Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informačních technologií

Koncept využití Umělé inteligence k rozpoznání patternů a logických celků textu

Diplomová práce

Autor: Bc. Václav Tomíček
Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Pavel Blažek

Odborný konzultant: Ing. Jakub Vlk, Quadient Technologies Czech s.r.o.

Hradec Králové srpen 2022

Prohlášení:

Prohlašuji, že jsem diplomovou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 15.8.2022 Bc. Václav Tomíček

Poděkování:

Děkuji vedoucímu diplomové práce Ing. Pavlu Blažkovi za metodické vedení práce a odbornému vedení Ing. Jakuba Vlka. Také bych chtěl poděkovat společnosti Quadient, která mi poskytla veškeré potřebné nástroje a tolik potřebnou podporu.

Anotace

Diplomová práce se zabývá problematikou rozpoznávání a zpracování textu v grafických dokumentech. Práce je věnována seznámení se s termíny jako je strojové učení, optické rozpoznávání textu a API. Jsou zde rozebrány základní definice a oblasti využití. V další kapitole jsou představeny nástroje mezinárodních firem, jenž představují to nejlepší, co lze využít k rozpoznávání textu, a tedy i k řešení naší problematiky.

V rámci praktického řešení je řešen koncept migrace dokumentů, které jsou v grafickém formátu, a tedy nelze s nimi snadno nijak dále pracovat. V praktickém řešení jsou představené nástroje analyzovány a porovnány. Dle dosažených výstupů a jejich možností je zvolen nejvhodnější z nichž, jenž je dále využit pro následnou implementaci do již existujícího nástroje od společnosti Quadient. V tomto nástroji je vytvořen zcela nový přístup, jak migrovat grafické dokumenty mezi systémy a jsou rovněž implementovány nové funkcionality pro další zpracování výsledného textu.

Annotation

Title: The concept of using Artificial Intelligence to recognize patterns and logical units of text

The diploma thesis deals with the issue of text recognition and processing in graphic documents. The work is devoted to familiarization with terms such as machine learning, optical text recognition, and API. Basic definitions and areas of use are discussed here. In the next chapter, the tools of international companies are presented, which represent the best that can be used for text recognition, and therefore also for solving our problem.

As part of the practical solution, the concept of migrating documents that are in graphic format and therefore cannot be easily worked with in any further way is solved. In a practical solution, the introduced tools are analyzed and compared. According to the achieved outputs and their possibilities, one of them is chosen, which is further used for subsequent implementation into an already existing tool from the company Quadient. In this tool, a completely new approach to how to migrate graphic documents between systems is created, and new functionalities are also implemented for further processing of the resulting text.

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Dnes jsme v době, kdy se veškeré technologie jdou dopředu nesmírným tempem a mnoho společností stále nepřešlo zcela do digitálního věku. Jelikož se jedná o velmi nákladný a náročný proces, společnosti se zaměřují na co možná nejlevnější, ale přitom efektivní a nejvhodnější řešení tohoto problému.

Mnoho společností tedy investuje do digitalizace a s tím se pojí i zvolené téma. Každá společnost má spousty druhů dokumentů v různých formách. Mezi ně patří různé smlouvy, dopisy zákazníkům nebo třeba výpisy z účtů. Pokud by chtěla společnost migrovat tyto dokumenty do digitální podoby, bylo by velmi časově náročné všechny vzít a vytvářet je znovu. Nehledě na to, že pokud firma existuje již desítky let, mohlo by dojít mezi dokumenty k redundancím a v tu chvíli by samotná digitalizace nepřinesla téměř žádný užitek. Pokud k tomu připočteme vytvoření i vhodného systému dle požadavků firmy, celý proces je velmi zdlouhavý a může zabrat měsíce ale i roky.

Spousta firem, co se zabývají migrací dokumentů se pokoušejí vytvářet nové systémy či nástroje, které by uspíšily proces migrace. Tím poté snižují nejen časovou náročnost ale i koncové náklady zákazníků za migraci dokumentů a upevňují si svojí pozici na trhu.

Samotné zpracování textových dokumentů ve formátech typu PDF, DOCX a dalších je vcelku běžná věc. Jejich zpracování lze již efektivně učinit, a to včetně vyřešení redundancí a provedení konsolidace. Oproti tomu například naskenovaný, vyfocený nebo jinak zpracovaný dokument, jenž se zobrazuje pouze jako grafický dokument se v zásadě nechová stejně jako klasický dokument. Takové dokumenty můžeme mít v obrázkových formátech typu PNG, ale může být také ve formátu PDF. V tu chvíli je potřeba co možná nejpřesnější a nejspolehlivější detekce a analýza textu z obrázku.

2 Cíl práce

Cílem diplomové práce je seznámit se s koncepty optického rozpoznávání znaků založenými na strojovém učení a vymyslet postup, jak tyto koncepty využít k migraci dokumentů. Práce by měla popsat teoretickou stránku všech problémů a samotné využití konceptů v praktickém řešení. Nástroje vycházející z teoretické části, budou mezi sebou porovnány, zjištěny jejich klady a zápory a jejich cenová nákladnost. Poté by nejlepší nástroj měl být využit k implementaci do již existujícího nástroje pro zjednodušení migrace dokumentů. V implementaci by měla být začleněna také další práce s výstupním textem.

3 Metodika zpracování

K dosažení všech stanovených cílů, bylo nutné seznámení se základními koncepty optického rozpoznávání znaků a strojového učení. Dále seznámení se s již existujícími nástroji a technologiemi využívající výše zmíněné koncepty. Na základě těchto informací pak bylo možno vyhledat nejefektivnější technické řešení definovaného problému a vytvoření praktické části diplomové práce. V posledním kroku bylo řešení nasazeno, do již existujícího nástroje, kde otevírá zcela nové možnosti k migraci dokumentů včetně pokročilého zpracování.

4 Strojové učení

Strojové učení, známé také jako Machine learning je podoblastí umělé inteligence (AI – Artificial Intelligence). Jednoduše řečeno, umělá inteligence je definována jako studium inteligentních agentů. Jedná se tedy o systém, který vnímá své prostředí a podniká akce, které maximalizují šanci na dosažení svých cílů. [1]

Většina počítačového softwaru, včetně i rané umělé inteligence většinou zahrnuje předem napsané přesné kroky, které poté software provádí. Nebo různé specifikace pravidel, která jsou přesně dodržována. Software obsahující strojové učení funguje tak, že místo toho, aby byly algoritmy naprogramovány co mají dělat, mají naopak schopnost se učit co dělat. Ať už jde například o rozpoznávání obrazu nebo řeči, samořídící auta, počítačovou biologii (využívané například k identifikaci nádorů), digitální společníky jako je Alexa od společnosti Amazon nebo stejně tak program od Google Deepmind AlphaGo, který porazil světového hráče číslo jedna ve hře Go. Toto všechno bylo dosaženo díky strojovému učení. Dnes je strojové učení tak rozšířené, že se každý z nás již v nějaké formě se strojovým učením setkal, aniž by si toho byl vědom. Nutno také podotknout že díky jeho rozšíření se strojové učení stalo pro některé výzkumníky synonymem k umělé inteligenci. [2]

Existuje zde několik druhů strojového učení, kde každý druh definuje určitý postup, jak naučit software vykonávat potřebné procesy nebo ho zdokonalovat. Mezi hlavní druhy strojového učení spadá především učení s učitelem (Supervised learning), učení bez učitele (Unsupervised learning) a zpětnovazební učení (Reinforcement learning).

4.1 Učení s učitelem

Jak je psáno v Practical Machine Learning with Rust od autora Joydeep Bhattacharjee v podkapitole Supervised Learning [3], k druhu strojové učení nazvané učení s učitelem dochází tehdy, když systému předáme potřebná vstupní data a řekneme mu, k jakým výstupům by měl dojít. Systém pak porovnává svoje výstupní data s nastavenými daty. Model využívající učení s učitelem nikdy nebude umět věci, které jsme ho nenaučili, musí mít každou činnost určitým způsobem nadefinovanou.

Strojové učení s učitelem můžeme dále rozdělit podle typu procesu, který chceme dělat. Pokud máme procesy, ve kterých je cílová proměnná spojitá, jsou tyto případy nazvány jako regresní problémy. Dobrým příkladem může být cena produktu, kde cena může mít libovolnou hodnotu. Pokud bychom chtěli regresní techniky využít k předpovědím, je to potřeba provést na něčem co vyžaduje předpověď určitého množství. Kvalita regresního modelu je obecně měřena pomocí určité formy chybových měření, to znamená rozdílu mezi cílovými hodnotami a předpokládanými hodnotami, jež jsme nastavili na začátku procesu.

Druhým typem strojového učení s učitelem je takzvaný kvalifikační problém. Kvalifikační problémy se odlišují od regresních problémů v tom, že jsou diskrétní a konečné. Například emailovou zprávu můžeme kategorizovat jako spam nebo jako cokoliv si určíme. Problém je tedy samotná kvalifikace, kdy nás zajímá výsledný segment, do kterého bude spadat konkrétní sada 32 čtecích funkcí. Kvalita klasifikačního modelu se obecně poměřuje pomocí míry přesnosti k zadaným výstupním datům, což v podstatě představuje počet, kolikrát byl model správný.

4.2 Učení bez učitele

Další typ strojového učení, nazvaný učení bez učitele (Unsupervised learning) se dle odborné literatury Machine Learning and Artificial Intelligence [4] zabývá problémy, kde není k dispozici žádné označení dat, díky kterému by šlo proces reverzně procházet. Co se týče průběhu zpracování, není způsob, jak získat jakoukoliv formu zpětné vazby během procesu. Tato absence označených vzorků znamená zásadní myšlenkový posun od kontrolovaných metod. Můžeme nabýt dojmu, že tento druh zpracování nemusí přinést žádné užitečné informace, ale opak je pravdou. Existuje mnoho situací, kdy strojové učení bez učitele je velmi užitečný druh zpracování.

První a nejdůležitější jsou náklady označování. Ve spoustě situací může být úplné označení všech trénovacích dat velmi výpočetně nákladné a prakticky i nemožné. Proto je potřeba pracovat pouze s menší sadou označených dat. V takových případech lze začít pracovat se strojovým učením s učitelem a poté model postupně rozšiřovat pomocí strojového učení bez učitele na větší sadu dat. V dalších

případech nemusí být označení vůbec k dispozici a v tu chvíli je potřeba porozumět struktuře a složení dat provedením průzkumné, takzvané nekontrolované analýzy. Pochopení struktury dat může také pomoci s výběrem správného algoritmu nebo lepší sady inicializovaných parametrů pro algoritmus v případech, kdy bude nakonec použito učení s učitelem. Obecně platí, že učení bez učitele představuje důležitý pilíř moderního strojového učení.

4.3 Zpětnovazební učení

Zpětnovazební učení (Reinforcement learning) nebo též známé jako učení posilováním je stručně popsáno v odborné literatuře Computational Methods for Deep Learning [5]. Zpětnovazební učení je o tom, jak mapovat situace k akcím. Akce mohou ovlivnit okamžité odměny, ale také situace, které následují a tím i následné možné odměny. Zpětnovazební učení se týká především dynamických systémů, konkrétně optimálního řízení a Markovova rozhodovacího procesu (MDP). Zpětnovazební učení explicitně bere v úvahu celý problém cíleně orientovaného agenta v interakci s nejistým prostředím a zároveň hledá kompromis mezi zkoumáním a využíváním. Systém zahrnuje takzvané zásady, signál odměny, hodnotovou funkci a model prostředí. Zásady definují způsob chování učícího se agenta v určeném čase. Odměnový signál jasně definuje cíl v problému zpětnovazebního učení. Hodnotová funkce určuje, co je z dlouhodobého hlediska dobré pro model. Hodnota stavu je celková hodnota odměny, kterou může agent očekávat v budoucnosti, počínaje tímto stavem. Odměny určují okamžitou vnitřní potřebnost stavů prostředí. Posledním prvkem zpětnovazebního učení je model prostředí. Pro příklad zde můžeme uvést Google Street View. Ten by nás mohl navigovat po městě, po vesnicích a celkově ve venkovním prostředí, ale v jakékoli budově už nám moc nepomůže. V tomto vnitřním prostředí by nás mohlo navigovat zpětnovazební učení. Předpokládejme, že máme mapu budovy a chceme dosáhnout určité místnosti, ve které se koná důležitá schůzka. Úspěšné vyřešení tohoto problému nám může pomoci najít nejkratší cestu do místnosti bez využití informací získané z GPS. Stručně řečeno, zpětnovazební učení silně závisí na statusu a zásadách, což je výpočetní přístup k pochopení a automatizaci. Zpětnovazební

učení využívá Markovovy rozhodovací procesy k definování interakce mezi učícím se agentem a jeho prostředím z hlediska stavů, akcí a odměn.

5 Hluboké učení a neuronové sítě

Hluboké učení, známo spíše pod anglickým pojmem Deep learning je podmnožinou strojového učení. Jak je napsáno v článku What is Deep Learning? [6]. Hluboké učení se pokouší napodobit lidský mozek, i když zdaleka neodpovídá jeho schopnostem. Ve své podstatě využívá neuronové sítě, které umožňují se učit z velkého množství dat. Zatím co neuronová síť s jednou vrstvou může provádět přibližné předpovědi, další skryté vrstvy mohou optimalizovat a vylepšovat celý model a tím získávat přesnější data. Díky tomu pak systémy mohou shlukovat obrovské množství dat a předpovídat s velmi velkou přesností další výstupy.

Hluboké učení se využívá v mnoha aplikacích a služeb umělé inteligence, které zlepšují automatizaci a provádějí analytické a fyzické úkoly bez lidského zásahu. Technologie hlubokého učení se skrývá za každodenními produkty a službami, jako jsou různí digitální asistenti, hlasové ovládání zařízení, příklad: televize, anebo i k detekci podvodů s kreditními kartami. Hluboké učení ale najdeme také u nejnovějších technologiích jako jsou například autonomní vozidla, která by se mohla sama pohybovat po silnicích. Za vším tímhle je podepsaná technologie hlubokého učení.

