuje zašifrované uživatelské heslo
username	Varchar	Obsahuje uživatelské jméno využívající se pro přihlášení do administrace

3.2.7 Instalace

Pro přípravu administrace k nasazení na webový hosting je vyžadováno doinstalování potřebných závislostí. K tomu je vyžadováno PHP a správce závislostí Composer. Nejdříve je nutné stáhnout a nainstalovat PHP. To je možné na operačním systému Windows s využitím aplikace XAMPP nebo stažení potřebných instalačních souborů z URL adresy https://www.php.net/downloads.php [57]. Při stahování aplikace XAMPP nebo samotného PHP je nezbytné zkontrolovat verzi PHP, která pro optimální funkcionalitu musí být alespoň 7.2.5.

Po nainstalování PHP je vyžadováno stažení softwaru Composer ve verzi, která je specifikována v kapitole Softwarové požadavky. Instalační soubory jsou dostupné na adrese https://getcomposer.org/download/ [58].

Jakmile je dokončena instalace vyžadovaného softwaru je vyžadováno zadání příkazů prostřednictvím příkazového řádku v kořenové složce webové administrace. Tuto složku lze určit podle toho, že obsahuje například složky public, bin, config nebo templates.

Před spuštěním prvního příkazu musí být vyplněné správné hodnoty v souboru *.env*, který se také nachází v kořenové složce. V tomto souboru se musí nahradit hodnoty u řádků začínajících *MAILER_FROM, MAILER_DSN* a *DATABASE_URL*. Hodnoty, které jsou určeny k nahrazení jsou označené špičatými závorkami. Při vyplnění hodnoty u řádku *MAILER_FROM* je vyžadováno, aby e-mailová adresa, která bude vyplněna se nacházela na stejné doméně jako hodnota u řádku *MAILER_DSN*.

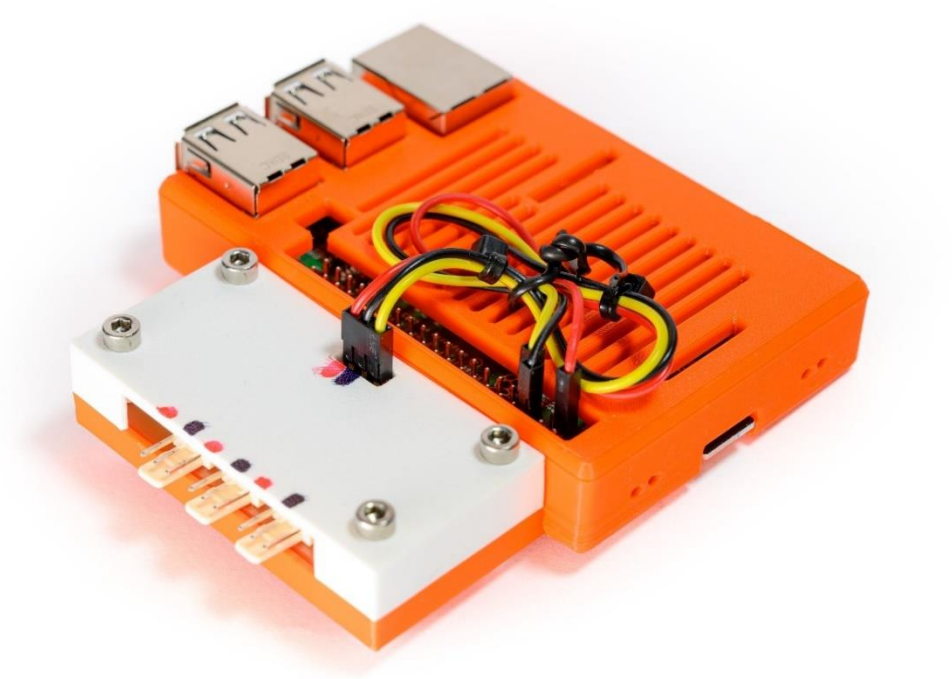
Prvním příkazem, který je vložen do příkazové řádky po dokončení instalace a konfigurace je příkaz "*composer install*". Tento příkaz doinstaluje všechny závislosti specifikované v souboru *composer.json*.

Dalšími příkazy jsou "*php bin/console d:d:c*" a "*php bin/console d:s:u --force*". Tyto příkazy jsou určeny pro vytvoření databáze na základě nastavených údajů v souboru *.env* a vytvoření tabulek v této databázi. Provedení těchto příkazů není povinné, protože databázi s tabulkami lze vytvořit i importováním souboru *temperature-measurement.sql* do databáze. Tento soubor se nachází v kořenové složce webové administrace.

3.3 Zařízení

V této kapitole jsou popsány skripty nacházející se na zařízení, zvolený hardware s následným způsobem zapojení a instalace veškerého potřebného softwaru na zařízení.

Pro ochranu zařízení a usnadnění manipulace byl pro zařízení pořízen, upraven a vytisknut box s využitím technologie 3D tisku z materiálu nazývaného PETG. Originální box byl pořízen na URL adrese https://www.thingiverse.com/thing:3503702.



Obrázek 42 Zkompletované zařízení

3.3.1 Hardware

Kapitola hardwarová část se zaměřuje na zvolený hardware využitý pro měření teplot a jeho zapojení.

3.3.1.1 Použité komponenty

Základní komponenty pro využití Raspberry Pi k měření teplot je počítač samotný, senzor DS18B20, 4.7k Ω rezistor a vhodné vodiče k propojení senzorů s deskou a rezistorem. [59]

V případě, že uživatel chce využít více senzoru s možností jejich jednoduché výměny je možné navrhnout vlastní desku tištěných spojů a tuto desku osadit konektory. Pro tento účel lze využít například konektorovou zásuvku 1x3pin s roztečí 2,54mm a konektorovou vidlici 1x3pin s roztečí 2,54mm.

