FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Diplomová práce

Webové rozhraní inteligentního domu pro PLC Siemens S7-1200

Studijní program: N0714A270010 Mechatronika
Autor práce: **Bc. Petr Javorský**
Vedoucí práce: Ing. Lukáš Hubka, Ph.D.
Ústav mechatroniky a technické informatiky

Liberec 2023

FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Zadání diplomové práce

Webové rozhraní inteligentního domu pro PLC Siemens S7-1200

Jméno a příjmení:	**Bc. Petr Javorský**
Osobní číslo:	M21000177
Studijní program:	N0714A270010 Mechatronika
Zadávající katedra:	Ústav mechatroniky a technické informatiky
Akademický rok:	2022/2023

Zásady pro vypracování:

1. Seznamte se s možnostmi webového serveru implementovaného v PLC S7-1200.
2. Prozkoumejte možnosti tvorby dynamických webových stránek, stránek s dynamicky měněným obsahem pro web server PLC S7-1200 (HTML5, JavaScript, ...). Identifikujte a porovnejte možnosti freeware a open-source nástrojů pro tvorbu Smart HMI / SCADA aplikací.
3. Vytvořte vzorovou aplikaci pro webové HMI inteligentního domu a testujte jeho vzhled na PC i v mobilním telefonu.
4. Implementujte a prozkoumejte možnosti ovládání PWM a komunikačních sběrnic pomocí webového rozhraní.
5. Vytvořte knihovnu komponent a funkcí pro TIA STEP7 pro potřeby webového HMI inteligentních domů.
6. Sestavte podrobnou dokumentaci a metodický pokyn pro vytvoření vlastních stránek.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	40-50 stran
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	Čeština

Seznam odborné literatury:

[1] Creating User-Defined Web Pages for S7-1200/S7-1500 [online]. V4.0 03/2020. Siemens, 2020 [cit. 2022-10-17]. Dostupné z: https://support.industry.siemens.com/cs/attachments/68011496/68011496_S7--1200_1500_Webserver_DOC_v4_en.pdf

[2] Basics on Creating HTMLs for SIMATIC CPUs: Application Description [online]. V1.0 02/2014. Siemens, 2014 [cit. 2022-10-17]. Dostupné z: https://support.industry.siemens.com/cs/attachments/68011496/68011496_html_basics_for_simatic_cpus_en.pdf

[3] Creating and using user-defined web pages on S7-1200 / S7-1500: ID: 68011496. Industry Support Siemens [online]. Siemens, 2020, 04/07/2020 [cit. 2022-10-17]. Dostupné z: https://support.industry.siemens.com/cs/document/68011496/creating-and-using-user--defined-web-pages-on-s7-1200-s7-1500?dti=0&lc=en-SK

[4] WebIQ: The Web HMI / SCADA Toolbox for Professional HMIs [online]. Meerbusch: Smart HMI, 2019 [cit. 2022-10-17]. Dostupné z: https://www.smart-hmi.com/

Vedoucí práce:	Ing. Lukáš Hubka, Ph.D. Ústav mechatroniky a technické informatiky
Datum zadání práce:	12. října 2022
Předpokládaný termín odevzdání:	15. května 2023

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

doc. Ing. Josef Černohorský, Ph.D.
vedoucí ústavu

V Liberci dne 12. října 2022

Prohlášení

Prohlašuji, že svou diplomovou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Jsem si vědom toho, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má diplomová práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

15. května 2023 Bc. Petr Javorský

Abstrakt

Tato diplomová práce se zabývá tvorbou HMI pro ovládání PLC přes webové stránky. V první části diplomové práce je popsán systém Simatic, který je v této diplomové práci využíván a je provedena rešerše již existujících HMI systémů. V praktické části diplomové práce je rozveden postup tvorby zadané webové aplikace s využitím HTML a Javascriptu a následné propojení vytvořených webových stránek s PLC umístěním webové aplikace na webový server v PLC řady Simatic. Pro snadnější ovládání PLC bude nakonec pro program TIA Portál vytvořena knihovna funkcí vytvořených pro komunikaci mezi PLC a pro vládání nových zařízení do programu.

Klíčová slova

Simatic, Siemens, webserver, HTML, Javascript, HMI, webové stránky, TIA Portal, DOM

Abstrakt

This thesis primarily focuses on the development of a Human-Machine Interface (HMI) for Programmable Logic Controller (PLC) control through web pages. The first part of the thesis describes the system that is used in this thesis and a search of existing HMI systems is conducted. In the practical part of the thesis, the process of creating the required web application using HTML and Javascript and the further interconnection of the created web pages with the PLC by placing the web application on the web server in the Simatic PLC series is described. For easier PLC control, a library of functions will eventually be created for the TIA Portal program for communication between PLCs and for adding new devices to the program.

Keywords

Simatic, Siemens, webserver, HTML, Javascript, HMI, website, TIA Portal, DOM

Obsah

Seznam obrázků

Úvod

Myšlenka inteligentního domu není nijak nová, přeci každý člověk by chtěl vědět, co se v jeho domě děje. Od včasného zjištění přítomnosti lupičů, kteří vnikli na náš pozemek po obyčejnou starost o to, zda jsme vypnuli sporák, téměř každý se chce ujistit, že při návratu domů náš dům bude stát v takovém stavu, ve kterém jsme ho opustili. Pokud k tomuto přidáme možnost náš dům ovládat tak, aby byla například při našem příchodu v domě námi nastavená teplota nebo možnost zhasnutí světel na druhé straně budovy pomocí jednoho zařízení, které máme téměř vždy u sebe, dělá z našeho domu příjemnější místo. O tyto problémy se v inteligentních domech starají nejrůznější systémy, některé obecnější pro použití v různých typech budov, od bytů po kanceláře, některé dělané na klíč pro konkrétní dům.

Cílem této diplomové práce je nejprve rešerše jednotlivých HMI systémů, které by bylo možné použít pro vytvoření vizualizace systému inteligentního domu. V další části diplomové práce bude vytvořen HMI systém využívající webserver umístěný v PLC od značky Siemens, díky kterému bude možné ovládat danou domácnost pomocí webového prohlížeče.

1. Řídící systémy SIMATIC

Pod pojmem SIMATIC se skrývá série PLC automatů vyvíjených firmou Siemens. Tyto PLC neboli programovatelné řídící automaty jsou druhem průmyslových počítačů určených pro ovládání technologických procesů v reálném čase, ku příkladu ovládání strojů či řízení nebo monitoring inteligentního domu. [1]

PLC se od běžných počítačů liší v druhu úloh, pro které je využíván a je pro tyto typy úloh přizpůsoben jak po hardwarové, tak i po softwarové stránce. Po hardwarové stránce se liší hlavně v přítomnosti portů pro čtení a zápis vnějších signálů, jak digitálních, tak analogových. V případě modulárních PLC existuje i možnost připojení přídavných modulů rozšiřující možnosti základního PLC od přídavných modulů vstupů či výstupů přes komunikační moduly po moduly řídící krokové motory. [2]

Takovéto řídící systémy existují v mnoha podobách, lišící se svou velikostí od systémů kompaktních až k systémům modulárním, které obsahují až několik desítek modulů. Dále se tyto systémy liší svým výkonem, vnitřní pamětí, nebo svými přídavnými funkcemi, jako je například vnitřní webový server nebo displej. [2]

Obecně jsou tyto systémy velmi robustní a spolehlivé, často vybavené komplexními diagnostickými funkcemi, které kontrolují činnost daného systému a umožnují rychlou realizaci projektu. Na rozdíl od řídících systémů s pevnou logikou jsou schopné pružně reagovat na změny v zadání projektu, v lepším případě přepsáním řídícího programu, v horším zakoupením a připojením potřebného modulu. [2]

Momentálně nejnovější řadou od Siemensu je SIMATIC S7, která je tu s námi již od roku 1995. Tato řada byla uvedena se třemi modely S7-200, S7-300 a S7-400. Tyto PLC jsou však poslední dobou nahrazovány novějšími modely PLC řady S7-1200 a S7-1500. [1]

1.1. Parametry PLC S7-1200

Řada SIMATIC S7-1200 byla poprvé představena v roce 2012 společně s řadou S7-1500. Tato řada je primárně určená pro méně až středně složité projekty. Stejně jako ostatní PLC řady S7 je tato řada nastavována a programována pomocí softwaru TIA portal. [1]

Tato řada je dostupná ve třech verzích dle základní řídící jednotky, z nichž první je klasické CPU pro běžný provoz. Druhou verzí jsou takzvané failsafe CPU, které společně se speciálními moduly zajišťují zvýšenou bezpečnost provozu. Poslední verzí je SIPLUS CPU používaná pro provozy v extrémním prostředí a jsou odolnější například před vysokými teplotami, mechanickým namáháním či před žíravými látkami. [1,3]

I nejzákladnější model S7-1200 je osazen několika digitálními vstupy a výstupy a pár vstupy analogovými, složitější modely jsou osazeny i analogovými výstupy. Digitální I/O pracují nejčastěji s hodnotami 5 V nebo 24 V, zatímco analogové vstupy nejčastěji pracují v rozmezí 4-

20 mA nebo 0-10 V. Pro zvýšení počtu I/O nebo pro speciální případy je možné základní CPU rozšířit až o 8 signálových modulů. [1]

Pro zajištění komunikace jsou všechny modely S7-1200 vybaveny jedním nebo dvěma zabudovanými porty PROFINET. Některé modely mají v sobě zabudovanou desku CB 1241 pro komunikaci dle standartu RS-485. Dále mohou být k základnímu CPU připojeny až 3 komunikační modely pro komunikaci nejčastěji pomocí protokolů PROFIBUS, AS-Interface, ale i jiných průmyslových sběrnic či bezdrátového připojení. Jednou z novinek, co řada S7-1200 přinesla je integrovaný webový server pro možnost internetové komunikace uživatele s PLC. [1]

1.2. TIA Portal

TIA Portal neboli Totally Integrated Automation Portal je inženýrský program vyvinutý firmou Siemens používaný při práci se systémem SIMATIC od jednoduchého programování PLC po řízení a diagnostiku celého projektu. Program je vlastně kombinací několika programů, z nichž nejdůležitějších je STEP7, který se používá pro samotné nastavování a programování řídících kontrolerů SIMATIC a softwarových kontrolerů WinAC. STEP 7 je možné získat ve 2 variantách. STEP 7 Basic je používán pouze pro ovládání modelů S7-1200, zatímco s verzí Professional je možné řídit i ostatní systémy včetně S7-1500 a WinAC. [1,4]

Dalším důležitým programem integrovaným do prostředí TIA Portal je SIMATIC WinCC používaným pro ovládání či sledování stavů projektů z různých HMI systémů včetně operátorských panelů, průmyslových počítačů nebo klasických počítačů se speciálním softwarem. [4]

Samotný TIA Portal nabízí uživatelsky přívětivé prostředí se snadnou orientací uvnitř projektů. Připojení jednotlivých prvků do projektu je řešeno vytvořením nového prvku s pomocí vnitřní knihovny a jeho následným spárováním s prvkem reálným. Do takto vytvořených prvků lze následně nahrávat uživatelské programy. TIA Portál taktéž umožnuje rychlé nastavení komunikace mezi jednotlivými prvky v systému. [1]

Samotný program, nebo jeho části je možné vytvořit ze 4 druhů stavebních bloků. Nejzákladnějším blokem je blok organizační (OB). OB existuje několik druhů a liší se především tak, za jakých okolností je samotný OB spuštěn. Nejzákladnějšími jsou cyklické OB jako blok Main (OB1), ve kterých je vnitřní program vykonáván stále dokola. Mezi další významné OB patří blok Startup (OB100), který je vykonán po zapnutí či zresetování PLC, bloky přerušení nebo bloky reagující na chybové stavy. Dalšími dvěma stavebními bloky jsou funkce (FC) a funkční bloky (FB), do kterých jsou zapisovány, části programu. FB se od FC odlišuje tím že hodnoty, se kterými pracuje, ukládá do vlastní paměti (DB) a tyto hodnoty jsou díky tomu dostupné pro ostatní funkce v programu i po zpracování daného bloku. Posledním typem je globální data blok (DB), který je používán pro ukládání dat z programu. Lze z něj číst či do něj zapisovat hodnoty ze všech částí programu a při správném nastavení i z ostatních PLC. [4]

Samotné programování je prováděno pomocí specializovaných programovacích jazyků, které jsou definovány pomocí normy IEC 1131-3. Mezi textové jazyky normy IEC1131-3 patří jazyk

mnemokódů, který je svou strukturou podobný asembleru a jazyk strukturovaného textu, který připomíná spíše svou strukturou jazyk C. Mezi grafické jazyky patří jazyk kontaktních schémat, který je založen na schématech s relé a kontaktními prvky, jazyk logických schémat, ve kterém je program psán propojováním logických prvků a jazyk sekvenčního programování ve kterém je vytvářen sekvenční diagram. Kromě základních logických prvků TIA Portal obsahuje seznam instrukcí, ze kterého lze do programu přetahovat složitější funkce od čítačů po zajištění komunikace s ostatními prvky v síti. [2]

1.3. Webový server

Modely SIMATIC řady S-1200 a S7-1500 mají v sobě z výroby integrovaný webový server. Tento webový server může sloužit ke vzdálenému dohledu či řízení procesů v PLC systému. Takovéto dynamické webové stránky jsou tvořeny pomocí skriptovacího jazyka HTML, často doplněné o kaskádové styly. Dynamická část stránek je často vytvořena pomocí programovacích jazyků Javascript nebo PHP. Pro komunikaci webserveru s CPU PLC jsou používány příkazy AWP, které inicializují spojení mezi tagy vytvořenými v PLC s proměnnými vytvořenými na serveru webových stránek. [5]

Nahrání stránek do PLC je prováděno v TIA Portálu vybráním složky se soubory webových stránek a nastavením defaultní HTML stránky. Pro uložení dat stránek je v PLC vytvořen data blok DB333. Kromě samotných dat stránek obsahuje blok DB333 i dodatečné informace např. o stavu komunikace s CPU či informace o chybách. Pokud se do tohoto jednoho bloku všechny data webových stránek nevejdou, jsou automaticky vytvořeny další data bloky zvané fragmenty DB, do kterých jsou zbylá data zapsána. Načtení webových stránek k paměti PLC je provedeno pomocí instrukce WWW (SFC99), která je vložená do bloku Main (OB1). [5]

1.3.1. HTML

Pro tvorbu webových stránek se využívá především značkovacího jazyka HTML. HTML bylo světu poprvé představeno v roce 1991 a od té doby prošlo značným vývojem. Momentálně nejnovější verzí HTML je HTML5, které bylo představeno v roce 2014. Dokument čistého HTML je tvořen obyčejným textem, značkami (tagy) a jejich vlastnostmi (atributy). Značky slouží k tvorbě elementů stránky a v daném dokumentu určují jejich význam, který je často dále specifikován jednotlivými atributy. [6]

1.3.2. CSS

CSS neboli kaskádové styly je jazyk používaný pro popis způsobu vizualizace jednotlivých elementů umístěných na webových stránkách, který byl poprvé představen v roce 1996. Tyto styly mohou být zapsány jak v samotném HTML dokumentu, ve kterém se používají, tak i v externím souboru s příponou .css. Kaskádové styly dokáží změnit vizuální podobu jednotlivých elementů v dokumentu buď změnou všech elementů v dokumentu, které byly vytvořeny stejnou HTML značkou anebo změnou pouze HTML elementů s přiřazenou specifickou třídou a umožnují provádět vizuální změny v dokumentu od změny barvy pozadí až po komplikované animace. [6]

