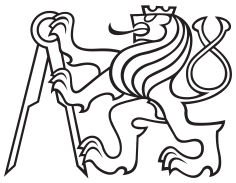
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačů

Aplikace pro správu chytré domácnosti

Petr Švagr

Vedoucí: Ing. Stanislav Vítek, Ph.D.
Studijní program: Softwarové inženýrství a technologie
Leden 2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Švagr** Jméno: **Petr** Osobní číslo: **465895**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Aplikace pro správu chytré domácnosti

Název bakalářské práce anglicky:

The application for smart home automation

Pokyny pro vypracování:

1. Proveďte přehled a analýzu dostupných aplikací pro správu chytrých domácností.
2. Proveďte analýzu HW modelu chytré domácnosti vybavené senzory a zařízeními s IoT konektivitou. HW model dodá vedoucí práce.
3. Na základě provedené analýzy navrhněte a na HW modelu implementujte aplikaci, která umožní uživateli jednoduchou správu chytré domácnosti. Uživatel by měl mít možnost získat přehled o stavu domácnosti, jednoduchým způsobem nastavit alarmy a ovlivňovat stav jednotlivých zařízení.
4. Proveďte analýzu zabezpečení vašeho řešení na úrovni komunikace bezi jednotlvými HW komponentami a ochrany jednotlivých komponent. Navrhněte vhodnou metodu ochrany dat, pokud by bylo nezbytné zvýšení výpočetního výkonu, tak případný upgrade jednotlivých komponent. Diskutujte, jakým způsobem ovlivní zabezpečení aplikace její návrh.

Seznam doporučené literatury:

[1] WAHER, Peter. Learning internet of things. Packt Publishing Ltd, 2015.
[2] LIN, Huichen; BERGMANN, Neil W. IoT privacy and security challenges for smart home environments. Information, 2016, 7.3: 44.
[3] RUSSELL, Brian; VAN DUREN, Drew. Practical internet of things security. Packt Publishing Ltd, 2016.
[4] KESTŘÁNEK, Tomáš. IoT technologie pro automatizaci domácnosti. Praha, 2019. Bakalářská práce. České vysoké učení technické v Praze.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Stanislav Vítek, Ph.D., katedra radioelektroniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **20.09.2019** Termín odevzdání bakalářské práce: **07.01.2020**

Platnost zadání bakalářské práce: **19.02.2021**

Ing. Stanislav Vítek, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

.	
Datum převzetí zadání	Podpis studenta

Poděkování

Rád bych poděkoval svému vedoucímu práce Ing. Stanislavu Vítkovi, Ph.D. za odborný dohled, cenné rady a vstřícnost během práce na projektu. Dále bych rád poděkoval své rodině za podporu během psaní práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 6. ledna 2020

Abstrakt

První část práce je zaměřena na technologie a vymezení základních pojmů spojených s Internetem věcí. Dále se zaměřuje na dostupné aplikace pro správu chytrých domácností. Práce pokračuje porovnáním dostupných nástrojů pro vizuální programování a jejich vlastností. Následuje analýza modelu chytré domácnosti použitého v práci. Praktickou část tvoří návrh a implementace aplikace pro ovládání chytré domácnosti za pomocí vizuálního programování. V závěru je řešeno zabezpečení dané aplikace.

Klíčová slova: Internet věcí, backend, frontend, WIFI, bezpečnost, Node-RED, chytrá domácnost

Vedoucí: Ing. Stanislav Vítek, Ph.D.

Abstract

The first part is focused on technology and definition of basic concepts associated with the Internet of Things. It also focuses on available applications for managing smart homes. Work continues by comparing the available visual programming tools and their properties. That follows an analysis of the smart home model used at work. The practical part consists of design and implementation of application for control of smart home using visual programming. In the final part the focus is on security of the aplication.

Keywords: Internet of Things, backend, frontend, WIFI, security, Node-RED, smart home

Title translation: The application for smart home automation

Obsah

Obrázky

Tabulky

Kapitola 1
Úvod

Internet věcí v dnešní době roste na popularitě. Tento pojem se v současnosti velmi často objevuje ve světě informačních a komunikačních technologií.

Hlavní myšlenka o propojení všemožných zařízení, senzorů a čidel není nová, ale až v dnešní době zažívá skutečný průlom v podobě internetu věcí (většinou označovaný jako IoT, z anglického Internet of Things). Má potenciál přetransformovat náš svět do něčeho, co jsme schopni si nyní jen stěží představit. Jeho základem je připojování „věcí“ do internetu. Popularita a zájem o Internet věcí roste pro firmy i jednotlivce. Díky tomu se o dostává v poslední době do povědomí o něm čím dál tím více lidí. Pochopení o co přesně jde a jak na něj nahlížet se zdá být problematické pro mnohé z nás. Internet věcí se stále rozrůstá a problematika s ním spojená je aktuálním tématem.

Tato práce se v první sekci zaměřuje na vysvětlení pojmů spojených s chytrou domácností, aby čtenář získal základní představu o zařízeních, technologiích a jejich využití ve spojení s chytrými domácnostmi a příklady dostupných řešení. Další část práce se zabývá navržením a implementací aplikace pro kontrolu chytré domácnosti. Poslední část je zaměřena na bezpečnost navrženého řešení a jak ovlivní změny zabezpečení aplikace její návrh.

1.1 Definice pojmů

1.1.1 Internet věcí

Podle světové výzkumné společnosti Gartner je možné internet věcí (IoT) definovat jako „síť fyzických objektů, která obsahuje vestavěné technologie pro komunikaci a vnímání nebo ovlivňuje jejich vnitřní stavy či vnější prostředí.“ [1].

Ke snazšímu pochopení pojmu internet věcí pomáhá i vysvětlení od společností Accenture a Bankinter Foundation of Innovation. Tyto společnosti ve své publikaci The Internet of Things: In a Connected World of Smart Objects říká, že: „Internet věcí se skládá z věcí připojených k internetu kdykoliv a kdekoliv. V technickém smyslu internet věcí začleňuje senzory a zařízení do běžných objektů, které jsou připojeny k internetu přes pevné nebo bezdrátové sítě.“ [2].

Definovat pojem internetu věcí je velmi složité. Každá společnost vysvětluje tento termín trochu odlišně, podstata je však u všech definic stejná. Jde tedy o systém vybudovaný pro komunikaci různých elektronických zařízení, jejich kontrolu i aktivní řízení prostřednictvím internetu. Do sítě je možné zapojit téměř libovolná zařízení, která si je člověk schopen představit.

Koncept internetu věcí popisuje, že věci lze připojit pomocí přenosových sítí k internetu. Vzniklé připojení zajistí komunikaci zařízení mezi sebou a s uživateli. Příklady zařízení, které lze považovat za tyto věci jsou chytré lednice, chytré váhy, senzory atd. Svět internetu věcí se rozšiřuje rychle a společnosti vyvíjejí stále nové věci, které je možné připojit, a považovat je za věci internetu věcí. Základem internetu věcí však nejsou věci jako takové, ale data, která zařízení poskytují. [3]

Pro hlubší pochopení konceptu IoT je třeba si definovat, co znamená označení „věc“. Jde o neživý objekt (fyzický nebo virtuální), který se skládá z hardwaru a softwaru a je schopen snímat určitá data (fyzikální veličiny, polohy strojů apod.). Jedná se tedy o zařízení, které autonomně poskytuje získaná data, která jsou dále sdílena s dalšími zařízeními, tj. „věcmi“. Příkladem mohou být např. senzory intenzity světla, teploty, vlhkosti, obsahu CO_2 a další. IoT tedy označuje síť fyzických objektů, které jsou vybavené technologií umožňující zachytit jejich vnitřní stavy a dále získaná data pomocí komunikace poskytovat svému okolí [6]. „Cílem IoT je propojení zařízení, systémů a služeb za účelem poskytnutí více dat, která mohou být převedena na informace a informace na znalosti, které lze následně aplikovat.“ [3]. To znamená, že čím více dat budeme schopni získávat a analyzovat, tím budeme mít k dispozici podrobnější informace, které povedou k rozvoji a pokroku.

Označení IoT se dá dále rozdělit podle oblastní jeho využití na „spotřebitelský internet věcí“ (CIoT – Consumer IoT) a „průmyslový internet věcí“ (IIoT – Industrial IoT). Oba sektory dohromady představují vše, co se od IoT očekává. CIoT lze charakterizovat jako odvětví, ve kterém jsou aplikace orientované na spotřebitele, především pak na spotřebitelská zařízení. Tato zařízení, často označovaná jako inteligentní, denně usnadňují životy lidí v podobě automatizace procesů v domácnosti (chytré hodinky, ledničky a jim podobná zařízení). Selhání spotřebitelských aplikací nemívá kritické následky, například selhání chytrého náramku nezpůsobí jiné škody mimo finančních. IIot se zaměřuje na průmyslové aplikace. Jde o zařízení pro chytrá města, průmysl, dopravu atd. Selhání zařízení v těchto případech může mít výrazný vliv hlavně na bezpečnost. Aby IIoT bylo efektivním řešením, musí splňovat základní požadavky. Prvním a nejvíc diskutovaným požadavkem je bezpečnost přenosu dat. Při jejich přenosu musí být nutně zajištěno šifrování nebo jiný způsob zabezpečení, a to především při bezdrátovém přenosu, aby nedošlo k odposlechu či jinému zneužití dat neoprávněným prostředkem/uživatelem. Mezi další požadavky patří například interoperabilita, efektivní i bezpečný přenos a zpracování dat.

1.1.2 Cloudové služby

Jde o služby, aplikace a prostředky, které jsou poskytovány prostřednictvím internetového spojení. U cloudu poskytovatel disponuje vlastními výpočetními prostředky (servery, úložiště, aplikace), které rychle poskytne zákazníkovy s minimálním úsilím a interakcí. Tyto služby jsou pomocí internetu zpřístupněné zákazníkovi za nízké poplatky s možností okamžité rozšiřitelnosti dle potřeby zákazníka. Tedy veškeré prostředky jako vývojové platformy, infrastruktura, software jsou poskytovány jako služba, ke které zákazník přistupuje přes internet pomocí webového rozhraní nebo poskytované aplikace. Služby jsou zpoplatněny dle různých tarifů, nejčastějším bývá tarif, ve kterém platí zákazník za to, co doopravdy využívá („pay as you go“). Dalším možným je platba za každého uživatele(„pay per user“). Tyto tarify jsou nejčastěji placené v měsíčních či ročních intervalech. Mnoho poskytovatelů těchto služeb nabízí tzv. „Free-trial“, tedy uživatelský účet, který na jisté období zpřístupní všechny služby. Zákazník má možnost bezplatně vyzkoušet služby, aniž by se na poskytovatele vázal.

