

System pro vzdálené řízení otopných soustav rodinného domu

Jaroslav Kadleček

Bakalářská práce
2015



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2014/2015

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Jaroslav Kadleček**
Osobní číslo: **A12259**
Studijní program: **B3902 Inženýrská informatika**
Studijní obor: **Informační a řídicí technologie**
Forma studia: **kombinovaná**

Téma práce: **Systém pro vzdálené řízení otopných soustav rodinného domu**
Téma anglicky: **A Boiler Remote Control System for a Domestic House**

Zásady pro vypracování:

1. Analyzujte a vyberte vhodné technologie pro implementaci vzdáleného ovládání kotle ústředního topení. Uvažujte PLC nebo embedded zařízení s rozhraním pro vzdálený přístup a moduly pro měření teploty a ovládání kotle.
2. Sestavte schéma zapojení celého systému, včetně zapojení embedded počítače a senzorů.
3. Realizujte a otestujte funkčnost tohoto zapojení.
4. Navrhněte architekturu SW části systému, včetně rozhraní pro mobilní zařízení.
5. Implementujte SW část a proveďte její bezpečnostní audit.

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **MERZ, Hermann, Thomas HANSEMANN a Christof HÜBNER. Automatizované systémy budov: sdělovací systémy KNX/EIB, LON a BACnet. 1. vyd. Praha: Grada, 2008, 261 s. ISBN 80-01-01981-0.**
2. **UHLÍŘ, Jan, Přemek NEUMANN a Christof HÜBNER. Elektronické obvody a funkční bloky: sdělovací systémy KNX/EIB, LON a BACnet. Vyd. 1. Praha: Vydavatelství ČVUT, 1999, vi, 270 s. ISBN 80-010-1981-0.**
3. **MARIE MARTINÁSKOVÁ, Ladislav Šmejkal, Přemek NEUMANN a Christof HÜBNER. Řízení programovatelnými automaty: sdělovací systémy KNX/EIB, LON a BACnet. Vyd. 1. Praha: ČVUT, 1998, vi, 270 s. ISBN 80-010-1766-4.**
4. **DOSEDĚL, Tomáš, Přemek NEUMANN a Christof HÜBNER. Počítačová bezpečnost a ochrana dat: sdělovací systémy KNX/EIB, LON a BACnet. Vyd. 1. Brno: Computer Press, 2004, 190 s. ISBN 80-251-0106-1. 8025101061**
5. **PFISTER, Cuno, Přemek NEUMANN a Christof HÜBNER. Getting started with the Internet of things: sdělovací systémy KNX/EIB, LON a BACnet. 1st ed. Sebastapool, Calif.: O'Reilly Media, Inc., 2011, xiii, 176 p. Make: projects. ISBN 14-493-9357-8.**

Vedoucí bakalářské práce:

Ing. Michal Bližňák, Ph.D.

Ústav informatiky a umělé inteligence

Datum zadání bakalářské práce:

6. března 2015

Termín odevzdání bakalářské práce:

22. května 2015

Ve Zlíně dne 6. března 2015



L.S.

doc. Mgr. Milan Adámek, Ph.D.
děkan

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové/bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně 21.5.2015

.....
Kadlec
.....
podpis diplomanta

ABSTRAKT

Tématem této práce je vytvoření systému pro vzdálené řízení otopných soustav rodinného domu. Důraz je kladen na jednoduché uživatelské rozhraní a bezpečnost celého systému z pohledu odolnosti proti náhodným poruchám a proti neoprávněným přístupům. V teoretické části byla provedena analýza stávajících dostupných řešení na trhu splňující daná kritéria a byla zvolena vhodná varianta. V praktické části byl proveden návrh zapojení komponent zvoleného řešení, analýza a návrh programového vybavení. Byla provedena implementace embedded řídicího panelu s dotykovým displejem, veřejné API rozhraní a responsivní webové uživatelské rozhraní pro vzdálený přístup. V závěru byla provedena analýza odolnosti proti bezpečnostním rizikům a poruchám. Výsledkem práce bylo vytvoření komfortního ovládacího panelu pro otopné soustavy, které uživatel může ovládat jak na panelu přímo v domě, tak na dálku z www klienta.

Klíčová slova: Otopná soustava, PLC, vzdálené ovládání, uživatelské rozhraní, distribuované řízení, .NET

ABSTRACT

The subject of this theses is to create a remote control system for heating a house. Emphasis is placed on a simple user interface and safety of the entire system from the viewpoint of resistance against accidental failures and unauthorized access. In the theoretical part were analyzed existing solutions available on the market that satisfied the criteria and was chosen the appropriate solution. In the practical part was created a draft of a wiring components, analysis and draft for software implementation. Implementation was done for the embedded control panel with touch screen, public API interface and responsive Web-based user interface for remote access. The work also includes analysis of resistance against security risks and system failures. Result of this work was to create a comfortable control panel for the heating system. User can use a control panel in the house, and remote control panel from a web client.

Keywords: Heating System, PLC, Remote Control, User Interface, distributed control, .NET

Chtěl bych poděkovat svému vedoucímu diplomové práce Ing. Michalu Bližňákovi, Ph.D. za odbornou pomoc, vedení a poskytnutí potřebných informací při tvorbě bakalářské práce. Rovněž bych chtěl ještě poděkovat rodině za morální podporu v průběhu celého studia na vysoké škole.

OBSAH

ÚVOD	9
I TEORETICKÁ ČÁST	10
1 ANALÝZA VHODNÉ TECHNOLOGIE	11
1.1 KOMUNIKACE.....	11
1.1.1 PLC, senzory a relé	11
1.1.2 Řídící panel a PLC	12
1.1.3 Vzdálení klienti a server.....	13
1.2 VOLBA VHODNÉHO HARDWARE.....	13
1.2.1 Řešení Tecomat	14
1.2.2 Řešení Papouch	14
1.2.3 Řešení GHI Electronics	15
1.3 ZÁVĚR.....	16
II PRAKTICKÁ ČÁST	17
2 SCHÉMA ZAPOJENÍ HARDWARE	18
2.1 TECHNICKÝ POPIS POUŽITÉHO HARDWARE	18
2.1.1 Senzory.....	18
2.1.2 Relé.....	18
2.1.3 PLC.....	19
2.1.4 Řídící panel	20
2.1.5 Server	21
2.2 CELKOVÉ SCHÉMA.....	21
2.3 DETAIL ZAPOJENÍ PLC, SENZORŮ A RELÉ	22
2.4 DETAIL ZAPOJENÍ ŘÍDÍCIHO PANELU	23
3 NÁVRH SW ČÁSTI	25
3.1 POPIS PŘÍPADŮ UŽITÍ	25
3.1.1 Vzdálený přístup přes webové rozhraní	25
3.1.1.1 Zobrazení stavu systému.....	25
3.1.1.2 Autentizace nepřihlášeného uživatele.....	25
3.1.1.3 Změna programu.....	26
3.1.1.4 Ruční korekce požadované teploty	26
3.1.1.5 Detailní nastavení programu	27
3.1.2 Lokální přístup přes řídicí panel.....	28
3.1.2.1 Zobrazení stavu systému na řídicím panelu.....	28
3.1.2.2 Změna zobrazení stavu	28
3.1.2.3 Změna programu.....	29
3.1.2.4 Ruční korekce požadované teploty	29
3.1.3 Periodická aktualizace údajů	30
3.2 VOLBA KOMUNIKAČNÍ TECHNOLOGIE KLIENT-KLIENT, KLIENT-SERVER.....	30
3.2.1 Azure mobile services	31
3.2.2 X.Sockets.NET	31
3.2.3 SignalR	32
3.2.4 Závěr.....	32

3.3	SDÍLENÉ STRUKTURY PRO PŘENOS DAT	32
3.4	DATABÁZE	35
3.5	SERVEROVÁ ČÁST - API	37
3.5.1	Komunikace	37
3.5.2	Autentizace	38
3.6	WEBOVÉ ROZHRAŇÍ	38
3.6.1	Přihlašovací obrazovka	39
3.6.2	Hlavní obrazovka	39
3.6.3	Nastavení PLC	40
3.6.4	Detailní nastavení programu, plánovací kalendář	41
3.6.5	Nastavení detailu položky programu v kalendáři	42
3.7	MOBILNÍ KLIENT	43
3.7.1	Webové rozhraní	43
3.7.2	Nativní aplikace	44
3.8	ŘÍDÍCÍ PANEL	45
3.8.1	Příprava fontů s podporou české diakritiky a speciálních znaků	45
3.8.2	Návrh obrazovek	45
3.8.2.1	Hlavní obrazovka	46
3.8.2.2	Výběr a nastavení programu	47
4	IMPLEMENTACE	49
4.1	VÝVOJOVÉ PROSTŘEDÍ	49
4.1.1	Emulátor řídicího panelu	49
4.1.2	Unittesty	50
4.1.3	Emulátor serverové části	50
4.1.4	Řídicí panel	50
4.1.5	Webové rozhraní	50
4.1.6	Mobilní klient webový	50
4.2	IMPLEMENTACE API SERVERU	50
4.3	IMPLEMENTACE WEBOVÉHO ROZHRAŇÍ	52
4.3.1	Přihlašovací formulář	52
4.3.2	Hlavní obrazovka	52
4.3.3	Nastavení programu	52
4.3.4	Nastavení položek programu v kalendáři	53
4.4	IMPLEMENTACE ŘÍDÍCÍHO PANELU	53
4.4.1	Implementace knihovny klienta SignalR pro NETMF	53
4.4.2	Implementace knihovny pro komunikaci s PLC	54
4.4.3	Připojení k API	54
4.4.4	Displej	55
5	BEZPEČNOSTNÍ AUDIT	56
5.1	BEZPEČNOST KOMUNIKACE	56
5.2	ODOLNOST PROTI SQL INJECTION	56
5.3	ODOLNOST PROTI NEOPRÁVNĚNÉMU PŘÍSTUPU	57
5.4	ODOLNOST PROTI DDoS ÚTOKU	57
5.5	ODOLNOST PROTI VÝPADKŮM A PORUCHÁM	58
5.5.1	Senzory	58

5.5.2	PLC.....	58
5.5.3	Relé.....	58
5.5.4	Řídící panel	59
5.5.5	Server API.....	59
ZÁVĚR		60
SEZNAM POUŽITÉ LITERATURY.....		61
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....		64
SEZNAM OBRÁZKŮ		65
SEZNAM TABULEK.....		66

ÚVOD

Od pradávna bylo snahou člověka si život co nejvíce zpříjemnit a zjednodušit, což se projevuje ve všech oblastech lidské činnosti. Moderní technologie a výpočetní technika se používala zejména v průmyslu, ale nyní se již začínají objevovat i u rodinných domů. Levná zařízení byla bez displeje a uměla pouze základní funkci udržet teplotu dle programu. Nároky uživatele na komfort se ale postupně zvyšují a tak je potřeba aplikovat stále lepší a propracovanější systémy.

Cílem této práce je vylepšit stávající systém otopné soustavy tak, aby byla obsluha ovládacího panelu pro uživatele co nejpříjemnější a aby byla možnost bezpečného a jednoduchého ovládání na dálku z libovolného uživatelského zařízení. Pokud dojde k problému v systému, tak musí být uživatel včas informován, aby nedošlo z důvodu chybné funkce k poškození majetku.

V první části této práce byla provedena analýza požadavků na systém a volba vhodných technologií a komponent. V druhé části byl pak proveden návrh architektury a implementace řešení. V závěru pak jsou bezpečnostní testy a zhodnocení dosažených výsledků.

I. TEORETICKÁ ČÁST

1 ANALÝZA VHODNÉ TECHNOLOGIE

Nejprve je potřeba si ujasnit čeho je potřeba dosáhnout. Tedy je potřeba měřit teplotu na několika místech v budově i mimo ni a na základě změřených údajů pak spínat otopnou soustavu a solární ohřev. Pro návrh je možné využít stávající strukturovanou kabeláž, která je v domě rozvedena. Vnitřní síť je připojena k internetu přes malý router, který má v sobě firewall a DHCP server.

1.1 Komunikace

Komunikační kanály se z pohledu bezpečnosti dělí na dvě části. Vnitřní kanály v rámci domu mohou být považovány za zabezpečené. Pak je komunikace vzdáleného přístupu, která bude přes veřejnou síť. Tato část musí být zabezpečena.

1.1.1 PLC, senzory a relé

Velmi kritickou částí je spojení mezi PLC a senzory. Musí zde být minimalizováno riziko výpadku spojení a minimalizováno rušení okolního prostředí na signál sensorů. Z toho důvodu by mělo být toto komunikační vedení co nejkratší.

Napájecí napětí pro relé bude vhodné použít stejné jako napájecí napětí ostatních prvků ve vnitřní síti, aby bylo možné minimalizovat množství napájecích zdrojů. Základní předpoklad pro napájení vnitřních prvků bude jeden zálohovaný napájecí zdroj.

Komunikace mezi senzory a PLC záleží na typu koncového senzoru a může být řešena těmito způsoby (seřazeno sestupně podle pořizovací ceny daného řešení):

- Smart senzor
Odolný proti rušení, autokalibrace, možnost delšího komunikačního vedení, schopnost diagnostikovat chybu čidla na dálku. Nevýhoda vyšší pořizovací cena.
- Digitální senzor
Stejně vlastnosti jako smart senzor. Není k dispozici autokalibrace, vzdálená diagnostika chyby. Nízká pořizovací cena.
- Analogový senzor
Signál je přenášán analogově. Velmi náchylné na rušení. Vzdálenost přenosu informace velmi malá. Velmi nízká pořizovací cena.

Základním požadavkem je bezpečnost systému, tedy není vhodné použít analogové senzory. V konečném důsledku by se prodražilo řešení z důvodu implementace pomocných prvků pro korekci chyb měření. Smart senzor by byl dobrá volba, ale údaje, které se budou měřit, nevyžadují tak přesné měření a údaje nejsou tak kritické, aby to vyvážilo vyšší pořizovací náklady. Vhodným řešením z pohledu na poměr cena/výkon střední cesta, tedy digitální senzor.

1.1.2 Řídící panel a PLC

Řídící panel bude mít funkci uživatelského rozhraní s dotykovým displejem. Bude zobrazovat informace o stavu systému a umožní uživateli nastavení měnit.

Tento panel bude propojen s jednotlivými PLC v rámci distribuovaného řízení a komunikaci je možno řešit těmito způsoby:

- RS232

Nevýhoda, umožňuje spojení pouze bod-bod. Rychlost přenosu je malá, ale pro požadovaný účel dostatečná.

- Sériová linka / CAN Bus

Umožňuje spojit více zařízení. Nevýhoda je nutnost podpory CAN u všech zařízení. Náročnější diagnostika chyb, například při přerušení vedení. Nutno proměřit každý obvod ručně pomocí multimetru, nebo mít na to specializovaný měřicí přístroj.

- Ethernet

Umožňuje spojit více zařízení. Snadná diagnostika chyb každé větve přímo na routeru. Pro diagnostiku chyb a odstranění závad je možné použít stejné nástroje jako pro opravu strukturované kabeláže. Pro přenos je možno použít více protokolů:

- Proprietární protokol pro PLC dané výrobcem

Nevýhodou je obtížná implementace dalšího typu PLC od jiného výrobce do systému.

- protokol SNMP [1]

Jedná se o standardní protokol, většina zařízení jej podporuje. Identifikátory jednotlivých dat jsou strukturovány do MIB stromu, který se ale bohužel u každého výrobce liší. Nevýhodou je poměrně složitá implementace v případě, že nemám k dispozici odpovídající SNMP knihovnu.

- protokol HTTP [2]

HTTP je obecná specifikace přenosu dat, není zde tedy přesná specifikace jak data formátovat, nicméně jsou prosazovány dva standardy pro serializaci

údajů: XML a JSON. Jedná se o textové formáty a zpracování dat. Je tedy velmi jednoduché.