1.3.3. JavaScript

Javascript je programovací jazyk vytvořený v roce 1997, který výrazně usnadňuje tvorbu webových aplikací. JS je jazyk, který je vykonáván na straně klienta ve webovém prohlížeči a díky této vlastnosti JS nezatěžuje webový server. Jeho nevýhodou je však právě kvůli provádění scriptů ve webovém prohlížeči jeho horší zabezpečení. Další nevýhodou JS je závislost na aktuálnosti webového prohlížeče, kdy starší verze prohlížečů nemusejí podporovat některé funkce, nebo u novějších verzí prohlížečů mohou být některé funkce nahrazeny funkcemi novými. [6]

1.3.3.1. jQuery

jQuery je JS knihovna, která se stará o zjednodušení některých funkcí v Javascriptu. Knihovna byla poprvé představena v roce 2006. Tato knihovna je využívána hlavně pro asynchronní komunikaci s webovým serverem pomocí AJAX, dále pak pro manipulaci elementů webových stránek nebo pro pracování s událostmi. [6]

1.3.3.2. DOM (Document Object Model)

DOM je webové programovací rozhraní, které vnímá HTML dokument jako strom, jehož každá větev reprezentuje element nebo objekt v daném dokumentu. Díky tomuto stromovému uspořádání DOM umožnuje snadné hledání nebo přidávání HTML elementů na webovou stránku a umožnuje upravování jejich vlastností či obsahu. Jednotlivé objekty mohou mít díky DOM připojeny spouštěče událostí, které mohou po zapnutí volat JS funkce. DOM je již implementován ve všech běžných webových prohlížečích. [6]

2. Přehled SCADA/HMI systémů

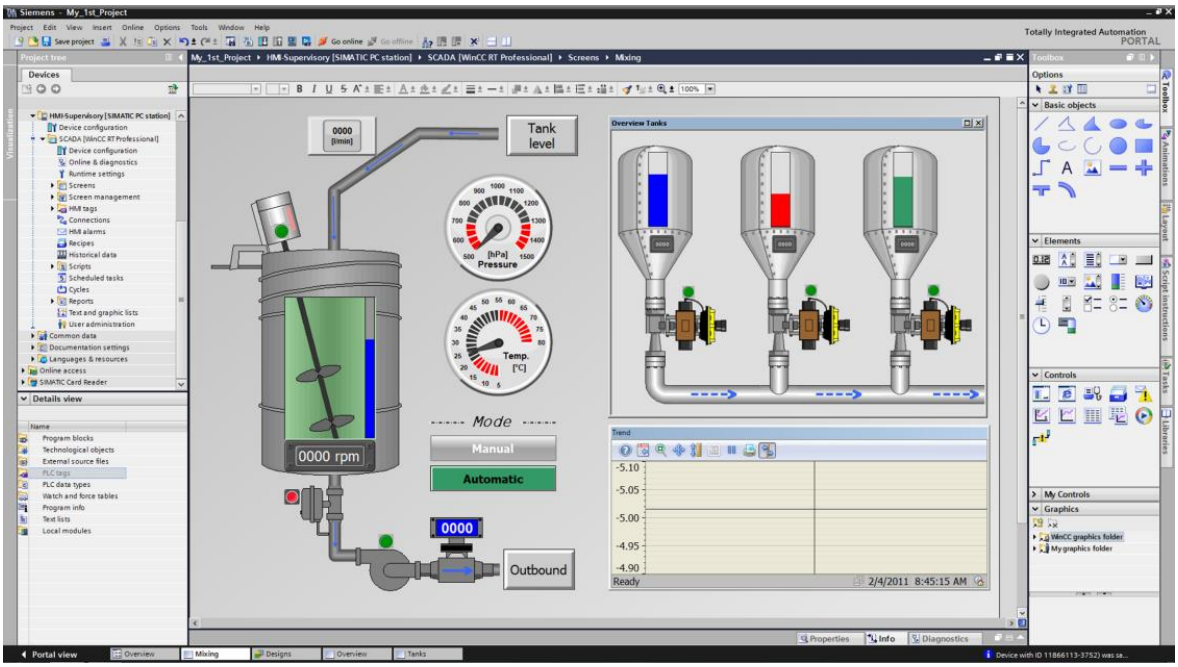
SCADA, z anglického Supervisory Control And Data Acquisition, je systém určený pro dohled, řízení a sběr dat z technologických procesů. SCADA ve valné většině případů nezastává funkci řídícího systému, ale stará se o dohled nad přiřazeným projektem a jeho případným řízením. Termín SCADA je proto ve většině případů označení softwarových systémů umístěných nad skutečnými řídícími systémy, jako jsou nejčastěji PLC nebo RTU. [7]

Moderní SCADA systémy jsou nejčastěji založené na principu webových aplikací, které umožnují jednoduchou možnost vzdáleného přístupu k systému s pomocí internetového prohlížeče. Tyto systémy dokáží komunikovat se svým okolím pomocí velkého množství průmyslových sítí, v poslední době však dochází na přechod k využívání ethernetových protokolů jako S-BUS či Modbus TCP/IP. S postupným vývojem SCADA systémy začali pronikat i mimo čistě průmyslové oblasti využití, jako jsou například chytré domácnosti. [7]

2.1. WinCC

WinCC je SCADA program vyvinutý firmou Siemens v roce 1996, který byl následně v roce 2010 integrován do TIA Portálu. Program je určen především k vytváření HMI pro systémy řady SIMATIC. [8]

HMI vytvořené pomocí WinCC lze nahrát jak do ovládacích panelů od Siemensu, tak lze i využít WinCC Runtime pro běh HMI na počítačích připojených k síti s PLC. Pro vytvoření nového HMI je nejdříve nutné do projektu v TIA Portálu přidat nový hardware panelu nebo PC, který se do projektu vkládá stejně jako PLC. Po vložení je zároveň nutné spojit nový prvek do sítě s existujícím PLC pro umožnění přenosu dat. Po prvotním nastavení je možné na obrazovku v grafickém editoru vkládat prvky z Toolboxu. Tyto prvky lze po vložení do projektu jednoduše propojit s tagy umístěnými v PLC v daném projektu. [9]



Obr. 1: Vývojové prostředí WinCC [8]

Pro zobrazení HMI vytvořeného v programu WinCC je často používán operátorské panely SIMATIC HMI Basic a SIMATIC HMI Comfort. Pro jejich využití je nutné vlastnit základní licenci WinCC Basic či WinCC Comfort pro ovládání všech typů uživatelských panelů. Druhou možností, jak zobrazit HMI je pomocí PC, pro jehož použití je nutné vlastnit licence WinCC Advanced nebo WinCC Profesional. Poměrně novou možností je využití licence WinCC Unified, která umožnuje vytvoření HMI přístupného přes webový prohlížeč. [8]

Mezi největší výhody použití WinCC patří uživatelská přívětivost při vytváření nového HMI a jednoduché zajištění komunikace s ostatními prvky v síti. Mezi další výhody patří to, že samotná firma Siemens poskytuje všechny potřebné materiály pro práci s programem, spravuje vlastní fórum pro dotazy uživatelů a zajišťuje i placené kvalifikační kurzy pro práci s jejich programy. [8]

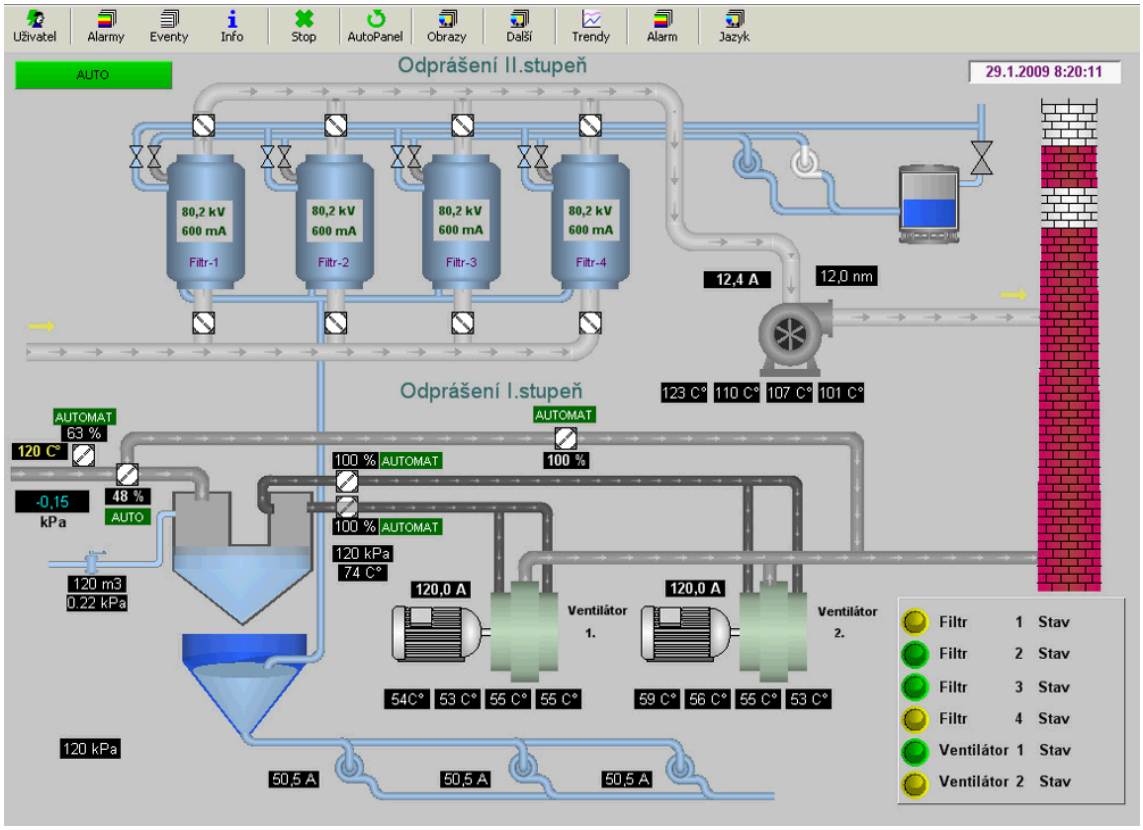
2.2. PROMOTIC

PROMOTIC je SCADA systém vyvinutý českou společností MICROSYS v roce 1991. V tomto systému jsou zabudovány všechny důležité části pro tvorbu vizualizačních a řídících systémů. Momentálně nejnovější verzí tohoto systému je PROMOTIC 9 z roku 2018. [7]

Program PROMOTIC lze rozložit na vývojové a runtime prostředí. Vývojové prostředí se skládá z několika editorů projektu. Editor Pma objektů se stará o vnitřní strukturu objektů v programu, nastavování jednotlivých objektů a definování uživatelských algoritmů. V druhém editoru PmaPanel se vytváří grafické prostředí systému. Editor obsahuje obsáhlou knihovnu prvků, které lze vložit jednoduchým přetažením do grafického prostředí. Knihovnu lze dále rozšiřovat pomocí vlastní grafiky. Samotné prvky je možné konfigurovat dynamicky pomocí propojení s Pma prvky projektu. Dalším dostupným editorem je editor skriptů. V systému PROMOTIC lze psát skripty pomocí Javascriptu nebo VBScriptu, které umožnují tvorbu uživatelských algoritmů nebo editaci

již předvytvořených událostí. Dále jsou v programu integrovány další editory, jako jsou správci konfiguračních souborů anebo správce záloh aplikací. [7]

Systém PROMOTIC díky své uživatelské přívětivosti vhodný i pro začátečníky. Mezi jednoznačné výhody patří české vývojové prostředí a přehledná dokumentace s videotutoriály usnadňující začátky používání systému. K systému je možné připojit velké množství komunikačních ovladačů jak pro komunikaci s jednotlivými protokoly, tak i přímo s PLC automaty. [7]



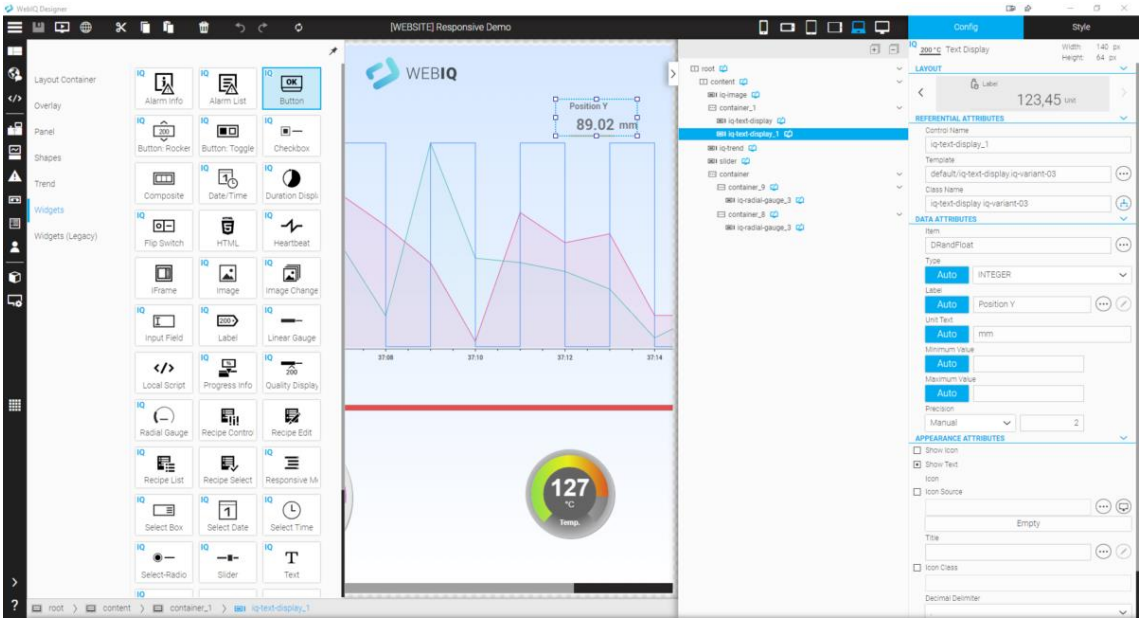
Obr. 2: Ukázka HMI vytvořeného v programu PROMOTIC [7]

PROMOTIC je komerční produkt, k jehož použití je zapotřebí zakoupení licence jak pro samotné vývojové a runtime prostředí, tak i pro jednotlivé komunikační rozhraní. Tento systém je však možné využívat zdarma ve své freeware licenci. Tato licence je však omezená ve svém vývojovém prostředí na maximálně 100 proměnných, v runtime prostředí je nutné mít nanejvýše 30 proměnných a 10 objektů grafiky pro kontinuální běh programu. Runtime prostředí lze ve freeware licenci spustit i s počtem proměnných do 100, program se však po 1 hodině přeruší. Freeware verze dále obsahuje všechny dostupné komunikační moduly a web server s možností připojení až 2 klientů najednou.[7]