1.1.3 Chytrá domácnost

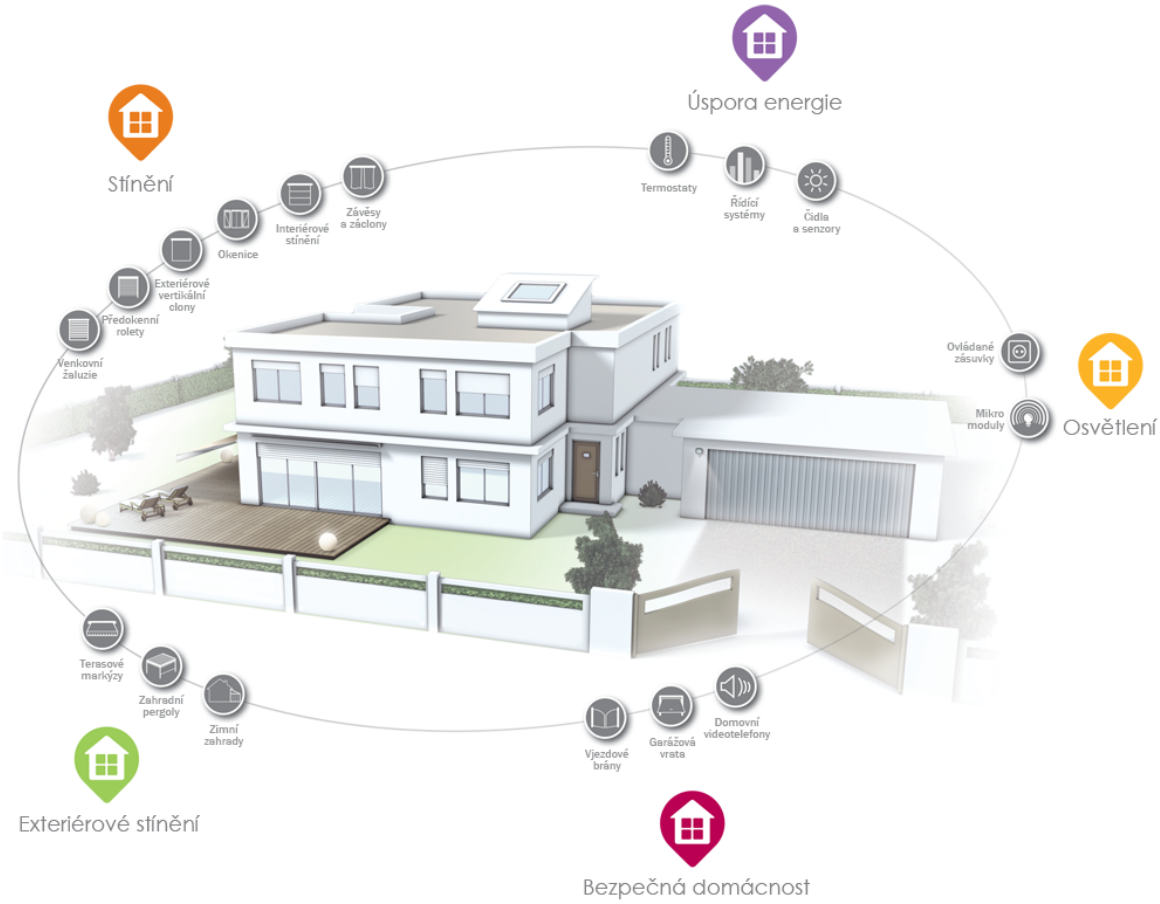
Občas je také nazývána inteligentní domácností. Jde o domácnost s přizpůsobeným nastavením, kde lze spotřebiče a zařízení automaticky ovládat na dálku z libovolného místa s přístupem k internetu, za pomoci mobilního telefonu nebo jiného síťového zařízení. Dům má svá zařízení propojená přes internet, takže uživatel může ovládat funkce, jako je bezpečnostní přístup do domu, teplota, osvětlení, domácí kino a podobně.

1.1.4 Sensory

Hlavním zdrojem dat v IoT jsou senzory. Detailněji lze říct, že se jedná o zařízení, která zajišťují přeměnu měřené veličiny na veličinu snáze měřitelnou (napětí). Část senzoru zaznamenávající stav měřené veličiny je obvykle označovaná jako čidlo. Informace z čidla se zpracovává ve vyhodnocovacím obvodu senzoru. Důležitou částí je také část výstupní, která slouží ke komunikaci s okolím, tedy k přenosu získané informace do sběrného zařízení. Senzory lze dělit podle[15]:

- Typu měřené veličiny (teplota, tlak, průtok atd.)
- Fyzikálního principu činnosti (magnetické, odporové, indukční, apod.)
- Aktivní (vyžaduje externí napájení) a pasivní (bez externího napájení)
- Styku senzorů s měřeným prostředím (dotykové, bezdotykové)
- Tvaru výstupní veličiny na spojité (analogové) a nespojité (diskrétní)

S velkým rozmachem IoT v posledních letech se trh zaplavil především malými levnými senzory, které jsou nabízeny doslova za pár korun a poskytují tak jednoduchá a laciná řešení. Tyto senzory je možné zakoupit jako



Obrázek 1.1: Ilustrace chytré domácnosti. Zdroj: [7]

jednoduché drátové senzory nebo jako hotová bezdrátová řešení připravená k bezdrátové komunikaci.

1.1.5 Vizuální programování

Jak již z názvu plyne, jde o programovací techniku, u které pracujeme s grafickými objekty. Programování je spíše spjato s psaním textu, který respektuje pravidla jednoho z mnoha programovacích jazyků, kde se pracuje se vším pouze pomocí textových odkazů.

Ve vizuálním programování pracujeme s grafickými reprezentacemi objektů, funkcí a proměnných, na rozdíl od dřívějších způsobů, kdy jsme se na ně odkazovali slovně. V praxi to znamená, že funkcionalitu, kterou jsme dříve v programovacím jazyce pouze slovně popisovali, nyní vidíme na ploše znázorněnou jako obrázky a tvary.

Kapitola 2
Technologie

2.1 Komunikační technologie

Propojení zařízení je velice důležité nejen pro chytré domácnosti. Tyto technologie používáme denně, jen o nich například nevíme. Tyto technologie dělíme na fyzické či bezdrátové, podle přenosového media, které je použito k přenosu dat.

Fyzické přenosové cesty jsou nejčastěji metalické nebo optické. U metalického spojení jde nejčastěji o koaxiální kabel, krouceného dvoudrátu, či jen vodiče položené paralelně vedle sebe. Optické vlákno pracuje se světlem využívá principu tzv. absolutního lomu, jde o rozhraní dvou prostředí s odlišným indexem lomu, které zajišťují absolutní odražení paprsku světla. Tento způsob přenosu dat nabízí bezpečnější přenos dat a je oproti metalickému vedení složitější na odposlech. Optika nabízí v porovnání s metalickým vedení větší šířku pásma a díky nižšímu útlumu je vhodnější na větší vzdálenosti.

V chytrých domácnostech je z důvodu náročnosti fyzických spojení na infrastrukturu nejrozšířenější bezdrátová komunikace. Díky právě nižší náročnosti na infrastrukturu bezdrátové komunikace je možné bez větších obtíží předělat i starší dům na chytrou domácnost. Ale v případě nové stavby chytré domácnosti jsou použity i metalická vedení a optická vlákna, neboť jsou méně náchylná k rušení. Absence elektromagnetického záření, které přichází s bezdrátovými řešeními, může být pro mnoho lidí důležitým faktorem pro volbu fyzických spojení.

2.1.1 Ethernet

Jedná se o jednu z nejtypičtějších technologii používanou pro fyzické spojení více zařízení v síti. Za rozšířenost vděčí vlastnostem jako je široké uplatnění, spolehlivost a jednoduchá správa. Rychlosti přenosu se pohybují od 1 Mbit/s do 100 Gbit/s.

K prvnímu zveřejnění došlo v roce 1980. Tento první zveřejněný standard je označován jako DIX Ethernet dosahoval rychlosti 10 Mbit/s byl postaven na technologii 10Base5. Do vývoje přispěla s vlastní verzí standardu v roce 1983 společnost IEEE[1], která rozšířila původní verzi, nesla název 802.3 CSMA/CD.

[1]Institute of Electrical and Electronics Engineers, mezinárodní nezisková profesní orga-

Přestože tyto názvy nenesou slovo Ethernet, jedná se totiž o neoficiální název, kterým se označují standardy z řady IEEE 802.3. Řada je postupem času s vznikem nových technologii rozšiřována a vylepšována i dnes. [8]

2.1.2 PowerLine Communication

PowerLine Communication (PLC) jedná se o technologii, která k přenosu dat využívá elektrickou síť. Komunikační systémy elektrického vedení fungují přidáním modulovaného nosného signálu do elektroinstalačního systému. Různé typy komunikace po elektrické síti používají různá frekvenční pásma. Protože systém distribuce energie byl původně určen pro přenos střídavého výkonu při typických frekvencích 50 nebo 60 Hz, mají obvody silového drátu pouze omezenou schopnost přenášet vyšší frekvence. Problém šíření je omezujícím faktorem pro každý typ komunikace po elektrické síti.[18]

Úzkopásmové PLC

Úzkopásmové PLC pracuje při nižších frekvencích (3 500 kHz), nižších přenosových rychlostech (až 100 kbps) a má delší dosah (až několik kilometrů), který lze rozšířit pomocí opakovačů.

V poslední době k sobě úzkopásmové PLC přitahuje více pozornosti díky svému využití v chytrých domácnostech. Jednou z dalších aplikací, ve které se úzkopásmové PLC používá, je inteligentní výroba energie, zejména v mikrostřídačích pro solární panely.[18]

Širokopásmové PLC

Širokopásmové PLC pracuje na vyšších frekvencích (1,8-250 MHz), vysokých datových rychlostech (až 100 Mbps) a používá se v aplikacích kratšího rozsahu.

