

Využití AI při získávání informací z vnitropodnikové vědomostní databáze

Bc. Ing. Veronika Šošolíková

Diplomová práce
2025



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2024/2025

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: Bc. Ing. Veronika Šošolíková
Osobní číslo: A23446
Studijní program: N0613A140022 Informační technologie
Specializace: Softwarové inženýrství
Forma studia: Kombinovaná
Téma práce: Využití AI při získávání informací z vnitropodnikové vědomostní databáze
Téma práce anglicky: The Use of AI in Obtaining Information from an In-House Knowledge Database

Zásady pro vypracování

- Seznamte se s tématem moderních AI nástrojů na bázi velkých jazykových modelů pro zpracování dat.
- Seznamte se s provozními daty z firemní vědomostní databáze, která jsou pro zpracování uvažována.
- Zpracujte rešeršní část popisující současné možnosti AI nástrojů na bázi velkých jazykových modelů při zpracování velkých dat z firemních vědomostních databází.
- Zvolte vhodné AI nástroje a proveďte možnosti, aplikovatelnost a efektivitu při zpracování dat.
- Proveďte předzpracování vstupních dat, jejich selekci a transformaci a dále výběr vhodné AI technologie na základě požadavků a dosažených pilotních výsledků.
- Pokuste se o praktické nasazení ve zvolených scénářích zpracování znalostních a provozních velkých dat.
- Proveďte důkladnou diskuzi dosažených výsledků, popište omezení a nedostatky zvolených přístupů a zformulujte doporučení pro strukturu a formátování v budoucnu vytvářených provozních dokumentů.

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

Seznam doporučené literatury:

1. AUFFARTH, Ben. *Generative AI with LangChain: build large language model (LLM) apps with Python, ChatGPT, and other LLMs. Expert insight*. Birmingham: Packt, 2023. ISBN 978-1-83508-346-8.
2. RALTE, Zonunfeli a KAR, Indrajit. *Learn Python Generative AI: journey from autoencoders to transformers to large language models*. [India]: bpb, 2024. ISBN 978-93-55518-972.
3. OZDEMIR, Sinan. *Quick Start Guide to Large Language Models: Strategies and Best Practices for using ChatGPT and Other LLMs*. Pearson Education, 2024. ISBN 9780138199197.
4. CHOLLET, François. *Deep learning v jazyku Python – 2., rozšířené vydání: Knihovna Keras, TensorFlow*. Grada, 2023. ISBN 978-80-271-7081-4. Dostupné také z: <https://www.bookport.cz/kniha/deep-learning-v-jazyku-python-2-roz-sirene-vydani-12369/>.

Vedoucí diplomové práce: **prof. Ing. Roman Šenkeřík, Ph.D.**
Ústav informatiky a umělé inteligence

Konzultant diplomové práce: **Ing. Dušan Hrabec, Ph.D.**
Ústav matematiky

Datum zadání diplomové práce: **27. října 2024**

Termín odevzdání diplomové práce: **2. června 2025**



L.S.

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 17. listopadu 2024

PROHLÁŠENÍ AUTORA DIPLOMOVÉ PRÁCE

Beru na vědomí, že

- odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., v platném znění bez ohledu na výsledek obhajoby;
- diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému a bude dostupná k nahlédnutí;
- jedno vyhotovení diplomové práce v listinné podobě bude ponecháno Univerzitě Tomáše Bati ve Zlíně k uložení;
- na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- podle § 60 odst. 2 a 3 mohu užít své dílo – diplomovou práci – nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tj. k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá; neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji, že

- jsem na diplomové práci pracoval(a) samostatně a použitou literaturu jsem řádně citoval(a); v případě publikace výsledků budu uveden(a) jako spoluautor;
- odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou obsahově totožné.

Ve Zlíně, dne

.....
podpis autora

ABSTRAKT

Diplomová práce se zabývá možností provozu systému Retrieval-Augmented Generation (RAG) v plně lokálním prostředí bez nutnosti odesílání dat mimo podnik. Cílem práce bylo ověřit, zda lze pro tento účel využít i dostupné menší jazykové modely. V rámci práce byla vytvořena vlastní RAG aplikace, která umožňuje načítání dokumentů ve formátu PDF, Markdown a DOCX, jejich rozdělení na chunky, vektorové vyhledávání a generování odpovědí na dotazy.

Experimenty probíhaly na syntetických dokumentech simulující provozní směrnice. Testovaly se různé konfigurace systému jako formát dokumentu, strategie chunkování, embedding a jazykový model. Výsledky byly hodnoceny pomocí kvantitativních a kvalitativních metrik a statisticky zpracovány.

Na základě testování byla formulována doporučení pro optimální konfiguraci RAG systému a přípravu vstupních dat. Formát Markdown se ukázal jako nejvhodnější díky své struktuře a snadné převoditelnosti do PDF s firemní šablonou. Práce dokládá, že efektivní strojové zpracování podnikové dokumentace je možné i bez využití cloudových služeb.

Klíčová slova: Retrieval-Augmented Generation (RAG), jazykové modely (LLM), sémantické vyhledávání, embedding modely, vektorové databáze, vnitropodniková vědomostní databáze, lokální AI řešení, strategie chunkování

ABSTRACT

This thesis explores the possibility of running a Retrieval-Augmented Generation (RAG) system in a fully local environment without sending data outside the organization. The aim was to verify whether smaller, readily available language models could be effectively used for this purpose. As part of the thesis, a custom RAG application was developed, enabling document upload in PDF, Markdown, and DOCX formats, chunking, vector-based retrieval, and response generation.

Experiments were conducted using synthetic documents simulating operational guidelines. Various system configurations were tested, including document format, chunking strategy, embedding model, and language model. Results were evaluated using both quantitative and qualitative metrics and statistically processed.

Based on the testing, recommendations were formulated for the optimal configuration of the RAG system and for preparing input documents. Markdown proved to be the most suitable format due to its clear structure and easy conversion to PDF using a corporate template. The thesis demonstrates that effective machine processing of enterprise documentation is achievable without relying on cloud-based services.

Keywords: Retrieval-Augmented Generation (RAG), large language models (LLM), semantic search, embedding models, vector databases, In-House knowledge database, local AI solutions, chunking strategy

Ráda bych poděkovala vedoucímu diplomové práce prof. Ing. Romanu Šenkeříkovi, Ph.D. za pomoc při výběru tématu a směřování během konzultací.

OBSAH

ÚVOD.....	15
I TEORETICKÁ ČÁST	17
1 VELKÝ JAZYKOVÝ MODEL (LLM)	18
2 PROVOZOVÁNÍ LLM	20
2.1 CLOUDOVÉ ŘEŠENÍ	20
2.2 LOKÁLNÍ PROVOZOVÁNÍ MALÝCH JAZYKOVÝCH MODELŮ	20
2.3 SHRnutí.....	22
3 RAG – RETRIEVAL – AUGMENTED GENERATION.....	23
3.1 POTŘEBA EXTERNÍ ZNALOSTI V LLM	23
3.2 PRINCIP RAG ARCHITEKTURY.....	23
3.2.1 Jednoduchý RAG	25
3.2.2 Komplexní RAG	26
3.3 STRATEGIE CHUNKOVÁNÍ	28
3.4 EMBEDDING METODY	31
4 SÉMANTICKÉ VYHLEDÁVÁNÍ	33
5 METRIKY PRO HODNOCENÍ PŘESNOSTI A KVALITY ODPOVĚDÍ V SYSTÉMU RAG.....	35
5.1 METRIKA RECALL	36
5.2 METRIKA PRECISION	36
5.3 METRIKA F1-SCORE	36
5.4 METRIKA IoU	37
5.5 METRIKA PRECISION WITH PERFECT RECALL	37
5.6 METRIKA HUMAN EVALUATION	38
6 ZMĚNY CHOVÁNÍ SYSTÉMU RAG.....	39
7 FRAMEWORKY PRO VÝVOJ RAG APLIKACÍ	41
8 VEKTOROVÉ DATABÁZE.....	43
9 VEKTOROVÉ INDEXOVÁNÍ.....	45
10 BEZPEČNOST VYUŽÍVÁNÍ AI VE FIREMNÍ PRAXI.....	46
II PRAKTICKÁ ČÁST.....	48
11 PODNIKOVÁ ZNALOSTNÍ DATABÁZE	49
11.1 POPIS DAT, KTERÁ JSOU PRO ZPRACOVÁNÍ UVAŽOVÁNA.....	50
11.2 UMÍSTĚNÍ DAT	52
12 POUŽITÝ HARDWARE.....	53
13 SYNTETICKÁ DATA PRO TESTOVÁNÍ	56

13.1	SYNTETICKÁ DATA – ZPRÁVA DIGITALIZACE DÁVEK NEMOCENSKÉHO POJIŠTĚNÍ.....	57
13.2	SYNTETICKÁ DATA – SMĚRNICE STANDARDIZACE VÝPOČETNÍ TECHNIKY	57
13.3	SYNTETICKÁ DATA – PRACOVNÍ POSTUP PROVEDENÍ TAHOVÉ ZKOUŠKY	57
13.4	SYNTETICKÁ DATA – VIZUÁLNÍ STANDARD DRESSCODE SPOLEČNOSTI	58
14	APLIKAČNÍ PROSTŘEDÍ PRO TESTOVÁNÍ RAG	59
14.1	ZÁKLADNÍ PŘEHLED MOŽNOSTÍ A OVLÁDÁNÍ APLIKACE	59
14.2	VÝBĚR A CHARAKTERISTIKA TESTOVANÝCH MODELŮ.....	61
14.2.1	Embedding model TF-IDF (Term Frequency-Inverse Document Frequency).....	61
14.2.2	Embedding model all-MiniLM-L6-v2	61
14.2.3	Embedding model intfloat/e5-base-v2	61
14.2.4	Jazykové modely	62
14.3	POPIS RAG APLIKACE	62
14.4	ARCHITEKTURA RAG APLIKACE	64
14.4.1	Modul načítání a předzpracování dokumentu	64
14.4.2	Modul chunkování textu.....	65
14.4.3	Modul tvorby embeddingů a indexace	65
14.4.4	Modul dotazování a komunikace s jazykovým modelem	65
14.4.5	Modul pro evaluaci modelů.....	66
14.4.6	Řídící funkce	66
14.4.7	Uživatelské rozhraní.....	67
15	TESTOVACÍ SCÉNÁŘ A METODIKA HODNOCENÍ.....	68
15.1	POPIS TESTOVACÍHO POSTUPU	68
15.2	METODY HODNOCENÍ	69
15.3	VÝPOČET METRIK	70
15.3.1	Cosine similarity.....	70
15.3.2	Metriky Precision, Recall a F1-score	70
15.3.3	Přítomnost dalších informací.....	71
15.3.4	Jazyková nepřesnost.....	71
15.3.5	Nesrozumitelnost odpovědi	72
15.4	NÁVRH TESTOVACÍCH OTÁZEK	72
15.4.1	Kategorie Přesné vyhledání.....	73
15.4.2	Kategorie Sémantická podobnost.....	74
15.4.3	Kategorie Tabulkové údaje	74
15.4.4	Kategorie Výčty	75
15.4.5	Kategorie Vizuální obsah	75
15.4.6	Kategorie Více větné uvažování.....	75
15.4.7	Kategorie Dotazy mimo znalosti dokumentu	76
16	STATISTICKÉ VYHODNOCENÍ VÝSLEDKŮ	77
16.1	VYHODNOCENÍ KATEGORIE PŘESNÉ VYHLEDÁNÍ.....	78
16.2	AGREGOVANÉ VYHODNOCENÍ	81

16.2.1	Hodnocení výskytu klíčových informací v odpovědích.....	82
16.2.2	Nakolik automatické metriky odpovídají lidskému vnímání hodnocení kvality odpovědí.....	84
16.2.3	Vyhodnocení jazykové kvality výstupů	86
16.2.4	Posouzení výskytu odpovědí typu „Nevím“	89
16.2.5	Statistické vyhodnocení kvality odpovědí pomocí metrik F1-score a Kvalitativní hodnocení	95
16.2.6	Analýza výkonnosti RAG aplikace podle typu dotazu a konfigurace nastavení.....	97
17	DOPORUČENÍ VHODNÉHO NASTAVENÍ RAG SYSTÉMU	101
18	DOPORUČENÍ PRO PŘÍPRAVU DOKUMENTACE PRO RAG SYSTÉMY.....	103
18.1	PRAKTICKÁ DOPORUČENÍ VYCHÁZEJÍCÍ Z TESTOVÁNÍ	103
18.2	MARKDOWN JAKO PREFEROVANÝ FORMÁT PRO FIREMNÍ DOKUMENTACI	104
18.3	OBECNÉ ZÁSADY PŘÍPRAVY DOKUMENTŮ PRO RAG SYSTÉMY.....	106
19	MOŽNOSTI HYBRIDNÍHO NEBO CLOUDOVÉHO ŘEŠENÍ.....	107
20	NÁHRADA CITLIVÝCH DAT SYNTETICKÝMI DOKUMENTY	108
	ZÁVĚR	109
	SEZNAM POUŽITÉ LITERATURY.....	111
	SEZNAM OBRÁZKŮ	116
	SEZNAM TABULEK.....	117
	SEZNAM PROGRAMŮ	118
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	119
	SEZNAM PŘÍLOH.....	120

ÚVOD

Umělá inteligence a zejména velké jazykové modely (LLM) pronikají stále více do každodenního života běžných uživatelů. Zatímco v osobní sféře se využívají pro generování textů, překlady či asistenci při programování, ve firemní praxi, a zvláště v oblasti průmyslové výroby, zůstávají jejich možnosti často nevyužité. Přitom právě zde existuje značný potenciál pro zefektivnění práce s dokumentací, předávání znalostí i podporu rozhodování.

Praktické využití LLM však naráží na dvě zásadní překážky. První spočívá v uzavřenosti jazykového modelu, který byl natrénován na obecných datech do určitého data a neobsahuje aktuální či podniková specifika. Druhým limitem je otázka bezpečnosti, protože firmy obvykle nemohou sdílet citlivé dokumenty s cloudovými službami. Tyto problémy řeší metoda Retrieval-Augmented Generation (RAG), která kombinuje výstupy LLM s vyhledáváním ve znalostní bázi podniku. Umožňuje tím generovat odpovědi na základě konkrétního kontextu bez nutnosti opětovného trénování modelu.

Cílem diplomové práce je ověřit, zda tuto RAG architekturu lze provozovat lokálně, s využitím menších jazykových modelů, které nejsou trénovány specificky pro češtinu, a přesto poskytují užitečné odpovědi při dotazování se nad podnikovou dokumentací. Pro ověření tohoto konceptu byla vytvořena vlastní aplikace, která slouží jako experimentální nástroj pro testování vlivu jednotlivých parametrů systému na kvalitu generovaných odpovědí.

Systém RAG byl testován na syntetických datech, která simulují reálné podnikové dokumenty. Práce se zaměřuje na výběr vhodných embedding modelů, jazykových modelů, strategií dělení textu a na analýzu výstupů s využitím hodnotících metrik. Výstupem práce je jednak nalezení vhodné konfigurace nastavení RAG systému a zároveň formulace doporučení pro podniky, které chtějí své provozní dokumenty začít vytvářet tak, aby byly do budoucna schopné zpracování podobnými systémy.

Diplomová práce je rozdělena na teoretickou a praktickou část. V teoretické části je představen koncept RAG a jeho klíčové komponenty, včetně strategií pro dělení dokumentů, metod pro generování embeddingů, a možností jejich uložení a vyhledávání pomocí vektorových databází. Dále jsou popsány nástroje a frameworky vhodné pro vývoj RAG systémů.

Praktická část se zaměřuje na vývoj a testování vlastní aplikace, která umožňuje lokální provoz systému RAG bez nutnosti odesílání dat do cloudu. Aplikace byla vytvářena iterativním způsobem s využitím nástroje ChatGPT, což umožnilo flexibilní vývoj a rychlé ověřování různých přístupů. V rámci experimentálního prostředí byla implementována možnost načítání PDF, MD a DOCX dokumentů, jejich rozdělování na chunky různými strategiemi, a použití několika embedding a jazykových modelů. K ověření výkonu systému byl vytvořen testovací data set otázek a referenčních odpovědí, na základě kterého byly vyhodnoceny přesnost, srozumitelnost, úplnost a další metriky kvality odpovědí.

Z výsledků vyplývá, že i malé jazykové modely dokážou ve spojení s dobře navrženou architekturou a vhodně strukturovanými dokumenty poskytovat srozumitelné a relevantní odpovědi. Řešení tak demonstruje, že lokální nasazení systému RAG je technicky realizovatelné a přináší potenciál pro efektivnější využití interního know-how. Vzhledem k univerzálnosti řešeného problému lze přístup snadno přenést i do dalších podniků, které čelí obdobným výzvám při práci s provozní dokumentací a předáváním znalostí.

I TEORETICKÁ ČÁST

1 VELKÝ JAZYKOVÝ MODEL (LLM)

Velký jazykový model (LLM – Large Language Model) je typ systému umělé inteligence (AI), který je schopen generovat text podobný lidskému na základě vzorců a vztahů, které se učí z obrovského množství dat. LLM používají techniky strojového učení jako je hluboké učení k analýze a zpracování velkých data setů což jsou knihy, články či webové stránky. Modely jsou navrženy tak, aby chápaly kontext a význam textu, a proto mohou generovat text, který je gramaticky správný a sémanticky relevantní. Modely mohou být vyškoleny v široké škále úkolů, včetně jazykového překladu, shrnutí obsahu, odpovídání na otázky a doplňování textů. LLM mohou generovat výstupy, které jsou téměř nerozeznatelné od textu vytvořeného člověkem. Přitom se LLM učí samostatně a s minimálním lidským zásahem. To představuje obrovské zlepšení oproti dřívějším modelům, které byly založeny především na pravidlech. Nemohly se samy učit a ani úspěšně řešit úkoly, na které nebyly trénovány.

[3]

Zpracování přirozeného jazyka NLP (NLP – Natural Language Processing) využívá techniky strojového učení, mimo jiné k pochopení a zpracování lidské řeči počítači. [4] Modelování jazyka vyžaduje pochopení sekvenční povahy textu, protože odlišné umístění i jednoho slova může zcela změnit význam věty. Jazyk je vysoce kontextový, což znamená, že stejné slovo může mít různé významy v závislosti na kontextu, ve kterém je použito. Například slovo „běžet“ může mít v různých kontextech různé významy: „běhat“, „řídít nebo organizovat“ či třeba „obsluhovat stroj“. Aby bylo možné přesně modelovat jazyk, musí jazykové modely analyzovat a interpretovat okolní slova, fráze a věty, aby identifikovaly nejpravděpodobnější význam daného slova. [4]

Novinkou LLM je schopnost detekovat lingvistické nuance, jako je ironie, humor, slovní říčka, sarkasmus, intonace a záměr. [4]

LLM představují jazykové modely, které rozpoznávají, shrnují, překládají, předpovídají a generují text i další typy obsahu. Označení „velké“ odkazuje nejen na rozsah dat, na kterých byly trénovány, ale i na počet jejich parametrů – u nejnovějších modelů jde často o stovky miliard. [3]

Praktické využití LLM často souvisí se zpracováním rozsáhlého množství textových dat. Zvláštní pozornost je třeba věnovat nestrukturovaným datům, což jsou data kvalitativní povahy, která nejsou uložena ve standardizované formě. Typicky jde o e-maily, poznámky, reporty nebo recenze. Často jsou tato data dostupná ve formě PDF dokumentů, které

vyžadují předzpracování, aby je bylo možné efektivně využít při vyhledávání a generování odpovědí. [3]

LLM se tak stávají postupně základem moderních aplikací, které využívají zpracování přirozeného jazyka. Jejich rostoucí schopnosti však zároveň vyvolávají otázky ohledně praktického nasazení, tj. kde a jakým způsobem mohou být tyto modely provozovány. Různé scénáře nasazení mají své výhody i omezení a volba mezi nimi závisí nejen na technických parametrech, ale i na bezpečnostních, licenčních a ekonomických požadavcích. [4] V následující kapitole jsou proto popsány dva hlavní přístupy k provozování LLM – cloudové a lokální řešení.

2 PROVOZOVÁNÍ LLM

V předchozí části byla popsána základní funkce velkých jazykových modelů a jejich schopností. Pokud je ale chceme opravdu využít v praxi, je potřeba zaměřit se i na způsob, jakým se tyto modely provozují. Využití LLM v reálném prostředí přináší celou řadu otázek od bezpečnosti a nákladů až po technické požadavky.

Obecně existují dvě hlavní cesty: buď využijeme cloudová řešení od poskytovatelů jako je OpenAI, Google, Anthropic a jiné, nebo zvolíme lokální provoz open-source modelů přímo na vlastním zařízení. [18] Každý přístup má své výhody i limity, a právě jejich srovnání je obsahem následujících podkapitol.

2.1 Cloudové řešení

Cloudová řešení LLM představují jednu z cest, kterou se podniky při potřebě implementace umělé inteligence mohou vydat. Ukládání a zpracování dat je prováděno na externích serverech, což ale vyvolává otázky ohledně soukromí a bezpečnosti dat. Z hlediska nákladů fungují cloudové LLM na principu platby za použití, což může být výhodné pro nižší a proměnlivou zátěž, avšak s vysokým objemem využití se náklady mohou navyšovat. [17]

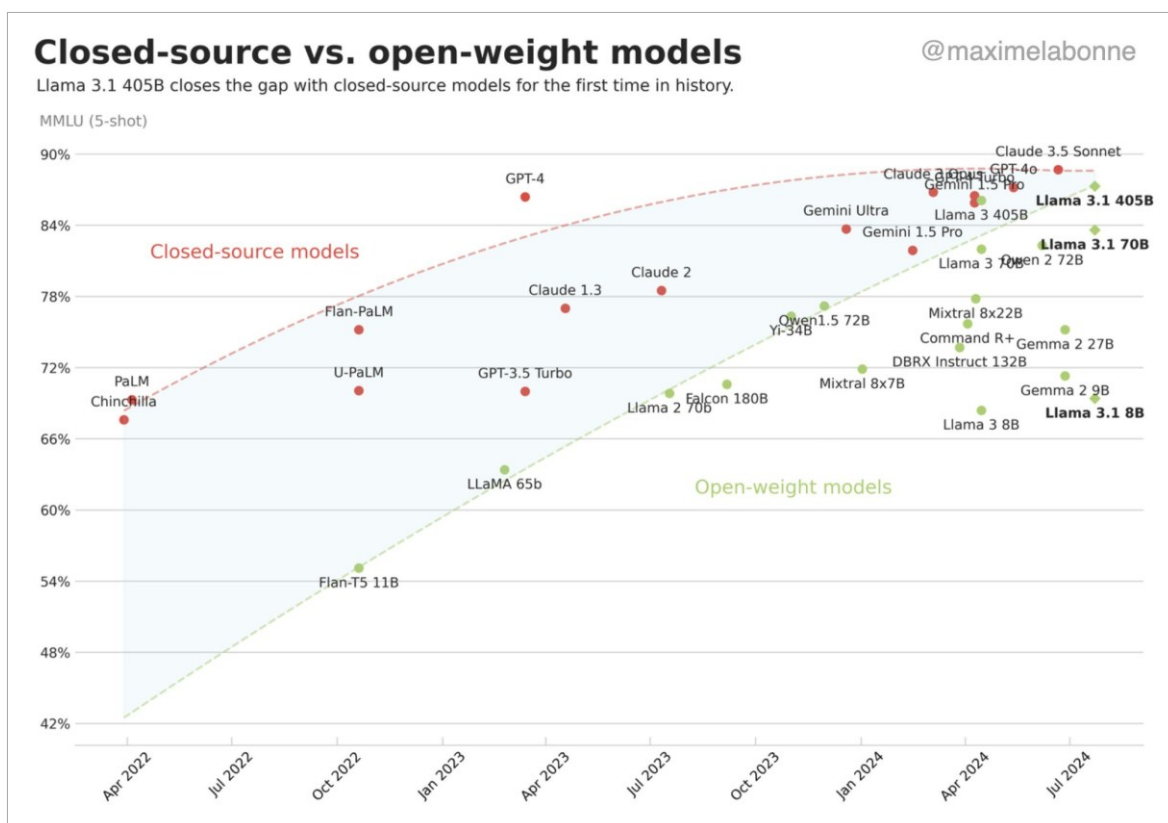
Jednou z hlavních výhod je vysoká škálovatelnost, umožňující růst současně s poptávkou bez nutnosti investic do vlastní infrastruktury. Naopak závislost na cloudových sítích může vést k vyšší latenci a obecně míra přizpůsobení je menší. Uživatel má také omezenou kontrolu nad aktualizacemi a údržbou. Pro fungování aplikace je nezbytné připojení k internetu. [17]

Hlavními faktory ovlivňujícími celkové náklady na provozování SaaS (Software as a Service) jsou cena za vstupní a výstupní token. Různí poskytovatelé (OpenAI, Anthropic, Google, Cohere) mají odlišné ceny, které jsou obvykle uváděné za 1000 tokenů. [18]

2.2 Lokální provozování malých jazykových modelů

Lokální jazykové modely představují variantu umělých inteligencí, které jsou provozovány plně na zařízení uživatele, bez nutnosti připojení k internetu nebo odesílání dat do cloudu. Na rozdíl od komerčních cloudových řešení (např. ChatGPT, Claude, Gemini, aj.), které běží na vzdálených serverech, jsou lokální modely navrženy tak, aby mohly být nasazeny přímo na osobním počítači, pracovních stanicích nebo serverech v rámci firemní infrastruktury. Tento přístup se prosazuje zejména ve firmách a institucích, které kladou důraz na ochranu

soukromí, bezpečnost dat a nezávislost na externích poskytovatelích. Díky otevřeným iniciativám v oblasti vývoje jazykových modelů (např. Meta AI, Mistral, Hugging Face, aj.) existuje celá řada modelů, které jsou dostupné pod otevřenou licencí a lze je provozovat bez omezení. [16]



Obrázek 1 Srovnání výkonu otevřených a uzavřených jazykových modelů.

(zdroj: [15])

Graf (Obrázek 1) z příspěvku Maxima Labonnea (2024) [15] ukazuje srovnání výkonu několika velkých jazykových modelů na testu MMLU (5-shot). Tento benchmark hodnotí schopnost modelů porozumět široké škále lidských znalostí napříč více než padesáti obory, jako je historie, právo, fyzika nebo medicína. Čím vyšší procentuální skóre model dosáhne, tím lepší má výsledky v odpovídání na otázky. Z grafu je zřejmé, že nový open-source model LLaMA 3 405B se výkonově velmi přiblížil komerčně dostupnému modelu GPT-4 od OpenAI, což ukazuje, jak rychle se vyvíjejí otevřené modely. [15]

Jeffrey Morgan (zakladatel Ollama) zároveň upozorňuje, že i když jsou velké modely stále důležité pro specifické úlohy, ve spoustě případů si vystačíme s menšími modely, které navíc mohou velmi dobře fungovat ve spolupráci. Malé open-source modely rychle dohánějí ty nejvýkonnější uzavřené varianty, což potvrzují i sbíhající se křivky v grafu (Obrázek 1).

Ačkoliv jsou některé z nejlepších open-source modelů stále velmi rozsáhlé, i ty menší postupně zvyšují svůj výkon. Je tak pravděpodobné, že brzy nebudou jen malé, ale i dostatečně výkonné pro reálné nasazení. [14]

Lokální nasazení přináší řadu výhod, kterými jsou:

- Plná kontrola nad daty – žádná data neopouštějí zařízení nebo firemní síť.
- Nižší provozní náklady – odpadá závislost na předplatných cloudových API.
- Možnost přizpůsobení modelu – model lze upravit, doladit (fine-tuning) nebo rozšířit o konkrétní firemní znalosti.
- Offline provoz – model může fungovat i bez přístupu k internetu. [17]

Na druhou stranu je nutné zmínit i omezení, která spočívají v:

- Vyšší nároky na hardware (zejména při práci s většími modely).
- Nutná lokální instance a konfigurace prostředí.
- Menší modely mívají omezené znalosti o světě a hůře si poradí s obecnými dotazy, pokud nejsou napojeny na vlastní data.
- Bezpečnost a správa prostředí je plně v kompetenci uživatele (aktualizace modelu, hlídání chybových stavů). [17]

2.3 Shrnutí

SaaS LLM mohou mít omezenou kontrolu v oblasti compliance a soukromí, zvláště pro uživatele v EU. Na druhou stranu open-source LLM poskytují větší kontrolu nad compliance a soukromím, protože si sami spravujete celou infrastrukturu a data.

Bezpečnostní aspekty jsou jedním z hlavních kompromisů při výběru mezi SaaS a open-source LLM. Bezpečnost a soukromí jsou relevantními faktory, které je třeba zvážit při rozhodování mezi těmito dvěma přístupy.

3 RAG – RETRIEVAL – AUGMENTED GENERATION

Ačkoli jsou dnešní jazykové modely schopny generovat vysoce kvalitní odpovědi, jejich znalosti jsou omezené na data, která byla dostupná v době tréninku. To představuje problém zejména v případech, kdy je potřeba pracovat s aktuálními, soukromými nebo specifickými firemními daty.

Jedním z elegantních způsobů, jak tyto limity překonat je metoda Retrieval-Augmented Generation (RAG). Tento přístup kombinuje schopnosti jazykových modelů s externí znalostní databází, ze které lze v reálném čase získávat relevantní informace. Model je tak obohacen o aktuální kontext, což umožňuje odpovídat přesněji, i když původně nebyl vyškolen na dané znalosti.

3.1 Potřeba externí znalosti v LLM

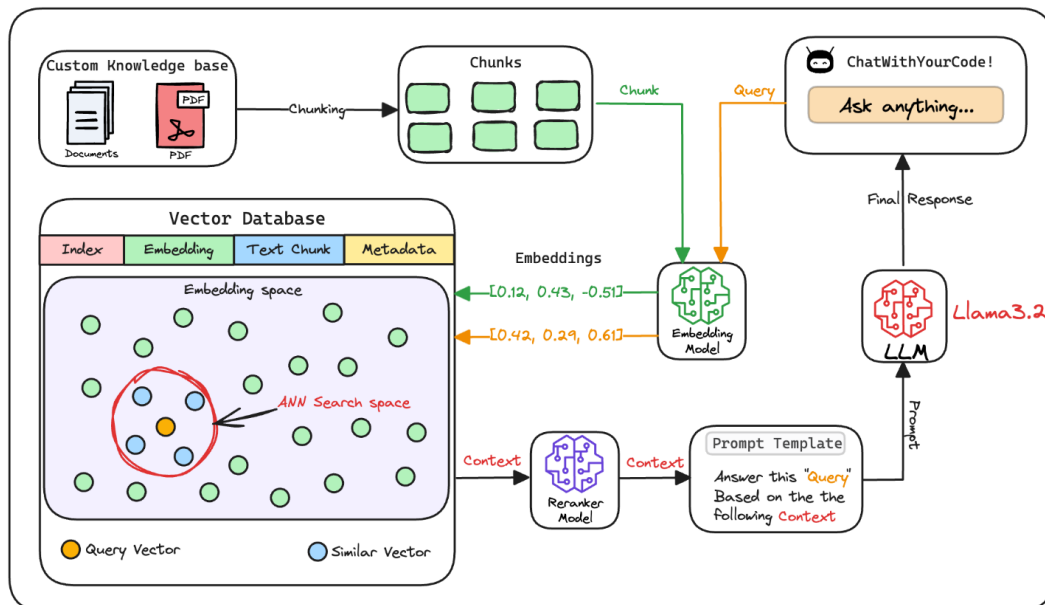
Hlavním cílem RAG aplikací je zlepšit schopnost LLM odpovídat na otázky a generovat text tím, že jazykovým modelům poskytují přístup k relevantním informacím z externích zdrojů. Na rozdíl od tradičních přístupů, kdy LLM spoléhá pouze na znalosti získané během trénování, RAG dynamicky načítá informace, které jsou sémanticky relevantní k aktuálnímu úkolu. Tyto načtené informace se poté použijí k rozšíření kontextu pro generování odpovědi. [10]

Tento přístup je užitečný pro úkoly, které vyžadují aktuální informace nebo znalosti specifické pro určitý obor či doménu, které LLM nemusely zahrnovat ve svém tréninkovém souboru. [10] RAG je vhodné použít také tehdy, když je důležitá dohledatelnost a k odpovědi potřebujeme i odkaz na dokument, kde se daná informace nachází. RAG může běžet lokálně, čehož se využívá v případech, kdy chceme jazykový model používat bezpečně offline. Například při práci nad citlivými firemními daty.

3.2 Princip RAG architektury

RAG architektura je postavena na propojení dvou hlavních komponent Retrieval a Generation. V rámci „retrieval“ dochází k dohledání relevantní informace z databáze nebo znalostní báze. Druhá komponenta „generation“ použije jazykový model pro vytvoření odpovědi na základě těchto informací. [8]

Základní RAG flow zahrnuje několik klíčových kroků (Obrázek 2), které umožňují jazykovému modelu generovat odpovědi s využitím externích znalostí, které nebyly součástí jeho tréninkových dat.



Obrázek 2 Základní schéma RAG flow

(zdroj: [8])

V následujících krocích je popsán typický postup zpracování dotazu v rámci RAG:

1. **Vytvoření chunků a tokenizace** (Create chunks and tokenize) – Načtený dokument se rozdělí na menší části, tzv. chunky. Důvodem je, že původní dokument může být příliš rozsáhlý a nevešel by se do vstupního okna embedding modelu. Navíc pokud by celý dokument měl jen jedno vektorové vyjádření, bylo by nemožné získat relevantní kontext. [8]

Chunkování často vychází z délky měřené v tokenech, nikoliv ve slovech či znacích. Tokeny lépe odrážejí skutečný vstupní rozsah, který dokáže model zpracovat. Právě proto hraje tokenizace klíčovou roli při dělení textu. Umožňuje totiž přesně určit, jak dlouhé chunky budou a zda se vejdu do požadovaného limitu pro embedding nebo samotný jazykový model. [8]

2. **Generování embeddingů** (Generate embeddings) – Každý chunk se převede na vektorové vyjádření (embedding) pomocí embedding modelu. K převodu se používají modely kontextových embeddingů, jako jsou například bi-encodery. [8]

3. **Uložení embeddingů ve vektorové databázi** (Store embeddings in a vector database) – Vytvořené embeddingy se uloží do vektorové databáze. Tato databáze slouží jako paměť pro RAG aplikaci a ukládá dodatečné znalosti, které se použijí k zodpovězení uživatelských dotazů. Vektorová databáze kromě samotných vektorů ukládá i metadata a původní obsah chunků. [8]
4. **Uživatelský vstup** (User input query) – Uživatel zadá dotaz ve formě textového řetězce. [8]
5. **Vložení dotazu** (Embed the query) – Dotaz se převede na vektor pomocí stejného embedding modelu, který byl použit pro chunky. [8]
6. **Získání podobných chunků** (Retrieve similar chunks) – Vektorizovaný dotaz se porovná s vektory uloženými ve vektorové databázi, aby se našly sémanticky nejpodobnější informace. Vektorová databáze vrátí k nejvíce podobných chunků pomocí přibližného vyhledávání nejbližších sousedů. [8]
7. **Opětovné seřazení chunků** (Re-rank the chunks) – Tento krok je realizován u pokročilejších implementací. Vyhodnotí se relevance získaných chunků vůči dotazu a seřadí se podle skóre relevance. Tím se zajistí, že nejdůležitější informace budou mít prioritu při generování odpovědi. [8]
8. **Generování finální odpovědi** (Generate the final response): Nejrelevantnější chunky se spolu s původním uživatelským dotazem vloží do šablony promptu pro LLM. Jazykový model poté zkombinuje tyto informace a vygeneruje finální odpověď, která syntetizuje informace z vybraných dokumentů. [8]

V závislosti na typu dotazů, které má systém zodpovídat, rozlišujeme dvě hlavní varianty architektury RAG a to jednoduchý (naivní) a komplexní (pokročilý). Toto rozdělení má zásadní vliv na návrh celého systému i na způsob zpracování dat.