2.3. WebIQ

WebIQ je HMI/SCADA systém využívající pouze webové prvky HTML5, CSS a Javascriptu, který byl poprvé představen v roce 2012. Od této doby je systém pravidelně aktualizován a nejnovější verze 2.13.0 byla představena na konci roku 2022. [10]

Jako většina webových HMI systémů je WebIQ tvořen vývojovým prostředím WebIQ Designer a runtime prostředím WebIQ Server. Grafické vývojové prostředí umožnuje vkládat do HMI rozsáhlé množství předpřipravených prvků, ať už se jedná o widgety nebo o oddíly rozdělující prvky na stránkách do specifických bloků, které usnadňují rozložení webové stránky pro různá rozlišení displejů včetně mobilních telefonů. Kvůli rozdílnému rozlišení platforem grafické vývojové prostředí umožnuje snadné přepínání rozlišení stránky pro kontrolu vzhledu HMI. Mezi widgety, které lze vložit do HMI, patří většina běžných prvků jako tlačítka, slidery či dynamicky měněný text společně s bloky pro vládání kusů skriptů, oznamování chyb či virtuální klávesnicí. Pro přiřazení widgetů k reálným systémům je použit I/O Handler. [10]



Obr. 3: Vývojové prostředí WebIQ [10]

Mezi největší výhody WebIQ patří možnost libovolně vkládat předpřipravené prvky do svého prostředí a tyto prvky následně modifikovat jednoduchou úpravou HTML a CSS prvků, ale i možnost vkládáních prvků vlastních. Mezi další výhody WebIQ patří rozsáhlá dokumentace a návody pro začínající uživatele. Systém je proto vhodný jak pro začátečníky, kteří nemají zkušenosti s programováním, tak i pro zkušené uživatele, kterým umožní téměř libovolné přizpůsobení HMI pro potřeby jejich projektů. [10]

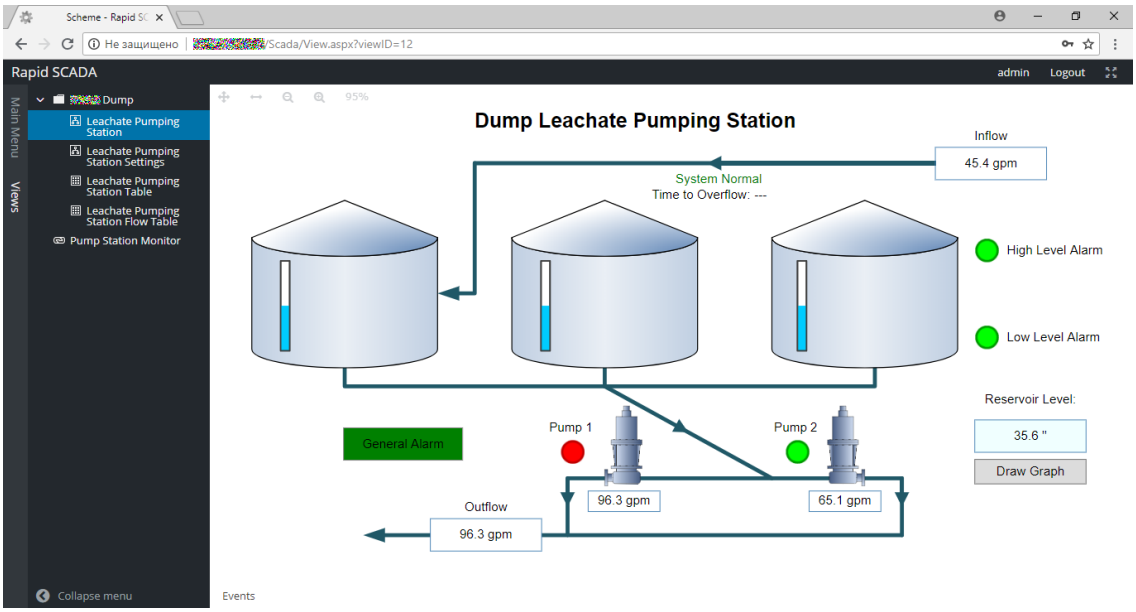
WebIQ Designer je komerční produkt vyžadující měsíční předplatné s možností získání 30denní trial verze programu. Pro provoz je nutné i zakoupení jednorázové licence programu WebIQ Server, který slouží jako webový server pro vytvořený projekt. [10]

2.4. Rapid SCADA

Rapid SCADA je open-source systém vyvinutý společností Rapid Software LLC v roce 2016. [11]

Program systému Rapid SCADA je velmi prostý, vše potřebné k vytvoření MHI je obsaženo ve stromové struktuře hlavní nabídky programu. V té můžeme najít položku databáze, ve které jsou obsaženy vlastnosti objektů projektu, komunikační sběrnice, I/O kanály nebo seznam uživatelů.

Další hlavní větví je interface, které dokáže zobrazit tabulky ze svých databází, textové a XML soubory a základní grafické rozhraní s pár jednoduchými prvky. Poslední větví ve stromovém seznamu aplikace je položka instancí obsahující konfigurace serveru, komunikačních sběrnic a webstation umožňující řízení projektu přes webový prohlížeč. V samotném programu jsou už integrovány protokoly pro běžné komunikační sběrnice. [11]



Obr. 4: Ukázka HMI vytvořeného v programu Rapid SCADA [11]

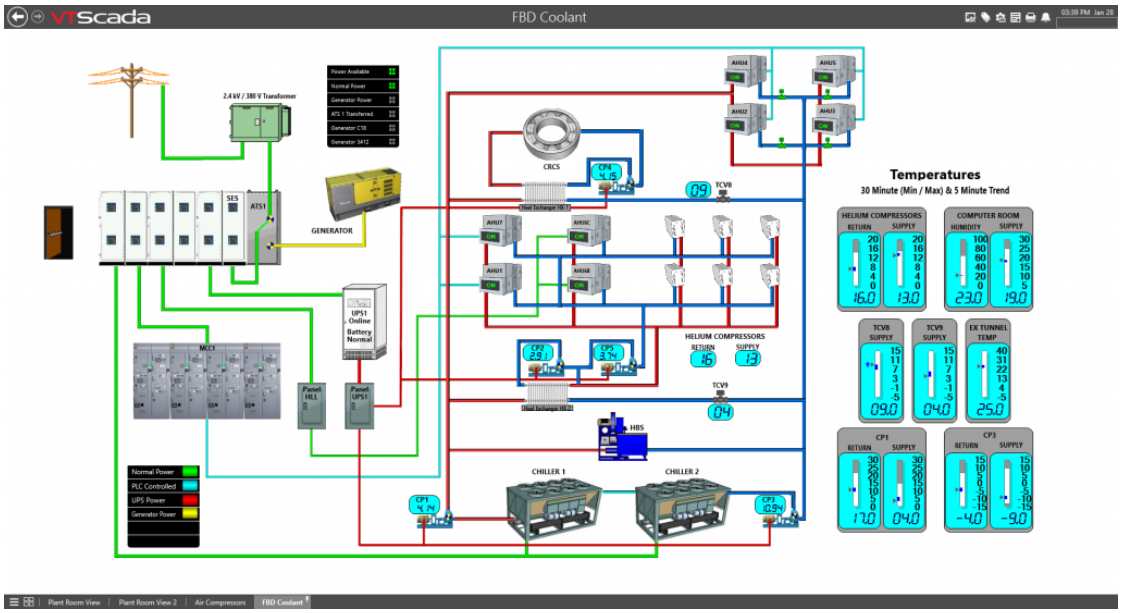
Výhodou Rapid SCADA je její open-source licence, která umožnila komunitě kolem ní vytvářet nejrůznější moduly k základnímu programu. Tudíž i když je samotný program poměrně holý, obsahující pouze součásti nezbytné pro běh SCADA systému, lze program rozšířit specifickými moduly určenými pro potřeby uživatele. [11]

Rapid SCADA je freeware program, stáhnutelný na stránkách výrobce zdarma. Autoři zároveň vytvořili několik vlastních modulů rozšiřujících program, které jsou ale již placené. [11]

2.5. VTScadaLIGHT

VTScadaLIGHT je freeware verzí softwaru VTScada od firmy Trihedral. [11]

Vývojové prostředí VTScadaLIGHT umožnuje velmi přehledně konfigurovat komunikaci s okolím s pomocí Tag browseru. Samotný program již má v sobě integrované ovladače pro velké množství komunikačních sběrnic a zjednodušuje práci se samotnými tagy, včetně jejich škálování, varování po překročení limitů nebo přiřazení fyzikální jednotky k její hodnotě. Nastavené tagy lze následně vložit do grafického rozhraní Idea Studio pro tvorbu HMI pomocí klasických widgetů. Program v sobě má zabudovaný vlastní server, který umožnuje sledování projektu vzdáleně pomocí webového prohlížeče s pomocí VTScada Anywhere Klienta založeného na jazyce Javascript. [12]



Obr. 5: Ukázka HMI vytvořeného v programu VTScada [12]

Mezi výhody VTScada patří jeho uživatelská přívětivost v kombinaci s vlastní knihovnou zabudovanou v základním programu. Samotná komunikace s vnějším klientem je zabezpečená heslem a vlastním šifrováním. [12]

Jakožto freeware verze komerčního programu je VTScadaLIGHT i pár omezení, mezi něž patří omezení na maximálně 50 tagů a nefunkčnost notifikací a alarmů přes email nebo SMS. [12]

2.6. HMI z integrovaného webového serveru

Další z možností, jak řídit procesy v projektu, je použití dynamických webových stránek na webserveru integrovaném v PLC. Statické webové stránky jsou klasicky psané ve skriptovacím jazyce HTML, často v kombinaci s kaskádovými styly CSS, které upravují vzhled jednotlivých prvků na webových stránkách. Styly CSS mohou být obsaženy v samotném html dokumentu, nebo jsou zapsány v externím dokumentu, na který se html soubor odkazuje. V dnešní době se však používají modernější verze skriptovacích jazyků jako HTML5 které buď rozšiřují původní jazyky o mnohé funkce nebo některé vlastnosti výrazně zjednodušují. Pro propojení tagů v PLC systému s proměnnými na stránkách se používají příkazy AWP. [5,6]

Pro vytvoření dynamiky stránky je nutné použití programovacích jazyků. Pro vývoj webu jsou nejčastěji používány jazyky Javascript a PHP. Jazyk PHP je programovací jazyk vytvořený přímo pro ovládání webových stránek. Jednou z jeho nevýhod pro použití v integrovaném webové serveru je to, že skripty jsou prováděny na straně serveru, což může způsobit zpomalení samotného PLC. Pro programování stránek se proto používá převážně objektově orientovaný jazyk Javascript. Skripty napsané v jazyce Javascript se provádí až ve webovém prohlížeči, tudíž nijak nezatěžují běh PLC. Na druhou stranu zde existuje riziko, že některé novější funkce nebo knihovny Javascriptu nemusí být podporovány staršími internetovými prohlížeči. [5]

Největší výhodou zároveň a nevýhodou oproti SCADA systémů je to, že si tvůrce webových stránek musí napsat všechny části webu sám. Tvůrce se tudíž musí umět orientovat v oblasti programování a tvorbě webu, na druhou stranu mu tento přístup umožnuje vytvoření systému na míru pro konkrétní problém. Pro vytvoření HTML, CSS a Javascript souborů je zapotřebí pouze jakýkoliv textový editor, ale existují i webové editory jako Adobe XD, které tvorbu stránek mohou velice zjednodušit. [13]

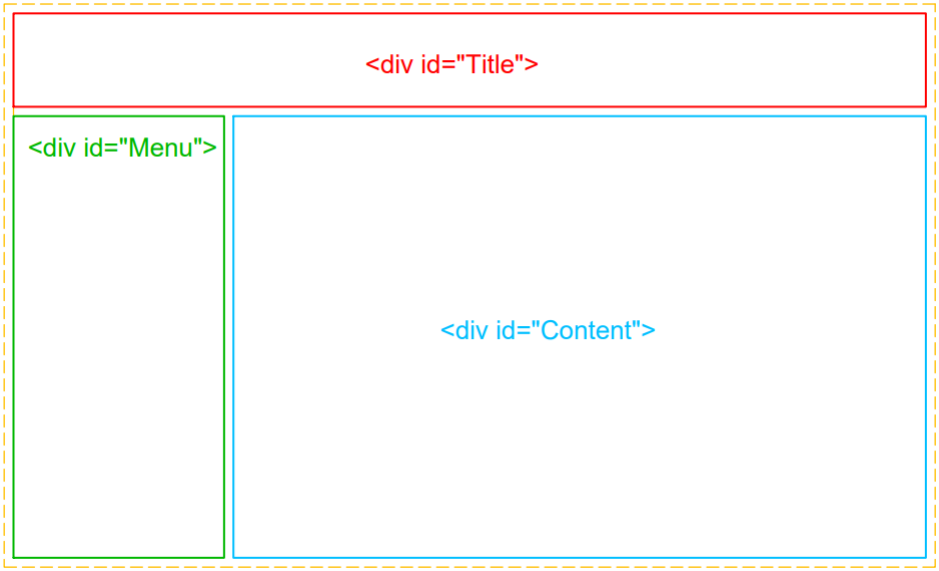
3. Tvorba webové aplikace

Systém, který je vyvíjený jako součást této diplomové práce, lze rozdělit na dvě samostatně stojící části. První částí této práce bude tvorba dynamického webového rozhraní pro čtení a zápis dat z PLC a srozumitelné zobrazení či ovládání těchto dat koncovým uživatelem. Druhou částí bude vytvoření knihovny funkcí pro program TIA Portál, která bude složit ke zjednodušení tvorby komunikace s ostatními PLC v dané síti a k přeposílání dat mezi PLC a webovým serverem.

3.1. Tvorba základního webu

Jako první věc jsem si vytvořil soubor index.html, který bude kostrou tvořené webové stránky. Pro zjednodušení vytváření a úpravu souborů webových stránek jsem použil program Visual Studio Code [14]. Jedná se o zdarma stažitelný editor zdrojového kódu od společnosti Microsoft. Jako u podobných vývojových prostředí VS Code obsahuje správu projektů pro rychlé nalezení souvisejících souborů či přidání souborů nových, automatické doplňování částí kódu dle vybraného programovacího či skriptovacího jazyka nebo seznam nalezených chyb syntaxe programu. VS Code taktéž nabízí velké množství stáhnutelných rozšíření. Pro účely této práce jsem využíval rozšíření Live Server [15], které na počítači vytvoří lokální server, na kterém mohou být vyzkoušeny ty části stránek, které jsou při spuštění ze správce souborů zablokovány.

V souboru index.html jsem vytvořil hlavičku dokumentu (head) a tělo dokumentu (body). Do hlavičky jsem umístil všechny skripty napsané v Javascriptu tak, že jsem načetl skripty z ostatních souborů v projektu. Dále jsem v hlavičce načetl CSS styly použité v projektu a nastavil jsem titulek stránek na "Řízení inteligentního domu". S pomocí elementu div jsem do těla stránky umístil 2 bloky s názvy Title a Main a následně jsem do bloku Main stejným způsobem umístil bloky s názvy Menu a Content. Blok Title slouží jako záhlaví stránky, blok Menu k umístění navigačních prvků a blok Content k zobrazení samotného obsahu webových stránek.



