FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání diplomové práce

Název:	Modul pro generování profesních životopisů do firemního informačního systému
Student:	Bc. Miroslav Halamka
Vedoucí:	Ing. Marek Suchánek, Ph.D. et Ph.D.
Studijní program:	Informatika
Obor / specializace:	Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2024/2025

Pokyny pro vypracování

Cílem diplomové práce je navrhnout a implementovat modul do existujícího firemního informačního systému (IS) pro vytváření profesních životopisů (CV) zaměstnanců. Tento modul postavený na Java technologiích bude s existujícím IS komunikovat prostřednictvím API. Tímto způsobem bude získávat a agregovat informace o zaměstnancích za účelem vygenerování CV jako souboru v editovatelném formátu. Výsledný dokument musí respektovat firemní grafický manuál a zohlednit případné manuální úpravy. V rámci práce bude postupováno v souladu s metodami softwarového inženýrství:

- Stručně popište existující firemní informační systém, jeho architekturu a další relevantní části pro tvorbu CV modulu.
- Sestavte katalog funkčních a nefunkčních požadavků. Připravte dopadovou analýzu – požadavky na úpravu stávajících částí systému nebo jiné systémy ve firmě.
- Proveďte stručnou rešerši integračních řešení v kontextu stanovených požadavků.
- Na základě požadavků, analýzy a rešerše navrhněte a popište vlastní řešení.
- Implementujte CV modul dle návrhu, řádně jej zdokumentujte, otestujte a připravte pro nasazení do firemního prostředí.
- Zhodnoťte výsledné řešení a navrhněte další možný rozvoj.

Elektronicky schválil/a Ing. Michal Valenta, Ph.D. dne 21. prosince 2023 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Modul pro generování profesních životopisů do firemního informačního systému

Bc. Miroslav Halamka

Katedra softwarového inženýrství
Vedoucí práce: Ing. Marek Suchánek, Ph.D. et Ph.D.

9. května 2024

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. Dále prohlašuji, že jsem s Českým vysokým učením technickým v Praze uzavřel dohodu, na jejímž základě se ČVUT vzdalo práva na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona. Tato skutečnost nemá vliv na ustanovení § 47b zákona č. 111/1998 Sb., o vysokých školách, ve znění pozdějších předpisů.

V Praze dne 9. května 2024

České vysoké učení technické v Praze
Fakulta informačních technologií

Odkaz na tuto práci

Halamka, Miroslav. *Modul pro generování profesních životopisů do firemního informačního systému*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2024.

Abstrakt

Tato diplomová práce se zaměřuje na rozšíření firemního systému o modul pro vytváření profesních životopisů v upravitelném formátu. Důraz je kladen na začlenění nového modulu do ekosystému firmy. Generátor je implementován jako Spring aplikace s využitím knihovny Apache POI. Kromě toho je rozšířeno uživatelské rozhraní firemního systému a vytvořena integrace s generátorem. Rozšíření umožňuje vytvářet životopisy ve formátu DOCX skrz formulář uživatelského rozhraní. Uživatel má možnost vygenerovat životopis v různých jazykových verzích a s různými orientacemi. Výsledný dokument respektuje stanovenou firemní šablonu a nabízí možnost estetické úpravy po jeho vygenerování.

Klíčová slova Životopis, Informační systém, DOCX, Spring, .NET, Angular

Abstract

This thesis focuses on extending the company's system with a module designed for creating professional resumes in an editable format. Special attention is given to the integration of the new module into the company's ecosystem. The resume generator is developed as a Java application utilising the Apache POI library. Furthermore, new screens have been incorporated into the system's user interface, along with integration with the generator. The user is able to generate a resume in multiple language variations and orientations. The final document adheres to the prescribed corporate template and is open to aesthetic adjustments post-generation.

Keywords Resume, Information System, DOCX, Spring, .NET, Angular

Obsah

Seznam obrázků

Seznam zdrojových kódů

Úvod

Profesní životopisy hrají klíčovou roli v procesu vytváření nabídek pro zákazníky. Skrze jejich estetickou formu lze efektivněji a atraktivněji prezentovat znalosti zaměstnanců a tak získat potřebnou výhodu nad konkurencí. Kvalitní životopis obsahuje aktuální a relevantní data vzhledem k nabídce a ctí šablonu využívanou ve firmě. S růstem firmy přibývají nároky na správu životopisů.

Cílem diplomové práce je navrhnout a implementovat modul do existujícího firemního informačního systému pro vytváření profesních životopisů zaměstnanců v editovatelném formátu. K dosažení cíle práce je provedena analýza životního cyklu životopisu ve firmě Profinit, jeho textové formy, interního systému Profis a také je proveden sběr požadavků na rozšíření. Práce obsahuje analýzu Java knihoven generující dokumenty formátu DOCX a rešerši integračních řešení. Dále je vytvořena dopadová analýza na ostatní části firemního informačního systému.

V návaznosti na analýzu je navržen modul generátoru a také rozšíření ostatních systémů ve firmě, jako například nutné úpravy databáze nebo přídání nových integrací a obrazovek do systému Profis. V rámci implementace je vytvořen Java generátor, jeho integrace do systému Profis a obrazovka tvorby životopisu. Po celý vývojový cyklus jsou prováděny schůzky se zainteresovanými osobami společnosti Profinit pro zajištění snadné integrace nových funkcionalit do firemního ekosystému.

Po dokončení implementace následuje jednotkové testování a statická analýza kódu Java generátoru a uživatelské testování celého procesu tvorby profesního životopisu. Nakonec je proveden návrh možných rozšíření systému a obrazovek, která byla objevena na společných schůzkách.

Kapitola 1

Analýza

V této části je provedena analýza procesu vytváření, úprav a odstraňování životopisů, představen interní systém Profis a proveden sběr požadavků na nový modul s jejich následnou analýzou. Dále je zpracována analýza architektury nového modulu, rešerše integračních řešení, analýza formátu DOCX a knihoven pracujících s tímto formátem. Nakonec je provedena dopadová analýza s cílem identifikovat dopady nového modulu na ostatní části systému.

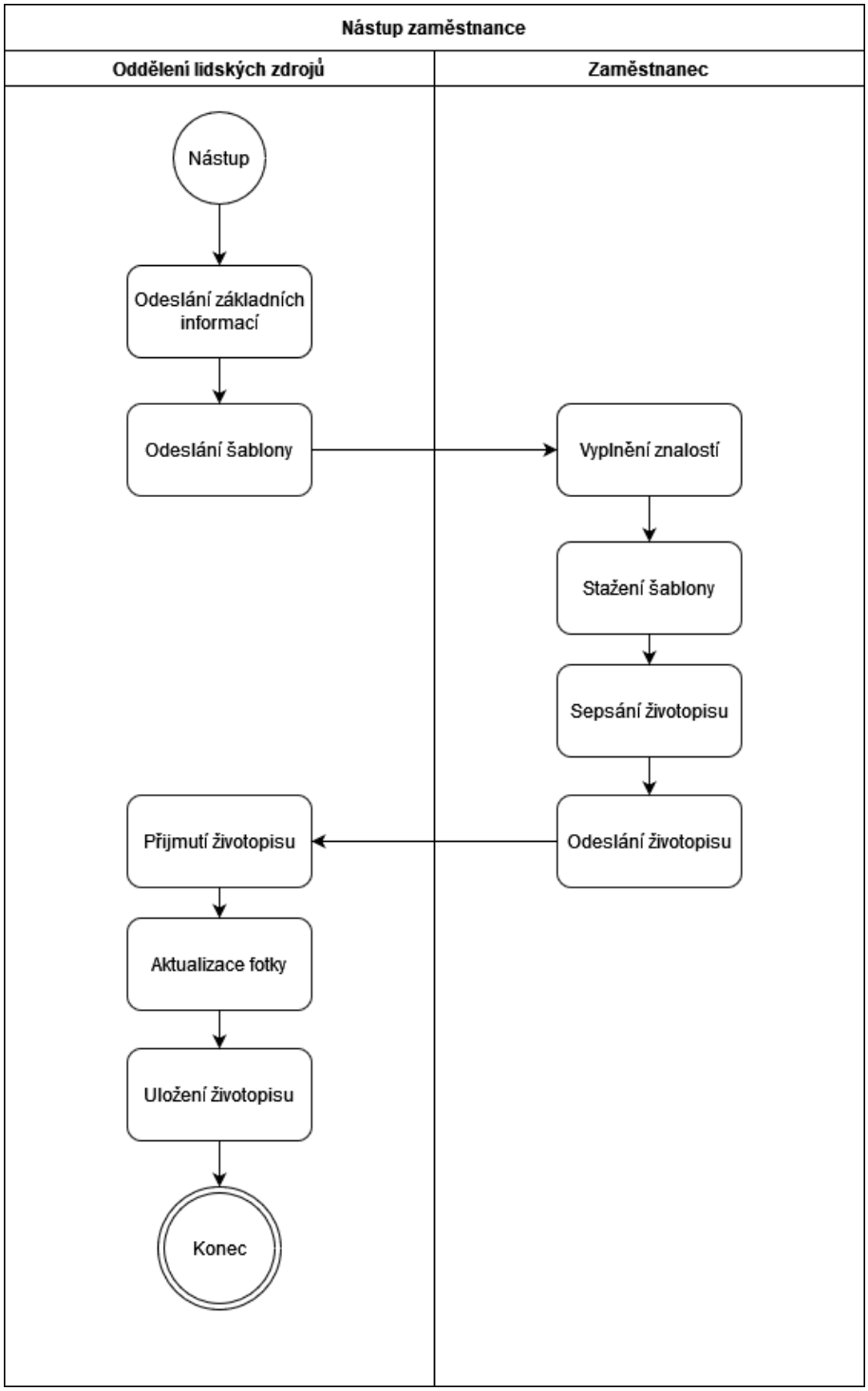
1.1 Aktuální stav

Prvním krokem je provedení analýzy současného stavu ve firmě Profinit. Tato analýza se zaměřuje na identifikaci procesů, které má nový modul zjednodušit, analýzu formátu životopisu a na související systém Profis, do kterého bude modul integrován a z něhož budou získávána data. Informační systém Profis je interní systém, který je přístupný skrze firemní virtuální privátní síť (VPN) a poskytuje omezený přístup přes internetové rozhraní. Systém je přístupný pouze autorizovaným uživatelům.

1.1.1 Životní cycklus životopisu

O agendu profesních životopisů se v současné době stará oddělení lidských zdrojů. Většina úkonů je dělána manuálně, kdy jednotlivé úkony dělá buď zaměstnanec, nebo pracovník oddělení. Největším procesem je proces náboru nového zaměstnance, kdy je vytvořen zcela nový životopis. Dalšími procesy jsou průběžné aktualizace životopisů, může se jednat o novou šablonu, fotografii pracovníka, ale také přidání nového projektu či technologie. Životopisy jsou po vytvoření uloženy pracovníkem oddělení lidských zdrojů do interního Apache Subversion úložiště (SVN), kde každý zaměstnanec disponuje svoji složku, ve které se nachází jeho aktuální životopisy ve všech variantách. Naznačený proces nástupu zaměstnance lze vidět na obrázku 1.1.

Každý zaměstnanec má minimálně dvě varianty životopisu – český a anglický. Zkušenější zaměstnanci mají i varianty dle pozice na kterou jsou nabízeni, například životopis pro práci analytika, vývojáře, vedoucího týmu a jiné. V současné době zde neexistuje žádná forma autorizace, proto každý vidí životopis každého, což přináší klady a zápory. Pro každého zákazníka je na síťovém



Obrázek 1.1: Diagram nástupu zaměstnance.

disku vytvořena složka, kam jsou ukládány životopisy zaměstnanců pracujících na projektu daného zákazníka.

Při tvorbě nové nabídky je nutné, aby zaměstnanci, kteří jsou součástí nabídky, aktualizovali svůj životopis v aktuálně využívané šabloně. Po ukončení pracovního poměru je nezbytné provést vymazání všech životopisů příslušného zaměstnance.

Grafický styl životopisu se postupem času mění, na grafické podobě pracuje marketingové oddělení v kooperaci s oddělením lidských zdrojů. Výstupem je ukázkový DOCX dokument.

1.1.2 Analýza textového formátu životopisu

Životopis zaměstnance společnosti Profinit se skládá z několika částí. V hlavičce dokumentu se nachází základní údaje zaměstnance – jméno, pracovní pozice a fotografie. První stránka životopisu se skládá z několika částí, pod které uživatel vkládá relevantní informace. Kategorie se skládají z výčtu informací s odlišnými styly. V šabloně se nacházejí následující sekce:

- Dovednosti – jedná se o soupis měkkých dovedností zaměstnance.
- Jazyky – jedná se o soupis jazyků s úrovní, kterou daná osoba mluví.
- Vzdělání – obsahuje nejvyšší dosažené vzdělání.
- Školení a certifikace – obsahuje seznam školení a certifikací, které zaměstnanec absolvoval.
- Technologie – je soupis technologií, ve kterých dotyčný dokáže pracovat.
- Domény – jsou oblasti, kde zaměstnanec pracoval a ovládá jejich prostředí.

Následující stránky životopisu obsahují záznamy pracovních zkušeností. Tyto záznamy jsou ve formě tabulek, které obsahují informace o názvu projektu, časovém období, během kterého byl zaměstnanec na projektu angažován, jeho rolích, prováděných aktivitách, použitých technologiích a popisu projektu. Část životopisu lze vidět na obrázku 1.2, celou šablonu životopisu lze vidět v příloze C.

1.1.3 Interní systém Profis

Informační systém Profinitu sloužící pro zajištění mnoha agend různých oblastí fungování firmy, zejména pro finanční, sales, backoffice a marketingové oddělení. Systém je využíván pro evidenci práce, dovolených a dalších agend, které využívají všichni zaměstnanci. Nedílnou součástí systému je integrace na další interní systémy Profinitu a poskytování dat pro reportingovou platformu. Systém je postaven jako webová aplikace, která běží v interní a externí síti. Využívá komunikaci prostřednictvím webových služeb a s pomocí frameworku objektově relačního mapování (ORM) je napojena na MSSQL databázi. Nad tímto systémem je vybudován interní reporting, který využívá technologii SQL Server Reporting Services (SSRS) a jednotlivé reporty jsou

Obrázek 1.2: Výřez ze šablony profesního životopisu.

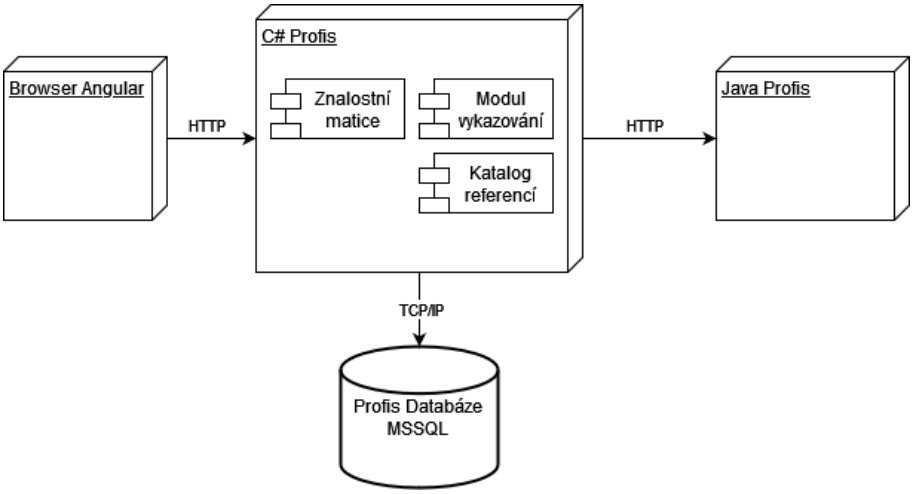
dále zobrazeny v Profisu. V této kapitole jsou rozebrány části systému, ze kterých nový modul generování životopisů může čerpat data a integrovat se na ně. Jednotlivé moduly jsou napsány v jazyce Java a frameworku Spring, nebo v jazyce C# a frameworku .NET. Uživatelské rozhraní je napsáno v Angularu a ASP.NET. Rozhraní komunikuje s .NET částí systému, která zaštiťuje veškerou interakci s uživatelem. Zobrazení systému Profis s relevantními moduly lze vidět na obrázku 1.3. Při navrhování uživatelského rozhraní je nutné dodržovat stylové prvky a komponenty existující stránky, takže nové obrazovky působí jednotně a ladí s celkovým vzhledem uživatelského rozhraní. Výřez současného uživatelského rozhraní je zobrazen na obrázku 1.4. V současné době je systém přepisován do .NET 8 a Angularu. Z tohoto důvodu nelze přepoužít staré komponenty a integrace, ale je nutné využít API Profisu využívající starší technologie. Po přepisu potřebných komponent bude změněno volání z REST rozhraní na rozhraní jednotlivých tříd.

1.1.3.1 Modul vykazování

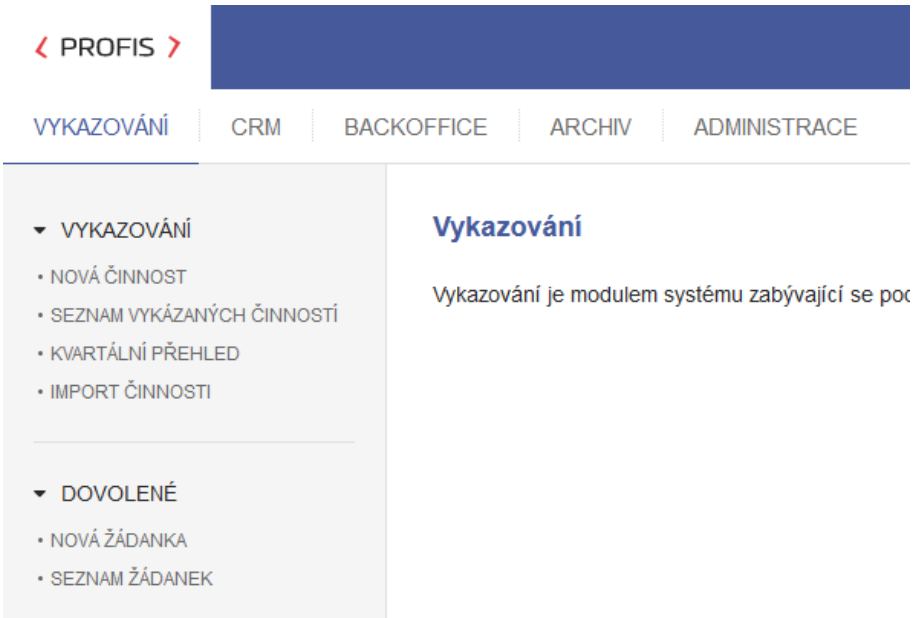
Klíčovým modulem a úlohou systému je správa pracovního výkazu. Zaměstnanci jsou povinni pečlivě evidovat všechny své aktivity na projektech. Tato evidence zahrnuje informace o tom, který zaměstnanec vykonával danou činnost, datum, dobu trvání a projekt, pod kterým byla činnost prováděna. Tato data lze využít a agregovat je pro využití při generování.

1.1.3.2 Katalog referencí

V tomto modulu jsou uloženy všechny reference od zákazníků, které zaměstnanec obdržel. Reference nejsou a nemohou být úplná, protože někteří zákazníci



Obrázek 1.3: Diagram relevantních komponent systému Profis.



Obrázek 1.4: Výřez uživatelského rozhraní Profisu.

nechtějí, nebo nemohou přidat doporučení kvůli citlivosti projektů. Tento modul by mohl být využit pouze jako doplnění reference k určitým projektům.