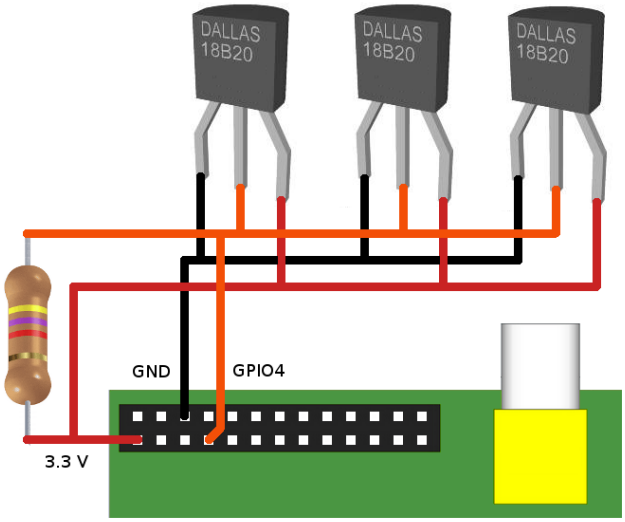
3.3.1.1.1 Teplotní senzor DS18B20

DS18B20 je digitální teplotní senzor schopný měřit v rozsahu od -55 do +125 °C s přesností na ± 0,5 °C v rozmezí od -10 do +85 °C. Senzor je schopný pracovat s napětím od 3 V do 5 V, je tedy vhodný pro použití s Raspberry Pi, který toto napájení využívá pro připojení rozšiřujících modulů a zařízení. [60]

Obrázek 43 Senzor DS18B20 [60]

3.3.1.2 Zapojení

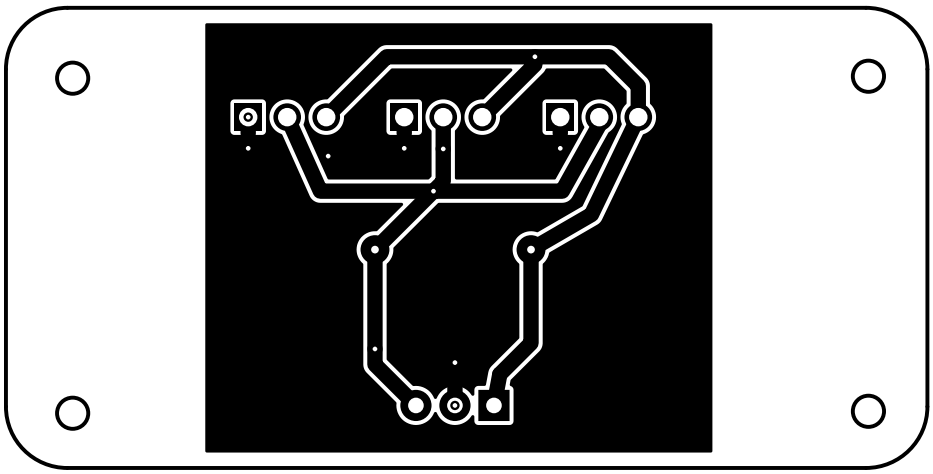
Připojení senzorů k Raspberry Pi by mělo vypadat jako na obrázku výše. Tedy mezi vodičem pro napájení a data je umístěn 4.7k Ω rezistor. Tyto vodiče se pak následně zapojí na pin pro 3.3 V, uzemnění a GPIO4.



Obrázek 44 zapojení senzorů [59]

Z důvodu usnadnění práce se senzory byla pro tento projekt navržena deska tištěných spojů se 3 konektory pro připojení senzorů, které nejsou limitem zařízení a v případě využití jiného způsobu zapojení jich lze připojit více. Do horní části s celkem 9 otvory patří konektory pro teplotní senzory, prostřední 2 otvory jsou určeny pro rezistor a spodní 3 otvory slouží pro připojení k Raspberry Pi.

U výše zmíněné desky tištených spojů je při zapojení nezbytné dbát na správné zapojení. V horní a spodní řadě konektoru je vždy jeden otvor, který je napojen přímo na vodivou oblast celé desky a je vyhrazen pro uzemnění. Zbylé 2 otvory jsou vyhrazeny pro napájení senzoru a data.



Obrázek 45 návrh desky tištěných spojů

3.3.2 Software

Tato kapitola je zaměřena na skripty nacházející se na zařízení s vybranými ukázkami kódu. Kód nenacházející se v této kapitole je zdokumentován v kódu za využití komentářů.

Software nacházející se na zařízení je složen ze 3 hlavních skriptů a 1 konfiguračního souboru, který je generován pro každé zařízení zvlášť. Názvy těchto hlavních skriptů jsou *main.py*, *functions.py* a *init.py*. Soubor obsahující konfiguraci je pojmenován *config.json*.

3.3.2.1 Main.py

Ve skriptu *main.py* se nachází základní funkcionalita, pod kterou patří načítání dat z konfiguračního souboru a je rozšířena o funkce pro zápis hodnot, načtení senzorů, aktualizaci konfiguračního souboru a aktualizace informací o zařízení na serveru.

```
# prepares config
configFile = open('config.json')
# loads config from file
configData = json.load(configFile)

# if values exists sets them to variable
if 'writeUrl' in configData.keys():
    writeUrl = configData['writeUrl']
else:
    raise ValueError("Write URL is missing in config file.")
if 'uniqueHash' in configData.keys():
    uniqueHash = configData['uniqueHash']
else:
    raise ValueError("Unique hash is missing in config file.")
if 'touchUrl' in configData.keys():
    touchUrl = configData['touchUrl']
else:
    raise ValueError("Touch URL is missing in config file.")
if 'updateUrl' in configData.keys():
    updateUrl = configData['updateUrl']
else:
    raise ValueError("Update URL is missing in config file.")

# updates info about device on api
touchServer(uniqueHash, touchUrl)
# updates config
updateConfig(uniqueHash, updateUrl)
# writes temperatures
saveTemperatures(sensors, writeUrl, uniqueHash)
```

Obrázek 46 Skript main.py – načtení proměnných, zavolání funkcí

3.3.2.2 Functions.py

Ve skriptu *function.py* se nachází nadefinované funkce potřebné pro správnou funkcionalitu zařízení. To umožňuje kód znovu použít nebo upravovat na jednom místě. Pro ukázku kódu byly vybrány následující funkce, které jsou pro správnou funkcionalitu nezbytné.