Pokud je hluboké učení podmnožinou strojového učení, je nutné vysvětlit, v čem jsou tedy odlišné. Hluboké učení se od klasického strojového učení odlišuje typem dat, se kterými pracuje a metodami, kterými se zdokonaluje. Algoritmy strojového učení využívají k vytváření předpovědí strukturovaná a označená data, to znamená, že konkrétní funkce jsou definovány ze vstupních dat pro model a organizovány do tabulek. To nutně neznamená, že se nepoužívají nestrukturovaná data. Znamená to pouze to, že pokud máme nestrukturovaná data, je nutné, aby prošla určitým předzpracováním a byla uspořádána do strukturovaného formátu. Díky tomu budou použitelné pro strojové učení. Oproti tomu hluboké učení eliminuje některé tyto předzpracovávání dat, a tudíž mohou algoritmy hlubokého učení přijímat a zpracovávat nestrukturovaná data, jako je například text anebo obrázky. Díky tomu může automatizovat extrakci prvků a odstraňuje určitou závislost na lidském

zásahu. Dobrým příkladem může být sada fotografií s domácími mazlíčky, které chceme rozřadit podle druhu mazlíčka, kterého vidíme na fotce. Algoritmus hlubokého učení určuje sám aspekty, které jsou důležité pro rozlišení domácích mazlíčků, zatím co ve strojovém učení, by toto rozlišení muselo být vytvořeno ručně. Ohledně fungování strojového učení. Jak již bylo zmíněno výše, neuronové sítě s hlubokým učením nebo umělé neuronové sítě se především pokoušejí napodobit lidský mozek prostřednictvím kombinace datových vstupů, vah a zkreslení. Tyto prvky spolupracují na přesném rozpoznání, klasifikaci a popisu objektů v datech. Hluboké neuronové sítě se skládají z několika vrstev vzájemně propojených uzlů, z nichž každá navazuje na předchozí vrstvu a zpřesňuje predikci nebo kategorizaci. Tento postup výpočtů sítí se nazývá dopředné šíření. Vstupní a výstupní vrstvy hluboké neuronové sítě se nazývají viditelné vrstvy. Vstupní vrstva je místo, kde model hlubokého učení přijímá data pro zpracování. Výstupní vrstva je místem, kde se provádí konečná předpověď nebo klasifikace.

Další proces zvaný zpětné šíření (backpropagation) využívá algoritmy, jako je sestup gradientu k výpočtu chyb v předpovědích a poté upravuje váhy a funkce zkreslení pohybem zpět přes vrstvy ve snaze trénovat model. Dopředné šíření a zpětné šíření společně umožňují neuronové síti předpovídat a odpovídajícím způsobem korigovat případné chyby. Postupem času se algoritmus postupně zpřesňuje.

Je nutné zdůraznit, že výše uvedené informace se týkají spíše těch nejjednodušších typů neuronových sítí a jsou použity nejjednodušší termíny k potřebnému pochopení. Algoritmy hlubokého učení jsou neuvěřitelně složité a existuje více typů neuronových sítí, kdy každý typ pracuje s konkrétními případy, problémy nebo se specifickými soubory dat.

Mezi příklady dalších neuronových sítí lze zařadit Konvoluční neuronové sítě (Convolutional neural networks – CNNs), které jsou používané především v aplikacích počítačového vidění a klasifikace obrazu. Tento druh neuronové sítě dokáže detekovat prvky a vzory v obraze, což umožňuje rozpoznávat objekty.

Pro zajímavost zde můžeme uvést, že v roce 2015 konvoluční neuronové sítě porazily člověka ve výzvě rozpoznávání objektů. Dalším typem neuronové sítě, o které se tu zmíníme, jsou Rekurentní neuronové sítě (Recurrent neural network – RNNs). Tyto sítě jsou obvykle využívány v aplikacích pro rozpoznávání přirozeného jazyka a řeči, protože využívají sekvenční data, nebo data časových řad.

Aplikace pro hluboké učení jsou v reálném světě součástí každodenního života, ale většinou jsou tak dobře integrovány do produktů a služeb, že si uživatelé ani neuvědomí, že s nimi přichází do styku. Na pozadí aplikací probíhá složité zpracování dat. Mezi dobré příklady využití určitě zapadají chatboti, které se využívají v různých aplikacích, službách a portálech zákaznických služeb a jsou přímou formou umělé inteligence. Tradiční chatboti využívají přirozený jazyk a vyskytují se běžně v nabídkách podobných call centru. Sofistikovanější řešení chatbotů však usiluje o to, pomocí učení zjistit, zda existuje více odpovědí na nejednoznačné otázky. Na základě odpovědí, které získá, se pak chatbot pokouší na tyto otázky přímo odpovědět nebo nasměrovat konverzaci k lidskému uživateli. Spadají sem také finanční služby, virtuální asistenti ale i zdravotnický průmysl. Poslední jmenovaný získává z aplikace pro rozpoznávání snímků důležité informace, které mohou podporovat lékařské specialisty a radiology tím, že jim zanalyzuje nebo vyhodnotí snímky pacienta velmi rychle a přesně. To vše díky hlubokému učení.

Hardwarové požadavky na hluboké učení jsou velmi vysoké a vyžadují ohromné množství výkonu a energie. Vysoce výkonné grafické jednotky jsou ideální, protože zvládají velké objemy výpočtů ve více jádrech s velkou dostupnou pamětí. Správa více grafických jednotek na místě však může mít vysoké nároky na interní zdroje a může být neuvěřitelně nákladná na škálování.

6 Optické rozpoznávání znaků

Optické rozpoznávání znaků, známé jako OCR – Optical Character Recognition je technologie, která dokáže rozpoznávat text z digitálního obrázku. Technologii lze použít k převodu fyzického dokumentu a tím se ušetří čas, náklady a další zdroje. Tyto dokumenty mohou být jakéhokoliv typu: kniha, výpis z účtu, nebo třeba

zápisky psané vlastní rukou. Dokumenty předáme počítači pomocí skeneru nebo fotoaparátem. V tu chvíli máme vytvořený soubor nejčastěji v obrázkovém formátu typu JPG nebo v souboru typu PDF obsahující obrázky skenu. Tento dokument načteme do programu, který pracuje s optickým rozpoznávání znaků a převede nám námi vytvořené obrázkové dokumenty na upravitelné textové soubory, jež budeme moci snadno editovat a přepisovat. [7]

Na jednoduchém případu lze demonstrovat jak funguje optické rozpoznávání textu z pohledu počítače. Příklad je vypůjčen z článku How does OCR document scanning work. [8] Když čtete tato slova na obrazovce počítače, vaše oči a mozek provádějí optické rozpoznávání znaků, aniž byste si toho všimli. Podle tvaru, který vytvářejí pixely na obrazovce, váš mozek vyhodnotí a dokáže říct, co se kdo pokouší sdělit. V tomto případě se jedná o písmena, čísla a interpunkční znaménka. Počítače toto umí taky, jen je to o něco složitější. Jelikož počítač nemá oči, musíme například starou knihu převést do takového stavu, aby s ní počítač mohl pracovat. Knihu tedy počítači poskytneme pomocí skeneru, nebo fotoaparátu ve formě grafického obrázku, který je nejčastěji ve formátu JPG. Počítač vidí obrázek ve formě barevných bodů, které vytvářejí výsledný obraz. Technologie OCR tyto barevné body rozpoznává a pokud vytvářejí nějaký tvar, který připomíná určitý znak, je to převedeno z obrázkového, do textového formátu. [8]

Optické rozpoznání textu může fungovat jako off-line proces, který nepotřebuje ke zprocesování dokumentu internet, ale samozřejmě existuje optické rozpoznávání znaku jako služba, která funguje jako cloudová služba. Takovou službu můžeme volat například pomocí API, o kterém je psáno v další části práce. Optické rozpoznávání textu má také více typů.

Prvním typem je optické rozpoznávání znaků, které bylo popsáno výše, zaměřuje se tedy především na strojově psaný text, který prochází znak po znaku. Dalším typem je optické rozpoznávání slov (Optical Word Recognition – OWR), jenž funguje téměř stejně jako optické rozpoznávání znaků, a proto je i často zaměňován za tento typ. Optické rozpoznávání slov se také zaměřuje na strojově psaný text a prochází text po slovech.

Třetím typem je takzvané optické rozpoznávání značek (Optical Mark Recognition – OMR), jenž je označení pro elektronickou metodu shromažďování dat

zpracovávaných lidmi pomocí rozpoznávání určitých značek nebo vzorů přímo v dokumentu.

Dalším typem již známějším je inteligentní rozpoznávání znaků (Intelligent Character Recognition – ICR). Jedná se o vylepšené optické rozpoznávání znaků s využitím umělé inteligence a neuronových sítí, díky čemuž se systém dokáže lépe vyvíjet. Pokaždé, když služba zpracovává nová data, systém se z nich zároveň učí.

Posledním typem je takzvané inteligentní rozpoznávání slov (Intelligent Word Recognition – IWR). Podobně jako inteligentní rozpoznávaní znaků, i inteligentní rozpoznávání slov využívá umělou inteligenci k rozeznání celých slov v dokumentu. Na rozdíl od optického rozpoznání slov, které slovo bere po písmenech, tento typ spojí všechna písmena do slovníku, aby extrahoval celá slova na základě rozeznání vzorů a různých algoritmů. Inteligentní rozpoznávání slov umožňuje interpretaci ručně psaných informací, jejichž dešifrování a zadávání by personálu normálně zabralo velké množství času. To znamená, že společnosti mohou být výrazně efektivnější a produktivnější a také minimalizovat pravděpodobnost lidské chyby.

7 API

API (Application Programming Interface) je termín, který označuje balíček funkcí, tříd a knihoven. Umožňují komunikaci dvou aplikací, nástrojů, produktů či služeb, aniž byste museli vědět, jak jsou implementovány. [9] To může zjednodušit vývoj aplikací, ušetřit drahocenný čas a v neposlední řadě peníze. Když navrhujete nové nástroje a produkty, nebo spravujete již existující, API vám může velmi zjednodušit návrh, správu, použití ale také poskytnou určitou flexibilitu a příležitost pro inovace. Rozhraní API jsou někdy považována za „smlouvy s dokumentací“. To představuje dohodu mezi stranami, kdy jedna strana odešle vzdálený požadavek s požadovanou strukturou a software druhé strany odpoví jasně definovanou strukturou zpět.

Protože API zjednodušuje vývojářům integraci nových aplikačních komponentů do existující architektury, pomáhají tím obchodním a IT týmům spolupracovat. Obchodní potřeby se často rychle mění v reakci na neustále se měnící digitální trhy, kdy nový konkurenti mohou pomocí nové aplikace změnit celé odvětví. Aby firmy zůstaly konkurenceschopní, je důležité, aby podporovaly rychlý vývoj a zavádění inovativních služeb. Vývoj cloudových nativních aplikací je identifikovatelný

způsob, jak zvýšit rychlost vývoje a spolehnutí se na propojení aplikační architektury mikro služeb prostřednictvím rozhraní API. [10]

7.1 Vzdálená API

Vzdálená API jsou navržena pro interakci prostřednictvím komunikační sítě. Vzdáleným máme na mysli, že prostředky se kterými se manipuluje, se vyskytují mimo počítač, který požadavek odesílá. Požadavek je posílán přes komunikační síť, což je v dnešní době především internet. Díky tomu je většina API navržena na základě webových standardů. Ne všechna vzdálená rozhraní API jsou webového typu, ale je dobré předpokládat, že jsou vzdálená.

Webové rozhraní API obvykle využívají HTTP pro příchozí zprávy a poskytují strukturu v odchozí zprávě. Tyto odpovědi většinou obsahují zprávu ve formě souboru XML nebo JSON. XML i JSON jsou preferované formáty, protože prezentují data takovým způsobem, s nimiž se dá snadno pracovat. [10]

7.2 SOAP vs. REST

REST (Representational state transfer) a SOAP (Simple object access protocol) jsou dva různé přístupy k online přenosu dat. Konkrétně oba definují, jak vytvořit aplikační programovací rozhraní, která umožňují komunikaci dat mezi webovými aplikacemi. REST je soubor architektonických principů, zatímco SOAP je oproti tomu oficiální protokol spravovaný W3C (World Wide Web Consortium). W3C je dle jejich stránek www.w3.org [11] mezinárodní společenství, kde členské organizace, zaměstnanci na plný úvazek a veřejnost spolupracují na vývoji webových standardů. Hlavním rozdílem mezi SOAP a REST je tedy ten, že SOAP je protokol na rozdíl od REST. Typicky tedy API využívá buď REST nebo SOAP, v závislosti na případu použití nebo na samotných preferencích vývojáře.

Když je požadavek na data odeslán do REST API, je to obvykle provedeno prostřednictvím protokolu hypertextového přenosu, jenž je běžně označován jako protokol HTTP. Jakmile je tento požadavek přijat, mohou rozhraní API, která jsou navržená pro REST vracet zprávy v různých formátech jako jsou HTML, XML, prostý text nebo jako JSON. Tyto rozhraní mohou být nazývané také jako RESTful API.

JSON (JavaScript object notation) je upřednostňovaný formát zprávy, protože jej lze číst jakýmkoli programovacím jazykem a je také velmi snadno čitelný pro lidi i stroje. Tímto způsobem jsou RESTful Api flexibilnější a lze je snadněji nastavit. RESTful aplikace by se měla řídit šesti architektonickými pokyny mezi něž spadají:

- Architektura klient-server, která je složená z klientů, serverů a prostředků.
- Bezstavová komunikace klient-server, což znamená že mezi požadavkem není na serveru uložen žádný klientský obsah a místo toho jsou informace o stavu relace uchovávány u klienta.
- Data uložitelná do mezipaměti, aby se eliminovala potřeba některých interakcí klient-server.
- Jednotné rozhraní mezi komponentami. Myšleno tak, že informace jsou přenášeny ve standardizované formě namísto specifických pro potřeby aplikace. Toto popisuje tvůrce REST Roy Fielding jako ústřední rys, který odlišuje architektonický styl REST od jiných síťových stylů.
- Vrstvené systémové omezení, kde interakce klient-server mohou být zprostředkovány hierarchickými vrstvami.
- Kód na vyžádání, to serverům umožňuje rozšířit funkčnost klienta přenosem spustitelného kódu. Zde nutno podotknout že také snižuje viditelnost, což z něj činí volitelné vodítko.

Oproti tomu SOAP je standardní protokol, který byl poprvé navržen tak, aby mohly komunikovat aplikace vytvořené v různých jazycích a na různých platformách. Protože se jedná o standardní protokol, má jasně definovaná pravidla, které zvyšují jeho složitost a režii, což může vést k delší době načítání stránky. Tyto standardy však také nabízejí vestavěné shody, díky nimž jsou vhodnější pro podnikové scénáře. Mezi vestavěné standardy lze zařadit zabezpečení, atomičnost, konzistenci, izolaci a trvanlivost (ACID), což je soubor vlastností pro zajištění spolehlivých databázových transakcí. Mezi běžné specifikace SOAP lze zařadit:

- Zabezpečení webových služeb (WS-security), standardizující způsob zabezpečení a přenosu zpráv prostřednictvím jedinečných identifikátorů nazývaných tokeny.