1.1.3.3 Znalostní matice

Firma využívá znalostní matici jako centrální evidenci nejpoužívanějších technologií napříč zákazníky. Každý zaměstnanec je povinen aktualizovat tuto matici tím, že označí technologie, se kterými pracoval, a vyjádří úroveň svých znalostí s danou technologií na stupnici od 1 do 5. Je však důležité poznamenat, že i přestože je údržba matice povinná, data v ní často zastarávají a mohou být značně subjektivní. Tyto informace o znalostech s určitými technologiemi mohou být využity při tvorbě životopisů zaměstnanců a také jako doporučení pro technologie, se kterými daný uživatel pracoval.

1.1.3.4 Možné budoucí integrace

Profis se neustále vyvýjí a jsou do něj přidávány nové funkcionality. Jedním z plánovaných rozšíření je modul školení, ze kterého bude možné získat data ohledně kurzů a certifikátů, které zaměstnanec dokončil. Z tohoto důvodu by měl modul životopisů být snadno rozšiřitelný a některé kroky nahraditelné za jiný vstup.

1.2 Funkční specifikace

Základem každého projektu je funkční specifikace, formální dokument sloužící k popisu funkcí vyvíjeného systému, jeho vzhledu a charakteristik. Tento dokument hraje klíčovou roli při identifikaci a sladění požadavků klienta s funkcionalitou aplikace. Jeho účelem je poskytnout jasný rámec pro tým vývojářů a zákazníka po celou dobu vývojového cyklu projektu. Mimo zadání diplomové práce byla vytvořena specifikace, která byla následně schválena zainteresovanými osobami zákazníka.

1.2.1 Zainteresované strany

Vzhledem k charakteru projektu, který se zabývá rozšířením již existujícího systému a organizace je vyžadována komunikace s širokým spektrem osob po celou dobu vývojového procesu. V rámci projektu je nezbytné komunikovat s následujícími týmy nebo osobami:

- Manažer zaštiťující systém
- Oddělení lidských zdrojů
- C# Profis tým
- Java Profis tým

Pro úspěšnou realizaci diplomové práce a správné implementaci modulu pro generování profesních životopisů je nezbytné konzultovat všechny kroky a rozhodnutí s příslušnými osobami.

1.2.2 Teorie sběru požadavků

Metoda sběru požadavků se odvíjí od prostředí a povahy projektu. Mezi typické metody [1] sběru požadavků patří:

- Brainstorming
- Rozhovor
- Prototypování
- Pozorování
- Reverse Engineering
- Dotazník

1.2.2.1 Brainstorming

Brainstorming je populární technika využívána na začátku projektu. Je to technika ve které je navrženo, co nejvíce nápadů za krátkou dobu, bez ohledu na kvalitu a proveditelnost. Brainstorming pomáhá s udáním směru na začátku projektu. Zainteresované osoby jsou posazeny do jedné místnosti, je jim sdělena podstata schůzky. Po chvilce přemýšlení jsou vyzváni ke sdělení svých nápadů. Tyto ideje jsou zapsány a následně diskutovány. Neproveditelné návrhy jsou vyškrtnuty, zbytek je rozveden v dalších iteracích nebo technikách. [2]

1.2.2.2 Rozhovor

Rozhovor se zainteresovanými osobami je důležitý k vytvoření úspěšného systému. Je možné vést osobní, nebo skupinový rozhovor. Výhodou skupinového rozhovoru je více pohledů na proces a snižuje šanci, že dotazovaní nebudou mít plné povědomí o všech aspektech procesu. [2]

1.2.2.3 Prototypování

Při prototypování jsou nejdříve získány předběžné požadavky a na jejich základě je vytvořen prototyp - prvotní verze systému. Tento prototyp je poskytnut klientovi, který poskytne zpětnou vazbu a případně upraví požadavky. Po úpravě aplikace lze opět poskytnout novou verzi klientovi a takto dále iterovat, dokud systém nesplní hranici obchodního přínosu, nebo neproběhne smluvený počet iterací. [1]

1.2.2.4 Pozorování

Pozorováním se snaží analytik identifikovat jednotlivé kroky procesu, jeho slabá místa a možnosti vylepšení. Pozorování může probíhat aktivně a pasivně. Pasivní lze vhodně kombinovat s prototypováním, protože lze získat uživatelovu odezvu na poskytnutý prototyp, zatímco aktivní pozorování lze využít k lepšímu pochopení existujícícho byznys procesu. [1]

1.2.2.5 Reverzní inženýrství

Reverzní inženýrství odkrývá princip fungování migrovaného systému. Je využíván v případě, kdy není dostupná dostatečná dokumentace. Je nutné podotknout, že technika nedokáže identifikovat, co by systém měl dělat, nebo co dělá špatně. [1]

1.2.2.6 Dotazník

Dotazník je technika využívána v případě, kdy je zapotřebí získat informace od velkého množství lidí a rozhovory jsou neproveditelné, ať už z hlediska financí, času nebo lokality jednotlivých osob. Návrh dotazníku je složitý, chybí zpětná vazba, a proto je úspěch silně závislý na jeho kvalitě. Tato technika je vhodná pro méně komplexní systémy. [1]

1.2.2.7 Výběr metody sběru požadavků

Pro sběr požadavků byl zvolen skupinový rozhovor. Většina z výše zmíněných technik byla zavrhnuta z důvodu povahy projektu. Rozhovor byl veden se všemi zainteresovanými osobami. Během tohoto procesu bylo identifikováno mnoho potenciálních požadavků, což vyžadovalo stanovení rozsahu systému, který bude implementován v rámci diplomové práce.

Po konzultaci s vedoucím práce byla vybrána metoda MoSCoW pro prioritizaci požadavků. V technice MoSCoW jsou požadavky rozděleny do čtyř kategorií dle jejich priority. Mezi kategorie patří:

- **Must have** - jedná se o požadavky, které musí být implementovány, bez těchto požadavků nelze považovat systém za dokončený.
- **Should have** - požadavky, které nejsou nezbytné pro fungování systému, ale přináší velký přínos.
- **Could have** - požadavky, které nemají dopad na fungování systému, pokud nejsou implementovány.
- **Will not have** - požadavky, které nejsou v současné době prioritní, ale mohou být implementované v budoucnu.

Požadavky jsou barevně odlišeny podle kategorie, do které patří. Must have požadavky jsou značeny červeně , Should have modře , Could have černě a Will not have šedě.

1.2.3 Funkční požadavky

- Uživatelé mohou vyplnit životopis
- Uživatelé mohou vygenerovat životopis
- Reprezentaci životopisu je možné uložit do databáze
- Systém umožní smazat záznamy z databáze pro vybraného uživatele
- Systém umožní vyhledávat uživatele dle projektu a znalostí

- Systém umožní uživateli vytvořit různé verze životopisu
- Systém bude ukládat historii verzí životopisů
- Vygenerovaný životopis lze skrz systém uložit na síťový disk

1.2.4 Nefunkční požadavky

- Vygenerovaný životopis musí být ve formátu docx.
- Životopis musí ctít aktuální šablonu firmy.
- Modul musí být napsán v jazyce Java ve frameworku Spring
- Aplikace musí být napsána modulárně, logika nesmí být pevně spjata s modelem ani komunikační vrstvou.
- Použité technologie musí být open-source.
- Systém musí být důkladně zdokumentován na interní XWiki.
- Systém musí využít databázi systému Profis.
- Zaměstnanec může vytvořit pouze vlastní životopis.
- Generátor bude dostupný jako stand-alone aplikace pro externí zaměstnance.
- Systém bude využívat autorizaci založenou na pozici ve firmě a projektech na kterých pracovník pracoval.
- Životopisy všech zaměstnanců budou dostupné pouze pracovníkum oddělení lidských zdrojů a manažerům.

1.2.5 Shrnutí funkční specifikace

Z funkční specifikace vyplývá, že klíčovým aspektem je správně navrhnout modul s ohledem na jeho budoucí rozšíření. Množství funkcí, které modul nabídne, není tak důležité jako úspěšné zaintegrování rozšíření do existujícího systému. Z tohoto důvodu je nezbytné provést důkladnou analýzu a návrh modulu s případnými úpravami v existujících částech systému. Důležitou součástí je vytvoření podrobné dokumentace a kódu, který bude snadno rozšiřitelný a srozumitelný, což je hlavním cílem práce.

1.3 Architektura

V této části jsou analyzovány různé architektonické varianty s cílem vybrat nejvhodnější variantu pro implementační část diplomové práce.

1.3.1 Spring framework

Využití Springu je podmíněno v požadavcích na systém. Spring framework zaujímá dominantní postavení v prostředí jazyka Java [3] a je přirozenou volbou při tvorbě webových aplikací. Spring poskytuje komplexní infrastrukturní podporu pro vývoj a umožňuje vývojářům se soustředit na vývoj samotné aplikace.

Hlavní výhodou frameworku je kontejner Inversion of Control (IoC), který se stará o běh celé aplikace a vkládá závislosti, až v moment, kdy jsou vyžadovány. Díky tomu mohou být jednotlivé komponenty systému na sobě, co nejméně závislé. Mezi další výhody patří technologie pro programování s aspekty, pro připojení k databázím, pro integraci na externí služby, email nebo messaging služby. Spring je všestranný volně dostupný framework s širokou komunitou. [4]

Spring aplikace typicky využívá vrstevnatou architekturu skládající se ze 4 vrstev:

- **Prezentační vrstva** – zpracovává HTTP požadavky, překládá JSON formát do objektů, autentikuje požadavky a předává je do byznysové vrstvy.
- **Byznys vrstva** – obsahuje byznys logiku celé aplikace. Skládá se ze servisních tříd, požadavky dostává od prezentační vrstvy a používá perzistenční vrstvu pro přístup k datům. Také se stará o autorizaci a validaci dat.
- **Perzistenční vrstva** – stará se o všechnu logiku ohledně přístupu k databázi a mapování z objektů na databázové entity a naopak.
- **Databázová vrstva** – poskytuje CRUD operace - vytvoření, získání, upravení a smazání záznamů z tabulky.

Každá z vrstev by měla pouze volat vrstvu pod ní a být volána vrstvou nad ní [5]. V implementační části práce by měl být ctěn nejlepší přístup k vývoji ve frameworku Spring a jazyce Java.

1.4 Rešerše integračních řešení

V této sekci je provedena rešerše možných integračních řešení a proveden výběr technologie, která splňuje definované požadavky.

1.4.1 Požadavky na integrační řešení

Vybraný typ komunikace by měl pracovat s JSON formátem, neboť uživatelské rozhraní komunikuje se serverovou částí výměnou JSON zpráv. Vygenerovaný životopis by měl být dostupný ihned po vygenerování. Není vyžadována extrémní odolnost vůči chybám, jelikož modul nebude kritickou částí systému. Při nedostupnosti, nebo chybě bude možné požadavek zaslat z uživatelského rozhraní znovu. Preferováno je jednoduché řešení pro zamezení předčasné optimalizace.

1.4.2 Synchronní komunikace

Jednou z možností komunikace mezi systémy je synchronní komunikace, pro kterou existuje nezpočet způsobů. Níže jsou rozebrány nejpoužívanější komunikační rozhraní mezi podnikovými systémy.

1.4.2.1 REST rozhraní

Representational State Transfer (REST) je v nejvyužívanějším architektonickým stylem aplikačního rozhraní (API). Styl je založený na struktuře URL a protokolu HTTP. Nejedná se o framework nebo knihovnu, ale o popis bezestavové a kešovatelné interakce mezi klientem a serverem. REST využívá URL adresy webových zdrojů a HTTP metod k akci, kterou má server provést. Možné HTTP metody jsou:

- **GET** slouží k získání existujícího prostředku nebo více prostředků
- **POST** slouží k vytvoření nového prostředku
- **PUT** slouží k aktualizaci existujícího prostředku nebo k jeho vytvoření, pokud neexistuje
- **DELETE** slouží k odstranění prostředku
- **PATCH** slouží k částečné aktualizaci existujícího prostředku

Největší výhodou rozhraní REST je jeho vyzrálost a oblíbenost. Většina vývojářů s ním již pracovala, nebo ho zná. Proto správa REST rozhraní je jednodušší než u ostatních technologií. Díky tomu, že každý prostředek se nachází za jedinečnou URL, se monitoring a limitování počtu požadavků stává jednodušším oproti ostatním technologiím. Další výhodou je snadné kešování, které využívá výhody HTTP protokolu.

Nevýhodou rozhraní REST je nedostatečené nebo přebytečné posílání dat. Pro získání vnořených entit může být zapotřebí provést více požadavků. Dalším příkladem je získání podmnožiny entity. V takovém případě server posílá všechna data, na která je daný koncový bod nakonfigurován. [6]

1.4.2.2 SOAP rozhraní

SOAP rozhraní je založené na XML (Extensible Markup Language) a nezávislé na transportním protokolu. Nejčastěji je využíván v kombinaci s WSDL (Web Services Description Language), který poskytuje popis jak k službě přistoupit, dostupné operace, parametry, které služba přijímá a využíváné zprávy (požadavek/odpověď). S těmito informacemi je možné vygenerovat klientskou proxy v jakémkoliv programovacím jazyku. Pro SOAP existuje mnoho rozšíření, jako WS-Addressing nebo WS-Security. SOAP poskytuje široké spektrum užití, nicméně lze využít pouze rozšíření potřebné k dané úloze. Nevýhodou SOAPu je velikost jeho XML zpráv, které jsou oproti JSONu více paměťové náročné a mohou více zatěžovat síť. Další nevýhodou je menší intuitivnost a větší komplexita oproti RESTu. [7]

1.4.2.3 gRPC rozhraní

gRPC framework je postavený na modelu klient-server volání procedur. Klientská aplikace volá metody serveru, jako by to byly lokální metody. Jedná se o striktní komunikaci, kdy klient i server musí mít stejnou definici schématu, neboli kontrakt. Kontrakt se v gRPC definuje pomocí domain-specific jazyka (DSL) zvaného Protocol Buffer, který díky svému kompilátoru vygeneruje serverové a klientovské třídy. Třídy lze jakkoliv rozšiřovat a přidávat vlastní logiku.

gRPC poskytuje několik způsobů klient-server komunikace.

- Request-Response komunikace - pro jeden požadavek klienta vrátí server jednu odpověď
- Server streaming komunikace - pro jeden požadavek od klienta je ze serveru posláno několik odpovědí
- Client streaming komunikace - pro několik požadavků od klienta je vrácena jedna odpověď

Největší výhodou gRPC je jeho rychlost a jazykově nezávislý generátor kódu. Nicméně gRPC je méně populární než REST rozhraní a případné dohledávání chyb je obtížné, protože formát zpráv je pro člověka nečitelný a je zapotřebí použít speciální nástroje k provedení analýzy komunikace. [6]

1.4.2.4 GraphQL rozhraní

GraphQL poskytuje dotazovací jazyk pro API a framework obstarávající GraphQL dotazy. Rozhraní nevyužívá všechny HTTP metody pro manipulaci s daty a využívá zejména metodu POST. Oproti tomu využívá dotazy pro získání dat, mutace pro modifikování dat a subskripci pro získávání dat při jejich změně.

Rozhraní dovoluje klientovi požádat a získat pouze potřebná data, tudíž žádná nepotřebná data nejsou posílána přes síť. Proto GraphQL může vést k lepšímu výkonu. Popis rozhraní je více restriktivní a specifický oproti RESTu, poskytuje detailní chybové hlášky pro debugování a automaticky generovanou dokumentaci API.

Jelikož každý dotaz může být jiný, tak GraphQL nemůže využívat HTTP kešování a proto keš je složitější na implementaci. Další nevýhodou může být velká komplexita dotazů, které mohou přetížit server, tudíž je zapotřebí komplexitu dotazů patřičně omezovat. [6]

1.4.3 Asynchronní komunikace

V této kapitole jsou rozebrány typy asynchronní komunikace, která oproti asynchronnímu zpracování nevyžaduje, aby server byl dostupný. Hlavním rozdílem mezi synchronní komunikací a asynchronním zpracováním je, že klient nečeká na vyhodnocení požadavku a pokračuje bez odpovědi. Asynchronní zpracování může mít několik implementací. Například rozhraní REST podporuje asynchronní zpracování, kdy při požadavku je klientu vrácen odpovídající HTTP kód a pro výsledek musí klient znovu zavolat server. Těchto způsobů existuje mnoho, ale v této kapitole jsou rozebrány pouze čistě asynchronní komunikace, při kterých klient nevolá přímo server, ale takzvaný Message broker. Message

broker se stará o správné rozeslání zpráv, poskytuje ochranu proti výpadkům, nebo přetížení systému. Hlídá také dostupnost jednotlivých serverů (konzumentů).

1.4.3.1 Publish-subscribe komunikace

Publish-subscribe komunikace je využívána v moderních distribuovaných systémech. V tomto komunikačním modelu odesílatel neposílá zprávu přímo příjemci, ale prostředníkovi (Message broker). Publish-subscirbe komunikace se skládá z následujících komponent [8]:

- Topic – každá zpráva má definovaný topic, do kterého spadá. Topic funguje jako jakýsi kanál mezi odesílatelem a příjemcem. Každý topic disponuje seznamem příjemců, kteří daný topic odebírají.
- Subscriber – je přijemcem zprávy. Každý příjemce se přihlásí k topicu od kterého bude odebírat zprávy. Příjemců může být několik a mohhou danou zprávu zpracovávat paralelně.
- Publisher – je odesílatel zprávy do topicu. Interakce je v relaci one-to-many, odesílatel odešle jednu zprávu a ta je doručena všem odebíratelům daného topicu.

Díky využití publish-subscribe docílíme nízkého provázání mezi bloky, protože odesílatel zprávy neví, kdo je příjemcem a odebíratelé neví od koho zpráva dorazila. Další výhodou je paralelní zpracování zpráv a zvýšená odolnost vůči výpadkům konzumentů zpráv. [9]

1.4.3.2 Message Queueing komunikace

Message Queueing je typ asynchronní point-to-point komunikace, tzn. mezi dvěmi systémy, nebo části systému. Zprávy jsou zasílány do fronty, ze které si je konzument vyzvedává. Producent a konzument není pevně svázán a mohou pracovat nezávisle na sobě. Pokud konzument je zaneprázdněn, nebo je nedostupný, tak zprávy nejsou ztraceny, ale drženy ve frontě, dokud není konzument opět připraven k jejich vyzvednutí. Výhodou Message Queueingu je vyšší flexibilita, škálovatelnost a spolehlivost komunikace. Proto je tento typ komunikace využíván v systémech, které musí mít vysokou dostupnost a je možné výpočetní proces provádět asynchronně. [10]

1.4.4 Shrnutí integračních možností

Všechny zmíněné integrační možnosti, spolu s jejich klady a zápory, byly předvedeny technickému konzultantovi z firmy Profinit a byla vybrána synchronní komunikace využívající rozhraní REST. Hlavní důvody výběru REST rozhraní byly následující:

- Životopis by měl být vrácen ihned.
- Není očekáván velký objem požadavků, neboť se jedná o interní systém, který nebude využíván mnoha uživateli najednou.

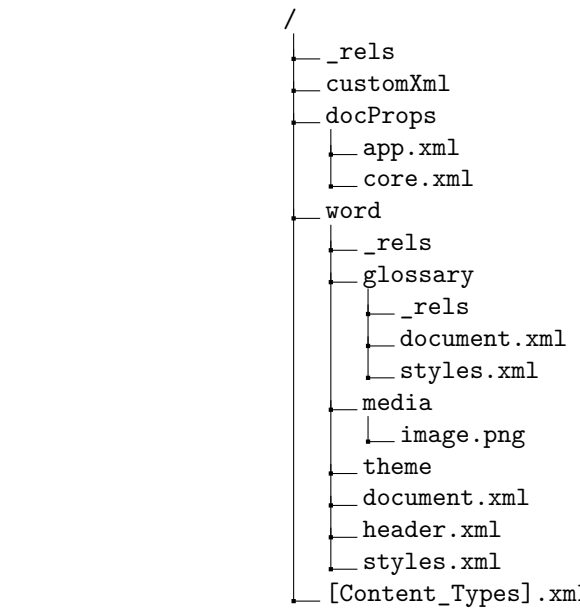
- Profis momentálně využívá REST rozhraní, proto je nasnadě využít stejné technologie i v novém modulu.
- Data pro modul budou nejčastěji sbírána v uživatelském rozhraní, proto pro jednoduchost práce je preferován formát JSON odpovědi.

