Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Diplomová práce

Systém pro správu LCD informační tabule

Bc. David Kabelka

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. David Kabelka

Informatika

Název práce

Systém pro správu LCD informační tabule

Název anglicky

System for administration and management of LCD information board

---

Cíle práce

Cílem diplomové práce je navrhnout a vytvořit systém pro správu obsahu obrazovek, který bude zobrazen na více informačních tabulích. Uživatel bude schopen měnit obsah na displejích základě změn konfigurace v administraci, která bude nastavení ukládat na server. LCD obrazovku bude obsluhovat jednodeskový počítač, který bude schopen základního ovládání a zobrazování požadovaného obsahu.

Metodika

Nejdříve bude provedena analýza systému a definovány požadavky na informační systém. Dále bude proveden jeho návrh, v jehož rámci bude proveden výběr jednodeskového počítače. Dále bude použita vhodná LCD obrazovka, serverové řešení, databáze a další vhodné technologie. Bude změřena energetická náročnost s návrhy na možnou úsporu. Bude popsána konkurence na trhu s podobným řešením. Dále bude vytvořen softwarový prototyp informačního systému a prototyp samotné informační LCD tabule. Pro jednodeskový počítač bude za použití softwaru pro 3D modelování navrženo šasi, které bude vytištěno na 3D tiskárně.

Doporučený rozsah práce

50-60 stran

Klíčová slova

LCD, mikropočítače, informační systémy, správa obsahu obrazovek

---

Doporučené zdroje informací

BRUCKNER, Tomáš. *Tvorba informačních systémů : principy, metodiky, architektury.* Praha: Grada, 2012. ISBN 978-80-247-4153-6.

CASCIARO, Mario.; MAMMINO, Luciano. *Node.js Design Patterns – Second Edition. [elektronický zdroj] /.* Birmingham: Packt Publishing, Limited, 2016. ISBN 9781785887383.

KULA, Piotr J. *Raspberry pi server essentials.* Birmingham, England: Packt Publishing, 2014. ISBN 978-1-78328-469-6.

ROBINSON, Andrew; COOK, Mike. *Raspberry Pi projects :.* Chichester, England: Wiley, 2014. ISBN 978-1-118-55543-9.

SARRION, E. JavaScript from Frontend to Backend Learn Full Stack JavaScript Development Using the MEVN Stack with Quick and Easy Steps, 2022. 9781801074148

Předběžný termín obhajoby

2023/24 LS – PEF

Vedoucí práce

Ing. Marek Pícka, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 28. 11. 2023

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 9. 2. 2024

doc. Ing. Tomáš Šubrt, Ph.D.

Děkan

V Praze dne 31. 03. 2024

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Systém pro správu LCD informační tabule" jsem vypracoval(a) samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 31.3.2024 _____

Poděkování

Rád bych touto cestou poděkoval panu Ing. Marku Píckovi, Ph.D. za odborné a příkladné vedení mé diplomové práce, především za skvělou komunikaci, ochotu, individuální přístup, odbornost a cenné rady, též děkuji své přítelkyni za oporu a trpělivost.

Systém pro správu LCD informační tabule

Abstrakt

Diplomová práce se zaměřuje na návrh a vytvoření prototypu informačního systému pro správu koncových zařízení a obsahu, který zobrazují. Tento systém je provozován za pomocí cloudového řešení. Uživatel má možnost měnit obsah na displejích prostřednictvím webové administrace, která ukládá data do databázového systému, kde je konfigurace uložena. K obsluze LCD obrazovek je využit jednodeskový počítač schopný komunikace s vytvořeným systémem, základní obsluhy připojené LCD obrazovky a zobrazování požadovaného obsahu.

Teoretická část se věnuje východiskům v oblasti informačních systémů a technologií, které slouží k vývoji informačních systémů – programovacímu jazyku Java, frameworkům, databázovým technologiím, cloudovým službám, dále standardům uchycování LCD obrazovek a 3D tisku.

Praktická část se věnuje návrhu informačního systému a následně jeho vytvoření pomocí zvolených technologií. Popisuje postupy instalací a konfigurací softwarových a hardwarových komponent, programování frontend a backend části informačního systému, 3D modelování a tisk šasi.

Klíčová slova: LCD, mikropočítače, informační systémy, správa obsahu obrazovek, backend, frontend

System for administration and management of LCD information board

Abstract

The thesis focuses on design and creation of a prototype of an information system for content and end devices management, that display content. This system is operated using a cloud solution. The user has the ability to change content on the displays through web administration, which saves data to a database system where the configuration is stored. A single-board computer capable of communicating with the created system is used to operate the LCD screens, performing basic operations on connected LCD screens and displaying the desired content.

The theoretical part addresses the fundamentals in the field of information systems, technologies used for the development of information systems - the Java programming language, frameworks, database technologies, cloud services, as well as standards for mounting LCD screens and 3D printing.

The practical part focuses on the design of the information system and subsequently its creation using selected technologies. It describes procedures for installing and configuring software and hardware components, programming the frontend and backend parts of the information system, and 3D modelling and printing of the chassis.

Keywords: LCD, microcomputers, information system, content management, backend, frontend

Obsah

1 Seznam obrázků, tabulek, grafů a zkratek

1.1 Seznam obrázků

1.2 Seznam tabulek

1.3 Seznam použitých zkratek

OS – Operační systém
LCD – Liquid Crystal Display
3D – three-dimensional
VESA – Video Electronics Standards Association
VPS – Virtual Private Server
AWS – Amazon Web Services
GCP – Google Cloud Platform
IoT – Internet of Things
W – Watt
JDK – Java Development Kit
API – Aplication Programming Interface
JRE – Java Runtime Environment
SDK – Software development kit
JVM – Java Virtual Machine
SCM – Supply Chain Management
APS – Advanced Planning and Scheduling
MIS – Management Information System
HRM – Human Resource Management
EAM – Enterprise Asset Management
DMS – Document Management System
BPM – Business Process Management
BI – Business Intelligence
CMS – Content Management System
ERP – Enterprise Resource Planning

CRM – Customer Relationship Management

MAC – Media access control

JPA – Jakarta Persistence API

UI – User interface

DNS – Domain name server

FDMI – Flat Display Mounting Interface

VESA – Video Electronics Standards Association

SSH – Secure shell

CAD – Computer-aided design

2 Úvod

Přístup k informacím je jednou z nejvíce vystihujících skutečností dnešní doby, informací je stále více a přístup k nim se každým rokem zrychluje. S nabývajícím množstvím dat a informací je ale třeba je nějakým způsobem uchovávat, zpracovávat, analyzovat. S tím nám pomáhají různé druhy informačních systémů. Každý podnik v dnešní době používá nějaký informační systém, ať už jde o formu eGovernmentu a komunikace se státními institucemi nebo o používání internetového bankovnictví. Pokud ale jde o střední a velké podniky, ty se neobejdou bez jisté formy podnikového systému, v době globalizace již nelze vše uchovávat v hlavě nebo na papíře.

Tato diplomová práce se zabývá aspekty tvoření nového informačního systému, který pomůže středním a větším podnikům se sdílením potřebných informací a metrik se svými zaměstnanci. Cílí hlavně na podniky, pro které je důležitá skladová efektivita. Firmy mají potřebu zpracovaná výkonnostní data některým způsobem předat zaměstnancům – skladovým operátorům a jejich vedoucím v reálném čase. Pro tyto účely analytické BI oddělení většinou tvoří přehledné dashboardy s daty. Tyto dashboardy jsou ale často otevírány zaměstnanci v telefonech nebo jsou použity počítače. V lepších případech jsou použity LCD tabule, ale jejich vzdálená správa bývá složitá a často trvá několik hodin až dnů, než se požadavek na změnu zobrazovaných dat zpracuje.

Proto je podstata diplomové práce věnována vytvoření informačního systému na vzdálenou správu informačních tabulí, kde bude moci kompetentní koncový uživatel mít možnost měnit zobrazovaná data a zjišťovat stav jednotlivých tabulí.

3 Cíl práce a metodika

3.1 Cíl práce

Cílem diplomové práce je navrhnout a vytvořit prototyp informačního systému pro vzdálenou správu informačních tabulí. Jednotlivé tabule budou schopny zobrazovat uživatelem stanovený webový obsah. Informační systém bude mít uživatelské webové rozhraní, které bude přístupné na webové adrese. IS bude schopen zobrazit seznam používaných obrazovek, jejich stav. Uživatel bude mít možnost měnit konfiguraci každé obrazovky, zejména orientaci displeje, čas úsporného režimu a požadovaný obsah k zobrazení. LCD obrazovku bude obsluhovat vybraný jednodeskový počítač, který bude schopen stahovat konfiguraci z informačního systému a zobrazování požadovaného obsahu, bude také reportovat svůj stav zpět do IS.

3.2 Metodika

Podstata práce bude primárně založena na internetových podkladech, zejména se bude jednat o dokumentace jednotlivých výrobců použitých zařízení, produktové údaje poskytnuté vývojáři softwarových systémů, technologií a řešení, dokumentace aplikovaných softwarových knihoven, elektrotechnická fóra se zaměřením na jednodeskové počítače, serverové služby a softwarový vývoj. Dále na využití autorových zkušeností, znalostí a schopností v oblasti informačních systémů, databází a hardwaru.

V teoretické části budou popsána teoretická východiska týkající se informačních systémů, popsány budou technologie, které jsou na projekt tohoto typu vhodné a potřebné. Přiblíženy budou serverové technologie, programovací jazyk, databázové systémy a použitelná serverová backend a frontend řešení.

Praktická část této diplomové práce bude věnována výběru náležité jednotky, která bude sloužit pro zobrazování potřebných obsahových dat. Důležitým kritériem bude přítomnost kompatibilního grafického výstupu, bezdrátové komunikační technologie Wi-Fi, také bude podstatná nízká spotřeba elektrické energie.

Dále v praktické části dojde k analýze a vypracování návrhu informačního systému, který bude následně naprogramován pomocí zvolených technologií. Popsány budou postupy, procesy instalací, zprovoznění technologií a nutné změny konfigurací jak v

software, tak hardware části práce. Také bude vyvinut zdrojový kód serverové aplikace, která bude zpracovávat data z uživatelského rozhraní a zároveň bude poskytovat data jednotce určené pro obsluhu LCD obrazovky. Pro jednodeskový počítač bude za použití softwaru pro 3D modelování navrženo šasi, které bude kompatibilní s VESA uchycovacím standardem pro stabilní a spolehlivé uchycení. Toto šasi bude vytištěno na 3D tiskárně.

4 Teoretická východiska

4.1 Informační systém

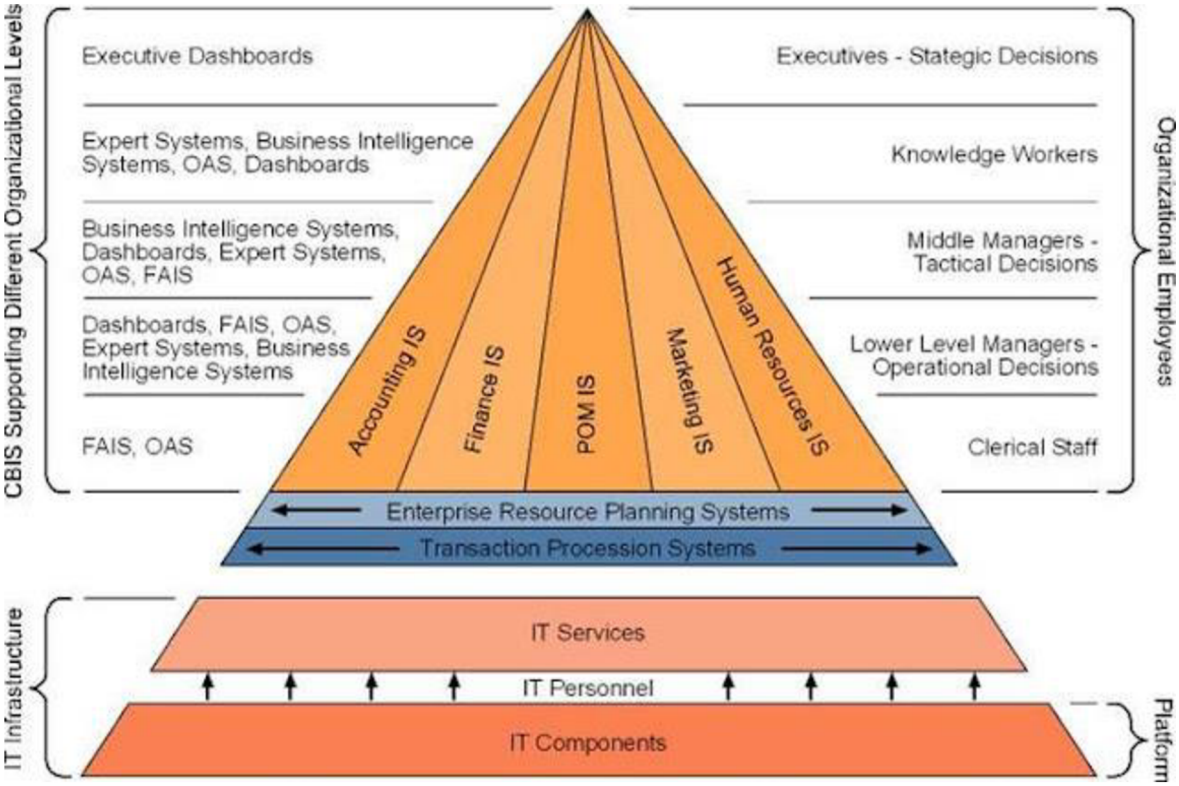
Informačním systémem se nerozumí jen software, ale je definován jako celek, který je složen z lidí, HW i SW prostředků, metod, organizačních opatření a který zajišťuje sběr, přenos, zpracování a uchování dat s cílem další prezentace pro uživatele, jež jsou součástí řídící struktury daného podniku. Díky IS je možné provést informované rozhodování a přispívá k efektivnějšímu řízení společnosti. [1] [2]

Každý podnik, ať jde o firmu, veřejnou instituci nebo jakoukoli jinou organizaci, je závislý na přístupu k informacím o dění ohledně vlastních zaměstnanců, finančních transakcích, pořízeném majetku nebo zákaznících. Původ těchto informací nebývá jeden a je tak nutné tato data shromáždit z různých zdrojů na jedno místo, což je jedna z hlavních funkcí informačního systému. Následně jsou tato data již v IS použita a zpracovaná do přehledů pro kompetentní osoby a dále slouží k plánování činností, nebo pro komunikaci se zákazníky. IS může pomoci s nalezením problémů v nastavených procesech ve firmě. [3]

Dříve bylo nasazení informačního systému do podniků spíše otázkou trendu digitalizace, díky čemuž bylo možné nahradit papírové fyzické záznamy elektronickou podobou a přejít na moderní způsob vedení a správy podniků. Dnes jde ale již o nezbytný krok i pro ty nejmenší podniky a živnostníky. Zavedení samotného informačního systému samo o sobě nedokáže zajistit perfektní fungování firmy, velmi důležitý je přístup zaměstnanců a jejich kontinuální školení v této oblasti. IS může mít ale i svá úskalí, vzhledem ke skutečnosti, že uživatelé IS pracují se stejnými daty najednou, může selhání v oblasti lidského faktoru ovlivnit celý podnik. [3]

Jedním ze základních použití informačních systémů jsou lidské zdroje, podnik tak může sledovat docházku, přehledy ohledně náborů, školení, výkonu nebo výdajů, dají se také plánovat směny a je k dispozici i přehled mzdových dat. Pomocí IS lze i sledovat finanční a ekonomické ukazatele firmy, nákupy, faktury, dodavatelé atd. Spravovat lze také majetek, plánovat v oblasti logistiky, pohybu zboží, výrobní plány nebo stavy skladů. Důležitá je také bezpečnost dat, IS umožňuje přehledné a jasné nastavení rolí zaměstnanců ve společnosti, ti tak mají přístup pouze k informacím, která jsou k jejich pozici adekvátní, lze tak efektivně distribuovat data napříč zaměstnaneckými úrovněmi. IS také pomůže při administraci v rámci e-shopu sestavit přehled prodávaných výrobků nebo služeb. S tím

souvisí také řízení marketingu a sledování chování zákazníků. IS rovněž přispívá ke sledování plnění cílů podniku, nastavené metriky tak lze sledovat i v reálném čase, strategické řízení je díky tomu lepší. [3]



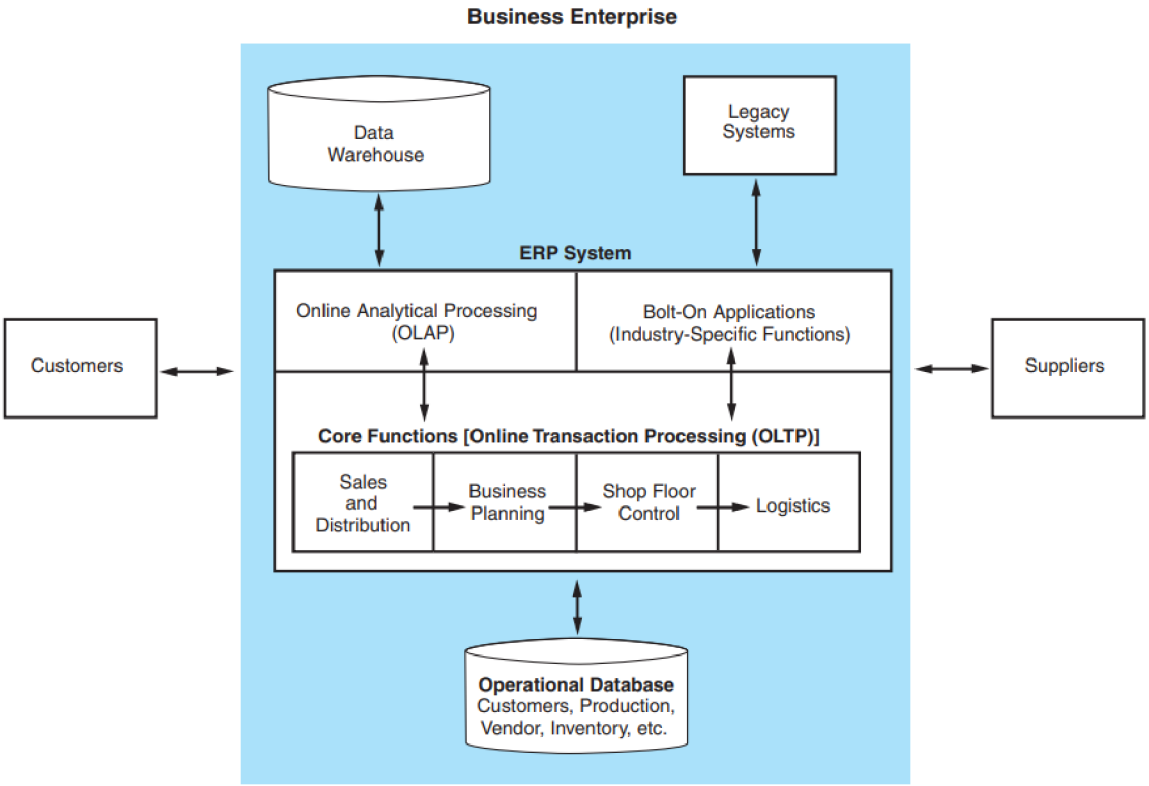
Obrázek 1: Architektura IS [4]

V oblasti typů informačních systémů se lze setkat s mnoho zkratkami, které popisují určení. Funkce jednotlivých systémů se mohou vzájemně do určité míry překrývat. Supply Chain Management, zkráceně SCM, se zabývá řízením dodavatelského řetězce. APS, což je Advanced Planning and Scheduling, slouží k pokročilému plánování a řízení dodavatelského řetězce a často je součástí ERP systému. HRM neboli Human Resource Management, se specializuje na správu lidských zdrojů, typicky spadá do kompetence personálního oddělení, kde pomáhá s vytvářením záznamů a analýzami náborů, školení a dalšího vzdělávání zaměstnanců. MIS, tj. Management Information System, slouží k operativnímu a taktickému rozhodování a řízení procesů v oblasti nákupu nebo prodeje a je nadstavbou ERP systému. EAM, Enterprise Asset Management, se stará o správu podnikových zdrojů a životního cyklu zařízení s cílem snížit náklady na údržbu a obnovu strojů. DMS, neboli Document Management System, centralizuje manipulaci s elektronickými dokumenty a jejich

obsahem. BPM, Business Process Management, slouží k znázornění a realizaci podnikových procesů v kontextu okolí, usiluje o optimalizaci procesů a efektivní časové plánování, což přispívá k vyšší konkurenceschopnosti podniku. BI, zkratka pro Business Intelligence, se zaměřuje na statistické a analytické výpočty, práci s big daty a propojení různých datových zdrojů. [3]

4.1.1 ERP

Enterprise Resource Planning jsou nejčastěji používanými informačními systémy v podnikové sféře, tento pojem se často zaměňuje za termín celkové kategorie – Informační systémy. Hlavním cílem a využitím ERP informačních systémů, jak již z anglického názvu vyplývá, je plánování podnikových zdrojů. Díky implementaci tohoto druhu IS lze v podnicích standardizovat procesy a nacházet „best practises“ – objektivně nejlepší a nejefektivnější způsob, jak danou činnost, proces nebo proceduru vykonávat. [3]



Obrázek 2: Diagram ERP systému [5]

Zavedení ERP systému je ideální i pro situaci, kdy má podnik více poboček, zajistí se tak, aby všechny pobočky používaly stejné, konzistentní a funkční metody, to zajistí jednotnost a úsporu nákladů. Zavedené procesy jsou také více transparentní a je nad nimi

větší kontrola. ERP systémy se často vytvářejí na míru konkrétnímu klientovi na základě jeho specifických potřeb. ERP v sobě často integrují ostatní druhy IS např. CRM, SCM a MIS. [3]

Existují ale i větší dodavatelé informačních systémů, kteří nabízí ERP, ve kterých lze používat jednotlivé moduly. Tyto moduly jsou často univerzální a fungováním typicky společné pro většinu podniků. Nasazení tak bývá rychlejší. [6] Do těchto ERP lze následně doprogramovat moduly se specifickými účely na míru. [7]