Obr. 6: Základní rozložení webových stránek

3.2. Tvorba obsahu stránek

Hlavní obsah webových stránek bude tvořen z předpřipravených bloků. Každý z těchto bloků je určený k ovládání jednotlivých zařízení připojených v soustavě PLC. Největší výhodou předvytvořených bloků je, že při připojení více stejných či podobných zařízení není nutné pro každé z nich vytvářet znovu části kódů pro jejich zobrazení, ale při načítání stránky stačí zavolat funkci, která se postará o tvorbu daného bloku s námi danými parametry.

Pro tvorbu těchto bloků jsem v hlavním adresáři webových stránek vytvořil soubor seznam_zarizeni.js, který bude obsahovat jak funkce pro samotnou tvorbu bloku na webových stránkách, tak i seznam jednotlivých zařízeních připojených k PLC, podle kterých se tyto bloky budou tvořit. Vytvoření nového bloku je provedeno otevřením tohoto souboru v textovém editoru a napsáním

```
function Novy_blok(Jmeno,Room,DB){}
```

do jeho obsahu. Tímto se vytvořil nový blok, který může být následně umístěn na webové stránky. Funkce kromě třech základních vstupních argumentů může obsahovat argumenty:

- Out_1 až Out_15 pro definici výstupů funkce do PLC
- In_1 až In_15 pro definici vstupů z PLC
- Konst_1 až Konst_15 pro námi zadané konstantní hodnoty nebo texty pro jednotlivá zařízení

Poté, co jsme vytvořili novou funkci pro tvorbu webového bloku pro dané zařízení a definovali jsme vstupy a výstupy, které budeme používat pro komunikaci s PLC, je nutné vytvořit obsah daného bloku, který se bude zobrazovat na webových stránkách. Ten můžeme vytvořit pomocí jednotlivých předpřipravených funkcí využívajících kombinace Javascriptu s DOM (Document Object Model). Pro přidání jednotlivých komponent stačí přidat do obsahu takto vytvořené funkce jednotlivé funkce tvořící dané komponenty stránek. Samotné funkce komponentů stránek jsem umístil do souboru s názvem komponenty.js ve složce Scripty.

Pro tento projekt jsem vytvořil několik základních funkcí, které přidávají HTML prvky na webové stránky a díky tomu umožnují jednodušší tvorbu bloků pro nová zařízení. Před spuštěním těchto funkcí se při načtení stránky automaticky vytvoří pomocí DOM funkce *.createElement("div")* objekt s názvem skupina_prvku, který se v dokumentu chová jako HTML element . Do tohoto objektu se budou umisťovat všechny námi vytvořené prvky určené pro danou stránku. Seznam jednotlivých funkcí a princip jejich fungování je popsán níže.

3.2.1. Přidání textu a textových polí

3.2.1.1. Obyčejný text

Pro přidání neměnného textu na webovou stránku byla vytvořena funkce n_text().

```
n_text(text,styl_vl,styl)
```

Funkce vytvoří pomocí příkazu *createElement("span")* objekt s názvem temp, který se na webové stránce bude chovat jako element . Následně je pomocí příkazu *setAtribute()* možno nastavení stylu tohoto elementu pomocí CSS. K objektu temp je následně pomocí příkazu *.className* přidán atribut stylu s názvem *div_inline*, který zajišťuje, že při více textech za sebou se zobrazí na stejném řádku. Pomocí stejného příkazu je následně možné vložit název již předvytvořeného stylu, který se bude nacházet v jednom z CSS souborů tohoto projektu. Dále je příkazem *.innerHTML* vložen text do elementu <a> a příkazem .append() je prvek vložen do objektu skupina_prvku.

3.2.1.2. Text závislý na vstupních hodnotách

Pro přidání textu na webovou stránku, který se bude měnit podle vstupních argumentů byla vytvořena funkce n_text_change().Tato funkce po načtení tohoto prvku na stránce zobrazí v prvku takový text, jehož přidružená argument val bude mít stejnou hodnotu jako argument vstup. Pro příklad, pokud bude hodnota argumentu val2 stejná jako hodnota argumentu vstup, v prvku se zobrazí text zapsaný v argumentu text2.

```
n_text_change(vstup,val1,txt1,val2,txt2,val3,txt3,val4,txt4)
```

Funkce pomocí příkazu *createElement("span")* vytvoří objekt temp jako u klasického textu. Pomocí příkazu *.className* k prvku přidáme třídy *txt_switch* a *div_inline*. Třída div_inline slouží k nastavení zobrazení prvku <a> jako inline, což umožnuje přidání textu na stejný řádek, jako byl text předchozí a třída txt_switch slouží jako identifikátor pro pozdější dynamickou změnu samotného textu. Dále je pomocí funkce *.setAttribute()* přidány atributy typu *data-**, které slouží k zapsání dat ze scriptu jako atributy HTML elementu, které mohou být čteny a využívány při aktualizaci daného prvku. Poté, co se vytvoří atributy *data-value*, *data-val*(i) a *data_text*(i) a jejich hodnota je nastavena podle vstupních argumentů funkce *n_text_change()*, je připravený objekt temp vložen do objektu skupina_prvku pomocí příkazu *.append()*.

3.2.1.3. Text aktuální hodnoty PLC tagu

Pro přidání textu který bude obsahovat hodnotu přiřazenému PLC tagu byla vytvořena funkce n_text_tag().

```
n_text_tag(vstup)
```

Tato funkce slouží k zapsání výstupní hodnoty z PLC. Funkce vytvoří pomocí DOM funkcí objekt HTML elementu s názvem temp. Následně jsou danému objektu nastavena třída *div_inline* a atribut data-value o hodnotě vstupního argumentu vstup. Objekt temp je následně připojen do objektu skupina_prvku.

3.2.1.4. Text vstupní hodnoty v procentech

Pro zobrazení hodnoty vstupního argumentu vstup1 v procentech, kdy hodnota vstup2 je určená jako 100% vstupní hodnoty proměnné vstup1, byla vytvořena funkce n_text_pr().

```
n_text_pr(vstup1,vstup2)
```

Funkce vytvoří objekt HTML elementu s názvem temp. Následně objektu temp přidá atributy *data-value* s hodnotou argumentu vstup1 a *data-value2* s hodnotou argumentu vstup2. Dále jsou k objektu temp přidány třídy *div_inline* a *percent_input*. Objekt je poté vložen do objektu skupina_prvku.

3.2.2. Prvky pro správné rozložení ostatních prvků

3.2.2.1. Konec řádku

Pro zalomení textu do dalšího řádku byla vytvořena funkce n_br().

```
n_br()
```

Funkce pomocí DOM funkce *createElement("div")* vytvoří objekt HTML elementu s názvem temp. Tomuto prvku je přidána pomocí příkazu *.className* třída *div_br*. Tento objet je následně vložen do objektu skupina_prvku. Tato funkce funguje tak, že se třídou div_fill nastaví vlastnosti vkládaného elementu pomocí CSS na prvek s pružnou délkou. Toto způsobí, že se tento automaticky umístí pod prvek předchozí, má nulovou výšku a 100% šířku v daném prostoru. Jakékoliv elementy, které jsou vytvořeny po tomto prvku se umístí až pod tímto oddílem.

3.2.2.2. Konec oddílu

Pro zalomení obsahu v bloku na webové stránce byla vytvořena funkce n_bl().

```
n_bl()
```

Na rozdíl od funkce předchozí, která zalamovala pouze text, tato funkce dovolí vytvoření nových prvků až pod nejnižším bodem prvků předchozích. Samotná funkce vytvoří objekt HTML elementu s názvem temp a přidělí mu třídu div_fill. Elementu je nastavena CSS vlastnost *clear* na hodnotu *both* která způsobí, že je prvek umístěn až na takové místo, kam se se svou šířkou vejde. Jelikož je jeho šířka nastavená na 100% šířky oddílu, ve kterém jsou umístěny těmito funkcemi tvořené prvky, je prvek umístěn pod všechny již vytvořené prvky a prvky, které budou vytvořeny poté, se umístí pod tento element.

3.2.2.3. Ukončení bloku

Pro vizuální ukončení celého bloku pro jedno specifické zařízení byla vytvořena funkce n_konec().

```
n_konec()
```

Tato funkce automaticky vytvoří objekt elementu a přidělí mu třídu s názvem *konec_prvku*. Tento prvek je svou funkcí velmi podobný prvku tvořeným funkcí n_bl() a jediným rozdílem v těchto prvcích je jejich vizualizace. Zatímco předchozí oddíl měl nulovou výšku a byl proto neviditelný, prvek vytvořený touto funkcí má výšku 5px a nachází se v něm přerušovaná tlustá čára, která vizuálně rozděluje rozdílné bloky od sebe.

3.2.3. Přidání číselných vstupů

3.2.3.1. Pole pro zadání čísla

Pro přidání pole číselného vstupu na webovou stránku byla vytvořena funkce n_numin().

```
n_numin(vstup,min,max)
```

Tato funkce na vytvoří pomocí příkazu DOM objekt elementu <input> s názvem *temp*. Tomuto objektu byl přiřazen atribut *type* s hodnotou *number*, pomocí kterého je do tohoto vstupního pole možné zapisovat pouze čísla. Dále funkce objektu přiřadí atributy min a max s hodnotami ze vstupních argumentů, které slouží k určení rozsahu čísel, které mohou být tomuto poli zadány pomocí postranního posuvníku, atributy *data-input* a *data-value* s hodnotou vstupního argumentu vstup, které slouží k identifikaci tohoto prvku při čtení a posílání dat do PLC a prázdný atribut value. Objektu temp budou funkcí přiřazeny třídy num_div a text_value. Třída num_div se stará o umístění prvku do textu a upravuje jeho výšku podle velikosti základního fontu webové stránky. Poté je příkazem

```
.addEventListener("focus",function(){submit_value(vstup)})
```

při načtení stránky zapnuta funkce submit_value(), která se stará o posílání dat do PLC při změně hodnoty tohoto pole. Takto vytvořený objekt je následně vložen do objektu skupina_prvku.

3.2.3.2. Pole pro zadání času

Pro přidání pole na webovou stránku, ve kterém je možné nastavit čas, byla vytvořena funkce n_timein()

```
n_timein(vstup)
```

Tato funkce vytvoří pole stejným způsobem jako funkce n_numin(), jediným rozdílem je přenastavení hodnoty atributu *type* na hodnotu *time*.

3.2.4. Přidání tlačítka

Pro přidání přepínače na webovou stránku byla vytvořena funkce n_switch().

```
n_switch(input,text)
```

Funkce vytvoří pomocí DOM příkazů celkem 6 objektů. Objekty temp0, temp1 a temp5 s vlastnostmi elementu , temp2 jako objekt elementu <label>, temp3 jako objekt elementu <input> a temp4 jako objekt elementu . Objekt temp0 v tomto případě slouží jako blokový oddíl, ve kterém bude na levé straně umístěn element z objektu temp5 obsahující vlastní text a na straně levé element objektu temp1, který v sobě ukrývá přepínač. Samotný přepínač je tvořen

objektem temp2, do kterého jsou vloženy objekty elementů <input> s názvem temp3 a s názvem temp4. Tuto stromovou strukturu elementů následně vložíme do objektu skupina_prvku.

Po definování stromové struktury tvořených elementů přiřadíme jednotlivým elementům jejich atributy. Objektu temp0 přiřadíme třídu *switch_body*, která se postará o blokové zobrazení celé soustavy prvků, nastaví jeho rozměry a správné umístění v těle stánky. Do objektu temp5 vložíme pomocí příkazu *.innerHTML* text ze vstupního argumentu a nastavíme mu třídu *switch_text* Objektu temp1 přiřadíme třídu *switch_value*, která umístí přepínač na pravou stranu oddílu vytvořeného z objektu temp0. Do objektu temp1 je následně vložen objekt temp2 s třídou *switch*, který nastaví velikost přepínače. Objekt temp3 má funkci elementu <input> a stará se o posílání dat na webový server. Temp3 jsou přiřazeny atributy *data-value* a *data-input*, které slouží jako identifikátory do příjímání a posílání dat, atribut *type* na checkbox, který nastaví vstup jako přepínač s booleovskou logikou a atribut *value* na Hodnou "0". Dále je pomocí příkazu

.addEventListener("focus",function(){submit_checkbox(vstup)})

nastaven to, že při vybrání přepínače program automaticky zavolá funkci submit_checkbox(vstup), která se postará o zaslání dat z přepínače na webový server. Jelikož je temp3 umístěn ve stromové struktuře pod temp2 s třídou switch, je pomocí CSS velikost tohoto vstupu nastavena tak, aby na stránce nebyl vidět. Objekt temp4 se stará o vizualizaci přepínače pomocí třídy *switch_slider*. Ten pomocí CSS do daného elementu vloží šedý obdélník o rozměru 50px na 26px se zaoblenými rohy s poloměrem 26px. Následně je do obdélníku vložen čtverec o délce strany 20px, který je umístěn 2px od levé strany obdélníku a vertikálně do středu předchozího obdélníku. Poté mu byl nastaven poloměr na 50 % své velikosti, z čímž se z něj stal kruh. Poté jsem nastavil vlastnosti vstupu, pokud je přepínač ve své druhé poloze. Pomocí třídy *input:checked* je šedý obdélník obarven na modrou a poloha přepínače je pomocí vlastnosti *transform* přesunuta na druhou stanu již modrého obdélníku.

Obr. 7: Ukázka vytvořeného přepínače

3.2.5. Přidání obrázku

3.2.5.1. Obyčejný obrázek

Pro vložení obrázku na webovou stránku byla vytvořena funkce n_img().

```
n_img(size_x,img)
```

Funkce pomocí DOM příkazů vytvoří 2 objekty, objekt temp1 pro element a temp 2 pro element <img>. Funkce dále nastaví atribute *src* na vstupní argument img, ve kterém se nachází adresa umístění požadovaného obrázku. Dále se také nastaví pomocí CSS stylu šířka obrázku podle vstupního argumentu size_x. Dále se nastaví objektu temp1 třída img, která se stará o správné

formátování obrázku. Po nastavení argumentů je element <img> vložen do objektu temp1. Ten se následně vloží do objektu skupina_prvku.

Pokud bude stránka spuštěna na dostatečně široké obrazovce, obrázek se objeví na levé straně stránky a z pravé stany bude obtékán textem. Pokud se ale stránka spustí na zařízení s menší šířkou obrazovky, jako například na mobilních telefonech, obrázek bude umístěn horizontálně do středu stránky a jakýkoliv text bude zapsán až pod ním. Pokud bude obrázek větší, než je momentální šířka obrazovky, obrázek bude zmenšen tak, aby se na ní celý vešel.

3.2.5.2. Ikona v textu

Pro přidání ikonky do textu stránek byla vytvořena funkce n_icon().

```
n_icon(img)
```

Tato ikona má danou výšku podle základního textu a lze jí vložit do řádku s textem. Funkce vytvoří pomocí DOM příkazů 2 objekty, objekt temp1 pro element a temp 2 pro element <img>. Funkce poté nastaví objektu temp2 atribut src podle vstupního argumentu na cestu k této ikoně a pomocí atributu style se nastaví výška ikony na 1em. Dále je do temp1 přidána třída div_inline, která se stará o správné vložení ikony na správné místo. Objekt je následně vložen do objektu *skupina_prvku*.