3.3.2.2.1 Funkce loadSensors

Funkce *loadSensors* se stará o načtení identifikátorů všech senzorů začínajících číslem 28, protože názvy u všech testovaných čidel začínaly právě tímto číslem.

Ve funkci se nejprve připraví prázdné pole, do kterého jsou později přidávány sériová čísla senzorů. Poté se pomocí metody *popen* a příkazu pro vylistování všech senzorů začínajících 28 načtou celá sériová čísla všech připojených senzorů, které se poté pomocí cyklu přidají do pole *sensors*, které je následně využito jako návratová hodnota pro tuto funkci.

```
# loads every connected sensor
def loadSensors():
    try:
        sensors = []
        # loads sensors with their names
        cmdLs = os.popen('ls /sys/bus/w1/devices/ | grep ^28').read()
        cmdLs = cmdLs.splitlines()
        for sensorId in cmdLs:
            # adds sensor ids to array
            sensors.append(sensorId)
        return sensors
    except RuntimeError as exception:
        logging.error("Internal error: " + str(exception))
```

Obrázek 47 skript functions.py - funkce loadSensors

3.3.2.2.2 Funkce saveTemperatures

Funkce *saveTemperatures* se stará o načtení naměřených teplot ze senzorů a jejich následné odeslání na server, kde dojde k vyhodnocení těchto dat a jejich následnému zapsání do databáze.

Funkce se volá s parametry *sensors*, *writeUrl* a *uniqueHash*. Parametr *sensors* je typu list a obsahuje dříve načtená sériová čísla senzorů. Parametry *writeUrl* a *uniqeHash* jsou textového typu, první zmíněný obsahuje cílovou URL adresu pro uložení teplot. Druhý zmíněný je využíván jako unikátní identifikátor pro zařízení, který ve spojení s MAC adresou slouží pro ověření zařízení při komunikaci se serverem.

Při zavolání této funkce se nejprve načte MAC adresa za využití funkce *getMainNetworkInterfaceMacAddress*. Poté jsou *for* cyklem procházena všechna poskytnutá

sériová čísla, která jsou za využití Python funkce *open* použita pro načtení dat z podsložky *w1_slave* u každého senzoru. Tyto data jsou následně procházena po řádcích, u kterých je kontrolováno, zda neobsahují textový řetězec "*t=*". Pokud se v řádku nachází zadaný řetězec, dojde k rozdělení řádku na základě mezer do pole a hledá se prvek začínající daným textovým řetězcem. Jakmile je nalezen, dojde k načtení hodnoty, která je za tímto řetězcem a hodnota je vydělena tisícem, aby došlo k jejím převedení na správnou jednotku.

Poté už je hodnota se sériovým číslem senzoru, proměnnou *uniqueHash* a *macAddress* odeslána na cílovou adresu.

```python
# saves temperatures to server
def saveTemperatures(sensors: list, writeUrl: str, uniqueHash: str):
    try:
        # gets MAC address
        macAddress = getMainNetworkInterfaceMacAddress()
        # go through every sensor in array
        for singleSensor in sensors:
            temperature = None
            # loads values from sensors
            with open("/sys/bus/w1/devices/" + singleSensor + "/w1_slave", "r") as file:
                for line in file:
                    if "t=" in line:
                        lineSplit = line.split(" ")
                        # iterates through values
                        for value in lineSplit:
                            if value.startswith("t="):
                                # gets correct value by dividing with 1000
                                temperature = float(value[2:]) / 1000
                                print(temperature)
            # if temperature is defined
            if temperature:
                session = requests.Session()
                # prepares data
                data = {'sensorId': singleSensor, 'rawSensorData': temperature, 'uniqueHash': uniqueHash,
                        'macAddress': macAddress}
                # sends data to api
                insert_request = session.post(url=writeUrl, data=data)
                # prints response
                print(insert_request.text)
            else:
                raise ValueError("Temperature is missing.")
    except RuntimeError as exception:
        logging.error("Internal error: " + str(exception))
```

Obrázek 48 skript functions.py - funkce saveTemperatures

3.3.2.2.3 Funkce setCronJob

Funkce *setCronJob* je určena k vytvoření úlohy Cron, ale v případě, že úloha již existuje, zavolá funkci sloužící pro aktualizaci. Tato funkce při svém zavolání vyžaduje parametr *interval*, který je textového formátu a měl by obsahovat zápis intervalu validní pro úlohy Cron (hodnota *writeInterval* z konfiguračního souboru).

Při zavolání funkce *setCronJob* dojde nejprve k ověření, zda úloha ještě neexistuje. Pokud je úloha již nastavena, dojde k zavolání funkce *updateCronJob*, která aktualizuje úlohu na základě parametru *interval*. V případě, že úloha neexistuje, dojde k vytvoření nové úlohy za využití proměnných *mainScriptPath* a *rowComment*, které jsou nadefinovány v horní části skriptu *functions.py*. První zmíněná proměnná obsahuje celou cestu ke skriptu *main.py* a druhá proměnná obsahuje komentář sloužící pro identifikaci úlohy touto funkcí.

```python
# creates new cron job
def setCronJob(interval: str):
    try:
        # checks if cronjob exists
        if checkIfCronJobExist(rowComment):
            # updates cronjob
            updateCronJob(interval)
        else:
            # creates new cronjob for main script
            job = cron.new(command='python ' + mainScriptPath, comment=rowComment)
            job.setall(interval)
            cron.write()
    except RuntimeError as exception:
        logging.error("Internal error: " + str(exception))
```

Obrázek 49 skript functions.py - funkce setCronJob

3.3.2.2.4 Funkce updateConfig

Funkce *updateConfig* je určena pro aktualizaci konfiguračního souboru ze serveru, kde se nachází webová administrace. Tato funkce se volá při každém spuštění skriptu *main.py*.