3.2.1 Jednoduchý RAG

Varianta jednoduchého (naivního) RAG se používá u přímočarých dotazů, které vyžadují nalezení jedné konkrétní informace z dokumentu. Systém provede jednoduché vyhledání v databázi, najde relevantní chunk a předá ho jazykovému modelu ke generování odpovědi. [24]

Uživatel položí například dotaz: „*Jaké ochranné pomůcky jsou vyžadovány při manipulaci s látkou XY?*“ Systém vyhledá bezpečnostní směrnici, kde je přímo uvedeno, že

při práci s touto látkou je nutné použít rukavice, brýle a respirátor. Tuto pasáž předá jazykovému modelu, který vytvoří skutečnou odpověď.

Tento přístup je rychlý, nenáročný na výpočetní výkon a vhodný pro standardizované dokumenty, jako jsou SOP, bezpečnostní pokyny nebo montážní návody.

3.2.2 Komplexní RAG

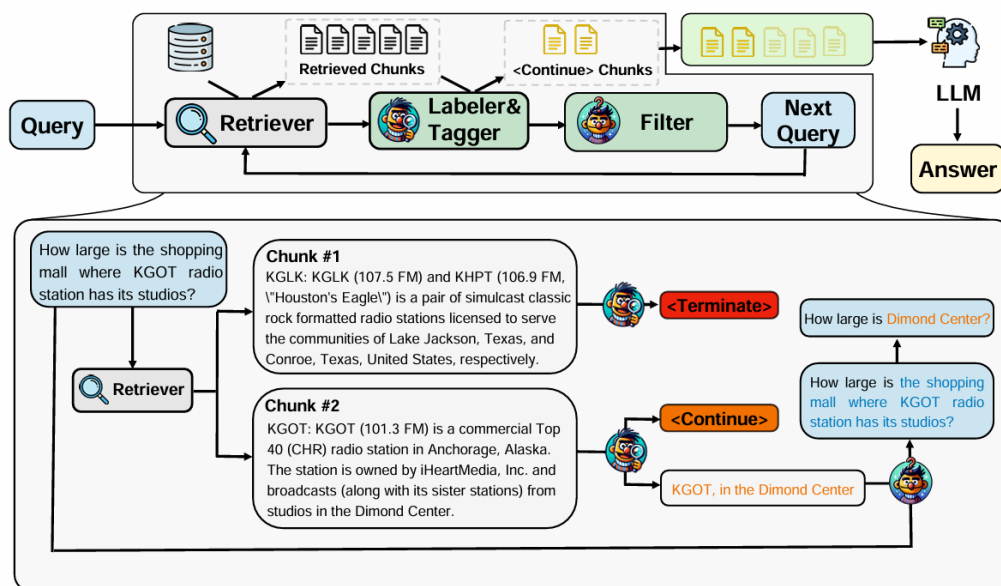
Naopak u složitějších dotazů je nutné využít tzv. multi-hop retrieval. Jedná se o víceetapové vyhledávání, které kombinuje více informací z různých dokumentů. Nalezené části jsou potom syntetizovány do jedné odpovědi. [24]

Uživatel položí například dotaz: „*Jaký je správný postup pro výměnu čerpadla typu ZT-500 při poruše v provozu B, a jaká je záruční lhůta na nový díl?*“ Systém nejprve vyhledá pracovní instrukce k výměně čerpadla ZT-500, poté najde v jiném dokumentu informace o lokalizaci provozu B a podmínkách výměny. Nakonec dohledá záruční podmínky ve skladovém nebo nákupním systému. Všechny informace zkombinuje a jazykový model vytvoří odpověď, která pokrývá celý dotaz.

Komplexní RAG je vhodný zejména pro případy, ve kterých je potřeba propojit informace z více zdrojů. Typicky se jedná o situace, kdy jsou znalosti rozděleny napříč několika dokumenty nebo oddíly v rámci jednoho rozsáhlého dokumentu. Tento přístup je ideální například pro práci s navazujícími dokumenty a návody, kde jednotlivé části poskytují doplňující nebo navazující informace. Komplexní RAG dále umožňuje zodpovídat dotazy, které vyžadují logické spojování znalostí a kombinaci více informací k vytvoření smysluplné odpovědi. [24]

Pro ilustraci multi-hop vyhledávání v rámci komplexních RAG systémů je uvedena metoda Efficient RAG, která byla navržena výzkumníky ve State Key Laboratory (2024) [25]. Tento přístup se snaží optimalizovat vyhledávání informací tak, aby bylo možné efektivně odpovídat i na složité dotazy, které vyžadují více než jeden krok k získání dat.

Efficient RAG (Obrázek 3) zavádí do procesu několik specializovaných komponent, které se starají o to, aby dotazy byly nejen zodpovězeny, ale zároveň aby byly co nejrelevantnější a nejúspěšnější z hlediska výpočetních nákladů. Standardní přístup RAG pracuje s jednotlivými chunky, které se vybírají pouze na základě podobnosti s dotazem.



Obrázek 3 Architektura Efficient RAG

(zdroj: [25])

Efficient RAG oproti tomu zavádí chytrou orchestraci vyhledávání, která je rozdělena do několika fází: [25]

1. **Labeler** – Simuluje ideální tok informací tím, že na základě známé odpovědi zjišťuje, které dokumenty nebo chunky byly pro vytvoření odpovědi skutečně důležité. Například: U dotazu: „*Jaký je doporučený postup při výměně pojistky ve stroji XYZ?*“ Labeler identifikuje, že odpověď se opírá o dva dokumenty – Standardní provozní směrnici a Servisní manuál ke stroji XYZ, konkrétně kapitolu o elektronických částech. [25]
2. **Retriever**: Na základě dat od Labeleru se trénuje Retriever, který dokáže lépe vyhledat právě ty části dokumentů, které skutečně vedou k odpovědi – nikoli jen ty, které se tváří jako podobné podle embeddingu. [25]
3. **Filter**: V této fázi se aplikuje filtrace nadbytečných nebo irelevantních informací, které se sice mohou jevit jako sémanticky podobné, ale ve skutečnosti nepřispívají k lepším odpovědím. Cílem je zúžit výběr na skutečně přínosné informace. Například: Pokud je v dokumentaci více obecných informací o pojistkách, ale jen jedna pasáž mluví konkrétně o stroji XYZ, filtr zamezí tomu, aby se ostatní části dostaly do promptu. [25]

4. **Re-ranker:** Vybrané chunky jsou dále seřazeny podle relevance vůči dotazu. To zajistí, že nejdůležitější informace se dostanou do popředí a mají největší vliv na výslednou odpověď. [25]

Díky tomuto přístupu je Efficient RAG schopný lépe pracovat s komplexními dotazy, které vyžadují spojení informací z různých částí znalostní databáze. Ačkoli je tato metoda technicky náročná a vyžaduje pokročilou orchestraci mezi jednotlivými kroky, její principy mohou být inspirací i pro jednodušší RAG aplikace. Například formou vícefázového dotazování nebo manuálního rozkladu dotazu na dílčí části. [25]

I když se některé dotazy mohou zdát jednoduché, často vyžadují složitější způsob vyhledávání. Zejména pokud odpověď závisí na více zdrojích. Je proto důležité návrh RAG systému přizpůsobit potřebám konkrétního scénáře a typu dokumentů, se kterými bude pracovat.

3.3 Strategie chunkování

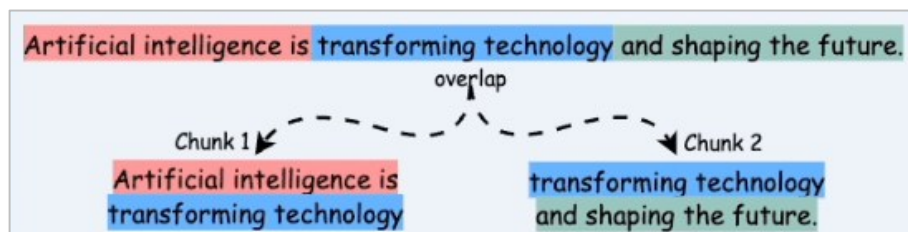
Rozdělování textu na menší části tzv. chunky je jedním z nejdůležitějších kroků při přípravě dokumentů pro RAG aplikaci. Právě chunky představují základní jednotky, se kterými systém dál pracuje. LLM si z nich vybírá při vyhledávání informací a staví z nich odpovědi. Aby model našel opravdu relevantní část textu, je důležité, aby chunky byly dotatečně malé, ale zároveň zachovaly smysl a souvislosti. [10]

Podle technické zprávy společnosti Chroma (2024) [10] se pro chunkování textu v systémech RAG uplatňuje několik klíčových strategií:

- **Character-based chunking** – Nejjednodušší metodou je chunkování založené na znacích. Tato metoda počítá stanovený počet znaků a provádí rozdělení textu v těchto intervalech. Volitelně může zahrnovat překrývání (overlap) mezi sousedními chunky. Což je dobré v tom, že se zachová kontext mezi těmito chunky. Tato metoda ale nezohledňuje sémantiku ani strukturu jazyka a může rozdělit slova nebo věty uprostřed. [10], [28]
- **Token-based chunking** – Další strategií je chunkování založené na tokenech. Tato strategie místo znaků počítá tokeny, což je pro jazykové modely přirozenější, protože tyto modely zpracovávají slova a fráze jako sekvence tokenů. Pro tokenizaci se využívá tokenizer, což je algoritmus pro kompresi, který mapuje slova nebo části slov na číselné reprezentace. Podobně jako chunkování založeného na znacích, i zde

je možné použít překrývání tokenů. Tato metoda je oproti předchozí strategii citlivější k jazykové struktuře. [10], [28]

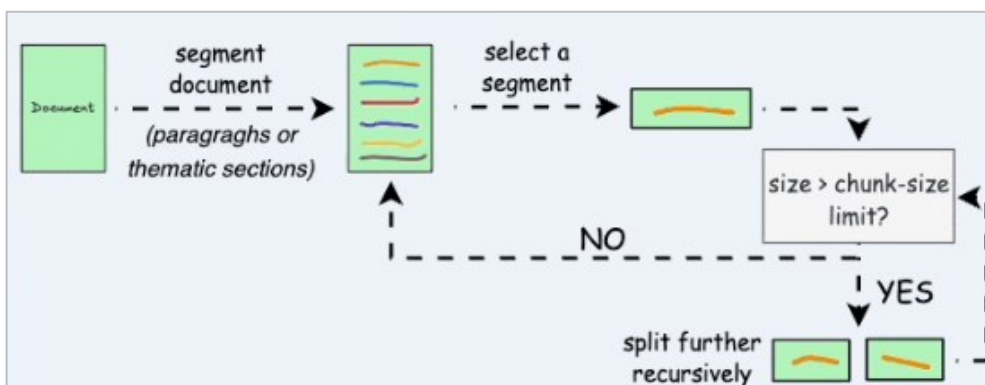
Překrývání tokenů je znázorněno na obrázku Obrázek 4.



Obrázek 4 Překrývání tokenů u Fixed-size chunking strategie

(zdroj: [10])

- **Recursive character and token-based chunking** – Přístup rekurzivního chunkování založeného na znacích a tokenech se snaží využít přirozené oddělovače v textu, jako jsou odstavce, zalomení řádků a věty. Začíná s odstavci a pokud je chunk stále příliš velký, rekurzivně se pokouší o menší oddělovače např. věty, slova, jednotlivé znaky (Obrázek 5). Prioritou je zachování přirozené struktury textu před striktním dodržáním maximální velikosti chunku. [10], [29]

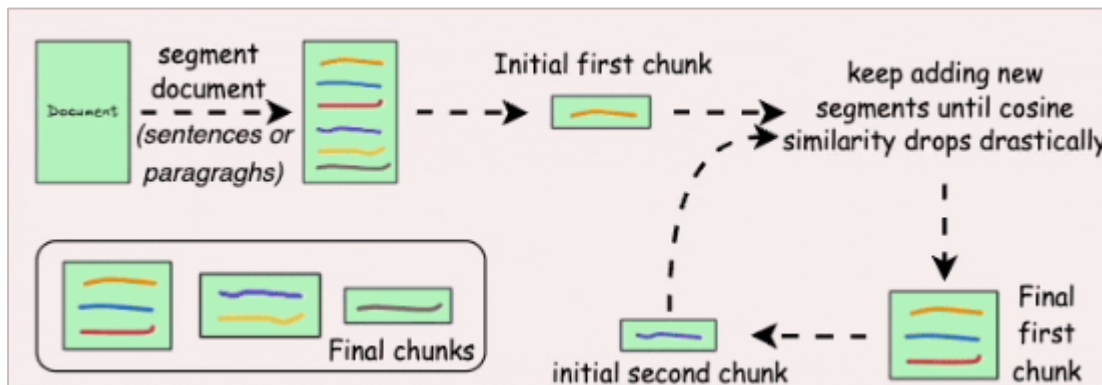


Obrázek 5 Recursive chunking

(zdroj: [31])

- **Semantic chunking** – Sémantické chunkování využívá embedding modely k nalezení sémantické podobnosti mezi chunky, aby se lépe seskupily tematicky související části textu. Text se nejprve rozdělí na menší fixní části (např. kolem 50 tokenů pomocí rekurzivního dělení) a poté se pro každou část zváží okolní kontext. Vypočítá se kosinová podobnost mezi embeddingy těchto segmentů, a pokud je vzdálenost mezi segmentem a následujícím segmentem dostatečně vysoká (nad stanoveným prahem), provede se rozdělení (Obrázek 6). ChromaDB implementovala

modifikovanou verzi, která pomocí binárního vyhledávání cílí na určitou průměrnou velikost chunku, na rozdíl od původní implementace, která mohla vytvářet chunky nepředvídatelné velikosti. [10], [29]



Obrázek 6 Semantic chunking

(zdroj: [31])

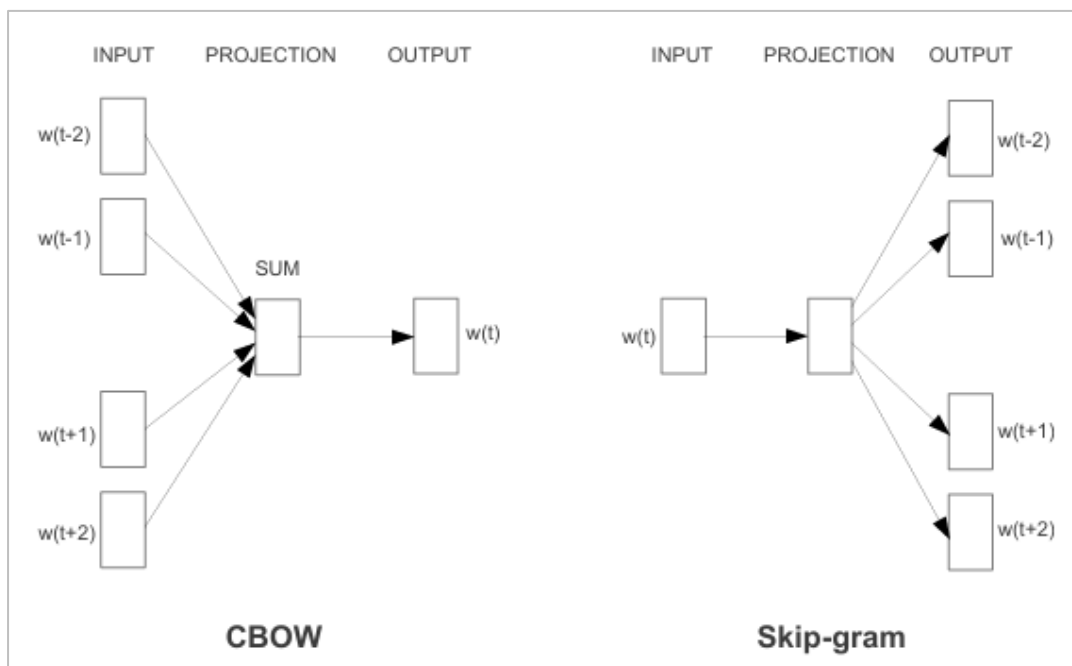
- **Cluster semantic chunking** – Clussterové sémantické chunkování je pokročilejší přístup, který bere v úvahu sémantické vztahy mezi všemi částmi textu současně s cílem najít sémanticky nejkoherentnější seskupení při dodržení omezené velikosti. Vytváří se matice podobnosti mezi embeddingy všech malých segmentů, které byly získány tradičním rekurzivním dělením. Pomocí dynamického programování se pak hledá optimální způsob seskupení těchto segmentů do chunků na základě maximalizace sémantické podobnosti uvnitř chunků. Tento přístup zohledňuje globální kontext na rozdíl od lokálního okna u standardního sémantického chunkování.[10], [30]
- **LLM semantic chunking** – Strategie LLM sémantického chunkování přímo využívá velké jazykové modely k určení nejlepšího způsobu chunkování textu. Text se nejprve rozdělí na menší části (např. kolem 50 tokenů rekurzivním dělením) a ty se pak předloží LLM s instrukcí, aby identifikoval body, kde by mělo dojít k rozdělení, aby sousední chunky měly podobné téma. LLM pak vrací ID chunků, kde by se mělo provést rozdělení. Podle ChromaDB se ukázalo, že nejlepšího výkonu se dosahuje, když LLM udává, kde by se mělo rozdělit, místo aby přímo přepisoval chunky. [10], [30]

3.4 Embedding metody

Embedding modely převádějí text na vektorové reprezentace, které zachycují sémantický význam slov. Existuje několik přístupů při tvorbě embeddingů, které se liší v tom, jakým způsobem zachycují sémantické vztahy a kontextové informace slov.

- **Frekvenční metody** – Mezi tyto metody patří například TF-IDF (Term Frequency Inverse Document Frequency), které jsou založené na myšlence, že význam slova lze odvodit od toho, jak často se dané slovo vyskytuje v textu. TF-IDF zdůrazňuje slova, která jsou častá v konkrétním dokumentu, ale vzácná v celé kolekci dokumentů. Například v dokumentu o kávě by TF-IDF vyzdvihlo slova jako „espresso“ nebo „cappuccino“. Běžná slova jako „a“ nebo „nebo“, která se vyskytují často ve všech dokumentech, by získala nízké skóre TF-IDF. Zásadní rozdíl oproti jiným metodám spočívá v tom, že frekvenční metody vytvářejí reprezentace slov na základě jejich výskytu, nikoli na základě predikce nebo společného výskytu v kontextu. [32]
- **Predikční metody** – Tyto metody se snaží zachytit sémantické vztahy mezi slovy a jejich kontextem. Příkladem může být, že se model naučí asociovat slovo „pes“ se slovy jako „štěkat“, „vrtět“ a „ocas“ z vět jako „pes štěká hlasitě“ a „pes vrtí ocasem“. Predikční modely přiřazují každému slovu jednu pevnou reprezentaci, která zahrnuje různé významy s ním spojené. [32] Mezi známé modely patří:
 - **word2vec** – Model vyvinutý společností Google v roce 2013 se dvěma základními architekturami (Obrázek 7):
 - **CBOW (Continuous Bag of Words)** – predikuje cílové slovo na základě okolních slov.
 - **Skip-gram** – predikuje okolní slova na základě cílového slova. [33]
 - **GloVe (Global Vectors for Word Representation)** – metoda vyvinutá na Stanfordově univerzitě v roce 2014, která vytváří vektory na základě statistiky společného výskytu slov v celém souboru dokumentů. [32]

Zatímco word2vec se zaměřuje na učení z bezprostředního kontextu kolem každého slova, GloVe zohledňuje, jak často se slova objevují společně v celém souboru, a na základě toho vytváří word vektory. Oba přístupy se oproti frekvenčním metodám snaží lépe vystihnout význam slov prostřednictvím predikce a kontextu.



Obrázek 7 CBOW a Skip-gram architektury

(zdroj: [33])

- Kontextové metody** – Tyto metody, využívané především v transformerových modelech, představují významný pokrok oproti tradičním word embeddingům. Zásadní rozdíl spočívá v tom, že reprezentace slova se mění v závislosti na jeho okolním kontextu. Například slovo „kolo“ bude mít odlišnou vektorovou reprezentaci ve větě „Jela na kole po lesní cestě“ než ve větě „První kolo soutěže bylo nejtěžší“. V prvním případě se jedná o dopravní prostředek, zatímco ve druhém o fázi soutěžního klání. Tato kontextová citlivost umožňuje modelům lépe zachytit jemné významové rozdíly a vztahy mezi slovy. Na rozdíl od frekvenčních a predikčních metod, které přiřazují každému slovu jeden univerzální vektor, kontextové metody generují dynamické reprezentace v závislosti na okolí. [32], [34], [35]

Metody se tedy liší v tom, zda vycházejí z pouhé četnosti výskytu slov (frekvenční metody), z predikce jejich výskytu na základě okolí (predikční metody), nebo z tvorby významu slov na základě aktuálního kontextu (kontextové metody). Predikční, a zejména kontextové metody, obecně lépe zachycují sémantické vztahy a zvládají i jazykovou mnohoznačnost.

4 SÉMANTICKÉ VYHLEDÁVÁNÍ

V RAG architektuře hraje zásadní roli koncept sémantického vyhledávání. V tradičním přístupu vyhledávání, které je založeno na klíčových slovech se porovnávají texty na základě výskytu přesných slovních spojení. Sémantické vyhledávání porovnává významové podobnosti mezi dotazem a texty ve znalostní bázi. To umožňuje systému porozumět různým formulacím téhož dotazu, vyhledávat v přirozeném jazyce a nacházet relevantní informace i tehdy, když se v dokumentech nevyskytují přesně stejná slova jako v dotazu uživatele. [11]

Když tedy mluvíme o sémantickém vyhledávání nebo sémantickém porovnávání, máme na mysli hledání nebo porovnávání textů na základě významu, a nejen na základě přesných slov. Díky tomu může například chatbot najít relevantní informace i když se uživatel zeptá úplně jinými slovy, než jsou použity ve zdrojových textech. [11]

Jednoduchý příklad dokresluje celý proces. V tabulce (Tabulka 1) jsou zobrazeny ukázky dokumentů rozdělené do bloků textu a dotaz uživatele. Z textů jednotlivých dokumentů se připraví embeddingy. Je to proces, kdy se texty převedou na sémantické vektory, což jsou číselné reprezentace významu. Embeddingy se uloží do vektorové databáze. [12]

Dotaz uživatele se také převede na embedding. Vyhledá se několik nejpodobnějších embeddingů v databázi. Vyhledává se obsah, který je významově nejbližší otázce. Dva texty mohou být sémanticky podobné i když se skládají z různých slov. Například dotazy „*Jak mohu nastavit nový senzor?*“ a „*Návod na konfiguraci detekčního zařízení*“ jsou si sémanticky podobné, i když slova jsou skoro úplně jiná. [12]

Tabulka 1 Příklad – sémantické vyhledávání pomocí embeddingů v RAG

Dokument	Text	Embedding (vektor)	Sémantické vyhledání a porovnání
A	Kalibrace senzoru se provádí pomocí tlačítka RESET.	[0.14, 0.85, -0.22, ..., 0.03]	Mluví o „kalibraci senzoru“ ANO
B	Pro nastavení detekčního zařízení je třeba vstoupit do menu konfigurace.	[0.15, 0.84, -0.20, ..., 0.01]	Mluví o „nastavení zařízení“ ANO

C	Běžná údržba zahrnuje kontrolu kabeláže a napájení.	[0.02, -0.14, 0.73, ..., -0.55]	Mluví o údržbě, což významově nesouvisí Ne
Dotaz uživatele:	Jak mohu nastavit nový senzor?	[0.13, 0.83, -0.21, ..., 0.02]	

(zdroj: sestaveno dle [11] a [12])

Systém najde dokumenty, jejichž embeddingy jsou nejbližší embeddingu dotazu. Porovnání je uvedeno v posledním sloupečku (Tabulka 1).

Vyhledaný obsah se předá jazykovému modelu jako kontext, a ten pak na základě toho vytvoří odpověď ve stylu: „Pro nastavení senzoru vstupte do menu konfigurace nebo použijte tlačítko RESET dle typu zařízení.“ [12]

V rámci sémantického vyhledávání se využívají různé typy embeddingů, které se liší podle toho, z jakých dat vycházejí a jaký typ vztahů mají zachytit:

- **Uni-modální embeddingy**

Jsou generovány z jednoho typu vstupních dat (např. z textu) a zachycují sémantický kontext v rámci dané modality. Uni-modální embeddingy se používají v úlohách specifických pro jeden typ dat. Například textové embeddingy se používají v úlohách NLP, jako je klasifikace textu, analýza sentimentu a strojový překlad. Obrazové embeddingy se potom podobně používají v úlohách jako je detekce objektů a klasifikace obrázků. [13]

- **Multi-modální embeddingy**

Generují se z více typů vstupních dat (např. textu a obrázků) a zachycují vztahy a interakce mezi různými modalitami. Multi-modální embeddingy jsou klíčové pro úlohy, které vyžadují pochopení vzájemného působení různých typů dat. Například ve videu s titulky mohou multimodální embeddingy pomoci modelu pochopit vztah mezi vizuálním obsahem a doprovodným textem. [13]

Sémantické embeddingy tak tvoří základní kámen pro efektivní vyhledávání v RAG systémech. Díky nim je možné překlenout rozdíl mezi formulací dotazu a zněním dokumentu a zajistit, že jazykový model bude pracovat s co nejrelevantnějším kontextem.

5 METRIKY PRO HODNOCENÍ PŘESNOSTI A KVALITY ODPOVĚDÍ V SYSTÉMU RAG

Pro objektivní posouzení kvality odpovědí generovaných aplikací využívající přístup RAG se využívají evaluační metriky. Tyto metriky umožňují analyzovat jednotlivé části celého RAG řetězce – od vyhledání relevantních částí textu až po finální generování odpovědi jazykovým modelem (Tabulka 2).

V této práci budou metriky využívány nejen pro experimentální porovnávání různých nastavení systému, ale také jako nástroj pro sledování dopadu jednotlivých vylepšení na výkonnost aplikace. Cílem je nejen vytvořit funkční aplikaci, ale zároveň porozumět tomu, jak konkrétní parametry a komponenty ovlivňují její výsledky.

Zvolený soubor metrik umožňuje komplexní pohled na fungování systému. Metriky Recall a Intersection over Union (IoU) se zaměřují na úspěšnost vyhledávání. Precision, F1 a Precision with perfect recall testují schopnost jazykového modelu správně využít poskytnutý kontext. Human evaluation metrika zohledňuje lidské vnímání kvality odpovědi. [20]

Tabulka 2 Evaluační metriky

Metrika	Fáze RAG	Popis
Recall	Vyhledávání	Kolik relevantních chunků bylo úspěšně nalezeno.
Precision	Vyhledávání	Měří kolik z vrácených chunků je skutečně relevantních.
F1 skóre	Obě fáze	Vyvážené skóre mezi Recall a Precision.
IoU (Intersection over Union)	Generování odpovědi	Míra překryvu mezi správnou a vygenerovanou odpovědí.
Precision with perfect recall	Generování (při známém kontextu)	Zda model dokáže správně odpovědět, pokud má všechny správné informace.
Human evaluation	Celkový výstup	Člověkem hodnocená kvalita a srozumitelnost odpovědi.

(zdroj: zpracováno dle [10], [20], [21])

5.1 Metrika Recall

Recall je metrika, která se využívá při hodnocení kvality vyhledávání relevantních informací z dokumentové báze, ze které pak jazykový model generuje odpověď.

Metrika Recall měří, jak velký podíl všech relevantních dokumentů (nebo chunků), které v databázi existují, systém skutečně našel.

$$Recall = \frac{\text{Počet relevantních nalezených chunků}}{\text{Celkový počet relevantních chunků v databázi}} \quad (1)$$

V rámci RAG flow se Recall uplatňuje tak, že uživatel položí dotaz. Retriever vybere například 3 nejrelevantnější chunky z databáze ze kterých se vygeneruje odpověď.

Když víme, které chunky opravdu obsahují správnou odpověď, můžeme zjistit, zda jsou mezi těmi, které retriever vybral. Pokud ano, je Recall 1,0 tj. 100 %. Pokud retriever nevybral ani jeden relevantní chunk, Recall je 0. [21]

Recall je vhodná metrika pro hodnocení retrieveru, ale nepostihuje kvalitu finální odpovědi, kterou generuje jazykový model. Proto se často kombinuje s dalšími metrikami jako jsou precision, F1 nebo human evaluation.

5.2 Metrika Precision

V kontextu RAG systémů měří Precision metrika jak, relevantní byly odpovědi, které systém poskytl.

$$Precision = \frac{\text{Počet relevantních chunků}}{\text{Celkový počet vrácených chunků}} \quad (2)$$

Po dotazu model vrátí určité množství chunků jako relevantní. Precision říká, kolik z těchto vrácených chunků bylo skutečně relevantních k položenému dotazu. Podává tedy zprávu o tom, jak moc se můžeme spolehnout na to, co model našel. [21]

5.3 Metrika F1-score

Metrika F1-score je kombinací Precision a Recall a slouží k vyvážení jejich případných extrémů.

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

F1-score představuje harmonický průběh těchto dvou metrik. Pokud je jedna z metrik nízká, F1-score taky klesne. F1-score tedy nutí mít dobrý jak Precision tak Recall a ne jenom jeden z těchto ukazatelů. [22]

5.4 Metrika IoU

Intersection over Union neboli IoU metrika se používá pro měření přesnosti generované odpovědi ve srovnání s referenční (správnou) odpovědí. Výpočet spočívá v tom, že se zjistí, kolik tokenů mají společných správná odpověď a odpověď od modelu. Tento průnik se uvede v čitateli. Do jmenovatele se zapíše počet všech unikátních tokenů ve spojení obou odpovědí.

$$IoU = \frac{\text{průnik tokenů}}{\text{sjednocení tokenů}} \quad (4)$$

IoU se vždy pohybuje mezi 0 a 1. Čím vyšší je hodnota IoU, tím lepší je shoda mezi generovanou a referenční odpovědí. [22]

5.5 Metrika Precision with perfect Recall

Tato metrika se používá při testování systémů, které pracují s omezeným, ale známým množstvím dokumentů, což u RAG můžeme využít. Metodu můžeme použít tehdy, když víme, že systém měl možnost vidět všechny relevantní informace.

Perfect Recall znamená, že $\text{Recall} = 1$ (100 %) tj. všechny relevantní dokumenty byly vybrány jako kontext. To se v praxi může stát tehdy, pokud víme, že retriever opravdu našel všechno důležité, anebo testujeme na syntetických datech, kde máme kontrolu nad tím, co je a co není relevantní. [20]

$$\text{Prec. with perf. Recall} = \frac{\text{Počet správně vygenerovaných odpovědí}}{\text{Počet testovacích dotazů, u kterých byly poskytnuty všechny relevantní chunky}} \quad (5)$$

Tato metrika vyhodnocuje, jak dobře model odpoví, když má všechny potřebné informace. To znamená, že fáze retrieval proběhla perfektně a testuje se jen generativní schopnost modelu. Užitečnost metriky spočívá tedy v tom, že se díky ní dokážou oddělit chyby generátoru od chyb retrieveru. [20]

5.6 Metrika Human evaluation

Metrika Human evaluation ověřuje, jak srozumitelně a správně model odpovídá z pohledu člověka. Hodnocení se provádí s reálnými uživateli nebo odborníky. Při hodnocení se berou v úvahu tyto aspekty:

Srozumitelnost – odpověď je čitelná, jazykově správná a stylisticky přirozená

Relevantnost – odpověď skutečně reaguje na položenou otázku

Faktická správnost – odpověď je v souladu se znalostmi v dokumentech nebo realitou

Užitečnost – odpověď pomáhá řešit problém uživatele. [23]

6 ZMĚNY CHOVÁNÍ SYSTÉMU RAG

Změny chování RAG (Retrieval-Augmented Generation) systému lze navodit několika způsoby. Při ladění a experimentování s různým nastavením je pak možné sledovat různou výkonnost systému RAG. [1]

- **Změna system promptu**

Součástí kódu je definován system prompt, který určuje, jak má RAG systém odpovídat na otázky. Úpravou tohoto promptu je možné změnit styl a zaměření odpovědi. Například je možné změnit instrukci na *„Jste nápomocni při zodpovídání otázek o výrobním zařízení. Odpovídáte pouze na základě znalostí, které vám poskytují. Nepoužíváte své vnitřní znalosti a nevymýšlíte si, pokud neznáte odpověď, řekněte že nevím.“* [1]

- **Úprava počtu výsledků**

V kódu je parametr `end results`, který určuje, kolik relevantních výsledků se má použít pro generování odpovědi. Zvýšením tohoto čísla lze RAG systému umožnit zvážit více informací a potenciálně generovat komplexnější odpovědi. Nicméně nastavení příliš vysokého čísla může vést k méně relevantním odpovědím. [1]

- **Volba strategie členění textu**

Chunky obsahující všechny tokeny relevantní pro dotaz se mohou vyskytovat v rámci nebo napříč mnoha dokumenty. Chunky mohou obsahovat relevantní i irelevantní tokeny a relevantní chunky mohou být rozděleny mezi více částí. Z toho důvodu může mít výběr strategie členění významný dopad na výkonnost vyhledávání. Podle článku (Smith et al., 2024) některé strategie členění překonávají jiné až o 9 % v metrice Recall (5.1). Zdůrazňují, že hodnocení různých strategií členění je klíčové pro optimalizaci přesnosti a efektivity vyhledávání. [10]

- **Použití různých jazykových modelů**

Ve skriptu je možné volit mezi různými jazykovými modely (GPT-4.0, Llama, ..) Různé modely mají různé silné a slabé stránky, takže experimentováním s různými modely je možné dosáhnout lepších výsledků pro konkrétní případ použití. [1]

- **Úprava parametrů text splitteru**

Při vytváření databáze se používá `RecursiveCharacterTextSplitter` k rozdělení textu na menší části. Parametry tohoto splitteru jako je `chunk_size` (velikost textu) a

chunk_overlap (překryv mezi kusy textu), ovlivňují, jak jsou data rozdělena. Úpravou tohoto parametru je možno ovlivnit, jak dobře RAG systém dokáže extrahovat relevantní informace. Překrývání zajišťuje, že důležité informace nebudou ztraceny na hranicích chunků. [1]

- **Zahrnutí metadat**

Do odpovědí je možné zahrnout metadata o zdroji informace. To je užitečné zejména tehdy, pokud se používá více zdrojů dat. Uživatel pak může vidět, odkud informace pochází. [1]

7 FRAMEWORKY PRO VÝVOJ RAG APLIKACÍ

PDF dokumenty jsou v podnicích běžným formátem pro uchování důležitých informací, jako jsou technické manuály, směrnice či pracovní instrukce. Tyto dokumenty bývají složité svou strukturou a často obsahují kombinace textu, tabulek, schémat a obrázků. Pro jejich efektivní využití v systémech, které umožňují vyhledávat a generovat odpovědi na základě obsahu dokumentů (RAG), je důležité zvolit vhodný nástroj pro jejich předzpracování.

Tato kapitola představuje přehled frameworků a nástrojů, které jsou zaměřené na zpracování dokumentů pro RAG systémy, a to zejména z pohledu formátu PDF (Tuchiev, 2025).

- **LlamaIndex**

LlamaIndex je open-source framework zaměřený na práci s externími zdroji dat, jako jsou PDF soubory, Word dokumenty, databáze, webové stránky nebo emaily. Jeho silnou stránkou je schopnost rychle vytvořit tzv. knowledge graf nebo index z dokumentů, které následně slouží jako zdroj informací pro dotazy zadané uživatelem. LlamaIndex umožňuje pokročilou správu chunkování a integraci s různými embedding modely i LLM. Podporuje také „retriever chain“. [27]

- **Unstructured**

Unstructured je nástroj i knihovna v Pythonu, který se specializuje na parsování nestrukturovaných dokumentů. Využívá pokročilé heuristiky a modely pro detekci sekcí, nadpisů, tabulek a obrázků. Dokáže převést dokument do dobře strukturovaného JSON výstupu. Podporuje dvě strategie parsování tj. rychlou a vysokého rozlišení. [26]

- **LangChain**

LangChain je framework pro orchestraci a vývoj aplikací využívajících LLM. Dokumenty jsou zde zpracovávány pomocí dokumentových loaderů (PyMuPDF, pdfminer). Tyto loadery umožňují extrakci a rozdělení dokumentů do chunků, které se dále embeddují a ukládají do vektorové databáze. Umožňuje vytváření složitých RAG pipeline s možností přizpůsobení jednotlivých komponent. [26][27]

Pro firemní prostředí, kde se pracuje s technickou dokumentací ve formátu PDF a je kladen důraz na přesnost extrakce, offline režim a bezpečnost, se jako nejlepší kombinace jeví využití Unstructured loaderu v rámci LangChain nebo LlamaIndex. Frameworky jako

LangChain nebo LlamaIndex pak umožní tyto extrahované informace napojit do systému dotazování a generování odpovědí.

