Návrh a implementace aplikace pro interní sdílení zdrojů

Petr Svoboda

Bakalářská práce
2023

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

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

Akademický rok: 2022/2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Petr Svoboda**
Osobní číslo:	**A20510**
Studijní program:	**B0613A140020 Softwarové inženýrství**
Forma studia:	**Prezenční**
Téma práce:	**Návrh a implementace aplikace pro interní sdílení zdrojů**
Téma práce anglicky:	**Design and Implementation of an Application for Internal Resource Sharing**

Zásady pro vypracování

1. Nastudujte a popište problematiku uchování a sdílení zdrojů.
2. Prostudujte stávající řešení a možnosti.
3. Navrhněte webovou aplikaci pro uchování a sdílení zdrojů.
4. Zvolte a popište vhodné prostředky pro implementaci.
5. Implementuje navrženou aplikaci.
6. Řešení vhodně otestujte a popište výsledky.

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

Seznam doporučené literatury:

1. Gorman, B. L., Gorman, B. L. (2022). Practical entity framework core 6 database access for enterprise applications. Apress.
2. Esposito, D. (2018). Programming Asp.Net Core. MicrosoftPress.
3. Efron, B., Tibshirani, R., & Tibshirani, R. J. (1994). An introduction to the bootstrap. Chapman & Hall/CRC. https://doi.org/10.1007/978-1-4899-4541-9
4. Meloni, J. C., Kymin, J. (2019). Html, Css, and JavaScript all in one. Sams.
5. White, M. S. (2011). The Intranet Management Handbook. Information Today.

Vedoucí bakalářské práce: **Ing. Petr Žáček, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání bakalářské práce: **2. prosince 2022**
Termín odevzdání bakalářské práce: **26. května 2023**

L.S.

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

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

Ve Zlíně dne 7. prosince 2022

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 25.5.2023

Petr Svoboda, v.r.
podpis studenta

ABSTRAKT

Tato bakalářská práce se zabývá vytvořením webové aplikace, která slouží jako úložiště pro interní sdílení digitálních dokumentů a záznamech o fyzických předmětech. Teoretická část se zaměřuje na nastínění potřeb sdílení těchto zdrojů v rámci organizace a jejich souvislostí se znalostním managementem. Jsou popsány přínosy, které efektivní sdílení těchto zdrojů může mít na spolupráci a správu znalostí členů organizace. Dále jsou v teoretické části popsány již existující aplikace pro obdobné účely, technologie vhodné pro implementaci této aplikace a porovnání výhod tvorby vlastní aplikace pro podobné sdílení namísto použití již existujících. Praktická část práce se zaměřuje na samotnou implementaci aplikace, popsána je její struktura zahrnující popis způsobu práce se soubory, jejich členění do sekcí a nastavování přístupových práv sekcím pro jednotlivé uživatele.

Klíčová slova: interní sdílení zdrojů, webová aplikace, úložiště, znalostní management, ASP. NET Core

ABSTRACT

This bachelor thesis deals with the creation of a web application that serves as a repository for internal sharing of digital documents and records of physical objects. The theoretical part focuses on outlining the needs for sharing these resources within an organisation and their relationship to knowledge management. The benefits that effective sharing of these resources can have on collaboration and knowledge management among members of an organization are described. Furthermore, the theoretical section describes existing applications for similar purposes, the technologies suitable for implementing such an application, and a comparison of the benefits of creating a custom application for similar sharing instead of using existing ones. The practical part of the thesis focuses on the actual implementation of the application, describing its structure including a description of how application works with files, how application divides them into sections and how application sets access rights to the sections for individual users.

Keywords: internal resource sharing, web applications, storage, knowledge management, ASP. NET Core

OBSAH

ÚVOD

V dnešním dynamickém světě hraje efektivní organizování a sdílení zdrojů velkou roli, proto se prostředí organizací zabývá otázkou efektivního sdílení svých zdrojů. To může představovat potřebu uchovávání různých informací jak v podobě digitálních dokumentů, tak záznamů o předmětech, které nebyly digitalizovány a jsou k dispozici ve fyzické podobě. Jejich efektivní správa pomáhá maximalizovat produktivitu a optimalizovat využití těchto zdrojů, s čímž je spojeno zlepšení spolupráce a komunikace mezi zaměstnanci. Sdílení zdrojů poskytuje prostředí, ve kterém lze snadno vyměňovat informace, soubory a znalosti. Zaměstnanci mají rychlý a jednoduchý přístup k potřebným zdrojům, což podporuje efektivitu jejich práce a umožňuje jim lépe spolupracovat na projektech a úkolech.

Důležitým aspektem sdílení zdrojů je také zmíněná možnost uchovávat informace o fyzických zdrojích. Mnoho organizací stále vlastní materiály, zařízení a další fyzické objekty, které je třeba spravovat a sdílet. Z toho důvodu se navržená aplikace zaměřuje mimo sdílení digitálních zdrojů i na tento aspekt a umožňuje uchovávat informace o fyzických zdrojích společně s digitálními.

Cílem této bakalářské práce je tedy popsat možnosti a výhody sdílení takových zdrojů a implementovat aplikaci, která usnadní tento proces.

I. TEORETICKÁ ČÁST

1 PROBLEMATIKA SDÍLENÍ ZDROJŮ

Ačkoliv je tato práce zaměřena na sdílení zdrojů v rámci organizace, je vhodné na začátek nastínit další oblasti, kterých se sdílení jako takové týká a co může představovat, jelikož se jedná o široký pojem. Při vymezování problematiky sdílení zdrojů je v prvé řadě potřeba definovat, co se takovými zdroji může rozumět, jelikož zde velkou roli hraje kontext, ve kterém se dané sdílení odehrává.

Na jedné straně se ve firemním prostředí může jednat o sdílení interních dokumentů, znalostí nebo firemního vybavení mezi zaměstnanci organizace, zatímco v kontextu hardwaru počítačů můžeme mluvit třeba o sdílení prostředků typu tiskárna, skener paměti nebo procesorový čas. V poslední době zaznamenal rozšíření pojem tzv. sdílená ekonomika, jejíž koncept přináší užitek tím způsobem, že majitelé předmětu, kteří jej využívají pouze příležitostně, tento předmět poskytují k užívání těm, pro které je nemožné nebo při jejichž míře užívání tohoto předmětu nevýhodné tento předmět vlastnit. Týká se například carsharingu, který umožňuje užívání automobilů lidem, aniž by byli majiteli, sdílení nemovitostí (Airbnb) a dalších [29, 30].

Sdílení zdrojů se tedy vyskytuje v různých oblastech a každá z těchto oblastí může svou podstatou vyžadovat rozlišné přístupy s řešením. Z výše uvedeného je zřejmé, že zdroje a oblasti sdílení těchto zdrojů můžou mít různou formu, nicméně u všech případů sdílení je s rozvojem digitálních technologií snaha o (pokud je to u typu sdíleného zdroje možné) zapojení softwaru do tohoto sdílení za účelem jeho zefektivnění a lepšího zpřístupnění těchto zdrojů a z tohoto důvodu vznikají různé informační systémy, které toto sdílení podporují.

Na základě výše uvedeného se dá říci, že řízené sdílení zdrojů představuje způsob, jak pokud možno co nejefektivnějším způsobem hospodařit s prostředky, které máme k dispozici.

1.1 Řízení a sdílení informačních zdrojů v organizaci

Tato kapitola si klade za cíl nastínit potřeby organizací v oblastech, které se zabývají sdílením znalostních zdrojů a dovednostmi zaměstnanců v rámci organizací tak, aby bylo bližší, v důsledku jakých důvodů vznikají informační systémy, které tyto oblasti řízení podniku podporují.

Organizace různých typů obzvláště v dnešní době, která s sebou nese velké množství informací, shromažďují data potřebná ke svému fungování, ať už o sobě samých, o svých zaměstnancích nebo zákaznících atd. přičemž organizace potřebují tato data interně sdílet.

Uchovávání všech takových informací na papírech založených ve složkách a kartotékách je dnes vzhledem k možnostem informačních technologií neefektivní, zdlouhavé až nemožné. Data se mohou uchovávat různě pojmenována na sdíleném disku, kde si je zaměstnanci můžou sdílet, což se ale s rostoucím množstvím těchto dat může stávat nepřehledné, mohou vznikat problémy s vyhledáváním, protože nikdo vlastně přesně neví, kde co hledat a zaměstnanci se v záplavě nijak více nestrukturovaných dat mohou ztrácet [17]. Vzhledem k tomu, jak se organizace může vyvíjet, je často potřeba publikovat různá oznámení, informovat o novinkách, umožnit diskusi o nadcházejících změnách nebo jen na jednom místě přehledně poskytnout seznam zajímavých odkazů a dalších zdrojů [18].

Podniky tedy disponují aktivy, které dokážeme přímo zaznamenat do finančních výkazů a s tím souvisí charakteristické rysy podnikání, které představují stále se zvětšující rozdíl mezi účetní a tržní hodnotou společností [7, 1].

V případě některých organizací tvoří intelektuální kapitál značnou část jejich tržní hodnoty, kdy například u společností ABB (Arsea Brown Boveri) a GE (General Electric) dosahuje i přes 80% [1, str. 31].

Důležitost tohoto typu bohatství může popisovat tento úryvek: *„V roce 1983 vlastní jmění firmy Coca-Cola představovalo 40% její tržní kapitalizace, tzn. celkové hodnoty akcií v oběhu na kapitálovém trhů. V roce 1997 tento podíl poklesl až na 4%. Toto „neviditelné“ bohatství přestavuje pro firmy čím dál tím větší konkurenční výhodu. … Tak například bohatství Microsoftu spočívá převážně v hlavách jejich zaměstnanců.“* [7, str. 1].

Dle P. F. Druckera jsou pro tuto problematiku řešením pouze *„obaly na znalosti“*. (Drucker, 1993). Tyto obaly můžou být reprezentovány například pro tuto oblast určenými informačními systémy [7 ,1].

1.2 Knowledge management

Jak je uvedeno výše, za jeden z nejcennějších podnikových zdrojů můžeme považovat tzv. intelektuální kapitál. Ten se vztahuje na všechny znalosti, informace a dovednosti, které organizace vlastní nebo které jsou organizaci dostupné. Tyto zdroje jsou pro ni často velmi hodnotné a závisí na nich její úspěšné fungování a další rozvoj [5].

Management řízení firmy obsahuje několik oblastí a jednou z nich je knowledge management (znalostní management). Vymezení tohoto pojmu může být značně široké, nicméně můžeme jej chápat jako systematický proces, který se zmíněným intelektuálním

kapitálem zabývá a snaží se jej řídit [7, str. 5], [1. str. 15]. Jinými slovy jde o činnost, která si klade za cíl získat co nejlepší zdroje znalostí a co nejlépe je využít [2, str. 4]. Podporuje tedy získávání informací a rozvíjení znalostí u zaměstnanců a s tím související rozpoznávání, tvoření, ukládání a sdílení znalostí v rámci organizace[4].

Mezi zaměstnanci se může často vyskytovat potřeba vědění, kterou již společnost jako taková má a je důležité, aby existoval způsob, jakým tyto znalosti efektivně mezi zaměstnance distribuovat [4].

1.2.1 Metody knowledge managementu

Existuje několik možností, které lze k aplikaci znalostního managementu využít. Může se jednat o školení s praktickými ukázkami, které bude provádět přímo zaměstnanec firmy, který již potřebné znalostí má a pomůže je tímto školením distribuovat mezi své kolegy v rámci celopodnikového nebo individuálního školení. Forma školení přináší výhodu v tom, že je během něj možné vést v reálném čase diskuse a řešit případné dotazy zaměstnanců a v případě potřeby vše znovu vysvětlit. Nevýhodou může být větší časová náročnost [4].

Další možností je dokumentace s návody na řešení často se vyskytujících problémů a dotazů. Písemná sdělení umožňují informace jednoduše uchovat a v případě potřeby je poskytnout k dispozici těm, kteří je potřebují. Dokumenty mají výhodu ve snadné znovupoužitelnosti, nicméně je potřeba myslet na jejich aktuálnost a s tím související aktualizace informací v nich [4].

S rozšiřující se technologickou dobou začala vznikat internetová fóra/intranety představující online platformu pro předávání znalostí, která je obvykle dostupná interně pouze zaměstnancům. Stejně jako školení může mít výhodu v možnosti diskuzí, není zde ale potřeba schůze zaměstnanců na konkrétním místě v daný čas, je více flexibilní. Vyhledávání může být snazší než v případě obyčejných fyzických dokumentů. Rovněž je zde potřeba myslet na aktuálnost informací a vyhnout se čerpání ze starých neaktuálních diskusí [4].

1.3 Jak může řízení znalostí pomoci

Pokud firma neumí dokumentovat a uchovávat různé klíčové informace a znalosti potřebné pro své fungování, marní tím drahý čas svých zaměstnanců. Čím efektivnější společnost v oblasti znalostního managementu bude a čím efektivnější bude mít prostředky na sdílení informací mezi zaměstnanci, tím lepší výsledky bude mít [4].

Při efektivním řízení znalostí je možné vyhnout se řešení opakovaně kladených otázek, opakovaným chybám. Zkušení zaměstnanci budou méně zatěžováni začátečnickými problémy méně zkušených kolegů a budou mít více prostoru na řešení klíčových problémů při své práci. S tím souvisí nižší náklady na zaškolení a adaptaci nových zaměstnanců do procesů organizace. Snadno dostupným řešením pro často vyskytující se komplikace tedy umožníme zaměstnancům tyto komplikace snáze a rychleji řešit. S každým odcházejícím zaměstnancem odchází i důležité znalosti, které nemusí být ostatním zaměstnancům zřejmé a které by se mohly časem vytratit. Z toho důvodu je potřeba, aby (pokud to typ znalosti umožňuje) tyto znalosti měla organizace vhodným způsobem zaznamenána. Jasná promyšlená struktura nejen vzdělávacích dokumentů tak, aby obsahovala vše důležité, minimalizuje možnost, že se důležité znalosti při začleňování nových zaměstnanců zapomenou těmto zaměstnancům sdělit, stávající zaměstnanec má možnost kdykoliv do sdílených dokumentů nahlédnout a nezatěžovat tak ostatní. Řízení znalostí nemusí být prospěšné pouze v interních záležitostech organizace, ale může pomoci i k lepšímu porozumění zákazníkům, dodavatelům a dalším zájmovým skupinám. Správná optimalizace práce s intelektuálním kapitálem může vést ke snížení nákladů [6].

Výhody sdílení znalostí bychom mohli shrnout jako:

- Zlepšení efektivity práce v důsledku lepšího využití znalostí
- Optimalizace zaškolení nových zaměstnanců
- Zamezení ztrát znalostí v důsledku fluktuace zaměstnanců nebo jiných příčin
- Podpora inovativnosti (sdílení znalostí mezi zaměstnanci vede ke vzájemné spolupráci a s tím souvisejícímu vzniku nových nápadů)
- Snížení duplicitních činností (prostředky nebudou vynakládány na opakující se činnosti, nebude se řešit již vyřešené)
- Zlepšení komunikace a spolupráce napříč organizací
- Zvýšení konkurenceschopnosti (sdílení poznatků mezi zaměstnanci podporuje lepší reakci na změny v trhu)
- Znalosti jsou ve správný čas na správném místě

[4]

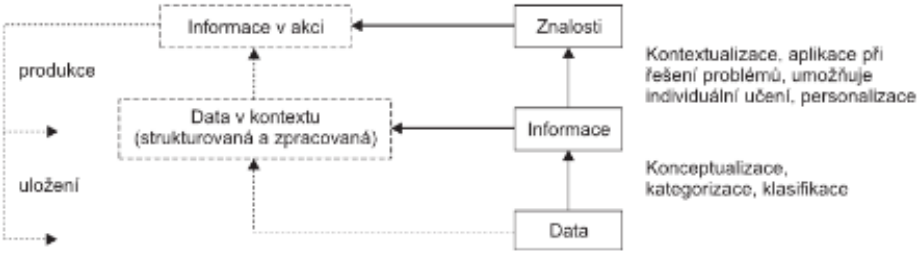
1.3.1 Důležitost organizace znalostí

Pokud jsou znalosti vhodným způsobem organizované, resp. je nám jejich organizační struktura známa a tvořena, bude se v nich i lépe vyhledávat. Snadná možnost vyhledávání s sebou přináší celkové zefektivnění práce se znalostmi, tudíž je možné je snáze aplikovat na konkrétní situace a problémy [3].

1.4 Co znalost představuje

V předchozích kapitolách byl nastíněn kontext sdílení zdrojů a s tím související informace a znalosti. Tyto pojmy a rozdíl mezi nimi nemusí být vždy zřejmé, proto si tato kapitola klade za cíl nastínit určité spojitosti týkající se tématu znalostí.

