Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Optimalizace podnikových procesů

Bc. Dominik Janák

Diplomová práce

2021

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Dominik Janák**
Osobní číslo:	**I19279**
Studijní program:	**N0613A140007 Informační technologie**
Studijní obor:	**Informační technologie**
Téma práce:	**Optimalizace podnikových procesů**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování

Práce se zabývá návrhem a implementací webového informačního systému sloužícímu k optimalizaci a částečné automatizaci vnitropodnikových procesů. Systém automaticky vytěžuje data z emailové schránky a dává možnost jednotlivé přijaté emaily zařadit ke zpracování. V průběhu zpracování dochází, prostřednictvím definovaného workflow, k delegování akcí na patřičné uživatele. Celý sytém využívá moderních vývojových technik (MVC, ORM, REST, Dependency Injection) a je implementován za pomoci webových technologií Symfony 5 a React.

Rozsah pracovní zprávy: **50 – 60 stran**
Rozsah grafických prací:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

*CARDA, Antonín a Renata KUNSTOVÁ. Workflow: nástroj manažera pro řízení podnikových procesů. 2. rozš. a aktualiz. vyd. Praha: Grada, 2003. Management v informační společnosti. ISBN 80-247-0666-0.
*GAMMA, Erich. Design patterns: elements of reusable object-oriented software. Boston: Addison-Wesley, 1995. ISBN 978-0-201-63361-0.
*SVOZILOVÁ, Alena. Zlepšování podnikových procesů. Praha: Grada, 2011. Expert (Grada). ISBN 978-80-247-3938-0.
*ŠPERKA, Roman. Informační podpora podnikových procesů. Jesenice: Ekopress, 2019. ISBN 978-80-87865-55-2.

Vedoucí diplomové práce: **Ing. Jaroslav Dvořák**
Autocont CZ, a.s.

Datum zadání diplomové práce: **6. listopadu 2020**
Termín odevzdání diplomové práce: **15. května 2021**

L.S.

Ing. Zdeněk Němec, Ph.D. v.r.
děkan

prof. Ing. Antonín Kavička, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 30. listopadu 2020

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, Pravidla pro zveřejňování závěrečných prací a jejich základní jednotnou formální úpravu, ve znění pozdějších dodatků, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 8. srpna 2021

Bc. Dominik Janák

Poděkování

Rád bych touto cestou poděkoval vedoucímu diplomové práce, Ing. Jaroslavu Dvořákovi a odborníkovi z praxe, Josefu Lindourkovi, za jejich připomínky a cenné rady, díky nimž se mohla tato práce stát skutečností.

Mé poděkování za obětavost a podporu patří také mé drahé mamince, která mě v průběhu mých studií bezmezně podporovala.

ANOTACE

Práce se zabývá návrhem a implementací webového informačního systému, který sloužící k optimalizaci a částečné automatizaci podnikových procesů. Systém automaticky vytěžuje data z e-mailové schránky a dává možnost zařadit přijaté e-maily ke zpracování. V jeho průběhu dochází prostřednictvím definovaného workflow k delegování akcí na patřičné uživatele. Celý systém využívá moderních vývojových technik MVC, ORM, REST a Dependency Injection a je implementován za pomoci webových technologií Symfony 5 a React.

KLÍČOVÁ SLOVA

Symfony 5, React, REST, workflow, proces, vytěžování dat

TITLE

Business process optimization

ANNOTATION

The master thesis deals with the design and implementation of a web-based information system. It is used for optimization and partial automation of business processes. The system automatically extracts data from the inbox and classifies the received e-mails for processing. During the process, actions are delegated to the proper users through a defined workflow. The whole system uses modern development techniques such as MVC, ORM, REST, and Dependency Injection and is implemented by web technologies Symfony 5 and React.

KEYWORDS

Symfony 5, React, REST, workflow, process, data mining

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM ZKRATEK

ABAP	Advanced Business Application Programming
ADC	Application Delivery Controller
AJAX	Asynchronous JavaScript And XML
API	Application Programming Interface (*Aplikační rozhraní*)
BPR	Business Proces Reengineering (*Reengineering podnikových procesů*)
CC	Carbon Copy (*Veřejná kopie*)
CD	Continuous delivery (*Průběžné doručení*)
CI	Continuous integration (*Průběžná integrace*)
CLI	Command Line Interface (*Příkazový řádek*)
ČSN	Česká Technická Norma (*dříve Československá státní norma*)
ERD	Entity-Relationship Diagram (*Entitně-relační diagram*)
ERP	Enterprise Resource Planning (*Plánování podnikových zdrojů*)
GDPR	General Data Protection Regulation
GNU GPL	GNU General Public License
IDE	Integrated Development Environment (*Vývojové prostředí*)
IT	Informační technologie
JIT	Just In Time
JSON	JavaScript Object Notation (*JavaScriptový objektový zápis*)
JWT	JSON Web Token
MVC	Model View Controller
OCR	Optical Character Recognition (*Optické rozpoznávání znaků*)
ORM	Object–relational mapping (*Objektově relační mapování*)
PaaS	Platform-as-a-Service (*Platforma jako služba*)
PHP	PHP: Hypertext Preprocessor (*rekurzivní zkratka*)
REST	Representational State Transfer
SaaS	Software-as-a-Service (*Software jako služba*)
SOAP	Simple Object Access Protocol
SQL	Structured Query Language (*Strukturovaný dotazovací jazyk*)
UI	User interface (*Uživatelské rozhraní*)
WS	Web Services (*Webové služby*)

ÚVOD

Žijeme ve velmi progresivní době, rychlost pokroku, kterou tak dnes prožíváme, je enormní a ne nadarmo se tak říká, že co je dnes čisté sci-fi, zítra se může stát realitou. Tato skutečnost dnes už neplatí pouze pro vědu a výzkum, podobně na tom jsou i různé společnosti a v přední řadě jejich zákazníci, kteří stále intenzivněji a v nekonečné smyčce prahnou po spotřebě rozmanitého a neustále inovativnějšího zboží a služeb. V tomto důsledku spotřebitelské chování klade na společnosti nezanedbatelný nátlak, který je nutí rozšiřovat své kapacity, tak aby chrlily mnohem více, byť nekvalitních, produktů a zvládly obsloužit větší množství zákazníků, a to vše ideálně za kratší nebo alespoň za stejnou jednotku času. Mnoho společností tak investuje nemalé finanční obnosy do rozšiřování produkce formou stavby nových průmyslových hal.

Jedinou cestou, jak rozšířit výrobu, progresivně zvýšit příjmy a minimalizovat provozní náklady za předpokladu zachování současné výrobní plochy (bez nutnosti budovat nové výrobní prostory) je právě zavedení optimalizace podnikových procesů. Tento tlak je s námi již mnoho let a je zde příhodné citovat pana profesora Václava Řepu, který již v roce 2006 v prvním vydání své knihy, *Podnikové procesy: procesní řízení a modelování*, psal: „*Zlepšování podnikových procesů je dnes holou nezbytností pro udržení firmy na trhu. Během uplynulých dvaceti let se již stalo zvykem, alespoň ve zdravějších ekonomikách, že podniky, nuceny svými zákazníky, kteří žádají stále lepší produkty a služby, soustavně uvažují o zlepšení svých procesů. Pokud totiž zákazník nedostane, co žádá, má možnost se obrátit na mnoho konkurenčních firem.*“ [1]

V každém podniku existuje nepřeberné množství procesů, které se dále dle oblastí užití dělí na mnoho rozličných kategorií. V této diplomové práci tak bude věnována pozornost primárně procesům zprostředkovávajícím rozhodování a komunikaci uvnitř i vně společnosti. Lze uvést například komunikaci mezi jednotlivými odděleními, s obchodními partnery, či schvalování nového vývoje.

Aby bylo možné dosahovat efektivity všech procesů, tak aby nedocházelo ke:

- zbytečnému zahlcování konkrétního uzlu,
- pomalé distribuci potřebných informací všem zainteresovaným subjektům,
- ztrácení předešlého, ale i aktuálního průběhu (tedy částí, ale i celé historie),
- zamlčování faktů některým uzlům,

musí být nasazeno dostatečně kvalitní řešení, umožnující co nejpružněji pracovat s potřebami každého jednotlivého procesu. Vytvořením systému, jenž umožní tvorbu propracovaných postupů i s řešením jejich konsekvencí a plně automatickým průchodem, je tak možné minimalizovat celkovou náročnost komunikace při rozhodování. Ta se tak stává organizovanou, plně synchronní, okamžitou a hlavě úplnou.

V současné době (rok 2021) již nemalá část podniků ví, že bez optimálních procesů (nejen rozhodovacích) není možné dosahovat ideálních výsledků. Tato práce tak dává náhled do jedné z možností, jak zvyšovat efektivnost a v patřičné situaci dokonce snížit dodatečné náklady spojené s návrhem a implementací vlastního proprietárního řešení.

Další motivací, tentokrát silně spojenou s autorem práce, je prohloubení jeho znalostí návrhu a implementace business informačních systémů.

Rozborem a přímou optimalizací podnikových procesů se dnes zabývá mnoho odborných prací i článků, nicméně jen málo z nich řeší přímý návrh a implementaci univerzálního systému, do kterého by bylo možné přesunout notnou část podnikových procesů, spočívajících v komunikaci, výměně informací a rozhodování.

Cílem této diplomové práce se tak stává vytvoření příjemného a intuitivního webového prostředí, které bude splňovat požadavky pro obsloužení i komplexních rozhodovacích procesů. Implementací takového systému do podnikového prostředí by mělo řízenou cestou dojít k zjednodušení rozhodování a předávání informací při komunikaci, čímž by se zvedla i její efektivnost.

Každá komunikace, v jejímž závěru je očekáváno patřičné rozhodnutí nad konkrétní věcí, v sobě skýtá jednotlivé kroky, ve kterých probíhá. Celý informační systém by tak měl umožňovat separování každého komunikačního procesu na zmíněné kroky. Při každém z nich musí dojít k delegování akcí pouze na zainteresované subjekty. Tato definice je však poněkud složitá na pochopení, a proto si žádá obšírnější vysvětlení na drobném příkladu *postupu návrhu designu světlometu pro nový model automobilu*:

Hlavní pracovník starající se o celkový design automobilu vytvoří nový rozhodovací proces a přiloží k němu, ve formě souborů, všechny parametry, které od výsledného návrhu očekává. Proces následně odešle ke zpracování. Ten bude bezprostředně poté doručen k vedoucímu pracovníkovi designérské pracovní skupiny, která navrhne nový světlomet dle dodaných specifikací. Po dokončení vedoucí přiloží veškeré návrhy ke schvalování a předá rozhodování na další subjekt v pořadí. Tím je opět hlavní designer,

na kterém nyní je rozhodnutí, zda mu design vyhovuje, či nikoliv. V případě, že ne, bude návrh připomínkovat a udělí své zamítavé stanovisko. Proces se tím navrátí zpět k designérské skupině, která design předělá podle přiložených podnětů. V opačném případě, kdy se mu design líbí, udělí souhlasné stanovisko a proces může být ukončen jako schválený. Není nutné obšírněji popisovat, co vše se může dále dít.

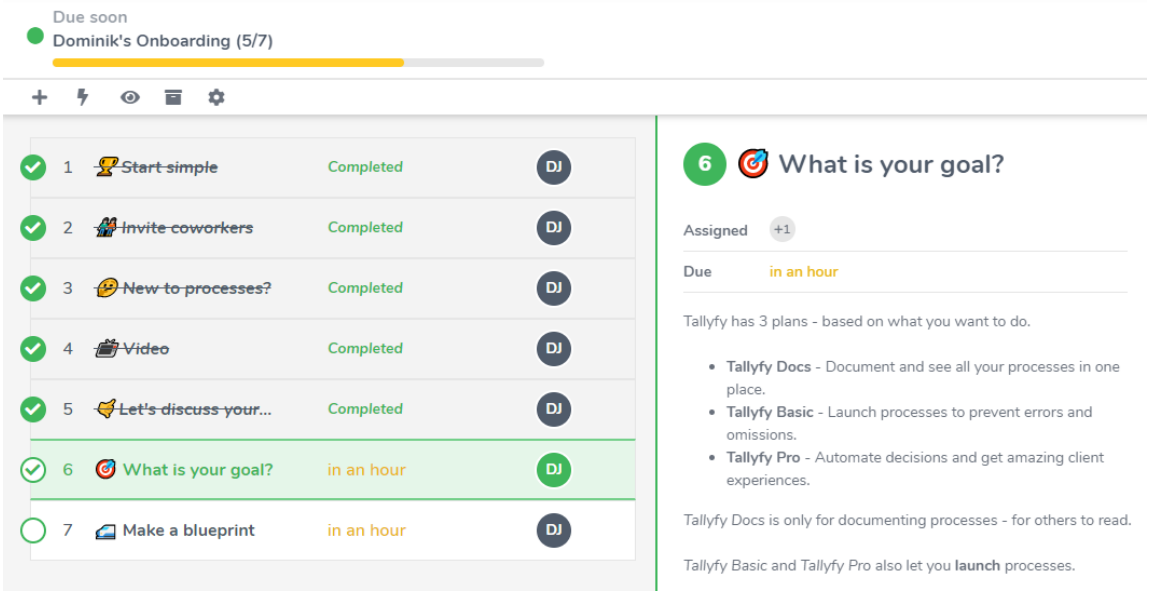
Z předchozího příkladu je snadno patrné, že dochází k přehlednému a plynulému delegování akcí právě na subjekty, které jsou na řadě. Každý krok je také charakterizován patřičnou akcí skupiny uživatelů systému (jeden nebo více), kteří chronologicky plní předem definovaný komunikační scénář a společně se tak podílí na finální podobě výstupu.

1 ALTERNATIVNÍ NÁSTROJE

Na trhu existuje nepřeberné množství alternativních, již hotových, nástrojů, které umožňují definování podnikového workflow k dosahování procesní optimalizace. V této kapitole je tedy představeno několik zajímavých nástrojů, které těmto potřebám plně vyhovují.

1.1 Tallyfy

Velmi zajímavým nástrojem pro automatizaci je nástroj Tallyfy, který obsahuje spoustu zajímavých funkcí zabalených v ladné a intuitivní grafice (obrázek 1). Při práci s podnikovými procesy uživatele provede jak tvorbou celé procedury, tak i následně jejím spuštěním a vyřízením.



Obrázek 1 – Workflow v Tallyfy [2]

Procesy převedené do systému se nazývají procedurami, které tvoří jen část celé funkcionality. Dále stojí za to uvést například možnost zakládat a řídit úkoly, popřípadě i publikovat jak interní, tak veřejné formuláře. Samozřejmostí je tvorba šablon a vestavěný archiv. Každá procedura je tvořena jednotlivými kroky, Tallyfy jim říká úkoly [2], kdy je možné u každého z nich definovat rozličné a hlavně žádané parametry.

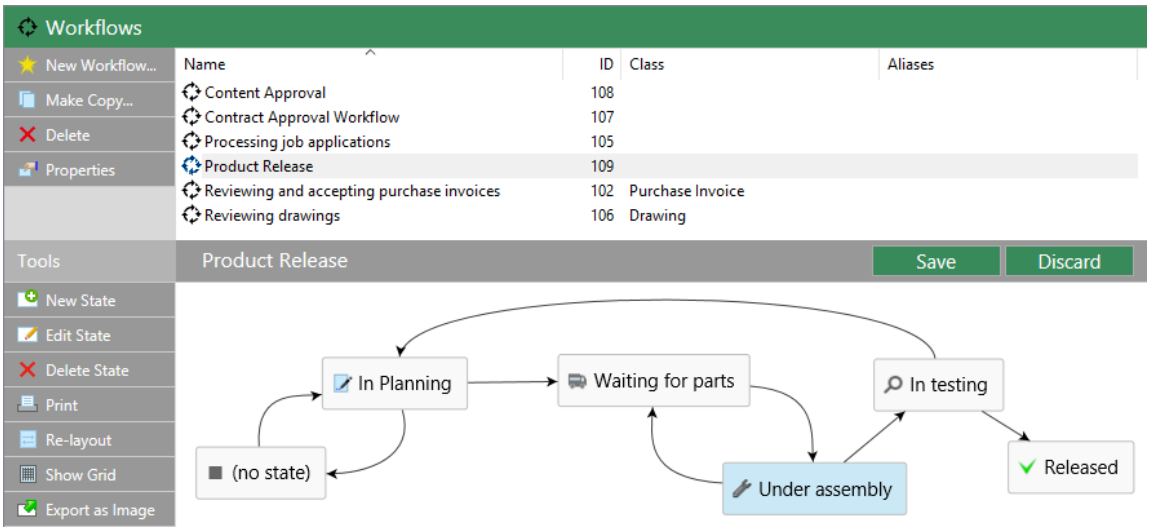
Není tak žádným překvapením, že je možné úkolu přiřadit jednoho nebo více uživatelů nebo též definovat jeho termín dokončení.

Co je však na systému zajímavé, je právě jeho možnost nastavovat interní podmínky každému z úkolů procedury, které tak dokáží dynamicky zobrazovat nebo skrývat jiné, následující kroky, v závislosti na předchozím stavu. Dále možnost vytváření jedinečných formulářů pro sběr dat v každé úkolu a v neposlední řadě i pokročilé nastavení každého kroku procedury, které tak umožní například automatické spuštěni jiné procedury, čekání na schválení všemi uživateli, nebo automaticky provolat webhook po úspěšném dokončení dílčího úkolu. [2]

Tallyfy má své nesporné přednosti a řeší podstatnou část problému této práce, avšak nemá například vnitřní integraci pro automatické odbavení e-mailové schránky nebo striktnější řízení průběhu procedury s automatickým návratem a opakováním v případě vrácení s připomínkami.

1.2 M-Files

Dalším nástrojem pro automatizaci je M-Files. Jedná se o nástroj primárně určený k automatizaci toku dokumentů ve firemním prostředí, který umožňuje tvorbu opravdu propracovaných a detailních scénářů. Jednou z velkých předností systému je integrovaný systém správy obsahu využívající prostředí Microsoft Dynamics 365 a Microsoft 365 [3].



Obrázek 2 – Workflow v M-Files [3]

Stejně jako u Tallyfy je systém poměrně robustní a obsahuje mnoho funkcí, bohužel však v nepříliš intuitivním designu (obrázek 2), který je místy poněkud jednodušší, zastaralý a nepřehledný. Obsahuje plánování a klade patřičný důraz na enterprise bezpečnost dat. Na úkor designu může být určitou přidanou hodnotou dostupnost automatizovaného vytěžování digitalizovaných dokumentů, které prostřednictvím optického rozpoznání znaků (OCR) umožní udělat z obyčejných oskenovaných PDF dokumentů dokumenty, s nimiž lze plně pracovat za pomoci různých pravidel a procesů automatizace. Snadno a jednoduše tak dokáže plně automaticky vytěžovat ohromné množství informací právě i z původně tištěných dokumentů, kdy uživatel nemusí ztrácet čas zbytečným přepisováním důležitých metadat. [3]

Systém, jak je již z názvu patrné, je z velké míry specializovaný právě na automatizaci práce s již existujícími firemními dokumenty. Pokud je tedy žádané vytvořit řízený scénář (pracovní postup), jehož výstupem nemusí být žádný dokument, ale pouze konsenzus určité skupiny uživatelů v dané věci, tento systém tak nemusí být přesně to, co by zákazník v dané situaci dokázal ocenit.

