



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ

DEPARTMENT OF INFORMATION SYSTEMS

NÁSTROJ PRO PODPORU MANAGEMENTU RIZIK

RISK MANAGEMENT SUPPORT TOOL

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE

AUTHOR

Bc. SÁRA ŠKUTOVÁ

VEDOUcí PRÁCE

SUPERVISOR

doc. RNDr. JITKA KRESLÍKOVÁ, CSc.

BRNO 2020

Zadání diplomové práce



Studentka: **Škutová Sára, Bc.**
Program: Informační technologie Obor: Informační systémy
Název: **Nástroj pro podporu managementu rizik**
Risk Management Support Tool
Kategorie: Softwarové inženýrství

Zadání:

1. Seznamte se se znalostními oblastmi managementu projektů dle aktuálního standardu PMI. Zaměřte se na procesy znalostní oblasti managementu rizik zejména na plánování, identifikaci a kvalitativní analýzu rizik.
2. Specifikujte požadavky na systém pro podporu plánování, identifikaci a kvalitativní analýzu rizik v projektech vývoje softwarových produktů. Systém navrhnete.
3. Zvolte vhodné vývojové prostředí a implementujte prototyp funkcí navrženého systému vybraných po dohodě s vedoucí.
4. Na vzorku dat, vybraném po dohodě s vedoucí, ověřte funkčnost vytvořeného prototypu nástroje.
5. Zhodnoťte dosažené výsledky a diskutujte možnosti jeho dalšího rozšíření.

Literatura:

- *A Guide To The Project Management Body Of Knowledge: Sixth Edition*, Project Management Institute, Inc. 2017. ISBN 978-1-62825-184-5.
- SCHWALBE Kathy. *Řízení projektů v IT*: Computer Press, a.s., 2007, 720 s., ISBN 978-80-251-1526-8.
- PANDIAN C. Ravindranath.: *Applied Software Risk Management - A Guide for Software Project Managers*: Auerbach Publications, 2007. ISBN 0-8493-0524-1.
- VOSE David. *Risk Analysis A Quantitative Guide*: John Wiley & Sons, Inc., 2008. ISBN 978-0-470-51284-5.
- RAIS Karel, SMEJKAL Vladimír. *Řízení rizik ve firmách a jiných organizacích*: Grada Publishing, a.s., 2006. ISBN 80-247-1667-4.

Při obhajobě semestrální části projektu je požadováno:

- Splnění bodů 1 a 2 zadání a rozpracování bodu 3.

Podrobné závazné pokyny pro vypracování práce viz <https://www.fit.vut.cz/study/theses/>

Vedoucí práce: **Kreslíková Jitka, doc. RNDr., CSc.**

Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.

Datum zadání: 1. listopadu 2019

Datum odevzdání: 3. června 2020

Datum schválení: 21. října 2019

Abstrakt

Tato diplomová práce se zabývá řízením rizik v projektech. Cílem bylo nastudovat znalostní oblasti řízení projektu, hlavně pak plánování, identifikaci a kvalitativní analýzu z oblasti řízení rizik. Na základě informací z požadované oblasti se navrhl nástroj, který má pomoci k lepší vizualizaci rizik na projektu a zároveň podpořit jejich identifikaci a analýzu. Navržený nástroj byl naprogramován v kombinaci Java Spring Boot server a React klient. V závěru se pak pojednává o testování a možném budoucím rozšíření aplikace.

Abstract

This master thesis deals with risk management in projects. The goal was to study the knowledge areas of project management, especially planning, identification and qualitative analysis of risk management. Based on the information from the required area, a tool was designed to help visualize the risks on the project while supporting their identification and analysis. The designed tool was programmed with the combination of Java Spring Boot server and React client. At the end of this thesis you can find result of the testing and possible future expansion of the application.

Klíčová slova

znalostní oblasti řízení projektu, PMBOK, PMI, řízení rizik, riziko, webová aplikace, REST API, Java, Spring Boot, JavaScript, React, Material-UI

Keywords

knowledge areas of project management, PMBOK, PMI, risk management, risk, web application, REST API, Java, Spring Boot, JavaScript, React, Material-UI

Citace

ŠKUTOVÁ, Sára. *Nástroj pro podporu managementu rizik*. Brno, 2020. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. RNDr. Jitka Kreslíková, CSc.

Nástroj pro podporu managementu rizik

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracovala samostatně pod vedením paní doc. RNDr. Jitky Kreslíkové, CSc. Uvedla jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpala.

.....
Sára Škutová
30. května 2020

Poděkování

Ráda bych poděkovala doc. RNDr. Jitce Kreslíkové, CSc. za motivaci, podporu a odborné rady, které jsem využila při řešení této diplomové práce.

Obsah

1	Úvod	3
2	Znalostní oblasti managementu projektů	4
2.1	Základní pojmy projektového řízení	4
2.2	Procesy a znalostní oblasti řízení projektu	5
2.3	Řízení integrace projektu	5
2.4	Řízení rozsahu prací projektu	10
2.5	Řízení časového plánu v rámci projektu	13
2.6	Řízení nákladů projektu	17
2.7	Řízení kvality v rámci projektu	19
2.8	Řízení zdrojů v rámci projektu	21
2.9	Řízení komunikace v rámci projektu	25
2.10	Řízení rizik projektu	27
2.10.1	Plánování řízení rizik	27
2.10.2	Identifikace rizik	30
2.10.3	Kvalitativní analýza rizik	33
2.11	Řízení obstarávání v rámci projektu	37
2.12	Řízení zainteresovaných stran	40
3	Agilní řízení projektů	43
3.1	Agilní přístup ke znalostním oblastem	43
4	Návrh nástroje pro podporu řízení rizik	46
4.1	Specifikace požadavků	46
4.2	Diagram případů užití	47
4.3	Schéma databáze	49
4.4	Návrh uživatelského rozhraní	51
5	Realizace a implementace	53
5.1	Vybrané aplikační rámce serverové části	53
5.1.1	Spring	53
5.1.2	Struts	53
5.1.3	Play	54
5.1.4	DropWizard	54
5.2	Vybrané aplikační rámce klientské části	54
5.2.1	Angular	54
5.2.2	React	54
5.2.3	Vue.js	54

5.3	Zvolené programovací nástroje	55
5.3.1	Spring a Spring Boot	55
5.3.2	React	57
5.3.3	Další důležité knihovny	58
5.3.4	Podpůrné programovací nástroje	59
5.4	Implementace nástroje	60
5.4.1	Server	60
5.4.2	Klient	63
6	Testování	76
6.1	Automatizované testování	76
6.2	Postman a API	76
6.3	Uživatelské testování	77
7	Závěr	79
7.1	Možná rozšíření	80
	Literatura	81
	A Návrh grafického uživatelského rozhraní	83
	B API serveru	85
B.1	RoleController	85
B.2	UzivatelController	85
B.3	KategorieController	86
B.4	RegistrRizikController	86
B.5	OblastOtazkyController	86
B.6	OdpovedController	86
B.7	OtazkaController	86
B.8	DotaznikController	87
B.9	ProjektController	87
	C Příklad dotazníku	89
	D Testovací scénáře použitelnosti aplikace	90
	E Použité knihovny při tvorbě klientské části	96
	F Návod ke zprovoznění aplikace	98
F.1	Server	98
F.2	Klient	99
F.3	Přihlašovací údaje	99
	G Obsah paměťového média	100

Kapitola 1

Úvod

Ačkoliv by se některým lidem mohlo zdát, že management projektů je relativně mladým oborem, tak opak je pravdou. Už i v dávné minulosti se uplatňovaly jisté techniky řízení projektů. Vezměme si například mnohé české, ale i světové památky, bez dobrého řízení a plánování by jejich stavby nikdy nebyly dokončené a jistě by se nedochovaly do takového stavu v jakém jsou dnes k vidění.

K dobrému plánování a řízení patří i kvalitní analýza rizik, které se mohou v průběhu životního procesu projektů vyskytnout. Riziko v tomto případě můžeme považovat za vše, od finančních problémů a rizik, které můžeme předpokládat, až po nenadálé události, které negativně mohou ovlivnit průběh a výsledek projektu. Často se stává, že právě díky správnému plánování a řízení rizik se zakázka zdárně dovede do uspokojivého ukončení.

Cílem této diplomové práce je vytvořit nástroj, který by pomohl řízení projektů, a to hlavně při plánování, identifikaci a kvalitativní analýze rizik.

Tento dokument je rozdělen do sedmi kapitol. Ve 2. kapitole bude představen současný standartní pohled na řízení projektů, přičemž hlavní zaměření je na znalostní oblasti managementu rizik. Dále bude ve 3. kapitole představeno, jaký má na řízení rizik pohled agilní plánování projektu včetně i nejpoužívanějších metod, podle kterých probíhá agilní řízení. 4. kapitola, Návrh nástroje pro podporu řízení rizik, specifikuje požadavky na požadovaný nástroj a popisuje celý návrh aplikace včetně i návrhu obrazovek. Pro znázornění návrhu se použil diagram případů užití a schéma databáze. Implementace je popsána v kapitole 5, kde na začátku jsou rozepsány nejpoužívanější aplikační rámce pro naprogramování serveru v jazyce Java, které je následováno rozborem knihoven a aplikačních rámců pro implementaci klientské části v JavaScriptu. Pokračuje se pak popisem vybraných aplikačních rámců jimiž jsou Spring Boot a React. V druhé části kapitoly jsou pak odděleně představeny implementace serverové i klientské části. V kapitole 6 se popisují použité techniky při testování nástroje – automatické testování funkčnosti serveru, testování odpovědí na dotazy posílané na server a uživatelské testování použitelnosti klientské části. V závěru se pak shrnují dosažené výsledky a pojednává o možných budoucích rozšířeních vytvořeného nástroje.

Kapitola 2

Znalostní oblasti managementu projektů

V této kapitole lze nalézt souhrn znalostních oblastí managementu projektů, jejich vstupy a výstupy, přičemž se text více zaměřuje na procesy znalostní oblasti managementu rizik zejména na plánování, identifikaci a kvalitativní analýzu rizik. Ještě předtím se zde vysvětlí, co se v tomto textu považuje za projekt a projektové řízení. Pokud není určeno jinak, tak se informace převzaly z *A guide to the project management body of knowledge (PMBOK guide)* [2], kde se popisuje nejnovější standard od firmy **PMI (Project Management Institute)**. V další části jsou pak popsány informace, jak se k řízení projektů a rizik staví agilní techniky. Je vhodné také zmínit, že ačkoliv následující text popisuje vhodné techniky při řízení projektu, vždy záleží na specifických potřebách daného projektu a doporučené praktiky by se proto měly přizpůsobovat.

2.1 Základní pojmy projektového řízení

Velmi často se budou v textu vyskytovat slova **projekt**, **riziko** a **řízení projektů**, je proto vhodné si na začátek ujasnit co se danými pojmy v kontextu této práce myslí.

Projekt

Projektem v tomto textu lze chápat jako snahu, která má svůj definovaný začátek i konec. Jeho cílem je vytvořit produkt, službu nebo výstup, a to dle předem dohodnutých požadavků a omezení. Části jednoho projektu se pak v jiných mohou opakovat [12].

Riziko

Riziko je nejistá událost nebo podmínka, která pokud nastane, má negativní nebo pozitivní vliv na dosažení cíle projektu [12].

Řízení projektů

Řízení projektů lze chápat jako uplatnění znalostí, dovedností, nástrojů a technik v projektových činnostech s cílem splnit nebo překročit potřeby zájmových skupin a jejich očekávání od projektu [12].

2.2 Procesy a znalostní oblasti řízení projektu

Procesy řízení projektů jsou aktivity, které se všeobecně provádějí ve všech projektech. Cílem každého procesu je vytvořit na základě svých vstupů požadované výstupy, a to pomocí doporučených nástrojů či technik. Jednotlivé procesy se mohou prolínat a některé se také mohou několikrát opakovat v průběhu životního cyklu projektu (záleží to na specifických potřebách projektů).

Procesy lze různě kategorizovat. Jedním z možných způsobů je jejich rozdělení do pěti **procesních skupin**:

- Zahajovací – procesy prováděné při začátku nového projektu nebo nové fáze projektu.
- Plánovací – procesy, které mají za cíl zmapovat rozsah projektu, jeho cíle a způsoby, jak jich dosáhnout, do této skupiny patří největší část procesů.
- Prováděcí – procesy potřebné k realizaci cílů projektů.
- Monitorovací a kontrolní – procesy pro regulaci a monitorování postupů při práci na projektu.
- Uzavírací – procesy potřebné pro ukončení práce na projektu (nebo fázi) a jeho celkové uzavření.

Další možnou kategorizaci jsou **znalostní oblasti** (někdy také nazýváno jako oblasti poznatků [16]). PMI popisuje celkem deset oblastí, každá z nich je pak popsána pomocí svých vstupů, výstupu, doporučených technik, praktik a procesů, které obsahují. V následující části této kapitoly jsou představeny jednotlivé oblasti, přičemž řízení rizik je pak popsáno v samostatné části.

Ještě předtím je ale vhodné zde uvést dvě zkratky vlivů na projekt, které se vyskytují ve většině znalostních oblastí (hlavně při plánování), a to **EEF** a **OPA**.

EEFs neboli *Enterprise Environmental Factors*, lze přeložit jako **faktory podnikového prostředí**. Jsou to vlivy na projekt, které často vycházejí zpoza prostředí projektu nebo organizace (projektový tým nemá vliv na to, zda se faktor objeví). Příkladem můžou být: infrastruktura, dostupný software, schopnosti zaměstnanců, firemní kultura, legislativa, stav na trhu, komerční databáze, standarty.

OPAs neboli *Organizational process assets*, lze přeložit jako **procesní aktiva organizace**. Jsou to interní vlivy na projekt. Jde o různé plány, procedury a znalosti nabyté z předcházejících projektů, které si organizace uchovává k pozdějšímu znovuvyužití. Patří tam také interní databáze znalostí, například registr rizik.

2.3 Řízení integrace projektu

Tato oblast se zejména věnuje kombinování a koordinaci všech ostatních znalostních oblastí a procesů v nich. Řízení integrace má na starosti manažer projektu, který má všechny potřebné informace k jeho úspěšné koordinaci. Jeho hlavním úkolem je rozhodovat o tom jaké jsou potřeby prostředky pro úspěšné dokončení projektu, vyhledávat nové alternativní přístupy k řízení a přizpůsobování všech znalostních oblastí k jeho potřebám. Skládá se ze sedmi následujících procesů.

Vytvoření zadávací listiny projektu

Cílem je vytvořit dokument, který oficiálně potvrzuje vznik projektu. V době vykonávání tohoto procesu by se také měl určit manažer projektu. Tento dokument pak umožňuje manažerovi přiřazovat jednotlivé zdroje k projektovým aktivitám.

Vstupy

- Projektový záměr, obsahuje informace z obchodního prostředí, umožňuje určit, zda je důvod projekt řešit.
- Dohody a smlouvy mezi organizací a zákazníkem.
- EEF
- OPA

Výstupy

- Zakládající listina projektu, která kromě formálního potvrzení vzniku projektu obsahuje také další dodatečné informace, například: důvod vzniku, hlavní požadavky, celková rizikovost, seznam klíčových zainteresovaných stran, ukončovací kritéria, atd.
- Protokol předpokladů obsahující hlavní strategické předpoklady a omezení.

Vytvoření plánu řízení projektu

Proces, ve kterém se koordinují všechny plánovací části na projektu (plánovací aktivity ze všech znalostních oblastí). Výsledkem je jednotný plán řízení projektu, který určuje, jak se budou provádět jednotlivé aktivity po dobu celého životního cyklu projektu.

Vstupy

- Zakládající listina projektu sloužící jako inicializační dokument k plánování.
- Výstupy týkající se plánování z procesů z dalších znalostních oblastí.
- EEF
- OPA

Výstupy

- Plán řízení projektu, který obsahuje jednotlivé plánovací výstupy z ostatních znalostních oblastí, příkladem může být plán řízení rizik popsany v části [2.10](#).

Realizace plánu projektu

Jedná se o proces, který je vykonáván po celou dobu trvání projektu. Provádí se v něm jednotlivé aktivity, které byly popsány v plánu řízení projektu. Umožňuje celkové řízení činností na projektu a kontrolu požadovaných výstupů a tím zvyšuje pravděpodobnost jeho úspěšného dokončení.

Vstupy

- Plán řízení projektu.
- Projektové dokumenty, do kterých patří informace o stavu projektu, hotových částech, časový plán projektu, ale také registry rizik a reporty o rizicích atd.
- Schválené požadavky na změnu, ať už implementace projektu nebo samotného plánu řízení. Navržené změny musí vycházet z procesu celkové koordinace změn, který je popsán později v této znalostní oblasti.
- EEF
- OPA

Výstupy

- Výstupy, které jsou požadované z projektu, jeho fáze nebo procesu.
- Údaje o výkonu práce, kde se jedná o data zpozorovaná při provádění aktivit na projektu, například: začáteční a ukončující data naplánovaných aktivit, technické parametry, počet chyb, skutečné náklady atd.
- Záznam o problémech a nedostatecích, které se musí řešit včetně jejich řešení.
- Seznam požadavků na změny. Jsou to formální požadavky na jakékoliv změny v projektu. Většinou se aplikuje, pokud se zjistí nějaký nedostatek.
- Editace plánu řízení projektu, které vzešly z požadavků na změny.
- Editace projektových dokumentů o nové poznatky a změny, například změny v registru rizik nebo v seznamu aktivit.
- Editace OPA.

Řízení projektových znalostí

V tomto procesu se zkoumají již existující znalosti, které organizace dříve nabyla a zjišťuje se, které z nich by se mohly využít v aktuálním projektu. Pokud potřebná znalost neexistuje je ji potřeba nabýt a v ideální případě i zaevidovat. Mezi znalosti se nepočítají pouze ty uvedené v dokumentech, ale také zkušenosti a „know-how“ členů týmu. Proces probíhá po celou dobu vykonávání činností na projektu.

Vstupy

- Plán řízení projektu.
- Projektové dokumenty, zde myšleno jako registr nabytých poznatků nebo informace o přerozdělení členů týmu.
- Cíle a výstupy projektu.
- EEF
- OPA

Výstupy

- Registr nabytých poznatků, kde se může popisovat vyskytnutá situace včetně vlivů a možných řešení.
- Editace plánu řízení projektu.
- Editace OPA o nové znalosti.

Monitorování a kontrola projektových prací

Další z procesů, co se provádějí po celou dobu životního cyklu projektu. V jeho průběhu se monitorují veškeré pracovní činnosti na projektu a posuzuje se, jak pomáhají naplňovat jeho cíl. Díky tomu lze zhodnotit aktuální stav projektu a navrhnout případné změny či opravy.

Vstupy

- Plán řízení projektu, podle kterého se kontrolují jednotlivé aktivity.
- Projektové dokumenty ze všech znalostních oblastí, včetně registru rizik a zprávy o rizicích.
- Informace o výkonu práce, kde se porovnávají údaje o výkonu práce s komponentami z plánu řízení. Umožňuje posoudit stav projektu.
- Dohody a smlouvy se zákazníkem nebo s externími dodavateli.
- EEF
- OPA

Výstupy

- Zpráva o pracovním výkonu, která reprezentuje získané informace o pracovním výkonu. Pomáhá při rozhodování.
- Seznam požadavků na změny, které vzešly z informací o pracovním výkonu.
- Editace plánu řízení projektu.
- Editace projektových dokumentů.

Celková koordinace změn

Cílem tohoto procesu je zhodnotit všechny požadavky na změny a jejich pozitivní či negativní vlivy na projekt. Proces má na starosti manažer projektu a je vykonáván po celou dobu provádění pracovních aktivit na projektu. Přijaté požadavky na změny si můžou vynutit aktualizací plánu řízení a projektových dokumentů. Vstupem do tohoto procesu je velká část dokumentů, které byly vytvořeny v předcházejících procesech.

Vstupy

- Plán řízení projektu, který hlavně obsahuje plán řízení změn a informace o rozpočtu projektu.
- Projektové dokumenty.
- Zprávy o pracovním výkonu.
- Seznam požadavků na změny.
- EEF
- OPA

Výstupy

- Seznam přijatých požadavků na změny, které budou v pozdějších procesech zavedeny do projektu.
- Editace plánu řízení projektu.
- Editace projektových dokumentů.

Uzavření projektu nebo fáze

Má za úkol ukončování všech aktivit na projektu nebo fázi jeho cyklu. Kontroluje se, zda všechny požadavky byly splněny a zda všechny dokumenty obsahují aktuální informace. Projekt se archivuje a zabrané zdroje organizace se uvolňují k dalšímu použití. V případě neúspěšného ukončení projektu se zkoumají a dokumentují důvody neukončení. Vstupem do procesu mohou být všechny dokumenty vytvořené na projektu.

Vstupy

- Zadávací listina projektu obsahuje akceptační kritéria.
- Plán řízení projektu.
- Projektové dokumenty.
- Akceptované cíle a výstupy.
- Obchodní dokumenty.
- Dohody a smlouvy se zákazníkem.
- Zadávací dokumentace.
- OPA

Výstupy

- Editace projektových dokumentů, ve kterých se zaznamenávají poslední informace a určují se jako finální verze. Hlavně se to týká registru nabytých poznatků.
- Hotová verze výstupu projektu (finální produkt).
- Finální zpráva obsahující zhodnocení celého projektu.
- Editace OPA.