Různi autoři zabývající se tématy týkající se znalostí můžou mít na definici znalosti individuální pohledy. Pro tyto pohledy je ale typické, že znalost začleňují do určité pozice v hierarchii mezi souvisejícími pojmy – nejčastěji se vyskytující je trojice data, informace, znalosti, kterou popisuje následující obrázek.



Obrázek 1. Vztah mezi daty, informacemi a znalostmi [1, str. 25]

Daty se obvykle rozumí prosté údaje typu text, obrázky, číselné údaje (výsledky měření aj.), dá se tedy říct, že představují „základní suroviny“ pro bližší specifikaci a zpracování, na kterých je možné dále stavět informace. Informaci můžeme chápat jako interpretaci analyzovaných dat, která jsou již nějakým způsobem formátována, filtrována, sumarizována a jsou mezi nimi popsány vztahy. Znalost představuje schopnost práce s informacemi, další zpracování a uvažování nad informacemi, které vede ke konkrétním akcím, procedurám a rozhodnutím potřebných pro řešení problémů. Jedná se také o schopnost pochopení příčin a důsledků vztahů. To, že máme informace, ještě nutně nemusí znamenat, že je umíme efektivně využít v náš prospěch. Přečtení kuchařky nutně neznamená, že je z člověka dobrý kuchař, samotné informace o pilotování letadla z člověka neudělají profesionálního pilota (můžeme sem tedy zařadit zkušenosti, praxi, intuici) [1, str. 26].

„Znalost je uvažování nad daty a informacemi za účelem aktivního umožnění výkonu, řešení problémů, rozhodování, učení a výuky.“ [Beckman 1997, 1. str. 26]

Výše zmíněnou hierarchii je mimo další možné najít jinými autory rozšířenou o úroveň „moudrost“ [1, 25], která je znázorněna na osách popisujících vztah mezi poznáním a chápáním souvislostí.

Obrázek 2. Data, informace, znalost a moudro [1, str. 26]

Čím vyšší úroveň v pyramidě, tím více je potřeba lidské myšlení.

1.4.1 Klasifikace znalostí

Důležité je zmínit, že ne každou znalost lze jednoduše popsat slovy, v dokumentu či jinak zachytit tak, aby se následně dala sdílet například prostřednictvím informačních systémů a jiných komunikačních technologií [1, str 29].

Stejně jako na obecnou definici znalostí, tak i na jejich bližší klasifikaci se názory autorů zabývajících se tímto tématem můžou lišit a existuje více kritérií, dle kterých je lze kategorizovat [1, str 28].

Autorem jedné z těchto klasifikací používajících se v oblasti znalostního managementu je M. Polanyiny (Polanyiny 1996). Jeho velmi používaná kategorizace rozděluje znalosti na základní typy – explicitní a tacitní [1, str. 29].

Explicitní znalost lze nějakým způsobem vyjádřit – třeba formou vyslovení, výrokové logiky, nebo znázornit nakreslením jako obrázek a z toho důvodu je obvykle dobře přenositelná [7, str 17]. Může se jednat o deklarativní tvrzení a učebnicová fakta (věc A implikuje B) a ne vždy musí být jasně zřejmé, zda se jedná o znalost, nebo informaci [8, str. 85].

U některých autorů zabývajících se touto problematikou existuje myšlenka toho, že explicitní znalosti a informace jsou synonyma a není mezi nimi rozdíl [7, str. 17]. Jsou to ty druhy znalostí, které je možné nalézt v podnikových intranetech nebo v jiných informačních systémech.

Tacitní znalosti (neformulované, někdy uváděné jako „nevyslovené“ nebo „tiché“) [1. str. 29] je na rozdíl od znalostí explicitní formy složité popsat slovy nebo jinak zaznamenat a sdílet a je víceméně nemožné převést je do explicitní formy. Jsou více vázány na konkrétní osobu, která je jen obtížně může předat někomu jinému. Řadí se sem typicky zkušenosti, intuice, schopnost představ a odhadů, styl chování. Jsou spíše na úrovni podvědomí. Lze tedy říci, že jsou do značné míry získávány praxí. Člověk si tyto znalosti ani nemusí uvědomovat a považuje je za něco samozřejmého. I když tedy v rámci organizace může existovat dobrá dokumentace, je potřeba myslet na to, že zaměstnanci nabývají i tacitních znalostí, které při odchodu zaměstnance odchází spolu s ním a jsou vázány jen na něj [7, str. 18].

Jednou z diskusí ohledně klasifikace znalostí dle formulovatelnosti je převoditelnost tacitních znalostí na explicitní, čehož může být do jisté míry dosaženo například učňovstvím a koučováním [9, str. 25].

Další druh znalostí se označuje jako implicitní – ty představují znalost, která sice není přímo dostupná a vyjádřená, ale je možné ji nějakým způsobem vyvodit například ze způsobu řešení úloh. Může to být do jisté míry tacitní znalost, kterou lze v případě potřeby do explicitní jednoduše převést [8, str. 85].

Přehledněji zmíněné znalosti popisuje následující tabulka:

Tabulka 1. Typy znalostí. Podle [8, str. 29]

	Typ znalostí		
	Explicitní	**Implicitní**	**Neformulované (tacitní)**
Popis	Formalizovaná nebo dokumentovaná znalost, která je většinou dobře strukturovaná a snadno přenositelná. Je převážně zpracována pomocí ICT.	Znalost, která je uložena v hlavách pracovníků, avšak je možné ji kdykoliv převést do explicitní formy.	Znalost ukrytá v hlavách jednotlivých zaměstnanců. Není lehké nebo dokonce není možné ji převést do explicitní formy a formalizovat či dokumentovat.
Příklad	Dokumenty, manuály, počítačové kódy apod.	Znalost procesu a jeho omezujících podmínek v hlavě vlastníka procesu apod.	Znalost experta v určité oblasti, získané zkušenosti atd.

2 INFORMAČNÍ SYSTÉMY V PODNIKU

Tato kapitola se zabývá problematikou informačních systémů v podniku a popisuje vybraná existující řešení.

2.1 Informační systém

Informační systémy jsou druhem počítačového softwaru, který zajišťuje shromažďování, uchovávání a distribuci informací a poskytuje různé možnosti jejich zpracování tak, aby docházelo k co možná nejefektivnější práci s těmito informacemi a daly se co nejlépe využít dle potřeb. Umožňuje lidem práci s informacemi na správném místě ve správný čas, což zejména v případech podnikových systémů vede k dosahování zisku [10, str. 32] (v případě aplikace pro interní sdílení zdrojů nemusí docházet k tak velkému plýtvání placeného času zaměstnanců na vyhledávání informací).

Díky výše popsanému dnes představují nedílné součásti v řídících procesech v mnoha různých oblastech včetně státní infrastruktury a efektivní strategie jejich využívání je klíčovou složkou vedoucí ke zvýšení předpokladů konkurenceschopnosti a poskytování kvalitních služeb ve všemožných typech organizací [11].

Jednotlivé informační systémy obvykle nejsou natolik komplexní, aby jeden dokázal uspokojit veškeré potřeby ve všech oblastech, kde může být IS vyžadován. Z toho důvodu existuje velké množství IS pro různé oblasti.

2.2 Klasifikace IS

Po čase, kdy IS vznikaly velmi individuálně pro řešení konkrétních problémů, se začaly objevovat jejich společné rysy a oblasti použití, v důsledku čehož přišly snahy o možnosti jejich bližší klasifikace [16].

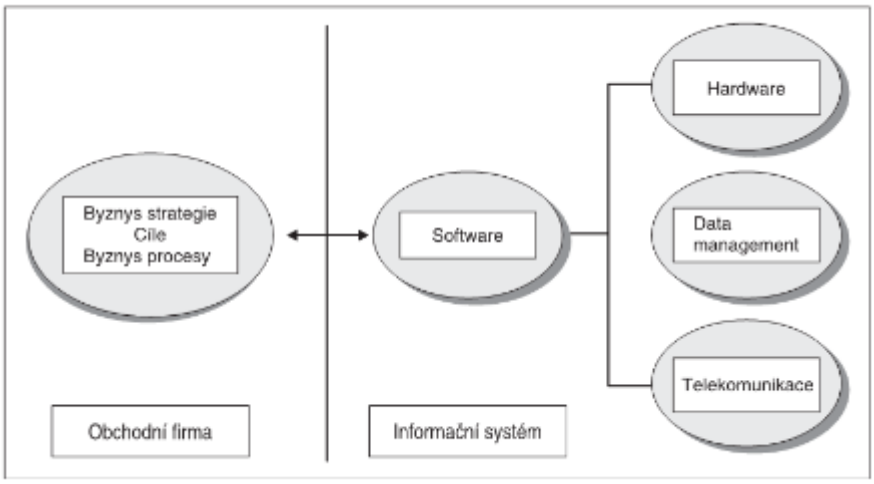
Dvě velké oblasti využití IS představují podnikové informační systémy a veřejné informační systémy.

2.2.1 Podnikové IS

Jak z názvu vyplývá, podnikové informační systémy jsou provozovány pro podporu efektivního řízení chodu organizace, přičemž přístup do těchto systémů mají obvykle pouze zaměstnanci a další lidé spojení s provozem dané organizace [12]. Obsahují různá podniková data a umožňují jejich distribuci mezi zaměstnance, případně další IS.

Pro tyto IS je typické, že vznikají za účelem zefektivnění a zautomatizování podnikových procesů a jejich funkcionalita se tedy od těchto procesů odvíjí a je s nimi spjata.

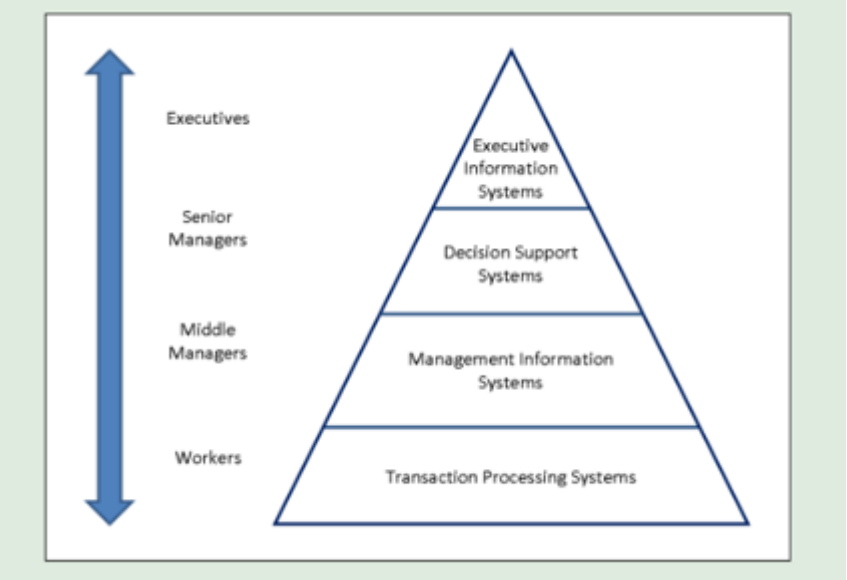
V organizacích tedy existují různé druhy informačních systémů dle toho, kterou oblast řízení pokrývají. Závislost mezi řízením podniku a informačními systémy znázorňuje tento obrázek.



Obrázek 3. Závislost mezi firmami a IS [13, str. 13]

2.2.2 Pyramida DIKW

Výše zmíněnou závislost oblastí podniku a informačních systémů blíže specifikuje pyramida DIKW, která sleduje hierarchické rozložení organizace, dle kterého podnikové IS kategorizuje. Tato pyramida může mít více úrovní, avšak nejběžnější z nich je čtyřúrovňová varianta [16].



Obrázek 4. Pyramidový model hierarchie organizace [16]

Jak je vidět na obrázku, dle této klasifikace má každá úroveň řízení podniku odpovídající informační systém navržen tak, aby plnil požadavky specifické pro danou úroveň.

1. **Transakčně procesní systémy** (TPS) nacházející se ve spodní části pyramidy jsou určeny na podporu operativní úrovně řízení. Jejich funkcionalitou je schopnost zpracování základních transakčních událostí při každodenní provozní činnosti podniku, nad kterými provádí různé operace typu validace, třídění, výpis a různé výpočty vedoucí k seznamům nebo zprávám o těchto transakcích (např. zpracování transakce objednávky zboží od pracovníka obchodního oddělení). Můžeme sem zařadit například mzdové systémy, systémy zpracování objednávek. [16 a 13, str 12].
2. **Manažerské informační systémy** (MIS) se nacházejí více na úrovní managementu, kde zpracovávají informace potřebné pro efektivní řízení organizace na základě dat z různých obchodních aktivit, a slouží zejména ke sledování krátkodobého až střednědobého horizontu chodu organizace [16].
3. **Systémy pro podporu rozhodování** (DSS) jsou používány střední až vyšší složkou managementu a díky jejich analytickým nástrojům jsou schopny na základě stávajících informací pomoci manažerům promítnout možné dopady jejich rozhodnutí i do budoucnosti. Umožňují managementu pohodlné provádění analýz, potřebných výpočtů a metod nad zadanými vstupními údaji. Očekává se, že jejich uživatel rozumí podstatám nabízených metod a ví, kdy jak vhodně jaká vstupní data použít pro řešení konkrétního problému. Mohou poskytovat souhrnné zprávy, předpovědi a grafickou formu zobrazovat různé výsledky [16 a 13 str 22].

4. **EIS systémy** na vrcholu pyramidy jsou určeny zejména pro vrcholné vedení, které bere v potaz i externí informace mimo organizaci (situace trhu, politická situace, informace o konkurenci), tudíž zajišťují i možnost přístupu k externím údajům v kombinaci s napojením k dalším informačním systémům uvnitř organizace. Z kombinací těchto údajů jsou schopny vytvořit vysoce agregovaná a strukturovaná data, případně tato data řadit do dalších souvislostí a vyhledat v nich možné trendové charakteristiky nebo nalézt klíčové ukazatele. Je pro ně typické intuitivní a jednoduché ovládací rozhraní poskytující nástroje pro přehlednou prezentaci údajů v grafech a tabulkách [13, str. 13].

2.2.3 Holisticko-procesní klasifikace

Holisticko-procesní klasifikace konkrétněji klasifikuje IS dle jejich praktických uplatnění v odpovídajících podnikových procesech.

Rozděluje jej na několik hlavních kategorií:

- **ERP** (Enterprise Resource Planning) jádrový IS navržený tak, aby sloužil pokud možno jako komplexní řešení pro řízení podniku zahrnující podporu všech fází výrobního procesu od plánování a nákup až po prodej a expedici [49][50].
- **CRM** (Customer Relationship Management) podporující efektivní řízení vztahů se zákazníky [51].
- **SCM** (Supply Chain Management) pro řízení logistických řetězců. Poskytuje dohled nad daty spojenými s produkty a materiálem tak, jak postupuje dodavatelským řetězcem od dodavatele ke spotřebiteli [51].
- **MIS** (Management Information System) pro operativní a taktické rozhodování, často sbírá data z dalších systémů, na základě kterých poskytuje informace [14, str. 77].

ERP často zahrnuje více modulů řešící potřeby v různých oblastech: účetnictví, marketing, prodej, řízení zásob, často může obsahovat všechny výše zmíněné kategorie. Mimo již uvedené se může jednat například o tyto moduly [15]:

- **APS** (Advanced Planning and Scheulding) pro pokročilé plánování a řízení dodavatelského řetězce.
- **HRM** (Human Resources Management) starající se o řízení lidských zdrojů, obvykle využívané personálním oddělením pro analýzu náborů nebo školení.

- **EAM** (Enterprise Asset Management) pro plánování a řízení podnikové údržby a servisu a správu podnikového majetku.
- **DMS** (Document Management System) sloužící jako nástroj pro efektivní správu a oběh dokumentů [31].
- **BPM** (Business Process Management)
- **BI** (Business Inteligence) pro statistické a analytické výpočty a propojení různých datových zdrojů.

2.3 Firemní intranet

Firemní intranet představuje interní (soukromý) web společnosti, který není dostupný veřejnosti a jehož cílem je usnadnění výměny sdílení informací nebo zdrojů zaměstnancům a slouží tak jako obecný zdroj informací firmy [18].

2.3.1 Oblasti využití intranetu

Intranety mohou mít různou složitost a funkcionalitu v závislosti na potřebách konkrétní organizace. Zatímco jednodušší intranety mohou sloužit pouze k prezentaci novinek, sdílení souborů, složitější z nich mohou nabízet mnoho dalších pokročilejších nástrojů pro zlepšení interní komunikace a sdílení dat a může se tak stát velmi komplexní platformou [18].