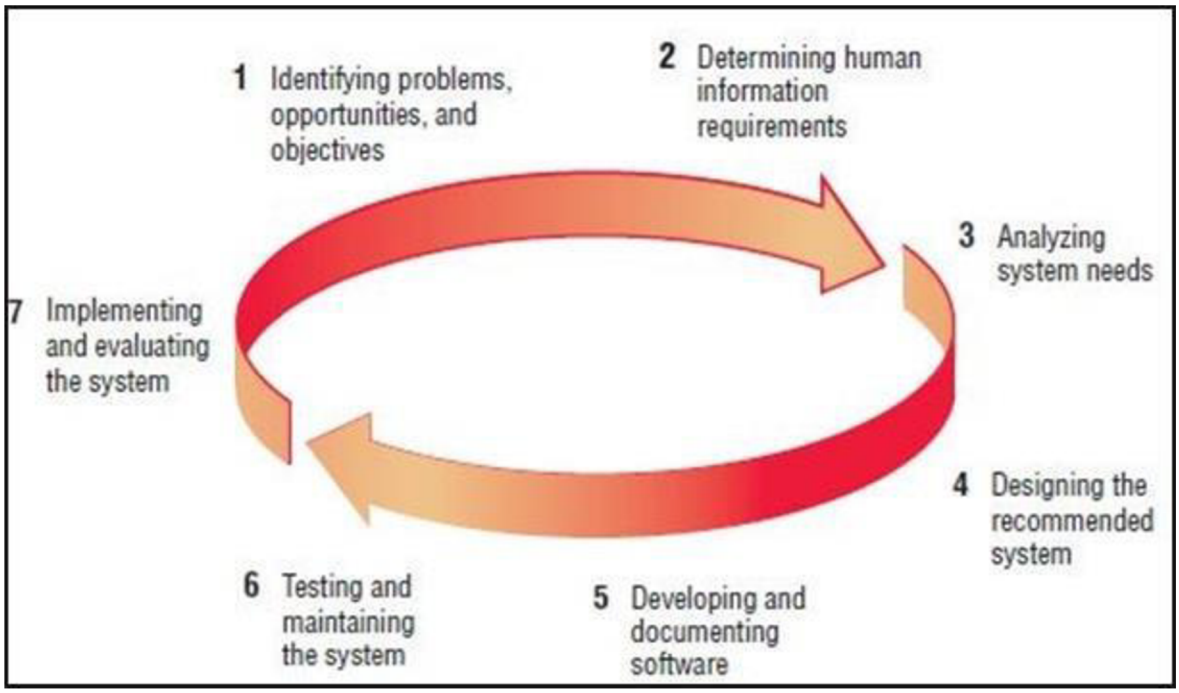
4.1.2 CRM

Customer Relationship Managemen systémy, zkráceně CRM, představují formu informačního systému zaměřenou na správu komunikace se zákazníky. Tyto systémy jsou klíčové pro firmy s rozsáhlejší sítí klientů a slouží k optimalizaci interakcí se zákazníky. Trendem poslední doby je mobilní CRM, který se prezentuje formou mobilní aplikace. CRM systémy fungují jako úložiště všech relevantních informací o zákaznících a jejich interakcích, což znamená například nákupy, prohlížené položky v e-shopu apod. Analýzou těchto dat, někdy podpořenou některou verzí umělé inteligence, lidé v řídících pozicích získávají přehled o prodejích a lépe predikují potřeby zákazníků. CRM systémy umožňují lepší organizaci skladových zásob a nastavení cenové politiky. Plně využitý informační systém CRM je nejen prostředkem k porozumění potřebám zákazníků, ale také k udržení stávajících a získání nových. Díky komplexním informacím získávají zákazníci dojem, že firma či e-shop pečuje o jejich potřeby, což zvyšuje pravděpodobnost opětovného využití služeb. [3]

4.2 Životní cyklus IS

Pod pojmem životní cyklus informačního systému si můžeme představit celkový souhrn procesů, který udává pravidla a směr návrhu, vývoje, implementace, provozu i údržby IS. Pevně stanovený cyklus IS umožňuje podnikům systematický a tím pádem efektivní přístup k jednotlivým etapám tvorby IS. [8]

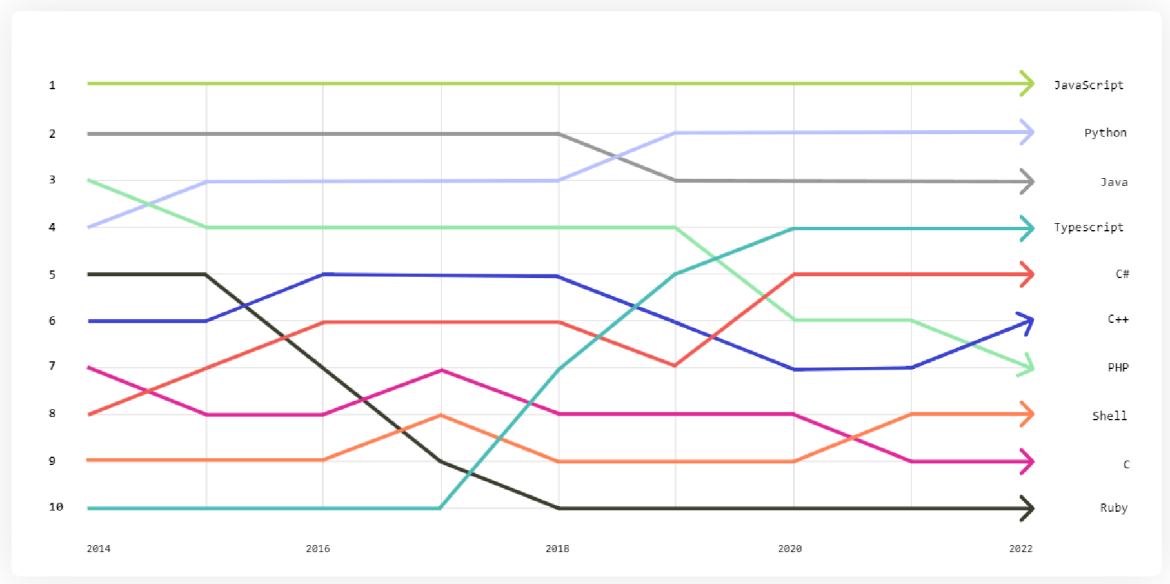
Různí autoři definují etapy návrhu informačního systému odlišným způsobem a mají na proces jiné pohledy. Dle Kendalla se návrh skládá z identifikace problémů a cílů, definování informačních potřeb, analýzy systémových potřeb, návrhu systému, vývoje softwaru, testování a údržby, zavedení systému a následné evaluace. [9]



Obrázek 3: Životní cyklus vývoje IS [9]

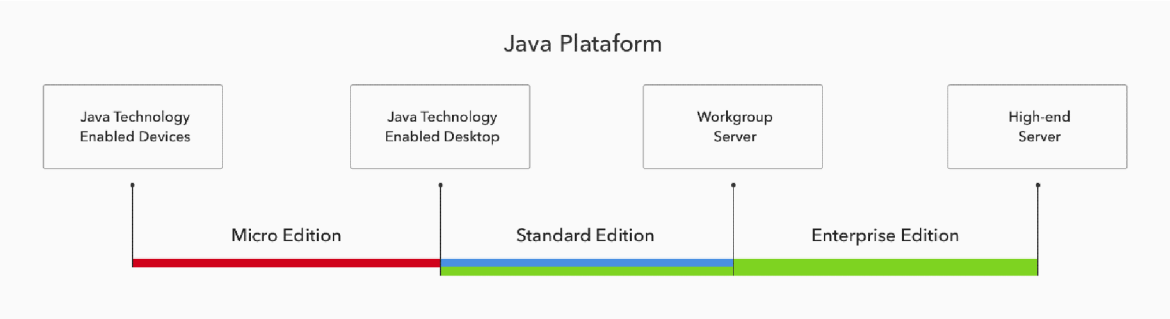
4.3 Programovací jazyk Java

Tento objektově orientovaný vysokoúrovňový programovací jazyk je v dnešní době jedním z nejrozšířenějším a nejoblíbenějším. Jeho historie sahá do 90. let minulého století, kdy její přijetí a následný vzestup byl velmi rychlý. Jedná se o interpretovaný jazyk, což znamená, že se místo klasického strojového kódu, se kterým se můžeme setkat u nízko úrovňových jazyků, generuje byte kód. Díky byte kódu je možné vytvořený program nebo aplikace provozovat na kterékoli jiné platformě nebo zařízení bez nutnosti úprav ve zdrojovém kódu. Služba GitHub, která poskytuje nástroje pro verzování a online repositář pro vyvíjený software, nabízí k dispozici statistiku věnující se popularitě používaných programovacích jazyků v čase, a to až do roku 2014. Na Obr. 3 můžeme postupný vývoj vidět. [10] [11]



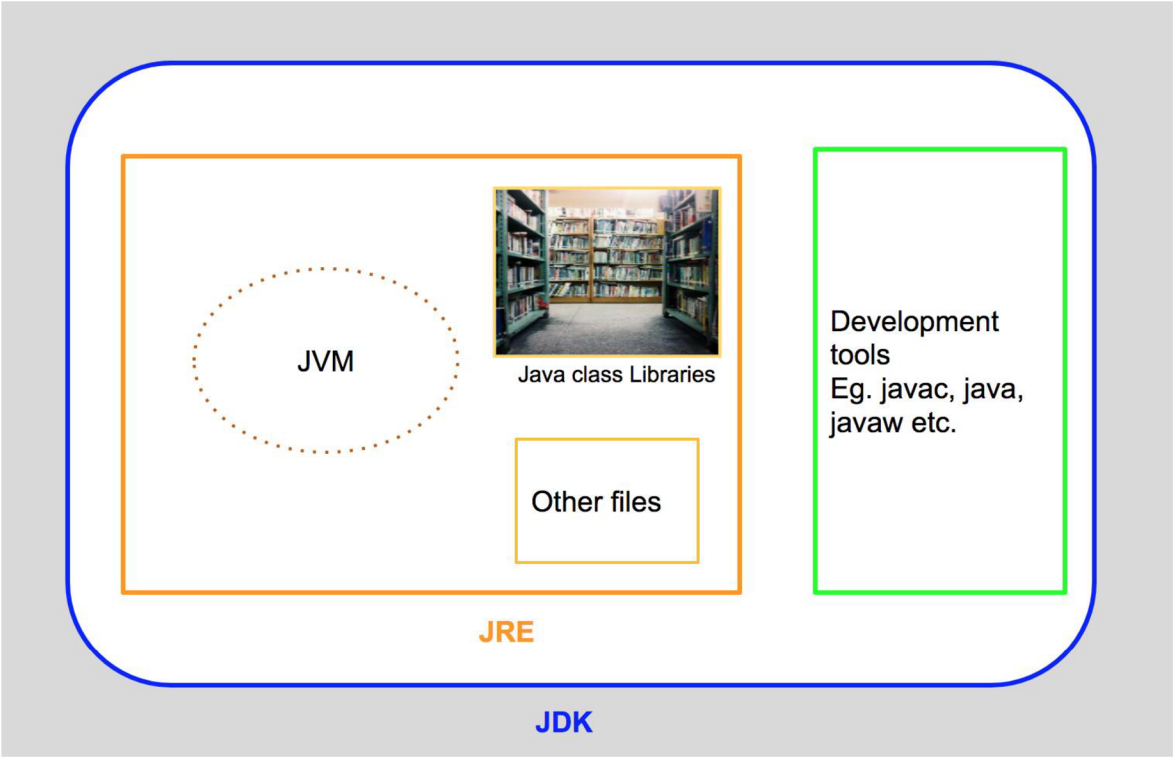
Obrázek 4: Vývoj popularity programovacích jazyků [11]

Java je k dispozici v několika verzích, které jsou navzájem odlišné a nabízí různé funkce. Každá verze je navržena na jiný způsob cílového využití, výběr závisí na účelu vyvíjeného programu. Základní jsou 3 verze, Java Standard Edition, Java Enterprise Edition a Java Micro Edition. [12]



Obrázek 5: Překryv verzí Javy [12]

Základní komponentou je Java Development Kit, zkráceně JDK, jenž je celkovým balíčkem, který je nutný pro vývoj software v programovacím jazyce Java a obecně spadá do SDK, což znamená Software Development Kit. JDK obsahuje JRE – Java Runtime Environment, která umožňuje běh aplikací nad používaným operačním systémem. JRE obsahuje knihovny tříd a další zdroje, které potřebuje Java k běhu. [10] [12]



Obrázek 6: JDK Diagram [13]

4.3.1 Java Standard Edition

Zkráceně se Java Standard Edition nazývá Java SE a je základní edicí a platformou, která obsahuje množství použitelných API, zahrnuje specifikaci jazyka Java a informace pro práci s Java Virtual Machine. Umožňuje použití a práci se soubory a souborovým systémem, sítěmi, umožňuje vytvářet uživatelské prostředí atd. Od verze 7 je referenční verzí Javy SE OpenJDK. Aktuální verze je již 22 vydaná edice. [12] [14]

4.3.2 Java Enterprise Edition

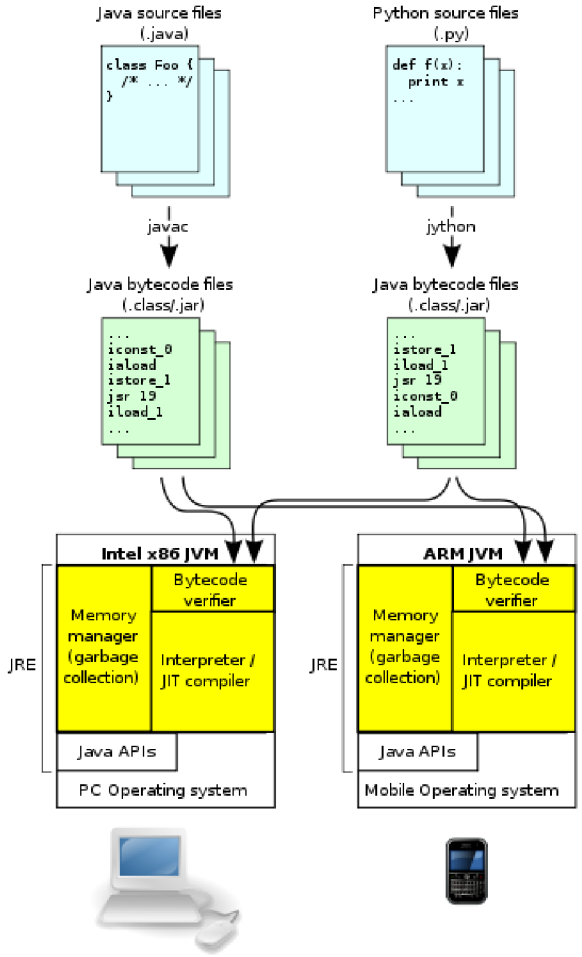
Tato edice se používá hlavně v podnikové sféře a umožňuje lepší vývoj informačních systémů, určených pro běh na serverovém prostředí. Je to nadstavba Java SE, která rozšiřuje některé funkce a přidává další hlavně v oblasti spolehlivosti, škálovatelnosti a bezpečnosti. Nevýhodou jsou ale vyšší nároky na investice, provoz a také znalosti programátorů, kvůli velké míře různých specifikací. [15]

4.3.3 Java Micro Edition

Tato edice byla vytvořena pro pokrytí potřeb vývoje pro aplikace, které jsou provozovány na integrovaných a mobilních zařízeních. Jedná se o mnohem méně populární verzi Javy než Java SE a Java EE. Kvůli nástupu Android a iOS zařízení zažila Java ME podstatný pokles, v posledních letech se tato verze používá v IoT odvětví a to hlavně díky svým nízkým nárokům na výpočetní výkon a také nabízí síťové funkcionality. [12] [16]

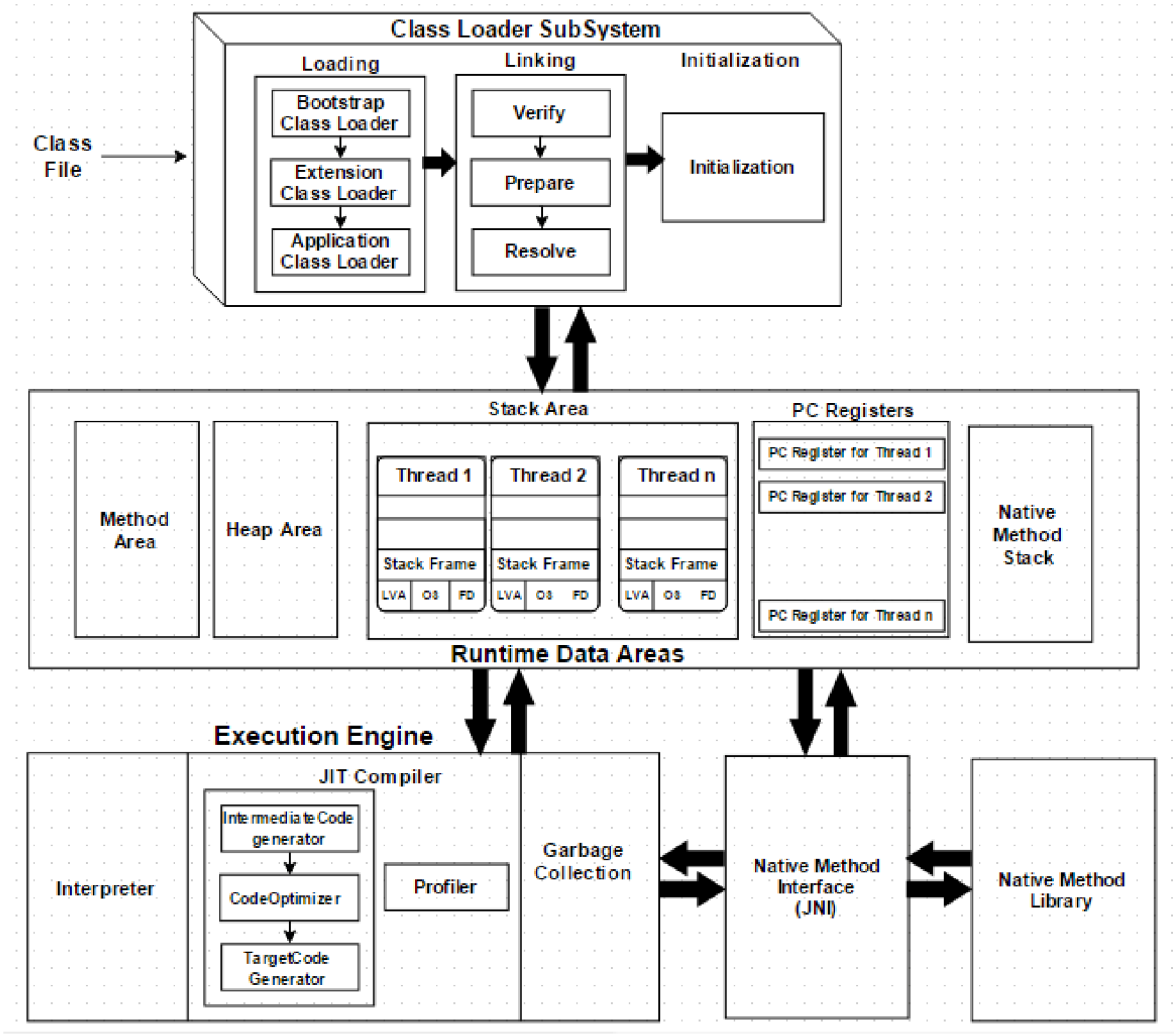
4.3.4 Java Virtual Machine

Java Virtual Machine, zkratkou také jako JVM je jednou z částí JDK, která je virtuálním zařízením a prostředím, které umožňuje počítači spouštět programy napsané v Jazyce Java, ale také i programy, které byly napsané v dalších programovacích jazycích, dokud jsou zkompilovány do Java Byte kódu. [17]



Obrázek 7: JVM architektura

JVM je rozšířeno o specifikaci, která udává, co vše je potřeba k implementaci JVM. Díky specifikaci je zajištěna kompatibilita Java programů mezi více platformami. JVM nabízí také automatickou práci výjimek, což v případě chyb a pádů programů umožňuje jednodušší nalezení příčiny problémů a následně jejich jednodušší řešení a odstraňování. JVM má několik základních komponent. Class Loader Subsystem je zodpovědný za načtení, přidání a inicializaci souboru s Java třídou. Dále komponenta Runtime Data Areas, která obsahuje metody, PC registry, zásobníky a vlákna. Poslední komponentou je Execution Engine, jejíž součástí je interpreter, kompiler a garbage collector. [17] [18]



Obrázek 8: Diagram JVM komponent [17]

4.4 Spring Framework

Spring Framework je populární nástroj pro vývoj robustních aplikací v programovacím jazyce Java. Spring je modulární, to znamená, že vývojář má k dispozici mnoho komponent, které může při vytváření aplikací využít. Použití tohoto frameworku má nevýhodu v značně náročné konfigurační režii prostřednictvím XML souborů nebo přímo v jazyce Java. Široké možnosti konfigurace dávají vývojáři velkou flexibilitu a počáteční fáze projektů bývají pomalé. [19]

4.4.1 Spring Boot

Spring Boot je nadstavba samotného Spring frameworku, která výrazně ulehčuje práci vývojáře díky eliminaci rozsáhlé konfigurace. Pomocí funkce Springboot starter, který poskytuje sestavení startovacího projektu do jednoho celku, je počáteční fáze vývoje aplikace značně jednodušší a efektivnější. Starter také podporuje mnoho různých doplňků a závislostí, například připojení databáze, práce s API, možnosti zabezpečení atd., ty se přidávají jednoduchým označením konkrétní závislosti do souboru „pom.xml“ v případě použití Mavenu, případně „build.gradle“ při použití Gradlu, to jsou nástroje pro řízení závislostí v Javě a také zajišťují sestavování. [19]

4.5 Vaadin framework

Vaadin framework je projekt s otevřeným zdrojovým kódem, který umožňuje programovat jak frontend, tak backend logiku a procesy v jednom prostředí v programovacím jazyce Java umožňuje vývojáři výrazné zlehčení při vývoji uživatelských rozhraní. Vaadin využívá modelu server-side programování, tím pádem jsou všechny prvky a moduly uživatelského rozhraní definovány a dále upravovány na serverové straně a poté jsou vykresleny v prohlížeči, kde s rozhraním pak interaguje koncový uživatel. Jednotlivé moduly jsou modifikovatelné, je tedy možné přizpůsobení na míru konkrétnímu projektu a záměru. Vaadin Framework se snaží využívat poslední standardy HTML5 a CSS a zajistit kompatibilitu napříč prohlížeči. Podporuje také responsivní prostředí. Jeho verze Core je k použití zdarma. Nabízí také placenou verzi Pro. Dává na výběr velké množství komponent, z kterých lze UI sestavovat. Jedná se o jednoduché komponenty jako tlačítka, text boxy, tabulky, formuláře a grafy, ale nabízí také komponenty na výběr data a času nebo složitějším

tabulkám podobné moduly. Vaadin nabízí jednoduchý proces mapování datových modelů na využité UI komponenty, to značně ulehčuje vypisování a aktualizaci dat v uživatelském rozhraní. Vaadin také nabízí podporu integrace se Spring bootem, pro již zavedené fungující aplikace tak lze jednoduše vytvořit uživatelské rozhraní. Velkou výhodou je podpora add-onů, ty mohou být tvořeny komunitou a nabízeny v repositáři, odkud je lze stahovat a dále využívat. [20] [21]

4.6 Databáze

4.6.1 SQL

Databáze jsou uspořádané systémy dat, které existují v různých formách, přičemž elektronická forma je často preferovaná pro svou efektivitu v úpravách, aktualizacích a vyhledávání informací. Jejich historie sahá až do 50. let minulého století, kdy se začalo usilovat o digitalizované zpracování dat a vznikaly první programy pro hromadné zpracování dat. [22]

Relační databáze, které vznikly v 70. letech minulého století, se staly základem pro většinu aplikací pracujících s daty. Základem relačních databází jsou tabulky, které mohou být vzájemně propojeny. Standardizovaný jazyk SQL je klíčový nástroj pro manipulaci s daty v relačních databázích, vznikl v 70. letech, prošel řadou revizí a dnes je podporován různými typy databázových systémů. SQL databáze nabízí výhody jako univerzálnost, jednoduchost, výkonnost a flexibilitu, které přispívají k širokému využití a oblíbenosti mezi vývojáři aplikací. [22]