2.4 Řízení rozsahu prací projektu

Má za cíl zajistit, aby se při řešení projektu vyskytovaly pouze ty pracovní činnosti, které vedou k jeho úspěšnému dokončení a má pomoci definovat a vyvarovat se zbytečným pracím. Dle [16] patří určení rozsahu práce k jednomu z nejobtížnějších stránek řízení projektu. Skládá se ze šesti procesů.

Plánování řízení rozsahu

Proces, který vytváří plán řízení rozsahu (je součástí plánu řízení projektu). V tomto plánu má být popsáno, jakým způsobem se bude definovat rozsah projektu a jak bude tento rozsah kontrolován.

Vstupy

- Zakladající listina projektu obsahující základní informace o projektu.
- Plán řízení projektu, včetně plánu řízení kvality či popisem životních cyklů projektu.
- EEF
- OPA

Výstupy

- Plán řízení rozsahu projektu, popisující jak budou probíhat ostatní procesy v této znalostní oblasti. Jeho přesná forma závisí na potřebách daného projektu.
- Plán řízení požadavků je součástí plánu řízení. Popisuje celkovou správu požadavků na daný projekt.

Sběr požadavků

Umožňuje definovat rozsah projektu na základě požadavku zainteresovaných stran na jeho výstupy. Tyto požadavky se vhodně definují a dokumentují, a tím zjednodušují řízení daného projektu. Další znalostní oblasti pak tyto požadavky dále zpracovávají a následně aktualizují plán řízení.

Vstupy

- Zakladající listina projektu.
- Plán řízení projektu, skládající se hlavně z plánů vytvořených v předcházejícím procesu.
- Projektové dokumenty, hlavně registr nabytých poznatků, předpoklady a registr zainteresovaných stran.
- Obchodní dokumenty.
- Dohody a smlouvy s určenými požadavky.
- EEF
- OPA

Výstupy

- Dokumentace požadavků, ve které se analyzují jednotlivé požadavky a jejich propojení s obchodními potřebami projektu. Požadavky je pak možné rozdělovat do kategorií, například obchodní, požadavky zainteresovaných stran či požadavky na kvalitu.
- Matice sledování požadavků, která spojuje požadavky od jejich zadání (základu) až po odpovídající výstup.

Definování rozsahu

Ze všech požadavků, které se vyskytly při sběru, má proces definování rozsahu za úkol vybrat ty, co se skutečně budou uspokojovat. Díky tomu se pak může vytvořit detailnější popis projektu. Dále se stanoví jeho předběžný rozsah, který se postupem dalšího zpracování a plánování upravuje.

Vstupy

- Zadávací listina projektu.
- Plán řízení projektu obsahující plán řízení rozsahu.
- Projektové dokumenty, včetně registru rizik a dokumentace požadavků.
- EEF
- OPA

Výstupy

- Deklarace rozsahu projektu umožňuje provádět podrobnější plánování. Stanovuje rozsah projektu a blíže popisuje požadované výstupy.
- Editace projektových dokumentů. Patří tam například registr zainteresovaných stran.

Vytváření WBS

V tomto procesu se seznam požadovaných výstupů a tím i práce na nich rozdělí na menší, lépe zvládnutelné části. Výsledkem je pak hierarchická struktura **WBS** (*work breakdown structure*). Nejnižším komponentám ve WBS struktuře se říká pracovní balíčky. Ty umožňují seskupovat potřebné aktivity na projektu.

Vstupy

- Plán řízení projektu obsahující plán řízení rozsahu, který popisuje, jak se má WBS struktura vytvořit.
- Projektové dokumenty, včetně deklarace rozsahu projektu (výstup z předcházejícího procesu) a dokumentace požadavků.
- EEF, například standardy o tvorbě WBS.
- OPA

Výstupy

- Směrný plán rozsahu obsahující popis rozsahu, strukturu WBS s pracovními balíčky a slovníkem WBS.
- Editace projektových dokumentů.

Ověřování rozsahu

Zájemové skupiny zde formálně ověřují výstupy z jednotlivých částí projektu, čímž se zároveň také akceptuje celkový rozsah projektu. V případě neakceptovaných výstupů se navrhuje opravné změny. Proces je vhodné provádět periodicky po celou dobu životního cyklu projektu.

Vstupy

- Plán řízení projektu s plánem řízení rozsahu, plánem řízení požadavků a směrným plánem rozsahu.
- Projektové dokumenty, tedy registr znalostí nebo zprávy o kvalitě.
- Ověřené cíle a výstupy.
- Údaje o výkonu práce.

Výstupy

- Schválené cíle a výstupy, které splňují akceptační kritéria.
- Informace o výkonu práce, tj. seznamy přijatých, případně nepřijatých (i s odůvodněním) výstupů.
- Požadavky na změny a opravy nepřijatých výstupů.
- Editace projektových dokumentů, například editace matice sledování požadavků nebo aktualizace registru nabytých znalostí.

Kontrola rozsahu

Další z periodicky se opakujících procesů. Jeho cílem je ujistit se, že všechny požadavky a doporučení na změny projdou procesem Celkové koordinace změn ze znalostní oblasti Řízení integrace změn 2.3. Dále má také kontrolovat změny, které se vyskytnou a tím zajistit úspěšné ukončení projektu.

Vstupy

- Plán řízení projektu s plány zaměřenými na řízení změn a rozsah projektu.
- Projektové dokumenty.
- Údaje o výkonu práce.
- OPA

Výstupy

- Informace o výkonu práce s porovnáním aktuálního stavu projektu s tím, jak by měl vypadat dle směrného plánu rozsahu.
- Požadavky na změny v rozsahu projektu.
- Editace plánu řízení a jeho součástí.
- Editace projektových dokumentů o nové poznatky.

2.5 Řízení časového plánu v rámci projektu

Tato znalostní oblast má za cíl spravovat celé časové plánování projektu. Jejím hlavním výstupem je detailní plán popisující dobu a způsob dodání jednotlivých výstupů z projektu. Časový plán se také využívá pro komunikaci se zainteresovanými stranami a pomáhá v posuzování stavu projektu. Jednou z jeho vlastností by měla být flexibilita, neboť se často stává, že s nabývajícím znalostmi a postupem řešení projektu se mohou objevovat nové informace a aktivity, které se musí vzít v úvahu. Kvůli této neurčitosti se často adaptují agilní metody plánování (více o agilním řízení v kapitole 3.1). K této oblasti bylo přiřazeno šest následujících procesů.

Plánování řízení časového rozvrhu

Proces, ve kterém se stanovuje, jakým způsobem se bude spravovat a dokumentovat celková správa časového rozvrhu projektu, tj. od plánování až po kontrolu. Výstupem je pak časový plán projektu.

Vstupy

- Zakladající listina projektu obsahující dohodnuté časové milníky.
- Plán řízení projektu s plánem řízení rozsahu a dokumentem o způsobu vývoje.
- EEF, můžou se použít pomocné programy na tvoření časového rozvrhu.
- OPA

Výstupy

- Plán řízení časového rozvrhu, který mezi jinými umožňuje určit jaké metody a nástroje se použijí při tvorbě modelu časového rozvrhu. Dále také nastavuje kontrolní prahové hodnoty, kdy je ještě akceptovatelná odchylka od původního plánu.

Definování činností

Proces patřící do plánovací skupiny, jehož cílem je identifikovat a zdokumentovat jednotlivé aktivity na projektu, které vedou k jeho úspěšnému dokončení. Jednotlivé aktivity se vytvoří pomocí analýzy pracovních balíčků ze struktury WBS. Definování činností je jedním z procesů, co se vykonávají po celou dobu životního cyklu projektu.

Vstupy

- Plán řízení projektu, který obsahuje dříve vytvořený plán řízení časového rozvrhu a směrný plán rozsahu, který hlavně obsahuje strukturu WBS.
- EEF
- OPA

Výstupy

- Seznam činností, které se na projektu musí provést, včetně i jejich popisku.
- Atributy činností rozšiřují předcházející výstup, když umožňují identifikovat dodatečné komponenty k aktivitám.
- Seznam milníků, kde každý milník představuje důležitou událost v projektu.
- Požadavky na změny, které vzešly z dodatečné analýzy projektu.
- Editace plánu řízení projektu.

Řazení činností

Cílem procesu je identifikovat a zdokumentovat vztahy mezi jednotlivými aktivitami, tedy určit jejich předchůdce a následovníky. Umožňuje to lépe přerozdělovat práci a pospojovat činnosti co spolu souvisejí. Patří do skupiny procesu, které se vykonávají po celou dobu vykonávání práce nad projektem.

Vstupy

- Plán řízení projektu s plánem řízení časového rozvrhu a směrným plánem.
- Projektové dokumenty popisující jednotlivé aktivity a milníky.
- EEF
- OPA

Výstupy

- Síťový graf projektu, který graficky reprezentuje logické vztahy mezi naplánovanými aktivitami.
- Editace projektových dokumentů popisující jednotlivé aktivity a milníky.

Odhadování doby trvání činností

Jedná se o určení délky provádění jednotlivých aktivit. K tomuto odhadu se využívají informace o dostupných zdrojích, schopnostech, ale i o motivaci členů týmu. Kromě odhadu doby se také odhaduje, kolik zdrojů bude potřeba k dokončení daných činností. Výsledná doba je pouze předpokládaný odhad a často obsahuje dodatečný rezervní čas.

Vstupy

- Plán řízení projektu se použije podobně jako v předcházejících procesech.
- Projektové dokumenty od seznamů aktivit přes kalendáře dostupnosti zdrojů až po registry rizik.
- EEF
- OPA

Výstupy

- Odhady dob trvání činností, které určují pravděpodobný počet pracovních period potřebných k dokončení činnosti.
- Podklady k odhadům obsahující dodatečné informace k tomu, jak se určily.
- Editace projektových dokumentů týkající se atributů aktivit nebo i registru znalostí.

Sestavení časového rozvrhu

Na základě předcházejících procesů, ze znalostní oblasti řízení časového plánu v rámci projektu a jejich výstupů, se sestaví časový rozvrh. Celé sestavování většinou probíhá v několika iteracích, kdy se z dostupných informací sestaví jedna verze, která se následně reviduje a opravuje. Výsledkem je pak konečná reálná verze časového plánu, podle které lze monitorovat aktivity na projektu.

Vstupy

- Plán řízení projektu.
- Projektové dokumenty zabývající se aktivitami a odhady, seznam milníků, přiřazení členů týmu anebo i registr rizik.
- Dohody a smlouvy.
- EEF
- OPA

Výstupy

- Směrný plán časového rozvrhu je komponentou plánu řízení projektu. Jde o schválený model časového rozvrhu, který se používá při kontrole, kdy se porovnává se skutečnými údaji projektu.
- Harmonogram projektu přiřazuje aktivitám jejich naplánovanou dobu provádění, milníky a zdroje. Může mít podobu Ganttových diagramů, diagramu milníků či obyčejné tabulky.
- Informace k rozvrhu.
- Projektové kalendáře identifikují skutečnou časovou dobu práce.
- Požadavky na změny v komponentách plánu řízení projektu.
- Editace plánu řízení projektu, například plán řízení časového rozvrhu.
- Editace projektových dokumentů týkajících se například aktivit, rizik, získaných znalostí.

Kontrola časového rozvrhu

Poslední proces z této znalostní oblasti a zároveň další z řady těch co se realizují od začátku až do ukončení projektu. Jeho úkolem je ho monitorovat a v případě neshod aktualizovat jeho časový rozvrh.

Vstupy

- Plán řízení projektu s komponentami, které se týkají řízení času nebo rozsahu.
- Projektové dokumenty časového rozvrhu nebo i registr znalostí.
- Údaje o výkonu práce s informacemi o započatých a ukončených aktivitách.
- OPA

Výstupy

- Informace o výkonu práce s porovnáním aktuálního stavu projektu a tím, jak by měl vypadat dle směrného plánu časového rozvrhu.
- Prognózy časového rozvrhu v závislosti na předcházejících a aktuálních informacích.
- Požadavky na změny, které vzešly z dodatečného přezkoumání pokroku práce.
- Editace plánu řízení projektu.
- Editace projektových dokumentů.

2.6 Řízení nákladů projektu

Dle [16] náklady lze považovat za něco čeho se vzdáme výměnou za něco jiného, a ačkoliv se velmi často myslí pouze peněžní náklady, lze do nich zařadit i jiné zdroje. Cílem této znalostní oblasti je celková správa všech výdajů potřebných pro úspěšné dokončení všech aktivit na projektu. Stará se také o to, aby náklady na všechny tyto aktivity nepřekročily schválený rozpočet. V oblasti se nacházejí čtyři následující procesy.

Plánování řízení nákladů

Proces by se měl provést minimálně jednou a nejlépe jako jeden z prvních. Jeho výstupem je plán řízení nákladů, který určuje, jak se budou jednotlivé náklady definovat, dokumentovat, řídit či monitorovat.

Vstupy

- Zakladající listina projektu popisuje již povolené finanční zdroje.
- Plán řízení projektu, který by měl obsahovat plán řízení časového rozvrhu a plán řízení rizik.
- EEF
- OPA

Výstupy

- Plán řízení nákladů jako komponenta plánu řízení projektu popisuje celkovou správu nákladu. Mohou se v něm objevovat informace jako přesnost nákladů, seznam kontrolních mezí či způsob řešení odchylek.

Odhadování nákladů

V tomto procesu je úkolem vytvořit hrubý odhad nákladů na zdroje potřebné k dokončení projektu. Proces je vhodné provádět periodicky, neboť každý odhad se získává z aktuálních informací o projektu, přičemž ty se postupem času mohou měnit. Jak se také projekt bude blížit ke svému dokončení, tak i odhady se budou více blížit ke skutečným celkovým nákladům. Je také vhodné vzít na vědomí možnou inflaci a nedefinovat náklady pouze v peněžní hodnotě, ale také jako nutný počet odpracovaných hodin neboli „člověkohodiny“.

Vstupy

- Plán řízení projektu s plánem řízení nákladů a kvality, lze použít i směrný plán rozsahu a kvality.
- Projektové dokumenty včetně registru nabytých znalostí, registrem rizik nebo i požadavky na zdroje.
- EEF
- OPA

Výstupy

- Odhady nákladů s rezervou pro dodatečnou práci. Do odhadů je připočtena i částka pro řešení nepředvídatelných událostí.
- Podklady odhadování poskytující dodatečné informace o tom, jak se k odhadům došlo.
- Editace projektových dokumentů, například registru znalostí.

Sestavení rozpočtu

Cílem procesu je sloučení odhadů nákladů na jednotlivé aktivity na projektu do jednoho celku – směrného plánu nákladů. Díky němu lze pak lépe monitorovat a kontrolovat výkon pracovních aktivit na projektu.

Vstupy

- Plán řízení projektu, může obsahovat plán řízení nákladů a zdrojů, ale i směrný plán rozsahu.
- Projektové dokumenty týkající se odhadů nákladů, dále registr rizik nebo i projektový rozvrh.
- Obchodní dokumenty neboli projektový záměr a plán řízení přínosů.
- Dohody a smlouvy o zdroje, které je potřeba zakoupit pro práci s projektem.
- EEF
- OPA

Výstupy

- Směrný plán nákladů je součástí rozpočtu projektu. Přesněji se jedná o jeho časově rozfázovanou verzi bez rezerv pro dodatečnou práci.
- Požadavky na financování projektu, které jsou odvozené od směrného plánu nákladů.
- Editace projektových dokumentů o nová rizika či odhady nákladů.

Kontrola nákladů

V tomto procesu, který probíhá po celou dobu běhu projektu, se monitoruje stav všech aktivit a nákladů na ně a pokud je potřeba, tak se aktualizuje směrný plán nákladů a tím i celkový rozpočet.

Vstupy

- Plán řízení projektu s plány týkajícími se kontroly nákladů a výkonnosti.
- Projektové dokumenty společně s registrem nabytých znalostí.
- Požadavky na financování projektu.
- Údaje o výkonech s daty o provedených nákladech.
- OPA

Výstupy

- Informace o výkonech, jak si projekt vede s porovnáním se směrným plánem nákladů.
- Prognózy rozpočtu.
- Požadavky na změny nákladů nebo časového rozvrhu.
- Editace plánu řízení projektu, hlavně jeho částí týkajících se nákladů.
- Editace projektových dokumentů včetně registru rizik nebo protokolu předpokladů.

2.7 Řízení kvality v rámci projektu

Existuje mnoho definic kvality, jedna z nich, ale říká, že jde o shodu s požadavky zainteresovaných stran [16]. Cílem řízení kvality je tedy aplikovat znalosti týkající se plánování, správy a kontroly k dosažení těchto požadavků. Zdroj [16] se dále zmiňuje i o tom, že pouhé dodržení požadavků nemusí být známkou kvality, když o tom, zda výsledný produkt má přijatelnou kvalitu rozhoduje až zákazník, a pokud se nenaplní jeho očekávání, tak to může způsobit další náklady na projekt kvůli případným opravám. V této znalostní oblasti se nacházejí tři procesy zaměřené na kvalitu.

Plánování řízení kvality

V tomto procesu se identifikují požadavky na projekt a normy, které umožňují dodržet kvalitu daných nároků. Dále se také určuje, jak se bude dokumentovat prolínání standardů a požadavků. Patří do skupiny plánovacích procesů a měl by probíhat paralelně s ostatními procesy a případně i požadovat opravy ostatních plánů z této skupiny.

Vstupy

- Zakladající listina projektu obsahuje vlastnosti projektu včetně i jeho požadavků a kritérií.
- Plán řízení projektu obsahující plány řízení rizik, řízení požadavků nebo i směrný plán rozsahu.
- Projektové dokumenty, ve kterých může být registr rizik, registr zainteresovaných stran, ale i dokumenty požadavků.
- EEF
- OPA

Výstupy

- Plán řízení kvality popisuje jak dosáhnou požadovaných kvalit, včetně potřebných zdrojů.
- Metriky kvality popisují jednotlivé atributy projektu a jak je bude proces kontroly kvality ověřovat.
- Editace plánu řízení projektu hlavně pak plán řízení rizik a směrný plán rozsahu.
- Editace projektových dokumentů včetně registru rizik a zainteresovaných stran.

Řízení kvality

Cílem procesu je aplikovat informace z plánu řízení kvality do skutečných aktivit. Jeho součástí je i zajištění kvality, kde se ujišťuje, že jednotlivé procesy projektu jsou efektivně využívány, dle doporučených standardů, k dosažení co nejlepší kvality. Proces se provádí po celou dobu životního cyklu projektu.

Vstupy

- Plán řízení projektu, který obsahuje plán řízení kvality.
- Projektové dokumenty týkající se kvality a rizik.
- OPA

Výstupy

- Zprávy o kvalitě, které umožňují reagovat na problémy týkající se nedostatečné kvality.
- Dokumenty testů a vyhodnocení dosažení cílů kvality.
- Požadavky na změny plánů, procesů a dokumentů projektu.
- Editace plánu řízení projektu o nové informace získané z tohoto procesu.
- Editace projektových dokumentů, hlavně pak registru rizik nebo registru nabytých znalostí.

Kontrola kvality

V tomto procesu se vyhodnocují výsledky z jednotlivých výstupů projektu, přičemž se určuje, zda výsledky jsou správné a splňují očekávání zákazníka. Musí tedy splňovat příslušné normy a standardy. Dalším cílem je také určit korigující akce k opravě nedostatků na kvalitě. K získání co nejlepších výsledků se tento proces provádí po celou dobu běhu projektu.

Vstupy

- Plán řízení projektu s plánem řízení kvality určuje, jak bude kontrola probíhat.
- Projektové dokumenty, například metriky kvality či dokumenty testů a vyhodnocení.
- Schválené požadavky na změny plánů, dokumentů či opravy projektu.
- Cíle a výstupy projektu, které se porovnávají s akceptačními kritérii.
- Údaje o výkonech a kvalitě pracovních činností na projektu.
- EEF
- OPA

Výstupy

- Kontrolní měření kvality jsou zdokumentované aktivity z tohoto procesu.
- Ověřené cíle a výstupy, které prošly kontrolou kvality a poté se formálně kontrolují v procesu Ověřování rozsahu z oblasti Řízení rozsahu prací projektu 2.4.
- Informace o výkonu práce, kde se popisuje, jak projekt splňuje požadavky, s jakou kvalitou a jakým způsobem lze opravit případné nedostatky.
- Požadavky na změny vyplývající z kontroly.
- Editace plánu řízení a jeho komponent, hlavně pak plánu řízení kvality.
- Editace projektových dokumentů, například registru rizik a správy problémů.