1.3 Process Street

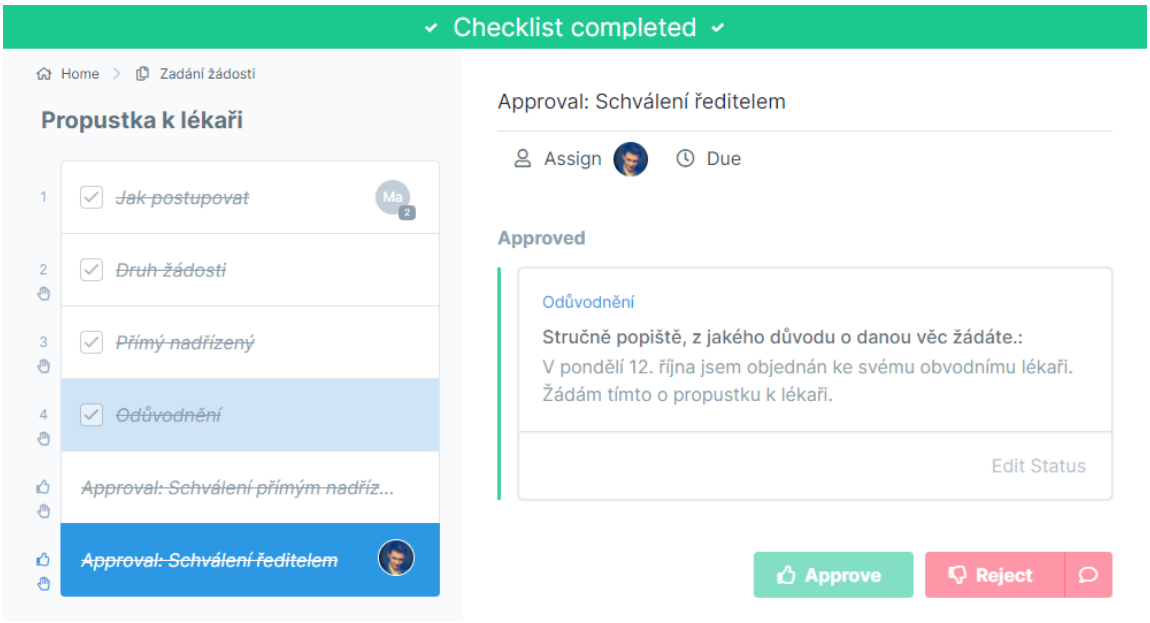
Process Street je nástroj zaměřující se čistě na procesní a workflow management. Jednou z jeho hlavních funkcí je tvorba automatizovaných scénářů, které mají možnost vytvářet vlastní formulářové vstupy, s jejichž daty lze následně dynamicky pracovat a pružně tak měnit budoucí postup a chování celého procesu.

Celý systém je do značné míry podobný již dříve představenému Tallyfy (str. 17), významným rozdílem je například absence správy jednotlivých úkolů (mohou být pouze v rámci procesu) nebo přítomnost knihovny šablon s více než osmi sty předdefinovanými procesy (například z oblasti financí, lidských zdrojů, IT, marketingu, nemovitostí a dalších).

Vítanou vlastností je možnost vložení formulářového prvku, který umožní nahrání souboru (i více) a jeho postoupení do dalších fází procesu. Vzhled celého systému je velice příjemný, moderní a adekvátně minimalistický. [4]

Jak je patrné z obrázku 3, každý proces je členěný na jednotlivé úlohy, kdy každý z nich může být různého druhu. U některých stačí kliknout na tlačítko „Next“ a pokračovat

dále, u jiných je nutné vyplnit formulář nebo například potvrdit či zamítnout postup. Samozřejmostí je přikládání komentářů ke každému kroku procesu.



Obrázek 3 – Workflow v Process Street. [4]

I pro platformu Process Street platí velmi podobné nedostatky, jako má Tallyfy. Není tedy například dostupné zpracování e-mailové schránky, dále ačkoliv je dostupný návrat ve vykonávání jednotlivých kroků procesu, není automatizován. Je tak nezbytně nutné ručně zrušit předchozí postup, a to až do bodu, kam je nutné se vrátit.

1.4 SAP

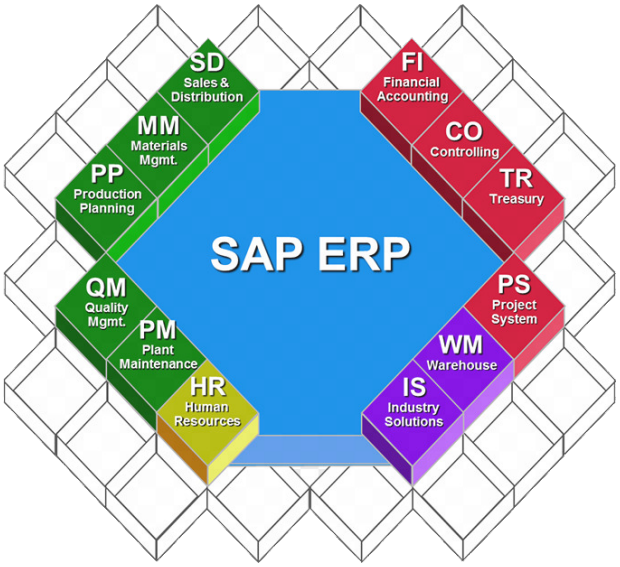
Jedním z dnes nejrozšířenějších systémů pro zpracování podnikových dat a procesů je SAP. Jedná se o rozsáhlý systém, který poskytuje velmi široké spektrum podnikových služeb. Je uzavřený a pro vnější komunikaci zpřístupňuje Web Services (WS) rozhraní, postavené na technologiích SOAP a REST. Skrze ně je možné docílit integrace libovolných externích aplikací, jako jsou ty dodavatelské či odběratelské, ale i další.

Jádrem WS je aplikační server SAP NetWeaver Aplication Server ABAP, který poskytuje konfigurační rozhraní použitelné jak pro tvorbu poskytovatele, tak i konzumenta WS. Jeho prostřednictvím je možné libovolné služby snadno vytvářet, konfigurovat, publikovat i konzumovat, tedy dosahovat plné integrace s všemi potřebnými externími systémy.

Hlavním prvkem celého systému je ERP, které je dále rozšířeno jednotlivými moduly. Z jejich velkého množství je příhodné zmínit například:

- FI – finanční účetnictví,
- SD – prodej a distribuce,
- PP – plánování a výroba,
- QM – řízení kvality,
- HR – řízení lidských zdrojů,
- RE – evidence a správa nemovitostí,
- PS – řízení projektů,
- WF – workflow.

Velice zajímavý modul, právě v souvislosti s touto prací, je Business Workflow (WF). Jedná se o jednu z klíčových komponent celého systému, která přidává rozšíření o automatizaci podnikových procesů. Díky tomu, že je velice žádaná, je běžně dodávána společně s jádrem ERP. Není proto vizualizována na diagramu základních modulů (obrázek 4).



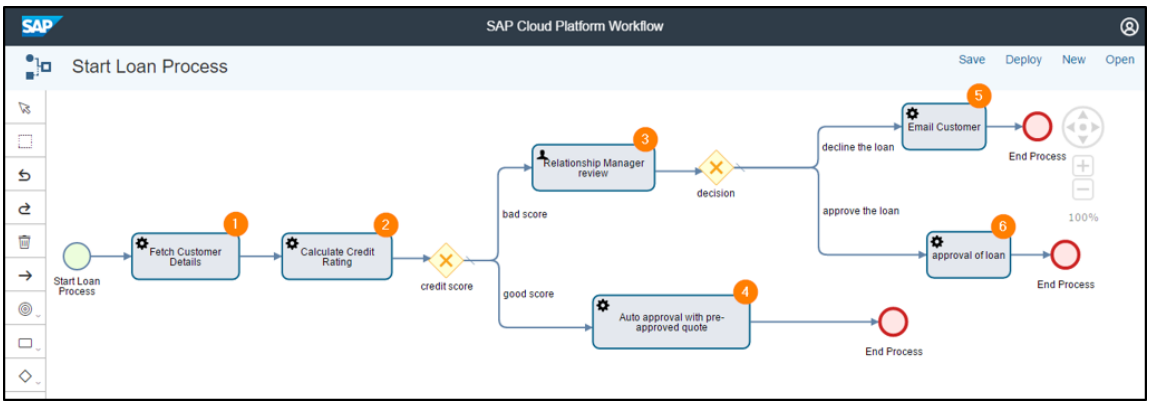
Obrázek 4 – Základní moduly SAP [5]

Tvorba scénářů je v prostředí SAP velmi komplexní disciplínou. Procesy se skládají z posloupnosti kroků, úkolů a nebo událostí. Lze uspořádat a řídit závislosti úloh. Uživatelům lze zasílat oznámení o úkolech, které čekají na schválení, akci nebo mohou být informováni o milnících v procesu.

Pomocí SAP WF lze snadno definovat obchodní procesy, které ještě nejsou součástí systému SAP (některé jsou dodávány jako předkonfigurované, záleží však na dostupných modulech). Může se jednat o jednoduché uvolňovací nebo schvalovací postupy nebo i o složitější obchodní procesy. Pro již stávající je dostupná možnost analýzy a statistik.

Pracovní postup je možné spustit ručně nebo automaticky jako reakci na nějakou událost. Též je umožněno pracovat s již stávajícími funkcemi systému nebo je rozšiřovat, popřípadě vytvářet nové. Modul WF nemusí být vždy nutně součástí systému, při integraci však vždy plně přebírá kontrolu nad všemi existujícími podnikovými procesy.

Jedním z hlavních pilířů a dnes již praktickou samozřejmostí je cloudová platforma SAP Business Technology Platform, která vznikla transformací ze starší technologie SAP Cloud Platform. Je tak umožněn vznik služeb, skrze které mají uživatelé k dispozici cloudové prostředí pro vývoj, správu, rozšiřování a poskytování aplikací. K náhledu je toto prostředí na obrázku 5. Nasazením této služby vstoupil SAP do moderní éry cloudových multiplatformích aplikací. Cizí však této platformě není ani blockchain, strojové učení, internet věcí nebo umělá inteligence. [6]



Obrázek 5 – Business Workflow v SAP [6]

Díky tomu, že je platforma opravdu široká, řeší mnoho aspektů podnikového prostředí a je plně rozšiřitelná, umožňuje tak vybudovat kompletní podnikový systém. Je to však také jeden z důvodů, proč není příliš zajímavá a přívětivá právě pro malé podniky balancující lehce nad hranicí životaschopnosti. Jeho integrace do podnikového prostředí je poměrně náročná a též i finančně nákladná. Ve všech ostatních aspektech však převyšuje požadavky na systém stanovený touto diplomovou prací.

1.5 Shrnutí

Jednotlivé výše popsané produkty nabízejí velmi zajímavé funkce. Ty se ne vždy točí jen kolem správy a oběhu dokumentů. Společnost SAP při svém vývoji zašla až tak daleko, že nabízí prakticky kompletní firemní řešení pro široké spektrum agend. Za tyto služby

si však nechá velmi dobře zaplatit. Podobné řešení nabízí i M-Files, které se soustředí primárně na spolupráci a práci s dokumenty. Produkty Tallyfy a Process Street jsou navzájem velmi podobné. Soustředí se na tvorbu propracovaných scénářů sloužících nejen pro oběh dokumentů. Aby bylo shrnutí kompletní, sluší se popsat i systém navrhovaný v této práci, zvaný jako DokFlow[1], který se snaží poskytovat vhodnou alternativu.

Přehledné porovnání hlavních kritérií jednotlivých služeb je dostupné v tabulce číslo 1.

	Tallyfy	**M-Files**	**Process.st**	**SAP**	**DokFlow**
Min. interval fakturace	rok	-	měsíc	měsíc	-
Cena za uživatele měsíčně	od 4,2 $ do 25 $	nezávazná kalkulace	od 12,5 $ do 25 $, custom	dle modulů, od 99 $ do 255 $[2]	-
Forma dodání	cloud	cloud, on premises	cloud	cloud, on premises	cloud, on premises
Počet plánů	3	-	2	-	1
Výběr funkcí	dle plánů	dle modulů	dle plánů, custom	dle modulů	dle plánů
Jednorázová licence	ne	ne	ne	ano 1600 - 3213 $	ano
Minimum uživatelů	10	1	1	1	1
Náročnost implementace	snadná	střední	snadná	složitá	snadná
Platforma	web/cloud	web, cloud desktop Win, Android, iOS	web/cloud	web, cloud desktop Win, Android, iOS	web/cloud
API	ano	ano	ano	ano	ano
Správa e-mailu	ne	ne	ne	ano	ano

Tabulka 1 – Porovnání jednotlivých produktů

V porovnání s touto prací jsou všechny prezentované systémy robustnější a poskytují větší spektrum odladěných funkcí. Tyto funkce jsou buď příliš široké, navyšují[3] tak cenu nebo neposkytují řešení odbavení přes e-mailovou schránku. Dalším, ne příliš průhledným aspektem je financování. Konkrétně financování produktů SAP, kdy implementace do podnikového prostředí může vyjít až na desítky milionů korun. M-Files bude levnější.

Tato práce se tak snaží vytvořit systém, který částečně využívá možností již výše popsaných produktů a ty rozšiřuje o možnost plně automatizované komunikace a odbavení skrze e-mailovou komunikaci.

[1] Autorův pracovní název pro vytvářený informační systém.

[2] Licence jsou velmi specifické a nezahrnují, mnohdy nemalé, náklady na implementaci.

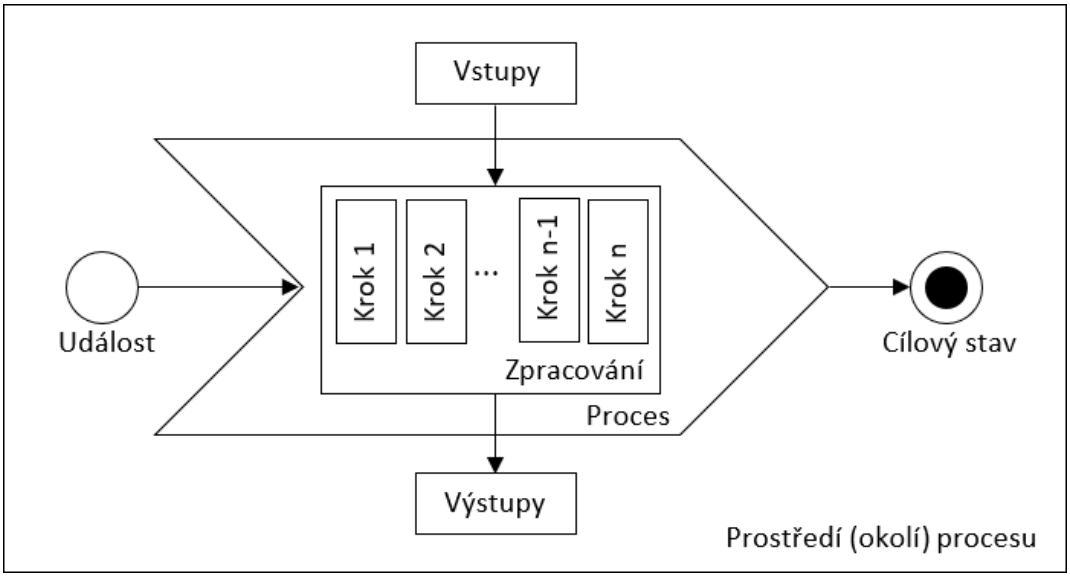
[3] Dodáno je více, než uživatel potřebuje.

2 PODNIKOVÉ PROCESY

Než se bude možné detailně podívat na to, co je to vlastně podnikový proces a jak probíhá jeho optimalizace, je nezbytně nutné čtenáře seznámit se základní obecnou definicí procesu.

2.1 Proces

Proces je možné charakterizovat jakožto ucelenou posloupnost předem definovaných činností, které v reakci na specifikovanou událost zpracovávají vstupní zdroje (peníze, čas, energii, materiál aj.) a v konečném počtu kroků produkují výstupy, jež jsou cílovým stavem vyžadovány.



Obrázek 6 – Vizualizace obecného procesu, zdroj: vlastní

Pro snadnější pochopení definice procesu není třeba chodit příliš daleko. Z obrázku číslo 6 je patrné, že se jedná o posloupnost elementárních (základních) kroků, která přijme vyžadované vstupy a předem definovaným postupem je převede na očekávané výstupy.

Takový výsledek jednoho procesu může být následně použit jakožto očekávaný vstup procesu jiného. Tímto způsobem lze procesy donekonečna spojovat a větvit, čímž může dojít k vytvoření neuvěřitelně komplexních postupů, jejichž výstupem může být bezesporu tak složitá věc, jakou je dopravní letadlo určené k přepravě osob i nákladu.

Existuje však celá řada dalších definic procesu. Zde je seznam těch nejpoužívanějších:

- Norma ČSN EN ISO 9001 proces definuje jako: „soubor vzájemně působících činností, který přeměňuje vstupy na výstupy“. [7]
- Hammer, M., Champy, J. ve své knize *Reengineering the Corporation* píší, že: „proces je soubor činností, který vyžaduje jeden nebo více druhů vstupů a tvoří výstup, který má pro zákazníka hodnotu“. [8]
- Pan profesor Václav Řepa použil definici procesu jakožto: „souhrn činností transformujících souhrn vstupů na souhrn výstupů (zboží nebo služeb) pro jiné lidi nebo procesy, používajíce k tomu lidi nebo nástroje“. [1]

Všechny výše zmíněné definice v čistém důsledku říkají jedno a to samé pouze jinými slovy. Záleží tak jen na úhlu pohledu, kterým daný autor na problém nahlížel.

2.1.1 Podnikový proces

Jedním z velmi často skloňovaných pojmů je dnes právě business process, česky zvaný jako podnikový proces. Jedná se o celkem nový pojem, který se do popředí dostal právě díky intenzivnímu nástupu informačních technologií společně s implementací komplexních ERP systémů (plánování podnikových zdrojů) do podnikového prostředí.

Podnikové procesy se postupem času staly základem pro nově vznikající odvětví procesního řízení, což vedlo k nutným změnám uvnitř struktury podniků, tak aby bylo dosaženo vyšší konkurenceschopnosti a udržení se na trhu. [1]

Každá organizace je tvořena nepřeberným množstvím druhů procesů, které na sebe navazují a interagují. Je na ně možné nahlížet z celé řady úhlů. Jedním takovým je rozdělení podle významu pro plnění cílů organizace:

- Hlavní podnikové procesy vytváří výstupy, které jsou bezprostředně spojeny s uspokojováním potřeb zákazníků. Mají tedy zásadní podíl na hodnotě finálního produktu a tedy i na výkonnosti a kvalitě celého podniku, např. řízení výroby automobilu. [9]
- Podpůrné procesy probíhají uvnitř podniku a slouží jako podpora hlavních procesů, např. zásobování materiálem. [9]
- Řídící procesy definují organizaci firmy a administrativní aktivity, které plánují a řídí vše ostatní. [9]

Další rozdělení je podle jejich vztahu k subjektům, které do nich vstupují nebo jsou jimi ovlivňovány:

- Interní procesy probíhají v rámci jednoho podniku, popřípadě pouze jeho dílčích organizačních jednotek. [9]
- Externí procesy zahrnují vztahy podniku k externím subjektům, které překračují hranice podniku; jsou částečně realizovány i u dodavatelů, spolupracujících firem nebo přímo u konečného zákazníka. [9]

Pokud jsou nejen hlavní procesy dobře nastaveny, mění dodané vstupy na peníze. Při fungování podniku také vznikají určité režijní náklady. Aby mohl podnik dosáhnout zisku (tedy kladného výsledku hospodaření za dané účetní období), musí být jeho procesy dostatečně optimalizovány. Je tak nezbytně nutné, aby ze strany vedení probíhala správná organizace a řízení toku práce. To podléhá procesnímu managementu.

Díky tomu, že management svou činnost vykonává efektivně, je možné dosáhnout stavu, kdy jsou provozní náklady nižší než příjmy a podnik se tak dostává do již výše zmíněného provozního zisku. A to je právě hlavní cíl základního podnikového procesu (to však neplatí například u neziskových společností).

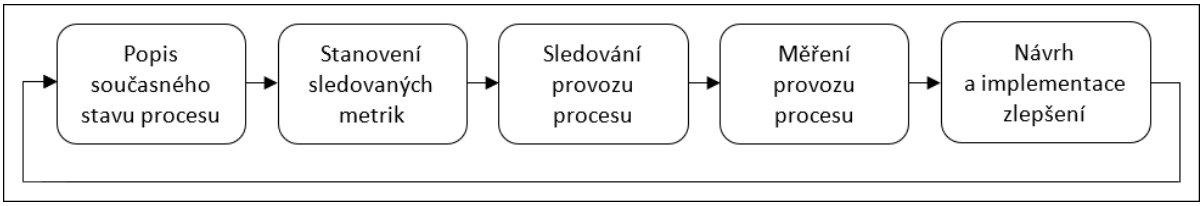
2.2 Optimalizace procesů

Cílem téměř každého podniku je dosahování zisku. Aby to však bylo možné, musí procesní management procesy optimalizovat dostatečně kvalitně, tak aby vyhovovaly aktuálním zájmům spotřebitelů a posouvaly se kupředu i vůči konkurenci. Obvykle se stává, že společnosti své procesy spíše komplikují. Má na to určitý vliv jak samotný management, tak například i státní legislativa. Je tak nezbytně nutné, aby byla nalezena střední (ideální) cesta mezi jednoduchostí a nároky managementu/legislativy.

2.2.1 Průběžné zlepšování

Ve chvíli, kdy se podnik rozhodne pro optimalizaci, musí si nejprve stanovit své cíle a postupy, jak chce postupovat a čeho hodlá dosáhnout. Většina firem začne optimalizaci tou úplně nejtriviálnější, evoluční metodou. Jedná se o metodu průběžného (přírůstkového) zlepšování. Ta spočívá v porozumění a měření stávajících procesů, kdy ze sesbíraných dat přirozeně vyplynou podněty ke zlepšení.