Naproti tomu širokopásmové PLC našlo řešení jako poslední prvek u internetovou distribuce a domácích sítí. Díky vysokému datovému toku a bez dodatečného zapojení je širokopásmové PLC považováno za vzrušující a efektivní technologii pro multimediální distribuci v domácnostech. [18]

2.1.3 WiFi

WiFi je pravděpodobně nejznámější bezdrátová technologie, kterou najdeme téměř v každé domácnosti. Rozsah WiFi sítě se pohybuje okolo 30 metrů uvnitř budov a 90 metrů na volném prostranství. Základ technologie definuje standard IEEE 802.11, který byl rozšířen a vznikly z něho další standardy, které se značí písmeny (IEEE 802.11n, IEEE 802.11ac, apod.). Komunikace probíha v pásmu ISM[2] na frekvenci 2,4 GHz nebo 5 GHz. [13]

Nejnovějším standardem do rodiny WiFi je standard IEEE 802.11ax (také označován WIFI 6). Tato verze oproti přechůdci (WiFi 5 - IEEE 802.11ac)

nizace sídlící ve Spojených státech amerických

[2]ISM (zkratka z anglického Industrial, scientific and medial) jde o souhrn volných pásem radiové komunikace používaných v průmyslových, vědeckých a zdravotnických zařízeních

nabízí několik nových funkcí a vyšší rychlost zhruba o 40%. Jendou z nových funkcí je funkce TWT (Target wake time) funkce umožňuje přístupovému bodu pro jednotlivá zařízení zarezervovat čas, kdy má komunikace probíhat. Díky této funkci se mohou například zařízení IoT pravidelně vypínat a šetřit tak baterii. Další funkcí je OFDMA (Orthogonal frequency division multiple access) jde o princip sdílení WiFi kanálů pro zvýšení efektivity a nižší latenci v oblastech s velkou hustotou sítí. MU-MIMO (Multi-user multiple input multiple output) ve zkratce zvyšuje limit zařízení, která mohou v jednu chvíli přijímat data. [14]

Historie Wifi Verzí:

- **WiFi 1 - 802.11b z roku 1999**
- **WiFi 2 - 802.11a z roku 1999**
- **WiFi 3 - 802.11g z roku 2003**
- **WiFi 4 - 802.11n z roku 2009**
- **WiFi 5 - 802.11ac z roku 2014**
- **WiFi 6 - 802.11ax z roku 2019**

2.1.4 Bluetooth

Jde o další velmi dobře známou bezdrátovou technologii, která si v poslední době nachází cestu do inteligentních domácností. Pomocí technologie Bluetooth vytváříme tzv. WPAN[3] sítě. Tato technologie byla vyvíjena jako náhrada komunikace přes USB a sériovou linku. Stejně jako v případě WiFi, tak i Bluetooth pracuje v ISM pásmu 2.4 GHz.

Pro komunikaci IoT zařízení mezi sebou je nejčastěji používán BLE (Bluetooth Low Energy). Jde o verzi bluetooth, která byla světu představena v roce 2011. Tento typ pracuje na frekvenci 2.4 GHz. Jak je z názvu patrné oproti klasickému Bluetooth spotřebovává menší množství energie. BLE je stále v režimu spánku a komunikuje periodicky jen na pár milisekund a díky tomu se jeho spotřeba pohybuje jen v řádech jednotek A. Je tedy ideální pro zařízení napájená baterii.[19]

2.1.5 ZigBee

Jedna z populárních technologii k Bluetooth a WiFi je nazývána ZigBee, jde o nadstavbu nad standardem IEEE 802.15.4. Tato technologie je určena k vytváření bezdrátových osobních sítí (WPAN). Standard, nad kterým je postavena, definuje fyzickou vrstvu a podvrstvu MAC z vrstvy spojové. Technologie komunikuje v pásmu ISM na 2.4 GHz s možnou přenosovou rychlostí 250 kbit/s. V USA existuje také varianta s rychlostí 40 kbit/s

[3]WPAN (z anglického Wireless Personal Area Network) označuje bezdrátovou osobní síť, která je obvykle tvořena komunikujícími zařízeními (mobil, PDA, notebook) poblíž jejich vlastníka.

pracující v ISM na 915 MHz. Evropský ekvivalent k této americké verzi je v pásmu 868 MHz s přenosovou rychlostí do 20 kbit/s. Tato technologie pracuje s nízkým množstvím napájení (pouze 1 mW) a je tedy vhodná pro zařízení s bateriovým napájením. Rozsah se podle vlastností prostředí pohybuje okolo 10 až 30 metrů.

2.1.6 RFID

Radio Frequency Identcation (RFID) je technologie vyvinuta především jako náhrada čárových kódů, ale své uplatnění si našla i v chytrých domácnostech. V této technologii existují dva hlavní typy zařízení. Prvním typem je takzvaný TAG, u kterého dále rozlišujeme 3 druhy. Aktivní má vlastní zdroj napájení (například baterii), pasivní získává napájení během čtení z elektromagnetického pole čtečky a třetím druhem je kombinace tzv. semi-aktivní, kdy je napájena pouze paměť čipu. Velikosti pamětí na těchto zařízeních se pohybují mezi 4 byte a 8 kByte. Druhý typ zařízení je čtečka, která slouží pro čtení TAGů. Pro přenos dat mezi čtečkou a TAGem jsou využívána bezlicenční pásma nejčastěji jde o nízkofrekvenční pásma (125 - 134 kHz) na vyšších frekvencích jde o HF (13.56 MHz), velmi vysokých frekvencích VHF (860 - 930 MHz) a mikrovlnných frekvencích MW (2.4 GHz a 5.8 GHz). Díky rozdílným frekvencím se čtecí vzdálenost liší v rozsahu od 80 cm až k několika desítkám metrů. [20]

2.1.7 Z-Wave

Nejrozšířenější technologii v oblasti IoT je technologie z-wave. Technologie nabízí řešení, které pokrývá všechny vrstvy ISO/OSI modelu od fyzické po aplikační. Komunikace funguje na vzdálenost 40 metrů s nízkou latencí a přenosovou rychlost až 100 kbit/s. Počet zařízení v jedné síti z-wave je limitováno na 232, ale sítě je možné vzájemně propojovat, čímž dochází k navýšení limitu.

Z-wave rozlišuje dva typy zařízení jedním je Controller (řídící jednotka) a druhým je Slave (podřízené zařízení). Všechna zařízení mají Home ID, jedná se o identifikátor zařízení v rámci sítě. V jedné síti je vždy pouze jedna primární řídící jednotka, všechna ostatní zařízení v síti jsou podřízena této jednotce a své Home ID přebírají od primární řídící jednotky. Každé zařízení má v rámci jedné sítě své vlastní Node ID, podzařízení s různými Home ID mezi sebou komunikovat nemohou. Celá tato síť je meshového typu, což znamená že zařízení ve stejné síti mezi sebou mohou komunikovat a tímto způsobem může zařízení které je mimo dosah primární řídící jednotky komunikovat s touto jednotkou přes ostatní podzařízení ve stejné síti. Řídící uzel si postupně skládá tabulku s cestami k jednotlivým podzařízením v síti. Při připojení tzv. inkluzi, při ní zařízení získá informace od řídící jednotky. Při odpojení zařízení dochází k resetování Home ID i Node ID, této akci se říká exkluze. [9]

Technologie je díky topologii mesh a frekvenci pod 1 GHz ideální do domácností. Díky své nízké frekvenci je méně náchylná na rušení od technologii

jako WiFi, která je především v místech s hustou obydleností, díky své rozšířenosti, velice hojná a dochází tedy k rušení sítí na frekvenci 2,4 GHz. Zároveň oproti své konkurenci jako ZigBee se Z-wave jeví jako kompletnější řešení díky pokrytí celého ISO/OSI modelu a garanci kompatibility současných i budoucích zařízení.

2.1.8 Internet Protocol address

Každé zařízení v rámci internetu věcí musí mít unikátní identifikátor, aby mohla být umožněna komunikace s daným zařízením. Původní myšlenkou internetu věcí bylo zařízení identifikovat pomocí RFID (Radio Frequency Identification) kódů podle Kevina Ashtona (jeden ze zakladatelů původního Auto-ID Centra) z MIT. Později se od RFID kódů přešlo právě k adresám IP (Internet Protocol address).

IP adresa je unikátní identifikační číslo, které je přiděleno zařízení komunikujícímu prostřednictvím internetového protokolu (IP). Pro úspěšnou komunikaci dvou zařízení je nutné znát IP adresu odesilatele i příjemce. O směrování datových paketů na základě IP adres se pak starají směrovače (routery), které disponují svoji vlastní IP adresou. Tento identifikátor může mít různé podoby. Záleží na verzi protokolu (IPv4 vs IPv6).

Starší verze 4 (IPv4) je v současné době stále ještě nejrozšířenější. Tento typ IP používá 32bitové adresy, které jsou zapisovány po osmi bitech. Nejčastěji zobrazovány v podobě čtyř čísel v rozsahu 0 až 255 oddělené tečkami. Tvůrcům se původní specifikace, s maximálním využitím cca 4 miliard IP adres, zdála dostatečná. V aktuální době se těchto adres zřízením nedostává a je zapotřebí používat různé triky, aby nedošlo ke kompletnímu vyčerpání všech adres.

Tyto nedostatky adres se snažíme zmírnit pomocí NAT (Network address translation), kdy k jedné veřejné IP adrese je připojena síť zařízení. Nevýhodu ovšem je, že nejsme schopni se z venku NAT sítě připojit k zařízení v této síti přímo.

Jak je již zmíněno výše, IPv4 nabízí omezený počet IP adres, proto došlo okolo roku 2008 k pozvolnému nástupu nového standardu IPv6. Ten pracuje s adresami o velikosti 128bitů, které jsou nejčastěji zobrazovány v hexadecimálním zápisu. Počet adres v této verzi vzrostl na 3,4 * 10$\hat{3}$2. IPv6 je budoucnost a oproti svému předchůdci nabízí několikanásobné množství adres, které se zdá být až nemožné využít, ale to si tvůrci IPv4 říkali také.