Tato funkce se volá s parametry *uniqueHash* a *updateUrl*. Parametry *uniqueHash* a *updateUrl* jsou textového typu. *UniqueHash* obsahuje unikátní řetězec načtený z konfiguračního souboru. Parametr *updateUrl* obsahuje adresu pro stažení nového konfiguračního souboru vygenerovaného pro dané zařízení.

Při zavolání této funkce se nejprve načte MAC adresa a společně s poskytnutým parametrem *uniqueHash* se pošle požadavek na server, kde se na základě těchto údajů vygeneruje konfigurace pro odpovídající zařízení. V případě úspěšného vygenerování se vrátí nová konfigurace ve formátu JSON. Touto novou konfigurací se následně přepíše aktuální obsah konfiguračního souboru *config.json* a opět se načte pro aktualizaci Cron úlohy v případě, že je *writeInterval* v konfiguraci definován.

```python
# gets new config from server
def updateConfig(uniqueHash: str, updateUrl: str):
    # gets MAC address
    macAddress = getMainNetworkInterfaceMacAddress()
    # prepares data
    postData = {'uniqueHash': uniqueHash, 'macAddress': macAddress}
    # sends data to api
    response = requests.post(updateUrl, data=postData)
    # checks if status code is 200
    if response.status_code == 200:
        # gets content
        data = response.content
        # prepares path to config
        pathToConfig = parentDirectory + '/config.json'
        # writes content
        with open(pathToConfig, 'wb') as s:
            s.write(data)

        # prepares new config
        configFile = open('config.json')
        # loads config from new file file
        configData = json.load(configFile)

        if 'writeInterval' in configData.keys():
            # updates cron
            updateCronJob(configData['writeInterval'])
        else:
            logging.error("Write interval is missing in config file.")
    else:
        # prints responses
        print(response.content)
        print(response.status_code)
```

Obrázek 50 skript functions.py - funkce updateConfig

3.3.2.2.5 Funkce touchServer

Funkce *touchServer* je využívána pro aktualizaci základních informací o zařízení na serveru. Funkce je zavolána při každém spuštění skriptu *main.py* s textovými parametry *uniqueHash* a *touchUrl*. Parametr *uniqueHash* obsahuje unikátní textový řetězec načtený z konfiguračního souboru. *TouchUrl* obsahuje URL adresu pro aktualizaci informací na serveru.

Při zavolání funkce dojde nejprve k načtení MAC adresy a IP adresy za využití funkcí *getMainNetworkInterfaceMacAddress* a *getIpAddress*. Poté jsou tyto hodnoty spolu parametrem *uniqueHash* odeslány na server, kde jsou následně zpracovány.

```python
# updates information about device on server
def touchServer(uniqueHash: str, touchUrl: str):
    try:
        # assigns values to variables
        macAddress = getMainNetworkInterfaceMacAddress()
        ipAddress = getIpAddress()

        session = requests.Session()
        # prepares data
        data = {'uniqueHash': uniqueHash, 'macAddress': macAddress, 'ipAddress': ipAddress}

        # sends data to api
        request = session.post(url=touchUrl, data=data)

        # returns status
        return request.text
    except RuntimeError as exception:
        logging.error("Internal error: " + str(exception))
```

Obrázek 51 skript functions.py - funkce touchServer

3.3.2.3 Init.py

Skript *init.py* se pouští pouze při inicializaci zařízení a je určeno pro jeho prvotní nastavení. Při spuštění tohoto skriptu dojde k nastavení úlohy Cron dle parametru *writeInterval* nacházejícího se v konfiguračním souboru.

```python
import json
import logging
from functions import setCronJob

logging.basicConfig(filename='log.txt', encoding='utf-8', level=logging.ERROR)

try:
    # variable init
    configFile = open('config.json')
    sensors = []
    writeInterval = None
    # loads config
    configData = json.load(configFile)

    # if write interval exist set value to variable
    if 'writeInterval' in configData.keys():
        writeInterval = configData['writeInterval']
    else:
        raise ValueError("Write interval is missing in config file.")

    # if write interval is efined sets cron job
    if writeInterval:
        setCronJob(writeInterval)

except FileNotFoundError as exception:
    logging.error("Internal error: " + str(exception))
```

Obrázek 52 skript init.py

3.3.3 Instalace

3.3.3.1 Operační systém

Pro nainstalování operačního systému Raspberry Pi OS je vyžadována paměťová microSD karta s kapacitou alespoň 8 gigabyte, čtečka paměťových karet pro microSD a pro snadnou instalaci software Raspberry Pi Imager. Tento software lze stáhnout na URL adrese https://www.raspberrypi.com/software/. [61]

Obrázek 53 Dialogové okno Raspberry Pi Imager [61]

Po nainstalování tohoto softwaru se otevře dialogové okno, které je určeno pro výběr operačního systému a uložiště pro jeho nainstalování. Ve výběru operačního systému se zvolí možnost s názvem Raspberry Pi OS (32-bit), ve výběru uložiště se zvolí připojena microSD karta. Po vybrání uložiště a operačního systému je nezbytné kliknout na tlačítko s ikonou ozubeného kola pro další nastavení.

V dalším nastavení je nezbytné povolit SSH s ověřením pomoci hesla a následně nastavit uživatelské jméno a heslo. Pro účely této práce je zvoleno uživatelské jméno *pi* a heslo *raspberry*. V případě potřeby je možné nastavit i připojení WiFi.

Po dokončení změn se musí kliknout na tlačítko *save* a v původním dialogovém okně na tlačítko *write*. Kliknutím na tlačítko *write* se spustí zápis na microSD kartu a po jeho dokončení je možné kartu odpojit a vložit do slotu na Raspberry Pi.