- WS-Reliablemessaging standardizující zpracování chyb mezi zprávami přenášenými přes nespolehlivou síťovou infrastrukturu.
- Adresování webových služeb (WS-addressing), balící informace o směrování jako metadata v hlavičkách SOAP, místo toho, aby se tyto informace uchovávaly hlouběji v síti.
- Popisný jazyk webových služeb (WSDL), popisuje, co webová služba dělá, kde tato služba začíná a končí.

Když je požadavek na data odeslán do SOAP API, může být zpracován prostřednictvím kteréhokoli z protokolů aplikační vrstvy, jako je HTTP pro webové prohlížeče, SMTP pro e-mail, TCP a další. Jakmile je však požadavek přijat, musí být vrácen SOAP zprávou jako XML, což je značkovací jazyk, který je lehce čitelný pro lidi i stroje. Také je dobré vědět, že na rozhraní SOAP API nelze uložit dokončený požadavek do mezipaměti prohlížeče, tudíž k němu nelze později přistupovat bez opětovného odeslání do rozhraní API.

Mnoho starších systémů může stále dodržovat SOAP, jelikož REST přišel později. Také je ale často považován za rychlejší alternativu ve webových scénářích. Samotný REST je sada pokynů, která nabízí flexibilnější implementaci, zatímco SOAP je protokol se specifickými požadavky jako je zasílání zpráv pomocí XML. S tím jde také shrnout to, že REST API jsou řekněme lehčí, a tak že jsou ideální pro novější technologie jako je internet věcí (znám pod zkratkou IoT), pro vývoj mobilních aplikací a bezserverové výpočty. Naopak webové služby SOAP nabízejí vestavěné zabezpečení a určitý soulad s transakcemi, jenž korespondují s podnikovými potřebami. [12]

7.3 Moderní API

V průběhu let bylo API označeno jako druh generického rozhraní pro připojení k aplikacím. V poslední době ale moderní API převzalo některé velmi užitečné vlastnosti, díky kterým je toto řešení velmi cenné a užitečné.

Moderní API dodržuje standardy, kde mezi nejpoužívanější spadá kombinace HTTP a REST. Tyto standardy jsou přívětivé pro vývojáře, snadno dostupné a hlavně srozumitelné. Jsou považovány spíše za produkty nežli za kód. Jsou určeny

k používání pro konkrétní skupiny, jakou jsou například vývojáři mobilních aplikací. Také jsou velmi dobře zdokumentovány a verzovány tak, aby je uživatelé mohli snadno využívat a měli určitý přehled o jejich údržbě a životním cyklu. [13]

8 Quadient

Mezinárodní firma Quadient Ltd. (Quadient Technologies Czech s.r.o.) se zaměřuje na Customer Communications Management (CCM), kdy hlavní vývojové centrum je v České republice. Firma se zaměřuje na organizace, které potřebují zjednodušit komunikaci se zákazníky a chtějí zefektivnit procesy produkující dokumenty. Quadient nabízí hned několik produktů, které ve vhodné kombinaci a správným nastavením dokážou vytvořit komunikační procesy zákazníkům dle jejich požadavků. Všechny zmiňované produkty spadají do rodiny Inspire a jsou vyvíjeny především v Hradci Králové.

8.1 Inspire Designer a Inspire Content Manager

První dva produkty jsou určeny ke správě komunikace se zákazníky z jediné centrální platformy. Mapují, spojují a zpracovávají data ze všech ostatních podnikových aplikací a systémů. Umožňují rychle vytvářet komunikační šablony z jediného designového rozhraní za pomoci jednoduchého systému přetažení (drag and drop). Je zde možnost kontroly verze obsahu a návrhů pro usnadnění spolupráce a přesného vytvoření komunikace. Inspire Designer poskytuje kompletní řešení pro správu komplexního datového prostředí. Dokáže extrahovat, transformovat a načítat data ze všech starších, ale i moderních aplikací a datových zdrojů jako jsou PDF, PostScript, AFP, Quark nebo InDesign. Umožňuje slučovat a vylepšovat datové zdroje a k tomu vytvářet responzivní HTML e-mailové šablony a tím personalizovat komunikaci. Velkou výhodou je rychlost a rozsah, který produkty zvládnout. Velké objemy fyzické a digitální komunikace nejsou žádný problém a výsledná variabilita je velkou výhodou. Obzvlášť v případě, kdy je potřeba připravit personalizované komunikační výstupy, jako jsou různá oznámení, všechny druhy korespondence nebo e-maily. [14]

8.2 Inspire Interactive

Tento produkt je především určen pro firemní uživatele, aby mohli snadno vytvářet korespondenci pro své zákazníky. Odstraňuje nákladné procesy související s IT a zajišťuje konzistentnost a shodu prostřednictvím vytváření, kontroly a schvalování komunikace z jediného prostředí. Pro tvorbu obsahu jsou využity předdefinované bloky obsahu. Veškerý obsah lze před odesláním zkontrolovat ve všech komunikačních kanálech, včetně e-mailu, SMS, HTML nebo mobilního oznámení.

Inspire Interactive také nabízí možnost konfigurace schvalovacích procesů, které zajistí, že veškerý obsah bude řádně zkontrolován a schválen ještě předtím, než bude řešení nasazené mezi zákazníky. Hlavní výhodou produktu je především možnost vytvářet komunikaci pro zákazníky snadno a rychle bez větších obtíží. Tím je zajištěna velká osobní úroveň a flexibilita. [15]

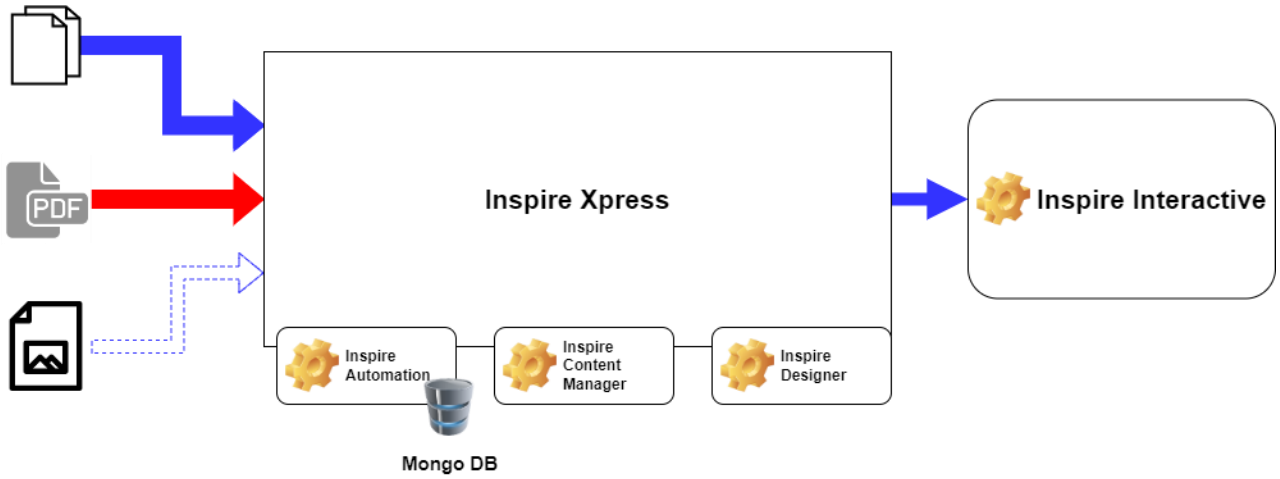
8.3 Inspire Automation

Dalším produktem je aplikace typu server-klient. Jeho produktivní částí je server, který přijímá nové úlohy a zpracovává je podle workflow konfigurace. Procesy na serveru běží buď automaticky anebo jsou vyzvány uživatelem. Na straně klienta produkt umožňuje spravovat server, prohlížet úlohy a konfigurovat pracovní postup. Pracovní postup lze konfigurovat pomocí uživatelského rozhraní (GUI), anebo pomocí příkazového řádku (CLI). Veškerá data, která jsou využívána produktem jsou uložena v databázi. Je zde široká podpora databází, mezi něž patří typy jako je MySQL, MS SQL, HSQL a Oracle.

Samotná logika zpracování je nastavena pomocí front. Fronta je základní strukturní jednotkou pracovního procesu. Výchozím bodem procesu je vstup, který přijímá úlohy a odesílá je dále do fronty ke zpracování. Každý vstup by měl mít následně alespoň jednu frontu, které lze mezi sebou různě napojovat. Fronta by také měla obsahovat další produkční moduly, kde mezi nejpoužívanějšími je takzvaný executor. V takovém modulu pak probíhají samotné operace. Jakmile úloha opustí frontu, je odeslána do jedné z následujících front pouze v případě, že jsou splněny vstupní podmínky té další fronty. Pokud není nalezena žádná odpovídající fronta nebo úloha neodpovídá požadovanému filtru, zůstane v aktuální frontě, dokud se

nesplní požadavky. Samotné fronty se využívání k definování logiky zpracování úloh.

8.4 Inspire Xpress



Obrázek 1 Jednoduché vyobrazení nástroje Inspire Xpress využívající produktů rodiny Inspire (Vlastní zpracování)

Inspire Xpress je nástroj, který využívá všech zmíněných produktů výše. Jednoduché znázornění můžete vidět na obrázku č. 1. Tento nástroj je určen převážně pro migraci dokumentů. Přístup je především založen na analýze obsahu, zpracování a optimalizaci. K tomu se využívá technologie umělé inteligence ve shodě s dalšími technologiemi. Samotný proces začíná čtením dokumentů a extrahuje jejich obsah. Dokumenty jsou rozřazeny dle daných kategorií uživatele a samotný obsah je rozdělen na bloky. V další fázi se z kategorií vytvářejí balíčky. Během zpracování se bloky a celé šablony mezi sebou porovnávají dle zadaných parametrů a provádí se konsolidace. Obecně platí že pokud se extrahují stovky nebo tisíce dokumentů, tento přístup snižuje množství obsahu až o 75 % a to pouze prostřednictvím konsolidace a optimalizace. Nástroj také dále identifikuje proměnné v obsahu. Po dokončení procesu, Inspire Xpress vytváří knihovnu extrahovaného obsahu, jako jsou bloky textu, nadpisy anebo třeba obrázky. Další úpravy se provádí v editoru, kde je možnost uspořádání, přejmenování a mnoho dalších operací s bloky a šablonami, které usnadňují výslednou optimalizaci.

Po dokončení optimalizace se obsah automaticky exportuje do Inspire Content Manageru a je poskytnut v aplikaci Inspire Interactive, což je základní vstup do zákaznického řešení. Jednoduše řešení Inspire Xpress dokáže efektivně migrovat

dokumenty a pomocí konsolidace redukuje požadované komponenty a zabraňuje zbytečným redundancím. To zajišťuje zvýšení provozní kvality a snížení provozních nákladů. [16]

9 Problematika migrace dokumentů

Migrace dokumentů je i dnes pro spoustu firem velmi klíčovou záležitostí, ať už potřebují přejít mezi produkty s existující strukturou, nebo přejít z papírového formátu na digitální. Vždy se jedná o velmi náročný a nákladný proces. V časových horizontech to může být pro menší firmy v řádech měsíců, pro větší firmy i několik roků. Firmy se tedy snaží najít možnosti, jak migrovat dokumenty co možná nejrychleji, nejlevněji a hlavně kvalitně.

Spousta podniků proto investuje prostředky, aby jim mohli poskytnout nástroje, které dokážou, co nejefektivněji přenášet dokumenty z jednoho systému do druhého. Problém nastává v bodě, kdy podniky mají spousty druhů dokumentů, ať se jedná o smlouvy, dopisy zákazníkům anebo třeba výpisy z účtů. To vše je vytvořené v určité podobě a pokud by se všechny tyto dokumenty měly znovu vytvářet ať už z fyzické nebo z té digitální podoby, zabralo by to mnoho času, peněz, a ještě by mohlo dojít k redundancím. Pokud k tomu připočteme vytvoření vhodného systému na míru, dle požadavků firmy jedná se v tu chvíli o velmi nákladnou záležitost s velmi dlouhým časovým horizontem.

Jak již bylo nastíněno dříve, dnes již spousta firem nabízí nástroje pro firmy, které ulehčují práci s dokumenty a nabízí mnoho způsobů, jak samotnou migraci dokumentů ulehčit, zlevnit, anebo případně alespoň uspíšit.

10 Nástroje a služby

Dnes existuje mnoho nástrojů a služeb, které dokážou efektivně zpracovávat text z grafických dokumentů. Dostupné nástroje mají různou kvalitu a existuje jich nepřeberné množství. Jelikož chceme naše řešení využívat i u zákazníků, je nutné, aby požadovaný nástroj splňoval všechny bezpečnostní kritéria a byl cenově přístupný. Nejlépe abychom mohli platit jen za službu, kterou opravdu využíváme. V neposlední řadě by měl být výstup co možná nejlepší a samozřejmě v takové struktuře, ve které bude dále využitelný v našem řešení. Pro účel porovnání a

výzkumu byly využity tři služby/nástroje od největších firem na trhu. Mezi tyto produkty spadá Amazon Textract od americké společnosti Amazon.com, Inc., dále Google Vision API od americké mezinárodní společnosti Google LLC a v neposlední řadě Microsoft Computer Vision od americké mezinárodní společnosti Microsoft Corporation.

10.1 Amazon Textract

Služba Amazon Textract od mezinárodní společnosti Amazon, je založená na strojovém učení a dokáže automaticky extrahovat text, ručně psaný text a data ze skenovaných dokumentů. Pro rozpoznávání, pochopení a extrahování textu v dokumentech, využívá již zmíněné optické rozpoznávání textu. Služba je také rozšířena o funkci získání dat z formulářů a tabulek.

Hlavní motivací vytvoření tohoto nástroje bylo usnadnit firmám extrakci dat z naskenovaných dokumentů typu PDF, obrázků, tabulek a formulářů. Firmy i dnes využívají jednoduchých nástrojů založených na optickém rozpoznávání znaků, ale většinou vyžadují speciální nastavení, které musí být pro každý dokument jedinečné.