8 VEKTOROVÉ DATABÁZE

Tradiční databáze ukládají data do tabulek s předem definovanou strukturou. Ve sloupcích se nacházejí jednotlivé atributy a řádky představují konkrétní datové záznamy. Dotazování na takto uložená data je velmi efektivní, zejména díky využívání indexů, které urychlují vyhledávání ve strukturovaném prostředí. Nicméně při práci s nestrukturovanými daty, jako jsou texty, obrázky nebo zvukové záznamy, tyto systémy narážejí na své limity.

I když vektorové databáze nejsou zcela novým konceptem, jejich význam dramaticky vzrostl s nástupem generativní AI a použitím v systémech typu RAG. [8]

Vektorové databáze ukládají nestrukturovaná data jako jsou text, obrázky, zvuk, video aj. ve formě vektorových embeddingů (vector embeddings). Každý datový bod je transformován do numerického vektoru pomocí technik ML (machine learning). Vektory si můžeme představit jako seznamy čísel s pevnou délkou. Vektorová databáze převádí informace na vektory ve vícerozměrném prostoru a umožňují provádět vysoce efektivní dotazy. Díky tomu je možné realizovat sémantické vyhledávání, což je vyhledávání na základě významové podobnosti, nikoliv pouze podle shody klíčových slov.

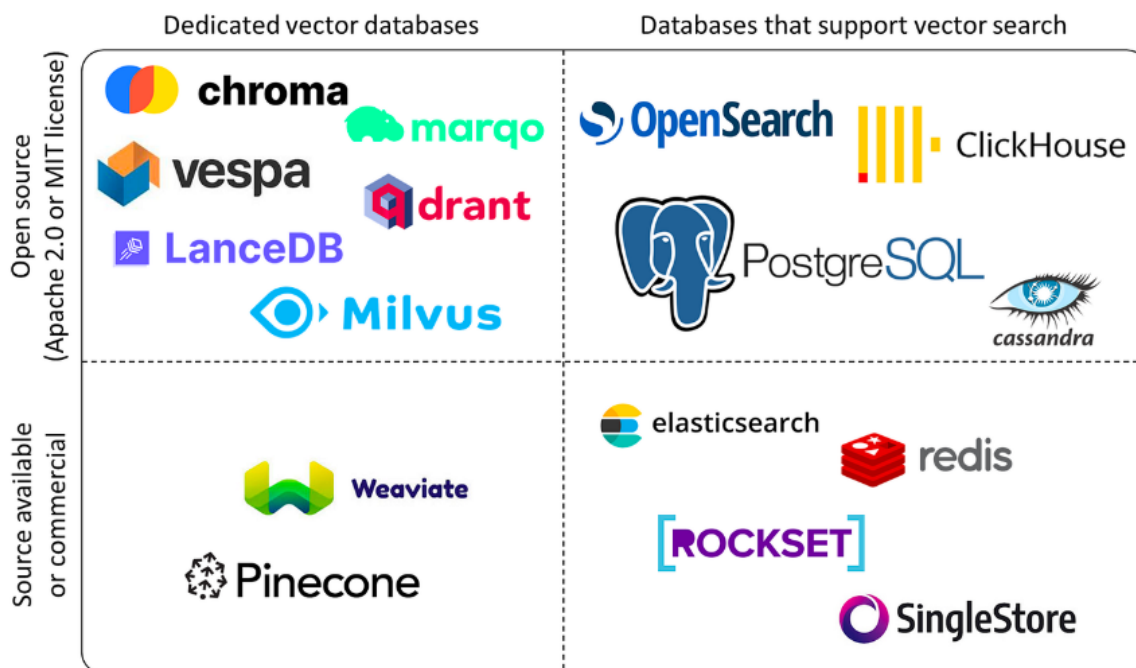
Vektorové databáze tak překonávají takzvanou sémantickou mezeru mezi tím, jak počítače ukládají data a jak jim rozumí lidé. Právě díky této schopnosti nachází tyto technologie klíčové uplatnění v RAG systémech a dalších aplikacích, které využívají jazykové modely.

Při výběru vhodné vektorové databáze je možné zohlednit následující kritéria [2]:

- Open-source vs. uzavřená řešení – Otevřené databáze nabízejí větší flexibilitu a možnost přizpůsobení. Uzavřená řešení mohou být pohodlnější pro komerční použití, ale s menší kontrolou.
- Cloud vs. On-premise nasazení – Důležitým faktorem je, zda je nutné, aby data zůstávala výhradně ve firemní infrastruktuře, nebo je možné je ukládat na cloudové servery.
- Odlehčená vs. Výkonná databáze – Podle požadavků na velikost úložiště, latenci a výkon lze zvolit buď jednoduchou nebo robustní vektorovou databázi.

Obrázek (Obrázek 8) znázorňuje rozdělení vektorových databází podle typu licence (open-source vs. komerční) a podle zaměření (specializované vektorové databáze a klasické databáze s podporou vektorového vyhledávání). V levém horním kvadrantu se nacházejí

open-source nástroje jako Chroma, Milvus či LanceDB. Tyto nástroje nabízejí jednoduché API, integraci s frameworky jako LangChain nebo LlamaIndex a jsou optimalizované pro rychlé dotazy podle podobnosti vektorů. Naproti tomu v pravém polovině obrázku jsou nástroje jako PostgreSQL, Redis nebo OpenSearch, které byly původně navrženy pro klasické relační nebo dokumentové dotazy, ale nově podporují i vektorová vyhledávání. [2]



Obrázek 8 Vektorové databáze

(zdroj: [2])

Hojně využívanou je vektorová databáze ChromaDB, která se používá k ukládání a správě vektorových reprezentací textových dat. Použití ChromaDB v RAG systému slouží k ukládání a správě „chunků“ textu, na které je rozdělen vstupní Adokument. ChromaDB umožňuje sémantické vyhledávání, což znamená, že dokáže najít relevantní informace i na základě významu slov, nikoliv jen na základě klíčových slov. Databáze je integrována s Langchain, což je framework pro vývoj aplikací s velkými jazykovými modely. [1]

9 VEKTOROVÉ INDEXOVÁNÍ

Vektorové indexování je nezbytné pro efektivní vyhledávání ve vektorových databázích, protože přímé porovnání dotazovacího vektoru s každým jedním vektorem v databázi by bylo příliš pomalé. Pokud je totiž v databázi miliony vektorů, které mohou mít navíc stovky či tisíce dimenzí, porovnání dotazovacího vektoru s každým vektorem by rozhodně nebylo rychlé a efektivní. Proto se používá proces zvaný vektorové indexování. [9]

Technika vektorového indexování využívá algoritmy přibližného nejbližšího souseda (Approximate Nearest Neighbor – ANN). Tyto algoritmy nehledají přesnou shodu, ale rychle nacházejí vektory, které s velkou pravděpodobností patří mezi nejbližší. [9]

Ve vektorovém vyhledávání se uplatňuje několik přístupů. Například HNSW (Hierarchical Naavigable Small World) při kterém se vytváří vícevrstvé grafy propojující podobné vektory. Dalším přístupem je IVF (Inverted File Index) při kterém se dělí vektorový prostor do shluků a prohledává pouze ty nejrelevantnější. [9]

Společné pro tyto metody indexování je, že na úkor drobné nepřesnosti se dosáhne dobré rychlosti při vyhledávání. Bez využití vektorového indexování by vyhledávání sémanticky podobného obsahu ve vektorových databázích trvalo příliš dlouho, což by omezovalo jejich praktické využití. [9]

10 BEZPEČNOST VYUŽÍVÁNÍ AI VE FIREMNÍ PRAXI

Firmy se ve svém každodenním fungování setkávají s obavami a riziky, která jsou spojena s využíváním velkých jazykových modelů. Patří mezi ně například nadměrné sdílení citlivých údajů, úniky dat nebo porušení autorských práv. [6]

Jedním z hlavních rizik je tzv. neúmyslné sdílení dat prostřednictvím promptů, které zaměstnanci zadávají do AI nástrojů. Hrozí únik obchodního tajemství, interních kódů, osobních údajů nebo materiálů chráněných autorskými právy. [5]

Podívejme se nyní na příklad populární služby ChatGPT jejíž provozovatel je společnosti OpenAI. Ve svých Podmínkách užívání (Terms of Use) uvádí, že vstupy uživatelů mohou být použity ke zlepšování služeb. Podobný článek nalezneme také v Privacy policy OpenAI. To znamená, že cokoli uživatel do rozhraní zadá, může být potenciálně použito pro další trénink, pokud není tato funkce explicitně deaktivována. [7]

Reakcí na toto riziko bývá například využití placené verze ChatGPT, kde lze nastavit, aby model nepoužíval uživatelská data k dalšímu učení, čímž se zvyšuje úroveň soukromí. [5]

Další hrozbou spojenou s generativními jazykovými modely ve vztahu k úniku firemních informací je prompt hacking. Hacker se může například pokusit ChatGPT donutit, aby prozradil interní data společnosti v rámci žádosti o pomoc s konkrétní zprávou nebo typem dokumentu. [5]

Problém s únikem informací nesouvisí jen s pokročilými konverzačními modely umělé inteligence, ale může se týkat i různých aplikací, jako jsou třeba jazykové překladače založené na umělé inteligenci. [5]

Firmy však vidí na druhou stranu značné výhody z využití AI nástrojů, a proto svým zaměstnancům umožňují využívat např. ChatGPT integrací zabezpečené instance přímo do firemního komunikátoru MS Teams. [5]

Podle průzkumu společnosti IBM patří mezi hlavní faktory, které vedou k rychlému přijetí AI ve firmách:

- pokrok ve vývoji nástrojů, které se rovněž staly dostupnější,
- potřeba snižovat náklady a automatizovat klíčové procesy ve firmách,

- rostoucí množství AI zabudované do standardních hotových podnikových aplikací. [5]

Určitě tedy není správnou cestou AI nástroje preventivně zakazovat nebo ignorovat. [7] „Ve firmách by se měly nastavit jasná pravidla a směrnice pro práci s AI nástroji a dodržování těchto interních pravidel vyžadovat. Dále prověřit, zda a jakým způsobem poskytovatel služby ukládá a využívá vstupní data. A pokud to nástroje umožňují, tak zakázat učení na vložených datech nebo využívat izolované instance nástrojů, které nikam data neposílají. Nejbezpečnější formou pro použití generativní umělé inteligence je provozovat soukromé modely na vlastní infrastruktuře. Nezbytnou součástí interní strategie by mělo být i vzdělávání a školení zaměstnanců.“ [5]

V kontextu této práce je bezpečnost klíčovým faktorem, který vedl k rozhodnutí nepoužívat cloudové nástroje, ale vyvíjet lokální AI aplikaci typu RAG, která běží offline a žádná data neopouštějí zařízení. Tento přístup eliminuje rizika úniku informací a nabízí firmám možnost bezpečně pracovat s vlastní dokumentací, aniž by musely dělat kompromisy v oblasti ochrany dat.

II PRAKTICKÁ ČÁST

11 PODNIKOVÁ ZNALOSTNÍ DATABÁZE

Podnikové znalostní databáze (Knowledge Base – KB) obvykle obsahují strukturované i nestrukturované informace, které slouží k uchování a sdílení firemních znalostí mezi zaměstnanci. Typicky se v nich nacházejí následující data z těchto oblastí:

- Dokumentace a pracovní postupy
- Firemní směrnice a politika
- Technická a produktová dokumentace
- Zákaznická podpora a řešení problémů
- Firemní know-how a odborné znalosti
- Databáze projektů a úkolů
- Obchodní a marketingové materiály
- Personální informace a HR dokumentace
- Finanční a právní dokumenty

Pokud bychom se zaměřili blíže na dokumenty pocházející z výrobního a provozního prostředí, mohli bychom zmínit následující:

- Technologické postupy (výrobní, servisní, montážní návody)
- Standardní operační postupy (SOP)
- Pracovní instrukce a manuály
- Metodiky a best practices
- Bezpečnostní směrnice a pravidla BOZP
- Kvalitativní normy
- A další...

Tyto dokumenty bývají různé struktury a formátu a typicky jsou ukládána v podnikových databázích jako PDF dokumenty.

Jedná se veskrze o dokumenty, které je pro zachování konkurenční výhody nutné udržovat v tajnosti a nevystavovat veřejně na internetu. Proto se v souvislosti s výběrem

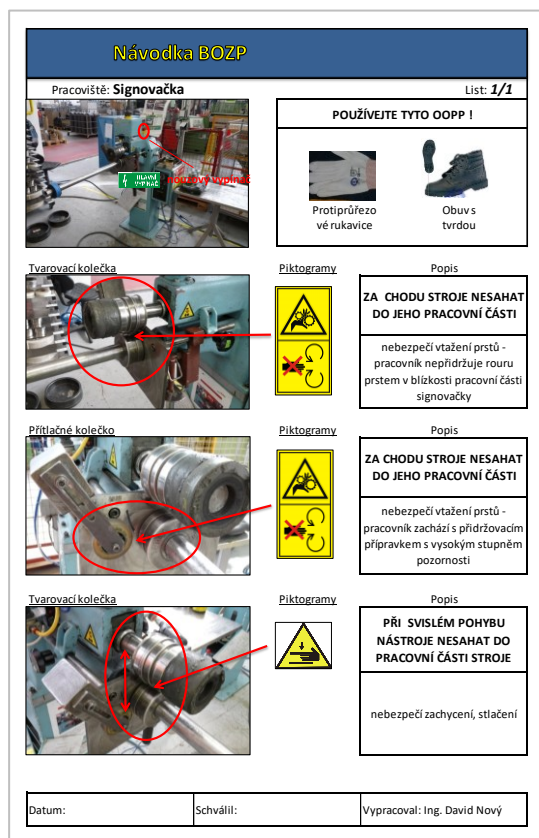
technologie pro získávání dat ze znalostní databáze pomocí nástrojů umělé inteligence musí zvažovat i otázka bezpečnosti.

11.1 Popis dat, která jsou pro zpracování uvažována

Firma, jejíž data jsou pro zpracování diplomové práce uvažována, se zaměřuje na výrobní a technologické procesy, a proto znalostní databáze obsahuje převážně technické dokumenty, pracovní návody a popisy zařízení. Tyto dokumenty slouží jako interní podpora pracovníků ve výrobě, údržbě a kontrole kvality. Dokumenty jsou vytvářeny interními pracovníky většinou v MS Word a následně pro účely zveřejnění jsou převáděny do PDF formátu se zachovanou textovou vrstvou.

V databázi se vyskytují tyto typy dokumentů:

- Technologické postupy – Jedná se o návody s obrázky, poznámkami
- Bezpečnostní směrnice a BOZP – Popisují postupy při mimořádných situacích, pravidlech používání ochranných pomůcek aj. (Obrázek 9)

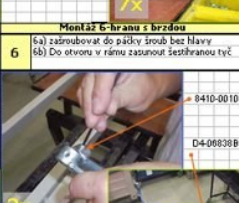
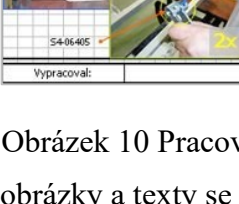
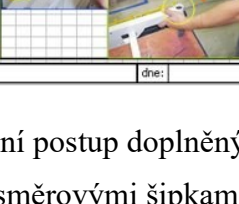
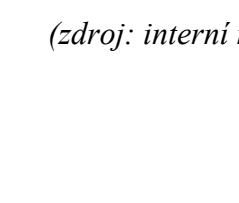
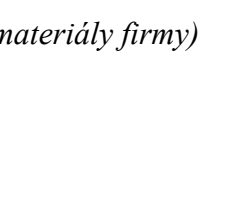


Obrázek 9 Příklad BOZP návodky
(zdroj: interní materiály firmy)

- Standardní operační postupy – Dokument s přesně stanovenými kroky pro rutinní úkony
- Pracovní instrukce – Manuály pro konkrétní zařízení, často se zde vyskytuje kombinace textu a schémat

Uvažované dokumenty obsahují různé typy dat, které je pro další zpracování potřeba extrahovat. Jedná se zejména o:

- Text: Základní text, který může být součástí odstavců, nadpisů, popisků nebo jiných textových prvků.
- Tabulky: Strukturované tabulky s daty
- Schémata: Jednoduché obrázky s texty a směrovými šipkami (Obrázek 10)
- Grafy: Vizualizace dat ve formě grafů.
- Obrázky: Obrázky, které mohou být relevantní pro obsah dokumentu.

Pracovní postup (Standard)			
Výrobek:	ELEGANZA	Počet obsluh:	1
Manipulace s podvozkem		Montáž koleček na podvozek	
1	1a) Přinout podvozek na pracovnístě 1b) Zabrzdí vozík	2	2a) Do otvorů v podvozku vložit postupně 2b) jedno kolečko ze zemi (se žlutou krycíkou)
			
Varianta podvozku dle specifikace		Varianta koleček dle specifikace	
			
Montáž koleček na podvozek		Montáž koleček na podvozek	
3	3a) Zemi kolečko zasrubovat se zemi 3b) na šroub nasadit podložku a zemi kabel s podložkami dle obrázku	4	4a) Opatrně kolečka posunout na každé straně (2x) šroubem s prstovou podložkou 4b) vystrčit prstovou podložku z zemi kabelu
			
1x		7x	
			
Montáž 6-kranu s brzdou		Montáž 6-kranu s brzdou	
5	5a) Po montáži koleček otočit podvozek 5b) Připravit ovládací páčky na táhlo (2x) 5c) Do závitu v páčce nanést lepidlo	6	6a) zasrubovat do páčky šroub bez hlavy 6b) Do otvoru v rámu zasunout šestizubou tyč
			
2x		2x	
			
Vypracoval:		dne:	

Obrázek 10 Pracovní postup doplněný
obrázky a texty se směrovými šipkami
(zdroj: interní materiály firmy)

- Další specifické detaily: Například názvy firem, matematické či chemické vzorce, telefonní čísla, jména osob, adresy, data událostí, finanční částky a další identifikátory, jež jsou specifické pro daný typ dokumentu.

PDF dokumenty mohou mít složité a nekonzistentní formáty, což ztěžuje extrakci dat. Tradiční metody parsování často selhávají u dokumentů s nekonzistentním rozložením, různými typy obsahu (text, tabulky, grafy, obrázky) a složitou strukturou. Proto je obecně doporučováno používat pokročilé techniky parsování.

Kromě běžného textu obsahují dokumenty také vložené obrázky (například fotografie zařízení nebo schémata), které je možné analyzovat pomocí OCR nástrojů nebo i multimodálních modelů. Tato schopnost je důležitá zejména tehdy, když jsou důležité pokyny uvedeny pouze na fotce nebo jako naskenovaný dokument.

11.2 Umístění dat

Uvažovaná znalostní báze je umístěna na firemním síťovém úložišti. K jednotlivým dokumentům se přechází prostřednictvím hypertextových odkazů z jednoho Excel sešitu. Hierarchická struktura dokumentů je řešena následovně:

- První úroveň – členění je realizováno v rámci jednotlivých listů Excel sešitu. Každá kapitola první úrovně má svůj samostatný list.
- Druhá a další úroveň – členění je řešeno v rámci daného listu, kde dochází k rozpadu na další úrovně.

Vzhledem k tomu, že uvažované dokumenty obsahují firemní know-how a neveřejné informace, není možné je použít přímo v diplomové práci. Kromě toho jsou velmi specifické – plné technických detailů, zkratk a interní terminologie. Bez hlubší znalosti firemního prostředí by bylo pro autorku i ostatní čtenáře obtížné s nimi efektivně pracovat a hodnotit výsledky testů.

Proto bylo rozhodnuto vytvořit vlastní syntetické dokumenty, které se strukturou a stylem co nejvíce podobají těm reálným. Tato cesta umožnila bezpečně a přehledně testovat funkce vyvíjené aplikace. Také díky tomu je možné jasně ukázat, jak různá nastavení a přístupy ovlivňují chování systému.

12 POUŽITÝ HARDWARE

Vzhledem k charakteru uvažovaných provozních dat, která jsou rozsáhlá, složitě strukturovaná a obsahují nejen textové, ale i vizuální prvky, bylo zapotřebí vytvořit výpočetně dostatečně výkonné prostředí pro testování lokální aplikace. Původní firemní notebook neposkytoval dostatečný výkon ani podporu pro běh modelů s využitím GPU, a proto byla za tímto účelem sestavena vlastní výkonnější počítačová sestava. Následující kapitola se věnuje jejímu detailnímu popisu.

Počítačová sestava je podrobně popsána v následující tabulce (Tabulka 3). Při výběru komponentů byl důraz kladen především na dostatek paměti, solidní procesorový výkon a samostatnou grafickou kartu s podporou CUDA, což je důležité při práci s některými LLM.

Tabulka 3 Použitý hardware

Komponenta	Specifikace
Procesor:	Intel Core i5
Operační paměť (RAM)	32 GB DDR4 (Kingston FURY 3200 MHz CL16)
Grafická karta (GPU)	NVIDIA GeForce RTX 3060 GAMING OC 12G
Úložiště (SSD)	1 TB NVMe SSD Kingston NV3
Základní deska	ASUS TUF GAMING B550-PLUS (s podporou PCIe 4.0, USB 3.2), formát ATX
Zdroj	Modulární zdroj napájení 1000W
Skříň	FSP Fortron CMT160
Operační systém	Windows 11

(zdroj: vlastní)

• **Základní deska**

Základní deska Asus TUF B550-PLUS patří do řady TUF (The Ultimate Force), která je zaměřena na odolnost, stabilitu a dlouhou životnost. Asus TUF B550-PLUS podporuje procesory AMD Ryzen 3. a 5. generace díky patici AM4 a čipsetu B550. Nabízí dostatečný počet portů USB, podporu PCIe 4.0 (pro grafické karty a SSD), kvalitní integrovanou zvukovou kartu i síťovou kartu s podporou gigabitového

připojení. Má také moderní BIOS s přehledným uživatelským rozhraním a možností přetaktování. [19]

- **Procesor**

Procesor – AMD Ryzen 5 5600 je šestijádrový dvanáctivláknový procesor postavený na architektuře Zen 3. Nabízí velmi dobrý poměr výkonu a ceny. Základní frekvence je 3,5 GHz s možností automatického boostu až na 4,4 GHz. Díky vysokému výkonu a efektivní správě spotřeby je tento procesor vhodný jak pro hry, tak i pro práci s náročnějšími aplikacemi – jako je střih videa, rendering, programování nebo experimentování s AI modely. K chlazení procesoru je využitý základní dodaný chladič. [19]

- **Operační paměť**

V sestavě byla použita operační paměť Kingston Fury 32 GB DDR4 (2×16 GB). Tato kapacita poskytuje dostatek prostoru pro multitasking, práci s náročnými aplikacemi i provoz výpočetně náročných skriptů. Vysoké množství operační paměti je také klíčové při testování a ladění velkých jazykových modelů, které často vyžadují vyšší objemy dat a paralelní výpočty v paměti. [19]

- **Úložiště SSD**

SSD disk Kingston NV2 1TB využívá rozhraní NVMe a M.2 slot, což zajišťuje velmi rychlé přenosy dat v porovnání s klasickými SATA disky. S kapacitou 1 TB poskytuje dostatek prostoru pro operační systém, aplikace, knihovny, datové sady i výstupy z různých experimentů. Díky rychlému načítání systémů a souborů se zvyšuje celková svižnost počítače. [19]

- **Grafická karta**

Grafická karta NVIDIA GeForce RTX 3060 s 12 GB paměti GDDR6 je výkonná grafická karta střední třídy, která si bez problémů poradí s hraním her ve Full HD a ve většině případů i ve 1440p rozlišení. Kromě toho podporuje ray tracing i DLSS, což zajišťuje vyšší vizuální kvalitu a lepší výkon v podporovaných hrách. Významná je také podpora výpočetních operací prostřednictvím CUDA jader, což je využitelné právě při vývoji a testování AI modelů, včetně LLM. [19]

- **Zdroj**

Modulární zdroj s výkonem 1000 W zajišťuje dostatečný výkon s rezervou pro všechny komponenty i případný budoucí upgrade. Díky modulární konstrukci je možné zapojit pouze potřebné kabely, což usnadňuje kabelový management a zlepšuje proudění vzduchu ve skříni. Tento výkon je dostatečný i při intenzivní zátěži, jako je např. trénování nebo inferování modelů v rámci GPU výpočtů. [19]

- **Skříň**

Skříň formátu ATX poskytuje dostatek prostoru pro všechny komponenty a jejich chlazení. I když jde o základní model, je důležité, aby měla dobré proudění vzduchu, možnost instalace ventilátorů a dostatek místa pro případné rozšíření. V kombinaci s modulárním zdrojem a moderní deskou bylo dosaženo přehledného a funkčního uspořádání vnitřku počítače. [19]

Tato počítačová sestava umožňuje běh lokálních jazykových modelů až do velikosti přibližně 7-8 miliard parametrů (například gemma:7B, llama3, mistral), zatímco větší modely (např. llama3.3:70B nebo llama3.2:11B) již z důvodu limitované VRAM nebo výpočetní náročnosti běží pouze na CPU což vede k výrazně nižší rychlosti a vyšší latenci.

Modely jsou spouštěny lokálně pomocí nástroje Ollama, který využívá GPU akceleraci a optimalizované formáty modelů jako GGUF. Původní zařízení, tj. firemní notebook, který byl k dispozici pro zpracování diplomové práce, neposkytoval dostatečný výkon pro žádné z testovaných LLM, a proto byla sestavena nová výkonnější počítačová sestava. Ta byla navržena tak, aby poskytovala dobrý kompromis mezi výkonem a dostupností. Ačkoli výkonnější grafické karty s vyšší VRAM by umožnily spouštět i větší modely, byly z finančních důvodů mimo možnosti této práce. Přesto sestava umožnila plnohodnotný vývoj a testování pokročilé RAG aplikace v offline režimu bez nutnosti využití cloudových služeb.

Zároveň je však třeba zohlednit, že tato vlastní sestava není firemním zařízením, a tudíž není možné z důvodu bezpečnosti na tomto zařízení zpracovávat skutečná provozní data. Tato omezení znovu potvrzují vhodnost využití syntetických dokumentů, které umožňují bezpečné testování funkcí aplikace bez rizika úniku citlivých informací.

13 SYNTETICKÁ DATA PRO TESTOVÁNÍ

Vzhledem k omezením v oblasti bezpečnosti a dostupnosti firemních dat byla pro vývoj a testování aplikace vytvořena vlastní syntetická testovací sada dokumentů. Data byla navržena tak, aby co nejvěrněji simulovala reálné dokumenty, nad kterými by v podnicích dotazování probíhalo. Typicky se jedná o technologické návody, pracovní instrukce, směrnice a jiné. Díky tomu půjde otestovat důležité funkcionality, jako je rozdělení textu na chunky, extrakce textu pomocí OCR, sémantické vyhledávání a generování odpovědi pomocí jazykového modelu.

Dalším důležitým přínosem použití syntetických dat je to, že jejich obsah je dobře srozumitelný i laikovi, který nemá hlubší znalosti konkrétního oboru. Tím je usnadněno ověřování správnosti odpovědí, což je zásadní při použití metrik kvality odpovědí (kapitola 5), které budou v rámci diplomové práce vyhodnocovány. Díky těmto vlastnostem syntetická data umožňují testovat celý proces fungování systému RAG bez nutnosti použití reálných citlivých dat.

Každý syntetický dokument byl vytvořen ve třech různých variantách:

- Původní textový dokument vytvořený v MS Word ve formátu .docx: slouží jako výchozí editovatelný soubor s plně strukturovaným obsahem
- PDF dokument: exportovaný z .docx, představuje typický firemní výstup. Umožňuje testovat schopnosti parsování (textová vrstva, zachování struktury, nadpisy, tabulky).
- Markdown soubor (.md): jedná se o verzi dokumentu, která je vytvořena ve značkovacím jazyce Markdown. Tento formát umožňuje testování podpory strukturovaných textů s hierarchií nadpisů (#, ##, ###) a specifických oddělovačů odstavců.

V následujících kapitolách se budu detailně věnovat konkrétním dokumentům, které byly zvoleny pro ladění a testování. Každý dokument byl vybrán s ohledem na jiný typ obsahu nebo struktury, aby bylo možné ověřit funkcionalitu aplikace a zároveň simulovat realistické scénáře dotazování ve firemním prostředí.

13.1 Syntetická data – zpráva Digitalizace dávek nemocenského pojištění

Název dokumentu:	Zpráva Digitalizace dávek nemocenského pojištění
Ukázka:	Příloha P II
Formát:	PDF, DOCX, Markdown
Charakteristika:	Dvoustránkový textově obsáhlý dokument, který popisuje procesní změny v oblasti nemocenského pojištění. Obsahuje logické členění, ale méně explicitní strukturu než směrnice Standardizace výpočetní techniky (kapitola 13.2).
Účel testu:	Ověření kvality odpovědí při použití různých embedding modelů a strategií chunkování. Slouží jako vstupní dokument pro testování metrik Precision, Recall a F1 vůči referenční odpovědi.

13.2 Syntetická data – směrnice Standardizace výpočetní techniky

Název dokumentu:	Směrnice Standardizace výpočetní techniky
Ukázka:	Příloha P III
Formát:	PDF, DOCX, Markdown
Charakteristika:	Dokument představuje dvoustránkovou interní směrnici popisující pravidla pro přidělování a obnovu ICT vybavení. Obsahuje víceúrovňovou hierarchii nadpisů, číslovaný seznam a tabulku s plánem obnovy.
Účel testu:	Ověření chunkování podle hierarchických nadpisů, práce s dlouhými texty, vyhledávání v tabulkových datech, posouzení přesnosti odpovědí.

13.3 Syntetická data – Pracovní postup provedení tahové zkoušky

Název dokumentu:	Pracovní postup Tahová zkouška
Ukázka:	Příloha P IV
Formát:	PDF, DOCX, Markdown
Charakteristika:	Dokument reprezentuje typický technický návod na obsluhu zařízení. Text je stručný a členěný do očíslovaných kroků. Obsahuje specifické

instrukce k obsluze tahového zařízení. V textu se nachází odkazy na konkrétní obrázky, které popisují ovládací prvky zařízení a polohu vzorku.

Účel testu: Testuje schopnost modelu interpretovat odkazy typu „Obr.1“ a vztáhnout je ke kontextu. Dokument je vhodný pro testování přesnosti generovaných odpovědí na dotazy typu „Jaká je první operace?“ nebo „Co se provádí po upnutí vodiče?“. Ověřuje, zda jazykový model chápe technické instrukce a zachovává jejich smysl při parafrázování odpovědí.

13.4 Syntetická data – Vizuální standard Dresscode společnosti

Název dokumentu: Vizuální standard Dresscode společnosti

Ukázka: Příloha P V

Formát: PDF, DOCX, Markdown

Charakteristika: Graficky zpracovaný dokument s krátkými textovými popisky a množstvím obrázků, ikon a barevně zvýrazněných pravidel. Text často odkazuje na vizuální prvky.

Účel testu: Ověření chování systému RAG při zpracování dokumentu, kde převládají vizuální informace. Zkoumá se, jak jazykový model reaguje v situaci, kdy má k dispozici pouze textová metadata, a nikoliv význam zobrazených obrázků.

14 APLIKAČNÍ PROSTŘEDÍ PRO TESTOVÁNÍ RAG

Aby bylo možné testovat různé varianty nastavení RAG systému, bylo v rámci diplomové práce přistoupeno k vývoji vlastní aplikace. Cílem bylo vytvořit nástroj, který umožní lokální práci s podnikovou dokumentací a nabídne interaktivní prostředí pro ověřování různých kombinací nastavení. Vývoj aplikace probíhal iterativně a opíral se o studium dostupné dokumentace a využití nástroje ChatGPT 4o [41]. Tento přístup umožnil rychlé ověřování různých variant a přispěl k celkové efektivitě vývoje.

14.1 Základní přehled možností a ovládání aplikace

Vyvinutá aplikace pro testování architektury RAG umožňuje lokální práci s dokumenty ve formátu PDF, DOCX a Markdown. Aplikace je postavena nad frameworkem Gradio a využívá kombinaci embedding modelů, vektorového vyhledávání a generativních jazykových modelů (LLM). Umožňuje interaktivní dotazování uživatele a zároveň nabízí nástroje pro testování kvality odpovědí při různých nastavení.

Funkčnost aplikace je rozdělena do několika modulů:

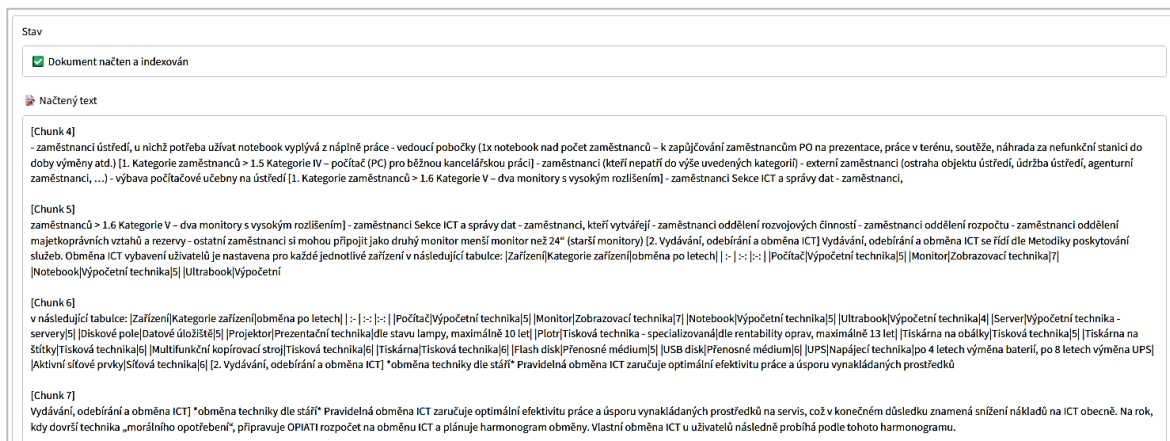
- Předzpracování dat – Dokumenty jsou načteny a rozděleny na menší části (chunks) dle strategií (Obrázek 11):
 - podle odstavců
 - pomocí fixní velikosti s volitelným překryvem.

The screenshot displays the 'Testovací prostředí pro RAG' application. At the top, there's a header with a green icon and the title. Below it is a file selection area with a button 'Vyber dokument (.pdf, .md, .docx)' and a file 'Směrnice výpočetní techniky.md' (5.0 KB) is shown. The main settings section includes: 'Typ rozdělení' set to 'Fixní velikost', 'Chunk (slova)' set to '100', 'Překrytí (slova)' set to '20', and 'Embedding model' set to 'TF-IDF'. At the bottom, there is a 'Načíst dokument' button.

Obrázek 11 Aplikace RAG – vložení zdrojového dokumentu, volba strategie chunkování a embedding modelu, uložení indexů

(zdroj: vlastní)

- Volba embedding modelu – Uživatelské rozhraní umožňuje zvolit mezi třemi embedding modely (podrobně jsou popsány v kapitole 14.2):
 - Klasický TF-IDF
 - Sémantický model all-MiniLM-L6-v2
 - Pokročilý sémantický model intfloat/e5-base-v2
- Uložení embeddingů a vyhledávání – Vytvořené embeddingy jsou uloženy do FAISS indexu a dotazy uživatele jsou porovnávány pomocí sémantické podobnosti (cosine similarity).
- Výběr jazykového modelu (LLM) – Uživatel si může vybrat, jazykový model, který pro generování odpovědi použije:
 - gemma:7b
 - llama3
 - llama3.1:8b
 - mistral
- Vizualizace a export výsledků – Výsledky testování se zobrazují v tabulce a jsou doplněny o metriky, jako je délka odpovědi, výskyt fráze „nevím“ a podobnost s referenční odpovědí. Data lze exportovat pro další analýzu.
- Součástí rozhraní je i pole pro zobrazení úplného textu načteného dokumentu (Obrázek 12). To umožňuje ověřit, které informace se skutečně dostávají do dalšího zpracování a jak jsou interpretovány embedding modelem a LLM.



