Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Diplomová práce

SharePoint Add-In pro vytěžování dat z dokumentů

Plzeň 2017 Veronika Kutková

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracovala samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 16. května 2017

Veronika Kutková

Abstract

Data Mining SharePoint Add-in

This thesis is focused on design and implementation of invoice data mining add-in for SharePoint. A couple of OCR tools were tested and compared, as well as several invoice data mining methods. Experiments proved that OnliceOCR.net is the most precise free OCR tool. As for invoice data mining, Locality-Sensitive Hashing based method turned out to be the most suitable. Observed results were used to design and implement the add-in. The complete solution is hosted on Microsoft Azure cloud platform. Few Azure components (storage queue, webjob) were used to process the data mining requests asynchonously. The outcome of this thesis is the add-in for SharePoint Online, partially automating the invoice transcription process and reducing data entry errors.

Abstrakt

Tato práce je zaměřena na návrh a vytvoření SharePoint add-inu zajišťujícího vytěžování dat z naskenovaných faktur. V rámci experimentu byl vyzkoušen a porovnán vzorek OCR nástrojů. Nejlepších výsledků z bezplatných řešení dosáhl nástroj OnlineOCR.net. Vyzkoušeno bylo také několik metod vytěžování dat z faktur. Jako nejvhodnější se ukázala metoda založená na Locality-Sensitive Hashing. Tyto poznatky byly využity při návrhu a implementaci výsledného add-inu. Kompletní řešení je hostováno na cloudové platformě Microsoft Azure. Pro asynchronní zpracovávání požadavků na vytěžování dat bylo použito několik Azure komponent (fronta, webová úloha). Výsledkem je funkční add-in do SharePoint Online, jehož použití částečně automatizuje ruční přepis faktur do systému a minimalizuje vzniklé chyby.

Poděkování

Ráda bych poděkovala vedoucímu diplomové práce panu Ing. Davidu Wegschmiedovi za jeho ochotu, cenné rady a připomínky při vedení práce. Díky patří také panu Ing. Lukáši Burešovi za pomoc s provedenými experimenty a paní doc. Dr. Ing. Janě Klečkové za vstřícnost a čas, který mi během konzultací věnovala.

Dále děkuji svým nejbližším za podporu během celého studia.

Obsah

1 Úvod

Na světě je každoročně vyprodukováno obrovské množství papíru (více než 300 milionů tun), z nichž nezanedbatelný podíl tvoří firemní dokumenty [2]. Přestože současný rozvoj digitálních technologií s sebou přináší možnosti, jak používání dokumentů v papírové formě alespoň omezit či úplně nahradit, velké množství firem na ně stále nedá dopustit. Dle průzkumů se v papírové podobě vyskytuje až 85 % firemních dokumentů [1].

Kromě ekologického dopadu této skutečnosti lze ale pozorovat i dopad finanční, zahrnující kromě vlastních nákladů na tisk dokumentů také náklady spojené s jejich správou a archivací – podle odhadů stráví zaměstnanci hledáním potřebných dokumentů i 30–40 % svého času [1]. Důsledkem včasného nenalezení požadovaného dokumentu pak může být pokuta za pozdní zaplacení faktury a potažmo i ztráta obchodního partnera.

Cílem této diplomové práce je navrhnout a vytvořit SharePoint add-in sloužící k jednoduchému vytěžování dat z účetních dokumentů tak, aby jeho použití přineslo částečnou automatizaci procesu ručního přepisování dokumentů do systému a minimalizaci chyb vzniklých během tohoto procesu. Vzhledem k tomu, že je vyžadováno bezchybné vytěžení dat, je nutné aby uživatel vytěžená data následně zvalidoval.

V teoretické části se čtenář nejprve seznámí s procesem rozpoznávání znaků, vytěžování dat a jejich aplikací na účetní dokumenty (konkrétně faktury). Následující dvě kapitoly pojednávají o cloudové platformě Office 365 a produktu SharePoint.

V rámci několika experimentů je porovnán vzorek existujících prostředků pro rozpoznávání a vytěžování dat a čtenáři je nastíněn také návrh vlastního vytěžovacího nástroje. Zjištěné poznatky jsou aplikovány při návrhu a implementaci výsledného SharePoint add-inu.

2 Vytěžování dat z účetních dokumentů

S účetními dokumenty přicházejí firmy do styku každý den. Podkladem pro nákup zboží či služeb jsou pro ně *objednávky*, k dodávanému zboží mohou dostávat, resp. vydávat *dodací listy*, a vyjádření pohledávek vůči odběratelům, resp. závazků vůči dodavatelům představují *faktury*.

Přestože je v současnosti na trhu nepřeberné množství účetních softwarů (např. Pohoda, ABRA FlexiBee), většina firem stále dává přednost účetním dokumentům v papírové podobě [19].

2.1 Účetní dokumenty

Rozsah a způsob vedení účetnictví upravuje zákon č. 563/1991 Sb., o účetnictví. Tento zákon sice neobsahuje taxativní výčet účetních dokumentů, ale stanovuje náležitosti, které musí doklad mít, abychom jej mohli označit za účetní doklad a použít jej pro zaúčtování účetního případu s ním spojeným [64].

Podle § 11 zákona o účetnictví jako účetní doklad označujeme průkazný účetní záznam s těmito náležitostmi:

a) označení účetního dokladu,

b) obsah účetního případu a jeho účastníci,

c) peněžní částka nebo informace o ceně za měrnou jednotku a vyjádření množství,

d) okamžik vyhotovení účetního dokladu,

e) okamžik uskutečnění účetního případu (není-li shodný s okamžikem vyhotovení),

f) podpisový záznam (vlastnoruční podpis nebo uznávaný elektronický podpis) osoby odpovědné za účetní případ a osoby odpovědné za jeho zaúčtování.

Pro účetní doklady dále platí, že musí být čitelné a nesmí vykazovat žádné známky dodatečné úpravy. Vzhledem k technologickému vývoji se nemusí jednat výhradně o doklady v papírové formě, ale povoleny jsou i jiné průkazné technické záznamy [64].

Mezi běžně používané účetní doklady řadíme např. příjemky, výdejky, příjmové a výdajové pokladní doklady, přijaté a vydané faktury, nebo výpisy z bankovních účtů [64].

Přestože jednotlivé účetní doklady obsahují společnou množinu povinně uváděných údajů a jsou zpravidla formulářového typu, jejich struktura obvykle není pevně daná. V papírnictví lze pořídit různé typy pokladních dokladů, účetní programy produkují faktury různých podob a každá banka poskytuje výpisy z bankovních účtů v jiné formě.

Na rozdíl od např. daňového přiznání tedy nelze předem jednoznačně říci, jaký údaj se nachází v levém horním rohu dokladu. Člověk si s různým rozložením údajů na stránce snadno poradí, ale v případě vytěžování dat počítačem je určení významu jednotlivých údajů o něco komplikovanější.

Jelikož většinu účetních dokumentů, které do firmy přicházejí, představují naskenované dokumenty, případně dokumenty v papírové formě, je před samotným vytěžováním dat zapotřebí tyto dokumenty digitalizovat. Prvotním krokem je tedy rozpoznání znaků.

2.2 Optické rozpoznávání znaků

Jako optické rozpoznávání znaků (Optical Character Recognition, OCR) označujeme technologii převodu naskenovaných dokumentů do digitální textové podoby, bez nutnosti ručního přepisování. Takto digitalizované dokumenty lze následně snadno upravovat, třídit, fulltextově prohledávat a archivovat [21].

OCR software funguje na principu analýzy struktury dokumentu. Dokument je nejprve rozdělen na části (odstavce, obrázky, tabulky), poté na řádky, slova a následně na jednotlivé znaky, které pak porovnává se sadou vzorových obrazů. Výsledkem tohoto porovnání je několik pravděpodobnostních hypotéz, na základě kterých OCR software předkládá výsledný rozpoznaný text [11].

Přestože je v současnosti k dispozici relativně velké množství řešení dosahujících vysokých kvalit převodu (viz kapitola 5.1), nikdy nelze převod považovat za zcela bezchybný a pro zvýšení přesnosti je vyžadována kontrola výsledného textu lidským okem. Kvalitu převodu samozřejmě ovlivňuje také kvalita původního vytištěného a následně naskenovaného dokumentu.

Jednotlivá řešení se kromě kvality převodů a ceny liší např. i počtem jazyků, ve kterých dokáží texty rozpoznávat. Použitím slovníků se kvalita převodu zvyšuje, jelikož jsou jednotlivá rozpoznaná slova s obsahem slovníků porovnávána. Je potřeba si také uvědomit, že strukturované texty (např. formuláře či tabulky) vyžadují při digitalizaci odlišný přístup než souvislé bloky textu (např. knihy nebo články). Při převodu strukturovaných textů totiž nelze postupovat řádek po řádku, ale je nutné určit jednotlivé oblasti textu a zajistit jejich soudržnost.

Podrobnějšímu srovnání různých existujících řešení optického rozpoznávání znaků je věnována kapitola 5.1.2.

2.3 Vytěžování dat

Vytěžování (dolování) dat (angl. Data Mining) je proces extrakce skrytých, potenciálně zajímavých a užitečných informací, vzorců a vztahů z dat velkého objemu. Tato disciplína v sobě kombinuje přístupy ze statistiky a umělé inteligence (jako např. strojové učení) [17].

Vytěžování dat se v současnosti nejčastěji věnují společnosti orientované na zákazníka, a to za účelem zvyšování tržeb. Sledováním transakčních dat prodeje a analýzou chování spotřebitelů pak např. dokáží lépe odhadnout, na jaké zboží či služby se dále zaměřovat a jak je s co nejvyšší úspěšností propagovat. Kromě komerční sféry však mají metody vytěžování dat své místo např. i v oblasti výzkumu (při čištění a přípravě dat), zdravotnictví (předpovědi úspěšnosti různých typů léčeb) nebo kriminalistiky (vytipování nebezpečných míst) [70].

2.3.1 Vytěžování účetních dokumentů

Jak již bylo zmíněno v kapitole 2.1, každý účetní doklad se vyznačuje určitými náležitostmi. Při práci s účetními dokumenty jsou pro nás tyto údaje důležité, zajímá nás např. kdy byl doklad vystaven a na jakou částku. Abychom mohli s digitalizovanými dokumenty dále plnohodnotně pracovat, je zapotřebí tyto informace v textu správně určit.

V dalším textu se (bez újmy na obecnosti) omezíme na jeden konkrétní typ účetních dokumentů, a to na *faktury*. Důvodem je kromě jejich komplexnosti a rozšířenosti také expertíza společnosti CCA Group v jejich zpracování.

3 Cloudová platforma Office 365

3.1 Cloud computing

Cloud computing je jedním ze současných trendů v oblasti IT, spočívající v dodávání výpočetních služeb (softwarových i hardwarových) přes internet. Poskytovatelé cloudu nejčastěji zprostředkovávají servery, sítě, nebo např. databáze či jiná úložiště (viz dále, kapitola 3.1.3). Zákazníci za tyto zdroje a služby platí podle jejich skutečného využití, v závislosti na počtu minut, resp. objemu dat [34], [25].

3.1.1 Výhody

Finanční stránka věci je jedním z důvodů, proč se firmy pro cloud computing rozhodují – eliminace nutnosti pořízení vlastního software a/nebo hardware může pro firmu představovat značnou úsporu nákladů [34]. Toto představuje výhodu zejména pro začínající společnosti, které nejsou zatím schopny přesně definovat rozsah požadovaných prostředků. Poskytovatelé cloudu řeší také instalace a aktualizace software, zálohování, i případné výpadky či bezpečnostní problémy. Pokud by se firma rozhodla provozovat požadovanou infrastrukturu svépomocí, mohlo by to pro ni znamenat další náklady (nejen na software a hardware, ale např. i mzdové náklady na IT pracovníky) [29].

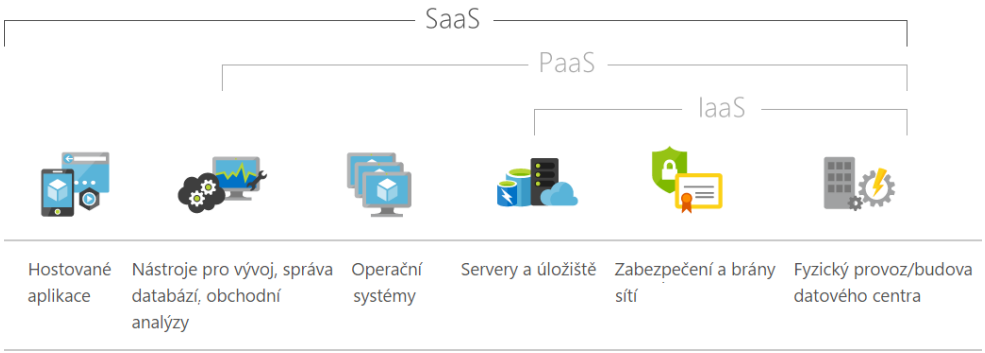
Další klíčovou vlastností cloud computingu je škálovatelnost, tj. schopnost navyšovat (resp. snižovat) objem poskytovaných výpočetních služeb podle aktuální potřeby. Toto představuje výhodu např. pro internetové obchody, které v předvánočním období zaznamenávají zvýšení návštěvnosti a prodeje, a vyžadují proto větší výpočetní výkon, který by však po zbytek roku nevyužily a musely za něj zbytečně platit [34], [29].

3.1.2 Nevýhody

Za určité riziko cloud computingu lze považovat fakt, že data jsou fyzicky uložena u poskytovatele cloudu, někdy dokonce i v jiné zemi. Firmy se proto mohou obávat možnosti zneužití jejich dat, ale je třeba podotknout, že zabezpečení předních poskytovatelů cloudu je často na vyšší úrovni, než by firma dokázala vlastními prostředky zajistit [25].

3.1.3 Typy cloudových služeb

Podle typu poskytovaných služeb cloud computingu rozlišujeme tři hlavní kategorie, viz obrázek 3.1.



Obrázek 3.1: Typy cloudových služeb [38].

Infrastructure as a Service (IaaS)

IaaS představuje základní kategorii cloud computingu. V tomto případě je poskytován jen hardware a infrastruktura (servery, datová úložiště, ...), nejčastěji prostřednictvím virtualizace. Typicky se jedná např. o webhosting nebo analýzy velkých dat [26], [35].

Mezi lídry v poskytování cloudových služeb kategorie IaaS patří podle Gartnerova magického kvadrantu z roku 2016 (viz obrázek 3.2) Amazon Web Services (již pošesté v řadě) a Microsoft se svojí cloudovou platformou Microsoft Azure [13].

Platform as a Service (PaaS)

Druhou úrovní je poskytování platformy jako služby. Tento typ služby přidává k IaaS vývojové nástroje, systémy správy databáze a další prostředky potřebné pro vývoj a údržbu aplikací. Vývojáři tak mají k dispozici prostředí podporující celý životní cyklus aplikace [27], [36].

Mezi poskytovatele PaaS řadíme např. Microsoft Azure či Heroku.

Software as a Service (SaaS)

Za nejkomplexnější z těchto kategorií označujeme poskytování softwarových aplikací jako služby, tj. bez nutnosti koupě licence. Zákazník software využívá na základě zaplaceného předplatného, ale není jeho vlastníkem.

Obrázek 3.2: Magický kvadrant - IaaS celosvětově (Gartner, 2016)

Některé aplikace SaaS jsou spustitelné přímo z webového prohlížeče a jejich uživatelé tak mají k dispozici vždy aktuální verzi aplikace bez nutnosti vlastní instalace nebo aktualizace. V rámci předplatného SaaS je ale možné využívat i tzv. on-premise řešení, tedy software instalovaný a provozovaný přímo u zákazníka [28]. Populárním řešením je také tzv. *hybrid SaaS*, kombinující on-premise s cloudovým řešením [77].

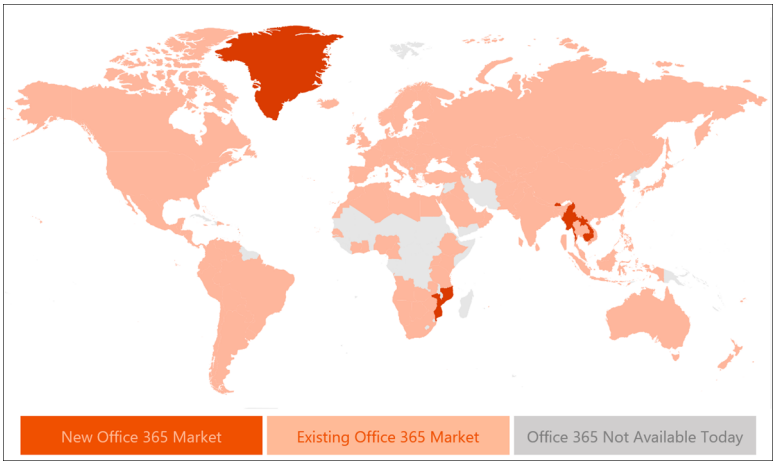
Typickým příkladem SaaS může být e-mailový klient nebo např. *Microsoft Office 365* [38].

3.2 Office 365

Office 365 je jedním ze SaaS řešení, které poskytuje společnost Microsoft (dále např. Power BI určené k analýze dat velkého rozsahu či Dynamics 365

kombinující ERP s CRM systémy). Kromě kancelářských nástrojů ze známého balíku Microsoft Office (např. Word, Excel, PowerPoint, ...) zahrnuje Office 365 také nástroje usnadňující týmovou spolupráci (jako např. Skype, či Outlook). Uživatelé mohou k jednotlivým produktům a službám přistupovat prostřednictvím webového prohlížeče a využívat online verze jednotlivých produktů z jakéhokoliv zařízení s internetovým připojením, případně si stáhnout desktopové verze produktů až na pět svých zařízení (dle předplatného, viz kapitola 3.2.1). Nástroje Office 365 jsou uživatelům dostupné po přihlášení na adrese `http://portal.office.com`.

Produkt Office 365 byl společností Microsoft poprvé ohlášen v říjnu 2010 a od června následujícího roku byl zpřístupněn ve 40 zemích světa ve 20 jazykových mutacích. V současné době je Office 365 dostupný již ve 150 zemích a ve 44 jazycích, viz obrázek 3.3 [65], [75].



Obrázek 3.3: Rozšíření Office 365 ve světě (stav k 23. 11. 2016) [39].

3.2.1 Předplatné Office 365

Office 365 je uživatelům poskytováno na základě měsíčního či ročního předplatného ve dvou základních variantách (plánech), odlišujících se rozsahem aplikací a cenou – ve variantě pro domácí a pro firemní použití. V každé z těchto variant mají uživatelé k dispozici kromě standardního kancelářského software i 1 TB v cloudovém úložišti OneDrive [63], [58].

Kromě těchto dvou základních plánů jsou k dispozici i další varianty, určené specifickým skupinám uživatelů. Studenti a zaměstnanci akreditovaných akademických institucí mohou využívat předplatné Office 365 ve variantě Education, pro organizace státní správy (např. státní úřady, kraje

nebo obce) pak Microsoft připravil několik plánů označovaných jako Government [60], [62].

Office 365 pro domácí použití

Ve variantě pro domácí použití rozlišujeme dvě verze Office 365 – Office 365 Personal a Office 365 Home. V obou těchto verzích má každý uživatel k dispozici 60 minut měsíčně pro volání na mobilní telefony i pevné linky prostřednictvím Skype (v České republice pouze na pevné linky) [63].

Verze Personal je určena pro jednoho jediného uživatele a lze ji nainstalovat na jeden počítač, jeden tablet a jeden mobilní telefon. Při volbě měsíčního předplatného vychází jeden měsíc užívání Office 365 na 189,99 Kč (6.99 $), roční předplatné stojí 1899,00 Kč (69.99 $) [63].

Office 365 ve verzi Home je určeno až pro pět uživatelů, přičemž 1 TB dat na OneDrive i 60 minut na Skype má k dispozici každý z nich. Instalaci je možné provést až na pěti počítačích, pěti tabletech a pěti mobilních telefonech. Za měsíční předplatné uživatelé zaplatí 269,99 Kč (9.99 $), v případě ročního předplatného je cena 2699,00 Kč (99.99 $) [63].

Office 365 pro firemní použití

Office 365 ve variantě pro firemní použití obsahuje navíc i komunikační nástroje, jejichž používání usnadňuje týmovou spolupráci. Microsoft nabízí tři odstupňované verze s omezením na 300 uživatelů – Office 365 Business, Office 365 Business Premium a Office 365 Business Essentials. Podle zvolené verze je třeba počítat s cenou od 6 $ do 15 $ za uživatele měsíčně, v případě ročního závazku se ceny pohybují již od 5 $ do 12.50 $ měsíčně [58], [59].

Ve verzi Business má každý uživatel k dispozici již zmíněný 1 TB pro ukládání a sdílení souborů na OneDrive. Plnou verzi aplikací Office si může každý uživatel nainstalovat až na 5 počítačů a mobilní verzi může využívat až na pěti tabletech a telefonech. Používat lze také aplikace Office v online verzi [58].

Verze Business Premium zahrnuje navíc e-mail s 50GB poštovní schránkou na každého uživatele (služba Exchange online), Skype pro firmy umožňující videokonference v HD a Microsoft Teams (software pro skupinový chat) [58].

E-mail a videokonference nabízí i Office 365 Business Essentials, v této verzi však mohou uživatelé využívat pouze online verze aplikací Office, nikoli jejich plné verze nainstalované na počítači či mobilním zařízení [58].

Firmám, pro něž představuje limit 300 uživatelů překážku, nabízí Microsoft plány Enterprise (v několika různých úrovních), u nichž není maximální

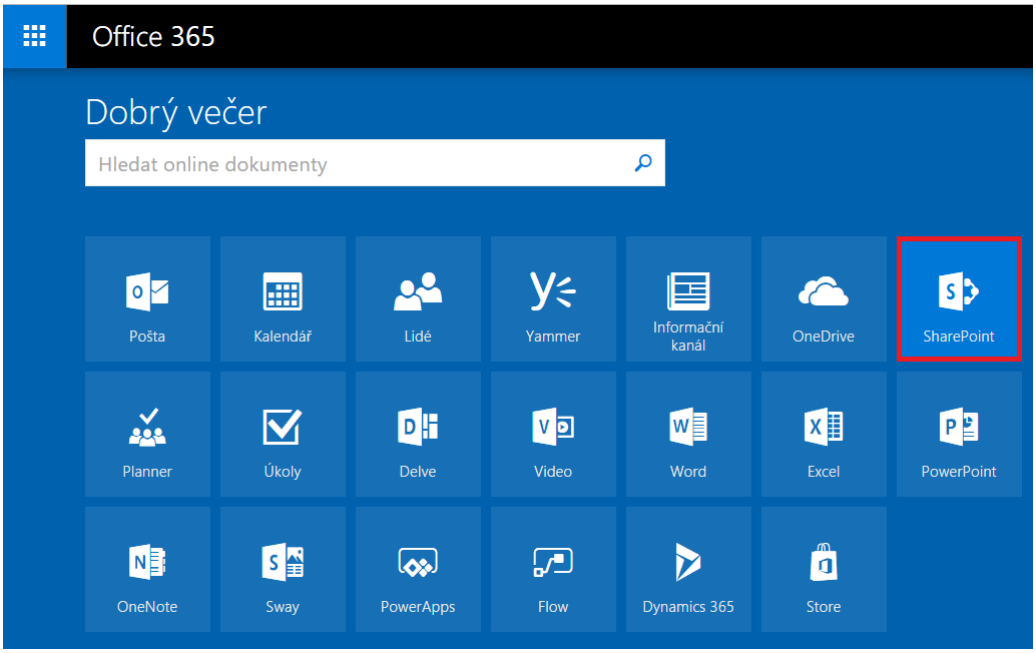
počet uživatelů omezen. Měsíční poplatky (s ročním závazkem) za jednoho uživatele začínají na 8 $ (Enterprise E1 s online verzemi aplikací Office), nejdráž pak vychází verze Enterprise E5 (35 $ měsíčně), která mimo jiné umožňuje provádět pokročilé analýzy, či pořádat konference přes veřejnou telefonní síť [61].

O zajištění přístupu jednotlivých uživatelů v rámci Business a Enterprise plánů předplatného se stará správce Office 365 konkrétní organizace (tzv. *tenant*). Pro jednoduchou správu přihlašovacích údajů a přístupových práv uživatelů je možné využít integrace Active Directory, díky čemuž nejsou uživatelé nuceni si pamatovat další přístupové údaje [58], [61].

4 SharePoint

Za obecným označením *SharePoint* se skrývá několik produktů společnosti Microsoft, mezi něž patří kromě on-premise řešení (SharePoint Server, SharePoint Foundation) i SharePoint Online [55]. Vzhledem k tomu, že je tato práce zaměřena na cloudová řešení, je další text věnován online verzi SharePointu a pojmem *SharePoint* je tedy myšlen *SharePoint Online*.

Webová platforma SharePoint, zahrnutá ve většině plánů předplatného Office 365 (kromě varianty pro domácí použití), představuje nástroj sdílené spolupráce. V rámci Office 365 je SharePoint dostupný z portálu Office 365 (viz obrázek 4.1) a je propojen s ostatními službami jako jsou např. pošta nebo kalendář. Od tohoto propojení je však možné upustit a pořídit si SharePoint samostatně [37].



Obrázek 4.1: Portál Office 365 s aplikacemi na dlaždicích.

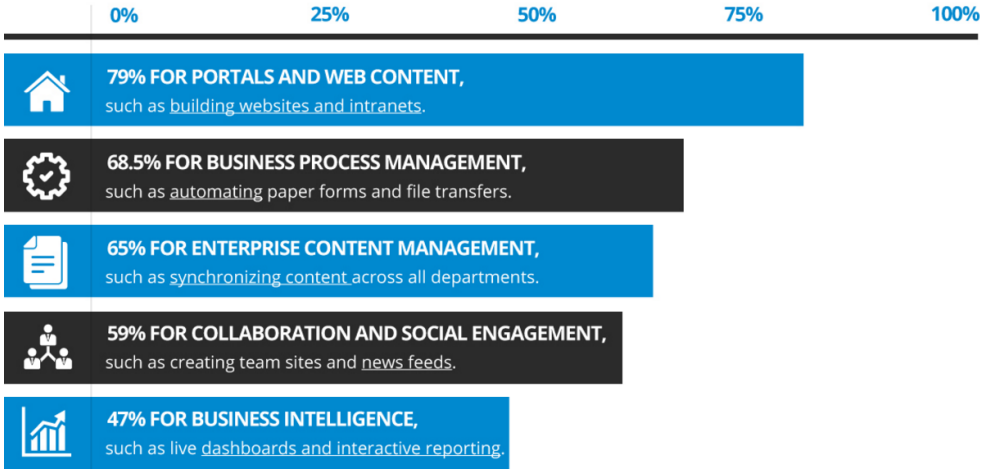
SharePoint bývá často označován jako firemní úložiště dokumentů, ale se svým cílem zajištění efektivní spolupráce toho umožňuje mnohem více. V SharePointu je kromě samotné správy dokumentů možné spravovat i týmové kalendáře a úkoly, navrhovat a řídit pracovní postupy (angl. workflow[1])

[1]Tento pojem je v souvislosti s SharePointem používán i v českém prostředí a představuje nástroje pro automatizaci firemních procesů, jako např. schvalování dokumentů [57].

nebo např. vytvářet firemní intranetové stránky. Využitím možnosti kombinace SharePointu s Office Online lze se spravovanými dokumenty pracovat přímo ve webovém prohlížeči [66].

4.1 Úložiště

Přestože SharePoint není jen firemním úložištěm dokumentů, představuje tato funkčnost nezanedbatelný podíl na způsobech, kterým uživatelé SharePoint využívají, viz obrázek 4.2. Správa firemního obsahu je dle studie [18] důvodem pro využívání SharePointu pro 65 % uživatelů.



Obrázek 4.2: Způsoby využití SharePointu. Zpracováno AllianceTek [12] na základě studie [18].

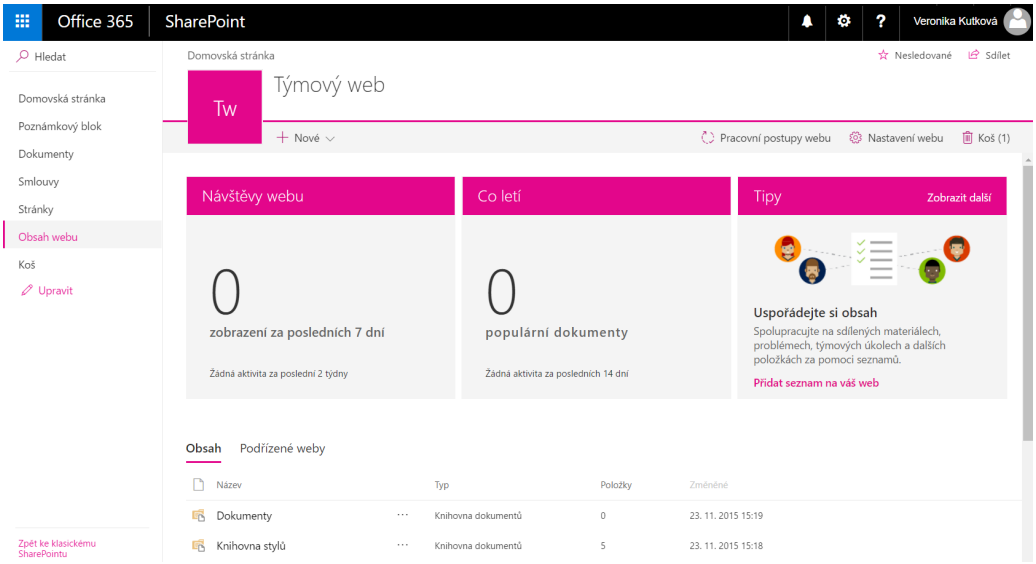
Uživatelům SharePointu je kromě standardního ukládání dat do cloudu umožněna také synchronizace s jejich lokálními úložišti prostřednictvím synchronizačního klienta pro OneDrive. Tímto způsobem mají uživatelé zajištěný přístup k datům i v offline režimu a pro práci se soubory nepotřebují internet. Jakákoliv změna provedená offline se automaticky synchronizuje v okamžiku připojení k internetu [49].

4.2 Kolekce webů

Podle obrázku 4.2 jsou nejpoužívanější funkčností SharePointu webové portály a obecně funkce související se správou webového obsahu. K tomuto účelu slouží v Sharepointu tzv. kolekce webů (angl. *site collection*).

Tímto pojmem označujeme část osobních či firemních úložišť. Kolekce webů je složena z jednotlivých webů (*sites*), které mohou obsahovat další

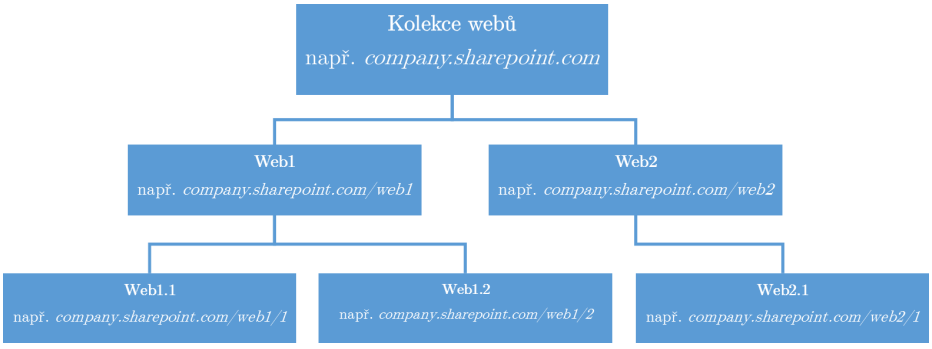
podřízené weby (*subsites*) [48]. Weby představují prostor pro týmovou spolupráci a usnadňují sdílení informací mezi uživateli napříč celou organizací. Typickým případem využití kolekce webů může být firemní intranet či projektová dokumentace [50]. Podoba Týmového webu je zachycena na obrázku 4.3.



Obrázek 4.3: Ukázka Týmového webu.

Nastavení vlastností kolekce webů a jejích přístupových práv je vždy společné pro všechny weby, které jsou v kolekci seskupeny, ale v případě potřeby je možné je definovat i jednotlivě [48].

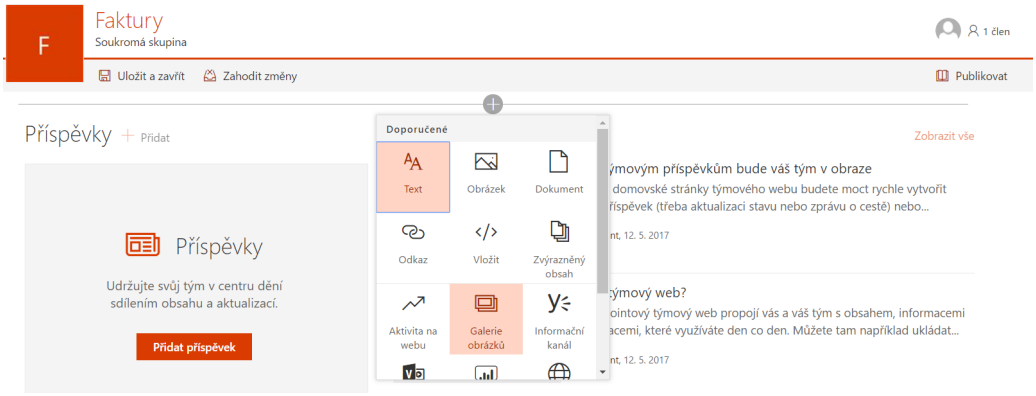
Kořenový adresář kolekce webů je vrcholem celé stromové struktury podřízených webů (viz obrázek 4.4). Kolekce webů je uživatelům defaultně dostupná na adrese `https://company.sharepoint.com` [48].



Obrázek 4.4: Struktura kolekce webů (vlastní zpracování dle [48]).

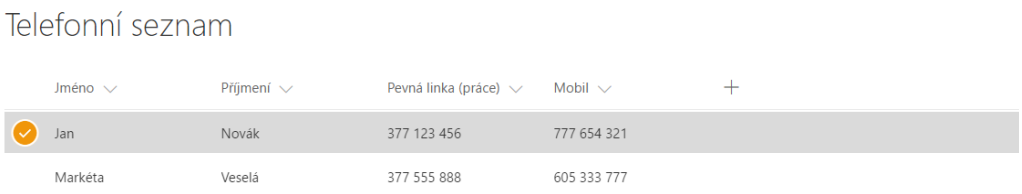
Na úrovni kolekce webů či jednotlivých webů mohou uživatelé vytvářet různé druhy obsahu: webové stránky, knihovny (např. dokumentů, obrázků

nebo multimédií), seznamy a další různě přizpůsobitelné webové prvky, jako např. obrázky, videa či mapy (viz obrázek 4.5) [50], [51].



Obrázek 4.5: Ukázka nastavení obsahu webu.

Seznam (*List*) je prostředkem pro ukládání záznamů do řádků a sloupců, můžeme si ho tedy představit jako obyčejnou tabulku (viz obrázek 4.6). **Knihovna dokumentů** (*Document library*) je speciálním typem seznamu, která navíc umožňuje k jednotlivým záznamům ukládat i soubory [76].



Obrázek 4.6: Ukázka SharePoint seznamu.

Pro často používané typy obsahů je již předpřipraveno několik aplikací, mezi které patří např. kalendář, seznam úkolů, kontakty apod. Pro případy, kdy nejsou tyto aplikace dostačující, zde existuje možnost využití aplikací třetích stran, případně vývoje vlastních aplikací pro SharePoint (viz dále).

4.3 Možnosti přizpůsobení a rozšíření

Vzhledem k tomu, že SharePoint Online je nepřetržitě vyvíjen a aktualizován, je potřeba k případnému přizpůsobování a rozšiřování přistupovat jiným způsobem, než při úpravách klasického on-premise řešení. U online verze SharePointu se totiž nelze spolehnout na to, že námi upravené soubory nebyly při častých aktualizacích automaticky přepsány a provedené změny vráceny [44].