Na obrázku 7 je popsán celý postup metody. Jeho základem je detailní popis současného fungování procesu, za nímž následuje stanovení základních ukazatelů, plynoucích převážně z potřeb zákazníků. Soustavným sledováním běhu celého procesu jsou identifikovány příležitosti k jeho zlepšení, které jsou následně implementovány. Provedené změny je též nutné zdokumentovat, čímž se celý proces optimalizace dostane opět na začátek. Tuto metodu tak lze díky nekonečnému opakování nazývat průběžným (soustavným) zlepšováním podnikových procesů. [1]



Obrázek 7 – Diagram průběžného zlepšování procesu [1]

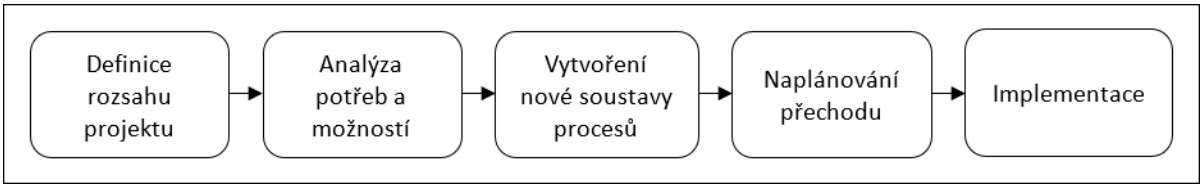
Od počátku devadesátých let se do popředí začaly dostávat moderní technologie, které měly za následek, že podniky potřebovaly akcelerovat tempo zlepšování svých procesů. Jednou z těchto technologií byl postupný příchod internetu na světové trhy, který přinesl revoluční, do té doby neslýchané možnosti. [1] Postupem času zasáhl téměř všechny odvětví lidské činnosti a umožnil rapidní zvýšení jejich efektivity (včetně práce). To však nebyla jediná technologie, která se začala stále výrazněji prosazovat nejen do světa businessu. Další takovou byla právě robotizace, která v určité podobě vznikla již v 50. letech 20. století [10], ale svůj hlavní boom zažívá až od počátku tohoto tisicíletí[4] do současnosti (a zdaleka nedosáhla stropu svých možností).

2.2.2 Reengineering podnikových procesů

Aby bylo možné takto progresivní technologie začlenit do stávajících procesů, cesta evolučního, tedy postupného vývoje se stává tou první, která jde do popředí, ale je také tou, která ve většině případů vede do slepé uličky. Důvodem nemožnosti užití této metody je právě samotná implementovaná technologie. Stávající procesy jsou stavěny na úplně jiném základu, a i když by do určité míry mohlo dojít k jejich uzpůsobení, nikdy nebudou tak efektivní jako ty upravené metodou reengineeringu (BPR). [11]

[4] 3. tisíciletí

BPR je radikální metoda, která bere celý proces a od jeho základu ho mění. V jednoduchosti by se dalo říci, že v extrémním případě považuje celý proces za nevyhovující, vytváří proces úplně nový, který má stejné, nebo lepší výstupy, ovšem technologie a postupy používané při jeho průběhu se diametrálně liší.



Obrázek 8 – Diagram reengineeringu podnikového procesu [1]

Reengineering je na rozdíl od průběžného zlepšování jednocestný (obrázek 8). Neexistuje tak nic jako jeho „postupné zlepšování“. Na začátku projektu stojí opět otázka, co má být výstupem reengineeringu. Definuje se tedy cílový stav, kterého se má dosáhnout. Následně se pokračuje důkladnou a detailní analýzou. Poté je již možné vytvořit budoucí podobu nových, optimálních procesů. Poté, co jsou všechny nové procesy připraveny, je možné naplánovat samotný přechod, který bývá největším „oříškem“ [1]. Pokud byly všechny kroky provedeny správně a v dostatečné kvalitně, je velmi pravděpodobné, že dojde k dosažení kýžených výsledků.

Ve chvíli, kdy se vedení podniku rozhodne pro optimalizaci procesů pomocí BPR, znamená to, že výrazně zasáhne do stávajícího fungování společnosti. Po samotné implementaci změn pak obvykle nějakou chvíli trvá, než se jednotliví pracovníci s novými postupy sžijí a přijmou je za své. To platí dvojnásob, pokud bylo vedení odvážné a rozhodlo se provádět velmi radikální změny. Tím se sice podaří dosáhnout mnohem lepších hodnot efektivnosti, nicméně pro mnoho podniků je tato cesta velice riziková, neboť v prvotních fázích může, a často i bude efektivita klesat pod nerentabilní hranici. Managementu tak může hrozit, že procesy nebyly nastaveny dostatečně kvalitně a významně tak ohrozí budoucí pozici společnosti na trhu.

Náročnost a do určité míry i konečná nepředvídatelnost je jedním z hlavních důvodů, proč se převážně větší společnosti do optimalizací touto metodou příliš nehrnou. Volí buď pozvolnou cestu nebo budování nových závodů (nebo divizí) postavených již kompletně na nové technologii. [1]

Po úspěšném provedení reengineeringu by mělo vedení na nějaký čas sáhnout k průběžné metodě zlepšování, čímž stabilizuje celý proces a připraví ho pro budoucí radikální optimalizace.

2.2.3 Rozdíl mezi BPR a průběžným zlepšováním

Oba přístupy k optimalizaci jsou v procesním managementu velmi významné a nezanedbatelné, je však nutné dávat si velký pozor, jaké mohou vzniknout konsekvence po jejich aplikaci.

Mezi nejvýznamnější rozdíly obou přístupů, které stojí za zmínku, patří: [1]

- Rizikovost – zatímco průběžné zlepšování s sebou nenese významnější rizika, BPR může v případě nedostatečně kvalitního návrhu vést k fatálnímu konci.
- Počátek – u průběžné optimalizace je počátečním bodem již fungující proces, který se pouze vylepšuje. U BPR není na začátku nic více než záměr, jehož cílem je vybudovat celý proces znovu a efektivněji.
- Náročnost – se odvíjí právě od rizikovosti. Čím větší zásah je do procesu proveden, tím je pravděpodobnější, že se zvýší i náročnost. Z toho důvodu jsou optimalizace prostřednictvím BPR hodnoceny jako jedny z nejsložitějších.
- Časová složitost – je opět faktor, který záleží primárně na náročnosti. Je tak patrné, že BPR bývá časově velmi náročný.
- Frekvence – zastává možnost cyklického zlepšování. BPR je však metoda, která si stanový svůj cíl a po jeho dosažení již není možné jakkoliv pokračovat. To však neplatí pro průběžné zlepšování, které dokáže provádět jak jednorázové změny, tak i dlouhodobá sledování a optimalizace.
- Rozsah – bývá prvním kritériem, které je při analýze hodnoceno. Zatímco průběžné zlepšování dokáže efektivně analyzovat pouze omezenou oblast, BPR zvládne provést klidně změnu všech existujících procesů současně[5].

2.3 Digitalizace a metadata

S rozšířením moderních výpočetních technologií se začalo stále častěji přistupovat k digitalizaci některých podnikových procesů. Umožnilo to společnostem mnohem

[5]Je ale otázkou, zda-li by tak vůbec někdo chtěl postupovat.

jednodušší správu své vlastní agendy společně se zlepšením výkonnosti a efektivnosti činností. Digitalizace jako taková však stále v řadě firem není tématem číslo jedna a její zaměstnanci si vystačí pouze se systémem papír-tužka, který je sice roky prověřený a funguje celkem dobře, není však příliš efektivní, co se týče rychlosti předávání partikulárních změn všem zainteresovaným subjektům. Otázkou tedy v dnešním světě není, zda začít digitalizovat. Každá z firem, která se bude v rostoucím konkurenčním prostředí chtít na trhu udržet, tak bude muset chtě nechtě digitalizaci dříve či později podstoupit[6].

Moderní podnikové nástroje tvořené progresivními ERP systémy mají významný vliv při budování vnitropodnikové infrastruktury. Takové systémy doplněné o vyladěné funkce umělé inteligence pak dokáží data nejen velice efektivně sbírat, kategorizovat a zpracovávat, ale i hledat efektivnější a zajímavější postupy. Aby toho bylo možné dosahovat, je nezbytně nutné, aby systém využíval takzvaná metadata.

Metadata jsou sestavená nebo uspořádaná *data o datech*. Dělí se na mnoho typů, avšak pro tuto práci jsou nejzajímavější právě technická a business metadata. Zatímco technická metadata jsou získávána automaticky z atributů zdrojů, jako je e-mail nebo soubor, a poskytují informace o formátu a struktuře dat podle potřeby výpočetních systémů, business metadata definují kontext dat v rámci organizace. Je jimi například vlastnictví dat, jejich podrobnější klasifikace[7] nebo vztahy mezi jednotlivými business objekty. Metadata se tak stávají nepostradatelnou součástí automatizovaných systémů a slouží pro snadnější práci se samotnými daty. Velmi triviálními příklady pro jejich použití je například kategorizace a vyhledávání konkretních dat. [12]

S rozmachem cloudových služeb se i podnikové ERP systémy stále častěji začínají dodávat jakožto softwarové prostřednictvím cloudu (SaaS) [13]. Společnostem je tímto způsobem značně uleveno od budování složité IT infrastuktury, na které jejich *on premise* řešení poběží. K této možnosti se mnohdy přiklání spíše menší až střední společnosti. Cloudová řešení spolu však přináší i určitá rizika. Jelikož se jedná o řešení provozované na serverech dodavatelské společnosti, jsou vyváděna kritická data společnosti mimo ni. Zvyšuje se tak riziko pro zneužití těchto vysoce citlivých a soukromých dat. Z toho důvodu tato řešení především konzervativnější společnosti příliš často nevyužívají.

[6]Existuje jen velmi málo společnosti, kterým se to podaří.

[7]hledání dat

3 PŘEHLED TECHNOLOGIÍ

Při zahájení každého projektu v IT je v první řadě nezbytně nutné zvolit ty správné technologie, nástroje a architektury, které dokáží utvořit ideální prostředí pro vznik celého projektu. Spojením všech zvolených vývojových „nástrojů“, ať už jde o vývojová prostředí (IDE), návrhové vzory (development design patterns), frameworky či celé architektury, společně s dobře promyšleným návrhem aplikace[8] je možné dosáhnout výsledného produktu, který dává pro spotřebitele smysl a má pro něho významnou přidanou hodnotu.

Následující podkapitoly se zabývají popisem použitých technologií, které jsou vhodné pro vznik webového informačního systému popisovaného v této práci.

3.1 PHP 7.4

Někdo mu říká pouze programovací jazyk, ve skutečnosti se jedná o skriptovací programovací jazyk. PHP je dnes velmi populární hlavně pro tvorbu webových stránek. Pro takové stránky je charakteristická dynamika obsahu, to znamená, že PHP umí poskytovat obsah, který se mění dle preferencí každého návštěvníka, ale současně i v čase. Dále se jedná primárně o serverový jazyk, což s sebou přináší velká pozitiva. Každý požadavek o obsah, který je na server odeslán, musí být nejprve zpracován a jeho výstup je následně poslán do návštěvníkova prohlížeče. Tím je možné zajistit určitou úroveň bezpečnosti a práci jak s hesly, tak i citlivými údaji a daty jiných uživatelů, aby nedošlo k jejich vyzrazení. [14]

PHP je také velmi oblíbené pro svou nízkou náročnost na pochopení[9] základních operací a má strmou rychlost učení. Distribuován je pod *PHP licencí*, která je velmi svobodná a benevolentní. Nedílnou součástí tohoto jazyka je nepřeberné množství rozšiřujících knihoven. Za zmínku, v souvislosti s touto prací, stojí převážně tyto:

- IMAP – přidávající možnosti přímé správy e-mailového serveru,
- SMTP – odesílání e-mailů prostřednictvím systémového poštovního serveru,
- PDO_MYSQL – pro práci s nejen relačními[10] databázemi,

[8] V případě této práce se jedná o webovou aplikaci.

[9] U větších projektů, společně s použitím mnoha knihoven, náročnost stoupá.

[10] Pro ostatní druhy databází existují další obdobné knihovny.

- SODIUM – obsahující moderní a bezpečné kryptografické algoritmy.

Jazyk PHP vychází z osvědčené a rozšířené syntaxe jazyků C a Java. Je oběma velice podobný. Na první pohled asi nejvýraznějším rozdílem je zápis názvů proměnných, které v PHP vždy začínají symbolem dolaru (např. `$summary`). Každá proměnná je také dynamicky typovaná, což znamená, že její typ je vázán na hodnotu, která se v průběhu běhu může mnohokrát změnit (společně s typem). Do verze 7.0 (vydání prosinec 2015) v jazyce absentovala typová kontrola úplně a byly dostupné pouze základní systémové funkce pro její omezenou kontrolu. Od verze 7.0 byla přidána základní typová kontrola (argumenty funkcí a návratové typy) a v aktuální verzi 8.0 (2021) byla typová kontrola rozšířena o další možnosti.[15] Stále však například není možné vytvořit netřídní typovanou proměnnou `int $summary = 5;`. Verze 8.0 současně s rozšířenou typovou kontrolou přinesla i nový kompilační *just in time* (JIT) engine, který v ideálním případě dosahuje až trojnásobného zrychlení zpracování požadavku (průměrně však 1,5 – 2 krát) [16].

Další zajímavostí jazyka jsou pole, která ve skutečnosti tvoří hashovací tabulka (hodnota je v nich vázána na klíč) a současně se drží i záznamy o pořadí jednotlivých prvků. Tato vlastnost PHP polí může být pro začínající programátory, kteří mají zkušenosti spíše s konvenčními silně typovanými jazyky, jako je právě C/C++, C# nebo Java, poněkud matoucí.

Jazyk PHP má mnoho nesporných konkurenčních výhod:

- obsahuje mnoho funkcí distribuovaných společně se základní knihovnou,
- společně s vývojem vzniká poměrně kvalitní a obsáhlá dokumentace,
- má nativní podporu velkého množství databázových systémů,
- pro publikování je dostupných mnoho hostingových služeb.

Každý systém má i svá negativa a ta neminula ani PHP:

- neudržuje kontext aplikace (dochází ke zpomalení),
- mnoho funkcí má nekonzistentní pojmenování,
- ladící nástroj není triviální doinstalovat,
- je možné se setkat s nejednotným pořadím argumentů funkcí.

Největším a nejznámějším hráčem na trhu, který svoji platformu staví na upravené[11] variantě jazyka PHP, je společnost *Facebook Inc.*

[11] Používá *HipHop for PHP*, čímž transformuje kód v PHP do optimalizovaného kódu v C++.

3.2 Symfony 5

Jedná se o velmi výkonný webový aplikační framework napsaný v jazyce PHP, který je open-source a vydávaný pod licencí MIT. Je tvořen sadou znovupoužitelných PHP komponent, kde každá je samostatně fungujícím balíkem specifických funkcí. Dalo by se říci drobnou knihovnou. Tato rozšíření se do projektu instalují prostřednictvím nástroje Composer, který slouží ke správě balíčků a plně řeší závislostí. [17]

Symfony je založeno na standardním vývojovém návrhovém vzoru MVC. Ten rozděluje řídící logiku od uživatelského rozhraní a dat. Skládá se ze tří základních úrovní [18]:

- Model – centrální komponenta zodpovědná za logiku a správu dat,
- View – šablona čekající na data určená k doplnění do finálního výstupu,
- Controller – definuje možnosti, jak uživatelské rozhraní reaguje na vstupy, tedy přijímá požadavky, které dále distribuuje jako příkazy vrstvám Model a View.

Velmi užitečnou a v mnoha projektech takřka nepostradatelnou komponentou pro práci s databází je balík s názvem Doctrine. Rozšiřuje funkčnost frameworku o objektově relační mapování (ORM), které konvertuje data mezi relační databází a objektově orientovaným jazykem. Tím je zajištěno, že s daty z relační databáze je možné pracovat v kódu jako s objekty. [17]

Hlavní myšlenkou frameworku je urychlit vývoj a údržbu webových aplikací a zbavit tak programátora nutnosti vytvářet pro každou webovou aplikaci stejné nebo podobné jádro, které vždy musí implementovat nutné základní rozhraní (například MVC, ORM, autorizaci a další). Symfony na to jde ještě trochu jinak, aby projekt neobsahoval naprosto zbytečné balíky funkcí, které nejsou používány. Stažený základní skelet nového projektu obsahuje jen několik základních balíků. Ty se starají o možnost automatizované instalace nových rozšíření a také o tzv. bootstrapping. Ten se postará o zavedení všech instalovaných komponent do paměti tak, aby se daly v projektu používat.

Symfony je vhodné pro velké a komplexní projekty, protože dokáže zpracovat rozsáhlý kód velmi čistým způsobem. To je jeden z hlavních důvodů, proč většinou podniky dávají přednost Symfony před konkurenčním Laravelem. Symfony také poskytuje větší modularitu a flexibilitu, neboť obsahuje znovupoužitelné komponenty, čímž je výrazně usnadněno vylepšování a škálování složitých webových aplikací. [19]

Další výhodou je dostupnost opravdu velkého množství[12] rozšiřujících balíků, které pochází z komunitního vývoje. To vývojářům umožňuje úzce se soustředit na jejich problém, bez nutnosti vytvářet veškerou funkcionalitu samostatně.

Síla a efektivita celého frameworku však spočívá v automatické „předkompilaci“ zdrojového kódu. Ten je při prvním průchodu transformován na vysoce optimalizovaný kód (opět v PHP) a je také při každém dalším požadavku prováděn. Ve chvíli, kdy dojde k nalezení změny v původním neoptimalizovaném zdrojovém kódu, pak je nutné znovu vygenerovat jeho optimalizovanou variantu. To je jeden z hlavních důvodů, který pomáhá tomuto vysoce optimalizovanému frameworku dosahovat velmi krátké odezvy při zpracování požadavků. S kompilovanými jazyky se však rovnat nemůže.

3.3 API Platfrom

Je balík rozšiřujících funkcí pro Symfony a také jeden z nejdůležitějších balíků, který byl použit při vývoji informačního systému popisovaného v této práci. To je také hlavní důvod, proč si vysloužil tuto krátkou podkapitolu.

Jedná se o poměrně rozsáhlý balík, který do projektu přináší možnost vybudování kompletního *RESTfull* a *hypermediálního* API nebo také rozhraní *GraphQL*. Je plně optimalizován pro požadavky Symfony a umí automaticky generovat dokumentaci aplikačního rozhraní v grafických rozhraních Swagger a OpenAPI. Celá platforma také podporuje nejpoužívanější otevřené webové standardy, jako je OpenAPI, RDF/Json-LD/Hydra, GraphQL, JSON:API, HAL, OAuth... [20]

Velkou nevýhodou pro někoho může být celkem komplikované publikování čistého a minimálního JSON API bez přítomnosti Hydra tagů. Pro tyto případy je nutné upravit jak normalizační, tak denormalizační komponenty.

3.4 JavaScript

S tím, jak se začaly rozšiřovat webové technologie, začal růst zájem o možnost vykonávat kód i na straně samotných klientů, a to bez přímé kontroly serveru. To byla první myšlenka, která stála za vznikem jazyka zvaného JavaScript.

[12] Více než 8 500

Je to multiplatformní, objektově orientovaný, skriptovací, programovací jazyk, který je dnes nejčastěji interpretovaný právě na straně webového prohlížeče a rozšiřuje tak dynamické možnosti již webových stránek. V poslední době se do popředí dostávají i různé jiné implementace, fungující pouze na serverové straně (např. Node.js), které tak tvoří silnou konkurenci a alternativu například pro PHP backend. Tím je umožněno vzniku tzv. full-stacku (frontend + backend) běžícímu pouze na JavaScriptu. Stejně jako PHP a většina ostatních interpretovaných jazyků má dynamické typování proměnných závislé na jejich hodnotě. Tuto nevýhodu se snaží řešit různé knihovny (např. PropTypes), popřípadě alternativní implementace (např. TypeScript).

Hlavní podstatou jazyka je jeho událostmi řízený přístup. To mu umožňuje v reálném čase reagovat na události, které byly vyvolány například uživatelem. Díky tomu je JavaScript schopný zajistit plynulý uživatelský zážitek, avšak plně na úkor spotřeby výkonu výpočetního zařízení, na kterém je spuštěno. Syntaxe jazyka vychází stejně jako u PHP z rodiny C/C++ a Java.

