UNIVERZITA PARDUBICE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

DIPLOMOVÁ PRÁCE

2024 Bc. Tomáš Dokoupil

Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Multiplatformní mobilní aplikace pro řízení IT projektů

Bc. Tomáš Dokoupil

Diplomová práce

2024

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2023/2024

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Tomáš Dokoupil**
Osobní číslo:	**I22155**
Studijní program:	**N0613A140007 Informační technologie**
Téma práce:	**Multiplatformní mobilní aplikace pro řízení IT projektů**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování

Diplomová práce se zabývá tvorbou a implementací multiplatformní mobilní aplikace pro řízení IT projektů v různých úrovních dodavatelských řetězců a s odlišnými rolemi. V teoretické části práce budou popsány základní principy a specifika spojená s řízením IT projektů a modelování podnikových procesů ve vztahu k IT projektům (proces, workflow, aktivita, událost, aktér, apod.). V praktické části bude vytvořen diagram podnikových procesů (BPD) na jehož základě dojde k analýze podkladů pro vytvoření multiplatformní mobilní aplikace pro řízení IT projektů. Mobilní aplikace bude respektovat UX a zásady tvorby moderních mobilních aplikací.

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

Seznam doporučené literatury:

VON ROSING, Mark, WHITE, Stephen, CUMMINS, Fred, DE MAN, Henk. Business Process Model and Notation—BPMN, in *The Complete Business Process Handbook*. Elsevier, 2015, s. 433–457. doi: 10.1016/B978-0-12-799959-3.00021-5.

BURKE, Rory. *Fundamentals of Project Management*. 2ed. Burke Publishing, 2017. ISBN 9780994149213.

SCHWALBE, Kathy. *Introduction to project management*, Fifth edition. Minneapolis, MN: Schwalbe Publishing, 2015. ISBN: 978-0-12-799959-3. DOI: 10.1016/B978-0-12-799959-3.00021-5

Vedoucí diplomové práce: **Ing. Monika Borkovcová, Ph.D.**
Katedra informačních technologií

Datum zadání diplomové práce: **8. listopadu 2023**
Termín odevzdání diplomové práce: **17. května 2024**

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 2023

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., autorský zákon, 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, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 17. 05. 2024

Bc. Tomáš Dokoupil

Poděkování

Rád bych poděkoval paní Ing. Monika Borkovcová, Ph.D. za trpělivost, cenné rady a odborný dohled při konzultacích na zpracování této diplomové práce. Také bych rád poděkoval své rodině za trpělivost a podporu při psaní práce.

ANOTACE

Diplomová práce se zabývá tvorbou a implementací multiplatformní mobilní aplikace pro řízení IT projektů v různých úrovních dodavatelských řetězců a s odlišnými rolemi. V teoretické části práce budou popsány základní principy a specifika spojená s řízením IT projektů a modelování podnikových procesů ve vztahu k IT projektům (proces, workflow, aktivita, událost, aktér, apod.). V praktické části bude vytvořen diagram podnikových procesů (BPD) na jehož základě dojde k analýze podkladů pro vytvoření multiplatformní mobilní aplikace pro řízení IT projektů. Mobilní aplikace bude respektovat UX a zásady tvorby moderních mobilních aplikací.

KLÍČOVÁ SLOVA

Multiplatformní mobilní aplikace, BMPN, řízení IT projektů

TITLE

Multiplatform mobile application for IT project management

ANNOTATION

The thesis deals with the creation and implementation of a multi-platform mobile application for IT project management at different levels of supply chains and with different roles. The theoretical part of the thesis will describe the basic principles and specifics associated with IT project management and business process modelling in relation to IT projects (process, workflow, activity, event, actor, etc.). In the practical part, a Business Process Diagram (BPD) will be created to analyse the basis for the creation of a multi-platform mobile application for IT project management. The mobile application will respect the UX and the principles of modern mobile application development.

KEYWORDS

Multiplatform mobile applications, BMPN, IT project management

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM ZDROJOVÝCH KÓDŮ

SEZNAM ZKRATEK

BPD	Business Process Diagram
BPM	Business Process Management
HRMS	Human Resources Management System
CRM	Customer Relationship Management
ERP	Enterprise Resource Planning
BPMN	Bussiness Process Model and Notation
XML	Extensible Markup Language
WBS	Work Breakdown Structure
IPMA	International Project Management Association
MMA	Multiplatformní mobilní aplikace
AI	Artificial Intelligence
MAUI	Multi-platform App UI
HTTP	Hypertext Transfer Protocol
XAML	Extensible Application Markup Language
GPS	Global Positioning System
JSON	JavaScript Object Notation
ITIL	Information Technology Infrastructure Library
API	Application Programming Interface
REST	Representational state transfer
EF	Entity Framework
UI	User Interface
UX	User Experience
DTO	Data Transfer Object
JWT	JSON Web Token
IT	Informační technologie
CRUD	Create, Read, Update, Delete

ÚVOD

V budoucnu se stane řízení IT projektů ještě složitějším úkolem, vyžadujícím inovativní přístupy. Tato diplomová práce poskytne základní uvedení do problematiky s cílem navrhnout, implementovat a otestovat multiplatformní mobilní aplikaci pro řízení IT projektů. V rámci rostoucí potřeby mobilních řešení v oblasti IT se práce zaměřuje na vytvoření aplikace, která bude odpovídat moderním potřebám projektových týmů. Aplikace tak reaguje na trend 21. století spočívající ve vysoké mobilitě, což umožní uživatelům řídit své projekty odkudkoliv a kdykoliv. Aplikace se bude zaměřovat na efektivní správu zdrojů, plánování a monitorování průběhu projektů přímo z mobilních zařízení. Praktický výstup práce bude tvořen tak, aby splnil moderní požadavky na vývoj mobilních aplikací.

Důraz v této práci je kladen na pochopení procesů, workflow, aktivit, událostí a rolí. Mapování těchto procesů se často vyskytují ve vývojových softwarových týmech při řešení IT projektů. V práci dojde k zpracování diagramů a návrhu optimalizace pro aplikaci. Výsledek tohoto zkoumání se také stane podkladem pro analýzu požadavků vyvíjené multiplatformní aplikace.

Práce je rozdělena do několika kapitol, které postupně povedou čtenáře od teoretických východisek až k praktickému návrhu a popisu implementace aplikace. V teoretické části budou definovány klíčové pojmy a principy řízení IT projektů, modelování podnikových procesů a použitých technologií při vývoji multiplatformní aplikace.

Praktická část se soustředí na vytvoření diagramu podnikových procesů, který bude sloužit jako základ pro analýzu a následný návrh mobilní aplikace. V této části je kladen důraz na uživatelskou přívětivost (UX) a na integraci osvědčených postupů moderního designu mobilních aplikací. Výstupem praktické části bude funkční prototyp aplikace. Tento prototyp bude demonstrovat klíčové funkcionality potřebné pro správu IT projektů a bude testován s ohledem na uživatelskou spokojenost a efektivitu práce.

Mobilní aplikace bude otestována a budou popsány procesy a metody použité při testování výstupní aplikace. Testování se bude zaměřovat na testování nejmenších funkčních jednotek kódu, jako jsou jednotlivé endpointy API a integrační testy se budou soustředit na testování funkčnosti celých komponent nebo skupin komponent společně. V závěru práce bude zhodnocen stávající stav mobilní aplikace a navrženy další možnosti vývoje a přizpůsobení aplikace stále se rozvíjejícím potřebám při řízení IT projektů.

1 BPM

Business Process Management (dále jen BPM) je disciplínou, která zahrnuje použití různých metodik pro zkoumání, navrhování, kontrolu, měření, zlepšování a zdokonalování podnikových operací. Organizuje interakce mezi jednotlivci, systémy, daty a aktivitami tak, aby se dosáhlo požadovaných obchodních výsledků v souladu s cíli organizace.[1] BPM představuje ucelení principů pro vedení a řízení podnikových procesů.[47]

1.1 Typy BPM

Existují 3 typy BPM, které se liší na základě účelu, ke kterému se využívají.[2],[4]

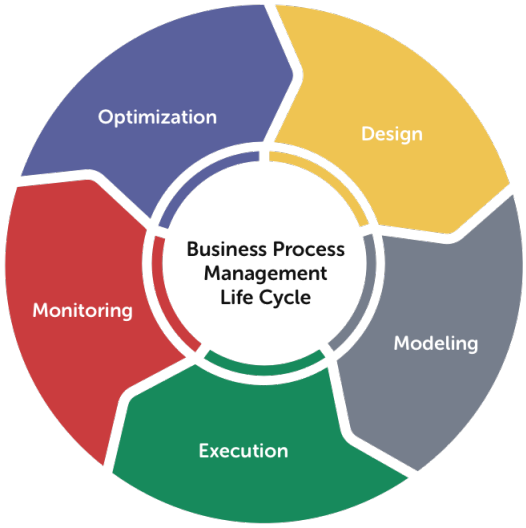
- **Integration-centric BPM** (někdy též nazývaný System-Centric BPM [4]) je typ systému, který se pro řízení obchodních procesů specializuje na ty procesy, které fungují pomocí existujících obchodních systémů, jako jsou systémy pro správu lidských zdrojů (HRMS), správu vztahů s klienty (CRM) nebo systémy plánování podnikových prostředků (ERP), a nepotřebují mnoho manuální práce.
- **Human-centric BPM** se, na rozdíl od integračně zaměřeného BPM, zaměřuje na lidskou účast, obvykle tam, kde jsou vyžadovány schválení. BPM zaměřený na člověka klade lidi na první místo, podporovaný různými funkcemi automatizace. Jedná se o procesy, které jsou převážně prováděny lidmi, a automatizace je nemůže snadno nahradit. Tyto procesy často zahrnují mnoho schvalování a úkolů prováděných jednotlivci.
- **Document-centric BPM**. Tento typ BPM se zaměřuje na konkrétní dokument, jako je například smlouva. Když společnosti nakupují produkt nebo službu, je potřeba projít různými formuláři a koly schvalování, aby se vypracovala dohoda mezi klientem a dodavatelem. Tento proces zahrnuje různé kroky, které vedou k vytvoření a schválení smlouvy mezi oběma stranami.

1.2 Životní cyklus

Pro úspěch systému je nutné definovat fáze spojené s pracovním postupem. Pracovní tým s využitím tohoto postupu může lépe identifikovat oblasti, které jsou vhodné pro

zlepšení a také sledovat ukazatele pokroku. Aby organizace využila všech výhod spojených s životním cyklem, je nejdříve nutné jej pochopit. Životní cyklus projektů zahrnuje[3],[4]:

1. **Process design:** Prvním krokem v BPM je vymezení důležitých událostí v procesu. Z této operace jsou poté identifikovány jednotlivé úkoly v rámci celého procesu. Pro každý úkol je také určen vlastník pro jeho workflow. Je důležité kroky definovat přesně, aby se mohly identifikovat oblasti pro optimalizaci a metriky pro zlepšení.
2. **Model:** Vizualizace procesního modelu je druhým krokem. Pro kvalitní model je nutné, aby zahrnoval určité detaily, příkladem mohou být popisy úkolů, toky dat v procesu nebo časové osy. Během této fáze je užitečné využívat software pro řízení procesů.
3. **Execute:** V této fázi se na základě modelu začne spouštět a testovat nový systém. Využívá se tzv. proof of concept. V počátku je do tohoto testování zahrnuta pouze omezená skupina, ale s úpravou po zpětné vazbě se proces rozšíří na širší společnost.
4. **Monitor:** Během fáze dochází ke sledování procesu, dochází k různým měřením efektivity a identifikuje se tzv. bottleneck, tedy úzké místo, které v procesu způsobuje například zbytečné čekání.
5. **Optimize:** Posledním krokem je aplikace úprav procesu ke zlepšení. Po tomto kroku může opět BPM následovat první krok.



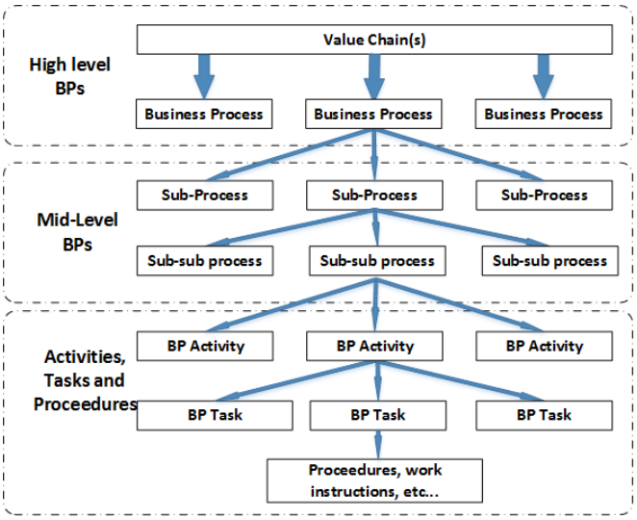
Obrázek 1: BPM životní cyklus procesu (zdroj [4])

1.3 Hierarchie procesů

Správně vytvořený procesní model pro společnost je důležitým základem pro správnou tvorbu diagramů, pochopení celku a hlavně strukturu organizace. Spojování procesů do hierarchie se sub-processy až na úroveň aktivit dává společnosti přehled nad tím, jak fungují jejich procesy. Samotná hierarchie pak slouží převážně k prioritizaci, optimalizaci a vylepšování částí, které jsou nejvíce kritickými.[13]

Hierarchie se dělí na 3 základní úrovně. První je High-level, druhou Mid-level a třetí se nazývá Tasks and Procedures. [13] V některých případech zpracování je možné nalézt dělení na 4 až 5 úrovní.[14]

- **High-level** označuje hlavní problémy podnikové architektury. Spolupráci a sdílení zodpovědnosti mezi odděleními v organizaci a funkčními jednotkami. V této části je také Value chain dané organizace, který reprezentuje hlavní obchodní linie.
- **Mid-level** je specificky určený pro každou organizaci a také v závislosti na komplexnosti všech daných procesů a sub-procesů v orgnizaci. Na této úrovni jsou už procesy a sub-procesy, které se nepřímo váží na Value chain. Primární zaměření je na jednotlivá oddělení, optimalizaci a automatizaci.
- **Tasks and Procedures** zastupuje level nejnižší úrovně, kde se vyskytují aktivity, které není dále možné dělit. Tyto aktivity poté popisuje workflow diagram.



Obrázek 2: Hierarchie procesu ve třech úrovních (zdroj [5])

1.3.1 Výhody použití hierarchie

Hierarchie přináší několik výhod při použití v organizaci.[13]

- Hierarchie poskytuje informace o tom, jak jdou úkoly za sebou v rámci organizace. To pomáhá lépe porozumět obchodním procesům organizace, zvyšuje odpovědnost zaměstnanců a umožňuje identifikovat lidi odpovědné za klíčová rozhodnutí.
- Využití principu hierarchie procesů slouží jako struktura dokumentace pro řízení procesů, protože následuje princip od abstrakce ke konkrétnímu řešení.
- Na rozhodnutí a souhlas s úrovněmi hierarchie je nutná přítomnost všech zúčastněných stran, proto je zajištěno, že nedojde k nedorozumění a nesouladu.
- Pomáhá organizaci lépe využívat a rozdělovat své zdroje pro snížení ztrát a dosažení firemních cílů.
- Jedná se o přístup, který umožní celé organizaci přehled a identifikuje možné budoucí procesy, včetně propojení s celou organizací.

1.4 Elementy

Prvním elementem v hierarchii je proces. Proces v hierarchii procesů je částí nebo úrovní celkové struktury procesů. Každý proces v této hierarchii a obvykle spadá pod nějaký větší rámec procesů. Díky tomu, že procesy jsou organizovány do hierarchické struktury, je snazší je spravovat a optimalizovat na různých úrovních složitosti.[6],[7]

Sub-proces v hierarchii procesů je podproces nebo pod-úroveň v rámci celkového procesu. Proces samotný může být rozdělen tedy na sub-proces a ten sám přebírá určité vlastnosti procesu, včetně cílů a kroků nebo úkolů. Sub-procesy jsou využívány k tomu, aby nám pomohly lépe organizovat a strukturovat hlavní proces na menší a snadněji spravovatelné části. Jsou součástí celkového procesu a mohou být dále rozděleny na ještě menší kousky nebo detailnější úkoly podle toho, co potřebujeme.[6],[7]

Task nebo procedura v hierarchii procesů je konkrétní aktivita, který je součástí celkového procesu. Je to již nedělitelná složka hierarchie. Task nebo procedura může být jakýmkoliv úkonem, který musí být vykonán v rámci procesu. Tyto úkoly jsou typicky detailněji specifikovány a popsány, aby bylo jasné, co je třeba udělat a jakým způsobem. Většinou se k popisu úkolu modeluje workflow těchto aktivit.[6],[7]

1.5 Workflow

Workflow je popis pro nejnižší úroveň hierarchie procesů. Představuje strukturovaný postup nebo sekvenci kroků, které musí být provedeny k dosažení určitého cíle. Jedná se o systematické uspořádání činností, které určuje, jaké úkoly musí být vykonány a v jakém pořadí. Workflow je často vizualizován pomocí diagramů, které znázorňují jednotlivé kroky a spojení mezi nimi. Cílem workflow je zajistit efektivní a konzistentní provádění činností, minimalizaci chyb a zlepšení celkového výkonu procesu. Jedním ze způsobů zápisu workflow je jazyk BPMN.[8]

2 PRINCIPY BPMN

Bussiness Process Model and Notation (dále jen BPMN) je grafická notace, která se používá k modelování procesů za pomocí diagramů. Účelem je poskytnout standardizovaný způsob, jak popsat a vizualizovat různé podnikové procesy, a to od jednoduchých úloh až po komplexní provádění. BMPN jako notace je také zapsaným standardem pro modelování podnikových procesů jako norma ISO/IEC 19510:2013.[10]

Hlavním cílem je vytvořit normu, která bude jednoduchá a srozumitelná pro uživatele napříč podnikovými procesy. Od podnikových analytiků, přes technické vývojáře až po jednotlivé zaměstnance, které budou tento proces řídit nebo budou jeho součástí. Jedná se tedy o způsob propojení mezi všemi těmito vrstvami v podnikovém modelu.[8]

BPMN bylo vyvinuto skupinou Business Process Management Initiative (BPMI) v roce 2004. V roce 2005 však společnost BPMI a také normu BPMN převzala organizace Object Management Group (OMG), která v aktuální době tuto normu vlastní a spravuje. [8]. Notace prošla několika úpravami od svého vzniku a v roce 2014 vznikla verze BPMN 2.0, která je do doby psaní této práce uznávána. Norma ISO/IEC 19510:2013 je identická pro BPMN 2.0 společnosti OMG.[10]

OMG poskytuje rozsáhlou dokumentaci k modelování v BPMN, která obsahuje více než 500 stránek a řadu ukázek, jak správně modelovat různé situace. Společnost také nabízí na svých webových stránkách kurzy pro tuto notaci s možností zisku certifikátů po dokončení těchto kurzů. Společně s tím nabízí i různé návody a stručného průvodce této notace.[12]

Jedním z cílů BPMN je také umožnit vizualizaci různých formátů jazyka XML. Tato funkce umožňuje snadnou integraci BPMN diagramů s dalšími systémy a nástroji, což zvyšuje jejich univerzálnost a praktičnost pro použití v reálných podnikových prostředích.[8] Částí popsání jazyka BMPN ve formátu XML se ale autor v této práci rozhodl nezabývat do větších detailů.

Pro pochopení jednotlivých bloků a jejich funkcí pro celý podnikový proces je nutné také proniknout hlouběji do slovníku BPMN a pochopit jednotlivé termíny, které jsou v práci obsaženy. Pochopení termínu představuje důležitou roli ve schopnosti využít nejen tento popis, ale také popis BPMN diagramů v jakémkoliv textu. A samotné porozumění má také vliv na celkovou schopnost efektivní práce a modelování diagramu.