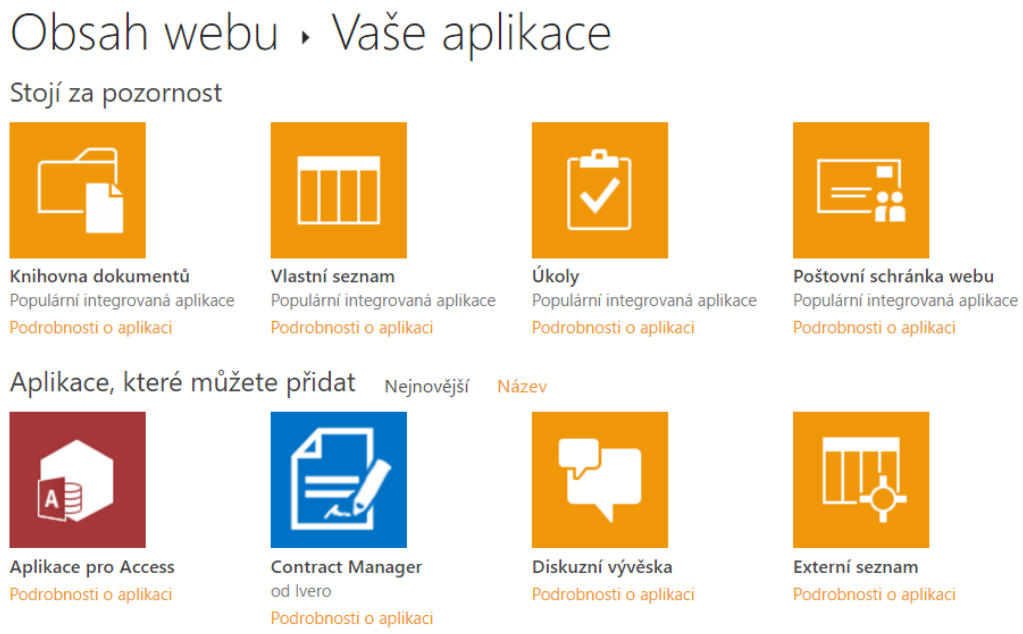
Nejjednodušších úprav lze dosáhnout přímo z prohlížeče – úpravu zvládne běžný uživatel a není tedy vyžadován zásah programátora. Jedná se především o změny obsahu stránky, zobrazovaného loga, motivu webu nebo způsobu zobrazení knihovny či seznamu [44].

V případě potřeby specifických funkčností či nutnosti složitějších úprav je možné využít aplikací třetích stran (pokud jsou k dispozici v SharePoint Store[2]), případně se postarat o vývoj vlastních aplikací, pokud žádné z existujících aplikací nesplňují potřebné požadavky [44].

Vzhledem ke změnám v terminologii se nyní místo označení „aplikace“ používá termín „add-in“, tedy „doplněk“. Jelikož dokumentace Microsoftu nebyla zatím kompletně aktualizována, prozatím se v ní můžeme setkat jak s pojmem „app for SharePoint“, tak „SharePoint Add-in“ [46]. V následujícím textu budeme používat termín „add-in“.

4.3.1 SharePoint Add-in

SharePoint add-in představuje doplněk stávající funkčnosti SharePointu. Všechny add-iny jsou po instalaci v SharePointu dostupné prostřednictvím dlaždic umístěných na stránce Obsah webu (viz obrázek 4.7).



Obrázek 4.7: Dlaždice pro spuštění jednotlivých add-inů (aplikací).

Abychom byli v prostředí add-inu schopni pracovat s daty uloženými v SharePointu, přistupujeme k nim prostřednictvím SharePoint API. O zá-

[2]Obchod s komerčními i bezplatnými aplikacemi, dostupný přímo z SharePointu.

kladní konfiguraci add-inu (např. určení místa běhu a přístupových práv ke stránce, ve které add-in běží, stanovení verze add-inu apod.) se stará konfigurační soubor *manifest.xml* [41].

Možnosti integrace add-inu

Add-in lze do SharePointu zakomponovat třemi různými způsoby (viz obrázek 4.8):

a) jako samostatnou celou stránku,

b) jako součást existující stránky,

c) jako prvek uživatelského rozhraní (např. vlastní položka v menu).

a) b) c)

Obrázek 4.8: Možnosti integrace SharePoint add-inu ([69], upraveno)

Pokud se v rámci rozšiřování funkčnosti SharePointu rozhodneme pro vývoj vlastního add-inu, je zapotřebí si před samotným zahájením vývojových prací zvolit způsob hostování vyvíjeného doplňku. Podle způsobu hostování rozlišujeme následující dva druhy SharePoint add-inů – *SharePoint-hosted add-in* a *provider-hosted add-in* [41].

SharePoint-hosted add-in

Sharepoint-hosted add-in je tvořen výhradně komponentami SharePointu, jako jsou např. knihovny nebo seznamy, jejichž prostřednictvím je zajištěna maximální integrace add-inu do SharePointu po stránce vzhledu i chování [41], [42].

O veškerou business logiku se v případě add-inu hostovaného přímo v SharePointu stará JavaScript na jednotlivých stránkách, tzn. žádná část vlastního kódu není vykonávána na straně serveru. Možností je také využití pracovních postupů (workflow) [42].

Data jsou v případě SharePoint-hosted add-inu ukládána do knihoven či seznamů [42].

Tento způsob hostování je vhodný v případě, kdy jsou požadovány úpravy menšího rozsahu, kterých lze dosáhnout i jen pomocí HTML, CSS a JavaScriptu [41], [42].

Provider-hosted add-in

Provider-hosted add-in tvoří kromě SharePoint komponent alespoň jedna vzdálená komponenta, jejíž uživatelské rozhraní je zcela v rukou vývojáře [42].

Při nasazení a hostování add-inu mimo SharePoint je na rozdíl od předchozího typu add-inu většina business logiky soustředěna na straně serveru. Díky tomu není vývoj technologicky omezen pouze na HTML, CSS a JavaScript a vyvíjený doplněk může být proto mnohem flexibilnější než v předchozím případě [41], [42].

Označením *provider* (poskytovatel) je myšlen jakýkoliv vlastník serveru nebo cloudového účtu (např. Microsoft Azure), na kterém je add-in nasazen, tj. zpravidla správce konkrétního předplatného SharePoint Online, do kterého je add-in instalován (pokud je add-in určen pro jednu jedinou organizaci), případně vývojář zajišťující hostování add-inu pro více organizací zároveň (add-in je dostupný organizacím prostřednictvím SharePoint Store) [41].

Pro ukládání dat lze využít relační databázi či jiné úložiště dostupné na platformě Microsoft Azure, případně jakékoliv jiné cloudové službě. Další možností je také ukládání dat na vlastní server [42].

5 Nástroje pro rozpoznávání a vytěžování dat

V následujícím textu se zaměříme na vytěžování dat z jednoho konkrétního typu účetních dokumentů – z přijatých faktur. Z náležitostí vyjmenovaných v kapitole 2.1 nás na přijatých fakturách zpravidla zajímají tyto údaje: číslo faktury, dodavatel faktury, částka, na níž byla faktura vystavena, datum vystavení a datum splatnosti.

Jak již víme z kapitoly 2.2, před tím, než se pustíme do samotného vytěžování dat, je nejprve zapotřebí převést fakturu do digitální textové podoby, tj. provést rozpoznání znaků.

Pro implementaci byl vybrán programovací jazyk Python pro jeho jednoduchost a čitelnost.

5.1 Nástroje pro rozpoznávání znaků

Tato kapitola je věnována popisu experimentu, jehož cílem bylo porovnat úspěšnost (resp. chybovost) digitalizace vzorku naskenovaných faktur v různých jazycích a s různou strukturou pomocí vybraných existujících OCR programů, a to jak neplacených, tak i komerčních.

5.1.1 Přehled existujících OCR

Následující tabulky (tabulka 5.1 a tabulka 5.2) obsahují přehled a stručnou charakteristiku OCR programů, jejichž úspěšnost byla zkoumána.

Freeware a online OCR

Název	OS	Počet jazyků	Deklarovaná přesnost [%]
FreeOCR	Windows	11 (bez češtiny)	neuvedeno
MeOCR	Windows	23 (vč. češtiny)	neuvedeno
Cuneiform	Windows, Linux	24 (vč. češtiny)	neuvedeno
OnlineOCR.net	online	46 (vč. češtiny)	99
NewOCR	online	75 (vč. češtiny)	neuvedeno
Free OCR	online	29 (vč. češtiny)	neuvedeno
Google Docs	online	226 (vč. češtiny)	neuvedeno

Tabulka 5.1: Přehled freeware a online OCR řešení.

Komerční OCR

Název	Cena	OS	Počet jazyků	Deklarovaná přesnost [%]
OmniPage Ultimate	499,99 $	Windows	123 (vč. češtiny)	98
Adobe Acrobat Pro DC	449,00 $	Windows, Mac OS	42 (vč. češtiny)	95
ABBYY FineReader Corporate	395,00 $	Windows, Mac OS	190 (vč. češtiny)	90
Readiris 15 Corporate	199,00 $	Windows, Mac OS	130 (vč. češtiny)	80
PowerPDF Advanced	139,00 $	Windows	123 (vč. češtiny)	85
ABBYY PDF Transformer+	75,21 $	Windows	189 (vč. češtiny)	90
Soda PDF	139,99 $	Windows, Mac OS	8 (bez češtiny)	90
Presto! Page Manager	99,95 $	Windows	54 (vč. češtiny)	75

Tabulka 5.2: Přehled komerčních OCR řešení.

5.1.2 Srovnání přesnosti existujících OCR

Za účelem porovnání výše zmíněných OCR řešení jsme jako testovací data zvolili čtyři faktury naskenované ve formátu *.tif* nebo *.pdf* a následně jsme je konvertovali do formátu *.png* (viz příloha A).

Faktura má často podobu tabulky – její obsah je strukturován do sloupců, tj. není tvořen jedním souvislým blokem textu. Kromě textu obsahují faktury typicky také obrazové informace různého druhu (např. logo společnosti). Z tohoto důvodu obsahuje i každá ze čtyř zvolených faktur kromě textu také čárový kód a logo, popřípadě razítko společnosti. Všechny testované faktury se kromě jiného vyznačují různou čitelností a použitým jazykem:

faktura1: dobře čitelná, v češtině (obrázek A.1)

faktura2: dobře čitelná, v němčině (+ kombinace dalších jazyků – angličtina/francouzština/španělština) (obrázek A.2)

faktura3: hůře čitelná, v češtině (obrázek A.3)

faktura4: dobře čitelná, v angličtině (obrázek A.4)

Abychom byli schopni vyhodnotit správnost převodu faktury do textové podoby, bylo nutné zvolené faktury ručně přepsat do textových souborů ve formátu *.txt*.

Poznámka: Experiment byl prováděn na reálných fakturách získaných od zákazníka společnosti CCA Group. Tato data však není možné poskytnout v kompletním znění z důvodu nepovolení jejich zveřejnění zákazníkem. V příloze A jsou proto umístěny anonymizované faktury a součástí přiloženého CD je jiná sada faktur, na kterých lze experiment vyzkoušet.

Průběh experimentu

Prvním krokem experimentu bylo postupně nahrát všechny čtyři naskenované faktury do každého ze zvolených testovaných OCR programů. Výsledkem procesu rozpoznávání znaků je vždy textový soubor, zpravidla ve formátu *.txt* nebo *.doc*. Jelikož nás pro účely tohoto experimentu zajímá pouze výsledný text a není tedy třeba zachovávat formátování výchozího textu, byl jako jednotný formát výstupních souborů zvolen formát čistého textu *.txt*. Výsledkem testování každého z OCR programů je tedy vždy sada čtyř textových souborů s názvy *faktura1.txt* až *faktura4.txt*.

K vyhodnocení shodnosti přepsaného textu (tzv. *ground truth*) a rozpoznaného textu byl zvolen následující postup, automatizovaný použitím skriptu napsaného v Pythonu (viz algoritmus 1): každé slovo, které bylo nalezeno v obou textech (očištěných o bílé znaky), bylo z obou těchto souborů odstraněno. Pro každý testovaný OCR software byl následně vytvořen adresář *not_found*, který pro každou ze čtyř faktur obsahuje dva nové textové soubory s příponami *_gt.txt* a *_ocr.txt*.

```
Input: GT: ground truth - přepsaný text
Input: OCR: rozpoznaný text
Output: gt.txt: slova z přepsaného textu, která nebyla rozpoznána
Output: ocr.txt: slova, která byla rozpoznána, ale nejsou
                 z přepsaného textu
for each GTword in GT do
    for each OCRword in OCR do
        if GTword == OCRword then
            GT.remove(GTword)
            OCR.remove(OCRword)
        end
    end
end
gt.txt ← GT
ocr.txt ← OCR
```

Algoritmus 1: Výpočet chybovosti převodu

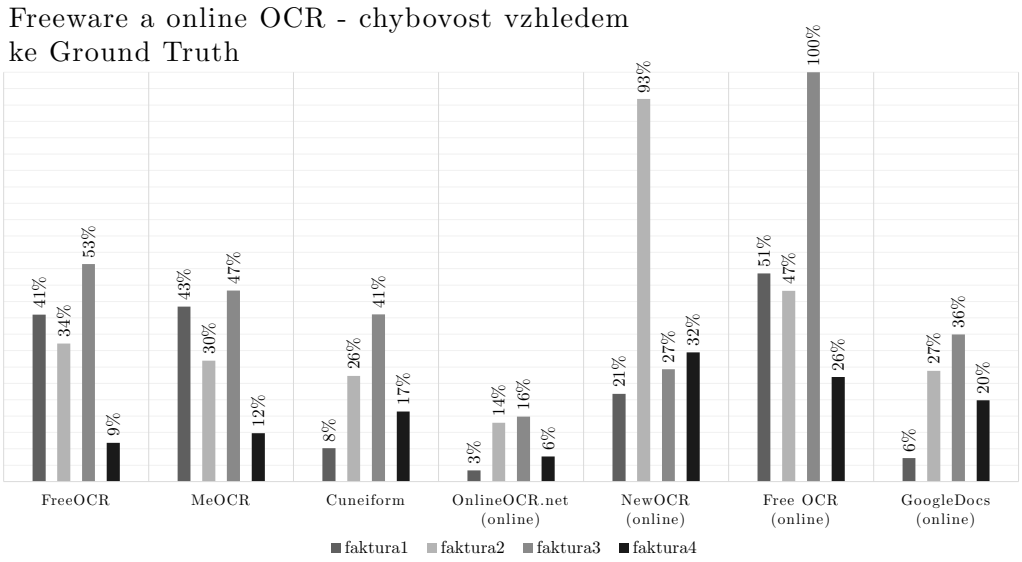
V souboru s příponou *_gt.txt* (např. *faktura1_gt.txt*) se nacházejí pouze ta slova, která byla obsažena v původním ručně přepsaném souboru, ale nebyla rozpoznána (buď vůbec, nebo byla rozpoznána s chybou). Soubor s příponou *_ocr.txt* naopak zahrnuje slova, která byla OCR programem rozpoznána, ale v původním přepsaném textu obsažena nebyla. Může se opět jednat o slova rozpoznaná chybně, případně o „slova“, která vznikla snahou o převod např. čárového kódu. Počet znaků ve všech ručně přepsaných fak-

turách, rozpoznaných textech i souborech v adresáři *not_found* byl uložen do souboru *chars_stat.txt*.

Některé OCR programy byly schopné zaznamenat i hranice jednotlivých sloupců textu a při převodu pak dokázaly udržet text z jedné souvislé oblasti pohromadě. Jiné však převáděly text řádku po řádce bez ohledu na strukturu faktury. Jelikož však cílem tohoto experimentu bylo pouze zjištění přesnosti převodu textu existujícími OCR, bylo pořadí rozpoznaných slov v tomto případě zanedbáno. Následující výsledky experimentu tedy nezohledňují přesnost zachování členění faktury.

Výsledky experimentu

Následující dva grafy (graf 5.1 a graf 5.2) zachycují **chybovost** neplacených i komerčních OCR řešení při rozpoznávání znaků **vzhledem ke ground truth datům**, tzn. jedná se o procentuálně vyjádřený poměr počtu znaků těch slov z přepsané faktury, která nebyla OCR programem rozpoznána (tj. počtu znaků v souboru *_gt.txt* se zbývajícími slovy) a celkového počtu znaků v této původní přepsané faktuře. Např. hodnota 41 % v grafu 5.1 tedy znamená, že 41 % znaků ručně přepsané faktury1 nebylo programem FreeOCR rozpoznáno.

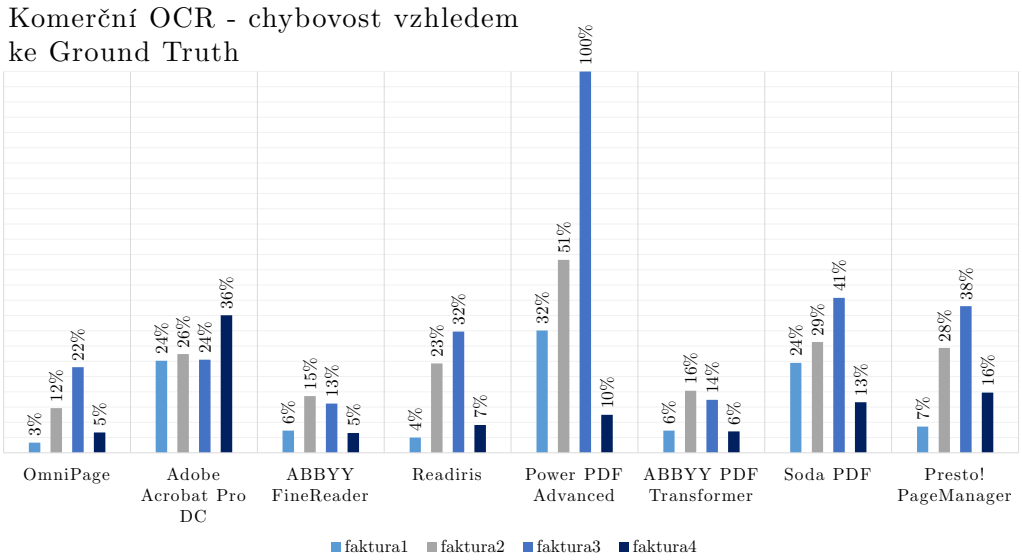


Graf 5.1: Chybovost freeware a online OCR řešení vzhledem ke ground truth datům.

Chybovost 93 % při převodu německé faktury2 nástrojem NewOCR je zapříčiněna tím, že se povedlo převést pouze začátek. Stoprocentní neúspěch

online nástroje Free OCR je způsoben horší čitelností faktury3, kvůli které nebylo možné soubor ani načíst.

S fakturou3 měl problémy i komerční program Power PDF Advanced, kterému se také nepodařilo hůře čitelný text rozpoznat. Poměrně vysoká neúspěšnost převodu (ve srovnání s ostatními OCR) anglické faktury4 programem Adobe Acrobat Pro DC je způsobena tím, že některé bloky textu byly při převodu úplně vynechány.



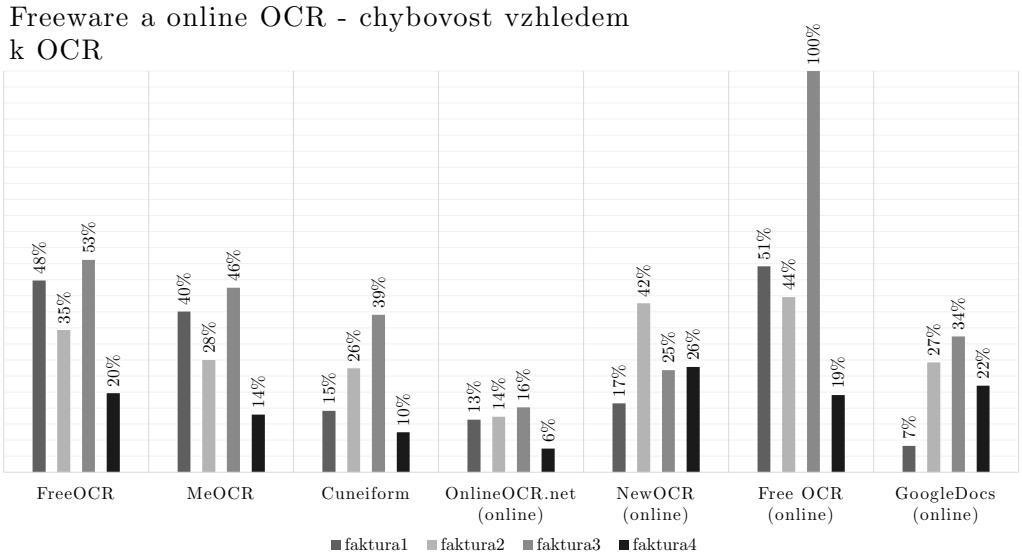
Graf 5.2: Chybovost komerčních OCR řešení vzhledem ke ground truth datům.

Procentuální **chybovost vzhledem k OCR datům** je zobrazena v grafech 5.3 a 5.4. Tato hodnota představuje poměr počtu znaků všech slov, která byla rozpoznána OCR softwarem, ale nenacházela se v původním textu faktury (tj. počtu znaků v souboru *_ocr.txt*) a celkového počtu znaků v rozpoznaném textovém souboru.

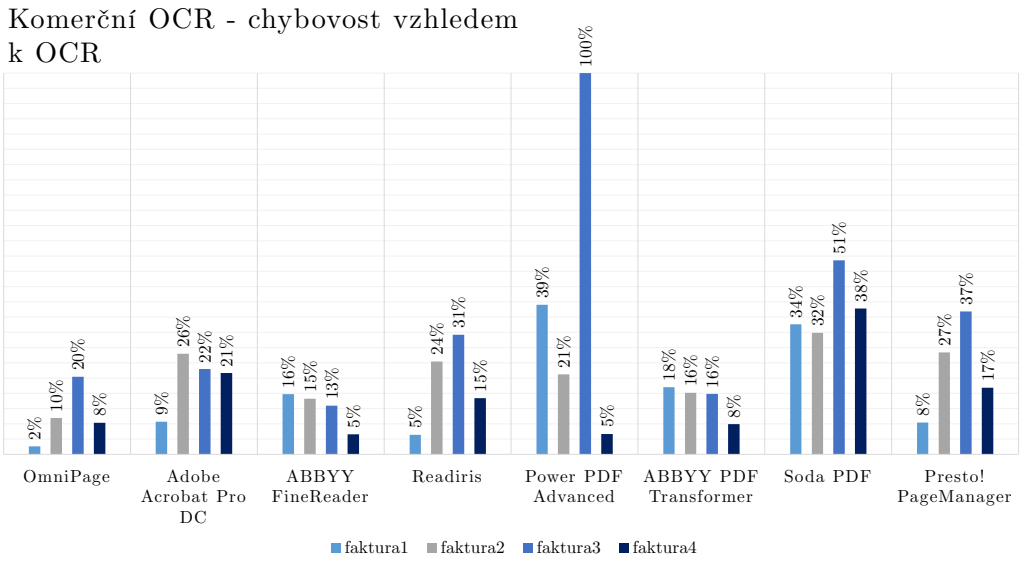
Z grafu 5.3 je tedy např. možné vidět, že 48 % všech znaků faktury1, které byly rozpoznány programem FreeOCR, bylo rozpoznáno buď chybně, nebo úplně navíc (např. při pokusu o převod razítka nebo loga společnosti na text).

Hodnoty grafu 5.3 přibližně odpovídají hodnotám grafu 5.1. Výrazný rozdíl lze zaznamenat u faktury2, která byla nástrojem NewOCR (jak již bylo zmíněno) rozpoznána pouze zčásti (7 % původního textu). Ze všech znaků, které se NewOCR podařilo rozpoznat, však bylo 42 % rozpoznáno nesprávně.

Podobnost můžeme spatřit i mezi hodnotami grafů 5.4 a 5.2, s výraznějším rozdílem např. u faktury4 (Adobe Acrobat Pro DC). Přestože některé



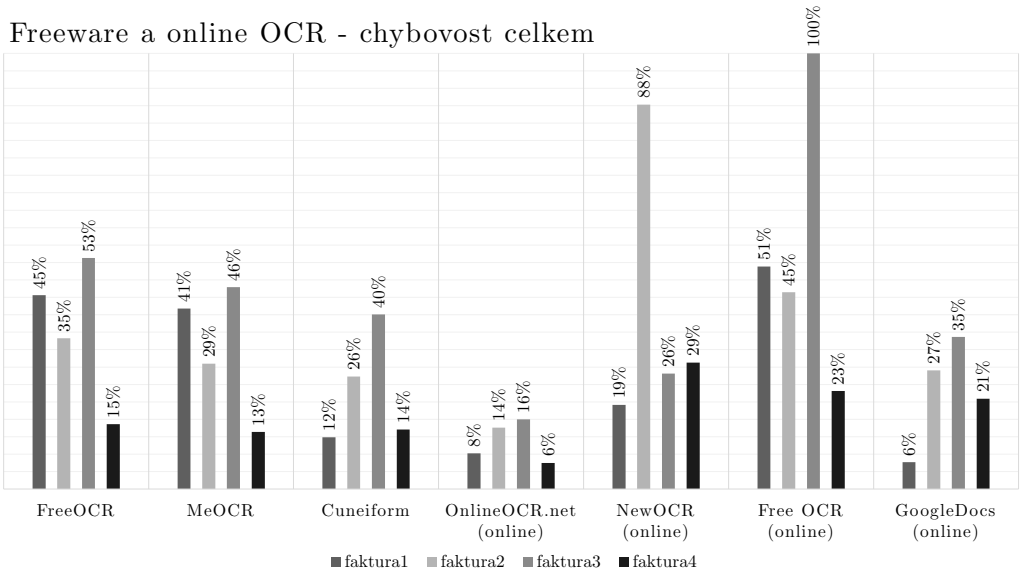
Graf 5.3: Chybovost freeware a online OCR řešení vzhledem k OCR datům.



Graf 5.4: Chybovost komerčních OCR řešení vzhledem k OCR datům.

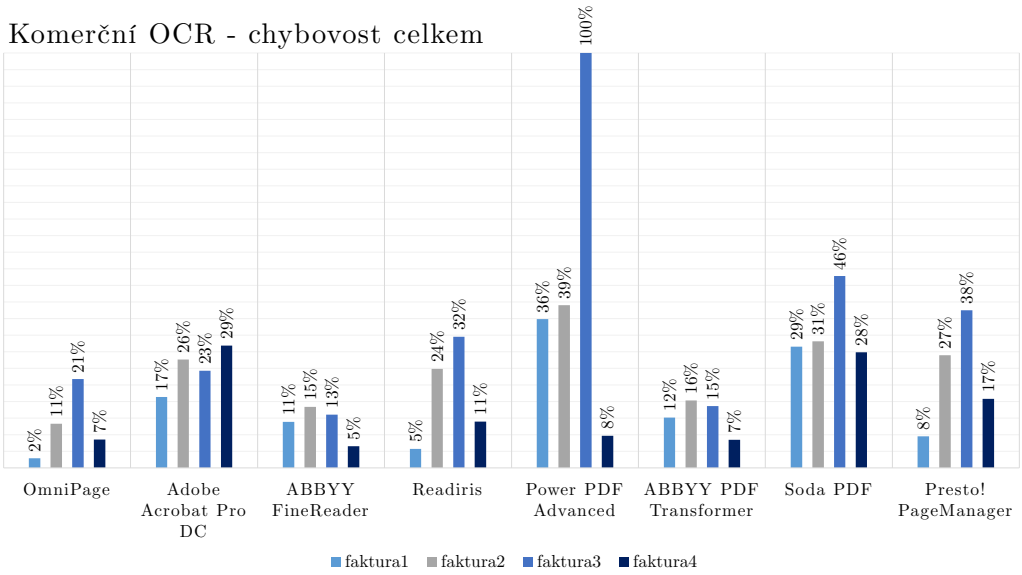
části textu byly při převodu vynechány (chybovost 36 %, viz graf 5.2), většina znaků je převedena správně (nesprávnost 21 %, podrobnější analýzou je však možné zjistit, že většina chyb byla způsobena pokusem o převod razítka).

Grafy 5.5 a 5.6 zachycují **celkovou procentuální chybovost** všech testovaných OCR řešení, tedy poměr součtu počtu znaků v obou souborech se zbylými slovy (soubory s příponami *_gt.txt* a *_ocr.txt*) a součtu počtu znaků v původním přepsaném textu (ground truth) a rozpoznaném textu.



Graf 5.5: Celková chybovost freeware a online OCR řešení.

Je vhodné poznamenat, že jelikož některé z testovaných OCR nástrojů neumožňují převod textu v českém jazyce, byly převody českých faktur otestovány za použití angličtiny. Toto je případ bezplatného FreeOCR a komerčního Soda PDF. S češtinou (a němčinou) se dále potýkal i nástroj MeOCR, který nezvládl zobrazit znaky s diakritikou, a Power PDF Advanced, který měl problémy převést faktury psané v češtině do čistého textu.



Graf 5.6: Celková chybovost komerčních OCR řešení.

Zhodnocení

Tabulka 5.3 zobrazuje přehled zjištěných přesností jednotlivých OCR softwarů. Tato přesnost byla vypočítána podle následujícího vzorce 5.1 jako aritmetický průměr celkových chybovostí všech čtyř faktur, odečtený od 100 % (vyjadřujících bezchybnost):

$$\text{vypočítaná přesnost } [\%] = 100\% - \frac{\sum_{i=1}^{4} score_i}{4}, \tag{5.1}$$

kde $score_i$ vyjadřuje hodnotu chybovosti daného OCR řešení při překladu i-té faktury.

Název	**Vypočítaná přesnost** [%]	**Další informace**
FreeOCR	63	Sloupce je třeba naklikat v požadovaném pořadí, neumí češtinu (při testování použita angličtina).
MeOCR	68	Zachovává sloupce, problém se znaky s diakritikou.
Cuneiform	77	Zachovává strukturu, snadno lze označit, co nechceme převádět a co k sobě patří.
OnlineOCR.net	**89**	Zachovává strukturu i v *.txt*, limitováno velikostí souboru a jejich počtem (max 5 MB a 15 souborů/hod, po registraci až 100 MB).
NewOCR	59	Není limitováno velikostí ani počtem souborů, umožňuje zvolit kombinaci více jazyků. Málokdy rozpozná sloupce, zpravidla převádí řádku po řádce a občas něco vynechá.
Free OCR	45	Nezachovává strukturu, špatná úroveň převodu textu. Limitováno velikostí souboru a jejich počtem (max 2 MB a 10 souborů/hod).
Google Docs	78	Ne vždy správně rozpozná sloupce. Limitováno velikostí souboru (max 2 MB), ale nikoliv počtem stránek.
OmniPage Ultimate	**90**	Používá slovníky, na nerozpoznané znaky se hned ptá, snadná úprava, automatické trénování. Zachovává kompletně formát, občas má problém s tabulkami.
Adobe Acrobat Pro DC	76	Zachovává strukturu, primárně export do Wordu. Snadno lze označit, co nechceme převádět (např. razítko).
ABBYY FineReader Corporate	**89**	Používá slovníky, „málo věrohodné“ znaky a slova zvýrazňuje a umožňuje jejich opravu. Zachovává související bloky textu, ne vždy zaznamená konce řádků. Rozpozná i kombinaci více jazyků.
Readiris 15 Corporate	82	Rychlý a přehledný. V případě více sloupců je třeba je postupně naklikat v požadovaném pořadí, jinak převádí řádku po řádce.
PowerPDF Advanced	54	Problémy s převodem češtiny do textového souboru, němčinu i angličtinu zvládá lépe.
ABBYY PDF Transformer+	**87**	Při převodu nelze vybrat jen části, zvládá kombinaci více jazyků.
Soda PDF	67	Neumí češtinu (při testování použita angličtina). Němčinu i angličtinu zvládá lépe.
Presto! Page Manager	78	Výsledný textový soubor je někdy seřazen opačně, samotný převod textu je poměrně přesný.

Tabulka 5.3: Zhodnocení úspěšnosti testovaných OCR řešení.

Mezi neplacenými OCR řešeními dopadl nejlépe online nástroj Online-OCR.net s vypočítanou přesností 89 %. Jeho přesnost je srovnatelná s těmi nejlepšími z komerčních nástrojů, tj. OmniPage Ultimate s úspěšností 90 %, ABBYY FineReader s 89 % a ABBYY PDF Transformer+ s 87 %.

5.2 Nástroje pro vytěžování dat z dokumentů

Převedení faktury do digitální podoby, tj. rozpoznání znaků faktury je důležitým prvním krokem před zahájením samotného vytěžování dat.

Trh s vytěžovacím software se neustále rozvíjí. Aktuálně dostupné nástroje v sobě zpravidla zahrnují oba kroky (tj. rozpoznání znaků i vytěžování dat) zároveň. Přední místa v práci s účetními dokumenty zastávají společnosti Kofax a ABBYY, na českém trhu má své místo také pražská společnost SOCOS IT s.r.o. se svým produktem DOCU-X OCR.

5.2.1 Kofax

Jedním z mnoha produktů, které americká společnost Kofax nabízí, je Kofax ReadSoft Online^{TM}. Toto cloudové řešení umožňuje automaticky zachycovat a dále zpracovávat data faktur. Podle Kofaxu jde o řešení, které je snadno integrovatelné do různých ERP či jiných systémů. Kofax ReadSoft Online zohledňuje při vytěžování dat specifika jednotlivých zemí (sazba daně, formáty dat, měna, ...) a kromě záhlaví faktury dokáže pracovat i s jednotlivými položkami faktury [22].

5.2.2 ABBYY

Elektronickému zpracování dokumentů se věnuje také společnost ABBYY. Některé jejich produkty byly testovány již v rámci experimentu týkajícího se rozpoznávání znaků (ABBY FineReader Corporate a ABBYY PDF Transformer+). Na vytěžování dat z faktur je zaměřen produkt ABBYY FlexiCapture for Invoices, jehož použitím je dle ABBYY možno snížit náklady na zpracování faktur až dvakrát, a to při současném 50% zrychlení celého procesu [10].

5.2.3 SOCOS IT

O automatizované vytěžování firemních dokumentů se dokáže postarat i produkt DOCU-X OCR vyvíjený českou společností SOCOS IT s.r.o. Díky využití učících algoritmů se tento software postupně učí, rozpoznává různé

šablony strukturovaných či polostrukturovaných dokumentů a zdokonaluje tak své výsledky vytěžování. Není tedy třeba se omezovat jen na faktury, neboť DOCU-X OCR se dokáže naučit vytěžovat i např. objednávky či dodací listy [78].

5.2.4 Zhodnocení použitelnosti

Vzhledem ke směřování řešení směrem do cloudu a potřebě integrace vytěžování dat do vyvíjeného SharePoint add-inu jsme se místo výběru kompletního řešení zaměřili na nalezení vhodného poskytovatele vytěžování dat jako služby.

Ani s jedním z poskytovatelů uvedených v podkapitolách 5.2.1 až 5.2.3 se bohužel nepodařilo domluvit bezplatné vyzkoušení jejich řešení pro účely této práce. Následující kapitola 6 je proto zaměřena na návrh vlastního jednoduchého vytěžování, navazujícího na experiment týkající se rozpoznávání znaků popsaný v kapitole 5.1.

6 Návrh vlastního jednoduchého vytěžování

V okamžiku, kdy již máme k dispozici fakturu v digitální podobě, se můžeme začít věnovat dodávání významu získanému textu – vytěžování dat.

U každé přijaté faktury je vždy pro společnost důležité určit, vůči komu tím vzniká závazek, tj. kdo je jejím dodavatelem. Informace o svých dodavatelích každá společnost nějakým způsobem uchovává, a to zpravidla ve svém informačním systému. Ve chvíli, kdy společnost eviduje nově přijatou fakturu, je tedy vhodné zjistit, zda se jedná o nového, dosud neznámého dodavatele, nebo o dodavatele, který je již v systému zapsán.

Popisu experimentu zaměřeného na vyzkoušení několika jednoduchých metod pro identifikaci dodavatelů jednotlivých testovaných faktur na základě dodavatelské databáze je věnována kapitola 6.1, zhodnocení výsledků je obsaženo v kapitole 6.2. Na popis metod vytěžování číselných údajů z faktur je zaměřena kapitola 6.3, dosažené výsledky jsou shrnuty v kapitole 6.4.

Implementace jednotlivých metod vytěžování a jejich měření bylo realizováno v programovacím jazyce Python (verze 2.7), na stroji s operačním systémem Windows 10 (64bit), procesorem Intel® Core™ i5-2400S 2.50GHz a 8 GB RAM.

6.1 Metody vytěžování dodavatelů

Vstupními daty pro tento experiment bylo 39 naskenovaných faktur[1], převedených do textové podoby pomocí nástroje OnlineOCR.net, který ze všech bezplatných testovaných OCR řešení vykazuje při převodu faktur nejvyšší přesnost (viz tabulka 5.3). Ke každé z těchto faktur byla přidružena informace o jejím skutečném dodavateli, aby bylo později možné ověřit, zda byl dodavatel testovanými metodami úspěšně identifikován.

Dalším vstupním prvkem důležitým pro porovnání úspěšnosti metod vytěžování dodavatelů pak byly informace o zhruba 4000 dodavatelích, exportované z ERP systému a uložené ve formátu *.csv*, kde jednomu dodavateli odpovídá jedna řádka tohoto textového souboru. V tomto souboru bylo zapotřebí určit sloupce, podle kterých je možné z textu faktury příslušného

[1]Stejně jako v případě experimentu popsaného v kapitole 5 byl i tento experiment prováděn na reálných datech (fakturách, *.csv* souboru). Pro vyzkoušení experimentu použijte a případně doplňte ukázková data dostupná z přiloženého CD.

dodavatele jednoznačně identifikovat. Pro všechny testované metody byly tyto sloupce zvoleny shodně, aby bylo možné srovnat jejich úspěšnost.