Obrázek 12 Aplikace RAG – Zobrazení načteného dokumentu rozděleného na chunky

(zdroj: vlastní)

14.2 Výběr a charakteristika testovaných modelů

Pro převod textových chunků do vektorové podoby byly v rámci testovací aplikace implementovány tři různé embedding modely, jež představují různé přístupy k reprezentaci významu textu:

14.2.1 Embedding model TF-IDF (Term Frequency-Inverse Document Frequency)

Model vytváří vektory na základě četnosti slov v dokumentu (term frequency) a vzácnosti v celém souboru dokumentů (inverse document frequency). Zachycuje, jak důležité je konkrétní slovo v daném textu. Je to rychlý a nenáročný model. Funguje dobře při shodě klíčových slov, ale nerozpozná význam slov.

Například pokud je v provozní směrnici uvedeno: „Operátor zkontroluje ventilátor chlazení před spuštěním výrobní linky“. Pokud uživatel položí dotaz: „Co má zaměstnanec udělat s větrákem před spuštěním výroby?“

TF-IDF nevrátí tento odstavec jako relevantní, protože slovo „větrák“ se v dokumentu vůbec nevyskytuje, přestože jde o běžné synonymum pro „ventilátor“.

14.2.2 Embedding model all-MiniLM-L6-v2

Jedná se o sémantický embedding model z rodiny SentenceTransformers, který vyvinuli Hugging Face a Microsoft. Využívá bi-encoder architekturu díky které se vstupní texty vektorizují nezávisle a dokáže rychle porovnávat jejich podobnost (cosine similarity). Je to lehký model vhodný pro lokální běh. I pro větší objemy dat zvládá rychlé výpočty. Dobře zachycuje význam slov ve větách. Kontext zachycuje na úrovni vět, a dokáže tak propojit pojmy, které nejsou totožné, ale jsou významově blízké.

Například v dokumentaci může být napsáno: „Obsluha zařízení musí zkontrolovat těsnost potrubí před zahájením provozu.“ Uživatel zadá dotaz: „Co se má udělat s trubkami před spuštěním výroby?“

Model MiniLM by měl správně odvodit, že „trubky“ a „potrubí“ jsou významově velmi blízké výrazy a vrátí příslušný odstavec.

14.2.3 Embedding model intfloat/e5-base-v2

Jedná se o pokročilý sémantický model trénovaný pro RAG úlohy. Model byl vyvinutý pro lepší výkon v úlohách sémantického vyhledávání. Je speciálně trénovaný na úlohy typu

otázka – odpověď a přidává důraz na relevantnost obsahu vůči dotazu, nejen na sémantickou podobnost.

Například: v dokumentaci může být uvedeno: „Po výměně filtru je nutné provést kalibraci senzoru a zápis do provozního deníku.“ Pracovník se zeptá: „Co dělat po výměně filtrační jednotky?“

Model by měl chápat, že „filtrační jednotka“ je významově blízký pojem k „filtru“ a že uživatel směřuje na následné kroky. Výsledkem je, že e5-base-v2 model upřednostní relevantní akce (kalibrace, zápis), zatímco MiniLM by mohl vrátit jakýkoli odstavec obsahující „filtr“, i kdyby nepopisoval další kroky.

14.2.4 Jazykové modely

Pro aplikaci byly voleny open-source jazykové modely o velikosti do 7-8 miliard parametrů, aby bylo možné zajistit jejich běh na GPU dostupném v použité počítačové sestavě. Tím se minimalizuje riziko, že by při nedostatku paměti došlo k automatickému přepnutí výpočtu na CPU, což by výrazně zpomalilo odezvu. Podporované modely jsou uvedeny v tabulce (Tabulka 4).

Tabulka 4 Jazykové modely použité v aplikaci

LLM	Vyvíjející společnost	Kontextové okno	Vydání modelu	Počet parametrů
gemma:7b	Google	8 192 tokenů	2/2024	7 miliard
llama3	Meta Inc.	8 192 tokenů	3/2023	8 miliard
llama3.1:8b	Meta Inc.	128 k tokenů	12/2023	8 miliard
mistral	Mistral AI	8 192 tokenů	9/2023	8 miliard

(zdroj: [36], [37], [38], [39])

V aplikaci byly implementovány čtyři jazykové modely od různých vývojářů a všechny jsou podporovány nástrojem Ollama [40], který zajišťuje jejich spuštění v lokálním prostředí.

14.3 Popis RAG aplikace

Aplikace slouží jako nástroj pro dotazování nad obsahem interních dokumentů pomocí metody Retrieval-Augmented Generation (RAG). Uživatel do systému nahraje dokument ve formátu PDF (.pdf), Markdown (.md) nebo Word (.docx), zadá dotaz a spustí

několikastupňové zpracování dokumentu, které vede ke generování odpovědi jazykového modelu.

Po načtení dokumentu a rozdělení textu do bloků (chunků) jsou jednotlivé chunky převedeny na vektorovou reprezentaci (embedding) pomocí uživatelem zvoleného modelu (TF-IDF, all-MiniLM-L6-v2, e5-base-v2). Vektory jsou uloženy v indexu knihovny FAISS, která umožňuje efektivní vyhledávání podobných bloků na základě uživatelského dotazu.

Dotaz zadaný uživatelem (Obrázek 13) je rovněž převeden do embeddingu a následně se pomocí metody nejbližších sousedů (k-NN) vyhledají nejrelevantnější části dokumentu.

The screenshot shows a web interface for a Retrieval-Augmented Generation (RAG) application. It contains three main sections:

- Dotaz (Question):** A text input field with the question: "Jakým způsobem se určuje čas pro výměnu zařízení a kdo připravuje rozpočet na jeho obměnu?".
- Referenční odpověď (Reference Answer):** A text area displaying a reference answer: "Čas výměny zařízení se určuje podle stáří zařízení a míry jeho opotřebení. Rozpočet na obměnu ICT připravuje OPIATI v roce, kdy zařízení dosáhne morálního opotřebení."
- LLM model (LLM Model):** A dropdown menu currently showing "llama3.1:8b".

 At the bottom, there is a button labeled "Odeslat dotaz" (Send question) with a magnifying glass icon.

Obrázek 13 Aplikace RAG – Vložení dotazu, referenční odpovědi a volba LLM
(zdroj: vlastní)

Výsledné relevantní chunky jsou zobrazeny uživateli spolu s odpovědí generovanou jazykovým modelem (Obrázek 14).

The screenshot shows the output of the RAG application. It contains four main sections:

- Výstup modelu (Model Output):** A text area showing the generated answer: "Podle textu je určený čas pro výměnu zařízení stanoven podle toho, kdy dané zařízení dosáhne „morálního opotřebení“. Tento rok je součástí rozpočtu připravovaný organizací OPIATI."
- Tokeny (Tokens):** A text area showing the token count: "866 tokenů".
- Trvání (Duration):** A text area showing the execution time: "6.93 s".
- Použitý kontext (Used Context):** A text area displaying several context chunks (Chunk 7, Chunk 5, Chunk 11) related to ICT equipment replacement and budgeting.

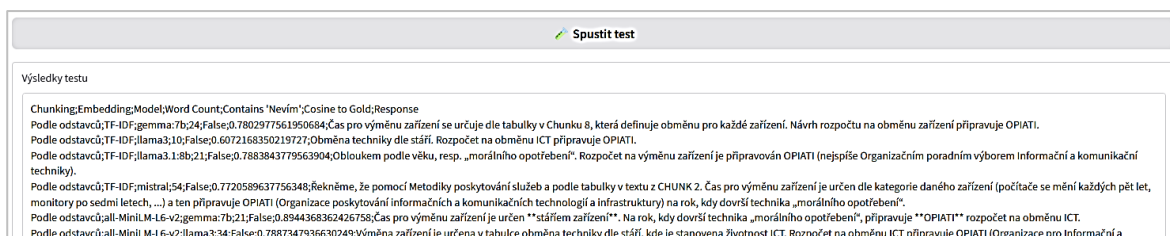
Obrázek 14 Aplikace RAG – Výstup modelu, počet tokenů, trvání výpočtu a použitý kontext
(zdroj: vlastní)

Komunikace s jazykovým modelem probíhá pomocí lokální instance nástroje Ollama, která umožňuje využívat open-source modely bez nutnosti odesílání dat mimo zařízení.

Kromě samotné odpovědi jsou uživateli zobrazeny i doplňkové informace:

- načtený vstupní text rozdělený na chunky,
- počet použitých tokenů v dotazu,
- doba trvání generování odpovědi,
- přehled chunků použitých při odpovědi (včetně jejich čísel a obsahu).

Součástí aplikace je i testovací režim (Obrázek 15), který umožňuje porovnat výstupy různých kombinací embedding a jazykových modelů na základě referenční odpovědi.



Obrázek 15 Aplikace RAG – Testovací režim pro porovnání různých kombinací systému
(zdroj: vlastní)

14.4 Architektura RAG aplikace

Aplikace je navržena jako modulární desktopová webová aplikace běžící na lokálním serveru s uživatelským rozhraním vytvořeným pomocí knihovny Gradio. Architektura je rozdělena do několika klíčových funkčních bloků, které zajišťují oddělení vstupního zpracování dokumentu, vyhledání relevantních informací a komunikace s jazykovým modelem. Celý programový kód je uveden v příloze (Příloha P VIII).

Hlavní moduly aplikace:

14.4.1 Modul načítání a předzpracování dokumentu

Tento modul zajišťuje rozpoznání typu dokumentu podle přípony a použití odpovídající funkce pro extrakci textu:

- `extract_text_from_pdf()` pro formát PDF využívá knihovnu fitz (PyMuPDF),
- `extract_text_from_md()` pro Markdown soubory, přičemž analyzuje hierarchii nadpisů (#, ##, ###) pro správné přiřazení kontextu jednotlivým odstavcům a pro

každý chunk vytváří popis, který kombinuje nadřazené nadpisy. Rovněž se zabývá rozpoznáváním tabulek, aby byly zpracovány jako celek v jednom chunku.

- `extract_text_from_docx()` pro .docx dokumenty pomocí knihovny python-docx.

14.4.2 Modul chunkování textu

Na základě uživatelské volby jsou textové bloky rozděleny buď podle přirozených odstavců funkcí `split_by_paragraph()`, nebo na fixní počet slov s překryvem funkcí `split_fixed_size()`. Cílem je vytvořit smysluplné jednotky textu, které lze dobře vyhledávat.

14.4.3 Modul tvorby embeddingů a indexace

Modul zajišťuje převod chunků do vektorové podoby. TF-IDF model využívá `TfidfVectorizer` z knihovny `sklearn`. Modely all-MiniLM-L6-v2 a e5-base-v2 využívají knihovnu `sentence-transformers`.

Vektory jsou uloženy do indexu `faiss.IndexFlatL2`, který umožňuje rychlé vyhledávání podobnosti vůči chunkům v dokumentu.

Funkce `build_index()` vytváří vektorový index dokumentu podle zvoleného embedding modelu. Tento index se následně využívá pro vyhledávání nejrelevantnějších textových chunků vzhledem k dotazu uživatele. Funkce na vstupu přijímá seznam chunků, které vznikly rozdělením vstupního dokumentu a název uživatelem zvoleného embedding modelu. Funkce potom vrátí samotný FAISS index, všechny embeddingy chunků a použitý vektorizační nástroj.

14.4.4 Modul dotazování a komunikace s jazykovým modelem

Po zadání dotazu se pomocí funkce `find_similar_chunks()` vyhledají nejrelevantnější chunky v dokumentu. Jedná se o klíčovou komponentu RAG systému. Podle zvoleného typu embedding modelu se u TF-IDF dotaz převede na vektor pomocí klasického vektorizéru, nebo u transformerů se dotaz zakóduje do embeddingu pomocí příslušného modelu.

Funkce `search()` porovná dotaz s vektory všech chunků v indexu a vrátí indexy top_k nejpodobnějších chunků. Také je vypočítána euklidovská vzdálenost od dotazu, pro kterou platí čím menší hodnota, tím jsou si vektory blíže v prostoru, tedy jsou si obsahově podobnější. Funkce vrátí seznam top_k textových chunků a jejich odpovídající vzdálenosti.

Následně je dotaz i kontext předán do funkce `query_llm()`, která komunikuje s jazykovým modelem spuštěným přes nástroj Ollama (Obrázek 16). Funkce očísluje

chunky, aby uživateli umožnil lépe sledovat, odkud informace pochází. Chunky se spojí do jednoho kontextového textu, který se následně v promptu předává modelu. Poté je sestaven prompt, který modelu říká, aby čerpal informace jen z kontextu a pokud odpověď z textu nevyplývá, tak má odpovědět „Nevím“. Modelu je také předána systémová instrukce, aby odpovídal stručně a česky. Funkce *query_llm()* tedy vrátí odpověď, počet tokenů, čas trvání výpočtu a vstupní text.

```
prompt = (  
    f"Odpověz na otázku výhradně na základě poskytnutého textu. "  
    f"Pokud informace chybí, napiš 'Nevím'.\n\n"  
    f"---\n{context_text}\n---\n\n"  
    f"Dotaz: {question}"  
)  
  
start = time.time()  
result = ollama.chat(  
    model=model,  
    messages=[  
        {"role": "system", "content": "Odpovídej stručně a česky."},  
        {"role": "user", "content": prompt}  
    ]  
)
```

Obrázek 16 Prompt modelu a systémová instrukce

(zdroj: vlastní)

14.4.5 Modul pro evaluaci modelů

Funkce *evaluate_for_strategy()* umožňuje porovnat výstupy různých kombinací embedding modelů a jazykových modelů. Výsledky jsou ukládány do tabulky a rovněž je vytvořen graf, který porovnává počty slov v odpovědích za jednotlivé LLM modely, embeddingy a strategie chunkování.

14.4.6 Řídící funkce

V této části jsou popsány funkce, které propojují vícero modulů aplikace a zajišťují jejich koordinované spouštění v odpovědi na akce uživatele jako je nahrání dokumentu nebo zadání dotazu.

Funkce *load_and_index()* je zodpovědná za předzpracování vstupního dokumentu. Funkce načítá soubor, aplikuje zvolenou chunkovací strategii a vytvoří embeddingy, které uloží do vektorového indexu FAISS. Tyto embeddingy slouží k následnému vyhledávání relevantních částí textu.

Funkce *single_query()* zajišťuje hlavní tok dotazu. Od zadání otázky uživatelem přes vyhledání relevantních chunků až po odeslání dotazu do LLM spolu s kontextem.

Výsledkem je odpověď generovaná na základě obsahu dokumentu, doplněná o vizualizaci použitých částí textu.

14.4.7 Uživatelské rozhraní

Uživatelské rozhraní je vytvořeno pomocí frameworku Gradio, který umožňuje snadné sestavení interaktivních komponent pro zadávání vstupů, zobrazování výstupů a spouštění funkcí. Komponenty jsou uspořádány do bloků v rámci *gr.Blocks()* a tvoří prostředí, které uživateli umožňuje:

Hlavní prvky UI

- Nahrát dokument ve formátu .pdf, .md, .docx,
- Zvolit strategii chunkování, velikost chunku a překrytí,
- Vybrat embedding model,
- Vybrat jazykový model,
- Položit dotaz nad dokumentem a zobrazit výsledek odpovědi,
- Zobrazit výsledky testu v podobě textového výpisu a grafu.

Tlačítka `load_btn`, `ask_btn` a `test_btn` jsou pomocí *.click()* propojeny s konkrétní funkcí v aplikaci (*load_and_index()*, *single_query()*, *evaluate_for_strategy()*). Tato propojení zajišťují reakci na uživatelské akce a přenos dat mezi UI a logickou aplikací.

15 TESTOVACÍ SCÉNÁŘ A METODIKA HODNOCENÍ

Cílem experimentální části je vyhodnotit kvalitu odpovědí generovaných aplikací v různých konfiguracích architektury RAG. Testování se zaměřuje na klíčové komponenty systému:

- formát vstupního dokumentu,
- strategii chunkování vstupních dokumentů,
- použitý embedding model,
- zvolený jazykový model.

Testování v různých kombinacích umožňuje zjistit, která konfigurace systému poskytuje nejrelevantnější odpovědi a je tak nejvhodnější pro firemní nasazení. Zároveň přináší poznatky o tom, jak formát dokumentu, strategie dělení textu a typy embedding a jazykových modelů ovlivňují výsledek.

Tyto poznatky slouží jako základ pro doporučení ohledně struktury dokumentů, volby technologie a budoucí optimalizaci systému.

15.1 Popis testovacího postupu

Pro testování byla připravena sada konkrétních dotazů nad reálnými firemními dokumenty. Ke každé otázce byla definována:

- Referenční odpověď (tzv. „gold answer“), která vychází z informací ve zdrojovém dokumentu.
- Sada klíčových faktů, které musí odpověď obsahovat, aby byla považována za správnou.

Následně bylo provedeno dotazování na tyto otázky v různých kombinacích:

- Formát vstupního dokumentu (.pdf, .docx, .md).
- Strategie chunkování (podle odstavců, fixní velikost 100 slov s překryvem 1 slovo, fixní velikost 200 slov s překryvem 20 slov).
- Embedding modelu (TF-IDF, all-MiniLM-L6-v2, intfloat/e5-base-v2).
- Jazykového modelu (gemma:7b, llama3, llama3.1:8b, mistral).

15.2 Metody hodnocení

Každá vygenerovaná odpověď je porovnána s referenční odpovědí (gold answer). Dále je zjišťována přítomnost definovaných faktů (F1, F2, ..Fn), které reprezentují klíčové prvky správné odpovědi. Tyto faktory umožňují posoudit, zda model zachytil všechny zásadní informace, a to i v případech, kdy odpověď není formulována přesně stejně jako gold answer. Vyhodnocení kvality odpovědí kombinuje automatizované metriky s ručním hodnocením.

Pro každou otázku se vyhodnocují tyto metriky:

- **Cosine similarity** mezi odpověďmi modelu a referenční odpovědí, který měří jejich sémantickou podobnost.
- **Precision, Recall a F1-score** počítané na základě výskytu správně uvedených či chybějících klíčových faktorů.
- **„Přiznání, že nevím“** signalizuje schopnost modelu rozpoznat nejistotu.
- **Počet slov v odpovědi** jako ukazatel stručnosti nebo rozvláčnosti odpovědi.

Dále jsou odpovědi ručně hodnoceny z hlediska následujících tří kvalitativních aspektů, které poskytují hlubší vhled do reálné použitelnosti výstupů:

- **Přítomnost dodatečných informací** – odhaluje, zda model doplňuje do odpovědi další informace, které nejsou obsaženy v gold answer.
- **Jazyková nepřesnost** – posuzuje gramatickou a stylistickou kvalitu odpovědi.
- **Nesrozumitelnost odpovědi** – určuje, zda odpověď dává smysl, je pravdivá a skutečně reaguje na položenou otázku.

Výsledky všech metrik jsou zpracovány pomocí tabulek a grafů, což umožňuje srovnání výstupů napříč jednotlivými jazykovými modely, embeddingovými modely, strategiemi chunkování a formátem vstupních dokumentů. Kombinace kvantitativních i kvalitativních kritérií, tak představuje komplexní hodnocení schopnosti systému poskytovat přesné, stručné a věrohodné odpovědi.

15.3 Výpočet metrik

15.3.1 Cosine similarity

Cosine similarity je metrika, která měří úhel mezi dvěma vektory v prostoru. Tedy jak moc vektory směřují stejným směrem bez ohledu na jejich délku.

V aplikaci je cosine similarity počítána tak, že se gold answer a odpovědi převedou pomocí embedding modelu *SentenceTransformer*(„all-MiniLM-L6-v2“) na vektorové reprezentace. Pomocí *cos_sim()* se potom vypočítá kosinová podobnost. Aplikace následně vrací číselný údaj, který vyjadřuje to, jak moc je odpověď od modelu sémanticky podobná s gold answer nezávisle na formulaci odpovědi nebo počtu slov.

- Hodnota 1 odpovědi jsou ve svém významu totožné.
- Hodnota 0 odpovědi jsou zcela nesouvisející.
- Hodnota -1 odpovědi jsou významově opačné.

15.3.2 Metriky Precision, Recall a F1-score

Pro kvantitativní hodnocení správnosti odpovědí generovaných jazykovým modelem byly zvoleny klasické metriky Precision, Recall a F1-score. Tyto metriky umožňují vyjádřit, jak přesně a úplně dokázal model reprodukovat jednotlivé faktory referenční odpovědi.

Každé testované otázce byla přiřazena sada hodnotících faktorů, které reprezentují nejdůležitější prvky správné odpovědi. Každá odpověď byla prostudována a ručně bylo zaznamenáno, zda daný byl daný fakt v odpovědi přítomen (ohodnocení 1) nebo chyběl (ohodnocení 0).

Hodnocení probíhá podle následujících pravidel:

- True Positive (TP): fakt je obsažen v gold answer a model ho správně uvedl
- False Negative (FN): fakt je obsažen v gold answer a model ho neuvedl
- False Positive (FP): model uvedl fakt, ale v gold answer se nevyskytuje

Za chybu je považováno uvedení takových dalších informací, které se nevyskytují v gold answer a zároveň působí zmatečně anebo se nenacházejí ve zdrojovém dokumentu. Hodnota 1 pro FP je získána tehdy, pokud je současně „Přítomnost dodatečných informací“ a „Nesrozumitelnost odpovědi“ > 0 .

Metriky se vypočítají takto:

- Precision (přesnost): Podíl správně uvedených faktů vůči všem faktům, které model uvedl. Výsledek 1 znamená, že model neuvedl navíc žádné nesprávné informace.

$$P = \frac{TP}{TP + FP} \quad (6)$$

- Recall (úplnost): Podíl správně uvedených faktů vůči všem faktům, které měly být zmíněny. Výsledek 1 znamená, že model zachytil všechny požadované informace.

$$R = \frac{TP}{TP + FN} \quad (7)$$

- F1-score: Harmonický průměr přesnosti a úplnosti. Výsledná hodnota 1 značí ideální rovnováhu mezi přesností a úplností.

$$F1 = \frac{2 \times P \times R}{P + R} \quad (8)$$

15.3.3 Přítomnost dalších informací

Tento parametr testuje, zda model uvádí informace nad rámec „gold answer“ a případně přidává informace mimo zdrojový dokument. V promptu vloženém do modelu je uveden pokyn „Odpověz pouze na základě uvedeného textu“, proto je přidávání informací považováno za nežádoucí. Může jít o tzv. halucinaci modelu.

Škála hodnocení (počet bodů):

- 0 odpověď čerpá pouze z poskytnutého textu
- 1 odpověď obsahuje drobnou informaci navíc např. parafrázi nebo shrnutí
- 2 odpověď obsahuje zcela nové informace, které se ve zdrojovém dokumentu nevyskytují.

15.3.4 Jazyková nepřesnost

Odpověď může obsahovat drobné gramatické nebo stylistické chyby či překlapy.

Škála hodnocení (počet bodů):

- 0 bez chyb
- 1 menší nepřesnosti, méně rušivé chyby, ale význam zůstává stejný
- 2 výrazné chyby snižující srozumitelnost nebo působí neprofesionálně

15.3.5 Nesrozumitelnost odpovědi

Nesrozumitelná odpověď je prakticky nepoužitelná, a to i kdyby byla gramaticky správná. Proto se pozornost věnuje i parametru srozumitelnosti.

Škála hodnocení (počet bodů):

- 0 odpověď je srozumitelná a věcná
- 1 mírně odbočuje, nebo není přesná
- 2 odpověď je zmatená, nejasná nebo nesouvisí s otázkou.

15.4 Návrh testovacích otázek

Aby bylo možné důkladně ověřit funkčnost aplikace, byly pro jednotlivé testovací dokumenty navrženy dotazy, které pokrývají různé typy schopností systému RAG. Tyto otázky byly rozděleny do kategorií podle toho, co konkrétně testují, což umožňuje komplexněji zhodnotit silné a slabé stránky systému a jednotlivých modelových konfigurací.

Každá z níže testovaných kategorií představuje jiný typ výzvy, které může systém při práci s reálnými dokumenty čelit:

1. Přesné vyhledání

Dotazy cílené na jednoznačnou informaci doslovně přítomnou v textu. Slouží k ověření, zda systém umí najít přesné odpovědi, například čísla nebo výčty.

2. Sémantická podobnost

Otázky, které nejsou formulovány stejně jako text v dokumentu, ale systém by je měl pochopit na základě významu. Testuje porozumění obsahu a kvalitu embeddingů.

3. Tabulkové údaje

Tyto otázky testují schopnost systému porozumět a extrahovat informace z tabulek. Je důležitá pro dokumenty, které často obsahují technické parametry nebo výčet hodnot.

4. Výčty

Dotazy vyžadující, aby model správně identifikoval a reprodukoval seznam položek. Hodnotí se zejména úplnost výčtu, správné oddělení jednotlivých prvků. U pořadových položek potom schopnost reprodukovat jejich smysl ve správném pořadí.

5. Vizuální obsah

Cílí na dokumenty s obrázky, značkami nebo schémata. Ačkoli samotný model nevidí obrázky, dobře napsané věty (např. popisky obrázků) mohou pomoci LLM odpovědět.

6. Více větové uvažování

Dotazy vyžadující syntézu více částí textu nebo vícero odstavců. Zde se testuje schopnost modelu udržet kontext a logicky usuzovat.

7. Dotazy mimo znalosti dokumentu

Sem patří dotazy, na které se nedá odpovědět na základě daného dokumentu. Cílem je zjistit, zda model správně odpoví „Nevím“ a nehalucinuje.

Díky této kategorizaci lze jednotlivé modely a embedding strategie hodnotit nejen podle přesnosti odpovědí, ale i podle typu úloh, které zvládají lépe či hůře. To má praktický význam pro výběr modelů do různých scénářů.

15.4.1 Kategorie Přesné vyhledání

Kategorie: **Přesné vyhledání**

Dokument: Zpráva Digitalizace dávek nemocenského pojištění

Otázka: **Co má zaměstnanec udělat, pokud onemocní?**

Gold answer: Zaměstnanec musí neprodleně informovat svého přímého nadřízeného e-mailem nebo telefonicky. Nic dalšího není třeba dělat.

- F1: Zaměstnanec musí informovat zaměstnavatele.
F2: Informace musí být podána neprodleně.
F3: Informování se provádí e-mailem nebo telefonicky.
F4: Informace se posílá přímému nadřízenému.

15.4.2 Kategorie Sémantická podobnost

- Kategorie: Sémantická podobnost
Dokument: Tahová zkouška
Otázka: **Jak zjistím, které tahové zařízení mám použít pro testování konkrétního vodiče?**
Gold answer: Správné zařízení se vybírá podle průměru drátu a maximální tahové síly. Tyto údaje jsou uvedeny přímo na zařízení.
F1: Zmínka o průměru drátu jako kritériu výběru
F2: Zmínka o maximální tahové síle
F3: Informace, že tyto údaje jsou uvedeny na zařízení
F4: Vyplývá, že se jedná o výběr zařízení (volba odpovídajícího přístroje)

Tato otázka není doslovnou kopií žádné věty z dokumentu, ale ptá se na informaci, která je v textu obsažena, jen je tam formulována jinými slovy a strukturou. Nestačí jen vyhledat klíčová slova typu „použít“, „zařízení“, „vodič“, ale model musí porozumět významu dotazu a najít, že volba zařízení závisí na průměru vodiče a maximální tahové síle.

15.4.3 Kategorie Tabulkové údaje

- Kategorie: Tabulkové údaje
Dokument: Směrnice Standardizace výpočetní techniky
Otázka: **Jaká je doporučená doba obměny pro zařízení typu Ultrabook?**
F1: Zmíněno zařízení „Ultrabook“
F2: Zmíněna hodnota „4 roky“ nebo „čtyři roky“

15.4.4 Kategorie Výčty

Kategorie: **Výčty**

Dokument: Zpráva Digitalizace dávek nemocenského pojištění

Otázka: **Které dávky nemocenského pojištění byly digitalizovány?**

Gold answer: Ošetřovné, dlouhodobé ošetřovné, peněžitá pomoc v mateřství, otcovská, vyrovnávací příspěvek v těhotenství a mateřství.

F1: Ošetřovné.

F2: Dlouhodobé ošetřovné.

F3: Peněžitá pomoc v mateřství.

F4: Otcovská.

F5: Vyrovnávací příspěvek v těhotenství a mateřství.

15.4.5 Kategorie Vizuální obsah

Kategorie: **Vizuální obsah**

Dokument: Dresscode společnosti

Otázka: **Jaké typy obuvi jsou nevhodné do kanceláře?**

Gold answer: Nevhodnou obuví jsou sportovní a plážové boty, domácí obuv, žabky, pantofle a jakákoli jiná obuv, která se nehodí ke zvolenému typu oblečení.

F1: Je uvedeno, že sportovní nebo plážová obuv není vhodná.

F2: Jsou zmíněny žabky a/nebo pantofle.

F3: Je uvedeno, že se obuv musí hodit ke zvolenému typu oblečení.

F4: Je zmíněna domácí obuv jako nevhodná.

15.4.6 Kategorie Více větné uvažování

Kategorie: **Více větné uvažování**

Otázka: **Jakým způsobem se určuje čas pro výměnu zařízení a kdo připravuje rozpočet na jeho obměnu?**

Gold answer: Čas výměny zařízení se určuje podle stáří zařízení a míry jeho opotřebení. Rozpočet na obměnu ICT připravuje OPIATI v roce, kdy zařízení dosáhne morálního opotřebení.

F1: Uvedeno, že výměna závisí na stáří nebo morálním opotřebení

F2: Uvedeno, že rozpočet připravuje OPIATI

F3: Je uvedeno, že rozpočet se připravuje na konkrétní rok

F4: Logická návaznost informací (zařízení → stav → rozpočet → výměna)

15.4.7 Kategorie Dotazy mimo znalosti dokumentu

Kategorie: Dotazy mimo znalosti dokumentu

Otázka: **Jak často je nutné kalibrovat zařízení pro tahovou zkoušku?**

Gold answer: Nevím. Tato informace není v dokumentu uvedena.

F1: Model správně uvede, že informace není k dispozici

F2: Model nehalucinuje žádné technické podrobnosti

F3: Neobsahuje smyšlené intervaly ani odkazy na údržbu

F4: Jasně komunikuje absenci údajů v dokumentu

16 STATISTICKÉ VYHODNOCENÍ VÝSLEDKŮ

Tato kapitola se věnuje statistickému vyhodnocení výstupů testování jednotlivých nastavení RAG aplikace nad sadou testovacích otázek (kapitola 15.4). Všechna data použitá v analýze byla získána jako výstup z vytvořené aplikace RAG. Pro následné zpracování a vyhodnocení výsledků byly využity nástroje DATAtab a Microsoft Excel.

Obsahová úplnost odpovědí byla zhodnocena pomocí radarového grafu pokrytí klíčových faktorů odpovědi. U každé odpovědi modelu se ručně zaznamenávalo, zda daný faktor byl v odpovědi přítomen či nikoliv. Výsledné hodnoty byly zprůměrovány a zaneseny do grafu, který umožňuje přehledně porovnat, do jaké míry jednotlivé modely dokázaly pokrýt obsahovou strukturu odpovědi podle očekávání.

V rámci rozšíření pohledu na výstupy jazykových modelů byla provedena také analýza délky generovaných odpovědí, měřená počtem slov. Cílem této analýzy je zjistit, zda se jednotlivé modely výrazně liší v rozsahu svých výstupů, což může ovlivňovat nejen celkovou srozumitelnost, ale i efektivitu při vyhledávání konkrétní informace. Zároveň byla pro každý model zjištěna také kvalita výstupu z hlediska jazykové nepřesnosti (gramatické a stylistické chyby) a nesrozumitelnosti odpovědi (subjektivní určení toho, zda odpověď dává smysl, je pravdivá a skutečně reaguje na položenou otázku). Cílem je tedy doplnit kvantitativní pohled na délku výstupu o kvalitativní jazykový rozměr, který může ovlivňovat praktickou použitelnost generovaných textů.

Pro lepší porozumění tomu, jak jednotlivé metriky zachycují kvalitu generovaných odpovědí, byla provedena korelační analýza mezi třemi různými charakteristikami. F1-score a Cosine to Gold byly získány automatizovaně. F1-score měří přesnost a úplnost odpovědi ve srovnání s referenční odpovědí. Cosinová podobnost s referenční odpovědí hodnotí sémantickou blízkost mezi odpovědí modelu a referenční odpovědí. Třetí charakteristika, kvalitativní hodnocení, byla získána ručně na základě subjektivního posouzení relevance, přesnosti a srozumitelnosti odpovědí. Cílem analýzy bylo ověřit, do jaké míry automatizované metriky odpovídají lidskému vnímání kvality a zda je možné je použít jako spolehlivou náhradu ručního hodnocení.

Pro analýzu jednotlivých faktorů a jejich vzájemných kombinací na kvalitu generovaných odpovědí byla zvolena metoda vícefaktorové analýzy rozptylu (3-way ANOVA). Tato metoda umožňuje nejen posoudit, zda jednotlivé faktory (formát vstupního dokumentu, strategie chunkování, embedding model a jazykový model) mají statistický

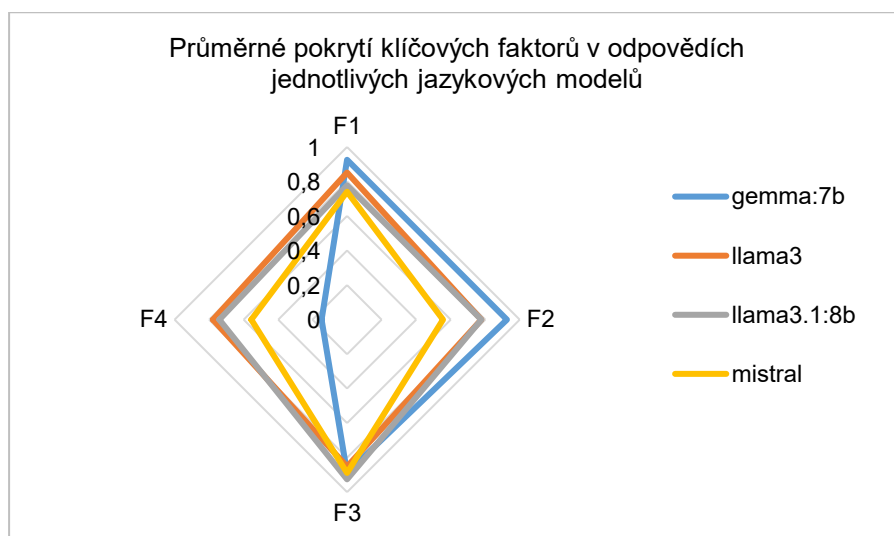
významný vliv na zvolený ukazatel (F1-score a kvalitativní hodnocení), ale také odhalit interakce mezi faktory. Hledají se takové situace, kdy účinek jednoho faktoru závisí na nastavení jiného. Díky této metodě je možné zjistit například to, zda určitý embedding model funguje lépe jen s konkrétní strategií chunkování, nebo zda výběr modelu ovlivňuje výstup jen v kombinaci s určitým formátem dokumentu. Výsledky této analýzy by měli přispět k pochopení toho, které parametry nastavení systému mají skutečný dopad na výstup, a které jsou méně podstatné.

V následujících kapitolách budou výsledky podrobně analyzovány na příkladu jedné konkrétní otázky z kategorie Přesného vyhledání. Tato otázka byla zvolena jako reprezentativní příklad, na kterém lze ukázat celý proces vyhodnocení kvality odpovědí a vlivu různých parametrů systému.

V další části jsou výsledky pro všech sedm kategorií dotazů zpracovány agregovaně. Cílem této části je zobecnit zjištění a určit, jaké nastavení systému dosahuje nejlepších výsledků napříč různými typy dotazů.