2.1 Process

V BPMN je podnikový proces základním prvkem, který je popisován diagramem. Obsahuje sekvenci aktivit, událostí, rozhodnutí, toků a dalších objektů, které tvoří obchodní operace nebo činnost v organizaci.[33] Model samotný umožňuje společně se všemi částmi diagramu plné pochopení jednotlivých kroků, jak jsou kroky spuštěny, jak jdou za sebou, jakými způsoby je možné přejít na další krok v toku. Klíčové je, že toto je umožněno právě konstrukcí přesných BPMN diagramů.[8]

Proces může být firemní postup pro určitou operaci nebo také jednoduchý proces denní činnosti jednoho zaměstnance. Tyto diagramy lze poskládat dohromady a tím vzniká komplexní spolupracující projekt, který využívá objekty jako Pool a Lines. BPMN vnímá pojem proces jako "set of flow elements".[8]

Součástí procesu jsou jednotlivé aktivity, které jsou potřeba vykonávat při provádění procesu. Jsou buď atomické nebo ne-atomické (označované také jako sloučeniny[8]). Aktivitou je samotný proces, ale také sub-process nebo task.[8]

2.2 Flow

Tok dat je v BPMN vizualizován za pomocí čar mezi objekty. Pro BPMN 2.0 je běžné několik typů tohoto toku. Sequence flow, který je nejčastější a ukazuje zpracování jednotlivých úkonů, jak jdou za sebou. Každý jeden tok má právě jeden začátek a právě jeden konec. Dalším typem je například Message flow, který v diagramu reprezentuje tok zpráv z příslušných BPMN objektů. Ale najdeme zde také Compensation flow, který ukazuje, jaké jednotlivé aktivity jsou nutné provést při rollback transakce. V BPMN nechybí ani Exception flow. Ten naznačuje právě tok aktivit při vyvolání vyjímky v procesu.[8]

2.3 Lines

Linie je způsob rozdělení procesu do více úrovní (většinou v rámci objektu Pool. Rozšíření je možné buď horizontální nebo vertikální. V rámci lepšího pochopení je pro každou linii určen text. Ať už se jedná o jméno nebo určitý atribut procesu. Tento text je možné umístit jakýmkoliv směrem, v jakémkoliv místě uvnitř dané linie. Konečné umístění je záležitostí firmy nebo zodpovědné osoby, která diagram tvoří. Ve valné většině se ale můžeme setkat

s tím, že v případě horizontální linie je tento text umístěn kolmo na levé straně. V případě vertikálních linií je poté na horní straně. Pro oddělení tohoto textu a názvu objektu Pool se využívá toho, že objekt má jméno odděleno čárou.[8]

2.4 Collaboration

Colaborace v BPMN označuje proces interakce mezi dvěma nebo více entitami, které spolupracují. V grafické podobě podnikového procesu termín označuje a definuje vztah pro komunikaci většinou v objektu Pool. Spolupráci v diagramu lze najít snadno, protože pokud jsou v diagramu více, než dva objekty typu Pool a mají vzájemnou interakci, pak se jedná o Spolupráci.[8],[9]

2.5 Choreography

Jedná se o způsob zápisu interakce mezi různými subjekty nebo účastníky v podnikovém procesu. Choreografie dává důraz na vzájemné vztahy a komunikaci mezi účastníky bez řízení centrálního prvku. V BPMN se reprezentuje pomocí choreografických diagramů. Ten modeluje způsob, jak spolu subjekty komunikují při provádění podnikových procesů. Tento typ diagramu ukazuje, jak se proces vyvíjí z pohledu subjektů, ne pouze jednoho centrálního řídícího prvku. Ve zkratce choreografie poskytuje globální pohled na spolupráci mezi subjekty v podnikovém procesu a přibližuje další pochopení a porozumění procesu.[8],[9]

2.6 Tvorba diagramu

Při modelování workflow se využívá několik druhů symbolů. Pro události, které se mohou stát v průběhu podnikového procesu se využívají symboly Event. Pokud je nutné workflow větvit paralelně nebo je nutný rozhodovací proces, využívá se Gateway. Poté můžeme modelovat činnost několika způsoby, sub-process pro složitější a Task pro takové, které nelze nebo není potřeba dále dělit na menší části. Pro modelování aktérů zodpovědných za tyto činnosti jsou pak využívány symboly Pool a Lines. Jak fungují a také jak se spojují dohromady je obsahem následujících kapitoly.

3 OBJEKTY BPMN

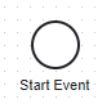
Pro tvoření podnikových procesů s použitím BPMN je nutné pochopit základní prvky, ze kterých se model sestavuje. Jelikož BPMN disponuje grafickou notací pro jednotlivé prvky, není složité po pochopení několika z nich začít modelovat proces. Prvky jsou od sebe rozlišitelné graficky a také každý má svoji specifickou roli v procesu. BPMN 2.0 (aktuální verze) obsahuje více než 100 jedinečných grafických notací.[15],[16] Nicméně běžné firmy používají pro vytvoření samotného modelu procesu okolo 20 % těchto notací.[15]

3.1 Event

Event označuje událost, která se stane během procesu. Eventy ovlivňují průběh daného procesu. Samotné označení Event je něco, co v BPMN pokrývá několik objektů. Některé procesy se dají dělit dle umístění daného eventu, což znamená na počátek procesu, v rámci jeho průběhu a na jeho ukončení.[8] V následující části jsou popsány jednotlivé vybrané eventy, jejich symboly a ukázky použití.

3.1.1 Start Event

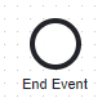
Start event je důležitou součástí BPMN modelování. Vytváří novou instanci daného procesu, který chce uživatel modelovat. Z počátku tohoto eventu vychází poté cesta, po které proces pokračuje. V jednom modelu může být start eventů více[9], ale pokud je v běhu již jeden token, pak spuštění jiného start eventu nevytváří další token. Start event se značí kruhem s jednoduchou čárou a dle typu značkou eventu uvnitř tohoto kruhu.[8],[11]



Obrázek 3: Start event (zdroj vlastní)

3.1.2 End Event

End Event je eventem, který proces ukončuje. Protože je konečným eventem, pak z něj nelze uvádět žádné spojení do dalších částí. Pro správný význam End eventu si lze představit, že Start event vygeneruje při spuštění token, který projde celým procesním modelem a právě end event tento token přijme, tím je zaručeno, že se ukončí proces dané úrovně. Před plnohodnotným dokončením procesu je nutné všechny vygenerované tokeny doručit do end eventů. V jednom procesu může být více než jeden end event v případě více možností, jak daný proces může skončit. End event se značkou ukončení je značen kruhem s tlustou čárou.[8]



Obrázek 4: End event (zdroj vlastní)

3.1.3 Intermediate Event

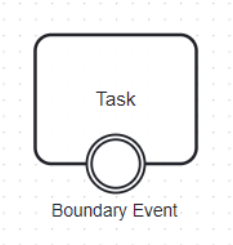
Třetím typem eventu je Intermediate. Značí událost, která se může stát v průběhu procesu mezi Start a End event. Tento typ nemůže sám započít spuštění procesu, ale také ho neukončí. Event je značen kruhem se značkou uvnitř, která upřesňuje tento event a dvojitou čarou.[8]

Obrázek 5: Intermediate Event (zdroj vlastní)

3.1.4 Boundary Event

Boundary typ eventu je speciálním typem, který se využívá ke zpracování výskytu jiné události a zrušení aktivity či daného eventu. Pokud má být aktivita zrušena, je tak učiněno v atributu cancelActivity. To je většinou v místě, kdy v aktivitě nedošlo ke splnění podmínky. Při zrušení je vygenerován token, který navazuje po ukončení dané události

v aktivitě a tím může dál pokračovat průběh.[8] Samotný typ není možné použít, využívá se ve spojení jako typ dalších eventů.



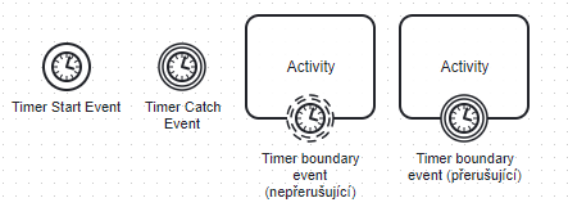
Obrázek 6: Boundary Event (zdroj vlastní)

3.1.5 Timer Event

Tento event může být typu Start, typu Catch a typu Boundary. V případě startovního eventu se využívá ke spuštění procesu v intervalech, popřípadě v daném časovém okamžiku. Ve chvíli spuštění tohoto eventu dojde k vytvoření nové instance daného procesu.[17]

Při spuštění v daném časovém okamžiku se event používá pro kontrolu splnění časové podmínky, kdy se proces zastaví a čeká, než uplyne daný čas, aby pokračoval.[17]

Jako poslední možnost použití Timer eventu je zde hraniční event. Lze říci, že jakmile uběhne čas stanovený pro nějakou aktivitu, pak se spustí, aby tuto aktivitu zrušil.[17]



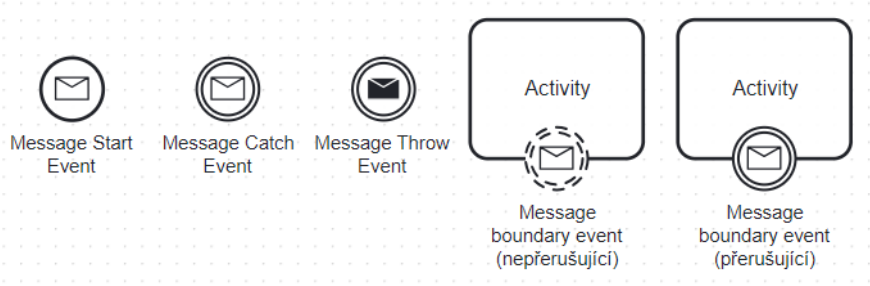
Obrázek 7: Timer eventy (zdroj vlastní)

3.1.6 Message Event

Stejně jako u Timer eventu se využívá tento event v typu Start, který vytvoří novou instanci při zprávě. Ale narozdíl od Timer eventu, má Message event navíc Throw typ. Ten se využívá v případě, kdy chceme v procesu naznačit událost pro poslání zprávy a většinou také pro její doručení.[18]

Doručení zprávy a její zpracování má za úkol Message event typu Catch. V tomto případě se instance daného procesu zastaví do chvíle, než je požadovaná zpráva doručena. Poté pokračuje dál.[18]

Boundary typ message event označuje takový event, který může posílat zprávy z vnitřku aktivity ven. Takových eventů může být na aktivitě několik a zde může být možnost nastavení, že se spuštěním takového eventu dojde k ukončení celé aktivity.[18]

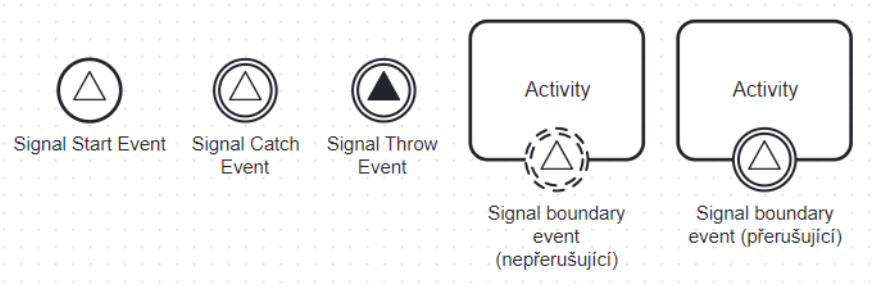


Obrázek 8: Message eventy (zdroj vlastní)

3.1.7 Signal Event

Signal event je eventem, který spouští a vytváří nové instance v případě typu start při zachycení signálu s jeho určeným značením. Využívá se tak k vytvoření několika instancí jednoho procesu najednou skrze jeden broadcast signálu. Ten vysílá event typu Throw.[19]

Tento even má také k dispozici typy boundary i catch. Catch funguje na podobné bázi jako start s tím rozdílem, že zastaví danou instanci procesu a čeká na signál, který je opět vyslán typem Throw. Pro boundary typ eventu platí, že zachycuje signál z vnějšího prostředí aktivity a spouští případě ukončuje určité procesní cesty v dané aktivitě.[19]

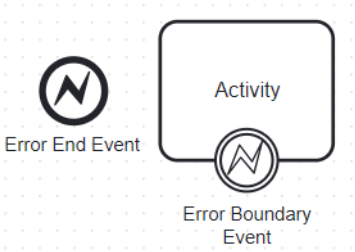


Obrázek 9: Signal eventy (zdroj vlastní)

3.1.8 Error Event

Posledním uvedeným eventem je error event. Má pouze typy End a Boundary. End je uvnitř dané aktivity a v případě chyby dojde k vyvolání události, kterou zachytí typ Boundary a pokračuje poté dál. Pokud dojde k nějaké chybě při průběhu procesu, je možné tuto chybu vyvolat uměle a zamezit pokračování zpracování procesu. V případě nevyužití takových eventů může později při průběhu docházet k takzvaným incidentům.[20]

Je důležité rozlišit rozdíl mezi chybou v procesu a technickou chybou při jejím provádění. V reálném provozu je nutné řešit spoustu technických problémů a chyb, kterými se při modelování nechcete zabývat. Někdy je ale složité rozhodnout, zda je chyba technická nebo procesní. Proto je důležité rozhodnout, která chyba by mohla ovlivnit negativně průběh procesu a takovou ošetřit v modelu.[20]



Obrázek 10: Error eventy (zdroj vlastní)

3.2 Gateway

Objekty typu Gateway používá BPMN pro rozhodování nebo rozdělování či spojování. V modelu se používají právě pro účely, zda je nutné rozdělit tok procesu jedním nebo druhým směrem, popřípadě paralelizovat tento tok. Lze také učinit rozhodnutí na základě události, která má určité parametry pro splnění. V případě rozhodovacího typu, je pak možno rozdělit tok typem AND, OR a XOR.[8],[11]

3.2.1 Exclusive

Exclusive Gateway reprezentuje typ XOR. Jedná se o typ, který může mít několik výstupních bodů a každý jeden bod má svoji podmínku, která určuje, zda tok bude pokračovat tímto směrem. Jakmile dojde k vyhodnocení, je tomuto směru toku předán token. Pokud

je více podmínek splněných, pak bude tok pokračovat prvním, který je splněný. Pokud nebude splněná žádná podmínka, je nutné určit výchozí výstupní bod pro pokračování toku. Pokud není určen tento výchozí výstupní bod, pak dojde k incidentu.[8],[21]

Dalším možným způsobem použití tohoto typu Gateway je, že může spojovat několik vstupních toků do sebe a tím zlepšit čitelnost modelu. Pokud se při spojení toků použije tento typ, pak nedojde ke spojení konkurenčních toků a informací z nich.[21]

Obrázek 11: Exclusive Gateway (zdroj vlastní)

3.2.2 Parallel

Druhým typem je Parallel Gateway, který představuje typ AND. Jak vyplývá z názvu, jeho účelem je rozdělit jeden vstupní tok na více výstupních toků bez kontroly podmínek. Každý tok je pak prováděn nezávisle na ostatních. Každý tok obdrží svůj token, proto je možné provádět tyto toky nezávisle.[8],[22]

Při spojení se paralelně prováděné toky mohou být spojeny zpět skrze použití další Parallel Gateway v modelu. V tomto případě se tok zastaví a čeká, než se dostaví všechny ostatní toky.[22]

Obrázek 12: Parallel Gateway (zdroj vlastní)

3.2.3 Inclusive

Posledním klasickým typem pro vyjádření je OR. K tomu slouží Inclusive Gateway. Tento typ dovolí v modelu provádět více toků při splění jejich podmínek. Jedná se o spojení funkcí z předchozích dvou typů. Každý výstupní bod je vyhodnocován nezávisle a při splnění více výstupních bodů tak dojde k paralelnímu rozdělení toků, kde každý tok získá

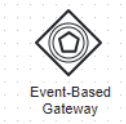
svůj vlastní token. Je zde možné také mít výchozí výstupní bod, který lze využít v případě, že není splněna žádná výstupní podmínka. Pokud objekt nedisponuje výchozím bodem, pak při nesplnění žádné podmínky je vyvolán incident.[8],[23]

V aktuální verzi modelu není možné tento typ použít pro spojení dvou a více toků. Zajištění možného spojení je skrze použití více Gateway typu Exclusive a Parallel.[23]

Obrázek 13: Inclusive Gateway (zdroj vlastní)

3.2.4 Event Based

Event Based Gateway je speciálním případem. Je nutné, aby měl minimálně 2 výstupní body a na nich byl Message, Signal nebo Timer Catch Event. Ve chvíli, kdy tok dojde na tento objekt, pak se zastaví a čeká, až se jeden z eventů spustí. Ve skutečnosti tedy je určení výstupního bodu závislé na jiném účastníkovi modelu. Proto je tento typ speciální tím, že nejde o nastavení samotného objektu, ale o připojení bloků, které následují.[8]



Obrázek 14: Event-based Gateway (zdroj vlastní)

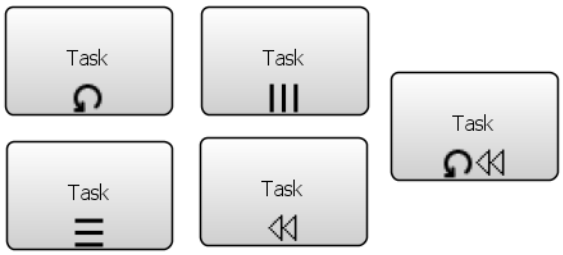
3.3 Task

Task označuje provedení atomické aktivity v procesním workflow. Obvykle se používá v případech, kdy samotnou aktivitu nelze rozdělit na dílčí úkoly a zároveň nelze popsat jiným úkolem. Samotný objekt může mít také svoji značku (tzv. Marker), který určuje posloupnost nebo pravidla vykonávání dané aktivity. Jak napovídají názvy typů jednotlivých tasků, tak se jedná právě o aktivity, které provádí externí entity (koneční uživatelé nebo jiné aplikace).[8]

Značky pro objekty se dělí na 3 základní a v některých dokumentech lze najít ještě 4. typ. Mezi tyto 3 základní patří Loop, Multi-Instance a Compensation. V určitých případech na objektech lze najít také kombinaci značek Loop a Compensation.[8]

V některých pracích je možné nalézt, že typ MultiInstance se dělí na Parallel a Sequencial. Rozdíl je v tom, zda se aktivita vykonává paralelně ve stejný čas, nebo sekvenčně za sebou. Grafický rozdíl je v tom, zda jsou čáry vodorovně nebo svisle.[25] Vodorovné určují paralelní zpracování, svislé zase sekvenční.[8]

- **Loop marker** se využívá při modelování situací, při kterých se aktivita opakuje znovu a znovu. Ukončující podmínka pro opakování se uvádí do anotace nebo v některých případech do atributu daného objektu.
- **MultipleInstance Task** bude v modelu tam, kde stejný úkol zadáme například více lidem a proto nedochází k opakování tohoto úkolu jednou osobou, ale aktivita se vykoná několikrát.
- **Compensation Task** je specifickou verzí úkolu, který je nutné vykonat pouze za podmínky, že se někdy dříve vyskytl jiný vybraný úkol. Nejčastěji se tento typ používá, pokud je nutné kompenzovat nebo zaplatit za předchozí úkoly.



Obrázek 15: Značky pro aktivity (zdroj vlastní)

3.3.1 Service Task

Service Task je prvním typem z druhé skupiny a také nejjednodušším typem. V procesu se přidává tam, kde je potřeba nějaká aktivita, kterou zajistí libovolný pracovník na procesu (pracovníkem může webová služba nebo jiná aplikace [8]). Při vstupu procesu do tohoto tasku pak dojde k vytvoření Service job, který je nutné dokončit. Při vykonávání je zastaven celý proces, který čeká na dokončení. Pro dokončení je nutné, aby daný pracovník potvrdil dokončení.[8],[26]

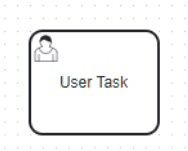
Povinným prvkem je definice pro tento typ tasku. Definice udává dvě zásadní informace. První je type, který udává typ pracovníka pro obsluhu dané aktivity - v definici povinné. Druhým je retries, který slouží pro definici toho, kolikrát může být daná aktivita zopakována v případě, že selže. Druhou informaci definovat nemusíme, ve výchozí hodnotě je nastavena na 3 pokusy.[26]

Obrázek 16: Service Task (zdroj vlastní)

3.3.2 User Task

User Task, jak vypovídá z názvu, zastává v modelu roli aktivity, která je nutná vykonat člověkem s využití aplikace. Stejným způsobem, jako je tomu u Service task, dojde k vytvoření specifického User job, který je nutné splnit určitým uživatelem. Při vykovávání této aktivity je proces pozastaven a znovu pokračuje až ve chvíli, kdy dojde k potvrzení přiřazeným uživatelem.[8],[27]

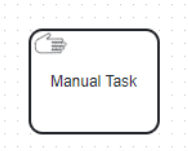
Pro tento typ tasku jsou opět přiděleny specifické definice. Hlavní definicí je přidělení informace, který člověk (nebo skupina) má za úkol tuto aktivitu vykonat a to ve třech verzích. První je assignee, který udává, jaká osoba má přímo vykonat aktivitu. Druhou je candidateUser, ten označuje uživatele, kterému může být přiřazena (popř. může si vzít na starost). Posledním v tomto seznamu je candidateGroups, který vyjmenuje list skupin uživatelů, které jsou způsobilé pro vykonání této aktivity.[27]



Obrázek 17: User Task (zdroj vlastní)

3.3.3 Manual Task

Manual Task je typ, který v modelu naznačuje aktivitu, která je nutná vykonat manuálně. Tedy bez použití aplikací nebo nástrojů pro provádění bussiness pravidel. Zpravidla se využívá tam, kde aktivitu vykonává externí pracovník, o kterém nemusí být v modelu evidence a nepotřebuje přístup k systému. V případě, že proces při vykonávání narazí na tento typ tasku, nedojde k zastavení a dál pokračuje. Jedná se o zvláštní případ, který se v případě automatizovaných procesů nevyužívá.[8],[28]



Obrázek 18: Manual Task (zdroj vlastní)

3.3.4 Undefined Task

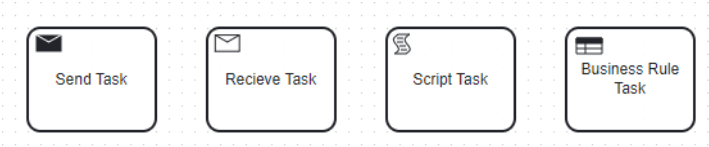
Posledním více popsaným typem je Undefined Task, který v modelu představuje aktivitu, která je abstraktní. Využívá se v případech, kdy nemusí být modelu jasné, jakým způsobem bude tato aktivita vykonána. Nejvíce uplatnění má v případech, kdy není proces automatizovaný. Druhým případem je vývoj modelu, kdy zatím nevíme, kterou aktivitu bude obsluhovat jaká část, proto využijeme tento typ, abychom nechali toto rozhodnutí na později. Ve skutečnosti se tento typ nevyužívá u konečných modelů a pouze nabízí možnost abstrakce pro budoucí řešení.[8],[27]