1.5 Analýza práce s DOCX dokumenty

Výstupem práce by měl být dokument vygenerovaný ve formátu DOCX, který vytváří textový editor Word z rodiny nástrojů Microsoft Office. Roku 2007 Microsoft přestal využívat starý formát souborů ukládající dokumenty v binární formě za formáty založené na jazyce XML. Tato změna přinesla nespočet výhod nejen pro vývojáře. Hlavními přednostmi nového stylu ukládání je využití komprese dat, zlepšené obnovování poškozených souborů, podpora pokročilých funkcí (jako například automatické uložení) a mnoho dalších. [11]

Vzhledem k této práci je nejdůležitější fakt, že je možné extrahovat jakýkoliv DOCX dokument, upravit jeho XML podobu a po úpravách jej zazipovat zpět. Tímto je možné využívat existující styly, nahrazovat a přídávat nové elementy stejně jako při práci s formáty XML. Pokud je dokument upraven a zazipován správně, lze jej opět otevřít ve Word editoru.

Typickou strukturu extrahovaného DOCX dokumentu lze vidět na adresářovém stromě 1.5. Složky **_rels** obsahují XML soubor s definovanými relacemi na ostatní soubory, jako jsou například obrázky, styly, nastavení a jiné XML soubory. Pokud chceme využít data z jiného souboru v upravované části dokumentu, je nutné přidat jeho relaci do odpovídajícího relačního souboru a zapsat identifikátor, který je poté využíván jako reference v upravovaném XML souboru. Ve složce **docProps** se nachází soubory obsahující informace



Obrázek 1.5: Stromová struktura extrahovaného DOCX dokumentu.

o Word dokumentu, jako například data o uživateli, který dokument vytvořil nebo který jej jako poslední upravil, kolik stránek, slov, odstavců se nachází v dokumentu a další informace. Pro vytvářený modul je nejdůležitější složka **word**, ve které se nachází všechny soubory obsahu dokumentu. V podsložce **glossary** je doplňkový dokument obsahující definice stylů, číslování, komentáře a jiné konstrukty, na které je odkazováno z hlavního XML souboru. Složka **media** obsahuje všechny média použitá v dokumentu, jako jsou obrázky, videa, zvuky a jiné. Ve složce **theme** se nachází definice motivu dokumentu, měnit tento soubor je užitečné spíše v editoru PowerPoint, pro změnu vlastností předdefinovaného motivu.

1.5.1 Manipulace pomocí Java knihoven

Z nefunkčních požadavků vyplynulo, že systém musí být naprogramován v Javě. V této sekci jsou rozebrány možné technologie a knihovny, díky kterým lze v Javě generovat dokumenty, nebo pracovat s XML podobou Word dokumentu.

1.5.1.1 Požadavky na techniku generování

Vybraná technologie musí dokázat vygenerovat životopis v požadovaném formátu a stylu. Musí být volně dostupná a podporovat jazyk Java. Práce s ní by měla být intuitivní. Úprava předlohy životopisu, by měla být co nejjednodušší, jak pro programátora, tak pro netechnického uživatele.

1.5.1.2 JasperReports

JasperReports je volně dostupná knihovna pro tvorbu reportů ve formátech PDF, HTML, XLSX, DOCX a dalších. Knihovna je výborná pro vizualizaci dat a tvorbu reportů přímo z jejich surové podoby. JasperReports vedle knihovny poskytuje designové prostředí pro návrh reportů přímo v uživatelském rozhraní. Výstupem návrhu je jeho XML definice (jrxml), kterou lze poté v Java kódu naplnit daty. Velkou výhodou JasperReports je možnost přímo navázat datový zdroj na šablonu. Samotné generování v kódu je jednoduché, všechno mapování je nastavováno už při tvorbě šablony [12]. Ukázku vygenenerování reportu z datového zdroje lze vidět na algoritmu 1.

1.5.1.3 Apache POI

Apache POI poskytuje Java API pro manipulaci se soubory založených na Office Open XML (OOXML) standardech a OLE2 Compound Document (OLE2) formátu. Knihovna poskytuje metody pro čtení a zápis do dokumentů vytvořených v nástrojích Microsoft Office. Apache POI podporuje, jak staré binární formáty souborů, tak nové formáty založené na XML. [13]

Třída `XWPFDocument` zašťiťuje operace nad dokumenty. V konstruktoru přijímá `InputStream` upravovaného dokumentu a poskytuje nespočet metod k úpravám. Lze takto přistoupit k hlavičkám, patičkám, obrázkům nebo jednotlivým elementům těla dokumentu. Jednou z výhod je možnost využít předdefinované styly z DOCX dokumentu dle jejich identifikátoru. Tudíž je možné využít prázdný DOCX soubor jako předlohu a vkládat do něj text, úpravy stylu by znamenaly pouze změnit styl ve DOCX dokumentu a nové soubory

```
JasperReport jasperReport = JasperCompileManager.compileReport(
↪  "src/main/resources/tabulka.jrxml");

JRDataSource dataSource = new
↪  JRBeanCollectionDataSource(dataList);

Map<String, Object> parameters = new HashMap<>();
parameters.put("tableSource", DATA_CLASS);

JasperPrint jasperPrint =
↪  JasperFillManager.fillReport(jasperReport, parameters,
↪  dataSource);
File file = new File("report.docx");
file.createNewFile();

JRDocxExporter exporter = new JRDocxExporter();
exporter.setExporterInput(new SimpleExporterInput(jasperPrint));
exporter.setExporterOutput(new
↪  SimpleOutputStreamExporterOutput(file));
exporter.exportReport();
```

Zdrojový kód 1: Ukázka práce s JasperReports.

byly automaticky vygenerovány v novém stylu. Načtení dokumentu a vytvoření nových odstavců s předdefinovaným stylem v Apache POI lze vidět na algoritmu 2.

1.5.1.4 Spire.Doc

Spire.Doc poskytuje Word API pro jazyky Java, C#, Python a C++. Knihovna poskytuje metody pro vytváření DOCX dokumentů, jejich manipulaci, tisk a mnoho dalšího. Pomocí knihovny je možné vkládat do dokumentů obrázky, podpisy, tabulky, nastavovat hlavičky, patičky a mnoho dalšího. Umožňuje také porovnání dokumentů, jejich spojování nebo extrahování dat [14]. Spire.doc je poskytováno ve dvou verzích – placené a zdarma. Placená verze disponuje zkušebním módem, který poskytuje konverzi 10 stránek a všechny vytvořené dokumenty jsou opatřeny vodoznakem. Zdarma verze knihovny nedisponuje pokročilými funkcemi a je limitována na 500 odstavců a 25 tabulek v dokumentu, konverze do PDF je omezena na první tři strany [15]. Práce se Spire.Doc je podobná jako práce s Apache POI.

1.5.2 Shrnutí analýzy práce s DOCX dokumenty

Všechny knihovny dokáží generovat DOCX dokumenty. Nicméně v JasperReports je zapotřebí přepsat Word šablonu do JasperReports šablony přes poskytnuté vývojové prostředí a při každé změně stylu je nutné šablonu stejným způsobem aktualizovat. JasperReports je uzpůsobený pro reporty s velkým množstvím dat, proto nemusí být vhodný pro tento typ úlohy. Apache POI a Spire.Doc dokáží pracovat s DOCX dokumentem přímo a využívat jeho vy-

```
XWPFDocument document =
    new XWPFDocument(new FileInputStream("cv.docx"));

XWPFParagraph paragraph = document.createParagraph();
XWPFRun run = paragraph.createRun();
paragraph.setStyle("Nadpis-1");
run.setText("Programovací jazyky");

paragraph = document.createParagraph();
paragraph.setStyle("Nadpis-2");
run = paragraph.createRun();
run.setText("Java, SQL, JavaScript, C++, Python");

FileOutputStream out = new FileOutputStream("example.docx");
document.write(out);
out.close();
document.close();
```

Zdrojový kód 2: Ukázka práce s Apache POI s nastavením stylů.

tvořené styly. Není nutné přepisovat Word šablonu do žádné jiné, proto jsou úpravy snažší. Spire.Doc ve verzi zdarma je značně omezen a disponuje pouze základními funkcemi. Apache POI je zdarma v plné verzi a stále rozšiřován.

S ohledem na rozsah generovaného souboru a nutného ohýbání JasperReports byla zvolena knihovna Apache POI, která je pravidelně aktualizována a dokáže spolehlivě pracovat s DOCX formátem.

1.6 Dopadová analýza

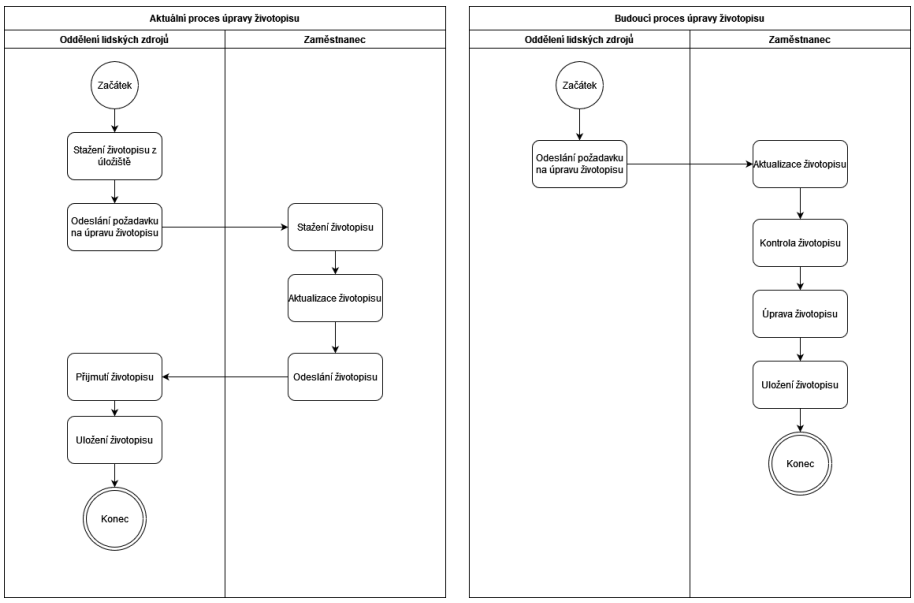
Přidání nového modulu do interního systému může mít dopad na jiné, již existující části. V této sekci je provedena dopadová analýza, jejímž úkolem je zmapovat potřebné úpravy a rozšíření systému Profis, ale také procesy, do kterých nový systém bude zasahovat.

1.6.1 Procesní dopady

Systém by měl zefektivnit práci se životopisy a proto bude ovlivňovat procesy, kde životopisy figurují. Mezi zasažené procesy patří:

- Nástup nového zaměstnance.
- Tvorba nabídek zákazníkům.
- Ukončení pracovního poměru zaměstnance.

Jedním z hlavních přínosů této práce bude zjednodušení procesu tvorby a aktualizace životopisu. Obě tyto činnosti sdílejí zásadní část procesu a liší se pouze v tom, zda oddělení lidských zdrojů přikládá k e-mailu životopis zaměstnance nebo šablonu. Poté probíhá proces stejně, ať už jde o jeho vytvoření či



Obrázek 1.6: Diagram aktuální a budoucí úpravy životopisu.

úpravu. V novém procesu by měla být veškerá manipulace s životopisem prováděna výhradně prostřednictvím systému Profis. Pracovník z oddělení lidských zdrojů zašle zaměstnanci upozornění na potřebu aktualizace životopisu, které v budoucnu může být odesíláno systémem Profis. Po obdržení notifikace zaměstnanec aktualizuje svůj životopis. Po dokončení aktualizace se nový datový model životopisu odesílá do nového modulu a je vytvořen ve formátu DOCX. Zaměstnanec tento životopis stáhne, zkontroluje a případně provede potřebné úpravy. Následně může uživatel tento životopis uložit do úložiště. Podstatné rozdíly mezi původním a novým procesem jsou zachyceny na diagramu 1.6.

Dalším významným přínosem bude zjednodušení procesu mazání životopisů při ukončení pracovního poměru zaměstnance. V rámci tohoto procesu je nezbytné odstranit veškeré verze životopisu. S ohledem na zachování bezpečnosti dat je nezbytné pečlivě nastavit pravidla pro správné provedení mazání, včetně specifikace toho, kdo bude mít oprávnění k provedení této operace a za jakých podmínek. Po vypracování návrhu je nutné tento proces konzultovat s oddělením lidských zdrojů, aby bylo zajištěno, že je navržen v souladu s interními předpisy a očekáváními organizace.

1.6.2 Dopad na databázové schéma

Jedním z klíčových požadavků je integrace datové struktury životopisu do databáze systému Profis. Současná databáze obsahuje informace o všech zaměstnancích firmy a je nezbytné schéma rozšířit o nové tabulky pro uchování životopisů. V rámci návrhové fáze práce budou navrženy nové tabulky a jejich implementace do existující databáze. Proces navrhování nových tabulek vyžaduje konzultaci s týmem odpovědným za správu databázového schématu, neboť nové návrhy musí respektovat standardy a konvence, které jsou v rámci databáze používány. Provedení změn bude v kompetenci databázového týmu.

1.6.3 Dopad na interní systémy

Nový modul bude mít dopad na všechny části Profisu. Uživatelské rozhraní musí být rozšířeno o obrazovky obsluhující práci s modulem a formulářem pro zadání dat pro životopis. Jak bylo zmíněno v analýze současného stavu, uživatelské rozhraní volá C# část systému, která se stará o přístup k databázi. Z konzultací s technickým zástupcem Profinitu vyplynulo, že C# Profis bude volat modul generování životopisů a obstará i ukládání dat k tvorbě životopisu. Samotný modul bude vložen do Java Profisu, který není v současné době připraven k využití, ale je nutné navrhnout modul tak, aby bylo možné jej jednoduše zakomponovat do této vznikající aplikace. Jednotlivé části systému jsou spravovány různými týmy, proto je nutná neustálá součinnost při návrhu změn a implementaci modulu.

Kapitola 2

Návrh

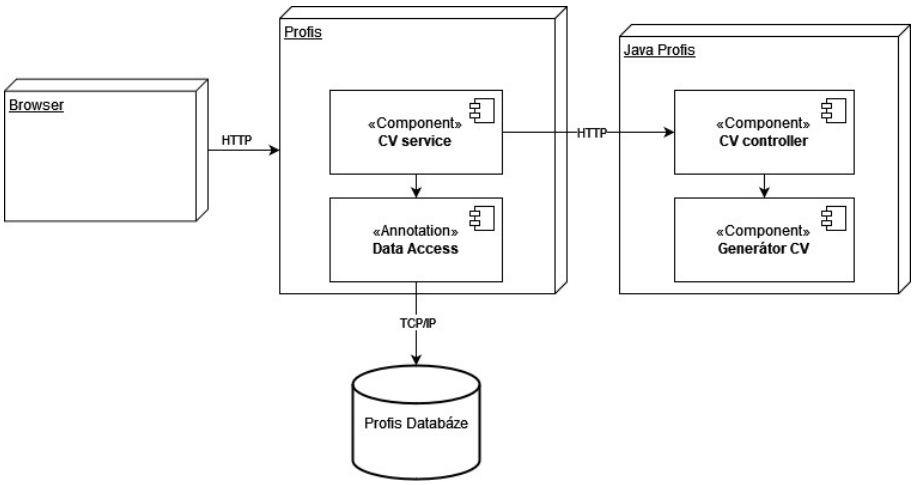
V této kapitole je proveden detailní návrh potřebných úprav a implementací v rámci jednotlivých částí systému Profis. Tyto navrhované změny jsou výsledkem analýzy systému a jsou projednávány a schvalovány vedoucími pracovníky společnosti Profinit. Celková úprava systému bude zahrnovat několik klíčových částí. Prvním krokem je vytvoření modulu, který umožní generování dokumentů ve formátu DOCX. Tento modul musí být následně integrován do stávajícího Profisu. Je také nezbytné provést úpravy, které zajistí, že systém bude připraven pro budoucí rozšíření a další vývoj. Dalším důležitým úkolem je navržení datového modelu pro životopisy a návrh nezbytných úprav v databázové struktuře systému. Tato část je klíčová pro správné fungování nového modulu a jeho integraci s existujícími daty. Následně je nutné navrhnout uživatelské rozhraní pro práci se životopisy, což zahrnuje vytvoření obrazovek a formulářů pro manipulaci s daty. V neposlední řadě je třeba podrobně popsat procesní přínos nového modulu, tedy jakým způsobem bude nový systém přispívat k efektivitě a produktivitě uživatelů.

2.1 Architektura

Do Java části Profisu bude přidán modul generátoru, který bude přijímat datový model z C# části Profisu a bude se starat o generování DOCX dokumentů. C# Profis bude sloužit, jako meziprvek mezi uživatelským rozhraním a generátorem. V budoucnu zde bude probíhat ukládání životopisů do databáze, ukládání vygenerovaných DOCX dokumentů a obecně veškeré manipulace s životopisy. Důvodem je přítomnost integrace na uživatelskou databázi, business logiky práce s uživateli, jako je Matice znalostí nebo Katalog referencí, tak i autorizace a autentikace. Návrh budoucí architektury je vyobrazen na diagramu 2.1.

2.2 Databázový model

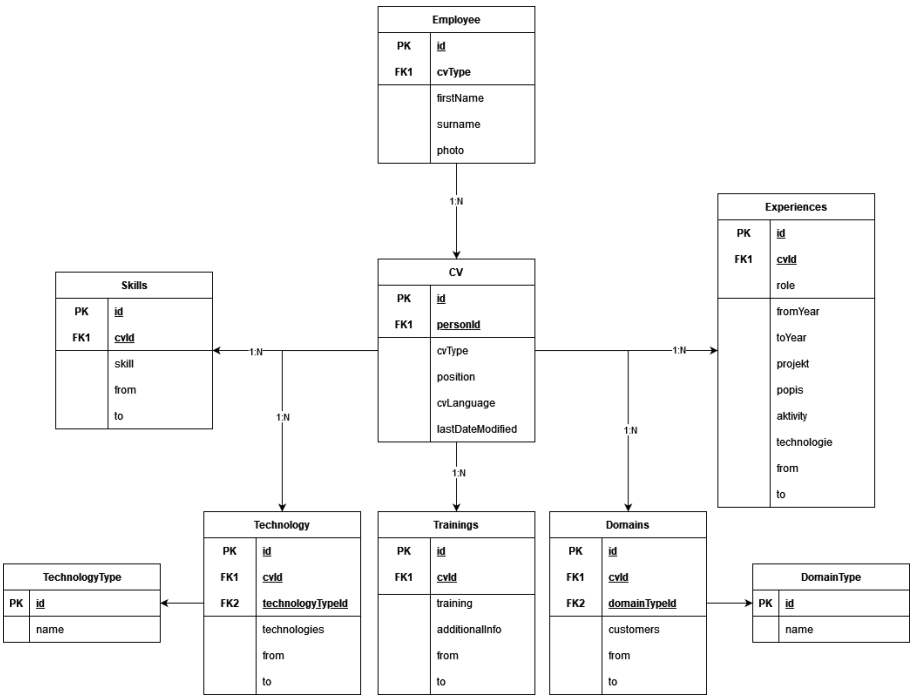
Databázi Profisu bude v budoucnu nutné rozšířit o nové tabulky životopisu. Některé informace již databáze obsahuje, proto budou přidány pouze tabulky a atributy, které není možné získat z jiných zdrojů. Data o uživateli, jako je jeho jméno, nebo fotografie, jsou již v databázi Profisu přístupná. Důležitým



Obrázek 2.1: Návrh architektury modulu pro generování profesních životopisů.

rozšířením budou data specifická pro profesní životopis. Databázi bude nutné rozšířit o následující tabulky:

- CV
  - Tabulka bude obsahovat hlavní informace o životopisu, jako je definovaný typ životopisu, jeho jazyk, datum poslední modifikace a pozice, kterou má uživatel v životopisu zapsanou.
- Skills
  - Tabulka bude obsahovat soupis měkkých znalostí zapsaných v životopisu.
- Trainings
  - Obsahuje název školení nebo certifikace s možností vložit dodatečné informace.
- Domains
  - Obsahuje odkaz na název domény a seznam zákazníků.
- DomainType
  - Jedná se o číselník, který obsahuje názvy domén. Byl vložen z důvodu navazujících požadavku od oddělení lidských zdrojů na možnost sjednotit názvy domén a technologií.
- Technology
  - Obsahuje odkaz na název typu technologií a soupis jednotlivých technologií.
- TechnologyType
  - Číselník obsahující názvy typů technologií.