Může se jednat o:

- Správce souborů a firemních dokumentů
- Databáze školících materiálů, směrnic
- Vnitrofiremní wikipedie
- Rezervační systém (pro místnosti, firemní vozidla, parkovací místa)
- Diskusní fórum pro zaměstnance
- Správa firemních procesů
- Prostor pro ankety
- Fulltextové vyhledávání v obsažených datech

[19]

2.3.2 Výhody využívání intranetu

Intranet může mít pozitivní vliv na vnitrofiremní procesy:

- Efektivnější komunikace – diskuse ve vláknech pod oznámeními a články

- Jednodušší organizace projektů – lze dohledat informace o tom, kdo jaké problémy řešil, využít případné existující řešení, sledovat termíny, každý může vědět, kdo na čem pracuje a jaké mají jeho úkoly prioritu, na koho se v případě konkrétních projektů obrátit
- Snazší sdílení a archivace dokumentů – ke sdíleným souborům na intranetu může přistupovat každý, kdo má příslušná přístupová práva, i když zrovna pracuje z domova. Tímto způsobem se ušetří velká část komunikace. Intranet také zjednoduší předávání aktualizovaných verzí dokumentů příslušným pracovníkům a aktualizaci dokumentů.
- Zvýšená produktivita – pracovníci mohou snáze předávat své znalosti a zkušenosti ostatním, což může zvýšit produktivitu. Také mají přístup ke školicím materiálům, osvědčeným postupům a dalším zdrojům, které mohou pomoci při rozvoji jejich znalostí a schopností.
- Snazší delegace práce – pokud jsou všechny postupy a procesy k dohledání na intranetu a všechny informace dané tématiky jsou na intranetu sdílené, je snazší někoho na tyto informace odkázat a každý si je v případě potřeby může sám jednoduše vyhledat.
- Podpora firemní kultury – pokud jsou zaměstnanci informováni o dění ve firmě a mají možnost na něj zveřejnit svůj názor, mohou se cítit více angažovaní a oceňovaní.

[19]

3 EXISTUJÍCÍ ŘEŠENÍ

Následující kapitola uvádí a popisuje některé z dostupných existujících řešení pro sdílení zdrojů a spolupráci. Je potřeba zmínit, že vzhledem k velmi široké škále možností, které tato řešení mohou poskytovat (některé příručky dosahují až tisíce stran), je velmi složité obsáhnout všechny tyto funkcionality v rámci obdobné práce. Vybrány tedy byly jen klíčové a nejvíce diskutované prvky těchto řešení pro poskytnutí obecného přehledu o nich.

Tato řešení mohou být více či méně robustní. Zatímco některá mohou poskytovat prostor pouze pro jednoduché ukládání a sdílení dokumentů, jiná podporují i další oblasti a nástroje pro správu téměř jakéhokoliv obsahu a je na každém, aby vyhodnotil, které řešení je pro jeho potřeby nejoptimálnější. Zmíněny jsou systémy velmi robustní i ty menší, které ale mohou svou jednoduchostí poskytovat praktičtější možnost.

3.1 Microsoft Sharepoint

SharePoint je velmi populární nástroj pro sdílení zdrojů ve formě webové aplikace od společnosti Microsoft, která od svého vydání v roce 2001 umožňuje podnikům ukládat a spravovat jakýkoli typ materiálů a dat (písemné materiály, multimediální materiály, zprávy, odkazy, seznamy informací, webové stránky a úkoly, ...). Můžeme jej považovat za centralizované místo pro správu téměř jakéhokoli obsahu, který máme [33]. Cena základní verze začíná na $5.00 za měsíc [65].

Nabízí spoustu možností, nástrojů a funkcionalit pro správu obsahu:

- Ukládání a správa dokumentů
- Správa informací, které mohou být ukládány v seznamech (obdoba tabulek) nebo pomocí článků, které lze zveřejnit konkrétním uživatelům nebo skupinám uživatelů.
- Organizace týmů pomocí kalendáře (plánování schůzek, porad, ...) nebo vytváření úkolů přes nástroj úkoly, který tyto úkoly umožňuje přiřadit konkrétním uživatelům, sledovat jejich průběh, stav, plnění.
- Řízení přístupu je realizováno systémem pro nastavování přístupových práv uživatelům ke konkrétním zdrojům. Tato práva mohou povolovat akce od prohlížení obsahu přes jeho editaci až po plnou kontrolu nad správou webu.
- Vyhledávací a integrační funkce umožňující vyhledávání napříč všemi součástmi sharepointu včetně filtrování, vytváření vlastních pohledů pro zobrazování obsahu, agregaci dat aj.

- Sociální funkce v podobě diskusí, anket, mikroblogů nebo propojení s aplikací Teams či Yammer (sociální síť pro firemní komunikaci).
- Integrace s dalšími nástroji Office 365 zvyšující flexibilitu Sharepointu pomocí propojení s aplikacemi Outlook, OneNote, Yammer, Teams. V rámci sharepointu je poté možné zobrazovat informace z těchto aplikací (například obsah z poznámkových bloků OneNote, synchronizaci kalendářů a úkolů s Outlookem).
- Rozšiřitelnost v podobě customizací a možnosti využití aplikací třetích stran dostupných na obchodu SharePointu. Organizace si může vytvořit své vlastní aplikace a ty v rámci sharepointu využívat [34].
- Zavádění automatizovaných procesů nad položkami a dokumenty umožňuje vyvolávat akce v důsledku detekovaných změn v podobě upozornění nebo schvalovacích procesů.

[33]

3.1.1 Hierarchie a základní stavební bloky

Sharepoint Sites jsou hlavní konstrukcí pro ukládání a sdílení veškerých zdrojů, podporu komunikace a spolupráce zaměstnanců. Sites obsahují seznamy, knihovny, workflow, aplikace (apps) a další weby, resp. webové části, kde je díky kombinaci nabízených položek a předpřipravených šablon umožněno vytvořit uživatelsky přívětivou a přizpůsobivou webovou stránku, aniž by uživatel musel mít jakoukoliv znalost programovacích jazyků. Proces vytváření tohoto webu začíná právě výběrem účelu webu a vzhledové šablony různých typů pro různé účely. Po zvolení šablony a vyplnění dodatečných informací, jako je název webu, je uživateli umožněno obsah webu různě modifikovat a snadno přidávat různé položky a další části webu tak, aby web co nejvíce vyhovoval konkrétním požadavkům uživatele a organizace [35, 36].

Mezi typické položky umístitelné na takto vytvořený web patří například:

- **Seznamy** (obdoba tabulek) přestavující možnost pro uchovávání a sledování strukturovaných dat. Sharepoint poskytuje řadu předdefinovaných seznamů a možnost vytvářet vlastní seznamy. Tyto seznamy mohou mít další rozšiřující funkcionality podporující jejich základní funkce. Například kontaktní seznamy mají možnost integrace s Outlookem, čímž umožňují přímo z Outlooku synchronizovat kontakty se

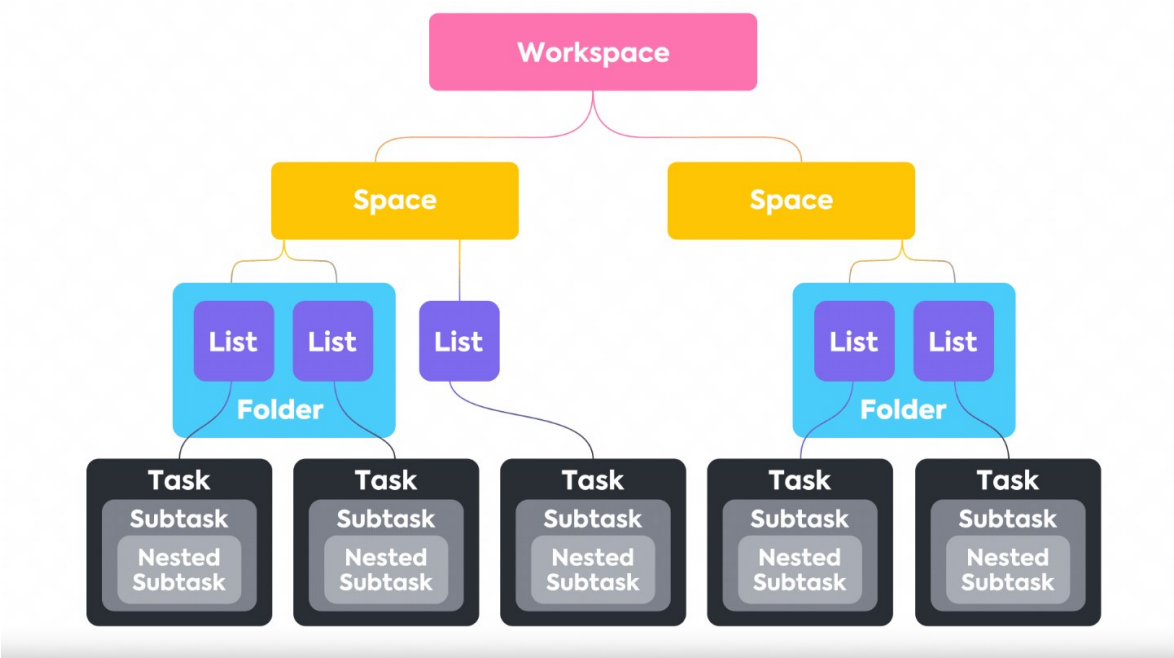
sharepointem nebo je upravovat. Seznamy je také možné upravovat a přizpůsobit vlastním potřebám [36, str. 215].

- **Knihovny dokumentů** slouží jako místo pro ukládání všech typů souborů (Microsoft Office soubory, PDF soubory, ...) spojených s konkrétním projektem nebo klientem, které je zde možné jednoduše upravovat a sdílet. Soubory lze i jednoduše přesouvat a přidávat pomocí drag and drop (přetáhnutí z jednoho místa na druhé). O souborech jsou kromě názvu uchovávána další metadata obsahující informace o tom, kdo soubor upravoval, kdo jej vytvořil apod. Tato metadata je možné využít k uspořádání a nalezení souborů. S knihovnou dokumentů jsou spjaty další funkcionality jako:
    - Řízení přístupu ke knihovně nebo jednotlivým souborům v ní.
    - Na základě sledování aktivit u souborů vyvolat upozornění v případě, že u souboru nastanou nějaké změny.
    - Přidat odkazy na položky nebo soubory uložené mimo knihovnu, ať už jde o odkazy na externí zdroj, nebo jen na zdroj umístěný v jiné knihovně.
    - Možnost zvýraznění položek v knihovně.
    - Vytvoření více pohledů knihovny, mezi kterými lze přepínat (každý pohled může mít jiné uspořádání, lze vybrat, jaká data budou v sloupci u souboru zobrazena) [36, str. 261] [37].
- **Web Parts** představující jednotlivé webové části na webové stránce.
- **Workflows**, díky kterým může organizace definovat a automatizovat různé pracovní postupy a procesy a sledovat jejich postup (například schvalování dokumentů různými osobami).
- **Apps** jsou samostatné aplikace, které je možné integrovat jako část webu sharepointu a tím ho rozšířit o novou funkcionalitu, kterou běžně sharepoint nenabízí. Jak již bylo zmíněno, je možné je získat z obchodu sharepointu, nebo si vytvořit vlastní (typicky za použití HTML, CSS, JavaScriptu). Pro komunikaci a přístup k datům a funkcím SharePointu tyto aplikace mohou využívat tzv. Sharepoint API, které umožňuje čtení, zápis a aktualizaci dat v SharePointu. Apps tedy klíčově rozšiřují již tak velké možnosti customizace SharePointu [38].

3.2 ClickUp

ClickUp je stejně jako SharePoint jedna z platforem pro sdílení souborů. Jedná se o všestrannější nástroj pro nejen sdílení souborů, ale i pro podporu projektového managementu a spolupráci.

Umožňuje vytvářet projekty, které mohou být rozděleny do hierarchie na obrázku, což zlepšuje organizaci.



Obrázek 5. Struktura ClickUp [39]

- Workspace je hlavním stavebním blokem a kontejnerem celé hierarchie, do které spadají všechny další zdroje týkající se organizace. Pro jednu organizaci je doporučeno mít pouze jeden workspace, uživatelé mohou mít přístup do více workspaců.
- Druhou největší jednotkou v hierarchii jsou Spaces, které slouží pro vysokoúrovňovou kategorizaci. Space typicky může reprezentovat konkrétní oddělení nebo tým. Každému Space je možno nastavit specifická přístupová práva pro dané uživatele, nebo ho v daném workspace ponechat zcela veřejný.
- Folders (složky) nejsou povinné, nicméně mohou pomoci při organizaci projektů. Lze do nich členit seznamy a nahrávat soubory.
- Seznamy v sobě seskupují úkoly s podobným cílem. Lze je přidat přímo do Spaces nebo do zmíněných složek.

- Úkoly slouží ke sledování plnění konkrétních postupů. Stejně jako seznamům jim lze přiřadit přílohu ve formě souborů.

[47]

Díky této hierarchii je možné nahrávat soubory na různé úrovně, což je umožňuje organizovat a sdílet v souladu s pracovní strukturou. Soubory jsou tak k dispozici na správném místě [47].

Základní verzi lze využívat zdarma se 100 MB úložištěm, v případě potřeby lze zakoupit [64].

3.3 Cloudová úložiště DropBox, OneDrive, Google disk

DropBox, OneDrive a Google disk ve své podstatě poskytují velmi obdobnou funkcionalitu. Slouží jako cloudové platformy speciálně pro ukládání, sdílení souborů a složek mezi uživateli a zařízeními a nabízí různé možnosti práce s nimi. Typicky je k nim možné přistupovat přes webový prohlížeč nebo aplikace pro Windows, Mac, Android nebo iOS a je potřeba provést registraci.

Na rozdíl od komplexnějších řešení typu SharePoint (kde je k nalezení široká škála možností), uživatelské rozhraní těchto úložišť v podstatě z většiny představuje hierarchickou strukturu souborového systému s lištou pro akce se soubory (nahrání, stažení, sdílení souborů) a vyhledávacím panelem. Lze zobrazovat detaily souborů.

Sdílení souborů je poměrně rychlé a jednoduché. Tyto platformy umožňují vytvořit odkazy, pomocí kterých lze sdílet téměř cokoliv, ať už jde o fotografie, videa, zazipované soubory nebo velké CAD soubory. Tyto soubory lze nasdílet i osobám, které nemají uživatelský účet. Je umožněno nastavovat oprávnění pro konkrétní uživatele (možnosti zobrazení, úprav nebo stažení souboru) [63].

Maximální možná velikost nahrávaných souborů se může lišit. Například u Dropboxu jde o limit 2TB v rámci desktopových a mobilních aplikací, soubory nahrávané přes web se musí vejít do limitu 50 GB. U nahrávání souborů přes API rozhraní je limit 350 GB. OneDrive web umožňuje nahrávat soubory do velikosti 250 GB, u ZIP souborů je limit 20 GB. Google Drive umožňuje nahrát nebo synchronizovat soubory až do velikosti 5 TB. Pokud sdílený soubor někdo upraví, změny jsou okamžitě dostupné všem uživatelům včetně informací o tom, kdo soubor změnil [42].

Tato úložiště dále podporují správu verzí a nabízí možnost automatické synchronizace souborů uložených v zařízení s cloudovým úložištěm dropboxu. V případě nechtěného odstranění souborů z těchto úložišť je možné je kdykoliv do určitého počtu dnů obnovit, stejně tak je možné obnovovat předchozí verze souborů (počet dnů pro obnovu a maximální možný počet historických verzí se může v závislosti na typu zakoupeného plánu lišit, typicky jde o 30-180 dní nebo 200-500 verzí souborů) [43].

Všechny tři zmíněné služby jsou v principu placené, nicméně všem registrovaným uživatelům nabízí určitý počet GB úložiště zdarma.

Dropbox nabízí registrovaným uživatelům 2 GB zdarma a k dispozici je hned několik typů předplatných pohybujících se mezi 2 TB až 5 TB místa v cenách okolo $9.99 do $19.99 měsíčně v závislosti na výběru roční nebo měsíční platby a počtu uživatelů. Nabízeny jsou také individuální plány, jejichž cena a velikost úložiště záleží na dohodě [44].

Stejně tak cena OneDrive záleží na zvoleném tarifu a doby předplatného (rok, měsíc). Microsoft se do něj často snaží zakomponovat i své další Office 365 produkty. Firma Microsoft nabízí na své úvodní stránce tři základní balíčky pro běžné uživatele. Balíček *365 Basic* vydává s cenou od $1.99 měsíčně zahrnující 100 GB úložiště a jako bonus Outlook. Dražší balíček *365 Personal* má k dispozici 1 TB úložiště zahrnující hned několik „bonusových“ položek (Word, Excel, PowerPoint...). Nejdražší z těchto tří balíčků je *Family* naceněný na $9.99, obsahuje 6 TB úložiště (1 TB na osobu, na rozdíl od předchozích je určen až pro šest osob) a další bonusové položky z předchozího balíčku *Personal* [45].