Soubor *.csv* byl upraven tak, aby obsahoval těchto pět sloupců: *NAME_1*, *STREET*, *CITY*, *REGISTRATION_ID* a *VAT_NUMBER*, tj. sloupce obsahující název dodavatele, jeho adresu, identifikační číslo (IČO) a daňové identifikační číslo (DIČ). Jedna řádka tohoto souboru pak může vypadat např. tímto způsobem: `'CCA Group a.s.', 'Karlovo nám. 17', 'Praha 2', '25695312', 'CZ25695312'`.

Texty faktur a jednotlivé řádky zdrojového souboru bylo nutné pro vstup do metod testovaných v Pythonu připravit následujícím způsobem: všechny nepísmenné a nečíselné znaky, stejně jako dvojité či vícenásobné mezery, byly nahrazeny jednoduchou mezerou, podle které pak byly výsledné textové řetězce rozděleny. Připravená faktura má tedy podobu kolekce obsahující jednotlivá slova očištěná o speciální znaky. Stejným způsobem byly upraveny i jednotlivé řádky zdrojového souboru odpovídající vždy informacím o jednom konkrétním dodavateli, tzn. v případě např. víceslovného názvu společnosti je každé takové slovo uloženo v kolekci zvlášť a velikost výsledné kolekce tedy může být větší než výchozí počet sloupců (tj. pět). Logicky se tedy mohou velikosti kolekcí pro jednotlivé dodavatele lišit. Výše uvedená řádka zdrojového *.csv* souboru s dodavateli proto vypadá po popsané úpravě takto: `'CCA', 'Group', 'a', 's', 'Karlovo', 'nám', '17', 'Praha', '2', '25695312', 'CZ25695312'`.

V rámci experimentu byla na takto připravených datech pomocí několika různých metod porovnána podobnost obou připravených textových vstupů, tj. podobnost vždy jedné faktury postupně se všemi řádky zdrojového souboru (tzn. se všemi dodavateli). Při implementaci testovaných metod byly využity knihovny pro porovnávání podobnosti řetězců Levenshtein a FuzzyWuzzy a knihovny ssdeep, lshhdc, lshash a pyflann, podporující Context Triggered Piecewise Hashing, resp. Locality-Sensitive Hashing (viz dále).

Pro každou z 39 faktur pak bylo postupně všem dodavatelům přiřazeno skóre podobnosti (viz algoritmus 2), podle kterého pak byli dodavatelé sestupně seřazeni. Pro srovnání úspěšnosti jednotlivých metod vytěžování byly definovány a sledovány tři typy úspěchů:

úspěch$_1$ představuje situaci, kdy nejvyšší skóre podobnosti vykazuje skutečný dodavatel,

úspěch$_{10}$ znamená, že skutečný dodavatel sice nezískal nejvyšší skóre, ale umístil se mezi prvními deseti nejpravděpodobnějšími,

úspěch$_{20}$ značí umístění alespoň mezi prvními dvaceti dodavateli.

Input: invoice: text faktury
Input: csv: zvolené sloupce dodavatelské databáze
Output: score: vektor vypočítaných skóre podobnosti
for *each i, row in csv* **do**
| score[i] ← count_similarity(invoice, row)
end
return *score*

Algoritmus 2: Výpočet skóre podobnosti

6.1.1 Levenshtein$_1$

Jako první byla otestována funkce *ratio(string1, string2)* z knihovny Levenshtein [6], která počítá podobnost dvou řetězců (zadaných jako parametry) v závislosti na Levenshteinově vzdálenosti. Tato vzdálenost dvou řetězců je definovaná jako minimální počet operací vložení, mazání a substituce jednotlivých znaků řetězce, pomocí kterých lze jeden řetězec převést na druhý. Funkce *ratio()* vrací skóre podobnosti z intervalu $\langle 0, 1\rangle$, kde 0 znamená, že řetězce nemají žádný společný znak a 1 naopak vyjadřuje jejich absolutní shodnost. Dokumentace této knihovny je k dispozici ve zdroji [4].

Pro každou fakturu f a pro každého dodavatele d (tj. pro každou řádku ze zdrojového *.csv* souboru) byla sestrojena tabulka o rozměrech $m \times n$, kde m je celkový počet slov obsažených ve faktuře f a n je počet slov v řádce d. Tato tabulka obsahuje vypočítanou hodnotu podobnosti každého slova z f s každým slovem z d.

V každém sloupci pak byla nalezena maximální hodnota, tj. pro každé slovo z řádky d zdrojového souboru byla nalezena nejvyšší shoda s některým slovem z faktury f. Toto maximum bylo uloženo do n-rozměrného vektoru dílčích pravděpodobností p_i. Podle vzorce 6.1 pak bylo určeno, s jakou pravděpodobností je dodavatelem uvedeným na faktuře f právě dodavatel d.

$$\text{skóre podobnosti} = \prod_{i=1}^{n} p_i \tag{6.1}$$

6.1.2 Levenshtein$_2$

Jako další byla vyzkoušena funkce *seqratio(string_seq1, string_seq2)*, opět z knihovny Levenshtein [6]. Tato funkce vrací stejnou návratovou hodnotu jako předchozí funkce *ratio*, ale na rozdíl od ní požaduje jako parametry celé sekvence řetězců. Počítá tedy podobnost celého textu jedné faktury f s celou řádkou d obsahující informace o jednom dodavateli.

6.1.3 FuzzyWuzzy$_1$

Pro popis funkce *token_set_ratio(string1, string2)* z modulu *fuzz* knihovny FuzzyWuzzy je třeba definovat následující množiny [3]:

t_0 abecedně seřazená množina slov společných pro oba řetězce,

t_1 sjednocení t_0 a seřazené množiny zbývajících slov prvního řetězce,

t_2 sjednocení t_0 a seřazené množiny zbývajících slov druhého řetězce.

Porovnávány jsou všechny dvojice těchto množin a za výsledné skóre podobnosti vstupních řetězců je pak považován nejlepší z těchto výsledků. Jelikož slova z množiny t_0 jsou společná pro množinu t_1 i t_2, platí, že skóre podobnosti obou řetězců se zvyšuje, pokud množina společných slov t_0 tvoří podstatnou část jednoho z řetězců, nebo pokud jsou si obě množiny zbývajících slov podobné.

Návratovou hodnotou funkce *token_set_ratio(string1, string2)* je číslo z intervalu $\langle 0, 100 \rangle$, kde 0 znamená, že se jedná o dva naprosto odlišné řetězce a hodnota 100 naopak odpovídá identickým řetězcům, přičemž zanedbává možnou duplicitu jednotlivých slov.

6.1.4 FuzzyWuzzy$_2$

Další vyzkoušenou funkcí z knihovny FuzzyWuzzy, v tomto případě z modulu *process*, byla funkce *extractOne(string, string_sequence)*, která porovnává řetězec zadaný jako první parametr se sekvencí řetězců zadanou jako druhý parametr. Z této sekvence vybere řetězec s nejvyšší podobností a vrací ho jako dvojici tvaru *(string, score)*, kde *score* je hodnota z intervalu $\langle 0, 100 \rangle$ jako u předchozí metody.

V tomto případě bylo postupně porovnáváno každé slovo z řádku d ve zdrojovém *.csv* souboru s textem celé faktury f. Pro každého dodavatele d bylo sledováno, kolikrát se vyskytlo *score* s hodnotou 100. Výsledné skóre podobnosti bylo následně pro každého dodavatele vypočítáno podle vzorce 6.2 a vyjadřuje tedy, kolik procent slov z řádky d bylo nalezeno i ve faktuře f.

$$\text{skóre podobnosti} = \frac{\text{počet výskytů } score \text{ } 100}{\text{celkový počet slov v } d} \tag{6.2}$$

6.1.5 CTPH

U běžného hashování platí, že i malá změna původních hashovaných dat způsobí velkou změnu jejich výsledného otisku (hashe). Context Triggered

Piecewise Hashing (CTPH) však pro dostatečně podobná data vytváří stejné otisky a díky tomu stačí při zjišťování podobnosti řetězců porovnávat pouze jejich výsledné hashe [23].

Práci s CTPH v Pythonu umožňuje knihovna ssdeep [9]. Pomocí funkce *hash()* této knihovny byl pro každé slovo faktury f a každé slovo z jedné řádky d ve zdrojovém souboru vytvořen a uložen hash. Celková podobnost f a d pak byla vypočítána podle následujícího vzorce 6.3 (Jaccardův index podobnosti).

$$\text{Jaccardův index podobnosti} = \frac{|F \cap D|}{|F \cup D|}, \tag{6.3}$$

kde F je množina hashů slov faktury f a D je množina hashů slov z řádky d. Výsledkem je hodnota z intervalu $\langle 0, 1 \rangle$, kde 1 znamená, že f a d jsou shodné [24].

6.1.6 LSH_1

Také Locality-Sensitive Hashing (LSH) hashuje podobná data do stejných nebo podobných otisků, důsledkem malé změny původního řetězce je tedy žádná nebo pouze malá změna hashe. Při porovnávání podobnosti dvou textových řetězců lze proto tyto řetězce opět rozdělit na menší části, zahashovat je a porovnávat pouze jejich výsledné otisky. LSH však vyžaduje, aby měly všechny tyto podřetězce (tzv. *shingle*) stejnou délku. Vstupem pro testované LSH metody tedy nebyly kolekce jednotlivých slov jako v předchozích případech, ale dva řetězce (jeden odpovídající celému obsahu faktury f a druhý jednomu řádku zdrojového souboru d), které pak byly podle potřeby jednotlivých metod dále zpracovány. LSH je věnováno několik kapitol publikace [24].

Jednou z knihoven implementujících LSH je knihovna lshhdc [8]. Tato knihovna obsahuje funkci *hshingle(s, k)*, která z řetězce s vytvoří překrývající se podřetězce délky k a vrací rovnou jejich hashe. Právě touto funkcí byla zpracována faktura f i každý řádek d. Podobnost těchto dvou množin hashů, potažmo původních dvou řetězců pak byla vypočítána opět pomocí Jaccardova indexu podobnosti (vzorec 6.3).

Otázkou však je, jak velké k zvolit. Následující tabulka 6.1 zobrazuje, jaké úspěšnosti při identifikaci dodavatelů dosáhla tato metoda za použití několika různých velikostí k.

Při použití příliš malého k ($k = 2$) je většina těchto dvojic znaků jednoho řetězce nalezena i v druhém řetězci a určení správného dodavatele je proto obtížné. Pro $k = 3$ už se úspěšnost výrazně zvyšuje, a to z 2,6 % na 76,9 %. Nejlepších výsledků dosahuje metoda při použití $k = 4$, kdy je úspěšnost

	Počet úspěchů$_1$	Procento úspěchů$_1$	Počet úspěchů$_{10}$	Procento úspěchů$_{10}$	Počet úspěchů$_{20}$	Procento úspěchů$_{20}$	Celk. doba zpracování	Průměrná doba/faktura
$k = 2$	1	2,6 %	7	18,0 %	13	33,3 %	30,21 s	0,77 s
$k = 3$	30	76,9 %	37	94,9 %	37	94,9 %	35,22 s	0,90 s
k = 4	33	**84,6 %**	39	100 %	39	100 %	37,64 s	0,97 s
$k = 5$	33	84,6 %	39	100 %	39	100 %	38,80 s	0,995 s
$k = 6$	33	84,6 %	38	97,4 %	39	100 %	40,37 s	1,04 s

Tabulka 6.1: Srovnání *lshhdc* pro různé hodnoty k.

identifikace dodavatele ještě o něco vyšší (84,6 %) a průměrný čas na zpracování jedné faktury je 0,97 s. S rostoucím k ($k = 5, k = 6$) dokáže metoda správně určit stále stejný počet dodavatelů (33 z 39), ale průměrná doba na zpracování už se zvyšuje.

Hodnota $k = 4$ byla použita i v následujících metodách LSH_2 a LSH_3.

6.1.7 LSH_2

Další vyzkoušenou knihovnou podporující LSH je lshash [7], která na rozdíl od předchozího případu vyžaduje jako vstup celá čísla, nikoli řetězce. Pomocí funkce *index()* z této knihovny byly tzv. indexovány všechny k-tice čísel odpovídající hodnotám jednotlivých znaků podřetězců délky k zpracovávané faktury.

S těmito k-ticemi čísel pak voláním funkce *query()* postupně porovnáváme všechny k-tice čísel odpovídající jednomu dodavateli. Výsledkem volání *query(k-tice, pocet_vysledku)* je libovolný počet dvojic (určený parametrem *pocet_vysledku*) ve tvaru *(indexovaná k-tice, vypočítaná vzdálenost)*. Pro výpočet vzdálenosti indexované a dotazované k-tice pak lze použít např. Eukleidovu, Hammingovu nebo kosinovou vzdálenost.

Aby bylo možné říci, že dodavatel d je opravdu dodavatelem uvedeným na faktuře, pak by co nejvíce vypočítaných vzdáleností nejlepších shod (tj. *pocet_vysledku = 1*) mělo být rovno nule. Výsledné skóre podobnosti je pro každého dodavatele vypočítáno podle vzorce 6.4.

$$\text{skóre podobnosti} = \frac{\text{počet výskytů vzdálenosti 0}}{\text{celkový počet } k\text{-tic}} \tag{6.4}$$

6.1.8 LSH_3

Vyzkoušena byla také knihovna pyflann [5], která je v několika ohledech podobná předchozí knihovně lshash, ale pracuje s čísly typu float. Analogií k funkci *index()* knihovny lshash je v této knihovně funkce *build_index()*, pro porovnání k-tice s indexovanými prvky voláme funkci *nn_index()*, pro

kterou lze také stanovit počet vrácených nejbližších shod.

Pro výpočet skóre podobnosti jsme zvolili shodný přístup jako u předchozí metody, tzn. zjišťovali jsme, jak často byla vrácena vzdálenost 0 a skóre pro každého dodavatele jsme opět vypočítali podle vzorce 6.4.

6.2 Zhodnocení metod vytěžování dodavatelů

Úspěšnost a doba zpracování jednotlivých metod při identifikaci dodavatelů všech 39 testovaných faktur jsou shrnuty v tabulce 6.2.

Nejlepších výsledků při identifikaci správného dodavatele na prvním místě dosahuje metoda LSH_3 s úspěšností 87,2 %. Její nevýhodou je však vysoká průměrná doba zpracování (34,1 s). Jako nejvhodnější se tedy na základě tabulky 6.2 jeví metoda LSH_1, jejíž úspěšnost je sice nepatrně nižší (84,6 %), ale potřebuje v průměru pouze 0,97 sekundy na to, aby z textu faktury správně identifikovala dodavatele. Rychlejší průměrnou dobu potřebnou ke zpracování jedné faktury pak vykazuje už jen metoda CTPH (0,5 s), ale její úspěšnost (25,6 %) je ve srovnání s ostatními nedostatečná.

	Počet úspěchů$_1$	**Procento úspěchů$_1$**	**Počet úspěchů$_{10}$**	**Procento úspěchů$_{10}$**	**Počet úspěchů$_{20}$**	**Procento úspěchů$_{20}$**	**Celk. doba zpracování**	**Průměrná doba/faktura**
Levenshtein$_1$	28	71,8 %	35	89,7 %	38	97,4 %	379,3 s	9,7 s
Levenshtein$_2$	12	30,8 %	22	56,4 %	27	69,2 %	82,8 s	2,1 s
FuzzyWuzzy$_1$	20	51,3 %	39	100 %	39	100 %	285,1 s	7,3 s
FuzzyWuzzy$_2$	12 z 15	80 %	15 z 15	100 %	15 z 15	100 %	17097,5 s	1139,8 s
CTPH	10	25,6 %	25	64,1 %	27	69,2 %	20,8 s	0,5 s
LSH$_1$($k = 4$)	33	**84,6 %**	39	100 %	39	100 %	37,6 s	**0,97 s**
LSH$_2$($k = 4$)	8 z 10	80 %	10 z 10	100 %	10 z 10	100 %	10857,8 s	1085,8 s
LSH$_3$($k = 4$)	34	**87,2 %**	39	100 %	39	100 %	1329,4 s	**34,1 s**

Tabulka 6.2: Zhodnocení metod vytěžování.

6.3 Metody vytěžování čísel

Kromě dodavatelů byly z faktur dále vytěžovány některé z důležitých číselných údajů, jako je např. datum vystavení a splatnosti faktury, variabilní symbol nebo částka, na kterou byla faktura vystavena.

Experiment byl prováděn nad textem získaným způsobem popsaným v kapitole 6.1, tj. optickým rozpoznáním textu naskenované faktury. Důležitým krokem při přípravě textu pro vytěžování čísel bylo zachovat pouze slova obsahující čísla, a to jak výhradně číselné řetězce, tak i čísla v kombinaci s jinými znaky. Zachovány také zůstaly různé měny a jejich symboly (např. Kč, CZK, EUR, €, $, . . .) a některé další znaky („.“, „,“, „-“ a „/“).

Následující podkapitoly jsou věnovány postupně všem číselným údajům, na jejichž vyhledávání jsme se v rámci experimentu zaměřili.

6.3.1 Čárový kód

Jedním z číselných údajů, které se vyskytují na poskytnutých testovaných fakturách, jsou čárové kódy. Čárové kódy jsou na přijaté faktury lepeny ručně jejich příjemcem, v některých případech se tedy může stát, že faktura čárový kód neobsahuje.

Dle vyjádření zadavatele jsou čárové kódy na testovaných fakturách definovány jako řetězce ve tvaru KCS*xxxxxxx*, resp. KCSE*xxxxxxx*, kde x reprezentuje právě jednu číslici (např. KCS00000061). Vzhledem k tomu, že se jedná o číselný údaj s přesně danou strukturou, bylo možné k vyhledání čárových kódů v textu použít regulární výrazy.

6.3.2 IČO dodavatele

Dalším údajem reprezentovaným na faktuře jako číslo je identifikační číslo (IČO) dodavatele, skládající se z 8 číslic. Pomocí regulárního výrazu vybereme *kandidáty* na IČO, tedy řetězce odpovídající tomuto vzoru. Z těchto kandidátů následně odstraníme čísla, která

- představují IČO odběratele (tj. IČO firmy, která fakturu přijala a dále ji uchovává a zpracovává),
- nemohou být platným identifikačním číslem (ověřeno podle pravidel zveřejněných ministerstvem vnitra [71]).

Mezi kandidáty na IČO dodavatele faktury je dále zařazeno IČO dodavatele, který byl při vytěžování dodavatelů (viz předchozí kapitoly 6.1 a 6.2) určen jako nejpravděpodobnější. Všichni tito kandidáti jsou pak postupně ohodnoceni body podle toho, jakým způsobem se mezi kandidáty na IČO dostali, tj. zda byli nalezeni na faktuře pomocí regulárního výrazu (nejvyšší počet bodů), nebo zda odpovídají nejpravděpodobnějšímu záznamu z databáze dodavatelů (nižší počet bodů, dodavatel nemusel být určen správně). U kandidáta, který byl nalezen jak v textu faktury, tak v záznamu nalezeného dodavatele tedy existuje vyšší pravděpodobnost, že je hledaným identifikačním číslem, než u kandidáta, který byl sice nalezen pomocí regulárního výrazu, ale jemuž neodpovídá IČO nejpravděpodobnějšího dodavatele.

V případě chybné identifikace dodavatele není ani IČO tohoto dodavatele (tj. chybně určené) s vysokou pravděpodobností obsaženo v textu testované faktury. V seznamu kandidátů je tedy toto IČO hodnoceno nízkým počtem

bodů, zatímco všichni ostatní kandidáti (nalezení pomocí regulárního výrazu) mají vyšší (a shodný) počet bodů. Nelze ale jednoznačně rozhodnout, u kterého z kandidátů nalezených regulárním výrazem je nejvyšší pravděpodobnost, že je tím správným identifikačním číslem dodavatele.

K upřesnění bylo využito jednoduchého přístupu vycházejícího z předpokladu, že v případě českých firem se jejich IČO shoduje s číselnou částí daňového identifikačního čísla (DIČ), tj. bez prefixu CZ. Jedinou výjimku tvoří OSVČ, jejichž IČO odpovídá jejich rodnému číslu.

Po nalezení pravděpodobného DIČ dodavatele (viz další podkapitola) jsou tedy některým z kandidátů na IČO zpětně přidělovány dodatečné body. Číselná část DIČ je porovnána s kandidáty na IČO a v případě shody je příslušný kandidát ohodnocen ještě jedním bodem. V některých testovaných případech právě tento přístup pomohl určit správné IČO z více kandidátů se shodným počtem bodů.

6.3.3 DIČ dodavatele

Daňové identifikační číslo je tvořeno dvěma až čtyřmi znaky, určujícími zemi původu, následovanými osmi až dvanácti číslicemi (podle země). Kandidáty na DIČ určujeme podobným způsobem jako v případě IČO, tj. regulárním výrazem a záznamem z databáze a podle způsobu určení je i ohodnotíme. Odstraněny jsou opět řetězce odpovídající DIČ odběratele.

Další dodatečný bod mohou kandidáti na DIČ získat v tom případě, že se (až na prefix CZ) shodují s některým z kandidátů na IČO (před upřesňujícím krokem, viz předchozí podkapitola).

6.3.4 Variabilní symbol

V případě určování variabilního symbolu faktury je situace o něco složitější, a to z toho důvodu, že variabilní symbol nemá pevně daný tvar (může to být libovolné, maximálně desetimístné číslo). Pokud není na faktuře variabilní symbol explicitně uveden, obvykle je za něj považováno číslo faktury, s odstraněnými znaky typu „-“, „/“, apod.

Z kandidátů na variabilní symbol získaných pomocí regulárního výrazu jsou odstraněni ti, kteří jsou obsaženi v číselné části čárového kódu a v IČO, DIČ a PSČ dodavatele (určeného jako nejpravděpodobnější), resp. odběratele.

6.3.5 Číslo objednávky

Podle podkladů dodaných zadavatelem jsou za čísla objednávky považovány řetězce ve tvaru 26*xxxxx*, resp. 28*xxxxx*, kde x představuje právě jednu číslici, např. tedy 26000059. Tomuto vzoru byl přizpůsoben i regulární výraz použitý pro vyhledávání čísel objednávek v textu faktur. Pouze u 10 faktur z 39 je však tento tvar dodržen, což se projevilo i na výsledném skóre (viz 6.4). Čísla objednávek některých dalších faktur totiž např. překračují délku 8 znaků, obsahují i písmena a na některých fakturách není číslo objednávky vůbec uvedeno.

V případě, že faktura obsahuje IČO začínající dvojčíslím 26 nebo 28, by mohlo být chybně považováno za možné číslo objednávky. Z tohoto důvodu je nutné tyto případy rozeznávat a identifikační čísla ze seznamu kandidátů na čísla objednávky vyřadit.

6.3.6 Datum vystavení a datum splatnosti

Dalšími důležitými číselnými údaji vyskytujícími se na fakturách jsou datum vystavení faktury a datum splatnosti. Jelikož různé země používají pro vyjádření data různé formáty, je zapotřebí tyto formáty rozlišit a následně sjednotit.

V testovaných fakturách se vyskytují data ve formátech *den měsíc rok (DMY)* i *rok měsíc den (YMD)*, a to v kombinaci s různými oddělovači („.“, „/“, „-“ a „:“). Všechna tato data byla nalezena opět pomocí regulárních výrazů a převedena na jednotný formát ve tvaru *den měsíc rok*, používaný v našich podmínkách. U dnů a měsíců byly odstraněny případné nadbytečné nuly a roky obsahující pouze dvě číslice byly převedeny na celý čtyřmístný tvar[2] (např. 01.01.99 → 1.1.1999).

Jako datum vystavení faktury bylo označeno nejstarší nalezené datum, jako datum splatnosti pak to nejnovější. Jedná se o velmi primitivní způsob, který samozřejmě není 100% bezchybný a je zde obrovský prostor pro zlepšení. V některých případech faktury obsahují např. informaci o datu zapsání společnosti do obchodního rejstříku, a v takovém případě je pak toto datum chybně určeno jako datum vystavení faktury. Tento přístup selhává také v případě, že je na faktuře místo konkrétního data splatnosti uvedeno, že faktura je splatná do x dnů.

[2]Hodnoty do 20 se převádí na rok 2000, v opačném případě na 1900.

6.3.7 Částky

Stejně jako se může datum vyskytovat v různých formátech, i částky na faktuře mohou být v různém tvaru. Částka může obsahovat symbol nebo kód měny před číslem, za ním, a nebo ho nemusí obsahovat vůbec. Dále se jednotlivé tvary částek mohou lišit použitím oddělovače tisíců („.“, mezera, nebo nic) a oddělovače desetinných míst („.“, nebo „,“).

Pro nalezení všech částek v textu faktury jsme použili několik regulárních výrazů a nalezené výsledky jsme upravili na jednotný formát – žádný oddělovač tisíců, desetinná místa jsou oddělena desetinnou tečkou a měna je uvedena až za číslem, nikoliv před ním. Sjednotili jsme také použití symbolů a kódů měn, např. EURO i € jsou nahrazeny označením EUR a místo Kč je použito CZK.

V některých případech se může stát, že u žádné z částek nalezených regulárních výrazem není uvedena měna, např. pokud byl text nesprávně rozpoznán OCR programem, nebo pokud je měna uvedena na faktuře v záhlaví tabulky a nikoliv před nebo za samotným číslem (a proto není zachycena regulárním výrazem). V takovém případě je zapotřebí projít text faktury upravený způsobem popsaným na začátku kapitoly 6.3 (tj. pouze řetězce obsahující čísla, měny a některá interpunkční znaménka) a ověřit v něm výskyt jednotlivých kódů měn[3].

Pokud byl v textu nalezen právě jeden výskyt měny, je tato měna přiřazena ke všem rozpoznaným částkám. Jestliže však bylo měn nalezeno více, nelze jednoznačně rozhodnout, ke kterým částkám která měna patří. V případě, že na faktuře není měna uvedena vůbec, pro zjednodušení předpokládáme, že faktura přišla od tuzemského dodavatele, a je tedy v českých korunách.

Za nejpodstatnější částku faktury je možné považovat celkovou částku, na kterou je faktura vystavena. Tou by logicky měla být nejvyšší nalezená částka, v ideálním případě obsahující i kód měny, ale ne vždy tomu tak je. Nesprávnost je často zapříčiněna chybným rozpoznáním textu OCR softwarem, označením čísla vyhovujícího regulárnímu výrazu za částku, přestože částkou není (různé kódy výrobků apod.), případně přítomností vysoké částky s jiným významem (např. výše kapitálu společnosti).

[3]Program je přizpůsoben měnám CZK, EUR, GBP a USD, které se v testovaných fakturách vyskytují.

6.4 Zhodnocení metod vytěžování čísel

Správnost vytěžených číselných informací z testovaných faktur je vypočítána podle následujícího vzorce 6.5 (F1 score). Výsledkem F1 score je hodnota z intervalu $\langle 0, 1 \rangle$, kde 1 vyjadřuje 100% úspěšnost.

$$\text{F1 score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{6.5}$$

kde *precision* (viz vzorec 6.6) vyjadřuje, jaká část odpovědí, které byly nalezeny, byla nalezena správně a *recall* (viz vzorec 6.7) říká, jaká část všech odpovědí, které měly být nalezeny, byla opravdu správně určena [15].

$$\text{Precision} = \frac{\text{True positives}}{\text{Predicted as positive}} = \frac{\text{True positives}}{\text{True positives} + \text{False positives}} \tag{6.6}$$

$$\text{Recall} = \frac{\text{True positives}}{\text{Actual positive}} = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}} \tag{6.7}$$

	Precision	**Recall**	**F1 score**
Čárový kód	0,833	0,270	0,408
IČO	1,000	1,000	1,000
DIČ	0,973	0,947	0,960
Variabilní symbol	0,513	0,513	0,513
Číslo objednávky	0,444	0,129	0,200
Datum vystavení	0,436	0,436	0,436
Datum splatnosti	0,774	0,889	0,828
Částky	0,718	0,718	0,718
Celková doba zpracování	38,724 s		
Průměrná doba/faktura	0,993 s		

Tabulka 6.3: Úspěšnost vytěžování čísel.

Tabulka 6.3 obsahuje srovnání úspěšnosti vytěžování jednotlivých číselných údajů z testovaných 39 faktur. Do celkové doby zpracování je zahrnuta i doba potřebná k vytěžení nejpravděpodobnějšího dodavatele faktury metodou LSH_1 (knihovna lshhdc).

Přestože samotné získání čárového kódu z textu faktury nepředstavuje výraznější problém (lze ho docílit použitím jednoduchého regulárního výrazu, jak bylo zmíněno v kapitole 6.3.1), úspěšnost určení není stoprocentní. Fakt, že 10 z 12 nalezených čárových kódů bylo správně určeno, vyjadřuje hodnota precision 0,833 v tabulce 6.3. Jelikož dvě testované faktury čárový

kód vůbec neobsahují, hodnota recall je $10/37 = 0,270$. U dvou faktur byla hodnota čárového kódu OCR programem chybně rozpoznána, u zbývajících dvaceti pěti nebyla rozpoznána vůbec. Veškerá chybovost je tedy v tomto případě způsobena již při rozpoznávání textu naskenované faktury.

Přibližně poloviční úspěšnost (0,513 %) určení variabilního symbolu faktury by bylo možné zlepšit např. určením vzdálenosti výrazu „variabilní symbol“ (případně „VS“) od jednotlivých kandidátů na variabilní symbol a následným výběrem toho nejbližšího. Toto by ale neřešilo případ, kdy je uvedeno, že se má jako variabilní číslo použít číslo objednávky.

Obdobným způsobem by bylo možné vylepšit i 20% úspěšnost vytěžování čísel objednávek. Hlavním důvodem 80% neúspěšnosti je však to, že většina faktur obsahuje číslo objednávky v jiném tvaru, než bylo zadavatelem specifikováno.

7 Vlastní SharePoint Add-In

V kapitole 4.3 byly zmíněny možnosti přizpůsobení a rozšíření SharePoint Online tak, aby co nejlépe odpovídal potřebám uživatelů. Jednou z těchto možností rozšíření je vývoj vlastního SharePoint add-inu, na jehož návrh a následnou implementaci se zaměříme v této kapitole.

Cílem je navrhnout a vyvinout add-in, který zajistí vytěžení dat z faktur nahraných do SharePointu. Využijeme proto získané znalosti nejen o rozpoznávání znaků a vytěžování dat z účetních dokumentů, ale i poznatky ohledně cloud computingu, konkrétně na platformě Microsoft Azure.

7.1 Požadavky

7.1.1 Business požadavky

- Zrychlení rutinního procesu práce účetních se zakládáním faktur v papírové podobě.
- Minimalizace chyb vzniklých přepisováním faktur do systému a minimalizace nákladů plynoucích z těchto chyb (pokuty za pozdní zaplacení, ztráta dobrého jména, ztráta obchodního partnera, ...).

7.1.2 Funkční požadavky

- Add-in bude možné aktivovat pro libovolnou knihovnu v SharePoint Online.
- Add-in zajistí rozpoznání textu z obrázku (faktury) vloženého do SharePoint knihovny s aktivovaným add-inem a ze získaného textu následně vytěží data.
- Princip add-inu musí být navržen maximálně jednoduše tak, aby uživatel pracující v systému SharePoint provedl pouze kompletní validaci dat dokumentu a popřípadě doplnění dalších informací bez nutnosti jiných zbytečných kroků, které by zpomalily jeho rutinní činnosti.
- Add-in bude jednoduše rozšiřitelný pro libovolný vytěžovací prostředek.
- Add-in bude primárně napojený na vlastní vytěžovací prostředek.

- Add-in bude obsahovat konfiguraci uloženou v SharePointu – umožní např. výběr požadovaných dat k vytěžení a jejich namapování na zvolené sloupce knihovny.

7.2 Analýza problému

Abychom dokázali navrhnout a implementovat add-in splňující navržené požadavky, musíme být schopni rozlišovat, pro které knihovny je add-in aktivovaný, tj. ve kterých umístěních je vyžadováno, aby byla z vkládaných obrázků vytěžována data. V těchto knihovnách je pak nutné vhodným způsobem identifikovat okamžik vložení nového dokumentu a příslušně na něj zareagovat, tj. v našem případě zahájit rozpoznávání a vytěžování a nově získaná data k faktuře uložit.

Při návrhu add-inu je dále nutné počítat s tím, že se vytěžování dat pro různé zákazníky může (a s vysokou pravděpodobností bude) v některých ohledech lišit. Add-in by měl proto umožňovat určitou míru variability. Někteří zákazníci mohou např. požadovat pouze vytěžování celkových částek faktur, jiní se mohou zaměřit třeba na data splatnosti a v kombinaci s dalšími funkcemi poskytovanými SharePointem tak zajistit kontrolu včasných plateb.

7.2.1 Identifikace událostí

To, že ve sledované knihovně došlo k nějaké události, lze v SharePoint add-inu zaznamenat a zpracovat několika různými způsoby. Dvěma základním přístupům jsou věnovány následující podkapitoly.

Remote event receiver

Remote event receiver (RER) je jednou z možností, jak zpracovávat a reagovat na libovolnou událost, ke které ve sledované knihovně došlo (např. přidání nového souboru, změna vlastností existujícího souboru apod.).

Zdrojový kód RER je svázán s konkrétním webem v SharePointu, ale běží externě na serveru mimo SharePoint. Remote event receiver zaznamená, že v knihovně nastala nějaká událost, a zareaguje zavoláním požadované webové služby. Toto je zpravidla řešeno prostřednictvím služeb Windows Communication Foundation (WCF), ale není nutné se omezovat pouze na služby Microsoft [47].

Události nastávající *před* samotným promítnutím změn do SharePointu probíhají synchronně, díky čemuž je kromě jiného možné provádět validace.

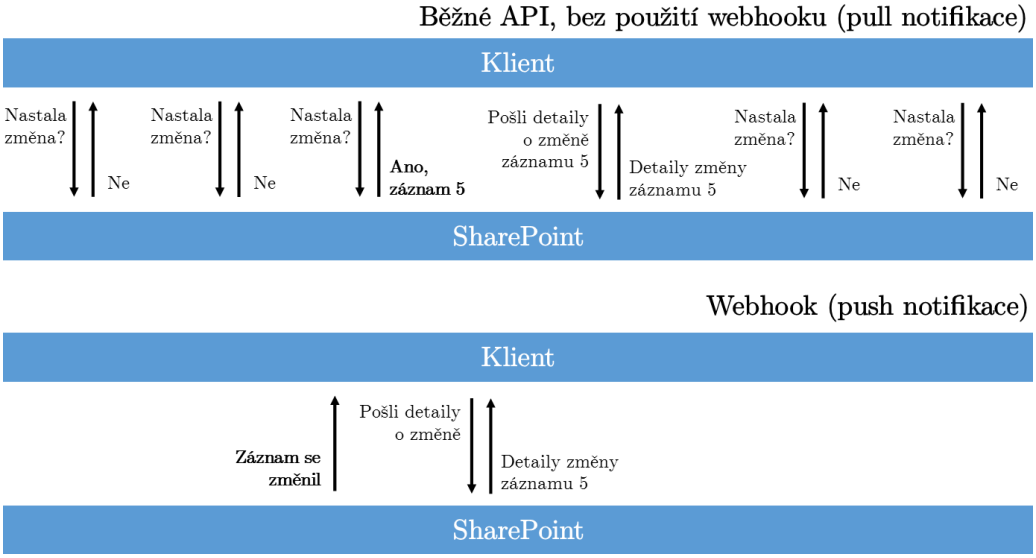
Při případném splnění (resp. nesplnění) určitých podmínek lze zrušit vykonávání celé sekvence příkazů a zamezit tak např. smazání záznamu. Jedná se o události vyjadřující průběh[1], tj. události typu „přidávání záznamu“, „mazání záznamu“ a podobně [45].

Asynchronně naopak probíhají události vyvolané až *poté*, co byla událost zpracována přímo v SharePointu. Mezi ně patří události vyjadřující minulý čas[2], tedy události typu „záznam byl přidán“ nebo „záznam byl smazán“, po nichž již není vyžadována žádná validace [45].

Webhook

Od ledna 2017 je v SharePoint Online k dispozici novější způsob identifikace událostí zpracovávaných asynchronním způsobem. Jde o tzv. *webhook*, tedy jakýsi „háček“ zachycující vznik události. V SharePointu je aktuálně dostupný pouze pro záznamy v knihovnách a seznamech [52].

Webhook není specifickým termínem používaným pouze v SharePointu. Jedná se o obecný pojem vyjadřující způsob obdržení upozornění na to, že nastala nějaká událost (tzv. push notifikace). Tohoto principu využívá mimo jiné i Gmail API, u kterého jsme díky webhooku na přijetí nového e-mailu upozorněni, aniž bychom se museli opakovaně dotazovat (tzv. pull notifikace) [14], [20]. Rozdíl mezi těmito dvěma způsoby notifikací zachycuje schematický obrázek 7.1.