3.3.3.1.1 Konfigurace

Při první spuštění je doporučeno zařízení připojit k monitoru, klávesnici a myši pro jednodušší prvotní konfiguraci. Po spuštění je zobrazeno dialogové okno určené pro nastavení lokality, časové zóny, jazyku operačního systému a nastavení klávesnice. Po vyplnění libovolných hodnot a kliknutí na tlačítko *next* se zobrazí možnost pro změnu hesla, které není potřeba vyplňovat, pokud se bude systém využívat s výchozím heslem nebo heslem nastaveným při instalaci. Poté už se systém zeptá pouze na to, zda je rozhraní správně zobrazeno, ke které WiFi se má případně zařízení připojit a zda má zařízení zkontrolovat aktualizace.

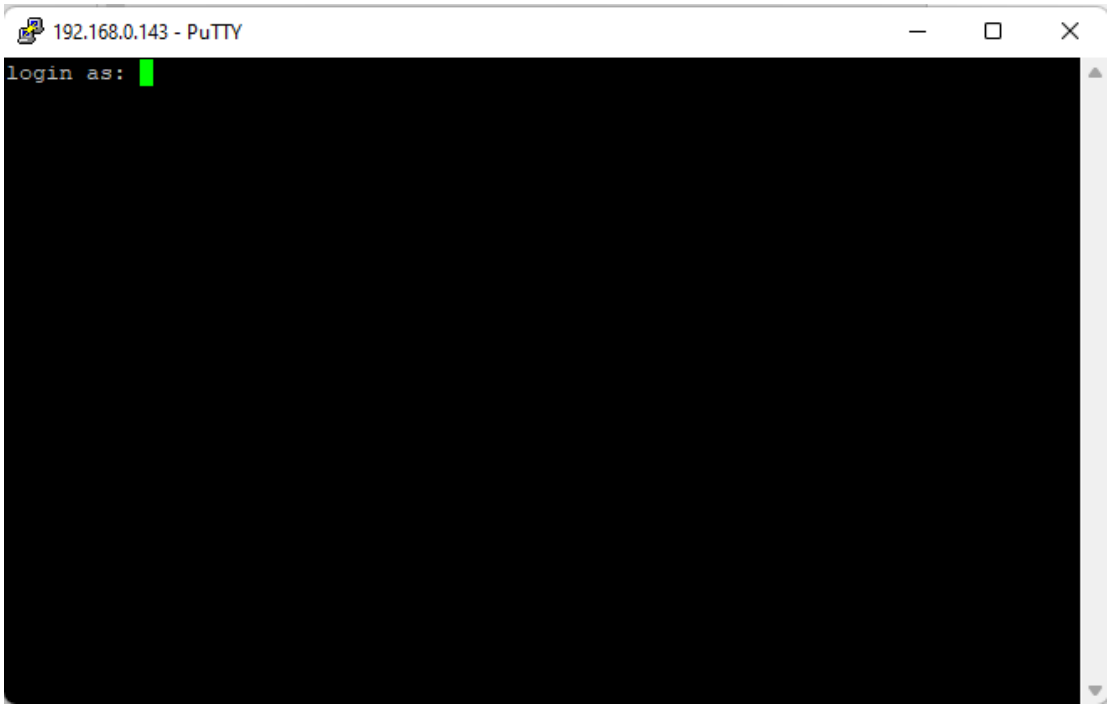
Po zkontrolování a případné aktualizaci softwaru je možné dokončit základní konfiguraci pro využití teplotních senzorů. Tato konfigurace se nachází v aplikačním menu (symbol maliny v liště), *Preferences*, *Raspberry Pi Configuration* a záložka *Interfaces*. V záložce Interfaces je nezbytné povolit 1-Wire rozhraní a restartovat zařízení. Po dokončení tohoto nastavení je již možné nainstalovat skripty a začít využívat zařízení pro měření teplot.

3.3.3.2 Konfigurační skripty

Pro získání konfiguračních skriptů je nejprve potřeba přidat a nastavit zařízení v administraci. Nově přidané zařízení se bude nacházet na stránce neaktivní zařízení nebo také neaktivováno v menu ve skupině zařízení. Zde se v případě dokončeného nastavení nachází u vybraného zařízení tlačítko pro jejich stažení. Skripty lze stáhnout formou souboru ZIP nebo příkazy umožňujícím stažení skriptů přímo na zařízení prostřednictvím příkazové řádky.

Pro využití příkazů určených ke stažení konfiguračních souborů je potřeba využít příkazovou řádku na zařízení, toho je možné docílit využitím monitoru, klávesnice a myši u fyzického zařízení nebo prostřednictvím softwaru umožňující připojení skrze SSH. K připojení skrze SSH lze například využít softwaru PuTTY.