Obrázek 2.2: Návrh databázového modelu.

- Experiences
  - Tabulka zastřešující jednolitvé projekty, na kterých uživatel pracoval. Skládá se názvu role, projektu, popisu projektu, aktivity a využitých technologií.

Navržený databázový model lze vidět na diagramu 2.2. Nezmíněná data, například soupis jazyků, jsou k dispozici z jiných částí systému a není nutné je duplikovat. Jednotlivé části životopisu obsahují informace o časovém rozsahu jejich platnosti pro jednoduché verzování.

2.3 Datový model

Datový model bude přemapování databázového modelu a dalších potřebných atributů do jednoho datového objektu (DTO). Tento model specifikuje strukturu životopisu, který má být vygenerován. S ohledem na to, že databázové relace v tomto kontextu nejsou relevantní, budou společné atributy přesunuty do hlavního objektu ve formě listů. Tento datový model životopisu bude očekáván generátorem jako vstupní parametr, a proto je nezbytné, aby veškerá data byla před jejich vložením do generátoru mapována na definovaný model. Model bude vyžadován v generátoru, což umožní agregaci dat v samotném modulu.

2.4 Struktura modulu

Z požadavků vyplývá, že i samotný modul musí být psán modulárně kvůli budoucímu začlenění generátoru do připravovaného Java Profisu. Modul se bude skládat ze dvou částí – generátoru a komunikačnímu rozhraní. Modul generátoru bude poskytovat logiku generování a požadovanou strukturu na vstupu. Komunikační vrstva se bude starat o příjem požadavků a jejich mapování na strukturu definovanou generátorem. Díky tomuto přístupu bude možné generátor, kdekoliv přepoužít a nebude pevně vázán na žádný typ komunikace.

2.4.1 Generátor

Vybranou knihovnou pro generování životopisů je dle výsledků analýzy 1.5 knihovna Apache POI. Generátor bude poskytovat interface a kontrakt, aby v bylo možné jednodušše implementovat nový generátor, bez potřeby přepisovat starý. Implementace generátoru se bude skládat z několika funkcí, které budou vždy generovat pouze určitou ucelenou část dokumentu, aby bylo docíleno seperation of concerns. Výsledný dokument bude produktem orchestrace jednotlivých metod. Při úpravách dokumentu bude nutné pouze změnit odpovídající metody. Výstupem generátoru bude DOCX dokument ve formě ByteArray, který lze převést na `Resource` a odeslat přes REST rozhraní zpět volajícímu. Vícejazyčnost při generování bude docílena pomocí internaciolizační podpory z frameworku Spring.

2.4.2 Komunikační rozhraní

Z rešerše 1.4 a následné analýzy bylo vybráno REST rozhraní, jako způsob integrace na ostatní části systému. Komunikační rozhraní slouží pouze pro účely diplomové práce a v budoucnu nebude využíváno. Jeho implementace bude jednoduchá, ale úplná, aby poskytovala dostatečný popis a návod, jak generátor integrovat.

Modul generátoru bude poskytovat následující endpoint:

- **/generate-cv**
  - **Metoda**
    - POST
  - **Tělo požadavku**
    - Datový model životopisu
  - **Možné odpovědi**
    - **200** - Dokument ve formátu DOCX
    - **400** - Chyba validace ve vstupních datech
    - **500** - Neočekávaná chyba při generování dokumentu

2.4.2.1 OpenAPI specifikace

Pro popis komunikačního rozhraní bude využita OpenAPI specifikace. Jedná se o jazykově nezávislý standard popisu HTTP API, který umožňuje jednoduché objevování a porozumění schopností služby, bez přístupu ke zdrojovému kódu

nebo dokumentace. OpenAPI specifikace může být využita generačními nástroji k zobrazení API, ke generování serverové a klientské části, nebo k využití v testovacích nástrojích. [16]

2.5 Ostatní části Profisu

Modul je nutné zaintegrovat do současné .NET části Profisu, aby bylo možné využít uživatelského rozhraní, databázového rozhraní a dalších integrací. V diplomové práci bude Profis rozšířen o end-to-end integraci. Od kontroleru komunikujícího s uživatelským rozhraním až po klienta, který bude volat Java generátor. Při implementaci bude dodržována konvence nastavena .NET týmem, proto je nutné připravit integraci na autorizaci a při návrhu uživatelského rozhraní navrhnout i autorizační pravidla. Důležité je připravit modul pro budoucí rozšíření, z tohoto důvodu budou data do rozbalovacích nabídek uložena v serverové částí, aby je bylo možné v budoucnu uložit do databáze a vytvořit obrazovky pro jejich manipulaci.

Profis bude rozšířen o následující endpointy:

- **/api/BackOffice/CVGenerator/GenerateDocx**
  - **Metoda**
    - POST
  - **Tělo požadavku**
    - Datový model z formuláře uživatelského rozhraní
  - **Možné odpovědi**
    - **200** - Dokument ve formátu DOCX
    - **400** - Chyba validace vstupních dat
    - **500** - Neočekávaná chyba při volání metody
- **/api/BackOffice/CVGenerator/GetPolicy**
  - **Metoda**
    - GET
  - **Možné odpovědi**
    - **200** - Seznam nastavených autorizačních omezení
- **/api/BackOffice/CVGenerator/GetOptions**
  - **Metoda**
    - GET
  - **Možné odpovědi**
    - **200** - Datový model obsahující možnosti rozbalujících nabídek v uživatelském rozhraní.

2.6 Uživatelské rozhraní

Uživatelské rozhraní bylo navrženo v několika iteracích a po každé iteraci bylo podrobeno konzultacím s týmem Profis a zástupci oddělení lidských zdrojů během společných schůzek. Při návrhu bylo dbáno na dodržení grafického designu uživatelského rozhraní Profisu a co největšího využití existujících komponent. Návrh byl proveden pomocí nástroje Figma. Současně s konzultacemi navrženého uživatelského rozhraní byla prováděna verifikace požadavků na systém, což vedlo k identifikaci několika dalších potenciálních rozšíření systému, které jsou popsány v sekci 5.3.

Návrh uživatelského rozhraní zahrnoval dvě nové obrazovky. První obrazovka byla navržena pro tvorbu životopisu, která bude realizována v rámci diplomové práce, zatímco druhá obrazovka sloužila k zobrazení seznamu životopisů. Tato obrazovka sloužila k vizualizaci možného budoucího rozšíření a k prezentaci přínosů nového modulu pro zástupce oddělení lidských zdrojů.

2.6.1 Obrazovka tvorby životopisu

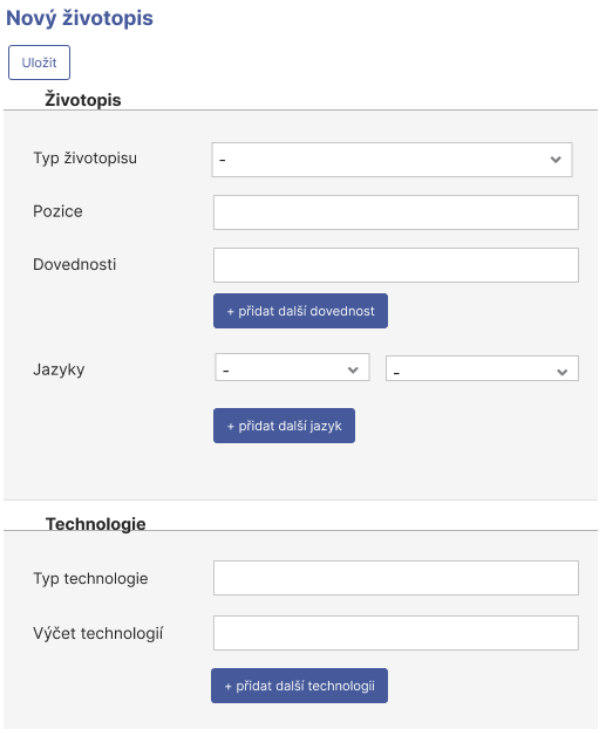
Obrazovka je určena pro tvorbu a v budoucnu i úpravu datového modelu životopisu. Cílem obrazovky je poskytnout uživateli co nejjednodušší možnost vytvoření životopisu prostřednictvím našeptávání, předvyplnění nebo doporučení nejrelevantnějších informací.

V první fázi bylo navrženo rozhraní, jehož výřez z formuláře je možné vidět na obrázku 2.3, celou stránku lze nalézt v příloze B.1. Během konzultace bylo identifikováno několik nedostatků, jako například nevhodné využití existujících komponent, nezobrazení fotky, která bude využita v životopisu, a nemožnost ji nahradit. Dále bylo identifikováno několik dalších hodnot, pro které bude možné využít číselník a rozbalovací menu

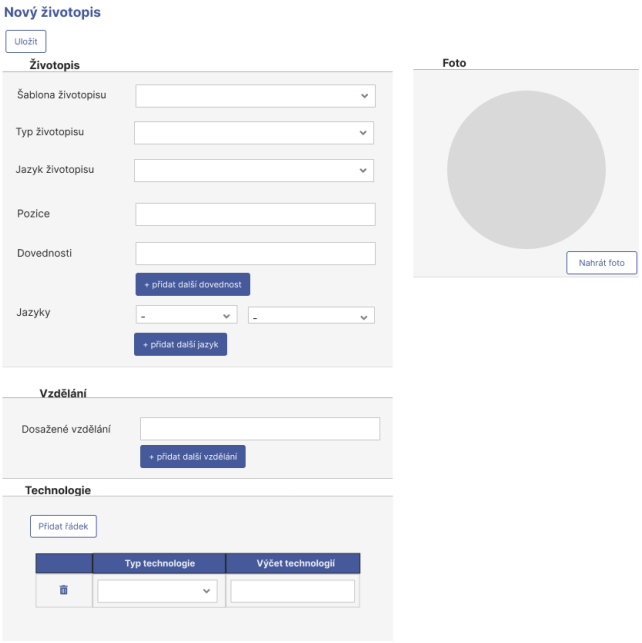
Výřez upraveného uživatelského rozhraní dle zpětné vazby lze vidět na obrázku 2.4, celá stránka je k nahlédnutí v příloze B.2. Vylepšení oproti původnímu návrhu zahrnují přidání možnosti nahrát fotografii, rozšíření o rozbalovací menu u několika políček a úpravy určitých částí formuláře. Tato opatření byla provedena z důvodu, že tyto komponenty jsou již používány v jiných formulářích v systému a jsou uživatelům dobře známé. Dále bylo přidáno pole pro jméno na životopisu, což umožní snadnější identifikaci zaměstnance, pro kterého je životopis upravován, zejména v případě, že oddělení lidských zdrojů bude upravovat profesní životopisy za zaměstnance. Také bude možné zahrnout titul do jména. Tím budou zaměstnanci schopni přidat svůj titul do životopisu dle své preference.

2.6.1.1 Autorizační omezení

Obrazovka bude dostupná ve dvou módech. **Každý uživatel** bude mít možnost vytvořit, zobrazit a upravit **svůj vlastní** životopis. **Oddělení lidských zdrojů** bude mít možnost vytvářet, zobrazovat a upravovat životopisy **všech** uživatelů.



Obrázek 2.3: Výřez prvního návrhu uživatelského rozhraní tvorby životopisu.



Obrázek 2.4: Výřez druhého návrhu uživatelského rozhraní tvorby životopisu.

2.6.1.2 Rozbor formuláře

V této sekci jsou rozebrány jednotlivé položky formuláře, jejich datový typ a popis. Políčka ve formuláři jsou:

- **Šablona životopisu**
  - Číselník, kdy v současné době se bude skládat z formátu portrait a landscape, není nutné zpětně podporovat staré formáty.
- **Typ životopisu**
  - Číselník, kde každý uživatel bude vyplňovat základní verzi životopisu, někteří mohou vyplnit i specializované verze, které budou v číselníku. Typicky se mezi verzemi liší pouze první strana životopisu.
- **Jazyk životopisu**
  - Číselník, kde dostupnými jazyky bude čeština a angličtina. V budoucnu možné rozšířit o němčinu a jiné jazyky.
- **Dovednosti**
  - Textový řetězec jakýchkoliv měkkých znalostí uživatele.
- **Jazyky**
  - Čísleník, obsahující o seznam relevantních jazyků a jejich úrovně dle standardní stupnice Společného evropského referenčního rámce. [17] V životopise bude úroveň namapována na grafickou reprezentaci podle šablony životopisu.
- **Fotografie**
  - Soubor fotografie ve formátu JPEG, nebo PNG.
- **Vzdělání**
  - Textový řetězec, kde bude vyplněn jakýkoliv formát nejvyššího dosaženého vzdělání
- **Technologie**
  - Číselník a textový řetězec, kde v číselníku budou unifikované kategorie technologií a uživatel do textového pole vypíše relevantní technologie.
- **Domény**
  - Číselník a textový řetězec, kde v číselníku budou unifikované kategorie domén a uživatel do textového pole vypíše relevantní zákazníky, nebo zaměstnavatele. Pokud si uživatel přeje vyplnit jiné domény, než ty které se nachází v číselníku, použije kategorii *Jiné*.
- **Pracovní zkušenosti**
  - Jednotlivé položky jsou textové řetězce, do kterých uživatel vypíše relevantní informace.

2.6.1.3 Budoucí rozšíření

Obrazovku je v budoucnu možné rozšířit o prvky usnadňující tvorbu a úpravu životopisu. Důvodem jejich vynechání je nepřipravenost ostatních částí Profisu na jejich integraci. Soupis těchto rozšíření byl vytvořen pro demonstraci potenciálu projektu zainteresovaným osobám.

V Profisu je v současné době ve fázi implementace nový modul evidence studia, který poskytne pro každého zaměstnance detailně popsané dosažené vzdělání v textovém formátu, kterým bude možné předvyplnit políčko formuláře. Dalším možným předvyplněním jsou jazyky, a jejich znalost, neboť se v Matici znalostí nachází evidence jazyků a úroveň, kterou zaměstnanec vyplnil. Matice znalostí využívá Společný evropský referenční rámec, proto by bylo možné data využít i zde.

Neúplná, nebo nepřesná data lze našeptávat uživateli, který se rozhodne, zda je chce předvyplnit nebo nikoliv. Mezi tato políčka patří Technologie a Domény z Katalogu referencí a Matice znalostí a Pracovní zkušenosti z Katalogu referencí nebo Modulu vykazování. Nicméně tato data nejsou tolik spolehlivá neboť nemají žádný pevný bodový rámec, dle kterého by uživatel vyplnil jednotlivé znalosti, proto budou sloužit jako nápověda.

Dalším možným rozšířením je možnost vybrat kategorie, nebo projekty, které uživatel chce vygenerovat do životopisu, neboť jsou v daný moment více relevantní než jiné, ale chce mít možnost v další verzi životopisu navrátit vynechané technologie. Toto by mohlo být docíleno přidáním checkboxu ke relevantním políčkům a do databáze přidat hodnotu, zda jsou části aktivní, nebo nikoliv.

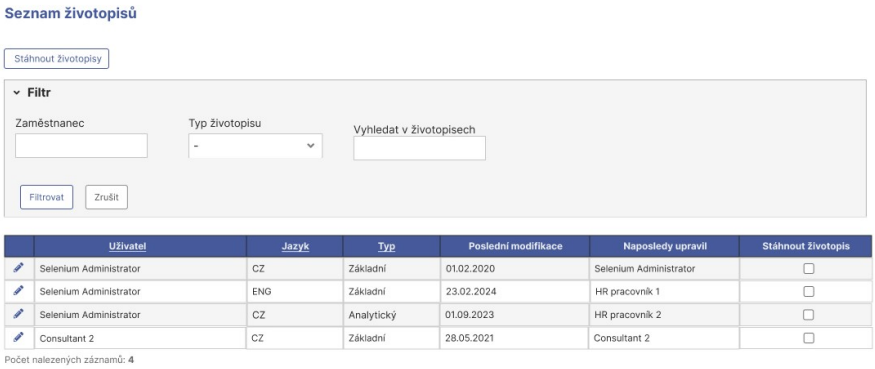
2.6.2 Obrazovka seznamu životopisů

Obrazovka byla navržena pro prezentaci přínosu a možného rozšíření modulu. Návrh byl proveden stejně, jako obrazovka tvorby rozvrhu. Byl konzultován a mělo by být úplný, i když obrazovka nebude implementována v rámci diplomové práce.

Obrazovka bude poskytovat výčet životopisů, které odpovídají zadaným filtrům. Ze seznamu si bude možné stáhnout jednotlivé životopisy, nebo se prokliknout na úpravu jednotlivých životopisů. Výřez finálního návrhu je vyobrazen na obrázku 2.5, celý návrh je k nahlednutí v příloze B.3.

2.6.2.1 Autorizační omezení

Každý uživatel bude schopen vidět **pouze vlastní** životopisy. **Zaměstnanci oddělení lidských zdrojů, manažeři a pracovníci obchodního oddělení** budou schopni vidět **všechny** životopisy.



Obrázek 2.5: Výřez návrhu obrazovky seznamu životopisů-

2.6.2.2 Rozbor filtrů

V této sekci jsou popsány filtry, podle kterých lze v životopisech vyhledávat informace.

- **Zaměstnanec**
  - Nalezne životopisy patřící danému zaměstnanci. Políčko může být jakýkoliv textový řetězec. V budoucnu je možné rozšířit o našeptávání z databáze uživatelů.
- **Typ životopisu**
  - Vyfiltruje životopisy s daným typem. Políčko bude číselník s dostupnými typy.
- **Vyhledat v životopisech**
  - Fulltextové vyhledávání informací v životopisech. Může se jednat o libovolný textový řetězec. Vyhledání probíhá přes celý životopis, je možné vyhledat technologie, domény, projekty, ale i dovednosti.
- **Ikona tužky u životopisu**
  - Kliknutím na ikonu se otevře okno úpravy životopisu.
- **Checkbox u životopisu**
  - Umožňuje stáhnout několik životopisů najednou. Po kliknutí na tlačítko *Stáhnout životopisy* budou označené životopisy staženy ve formátu ZIP.

Kapitola 3

Implementace

Práce na C# a Angular části systému byla realizována ve spolupráci s Profis týmem. Pravidelně se konaly interní stand-up meetingy, informativní schůzky po vydání nových verzí, ukázkové demonstrace generátoru, konzultace specifikací a revize kódu. Zvláštní důraz byl kladen na co nejpečlivější začlenění používaného vývojového procesu do implementační části diplomové práce.