Nejvhodnější je využít stávající rozvody ethernetové sítě. Pro komunikaci mezi PLC a řídicím panelem bude nejvhodnější použít protokol HTTP kvůli jednoduchosti a univerzálnosti. Komunikace po HTTP protokolu se dá velmi jednoduše diagnostikovat běžně dostupnými prostředky.

1.1.3 Vzdálení klienti a server

Tato komunikace bude muset být již zabezpečená. Je zde několik možností řešení:

- SMS přes GSM síť

Výhodou by byla odolnost proti výpadku. Nevýhoda je, že umožňuje posílat pouze omezené množství informací a při velkém počtu zpráv by to byla velmi drahá komunikace.

- Spojení přes internetovou síť

Výhoda je rychlá odezva, dostupnost dat a propustnost spojení. Nevýhoda je, že ne všechny budovy disponují připojením k internetu. Typicky toto spojení nebývá dobře zálohováno a je zde vyšší riziko výpadku.

Po zvážení výhod a nevýhod [3] a skutečnosti, že budova, kde bude systém implementovat má internetovou přípojku, je nejvhodnější technologií internetová síť.

1.2 Volba vhodného hardware

V předchozí kapitole byly zvoleny vhodné komunikační technologie. Podle těchto parametrů tedy budou vybírány i fyzické prvky systému. Na trhu je hodně výrobců, ale byly stanoveny specifické požadavky, které hodně zúžily výběr. Hlavní kritéria pro výběr řešení:

- Připojení prvků přes ethernet
- Možnost vzdáleného přístupu přes internet
- Minimálně dva digitální teploměry
- Minimálně dvě výkonová relé
- Komunikace do veřejné sítě musí být zabezpečena šifrováním
- Uživatelské rozhraní s dotykovým panelem pro maximální komfort

1.2.1 Řešení Tecomat

Standardní řešení od firmy Tecomat obsahuje jedno základní PLC, na které je možné připojit až čtyři teplotní senzory a má dva výkonové výstupy. Uživatelské rozhraní je řešeno dotykovým panelem na zdi, které komunikuje přes ethernet. Pro vzdálený přístup je možné využít službu TecoRoute, která umožňuje ovládat PLC vzdáleně přes webový prohlížeč, nebo pomocí programu Mozaic. Technická specifikace je dostupná na webu dodavatele [4].

Tabulka 1 Hardware - cenová kalkulace Tecomat

Označení	Vlastnosti	Ks	Cena/ks	Cena celkem
CP-1000	PLC - Porty: 4x vstup, 2x výstup	1	9900	9900
ID-18/28	Vizualizační panel 5.7" s rámečkem	1	16800	16800
S-TS-01R-Future	Teplotní senzor	4	600	2400

Služba TecoRoute je placená, pro tuto variantu by stála 600Kč ročně.

Celková pořizovací cena za řešení: 29100 Kč

Celkové roční náklady: 600 Kč / Irok

1.2.2 Řešení Papouch

Firma Papouch vyrábí velké množství druhů produktů pro průmyslovou elektroniku, datové komunikace a měřicí techniku. Z jejich produktů nejlépe vyhovuje produkt QUIDO_Eth_4/4_1, který má v sobě již jeden teplotní senzor a je možno připojit další dle potřeby. Má rovněž několik výstupů, které je možné použít na ovládání. V rámci standardního řešení [5] nabízí firma Papouch zdarma program WIX, který slouží jako standardní uživatelské rozhraní pro ovládání svých produktů. Nemá přímé řešení pro vzdálenou správu. To by bylo nutno udělat doprogramováním vlastní aplikace, který by byla spuštěna na počítači [6], kde bude provozován program WIX. Současně by zajišťovala spojení s veřejným serverem, kde by bylo veřejně přístupné rozhraní pro uživatele, včetně přístupu z mobilních zařízení. Počítač byl vybrán s dotykovým displejem, což je požadavek na maximální ergonomii uživatelského rozhraní.

Tabulka 2 Hardware - cenová kalkulace Papouch

Označení	Vlastnosti	Ks	Cena/ks	Cena celkem
QUIDO_ETH_4/4_1	PLC 4x vstup, 4x výstup + teploměr	2	3900	7800

QUIDO_THERMO	Dodatečný teploměr PVC (5m)	2	350	700
Program WIX	Webové rozhraní pro ovládání PLC QUIDO	1	0	0
Lenovo AIO C260	ALL-IN-ONE PC na kterém bude spuštěn program WIX včetně 19“ dotykového LCD displeje [6]	1	10000	10000

Rozhraní pro vzdálený přístup by muselo být na veřejném serveru. Pro tuto variantu je cena za webhosting 250Kč / 1rok. Toto rozhraní není součástí dodávky, je potřeba jej vytvořit.

Celková pořizovací cena za řešení: 18500 Kč

Celkové roční náklady: 250 Kč / 1rok

1.2.3 Řešení GHI Electronics

Firma GHI Electronics má ve své nabídce mikrokontroléry a doplňkové moduly založené na technologii .NET Gadgeteer. Nabízí více typů mikrokontrolérů. Vhodným řešením je FEZ Spider [7]. Výkonově by se dal zařadit do kategorie střední třídy. Důležitým faktorem je, že podporuje SSL šifrovací protokol, který je nezbytný pro komunikaci po veřejné síti. Nižší třída základních typů mikroprocesorů disponuje velmi malou pamětí a malým výkonem. Tyto parametry jsou limitující pro šifrování, které vyžaduje vysoký výkon a dostatečně velkou kapacitu operační paměti. Tento nedostatek by se dal obejít tak, že by se řídicí panel připojil do sítě přes ethernetový radič s hardwarovým šifrováním [8]. To by ale zvýšilo složitost zapojení.

Zbývá vyřešit, jak efektivně ovládat systémy, které jsou na delší vzdálenost. Typicky jsou od sebe vzdáleny až 30m. Připojení digitálního senzoru na tuto vzdálenost by bylo velmi náchylné na rušení a způsobovalo by výpadky. Řešením je využít jednoduché programovatelné PLC, na které připojím teplotní senzor i výkonové relé a připojím ho do ethernetové sítě. Bude tak sestaveno zařízení, které bude odolné proti výpadku spojení mezi PLC a řídicím panelem. Pro tento účel je vhodným řešením PLC IP Smartboard [9]. Má možnost připojení dvou teplotních senzorů a jednoho relé. Hlavní výhodou je, že umožňuje nastavit PLC tak, že na základě vyhodnocování teplotních senzorů spíná výkonové relé.

Tabulka 3 Hardware - cenová kalkulace GHI Electronics

Označení	Vlastnosti	Ks	Cena/ks	Cena celkem
FEZ Spider 1.0	FEZ Spider Mainboard	1	1800	1800
Display TE35	3.5" Barevný dotykový displej	1	1600	1600
Ethernet J11D	Modul pro ethernetové připojení	1	300	300

USB Client DP	Napájecí modul + programátor	1	600	600
IP Smartboard	PLC 2x digitální vstup 1x výstup	2	650	1300
RELEF4052	Výkonové relé 12V DC, 8A (2,2KW)	2	50	100
751-193	Napájecí adaptér 12V DC 1500mA	1	200	200
POE splitter	Power over ethernet - splitter	2	125	250
DS18B20	Digitální teplotní senzor Dallas	4	75	300

Pro vzdálený přístup bude potřeba vytvořit vlastní webové rozhraní na veřejném www serveru a program pro řídicí panel, který s ním bude komunikovat. Hosting webu na Azure website je zdarma. Měsíční poplatek za databázi je 120Kč/1měs. Alternativně by šla SQL databáze nahradit řešením Azure table storage, které je zdarma.

Celkové pořizovací náklady: 6450 Kč

Celkové roční náklady: 1400 Kč / 1rok

1.3 Závěr

Nejvhodnějším řešením z nalezených produktů je varianta popsána v kapitole 1.2.3. Umožní nejvíce flexibilní řešení. Jednotlivé komponenty budou jednoduše nahraditelné v případě jejich havárie. Systém bude modulární a v případě potřeby je možné využít komponenty od jiných dodavatelů, systém tak nebude závislý na jednom dodavateli. Protože je systém složen z několika malých komponent, tak v případě poškození prvku je jeho výměna méně náročná, než výměna jednoho velkého zařízení, které má vše integrováno v sobě. Pro mikrokontrolér je možno použít vyšší programovací jazyk .NET, což by mělo umožnit efektivnější a rychlejší vývoj řešení. Pokročilá technologie WPF pro práci s grafikou by měla velmi zjednodušit implementaci designu a uživatelskou přívětivost uživatelského rozhraní. Technologie pro podporu realtime komunikace, by měla umožnit efektivní a rychlý vývoj integrace jednotlivých webových klientů k jednotnému API. Využití služby SQL jako úložiště je v této variantě placená služba. Později je možné tyto periodické náklady vyřešit přestěhováním projektu na jiný server, nebo data například ukládat přes službu Azure table storage [10], která je zdarma. Nicméně kvůli rychlosti implementace bylo zvoleno MS-SQL.

II. PRAKTICKÁ ČÁST

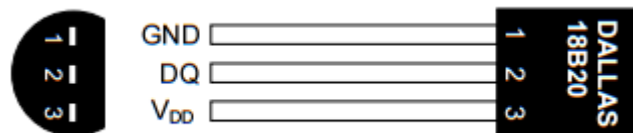
2 SCHÉMA ZAPOJENÍ HARDWARE

Než bude možné jednotlivé komponenty zapojit, tak je potřeba provést detailní analýzu jejich možností a způsobu zapojení dle technické dokumentace.

2.1 Technický popis použitého hardware

2.1.1 Senzory

Dle dokumentace k PLC IP Smartboard [9], je doporučeno připojení senzoru DS18B20, který je běžně dostupný, například v obchodě GM Electronic [11].



Obrázek 1 Senzor DS18B20

Popis jednotlivých pinů je popsán v dokumentaci [11] takto:

- a. GND Ground
- b. DQ Data In/Out
- c. Vdd Power Supply voltage

Rozsah měřených teplot je od -55°C do $+125^{\circ}\text{C}$. Rozlišení je 0.1°C . Tyto parametry jsou dostatečné pro měření vnitřní i venkovní teploty.

2.1.2 Relé

Pro stanovení parametrů relé je potřeba vzít do úvahy napájení cívky. Jestli bude stejnosměrné, střídavé a jak bude napětí na cívce velké. V tomto případě je napětí dáno zdrojem, kterým je napájena deska PLC, tedy 12V stejnosměrné. Pak bylo potřeba zjistit, jaký výkon se bude spínat. U čerpadla to bylo dáno jeho příkonem. Čerpadlo má příkon 1,8KW. Připojení k elektrokotli bude řešeno nahrazením pokojového termostatu novým relé. Výkon relé v termostatu je 250W. Takže je možné použít stejné relé, jako pro čerpadlo. Aby byla dostatečná rezerva, tak bylo zvoleno relé s výkonem 2,2KW [12].

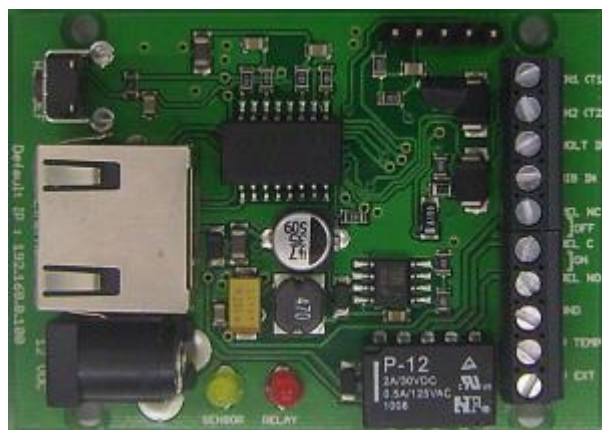


Obrázek 2 Výkonové relé (2,2KW)

Zapojení popsané v dokumentaci je velmi jednoduché. 2 kontakty pro napájení cívky a 3 kontakty pro výstupní přepínač.

2.1.3 PLC

IP Smartboard je víceúčelové zařízení, které umožňuje vzdáleně nastavit požadovanou teplotu, kterou potom udržuje. Konfiguraci lze měnit pomocí HTTP protokolu metodou PUT, pro načítání informací ze zařízení slouží metoda GET, data jsou ve formátu XML. Více informací o možnostech zařízení je k dispozici v podrobném manuálu [9].



Obrázek 3 PLC IP Smartboard



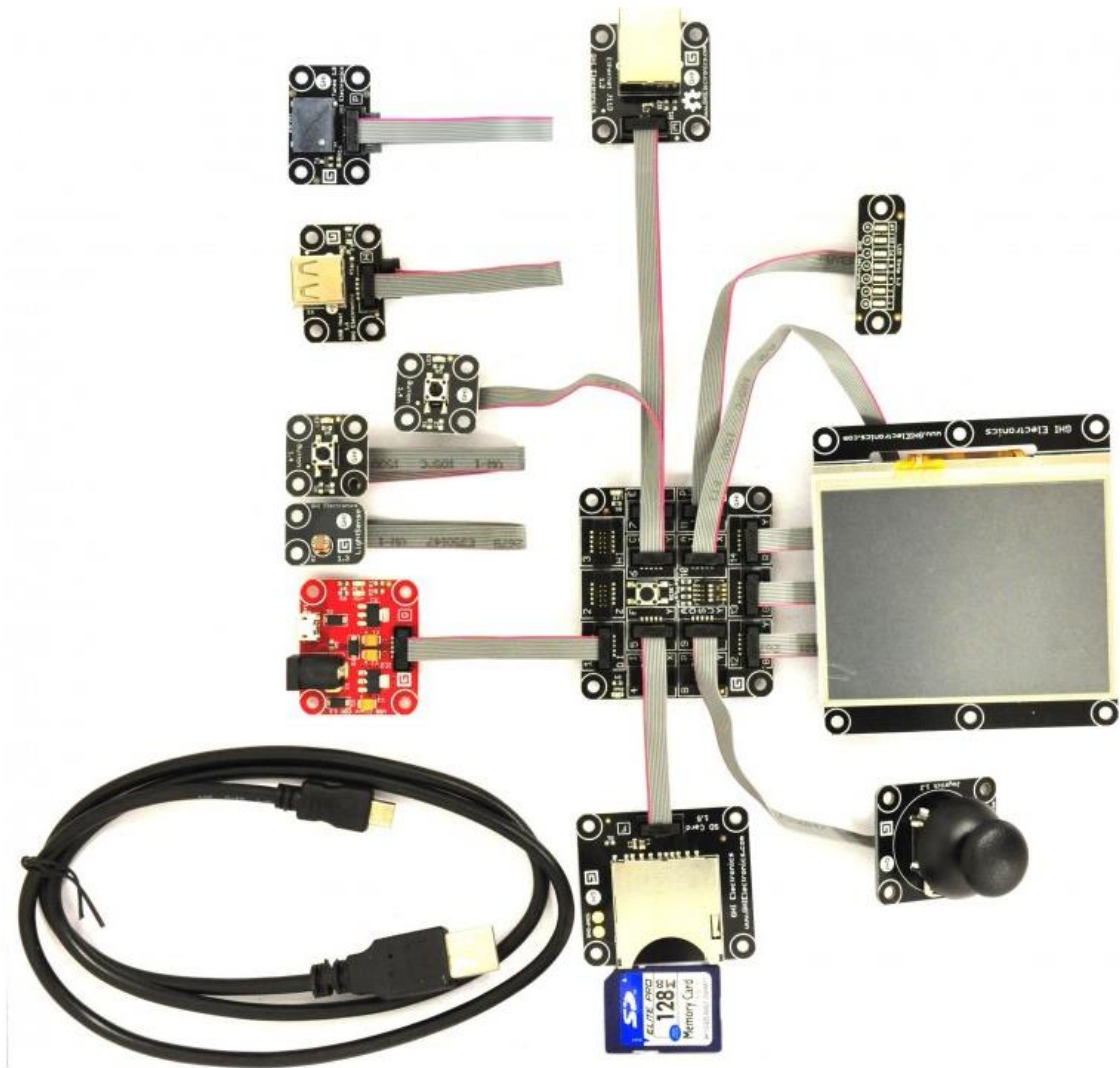
Obrázek 4 Napájecí adaptér 12V DC 1500mA



Obrázek 5 POE splitter

2.1.4 Řídící panel

Fez Spider je mikrokontrolér s podporou šifrování SSL. Programuje se pomocí jazyka C#.NET. K dispozici je široká nabídka modulů a knihoven. Detailní dokumentace je na stránkách dodavatele GHI Electronics [7].