Díky tomu že Amazon Textract využívá strojové učení, služba dokáže eliminovat potřebu manuálního nastavení pro každý dokument, a tím zjednodušit proces extrakci dat z dokumentů. Tímto způsobem tedy lze velmi zrychlit zpracování a získávání tolik potřebných dat. S předem vyškolenými modely strojového učení Amazon Textract není potřeba psát žádný kód pro extrakci dat. Modely byly vytrénovány na desítkách milionech dokumentů z mnoha průmyslových odvětví jako jsou faktury, účtenky, smlouvy, daňové doklady nebo pojistné dokumenty. [17]

10.2 Google Vision API

Podobně jako služba od Amazonu, i služba od Googlu je založená na strojovém učení a dokáže automaticky extrahovat text, ručně psaný text a data z naskenovaných dokumentů. Celkově Google Vision Api umožňuje vývojářům snadno integrovat funkce detekce vidění do aplikací, včetně označování obrázků, detekce obličejů, orientačních bodů anebo námi chtěné optické rozpoznávání znaků. Podobně jako

ostatní nástroje, i Google Vision podporuje mnoho formátů, jak obrázkového typu, tak i dokumentového. Mezi takové formáty patří JPEG, PNG, GIF nebo třeba TIFF. [18]

10.3 Microsoft Computer Vision

Služba Microsoft Computer Vision od společnosti Microsoft Corporation, nám dokáže poskytnout přístup k nejpokročilejším algoritmům, které zpracovávají obrázky nebo grafické dokumenty. Tyto algoritmy nám vracejí informace, které byly zjištěny na základě vizuálních funkcí a které nás zajímají. Mezi podporované grafické formáty spadají JPEG, PNG, GIF anebo třeba BMP.

Služba využívá optické rozpoznávání znaků a tím extrahuje text z obrázku. Pro extrahování tištěného a ručně psaného textu lze snadno využít jejich API. Samotné rozpoznávání znaků by k tomu samozřejmě nestačilo, tudíž služba dále využívá také modely založené na hlubokém učení, a pracuje s textem na různém povrchu a pozadích. Mezi dokumenty, který tento nástroj dokáže zpracovat jsou obchodní dokumenty, faktury, účtenky, plakáty, vizitky, dopisy a třeba i tabule. Jejich OCR API podporují také extrahování dokumentů v několika jazycích.

Služba Microsoft Computer Vision nabízí také celou řadu dalších funkcí, jako jsou scénáře správy digitálních aktiv (Digital Asset Management – DAM). Jedná se o proces organizování, ukládání a získávání multimediálních aktiv, správy digitálních práv a oprávnění. Společnost může například chtít seskupit a identifikovat obrázky na základě viditelných a známých log firem+, nebo tváře objektů, barev a dalších poznávacích znaků. Umožňuje také například automaticky generovat popisky pro obrázky a připojit klíčová slova, aby v nich bylo možné vyhledávat.

11 Porovnání nástrojů a jejich analýza

Všechny zmíněné nástroje se jeví jako zajímavé možnosti, pro naši finální realizaci. Je nutné zjistit, který nástroj bude nejvíce kompatibilní s našimi požadavky. Ty jsou potřeba správně vydefinovat tak, abychom došli ke správnému výsledku a aby finální implementace proběhla co možná nejlépe a nejefektivněji.

11.1 Specifikace

Pro správné vydefinování specifikací a požadavků, bylo nutné komplexního pochopení fungování produktu Inspire Xpress. Jak již bylo nastíněno, Inspire Xpress využívá další produkty rodiny Inspire a tudíž je potřeba si uvědomit, že veškeré implementace nástrojů budou případně probíhat v Inspire Automation, kde bude nutné veškeré potřebné knihovny naimportovat a vytvořit spojení s nástroji tak, aby se odtud dali volat veškeré potřebné API a tím získat žádoucí výstupy. Dle toho lze vyspecifikovat první požadavek, a to, aby šel nástroj jednoduše naimplementovat s existujícím řešením, spravovat a také udržovat.

Další specifikací, se kterou je nutné počítat je cena samotných nástrojů. V analýze bude potřeba zohlednit, jak se nástroje účtují. Můžou zde být poplatky za užívání, za počet požadavků na API, nebo za další nástroje a služby. Je také možné že tyto nástroje půjdou zakoupit jako služba a budou fungovat v jasně daném měsíčním tarifu, kde budou přesně definované hodnoty využitelnosti různých řešení.

S třetí a neméně důležitou specifikací, se kterou je třeba se vypořádat, je samotné řešení. Také je potřebné zjistit, zda nám nástroj poskytuje, to, co od nástroje očekáváme. Nástroj může mít veškeré pozitivní vlastnosti, jako je jeho jednoduchost nebo cena. Je však také nutné, aby nástroj fungoval tak, jak potřebujeme my. Proto je zapotřebí získat z každého nástroje výstupy pro jasně definované testovací dokumenty a tyto výstupy zanalyzovat. Pokud se ukáže že nástroj funguje správně, a výstupní data jsou ve správném formátu, pak by nic nebránilo finální implementaci do finálního nástroje. V našem případě tedy očekáváme implementaci s nástrojem Inspire Xpress.

11.2 Testovací dokumenty

Pro správné otestování nástrojů, bylo vytvořeno pět základních dokumentů. V prvním dokumentu máme čistě anglický text, bez jakýchkoliv obrázků, sloupců a dalších věcí, které by pro zpracování mohli být problematické. Tento dokument je uložen ve formátu PNG. Jedná se o grafický formát, který by měl být se všemi nástroji plně kompatibilní.

Druhý testovací dokument je stejný dokument, pouze uložen ve formátu PDF, přesněji, obsah dokumentu PDF je předešlý grafický obrázek. Tudíž z tohoto testovacího dokumentu není možnost nijak zkopírovat text.

Třetí a již komplexnější testovací dokument, obsahuje také anglický text. Text je formátován do dvou sloupců a obsahuje i reklamní obrázek. Podobně jako první testovací dokument, i tento bude uložen v grafickém formátu PNG. Tento třetí testovací dokument má hlouběji otestovat, jak komplexně dokážou nástroje zanalyzovat a rozpoznat text a případně připravit po další zpracování. Také bude zajímavé sledovat, co se děje s obrázkem v dokumentu, zda se nástroje pokusí rozpoznat text i z této oblasti dokumentu, nebo zda dokáže rozpoznat, že se jedná o obrázek.

Čtvrtý testovací dokument je také komplexnějšího typu a obsahuje problematické prvky jako jsou odrážky, reklamní oblast na levé straně obsahující text ale především také velmi problematickou část známou jako tabulka. Dokument stejně jako již dva předchozí dokumenty je uložen ve formátu PNG.

Poslední testovací dokument je stejně tak jako první dokument velmi chudý, obsahuje pouze jednoduše strukturovaný text, bez jakýchkoliv problematických prvků a je samozřejmě uložen v grafickém formátu PNG. Jedinou změnou je jazyk dokumentu. Testovací jazyk je kompletně v českém jazyce se všemi náležitostmi, jakou jsou diakritická a interpunkční znaménka.

Nutno podotknout, že všechny dokumenty byli vytvořeny ručně v nástroji Microsoft Word a poté převedeny do požadovaných formátů.

11.3 Analýza a její implementace

Abychom mohli dosáhnout kvalitní analýzy, bylo třeba nástroj důkladně prozkoumat a získat potřebnou znalost každého nástroje zvlášť. Také bylo potřeba se rozhodnout jaké metody využívat a celkově vystavět řešení tak, abychom mohli snadno dokumenty testovat. U všech nástrojů byl k dispozici účet zdarma na vyzkoušení nástroje. Nástroje byli omezeny počtem požadavků a některé i časově. Pro naše testování v řádech jednotek dokumentů, byly tyto účty plně dostačující, což, pro finální řešení samozřejmě již dostačující být nemusí.

Pro vytvoření testovací implementace, bylo využito nástroje IntelliJ IDEA a programovacího jazyku Java. Pro správné fungování byla vytvořena jednoduchá logika využívající tzv. „hot folder“. Jedná se o vytvoření složky na disku, která je sledována našim kódem. Byl tedy vytvořen kód, který aktivuje akci na sledování složky a pokud je vložen dokument, je vyvolán proces na analýzu a zpracování dokumentů. Abychom měli jistotu, že budeme zpracovávat všechny dokumenty které chceme, jsou dokumenty brány z jiné složky, než je naše „hot folder“. Pro tyto účely byla vytvořena jednoduchá struktura složek na disku, kde složka s názvem „Input“ obsahuje dokumenty, které chceme zpracovávat. Dále složka InputTrigger, je složka, do které poté, co máme všechny dokumenty připravené ke zpracování, vkládáme dokument a dáváme tím signál na zpracování dokumentů, jedná se tedy o naší „hot folder“. Poslední složkou je složka Output, do které jsou ukládány výsledné dokumenty.

11.4 Amazon Textract

První nástroj, který byl zvolen pro analýzu a otestování, byl již několikrát zmiňovaný Amazon Textract. Abychom mohli v tomto nástroji analyzovat dokumenty, bylo nutné si detailně nastudovat jejich dokumentaci a poté zvolit správné metody pro testování našich dokumentů. Důležitou podmínkou je také samotné uchopení výsledných výstupů.

Jak již bylo nastíněno výše, bylo využito testovacího účtu zdarma. Pro zprovoznění Textractu, bylo nutné vykonat několik kroků. V prvé řadě bylo potřeba zjistit, která operace Amazonu Textract bude nejlepší pro řešení naší problematiky. Amazon má velice obsáhlou dokumentaci, která se nám snaží ukázat správnou cestu k implementaci. Nástroj nám poskytuje několik operací na zpracování dokumentů. Na výběr máme mezi synchronními a asynchronními operacemi. Synchronní operace jsou určeny především pro jednostránkové dokumenty ve formátu JPEG, PNG, PDF a TIFF. Tyto operace mají výhodu v tom, že vrací všechny výsledky téměř v reálném čase. Oproti tomu asynchronní operace dokáže detekovat a analyzovat text ve vícestránkových dokumentech, které jsou ve formátu PDF nebo TIFF. Například zpracování PDF dokumentu s více než 1000 stránkami může zabrat spoustu času. K tomu slouží asynchronní zpracování, které umožňuje aplikaci

pokračovat v úkolech, zatím co se čeká na dokončení operace. Jak již bylo zmíněno, asynchronní operace jsou především pro vícestránkové dokumenty ve formátu PDF nebo TIFF, ale dokážou zpracovat i jednostránkové dokumenty ve formátu JPEG, PNG, TIFF nebo PDF.

Pro naše řešení byla zvolena cesta synchronní operace. Jelikož naše testovací dokumenty jsou jednostránkové a výsledek chceme mít okamžitě k dispozici. Amazon Textract nabízí hned 4 druhy synchronních operací.

První z nich je analýza dokladů totožnosti, jako jsou například řidičské průkazy. Pomocí této operace mohou podniky velice snadno a rychle extrahovat informace z dokladů. Další operací je Analýza faktur a účtenek. Amazon Textract extrahuje relevantní data, jako jsou kontaktní informace, zakoupené položky, anebo třeba jméno dodavatele, aniž by bylo potřeba jakýchkoliv šablon nebo konfigurace. Faktury a účtenky často využívají různé rozvržení, takže je obtížné a časově náročné ručně extrahovat data ve velkém měřítku. K pochopení celkového kontextu faktur nebo účtenek je využito strojového učení a díky tomu lze extrahovat data jako je například datum faktury, ceny položek, anebo také číslo dokumentů. V pořadí třetí operací je detekce textu. Tato operace pouze vrací text zjištěný ve vstupním dokumentu. A poslední nabízenou operací je analýza textu v dokumentu. Tato operace dokáže vracet tři různé extrakce dokumentů a těmi jsou text, formuláře a tabulky. Všechny operace vrací výstupy ve formátu JSON.

11.4.1 Testovací implementace

Jak již bylo popsáno výše, pro naše testování aplikace jsme využili synchronní operace, přesněji operaci analýzy textu v dokumentu. Ale aby mohlo dojít k testování našich testovacích dokumentů, bylo nutné nejdříve vytvořit testovací účet pro každý z nástroj. K tomu bylo třeba dále vytvořit testovací implementaci, která by se měla dále využívat v reálném řešení.

Samotný proces zpracování se skládá z několika kroků. První je vytvoření instance do nástroje Textract. Dále se volá metoda na jeho spuštění, kde se zároveň služba nastaví a spustí. Vzniká nám tedy klient. Klient může využívat téměř všechny metody, které nám jsou Textractem nabízeny. V tu chvíli přecházíme k samotnému zpracování našich vložených dokumentů. Vstupní dokumenty jsou převedeny na

pole bytů a ty jsou následně analyzovány naší zvolenou operací. Výsledky jsou zapsány do pole a může se zpracovávat další dokument. Po zpracování všech vložených dokumentů je klient do Textractu ukončen.

Dalším krokem je samotné zpracování výsledků a vytvoření výstupního dokumentu. Pro testování bylo zvoleno vytvoření jednoduchého DOCX dokumentu, ve kterým budeme moci snadno upravovat náš získaný text. Dále začneme procházet výsledky z Textractu. Po analýze výstupu víme, že nás pro toto řešení budou zajímat bloky textu pod klíčovým typem „LINE“. Jedná se o výstup ze synchronní operace. Tento výstup je určitý blok textu, který je na jednom řádku. Výstupní data ve formátu JSON se tedy prochází a z každého záznamu typu „LINE“ získáváme text a tento text zapisujeme do našeho výstupního dokumentu. Všechny výstupní dokumenty se ukládají na disk, do předem stanovené složky. Po zpracování všech záznamů, se celý proces ukončuje a vyvolává se znovu metoda, na sledování naší složky, zda do ní není vložen dokument, který nám dá znamení na další zpracování dokumentů.

11.4.2 Analýza výstupů

Proces zpracování souborů v nástroji Amazonu Textract proběhl úspěšně. Všech pět souborů se povedlo zprocesovat a analyzovat. Výsledná analýza výstupů nám ukázala, že nástroj dokáže efektivně zpracovávat anglické texty a jejich přesnost je téměř bezchybná. Oproti tomu české texty nebyli tak dokonalé. Některým slovům byla zaměněna písmena jako například písmeno „v“ za „y“ ve slově „prvními“. Dalším vážným nedostatkem byla diakritické znaménka, která byla buď úplně ignorována, nebo zaměněna za jiné. Je to určitě dáno tím, že Amazon Textract podporuje šest jazyků, přesněji angličtinu, francouzštinu, španělštinu, italštinu, němčinu a portugalštinu.

Dalším nedostatkem, se kterým jsme se mohli u nástroje setkat bylo samotné zpracování. Jak již bylo popsáno výše, výstupní soubor z Textractu, ve formátu JSON procesujeme. Získáváme v tomto souboru podrobnou strukturu o stránkách, co se na nich nachází a kde. Bohužel z této struktury dokážeme získat pouze text po řádcích, a tudíž je takto zapsán i do výstupního dokumentu. Máme tedy soubor, kde jednoznačně nelze určit odstavce ani větší bloky textu.

Posledním velmi výrazným nedostatkem je, že není dodržen tok textu v dokumentu obsahující text ve sloupcích, případně v dalším dokumentu se sdělením na okraji stránky, tudíž se text stává nesrozumitelným, případně se ztrácí kontext obsahu.