S nárůstem komplexity dat a potřeb efektivnějšího zpracování se v současném vývoji databázových systémů objevuje zájem o objektově orientované databáze jako možnou alternativu k relačním databázím. Tyto databáze nabízejí nový způsob ukládání a manipulace s daty, který lépe odpovídá složitým a dynamickým strukturám moderních informačních systémů. [23]

SQL má i jisté nevýhody, jako je omezení na relační data, nutnost detailní znalosti struktury databáze a možné bezpečnostní problémy spojené s neopatrným používáním. Je tedy důležité brát tyto aspekty v úvahu při práci s SQL a databázemi obecně. [22]

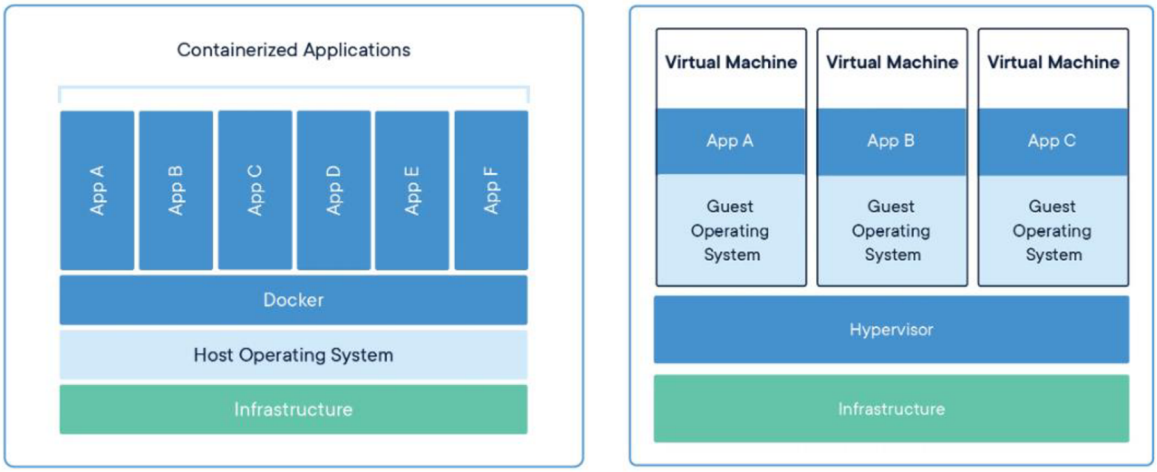
4.6.2 NoSQL

NoSQL databáze, zkráceně "Not Only SQL", nabízejí alternativní přístup k ukládání dat v porovnání s tradičními relačními SQL databázemi. Jejich význam vzrostl zejména díky schopnosti zpracovávat velké objemy nestrukturovaných či částečně strukturovaných dat. NoSQL databáze přinášejí výhody jako jednodušší škálovatelnost, flexibilitu a zlepšený výkon zpracovávání dat. Vlastnosti NoSQL databází zahrnují flexibilní databázová schémata, škálovatelnost, výkon a flexibilitu při vývoji aplikací. Tyto databáze podporují různé datové modely, včetně dokumentových, klíč-hodnota, sloupcových a grafových. NoSQL databáze jsou využívány například pro odhalování podvodů, aplikace pro internet věcí (IoT), hry nebo chatování. Mezi nevýhody NoSQL databází patří zvýšená složitost modelování dat, kompromisy v oblasti konzistence dat, nedostatek vyspělých nástrojů a obtížnost migrace a integrace dat. [24]

4.7 Docker

Docker je populární open source nástroj používaný při vývoji, testování i produkčním nasazení serverových aplikací. Tato platforma využívá technologie kontejnerizace. Kontejnerizace umožňuje vývojářům softwaru zapouzdřit aplikaci a všechny její závislosti do jednoho standardizovaného celku, který se nazývá kontejner. Tyto kontejnery lze poté provozovat na kterémkoli dalším prostředí, bez nutnosti složité konfigurace všech částí aplikace. [25]

Na rozdíl od klasické virtualizace, kde je emulován celý operační systém v každé instanci virtuálního počítače, nabízí kontejnerizace sdílení kernelu jednoho hostitelského OS, nad kterým pomocí Docker Enginu kontejnery běží. Tyto kontejnery jsou plně izolované, zároveň se vzájemně neovlivňují, běží paralelně vedle sebe a každý se svojí konfigurací a daty. V případě virtualizace má každá instance virtuálního počítače velké nároky na úložný prostor – v rámci gigabajtů, kontejnery si nárokují většinou desítky až stovky megabajtů paměti, což z nich dělá mnohem menší, rychlejší a zároveň efektivnější způsob jakým serverové aplikace provozovat. [25]

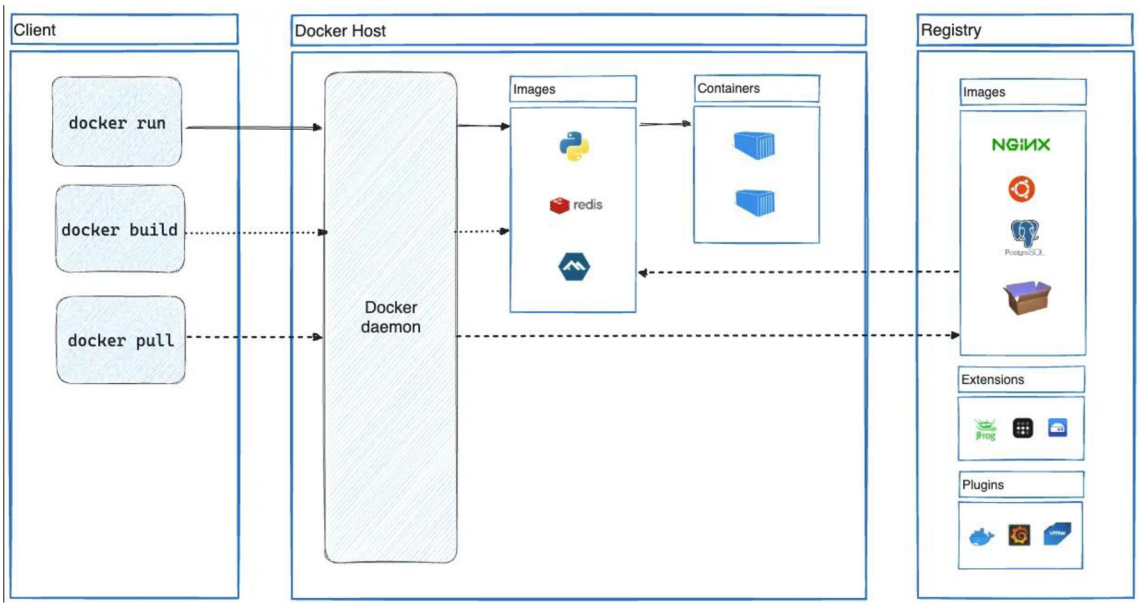


Obrázek 9: Diagram srovnávající Docker kontejnery a virtualizaci. [26]

Docker se skládá z několika hlavních komponent [27]:

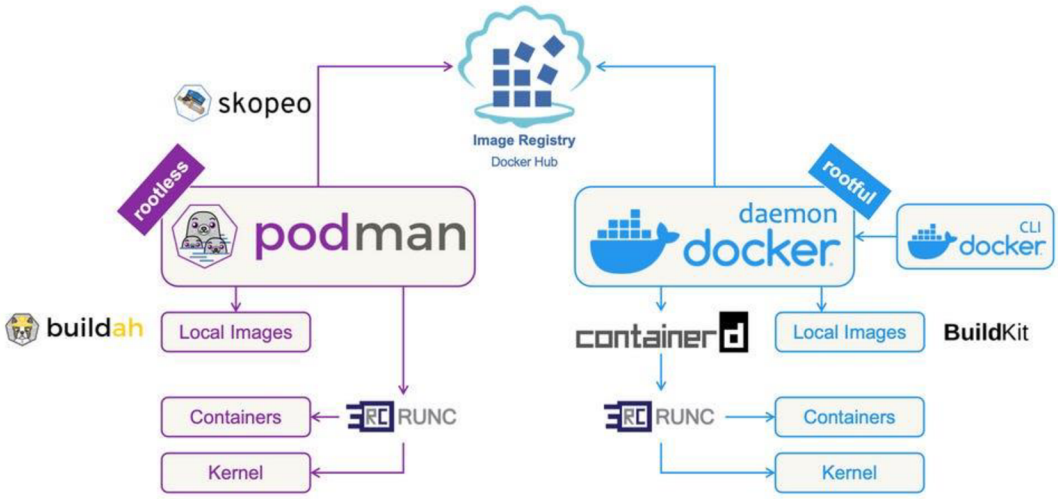
- Docker Daemon, který se také často nazývá Docker Engine je služba, která běží na pozadí operačního systému a stará se o životní cyklus každého kontejneru, čímž je prvotní konstrukci, běh a ukončení.
- Docker Client je rozhraním ve formě příkazového řádku, které umožňuje koncovému uživateli komunikovat s Docker Daemonem a interpretuje všechny zadané instrukce.
- Docker Image je šablona nebo obraz, který v sobě obsahuje všechny potřebné komponenty k běhu každé aplikace, zejména operační systém, závislosti a aplikační kód.
- Dockerfile je textový soubor, který popisuje konfigurační instrukce pro práci s Docker image, tato kombinace pak umožňuje vznik a běh kontejneru.
- Docker Registry např. Docker Hub je služba, která uchovává a umožňuje distribuci často používaných, veřejných a předpřipravených Docker Images. Tyto služby lze také použít pro uložení vlastních upravených obrazů.
- Docker Network umožňuje jednotlivým kontejnerům mezi sebou komunikovat a zároveň slouží pro komunikaci s internetem. Jelikož jsou kontejnery ve výchozím stavu plně izolované, nebylo by bez Docker sítě možné s aplikací v kontejneru nikterak interagovat, včetně jejího použité koncovým uživatelem.

- Docker Compose je pokročilá metoda sloužící k vytváření více kontejnerové aplikace za použití jediného YAML konfiguračního souboru. V tomto souboru uživatel nadefinuje všechny potřebné části aplikace, včetně jednotlivých kontejnerů nutných k běhu, datových adresářů pro uložení dat a síťové konfigurace.



Tabulka 1: Architektura Dockeru [28]

Zajímavou alternativou Dockeru je Podman. Jde o nástroj pro práci s kontejnery, který ale nepoužívá Daemon, nýbrž vše ovládá klient a výhodou je odstranění možného bodu selhání právě Daemona. [29]



Obrázek 10: Podman VS Docker [30]

Samotný Docker může být nejvíce užitečný právě pro vývoj aplikací, pokud jde ale o produkční prostředí, je Docker často doplněn o nějaký orchestrační systém. Přímo Docker má k dispozici Docker Swarm, avšak nejpoužívanější je Kubernetes. Tyto nástroje umožňují automatické škálování podle aktuálních nároků – např. počet uživatelů přistupujících k aplikaci nebo náročný výpočet. Orchestrační systém pak může sám potřebnou aplikaci spustit v několika instancích a zatížení distribuovat podle potřeby tak, aby nebyla ani jedna aplikace příliš přetížená, nebo naopak podprůměrně utilizovaná. Tyto nástroje jsou ideální volbou pro horizontální škálování. [31]

4.8 Cloud

Jedná se o model vývoje a používání technologií a dat, jež jsou uložena v internetu a přístupná vzdáleně pomocí internetového připojení. Do cloudových služeb lze začlenit email, úložiště dat nebo VPS server. Používání cloudu má celou řadu výhod, ale také nevýhod. Rovněž existuje více typů cloudů a jejich členění. Základní 3 typy jsou Infrastructure as a Service, Platform as a Service a Software as a Service. [32]

	On-premises	**IaaS**	**PaaS**	**SaaS**
Aplikace	NE	NE	NE	ANO
Runtime	NE	NE	ANO	ANO
Middleware	NE	NE	ANO	ANO
Operační systém	NE	NE	ANO	ANO
Virtualizace	NE	ANO	ANO	ANO
Úložiště	NE	ANO	ANO	ANO
Připojení	NE	ANO	ANO	ANO
Server	NE	ANO	ANO	ANO
Datacentrum	NE	ANO	ANO	ANO

Tabulka 2: Přehled zodpovědnosti poskytovatele služby [33]

4.8.1 Infrastructure as a Service

Při použití cloudového modelu typu infrastruktura jako služba, známá také anglickou zkratkou IaaS, dodavatel poskytuje a zaštiťuje servery, virtualizaci, úložiště, sítě atd. Všechny ostatní oblasti, jako je instalace a konfigurace OS, přísun dat, instalace a provoz aplikací, jsou zodpovědnost klienta, jakožto zákazníka cloudového poskytovatele. Zákazník platí za systémové prostředky, které využije. Těmito prostředky jsou CPU cykly, operační paměť, využitý diskový prostor apod. Příkladem poskytovatelů jsou AWS, GCP, Microsoft Azure, ale také Alibaba Cloud a Oracle Cloud Infrastructure. Pomocí této cesty lze do cloudu přesunout veškeré provozované systémy z fyzických serverů do cloudového řešení. IaaS lze také použít jako záložní řešení v případě selhání podnikové infrastruktury provozované konkrétní společností. [34] [35]

4.8.2 Platform as a Service

Platforma jako služba s anglickou zkratkou PaaS umožňuje zákazníkovi provozovat, vyvíjet a testovat aplikace, aniž by se musel starat o cokoli ostatní. Dodavatel nebo provozovatel takovýchto platforem se stará o veškerou konfiguraci operačních systému, jejich aktualizace, nároky na úložiště a o veškerý hardware. Zákazník platí za využité systémové prostředky a příklady jsou Google App Engine a SAP Cloud. [34] [35]

4.8.3 Software as a Service

V případě modelu software jako služba, též známo pod anglickou zkratkou SaaS, je zákazníkovi umožněno využívání softwarové aplikace, kterou plně provozuje její poskytovatel. Platební model je většinou realizován pomocí fixní tarifní částky fakturované měsíčně nebo ročně za licenci, která je často za jednoho nebo více uživatelů. Tento typ cloudových služeb je velice lákavý hlavně pro menší podniky, které nemají velké IT oddělení nebo zázemí na provoz vlastních služeb. Je ale populární i mezi velkými firmami a to díky jednoduché flexibilitě v případě náhlého růstu nebo úpadku. [34] [35]

4.8.4 Public cloud

Veřejný cloud je nejvíce rozšířeným použitím cloudových služeb. Je přístupný skrze veřejný a volně přístupný internet, kde jej poskytovatel vystavuje. Většinou se jedná o emailové schránky, ale jde i například o datová úložiště, online editory fotografií atd. Nejčastěji jsou takovéto služby k užití zdarma s daným limitem. V případě emailů to může být počet, velikost příloh a v případě úložiště pak jeho velikost, maximální velikost jednoho souboru, případně limitovaný objem přenesených dat nebo rychlost komunikace. V tomto případě je nutné počítat s tím, že poskytovatel takovýchto služeb používá anonymizovaná data o používání a o kontextu nahraného obsahu. Tyto služby většinou nabízejí možnost předplatného, které je pravidelně účtováno za určité období, většinou měsíčně, kvartálně nebo ročně. V případě placených tarifů jsou limity často podstatně větší a mají lepší funkce. [36] [37]

4.8.5 Private cloud

Privátní cloud slouží převážně pouze jednomu subjektu, který takový cloud provozuje v rámci společnost. Je možné, že si takovýto cloud společnost vyvíjí samostatně a má naprosto proprietární řešení na míru svých specifických potřeb. Tato cesta bývá velmi nákladná, ale v případě firem, které mají velmi striktní požadavky na ochranu uživatelských dat, nebo mají nějaké podmínky vynucené zákonem, je to jediná cesta, jak využívat cloudových služeb. Možností je také provozování instance některého existujícího cloudového řešení na vlastním serverovém prostoru, například on-premise, což znamená, že jsou servery umístěny na stejné fyzické lokaci jako jejich uživatelé, kteří k těmto serverům přistupují, jedná se o lokální infrastrukturu. [37] [34]

4.8.6 Hybrid cloud

Hybridní cloud je kombinací privátního a veřejného cloudu. Jedná se o řešení, jež lze nalézt ve firmách, které mají data uložená na on-premise serveru, ale chtějí k datům přistupovat vzdáleně z více poboček nebo při práci zaměstnanců z domova. Případně pokud nestačí výkon vlastního privátního cloudu, lze škálovat na veřejný cloud, aniž by měl poskytovatel přístup k soukromým datům. [37] [34] [36]

4.8.7 Platební modely

Cloudová řešení mají různé platební metody. K těm nejvýznamnějším patří [38]:

- Využitý čas – Klient platí pouze za čas, kdy server zpracovává nějaká data a využívá systémové prostředky.
- Využité prostředky – Klient platí za využité systémové prostředky, například cykly procesoru, paměť, úložiště. V případě tohoto modelu je však důležité nastavit limity využití, protože pokud v provozované aplikaci nastane chyba, může nekontrolovatelně spotřebovat mnoho prostředků a vytvořit tak neočekávané náklady.
- Fixní částka – Klient platí předem stanovenou fixní částku za určitý časový interval.

4.8.8 Výhody cloudu

Flexibilita je jednou z nejpodstatnějších výhod při používání cloudových služeb. Protože se za cloudové služby platí v pravidelných intervalech, je jednodušší predikovat náklady za budoucí provoz a používání, pokud jsou však nároky provozovaných aplikací nebo služeb stabilní. Platebních modelů je však více. Flexibilní je i možnost zrušení všech služeb téměř okamžitě, v případě delšího časového úseku v rámci kterého je služba již předplacena je velmi jednoduché předplatné neprodlužovat. [38]

Pro menší podniky a živnostníky jsou výhodou nulové prvotní pořizovací náklady, kde není třeba pořizovat např. servery, kde je nutné jejich prvotní nastavení, konfigurace a je také nutné řešit jejich zabezpečení a umístění na vhodné místo. [32]

Další z velkých výhod je jednoduchá škálovatelnost, protože si klient nepronajímá konkrétní hardware, lze tak téměř neomezeně škálovat potřebné parametry v rámci velmi krátké doby, případně instantně dle potřeby. To v případě fyzických serverů není po dosažení maximální utilizace možné a je nutné stávající servery vylepšovat, koupit další nebo nové. [32]

4.8.9 Nevýhody cloudu

Podstatnou nevýhodou je jednodušší cílení útoků na již zavedené cloudové poskytovatele, jelikož jsou nabízené systémy stejné pro všechny uživatele, je pro hackery snazší nacházet a vyvíjet viry a další škodlivý software a poté ho využívat na mnohem širší spektrum obětí. Útočníci se také mohou vydávat za firmu dodavatele cloudových služeb, převzít kontrolu nad užívanými systémy, a tím získat přístup k citlivým firemním informacím. [39] [40]

Velkou nevýhodou je, že uživatel nemá žádnou kontrolu nad situací v případě výpadku cloudových služeb poskytovatele, a že data jsou uložena na serverech a infrastruktuře třetí strany. Útoky na infrastrukturu cloudových poskytovatelů jsou také mnohem častější. [41]

V zájmu každého jednoho poskytovatele cloudových služeb je, aby zákazník používal co nejvíce služeb exkluzivně. Snaží se tak zákazníky dostat do vendor lock-in stavu a to způsobí, že klient je omezen v možnostech změny nebo přechodu ke konkurenci. V případě růstu společnosti nemá pak jinou možnost, než zvýšit odběr služeb a nemá výhodnou vyjednávací pozici. [41]

4.9 Internetová doména

Doménové jméno je základní součást internetové domény. Je to textová reprezentace IP adresy v internetu, která je jednodušeji čitelná pro lidi. Doména je vždy unikátní, nemůže tak nastat, že má více stránek jednu doménu. Je však možné použití subdomén. Domény takto rozdělujeme do skupin, dle jejich řádu. [42]

4.10 Konkurenční řešení

NoviSign nabízí cloudovou platformu pro tvorbu, správu a plánování obsahu na displejích s možností aktualizací v reálném čase. SignageStudio od MediaSignage poskytuje webový správce obsahu podporující různé formáty médií, plánování a interaktivitu. Xibo je open-source řešení nabízející správu obsahu a softwarového přehrávače a podporuje dynamickou tvorbu obsahu a plánování. Screenly se specializuje na software pro přehrávače založené na Raspberry Pi s cloudovou správou obsahu, s plánováním a možností vzdálené

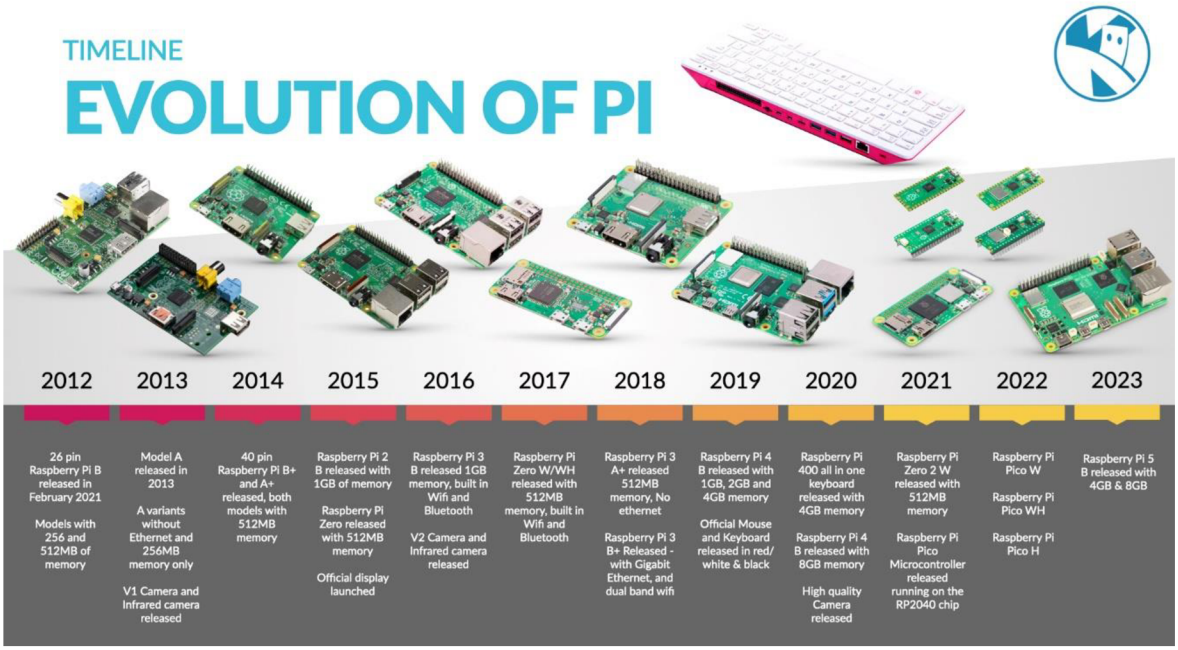
správy. Scala poskytuje komplexní softwarová řešení pro tvorbu, správu a distribuci obsahu s funkcemi pro dynamický obsah, plánování, měření publika a integraci. [43]