Obrázek 19: Undefined Task (zdroj vlastní)

3.3.5 Ostatní

Existují také další typy pro objekty Task. Jedná se o Bussiness rule, který udává, že musí dojít k vykonání nějakého podnikového pravidla. Dalším může být Script Task, který

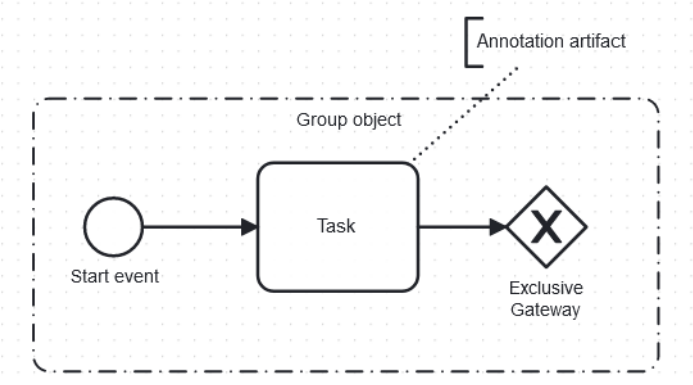
automaticky spustí provádění nějakého scriptu. Posledními typy jsou Send a Recieve Task. U těchto dvou typů se jedná o podobné provedení a logiku jako v případě Message Event s tím rozdílem, že zprávy pro aktivity jsou externě vykonané. Pro obě varianty platí, že proces je zastaven. Ukončujícím pravidlem pro zastavení je pro Recieve příjem zprávy a pro Send potvrzení odeslání. Send Task je velmi podobný Service Task s ohledem na funkčnost.[8]



Obrázek 20: Ostatní objekty Task (zdroj vlastní)

3.4 Artifact

Artifact je v diagramu prvek, který slouží k reprezentaci informací nebo dat relevantních pro daný podnikový proces. Obvykle se používá k zaznamenání dokumentace, popisu nebo dodatečných informací. Samotný artifact není významně spojený s tokem procesu, ale jsou důležité pro porozumění a optimalizaci procesu. Obvykle se v diagramu objevují 2 typy tohoto prvku. Prvním je skupinový objekt a slouží k vizuálnímu seskupení elementů, které spolu souvisí. Druhým je Anotace, ta umožňuje přidání komentáře textového charakteru pro vysvětlení určité části diagramu.[8],[34] V některých textech je možné najít ještě třetí typ - Asociace.[8]



Obrázek 21: Objekt Group s objektem annotation artifact (zdroj vlastní)

3.5 Data a Data storage reference

V rámci BPMN se prvek Data storage reference objevuje v roli datového úložiště, které je v podnikovém procesu určeno k ukládání dat a informací. Primární použití je pro jednoznačné definování toho, jaká data jsou s procesem spojena a ukládána, jakým způsobem jsou data přístupná a také jak jsou modifikována v průběhu. Data storage reference může být úzce spojen s jinými prvky v diagramu, aby bylo zřetelné, jak v procesu jsou data využívána.[8]

Data object je celý název prvku, který v BPMN 1.2 označoval právě Asociation Artifact. V BPMN 2.0 je prvek osamostatněn a je důležitý pro reprezentaci dat, které jsou těsně svázána s podnikovým procesem. Často jsou spojena také s jedním prvkem nebo celou skupinou prvků. Příkladem může být znění emailové zprávy,[8]

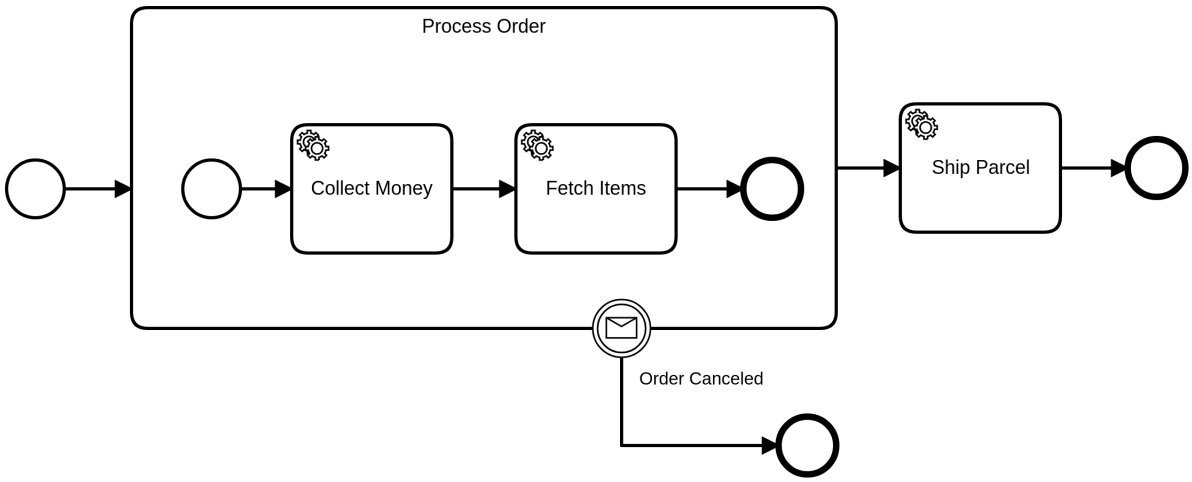
Obrázek 22: Data a data storage reference (zdroj vlastní)

3.6 Sub-process

Sub-process, jak nám napovídá samotný názvu, je procesem, který je zahrnutý do jiného procesu. Samotný blok je reprezentací pod-procesu, který je možno v některých případech brát jako samostatnou část, jindy je přímo závislý na vnějším procesu. Objekt je možné otevřít a zobrazit jednotlivé události, úkoly, brány a jiné objekty BPMN.[8] Vnitřní procesy se také často používají s Boundary událostmi, které mohou provádění jednotlivých bloků v diagramu přerušit nezávisle na tom, která část se právě provádí. Toto přidává k diagramu flexibilitu a adaptabilitu k reálném pracovnímu postupu.[11],[29]

3.6.1 Embedded sub-process

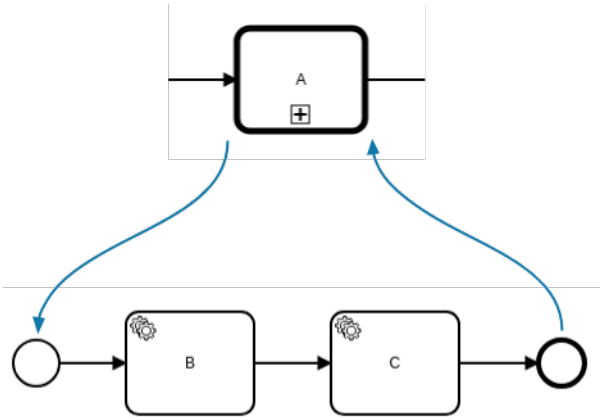
Embedded sub-process je klíčový pro to, když je nutné v diagramu jednotlivé prvky skládat do skupin. Omezující podmínkou je to, že sub-process smí mít pouze jednu počáteční událost. Ve chvíli, kdy je spuštěn tento subprocess, dojde ke spuštění právě této jedné počáteční události a dojde k vykonávání jednotlivých operací (objektů) ve vnitřním procesu v souladu s diagramem. Tento vnitřní proces má také svůj konec. Jakmile je průběh u konce, pokračuje vnější proces ve vykonávání dalších částí.[8],[30]



Obrázek 23: Embedded sub-process příklad (zdroj [30])

3.6.2 Call-Activities

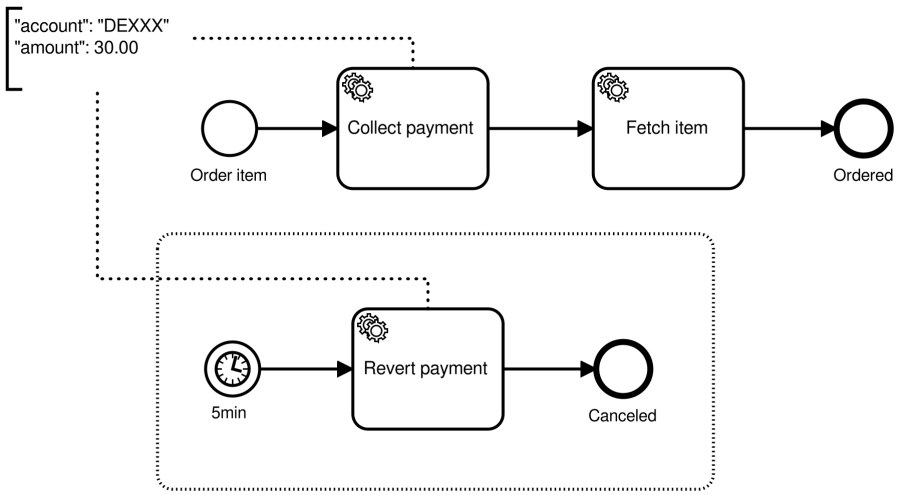
Call-Activities, též označován jako Reusable Sub-process[8] je velmi podobný Embedded verzi. Hlavní rozdíl je ten, že se vnitřní proces nachází mimo hlavní diagram. Ve chvíli, kdy proces dorazí do události na zavolání tohoto externího procesu, přejde vykonávání právě do něj. Uvnitř je Start event, který se spustí a jakmile je proces dokončen, vrací se průběh zpátky do původního procesu, kde se pokračuje ve vykonávání pokračování. Označení "Reusable sub-process" má právě z toho důvodu, že je možné tento sub-process volat několikrát a nemusí být vložen v diagramu duplicitně.[8],[31]



Obrázek 24: Call-Activities příklad (zdroj [31])

3.6.3 Event sub-process

Timer, Message a Error události jsou typy událostí, které mohou spustit tento typ pod-procesu. Jedná se o spojení pod-procesu a boundary event. Ve chvíli, kdy dojde ke spuštění události, která je na subprocess umístěna, pak dojde ke spuštění vnitřního procesu.[11] Na rozdíl od ostatních dvou typů pod-procesů, tento typ není součástí toku procesu. Nevyužívá tak tohoto toku pro svoje spuštění, ale pouze reaguje na danou událost. Při spuštění události pro daný pod-proces, pak dojde ke spuštění Start události, která může být pouze jedna.[8]



Obrázek 25: Event sub-process příklad (zdroj [32])

3.7 Pool / Participants

Pool je grafickou reprezentací jednotlivých účastníku podnikového procesu a spolupráce mezi takovými účastníky. Jedná se o prvek, který v diagramu není povinný a není nutné ho využívat. Prvek není omezen nijak na velikosti a proto může zabírat velkou část diagramu. Pool se chová jako kontejner, který obaluje část procesu určenou pro vykonávání daného aktéra v procesu. Pro spolupracující aktéry je možné volně přecházet mezi sebou, ale pro tok mezi jednotlivými aktéry v procesu je nutné využívat Message. Sekundárně může sloužit prvek ke skrytí přesné posloupnosti v toku. Toto použití se označuje jako "black box". V takovém případě není viditelné v procesu, co se přesně děje, ale jsou zde pouze naznačení pro Message.[8]

Participant, někdy také označován jako aktér, v diagramu naznačuje, že provádění celého procesu v diagramu není pouze úkolem pro jednu entitu. Entitou můžeme rozumět jak lidskou práci, tak systém nebo celou firmu. Pokud je to pro diagram důležité, použijeme právě několik plaveckých drah, kde každá jedna představuje jednoho aktéra v procesu.[8]

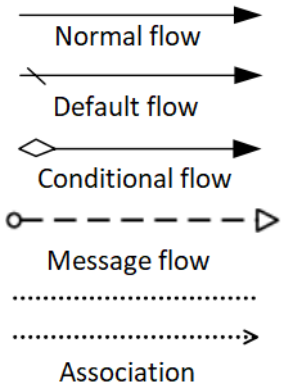
Obrázek 26: Pool se dvěma aktéry (zdroj vlastní)

3.8 Spojování objektů

Pro modelování je nutné spojovat jednotlivé prvky ve dvou případech s použitím toku. Ten může být v několika verzích na základě chování a charakteristiky toku. Klasickým je Normal Flow (někdy označovaný jako Sequence flow), který se používá v případě sekvenční návaznosti objektů na sebe. Značí se jednoduchou čárou s šipkou na straně bloku, kam

tok plyne. Dalším je Message flow, který označuje tok zpráv v diagramu. V diagramu je zobrazen přerušovanou čarou a šipkou ve směru toku. Poslední základní možností spojení objektů je Association, který v diagramu reprezentuje spojení funkčních objektů s objektem typu Artifact. Nejedná se tedy o tok.[8]

Oficiálně existují 2 další druhy toku. Prvním je default, který se značí krátkou čarou s krátkým škrtnutím na začátku. Využívá se ve spojení s určitými typy objektů Gateway, kde je nutné modelovat výchozí cestu. Normální tok může mít aktivní atribut pro podmínku, která musí být splněna, aby tok pokračoval, tomuto toku poté říkáme Conditional flow a v diagramu je zobrazený s kosočtvercem z objektu, ze kterého vychází.[8]



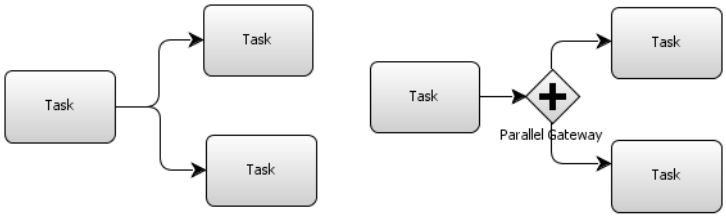
Obrázek 27: Čáry pro spojení objektů (zdroj [8])

4 NÁVRHOVÉ VZORY BPMN

Stejně jako u jiných diagramů existují i v BPMN určitá ustálená řešení pro řešení modelování častých podnikových procesů. V této části jsou některá z nich popsána a vysvětlena. Je užitečné tyto vzory znát a při modelování myslet na to, že pro pochopení diagramu ostatními účastníky projektu je nutno jejich dodržování.[8],[45]

4.1 Paralelní rozdělení

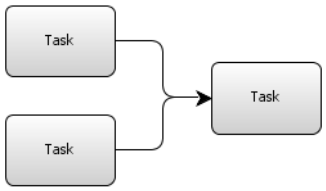
Paralelní rozdělení je prvním ukázaným návrhovým vzorem a jak vyplývá z názvu, jedná se o vzor, který popisuje rozdělení aktivity paralelně. K tomu se dá využít Parallel Gateway, která rozděluje tok pro paralelní zpracování. Druhou možností je využití objektu Group nebo Sub-process. A posledním validním řešením je využití více toků z dané aktivity. [45]



Obrázek 28: Paralelní rozdělenení (zdroj vlastní)

4.2 Simple merge

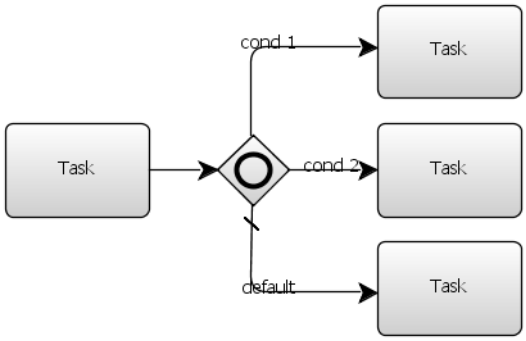
Ke spojení dvou nezávislých vláken používáme návrhový vzor Simple merge. V tomto případě není důležité, které vlákno dosáhne na pokračování jako první nebo pokud je zde výlučné rozdělení. Pokračování v procesu začne hned, jak skončí aktivita jednoho z vláken. Grafickou reprezentací takového vzoru je pouze sekvenční tok ze dvou paralelních aktivit do jedné následující.[45]



Obrázek 29: Simple merge (zdroj vlastní)

4.3 Multiple choice

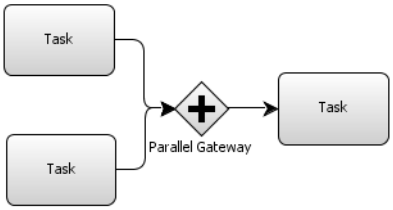
Pokud je nutné modelovat situaci, kdy se rozdělí tok do několika na základě splnění podmínek, pak se využije vzor Multiple choice. Zároveň se pro tento vzor přidává výchozí cesta pro případ, že nebude splněna žádná podmínka. Podmínka pro splnění se váže k danému toku a je určena za běhu. Důležité je, že v případě použití tohoto vzoru nejsou toky navzájem výlučné (změnit slovo asi..), tok se tedy může paralelně rozvětvit, ale nemusí.[45]



Obrázek 30: Multiple choice (zdroj vlastní)

4.4 Synchronizace

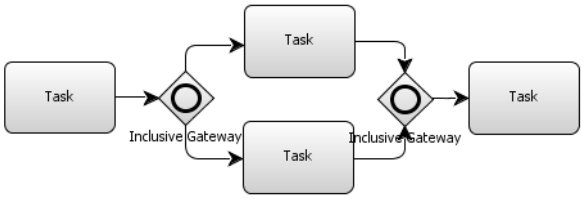
Návrhový vzor synchronizace je vzor, který zaručuje synchronizaci po paralelním rozdělení. K tomu slouží dvě možné řešení. Prvním je využití Parallel gateway pro sloučení prvků. Také někdy označovaná jako Joining Gateway. Druhou možností je využití objektu Group nebo Sub-process, kde není End event, který by ukončil daný pod-proces. K pokračování ve vnějším procesu pak dochází až ve chvíli, kdy všechny vnitřní aktivity jsou dokončeny.[45]



Obrázek 31: Synchronizace (zdroj vlastní)

4.5 Synchronization merge

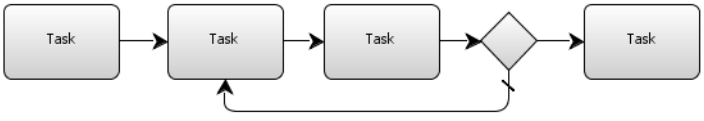
Cílem tohoto vzoru je synchronizovat všechna vlákna, která byla dříve v procesu rozdělena. K tomuto řešení se používá dvojice Inclusive decision gateway, kde je nutné znát počet vláken, která pokračují. Pro funkčnost je nutné, aby v druhém objektu mohlo být zjištěno, zda stejný počet vláken dorazil. Nastaveno je to zde díky vstupní podmínce v druhém objektu.[45]



Obrázek 32: Synchronization merge (zdroj vlastní)

4.6 Arbitrary Cycles

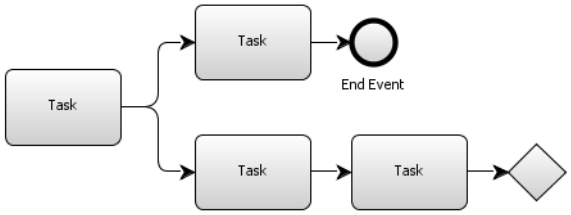
Doplňkové cykly jsou jednoduchým návrhovým vzorem, který slouží pro zopakování částí pracovního toku v podnikovém procesu. Vyplatí se především v případě, že je nutné vizualizovat složitou strukturu aktivit s opakováním. Využívá se k tomu neoznačená gateway, která má výchozí cestu právě zpět na některou z předchozích aktivit.[45]



Obrázek 33: Arbitrary Cycles (zdroj vlastní)

4.7 Implicit termination

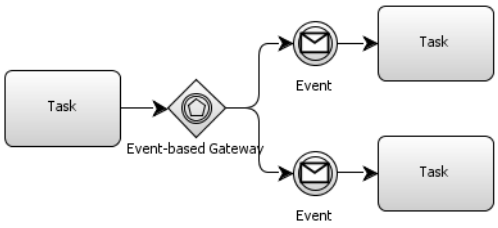
BPMN umožňuje ukončit jedno určité vlákno bez toho, aby byla ukončena všechna ostatní. Pro tyto případy je zde implicitní ukončení. Vlákno po paralelizaci a vykonání určitých aktivit není nutno spojit ani synchronizovat a tak má vlastní End event. Zde je velmi lehké udělat chybu a vložit Terminate End event, který ukončí právě i ostatní vlákna.[45]



Obrázek 34: Implicit termination (zdroj vlastní)

4.8 Deferred choice

Jedná o způsob odložení rozhodnutí, která z výlučních cest bude ta správná. Většinou se k tomu využívá Event-based gateway, která čeká na vykonání událostí následujících. Která událost nastane jako první, tou se vydá tok procesu.[45]



Obrázek 35: Deferred choice (zdroj vlastní)

5 CHYBY PŘI MODELOVÁNÍ

Při tvorbě diagramů je možné narazit na celou sérii nejčastějších chyb z obecného pohledu na podnikový proces, ale v této práci jsou popsány také nejčastější chyby uživatelů pro modelování diagramů.