Oproti výše zmíněným jazykům má JavaScript jednu výraznou zvláštnost. Každý objekt je v paměti reprezentován jakožto asociativní pole. Navenek se tak tváří současně jako objekt, ale i jako právě zmíněné asociativní pole. Z toho vychází možnost dvojího (zaměnitelného) přístupu k jeho atributům. To je možné buď přes klíče asociativního pole nebo přímo přes názvy konkrétních atributů. Z objektů jazyka také vznikl dnes velmi populární formát pro přenos dat zvaný JSON (JavaScript Object Notation). [21]

Jelikož možnosti JavaScriptu jsou opravdu rozsáhlé a veškerá jeho data jsou uložena v klientském počítači, musela být vyřešena i patřičná bezpečnostní rizika. Tím hlavním je zneužití všech dat v lokálním souborovém systému počítače. Tento problém byl vyřešen plným zákazem práce se souborovým systémem na discích uživatele. Se soubory je možné pracovat pouze v případě, že je uživatel přes formulářový prvek ručně nahrál do prostředí webové stránky. I přes toto výrazné omezení nejsou eliminována naprosto všechna bezpečnostní rizika. Data mohou být zneužita v případě výskytu kritických chyb v prohlížeči, operačním systému či samotném hardwaru.

Dále je to zvýšené riziko zneužití citlivých dat webové stránky uložených přímo v paměti prohlížeče. Tento problém už musí řešit programátor sám. Průběžným mazáním již nepotřebných dat se musí postarat o to, aby veškerá citlivá data nebyla v paměti držena zbytečně dlouho.

Jak by již mělo být čtenáři patrné, jazyk má velmi blízko k HTML a CSS a slouží k nim jako doplněk.

3.5 React.js

Snaha o lépe udržovatelný kód vedla tak daleko, až jednoho dne Facebook přišel s knihovnou React. Ta celou funkcionalitu webové aplikace rozkládá na jednotlivé komponenty, kdy každá má na starosti pouze elementární funkce. Postupným vnořováním a spoluprací jednotlivých komponent je pak zajištěn vznik webové multiplatformní aplikace, která běží čistě na straně klienta.

React je JavaScriptová open-source knihovna, která je aktivně vyvíjena nadnárodní společností Facebook a komunitou nezávislých přispěvatelů. Je vhodná pro dynamické weby a je tedy hojně využívána pro tvorbu rozsáhlých a interaktivních uživatelských rozhraní. Jednou z jejích hlavních předností je běh pouze na straně klienta.

Pro svou funkci využívá React speciálního rozhraní, které slouží pro popis struktury HTML a XML dokumentů. Toto rozhraní zvané jako Document Object Model, ve zkratce DOM, je určeno k přímému popisu struktury stránky. Zpracování změn v této struktuře je obvykle celkem náročné na čas i systémové zdroje, neboť musí dojít k mnohonásobnému překreslení. Proto React přišel s tzv. virtuálním DOMem, který drží kopii toho původního. Veškeré změny jsou provedeny ve virtuální kopii a až následně přeneseny do reálného DOMu. Tím je ušetřeno mnoho drahocenných zdrojů a stránka se zdá být plynulejší. [22]

3.6 RESTfull API

Representational state transfer, ve zkratce jen REST, sice není technologií v pravém slova smyslu, jedná se spíše o architekturu. Ta je dnes de facto standardem pro distribuovaná prostředí webových služeb (Web Services). Velmi často se s ní setkáme u interaktivních webových aplikací a z toho důvodu si našla místo v tomto krátkém výčtu.

Každá webová služba poskytující REST API by měla vždy a bezpodmínečně zachovávat bezstavovost (statelessness). To však neznamená, že nemohou být na serveru uchována dynamická data. K plnému pochopení celého problému je nutné si uvědomit, že existuje vedle stavu aplikace i konkrétní stav každého zdroje (dat). Tento stav je stavem zdroje

v určitém bodě času. Jedná se tedy pouze o data, jež klient získá po úspěšném zaslání a vyřízení požadavku. Nemají však nic společného s interakcí mezi klientem a serverem.

Druhým, tentokrát podstatnějším stavem je stav samotné aplikace. U tohoto stavu server ukládá informace, které slouží k identifikaci příchozích klientských požadavků a uchování detailů o nich.

Tím, že je server bezstavový (stateless), je tedy myšleno, že není závislý na aplikačním stavu a nepotřebuje ke správnému fungování uchovávat data klientských relací.

Fungování je založeno na krátkých textových požadavcích, které výstižně popisují, jaké se mají vykonat akce. Současně mohou předávat data ve formátu JSON, jsou očištěna od zbytečných informací. Drží tak nejmenší možnou strukturu, jež je ještě dostačující k úplnému popisu kontextu dat.

Použití bezstavového REST API má své nesporné výhody [23]:

- Každý požadavek, který server přijme, může být odbaven nezávisle na ostatních.
- Je možné distribuovat jednotlivé požadavky na různé servery, a to například pomocí „load balancingu“, sloužícího pro vyvažování zátěže hardwarových i softwarových prostředků.
- Odstraněním veškeré logiky určené pro synchronizaci aplikačních stavů se stává REST API méně komplexním a tedy snadnějším na použití.
- V mnoha případech není potřebné, aby byla data zpracovávána při každém dotazu. Již dříve připravená data mohou být snadno uložena v cache paměti a poskytnuta, kdykoliv je to možné. Tím, že se data cachují, lze obsloužit mnohem více klientů, u nichž se často opakují stejné požadavky. Dále se také snižuje odezva systému a do určité míry se zvyšuje odolnost vůči specifickým poruchám. Například v případě výpadku databází mohou být některé dotazy stále obslouženy. Tento aspekt bezstavového API však silně závisí na povaze konkrétních dat.
- Server nikdy neztratí přehled o tom, v jakém stavu se každý klient aktuálně nachází, neboť jsou v každém požadavku zasílány kompletní informace, jež jsou nutné pro správné vyřízení žádosti.

3.7 MySQL

Kvalitní databáze je základem každého většího projektu. Jedná se o systém, který je zodpovědný za efektivní a konzistentní ukládání dat. MySQL a z něho odvozené

databázové systémy jsou jedny z nejčastěji užívaných databázových systémů při tvorbě webových stránek za pomocí jazyka PHP [24]. Od svého počátku jsou primárně optimalizovány pro rychlost. Z toho důvodu v některých oblastech trpí nedostatky. Příkladem takového nedostatku je implementace jen základních nástrojů určených pro práci s daty a také většinou nemají propracovaný systém zálohování.

Jde o relační typy databází, což znamená, že data jsou ukládána ve struktuře jednotlivých tabulek, které jsou tvořeny řádky a sloupci. Řádky je možné chápat jako konkrétní záznamy a naproti tomu sloupce definují jednotlivé atributy, které je žádané v databázi uchovávat. Každá tabulka by měla obsahovat sloupec, popřípadě sloupce, s klíčem, který jednoznačně charakterizuje konkrétní záznam. Dále je také možné vytvářet atributy s vlastností cizího klíče, které mohou vytvářet vazby mezi daty. Těmto vazbám se říká relace a jsou trojího druhu: 1:1, 1:N a M:N. Co jednotlivé relace znamenají, nebude v rámci této práce rozebíráno.

MySQL (a z něho odvozené) je multiplatformní databázový systém vydávaný pod různými licencemi. Tou nejčastěji používanou je svobodná licence GNU GPL, kdy je systém poskytován bez úplaty. Ostatní dostupné licence jsou již pouze komerčního rázu a tedy placené. [25]

3.8 Bootstrap

Obsahuje sadu zpracovaných kaskádových stylů, která je připravena pro efektivní tvorbu webu a webového designu obecně. Ke svému plnému fungování vyžaduje JavaScriptovou knihovnu jQuery.

Obsahuje styly například pro:

- tlačítka,
- skupiny tlačítek,
- informační zprávy,
- breadcrumb navigaci,
- navigační menu,
- stránkování,
- přepínatelné záložky,
- formuláře,
- rozbalovací pole s nabídkou.

Výhodou Bootstrapu oproti standardnímu vývoji grafického designu je předpřipravená sada již hotových grafických komponent a snadno použitelný systém pro práci s rozložením stránky. Ten umožňuje prvky různě vertikálně a horizontálně organizovat a skládat. Též je

možné každému prvku říci, jakou šířku svého rodiče má zabírat. Celková šířka je 12 sloupců a značí 100% šířky rodiče. Této hodnoty by tak měl dosáhnou součet všech prvků na horizontální linii. Pokud bude součet větší, dojde k poskládání některých koncových prvků pod ty první. Dále umožňuje rozsáhlou práci s typografií. Jeho možnosti jsou však mnohem širší a pro jejich popis v této práci není dostatek místa.

Za zmínku dále stojí, že se jedná o otevřený software, jehož vývoj byl iniciován společností Twitter. Dnes je jím vyvíjen společně s širokou komunitou.

3.9 Heroku

Heroku je moderní cloudová platforma poskytovaná jako služba (PaaS) a hlavně rychlý způsob, jak elegantně řešit continuous integration (CI) a continuous deployment (CD).

Poskytuje tři možné metody nasazení. Tou první je odeslání zdrojového kódu se základní konfigurací přímo z lokálního počítače, a to za pomoci nástroje Heroku CLI. Tato metoda je následně použita i druhou, která poskytuje obdobné nasazení aplikací, tentokrát však založených na technologii Docker. Poslední metoda nabízí napojení na velmi populární verzovací službu GitHub, kdy dochází k nasazení plně automaticky po každém commitu na zvolených větvích. Společně tyto metody poskytují rozhraní pro plnou automatizaci v případě nasazovaní vývojových, testovacích, ale i produkčních verzí softwaru. Nasazený software pak běží ve virtuálních kontejnerech na službě AWS, jež je poskytována společností Amazon.

V současné chvíli Heroku oficiálně podporuje 8 programovacích jazyků. Za zmínku stojí například Node.js, PHP, Java nebo Python. [26]

3.10 Vercel

Stejně jako Heroku, tak i Vercel je cloudová služba poskytující rychlý způsob CI/CD. Hlavní odlišností však je podpora pouze frontendových aplikací používajících Node.js. Při požadavku klienta se pouze postará, aby mu byl předán kompletní zkompilovaný zdrojový kód frontendové aplikace, a svůj běh končí.

Oproti Heroku poskytuje výrazně intuitivnější a uživatelsky příjemnější rozhraní.

4 ANALÝZA STRUKTURY ŘEŠENÍ

Analýza řešení podrobně rozebírá teoretickou strukturu celé navrhované aplikace, možnosti designu a doručení k uživatelům. Přináší také zdůvodnění technologií vybraných pro vývoj.

4.1 Struktura aplikace

Nejprve je nutné definovat, jak bude samotná aplikace fungovat. Jelikož je tento projekt poměrně rozsáhlý, bylo potřeba dopředu důkladně promyslet i jeho přesnou strukturu.

4.1.1 Jeden systém

Mnoho webových prezentací je postaveno na technologii, ve které se pro uživatele webu stává server hlavním partnerem. Řeší správu všech jeho požadavků a neustále spolu komunikují. Uživatel zasílá požadavky a data a na ně mu server odpovídá kompletním zdrojovým kódem vyžadované webové stránky. Klientský prohlížeč má tak za úkol pouze tuto stránku vykreslit.

Tento přístup klade velký důraz na server a ten tak musí stíhat u velmi frekventovaných webů obsloužit nemalé množství požadavků. Společně s tím, že musí ještě na závěr každého takového požadavku generovat zdrojový kód, může docházet v nárazových situacích k jeho přetěžování. Tento problém může řešit kompletní oddělení Uživatelského rozhraní (UI) a přenechání jeho generování klientskému prohlížeči.

4.1.2 Oddělení uživatelského rozhraní

Samostatné frontendové aplikace se dnes dostávají stále více do popředí. Umožňují plně oddělit logiku správy dat od prezentační vrstvy. Aby toho bylo možné docílit, musí vzniknout v podstatě dvě nezávislé aplikace, které spolu asynchronně komunikují prostřednictvím domluveného RESTového API. Tato komunikace probíhá formou jednoduchých HTTP požadavků. Serveru je tak uleveno od zbytečného generování zdrojových kódů a pouze zpracovává data, která buď získává dle požadavku z databáze a předává je k dalšímu zpracování grafické komponentě, popřípadě přijatá data ukládá do databáze. Tyto základní požadavky umí obsloužit velmi rychle. Obvykle v souvislosti

s touto praktikou programátor narazí na pojem AJAX. Jedná se o zkratku *Asynchronous JavaScript And XML*, což je název pro technologii, která zprostředkovává výše zmíněnou asynchroní komunikaci se serverem [27].

V případě, že je nutné vybudovat více aplikací pro různé platformy (web, desktop, mobil, ...), stačí tak kompletně vytvořit pouze jednu z nich. Ostatní pak již mají dostupné backendové komunikační rozhraní v rámci první aplikace, nad kterým se již pouze vybuduje nové UI jiné platformy. Tato cesta se tak jeví velmi zajímavou.

4.1.3 Zvolení struktury

Obě dvě výše popsaná řešení mají své klady i zápory. Výhodou spojení backendové části s frontendovou a přenechání generování kódu na server je to, že značně urychlí vývoj a sníží jeho náklady. Společně s tím, že tento projekt nepředpokládá multiplatformní rozvoj ani nasazení více grafických rozhraní, jeví se první varianta jako ta ideální.

Do hry však vstupuje i motivace autora projektu, který je v rámci procvičení a zdokonalení programátorských dovedností rozhodnut vytvořit tento informační systém prostřednictvím dvou oddělených aplikací.

4.2 Výběr správných technologií

Pro správný návrh aplikace je také nutné zvolit i správné technologie. Seznam s jejich detailním popisem je možné najít výše v kapitole 3. Tato podkapitola se nadále bude detailněji zabývat pouze důvody, proč byly k vypracování této práce použity právě ony výše zmíněné technologie.

4.2.1 Serverová část

Technologií pro použití na serverové straně (backend) je opravdu mnoho. Nejčastěji skloňovanou je dnes Microsoft .NET framework. Součástí tohoto frameworku, psaného v jazyce C#, je i technologie zvaná ASP.NET, jež je dnes velmi často používána pro tvorbu webových stránek. Nejen, že podporuje tvorbu konvenčních webů, u kterých probíhá generování kódu na straně serveru, současně má i podporu pro tvorbu kompletních REST API řešení. Jednou z nepopiratelných výhod platformy .NET a konkrétně jazyka C# je jeho kompilovaná podoba do bytekódu. Díky tomu je ušetřeno

mnoho cenného času při zpracovávání požadavků. Těší se tak opravdu velké oblibě, převážně na velmi vytížených projektech s velkou datovou propustností. Jeho hlavní nevýhodou však je, že pro svůj běh potřebuje speciální server zvaný IIS, který je poměrně finančně náročný na pořízení. V současnosti (2021) již existuje oficiální řešení, jak IIS nasadit a projekty zprovoznit na Linuxovém serveru zcela zdarma. Není to však rozšířené řešení a z důvodu potřeby hostování tohoto projektu prostřednictvím některé online služby je to pro použití na tomto projektu naprosto nevhodné.

Další zajímavou alternativou, která stojí za krátkou zmínku, je framework SpringBoot vytvořený v jazyce Java. Jedná se o moderní, kompilované prostředí, které má velmi podobné vlastnosti jako platforma .NET.

Ideální backendovou technologií pro použití na tomto projektu se zdá být Symfony. Jedná se o backendový aplikační framework psaný v PHP, který je primárně stavěný pro tvorbu MVC aplikací. Prostřednictvím rozšiřujících balíčků je však možné jeho funkce rozšířit o tvorbu RESTových aplikačních rozhraní. Jedním z hlavních důvodu použití právě této technologie jsou autorovy hlubší znalosti a zkušenosti právě s tímto frameworkem.

4.2.2 Klientská část

Pro tvorbu webových uživatelských rozhraní, které běží samostatně na straně klienta (frontend), také existuje nepřeberné množství různých frameworků a knihoven. Většina z nich je psána v jazyce JavaScript. Jimi jsou například Vue.js, React.js nebo AngularJS. Existují však i řešení postavená na jiných technologiích. Jedním takovým je Blazor od společnosti Microsoft, který běží v dnes již nativním prostředí WebAssembly a pro svůj běh využívá jazyka C#. [28] Není však zatím příliš rozšířen.

Nejčastěji používanou technologií pro budování samostatných moderních webových rozhraní je React. Ten rozkládá celou aplikaci na jednotlivé elementární provázané komponenty. Tento přístup umožňuje snadný vývoj aplikace. Výhodou je snadná údržba a přehlednost kódu. Pokud se vezme v potaz, že s touto technologií má rozsáhlé zkušenosti i autor, jeví se opět pro použití na tomto projektu jako nejvhodnější.

4.2.3 Databáze

Na trhu je dnes také nepřeberné množství různých databázových systému, které mají různé vlastnosti a funkce. Je možné zmínit například objektové, relační či NoSQL.

NoSQL jsou nerelační databázové systémy, které pro svou funkci nevyžadují jazyk SQL. Název je trochu matoucí, znamená *Not Only SQL*, tedy „nejen SQL“, nikoliv na první pohled zřejmé *No SQL*, což by znamenalo „žádné SQL“. Pro ukládání strukturovaných dat tak nespoléhají na jazyk SQL a využívají k tomu jiné technologie a programovací jazyky. Mají velmi dobrou horizontální[13] a vertikální[14] škálovatelnost. Díky odlišnému principu ukládání dat se tyto databáze stávají zajímavými řešeními pro specifické projekty, například ze segmentu big data. Pro vývoj běžných aplikací dnes nejsou příliš typické z důvodu neplnohodnotné podpory transakčního modelu ACID (atomicita, konzistence, izolovanost, trvanlivost) [29].

Další možností jsou objektové databáze, které, jak je z názvu patrné, reprezentují data ve formě objektů. Tento typ databází je tak nejvhodnější k použití s objektově orientovanými jazyky, které tak mohou svou strukturu dat přímo uložit do databáze bez jakýchkoliv zbytečných konverzí. Ačkoliv je jazyk PHP objektově orientován, velmi často se společně s ním používají právě relační databáze doplněné o objektově relační mapování (ORM). Framework Symfony pro své fungování standardně používá relační databáze a veškerou podporu poskytuje v rámci nich. Z toho důvodu se objektové databáze neuvažují k použití.

V současné době jsou jedním z nejpoužívanějších databázových systémů relační databáze. [24] Data jsou v nich reprezentována prostřednictvím tabulek. Pro to, aby bylo možné snadno ukládat data z objektově orientovaných jazyků, které jsou dnes dominantními při vývoji webových aplikací, do relačních databází, vznikla technika ORM. Ta si vzala za cíl vytvořit rozhraní, které se postará o automatickou konverzi dat mezi objektově orientovaným jazykem a relační databází. Tím bylo umožněno propojení dvou nejrozšířenějších a nejpoužívanějších technologií. Relační databáze je tak ideální pro použití v tomto projektu.

[13]Zvýšení kapacity HW a SH přidáním dalších samostatných jednotek, například serverů do clustru, které fungují společně.

[14]Zvýšení kapacity HW a SH přidáním dalších zdrojů v rámci konkrétní jednotky. Například přidáním RAM pamětí disků nebo procesorů.

4.3 Doručení aplikace

Jedním z kritických parametrů, který je nutné u každé budované aplikace stanovit, je i počet uživatelů, kteří tento systém mohou v jednu chvíli využívat. Pokud je žádané zvládat obsloužit tisíce až miliony uživatelů, je nutné dostatečně kvalitně integrovat proces takzvaného *Application Delivery* (AD), česky řečeno „doručování aplikace".

Jedná se o využívání technologií, které se starají o to, aby byly cloudové aplikace a služby spolehlivě dostupné velkému počtu uživatelů. Tyto služby musí obvykle v krátkém čase zvládat odbavovat nemalé množství požadavků. O to, aby vše spolehlivě fungovalo, se stará řadič *Application Delivery Controller* (ADC), který poskytuje služby pokročilého *Load Balancingu*. Ten při své činnosti distribuuje jednotlivé požadavky na konkrétní stroje v rámci celé skupiny serverů. Každý z těchto výkonných strojů je nezávislý a s ostatními běžně sdílí například databázi. Prostřednictvím navenek vystupujícího Load Balanceru poskytuje každý z těchto serverů stejnou službu. Load Balancer tak může požadavky distribuovat k odbavení tím způsobem, aby nedocházelo k zahlcování pouze některých strojů. Současně je vyřešena i situace, kdy jeden ze serverů spadne nebo je na něm potřeba provést údržba. Jeho úlohu v clusteru převezmou ostatní servery. Uživatelům se takový systém jeví jako celek a je v jednu chvíli schopen obsluhovat tisíce či miliony uživatelů, záleží čistě na počtu a výkonu jednotlivých serverů. Je vhodné si uvědomit, že rozšiřování výkonu není nikterak náročné. [30]

Tento projekt nepředpokládá velmi vysokou vytíženost, i tak tato technologie může být vítaným pomocníkem, jak dosahovat maximálního výkonu. Veškeré tyto služby pro *Application Delivery* je schopna dodat cloudová platforma Heroku.