Obrázek 6 Základní vývojová sada FEZ Spider Tinker Kit

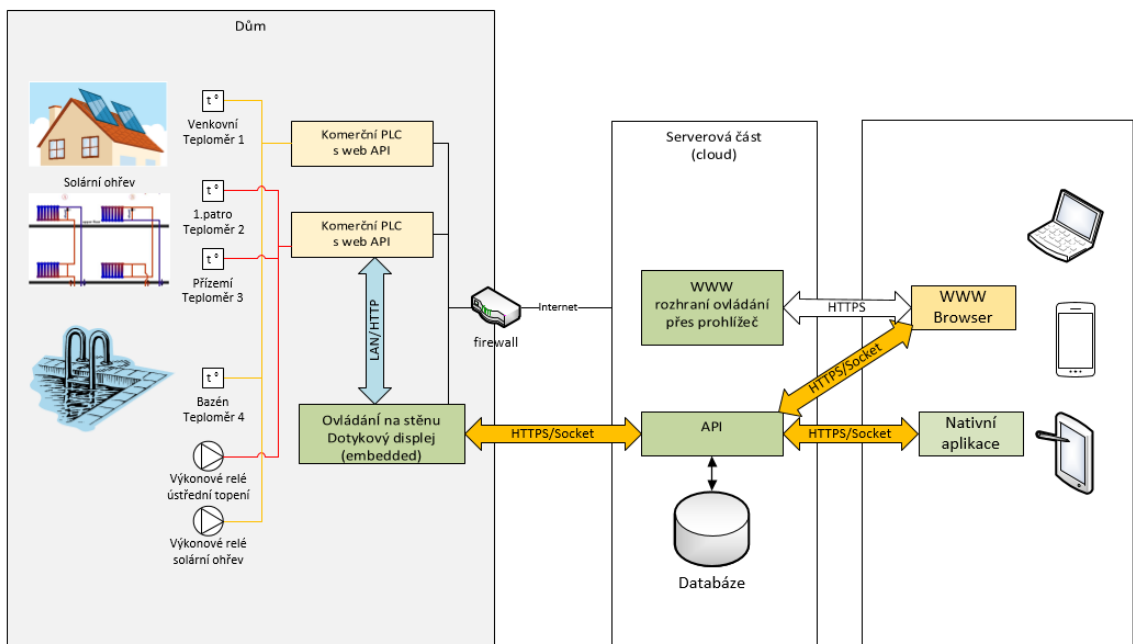
2.1.5 Server

Základním požadavkem je provoz webového portálu s připojením na SQL databázi. MS Azure nabízí službu Azure website, ke které byla objednána služba SQL Web v základní verzi. To by mělo na požadavky řešení stačit a umožní to efektivně načítat a ukládat data i bezpečně komunikovat online s klienty.

2.2 Celkové schéma

V systému jsou plánovány dva nezávislé otopné subsystémy. Každý z těchto subsystémů je ovládán PLC jednotkou, která systém monitoruje dvěma teplotními senzory a ovládá výkonné prvky pomocí relé. PLC jednotky jsou společně s řídicím panelem propojeny ethernetovou sítí do routeru, který zajišťuje připojení do internetu. Řídicí panel s dotykovým displejem zajišťuje rozhraní s uživatelem v rámci budovy. Má přímé spojení s jednotlivými

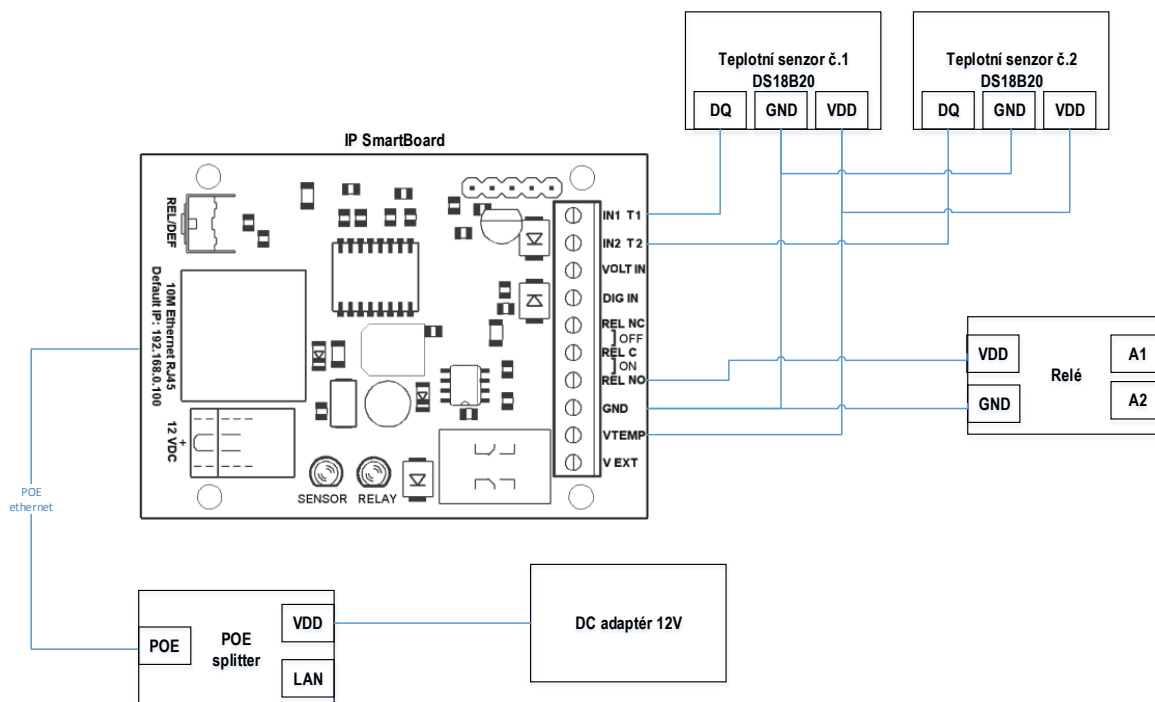
PLC, se kterými komunikuje přes HTTP protokol. Udržuje permanentní spojení se serverem. Toto spojení je zabezpečeno pomocí protokolu SSL. Na serveru je rozhraní API, které umožňuje bezpečné připojení řídicího panelu i ostatních klientských aplikací. Rozhraní je napojeno na databázi, kde je uložena základní konfigurace řídicího systému a jeho jednotlivých podsystémů.



Obrázek 7 Blokové schéma systému

2.3 Detail zapojení PLC, senzorů a relé

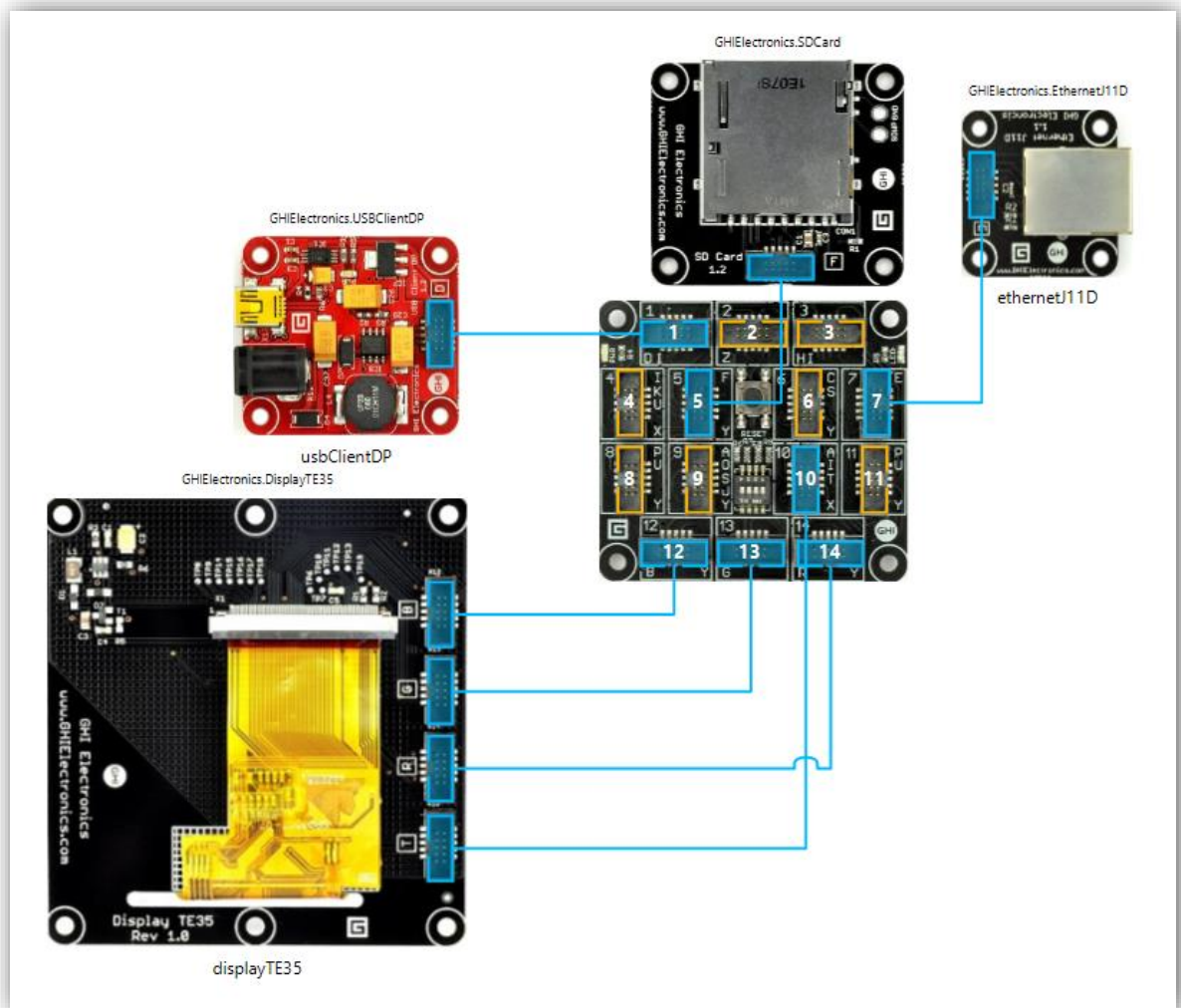
PLC bude připojeno do ethernetu přes POE splitter, do kterého bude zapojen napájecí adaptér. Tím bude možné napájet centrálně z jednoho místa také všechny senzory a relé. Celkem bude v systému zapojeno více PLC, pro zjednodušení je zde detail zapojení jednoho PLC.



Obrázek 8 Schéma zapojení PLC, senzorů relé a společného napájení

2.4 Detail zapojení řídicího panelu

Pro návrh zapojení řídicího panelu je k dispozici grafický editor přímo ve Visual Studiu. Stačí z nástrojové lišty vybrat mikrokontrolér a potřebné komponenty. O správné propojení komponent se postará Visual Studio.



Obrázek 9 Visual Studio designér - schéma zapojení řídicího panelu

Kromě vizuálního náhledu Visual Studio společně nastaví i prostředí pro vývoj programu a nastaví správnou konfiguraci jednotlivých modulů. Toto velmi šetří čas při zapojování komponent i při vlastním programování.

3 NÁVRH SW ČÁSTI

Ze schématu, které bylo popsáno v kapitole 2.2. vyplývá, že bude potřeba navrhnout SW řešení pro řídicí panel a serverové API a pro klientské rozhraní. Všechna tato uživatelská rozhraní musí být schopná mezi sebou přenášet zprávy. Nejprve tedy bude nutno navrhnout způsob přenášení zpráv a následně ukládání těchto dat na server. Návrh bude vycházet z požadavků případů užití.

3.1 Popis případů užití

3.1.1 Vzdálený přístup přes webové rozhraní

3.1.1.1 Zobrazení stavu systému

Případ užití předpokládá, že uživatel si aktivoval funkci „zapamatovat přihlašovací údaje“

Tabulka 4 Případ užití - zobrazení stavu systému na vzdáleném rozhraní

Krok	Role	Akce
1	Uživatel	Spustí webové rozhraní pro vzdálené ovládaní.
3	Webové rozhraní	Připojí se trvale s rozhraním API. Pošle požadavek na načtení údajů z PLC. Současně s požadavkem je poslán autentizační token.
4	Server	Kontrola autentizačního tokenu klienta, POKUD není validní, tak zobrazí přihlašovací formulář. KONEC-POKUD
5	Server	Vyžádá si od řídicího panelu aktualizaci stavu.
6	Řídicí panel	Vyžádá si od PLC aktualizaci stavu.
7	PLC	Změří údaje a vrátí je řídicímu panelu.
8	Řídicí panel	Odešle údaje na server.
9	Server	Odešle nové údaje všem aktuálně přihlášeným webových rozhraním.
10	Uživatel	Zaktualizují se údaje na hlavní obrazovce.

3.1.1.2 Autentizace nepřihlášeného uživatele

Případ užití předpokládá, že uživatel ještě nikdy nebyl přihlášen, nebo si nezaktivoval funkci „zapamatovat přihlášení“

Tabulka 5 Případ užití - autentizace nepřihlášeného uživatele

Krok	Role	Akce
1	Uživatel	Spustí webové rozhraní pro vzdálené ovládaní
2	Webové rozhraní	Odešle požadavek na server na načtení údajů bez validního autentizačního tokenu.
3	Server	Odmítne spojení.

4	Webové rozhraní	Zobrazí uživateli přihlašovací formulář.
5	Uživatel	Zadá přihlašovací údaje.
6	Webové rozhraní	Vyžádá si token na základě zadaných autentizačních údajů.
7	Server	POKUD není IP adresa webového rozhraní na dočasném tzv. „černé listině“ kvůli překročení neplatných pokusů o přihlášení a současně autentizační údaje jsou platné, tak vygeneruje autentizační token. JINAK zamítne požadavek. KONEC-POKUD
8	Webové rozhraní	POKUD Uživatel zaktivoval funkci „zapamatovat přihlášení“, tak si uloží token do úložiště prohlížeče s platností 1měsíc. KONEC-POKUD Zobrazí uživateli hlavní obrazovku.

3.1.1.3 Změna programu

Případ užití předpokládá, že uživatel je přihlášen a má na vzdáleném webovém rozhraní zobrazenou hlavní obrazovku.

Tabulka 6 Případ užití - změna programu ze vzdáleného rozhraní

Krok	Role	Akce
1	Uživatel	Zvolí akci nastavit program u konkrétního PLC.
2	Webové rozhraní	Zobrazí uživateli formulář pro výběr z dostupných programů pro dané PLC
3	Uživatel	Zvolí program z nabídky
4	Webové rozhraní	Odešle požadavek na server
5	Server	Zaktualizuje konfigurační údaje v SQL databázi. POKUD není řídicí panel přihlášen, tak konec. KONEC-POKUD Odešle informaci, že došlo ke změně konfigurace PLC na řídicí panel.
6	Řídicí panel	Uloží si do interní paměti aktualizovaný stav konfigurace systému. Odešle na PLC požadavek na aktualizaci programu.
7	PLC	Nastaví program.
8	Řídicí panel	Odešle na PLC požadavek na aktualizaci informací o stavu.
9	PLC	Změří data a vrátí požadované informace.
10	Řídicí panel	Odešle aktuální informace na server.
11	Server	Odešle aktuální informace na všechny aktuálně připojené webová rozhraní.
12	Webové rozhraní	POKUD je zobrazena hlavní obrazovka, tak se zaktualizují údaje. KONEC-POKUD

3.1.1.4 Ruční korekce požadované teploty

Případ užití předpokládá, že uživatel je přihlášen a má na vzdáleném webovém rozhraní zobrazenou hlavní obrazovku.