2.1.9 Message Queuing Telemetry Transport

Jde o jednoduchý a nenáročný protokol, založený společností IBM. Slouží pro předávání zpráv mezi klienty prostřednictvím centrálního bodu – brokeru. Krátce po přechodu MQTT pod společnost Eclipse Foundation, proběhla standardizace OASIS (Organization for the Advancement of Structured Information Standards). Díky své nenáročnosti a jednoduchosti je snadno implementovatelný i do zařízení s „malými“ procesory a poměrně rychle se rozšířil. [16]

U protokolu MQTT (Message Queuing Telemetry Transport) probíhá přenos pomocí TCP (Transmission Control Protocol) a používá návrhový vzor publisher – subscriber. V němž se nachází jeden centrální bod (MQTT broker), který řídí výměnu zpráv. Zprávy jsou řazeny do tzv. témat (topic). Koncová zařízení buď posílají aktualizace (publish) k danému tématu nebo jsou přiřazeny k odběru (subscribe) tématu či více témat a přijímají změny, které se s tématem pojí. Jedno zařízení může najednou být v některých tématech publisher, v jiných subscriber.[16]

Obsah zpráv nemá žádná pevně stanovená pravidla je „payload agnostic“. Obsahem zprávy jsou libovolná binární data. Nejčastěji se pro reprezentaci dat používá JSON (JavaScript Object Notation) nebo BSON (Binary JavaScript Object Notation). V současné době je maximální velikost zprávy limitována na 256 MB, ale velikost většiny zpráv je mnohem menší.[16]

„MQTT minimalizuje množství balastních dat, takže přidává jen minimum servisních dat. Zavádí tři úrovně QoS (Quality of Service) – tedy potvrzování zpráv, kde ta nejnižší znamená, že zpráva je odeslána bez potvrzení a není zaručeno její doručení (at-most-once), prostřední úroveň říká, že zpráva je doručena alespoň jednou (at-least-once), a nejvyšší úroveň QoS 2 znamená, že každá zpráva je doručena právě jednou. Klient však nemusí podporovat všechny tři úrovně QoS.“[16]

2.1.10 Hypertext Transfer Protocol

Hypertext Transfer Protocol (HTTP) je protokol hojně využívaný pro přenos webových stránek, ale jeho využití na ně není omezeno. Protokol vznikl v roce 1991 jeho první verze nesla název HTTTP/0.9. V tuto chvíli protokol sloužil k přenosu pouze binárních dat (používá pouze metodu GET). Další verze HTTP/1.0 přidala k požadavku hlavičky (krátký textový blok s informacemi o přenosu). Přibývá MIME hlavička, která rozšiřuje možné typ přenášených dokumentů a nové metody HEAD, POST. Dnes nejpoužívanější verzí je verze HTTP/1.1, která rozšiřuje funkce protokolu o možnost přenosu více souborů v sekvenci po sobě v jednom spojení a možnost udržet trvalé TCP spojení. V této verzi list metod vypadá následovně:[21]

- GET - Výchozí metoda pro zobrazení hypertextových stránek. Proměnné jsou odesílány jako součást url (limit 512 bajtů dat) (nejpoužívanější metoda).
- HEAD - Podobá se GET, avšak slouží pouze k získání metadat o cíli.
- POST - Slouží k odesílání dat (limit velikosti je nastaven na serveru většinou v řádech MB) na server. Tato metoda je používána pro velký objem dat, odesílání webových formulářů nebo pokud odesílaná data nemají být součástí url.
- PUT - Metoda sloužící k nahrání dat na server nebo modifikaci již existujících dat

- DELETE - Smaže cílový objekt na serveru.
- TRACE - Odesílá kopii požadavku zpět odesilateli, aby bylo možné zjistit, jak požadavek upravují servery, kterými prošel.
- OPTIONS - Požadavek na server pro zjištění podporovaných metod
- CONNECT - Slouží k vytvoření TCP/IP tunelu (nejčastěji používán pro šifrovanou SSL komunikace přes HTTP proxy).
- PATCH - Provádění částečných změn existujících objektů.

Protokol pracuje na bázi dotaz a odpověď. Odpovědi od serveru mohou obsahovat následující stavové kódy:

- 1xx - jde pouze o informační zprávu
- 2xx - požadavek úspěšně zpracován
- 3xx - server potřebuje zaslat doplňující informace
- 4xx - chyba v požadavku
- 5xx - chyba na straně serveru

Hypertext Transfer Protocol Secure

Rozšiřuje protokol HTTP o šifrovací vrstvu. K šifrování je použit protokol SSL(Secure Sockets Layer) nebo TLS(Transport Layer Security). V dnešní době je častěji používán novější protokol TLS. Ovšem princip obou protokolů je založen na autentizaci pomocí X.509 certifikátů a šifrovaní pomocí veřejného a privátního klíče. O zajištění důvěryhodnosti certifikátu se starají certifikační autority, které tyto certifikáty vydávají. [22]

2.2 Machine-to-Machine

Tradičně M2M (Machine-to-Machine) systémy byly typicky používány k monitorování, kontrole nebo optimalizaci jediného procesu. Tyto systémy byly navrženy a zavedeny za specifickým účelem řešit vzniklé problémy okamžitě. Typickým příkladem použití staré M2M architektury by mohla být budova s oddělenými systémy pro řízení klimatizace, bezpečnosti a detekci požáru. V celku častým jevem je, že tyto diskrétní systémy přijímají stejná měřená data ze stejných míst od samostatných snímačů (Senzorů). Vše je zapojeno tak, aby data ze senzorů proudila do každého systémů separátně. Všechny tyto systémy tím pádem komunikují interně s využitím vlastních (často i proprietárních) protokolů a standardů. Sdílení dat mezi takovými proprietárními systémy je obtížné, často i nemožné.

U internetu věcí se jedná o systémy systémů. Jde o to, že zařízení produkující data zveřejňují informace, aniž by musela vědět, jaké aplikace nebo jací

uživatelé budou s těmito daty dále pracovat. Data v IoT jsou k dispozici více systémům tak, že mohou být použita v rozšiřujících systémech a revidována v okamžiku, kdy se objeví nové požadavky. Je zapotřebí zkoumat závislosti mezi zdánlivě nesouvisejícími datovými sadami. Optimalizovat tedy nelze jen jeden proces, ale celý systém, při získání schopnosti přizpůsobovat a vyvíjet nové požadavky a používat výchozí případy. Je to o měnícím se využívání stávajících dat a jejich volném kombinování, aniž by bylo třeba definovat vše, co chcete dělat, v první den projektu. Z prvního příkladu je jasné, že M2M systémy jsou určeny pro jednu činnost s jejich rozšířením se již nepočítá, ale v případě potřeby systém rozšiřovat je internet věcí lépe připraven na nové požadavky. Architektury IoT musí poskytnout tuto flexibilitu, která nikdy nebyla nejdůležitějším aspektem pro tradiční M2M systémy.[5]

Kapitola 3

Dostupné aplikace a řešení pro správu chytrých domácností

3.1 Hotové produkty pro správu chytrých domácností

Vzhledem k rostoucí popularitě chytrých domácností se trh s řešeními pro jejich správu stále rozrůstá. Ať jde o výrobu koncových zařízení jako jsou chytré žárovky, kamery, žaluzie, atd. nebo o možnosti jejich kontroly pomocí centrálních jednotek, jejichž výběr je také velice rozsáhlý. Do oblasti chytrých zařízení se pustili i velikáni jako Apple, Google nebo Amazon.

3.1.1 Google Home

Google Home je chytrý domácí asistent, který je přímým konkurentem Amazon Echo (více v sekci 3.1.3). Asistent nabízí hlasové ovládání pomocí Google Asistenta v něm právě tkví síla Google Home. Asistent je schopen odpovídat i na komplikovanější otázky. Ovládat je možné libovolné zařízení. Párování je o něco složitější v porovnání například s Apple HomeKit (více v sekci 3.1.2), na druhou stranu nabízí větší množství kompatibilních IoT zařízení.

3.1.2 Apple Homekit

Apple HomeKit je řešení od společnosti Apple, které slouží pro ovládání a automatizaci procesů v chytré domácnosti. Nabízí možnost hlasového ovládání za pomoci chytré asistentky Siri. Apple toto řešení představil již v roce 2014 a od té doby je zdokonalováno. Jedná se o uzavřený systém, do kterého je možné přidávat pouze zařízení přímo od společnosti Apple nebo zařízení od jiných výrobců s certifikací Apple Homekit. V roce 2016 přibyla aplikace Domácnost, která nabízí možnosti ovládat zařízení v domácnosti z libovolného zařízení Apple. Dále nabízí možnost rozdělit zařízení do virtuálních místností, které je možné ovládat pomocí hlasových příkazů jako například "Hey Siri, turn off the lights in living room"(česky - "Hey Siri, vypni světla v obývacím pokoji"). Celé řešení je zároveň velice dobře zabezpečeno díky End-to-End šifrováni, které probíhá při komunikaci mezi zařízeními. [11]

3.1.3 Amazon Echo

Jde o velice podobné řešení k již zmíněnému Google Home. Amazon ke svému asistentu Alexa dodává funkcí pro ovládání IoT zařízení stejně jako Google. Ale nabízí ještě větší množství kompatibilních zařízení, než jejich konkurenti. Hlasové ovládání je srovnatelné s Google Home, zařízení je spojené s obchodem firmy Amazon a je možné si pomocí tohoto asistenta objednávat produkty přímo z obchodu.

3.1.4 FIBARO

Jde o systém pro domácí automatizace, ovládání a sledování stavů. Zařízení od firmy Fibaro komunikují přes technologii Z-wave (více 2.1.7). Společnost se zaměřuje především na koncová zařízení jako chytrá topení, žaluzie, zásuvky, apod. nabízí, ale i vlastní aplikace pro správu celého systému chytrých zařízení, tyto aplikace však nemají tolik funkcí jako konkurence. Pokud by základní funkce uživateli nestačili, je možné díky kompatibilitě ovládat zařízení přes Apple Homekit.