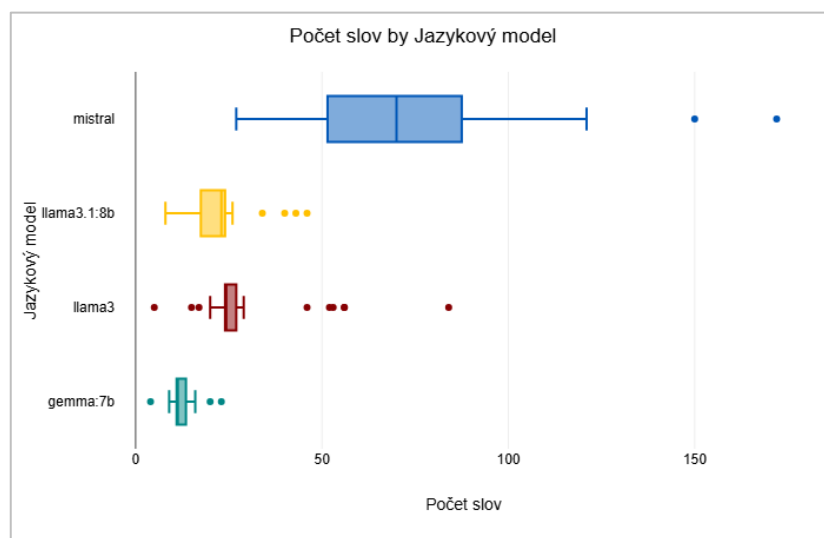
16.1 Vyhodnocení kategorie Přesné vyhledání

Z radarového grafu (Obrázek 17) je patrné, že ač si model gemma:7b udržel vysoké pokrytí u faktorů F1-F3, tak výrazně zaostal u faktoru F4, kdy ve svých odpovědích opomíhal zmínit, že: „Informace se posílá přímému nadřízenému“. Z hlediska celkového průměrného pokrytí všech čtyř faktorů vychází nejlépe model llama 3.



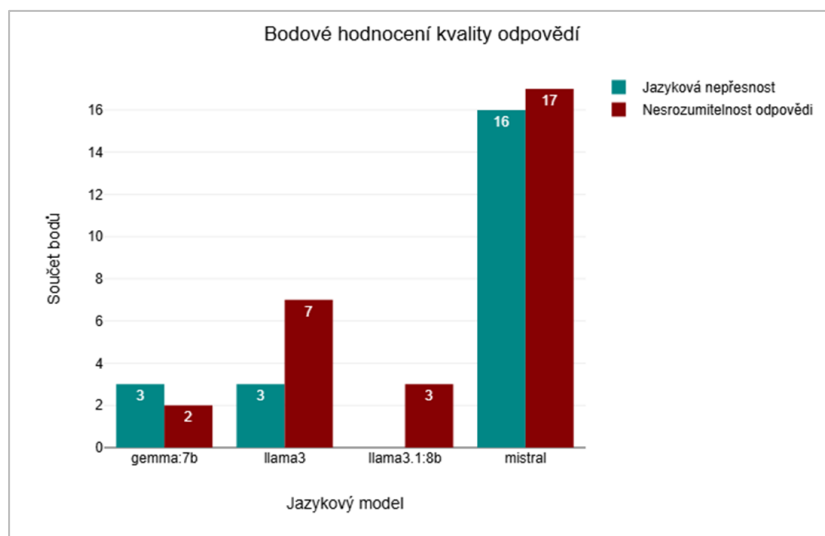
Obrázek 17 Průměrné pokrytí klíčových faktorů v odpovědích
(zdroj: vlastní)

V rámci sledování délky generovaných odpovědí byly mezi jednotlivými jazykovými modely zaznamenány výrazné rozdíly, které jsou znázorněny v krabicovém grafu (Obrázek 18). Model gemma:7b produkoval nejkratší odpovědi s průměrným počtem slov 12,3, zatímco mistral generoval odpovědi výrazně delší o průměrném počtu 75,7 slov. Modely llama3 a llama3.1:8b se nacházely mezi těmito extrémy s průměrem 30,3 a 23,0 slov. Zároveň lze pozorovat, že modely s delšími odpověďmi vykazovaly i větší rozptyl v délce. Například mistral $\pm 33,8$ slov, což může svědčit o větší flexibilitě, ale i nestabilitě v generovaném výstupu.



Obrázek 18 Porovnání jazykových modelů z hlediska počtu slov v odpovědi
(zdroj: vlastní)

Následující graf (Obrázek 19) porovnává kvalitu odpovědí jazykových modelů z hlediska subjektivního posouzení jazykové nepřesnosti a nesrozumitelnosti odpovědí. Model Mistral dosáhl nejvyššího počtu bodů za oba typy nedostatků. Naopak ostatní modely vykazovaly v těchto oblastech výrazně nižší skóre. Model llama3.1:8b měl nejnížší jazykové nedostatky.



Obrázek 19 Porovnání jazykových modelů z hlediska jazykové kvality
(zdroj: vlastní)

Výsledky obou grafů (Obrázek 18) (Obrázek 19) naznačují, že delší odpovědi byly méně konzistentní a obsahovaly více chyb. Oproti tomu modely s kratšími odpověďmi např. gemma:7b si udržely lepší jazykovou kvalitu, ale jejich výstupy mohly být informačně omezenější.

Korelační matice (Tabulka 5) mezi třemi různými metrikami hodnocení kvality odpovědi ukazuje, že velmi silná pozitivní korelace existuje mezi Kvalitativním hodnocením a F1-score ($r = 0,8899$). F1-score tedy velmi dobře odráží vnímanou kvalitu odpovědi. Docela silný vztah je i mezi Kvalitativním hodnocením a Cosine to Gold ($r = 0,6644$).

Tabulka 5 Korelační matice metrik hodnocení odpovědi

Korelace $r(108)$	Cosine to Gold	F1-score	Kvalitativní hodnocení
Kvalitativní hodnocení	0,6644		
Cosine to Gold		0,622	
F1-score			0,8899

(zdroj: vlastní)

Dále byla provedena třicestná analýza rozptylu ANOVA. Pro lepší přehled byly výsledky jednotlivých variant analýzy rozptylu shrnuty do dvou tabulek podle metrik (Tabulka 6). Zeleně jsou zvýrazněny statisticky významné hodnoty ($p < 0,05$). Tento přehled

umožňuje porovnání toho, které faktory či kombinace mají největší vliv na kvalitu výstupu při různém nastavení vstupních parametrů.

Výsledky p-hodnot ukazují, že kvalitu výstupu výrazně ovlivňuje jednak výběr konkrétního jazykového modelu (LLM), a dále strategie chunkování a embedding model. Zejména interakce Formát dokumentu × Strategie chunkování a Strategie chunkování × Embedding model se ukázaly jako opakovaně významné pro obě metriky.

Tabulka 6 Souhrnné p-hodnoty faktorů a jejich interakcí
pro metriky F1-score a Kvalitativní hodnocení

F1-score p value	Formát dokumentu*Chunking*Embedding	Formát dokumentu*Chunking*LLM	Chunking*Embedding*LLM	Formát dokumentu*Embedding*LLM	Kvalitativní hodnocení p value	Formát dokumentu*Chunking*Embedding	Formát dokumentu*Chunking*LLM	Chunking*Embedding*LLM	Formát dokumentu*Embedding*LLM
Model	< .001	0,065	0,655	0,936	Model	0,001	< .001	0,132	0,438
Formát dokumentu	0,968	0,981	---	0,986	Formát dokumentu	0,483	0,459	---	0,589
Chunking	0,048	0,167	0,229	---	Chunking	0,078	0,066	0,128	---
Embedding	0,038	---	0,203	0,233	Embedding	0,137	---	0,202	0,235
LLM	---	0,375	0,465	0,505	LLM	---	< .001	0,001	0,002
Formát dokumentu*Chunking	< .001	< .001	---	---	Formát dokumentu*Chunking	< .001	< .001	---	---
Formát dokumentu*Embedding	0,001	---	---	0,061	Formát dokumentu*Embedding	0,082	---	---	0,194
Chunking*Embedding	< .001	---	0,008	---	Chunking*Embedding	0,004	---	0,015	---
Formát dokumentu*LLM	---	0,925	---	0,962	Formát dokumentu*LLM	---	0,857	---	0,939
Chunking*LLM	---	0,936	0,96	---	Chunking*LLM	---	0,657	0,792	---
Embedding*LLM	---	---	0,969	0,975	Embedding*LLM	---	---	---	0,954
Formát dokumentu*Chunking*Embedding	0,046	---	---	---	Formát dokumentu*Chunking*Embedding	0,733	---	---	---
Formát dokumentu*Chunking*LLM	---	0,383	---	---	Formát dokumentu*Chunking*LLM	---	0,08	---	---
Chunking*Embedding*LLM	---	---	0,977	---	Chunking*Embedding*LLM	---	---	0,978	---
Formát dokumentu*Embedding*LLM	---	---	---	0,972	Formát dokumentu*Embedding*LLM	---	---	---	0,891

(zdroj: vlastní)

16.2 Agregované vyhodnocení

Po podrobné analýze otázky z kategorie Přesného vyhledání byly dále vyhodnoceny výsledky všech sedmi otázek v datasetu agregovaně. Cílem této části je určit, které kombinace parametrů (jazykový model, embedding model, strategie chunkování a vstupní formát dokumentu) přináší nejlepší výsledky celkově, a případně také rozlišit jejich výkonnost podle typu otázky.

16.2.1 Hodnocení výskytu klíčových informací v odpovědích

Pro vyhodnocení kvality odpovědí generovaných RAG aplikací byla provedena ruční analýza úspěšnosti výskytu klíčových informací v jednotlivých odpovědích. U každé testované otázky byly předem definovány konkrétní faktické prvky (dále jen faktory F1, F2, ...), které by měla správná odpověď obsahovat. Poté byly analyzovány všechny odpovědi vygenerované různými kombinacemi parametrů RAG aplikace. Celkem se jednalo o 108 odpovědí na každou otázku. Pokud byl daný fakt v odpovědi obsažen, byl výskyt hodnocen hodnotou 1, pokud nebyl, tak hodnotou 0.

V tabulce (Tabulka 7) jsou přehledně znázorněné jednotlivé otázky se svými zdrojovými dokumenty a hledanými faktickými údaji. Výsledky jsou shrnuty v poslední sloupci tabulky, který uvádí míru úspěšnosti detekce každého faktu napříč všemi odpověďmi.

Nejvyšší úspěšnost vykazovaly dotazy typu Výčty a Sémantická podobnost, kdy se správné informace objevily ve více než 85 % případů. Velmi dobrých výsledků bylo dosaženo i u dotazů typu Mimo znalosti dokumentu. Zde dosahovala úspěšnost více než 95 %. Naopak nejnižší výskyt klíčových informací byl zaznamenán u dotazů na Tabulkové údaje a u otázek týkajících se Vizuálního obsahu.

Tato analýza ukazuje, že přestože některé odpovědi mohou na první pohled působit relevantně, často v nich chybí klíčové faktické prvky.

Tabulka 7 Přehled výskytu klíčových informací v odpovědích
podle typu dotazu

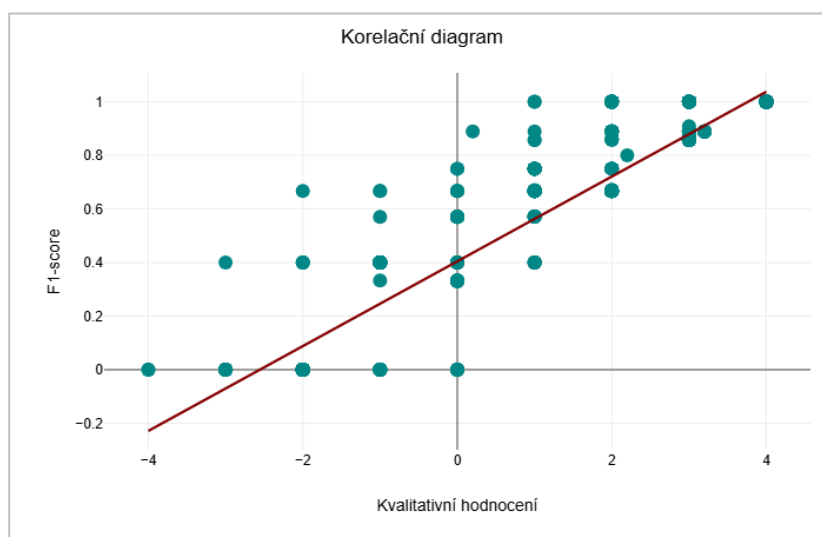
Druh dokumentu	Název dokumentu	Kategorie	Otázka	Faktor	Co musí odpověď obsahovat	Úspěšnost výskytu faktoru v odpovědi (%)
Zpráva nemocenského pojištění	Digitalizace dávek nemocenského pojištění	Přesné vyhledání	1. Co má zaměstnanec dělat, pokud onemocní?	F1 Zaměstnanec musí informovat zaměstnavatele.		82%
		Výčty	2. Které dávky nemocenského pojištění byly digitalizovány?	F2 Informace musí být podána nepodléh.		76%
				F3 Informování se provádí e-mailem nebo telefonicky.		89%
				F4 Informace se posílá přímému nadřízenému.		56%
				F1 Ošetřovné		89%
				F2 Dlouhodobé ošetřovné		89%
				F3 Peněžní pomoc v mateřství		89%
				F4 Otcovská		89%
				F5 Vyrovňovací příspěvek v těhotenství a mateřství		79%
				F1 Zmíněno zařízení „Ultrabook“		15%
Směrnice výpočetní techniky	Standardizace výpočetní techniky	Tabulkové údaje	3. Jaká je doporučená doba obměny zařízení typu Ultrabook?	F2 Zmíněna hodnota „4 roky“ nebo „čtyři roky“		33%
		Více větné uvažování	4. Jakým způsobem se určuje čas pro výměnu zařízení a kdo připravuje rozpočet na jeho obměnu?	F1 Uvedeno, že výměna závisí na stáří nebo morálním opotřebení		79%
				F2 Uvedeno, že rozpočet připravuje OPIATI		77%
				F3 Je uvedeno, že rozpočet se připravuje na konkrétní rok		26%
				F4 Logická návaznost informací (zařízení → stav → rozpočet → výměna)		54%
				F1 Zmínka o průměru drátu jako kritériu výběru		69%
				F2 Zmínka o maximální tahové síle		69%
				F3 Informace, že tyto údaje jsou uvedeny na zařízení		68%
				F4 Vyplývá, že se jedná o výběr zařízení (volba odpovídajícího přístroje)		96%
Pracovní postup	Provedení tahové zkoušky	Sémantická podobnost	5. Jak zjistím, které tahové zařízení mám použít pro testování konkrétního vodiče?	F1 Model správně uvede, že informace není k dispozici		100%
		Mimo dokument	6. Jak často je nutné kalibrovat zařízení pro tahovou zkoušku?	F2 Model nehašluje žádné technické podrobnosti		97%
				F3 Neobsahuje smyšlené intervaly ani odkazy na údržbu		97%
				F4 Jasně komunikuje absenci údajů v dokumentu		87%
				F1 Je uvedeno, že sportovní nebo plážová obuv je nevhodná		33%
				F2 Jsou zmíněny zábrty a/nebo pantoře		37%
				F3 Je uvedeno, že se obuv musí hodit ke zvolenému typu obléčení		26%
				F4 Je zmíněna domácí obuv jako nevhodná		31%
Vizuální standard	Dresscode společnosti	Vizuální obsah	7. Jaké typy obuvi jsou nevhodné do kanceláře?			

(zdroj: vlastní)

16.2.2 Nakolik automatické metriky odpovídají lidskému vnímání hodnocení kvality odpovědí

Zatímco výše zmíněná analýza se soustředila na výskyt konkrétních faktů v odpovědích, následující část se zabývá vztahem mezi ručním hodnocením kvality výstupů a automatickými metrikami. Cílem je zjistit, nakolik metriky jako F1-score nebo Cosine to gold odpovídají lidskému vnímání kvality odpovědí a za je lze spolehlivě použít pro vyhodnocování výstupů systému.

Pro komplexní posouzení vztahů mezi metrikami F1-score, Cosine to Gold a Kvalitativním hodnocením byla vytvořena korelační matice (Tabulka 8) a párové korelační diagramy (Obrázek 20). Z výsledků vyplývá, že nejvyšší korelace byla mezi F1-score a Kvalitativním hodnocením ($r = 0,9522$, $p < 0,001$), což potvrzuje, že F1-score velmi dobře odpovídá lidskému vnímání kvality odpovědí.



Obrázek 20 Korelační diagram pro porovnání vztahů F1-score a Kvalitativního hodnocení
(vlastní zpracování)

Korelace mezi Cosine to Gold a Kvalitativním hodnocením byla rovněž statisticky významná ($r = 0,6235$, $p < 0,001$), ale viditelně slabší. To naznačuje, že cosinová podobnost sice vystihuje sémantickou blízkost odpovědi k referenční odpovědi, ale méně zachycuje jazykovou a stylistickou kvalitu.

Tabulka 8 Souhrnná korelační matice metrik hodnocení odpovědí

Korelace r(756)	Cosine to Gold	F1-score	Kvalitativní hodnocení
Kvalitativní hodnocení	0,6235		
Cosine to Gold		0,6077	
F1-score			0,9522

(zdroj: vlastní)

Také korelace mezi F1-score a Cosine to Gold ($r = 0,6077$, $p < 0,001$) potvrzuje, že obě metriky se do určité míry shodují, ale každá z nich hodnotí jiný aspekt kvality. F1-score vyžaduje přesné překrytí klíčových prvků odpovědi, cosinová podobnost hodnotí obecnou tematickou a sémantickou podobnost.

Při bližším pohledu na otázku zaměřenou na přesné vyhledání konkrétní hodnoty z tabulky (Tabulka 9) se ukázalo, že Cosine to Gold nemusí být spolehlivým ukazatelem správnosti odpovědi. Přestože totiž některé odpovědi byly věcně nesprávné (např. „5 let“, nebo „Nevím“), dosáhly relativně vysoké hodnoty Cosine to Gold (cca 0,7). Naproti tomu odpověď „4 roky.“ získala nižší Cosine to Gold (0,37), ačkoliv byla částečně správná. Tento příklad ukazuje, že cosinová podobnost měří pouze tematickou nebo sémantickou podobnost, ale nezachycuje přesnost odpovědi ve vztahu k faktickému údaji, což je zásadní při otázkách směřujících na konkrétní hodnoty.

Tabulka 9 Porovnání F1-score a Cosine to Gold pro různé odpovědi na otázku zaměřenou na získání konkrétní hodnoty z tabulky

Otázka: Jaká je doporučená doba obměny pro zařízení typu Ultrabook? Referenční odpověď: Ultrabook má stanovenou dobu obměny 4 roky.	Zmíněno zařízení „Ultrabook“	Zmíněna hodnota „4 roky“ nebo „čtyři roky“	Metriky		
			Kvalitativní hodnocení	F1-score	Cosine to Gold
Odpověď jazykového modelu	F1	F2			
Doba obměny pro zařízení typu Ultrabook je podle dokumentu nastavena na 5 let.	0	0	-2	0,00	0,726141
Nevím.	0	0	-2	0,00	0,161396
4 roky.	0	1	1	0,67	0,372853
Doporučená doba obměny pro zařízení typu Ultrabook je po 4 letech.	1	1	2	1,00	0,737179
Nevím. Článek neuvádí žádnou konkrétní dobu obměny pro Ultrabook nebo jiná zařízení.	0	0	-2	0,00	0,748432

(zdroj: vlastní)

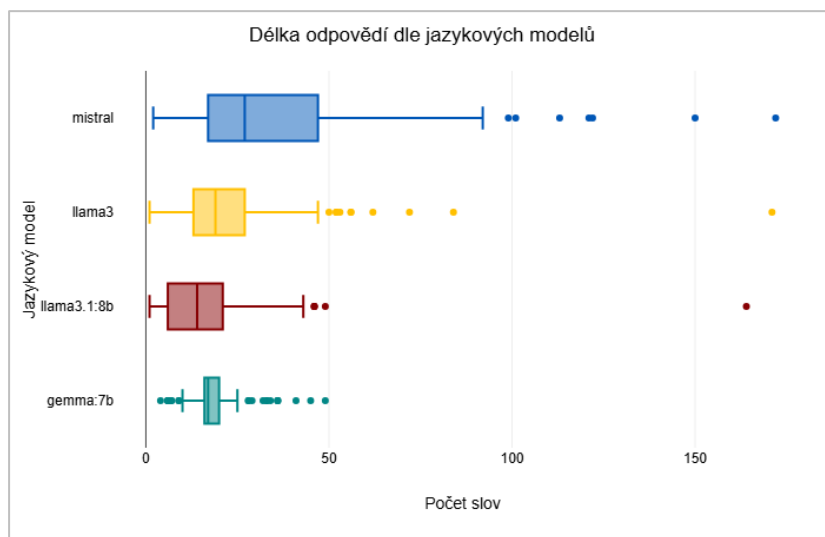
Na základě těchto zjištění doporučuji při hodnocení kvality odpovědi preferovat metodu F1-score na úkor Cosine to Gold. Cosinová podobnost může být nadále užitečná pro orientační posouzení sémantické blízkosti. Zvolená metoda ručního kvalitativního

hodnocení představuje důležitý doplněk k automatickým metrikám. Pro praktické ladění a ověřování výstupů různých nastavení RAG aplikace lze tuto metodu doporučit.

16.2.3 Vyhodnocení jazykové kvality výstupů

Aby bylo možné lépe vyhodnotit praktickou použitelnost výstupů jednotlivých jazykových modelů, byla kromě kvantitativních metrik analyzována také jazyková kvalita a srozumitelnost odpovědí. Hodnocena byla jazyková nepřesnost, kam se zařazovaly gramatické a stylistické chyby. A nesrozumitelnost odpovědí, kde negativní hodnocení dostávaly odpovědi, které nedávaly smysl, byly nelogické nebo nesprávné. V situaci, kdy model uvedl, že neví odpověď, ale při jiném nastavení model odpověď našel, byla daná konfigurace penalizována dvěma body.

Vzhledem k tomu, že srozumitelnost a jazyková přesnost odpovědí může souviset i s jejich délkou, byla nejprve provedena analýza počtu slov v jednotlivých odpovědích. Následující graf (Obrázek 21) ukazuje rozdělení délky odpovědí generovaných použitými jazykovými modely. Data jsou prezentována formou krabicového grafu, který zobrazuje medián, kvartily, rozsah i odlehlé hodnoty.



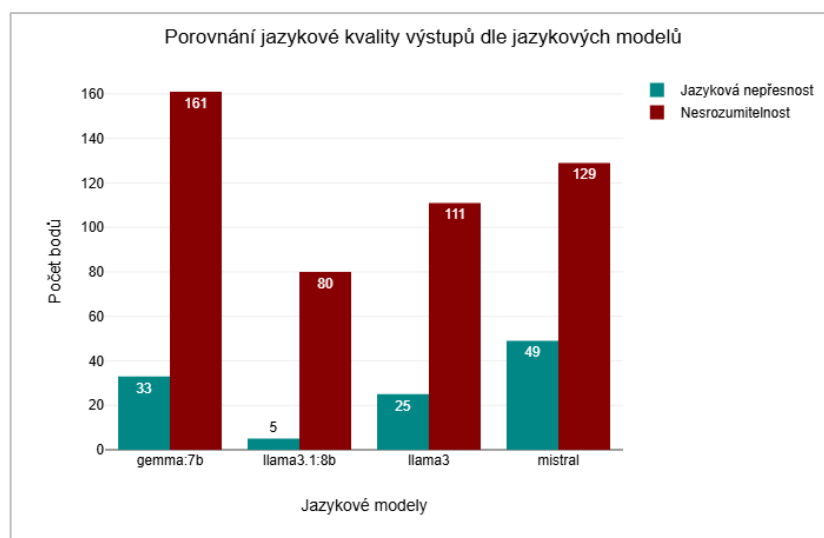
Obrázek 21 Porovnání délky odpovědí podle jazykového modelu

(zdroj: vlastní)

Z grafu lze vyčíst, že medián počtu slov u jazykového modelu mistral je výrazně vyšší (27 slov), než u ostatních modelů. Některé odlehlé hodnoty jsou až nad 150 slov. To může vést k lepšímu pokrytí tématu, ale také k vyššímu riziku jazykové nepřesnosti nebo nesrozumitelnosti odpovědi. Llama3 produkuje odpovědi střední délky a má méně odlehlých

hodnot než mistral. Gemma:7b má medián i celý rozsah odpovědí nízký, což odráží schopnost modelu generovat pouze stručné výstupy.

Pro hlubší pochopení těchto kvalitativních aspektů byly následně porovnány jazykové modely a jednotlivé kategorie dotazů z hlediska získaných negativních bodů za jazykové nepřesnosti a nesrozumitelnost odpovědí (Obrázek 22). Nejhuře si v této analýze vedl model gemma:7b, který získal nejvyšší počet penalizací za nesrozumitelnost (160 bodů) i měl vyšší počet jazykových chyb (33). Nejlepší výsledek vykázal model llama3.1:8b, který obdržel pouhých 5 bodů za jazykové chyby a 80 bodů za nesrozumitelnost. Model mistral měl nejvyšší podíl jazykových nepřesností (49), což může souviset s jeho delšími odpověďmi, jak bylo ukázáno v předchozím grafu (Obrázek 21).



Obrázek 22 Porovnání jazykové kvality dle jazykových modelů
(zdroj: vlastní)

Jedním z příkladů jazykového selhání bylo i nesprávné vyhodnocení zkratky OPIATI, která se vyskytla v otázce zaměřené na více větné uvažování. Jelikož v dokumentu chyběl výklad zkratky, model nebyl schopen správně interpretovat její význam. Model místo toho halucinoval různé neexistující výrazy jako například „Odbor počítačů a informační technologie a informatiky“, namísto správného významu „Odbor provozu ICT a technické infrastruktury“. Další varianty jsou uvedeny v tabulce (Tabulka 10).

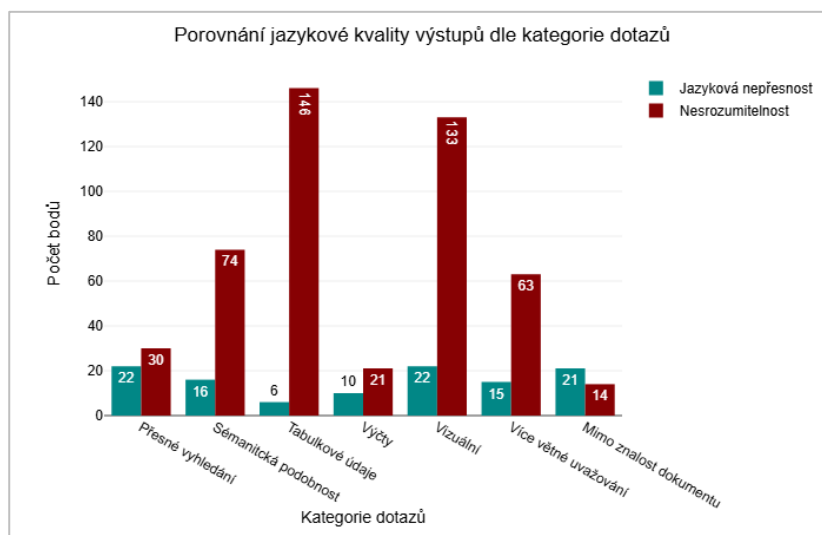
Tento případ poukazuje na důležitost uvádění zkratk spolu s jejich vysvětlením. Bez této informace může jazykový model generovat odpovědi, které jsou gramaticky správné, ale věcně zcela nepřesné. V rámci hodnocení byly tyto odpovědi penalizovány jedním bodem.

Tabulka 10 Ukázka halucinací jazykových modelů při interpretaci zkratky OPIATI

Cosine to Gold	Response	F1	F2	F3	F4	Dodatečné informace	Jazyková nepřesnost	Nesrozumitelnost
0,716047	Čas pro výměnu zařízení je určen podle tabulky obměny techniky dle stáří. OPIATI (Organizace pro podporu informační a komunikační technologie) připravuje rozpočet na jeho obměnu a plánuje harmonogram obměny.	1	1	0	0	1	0	1
0,771256	Odpověď: Čas pro výměnu zařízení je stanoven v harmonogramu, který se připravuje OPIATI (Odbor počítačů a informační technologie a informatiky) na rok, kdy bude dovršena technika „morálního opotřebení“. Rozpočet na obměnu se také připravuje OPIATI.	1	1	1	1	1	0	1
0,582828	Úměrně stáří techniky. OPIATI (Organizace pro podporu informačních technologických aplikací) připraví rozpočet na obměnu ICT.	1	1	0	0	1	0	1
0,812385	Čas pro výměnu zařízení je připravován OPIATI (Oddělením požívání a technického zabezpečení) , které vytvoří roční rozpočet na obměnu ICT. Tento rozpočet vychází z harmonogramu, který je definovaný podle stáří zařízení („morálního opotřebení“). Obměna ICT uživatelů následně probíhá dle tohoto harmonogramu.	1	1	0	1	1	0	1

(zdroj: vlastní)

Zajímavým výstupem je také srovnání kvality jazykového výstupu ve vztahu k jednotlivým druhům dotazů (Obrázek 23). Nejvíce penalizací za nesrozumitelnost bylo uděleno v kategorii Tabulkové údaje (146 bodů) a Vizualní obsah, což naznačuje, že tyto typy otázek jsou pro jazykové modely nejnáročnější. Naopak nejméně nesrozumitelných odpovědí bylo v kategoriích Výčty a Přesné vyhledání. Velmi úspěšné byly jazykové modely i v kategorii „Mimo znalosti dokumentu“, kdy až na 3 případy (ze 108) všechny testované varianty nastavení RAG aplikace odhalili, že odpověď na danou otázku se v textu nevyskytuje.



Obrázek 23 Porovnání jazykové kvality podle kategorie dotazu

(zdroj: vlastní)

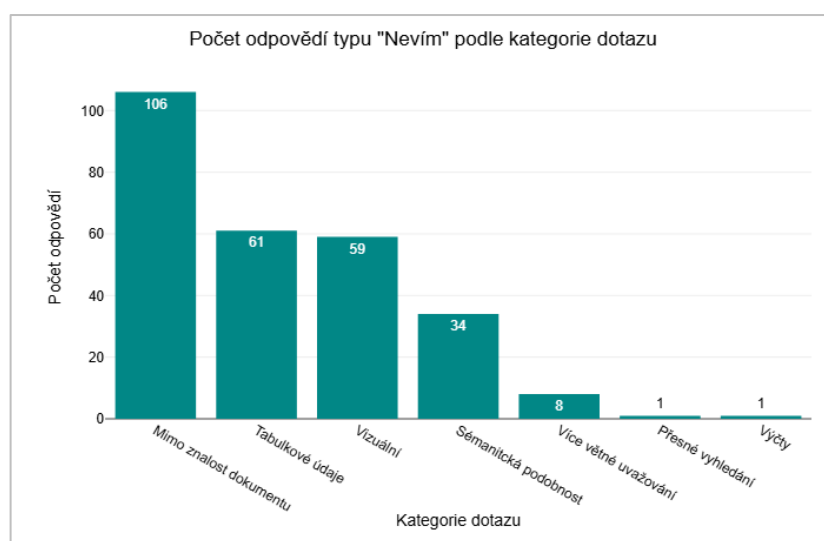
Výsledky potvrzují, že jazykové modely vykazují různé chování v závislosti na typu dotazu. Model llama3.1:8b si zachovává konzistentní kvalitu ve všech kategoriích, zatímco gemma:7b má výrazné problémy se srozumitelností odpovědí napříč celým spektrem.

Nejproblematictějším typem otázek z hlediska srozumitelnosti jsou ty, které vyžadují interpretaci tabulkových dat a vizuálního obsahu. Kombinace složité struktury a vyšších požadavků na kontext zde vede ke ztrátě kvality výstupu.

16.2.4 Posouzení výskytu odpovědí typu „Nevím“

Důležitou součástí hodnocení kvality výstupů RAG aplikace bylo sledování toho, ve kterých případech jazykové modely odpověděly výrokem „Nevím“, (případně významově podobným vyjádřením, že model nezná odpověď), ačkoli jiné konfigurace systému dokázaly na stejnou otázku odpovědět. Taková situace byla považována za chybu, neboť ukazuje na omezení schopnosti dané konfigurace nastavení zpracovat informace dostupné v dokumentu.

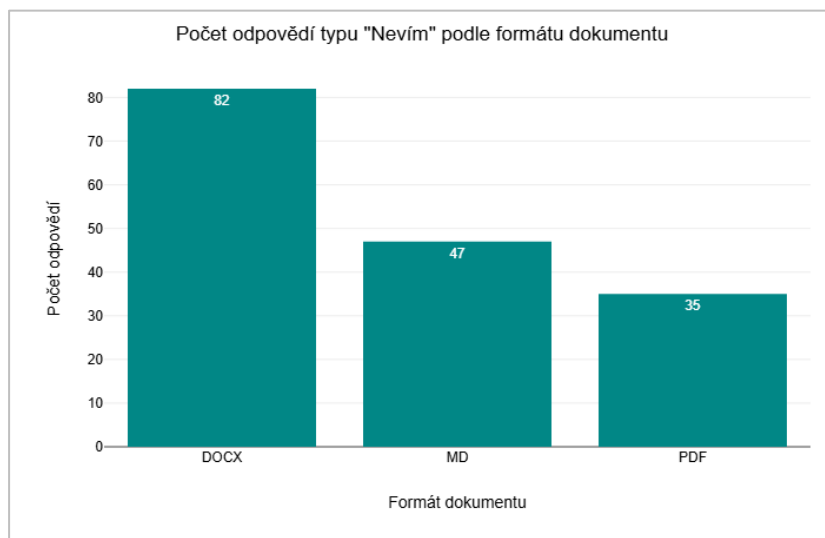
Graf (Obrázek 24) ukazuje, že odpověď „Nevím“ se nejčastěji vyskytovala v kategorii dotazů zaměřených na tabulkové údaje a vizuální obsah. Tato skutečnost naznačuje, že určité typy informací jsou pro systém náročnější na zpracování. V kategorii „Mimo znalosti dokumentu“ je odpověď modelu „Nevím“ žádoucí. V celkově 106 případech ze 108 možných, jazykové modely správně určili, že požadovaná informace se v dokumentu nenachází.



Obrázek 24 Počet odpovědí typu „Nevím“ podle kategorie dotazu
(zdroj: vlastní)

Během dalšího hodnocení, v rámci této kapitoly (16.2.4), už budou konfigurace systému pro kategorii dotazu „Mimo znalost dokumentu“ z dat vyloučeny, aby nedošlo ke zkreslení výsledků.

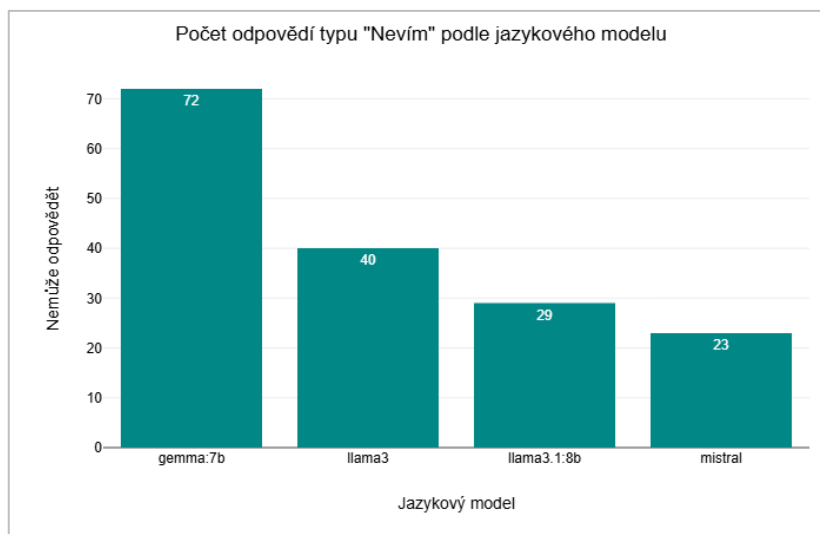
Rozdíly byly zaznamenány i v závislosti na formátu dokumentu (Obrázek 25). Nejvyšší podíl odpovědí typu „Nevím“ byl zaznamenán u dokumentů ve formátu DOCX, zatímco PDF dosahovalo nejlepších výsledků. Markdown formát se umístil mezi těmito dvěma variantami. Tyto výsledky podporují myšlenku, že standardizace dokumentů do jednoduššího formátu může pomoci při extrakci informací.