Tabulka 7 Příklad užití - korekce žádané teploty ze vzdáleného rozhraní

Krok	Role	Akce
1	Uživatel	Zvolí akci nastavit program u konkrétního PLC.
2	Webové rozhraní	Zobrazí uživateli formulář pro výběr z dostupných programů pro dané PLC s nabídkou ruční korekce v rozsahu od -4°C do +4°C
3	Uživatel	Zvolí hodnotu požadované korekce z nabídky.
4	Webové rozhraní	Odešle požadavek na server.
5	Server	Zaktualizuje konfigurační údaje v SQL databázi. POKUD není řídicí panel přihlášen, tak konec. KONEC-POKUD Odešle informaci, že došlo ke změně konfigurace PLC na řídicí panel.
6	Řídicí panel	Uloží si do interní paměti aktualizovaný stav konfigurace systému. Odešle na PLC požadavek na aktualizaci programu.
7	PLC	Nastaví program.
8	Řídicí panel	Odešle na PLC požadavek na aktualizaci informací o stavu.
9	PLC	Změří data a vrátí požadované informace.
10	Řídicí panel	Odešle aktuální informace na server.
11	Server	Odešle aktuální informace na všechny aktuálně připojené webová rozhraní.
12	Webové rozhraní	POKUD je zobrazena hlavní obrazovka, tak se zaktualizují údaje. KONEC-POKUD

3.1.1.5 Detailní nastavení programu

Příklad užití předpokládá, že uživatel je přihlášen a má na vzdáleném webovém rozhraní zobrazenou hlavní obrazovku.

Tabulka 8 Příklad užití - detailní nastavení programu ze vzdáleného rozhraní

Krok	Role	Akce
1	Uživatel	Zvolí akci nastavit program u konkrétního PLC.
2	Webové rozhraní	Zobrazí uživateli formulář pro výběr z dostupných programů pro dané PLC s nabídkou ruční korekce v rozsahu od -4°C do +4°C.
3	Uživatel	Zvolí akci „detail“ u daného programu ze seznamu.
4	Webové rozhraní	Zobrazí týdenní kalendář a v něm seznam akcí tak jak jsou plánovány zvoleným programem.
5	Uživatel	Nastaví změny v kalendáři. Je možné přidávat a odebírat jednotlivé položky. Jakákoliv změna v kalendáři se informací o změně na server.
6	Server	Zaktualizuje konfigurační údaje v SQL databázi. POKUD není řídicí panel přihlášen, tak konec. KONEC-POKUD Odešle informaci, že došlo ke změně konfigurace programu na řídicí panel.
7	Řídicí panel	Uloží si do interní paměti aktualizovaný stav konfigurace systému. Odešle na PLC požadavek na aktualizaci programu.
8	PLC	Nastaví program.
9	Řídicí panel	Odešle na PLC požadavek na aktualizaci informací o stavu.
10	PLC	Změří data a vrátí požadované informace.

11	Řídicí panel	Odešle aktuální informace na server
12	Server	Odešle aktuální informace na všechny aktuálně připojené webová rozhraní.
13	Webové rozhraní	POKUD je zobrazena hlavní obrazovka, tak se zaktualizují údaje. KONEC-POKUD

3.1.2 Lokální přístup přes řídicí panel

3.1.2.1 Zobrazení stavu systému na řídicím panelu

Případ užití předpokládá, že řídicí panel je připojen k systému.

Tabulka 9 Případ užití - zobrazení stavu na řídicím panelu

Krok	Role	Akce
1	Uživatel	Spustí řídicí panel připojením napájecího zdroje.
2	Řídicí panel	Provede inicializaci modulů a spustí periodický proces automatické synchronizaci času se světovým časem. POKUD je k dispozici na lokálním úložišti záloha konfigurace, tak záloha bude použita pro obnovení do výchozího stavu. KONEC-POKUD
3	Řídicí panel	Načte konfiguraci ze serveru. S odesláním požadavku je současně odeslán i autentizační token.
4	Server	Provede ověření autentizačního tokenu. POKUD není validní, tak zamítne další spojení. JINAK Načte data z SQL a vrátí je řídicímu panelu. KONEC-POKUD
5	Řídicí panel	POKUD je připojení pořádku a načetla se konfigurace, tak se uloží do interní paměti aktualizovaný stav konfigurace systému a odešle na PLC požadavek na aktualizaci programu. JINAK zobrazí se chybová obrazovka a panel se pokouší periodicky zopakovat spojení. KONEC POKUD
6	PLC	Nastaví program
7	Řídicí panel	Odešle na PLC požadavek na aktualizaci informací o stavu.
8	PLC	Změří data a vrátí požadované informace řídicímu panelu.
9	Řídicí panel	Odešle aktuální informace na server.
10	Server	Odešle aktuální informace na všechny aktuálně připojené webová rozhraní.
11	Webové rozhraní	POKUD je zobrazena hlavní obrazovka, tak se zaktualizují údaje. KONEC-POKUD

3.1.2.2 Změna zobrazení stavu

Případ užití předpokládá, že řídicí panel je připojen do systému a má zobrazenou úvodní obrazovku.

Tabulka 10 Příklad užití – změna zobrazení stavu na řídicím panelu

Krok	Role	Akce
1	Uživatel	Na panelu klikne na akci „zobrazit další“
2	Řídicí panel	Z tabulky dostupných PLC vezme další v pořadí a zobrazí ho na panelu
3	Řídicí panel	Odešle na PLC požadavek na aktualizaci informací o stavu.
4	PLC	Změří data a vrátí požadované informace řídicímu panelu.
5	Řídicí panel	Odešle aktuální informace na server.
6	Server	Odešle aktuální informace na všechny aktuálně připojené webové rozhraní.
7	Webové rozhraní	POKUD je zobrazena hlavní obrazovka, tak se zaktualizují údaje. KONEC-POKUD

3.1.2.3 Změna programu

Příklad užití předpokládá, že řídicí panel je připojen do systému a má zobrazenou úvodní obrazovku.

Tabulka 11 Příklad užití - změna programu na řídicím panelu

Krok	Role	Akce
1	Uživatel	Zvolí akci nastavit program u konkrétního PLC.
2	Řídicí panel	Zobrazí uživateli formulář pro výběr z dostupných programů pro dané PLC
3	Uživatel	Zvolí program z nabídky
4	Řídicí panel	Uloží si do interní paměti aktualizovaný stav konfigurace systému. A odešle požadavek na server o uložení změny konfigurace
5	Server	Uloží konfiguraci do SQL
6	Řídicí panel	Odešle na PLC požadavek na aktualizaci programu.
7	PLC	Nastaví program.
8	Řídicí panel	Odešle na PLC požadavek na aktualizaci informací o stavu.
9	PLC	Změří data a vrátí požadované informace.
10	Řídicí panel	Odešle aktuální informace na server.
11	Server	Odešle aktuální informace na všechny aktuálně připojené webové rozhraní.
12	Webové rozhraní	POKUD je zobrazena hlavní obrazovka, tak se zaktualizují údaje. KONEC-POKUD

3.1.2.4 Ruční korekce požadované teploty

Příklad užití předpokládá, že řídicí panel je připojen do systému a má zobrazenou úvodní obrazovku.

Tabulka 12 Příklad užití - korekce žádané teploty z řídicího panelu

Krok	Role	Akce
1	Uživatel	Zvolí akci nastavit program u konkrétního PLC.
2	Řídící panel	Zobrazí uživateli formulář pro výběr z dostupných programů pro dané PLC s nabídkou ruční korekce v rozsahu od -4°C do +4°C
3	Uživatel	Zvolí požadovanou hodnotu korekce
4	Řídící panel	Uloží si do interní paměti aktualizovaný stav konfigurace systému. A odešle požadavek na server o uložení změny konfigurace
5	Server	Uloží konfiguraci do SQL
6	Řídící panel	Odešle na PLC požadavek na aktualizaci programu.
7	PLC	Nastaví program.
8	Řídící panel	Odešle na PLC požadavek na aktualizaci informací o stavu.
9	PLC	Změří data a vrátí požadované informace.
10	Řídící panel	Odešle aktuální informace na server.
11	Server	Odešle aktuální informace na všechny aktuálně připojené webové rozhraní.
12	Webové rozhraní	POKUD je zobrazena hlavní obrazovka, tak se zaktualizují údaje. KONEC-POKUD

3.1.3 Periodická aktualizace údajů

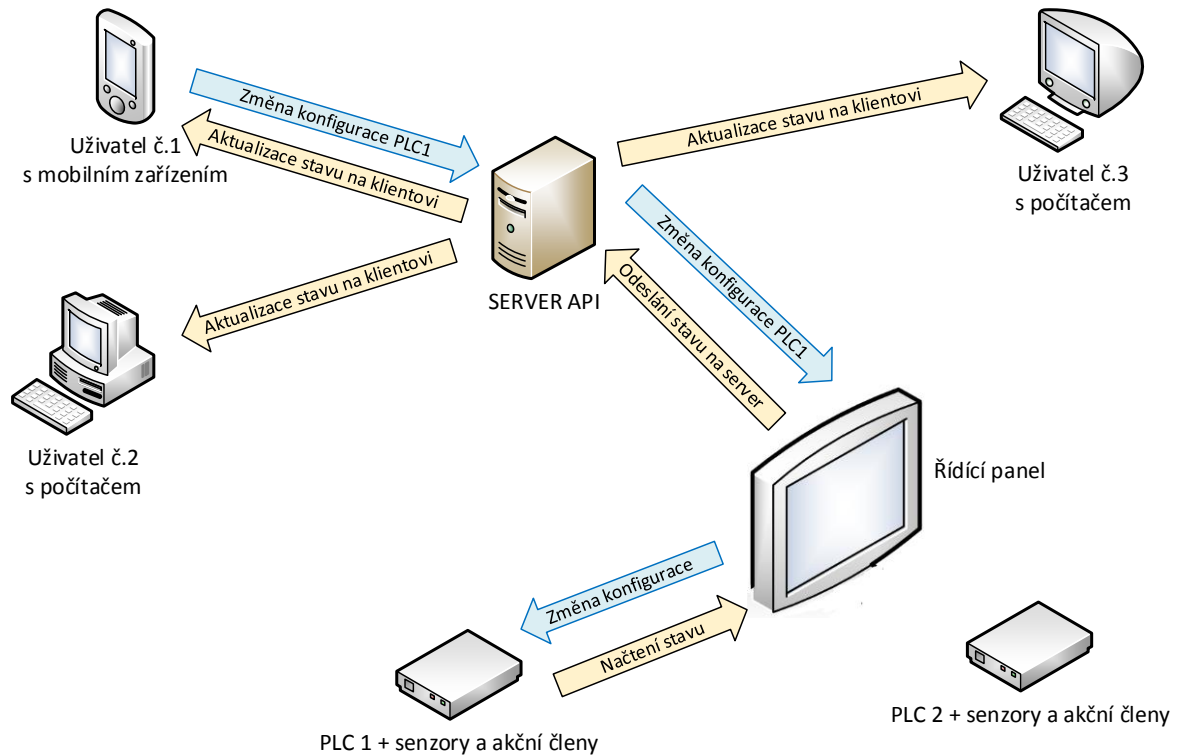
Příklad užití předpokládá, že řídicí panel je připojen do systému. Akce se spouští periodicky 1x za 10 minut.

Tabulka 13 Příklad užití - periodická aktualizace stavu

Krok	Role	Akce
8	Řídící panel	Odešle na PLC požadavek na aktualizaci informací o stavu.
9	PLC	Změří data a vrátí požadované informace.
10	Řídící panel	Uloží si naměřená data pro pozdější vyhodnocení trendu změn teploty. Odešle aktuální informace na server.
11	Server	Odešle aktuální informace na všechny aktuálně připojené webové rozhraní.
12	Webové rozhraní	POKUD je zobrazena hlavní obrazovka, tak se zaktualizují údaje. KONEC-POKUD

3.2 Volba komunikační technologie klient-klient, klient-server

API je aplikační rozhraní, které umožňuje různorodým aplikacím vzájemnou výměnu dat. Nejlépe by bylo použít takovou technologii, která umožní toto API sdílet pro všechny typy klientských rozhraní, včetně řídicího panelu. Zároveň z případu užití v předchozí kapitole vyplývá, že to musí být realtime komunikace, tedy jakákoliv změna v systému se musí ihned roz distribuovat jak na koncové PLC, tak i na všechny přihlášené klienty.



Obrázek 10 Zjednodušené schéma distribuovaného řízení

Existuje více řešení, které splňují výše uvedené požadavky.

3.2.1 Azure mobile services

Systém push notifikací je univerzální a jednoduše implementovatelný. Umožňoval by i horizontální škálování pro případné pozdější navyšování výkonu a dostupnosti. Další výhodou je, že systém dokáže pracovat i off-line, tedy v případě, že příjemce není dostupným tak zpráva čeká ve frontě, dokud se klient nepřipojí, nebo dokud nevyexpiruje platnost zprávy. Nevýhodou je, že fronta zpráv je asynchronní a v případě řešení nutnosti mít potvrzení o zpracování zprávy nebo očekávané návratové informace to bude působit obtíže při implementaci. [13]

3.2.2 XSocket.NET

Systém řeší realtime komunikaci mezi klientskými aplikacemi a serverem. SDK podporuje mimo jiné i NET-Microframework, což umožní jednoduchou implementaci na řídicím panelu. Nevýhodou je poměrně vysoká pořizovací cena. V nejmenší možné konfiguraci stojí licence 8000 Kč. Neomezená licence pak stojí 25000 Kč. [14]

3.2.3 SignalR

Tato knihovna řeší rovněž realtime komunikaci mezi klientskými aplikacemi a serverem. SDK bohužel ale nepodporuje NET-Microframework. Nicméně tento problém by šel obejít, protože protokol používaný v SignalR je poměrně jednoduchý a dobře dokumentovaný. Praktický význam této technologie tkví v tom, že umožňuje libovolnému klientovi jak volat metody na serveru, tak i ze serveru volat metody na přihlášeném klientovi. Přitom nezáleží na tom, jestli ten klient je v Javascriptu, .NETu. nebo jiném podporovaném jazyce [15].

3.2.4 Závěr

Vhodnější je tedy použít systém, který dokáže zpracovat zprávy synchronně. Z dostupných řešení je tedy nejvhodnější technologie SignalR a to i za cenu, že bude vyšší nárok na implementaci na straně řídicího panelu.

3.3 Sdílené struktury pro přenos dat

Mezi klienty se budou přenášet datové struktury, které se budou na jedné straně serializovat a na druhé zase deserializovat. Z toho důvodu je žádoucí, aby tyto struktury byly sdíleny mezi řídicím panelem, API rozhraním i ostatními klienty. Bude potřebovat přenášet tyto struktury:

- Konfiguraci PLC

Zde bude uložena zejména informace o IP adrese a pomocné údaje o připojených senzorech a režimu řízení. Podporovány budou režimy:

- o Režim „Terms“

PLC se snaží udržovat nastavenou teplotu. To se bude používat na udržování teploty uvnitř domu

- o Režim „Difference“

PLC porovnává teploty z obou senzorů a pokud je rozdíl větší jak daná hodnota, tak se sepne relé. Toto bude využíváno pro spínání solárního ohřevu. PLC tedy sepne relé, pokud je teplota venkovního prostředí vyšší, než teplota ohřívání média.