4.11 Jednodeskové počítače

Na rozdíl od klasických počítačových sestav, které mají mnoho různých komponent, bez kterých by byl jejich provoz nemožný nebo velmi omezený, jsou jednodeskové počítače samostatně funkční celky, které ke své funkci nepotřebují žádné přídavné komponenty. Na jedné základové desce je přítomen jak procesor, tak paměť, síťové rozhraní a všechny výstupní a vstupní konektory, např. výstup pro video, audio, porty pro periferie jako USB nebo DSI port pro displej nebo CSI port pro kameru, případně také pevné úložiště nebo slot pro paměťovou kartu. V některých případech můžeme na desce nalézt i zmenšený PCIE konektor. Jednotlivé komponenty většinou nelze měnit nebo vylepšovat, což může být limitující, ale tato nevýhoda je vykoupena velkou jednoduchostí a praktičností při provozování těchto zařízení. [44] [45]

Podstatným benefitem při použití jednodeskových počítačů, jsou nízké pořizovací náklad. Také provozovací náklady bývají velice nízké. Příkladem nejznámějšího a nejrozšířenějšího jednodeskového počítače je Raspberry Pi, které má v době psaní této práce již pátou evoluční variantu. Pořizovací náklady Raspberry Pi 5 jsou dle typu konfigurace za samotný jednodeskový počítač v rozmezí 1600 Kč až 2200 Kč. Spotřeba takového modelu je pak přibližně 5 W. [44] [46]

Jednotlivci využijí jednodeskové počítače jako velmi levnou, dostupnou a jednoduchou bránu do světa prototypování jak softwarových, tak hardwarových systémů a zařízení. V případě, že provozování aplikace nemá velké nároky a nezpracovává velké množství dat, je možné tato zařízení použít jako serverové zařízení. Jednodeskové počítače se často používají jako nejrůznější kontroléry, IoT zařízení, v robotice, kioscích a multimediálních serverech. Zajímavým použitím je také v oblasti domácí automatizace. [47]



Obrázek 11: Evoluce modelů Raspbery Pi [48]

Výkon jednodeskových počítačů je v dnešní době již na dostatečné úrovní, díky níž umožňuje běh více aplikací najednou a nedochází k pomalým odezvám při komunikaci. V podnikové sféře se díky nízké ceně, vysoké dostupnosti a rozšířenosti používají jako klientské zařízení, které nemusí zpracovávat data a slouží pouze jako výstupní zařízení pro zobrazování nějakého obsahu nebo pro měření libovolných veličin pomocí přídavných senzorů. Raspberry Pi je jako platforma velmi rozšířená a díky tomu udává standard a směr v oblasti budoucího rozvoje. Existence velkého množství dostupných materiálů velmi usnadňuje začátky jak začátečníkům, tak profesionálům. [49]

4.12 3D tisk

3D tisk je aditivní proces výroby, což znamená, že se materiál postupně přidává. Opakem je zavedený a velmi známý subtraktivní proces např. obrábění. [50]

Na začátku procesu 3D tisku je 3D model, jenž lze vytvořit a vymodelovat za použití CAD softwaru. 3D model je následně rozdělen do jednotlivých vrstev, tento proces se nazývá slicing. Slicerů existuje celá řada, většinou má každý výrobce 3D tiskáren svůj, nebo nějak upravený open source od jiných vývojářů. Slicer má celou řadu nastavení, hlavními parametry jsou výška jednotlivých vrstev, tloušťka, typ podpor nebo rychlosti tisku. Slicer

generuje gcode, který pak 3D tiskárna používá jako instrukce. Pro 3D tiskárny je k dispozici velké množství materiálů, od standardních plastových až po typy s příměsí kovu nebo dřeva. Na různé typy materiálů jsou pak potřeba různé typy tiskových konců. Velká výhoda 3D tisku je velmi rychlé a nízkonákladové prototypování. [51]

4.13 VESA uchycení

VESA mount je spíše lidový a vžitý pojem pro rozměr standardizovaného držáku pro televizní a počítačové obrazovky. V profesionálním prostředí je zaveden anglický pojem FDMI neboli „Flat Display, Mounting Interface“, také je označován jako „VESA Mounting Interface Standard“ – MIS. Držáky tohoto typu jsou nejrozšířenějším standardem sloužícím pro uchycení obrazovek a je k dispozici v několika různých rozměrech a váhových omezení. [52]

Part	Max. váha	Rozměr v mm	Velikost šroubu
B	2	20×50	M4
C	4,5	35×75	
D	8	75×75, 50×75	
	14	100×100, 50×100	
E	22,7	100×200, 50×200	
F	50	200×200 and up	M6
F	113,6	200×200 and up	M8

Tabulka 3: Přehled FDMI standardů [52]

5 Vlastní práce

Žijeme v době, kdy se podniky stále více zabývají vizualizací dat a s tím souvisí jejich zobrazovaní v rámci firemních prostor. K tomu slouží informační tabule a LCD obrazovky, které jsou umístěny v kancelářích, skladech, pobočkách apod. V rámci praktické části této práce je za použití technologií popsaných v teoretické části práce navržen a vytvořen prototyp informačního systému, který umožní hromadnou správu koncových zařízení a správu obsahu na těchto tabulích. Tento systém je určen fiktivní firmě, jejíž cílem je zavést zobrazování metrik v rámci poboček a efektivně a vzdáleně spravovat zobrazovaný obsah.

Tato práce přináší pohled na proces navrhování a implementace informačních systémů. Tento příklad IS pro informační tabule může sloužit jako základ pro další rozvoj možností správy a vizualizace dat v podnikovém prostředí.

5.1 Analýza

Při záměru vyvinout nový informační systém, případně nějakým způsobem upravit či rozšířit ten stávající, je třeba definovat, jaké nároky jsou na takové změny kladeny. Tyto požadavky vznikají z potřeb zaměstnanců v podnicích.

V analytické části bylo třeba tyto požadavky a nároky na vyvíjený informační systém zjistit a vytyčit. Pro následující vývoj a návrh aplikace byly vymezeny funkcionality, které systém musí splňovat a vykonávat. Všechny tyto nároky byly následně dokumentovány a rozděleny do dvou typů, funkčních a nefunkčních požadavků.

Funkční požadavky pro tento systém byly následující:

- Systém má několik stupňů přístupu
- Uživatel má možnost měnit obsah zobrazovaný na displejích
- Administrátor má v systému možnost přidávat nebo odebírat koncová zařízení
- Možnost monitorování stavu zařízení
- Možnost restartování koncových zařízení
- Systém by měl být pouze pro pověřené osoby

Nefunkční systémové požadavky:

- Systém je zabezpečený pomocí přihlášení uživatele

- Systém je kompatibilní s displeji více výrobců a typů
- Systém je jednoduchý na použití
- Systém je dostupný z internetu

5.1.1 SWOT analýza

SWOT analýza se zaměřuje na silné a slabé stránky, příležitosti a hrozby při vývoji řešení digitálních informačních tabulí.

Silné stránky

- Obrazovky poutají pozornost
- Zobrazení jen potřebného obsahu
- Zvyšuje povědomí zaměstnanců o dění v podniku

Slabé stránky

- Počáteční investice do hardwaru a softwaru
- Je nutná pravidelná aktualizace obsahu
- Nutnost údržby

Příležitosti

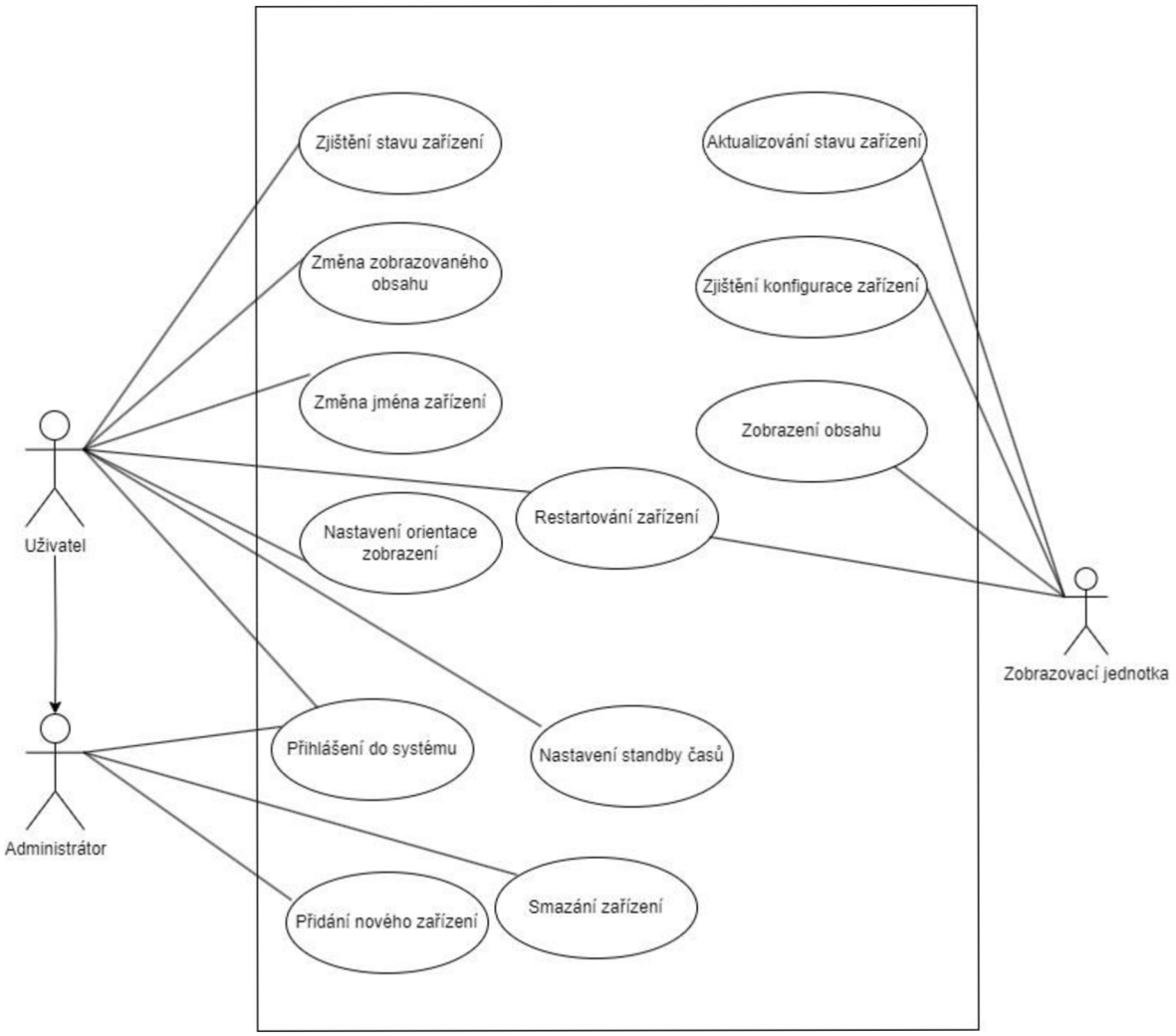
- Zvyšující se poptávka
- Integrace s AI

Hrozby

- Velké tempo rozvoje a změn obdobných systémů
- Vysoká konkurence
- Kyberbezpečností obavy – útoky, krádež informací

5.1.2 Případy použití

Pro vizualizaci použití byl vytvořen diagram případů užití, neboli use case diagram. Dále byly stanoveni aktéři, kteří s informačním systémem interagují nebo provádí nějaké operace. Několik případů použití bylo popsáno detailně, včetně podmínek spuštění, základních a alternativních toků a diagramů aktivit.



Obrázek 12: Diagram případů použití

5.1.3 UC Zjištění konfigurace zařízení

Aktéři

- Systém
- Zobrazovací jednotka

Podmínky spuštění

- Zobrazovací jednotka je připojena k elektrické energii

Základní tok

1. Zobrazovací jednotka zjistí svoji MAC adresu
2. Zobrazovací jednotka přistupuje na API rozhraní systému a předává MAC adresu

3. Systém zjistí, zda Zobrazovací jednotka s předanou MAC adresou je zaregistrováno
4. Systém odpoví s konfigurací
5. Zobrazovací jednotka odpověď rozdělí do jednotlivých proměnných
6. Zobrazovací jednotka nastaví orientaci displeje
7. Zobrazovací jednotka se nepotřebuje restartovat
8. Zobrazovací jednotka není v čase, kdy je vynucen režim standby
9. Zobrazovací jednotka zapíná obrazovku
10. Zobrazovací jednotka nepotřebuje změnit zobrazovaný obsah
11. Zobrazovací jednotka systému reportuje že je dostupné
12. Zobrazovací jednotka čeká 10 sekund
13. Zobrazovací jednotka přejde k bodu 1

Alternativní tok 1

1. Systém zjistí, zda Zobrazovací jednotka s předanou MAC adresou je zaregistrováno
2. Systém odpoví chybou
3. Zobrazovací jednotka na příkazový řádek vypíše chybu
4. Zobrazovací jednotka čeká 10 sekund
5. Zobrazovací jednotka přejde k základnímu toku od bodu bodu 1

Alternativní tok 2

1. Zobrazovací jednotka je třeba restartovat
2. Zobrazovací jednotka se restartuje
3. Zobrazovací jednotka přejde k základnímu toku od bodu bodu 1

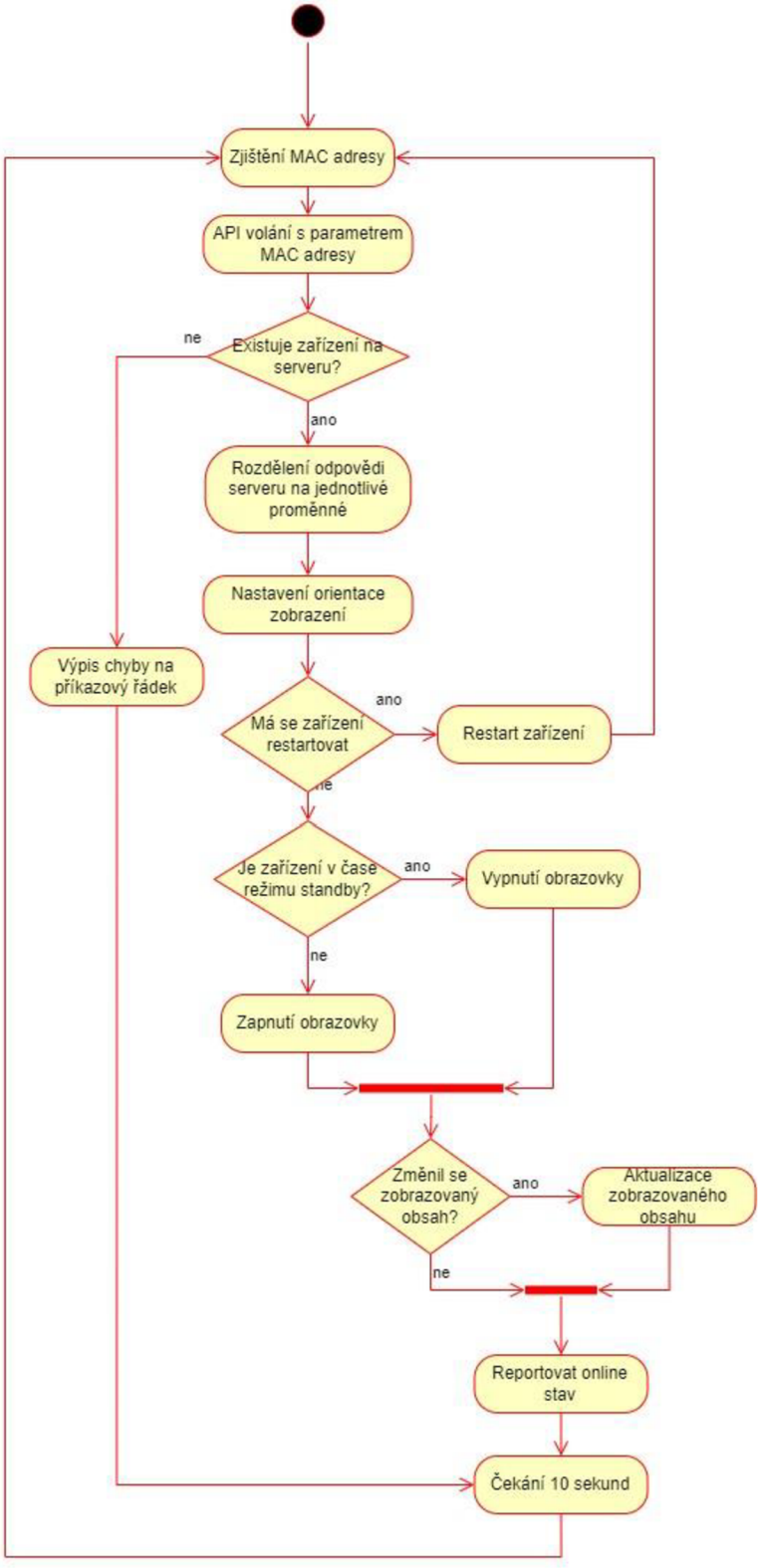
Alternativní tok 3

1. Zobrazovací jednotka je v čase, kdy je zapnutý standby režim
2. Zobrazovací jednotka vypne obrazovku
3. Zobrazovací jednotka přejde k základnímu toku od bodu bodu 10

Alternativní tok 4

1. Zobrazovací jednotka potřebuje změnit zobrazovaný obsah
2. Zobrazovací jednotka aktualizuje zobrazovaný obsah
3. Zobrazovací jednotka přejde k základnímu toku od bodu bodu 11

Protože je celý script nekonečná smyčka, končí jen v případě fatální chyby nebo vypnutím zobrazovací jednotky.



Obrázek 13: Diagram aktivit zjištění konfigurace zařízení

5.1.4 UC Přidání nového zařízení

Aktéři

- Administrátor
- Systém

Podmínky spuštění

- Administrátor musí být přihlášen

Základní tok

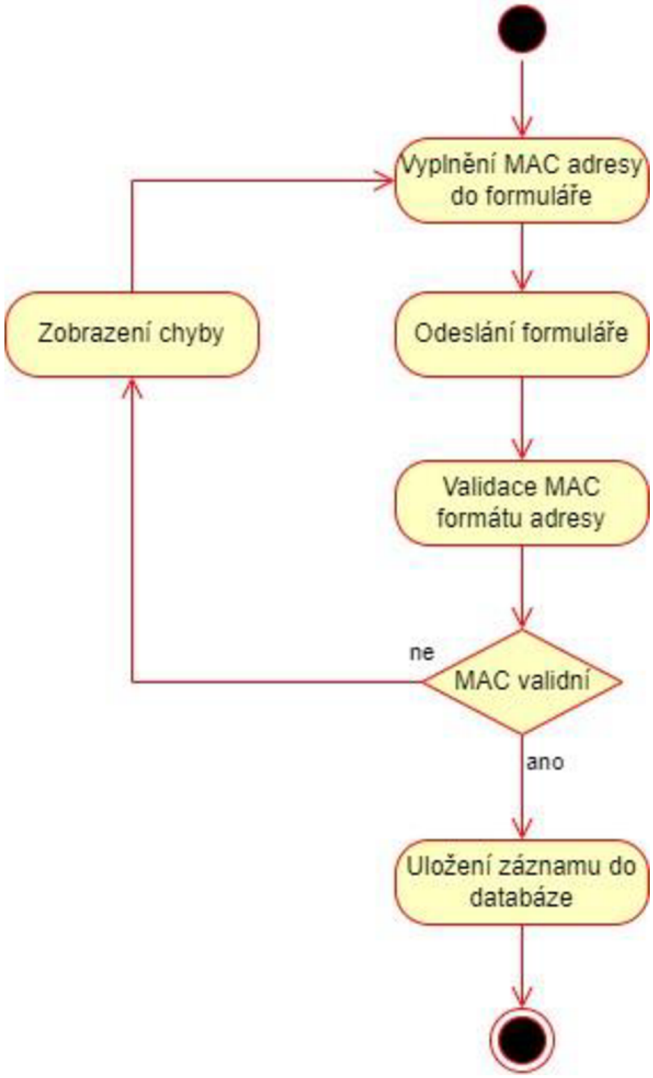
1. Uživatel vyplní MAC adresu do příslušného textového pole
2. Uživatel potvrdí přidání zobrazovací jednotky
3. Systém validuje správnost formátu zadané MAC adresy
4. Systém uloží nový záznam do databáze

Alternativní tok

1. Systém vyhodnotí nesprávný formát MAC adresy
2. Systém zobrazí chybu
3. Uživatel opraví zadanou MAC adresu
4. Systém pokračuje v základním toku od bodu 2

Podmínky dokončení

- Systém úspěšně založí nové zařízení



Obrázek 14: Diagram aktivit přidání nového zařízení

5.2 Návrh informačního systému

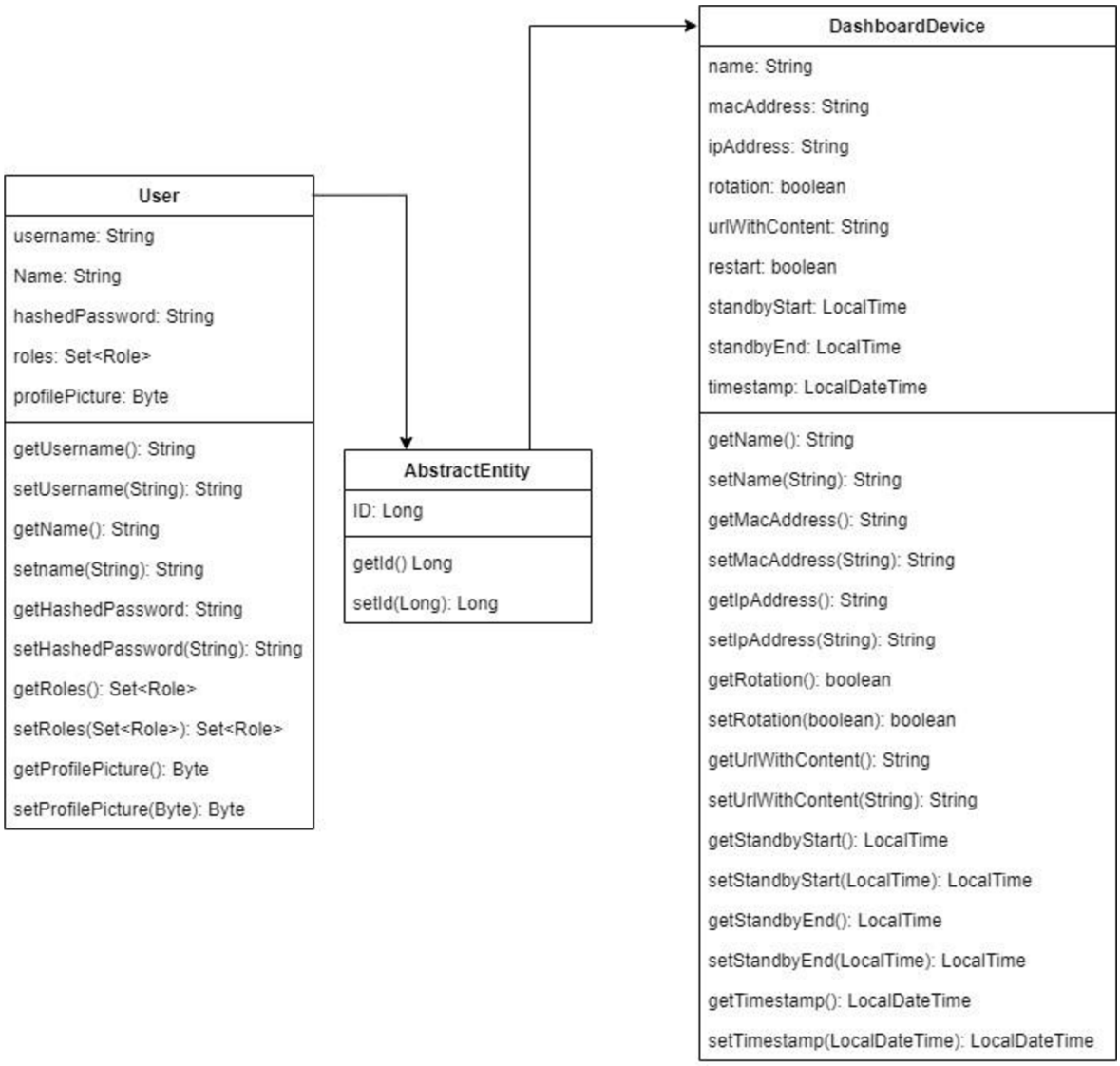
Uživatelské rozhraní by mělo být přehledné, intuitivní a lehce použitelné. Informační systém bude pro komunikaci s uživatelem používat dvě obrazovky. Pro navigaci uživatele bude vytvořeno menu a bude také zobrazena ikona a název uživatele. Návrh byl vypracován v programu Figma.