Obrázek 25 Počet odpovědí typu „Nevím“ podle formátu dokumentu

(zdroj: vlastní)

Dále byly analyzovány rozdíly mezi jednotlivými jazykovými modely (Obrázek 26). Nejčastěji odpovídal „Nevím“ model gemma:7b (v 72 případech ze 108 možných), naopak nejlépe si vedl jazykový model mistral (23 případů). Tato informace odpovídá i ostatním porovnáním týkající se kvality a spolehlivosti výstupů.



Obrázek 26 Počet odpovědí typu „Nevím“ podle jazykového modelu
(zdroj: vlastní)

Dalším zjištěním, které může vysvětlovat vyšší podíl odpovědí „Nevím“ u dokumentů ve formátu DOCX, jsou technická omezení při jejich zpracování. U některých vstupních souborů nebyla aplikace RAG schopna načíst obsah textových polí (Obrázek 27). Textovými poli se běžně ve Wordu zvýrazňují nebo oddělují části textu a v provozních dokumentech se velmi často používají.

NEVHODNÉ

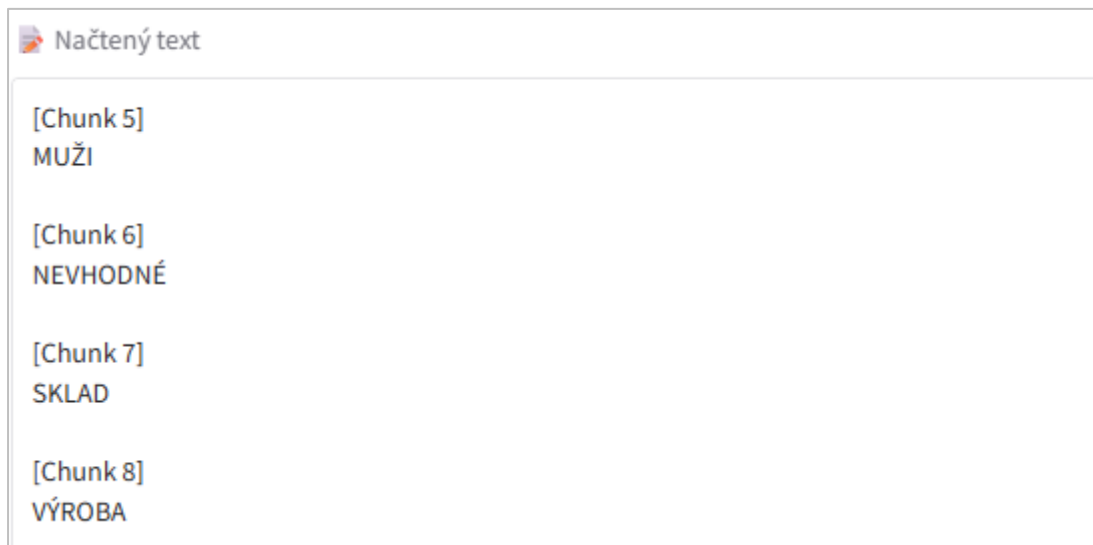
Krátká, vytahaná či děravá trika, nátělníky, mikiny, triko s potiskem, džínové košile, džíny, šortky, bermudy, kalhoty s dírami, domácí obuv, žabky, sportovní obuv či jiná obuv nehodící se ke zvolenému typu oblečení.

SKLAD

- Triko a mikina oranžové barvy s logem společnosti.

Obrázek 27 Textová pole v dokumentu Vizuální Dresscode společnosti
(zdroj: vlastní)

Na obrázku (Obrázek 27) jsou textová pole s texty zdrojového DOCX souboru. Pro srovnání je zobrazen výstup RAG aplikace (Obrázek 28), kde můžeme vidět, jaké informace aplikace skutečně načetla do jednotlivých chunků. Byli načteny pouze nadpisy, ale obsah textových polí chybí.



Obrázek 28 Chunkování podle odstavců u dokumentu Vizuální standard
Dresscode společnosti
(zdroj: vlastní)

Toto omezení pak mohlo mít vliv na výsledky kategorie Vizuální obsah, která se právě dotazovala nad tímto dokumentem.

Další problém se vyskytl u dokumentu Směrnice Standardizace výpočetní techniky, který obsahuje tabulku. Při načítání DOCX dokumentu aplikace RAG tuto tabulku nevidí a informace z tabulky potom chybí při hledání odpovědi.

Potíže se dále vyskytly u odrážkových seznamů DOCX dokumentů. Pokud byla zvolena strategie chunkování Po odstavcích (Obrázek 29), došlo k tomu, že jednotlivé odrážky byly načteny jako samostatné chunky (Obrázek 30). Tím pádem se ztratila návaznost na nadpis (téma), ke kterému odrážky patří. To vedlo ke ztrátě kontextu, protože jazykový model tak neměl dostatek informací pro správnou interpretaci jednotlivých částí původního seznamu.

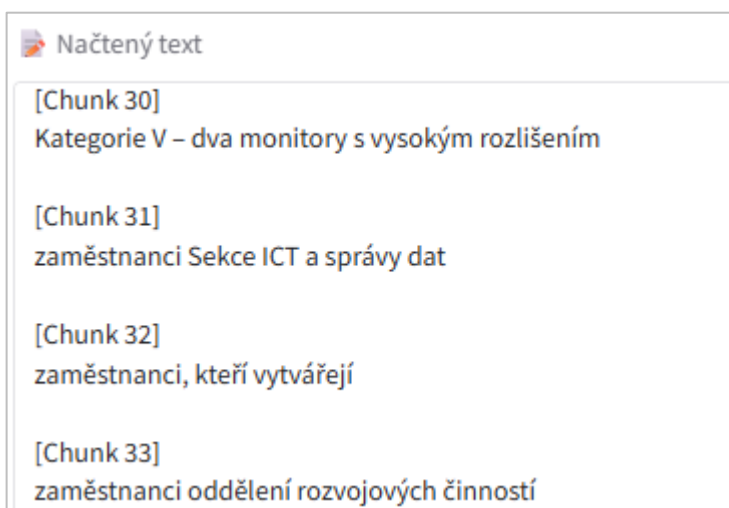
1.6. Kategorie V – dva monitory s vysokým rozlišením

- zaměstnanci Sekce ICT a správy dat
- zaměstnanci, kteří vytvářejí
- zaměstnanci oddělení rozvojových činností

Obrázek 29 Odsazení odrážek ve zdrojovém DOCX dokumentu

(zdroj: vlastní)

Tyto jevy neukazují na chybu aplikace, ale spíše na limity zvolených metod zpracování formátu DOCX, u kterého je vnitřní struktura složitější a méně předvídatelná než u jednodušších formátů.



Obrázek 30 Původní seznam s odrážkami po rozdělení na chunky pomocí strategie Podle odstavců

(zdroj: vlastní)

Pro statistické ověření významnosti těchto rozdílů byla provedena 3-way ANOVA s faktory: kategorie dotazu, formát dokumentu, strategie chunkování, embedding model a jazykový model. Hlavní výsledky jsou shrnuty v tabulce (Tabulka 11). Kategorie dotazu ($p < 0,001$), formát dokumentu ($p < 0,001$) a jazykový model ($p < 0,001$) byly vysoce signifikantní faktor ovlivňující výskyt odpovědí typu „Nevím“. Strategie chunkování sama o sobě nebyla signifikantní, ale projevila se v interakci s jinými faktory. Významné byly mnohé dvou a tří faktorové interakce, například Formát dokumentu * Embedding model * Jazykový model a Kategorie dotazu * Formát dokumentu * Strategie chunkování.

Tabulka 11 Přehled statistické významnosti vlivů jednotlivých faktorů a jejich interakcí na výskyt odpovědi typu „Nevím“

Odpověď typu "Nevím"	Hodnoty: p value									
	Kategorie dotazu*Formát dokumentu*Strategie chunkování	Kategorie dotazu*Strategie chunkování*Embedding model	Kategorie dotazu*Embedding model*Jazykový model	Kategorie dotazu*Strategie chunkování*Jazykový model	Kategorie dotazu*Formát dokumentu*Jazykový model	Kategorie dotazu*Formát dokumentu*Embedding model	Formát dokumentu*Strategie chunkování*Embedding model	Formát dokumentu*Strategie chunkování*Jazykový model	Strategie chunkování*Embedding model*Jazykový model	Formát dokumentu*Embedding model*Jazykový model
Model	<.001	<.001	<.001	<.001	<.001	<.001	<.001	<.001	0,003	<.001
Kategorie dotazu	<.001	<.001	<.001	<.001	<.001	<.001	---	---	---	---
Formát dokumentu	<.001	---	---	---	<.001	<.001	<.001	<.001	---	<.001
Strategie chunkování	0,753	0,794	---	0,752	---	---	0,85	0,844	0,851	---
Embedding model	---	0,012	0,003	---	---	0,003	0,045	---	0,046	0,037
Jazykový model	---	---	<.001	<.001	<.001	---	---	<.001	<.001	<.001
Kategorie dotazu*Formát dokumentu	<.001	---	---	---	<.001	<.001	---	---	---	---
Kategorie dotazu*Strategie chunkování	0,053	0,141	---	0,051	---	---	---	---	---	---
Kategorie dotazu*Embedding model	---	<.001	<.001	---	---	<.001	---	---	---	---
Kategorie dotazu*Jazykový model	---	---	<.001	<.001	<.001	---	---	---	---	---
Formát dokumentu*Strategie chunkování	0,126	---	---	---	---	---	0,387	0,364	---	---
Formát dokumentu*Embedding model	---	---	---	---	---	<.001	0,021	---	---	0,015
Formát dokumentu*Jazykový model	---	---	---	---	0,071	---	---	0,582	---	0,57
Strategie chunkování*Embedding model	---	0,54	---	---	---	---	0,7	---	0,703	---
Strategie chunkování*Jazykový model	---	---	---	0,66	---	---	---	0,874	0,887	---
Embedding model*Jazykový model	---	---	0,965	---	---	---	---	---	0,993	0,992
Kategorie dotazu*Formát dokumentu*Strategie chunkování	<.001	---	---	---	---	---	---	---	---	---
Kategorie dotazu*Strategie chunkování*Embedding model	---	0,76	---	---	---	---	---	---	---	---
Kategorie dotazu*Embedding model*Jazykový model	---	---	1	---	---	---	---	---	---	---
Kategorie dotazu*Strategie chunkování*Jazykový model	---	---	---	0,999	---	---	---	---	---	---
Kategorie dotazu*Formát dokumentu*Jazykový model	---	---	---	---	<.001	---	---	---	---	---
Kategorie dotazu*Formát dokumentu*Embedding model	---	---	---	---	---	0,104	---	---	---	---
Formát dokumentu*Strategie chunkování*Embedding model	---	---	---	---	---	---	0,767	---	---	---
Formát dokumentu*Strategie chunkování*Jazykový model	---	---	---	---	---	---	---	0,994	---	---
Strategie chunkování*Embedding model*Jazykový model	---	---	---	---	---	---	---	---	0,996	---
Formát dokumentu*Embedding model*Jazykový model	---	---	---	---	---	---	---	---	---	0,998

(zdroj: vlastní)

Analýza ukazuje, že odpovědi typu „Nevím“ nejsou náhodným jevem, ale často odrážejí kombinaci podmínek, které ztěžují vyhledávání nebo interpretaci informace. Zejména se potvrdilo, že formát dokumentu DOCX představuje z hlediska automatizovaného zpracování určitá omezení. Problémy jako ignorování obsahu textových polí, oddělení odrážek od nadpisů nebo nenačtení tabulek mají přímý dopad na to, zda systém dokáže odpovědět. Tato zjištění byla statisticky podložena výsledky 3-way ANOVA, kde se formát dokumentu opakovaně ukázal jako významný faktor ovlivňující výskyt odpovědi „Nevím“. Tyto jevy vyplývají z vnitřní struktury a nejednoznačné reprezentace dokumentů ve formátu DOCX. Z pohledu konfigurace RAG aplikace lze doporučit

upřednostnění dokumentů ve formátu PDF nebo vytvářet interní dokumentaci ve strukturovaném formátu markdown, který eliminuje problémy se zpracováním tabulek, textových polí a odrážek. Používat robustnější jazykové modely jako mistral nebo llama3.1:8b. U tabulek a vizuálního obsahu zvážit vhodnou strategii chunkování.

16.2.5 Statistické vyhodnocení kvality odpovědí pomocí metrik F1-score a

Kvalitativní hodnocení

Tabulka 12 Statistická významnost vlivu jednotlivých parametrů RAG aplikace na metriku F1-score

F1-score p value	Kategorie dotazu*Formát dokumentu*Strategie chunkování	Kategorie dotazu*Strategie chunkování*Embedding model	Kategorie dotazu*Embedding model*Jazykový model	Kategorie dotazu*Strategie chunkování*Jazykový model	Kategorie dotazu*Formát dokumentu*Jazykový model	Kategorie dotazu*Formát dokumentu*Embedding model	Formát dokumentu*Strategie chunkování*Embedding model	Formát dokumentu*Strategie chunkování*Jazykový model	Strategie chunkování*Embedding model*Jazykový model	Formát dokumentu*Embedding model*Jazykový model
Model	<.001	<.001	<.001	<.001	<.001	<.001	<.001	<.001	0,633	<.001
Kategorie dotazu	<.001	<.001	<.001	<.001	<.001	<.001	---	---	---	---
Formát dokumentu	<.001	---	---	---	<.001	<.001	<.001	<.001	---	<.001
Strategie chunkování	0,003	0,019	---	0,014	---	---	0,085	0,087	0,102	---
Embedding model	---	0,001	<.001	---	---	<.001	0,009	---	0,013	0,01
Jazykový model	---	---	<.001	<.001	<.001	---	---	0,001	0,002	0,001
Kategorie dotazu*Formát dokumentu	<.001	---	---	---	<.001	<.001	---	---	---	---
Kategorie dotazu*Strategie chunkování	<.001	<.001	---	<.001	---	---	---	---	---	---
Kategorie dotazu*Embedding model	---	0,022	0,013	---	---	0,004	---	---	---	---
Kategorie dotazu*Jazykový model	---	---	<.001	<.001	<.001	---	---	---	---	---
Formát dokumentu*Strategie chunkování	<.001	---	---	---	---	---	0,009	0,01	---	---
Formát dokumentu*Embedding model	---	---	---	---	---	<.001	0,028	---	---	0,03
Formát dokumentu*Jazykový model	---	---	---	---	0,1	---	---	0,599	---	0,601
Strategie chunkování*Embedding model	---	0,858	---	---	---	---	0,936	---	0,944	---
Strategie chunkování*Jazykový model	---	---	---	0,965	---	---	---	0,992	0,993	---
Embedding model*Jazykový model	---	---	0,996	---	---	---	---	---	0,999	0,999
Kategorie dotazu*Formát dokumentu*Strategie chunkování	<.001	---	---	---	---	---	---	---	---	---
Kategorie dotazu*Strategie chunkování*Embedding model	---	0,168	---	---	---	---	---	---	---	---
Kategorie dotazu*Embedding model*Jazykový model	---	---	1	---	---	---	---	---	---	---
Kategorie dotazu*Strategie chunkování*Jazykový model	---	---	---	1	---	---	---	---	---	---
Kategorie dotazu*Formát dokumentu*Jazykový model	---	---	---	---	0,006	---	---	---	---	---
Kategorie dotazu*Formát dokumentu*Embedding model	---	---	---	---	---	0,047	---	---	---	---
Formát dokumentu*Strategie chunkování*Embedding model	---	---	---	---	---	---	0,978	---	---	---
Formát dokumentu*Strategie chunkování*Jazykový model	---	---	---	---	---	---	---	0,968	---	---
Strategie chunkování*Embedding model*Jazykový model	---	---	---	---	---	---	---	---	1	---
Formát dokumentu*Embedding model*Jazykový model	---	---	---	---	---	---	---	---	---	1

(zdroj: vlastní zpracování)

Pro detailní posouzení vlivu jednotlivých parametrů nastavení RAG aplikace a jejich kombinací na výstupní kvalitu odpovědí byly provedeny dvě samostatné analýzy rozptylu

(3-way ANOVA). První metrika F1-score měří přesnost a úplnost odpovědi (Tabulka 12). Druhá metrika Kvalitativní hodnocení reflektuje jazykovou správnost a srozumitelnost (Tabulka 13).

Z tabulek vyplývá, že jak formát dokumentu, tak kategorie dotazu, embedding model a zejména jazykový model mají statisticky významný vliv na obě hodnocené metriky (všechny $p < 0,001$). To potvrzuje, že volba těchto parametrů zásadně ovlivňuje výkon systému.

Tabulka 13 Statistická významnost vlivu jednotlivých parametrů RAG aplikace na metriku Kvalitativní hodnocení

Kvalitativní hodnocení p value	Kategorie dotazu*Formát dokumentu*Strategie chunkování	Kategorie dotazu*Strategie chunkování*Embedding model	Kategorie dotazu*Embedding model*Jazykový model	Kategorie dotazu*Strategie chunkování*Jazykový model	Kategorie dotazu*Formát dokumentu*Jazykový model	Kategorie dotazu*Formát dokumentu*Embedding model	Formát dokumentu*Strategie chunkování*Embedding model	Formát dokumentu*Strategie chunkování*Jazykový model	Strategie chunkování*Embedding model*Jazykový model	Formát dokumentu*Embedding model*Jazykový model
Model	<.001	<.001	<.001	<.001	<.001	<.001	0,002	<.001	0,291	<.001
Kategorie dotazu	<.001	<.001	<.001	<.001	<.001	<.001	---	---	---	---
Formát dokumentu	<.001	---	---	---	<.001	<.001	<.001	<.001	---	<.001
Strategie chunkování	0,009	0,03	---	0,012	---	---	0,127	0,121	0,135	---
Embedding model	---	0,016	0,007	---	---	0,008	0,09	---	0,096	0,087
Jazykový model	---	---	<.001	<.001	<.001	---	---	<.001	<.001	<.001
Kategorie dotazu*Formát dokumentu	<.001	---	---	---	<.001	<.001	---	---	---	---
Kategorie dotazu*Strategie chunkování	<.001	0,01	---	0,001	---	---	---	---	---	---
Kategorie dotazu*Embedding model	---	0,277	0,133	---	---	0,151	---	---	---	---
Kategorie dotazu*Jazykový model	---	---	<.001	<.001	<.001	---	---	---	---	---
Formát dokumentu*Strategie chunkování	0,001	---	---	---	---	---	0,092	0,086	---	---
Formát dokumentu*Embedding model	---	---	---	---	---	0,017	0,2	---	---	0,193
Formát dokumentu*Jazykový model	---	---	---	---	0,018	---	---	0,48	---	0,485
Strategie chunkování*Embedding model	---	0,975	---	---	---	---	0,991	---	0,991	---
Strategie chunkování*Jazykový model	---	---	---	0,782	---	---	---	0,957	0,962	---
Embedding model*Jazykový model	---	---	0,994	---	---	---	---	---	0,999	0,999
Kategorie dotazu*Formát dokumentu*Strategie chunkování	<.001	---	---	---	---	---	---	---	---	---
Kategorie dotazu*Strategie chunkování*Embedding model	---	0,393	---	---	---	---	---	---	---	---
Kategorie dotazu*Embedding model*Jazykový model	---	---	1	---	---	---	---	---	---	---
Kategorie dotazu*Strategie chunkování*Jazykový model	---	---	---	1	---	---	---	---	---	---
Kategorie dotazu*Formát dokumentu*Jazykový model	---	---	---	---	0,001	---	---	---	---	---
Kategorie dotazu*Formát dokumentu*Embedding model	---	---	---	---	---	0,207	---	---	---	---
Formát dokumentu*Strategie chunkování*Embedding model	---	---	---	---	---	---	0,997	---	---	---
Formát dokumentu*Strategie chunkování*Jazykový model	---	---	---	---	---	---	---	0,991	---	---
Strategie chunkování*Embedding model*Jazykový model	---	---	---	---	---	---	---	---	0,998	---
Formát dokumentu*Embedding model*Jazykový model	---	---	---	---	---	---	---	---	---	1

(zdroj: vlastní zpracování)

Zajímavým zjištěním je, že strategie chunkování, ačkoliv se jako samostatný faktor často neprojevuje jako signifikantní, hraje důležitou roli v kombinaci s jinými faktory.

Zejména ve spojení s formátem dokumentu a embedding modelem (např. Formát dokumentu × Strategie chunkování × Embedding model, $p = 0,009$ u F1 score.

Dále lze pozorovat, že i složitější tří faktorové spojení jako např.

- Kategorie dotazu × Formát dokumentu × Strategie chunkování (F1-score: $p < 0,001$; Kvalitativní hodnocení: $p < 0,001$) nebo
- Kategorie dotazu × Formát dokumentu × Jazykový model (F1-score: $p = 0,006$; Kvalitativní hodnocení: $p < 0,001$),

byly u obou metrik statisticky významné, což podtrhuje komplexní charakter RAG systémů, kde výsledná kvalita odpovědi nevzniká jako důsledek jednoho faktoru, ale jako souhra více komponent.

16.2.6 Analýza výkonnosti RAG aplikace podle typu dotazu a konfigurace nastavení

Statistické vyhodnocení předchozí kapitoly ukázalo, že na kvalitu výstupu mají významný vliv faktory jako formát dokumentu, jazykový model či kategorie dotazu. Abychom tato zjištění mohli uplatnit v praktickém návrhu doporučení pro konfiguraci systému, je nezbytné zaměřit se na konkrétní kombinace nastavení a jejich dopad na úspěšnost zodpovězení jednotlivých dotazů.

Cílem této části je proto detailně posoudit, jak si různé kombinace nastavení RAG aplikace vedou při odpovídání na různé typy dotazů. Zatímco v předchozích kapitolách byly hodnoceny dílčí aspekty kvality výstupů (například jazyková kvalita, srozumitelnost nebo výskyt odpovědi typu „Nevím“), zde je kladen důraz na praktickou úspěšnost konkrétních odpovědí z pohledu zadání otázky.

Pro každou kategorii dotazu byla sestavena přehledná tabulka všech kombinací vstupních parametrů, tj. formátu dokumentu, strategie chunkování, embedding modelu a jazykového modelu. Pomocí podmíněného formátování bylo možné vizuálně zvýraznit nejúspěšnější a nejproblematictější konfigurace. Tento způsob zobrazení napomáhá odhalit opakující se vzory a souvislosti, které jsou klíčové pro formulaci doporučeného nastavení systému.

Vzhledem k rozsahu těchto tabulek je v hlavní části práce uvedena pouze souhrnná tabulka výsledků (Tabulka 14) za všechny kategorie dotazů. Kompletní přehledy pro jednotlivé otázky jsou zařazeny do přílohy (P VI). V textu této kapitoly jsou souhrnně prezentována hlavní zjištění a rizikové kombinace, které se opakovaly napříč dotazy.

Hodnocení jednotlivých otázek:

- Přesné vyhledání: Kombinace formátu Markdown, chunkování podle odstavců a embedding modelu all-MiniLM-L6-v2 vedla u všech jazykových modelů ke zcela nulové úspěšnosti. Ostatní embedding modely však na stejnou otázku odpověď nalézt dokázaly, což poukazuje na nevhodnost této konkrétní konfigurace.
- Sémantická podobnost: Jazykový model gemma:7b vykazoval závažné problémy. Ve všech kombinacích rozpoznal pouze obecnou tematiku výběru zařízení, ale nedokázal uvést konkrétní informace. Obtíže s úplností odpovědí se projevily i při použití embedding modelů all-MiniLM-L6-v2 a intfloat/e5-base-v2 ve formátu Markdown se strategií podle odstavců.
- Tabulkové údaje: U formátu DOCX nebyla otázka nikdy úspěšně zodpovězen, což souvisí s technickým omezením při načítání tabulek. Ve formát Markdown zcela selhal embedding model all-MiniLM-L6-v2. U PDF pak všechny embedding modely vykazaly špatné výsledky při použití chunkovací strategie fixní velikost 100/1.
- Výčty: Výsledky byly ve většině případů velmi dobré. Výjimkou byla konfigurace s DOCX dokumenty ve strategii chunkování podle odstavců, kde systém selhal ve všech případech. V ostatních kombinacích se podařilo získat všechna nebo téměř všechna požadovaná fakta.

Tato zjištění potvrzují, že ne všechny konfigurace jsou univerzálně vhodné a že výběr správné kombinace parametrů je klíčový s ohledem na typ úlohy. Podrobné tabulky k jednotlivým otázkám jsou uvedeny v příloze (P VI).

Souhrnné hodnocení:

Souhrnná tabulka (Tabulka 14) zachycuje celkový součet bodů za správné odpovědi napříč všemi otázkami. Umožňuje tak zhodnotit robustnost jednotlivých konfigurací bez ohledu na konkrétní typ dotazu.

Tabulka 14 Celkové srovnání výkonnosti konfigurací RAG aplikace napříč všemi kategoriemi dotazů

Celkové srovnání výkonnosti konfigurací napříč všemi kategoriemi dotazů.									
	Formát dokumentu		Strategie chunkování		Embedding model		Jazykový model	Suma	
Celkové srovnání výkonnosti konfigurací RAG aplikace napříč všemi kategoriemi dotazů.	DOCX	555	Fixní velikost (100/1)	199	all-MiniLM-L6-v2	66	gemma:7b	15	
							llama3	19	
							llama3.1:8b	14	
					intfloat/e5-base-v2	59	mistral	18	
							gemma:7b	12	
							llama3	17	
			Fixní velikost (200/20)	211	TF-IDF	74	llama3.1:8b	15	
							mistral	15	
							gemma:7b	16	
					all-MiniLM-L6-v2	72	llama3	21	
							llama3.1:8b	18	
							mistral	19	
			Podle odstavců	145	intfloat/e5-base-v2	67	gemma:7b	15	
							llama3	15	
							llama3.1:8b	18	
					TF-IDF	72	mistral	19	
							gemma:7b	15	
							llama3	20	
	MD	682	Fixní velikost (100/1)	230	all-MiniLM-L6-v2	69	llama3.1:8b	17	
							mistral	19	
					intfloat/e5-base-v2	75	gemma:7b	16	
							llama3	19	
							llama3.1:8b	21	
			Fixní velikost (200/20)	250	TF-IDF	86	mistral	21	
							gemma:7b	18	
							llama3	21	
			Podle odstavců	202	all-MiniLM-L6-v2	56	llama3.1:8b	24	
							mistral	24	
					intfloat/e5-base-v2	69	gemma:7b	20	
							llama3	22	
							llama3.1:8b	23	
	PDF	739	Fixní velikost (100/1)	234	TF-IDF	80	mistral	26	
							gemma:7b	13	
					all-MiniLM-L6-v2	80	llama3	14	
							llama3.1:8b	14	
			Fixní velikost (200/20)	240	intfloat/e5-base-v2	74	mistral	15	
							gemma:7b	15	
					TF-IDF	80	llama3	19	
							llama3.1:8b	19	
			Podle odstavců	265	all-MiniLM-L6-v2	85	mistral	21	
							gemma:7b	15	
					intfloat/e5-base-v2	92	llama3	21	
							llama3.1:8b	21	
					TF-IDF	88	mistral	23	
							gemma:7b	16	
							llama3	27	
							llama3.1:8b	22	
							mistral	23	

(zdroj: vlastní)

Z výsledků vyplývá:

- Nejvyšší úspěšnosti bylo dosaženo u dokumentů ve formátu PDF, zejména při použití strategie chunkování podle odstavců, embedding modelu intfloat/e5-base-v2 a jednoduchého vyhledávání pomocí TF-IDF.
- Naopak dokumenty ve formátu DOCX vykazovaly obecně nižší výkonnost. Nejnižší souhrnné výsledky byly zaznamenány u konfigurací využívajících strategii chunkování podle odstavců a embedding modelu intfloat/e5-base-v2.
- Nejnižší souhrnné výsledky byly zaznamenány u konfigurací využívajících strategii Fixní velikosti chunku 100/1 s embedding modelem intfloat/e5-base-v2 a all-MiniLM-L6-v2. Tyto kombinace dosahovaly jen kolem 43-59 bodů, což způsobily potíže s extrakcí a reprezentací informací z komplexních DOCX struktur.
- Při srovnání embedding modelů se jako nejvýkonnější jeví kombinace TF-IDF a intfloat/e5-base-v2, které v různých formátech dokumentů stabilně dosahovaly vyšších výsledků než model all-MiniLM-L6-v2.
- Mezi jazykovými modely, nejlepší průměrné skóre napříč všemi konfiguracemi vykazovaly modely mistral a llama3.1:8b.

Tato souhrnná analýza potvrzuje předchozí zjištění, že volba formátu dokumentu a strategie chunkování mají zásadní vliv na úspěšnost RAG aplikace. Zejména se ukazuje výhodnost využívání PDF dokumentů se strategií chunkování podle odstavců, kde lze efektivněji zachovat kontext než komplexní struktury DOCX. Z pohledu embedding modelů je překvapivě úspěšný jednoduchý přístup založený na TF-IDF, který v některých případech překonává složitější neuronové embeddingy.

17 DOPORUČENÍ VHODNÉHO NASTAVENÍ RAG SYSTÉMU

Na základě praktické části diplomové práce, ve které byly testovány různé kombinace vstupních parametrů RAG aplikace, lze formulovat praktická doporučení pro konfiguraci, která vedla k nejlepším výsledkům napříč různými typy dotazů. Cílem této kapitoly je shrnout osvědčené nastavení jednotlivých komponent RAG systému, a tím poskytnout základní vodítko pro budoucí implementace v podnikové praxi.

- **Doporučený formát dokumentů**

PDF byl nejrobustnější formát z hlediska zachování struktury, čitelnosti textu a přesnosti odpovědí.

Markdown je preferovaný formát pro tvorbu nové dokumentace. Je výborně strukturovatelný a umožňuje přesné ovládání chunkování i následný převod do PDF bez ztráty formátu.

DOCX se v testech ukázal jako nejméně vhodný, a to především kvůli komplikované vnitřní struktuře a ztrátám informací při parsování u tabulek a textových polí.

- **Doporučená strategie chunkování**

Nejvyšší konzistenci a relevanci odpovědí přinesla strategie chunkování podle odstavců. Strategie fixní velikosti chunku (100/1 slov) vedla k častému rozpadu významových celků a horším výsledkům u složitějších dotazů.

- **Doporučené embedding modely**

Nejlepší výsledky dosáhl embedding model intfloat/e5-base-v2, který ve většině případů překonal model all-MiniLM-L6-v2, zejména u složitějších dotazů typu Více větné uvažování nebo Sémantická podobnost. I přes svou jednoduchost byl embedding model TF-IDF překvapivě výkonný, a to především u přesného vyhledávání nebo výčtů.

- **Doporučené jazykové modely**

Nejlépe si vedl llama3.1:8b, který vykazoval nízký počet jazykových chyb a vysokou konzistenci napříč kategoriemi dotazů.

Model mistral podával velmi dobré výkony v generování obsahu, ale měl vyšší míru jazykových nepřesností.

Model gemma:7b byl nejméně spolehlivý, zejména u složitějších otázek nebo u dotazů s požadavkem na sémantické porozumění.

Doporučená kombinace pro obecné nasazení RAG aplikace

Pokud firma hledá základní doporučení pro vytvoření funkční RAG aplikace pro dokumentaci běžného provozního charakteru, lze doporučit následující konfiguraci:

- Formát dokumentu: Markdown nebo dobře strukturované PDF s textovou vrstvou
- Strategie chunkování: podle odstavců
- Embedding model: intfloat/e5-base-v2 nebo TF-IDF
- Jazykový model: llama3.1:8b nebo mistral (s kontrolou jazykové přesnosti)

Tato konfigurace dosahovala v rámci experimentu nejvyšší míry pokrytí klíčových informací, nízké míry jazykové chybovosti a vysoké relevance odpovědí v rámci všech kategorií dotazů.

Jednotlivá nastavení RAG aplikace byly otestovány v rámci praktického experimentu, který probíhal výhradně v lokálním prostředí, bez odesílání dat na externí servery. To je klíčové zejména pro nasazení v podnicích, kde je nezbytná ochrana citlivých provozních informací. Výsledky testování potvrdili, že i malé open-source jazykové modely nasazené lokálně dokážou efektivně pracovat s českými texty a poskytovat odpovědi, které jsou sémanticky přesné a často i formálně kvalitní.

Navzdory tomu, že některé z použitých modelů nebyly výslovně trénovány na češtině, prokázaly schopnost porozumět českému textu a správně reagovat na dotazy formulované v češtině. Výsledky tak ukazují, že doporučené nastavení může být prakticky využitelné nejen v testovacím scénáři, ale i v reálném firemním provozu.

18 DOPORUČENÍ PRO PŘÍPRAVU DOKUMENTACE PRO RAG SYSTÉMY

18.1 Praktická doporučení vycházející z testování

Z výsledků provedené analýzy vyplývá, že kvalita odpovědí generovaných pomocí RAG aplikace silně závisí na kvalitě vstupních dokumentů, které tvoří znalostní bázi systému. Na základě získaných poznatků lze formulovat následující praktická doporučení pro strukturu a formátování dokumentace určené pro budoucí strojové zpracování v rámci RAG aplikací.

- **Struktura a významový obsah odstavců**

Každý odstavec by měl představovat ucelenou myšlenku a obsahovat kontextově související informace. Během testování bylo zjištěno, že například v dokumentech formátu DOCX se strategií chunkování podle odstavců vedly krátké nebo neúplné věty k nulové úspěšnosti odpovědi (viz dotaz na výčty kapitola 16.2.6). Důležité informace by neměly být rozčleněny do jednotlivých odrážek bez vazby na nadpis nebo úvodní text.

- **Srozumitelnost jazyka a vysvětlování zkratk**

V dokumentech je vhodné minimalizovat užívání zkratk nebo specifického žargonu. Pokud jsou zkratky nezbytné, měly by být při prvním použití vysvětleny. Například zkratka OPIATI použitá v dotazu na rozpočet zařízení by model správně interpretoval pouze v případě, že by její význam byl v textu jednoznačně vysvětlen.

- **Obrázky, grafika a vizuální prvky**

Během testování se ukázalo, že pokud obrázky s klíčovou informací nebyly doplněny popiskem nebo titulkem, jazykový model nebyl schopen relevantní informaci získat. Obrázky proto musí být doplněny o popis, který shrnuje, co je na obrázku zobrazeno a jaký je význam případných šipek či grafických prvků.

- **Hierarchie nadpisů**

Hierarchie nadpisů by měla být striktně dodržována. Nadpisy vyšších úrovní by měly být používány pro obecnější témata a podnadpisy pro konkrétnější. V diplomové práci bylo prokázáno, že při chunkování podle odstavců je nezbytné zajistit, aby byl nadpis součástí odpovídajícího textového bloku, jinak dojde ke ztrátě kontextu.

- **Formát dokumentu**

Nejvyšší celkové úspěšnosti bylo dosaženo u dokumentů ve formátu PDF, které se ukázaly jako nejrobustnější napříč různými kombinacemi nastavení parametrů. Markdown se osvědčil zejména v testovaných případech svou přehlednou strukturou a možností snadného ovlivnění chunkování, ale v některých kombinacích (například s embeddingovým modelem all-MiniLM-L6-v2) docházelo ke ztrátě kontextu. Formát DOCX vykazoval nejvyšší výskyt chyb typu „Nevím“ a ztrátu informací z tabulek a textových polí, a proto je pro použití v RAG aplikacích nevhodný.

Tato doporučení vycházejí z praktických poznatků získaných během testování reálných dokumentů a ukazují, že zlepšení struktury, jazyka a formátu provozní dokumentace povede k výrazně vyšší kvalitě a konzistenci generovaných odpovědí.