4.4 Uživatelské rozhraní

Velmi častým problémem, se kterým se mnoho aplikací potýká, je nevhodný design uživatelského rozhraní (UI). To však nutně nemusí znamenat, že je špatný. Ve skutečnosti pouze stačí, aby byl mírně odlišný od běžných standardů, na které jsou uživatelé zvyklí, a ti se v něm následně začnou ztrácet. Potřebují pak mnohem více času, aby se s takovým systémem naučili pracovat. Aby k tomu nedocházelo, je potřeba věnovat dostatek času studiu zažitých zvyklostí a návrhu správného layoutu stránky. Ten definuje umístění a velikosti jednotlivých grafických komponent na obrazovce. To

vše záleží na zařízeních, která budou uživatelé používat, a také na jejich kulturních zvyklostech.

U nás ve střední Evropě jsme zvyklí tyto UI prvky organizovat od horního levého rohu směrem do pravého dolního, závisí to tedy na směru, kterým se zapisuje naše písmo. Ve světě však existuje mnoho různých kultur. Zajímavým příkladem je písmo arabské, které se zapisuje od pravé strany směrem k té levé. I s tímto by měl dobrý design počítat (pokud bude nutné zobrazovat obsah v arabštině).

Mnoho běžných uživatelů se neřídí při používání aplikací manuály. Podle svých předchozích znalostí a zkušeností se snaží vždy najít co nejrychlejší a nejefektivnější cestu, která je dovede k cíli. Steve Krug v roce 2014 ve své knize *Don't make me think* napsal: „*We don't read pages. We scan them.*“ [31], což by se dalo přeložit jako: „*My stránky nečteme. Skenujeme je.*“ A měl naprostou pravdu. Ve chvíli, kdy se uživatel snaží získat konkrétní informaci, popřípadě se dostat na konkrétní obrazovku aplikace či webové stránky, tak její obsah pouze skenuje a soustředí se na ty části, které jsou pro něj alespoň trochu zajímavé.

Když se takovému uživateli nedaří dosáhnout svého cíle, zběsile kliká na vše možné ve snaze posunout se dále. Obvykle tak narazí na nedostatky aplikací, které UI návrhář nebo programátor neočekával a z toho důvodu ani nevěnoval příliš pozornosti jejich ošetření. Při takové situaci může v uživateli růst míra frustrace až do takové míry, že je schopen fyzicky poškozovat buď samotný počítač nebo jeho periferní zařízení. Velmi příhodné je zmínit tvrzení Jonathana Wilkinse, který ve své studii z roku 2007 popisuje, že „*75 % kancelářských pracovníků přiznává, že se někdy uchýlili k fyzickému násilí vůči svému počítači a zbylých 25 % mu pravděpodobně jen nadávalo.*“[15]

Aby k ničemu takovému nedocházelo, je nutné navrhnout rozhraní aplikace ve spolupráci s jejími budoucími uživateli, popřípadě využít služeb zkušeného UI návrháře. Spojení obou možností je však naprosto nejlepší.

4.4.1 Responsivita

Velmi podstatnou součástí dnes již každé webové aplikace je její responsivita. Jedná se o způsob vykreslování stránky, který dynamicky reaguje na velikost uživatelova displeje.

[15]Informace není jednoznačně podložená, avšak z osobní zkušenosti si autor dovolil toto tvrzení do práce zařadit.

Dříve bylo běžné navštěvovat webové stránky pouze ze stolního počítače, kde byla standardem obrazovka ve formátu 4:3 s rozlišením 1024x768 pixelů. Postupem času se toto rozlišení stále měnilo a zlepšovalo, až dosáhlo současného (2021) nejčastějšího poměru 16:9 o rozměrech 1920x1080 pixelů. V průběhu tohoto vývoje, kolem roku 2014, do toho výrazně vstoupil úplně nový druh zařízení (chytré telefony, tablety, ...), která měla naprosto odlišný formát obrazovky. [32] Tato zařízení se následně začala stále více prosazovat a kolem roku 2018 dosáhla dominantního postavení, co se týče přístupů na webové stránky. Tato skutečnost výrazně zvedla tlak na vývojáře, po kterých bylo nově žádáno, aby se weby vykreslovaly správně na různých zařízeních, tak aby byl uživateli zachován maximální komfort.

U business prostředí je to s touto problematikou trochu složitější. Většina aplikací potřebuje zobrazovat na obrazovce nemalé množství informací. Aby bylo možné docílit toho, že bude takový systém skutečně plně responzivní mezi nejpoužívanějšími druhy zařízení, vývoj takové aplikace se velice prodraží. Velmi často se pak přistupuje k alternativám, kdy se vytváří vícero nezávislých aplikací, kde jedna je určena pro velké obrazovky a druhá pro ty menší. Uživatel tak může pohodlně systém ovládat v plném rozlišení na počítači a při cestách v omezeném, avšak plnohodnotném režimu na svém mobilním zařízení.

Velmi často je tak otázkou, jak se aplikace bude používat, a v případě výhradního použití na větších zobrazovacích panelech se hledá ideální kompromis, který je ještě přijatelný pro zobrazení na mobilním zařízení. Takový uživatelský požitek však již není dokonalý. Často také vzniká rozhraní, které s mobilním zobrazením ani částečně nepočítá.

5 NÁVRH INFORMAČNÍHO SYSTÉMU

Návrh je nezanedbatelnou součástí každého vyvíjeného systému. V této kapitole tak bude čtenář seznámen s jednotlivými požadavky na aplikaci. Budou popsány jak funkční, ale i nefunkční požadavky. Pozornost bude také věnována celkové požadované bezpečnosti dat i provozu aplikace.

5.1 Funkční požadavky

Co by měl systém umět a jaké by měl mít funkce, to vše popisují funkční požadavky. Detailně tak specifikují potřeby všech uživatelů na funkce systému. Může se jednat o výpočty, zpracování dat, popřípadě i o jiné procesy.

5.1.1 Přihlášení do systému

Přihlášení do systému bude probíhat prostřednictvím formuláře, do kterého uživatel zadá své uživatelské jméno a heslo. Pokud dojde k nalezení uživatele v databázi a jeho zadané údaje se budou shodovat, dojde k přihlášení. Následně bude přesměrován na stránku uvnitř informačního systému. Pokud byl na přihlášení přesměrován ze zabezpečené stránky, bude na ni automaticky vrácen, v opačném případě, kdy přihlášení inicioval sám, kliknutím na tlačítko *Přihlásit se*, systém ho nasměruje na hlavní přehledovou stránku (dashboard).

5.1.2 Uzamčení systému

Ve chvíli, kdy uživatel potřebuje na krátkou dobu odejít od počítače, může z bezpečnostních důvodů systém uzamknout. Dojde tak k částečnému smazání uložených dat v paměti prohlížeče. Ten si nadále uchová pouze uživatelské jméno. Tlačítko *Příhlásit se* bude nahrazeno tlačítkem *Odemknout*. Po opětovném přístupu do autentizované části systému, popřípadě kliknutím na již výše zmíněné tlačítko *Odemknout*[16], bude uživatel vyzván k zadání uživatelského hesla k zapamatovanému účtu. Po odeslání hesla a zapamatovaného uživatelského jména se pokračuje, jako kdyby došlo k běžnému přihlášení (5.1.1).

[16]Při přístupu na přihlašovací stránku bude uživatel přesměrován na stránku pro odemknutí.

5.1.3 Odhlášení ze systému

Nepostradatelnou funkcí bude odhlášení uživatele ze systému. V případě, že se k tomu uživatel rozhodne, dojde k zneplatnění uživatelských autentizačních tokenů (5.2.3) pro konkrétní relaci (session) a dále budou z paměti prohlížeče odstraněna veškerá uložená data. Uživatel také bude přesměrován na hlavní stránku.

5.1.4 Správa vlastních uživatelských údajů

Každý uživatel bude moci v průběhu času v nastavení systému měnit některé své osobní údaje. Těmito údaji budou: křestní jméno, příjmení, heslo (5.1.5), e-mail (5.1.6) a avatar. Z bezpečnostních důvodů však již nebude umožněna změna uživatelského jména a systémem generovaného unikátního identifikátoru.

Funkce bude dostupná pouze po přihlášení.

5.1.5 Změna hesla

Uživatel bude mít možnost změnit si své heslo. Jeho složitost by měla být dostatečná pro zachování žádané úrovně bezpečnosti. V této verzi bude umožněno používat i slabá hesla, uživatel ovšem musí potvrdit, že tuto skutečnost bere na vědomí a souhlasí s ní.

Funkce bude dostupná pouze po přihlášení.

5.1.6 Změna e-mailové adresy

Každý uživatel bude moci změnit v nastavení svého profilu svou e-mailovou adresu. Zadá ji do připraveného formuláře, čímž se provede automatické ověření správnosti formátu. Pokud je správný, dojde k uvolnění tlačítka *Uložit*, které uživatel následně použije k odeslání žádosti. Server po jejím přijetí provede opět ověření formátu a uloží e-mail k uživateli jako neověřený. Současně s tím vygeneruje jednorázový ověřovací kód, který odešle na zadaný e-mail. Uživatel následně ve své e-mailové schránce bude muset kliknout na potvrzovací odkaz, čímž se provede samotné ověření adresy. V případě úspěchu dojde k zneplatnění jednorázového kódu a uživatel bude přesměrován na dashboard. V případě neúspěchu zůstane zobrazena zpráva o chybě.

Funkce bude dostupná pouze po přihlášení. Samotný proces ověření nevyžaduje přihlášení.

5.1.7 Blokace uživatelů

Administrátor může uzamknout uživatelský účet a znemožnit tak jeho trvalé používání. Odblokovat účet může opět pouze administrátor. Umožněno též bude dočasně zablokovat účet, kdy dojde k automatickému odblokování, a to bezprostředně po dosažení definovaného data a času.

Funkce bude dostupná pouze po přihlášení a pouze administrátorům.

5.1.8 Odhlášení na všech zařízeních

Z důvodu maximalizace bezpečnosti systému bude dostupná funkce odhlášení uživatele na všech jeho přihlášených zařízeních. Pro provedení této akce musí uživatel v nastavení systému kliknou na tlačítko *Odhlásit ze všech zařízení*, následně kliknout na ověřovací tlačítko *Potvrdit odhlášení*. Systém po provedení tohoto požadavku již nesmí odbavit žádný jiný požadavek[17], který byl zaslán s tokenem, jenž byl vygenerován před časem odhlášení.

Funkce bude dostupná i administrátorovi a pouze po přihlášení.

5.1.9 Správa skupin pro scénáře

Uživateli bude umožněno spravovat skupiny uživatelů pro tvorbu scénářů. Budou existovat tyto možnosti:

- Vytvoření skupiny – bude umožněno vytvoření nové skupiny uživatelů sloužící pro scénáře. Je nutné vyplnit název skupiny a volitelně i její popis. Dále také může být skupina nastavena jako veřejná (výchozí stav je neveřejná), čímž k ní dostanou přístup i ostatní uživatelé systému, nebudou ji však moci aktualizovat, ani odstranit. K vytvoření dojde kliknutím na tlačítko *Vytvořit* a uživatel bude přesměrován do jejího detailu, kde budou výše zmíněné údaje a též i formulář určený pro správu uživatelů skupiny (5.1.10).
- Aktualizace skupiny – prostřednictvím detailu skupiny bude moci uživatel provést aktualizaci názvu a popisu skupiny, současně se správou přiřazených uživatelů (5.1.10).

[17]Platí pouze v případě přístupu do zabezpečené části systému.

- Odstranění skupiny – uživatel může v sezmanu skupin zvolit možnost *Odstranit*, čímž ji vymaže ze systému.

Každému uživateli je umožněno, aby figuroval v neomezeném množství skupin. V každé však pouze jednou. Funkce budou dostupné pouze po přihlášení.

5.1.10 Správa uživatelů skupiny

Bude se jednat o univerzální komponentu pro správu uživatelů ve skupině, která bude použita, jak pro skupiny u scénářů (5.1.9), tak i oprávnění (5.1.17).

Formulář bude standardně zobrazovat seznam uživatelů v tabulce, kdy každý bude vypsán na vlastním řádku ve formátu `Jméno Přijmení [nickname]`[18]. Dále bude obsahovat tlačítko *Upravit*, které umožní aktualizaci přiřazených uživatelů. Samotnému návrhu principu aktualizace je ponechána volnost. Při úpravě však budou dostupná dvě tlačítka: *Uložit* – provede uložení změn a *Zrušit* – zruší změny. Po kliknutí na libovolné z obou tlačítek dojde k ukončení editace.

Funkce bude dostupná pouze po přihlášení.

5.1.11 Správa šablon scénářů

Uživatel bude moci využívat předem definovaných scénářů, připravených jinými uživateli. Pro šablony scénářů existují následující možnosti:

- Vytvoření šablony scénáře – uživateli bude umožněno vytvořit novou šablonu scénáře, kde bude moci zadat: počet kroků oběhu, popis. Dále u každého z kroků také uživatelské skupiny, *pravidlo návratu* a *pravidlo schválení*. K vytvoření dojde kliknutím na tlačítko *Vytvořit*. Návrhu je ponechána volnost.
- Aktualizace šablony scénáře – uživatel, který vlastní šablonu, bude moci upravit veškeré údaje. Aktualizace na serveru bude probíhat dynamicky s každou změnou.
- Duplikace šablony scénáře – každý uživatel může vytvořit kopii své a cizí veřejné šablony. Může tak pohodlně vytvořit editovatelnou kopii. Po vytvoření kopie bude přesměrován na detail. Nahrané soubory se neduplikují.
- Odstranění šablony scénáře – majiteli šablony bude umožněno její odstranění.
- Odeslání scénáře – po kliknutím na tlačítko *Odeslat ke schválení*, dostupné pouze v detailu, bude umožněno správně vyplněnou šablonu převést na plnohodnotný

[18] Příklad výpisu autorova účtu: `Dominik Janák [janakdom]`

scénář a spustit tak jeho proces schvalování. Veškerá data ze šablony budou duplikována, vyjma souborů.

Uživatel může pracovat pouze se svými šablonami. Duplikovat může navíc i cizí veřejné šablony. Kopie se stane jeho vlastní. Funkce budou dostupné pouze po přihlášení.

5.1.12 Podmínky a pravidla při schvalování

Při vytváření scénáře je možné definovat jednotlivé úrovně (kroky/fáze), přes které bude schvalování postupně procházet, v každém z těchto kroků je možné definovat schvalovací podmínku a pravidlo návratu.

- Pravidlo návratu – definuje, do kterého kroku se má zpracování vrátit, v případě, že v tomto kroku dojde k zamítnutí.
- Schvalovací podmínka – definuje, za jakých okolností je možné krok považovat za schválený. Má následující možnosti:
  - Všichni – schválení musí provést všichni uživatelé definovaní v tomto kroku.
  - Alespoň jeden – schválení musí provést jeden libovolný uživatel v kroku.
  - Alespoň jeden z každé skupiny – u všech přiřazených uživatelských skupin k danému kroku musí provést schválení alespoň jeden uživatel z každé z nich.

5.1.13 Schvalovací proces

V průběhu schvalovacího procesu mohou být použity tyto možnosti:

- Schválení v konkrétním kroku – ve chvíli, kdy schvalovatel klikne na tlačítko *Schválit*, dojde k zaznamenání jeho akce. V případě, kdy je splněna definovaná podmínka (5.1.12), je možné pokračovat do další fáze, jinak se čeká na další uživatelské akce. Pokud došlo ke schválení aktuální fáze a neexistuje další, celý schvalovací proces je ukončen a označen za schválený.
- Zamítnutí v konkrétním kroku – kliknutím na tlačítko *Zamítnout* dojde k zaznamenání akce schvalovatele. Pokud již neexistuje cesta, jak schválení dosáhnout, systém se v případě definovaného návratového pravidla vrátí do předchozího stavu, kterým schvalovací proces pokračuje. Pokud není toto pravidlo definováno, schvalování je ukončeno a je označeno za zamítnuté.
- Vkládání komentářů – ke každé akci, kterou uživatel může se schvalovacím procesem provést, může přidat komentář.

- Storno schvalování – odesílatel (majitel) schvalovacího procesu ho může v jakémkoliv stavu stornovat.

Schválený a zamítnutý proces není možné odstranit. Funkce budou dostupné uživateli, který je oprávněn vykonávat v souvislosti s daným schvalovacím procesem patřičnou akci, a pouze po přihlášení.

5.1.14 Přístup k veřejnému scénáři

Každý scénář bude moci být opatřen před svým odesláním ke schvalování příznakem, který bude určovat, zda je k němu umožněn veřejný přístup[19]. Přes sekci hlavní stránky *Zobraz své flow* pak bude možné za pomoci speciálního identifikačního kódu schvalovacího procesu přistoupit k online přehledu. Identifikační kód bude ve formátu univerzálního unikátního identifikátoru (UUID) v4, který je `xxxxxxxx-xxxx-4xxx-Nxxx-xxxxxxxxxxxx`:

- `x` – alfanumerické znaky,
- `4` – označení verze,
- `N` – nejvýznamnější bit v 1–3 bitovém poli obsahuje variantu.

Náhled je dostupný pouze pro veřejné schvalovací procesy a i mimo autentizovanou část systému.

5.1.15 Vytěžování e-mailové schránky

Systém bude napojen na e-mailovou schránku zvanou podatelna. E-maily v této schránce budou automaticky vyzvedávány, v intervalu ne však delším než 15 minut. Mohou nastat tyto možnosti:

- Načtení nové zprávy bez identifikátoru – systém vyhodnotí zprávu jako nové podání. Následně zprávě přiřadí nový unikátní identifikátor, načte e-mail odesílatele a všechny přiložené přílohy a společně s obsahem e-mailu uloží do databáze. Zpráva tak bude připravena na ruční kategorizaci a odeslání ke zpracování. Odesílatel e-mailu bude o všem získávat notifikace (5.1.19).
- Načtení nové zprávy s identifikátorem (nepovinné) – pokud bude ve zprávě nalezen identifikátor podání, bude obsah zprávy přiložen k aktivnímu schvalovacímu procesu jako *Komentář zadavatele*.

[19] Mimo autentizovanou části systému.

- Přiřazení scénáře k podání – oprávněný uživatel (prostřednictvím role) k čekajícímu podání přiřadí nejlépe vyhovující scénář (popřípadě vytvoří nový) a zprávu odešle ke zpracování. Odeslaný scénář se automaticky stává veřejným (5.1.14). Funkce bude dostupná pouze po přihlášení.

5.1.16 Správa uživatelských účtů

U každého uživatele se evidují následující údaje: uživatelské jméno, křestní jméno, příjmení, ověřený, popřípadě neověřený e-mail, časy registrace, posledního přihlášení, informace o blokaci účtu, uživatelské skupiny a role oprávnění. Volitelně dále uživatelský avatar. Možnosti správy uživatelských účtů jsou následující:

- Vytvoření uživatele – uživatelský účet musí být ručně založen. Při této akci musí být vyplněny tyto údaje: uživatelské jméno, křestní jméno, příjmení, e-mail, dále volitelně uživatelské skupiny a role. K vytvoření profilu dojde kliknutím na tlačítko *Vytvořit*.
- Aktualizace uživatele – u každého uživatele bude moci pověřený uživatel aktualizovat tyto údaje: křestní jméno, příjmení, e-mail, uživatelské skupiny a role.
- Odstranění uživatele – není reálně umožněno; uživatel bude pouze označen jako odstraněný a nebude mu umožněn přístup do systému. Také se nebude dále zobrazovat v nabídkách pro přiřazení do skupin.

Funkce budou přístupné pouze administrátorovi systému a uživatelům s přiděleným patřičným oprávněním (rolí). Vyžaduje přihlášení.