Obrázek 7.1: Srovnání pull a push notifikací (vlastní zpracování dle [16]).

[1] V angličtině události s koncovkou *-ing*, např. ItemAdding, ItemDeleting.
[2] V angličtině události s koncovkou *-ed*, např. ItemAdded, ItemDeleted.

Pro zpracování asynchronních událostí nyní Microsoft doporučuje využívat webhooky místo původních remote event receiverů. Výhodou oproti RER je kromě vyšší míry zabezpečení také opakované odesílání požadavků v případě, že ty předchozí skončily nezdarem (celkem pětkrát po pětiminutových intervalech, tzv. *retry mechanism*). Pokud je volaná služba nedostupná delší dobu, k nápravě dojde při dalším úspěšném zpracování vyslaného požadavku z SharePointu [52], [53].

Práce s webhooky je pro vývojáře zpravidla příjemnější a jednodušší. Webhook reaguje na událost zavoláním libovolného API posláním běžného HTTP požadavku POST, bez nutnosti využívání WCF služeb jako v případě remote event receiverů [52].

7.3 Architektura

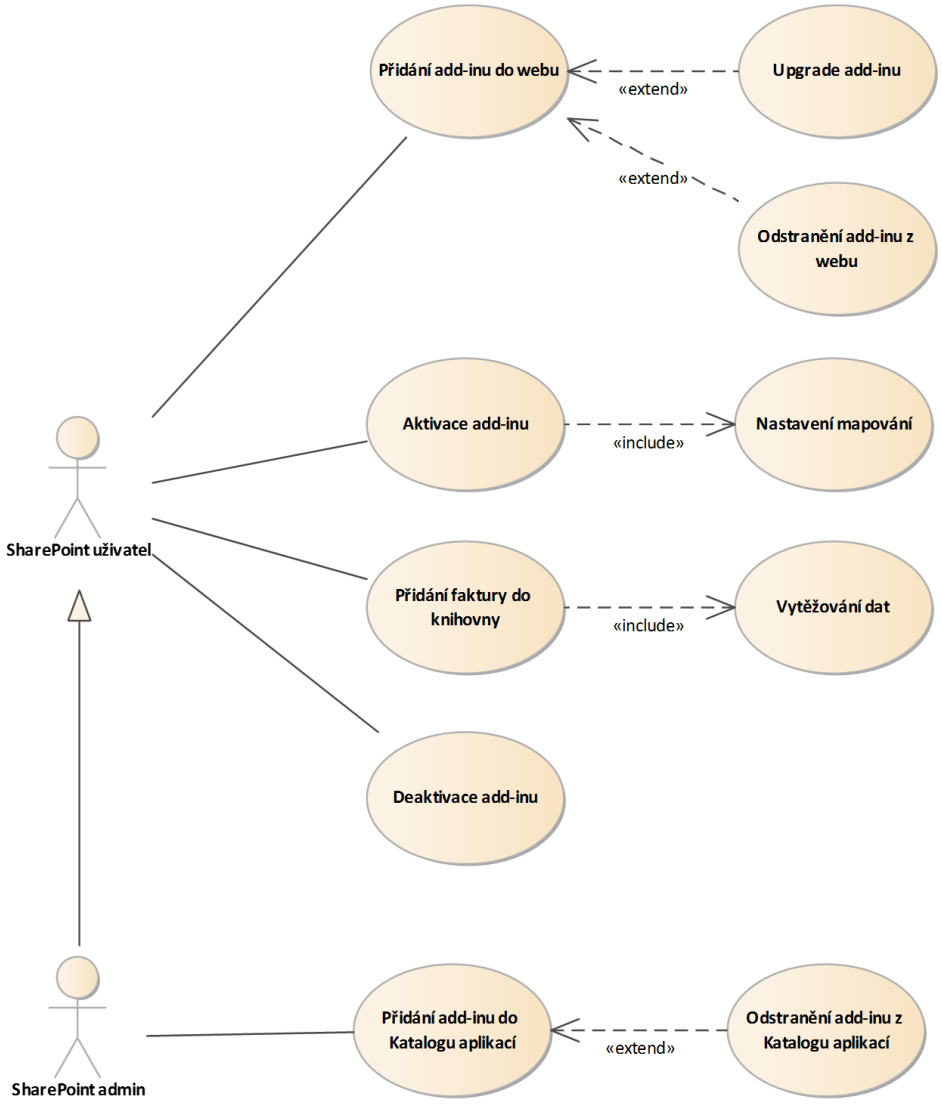
Pro sestavení, nasazení a správu SharePoint add-inu byla zvolena cloudová platforma Microsoft Azure, se kterou již má CCA Group a.s. zkušenosti. Microsoft navíc nabízí možnost vytvoření trial účtu s platností 1 měsíc a předplatným v hodnotě 170 € na vyzkoušení všech služeb, které Azure poskytuje (viz příloha B.1.1). Díky partnerství Microsoftu a CCA Group a.s. jsem navíc měla k dispozici předplatné Microsoft Partner Network, které není časově omezené.

Vzhledem k závěru kapitoly 7.2.1 se jako vhodný způsob identifikace událostí jeví použití webhooku. Při návrhu add-inu i jeho následné implementaci bylo využito vzorové implementace webhooku v kombinaci s SharePoint provider-hosted add-inem, kterou poskytuje společnost Microsoft prostřednictvím GitHubu[3]. Tato implementace využívá několika Azure komponent umožňujících asynchronní zpracovávání událostí (viz dále) [43].

[3] `https://github.com/SharePoint/sp-dev-samples/tree/master/Samples/WebHooks.List`

7.3.1 Diagram případů užití

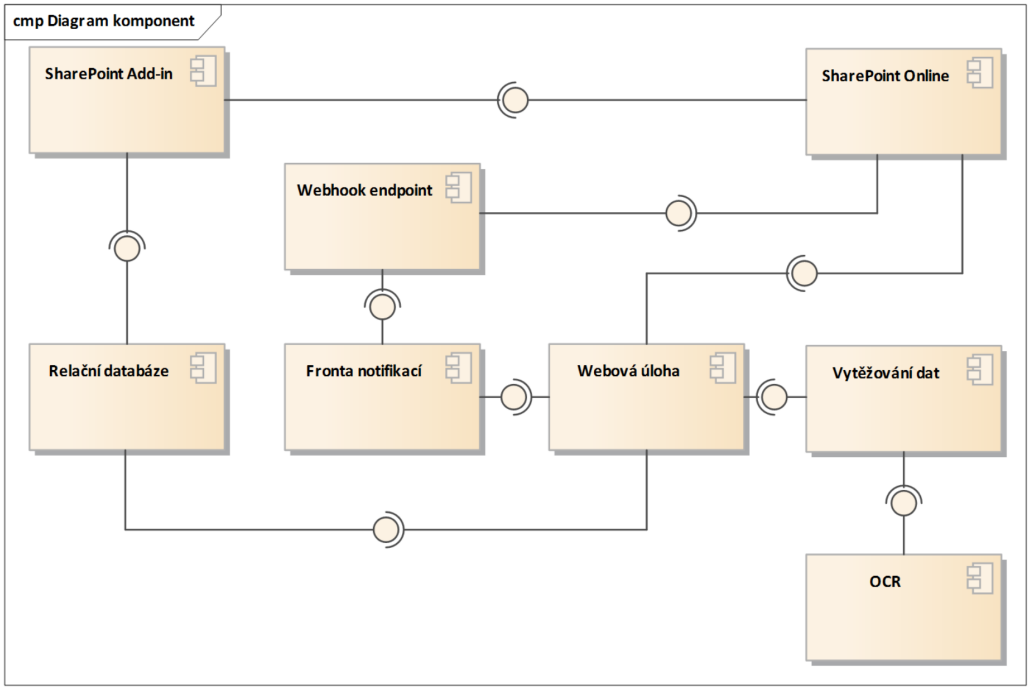
Následující obrázek 7.2 představuje diagram případů užití.



Obrázek 7.2: Diagram případů užití.

7.3.2 Popis komponent

Vyvíjený SharePoint add-in můžeme rozdělit do několika komponent, jejichž vzájemné propojení je zachyceno na následujícím obrázku 7.3. Jednotlivé komponenty mezi sebou komunikují prostřednictvím definovaných rozhraní a jsou v případě potřeby snadno nahraditelné (např. nahrazení služby pro vytěžování dat za jinou).



Obrázek 7.3: Diagram komponent.

SharePoint Online

SharePoint Online je stěžejní komponentou celého modelu. Představuje platformu, pro níž je vyvíjený add-in určen a v rámci které mohou být faktury spravovány.

SharePoint Online komunikuje s okolními komponentami prostřednictvím REST API. Přijímá HTTP požadavky a vrací odpovědi ve formátu JSON nebo XML. Jelikož implementace HTTP požadavků není omezena na konkrétní programovací jazyky, nejsme při práci s SharePoint API v tomto ohledu limitováni. Ukázky volání SharePoint API jsou k dispozici na [40].

SharePoint Add-in

Komponenta představující samotný add-in do SharePointu, zajišťující vytěžování nově přidaných faktur do uživatelem zvolených knihoven/seznamů. Pro persistenci dat využívá relační databázi a s SharePoint Online komunikuje přes SharePoint REST API.

Relační databáze

Do databáze ukládáme data týkající se jednotlivých změn, ke kterým v SharePointu došlo. Dále potřebujeme uchovat uživateli definované mapování

metadat vrácených vytěžovací komponentou na sloupce zvolené knihovny v SharePointu.

Webhook endpoint

Webhook endpoint představuje koncový bod, na který SharePoint Online posílá informace (notifikace) v případě, že došlo k libovolné změně v některé ze sledovaných knihoven.

Jelikož SharePoint očekává potvrzení o přijetí notifikace do 5 sekund, není možné pro každou přijatou notifikaci okamžitě zahájit vytěžování. Webhook se proto v případě nastalé změny v knihovně postará o uložení notifikace do definované fronty z důvodu nutnosti následného asynchronního zpracování.

Fronta notifikací

Fronta notifikací je jednoduché úložiště sloužící k ukládání notifikací zaslaných SharePointem. Notifikace jsou ukládány v textové podobě odpovídající serializovanému JSON objektu (serializaci provádí webhook).

Webová úloha

Obsah fronty notifikací je nepřetržitě monitorován webovou úlohou. V případě, že tato úloha zaznamená ve frontě novou zprávu, zahájí její další zpracování. Toto zpracování spočívá ve zjištění podrobnějších informací o nastalé změně. Pokud je zaznamenanou událostí změna typu „přidání nové faktury“, webová úloha dále zajistí vytěžení dat této nové faktury zavoláním API vytěžovací komponenty a získaná metadata uloží k faktuře do SharePointu.

Vytěžování dat

Komponenta zajišťující vytěžování dat z faktur je stěžejní komponentou celého řešení. S okolím komunikuje prostřednictvím REST API. Očekává soubor s naskenovanou fakturou a vrací vytěžená metadata.

Vzhledem k závěru kapitoly 5 (viz 5.2.4 – Zhodnocení použitelnosti) nepoužijeme pro vytěžování dat z faktur žádné z existujících profesionálních vytěžovacích řešení, ale využijeme poznatků z kapitoly 6 pro vytvoření vlastní jednoduché vytěžovací komponenty.

OCR

Jak již v předchozím textu několikrát zaznělo, s vytěžováním dat naskenovaných faktur souvisí i rozpoznávání znaků. Při využití externí služby pro vytěžování dat z faktur je OCR ve službě již zpravidla zahrnuto a nemusíme se o něj žádným způsobem starat. Vzhledem k rozhodnutí použít vlastní vytěžovací komponentu je však nutné rozpoznávání znaků explicitně zajistit.

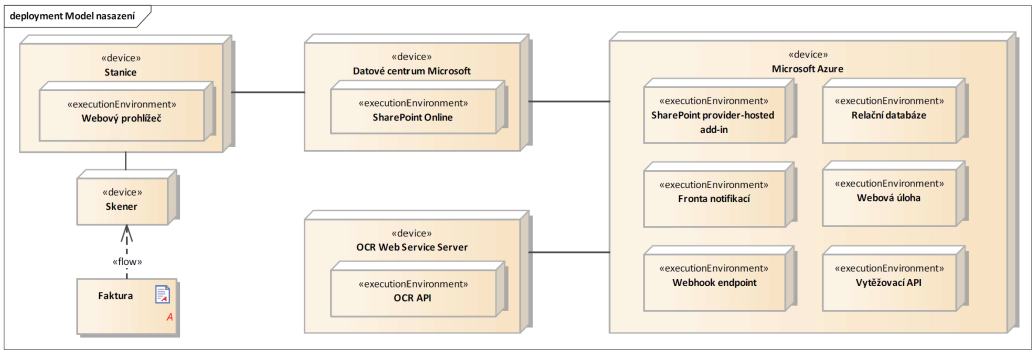
Na základě experimentu provedeného v kapitole 5 jsme pro rozpoznávání znaků zvolili nástroj OnlineOCR.net, který dle tabulky 5.3 vykazuje ze všech vyzkoušených bezplatných řešení nejvyšší přesnost a zároveň poskytuje přístup prostřednictvím API[4]. Na toto API zašle vytěžovací komponenta soubor (naskenovanou fakturu) a v odpovědi obdrží rozpoznaný text, nad kterým následně provede samotné vytěžení dat.

7.3.3 Model nasazení

SharePoint add-in je rozšířením SharePoint Online, běžícího v datových centrech společnosti Microsoft. Uživatelé pro přístup k SharePointu používají webový prohlížeč.

Add-in je vzhledem ke své komplexnosti vyvíjen jako provider-hosted add-in a společně s několika dalšími znázorněnými komponentami je hostován na cloudové platformě Microsoft Azure. Pro svoji funkčnost využívá externí OCR službu.

Kompletní model nasazení je znázorněn na obrázku 7.4.

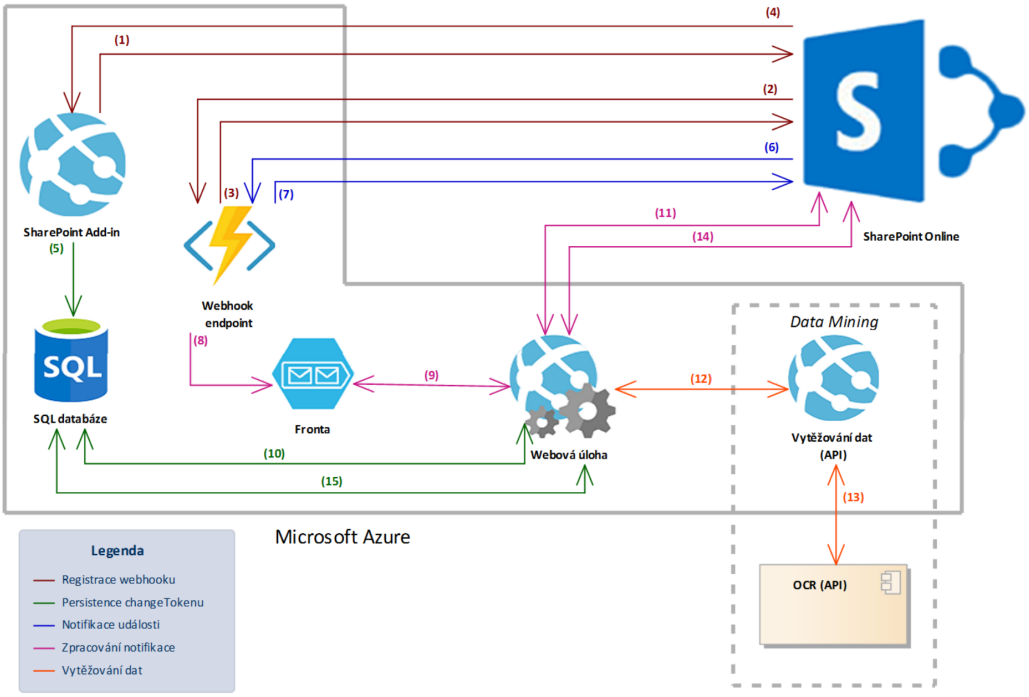


Obrázek 7.4: Model nasazení.

[4] `http://www.ocrwebservice.com`

7.3.4 Interakce komponent

Popis interakce komponent navrženého SharePoint add-inu zachycuje obrázek 7.5. Jednotlivé barvy odpovídají jednotlivým fázím od zaregistrování webhooku až po uložení vytěžených dat z faktury do SharePointu [43].



Obrázek 7.5: Interakce komponent (vlastní zpracování dle [43]).

(1) Vyslání POST požadavku na SharePoint API s žádostí o registraci webhooku ke konkrétní knihovně či seznamu, společně s informací o URL adrese, na které budou přijímány notifikace (tzv. *webhook endpoint*).

(2) Ověření existence *webhook endpointu* odesláním validačního tokenu v parametru POST požadavku na příslušnou URL adresu.

(3) Potvrzení přijetí validačního tokenu. Nutno provést do 5 sekund, jinak celý proces registrace webhooku končí.

(4) Odeslání odpovědi se stavovým kódem *201 Created* obsahujícího ID úspěšně vytvořeného webhooku.

(5) Uložení posledního tzv. *change tokenu*[5] do databáze.

[5]Change token představuje identifikátor jednotlivých změn.

(6) Odeslání POST požadavku na webhook endpoint při identifikaci nové události ve sledované knihovně nebo seznamu. Požadavek obsahuje mimo jiné ID webhooku a ID knihovny (seznamu), ve které událost nastala. Neobsahuje však podrobnější informace o tom, co se změnilo.

(7) Potvrzení přijetí notifikace, opět nutno provést do 5 sekund.

(8) Uložení notifikace do fronty.

(9) Webová úloha pravidelně kontroluje frontu. Pokud v ní zaznamená novou notifikaci (resp. nové notifikace), zahájí její (jejich) zpracování.

(10) Získání posledního change tokenu z databáze. Zpracovávány jsou vždy pouze ty změny, které nastaly po změně odpovídající získanému change tokenu (tj. dosud nezpracované).

(11) Zjištění podrobnějších informací o jednotlivých změnách voláním metody *GetChanges()* poskytované SharePoint API, konkrétně získání nových naskenovaných faktur nahraných do sledované knihovny.

(12) Volání API pro vytěžení informací z faktury.

(13) Vytěžovací aplikace nejprve interně volá OCR API pro rozpoznání znaků faktury, aby mohla následně provést vytěžování.

(14) Uložení výsledků vytěžování k faktuře do příslušné knihovny.

(15) Uložení nového posledního change tokenu do databáze.

Sekvenční diagramy

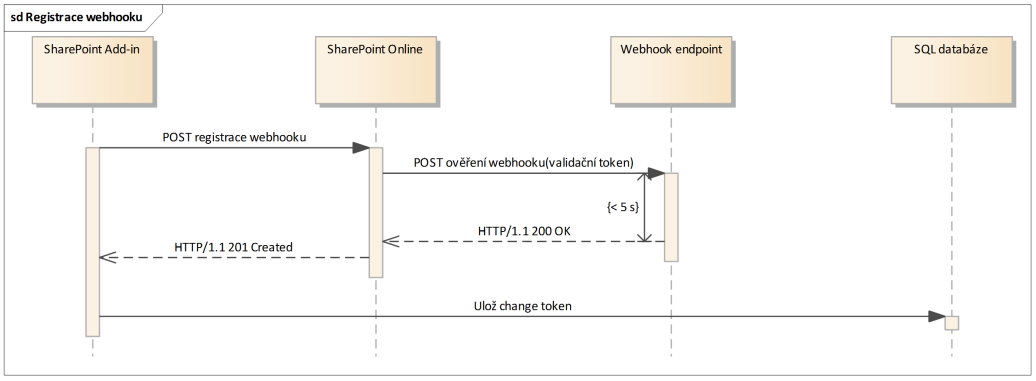
Detailněji zachycuje posloupnost posílání zpráv při registraci webhooku obrázek 7.6.

Pro vytvoření webhooku nad konkrétní knihovnou či seznamem pošle SharePoint add-in následující požadavek (viz příklad 7.1) [43], [54].

```
POST /_api/web/lists('list-id')/subscriptions
Accept: application/json
Content-Type: application/json
Authorization: Bearer + 'access-token'

{
    "resource": "https://ccagroupas173.sharepoint.com/_api/web/lists('list-id')",
    "notificationUrl": "https://invoice-processing-app.azurewebsites.net/api/
        WebHookFunction?code=abcdefgh==",
    "expirationDateTime": "2017-07-01T19:23:18+00:00"
}
```

Příklad 7.1: Žádost o vytvoření webhooku.



Obrázek 7.6: Sekvenční diagram – registrace webhooku.

Atribut *resource* představuje URL adresu knihovny (seznamu), pro kterou chceme webhook vytvořit, adresa v atributu *notificationUrl* odpovídá adrese webhook endpointu, na které budeme přijímat notifikace na nastalé změny.

Platnost webhooku je časově omezená. Výchozí hodnotou je 6 měsíců a tato doba nesmí být překročena hodnotou uvedenou v atributu *expirationDateTime*. Po uplynutí doby platnosti je nutné webhook obnovit [56].

SharePoint v reakci na tuto zprávu pošle na adresu uvedenou v atributu *notificationUrl* požadavek obsahující validační token (viz příklad 7.2).

```
POST https://invoice-processing-app.azurewebsites.net/api/WebHookFunction?code=abcdefgh==

{
    "validationToken": 'random string'
}
```

Příklad 7.2: Odeslání validačního tokenu.

SharePoint do 5 sekund očekává od webhook endpointu odpověď obsahující přijatý validační token, tj. odpověď odpovídající následujícímu příkladu 7.3.

```
HTTP/1.1 200 OK
Content-Type: text/plain

{
    'random string'
}
```

Příklad 7.3: Potvrzení validačního tokenu.

Pokud se webhook podařilo vytvořit, pošle SharePoint add-inu následující odpověď (viz příklad 7.4).

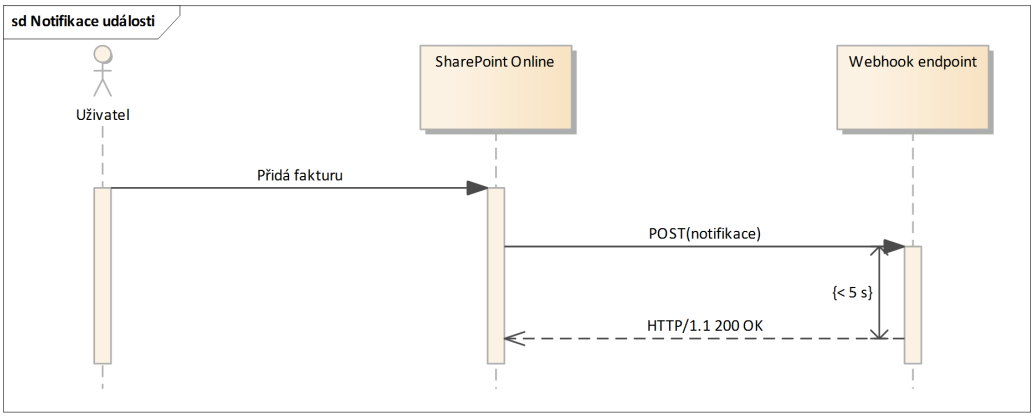
```
HTTP/1.1 201 Created
Content-Type: application/json
```

```
{
    "id": 'subscription-id',
    "expirationDateTime": "2017-07-01T19:23:18Z",
    "notificationUrl": "https://invoice-processing-app.azurewebsites.net/api/
        WebHookFunction?code=abcdefgh==",
    "resource": 'list-id'
}
```

Příklad 7.4: Úspěšné vytvoření webhooku.

Tato odpověď obsahuje identifikátor registrace webhooku (*id*), platnost webhooku (*expirationDateTime*), jeho endpoint (*notificationUrl*) a identifikátor knihovny, pro kterou byl webhook vytvořen (*resource*). Hodnotu atributu *id* také uložíme do databáze (více viz 7.3.5 – Návrh databáze).

V okamžiku, kdy uživatel přidá novou fakturu do knihovny sledované webhookem, pošle SharePoint notifikaci na příslušný webhook endpoint. Tato interakce je zachycena na obrázku 7.7.



Obrázek 7.7: Sekvenční diagram – notifikace události.

Notifikace, kterou SharePoint posílá na webhook endpoint, odpovídá tvaru uvedenému v příkladu 7.5.

```
POST https://invoice-processing-app.azurewebsites.net/api/WebHookFunction?code=abcdefgh==
Accept: application/json
Content-Type: application/json

{
    "subscriptionId": 'subscription-id',
    "clientState": "00000000-0000-0000-0000-000000000000",
    "expirationDateTime": "2017-07-01T19:23:18.0000000Z",
    "resource": 'list-id',
    "tenantId": "00000000-0000-0000-0000-000000000000",
    "siteUrl": "/",
    "webId": 'web-id'
}
```

Příklad 7.5: Notifikace události.

Význam jednotlivých atributů notifikace z příkladu 7.5 je vysvětlen v následující tabulce 7.1 [52].

Název atributu	**Popis**
subscriptionId	identifikátor registrace webhooku
clientState	řetězec použitelný pro různé validace notifikací (volitelný atribut)
expirationDateTime	platnost webhooku
resource	id knihovny/seznamu sledované webhookem
tenantId	identifikátor tenanta
siteUrl	URL adresa webu, na kterém je add-in používán (relativně k adrese serveru)
webId	identifikátor webu, na kterém je add-in používán

Tabulka 7.1: Popis jednotlivých atributů notifikace.

Na přijatou notifikaci je webhook endpoint opět povinen do 5 sekund reagovat odpovědí v následujícím tvaru (viz příklad 7.6).

```
HTTP/1.1 200 OK
```

Příklad 7.6: Potvrzení přijetí notifikace.

Poznámka: SharePoint nevolá webhook endpoint okamžitě poté, co událost nastane, ale ukládá si interně notifikace do fronty (vytváří dávku) a endpoint zavolá jen jednou za minutu.

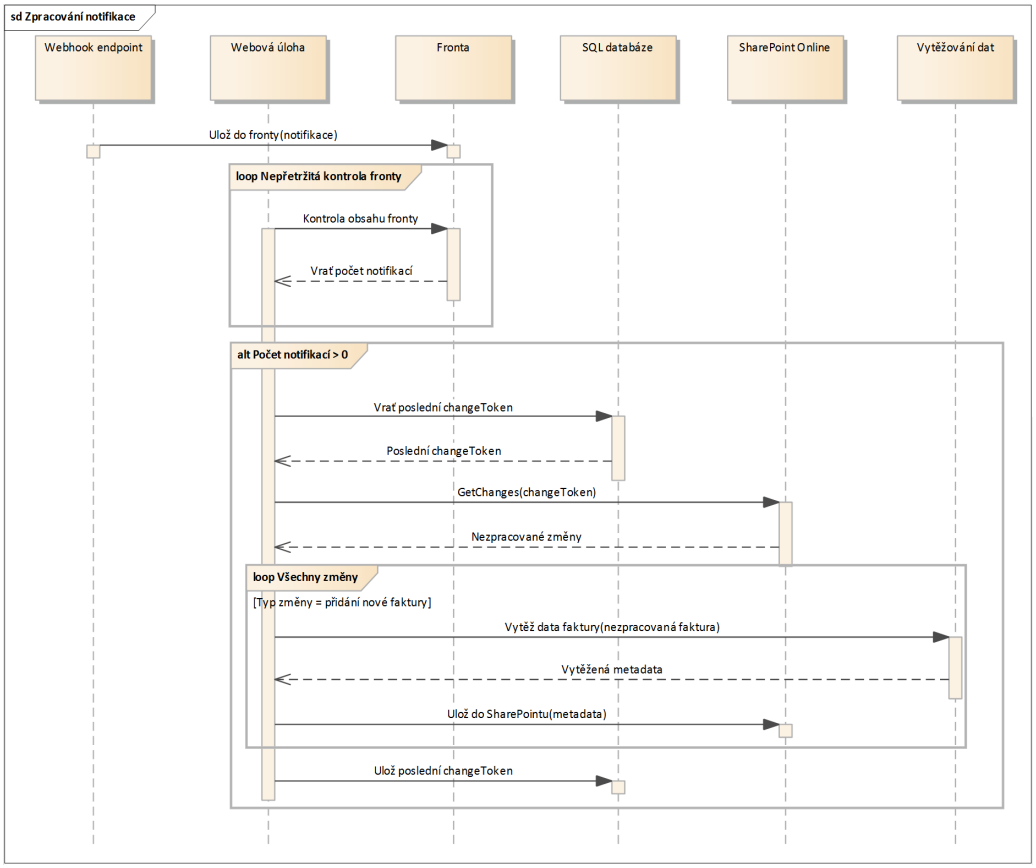
V případě, že do této doby došlo např. ke stovce nových událostí, je webhook endpoint zavolán pouze jednou, ale později při volání metody *GetChanges()* již přistupujeme postupně ke každé ze sta změn [43].

Z tohoto důvodu může nastat situace, kdy SharePoint bude muset poslat více notifikací v jednom požadavku zároveň. V takovém případě představují jednotlivé notifikace prvky pole s názvem *value* (viz příklad 7.7).

```
{
   "value": [
     {
       "subscriptionId": 'subscription-id',
       "clientState": "00000000-0000-0000-0000-000000000000",
       "expirationDateTime": "2017-07-01T19:23:18.0000000Z",
       "resource": 'list-id',
       "tenantId": "00000000-0000-0000-0000-000000000000",
       "siteUrl": "/",
       "webId": 'web-id'
     },
     ...
   ]
}
```

Příklad 7.7: Více notifikací v jednom požadavku zároveň.

Následuje samotné zpracování notifikace, probíhající podle obrázku 7.8. Kromě odeslání potvrzující odpovědi SharePointu reaguje webhook endpoint na přijetí notifikace také jejím přidáním do fronty. Tato fronta je hostovaná v Azure a je dostupná prostřednictvím tzv. *Primary connection string* (více viz Instalační příručka B.1.2 – Účet úložiště). Obsah této fronty je nepřetržitě kontrolován webovou úlohou běžící na pozadí SharePoint add-inu.



Obrázek 7.8: Sekvenční diagram – zpracování notifikace.

V případě, že webová úloha zaznamená ve frontě novou zprávu, následuje dotaz na poslední change token uložený v databázi (tj. určení poslední zpracované změny týkající se daného webhooku a knihovny/seznamu). Všechny změny, které nastaly až po této změně, jsou zatím nezpracované. K získání detailnějších informací o těchto změnách slouží metoda *GetChanges()*, kterou poskytuje SharePoint API.

Jedním z atributů, které tento požadavek vrací, je atribut *ChangeType* definující typ změny, ke které došlo. Jelikož nás zajímají pouze nově přidané faktury, identifikujeme podle tohoto atributu všechny změny typu „přidání nového objektu“ a zbylé ignorujeme.

Pro každou z těchto změn následně zavoláme REST API naší vytěžovací komponenty (viz příklad 7.8). V tomto požadavku předáme soubor, který je s touto konkrétní změnou svázaný (tj. naskenovanou fakturu).

```
POST https://invoice-processing-api.azurewebsites.net
Accept: application/json
Content-Length: 60510
Content-Type: multipart/form-data; boundary=629a7b6e1afe40d7961be95a320b3938

--629a7b6e1afe40d7961be95a320b3938
Content-Disposition: form-data; name="file"; filename="file"


... file content ...


--629a7b6e1afe40d7961be95a320b3938--
```

Příklad 7.8: Požadavek na vytěžení dat z naskenované faktury.

Výsledek vytěžování, který obdržíme jako odpověď na předchozí požadavek, je ve tvaru odpovídajícímu příkladu 7.9.

```
{
    "Barcode":"",
    "Currency":"CZK",
    "DateOfIssue":"1.10.2015",
    "Dic":"CZ25695312",
    "DueDate":"15.10.2015",
    "Ico":"25695312",
    "OrderNumber":"",
    "Status":"Zpracovano",
    "Supplier":"CCA Group, a.s.",
    "TotalAmount":16577.0,
    "VarSymbol":"53012"
}
```

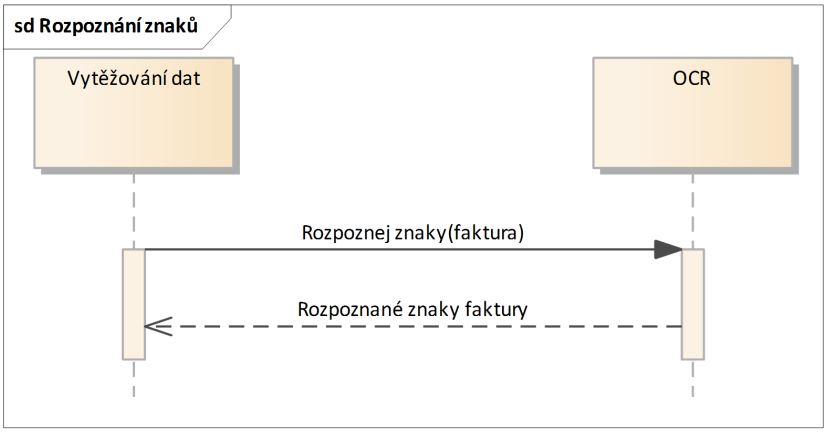
Příklad 7.9: Výsledek vytěžování.

Získaná metadata uložíme do sloupců knihovny podle mapování, které uživatel definoval při aktivaci add-inu (viz tabulka *ResponseMapping* v kapitole 7.3.5). Na závěr ještě uložíme change token poslední zpracované změny do tabulky *ListWebHooks*, abychom při dalším zpracování notifikací věděli, od které změny začít.

Vzhledem k tomu, že jsme se rozhodli použít vlastní jednoduchou implementaci vytěžování dat z faktur, musíme nejprve zajistit také rozpoznávání znaků naskenované faktury. Interakci vytěžovací a rozpoznávací komponenty znázorňuje obrázek 7.9.

REST API zvolené OCR služby[6] přijímá požadavky v následujícím tvaru (viz příklad 7.10).

[6] `http://www.ocrwebservice.com/api/restguide`



Obrázek 7.9: Sekvenční diagram – rozpoznání znaků.

```
POST http://www.ocrwebservice.com/restservices/processDocument?language=czech&gettext=true
Accept: application/json
Content-Length: 60510
Content-Type: multipart/form-data; boundary=629a7b6e1afe40d7961be95a320b3938
Authorization: Basic + 'access-token'

--629a7b6e1afe40d7961be95a320b3938
Content-Disposition: form-data; name="file"; filename="file"

... file content ...

--629a7b6e1afe40d7961be95a320b3938--
```

Příklad 7.10: Požadavek k rozpoznání znaků.