5.1 BPM chyby

BPMN jsou považovány za důležité z hlediska vnímání celého procesního řízení. Poukazuje se na chyby při tvoření základních prvků pro budoucí workflow.[8] Mezi tyto chyby patří zejména:

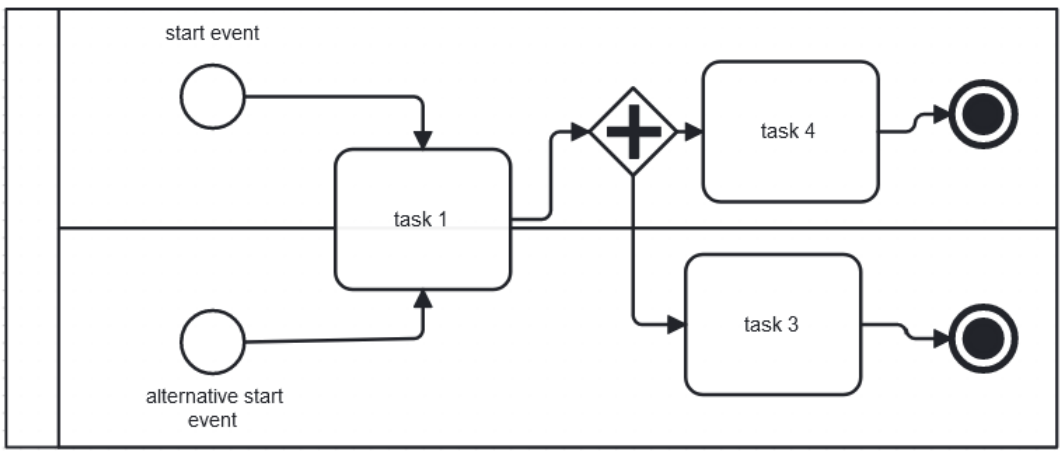
- **Modelování organizační struktury** namísto modelování hierarchie procesů. Kdy je například chybně modelováno každé oddělení firmy.
- **Nedostatečná modelace** podpůrných procesů a sub-procesů, které jsou pro nadřazený proces důležité. Je nutné modelovat všechny procesy, pokud jsou důležité pro jejich nadřazený proces.
- **Příliš složitý/detailní procesní model**, který popisuje v modelu každou prostou činnost v organizaci, která nemusí být stěžejní pro popis daného procesu.
- **Model ukazuje tok dat** namísto toku aktivit. Diagram by měl vždy ukazovat primárně tok aktivit, který popisuje podnikový proces.
- **Nedostatečný popis aktivit a úkolů** v celém diagramu. Tato chyba může vést hlavně k nedorozumění a tím i nepochopení, jak má být proces proveden.

5.2 BPMN chyby

Pro správné pochopení je také nutné nedělat chyby ve skladbě diagramu a řídit se tedy pravidly pro správné modelování. Tyto chyby může pomoci odstranit také jeden z níže popsaných nástrojů, který disponuje validátorem.[44],[43] Mezi nejčastější chyby tohoto typu se řadí:

- **Proces nemá start a end event** je jedna z hlavních chyb, které se uživatelé při modelování dopouští. BPMN tyto dva objekty udává jako nepovinné, nicméně je velmi doporučeno je používat. Pokud uživatel do modelu nepřidá tyto dvě události, nemusí být zřejmé, jak samotný proces začíná a může tak dojít k nedorozumění.

- **Nesprávné pojmenování událostí** může způsobit mnohdy více škody. Pokud uživatel pojmenuje objekty podle jejich role v procesu a nedojde k použití anotací, pak je diagram nepřesný. Ideální je pojmenovávat objekty dle pochopení všech členů.
- **Použití více objektů, než je nutné.** BPMN umožňuje vytváře více, než pouze jednu startovací a cílovou událost. Nicméně pokud se uživatel může odkazovat pouze na jedinou, je zbytečné používat jich více. Doporučuje se také aby se uživatel vyhýbal duplicitám částí procesu a vytvářeli proto pod-procesy.
- **Terminate event vs End event.** Uživatelé běžně používají Terminate event více, než je nutné. Terminate end event je silnějším ukončením procesu a tudíž pokud dojde k paralelizaci procesu, při použití právě tohoto objektu dojde k ukončení všech vláken. A většinou to není ani úmysl pro diagram.
- **Aktivity jsou umístěny tak, že překrývají linie.** V tomto případě se jedná o vážnou chybu v diagramu, ve kterém není zřejmé vykonávání této aktivity.
- **Každá linie obsahuje vlastní startovní událost.** Pro některé uživatele může být logické, že každá linie v procesu má svůj začátek, ale pro diagram to není důležité. Modeluje se celý proces, který obsahuje ideálně jeden start v objektu Pool.
- **Použití Message event pro předání dat mezi aktivitami v sekvenci.** Touto chybou je myšleno, že pokud se v diagramu nachází vedle sebe dvě aktivity, lze využít pro přenos dat Message flow a Data object.



Obrázek 36: Diagram s chybami (zdroj vlastní)

6 NÁSTROJE PRO BPMN DIAGRAM

Pro samotnou tvorbu diagramu se dá využít hned několik nástrojů, Všechny, které jsou v práci popsány, mají velmi podobný základ a každý má svoje dodatečné věci, které jsou pro nástroj specifické. Dalším důležitým faktorem je cena za použití nástroje. Představenými nástroji v této práci jsou:

- Yaoqiang BPMN Editor
- Camunda
- VisualParadigm
- Creately
- BPMN.io

6.1 Yaoqiang BPMN Editor

Prvním nástrojem pro ukázku je Yaoqiang, což je aplikace, která se v poslední době stala vyhledávanou volbou zejména pro tvorbu BPMN diagramů. Yaoqiang vyniká svou snadnou použitelností a intuitivním prostředím, což jej činí ideálním nástrojem pro ty, kdo se zaměřují pouze na vytváření diagramů bez složitých technických požadavků. Yaoqiang získal velký ohlas díky své kompletnosti a absenci nadbytečných funkcí, které by některým uživatelům nepřipadaly potřebné. Navíc je rychlý a pohodlný pro začátečníky díky svému odlehčenému stylu.[42]

6.1.1 Výhody

- Možnost nahrávání originálních OMG BPMN souborů.
- Real-time BPMN syntax validátor pro kontrolu diagramů.
- Zcela zdarma a open-source.
- Není nutnost nic instalovat.
- Automatické aktualizace.
- Zabudovaný verzovací systém.

6.1.2 Nevýhody

- Design aplikace není příliš moderní, což může některé uživatele zaskočit.

- Nástroj působí dojmem patřícím do minulosti, ale stále se vyvíjí a zdokonaluje.
- Nástroj nelze integrovat s externími nástroji a využít pro celé řízení podnikových procesů.

6.2 Camunda

Camunda je open-source nástroj pro řízení práce, který poskytuje uživatelům kontrolu a flexibilitu. Kromě tvorby diagramů umožňuje platforma správu podnikových procesů a má schopnost spravovat celé týmy. Pro vývojáře nabízí další funkce, jako je automatizace procesů nebo simulace toku postupu. Integrace s nástroji třetích stran je také snadná a efektivní cestou.[35],[36] Co se týče ceny nástroje, je zdarma pro běžné použití bez automatizace částí procesů. Ale nabízí i placené a komplexnější řešení s podporou a možnostmi automatizace.[37]

6.2.1 Výhody

- Poskytuje kontrolu a flexibilitu uživatelům.
- Umí spravovat podnikové procesy a celé týmy.
- Nabízí další funkce pro vývojáře, jako je automatizace procesů.
- Jednoduchá a efektivní integrace s nástroji třetích stran.

6.2.2 Nevýhody

- Velká komplexnost při prvním nastavení.
- Složitost úpravy existujících modelů.
- Pro běžné uživatele je považován za příliš technický.
- Některé funkce vyžadují placené řešení.

6.3 VisualParadigm

Druhým nástrojem pro tvorbu BPMN diagramů je VisualParadigm, který mimo jiné nabízí tvorbu dalších typů diagramů a je nejčastěji porovnáván s nástrojem Camunda. Primárně se jedná o nástroj pro tvorbu UML diagramů, ale v nabídce má také ER diagramy nebo právě BPMN diagramy. Umožňuje stejně jako Camunda simulovat pracovní postup

a animovat jednotlivé kroky. Používá se pro menší i komplexní řešení a hlavní výhodou je jednoduchost. Nástroj umožňuje integraci s některými nástroji, ale není jich mnoho.[36] Co se týče ceny nástroje, je zdarma pro běžné použití bez automatizace částí procesů. Ale nabízí i placené a komplexnější řešení s podporou a možnostmi automatizace.[39]

6.3.1 Výhody

- Jednoduchost použití.
- Možnost simulace pracovního postupu.
- Nabídka různých typů diagramů.

6.3.2 Nevýhody

- Náročnost pro běh a využití prostředků.
- Omezené možnosti integrace s dalšími nástroji.
- Nutnost platby za trvalou licenci nebo periodické používání.

6.4 Creately

Nástroj Creately je založený na cloudu a umožňuje pracovat na více typech diagramů v jednom prostředí. Charakteristickou výhodou je možnost kombinace tvorby diagramů a spolupráce v týmu s jasným pracovním postupem. Vše v tomto nástroji je čistě vizuální a není zde zapotřebí žádný kód. Nástroj sám umožňuje několik užitečných funkcí, jako například nekonečné pracovní plátno nebo ukazatel myši v reálném čase. Integrace s dalšími nástroji pro vývoj je také možná. Cenově se nástroj dá používat s omezením zdarma, ale pro plné využití je možnost zakoupit plnou verzi.[35] Cenově se nástroj dá používat s omezením zadarmo. Pro plné využití všech funkcí nástroje je zde možnost koupit pro jednotlivce i pro firmu. V druhé možnosti pro firmu je zároveň v ceně podpora v reálném čase.[40]

6.4.1 Výhody

- Práce na více typech diagramů v jednom prostředí.
- Sdílení obsahu v reálném čase s více uživateli.
- Možnost kombinace tvorby diagramů a spolupráce v týmu.

- Vizuální prostředí bez nutnosti programování.
- Několik užitečných funkcí, jako nekonečné pracovní plátno nebo ukazatel myši v reálném čase.

6.4.2 Nevýhody

- Pro plné využití základních funkcí je potřeba zakoupit plnou verzi.
- Někteří uživatelé mohou pociťovat omezení v omezené verzi zdarma.

6.5 Bpmn.io

Narozdíl od ostatních nástrojů se nejedná o komplexní aplikaci pro tvorbu BPMN diagramů. Bpmn.io je javascript knihovna, kterou lze stáhnout ze stejnojmenné webové stránky. Kromě samotné knihovny jsou k dispozici i ukázky pro rozšíření nebo použití této knihovny. Tato knihovna je vytvořena a spravována týmem, který stojí za vytvořením nástroje Camunda. Je plně zdarma a na webové stránce poskytuje dokumentaci k práci s knihovnou.[41]

6.5.1 Výhody

- Plně zdarma k použití.
- Poskytuje množství ukázek a dokumentaci.
- Umožňuje online vyzkoušení nástroje na webových stránkách.

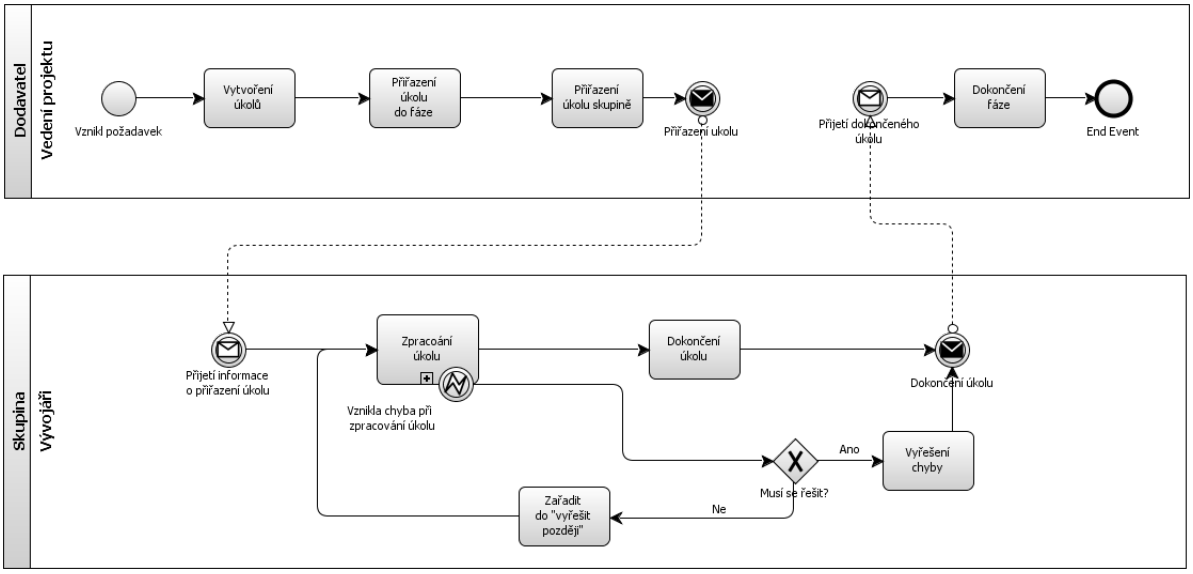
6.5.2 Nevýhody

- Není komplexní aplikací pro tvorbu BPMN diagramů.
- Chybí informace o tvorbě BPMN diagramů na webových stránkách.

7 UKÁZKY BPMN DIAGRAMŮ

V této sekci budou představeny vybrané ukázky použití BPMN diagramu s ohledem na vývoj IT projektů. Prvním ukázkou je přidělení a zpracování úkolu v rámci IT firmy. Druhou bude vytváření skupin a rozdělení dílčích úkolů při vývoji, včetně plné kontroly přebírané práce. Posledním bude ukázka při agilním vývoji, kde zadavatel kontroluje výstupy a průběh vývoje. V této práci jsou ukázky modelů vytvářené v nástroji Yaoqiang BPMN Editor. Autor nástroj využil právě z důvodu, že je tato část práce zaměřena na tvorbu BPMN diagramů a ostatní nástroje mají přebytečné funkce nebo složitou instalaci, popřípadě složité první nastavení.

7.1 První ukázka

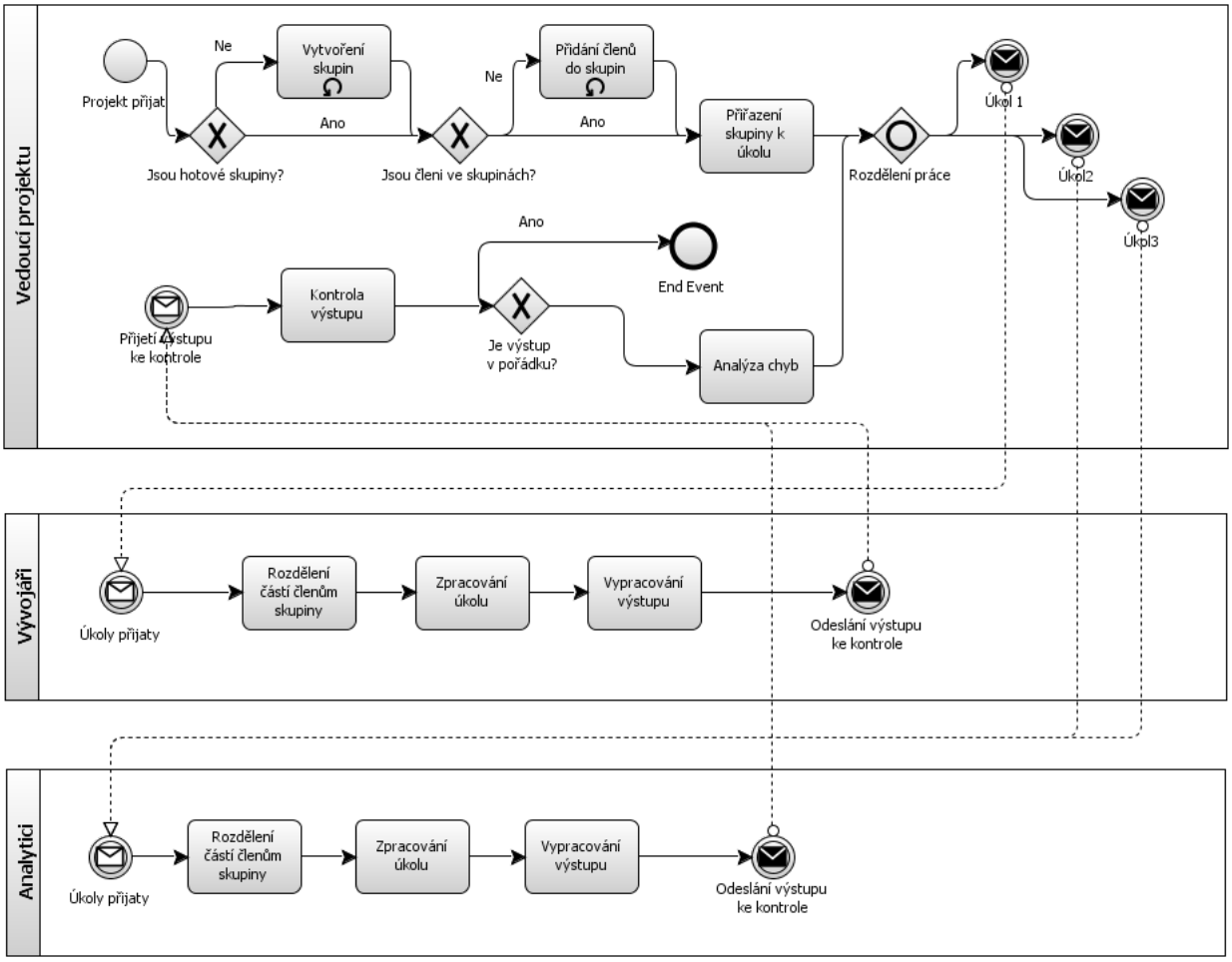


Obrázek 37: BPMN diagram první ukázky (zdroj vlastní)

První ukázka diagramu řeší jednoduché přiřazení a zpracování úkolu s možností chyby. Po vytvoření úkolu vedoucím projektu a přiřazení skupině vývojářů dojde k odeslání zprávy o přiřazení a pracovník ze skupiny vývojářů může zpracovat tento úkol. Na subprocess je přidán Boundary Error event, který umožňuje vyvést výstup pro chybové stavy ven a v případě tohoto diagramu dojde k odložení zpracování chyby na později. Pokud chyba nenastane, dojde k dokončení úkolu a předání zpět vedoucímu projektu, který může

ukončit fázi, ve které úkol je. Pokud se chyba bude řešit, pak je potřeba zpracovat úkol a vyřešit chybu.

7.2 Druhá ukázka

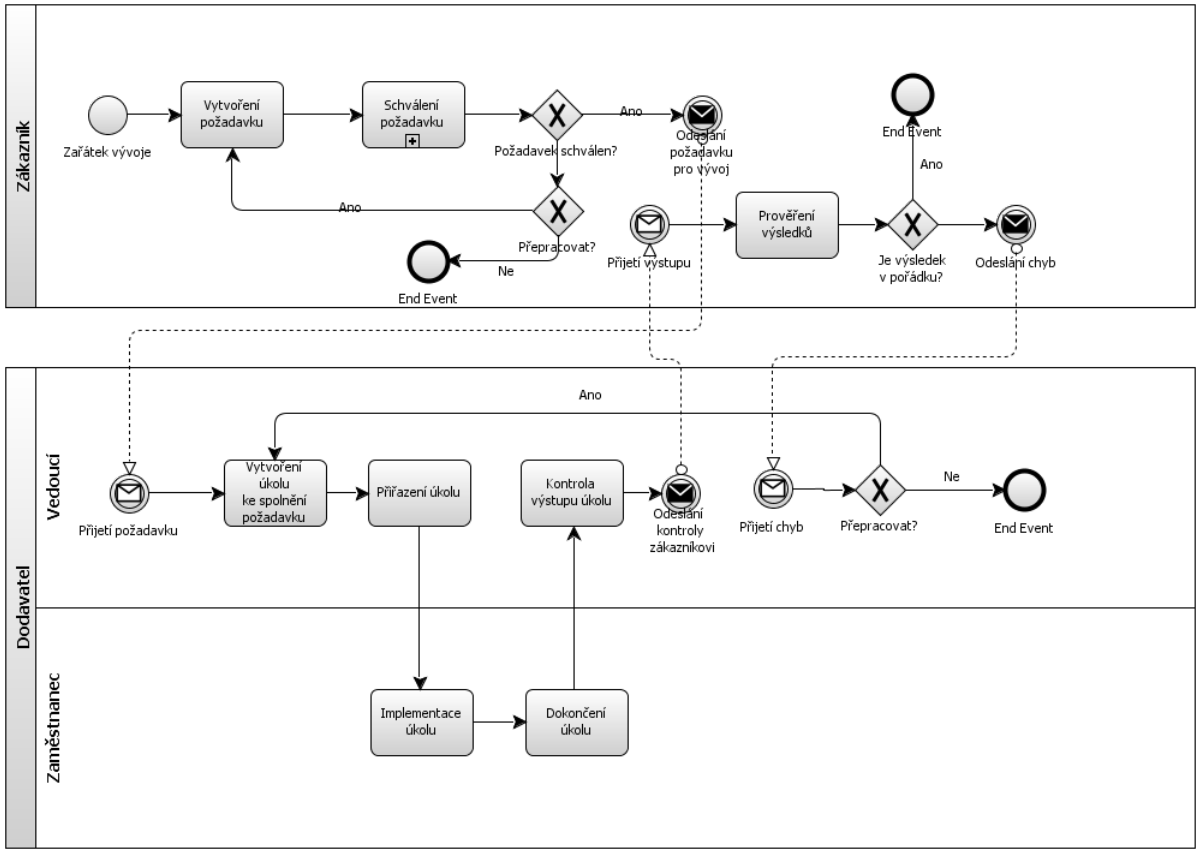


Obrázek 38: BPMN diagram druhé ukázky (zdroj vlastní)

Druhá ukázka je komplikovanější. Ukázka popisuje vytvoření skupin. Tato operace může být cyklická, protože je potřeba vytvořit více skupin, pokud ještě neexistují. Poté dojde k přiřazení členů, který ještě ve skupinách nejsou a to je také cyklický úkol, jelikož se provádí pro každého člena a skupinu zvlášť. Teprve po vytvoření a přidání členů může dojít k přiřazení jednotlivých úkolů, kde každá skupina dostává svůj úkol, který paralelně zpracovává. Po dokončení úkolu skupinou dojde k odeslání výsledku a kontrole na straně vedoucího projektu. Pokud je řešení úkolu přijato, dojde k ukončení. Jelikož je v běhu

ještě paralelní zpracování dalších úkolů a event je klasický End event, nedojde k ukončení celého procesu, ale pouze k zániku tokenu pro daný úkol.