Java část generátoru byla vyvinuta mimo C# Profis tým, neboť si jej bude přebírat a rozvíjet Java Profis tým. Z důvodu nedostatečných zkušeností v C# a Angularu bylo rozhodnuto, že bude dohlíženo pouze na implementaci ve zmíněných technologiích. Vývoj Javového generátoru probíhal v rámci pravidelných konzultací s odpovědnou osobou ze společnosti Profinit a byly pořádány průběžné ukázkové schůzky.

Velká pozornost byla věnována využitelnosti, udržitelnosti a rozšiřitelnosti implementace. Cílem bylo vytvořit kód, který by odpovídal konvencím týmů Profis a byl pečlivě zdokumentován jak v kódu, tak v interním systému XWiki.

3.1 Generátor

Genenerátor je naprogramován v jazyce Java verze 17 s využitím Spring Boot frameworku verze 3. Generátor se skládá ze dvou modulů. Modulu generátoru, který bude v budoucnu vložen do Java Profisu a modulu starajícího se o komunikaci, který v současné době bude sloužit pro testování integrace na C# Profis. Na úryvku 3 je možné vidět úvodní část kořenového `POM` souboru s nastavením rodičovského `POM`, pojmenování aplikace s vývojovou verzí a modulů aplikace. Implementace kódu respektuje konvence definované v Google Java Style [18]. Tento styl je zajištěn použitím přídavku do vývojového prostředí IntelliJ IDEA nazvaného google-java-format [19].

Používáné knihovny v generátoru jsou následující:

- **Spring Boot Starter Test** poskytuje závislosti usnadňující testování modulu.
- **Spring Boot Starter Web** přináší závislosti potřebné pro implementaci REST rozhraní nebo internacionalizace.
- **Spring Boot Starter Validation** je využívaný pro validaci vstupního modelu.

```
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.2.3</version>
    <relativePath/>
</parent>
<groupId>eu.profinit.profis</groupId>
<artifactId>cv-generation-module</artifactId>
<version>0.0.1-SNAPSHOT</version>
<packaging>pom</packaging>
<name>cv-generation-module</name>
<description>CV generation module</description>
<modules>
    <module>generator</module>
    <module>controllers</module>
</modules>
```

Zdrojový kód 3: Úryvek z kořenového POM souboru.

- **Apache POI OOXML** umožňuje načítat a pracovat s OOXML dokumenty.
- **Apache Commons Lang 3** poskytuje pomocné nástroje pro základní Java konstrukty – retězce, pole, listy a jiné.
- **Lombok** snižuje množství boiler plate kódu využitím anotací a generování.
- **SpringDoc OpenAPI Starter WebMVC UI** automazituje generování API dokumentace v JSON/YAML/HTML formátu z OpenAPI anotací.

3.1.1 Příprava šablony

Jedním z klíčových prvků pro správnou funkčnost generátoru je Wordová šablona, která bude plněna daty. Pro zjednodušení práce šablona obsahuje všechny důležité náležitosti jako jsou styly, statické obrázky nebo dvojí hlavičky. Marketingové oddělení při návrhu a akceptaci nové šablony využívá návrh ve Wordu, tudíž šablona ve Wordu bude vytvořena vždy. Nutnou úpravou je vymazání veškerých dat, která byla marketingovým oddělením do šablony vložena, aby byla šablona připravena pro následné plnění uživatelskými daty.

3.1.2 Modul generování

Modul generování poskytuje třídu využívající návrhový vzor fasáda, která využívá datového modelu z návrhu 2.3 a na jeho základě rozhodne v jakém jazyce a jakým generátorem bude dokument vygenerován. Výstupem generátoru je implementace rozhraní `Resource` ze Spring Frameworku – `ByteArrayResource`.

Využitý typ generátoru závisí na nastaveném řetězci ve vstupním modelu. Pro snadnou rozšiřitelnost a udržovatelnost bylo využito pojmenování `Bean`.

```
public XWPFDocument generateCV(CvModel cvModel) throws
↪  IOException {
    XWPFDocument document =
    ↪  templateProvider.getPortraitTemplate();
    CTBody ctBody = document.getDocument().getBody();
    CTSectPr ctSectPr = ctBody.getSectPr() == null ?
    ↪  ctBody.addNewSectPr() : ctBody.getSectPr();

    generationService.updateHeader(document, cvModel);
    generationService.createColumnPage(ctSectPr, 2);
    generationService.createSoftSkillsSection(document,
    ↪  cvModel);
    generationService.createLanguageSectionTable(document,
    ↪  cvModel.languages());
    generationService.createEducationSection(document,
    ↪  cvModel.education());
    generationService.createTrainingSection(document,
    ↪  cvModel.trainings());
    generationService.createTechnologySection(document,
    ↪  cvModel.technologies());
    generationService.createDomainSection(document,
    ↪  cvModel.domains());
    generationService.createSectionEnd(document, ctSectPr,
    ↪  ctBody);
    generationService.createExperiences(document,
    ↪  cvModel.projects());

    return document;
}
```

Zdrojový kód 4: Ukázka implementace generátoru.

Každý generátor musí implementovat interface `CVGenerator` a nastavit implementované třídě anotaci `Service` s parametrem značícím jméno dané `Bean`. Spring při startu aplikace provede Dependency Injection do mapy v hlavní metodě generátoru, kde klíčem je definované jméno v anotaci a hodnotou je implementace konkrétního generátoru. Při obdržení požadavku je z modelu získán požadovaný typ životopisu, podle kterého je z mapy získána odpovídající implementace. Další typy životopisů lze vkládat implementací `CVGenerator` rozhraní a přídání jména anotaci `Service`. Stávající proces není nutné měnit, nový generátor bude sám přidán do mapy generátorů.

V rámci diplomové práce byly napsány dva typy generátorů, jelikož Profinit v současné době využívá dva formáty životopisů – na výšku a na šířku. Každý z generátorů je složen z hlavní orchestrační metody, která skládá životopis z metod představujících jednolitvé části. Šablona dokumentu je otevřena pomocí Java konstruktu `try-with-resource`, který zaručuje uzavření všech definovaných zdrojů [20]. Ukázka hlavní metody jednoho z generátorů a způsob orchestrace jednotlivých částí dokumentu je k nahlédnutí na ukázce 4.

```
<wp:anchor xmlns:wp= "http://schemas.openxmlformats.org/drawin
↪  gml/2006/wordprocessingDrawing "simplePos= "0 "
↪  relativeHeight= "0 " behindDoc= "1 " locked= "0 "
↪  layoutInCell= "1 " allowOverlap= "1 ">
   <wp:simplePos x= "0 " y= "0 "/>
   <wp:positionH relativeFrom= "column ">
       <wp:posOffset>0</wp:posOffset>
   </wp:positionH>
   <wp:positionV relativeFrom= "paragraph ">
       <wp:posOffset>0</wp:posOffset>
   </wp:positionV>
   <wp:extent cx= "59" cy= "59"/>
   <wp:effectExtent l= "0 " t= "0 " r= "0 " b= "0 "/>
   <wp:wrapTight wrapText= "bothSides ">
       <wp:wrapPolygon edited= "0 ">
           <wp:start x= "0 " y= "0 "/>
           <wp:lineTo x= "0 " y= "21600 "/>
           <wp:lineTo x= "21600 " y= "21600 "/>
           <wp:lineTo x= "21600 " y= "0 "/>
           <wp:lineTo x= "0 " y= "0 "/>
       </wp:wrapPolygon>
   </wp:wrapTight>
   <wp:docPr id= "1 " name= "Drawing 0 " descr= "fileName"/>
   <wp:cNvGraphicFramePr/>
</wp:anchor>
```

Zdrojový kód 5: Ukázka OOXML definice ukotvení obrázku.

První volanou metodou je úprava hlavičky dokumentu, kde se nachází jméno, přijmení, pozice a fotografie. Struktura hlavičky je jiná pro první a ostatní stránky. Liší se vlastnosti fotky a textu. Obrázku je nutné přidat i ukotvení v rámci stránky, aby bylo možné využít vhodné obtékání textu a obrázek nebyl na jednom řádku s textem. K vytvoření ukotvení je nutné vytvořit Anchor OOXML definici, kterou lze vidět na ukázce 5. Definice je vždy stejná s možností parametrizace, proto byla vytvořena metoda, jenž z parametrů vytvoří Anchor OOXML, které je možné vložit do objektu `CTDrawing`. Rozměry obrázku se vkládají v English Measure Unit (EMU) jednotkách, proto je nutné využít Apache POI třídu Units a převést pixely nebo cm do EMU.

Po úpravě hlavičky následuje tvorba první stránky. Úvodní stránka se skládá ze dvou sloupcového rozložení, které je možné v Apache POI naimplementovat vytvořením nových sloupců pro danou sekci. Příklad implementace lze vidět na ukázce 6.

Po úspěsném nastavení rozložení je první stránka plněna textem s definovanými styly. Ve Wordu lze využít styly definované v souboru `styles.xml` nacházející se uvnitř zazipovaného DOCX adresáře. Každý styl má svůj název, který je zobrazen v uživatelském rozhraní Wordu a `styleId` identifikátor. Tento identifikátor je využíván v OOXML pro referenci na styl, který má být při vyobrazení Wordem využit, proto je nutné v kódu nastavit právě tento

```
public void createColumnPage(CTSectPr ctSectPr, int
↪  numberOfColumns) {
    ctSectPr.addNewCols().setNum(BigInteger.valueOf(numberOfCol⌋
    ↪  umns));
    ctSectPr.addNewVAlign().setVal(STVerticalJc.TOP);
}
```

Zdrojový kód 6: Funkce pro vytvoření sloupcového rozložení stránky.

```
<w:style w:type="paragraph" w:customStyle="1" w:styleId="Odrky">
    <w:name w:val="Odrážky"/>
    <w:basedOn w:val="ListParagraph"/>
    <w:link w:val="OdrkyChar"/>
    <w:qFormat/>
    <w:rsid w:val="00F63C97"/>
    <w:pPr>
        <w:keepLines/>
        <w:numPr>
            <w:numId w:val="2"/>
        </w:numPr>
        <w:spacing w:after="200"/>
        <w:ind w:right="45"/>
    </w:pPr>
    <w:rPr>
        <w:b/>
    </w:rPr>
</w:style>
```

Zdrojový kód 7: Ukázka OOXML definice stylu.

parametr. Definice stylu je na ukázce 7.

Při tvorbě sekce je nutné vytvořit nový odstavec, nastavit mu styl, a vložit text přes objekt `XWPFRun`, který je využíván pro evidenci jednotlivých textových řetězců obsažených v odstavcích. V sekci jazyky jsou kromě textů ještě obrázky čáry vyplněné podle úrovně znalostí daného jazyka zarovnané tabulátorem. Jelikož tabulátor ve Wordu slouží k zarovnání textu na definovanou délku v centimetrech. Délka textu závisí na počtu znaků a použitém stylu. Word počítá všechny délky dynamicky při renderování dokumentu. Proto nelze pomocí OOXML zajistit správné tabulátorové zarovnání, neboť nelze spolehlivě nastavit počet tabulátorů pro každý typ vstupu. Pro zajistění správného zarovnání byla využita tabulka bez ohraničení. Příklad tvorby sekce jazyků je naznačena na ukázce 8.

Po vytvoření první strany je nutné ukončit dvou sloupcovou sekci a vytvořit tabulku zkušeností. Apache POI poskytuje komplexní nastavení tabulky, spolu s nastavením jejího stylu, upravení ohraničení, tloušťky buňek a jiné. V kódu je vytvořena metoda ve stejném duchu jako výše zmíněná metoda vytvoření sekce jazyků.

Pro podporu vícejazyčnosti je použita třída `MessageSourceAccessor`. Nad-

```
public void createLanguageSectionTable(XWPFDocument document,
↪  List<Language> languages) {
    XWPFParagraph paragraph = document.createParagraph();
    XWPFRun run = paragraph.createRun();
    paragraph.setStyle(Styles.TITLE_11.getWordStyle());
    run.setText(messageSource.getMessage(Titles.LANGUAGES.getNa⌋
    ↪  me()));

    XWPFTable table = createBorderlessTable(document);
    table.setCellMargins(0, 0, 0, 0);

    for (Language language : languages) {
      XWPFTableRow row = getNextRowFromTable(table);
      XWPFTableCell cell = row.getCell(0);
      paragraph = cell.getParagraphs().get(0);
      run = paragraph.createRun();S
      cell.setWidth("50%");
      paragraph.setStyle(Styles.LANGUAGES.getWordStyle());
      run.setText(language.language());

      cell = row.getCell(1);
      cell.setWidth("50%");
      paragraph = cell.getParagraphs().get(0);
      run = paragraph.createRun();
      File image = pictureProvider.getSpecifiedLanguageImage(la⌋
      ↪  nguage.proficiency());

      int width = 115; // cm to pix
      int height = 4; // cm to pix

      addPictureToRun(image, run, PictureType.PNG.getOoxmlId(),
      ↪  width, height);
    }
}
```

Zdrojový kód 8: Tvorba sekce jazyků.

```
private BufferedImage downScaleBufferedImage(BufferedImage
↪  originalImage, int width, int height) {
    Image downscaledImage =
    ↪  originalImage.getScaledInstance(width, height,
    ↪  SCALE_AREA_AVERAGING);
    BufferedImage downscaledBufferedImage =
        new BufferedImage(
            width,
            height,
            BufferedImage.TYPE_INT_RGB);

    Graphics2D g2d = downscaledBufferedImage.createGraphics();
    g2d.drawImage(downscaledImage, 0, 0, null);
    g2d.dispose();

    return downscaledBufferedImage;
}
```

Zdrojový kód 9: Zmenšování rozměrové velikosti obrázků.

pisy jsou definovány v souboru messages.properties se suffixem jazyka nadpisů. Generátor z jazyka dokumentu určí lokalizaci a vloží ji do `LocaleContextHolder`, který drží nastavení jazyka pro dané vlákno. V konfigurační třídě je nastavena inicializace `MessageSourceAccessor` přes konstruktor požadující pouze třídu `MessageSource`. Nastavené `Locale` je automatickz převzato z třídy `LocaleContextHolder`. Díky tomu není nutné v kódu předávat požadovanou lokalizaci, ani ji nijak získávat.

Pro účely vytváření životopisů je nezbytné oříznout fotografie zaměstnanců do specifického tvaru. V aktuální verzi knihovny Apache POI nejsou dostupné pokročilé funkce pro manipulaci s obrázky a veškeré transformace jsou prováděny přímo v rámci aplikace Word. Z tohoto důvodu byla pro manipulaci s obrázky využita standardní grafická Java knihovna AWT [21]. V budoucích verzích budou životopisy generovány automaticky z datového modelu, proto bylo rozhodnuto, že ořez obrázku bude probíhat v generátoru. Z tohoto důvodu bude uživatelské rozhraní rozšířeno o vizuální ořezávač, který zašle potřebné rozměry serveru. V generátoru je obrázek oříznut a rozměrová velikost je snížena na velikost obrázků v životopise, aby byla docílena optimální velikost vygenerovaných souborů. V průběhu implementace bylo nutné otestovat dostupné přístupy ke změně rozměrové velikosti obrázků na základě rychlosti transformace a výsledné kvality, neboť je nutné vložit to životopisu fotografii v nejvyšší možné kvalitě. Z testu vyplynulo, že nejlepší poměr kvality a rychlosti překreslení poskytuje funkce `getScaledInstance` ze třídy `Image` s využitím Area Averaging algoritmu [22]. Výslednou funkci snížení kvality obrázku lze vidět na ukázce 9.

3.1.3 Komunikační modul

V komunikačním modulu jsou implementovány dvě třídy. První z nich je kontroler, který slouží k obsluze HTTP požadavků, a druhá je třída obsahující

```
@PostMapping(
    value = "/generate-cv",
    produces = "application/vnd.openxmlformats-officedocument.w
    ↪  ordprocessingml.document")
public HttpEntity<Resource> getCv(@RequestBody CVModel cvModel)
↪  {
    return new HttpEntity<>(generator.generateCv(cvModel));
}
```

Zdrojový kód 10: Endpoint pro generování životopisů.

metodu main, která zajišťuje spuštění celé aplikace. Kontroler nabízí jediný POST endpoint, který přijímá kompletní model životopisu a vrací `HttpEntity<Resource>` s media typem podle standardů definovaných organizací Internet Assigned Numbers Authority (IANA) [23]. Dále byl přidán jednoduchý `ExceptionHandler`, který překládá validační chyby do uživatelský přívětivé odpovědi.

3.2 C# část Profisu

Implementace v C# části musí odpovídat zavedeným konvencím a z důvodu nedostatečné znalosti a maximální využitelnosti výstupů diplomové práce probíhaly denní schůzky s Profis týmem a týdenní konzultace s vedoucími Profis týmu. Byl kladen velký důraz na rozšiřitelnost a jednoduchost následných integrací.

Byla vytvořena podpora pro autorizaci, ale pravidla byla nastavena volně. Pokud bude v budoucnu nutné implementovat pokročilejší autorizaci, nebude zapotřebí zasahovat do stávajícího procesu, ale pouze změnit autorizační politiku. Bylo připraveno rozhraní pro získání číselníků a jejich mapování na datový model životopisu.

Na ukázce 11 lze vidět integraci na Java generátor z .NET části Profisu. Při implementaci byla navržena integrace na získání aktuální fotografie zaměstnanců z jiných částí systémů, nicméně bylo zjištěno, že fotografie nedisponují požadovaným rozlišením. Integrace proto byla, pro potřeby diplomové práce, nahrazena nahráváním fotek z uživatelského rozhraní.

3.3 Uživatelské rozhraní

Uživatelské rozhraní je napsáno v Angularu verze 16. Komponenty byly rozpadnuty podle částí formuláře. Byly využity stávající styly, některé komponenty a služby. Objekty formulářů a validace jsou vytvořeny pomocí formulářových validátorů z balíčků `rxweb` [24]. Validace jsou hlídány v kontejnerové komponentě a zobrazeny Profis komponentou pro zajištění konzistence mezi stránkami systému.

První části stránky je komponenta s ořezávačem obrázků, kam uživatel vloží fotografii a následně ji napozicuje dle svých potřeb. Při odeslání požadavku jsou získány souřadnice vyřezávané části a odeslané na server. Jako komponenta ořezávače byl využit `ngx-image-cropper` [25].

```
public byte[] GenerateCV(CVDto dto)
{
    using HttpClient client = new();

    var payload = JsonConvert.SerializeObject(dto, new
    ↪  JsonSerializerSettings { ContractResolver = new
    ↪  CamelCasePropertyNamesContractResolver() });
    using var webRequest = new
    ↪  HttpRequestMessage(HttpMethod.Post, _cvGenerationModule
    ↪  ApiOptions.Value.CVGenerationModuleUrl)
    {
        Content = new StringContent(payload, Encoding.UTF8,
        ↪  "application/json")
    };

    using HttpResponseMessage response =
    ↪  client.Send(webRequest);
    byte[] result =
    ↪  response.Content.ReadAsByteArrayAsync().Result;

    return result;
}
```

Zdrojový kód 11: Integrace na generátor z Profisu

Druhá část formuláře se skládá z nastavení životopisu, jako je šablona, jazyk a typ životopis, a úvodních informací, jako je jméno, příjmení, dovednosti a jazyky uživatele. První integrací na získání informací využitelných pro předvyplnění je získání jména pro přihlášeného uživatele z `CurrentEmployeeInfoService`.

Poté následuje podmíněné upozornění při výběru jiného jazyka životopisu než je čeština, aby uživatelé vyplnili formulář ve zvoleném jazyce a sekce pro vyplnění vzdělání.