Odpověď přichází zpět ve tvaru zřejmém z příkladu 7.11.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
   "ErrorMessage": "",
   "OutputInformation": null,
   "AvailablePages": 24,
   "ProcessedPages": 1,
   "OCRText": [
      [
         "rozpoznany text"
      ]
   ],
   "OutputFileUrl": "",
   "OutputFileUrl2": "",
   "OutputFileUrl3": "",
   "Reserved": [],
   "OCRWords": [],
   "TaskDescription": null
}
```

Příklad 7.11: Odpověď OCR API.

Pro naše účely je zajímavý zejména atribut *OCRText*, obsahující rozpoznaný text odeslané faktury.

Poznámka: Vzhledem k použití trial účtu služby OCR Web Service je možné rozpoznat pouze omezený počet stránek (25 stran denně). Informaci o počtu zpracovaných stránek (resp. počtu stále dostupných stran) obsahuje atribut *ProcessedPages* (resp. *AvailablePages*). V případě, že je daný počet stran překročen, služba vrací odpověď se stavovým kódem *402 Payment Required*. Atribut *ErrorMessage* obsahuje vysvětlení ve znění „Daily page limit exceeded“.

7.3.5 Návrh databáze

Abychom byli schopni určit, které ze změn, ke kterým v knihovně či seznamu došlo, jsou zatím nezpracované, potřebujeme nějakým způsobem identifikovat poslední zpracovanou změnu. Tato informace je dostupná z vlastnosti *CurrentChangeToken* objektu *List* reprezentujícího sledovanou knihovnu (seznam).

Z důvodu konfigurovatelnosti add-inu musíme do databáze ukládat také mapování získaných metadat faktury na sloupce knihovny zvolené uživatelem při vytváření nového webhooku.

Obrázek 7.10 představuje návrh obou databázových tabulek. Mezi tabulkami není zapotřebí žádná vazba.

Obrázek 7.10: Návrh databázových tabulek.

Tabulka *ListWebHooks*

Tato tabulka se skládá ze tří sloupců. Primární klíč *Id* představuje identifikátor registrace webhooku. Do sloupce *ListId* ukládáme identifikátor knihovny (či seznamu), ve které došlo ke změně. Token reprezentující poslední zpracovanou změnu uchováváme ve sloupci *LastChangeToken*.

Do tabulky *ListWebHooks* ukládáme nový záznam vždy při vytvoření nového webhooku. Po úspěšném zpracování změn příslušný záznam aktualizujeme (uložíme novou hodnotu atributu *LastChangeToken*).

Tabulka *ResponseMapping*

Primárním klíčem tabulky *ResponseMapping* je sloupec *Id* představující jednoznačný identifikátor záznamu. Identifikátor knihovny či seznamu ukládáme stejně jako v případě tabulky *ListWebHooks* do sloupce *ListId*. Typ vytěžené informace patří do atributu *Response* a název sloupce, do kterého si uživatel přeje tuto informaci ukládat, uchováme v atributu *CustomColumn*.

7.4 Popis implementace

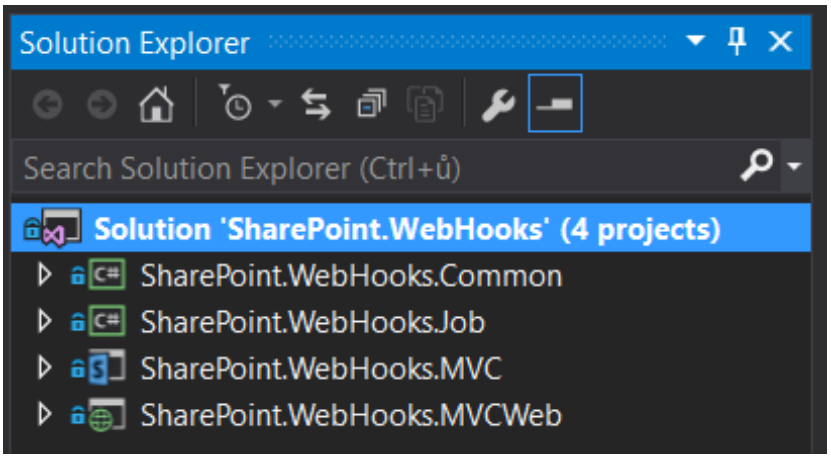
Jak již bylo zmíněno v předchozí kapitole 7.3, při implementaci bylo využito dostupné kostry ukázkové implementace, kterou připravili vývojáři SharePoint ze společnosti Microsoft.

7.4.1 SharePoint add-in

Samotný add-in do SharePoint Online je vyvíjený v nástroji Microsoft Visual Studio v programovacím jazyce C# a nasazuje se jako webová aplikace na platformu Azure. Ostatní vyvíjené komponenty, které add-in pro svoji správnou funkčnost vyžaduje, jsou hostovány na stejné platformě.

Vytvoření a nastavení všech potřebných služeb Azure je popsáno v přílohách v instalační příručce (viz kapitola B.1.2).

Add-in je koncipován jako soubor čtyř projektů (viz obrázek 7.11) tvořících tzv. *solution* (řešení).



Obrázek 7.11: Přehled projektů ve Visual Studiu.

SharePoint.WebHooks.MVCWeb

Tento projekt představuje webovou aplikaci nasazovanou na Azure. Architektura aplikace je založena na architektuře Model-View-Controller (MVC), tj. odděluje od sebe datovou vrstvu, vrstvu business logiky a prezentační vrstvu.

Projekt obsahuje třídu *SharePointSiteModel.cs* definující použitý datový model. Přidávání nových, respektive odebírání stávajících sledovaných knihoven/seznamů a přípravu dat pro vykreslení webové stránky aplikace zajišťuje třída *HomeController.cs*. Samotnou webovou stránku pak představuje soubor *Index.cshtml*. V souborech formátu *.cshtml* jsou standardní HTML a JavaScript doplněny o části kódu v jazyce C# zpracovávané na straně serveru.

V projektu je dále obsažen také konfigurační soubor *Web.config*, ve kterém jsou definovány vztahy k ostatním komponentám v Azure (připojení k databázi, frontě, apod.).

SharePoint.WebHooks.MVC

Důležitým souborem tohoto projektu, který je nasazován přímo do SharePointu, je soubor *AppManifest.xml*. V tomto souboru je uveden mimo jiné název add-inu či označení jeho verze. Definována jsou zde také přístupová práva, jejichž potvrzení bude add-in při instalaci do SharePointu vyžadovat od uživatele (např. právo čtení v kolekci webů).

SharePoint.WebHooks.Job

Projekt *SharePoint.WebHooks.Job* reprezentuje webovou úlohu, která se stará o kontrolu fronty notifikací. O zaznamenání nové notifikace ve frontě se stará metoda *ProcessQueueMessage()* třídy *Functions.cs*.

Součástí projektu je také konfigurační soubor *App.config* umožňující nastavení přístupu k frontě a také přístupových údajů k databázovému serveru a databázi.

SharePoint.WebHooks.Common

V tomto projektu je sdružena společná funkčnost. Třída *WebHookManager.cs* zajišťuje posílání POST požadavků (souvisejících se správou webhooků) na SharePoint API.

Zpracování nových notifikací (zachycených webovou úlohou) zajišťuje třída *ChangeManager.cs*. Voláním metody *GetChanges()* třídy *List* (zprostředkovávající volání metody *GetChanges()* SharePoint API) získáváme po-

drobnější informace o nastalých změnách. Pro každou takovou změnu je následně volána metoda *DoWork()*, která v případě, že se jedná o přidání nového souboru, provede vytěžení dat.

Do tohoto projektu jsou dále zahrnuty např. také SQL skripty pro vytvoření databázových tabulek či správu uživatelů a jejich přístupových práv (viz kapitola B.3).

7.4.2 SQL databáze

Pro persistentní uchování dat vyžadujeme relační databázi. Azure pro tyto účely nabízí SQL databázi a umožňuje její škálování. V nejnižší cenové úrovni Basic, sloužící pro méně častý přístup a méně náročné úlohy, je k dispozici úložiště o velikosti 2 GB. Tato velikost je pro naše potřeby (ukládání knihoven se zaregistrovanými webhooky a mapování metadat) dostačující.

7.4.3 Webhook endpoint

Jednoduchou možností implementace webhook endpointu, který musí SharePointu odpovídat do 5 sekund, je využití tzv. *Azure Function App*. Function App (aplikace funkcí) je služba Azure umožňující vytvoření libovolné funkce bez ohledu na zvolenou platformu či operační systém, a to přímo ve webovém prohlížeči. Při implementaci je na výběr mezi několika jazyky jako např. C#, F#, Python či PHP [32].

Použitý zdrojový kód v jazyce C# je součástí Microsoft dokumentace věnované použití webhooků, viz [33].

7.4.4 Fronta a webová úloha

Důležitou skutečností v komunikaci mezi SharePoint Online a webhook endpointem je to, že notifikace musejí být ze strany webhooku vyhodnoceny do 5 sekund. Aby bylo možné tento časový limit splnit i v případě více notifikací zároveň, je zapotřebí zpracovávat notifikace asynchronně [43].

Azure pro tyto účely poskytuje služby Storage Queue (fronta) a WebJob (webová úloha). Všechny notifikace, které webhook endpoint zaznamená a obratem potvrdí, zároveň také ukládá do fronty, viz příklad 7.12.

```
foreach(var notification in notifications)
{
    CloudStorageAccount storageAccount = CloudStorageAccount.Parse("
        DefaultEndpointsProtocol=https;AccountName=spinvoiceprocessing;
    AccountKey='account-key';EndpointSuffix=core.windows.net");
    // Get queue... create if does not exist.
    CloudQueueClient queueClient = storageAccount.CreateCloudQueueClient();
    CloudQueue queue = queueClient.GetQueueReference("sharepointlistwebhookevent");
```

```
    queue.CreateIfNotExists();

    // add message to the queue
    string message = JsonConvert.SerializeObject(notification);
    queue.AddMessage(new CloudQueueMessage(message));
}
```

Příklad 7.12: Uložení notifikace do fronty.

Nepřetržitě běžící webová úloha pak kontroluje obsah této fronty (viz příklad 7.13) a voláním metody *ProcessNotification()* zajišťuje další zpracování notifikací, tj. vytěžování dat.

```
// This function will get triggered/executed when a new message is written on an Azure
// Queue. This triggering is done due to the QueueTrigger attribute
public static void ProcessQueueMessage([QueueTrigger(ChangeManager.StorageQueueName)]
     NotificationModel notification, TextWriter log)
{
    log.WriteLine(String.Format("Processing subscription {0} for site {1}", notification.
        SubscriptionId, notification.SiteUrl));
    ChangeManager changeManager = new ChangeManager();
    changeManager.ProcessNotification(notification);
}
```

Příklad 7.13: Kontrola obsahu fronty.

7.4.5 Vytěžovací API

Při implementaci vlastní komponenty zajišťující vytěžování dat jsme využili poznatků z kapitoly 5 – Nástroje pro rozpoznávání a vytěžování dat a kapitoly 6 – Návrh vlastního jednoduchého vytěžování. Vytěžovací komponenta je realizována v programovacím jazyce Python verze 2.7, neboť vychází ze zdrojových kódů vytvořených v rámci experimentů popsaných v těchto dvou kapitolách.

Jak již bylo řečeno v kapitole 7.3.2, vytěžovací komponenta se kromě vytěžování dat věnuje také rozpoznávání znaků. Kompletní řešení je nasazeno v Azure jako webová aplikace. Pro vytvoření API ke stávajícímu kódu bylo využito webového frameworku Flask[7].

Framework Flask

Flask je jednoduchý webový framework v Pythonu, založený na knihovnách Werkzeug a Jinja2. Werkzeug představuje knihovnu nástrojů pro práci s WSGI (specifikace rozhraní mezi webovým serverem a webovou aplikací), Jinja2 slouží pro vytváření šablon stránek [72], [74], [67], [73].

[7] `http://flask.pocoo.org/`

Obecné zásady pro práci s Flask frameworkem a správou závislostí v Pythonu jsou shrnuty v Instalační příručce (kapitola B.5). Stažení kostry Flask projektu je popsáno v Instalační příručce v kapitole B.1.2, Webová aplikace – Flask.

Struktura projektu

Výsledná webová aplikace je k dispozici na přiloženém CD a obsahuje tři důležité adresáře – *env*, *FlaskWebProject1* a *wheelhouse*.

Adresář *env* představuje virtuální prostředí projektu. Informace o virtuálním prostředí jsou k dispozici v Instalační příručce (viz kapitola B.5.1).

Ve složce *FlaskWebProject1* jsou umístěny veškeré potřebné zdrojové kódy související s rozpoznáváním znaků a vytěžováním dat z faktur (viz dále).

Předkompilované balíčky (tzv. *wheels*) ve formátu *.whl* jsou umístěny v adresáři *wheelhouse*. Důvodem jejich použití je to, že knihovna používaná pro vytěžování dodavatelů faktur není zahrnuta do databáze Python balíčků PyPI (podrobnosti viz Instalační příručka B.5.3).

Testovací webový server je možné spustit na portu 5555 zadáním příkazu `python runserver.py` do příkazové řádky.

Rozpoznávání znaků

Rozpoznávání znaků faktury přidané do knihovny s aktivovaným add-inem probíhá prostřednictvím volání REST API nástroje OnlineOCR.net (viz příklad 7.10). Jeho použití je podmíněno registrací a následným získáním licenčního klíče. Trial účet lze využívat po dobu jednoho měsíce s omezením na 25 stran denně [68].

Následující příklad 7.14 představuje ukázku kódu volání OCR API. Při tomto volání je potřeba se prokázat uživatelským jménem a přiděleným licenčním klíčem (viz řádka 7).

```
1  import requests
2
3  upload_url = 'http://www.ocrwebservice.com/restservices/processDocument?language=
       czech&gettext=true'
4
5  def ocr(invoice_image):
6      r = requests.post(upload_url, auth=('USERNAME', 'TRIAL_LICENSE_CODE'),
           headers={'Accept': 'application/json'}, files={'file': invoice_image})
7
8      ocr_text = r.json().get("OCRText")[0][0]
9      return ocr_text
```

Příklad 7.14: Volání OCR API (soubor *ocr.py*).

Z vrácené odpovědi nás pro další zpracování (tj. vytěžování dat) zajímá pouze rozpoznaný text, obsažený v atributu *OCRText* (viz řádka 10). Přesný tvar odpovědi je uveden v příkladu 7.11 v závěru kapitoly 7.3.4.

Vytěžování dat

Na základě výsledků experimentu z kapitoly 6 (viz tabulka 6.2) používáme pro určení dodavatele faktury metodu pracující s knihovnou *lshhdc* [8]. Vytěžování dodavatelů i vytěžování číselných údajů faktur jsme pro zjednodušení a pro účely add-inu upravili tak, aby vracelo vždy pouze ten nejpravděpodobnější výsledek, nikoliv kolekci dvojic *výsledek–pravděpodobnost*.

Vytvoření API je díky použití frameworku Flask jednoduché. Na příkladu 7.15 je vidět definice funkce *post_home()*, která se zavolá, pokud přijde na adresu `'/'` požadavek POST.