Google Disk poskytuje základní tarif pro všechny účty zdarma o velikosti 15 GB, tarif *Basic* nabízí 100 GB úložiště za cenu $1.99 měsíčně, Standard s velikostí 200 GB za $2.99 a Premium s 2 TB úložiště za $9.99 [62].

4 EXISTUJÍCÍ ŘEŠENÍ VS VLASTNÍ

Při pořizování IS se nabízí možnost nákupu již existujícího řešení, nebo vývoj zcela vlastního. Oba přístupy mají přirozeně své výhody a nevýhody. Hlavní nevýhodou vlastního vývoje IS je časová a finanční náročnost. Firmy specializované na vývoj takových IS mohou do jejich vývoje investovat ročně až desítky milionů dolarů. Pokrýt všechny možnosti takových IS je tedy jistě velmi náročné a vlastní řešení může být již v okamžiku začátku jeho provozu zastaralé. Na druhou stranu výhodou vlastního řešení může být pokrytí specifických požadavků dané společnosti na míru, které existující řešení nemusí poskytovat vůbec nebo takové existující řešení pro společnost není ideální [13, str 36].

Další otázkou je samotný provoz IS (servery a další hardwarová infrastruktura, monitoring, údržby a aktualizace softwaru, bezpečnost..). Provoz může společnost opět řešit svépomocí nebo outsourcingem. Při outsourcingu služby spojené s provozem IS zajišťuje dodavatelská společnost, která spravuje veškeré zdroje pro takový provoz, jaký organizace očekává.

4.1 Klady a zápory jednotlivých řešení

Konkrétní klady a zápory vývoje vlastního systému jsou zobrazeny v této tabulce:

Tabulka 2. Klady a zápory vlastního vývoje. Podle [13, str. 36]

Vlastní vývoj	
+ Klady	**- Zápory**
• IS šitý na míru potřebám firmy • Možnost růstu IS dle potřeb firmy • Detailní znalost provozovaného IS je přímo ve firmě • Konkurence nezná silné a slabé stránky IS firmy • Dodavatel neodhalí strategii firmy • Snadná reakce na potřeby uživatelů	• Vysoké náklady • Časová náročnost • Obvykle nižší kvalita IS zapříčiněná ne vždy špičkovou kvalitou interních řešitelů • Značné riziko nekonzistence systému při fluktuaci řešitelů • Kooperativní náročnost (nebudování vztahů se subdodavateli)

Klady a zápory vývoje firemního informačního systému softwarovou firmou:

Tabulka 3. Klady a zápory externího vývoje. Podle [13, str 36]

Vývoj externí softwarovou firmou	
+ Klady	**- Zápory**
• IS šitý na míru potřebám firmy • Konkurence nezná silné a slabé stránky • Optimálně využity znalosti interních a externích specialistů	• Vysoké náklady (obvykle ještě vyšší než při vlastním vývoji) • Časová náročnost • Riziko přenosu vnitřních informací mimo firmu

Klady a zápory vytvoření firemního informačního systému prostřednictvím nákupu aplikací od různých výrobců:

Tabulka 4. Klady a zápory hotového řešení. Podle [13, str. 36]

Nákup hotových řešení	
+ Klady	**- Zápory**
• Rychlá realizace • Nejnižší náklady • Lze vybrat osvědčená řešení pro každou část IS	• Obtížná integrace různých aplikací do jednoho IS • Obtíže údržby vazeb mezi aplikacemi a tím relativně nízká stabilita IS

Klady a zápory nákupu informačních systémů od generálního dodavatele:

Tabulka 5. Klady a zápory IS od gen. dodavatele, Podle [13, str 36]

Nákup IS od generálního dodavatele – systémového integrátora	
+ Klady	**- Zápory**
• Nejrychlejší realizace • Nízké náklady • Profesionální řešení každé komponenty i celého IS • Osvědčená řešení pro každou část IS • Integrace komponent je garantována dodavatelem	• Vysoká závislost na dodavateli a jeho schopnostech, serióznosti a stabilitě • Riziko přenosu vnitřních informací mimo firmu

Klady a zápory outsourcingu provozu komplexního IS:

Tabulka 6. Klady a zápory outsourcingu provozu IS, Podle [13, str. 36]

Outsourcing provozu komplexního IS	
+ Klady	**- Zápory**
• Možnost soustředění se na hlavní předmět činnosti (využití firemních aktiv v oblastech největšího zhodnocení) • Firma se nemusí zabývat technologickými aspekty, kterými bude dosaženo požadovaného cílového stavu • Možnost vyřešení finančního zabezpečení vývoje, provozu a údržby IS • Možnost změny odebíraného rozsahu služeb podle potřeb	• Dlouhodobost a nevratnost důsledku tohoto rozhodnutí • Úplná závislost na outsourcingovém partnerovi • Riziko přenosu vnitřních informací mimo firmu (větší než při nákupu od generálního dodavatele) • Vysoké náklady (většinou jsou outsourcovány nestandardní aplikace)

5 POUŽITÉ TECHNOLOGIE

Cílem této kapitoly je představit technologie a nástroje (programovací jazyk, frameworky, vývojové prostředí, ...), které byly použity pro implementaci aplikace, případně uvést menší srovnání s dalšími používanými technologiemi.

5.1 C# a .NET

Aplikace byla vytvořena v jazyce C#. Jde o moderní, objektově orientovaný, typově bezpečný programovací jazyk vyvíjený společností Microsoft. Vývojářům slouží pro vytváření mnoha typů aplikací.

Může se jednat o:

- Webové aplikace
- Mobilní aplikace
- Desktopové aplikace
- Microservices
- Cloud
- Machine learning
- Vývoj her
- Internet věcí

[22]

C# syntaxí vychází z rodiny jazyků C (tzv. C-like jazyk), je velmi podobný jazyku Java, nicméně jejich odlišnost lze zpozorovat například v podpoře vlastností (properties), indexerů nebo událostí, které v Javě nejsou dostupné. C# také nabízí různé funkcionality usnadňující práci s asynchronními operacemi v podobě klíčových slov „await“ a „async“ [52].

Jak již bylo zmíněno, vývojáři mohou používat C# pro různé účely. Například pro vývoj desktopových aplikací s grafickým uživatelským rozhraním je k dispozici technologie WPF, pro vývoj webových aplikací ASP.NET, pro mobilní aplikace je k dispozici platforma Xamarin. C# je také možno využít pro vývoj her pomocí herního enginu Unity, díky čemuž se stal populárním i mezi vývojáři her, ačkoliv pro tyto potřeby svými vlastnostmi (zejména při vývoji náročnějších her) vyhovuje spíše jazyk C++ [22].

S .NET 6.0 byl představen .NET MAUI, který představuje multiplatformní architekturu pro vytváření nativních mobilních a desktopových aplikací za použití C# a XAML. Díky .NET

MAUI je tedy možné mít z jediného sdíleného základu kódu spustitelné aplikace pro Android, iOS, MacOS a Windows [21].

5.2 .NET

.NET je platforma vyvinutá společností Microsoft, která poskytuje rámec pro vývoj, běhové prostředí a virtuální stroj umožňující spouštění aplikací založených na této platformě. Kromě C# je možné pro vývoj na platformě .NET využít jazyk F# a Visual Basic.

Virtuální stroj platformy .NET zvaný Common Language Runtime (CLR) slouží k interpretaci a běhu kódu napsaného v jazycích .NET a zároveň poskytuje další služby jako například správa paměti, řízení výjimek, dynamické linkování a další.

Díky CLR je možné, aby aplikace napsané v jazycích .NET byly platformně nezávislé, díky čemuž jsou spustitelné na různých operačních systémech a architekturách procesorů.

Jednou z hlavních důležitých částí .NET je Garbage Collector, který funguje jako automatický proces starající se o správu paměti (tzv. managed kód) .NET aplikací. Princip funkcionality spočívá v tom, že automaticky uvolňuje paměť, která již v aplikaci není používána, čímž pomáhá zamezit chybám vedoucím ke zbytečnému nadměrnému využívání paměti nebo zbytečnému obsazení celé paměti, které může vést k pádu programu a dalším problémům, díky čemuž přispívá ke stabilitě aplikace. Tyto problémy jsou známy pod pojmem „memory leak“ a dochází k nim při nesprávném využívání dynamické alokace paměti nebo nesprávném uvolňování v průběhu běhu programu. Programátoři díky této funkcionalitě nemusí manuálně uvolňovat dynamicky alokovanou paměť (jako například ve více nízkoúrovňových jazycích typu C/C++) [53].

Strukturu platformy .NET blíže popisuje následující obrázek:

Obrázek 6. Struktura .NET [23]

5.3 ASP .NET CORE

ASP.NET je skupina open source multiplatformních (Windows, Linux, MacOS) webových frameworků vyvinutých společností Microsoft sloužících pro tvorbu webových aplikací a služeb za použití C#, HTML, CSS, JavaScriptu. Poskytuje sadu funkcionalit a knihoven pro efektivní a pohodlné vytváření těchto aplikací. Přináší řešení pro často opakující se problematiku při vývoji v této oblasti, jako je například:

- **Model binding** sloužící k mapování hodnot z HTTP požadavků na parametry metod v kontrolerech.
- **Autentizace a autorizace,** díky čemuž je možno jednoduše řídit přístup k určitým částem aplikace (Windows Authentication, Forms Authentication, OAuth a další).
- **Routování** pro definování URL cest, které se používají pro navigaci v aplikaci a volání metod v kontrolerech.
- **Poskytování různých způsobů pro práci s databázovými systémy** jako například Entity Framework (pro ORM) nebo ADO. NET.
- **Session management** pro ukládání a správu dat specifických pro každého uživatele, který může být užitečný pro udržování stavu aplikace při opakovaných interakcích uživatele.

- **Cachování** umožňující odeslání dříve připravených výsledků a dat, která se nezměnila a není potřeba je znovu kompletně zpracovávat, což může zrychlit odezvu aplikace.
- **Podpora pro vytváření různých typů odpovědí** ve formě HTML stránek, JSON, XML a dalších.

[54]

Pro vytváření webových aplikací nabízí několik frameworků, přičemž každý z nich cílí na jiný druh stylu vývoje a samotná volba přirozeně záleží na požadavcích konkrétní aplikace.

- **ASP.NET MVC** (Model-View-Controller) pro vývoj webových aplikací za podpory architektury MVC, která pomáhá oddělovat logiku aplikace od prezentační vrstvy, díky čemuž se snáze dosahuje čistějšího a udržovatelnějšího kódu.
- **ASP.NET Web API** který, jak název vypovídá, slouží pro vytváření REST API pro komunikaci mezi různými částmi aplikace.
- **ASP.NET Web Pages**
- **ASP.NET Web Forms**

5.4 Framework obecně

Framework představuje předpřipravenou funkcionalitu (kód), která vývojářům usnadňuje a urychluje vývoj softwaru tím, že poskytuje základní strukturu pro řešení často se opakující problematiky, kterou programátor může využít pro své řešení. Není tedy potřeba znovu vynalézat něco, co už bylo vynalezeno. [24].

5.5 Entity Framework

Entity Framework je open-source framework vyvinutý společností Microsoft, jehož hlavní princip spočívá v poskytování určité abstrakce pro práci s databází v rámci tzv. objektově-relačního mapování (ORM), čímž zjednodušuje získávání, ukládání a manipulaci s uloženými záznamy a dále poskytuje možnosti například pro validaci dat, podporu transakcí nebo cachování paměti [55].

Ve vytvořené aplikaci je použit výhradně přístup Code First.

5.5.1 ORM

Přístupy objektově orientovaného programování a relačních databází mají odlišnou filozofii pro práci s daty, což může vytvářet problémy při vytváření objektů v programovacím jazyce z dat z relační databáze a naopak. Hlavním cílem ORM je tedy zjednodušit práci s databází tím, že objekty z programovacího jazyka převádí do takové podoby, která vyhovuje konceptům relačních databází, díky čemuž mohou programátoři s databázovými daty pracovat pomocí samotných objektů a tříd v programovacím jazyce, namísto aby tento převod museli zajišťovat sami, což v podstatě odstraňuje potřebu psát SQL kód [56].

5.5.2 Přístupy

Entity Framework nabízí několik přístupů pro práci s databází:

- **Database First** přístup umožňuje z již vytvořeného databázového schématu vygenerovat doménové třídy (modely) a kontext. Tento přístup je vhodné využít v případě, kdy máme od databázových architektů hotové komplexní databáze, které chceme využít.
- **Code First** nabízí možnost vygenerování databáze na základě existujících tříd (modelů) v programovacím jazyce. Tabulky v databázi poté odpovídají vlastnostem těchto modelů a jejich dodatečným atributům. Toto řešení přináší výhodu v tom, že není třeba řešit návrh a manuální vytváření databáze. Každá změna v namapovaném modelu je (na základě jednoduchých příkazů) automaticky promítnuta do databáze.
- **Model First** umožňuje vytvořit databázová schémata a potřebné modely pomocí grafického rozhraní Entity Framework Designer.

[57]

5.5.3 DbContext

Třída DbContext je klíčová součást Entity Frameworku, která představuje primární abstraktní vrstvu umožňující komunikaci s databází a zajišťuje tak most pro mapování mezi objekty z aplikace a tabulkami v databázi.

Dále zodpovídá za tyto činnosti:

- Dotazování – převádí LINQ dotazy na dotazy SQL, které následně spustí nad databází.
- Sleduje změny, ke kterým došlo na entitách mapovaných do databáze.

- Provádí operace vložení, aktualizaci a mazání na základě stavu objektů (entit).
- Cachování
- Spravuje vztahy mezi entitami.
- Převádí nezpracovaná data z databáze na objekty.

[26]

5.5.4 Možnosti načítání dat

Entity Framework v rámci podpory efektivity dotazování na databázi nabízí tři způsoby načítání dat z databáze. Jedná se o to, jakým způsobem jsou načítány entity navzájem provázané vztahy.

Tyto typy jsou:

- Eager Loading
- Lazy Loading
- Explicit Loading

Eager Loading je technika, která umožňuje načíst související entity v rámci jednoho dotazu, takže nemusíme provádět další dotazy pro načtení těchto souvisejících entit. Snižováním počtu dotazů na databázi zlepšujeme výkon aplikace. K dispozici je metoda *Include()*, pomocí které zvolíme, které související entity se mají z databáze v rámci daného dotazu získat. Bez specifikování, které související entity mají být načteny, nebudou k dispozici žádné [66].

Lazy loading automaticky zajišťuje načtení souvisejících entit z databáze až v době, kdy je k nim přistupováno. Ve výsledku to znamená, že jsou data automaticky načítána pouze v situacích, kdy jsou potřeba. Načítání bude funkční automaticky po označení vazebního atributu klíčovým slovem *virtual.* Je možné jej explicitně vypnout pouze pro vybrané případy (vazební atributy nebudou označeny jako *virtual* a nebo pro celý DbContext v jeho konstruktoru za pomocí *this.Configuration.LazyLoadingEnabled = false*;) [27].

Explicit loading je způsob načítání dat, pomocí kterého je v podstatě možné související entity načíst pomocí lazy loadingu, ačkoliv je zakázaný. Je vhodný v situacích, kdy chceme načíst související entity pouze v případě, že jsou skutečně potřebné bez nutnosti načítat všechny související entity při načítání hlavní entity. Jsou použity dva oddělené dotazy [67].

5.6 Identity Framework

ASP.NET Identity Framework poskytuje bezpečnou robustní funkcionalitu autentizace a autorizace uživatelů, kterou lze v aplikaci snadno implementovat bez nutnosti psaní vlastních řešení.

Poskytuje řešení například pro:

- Přihlašování, registraci a správu uživatelských účtů (změna hesla, obnova hesla, potvrzení registrace na emailu, ...), ať už pomocí uživatelského jména a hesla, emailové adresy a hesla, nebo sociálních sítí a dalších služeb pro externí autentizaci (účet Microsoft, Google, Facebook, Twitter, ...). Poskytuje možnost zapamatování přihlášení.
- Podpora pro dvoufaktorovou autentizaci, hashování hesel.
- Správa rolí a oprávnění – je možno vytvářet role a ty přiřadit konkrétním uživatelům, což usnadňuje přidělování přístupových práv ke konkrétním částem aplikace, ať už jednotlivým uživatelům, nebo uživatelům patřícím do konkrétních skupin.