3.2 Aplikace pro správu

3.3 Prostředí pro vizualní programování IoT

3.3.1 Node-RED

Je open-source vizuální programovací nástroj, který je uživatelsky přívětivý. Používá k programování tzv. flow-based systém. Chování je popsáno pomocí série uzlů (nodů), kde každý má svůj specifický účel. Složením několika nodů za sebe, vzniká datový tok (flow). Data tedy postupně proudí skrz uzly, kde dojde k jejich úpravě na základě funkce nodu.[17]

Tento způsob programování vyniká svou jednoduchostí a uživatelskou přívětivostí, kdy je možné vytvářet vlastní aplikace bez nutnosti psaní vlastního kódu. Tomu přispívá i fakt, že možnost editovat jednotlivá flow jde jednoduše z libovolného webového prohlížeče. Prostředí je velmi intuitivní na ovládání. Z toolbaru na levé straně stačí vybrat potřebné nody přetáhnout je pomocí myši na pracovní plochu a propojit je vzájemně opět za použitím myši a vytvářet tak unikátní řídicí jednotky. K dispozici je velká škála nodů, kterou lze kdykoliv rozšířit o nové nody vytvářené komunitou. Následně je možné hotové aplikace nebo jen jednotlivé skupiny nodů sdílet či přenášet ve formě JSON souborů.[17]

Node-RED již od základů podporuje širokou škálu zařízení a systémů jako MQTT, Amazon Alexa, Watson, Raspberry Pi a další. Pokud systém nebo typ přenosu zařízení není v základní verzi nástroje podporován, existuje možnost stáhnout si komunitou vytvořené nody, které toto zařízení či systém podporují. List těchto komunitních nodů je opravdu rozsáhlý a proto by neměl být problém najít ten co uživatel potřebuje. Pokud přesto neexistuje

node pro zařízení existuje funkční uzel, který nabízí možnost napsat do něho vlastní funkcionalitu nebo je možné si vytvořit vlastní node, který se o vše postará.

3.3.2 Blockly

Jedná se o JavaScriptovou knihovnu pro tvorbu visuálního programovacího prostředí. Knihovna pracuje na straně klienta, není k ní tedy potřeba žádný server a zároveň je licensována jako open-source. Blockly nabízí rozsáhlé možnosti pro přizpůsobení ať jde o vzhled bloků, tvorbu bloků nebo možnosti přeložení do textového programovacího jazyka. Již ze základu blockly nabízí překlad do jazyků: JavaScript, Python, PHP, Lua, Dart.

Kapitola 4

Porovnání nástrojů pro vizuální programování

4.1 Frontend Blockly vs Scratch

Jak je vidět z tabulky 4.1 Blockly je pro tvorbu vlastního visuálního programovacího prostředí lepší. Nabízí širší možnosti, v případě této práce je rozhodujícím faktorem pro volbu Blockly možnost tvorby vlastních bloků. Scratch se mnohem více hodí pro tvorbu animací či her nebo jako pomocník pro učení základů programování i pro mladší generace.

4.2 Backend Blockly vs Node-RED

Co se týče funkcí má Node-RED výrazně navrch, již od základů umí pracovat s MQTT, podporuje Raspberry Pi, atd. Zatímco v Blockly by bylo nutné pro podporu těchto technologii vytvářet vlastní bloky a psát pro ně kód. Zatímco Blockly nabízí pouze generování kódu, který musí uživatel sám spustit na serveru, součástí prostředí Node-RED je možnost rovnou aplikaci spustit na serveru. 4.2

Je jednoznačné, že Blockly je spíše základní stavební kámen, pomocí kterého je možné vytvořit visuální programovací prostředí pro libovolný účel. Zatímco Node-RED je spíše nástroj vytvořený pro serverový IoT backend.

	Blockly	Scratch
Tvorba vlastních bloků	x	
Přeložení do dalších jazyků	x	
Kompatibilní s webovými prohlížeči	x	x
Animace		x
Matematické operace	x	

Tabulka 4.1: Tabulka porovnání nativních funkcí Blockly a Scratch

	Blockly	Node-RED
Tvorba vlastních bloků	x	x
Přeložení do dalších jazyků	x	
Kompatibilní s webovými prohlížeči	x	x
Matematické operace	x	x
Logické operace	x	x
Podpora asynchronního chování		x
Práce přímo s HW zařízením		x
Komunikace s domácím asistentem		x

Tabulka 4.2: Tabulka porovnání nativních funkcí Blockly a Node-RED

Kapitola 5
Analýza modelu

Více informací o modelu naleznete v bakalářské práci Tomáše Kestřánka, který model vytvořil. [12]

5.1 Hardware

Arduino Ethernet

Jedná se o programovatelné desky s mikrokontrolerem, které v tomto modelu zajišťují propojení senzorů a ovladatelných zařízení se sítí přes rozhraní RJ-45. Deska obsahuje řadič W5100, který usnadňuje komunikaci přes MQTT.

Raspberry PI

Jde o kompletní mini počítač s půdorysem o velikosti platební karty. Operační systém je linuxovou distribucí odvozenou z distribuce Deban a optimalizovanou speciálně pro toto mini PC. S čtyřjádrovým 64bitovým procesorem ARM Cortex-A53 na frekvenci 1,4 Ghz, 1 GB LPDDR2 SDRAM, 4 x USB 2.0 porty, gigabitový Ethernet s RJ-45 konektorem a jeden HDMI port. Pro bezdrátovou komunikaci nabízí Wifi (ve verzi 5) s podporou pásem 2,4 GHz a 5 GHz.

TFT Displaye

Uživatel může pomocí dotykového displeje ovládat některá zařízení na modelu. Jde o display s LCD panelem o rozlišením 320 x 240 pixelů zároveň je display potažen dotykovou vrstvou, která slouží k detekci intenzity a místa stisku displaye. Displaye jsou na modelu celkem čtyři. Dva slouží k ovládání větráku, jeden k ovládání světla a poslední je určen k zobrazování informací o intenzitě světla získaných z fotorezistoru.

Síť

Pro komunikaci zařízení mezi zařízeními je vytvořena síťová infrastruktura. Router slouží jako hlavní řídící síťová jednotka, ten nabízí možnost do sítě připojovat další zařízení přes síť WiFi a řídí adresaci zařízení v této síti.

Vzhledem k tomu, že použitý router nabízí pouze připojení 4 zařízení, byl do sítě přidán switch, který propojuje senzory a router.

Napájení

Všechna zařízení jsou napájena externě průmyslovým zdrojem s napětím 5V a maximálním proudem 12 A. Tento zdroj je dostatečně výkonný nabízí i menší rezervu při maximální možné spotřebě modelu.

5.2 Software a funkce

5.2.1 Komunikace

Komunikace probíhá po již zmíněné síti jako komunikační prostředek je u tohoto modelu zvolen protokol MQTT(viz. 2.1.9). Především kvůli nízké velikosti jednotlivých požadavků. Jako řídící jednotka komunikace MQTT slouží Raspberry PI se softwarem Eclipse Mosquitto. Eclipse Mosquitto jde o open-source službu, která poskytuje funkci MQTT brokera. Broker přijímá zprávy v podobě topic (tématu) a hodnoty pokud se klient chce přihlásit k získávání informací od určitého zařízení pošle název topicu brokerovi a ten mu následně odesílá aktuální informace.

Pro zobrazení informací uživateli je vybudováno uživatelské rozhraní pomocí nástroje Node-Red. Toto rozhraní odebírá informace o senzorech z MQTT a posílá uživatelské požadavky (jako např. zapnutí světla) koncovým zařízením. Zároveň dochází k synchronizaci stavu při kontrole pomocí fyzických tlačítek.

5.2.2 Koncová zařízení

Chytrý ventilátor

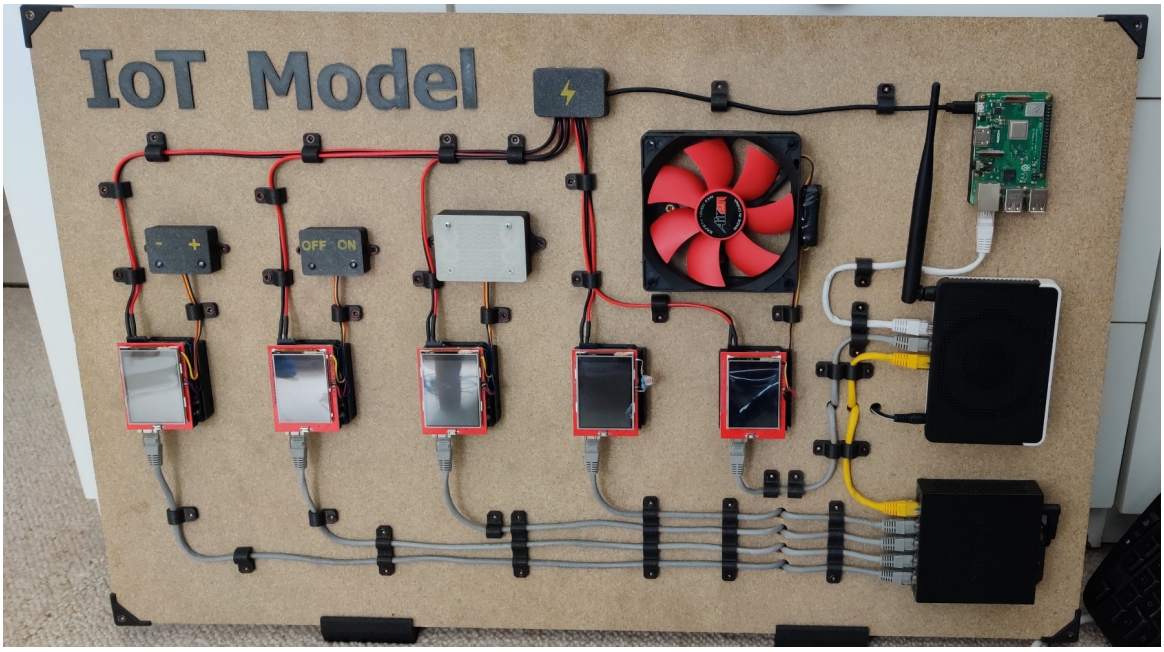
Jde o 120x120mm počítačový chladič, který slouží jako ukázka ventilátoru. Je možné jej ovládat v šesti stupních intenzity otáček, přičemž první stupeň je stav vypnuto. Lze jej ovládat ze tří míst pomocí webového rozhraní, z dotykového displaye nebo pomocí dotykových desek.

Chytré osvětlení

Demonstruje možnost ovládání chytrého osvětlení za pomoci webového rozhraní a dotykového displaye.

Senzor osvětlení

Toto zařízení slouží k získání informací o aktuálním stavu intenzity osvětlení v místnosti.



Obrázek 5.1: Obrázek HW modelu

Kapitola 6
Návrh a Implementace

Cílem projektu je navrhnout a implementovat aplikaci pro správu modelu chytré domácnosti, který byl dodán vedoucím práce. Aplikace bude sloužit jako pomůcka při výuce. Tato kapitola je zaměřena na vývoj aplikace.