5.2.1 Diagram tříd

V informačním systému budou stanoveny 3 základní třídy, základní třídou je abstraktní entita, která je použita pro všechny ostatní třídy, které dědí její vlastnosti. Uvnitř třídy je deklarována proměnná Id, které jsou nastaveny funkce jako automatické

inkrementování hodnoty pro identifikaci jednotlivých záznamů. Není tak nutné nastavovat identifikační číslo pro každou jinou třídu zvlášť a konfigurovat jeho chování.

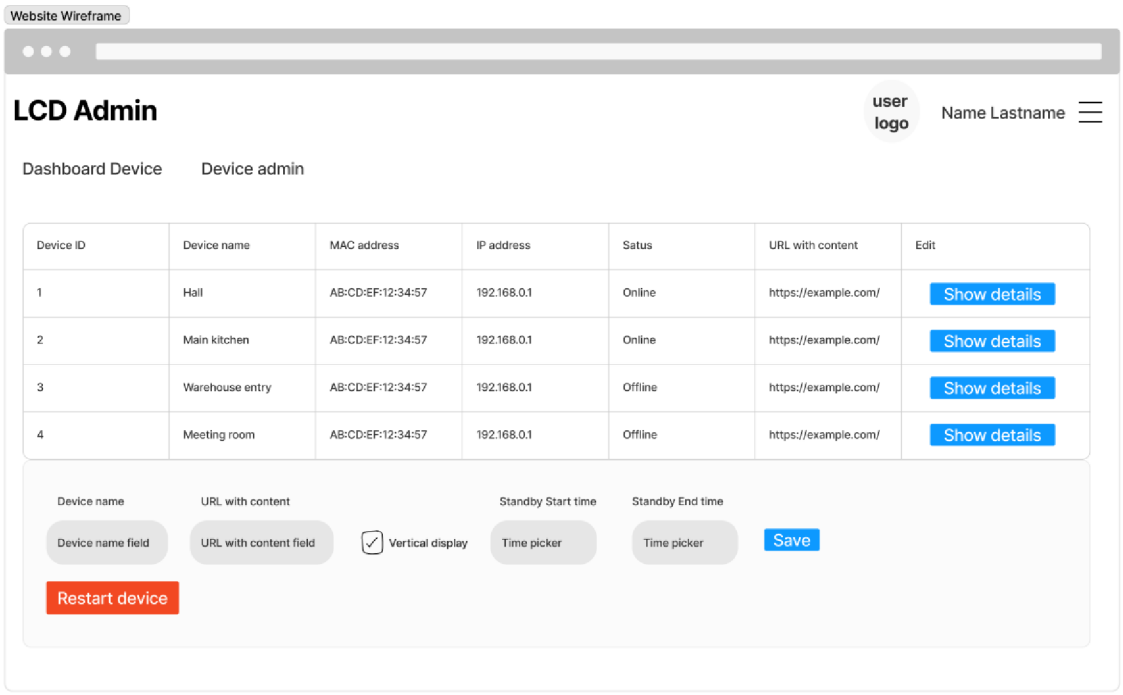
Další důležitou třídou je třída určená pro definování zobrazovací jednotky nazvaná „DashboardDevice“, která poslouží k uložení všech potřebných vlastností, které jsou pro toto použití vhodné.



Obrázek 15: Diagram tříd

5.2.2 Rozhraní běžného uživatele

Základním úkolem je zobrazit uživateli seznam obrazovek, neboli koncových zařízení. Vaadin Framework má pro tyto účely vhodný modul grid. Grid bude použit na zobrazení seznamu zařízení ve formě tabulky. Na každém řádku bude jedno ze zařízení a budou zobrazeny informace o jméně zařízení, MAC adrese, IP adrese, statusu a zobrazovaném obsahu. Dále zde bude umístěno jedno tlačítko, které bude sloužit k otevření menu pro úpravu konfigurace jednotlivých zařízení. Bude možné upravit jméno zařízení, zobrazovaný obsah, rotaci displeje a časy pro přepnutí obrazovky do režimu standby. K uložení změn bude sloužit tlačítko. Dále zde bude umístěno tlačítko sloužící pro restartování zařízení.



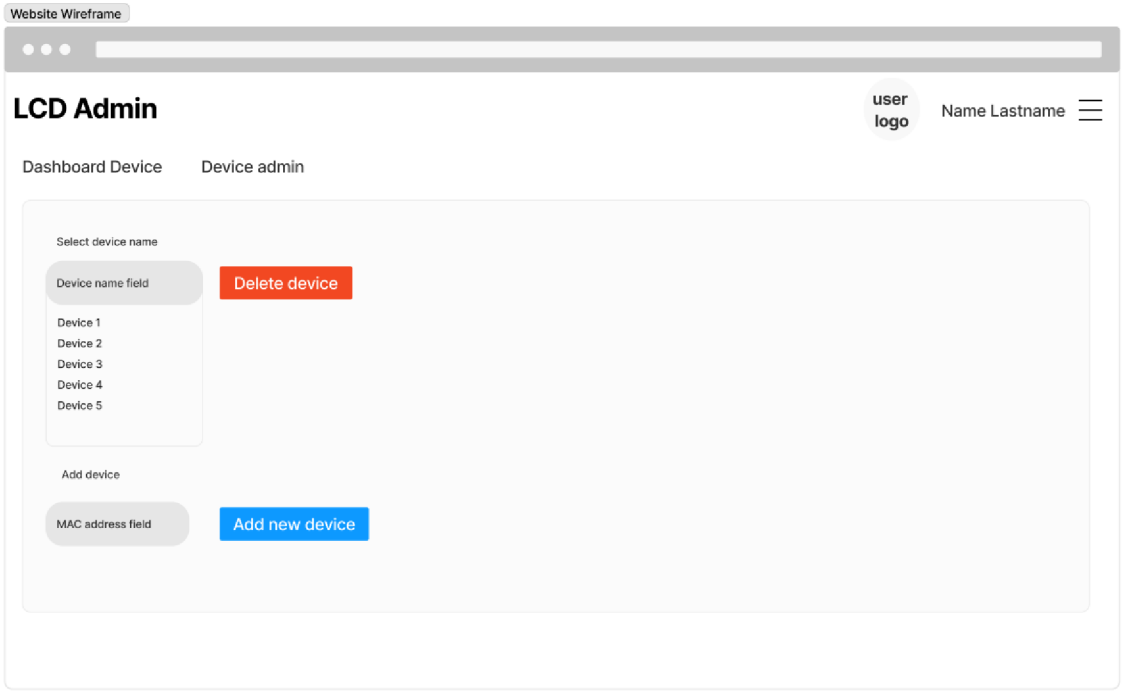
Obrázek 16: Návrh rozhraní běžného uživatele

5.2.3 Rozhraní administrátora

Administrátor IS bude mít všechny možnosti jako běžný uživatel, bude se mu navíc zobrazovat další stránka, kde bude možné přidávat a mazat zařízení.

Mazání zařízení bude uskutečněno výběrem konkrétního zařízení z rozbalovacího seznamu a následně potvrzeno tlačítkem. Tlačítko by mělo být červené, naznačující určitou důležitost.

Přidávání zařízení bude fungovat pomocí textového pole, které bude sloužit pro zadání MAC adresy zařízení, které chce administrátor přidat. Textové pole by mělo být povinné a akceptovat pouze formát MAC adres. Vedle pole bude tlačítko pro potvrzení přidání zařízení.



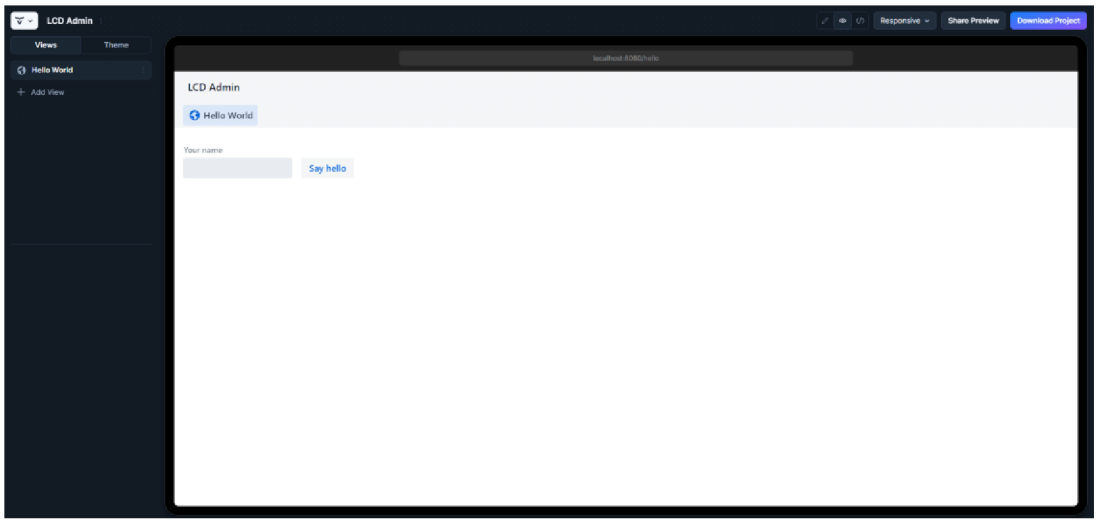
Obrázek 17: Návrh rozhraní administrátora

5.3 Vývoj IS

Samotný informační systém byl vyvíjen v programovacím jazyce Java ve verzi OpenJDK 21 za pomocí open source vývojářského prostředí Visual Studio Code do něhož bylo pro práci s javou potřeba nainstalovat několik rozšíření, jmenovitě jde o balík Extension Pack for Java(https://marketplace.visualstudio.com/items?itemName=vscjava.vscode-java-pack), který v sobě obsahuje 6 doplňků - Language Support for Java™ by Red Hat, Debugger for Java, Test Runner for Java, Maven for Java, Project Manager for Java. Tyto balíčky zajišťují pohodlnou práci s Java projekty, lze se díky nim efektivně orientovat v kódu, validovat kód, zjišťovat chyby v projektu a dostávat upozornění na vylepšení, automatické doplňování běžných sekcí kódu atd.

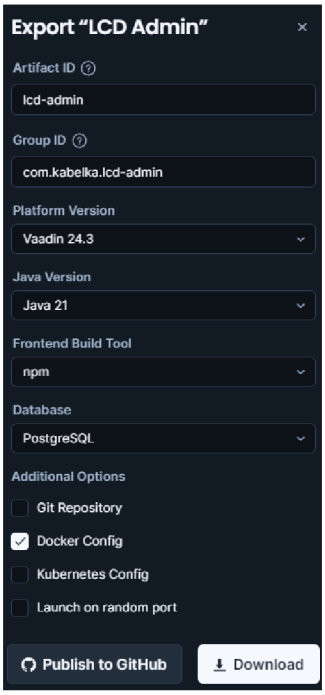
Protože byl pro potřeby této práce použit Vaadin Framework, který je rozšířením Spring boot, bylo za použití nástroje *Vaadin Start*, který je přístupný na adrese https://start.vaadin.com/, vygenerován prázdný startovací projekt, který zajišťuje základní

jednoduché webové rozložení, do kterého se následně doprogramují další moduly a UI prvky.



Obrázek 18: Vaadin Start rozhraní

Před exportem bylo třeba nastavit jméno projektu, identifikátor vývojáře, verzi samotného frameworku, verzi používané Javy a aby Vaadin framework využíval databázi PostreSQL. Posledním krokem bylo, zda uživatel chce aby se vygeneroval Dockerfile.



Obrázek 19: Konfigurace Vaadin projektu

5.3.1 Prvotní konfigurace

Prvním krokem byla konfigurace databázových údajů. To bylo provedeno úpravou souboru „application.properties“, kde bylo nutné nastavit url adresu databázového serveru a databázi, kterou má tato aplikace využívat. Dále bylo nastaveno uživatelské jméno a heslo, pomocí kterého se aplikace připojuje k databázi. V tomto souboru je také možné upravit síťový port, na kterém bude aplikace fungovat. Pro účely vývoje je zvolen port 8080, ale v produkčním prostředí po dokončení vývoje aplikace bude nastaven port 80. Další konfigurační položky jsou ponechány ve výchozím stavu a slouží k optimalizaci při vývoji a produkčnímu fungování.

```
server.port=${PORT:8080}
logging.level.org.atmosphere = warn
spring.mustache.check-template-location = false

vaadin.launch-browser=true

spring.datasource.url = jdbc:postgresql://34.74.17.206:5432/lcdadmindb
spring.datasource.username = lcd_admin_user
spring.datasource.password = oMcgN8GqiEjrgW
spring.jpa.hibernate.ddl-auto = update

vaadin.allowed-packages = com.vaadin,org.vaadin,dev.hilla,com.kabelka.lcd.admin
spring.jpa.defer-datasource-initialization = true
spring.sql.init.mode = always
```

Obrázek 20: Obsah souboru application.properties

5.3.2 Třída DashboardDevice

Po konfiguraci a napojení databáze bylo třeba vytvořit vhodnou třídu, která obsáhne veškeré potřeby pro uložení dat o jednotlivých zařízeních. Ve Vaadin frameworku je k dispozici třída abstraktní entity, která obstarává unikátní identifikační indexy pro užití v databázi, což usnadní vytváření dalších tříd, protože již není nutné zakládat index pokaždé znovu. Třída zařízení byla nazvaná „DashboardDevice“ a rozšiřuje již zmíněnou abstraktní třídu. Třída je složena z několika proměnných.

Pro lepší uživatelský zážitek je použitá proměnná, která umožňuje pojmenovat každé zařízení samostatným jménem, byla zvolená proměnná typu String, do které lze uložit textový řetězec.

Aby bylo možné unikátně identifikovat každé zařízení, byla za tímto účelem vytvořená proměnná „macAddress“ typu String, která uchovává unikátní MAC adresu každého zařízení.

Proměnná „ipAddress“ je použita za účelem ukládání lokální IP adresy zařízení za účelem informovanosti uživatele informačního systému. Jedná se proměnnou typu String.

Díky proměnné „timestamp“ typu LocalDateTime, která může obsahovat časový údaj posledního přístupu a může uživateli zjistit stav zařízení, zda je dostupné nebo ne. To může napovědět o provozuschopnosti daného zařízení a umožňuje tak proaktivně monitorovat a dále řešit možné výpadky nebo selhání.

Zařízení má mít možnost být přepnuto do režimu horizontálního nebo vertikálního zobrazení, aby bylo možné toto chování stanovit, byla za tímto účelem vytvořena proměnný „Rotation“, která je typu boolean a nastavuje právě rotaci zobrazení.

Nejpodstatnější funkcí vyvíjeného informačního systému je správa zobrazovaného obsahu, proto je vytvořená také proměnná „urlWithContent“ typu String, do které lze uložit odkaz na obsah, který chce uživatel zobrazit na displejích jednotlivých zařízení.

Funkčním požadavkem informačního systému byla také možnost vzdáleného restartování jednotlivých zařízení. K tomu účelu je použitá proměnná „Restart“ typu boolean, která udává informaci, zda se zařízení má restartovat.

Proměnné „standbyStart“ a „standbyEnd“ typu LocalTime jsou použity pro stanovení doby, kdy mají být jednotlivé obrazovky přepnuty do režimu standby, který vypne obrazovku a je díky tomu šetřena elektrická energie.

```java
@Entity
public class DashboardDevice extends AbstractEntity{

    private String Name;

    private String macAddress;

    private String ipAddress;

    private boolean Rotation;

    private String urlWithContent;

    private boolean Restart;

    private LocalTime standbyStart;

    private LocalTime standbyEnd;

    private LocalDateTime timestamp;
```

Obrázek 21: Třída deklarující koncové zařízení

5.3.3 JPA Repositář a služba

Spring boot závislosti „spring-boot-starter-data-jpa“ a „postgresql“ byly použity za účelem zjednodušení práce s databázovým systémem. Aby bylo možné zapisovat data o jednotlivých zařízeních do databáze, je třeba vytvořit repositář pro zmíněnou třídu. Tento repositář zajistí vytvoření potřebných tabulek přímo v databázi a překlad Java instrukcí na SQL kód, pomocí něhož konkrétní databáze PostgreSQL komunikuje. JPA nabízí základní funkce, jako je hledání podle ID, ukládání nových záznamů, aktualizaci již existujících záznamů, mazání podle ID, počítání, vypisování atd. Protože budou jednotlivá zařízení identifikovaná pomocí jejich síťových MAC adres, bylo do repositáře tuto nevýchozí funkci vytvořit pomocí příkazu „DashboardDevice findByMacAddress(String mac Address);“

```java
package com.kabelka.lcd.admin.data;

import org.springframework.data.jpa.repository.JpaRepository;

public interface DashboardDeviceRepository extends JpaRepository<DashboardDevice, Long> {
    DashboardDevice findByMacAddress(String macAddress);
}
```

Obrázek 22: Obsah repositáře třídy DashboardDevice

Při použití Spring bootu je možné komunikovat s repositářem napřímo, avšak běžným a také doporučeným postupem je vytvoření služby, která se stará o komunikaci s repositářem a výrazně zpřehledňuje celý proces vývoje a prací s daty.

```java
@Service
public class DashboardDeviceService {
    @Autowired
    private DashboardDeviceRepository dashboardDeviceRepository;
    public List<DashboardDevice> getAllDashboardDevices() {
        return dashboardDeviceRepository.findAll();
    }
    public Optional<DashboardDevice> getDashboardDeviceById(Long id) {
        return dashboardDeviceRepository.findById(id);
    }
    public DashboardDevice createDashboardDevice(DashboardDevice dashboardDevice) {
        dashboardDevice.setRestart(false);
        dashboardDevice.setRotation(false);
        dashboardDevice.setName("Just added");
        dashboardDevice.setTimestamp(LocalDateTime.now().minusMinutes(10));
        return dashboardDeviceRepository.save(dashboardDevice);
    }
    public DashboardDevice updateDashboardDevice(Long id, DashboardDevice updatedDashboardDevice) {
        DashboardDevice dashboardDevice = dashboardDeviceRepository.findById(id).orElse(null);
        if (dashboardDevice != null) {
            dashboardDevice = updatedDashboardDevice;
            return dashboardDeviceRepository.save(dashboardDevice);
        }
        return null;
    }
    public void deleteDashboardDevice(Long id) {
        dashboardDeviceRepository.deleteById(id);
    }
    public DashboardDevice findByMacAddress(String macAddress) {
        return dashboardDeviceRepository.findByMacAddress(macAddress);
    }
}
```

Obrázek 23: Kód služby pro komunikaci s repositářem

5.3.4 Uživatelské rozhraní

Následujícím krokem bylo vytvoření uživatelského rozhraní, kombinace Spring bootu a Vaadin frameworku umožnila použití standardizovaných UI komponentů. Toto rozhraní je označeno jako view – pohled. Pomocí syntaxe začínající „@“ byl nastaven titulní název stránky, následně bylo specifikováno, že je přístup na tuto stránku možný pouze po přihlášení a s rolemi „ADMIN“ a „USER“. Také je stanovena cesta, pod kterou je možné na webovou stránku přistupovat. Je také specifikováno, aby stránka používala výchozí rozložení, pomocí kterého se zobrazuje hlavička s nadpisem, navigační menu a ikona aktuálního uživatele.

```
@PageTitle("Dashboard devices")
@RolesAllowed({"ADMIN", "USER"})
@Route(value = "dashboard-devices", layout = MainLayout.class)
@RouteAlias(value = "", layout = MainLayout.class)
@Uses(Icon.class)
public class DashboardDeviceView extends VerticalLayout {
    private final DashboardDeviceService dashboardDeviceService;
    private final Grid<DashboardDevice> grid;

    public DashboardDeviceView(DashboardDeviceService dashboardDeviceService) {
        this.dashboardDeviceService = dashboardDeviceService;
```

Obrázek 24: Deklarace třídy pro UI přehledu zařízení

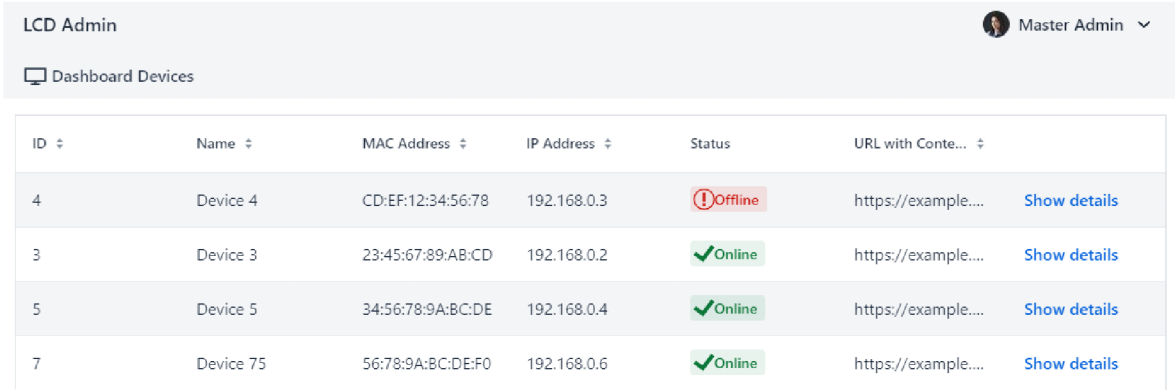
Díky napojení služby, která umožňuje přístup k datům v databázi lze v pohledu data používat. Je deklarován modul Grid, který má původ ve Vaadin frameworku, tomuto gridu je přiřazen objekt „DashboardDevice“. Pohled je pojmenován „DashboardDeviceView“ a jako parametr používá dříve napojenou službu „DashboardDeviceService“. Grid ve výchozím stavu zobrazuje kvůli výkonu pouze omezené množství řádků. Protože bude aplikace fungovat s maximálně desítkami koncových zařízení, není toto omezení nutné a je vynuceno zobrazení všech řádků najednou. Toho je docíleno za pomocí příkazu „grid.setAllRowsVisible(true);“

Zobrazovaná data jsou do gridu v tomto případě přidána po jednotlivých sloupcích pomocí příkazu „grid.addColumn().setHeader()“, kde první parametr je název proměnné třídy „DashboardDevice“ a druhý parametr je zobrazovaný popisek sloupce.

```
grid.addColumn("id").setHeader("ID");
grid.addColumn("name").setHeader("Name").setAutoWidth(true);
grid.addColumn("macAddress").setHeader("MAC Address");
grid.addColumn("ipAddress").setHeader("IP Address");
grid.addColumn("urlWithContent").setHeader("URL with Content")
```

Obrázek 25: Kód přidání sloupců do gridu

Aby měl uživatel také přehled o aktuálním stavu zařízení, bylo nutné graficky vyjádřit příslušnou použitou proměnnou. K tomu byla zvolena Vaadin komponenta „Badge“, která pomocí červené, respektive zelené ikony a nápisu online a offline uživateli jasně předá chtěnou informaci.