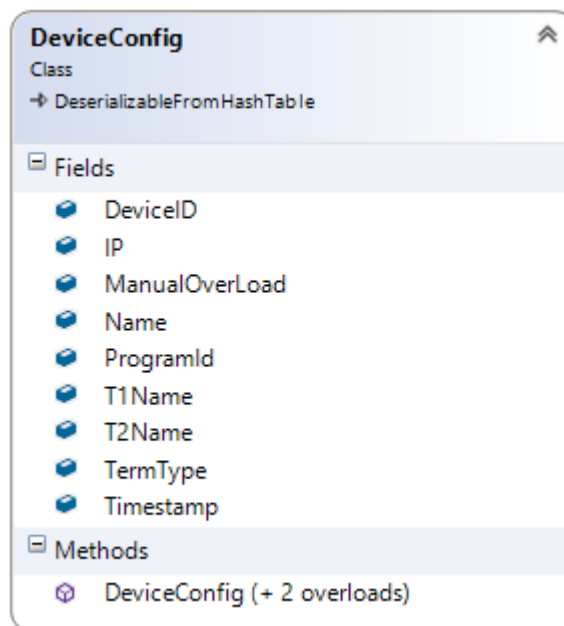
- Naměřená data

Struktura bude používána pro přenos změřených dat a stavu relé z PLC na server API a na všechny klientské aplikace. Současně s těmito údaji je přenášena i informace o

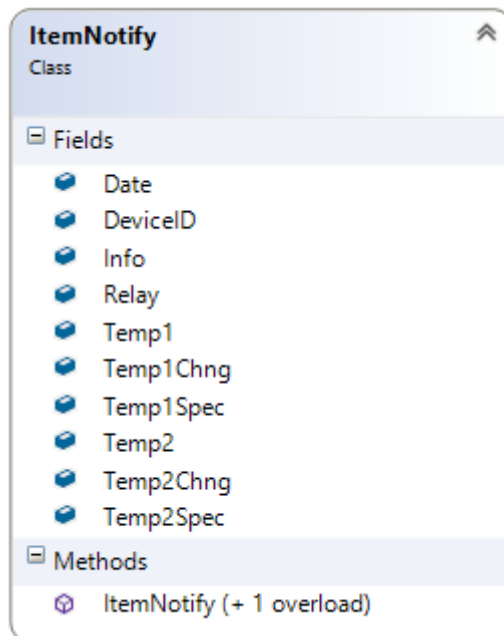
aktuálně probíhajícím programu a jaký program bude následovat a jaký trend má měřená teplota. Tedy jestli klesá, nebo stoupá.

- Nastavení jednotlivých programů pro PLC

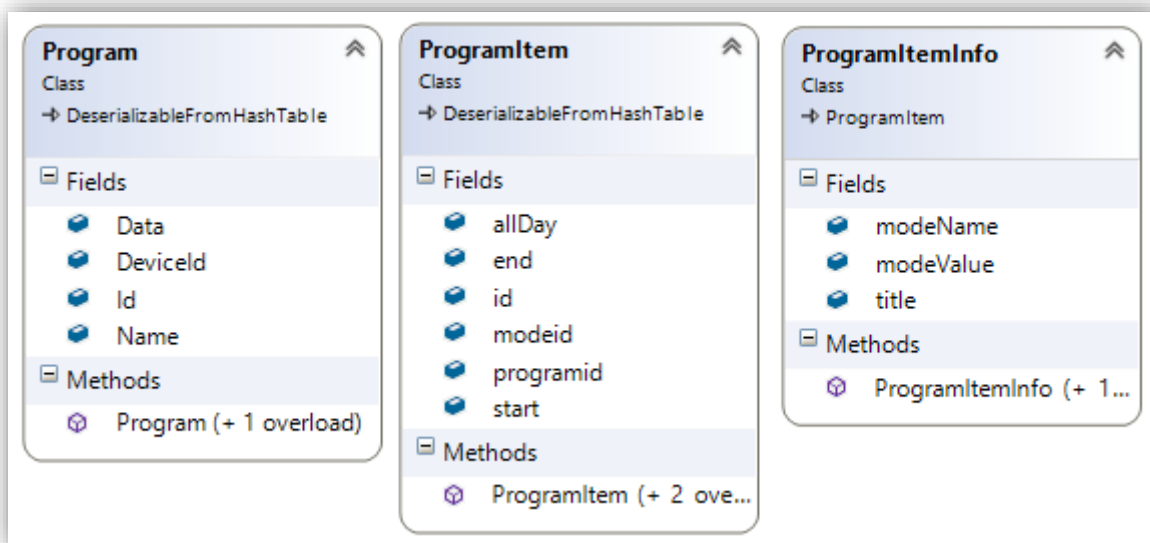
Pro každé PLC je možné nastavit sadu příkazů. Každý příkaz má dobu trvání od-do a teplotu, jakou je potřeba v daný čas udržovat.



Obrázek 11
Struktura konfigurace PLC



Obrázek 12
Struktura pro změřená data



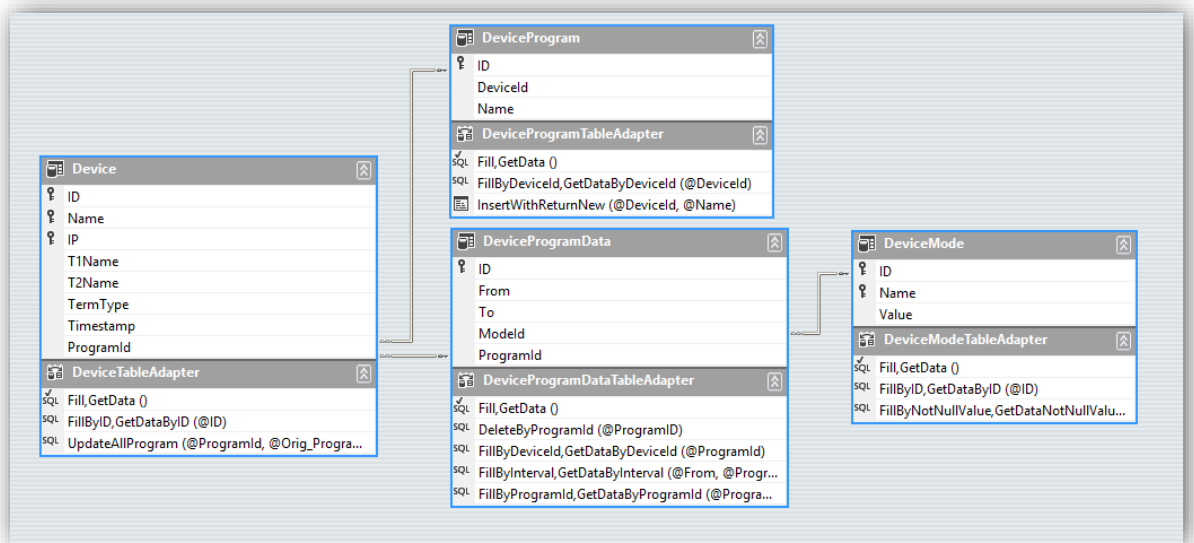
Obrázek 13 Struktury pro přenos konfigurace programu

Poznámka:

Kvůli použití sdílené struktury ProgramItem v javascript klientovi, bylo pojmenování polí struktury záměrně pojmenováno tak, aby bylo možné tuto strukturu použít bez úprav jako zdroj dat pro jQuery knihovnu fullCalendar [16].

3.4 Databáze

Je potřeba ukládat informace o zařízeních. K tomu bude sloužit tabulka Device. Na ní budou závislá tabulka DeviceProgram, která umožní danému zařízení (PLC) nastavit vlastní program. V tabulce ProgramData jsou uložena data programu. Zejména informace jaký režim se má na zařízení nastavit a jak dlouho to má trvat. Poslední tabulka je DeviceMode, která slouží pouze ke zjednodušení administrace. Obsahuje několik předdefinovaných teplot a jejich název.



Obrázek 14 Celkové schéma databáze

V tabulce Device je uložena konfigurace jednotlivých PLC. Důležitá je IP adresa a typ řízení daného PLC. Každé PLC má připojené dva teplotní senzory a jedno relé které řídí. Tabulka DeviceMode je pomocná, umožňuje klientovi při konfiguraci programu vybírat předdefinované hodnoty z číselníku, místo toho aby je psal ručně u každé položky programu.

ID	Name	IP	T1Name	T2Name	TermType	ProgramId	Timestamp	ManualOverload
1	Dům	192.168.1.10	Přízemí	1.NP	Terms	2	16. 5. 2015 20:37:23	0
2	Bazén	192.168.1.11	Vzduch	Voda	Difference	10	8. 5. 2015 23:57:19	0

Obrázek 15 Ukázka tabulky Device

ID	Deviceld	Name
1	1	Běžný pracovní týden
8	1	Dovolená
2	1	Pouze víkend
3	1	Stále topit
10	2	Vypnuto

Obrázek 16

Ukázka tabulky DeviceProgram

ID	From	To	Modeld	ProgramId
10040	1. 12. 2014	1. 12. 2014 23:59	4	8
10039	2. 12. 2014	2. 12. 2014 23:59	4	8
10034	3. 12. 2014	3. 12. 2014 23:59	4	8
10035	4. 12. 2014	4. 12. 2014 23:59	4	8
10036	5. 12. 2014	5. 12. 2014 23:59	4	8
10037	6. 12. 2014	6. 12. 2014 23:59	4	8
10038	7. 12. 2014	7. 12. 2014 23:59	4	8
10068	7. 12. 2014 1:00	7. 12. 2014 2:00	2	8
10047	1. 12. 2014	1. 12. 2014 23:59	2	3
10046	2. 12. 2014	2. 12. 2014 23:59	2	3

Obrázek 17 Ukázka tabulky DeviceProgramData

ID	Name	Value
2	Den	24,0
3	Noc	19,0
4	Útlum	13,0

Obrázek 18

Ukázka tabulky DeviceMode

3.5 Serverová část - API

3.5.1 Komunikace

Komunikaci zajišťuje knihovna SignalR. Řeší transparentní serializaci a deserializaci při přenosu dat. Pro přenášení dat budou využity sdílené datové struktury popsané v kapitole 3.2. Knihovna se nainstaluje standardně pomocí NuGet.

Interní metody, které slouží pro komunikaci ve směru server→klient

- **updateDeviceConfig(DeviceId)**
zavolat aktualizaci konfigurace PLC na všech klientech. Typicky se bude volat po změně programu, nebo změně nastavení programu. Proběhne i aktualizace programu.
- **deviceProgramChange(DeviceId, ModeId)**
informovat klienty, že došlo ke změně zvoleného programu u daného PLC.
- **OnDisconnected(StopCalled)**
Metoda se spustí při odpojení klienta od serveru. V případě, že je to řídicí panel, tak je potřeba informovat klienty, že je panel odpojen.
- **OnConnected()**
Metoda se spustí vždy, když se připojí nový klient, Pokud je to řídicí panel, tak by se poslala všem připojeným klientům zpráva, že je panel připojen k serveru.

Externí metody, které slouží pro komunikaci ve směru klient→server

- **deviceGetConfig()**
Vrátí klientovi seznam všech připojených PLC včetně jejich konfigurace
- **deviceUpdateConfig(DeviceConfig)**
změní hodnoty základních parametrů PLC. Zejména popis zařízení a popisky jednotlivých senzorů
- **deviceGetData()**
Předá informaci řídicímu panelu, že klient/klienti si přeje zaktualizovat stav PLC a naměřená data.
- **deviceProgramUpdate(Program)**
umožní klientovi zaktualizovat program, včetně všech plánovaných akcí, pro dané PLC.
- **deviceProgramDelete(ProgramId)**

- odstranění celého programu včetně plánovaných akcí
- **deviceProgramDataUpdate(ProgramDataItem)**
aktualizace jedné plánované akce programu.
 - **deviceProgramDelete(Id)**
smaže jednu plánovanou akci z programu.
 - **deviceProgramChange(DeviceId, ProgramId)**
změní u PLC aktuálně vybraný program.
 - **deviceManualOverloadChange(DeviceId, Value)**
nastavení offsetové hodnoty pro dané zařízení, která ovlivní výslednou požadovanou teplotu z programu.
 - **modesGet()**
načtení číselníku předdefinovaných teplotních režimů pro plánovací kalendář.
 - **controllerConected()**
tuto metodu volá řídicí panel a oznamuje tak, že je připraven zpracovávat příchozí komunikaci ze serveru.

3.5.2 Autentizace

Bude použita standardní Webform autentizace [17] a bude implementována ochrana proti opakovaným pokusům o uhádnutí loginu/hesla. Při 10-ti neplatných pokusech o přihlášení je uživateli na 10 minut zamezeno se přihlásit a zobrazí se mu varovné hlášení.

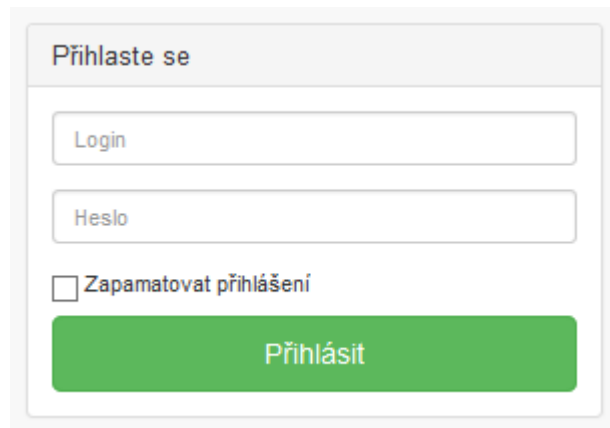
3.6 Webové rozhraní

Webové rozhraní musí mít responsivní design, aby bylo možné web prohlížet na různě velkých prohlížečích. Pro tento účel je vhodná velmi používaná knihovna Bootstrap [18], kterou je možné doinstalovat do projektu pře NuGet. Knihovna má k dispozici sadu ovládacích prvků, které lze využít při návrhu rozhraní jednotlivých obrazovek.

Celkem bude stačit udělat tři webové formuláře: přihlašovací obrazovku, hlavní panel a nastavení programu pro PLC.

3.6.1 Přihlašovací obrazovka

Při prvním vstupu na stránky je uživatel přesměrován na obrazovku přihlášení. Pokud se úspěšně přihlásí, tak se mu zobrazí hlavní panel s přehledem všech PLC. V případě, že vloží neplatné autentizační údaje, tak se mu zobrazí chybové hlášení. Je zde uživateli umožněno zvolit trvale zapamatovat přihlášení. Při dalších vstupech je již uživatel automaticky přihlášen a zobrazí se mu hlavní obrazovka.



Přihlaste se

Login

Heslo

Zapamatovat přihlášení

Přihlásit

Obrázek 19 Přihlašovací obrazovka

3.6.2 Hlavní obrazovka

Po úspěšném přihlášení je uživatel přesměrován na hlavní obrazovku. Protože je dostatek místa, tak je možné zobrazit informace ze všech připojených zařízení současně. Z původního návrhu vyplynulo, že každé PLC bude mít dva teplotní senzory, které budou zobrazovat aktuální teplotu a její trend vůči střední hodnotě měřené v 10-ti minutových intervalech.