5.7 ASP.NET MVC

ASP.NET MVC je jedním z populárních MVC frameworků určených pro tvorbu webových aplikací vyvinutý společností Microsoft. Jak bylo zmíněno výše, jeho hlavní princip stojí na podpoře MVC architektury, která je společně s životním cyklem požadavku na ASP.NET MVC aplikaci popsána v následujících podkapitolách.

5.7.1 MVC

MVC (Model-View-Controller) představuje architektonický vzor, který se stejně jako ostatní architektonické vzory snaží rozložit části aplikace na různé části tak, aby docházelo k efektivnějšímu vývoji dané aplikace a nedocházelo k tzv „špagetovému kódu“, při kterém mohou být logické operace, dotazování na databázi a vykreslování výstupu různě nepřehledně rozházeny obvykle v jednom souboru dohromady [24].

Princip, kterým se MVC snaží tento problém vyřešit, spočívá v rozdělení aplikace na tři hlavní části, kterými jsou Modely, Views (pohledy) a Controllery, přičemž každá plní funkci v jiné oblasti.

Model reprezentuje a uchovává data (například data z databáze), s kterými aplikace pracuje, navíc obsahuje business logiku. Jedná se v podstatě o „obyčejné“ třídy jazyka s různými

atributy a metodami (pro výpočty, validaci...), přičemž modelu nezáleží na tom, jak se k němu data dostala, jak budou dále zformátována a kde se budou vypisovat, jelikož výstup nikam neodesílá, pouze zapouzdří klíčovou funkcionalitu [24].

View (pohled) zajišťuje samotné vygenerování výstupu uživateli – v případě webové aplikace vygeneruje strukturu HTML dokumentu. View od controlleru přijme data, která bude pro vykreslení dané HTML stránky potřebovat (například list položek nákupního košíku). ASP.NET obsahuje tzv. razor engine, který ve view za pomocí specifické syntaxe dovoluje používat C# kód pro vytváření HTML. V praxi to může znamenat, že view obsahuje cyklus, který prochází všechny položky v listu, který byl do view předaný z controlleru, a v každé iteraci generuje řádek HTML tabulky s odpovídajícím textem iterované položky. View neřeší, odkud data přišla, zajímá jej pouze vygenerování výstupu [24].

Controller představuje hlavní řídící část architektury MVC, která zajišťuje komunikaci mezi view a modelem. Každý uživatelský (v případě webové aplikace HTTP) požadavek je pomocí routování předán controlleru, resp. jeho konkrétní akci (metodě), která za pomocí funkcionality příslušných modelů požadavek zpracuje a výsledek předá pohledu.

Mimo již zmíněné přináší MVC přístup následující výhody:

- U větších aplikací zůstává kód díky oddělení logiky od prezentační vrstvy přehlednější a lépe udržovatelný.
- Oddělení na části může zjednodušit kooperaci členů týmu, případně lépe uplatnit jejich specializaci.
- Každá část MVC může být vyvíjena nezávisle na ostatních.
- Poskytuje prostor pro logické seskupení souvisejících akcí do oddělených částí aplikace.
- Díky oddělení na části může být snazší a rychlejší doplnit nebo upravit funkcionalitu.
- Snazší testování, kdy každá část MVC může být testována nezávisle na ostatních.
- Úpravu jednotlivých částí lze provést izolovaně bez zásahu do ostatních komponent.

[24]

5.7.2 Životní cyklus požadavku

Životní cyklus požadavku začíná ve chvíli, kdy uživatel do prohlížeče zadá adresu webu s parametry. Tento požadavek poté dojde k aplikaci, kde:

- Je požadavek zachycen routerem, který požadavek dle jeho parametrů namapuje na konkrétní akci controlleru.
- Controller reaguje na přijaté parametry – tzn. zavolá metody patřičného modelu, kterým předá odpovídající data a model tato data zpracuje (validace, vyhledání v databázi).
- Controller informuje pohled a předá mu zpracovaná data z modelu.
- View na základě dat předaných z controlleru vytvoří šablonu pro zobrazení těchto dat.
- Hotová stránka je odeslána uživateli.
- Uživatel interaguje s přijatým pohledem, pomocí kterého vytváří další požadavky.

[25]

5.8 Relační databáze

Entity Framework pracuje s relačními databázemi. Ty jsou druhem databáze, který funguje na principu tabulek, ve kterých jsou dané záznamy uložené, přičemž mezi těmito tabulkami mohou být definované různé vztahy pro popis souvisejících entit [58].

V praxi sloupce tabulky popisují vlastnosti dané entity (id, jméno, příjmení, cizí klíč) a řádek odpovídá konkrétním datům (záznamu) pro každý sloupec. Každý sloupec uchovává data pouze v určitém datovém typu, může se jednat například o číslo, text, datum. Typicky je každému záznamu přidělen tzv. primární klíč (ID), který záznam jednoznačně odlišuje od ostatních [58].

Vztahy mezi tabulkami, resp. jednotlivými záznamy v nich jsou popsány pomocí tzv. cizích klíčů. Cizí klíč u záznamu v tabulce reprezentuje primární klíč záznamu v jiné tabulce. Díky tomu můžeme popsat, které záznamy spolu souvisí. Příkladem může být tabulka reprezentující seznam automobilů, přičemž každý automobil může mít svého majitele. Pro majitele bude existovat další tabulka. Abychom vyjádřili vztah mezi automobilem a jeho majitelem, bude mít automobil cizí klíč reprezentující primární klíč konkrétního majitele, čímž umožníme snadnější práci s těmito souvisejícími záznamy (například vyhledání všech aut, které patří danému majiteli). Tento vztah je označován jako **1:N** (jednomu záznamu odpovídá více záznamů v jiné tabulce) [59].

Dalšími možnými vztahy mohou být

- **1:1** (One-to-one), u kterého jeden záznam v tabulce odpovídá pouze dalšímu jednomu záznamu v druhé tabulce a naopak.
- **M:N** (Many-to-Many) reprezentuje vztah, kdy více záznamů v jedné tabulce může odpovídat více záznamům v druhé (například produkt může mít více kategorií a jedna kategorie může mít více produktů). Pro uskutečnění tohoto vztahu je potřeba tzv. vazební tabulka. Jedná se o další tabulku v databázi, která je vytvořena speciálně pro popis M:N vztahu. Jejím obsahem jsou sloupce s cizími klíči, které odkazují na primární klíče v daných souvisejících tabulkách (případně další data popisující daný vztah). Pro zmíněný příklad produktů a kategorií by tak mohla existovat vazební tabulka se sloupci pro cizí klíč produktu a cizí klíč kategorie. Opět se tak usnadní práce s daty (lze snáze vyhledat všechny produkty v určité kategorií nebo všechny kategorie, ve kterých je daný produkt).

[59]

5.8.1 Jazyk SQL

Ačkoliv se ve vytvářeném projektu s databází pracuje výhradně za použití Entity Frameworku, který poskytuje abstrakci pro komunikaci s databází a odstiňuje tak od nutnosti používání SQL dotazů, může být vhodné pro bližší nastínění principů databází popsat, jak jinak může probíhat komunikace s databází bez tohoto frameworku.

Jazyk SQL (Structured Query Language) je dotazovací deklarativní jazyk používaný při práci s relačními databázemi. Obsahuje několik základních příkazů, které se dle své činnosti rozdělují do několika kategorií: DDL, DQL, DML, DCL [59].

- **DDL** (Data Definition Language) poskytují možnosti pro definici datového modelu databáze. To může zahrnovat vytváření, modifikaci nebo odstraňování databázových tabulek, indexů, pohledů, procedur. (Příkazy CREATE pro vytvoření databázového objektu, ALTER pro modifikaci existujícího nebo příkaz DROP pro odstranění objektu)
- **DML** (Data Manipulation Language) slouží pro samotnou manipulaci s daty v databázi. Může jít o vkládání záznamů, jejich aktualizaci nebo mazání. (Příkazy INSERT pro vložení záznamu, UPDATE pro aktualizaci existujícího záznamu, DELETE pro odstranění záznamů)

- **DQL** (Data Query Language) pro dotazování a výběr záznamů pomocí SELECT.
- **DCL** (Data Control Language) pro řízení a definici přístupových práv pomocí příkazů GRANT pro udělení přístupových práv a REVOKE pro jejich odebrání.

[60]

5.9 HTML

HTML (Hypertext Markup Language) je značkovací jazyk sloužící k popisu struktury webových stránek a představuje tak základní stavební kámen pro jejich tvorbu.

Jeho hlavní funkcionalita je založena na značkách (tzv. tagy), které se k definování obsahu a struktury webové stránky používají, přičemž tyto tagy můžou být párové či nepárové dle toho, zda jsou tvořeny počáteční a koncovou částí ( ), nebo se jedná pouze o počáteční část, která se uzavírá sama (<img src="image.jpg">). Díky těmto tagům můžeme jednotlivým částem webové stránky přiřadit konkrétní význam. Tyto tagy následně prohlížeč zpracuje a vytvoří na základě nich vizuální podobu stránky, která je zobrazena uživateli. Každý tag má tedy jiný vzhled – některé způsobí vykreslení tabulky, jiné třeba číslovaného seznamu.

Tag může obsahovat tzv. atributy, které jsou použity k jeho identifikaci nebo úpravě vzhledu a chování. Atribut může sloužit například k definování cesty k zobrazovanému obrázku. Pro zobrazení obrázku by tedy byl použit tag img s atributem src - *<img src="image.jpg">*.

Dokument může kromě samotných HTML tagů obsahovat i další prvky

- DTD direktivy začínající *<!*, které definují povolené elementy a atributy dokumentu (určení, že se jedná o HTML dokument).
- Komentáře, které se v samotném vizuálním obsahu stránky nezobrazují a jsou prohlížečem ignorovány, slouží pouze jako pomocné texty pro vývojáře. Jejich obsah se píše do speciálního tagu *<!— komentář -->*.
- JavaScript kód sloužící pro vytváření dynamických interaktivních prvků. Může být přímo součástí HTML dokumentu v párovém tagu *<script>* nebo definován v externím souboru.
- Kaskádové styly pro definici vzhledu HTML dokumentu.

Samotná struktura dokumentu se skládá z následujících částí:

- Deklarace toho, že se jedná o HTML dokument (DTD direktiva <!DOCTYPE html>).
- Kořenový párový tag <html>, které reprezentuje celý dokument.
- Hlavička dokumentu reprezentována párovým tagem <head>, která obsahuje metadata (kódování, název dokumentu, odkazy na externí zdroje, kaskádové styly, autor, klíčová slova).
- Tělo dokumentu definované párovým tagem <body> obsahující samotný viditelný obsah dokumentu.

[48]

5.10 CSS

CSS (Cascading Style Sheets), neboli kaskádové styly, jsou jazykem pro definování vzhledu zobrazovaných elementů v rámci webové stránky.

Zápis CSS představuje předpis několika pravidel. Každé takové pravidlo obsahuje selektor, pomocí kterého určujeme, na které prvky se bude daný vzhled aplikovat a specifikaci toho, jaký vzhled se bude aplikovat.

CSS je možné uvádět pomocí:

- **Inline zápisu** – konkrétní styly jsou zapsány přímo do atributu *style* daného elementu. Toto řešení není příliš vhodné vzhledem k tomu, že nedochází k dostatečnému oddělení vzhledu dokumentu od jeho obsahu, což může mít především u komplikovanějších stránek negativní vliv na udržovatelnost.
- **Interního zápisu**, kdy jsou styly zapsány v hlavičce dokumentu v tagu <*style*>.
- **Externího zápisu**, který umožňuje uvést styly do zcela separovaného souboru od HTML dokumentu, z kterého jsou tyto externí styly pouze načítány. Zápis stylů v externím souboru se podobá zmíněnému internímu zápisu – styly jsou zapsány pouze v <*style*> tagu.

[61]

Protože mimo inline zápis by nebylo jasné, na jaký element se má dané CSS aplikovat, využívají se tzv. selektory. Selektory umožňují vybrat specifické elementy stránky, na které se daná pravidla budou aplikovat.

- **Typové selektory** slouží k vybrání všech elementů daného typu. (nadpisy, odstavce, seznamy, tabulky).
- **Třídní selektory** se využívají ve chvíli, kdy nechceme styl aplikovat na všechny elementy daného typu. Třídní nejsou na rozdíl od typových předem určené, můžeme je tedy pojmenovat dle nejlepší výstižnosti. Aplikování stylu daného třídního selektoru na požadovaný element zajistíme uvedením názvu tohoto selektoru do *class* atributu v tagu tohoto elementu.
- **ID selektory** funguje obdobně jako selektor třídní, nicméně dané ID by se mělo přiřadit vždy jen jednomu elementu na zobrazované stránce.

[61]

5.10.1 CSS Frameworky

CSS Frameworky představují soubory předem předpřipravených CSS stylů, které lze aplikovat při tvorbě vlastních stránek, čímž samotnou tvorbu usnadňují. Mohou nabízet řešení pro často opakující se problémy, jako může být responzivita nebo design ovládacích prvků a často ke své funkcionalitě přidávají i JavaScript pluginy pro podporu této funkcionality. Tyto frameworky nicméně nemusí být vždy dobrou volbou, jelikož jeho použití může vést k tomu, že takto vytvořená stránka může vypadat velmi podobně jako mnoho dalších takto vytvořených stránek a pro důležitější projekty je jistě vhodnější, aby se web vyznačoval více originálním vzhledem. Nevýhodou může být jejich velikost, kdy není efektivní v kódu linkovat takto velké CSS/JS soubory, které nemusí být z velké části využity.

5.10.2 Bootstrap

Bootstrap je jedním z nejpoužívanějších CSS frameworků pro tvorbu webových stránek a aplikací. Je open-source a je možné jej využít zdarma v jakémkoliv projektu. Jeho popularita také znamená velké množství materiálů, které mohou pomoci pochopit principy tohoto frameworku a pomoci s vývojem.

Nabízí širokou škálu předdefinovaných komponent a stylů pro podporu různých oblastí. Může se jednat o předdefinované styly tlačítek, tabulek, formulářů, funkcionalit pro dropdown/collapse seznamy, modální okna, carousel pro slide-show prezentace, progress bar a mnoho dalších [28].

Pro jeho použití je nejdříve potřeba jej nalinkovat jako každý jiný CSS soubor. To je možné pomocí přímého stažení jeho zdrojových souborů a jejich následného nalinkování v hlavičce

HTML dokumentu, nebo můžeme využít CDN, kdy soubory není nutné přímo stahovat, ale stačí použít odkaz na tyto soubory v HTML hlavičce. Na webových stránkách bootstrapu je možné zvolit konkrétní části, které máme v plánu využívat a pouze tyto části následně stáhnout.

5.11 Chart.js

Tato JavaScriptová knihovna slouží k vykreslování velkých datových sad v podobě různých diagramů a grafů. Je šířena pod svobodnou licencí MIT a zastává značnou komunitu, která do této knihovny přidává novou funkcionalitu či opravuje stávající. V případě nalezení nějaké chyby je tedy velmi rychle opravena. Navíc disponuje důkladnou dokumentací a jelikož je rozšířena mezi komunitou, lze najít řešení pro různé problémy na platformě *Stack Overflow*. Spousta běžných grafů (např. histogram) je defaultně podporována, případně lze nové přidat pomocí komunitních rozšíření.

Hlavní výhodou této knihovny oproti ostatním knihovnám, které plní podobnou funkci, je ta, že byla vytvářena s ohledem na velké datové sady. Proto se grafický výstup vykresluje v HTML elementu *Canvas*, který vyžaduje vykreslovací kontext. Většina ostatních knihoven založených například na D3.js vykresluje grafický výstup v podobě SVG, které se skládá z několika uzlů. V případě velkých datových sad může mít tohle značný dopad na výkon klientské aplikace (muselo by se vytvořit přes tisíce uzlů).

Nevýhodou canvas přístupu ovšem je, že nelze přidávat vlastní CSS styly nebo navazovat volání JS funkcí na klientské události (např. *onclick*). Knihovna **Chart.js** však poskytuje širokou škálu možností, díky kterým lze přizpůsobit jednotlivé grafy. Výhodou tohoto řešení také je, že po integraci není třeba moc věcí upravovat a vytvořené grafy jsou již nachystané na produkční nasazení [68].

5.12 Vývojové prostředí

Jako vývojové prostředí bylo využito Visual Studio 2022, které je doporučeno jako vývojové prostředí pro .NET vývoj a zároveň je v této oblasti i velmi populární. Alternativou může být odlehčená verze Visual Studio Code, případně JetBranis Rider.

5.13 Systém správy verzí

Pro správu verzí je využíván distribuovaný systém pro správu verzí GIT, díky kterému je možné ukládat a sledovat změny v kódu v průběhu vývoje. Systém GIT je v současnosti nejpoužívanějším systémem pro správu verzí v softwarovém vývoji.