2.8 Řízení zdrojů v rámci projektu

V pořadí šestá znalostní oblast se zajímá o zdroje, které se v projektu využívají a které jsou potřeba k jeho úspěšnému dokončení. Pomáhá hlavně k tomu, aby správné zdroje byly k dispozici ve správný čas a na správném místě. Lze je rozdělit na dvě skupiny: **lidské** a **fyzické**. Fyzické zdroje představují materiály, přístroje, potřebný software atd. Je důležité, aby tyto zdroje byly správně a efektivně řízené, v opačném případě hrozí vznik rizik, které mohou projekt ohrozit. Druhou kategorií jsou lidské zdroje, kterou reprezentují všichni členové projektového týmu. Obecně má manažer na starosti ne jenom úspěšně řídit projekt, ale také se starat o blaho lidských zdrojů a také podporovat jejich motivaci k práci, když dle [16] jsou lidé nejdůležitějším aktivem firmy. V oblasti se nachází šest procesů ze správy zdrojů.

Plánování řízení zdrojů

Další z plánovacích procesů, jeho úkolem je určit jakým způsobem získat a spravovat zdroje potřebné k úspěšnému dokončení projektu. Mělo by se brát na vědomí i to, že některé zdroje nemusí být vždy dostupné anebo o ně bude muset firma soutěžit s konkurencí (anebo i s jinými projekty v samotné organizaci).

Vstupy

- Zakladající listina projektu obsahuje o něm informace, ale také i dohodnuté finanční zdroje.
- Plán řízení projektu s plánem řízení kvality a směrným plánem rozsahu.
- Projektové dokumenty s registrem zainteresovaných stran, rizik, projektovým rozvrhem a dokumentací požadavků.
- EEF
- OPA

Výstupy

- Plán řízení zdrojů popisuje celkovou správu zdrojů, jejich identifikaci, role, rozvojové aktivity pro členy týmu atd.
- Zakládací listina týmu definuje informace o týmu, akceptovatelné vztahy, komunikační návody atd.
- Editace projektových dokumentů např. registr rizik nebo protokol předpokladů.

Odhad množství zdrojů

V tomto procesu se určuje druh a počet zdrojů potřebných k práci na projektu. Probíhá paralelně s ostatními procesy např. s Odhadováním nákladů z oblasti Řízení nákladů 2.6. Ke zdrojům se také specifikuje jejich charakteristika.

Vstupy

- Plán řízení projektu, který mezi jinými obsahuje plán řízení zdrojů a směrný plán rozsahu.
- Projektové dokumenty, hlavně zaměřené a náklady, aktivity, rizika a zdroje.
- EEF
- OPA

Výstupy

- Požadavky na zdroje určují druhy a počty potřebných zdrojů pro každou aktivitu na projektu.
- Podklady k odhadům poskytují dodatečné informace k jednotlivým požadavkům.
- Hierarchická struktura zdrojů třídící zdroje do kategorií, typů, rolí atd.
- Editace projektových dokumentů, např. aktivitám se přiřadily zdroje.

Získávání zdrojů

Cílem procesu je získat potřebné zdroje, ať už z interního (podnikového) nebo externího prostředí. Dále umožňuje přiřadit je k jednotlivým aktivitám. V případě, že se zdroje nepodaří získat, tak hrozí i neúspěšné ukončení projektu.

Vstupy

- Plán řízení projektu, hlavně obsahující plán řízení zdrojů, plán řízení obstarávání (znalostní oblast Řízení obstarávání v rámci projektu 2.11) a směrný plán nákladů.
- Projektové dokumenty týkající se zdrojů a aktivit.
- EEF
- OPA

Výstupy

- Přidělení fyzických zdrojů, které se v projektu využijí.
- Přiřazení rolí a odpovědností členům týmu.
- Kalendáře zdrojů, které specifikují kdy jsou k dispozici.
- Požadavky na změny jakékoliv části projektu.
- Editace plánu řízení projektu, hlavně plánu řízení zdrojů a směrného plánu nákladů.
- Editace projektových dokumentů týkajících se zdrojů a rizik, ale také i např. registr nabytých znalostí.
- Editace EEF, kde se popíše počet použitých vlastních zdrojů a také nově nabyté zdroje.
- Editace OPA o informace týkající se získávání nových zdrojů.

Vytvoření projektového týmu

Proces, který probíhá po celou dobu životního cyklu projektu. Jedná se o zdokonalování schopností členů týmu a zlepšení vztahů mezi nimi. Díky tomu se vylepšuje týmová práce a tým i zpříjemňuje a zefektivňuje práci na projektu. Hlavním aktérem procesu je manažer projektu, který má na starosti motivaci členů týmu a vytvoření efektivního týmového prostředí.

Vstupy

- Plán řízení projektu společně s plánem řízením zdrojů poskytuje návody, jak zvládat celkovou správu členů týmu např. jak je trestat nebo kdy a jak odměňovat.
- Projektové dokumenty, např. o přiřazení členů týmu, projektových rozvrh, kalendáře zdrojů.
- EEF
- OPA

Výstupy

- Hodnocení výkonu týmu, včetně toho, jak se zlepšily schopnosti a kompetence členů týmu.
- Požadavky na změny, které vzešly v důsledku tohoto procesu.
- Editace plánu řízení projektu a jeho komponent z důvodu nových informací.
- Editace projektových dokumentů týkajících se lidských či fyzických zdrojů.
- Editace EEF o hodnocení dovedností.
- Editace OPA doplněním informací o nových schopnostech zaměstnance.

Vedení týmů

Další z procesů, co probíhají po celou dobu provádění projektu. V tomto procesu se sledují výkony jednotlivých členů týmu, řeší jejich vzájemné konflikty a celkově se snaží optimalizovat jejich výkon a tím i efektivitu práce na projektu. Podobně jako v předcházejícím procesu i zde je hlavním vykonavatelem manažer projektu.

Vstupy

- Plán řízení projektu s plánem řízení zdrojů obsahují informace, jak zdroje řídit a uvolňovat k dalšímu využití.
- Projektové dokumenty týkající se správy zaměstnanců.
- Zpráva o pracovním výkonu každého zaměstnance umožňuje naplánovat budoucí požadavky na zdroje pro daného zaměstnance, ale také i jeho případné odměny či tresty.
- Hodnocení výkonu týmu je výstupem z předcházejícího procesu.
- EEF
- OPA

Výstupy

- Požadavky na změny všech aspektů projektu, příkladem může být změna přiřazení člena týmu.
- Editace plánu řízení projektu a jeho komponent (plán řízení zdrojů, směrný plán rozvrhu nebo rozsahu).
- Editace projektových dokumentů podle potřeby.
- Editace EEF o schopnosti zaměstnance.

Kontrola zdrojů

Podobně jako v předcházejících oblastech i v této se provádí kontrola. Jejím cílem je kontrolovat, zda potřebné zdroje jsou vždy dostupné, zda jsou využívány přesně jak bylo naplánováno a pokud již nejsou potřeba, tak zda byly správně uvolněné k dalšímu využití.

Vstupy

- Plán řízení projektu a jeho komponenty.
- Projektové dokumenty hlavně zaměřené na zdroje, ale také i registr rizik.
- Údaje o výkonech můžou obsahovat informace o použitých zdrojích.
- Dohody a smlouvy s externími dodavateli zdrojů.
- OPA

Výstupy

- Informace o výkonu práce, ve kterých se porovnávají požadavky na zdroje a jejich následné využití.
- Požadavky na změny nedostatků, které vyšly najevo při kontrole.
- Editace plánu řízení projektu dle potřeby.
- Editace projektových dokumentů o nově vzniklé informace.

2.9 Řízení komunikace v rámci projektu

Ve [16] se zmiňuje, že nedostatečná komunikace může být jednou z největších hrozeb projektu. V této znalostní oblasti se tedy zaručuje, že ke projektu jsou vždy k dispozici potřebné informace neboli se zajišťuje efektivní komunikace se zainteresovanými stranami. Zároveň se oblast také věnuje aktivitám, které jsou potřebné pro tvorbu komunikační strategie.

Komunikuje se hlavně z důvodu předávání informací, a to způsobem verbálním či neverbálním. Přenášet informace lze i v psané podobě či pomocí žurnalistických médií. Aktivity v komunikaci mohou být formální, neformální, v interním či externím prostředí atd. Důležité ovšem je, aby všechny strany, které se komunikace účastní, plně porozuměly obsahu této výměny informací. V oblasti jsou tři následující procesy.

Plánování řízení komunikace

Proces, který podle potřeby lze periodicky opakovat v průběhu životního cyklu projektu. Jeho cílem je se dohodnout na způsobu komunikace a také vytvoření plánu řízení, a to na základě potřeb všech účastníků na projektu. Je potřeba také určit, jak se získané informace budou ukládat.

Vstupy

- Zakládací listina projektu obsahuje seznam zainteresovaných stran.
- Plán řízení projektu s plánem řízení zdrojů a zainteresovaných stran (znalostní oblast Řízení zainteresovaných stran 2.12).
- Projektové dokumenty s dokumenty požadavků a registrem zainteresovaných stran.
- EEF
- OPA

Výstupy

- Plán řízení komunikace umožňuje určit, jak se informace budou sbírat, předávat, uchovávat, může také obsahovat šablony k zápisu schůzek týmu nebo k emailových zpráv.
- Editace plánu řízení projektu, hlavně jeho komponenty – plánu zapojení zainteresovaných stran.
- Editace projektových dokument např. v projektovém rozvrhu se přidají informace o komunikačních aktivitách.

Řízení komunikace

V tomto procesu je hlavním cílem celková správa informací týkajících se projektu, týká se to také výběru komunikačních kanálů. To umožňuje plynulou a efektivní výměnu potřebných dat mezi všemi účastníky projektu. Je tedy potřebné, aby se proces prováděl po celou dobu prací na projektu.

Vstupy

- Plán řízení projektu společně s plány řízení zdrojů, komunikace a plánem zapojení zainteresovaných stran.
- Projektové dokumenty, obsahující jakékoliv informace, které se musí předat, příkladem mohou být zprávy o kvalitě nebo o rizicích.
- Zpráva o pracovním výkonu se předkládá zainteresovaným stranám.
- EEF
- OPA

Výstupy

- Projektové komunikace, popisují komunikaci probíhající na projektu.
- Editace plánu řízení projektu, hlavně plánu řízení komunikace, pokud se něco změní na přístupu v této oblasti.
- Editace projektových dokumentů o nově vzniklé údaje, např. nová rizika nebo nové komunikační aktivity.
- Editace OPA o zprávy o projektu či korespondenční listiny.

Monitorování komunikace

Tento proces se také vykonává po celou dobu běhu projektu. Zaručuje se v něm, že všichni účastníci mají aktuální informace o projektu. Dále také monitoruje, zda se plán řízení komunikace dodržuje a zda komunikace netvoří informační nedorozumění.

Vstupy

- Plán řízení projektu a jeho komponenty týkající se zdrojů, komunikace a zainteresovaných stran.
- Projektové dokumenty např. registr znalostí nebo projektové komunikace z minulého procesu.
- Údaje o výkonu práce mohou popisovat jaké typy komunikace proběhly.
- EEF
- OPA

Výstupy

- Informace o výkonu práce sdělují, jak komunikace v projektu probíhá.
- Požadavky na změny v komunikačních aktivitách.
- Editace plánu řízení projektu o informace na zlepšení efektivity komunikace.
- Editace projektových dokumentů o nové nabyté znalosti nebo nové požadavky na informace od zainteresovaných stran.

2.10 Řízení rizik projektu

Jak již bylo řečeno dříve, řízení rizik projektů není pouze o tom, jak se se vzniklými riziky vypořádat, ale také i o celém procesu plánování řízení rizik, zejména jejich identifikaci a analýze, a aby tým pracující na projektu věděl, do jaké míry jsou jednotlivá rizika ještě akceptovatelná. Náleží zde také plánování a následná implementace metod, jak se s danými událostmi vypořádat v případě, že nastanou a nesmí se zapomenou také i na aktivní monitorování projektů a sledování, zda se pravděpodobnost výskytu potenciálních rizik nezvyšuje a aktivně řešit ty, co se objeví.

Dříve v tomto textu, při vysvětlování pojmu rizika, bylo zmíněno, že nemusí mít pouze negativní smysl. Pozitivní rizika mohou mít na projekt povzbuzující účinky, mohou zjednodušit či zpříjemnit práci na projektu, případně i negovat nějaká negativní rizika, která se mohou vyskytovat v pozdějších fázích projektů. Cílem této znalostní oblasti je tedy maximalizování pravděpodobnosti vzniku pozitivní událostí a jejich případných kladných dopadů na projekt a také snížit možnosti vzniku negativní události (nebo i jejich celkové odvrácení) a jejich vlivů. Dále také spravovat rizika, o které se nepostaraly předcházející oblasti.

V každém projektu se vyskytují rizika, která by se mohla rozdělit do dvou kategorií [13]. **Vnitřní**, na které má manažer případně projektový tým vliv. Do této kategorie můžeme zařadit přiřazení pracovníků na dané pozice nebo potenciální neshody mezi členy týmu. Druhou kategorií jsou **vnější** rizika, na která tým pracující na projektů nemá vliv. Patří tam neočekávané změny zákonů, překvapivý vývoj na trhu nebo i obyčejné přerušení dodávky proudu. Jak lze vidět rizikům se nevyhne žádná společnost ani projekt. Jak se ovšem píše v [16], tak mnoho firem bylo schopné na trhu pouze díky tomu, že na sebe vzaly určité riziko, díky kterému se jim ve výsledku dostalo mnohonásobných výsledků.

Ať už daný tým rizika aktivně vyhledává nebo se jím snaží za každou cenu vždy předcházet, tak přístup k řízení rizik se musí vždy přizpůsobit velikosti projektu, jeho rozsahu, důležitosti a způsobu vývoje.

Každá znalostní oblast se skládá ze svých procesů, které mají své vstupy a výstupy. Podobně to probíhá i v oblasti řízení rizik projektů, která se skládá ze šesti procesů, které budou společně se svými vstupy a výstupy popsány na následujících stránkách. Je vhodné tady opět upozornit na to, že ačkoliv jsou procesy modelovány jako samostatné jednotky, tak ve skutečnosti se velmi často jednotlivé fáze procesů prolínají či propojují.

2.10.1 Plánování řízení rizik

Plánování řízení rizik je počáteční proces, ve kterém se má definovat, jak postupovat při aktivitách řízení rizik projektu a jak tyto aktivity naplánovat. Může být proveden jednou nebo vícekrát v životním cyklu projektu, ale doporučuje se, aby s ním začalo již při jeho

zadávání a měl by být ukončen v jeho počátečních fázích. Následně se také doporučuje, že pokud na projektu nastane nějaká vážnější změna nebo se v pozdějších procesech zjistí, že výstup z plánování řízení rizik již není dostačující, tak aby tým pracující na daném projektu neváhal a znovu daný proces provedl.

Při provádění plánování se musí řešitelský tým seznámit s předcházejícími projektovými dokumenty, které jsou hlavním vstupem do toho procesu. Jde hlavně o:

- **Zakladající listina projektu**, která popisuje projekt, jeho cíle a omezení a nejdůležitější rizika.
- **Plán řízení projektu se všemi komponentami**, a to právě z toho důvodu, aby výsledný plán byl v souladu se všemi předcházejícími plány.
- **Projektové dokumenty**, které jsou potřebné pro tento proces. Mezi těmito dokumenty se může nacházet i registr zúčastněných stran, který popisuje jednotlivé zúčastněné strany, jejich role a jak moc jsou ochotní na tomto projektu tolerovat rizika.
- **Faktory podnikového prostředí (EEF)**, na které projektový tým nemá vliv.
- **Procesní aktiva organizace (OPA)**, které mohou ovlivnit výstup tohoto procesu, konkrétně se může jednat o to jaký má společnost postoj k rizikovým faktorům, co považuje za rizika, stupně odpovědnosti za provádění rozhodnutí a také to co se naučili při podobných předcházejících projektech.

Při zpracovávání vstupů a tvoření výstupu z plánování řízení rizik se pak doporučuje použití alespoň jednoho způsobu z následujících technik či nástrojů:

- **Expertní posudek** od jednotlivce či skupiny, kteří aktivně používají techniky řízení rizik, případně jsou seznámeni s tím, jak organizace řídí rizika, či již ze zkušeností vědí jaká rizika se na daném projektu mohou vyskytnout.
- **Datová analýza**, která má mezi jinými taky za cíl analýzu tolerance rizik zainteresovaných stran.
- **Schůzky týmu**, na kterých se mají vytvořit plány pro jednotlivé aktivity v řízení rizik. Na schůzkách nemusí být přítomni pouze členové projektového týmu odpovědní za řízení rizik, manažeři či zainteresované strany, ale také osoby „zvenčí“ například zákazníci.

Výstupem celého procesu je pak **plán řízení rizik**, který je součástí plánu řízení projektu, a který má obsahovat, jak budou jednotlivé aktivity v plánu probíhat. Může se skládat z některý (dle potřeby i ze všech) následujících elementů:

- **Strategie rizik**, kde se popisuje obecný přístup k řízení rizik na daném projektu.
- **Metody**, které definují přístupy, nástroje a zdroje dat, které budou použity k řízení rizik
- **Role** a zodpovědnosti, tedy kdo je vedoucím, kdo má na starosti podporu a kteří členové týmu jsou také členy týmu na řízení rizik. Zároveň se také definují jejich zodpovědnosti za jednotlivé aktivity dle plánu.

RBS Úroveň 0	RBS Úroveň 1	RBS Úroveň 2
Všechny zdroje projektových rizik	Technická rizika	Technologie
		Definice rozsahu
		Definice požadavků
	Rizika řízení	Řízení projektu
		Komunikace
		Organizace
	Obchodní rizika	Interní procedury
		Klienti a zákazníci
		Dodavatelé a prodejci
	Externí rizika	Legislativa
		Regulace
		Konkurence

Tabulka 2.1: Příklad z Risk Breakdown structure (RBS) ukazující možné dělení rizik do kategorií (zdroj inspirováno z [2])

- **Financování** aktivit spojených s řízením rizik projektu včetně i protokolu pro uvolnění mimořádných rezerv při nepředvídatelných událostech, které mají snížit vyskytnuté riziko na přijatelnou míru.
- **Časové rozvržení**, které určuje kdy a jak často se budou provádět procesy spojené s řízením rizik. Zároveň také stanovuje, které aktivity z plánu se začlení do životního cyklu projektu.
- **Kategorie rizik** umožňuje seskupovat rizika do jednotlivých skupin. Tabulka 2.1 popisuje jedno možné rozdělení podle techniky **Risk Breakdown Structure (RBS)**. Jedná se o hierarchické rozdělení potenciálních rizik. Nejčastější kategorie jsou: technická rizika, obchodní, externí a rizika řízení projektu. Ke každému je pak určeno, z jakých zdrojů se potenciální rizika mohou objevit. To pak umožňuje projektovému týmu identifikovat rizika, což je další proces po plánování řízení rizik.
- **Tolerance zainteresovaných stran k rizikům**. Úrovně jednotlivých tolerancí by měly být zaznamenány a měly by být nastavené jako maximální prahové hodnoty, za které se nesmí překročit.
- **Stanovení pravděpodobnosti výskytu rizika a jeho důsledků**. Definice pravděpodobnosti mohou být vytvořeny speciálně na potřeby daného projektu a maximální tolerance klíčových zainteresovaných stran k rizikům nebo se mohou vybírat z obecných předem vytvořených definic. Typicky se skládá z třech nebo pěti stupnic, které určují, jak detailně se musí provádět jednotlivé procesy řízení rizik. Výsledek se může podobat tabulce na obrázku 2.1, kde se představují jednotlivé stupnice pro pravděpodobnost výskytu a jaké budou mít dopady na tři vybrané cíle v projektu. Díky tomu se pak můžou porovnávat jednotlivé příležitosti nebo hrozby a jejich dopady.
- **Matice pravděpodobnosti a dopadu rizika**. Informace se přebírají z hodnot, které se vytvářely v předcházejícím bodě, jde prakticky o jejich grafické zobrazení. Obrázek 2.2 představuje takovou matici, kdy příležitosti a hrozby jsou zde představeny jako události s pozitivním či negativním dopadem v závislosti na výši jejich

Důležitost	Pravděpodobnost	Vliv na cíle projektu		
		Čas	Náklady	Kvalita
Velmi vysoká	>70%	>6 měsíců	>\$5M	Velmi významný dopad na celkovou funkcionalitu
Vysoká	51 - 70%	3 - 6 měsíců	\$1M - \$5M	Významný dopad na celkovou funkcionalitu
Střední	31 - 50%	1 - 3 měsíce	\$501K - \$1M	Určitý dopad v klíčových funkčních oblastech
Nízka	11 - 30%	1 - 4 týdny	\$100K-\$500K	Menší dopad na celkovou funkcionalitu
Velmi nízká	1 - 10%	1 týden	<\$100K	Menší dopad na druhořadé funkce
Žádná	<1%	Beze změny	Beze změny	Žádné změny ve funkčnosti

Obrázek 2.1: Příklad definic pravděpodobností a jejich důsledků na tři cíle projektu (zdroj inspirováno z [2])

pravděpodobnosti. Výsledná hodnota v buňce pak vznikne slovním spojením hodnot na jednotlivých osách a v případně numerického ohodnocení os pak násobením jejich hodnot (jak je zobrazeno na obrázku 2.2). Výsledné skóre pak umožňuje posoudit prioritu s jakou se musí dané riziko posoudit.