```
1  @app.route('/', methods=["POST"])
2  def post_home():
3      invoice = ocr.ocr(request.data)
4      mined_numbers = process_invoice(invoice)
5
6      js = json.dumps(mined_numbers, ensure_ascii=False, encoding="utf-8")
7      resp = Response(js, status=200, mimetype='application/json')
8
9      return resp
```

Příklad 7.15: Definice API (soubor *views.py*).

Pomocí `request.data` přistoupíme k přijatému souboru (tj. naskenované faktuře), zajistíme rozpoznání znaků (řádka 3) a následně vytěžíme metadata faktury (řádka 4). Metadata jsou vrácena ve formě slovníku[8], který je funkcí *json.dumps()* převeden na standardní objekt typu JSON (řádka 6). Tento objekt je poté společně se stavovým kódem `200 OK` odeslán jako odpověď (řádka 7).

[8]Datový typ obsahující dvojice *klíč–hodnota*.

8 Zhodnocení výsledků

Podařilo se prokázat proveditelnost SharePoint add-inu pro vytěžování dat z účetních dokumentů. Podle návrhu zpracovaného v předchozí kapitole byl úspěšně vytvořen jednoduchý add-in splňující stanovené požadavky.

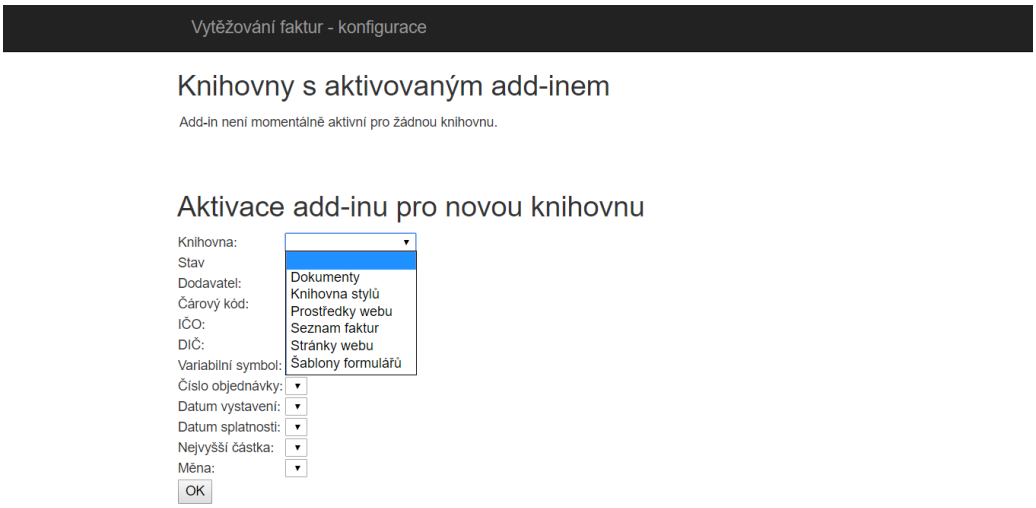
Add-in byl vyvíjen a průběžně ověřován v rámci řešení Faktur CleverDOC zaměřeného na správu a oběh elektronických dokumentů, které poskytuje společnost CCA Group a.s.

8.1 Výsledný add-in

Výsledkem implementace je soubor *SharePoint.WebHooks.MVC.app* představující výsledný add-in, připravený k umístění do Katalogu aplikací SharePoint. Přidat novou aplikaci (add-in) do SharePointu mohou uživatelé s admin přístupovými právy. Bližší informace o přidání nového add-inu jsou obsaženy v Instalační příručce v kapitole B.4.

Jakmile je add-in přidán do Katalogu aplikací, je možné ho začít používat v libovolné kolekci webů. Stručný návod k přidání a použití add-inu je obsažen v Uživatelské příručce (příloha C).

Podobu konfigurační stránky výsledného add-inu zachycuje následující obrázek 8.1.



Obrázek 8.1: Konfigurační stránka add-inu.

8.2 Možnosti rozšíření

Pro finalizaci celého řešení je dále počítáno s budoucím rozšířením na některý z komerčních vytěžovacích nástrojů, využívajících propracovanější vytěžovací mechanismy.

Funkčnost add-inu by bylo vhodné rozšířit o možnost výběru z většího počtu navržených hodnot vytěžovaných metadat, seřazených dle jejich pravděpodobnosti, či možnost propojení vytěžovací komponenty s vlastní databází dodavatelů.

V neposlední řadě by také bylo možné zapracovat konfiguraci jednotlivých vytěžovaných polí tak, aby např. bylo možné zadat požadovaný tvar čísla objednávky, který se má ve fakturách vyhledávat.

9 Závěr

Výsledkem této diplomové práce je použitelný SharePoint add-in, zajišťující vytěžování dat z naskenovaných faktur. Jeho použití přináší částečnou automatizaci přepisování obsahu faktur do systému spojenou s úsporou času a minimalizací chyb vzniklých přepisem.

Návrhu a implementaci add-inu předchází seznámení s problematikou rozpoznávání znaků, vytěžování dat a cloudovou platformou Office 365, potažmo SharePointem.

Bylo provedeno několik experimentů, pro jejichž realizaci byl použit programovací jazyk Python. V rámci prvního experimentu bylo porovnáno několik bezplatných i komerčních nástrojů pro rozpoznávání znaků. Výstupy OnlineOCR.net, dosahujícího pro testované případy nejlepších výsledků (z bezplatných řešení), jsou použity v experimentu, během kterého bylo vyzkoušeno a porovnáno několik metod pro vytěžování dodavatelů a číselných informací z faktur. Jako nejlepší se prokázala metoda založená na Locality-Sensitive Hashing.

Jelikož se nepodařilo zajistit bezplatné vyzkoušení některého z existujících vytěžovacích nástrojů zaměřených na účetní dokumenty, stala se právě tato metoda základem pro implementaci vytěžovací komponenty vyvíjeného SharePoint add-inu.

Návrhu architektury a popisu implementace SharePoint add-inu je věnována celá kapitola 7. Výsledný add-in je hostován na cloudové platformě Microsoft Azure. Kompletní řešení zahrnuje použití několika Azure komponent umožňujících asynchronní zpracovávání faktur přidávaných do SharePointu. Aby bylo možné do řešení integrovat vlastní vytěžovací metodu, bylo nutné k ní vytvořit API. K tomuto účelu bylo využito Python frameworku Flask.

Add-in byl vyvíjen a průběžně ověřován v rámci řešení Faktur CleverDOC. Po finalizaci add-inu spočívající v zapracování možných rozšíření navržených v kapitole 8 by bylo možné tento add-in považovat za plnohodnotné použitelné řešení.

Přehled zkratek

API	Application Programming Interface, rozhraní pro programování aplikací
CTPH	Context Triggered Piecewise Hashing
CRM	Customer Relationship Management, řízení vztahu se zákazníky
DIČ	Daňové identifikační číslo
ERP	Enterprise Resource Planning, podnikový informační systém
HTTP	Hypertext Transfer Protocol, protokol pro výměnu hypertextových dokumentů
IaaS	Infrastructure as a Service, infrastruktura jako služba
IČO	Identifikační číslo osoby
LSH	Locality-Sensitive Hashing
MVC	Model-View-Controller, softwarová architektura
OCR	Optical Character Recognition, optické rozpoznávání znaků
PaaS	Platform as a Service, platforma jako služba
PyPI	Python Package Index, databáze Python balíčků
RER	Remote Event Receiver
REST	Representational State Transfer, datově orientovaná architektura rozhraní
SaaS	Software as a Service, software jako služba
WCF	Windows Communication Foundation, rozhraní pro vytváření servisně orientovaných aplikací
WSGI	Web Server Gateway Interface, specifikace rozhraní mezi webovým serverem a webovou aplikací

Literatura

[1] *The Paperless Project* [online]. 2016. [cit. 2017-05-08]. Dostupné z: http://www.thepaperlessproject.com/.

[2] *Facts About Paper: The Impact of Consumption* [online]. 2016. [cit. 2017-05-08]. Dostupné z: http://www.thepaperlessproject.com/facts-about-paper-the-impact-of-consumption/.

[3] *FuzzyWuzzy: Fuzzy String Matching in Python* [online]. [cit. 2017-05-08]. Dostupné z: http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python.

[4] *Dokumentace knihovny Python-Levenshtein* [online]. [cit. 2017-05-08]. Dostupné z: https://rawgit.com/ztane/python-Levenshtein/master/docs/Levenshtein.html.

[5] *FLANN - Fast Library for Approximate Nearest Neighbors* [online]. [cit. 2017-05-08]. Dostupné z: http://www.cs.ubc.ca/research/flann.

[6] *Python-Levenshtein 0.12.0* [online]. [cit. 2017-05-08]. Dostupné z: https://pypi.python.org/pypi/python-Levenshtein/0.12.0.

[7] *Lshash 0.0.4dev* [online]. [cit. 2017-05-08]. Dostupné z: https://pypi.python.org/pypi/lshash/0.0.4dev.

[8] *Go2starr/lshhdc · GitHub* [online]. [cit. 2017-05-08]. Dostupné z: https://github.com/go2starr/lshhdc.

[9] *Fuzzy Hashing and ssdeep* [online]. [cit. 2017-05-08]. Dostupné z: http://ssdeep.sourceforge.net.

[10] ABBYY. *Invoice Processing and Accounts Payable Solution — ABBYY FlexiCapture for Invoices* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://www.abbyy.com/en-ee/flexicapture-for-invoices/.

[11] ABBYY. *Co je OCR* [online]. 2016. [cit. 2017-05-08]. Dostupné z: http://www.abbyy.cz/products/personal/finereader/about_ocr/whatis_ocr/.

[12] AllianceTek. *How are Companies Leveraging SharePoint?* [online]. 2016. [cit. 2017-05-08]. Dostupné z: http://www.alliancetek.com/Blog/post/2016/07/12/How-are-Companies-Leveraging-SharePoint.aspx.

[13] Amazon. *AWS named as a leader in the Infrastructure as a Service (IaaS) Magic Quadrant report for 6th consecutive year* [online]. [cit. 2017-05-08]. Dostupné z: https://aws.amazon.com/resources/gartner-2016-mq-learn-more/.

[14] Brown, P. *What are Webhooks?* [online]. 2014. [cit. 2017-05-08]. Dostupné z: http://culttt.com/2014/01/22/webhooks.

[15] Bureš, L. *Metody počítačového vidění* [online]. 2015. [cit. 2017-05-08]. Dostupné z: http://www.kky.zcu.cz/uploads/courses/mpv/08/exercise08_materials.pdf.

[16] Chakravorty, I. *Webhooks in SharePoint Online – An Introduction* [online]. 2016. [cit. 2017-05-08]. Dostupné z: http://www.netwoven.com/2016/12/webhooks-in-sharepoint-online-an-introduction/.

[17] Clifton, C. *Data Mining* [online]. 2009. [cit. 2017-05-08]. Dostupné z: http://www.britannica.com/technology/data-mining.

[18] ConceroIT. *How are companies really using SharePoint?* [online]. 2014. [cit. 2017-05-08]. Dostupné z: http://www.itweb.co.za/index.php?option=com_content&view=article&id=136713.

[19] Dostál, D. *V digitálním věku skladují firmy většinu dokumentů stále na papíře. A dělá jim to potíže* [online]. 2016. [cit. 2017-05-08]. Dostupné z: http://www.businessinfo.cz/cs/clanky/v-digitalnim-veku-skladuji-firmy-vetsinu-dokumentu-stale-na-papire-a-dela-jim-to-potize-82632.html.

[20] Google. *Push Notifications* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://developers.google.com/gmail/api/guides/push.

[21] Khosrow-Pour, M. *Encyclopedia of Information Science and Technology, Third Edition*. IGI Global, 2015. ISBN 978-1-4666-5889-9.

[22] Kofax. *Accounts Payable Automation - Applications* [online]. 2017. [cit. 2017-05-08]. Dostupné z: http://www.kofax.com/products/financial-process-automation/accounts-payable-automation/applications.

[23] Kornblum, J. Identifying almost identical files using context triggered piecewise hashing. Digital Investigation, 2006. doi: 10.1016/j.diin.2006.06.015. Dostupné z: http://dfrws.org/2006/proceedings/12-Kornblum.pdf.

[24] Leskovec, J. – Rajaraman, A. – Ullman, J. D. *Mining of Massive Datasets*. Dostupné z: http://infolab.stanford.edu/~ullman/mmds/book.pdf.

[25] ManagementMania. *Cloud computing* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://managementmania.com/cs/cloud-computing.

[26] ManagementMania. *IaaS (Infrastructure as a Service)* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://managementmania.com/cs/infrastructure-as-a-service.

[27] ManagementMania. *PaaS (Platform as a Service)* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://managementmania.com/cs/platform-as-a-service.

[28] ManagementMania. *SaaS (Software as a Service)* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://managementmania.com/cs/software-as-a-service.

[29] Mácha, P. *Historie a základní principy cloud computingu* [online]. 2015. [cit. 2017-05-08]. Dostupné z: https://www.systemonline.cz/clanky/historie-a-zakladni-principy-cloud-computingu.htm.

[30] Microsoft. *SharePoint web hooks reference implementation - Deployment guide* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://github.com/SharePoint/sp-dev-samples/blob/master/Samples/WebHooks.List/Deployment%20guide.md.

[31] Microsoft. *Creating web apps with Flask in Azure* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://docs.microsoft.com/en-us/azure/app-service-web/web-sites-python-create-deploy-flask-app.

[32] Microsoft. *An introduction to Azure Functions* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://docs.microsoft.com/en-us/azure/azure-functions/functions-overview.

[33] Microsoft. *Using Azure Functions with SharePoint webhooks* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://dev.office.com/sharepoint/docs/apis/webhooks/sharepoint-webhooks-using-azure-functions#create-an-azure-function.

[34] Microsoft. *Co je cloud computing?* [online]. 2017. [cit. 8.5.2017]. Dostupné z: https://azure.microsoft.com/cs-cz/overview/what-is-cloud-computing/.

[35] Microsoft. *Co je IaaS?* [online]. . [cit. 2017-05-08]. Dostupné z: https://azure.microsoft.com/cs-cz/overview/what-is-iaas/.

[36] Microsoft. *Co je PaaS?* [online]. . [cit. 2017-05-08]. Dostupné z: https://azure.microsoft.com/cs-cz/overview/what-is-paas/.

[37] Microsoft. *SharePoint Online – software pro spolupráci* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://products.office.com/cs-cz/sharepoint/sharepoint-online-collaboration-software.

[38] Microsoft. *Co je Saas?* [online]. . [cit. 2017-05-08]. Dostupné z: https://azure.microsoft.com/cs-cz/overview/what-is-saas/.

[39] Microsoft. *Bringing Office 365 to new markets* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://blogs.office.com/2016/11/23/bringing-office-365-to-new-markets/.

[40] Microsoft. *Complete basic operations using SharePoint REST endpoints* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://dev.office.com/sharepoint/docs/apis/rest/complete-basic-operations-using-sharepoint-rest-endpoints.

[41] Microsoft. *SharePoint Add-ins* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://msdn.microsoft.com/en-us/library/office/fp179930.aspx.

[42] Microsoft. *Three ways to think about design options for SharePoint Add-ins* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://dev.office.com/sharepoint/docs/sp-add-ins/three-ways-to-think-about-design-options-for-sharepoint-add-ins.

[43] Microsoft. *SharePoint webhooks sample reference implementation* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://dev.office.com/sharepoint/docs/apis/webhooks/webhooks-reference-implementation.

[44] Microsoft. *Plan customizations, solutions, and apps for SharePoint Online* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://support.office.com/en-us/article/Plan-customizations-solutions-and-apps-for-SharePoint-Online-b7898ebf-69b7-4196-81e3-b04e1a4e7d67.

[45] Microsoft. *Handle events in SharePoint Add-ins* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://msdn.microsoft.com/en-us/library/office/jj220048.aspx.

[46] Microsoft. *New name for apps for SharePoint* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://msdn.microsoft.com/en-us/library/office/fp161507.aspx.

[47] Microsoft. *Create a remote event receiver in SharePoint Add-ins* [online]. 2015. [cit. 2017-05-08]. Dostupné z: https://msdn.microsoft.com/en-us/library/office/jj220043.aspx.

[48] Microsoft. *Create or delete a site collection* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://support.office.com/en-us/article/Create-or-delete-a-site-collection-3a3d7ab9-5d21-41f1-b4bd-5200071dd539.

[49] Microsoft. *Sync SharePoint files with the new OneDrive sync client* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://support.office.com/en-us/article/Sync-SharePoint-files-with-the-new-OneDrive-sync-client-6de9ede8-5b6e-4503-80b2-6190f3354a88.

[50] Microsoft. *Create a team site in SharePoint Online* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://support.office.com/en-us/article/Create-a-team-site-in-SharePoint-Online-ef10c1e7-15f3-42a3-98aa-b5972711777d.

[51] Microsoft. *Using web parts on modern pages* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://support.office.com/en-us/article/Using-web-parts-on-modern-pages-336e8e92-3e2d-4298-ae01-d404bbe751e0.

[52] Microsoft. *Overview of SharePoint webhooks* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://dev.office.com/sharepoint/docs/apis/webhooks/overview-sharepoint-webhooks.

[53] Microsoft. *Office Dev PnP Web Cast – Introducing SharePoint WebHooks* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://dev.office.com/blogs/introducing-sharepoint-webhooks.

[54] Microsoft. *Create a new subscription* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://dev.office.com/sharepoint/docs/apis/webhooks/lists/create-subscription.

[55] Microsoft. *What is SharePoint? - Office Support* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://support.office.com/en-us/article/What-is-SharePoint-97b915e6-651b-43b2-827d-fb25777f446f.

[56] MICROSOFT. *How to work with webhook renewal* [online]. 2016. [cit. 2017-05-08]. Dostupné z: https://dev.office.com/sharepoint/docs/apis/webhooks/webhooks-reference-implementation#how-to-work-with-webhook-renewal.

[57] MICROSOFT. *Overview of workflows included with SharePoint* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://support.office.com/en-us/article/Overview-of-workflows-included-with-SharePoint-d74fcceb-3a64-40fb-9904-cc33ca49da56.

[58] MICROSOFT. *Porovnání plánů Office 365 Business* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://products.office.com/cs-cz/business/compare-office-365-for-business-plans.

[59] MICROSOFT. *Compare All Microsoft Office Products* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://products.office.com/en-us/compare-all-microsoft-office-products?tab=2.

[60] MICROSOFT. *Plány a ceny Office 365 Education* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://products.office.com/cs-cz/academic/compare-office-365-education-plans.

[61] MICROSOFT. *Porovnání všech plánů Office 365 pro firmy* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://products.office.com/cs-cz/business/compare-more-office-365-for-business-plans.

[62] MICROSOFT. *Plány a ceny Office 365 Government* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://products.office.com/cs-cz/government/compare-office-365-government-plans.

[63] MICROSOFT. *Porovnání produktů a plánů předplatného Microsoft Office* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://products.office.com/cs-cz/compare-microsoft-office-products.

[64] MÜLLEROVÁ, L. – ŠINDELÁŘ, M. *Účetnictví, daně a audit v obchodních korporacích*. GRADA Publishing, 2016. ISBN 978-80-247-5806-0.

[65] MOTAL, J. *Microsoft Office 365 Launching June 28* [online]. 2011. [cit. 2017-05-08]. Dostupné z: http://www.pcmag.com/article2/0,2817,2386447,00.asp.

[66] NAVISYS. *Microsoft Sharepoint, jednotná platforma pro spolupráci, sdílení dokumentů a řízení pracovních postupů v rámci společnosti, DMS systém – NAVISYS.cz* [online]. 2017. [cit. 2017-05-08]. Dostupné z: https://www.navisys.cz/produkty/sprava-dokumentu-dms/microsoft-sharepoint.

[67] Neto, C. P. *Introduction – WSGI Tutorial* [online]. 2015. [cit. 2017-05-08]. Dostupné z: http://wsgi.tutorial.codepoint.net/.

[68] OCRWebService. *OCR Web Service Pricing* [online]. 2009-2017. [cit. 2017-05-08]. Dostupné z: http://www.ocrwebservice.com/api/pricing.

[69] Panjkov, D. *All About Apps* [online]. 2012. [cit. 2017-05-08]. Dostupné z: http://www.dragan-panjkov.com/blog/archive/2012/12/24/ldquoall-about-appsrdquo-session-recording-and-slides.

[70] Pichaiyappan, R. *14 useful applications of data mining* [online]. 2014. [cit. 2017-05-08]. Dostupné z: http://bigdata-madesimple.com/14-useful-applications-of-data-mining/.

[71] Rak, R. *Kriminalistika: (základy teorie v bezpečnostní praxi)* [online]. 1999. [cit. 2017-05-08]. Dostupné z: http://www.mvcr.cz/clanek/kriminalistika-728588.aspx?q=Y2hudW09Ng%3d%3d.

[72] Ronacher, A. *Flask (A Python Microframework)* [online]. 2010 - 2017. [cit. 2017-05-08]. Dostupné z: http://flask.pocoo.org/.

[73] Ronacher, A. *Jinja2 (The Python Template Engine)* [online]. 2014. [cit. 2017-05-08]. Dostupné z: http://jinja.pocoo.org/.

[74] Ronacher, A. *Werkzeug (The Python WSGI Utility Library)* [online]. 2014. [cit. 2017-05-08]. Dostupné z: http://werkzeug.pocoo.org/.