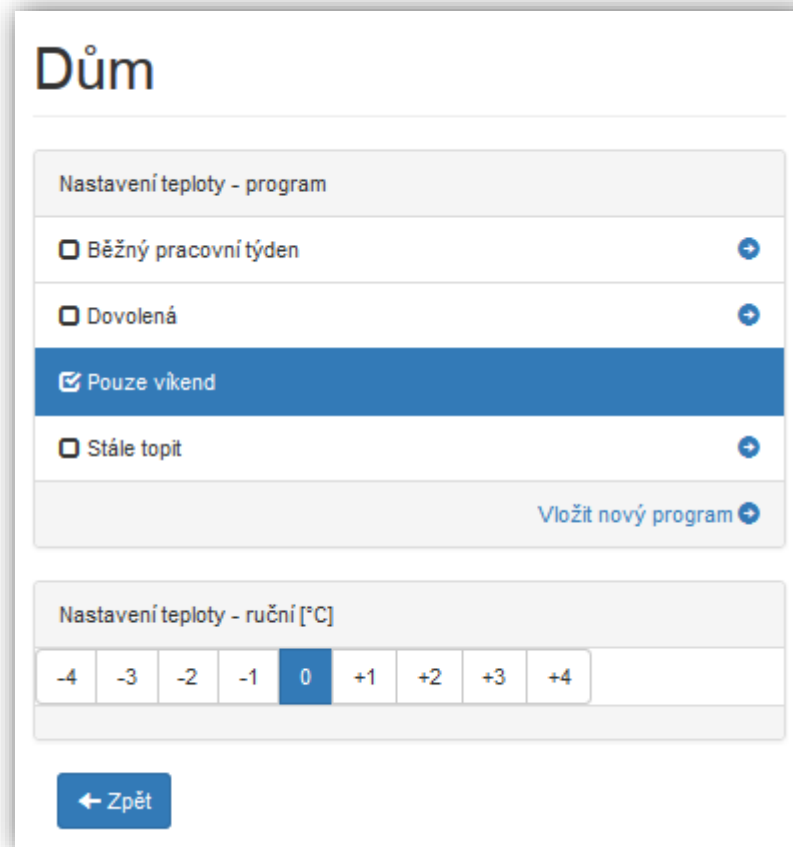
Dům	Bazén																
<table><thead><tr><th colspan="2">TEPLOTA</th></tr></thead><tbody><tr><td>1.NP</td><td>23.3°C ↓</td></tr><tr><td>Přízemí</td><td>20.3°C ↓</td></tr><tr><td colspan="2" style="text-align: right;">Nastavení</td></tr></tbody></table>	TEPLOTA		1.NP	23.3°C ↓	Přízemí	20.3°C ↓	Nastavení		<table><thead><tr><th colspan="2">TEPLOTA</th></tr></thead><tbody><tr><td>Voda</td><td>18.9°C —</td></tr><tr><td>Vzduch</td><td>11.1°C —</td></tr><tr><td colspan="2" style="text-align: right;">Nastavení</td></tr></tbody></table>	TEPLOTA		Voda	18.9°C —	Vzduch	11.1°C —	Nastavení	
TEPLOTA																	
1.NP	23.3°C ↓																
Přízemí	20.3°C ↓																
Nastavení																	
TEPLOTA																	
Voda	18.9°C —																
Vzduch	11.1°C —																
Nastavení																	
<p>Udržuje se teplota 19°C</p>	<p>Topení bude vypnuto celý den</p>																
<p>Řídicí jednotka připojena</p>	<p>Přihlášen uživatel: demo Odhlásit</p>																

Obrázek 20 Úvodní obrazovka, přehled základních informací

Pod panelem PLC je pak několik menších informačních panelů, které znázorňují aktuální program na daném PLC a stav připojení řídicího panelu (mikrokontroléru) k serverovému API. Pokud se řídicí panel od API odpojí, tak je uživatel o tomto stavu informován. Jako doplňková funkce je zde panel s informací o aktuálně přihlášeném uživateli a tlačítko pro odhlášení.

3.6.3 Nastavení PLC

Zobrazí se zde seznam dostupných programů pro vybrané PLC. Uživatel může vybrat program kliknutím na jeho název. Po kliknutí se okamžitě upraví konfigurace na PLC a všichni informace se zaktualizuje u všech současně připojených klientů. Kliknutím u programu na ikonku vpravo se zobrazí detailní nastavení programu. Je zde i možnost založit nový program. Ale to by neměla být častá operace. Jako doplňková funkce pro zvýšení uživatelského komfortu je zde možnost upravit požadovanou hodnotu na daném PLC pomocí offsetu. Tedy že offset hodnota se přičte k požadované teplotě dané aktuálním programem na daném PLC. Opět po kliknutí na požadovanou hodnotu se změna projeví jak v nastavení PLC, tak i na všech aktuálně připojených klientech.



Obrázek 21 Detail nastavení PLC, výběr programu

3.6.4 Detailní nastavení programu, plánovací kalendář

Toto zobrazení detailu programu je přizpůsobeno klientům, kteří mají velký displej. Pokud je detekován malý displej, tak je zobrazena úspornější forma kalendáře, ale to bude popsáno v dalších kapitole. Při implementaci kalendáře budou implementovány podobně funkce, na které je uživatel zvyklý z běžných používaných kalendářů, jako například MS-Outlook.

Zobrazen je týdenní režim. Každý sloupec znázorňuje jeden den. Řádky pak označují čas. Programové položky, které trvají celý den, se vkládají do záhlaví daného dne. Nová položka se do kalendáře vkládá kliknutím do požadovaného místa v kalendáři. Jednotlivé události se dají přesouvat mezi jednotlivými dny, nebo potáhnutím za dolní, nebo horní konec je možné rychle upravit start nebo konec dané události. Kliknutím na událost se zobrazí její detail.

Název Pouze víkend

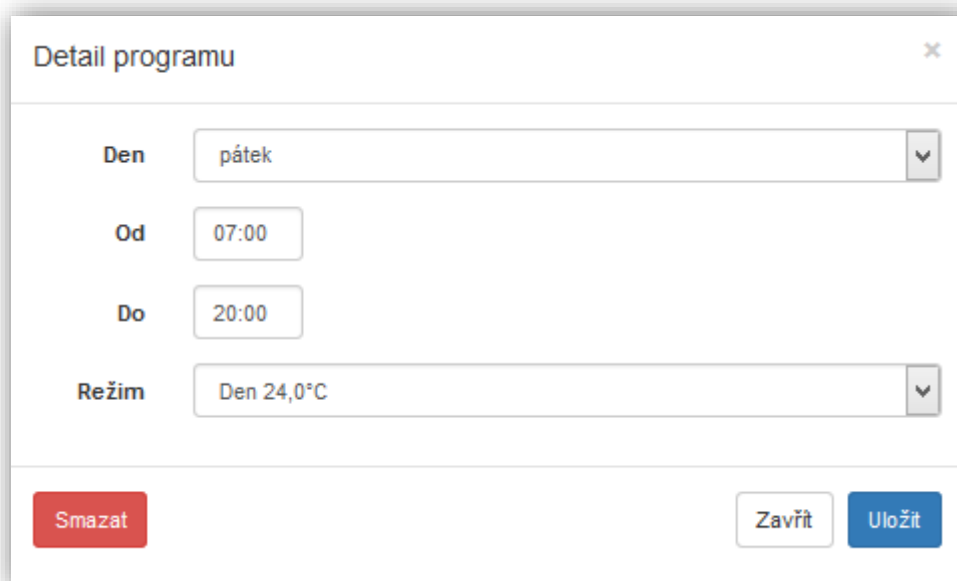
Nastavení kalendáře

	pondělí	úterý	středa	čtvrtek	pátek	sobota	neděle
Celý den	Útlum (13.0 °C)	Útlum (13.0 °C)	Útlum (13.0 °C)	Útlum (13.0 °C)	Noc (19.0 °C)	Noc (19.0 °C)	Noc (19.0 °C)
6							
7					7:00 - 20:00 Den (24.0 °C)	7:00 - 20:00 Den (24.0 °C)	7:00 - 16:00 Den (24.0 °C)
8							
9							
10							
11							
12							
13							
14							
15							
16							
17							
18							
19							

Obrázek 22 Nastavení detailu programu v týdenním kalendáři

3.6.5 Nastavení detailu položky programu v kalendáři

Po kliknutí na položku programu v kalendáři se zobrazí formulář, kde je možné upravit nastavení položky detailně. Je zde možné nastavit den, čas začátku a konce platnosti a požadovanou teplotu. Odstranění položky programu je možné kliknutím na tlačítko smazat. Po odstranění, nebo uložení hodnot položky programu je okamžitě zaktualizován program na řídicím panelu a provedena kontrola nastavení PLC, aby odpovídal novému programu.



Detail programu

Den pátek

Od 07:00

Do 20:00

Režim Den 24,0°C

Smazat Zavřít Uložit

Obrázek 23 Jemné nastavení průběhu programu v kalendáři

3.7 Mobilní klient

Pro mobilního klienta jsou dvě možnosti realizace rozhraní a to webové rozhraní, nebo nativní aplikace.

3.7.1 Webové rozhraní

Výhodou mobilního webového rozhraní je možnost sdílení většiny kódu s rozhraním pro větší displeje. Podmínkou je použití technologie responsivního designu. Responsivním designem rozumíme takové rozhraní, jehož vlastnosti se přizpůsobí schopnostem klienta. Mobilní klient má typicky menší obrazovku a může otáčet displejem na šířku/výšku. Na dotykových displejích není možné využít některé vlastnosti ovládání jako ostatních zařízení. Například událost drag&drop. Mobilní webová aplikace bude používat stejné prvky jako rozhraní popsané v kapitole 3.5. , pouze s výjimkou nastavení programu kalendáře. To bude z důvodu nedostatku místa na displeji zjednodušeno, jak je znázorněno na obrázku níže.

Název

Pouze víkend Uložit

Nastavení kalendáře

pondělí úterý středa čtvrtek pátek sobota neděle

Noc (19.0 °C)

07:00 - 20:00 Den (24.0 °C)

← Zpět

Obrázek 24 Zjednodušený kalendář
u mobilního webového rozhraní

Funkce kalendáře zůstala téměř stejná, pouze zde není podporována u položek možnost editace metodou drag&drop. Je zde kvůli nedostatku místa zobrazen pouze jeden den a přepínání mezi jednotlivými dny týdne, je možné v záhlaví tabulky.

3.7.2 Nativní aplikace

Výhoda nativní aplikace tkví v tom, že je lépe integrovaná do zařízení. Ovládací prvky jsou více optimalizovány pro dané zařízení. Aplikace je výkonnější a má přímý přístup k hardware. Nevýhodou je nutnost implementace nativního klienta pro různé typy operačních systémů. Tento problém se dá částečně kompenzovat využitím vhodných knihoven a frameworku [19], kdy se vytvoří společné jádro aplikace pro všechny operační systémy a pak se již jen udělá GUI rozhraní, které se přilinkuje na sdílený kód jádra aplikace. Další komplikací je nasazení takové aplikace. Každý operační systém má vlastní obchod se softwarem pro mobilní zařízení, kam je potřeba takovou aplikaci umístit. Předtím ale musí projít certifikací u společnosti, která tento obchod s aplikacemi pro mobilní platformy spravuje.

Vzhledem k výše zjištěným faktům náročnosti nativní aplikace a danému účelu, postačí pouze implementace webového mobilního rozhraní.

3.8 Řídící panel

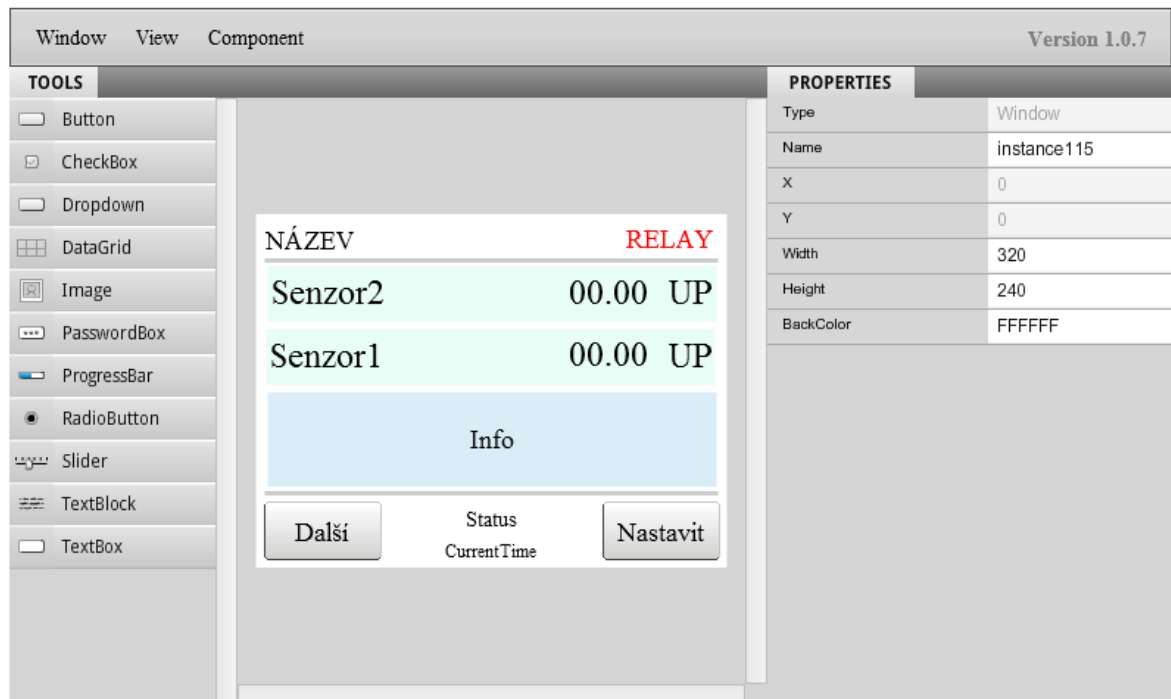
Knihovna pro displeje obsahuje standardní funkce pro psaní textů, vkládání bitmapových obrázků, nebo kreslení základních geometrických tvarů. Aby ale byl výsledek co nejlepší, tak je vhodné použít pokročilejší technologii pro ovládání obrazu, kterou je knihovna GLIDE [20]. Tato knihovna využívá stejné komponenty a principy jako známá technologie WPF [21]. Tedy jsou zde běžné základní ovládací prvky, možnosti animace, transformace, podpora virtuální klávesnice pro psaní na dotykovém displeji a mnoho dalšího.

3.8.1 Příprava fontů s podporou české diakritiky a speciálních znaků

Knihovna GLIDE má v sobě dva základní fonty. Ty v sobě ale nemají diakritiku, speciální znaky a symboly. Jako například znak stupně pro označení teploty a další speciální znaky, které je plánováno využít pro zobrazení stavu teploty a systému na displeji. Fonty lze použít jakékoliv TrueTypové, ale před použitím je potřeba je vyexportovat do formátu tinyfont, na který je přímo v SDK.NETMF [22] utilita TinyfontConverter. Smyslem konverze je maximálně ušetřit místo v paměti. Velikost souboru TrueType fontu se pohybuje v řádech jednotek MB a zmenšený optimalizovaný TinyFont je v řádech jednotek KB. Utilita se používá pomocí příkazové řádky což je poměrně nepraktické a pracné. Problém řeší produkt TinyFontTool-GUI od českého vývojáře Jana Kučery [23].

3.8.2 Návrh obrazovek

Pro návrh obrazovek WPF pro NETMF je k dispozici od dodavatele jednoduchý editor [24]. K dispozici jsou všechny základní komponenty jako u klasické aplikace pro Windows. Tedy: tlačítko, zaškrtačací tlačítko, roletkové menu, tabulka, obrázek, textové políčko a další. Výhoda je, že je zde implementována i podpora pro dotykové displeje. Takže například pokud je dotekem aktivováno textové políčko, tak se na displeji zobrazí virtuální klávesnice pro zadání textu, podobně jako je zvykem u displejů na mobilních zařízeních. Po vytvoření šablony se uloží Xaml soubor do projektu ve Visual Studiu do složky Resources. Každá obrazovka bude mít vlastní šablonu.



Obrázek 25 Editor pro návrh obrazovky řídicího panelu

3.8.2.1 Hlavní obrazovka

Bude zobrazovat nejdůležitější informace, tedy teplotu a stav jestli je vyhřívání aktivní, nebo ne. Pak ještě je důležitá informace jaký program právě probíhá a co bude v nejbližší době následovat.



Obrázek 26 Řídicí panel - hlavní obrazovka

Po kliknutí na tlačítko Další se zobrazí informace z dalšího PLC, které je zapojeno v systému. Tlačítko Nastavit přepne obrazovku na nastavení daného programu pro zvolené PLC. Dole je ještě zobrazen aktuální stav a čas systému.

3.8.2.2 Výběr a nastavení programu

Na řídicím panelu je potřeba umožnit změnit alespoň základní vlastnosti. Tedy vyměnit právě probíhající program za nějaký jiný.



Obrázek 27 Řídicí panel – nastavení programu

Seznam dostupných programů se zobrazí po rozkliknutí rolovacího seznamu. Případně umožnit uživateli upravit stávající program nastavením offsetové hodnoty pro požadovanou

teplotu. Na to je použita komponenta Slider, která umožňuje plynule posouvat posuvníkem a nastavit tak hodnotu od -4°C do $+4^{\circ}\text{C}$.