- **Formát zpráv.** Zde se popisuje jaký formát budou mít výstupy z jednotlivých procesů řízení rizik projektu, neboli jak se budou dokumentovat, analyzovat či komunikovat mezi členy týmu.
- **Sledování,** způsob, jak se budou dokumentovat rizikové aktivity a jak budou probíhat kontroly jednotlivých procesů.

2.10.2 Identifikace rizik

Druhý z řady procesů řízení rizik projektu je identifikace rizik. Cílem je identifikovat všechna rizika, která nějakým způsobem mohou ovlivnit daný projekt, ať už k pozitivnímu či negativnímu důsledku. K dobrému pochopení rizik je také potřeba zkoumat jejich zdroje. K činnostem procesu se může pozvat mnoho účastníků, tj. od projektového manažera, přes experty v dané oblasti, které se projekt týká, až po koncové uživatele. Je ovšem důležité, aby se identifikace účastnily zainteresované strany, a hlavně členové projektového týmu. Díky tomu pak členové týmu mohou mnohem snadněji rozpoznat a reagovat na rizika. Celý proces se často může opakovat, když s tím, jak pokračuje životní cyklus projektu, se mohou objevovat nová rizika. Četnost opakování by měla být již určena v plánu řízení rizik, který je výstupem z předcházejícího procesu.

Při zahájení identifikace je potřeba zrevidovat několik vstupních dokumentů:

- **Plán řízení projektu,** který může obsahovat některé z následujících komponentů.
 - **Plán řízení požadavků** může naznačovat cíle náchylné na rizika.
 - **Plán řízení časového rozvrhu** naznačuje ne plně pochopené oblasti.
 - **Plán řízení nákladů** může obsahovat seznam rizikových oblastí.

		Threats					Opportunities						
Probability	Very High 0.90	0.05	0.09	0.18	0.36	0.72	0.72	0.36	0.18	0.09	0.05	Probability	
	High 0.70	0.04	0.07	0.14	0.28	0.56	0.56	0.28	0.14	0.07	0.04		
	Medium 0.50	0.03	0.05	0.10	0.20	0.40	0.40	0.20	0.10	0.05	0.03		
	Low 0.30	0.02	0.03	0.06	0.12	0.24	0.24	0.12	0.06	0.03	0.02		
	Very Low 0.10	0.01	0.01	0.02	0.04	0.08	0.08	0.04	0.02	0.01	0.01		
		Very Low 0.05	Low 0.10	Moderate 0.20	High 0.40	Very High 0.80	Very High 0.80	High 0.40	Moderate 0.20	Low 0.10	Very Low 0.05		
Negative Impact						Positive Impact							

Obrázek 2.2: Příklad matice pravděpodobnosti a dopadu rizika, která umožňuje posoudit skóre priority u daných rizik (zdroj převzato z [2])

- **Plán řízení kvality** naznačuje oblasti s rizikem nedodržení kvality.
 - **Plán řízení zdrojů** může obsahovat informace o rizikových zdrojích.
 - **Plán řízení rizik**, který je výstupem z předcházejícího procesu, a který může určovat role a zodpovědnosti za rizika, ale také i aktivity a kategorie rizik.
 - **Směrný plán rozsahu**, obsahuje strukturu WBS, a také kritéria pro akceptaci výstupů.
 - **Směrný plán rozvrhu** má seznam milníku, které se musí dodržet.
 - **Směrný plán nákladů** naznačuje problematické náklady.
- **Projektové dokumenty**, mezi kterými můžeme počítat protokol předpokladů, kde jednotlivé předpoklady a omezení můžou ovlivnit důležitost jednotlivých rizik nebo i úroveň celkového rizika projektu. Dále odhady nákladů, odhady časové náročnosti nebo dokumenty správy problémů, kde jednotlivé záznamy mohou zvýšit úroveň rizika. Využít se mohou také registry znalostí nabytých při předcházejících projektech. Ale také dokumentace ze specifikací požadavků, s požadavky na zdroje a také registr zainteresovaných stran.
 - **Dohody a smlouvy** s externími dodavateli.
 - **Dokumenty obstarávání**, kdy některé zdroje jsou poskytovány externími dodavateli a je potřeba zhodnotit jejich potenciální vlivy na vznik rizik v projektu.
 - **Faktory podnikového prostředí (EEF)**, tím se například myslí přístup k různým komerčním databázím rizik, nebo k akademickým a průmyslovým studiím zabývajících se riziky v podobných projektech.

- **Procesní aktiva organizace (OPA)**, neboli aktuální informace o projektu nebo kontrolní seznamy z předcházejících projektů.

Způsob, jakým se jednotlivé dokumenty mají zpracovat a co z nich pomůže k identifikaci rizik záleží vždy na druhu projektu. Následuje soupiska užitečných technik a nástrojů, které k tomu mají nápomoci.

- **Expertní posudek** se doporučuje podobně jako v předcházejícím procesu.
- **Brainstorming**, kde je cílem co nejrychlejší vytvoření seznamu rizik. Děje se to technikou rychlého skoro až bezmyšlenkovitého navrhování různých rizik. Je doporučeno, aby diskusi vedl moderátor nebo skupina expertů, kteří by udržovali tok diskuse a zároveň by navrhovali nové kategorie, ve kterých se mohou nacházet rizika. U brainstormingu je důležité se ujistit, že všechna rizika jsou popsána správně a srozumitelně.
- **Analýza kontrolních seznamů**. V těchto seznamech se můžou nacházet informace o rizicích, které se vyskytovaly na předcházejících projektech, čímž se urychlí identifikace na aktuálním. Doporučuje se ovšem, aby celá identifikace neprobíhala jenom pomocí kontrolních seznamů, když často se tam nenacházejí všechna rizika. Je také vhodné tento seznam jednou za čas aktualizovat o nové záznamy.
- **Rozhovory** se zkušenějšími členy týmu nebo experty.

Získaná data se pak mohou analyzovat pomocí jedné z následujících technik.

- **Analýza hlavních příčin**, se používá k nalézání příčin, které mohou vést ke problémům, následně se pak ze zjištěných dat vytvoří protipatření. Technika je vhodná i na hledání příležitostí.
- **Analýza předpokladů a omezení**, kde cílem je zjistit, zda navržené předpoklady a omezení jsou validní a zda některé nevedou k vytvoření negativního nebo i pozitivního rizika.
- **Analýzou dokumentů** lze zjistit hrozby, které se zobrazují jako nekonzistence mezi podobnými dokumenty nebo jako různé nejasnosti.
- **SWOT analýza**, kde se na projektu zkoumají jeho silné a slabé stránky, ale i příležitosti a hrozby (odtud je název techniky, jde o spojení prvních písmen anglických slov *strengths, weaknesses, opportunities* a *threats*). Začíná se s identifikací silných a slabých stránek firmy, hlavně se zaměřením na projekt (může se ovšem použít i při analýze společnosti jako takové). Pokračuje se nalezením možných příležitostí vycházejících ze silných stránek a hrozeb vycházejících ze slabých. SWOT analýza také umožňuje určit, zda nějaká silná stránka organizace nekompensuje hrozby nebo naopak zda nějaká slabá nebrání příležitostem.

Proces identifikace rizik má několik výstupů. Patří mezi ně:

- **Registr rizik** je nejdůležitějším výstupem. Obsahuje soupis všech rizik, pozitivních i negativních, které se podařilo identifikovat při analýze ze vstupních souborů. Registr je důležitý i při navazujících procesech z oblasti řízení rizik. Každé riziko by mělo mít své unikátní identifikační číslo a mělo by být srozumitelně popsáno, také by k němu měla být přiřazena zodpovědná osoba a pokud je to už známo, tak je vhodné uvést

i možnou reakci na dané riziko. V registru se můžou vyskytovat i další dodatečné informace jako například kategorie, aktuální stav, zdroje, možné vlivy na cíle projektu nebo i informace co může způsobit, že se riziko objeví.

- **Zpráva o rizicích** obsahuje zdroje rizik celkového projektu a také souhrnné informace o jednotlivých hrozbách. Podobně jako i v předcházejícím dokumentu i tady se můžou dodat dodatečné informace.
- **Editace projektových dokumentů.** Podle potřeby je vhodné aktualizovat existující projektové dokumenty o nově zjištěné informace. Týká se to například dokumentu předpokladu, správy problémů či registru znalostí (který následně může pomoci identifikovat rizika na nějakém pozdějším projektu).

2.10.3 Kvalitativní analýza rizik

V pořadí třetím procesem ve znalostní oblasti řízení rizik je kvalitativní analýza rizik. Jejím cílem je analýza jednotlivých rizik nalezených při identifikaci. Zkoumá se pravděpodobnost jejich vzniku, ale také i jaké můžou mít následky na projekt, priorita, případně i jiné charakteristiky. Tento proces je prováděn po celou dobu trvání projektu. Zároveň se tady pro každé riziko identifikuje zodpovědná osoba, která má na starost naplánovat odpovídající reakci a také má dohlédnout na to, aby firma byla schopna danou reakci uskutečnit.

Vstupem do kvalitativní analýzy rizik je opět několik dokumentů.

- **Plán řízení projektu**, kde nás hlavně zajímá plán řízení rizik, co všechno tento dokument obsahuje se lze dočíst v sekci o Plánování řízení rizik. Pokud tento plán nebyl před tím vytvořen, je možné jej vytvořit na začátku tohoto procesu a před použitím si ho nechat dodatečně odsouhlasit sponzory.
- **Projektové dokumenty**, přesněji se to týká protokolu předpokladů a registru rizik (bližší informace o těchto dokumentech lze nalézt v sekci o identifikaci rizik). Doporučuje se mít také připravený registr zainteresovaných stran, protože někteří z nich mohou být „vlastníci“ nějakého rizika (znamená to, že jejich úkolem je mít připravené protiopatření k případné hrozbě).
- **Faktory podnikového prostředí (EEF)**
- **Procesní aktiva organizace (OPA).**

Jako v předcházejících procesech i tady se jako užitečnou techniku či nástroj doporučuje **expertní posudek**. Použít lze i **kategorizace rizik**, kdy hrozby a příležitosti jsou rozdělené podle jejich zdrojů nebo podle toho jakou sféru projektu můžou ovlivnit (druhů rozdělení může být i více). Díky kategorizaci pak může projektový tým zaměřovat větší pozornost do míst, o kterých se ví, že budou nejvíce zatížené riziky. Co se týče sběru dat, tak tam je vhodné provádět rozhovory se zainteresovanými stranami. Při analýze sesbíraných dat se pak můžou použít následující techniky:

- **Hodnocení kvality dat k rizikům**, sleduje, jak kvalitní jsou data k jednotlivým rizikům. Pokud by data měla nízkou kvalitu, tak by byl výsledek kvalitativní analýzy nepoužitelný v daném projektu. Hodnotu dat lze ocenit pomocí dotazníku, který se předloží zainteresovaným stranám a který zjistí jejich chápání různých charakteristik projektu.

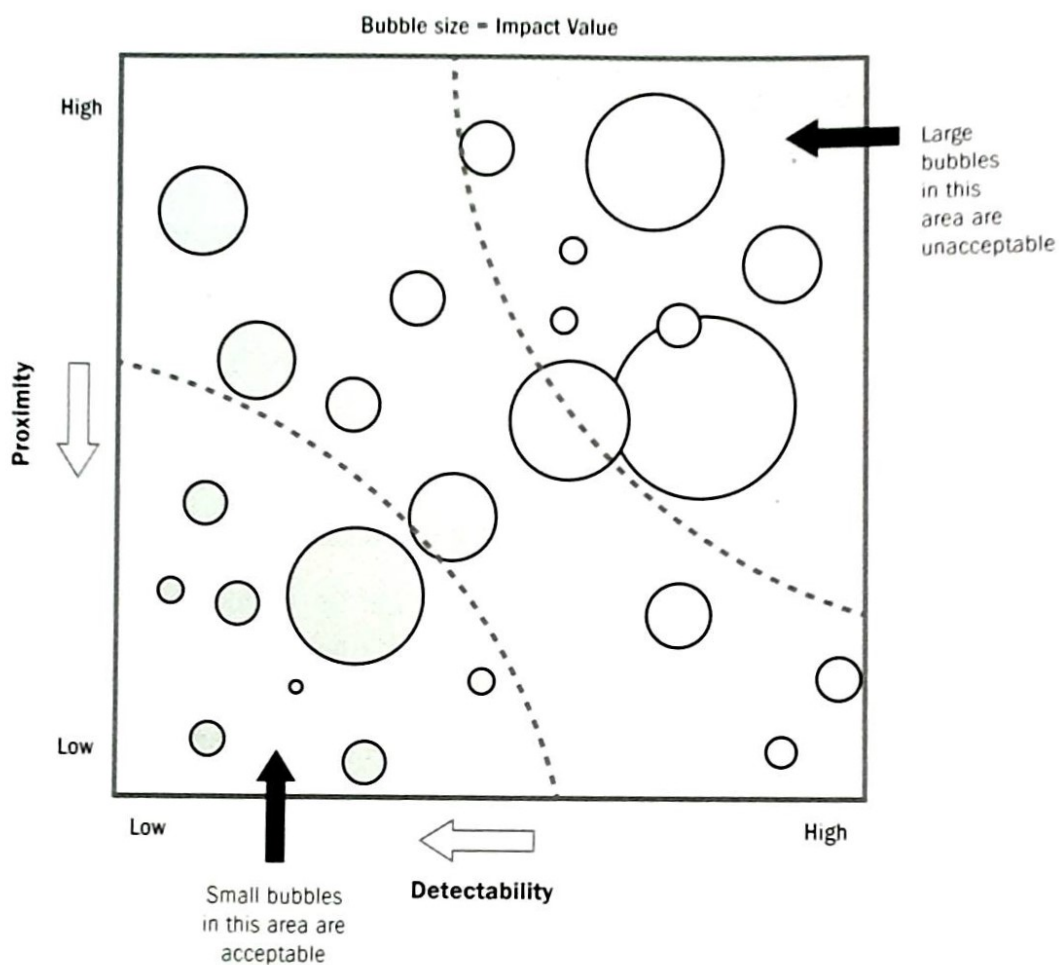
- **Posouzení pravděpodobnosti a dopadu rizika** zkoumá jaká je pravděpodobnost, že se dané riziko vyskytne. Dále se také ke každému riziku prověřuje jeho možné pozitivní či negativní dopady. Hodnocení rizik lze například provádět pomocí rozhovorů s účastníky na projektu, přičemž se ale musí brát v úvahu to, že někteří z nich mohou mít rozdílné názory na jejich důležitost a tím i pravděpodobnost výskytu a dopady. Možné hodnoty se vybírají z předem připravené tabulky, která se mohla vytvořit při tvorbě plánu řízení rizik. Možný příklad takové tabulky je na obrázku při definici pravděpodobnosti výskytu rizika a jeho důsledků. Je také doporučeno, aby se ke každému riziku přidalo zdůvodnění, proč dostalo dané hodnocení.
- **Posouzení ostatních parametrů rizik.** Pravděpodobnost výskytu a možné dopady nemusí být jediné charakteristiky, na které se projektový tým může zaměřit. Dále je vhodné zkoumat, jak mohou být hrozby propojené či jak dlouho od výskytu se objeví první vlivy na projektu. Jde například ale i o to, jak jednoduše může vlastník ovládat výskyt svého rizika případně kontrolovat jeho vlivy na projekt. Dodatečných vlastností může být i mnohem více. Díky nim lze pak mít mnohem lepší představu o prioritách jednotlivých rizik.

Získaná data je pak vhodné nějakým způsobem graficky prezentovat. Jednou z technik je **matice pravděpodobnosti a dopadu**. Jak již název napovídá jde o grafickou podobu dat z techniky při analýze dat se stejným jménem. Matice mapuje pravděpodobnost výskytu rizika s jeho dopady na cíle projektu, díky čemuž je pak lze rozdělit do prioritních skupin (určí se tím, která rizika se musí analyzovat jako první). Hodnoty možného výskytu a vlivů na jeden nebo více cílů v projektu jsou určeny pomocí tabulky definice pravděpodobnosti a dopadů rizika v plánu řízení rizik. Je také možné posoudit prioritu rizika pro každý cíl projektu odděleně a výslednou hodnotu určit jako jejich průměr nebo vybrat nejvyšší z nich. Další grafickou technikou může být **hierarchický graf**, který se doporučuje, když není možné použít matici a k porovnání rizik se používají více než jenom dva parametry. Příklad grafu lze vidět na obrázku 2.3, kde vliv rizika na projekt je zobrazen jako velikost kroužku.

Při výstupu z procesu kvalitativní analýzy rizik se podle potřeb **aktualizují projektové dokumenty**. V případě, že se objevily nové předpoklady, omezení nebo se některý z nich změnil, tak je nutné nové informace uvést do **protokolu předpokladů**. Je vhodné také aktualizovat **dokument správy problémů**. Hlavním výstupem je pak aktualizace **registru rizik**, kde se přidají nové informace o prioritách či skóre k jednotlivým rizikům. Dále se tam mohou přidat informace o pravděpodobnosti výskytu, možných dopadech, zda je nutné okamžitě riziko řešit, jeho kategorii či oznámení, že riziko se musí ještě více analyzovat. Posledním dokumentem může být **zpráva o rizicích**, kde se aktualizují informace o tom, která rizika jsou nejdůležitější (většinou ty s největší pravděpodobností vzniku a vlivu na projekt) a také se v něm vytvoří prioritní rizikový seznam.

Kvantitativní analýza rizik

Kvantitativní analýza se nemusí provádět na každém projektu, pokud je ovšem v plánu, tak se provádí po celou dobu běhu projektu a velmi často paralelně s kvalitativní analýzou. Často se provádí pouze při velmi rozsáhlých projektech. Cílem procesu je numericky analyzovat jednotlivá rizika a další možné nejasnosti (jejich pravděpodobnost, možné náklady atd.). Výstupy z toho procesu se velmi hodí do následujícího procesu plánování reakcí.



Obrázek 2.3: Příklad hierarchického grafu, velikost bubliny představuje vliv rizika na projekt (zdroj převzato z [2])

Vstupy

- Plán řízení projektu s plánem řízení rizik a směrnými plány rozsahu, nákladu a rozvrhu. Směrné plány umožňují určit od jakého bodu se má analýza provádět.
- Projektové dokumenty týkajících se rizik, požadavků, nákladů, rozvrhu atd.
- EEF
- OPA

Výstupy

- Editace projektových dokumentů, týká se to hlavně zprávy o rizicích, kde se popisuje celková úroveň rizikovosti projektu, jak vysoké musí být pohotovostní rezervy, seznam a analýza nejzávažnějších rizik, doporučené reakce na rizika atd.

Plánování reakcí na rizika

Tento proces umožňuje definovat, jak reagovat na jednotlivá rizika. Cílem je minimalizovat hrozby, maximalizovat příležitosti a celkově zmenšit rizikovost projektu. Adekvátní reakce by měly být nákladově efektivní. V případě potřeby se můžou definovat i náhradní reakce, které se použijí, pokud hlavní bude nedostatečná k řešení. Kromě reakcí se mohou definovat i strategie rozhodování. V tomto procesu se také identifikují sekundární rizika, která mohou vzniknout jako následek provádění dané reakce.

Vstupy

- Plán řízení projektu s plánem řízení rizik a zdrojů (zdroje potřebné pro reakci na rizika) a směrným plánem nákladů.
- Projektové dokumenty hlavně zaměřené na rizika, ale také registr zainteresovaných stran nebo dokumenty týkajících se zdrojů.
- EEF
- OPA

Výstupy

- Požadavky na změny zaměřené na náklady nebo rozvrh projektu.
- Editace plánu řízení, týká se to téměř všech jeho komponent, které se musí upravit, aby bylo možné reakce provést.
- Editace projektových dokumentů, kde se hlavně přidávají naplánované reakce do registru rizik, ale také se změní data v dokumentech týkajících se nákladů či rozvrhu.

Realizace reakcí na rizika

Proces, ve kterém se naplánované reakce implementují v předem dohodnutém pořadí. Vykonaává se po celou dobu běhu projektu.

Vstupy

- Plán řízení projektu s plánem řízení rizik identifikují člena týmu, co má na starosti dané riziko.
- Projektové dokumenty mezi jinými zprávy a registr rizik.
- OPA

Výstupy

- Požadavky na změny komponent plánu řízení projektu, příkladem můžou být směrný plán rozvrhu a nákladů.
- Editace projektových dokumentů např. dokumenty zaměřené na rizika nebo registry získaných znalostí.

Sledování rizik

V tomto procesu, který se vykonává po celou dobu prací na projektu, se kontroluje, zda se vykonávají naimplementované reakce, sledují se jednotlivá rizika a zachytávají se a analyzují dříve neznámé hrozby.

Vstupy

- Plán řízení projektu s plánem řízení rizik má zadáno, kdy a jak se kontrola provádí.
- Projektové dokumenty zaměřené na rizika.
- Údaje o výkonu práce obsahují informace o rizicích, které se v projektu vyskytly.
- Zprávy o výkonu práce, které se provedly.

Výstupy

- Informace o výkonu práce představují analyzovaná data o výkonu, zaměřené na to, jak dobře se projektu daří zvládat jednotlivé výskyty rizik.
- Požadavky na změny a opravy, hlavně pak směrných plánů nákladů a rozvrhu.
- Editace plánu řízení projektu dle potřeby.
- Editace projektových dokumentů rizik nebo různých předpokladů.
- Editace OPA o různé šablony z oblasti řízení rizik.