11.5 Google Vision API

Druhým nástrojem, který byl zvolen pro analýzu a otestování byl již jmenovaný Google Vision. Podobně jako tomu bylo u předchozího nástroje Amazon Textract, bylo nutné si nastudovat dokumentaci a poté zvolit správné metody, které by dokázaly otestovat naše dokumenty a získat uspokojující výstup.

Bylo využito účtu zdarma, byl tedy prostor si vyzkoušet veškeré funkcionality, které jsou poskytovány. Nástroj nabízí spoustu operací, jako je například detekce obličejů, log, objektů a mnoho dalších.

Nás zajímala především část o optickém rozpoznávání znaků, kde jsou hned tři druhy metod. První metodou je detekce znaků v obrázku. Tato metoda dokáže získat text z grafických souborů, jako je JPEG nebo PNG. Jedná se o synchronní operaci, tak že výsledky máme okamžitě k dispozici. Druhá metoda nazývající se detekce ručně psaného textu v obrázku, dokáže detekovat ručně psaný text v grafických souborech. Funguje podobně jako první metoda, je tedy synchronní a výsledky jsou poskytnuty okamžitě. Zpracovává také stejné formáty dokumentů. Třetí metoda detekování textu v dokumentu, přesněji v dokumentech ve formátu PDF a TIFF, je metoda asynchronní. Výsledky v tu chvíli nejsou poskytnuty okamžitě, ale je nutné si na ně určitý čas počkat.

Všechny výstupy z nástroje jsou ve formě JSON, je tedy nutné stejně tak, jako u Amazonu Textract, podrobně prozkoumat a získat zpracováním požadovaný výstup.

11.5.1 Testovací implementace

Pro testovací implementaci jsme použili první a třetí metodu. Tedy detekování textu z obrázku a detektování textu z dokumentu, byť se jedná o asynchronní operaci. Oproti Amazonu Textract, byla tato metoda velmi snadno implementovatelná tak, že jsme tuto metodu zahrnuli do procesu. To nám umožnilo analyzovat grafické soubory, ale i PDF a TIFF soubory. Díky tomu jsme byli schopni zpracovat veškeré

námi vytvořené testovací soubory. Samotné zpracování je tvořeno zjišťováním, zda se jedná o PDF dokument, pokud ano, spustí se asynchronní operace, která využívá Google Storage. Jedná se o cloudové úložiště vytvořené pro velké soubory a hodí se spíše pro firmy. Kombinuje výkon a škálovatelnost s pokročilými možnostmi sdílení a zabezpečením. Například u společnosti Amazon najdeme ekvivalent k tomuto úložišti pod názvem Amazon S3.

Proces začíná inicializací, nahráním vstupních dokumentů a poté se spustí samotná operace, která začne dokument zpracovávat. Proces čeká, dokud operace není ukončena. Poté je ve výstupní složce v Google Storage uložený výstupní JSON. Proces tento výstup vezme a začne zpracovávat. Klíčové elementy jsou „paragraphs“, „words“ a „detectedBreak“. První element nám říká, kolik má stránka odstavců. „Words“ nám říkají, jaké slova odstavec obsahuje a poslední element „detectedBreak“ nám říká, co za mezeru je mezi slovy. Tímto způsobem dokážeme získat text z celé stránky. Výsledné texty jsou uloženy do dokumentu. Stejně jako tomu je u Amazonu.

Synchronní operace detekce textu v obrázku funguje podobně, jen nevyužívá Google Storage k ukládání vstupů a výstupů. Vezme tedy soubor přímo ze vstupní složky a začne zpracovávání, kdy téměř okamžitě je vrácen výsledek ve formě JSON. Zbytek procesu je stejný jako v předchozím případě. Oba procesy jsou nakonec zpracovány tak, že vzniká do výstupní složky soubor ve formátu DOCX.

11.5.2 Analýza výstupů

Zpracování všech testovacích souborů proběhlo, i díky asynchronní operaci úspěšně. Všechny soubory byly přečteny dle očekávání. Výsledné výstupy obsahovaly text, který byl předložen v testovacích souborech. Dokonce i u textů v českém jazyce lze usoudit, že rozpoznání textů bylo bezchybné. Oproti předchozí analýze, kde chyběla diakritická znaménka je poznat, že Google Vision podporuje i jiné jazyky.

Další věcí, kterou je nutno ocenit je jednoduchost použití nástroje a jeho možné propojení s ostatními nástroji, především s Google Storage.

Jak již vyplynulo z předchozích odstavců, výstupní formát JSON z testovaného nástroje nám poskytl rozdělení odstavců na stránce. Díky tomu jsme byli schopni

vytvořit potřebné bloky textu, chceme-li odstavce. Vznikl nám tedy dokument s jasně rozdělenými bloky textu, místo pouze řádkově vypsaného textu. Lze také jasně definovat mezery mezi slovy a odstavci.

Velmi dobře si nástroj také poradil s dokumenty, jež obsahovaly jiný tok textu než jeden souvislý. Jednalo se přesněji o třetí a čtvrtý testovací dokumenty, kde první jmenovaný má dvousloupcové rozložení textu a druhý určité reklamní sdělení na levé straně stránky. Text byl zpracován postupně, tak jak by čtenář očekával. Je tedy zachován správný tok textu a texty se nemíchají do sebe. Tabulka dle očekávání nebyla nijak zpracována a jen obsah tabulky je zapsán po slovech do výsledného dokumentu.

11.6 Microsoft Computer Vision

Posledním testovaným nástrojem byl zvolen již zmiňovaný Microsoft Computer Vision. Podobně jako u předchozích dvou nástrojů, bylo nutné nastudovat dokumentaci a poté zvolit správné metody pro testování připravených dokumentů. Jak jsme se mohli domnívat, nástroj funguje velmi podobně jako výše zmíněné nástroje.

Znovu bylo využito testovacího účtu zdarma, který nabízí kredit a možnost si vyzkoušet mnoho nástrojů zdarma. Oproti předchozím nástrojům, vybrání správné metody na zpracování našich vstupních dokumentů nebylo vůbec těžké. Ve své podstatě je nabízena pouze jedna metoda, která dokáže zpracovat ručně psaný text, tištěný text, grafické soubory, klasické soubory, a to hned v několika různých jazycích. Pracuje se stejnými soubory jako předchozí nástroje. Podporuje tedy soubory ve formátu JPEG, PNG, BMP, PDF anebo TIFF. Pro poslední dva zmíněné formáty, jsou zde jasné omezení do 2000 stránek. Pro námi využitý testovací účet to bylo omezeno pouze na dvě stránky, což nás díky zvoleným testovacím dokumentům nijak neomezovalo. Dalším omezením byla velikost obrázků, která musela být menší jak 500 MB. Pro testovací účet znovu mnohem větší omezení a to pouze 4 MB. Jedná se o asynchronní operaci, je nutné tedy počkat na výsledek. To vše je obsaženo v základním kódu, který je poskytnut v dokumentaci. Nutno dodat že lze využít tzv. on-premises řešení, tedy místní nasazení s využitím docker kontejneru, který běží na lokálním prostředí a jedná se tedy o skvělé řešení pro

specifické požadavky na zabezpečení a správu dat. To by určitě uvítali zákazníci, kteří jsou citliví na svoje data, jako jsou banky a podobné organizace.

11.6.1 Testovací implementace

Byla využita tedy asynchronní operace, která dokáže veškeré analýzy velmi rychle zpracovat a získat výstupní data. Předtím ale bylo nutné si založit testovací účet a začlenit poskytnutý kód do naší testovací implementace. Jako tomu je u našich dvou nástrojů, i zde musela proběhnout inicializace, jež vyžadovala celkem dva parametry a poté vše fungovalo. S drobnými úpravami bylo brzo dosaženo potřebného výsledku, díky čemuž byl tento nástroj nejjednodušší na vytvoření funkční testovací implementace. Nutno podotknout že určité části již byli předpřipravené z předchozích nástrojů a jen znovu využity. U samotného zpracování výsledků, nebylo toho moc na řešení, struktura nám opět dovolila vzít text pouze ve formě řádků textu, jako tomu bylo u prvního testovaného nástroje.

11.6.2 Analýza výstupů

Podobně jako u předchozích nástrojů, u všech pěti souborů proběhlo zpracování úspěšně. Vzniklo nám pět výstupních souborů, které na první pohled vypadají totožně s výstupy z nástroje Amazon Textract. Po hlubším prozkoumání lze dojít k tomu, že nástroj od Microsoftu lépe zpracoval český text, kdy nechybí diakritika. Bohužel, podobně jako je to už již zmiňovaného soupeře, je text pouze v řádkové formě a nelze určit odstavce ani větší bloky textu. Co se týče zpracování toku textu, u testovacího souboru s dvousloupcovým rozložením. Text nebyl zpracován po sloupcích, nýbrž po řádcích, tudíž výsledek je velmi podobný tomu, jako je to u prvního zmiňovaného nástroje, kdy text se stává zmatečný a nesrozumitelný.

11.7 Vyhodnocení testů

V této kapitole shrneme, jakých výstupů a výsledků jsme naším testováním dosáhli a poté se podíváme i na další aspekty, které hrají roli, a to přesněji na implementaci a cenu jednotlivých nástrojů.

Jak je již zřejmé z výše provedených a popsaných analýz ke každému nástroji, výsledky prvního a třetího nástroje, tedy Amazonu Textract a Azure Computer

Vision, byly velmi podobné. Zpracovaný text byl formou řádků a nešlo vytvořit logické bloky textu, chceme-li odstavce. Dále u více sloupcového dokumentu, jako je třetí vstupní dokument nebo u čtvrtého testovacího dokumentu s reklamním sdělením na levé straně, nebyl text nijak oddělen. Byl přečten nastejno s textem v hlavním toku textu, tudíž byl do výsledného textu zamíchán a text přestal dávat smysl, nebo se stal značně hůře čitelným. U pátého testovacího dokumentu měl třetí nástroj navrch, a to přesněji v případě zpracování českého textu, kdy v textu nechyběla diakritická znaménka. Nejlépe z testovaných nástrojů přesto vyšel náš druhý testovaný nástroj, tedy Google Vision API. Tento nástroj jako jediný dokázal zpracovat texty tak, že našel mezery mezi odstavci, a dokázal zpracoval celé bloky textu odpovídající odstavcům v testovacích dokumentech. Také prokázal schopnost zachovat tok textu, jak u dvousloupcového dokumentu, tak i u dokumentu s reklamním sdělením na straně. Výstup tedy odpovídá tomu, co bychom od těchto nástrojů očekávali. Text je jasně rozčleněn a seřazen tak, jak je tok textu v dokumentu nastaven. Nástroj ovšem taky není zcela bezchybný a pokud nám například text přetékal do druhého sloupce, logicky ho není schopen spojit s předchozím odstavcem.

Dále je nutno zdůraznit, že žádný z nástrojů si nedokázal poradit s tabulkou. K tomuto problému nám Amazon nabízel jinou metodu, která by dokázala data získat. Vzhledem k tomu že nám ale nejde o samotné získání dat, ale spíše získání logiky a obsahu, nebyla tato metoda použita. Všechny nástroje tabulku přečetli jako text, tudíž výsledek v této části máme všude stejný, kdy na každém řádku ve výstupním dokumentu máme zapsaný obsah jedné buňky z tabulky. Nejkvalitnějším nástrojem pro naše řešení se tedy jeví Google Vision API.

Dalším aspektem je samotná implementace. U všech nástrojů se využívají maven závislost. Jednoduše řečeno tedy stačilo vložit požadované závislosti a o zbytek se postaralo vývojové prostředí. V něm byla vytvořena testovací implementace, která jak již bylo nastíněno dříve v naší práci, využívá systému složek, včetně využití tzv. „hot folder“ složky. Všechny nástroje jsme spouštěli na stejném základu a rovnou zjistili, jak půjdou zapracovat. Nutno říct, že nejjednodušší implementaci měl náš třetí testovaný nástroj, tedy Azure Computer Vision. Dokumentace byla velmi přehledně vytvořená a pomocí jasně napsaných kroků lze velmi snadno

nástroj uvést do provozu. Předpřipravený kód byl velmi snadno uchopitelný a velmi dobře čitelný. Druhý nástroj Google Vision API byl taky velmi rychle spustitelný. Předpřipravené metody byly velmi jednoduše rozčleněny. Bylo však nutné připravit proměnnou do prostředí systému, na kterou se kód odkazoval pro připojení se s účtem na platformě Google. Oproti Azure, který potřeboval jednu maven závislost, Google potřeboval hned tři pro náš účel. Přesto samotná implementace byla velmi jednoduchá. V zásadě bylo jen nutné si ze seznamu metod vybrat ty, které potřebujeme pro naše řešení, a poté je pouze vzít a s mírnou úpravou vložit do naší implementace.

Nejsložitějším na implementaci se zdál být náš první testovaný nástroj. Stránky se zdály velmi nepřehledné. Dostat se k samotné implementaci a k potřebným zdrojům bylo velmi uživatelsky nepřívětivé. Než jsme se dostali ke správné implementaci nástroje, bylo navštíveno mnoho uliček, které buď vedly jinam nebo byly slepé. K samotné implementaci, bylo potřeba vytvořit hned tři proměnné do prostředí systému a vložit hned několik maven závislostí. Samotný kód pro zprocesování dokumentů tak složitý nebyl a dal by se srovnat se třetím testovaným nástrojem. Rád bych tu podotkl, že to může být pouze subjektivní pocit, ale Amazon Textract vyšel ve výsledku jako nejsložitější nástroj na implementaci a zprovoznění. Přesto lze říct, že všechny nástroje mají podobnou implementaci a nejsou zde propastné rozdíly. Proto bych tomuto parametru přisoudil nejmenší prioritu, případně jako malé plus pro třetí testovaný a malé mínus pro první testovaný nástroj.

Posledním aspektem v naší analýze je cena nástrojů. Zde se zdá, že se společnosti navzájem hlídají a ceny jsou velmi podobné. Jak nastiňuje tabulka č. 1, můžeme vidět, že jsou na tom nástroje téměř identicky.

	Cena za 1000 transakcí			
	0-1000 t/zm	1001+ t/zm	1M+ t/zm	5M+ t/zm
Amazon Textract	$1.5	$1.5	$0.60	$0.60
Google Vision API	Free	$1.5	$1.5	$0.60
Azure Computer Vision	$1.5	$1.5	$0.60	$0.60

Tabulka 1 Porovnání cen mezi nástroji (Vlastní zpracování)
**t/zm – transakce/za měsíc*

Zatímco co Amazon Textract i Azure Computer Vision za prvních 1000 dokumentů chtějí zaplatit, tak v tomto případě by se jednalo o jeden dolar. Google Vision API nabízí prvních 1000 dokumentů v měsíci zdarma. Cena je dále u všech produktů stejná. Například za 4000 dokumentů bychom si shodně u každého nástroje zaplatili po šesti dolarech. Od milionu dokumentů, bychom se u prvních dvou zmiňovaných produktů dostali na cenu 0.60 dolaru za 1000 dokumentů. Google se na tuto cenovou hladinu připojuje až od pěti milionů dokumentů výše.