Další tři části se skládají ze standardních Profis tabulek pro vyplnění školení, technologií a domén. Technologie a domény obsahují možnosti výběru ze standardizovaných vstupů, aby byla zajištěna konzistence nadpisů přes firemní životopisy.

Poslední částí je formulář projektů, který se skládá z nově vytvořené komponenty výběru z ročního kalendáře a textových polí pro vložení zákazníka, názvu projektu, role, aktivit, technoligií a popisu projektu. Kliknutím na tlačítko lze přidat nebo odebrat stávající projekty a sbalit je pro udržení přehlednosti stránky.

Po klinutí na tlačítko *Generovat* jsou provedeny formulářové validace, formulářový model je namapován na datový model životopisu a odeslán na server. Po vygenerování je stažen výsledný DOCX soubor. Stránka není uzavřena, ani vyresetována, aby uživatel mohl provést úpravy a vygenerovat životopis znovu.

3.4 Statická analýza kódu

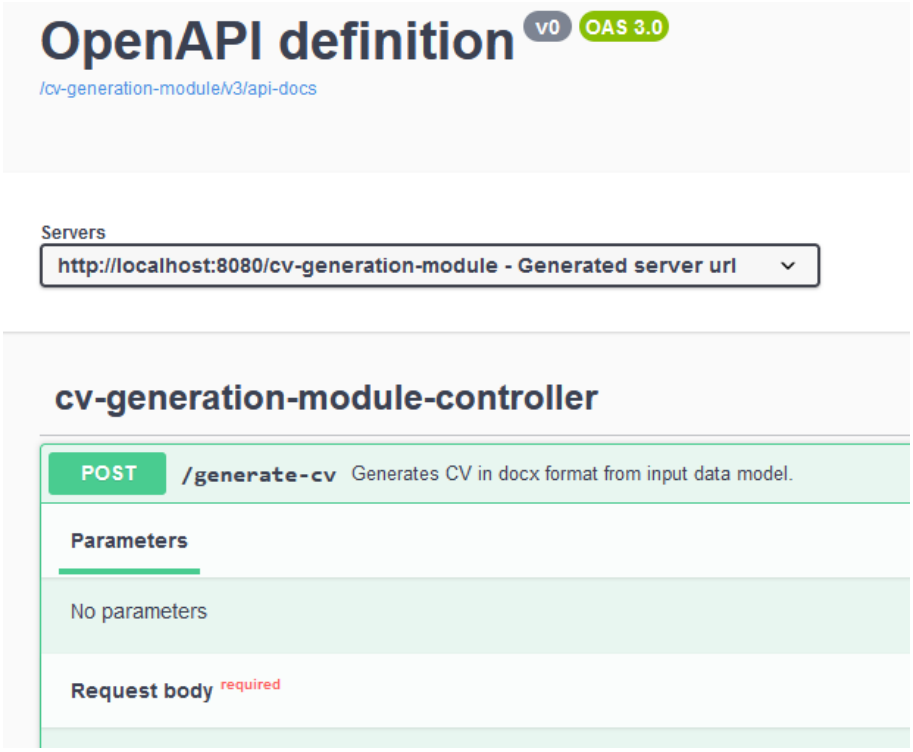
Statická analýza kódu slouží k nalezení možných zranitelnosti ve zdrojovém kódu už v implementační fázi vývoje. Analýza se snaží objevit zranitelnosti na neběžícím (statickém) kódu pomocí technik jako je Taint Analysis nebo Data Flow Analysis. V současné době je možné integrovat mnoho nástrojů statické analýzy do vývojového prostředí a získat zpětnou vazbu ihned po napsání problémového kódu. Tato okamžitá zpětná vazba je velmi cenná ve srovnání s nalezením zranitelností v pozdějších fázích vývoje. [26]

Pro zrychlení vývojového cyklu jsou využity dva nástroje statické analýzy – SonarLint do Intelijj IDEA pro analýzu Java kódu [27] a ESLint ve Visual Studio Code pro TypeScript kód [28].

3.5 Dokumentace

V rámci diplomové práce byla vypracována komplexní funkční specifikace nejen pro implementované části projektu, ale také pro celkovou problematiku. Projekt bylo nutné představit všem zainteresovaným osobám ve srozumitelné a přínosné formě, aby byl rozvoj a údržba systému, co nejjednodušší a jeho přínos, co největší. Důraz byl kladen na to, aby po přečtení specifikace čtenář porozuměl novému modulu, pochopil architektonická rozhodnutí a byl schopen efektivně spravovat systém.

Kód je obohacen o dokumentaci vysvětlující funkčnost jednotlivých funkcí a tříd. Komunikační vrstva je zdokumentována pomocí OpenAPI schématu. Kromě toho je v modulu generátoru implementován speciální endpoint, který umožňuje klientovi získat popis služby ve formátu JSON nebo prostřednictvím grafického uživatelského rozhraní. Část vygenerovaného uživatelského rozhraní s popisem služby a ukázkovým požadavkem, lze vidět na obrázku 3.1. Tento přístup poskytuje uživatelům přehled o dostupných funkcích a možnostech interakce s aplikací. V neposlední řadě byl sepsán soubor `README.md`, který obsahuje základní informace o modulu, jeho fungování, ukázkové requesty a odkazy na další informace.



Obrázek 3.1: Výřez Swagger UI pro popis Java generátoru.

Kapitola 4

Testování

V průběhu implementace byly prováděny pravidelné schůzky s odpovědnými osobami a získávána zpětná vazba na jednotlivé části systému. Jejich popis, průběh a počet odhalených chyb je rozepsán v sekcích pokrývající všechny části implementace.

4.1 Java generátor

Generátor, jako hlavní produkt diplomové práce, bylo nutné podrobit důkladnému testování, aby byl připravený pro budoucí využití. Z tohoto důvodu byly vytvořeny jednotkové testy, integrační testy, provedeny uživatelské testy vytvořených souborů a statická analýza kódu.

4.1.1 Unit testování

Prvním pilířem byla statická analýza kódu a dostatečné pokrytí jednotkovými testy. Jednotkové testy mají za úkol odhalit chyby a odchylky ve fungování systému. Testy musí být dostatečně robustní, aby je nebylo nutné upravovat při každém malém zásahu, který neovlivňuje funkčnost systému. Testy by, dle osvědčených postupů, měly pokrývat nejméně 80% kódu [29]. Testování bylo provedeno za využití knihovny JUnit 5 [30]. Testování pokrývá základní generativní metody, kdy do metody je vložen prázdný dokument a po zavolání metody je zkontrolováno, zda dokument byl rozšířen o potřebné odstavce, nebo tabulky. Dále jsou kontrolovány metody pracující s obrázky, které zaručují správnou výslednou rozměrovou velikost nebo upravený obrázek porovnáním bytů upraveného obrázku s referenčním. V neposlední řadě byly vytvořeny testy generátorů, které pouze testují, že generátor provolává požadované generativní metody se správnými parametry.

4.1.2 Uživatelské testování generovaných souborů

Generované soubory byly průběžně konzultovány s technickým konzultantem z firmy Profinit. Po několika iteracích vzešla verze, která byla předána vedoucím pracovníkům oddělení lidských zdrojů k připomínkám. V první testované verzi životopisu bylo odhaleno několik odlišností od předlohy, které bylo nutné vysvětlit a přijmout, nebo opravit. Mezi identifikované odlišnosti patří:

1. Jazyky jsou zarovnané za využití tabulky místo tabulátoru. Tato změna byla vítaným vylepšením neboť bylo uživatelsky příjemnější zarovnávat jednotlivé buňky, oproti variabilnímu počtu tabulátorů.

2. Nadpis *Pracovní zkušenosti* se nachází v hlavičce, místo v prvních řádcích tabulek zkušeností. V tomto případě bylo nutné pouze vysvětlit změnu oproti šabloně. Důvodem byla jednodušší implementace zajištění, že nadpis bude zobrazen ve správném místě na každé stránce. Při práci s formátem OOXML, nelze rozpoznat, který text bude vložen na kterou stránku. Vše Word vypočítává při renderování dokumentu, proto bylo jednodušší vložit nadpis do hlavičky. Jelikož se výsledná vizuální stránka neliší od šablony, bylo rozhodnuto, že tento přístup může být použit.

3. Mezi nadpisem *Pracovní zkušenosti* a tabulkou zkušeností se nachází příliš velká mezera. Mezera byla způsobena chybným nastavením obtékání textu u obrázku v záhlaví. Z tohoto důvodu bylo záhlaví delší a tabulka zkušeností začínala dál. Tato chyba byla odstraněna nastavením správného obtékání.

4. Posledním objeveným nedostatkem byly velké mezery mezi tabulkami zkušeností. Tyto mezery byly způsobeny nutností přidat volný odstavec mezi tabulky, aby nedošlo k jejich sloučení v editoru Word. Tento nedostatek byl konzultován a bylo rozhodnuto, že sloučení tabulek se nejeví býti problémem a bude vytvořena velká tabulka zkušeností, kdy mezery budou odpovídat šabloně.

Kromě výše zmíněných nedostatků nebylo nic dalšího odhaleno a všechny byly po testu vyřešeny úpravou generátoru.

4.1.3 Analýza implementace pomocí SonarQube

Pro zajištění dostatečné kvality kódu byl využit nástroj SonarQube [31], který byl nasazen do lokálního prostředí Docker a integrován do vývojového prostředí IntelliJ IDEA skrz SonarLint plugin a do Mavenu.

Pro využití analýzy a reportu ze SonarQube bylo nutné přidat SonarQube plugin a adresu serveru do Maven nastavení. Použité nastavení integrace do Mavenu lze vidět na ukázce 12. Následně je nutné v uživatelském rozhraní SonarQube vygenerovat token, kterým se autentizuje SonarQube plugin pro danou instanci. Analýzu je možné kdykoliv spustit Maven skriptem na ukázce 13.

SonarQube poskytuje přehledný report výstupů z bezpečnostní analýzy, duplikace kódu stejně tak hodnocení spolehlivosti, udržitelnosti, komplexity kódu a mnoho dalšího. Všechny nalezené problémy jsou ohodnoceny úrovní závažnosti. Veškeré problémy byly opraveny pro zajištění nejvyšší známky implementace a splnění kvalitativních nároků. Pro zobrazení pokrytí kódu testy je využit Jacoco plugin [32], který vytvoří report v podobě XML dokumentu. Tento dokument je poslán do SonarQube, kde je vizuálně zobrazen. Výsledek analýzy lze vidět na obrázku 4.1.

```
<pluginGroups>
    <pluginGroup>org.sonarsource.scanner.maven</pluginGroup>
</pluginGroups>
<profiles>
    <profile>
        <id>sonar</id>
        <activation>
            <activeByDefault>true</activeByDefault>
        </activation>
        <properties>
            <sonar.host.url>
                http://localhost:9000
            </sonar.host.url>
        </properties>
    </profile>
</profiles>
```

Zdrojový kód 12: Integrace SonarQube do Mavenu.

```
mvn clean verify sonar:sonar -Dsonar.projectKey=Diplomka
↪  -Dsonar.projectName='Diplomka'
↪  -Dsonar.token="generated_token" -Pcoverage
```

Zdrojový kód 13: Spouštěcí Maven skript.

Obrázek 4.1: Výsledky analýzy kódu v SonarQube.

4.2 Profis

Část systému ve frameworku .NET nebyla podrobena testováním, protože se v současné době uvnitř nenachází žádná byznysová logika a jedná se pouze o integraci mezi uživatelským rozhraním Profisu a Java generátorem. Korektní fungování bylo otestováno v rámci uživatelského testování.

4.3 Uživatelské rozhraní

Uživatelské rozhraní bylo podrobeno prvnímu testování ve fázi návrhu High-Fidelity prototypu a připomínky byly zapracovány do návrhu. První fáze uživatelského testování byla provedena technickým konzultantem firmy Profinit a vedoucími členy Profis týmu, pro zajištění použití správného grafického manuálu systému Profis. Mezi objevené problémy patří:

1. Přepnutí jazyka životopisu do anglického by mělo být doprovázeno varující hláškou, že uživatel má vyplňovat data anglicky.

2. Jednotlivé projekty (pracovní zkušenosti) by měly být pod rozbalovací tabulkou, aby je bylo možné sbalit a případně rozbalit pouze potřebné projekty.

3. Tlačítko odebrat projekt by mělo po stisku vyžadovat potvrzení o odebrání.

4. Období by mělo být zadávané pomocí Profis komponenty kalendáře.

Všechny nalezené problémy byly zanalyzovány a opraveny.

4.4 Porovnání výsledků

Výsledné dokumenty se od marketingových předloh liší výhradně v umělých zarovnáních, jako jsou prázdné odstavce nebo ručně změněná rozložení. Bohužel na tyto manuální úpravy nelze napsat generický generátor, který by bez chyb zarovnal jakýkoliv vstupní model. Cílem tedy bylo vytvořit generátor, který dokáže vygenerovat soubory s minimálními potřebnými úpravami. V této sekci jsou porovnány oba typy vygenerovaných dokumentů s poskytnutou předlohou z marketingového oddělení. Předloha je k nahlédnutí v příloze C a vygenerované dokumenty s drobnými úpravami jsou k nahlédnutí v příloze D.

4.4.1 Životopis na výšku

Prvním rozdílem je vzdálenost obrázků jazyků od jejich názvu. Tato skutečnost je dána využitím tabulky místo tabulátorů a byla potvrzena vedoucími pracovníky oddělení lidských zdrojů jako dostačující, proto není nutné s velikostí buněk hýbat. Nicméně není velkým problémem. jak v generátoru, tak ručně změnit vzdálenost, pokud to bude nutné. Dalším problémem první stránky je, že nadpis *Technologie* není na nové stránce v úrovni nadpisu *Dovednosti*, v šabloně k tomu byly použity prázdné odstavce, proto je nutné přidat dva prázdné odstavce i v prázdném dokumentu, tímto způsobem se nadpis zarovná na správnou úroveň.

Stránky projektů je liší nepatrně v záhlaví, protože v jméno a pozice zaměstnance byla vepsána do textového pole, což není snadno editovalený způsob vepsání textu do DOCX dokumentu. Byl proto využit standartní text a z toho důvodu se rozložení textu záhlaví drobně liší. Tato změna také byla potvrzena na společných schůzkách neboť měnit text v textovém poli vyžaduje větší manuální úpravy než v odstavci. Také do hlavičky byl přesunut nadpis *Pracovní zkušenosti*, který byl v šabloně v prvním řádku tabulky. Tímto způsobem je zajištěno, že nadpis bude na každé stránce, což byla vítáná změna. Jedinou nutnou úpravou je přidání nových řádků, pokud popis projektu není na stejné stránce jako úvod projektu. Jedná se ale pouze o jednotky prázdných odstavců, které by bylo nutné i při manuální tvorbě přidat.

Životopis na výšku se vyznamně neliší od jeho předlohy, k dosažení podobného výsledku je nutné přidat pouze pár prázdných odstavců.

4.4.2 Životopis na šířku

Jednotlivé části první stránky životopisu na šířku jsou generované stejnými funkcemi, jako životopisu na výšku, proto jsou zde stejné rozdíly. Soupis jazyků

využívá tabulku místo tabulátorů a je nutné přidat několik prázdných odstavců k docílení stejného zarovnání jako v předloze.

Stránky pracovních zkušeností mají opět nadpis v záhlaví, nicméně nyní není nutné přidávat žádné prázdné odstavce. Projekty zabírají pouze jeden řádek tabulky, který se přesune na další stránku, pokud se celý nevejde na konec stránky. Jediným rozdílem je upravená velikost tabulky v předloze, která zajistila, že pracovní zkušenosti budou zabírat pouze dvě stránky, v případě vygenerovaného dokumentu to jsou stránky tři. Pokud by uživatel chtěl mít tři projekty na stránce, musel by zmenšit velikost jednotlivých řádků.

Nicméně ani vygenerovaný životopis na šířku nedisponuje zásadními rozdíly a jeho úprava k esteticky přijatelnému výsledku vyžaduje pouze pár drobných úprav.

Kapitola 5

Rozšíření

V rámci diplomové práce je implementován generátor profesních životopisů. Dále je naintegrován do stávajícího systému Profis a jeho uživatelského rozhraní. Tímto proběhlo ověření konceptu a je ukázáno, že rozšíření systému má smysl. V této kapitole jsou popsány možná rozšíření modulu generování profesních životopisů pro lepší podporu a zefektivnění agendy životopisů oddělení lidských zdrojů.

5.1 Modul generátoru

Generátor v současné době generuje dokumenty ve dvou jazycích a formátech. Tímto jsou pokryty požadavky definované při úvodních schůzkách. Nicméně z dalších schůzek bylo zjištěno, že životopisy ve formátu na šířku jsou využívány pro přípravu prezentací zákazníkům. Proto dalším možným rozšířením je přidání generátoru prezentací, který by ze zadaných modelů životopisů vytvořil prezentaci ve formát PPTX.

V neposlední řadě integrace generátoru do Java Profisu, který v době psaní práce nebyl připravený.

Jedním z dalších kroků bude také integrace generátoru na existující nástroje pro zpracování nefiremních životopisů. Díky tomu bude možné převést jakýkoliv životopis do firemní šablony s minimem manuálních úprav.

5.2 C# část Profisu

V C# části Profisu je nejdůležitejším rozšířením přidání integraci na Profis databázi a její rozšíření o datový model životopisu dle datového modelu z návrhu 2.3. V současné době je využíván ORM (Object–relational mapping) nástroj NHibernate [33], který dokáže celý proces zjednodušit. S ohledem na přínos přidání možnosti ukládání životopisů, by integrace na databázi měla mít nejvyšší prioritu v dalších fázích projektu. V návaznosti na přidání uložení životopisů do databáze je možné přidat vyhledávání skrz životopisy a jejich odstranění.

Dalším rozšířením je povolení generátoru v internetové verzi Profisu, což umožní jeho využití pro externí zaměstnance. Dalším nutným rozšířením je uložení vygenerovaných životopisů do vhodného úložiště podle zákazníků a pro-

jektů. Je možné implementovat systém notifikací, který by notifikoval nové uživatele k vytvoření životopisu, nebo oddělení lidských zdrojů, pokud uživatel žádný životopis nevyplnil.

5.3 Uživatelské rozhraní

Uživatelské rozhraní lze rozšířit o různé integrace, vizuální objekty a zcela nové obrazovky. Jedná se o soupis rozšíření objevených v rámci analýzy, nebo při uživatelském testování, která nebyla zahrnuta do první fáze vývoje.

5.3.1 Integrace uživatelského rozhraní

Obrazovku tvorby životopisu je možné rozšířit o integraci na stávající nebo plánované moduly. Mezi existující moduly patří *Katalog referencí*, ze kterého lze čerpat bez transformací a rovnou je využít k předvyplnění formuláře pracovních zkušeností. Dále je možné využít modul *Matice znalostí*, kde v současné době je možné využít pouze předvyplnění jazyků, neboť jako jediná položka disponují referenční stupnicí. Ostatní vyplněné znalostí mohou být značně subjektivní a nelze spolehlivě namapovat jejich využitelnost v životopisu. Nicméně mohou sloužit jako nápověda pro uživatele, ať už ze strany životopisu, že nevyplnil některou ze svých vysoce hodnocených znalostí nebo naopak, že znalost zadaná do životopisu není hodnocena v *Matici znalostí*. Posledním existujícím využitelným modulem je *Modul vykazování*, který obsahuje všechny činnosti, které uživatel vykázal. Bohužel se nejedná o spolehlivá data neboť neobsahují žádné informace o projektu ani aktivitách nebo využitých technologiích. Data by bylo možné využít k našeptávání uživateli, ale v současné době není možné data jednoduše agregovat do využitelné podoby.