2.11 Řízení obstarávání v rámci projektu

V této znalostní oblasti se řídí výběr a nákup potřebných zdrojů od externích dodavatelů. Celkově se oblast hlavně zabývá uzavíráním potřebných dohod a smluv. Ačkoliv je manažer projektu většinou ten, který vyjednává jednotlivé zdroje, tak zpravidla nemá potřebné vzdělání či pravomoci k tomu, aby smlouvu podepsal a tím zavázal svou organizaci v plnění dané úmluvy. V oblasti se nacházejí tři následující procesy.

Plánování řízení obstarávání

Cílem procesu je rozhodnout a naplánovat co bude nutné obstarat z externích zdrojů pro potřeby projektu. Zároveň se také určuje způsob, místo a čas, kdy se bude produkt muset získat.

Vstupy

- Základající listina projektu obsahuje popis projektu a předem dohodnuté financování zdrojů.
- Obchodní dokumenty s projektovým záměrem a plánem řízení přínosů.
- Plán řízení projektu s komponentami zaměřenými na rozsah, kvalitu a zdroje.
- Projektové dokumenty s požadavky na zdroje a celkovými požadavky na projekt.

- EEF
- OPA

Výstupy

- Plán řízení obstarávání obsahuje seznam aktivit, co v průběhu provádění procesů řízení obstarávání budou provádět. Dále může obsahovat návody, jak řídit kontrakty a jejich právní normy, seznam preferovaných externích dodavatelů atd.
- Strategie obstarávání určují, jakým způsobem se produkt doručí a jak se za něho bude platit.
- Cenové nabídky jsou dokumenty na základě kterých lze porovnat nabídky různých externích dodavatelů.
- Prohlášení o práci pro každou zakázku určuje co má daný produkt umožňovat.
- Výběrová kritéria zdrojů, používá se, aby se vybral ten nevhodnější dodavatel.
- Rozhodnutí, zda vyrobit nebo koupit produkt se provádí pomocí analýzy, zda je samotný projektový tým schopen produkt dodat nebo se musí získat zvenčí.
- Nezávislé odhady ceny pro rozsáhlejší produkty. Může být zadáno externí firmě.
- Požadavky na změny týkajících se obstarávání produktů.
- Editace projektových dokumentů, které se hlavně týkají požadavků nebo možných rizik.
- Editace OPA s informacemi o kvalitních externích dodavatelích.

Řízení obstarávání

V tomto procesu se vyhodnocují nabídky od potenciálních dodavatelů. Následuje pak vybrání nejlepší nabídky a sepsání dohody s daným dodavatelem. Proces se periodicky opakuje.

Vstupy

- Plán řízení projektu, který mezi jinými obsahuje také plány řízení zaměřené na získávání zdrojů a možných rizik z toho plynoucích.
- Projektové dokumenty jako registr znalostí, rizik, nebo dokumenty týkajících se rozvrhu projektu či dokumenty popisující požadavky na produkty co se mají dodat.
- Dokumenty obstarávání, které mohou pomoci k vybrání vyhovujícího dodavatele.
- Nabídky dodavatelů určující, co jejich produkt je schopen provést.
- EEF
- OPA

Výstupy

- Vybraní dodavatelé, kteří nejlépe splňují požadovaná kritéria.
- Dohody a smlouvy s dodavateli, dokumentují způsoby dodání produktu, platby, podmínky atd.
- Požadavky na změny, které vzešly na základě procesu.
- Editace plánu řízení projektu a jeho komponent, informace vycházející z tohoto procesu.
- Editace projektových dokumentů týkajících se hlavně požadavků, zdrojů, rizik a zainteresovaných stran.
- Editace OPA, kde se můžou přidat informace a zkušenosti s důvěryhodnými dodavateli.

Kontrola obstarávání

Poslední proces této znalostní oblasti, ve kterém se monitorují, jak se dodržují dohodnuté kontrakty. Dále se zde také navrhuje požadované změny a zodpovídá i za uzavření celé dohody. Proces se provádí kdykoliv to projekt vyžaduje.

Vstupy

- Plán řízení projektu s plány na řízení požadavků, obstarávání, změn a rizik.
- Projektové dokumenty zaměřené na požadavky, rizika, kvalitu a podobně.
- Dohody a smlouvy s externími dodavateli produktů.
- Dokumenty obstarávání poskytují dodatečné informace k obstarávání.
- Schválené požadavky na změny podmínek v dohodách.
- Údaje o výkonu práce.
- EEF
- OPA

Výstupy

- Uzavřené dodávky, kde se dodavateli oznámí, že kontrakt byl naplněn a provádí se jeho dokončení.
- Informace o výkonu práce.
- Editace dokumentů obstarávání o dodatečné informace – schválené/neschválené změny atd.
- Požadavky na změny v projektu.
- Editace plánu projektu a jeho komponent k řízení rizik, řízení obstarávání a dále směrné plány rozvrhu a nákladů.

- Editace projektových dokumentů, hlavně pak dokumenty týkající se požadavků.
- Editace OPA s informacemi o rozvrhu plateb, s hodnocením dodavatelů atd.

2.12 Řízení zainteresovaných stran

Poslední znalostní oblast. Jejím cílem je určit všechny **zainteresované strany** (neboli všechny subjekty, které mohou mít vliv na projekt nebo být nějakým způsobem projektem ovlivněni). Dále se analyzují jejich očekávání od projektu, míru vlivu a tvoří se strategie, jak s nimi jednat a jak je začlenit do vývoje na projektu. Zajištění spokojenosti zainteresovaných stran by se měla považovat jako jeden z nejdůležitějších cílů projektu. V oblasti se nacházejí čtyři procesy.

Identifikace zainteresovaných stran

Jak již název napovídá v tomto procesu se identifikují všechny zainteresované strany. Zároveň se také analyzují jejich vlivy, zájmy a požadavky na projektu. Proces se periodicky opakuje a umožňuje projektovému týmu vybrat správné způsoby, jak zainteresované strany zapojit do procesu řízení projektu.

Vstupy

- Základající listina projektu obsahuje seznam nejdůležitějších zainteresovaných stran.
- Obchodní dokumenty, hlavně pak dokumenty projektového zájmu a plánu řízení přínosu mohou obsahovat informace o zainteresovaných stranách.
- Plán řízení projektu nemusí být při první identifikaci ještě zcela dokončen. V pozdějších fázích projektu, ale komponenty plánu řízení komunikace a plánu zapojení zainteresovaných stran, mohou pomoci v tomto procesu.
- Projektové dokumenty se využijí až v pozdějších fázích projektu, hlavně pak dokumenty správy změn, problémů nebo i dokumenty požadavků.
- Dohody a smlouvy týkající se projektu.
- EEF
- OPA

Výstupy

- Registr zainteresovaných stran může obsahovat jejich identifikaci, hodnocení či klasifikaci.
- Požadavky na změny, které vyplynuly z nově identifikovaných zainteresovaných stran.
- Editace plánu řízení, kde v pozdějších iteracích se mohou přidat, či změnit informace v plánech řízení komunikace, rizik, požadavků nebo plánu zapojení zainteresovaných stran.
- Editace projektových dokumentů rizik, předpokladů nebo problémů.

Plánování zapojení zainteresovaných stran

V tomto procesu je cílem zkoumat požadavky jednotlivých zainteresovaných stran. Dále se také analyzují jejich zájmy, vlivy a očekávání na projektu. To vše se pak použije k tomu, aby se vytvořila odpovídající strategie k zapojení zainteresovaných stran k činnostem na projektu.

Vstupy

- Zakládající listina projektu obsahuje informace o projektu.
- Plán řízení projektu s plány řízení rizik, zdrojů a komunikace.
- Projektové dokumenty s registry rizik, zainteresovaných stran, s dokumenty o předpokladech, problémech či změnách, použít by se měl i rozvrh projektu.
- Dohody a smlouvy se zainteresovanými stranami.
- EEF
- OPA

Výstupy

- Plán zapojení zainteresovaných stran, ve kterém se popisují strategie a aktivity potřebné k co nejlepšímu zapojení daného subjektu v projektu.

Řízení zapojení zainteresovaných stran

Další z procesu co se provádějí po celou dobu pracovních aktivit na projektu. V jeho průběhu se komunikuje se zainteresovanými stranami a ve spolupráci s nimi se řeší problémy na projektu. Tím se lze ujistit, že zainteresované strany jsou vždy plně informované o stavu projektu.

Vstupy

- Plán řízení projektu s komponentami zaměřenými na rizika, komunikaci, zainteresované strany a změny.
- Projektové dokumenty s registry znalostí a zainteresovaných stran nebo dokumenty správy změn a problémů.
- EEF
- OPA

Výstupy

- Požadavky na změny, které vzešly ze spolupráce se zainteresovanými stranami.
- Editace plánu řízení projektu pro lepší řízení komunikace a zapojení zainteresovaných stran.
- Editace projektových dokumentů, které se mohly využít i při vstupu.

Monitorování zapojení zainteresovaných stran

V posledním procesu oblasti řízení zainteresovaných stran se monitoruje jejich vztah s projektem a v případě potřeby se upravují strategie používané k podpoře jejich zapojení. Proces patří do skupiny těch, které se uskutečňují po celou dobu životního cyklu projektu.

Vstupy

- Plán řízení projektu a plány řízení zdrojů, komunikace a zainteresovaných stran.
- Projektové dokumenty zaměřené na rizika, zainteresované strany a komunikaci.
- Údaje o výkonu práce a stupni zapojení zainteresovaných stran.
- EEF
- OPA

Výstupy

- Informace o výkonu práce shrnují záznamy z dokumentu údajů o výkonu práce.
- Požadavky na změny zapojení zainteresovaných stran.
- Editace plánu řízení projektu týkajících se zdrojů, komunikace a zainteresovaných stran.
- Editace projektových dokumentů, hlavně pak registry rizik, zainteresovaných stran a znalostí.

Kapitola 3

Agilní řízení projektů

Jedním z problémů klasického řízení projektu jsou jeho reakce na změny v pozdějších fázích projektu. U tohoto typu řízení se často stává, že zákazník vidí až výsledný produkt a ten nemusí být podle jeho představ, následují pak vynucené změny a hrozí prodlení v nasazení produktu s možným překročením dohodnutého rozpočtu. Dalším problémem může být i to, že se nejdříve tvoří různé plány a návrhy, ale výsledný produkt se tvoří a dodává až později. Všechny tyto problémy se snaží řešit různé agilní metodiky. Pokud není určeno jinak, tak informace jsou převzaty z knihy *Agile Practice Guide*, která je také od firmy **PMI** [1].

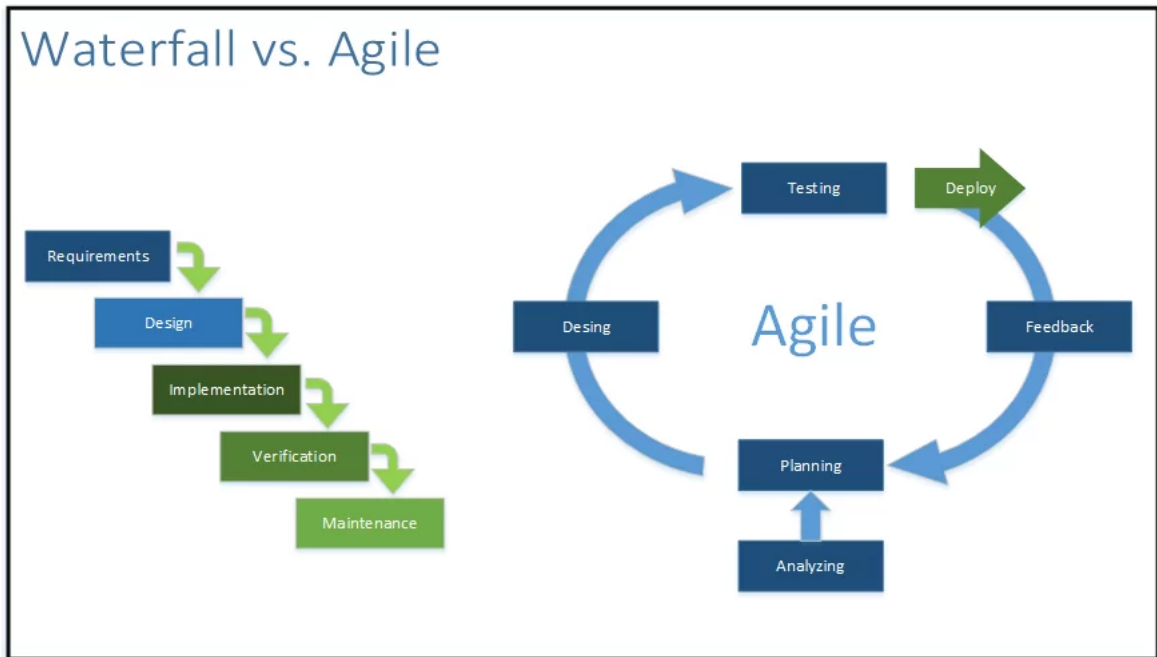
Agilní řízení projektů umožňuje pracovat a úspěšně dokončit projekty, které jsou charakteristické svou nejasností vycházející ze změn v požadavcích. Díky tomu lze také lépe reagovat na změny a tím i na nově se vyskytující rizika.

Agilní životní cyklus je kombinací iterativního, kde se vytvářejí prototypy, které se prezentují zainteresovaným stranám k posouzení, získané informace se použijí v dalších iteracích, a inkrementálního životního cyklu, kde se zákazníkovi v pravidelných cyklech dodávají jednotlivé části výsledného produktu. Díky tomuto přístupu se lze dříve odhalit nedorozumění v požadavcích. Příklad agilního řízení projektu v porovnání s vodopádovým modelem lze vidět na obrázku 3.1.

3.1 Agilní přístup ke znalostním oblastem

Znalostní oblasti, které lze využívat v klasickém řízení projektu, lze také v pozměněné formě využívat i v agilním řízení.

- **Řízení integrace projektu** je podobné jako v klasickém řízení. Jak bylo psáno v části 2.3, hlavním účastníkem je manažer projektu. Agilní řízení se v této znalostní oblasti více zaměřuje na zapojení členů týmu, kdy je jím přidělena větší zodpovědnost k plánování a dodávání výsledků. Úkolem manažera projektu je pak vytvořit odpovídající podmínky pro členy týmu a připravit je na lepší zvládnutí změn.
- **Řízení rozsahu prací projektu.** Agilní řízení projektu uznává, a dokonce i vítá skutečnost, že požadavky v projektu nejsou od začátku úplně známy a že se v průběhu řešení mohou měnit. To způsobuje, že při počátečním plánování nelze plně určit celkový rozsah projektu. Z toho důvodu se klade důraz na naplánování procesů, které se budou v průběhu projektu opakovat, jejichž cílem je postupně stanovit celkový rozsah aktivit na projektu.



Obrázek 3.1: Možný agilní vývoj v porovnání s klasickým vodopádovým modelem (zdroj převzato z [3])

- **Řízení časového plánu v rámci projektu.** Jak bylo již dříve zmíněno i v klasickém řízení projektu se používají jisté agilní metody k řízení časového plánu. U menších projektů a firem se osvědčilo provádění relativně krátkých cyklů, ve kterých se pracuje na části projektu. Na konci cyklu se pak zhodnotí dosavadní výsledky a v případě potřeby se přizpůsobí další cykly. Rozsáhlejší projekty a organizace, ale vyžadují alespoň částečné dlouhodobé časové plány (např. pro potřeby přerozdělení zdrojů). Pro ně se pak mohou použít hybridní metody spojující agilní a klasický přístup.
- **Řízení nákladů projektu** může být problematické, pokud ještě není znám celkový rozsah aktivit v projektu. Z důvodu těchto nejasností není možné vytvořit detailní plán celkových nákladů. Místo toho se vytvářejí nedetailní dlouhodobé plány, které představují pravděpodobné náklady a které lze kdykoliv přizpůsobit. Pokud jsou detailní plány nákladů potřeba, tak se vytvářejí pouze pro krátké časové úseky.
- **Řízení kvality v rámci projektu.** Jelikož se v agilním řízení projektu doručují jednotlivé části (iterace) produktu postupně, tak je možné již během vývoje kontrolovat kvalitu výsledků a tím lépe (a za méně nákladů) napravit případné nedostatky. Aby se kvalita práce na projektu udržela na konsistentní úrovni (případně i zlepšila), tak se periodicky kontroluje efektivita procesů týkajících se kvality. V případě, že se zjistí nedostatky, tak se uvažuje o změně či úplném vynechání procesu.
- **Řízení zdrojů v rámci projektu.** Projekty mívají často vysokou úroveň nejasností, může být proto problematické naplánovat, kdy a jaké zdroje (ať už fyzické nebo lidské) budou potřeba. Z tohoto důvodu je proto vhodné mít dohody a smlouvy s dodavateli, kteří jsou schopni velmi rychle potřebné zdroje zajistit. Je důležité se také zaměřit

na lidské zdroje projektu – členy týmu. Dle agilních principů mají na projektu pracovat efektivní, nejlépe samo se řídící týmy. Tyto spolupracující týmy, pak mají být základem k úspěšnému dokončení projektu.

- **Řízení komunikace v rámci projektu.** Jelikož agilní vývoj mnohem častěji vítá změny, tak se komunikace, ať už mezi členy týmu anebo se zainteresovanými stranami, stává jednou z nejdůležitějších složek agilního řízení projektu. Členové týmu by vždy měli mít svobodný přístup ke všem potřebným informacím a mělo by jim být umožněno se fyzicky účastnit týmových meetingů (tzv. kolokace členů týmu). Dále je také nutné více spolupracovat se zainteresovanými stranami.
- **Řízení rizik projektu.** Rozdělení projektu na menší části umožňuje posuzovat rizika v mnohem menším měřítku. Díky tomu se pak nemusí vytvářet zdlouhavé, dopředné identifikace a analýzy rizik celého projektu. Rizika se berou v úvahu již při výběru části projektu. V případě, že tým ví, že se jistá hrozba, na kterou není připraven, může v dané části vyskytnout, tak si vždy může vybrat jiný úsek projektu. Při tvorbě dané části se pak rizika identifikují, analyzují a v případě výskytu i řídí. Již hotové části projektu se pak ještě dodatečně opakovaně analyzují. Podporuje se také sdílení vědomostí a znalostí o rizicích mezi jednotlivými týmy. Rozdělení na části a jejich postupné doručování zákazníkovi umožňuje snížit riziko nedodržení kvality projektu, kterou zákazník požaduje. Problematickým rizikem v agilním řízení může být nedodržení rozpočtu, když, jak se již zmínilo v této kapitole v řízení nákladů projektu, neexistuje dopředný podrobný plán celkových nákladů.
- **Řízení obstarávání v rámci projektu.** V menších projektech může být vhodné navázat bližší spolupráci s dodavatelem produktu, případně ho začlenit do projektového týmu. Díky tomuto přístupu dodavatel i příjemce (v tomto případě firma co projekt řeší) sdílí rizika i výhody na projektu a mají větší motivaci ho úspěšně dokončit. U větších projektů může být potřeba použít kombinovaný přístup, když pro část zdrojů se použije adaptivní přístup a pro další klasický stabilní. Tuto kombinaci je ale nutné si zajistit smlouvou, kdy se adaptivní část může vyjednat v dodatcích k dohodě.
- **Řízení zainteresovaných stran.** Z důvodu toho, jak se projekty provádějí v agilním řízení, tak je zapojení zainteresovaných stran v mnohem větší míře než u klasického řízení. Často se pak stává, že členové týmu přímo komunikují se zainteresovanými stranami, aby od nich zjistili potřebnou zpětnou vazbu či podrobnosti k požadavkům. Díky tomuto přístupu se zmenšují rizika nespokojenosti zainteresovaných stran a zvětšuje pravděpodobnost úspěšného dokončení projektu.

Kapitola 4

Návrh nástroje pro podporu řízení rizik

Tato kapitola popisuje návrh nástroje pro řízení rizik. V první části je popsána specifikace požadavků na daný nástroj. Následuje pak návrh aplikace s use-case diagramem, schématem struktury databáze a návrhem několika obrazovek uživatelského rozhraní. Při popisu návrhu databáze se používá zkratka **CRUD**, jedná se o spojení anglických slov *create*, *read*, *update* a *delete*, představující základní operace nad daty v databázi. Do češtiny lze tyto slova přeložit jako **vytvořit**, **číst**, **editovat** a **smazat**.

4.1 Specifikace požadavků

V posledních letech se webové aplikace staly velice oblíbenou technologií. Je to způsobeno čím dál tím rychlejším internetovým připojením, ale také rozsáhlou kompatibilitou s různými operačními systémy, které běží na často velmi rozdílných zařízeních. Z tohoto důvodu by výsledný nástroj měl být webovou aplikací.