7.3 Třetí ukázka



Obrázek 39: BPMN diagram třetí ukázky (zdroj vlastní)

Třetí a poslední ukázka BPMN diagramu popisuje zpracování úkolu na základě vytvoření požadavku zákazníkem a následné schválení. V tomto konkrétním diagramu dojde k vytvoření požadavku a po interním schválení je odeslán dodavateli softwarového řešení. Poté je nutné vytvořit úkol pokrývající tento požadavek na straně vedoucího projektu, který ho zadá svým pracovníkům. Implementace a dokončení úkolu je na straně zaměstnance, který poté předává opět vedoucímu projektu řízení a ten výstup zkontroluje a teprve poté předává zákazníkovi. Zákazník po kontrole může výsledek úkolu přijmout a tím dojde k dokončení. Druhou možností je nepřijetí vypracovaného úkolu zákazníkem, který přejde do bodu, kdy musí informovat vedoucího projektu, který rozhodne o přepracování.

8 PROJEKTOVÉ ŘÍZENÍ

Projekt v projektovém řízení znamená určitý proces, který je příliš složitý, jedinečný a rizikový na to, aby se vnímal jako postup několika kroků. S rostoucím projektem je schopnost kontrolovat a plánovat jednotlivé kroky složitější.[46] Z toho důvodu se využívá funkcí pro projektové řízení. Mnohdy je pro projektové manažery složité pojmout celý projekt bez vytvoření funkčních informačních a kontrolních systémů. Takový systém by měl pomoci plánovat, monitorovat a kontrolovat velké objemy dat, které při podnikových procesech vznikají. V dřívějších dobách byla správa podnikových procesů záležitostí hierarchie společnosti, v dnešní době organizace využívají projektové manažery, kteří mají za úkol vést projekt po celou dobu.[47] Příkladem takového projektu je v oboru informačních technologií vývoj software.

Projektový manažer je osobou, která se podílí na vedení projektu. U některých projektů je možné vedení pouze jedním manažerem, ale některé velké projekty vyžadují přijetí více těchto osob, které se starají každá o svoji kategorii. Příkladem může být manažer, který se stará o integraci, správu času, kvality a má na starosti sledování rizik. Zatímco druhý manažer má na starosti sledovat rozsah projektu, finanční stránku, správu lidských zdrojů a komunikaci při vývoji. Podobným způsobem je možné projekt rozdělit pro více, než pouze 2 takové manažery.[47]

Každý projekt má svoje postupy, které se mohou mírně odlišovat na základě principů organizace. Ale všechny se zakládají na těchto krocích[46]:

1. Příprava projektu. Tato fáze vývoje vytváří základní kameny pro vývoj. Je důležitá pro přípravu projektového záměru a popis všech důležitých charakteristik projektu jako jsou například důvody, cíle, rozpočet nebo zdroje. Tento projektový záměr je vytvořen a projednáván. S jeho schválením přechází projekt do další fáze.
2. Naplánování projektu je druhou fází, která obsahuje převážně činnosti spojené s organizací pro projekt. Pro možný začátek je nutné mít schválený projektový záměr, ze kterého v tomto kroku vychází činnosti jako rozklad práce (WBS), kde se projekt rozdělí na menší ucelené části, ke kterým lze přiřadit zdroje. V tomto kroku se také provádí časové odhady a rezervy nebo přípravy projektových týmů.

3. Provedení projektu. Ačkoliv se jedná o fázi provedení, není součástí pouze vykonávání plánovaných kroků. Je nutné také reagovat a spravovat změny, které mohou do projektu přijít. V této části také dochází k akceptování jednotlivých výstupů.
4. Ukončení projektu jako poslední fáze nezahrnuje pouze odevzdání nebo nasazení projektu. Projektový manažer vyhodnotí projekt jako celek. Tato akce pomáhá k zapracování zkušeností a poznatků z právě dokončených projektů do těch budoucích.

Nároky na řízení projektů jsou jedním z důvodu založení a vedení několika organizací, které vydávají doporučení a sjednocují principy pro vedení projektu. Manažerům, kteří tyto principy praktikují je poté udělena certifikace například asociací IPMA (International Project Management Association).[46]

8.1 Časté chyby

Stejně jako při vývoji procesu nebo diagramů BPMN i zde jsou časté chyby, které lze vidět napříč projekty při vývoji IT projektu. Některé z nich mohou být[48],[46]:

- Nejednoznačné zadání projektu nebo obecně nedostatečné plánování projektu. Jak bylo řečeno při popisu první fáze projektu, plánování je základním kamenem, který v případě nedostatků může způsobit selhání celého projektu.
- Nedochází k uplatnění základních postupů projektového řízení. Tím vzniká chaos v projektu, nedodržují se termíny a ve výsledku nikdo nemá přehled, co se v projektu má udělat.
- Jednotlivé skupiny jsou zahlceny příliš mnoha procesy, došlo ke špatnému rozdělení zdrojů. V případě vytvoření velkého projektu a nedostatečnému odhadu na lidské zdroje dochází k zahlcení.
- Členové projektu nemají aktuální přehled o stavu projektu. Z toho důvodu není jasné, ve které části projekt je a pokud tyto informace nejsou centralizované, může dojít k tomu, že jednotliví členové projektu pracují paralelně na stejné věci.
- Problémy při vzniku projektu jsou ignorovány nebo se na ně zapomene.
- Špatná nebo žádná komunikace v jakékoliv fázi projektu mezi členy projektu nebo i zákazníkem. Tím může docházet zbytečně k více úpravám projektu, než je nutné.
- Nedostatečné ukončení projektu bez vyhodnocení a poučení do budoucna z vývoje. Pokud po dokončení projektu dojde pouze k předání/nasazení a přechází se k dalšímu projektu.

8.1.1 Doporučení pro aplikaci

Na základě ukázek diagramů a chyb, které je možné při řízení projektů v IT udělat autor vyhodnotil na ukázkách tyto chyby, pro které v aplikaci navrhne a popíše funkce, které se vyvarují těmto chybám. Chyby:

- V první ukázce je možné najít chybu s možným zapracováním chyby do dalších úkolů bez nutnosti vyřešit danou chybu. Později může s touto chybou dojít k odevzdání úkolu a dokončení fáze, ačkoliv úkol nemusí být kompletní. Pro aplikaci by bylo vhodné navrhnout funkci, která bude chyby zaznamenávat a poté nedovolí dokončit danou fázi do chvíle, kdy nedojde k dokončení všech dílčích úkolů.
- V druhé ukázce je rozdělení úkolů, ale lze zde najít také chybu, která byla výše popsána. Chybou v této ukázce je nedostatečná možnost komunikace mezi členy jednotlivých týmů při snaze návaznosti jednotlivých úkolů. Proto by pro aplikaci bylo vhodné implementovat možnost komunikace projektu také na úrovni fáze, ke které mají přístup všichni členové projektu. Druhým doporučením na základě druhé ukázky je implementovat způsob, jak zaručit, aby úkoly bylo možné navazovat na sebe.
- Ze třetí ukázky lze najít chybu v možnosti kontroly zákazníka při vývoji. V této ukázce má zákazník přístup pouze k výsledkům, které může schválit nebo odmítnout. Doporučení pro aplikaci je tedy udělit také zákazníkovi možnost nahlédnout do vývoje s omezenými pravomocemi, aby měl přehled o průběhu projektu nebo pouze jednotlivého úkolu a mohl například včas vznést námitky.

9 ANALÝZA KONKURENČNÍCH MOBILNÍCH APLIKACÍ

Pro vytvoření nástroje s tématem řízení projektů je nutné analyzovat již existující řešení. Využít výhod takových nástrojů a naopak se pokusit odstranit nevýhody. Pro analýzu mobilní aplikace na řízení projektů autor práce vybral dvě. Jednou je Freelo, které je česká aplikace od vývojářů z Pardubic a v České Republice se jedná o velice oblíbený nástroj na řízení projektů. Jako druhý nástroj autor zvolil Trello, se kterým má osobní zkušenosti a je naopak používán převážně v zahraničí pro vedení projektů.

9.1 Freelo

Freelo je projektový nástroj, který umožňuje týmům snadno organizovat svou práci, spolupracovat a sledovat pokrok projektů. Jedná se o českou aplikaci vytvořenou také bývalými studenty Univerzity Pardubice. Kromě samotné správy týmů a rozdělení úkolů nabízí také možnosti propojení pro fakturaci s pomocí určení hodinových sazeb.[50],[51],[53]

9.1.1 Výhody

Užívání aplikace Freelo má mnoho výhod, některé z nich jsou [52]:

- Jednoduché použití s intuitivním uživatelským rozhraním je jednou z důležitých aspektů, které musí splňovat aplikace pro řízení managementu při vývoji a nástroj tímto disponuje.
- Shodné prostředí s webovou aplikací pro usnadnění přechodu uživatele mezi těmito prostředími.
- Kalendářový přehled pro sledování termínů v projektu.
- Možnosti vytváření šablon pro projekty. Tato funkce nabízí tak možnost vytvářet projekty a části projektu na základě již vytvořených.
- Kompletní české prostředí i s českou podporou na chat i telefon.

9.1.2 Nevýhody

Jako komplexní systém má také svoje nevýhody, příkladem mohou být [52]:

- Cena nástroje pro organizaci se pohybuje ve vyšších částkách a pro malé týmy nebo jednotlivce tak není vhodným nástrojem.
- Omezené možnosti integrace s jinými nástroji je jednou z připomínek uživatelů. Freelo tento nedostatek řeší veřejným Application programming interface (API), které umožňuje uživatelům si vytvořit svoje možnosti napojení.
- Nepřehledný hlavní dashboard (úvodní přehled), který na první pohled neukazuje přesný přehled napříč projekty.
- Emailové upozornění na každý krok v aplikaci, ze kterého se v akčním prostředí stane rychle spam.

9.2 Trello

Trello představuje nástroj pro řízení projektů skrz Kanban karty. Kanban karty představují digitální převedení klasických tabulí s papírky, které popisují jednotlivé funkce a části se všemi informacemi. Nástroj je jednodušší než Freelo a je primárně užíván pro rozdělení práce, úkolů a přehled.[54],[55]

9.2.1 Výhody

Trello má mnoho výhod pro používání. Některé z nich jsou [55]:

- Prostředí nástroje je také intuitivní. Díky tomu je velmi rychlé adaptovat nové uživatele.
- Flexibilní organizace úkolů skrz seznam karet umožňuje uživateli vytvářet kategorie/seznamy, k nim přidávat karty, přesouvat je a přizpůsobit si panel dle potřeb.
- Základní verze zdarma je dostačující pro malé týmy nebo jednotlivce.
- Nástroj nabízí mobilní aplikaci velmi podobná té webové pro přehlednost uživatele při přechodu mezi nimi.
- Nástroj nabízí šablony pro různé typy projektů.
- Nástroj je možné přepnout do češtiny.

9.2.2 Nevýhody

Stejně jako u předchozího nástroje má ale i tento své nevýhody [55]:

- Nástroj neumožňuje integraci s externími nástroji a nástroj nedisponuje veřejným API pro integraci.
- Omezené možnosti hierarchie a strukturování karet. Na komplexní projekty je toto jednou z největších nevýhod, jelikož uživatel může vytvářet pouze kategorie a do nich zařazovat karty.
- Aplikace nemá k dispozici podporu v češtině a z recenzí uživatelů je při problému většina odkázána na manuál tohoto nástroje při řešení potíží.

9.3 Monday

Monday není na trhu krátce, ale v poslední době se stal populárním nástrojem v oboru řízení projektů. Ačkoli má podporu pro vedení projektů, je nástroj spíše verzí zobrazení úkolů a vedení informací o nich v rámci týmů. To znamená, že aplikace funguje na principu zaznamenávání úkolů a přehled těchto úkolů v rámci projektu.[56],[57]

9.3.1 Výhody

Organizace mají nástroj Monday ve velké oblibě i z těchto důvodů [57]:

- Monday má moderní design a je částečně intuitivní.
- Nástroj nabízí flexibilitu v přizpůsobení si potřeb pro správu projektu.
- Je možné zobrazovat stejná data v několika odlišných způsobech. Příkladem může být tabulkové zobrazení, ganntův diagram nebo kanban karty.
- Monday nabízí integraci s mnoha dalšími aplikacemi.
- K dispozici jsou také šablony pro tvorbu různých projektů, včetně vývoje software.

9.3.2 Nevýhody

Nástroj ovšem disponuje i určitými nevýhodami, zde je příklad některých z nich [57]:

- Nástroj nemá českou lokalizaci, nabízí lokalizaci pro 13 jiných jazyků.
- Některé pokročilejší funkce mohou být dostupné pouze v dražších plánech.
- Pro jednotlivce nebo menší týmy je nástroj pro plnohodnotné používání drahý.
- Nástroj nenabízí možnosti analytiky projektu.
- Nástroj neumožňuje určovat měnu v projektu.

9.4 YouTrack

Nástroj YouTrack patří do skupiny nástrojů JetBrains a je implementován v jazyce Java. Nástroj je nabízen ve verzi hostované, tedy cloudové řešení, ale nástroj je možné také stáhnout a provozovat na vlastním počítači. To umožní jednotlivcům mít všechny důležité věci u sebe a spravovat si pouze svůj prostor v aplikaci. Je to tedy jediná aplikace z aktuálního listu, která umožňuje nástroj používat i bez registrace uživatele.[59],[58]

9.4.1 Výhody

Pro nástroj se uživatelé rozhodnout i díky těmto výhodám [60],[58]:

- Plnohodnotný nástroj pro 10 členů zdarma s téměř všemi důležitými funkcemi pro vedení projektu.
- Detailní dokumentace pro každý krok, se kterým může mít uživatel problém.
- Jako každý nástroj JetBrains i YouTrack má vlastní fórum, ve kterém se uživatelé mohou poradit nebo zeptat na případné dotazy, pokud nechtějí jít na podporu.
- Je možné nástroj mít ve verzi cloudové, ale také provozovat na vlastním serveru.
- Nástroj umožňuje migrace z jiných nástrojů na vedení projektů a integraci s mnoha dalšími aplikacemi.
- YouTrack disponuje chytrým Arificial Inteligence (AI) asistentem, který usnadňuje práci, umožňuje shrnutí nebo generování odpovědí na komentáře. Tato funkce je dostupná ve všech verzích, včetně té zdarma.

9.4.2 Nevýhody

Uživatelé mohou zvolit jiné řešení například z některého z těchto důvodů [60],[58]:

- Ačkoliv se jedná o firmu se sídlem v České Republice, čeština není v nabídce pro lokalizaci v základu nástroje. Je možné dodat externě vlastní lokalizaci, která je volně dostupná na internetu pro český jazyk.
- Pro menší projekty může být nástroj příliš robustní a nabízet příliš mnoho nevyužitelných funkcí.
- Nástroj nabízí pouze univerzální šablony na základě typu vedení projektu.

10 MULTIPLATFORMNÍ MOBILNÍ APLIKACE

Multiplatformní mobilní aplikace (MMA) pro řízení projektů představuje nástroj pro organizace a jednotlivce, kteří se zabývají plánováním, sledováním a dokončením IT projektů různého rozsahu a složitosti. Tato kapitola poskytuje celkový přehled o aplikaci s ukázkami a také návrhy do budoucna pro optimalizaci a zlepšení aplikace.

Primárním cílem aplikace je usnadnit a zefektivnit proces vedení projektů prostřednictvím intuitivního uživatelského rozhraní. MMA poskytuje uživatelům prostředky k efektivní organizaci, plánování a sledování pokroku projektů, což v konečném důsledku přispívá k dosažení stanovených cílů a zvýšení produktivity.

10.1 Analýza požadavků

Před začátkem vypracování samotného projektu je nutné vytvořit seznam požadavků na základě analýzy. Analýzu požadavků lze vytvářet několika způsoby. Prvním z nich může být vyslechnutí zákazníka při dodání software pro organizaci. Při tomto způsobu se většinou určí, co je důležité a skrz doplňující otázky je možné více určit přesně potřeby, které je nutné aplikací naplnit. Druhým způsobem je komplikovanější popsání požadavků na základě podnikových potřeb. Od první metody se liší především tím, že trvá delší dobu a také na ní pracují analytici na základě dokumentů k projektu. Příkladem takových dokumentů může být projektový plán nebo sepsané požadavky zákazníka. Dalším bodem, který může při tvorbě požadavků pomoci je nějaký případ užití, který přinese vhled do problému, který má MMA řešit.[49] Pro výstup z takové analýzy může být seznam funkčních a nefunkčních požadavků. Pro tuto aplikaci byl vytvořen seznam následující.

10.1.1 Funkční požadavky

Mezi funkční požadavky se řadí právě požadavky na chování produktu. Seznam funkčních požadavků pro aplikaci:

1. Veškeré informace pro uživatele budou podřízeny autorizaci.
2. Aplikace umožní uživateli se registrovat.

3. K aplikaci bude připojen emailový klient pro zasílání emailů.
4. Uživatel si v aplikaci může resetovat své heslo.
5. Uživatel si v aplikaci může zažádat o obnovu hesla na základě kódu v emailu.
6. Uživatel bude moci ke svému profilu nahrát profilový obrázek.
7. V aplikaci bude režim rolí, které bude možné dynamicky spravovat administrátorem.
8. Práva pro přístup k projektu budou dána na základě oprávnění u role.
9. Při používání aplikace se bude ověřovat uživatel pomocí JWT tokenu.
10. Uživatel bude moci založit organizaci a projekt pro přístup do všech funkcí aplikace.
11. Uživatel bude moci přijmout pozvánku do organizace a projektu pro omezený přístup do všech funkcí aplikace.
12. Po přijetí pozvánky bude uživateli přiřazena nejnižší možná role.
13. Na hlavní obrazovce bude mít uživatel přehled napříč svými projekty.
14. Aplikace bude umožňovat lokalizaci v češtině a angličtině.
15. Aplikace bude mít možnost přepínat světlý a tmavý režim.
16. Uživatel může v projektu vytvářet nové fáze projektu.
17. Uživatel může vytvořit úkol do dané fáze s informacemi o úkolu, včetně pokrytí požadavků.
18. Vedoucí projektu může vytvářet týmy.
19. Vedoucí projektu může do týmů přidávat členy týmu.
20. Pro nutné entity bude možnost přidávat komentáře.
21. Vedoucí týmu může vytvořit výstup pro danou fázi.
22. Uživatel s rolí zákazník a vedoucí projektu mohou vytvářet požadavky pro aplikaci.
23. Uživatelé s rolemi zákazník a vedoucí projektu mohou spravovat rozpočet projektu.
24. Fáze musí být splněny všechny dílčí úkoly, aby mohla být uzavřena.
25. Úkoly bude možno exportovat do aplikací kalendáře.
26. Aplikace bude logovat změny v projektu.
27. Aplikace bude zobrazovat počet nových věcí napříč projekty uživatele.
28. V aplikaci bude možné zobrazit seznam změn napříč projekty uživatele.
29. Uživatel může zrušit svůj účet v aplikaci.
30. Aplikace bude nabízet možnost zobrazení poslední aktivity v projektu.
31. Uživatel bude moci v aplikaci změnit projekt nebo organizaci dynamicky.
32. Uživatel bude moci nastavit časovou zónu pro aplikaci.

33. Uživatel bude mít nastavení aplikace spojené se svým účtem.

10.1.2 Nefunkční požadavky

Nefunkční požadavky popisují nezbytné vlastnosti aplikace, které je potřeba znát vzhledem k prostředí nebo kontextu. Seznam nefunkčních požadavků pro aplikaci:

1. Aplikace bude vytvořena s použitím aplikačního rámce .NET MAUI pro multiplatformní přístup.
2. K aplikaci bude vytvořena MSSQL databáze.
3. Databáze bude hostována na cloudovém řešení Azure.
4. Aplikace bude uživatelsky přívětivá.
5. Aplikace bude využívat prvky moderního designu.
6. Aplikace bude přístupná ve formátu .apk pro nainstalování.