6.1 Backend

Celý backend je vytvořen v prostředí Node-Red, který běží na serveru NodeJS.

6.1.1 Přihlašovací systém

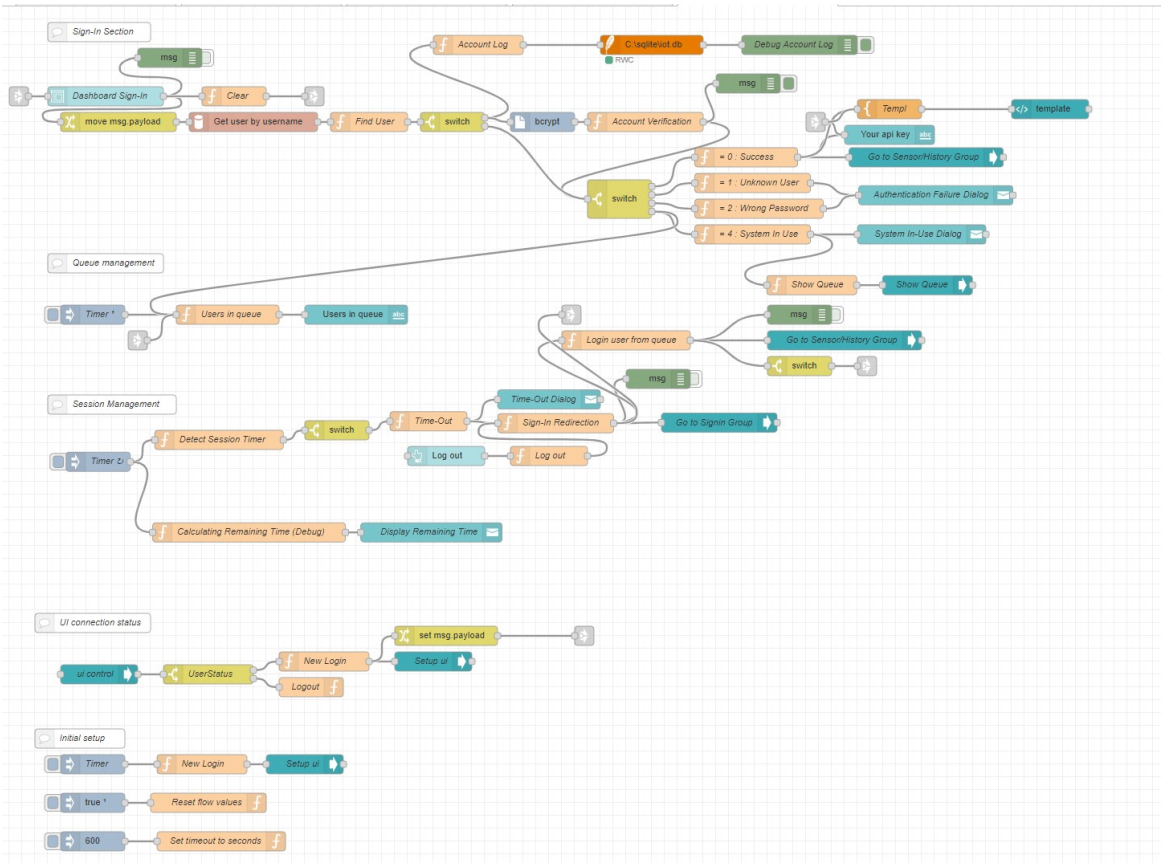
Protože aplikace bude v prostředí studentů a nebude v soukromé síti je zapotřebí nějaký systém pro přístup více uživatelů. Byl vytvořen systém pro přihlašování uživatelů, který umožňuje pouze jednomu z přihlášených uživatelů přístup k ovládání prvků modelu.

Systém pracuje na principu fronty (anglicky "Queue"), což znamená, že pokud jeden uživatel je aktuálně přihlášen a pracuje se senzory modelu nově příchozí uživatel bude zařazen do fronty, kde čeká dokud se předchozí uživatel neodhlásí. Přihlášený uživatel dostává desetiminutový časový limit, po jeho uplynutí je automaticky odhlášen, aby nedocházelo k blokování systému. Dalším opatřením je kontrola aktivity uživatele, frontend udržuje aktivní spojení s backendem pokud je toto spojení ukončeno, je uživatel automaticky odhlášen a přístup získává následující uživatel ve frontě.

Zároveň každý připojený uživatel získává API klíč, který je vytvořen pro každé spojení. Tento klíč slouží jako identifikace uživatele pro danou relaci. Pokud dojde k přihlášení k jednomu uživatelskému účtu z více míst, každému uživateli je přidělen vlastní klíč a jsou bráni jako rozdílní uživatelé. Umožňuje mu přístup k funkcím API jako je ovládání modelu (pouze pokud je daný uživatel přihlášen a není ve frontě). Pokud je přihlášený uživatel administrátor, má právo přes API přidávat nové uživatele, mazat stávající či je upravovat.

6.1.2 Databáze

Pro ukládání data byla zvolena databáze SQLite. Především kvůli své jednoduchosti a praktičnosti. Jednou z nevýhod databáze je její způsob práce



Obrázek 6.1: Flow pro přihlášení

s více uživatelskými přístupy. Pokud do databáze uživatel zapisuje je po dobu zápisu uzamčena. To znamená, že pokud by se další uživatel pokusil o zápis ve stejnou dobu jeho SQL dotaz by vyhodil chybovou hlášku. V aktuální implementaci projektu to není problém pokud se ovšem v budoucnu bude s projektem pracovat je možné, že bude nutné databázi změnit.

Databáze obsahuje následující tabulky users, flows a accountlog (model obrázek 6.2). Do tabulky users se ukládají informace o uživatelích. Ukládá se jejich uživatelské jméno, zašifrované heslo, sůl, datum vzniku účtu a informace zda jde o administrátorský účet nebo uživatelský. Tabulka accountlog slouží pro záznam historie přihlášených uživatelů a adres, ze kterých se přihlásili. Poslední tabulka slouží k ukládání prostředí blockly, které si uživatel vytvořil.

6.1.3 Application programming interface

API (Application programming interface) je označné pro rozhraní. V této aplikaci jde o webové rozhranní, které naslouchá HTTP požadavkům na adrese (/api/...). Rozhraní je rozděleno do několika sekcí pomocí adres:

- Blockly (/api/Blockly/...) - komunikace s blockly
- Users (/api/user/...) - práce s uživatelskými účty

- Devices (/api/devices/...) - práce s připojenými zařízeními

6.2 Blockly

6.2.1 Co je Promise

Jde o jeden ze základních způsobů programování asynchronního chování v javascriptu, ve kterém je Blockly napsáno. Promise objekt funguje jako zástupce ("příslib") budoucí hodnoty, kterou metoda jednou vrátí. S tímto zástupcem můžeme dále pracovat, je nezávislý na čase.

Promise objekt může mít tři stavy:

- Splněný: funkce vložená jako parametr pro splnění je zavolána
- Odmítnut: došlo k chybě nebo odmítnutí od serveru
- Probíhá: nedošlo k zamítnutí ani dokončení

6.2.2 Asynchronní úpravy

Pro spojení blockly a Node-RED slouží API (Více informací v sekci 6.1). Aby bylo možné používat toto API, bylo zapotřebí do knihovny blockly přidat bloky, které budou podporovat asynchronní chování. Blockly v základu funguje na plně synchronní bázi. Pokud by bylo v tomto trendu pokračováno a zachován systém komunikace přes API, znamenalo by to, že u každého dotazu (např. dotaz na intenzitu světla z čidla) by se čekalo na odpověď od serveru a celé webové rozhraní by bylo nepoužitelné (zamrzlé) do doby, než by odpověď přišla zpět. Aby bylo zamezeno této situaci, rozhodl jsem se předělat všechny bloky, se kterými uživatel bude pracovat na asynchronní za pomocí Promise (viz 6.2.1).

6.2.3 Implementace grafických bloků

Každý blok má definován vzhled, generátor a své vstupy. U každého bloku je také definováno, jaké typy bloků může obsahovat. To zabraňuje chybnému použití bloků, aby se například blok podprogramu s návratovou hodnotou text nemohl zanořit do bloku pro odmocninu a nebo aby se do bloku požadujícího číslo nevnořil blok pro cyklus. Každý grafický blok má v knihovně Blockly své vstupy, do kterých se zanořují další bloky. Rozlišujeme tři základní typy vstupů:

- `ValueInput` - Vstupem je blok, který vrací hodnotu
- `StatementInput` - vstupem je sekvence vnořených bloků
- `DummyInput` - vstupem není blok, ale vstupní prvek jako textové pole, výběrový seznam,...

Každý tento vstup má své jméno v rámci jednoho bloku musí být jména vstupů unikátní. Pomocí těchto jmen k datům na vstupu následně přistupuje generátor kódu.

Generátor kódu z celé sekvence bloků vygeneruje kód v textové podobě v tomto případě je vzniklý kód v jazyce javascript. Bloky jsou procházeny postupně a pomocí příkazu `statementToCode` je na každém bloku zavolán jeho generátor, ten pokud blok má na svých vstupech další bloky, volá opět jejich generátor, až vzniká celý kód.

V následující části bude popsáno několik bloků pro návrh jejich vzhledu byl využit nástroj Blockly Developer Tools[1]. Popis se zaměří na část generátoru.

Bloky pro práci se senzory

Do prostředí Blockly byly přidány bloky pro získávání dat z senzorů na modelu. Tyto bloky se dotazují přes API na aktuální stav senzorů. Tento dotaz je posílán s aktuálním API klíčem přihlášeného uživatele za pomoci javascriptového fetch, který již sám o sobě vrací Promise. Dotaz jde přes HTTP GET (HTTP v sekci 2.1.10).

Další jsou bloky, které nastavují stav zařízení na modelu. Tyto bloky, podobně jako předchozí, používají API klíč uživatele. Zde je opět použit fetch, tentokrát se v dotazu posílají data o novém nastavení zařízení. Server následně posílá v odpovědi informace o stavu (v pořádku/ s chybou) předání nového nastavení.