Zaměření aplikace by mělo být na plánování řízení, identifikaci a kvalitativní analýzu rizik. Nástroj by měl umožňovat zobrazovat a upravovat registr rizik, ve kterém lze vyhledávat. Dle 2.10.2 lze pomocí dotazníku identifikovat a analyzovat rizika na projektu, bude proto vhodné tyto dotazníky a jejich otázky v aplikaci vytvářet a uchovávat. Příklad možného dotazníku lze nalézt v příloze C. Jedna otázka se může vyskytovat ve vícero dotaznicích. K identifikaci rizik lze použít i SWOT analýza jejímž výstupem je SWOT tabulka, aplikace by proto měla umožňovat její vytvoření. Uživatelům by mělo být také dovoleno zobrazovat rizika v matici pravděpodobnosti a dopadů.

Co se týče uživatelů, tak těm by mělo být dovoleno se přihlašovat a odhlašovat, přičemž aplikace by si měla přihlášení na daném počítači pamatovat a v případě neaktivity uživatele by po jisté době mělo dojít k automatickému odhlášení. Mělo by existovat více uživatelských rolí, přičemž obyčejnému uživateli by mělo být dovoleno zobrazovat pouze projekty, ke kterým je přihlášen. Manažeru projektu by nástroj měl umožňovat spravovat své projekty, dále i přiřazovat a odhlašovat řešitele projektu. Ke každému řešiteli by mělo být možné přidat rizika, která má na starosti. Výsledná aplikace by měla být jednoduchá na používání s moderním a responzivním designem.

4.2 Diagram případů užití

Na základě specifikace požadavků byl navrhnout diagram případu užití, který je k vidění na obrázku 4.1. V systému se budou nacházet tři role: **člen týmu řešitelů**, **projektový manažér** (může být také manažerem řízení rizik) a **projektový administrátor**. Role jsou propojené tak, aby role s největšími pravomocemi mohly provádět i činnosti rolí s nižším oprávněním neboli administrátor může provádět případy užití člena týmu.

Projektový administrátor

Je to uživatel systému, který má na starost administrativní úkony na projektu. Jedná se tedy o tyto případy užití:

- **Vytvořit projekt**, ve kterém se založí projekt, přidají se mu potřebné atributy a zároveň se k němu také přiřadí jeho manažer a ostatní členové řešitelského týmu.
- **Smazat projekt**, odstraní se veškerá projektová data – přiřazení uživatelů, swot analýza, přidělení dotazníku, odpovědi na dotazník v daném projektu a přidělení rizik, včetně dat určených k danému riziku v daném projektu.
- **Správa uživatelů**, obsahuje všechny činnosti nad účty všech uživatelů (jak se zmiňuje v poznámce, jedná se o CRUD akce).
- **Vytvořit dotazník**, v tomto užití bude projektový administrátor tvořit dotazníky z předem vytvořených otázek.
- **Vytvořit otázku**, kde se vytvoří otázka a zároveň se vyberou i možné odpovědi. Jedna otázka se pak bude moct vyskytovat ve více dotaznících.

Projektový manažer

Může se také označovat jako manažer řízení rizik nebo jako manažer rizik. V jeho kompetencích je řízení projektu a členů, kteří na něm pracují. Může upravovat pouze projekty, ke kterým je přiřazen.

- **Přidat dotazník k projektu**, kde se předem vytvořený dotazník může přiřadit k projektu. Jeden dotazník se může využít i v jiných projektech.
- **Odebrat dotazník z projektu**, odebere dotazník z projektu a smaže údaje jak uživatelé na tento dotazník v tomto projektu odpovídali.
- **Správa řešitelů projektu**, zpřístupní funkce díky, kterým bude manažer schopen členy týmu ze svého projektu odhlašovat či přihlašovat nové řešitelé.
- **Aktivace a deaktivace projektu**, kde manažer aktivuje či deaktivuje projekt. Deaktivovaný projekt není možné dále upravovat (musí se znovu aktivovat).

Člen týmu

Představuje klasického řadového člena řešitelského týmu. Tato role má v nástroji nejméně oprávnění. Jsou mu přiřazené následující případy užití:



Často se zde objevuje správa určitého objektu. Pokud to není explicitně zmíněno v jiném případě užití, tak se jedná o CRUD operaci, neboli vytvoření, čtení, úpravu a smazání.

Obrázek 4.1: Diagram případů užití [zdroj vlastní]

- **Vyplnění dotazníku**, kde člen týmu bude moct odpovědět na dotazník týkající se projektu, ke kterému je přiřazen.
- **Správa registru rizik** umožní vyhledávat a filtrovat výsledky centrálním registru rizik dané společnosti.
- **Správa rizik v projektu**, v tomto případě užití bude moct člen týmu vykonávat celkovou správu rizik na daném projektu.
- **Správa SWOT tabulky**, kde člen týmu (ale také i také i ostatní role s většími právy) budou moci vytvářet, číst, upravovat či mazat SWOT tabulky, které obsahují výstup ze SWOT analýzy.
- **Zobrazit rizika v matici**, jde o grafickou prezentaci rizik v matici pravděpodobností a dopadů. Hodnoty pravděpodobnosti výskytu a dopadu rizika se získají z jeho atributů.
- **Správa projektu** umožní celkovou správu všech atributů projektu.
- **Zobrazit statistiku dotazníku** umožní zobrazovat, jak ostatní uživatelé odpovídali na dotazník, který byl k projektu přiřazen.

4.3 Schéma databáze

Jelikož se primárně předpokládá, že výsledný nástroj bude webovou aplikací, tak by bylo vhodné, aby se data ukládala do databáze. Návrh struktury databáze lze vidět na obrázku 4.2. Tabulky, které mají ve svém názvu podtržítka představují vazební tabulky.

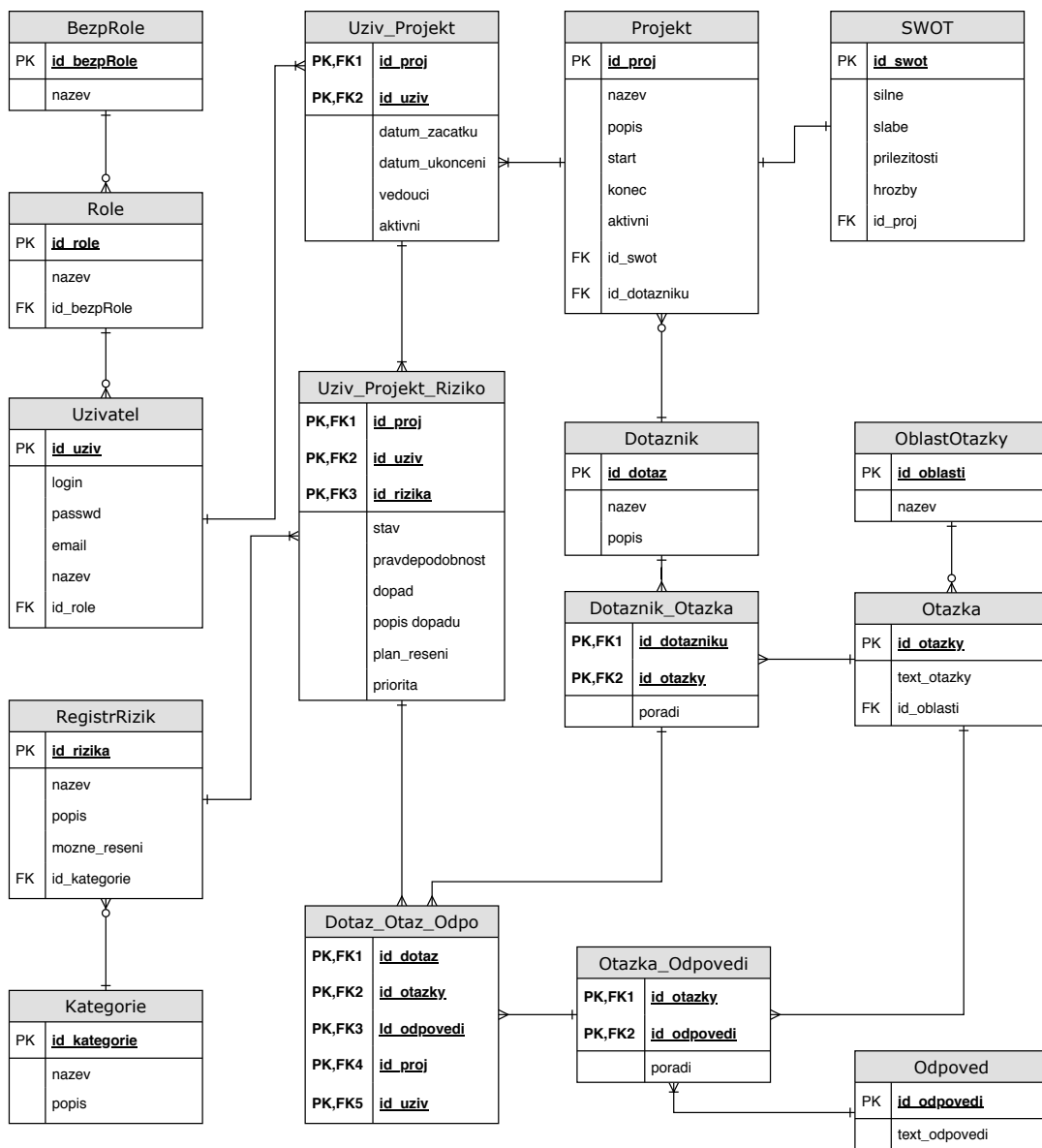
Uživatel je tabulka, která reprezentuje uživatele nástroje, půjde tedy buď o člena týmu, projektového manažera nebo projektového administrátora. To, jakou roli uživatel bude mít určuje cizí klíč z tabulky **Role**. Tabulka dále obsahuje přihlašovací informace daného uživatele. Dále také i jeho pracovní email a jméno. Role umožňuje přidávat do systému další druhy rolí, jejím základem je ovšem **BezpRole**, která určuje tři základní bezpečnostní role - USER, MANGER a ADMIN, které se využívají pro určování přístupu ke zdrojům. Záznamy v této tabulce nelze měnit.

Projekt určuje záznamy daného projektu. Ukládá se jeho název, základní popis, začátek prací a předpokládaný konec. Na daný projekt je pak navázaná tabulka **SWOT**, která obsahuje informace ze stejnojmenné analýzy. Tabulka projektu také obsahuje odkaz na dotazník, který je k němu přiřazen.

Aby bylo možné sledovat, kdo všechno je k danému projektu přiřazen, tak mezi tabulkami **Uživatel** a **Projekt** je vazební tabulka **Uziv_Projekt**, která dále umožňuje určit, kdy daný uživatel začal na projektu pracovat a kdy byl z projektu odhlášen a také zda byl jeho manažerem či zůstal na projektu přihlášen až do jeho dokončení.

Tabulka **RegistrRizik** představuje centrální registr zjištěných rizik. V tomto registru se uchovávají všechna unikátní rizika, která se kdy v dané organizaci zjistila. Pomocí něho se může urychlit identifikace rizik na aktuálním projektu. Obsahuje název rizika, jeho popis, možné obecné řešení a cizí klíč do tabulky **Kategorie**, která reprezentuje oblast, do které riziko spadá.

Vazební tabulka **Uziv_Projekt_Riziko** pak spojuje tabulky **Uživatel**, **Projekt** a **RegistrRizika**. Umožňuje určit k jakému projektu dané riziko patří a také kdo je jeho vlastníkem (neboli osobou odpovědnou za plnou identifikaci a analýzu rizika). Mezi další atributy



Obrázek 4.2: Schéma struktury databáze [zdroj vlastní]

patří pravděpodobnost jeho výskytu, míru a popis jeho dopadů, plán řešení, pokud se riziko vyskytne, jeho prioritu (může také označovat závažnost) a stav jeho identifikace a analýzy.

Pro lepší identifikaci rizik a jejich pozdější analýzu lze použít dotazníky. Tabulka **Dotazník** je pak jejich reprezentací. Jeho atributy jsou název a krátký popis. Dotazník se může vyskytovat samostatně, případně může být přiřazen k více projektům.

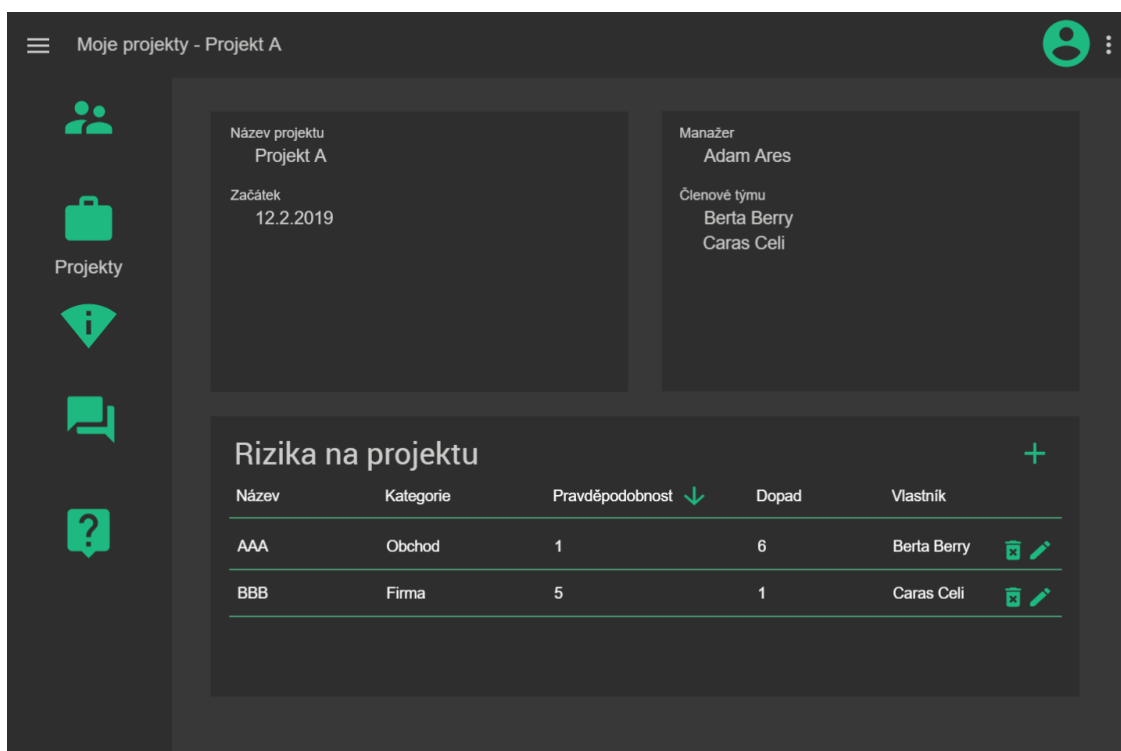
Otazka je tabulka, která obsahuje text potřebné otázky včetně i odkazu na oblast, které se otázka týká (tabulka **OblastOtazky**). Oblastí v tomto případě již lze rozumět jako skutečné oblasti např. procesní, obchodní, technologická, rizika spojená se zainteresovanými stranami atd.

Vytvořené otázky budou vždy uzavřené, při její tvorbě lze tedy využít již předem vytvořené možné odpovědi z tabulky **Odpoved**. To, jaké bude mít otázka možné odpovědi se uchovává ve vazební tabulce **Otazka_Odpovedi**, která zároveň obsahuje i atribut, který určuje pořadí dané odpovědi v otázce.

Vazební tabulka **Dotaznik_Otazka**, pak uchovává informace o tom, jaké otázky se v dotazníku vyskytují, přičemž atribut poradi určuje podobně jako v tabulce **Otazka_Odpoved**, pořadí dané otázky v dotazníku.

Celkové spojení uživatelů, projektů, dotazníků, otázek a odpovědí, pak symbolizuje vazební tabulka **Dotaz_Otaz_Odpo**. V této tabulce se uchovávají informace o tom, jak daný uživatel v daném dotazníku na otázku odpovídal (jakou odpověď z nabízených možných odpovědí vybral).

4.4 Návrh uživatelského rozhraní



Obrázek 4.3: Návrh karty projektu se zjištěnými riziky [zdroj vlastní]

Jak již bylo zmíněno dříve, výsledný nástroj má mít moderní vzhled s responzivním designem. Při studium konceptů návrhů moderních webových stránek jsem našla informace o Material Design od společnosti Google. Jedná se o systém pravidel, návodů, nástrojů a grafických komponent, které podporují návrh a vytváření moderních grafických uživatelských rozhraní. Hlavní stránka Material Design se pak dále zmiňuje, že se jedná o vizuální jazyk spojující návrh s inovativními myšlenkami z oblastí technologie a vědy, jehož hlavní inspirace se nachází v objektech reálného světa a jak tyto objekty reagují na přítomnost světla a stínu [8].

Material Design dále nabízí i několik studií, jejichž cílem je prezentovat možnosti a flexibilitu dostupných nástrojů a komponent. Návrh grafického uživatelského rozhraní této diplomové práce byl z velké části inspirován studiemi Rally [9] a Reply [10].

Ke tvorbě zjednodušeného možného návrhu uživatelského rozhraní několika obrazovek byl použit online nástroj Fluid UI [7], který při tvorbě návrhu umožňuje použití prvků ve stylu Material Design. Výsledný návrh obrazovky správy projektu lze vidět na obrázku 4.3.

Dále byly vytvořené i zjednodušené návrhy obrazovek přihlášení do systému a správy dotazníků, které lze nalézt v příloze A.

Kapitola 5

Realizace a implementace

V této kapitole se popisuje implementace požadovaného nástroje a představuje se výsledná realizace. Ještě před začátkem implementace bylo nutné zvolit framework, v českém překladu známý také jako aplikační rámec, který by odpovídal požadavkům návrhu, a to možnosti použití programovacího jazyka Java a umožnění odděleného vývoje klientské i serverové části. Dále by měl být jednoduchý na konfiguraci, používání a ideálně také by měl poskytovat určitý stupeň zabezpečení přístupu.

První část této kapitoly se zaměřuje na porovnání několika frameworků a knihoven pro naprogramování serverové i klientské části. Dále se pak blíže popisují zvolené programovací nástroje, a to je pak následováno popisem implementace a představení důležitých částí vytvořeného nástroje.

5.1 Vybrané aplikační rámce serverové části

Jak se již dříve zmínilo, při výběru aplikačního rámce se zaměřilo na ty z programovacího jazyka Java. Dle článků [15] a [14] se vybraly čtyři nejpoužívanější frameworky, které umožňují vytvářet serverovou část aplikace.

5.1.1 Spring

Spring framework (nebo některé z jeho zjednodušených součástí) patří již po mnoho let na vrchol nejpoužívanějších aplikačních rozhraní jazyka Java. Lze ho použít ke full-stack řešení projektů, to znamená, že v něm lze naprogramovat celou aplikaci od databáze až po grafické prvky klienta. Součástí frameworku je i Spring Security, který umožňuje snadnou implementaci autentizace i autorizace pomocí již existujících knihovnických funkcí. Nevýhodou Spring frameworku může být jeho pomalejší učící křivka, případně zmatenost, kdy nějakou funkcionalitu lze naprogramovat více způsoby.

5.1.2 Struts

Struts framework je velice podobný Spring aplikačnímu rámci. Také využívá konceptu Model-View-Controller a taktéž se více hodí pro rozsáhlejší projekty. Struts se vyskytuje ve více verzích – Struts 1 a Struts 2, přičemž více se využívá verze 2, která je rozšířená o framework OpenSymphony. Hlavní výhodou může být aktivní dokumentace či centralizovaný systém konfigurace.

5.1.3 Play

Play je jednoduchý framework, který umožňuje vytvářet nástroje, které celkově mají malou velikost a ke funkčnosti potřebují pouze minimum zdrojů, čímž se hodí i pro vývoj mobilních aplikací. Jádro aplikačního rámce je napsáno v jazyce Scala, ale umožňuje i podporu jazyka Java. Mezi výhody Play lze počítat plnou podporu NoSQL či neblokující vstup/výstupní operace. Tento framework používají společnosti EA, Walmart, Samsung a LinkedIn.

5.1.4 DropWizard

Cílem DropWizard frameworku je umožnit vývojářům rychlou implementaci a nasazení výsledné aplikace. Velice se hodí k implementaci REST aplikací či k tvorbě mikroslužeb. Má také vestavěnou podporu validace dat a zabezpečení zdrojů pomocí autentizace a autorizace.

5.2 Vybrané aplikační rámce klientské části

Při vyhledávání možných frameworků, pomocí kterých bych mohla vytvořit klientskou část aplikace, jsem rychle objevila, že téměř všechny články řadí následující aplikační rámce na nejvyšší místa v oblíbenosti.

5.2.1 Angular

Angular je framework založený na TypeScriptu, který je nadstavbou jazyka JavaScript a přidává do něho statickou typovou kontrolu. Vychází z aplikačního rámce Angular.js, ale není s ním zpětně kompatibilní a je vyvíjen společností Google. Jednou z výhod tohoto frameworku je, že má mnoho funkčnosti již vestavěných v sobě. Nevýhodou pak může být jeho strmější učicí křivka nebo složitější na porozumění dokumentace.

5.2.2 React

React je v současnosti nejoblíbenější „framework“ pro tvorbu webových stránek. Je vyvíjen týmem ve společnosti Facebook. Často se mylně označuje jako framework, ale ve skutečnosti se jedná o rozsáhlejší JavaScript knihovnu a z tohoto důvodu musí vývojáři častěji doinstalovat dodatečné podpůrné knihovny. Výhodou Reactu je, že změna stavu jedné komponenty neovlivňuje jiné komponenty, nevýhodou pak může být, že z důvodu vysoké oblíbenosti se knihovna velmi rychle rozvíjí a vývojář tak může ztratit přehled o všech změnách či vylepšeních.