Mezi připravované moduly se řadí *Evidence studia*, který bude obsahovat, jak komplexní číselník relevantních vzdělávacích institucí, tak dosažitelných titulů. Po dokončení *Evidence studia* bude možné předvyplnit dosažené vzdělání uživatelů a jejich tituly do formuláře. Dalším plánovaným modulem je *Evidence školení a certifikátů*, které poskytne komplexní přehled školení a certifikátů, které uživatel absolvoval. U všech těchto modulů se jedná o exaktní data, která není nutné nijak transformovat, proto se může jednat o značné zrychlení tvorby a úprav životopisů zaměstnanců.

Poslední možnou integrací, která bude vyžadovat více úprav na straně Profisu a nebyla v současné době plánovaná, je integrace na fotografie v dostatečné kvalitě. Při implementaci generátoru byla vytvořena integrace na získání fotografií uživatelů, nicméně bylo zjištěno, že nedisponují dostatečnou kvalitou a v Profisu se v současné době nenachází fotografie dostatečné kvality. Proto bude nutné zanalyzovat možnosti úložiště a přesunout fotografie ze síťových disků do Profisu. Poté budou fotografie dostupné v dostatečné kvalitě, bude možné je převyplnit ve formuláři.

5.3.2 Obrazovka tvorby životopisu

Obrazovku tvorby životopisu bude nutné upravit po přidání databázové integrace. Vedle tlačítka *Generovat* bude přidáno tlačítko *Uložit*, které uloží vyplněný životopis do databáze. Po uložení životopisu do databáze bude možné na

stejné obrazovce životopisy i upravovat a pomocí zaškrtávacích polí vybrat, co se má vygenerovat a co pouze uložit do databáze.

Posledním možným rozšířením je rámcové zobrazení rozvržení DOCX dokumentu před jeho plném vygenerování. Nicméně jelikož při generování není nutné dokument ukládat do databáze, ani nejsou data ve formuláři vymazána, může si uživatel vygenerovat dokumentů několik a upravit data ve formuláři podle potřeb. Z tohoto důvodu má tato funkce velmi nízkou prioritu.

5.3.3 Nové obrazovky generátoru

Agenda životopisů je v rámci firmy velmi rozsáhlá a další fáze implementace by měly obsahovat obrazovky, které pokryjí největší množství požadovaných funkcionalit. Proto bude nutné přidat obrazovku přehledu uložených životopisů s možností filtrací a vyhledávání podle návrhu ze sekce 2.6.2. Pro úpravu životopisů je možné přepoužít obrazovku tvorby životopisu s předvyplněnými hodnotami. Další obrazovkou je obrazovka umožňující manipulaci s číselníkem technologií a domén, které budou uložené v databázi a slouží pro sjednocení sekcí v životopise. Obrazovka může být totožná s jinými obrazovkami pro manipulaci s číselníky nacházející se v Profisu. Poslední možnou obrazovkou vycházející z možných rozšíření je obrazovka pro generování PPTX souborů, u které bude nutné provést kompletní analýzu, neboť generování PPTX dokumentů se nachází ve fázi konceptu.

Závěr

Cílem této práce bylo navrhnout a implementovat modul pro firemní systém umožňující vytváření profesních životopisů v editovatelném formátu. Úkol byl úspěšně dokončen a splnil všechny požadavky nezbytné pro integraci modulu do firemního prostředí. Pro dosažení tohoto cíle bylo nutné pořádat schůzky se zainteresovanými osobami s cílem zaručit použitelnost a rozšiřitelnost modulu, ačkoliv některé části překročily rámec původního zadání diplomové práce. Během práce bylo důkladně dbáno na dodržení firemních požadavků na vývoj rozšíření interního systému.

V první fázi projektu bylo nezbytné shromáždit požadavky, provést dopadovou analýzu a definovat rozsah funkcionalit, které měly být implementovány v rámci diplomové práce. Bylo rozhodnuto, že kromě Java generátoru bude vytvořena obrazovka pro tvorbu životopisů a její integrace přes C# část Profisu do generátoru. Dále byly provedeny technické analýzy pro správný návrh a implementaci generátoru.

Spolu s modulem generátoru byly navrženy úpravy v existujících částech systému Profis, jako jsou možné rozšíření obrazovek, integrace a úpravy v databázi. Všechny tyto návrhy byly konzultovány a byl představen plánovaný budoucí stav agendy životopisů.

Během implementace byly pravidelně pořádány denní schůzky s Profis týmem a ukázky implementace byly prezentovány vedoucím pracovníkům. Java generátor byl navržen jako modulární aplikace, která umožňuje snadnou integraci do stávajícího Java Profisu. Pro generování dokumentů byla využita knihovna Apache POI, která pracuje s OOXML reprezentací DOCX dokumentů. Generátor je schopen vygenerovat dva typy životopisů, jak bylo definováno v úvodních schůzkách. Modul podporuje anglickou a českou lokalizaci a nabízí jednoduché manipulace s fotografiemi, což umožňuje minimalizaci velikosti výsledných životopisů. Velký důraz byl kladen na rozšiřitelnost, což umožňuje přidání nového jazyka bez zásahu do kódu a implementaci nového typu dokumentu pouze přidáním implementace rozhraní, aniž by bylo nutné zasahovat do existujícího procesu.

Generátor byl plně otestován pomocí jednotkových testů a podroben statické analýze kódu s cílem zajistit dostatečnou kvalitu. Bylo provedeno uživatelské testování, které zahrnovalo testování uživatelského rozhraní a celého procesu tvorby životopisu, stejně jako testování vygenerovaných dokumentů. Identifikované problémy byly opraveny a výsledné životopisy se příliš neliší od

původní předlohy. Hlavní rozdíly spočívají v estetickém zarovnání, které vyžaduje zásah uživatele. Výsledky byly pozitivně přijaty jak zadavatelem, tak vedoucími pracovníky oddělení lidských zdrojů.

Ve finální fázi diplomové práce byl vytvořen soupis potenciálních rozšíření a funkční specifikace stávající implementace. Kromě toho byl kód doplněn o komentáře v Javadocu a vytvořena dokumentace generátoru, která má za cíl usnadnit budoucí rozšíření a údržbu aplikace.

Literatura

[1] Tutorials Point India Private Limited: *Requirement Gathering Techniques*. [online], 2024, [cit. 2024-02-11]. Dostupné z: https://www.tutorialspoint.com/business_analysis/business_analysis_requirement_gathering_techniques.htm

[2] Indeed: *What Are Requirement Gathering Techniques?* [online], říjen 2022, [cit. 2024-02-11]. Dostupné z: https://ca.indeed.com/career-advice/career-development/requirement-gathering-techniques

[3] JetBrains s.r.o.: *The State of Developer Ecosystem 2023*. [online], 2023, [cit. 2024-02-11]. Dostupné z: https://www.jetbrains.com/lp/devecosystem-2023/java/

[4] Broadcom Inc.: *Introduction to Spring Framework*. [online], 2024, [cit. 2024-02-12]. Dostupné z: https://docs.spring.io/spring-framework/reference/overview.html

[5] Chauhan, P.: *Spring Boot Architecture*. [online], březen 2024, [cit. 2024-02-13]. Dostupné z: https://www.codingninjas.com/studio/library/spring-boot-architecture

[6] Strmecki, D.: *REST vs. GraphQL vs. gRPC*. [online], leden 2024, [cit. 2024-02-13]. Dostupné z: https://www.baeldung.com/rest-vs-graphql-vs-grpc

[7] IBM Corporation: *SOAP*. [online], únor 2024, [cit. 2024-02-16]. Dostupné z: https://www.ibm.com/docs/en/wasdtfe?topic=applications-soap

[8] Amazon Web Services Inc.: *What is Pub/Sub Messaging?*

[9] Microsoft: *Publisher-Subscriber pattern*. [online], 2024, [cit. 2024-02-18]. Dostupné z: https://learn.microsoft.com/en-us/azure/architecture/patterns/publisher-subscriber

[10] Johansson, L.: *What is Message Queueing?* [online], březen 2023, [cit. 2024-02-21]. Dostupné z: https://www.cloudamqp.com/blog/what-is-message-queuing.html

[11] Microsoft: *Open XML Formats and file name extensions*. [online], 2024, [cit. 2024-02-21]. Dostupné z: https://support.microsoft.com/en-us/office/open-xml-formats-and-file-name-extensions-5200d93c-3449-4380-8e11-31ef14555b18

[12] Cloud Software Group Inc.: *Jaspersoft community edition*. [online], 2024, [cit. 2024-02-23]. Dostupné z: https://community.jaspersoft.com/files/file/41-jaspersoft-community-edition/

[13] The Apache Software Foundation: *Apache POI - the Java API for Microsoft Documents*. [online], 2024, [cit. 2024-02-22]. Dostupné z: https://poi.apache.org/index.html

[14] E-ICEBLUE: *Spire.Doc for Java*. [online], 2024, [cit. 2024-02-23]. Dostupné z: cc

[15] E-ICEBLUE: *Free Spire.Doc for Java*. [online], 2024, [cit. 2024-02-23]. Dostupné z: https://www.e-iceblue.com/Introduce/free-doc-for-java.html

[16] SmartBear Software: *OpenAPI Specification*. [online], 2024, [cit. 2024-03-17]. Dostupné z: https://swagger.io/specification

[17] Council of Europe: *Common European Framework of Reference for Languages (CEFR)*. [online], 2024, [cit. 2024-03-24]. Dostupné z: https://www.coe.int/en/web/common-european-framework-reference-languages

[18] Google: *Google Java Style Guide*. [online], 2024, [cit. 2024-03-18]. Dostupné z: https://google.github.io/styleguide/javaguide.html

[19] Google: *google-java-format*. [online], 2024, [cit. 2024-04-03]. Dostupné z: https://plugins.jetbrains.com/plugin/8527-google-java-format

[20] Oracle: *The try-with-resources Statement*. [online], 2022, [cit. 2024-04-02]. Dostupné z: https://docs.oracle.com/javase/tutorial/essential/exceptions/tryResourceClose.html

[21] Oracle : *Abstract Window Toolkit (AWT)*. [online], 2024, [cit. 2024-04-16]. Dostupné z: https://docs.oracle.com/javase/8/docs/technotes/guides/awt/index.html

[22] Oracle: *Class Image*. [online], 2024, [cit. 2024-04-20]. Dostupné z: https://docs.oracle.com/javase/8/docs/api/java/awt/Image.html

[23] Makoto, M.: *vnd.openxmlformats-officedocument.wordprocessingml.document*. [online], únor 2011, [cit. 2024-04-03]. Dostupné z: https://www.iana.org/assignments/media-types/application/vnd.openxmlformats-officedocument.wordprocessingml.document

[24] Radixweb: *@rxweb/reactive-form-validators*. [online], duben 2023, [cit. 2024-04-21]. Dostupné z: https://www.npmjs.com/package/@rxweb/reactive-form-validators

[25] mawi137: *ngx-image-cropper*. [online], leden 2024, [cit. 2024-04-21]. Dostupné z: https://www.npmjs.com/package/ngx-image-cropper

[26] Dewhurst, R.: *Static Code Analysis*. [online], 2024, [cit. 2024-04-03]. Dostupné z: https://owasp.org/www-community/controls/Static_Code_Analysis

[27] SonarSource: *SonarLint*. [online], 2024, [cit. 2024-04-03]. Dostupné z: https://plugins.jetbrains.com/plugin/7973-sonarlint

[28] Microsoft: *ESLint*. [online], 2024, [cit. 2024-04-03]. Dostupné z: https://marketplace.visualstudio.com/items?itemName=dbaeumer.vscode-eslint

[29] Bansal, A.: *Best Practices for Unit Testing in Java*. [online], leden 2024, [cit. 2024-04-12]. Dostupné z: https://www.baeldung.com/java-unit-testing-best-practices

[30] The JUnit Team : *JUnit 5 User Guide*. [online], 2024, [cit. 2024-04-12]. Dostupné z: https://junit.org/junit5/

[31] SonarSource SA: *SonarQube 10.5 Documentation*. [online], 2024, [cit. 2024-04-16]. Dostupné z: https://www.sonarsource.com/products/sonarqube/

[32] Mountainminds GmbH & Co KG: *JaCoCo - Java Code Coverage Library*. [online], 2024, [cit. 2024-04-25]. Dostupné z: https://www.eclemma.org/jacoco/trunk/index.html

[33] NHibernate Community: *NHibernate*. [online], 2024, [cit. 2024-04-26]. Dostupné z: https://nhibernate.info/