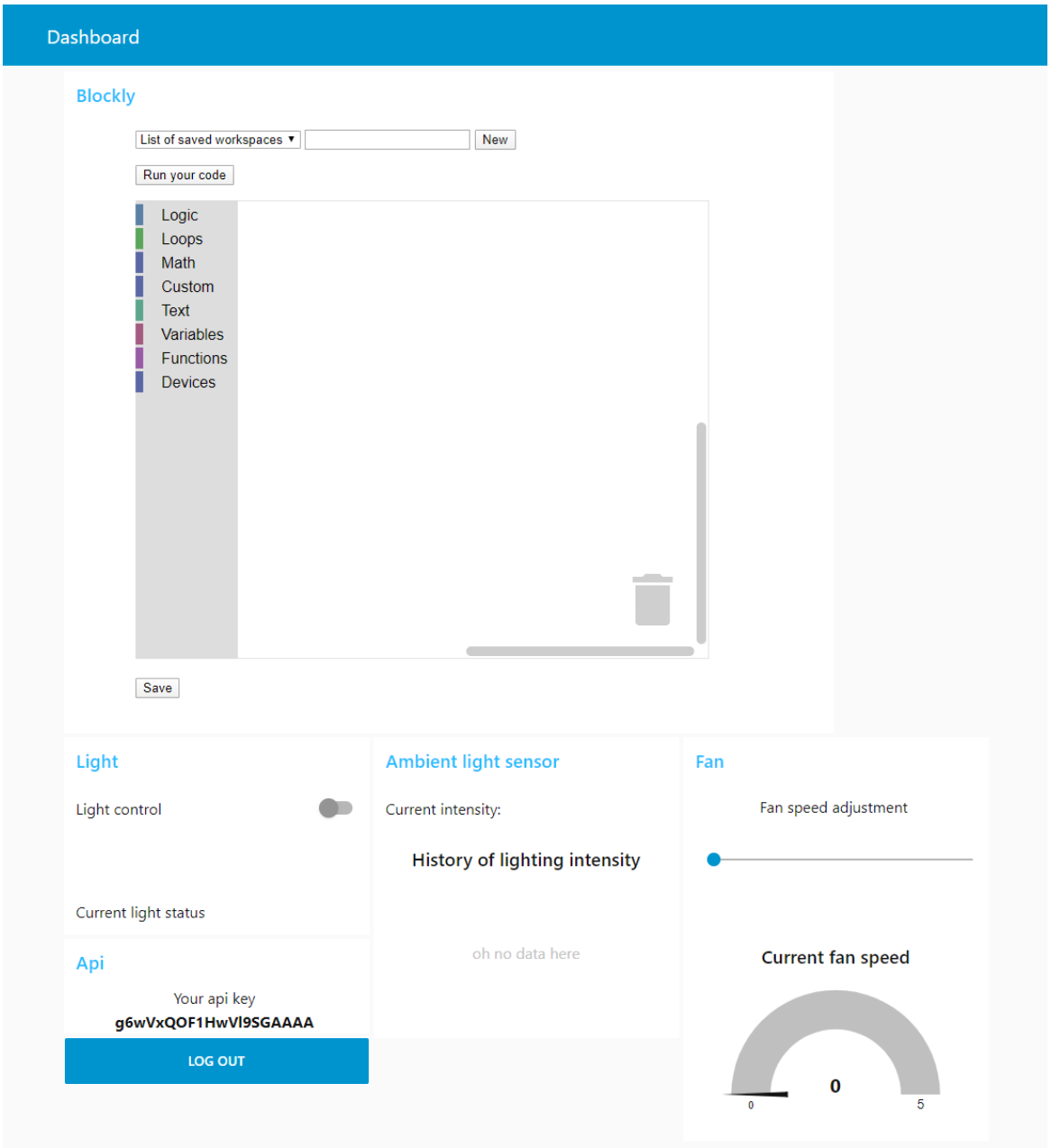
6.2.4 Ukládání stavu prostředí Blockly

Do prostředí Blockly byla přidána funkce umožňující uživateli, aby si svou hotovou či rozpracovanou aplikaci uložil. Ukládání funguje následovně - uživatel si zvolí název pod kterým si svou práci chce uložit. Při ukládání je do JSON objektu uložen stav bloků v prostředí tento objekt je následně převeden do textové podoby. Ten je pomocí API klíče odeslán na server, kde dojde ke spojení klíče a uživatelského účtu a data o prostředí jsou následovně uložena do databáze.

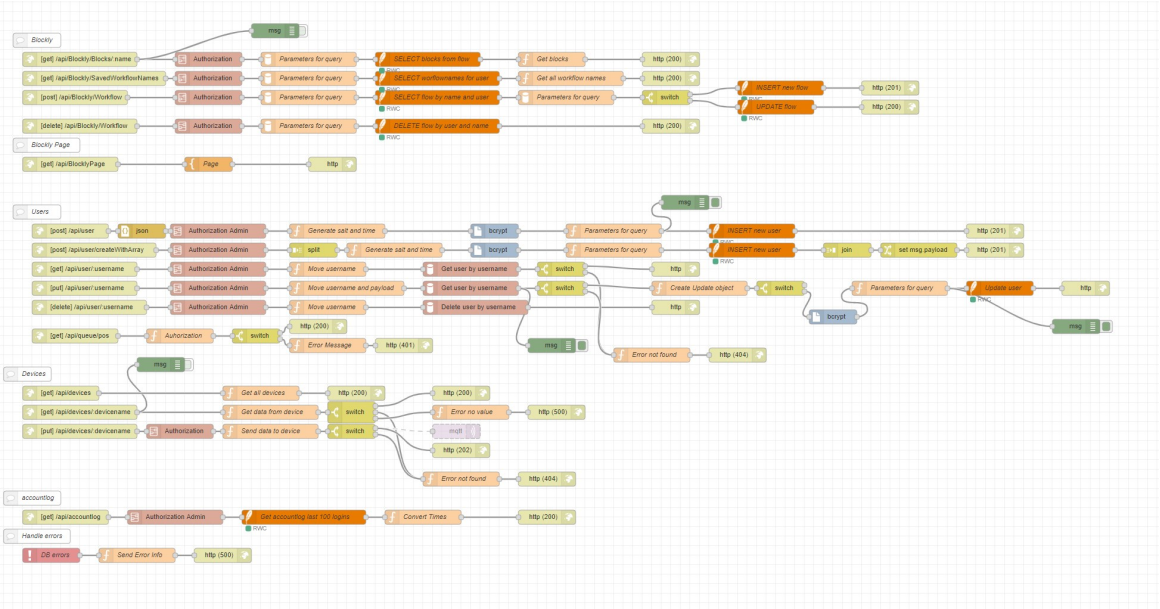
Načítaní existujících uložených prostředí funguje na podobném principu. V podobě rolovací nabídky jsou uživateli zobrazeny názvy prostředí, které má uloženy na serveru. Pokud uživatel zvolí z nabídky jedno z uložených prostředí je odeslán na server dotaz. Server po přijetí dotazu z databáze načte textový JSON, který odešle zpět klientovi. Na straně klienta se tento text převede na JSON objekt a je následně nahrán do prostředí, kde dochází k načtení uložených bloků.

[1] `https://blockly-demo.appspot.com/static/demos/blockfactory/index.html`

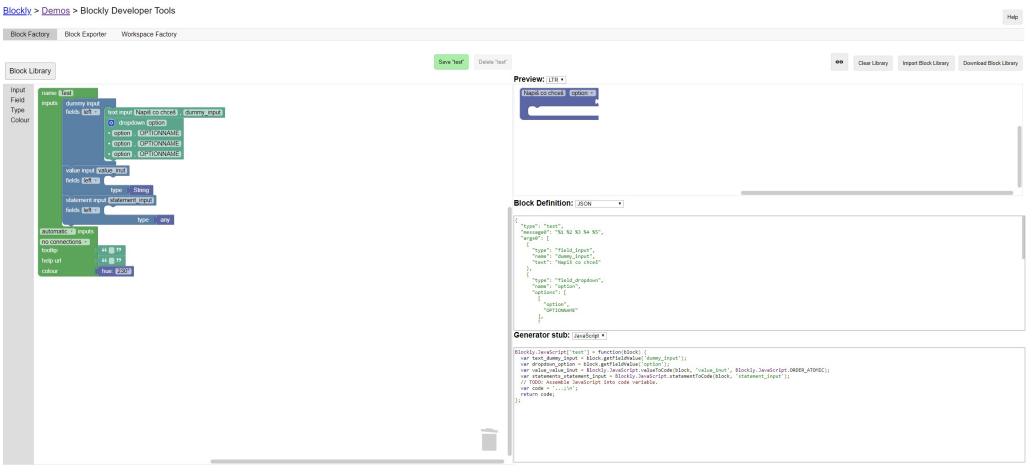
Obrázek 6.2: Schéma databáze



Obrázek 6.3: Ukázka uživatelského rozhraní



Obrázek 6.4: API flow z Node-Red



Obrázek 6.5: Ukázka Blockly Developer Tools (se všemi typy vstupů pro blok)

Kapitola 7
Bezpečnost

Tato kapitola se zabývá zabezpečením konkrétního řešení aplikace pro chytrou domácnost popsanou v předchozí kapitole.

7.1 Databáze

Aby se útočník mohl dostat k datům uloženým v databázi musí získat přístup k systému souborů. Vzhedem k použití jednoduché databáze SQLite databáze neobsahuje žádný šifrovací algoritmus a jediné co tedy útočník musí získat je databázový soubor. Ale i pokud by se útočník dostal k databázi a získal z ní data, nezíská hesla uživatelů, která jsou solená[1] a šifrována. Šifrování probíhá pomocí hashovací funkce Bcrypt.

Zároveň jsou veškeré dotazy připraveny (prepared statement), které používají uživatelský vstup dat (ať z api či z formuláře). Výhodu těchto připravených dotazů z hlediska bezpečnosti spočívá v tom, že jsou odolná vůči tzv. SQL injection. Jde o útok, který se za použití uživatelského vstupu na serveru snaží spustit vlastní sql příkaz. Útočník do například uživatelského jména vloží SQL dotaz a je jasné, že pokud server používá databázi, dotaz pro nalezení daného uživatele musí vypadat následovně:

```
"SELECT * FROM uzivatele WHERE jmeno = '" + zadaneJmeno + "';"
```

Pokud by do takového dotazu v proměnné "zadaneJmeno"byla hodnota:

```
a';sql útočníka;
```

Pokud dotaz není chráněn, útočník může na serveru spouštět vlastní příkazy, ke kterým má daný server práva. V případě SQLite existuje pouze administrátorský účet a je tedy nutné příkazy proti těmto útokům chránit. Připravené dotazy proti tomuto útoku brání tím způsobem, že veškeré speciální znaky jsou brány pouze jako text a ne jako součást dotazu. V případě výše uvedeném by se tedy hledal uživatel se jménem "a';sql útočníka;"a nedošlo by ke spuštění útočníkova dotazu.