5.2.3 Vue.js

Vue framework vyniká svojí jednoduchostí a malou velikostí výsledných souborů. Lze využít i ke tvorbě mobilních aplikací. Vue se v některých případech používá i jako knihovna. Další výhodou je i v jeho rozsáhlé dokumentaci. Nevýhodou může být, že komunita okolo tohoto aplikačního rámce není tak rozsáhlá, jak v předcházejících dvou.

5.3 Zvolené programovací nástroje

Na základě doporučení, jejich popularitě, počtu návodů a chuti se naučit novým technologiím, které bych mohla využít v zaměstnání, jsem zvolila kombinaci **Spring Boot** [17], který je rozšířením Spring frameworku, pro serverovou část a **React** [4] pro klientskou část.

5.3.1 Spring a Spring Boot

Jedním z hlavních cílů Spring frameworku bylo poskytnout vývojářům nástroj pro zjednodušení a urychlení vývoje, jak jsem ale již výše zmínila, tento aplikační rámec získal postupem let mnoho funkcí rozložených do mnoha modulů, to ale způsobilo, že už i založení projektů bylo zdlouhavé a muselo se speciálně dbát na správnou konfiguraci jednotlivých částí.

A právě to byl jeden z důvodů, proč se vytvořilo rozšíření pro Spring framework – Spring Boot. Tento samostatný modul, který ale umožňuje využívat funkčnosti Spring aplikačního rámce, má za cíl ještě více urychlit vývoj aplikací a to tím, že většinu konfigurace provede automaticky nebo pomocí anotací. Také zjednodušuje přidávání nových modulů a závislostí, kdy stačí do konfiguračního souboru vložit pouze tzn. **starter** dané závislosti a Spring Boot se o další nastavení postará sám. Názvy starterů některých modulů jsou: **spring-boot-starter-web**, **spring-boot-starter-security**, **spring-boot-starter-data-jpa**, atd. V základu se také nachází již vestavěný server pro rychlé testování či nasazení.

ORM, JPA a Hibernate

ORM neboli *Object-relational mapping*, v překladu **Objektově relační mapování**, je programovací technika, která umožňuje pracovat s databází jako s klasickými třídami a objekty v Javě (nebo jejich ekvivalenty v ostatních objektově orientovaných jazycích). Dále také ORM poskytuje jistou úroveň odstínění druhu databáze od samotného programu, díky čemuž vývojář nemusí plně znát jazyk jakým se s danou databází komunikuje a v případě potřeby také ulehčuje přechod z jednoho typu databáze na druhý. Nevýhodou této techniky pak může být skutečnost, že vývojář plně „nevidí“ do procesů, které se na pozadí provádějí a také, že komunikace např. přímo v SQL může být rychlejší a efektivnější než pomocí knihovny, která ORM implementuje.

JPA je součástí standartu Java a byla známá pod názvem *Java Persistence API*. Když ale Java Enterprise Edition (Java EE) změnila název na Jakarta EE, tak bylo JPA přejmenováno na *Jakarta Persistence*. Jedná se o specifikaci ORM pro Javu, ve kterém se specifikují rozhraní a anotace používané pro mapování objektů a vztahů mezi objekty do databáze. Mezi nejznámější anotace JPA patří: **@Entity** a **@Table** pro definici tabulek, **@Id** pro identifikaci primárního klíče **@OneToOne**, **@OneToMany**, **@ManyToOne**, **@ManyToMany** pro provázání tabulek pomocí cizích klíčů.

Hibernate, známý také jako **Hibernate ORM**, je framework, který implementuje ORM pro jazyk Java, zároveň také implementuje specifikaci JPA. Mapování objektů, lze provést pomocí XML souborů nebo pomocí anotací. Hibernate také poskytuje modul pro validaci dat, podporu pro persistenci dat v datových skladištích NoSQL a mnohem více [11].

Spring Boot používá Hibernate jako výchozí implementaci JPA. Výchozí data se do tabulek načtou ze SQL skriptu `import.sql`, který Spring Boot automaticky spustí po vytvoření databáze. Zdrojový kód 5.1 zobrazuje příklad třídy s JPA anotacemi, ze kterých Hibernate vytvoří odpovídající tabulku (zdrojový kód 5.2 představuje kód, který vytvoří Hibernate).

Starter pro všechny závislosti potřebné k používání Hibernate se jmenuje `spring-boot-starter-data-jpa`.

```
@Entity(name = "user")
@Table(name = "User")
public class User {

    @Id
    @GeneratedValue(strategy = IDENTITY)
    @Column(name = "id", unique = true, nullable = false)
    private long id;

    @Column(name = "login")
    private String login;

    @Column(name = "passwd")
    private String passwd;

    @Column(name = "email")
    private String email;

    @ManyToOne
    @JoinColumn(name = "fk_role", nullable = false)
    private Role role;
}
```

Zdrojový kód 5.1: Třída s anotacemi [zdroj vlastní]

```
create table user (
  id bigint generated by default as identity,
  email varchar(255),
  login varchar(255),
  passwd varchar(255),
  fk_role bigint not null,
  primary key (id)
)

alter table user
  add constraint FKp61v76usu5tblw5in3jlpe8rc
  foreign key (fk_role)
  references role
```

Zdrojový kód 5.2: SQL kód vygenerovaný Hibernate [zdroj vlastní]

MVC a REST

MVC je zkratka pro spojení slov *Model-View-Controller* a jedná se o softwarovou architekturu, která doporučuje rozdělení aplikace do tří modulů, které jsou na sobě navzájem nezávislé. Model, lze chápat jako datový model či datová vrstva, jejím úkolem je uchovávat data aplikace. View neboli prezenční vrstva zobrazuje uživateli data, které ji dle potřeby zpřístupní datová vrstva. Controller lze přeložit jako řídicí vrstva, je to vrstva nacházející se mezi datovou a prezenční vrstvou, jejím úkolem je řídit aplikaci, reagovat na požadavky pocházející z prezenční vrstvy a o případných změnách v datech informovat datovou vrstvu.

REST, neboli *Representational State Transfer*, popisuje principy pro jednoduchý přenos dat (zdrojů aplikace) mezi klientem a severem. V REST specifikaci každý zdroj uložený na serveru má své jedinečné URI (Uniform Resource Identifier, jednotný identifikátor zdroje), pomocí kterého lze k němu přistupovat. S každým zdrojem pak lze provádět čtyři následující operace – vytvoření, čtení, editace a smazání, což jsou přesně CRUD operace. Ke komunikaci s REST aplikací lze použít internetový protokol HTTP, kde se využijí jeho metody GET, POST, PUT, DELETE, atd. Data mezi klientem a serverem se pak mohou posílat ve formátu XML, XHTML či JSON.

K aktivaci MVC ve Spring Boot aplikaci se používá starter `spring-boot-starter-web`, který ze Spring frameworku naimportuje modul Spring MVC, pomocí kterého lze vytvářet tzv. RESTful aplikace (neboli aplikace implementující REST přístup ke zdrojům. Díky web starteru se také zpřístupní další požadované technologie pro vývoj webové aplikace: vestavěný Tomcat server, knihovnu Jackson pro práci s daty ve formátu JSON, validátor dat a další.

Spring Security

Pro zabezpečení přístupu k datům aplikace používá Spring Boot modul **Spring Security**, který se importuje starterem `spring-boot-starter-security`. Spring Security poskytuje jak autentizaci, tak i autorizaci (ověření identity uživatele a ověření, zda daný uživatel má povolení danou akci provádět). Jednotlivé požadavky přicházející na server musí ještě před tím, než dojdou ke Controlleru, projít řadou filtrů nastavených pomocí Spring Security. Pokud požadavek některým z filtrů neprojde, tak je odmítnut, typicky se pak klientské straně přepoše negativní odpověď se stavovým kódem 401, který se v HTTP protokolu používá ke označení, že uživatel není autentizován.

Spring Security poskytuje i další funkčnost jako zapamatování přihlášení či zašifrování hesla v databázi a při přenosu. Řeší také známé slabiny, které se při komunikaci mezi klientem a serverem mohou vyskytnout. Příkladem může být **CRSF** neboli *cross-site request forgery*, kdy útočník zneužije spojení mezi aktuálně přihlášeným uživatelem a serverem. Pomocí tohoto spojení pak může odeslat požadavky, o kterých si server myslí, že se jedná o legitimní požadavky uživatele. Problém je možné řešit tak, že po autentizaci klienta si obě strany synchronizují tzn. CRSF token, který klientská strana přidává ke požadavkům a zabezpečení serveru tento token ověřuje. Díky tomuto zabezpečení pak server přesně zná původ požadavku.

5.3.2 React

Jak jsem již zmínila, React je JavaScript knihovna pro tvorbu interaktivních uživatelských rozhraní. React umožňuje vytvářet samostatné znovupoužitelné komponenty (prvky uživa-

telského rozhraní), které jsou schopné spravovat svůj vnitřní stav. Jednoduché komponenty pak lze skládat do složitějších. React využívá konceptu **virtuálního DOM** a **JSX**.

Virtuální DOM (*Document Object Model*, **objektový model dokumentu**) staví na konceptu uložení virtuální reprezentace modelu v paměti a jeho následné synchronizace se skutečným DOM. React ho využívá k tomu, aby DOM vždy odpovídal stavu jednotlivých komponent [5].

JSX je syntaktické rozšíření JavaScriptu a slouží k definování vzhledu komponent v Reactu. Díky JSX může být definice vzhledu a funkce zodpovědné za logiku komponenty pohromadě v jednom souboru. Použitím JSX se také zabraňuje útokům typu **XSS** (*Cross-site scripting*), kdy se útočník snaží přes nějaký neošetřený vstup nahrát na stránku nežádoucí skript. Zdrojový kód 5.3 ukazuje použití JSX, právě prvek React DOM zajišťuje, že útok XSS nelze provést [6].

```
const name = 'Josh Perez';
const element = <h1>Hello, {name}</h1>;

ReactDOM.render(
  element,
  document.getElementById('root')
);
```

Zdrojový kód 5.3: JSX, vykreslení prvku H1 do HTML komponenty s id root [zdroj vlastní]

5.3.3 Další důležité knihovny

V sekci 5.2.2 jsem se zmiňovala, že k Reactu se často musí doinstalovat další knihovny. V této sekci popisuji další nejdůležitější knihovny používané v této diplomové práci. Celý soupis použitých knihoven pro tvorbu uživatelského rozhraní lze nalézt v příloze E.

Material-UI

Material-UI je nejpopulárnější knihovnou pro tvorbu uživatelského rozhraní vytvořenou pro React. Obsahuje mnoho již předem vytvořených prvků, které splňují požadavky kladené specifikací Material Design. Jedná se o alternativu ke knihovně Bootstrap. Prakticky celý nástroj využívá komponenty z této knihovny. Taktéž všechny použité ikony jsou převzaty z Material-UI icons.

Material-table

Knihovna material-table rozšiřuje komponentu tabulky z Material-UI a přidává do ní dodatečnou funkčnost. Poskytuje jednodušší přidávání, editaci a mazání záznamů. Dále také umožňuje vyhledávání v záznamech, filtraci a třídění dat, zobrazování panelu s detaily, export do požadovaných záznamů a mnohem více. Výchozí chování jednotlivých funkcí lze předefinovat dle potřeb uživatele.

Nivo

Nivo je knihovna grafových komponent pro React. Obsahuje značné množství nepoužívanějších druhů grafů, které lze v případě potřeby upravovat. Všechny grafy jsou interaktivní,

obsahují i legendy a každý z nich má i verzi, která je responzivní ke změně velikosti okna. Nivo má na svých stránkách i velice zajímavou interaktivní dokumentaci, ve které lze měnit parametry atributů daného grafu, které se okamžitě promítají do vykreslené komponenty.

Redux a React-Redux

V Reactu funguje předávání informace z rodičovské komponenty na dceřinou. Problém se může vyskytnout, pokud stejnou informaci potřebují dvě odlišné komponenty, které nejsou ve stejné větvi, případně ty, co jsou ve stejné větvi, ale jedná je od druhé až příliš moc vzdálená a pro ostatní komponenty je daná informace nepotřebná.

Redux je nástroj pro správu stavu aplikaci, který daný stav ukládá do speciálního objektu `store`, ze kterého mohou všichni číst. V rámci Reduxu se pak hovoří ještě o `actions` a `reducers`. `Actions` lze považovat za události, které říkají, že se má provést nějaká akce. Každá akce je definovaná svým typem a obsahem (`payload`) přenášeným z aplikace do `store`. `Reducers` jsou pak funkce, které dle dané `action` provedou nějakou modifikaci s `data`, co se nacházejí ve `store` a vrátí nový stav.

React-Redux byl vytvořen pro spojení Reactu a Reduxu. Přidává do aplikace komponenty a funkce díky kterým mohou všechny komponenty aplikace přistupovat ke stavům ve `store` a také vyvolávat akce, které `store` modifikují.

Formik a Yup

Formik je pomocná knihovna pro správu formulářů. Jejím hlavním úkolem je získávání dat z/do stavů formuláře, validace zadaných hodnot, zobrazování chybových zpráv a zpracování potvrzení formuláře.

Yup je knihovna pro validaci zadaných hodnot. Umožňuje vytvořit validační schéma pro různé typy objektů a hodnoty. Formik používá knihovnu Yup jako výchozí validátor.

Axios

Axios je jednoduchý HTTP klient pro komunikaci s REST API. Lze ho používat s Reactem nebo s libovolně jiným frameworkem. Axios je velice podobné `fetch API` z JavaScriptu, které umožňuje pohodlnější práci s AJAX (*Asynchronous JavaScript and XML*) dotazy a také vrací tzn. `Promise` objekt, který slouží jako příslib, že jednou v něm budou vrácena přijatá data.

React Router

React Router je jednoduchá knihovna, pomocí které lze řešit směrování na různá URL aplikace.

5.3.4 Podpůrné programovací nástroje

Při tvorbě implementace klientské i serverové částí se použilo IDE IntelliJ IDEA od společnosti JetBrains. Při doladování vstupů a výstupů API se pak použil program Postman, který umožňuje tvořit HTTP dotazy, poslat je na server a následně zobrazí odpověď, kterou vyslal server. Jako databáze se používá H2, která v Javě umožňuje vytvářet databáze v paměti počítače.

5.4 Implementace nástroje

Aplikace je rozdělena na dvě části, klientskou a serverovou. Klientská část je naprogramovaná pomocí JavaScript knihovny React, serverová pak využívá Java aplikační rámec Spring Boot, kde nejdůležitějšími moduly jsou Spring Data JPA, Spring MVC a Spring Security.

5.4.1 Server

Server se využívá jako RESTful API (*Application Programming Interface*), neboli se využívá jako webová služba, která reaguje na HTTP požadavky GET, POST, DELETE.

Metoda GET se používá pro získávání zdrojů, ať už se jedná o jedinou položku nebo seznam několika z nich. POST se využívá ve dvou případech, a to, pokud se nahrávají nová data anebo se data modifikují. DELETE slouží pro smazání požadované položky.

Server je rozdělený do pěti částí – **security**, **web**, **service**, **repository** a **domain**. Každý validní požadavek, který na server přijde, musí postupně projít každou z částí.

Domain

V adresáři domain se nacházejí soubory s definicemi tříd, které byly vytvořeny dle návrhu databáze, který byl popsán v části 4.3. Ke třídám jsou pak přidány potřebné anotace, podle kterých Hibernate vytvoří odpovídající tabulky v databázi. Nejpoužívanější anotace pro definici tabulek jsou:

- `@Entity` a `@Table` specifikují, že daná třída je entita, kterou lze namapovat na tabulku, s `@Table(name = "Jmeno")` se také udává název tabulky.
- `@Id` a `@GeneratedValue(strategy = IDENTITY)` se používají pro označení, který atribut třídy má být primárním klíčem entity a také definují strategii pro automatické generování hodnoty klíče.
- `@Column(name = "jmeno", nullable = false, columnDefinition="text")` definuje atribut třídy, který se má ukládat do databáze, vlastnost `name` udává název pod jakým se má vytvořit, `nullable` označuje zda hodnota může nabývat null hodnot a `columnDefinition` pak určuje typ.

Každý vztah mezi tabulkami je dvousměrný, tzn. že spojené třídy mají na sebe navzájem odkazy. V databázi ovšem bude mít jenom jedna z nich cizí klíč druhé tabulky. Toto spojení zajišťují anotace `@OneToMany` a `@ManyToOne` s `@JoinColumn`. Zdrojový kód 5.4 zobrazuje použití `@OneToMany` ve třídě `Role` nad atributem, který uchovává uživatele s danou rolí, `mappedBy` pak říká, že vlastníkem daného sloupce je jiná entita – tabulka nebude obsahovat cizí klíč. Použití inverzní anotace `@ManyToOne` s `@JoinColumn` ukazuje zdrojový kód 5.5, kde jsou použity nad atributem uchovávajícím odkaz na roli uživatele. Vlastnost `name` udává název sloupce s cizím klíčem.

```
@OneToMany(mappedBy = "role", cascade = CascadeType.ALL,  
orphanRemoval = true)  
private List<Uzivatel> uzivatele = new ArrayList<Uzivatel>();
```

Zdrojový kód 5.4: Anotace pro vytvoření vztahu na straně tabulky `Role` [zdroj vlastní]


```
@ManyToOne
@JoinColumn(name = "fk_role", nullable = false)
private Role role;
```

Zdrojový kód 5.5: Anotace pro vytvoření vztahu na straně tabulky Uživatel [zdroj vlastní]

V domain se také nacházejí další dva adresáře. Adresář id obsahuje definice složených primárních klíčů, které využívají některé spojené tabulky. Složený klíč tvoří třída, která je anotovaná `@Embeddable`, pomocí `@EmbeddedId` se pak objekt dané třídy označí jako klíč.

Druhým adresářem je serializer, který obsahuje definice pro transformování dat na JSON formát. Spring Boot má svůj vlastní výchozí serializér, který ale z důvodů dvousměrného odkazu může vytvořit nekonečnou smyčku dat. To, zda se má při serializaci použít vlastní serializér, se značí anotací `@JsonSerialize(using = RoleSerializer.class)`.

Repository

Adresář repository obsahuje soubory s rozhráním (interface), které rozšiřuje `JpaRepository` z modulu Spring Data JPA. Rozhraní jsou vytvořena pro každou třídu z domain a jsou označena anotací `@Repository`. `JpaRepository` zpřístupňuje základní metody pro komunikaci s databází neboli uložení jednoho či více záznamů, podobně pak vyhledávání a mazání dle id a mnohem více. Co se týče ukládání, tak `JpaRepository` rozeznává ukládání nového objektu od editace existujícího pomocí toho, zda atribut, který je označen jako primární klíč, má hodnotu `null` či jinou.

Spring Data JPA umožňuje také vytvářet vlastní metody pro přístup např.: `findByIdIdProjektuAndIdIdRizika(long idProjektu, long idRizika)` pro nalezení záznamu, jehož id je rovné `idProjektu` a `idRizika`. Takto vytvořené metody se ve Spring Data JPA dokumentaci nazývají *Query metody*.

Service

V adresáři service se nacházejí soubory se třídami, které mají anotaci `@Service`. V těchto souborech lze najít funkce, které mají na starosti úpravu dat před tím, než se zavolá odpovídající repository objekt, který dané úpravy promítne do databáze. Jedná se tedy o tzn. bussiness vrstvu, která obsahuje logiku celé aplikace. Tato funkčnost je velmi často spojována s funkčností řadiče (*controller*), je ovšem doporučeno jejich oddělení, díky čemuž, pokud se změní druh databáze, tak se budou muset změnit pouze funkce ve `@Service`. Dále to také umožňuje oddělení logiky od zpracování příchozího požadavku, díky čemuž bude mít řadič na starosti pouze příjem dat a vrácení výsledku dotazu.

Všechny třídy využívají mechanismu automatického vkládání závislého repository objektu (*dependency injection*), pomocí kterého se přikazují změny v databázi. Toto vkládání umožňuje anotace `@Autowired`. V každé `@Service` třídě se vyskytují minimálně tyto čtyři metody:

- **saveOrUpdate** pro ukládání nového či pozměněného objektu
- **findAll** pro vyhledání všech objektů
- **findById** pro vyhledání jednoho objektu podle jeho id
- **delete** pro smazání požadovaného objektu

FindAll, findById a delete jsou ve většině případů pouze jednoduché funkce, které předávají příkaz z řadiče do repository objektu. SaveOrUpdate metody provádějí ještě dodatečné úpravy dat před jejich uložením.

Dodatečnými úpravami se zde myslí, že pokud např. objekt má v jednom z atributů odkaz na jinou instanci (neboli záznam v tabulce obsahuje cizí klíč jiné tabulky), tak při tvorbě tohoto objektu z dat ve formátu JSON (provádí se v řadiči k adresáři web) bude u odkazované instance inicializovaná pouze jeho hodnota id. Z toho důvodu se musí dodatečně z databáze získat objekt s daným id a přiřadit se k právě tvořené instanci.