Příloha A

Seznam použitých zkratek

API Application Programming Interface

Apache POI Apache Poor Obfuscation Implementation

AWT Abstract Window Toolkit

CRUD Create, Read, Update, Delete

CV Curriculum vitae

DSL Domain Specific Language

DTO Data transfer object

DOCX Document Open XML

EMU English Measure Unit

GraphQL Graph Query Language

gRPC Google Remote Procedure Call

HR Human Resources

HTTP Hypertext Transfer Protocol

IANA Internet Assigned Numbers Authority

IoC Inversion of Control

JPEG Joint Photographic Experts Group

JSON JavaScript Object Notation

JUnit Java Unit

MSSQL Microsoft SQL Server

MoSCoW metoda Must have, Should have, Could have, Won't have

OLE2 Object Linking and Embedding, version 2

ORM Object–relational mapping

PDF Portable Document Format

PNG Portable Network Graphics

PPTX PowerPoint Open XML Presentation

REST Representational State Transfer

SOAP Simple Object Access Protocol

SSRS SQL Server Reporting Services

SVN Apache Subversion

URL Uniform Resource Locator

VPN Virtual private network

WSDL Web Services Description Language

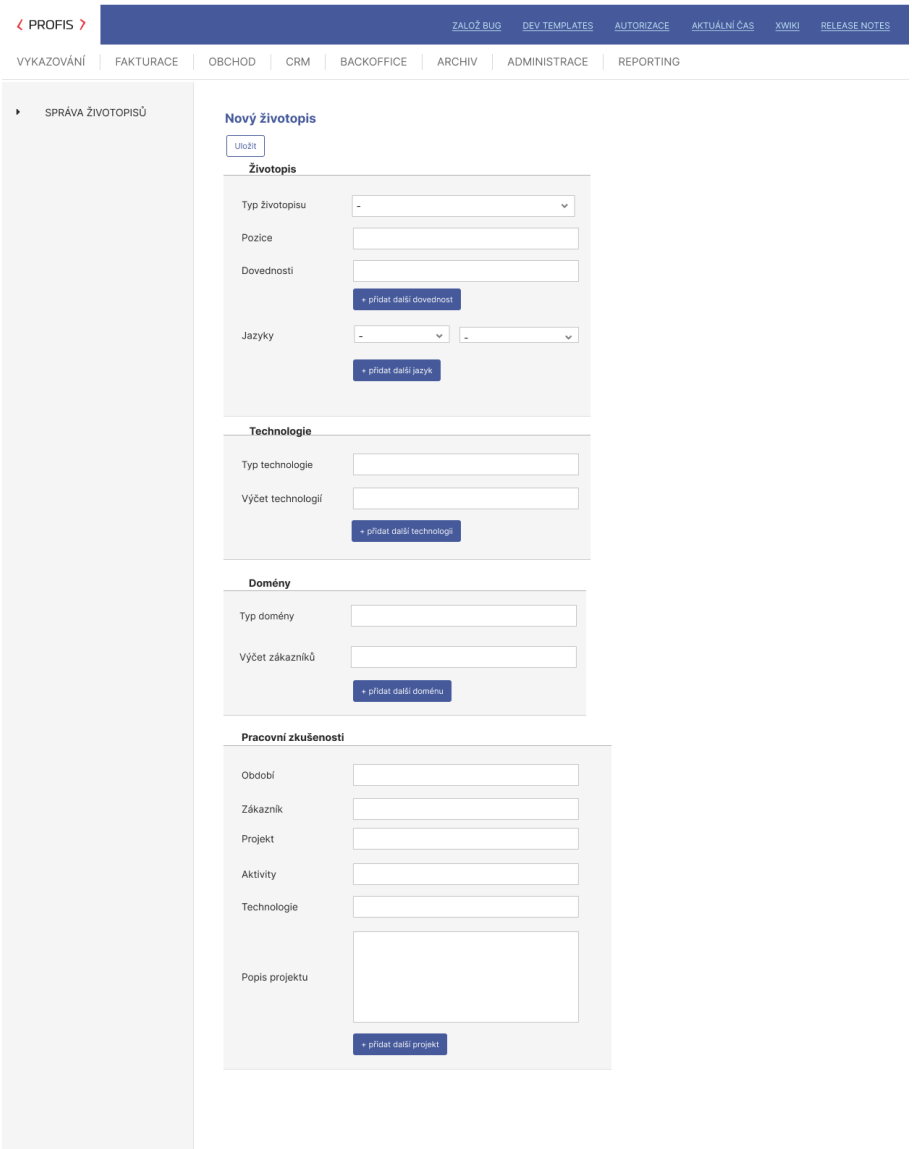
XML Extensible Markup Language

XLSX Excel Open XML Spreadsheet

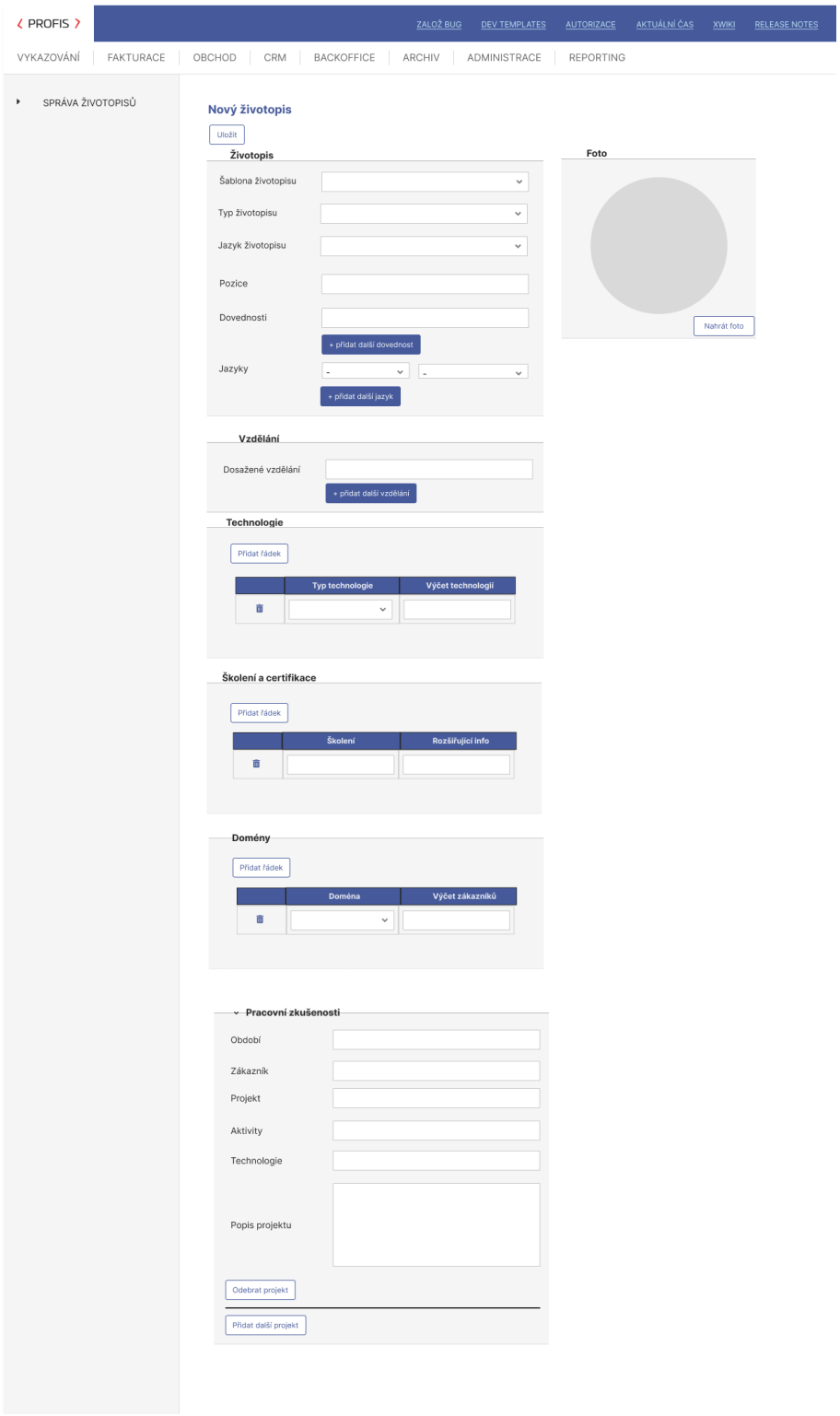
YAML Yet Another Markup Language

Příloha B

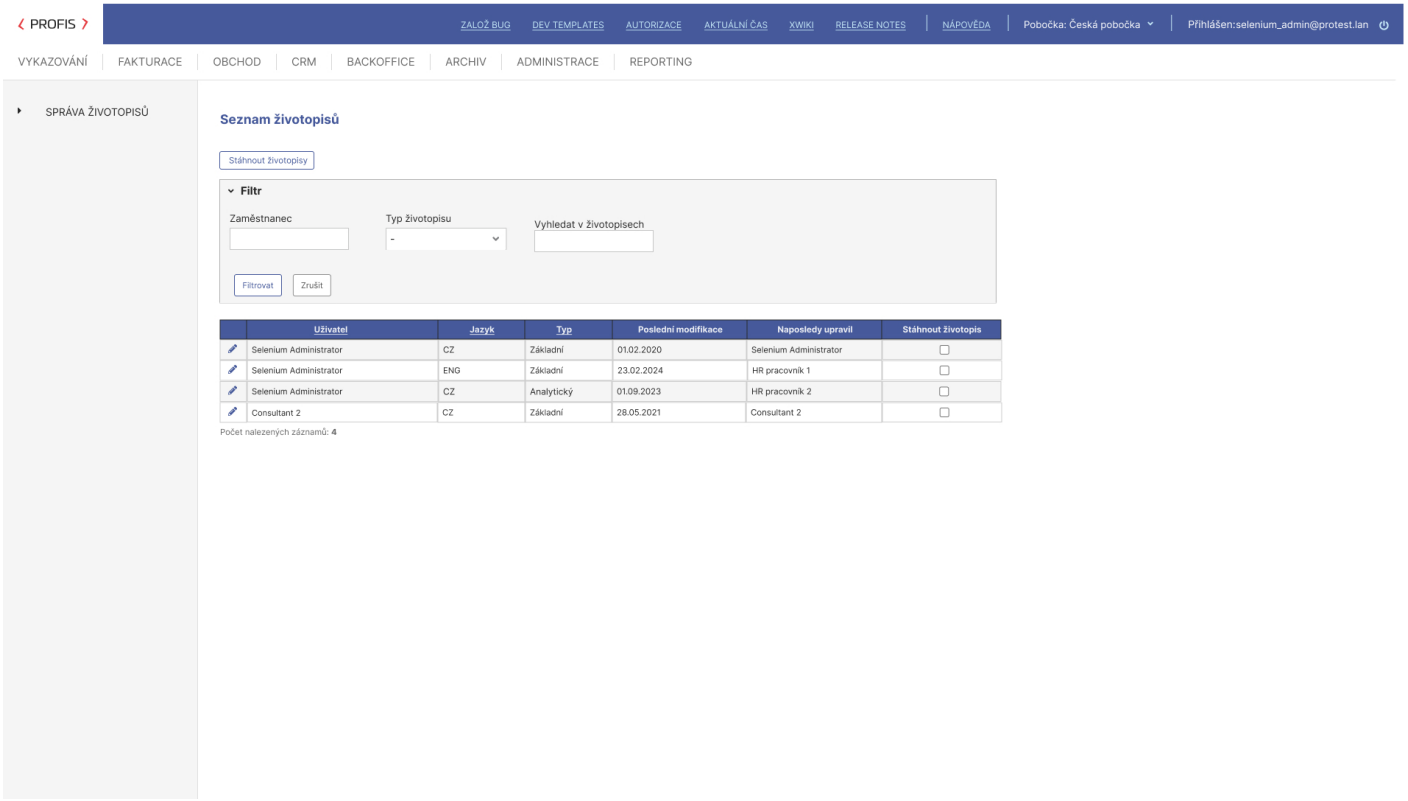
Obrazovky návrhu uživatelského rozhraní



Obrázek B.1: První návrh obrazovky tvorby životopisu.



Obrázek B.2: Druhý návrh obrazovky tvorby životopisu.



Obrázek B.3: Návrh obrazovky seznamu životopisů.

Příloha C

Předlohy životopisů

Test Testovný

Zajímavá pozice

Dovednosti | Délka praxe: 20 let

- **Řízení a koordinace projektů**
- **Vedení a řízení lidí a týmů**
- **Analytické a strukturální myšlení**
- **Analýza požadavků, aplikace technické analýzy od začátku do konce**
- **Schopnost práce v týmu i samostatně**
- **Skvělé komunikační schopnosti**

Jazyky

Čeština (rodný)

Angličtina

Němčina

Vzdělání

Karlova univerzita, Matematicko-fyzikální fakulta, Praha (2020 / Mgr.)

Školení a certifikace

Profinit Education: Presales, Spring, Business Intelligence and Data Warehouse, Bankovnictví II.

Software AG WebMethods (2021)

ARIS Business Architect, ARIS Business Designer (2020)

Technologie

Case nástroje

UML, ARIS Business Architect, ARIS Business Designer, Enterprise Architect, Oracle Designer, Power Designer

Industry standards

ERD, DFD, CORBA, RFC, ČBA EBPP, SOA, Web Services, JMS, XML/XSD/XSLT/XPath/ XQuery

SOA

Web Services, JMS, XML/XSD/XSLT/XPath/ XQuery

SAP

SAP R3, SAP CRM 7, SAP J2EE, HR module, ABAP, BSP, Dynpro, Sapscript, Smartforms, WebDynpro, JCo

Programovací jazyky

ASP.NET, C, C++, C#, Java SE/EE, SQL, UNIX Shell, (X)HTML, CSS, JavaScript, PHP

Frameworky & knihovny

Apache Wicket, Global Payments API, NHibernate, PayPal API, ElasticSearch

Databáze

Oracle, MySQL, PostgreSQL

Software development process

Iterative Waterall Model, Scrum, Profinit Software Engineering Minimal Practices a další

Domény

Bankovnictví a finance

Bankovní instituce, a.s., PersonalFinance, a.s

Pojišťovnictví

Velká pojišťovna, a.s.

Jiné

Paprika

Test Testovný

Zajímavá pozice

Pracovní zkušenosti

2016 – současnost

Bankovní instituce, a.s.

Role

Delivery Manager

Projekt

Lorem ipsum dolor sit amet

Aktivity

Vedení dodávek všech projektů pro zákazníka

Technologie

REST API, Java Enterprise Edition, Spring, Oracle, Oracle Weblogic, Google Analytics, Google, BigQuery, ElasticSearch

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat.

2016 – 2017

Bankovní instituce, a.s.

Role

Aplikační specialista, analytik, technický vedoucí a team leader

Projekt

Lorem ipsum dolor sit amet

Aktivity

API design, analýza, technický design, vývoj, dodávka projektů

Technologie

REST API, Java Enterprise Edition, Spring, Oracle, Oracle Weblogic, Google Analytics, Google, BigQuery, ElasticSearch

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

2014 – 2016

Bankovní instituce, a.s.

Role

Vedoucí týmu

Projekt

Lorem ipsum dolor sit amet

Aktivity

Funkční analýza, funkční design, vývoj Java EE, testovací analýza

Technologie

Enterprise Architect, HP Quality Center

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

Test Testovný

Zajímavá pozice

Pracovní zkušenosti

2013 – 2014

Bankovní instituce, a.s.

Role

Analytik, designér, vývojář

Projekt

Lorem ipsum dolor sit amet

Aktivity

Funkční analýza, funkční design, vývoj Java EE, testovací analýza

Technologie

Enterprise Architect, HP Quality Center

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

2013

Velká pojišťovna, a.s.

Role

Vedoucí týmu

Projekt

Lorem ipsum dolor sit amet

Aktivity

Funkční analýza, funkční design, vývoj Java EE, testovací analýza

Technologie

Enterprise Architect, HP Quality Center

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

Test Testovný

Zajímavá pozice

Dovednosti | Délka praxe: 20 let

- **Řízení a koordinace projektů**
- **Vedení a řízení lidí a týmů**
- **Analytické a strukturální myšlení**
- **Analýza požadavků, aplikace technické analýzy od začátku do konce**
- **Schopnost práce v týmu i samostatně**
- **Skvělé komunikační schopnosti**

Jazyky

Čeština (rodný)

Angličtina

Němčina

Technologie

Case nástroje

UML, ARIS Business Architect, ARIS Business Designer, Enterprise Architect, Oracle Designer, Power Designer

Industry standards

ERD, DFD, CORBA, RFC, ČBA EBPP, SOA, Web Services, JMS, XML/XSD/XSLT/XPath/ XQuery

SOA

Web Services, JMS, XML/XSD/XSLT/XPath/ XQuery

SAP

SAP R3, SAP CRM 7, SAP J2EE, HR module, ABAP, BSP, Dynpro, Sapscript, Smartforms, WebDynpro, JCo

Programovací jazyky

ASP.NET, C, C++, C#, Java SE/EE, SQL, UNIX Shell, (X)HTML, CSS, JavaScript, PHP

Frameworky & knihovny

Apache Wicket, Global Payments API, NHibernate, PayPal API, ElasticSearch

Databáze

Oracle, MySQL, PostgreSQL

Software development process

Iterative Waterall Model, Scrum, Profinit Software Engineering Minimal Practices a další

Vzdělání

Karlova univerzita,
Matematicko-fyzikální fakulta,
Praha (2020 / Mgr.)

Školení a certifikace

Profinit Education: Presales, Spring, Business Intelligence and Data Warehouse, Bankovnictví II.

Software AG WebMethods (2021)

ARIS Business Architect, ARIS Business Designer (2020)

Domény

Bankovnictví a finance

Bankovní instituce, a.s., PersonalFinance, a.s

Pojišťovnictví

Velká pojišťovna, a.s.

Jiné

Paprika

Test Testovný

Zajímavá pozice

Pracovní zkušenosti

2016 – současnost

Bankovní instituce, a.s.

Role

Delivery Manager

Projekt

Lorem ipsum dolor sit amet

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat.

Aktivity

Vedení dodávek všech projektů pro zákazníka

Technologie

REST API, Java Enterprise Edition, Spring, Oracle, Oracle Weblogic, Google Analytics, Google, BigQuery, ElasticSearch

---

2016 – 2017

Bankovní instituce, a.s.

Role

Aplikační specialista, analytik, technický vedoucí a team leader

Projekt

Lorem ipsum dolor sit amet

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

Aktivity

API design, analýza, technický design, vývoj, dodávka projektů

Technologie

REST API, Java Enterprise Edition, Spring, Oracle, Oracle Weblogic, Google Analytics, Google, BigQuery, ElasticSearch

Test Testovný

Zajímavá pozice

Pracovní zkušenosti

2014 – 2016
Bankovní instituce, a.s.

Role
Vedoucí týmu

Projekt
Lorem ipsum dolor sit amet

Popis projektu
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

Aktivity
Funkční analýza, funkční design, vývoj Java EE, testovací analýza

Technologie
Enterprise Architect, HP Quality Center

2013 – 2014
Bankovní instituce, a.s.

Role
Analytik, designér, vývojář

Projekt
Lorem ipsum dolor sit amet

Popis projektu
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

Aktivity
Funkční analýza, funkční design, vývoj Java EE, testovací analýza

Technologie
Enterprise Architect, HP Quality Center

2013
Velká pojišťovna, a.s.

Role
Vedoucí týmu

Projekt
Lorem ipsum dolor sit amet

Popis projektu
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

Aktivity
Analýza výkonnostních testů, implementace, exekuce, hodnocení a prezentace

Technologie
Apache Jmeter

Příloha D

Vygenerované životopisy

Selenium Administrator

Delivery manager

Dovednosti | Délka praxe: 11 let

- **Řízení a koordinace projektů**
- **Vedení a řízení lidí a týmů**
- **Analytické a strukturální myšlení**
- **Analýza požadavků, aplikace technické analýzy od začátku do konce**
- **Schopnost práce v týmu i samostatně**
- **Skvělé komunikační schopnosti**

Jazyky

Čeština	
Angličtina	
Němčina	

Vzdělání

Karlova univerzita, Matematicko-fyzikální fakulta, Praha (2020 / Mgr.)

Školení a certifikace

Profinit Education Presales, Spring, Business Intelligence and Data Warehouse, Bankovnictví II.

Software AG WebMethods (2021)

ARIS Business Architect, ARIS Business Designer (2020)

Technologie

Case nástroje

UML, ARIS Business Architect, ARIS Business Designer, Enterprise Architect, Oracle Designer, Power Designer

Databáze

Oracle, MySQL, PostgreSQL

Frameworky & knihovny

Apache Wicket, Global Payments API, NHibernate, PayPal API, ElasticSearch

Industry standards

ERD, DFD, CORBA, RFC, ČBA EBPP, SOA, Web Services, JMS, XML/XSD/XSLT/XPath/ XQuery

Programovací jazyky

ASP.NET, C, C++, C#, Java SE/EE, SQL, UNIX Shell, (X)HTML, CSS, JavaScript, PHP

Software development process

Iterative Waterall Model, Scrum, Profinit Software Engineering Minimal Practices a další

Domény

Bankovnictví

Bankovní instituce, a.s., PersonalFinance, a.s.

Pojišťovnictví

Velká pojišťovna, a.s.

Jiné

Paprika

Selenium Administrator

Delivery manager

Pracovní zkušenosti

2016 - současnost

Bankovní instituce, a.s.

Role

Delivery Manager

Projekt

Lorem ipsum dolor sit amet

Aktivity

Vedení dodávek všech projektů pro zákazníka

Technologie

REST API, Java Enterprise Edition, Spring, Oracle, Oracle Weblogic, Google Analytics, Google, BigQuery, ElasticSearch

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat.

2016 - 2017

Bankovní instituce, a.s.

Role

Aplikační specialista, analytik, technický vedoucí a team leader

Projekt

Lorem ipsum dolor sit amet

Aktivity

API design, analýza, technický design, vývoj, dodávka projektů

Technologie

REST API, Java Enterprise Edition, Spring, Oracle, Oracle Weblogic, Google Analytics, Google, BigQuery, ElasticSearch

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

2014 - 2016

Bankovní instituce, a.s.

Role

Vedoucí týmu

Projekt

Lorem ipsum dolor sit amet

Aktivity

Funkční analýza, funkční design, vývoj Java EE, testovací analýza

Technologie

Enterprise Architect, HP Quality Center

Selenium Administrator

Delivery manager

Pracovní zkušenosti

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

2013 - 2014

Bankovní instituce, a.s.

Role

Analytik, designér, vývojář

Projekt

Lorem ipsum dolor sit amet

Aktivity

Funkční analýza, funkční design, vývoj Java EE, testovací analýza

Technologie

Enterprise Architect, HP Quality Center

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

2013

Velká pojišťovna, a.s.

Role

Vedoucí týmu

Projekt

Lorem ipsum dolor sit amet

Aktivity

Funkční analýza, funkční design, vývoj Java EE, testovací analýza

Technologie

Enterprise Architect, HP Quality Center

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

Selenium Administrator

Delivery manager

Dovednosti | Délka praxe: 11 let

- **Řízení a koordinace projektů**
- **Vedení a řízení lidí a týmů**
- **Analytické a strukturální myšlení**
- **Analýza požadavků, aplikace technické analýzy od začátku do konce**
- **Schopnost práce v týmu i samostatně**
- **Skvělé komunikační schopnosti**

Jazyky

Čeština

Angličtina

Němčina

Technologie

Case nástroje

UML, ARIS Business Architect, ARIS Business Designer, Enterprise Architect, Oracle Designer, Power Designer

Databáze

Oracle, MySQL, PostgreSQL

Frameworky & knihovny

Apache Wicket, Global Payments API, NHibernate, PayPal API, ElasticSearch

Industry standards

ERD, DFD, CORBA, RFC, ČBA EBPP, SOA, Web Services, JMS, XML/XSD/XSLT/XPath/ XQuery

Programovací jazyky

ASP.NET, C, C++, C#, Java SE/EE, SQL, UNIX Shell, (X)HTML, CSS, JavaScript, PHP

Software development process

Iterative Waterall Model, Scrum, Profinit Software Engineering Minimal Practices a další

Vzdělání

Karlova univerzita, Matematicko-fyzikální fakulta, Praha (2020 / Mgr.)

Školení a certifikace

Profinit Education Presales, Spring, Business Intelligence and Data Warehouse, Bankovnictví II.

Software AG WebMethods (2021)

ARIS Business Architect, ARIS Business Designer (2020)

Domény

Bankovnictví

Bankovní instituce, a.s., PersonalFinance, a.s.

Pojišťovnictví

Velká pojišťovna, a.s.

Jiné

Paprika

Selenium Administrator

Delivery manager

Pracovní zkušenosti

2016 - současnost

Bankovní instituce, a.s.

Role

Delivery Manager

Projekt

Lorem ipsum dolor sit amet

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat.

Aktivity

Vedení dodávek všech projektů pro zákazníka

Technologie

REST API, Java Enterprise Edition, Spring, Oracle, Oracle Weblogic, Google Analytics, Google, BigQuery, ElasticSearch

---

2016 - 2017

Bankovní instituce, a.s.

Role

Aplikační specialista, analytik, technický vedoucí a team leader

Projekt

Lorem ipsum dolor sit amet

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

Aktivity

API design, analýza, technický design, vývoj, dodávka projektů

Technologie

REST API, Java Enterprise Edition, Spring, Oracle, Oracle Weblogic, Google Analytics, Google, BigQuery, ElasticSearch

---

Selenium Administrator

Delivery manager

Pracovní zkušenosti

2014 - 2016 Bankovní instituce, a.s. **Role** Vedoucí týmu **Projekt** Lorem ipsum dolor sit amet	**Popis projektu** Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.	**Aktivity** Funkční analýza, funkční design, vývoj Java EE, testovací analýza **Technologie** Enterprise Architect, HP Quality Center
2013 - 2014 Bankovní instituce, a.s. **Role** Analytik, designér, vývojář **Projekt** Lorem ipsum dolor sit amet	**Popis projektu** Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.	**Aktivity** Funkční analýza, funkční design, vývoj Java EE, testovací analýza **Technologie** Enterprise Architect, HP Quality Center

Selenium Administrator

Delivery manager

Pracovní zkušenosti

2013

Velká pojišťovna, a.s.

Role

Vedoucí týmu

Projekt

Lorem ipsum dolor sit amet

Popis projektu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam ut ultricies sapien. Nam ut tellus mauris. Fusce porta vestibulum lacus, ut porttitor urna luctus ut. Fusce interdum felis eu suscipit placerat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aliquam nulla nulla, facilisis at aliquam et, faucibus ut neque.

Aktivity

Funkční analýza, funkční design, vývoj Java EE, testovací analýza

Technologie

Enterprise Architect, HP Quality Center