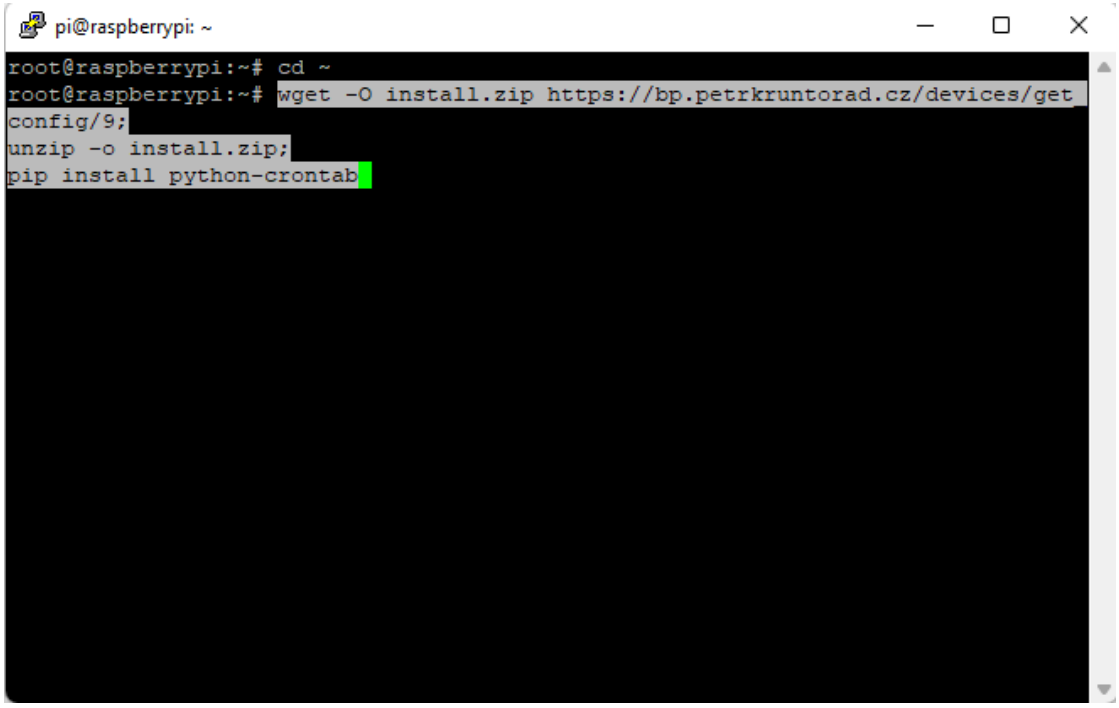
V případě připojení skrze software PuTTY musí uživatel znát IP adresu zařízení, tu je možné zjistit například v routeru nebo na zařízení a přihlašovací údaje, které jsou výchozí nebo byly nastaveny při instalaci. Pro účely této práce je využito přihlašovací jméno *pi* a heslo *raspberry*.



Obrázek 54 Přihlašovací obrazovka pro SSH

Po přihlášení je doporučeno se přepnout na uživatele *root*, toho lze docílit příkazem "*sudo su*" a následně zvolit adresář, do kterého mají být skripty staženy a odkud budou spouštěny.

Jakmile dojde k vybrání požadovaného adresáře, je nezbytné do příkazové řádky vložit příkazy zkopírované z webové administrace. Po úspěšném dokončení těchto příkazů je nezbytné pro aktivaci automatického odesílání dat spustit ve stejné složce jako se nachází skripty příkaz "*python init.py*", tím dojde aktivaci a zařízení začne ve stanovených intervalech odesílat data.



Obrázek 55 Změna adresáře, vložení příkazů

Posledním krokem pro zobrazení dat odesílaných je schválení tohoto zařízení v administraci, pokud zařízení není potvrzeno, není mu umožněn zápis dat do databáze.

4 Závěr

Cílem této závěrečné práce bylo vytvoření webového rozhraní, skriptů sloužících pro získávání dat na zařízení a kompletace zařízení sloužícího pro měření teplot.

V teoretické části byly vymezeny základní pojmy, specifikovány použité technologie, analyzovány existující i alternativní řešení a vytyčeny funkční a nefunkční požadavky. Z analyzovaných řešení bylo zjištěno, že existující řešení jsou omezena buď počtem připojitelných senzorů, cenově, uzavřenosti nebo dostupnými funkcemi.

V praktické části bylo vypracováno řešení pro získávání, odesílání a výpis teplot s využitím počítače Raspberry Pi, které k měření teplot využívá senzory DS18B20, které mají rozsah měření od -55 do +125 °C s rezistorem 4.7k Ω a skripty napsané v programovacím jazyce Python. Vytvořené skripty odesílají v pravidelném intervalu naměřené hodnoty a data s využitím Cron úloh. Pro snadné připojení senzorů byla navržena a vytvořena deska tištěných spojů, která umožňuje připojit až 3 senzory, které ale nejsou maximálním připojitelným počtem senzorů k zařízení.

Pro tvorbu webového rozhraní byla využita šablona AdminLTE a PHP framework Symfony. Oproti analyzovaným řešením je vytvořené řešení cenově dostupnější a upravitelnější na míru vzhledem k využitým technologiím.