II. PRAKTICKÁ ČÁST

6 NÁVRH A IMPLEMENTACE

Tato kapitola se zabývá definováním funkcionalit aplikace a popisuje, jak byla aplikace implementována.

6.1 Funkcionální požadavky

- **Přihlášení**: Aplikace musí mít funkční přihlašovací systém, který umožní neregistrovaným uživatelům vstoupit do aplikace pomocí uživatelského jména a hesla.
- **Registrace**: Neregistrovaní uživatelé mají možnost se sami zaregistrovat do aplikace. Registrace by měla obsahovat minimální požadavky na uživatelské heslo, aby byl účet chráněn před neoprávněným přístupem.
- **Schválení registrace**: Administrátor by měl mít možnost povolit registrace nových uživatelů do aplikace. To znamená, že dokud administrátor žádost o registraci nepovolí, uživatel se nemůže přihlásit.
- **Správa uživatelů a rolí**: Administrátor by měl mít možnost spravovat uživatelské účty. To zahrnuje možnost povolit a zakázat účty a přidělit uživatelům role (Administrátor a Editor).
- **Nahrávání souborů**: Uživatelé mají možnost nahrávat soubory do aplikace, resp. ukládat je do sekcí, kam mají přístup.
- **Stahování souborů**: Uživatelé by měli mít možnost stahovat soubory, ke kterým mají přístup.
- **Vytváření složek**: Uživatelé by měli mít možnost vytvářet složky uvnitř sekcí, do kterých mají přístup pro lepší organizaci souborů.
- **Vytváření a úprava sekcí**: Uživatelé s rolí editora nebo administrátora by měli mít možnost vytvářet nové sekce a spravovat stávající sekce. Každá sekce by měla mít možnost přidat popis a název, aby se dalo uživatelům vědět, co se v sekci nachází.
- **Odstraňování sekcí**: Uživatelé s rolí editor nebo administrátor musí mít možnost sekci odstranit. Se sekcí se odstraní i všechny její soubory.
- **Zobrazování statistik**: Aplikace bude zobrazovat základní statistický přehled (počet uživatelů, souborů, přehled o přidělených tags)
- **Správa přístupu k sekcím**: Administrátor a Editor mají možnost spravovat, s kým je sekce sdílena, respektive kdo do ní má přístup. Mají možnost vybrat konkrétní uživatele a těm zvolit typ přístupu (žádný, pouze prohlížení, prohlížení a editace).

- **Vyhledávání souborů**: Uživatelé mají možnost vyhledat soubory v aplikaci, ke kterým mají přístup. Vyhledávání je založeno na názvu souboru.
- **Omezení přístupu**: Každý uživatel by měl mít přístup pouze k sekcím, ke kterým mu byl přidělen přístup. To znamená, že uživatelé nebudou mít přístup k souborům a sekcím, ke kterým nemají oprávnění. Sekce, do kterých nemají přístup, nejsou těmto uživatelům ani zobrazeny ve výběru.
- **Vytváření záznamů o fyzických předmětech**: Uživatel má možnost přidat záznam o fyzickém předmětu s jeho umístěním.
- **Odstraňování záznamů**: Uživatel s dostatečným oprávněním má možnost odstranit existující soubory, složky a záznamy o fyzických předmětech.
- **Vyhledávání záznamů**: Uživatel má možnost dle názvu vyhledávat v uložených záznamech, ke kterým má přístup.

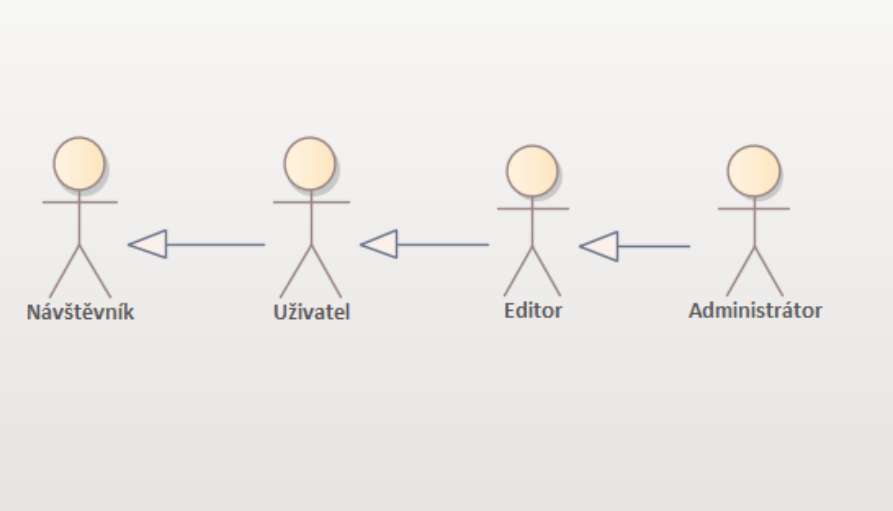
6.2 Nefunkcionální požadavky

- Aplikace by měla dodržovat základní bezpečnostní principy pro zamezení neoprávněného přístupu.
- Kód aplikace by měl být založen na MVC architektuře tak, aby byla aplikace snadněji udržovatelná a rozšiřovatelná.
- Aplikace by měla být intuitivní a jednoduchá na používání.

6.3 Aktéři a jejich oprávnění

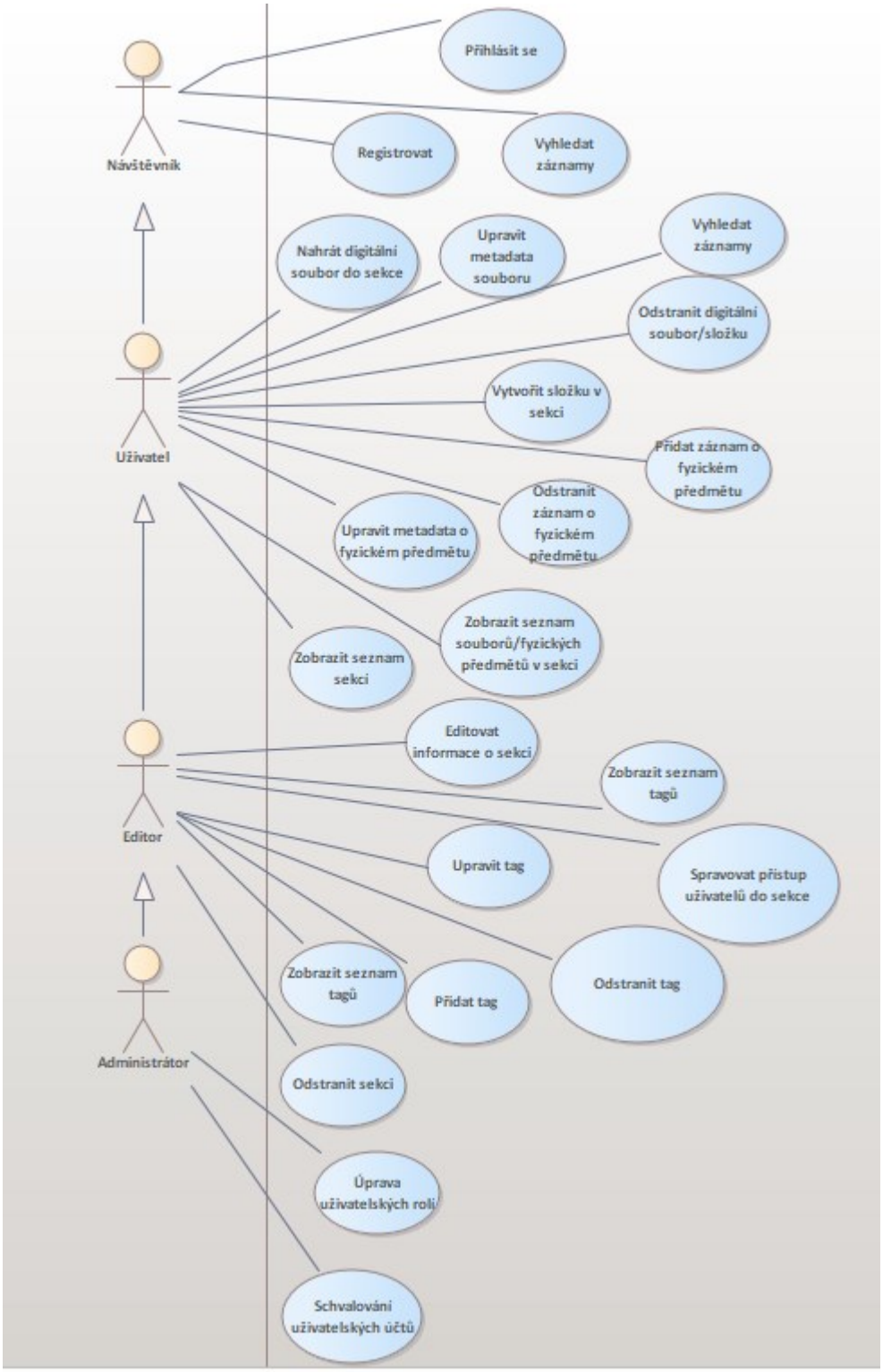
Aktéři (jejich diagram) slouží při návrhu k popisu rolí, které nějakým způsobem interagují s aplikací. Tato aplikace rozlišuje tři druhy aktérů.

- **Návštěvník** – každý, kdo není přihlášen nebo vůbec není registrován.
- **Uživatel** – je přihlášen, má možnost přistupovat do sekcí, ke kterým mu byl udělen přístup.
- **Editor** – přihlášen, disponuje vyššími oprávněními než běžný uživatel. Jsou mu vždy dostupné všechny sekce včetně možnosti řízení jejich přístupu, tzn. může nastavovat oprávnění uživatelům do jednotlivých sekcí, sekce přejmenovávat a mazat.
- **Administrátor** – přihlášen, má nejvyšší oprávnění. To zahrnuje všechny možnosti editora a možnost přiřazovat uživatelům administrátorské nebo editorské role, povolovat a zakazovat účty jiným uživatelům.



Obrázek 7. Diagram aktérů

6.4 USE CASE Diagram



Obrázek 8. Use Case diagram

6.5 Struktura projektu

Tato kapitola se zaměřuje na popsání organizace souborů v aplikaci, přičemž hlavní snahou je představit význam jednotlivých vrstev, do kterých byla rozložena.

Aplikace je ve snaze o udržovatelnější kód rozložena do čtyř vrstev, přičemž každá z nich má rozdílnou zodpovědnost a plní rozdílné skupiny úkolů, což umožňuje efektivnější správu kódu, budoucí rozšiřování a testování. Další z výhod je znovupoužitelnost. Jednotlivé vrstvy lze vyčlenit (jako samostatnou knihovnu) a následně využít v dalších projektech jako externí závislost a vyhnout se tak opakovanému psaní stejné funkcionality. Tímto konceptem se zabývá princip *Separation of Concerns (SoC)*.

V aplikaci byly vytvořeny tyto vrstvy:

1. Aplikační vrstva
2. Business vrstva
3. Datová vrstva
4. Souborová vrstva

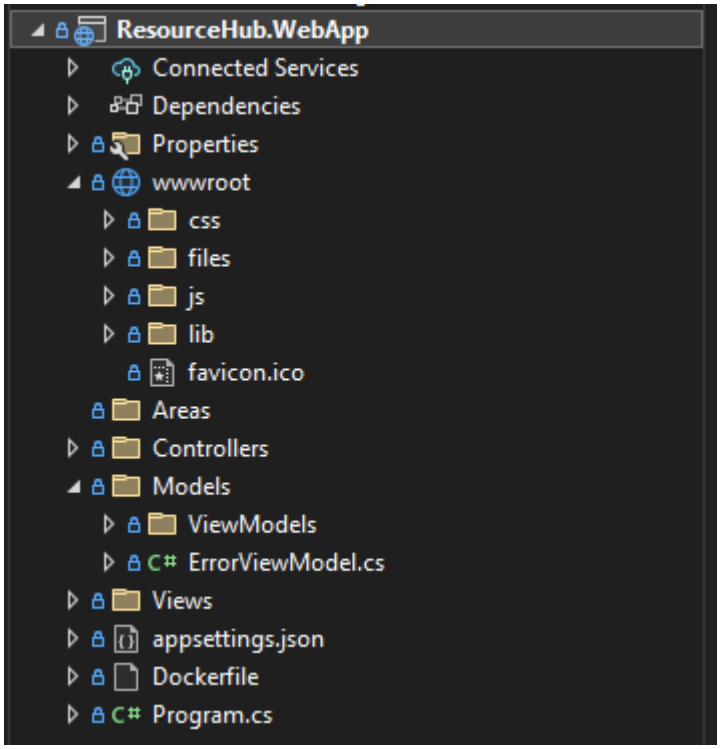
Mezi těmito vrstvami obecně existuje určitá hierarchie a ideálně by na sobě měly být co nejméně závislé. Pokud jsou zodpovědnosti správně rozděleny, pak například změny v aplikační vrstvě nemají žádný vliv na datovou vrstvu a naopak.

Realizace vrstev je provedena pomocí samostatných knihoven tříd (Class Libraries). Každá z nich tedy obsahuje (seskupuje) samostatné třídy a rozhraní, které se využívají v dalších vrstvách. Komunikace mezi jednotlivými knihovnami (vrstvami) je umožněna pomocí tzv. assembly references. Přidáním takové reference z jedné vrstvy na druhou umožníme projektu využívat požadované třídy a rozhraní, které jsou v těchto externích referencích obsaženy. Tuto referenci lze obvykle jednoduše přidat pomocí vývojového prostředí. Pro reprezentaci jednotlivých vrstev nemusí být využity knihovny tříd, může se jednat i o obyčejné třídy v rámci konkrétního projektu. Knihovny tříd byly zvoleny pro lepší rozdělení a znovupoužitelnost (vrstvu v podobě knihovny tříd můžeme dále samostatně jako DLL lépe referencovat v dalších projektech).

6.6 Aplikační vrstva

Aplikační vrstva představuje samotný MVC projekt reprezentující webovou aplikaci, která zpracovává požadavky od uživatele. Obsahuje tedy kontrolery, pohledy, modely a další soubory jako CSS styly, skripty a obrázky, které k ní patří.

Bližší struktura je zobrazena na obrázku:



Obrázek 9. Struktura aplikační vrstvy

- **Složka wwroot** slouží jako kořenový adresář pro statické zdroje (CSS, Javascript Soubory, ...). Mimo jiné se do této složky ukládají i soubory nahrané uživatelem (složka files). Konkrétně obsahuje další složky reprezentující jednotlivé sekce (tyto složky jsou pojmenovány dle ID sekcí). V obsahu těchto složek se dají nalézt soubory, které uživatel do dané sekce nahrál. Tyto soubory mají unikátní vygenerovaný název (neodpovídají tedy názvu, který uživatel zadal a který se zobrazuje v uživatelském rozhraní, ale používají se názvy z metadat uložených v databázi, viz datová vrstva). To je z důvodu jednodušší práce se soubory a s jejich identifikací (pokud se uživatel rozhodne přejmenovat soubor, změní se pouze metadata v databázi, které jsou s daným souborem spojeny, není potřeba přejmenovávat soubor na disku). Ve složce **lib** nalezneme kód bootstrapu a jQuery, případně sem můžeme nahrát další knihovny nebo frameworky dle potřeby.

- **Controllers** obsahuje všechny třídy kontrolerů.
- **Models** obsahuje především ViewModely, které slouží pro předávání dat mezi kontrolerem a pohledem.
- **Views** obsahuje všechny pohledy.
- **appsettings.json** slouží jako konfigurační soubor obsahující informace v JSON formátu. Lze v něm nalézt informace potřebné pro připojení k databázi.
- **Program.cs** obsahuje vstupní bod celého programu. Mimo jiné jsou v něm definovány náležitosti potřebné pro funkční dependency injection, která se využívá pro předávání závislostí (konkrétně manažery z business vrsvy, repozitáře z datové vrstvy, rozhraní pro práci se soubory ze souborové vrstvy). Lze zde nalézt také konfiguraci připojení k databázi, nastavení požadavků na minimální heslo a další.

6.7 Business Vrstva

Business vrstva obsahuje podstatnou část logiky řešení v podobě tzv. manager tříd. Umožňuje aplikační vrstvě na více místech jednotným rozhraním přistupovat k databázi a fyzickým souborům pomocí znovupoužitelných metod. Může se jednat o případ, kdy je potřeba zpracovat nahrávaný soubor, který je potřeba fyzicky uložit a zároveň k němu zapsat metadata do databáze. Komunikuje tedy s datovou a souborovou vrstvou a zapouzdřuje jejich funkcionalitu dohromady.