Obrázek 26: Uživatelské rozhraní

Indikace stavu zařízení byly přidány jako jeden ze sloupců v rozhraní, kde se pomocí rozdílu mezi aktuálním systémovým časem a časem posledního přístupu zařízení zjistí, zda je tento rozdíl vyšší než 10 sekund a dle toho se zobrazuje výsledná informace.

```
grid.addColumn(new ComponentRenderer<>(item -> {
    Span badge;
    Duration duration = Duration.between(item.getTimestamp(),LocalDateTime.now());
    long secondsDifference = duration.getSeconds();
        if (secondsDifference < 10) {
            badge = new Span(new Icon(VaadinIcon.CHECK), new Span(" Online"));
            badge.getElement().getThemeList().add("badge success");
        } else {
            badge = new Span(new Icon(VaadinIcon.EXCLAMATION_CIRCLE_O), new Span(" Offline"));
            badge.getElement().getThemeList().add("badge error");
        }
        return badge;
})).setHeader("Status");
```

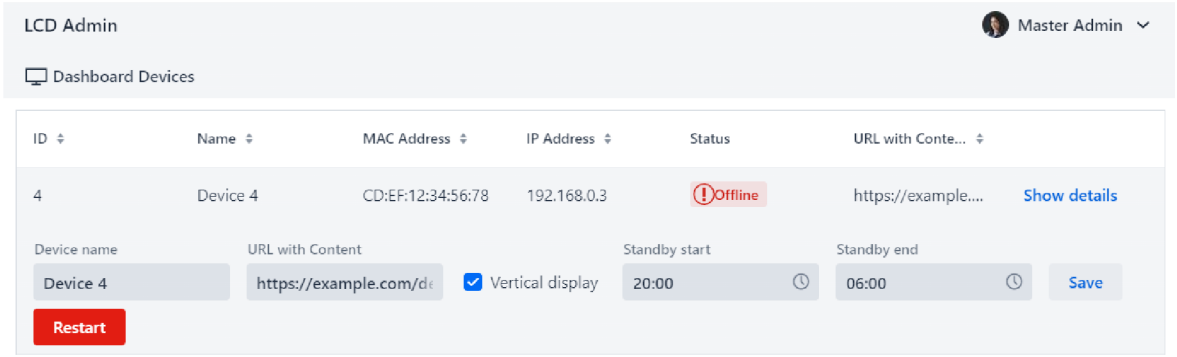
Obrázek 27: Kód pro zobrazení stavu zařízení v UI

Protože jedním z požadavků vyvíjení informačního systému je nejen zobrazení informací o jednotlivých zařízeních, ale i možnost úprav zvolených proměnných, bylo do dalšího sloupce přidáno tlačítko, které zobrazí panel, kde je možné upravovat jméno zařízení, URL adresu s požadovaným obsahem k zobrazení na displeji koncového zařízení, je zde možnost volby vertikálního zobrazení a dále nastavení času vypnutí a zapnutí obrazovky pro šetření elektrické energie. K uložení změněných dat je přidané tlačítko.

```
private void updateDevice(DashboardDevice device){
        dashboardDeviceService.updateDashboardDevice(device.getId(), device);
    }
```

Obrázek 28: Kód tlačítka pro uložení změn

Koncový počítač lze také restartovat pomocí tlačítka, pro které byla zvolena červená barva, aby se uživatel nespletl s tlačítkem pro uložení.



Obrázek 29: Uživatelské rozhraní detailu zařízení

Tlačítko pro zobrazení panelu s úpravami zařízení je také vloženo jako jeden ze sloupců a je vytvořeno funkcí „createToggleDetailsRenderer“, která zajistí umístění tlačítek do sloupce. Ve výchozím stavu je možné zobrazení detailů kliknutím na řádek s daty, tomu bylo zabráněno příkazem „grid.setDetailsVisibleOnClick(false);“. Aby nedocházelo k problémům s vykreslováním, bylo nutné nastavit indikovat přítomnost funkce detailů k čemuž slouží příkaz „grid.setItemDetailsRenderer(createDeviceDetailsRenderer());“.

```
grid.addColumn(createToggleDetailsRenderer(grid));
grid.setDetailsVisibleOnClick(false);
grid.setItemDetailsRenderer(createDeviceDetailsRenderer());
List<DashboardDevice> dashboardDevices = dashboardDeviceService.getAllDashboardDevices();
grid.setItems(dashboardDevices);
grid.addThemeVariants(GridVariant.LUMO_ROW_STRIPES);
add(grid)
```

Obrázek 30: Kód konfigurace gridu a přidání do UI

Funkce „createDeviceDetailsRenderer()“ je poměrně k ostatním komplexní. Obstarává přidání polí na úpravu zařízení. Pro změnu jména a URL odkazu jsou použity TextFieldy, do kterých jsou přiřazeny hodnoty z databáze. Pro volbu orientace displeje je použito zaškrtávací tlačítko, které pokud je zaškrtnuté, nastaví proměnnou „Rotation“ na hodnotu „true“. Dále je použit modul TimePicker, zajišťující volbu časů ve správném formátu pro uspání a probuzení obrazovky. Případné změny je vždy nutné uložit příslušným tlačítkem. Tlačítko pro restartování je vytvořené pomocí modulu Button. Jednotlivé komponenty vytvořené v kódu se musí do uživatelského rozhraní přidat pomocí „details.add(…);“

```
private ComponentRenderer<Div, DashboardDevice> createDeviceDetailsRenderer() {
    return new ComponentRenderer<>(device -> {
        Div details = new Div();
        TextField nameField = new TextField("Device name");
        nameField.setValue(device.getName());
        TextField urlField = new TextField("URL with Content");
        urlField.setValue(device.getUrlWithContent() != null ? device.getUrlWithContent() : "null");
        Checkbox verticalDisplay = new Checkbox();
        verticalDisplay.setLabel("Vertical display");
        verticalDisplay.setValue(device.isRotation());
        TimePicker stanbyStartTimePicker = new TimePicker("Standby start",device.getStandbyStart());
        stanbyStartTimePicker.setLocale(Locale.GERMAN);
        TimePicker standbyEndTimePicker = new TimePicker("Standby end",device.getStandbyEnd());
        standbyEndTimePicker.setLocale(Locale.GERMAN);
        Button saveButton = new Button("Save", event -> {
            device.setName(nameField.getValue());
            device.setUrlWithContent(urlField.getValue());
            device.setRotation(verticalDisplay.getValue());
            device.setStandbyStart(stanbyStartTimePicker.getValue());
            device.setStandbyEnd(standbyEndTimePicker.getValue());
            device.setTimestamp(LocalDateTime.now());
            device.setRestart(false);
            updateDevice(device);
        });
        Button restartButton = new Button("Restart", event -> {
            device.setRestart(true);
            updateDevice(device);
        });
        restartButton.addThemeVariants(ButtonVariant.LUMO_PRIMARY,ButtonVariant.LUMO_ERROR);
        HorizontalLayout componentsLayout = new HorizontalLayout();
        componentsLayout.setAlignItems(FlexComponent.Alignment.BASELINE);
        componentsLayout.add(nameField, urlField, verticalDisplay, stanbyStartTimePicker, standbyEndTimePicker, saveButton);
        details.add(componentsLayout, restartButton);
        return details;
    });
}
```

Obrázek 31: Kód funkce vytvářející panel s detaily zařízení

5.3.5 API pro koncová zařízení

Aby mohla koncová zařízení komunikovat se serverem, bylo nutné vyvinout příslušné API. Je použit „RestController“ ze Spring Boot, který je přístupný na adrese „/api“. Aby byla možná komunikace s databází, došlo k připojení „DashboardDeviceService“.

Koncové zařízení musí pro získání konfigurace přistoupit na API endpoint, který je dostupný na URL „server/api/device-config“ a je typu GET. Je také třeba připojit parametr „MAC“, který obsahuje fyzickou síťovou adresu daného zařízení a pokud je daná adresa nalezena, obdrží zařízení objekt s proměnnými, dle kterých se zařízení nastaveno. V případě nenalezení MAC adresy je zařízení vrácena odpověď 404, která indikuje, že požadovaný obsah neexistuje.

```java
@RestController
@RequestMapping("/api")
public class DeviceConfigController {
    private final DashboardDeviceService dashboardDeviceService;
    public DeviceConfigController(DashboardDeviceService dashboardDeviceService) {
        this.dashboardDeviceService = dashboardDeviceService;
    }
//http://localhost:8080/api/device-config?DeviceMACAddress=23:45:67:89:AB:CD&continue
    @GetMapping("/device-config")
    public ResponseEntity<?> getDeviceConfig(@RequestParam("MAC") String deviceMACAddress) {
      try {
           DashboardDevice device = dashboardDeviceService.findByMacAddress(deviceMACAddress);
            if (device != null) {
                return ResponseEntity.ok().body(device);
            } else {
               return ResponseEntity.status(HttpStatus.NOT_FOUND).body("Device not found");
            }
      } catch (EmptyResultDataAccessException e) {
        return ResponseEntity.status(HttpStatus.NOT_FOUND).body("Device not found");
     } catch (Exception e) {
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body("An error occurred");
     }
    }
}
```

Obrázek 32: Kód pro RestController a device-config endpoint

Dalším endpointem je „heartbeat“, který také potřebuje MAC adresu jako parametr. Zařízení na tuto adresu pomocí metody PUT přistupuje v pravidelném intervalu a svůj stav ukládá do databáze. Díky tomu může uživatel v téměř reálném čase s přiměřeným zpožděním sledovat stavy jednotlivých zařízení.

```java
//http://172.18.80.1:8080/api/heartbeat?MAC=89:AB:CD:EF:01:23
@PutMapping("/heartbeat")
public ResponseEntity<String> updateDeviceStatus(@RequestParam("MAC") String deviceMACAddress) {
    try {
        DashboardDevice device = dashboardDeviceService.findByMacAddress(deviceMACAddress);
        if (device != null) {
            device.setTimestamp(LocalDateTime.now());
            System.out.println("Device with ID: " + device.getId() + " just checked with timestamp: " + device.getTimestamp());
                dashboardDeviceService.updateDashboardDevice(device.getId(), device);
                return ResponseEntity.ok("Device status updated successfully.");
        } else {
            return ResponseEntity.notFound().build();
        }
    } catch (EmptyResultDataAccessException e) {
        return ResponseEntity.notFound().build();
    } catch (Exception e) {
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body("An error occurred");
    }
}
```

Obrázek 33: Kód heartbeat endpointu

5.3.6 Příprava na produkční nasazení

Aby bylo možné výslednou aplikaci spustit v produkčním prostředí na serveru v GPC, je nutné aplikaci zabalit do spustitelného jar souboru. Toho lze docílit za použití Mavenu, jenž má funkci package, ke které je nutné přidat parametr „-f“, který specifikuje umístění „pom.xml“ souboru. Dále je nutné použít přepínač „-Pproduction“.

```
& "g:\My Drive\lcd-admin\mvnw.cmd" package -f "g:\My Drive\lcd-admin\pom.xml" -Pproduction
```

Obrázek 34: Vytvoření produkčního jar souboru

Protože je cílem použít pro provoz aplikace Docker, bylo třeba vytvořit Dockerfile. S tím se snažil pomoci Vaadin Starter, avšak soubor vygenerovaný z této služby je pro tento projekt nevyhovující a bylo nutné vytvořit jiný.

Dockerfile popisuje, jaké základní prostředí použít k běhu aplikace a dále specifikuje soubor aplikace, který je nutné překopírovat do kontejneru ve kterém poběží. ENTRYPOINT udává, jaký příkaz se má vykonat po spuštění prostředí, v tomto případě se spouští aplikace příkazem „java – jar lcd-admi-1.0-SNAPSHOT.jar“.

```
FROM eclipse-temurin:21-jre
COPY /*.jar lcd-admin-1.0-SNAPSHOT.jar
ENTRYPOINT ["java", "-jar", "/lcd-admin-1.0-SNAPSHOT.jar"]
```

Obrázek 35: Obsah souboru dockerfile

Aby nebylo nutné po každé úpravě aplikace ručně vytvářet nový obraz, byl k tomu vytvořen soubor docker-compose.yml. Běžnou praxí je specifikace verze aplikace, ta je ponechána ve verzi 1. Kontejner bude fungovat jako služba s názvem lcd-admin, dále je specifikovaný kontext, ve kterém bude Docker compose pracovat. Kontextem je lokální složka. Tím, že je specifikovaný dockerfile soubor, dojde k jeho použití a spuštění a je zajištěno automatické vytvoření nové image. Protože je žádoucí, aby byla aplikace dostupná z internetu, je nutné otevřít síťové porty. V tomto konkrétním případě je vnitřní port 8080,

který aplikace používá přesměrován na venkovní port 80, který bude sloužit pro uživatele aplikace.

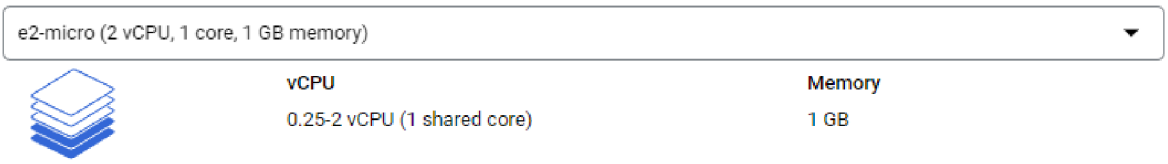
```yaml
version: '1'

services:
  lcd-admin:
    build:
      context: .
      dockerfile: dockerfile
    ports:
      - "80:8080"
```

Obrázek 36: Obsah souboru docker-compose.yml

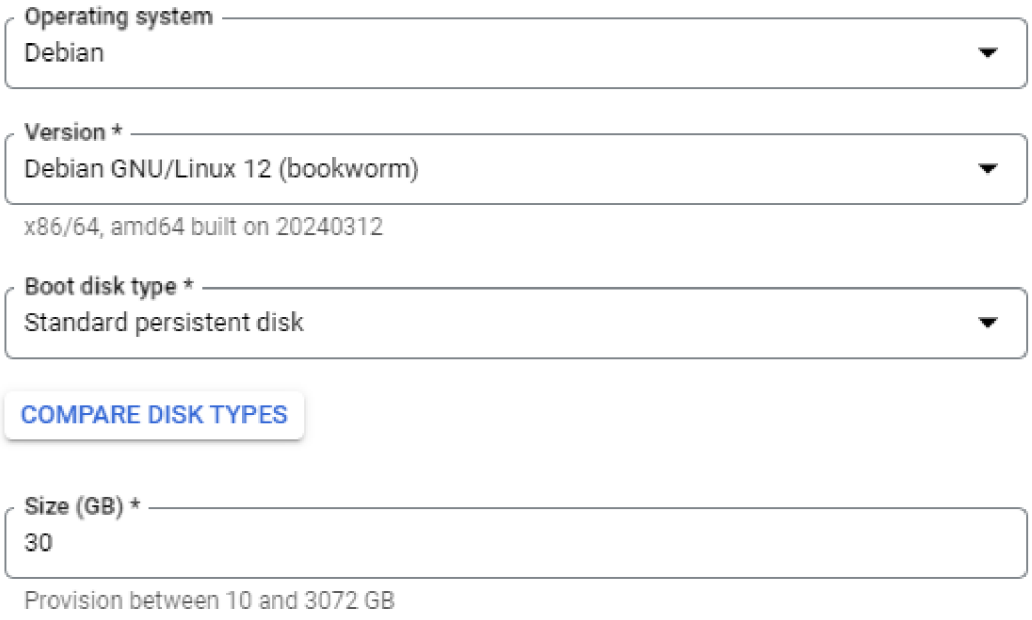
5.3.7 Nastavení cloudového serveru

Pro tuto práci je zvoleno řešení Compute Engine ve verzi e2-micro od Google Cloud Platform, které nabízí bezplatnou základní úroveň výpočetních virtuálních počítačů. Parametry jsou velmi základní, jedná se pouze o 0,25 vCPU, 1 GB operační paměti.

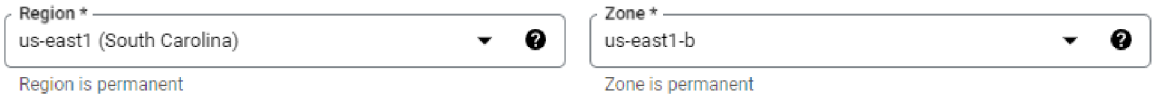


Obrázek 37: Výběr systémových parametrů

Velikost je daná kvótou 30 GB diskového úložiště a virtuální stroj je umístěn v datacentru s označením us-east1. Jako operační systém je zvolena linuxová distribuce Debian ve verzi 12 s označením bookworm.

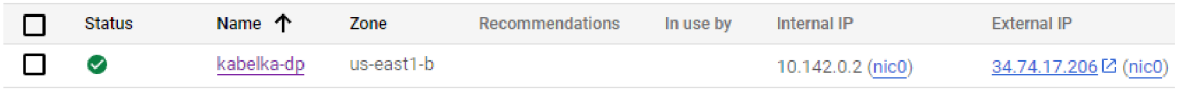


Obrázek 38: Konfigurace disku a OS



Obrázek 39: Umístění datacentra

Po odsouhlasení a několikaminutovém čekání byla vytvořena instance virtuálního počítače, kterému byla přidělena veřejná IP adresa, ta bude sloužit pro komunikaci a vzdálený přístup.



Obrázek 40: Výpis virtuálních strojů v GCP

Verze zdarma, je alespoň přinejmenším pro Evropu vykoupena nevýhodou v podobě vysoké fyzické vzdálenosti a je tedy nutné počítat s delší latencí. Příkazem ping byla ověřena dostupnost virtuálního stroje.

```
C:\Users\kabel>ping 34.74.17.206

Pinging 34.74.17.206 with 32 bytes of data:
Reply from 34.74.17.206: bytes=32 time=110ms TTL=60
Reply from 34.74.17.206: bytes=32 time=109ms TTL=60
Reply from 34.74.17.206: bytes=32 time=108ms TTL=60
Reply from 34.74.17.206: bytes=32 time=108ms TTL=60
```

Obrázek 41: Naměřená latence a odpověď serveru

Na vzdálené připojení do nově vytvořeného virtuálního počítače bylo použito SSH, k jeho zprovoznění bylo ale nejprve nutné vygenerovat šifrovací certifikát a ten následně importovat do konzole GCP. K vygenerování certifikátu byla použita open-source aplikace PuTTYgen.

Po úspěšném přístupu do operačního systému byly použity příkazy sudo apt-get update a sudo apt-get upgrade, které zajistí, aby byly nainstalovány nejnovější aktualizace jednotlivých balíčků zahrnutých v OS.

5.3.8 Konfigurace Dockeru

Jakmile je operační systém Debian plně aktualizovaný, dalším krokem je instalace Docker enginu a jelikož není obsažený ve výchozích repositářích Debianu, je nejprve nutné oficiální repositář Dockeru přidat. Předtím je ale nutné importovat klíče, které ověřují, zda je repositář autentický a oficiální. Po přidání repositáře pomocí příkazu *sudo apt-get update* je aktualizován jeho obsah a následně je již nainstalován samotný Docker engine

```
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/debian/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/debian \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

Obrázek 42: Příkazy instalace Dockeru

5.3.9 Konfigurace databáze

Pro tuto práci byla zvolena databáze Postgres, pro její provozování v dockeru byla zvolena aktuálně nejnovější verze. Aby mohl Docker databázi provozovat, bylo nejprve nutné pomocí příkazu „docker pull postgres“ stáhnout její image.

```
sudo docker pulI postgres
Using default tag: latest
latest: Pulling from library/postgres
8a1e25ce7c4f: Pull complete
902317ed8722: Pull complete
c223965bd9a8: Pull complete
847682431a68: Pull complete
8d29ba654727: Pull complete
fd133663e42b: Pull complete
13de11c6ecda: Pull complete
45bb35744214: Pull complete
d4082e63ce2c: Pull complete
269f33c511c1: Pull complete
7cbaf3c85093: Pull complete
fic82efaOdcd: Pull complete
e9dOd3c40657: Pull complete
68bf5c580643: Pull complete
Digest: sha256:6b841c8f6a819884207492f1209a8116844365df15fca8cf556fc54a24c70899
Status: Downloaded newer image for postgres:latest
docker.io/library/postgres:latest
```

Obrázek 43: Stažení Postgres docker image

Pro vytvoření nového kontejneru, ve kterém bude Postgres databáze fungovat je nutná její počáteční konfigurace. Základním příkazem je „docker run“, ten je však potřeba doplnit o několik parametrů. Aby byl kontejner jednoduše identifikovatelný, byla využita možnost pojmenování „--name lcd-admin-db-postgres“. Postgres image vyžaduje, aby se při vytváření kontejneru nastavilo heslo pro hlavního výchozího uživatele a toho lze dosáhnout za pomocí nastavení proměnné prostředí „POSTGRES_PASSWORD“.

Je podstatné, aby v případě smazání nebo restartování kontejneru nezmizela uložená data. Ve výchozím stavu se vše ukládá do paměti konkrétního kontejneru a po jeho zániku se všechna data ztrácí. Pomocí funkcionality Docker volume lze zajistit, aby se data ukládala na disk do operačního systému mimo kontejner, je tak zajištěno trvalé ukládání dat. V případě této práce je použit přepínač příkazu „docker run“, který definuje volume „-v postgres:/var/lib/postgresql/data“.

Databáze Postgres komunikuje na síťovém portu 5432, pomocí přepínače „-p“ je tento port otevřen a zpřístupněn ke komunikaci i mimo kontejner.

Aby mohl kontejner fungovat na pozadí systému a nebyl závislý na přihlášeném uživateli, je kontejner spuštěn v „detached“ režimu.

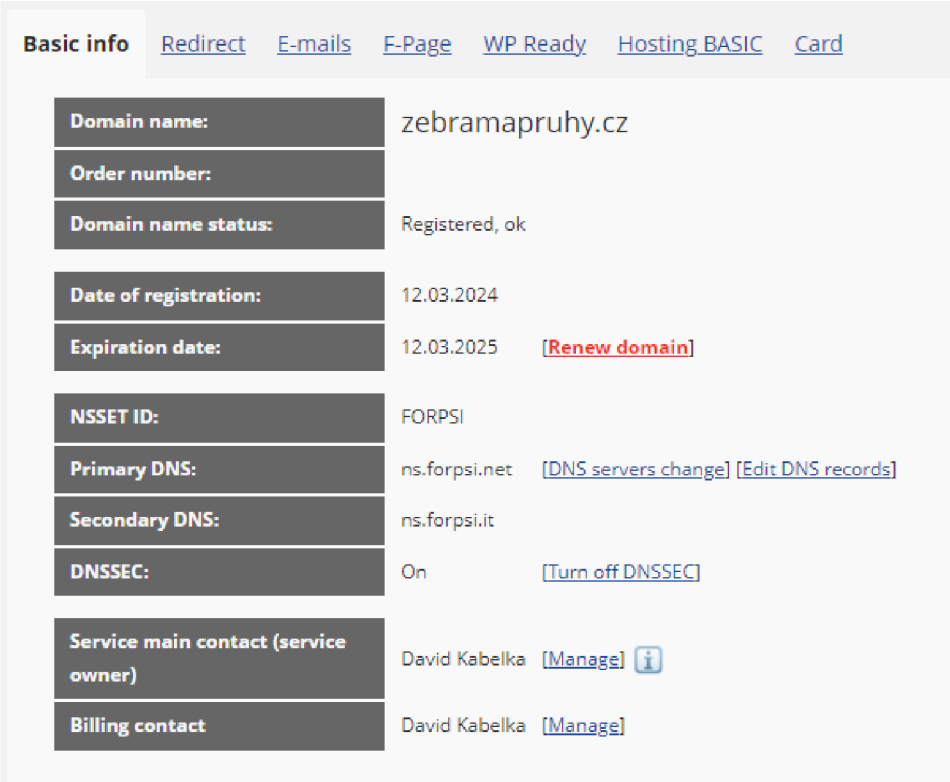
Posledním parametrem je definice, jaká image se má pro běh použít.