18.2 Markdown jako preferovaný formát pro firemní dokumentaci

V rámci diplomové práce se ukázalo, že formát Markdown představuje ideální kompromis mezi potřebami automatizovaného zpracování dat a požadavky na vizuální prezentaci dokumentů. Markdown umožňuje jednoznačně vyjádřit strukturu dokumentu pomocí značek, což výrazně usnadňuje rozčlenění textu do smysluplných částí během procesu chunkování. Díky tomu je možné zachovat kontext nadpisů, seznamů, tabulek, poznámek, hlaviček a předejít tak ztrátě významových vztahů mezi jednotlivými částmi textu.

Současně je Markdown velmi vhodný i pro finální vydání dokumentace ve formátu PDF a to prostřednictvím přímé aplikace firemní šablony při převodu (například pomocí Pandoc a LaTeX šablon). Tento přístup umožňuje firmám připravit jednu vstupní verzi dokumentu, která je jak optimalizovaná pro potřeby RAG systému, tak použitelná pro publikační účely bez dalšího ručního formátování.

V porovnání s formátem DOCX je výhodou Markdownu i jeho předvídatelné chování při načítání, absence skrytých stylů a možnost přímo ovlivnit strukturu chunků. Při testování se ukázalo, že formát Markdown dosahoval vysoké konzistence výsledků, zejména v kombinaci se strategií chunkování podle odstavců, a při správném návrhu dokumentu poskytoval dostatečný kontext pro generování přesných odpovědí.

Firemní dokumentace často obsahuje informace v hlavičkách a patičkách, jako jsou název oddělení, verze dokumentu, autor nebo datum aktualizace. Ve formátu DOCX jsou tyto informace často implementovány jako textová pole nebo grafické prvky, které se při

čtení ztrácí. V Markdownu lze tyto informace snadno zachovat jako běžný text na začátku nebo konci souboru a tím je zachovat ve vektorovém indexu. Například (Program 1):

```
> Dokument: Standardizace výpočetní techniky
> Autor: Oddělení OPIATI
> Verze: 1.2
> Datum: 2024-04-15
```

Program 1 Hlavička firemního dokumentu ve formátu Markdown

To zajišťuje, že AI model může pracovat s kompletním kontextem dokumentu včetně všech formálních údajů, které mohou být relevantní při vyhledávání nebo filtrování odpovědí.

Markdown poskytuje také možnost vložení vizuální prvků, jako jsou obrázky a schémata pomocí syntaxe (Program 2):

```
![Popis obrázku] (./obrazek1.png)
```

Program 2 Syntaxe formátu Markdown pro vložení obrázku

Důležité je, aby byl popis (alt text) formulován tak, aby vystihoval důležitou informaci z obrázku. Například:

```
![Na obrázku je znázorněno rozmístění serverů v racku. Šipka ukazuje na jednotku s poruchou.] (./server_rack.png)
```

Program 3 Popis obrázku ve formátu Markdown

Takto zapsaný popis obrázku se dostane do indexovaného obsahu a lze jej zahrnout do kontextu odpovědi. V případě, že dokumenty obsahují šipky, grafické symboly nebo barevné značení, je vhodné jejich význam přesně popsat v doprovodném textu.

Zejména pro práci s tabulkovými daty se Markdown ukázal jako výrazně efektivnější formát než DOCX. Běžné metody převádějí tabulky v PDF dokumentech na prostý text, kde je každý řádek tabulky interpretován jako samostatný řádek textu. Tím se ztrácí informace o sloupcích a vztazích mezi daty v jednotlivých buňkách. Pokud se takto zpracovaná data předloží jazykovému modelu, je pro něj velmi obtížné zachytit významové souvislosti, protože chybí kontext daný sloupcem a řádkem.

Markdown naopak přesně definuje strukturu tabulky, a tím zachovává řádkovou i sloupcovou orientaci. Díky tomu je pro model mnohem snazší rozpoznat relace mezi jednotlivými položkami a využít je pro formulaci odpovědi. Tento přístup se ukázal jako

výhodný i při testování RAG aplikace na dokumentech s technickými tabulkami. Příklad zápisu tabulky v Markdown formátu ze směrnice Standardizace výpočetní techniky:

Zařízení	Kategorie zařízení	Obměna po letech
Počítač	Výpočetní technika	5
Monitor	Zobrazovací technika	7
Notebook	Výpočetní technika	5
Ultrabook	Výpočetní technika	4

Program 4 Zápis tabulky ve formátu Markdown

Markdown je flexibilní a robustní formát, který spojuje snadnou úpravu textu s přehlednou strukturou. Díky tomu velmi dobře spolupracuje s RAG systémy. Jeho využití v podnikové dokumentaci může výrazně přispět ke zlepšení vyhledávání, interpretace i vizuální prezentace obsahu, a to bez nutnosti složitého formátování nebo složitých konverzních procesů. Z těchto důvodů lze Markdown doporučit jako preferovaný formát pro budoucí zpracování firemních dokumentů v systémech typu RAG.

18.3 Obecné zásady přípravy dokumentů pro RAG systémy

Na základě poznatků získaných v rámci této diplomové práce byla v samostatné příloze zpracována také obecně formulovaná doporučení, která mohou sloužit jako praktický návod pro tvorbu a údržbu provozní dokumentace s ohledem na budoucí využití technologií typu RAG. Příloha (P VII) obsahuje souhrn doporučení týkajících se formátu souboru, struktury dokumentu, předzpracování textu, práce s obrázky a tabulkami. Cílem je poskytnout firmám vodítko, jak připravovat provozní dokumentaci, tak aby byla snadno analyzovatelná a dlouhodobě udržitelná z pohledu nasazení RAG systémů.

19 MOŽNOSTI HYBRIDNÍHO NEBO CLOUDOVÉHO ŘEŠENÍ

Během vývoje a testování lokální aplikace bylo ověřeno, že přístup k podnikové dokumentaci lze plně realizovat bez nutnosti připojení k internetu a bez využívání cloudových služeb. Tento způsob zajišťuje maximální ochranu firemních dat, neboť veškeré zpracování probíhá výhradně na straně uživatele a žádná data neopouštějí jeho zařízení.

Přestože je lokální přístup bezpečný a samostatný, má určitá omezení vyplývající z výpočetního výkonu. Především větší jazykové modely, multimodální analýza nebo rychlá odpověď na rozsáhlé dotazy může být na běžném počítači velmi náročná nebo neproveditelná.

Z toho důvodu by bylo možné do budoucna uvažovat o rozšíření systému o cloudové řešení, například formou:

- Hybridního režimu – kde uživatel pracuje primárně lokálně, ale má možnost (po schválení) odeslat dotaz nebo část dat do cloudu za účelem lepší odpovědi.
- Bezpečného napojení na firemní cloud – jako je např. Azure AI, AWS Bedrock nebo privátní instance OpenAI GPT, kde je možné data šifrovat, anonymizovat nebo omezit přístupová práva.
- Využití veřejných LLM přes API (např. OpenAI, Gemini, Claude) - v případech, kde se nejedná o citlivá firemní data (např. obecné dotazy, testování).

Tato možnost by uživateli umožnila využívat sílu nejnovějších modelů s vyšší přesností, multimodálními schopnostmi a větším kontextovým oknem, a přitom si ponechat bezpečné prostředí pro práci s citlivými dokumenty.

Z pohledu diplomové práce je však prioritou lokální přístup a ověření možností čistě na lokálním zařízení, aby bylo možné navrhnout takový systém, který by byl okamžitě využitelný i bez potřeby připojení ke cloudu.

20 NÁHRADA CITLIVÝCH DAT SYNTETICKÝMI DOKUMENTY

Původní záměr diplomové práce počítal s využitím reálných dokumentů z firemní vědomostní databáze, jako jsou technologické postupy, provozní návody a další typy interních podkladů. Tyto dokumenty však obsahují citlivé informace, jejichž zpřístupnění by představovalo riziko úniku know-how a porušení interních bezpečnostních zásad podniku.

Z důvodu těchto omezení nebylo možné použít reálná data pro vývoj a testování aplikace. Pro zachování výzkumné relevance a současně dodržení etických a bezpečnostních pravidel bylo přistoupeno k vytvoření syntetických testovacích dokumentů. Tyto dokumenty byly navrženy tak, aby:

- Odpovídaly strukturou i jazykem reálným firemním dokumentům.
- Obsahovaly relevantní text, tabulky a obrázky podobné těm, se kterými by pracovala cílová aplikace.
- Umožňovaly testování dotazů v českém jazyce.
- Sloužily jako referenční základ pro hodnocení přesnosti odpovědí modelu.

Syntetické dokumenty zároveň poskytly bezpečné a kontrolované prostředí pro experimentování s různými strategiemi členění textu (chunking), a pro testování vlivu parametrů jako je velikost chunku nebo překrytí mezi chunky. Kromě toho byly využity pro aplikaci a porovnání hodnotících metrik jako jsou Recall, Precision, F1-score nebo některé aspekty Human evaluation.

Tento přístup nejen umožnil praktické ověření vývoje RAG systému, ale zároveň respektoval bezpečnostní omezení daná firemním prostředím. Vznikl tak modulární testovací rámec, který lze v budoucnu rozšířit o skutečné dokumenty za předpokladu, že budou dostupné v souladu s interními předpisy o ochraně dat.

ZÁVĚR

Firmy dnes hledají nové cesty, jak lépe využít své interní znalosti a zefektivnit práci s provozní dokumentací. Jednou z nadějných technologií je využití systémů Retrieval-Augmented Generation (RAG). Cílem této diplomové práce bylo ověřit, zda lze takový systém provozovat plně lokálně, bez nutnosti odesílat data mimo podnikové prostředí, a zároveň využít dostupné menší jazykové modely. Výsledkem je funkční RAG aplikace, umožňující dotazování nad dokumenty. Aplikace sloužila i jako experimentální nástroj, díky kterému bylo možností najít konfigurace RAG systému, které přinášejí nejlepší výsledky. Výstupem práce jsou praktická doporučení pro strukturování a předzpracování dokumentů, která mohou firmám sloužit jako vodítko při zavádění těchto technologií do praxe.

V teoretické části práce byl podrobně popsán koncept RAG, jeho klíčové komponenty a fáze zpracování, jako je dělení textů na menší části (chunky), generování embeddingů, jejich uložení a vyhledávání pomocí vektorových databází. Dále byly představeny nástroje a knihovny využitelné pro vývoj RAG aplikací.

Praktická část se zaměřila na návrh a implementaci vlastní RAG aplikace, která funguje výhradně v lokálním prostředí a umožňuje uživateli testovat různé konfigurace systému. Dokumenty lze nahrávat ve formátu DOCX, PDF a Markdown, dělit je na chunky podle různých strategií a volit embeddingový a jazykový model pro vytvoření odpovědi.

Pro účely testování bylo nutné vytvořit výpočetně výkonné prostředí. Proto byla sestavena vlastní pracovní stanice, která umožňuje běh open-source jazykových modelů s přibližně 7-8 miliardami parametrů s využitím GPU. K tomu byl použit nástroj Ollama s podporou akcelerace a optimalizovaných formátů GGUF. Tato sestava poskytla dostatečný výkon pro vývoj i testování RAG aplikace bez nutnosti cloudových služeb, které by mohly představovat riziko pro bezpečnost podnikových dat.

Při testování byly použity syntetické dokumenty, které věrně simulovaly strukturu a obsah reálných provozních směrnic a pokynů. Pro každý dokument byl připraven soubor dotazů, referenčních odpovědí a klíčových faktů. Testování probíhalo napříč kombinacemi různých parametrů (formátem dokumentu, strategií chunkování, embedding modelem a jazykovým modelem). Výsledky byly vyhodnocovány pomocí metrik Cosine similarity, Precision, Recall, F1-score a doplněny o ruční subjektivní hodnocení jazykové přesnosti, srozumitelnosti odpovědí a schopnosti přiznat modelu, že „neví“.

Výsledky testování ukázaly, že kombinace formátu PDF nebo Markdown, chunkování po odstavcích, embedding modelu intfloat/e5-base-v2 nebo TF-IDF a jazykového modelu llama3.1:8b vedla k nejlepším výsledkům napříč různými typy dotazů. Markdown se osvědčil jako výborný formát pro budoucí dokumentaci. Umožňuje zachovat významové souvislosti v textu a zároveň slouží jako vhodný podklad pro automatizovaný převod do PDF pomocí firemní šablony. Firmám tak stačí připravit jedinou verzi dokumentu pro RAG systém i finální výstup.

Na základě těchto zjištění byla sepsána sada konkrétních doporučení pro budoucí tvorbu dokumentace vhodné pro využití v RAG systémech. Mezi ně patří například důraz na významové celky v odstavcích, popisování vizuálních prvků, dodržování hierarchie nadpisů či vysvětlování zkratk. Doporučení byla shrnuta v samostatné příloze a mohou sloužit jako praktický návod pro firmy, které chtějí připravit své dokumenty pro budoucí automatizované zpracování.

Diplomová práce tímto přináší praktický důkaz, že moderní AI technologie lze nasadit i v citlivém prostředí bez nutnosti cloudových služeb. Výstupy jsou přenositelné i do dalších podniků, které čelí podobným výzvám. V budoucnu lze očekávat další rozšíření RAG systémů, například o schopnost extrakce dat z grafiky nebo automatické obohacení dokumentů o metainformace. Práce ukazuje, jak lze pomocí RAG systému lépe využívat znalosti uložené ve firemních dokumentech a zároveň poskytuje základ pro vývoj podobných nástrojů, které mohou usnadnit práci s dokumentací i v dalších odvětvích.

SEZNAM POUŽITÉ LITERATURY

- [1] JANSSEN, Thomas. Build a RAG in 10 minutes. In: *YouTube* [online]. 2024, 6.8.2024 [cit. 2025-02-18]. Dostupné z: <https://www.youtube.com/watch?v=JfSmffOyV-8>
- [2] WU, Yingjun a Jonathan BENNION. Developing LLM Applications with LangChain: Which vector database should I use? In: *DataCamp* [online]. [cit. 2025-03-04]. Dostupné z: <https://app.datacamp.com/learn/courses/developing-llm-applications-with-langchain>
- [3] CHOCKALINGAM, Annamalai, Ankur PATEL, Shashank VERMA a Tiffany YEUNG. *A Beginner's Guide to Large Language Models* [online]. In: NVIDIA, 2023, s. 24 [cit. 2025-03-11]. Dostupné z: <https://www.amax.com/content/files/2024/03/llm-ebook-part1-1.pdf>
- [4] CHUGH, Vidhi. DATACAMP. *Large language models (LLMs) concepts* [online]. [cit. 2025-03-15]. Dostupné z: <https://campus.datacamp.com/courses/large-language-models-llms-concepts/introduction-to-large-language-models-llm?ex=9>
- [5] JAVORNICKÁ, Petra. ChatGPT nemusí vědět všechno. Která data rozhodně nesdílet s modely AI? In: O2 CYBERNEWS. *O2cybernews* [online]. 2024, 16.5.2024 [cit. 2025-03-16]. Dostupné z: <https://o2cybernews.cz/clanky/chatgpt-nemusi-vedet-vsechno-ktera-data-rozhodne-nesdilet-s-ai>
- [6] ESET DIGITAL SECURITY GUIDE. Bezpečné nástroje umělé inteligence: Co by měly firmy vědět o nařízení AI Act? *ESET Digital Security Guide* [online]. 2024, 3.10.2024 [cit. 2025-03-16]. Dostupné z: <https://digitalsecurityguide.eset.com/cz/bezpecne-nastroje-umele-inteligence-co-by-mely-firmy-vedet-o-narizeni-ai-act>
- [7] OTEVŘEL, Petr. JANSÁ MOKRÝ OTEVŘEL & PARTNEŘI. Nasazujete ChatGPT ve firmě? A víte, co děláte? *Lawyer.cz* [online]. 2023, 21.4.2023 [cit. 2025-03-16]. Dostupné z: <https://www.lawyer.cz/nasazujete-chatgpt-ve-firme-a-vite-co-delate/>
- [8] CHAWLA, Avi a Akshay PACHAAR. A Crash Course on Building RAG Systems (With Implementation). In: *DailyDoseOfDataScience* [online]. 2025, 25.3.2025 [cit. 2025-03-25]. Dostupné z: <https://www.dailydoseofds.com/a-crash-course-on-building-rag-systems-part-1-with-implementations/>

- [9] IBM TECHNOLOGY. What is a Vector Database? Powering Semantic Search & AI Applications. *YouTube* [online]. 2025, 24.3.2025 [cit. 2025-03-28]. Dostupné z: <https://www.youtube.com/watch?v=g11r1XV0SLw>
- [10] SMITH, Brandon a Anton TROYNIKOV. Evaluating Chunking Strategies for Retrieval. In: *Chroma Technical Report* [online]. 3 July 2024 [cit. 2025-03-29]. Dostupné z: <https://research.trychroma.com/evaluating-chunking>
- [11] DENSER AI. Semantic Search vs Keyword Search: Which is Better? *Denser AI* [online]. 2024, 4.2.2024 [cit. 2025-03-29]. Dostupné z: <https://denser.ai/blog/semantic-search-vs-keyword-search/>
- [12] ESPEJEL, Omar. HUGGING FACE. *Getting Started With Embeddings* [online]. 2022, 23.6.2022 [cit. 2025-03-29]. Dostupné z: <https://huggingface.co/blog/getting-started-with-embeddings>
- [13] IRISAGENT. *Understanding LLM Embeddings: A Comprehensive Guide* [online]. 2024, 17.5.2024 [cit. 2025-03-29]. Dostupné z: <https://irisagent.com/blog/understanding-llm-embeddings-a-comprehensive-guide/>
- [14] MORGAN, Jeffrey. SMALL DATA SF. Build Bigger With Small Ai: Running Small Models Locally. *YouTube* [online]. 2025, 22.3.2025 [cit. 2025-03-29]. Dostupné z: <https://www.youtube.com/watch?v=P-55pV6ss3k&t=600s>
- [15] LABONNE, Maxime. *LLaMA 3 405B closes the gap between closed-source and open-source models* [online]. 2024, 19.4.2024 [cit. 2025-03-29]. Dostupné z: https://www.linkedin.com/posts/maxime-labonne_llama-3-405b-closes-the-gap-between-closed-source-activity-7221774846429384704-aRWK/
- [16] FERNANDEZ, Tomas. SEMAPHORE. *6 Ways For Running A Local LLM* [online]. 2024, 18.6.2024 [cit. 2025-03-30]. Dostupné z: <https://semaphore.io/blog/local-llm>
- [17] DOBOSEVYCH, Oles. GENIUSEE. *Why local LLMs are the future of enterprise AI* [online]. 2025, 2.3.2025 [cit. 2025-03-30]. Dostupné z: <https://geniusee.com/single-blog/local-llm-models>
- [18] METRIC CODERS. *How to Estimate the Cost of Running SaaS-Based vs. Open Source LLM Models* [online]. 2025, 28.3.2025 [cit. 2025-03-30]. Dostupné z: <https://www.metriccoders.com/post/how-to-estimate-the-cost-of-running-saas-based-vs-open-source-llm-models>

- [19] ALZA. *Alza Počítače a notebooky* [online]. c1994-2025 [cit. 2025-04-06]. Dostupné z: <https://www.alza.cz/pocitace-a-notebooky>
- [20] LUCEK, Adam. The best way to chunk text for RAG. *YouTube* [online]. 2024, 9.12.2024 [cit. 2025-04-11]. Dostupné z: <https://www.youtube.com/watch?v=Pk2BeaGbcTE>
- [21] EVIDENTLY AI. *Accuracy vs. precision vs. recall in machine learning: what's the difference?* [online]. 9 January 2025 [cit. 2025-04-11]. Dostupné z: <https://www.evidentlyai.com/classification-metrics/accuracy-precision-recall>
- [22] AMRUTKAR, Prathamesh. The Complete Guide to Object Detection Evaluation Metrics: From IoU to mAP and More. *Medium* [online]. 9 May 2024 [cit. 2025-04-11]. Dostupné z: <https://medium.com/@prathameshamrutkar3/the-complete-guide-to-object-detection-evaluation-metrics-from-iou-to-map-and-more-1a23c0ea3c9d>
- [23] VAN DER LEE, Chris, Albert GATT, Emiel VAN MILTENBURG a Emiel KRAHMER. *Human evaluation of automatically generated text: Current trends and best practice guidelines*. *Computer Speech & Language* [online]. 2020, 67, 101151. ISSN 0885-2308. Dostupné z: <https://doi.org/10.1016/j.csl.2020.101151>
- [24] YANG, Chia Jeng. An introduction to RAG and simple/ complex RAG. *Medium* [online]. 5 December 2023 [cit. 2025-04-13]. Dostupné z: <https://medium.com/biased-algorithms/word-embeddings-vs-sentence-embeddings-a0295be74d4b>
- [25] ZIYUAN, Zhuang, Zhang ZHIYANG, Cheng SITAO, et al. EfficientRAG: Efficient Retriever for Multi-Hop Question Answering. *Medium* [online]. 26 September 2024 [cit. 2025-04-13]. Dostupné z: <https://arxiv.org/pdf/2408.04259>
- [26] TUYCHIEV, Bex. FIRECRAWL. *15 Best Open-Source RAG Frameworks in 2025* [online]. 8 April 2025 [cit. 2025-04-18]. Dostupné z: <https://www.firecrawl.dev/blog/best-open-source-rag-frameworks>
- [27] SIEGLER, Ryan. MEDIUM. *RAG + LlamaParse: Advanced PDF Parsing for Retrieval* [online]. 3 May 2024 [cit. 2025-04-18]. Dostupné z: <https://medium.com/kx-systems/rag-llamaparse-advanced-pdf-parsing-for-retrieval-c393ab29891b>
- [28] D, Neil. ANALYTICS VIDHYA. *15 Chunking Techniques to Build Exceptional RAGs Systems* [online]. 4 April 2025 [cit. 2025-04-19]. Dostupné z:

<https://www.analyticsvidhya.com/blog/2024/10/chunking-techniques-to-build-exceptional-rag-systems/>

- [29] NAYAK, Plaban. MEDIUM. *Semantic Chunking for RAG* [online]. 21 April 2024 [cit. 2025-04-19]. Dostupné z: <https://medium.com/the-ai-forum/semantic-chunking-for-rag-f4733025d5f5>
- [30] QU, Renyi, Forrest BAO a Ruixuan TU. ARXIV. *Is Semantic Chunking Worth the Computational Cost?* [online]. 16 October 2024 [cit. 2025-04-19]. Dostupné z: <https://arxiv.org/html/2410.13070v1#:~:text=Clustering-based%20Semantic%20Chunker%20This%20type%20of%20chunkers%20leverage,global%20relationships%20and%20allowing%20for%20non-sequential%20sentence%20groupings.>
- [31] CHAWLA, Avi a Akshay PACHAAR. DAILYDOSEOFDATASCIENCE. *16 Techniques to Supercharge and Build Real-world RAG Systems—Part 1* [online]. 26 January 2025 [cit. 2025-04-21]. Dostupné z: <https://www.dailydoseofds.com/16-techniques-to-supercharge-and-build-real-world-rag-systems-part-1/#introduction>
- [32] IBM TECHNOLOGY. What are Word Embeddings? Powering Semantic Search & AI Applications. *YouTube* [online]. 2025, 20.8.2024 [cit. 2025-04-21]. Dostupné z: <https://www.youtube.com/watch?v=wgfSDrqYMJ4>
- [33] MIKOLOV, Tomas, Kai CHEN, Greg CORRADO a Jeffrey DEAN. GOOGLE. *Efficient Estimation of Word Representations in Vector Space* [online]. 7 September 2013 [cit. 2025-04-21]. Dostupné z: <https://arxiv.org/pdf/1301.3781>
- [34] ARORA, Simran, Avner MAY, Jian ZHANG a Christopher RÉ. ARXIV. *Contextual Embeddings: When Are They Worth It?* [online]. 18 May 2020 [cit. 2025-04-21]. Dostupné z: <https://arxiv.org/pdf/2005.09117>
- [35] MASETHE, Mosima Anna, Hlaudi Daniel MASETHE a Sunday O. OJO. MDPI. *Context-Aware Embedding Techniques for Addressing Meaning Conflation Deficiency in Morphologically Rich Languages Word Embeddings: A systematic Review and Meta Analysis* [online]. 17 October 2024 [cit. 2025-04-21]. Dostupné z: <https://www.mdpi.com/2073-431X/13/10/271>

- [36] MISTRAL AI. *Models Overview* [online]. c2025 [cit. 2025-05-16]. Dostupné z: https://docs.mistral.ai/getting-started/models/models_overview/#:~:text=Mistral%207B%E2%9C%94%E2%9C%94%EF%B8%8F%20Apache%E2%9C%94%E2%9C%94%EF%B8%8F%20first%20dense,latest
- [37] HUGGING FACE. *Access Gemma on Hugging Face* [online]. [cit. 2025-05-16]. Dostupné z: <https://huggingface.co/google/gemma-7b#:~:text=Context%20Length>
- [38] AWS. *Announcing Llama 3.1 405B, 70B, and 8B models from Meta in Amazon Bedrock* [online]. c2024 [cit. 2025-05-16]. Dostupné z: <https://aws.amazon.com/blogs/aws/announcing-llama-3-1-405b-70b-and-8b-models-from-meta-in-amazon-bedrock/#:~:text=All%20Llama%203,Portuguese%2C%20Hindi%2C%20Spanish%2C%20and%20Thai>
- [39] HUGGING FACE. *Meta-Llama-3-8B* [online]. [cit. 2025-05-16]. Dostupné z: <https://huggingface.co/meta-llama/Meta-Llama-3-8B#:~:text=Training%20Data%20Params%20Context%20length,Yes%20%20December%2C%202023>
- [40] OLLAMA. *Https://ollama.com/search* [online]. c2025 [cit. 2025-05-27]. Dostupné z: <https://ollama.com/search>
- [41] OPENAI. *ChatGPT* [online]. San Francisco: OpenAI, 2025 [cit. 2025-03-30]. Dostupné z: <https://chat.openai.com/>

SEZNAM OBRÁZKŮ

Obrázek 1 Srovnání výkonu otevřených a uzavřených jazykových modelů.	21
Obrázek 2 Základní schéma RAG flow	24
Obrázek 3 Architektura Efficient RAG	27
Obrázek 4 Překrývání tokenů u Fixed-size chunking strategie.....	29
Obrázek 5 Recursive chunking	29
Obrázek 6 Semantic chunking	30
Obrázek 7 CBOW a Skip-gram architektury	32
Obrázek 8 Vektorové databáze	44
Obrázek 9 Příklad BOZP návodky.....	50
Obrázek 10 Pracovní postup doplněný	51
Obrázek 11 Aplikace RAG – vložení zdrojového dokumentu, volba strategie chunkování a embedding modelu, uložení indexů	59
Obrázek 12 Aplikace RAG – Zobrazení načteného dokumentu rozděleného na chunky....	60
Obrázek 13 Aplikace RAG – Vložení dotazu, referenční odpovědi a volba LLM.....	63
Obrázek 14 Aplikace RAG – Výstup modelu, počet tokenů, trvání výpočtu a použitý kontext	63
Obrázek 15 Aplikace RAG – Testovací režim pro porovnání různých kombinací systému	64
Obrázek 16 Prompt modelu a systémová instrukce	66
Obrázek 17 Průměrné pokrytí klíčových faktorů v odpovědích	78
Obrázek 18 Porovnání jazykových modelů z hlediska	79
Obrázek 19 Porovnání jazykových modelů z hlediska jazykové kvality.....	80
Obrázek 20 Korelační diagram pro porovnání vztahů F1-score	84
Obrázek 21 Porovnání délky odpovědi podle jazykového modelu.....	86
Obrázek 22 Porovnání jazykové kvality dle jazykových modelů.....	87
Obrázek 23 Porovnání jazykové kvality podle kategorie dotazu.....	88
Obrázek 24 Počet odpovědí typu „Nevím“ podle kategorie dotazu	89
Obrázek 25 Počet odpovědí typu „Nevím“ podle formátu dokumentu	90
Obrázek 26 Počet odpovědí typu „Nevím“ podle jazykového modelu	91
Obrázek 27 Textová pole v dokumentu Vizualní Dresscode společnosti.....	91
Obrázek 28 Chunkování podle odstavců u dokumentu Vizualní standard Dresscode společnosti.....	92
Obrázek 29 Odsazení odrážek ve zdrojovém DOCX dokumentu	93
Obrázek 30 Původní seznam s odrážkami po rozdělení	93

SEZNAM TABULEK

Tabulka 1 Příklad – sémantické vyhledávání pomocí embeddingů v RAG	33
Tabulka 2 Evaluační metriky	35
Tabulka 3 Použitý hardware	53
Tabulka 4 Jazykové modely použité v aplikaci	62
Tabulka 5 Korelační matice metrik hodnocení odpovědí	80
Tabulka 6 Souhrnné p-hodnoty faktorů a jejich interakcí	81
Tabulka 7 Přehled výskytu klíčových informací v odpovědích	83
Tabulka 8 Souhrnná korelační matice metrik hodnocení odpovědí	85
Tabulka 9 Porovnání F1-score a Cosine to Gold pro různé odpovědi	85
Tabulka 10 Ukázka halucinací jazykových modelů při interpretaci zkratky OPIATI	88
Tabulka 11 Přehled statistické významnosti vlivů jednotlivých faktorů a jejich interakcí na výskyt odpovědi typu „Nevím“	94
Tabulka 12 Statistická významnost vlivu jednotlivých parametrů RAG aplikace na metriku F1-score	95
Tabulka 13 Statistická významnost vlivu jednotlivých parametrů RAG aplikace na metriku Kvalitativní hodnocení	96
Tabulka 14 Celkové srovnání výkonnosti konfigurací	99

SEZNAM PROGRAMŮ

Program 1 Hlavička firemního dokumentu ve formátu Markdown.....	105
Program 2 Syntaxe formátu Markdown pro vložení obrázku.....	105
Program 3 Popis obrázku ve formátu Markdown	105
Program 4 Zápis tabulky ve formátu Markdown	106

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

CSV	Comma-Separated-Values
DOCX	Document Open XML
E5	Označení embedding modelu intfloat/e5-base-v2
GPU	Graphics Processing Unit
ICT	Informační a komunikační technika
LLM	Large Language Model – Velký jazykový model
MD	Markdown
MiniLM	Označení embedding modelu all-MiniLM-L6-v2
MMLU	Massive Multitask Language Understanding
NLP	Natural language processing
OCR	Optical Character Recognition
OPIATI	Odbor provozu ICT a technické infrastruktury
PDF	Portable Document Format
RAG	Retrieval Augmented Generation
SML	Small Language Model – Malý jazykový model
TF-IDF	Term Frequency-Inverse Document Frequency
UI	User Interface – uživatelské rozhraní
VRAM	Video Random Access Memory
XML	eXtensible Markup Language

SEZNAM PŘÍLOH

- Příloha P I: Prohlášení o využití nástrojů umělé inteligence při zpracování diplomové práce
- Příloha P II: Zpráva Digitalizace dávek nemocenského pojištění
- Příloha P III: Směrnice Standardizace výpočetní techniky
- Příloha P IV: Pracovní postup provedení tahové zkoušky
- Příloha P V: Vizuální standard Dresscode společnosti
- Příloha P VI: Vyhodnocení úspěšnosti konfigurací RAG systémů pro jednotlivé kategorie dotazů
- Příloha P VII: Obecné zásady přípravy dokumentu pro RAG aplikace

PŘÍLOHA P I: PROHLÁŠENÍ O VYUŽITÍ NÁSTROJŮ UMĚLÉ INTELIGENCE PŘI ZPRACOVÁNÍ DIPLOMOVÉ PRÁCE

Při zpracování této diplomové práce byl využit jazykový model ChatGPT 4o jako nástroj pro technické konzultace, návrhy formulací a kontrolu struktury textu. Výstupy generované tímto nástrojem byly vždy upraveny, ověřeny a zasazeny do vlastního kontextu, přičemž autorka práce plně rozumí jejich významu a využití.

Žádná část této práce nebyla zcela automaticky převzata bez vědomí a pochopení obsahu. AI nástroje byly použity výhradně jako forma digitální nápovědy podobně jako jazykové korektury, doporučení stylistických úprav nebo návrhy textové struktury.

Tato práce je autorským dílem Bc. Ing. Veroniky Šošolíkové, která nese plnou odpovědnost za její obsah.

PŘÍLOHA P II: INFORMAČNÍ ZPRÁVA O DIGITALIZACI DÁVEK NEMOCENSKÉHO POJIŠTĚNÍ

Dobrý den, kolegyně a kolegové,

dovolte mi, abych Vás informovala, že po digitalizaci nemocenské spustila ČSSZ od **1.1.2025** digitalizaci i ostatních dávek nemocenského pojištění.

Konkrétně jde o:

- Ošetřovné
- Dlouhodobé ošetřovné
- Peněžitou pomoc v mateřství
- Otcovskou
- Vyrovnávací příspěvek v těhotenství a mateřství

Dávky fungují podobně jako eNeschopenka, čímž by se měl celý proces zrychlit a zefektivnit.

Níže je jednoduchý manuál, jak postupovat v případě nemoci (pro zopakování) a nově v případě ostatních dávek:

- **E-neschopenka (eNeschopenka)**
- Lékař vystaví Rozhodnutí o dočasné pracovní neschopnosti, které pošle elektronicky na příslušnou OSSZ
- Zaměstnavatel obdrží elektronicky notifikaci o vzniku/změně/ukončení DPN svých zaměstnanců
- Petra Hrochova na základě informace z portálu ČSSZ doplní údaje do docházky (Watt)
- **Zaměstnanec musí neprodleně informovat svého zaměstnavatele, že je nemocný a to tak, že kontaktuje přímého nadřízeného e-mailem nebo telefonicky. Nic dalšího není třeba dělat.**
- **Ošetřovné (OČR)**
- Lékař vystaví Rozhodnutí/Potvrzení elektronicky
- **Zaměstnanec sdělí lékaři svoji e-mailovou adresu, na kterou mu lékař zašle zprávu, která obsahuje identifikátor (tj. číslo rozhodnutí/potvrzení), informaci o druhu dávky, datum jejího vzniku, odkaz na ePortál ČSSZ pro zjištění podrobných údajů a pokyn, aby tyto informace předal zaměstnavateli**
- **Zaměstnanec průkazně přepoše celý tento e-mail na: petra.hrochova@dasta.com a doplní své osobní číslo, jméno a příjmení. Musí jít o prokazatelný způsob oznámení, protože tímto způsobem**

uplatňujete nárok na dávku. Jiná forma předání nebude akceptována (SMS apod.)

- Otcovská, Peněžitá pomoc v mateřství, Dlouhodobé ošetřovné
- Lékař vystaví Rozhodnutí/Potvrzení elektronicky
- Zaměstnanec vyplní formulář (viz příloha) a jeho zasláním na petra.hrochova@dasta.com informuje zaměstnavatele o vzniku příslušné sociální události
- Formuláře najdete v příloze tohoto e-mailu nebo na stránkách ČSSZ: [Úvod - eDávky - Česká správa sociálního zabezpečení](#)
- Do budoucna by mělo zpracovávání formulářů probíhat pomocí modulu Target, o případných změnách Vám dám včas vědět

(zdroj: interní materiály firmy)

PŘÍLOHA P III: SMĚRNICE STANDARDIZACE VÝPOČETNÍ TECHNIKY

SMĚRNICE – Standardizace výpočetní techniky

Základní ustanovení

Účelem tohoto dokumentu je stanovení standardizace přidělování a obměny ICT (Informační a komunikační technika) zaměstnancům tak, aby veškerá ICT byla obměňována v automatickém cyklu a aby nově pořizovaná ICT splňovala výkonnostní a bezpečnostní požadavky, které vycházejí z celkové architektury ICT a provozovaných informačních systémů.

Tento dokument má dále za úkol:

- specifikovat vzorový příklad vybavení ústředí a poboček;
- kategorizovat zaměstnance podle pracovního zařazení, druhu používaného programového vybavení, specifických priorit a přiřadit těmto kategoriím odpovídající typickou variantu HW/SW konfiguraci;
- stanovit minimální technologické a softwarové standardy pro nabývání;
- určit postupy pro pořizování ICT.