Obsahuje několik manažerů:

- **FileManager** obstarávající veškerou práci se soubory a složkami. Pracuje jak s datovou, tak se souborovou vrstvou. Pokud aplikační vrstva potřebuje zpracovat nahraný soubor uživatelem, zavolá metody z této vrstvy, které jednotně zpracují vše potřebné týkající se jak samotného fyzického souboru, tak s tímto souborem souvisejících metadat v databázi.
- **FilePermissionManager** obstarávající logiku týkající se kontroly přístupu k jednotlivým souborům, kterou může aplikační vrstva využít při potřebě kontroly oprávnění.
- **ItemManager** poskytující abstrakci pro práci se záznamy o fyzických předmětech.
- **SectionManager** pro práci se sekcemi.

6.8 Datová vrstva

Datová vrstva slouží k přímé manipulaci s daty v databázi (vkládání, čtení, aktualizace, odstraňování) pomocí Entity Frameworku. V této vrstvě je využíván návrhový vzor *Repository* (repozitář). Jeho smyslem je poskytnutí jednotného rozhraní pro komunikaci s daty a zapouzdření konkrétní implementace práce s datovým úložištěm, kdy pro každou entitu mapovanou do databáze existují třídy, které zajišťují základní operace pro manipulaci s danou entitou. Konkrétně jsou v této aplikaci zahrnuty tyto části:

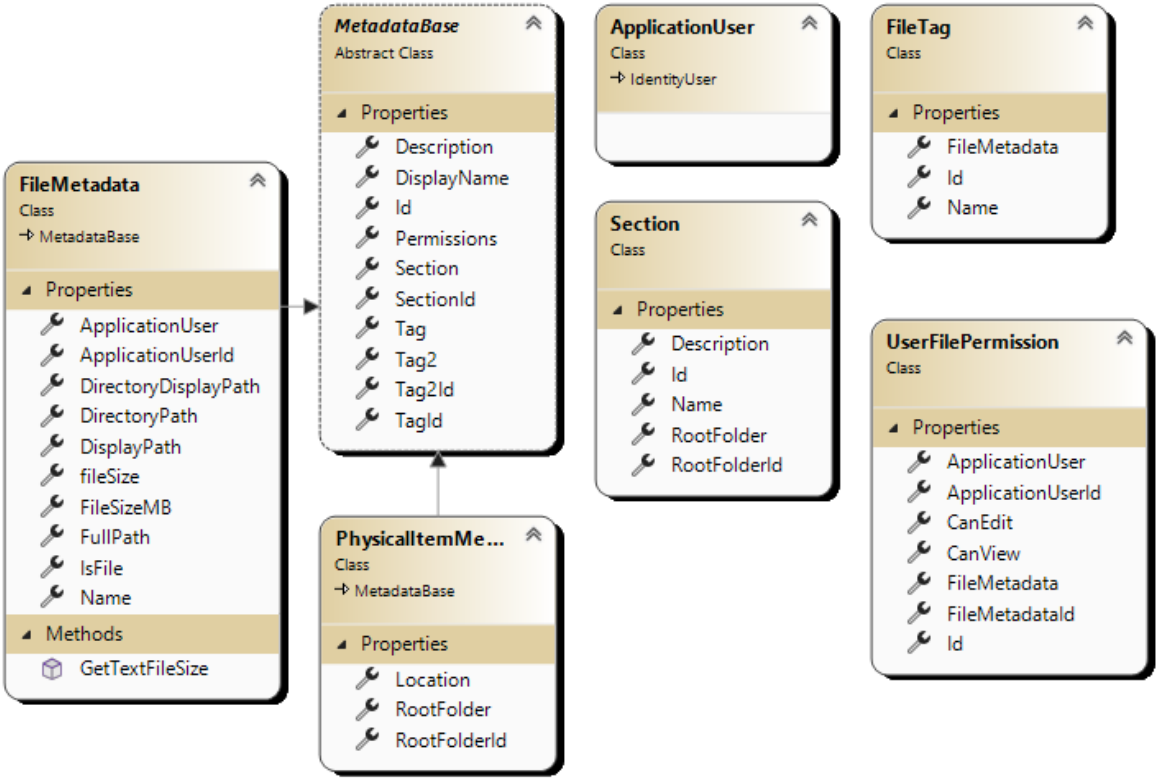
1. Rozhraní daného repozitáře, které slouží jako předpis metod pro manipulaci s danou entitou. Je vytvořeno rozhraní předepisující základní CRUD metody, přičemž od tohoto rozhraní dědí další rozhraní určené už pro konkrétní entity, které mohou přidávat další metody potřebné pro práci s danou entitou.
2. Třída implementující rozhraní repozitáře pro každou entitu. Existuje abstraktní předek všech repozitářů implementující základní CRUD metody, od kterého dědí další repozitáře pro konkrétní entity.
3. Samotné entity mapované do databáze rozdělené do tříd.

Vytvořeno je několik repozitářů:

- **FileMetadataRepository** pro operace s entitou FileMetadata
- **FilePermissionRepository** pro entitu UserFilePermission
- **FileTagRepository** pro entitu FileTag
- **ItemRepository** pro entitu PhysicalItemMetadata
- **SectionRepository** pro entitu Section

6.8.1 Diagram tříd a databázový model

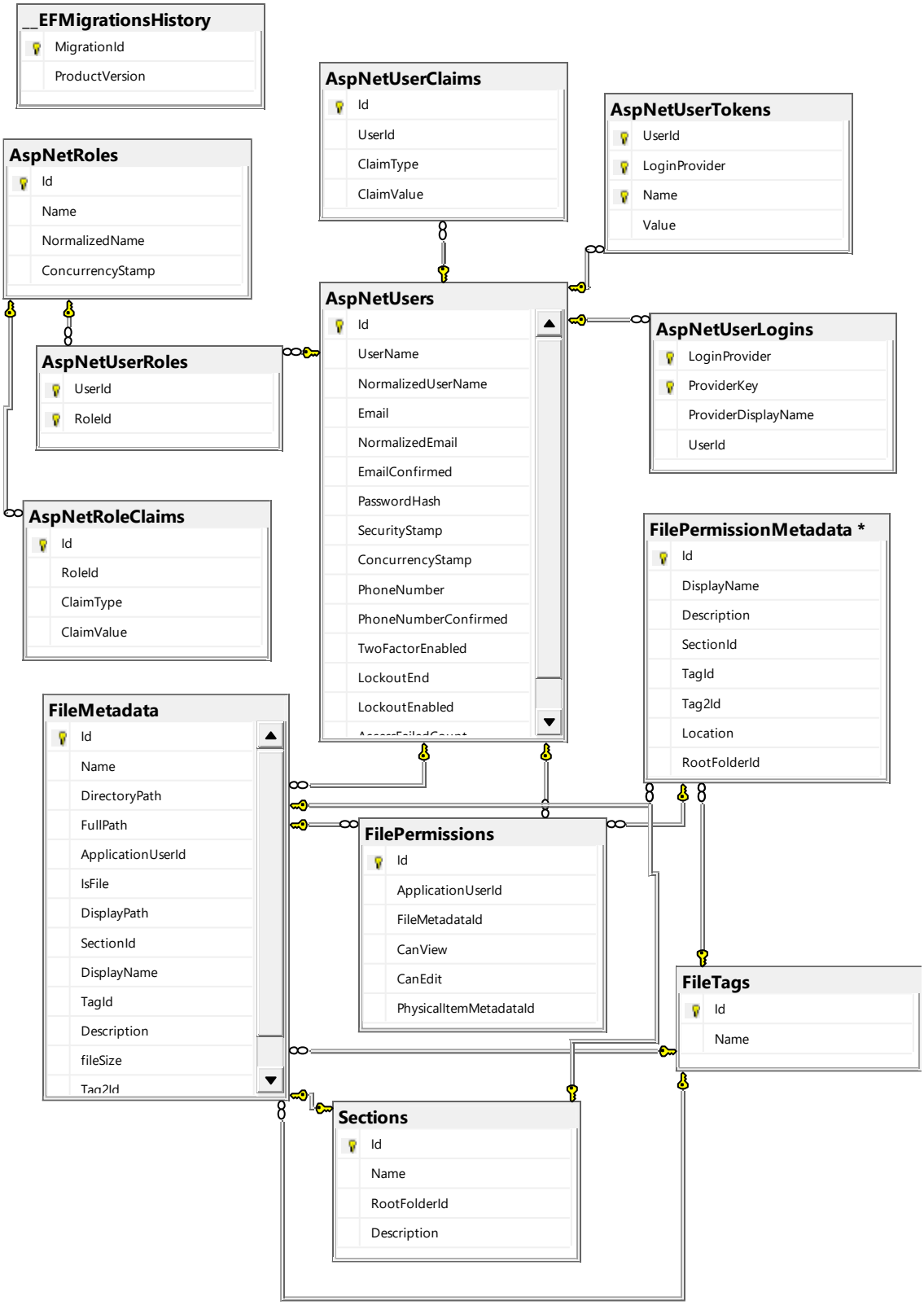
Jak bylo zmíněno, pro práci s databází byl využit Entity Framework s využitím metodiky *Code First* – databázový model byl vygenerován na základě těchto tříd:



Obrázek 10. Diagram tříd

MetadataBase (předek metadat záznamů) je abstraktní třídou poskytující základní vlastnosti pro třídy uchovávající metadata o souborech a fyzických předmětech (třídy FileMetadata a PhysicalItemMetadata). MetadataBase třídě je přiřazena instance UserFilePermission, která popisuje, jaká oprávnění k jednotlivým záznamům uživatelé mají. Třída FileTag reprezentuje, které tagy jsou přiřazeny jednotlivým metadatům. Třída Section reprezentuje jednotlivé sekce, do kterých jsou záznamy řazeny. ApplicationUser třída slouží jako rozšíření třídy IdentityUser poskytované *Identity Frameworkem*. Ačkoliv neimplementuje žádné nové vlastnosti, ostatní třídy aplikace s ní pracují a aplikace je lépe připravena na případné rozšiřování informací o uživateli.

Vygenerovaný databázový model Entity Frameworkem odpovídá tomuto schématu:



Obrázek 11. Databázový model

Jak vyplývá z obrázku, oba typy záznamů (záznamy o metadatech souborů reprezentované tabulkou FileMetadata a metadatech o fyzických předmětech reprezentované tabulkou PhysicalItemMetadata) mohou mít několik FilePermissions (vztah 1:M). Z principu existuje jeden záznam FilePermission pro jednu kombinaci uživatele a metadat. Sekce mají jednu kořenovou složku (RootFolderId), dle které se určuje oprávnění ke konkrétní sekci na základě této složce přiřazeným FilePermissions. Dále bylo Entity Frameworkem vygenerováno několik dalších tabulek sloužících pro práci s uživateli, jejich rolemi a další.

6.9 Souborová vrstva

Souborová vrstva poskytuje metody zodpovědné za manipulaci se soubory uloženými na fyzickém disku a slouží tedy jako abstrakce pro práci s fyzickými soubory pro ostatní vrstvy (konkrétně business vrstvu). Ostatní vrstvy se tak vůbec nemusí starat o způsob, jak soubory fyzicky ukládat a jak pracovat s diskem, stačí jim pouze využívat rozhraní, které souborová vrstva poskytuje.

6.10 Zabezpečení

Autentizace probíhá pomocí integrovaných mechanismů Identity Frameworku v ASP.NET Core. Tento systém poskytuje kvalitní a bezpečné funkcionality pro správu uživatelů od registrace přes přihlášení až po správu hesel a rolí.

Autentizace a autorizace uživatelů probíhá i na straně serveru, kde je zajištěna pomocí atributů *[Authorize]* u kontrolerů, případně jeho metod, čímž se zajišťuje, že do určitých částí aplikace nemají uživatelé s nedostatečným oprávněním přístup. Ačkoliv jsou na straně uživatele zobrazovány vždy jen soubory a sekce (případně možnosti jejich modifikace), ke kterým má přístup, mohl by se některý z uživatelů pokusit pozměnit parametry odesílané na server (např. ID záznamu), což by mohlo vést k získání možnosti modifikace pro tohoto uživatele jinak nedostupných záznamů. Pro tento případ jsou přístupová práva uživatele pro záznamy znovu kontrolována v odpovídajícím kontroleru.

Všechna hesla uživatelů jsou uložena v hashované podobě, přičemž dané hashování je automaticky zajištěno metodou *CreateAsync* na instanci UserManageru poskytovaného Identity Frameworkem.

Ochranu proti CSRF útokům zajišťuje atribut *[AutoValidateAntiforgeryToken]* nad kontrolery. Tato ochrana funguje na principu kontroly unikátního tokenu, který je ve view pomocí

ASP.NET Core Frameworku vypisován ke každému formuláři a následně kontrolován v odpovídajícím kontroleru.

Použití přístupu Code First v rámci Entity Frameworku zajišťuje ochranu proti SQL Injection útokům. Toho je dosaženo díky tomu, že Entity Framework automaticky generuje SQL dotazy na základě LINQ výrazů takovým způsobem, že je uživatelský vstup považován za parametry namísto jejich přímého spojení s dotazem. Parametry by tedy měly být správně ošetřeny a měla by být zajištěna bezpečná komunikace s databází.

7 TESTOVÁNÍ

Testování aplikace probíhalo zejména při samotném vývoji, konkrétně procházením jednotlivých funkcionalit dle use-casů po tom, co byly implementovány. Jednalo se tedy o formu manuálního testování, přičemž nevýhody tohoto testování se projevovaly zejména v komplikovanějším regresním testování, kdy úprava stávajících funkcionalit občasně přinášela chyby do dříve funkčních částí. Tím, že tyto testy nebyly zautomatizovány, se musely ručně provádět znovu, aby se takové zanesení případných chyb odhalilo.

Většina hlavních chyb se týkala nesprávného přiřazování metadat k záznamům, zejména oprávnění a jejich kontroly pro přístup k těmto záznamům. Hlavní bezpečnostní chybou bylo nekontrolování přístupových práv k souborům na straně serveru. Ačkoliv měl k danému kontroleru přístup pouze autorizovaný uživatel, nebrala se v potaz možnost, že by některý z uživatelů pozměnil parametry požadavku tak, aby přistupoval k jiným záznamům než těm, které mu byly vypsány v uživatelském rozhraní. Tento problém byl vyřešen kontrolou metadat s oprávněním daného souboru pro konkrétního uživatele. Během testování bylo rovněž zjištěno, že při nahrání nového záznamu do sekce se nenastavovala oprávnění pro tento záznam všem uživatelům dle jejich odpovídajících oprávnění pro tuto sekci, ale oprávnění byla nastavena pouze uživateli, který tento soubor nahrál. Další výraznější chyba se týkala odstraňování složek, resp. rekurzivního mazání jejich obsahu, kdy se složka jevila jako odstraněná, nicméně její obsah se na serveru stále nacházel.

Následně byla aplikace k dispozici nezávislým uživatelům, kteří měli za úkol dle diagramu projít všemi use-casy a případné chyby či nedostatky nahlásit. Při tomto testování bylo zjištěno, že při zavření modálního formuláře pro úpravu záznamů se neodstraňují vyplněné údaje a neexistuje ani žádná jiná možnost, jak pole ve formuláři „vyresetovat“. Tento problém byl rovněž opraven (tlačítko pro zavření modálního formuláře je typu *reset*).

ZÁVĚR

Hlavní cíle práce představovaly nastínění důležitosti sdílení informací a s tím souvisejících témat v rámci IS v organizacích a vytvoření informačního systému, které takové sdílení bude podporovat.

V teoretické práci bylo popsáno, co může sdílení zdrojů představovat, konkrétně se zaměřením na sdílení znalostních zdrojů v organizaci spolu s tím, proč je dobré jej využívat. Zmíněny tedy byly výhody sdílení zdrojů za pomocí informačních systémů. Popsán byl i znalostní management, který je se sdílením zejména znalostních zdrojů spjatý. Jelikož se práce zabývá vytvořením informačního systému, byla určitá část práce věnována informačním systémům obecně. Zmíněny byly jedny z nejznámějších systémů zabývajících se obdobnou problematikou. K závěru teoretické části je uvedeno srovnání pro případy volby již existujících informačních systému oproti tvorbě vlastních následováno vybranými technologiemi zvolenými pro implementaci systému v praktické části. Klíčovou roli pro výběr ASP.NET MVC hrála podpora MVC architektury a dalších frameworků usnadňující práci s databází a řešení zabezpečení, v neposlední řadě také znalosti těchto technologií.