3.2.5.3. Obrázek závislý na vstupních hodnotách

Pro vložení obrázku na webový server a jeho změnu podle momentální hodnoty argumentu vstup byla vytvořena funkce n_img_change().

```
n_img_change (size_x,vstup,val1,img1,val2,img2,val3,img3,val4,img4)
```

Podobně jako u funkce n_text_change() se po načtení prvku na stránce v prvku zobrazí takový obrázek, jehož přidružený argument val bude mít stejnou hodnotu jako argument vstup. Funkce vytvoří objekty temp1 pro element a temp 2 pro element <img>. Objektu temp2 se nejprve nastaví třída *img_change*, atribut *style* pro nastavení šířky obrázku podle vstupního argumentu size_x, prázdný argument *src* a argumenty *data-value*, *data-val*(i) a *data_text*(i) s hodnotami z příslušných vstupních argumentů této funkce. Objekt temp1 má stejnou funkci jako objekt temp1 z předchozí funkce n_img(). Objekt je následně vložen do objektu *skupina_prvku*.

3.2.5.4. Ikona závislá na vstupních hodnotách

Pro přidání proměnné ikony do textu na webové stránce byla přidána funkce n_icon_change().

```
n_icon_change(vstup,val1,img1,val2,img2,val3,img3,val4,img4)
```

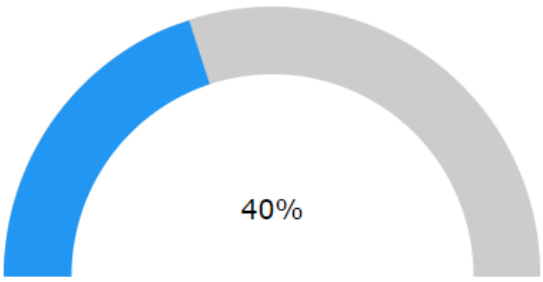
Funkce je identická s funkcí n_img_change(),kde pouze objekt temp1 má místo třídy *img* třídu *inline_div* a objekt temp2 má pouze jednu třídu *img_change*. Objekt temp2 má taktéž změněný CSS styl a jeho výška je nastavená na 1em.

3.2.6. Přidání vizuálního ukazatele

Pro přidání vizuálního ukazatele na webovou stránku, který zobrazuje procentuální stav vstupní veličiny, byla vytvořena funkce n_gauge().

```
n_gauge(vstup,min_val,max_val)
```

Funkce vytvoří pomocí DOM příkazů 4 objekty temp1, temp2, temp3 a temp4 s vlastností elementu . Dále objektu temp1 přidá třídu *gauge*, která slouží ke správnému umístění prvku na webovou stránku. Gauge bude umístěn na levé straně stránky a umožní případnému textu obtékat po jeho pravé straně, při menší šířce obrazovky ale bude umístěn do středu stránky a případný text bude umístěn pod ním. Velikost tohoto oddílu bude sice proměnná, ale bude mít pevný poměr šířky a výšky 2:1. Objektu temp2 bude přidaná třída s názvem *gauge_body*, která se stará o grafické znázornění tohoto oddílu s pomocí CSS. Oddíl bude o velikosti oddílu z objektu temp1, horní rohy budou zaobleny tak, že se z oddílu stane polokruh a barva pozadí je nastavena na světle šedou. Dále je objektu temp3 přidána třída *gauge_fill* a jsou nastaveny atributy *dataset-min, dataset-max* a *dataset-value* podle vstupních argumentů funkce a prázdný atribut *value*. Pomocí CSS je barva oddílu nastavena na modrou a je u něj definován bod otáčení ve středu své horní hrany. Díky tomuto nastavení je následně možné pomocí CSS příkazu *transform: rotate* natáčet daný prvek a tím graficky zobrazovat hodnotu tohoto ukazatele. Poslednímu objektu temp4 je přiřazena třída *gauge_cover*, který nastaví velikost a umístění tohoto oddílu do spodní části oddílu z objektu temp2, nastaví jeho barvu na bílou a upraví jeho tvar tak, aby se z něj stal polokruh. Posléze je pomocí příkazu *.innerHTML* zobrazena hodnota tohoto ukazatele v procentech. Po nastavení všech objektů jsou objekty temp3 a temp4 přidány do objektu temp2, objekt temp2 následně přidán do objektu temp1 a ten posléze do objektu skupina_prvku.



Obr. 8: Ukázka vytvořeného vizuálního ukazatele

3.2.7. Přidání posuvníku

Pro přidání posuvníku na webovou stránku, který slouží ke změně hodnot posílaných na webový server, byla vytvořena funkce n_slider().

```
n_slider(vstup,min_val,max_val,text)
```

Samotná funkce vytvoří pomocí příkazů DOM 4 objekty. Objekty temp1, temp3 a temp4 mají vlastnost elementu a objekt temp2 má vlastnost elementu <input>. Po vytvoření objektů funkce přidá objektu temp1 třídu *slider_div*, který zajišťuje správné vložení prvku na webové stránky. Objektu temp2 se nejprve změní atribut *type* na *range*, který přenastaví prvek HTML vstupu na posuvník. Do objektu temp2 je následně přidán příkaz:

```
.addEventListener("focus",function(){submit_value (vstup)})
```

který zajistí, že při vybrání posuvníku se automaticky zavolá funkce submit_value(), která se postará po zaslání dat na webový server. Dále se pomocí hodnot ze vstupních argumentů nastaví atributy *min*, *max*, *dataset_value* a *dataset-input* a objektu se přidají třídy *slider* a *text_input*. Pomocí těchto tříd se nastaví CSS styly posuvníku. Objektu temp3 se přidá třída *slider_text*, atribut *data-value* a pomocí příkazu *.innerHTML* se nastaví popisek posuvníku. Nakonec se objektu temp4 přiřadí třídy *slider_value* a *text_input* a atribut *data-value*. Objekty temp2, temp3 a temp4 se přidají do objektu temp1 a objekt temp1 je poté přidán do objektu skupina_prvku

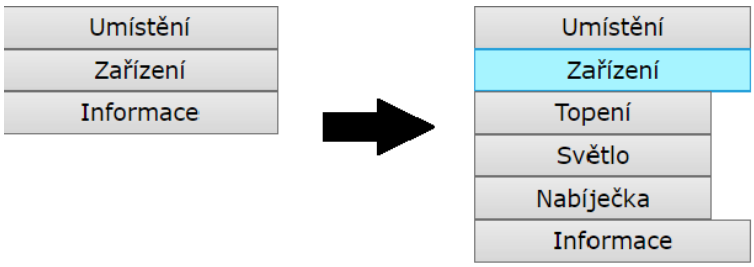
Obr. 9: Ukázka vytvořeného posuvníku

3.3. Komunikace s PLC

3.3.1. Zobrazení dat

Pro správnou komunikaci webových stránek s PLC je nutné nejdříve zobrazit požadované bloky jednotlivých zařízeních připojených k systému PLC. Pro vytvoření seznamu těchto zařízení je použit vytvořený soubor s názvem seznam_zařízeni.js. Poté, co jsem v tomto souboru definoval jednotlivé funkce pro tvorbu předpřipravených zařízení, jsem vytvořit JSON objekt s názvem Seznam_zarizeni, který v sobě obsahuje informace o všech unikátních zařízeních připojených k PLC. Jednotlivé zařízení jsou zaznamenány jako seznam prvků JSON. Každý prvek z tohoto pole obsahuje unikátní název zařízení, typ zařízení, jeho umístění a jméno databloku v PLC, ve kterém se nacházejí tagy daného zařízení. Dále každý prvek pole může obsahovat i proměnné s názvy Konst_1 až Konst_15, které mohou sloužit k přidání unikátního obsahu do vytvořených bloků.

Samotné zobrazení těchto bloků je inicializováno pomocí tlačítek umístěných v menu těchto webových stránek. Nejprve jsem v oddílu Menu vytvořil dvě tlačítka s názvy Umístění a Zařízení a pod každým tlačítkem jsem vytvořil oddíly s názvy *dropdown-room* a *dropdown-typ*, ve kterých budou umístěna tlačítka pro zobrazení zařízeních filtrovaných dle textu stištěného tlačítka na webovou stránku. Hlavním tlačítkům jsem následně přidal vlastnost onclick, která při stisknutí tlačítka spustí funkce button_drop_ro() a button_drop_pr(). Funkce se starají o defaultní skrytí oddílů vytvořených pod těmito tlačítky a jejich zobrazení jednoho z nich v případě stisknutí příslušného tlačítka. Tyto funkce jsem umístil do nového souboru button_drop.js v podadresáři Scripty.



Obr. 10: Rozklikávací menu

O vytvoření obsahu oddílů *dropdown-prvky* a *dropdown-room* tohoto menu se stará script nastaveni.js. Tento script při načtení stránky přečte JSON data z objektu Seznam_zarizeni a uloží je jako seznam objektů s názvem data. Tento seznam je následně projet algoritmem který postupně zjistí všechny unikátní hodnoty prvků data[i].Room a data[i].Typ a uloží je do objektu filterObj. Po zjištění všech unikátních hodnot prvků Room a Typ funkce pomocí DOM příkazu najde v HTML souboru element s identifikátorem *dropdown-room* a vymaže jeho obsah. Poté se cyklicky pro každý unikátní prvek v poli filterObj.Room vytvoří s pomocí DOM funkce *.createElement("button")* tlačítko, které v sobě obsahuje text tohoto unikátního prvku. Dále je tomuto prvku přidána pomocí příkazu *.addEventListener('click', function(e){ });* vlastnost, že po jeho stisknutí je spuštěna funkce hledej(this.textContent), která se postará o vytvoření bloků jednotlivých zařízení, které se nacházejí v příslušném místě, a funkce updateVal(), která to těchto

bloků vloží aktuální hodnoty z PLC. Podobným postupem vytvoříme obsah oddílu *dropdown-prvky*, ve kterém se budou nacházet tlačítka s názvy daných zařízeních. Tento script jsem umístil do podadresáře s názvem Scripty.

Při stisknutí příslušného tlačítka v menu se pomocí funkce hledej() vytvoří obsah webových stránek. Tato funkce se nachází v souboru hledej.js a je umístěná v podadresáři Scripty. Tato funkce nejprve vymaže všechen obsah v oddílu Content a následně z JSON objektu Seznam_zarizeni ze souboru seznam_zarizeni.js vytvoří pole objektů s názvem data. Poté postupně projede všechny prvky pole objektů data a porovnává hodnoty data[i].Room a data[i].Typ se vstupním argumentem této funkce, který obsahuje název tlačítka který vyvolá funkci hledej(). Pokud se vstupní argument input a jeden z těchto dvou prvků shodují, tak funkce pomocí DOM příkazu vytvoří nový prázdný oddíl s názvem skupina_prvku a nastaví jeho třídu na *div_prvku*. Třída zajistí vytvoření tohoto oddílu pod ostatními oddíly a vytvoří vnitřní okraje tohoto oddílu pro komfortnější zobrazení svého obsahu. Funkce poté zavolá funkci funkce_prvku(), která se nachází v souboru funkce_zarizeni.js. Tato funkce porovná hodnotu data[i].Typ s názvy předpřipravených funkcí ze souboru seznam_zarizeni.js a v případě schody tuto funkci spustí s hodnotami získanými z objektu data[i]. Funkce následně vykreslí požadovaný obsah do oddílu skupina_prvku a tu pomocí příkazu *.append()* umístí daný blok do oddílu Content, který slouží z zobrazení dat na webové stránce.

3.3.2. Čtení dat

Po vytvoření obsahu webových stránek je nutné jejich obsah v pravidelných intervalech aktualizovat. Pro tuto aktualizaci dat jsem vytvořil soubor komunikace.js v podadresáři Scripty a v něm vytvořil funkci updateVal(). Tato funkce pomocí knihovny Javascriptové knihovny jQuery čte data z webové stránky seznam_prvku.htm a ty používá k dynamickým změnám obsahu stránky. Tato funkce je pomocí příkazu

```
var intervalID = setInterval(function(){updateVal();}, 1000);
```

volána v určitém časovém cyklu. Obsahem webové stránky seznam_prvku.htm je seznam tagů vstupů do PLC a poté seznam všech tagů použitých na webovém serveru společně s jejich hodnotami. Obsah této stránky je přibližně následující.

```
<!-- AWP_In_Variable Name='"DB".Out_1' -->
<!-- AWP_In_Variable Name='"DB".Out_2' -->;
"DB".Out_1= := "DB".Out_1:;
"DB".Out_1= := "DB".Out_2:;
```

Po nahrání na webový server PLC se části uvedené v :=“název tagu“: automaticky změní text s hodnotami vepsaných tagů v PLC. Pro načtení obsahu požadované webové stránky jsem použil funkci $.get(), která slouží k nahrání obsahu z dokumentu pomocí HTML metody GET. Po přečtení těchto dokumentu se data rozdělí podle středníků na pole Stringů. Funkce následně spustí cyklus, který postupně projede všechny jednotlivé Stringy krom prvního, ve kterém se nacházejí pouze AWP příkazy pro definování vstupů. V každém cyklu je daný String rozdělen symboly “= “ na Stringy tag[1], který obsahuje název PLC tagu a tag[2] ve které je hodnota daného tagu. Po získání těchto dat jsou spuštěny následující funkce.

3.3.2.1. Aktualizace posuvníků

```
update_slider(tag)
```

Funkce update_slider() slouží k nastavení počáteční hodnoty posuvníku na hodnotu příslušného tagu z PLC. Funkce pomocí příkazu *.getElementsByClassName("slider")* nalezne v dokumentu všechny prvky s třídou *slider*. Funkce následně cyklicky projede všemi najitými prvky v dokumentu a zkouší jestli se atribut s názvem *name* daného prvku neshoduje s hodnotou argumentu tag[0]. Pokud se tyto proměnné schodují, tak funkce tomuto prvku pomocí příkazu *.setAttribute("value",tag1[1])* přidělí atributu *value* hodnotu tagu tag[1] a tím přesune knoflík posuvníku do správné pozice.

3.3.2.2. Aktualizace přepínačů

```
update_checkbox(tag)
```

Funkce update_checkbox() slouží k nastavení počáteční hodnoty přepínače na stav v tagu z PLC. Funkce pomocí příkazu *.getElementsByClassName("checkbox")* nalezne v dokumentu všechny prvky s třídou *checkbox*. Funkce následně cyklicky projede všemi najitými prvky v dokumentu a zkouší jestli se atribut s názvem *data-value* daného prvku neshoduje s hodnotou argumentu tag[0]. Pokud se tyto proměnné shodují, tak funkce tomuto prvku pomocí příkazu *.checked* přidělí atributu *checked* hodnotu tagu tag[1] a tím přenastaví přepínač do správné polohy.

3.3.2.3. Aktualizace textu na hodnotu PLC tagu

```
update_text(tag)
```

Tato funkce slouží k vypsání hodnoty tagu z PLC na webovou stránku. Tato funkce pomocí příkazu *.getElementsByClassName("text_input")* nalezne v dokumentu všechny prvky s třídou *text_input,* cyklicky projede všemi najitými prvky v dokumentu a zkouší jestli se atribut s názvem *data-value* daného prvku neshoduje s hodnotou argumentu tag[0]. Pokud se tyto proměnné shodují, tak funkce tomuto prvku pomocí příkazu *.innerHTML* přidělí text s hodnotou tagu tag[1].