Defaultně je nastaveno 0°C , tedy že program pracuje přesně podle původně nastavených hodnot. Tlačítkem Zpět se pak uživatel dostane zpět na hlavní obrazovku.

4 IMPLEMENTACE

Než bude započato s implementací, tak je potřeba udělat analýzu požadavků na prostředky, které budou potřeba pro vývoj.

4.1 Vývojové prostředí

Jako vývojové prostředí lze použít Visual Studio 2013 Community Edition, které je zdarma. Dále bude využita pro vývoj serverové části nejaktuálnější verze .NET frameworku 4.5. Pro řídicí panel, kde je NET Microframework bude použita aktuální verze NETMF-4.3.

4.1.1 Emulátor řídicího panelu

Pro efektivní vývoj řídicího panelu, je dobré využít emulátoru, který je dodáván v rámci SDK popsaného v kapitole 4.1. V základu má v sobě implementován malý dotykový displej, 5 tlačítek, síťové rozhraní a virtuální datové úložiště. Pro účel aplikace to bude stačit a nebude nutné dalších rozšíření.



Obrázek 28 Emulátor řídicího panelu

4.1.2 Unittesty

Vzhledem k tomu, že se bude na řídicím panelu implementovat vlastní řešení SignalR a bude se řešit serializace a deserializace datových struktur na řídicím panelu, tak bude vhodné si na tyto operace napsat základní testy. K tomu se využije emulátor, společně s knihovnou MFUnitTest, která se dá jednoduše doinstalovat přes balíčkovací systém NuGet ve Visual Studiu. Pro serializaci a deserializaci formátu JSON bude použita knihovna Json.NETMF [25], která je dostupná rovněž přes NuGet.

4.1.3 Emulátor serverové části

Serverové API bude provozováno v produkční verzi na službě Azure website. Pro vývoj bude ale využita schopnost Visual Studia spouštět a ladit programový kód lokálně. Visual Studio projekt spustí jako lokální webovou aplikaci, kde jsou k dispozici všechny funkce podobně jako na produkčním prostředí.

4.1.4 Řídicí panel

Pro vývoj bude využito SDK NETMF a SDK od firmy GHI Electronics, které rozšíří funkci Visual Studia o rozhraní pro návrh

4.1.5 Webové rozhraní

Pro webové rozhraní bude použita knihovna Bootstrap, jejíž výhody byly popsány v kapitole 3.5. Pro spojení s API serverem bude použita javascript knihovna SignalR a nainstaluje se do projektu pomocí NuGet. Současně s knihovnou SignalR se automaticky nainstaluje i závislá javascriptová knihovna jQuery. Tuto knihovnu je možné rovněž využít pro manipulaci s dynamickým HTML obsahem, při zpracování událostí ze serveru.

4.1.6 Mobilní klient webový

Knihovny budou použity stejné, jako u webového rozhraní popsaného v kapitole 4.1.5.

4.2 Implementace API serveru

Pro implementaci SignalR API bude použit návrhový vzor singleton, protože bude vyžadováno, aby si instance API držela informaci o aktuálně připojených klientech. Je to i jeden z doporučených postupů na webu dodavatele.

Byla tedy vytvořena třída `SignalrConnection`, která v sobě bude držet informace potřebné pro distribuované řízení připojených klientů:

- **Instance**
Zde je uložena instance třídy `SignalrConnection` z `ConnectionManageru`, jako společný singleton pro všechny připojené klienty
- **Clients**
Umožní volat ze serveru metody implementované na klientech
- **ControllerConnectionID**
Jednoznačný identifikátor (token) řídicího panelu
- **ControllerConnectionDate**
Datum připojení řídicího panelu k API

Třída `ConnectionHub` implementuje jednotlivé veřejně dostupné metody, které mohou klienti na serveru volat. Jejich seznam byl popsán v kapitole 3.4.1 v odstavci věnovanému metodám klient->server

Hlavní body implementace je pojmenování API pomocí anotace u definice třídy:

```
[HubName("connectionHub")]
```

Tento název pak budou používat všichni klienti pro identifikaci, na který HUB se chtějí připojit. Je to z toho důvodu, že těchto API zde může být implementováno víc.

Jednotlivé metody v této třídě, pokud mají být dostupné pro klienty z venku, stačí označit modifikátorem `public` jako veřejné. Je zde ještě speciální vlastnost, na kterou je potřeba si dát pozor a to že názvy metod musí začínat malým písmenem, aby bylo možné jejich bezproblémové volání ze strany Javascriptu. Jiní klienti SignalR tento problém nemají.

Připojení na zdroj je řešeno přes standardní `SqlDataAdaptéry` jak ukazuje Obrázek 14.

`SqlDataAdaptéry` umožňují dělat nad databázovými objekty standardně všechny CRUD operace, což pro účely projektu stačí a není potřeba implementovat další metody.

Zabezpečení přístupu na API je řešeno standardně přepnutím API do režimu, který vyžaduje autentizaci pro všechny metody. To se řeší ve třídě `Startup1.cs`, kde se provádí inicializace webového rozhraní OWIN. Zde stačí pouze zavolat metodu:

```
GlobalHost.HubPipeline.RequireAuthentication();
```

Dále se v konfiguraci `Web.Config` nastaví autentizace do režimu „Form“ a nastaví se adresa pro přihlašovací formulář „`Login.aspx`“. Od teď jakýkoliv přístup, který není autentizovaný,

bude přesměrován na přihlašovací stránku. Detail implementace přihlašovací stránky bude popsán v následující kapitole.

Jediný klient, který nebude využívat přihlašovací stránku, bude řídicí panel. Tento klient bude pro autentizaci používat trvalý autentizační token. Alternativně by šlo v případě potřeby udělat autentizaci dynamickou, že řídicí panel zavolá autentizační službu na serveru, která mu vygeneruje dočasný token podle zadaných přihlašovacích údajů.

4.3 Implementace Webového rozhraní

Pro oddělení společného kódu nebyla použita masterpage šablona, protože zde jsou jen čtyři stránky a z toho jen dvě budou mít společný kód pro inicializaci realtime spojení na API.

Společnou částí jsou tedy pouze styly, které jsou řešeny přes společné téma v adresáři ~/App_Themes/Default.

4.3.1 Přihlašovací formulář

Autentizace využívá společnou konfiguraci s API serverem. Autentizace je řešena prozatím staticky pro jeden daný login/heslo. Je zde implementována ochrana proti neoprávněným pokusům o prolomení hádáním loginu/hesla. Po 10-ti chybných přístupech je další pokus o autentizaci ze stejné IP adresy na 10minut zamítnut.

4.3.2 Hlavní obrazovka

V záhlaví stránky je provedena inicializace Javascriptu a CSS knihovny Bootstrap. Vlastní HTML šablona pro vykreslení je velmi jednoduchá. Skládá se jen ze dvou panelů pro zobrazení stavu PLC a několika dalších infopanelů. Po načtení HTML je spuštěn javascriptový kód, který provede vytvoření perzistentního spojení s API serverem. Pokud se vše zadaří, tak následně ještě zavolá na serveru metodu, že požaduje po řídicím panelu aktualizovat data z PLC. Z API pak přijde struktura s požadovanými daty ve formátu JSON, takže stačí jednoduše vzít položky z pole a přes DOM objekt zaktualizovat údaje na stránce.

4.3.3 Nastavení programu

Tato stránka nemá realtime propojení s API, není to nezbytně nutné pro požadovanou funkci. Na stránce je zobrazen seznam dostupných programů, po kliknutí na program se provede postback a na straně serveru se zavolá metoda API na změnu programu napřímo.

Pro nastavení offsetové hodnoty pro požadovanou teplotu je řešeno sadou tlačítek. Pro lepší objektové oddělení byl tento prvek vyřešen jako webový uživatelský ovládací prvek, který má veřejnou událost `OnClick()` a `DefaultValue`, pomocí které je možné nastavit výchozí hodnotu prvku.

4.3.4 Nastavení položek programu v kalendáři

Prakticky téměř celá funkce stránky je vyřešena v Javascriptu komponentou `FullCalendar` [16], která je doplňkem do `jQuery`. Stačilo se napojit na události kalendáře a změněná data posílat na server. Byla zde ještě navíc implementována funkce pro změnu názvu programu a odstranění položky programu. Změna nastavení kalendáře se okamžitě projeví na řídicím panelu a PLC. Realtime změna nastavení kalendáře na uživatelském rozhraní podle událostí ze serveru není prozatím implementována.

4.4 Implementace řídicího panelu

Program řídicího panelu se bude skládat ze dvou modulů a několika sdílených knihoven.

4.4.1 Implementace knihovny klienta `SignalR` pro `NETMF`

Jak již vyplynulo z analýzy, `SignalR` nemá zatím vyřešenou implementaci klienta pro `NETMF`. Z toho důvodu bylo potřeba naimplementovat alespoň nejzákladnější funkce tak, aby bylo možné se na API z `NETMF` připojit. Detailně je protokol popsán na stránkách dodavatele [15]. Zde tedy budou pouze zdůrazněny pouze základní funkce, důležité pro provoz.

Při prvním spojení je nutno provést autentizaci, protože API jinak všechny požadavky zamítne. Zjednodušená implementace autentizace spočívá pouze v tom, že do hlavičky všech odchozích spojení na API, bude přidáván autentizační token. Jeho délka je 192znaků. Bude trvale uložen na řídicím panelu, případně na SD kartě jako součást konfigurace řídicího panelu. Autentizační token byl získán jednorázovým vygenerováním na serveru.

Vlastní připojení k API je sled několik požadavků na API, které probíhají v tomto pořadí

- **Negotiate**

Zde dojde k prvotnímu spojení mezi serverem a klientem. Klient obdrží identifikační kód klienta, kterým se musí pro každé další volání v API identifikovat.

- **Connect**

Pro klienta nemá prakticky žádný vliv. Na server se zavolá metoda `OnConnected()`

- **Start**

Dá serveru pokyn, že řídicí panel je připraven přijímat požadavky ze serveru. V odpovědi se vrátí potvrzení „started“, což znamená, že server požadavek zpracoval.

Pak musí následovat udržování spojení, které je možné udržovat buď permanentně otevřeným soketem, nebo pomocí tzv. dlouhých volání. Pro zjednodušení implementace bude použit systém dlouhých volání. Tedy pošle se HTTP požadavek typu GET na server, ve kterém je zpráva pro server „poll“ a čeká se na odezvu od serveru. V případě, že ze serveru nepříjde odezva do vypršení timeoutu, tak je výjimka odchycena a bude vytvořeno nové spojení „poll“. Tyto operace jsou spouštěny v separátních vláknech, aby neblokovaly ostatní procesy.

Zprávy ze serveru přichází ve formátu JSON. Na deserializaci byla použita knihovna JSON.NETMF. Zprávy odesílané na server musí být rovněž serializovány do formátu JSON. Bylo nutno vyřešit serializaci polí a složených strukturovaných dat.

4.4.2 Implementace knihovny pro komunikaci s PLC

Pro přístup k PLC byla vytvořena třída DeviceDriver.Devices.IpSmartboard.

Byly v ní implementovány jen dvě funkce. Načtení aktuálního stavu senzorů a relé a funkce pro nastavení požadavku udržování dané teploty.

- Načtení stavu PLC

Dle dokumentace [9] se stav načítá pomocí metody GET z adresy /status.xml

Struktura dat je ve formátu XML, který je potřeba přeformátovat do struktury, která je definována pro přenos do API. Jsou zde informace o teplotě z obou senzorů a stav sepnutí/rozepnutí relé.

- Nastavení požadavku na udržování dané teploty

Nastavení požadovaných parametrů se realizuje pomocí metody POST na adrese /C3.

Parametry jsou formátovány standardně pomocí klíče a hodnoty.

4.4.3 Připojení k API

Pro správnou funkci systému je nutné zajistit synchronizaci systémových hodin. V SDK NETMF je na to knihovna TimeService. Po jejím spuštění a nakonfigurování zajišťuje periodickou synchronizaci systémových hodin s veřejným NTP serverem.

Pro připojení je použita knihovna SignalR.NETMF popsaná v předchozí kapitole. Pomocí ní je možné načíst do paměti mikrokontroléru konfiguraci pro všechna PLC, jejich nastavení

a programy. V případě, že dojde k přerušení spojení panelu a API, nebo jiné poruše, tak vlákno, které má za úkol udržovat spojení a je spuštěno ve smyčce, tak opakovaně zkouší v krátkých intervalech obnovit spojení. Po úspěšném spojení s API serverem je načten stav všech PLC a na displeji je pomocí třídy Display vykreslena obrazovka hlavní obrazovka pro první PLC které je v tabulce dostupných PLC.

4.4.4 Displej

Byla vytvořena třída Display, která má na starosti vykreslování obsahu na displej a obsluhovat události na ovládacích prvcích jako jsou tlačítka, přepínače a roletkové menu.

Třída načítá podle potřeby z resources aplikace soubory XAML s definicí prvků obrazovky a potřebné fonty pro vykreslení textu a symbolů.

Pro každou obrazovku byla vytvořena třída, která řeší vykreslení, aktualizací údajů a obsluhu událostí na aktivních formulářových prvcích.

Pro efektní animované přechody při přepínání mezi jednotlivými obrazovkami byla použita knihovna Tweens, která je součástí SDK NETMF [22].

5 BEZPEČNOSTNÍ AUDIT

Po spuštění systému do zkušebního provozu bylo potřeba celý systém zanalyzovat a otestovat, jestli v něm nejsou nějaké bezpečnostní chyby. Detailně je tato problematika popsána v publikaci Počítačová bezpečnost a ochrana dat. [3].

5.1 Bezpečnost komunikace

Veškerá komunikace je provozována přes protokol HTTPS, který používá asymetrické šifrování. Server i klient si vygenerují veřejný a privátní klíč. Navzájem si pak mezi sebou vymění veřejné klíče. Každá z obou stran si ověří, jestli veřejný klíč je podepsán nadřazenou certifikační autoritou, které důvěřuje a zkontroluje, jestli nevypršela platnost podpisových certifikátů. Následně by ještě měla proběhnout kontrola, že druhá strana je skutečně ta, se kterou chce první strana komunikovat. Tento poslední bod bohužel nešel na řídicím panelu splnit, protože v NETMF nejsou proti plnému NET Frameworku některé pokročilé funkce implementovány. Tento poslední problém tedy otevírá možnost využít prolomení bezpečnosti známou metodou „Man in the middle“. Jeho provedení je ale poměrně náročné a vyžaduje přístup útočníka na páteřní routery. Proto bylo toto riziko vyhodnoceno jako minimální. V případě, že by byla vyžadována kontrola certifikátů, tak je zde možnost každému řídicímu panelu vystavit vlastní certifikát. To je ale nepraktické řešení, Vzrůstá tak náročnost údržby o periodickou aktualizaci certifikátů, kvůli jejich omezené době platnosti. Nejvhodnějším řešením je využít autentizační token, který má omezenou časovou platnost a řídicí panel by byl schopen si tento klíč sám periodicky ze serveru aktualizovat, kdy by pro autentizaci operace aktualizace klíče, použil stávající platný klíč.

Současné řešení tedy v rámci plánovaného užití má zabezpečení dostatečné, ale do budoucna je dobré zvážit systém automatické aktualizace autentizačních klíčů.

5.2 Odolnost proti SQL Injection

Další typickou metodou útoku je SQL Injection. Tento typ útoku je založen na vkládání kousků SQL příkazů do uživatelského rozhraní tak, aby útočník získal citlivá data, nebo alespoň data poškodil. Jedná se o poměrně starou metodu a již jsou k dispozici nástroje, jak se proti ní účinně bránit. Všechna data, která přichází od uživatele na server, jsou potenciálně nebezpečná a možnosti ochrany jsou dvě:

- Používat databázovou vrstvu, která odděluje SQL dotazy od uživatelských dat

- Ošetřit vstup uživatele tak, aby nebyl schopen vložit nebezpečné znaky.