V praktické části byly definovány požadavky na systém, které vycházely z existujících řešení. Rozšíření a odlišení se od zmíněných existujících systémů je možnost přidání záznamu o fyzické položce. Tímto způsobem je možné podpořit sdílení zdrojů nejen v elektronické podobě, ale i ve fyzické. Jak bylo zmíněno, systém byl testován jak během vývoje po implementaci konkrétních use-casů, tak i po samotném dokončení nezávislými uživateli, kteří tyto use-casy procházeli.

Funkcionalitu pro podporu sdílení fyzických položek by však bylo vhodné do budoucna rozšířit (například o možnost definovat různé typy fyzických položek a jejich atributů v JSON formátu). Dalším nedostatkem je, že aplikační vrstva na několika místech nevyužívá business vrstvy, ale pracuje přímo s repozitáři z vrstvy datové. Navzdory tomu je aplikace funkční a lze ji využít jak pro sdílení zdrojů v digitální podobě, tak pro uchovávání informací o sdílených zdrojích v podobě fyzické, čímž je hlavní cíl práce splněn a aplikaci lze využívat.

SEZNAM POUŽITÉ LITERATURY

[1] BUREŠ, Vladimír. Znalostní management a proces jeho zavádění: průvodce pro praxi. Praha: Grada, 2007. Management v informační společnosti. ISBN 978-80-247-1978-8.

[2]. Becerra-Fernandez, Irma, and Rajiv Sabherwal. Knowledge Management Systems and Processes. Second Edition. New York: Routledge, 2015. ISBN 9780765639158.

3 [Joseph Kasten. Thoughts on the relationship of knowledge organization to knowledge management. Knowledge organization. 2007, 34 (1), s 10.]

[4] Co je management znalostí | myTimi.cz . Outsourcujte vše na jedno místo | myTimi.cz [online]. Copyright ©2018 http [cit. 12.11.2022]. Dostupné z: https://www.mytimi.cz/co-je-knowledge-management/

[5] What is Knowledge Management?. BINUS QMC | Quality Management Center [online]. Copyright © BINUS UNIVERSITY. All rights reserved. [cit. 12.11.2022]. Dostupné z: https://qmc.binus.ac.id/2017/11/10/what-is-knowledge-management/

[6] Jak podpořit sdílení znalostí v organizaci | CAFINews. Články | CAFINews [online]. [cit. 2023-03-23]. Dostupné z: https://news.cafin.cz/clanek/jak-podporit-sdileni-znalosti-v-organizaci

[7] TRUNEČEK, Jan. Management znalostí. Praha: C.H. Beck, 2004. C.H. Beck pro praxi. ISBN 80-7179-884-3.

[8] KUČEROVÁ, Helena. Organizace znalostí: klíčová témata. Praha: Univerzita Karlova, nakladatelství Karolinum, 2017. ISBN 978-80-246-3587-3.

[9] URBANCOVÁ, Hana. Kontinuita znalostí v organizacích [online]. Praha, 2011, 256 s. [cit. 2023-05-23]. Dostupné z: https://www.pef.czu.cz/dl/46204. Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta. Vedoucí práce Jan Hron.

[11] Oceňování informačních systémů | Grant Thornton - podporujeme váš růst. Grant Thornton - podporujeme váš růst [online]. Copyright © 2023 Grant Thornton [cit. 23.05.2023]. Dostupné z: https://grantthornton.cz/sluzba/ocenovani-informacnich-systemu

[12] MIT Press. Home Page - MIT Press [online]. Copyright © 2023 MIT Press. All Rights Reserved. [cit. 14.01.2023]. Dostupné z: https://mitpress.mit.edu/sites/default/files/titles/content/9780262015387_sch_0001.pdf

[13] TVRDÍKOVÁ, Milena. Aplikace moderních informačních technologií v řízení firmy: nástroje ke zvyšování kvality informačních systémů. Praha: Grada, 2008. Management v informační společnosti. ISBN 978-80-247-2728-8.

[14] SODOMKA, Petr a Hana KLČOVÁ. Informační systémy v podnikové praxi. 2., aktualiz. a rozš. vyd. Brno: Computer Press, 2010. ISBN 978-80-251-2878-7.

[15] Informační systémy v kostce: ERP, CRM, implementace. Experten in der digitalen Welt | Rascasone [online]. Copyright © [cit. 20.02.2023]. Dostupné z: https://www.rascasone.com/cs/blog/informacni-systemy-erp-crm-implemetace#moduly-informacn-iacute-ch-syst-eacute-mu

[16] Different Types of Information System and the Pyramid Model. Chris-Kimble.Com for books, papers and courses from Chris Kimble [online]. [cit. 2023-04-23]. Dostupné z: http://www.chris-kimble.com/Courses/World_Med_MBA/Types-of-Information-System.html

[17] Úvod do interní komunikace : Marketing journal. Kreativní PR a digitální agentura s garancí výsledku | Focus [online]. Copyright © 2004 [cit. 13.05.2023]. Dostupné z: https://www.focus-age.cz/m-journal/uvod-do-interni-komunikace__s317x547.html

[18] Firemní intranet a jeho funkce | Algotech. Podniková IT a cloudová řešení | Algotech [online]. Copyright ©2022 [cit. 12.11.2022]. Dostupné z: https://www.algotech.cz/novinky/2021-12-29-firemni-intranet-a-jeho-funkce

[19] ŠTRÁFELDA, Jan. Co je intranet. Cafin [online]. [cit. 2023-03-23]. Dostupné z: https://www.strafelda.cz/intranet

[20] The history of C# - C# Guide | Microsoft Learn. [online]. Copyright © Microsoft 2023 [cit. 13.05.2023]. Dostupné z: https://learn.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-version-history#c-version-10-1

[21] Co je .NET MAUI? - .NET MAUI | Microsoft Learn. [online]. Copyright © Microsoft 2023 [cit. 13.05.2023]. Dostupné z: https://learn.microsoft.com/cs-cz/dotnet/maui/what-is-maui?view=net-maui-7.0

[22] What is .NET? An open-source developer platform.. .NET | Build. Test. Deploy. Cafin [online]. [cit. 2023-03-23]. Dostupné z: https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet

[23] What Is New In .NET 6.0. C# Corner - Community of Software and Data Developers [online]. Copyright ©2023 C [cit. 13.05.2023]. Dostupné z: https://www.c-sharpcorner.com/article/what-is-new-in-net-6-0/

[24] Architektura MVC: definice, struktura, frameworky. Experten in der digitalen Welt | Rascasone [online]. Copyright © [cit. 13.05.2023]. Dostupné z: https://www.rascasone.com/cs/blog/architektura-mvc-struktura-frameworky

[25] Lekce 3 - Úvod do MVC architektury v ASP.NET. itnetwork.cz - Učíme národ IT [online]. Copyright © 2023 itnetwork.cz. Veškerý obsah webu [cit. 13.05.2023]. Dostupné z: https://www.itnetwork.cz/csharp/asp-net-mvc/zaklady/asp-dot-net-uvod-do-mvc-architektury

[26] Entity Framework Tutorial [online]. [cit. 2023-01-23]. Dostupné z: https://www.entityframeworktutorial.net/entityframework6/dbcontext.aspx

[27] Entity Framework Tutorial [online]. [cit. 2023-01-23]. Dostupné z: https://www.entityframeworktutorial.net/lazyloading-in-entity-framework.aspx

[28] Customize and download · Bootstrap. Bootstrap · The most popular HTML, CSS, and JS library in the world. [online]. [cit. 2023-01-23]. Dostupné z: https://getbootstrap.com/docs/3.4/customize/

[29] Dvě tváře sdílené ekonomiky. Co to je sdílená ekonomika? | by Lucie Scholzová | EDTECH KISK | Medium. Medium – Where good ideas find you. [online]. [cit. 2023-01-23]. Dostupné z: https://medium.com/edtech-kisk/dv%C4%9B-tv%C3%A1%C5%99e-sd%C3%ADlen%C3%A9-ekonomiky-8f83930765a

[30] What Is the Sharing Economy?. The Balance - Make Money Personal [online]. Dostupné z: https://www.thebalancemoney.com/what-is-the-sharing-economy-5188892#citation-1).

[31] SOCOS IT s.r.o.. SOCOS IT - efektivní a bezpečná správa elektronických dokumentů. Cafin [online]. [cit. 2023-01-23]. Dostupné z: Dostupné z: https://www.socosit.cz/novinky/co-je-dms-systém

[32] Co je CMMS/EAM? - TechIS – software pro plánování a řízení podnikové údržby a servisu a správu podnikového majetku. Domů - TechIS – software pro plánování a řízení podnikové údržby a servisu a správu podnikového majetku [online]. Copyright © 2023 [cit. 13.01.2023]. Dostupné z: https://techis.eu/co-je-cmms-eam/

[33] Microsoft SharePoint | IT služby Masarykovy univerzity. IT služby Masarykovy univerzity [online]. Copyright © 2023 [cit. 13.05.2023]. Dostupné z: https://it.muni.cz/sluzby/microsoft-sharepoint

[34] Přidání aplikace na web - Podpora Microsoftu. [online]. [cit. 2023-04-23]. Dostupné z: https://support.microsoft.com/cs-cz/office/p%C5%99id%C3%A1n%C3%AD-aplikace-na-web-ef9c0dbd-7fe1-4715-a1b0-fe3bc81317cb

[35] studuj. digital [online]. [cit. 2023-01-23]. Dostupné z: https://studuj.digital/2022/01/25/sharepoint-aneb-vse-na-jednom-miste-odkudkoliv

[36] [online]. Copyright © [cit. 13.05.2023]. Dostupné z: http://projanco.com/Library/SharePoint%202016%20User%E2%80%99s%20Guide.pdf

[37] Co je knihovna dokumentů? [online]. [cit. 2023-01-23]. Dostupné z: https://support.microsoft.com/cs-cz/office/co-je-knihovna-dokument%C5%AF-3b5976dd-65cf-4c9e-bf5a-713c10ca2872

[38] Develop SharePoint Add-ins | Microsoft Learn. [online]. Copyright © Microsoft 2023 [cit. 13.05.2023]. Dostupné z: https://learn.microsoft.com/en-us/sharepoint/dev/sp-add-ins/develop-sharepoint-add-ins

[40] What is the Dropbox file size limit? - Dropbox Help. Dropbox Help Center - How to use Dropbox - Dropbox Help [online]. [cit. 2023-01-23]. Dostupné z: https://help.dropbox.com/sync/upload-limitations

[41] Storage and upload limits for Google Workspace - Google Workspace Admin Help. Google Help [online]. Copyright ©2023 Google [cit. 13.05.2023]. Dostupné z: https://support.google.com/a/answer/172541?hl=en

[42] Share Files and Links - Dropbox. Dropbox.com [online]. [cit. 2023-01-23]. Dostupné z: https://www.dropbox.com/features/share

[43] Recover Older Versions of Files - Dropbox Help. Dropbox Help Center - How to use Dropbox - Dropbox Help [online]. [cit. 2023-01-23]. Dostupné z: https://help.dropbox.com/delete-restore/recover-older-versions

[44] Compare All Dropbox Plans - Dropbox. Dropbox.com [online]. [cit. 2023-01-23]. Dostupné z: https://www.dropbox.com/plans

[45] Microsoft Corporation. Microsoft Corporation [online]. [cit. 2023-01-23]. Dostupné z: https://www.microsoft.com/en-us/microsoft-365/onedrive/compare-onedrive-plans?activetab=tab:primaryr1

[46] Outsourcing – aneb trocha teorie nikoho nezabije a možná i pomůže | ISVS.CZ. ISVS.CZ | zpravodajství z oblastí ISVS a eGovernmentu [online]. Copyright © 2001 [cit. 13.05.2023]. Dostupné z: https://2011-2015.isvs.cz/outsourcing-aneb-trocha-teorie-nikoho-nezabije-a-mozna-i-pomuze/

[47] ClickUp's Project Hierarchy - Structuring Work in ClickUp™. ClickUp™ | One app to replace them all [online]. Copyright © [cit. 14.05.2023]. Dostupné z: https://clickup.com/hierarchy-guide

[48] Co jsou kaskádové styly (CSS). Jan Štráfelda: průvodce online projektem [online]. Dostup-né [online]. [cit. 2023-01-23]. Dostupné z: https://www.strafelda.cz/kaskadove-styly

[49] What is the difference between ERP CRM and SCM?. Top ERP consultant - NC, VA, SC, GA, FL - ERP Support [online]. Copyright © 2023 Intelligent Technologies, Inc. [cit. 23.05.2023]. Dostupné z: https://www.inteltech.com/what-is-the-difference-between-erp-crm-and-scm/

[50] Správa a integrace podnikových systémů | VITSOL. Outsourcing IT Ostrava, Helpdesk IT | VITSOL [online]. Copyright © 2019 [cit. 23.05.2023]. Dostupné z: https://www.vitsol.cz/sprava-a-integrace-podnikovych-systemu/

[51] What is the difference between ERP CRM and SCM?. Top ERP consultant - NC, VA, SC, GA, FL - ERP Support [online]. Copyright © 2023 Intelligent Technologies, Inc. [cit. 23.02.2023]. Dostupné z: https://www.inteltech.com/what-is-the-difference-between-erp-crm-and-scm/

[52] C# vs. Java: 5 Irreplaceable C# Features We'd Kill to Have in Java | Harness. Harness | The Modern Software Delivery Platform - CI, CD, Feature Flags, Cloud Costs & more [online]. [cit. 23.02.2023]. Dostupné z: https://www.harness.io/blog/c-vs-java-5

[53] Fundamentals of garbage collection | Microsoft Learn [online]. [cit. 2023-05-23]. Dostupné z: https://learn.microsoft.com/en-us/dotnet/standard/garbage-collection/fundamentals

[54] ASP.NET MVC Overview | Microsoft Learn. [online]. [cit. 2023-05-23]. Dostupné z: https://learn.microsoft.com/en-us/aspnet/mvc/overview/older-versions-1/overview/asp-net-mvc-overview

[55] What is entity framework [online]. [cit. 2023-05-23]. Dostupné z: https://www.entityframeworktutorial.net/what-is-entityframework.aspx

[56] What is an ORM – The Meaning of Object Relational Mapping Database Tools. [online]. [cit. 2023-05-23]. Dostupné z: https://www.freecodecamp.org/news/what-is-an-orm-the-meaning-of-object-relational-mapping-database-tools/

[57] What is the Difference Between Code First and Database First Approach in MVC - Pediaa.Com. Pediaa.Com - Know about Anything [online]. Copyright © 2017 [cit. 23.05.2023]. Dostupné z: https://pediaa.com/what-is-the-difference-between-code-first-and-database-first-approach-in-mvc/

[58] What Is a Relational Database | Oracle. Oracle | Cloud Applications and Cloud Platform [online]. Copyright © 2023 Oracle [cit. 23.05.2023]. Dostupné z: https://www.oracle.com/database/what-is-a-relational-database/

[59] Database Relationships [online]. [cit. 2023-05-23]. Dostupné z: https://condor.depaul.edu/gandrus/240IT/accesspages/relationships.htm

[60] SQL | DDL, DQL, DML, DCL and TCL Commands - GeeksforGeeks. GeeksforGeeks | A computer science portal [online]. [cit. 2023-05-23]. Dostupné z: https://www.geeksforgeeks.org/sql-ddl-dql-dml-dcl-tcl-commands/

[61] CSS Selectors. W3Schools Online Web Tutorials [online]. [cit. 2023-05-23]. Dostupné z: https://www.w3schools.com/css/css_selectors.asp

[62] Plans & Pricing [online]. [cit. 2023-03-23]. Dostupné z: [online]. Dostupné z: https://one.google.com/about/plans

[63] Supported File Types [online]. [cit. 2023-05-23]. Dostupné z: https://help.dropbox.com/view-edit/viewable-file-types

[64] ClickUp™ Pricing | Free Forever, Unlimited, & Business Plans. ClickUp™ | One app to replace them all [online]. Copyright © [cit. 24.05.2023]. Dostupné z: https://clickup.com/pricing

[65] Microsoft Corporation. Microsoft Corporation [online]. [cit. 2023-05-24]. Dostupné z: https://www.microsoft.com/en-ww/microsoft-365/sharepoint/compare-sharepoint-plans?market=af

[66] Eager Loading [online]. [cit. 2023-03-24]. Dostupné z: https://www.entityframework-tutorial.net/eager-loading-in-entity-framework.aspx

[67] Explicit Loading [online]. [cit. 2023-03-24]. Dostupné z: https://www.entityframe-worktutorial.net/EntityFramework4.3/explicit-loading-with-dbcontext.aspx

[68] Chart.js. Chart.js | Open source HTML5 Charts for your website [online]. [cit. 2023-04-24]. Dostupné z: https://www.chartjs.org/docs/latest/

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: CD-ROM

Přiložené CD obsahuje zdrojový kód aplikace ve formátu .zip a text bakalářské práce ve formátu .pdf.