5.1.17 Správa skupin s oprávněními

Bude dostupná funkce správy skupin uživatelů pro přidělování systémových oprávnění. Možnosti jsou následující:

- Vytvoření skupiny – uživatel bude moci vytvořit skupinu s oprávněními, kde bude nutné vyplnit název skupiny a volitelně i její popis. K vytvoření dojde po kliknutí na tlačítko *Vytvořit* a přesměrování na detail, kde budou veškeré údaje. Vyskytovat se tam budou i formuláře určené pro správu uživatelů skupiny (5.1.10) a správu přiřazených rolí s oprávněními (skrze stejné rozhraní jako správa uživatelů).

- Aktualizace skupiny – aktualizace údajů bude uživateli umožněna prostřednictvím detailu skupiny. Bude možné změnit jak název skupiny, tak i její popis a současně i přiřazené uživatele (5.1.10) a role s oprávněními.
- Odstranění skupiny – uživatel bude moci v přehledu odstranit skupinu ze systému.

Funkce budou přístupné pouze administrátorovi systému a uživatelům s přiděleným patřičným oprávněním (rolí). Vyžaduje přihlášení.

5.1.18 Správa nahraných dokumentů (nepovinné)

Veškeré nahrané dokumenty budou v systému přehledně organizovány a spravovány na jednom místě. V systému vystupují dva druhy dokumentů:

- Vlastní dokumenty – jsou právě ty dokumenty, které uživatel nahrál do systému k některému ze svých scénářů. Dokumenty z neodeslaných scénářů bude možné aktualizovat a také odstranit.
- Cizí dokumenty – jsou veškeré dokumenty, ke kterým bude mít uživatel přístup skrze schvalovací proces, ale sám je do systému nenahrál. Bude mu pouze umožněno do nich nahlížet.

Funkce bude dostupná pouze po přihlášení.

5.1.19 E-mailová notifikace

Systém bude průběžně zasílat notifikace buď na e-mailové adresy uživatelů nebo v případě zpracovávání požadavku přijatého e-mailem (5.1.15) na adresy odesílatelů e-mailové zprávy.

Vytěžování e-mailu:

- přijetí podání,
- přidělení podání ke scénáři,
- přidání dodatku podání,
- zamítnutí dodatku podání,
- neplatný formát dodatku podání.

Uživatel systému:

- vytvoření účtu,
- změna e-mailu,
- změna hesla,
- odhlášení na všech zařízeních.

5.1.20 Logování akcí (nepovinné)

Systém bude zprostředkovávat logování všech akcí ve výčtu:

- změna hesla,
- úspěšné přihlášení,
- neúspěšné přihlášení,
- přístup k zabezpečenému endpointu bez oprávnění,
- změna e-mailu,
- zablokování uživatele,
- registrace nového uživatele,
- odhlášení na všech zařízeních,
- vytvoření, aktualizace a odstranění položek všech seznamů,
- odeslání scénáře ke zpracování,
- akce uživatele u scénáře,
- zpracování přijatého e-mailu,
- úspěšné ověření e-mailové adresy

Funkce budou přístupné pouze administrátorovi systému a uživatelům s přiděleným patřičným oprávněním (rolí). Vyžaduje přihlášení.

5.2 Nefunkční požadavky

Definice požadovaných funkcí není to jediné, co je pro popis kvalitního systému vyžadováno. Další kategorií jsou tzv. nefunkční požadavky. Jedná se o omezení kladená na kvalitu, definují tedy, jaký by systém měl být (vlastnosti), ne co by měl dělat.

5.2.1 Jednoduchost a přehlednost

Vytvářený systém by měl být jednoduchý na používání a přehledný, aby se v něm uživatel co nejrychleji zorientoval. Též je žádaný moderní návrh designu grafického rozhraní.

5.2.2 Přenositelnost a kompatibilita

Je vyžadována podpora vykreslovacích jader Gecko, Webkit a Blink určených pro prohlížeče Google Chrome, Mozilla Firefox, Microsoft Edge a Safari. Žádaná je též co nejširší podpora aktuálních prohlížečů mobilních platforem.

Server poběží na Linuxové distribuci a bude využívat aktuální Apache, v krajním případě i Nginx, webový server. Pro ukládání dat bude využita relační databáze z typu MySQL.

Klientské soubory bude možné ukládat buď na lokální úložiště, případně do cloudu provozovatele – dle konfigurace při nasazení systému.

5.2.3 Zabezpečení

Hesla nebudou v databázi ukládána v čitelné podobě. Pro jejich ukládání bude použita některá z dostupných funkcí pro odvození klíče, která splňuje požadavky na bezpečné ukládání hesel (ideálně argon2id).

Webové služby budou primárně využívat šifrované komunikace. Klientská i serverová část systému bude dostupná pouze na šifrované (https) adrese.

Uživatelům systému není umožněno přistupovat k cizím datům bez patřičných oprávnění.

Pro udržování klientské session budou použity rotační tokeny. Autentizace všech požadavků do zabezpečené části systému bude probíhat pomocí JWT, jehož životnost bude omezena maximálně na 15 minut. Pro automatické prodloužení uživatelské session bude používán jednorázový refresh token s životností až 3 hodiny.

5.2.4 Lokalizace

Prostředí bude dostupné pouze v českém jazyce.

5.2.5 Systémové role

Systém bude implementovat role pro udělování oprávnění k určitým částem systému a jeho funkcím. Role bude možné přiřadit jak skupině s oprávněními (5.1.17), tak i přímo uživateli (5.1.16). Role není možné vytvářet, aktualizovat ani mazat.

5.3 Návrh databáze

Základem každého informačního systému je i dobře provedená analýza a samotný návrh struktury databáze. V případě, že dojde k nesprávnému promyšlení této struktury, bude velmi náročné takový systém implementovat. Tato skutečnost může vést až k nutnosti předělat celou databázi v průběhu vývoje. To by znamenalo, že bude také nutné přepsat velkou část již napsaného zdrojového kódu aplikace. Také může dojít ke zpoždění termínu dokončení a jeho celkovému prodražení.

Navrhovaný informační systém má strukturu databáze poměrně rozsáhlou a z toho důvodu nebude v této práci popsána úplně do detailu každá její entita. V příloze A na straně 80 je tak přiložen kompletní entitně-relační diagram (ERD). Jedná se o vztahový model, jenž je použit k abstraktnímu a konceptuálnímu znázornění dat. Obsahuje entity (uživatel, skupina, ...) a dále relace (vztahy) mezi nimi.

Pro vytvoření modelu byl použit nástroj MySQL Workbench, který je dostupný zdarma v komunitní verzi pod svobodnou licencí GNU GPL.

5.3.1 Stručný popis hlavních entit

Každé z hlavních entit bude věnován krátký prostor pro stručný popis její funkce v kontextu informačního systému.

`user`

Jedná se o nepostradatelnou entitu, která popisuje uživatele systému. Tato entita je propojena s velkým množstvím ostatních entit, kde buď označuje vlastnictví nebo uživatelské akce.

`security_key`

Obsahuje jednorázové klíče sloužící k ověření e-mailové adresy.

`refresh_token`

Entita uchovává jednorázové uživatelské tokeny, které slouží pro automatické prodlužování relací.

role

Tato entita je velmi jednoduchá, obsahuje výčet v systému dostupných rolí oprávnění.

security_group

Rozděluje uživatele do skupin, prostřednictvím kterých jim přiřazuje patřičné role s oprávněními.

flow_group

Funguje velmi podobně jako bezpečnostní skupina, člení ovšem uživatele do skupin pro řízení běhu procesů. Využívá se jak v šablonách, tak při běhu procesu.

flow_confirm_rule

Je další z řady entit, která drží pouze výčet hodnot. V případě této jde o seznam pravidel určujících, za jakých podmínek bude fáze oběhu schválena.

flow_template

Prostřednictvím šablony je možné navrhnout scénář procesu. Tato entita tak drží nezbytně nutné informace. Těmi například je jméno šablony, kdo ji vytvořil, zda proces z ní vytvořený bude veřejný, a další.

flow_template_step

K tomu, aby u šablony bylo možné zachytit celou komplexnost procesu, je nutné členit ho na jednotlivé fáze. K tomu je právě určena i tato entita. Drží potřebné informace o každé navržené schvalovací fázi. Těmi nejdůležitějšími jsou: pravidlo pro schválení; pořadí v procesu; volitelný odkaz na fázi, která bude prováděna v případě zamítnutí.

Prostřednictvím vazební tabulky také drží seznam přiřazených uživatelských skupin.

flow_running

Jedná se o hlavní entitu používanou při průběhu schvalování. Díky tomu je i poměrně rozsáhlá a má mnoho definovaných vazeb. Ty význačné tvoří například s entitami `flow_running_template_step` a `flow_running_template_group`, které drží kompletní scénář oběhu. Jedná se tedy o zduplikovanou entitu `flow_template` a jí podřízené.

Entita drží informace o průběhu samotného procesu společně s pomocnými a řídícími proměnnými. Těmi pomocnými jsou například pořadí posledního kroku, ID aktuálně aktivní fáze. Naopak těmi řídícími jsou: aktuální stav (probíhající, schváleno, ...) nebo příznak o veřejném přístupu.

`flow_running_decision_step`, `flow_running_decision_group`

Tyto dvě entity jsou používány pro zachycení struktury probíhajícího scénáře. V případě první z nich (step) se jedná o rozšířenou entitu šablony, která navíc drží pomocné a řídící atributy, potřebné pro uchování stavů konkrétní fáze procesu. Druhá (group) je shodná se šablonou, avšak narozdíl od ní vytváří skupiny uživatelských akcí.

`flow_running_decision_action`

Entita uživatelských akcí se stará o evidování již provedených, popřípadě právě čekajících akcí. Drží tedy informace o rozhodnutích, která uživatel provedl, nebo je právě provést může. Podstatným je příznak o druhu provedené akce (schváleno, zamítnuto, ...), čas provedení a přiložená poznámka, proč se tak uživatel rozhodl.

`flow_comment`

Uživatelé spolu mohou pod každým aktivním procesem diskutovat formou komentářů. O ukládání těchto dat se stará právě tato entita.

`uploaded_file`

Každý soubor nahraný do systému a uložený na disk má v databázi o sobě uložené doprovodné informace. Těmi je například název souboru, uživatel, který ho nahrál, k čemu se soubor vztahuje (proces, šablona., e-mail), cesta k uloženému souboru a typ jeho obsahu (content-type).

`file_metainfo`

Entita uchovává dodatečné informace o nahraných souborech. Těmito informacemi jsou tzv. metadata. Může jít například o autora, velikost, datum vytvoření a další.

extracted_email

Data z načteného a zpracovaného e-mailu z podatelny je nutné uložit do databáze. Tato entita tak ukládá základní atributy jako identifikátor podání, povolení odesílatelé a čas doručení.

extracted_message

U každého zpracovaného e-mailu jsou zpráva a její předmět uloženy separátně do této entity. Je to proto, aby bylo možné dodatkovat již vytvořené podání.

app_settings

Tato entita ukládá globální nastavení celého informačního systému.

log

Data o systémem zaznamenaných událostech jsou uložena do této entity.

Dále jsou v ERD dostupné vazební entity pro relace M:N[20]:

- `user-group`
- `user-role`
- `group-role`
- `flow_group-user`
- `flow_template_step-flow_group`
- `flow_running-uploaded_file`
- `flow_running_template-group_user`
- `uploaded_file-flow_template`
- `uploaded_file-extracted_email`.

Všechny tyto entity obsahují pouze dva atributy klíčů.

Názvy jsou v databázi psány pouze s podtržítky, v tomto výčtu však byla některá podtržítka nahrazena pomlčkou, která vyznačuje, mezi kterými entitami vazba vzniká.

[20]Umožňuje každému záznamu z první tabulky přiřadit libovolný počet záznamů z tabulky druhé.

6 POPIS A IMPLEMENTACE IS

V této kapitole budou prezentovány zajímavé části vyvinutého informačního systému DokFlow. Čtenář také bude uveden do problematiky použité automatizace vývoje.

6.1 Automatizace vývoje

Každý softwarový vývojář, analytik a popřípadě i tester velmi rád ocení, když je co možná největší část nutných kroků spojených s testováním a nasazením vyvíjené aplikace automatizována. Často je tak možné se setkat se zkratkou CI/CD. Jedná se o zkrácení slov *continuous integration* (průběžná integrace) / *continuous deployment* (průběžné nasazení).

6.1.1 Průběžná integrace

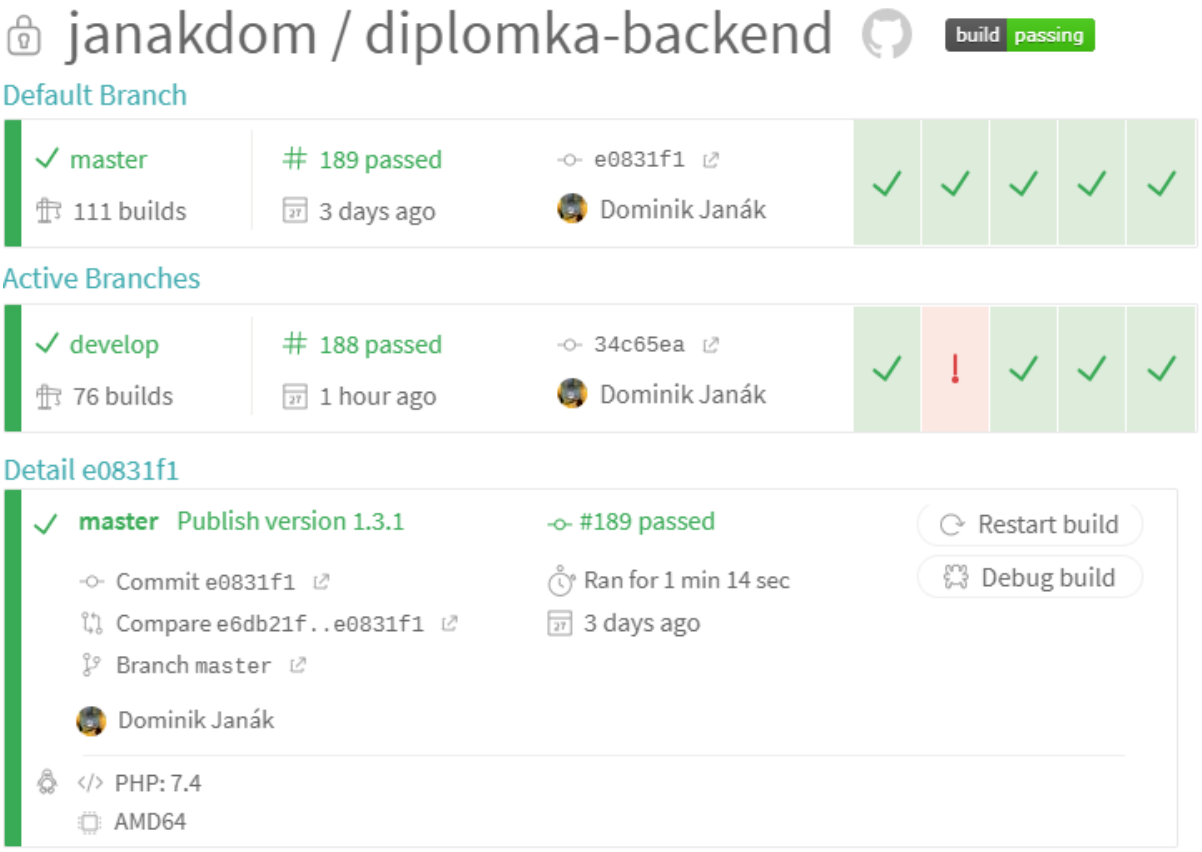
Průběžná integrace nebo též *Continuous Integration* (CI) je soubor technik, které pomáhají zrychlit a nepřímo zlevnit vývoj softwaru prostřednictvím automatizovaného hledání chyb. Celý postup CI předpokládá uvolňování drobných změn v krátkých intervalech. Jedná se o jednu z technik *extrémního programování*. [33]

Díky tomu, že jsou změny publikovány rychle za sebou, může na projektu pohodlně pracovat i větší skupina programátorů. Ve chvíli, kdy některý z nich do sdíleného repozitáře odešle své změny, spustí se proces *průběžná integrace*. Ten je obvykle rozdělen na několik kroků. Jejich počet závisí na konkrétní konfiguraci každého projektu. Jedním z nich by mělo být provedení všech testů (*jednotkových*, *integračních* a *systémových*). Pokud všechny projdou, může se pokračovat dále, k pokusu o sestavení *buildu*. [33]

V případě, že se build podaří sestavit, může dojít k navázaným automatickým operacím, například k *průběžnému nasazení* (CD) aplikace. V opačném případě, kdy dojde k nalezení jakékoliv chyby, a to platí i pro testy, jsou programátoři o této skutečnosti okamžitě informováni a mohou tak neprodleně začít problém řešit.

Tento přístup tak podstatně zkracuje čas potřebný k nalezení a odstranění části chyb. Stejně tak se *průběžná integrace* stará o minimalizaci výskytu těchto chyb v projektu. V žádném případě však nezaručuje, a to i v případě, kdy dojde k úspěšnému provedení všech kroků, že je projekt bez chyb.

Na obrázku 9 je ukázáno rozhraní nástroje Travis-CI. První dva přehledy ukazují detail posledních pěti provedených testů na vývojových větvích *master* a *develop*. Pod nimi je potom zobrazen detail posledního testu na větvi *master*.



Obrázek 9 – Průběžná integrace, rozhraní nástroje Travis-CI, zdroj: vlastní

6.1.2 Průběžné doručení

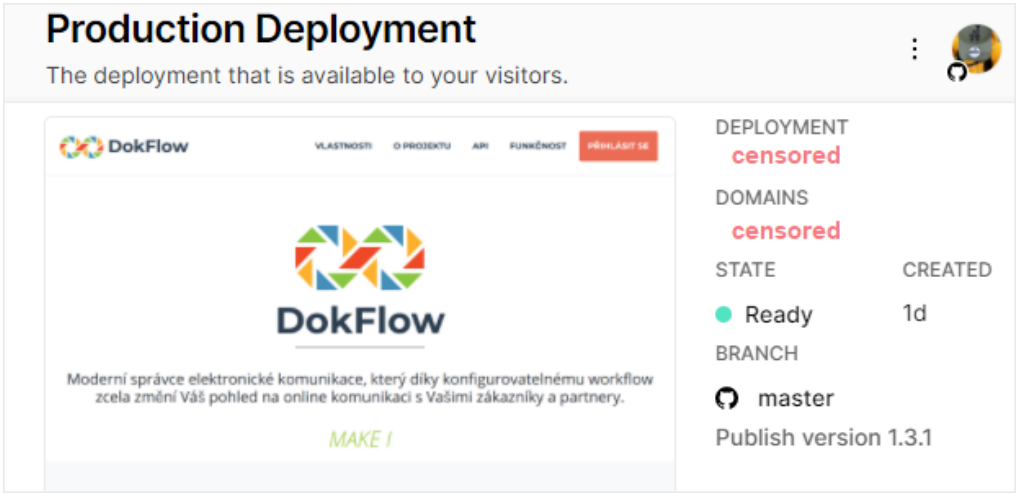
Průběrné doručení, anglicky *Continuous delivery* (CD), je rozšíření *průběžná integrace*. Jedná se o rychlý a bezpečný způsob, jak v průběhu vývoje plně automaticky ověřovat, že se projekt stále nachází ve funkčním a nasaditelném stavu.

Aby bylo možné zahájit průběžné doručení, musí být předtím úspěšně (bez chyb) dokončena *průběžná integrace*. Celý postup doručení je rozdělen do několika kroků, kdy jsou prováděny určité testy. Pokud všechny tyto testy projdou bez chyb, může se pokračovat do další fáze, kterou je obvykle *průběžné nasazení* [33]. Ve chvíli, kdy se vyskytne chyba, jsou o tom programátoři informováni stejně jako u CI.

6.1.3 Průběžné nasazení

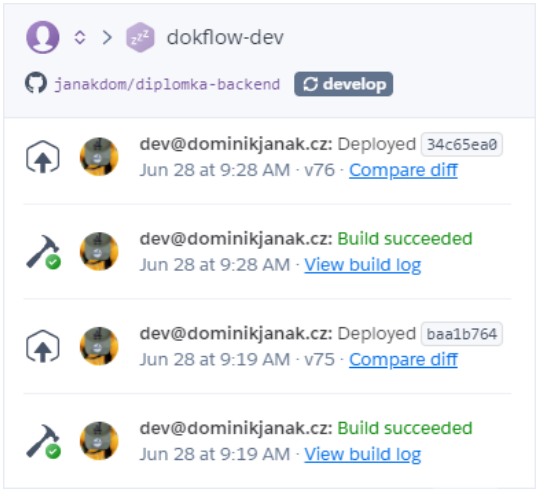
Poslední fází je *průběžné nasazení*, které rozšiřuje *průběžné doručení* [33]. Pokud všechny předešlé kroky (CI + CD) projdou bez chyb, dojde k automatickému nasazení projektu do produkčního prostředí a tím budou veškeré změny plynule dodány všem uživatelům.