10.2 Technologická architektura

Pro tvorbu MMA je důležitým krokem volba technologické architektury s podstatným odůvodněním. Pro tuto aplikaci byla vybrána hlavní platforma .NET s programovacím jazykem C# a pro mobilní aplikaci .NET MAUI. Vybrané technologie budou popsány v následujících podkapitolách včetně detailního rozboru databázového modelu.

10.2.1 .NET MAUI

.NET MAUI je multiplatformní nástroj umožňující vytváření mobilních a desktopových aplikací pomocí C# a XAML. S použitím tohoto nástroje lze vyvíjet aplikace pro Android, iOS, macOS a Windows, přitom využívat sdílený základní kód.[61]

.NET MAUI je open-source a představuje vývoj Xamarin.Forms, který je rozšířený z mobilní aplikace i pro desktop. Od předchozí verze disponuje ovládacími prvky uživatelského rozhraní přepracovanými od základu pro lepší výkon a rozšiřitelnost. Použití tohoto nástroje umožní vývojářům vytvořit pouze jediný projekt pro celé multi-platformní řešení. Pokud je to ale nutné, je možné dodat zdrojový kód i prostředky pouze vybraným platformám. Jedním z hlavních cílů .NET MAUI je umožnit implementovat co nejvíce logiky aplikace a rozložení uživatelského rozhraní v jediném zdrojovém kódu.Pro tvorbu aplikací v nativním i multi-platformním režimu umožňuje například [61]:

- Modul pro rozložení stránky - tvorba layout pro společné prvky mezi stránkami.
- Bohatý výběr navigačních a funkčních prvků.
- Možnosti pro obsluhu rutin pro lepší způsob, jak prezentovat UI prvky.
- API rozhraní pro platformy k nativním funkcím. Příkladem může být využití GPS, akcelerometru, stav sítě nebo i baterie. Toto API umožní také uvnitř aplikace zjistit základní informace o zařízení, na kterém je spuštěna. Součástí je také sdílení schránky telefonu pro kopírování nebo práce s úložištěm souborů v telefonu.
- Jeden projekt a jeden kód pro více platforem.
- Opětovné načítání za provozu, které při vývoji usnadní práci, jelikož není nutné celý projekt rekompilovat a zdrojový kód je tedy možné upravit a načíst do aplikace.

10.2.2 Blazor MAUI Hybrid App

V této práci byla použita nadstavba pro .NET MAUI, tedy Blazor MAUI. Podpora Blazor pro .NET MAUI je implementována a obsahuje tak prvek BlazorWebView. Ten umožňuje vykreslovat Razor komponenty přímo do Web View. To umožňuje ještě více udělat tuto technologii multiplatformní a přitom využívat nativní možnosti jednotlivých platforem. Mobilní, desktopová i webová aplikace tak mohou sdílet mezi sebou jednotlivé komponenty. Do vývoje mobilních aplikací tak přichází možnost psát kód v html, css a používat javascriptové funkce.[62]

Razor

Razor je označení pro komponenty, které využívá Blazor pro uživatelské rozhraní a přidává do ní logiku a dynamicky spravuje chování. Komponenty je možné použít znovu nebo vnořit jednu do druhé. Je také možné komponenty jednoho projektu sdílet s jiným projektem. Pro implementaci komponent se využívá jazyk HTML a také C# v souborech s příponou .razor. Ve výchozím nastavení je pro komponenty základní třídou ComponentBase, který je rozšiřuje o důležitou sadu předem definovaných událostí životního cyklu komponenty.[66]

Syntaxe těchto komponent je založena na dvou prvcích, která se označují v komponentě klíčovým slovem po znaku '@'. První jsou direktivy. Ty mohou měnit způsob pro funkčnost nebo parsování kódu. Konvence udává také pořadí těchto direktiv a jejich úrovně. První řádek může obsahovat direktivu @page a udává tím, že se jedná o směrovatelnou

komponentu, kterou lze tedy načíst v prohlížeči na konkrétní URL adrese. V druhé úrovni se udávají direktivy @using, které jsou označením pro jmenné prostory, které jsou přidány pro funkčnost kódu. Třetí úroveň poskytuje prostor pro @inject direktivy, které do dané komponenty přidávají servicies skrz dependency injection. Od .NET 8 je také nová direktiva @rendermode, díky které lze určit v jakém režimu bude probíhat vykreslování dané komponenty. Tato nová direktiva se řadí do druhé úrovně a tak posouvá direktivy @using a @inject na další. Běžně se mezi direktivy neumísťují prázdné řádky, ale není to pravidlem.[66]

Attributy jsou druhým prvkem, který komponenty využívají. Dělají to samé, co direktivy, ale pouze pro vybrané elementy. Attributy nejsou sdílené s direktivami a mohou například označovat referenci na vnitřní logiku komponent nebo se skrz attributy implementuje bind proměnných a vlastností. Rozdílem je již zmíněná direktiva @rendermode z .NET 8. Pokud ji umístíme mezi direktivy, je její funkce pro celou komponentu a její vnořené prvky. Pokud ale využijeme @rendermode jako attributu, můžeme tím určit tuto vlastnost vykreslování pouze pro danou komponentu.[66]

Na konci komponent je také důležitá část, která označuje kód C# a kde je možné psát funkce, přidávat do komponent proměnné nebo upravovat jednotlivé body životního cyklu komponent. Tento blok se označuje klíčovým slovem @code{} a tato část je určena pouze pro C# kód. Pro tuto část jsou určeny další možnosti s anotacemi. Příkladem může být [Parameter] který v kombinaci s direktivou @page přejímá z url adresy daný parametr. Někteří vývojáři tento kód upřednostňují mimo samotnout razor komponentu a tak lze vytvořit částečná třída s příponou .razor.cs, kde je možné tento kód vložit.[66]

Mezi direktivami a kódovým blokem je v komponentě místo pro elementy, které jsou v jazyce HTML. Pokud by chtěl vývojář vložit kód do této části, je možné využít k tomu značku '@', která označuje kódový blok. V mnoha případech je do projektu ale dodána grafická a funkční knihovna, která vývojářům pomáhá vyvíjet aplikace, které mají předem určený moderní design a již hotové funkční prvky.[66]

10.2.3 MudBlazor

MudBlazor je grafická a funkční knihovna pro Razor komponenty. Využívá vlastní css vlastnosti a pro funkčnost také vlastní javascript funkce. Jako knihovna nabízí několik důležitých prvků jako jsou tlačítka, vstupní pole, navigace nebo prvky pro zobrazení dat.

Nabízí také snadný a elegantní přístup k dialogům. Jedním z důležitých bodů je také uvedení celé MMA do vybraných témat, ve kterých si vývojář může předem definovat barevné palety i font, který v aplikaci bude používat v určitých místech.[63],[64]

MudBlazor nabízí také vyzkoušení MudBlazor komponent bez toho, aby si uživatel musel instalovat tuto knihovnu do projektu. Společně s tím nabízí přehlednou a důkladnou dokumentaci s několika ukázkami včetně kódu. Jednou z mnoha dalších výhod je také to, že prvky podporují responzivitu a usnadní vývojářům práci s ošetřováním přetečení u jednotlivých prvků. Pro všechny tyto aspekty je MudBlazor bezkonkurenční volbou ve výběru UI knihovny pro Blazor.[64]

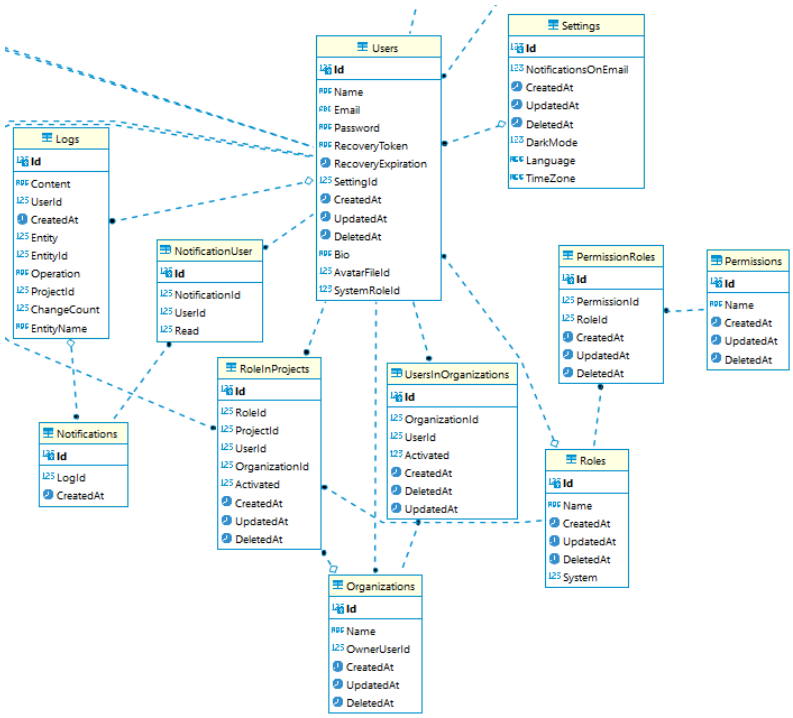
10.2.4 ASP.NET Web API

Pro plnohodnotné fungování mobilní aplikace je velmi užitečné přidat do technologické architektury také Web API. Webové aplikační rozhraní umožňuje zprostředkovat komunikaci mezi mobilní aplikací a databázovou aplikací. Samotné Web API představuje back-end aplikaci celého projektu. Přesněji je pro aplikaci implementována REST API, která využívá HHTP protokolu pro zpřístupnění všech CRUD operací na základě metod na jednotlivých endpointech. Endpoint představuje url adresu, kde je očekávána a poskytována předem definovaná komunikace. Uvnitř API pracuje s Entity Framework core (EF core), který se stará o práci s datovou vrstvou. Výhodou využití REST API a HTTP protokolu je také možnost umístění informací do hlavičky HTTP požadavků a tak je možné posílat na Web API s každým takovým požadavkem také JWT token pro ověření autorizace i autentizace. HTTP protokol má také svoje tělo, které umožňuje v různých formátech posílat data.[65]

10.2.5 Databáze

Jako databáze byla zvolena MSSQL. Microsoft SQL server je relační databáze vyvinuta a spravována firmou Microsoft. Jedná se o jednu z nejpopulárnějších databází na současném trhu. Databáze MSSQL byla vybrána s hostováním na cloudu Azure a obě technologie jsou vlastněny společností Microsoft.

Samotný návrh databáze byl připraven ještě před implementací a za dobu implementace se změnil pouze minimálně. Celý diagram je možné nalézt v příloze této práce. Vzhledem k jeho velikosti je tedy v této části práce rozdělen a jsou popsány jeho jednotlivé části.



Obrázek 40: Uživatelská část ER diagramu databáze (zdroj vlastní)

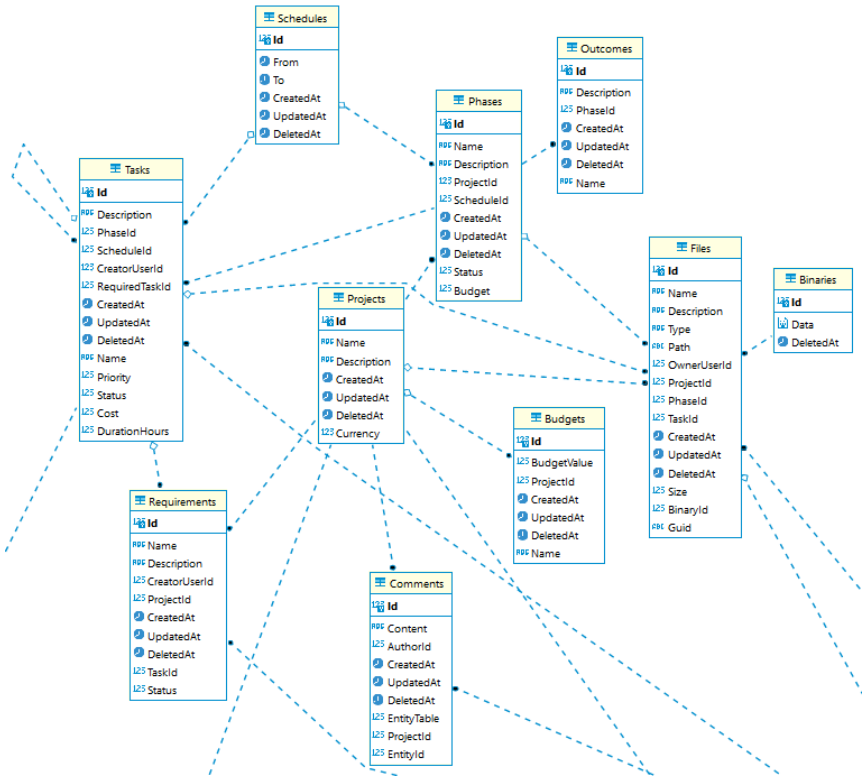
V první ukázce diagramu je část, která se věnuje převážně uživatelské části. K tabulce uživatelů se pojí tabulka s nastavením, jelikož je nutné ukládat uživatelské nastavení po přihlášení uživatele. Role v rámci systému jsou ve dvou verzích. První role je systémová, tedy například administrátor. Jelikož je možné do projektu přistupovat za více organizací a pokaždé s jinou rolí, je tedy nutné spojovací tabulka pro uživatele, projekt a roli. Tato tabulka RoleInProjects tedy spojuje všechny tyto záznamy a využívá druhou verzi rolí. Spojení s projektem a vlastnictví projektu je dáno rolí. Role Owner je statická v aplikaci a vytváří se v kódu, pokud neexistuje. K projektu samotnému tato role přináší všechna práva.

Pro role je dynamicky spravovatelný seznam oprávnění, které daná role má. Proto je zde tabulka PermissionRoles, která spojuje možnosti pro M:N vazbu mezi Roles a Permissions. Dynamicky spravovat role je možné pouze ve webové aplikaci s administrátorským účtem vzhledem k přehlednosti.

Následná část je důležitá pro možnost členství v organizaci. Jelikož může být uživatel členem více organizací, je zde opět spojovací tabulka. Pro obě výše zmíněné spojovací tabulky je nutný atribut pro aktivaci záznamu. To je převážně z důvodu posílání pozvá-

nek a jejich přijímání. Uživatel může být pozván do projektu nebo organizace, ale není právoplatným členem, dokud pozvánku nepřijme.

V této části diagramu zbývá složka pro logování. Jelikož je logování dynamické a nemá spojení na jednotlivé tabulky, je tabulka logů vázána pouze na uživatele, který daný log vytvořil. Na základě typu logu se vytváří notifikace pro každého uživatele. Tato operace je reprezentována spojovací tabulkou NotificationUser, která se vytváří pro každého uživatele v projektu, kde se vytvořil log. Pro každého zvlášť z důvodu, že je nutné ukládat záznam, zda danou novinku uživatel přečetl či nikoliv.



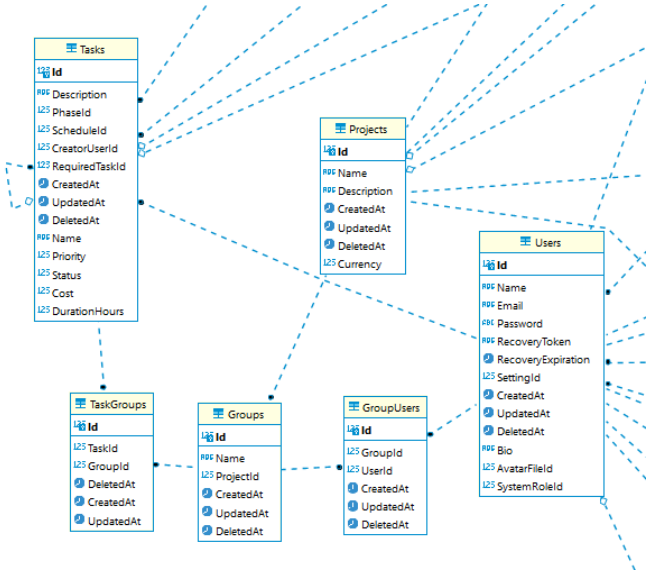
Obrázek 41: Projektová část ER diagram databáze (zdroj vlastní)

V projektové části diagramu je ve středu celého diagramu právě tabulka Projects. Směrem nahoru v diagramu je možné nalézt důležitou část projektů a to jsou jednotlivé fáze, kterých projekt může mít několik. Z pohledu projektu má fáze samotná nějaký časový plán a stejně tak časový plán má úkol. Pro projekt samotný je časový plán přebírán právě z časových plánů fází. Tabulka Files navazuje na projekt, ale primární účel je pro jednotlivé tabulky fází nebo úkolů, ke kterým je soubor připojen. Samotný soubor je rozdělen na část Files, který je tabulkou metadat souboru a soubor je poté v tabulce Binaries.

Pro fázi je seznam úkolů, které jsou vázané na skupiny (mimo obr.) a také úkol má rekurzivní vazbu, jelikož je nutné umožnit vytvoření úkolu, který je závislý na jiném a je započetí možné až po dokončení jiného. Pro úkol je potřeba vytvořit možnost vazeb s požadavky, které zákazník může mít na výslednou aplikaci. Jelikož zákazníka nemusí vždy zajímat, do které fáze je daný požadavek určen, je požadavek převážně vázán na celý projekt.

Další složkou diagramu je možnost přidávat komentáře k jednotlivým záznamům a vybraným entitám. Logování i komentáře jsou udělány dynamickým výběrem entit na základě entit v aplikaci.

Poslední část tohoto diagramu je část finanční. Na tu byl kladen velký důraz vzhledem k požadavkům a také doporučení z Information Technology Infrastructure Library (ITIL) na základě práce pana Bc. Ondřeje Matějky. Tabulka Budgets je vázána na projekt a reprezentuje finanční pohyby spojené s projektem. Příkladem může být vložení základního kapitálu a v rozpočtu je především přehled. Fáze a úkol má atributy Budget a Cost. Pro fázi je poté v aplikaci kontrola překročení rozpočtu.



Obrázek 42: Skupinová část ER diagram databáze (zdroj vlastní)

Třetí a zároveň nejmenší ukázka diagramu se týká týmové práce a přehledu úkolů pro skupinu. Tabulka Groups představuje skupinu, která může mít několik členů a několik úkolů na starosti. Z doporučení pro aplikaci autor převzal spojení skupiny s projektem a nutnost určení skupiny pro daný projekt, ne vzhledem k organizaci. Skupina v případě

aplikace není pro přidělení práv k projektu, ale zodpovědnosti za jednotlivé úkoly, ke kterým je přidána. Práva k projektu proto zůstávají stále na roli uživatele.

10.3 Implementace aplikace

V této části práce autor popisuje tvorbu a zpracování projektu pro výstupní aplikaci. Je zde struktura projektu, ale také jednoduchý průvodce aplikací nebo detailněji popsané funkce na základě výběru tří ukázkových diagramů.

10.3.1 Struktura projektu

Projekt pro aplikaci je tvořen z několika částí. Projekt tvoří samotná mobilní aplikace, ale k vývoji je přidána také aplikace API a mezi nimi je SharedLib knihovna, která obsahuje důležité prvky pro snadnější komunikaci mezi mobilní a API aplikací. K projektu je dále připojena knihovna pro sdílené komponenty Razor.

První na popis je API aplikace, ta představuje vrstvu aplikace, která zajišťuje komunikaci s databází s použitím entity framework core (EF core) technologie. Samotná API aplikace obsahuje řadu kontrolerů, ve kterých je ukryta logika práce s datovou vrstvou na každém jednom endpointu. Je zde také zajištěno například logování nebo odesílání emailů. Pro zabezpečení je zde také middleware služba, která při přijetí requestu kontroluje, zda pro daný endpoint má request dostatečné povolení. To se určí na základě JWT tokenu, který nese seznam oprávnění daného uživatele. Na každém endpointu je pak nastaveno, které oprávnění je zapotřebí pro přístup. Tato aplikace navíc uchovává migrace, které využívá EF core k vytvoření a úprav databáze.

Druhou aplikací je samotná MMA. Jelikož .NET MAUI využívá technologie WebAssembly je zde různé nastavení pro aplikaci. Také je zde složka Resources, která uchovává obrázky, fonty, splashscreen (obrazovka při načítání aplikace) a další věci, které jsou pro aplikaci důležité po zkompilování, aby došlo ke správné reprezentaci v mobilním zařízení. Jelikož je aplikace multiplatformní, je zde také nastavení pro každou platformu, na které může být aplikace nainstalována. Poslední velkou částí této aplikace jsou samotné stránky v komponentách Razor, které může uživatel zobrazit ve své aplikaci. Zde jsou stránky rozděleny na základě entit, se kterými pracují. Kromě takových se v projektu vyskytují další dvě složky - Auth a General. Auth složka uchovává všechny stránky, které jsou potřeba pro

práci s uživatelským účtem. To znamená vše od registrace přes přihlášení, editaci profilu, obnovu hesla až po odhlášení. Složka General obsahuje stránku pro seznam oznámení.

SharedLib je první knihovní aplikace, která se nachází v projektu. Obsahuje různé části aplikace, které je možné sdílet napříč jednotlivými aplikacemi. Například uchovává modely entit, se kterými pracují všechny ostatní aplikace, ale také validátory pro formuláře, různé pomocné komponenty nebo rozšíření. Důležitým prvkem jsou data transfer objects (DTOs) složka, která uchovává seznam request a response objektů pro komunikaci mobilní aplikace s API aplikací. V neposlední řadě se v této knihovně nachází API clienti. Každý kontroller pro API aplikaci má svého API clienta, kterého využívá mobilní aplikace k posílání requestů. Nad každým clientem je také vytvořen interface.