Tato směrnice bude sloužit jako podklad pro sestavování rozpočtů a pro přípravu podkladů pro VZ. Standardizací ICT se zamezí nehospodárnému pořizování výpočetní techniky.

1. Kategorie zaměstnanců

1.1. ICT uživatelů

Všem uživatelům ICT je přidělována ICT kategorie určené podle pracovního zařazení, podle druhu používaného programového vybavení a specifických priorit. Zaměstnancům je zásadně přidělována pouze jedna standardní koncová stanice (PC nebo notebook), jinak tomu je pouze u notebooků určených k zápůjčkám. Výjimku o přidělení ICT mimo kategorie uděluje na základě žádosti vedoucího organizačního útvaru/jednotky ředitel Sekce ICT a správy dat. Koncová stanice je přidělována pouze ve standardních HW a SW konfiguracích. Veškeré odůvodněné požadavky na případné nadstandardní HW a SW vybavení uplatňují uživatelé cestou ServiceDesku se souhlasným průvodním stanoviskem přímého nadřízeného. Životnost ICT je stanovena v tabulce obměna techniky dle stáří, ICT je obměňována pouze v případě nerentability opravy a v případech, kdy není zabezpečena řádná funkce užívaných aplikací.

1.2. Kategorie I – ultrabook

- ústřední ředitel
- ředitelé Sekcí
- ředitelé Odborů
- ředitelé poboček
- vedoucí samostatných oddělení na ústředí

1.3. Kategorie II – notebook pro práci s GIS

- zaměstnanci oddělení GIS
- zaměstnanci oddělení správy dat
- zaměstnanci oddělení rozvojových činností

1.4. Kategorie III – notebook pro běžnou kancelářskou práci

- vedoucí pobočky
- zaměstnanci ústředí, u nichž potřeba užívat notebook vyplývá z náplně práce
- vedoucí pobočky (1x notebook nad počet zaměstnanců – k zapůjčování zaměstnancům PO na prezentace, práce v terénu, soutěže, náhrada za nefunkční stanici do doby výměny atd.)

1.5. Kategorie IV – počítač (PC) pro běžnou kancelářskou práci

- zaměstnanci (kteří nepatří do výše uvedených kategorií)

- externí zaměstnanci (ostraha objektu ústředí, údržba ústředí, agenturní zaměstnanci, ...)
- výbava počítačové učebny na ústředí

1.6. Kategorie V – dva monitory s vysokým rozlišením

- zaměstnanci Sekce ICT a správy dat
- zaměstnanci, kteří vytvářejí
- zaměstnanci oddělení rozvojových činností
- zaměstnanci oddělení rozpočtu
- zaměstnanci oddělení majetkoprávních vztahů a rezervy
- ostatní zaměstnanci si mohou připojit jako druhý monitor menší monitor než 24" (starší monitory)

2. Vydávání, odebrání a obměna ICT

Vydávání, odebrání a obměna ICT se řídí dle Metodiky poskytování služeb.

Obměna ICT vybavení uživatelů je nastavena pro každé jednotlivé zařízení v následující tabulce:

Zařízení	Kategorie zařízení	obměna po letech
Počítač	Výpočetní technika	5
Monitor	Zobrazovací technika	7
Notebook	Výpočetní technika	5
Ultrabook	Výpočetní technika	4
Server	Výpočetní technika - servery	5
Diskové pole	Datové úložiště	5
Projektor	Prezentační technika	dle stavu lampy, maximálně 10 let
Plotr	Tisková technika - specializovaná	dle rentability oprav, maximálně 13 let
Tiskárna na obálky	Tisková technika	5
Tiskárna na štítky	Tisková technika	6
Multifunkční kopírovací stroj	Tisková technika	6
Tiskárna	Tisková technika	6
Flash disk	Přenosné médium	5
USB disk	Přenosné médium	6
UPS	Napájecí technika	po 4 letech výměna baterií, po 8 letech výměna UPS
Aktivní síťové prvky	Síťová technika	6

obměna techniky dle stáří

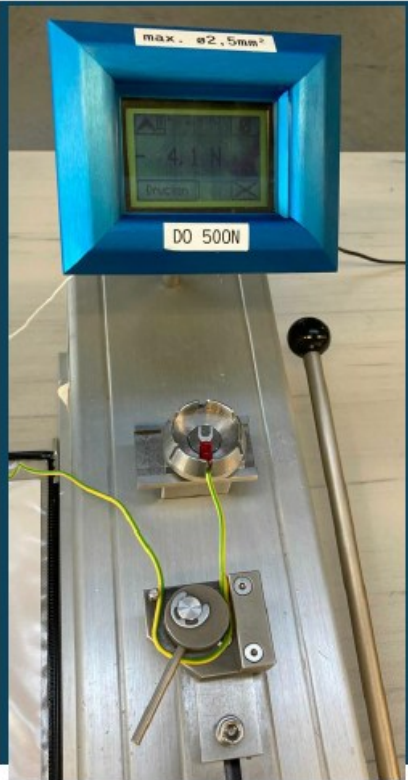
Pravidelná obměna ICT zaručuje optimální efektivitu práce a úsporu vynakládaných prostředků na servis, což v konečném důsledku znamená snížení nákladů na ICT obecně.

Na rok, kdy dovrší technika „morálního opotřebení“, připravuje OPIATI rozpočet na obměnu ICT a plánuje harmonogram obměny.

Vlastní obměna ICT u uživatelů následně probíhá podle tohoto harmonogramu.

(zdroj: interní materiály firmy)

PŘÍLOHA P IV: PRACOVNÍ POSTUP NA PROVEDENÍ TAHOVÉ ZKOUŠKY

8-215301	
TAHOVÁ ZKOUŠKA - návod na obsluhu tahového zařízení MAV Prüftechnik	
1. Vyberte správné zařízení dle průměru drátu a max.tahové síly - uvedeno na zařízení. 2. Ruční páka zařízení musí být před začátkem testu ve vodorovné poloze. Obr.1 U zařízení s motorovým pohonem, musí být zařízení uvedeno do počáteční pozice - provádí se pomocí membránové klávesnice na zařízení. Obr.3 (jedna šipka nahoru) 3. Před upnutím materiálu do zařízení je nutné vynulovat dotykový displej DOT - Matrix. Na dotykový displej je zakázáno příliš tlačít a používat nehty! Obr.2 4. Upevníme zkušební materiál do zařízení - nakrmpovaný kontakt se vloží do kruhové části a vodič upneme do fixačního bodu na zařízení. Obr.1 5. Provedeme tahovou destruktivní zkoušku vloženého materiálu, použitím ruční páky nebo tlačítkem "Spuštění" na membránové klávesnici. Obr.3 (jedna šipka směr dolů) 6. Ze zařízení odebereme veškeré zbytky materiálu a zařízení uvedeme do původní polohy.	
Obr.1 	Obr.2 
	Obr.3 
Vypracoval: KLK, dokument řízen elektronicky v DMS	Aktualizace: 1.10.2022

(zdroj: interní materiály firmy)

PŘÍLOHA P V: VIZUÁLNÍ STANDARD DRESS CODE SPOLEČNOSTI

DRESSCODE LAPP CZECH REPUBLIC S.R.O.



ŽENY

"ŠATY DĚLAJÍ ČLOVĚKA"

- Sukně - těsně nad kolena nebo delší
- Kalhoty - klasického elegantního střihu, tříčtvrteční nebo krátké v délce minimálně těsně nad kolena, kalhoty riflového střihu (nikoli ale z modré džínoviny).
- Klasické košilové blůzy, elegantní trička s krátkými nebo dlouhými rukávy.
- Šaty - krátký nebo dlouhý rukáv, délka těsně nad kolena nebo delší
- Boty - baleríny, klasická dámská obuv na tenké podrážce, lodičky, letní pásková obuv, boty s otevřenou špičkou nebo patou, kozačky, městské tenisky.
- Nalíčení a účes - jednoduchý a střízlivý bez extravagancí.
- Pro jednání s obchodními partnery je vhodné přizpůsobit oblečení protistraně.



NEVHODNĚ

Krátké sukně a minisukně, šortky, plážové šaty, tenká ramínka u triček, příliš vysoké rozparky, džínové sukně a košile, vytahaná nebo krátká trika nad pas a mini trička, která odhalují břicho. Džíny, kalhoty s dírami či, průsvitné oblečení. Sportovní či plážová obuv, domácí obuv, žabky, pantofle či jiná obuv nehodící se ke zvolenému typu oblečení.

1.11.2021

MUŽI

- Kalhoty - jednoduchý styl a barevnost.
- Košile dlouhý či krátký rukáv, polotriko, popř. triko bez potisku.
- Stylově vhodná obuv - polobotky, městské tenisky.
- Pro jednání s obchodními partnery je vhodné přizpůsobit oblečení protistraně.



NEVHODNÉ

Krátká, vytahaná či děravá trika, nátělníky, mikiny, triko s potiskem, džínové košile, džiny, šortky, bermudy, kalhoty s dírami, domácí obuv, žabky, sportovní obuv či jiná obuv nehodící se ke zvolenému typu oblečení.

SKLAD

- Triko a mikina oranžové barvy s logem společnosti.



1.11.2021

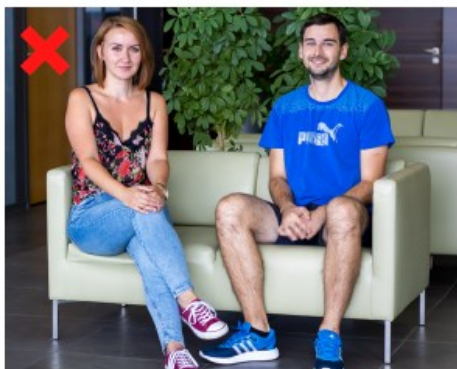
VÝROBA

- Triko šedé barvy s logem společnosti, příp. jiné barvy pro specializované pozice.
- Vesta oranžové nebo černé barvy s logem společnosti.



NEFORMÁLNÍ PÁTEK

- Džíny, džínová bunda, triko s potiskem, méně formální šaty, obuv na volný čas.
- Neformální oblečení musí být v dobrém stavu, celkový dojem musí zůstat upravený a barevně sladěný.



NEVHODNÉ

Roztrhané kalhoty, vyzývavé šaty, špinavé, potřhané triko, kraťasy, triko na špagetových ramínkách s výstřihem, sportovní oblečení, špinavá obuv.

1.11.2021

(zdroj: interní materiály firmy)

PŘÍLOHA P VI: VYHODNOCENÍ ÚSPĚŠNOSTI KONFIGURACÍ RAG SYSTÉMU PRO JEDNOTLIVÉ KATEGORIE DOTAZŮ

Kategorie dotazu: Přesné vyhledání								
Kategorie dotazu	Formát dokumentu	Strategie chunkování		Embedding model		Jazykový model	Suma	
Přesné vyhledání	DOCX	110	Fixní velikost (100/1)	27	all-MiniLM-L6-v2	10	gemma:7b	3
							llama3	3
							llama3.1:8b	1
			mistral	3				
				gemma:7b	0			
				llama3	2			
		llama3.1:8b	0					
			mistral	1				
				gemma:7b	3			
		llama3		4				
		llama3.1:8b	4					
		mistral	3					
	Fixní velikost (200/20)	39	all-MiniLM-L6-v2	15	gemma:7b	4		
					llama3	4		
					llama3.1:8b	4		
		mistral	3					
			gemma:7b	3				
			llama3	0				
	llama3.1:8b	4						
		mistral	4					
			gemma:7b	3				
	llama3		4					
	llama3.1:8b	4						
	mistral	3						
	Podle odstavců	44	all-MiniLM-L6-v2	15	gemma:7b	3		
					llama3	4		
					llama3.1:8b	4		
mistral		4						
		gemma:7b	3					
		llama3	4					
llama3.1:8b	4							
mistral	3							
MD	110	Fixní velikost (100/1)	43	all-MiniLM-L6-v2	15	gemma:7b	3	
						llama3	4	
						llama3.1:8b	4	
				mistral	4			
					gemma:7b	3		
					llama3	4		
		llama3.1:8b	4					
			mistral	3				
				gemma:7b	3			
		llama3		4				
		llama3.1:8b	4					
		mistral	3					
	Fixní velikost (200/20)	45	all-MiniLM-L6-v2	14	gemma:7b	3		
					llama3	4		
					llama3.1:8b	4		
mistral		3						
		gemma:7b	3					
		llama3	4					
llama3.1:8b	4							
mistral	3							
Podle odstavců	22	all-MiniLM-L6-v2	0	gemma:7b	0			
				llama3	0			
				llama3.1:8b	0			
mistral	0							
	gemma:7b	3						
	llama3	4						
llama3.1:8b	4							
mistral	4							
PDF	107	Fixní velikost (100/1)	34	all-MiniLM-L6-v2	12	gemma:7b	3	
						llama3	3	
						llama3.1:8b	3	
				mistral	3			
					gemma:7b	4		
					llama3	0		
		llama3.1:8b	2					
			mistral	3				
				gemma:7b	3			
		llama3		4				
		llama3.1:8b	3					
		mistral	3					
Fixní velikost (200/20)	36	all-MiniLM-L6-v2	12	gemma:7b	3			
				llama3	4			
				llama3.1:8b	4			
	mistral	1						
		gemma:7b	2					
		llama3	4					
llama3.1:8b	2							
	mistral	3						
		gemma:7b	3					
llama3		3						
llama3.1:8b	3							
mistral	3							
Podle odstavců	37	all-MiniLM-L6-v2	11	gemma:7b	3			
				llama3	4			
				llama3.1:8b	2			
	mistral	2						
		gemma:7b	4					
		llama3	4					
llama3.1:8b	4							
mistral	2							
gemma:7b	3							
llama3	4							
llama3.1:8b	3							
mistral	2							

Kategorie dotazu: Sémantická podobnost									
Kategorie dotazu	Formát dokumentu		Strategie chunkování		Embedding model		Jazykový model	Počet bodů	
Sémantická podobnost	DOCX	116	Fixní velikost (100/1)	38	all-MiniLM-L6-v2	12	gemma:7b llama3 llama3.1:8b mistral	1 4 3 4	
					intfloat/e5-base-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
					TF-IDF	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
			Fixní velikost (200/20)	39	all-MiniLM-L6-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
					intfloat/e5-base-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
					TF-IDF	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
			Podle odstavců	39	all-MiniLM-L6-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
					intfloat/e5-base-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
					TF-IDF	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
		MD	99	Fixní velikost (100/1)	39	all-MiniLM-L6-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4
						intfloat/e5-base-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4
						TF-IDF	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4
	Fixní velikost (200/20)			41	all-MiniLM-L6-v2	12	gemma:7b llama3 llama3.1:8b mistral	1 4 3 4	
					intfloat/e5-base-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
					TF-IDF	16	gemma:7b llama3 llama3.1:8b mistral	4 4 4 4	
	Podle odstavců			19	all-MiniLM-L6-v2	3	gemma:7b llama3 llama3.1:8b mistral	1 1 0 1	
					intfloat/e5-base-v2	4	gemma:7b llama3 llama3.1:8b mistral	1 1 1 1	
					TF-IDF	12	gemma:7b llama3 llama3.1:8b mistral	1 3 4 4	
	PDF	110	Fixní velikost (100/1)	39	all-MiniLM-L6-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
					intfloat/e5-base-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
					TF-IDF	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
			Fixní velikost (200/20)	33	all-MiniLM-L6-v2	10	gemma:7b llama3 llama3.1:8b mistral	1 1 4 4	
					intfloat/e5-base-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
					TF-IDF	10	gemma:7b llama3 llama3.1:8b mistral	1 1 4 4	
			Podle odstavců	38	all-MiniLM-L6-v2	12	gemma:7b llama3 llama3.1:8b mistral	0 4 4 4	
					intfloat/e5-base-v2	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	
					TF-IDF	13	gemma:7b llama3 llama3.1:8b mistral	1 4 4 4	

Kategorie dotazu: Tabulkové údaje									
Kategorie dotazu	Formát dokumentu		Strategie chunkování		Embedding model		Jazykový model	Počet bodů	
Tabulkové údaje	DOCX	0	Fixní velikost (100/1)	0	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
					intfloat/e5-base-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
					TF-IDF	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
			Fixní velikost (200/20)	0	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
					intfloat/e5-base-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
					TF-IDF	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
		Podle odstavců	0	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0		
				intfloat/e5-base-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0		
				TF-IDF	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0		
		MD	28	Fixní velikost (100/1)	10	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0
						intfloat/e5-base-v2	5	gemma:7b llama3 llama3.1:8b mistral	2 1 1 1
						TF-IDF	5	gemma:7b llama3 llama3.1:8b mistral	2 1 1 1
	Fixní velikost (200/20)			9	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
					intfloat/e5-base-v2	4	gemma:7b llama3 llama3.1:8b mistral	0 2 1 1	
					TF-IDF	5	gemma:7b llama3 llama3.1:8b mistral	2 1 1 1	
	Podle odstavců		9	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0		
				intfloat/e5-base-v2	5	gemma:7b llama3 llama3.1:8b mistral	2 1 1 1		
				TF-IDF	4	gemma:7b llama3 llama3.1:8b mistral	2 0 1 1		
	PDF		24	Fixní velikost (100/1)	0	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0
						intfloat/e5-base-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0
						TF-IDF	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0
		Fixní velikost (200/20)		10	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
					intfloat/e5-base-v2	4	gemma:7b llama3 llama3.1:8b mistral	2 1 1 0	
					TF-IDF	6	gemma:7b llama3 llama3.1:8b mistral	2 2 1 1	
		Podle odstavců	14	all-MiniLM-L6-v2	4	gemma:7b llama3 llama3.1:8b mistral	2 1 1 0		
				intfloat/e5-base-v2	6	gemma:7b llama3 llama3.1:8b mistral	2 2 2 0		
				TF-IDF	4	gemma:7b llama3 llama3.1:8b mistral	1 2 1 0		

Kategorie dotazu: Výčty									
Kategorie dotazu	Formát dokumentu		Strategie chunkování		Embedding model		Jazykový model	Počet bodů	
Výčty	DOCX	116	Fixní velikost (100/1)	58	all-MiniLM-L6-v2	20	gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
			intfloat/e5-base-v2			20	mistral	5	
							gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
							mistral	5	
			TF-IDF			18	gemma:7b	5	
							llama3	5	
							llama3.1:8b	4	
							mistral	4	
	MD	179	Fixní velikost (200/20)	58	all-MiniLM-L6-v2	20	gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
			intfloat/e5-base-v2			19	mistral	5	
							gemma:7b	5	
							llama3	5	
							llama3.1:8b	4	
							mistral	5	
			TF-IDF			19	gemma:7b	5	
							llama3	5	
							llama3.1:8b	4	
							mistral	5	
	PDF	174	Podle odstavců	0	all-MiniLM-L6-v2	0	gemma:7b	0	
							llama3	0	
							llama3.1:8b	0	
			intfloat/e5-base-v2			0	mistral	0	
							gemma:7b	0	
							llama3	0	
							llama3.1:8b	0	
							mistral	0	
			TF-IDF			0	gemma:7b	0	
							llama3	0	
							llama3.1:8b	0	
							mistral	0	
	MD	179	Fixní velikost (100/1)	59	all-MiniLM-L6-v2	20	gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
			intfloat/e5-base-v2			20	mistral	5	
							gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
							mistral	5	
			TF-IDF			19	gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
							mistral	4	
	PDF	174	Fixní velikost (200/20)	58	all-MiniLM-L6-v2	19	gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
			intfloat/e5-base-v2			20	mistral	4	
							gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
							mistral	5	
			TF-IDF			19	gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
							mistral	4	
	PDF	174	Fixní velikost (200/20)	57	all-MiniLM-L6-v2	19	gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
			intfloat/e5-base-v2			19	mistral	4	
							gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
							mistral	4	
			TF-IDF			19	gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
							mistral	4	
	PDF	174	Podle odstavců	59	all-MiniLM-L6-v2	19	gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
			intfloat/e5-base-v2			20	mistral	4	
							gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
							mistral	5	
			TF-IDF			20	gemma:7b	5	
							llama3	5	
							llama3.1:8b	5	
							mistral	5	

Kategorie dotazu: Vizuální obsah									
Kategorie dotazu	Formát dokumentu		Strategie chunkování		Embedding model		Jazykový model	Počet bodů	
Vizuální obsah	DOCX	2	Fixní velikost (100/1)	0	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
					intfloat/e5-base-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
					TF-IDF	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
			Fixní velikost (200/20)	0	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
					intfloat/e5-base-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
					TF-IDF	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0	
		Podle odstavců	2	all-MiniLM-L6-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0		
				intfloat/e5-base-v2	0	gemma:7b llama3 llama3.1:8b mistral	0 0 0 0		
				TF-IDF	2	gemma:7b llama3 llama3.1:8b mistral	0 0 0 2		
		MD	54	Fixní velikost (100/1)	14	all-MiniLM-L6-v2	4	gemma:7b llama3 llama3.1:8b mistral	0 0 0 4
						intfloat/e5-base-v2	3	gemma:7b llama3 llama3.1:8b mistral	0 0 1 2
						TF-IDF	7	gemma:7b llama3 llama3.1:8b mistral	0 0 3 4
	Fixní velikost (200/20)			20	all-MiniLM-L6-v2	7	gemma:7b llama3 llama3.1:8b mistral	0 4 2 1	
					intfloat/e5-base-v2	7	gemma:7b llama3 llama3.1:8b mistral	0 0 3 4	
					TF-IDF	6	gemma:7b llama3 llama3.1:8b mistral	0 0 2 4	
	Podle odstavců		20	all-MiniLM-L6-v2	7	gemma:7b llama3 llama3.1:8b mistral	0 0 3 4		
				intfloat/e5-base-v2	4	gemma:7b llama3 llama3.1:8b mistral	0 0 0 4		
				TF-IDF	9	gemma:7b llama3 llama3.1:8b mistral	0 4 2 3		
	PDF		81	Fixní velikost (100/1)	28	all-MiniLM-L6-v2	8	gemma:7b llama3 llama3.1:8b mistral	0 2 2 4
						intfloat/e5-base-v2	10	gemma:7b llama3 llama3.1:8b mistral	0 4 3 3
						TF-IDF	10	gemma:7b llama3 llama3.1:8b mistral	0 3 3 4
		Fixní velikost (200/20)		23	all-MiniLM-L6-v2	9	gemma:7b llama3 llama3.1:8b mistral	0 3 2 4	
					intfloat/e5-base-v2	3	gemma:7b llama3 llama3.1:8b mistral	0 0 2 1	
					TF-IDF	11	gemma:7b llama3 llama3.1:8b mistral	0 4 3 4	
		Podle odstavců	30	all-MiniLM-L6-v2	9	gemma:7b llama3 llama3.1:8b mistral	0 3 2 4		
				intfloat/e5-base-v2	10	gemma:7b llama3 llama3.1:8b mistral	0 4 4 2		
				TF-IDF	11	gemma:7b llama3 llama3.1:8b mistral	0 4 3 4		

Kategorie dotazu: Více větné uvažování									
Kategorie dotazu	Formát dokumentu		Strategie chunkování		Embedding model		Jazykový model	Počet bodů	
Více větné uvažování	DOCX	73	Fixní velikost (100/1)	31	all-MiniLM-L6-v2	9	gemma:7b	2	
							llama3	3	
							llama3.1:8b	2	
							mistral	2	
			intfloat/e5-base-v2	8			gemma:7b	2	
							llama3	2	
							llama3.1:8b	3	
							mistral	1	
			TF-IDF	14			gemma:7b	3	
							llama3	4	
							llama3.1:8b	3	
							mistral	4	
			all-MiniLM-L6-v2	8			gemma:7b	1	
							llama3	2	
							llama3.1:8b	2	
			intfloat/e5-base-v2	9			mistral	3	
							gemma:7b	2	
							llama3	2	
			TF-IDF	11			llama3.1:8b	3	
							mistral	4	
			Podle odstavců	14			gemma:7b	1	
							llama3	3	
							llama3.1:8b	2	
			all-MiniLM-L6-v2	8			mistral	2	
							gemma:7b	0	
							llama3	0	
			intfloat/e5-base-v2	0			llama3.1:8b	0	
							mistral	0	
							gemma:7b	0	
			TF-IDF	6			llama3	3	
							llama3.1:8b	1	
							mistral	2	
	MD	79	Fixní velikost (100/1)	21	all-MiniLM-L6-v2	4	gemma:7b	1	
							llama3	2	
							llama3.1:8b	1	
			intfloat/e5-base-v2	5			mistral	0	
							gemma:7b	1	
							llama3	1	
							llama3.1:8b	1	
			TF-IDF	12			mistral	2	
							gemma:7b	3	
							llama3	3	
			all-MiniLM-L6-v2	2			llama3.1:8b	3	
							mistral	3	
							gemma:7b	0	
			intfloat/e5-base-v2	12			llama3	0	
							llama3.1:8b	1	
							mistral	1	
			TF-IDF	14			gemma:7b	4	
							llama3	2	
							llama3.1:8b	3	
			all-MiniLM-L6-v2	11			mistral	3	
							gemma:7b	2	
							llama3	4	
			intfloat/e5-base-v2	9			llama3.1:8b	4	
							mistral	4	
							gemma:7b	4	
			TF-IDF	10			llama3	1	
							llama3.1:8b	3	
							mistral	3	
	PDF	102	Fixní velikost (100/1)	28	all-MiniLM-L6-v2	12	gemma:7b	2	
							llama3	4	
							llama3.1:8b	4	
			intfloat/e5-base-v2	7			mistral	2	
							gemma:7b	1	
							llama3	2	
							llama3.1:8b	2	
			TF-IDF	9			mistral	2	
							gemma:7b	2	
							llama3	1	
							llama3.1:8b	2	
			all-MiniLM-L6-v2	12			mistral	4	
							gemma:7b	3	
							llama3	4	
			intfloat/e5-base-v2	13			llama3.1:8b	3	
							mistral	2	
							gemma:7b	2	
			TF-IDF	9			llama3	4	
							llama3.1:8b	1	
							mistral	3	
			all-MiniLM-L6-v2	14			gemma:7b	3	
							llama3	3	
							llama3.1:8b	4	
			intfloat/e5-base-v2	14			mistral	4	
							gemma:7b	3	
							llama3	4	
							llama3.1:8b	4	
			TF-IDF	12			mistral	3	
							gemma:7b	2	
							llama3	4	
							llama3.1:8b	2	
							mistral	4	

Kategorie dotazu: Mimo znalost dokumentu									
Kategorie dotazu	Formát dokumentu		Strategie chunkování		Embedding model		Jazykový model	Počet bodů	
Mimo znalost dokumentu	DOCX	138	Fixní velikost (100/1)	45	all-MiniLM-L6-v2	15	gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	3	3
							mistral	4	4
			intfloat/e5-base-v2	15			gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	3	3
			TF-IDF	15			mistral	4	4
							gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	3	3
							mistral	4	4
	DOCX	138	Fixní velikost (200/20)	47	all-MiniLM-L6-v2	16	gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	4	4
			intfloat/e5-base-v2	16			mistral	4	4
							gemma:7b	4	4
							llama3	4	4
			TF-IDF	15			llama3.1:8b	4	4
							mistral	3	3
							gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	4	4
							mistral	4	4
	DOCX	138	Podle odstavců	46	all-MiniLM-L6-v2	15	gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	3	3
			intfloat/e5-base-v2	15			mistral	4	4
							gemma:7b	4	4
							llama3	4	4
			TF-IDF	16			llama3.1:8b	4	4
							mistral	4	4
							gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	4	4
							mistral	4	4
	MD	133	Fixní velikost (100/1)	44	all-MiniLM-L6-v2	13	gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	3	3
			intfloat/e5-base-v2	15			mistral	2	2
							gemma:7b	4	4
							llama3	4	4
			TF-IDF	16			llama3.1:8b	3	3
							mistral	4	4
							gemma:7b	4	4
			Fixní velikost (200/20)	47	all-MiniLM-L6-v2	16	llama3	4	4
							llama3.1:8b	4	4
							mistral	4	4
			intfloat/e5-base-v2	16			gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	4	4
	MD	133	TF-IDF	15			mistral	4	4
							gemma:7b	4	4
							llama3	4	4
			Podle odstavců	42	all-MiniLM-L6-v2	15	llama3.1:8b	3	3
							mistral	4	4
							gemma:7b	4	4
			intfloat/e5-base-v2	14			llama3	4	4
							llama3.1:8b	3	3
							mistral	3	3
			TF-IDF	13			gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	3	3
							mistral	2	2
	PDF	141	Fixní velikost (100/1)	47	all-MiniLM-L6-v2	16	gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	4	4
			intfloat/e5-base-v2	15			mistral	4	4
							gemma:7b	4	4
							llama3	4	4
			TF-IDF	16			llama3.1:8b	3	3
							mistral	4	4
							gemma:7b	4	4
			Fixní velikost (200/20)	47	all-MiniLM-L6-v2	15	llama3	4	4
							llama3.1:8b	4	4
							mistral	3	3
			intfloat/e5-base-v2	16			gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	4	4
	PDF	141	TF-IDF	16			mistral	4	4
							gemma:7b	4	4
							llama3	4	4
			Podle odstavců	47	all-MiniLM-L6-v2	16	llama3.1:8b	4	4
							mistral	4	4
							gemma:7b	4	4
			intfloat/e5-base-v2	15			llama3	4	4
							llama3.1:8b	3	3
							mistral	4	4
			TF-IDF	16			gemma:7b	4	4
							llama3	4	4
							llama3.1:8b	4	4
							mistral	4	4

PŘÍLOHA P VII: OBECNÉ ZÁSADY PŘÍPRAVY DOKUMENTU PRO RAG APLIKACE

Kapitola shrnuje obecná pravidla a doporučení pro strukturování a předzpracování dokumentů, které mají být použity v systémech využívajících technologii RAG (Retrieval-Augmented Generation). Doporučení vycházejí ze zjištění, která vzešla z analýzy v rámci diplomové práce.

Cílem tohoto textu je nabídnout ucelený přehled zásad, které mohou sloužit jako vodítko pro firmy plánující využívat technologii RAG k automatickému vyhledávání informací z interních dokumentů. Pokud jsou tyto dokumenty správně strukturované a formátované, lze výrazně zvýšit šanci, že jazykový model dokáže správně porozumět obsahu a vrátit přesnou a relevantní odpověď na položený dotaz.

- **Formát dokumentu**

Preferovanými formáty pro zpracování pomocí RAG technologie jsou strukturované textové soubory, jako je Markdown, HTML nebo čistý text. Tyto formáty umožňují jednoznačné označení nadpisů, seznamů, tabulek a poznámek. Z praktického hlediska se Markdown jeví jako ideální kompromis mezi lidskou čitelností a strojovou zpracovatelností.

Formát DOCX se naopak ukazuje jako problematický, a to zejména kvůli složitější vnitřní struktuře, použití textových polí, tabulek nebo grafických prvků. V diplomové práci bylo prokázáno, že tento formát často vedl ke ztrátě informací a chybným odpovědím.

V PDF dokumentech by měla být zachována textová vrstva. Díky přítomnosti textové vrstvy funkce `page.get_text()` dokáže vrátit čitelný a přesně rozdělený obsah na řádky, odstavce a nadpisy. Tím lze dokument dobře segmentovat do smysluplných bloků textu, které se použijí pro vyhledání kontextu ke konkrétnímu dotazu. Pokud je PDF dokument skenován pouze jako obrázek bez textu (např. naskenovaný list papíru bez OCR), je jedinou možností rozpoznávání textu pomocí OCR, které je však náchylné k chybám (např. čárky místo háčků, překlepy atd.). Výsledek OCR bývá méně strukturovaný, tj. bez odstavců, mohou chybět řádky, nebo se přehodí pořadí. Což nevede k dobrému rozdělení na kvalitní chunky.

- **Struktura a hierarchie obsahu**

Každý dokument by měl mít jednoznačnou strukturu, která reflektuje tematický obsah. Použití nadpisů a podnadpisů v hierarchickém pořadí pomáhá udržet kontext. Chunkovací strategie založené na odstavcích nebo sekcích využívají těchto struktur k tomu, aby zachovaly logické vazby.

Z praxe vyplynulo, že pokud byla důležitá informace umístěna do odrážkového seznamu bez nadpisu nebo byla uvedena jako samostatná věta bez kontextu, RAG aplikace ji často nedokázala relevantně zařadit. Striktní dodržování struktury pomáhá zachovat smysluplné rozdělení textu na části, které dávají smysl jak člověku, tak jazykovému modelu.

- **Použití srozumitelného a jednoznačného jazyka**

Texty by měly být psány srozumitelně a bez nadměrného odborného žargonu. Zkratky by měly být vysvětleny při prvním použití. Tím se zvyšuje pravděpodobnost, že i model s omezeným kontextem pochopí význam a správně na něj naváže.

- **Předzpracování textu**

Texty určené ke zpracování v RAG aplikacích by měly být před nahráním očištěny od irelevantních částí. Za nežádoucí se považují například právní prohlášení, nadbytečné záhlaví a zápatí, opakující se fráze nebo informace, které nesouvisí s doménou dokumentu. Tyto části totiž mohou zbytečně zatěžovat indexaci a snižovat přesnost vyhledávání.

Dále by měl být jazyk co nejjednodušší a věcný. Komplikované věty, složité výrazy nebo příliš odborný jazyk mohou snižovat přesnost vektorizace a ztěžovat vyhledávání. Jednoduchý jazyk zvyšuje pravděpodobnost, že model správně pochopí klíčové informace. Obecně je vhodné psát tak, jako bychom dokument připravovali pro nového kolegu bez předchozí znalosti problematiky.

Důvodem pro zjednodušení jazyka je i to, že embedding modely neprovádějí hluboké porozumění textu, ale pracují na základě matematické reprezentace významu. Čím srozumitelnější a jednoznačnější jazyk použijeme, tím lepší vektorovou reprezentaci model vytvoří.

- **Optimalizace netextových prvků (obrázky, tabulky)**

Protože LLM modely obvykle nepracují s vizuálními prvky, je nutné důležitou informaci obsaženou v obrázcích nebo tabulkách vhodně popsat textově. Obrázky by měly být opatřeny popisky, které sdělují, co je zobrazeno a jaký je význam případných značek nebo šipek. Bez těchto doprovodných informací by model nepochopil význam vizuálních údajů.

Tabulky by měly být ideálně reprezentovány pomocí struktury v Markdown formátu. V diplomové práci se ukázalo, že tabulky uvnitř DOCX souborů byly často při zpracování ztraceny. Převedení tabulek do textového formátu zajišťuje, že informace budou zachovány i při vektorizaci.

Běžné metody jako OCR převádějí tabulky v PDF dokumentech na prostý text, kde je každý řádek tabulky interpretován jako samostatný řádek textu. Tím se ztrácí informace o sloupcích a vztazích mezi daty v jednotlivých buňkách tabulky. Pokud se takto zpracovaná data z tabulky předloží jazykovému modelu, je pro něj obtížné interpretovat tyto informace, protože chybí kontext daný sloupcem a řádkem.

Formát Markdown je pro jazykové modely lépe srozumitelný, protože explicitně definuje hranice mezi řádky a sloupci. Markdown tedy dokáže zachovat řádkovou i sloupcovou strukturu tabulky. Jazykovému modelu to umožňuje lépe interpretovat tabulková data a provádět s nimi interakce v rámci RAG aplikace.

- **Shrnutí a metadata**

Na začátku nebo konci dokumentu se doporučuje uvést stručné shrnutí obsahu. Metadata jako název dokumentu, autor, oddělení, nebo datum aktualizace pomáhají při vyhledávání nebo filtrování odpovědí. Pro modely tato metadata představují důležitý kontext, díky kterému mohou dobře interpretovat obsah a poskytovat přesnější odpovědi.

- **Validace dokumentů**

Po nahrání dokumentů do znalostí báze je vhodné otestovat jejich zpracování pomocí dotazů. Pokud systém nenalezne odpovědi nebo je odpověď chybná, je nutné zrevidovat strukturu a obsah dokumentu. Tento krok je klíčový pro zajištění spolehlivosti odpovědí v ostrém provozu.

Testování navíc pomáhá odhalit, zda určité části textu nejsou nečitelné nebo chybně interpretovatelné. Pravidelná kontrola a zpětná vazba od uživatelů umožňuje průběžné zlepšování kvality znalostní báze.