5 Seznam použitých zdrojů

1. CHRISTENSSON, Per. Webová aplikace. *TechLib* [online]. Sharpened Productions, 2018 [cit. 2021-10-01]. Dostupné z: https://tech-lib.eu/definition/web_application.html
2. Webová aplikace (Web Application). *ManagementMania.com* [online]. Wilmington, 2018 [cit. 2021-10-01]. Dostupné z: https://managementmania.com/cs/webova-aplikace-web-application
3. ČÁPKA, David. Lekce 1 - Úvod do PHP a webových aplikací. *Itnetwork.cz* [online]. Praha, 2013 [cit. 2021-10-01]. Dostupné z: https://www.itnetwork.cz/php/zaklady/php-tutorial-uvod-do-webovych-aplikaci
4. NAVRÁTIL, Leoš a Jozef ROSINA. Medicínská biofyzika. 2., zcela přepracované a doplněné vydání. Praha: Grada Publishing, 2019. ISBN 978-80-271-2700-9.
5. VODIČKOVÁ, Kateřina. Jaké existují teplotní stupnice?. METEOPRESS [online]. 2019 [cit. 2022-02-09]. Dostupné z: https://www.meteopress.cz/vysvetleni/jake-existuji-teplotni-stupnice/
6. WIKISKRIPT, Přispěvatelé. Měření teploty. WikiSkripta [online]. 1. lékařská fakulta, Univerzita Karlova v Praze, 2020 [cit. 2021-10-28]. Dostupné z: https://www.wikiskripta.eu/index.php?title=M%C4%9B%C5%99en%C3%AD_teploty&oldid=440316
7. How Bimetallic Thermometers work. Tameson [online]. Tameson, 2022 [cit. 2022-02-26]. Dostupné z: https://tameson.com/bimetallic-thermometer.html
8. Odporové snímače, termistory a polovodičové snímače. New technologies research centre: University of West Bohemia [online]. c2021[cit. 2021-10-28]. Dostupné z: https://ttp.zcu.cz/en/laboratories/thermal-fields/wiki/contact-measurement/sensors
9. DÍTĚ, Ivan. Bezkontaktní infračervené teploměry. Elektro: časopis pro elektrotechniku [online]. FCC Public, 2021 [cit. 2021-10-28]. Dostupné z: http://www.odbornecasopisy.cz/elektro/casopis/tema/bezkontaktni-infracervene-teplomery--14006
10. BARÁŠEK, Jan. Proč a jak používat frameworky a knihovny. *Český PHP manuál* [online]. 2020 [cit. 2021-10-09]. Dostupné z: https://php.baraja.cz/proc-pouzivat-frameworky
11. What Is a Database?. Oracle [online]. Oracle, 2022 [cit. 2022-02-13]. Dostupné z: https://www.oracle.com/cz/database/what-is-database/
12. Types of Databases. Learntek [online]. Learntek, 2021 [cit. 2022-03-14]. Dostupné z: https://www.learntek.org/blog/types-of-databases/
13. ČAPKA, David. Lekce 1 - MySQL krok za krokem: Úvod do MySQL a příprava prostředí. Itnetwork.cz [online]. Praha, 2021 [cit. 2022-02-12]. Dostupné z: https://www.itnetwork.cz/mysql/mysql-tutorial-uvod-a-priprava-prostredi
14. CASTRO, Kristi. Object-relational Data Model. Tutorialspoint [online]. 2018 [cit. 2022-02-13]. Dostupné z: https://www.tutorialspoint.com/Object-relational-Data-Model
15. Object-Relational Database (ORD). Technopedia [online]. Technopedia, 2021 [cit. 2022-02-13]. Dostupné z: https://www.techopedia.com/definition/8714/object-relational-database-ord
16. What is Database Management Systems (DBMS)?. AppDynamics [online]. AppDynamics, 2021 [cit. 2022-03-14]. Dostupné z: https://www.appdynamics.com/topics/database-management-systems
17. KLÍMA, Tomáš. Architektura MVC. *Jak psát PHP?* [online]. 2017 [cit. 2021-10-24]. Dostupné z: https://jakpsatphp.cz/MVC/

18. ČÁPKA, David. MVC architektura. *Itnetwork.cz* [online]. Praha, 2020 [cit. 2021-10-24]. Dostupné z: https://www.itnetwork.cz/navrh/mvc-architektura-navrhovy-vzor
19. VAVREČKOVÁ, Šárka. Počítačová síť a internet. Opava: Filozoficko-přírodovědecká fakulta v Opavě, 2017. ISBN 978-80-7510-245-4.
20. HONEK, Lukáš. HTML: Kapesní přehled. Brno: Computer Press, 2004. ISBN 80-722-6958-5.
21. HTML History. *W3schools* [online]. 2021 [cit. 2021-10-24]. Dostupné z: https://www.w3schools.in/html-tutorial/history/
22. THOMAS, Jeremy. A valid HTML document: Some boilerplate. MarkSheet [online]. 2021 [cit. 2022-02-26]. Dostupné z: https://marksheet.io/html-valid-document.html
23. HTML <head> Tag. *W3schools* [online]. Refsnes Data, 2021 [cit. 2021-10-24]. Dostupné z: https://www.w3schools.com/tags/tag_head.asp
24. HTML <html> Tag. *W3schools* [online]. Refsnes Data, 2021 [cit. 2021-10-24]. Dostupné z: https://www.w3schools.com/tags/tag_html.asp
25. ŠTRÁFELDA, Jan. HTML atribut. *Štráfelda.cz* [online]. 2021 [cit. 2021-10-24]. Dostupné z: https://www.strafelda.cz/html-atribut
26. HTML Attributes. *W3schools* [online]. Refsnes Data, 2021 [cit. 2021-10-24]. Dostupné z: https://www.w3schools.com/html/html_attributes.asp
27. Co je to PHP a k čemu mi bude dobré? CoreIT [online]. CoreIT, 2021 [cit. 2021-10-24]. Dostupné z: https://www.coreit.cz/php/
28. CHRSTENSSON, Per. PHP. TechTerms.com [online]. Sharpened Productions, 2006 [cit. 2021-10-24]. Dostupné z: https://techterms.com/definition/php
29. BURETS, Alex. PHP vs Javascript: a Short Comparison. Scand [online]. Varšava: Scand, 2019 [cit. 2022-03-14]. Dostupné z: https://scand.com/company/blog/php-vs-javascript-difference-between/
30. VRÁNA, Jakub. 1001 tipů a triků pro PHP. Brno: Computer Press, 2010. ISBN 978-80-251-2940-1.
31. SUMMERFIELD, Mark. Programming in Python 3: A Complete Introduction to the Python Language. 2. vydání. New Jersey: Addison-Wesley Professional, 2010. ISBN 978-0-321-68056-3
32. STURTZ, John. Variables in Python. Real Python [online]. DevCademy Media, 2018 [cit. 2022-02-27]. Dostupné z: https://realpython.com/python-variables/
33. UPTON, Eben, Gareth HALFACREE a Jakub GONER. Raspberry Pi: uživatelská příručka. 2. aktualizované vydání. Brno: Computer Press, 2016. ISBN 978-80-251-4819-8
34. MÁCA, Jindřich. Lekce 1 - Úvod do Symfony frameworku pro PHP. Itnetwork.cz [online]. 2021 [cit. 2021-11-17]. Dostupné z: https://www.itnetwork.cz/php/symfony/zaklady/uvod-do-symfony-frameworku-pro-php
35. TICHÝ, Jan. Doctrine 2: úvod do systému. Zdrojak.cz [online]. 2010 [cit. 2021-12]. Dostupné z: https://zdrojak.cz/clanky/doctrine-2-uvod-do-systemu/
36. KONEČNÝ, Martin. Lekce 1 - Úvod do Doctrine 2 v Nette frameworku. Itnetwork.cz [online]. 2020 [cit. 2021-12]. Dostupné z: https://www.itnetwork.cz/php/nette/doctrine/tutorial-uvod-do-doctrine-2-v-nette-frameworku
37. VÍTEK, Rostislav. Symfony po krůčkách – Twig. Zdroják.cz [online]. 2016 [cit. 2021-12]. Dostupné z: https://zdrojak.cz/clanky/symfony-po-kruckach-twig/
38. GRUDL, David. Escapování – definitivní příručka. PhpFashion [online]. 2009 [cit. 2021-12]. Dostupné z: https://phpfashion.com/escapovani-definitivni-prirucka