```
sudo docker run --name admin-postgres -e POSTGRES_PASSWORD=EwDJY6j3XcB6RB
-v postgres:/var/lib/postgresql/data -p 5432:5432 -d postgres
```

Obrázek 44: Příkaz docker run

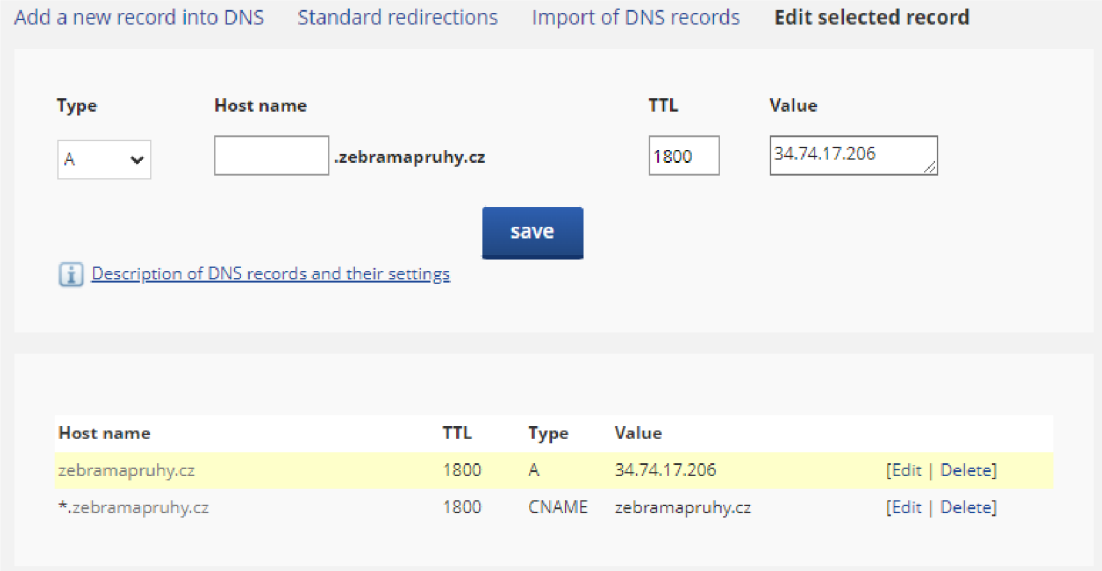
5.4 Nákup a nastavení domény

Nákup domény proběhl u českého registrátora forpsi.cz. Registrátor byl zvolen kvůli předchozím zkušenostem. Proces nákupu domény je velmi jednoduchý a stejný jako v typickém e-shopu. Po výběru doménového jména „zebramapruhy.cz“ byl zaplacen poplatek 175,45 Kč.



Obrázek 45: Rozhraní domény registrátora Forpsi

Doména je ve výchozím stavu přesměrovaná na univerzální stránku informující uživatele o její obsazenosti a o společnosti u které je zaregistrována. Bylo tedy třeba nastavit záznamy na DNS serveru. A záznam slouží k přesměrování domény na danou IP adresu, tento záznam byl změněn na adresu použitého virtuálního stroje v GCP. CNAME záznam zůstal beze změny a pouze přesměrovává všechny subdomény na hlavní doménu.



Obrázek 46: Prostředí úprav DNS záznamů

Po nastavení DNS záznamů je třeba vyčkat několik hodin, než dojde k propagaci do celé světové sítě. Zda již doména funguje cíleným způsobem lze zkontrolovat pomocí příkazu „nslookup“.

Obrázek 47: Odpověď příkazu nslookup

5.5 Výběr a konfigurace jednodeskového počítače

5.5.1 Provedení výběru

Prvním krokem provedení výběru bylo nalezení dostupných jednodeskových počítačů pomocí webového vyhledávače Google. Při hledání byly ustanoveny požadavky na výbavu daného počítače.

Vlastností je opravdu mnoho, je tedy nutné prioritizovat a zvolit ty důležité. Dle výsledného využití byl důraz kladen na přítomnost grafického HDMI výstupu, přítomnost bezdrátové síťové komunikace Wi-Fi, podpora operačního systému založeného na Linuxové bázi, cena, spotřeba energie a velikost komunitní podpory.

Vybrané vlastnosti byly sepsány do tabulky a u každého počítače, které byly jedny z možných kandidátů, byly přiděleny body v rozsahu 1 až 5, s tím, že 5 je nejvíce bodů a 1 nejméně. Body byly pro každý model sečteny dohromady a nejvyšší výsledek získal jednodeskový počítač „Raspberry Pi 4 Mode B“ s 29 body.

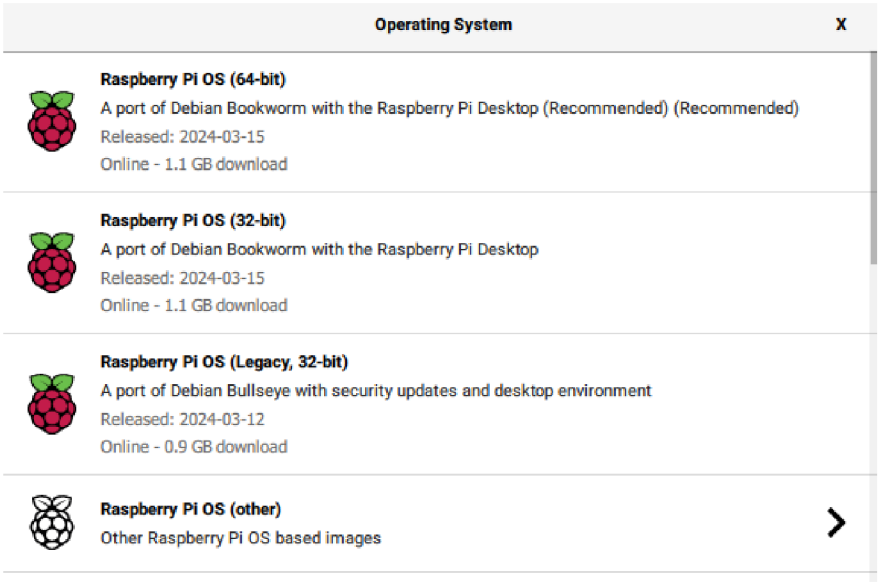
Criteria	Raspberry Pi 4 Model B	Raspberry Pi 5 Model B	Nvidia Jetson Nano	ODROID N2+
HDMI output	5	5	5	5
Wifi connectivity	5	5	3	1
Linux support	5	5	5	5
Community support	5	4	3	4
Cost	5	3	2	3
Power consumption	4	4	3	4
Suma bodů	29	26	21	23

Tabulka 4: Výběr jednodeskového počítače

5.5.2 Příprava operačního systému

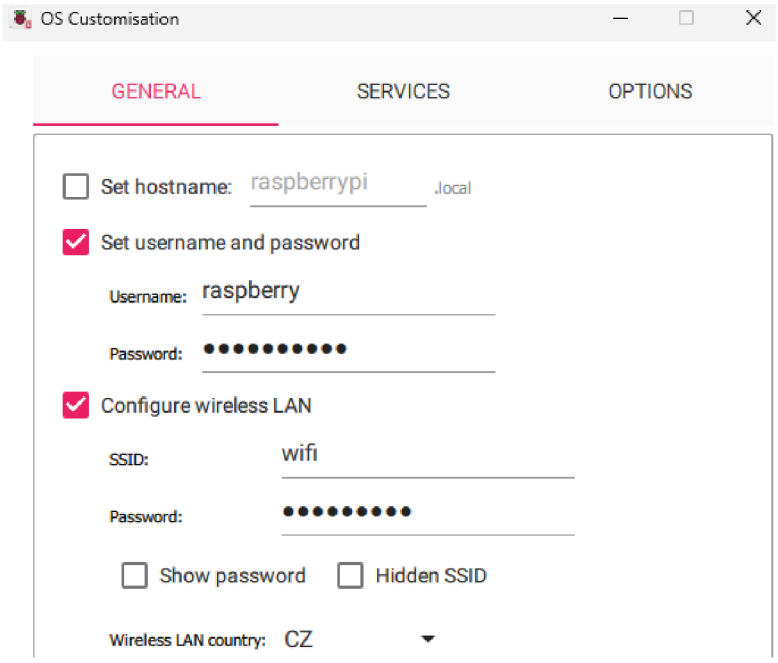
Konfigurace vybraného jednodeskového počítače započala stažením softwarového programu Raspberry Pi Imager v nejnovější verzi 1.8.5. Protože bude operační systém

pracovat s grafickým rozhraním, je zvolena verze „Raspberry Pi OS with Desktop“ v 64 bitové verzi. Tento systém je nainstalován na paměťovou MicroSD kartu.



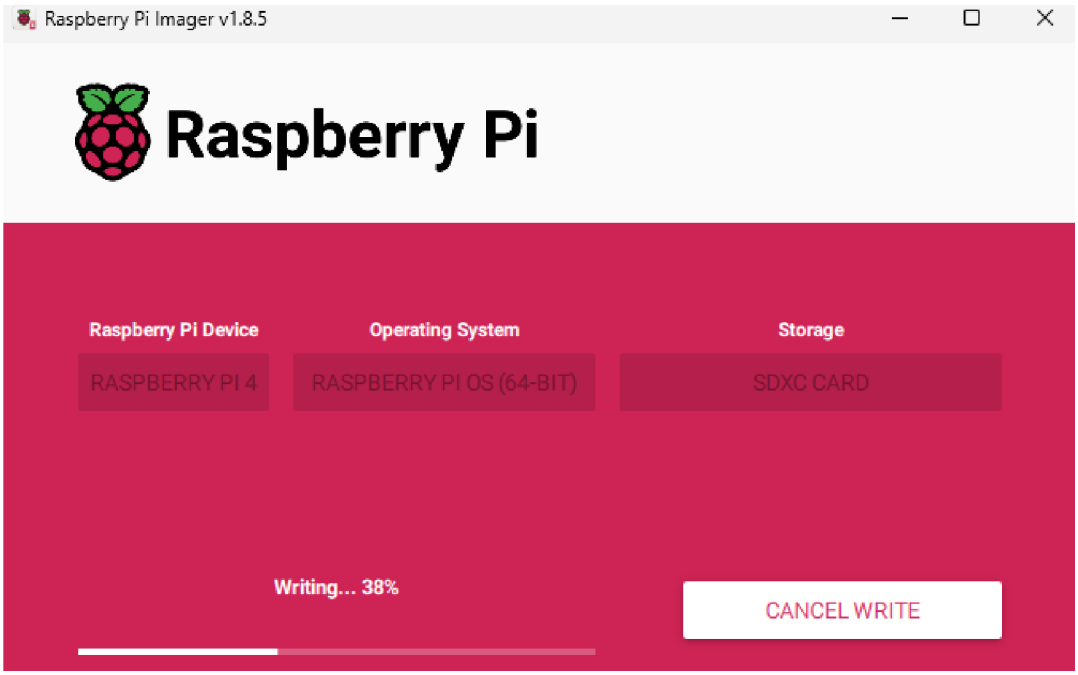
Obrázek 48: Raaspberry imager výběr OS

Pro ulehčení konfigurace lze pomocí klávesové zkratky CTRL+SHIFT+X vyvolat pokročilé menu, kde bylo nakonfigurováno uživatelské jméno a heslo profilu uživatele. Dále bylo nakonfigurováno Wi-Fi připojení a aktivováno SSH.



Obrázek 49: Pokročilá konfigurace RPi Imager

Samotné nahrání operačního systému na microSD kartu proběhlo pak v řádu několika minut a je možné jej ihned použít.



Obrázek 50: Průběh kopírování OS

5.5.3 Automatizovaný script

Koncové zařízení samo o sobě neumí vykonávat jakoukoliv automatizovanou činnost, bylo tak nutné vytvořit ovládací script a ten následně provozovat jako systémovou službu. Celkový proces automatizace byl rozdělen na menší části tak, aby hlavní script byl co nejjednodušší a jen spouštěl scripty navazující. Vše je tedy modulární a jednoduché pro další úpravy.

Skript funguje jako nekonečná smyčka pomocí „while do“, na konci smyčky je definováno čekání 10 sekund, poté se skript zase opakuje.

Na začátku scriptu bylo pomocí funkce xset nutné zakázat zhasínání displeje při neaktivitě, dále byl deaktivován spořič obrazovky. Grafické rozhraní operačního systému Debian podporuje integraci myši, pro potřeby zobrazování by však byla ikona myši rušivým elementem, a proto byla doinstalována utilita unclutter pomocí příkazu „DISPLAY=:0 unclutter – idle 0.5 -root &“, kdy po uplynutí poloviny sekundy dojde k zneviditelnění této ikony.

Pomocný skript pro změnu orientace obrazovky vždy nejprve vyhodnotí, zda je nutné zobrazení měnit na základě porovnání aktuálního konfiguračního souboru a nově nastavované hodnoty. Pokud není nutné, změna a hodnoty jsou stejné, skript se ukončí. Tímto postupem se podařilo zabránit nepříjemnému blikání obrazovky v případě opakovaného nastavování stejné hodnoty.

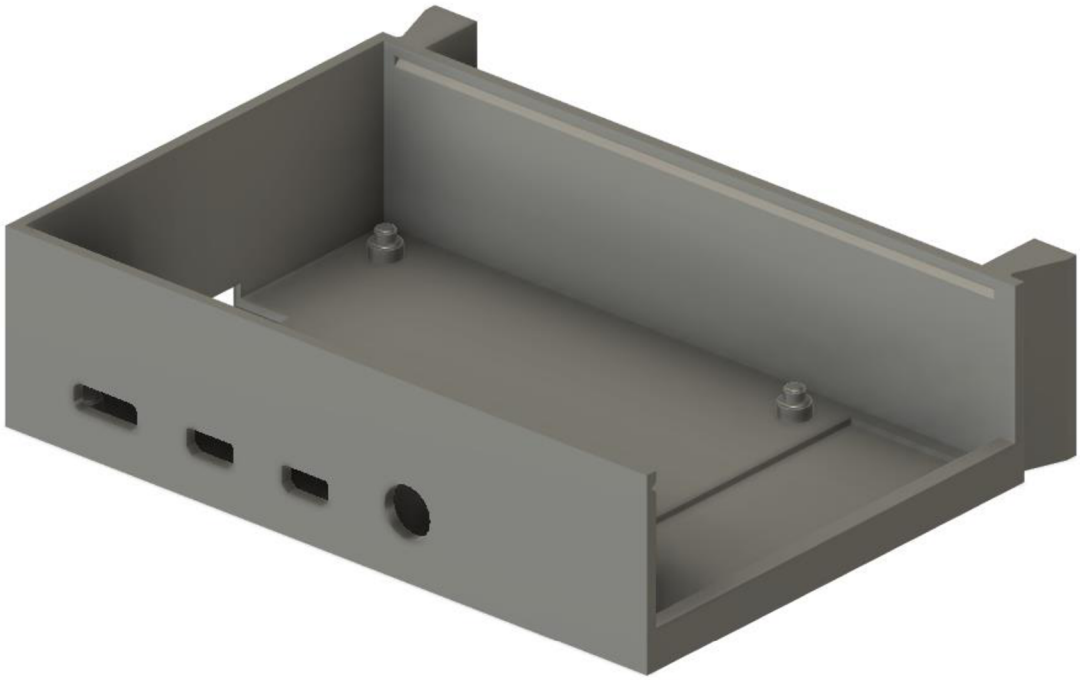
LCD obrazovka včetně jednodeskového zařízení má spotřebu cca 55–65 W. V případě více provozovaných zobrazovacích zařízení se v součtu jedná o nezanedbatelnou hodnotu. Proto je v rámci scriptu využito technologie HDMI-CEC, která umí komunikovat s obrazovkou pomocí HDMI a má možnost LCD obrazovku vypnout nebo zapnout. K tomu dochází, když script vyhodnotí, zda je v IS nastaven časový interval standby. V režimu standby je celkový odběr v rámci jednotek wattů, je tedy dosažena značná úspora např. v čase, kdy v prostoru, kde je řešení provozováno, nikdo není přítomen.

5.6 Šasi

Aby bylo možné uchytit jednodeskový počítač na LCD obrazovku, bylo k tomuto účelu vytvořeno šasi a uchycovací systém.

5.6.1 3D modelování

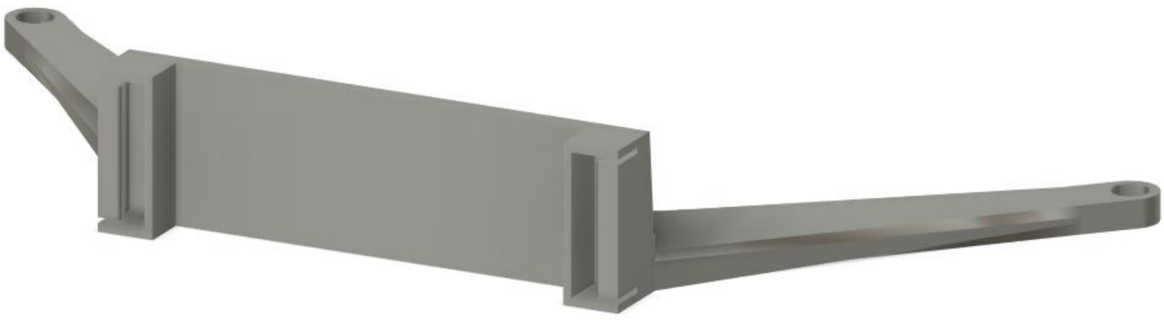
Jednotlivé 3D modely byly vymodelovány pomocí velice všestranného programu Autodesk Fusion, který nabízí základní i pokročilejší funkce. Předpokladem k modelování šasi pro jednodeskový počítač bylo pečlivé zaměření všech rozměrů. Každý model začal nejprve náčrtem, ze kterého byly následně extrudovány jednotlivé části.



Obrázek 51: 3D model šasi pro jednodeskový počítač

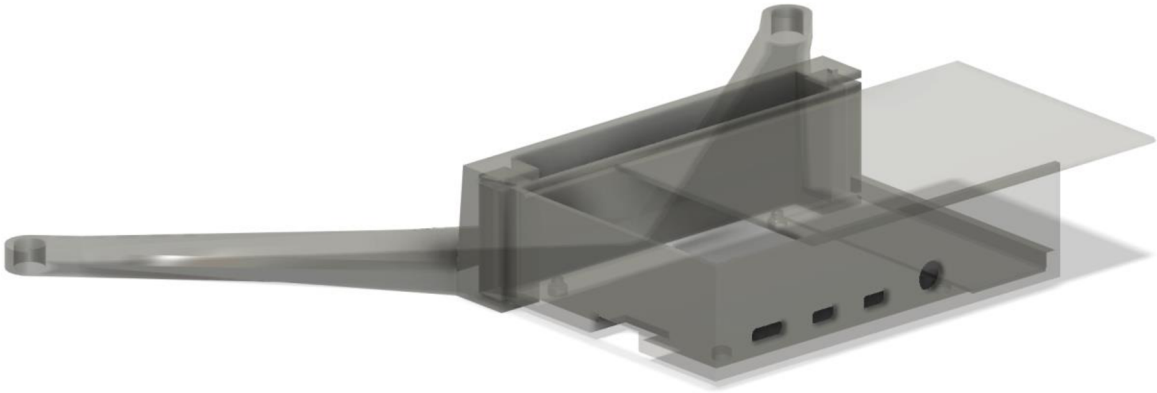
Pro základní tělo šasi bylo vytvořeno víko, které lze snadno nasadit a sundat dle potřeby, zabrání také nadměrnému výskytu prachu na zařízení. Bylo nutné vymodelovat otvory pro jednotlivé konektory. Na jedné ze stran byly vymodelovány úchyty, které následně slouží pro uchycení do VESA držáku.

Samotný VESA držák byl vymodelován pro standard 200x200, ale použity budou jen spodní otvory LCD obrazovky. Tento díl je určen na pevné a trvalé uchycení šrouby do obrazovek. K tomuto dílu lze pak nacvaknout tělo šasi.



Obrázek 52: 3D model FDMI

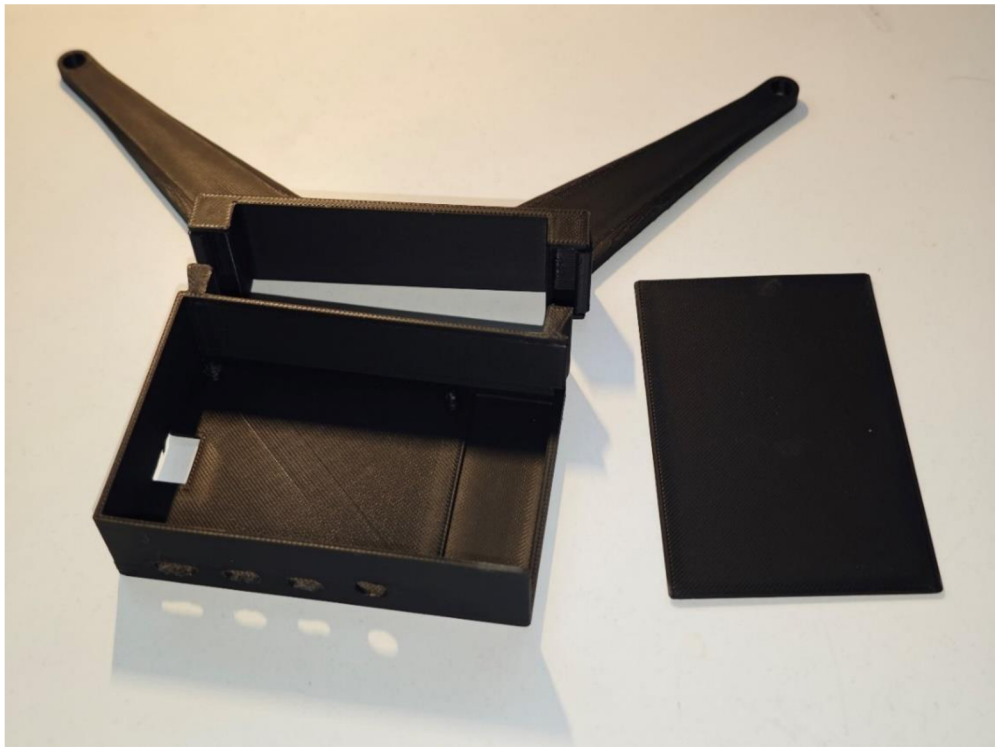
Z jednotlivých dílů byla složena sestava, která reprezentuje finální výrobek, pro ilustraci je použita určitá průhlednost, je tak snazší vidět jednotlivé detaily a představit si, jak pasují dohromady.



Obrázek 53: 3D pohled na složenou sestavu

5.6.2 3D tisk

3d tisk proběhnul na 3D tiskárně od společnosti Bambu Lab. Model tiskárny s názvem X1 – Carbon je velmi pokročilý a nabízí i použití čtyř materiálů v rámci jednoho tisku bez nutnosti uživatelského zásahu. Celý tisk za použití materiálu PLA trval cca 4 hodiny.

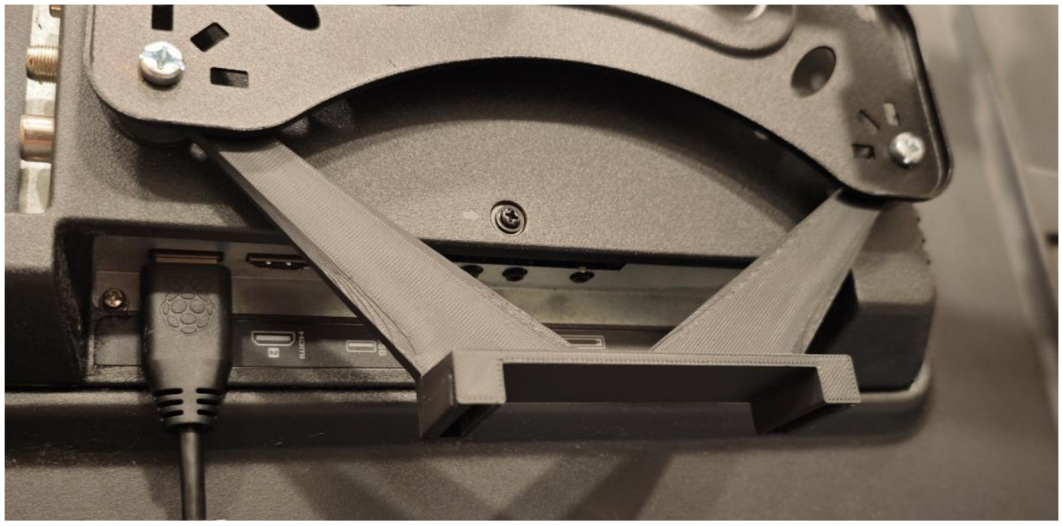


Obrázek 54: Vytištěné 3D modely

Po vytištění bylo třeba některé otvory lehce zbrousit, aby se jednodeskový počítač umístil bez použití větší námahy. Následně byly modely složeny dohromady a tvoří celek.

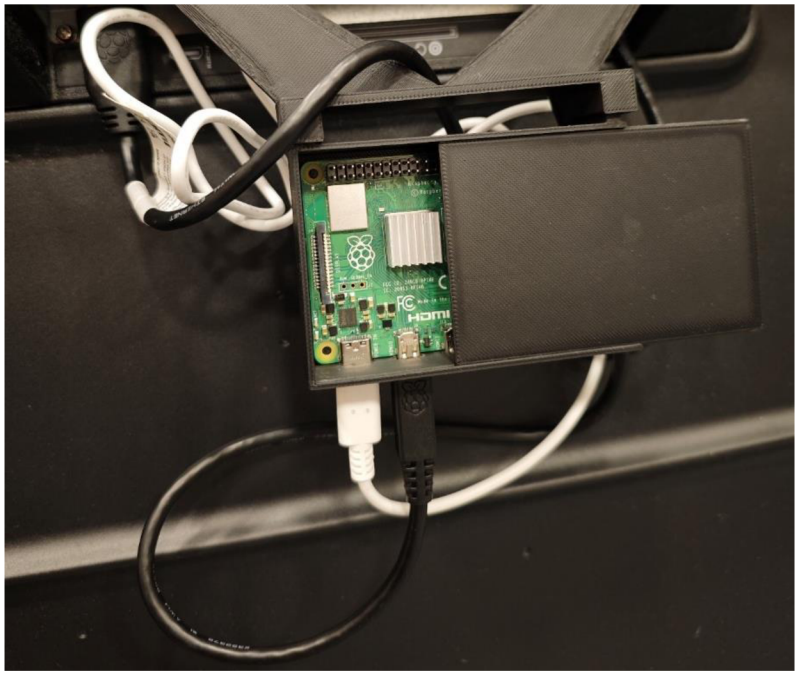
Obrázek 55: Raspberry Pi ve vytištěném šasi

Instalace na LCD obrazovku proběhla velmi jednoduše. Na místo, kde se obvykle instalují plastové podložky byl umístěn plastový díl a vše bylo uchyceno M6 šroubem se závitem.



Obrázek 56: Uchycený držák

Po uchycení držáku bylo Raspberry Pi uložené v šasi umístěno do příslušného místa, připojeny byly dva potřebné konektory – HDMI pro video výstup a USB C pro napájení. Na této konkrétní LCD obrazovce bylo třeba povolit HDMI-CEC, každý výrobce je jiný a v nějakých případech je nutná ruční aktivace.



Obrázek 57: Raspberry Pi v držáku

5.7 Možná budoucí vylepšení

V rámci každého projektu jsou možná nějaká vylepšení a budoucí vývoj. Protože je na realizaci vždy vyčleněna jen omezená doba, nelze do každého systému a projektu zapracovat vše. Prioritou byly základní funkce, ale lze přemýšlet o budoucnosti vývoje, kam směřovat a jaké funkce jsou prioritou.

5.7.1 Multitenantní přístup

Aktuálně je aplikace koncipována jako jeden systém pro jeden podnik. V případě, že by měla být aplikace nabízena jako software jako služba, bylo by nutné pro každý podnik vytvářet novou instanci databáze a aplikace. To lze vyřešit multitenantním přístupem, kdy aplikace funguje v jednom prostředí a na základě přihlašovaného uživatele nebo firmy zobrazí pouze data dané společnosti.

5.7.2 Dynamické obnovování

Pro lepší uživatelský zážitek je žádoucí, aby informace o stavu zařízení byly uživateli předkládány dynamicky nebo pravidelně za daný časový interval. Nebylo by tak nutné stránku s přehledem ručně aktualizovat.

5.7.3 SSL

V případě, že by mělo dojít ke komerčnímu nasazení, případně reálnému používání, je důležité, aby byla komunikace se serverem zašifrována pomocí SSL certifikátu a bylo možné použít HTTPS.

5.7.4 Pokročilé možnosti zobrazování, video, více obsahu, časování

Aby bylo celé řešení více flexibilní, bylo by přínosem, kdyby systém uměl zobrazovat více než jeden konkrétní obsah, například by mohl střídat mezi několika webovými odkazy, případně by také mohl přehrávat video. Další možné vylepšení v tomto spektru je možnost časování, kdy se jaký obsah má zobrazovat.

5.7.5 Komunikace s databází

Aktuálně aplikace komunikuje s databází přes externí IP adresu, mohou tak vznikat latence a odpovědi serveru mohou být delší. Aby byla možná komunikace aplikace a databáze napřímo interní sítí, je možné využít Docker Network.

6 Výsledky a diskuse

Cílem této práce bylo vytvořit informační systém umožňující vzdálenou správu obsahu, který je zobrazen na koncových zařízeních a dále administraci těchto zobrazovacích jednotek.

Výsledkem této diplomové práce je funkční informační systém, který je přístupný na zakoupené internetové doméně. Po autentizaci a autorizaci dává uživateli možnost získat přehled o množství spravovaných displejů zobrazujících vybraný obsah, dále o stavu koncových zařízeních, umožňuje horizontální nebo vertikální nastavení zobrazení obsahu dle umístění obrazovky, restartování koncového zařízení a nastavení času pro nižší spotřebu energie. Koncové zařízení má možnost komunikovat s informačním systémem, stahovat si aktuální konfiguraci a reportovat svůj stav. Rozhraní administrátora umožňuje přidávání nových a odstranění stávajících, již existujících zařízení z informačního systému.

Vyvinutý informační systém je provozován v cloudovém řešení od Google Cloud Platform, kde je pro běh využit virtuální server s operačním systémem Debian.

Samotný informační systém je provozován pomocí kontejnerizačního nástroje Docker. Po zkompilování zdrojového kódu aplikace IS do spustitelného jar souboru je pomocí vytvořeného automatizačního scriptu vytvořena image, která je následně v Dockeru spuštěna.

Databázový systém je také řešen a provozován v kontejneru a samotná databázová data jsou pak ukládána na persistentní úložiště.

Pro vyvinutý systém byla popsána možná budoucí vylepšení, které by v případě komercializace byly nutné implementovat nebo zajistit lepší konkurenceschopnost na trhu.

Výsledný vytvořený 3D model držáku a šasi má v částech určených pro spojení vůle a značné tolerance. Při vývoji nějakých fyzických komponent lze toto odstranit procesem prototypování a vytvořit tak několik iterací výsledného produktu, který již bude mít odstraněné nedostatky. Upínací mechanismus lze vyřešit jinými způsoby, například nasouváním.