[1] Kryptografická sůl jde o náhodné byty, které jsou generovány při ukládání hesla a stěžují jeho odhalení

Obrázek 7.1: Snímek socketové komunikace (konkrétně přihlášení) zachycený pomocí aplikace wireshark

7.2 Komunikace

7.2.1 Dashboard a server

Komunikace zde probíhá pomocí Síťových socketů. Při každém otevření webového rozhraní v nového okně webového prohlížeče je vytvořen nový komunikační kanál speciálně pro komunikaci s touto relací. Tento kanál je následně použit pro aktualizaci informací o jednotlivých senzorech k odeslání informací o akcích uživatele (například stisk tlačítka). Zároveň je pro každý tento komunikační kanál vytvořen unikátní klíč, který je následně možné použít i pro komunikaci pomocí protokolu http se serverovým API rozhraním.

Socketová komunikace není zabezpečená a je možné ji číst jak je vidno z obrázku 7.1. Tento bezpečnostní problém by vyřešilo požití protokolu HTTPS.

7.2.2 Šifrování pomocí HTTPS

Jak již bylo zmíněno velký rozdíl by vytvořilo šifrování dat pomocí HTTPS, které je podporováno serverem. Na straně serveru je potřeba nahrát X.509 certifikát, který slouží jako ověření (Více informací v sekci 2.1.10). Raspberry pi disponuje dostatečným výkonem a Node-Red nativně HTTPS podporuje, stačí tedy v nastavení odkázat na soubor s certifikátem, který Node-red následně použije pro šifrování komunikace. Bude tedy tímto způsobem šifrována komunikace přes API i socketová komunikace v dashboardu. Tím bude výrazně sníženo riziko odposlechu těchto dvou komunikačních kanálů.

7.2.3 Šifrování dat

Další úrovní v ochraně dat je šifrování přímo obsahu odesílaného z klienta. Javascript běžící na straně má jistou nevýhodu jeho kód je k dispozici libovolnému uživateli. Používání tedy šiforvacích metod jako AES, DES, RC4 apod. není ideální, tyto metody používají stejný klíč k šifrování a dešifrování. Tento klíč by bylo možné z kódu u klienta jednoduše přečíst. Proto je nutné šifrovat komunikaci na bázi veřejného a privátního klíče. V takové komunikaci je klientská správa zašifrována pomocí veřejného klíče serveru a ten následně používá svůj privátní klíč k dešifrování. Jde o stejný způsob šifrování jako

v případě HTTPS. K dosažení toho výsledku by bylo možné použít knihovnu jako TweetNaCl.js[2] nebo JSBN[3]. Tuto metodu je možné implementovat bez změn HW.

V případě komunikace jednotlivých zařízení s brokerem by implementace šifrování komunikace znamenala nutnost výměny desky arduino Ethernet. Z důvodu nedostatečného výkonu procesoru na deskách.

7.2.4 Blockly

Ke komunikaci používá HTTP požadavky spolu s unikátním klíčem, o kterém je zmínka v minulé sekci. Tento klíč slouží ke kontrole uživatele na serveru. Komunikace je nešifrována a je možné ji odposlouchávat opět by se dalo vyřešit pomocí HTTPS.

[2] `https://github.com/dchest/tweetnacl-js/blob/master/README.md#documentation`

[3] `http://www-cs-students.stanford.edu/~tjw/jsbn/`

Kapitola 8

Závěr

Cílem této práce bylo navrhnout a implementovat aplikaci pro ovládání inteligentní domácnosti. Zároveň má aplikace sloužit jako pomůcka při výuce pro ukázku grafického programování, které bylo zvoleno jako dobrý nástroj pro výuku základů programování. Tato aplikace byla vložena na model chytré domácnosti.

Úvodní kapitola čtenáři přibližuje co pojem Internet věcí znamená a jak je s ním spojena chytrá domácnost. Tato část zdůrazňuje důležitost Internetu věcí a jeho rozsáhlé možnosti pro tvorbu budoucích systémů.

Dále práce pojednává o technologiích, které jsou pro komunikaci v rámci chytré domácnosti a Internetu věcí používány. Následuje přehled dostupných řešení pro správu chytrých domácností. V přehledu jsou rozebrána řešení dostupná od nadnárodních korporací a zároveň řešení, která nabízí každému vytvořit si vlastní systém a to je i cesta, kterou se ubírám v této práci.

V páté části práce je analyzován dodaný model chytré domácnosti. Analýza se zabývá jak hardwarovou stránkou modelu tak jeho softwarovou stránkou. Na základě veškerých získaných informací je vytvořen návrh aplikace. V této části práce je rozebrána zvlášť klientská a serverová stránka práce. V serverové části je popsán systém ukládání dat, přihlašovací systém řešený pomocí přihlašovací fronty, kdy vždy pouze jeden uživatel má přistup k ovládání modelu. A v poslední serverové části je rozepsáno rozhraní API. V klientské části je popsána tvorba uživatelského rozhraní a využití prostředí Blockly a jeho spojení přes již zmíněné API se serverem.

Poslední část práce se zaobírá bezpečností celého řešení. Konkrétně je probráno zabezpečení uložených dat a šifrování komunikace mezi serverem a klientem.

Výsledkem je aplikace umožňující ovládání chytré domácnosti pomocí webového rozhraní i za pomoci programovacího prostředí Blockly.

Literatura

[1] Internet of Things [online]. Gartner, Inc. [21.05.2019] Dostupné z: http://www.gartner.com/it-glossary/internet-of-things

[2] The Internet of Things: In a Connected World of Smart Objects [online]. Fundación de la Innovación Bankinter. 2011.

[3] Pavel Pohanka - Internet věcí. [online]. 2015. [14.05.2019] Dostupné z: http://i2ot.eu/internet-of-things/

[4] Raspberry Pi Foundation. Raspberry Pi 3 model B+ product brief, [23.11.2019] Dostupné z: https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf

[5] Tim Taberner, Rozdíl mezi M2M a IoT. [21.05.2019] Dostupné z: https://www.cad.cz/strojirenstvi/38-strojirenstvi/6972-rozdil-mezi-m2m-a-iot.html

[6] Gartner IT Glossary [online]. Gartner, Inc. and/or its Affiliates, © 2017. [cit. 21.05.2019]. [20.05.2019] Dostupné z: http://www.gartner.com/it-glossary/?s=internet+of=things

[7] Schéma chytré domácnosti z: https://www.louver.cz/wp-content/uploads/automatizace-domacnosti.png [20.12.2019]

[8] SPURGEON, C. E. Ethernet: The Definitive Guide. 1. vydání. Sebastopol: O'Reilly & Associates, 2000. ISBN 1-56592-660-9.

[9] Understanding Z-Wave Networks, Nodes & Devices. Vesternet [online]. [23.12.2019] Dostupné z http://www.vesternet.com/resources/technology-indepth/understanding-zwave-networks

[10] FIBARO - systém chytré domácnosti. [25.12.2019] Dostupné z: https://www.mojefibaro.cz/

[11] Apple HomeKit. [25.12.2019] Dostupné z: https://www.imore.com/homekit-faq

[12] Kestřánek, Tomáš. *IoT technologie pro automatizaci domácnosti*. Praha, 2019. Bakalářská Práce. České vysoké učení technické v Praze.

[13] STRICKLAND, Jonathan. How will the Internet of Things communicate?. In: Fw:Thinking [online]. 2014. [31. 12. 2019] Dostupné z: http://www.fwthinking.com/blog/how-will-the-internet-of-things-communicate/

[14] HOFFMAN, CHRIS. Wi-Fi 6: What's Different, and Why it Matters. [29.12.2019] Dostupné z: https://www.howtogeek.com/368332/wi-fi-6-what%E2%80%99s-different-and-why-it-matters/

[15] ĎAĎO, Stanislav a Marcel KREIDL. Senzory a měřicí obvody. Praha: České vysoké učení technické, 1996. ISBN 80-01-01500-9

[16] Martin Malý, Protokol MQTT: komunikační standard pro IoT. [3.1.2020] Dostupné z: https://www.root.cz/clanky/protokol-mqtt-komunikacni-standard-pro-iot/

[17] UNIP TECHNOLOGI - Node-RED. [20.12.2019] Dostupné z: https://www.unipi.technology/cs/produkty/node-red-66

[18] Cypress Semiconductor, What is Power Line Communication? [4.1.2020] Dostupné z: https://www.eetimes.com/what-is-power-line-communication/

[19] Bluetooth Vs. Bluetooth Low Energy: What's The Difference? [online]. LinkLabs. 2015. [4.1.2020] Dostupné z: http://www.link-labs.com/bluetooth-vs-bluetooth-lowenergy/

[20] GLOVER, Bill a Bhatt HIMANSHU. RFID essentials. 1st ed. Beijing: O'Reilly, 2006, xiii, 260 s. ISBN 05960-0944-5

[21] HTTP (HyperText Transfer Protocol). [3.1.2020] Dostpuné z: https://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html

[22] Rescorla, E.; Schiffman, A.; aj.: The Secure HyperText Transfer Protocol. [online], srpen 1999, [3.1.2020]. Dostupné z: https://tools.ietf.org/html/rfc2660

Příloha - Obsah CD

- **/api/html/** - HTML dokumentace API
- **/api/api.json** - JSON soubor se strukturou API
- **/api/Node RED - Smart home.json** JSON soubor s příklady API dotazu pro program Postman
- **/blockly/** - Upravený kód knihovny Blockly
- **F3-BP-2020-Svagr-Petr-Aplikace pro správu chytré domácnosti.pdf** - PDF dokument s touto prací
- **flows.json** - exportovaný program pro Node-RED
- **Readme.md** - krátké readme s více informacemi k přiloženým dokumentům v textové podobě
- **Readme.pdf** - krátké readme s více informacemi k přiloženým dokumentům ve formátu pdf