39. WELLING, Luke a Laura THOMSON. Mistrovství PHP a MySQL. Přeložil Ondřej BAŠE. Brno: Computer Press, 2017. ISBN 978-80-251-4892-1.
40. LAURENČÍK, Marek. SQL: podrobný průvodce uživatele. Praha: Grada Publishing, 2018. Průvodce (Grada). ISBN 978-80-271-0774-2.
41. BEZBORODYKH, Iryna. 7 Reasons to Choose PHP Web Development for Business IT Solutions. Stfalcon [online]. Stfalcon, 2018, 2018-07-13 [cit. 2022-03-09]. Dostupné z: https://stfalcon.com/en/blog/post/PHP-advantages-for-business
42. BAROT, Saurabh. Laravel vs Symfony: Checkout What's The Key Difference In 2022?. AGLOWID [online]. Aglowid, 2022, 2022-02-07 [cit. 2022-03-06]. Dostupné z: https://aglowiditsolutions.com/blog/laravel-vs-symfony/
43. STE2 r2: LAN a WiFi teploměr pro vzdálené monitorování. HW group [online]. HW group, 2021 [cit. 2022-02-27]. Dostupné z: https://www.hw-group.com/cs/zarizeni/ste2-r2
44. STE2 R2 - WIFI A ETHERNETOVÝ TEPLOMĚR - SET. Obchod.hw.cz: monitoring prostředí a elektronika [online]. HW server, 2021 [cit. 2022-02-27]. Dostupné z: https://obchod.hw.cz/eshop/ste2-r2-wifi-a-ethernetovy-teplomer-s-di-vstupy-hw-group/
45. TME: Ethernetový teploměr. Papouch: store [online]. Papouch, 2021 [cit. 2022-02-27]. Dostupné z: https://papouch.com/tme-ethernetovy-teplomer-p4602/
46. ŠEVČÍK, Michal. Perfektní statistiky z naměřených hodnot. TMEP [online]. 2021 [cit. 2022-02-27]. Dostupné z: https://tmep.cz/
47. About SensDesk Technology. SensDesk.com [online]. Praha: HW group, 2020 [cit. 2022-02-27]. Dostupné z: https://sensdesk.com/user/login
48. SensDesk Technology: online služba pro správu vzdálených senzorů a zařízení od HW group. HW group [online]. Praha: HW group, 2022 [cit. 2022-02-27]. Dostupné z: https://www.hw-group.com/cs/software/sensdesk-technology
49. Dashboard [online]. HW group, 2020 [cit. 2022-03-06]. Dostupné z: https://sensdesk.com/sensdesk/dashboard/213
50. HWg-cloud.com Features. HWg: cloud [online]. HW group, 2021 [cit. 2022-02-27]. Dostupné z: https://hwg-cloud.com/user/login
51. Team dashboard demo [online]. HW group, 2021 [cit. 2022-03-06]. Dostupné z: https://hwg-cloud.com/sensdesk/dashboard/1
52. WESTON, Toby. Introduction. Temperature Machine [online]. 2020 [cit. 2022-02-27]. Dostupné z: https://temperature-machine.com/docs/
53. JEŽEK, Adam. Lekce 1 - Seznámení s Arduinem. Itnetwork.cz [online]. 2020 [cit. 2022-02-27]. Dostupné z: https://www.itnetwork.cz/hardware-pc/arduino/arduino-seznameni
54. TEJA, Ravi. What are the Differences Between Raspberry Pi and Arduino?. Electronics Hub [online]. Eletronicshub.org, 2021 [cit. 2022-02-28]. Dostupné z: https://www.electronicshub.org/raspberry-pi-vs-arduino/
55. What is an Arduino?. Opensource.com [online]. Red Hat, 2021 [cit. 2022-02-28]. Dostupné z: https://opensource.com/resources/what-arduino
56. AdminLTE Bootstrap Admin Dashboard Template [online]. AdminLTE, 2021 [cit. 2022-03-02]. Dostupné z: https://adminlte.io/
57. PHP: Downloads [online]. PHP Group, 2022 [cit. 2022-03-12]. Dostupné z: https://www.php.net/downloads.php
58. ADERMANN, Nils a Jordi BOGGAIANO. Download Composer [online]. 2022 [cit. 2022-03-12]. Dostupné z: https://getcomposer.org/download/

59. BERGMANS, San. Temperature Measurements. SB-Projects [online]. Udenhout, 2020 [cit. 2022-03-02]. Dostupné z: https://www.sbprojects.net/projects/raspberrypi/temperature.php
60. Teplotní senzor -50..125°C, čidlo 30x6mm, kabel 1m DS18B20 - 1m. GM Electronic [online]. GM Electronic, 2022 [cit. 2022-03-13]. Dostupné z: https://www.gme.cz/digitalni-teplotni-cidlo-s-ds18b20
61. Raspberry Pi OS [online]. Raspberry Pi, 2022 [cit. 2022-03-12]. Dostupné z: https://www.raspberrypi.com/software/

6 Přílohy

Příloha 1: CD se zdrojovými kódy aplikace