Hlavní nevýhodou průběžného nasazení je automatické zpřístupnění všech verzí, které projdou celým předchozím procesem bez možnosti výběru.



Obrázek 10 – Průběžné nasazení, rozhraní platformy Vercel, zdroj: vlastní

Průběžné nasazení je velmi užitečnou funkcí v průběhu vývoje. Na obrázku 10 je zobrazen výňatek z platformy Vercel, která slouží k průb. nasazení frontendových aplikací. A následně na obrázku 11 je vyobrazena část platformy Heroku, pro backendové aplikace.



Obrázek 11 – Průběžné nasazení, rozhraní platformy Heroku, zdroj: vlastní

6.1.4 Použité technologie

Pro úspěšné zavedení celého procesu průběžných metod dnes na trhu existuje celá řada placených i volně dostupných služeb. V tomto projektu bylo využito následujících možností:

- GitHub, jakožto veřejné úložiště sdíleného kódu,
- Travis-CI, poskytující služby *průběžná integrace*,
- Heroku, pro nasazení backendové části
- a Vercel frontendové části aplikace.

6.2 Hlavní stránka

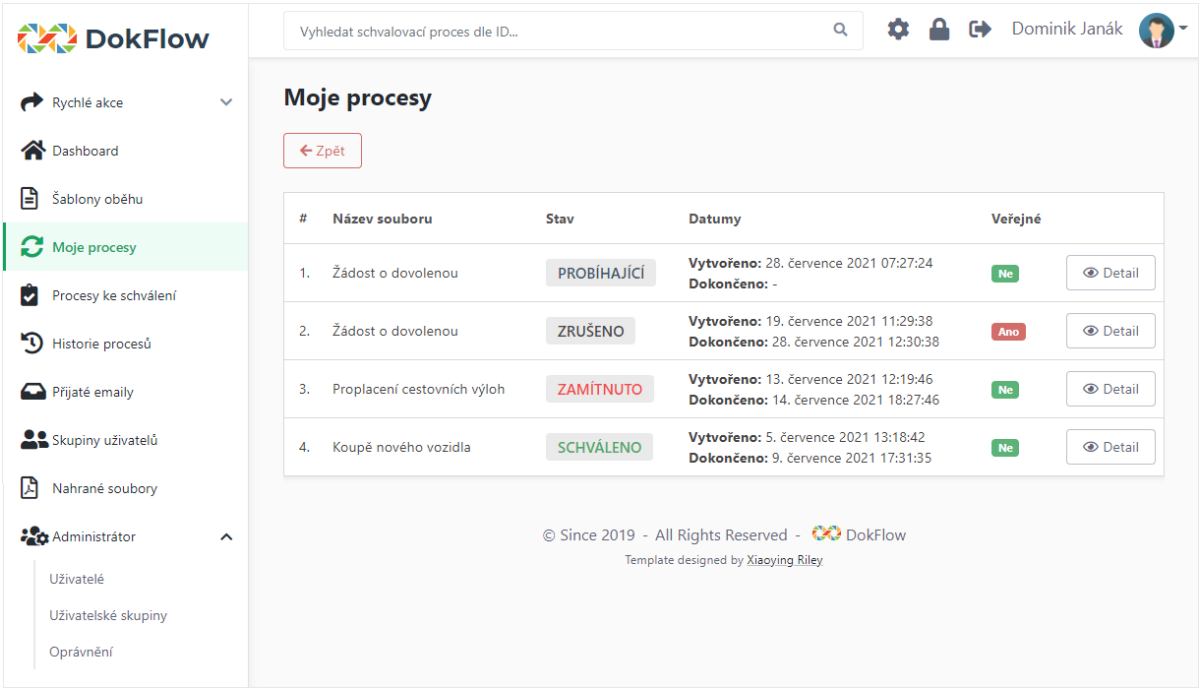
Hlavní strana je dostupná na kořenové adrese domény. Poskytuje přehledné rozhraní popisující celý informační systém a jeho funkce. Design je k náhledu na obrázku 12. Hlavní strana je tvořena pouze jednou stránkou. Ta je navržena v moderním a minimalistickém designu. V jejích spodních částech jsou stručně popsány základní funkce systému, informace o autorovi a také formulář pro přístup k veřejným schvalovacím procesům (6.6.2).

Obrázek 12 – Hlavní stránka s informacemi o informačním systému, zdroj: vlastní

Stránka je plně mobilně responzivní a je dostupná pouze v české lokalizaci, stejně jako celý informační systém. Ve spodní části je umístěn odkaz na licencované prvky, použité při vývoji, společně s odkazem na výpis změn v jednotlivých verzích systému.

6.3 Aplikace

Design samotné aplikace, na obrázku 13, je minimalistický a snaží se poskytnout co nejjednodušší a nejintuitivnější ovládání, jaké je možné. Společně s tím je layout částečně mobilně responsivní. Tato vlastnost však není nutně dostupná u komplexnějších a popřípadě datově obsáhlejších částí systému. Příkladem může být na obrázku 13 vyobrazený seznam dostupných schvalovacích procesů, nebo rozhraní pro jejich modelování (6.5.2).



Obrázek 13 – Ukázka designu aplikace, zdroj: vlastní

Pro návrh designu aplikace byla použita základní HTML šablona od produktové designérky *Xiaoying Riley*, která byla převedena do Reactu a plně přizpůsobena potřebám celého systému. Jedná se o volně dostupnou Bootstrap 5 šablonu, kterou lze při zachování odkazu na autora v zápatí každé stránky používat zdarma i komerčně.

6.4 Hlavní uživatelské rozhraní

V této krátké kapitole budou popsány základní grafické komponenty systému, které zprostředkovávají jeho hlavní funkce. I když nejsou konkrétně popsány v požadavcích, bez těchto grafických rozhraní by se informační systém nedokázal obejít.

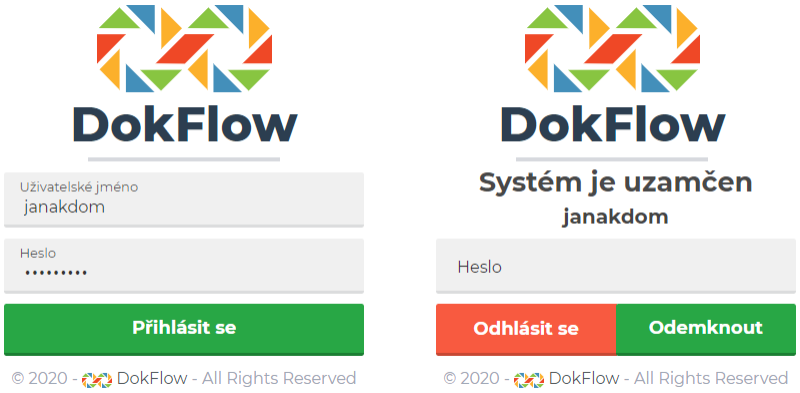
6.4.1 Notifikační lišta o používaných cookies

Ačkoliv se to může na první pohled jevit jako zbytečné snižování pohodlí uživatele, při prvním příchodu na stránku je mu zobrazena speciální informační notifikace, která ho důrazně upozorňuje, že systém ke svému fungování využívá soubory cookies. Uživatel s tím souhlasí v případě, že pokračuje v používání informačního systému.

Toto chování je nezbytné pro soulad s českým zákonem (č. 110/2019 Sb.) o ochraně osobních údajů, který je podmíněn nařízením o ochraně osobních údajů vydaným evropskou radou, známým pod pojmem GDPR (EU 2016/679).

6.4.2 Přihlášení

Jednou z nepostradatelných součástí aplikace je přihlašovací formulář, který zprostředkovává službu autentizace. Uživatel pro přihlášení musí zadat správnou kombinaci uživatelského jména a hesla. Formulář také reaguje na nově zřízený



Obrázek 14 – Rozhraní pro přihlášení (vlevo) a odemčení systému (vpravo), zdroj: vlastní

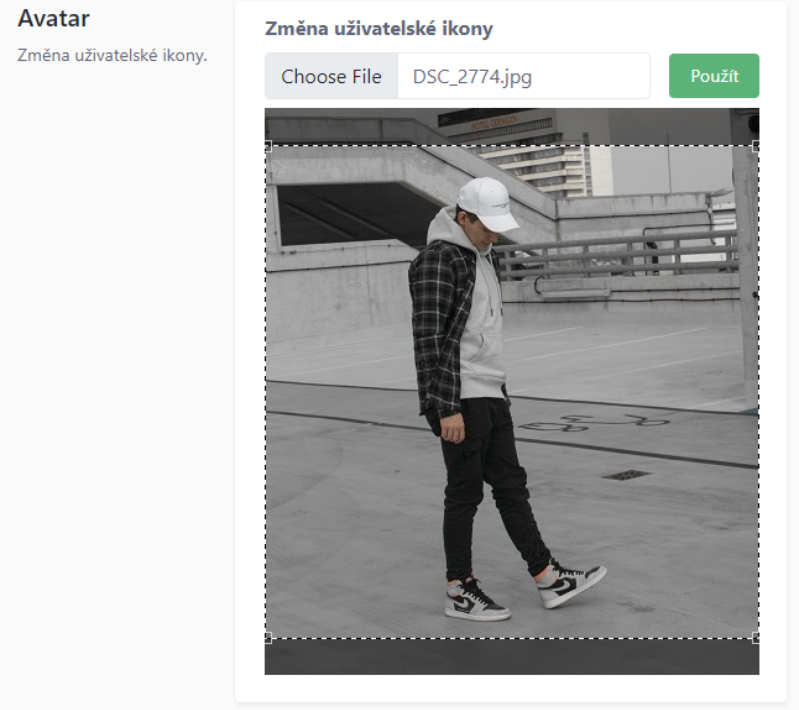
uživatelský profil, který ještě nemá validovaný první e-mail. Při takové skutečnosti uživateli nabídne možnost odeslání nového validačního e-mailu a ponechá ho na přihlašovací stránce. Komponenty jsou plně animovány, tak aby uživatel dostal

maximální zpětnou vazbu o probíhajících událostech. Formulář je dostupný k náhledu na obrázku 14, vlevo.

Formulář taktéž umožňuje uživateli opětovné přihlášení do systému v případě, kdy ho pouze uzamkl (obrázek 14, vpravo) a neprovedl kompletní odhlášení. V takovém případě je použito zapamatované uživatelské jméno a čeká se pouze na zadání hesla. Uživatel také může dokončit kompletní odhlášení.

6.4.3 Změna uživatelského avataru

Pro zvýšení celkového uživatelského komfortu systému do něj byla přidána i možnost změny uživatelského avataru. Jedná se o intuitivní rozhraní, které poskytuje snadný způsob, jak uživatelem vybraný obrázek plně přizpůsobit potřebám systému a získat tedy pouze čtvercový rozměr. Rozhraní je k náhledu na obrázku 15.



Obrázek 15 – Formulář pro oříznutí a změnu uživatelského avataru, zdroj: vlastní

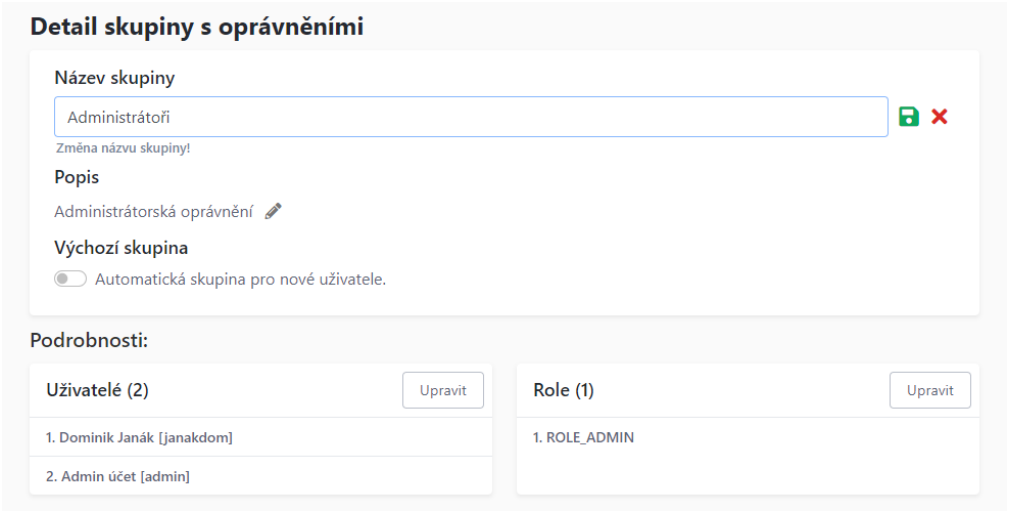
Uživatelský obrázek je kompletně zpracován již na straně klienta a serveru se předává oříznutý, tak aby stačilo ověřit pouze jeho požadované vlastnosti a následně ho uložit.

6.4.4 Správa skupin a rolí uživatele

Administrátor systému má dostupnou možnost správy uživatelů systému a také skupin určených k hromadnému aplikování rolí oprávnění. Každému uživateli je také možné přiřadit roli i samostatně. Na obrázku 16 je zobrazen formulář správy výše zmíněné uživatelské skupiny.

Zajímavou vlastností každé skupiny je její možnost označení jakožto výchozí. V takovém případě bude tato skupina uživateli automaticky přiřazena ve chvíli registrace.

Pokud by došlo k napojení systému na LDAP server, mohou být skupiny automaticky přiřazeny již při prvním přihlášení, neboť registrace v takovém případě není potřeba. Uživatel se totiž při každém přihlášení autentizuje proti LDAP serveru. Po úspěšném přihlášení předá LDAP server systému veškerá potřebná data pro zřízení lokálního účtu.



Obrázek 16 – Formulář pro editaci skupiny uživatelů a jejich rolí, zdroj: vlastní

Ve spodní části jsou dostupné dvě editovatelné tabulky, které jsou tvořeny univerzální komponentou (5.1.10). Jsou určeny ke správě uživatelů a rolí přiřazených ke skupině. V systému figuruje ještě jeden druh skupin, jedná se o skupiny uživatelů pro schvalování (5.1.9) a mají obdobné rozhraní pouze bez přítomnosti správy rolí a označení jakožto výchozí.

6.5 Správa šablon procesů

Správa šablon je důležitou částí systému, jenž umožňuje provádět správu, tedy definice, úpravy a odstraňování scénářů podnikových procesů.

6.5.1 Seznam dostupných šablon

Seznam šablon zobrazuje jednotlivé, uživateli dostupné, šablony a detailní informace o nich – počet fází říká, v kolika schvalovacích krocích je proces obsluhován; počet souborů charakterizuje počet přiložených souborů a příznak *veřejné* charakterizuje, zda je šablona sdílena s ostatními uživateli. Cizí šablony, které jsou uživateli dostupné, jsou déle podbarveny šedivým pozadím. Na obrázku 17 je prezentováno celé rozhraní.

Každá šablona v seznamu obsahuje několik tlačítek. Při kliknutí na *Detail* je uživatel přesměrován na detail konkrétní šablony. V případě dalších možností stačí kliknout na tlačítko šipky dolů, vpravo od tlačítka *Detail*, kdy dojde ke zpřístupnění tlačítka, které umožní ze šablony vytvořit duplikát a u vlastních šablon navíc druhého tlačítka pro odstranění šablony.

Obrázek 17 – Seznam dostupných šablon procesů v univerzální komponentě, zdroj: vlastní

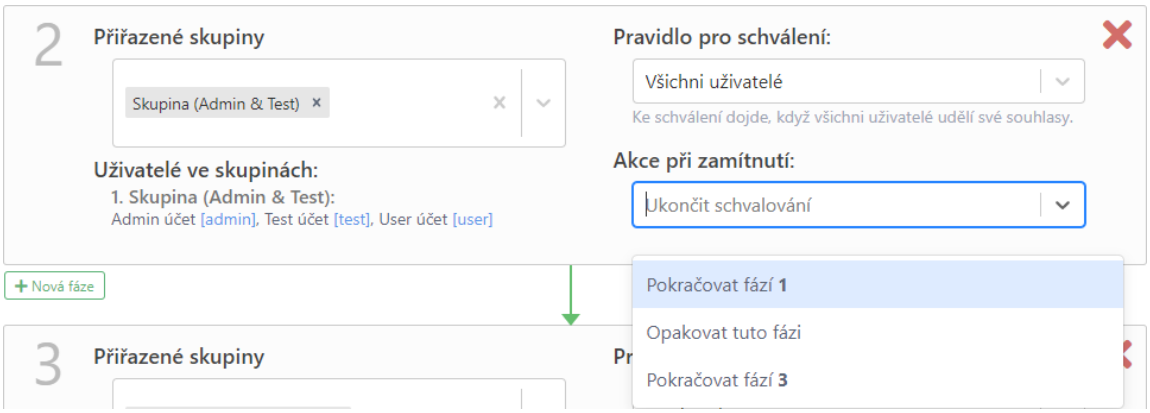
Seznam šablon je tvořen univerzální komponentou, která je napříč celým systémem použita pro vykreslování seznamů a tabulek. Má jednotné rozhraní pro definici zobrazovaných sloupců a jejich hodnot (přístupné přes definované accessory), dále dynamické možnosti podbarvení jednotlivých řádků a přidávání vlastních tlačítek.

6.5.2 Designer schvalovacího procesu

Designer schvalovacích procesů ve velmi jednoduchém a přehledném rozhraní zpřístupňuje možnosti tvorby jednoduchých i propracovaných scénářů. Jak je z obrázku 18 patrné, možné je vytvářet jednotlivé schvalovací fáze, u nichž je dostupná volba uživatelských skupin. Ty definují konkrétní uživatele, na které budou delegovány patřičné akce.

Dále je pro každou fázi možné zvolit *pravidlo pro schválení*, které říká, za jakých okolností bude daná fáze považována za schválenou. Systém také i umí vyhodnotit, zda stále existuje možnost, jak proces schválit, a může ho ponechat v aktivním stavu. Pokud již tato cesta neexistuje, je fáze považována za zamítnutou. Pro tuto skutečnost je k dispozici další parametr definující *akci při zamítnutí*. Tato akce rozhoduje, co se bude dít v případě, že již neexistuje cesta, jak danou fázi schválit.

Fáze jsou přehledně řetězeny pod sebou, přesně tak, jak bude probíhat schvalovací proces v případě pouze kladného postupu vyřízení. Vyobrazení postupu, za předpokladu zamítnutí některých fází (i opakovaně), se v běžícím procesu postará o vložení nového kroku dle definovaného pravidla na místo následující fáze. Dojde tak k rozšíření postupu, který byl definován původní šablonou.