Jako poslední je v projektu ComponentSharedLib knihovna, která je druhou a poslední knihovnou projektu. Tato knihovna je rozdělena na základě typu komponenty. Od toho se odlišuje pouze AuthCoponents složka, která uchovává specificky formuláře pro autentizaci a autorizaci. Každý formulář, tabulka nebo timeline je v této společné knihovně, aby byla znovupoužitelná kdekoliv jinde v projektu. Další důležitou komponentou je také AuthGate. AuthGate je vlastní komponenta, která zajišťuje v aplikaci zobrazení pouze těch částí, ke kterým má uživatel mít přístup na základě oprávnění v JWT tokenu.

10.3.2 Autorizace

První úvodní stránkou pro zobrazení je přihlašovací stránka, kde se vyskytuje formulář pro zadání emailu a hesla pro přístup do aplikace. Kromě tohoto formuláře je zde menu pro přepnutí na registraci, zapomenuté heslo nebo zadání tokenu pro reset hesla. Registrační formulář kromě emailu a hesla obsahuje pole pro jméno, pod kterým uživatel chce v aplikaci vystupovat. Zapomenuté heslo je pouze zadání emailové adresy účtu a posledním formulářem je právě reset hesla po obdržení tokenu, který přijde na email. Po přihlášení je uživateli zobrazen další formulář, který je druhou vstupní branou do aplikace. Jedná se o výběr projektu, pod kterým uživatel chce do aplikace přistupovat. Pokud uživatel nemá žádné projekty a není součástí žádných organizací, může založit vlastní nebo přijmout pozvánku (ukázka v kapitole Uživatelský profil). Po vybrání organizace je obnoven seznam projektů, ve kterých je daný uživatel přihlášen za danou organizaci. Vybrání těchto dvou atributů je důležité pro přidělení oprávnění do projektu.

10.3.3 Přehled

Jakmile je uživatel plně přihlášen, zobrazí se mu přehled jeho účtu. Je zobrazen seznam účastí a rolí v daných projektech. Dále seznam úkolů, které jsou zadané se základními informacemi. Tyto úkoly jsou zobrazeny na základě účasti ve skupinách, které jsou k úkolu přiděleny, V poslední řadě je na této úvodní stránce také seznam organizací, ve kterých je uživatel členem. Tato část se týká přehledu napříč projekty.

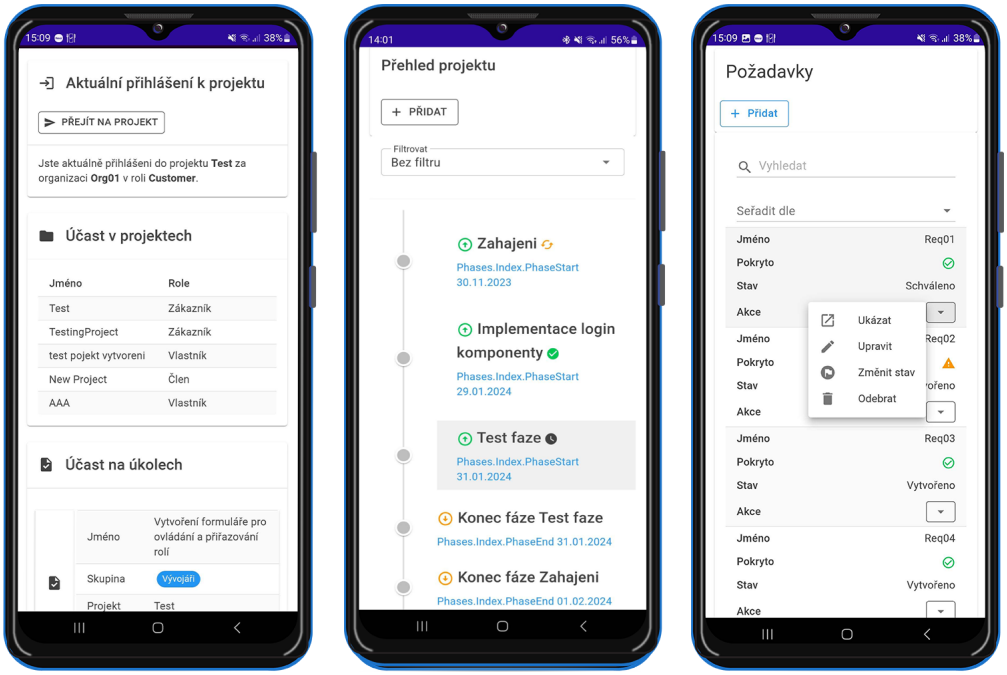
V horní části stránky je tlačítko pro přehled projektu. Na základě oprávnění se tak zobrazí seznam částí projektu a přehled celého projektu. Na základě oprávnění jsou také odděleny možnosti CRUD operací s jednotlivými záznamy, které zde uživatel nalezne.

V každé další stránce je odlišným způsobem zobrazen seznam dané entity v projektu. Jednoduchý popis každé stránky projektu:

- **Přehled** projektu zobrazuje všechny užitečné informace, které jsou s projektem spojeny. Na přehledu se také nachází tabulka poslední aktivity v projektu, kde je posledních pět záznamů změn.
- **Fáze** má přehled v časové ose, jelikož se jedná o důležitou vizualizaci, kdy jednotlivé fáze začínají, kdy končí a jaké je jejich pořadí. Po rozkliknutí jednotlivých fází přichází přehled fáze s jeho informacemi, úkoly, komentáři, soubory a výstupy. Na základě oprávnění také možnost editace tohoto záznamu.
- **Požadavky, výstupy a úkoly** jsou další stránky, které obsahují tabulky s informacemi a možností vytvoření, editace nebo zobrazení bližších informací k daným záznamům. U úkolů je možnost stažení .ics souboru, který je možné nahrát do aplikací kalendáře pro vložení záznamu.
- **Skupiny** uchovávají rozšiřitelný seznam skupin. Pro každou skupinu lze vyvolat akci, která může být přidání uživatele, upravit název skupiny nebo odstranění skupiny. Každá skupina může ale také být rozšířena a zobrazen seznam členů dané skupiny, zde je možnost člena ze skupiny odstranit. Všechny tyto akce vyžadují oprávnění.
- Stránka **Členové** obsahuje tabulky dvou úrovní členství. První je plnohodnotné členství, se kterým může uživatel do projektu přistupovat. V tomto přehledu je seznam členů s možnostmi pro smazání nebo změnu role. V případě změny role vlastníka dochází k předání vlastnictví projektu jinému členovi a bývalý vlastník se degraduje na nejnižší úroveň. V druhém okně je seznam pozvánek do projektu, které

lze smazat. Na stránce nechybí tlačítko pro přidání člena, kde po zadání emailové adresy existujícího uživatele musí být vybrána organizace, za kterou do projektu přistupuje (popř. externista).

- **Rozpočty** jsou předposlední možností menu a na této stránce je přehled rozpočtu daného projektu. Je možné rozpočet zvyšovat a snižovat jednotlivými položkami, ale také položky rozpočtu mazat. Rozpočet je vždy určen na základě měny projektu.
- **Soubory** jsou poslední v menu a zobrazují kompletní seznam souborů nahraných do projektu. Je zde možnost zobrazit detaily souboru a také stáhnout daný soubor.



Obrázek 43: Ukázky MMA (zdroj vlastní)

10.3.4 Uživatelský profil

Můj profil v menu odkazuje na stránku s přehledem uživatelského účtu, ke kterému je uživatel aktuálně přihlášen. Jednou z možností této stránky je nahrání profilového obrázku. V případě, že uživatel žádný nemá, je ve výchozím nastavení obrázek univerzální na základě prvního písmene jména v aplikaci. Ve spodní části jsou další možnosti. Nabídka pro editaci profilu, včetně smazání uživatele z aplikace, nastavení upozornění nebo reset hesla. Jsou zde také další dvě možnosti.

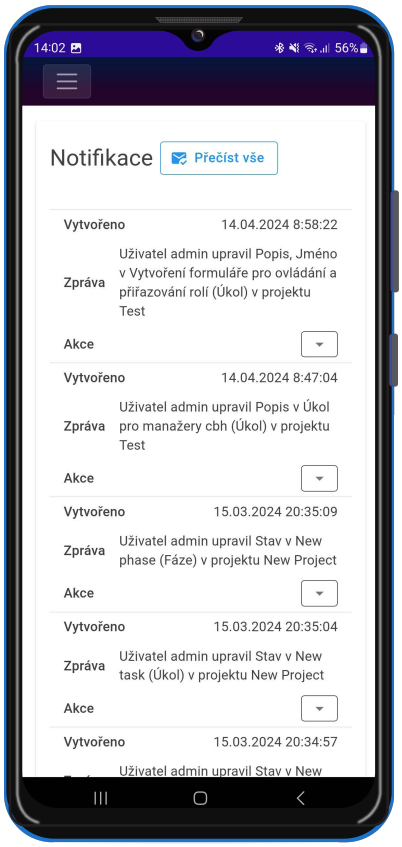
Nabídky k přijetí přesměrují na samostatnou stránku, která uchovává přehled pozvánek do projektů a organizací. Nabídky je možné přijmout, ale také odmítnout. V případě nabídky do projektu se zobrazí uživateli, za jakou organizaci je do projektu pozván.

Správa organizací je poslední stránkou, na kterou se může uživatel z uživatelského účtu dostat. Je zde možnost odeslání pozvánky do organizace, přehled členů a také možnost jejich odstranění z organizace.

10.3.5 Ostatní

Po rozbalení menu v horní části obrazovky je přístup k obecnému přehledu, změně projektu, seznamu novinek, nastavení a odhlášení z aplikace. V práci již byl popsán obecný přehled, změna projektu, uživatelský profil. V poslední části zbývají notifikace, nastavení a odhlášení.

Posledními stránkami v aplikaci je seznam novinek, který je také přístupný z menu. V menu se u položky Novinky zobrazuje také budge s číslem, které označuje počet nepřečtených novinek. V přehledu je zobrazeno datum, kdy se daná událost stala a popsána krátká zpráva, ve které jsou důležité informace k události. U každé položky novinek / notifikací je možné potvrdit přečtení nebo je možné hromadně stisknout tlačítko pro přečtení všech novinek.



Obrázek 44: Obrazovka s novinkami (zdroj vlastní)

Tlačítko Odhlásit provede odhlášení uživatele z aplikace a aplikaci uvede na úvodní obrazovku. V aplikaci zůstanou stále uživatelské nastavení vzhledu, které se pojí k poslednímu přihlášenému uživateli.

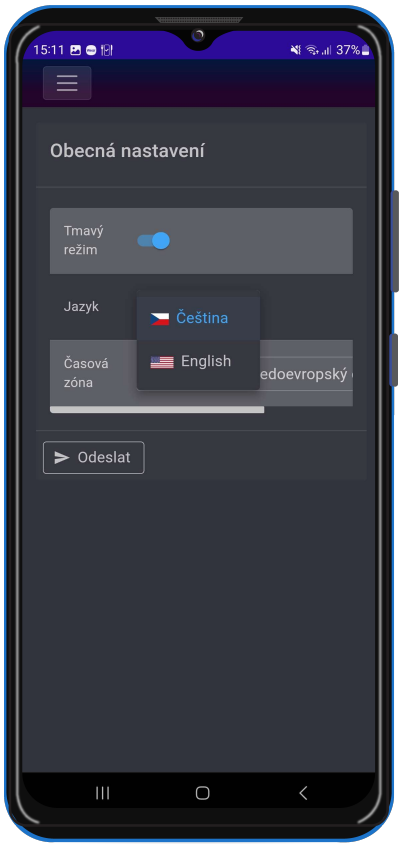
10.3.6 Nastavení

Jedním z požadavků na aplikaci bylo spojení uživatele s nastavením aplikace. Nastavení pro aplikaci je možné spustit už z úvodní stránky a toto nastavení poté s uživatelem zůstane při používání aplikace. Pro uložení a možnosti přenášení nastavení skrze jiná zařízení je ale nutné toto nastavení uložit a spárovat ho tak s uživatelem. Dokud se tedy nastavení neuloží, je pouze v aplikaci.

Nastavení aplikace vychází již z požadavků. S využitím Blazor je možné využít její globalizační a lokalizační možnosti pro uvedení aplikace do více jazyků. IStringLocalizer v rozhraní API v Microsoft.Extensions.Localization je jednoduchým způsobem, jak aplikaci vyvíjet včetně lokalizačního .resx souboru, který pro vypisování textů do aplikace využívá principu klíč/hodnota. .NET MAUI umožňuje konfigurovat CurrentCulture

a CurrentUICulture uživatelem nebo také na základě informací o zařízení. Pro výměnu textů je nutné, aby se aplikace přehrála a tím si z kořenové komponenty mohla zjistit novou jazykovou verzi. Tato operace je velmi rychlá a tak uživatel ani nezaznamená změnu na mobilní aplikaci.[62]

Druhým důležitým nastavením byla možnost přepínání světlého a tmavého režimu. Jak již bylo popsáno v části o knihovně MudBlazor, toto nastavení je velmi jednoduché na základě jejich komponenty ThemeProvider, kterému lze nastavit vlastnost IsDarkTheme. Společně s tím je možné s využitím této komponenty změnit, které barvy se budou vyskytovat v aplikaci.[63]



Obrázek 45: Nastavení aplikace (zdroj vlastní)

10.3.7 Ukázkové funkce

V této kapitole budou představeny a popsány čtyři funkce. První je důležitá pro projekt a další 3 jsou vybrány na základě doporučení z BPMN workflow a jejich implementace v aplikaci.

Práce s rozpočtem

Pro práci s rozpočtem projektu byl postup zvolen na základě doporučení autora, ale také z analýzy ITIL metodiky z práce pana Bc. Matějky Ondřeje.

Při vytváření je nutné zadat měnu, ve které je projekt veden. Toto rozhodnutí je později možné změnit v editaci projektu. Aby měl projekt znám pro všechny zúčastněné měnu, ve které bude veden je velmi důležitý pro mezinárodní spolupráce.

Přidávat, ale také odebírat částky z projektu může zákazník a vedoucí projektu v sekci Rozpočet. Jednoduchým formulářem určí změnu v rozpočtu a pro přehled určí zdroj nebo popis dané změny. Aplikace poté sečte všechny tyto hodnoty a výsledkem vypíše, jaký je celkový rozpočet daného projektu.

Při vytváření nebo při úpravě fází je z celkového rozpočtu možné rozdělit jednotlivé částky do fází. Fázi nelze vytvořit, pokud by měla svoji cenou přesáhnout celkový rozpočet projektu. Toto způsobí, že je možné pro zákazníka řídit rozpočet a pro vedoucího projektu mít pevné hranice, kolik může být na fáze projektu investováno. Při editaci projektu se rozdělí finanční prostředky zpět do celkového rozpočtu a to umožní přidávat i odebírat z dané fáze.

Posledním místem, kde je rozpočet spravován jsou úkoly. Každý jednotlivý úkol má svoji cenu provedení. Tato cena je oddělena od ceny projektu a lze tedy vytvořit úkol s cenou, která přesahuje cenu projektu. Pro schválení úkolu je ale už nutné, aby součet všech úkolů ve fázi nebyl vyšší, než částka pro celou fázi. Pokud by se tak mělo stát, nelze úkol schválit.

Implementace je převážně na straně komponent formulářů, kde není možné formulář odeslat bez splnění podmínek pro vytvoření fáze na základě výše popsaných pravidel. Některé z popsaných funkcí jsou také ale určeny výchozími hodnotami. Příkladem toho může být měna pro projekt, která je ve výchozím stavu CZK. V aplikaci se jedná také o integrační omezení, kde musí být všechny tyto hodnoty při vytváření entit zadány.

Zpracování úkolů

Vytvoření úkolu, přiřazení do fáze a přiřazení úkolu skupinám je v aplikaci implementováno v jednom kroku a to vytvoření úkolu. Po přiřazení se uživateli zobrazí jak notifikace, že byl vytvořen úkol pro skupinu, ve které je členem, tak také se tento úkol objeví v jeho přehledu.

Pokud při zpracování tohoto úkolu dojde k chybě, pak se zapíše chyba do komentářů v dané fázi a není možné do doby dokončení úkolu tedy tento úkol považovat za splněný. Proto tento úkol bude stále veden ve stavu "zpracovává se"a do doby dokončení a opravy chyb by se neměl změnit stav tohoto úkolu.

Při dokončení fáze je nutné změnit stav fáze na "Dokončeno". S tím proběhne i kontrola, že jsou všechny dílčí úkoly fáze dokončeny a není tak možné na žádnou chybu zapomenout.

Tvorba skupin a rozdělení úkolů

V aplikaci pro vedení IT projektů je nutné, aby každý tým měl přehled nad úkoly a bylo možné přiřazovat jednotlivé týmy k daným úkolům. Výstupní aplikace tuto možnost umožňuje.

Jak již bylo popsáno v databázové části, tým je určen pro každý projekt zvlášť a s oprávněním lze vytvářet týmy neomezeně. Pro vytvoření skupiny je nutné zadat pouze název. Po vytvoření skupiny je možné do ní přidávat jednotlivé členy projektu.

Přiřazování úkolů je možné vykonat v sekci úkolů, jelikož nechceme pro každou skupinu zvlášť vybírat seznam úkolů a mnohem snazší je tuto informaci vést přímo u úkolů. Rozdělení úkolů je tedy právě v tomto bodě při vybrání ze seznamu všech skupin, které v projektu figurují.

Svázání úkolů se skupinou má za následek dvě důležité věci. První je, že uživatelé této skupiny na základním přehledu uvidí tento úkol s časovým plánem. Druhou je, že při zobrazení detailů uživatele je možné zobrazit také jeho úkoly.

Implementace této funkce je dána datovým modelem a také formuláři na straně mobilní aplikace.

Agilní vývoj - schvalování výstupů

Agilní vývoj je v případě aplikace reprezentován jednotlivými fázemi. Každá jedna fáze k sobě váže seznam úkolů, komentářů, souborů a výstupů. Při vytváření fáze je možné zadat trvání pro přehled společně s názvem a popisem dané fáze.

Při vývoji jsou převážně upravovány stavy úkolů, přidávány komentáře a výstupy, které se týkají dané fáze. Pro dokončení fáze je nutné mít splněné všechny úkoly. Při dokončení fáze je také možné chtít po zákazníkovi kontrolu výstupů a potvrzení ukončení dané fáze.

Toto rozhodnutí může udělat ale také na zodpovědnost vedoucí projektu například po konzultaci se zákazníkem.

Pro implementaci této funkce jsou využity omezení na formulářích. Aplikace není určena jen pro agilní vývoj, ale při vytváření aplikace byl na tento typ vývoje také brán zřetel a byla jim přizpůsobena právě část fází.

Z doporučení pro aplikaci byl v této funkci byla implementována role zákazníka s možností nahlédnutí do průběhu projektu.

10.4 Testování

Při psaní kódu pro aplikace je velmi užitečné využívat testování. V případě výstupní aplikace bylo provedeno testování unit a integračními testy. Testována byla převážně aplikace API a to s využitím NUnit testovacího projektu v .NET. Provedení testování bylo založeno na hostování aplikace v testovacím prostředí s možností testovat jednotlivé funkce a endpointy v aplikaci.

Testovací třída obsahuje část s anotací [SetUp] pod kterým je stejnojmenná metoda, ve které se nastaví příprava pro provedení testu. Tento blok se provede při spuštění testů a je připraven pro část, kde se inicializují sdílené proměnné pro jednotlivé testy. V případě testování určitých prvků je možné do této části přidat také přihlášení, aby k němu nemuselo docházet s každým spuštěním jednotlivého testu. Další část už je plně vyhrazena testům, které mají anotaci [Test] a při jeho provedení je možné kdekoliv umístit příkaz Assert.That(), který kontroluje splnění podmínky.[68]

Pro testování byl založen nový uživatel v aplikaci a odděleno prostředí od jiných, které byly vytvořeny při vývoji aplikace. Tento uživatel nedisponuje administrátorskými právy a součástí testů je také změna role pro možnosti otestovat přístup k serverové části aplikace. Pro ověření funkčnosti je zde použita stejná databáze se stejným připojením, jaká byla použita ve vývoji a to z důvodu, že je aplikace stále v režimu vývoje a není nutné oddělit testovací prostředí.

Hlavní důvody, proč byly testy pro aplikaci vytvořeny:

- **Odhalení chyb v okrajových případech** je jeden z důvodů, proč je dobré psát testy. Je možné otestovat vložení špatných hodnot do funkce a ihned zjistit, zda se aplikace chová podle očekávání.

- **Rychlá možnost debuggingu** pro aplikaci se ukázala jako velmi výhodná. Místo spouštění aplikace a posílání požadavků v prostředí Swagger bylo jednodušší napsat testy, které obsahují předem zvolené hodnoty, na kterých se při vývoji testuje, zda je aplikace ve funkčním stavu.
- **Změna kódu s jistotou.** Bylo nutné několikrát změnit kód a přizpůsobit se změnám v celé aplikaci není vždy jednoduché. Zpětná kontrola kompatibility s kódem znamená testovat vše od úplného začátku. Pokud má aplikace napsané testy, lze pouze spustit testy a na základě výsledků zhodnotit, kde je potřeba udělat úpravy.
- **Testování před publikováním** v případě automatických testů může zabránit nefunkčním částem kódu dostat se do prostředí, které využívají uživatelé.
- **Podporuje kvalitu aplikace.** Je velmi nepravděpodobné napsat bez otestování kvalitní aplikaci bez žádných chyb. Testování pomáhá odhalit tyto chyby, které nemusí být hned zřejmé a tak zvyšuje podporu pro kvalitu projektu.