3.3.2.4. Aktualizace textu ve tvaru procent

```
update_percent(tag)
```

Tato funkce slouží k vypsání hodnoty tagu z PLC na webovou stránku v procentuální podobě. Tato funkce pomocí příkazu *.getElementsByClassName("percent_input")* nalezne v dokumentu všechny prvky s třídou *percent_input,* cyklicky projede všemi najitými prvky v dokumentu a zkouší, zda se atribut s názvem *data-value* daného prvku neshoduje s hodnotou argumentu tag[0]. Pokud se tyto proměnné shodují, tak funkce tomuto prvku zavolá funkci percentage(), který vrátí hodnotu tag[1] v podobě procent. Pro zjištění hodnoty 100% funkce přečte hodnotu atributu data-value2, ve které je tato hodnota uložena. Pomocí příkazu *.innerHTML* následně změní obsah prvku na výstup funkce percentage().

3.3.2.5. Aktualizace vizuálních ukazatelů

`update_gauge(tag)`

Tato funkce slouží ke správnému vykreslení grafického měřidla typu gauge. Tato funkce pomocí příkazu .getElementsByClassName*("gauge_fill")* nalezne v dokumentu všechny prvky s třídou *gauge_fill*, cyklicky projede všemi najitými prvky v dokumentu a zkouší, zda se atribut s názvem *data-value* daného prvku neshoduje s hodnotou argumentu tag[0]. Pokud se tyto proměnné shodují, tak funkce tomuto prvku zavolá nejprve funkci percentage(), který vrátí hodnotu tag[1] v podobě procent. Pro zjištění hodnoty rozsahu měrky funkce přečte hodnotu atributu data-min a data-max, ve kterých jsou tyto hodnoty uloženy. Po zjištění procentuální hodnoty tag[1] se pomocí CSS příkazu *.style.transform* nastaví hodnota natočení oddílu gauge_fill na *'rotate(+'+g_pro/200+'turn)'*, kde g_pro je výstup z funkce percentage().

3.3.2.6. Aktualizace obrázku ze seznamu

`update_img(tag)`

Tato funkce slouží k zobrazení jednoho obrázku ze seznamu dle hodnoty PLC vstupu. Tato funkce pomocí příkazu .getElementsByClassName*("img_change")* nalezne v dokumentu všechny prvky s třídou *img_change*, cyklicky projede všemi najitými prvky v dokumentu a zkouší, zda se atribut s názvem *data-value* daného prvku neshoduje s hodnotou argumentu tag[0]. Pokud se proměnné shodují, tak funkce u tohoto prvku následně porovnává hodnoty tag[1] a data-val[i]. V případě, že se hodnota data-val[i] a tag[1] shodují, tak atribut *src* obrázku je nastaven na hodnotu atributu data-img[i], který obsahuje cestu k volanému obrázku. V případě kdy se ani jedna hodnota data-val[i] nerovná hodnotě tag[1] se zobrazí obrázek defaultní.

3.3.2.7. Aktualizace textu ze seznamu

`update_text_switch(tag)`

Tato funkce slouží k vypsání textu ze seznamu dle hodnoty PLC vstupu. Tato funkce pomocí příkazu .getElementsByClassName*("txt_switch")* nalezne v dokumentu všechny prvky s třídou *txt_switch*, cyklicky projede všemi najitými prvky v dokumentu a zkouší, zda se atribut s názvem *data-value* daného prvku neshoduje s hodnotou argumentu tag[0]. Pokud se proměnné shodují, tak funkce u tohoto prvku následně porovnává hodnoty tag[1] a data-val[i]. V případě, že se hodnota data-val[i] a tag[1] shodují, tak se text v oddílu změní na text zapsaný v atributu data-txt[i]. V případě kdy se ani jedna hodnota data-val[i] nerovná hodnotě tag[1] se zobrazí defaultní text.

3.3.2.8. Aktualizace času

`update_time(tag)`

Tato funkce slouží k vypsání textu ze seznamu dle hodnoty PLC vstupu. Tato funkce pomocí příkazu .getElementsByClassName*("time_input")* nalezne v dokumentu všechny prvky s třídou

time_input, cyklicky projede všemi najitými prvky v dokumentu a zkouší, zda se atribut s názvem *data-value* daného prvku neshoduje s hodnotou argumentu tag[0]. Pokud se proměnné shodují, tak funkce pomocí funkce *.replace()* dekóduje proměnou tag1[1] aby jeho výsledná hodnota byla ve tvaru hh:mm:ss. Takto upravená proměnná se následně vloží do atributu *value* daného prvku.

1.1.1. Zápis dat

Posílání dat z webových stránek do PLC je prováděna pomocí jQuery metody GET. Tato metoda se používá primárně pro čtení dat, je jí ale i možné zasílání dat na webový server. V souboru komponenty.js jsem vytvořil dvě nové funkce submit_value() a submit_checkbox(), které jsou volány při vybrání prvku vstupu na webových stránkách. Tyto funkce po načtení naleznou v HTML dokumentu stránek první vstupní prvek s atributem data-input s hodnotou rovnou názvu hledaného PLC tagu a nastaví mu vlastnost *.oninput*, která při změně hodnoty daného vstupu zavolá funkci writeVal(). Rozdílem mezi těmito dvěma funkcemi je to, že funkce submit_value() posílá do funkce writeVal() číselnou hodnotu zatímco submit_checkbox() posílá stav přepínače v booleovské logice. Dále byly vytvořena funkce submit_time(), která se stará o zaslání dat času na server ve formátu TOD. Funkce je podobná funkci submit_value(), jen před zasláním dat na server je k hodnotě času je přidán String “TOD#“ a výsledná proměnná je následně kódována příkazem encodeURIComponent().

Funkce writeVal(), kterou jsem vložil do souboru komunikace.js, se stará o zasílání samotných dat na server. Pomocí jQuery funkce $.get() si program vyžádá z webového serveru dokument seznam_prvku.htm. Zároveň si ale pošle požadavek na server ve tvaru input=val, kde input je název PLC tagu a val je hodnota, kterou danému tagu chceme přidělit.

1.2. Provedení aktualizací

Po vytvoření komunikace webových stránek se serverem na PLC jsou stránky teoreticky připraveny k provozu. Menší potíže však nastanou, pokud budeme chtít na webové stránky zobrazit další zařízení připojené k PLC anebo definovat zařízení zcela nové. Pro připojení nového zařízení je nutné do souboru seznam_zarizeni.htm přidat příslušné AWP příkazy a PLC tagy pro správnou komunikaci s PLC. Dále je v případě definování nových zařízení nutné přidat do funkce funkce_prvku v souboru funkce_zarizeni.js přidat kus kódu, který danou novou funkci dokáže správně zavolat. Protože by se tyto soubory museli upravovat při každé větší změně programu a při jejich úpravě by mohlo dojít k chybám, rozhodl jsem se vytvořit funkci create_file(), která tyto 2 soubory automaticky vytvoří.

Funkce create_file() nejprve vytvoří String s názvem Funkce_prvku, ve kterém bude umístěn text JS kódu, který vytváří funkci volající příslušné funkce pro tvorbu webu. Do tohoto Stringu je po jeho vytvoření vložen text "function funkce_prvku(data,i){", který definuje název a argumenty dané funkce. Funkce dále vytvoří prázdný String SeznamAWP a String SeznamOUT, který v sobě již obsahuje středník. Program následně otevře soubor seznam_zarizeni.js a postupně ho prochází řádek po řádku. Pokud se na řádku nachází definice funkce, tak tento řádek rozdělí na několik

menších Stringů ve kterých je uložený název dané funkce a jeho vstupní argumenty. Nejprve je do Stringu Funkce_prvku přidán text JS kódu

```
'if (data[i].Typ === 
"'+Seznam_funkci2[0]+'"'+'){'+Seznam_funkci2[0]+'(data[i].Jmeno,data[i]
.Room,data[i].DB';
```

, který při splněné podmínce slouží pro zavolání funkce s daným jménem. Dále funkce projde všechny Stringy s jmény vstupních argumentů. Pokud se hodnota těchto Stringů shodná s hodnotami Out_[i], In_[i], nebo Konst_[i], tak k rozepsané funkci přidá text JS kódu, který přidá nový argument do dané funkce. Po přidání potřebných argumentů je do Stringu přidán ukončovací text ")}" a program pokračuje v hledání dalších funkcí. Po nalezení poslední funkce nakonec vloží do Stringu Funkce_prvku symbol "}", který ukončuje kód funkce funkce_prvku().

Tato funkce při nalezení nové funkce zároveň v souboru seznam_zarizeni.js načte a projde všechny JSON objekty, které jsou v tomto souboru uloženy a hledá objekt s vlastností Typ o stejné hodnotě jako je název nalezené funkce. Funkce pro každý nalezený objekt prochází jednotlivými Stringy vstupních argumentů a hledá argumenty s názvy Out_[i] a In_[i]. V případě nalezení argumentu s názvem In_[i] je programem do Stringu s názvem SeznamAWP přidán text

```
'<!-- AWP_In_Variable Name='+"'"+'"'+data[k].DB+'".Out_'+l+"'"+' -->'
```

, který vytvoří AWP příkaz sloužící k definování tagu v PLC jako tagu vstupního. Zároveň do Stringu SeznamOUT je přidán text

```
'"'+data[k].DB+'".Out_'+l+'= :="'+data[k].DB+'".Out_'+l+':;'
```

, který v souboru vypíše název PLC tagu společně s aktuální hodnotou daného tagu v PLC. Tento postup se ještě jednou opakuje pro vytvoření tagů typu Out_[i].

Po vytvoření těchto Stringů program spojí Stringy SeznamAWP a SeznamOUT do jednoho s Stringu s názvem content. Funkce dále pomocí JS kódu

```
var a = document.createElement("a");
a.href = window.URL.createObjectURL(new Blob([content], {type:
"text/plain"}));
a.download = "seznam_prvku.htm";
a.click()
```

tento String stáhne do počítače jako HTML soubor s názvem seznam_prvku.htm. Nakonec funkce provede stejný postup se Stringem Seznam_funkci, který uloží do počítače JS soubor s názvem funkce_zarizeni.js.

Pro spuštění této funkce jsem na novou webovou stránku vytvořil tlačítko, které při stisknutí zavolá funkci create_file(). Tato funkce automaticky otevře seznam souborů pro nahrání souboru seznam_zarizeni.js. Po vykonání dané funkce je spuštěno stahování 2 souborů, které je následně nutné vložit do složky s webovými stránkami pro přepsání předchozích souborů. Webovou stránku, na kterou jsem umístil toto tlačítko, je možné zobrazit pomocí nového tlačítka umístěného v menu stránek.

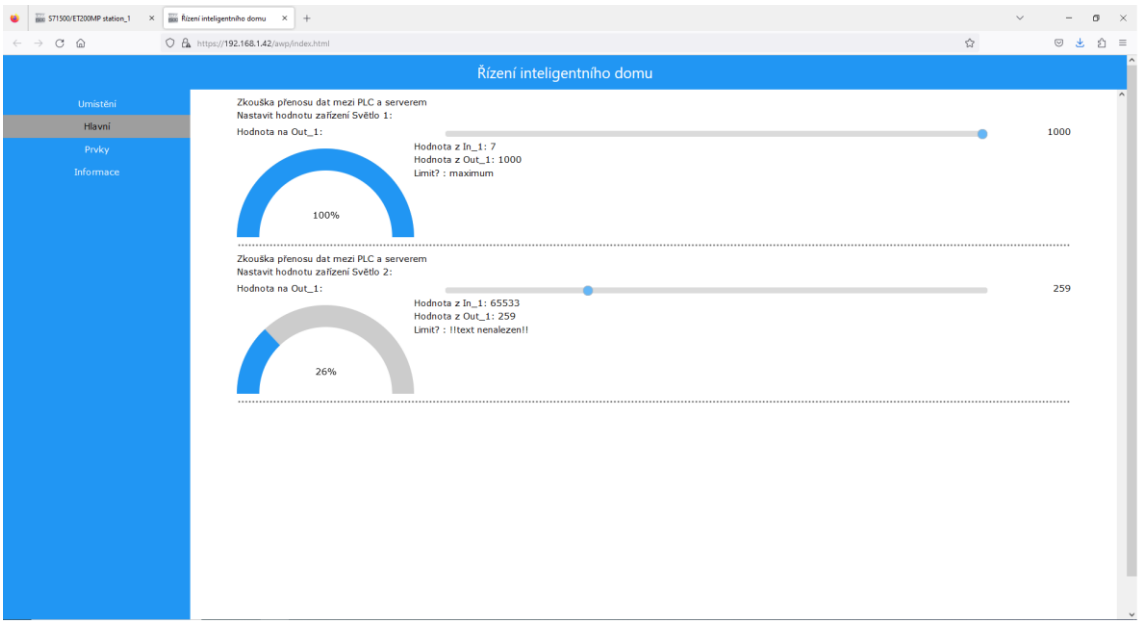
1.3. Grafické zpracování stránek

Po vytvoření funkčních webových stránek je již zapotřebí jen jejich grafická úprava. Pro vytvoření některých grafických aspektů stránek byl použit framework W3.css [16], který se používá pro tvorbu responzivních grafických prvků stránek. Ty části CSS, které nebudou použity z tohoto frameworku, budou zapsány do CSS souboru Styly.css. Části webové stránky pro zobrazení názvu stránky přiřadíme třídy *w3-row l12 m12 s12 title*. Třída w3-row zobrazí tento oddíl tak, že další elementy stránky se zobrazí až pod tímto oddílem. Skupina tříd *l12 m12 s12* se stará o nastavení šířky daného oddílu na této stránce pro různé velikosti obrazovky. Těmito třídami je tedy zajištěno, že tento oddíl bude široký 12/12 šířky obrazovky při jakékoliv šířce obrazovky. Poslední třídou nadpisu stránky je title, který se stará o nastavení barvy pozadí oddílu. Do tohoto oddílu byl následně přidán HTML element nadpisu <h1>, ve kterém je text nadpisu stránek.