Za celý proces od analýzy přes návrh až po samotné programování a nasazení IS na produkčním prostředí autor získal mnoho nových jak teoretických, tak praktických zkušeností a znalostí, které budou přínosem v osobním i profesním životě.

7 Závěr

Teoretická část práce vytvořila povědomí týkající se informačních systémů, programovacího jazyka Java, technologií pro vývoj webových aplikací za použití nástrojů Spring Boot a Vaadin Framework, popsány jsou typy databázových systémů, cloudové služby a dostupné typy možných řešení, internetové domény, technologie Docker a kontejnerizace, možnosti 3D tisku a standardy uchycování obrazovek.

V praktické části této práce byla nejdříve provedena analýza požadavků na vyvíjený informační systém. Vytvořeny byly případy použití a dva nejrozsáhlejší byly podrobně popsány včetně scénářů použití a jejich toků. Dále byl zpracován grafický návrh jednotlivých uživatelských rozhraní a také návrh použitých tříd.

Byl popsán postup vývoje, nastavení programovacího prostředí, použitých nástrojů a konfigurace systémů. Představeny jsou jednotlivé naprogramované funkce informačního systému, propojení jednotlivých komponent, vytvoření grafického rozhraní, konfigurace serveru, databáze a nástrojů na kontejnerizaci.

Také byl popsán proces návrhu 3D modelu sloužícího k uchycení jednodeskového počítače k LCD obrazovce za využití FDMI standardu a následně byl tento model vytištěn za pomoci 3D tiskárny. Vytištěný model byl nainstalován a vyzkoušen na LCD obrazovce.

8 Seznam použitých zdrojů

[1] R. Daniel, „Informační systémy - elektronická skripta,“ 4 5 2011. Online]. Available: https://homel.vsb.cz/~dan11/rd_is_skripta.htm. [Přístup získán 19 10 2023].

[2] Z. Molnár, Podnikové informační systémy, Praha: Vydavatelství ČVUT, 2004.

[3] B. Koďousková, „INFORMAČNÍ SYSTÉMY V KOSTCE: ERP, CRM, IMPLEMENTACE,“ 11 10 2021. [Online]. Available: https://www.rascasone.com/cs/blog/informacni-systemy-erp-crm-implemetace. [Přístup získán 11 10 2023].

[4] o. boy, „Types of Information Systems,“ 9 4 2014. [Online]. Available: https://infs-u094046.blogspot.com/2014/04/types-of-information-systems_8.html. [Přístup získán 20 11 2023].

[5] J. A. HALL, Accounting, Mason, USA: Rob Dewey, 2008.

[6] „9 Key Manufacturing ERP Modules to Consider for Your Next ERP System,“ 15 1 2024. [Online]. Available: https://www.linkedin.com/pulse/9-key-manufacturing-erp-modules-consider-your-next-ny3lc?trk=article-ssr-frontend-pulse_more-articles_related-content-card. [Přístup získán 17 10 2023].

[7] TechSpawn Solutions, „How Do Custom Modules in Odoo Enhance ERP Integration Capabilities?,“ 17 8 2023. [Online]. Available: https://medium.com/@ERPserviceprovider/how-do-custom-modules-in-odoo-enhance-erp-integration-capabilities-2ede56530127. [Přístup získán 17 10 2023].

[8] V. Notes, „Information System: Intro. & Development Life Cycle,“ 3 2 2022. [Online]. Available: https://medium.com/javarevisited/information-system-intro-overview-317d2cc0a97a. [Přístup získán 28 11 2023].

[9] E. J. Kendall a K. E. Kendall, System Analysis and Design, New Jersey: Pearson Publishing, 2011.

[10] P. Niththiyanantham, „Introduction to Java,“ 5 6 2021. [Online]. Available: https://medium.com/linkit-intecs/introduction-to-java-45ace43d77ca. [Přístup získán 2 8 2024].

[11] „The top programming languages,“ GitHub, Inc, 2022. [Online]. Available: https://octoverse.github.com/2022/top-programming-languages. [Přístup získán 05 01 2024].

[12] B. Krebs, „Java Platform and Java Community Process Overview,“ 30 3 2017. [Online]. Available: https://auth0.com/blog/java-platform-and-java-community-process-overview/. [Přístup získán 07 01 2024].

[13] R. Sao, „What is JDK?,“ 2018. [Online]. Available: https://www.quora.com/What-is-JDK/answer/Raghav-Sao. [Přístup získán 10 01 2024].

[14] Oracle, „Java Platform, Standard Edition 22 Reference Implementations,“ Oracle, [Online]. Available: https://jdk.java.net/java-se-ri/22. [Přístup získán 05 01 2024].

[15] epam, „difference between Java EE and Spring: which framework is the best choice?,“ 24 2 2024. [Online]. Available: https://anywhere.epam.com/en/blog/spring-vs-java-ee. [Přístup získán 02 03 2024].

[16] j. V, „Is Java ME still relevant? Exploring the Continued Significance of Java Micro Edition,“ 23 8 2023. [Online]. Available: https://medium.com/@etherservices.vimalraj/is-java-me-still-relevant-exploring-the-continued-significance-of-java-micro-edition-c1a0ff297b1b. [Přístup získán 02 01 2024].

[17] S. Naik, „System Design Blog Series - JVM,“ 29 10 2022. [Online]. Available: https://sivanaikk0903.medium.com/system-design-blog-series-jvm-94d168ed5a22. [Přístup získán 15 01 2024].

[18] P. A. Parit, „My first blog on JVM — JAVA virtual machine,“ 5 11 2022. [Online]. Available: https://medium.com/@prasadparit006/my-first-blog-on-jvm-java-virtual-machine-e20bf10e881a. [Přístup získán 16 01 2024].

[19] Y. Kods, „Introduction to Spring and Spring Boot.,“ 14 7 2023. [Online]. Available: https://medium.com/nerd-for-tech/introduction-to-spring-and-spring-boot-fdf1479691e. [Přístup získán 20 12 2023].

[20] N. Fränkel, „Why I (still) love Vaadin,“ 7 6 2020. [Online]. Available: https://nfrankel.medium.com/why-i-still-love-vaadin-e6dcfd99d92e. [Přístup získán 15 12 2023].

[21] S. Ekblad, „What's the deal with Vaadin add-ons?,“ 31 1 2023. [Online]. Available: https://dev.to/samiekblad/whats-the-deal-with-vaadin-add-ons-2f1n. [Přístup získán 05 01 2024].

[22] M. Laurenčík, SQL: podrobný průvodce uživatele., Praha: Grada Publishing, 2018.

[23] Tutort Academy, „What is an Object-Oriented Database?,“ 10 10 2023. [Online]. Available: https://medium.com/@-TutortAcademy/what-is-an-object-oriented-database-95763c5e374d. [Přístup získán 22 12 2023].

[24] K. Supe, „When to Use a NoSQL Database,“ 17 7 2023. [Online]. Available: https://memgraph.com/blog/when-to-use-a-nosql-database. [Přístup získán 20 12 2023].

[25] L. Santos, „What is Docker Used For? A Docker Container Tutorial for Beginners,“ 09 12 2020. [Online]. Available: https://www.freecodecamp.org/news/what-is-docker-used-for-a-docker-container-tutorial-for-beginners/. [Přístup získán 02 03 2024].

[26] Docker Inc. , „Use containers to Build, Share and Run your applications,“ Docker Inc. , [Online]. Available: https://www.docker.com/resources/what-container/. [Přístup získán 02 03 2024].

[27] P. Srivastav, „Learn to build and deploy your distributed applications easily to the cloud with Docker,“ [Online]. Available: https://docker-curriculum.com/. [Přístup získán 02 03 2024].

[28] „Docker overview,“ Docker Inc., [Online]. Available: https://docs.docker.com/get-started/overview/. [Přístup získán 12 01 2024].

[29] A. Gamela a R. Figueiredo, „Podman vs Docker: What are the differences?,“ 14 9 2023. [Online]. Available: https://www.imaginarycloud.com/blog/podman-vs-docker/#:~:text=Docker%20uses%20a%20daemon%2C%20an,does%20not%20need%20the%20mediator.. [Přístup získán 02 03 2024].

[30] A. Kashyap, „Docker vs Podman: A New Era in Secure Orchestration,“ 19 12 2023. [Online]. Available: https://levelup.gitconnected.com/docker-vs-podman-a-new-era-in-secure-orchestration-957ea2123098. [Přístup získán 20 01 2024].

[31] Z. Hira, „Kubernetes VS Docker Swarm – What is the Difference?,“ 5 7 2022. [Online]. Available: https://www.freecodecamp.org/news/kubernetes-vs-docker-swarm-what-is-the-difference/. [Přístup získán 02 03 2024].

[32] S. Ranger, „What is cloud computing? Everything you need to know about the cloud explained,“ 25 2 2022. [Online]. Available: https://www.zdnet.com/article/what-is-cloud-computing-everything-you-need-to-know-about-the-cloud/. [Přístup získán 02 02 2024].

[33] J. MŮČKA, „IaaS, PaaS a SaaS aneb V čem se liší služby „as a Service”,“ 29 09 2021. [Online]. Available: https://www.master.cz/blog/iaas-paas-a-saas-aneb-v-cem-se-lisi-sluzby-as-a-service/. [Přístup získán 18 01 2024].

[34] L. Davis a R. Watts, „What Is Cloud Computing? The Ultimate Guide,“ 19 10 2022. [Online]. Available: https://www.forbes.com/advisor/business/what-is-cloud-computing/. [Přístup získán 02 05 2024].

[35] THE INVESTOPEDIA TEAM, „What is Cloud Computing? Pros and Cons of Different Types of Services,“ Investopedia, 22 12 2023. [Online]. Available: https://www.investopedia.com/terms/c/cloud-computing.asp. [Přístup získán 05 02 2024].

[36] D. Daniels, „What Is Hybrid Cloud? Benefits and Disadvantages,“ 5 12 2019. [Online]. Available: https://blog.gigamon.com/2019/12/05/what-is-hybrid-cloud-advantages-and-disadvantages/. [Přístup získán 04 02 2024].

[37] P. Patil a C. B. Mallick, „What Is Cloud Computing? Definition, Benefits, Types, and Trends,“ 9 2 2022. [Online]. Available: https://www.spiceworks.com/tech/cloud/articles/what-is-cloud-computing/. [Přístup získán 04 02 2024].

[38] K. Shmuely, „Cloud Cost Management: A Complete Guide,“ [Online]. Available: https://granulate.io/blog/cloud-cost-management-key-models-and-tools-in-2023/. [Přístup získán 06 02 2024].

[39] Triskele Labs, [Online]. Available: https://www.triskelelabs.com/blog/cloud-cyber-attacks-the-latest-cloud-computing-security-issues. [Přístup získán 05 02 2024].

[40] PeoplActive, „Phishing Attacks Targeting Cloud Services and SaaS Platforms,“ 8 12 2023. [Online]. Available: https://peoplactive.com/blog/the-growing-menace-of-phishing-attacks-on-cloud-services-and-saas/. [Přístup získán 05 02 2024].

[41] M. Ramakrishnan, „Should You Adopt Cloud Computing for Your Business? What are its Disadvantages?,“ 16 12 2022. [Online]. Available: https://emeritus.org/blog/technology-disadvantages-of-cloud-computing/. [Přístup získán 07 02 2024].

[42] C. Team, „What Is a Domain Name, and How Does It Work?,“ 10 06 2021. [Online]. Available: https://www.codecademy.com/resources/blog/what-is-a-domain-name/. [Přístup získán 28 12 2023].

[43] 2. C. AV, „How to Choose the Right Digital Signage Software for Your Business,“ 10 03 2023. [Online]. Available: https://21stcenturyav.com/digital-signage-software/. [Přístup získán 17 12 2023].

[44] B. Roberts, „Why Your Business Should Be Using Raspberry Pi Single-Board Computers,“ [Online]. Available: https://www.bairesdev.com/blog/raspberry-pi-single-board-computers/. [Přístup získán 08 02 2024].

[45] Kalitut, „Raspberry Pi Connectors and Components,“ 17 5 2020. [Online]. Available: https://kalitut.com/connectors-and-components-of-raspberry/. [Přístup získán 09 02 2024].

[46] E. Upton, „Introducing: Raspberry Pi 5!,“ 28 9 2023. [Online]. Available: https://www.raspberrypi.com/news/introducing-raspberry-pi-5/. [Přístup získán 09 02 2024].

[47] H. Fowle, „A beginner’s guide to single-board computers,“ 16 1 2024. [Online]. Available: https://www.student-circuit.com/blog/a-beginners-guide-to-single-board-computers/. [Přístup získán 19 02 2024].

[48] K. McAleer, „RASPBERRY PI 5,“ 27 9 2023. [Online]. Available: https://www.kevsrobots.com/blog/raspberry_pi_5.html. [Přístup získán 09 02 2024].

[49] AutoPi.io, „What Can Developers Create with Raspberry Pi? Explore Now!,“ 23 1 2024. [Online]. Available: https://www.autopi.io/blog/what-is-raspberry-pi/. [Přístup získán 10 02 2024].

[50] R. Ye, „Additive Manufacturing vs Subtractive Manufacturing: In-depth Comparison & Differences,“ 24 4 2023. [Online]. Available: https://www.3erp.com/blog/additive-vs-subtractive-manufacturing/. [Přístup získán 18 12 2023].

[51] L. Gregurić, „3D Printing for Beginners: How to Get Started with FDM,“ 28 11 2023. [Online]. Available: https://all3dp.com/2/3d-printing-for-beginners-all-you-need-to-know-to-get-started/. [Přístup získán 20 12 2023].

[52] V. E. S. Association, „VESA FLAT DISPLAY MOUNTING INTERFACE STANDARD (for Flat Panel Monitors/Displays/Flat TVs),“ 16 1 2006. [Online]. Available: https://www.maxrev.de/files/2011/09/fdmiv1r1.pdf. [Přístup získán 15 11 2023].