Jelikož se zatím naše předpokládané využití, nepřibližuje ani tisícům dokumentů měsíčně, případně určitě ne milionům, vychází nám nejlépe s cenovou politikou Google Vision API. Zde je nutné podotknout, že pokud chceme tento nástroj využít i pro PDF dokumenty, je nutné k tomu využít Google Storage, který je do 2 GB plně zdarma. Neměl by tedy v případě využívání a průběžného odstraňování obsahu nijak pozměnit výslednou částku.

Z dosažených výsledků a vzhledem k potřebě mít obsah rozdělen do logických celků, tedy do jasně daných odstavců, nám i s přihlédnutím k ostatním aspektům vychází nejlépe druhý testovaný nástroj, tedy Google Vision API. Nástroj prokázal potřebné kvality a měl by být použit ve finální implementaci do nástroje Inspire Xpress.

12 Proces implementace

Hlavní motivací k vytvoření nové možnosti zpracování dokumentů, jsou především koncový zákazníci. Pokud zákazník chce migrovat dokumenty, které jsou v grafické formě, neexistuje jednoduchý způsob, jak jim jednoduše tyto dokumenty zpracovat a poskytnout. Jedinou možností je ruční přepsání dokumentů, nebo využití optického rozpoznání textu. Z naší analýzy vyplynulo, že vhodným nástrojem pro optické rozpoznání textu a jeho následné využití v nástroji Inspire Xpress je Google Vision API, který zpracovává text do logických celků a dokáže si poradit i s náročnější strukturou dokumentu v různých jazycích.

Další funkcionalitu, kterou budeme v naší implementaci zpracovávat je automatické pojmenovávání bloků textu. Na výstupu máme bloky textu, kterých může být i několik tisíc a potřebujeme je nějak rozlišit. Pro takové množství bloků, není příliš efektivní využívat generické názvy bloků. Proto bylo nutné najít alternativu, která by nám na základě obsahu dokázala vygenerovat název. Podle tohoto názvu bychom

měli ihned poznat, co je obsahem výsledného bloku. Na to nám poslouží RAKE (Rapid Automatic Keyword Extraction), což je algoritmus pro extrakci klíčových slov z dokumentu, která mají nejvyšší relevanci nebo důležitost pro obsah dokumentu. [19]

Poslední funkcionalitu, kterou využijeme je ohodnocení textu. Přesněji se bude jednat o takzvaný Flesch-Kincaid test čitelnosti. Je určen ke stanovení hodnoty o srozumitelnosti textu a tím poskytuje cenné informace o obsahu našeho dokumentu. To následně určuje, jak je dokument čitelný pro koncové zákazníky a zda mu vůbec porozumí. Flesch-Kincaid obsahuje dvě hodnoty, které srozumitelnost textu ohodnocují. První z nich je Flesch Reading Ease. [20] Hodnota se počítá pomocí vzorce uvedeného na obrázku č. 2, kde se kombinují jasně dané konstanty s počtem slov, slabik, a vět.

$$206.835 - 1.015\left(\frac{\text{počet slov}}{\text{počet vět}}\right) - 84.6\left(\frac{\text{počet slabik}}{\text{počet slov}}\right)$$

Obrázek 2: Vzorec pro výpočet Flesch Reading Ease (Vlastní zpracování)

Skóre se pohybuje většinou mezi 0 až 100. Krátká slova a krátké věty dostávají menší skóre, oproti tomu delší věty s delšími slovy dostávají naopak větší skóre. Pokud by byla výsledná hodnota například 100, znamená to, že je věta velmi jednoduchá a text je velmi jednoduše čitelný. Naopak nižší hodnocení znamená, že je text těžší na čtení. Celkově se podle této hodnoty dá říct, pro jaké čtenáře je text určen.

Druhou hodnotou je Flesch Grade Level. [20] Jedná se o široce využívaný vzorec čitelnosti uvedený na obrázku č. 3, který hodnotí přibližnou úroveň čtení textu. Dříve bylo nutné převést Flesch Reading Ease pomocí tabulky, k určení úrovně čtení. To nyní zastupuje tato hodnota. Určuje se od hodnoty 0 a více. Většinou bychom se měli držet do hodnoty 12 ale může nabývat i vyšších hodnot. Pokud má text úroveň 8, znamená to, že čtenář potřebuje úroveň čtení 8 a vyšší, aby textu porozuměl.

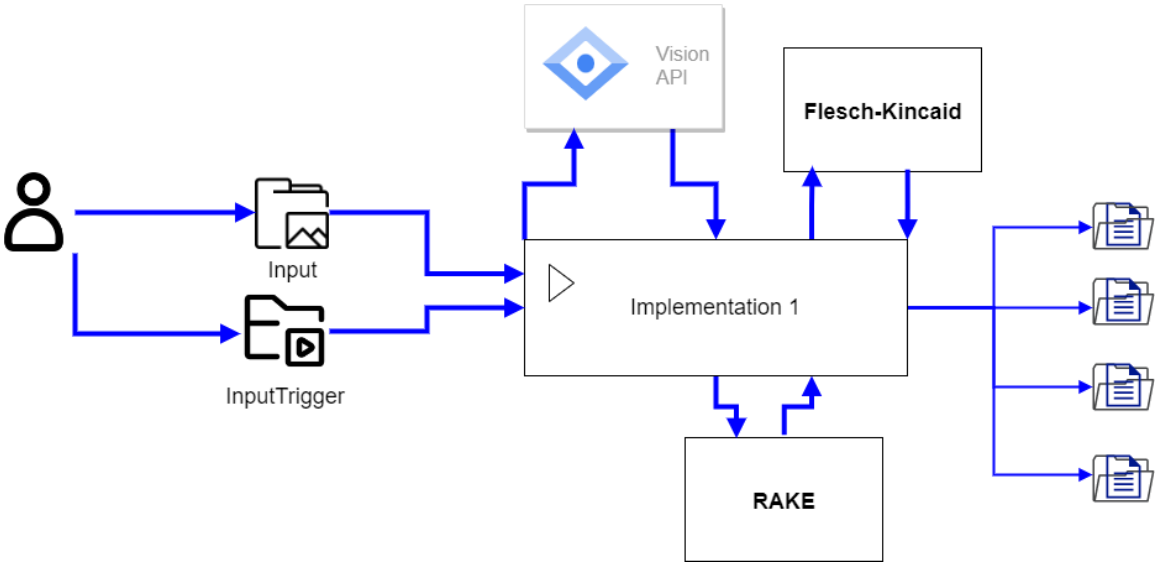
$$0.39\left(\frac{\text{počet slov}}{\text{počet vět}}\right) + 11.8\left(\frac{\text{počet slabik}}{\text{počet slov}}\right) - 15.59$$

Obrázek 3: Vzorec pro výpočet Flesch Grade Level (Vlastní zpracování)

Samotnou implementaci lze rozdělit na dvě části, kde obě staví na vytvořeném testovacím skriptu. První část generuje dokumenty, do podoby DOCX. Tato implementace je určena především k vytvoření souvislého řešení, jehož části budou využity v druhé implementaci. Taky zde bude ukázka DOCX dokumentů, které poskytnou výsledný text s pojmenovanými bloky. Každý blok textu bude pojmenován dle obsahu pomocí knihovny Rake. Tyto výsledné dokumenty je pak možné také dál upravovat a případně využívat. V poslední části budou dokumenty roztříděny dle hodnot Flesch Reading Ease a Flesch Grade Level. Pro tento účel je nutné vypočítat jednu hodnotu, která bude průnikem hodnot z testů čitelnosti.

Druhá část implementace je zaměřená na reálné využití naší práce. Jak již bylo zmíněno, jedná se o napojení řešení k nástroji Inspire Xpress, díky čemuž získáme zcela nový a ojedinělý přístup ke zpracování grafických dokumentů. Nástroj by měl být také rozšířen o již zmiňované funkcionality jako je automatické pojmenovávání bloků a využití testu čitelnosti. V nástroji by tento test měl být využit jak pro každý dokument, tak také pro každý blok. Nutné je také dodat, že aby tato část řešení fungovala správně, bylo potřeba detailně poznat fungování nástroje Inspire Xpress.

12.1 První část implementace



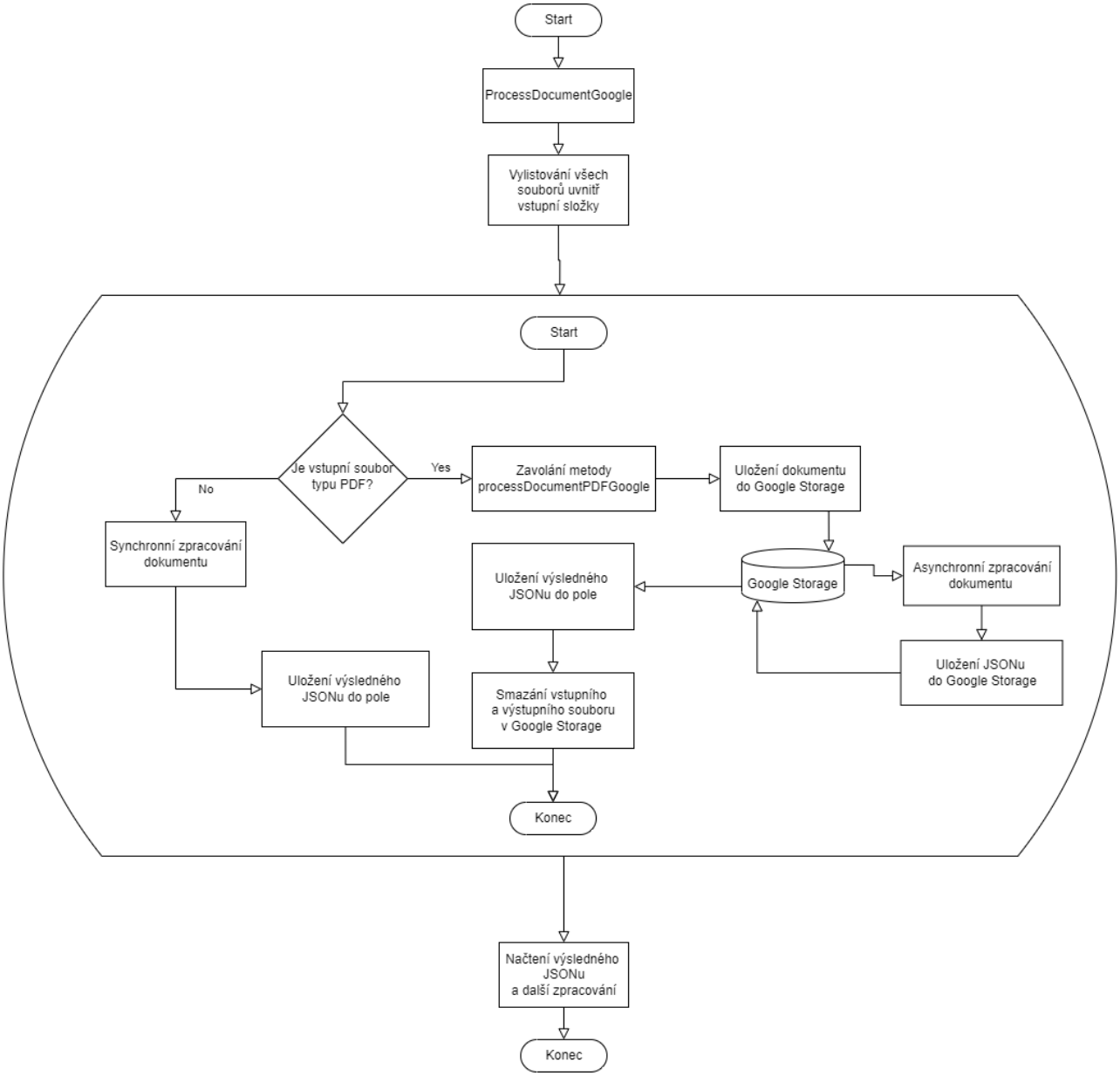
Obrázek 4 jednoduché schéma první implementace (Vlastní zpracování)

Jak již bylo nastíněno výše, první část je založena na již testovacím kódu, který je rozšířen o další velmi užitečné funkcionality. Jednoduché schéma na obrázku č. 4

nám zobrazuje, jak by první část implementace měla fungovat. Tato část byla vytvořena v IntelliJ Idea a byla napsána v programovacím jazyku Java. Pro získání všech knihoven byly využity maven dependence, které nám zajistí, že kód bude do budoucna vždy aktuální a že může být i dále vylepšován. Vstupní dokumenty se vkládají úplně stejně jako do testovací implementace. Tedy pomocí struktury tří složek, kdy jedna je určena pro vstupní dokumenty, druhá pro takzvaný trigger, což je soubor, který spustí celý proces a třetí složka určená pro výsledný výstup.

Samotná implementace musela být z velké části upravena a dále vylepšena o avizované zpracování textu. Celá implementace začíná ve třídě StartDocumentAnalysis, přes kterou celý proces pouštíme. Po spuštění je zavolána třída HotFolderCaller, přesněji metoda watchfolder. Tato metoda má za úkol spustit sledování námi chtěné složky, což je naše trigger složka. Po kontrole, zda složka vůbec existuje ve stanoveném souborovém systému, přecházíme k samotnému sledování. K tomu je využívaná takzvaná WatchService, která je spouštěna nad složkou. Reaguje celkem na tři druhy změň ve složce. Na úpravu, na vytvoření nebo na vymazání souboru. Zde se využívá druhá zmíněná funkce, tedy když se ve složce objeví nový soubor. Dokud se tak nestane, je složka kontrolovaná, zda k této události nedošlo. Poté co se tak stane, proces začíná se zpracováním. Získá se název trigger souboru z trigger složky a soubor je následně smazán. Hned vzápětí je zavolána třída DocumentProcess, přesněji metoda processDocumentGoogle. Tato metoda začíná tím, že listuje všechny dokumenty uvnitř vstupní složky. Prochází dokument po dokumentu a zpracovává postupně všechny. Pokud by tam byl soubor PDF, je spuštěna metoda na rozpoznávání textu v PDF dokumentech. V našem případě se metoda jmenuje processDocumentPDFGoogle. V obouch metodách probíhá proces velmi podobně, po inicializaci se vezme dokument a pomocí API se vygeneruje výsledek ve formě JSONu. U druhé metody je to složitější o již zmíněný Google Storage, kde je dokument nejdříve uložen a až poté odtud zpracován. Také je zde uložen výsledný JSON. Výsledky z JSONů jsou nakonec uloženy do pole a připraveny pro pozdější využití. Ve druhém procesu navíc dochází k takzvanému uklizení v Google Storage, kdy je úložiště vyčištěno od vstupního a výstupního dokumentu.