Obrázek 18 – Fragment designeru pro modelování procesů, zdroj: vlastní

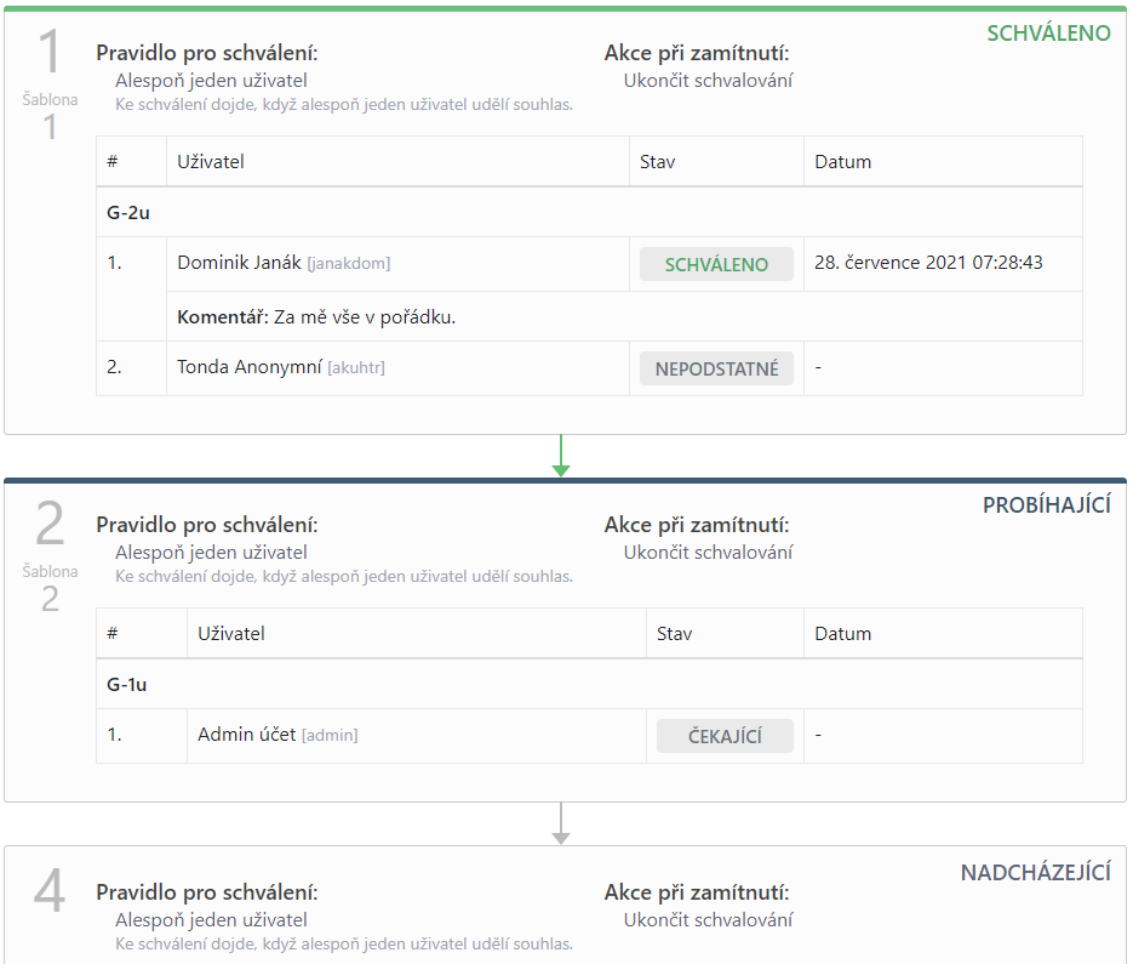
Součástí designeru je i možnost přidávat a odstraňovat jednotlivé fáze, a to i již mezi ty existující. Systém také ošetřuje stav, kdy znemožní odstranění fáze, která je referencována jinou, prostřednictvím definované *akce při zamítnutí*.

6.6 Správa procesů

Aktivní procesy vznikají odesláním z dříve definovaných šablon. Každému uživateli jsou v menu dostupné sekce: svých procesů, procesů ke schválení a kompletní historie předešlých rozhodování.

6.6.1 Náhled průběhu schvalování

Náhled průběhu procesu (obrázek 19) je tvořen komplexním rozhraním, ve kterém vidí každý oprávněný uživatel jeho celý průběh. Nejenže je obsažena kompletní historie všech provedených akcí, systém také vykresluje, co se pravděpodobně bude dít v budoucnu. Z důvodu dostupnosti podmínek při zamítnutí není tato předpověď vždy pravdivá. Pouze ukazuje, jak bude schvalování probíhat v případě, že nedojde k žádnému



Obrázek 19 – Formulář pro přístup k veřejným schvalovacím procesům, zdroj: vlastní

zamítnutí. Samotný odesílatel procesu také může ještě před jeho schválením nebo

zamítnutím iniciovat jeho stornování. Ani v jednom případě se již vytvořený proces nedá z důvodu trvanlivosti záznamů odstranit.

Aby bylo možné dosáhnou maximální uživatelské přívětivosti, je celý formulář automaticky aktualizován a tato skutečnost je uživateli signalizována nenápadným odpočtem. Také je každému uživateli, na kterého čeká akce, umožněno zanechat pro své rozhodnutí komentář a současně i přiložit další soubory, které budou dostupné v náhledu kroku, při kterém byly vloženy. Soubory též mohou být přiloženy k celému procesu a jsou pak dostupné níže pod náhledem průběhu.

Pod soubory je také dostupná sekce s komentáři, kde uživatelé mohou diskutovat o průběhu celého procesu. Komentáře jsou dostupné pouze v základní jednoúrovňové verzi a není u nich umožněno zadávat reakce formou zanořování. Jedná se spíše o doplněk k této práci a není tedy výrazněji propracován.

6.6.2 Veřejné rozhraní pro přístup

Na hlavní informační stránce (mimo zabezpečenou část systému) je dostupný formulář, prostřednictvím kterého si může kdokoliv nechat zobrazit aktuální stav libovolného veřejného procesu. Tento náhled je možné zobrazit v případě, že dotyčný uživatel zná unikátní identifikátor ve formátu UUID verze 4.

6.7 Zpracování e-mailů

Automatické zpracování přijatých e-mailů je další z výběru žádaných funkcí. Informační systém je napojen na e-mailovou schránku, která funguje jako podatelna. Veškeré do ní doručené e-maily jsou tak v definovaném časovém intervalu automaticky zpracovány a data vytěžena a následně uložena.

6.7.1 Obsloužení doručených e-mailů

V systému má obsloužení pouze jednoduché rozhraní, které slouží k listování všemi přijatými a neobslouženými e-maily. Uživateli, který se o toto obsloužení stará, je umožněn jak náhled na veškerá získaná data (včetně souborů), tak i možnost provádět jejich korekce. Na závěr může e-mail přiřadit ke konkrétnímu scénáři a spustit jeho vyřízení.

6.7.2 Rozhraní pro komunikaci

Pro vytvoření podání stačí z libovolné e-mailové adresy odeslat zprávu (klidně i s přílohami) na předem definovanou adresu (v případě této práce je to *podatelna@dokflow.eu*). Doručený e-mail je zpracován a jeho odesílateli je obratem zaslána e-mailová notifikační zpráva, ve které je informován o průběhu zpracování.

V této zprávě jsou také přiloženy kompletní pokyny, jak je možné dodatkovat již vytvořené podání. Aby tak mohl uživatel úspěšně učinit, musí splnit několik podmínek:

1. Tou hlavní, bezpečnostní, je odeslání dodatku z e-mailové adresy, která byla získána při vytvoření podání. Zpracovávané adresy jsou: adresa odesílatele, vyžádaná adresa odpovědi a adresy příjemců uvedené ve veřejné kopii (CC). Všechny tyto adresy[21] mohou podání bezpečně dodatkovat.
2. Systém nedokáže automaticky rozpoznat, jaká část obsahu je určena jakožto zpráva dodatku, z toho důvodu je nezbytně nutné před zprávu umístit speciální identifikátor `===MESSAGE===` a za zprávu `===END_MESSAGE===`.
3. Současně je nutné, aby měl e-mail s dodatkem ve svém obsahu (avšak mimo ohraničenou zprávu) uveden i klíč `===KEY===`, který jednoznačně identifikuje již existující podání.

Při splnění všech tří podmínek je umožněno rozšíření existujícího podání. Obsah zprávy pak může vypadat například následovně:

```
===MESSAGE===
Moje zpráva přiložená k tomuto e-mailu.
===END_MESSAGE===
===KEY===0C4E2AAA-3CA5-4EF0-B663-4ED3E6EB37C1===EKEY===
```

Obecně platí, že nejúčinnější cestou, jak podání dodatkovat, je odpověď na systémem vygenerovaný e-mail, kdy uživatel pouze vykopíruje speciální tag `===MESSAGE===`, který je uveden na konci e-mailu v poznámce a vloží ho nad původní zprávu.

[21] Povolené adresy k dodatkování podání jsou uvedeny na konci samotného e-mailu.

ZÁVĚR

V teoretické části práce byly popsány obecné principy fungování podnikových procesů, které následně stály u vzniku konkrétních požadavků na vývoj v praktické části popisovaného informačního systému. Ten nese pracovní název DokFlow a slouží pro modelování a obsluhu podnikových procesů a rozhodovacích postupů.

Hlavním cílem bylo vytvořit systém, který zvládne zpracovat i komplexnější procesy, jejichž data získá prostřednictvím e-mailové schránky. Tohoto i všech dalších stanovených cílů bylo úspěšně dosaženo. Systém je tak tvořen jednoduchým a intuitivním uživatelským rozhraním, které umožňuje procesy separovat na jednotlivé fáze (kroky). U každé definované fáze je možné deklarovat pravidlo pro schválení a chování při v případě zamítnutí.

Podstatné je zmínit, že funkce systému byly navíc mírně rozšířeny nad samotný rámec zadání této práce. Tím se informační systém stal více univerzálním pro business použití. Každý uživatel tak může vytvářet své vlastní procesy přímo v aplikaci a nemusí spoléhat pouze na e-mailovou schránku.

Též i já, autor této práce, jsem dosáhl cíle, který jsem si stanovil. Zdokonalil jsem se ve vývoji webových informačních systémů a rozšířil své obzory o nové užitečné technologie, které mi v budoucnu usnadní mnoho práce při dalším vývoji webových aplikací.

V budoucnu bych se rád zaměřit na rozvoj funkcí, které by uživatelům poskytly větší komfort při používání systému. Jednou z takových je možnost kompletního převodu práv stávajícího uživatele na jiného (u všech probíhajících nebo nadcházejících akcí). Tato funkce by umožnila jednoduchou cestou řešit změny zaměstnanců na firemních pozicích.

Za dodělání by též stálo automatické pozastavení procesu v případě jeho zamítnutí. Po detekci takové situace by vyřízení vyčkalo, až patřičná osoba dodá přepracovaná data, aby mohl proces dále pokračovat. Toto rozšíření vyžaduje značné přepracování algoritmu řízení průběhu procesu.

Mnoho administrátorů systému by též uvítalo blacklist, popřípadě i whitelist e-mailových adres odesílatelů zpráv do podatelny. Ten by byl aplikován při automatickém zpracování doručené korespondence.

POUŽITÁ LITERATURA

[1] ŘEPA, Václav. *Podnikové procesy: procesní řízení a modelování.* Praha: Grada, 2006. Management v informační společnosti. ISBN 80-247-1281-4.

[2] TALLYFY INC. Tallyfy. *Amazingly easy workflow and process management software* [online]. Copyright © 2020 [cit. 2020-10-08]. Dostupné z: https://tallyfy.com

[3] M-FILES INC. M-Files. *Intelligent Information Management Solutions* [online]. Copyright © 2020 [cit. 2020-10-08]. Dostupné z: https://www.m-files.com

[4] PROCESS STREET INC. Process Street. *Checklist, Workflow and SOP Software* [online]. Copyright © 2020 [cit. 2020-10-08]. Dostupné z: https://www.process.st

[5] DLPNG. *Free PNG Images and Free Vectors Graphics.* DLPNG.com [online]. [cit. 2020-10-12]. Dostupné z: https://dlpng.com/png/6523110

[6] SAP SE. *SAP Software Solutions, Business Applications and Technology* [online]. Copyright © 2021 [cit. 2021-01-15]. Dostupné z: https://www.sap.com

[7] ČSN EN ISO 9001. *Systémy managementu kvality – Požadavky.* Praha: Úřad pro technickou normalizaci, metrologii a státní zkušebnictví, 2016.

[8] HAMMER, Michael, James Champy. *Reengineering the Corporation: A Manifesto for Business Revolution.* New York: Harper Business Essentials, 2006. ISBN 978-0-06-055953-3.

[9] GÁLA, Libor, Jan POUR a Zuzana ŠEDIVÁ. *Podniková informatika. 2.*, přeprac. a aktualiz. vyd. Praha: Grada, 2009. Expert (Grada). ISBN 978-80-247-2615-1.

[10] BUBENÍČEK, Jaroslav. *Stručná historie průmyslových robotů.* ElektroPrůmysl.cz [online]. Copyright © 2011 [cit. 2021-03-02]. Dostupné z: https://www.elektroprumysl.cz/automatizace/strucna-historie-prumyslovych-robotu

[11] ONDROUŠEK, Tomáš. *Zlepšení podnikového procesu.* Brno, 2010. Bakalářská práce. Masarykova univerzita. Vedoucí práce Ing. Ondřej ČÁSTEK.

[12] *Metadata: Your File's Hidden DNA and You. Backblaze Blog* [online]. Copyright © 2021 [cit. 2021-03-19] Dostupné z: https://www.backblaze.com/blog/metadata-your-files-hidden-dna-and-you

[13] *The 9 Essential Elements of Modern B2B SaaS Applications.* Process Street. [online]. Copyright © 2021 [cit. 2021-03-19] Dostupné z: https://www.process.st/b2b-saas-applications

[14] POWERS, David. *PHP 7 Solutions: Dynamic Web Design Made Easy.* 4th ed. Powers: Apress, 2019. ISBN 978-1-4842-4338-1.

[15] PHP: *Releases.* PHP: Hypertext Preprocessor [online]. Copyright © 2001 [cit. 2021-04-01]. Dostupné z: https://www.php.net/releases

[16] PHP: *PHP 8.0.0 Release Announcement.* PHP: Hypertext Preprocessor [online]. Copyright © 2001 [cit. 2021-04-16]. Dostupné z: https://www.php.net/releases/8.0/en.php

[17] POTENCIER, Fabien. *Symfony 5: The Fast Track.* 1st ed. Paris: Sensio Sa, 2019. ISBN 978-2-918-39037-4.

[18] GAMMA, Erich. *Design patterns: elements of reusable object-oriented software.* 2015. Boston: Addison-Wesley, 1995. ISBN 978-0-201-63361-0.

[19] *Symfony vs Laravel: A detailed comparison of the best PHP frameworks* [online]. Copyright © 2020 [cit. 2021-04-25]. Dostupné z: https://www.valuecoders.com/blog/technology-and-apps/symfony-vs-laravel-php-framework-choose

[20] API Platform: *REST and GraphQL framework on top of Symfony and React.* API Platform. [online]. Copyright © 2021 [cit. 2021-05-03]. Dostupné z: https://api-platform.com

[21] DUCKETT, Jon. *JavaScript and JQuery: interactive front-end web development.* Indianapolis: Wiley, 2014. ISBN 9-781-118-53164-8.

[22] React – *A JavaScript library for building user interfaces.* React [online]. Copyright © 2021 Facebook Inc. [cit. 2021-05-04]. Dostupné z: https://reactjs.org

[23] FIELDING, Roy. *Architectural Styles and the Design of Network-based Software Architectures* [online]. Irvine, Copyright © 2000 [cit. 2021-05-05] Dostupné z: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm Disertační práce, University of California

[24] *What are the Most Popular Relational Databases.* C# Corner – Community of Software and Data Developers [online]. Copyright © 2021 [cit. 2021-05-05]. Dostupné z: https://www.c-sharpcorner.com/article/what-are-the-most-popular-relational-databases

[25] MySQL – *Support and Licensing – MySQL reference manual book.* O'Reilly Media – Technology and Business Training [online]. Copyright © 2021 [cit. 2021-05-06]. Dostupné z: https://www.oreilly.com/library/view/mysql-reference-manual/0596002653/ch01s04.html

[26] Heroku Development Languages: *Programming Languages Heroku supports.* Heroku – Cloud Application Platform [online]. Copyright © 2021 [cit. 2021-05-06]. Dostupné z: https://www.heroku.com/languages

[27] AJAX Introduction – *W3Schools Online Web Tutorials* [online]. Dostupné z: https://www.w3schools.com/js/js_ajax_intro.asp

[28] *Introduction to ASP.NET Core Blazor.* Microsoft Docs [online]. Copyright © 2021 [cit. 2021-05-07]. Dostupné z: https://docs.microsoft.com/cs-cz/aspnet/core/blazor

[29] GROLINGER, Katarina, Higashin WILSON, Abhinav TIWARI a Miriam CAPRETZ. Data management in cloud environments: *NoSQL and NewSQL data stores.* Copyright © 2021 [cit. 2021-05-10]. Dostupný z: https://journalofcloudcomputing.springeropen.com/track/pdf/10.1186/2192-113X-2-22.pdf

[30] *What is an Application Delivery Controller.* Secure Cloud Application Services and Delivery | A10 Networks [online]. Copyright ©2021 [cit. 2021-05-10]. Dostupné z: https://www.a10networks.com/blog/what-is-an-application-delivery-controller

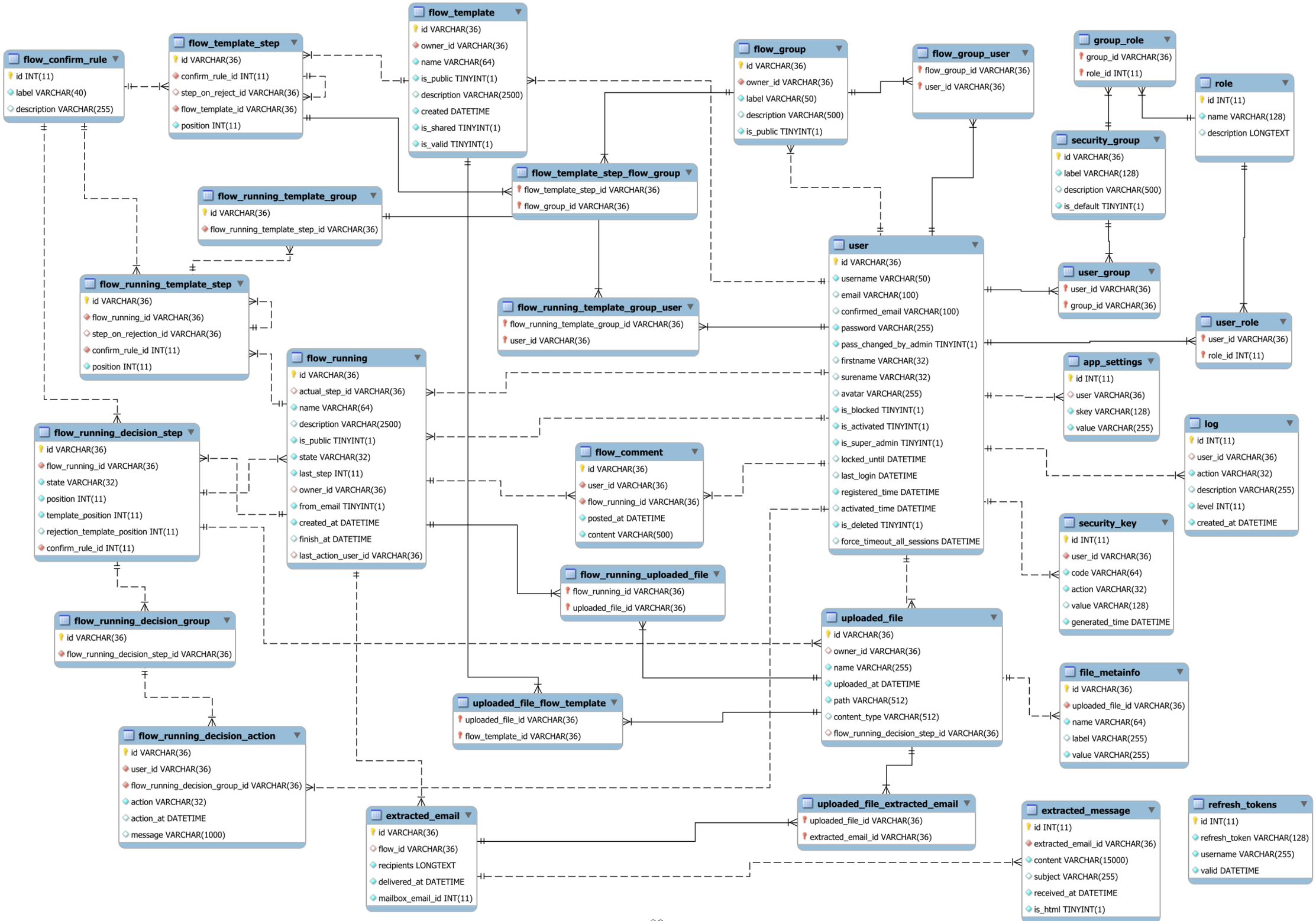
[31] KRUG, Steve. *Don't make me think, revisited: a common sense approach to web usability.* San Francisco: New Riders, 2014. ISBN 978-0-321-96551-6.

[32] *Mobile Screen Resolution Stats Worldwide: StatCounter Global Stats* [online]. Copyright © 1999 [cit. 2021-05-15]. Dostupné z: https://gs.statcounter.com/screen-resolution-stats/mobile/worldwide

[33] MACHÁČ, Martin. *Průběžná integrace a průběžná dodávka/nasazení projektu KYPO* [online]. Brno, Copyright © 2018 [cit. 2021-05-22]. Dostupné z: https://theses.cz/id/b85q8r/. Diplomová práce. Masarykova univerzita, Fakulta informatiky. Vedoucí práce RNDr. Daniel Tovarňák, Ph.D.

SEZNAM PŘÍLOH

PŘÍLOHA A – MODEL DATABÁZE



PŘÍLOHA B – CD

K práci je přiložen CD nosič, který obsahuje:

- tuto prací v digitálním formátu,
- zdrojový kód backendové aplikace,
- zdrojový kód frontendové aplikace.