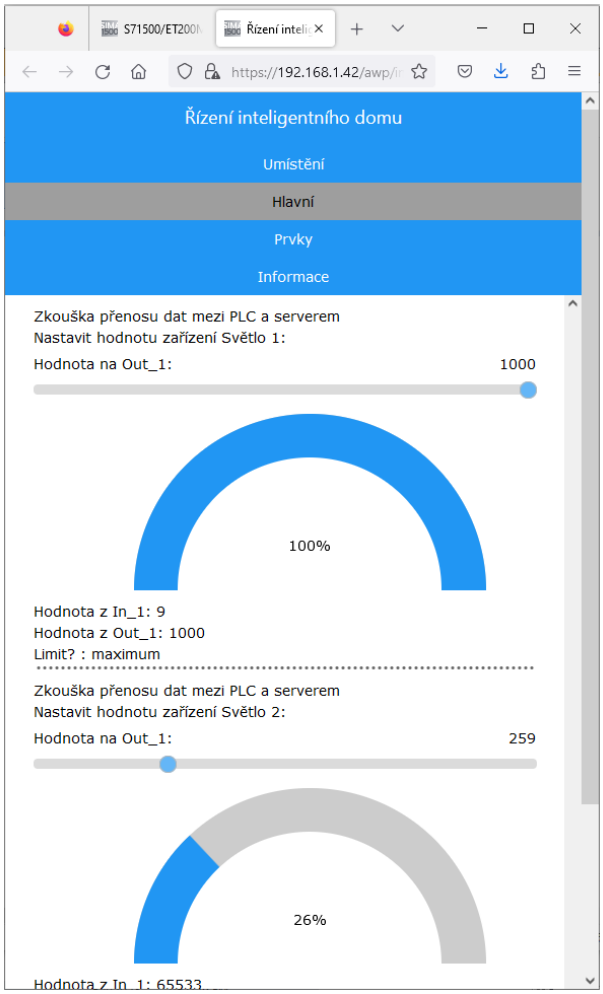
Dalším oddílem, který tvoří webové stránky, je oddíl Main, ve kterém se nacházejí oddíly Menu a Content. Oddílu Main byla podobně jako u nadpisu přidána třída *w3-row*. Dále byly do oddílu Menu přidány třídy *menu w3-col l2 m3 s12*. Třída w3-col zobrazí daný oddíl jako sloupec v oddílu Main. Šířka tohoto oddílu je proměnná v závislosti na šířce obrazovky. V případě veliké obrazovky, respektive obrazovky s šířkou větší než 994 pixelů, bude tento oddíl zabírat 2/12 šířky stránky. V případě střední obrazovky s šířkou nad 600 pixelů bude oddíl zabírat 3/12 stránky a při malé obrazovce bude prvek roztáhnut na celou obrazovku. Třída menu zajišťuje nastavení barvy tohoto oddílu a vytvoří horní vnitřní okraj v oddílu s velikostí 5px.

Dále byly upraveny tlačítka v tomto menu přidáním tříd *w3-button w3-block w3-blue w3-hover-sand*. Třídy *w3-button* a *w3-block* zobrazí tlačítko jako jednobarevný obdélník o šířce nadřazeného oddílu a s textem uprostřed tlačítka. Dále jsou pomocí tříd w3-blue a w3-hover-sand nastaveny barvy tohoto tlačítka na modrou, která se při najetí myši změní na béžovou. Stejným postupem byl změněn styl pro tlačítka, které se nacházejí v rozbalovací části menu. Jediným rozdílem od předchozích tlačítek byla změna jejich barvy na šedou kvůli nutnosti rozpoznání jednotlivých částí menu.

Posledním oddílem tohoto webu je oddíl Content. Tomuto oddílu byly přidány třídy *Content w3-col l10 m9 s12*. Třídy *l10 m9* zajistí, aby byl obsah tohoto oddílu umístěn vedle oddílu menu, zatímco třída *s12* způsobí to, že tento oddíl bude zobrazen až pod oddílem menu. Třída *Content* se stará především o zobrazení rolovací lišty v případě většího množství prvků na stránce.



Obr. 11: Konečná podoba webových stránek na PC

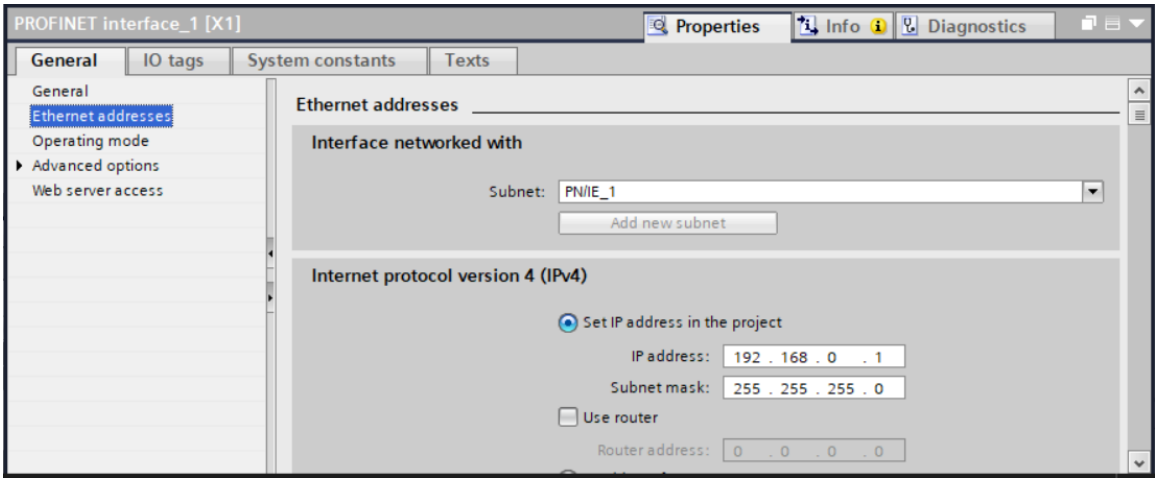


Obr. 12: Podoba webových stránek na obrazovce s menší šířkou

1.4. TIA Portál

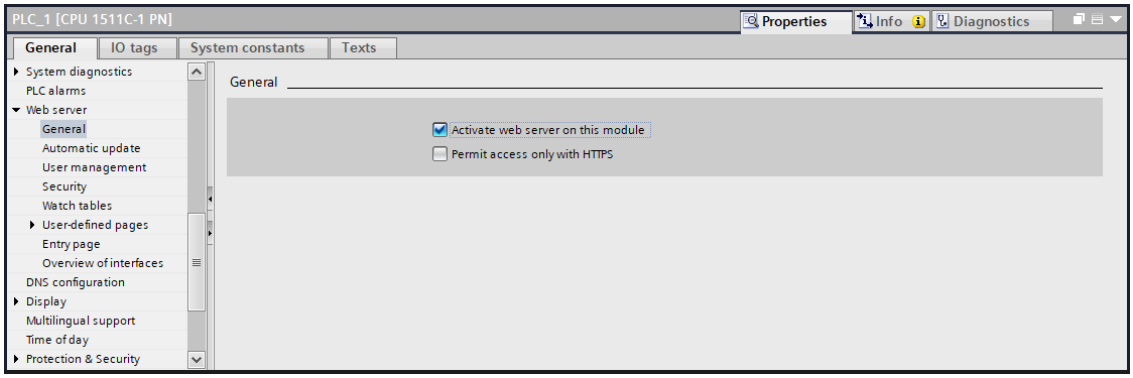
1.4.1. Nahrání webový stránek

Pro zprovoznění webových stránek je nutné jejich umístění do PLC. Před tímto bylo však nutné v PLC zprovoznit webový server. Jakožto první věc bylo nutné vytvořit komunikaci PLC s ostatními prvky na síti pomocí internetu. V TIA Portálu byl vytvořen nový projekt, do kterého bylo přidáno nové zařízení typu S7-1200. Toto zařízení bylo následně spárováno se skutečným PLC. Dále bylo v TIA Portálu pro dané PLC v záložce Ethernet addresses připojeno k internetové síti a byla mu přidělena IP adresa. Pomocí této adresy je možné k tomuto PLC přistupovat prostřednictvím počítače připojeného ve stejné síti.



Obr. 13: Nastavení IP adresy PLC

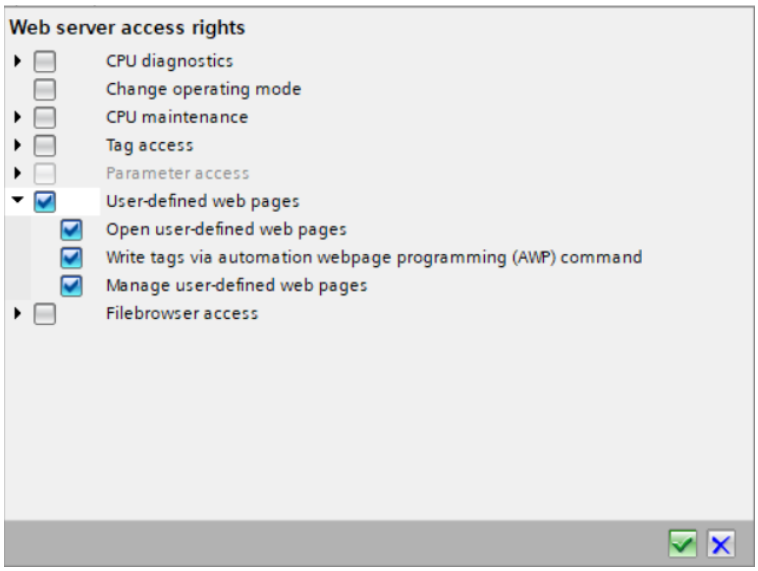
Dále je nutné zprovoznění webového serveru, které bylo provedeno zaškrtnutím možnosti *Activate web server on this module* v záložce Web server.



Obr. 14: Aktivování webového serveru v PLC

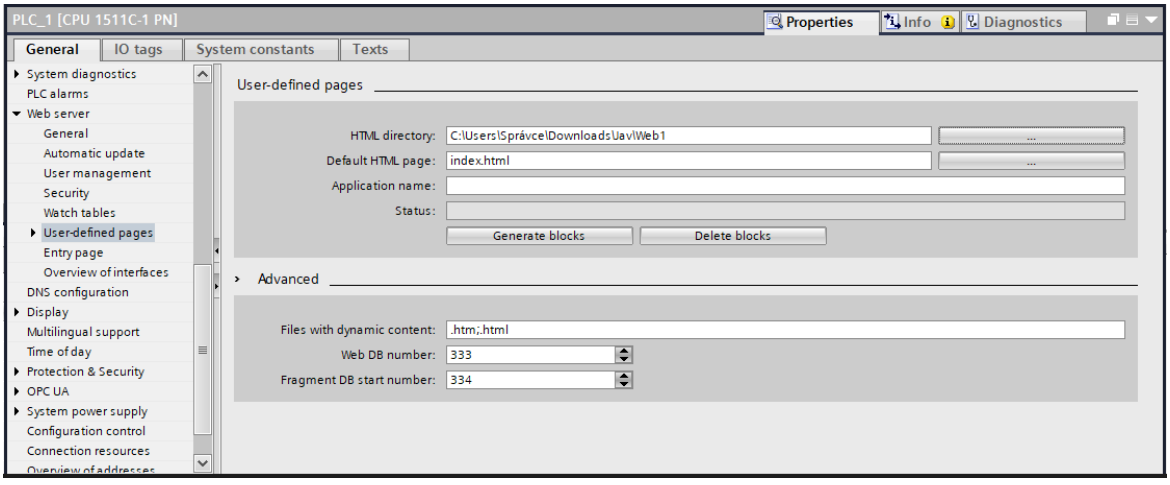
Po aktivaci webového serveru je nutné nastavení oprávnění jednotlivých uživatelů k využívání stránek. Toto nastavení provádíme v TIA Portálu v záložce User managment. Zde je na výběr možnost udělení uživatelských oprávnění k přístupu na webové stránky a oprávnění pro zápis dat v uživatelských webových stránek do PLC jednotlivým uživatelům. Tyto oprávnění je možné přiřadit jak jednotlivým uživatelům, kteří se pro jejich otevření musejí v hlavním menu, které se

zobrazí při zadání IP adresy daného PLC do prohlížeče, přihlásit pomocí unikátního jména a hesla, tak i ostatním uživatelům, kteří k těmto stránkám mohou přistupovat bez přihlašovacích údajů.



Obr. 15: Nastavení oprávnění k prohlížení webových stránek

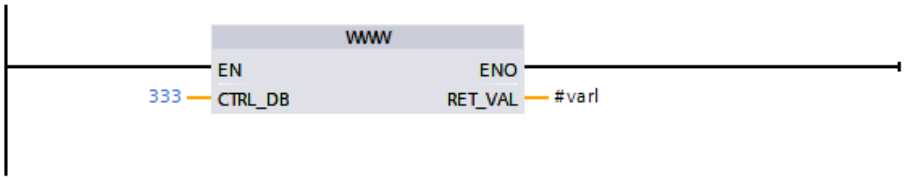
Poté, co byl webový server zprovozněn a byly nastaveny oprávnění k jeho prohlížení je nutné vytvořené stránky nahrát na tento server. V záložce User-defined pages zvolíme s pomocí prohlížeče souborů složku, ve které jsou připravené webové stránky umístěny a definujeme defaultní HTML dokument, který bude při spuštění stránek z PLC serveru automaticky načten. Dále zde v této záložce možné definovat typy souborů, ve kterých se nacházejí AWP příkazy a tagy pro čtení hodnot z PLC. Vytvořené webové stránky využívají tyto funkce pouze soubor seznam_prvku.htm a proto zde ponecháme výchozí hodnoty pro soubory s příponami .htm a .html. Je zde ještě možné změnit očíslování jednotlivých DB, ve kterých bude obsah těchto stránek uložen, ty však byly ponechány na defaultních hodnotách.



Obr. 16: Okno pro vkládání souborů webových stránek do PLC

Dále je nutné do programu PLC vložit blok WWW, který inicializuje spuštění těchto stránek. Tato systémová funkce byla proto vložena do OB1. Tomuto bloku je nutné pro jeho správnou funkci

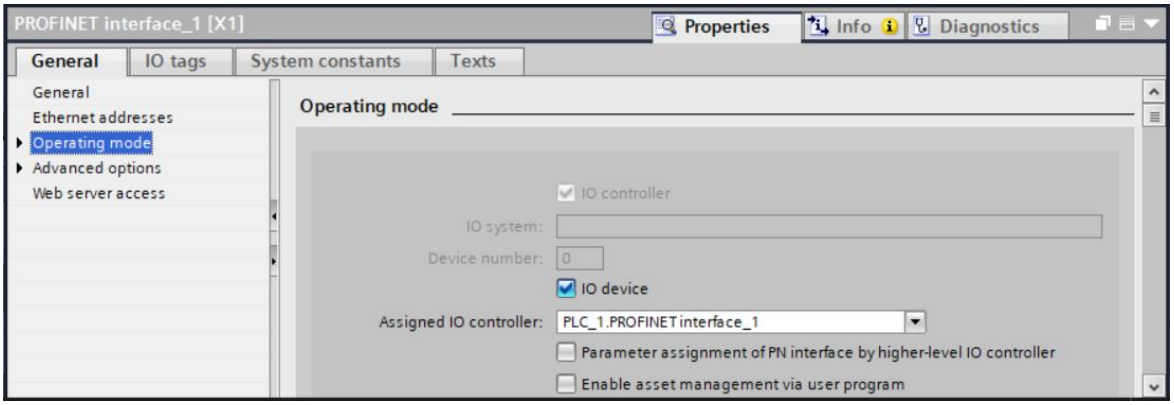
nastavit číslo BD, ve kterých jsou uloženy webové stránky a výstupní hodnotu RET_VAL, která slouží k indikaci stavu webového serveru.