Výše popsanou část procesu lze vidět na přiloženém obrázku č. 5 na další stránce.



Obrázek 5 Diagram toku dat zobrazující proces zpracování dokumentu. (Vlastní zpracování)

Poté co se tímto způsobem zpracují všechny dokumenty, je zavolána metoda na vytvoření DOCX dokumentů. Poté přecházíme k samotnému zápisu do dokumentu. Zde začíná další část implementace, která bude spouští naše další funkcionality. Následně začneme se zpracováním našich výsledků. Každý odstavec textu necháme vyhodnotit pomocí Rake knihovny, díky které získáme ojedinělý název bloku. Výsledek je zapsán do výstupního DOCX dokumentu. Text je dále uložen zvlášť pro další využití, dokud současný vstupní dokument není zpracován celý. Všechny odložené texty následně využíváme k výpočtu testu čitelnosti pro celý dokument.

Tímto testem získáváme dvě hodnoty, které ohodnocují text. Pomocí jednoduchého propočtu, je vypočítaná výsledná čitelnost a náš výstupní DOCX dokument je uložen do složky dle této hodnoty. Jak jsou tyto hodnoty rozděleny a jak jsou nazvány výsledné složky můžeme vidět na obrázku č. 6.

Grade	**Reading Ease**	Folder name	**Grade level**
5th grade	100-90	Elementary	< 6
6 grade	90-80	Middle	6 8
7 grade	80-70		
8 a 9 grade	70-60		
10th to 12th grade	60-50	High	9 12
College	50-30		
College graduate	30 10	University	> 12
Profesional	10 0		

Obrázek 6 Hodnoty testu čitelnosti s názvy složek pro výstupní docx dokumenty (Vlastní zpracování)

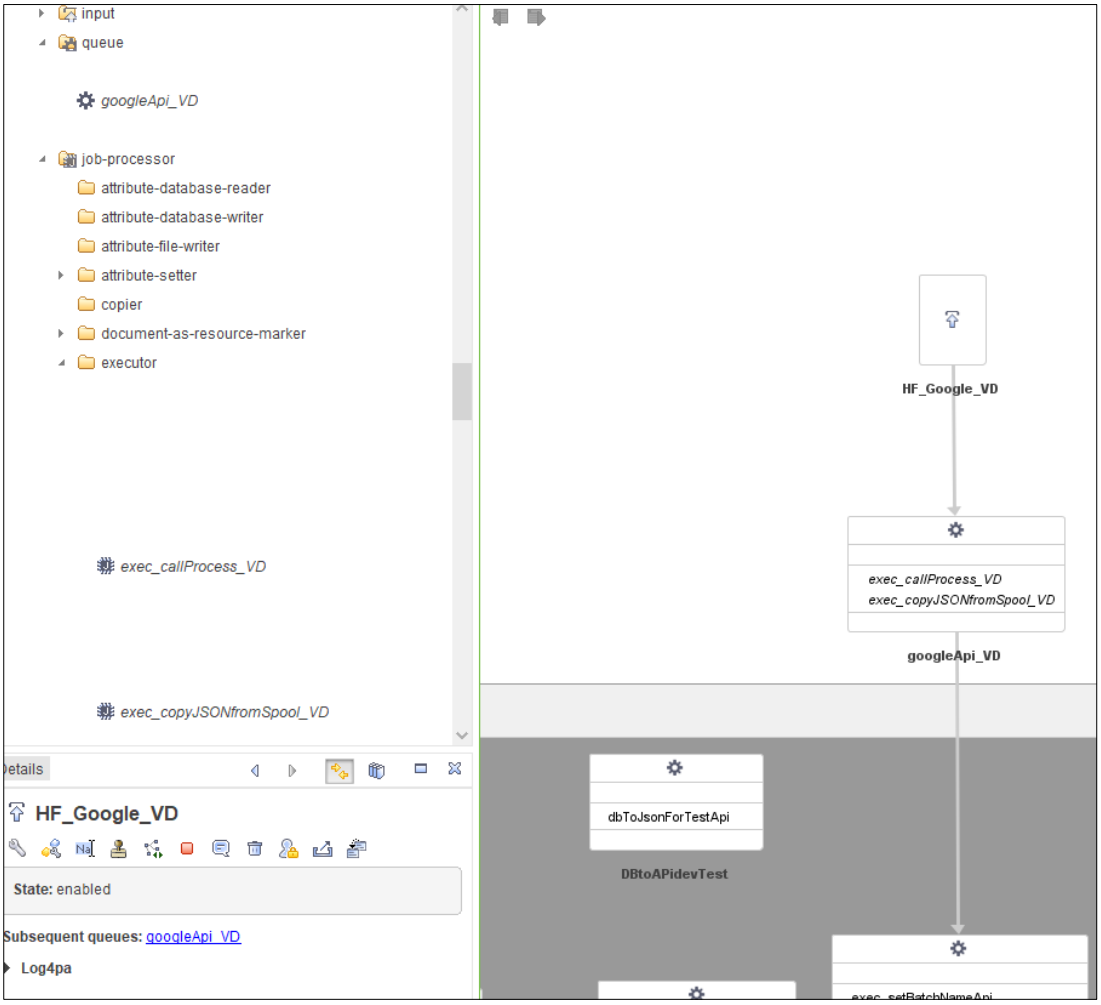
Skript dále pokračuje smazáním všech dokumentů ze vstupní složky a pokračuje k úvodnímu hlídání trigger složky, zda se v ní neobjeví nový dokument, který by spustil nové kolo zpracovávání.

12.2 Druhá část implementace

Jak již bylo napsáno, druhá část implementace se bude odehrávat především v nástroji Inspire Xpress, tedy přesněji v již zmiňovaném Inspire Automation. Ač proces bude z velké části vycházet z testovací a první implementace, samotné fungování bude maličko odlišné. Je nutné naše řešení uzpůsobit a správně zasadit do nástroje.

Úvodní část je téměř identická, sleduje se trigger složka a pokud se do ní cokoliv vloží, spustí se proces. Proces již nebude navázán na prostředí IntelliJ Idea ale na Inspire Automation. Zde bylo nutné naimportovat všechny potřebné knihovny, které naše implementace bude využívat. Poté co jsme naimportovali všechny potřebné knihovny, využili jsme vstupního modulu, který se správným nastavením dokázal sledovat naší trigger složku. Na to navazuje první řada pod názvem googleApi_VD, kde jsme vytvořili proces exec_callProcess_VD využívající náš skript na zpracování dokumentů a využití optického rozpoznání textu. Využívá již výše popsanou metodu processDocumentGoogle, ve které se pomocí Google Vision API

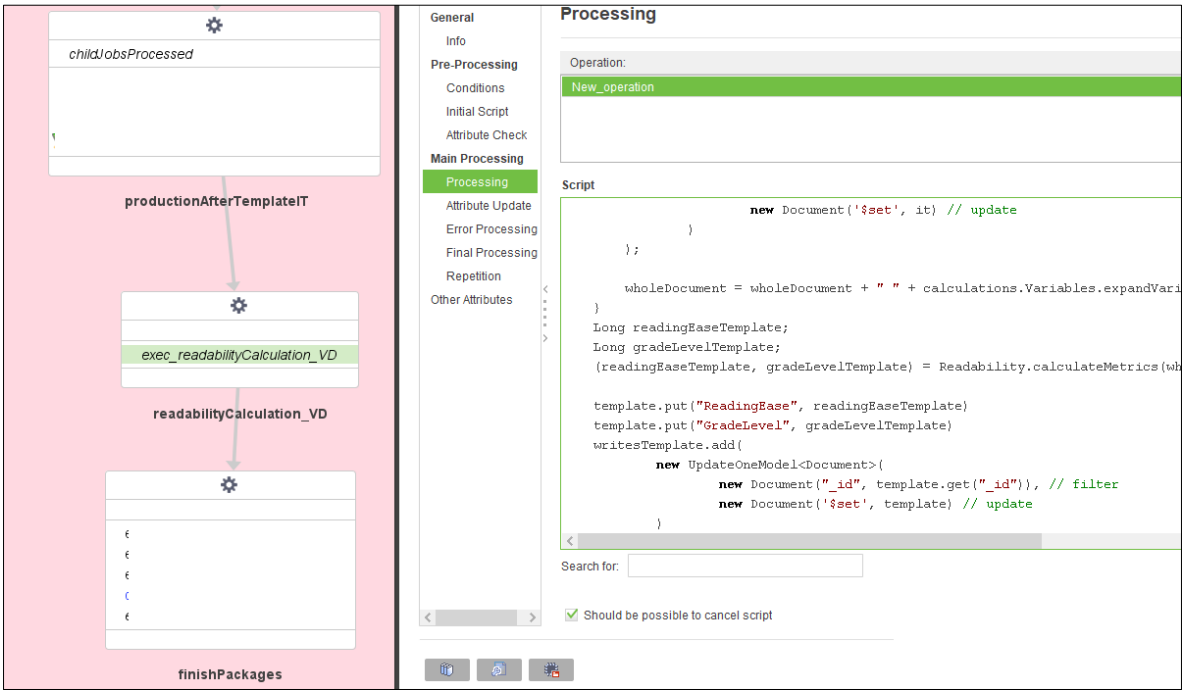
zpracovávají všechny vložené dokumenty ve vstupní složce. Tento proces je v takzvaném executoru. Uvnitř executoru lze vytvořit operaci typu skript. Do této operace můžeme vložit náš kód v jazyce Groovy. Vzhledem k tomu, že je to velmi podobný programovacímu jazyku Java, nebyl problém využít náš připravený kód pro zpracování dokumentů. Poté co jsou všechny dokumenty zpracovány, přecházíme ke zpracování výstupů. Zde se proces mění, nyní se nevytvářejí DOCX dokumenty a ani se nevyužívají žádné přidané funkcionality. Je nutné výsledek přetvořit do vstupního JSON dokumentu, který se shoduje s již vytvořenými procesy. Tento soubor obsahuje veškeré potřebné informace o každém bloku, jako je jeho pořadí na stránce, stránka v dokumentu, název vstupního dokumentu a samozřejmě výsledný text. Byl tedy vytvořen další executor pod názvem exec_copyJsonFromSpool_VD. Zde se nastavily nezbytné atributy pro další fungování procesu a výsledný JSON je zkopírován do produkční složky. Aby se naše část řešení správně napojilo, bylo nutné udělat menší dílčí úpravy v nástroji. Poté proces pokračuje do další řady, která je již součást řešení Inspire Xpress. Údaje ze vstupního JSON dokumentu jsou zpracovány a výsledkem je kategorie, prezentující naše soubory a bloky. Také vzniká kompletní záznam o této kategorii v NOSQL databázi Mongo. Vstupní modul a následnou řadu s exekutorem lze vidět na obrázku č. 7, který se nachází na další stránce.



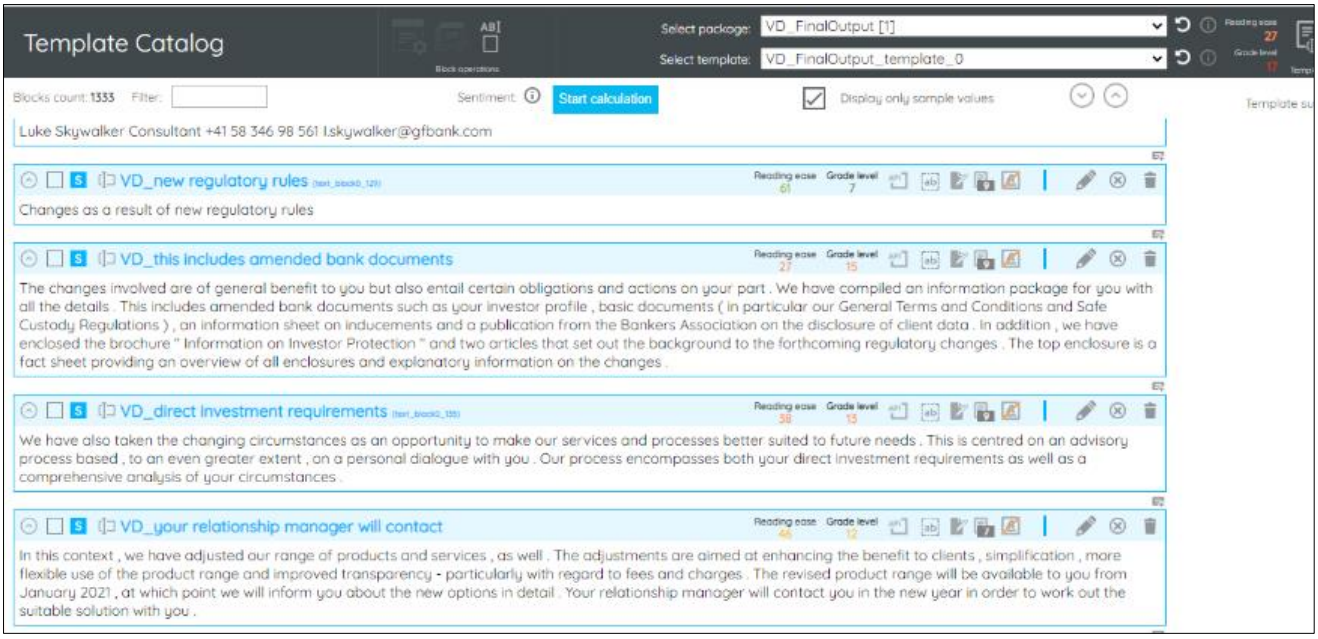
Obrázek 7 Začátek procesu v Inspire Automation včetně modulů na levé straně. (Vlastní zpracování)

Když je tento proces dokončen, je nutné v Inspire Xpress spustit naší vytvořenou kategorii, nastavit ideální hodnoty pro konsolidaci bloků a dokumentů, a hlavně vybrat naší funkci na automatické pojmenovávání bloků. Zde bylo zapotřebí drobné úpravy na front endu a přidat tuto volbu. Uživatel se může rozhodnout mezi generickým názvem a názvem vytvořený pomocí knihovny RAKE. Po zvolení vstupních parametrů a spuštění procesu začíná vytváření balíčku s našimi dokumenty. Je vytvořena základní struktura bloků a dokumentů v databázi Mongo a spouští se iterace, které redukují počty bloků, a i samotných dokumentů. Také probíhá detekce a produkce proměnných uvnitř bloků. V databázi vznikají výsledné bloky a dokumenty. Zde se nám pojmenovávají bloky, dle volby, kterou jsme zvolili na začátku generování.