V tomto projektu byla využita databázová vrstva ADO.NET. Veškeré vstupy jdou tedy přes databázovou vrstvu a systém je proti tomuto typu útoku zabezpečen.

5.3 Odolnost proti neoprávněnému přístupu

Všechny veřejně dostupné funkce serveru, tedy API i webové rozhraní jsou jednotně chráněny standardním autentizačním systémem. Navíc pokud útočník nezná heslo, tak mu nezbývá než náhodně zkoušet kombinace. V projektu byla implementována ochrana proti tomuto typu útoku tak, že po 10-ti chybných pokusech o přihlášení dojde zablokování dalších přístupů z IP adresy útočníka na 10minut. Klient je přesměrován na přihlašovací obrazovku a je mu zobrazena zpráva „Překročen povolený počet přihlášení. Přístup z Vaší IP adresy byl zakázán“. Vhodným rozšířením by bylo monitorovat množství neplatných pokusů o přihlášení, aby byl včas odhalen distribuovaný útok. Tedy systém, kdy útočník používá pro útok velké množství veřejných IP.

Stávající zabezpečení proti neoprávněnému přístupu je dostatečné. V případě, že by takový útok probíhal delší dobu tak by bylo vhodné limit 10-ti minut zvýšit.

5.4 Odolnost proti DDoS útoku

Tímto typem útoku se útočník snaží zahltit službu velkým množstvím požadavků tak, aby způsobil její výpadek. Typicky tento útok probíhá distribuovaně z velkého množství IP adres a ruční omezování přístupu IP adres na firewallu nemá praktický význam. Cloudová řešení ale typicky poskytují ochranu proti těmto útokům. Azure website má řešení tohoto problému, nicméně by to znamenalo další měsíční výdaje navíc. Aktivace tohoto typu ochrany by zabrala asi 1-2hodiny. Spustila by se služba Load balanceru, která tento problém bude řešit. Detailněji je toto popsáno v dokumentaci služby Azure Threat Management [26] v kapitole Microsoft Antimalware for Azure with real-time protection and remediation.

Proti tomuto typu útoku je možné se účinně bránit, ale vzhledem k tomu, že se v této chvíli nejedná o komerční projekt a potencionální riziko je velmi malé, tak problém by se řešil, až by nastal.

5.5 Odolnost proti výpadkům a poruchám

Další oblastí bezpečnosti je odolnost proti náhodným poruchám spojení. Bylo nutné tuto oblast prověřit a zhodnotit rizika a dopady jednotlivých druhů poruch.

5.5.1 Senzory

V případě že jeden ze dvou tepelných senzorů přestane fungovat, tak je o tom uživatel informován na řídicím panelu. Může závadu odstranit, nebo nouzově může úpravou konfigurace nastavit PLC tak, aby pro řízení teploty využíval druhý senzor, Tedy pokud je tím senzorem měřena teplota podobného charakteru. Například senzor ve vedlejší místnosti.

Dalším typem možné poruchy senzoru je elektromagnetické rušení. Typicky se to stává na dlouhém vedení, kde se indukují parazitní proudy.

Toto riziko bylo již při návrhu minimalizováno tak, že od PLC jsou senzory vzdáleny v rozmezí 1m až max. 5m. a byl použit kabel se stíněním.

5.5.2 PLC

Výpadek PLC způsobí nemožnost jakékoliv kontroly nad systémem. V případě poruchy PLC je o této poruše uživatel informován na řídicím panelu. Zde již musí uživatel zasáhnout a vyměnit vadný kus. Případně zkontrolovat kabeláž, jestli je v pořádku. Diagnostika chyby PLC je velmi jednoduchá. Vzhledem k tomu že PLC má na sobě WWW rozhraní, tak jde test udělat z libovolného WWW prohlížeče.

5.5.3 Relé

Výpadek relé nejde zjistit přímo. Není na to implementován žádný dodatečný senzor. Výpadek relé pak lze zjistit pouze nepřímo, tedy tak, že PLC není schopno udržet požadovanou teplotu, přestože všechny systémy hlásí, že je všechno v pořádku. Tento problém nebyl vyhodnocen jako kritický, když má uživatel možnost informaci o výpadku získat. Nicméně v případě potřeby je možné k výkonové části přidat kontrolní senzor, který bude hlídat, jestli výkonová část je skutečně sepnuta. PLC má k dispozici další digitální i analogové vstupy, do kterých by bylo možné tento dodatečný senzor připojit.

Výpadek tohoto typu není kritický, výměnu vadného kusu v patici je možno provést s reakční dobou i do několika hodin.

5.5.4 Řídící panel

V případě výpadku řídicího panelu uživatel vidí přímo, že je problém s displejem a vzdálený uživatel vidí na WWW rozhraní, že je řídicí panel odpojen. Tento problém je nutné vyřešit výměnou vadného řídicího panelu. Systém je navržen tak, že PLC i bez přítomnosti řídicího panelu nadále udržují poslední zvolený program. V případě nouze, je možné PLC monitorovat a konfigurovat přímo přes WWW prohlížeč bez nutnosti používat distribuovaný systém řízení.

5.5.5 Server API

Server je používán pro uchování a načítání konfigurace dat pro řídicí panel. Pokud je řídicí panel spuštěn a dojde k výpadku serveru, nebo konektivity, tak vnitřní systém funguje bez problému. Uživatel může sledovat stav systému na panelu a měnit program na jednotlivých PLC. Problém by nastal, pokud dojde k restartu řídicího panelu a po jeho opětovném startu nebude k dispozici spojení na API server. Tento problém je vyřešen zálohovaným napájecím zdrojem. Další možností je doimplementovat ukládání lokálních dat při každé jejich změně na připojenou SD kartu. Po restartu řídicího panelu se data z karty použijí pro inicializaci lokálních datových struktur, a pokud pak selže synchronizace se serverem, tak zůstane načten poslední stav, který byl před restartem, tedy řídicí panel bude fungovat bez problému.

ZÁVĚR

Cílem práce bylo vylepšit stávající systém řízení otopné soustavy v rodinném domě. Byla provedena analýza dostupných řešení na trhu a byla zvolena ekonomicky nejvýhodnější varianta. Bylo vyvinuto embedded řešení s dotykovým panelem, které slouží jako řídicí panel pro uživatele v interiéru budovy a současně slouží i jako centrální prvek distribuovaného řízení pro podřízené subsystemy. Dále bylo vyvinuto veřejně dostupné webové rozhraní s responzivním designem tak, aby mohli uživatelé vzdáleně přistupovat přes WWW prohlížeč jak z klasického PC, tak i z mobilních zařízení.

Hlavním přínosem práce je zvýšení uživatelského komfortu. Uživatel má větší kontrolu nad systémem. Lépe dokáže detekovat a zjistit poruchu. Informace má přístupné odkudkoliv z veřejné internetové sítě.

S ohledem na implementační náročnost a nutnost řešení vlastní knihovny NETMF pro komunikaci s API lze usoudit, že by byla vhodná i jiná alternativa. Tedy střední varianta firmy Papouch s využitím 19“ dotykového displeje, která má o něco vyšší pořizovací cenu, ale část řídicího panelu by měla být implementovatelná poměrně jednoduše. Nabídka od firmy Te-comat je komplexní a hotové řešení. Cena ale převyšuje pořizovací náklady prvních dvou řešení, a to i po započítání ceny za implementaci SW části.

SEZNAM POUŽITÉ LITERATURY

1. **Case, a další.** A Simple Network Management Protocol (SNMP). *Internet Engineering Task Force*. [Online] [Citace: 1. 5 2015.] <https://www.ietf.org/rfc/rfc1157.txt>.
2. **Fielding, R., Irvine, U. a Gettys, J.** Hypertext Transfer Protocol. *Internet Engineering Task Force*. [Online] 1999. [Citace: 6. 2 2015.] <https://www.ietf.org/rfc/rfc2616.txt>.
3. **Doseděl, Tomáš.** *Počítačová bezpečnost a ochrana dat*. Brno : Computer Press, 2004. 80-251-0106-1.
4. **Programovatelné automaty Tecomat foxtrot CP-1000.** *Tecomat*. [Online] 2015. [Citace: 10. 3 2015.] http://www.tecomat.com/wpimages/other/DOCS/cze/TXV00430_01_Foxtrot_CP_1000.pdf.
5. **Quido ETH 4-4 - 4 vstupy, 4 výstupy a teploměr.** *Papouch*. [Online] 2015. [Citace: 1. 5 2015.] <http://www.papouch.com/cz/shop/product/quido-eth-4-4-vstupy-vystupy-teplomer-ethernet/12v/>.
6. **Lenovo IdeaCentre C260 Touch Black.** *Alza*. [Online] 2015. [Citace: 1. 5 2015.] <https://www.alza.cz/lenovo-ideacentre-c260-touch-d2218438.htm>.
7. **GHI Electronics.** Fez spider Tinker Kit. *GHI Electronics*. [Online] 2015. [Citace: 15. 1 2015.] <https://www.ghielectronics.com/catalog/product/529>.
8. **Robenek, Jan.** Ethernetový radič s hardwarovým šifrováním. *HW.CZ*. [Online] 2014. [Citace: 26. 12 2014.] <http://www.hw.cz/soucastky/embedded-systemy/mcu/ethernetovy-radic-s-hardwarovym-sifrovanim.html>.
9. **IQtronic technologies Europe.** IP SMART BOARD manuál. *IQtronic technologies Europe*. [Online] 2014. [Citace: 20. 12 2014.] <http://www.mikrovlny.cz/content/file/manualy/IPBOARD.pdf>.
10. **Tamra Myers.** How to use Table storage from .NET Microsoft Azure. *Microsoft Azure*. [Online] [Citace: 20. 3 2015.] <http://azure.microsoft.com/cs-cz/documentation/articles/storage-dotnet-how-to-use-tables/>.
11. **GM electronic.** DS18B20 teplotní senzor. *GM electronic*. [Online] 2014. [Citace: 12. 1 2015.] <http://www.gme.cz/ds18b20-p530-067>.

12. GM electronic. Relé s DC cívkou 12V FINDER 40.51.9.012.0000. *GM electronic*. [Online] 2014. [Citace: 22. 12 2014.] <http://www.gme.cz/rele-s-dc-civkou-12v-finder-40-51-9-012-0000-p634-640>.
13. Common Scenarios with Windows Azure Mobile Services . *Chris Risner*. [Online] 2015. [Cited: 5 2, 2015.] <http://chrisrisner.com/Common-Scenarios-with-Windows-Azure-Mobile-Services>.
14. XSocket - mature real-time communication platform. *XSockets.NET*. [Online] [Citace: 2. 5 2015.] <http://xsockets.net/>.
15. SignalR Guidance. *The ASP.NET Site*. [Online] Microsoft. [Citace: 2. 5 2015.] <http://www.asp.net/signalr/overview>.
16. Shaw, Adam. JavaScript Event Calendar (jQuery plugin). *FullCalendar*. [Online] 2015. [Citace: 6. 2 2015.] <http://fullcalendar.io/>.
17. Security, Authentication, and Authorization in ASP.NET Web Forms. *The ASP.NET Site*. [Online] Microsoft. [Citace: 2. 5 2015.] <http://www.asp.net/web-forms/overview/security>.
18. *Bootstrap - The world's most popular mobile-first and responsive front-end framework*. [Online] [Citace: 15. 4 2015.] <http://getbootstrap.com/>.
19. Mobile Application Development to Build Apps in C#. *Xamarin*. [Online] [Citace: 19. 1 2015.] <http://xamarin.com/platform>.
20. GHI Electronics. Glide. *GHI Electronics*. [Online] GHI Electronics, 2015. [Citace: 12. 1 2015.] <https://www.ghielectronics.com/glide/>.
21. Windows Presentation Foundation. *MSDN Library*. [Online] [Citace: 9. 4 2015.] [https://msdn.microsoft.com/en-us/library/ms754130\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/ms754130(v=vs.110).aspx).
22. .NET Micro Framework. *Codeplex*. [Online] Microsoft. [Citace: 11. 1 2015.] <http://netmf.codeplex.com/>.
23. Kučera, Jan. Tiny Font Tool GUI. *.NET Micro Framework*. [Online] 2013. [Citace: 25. 1 2015.] <http://informatix.miloush.net/microframework/Utilities/TinyFontTool.aspx>.
24. Glide Designer. *GHI Electronics*. [Online] [Citace: 26. 3 2015.] <https://www.ghielectronics.com/glide/designer>.

25. Weimer, Matt. *Json.NetMF*. *GitHub*. [Online] [Citace: 18. 12 2014.]

<https://github.com/mweimer/Json.NetMF>.

26. Security & Threat Management. *Microsoft Azure Trust Center*. [Online] [Citace:

15. 5 2015.] <http://azure.microsoft.com/en-in/support/trust-center/security/threat-management/>.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

API	Application Programming Interface
DDoS	Distributed Denial of Service
PLC	Programmable Logic Controller
POE	Power over Ethernet
SSL	Secure Socket Layer
WPF	Windows Presentation Foundation
XAML	Extensible Application Markup Language

SEZNAM OBRÁZKŮ

Obrázek 1 Senzor DS18B20	18
Obrázek 2 Výkonové relé (2,2KW)	19
Obrázek 3 PLC IP Smartboard	19
Obrázek 4 Napájecí adaptér 12V DC 1500mA	20
Obrázek 5 POE splitter	20
Obrázek 6 Základní vývojová sada FEZ Spider Tinker Kit	21
Obrázek 7 Blokové schéma systému	22
Obrázek 8 Schéma zapojení PLC, senzorů relé a společného napájení.....	23
Obrázek 9 Visual Studio designér - schéma zapojení řídicího panelu.....	24
Obrázek 10 Zjednodušené schéma distribuovaného řízení.....	31
Obrázek 11 Struktura konfigurace PLC.....	33
Obrázek 12 Struktura pro změřená data	34
Obrázek 13 Struktury pro přenos konfigurace programu	34
Obrázek 14 Celkové schéma databáze.....	35
Obrázek 15 Ukázka tabulky Device	35
Obrázek 16 Ukázka tabulky DeviceProgram.....	36
Obrázek 17 Ukázka tabulky DeviceProgramData	36
Obrázek 18 Ukázka tabulky DeviceMode	36
Obrázek 19 Přihlašovací obrazovka	39
Obrázek 20 Úvodní obrazovka, přehled základních informací	39
Obrázek 21 Detail nastavení PLC, výběr programu	41
Obrázek 22 Nastavení detailu programu v týdenním kalendáři	42
Obrázek 23 Jemné nastavení průběhu programu v kalendáři	43
Obrázek 24 Zjednodušený kalendář u mobilního webového rozhraní	44
Obrázek 25 Editor pro návrh obrazovky řídicího panelu	46
Obrázek 26 Řídicí panel - hlavní obrazovka	47
Obrázek 27 Řídicí panel – nastavení programu.....	47
Obrázek 28 Emulátor řídicího panelu	49

SEZNAM TABULEK

Tabulka 1 Hardware - cenová kalkulace Tecomat.....	14
Tabulka 2 Hardware - cenová kalkulace Papouch.....	14
Tabulka 3 Hardware - cenová kalkulace GHI Electronics.....	15
Tabulka 4 Příklad užití - zobrazení stavu systému na vzdáleném rozhraní.....	25
Tabulka 5 Příklad užití - autentizace nepřihlášeného uživatele.....	25
Tabulka 6 Příklad užití - změna programu ze vzdáleného rozhraní.....	26
Tabulka 7 Příklad užití - korekce žádané teploty ze vzdáleného rozhraní.....	27
Tabulka 8 Příklad užití - detailní nastavení programu ze vzdáleného rozhraní.....	27
Tabulka 9 Příklad užití - zobrazení stavu na řídicím panelu.....	28
Tabulka 10 Příklad užití – změna zobrazení stavu na řídicím panelu.....	29
Tabulka 11 Příklad užití - změna programu na řídicím panelu.....	29
Tabulka 12 Příklad užití - korekce žádané teploty z řídicího panelu.....	30
Tabulka 13 Příklad užití - periodická aktualizace stavu.....	30