[75] Saintvilus, R. *Microsoft Launches Office 365 in 10 New Countries (MSFT)* [online]. 2016. [cit. 2017-05-08]. Dostupné z: http://www.investopedia.com/news/microsoft-launches-office-365-10-new-countries-msft/.

[76] SharePointMaven. *SharePoint lists vs libraries* [online]. 2016. [cit. 2017-05-08]. Dostupné z: http://sharepointmaven.com/sharepoint-lists-vs-libraries/.

[77] Signiant. *What is hybrid SaaS?* [online]. 2017. [cit. 2017-05-08]. Dostupné z: http://www.signiant.com/tag/hybrid-saas/.

[78] SocosIT. *DOCU-X OCR — Automatizované vytěžování firemních dokument;* [online]. 2017. [cit. 2017-05-08]. Dostupné z: http://www.socosit.cz/produkty/ocr-automatizovane-vytezovani-firemnich-dokumentu/.

Přílohy

A Faktury

9414

FAKTURA č. 3141553 - Daňový doklad

Dodavatel: [redacted]	Odběratel: [redacted] KCS00000061
Bankovní spojení: [redacted] IBAN: [redacted] BIC: [redacted] Konstantní symbol: 0008 Variabilní symbol: 3141553	Datum vystavení: 31.10.2014 Datum uskuteč. zdan. plnění: 31.10.2014 Datum splatnosti: 30.11.2014 Forma úhrady: převodem
Kontakt: [redacted] Meldung - Nr.: 0 QM-Auftrag: 0 Naše číslo zakázky: 2131996/2	Číslo dílu: 4567719302 Popis problému: Kontrola EKT + dvojtá kontrola zobáčku
Zaúčtování: 311/602	Středisko: 023 Dodávka: S

Název artiklu	Množství	MJ	Cena za MJ	Cena celkem
Pracovní dny	121.5	hod.	235,00 Kč	28 552,50 Kč

[redacted]		Cena celkem bez DPH:	28 552,50 Kč
		Celkem DPH 21 %	5 996,03 Kč
		Celkem k úhradě:	**34 548,53 Kč**

Obrázek A.1: Faktura 1

[redacted]

[redacted]

Rechnung/Invoice/
Facture/Factura

[redacted]

KCS00000040

Beleg-Nr./Document No./
No. du document/Núm. de comprobante
1651917087

Seite
1 von 1

Datum,Abteilung/Date,department/
Date,service/Fecha,departamento
12.11.2014, FAO/DM, CC33

[redacted]

Kunden-Kto.-Nr./Customer Account No./
no. compte client/Núm. cuenta cliente
[redacted]

Unsere Auftragsnummer/Our Order No./
Notre No. de commande/Nuestro núm. de pedido
6550251718

Versandart/Mode of Shipment/
Mode de livraison/Modo de entrega

Lieferdatum/Delivery Date/
Date de livraison/Fecha de entrega

Versand über/Delivery via/
Livraison par/Suministro por

Fahrzeug-Ident-Nr./VIN/
VIN/Núm.ident vehículo

Motor-Nr./Engine No./
No. de moteur/Motor núm.

Unsere LieferscheinNr./Our Delivery No./
Notre No. de bon de livraison/Nuestro núm. de talón de entrega

Ihre Bestellung/Your Order/
Votre commande/Su pedido

Wir berechnen Ihnen:

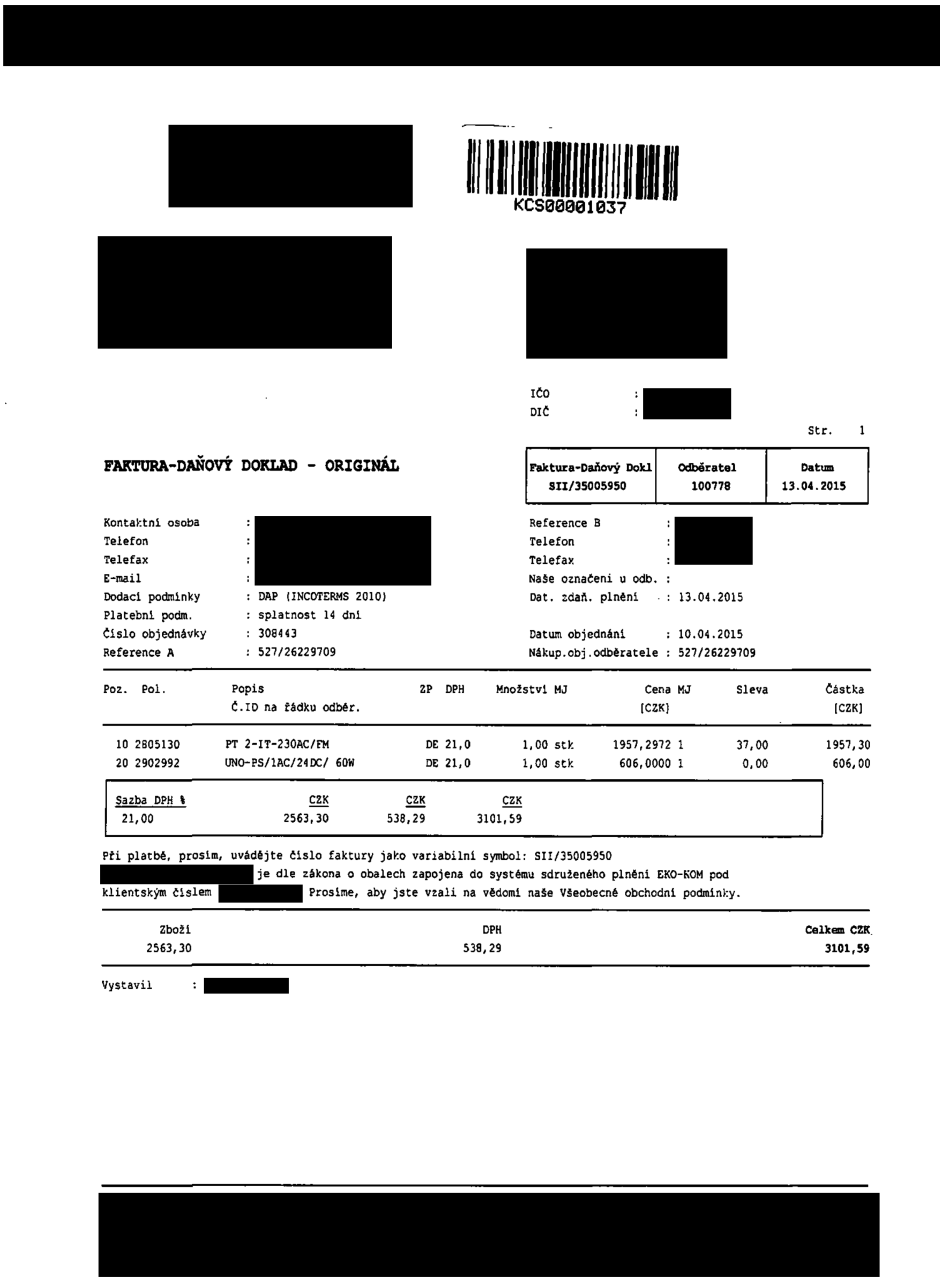
Pos.	Material Bezeichnung	Menge	Preis	Wert/EUR
0010	QEV 111 AATG9C KOSTENBETEILIGUNG SLT (DRITTE) Spezielladungsträgerkostenbeteiligung Nr. 2014/065A/0000049 nach Merkblatt Special Terms Nr. 28 Siehe Anlage Ansprechpartner DAG: Hr. Dünbier (0211 953-2880) Leistungserstellungsdatum: 12.11.2014	1 ST	22.441,00 EUR pro ST	22.441,00
	Summe			22.441,00
	Umsatzsteuer	0,00 %	22.441,00	0,00
	Endbetrag		**EUR**	**22.441,00**

Steuerfreie innergemeinschaftliche Lieferung gem. Art. 138 der Richtlinie 2006/112/EG des Rates.

Zahlungskondition wie vertraglich vereinbart
Bitte geben Sie bei Bezahlung folgende Nummer an: 1651917087

[redacted]

Obrázek A.2: Faktura 2



Obrázek A.3: Faktura 3

39426

Sales Invoice

[redacted]

Title of Goods.
The risk in the goods shall pass to the buyer on delivery but the goods remain the property of [redacted] until payment is made in full. [redacted] reserve the right to remove the goods if payment is not made.

[redacted]

Invoice No.	OP/I406819
Tax Date	31/10/14
Customer Order	[redacted]
Our Reference	176931
Account No.	[redacted]

Qty	Your Part Code	Description	Unit Price	Net Amt	VAT
660	1T4700408604	Assy-Shutface - Hex Free L6428 Shutface Box Qty 12 L600 Pallet Qty 1 L900 Lid Qty 1 Procedure number 86031607 ** Back-order to follow: 176931/1 **	1.55000	1023.00	0.00

[redacted]

KCS00000176

Customer VAT number:

E&OE

All amounts are quoted in GBP unless stated otherwise

TERMS STRICTLY 30 DAYS EOM

Registered Office: [redacted]
Registered No [redacted]

Total Net :	1023.00
Total VAT :	0.00
Grand Total :	1023.00
Currency :	GBP

Obrázek A.4: Faktura 4

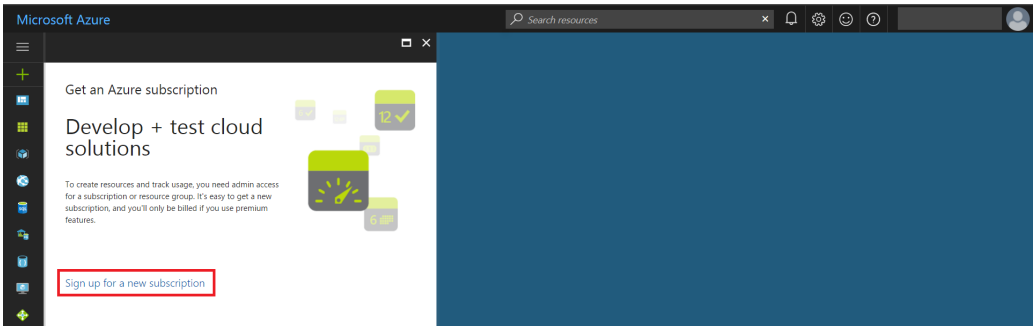
B Instalační příručka

B.1 Microsoft Azure

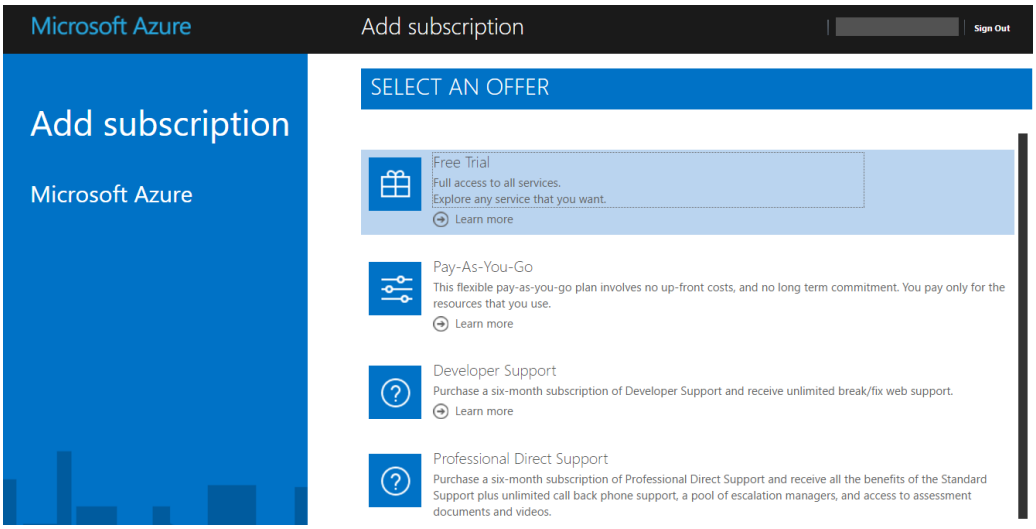
Tato příručka vychází z dokumentace dostupné na [30].

B.1.1 Vytvoření trial předplatného na MS Azure

K vytvoření trial předplatného na platformě Microsoft Azure potřebujete Microsoft účet[1]. Na adrese `http://portal.azure.com` po přihlášení tímto účtem uvidíte následující obrazovku.

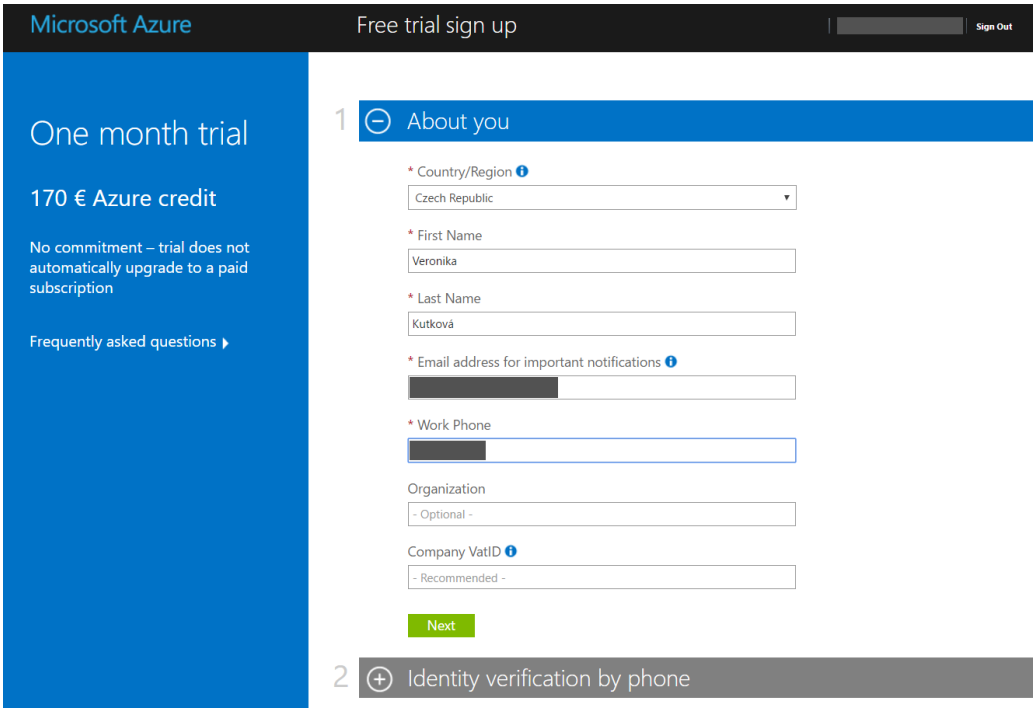


Pro založení nového předplatného klikněte na tlačítko *Sign up for a new subscription* a následně vyberte možnost *Free Trial*.

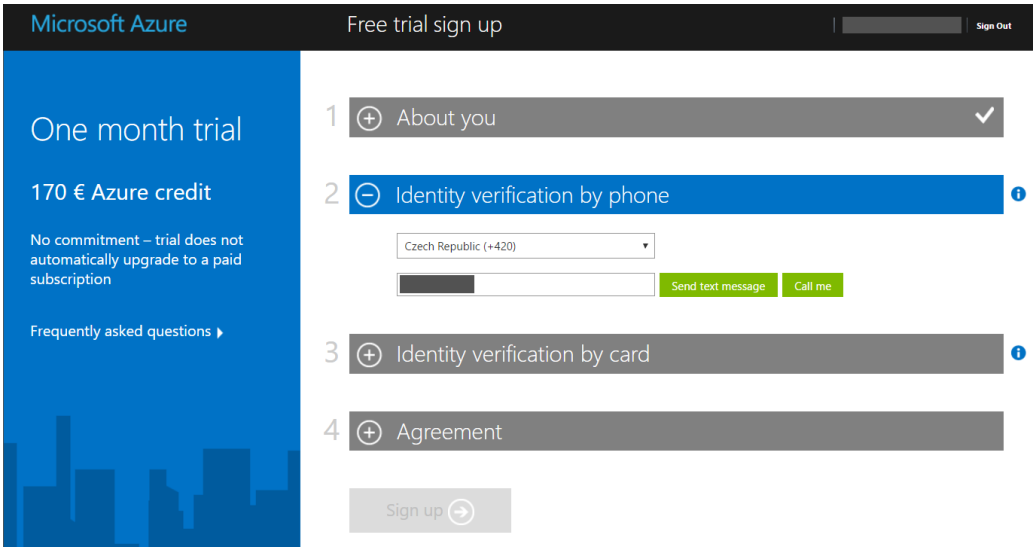


[1] K vytvoření na adrese `https://login.live.com/`.

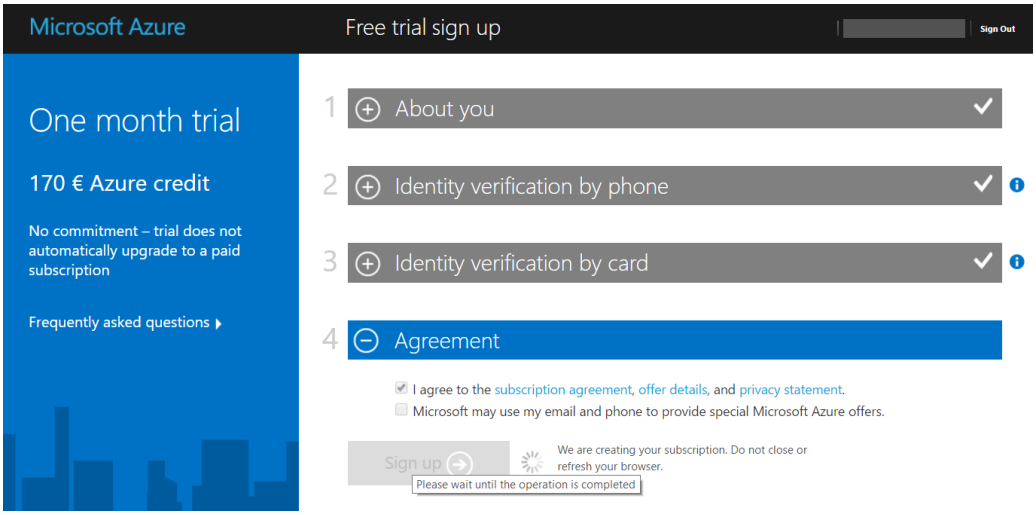
Vyplňte svoje jméno, příjmení, e-mailovou adresu a telefonní číslo a klikněte na *Next*.



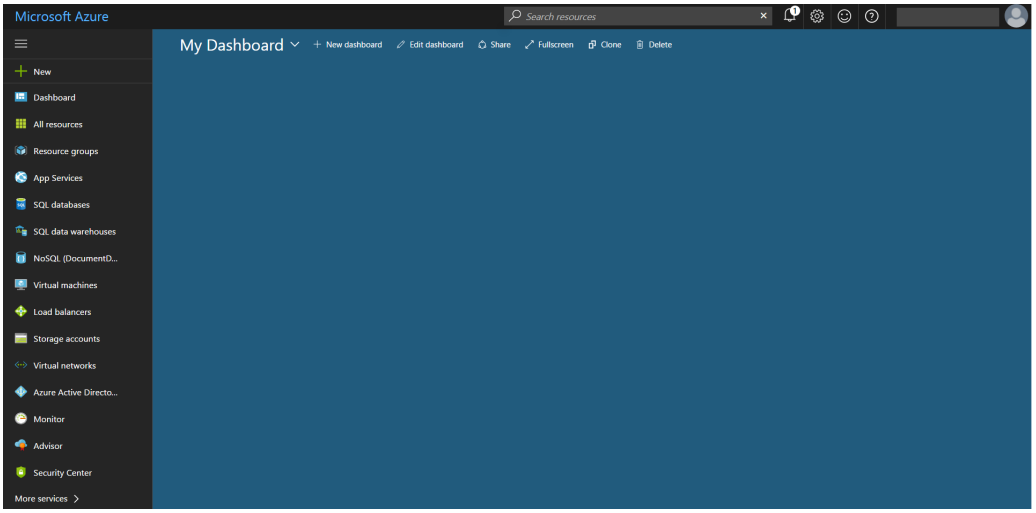
Následuje ověření identity prostřednictvím vašeho telefonního čísla a platební karty.



Pro dokončení klikněte na tlačítko *Sign up*. Je nutné vyjádřit souhlas s podmínkami.



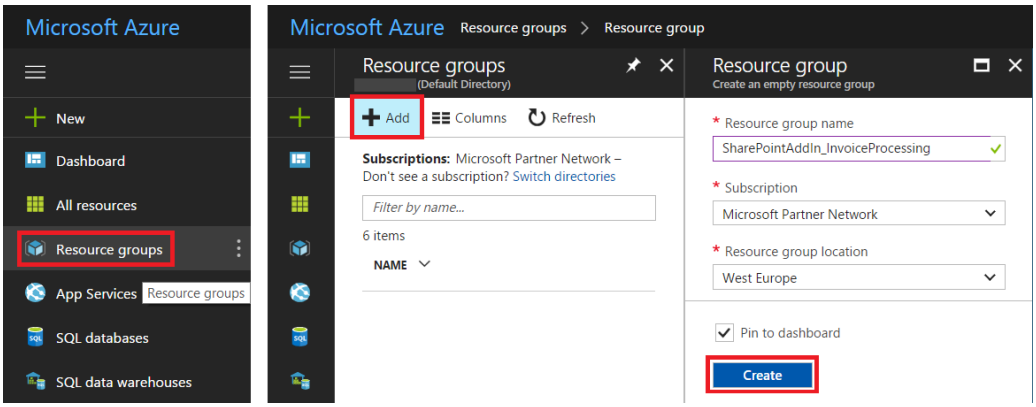
Po opětovném přihlášení do portálu Azure již vidíte svůj dashboard (řídicí panel) – zatím prázdný.



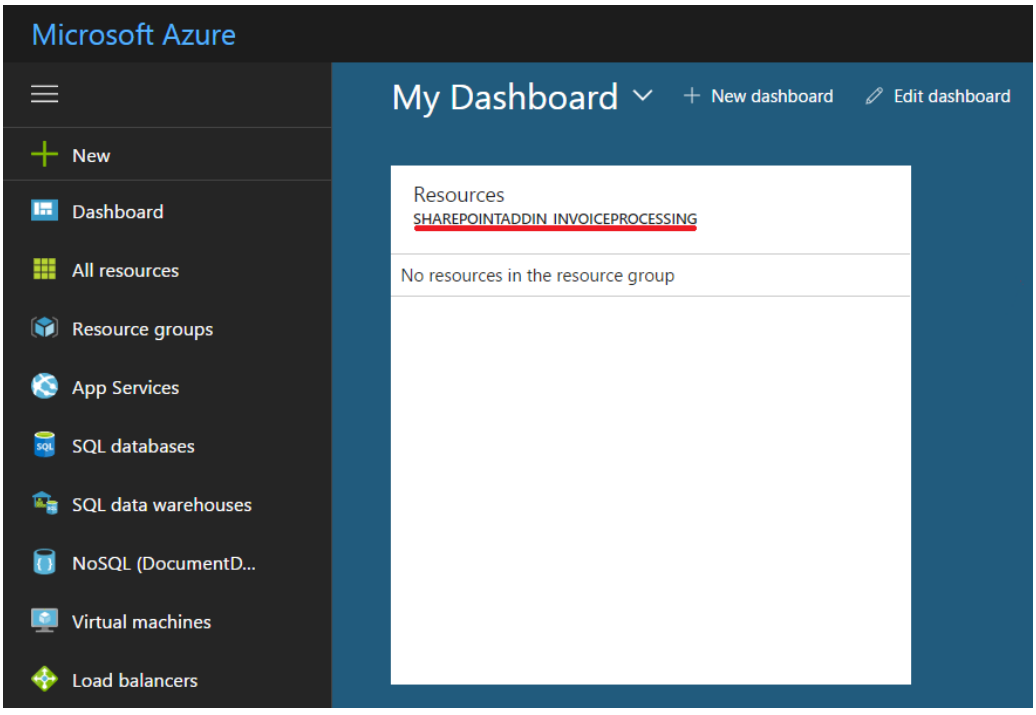
B.1.2 Založení potřebných služeb Azure

Skupiny prostředků

Nejprve si vytvořte vlastní skupinu prostředků pro všechny vaše služby. V levém menu klikněte na *Resource groups* a poté na tlačítko *Add*. Vyplňte název své skupiny prostředků, zvolte typ předplatného Free trial[2], vyberte umístnění a klikněte na tlačítko *Create*.



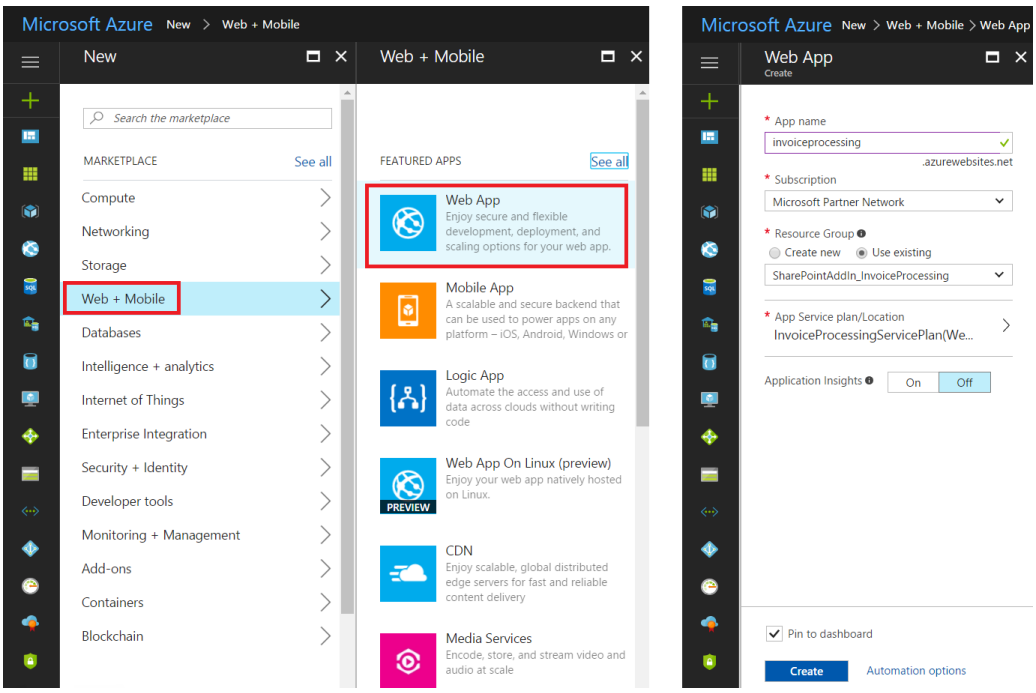
Po přesměrování zpět na dashboard již vidíte svoji vlastní skupinu prostředků, která je však zatím prázdná.



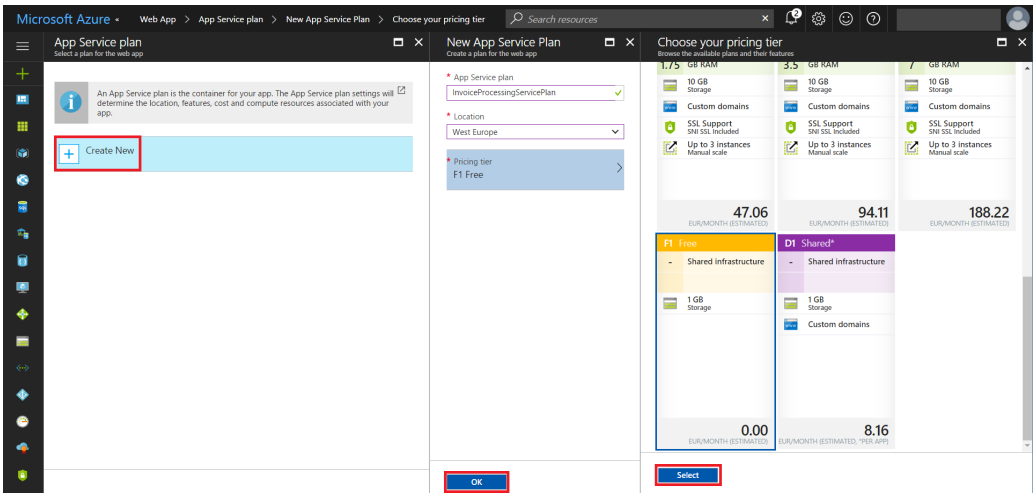
[2]Na obrázku zde je využito předplatné plynoucí z partnerství CCA Group a.s. s Microsoftem.

Webová aplikace – SharePoint add-in

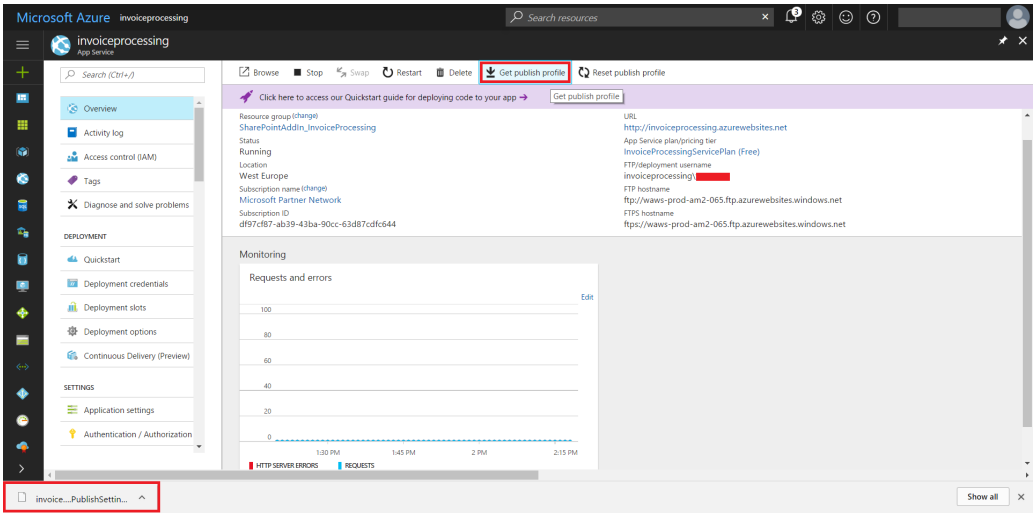
Pro vytvoření webové aplikace představující konfigurační stránku SharePoint add-inu vyberte z levého menu kategorii *Web + Mobile* a následně položku *Web App*. Vyplňte název aplikace, vyberte předplatné a skupinu prostředků z předchozího kroku.



Dále je potřeba vytvořit nový plán pro webovou aplikaci, sloužící jako její kontejner. Klikněte na tlačítko *Create New*, vyplňte název plánu, vyberte lokaci a cenovou úroveň. Pro naši potřebu je dostačující bezplatná úroveň F1 Free. Označte ji, klikněte na *Select* a potvrďte tlačítkem *OK*.

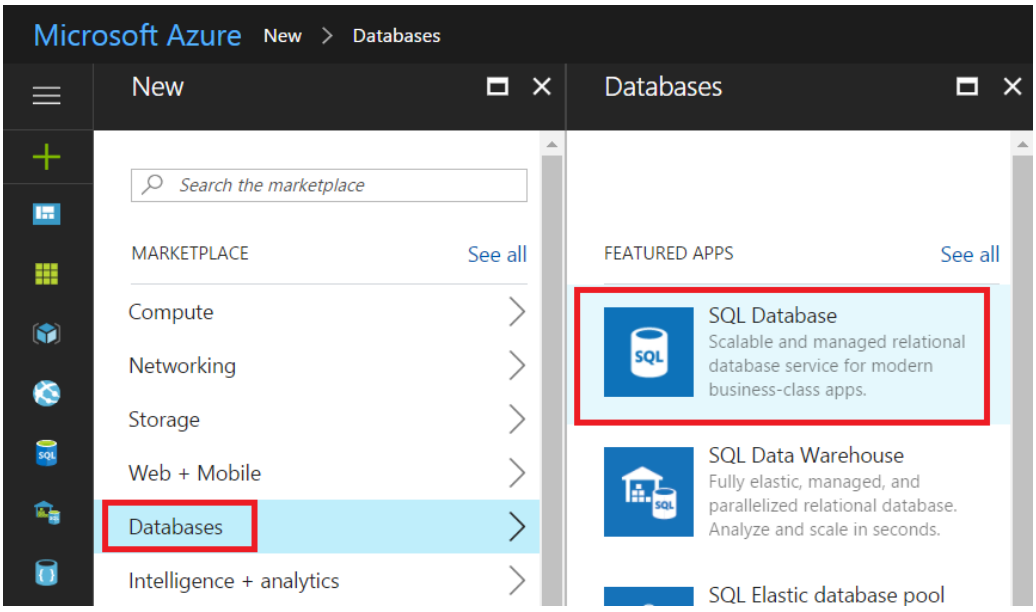


Po vytvoření aplikace a přesměrování na přehled (*Overview*) klikněte v záhlaví stránky na *Get publish profile*. Získáte tak tzv. profil publikování, který budete později potřebovat při nasazování vaší aplikace na Azure (viz B.4).

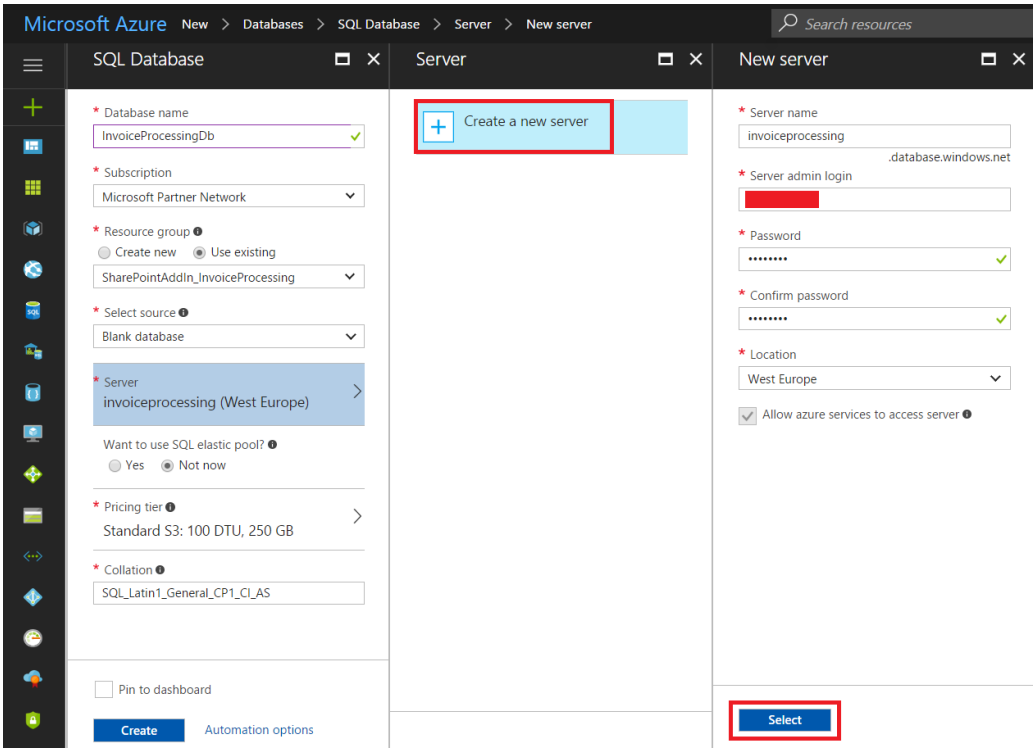


SQL databáze

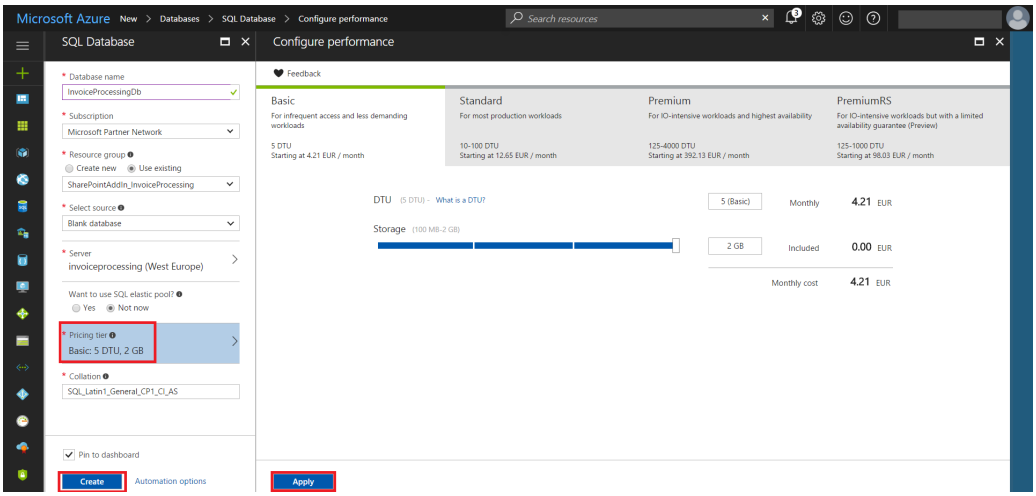
Dále je potřeba vytvořit databázi. V levém menu vyberte kategorii *Databases* a klikněte na položku *SQL Database*.



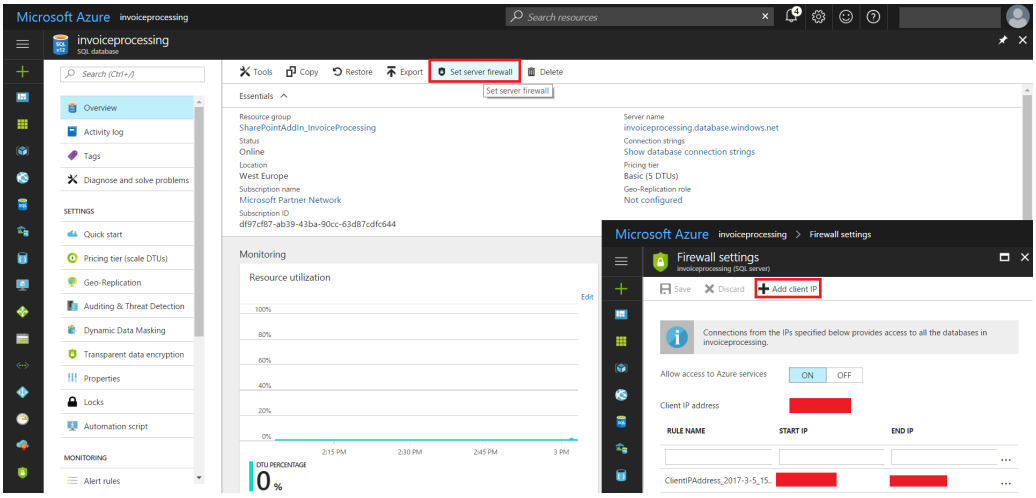
Vyplňte název databáze, zvolte předplatné a existující skupinu prostředků. Jako zdroj vyberte prázdnou databázi (*Blank database*). Kliknutím na tlačítko *Create a new server* zahajte vytvoření nového serveru. Vyplňte jeho jméno, zvolte si přihlašovací jméno a heslo a vyberte umístění. Ukončete stiskem tlačítka *Select*.



Místo nabízené cenové úrovně Standard S3 můžete vybrat levnější úroveň Basic. Potvrďte tlačítkem *Apply* a vytvoření databáze dokončete stiskem tlačítkem *Create*.

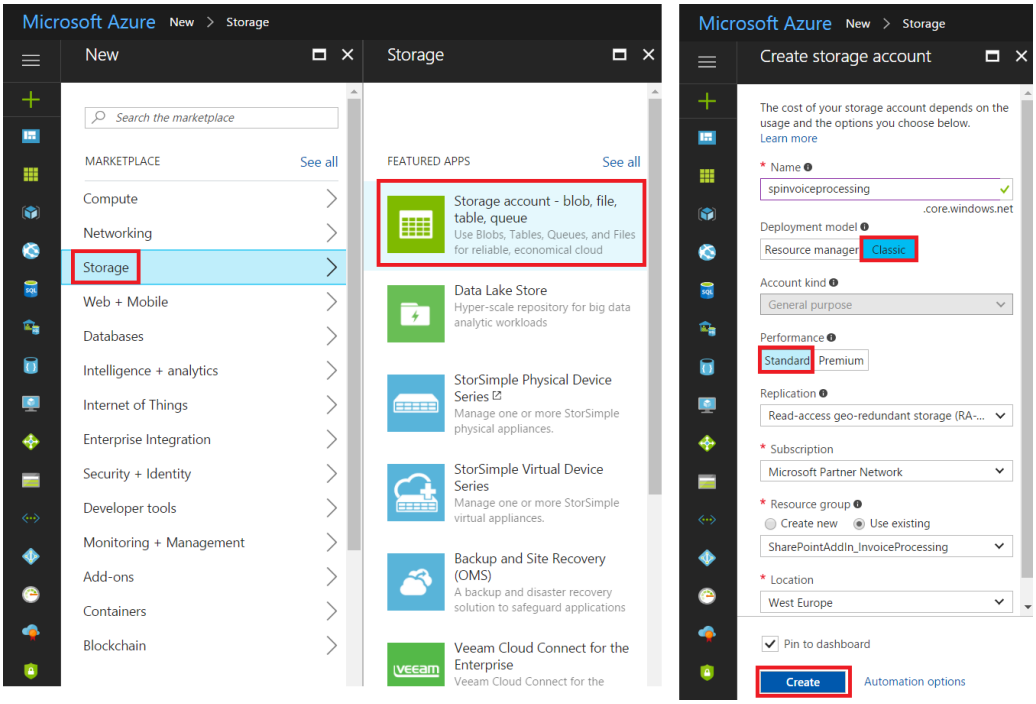


Pro přístup k nově vytvořené databázi je potřeba nastavit bránu firewall serveru. V záhlaví stránky s přehledem informací o databázi klikněte na *Set server firewall* a poté na *Add client IP*. Do seznamu klientských IP adres přibude vaše adresa. Změny uložte kliknutím na tlačítko *Save*.



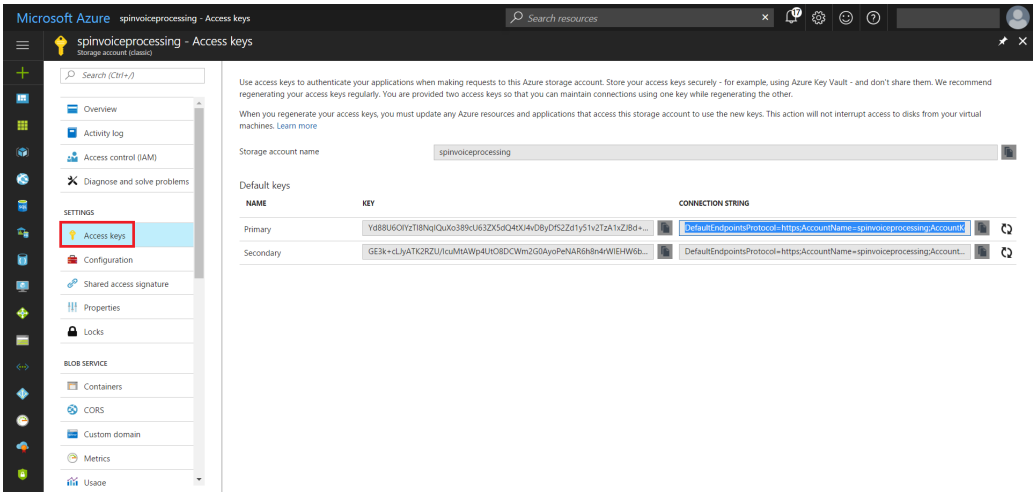
Účet úložiště

Pro ukládání notifikací je potřeba vytvořit frontu (queue). V levém menu vyberte *Storage* a poté položku *Storage account – blob, file, table, queue*.



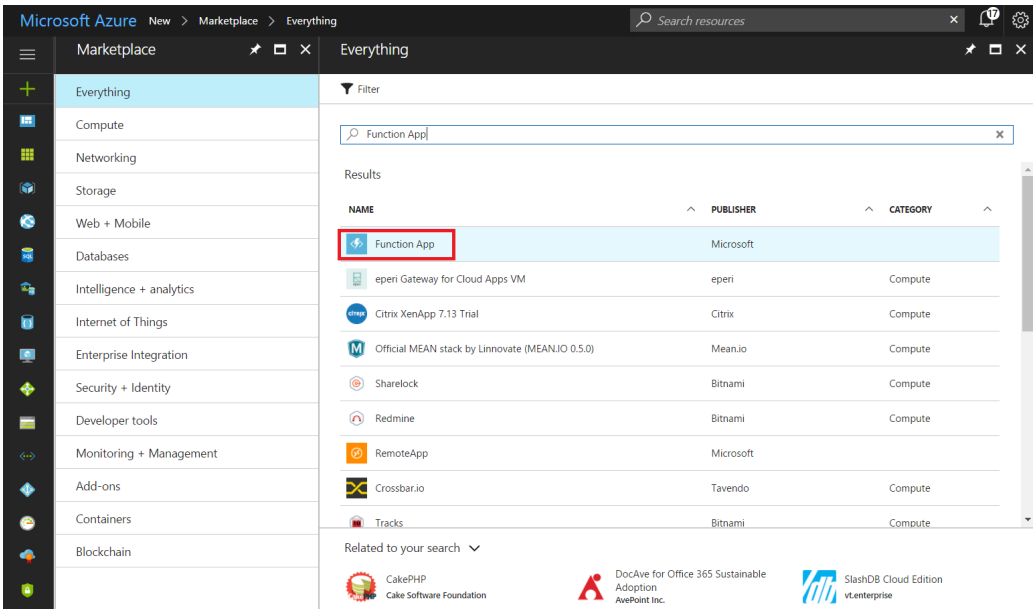
Vyplňte název účtu úložiště, vyberte *Classic Deployment model* (klasický model nasazení) a zkontrolujte, že je vybraná volba *Standard Performance*. Opět vyberte váš typ předplatného, skupinu prostředků a umístění. Dokončete vytvoření stiskem tlačítka *Create*.

Pro nastavení přístupu k frontě z vaší aplikace je nutné znát tzv. *Primary connection string*, ke kterému se dostanete kliknutím na *Access keys*.

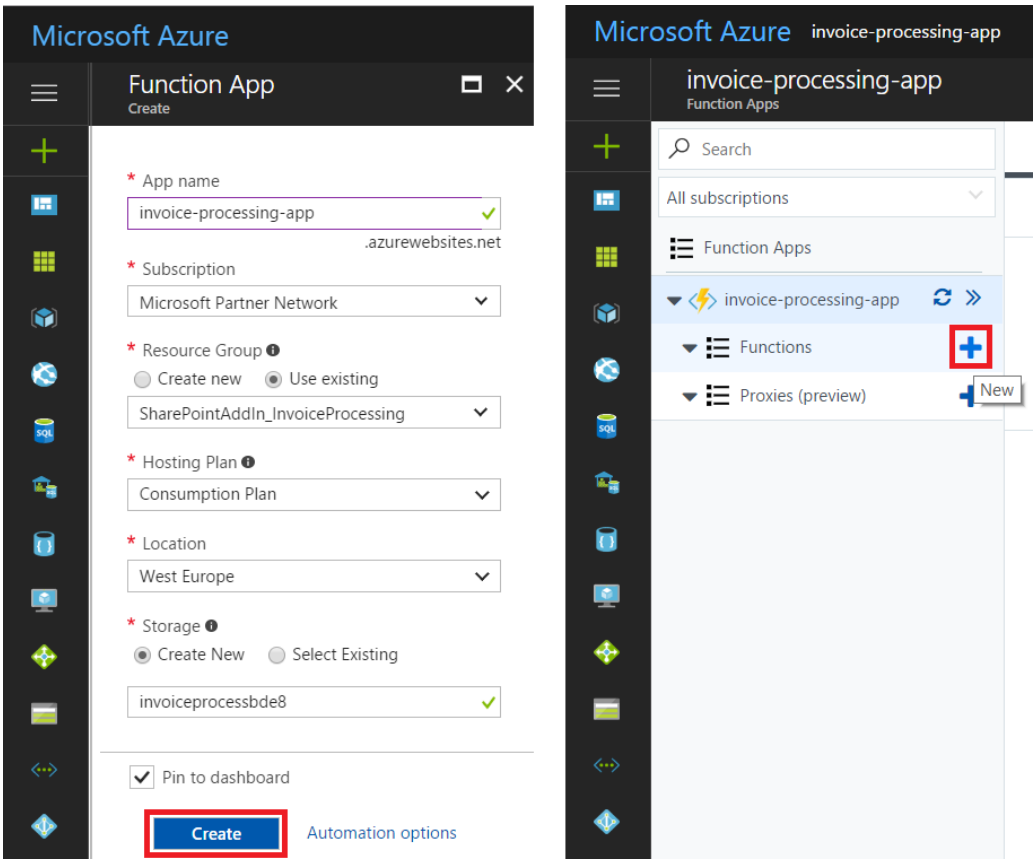


Aplikace funkcí

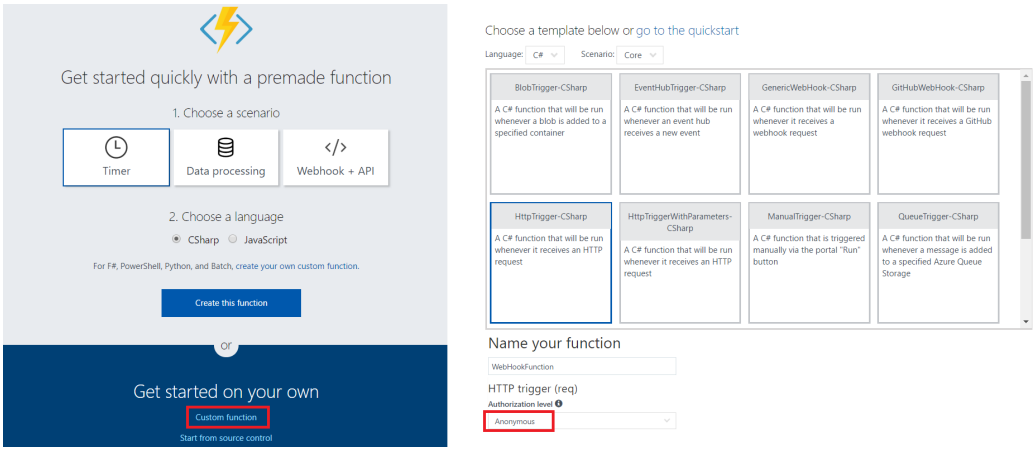
Aplikaci funkcí (Function App) zajišťující implementaci webhook endpointu vytvořte kliknutím na tlačítko *New* a poté do vyhledávacího pole zadejte *Function App* a vyberte první výsledek.



Vyplňte název aplikace a zvolte předplatné a skupinu prostředků. Vyberte plán hostování podle vaší potřeby[3]. Dokončete kliknutím na tlačítko *Create*. Novou funkci vytvořte kliknutím na symbol plus u položky *Functions*.



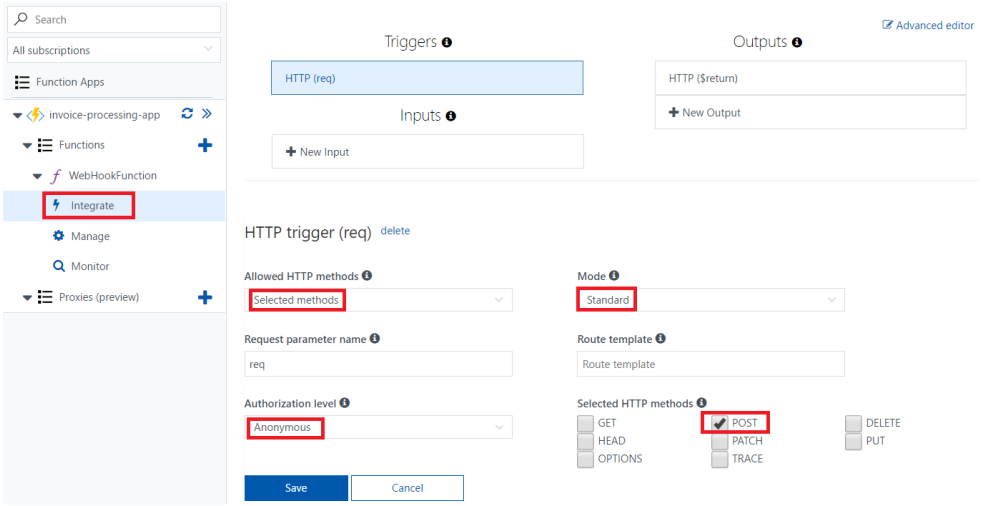
V následujícím okně vyberte možnost *Custom function* (vlastní funkce), a poté volbu *HttpTrigger-CSharp*.



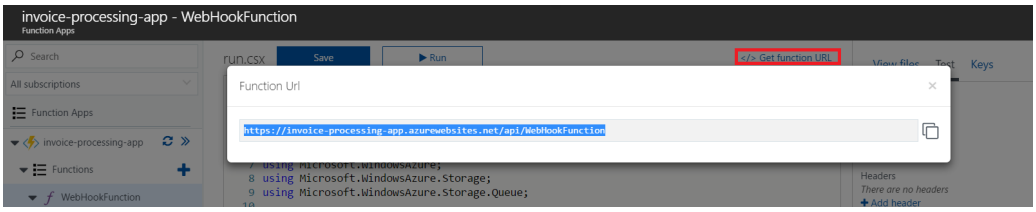
[3]V případě Free Trial předplatného zvolte App Service Plan.

Původní kód nahraďte zdrojovým kódem z [33]. Tento kód se stará o notifikace z SharePointu, reaguje na ně a ukládá je do fronty (viz předchozí podkapitola), jejíž *Primary connection string* je potřeba do kódu doplnit.

Na záložce *Integrate* nově vytvořené aplikace povolte pouze HTTP metodu POST a zkontroluje, že úroveň autorizace (*Authorization level*) je nastavena na *Anonymous* a režim (*Mode*) na hodnotu *Standard*.

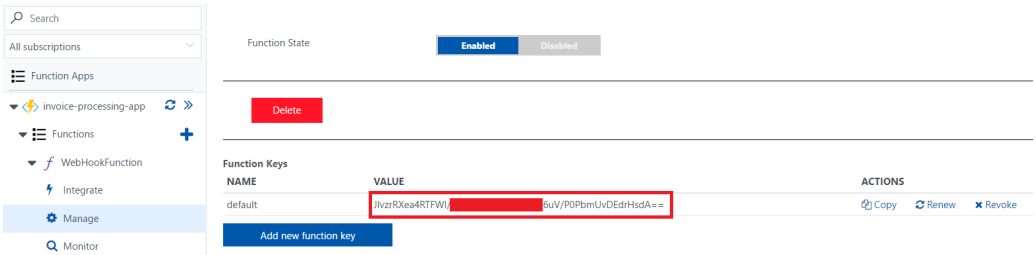


Pro propojení s SharePointem je nutné znát URL adresu vytvořené Function App. Tu získáte kliknutím na *Get function URL*.



Abyste zabránili neoprávněnému posílání požadavků, obstarejte si na záložce *Manage* kód, který připojte jako parametr `code` za získanou adresu.

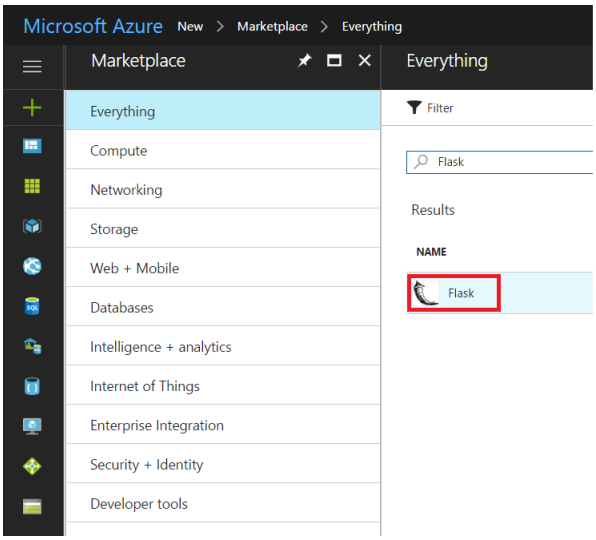
Výsledkem tak může např. být `https://function-app.azurewebsites.net/api/WebHookFunction?code=JIvzrRXea4RTFWI==`.

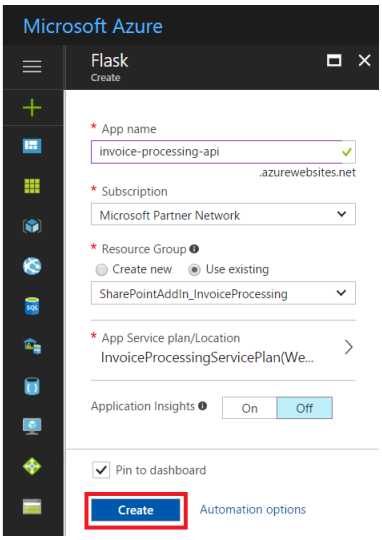


Webová aplikace – Flask

Vytvořte v Azure druhou webovou aplikaci, tentokrát pro vytěžovací komponentu implementovanou pomocí Python frameworku Flask. Do vyhledávacího pole zadejte *Flask* a zvolte první výsledek.

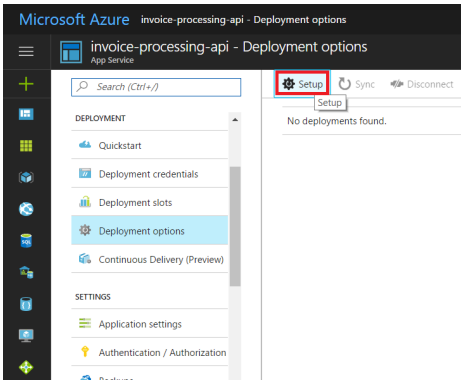
Opět vyplňte název aplikace, zvolte typ předplatného, skupinu prostředků a cenový plán a vytvořte aplikaci kliknutím na tlačítko *Create*.

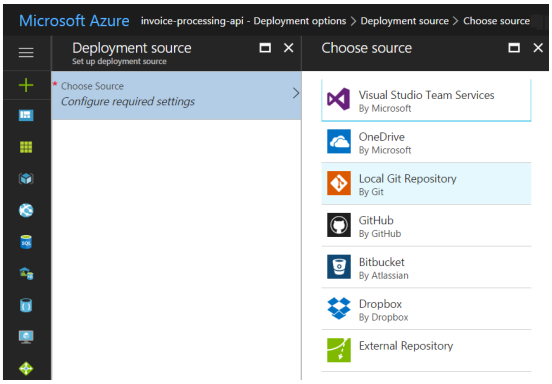




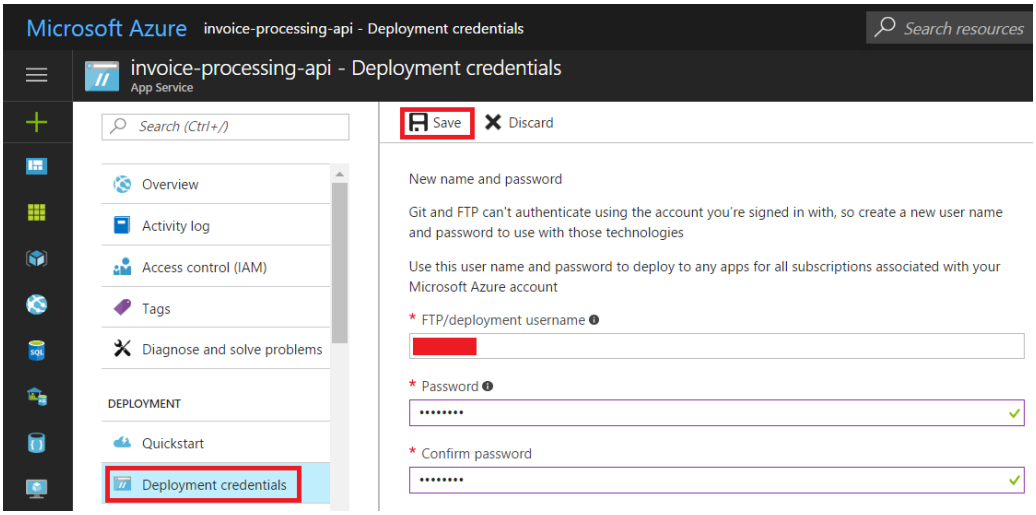
Klikněte na *Deployment options* (možnosti nasazení) v bloku *Deployment* (nasazení). Dostanete se do nastavení, kde si můžete definovat propojení s projektem v lokálním git úložišti a zajistit si tak rychlý způsob nasazení aplikace přímo do cloudu.

Pro nastavení klikněte na tlačítko *Setup* a vyberte možnost *Local Git Repository*.





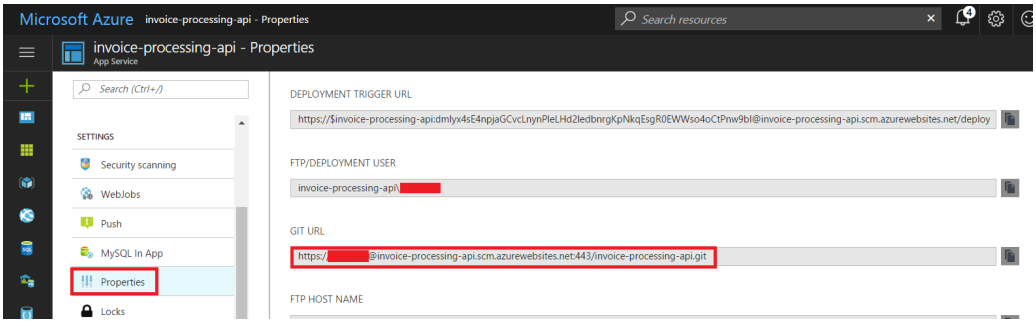
Pokračujte kliknutím na položku *Deployment credentials* (přihlašovací údaje pro nasazení). Zvolte si uživatelské jméno a heslo[4] a údaje uložte kliknutím na tlačítko *Save*.



Ve vlastnostech aplikace (*Properties*) zjistíte GIT URL. Příkazem `git clone GIT_URL` si naklonujte připravenou strukturu Flask projektu k sobě na disk a příkazem `cd <repository-folder>` přejděte do nově vytvořené složky.

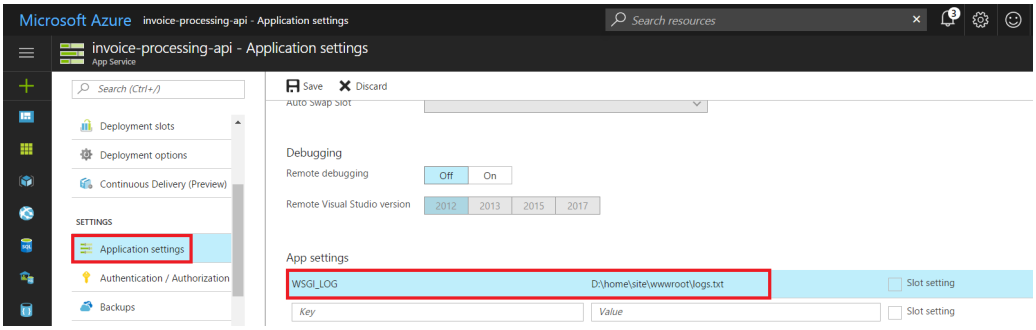
V tomto lokálním repository použijte příkaz `git remote add azure GIT_URL` a přidejte tak novou vzdálenou referenci s názvem *azure*. Jakékoliv lokální změny v aplikaci pak nasadíte na Azure jednoduše použitím příkazu `git push azure master`.

Podrobnější informace najdete na adrese `https://docs.microsoft.com/en-us/azure/app-service-web/app-service-deploy-local-git`.

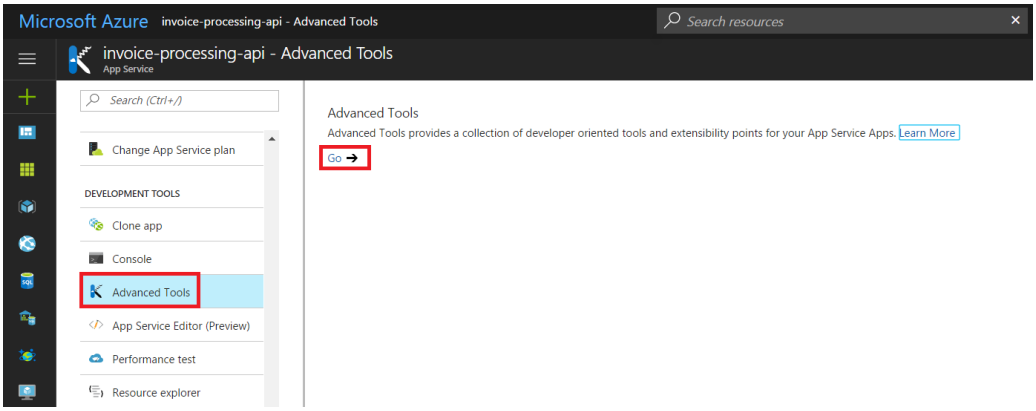


[4]Tyto údaje jsou společné pro všechna předplatná spojená s vaším Microsoft Azure účtem.

Dále je dobré nastavit si logování do souboru umístěného přímo na serveru. V nastavení aplikace (*Application settings*) vyplňte v bloku *App settings* pole *key* hodnotou „WSGI_LOG“ a do *value* doplňte absolutní cestu k textovému souboru (např. D:\home\site\wwwroot\logs.txt).



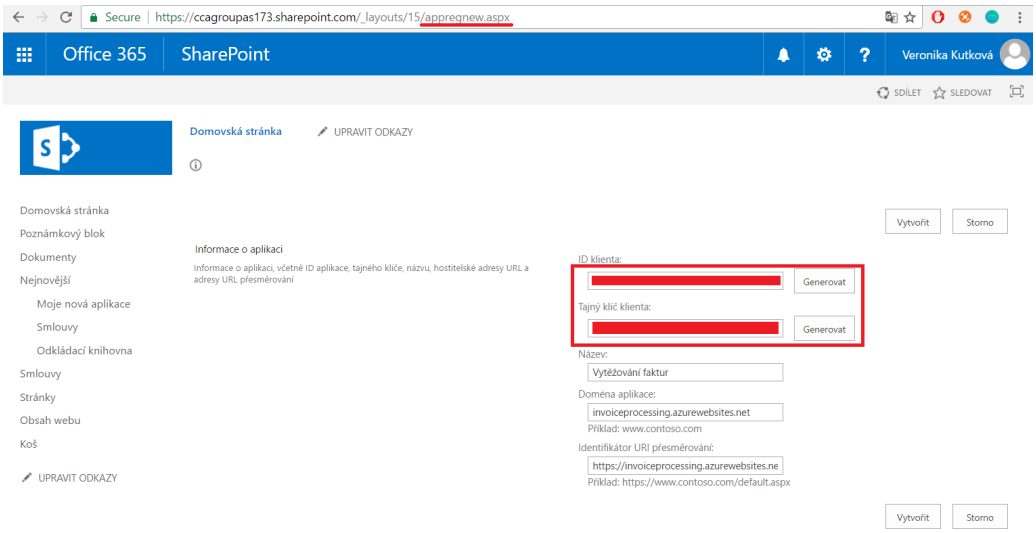
Pro přístup do debug konzole a k dalším zajímavým vývojářským nástrojům můžete využít *Kudu*, dostupné po kliknutí na *Advanced Tools* v bloku *Development Tools*.



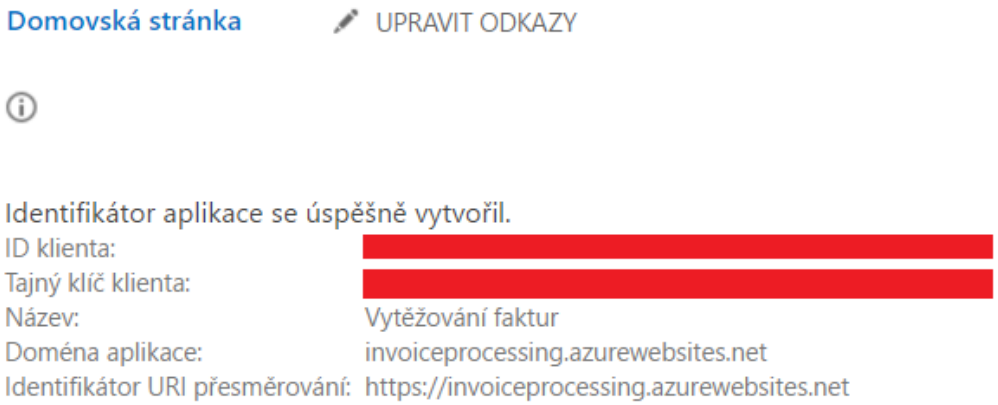
B.2 Registrace add-inu v SharePointu

Abyste mohli vytvořený add-in začít používat v SharePoint Online, musíte ho nejprve řádně zaregistrovat. Registrační stránku naleznete na adrese `http://<tenant>.sharepoint.com/_layouts/15/appregnew.aspx`.

Na této stránce si stiskem tlačítka *Generovat* vygenerujte ID klienta (*Client ID*) a tajný klíč klienta (*Client secret*). Dále si zvolte libovolný název aplikace a vyplňte doménu aplikace (viz B.1.2, Webová aplikace – SharePoint add-in). Jako identifikátor URI přesměrování doplňte stejnou adresu, tentokrát však včetně protokolu HTTPS umožňujícího zabezpečenou komunikaci. Registraci aplikace dokončete stiskem tlačítka *Vytvořit*.



Po úspěšné registraci add-inu budete přesměrováni na stránku obsahující shrnutí vyplněných údajů. Uchovejte si hodnotu ID klienta a tajný klíč klienta pro další použití!

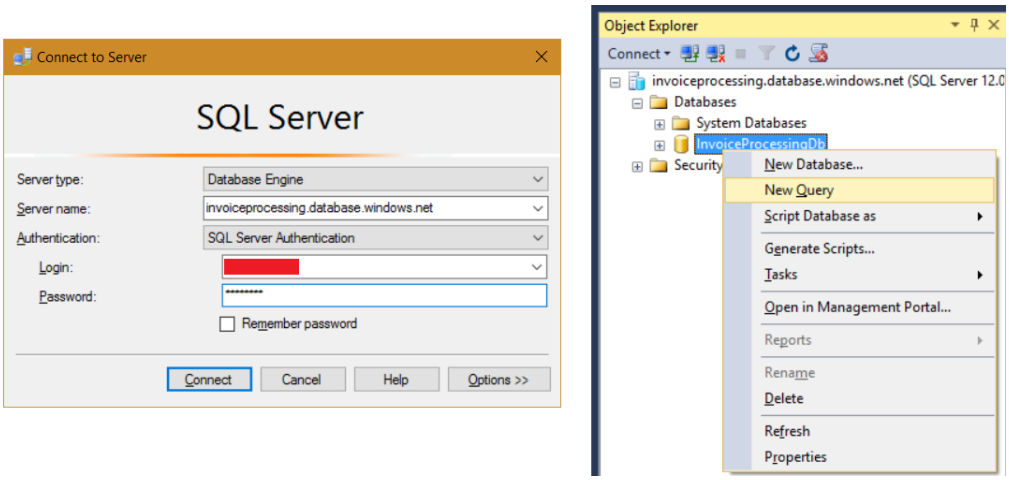


B.3 Nastavení databáze (Microsoft SQL Server Management Studio)

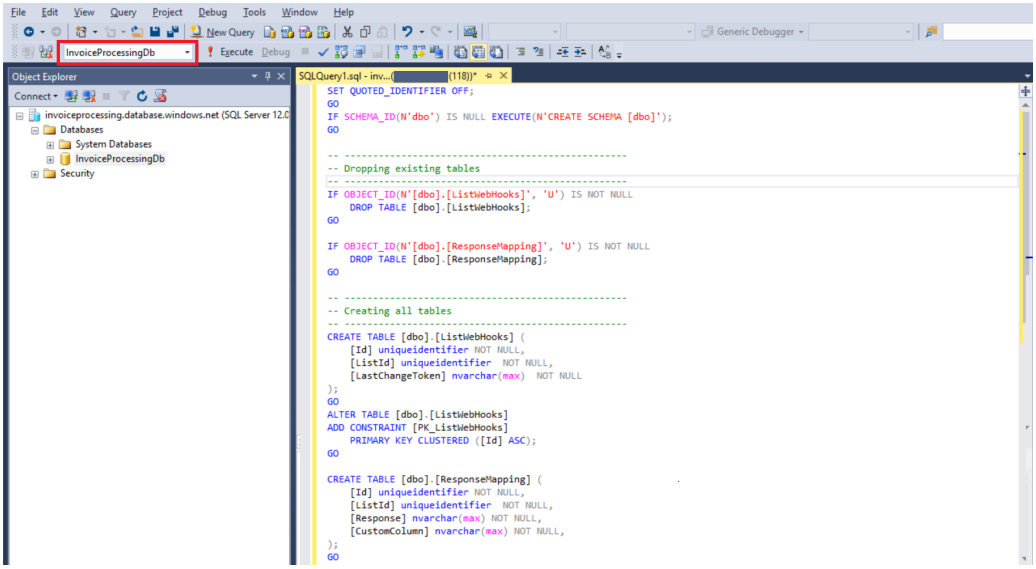
Pro připojení k vytvořenému databázovému serveru použijte např. nástroj Microsoft SQL Server Management Studio.

Po spuštění tohoto programu vyplňte v dialogovém okně název serveru a zadejte přihlašovací údaje (viz B.1.2, SQL databáze).

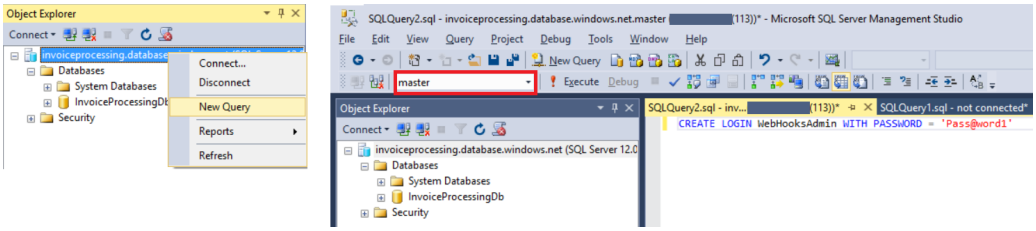
V okně *Object Explorer* pod uzlem *Databases* vidíte svoji databázi. Klikněte pravým tlačítkem na její název a zvolte *New Query* pro vytvoření nového dotazu.



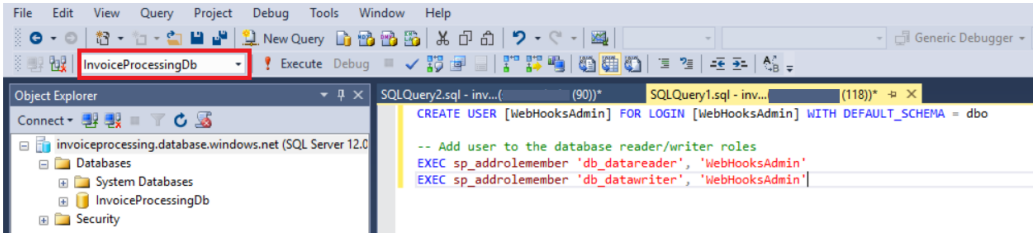
Zkontrolujte, že dotaz opravdu provádíte nad vaší databází a vytvořte požadované tabulky. Zobrazené SQL skripty jsou součástí přiloženého CD. Výsledný dotaz spusťte buď kliknutím na tlačítko *Execute*, nebo stiskem klávesy [F5].



Následně vytvořte nový účet v master databázi použitím SQL příkazu `CREATE LOGIN`.



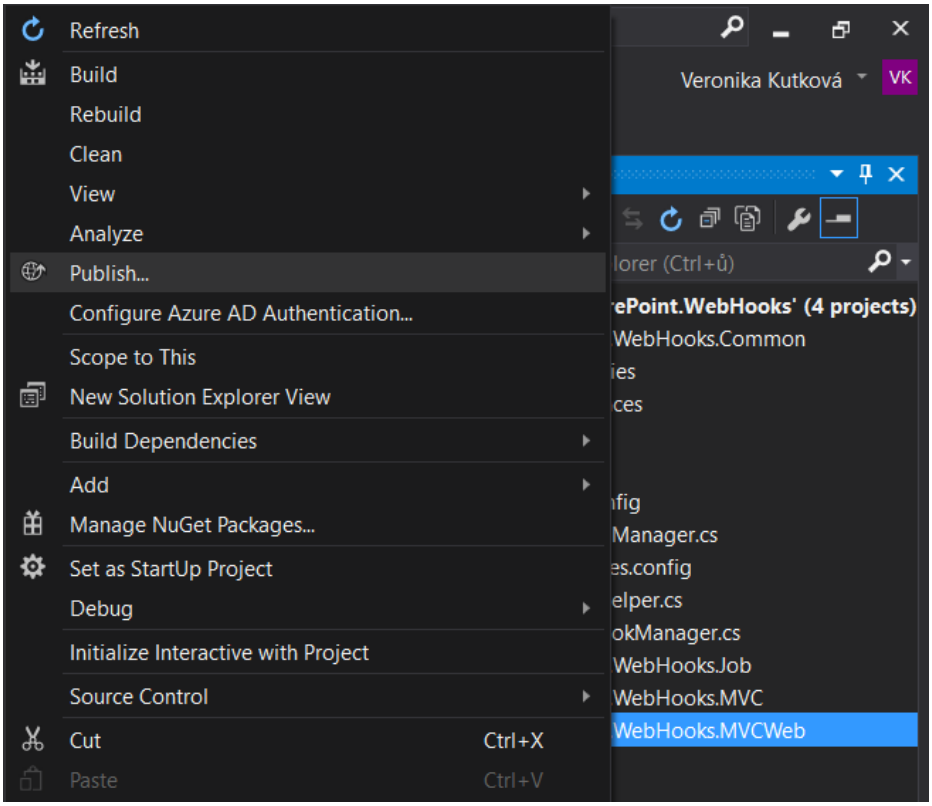
V dalším kroku (již opět nad vaší databází) vytvořte příkazem `CREATE USER` databázového uživatele a přidělte mu přístupová práva ke čtení a zápisu.



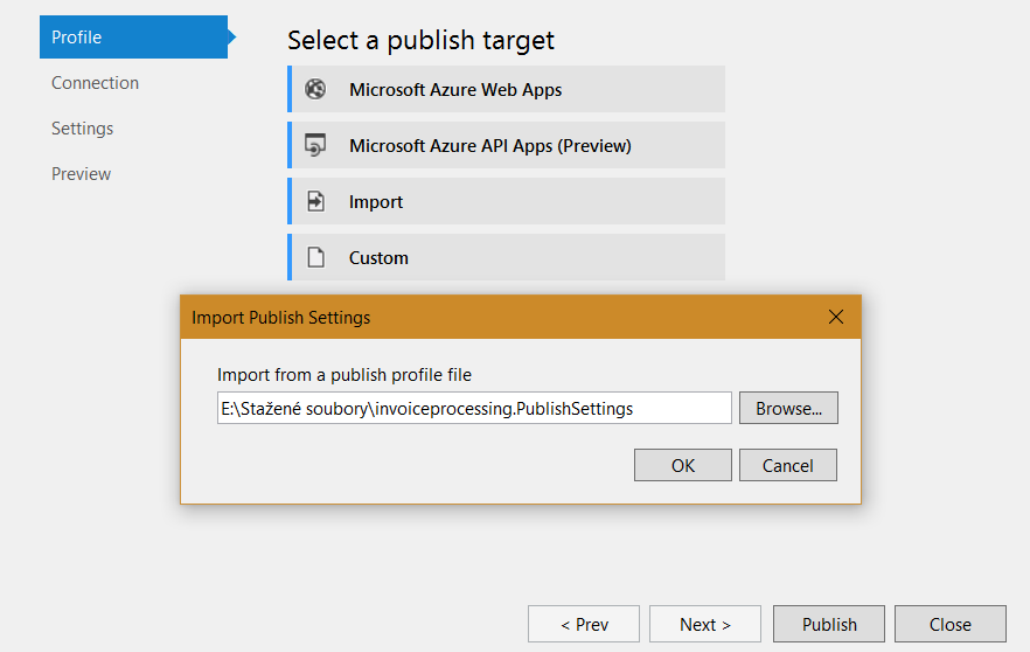
B.4 Nasazení add-inu (Visual Studio, SharePoint)

Spusťte Visual Studio a otevřete *SharePoint.WebHooks.sln* (k dispozici na přiloženém CD).

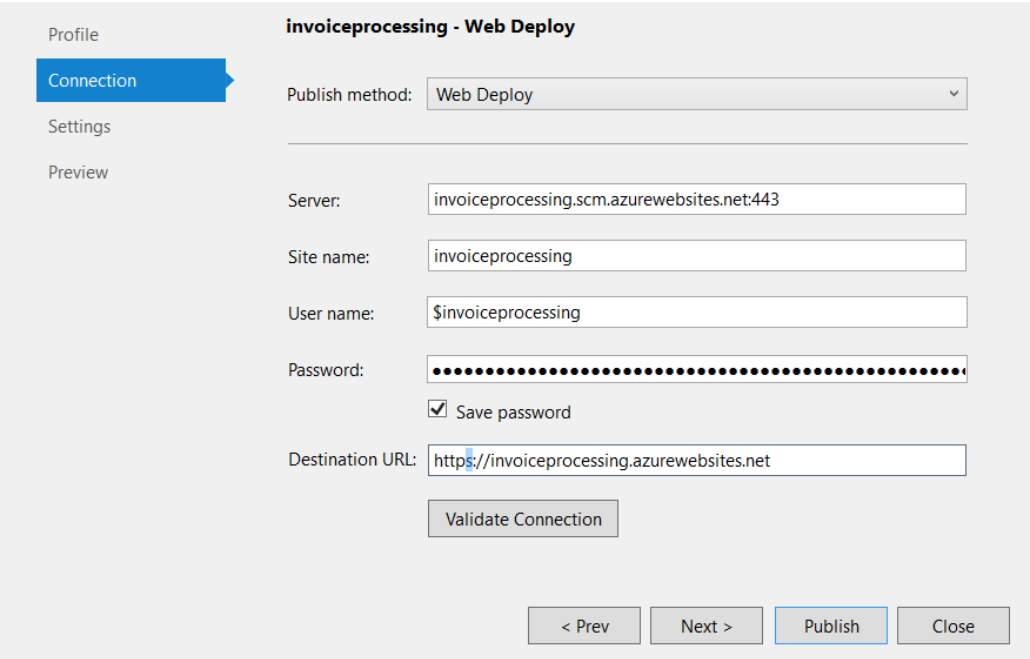
Jako první nasaďte SharePoint add-in do Azure. Klikněte pravým tlačítkem na projekt SharePoint.WebHooks.MVCWeb a z kontextové nabídky zvolte možnost *Publish*.



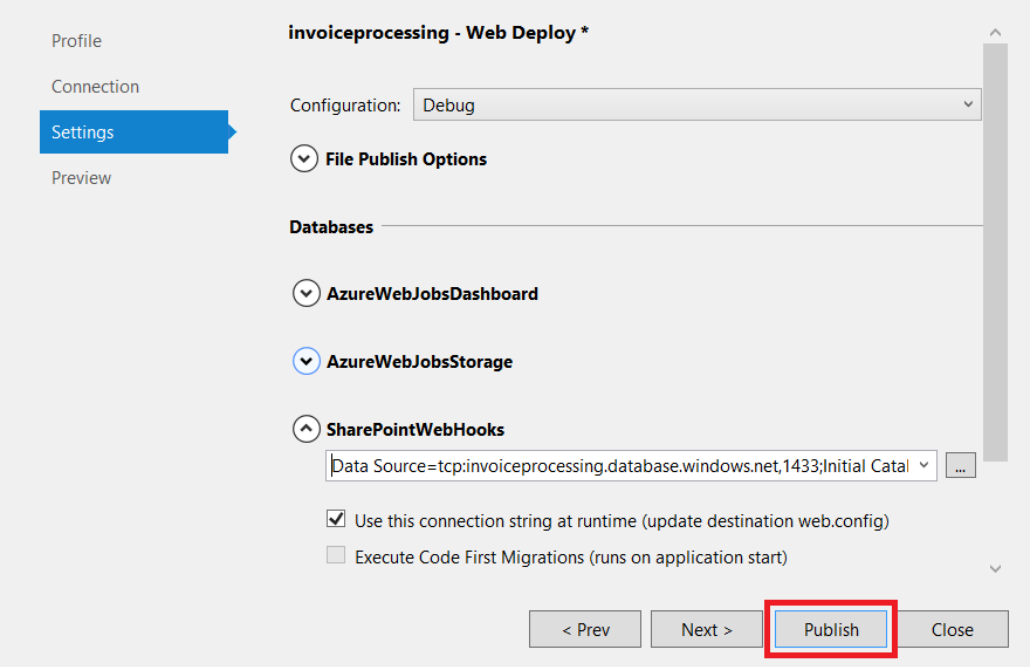
V okně *Publish Web* klikněte na tlačítko *Import* a vyberte profil publikování, který jste získali při vytváření webové aplikace v Azure. Potvrďte tlačítkem *OK* a klikněte na *Next*.



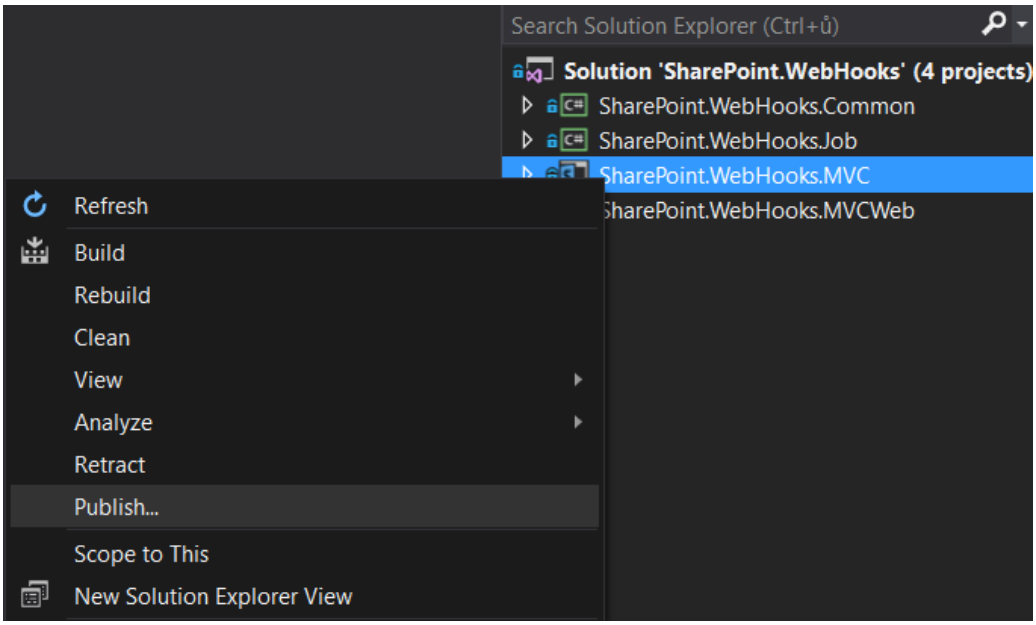
Zkontrolujte předvyplněné údaje, nahraďte v poli *Destination URL* protokol *http* za *https* a klikněte na *Next*.



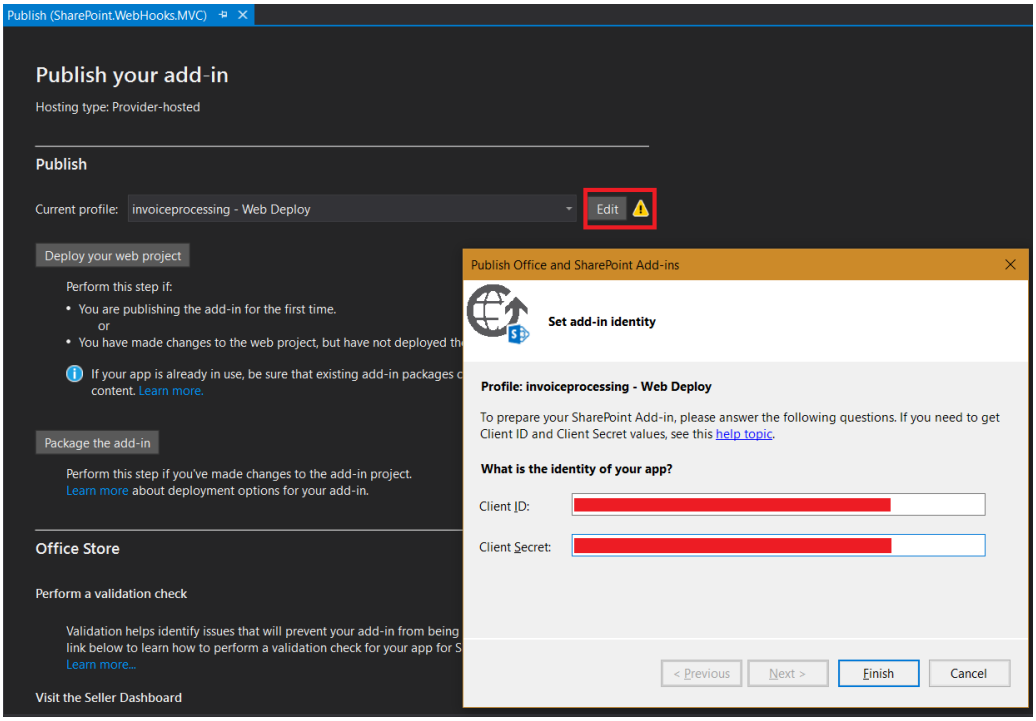
Pokud chcete vzdáleně debugovat, zvolte v poli *Configuration* možnost *Debug*. V bloku *Databases* vyberte v poli SharePointWebHooks vaši databázi a klikněte na tlačítko *Publish*.



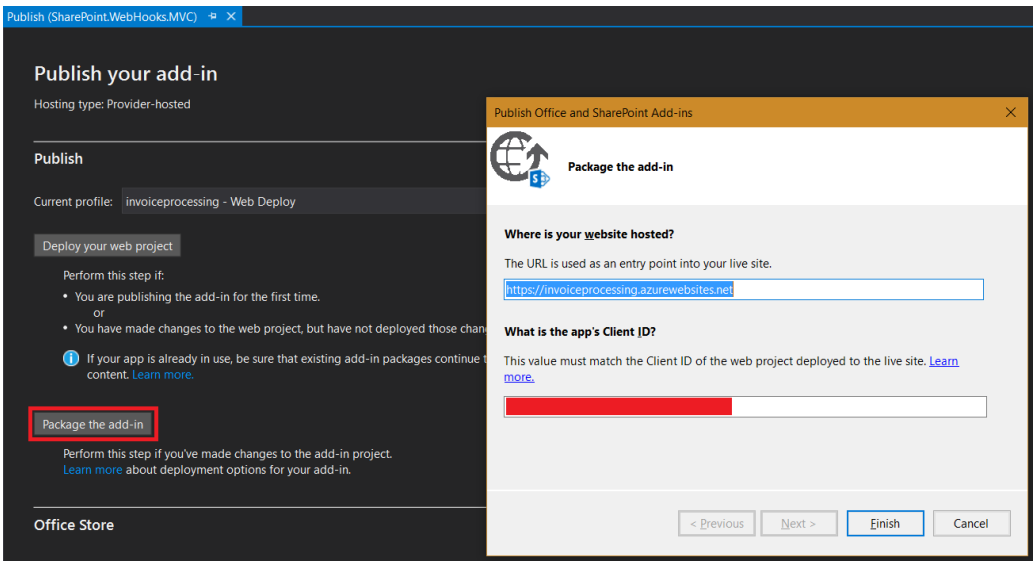
Nyní nasaďte add-in do SharePointu. Klikněte pravým tlačítkem na projekt SharePoint.WebHooks.MVC a zvolte možnost *Publish*.



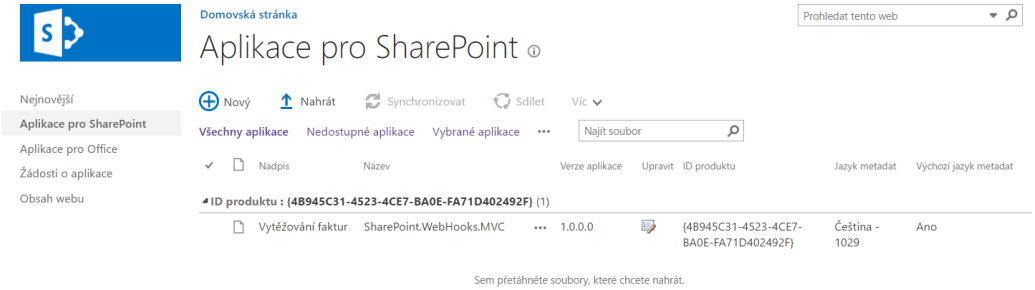
V následujícím okně klikněte na tlačítko *Edit* a vyplňte ID klienta a tajný klíč klienta, který jste si vygenerovali v SharePointu (viz B.2). Potvrďte stiskem tlačítka *Finish*.



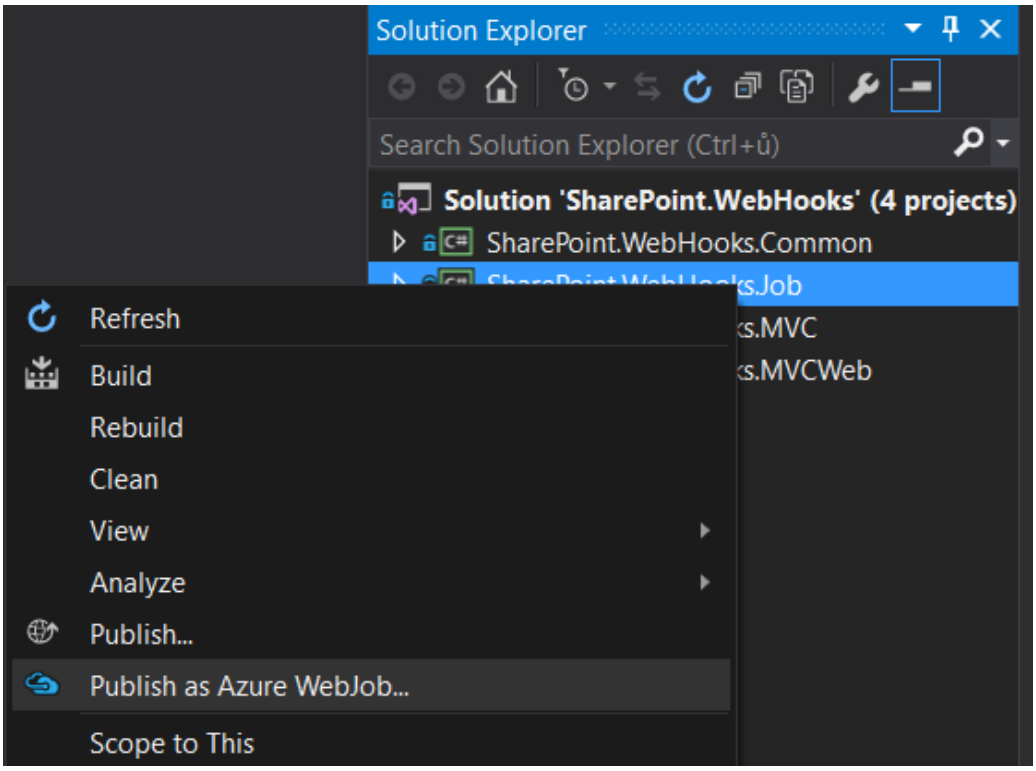
Klikněte na tlačítko *Package the add-in*, zkontrolujte předvyplněné údaje a potvrďte tlačítkem *Finish*.



Otevře se vám složka s výsledným add-inem (soubor s názvem *SharePoint.WebHooks.MVC.app*). Tento soubor nahrajte do Katalogu aplikací (*App Catalog*) v SharePointu (potřebujete admin přístupová práva). Zadejte adresu `http://<tenant>-admin.sharepoint.com`, klikněte na *Aplikace* a poté na *Katalog aplikací*. Zvolte *Aplikace pro SharePoint* a přidejte svůj add-in[5].

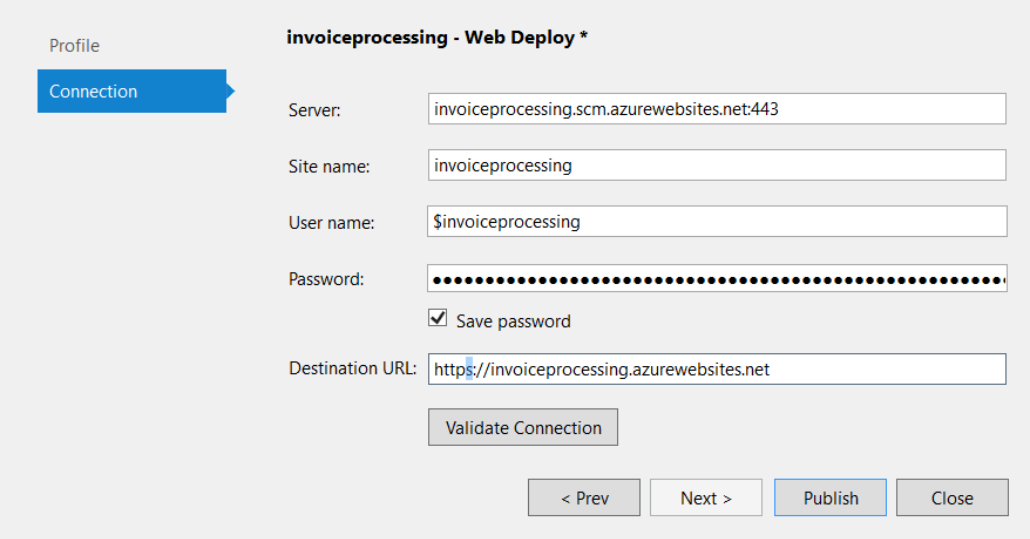


Posledním krokem je nasazení Azure WebJob (webové úlohy). Klikněte pravým tlačítkem na projekt SharePoint.WebHooks.Job a vyberte možnost *Publish as Azure WebJob*.



[5]Při publikaci nové verze je potřeba změnit číslo verze v souboru *AppManifest.xml* v projektu *SharePoint.WebHooks.MVC*.

Importujte profil publikování, klikněte na *Next* a v poli *Destination URL* nahraďte protokol *http* za *https*. Dokončete stiskem tlačítka *Publish*.



B.5 Python a Flask framework

Tato část příručky vychází z dokumentace dostupné na [31].

Doporučeným způsobem instalace závislostí v Pythonu je použití nástroje *pip*, který je od verze 2.7.9 standardní součástí instalace Pythonu. Příkazem `pip install <nazev-balicku>` lze jednoduše nainstalovat jakýkoliv balíček dostupný v databázi Python balíčků PyPI (Python Package Index)[6].

Pro používání jednoduchého frameworku Flask stačí mít nainstalovanou knihovnu *Flask* příkazem `pip install Flask`.

B.5.1 Virtuální prostředí

Při vývoji v Pythonu je vhodné používat tzv. virtuální prostředí (*virtual environment*). Virtuální prostředí představuje prostředí svázané vždy s jedním konkrétním projektem. Díky němu je možné snadno vyvíjet v různých verzích Pythonu a s různými verzemi závislostí na jednom počítači. Verze Pythonu ve virtuálním prostředí musí odpovídat verzi použité na serveru.

Nástroj pro správu virtuálních prostředí nainstalujete z příkazové řádky příkazem `pip install virtualenv`. Následným spuštěním příkazu `python -m virtualenv env` v adresáři projektu vytvoříte složku *env* představující

[6] `http://pypi.python.org/pypi`

virtuální prostředí tohoto projektu a příkazem `env\Scripts\activate` ho aktivujete. Veškeré závislosti projektu budou uloženy zde, aniž by jakýmkoliv způsobem ovlivňovaly okolní projekty.

B.5.2 Správa závislostí

Adresář *env* nepatří do git repository, každý vývojář si při práci na projektu vytváří své vlastní virtuální prostředí. Závislosti na knihovnách jsou uchovávány v textovém souboru *requirements.txt* umístěném v kořeni projektu ve tvaru zřejmém z příkladu B.1.