Proces dále pokračuje naší druhou funkcí, a to výpočtem srozumitelnosti textu. Jak je možné vidět na obrázku č. 8, v nové řadě readabilityCalculation_VD je umístěn executor exec_readabilityCalculation_VD. Zde je vytvořen kód, který si volá všechny vytvořené bloky a obohacuje je o hodnoty Reading Ease a Grade Level. Tento výpočet se dělá nejen pro celé dokumenty, ale také pro samotné bloky. Díky tomu budou tyto hodnoty poskytovány zákazníkům přímo v editoru a uvidí, která část textu je hůře čitelná. Poté se provedou poslední úpravy celého balíčku a proces je vytvořen. V tu chvíli lze vidět zpracované dokumenty v editoru uvnitř nástroje Inspire Xpress.



Obrázek 8 Část kódu a zobrazení zasazení pro počítání testu čitelnosti pro Templatu v Inspire Automation (Vlastní zpracování)



Obrázek 9 Pohled do Template Catalogu uvnitř Inspire Xpress, kde jsou zobrazeny bloky dokumentu i s hodnocením textu reading ease a grade level. (Vlastní zpracování)

Jak můžete vidět na přiloženém obrázku č. 9, takto vypadá výsledek našeho zpracování. Jsou vidět údaje o čitelnosti bloků a také pro zvolenou šablonu. Každý blok byl také automaticky pojmenován. Zde již zákazník může provádět svoje úpravy dle libosti. Údaje o srozumitelnosti textu jsou dále propsány do reportů, které lze vygenerovat v nástroji. Je zde také možnost převodu dokumentů do finálního systému Inspire Interactive, kde proběhne finální úprava dokumentu na šablonu tak, jak si přeje zákazník. Zde by již měly být napojeny další firemní procesy zákazníka.

Při zběžné analýze našeho řešení, jsme zjistili, že řešení tohoto typu může ušetřit zákazníkovy spoustu času. To můžeme demonstrovat na reálném příkladu, kdy nejmenovaný zákazník provedl analýzu svých dokumentů. Při 38 dokumentech, které obsahují celkem 114 stránek a 1699 bloků, zákazník odhadl, že migrace bude trvat celých 380 dní. Také je nutné zdůraznit, že jen tato analýza trvala tři týdny.

S využitím Inspire Xpress, byla provedena stejná analýza z identického vzorku dokumentů během několika hodin. Výsledkem analýzy bylo že z 38 dokumentů vzniklo celkem 28 konsolidovaných šablon. Z 1699 bloků vzniklo konsolidací 300 jedinečných textových bloků a výsledný čas pro migraci pomocí Inspire Xpress byl odhadnut na 30-40 dní. Pokud porovnáme výsledné časy, zjišťujeme že bychom

ušetřili zhruba 339 dní, a ještě k tomu zabránili redundancím ve výsledných šablonách.

Při další analýze zaměřené na automatické pojmenovávání bloků jsme zjistili, že ruční pojmenovávání 4000 bloků textu zabere pěti pracovník čtyři pracovní dny. S využitím našeho automatického pojmenovávání využívající RAKE, bylo dosaženo srovnatelného výsledku za necelých 15 minut. To v přepočtu představuje úsporu téměř 20 dnů a uvolnění všech pracovníků na jiné úkoly.

13 Shrnutí výsledků

V praktické části jsme se nejdříve zaměřili na analýzu a porovnání komerčních technologií na zpracování textu, které by nám měli pomoci v našem řešení. Každý nástroj bylo nutné detailně prozkoumat a poté si ke všem nástrojům zřídit testovací účet. Za pomocí dokumentací a využití našeho konceptu, bylo vytvořeno testovací prostředí, kde byly otestovány naše dokumenty. Z analýzy a s přihlédnutím na stávající systém, do kterého se integrovalo vyplynulo, že nejvhodnějším kandidátem pro implementaci je Google Vision API.

První implementace demonstrovala celkové možnosti práce s textem. Nejdříve bylo využito našeho nástroje ke zprocesování grafických dokumentů do textové podoby. Text byl dále zpracován testem čitelnosti Flesch-Kincaid, který díky našemu řešení dokázal dokumenty rozřadit dle předem stanovených tříd. V další části proběhlo zpracování samotných bloků textu, kde bylo ke každému bloku automaticky vygenerován název dle obsahu. Všechny výsledky byly uloženy do souborů typu DOCX a samotné soubory byly uloženy do složek dle výsledků z testu čitelnosti.

Po dokončení následovala druhá implementace, tentokrát do firemního nástroje Inspire Xpress. Bylo využito modulů a struktury uvnitř aplikace Inspire Automation. Zde byly zapracovány všechny námi probrané funkcionality, včetně testu čitelnosti. V nástroji Inspire Xpress bylo třeba upravit rozhraní a vytvořit skripty, díky čemuž jdou námi zpracované dokumenty prohlížet a přímo upravovat uvnitř nástroje Inspire Xpress a můžeme vidět u automatické pojmenování v názvech bloků a také jejich ohodnocení čitelnosti. Toto ohodnocení bylo použito i na samotné dokumenty. Díky vhodné implementaci bylo možné dále využít finální aplikaci Inspire Interactive, která je určená především zákazníkům k finálnímu vytváření šablon pro firemní procesy.

14 Závěry a doporučení

Z pohledu zadání diplomové práce byly splněny všechny předem stanovené body. V teoretické části byly vysvětleny základní koncepty strojového učení a optického rozpoznávání znaků. Dále zde byly uvedeny základní informace o API a produktech od firmy Quadient, které jsou nezbytnou součástí implementace. Nakonec zde byla probrána problematika migrace dokumentů, která s naším tématem přímo souvisí.
V praktické části byla předpokládaná implementace do existujícího nástroje na migraci dokumentů, který je hojně využíván zákazníky po celém světě. Samotná implementace měla prozkoumat možnosti zpracování grafických dokumentů a jejich možnosti zapracování do již existujícího migračního nástroje Inspire Xpress. Tímto byl vytvořen zcela nový přístup ke zpracování dokumentů, který je zcela ojedinělý a nebyl ještě nikde použit. Také byly v praktické části využity nové funkcionality, které usnadňují další zpracování obsahu přímo uvnitř jmenovaného nástroje. Před samotnou implementací do tohoto nástroje bylo vytvořeno řešení, které demonstrovalo všechny vytvořené funkcionality spojené do jednoho celku. Výsledkem byly DOCX dokumenty, jež jsou kategorizovány do několika složek podle čitelnosti dokumentu.
Řešení nám ukázalo že ani dnes nejsou všechny nástroje zcela dokonalé a že je stále co vylepšovat. Praktická část také ukazuje, že lze stále nacházet zcela nové koncepty zpracování textu spojením vícero nástrojů, které vytvářejí zcela nový systém. Námi demonstrovaná implementace také ukázala možnosti úspor mnohým firmám, jež měly své dokumenty pouze v grafické formě a do teď neměly možnost tak rychle a jednoduše migrovat svoje dokumenty na nový systém.
Samotné poznání všech nástrojů bylo časově náročné a vytvoření optimálního řešení nebylo úplně snadné. Dále detailní prozkoumání tak komplexního nástroje, jakým je Inspire Xpress, zabralo nemalé úsilí a nebylo to vždy zcela jednoduché.
Uvedená práce prokázala, že navržený systém je funkční a je připravena po technické stránce pro komerční využití. Po finálních úpravách a testech může být využita v reálném nasazení. Také jsou zde velké příležitosti na další rozšíření práce s textem, jako je například automatické stylování textu nebo počítání sentimentu textu.

15 Seznam použité literatury

[1] BURNS, Ed. What Is Machine Learning and Why Is It Important? *SearchEnterpriseAI* [online]. [vid. 2022-02-08]. Dostupné z: https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML

[2] TATNALL, Arthur, ed. Machine Learning. In: Arthur TATNALL, ed. *Encyclopedia of Education and Information Technologies* [online]. Cham: Springer International Publishing, 2020 [vid. 2022-04-16], s. 1117–1117. ISBN 978-3-030-10575-4. Dostupné z: doi:10.1007/978-3-030-10576-1_300426

[3] BHATTACHARJEE, Joydeep. Supervised Learning. In: Joydeep BHATTACHARJEE*Practical Machine Learning with Rust* [online]. Berkeley, CA: Apress, 2020 [vid. 2022-04-01], s. 31–105. ISBN 978-1-4842-5120-1. Dostupné z: doi:10.1007/978-1-4842-5121-8_2

[4] JOSHI, Ameet V. Unsupervised Learning. In: Ameet V JOSHI*Machine Learning and Artificial Intelligence* [online]. Cham: Springer International Publishing, 2020 [vid. 2022-04-01], s. 133–140. ISBN 978-3-030-26621-9. Dostupné z: doi:10.1007/978-3-030-26622-6_14

[5] YAN, Wei Qi. Reinforcement Learning. In: Wei Qi YAN*Computational Methods for Deep Learning* [online]. Cham: Springer International Publishing, 2021 [vid. 2022-04-01], Texts in Computer Science, s. 77–87. ISBN 978-3-030-61080-7. Dostupné z: doi:10.1007/978-3-030-61081-4_5

[6] *What is Deep Learning?* [online]. 16. srpen 2021 [vid. 2022-04-17]. Dostupné z: https://www.ibm.com/cloud/learn/deep-learning

[7] *OCR (Optical Character Recognition) Definition* [online]. [vid. 2022-03-18]. Dostupné z: https://techterms.com/definition/ocr

[8] How does OCR document scanning work? *Explain that Stuff* [online]. 3. prosinec 2010 [vid. 2022-03-18]. Dostupné z: http://www.explainthatstuff.com/how-ocr-works.html

[9] REDDY, Martin. *API design for C++*. Amsterdam Heidelberg: Morgan Kaufmann, 2011. ISBN 978-0-12-385004-1.

[10] *What is an API?* [online]. [vid. 2022-04-16]. Dostupné z: https://www.redhat.com/en/topics/api/what-are-application-programming-interfaces

[11] *About W3C* [online]. [vid. 2022-04-16]. Dostupné z: https://www.w3.org/Consortium/

[12] *REST vs. SOAP* [online]. [vid. 2022-04-16]. Dostupné z: https://www.redhat.com/en/topics/integration/whats-the-difference-between-soap-rest

[13] What is an API? (Application Programming Interface). *MuleSoft* [online]. [vid. 2022-03-17]. Dostupné z: https://www.mulesoft.com/resources/api/what-is-an-api

[14] *Quadient® Inspire Designer & Content Manager | Quadient* [online]. [vid. 2022-07-10]. Dostupné z: https://www.quadient.com/intelligent-communication/customer-communications/inspire-designer-icm

[15] *Quadient® Inspire Interactive | Quadient* [online]. [vid. 2022-07-10]. Dostupné z: https://www.quadient.com/intelligent-communication/customer-communications/quadient-inspire-interactive

[16] *Quadient® InspireXpress | Quadient* [online]. [vid. 2022-07-10]. Dostupné z: https://www.quadient.com/intelligent-communication/how-we-help/consolidate-ccm-platforms/inspirexpress

[17] Intelligently Extract Text & Data with OCR - Amazon Textract - Amazon Web Services. *Amazon Web Services, Inc.* [online]. [vid. 2021-12-27]. Dostupné z: https://aws.amazon.com/textract/

[18] Using the Vision API with Python. *Google Codelabs* [online]. [vid. 2022-05-06]. Dostupné z: https://codelabs.developers.google.com/codelabs/cloud-vision-api-python

[19] *Rapid Automatic Keyword Extraction (RAKE)* [online]. Java. B.m.: Linguistic, 2022 [vid. 2022-07-29]. Dostupné z: https://github.com/Linguistic/rake

[20] Flesch Reading Ease and the Flesch Kincaid Grade Level. *Readable* [online]. [vid. 2022-07-29]. Dostupné z: https://readable.com/readability/flesch-reading-ease-flesch-kincaid-grade-level/

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Akademický rok: 2020/2021

Studijní program: Aplikovaná informatika
Forma studia: Kombinovaná
Obor/kombinace: Aplikovaná informatika (ai2-k)

Podklad pro zadání DIPLOMOVÉ práce studenta

Jméno a příjmení:	**Bc. Václav Tomíček**
Osobní číslo:	**I2000324**
Adresa:	Paseky nad Jizerou 98, Paseky nad Jizerou, 51247 Paseky nad Jizerou, Česká republika
Téma práce:	Koncept využití Umělé inteligence k rozpoznání patternů a logických celků textu
Téma práce anglicky:	The concept of using Artificial Intelligence to recognize patterns and logical units of text.
Vedoucí práce:	Ing. Pavel Blažek, Ph.D. Katedra informačních technologií

Zásady pro vypracování:

Cílem práce je vytvořit nový přístup jak lépe rozpoznávat a zpracovávat dokumenty a tím zjednodušit a zrychlit migraci dokumentů. Předpokládá se implementace řešení do již existující služby Quadient® InspireXpress.
Osnova:
1) Úvod
2) Teoretický popis technologií
3) Návrh metody řešení a implementace
4) Realizace navrhnutého řešení
5) Shrnutí výsledku realizace
6) Závěr

Seznam doporučené literatury:

1. Amazon Textract | Extract Text & Data | AWS. Amazon Web Services (AWS) – Cloud Computing Services [online]. Copyright © 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved. [cit. 11.03.2021]. Dostupné z: https://aws.amazon.com/textract/

2) Quadient® InspireXpress | Quadient (formerly Neopost). Because connections matter | Quadient (formerly Neopost) [online]. Copyright © Quadient. All rights reserved [cit. 11.03.2021]. Dostupné z: https://www.quadient.com/experience/migration-solutions/inspirexpress

3) BENGFORT, B. Applied text analysis with Python. 1st ed. Sebastopol: O'Reilly, 2018. ISBN 978-1-491-96304-3.

4) GUTIERREZ, D. D. Machine learning and data science. 1st ed. Sebastopol: O'Reilly, 2015. ISBN 978-1-63462-096-3.

5) RUPARELIA, N. Cloud computing. 13th ed. Sebastopol: O'Reilly, 2016. ISBN 978-0-262-52909-9.

6) HOLUBOVÁ, I. Big Data a NoSQL databáze. 1st ed. Sebastopol: O'Reilly, 2015. ISBN 978-80-247-5466-6.

Podpis studenta: Datum: 6. 6. 2022

Podpis vedoucího práce: Datum: 6. 6. 2022