10.4.1 Unit testy

Unit testy jsou částí testů, které testují nejmenší funkční jednotku kódu. V případě testování aplikace API je to testování každého endpointu zvlášť pro zaručení jejich funkčnosti. Jedná se o rychlý způsob izolování chyby v případě úprav kódu. Výstupním blokem těchto testů je potvrzení, zda výstup souhlasí s jeho očekáváním. Pro testování jednoho endpointu je napsáno více testů, které pokrývají takzvané testovací případy.[67]

V některých případech může celý projekt být vytvořen na základě testování. Takovému vývoji se říká Test-driven a zakládá se na napsání testů pro funkčnost aplikace a teprve poté na implementaci řešení, aby tyto testy byli úspěšně splněny.[69] Tato metoda při vývoji této aplikace ale použita nebyla.

10.4.2 Integrační testy

Druhým typem testů napsaných pro tento projekt jsou integrační testy. Ty představují typ testování, který se zaměřuje na integraci komponent do skupin a fungování jako celku. Jednotlivé komponenty mohou být testovány pomocí unit testů a fungovat, ale mohou vykazovat chyby v případě, kdy jsou spuštěny v integraci s jinými bloky. Tyto chyby má nalézt právě tento typ testů s konflikty mezi jednotlivými bloky.[67]

```
[Test]
public async Task CreateTask_ReturnOk()
{
    var response = await _client
        .PostAsJsonAsync("/api/Tasks/CreateTask", task);
    var result = await response.Content
        .ReadFromJsonAsync<StatusResponse<SharedLib.Models.Task>>();
    TaskId = result.Data.Id;
    Assert.That(result.Code, Is.EqualTo(HttpStatusCode.OK));
}
```

Zdrojový kód 1: Příklad UNIT testu aplikace (zdroj vlastní)

Pro tento typ testů je možné využít několik přístupů. Prvním z nich je Big Bang, který najednou testuje spojení všech modelů, které mají spojenou nějakou logiku. Druhým typem je inkrementální přístup, ve kterém se využívá spojení pouze 2 modulů a po ověření funkčnosti se přidávají jednotlivé moduly. V případě testování výstupní aplikace této práce byl zvolen první přístup například na fungování CRUD operací s jednotlivými entitami.

```
[Test]
public async Task CrudTaskTest_ReturnNotFound()
{
    await CreateTask_ReturnOk();
    await GetTask_ReturnOk();
    await UpdateTask_ReturnOk();
    await DeleteTask_ReturnOk();
    var response = await _client
        .GetAsync("/api/Tasks/GetTask/" + TaskId);
    var result = await response.Content
        .ReadFromJsonAsync<StatusResponse<SharedLib.Models.Task>>();
    Assert.That(result.Code,Is.EqualTo(HttpStatusCode.NotFound));
}
```

Zdrojový kód 2: Příklad integračního testu aplikace (zdroj vlastní)

10.5 Budoucnost aplikace

V aktuální chvíli je mobilní aplikace vytvořena a dokončena se splněním funkčních požadavků. Aplikace může již v této fázi sloužit pro vedení IT projektů, ale je rozhodně důležité pokračovat v dalším vývoji a aplikaci více přizpůsobit potřebám projektů v oboru IT. Je zde několik možností pro další pokračování aplikace.

1. Přidat možnost přiřazení úkolu ve skupině jednomu danému uživateli. To by umožnilo v každém týmu jejímu vedoucímu více spravovat svůj tým a rozdělit úkoly na úrovni týmů.
2. Integrace s dalšími nástroji. V aktuální době je možné stáhnout soubor .ics, který slouží pro synchronizaci kalendáře, ale pro budoucí úspěch v oboru tohoto typu aplikací je nutné přidat integrace do dalších nástrojů, které se věnují vedení projektu.
3. Případné otevření API a umožnění uživatelům integrovat nástroj do svých aplikací.
4. Notifikace posílané emailem, ale v tomto případě se vyvarovat posílání každé novinky emailem. Pro toto rozšíření by bylo vhodnější zvolit určitý počet novinek, které po shromáždění odešlou uživateli email, aby nezůstal pozadu s tím, co se v jeho projektech změnilo.
5. Možnost přidat do aplikace logování chyb při pádu, která v aktuální verzi chybí. Také přidat možnost systému zpětné vazby, kdy uživatel může navrhnout změny nebo napsat svůj názor přímo v aplikaci.
6. Dynamické spravování rolí. Vytvoření nových rolí, přiřazování práv jednotlivým rolím, které nejsou v aktuální chvíli ve výchozím režimu. Při vedení projektu může být tato možnost zvolit si, kdo má jaké oprávnění pro jednotlivé entity užitečná.
7. Pojmenování oprávnění pro zlepšení přehledu při tvorbě dynamických rolí.
8. V případě uvedení aplikace do provozu v komerčních projektech je nutné vytvoření webové stránky s dokumentací, nabídkou a ukázkami aplikace, aby aplikace mohla být využita v praxi.
9. Implementace šablon pro usnadnění vytváření entit. Jak výchozí šablony, tak i uživatelsky uložitelné.

ZÁVĚR

Cílem této práce bylo seznámit čtenáře s principy vedení a správy projektů v oboru IT a také seznámit s modelováním BPMN diagramů. Prozkoumání podnikových procesů při řízení IT projektů a časté chyby při jejich řízení přispěly k doporučení při tvorbě multiplatformní mobilní aplikace.

Na základě této analýzy bylo cílem praktické části vytvořit funkční prototyp multiplatformní mobilní aplikace, která má uživateli usnadnit vedení IT projektů. Pro vytvoření aplikace autor využil platformu .NET a aplikaci implementoval s využitím .NET MAUI a Blazor technologií. K práci je vytvořeno také vlastní API a databázový systém hostovaný na cloudovém řešení Azure. Všechny použité technologie a principy jsou v práci popsány.

Před implementací byla také provedena analýza již existujících obdobných softwarových řešení, což přispělo při analýze, návrhu i implementaci MMA. Tato řešení byla ve všech případech obecná, a ačkoliv umožňovala nějakým způsobem rozvržení projektu, nebylo to na úrovni, s jakou se rozdělují mnohdy IT projekty. Pro rozvoj MMA mohou být některé body popsané v kapitole Budoucnost aplikace plynoucí z analýzy obdobných řešení podkladem pro další softwarový vývoj. Jedná se o funkcionality, které by umožnily integraci s dalšími nástroji, otevření API nebo implementaci šablon pro usnadnění vytváření entit.

Podstatným krokem při implementaci aplikace byla uživatelská přívětivost, a proto byla vybrána grafická a funkční knihovna MudBlazor, která umožňuje aplikaci sjednotit do jednotného designu s prvky moderního UI/UX prostředí. Druhým důležitým bodem práce bylo na základě analýzy implementovat jednotlivé body a rozvrhnout části projektu na základě praktických zkušeností autora.

Tato práce může pomoci organizacím dodávajícím softwarové řešení, ale také jednotlivcům, kteří chtějí mít přehled o svých projektech. Nástroj slouží převážně pro IT projekty, jelikož je specificky projekt rozdělen do etap (fází), má své úkoly, požadavky a rozpočet.

Při zpracování této práce autor rozšířil své znalosti z oblasti vedení projektu a také získal cenné zkušenosti v oblasti vývoje multiplatformních aplikací. Práce byla pro autora velkým přínosem v oblasti vývoje a projektového řízení.

POUŽITÁ LITERATURA

[1] GARTNER. Gartner Glossary - Business Process Management *Gartner* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://www.gartner.com/en/information-technology/glossary/business-process-management-bpm

[2] IBM. What is business process management. *IBM* [online]. 2024 [cit. 2024-05-09]. Dostupné z: https://www.ibm.com/topics/business-process-management

[3] TEAM KISSFLOW. The Complete Guide for Business Process Management (BPM) *kissflow* [online]. 2024 [cit. 2024-05-09]. Dostupné z: https://kissflow.com/workflow/bpm/business-process-management-overview/

[4] HAPPYFOX. What is BPM? A Guide to Business Process Management *happyfox* [online]. 2022 [cit. 2024-05-09]. Dostupné z: https://www.happyfox.com/what-is-business-process-management/

[5] WANGEN, Gaute Bjørklund, SNEKKENES, Einar. A Comparison between Business Process Management and Information Security Management. *Federated Conference on Computer Science and Information Systems* [online]. 2014, 901-910. [cit. 2024-05-09]. Dostupné z: doi: 10.15439/2014F77

[6] SYSTEM ARCHITECT. Documentation. *UNICOM Systems, Inc* [online]. 2022 [cit. 2024-05-09]. Dostupné z: https://support.unicomsi.com/manuals/systemarchitect/114100/starthelp.html #page/Architecting_and_designing/BusinessProcessAnalysis1006392.html

[7] WILLIS, Craig J. 4 REASONS YOU SHOULD USE PROCESS HIERARCHY WHEN MAPPING PROCESSES *SKORE* [online]. 2023 [cit. 2024-05-09]. Dostupné z: https://www.getskore.com/4-reasons-you-should-use-process-hierarchy-when-mapping-processes/

[8] OMG. BPMN documentation. *Business Process Model and Notation (BPMN): Version 2.0* [online]. 2011 [cit. 2024-05-09]. Dostupné z: https://www.omg.org/spec/BPMN/2.0/PDF

[9] VON ROSING, Mark, Stephen WHITE, Fred CUMMINS a Henk DE MAN. *Business Process Model and Notation—BPMN, in The Complete Business Process Handbook* [online]. Elsevier, 2015 [cit. 2024-05-09]. Dostupné z: doi:10.1016/B978-0-12-799959-3.00021-5

[10] AMERICAN NATIONAL STANDARDS INSTITUTE. *Object Management Group Business Process Model and Notation.* 2013.

[11] SHAPIRO, Robert, Conrad BOCK a kol. *BPMN 2.0 Handbook* [online]. 2nd ed. Florida: Future Strategies, 2012 [cit. 2024-05-09]. ISBN 978-0-9849764-1-6. Dostupné z: https://www.conradbock.org/white-bpmn2-process-bookmark-web.pdf

[12] OMG. BPMN documentation. *Object Management Group* [online]. © 1997 - 2024 [cit. 2024-05-09]. Dostupné z: https://www.bpmn.org/

[13] TEAM KISSFLOW. Business Processes Hierarchy - The Ultimate Guide *kissflow* [online]. 2024 [cit. 2024-05-09]. Dostupné z: https://kissflow.com/workflow/bpm/business-process-hierarchy/

[14] HOYIKJOON. Business Process Improvement. *Tech Blog* [online]. 2014 [cit. 2024-05-09]. Dostupné z: https://developsoftware.wordpress.com/2014/03/26/business-process-improvement/

[15] TOBOLKA, Martin. Synergie ArchiMate s BPMN™ + software zdarma. *Tayllorcox* [online]. 2019 [cit. 2024-05-09]. Dostupné z: https://www.tx.cz/blog/prinosy-vyuziti-archimate-s-bpmn-plus-bpmn-software-zdarma

[16] CCMI. BPMN. *Centrum pro konceptuální modelování a implementace* [online]. © 2016 [cit. 2024-05-09]. Dostupné z: https://ccmi.fit.cvut.cz/metodiky/bpmn/

[17] CAMUNDA. Timer Events. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/timer-events/

[18] CAMUNDA. Message Events. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/message-events/

[19] CAMUNDA. Signal Events. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/signal-events/

[20] CAMUNDA. Error Event. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/error-events/

[21] CAMUNDA. Exclusive Gateways. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/exclusive-gateways/

[22] CAMUNDA. Parallel Gateways. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/exclusive-gateways/

[23] CAMUNDA. Inclusive Gateways. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/inclusive-gateways/

[24] CAMUNDA. Event-Based Gateways. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/event-based-gateways/

[25] LUCIDCHART. BPMN Activity Types. *Lucidchart* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://www.lucidchart.com/pages/bpmn-activity-types

[26] CAMUNDA. Service Task. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/service-tasks/

[27] CAMUNDA. User Tasks. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/user-tasks/

[28] CAMUNDA. Manual Tasks. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/manual-tasks/

[29] CAMUNDA. Subprocesses. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/subprocesses/

[30] CAMUNDA. Embedded-subprocesses. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/embedded-subprocesses/

[31] CAMUNDA. Call-activities. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/call-activities/

[32] CAMUNDA. Event subprocesses. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://docs.camunda.io/docs/components/modeler/bpmn/event-subprocesses/

[33] MICROSOFT. What is Business Process Model and Notation? *Microsoft* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://www.microsoft.com/en-us/microsoft-365/visio/business-process-modeling-notation

[34] MCNILLSON, Samuel. BPMN Artifact Types Explained. *StackOverflow-academy* [online]. 2022 [cit. 2023-11-12]. Dostupné z: https://academy.stackflows.com/bpmn-artifact-types-explained/

[35] ASTON, Ben. 10 Best Workflow Diagram Software For Process Mapping In 2024 *dpm* [online]. 2024 [cit. 2024-05-09]. Dostupné z: https://thedigitalprojectmanager.com/tools/best-workflow-diagram-software/

[36] PEERSPOT. Camunda pros and cons. *Peerspot* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://www.peerspot.com/products/camunda-pros-and-cons#con-aspect-container

[37] CAMUNDA. Camunda Pricing. *Camunda* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://camunda.com/pricing/

[38] PEERSPOT. Visual Paradigm Reviews. *Peerspot* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://www.peerspot.com/products/visual-paradigm-reviews#products-show

[39] VISUALPARADIGMA. VisualParadigm Pricing. *VisualParadigm* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://www.visual-paradigm.com/shop/vp.jsp

[40] CREATELY. Creately documentation. *Creately* [online]. © 2008-2024 [cit. 2024-05-09]. Dostupné z: https://creately.com/plans/

[41] BPMN.IO. BPMN.iO documentation. *BPMN.iO* [online]. © 2023 [cit. 2024-05-09]. Dostupné z: https://bpmn.io/

[42] BLENTA. Yaoqiang BPMN Editor. *SourceForge* [online]. 2023 [cit. 2024-05-09]. Dostupné z: https://sourceforge.net/projects/bpmn/

[43] POLANČIČ, Gregor. Common BPMN Modeling Mistakes and Best-Practices: Basic Events. *GEL* [online]. 2013 [cit. 2024-05-09]. Dostupné z: https://goodelearning.com/common-bpmn-modeling-mistakes-best-practices-basic-events/

[44] ROZMAN, Tomislav. Analysis of Most Common Process Modelling Mistakes in BPMN Process Models. *Slideshare* [online]. 2009 [cit. 2024-05-09]. Dostupné z: https://www.slideshare.net/tomirozman/eurospi2007trozman

[45] WORKFLOWPATTERNS. Patterns. *WORKFLOWPATTERNS* [online]. © 2010-2023 [cit. 2024-05-09]. Dostupné z: http://workflowpatterns.com/patterns/

[46] BURKE, Rory. . *Fundamentals of Project Management* [online]. Second edition. Burke Publishing, 2017 [cit. 2024-05-09]. ISBN 9780994149213. Dostupné z: https://books.mec.biz/tmp/books/KHSKE2BOJHLTNSJMQXTQ.pdf

[47] SCHWALBE, Kathy. *Introduction to project management* [online]. Fifth edition. Minneapolis: Schwalbe Publishing, 2015 [cit. 2024-05-09]. ISBN: 978-0-12-799959-3. Dostupné z: doi: 10.1016/B978-0-12-799959-3.00021-5

[48] FREELO TEAM. Nejčastější chyby v projektovém řízení a jak se jim vyhnout. *Freelo* [online]. 2022 [cit. 2024-05-09]. Dostupné z: https://www.freelo.io/cs/nejcastejsi-chyby-v-projektovem-rizeni-a-jak-se-jim-vyhnout

[49] ŠNAJDR, Petr. Analýza požadavků. *Podnikatelské příběhy* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://www.podnikatelskepribehy.cz/analyza-pozadavku-neni-jen-proste-precteni-toho-co-si-zadavatel-preje/

[50] FREELO. Freelo documentation. *Freelo* [online]. © 2023 [cit. 2024-05-09]. Dostupné z: https://help.freelo.io/en/

[51] BRUK, Vojtěch. Freelo recenze (2024). *Vojtechbruk* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://vojtechbruk.cz/recenze/freelo/

[52] MARTÍNEK, Lukáš. Freelo recenze: Vyplatí se v roce 2024? *NástrojeProWeb* [online].2021 [cit. 2024-05-09]. Dostupné z: https://www.nastrojeproweb.cz/clanky/freelo-recenze

[53] HÁLEK, Roman. Recenze aplikace Freelo: komunikujte v týmu efektivně. *SmartMania* [online]. 2021 [cit. 2024-05-09]. Dostupné z: https://smartmania.cz/freelo-aplikace-recenze/

[54] TRELLO. Začínáme s Trellem. *Atlassian Trello* [online]. © 2023 [cit. 2024-05-09]. Dostupné z: https://trello.com/guide

[55] GONZÁLEZOVÁ, Julie. Trello recenze 2024: Vyplatí se populární týmový nástroj? *NástrojeProWeb* [online]. 2021 [cit. 2024-05-09]. Dostupné z: https://www.nastrojeproweb.cz/clanky/trello-recenze

[56] MONDAY. Mobile app - My Work. *Monday* [online]. 2024 [cit. 2024-05-09]. Dostupné z: https://support.monday.com/hc/en-us/articles/360019159959-Mobile-app-My-Work

[57] DUFFY, Jill a Khamosh PATHAK. Monday.com Review. *PC Mag* [online]. 2024 [cit. 2024-05-09]. Dostupné z: https://www.pcmag.com/reviews/mondaycom

[58] LAMBDEN, Duncan. Monday.com Review: Features, Pros and Cons. *Tech.co* [online]. 2024 [cit. 2024-05-09]. Dostupné z: https://tech.co/project-management-software/monday-review

[59] YOUTRACK. YouTrack Documentation. JETBRAINS. *YouTrack* [online]. © 2000–2024 [cit. 2024-05-09]. Dostupné z: https://www.jetbrains.com/help/youtrack/

[60] SECRET. Youtrack reviews. *Secret* [online]. © 2024 [cit. 2024-05-09]. Dostupné z: https://www.joinsecret.com/youtrack/reviews

[61] MICROSOFT. What is .NET MAUI? *.NET MAUI* [online]. 2023 [cit. 2024-05-05]. Dostupné z: https://learn.microsoft.com/en-us/dotnet/maui/what-is-maui?view=net-maui-8.0.

[62] MICROSOFT. Build a .NET MAUI Blazor Hybrid app. *ASP.NET Core* [online]. 2024 [cit. 2024-05-05]. Dostupné z: https://learn.microsoft.com/en-us/aspnet/core/blazor/hybrid/tutorials/maui?view=aspnetcore-8.0

[63] Mudblazor. What is MudBlazor? *MudBlazor* [online]. © 2020-2024 [cit. 2024-05-05]. Dostupné z: https://mudblazor.com/mud/introduction#first-step:-mudblazor-as-a-component-library-

[64] TEHERAN, Miguel. Starting With Blazor + MudBlazor. *C# Corner* [online]. 2022 [cit. 2024-05-05]. Dostupné z: https://www.c-sharpcorner.com/blogs/starting-with-blazor-mudblazor

[65] MICROSOFT. Create web APIs with ASP.NET Core. *ASP.NET Core* [online]. 2023 [cit. 2024-05-05]. Dostupné z: https://learn.microsoft.com/cs-cz/aspnet/core/web-api/?view=aspnetcore-8.0

[66] MICROSOFT. ASP.NET Core Razor components. *ASP.NET Core* [online]. 2024 [cit. 2024-05-05]. Dostupné z: https://learn.microsoft.com/cs-cz/aspnet/core/blazor/components/?view=aspnetcore-8.0

[67] SCHMITT, Jacob. Unit testing vs integration testing. *Circleci Blog* [online]. 2024, APR 2, 2024 [cit. 2024-05-09]. Dostupné z: https://circleci.com/blog/unit-testing-vs-integration-testing/

[68] MICROSOFT. Unit Testing ASP.NET Web API 2. *ASP.NET Core* [online]. 2022 [cit. 2024-05-09]. Dostupné z: https://learn.microsoft.com/en-us/aspnet/web-api/overview/testing-and-debugging/unit-testing-with-aspnet-web-api

[69] FOWLER, Martin. Test Driven Development. *Martinfowler* [online]. 2023 [cit. 2024-05-09]. Dostupné z: https://martinfowler.com/bliki/TestDrivenDevelopment.html

SEZNAM PŘÍLOH

PŘÍLOHA A - APLIKACE

Název přílohy: Dokoupil_st60982_app

Obsah přílohy:

- Složka (TechCollaborator) se zdrojovými kódy napsanými v jazyce C# spolu se spustitelným řešením ve Visual Studio 2022.
- Aplikace k instalaci pro mobilní zařízení TechCollaborator.apk.