```
Flask==0.10.1
Jinja2==2.9.6
Werkzeug==0.11.5
```

Příklad B.1: Ukázka souboru *requirements.txt*.

Veškeré závislé balíčky nainstalované ve virtuálním prostředí příkazem `pip install` lze promítnout do tohoto souboru příkazem `pip freeze > requirements.txt`. Při nasazování aplikace (viz Instalační příručka B.1.2, Webová aplikace – Flask) se Azure automaticky postará o instalaci závislostí uvedených v tomto souboru nástrojem *pip*.

Umístěním (prázdného) souboru s názvem *.skipPythonDeployment* do kořenového adresáře projektu je možné vynechat instalaci balíčků při každém dalším nasazení. Při dodatečné instalaci další závislosti stačí tento soubor opět smazat a vynutit tak čistou instalaci závislostí při dalším nasazení aplikace na Azure.

B.5.3 Možné problémy

Instalace balíčku, který není v PyPI

Při nasazení aplikace do Azure jsou všechny závislosti obsažené v textovém souboru *requirements.txt* nainstalovány příkazem `pip install`. Problém však nastává v případě, že některá z uvedených závislostí není součástí PyPI a nelze ji proto nainstalovat nástrojem *pip*.

Východiskem je použití předkompilovaných balíčků (tzv *wheels*) ve formátu *.whl*. Jejich vytvoření musí proběhnout na stroji se stejnou konfigurací jako má cílový server (tj. v případě nasazení na Azure se jedná o OS Windows/32-bit/Python verze 2.7 nebo 3.4). Pro vytvoření předkompilovaných balíčků potřebujete balíček *wheel*, který nainstalujete příkazem `pip install wheel` ve svém virtuálním prostředí.

V praxi se nám osvědčilo nekombinovat *wheels* s instalací pomocí příkazu `pip install`. Doporučujeme proto vytvořit *.whl* soubory postupně pro

všechny závislosti příkazem `pip wheel <package>==<version>`, např. tedy `pip wheel Flask==0.10.1`. Všechny *wheels* poté vložte do složky *wheelhouse* v kořenovém adresáři projektu a zahrňte tuto složku do git repository.

Závislost, která není dostupná v PyPI, ale na GitHubu (případ knihovny *lshhdc* používané ve vytěžovací komponentě), nainstalujte ve svém prostředí příkazem `pip install git+GIT_URL` (v případě knihovny *lshhdc* tedy `pip install git+https://github.com/go2starr/lshhdc.git`). Nezapomeňte ji zahrnout do *requirements.txt* příkazem `pip freeze > requirements.txt`. Vytvořte *.whl* stejným způsobem jako u ostatních knihoven.

Abyste zabránili nástroji *pip* v automatickém stahování a instalaci balíčků uvedených v *requirements.txt*, přidejte na začátek tohoto souboru řetězec `--find-links wheelhouse`. Díky tomu se *pip* vždy nejprve podívá do adresáře *wheelhouse* a teprve v případě nenalezení vhodného *.whl* souboru začne prohledávat PyPI. Pokud však máte v adresáři *wheelhouse* předkompilované všechny závislosti, přidejte na první řádku `--no-index`, který zajistí, že *pip* nebude prohledávat PyPI vůbec.

Výslednou podobu souboru *requirements.txt* zachycuje příklad B.2.

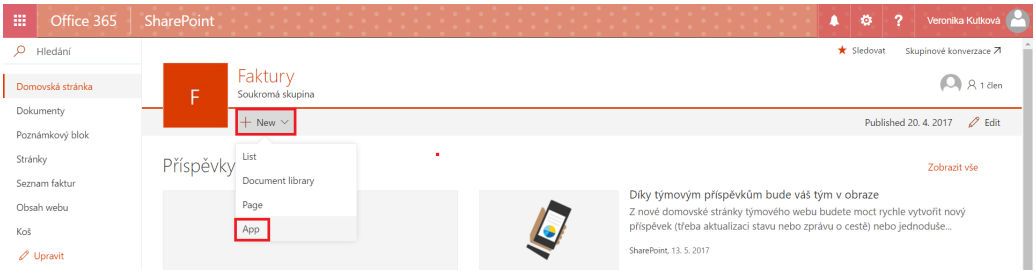
```
--no-index
--find-links wheelhouse
Flask==0.10.1
Jinja2==2.9.6
Werkzeug==0.11.5
```

Příklad B.2: Ukázka souboru *requirements.txt*.

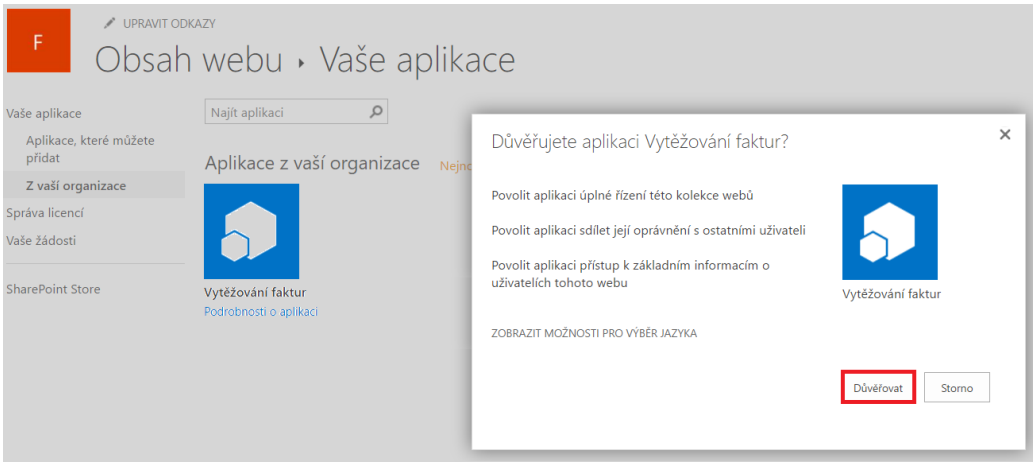
C Uživatelská příručka

C.1 Přidání add-inu do webu

Pokud chcete do webu v SharePointu přidat nový add-in (umístěný v Katalogu aplikací[1]), klikněte na Domovské stránce příslušného webu na tlačítko *New* (*Nové*) a zvolte možnost *App* (*Aplikace*).

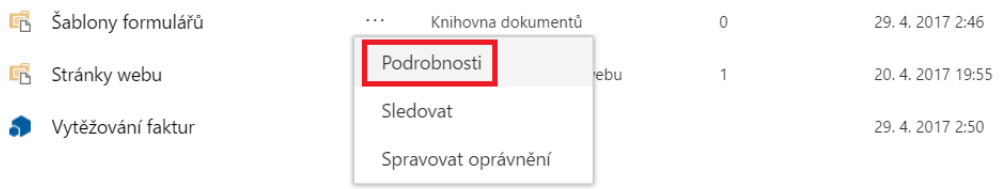


V menu na levé straně obrazovky se přepněte na aplikace *Z vaší organizace* a vyberte add-in kliknutím na jeho název nebo ikonu. Stiskem tlačítka *Důvěřovat* udělte add-inu potřebná oprávnění. Následně je add-in nainstalován a přidán mezi položky v Obsahu webu.



[1]Přidat add-in do Katalogu aplikací může pouze uživatel s admin přístupovými právy, viz Instalační příručka, kapitola B.4

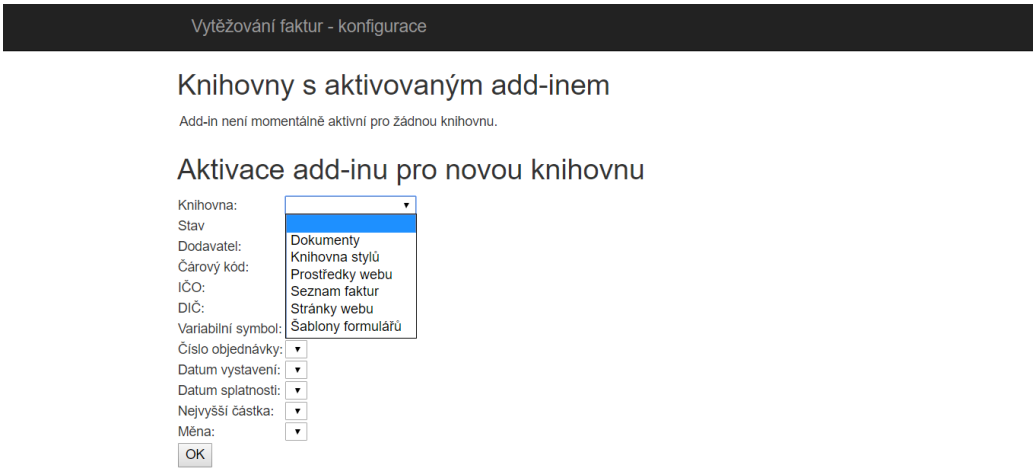
V případě, že byla vydána nová verze add-inu, proveďte jeho upgrade kliknutím na tři tečky vedle názvu add-inu a zvolte *Podrobnosti*.



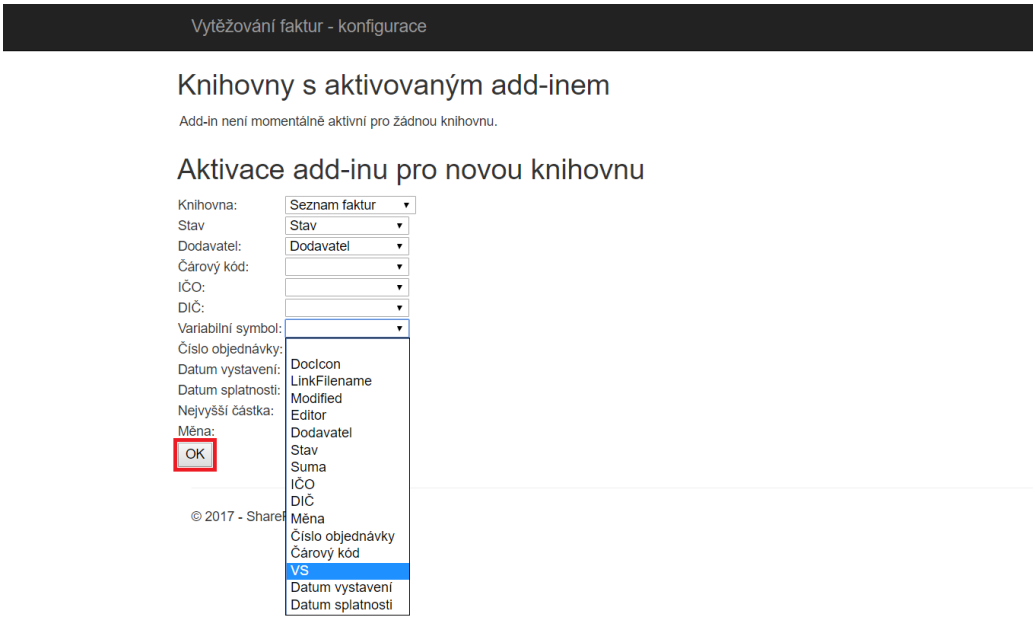
Novou dostupnou verzi získejte stiskem tlačítka *Získat*.

C.2 Použití add-inu

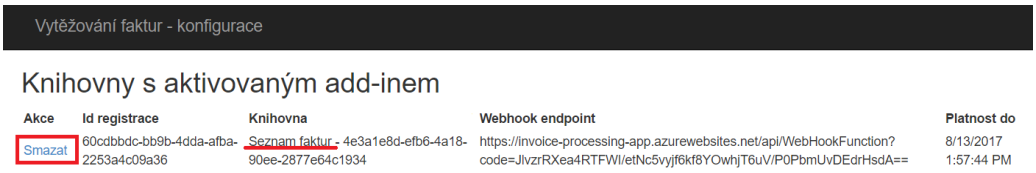
Spusťte add-in kliknutím na jeho název v Obsahu webu. Otevře se nová stránka, na které je možné add-in pro konkrétní knihovny aktivovat či deaktivovat.



Z prvního rozbalovacího seznamu vyberte knihovnu, pro kterou chcete add-in aktivovat (tj. chcete zajistit vytěžování dat pro všechny nové soubory přidané do této knihovny). Ostatní rozbalovací seznamy slouží k nastavení mapování jednotlivých údajů, které lze z faktury vytěžit, na sloupce vybrané knihovny. Pokud nechcete některý z údajů vytěžovat, ponechte volbu prázdnou. Stiskem tlačítka *OK* add-in aktivujte.



Po aktivaci se v horní části stránky objeví nový záznam potvrzující aktivaci add-inu pro zvolenou knihovnu. Pokud chcete add-in pro některou z knihoven opět deaktivovat (chcete zrušit vytěžování nově přidaných souborů), klikněte na *Smazat*.



Pokud nyní do knihovny s aktivovaným add-inem nahrajete novou fakturu, add-in se postará o vytěžení požadovaných informací. Po dokončení vytěžování je hodnota *Stav* změněna na *Zpracováno*.