Obr. 17: Funkce pro spuštění webového serveru

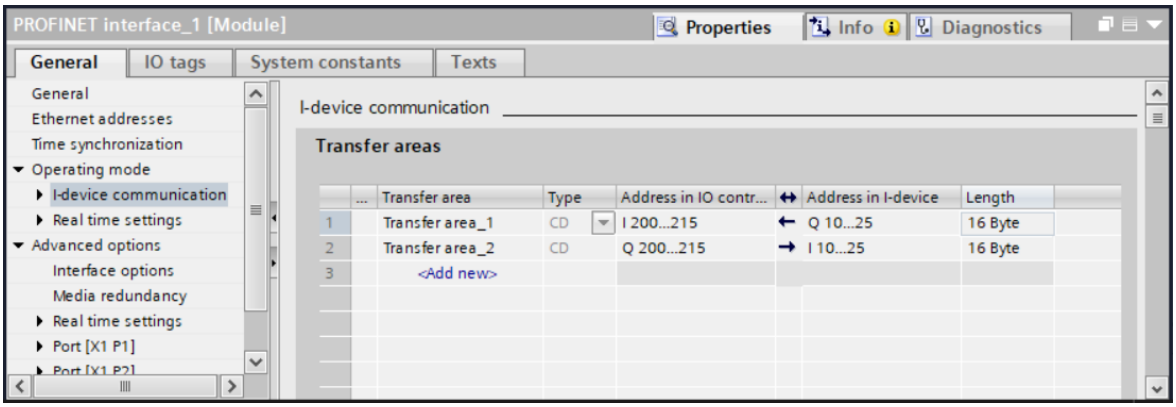
1.4.2. Zprovoznění komunikace PLC s PLC

Komunikace mezi 2 PLC je vytvořena metodou I-Device. Tato metoda umožnuje virtuální propojení výstupů jednoho PLC ke vstupům PLC druhého a naopak. Pro zprovoznění této metody je nutné nejprve obě PLC připojit ke stejné síti. Po jejich propojení je nutné v TIA Portálu hlavní PLC nastavit jako IO kontrolér a přiřadit mu sít, na které budou umístěny všechny podřízené PLC.



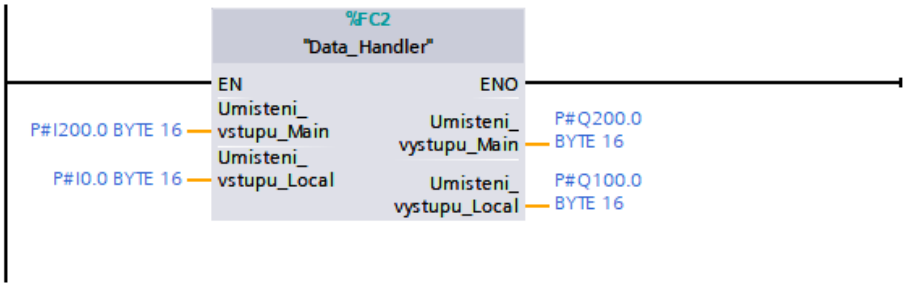
Obr. 18: Nastavení PLC jako IO device

Po nastavení hlavního PLC je nutné nastavit všechna další PLC, které budou k tomuto PLC připojeny. V TIA Portálu se u těchto podřízených PLC nejprve nastaví připojení k nadřízenému IO kontroléru. Po propojení těchto PLC je nutné vytvoření přenosových oblastí. Tyto oblasti fungují tak, že pokud jsou na výstupy podřízeného PLC, respektive do paměti Q o zadané adrese poslány data, tak se tyto data zobrazí v hlavním PLC na vstupní paměti I. Stejným způsobem lze spárovat výstupní data hlavního PLC ke vstupní paměti podřízeného PLC.



Obr. 19: Nastavení přenosových oblastí

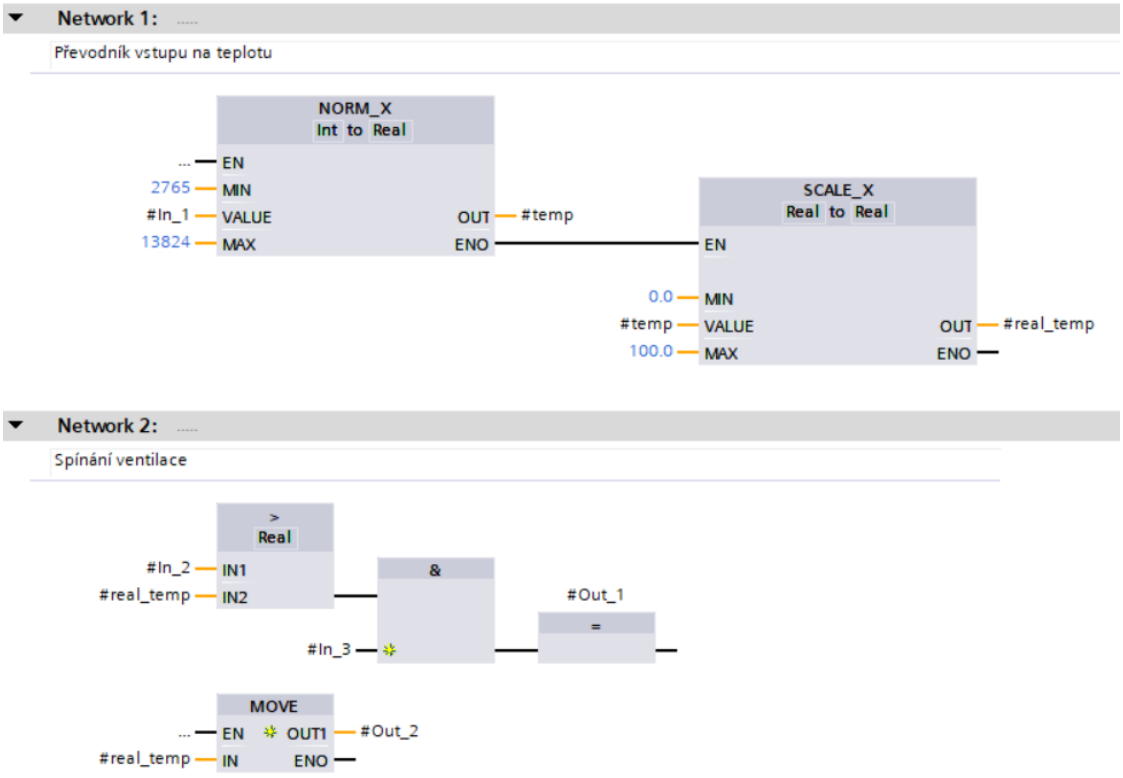
Jelikož však program zakazuje přenos dat mezi vstupy a výstupy které jsou již používané, je nutné tyto data nejprve poslat na jiné místo v pamětech I a Q. K tomuto přenosu byl vytvořen FC blok s názvem Data_handler. Tento blok je v podstatě tvořen 2 MOVBLK bloky, které posílají data ze vstupu na nepoužívané místo v paměti I, a naopak posílají data z nevyužívané paměti Q na výstupy ke kterým jsou připojena zařízení. Adresy těchto nevyužitých míst jsou použity v nastavení adresy přenosových oblastí pro přenos dat mezi PLC. Pro použití v PLC řady S7-1200 byla vytvořena obdobná funkce fungující na příkazu MOVE_BLK_variant.



Obr. 20: FC pro posílání mezi skutečnými IO a přenosovými oblastmi

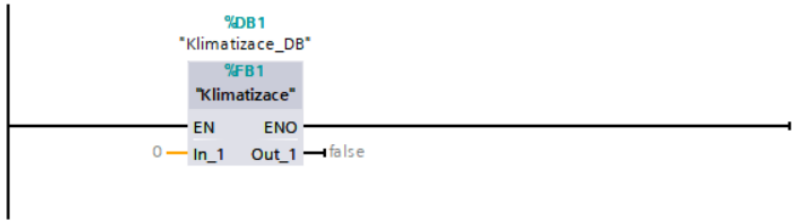
1.4.3. Vytvoření bloků pro zařízení v TIA Portálu

Pro připojení zařízení k předpřipraveným blokům na webových stránkách je nutné pro každé unikátní zařízení vytvořit vlastní FB. Pro znázornění jsem vytvořil FB pro klimatizaci, která udržuje v místnosti nastavenou teplotu. Teplota místnosti je snímána odporovým teploměrem Pt100 zapojeným do teplotního převodníku, který zobrazuje teplotu v rozmezí 0-100 °C pomocí napětí 1-5 V. U tohoto zařízení je nutné, aby ho bylo možné z webových stránek zapnout a vypnout, přečíst teplotu tohoto snímače a nastavit teplotu, při které se klimatizace zapne. V TIA Portálu byl tedy vytvořen FB, do jehož první části byly umístěny bloky NORM_X a SCALE_X, které vezmou hodnoty vstupu In_1 na který bude následně připojen snímač teploty. Tato teplota je následně porovnávána s hodnotou In_2 z webových stránek, která představuje teplotu, při které se má klimatizace zapnout. In_3 je připojen k tlačítku na webových stránkách které umožnuje spuštění klimatizace. Pokud je tedy teplota snímače vyšší než námi nastavená teplota a zároveň je funkce klimatizace zapnutá, tak program spustí tuto klimatizaci. Zároveň tento blok vypíše na web hodnotu Out_2, na které se nachází aktuální teplota snímače.



Obr. 21: Princip funkce FB Klimatizace

Po přidání FB do hlavního programu PLC je automaticky vytvořen nový DB, jehož název je následně zadán do scriptu seznam_zarizeni.js.



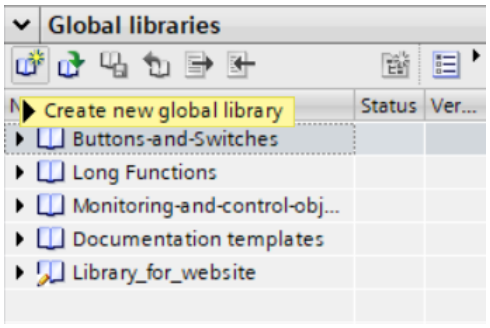
Obr. 22: FB Klimatizace

Při tvorbě těchto FB je důležité správné značení tagů v daném FB, protože z DB daného zařízení webový server čte pouze tagy s názvy In_[i] a Out_[i]. Pro webové stránky jsem následně vytvořil základní blok, pomocí kterého lze danou klimatizaci ovládat.

Obr. 23: Blok na webových stránkách pro FB Klimatizace

1.4.4. Knihovna zařízení

Pro uložení těchto bloků pro použití v nových projektech byla vytvořena globální knihovna, do které budou vloženy všechny bloky vytvořené v TIA Portálu pro nastavení komunikace s webovým serverem nebo s ostatními PLC. Takovou knihovnu lze vytvořit v okně Global libraries, kterou lze zobrazit z postranní listy Libraries. V tomto okně vytvoříme novou knihovnu, přičemž vložíme její název a zvolíme místo, kam bude na počítači uložena.



Obr. 24: Tvorba nové knihovny

Do této knihovny byly následně vloženy vytvořené FC a FB a ty byly následně roztříděny podle jejich využití do jednotlivých složek. V první složce se nachází prvky starající se o komunikaci mezi PLC. V druhé složce pojmenované Zařízení se nacházejí prvky, které připojují jednotlivá zařízení na webový server. V poslední části knihovny se nachází složka pro ostatní prvky. Tyto prvky slouží ke zjednodušení práce s ostatními prvky a patří mezi ně například FC Hodiny, která má na výstupu aktuální čas ve formátu TOD.

Obr. 25: Obsah vytvořené knihovny

Závěr

V práci byla vytvořena rešerše již existujících SCADA/HMI systémů, které se běžně používají pro zobrazování dat z PLC a pro nastavování proměnných v PLC.

Hlavním výsledkem této diplomové práce bylo vytvoření dynamických webových stránek a jejich propojení s PLC pomocí jejich umístění na webovém serveru v PLC. Nejprve byla vytvořena základní webová stránka a byly vytvořeny Javascriptové funkce pro tvorbu předpřipravených základních prvků, které se na tuto webovou stránku vkládají. Dále byly vytvořeny scripty pro komunikaci s PLC, pro aktualizaci jednotlivých prvků stránky podle vstupních hodnot a pro posílání dat do PLC. Webovým stránkám byla upravena jejich vizuální podoba a byla zajištěna změna vzhledu těchto stránek pro jejich ideální zobrazení při používání mobilního telefonu.

V programu TIA Portál byla poté vytvořena knihovna předpřipravených funkcí, které budou použity v programu PLC při použití dané webové aplikace.

Nakonec byla vytvořena stručná dokumentace s návodem pro používání webových stránek, vytváření dalších zařízení či pro nastavení TIA Portálu pro správnou komunikaci s ostatními PLC. Tato dokumentace byla následně vložena na webové stránky.

Seznam použité literatury

[1] Basics of PLCs: A quickSTEP Online Course [online]. 2016 [cit. 2023-01-05]. Dostupné z: https://sitrain.us/step/pdfs/version2/Basics_of_PLCs.pdf

[2] ŠMEJKAL, Ladislav a Marie MARTINÁSKOVÁ. PLC a automatizace. Praha: BEN - technická literatura, 1999. ISBN 80-86056-58-9.

[3] SIPLUS extreme – automation and drive technology for extreme ambient conditions. SIEMENS [online]. [cit. 2023-01-06]. Dostupné z: https://new.siemens.com/global/en/products/automation/products-for-specific-requirements/siplus-extreme.html

[4] SIMATIC STEP 7 and WinCC Engineering V17: System Manual [online]. 2021 [cit. 2023-01-07]. Dostupné z: https://cache.industry.siemens.com/dl/files/671/109798671/att_1071920/v1/STEP_7_WinCC_V17_enUS_en-US.pdf

[5] Basics on Creating HTMLs for SIMATIC CPUs: Application Description [online]. V1.0 02/2014. Siemens, 2014 [cit. 2022-10-17]. Dostupné z: https://support.industry.siemens.com/cs/attachments/68011496/68011496_html_basics_for_simatic_cpus_en.pdf

[6] W3schools [online]. [cit. 2023-01-21]. Dostupné z: https://www.w3schools.com

[7] PROMOTIC: SCADA visualization software [online]. [cit. 2023-01-19]. Dostupné z: https://www.promotic.eu/cz/pmdoc/WhatIsPromotic/WhatIsPromotic.htm

[8] SIEMENS [online]. [cit. 2023-02-07]. Dostupné z: https://www.siemens.com/global/en.html

[9] WinCC V7.5 SIMATIC HMI WinCC V7.5 Getting Started [online]. [cit. 2023-02-07]. Dostupné z: https://cache.industry.siemens.com/dl/files/479/109762479/att_968505/v1/GettingStartedenUS_en-US.pdf

[10] WebIQ [online]. [cit. 2023-01-20]. Dostupné z: https://www.smart-hmi.com

[11] Rapid SCADA [online]. [cit. 2023-01-20]. Dostupné z: https://rapidscada.org

[12] VTScada: by Trihedral [online]. [cit. 2023-01-20]. Dostupné z: https://www.vtscada.com

[13] DINITA, Madalina. 4 Best Dynamic Website Creator Software [2023 Guide]. Windowsreport [online]. 2021 [cit. 2023-01-21]. Dostupné z: https://windowsreport.com/software-create-dynamic-websites/

[14] Visual Studio Code [online]. [cit. 2023-04-25]. Dostupné z: https://code.visualstudio.com

[15] Live Server. Visual Studio Marketplace [online]. [cit. 2023-04-25]. Dostupné z: https://marketplace.visualstudio.com/items?itemName=ritwickdey.LiveServer

[16] W3.CSS Intro. W3Schools [online]. [cit. 2023-05-13]. Dostupné z: https://www.w3schools.com/w3css/w3css_intro.asp