Dalším případem je, pokud se ukládá pouze modifikace již existujícího objektu. Přijata instance neobsahuje seznamy všech objektů, se kterými je ve vztahu. Pokud by ale daný objekt v databázi, který chceme modifikovat, nějaké vztahy měl, ale při ukládání jeho novější verzi je neměl nastavené, tak by systém uvažoval, že je má ze společných tabulek smazat. Z toho důvodu se musí, ještě před uložením objektu, vybrat z databáze původní verze a zkopírovat potřebné záznamy do nové verze.

Další úpravy při ukládání se provádí také, pokud právě ukládaný objekt nějakým způsobem ovlivňuje záznamy, které se v databázi již nacházejí. V tomto případě, se nejdříve musí načíst ovlivněné objekty, pozměnit jejich atributy a uložit je. Až po této sekvenci kroků, se nový objekt uloží.

Web

Adresář web obsahuje řadiče, které určují API serveru. Každá třída řadiče je anotována s `@RestController` a `@RequestMapping`. `@RestController` určuje, že řadič přijímá REST požadavky. `@RequestMapping` pak nastavuje prefix API, kterým je `api/npr`. Podobně jako třídy v service i tady se v na začátku každé třídy vkládá závislý objekt, tentokrát ale je jeho instance ze service. Každá metoda třídy řadiče obsluhuje jeden požadavek a anotace nad ní pak určuje, na jaký druh HTTP metody reaguje a také ji přiděluje suffix adresy pro získání zdroje. Používané anotace určující druh HTTP metody jsou následující:

- `@PostMapping("/project")` pro POST požadavky, v tomto případě se jedná o uložení nového nebo editaci již existujícího projektu.
- `@GetMapping("/projects")` pro GET požadavky, zde jako získání všech projektů.
- `@DeleteMapping("/project/{id}")` pro DELETE požadavky, zde pro smazání projektu s daným id.

Metody řadiče pak na požadavky odpovídají seznamem `Iterable`, pokud se jedná o GET požadavek na získání více záznamů, anebo pomocí objektu `ResponseEntity`, který reprezentuje celý objekt HTTP odpovědi a umožňuje nastavit jeho hlavičku, tělo a návratový kód. Při POST požadavcích (zdrojový kód 5.6) metody řadiče přijímají objekt, který se má uložit do databáze. Tento objekt je původně ve formátu JSON, ale pomocí anotace `@RequestBody` se provede automatická deserializace na požadovaný formát objektu. `@Valid` pak aktivuje validaci atributu, u kterého se nachází. To jaká pravidla musí atribut validovaného objektu splňovat určují anotace použité při definování třídy v domain např. `@NotBlank(message = "Název role musí být zadán")`, který určuje, že daný String atribut nemůže být prázdný. V případě, že validace skončí neúspěchem, tak se její výsledek společně s chybovými zprávami nachází v objektu `BindingResult`.

```

@PostMapping("/role")
public ResponseEntity<?> addRole(@Valid @RequestBody Role role,
    BindingResult result) {
    ResponseEntity<?> errors = SpolecneFunkce.errorCheck(result);
    if (errors != null)
        return errors;
    Role newRole = roleService.saveOrUpdate(role);
    return new ResponseEntity<Role>(newRole, HttpStatus.CREATED);
}

```

Zdrojový kód 5.6: Metoda řadiče pro uložení nového či editovaného objektu Role [zdroj vlastní]

Pro naznačení, kterého objektu se dotaz týká, např. získání projektu, jehož id je 1 se v parametrech obsluhujících metod označuje anotací `@PathVariable`. Díky tomu lze pak mít nad metodou anotaci s např. `@DeleteMapping("/project/id")`, která se při volání z klienta použije jako `/project/1`. V případě, že se objekt s daným id v databázi nenachází, tak se vrátí objekt `ResponseEntity` se stavovým kódem **404**, který se v HTTP používá pro označení, že daný dokument není nalezen.

Odpověď s kódem 404 se také vrátí v případě, že server přijme požadavek, pro který ale neexistuje obsluhující metoda.

Seznam všech dostupných API lze nalézt v příloze **B**.

Security

Jak se již dříve zmiňovalo, zabezpečení aplikace zajišťuje modul Spring Security. Soubory týkající se zabezpečení se nacházejí v adresáři `security`. Konfigurace zabezpečení se provádí v souboru `SecurityConfiguration.java`. Přihlašování probíhá pomocí přihlašovacího formuláře a server očekává, že obdrží tři parametry – `username`, `password` a `remember`, kde `remember` má serveru napovědět, zda si pamatovat přihlášení uživatele. Server má nastaveno (soubor `MyAuthSuccessHandler.java`), aby při neaktivitě delší než **60 vteřin** uživatele automaticky odhlásil. Neaktivitou se myslí, že klient neodešle po dobu 60-ti vteřin žádný požadavek. Při zapamatování, se ale po přihlášení uživateli odešle cookie soubor, klient pak toto cookie připojuje ke každému požadavku, takže i když server odhlásí uživatele z důvodu neaktivity, tak po dobu platnosti tohoto cookie, server nebude znovu vyžadovat přihlášení.

Pro CSRF ochranu se také využívá cookie soubor. Server daný ochranný token pošle ve formě cookie. Klient pak tento token odesílá ve hlavičce každého POST požadavku.

5.4.2 Klient

Při představování technologií jsem zmiňovala, že základem klientské části je knihovna React. Aplikace se váže na HTML dokument pomocí `id` atributu `root` v `index.html`, který se nachází v adresáři `public`. Funkci navázání lze pak nalézt v hlavní zdrojové složce klienta `src` v souboru `index.js` pojmenovanou jako `ReactDOM.render()` (příklad této funkce již byl k vidění dříve na příkladu **5.3**), která připojuje komponentu `App` obsahující celou aplikaci.

Na začátku je ještě vhodné zmínit co všechno se do systému Redux, který byl vysvětlen dříve v této kapitole, ukládá a jak se používá. Objekt `store` se vytváří ještě před tím, než se celá aplikace načte a při jeho tvorbě se používá kombinovaný objekt `reducer`, který zpřístupňuje všechny funkce na změnu stavu. Store se pak zpřístupňuje komponentě `App`

a všem jejím potomkům, pomocí komponenty `Provider`, která ji obaluje. Používá se hlavně pro ukládání informací o projektu, jehož informace se mají zobrazit – základní údaje, kdo je manažer projektu a kdo řešitel, celková velikost projektového týmu, počet dnů do dokončení, informace o rizicích identifikovaných na projektu a také data dotazníku jež k němu byl přiřazen. Dále se také ukládají informace samotné aplikace – jaký mód je aktivní (světlý nebo tmavý), titulek aktivního okna, zda je menu otevřené nebo zavřené nebo zda se na serveru vyskytla chyba a klient musí zobrazit chybovou stránku. Dodatečně se do store ukládají i informace o aktivním uživateli jako je jeho id a role. Actions a reducers, které Redux používá se nacházejí ve svých odpovídajících adresářích actions a reducers.

Komponenty se primárně nacházejí v adresáři `components`, kde jsou rozdělené podle toho, jakou část aplikace tvoří. Nástroj lze rozdělit na sedm částí – **přihlašování**, **dashboard**, **správa uživatelů**, **projekty**, **centrální registr rizik**, **dotazníky**, **otázky a odpovědi**.

Většina koncových komponent je samostatných, což znamená, že si sami načítají případně odesílají nová nebo modifikovaná data pomocí knihovny `axios`.

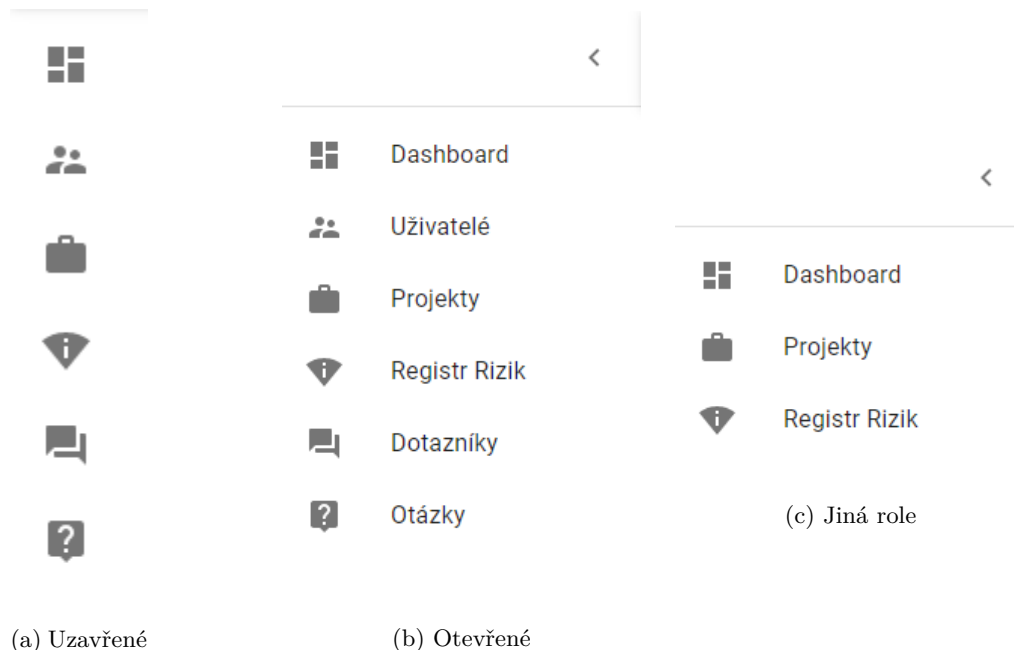
V aplikaci se často zobrazují data formou záznamů v tabulkách. Každá tabulka umožňuje dynamické vyhledávání v načtených záznamech pomocí vyhledávacího pole. Taktéž lze v každé provádět přidávání, mazání a editaci dat. Pro přidání a editaci záznamu se otevře dialog s odpovídajícími políčky. Pokud všechny zadané hodnoty budou validní, tak se při uložení provedou následující dvě akce – data se odešlou v požadovaném formátu na server pomocí `axios` s metodou `POST` a zadané hodnoty se vrátí do komponenty tabulky, kde se zobrazí nový nebo editovaný záznam. Sloupce v tabulkách lze uchytit a přesunout na místo jiného sloupce, případně lze hodnoty v každém z nich seřazovat vzestupně nebo sestupně dle abecedy.

Přihlašování

Při načtení se aplikace snaží ze serveru získat informace o uživateli. Pokud server vrátí kód 401 s tím, že uživatel není přihlášen, tak se zobrazí stránka s přihlašovacím formulářem, který tvoří komponenta `LoginForm`. V opačném případě, bude uživatel automaticky přeměrován na svou uvítací stránku – dashboard. Pokud se z autentizace na serveru opět vrátí 401 zobrazí se informace, že uživatel zadal špatné přihlašovací údaje. Při přihlašování lze vybrat možnost zapamatovat si aktuálně přihlášeného uživatele.

Boční menu a panel aplikace

Hlavní panel aplikace obsahuje ikonová tlačítka pro otevření a zavření bočního menu, přepnutí na světlý a tmavý mód a také tlačítko na odhlášení uživatele. To, zda je menu zavřené nebo otevřené se ukládá do Redux store, odkud si to následně komponenta `MenuAppBar` stáhne. Panel aplikace také obsahuje název, který určuje aktuální polohu v aplikaci, i tento název získává komponenta ze store. Přístup k jednotlivým částem nástroje je ovlivněn rolí, která byla uživateli přidělena. Uživatel s bezpečnostní rolí `ADMIN` má přístup ke všem částem aplikace. `MANAGER` a `USER` pak mají, co se týče menu, omezený přístup pouze k Dashboard, Projekty a Registr rizik. Rozdíl mezi otevřeným a uzavřeným menu lze vidět na obrázku 5.1. Na stejném obrázku se také nachází rozdíl zobrazení menu pro uživatele `ADMIN` a pro všechny ostatní role.



Obrázek 5.1: Porovnání uzavřeného a otevřeného menu role ADMIN s menu ostatních rolí [zdroj vlastní]

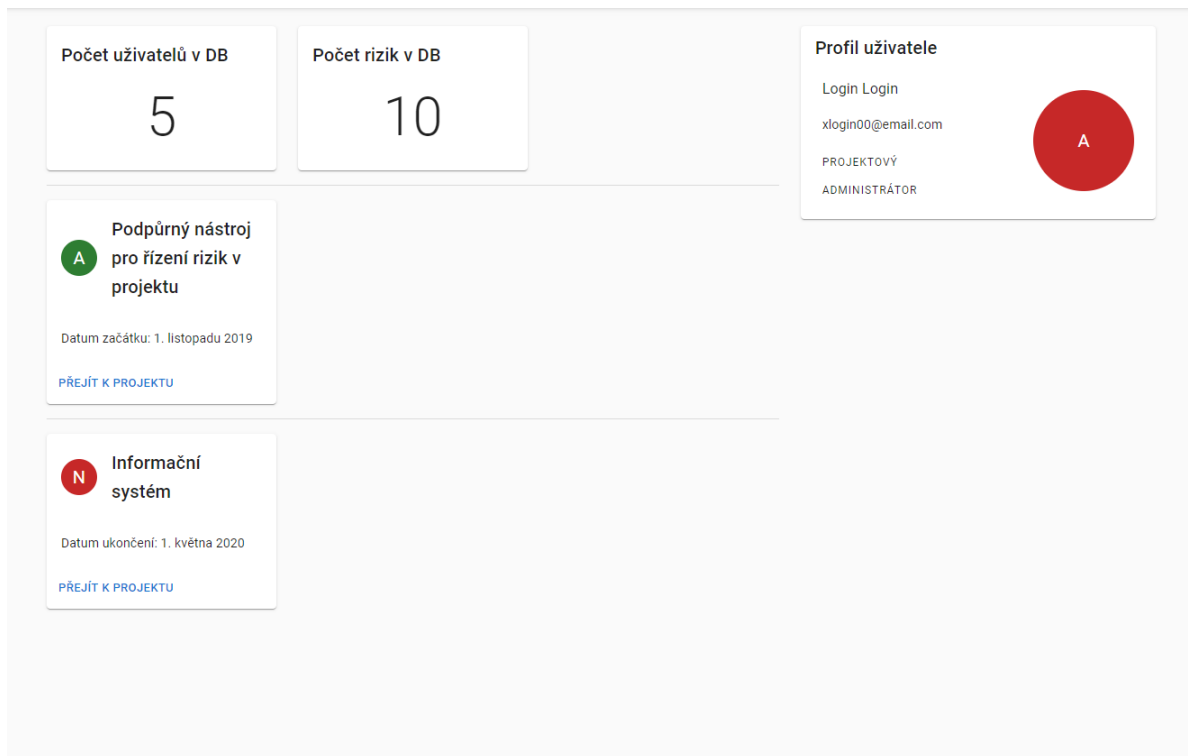
Dashboard

Komponenty týkající se Dashboardu se nacházejí v adresáři `dashboard`. Stránka se nachází na adrese `/app/dashboard`. Hlavní komponentou je `DashboardPage`, pomocí které se dashboard rozděljuje na dvě části – levou a pravou.

Pravou část tvoří komponenta `Profile`, která zobrazuje profil aktuálně přihlášeného uživatele. Na kartě profilu se nacházejí základní informace uživatele – jeho jméno a příjmení, e-mail a aktuálně přiřazená role.

Levá komponenta se pak liší dle přiřazené bezpečnostní role. Každá role má svou vlastní komponentu – `UserPart`, `ManagerPart`, `AdminPart`. V kódu `DashboardPage.js` se ještě před tím, než se vloží do `DashboardPage`, tak se vytvoří pomocné objekty, které se inicializují funkcemi, které pomocí informací ve Redux store určí, zda komponentu vykreslit nebo místo ní vrátí hodnotu `null`, která nic nezobrazuje. Ačkoliv jsou všechny tři objekty vložené, tak se vykreslí pouze jeden z nich.

Uživatel s rolí `USER` se zobrazují karty aktivních projektů, ke kterým je přiřazen. Karta obsahuje název projektu, datum, kdy k němu byl uživatel přidělen a odkaz, který uživatele přenesne na stránku s informacemi o daném projektu. `MANAGER` uživateli se zobrazují aktivní i neaktivní projekty, které řídil. V jeho případě se na kartách aktivních projektů zobrazuje jeho počáteční datum. U ukončených projektů pak jejich datum uzavření. U role `ADMIN` se kromě aktivních a neaktivních projektů zobrazují i informace o počtu uživatelů v databázi a počtu rizik v centrálním registru rizik. Obrazovku dashboardu administrátora lze vidět na obrázku 5.2. Pokud v databázi nebudou pro danou část data, tak se zobrazí odpovídající informace o chybějících datech.



Obrázek 5.2: Dashboard v aplikaci pro administrátora [zdroj vlastní]

Správa uživatelů

Správa uživatelů je přístupná pouze administrátorovi a lze ji nalézt na stránce `/app/users`. Soubory správy uživatelů se nacházejí ve složce `user`. Hlavní rodičovskou komponentou je zde `UserTabs`, která následně zobrazuje `RoleTable`, `UserBarChart` a `UsersTable`.

`RoleTable` zobrazuje data uživatelských rolí v tabulce. Klient získá ze serveru id role, její název a id bezpečnostní role. Následně ještě musí komponenta načít id a název bezpečnostních rolí. Výsledná data se zobrazí jako záznamy v tabulce `Role` jejíž sloupce jsou název role a název bezpečnostní role. Jako dialog pro přidání a editaci záznamu se používá `RoleDialog`.

`UsersTable` funguje na stejném principu jako `RoleTable`. Komponenta pomocí `GET` metody získá data všech uživatelů – id, jméno a příjmení, login, heslo, e-mail a id role. Následně se ještě musí získat data rolí, aby bylo možné vytvořit jejich seznam. Dialogem pro tuto tabulku je `UserDialog`.

`UserBarChart` je sloupcový graf vytvořený pomocí knihovny `Nivo`. Zobrazuje kolik uživatelů má přidělenou danou roli. Všechny grafy nacházející se v aplikaci zobrazují pouze hodnoty, které mají nějaké hodnoty. Pokud tedy danou roli nebude mít přiřazený žádný uživatel, tak se v grafu nezobrazí.

Komponenty často zobrazují nějaký společný údaj. V tomto případě všechny tři sdílejí data k rolím v systému. Pokud se tedy údaje role změní např. se přiřadí nový uživatel nebo se změní její název, tak musí všechny další komponenty znovu načít data. Informace o změně se ukládá do `Redux` store, a všechny komponenty, které mohou být změnou ovlivněné, tak daný údaj sledují a reagují na jeho změnu.

Správa projektů

Nejdůležitější částí vytvořeného nástroje je správa projektů. Komponenty, které tvoří tuto část se nacházejí v adresáři `project`. Správa projektu se pak dělí ještě na tři další části – zobrazení karet všech projektů, vytvoření a uložení projektu do databáze a potom následuje samotná správa informací na projektu, jeho rizik, swot tabulky a vyplňování dotazníku k projektu.

```
const validNazevPopis = Yup.object({
  nazev: Yup.string().required('Vyzadovano'),
  popis: Yup.string().required('Vyzadovano'),
});

const validDatумы = Yup.object( {
  start: Yup.date().typeError("Nespravny format datumu: DD. MM. RRRR"),
  konec: Yup.date().typeError("Nespravny format datumu: DD. MM. RRRR")
    .min(Yup.ref('start'), "Datum ukonceni nemuze byt driven
      nez datum zacatku"),
});

const validManager = Yup.object( {
  manager: Yup.number().typeError("Manazer musi byt zadan")
    .required("Manazer musi byt zadan"),
});

const validationSchemaAll = Yup.object({
  nazev: Yup.string().required('Vyzadovano'),
  popis: Yup.string().required('Vyzadovano'),
  start: Yup.date().typeError("Nespravny format datumu: DD. MM. RRRR"),
  konec: Yup.date().typeError("Nespravny format datumu: DD. MM. RRRR")
    .min(Yup.ref('start'), "Datum ukonceni nemuze byt driven
      nez datum zacatku"),
  manager: Yup.number().typeError("Manazer musi byt zadan")
    .required("Manazer musi byt zadan"),
});

const validationSchema = [validNazevPopis, validDatумы, validManager,
  validationSchemaAll, validationSchemaAll];
```

Zdrojový kód 5.7: Části validačního schématu formuláře pro tvorbu projektu [zdroj vlastní]

Zobrazení všech projektů se nachází na adrese `/app/project`. Hlavní komponentou je `ProjectCards`, pomocí které se zobrazují karty s informacemi ke každému projektu. Na kartě se nachází název projektu, jméno a příjmení vedoucího člena, jeho popis, označení, zda je aktivní nebo ukončený a tlačítka pro smazání a zobrazení podrobných informací o projektu. Tlačítko na smazání je zobrazitelné pouze administrátorovi, který případné smazání musí ještě odsouhlasit ve výzvě, která se mu zobrazí v dialogovém okně. Až po tomto odsouhlasení se smažou veškerá projektová data – samotný projekt, záznamy o přiřazení uživatelů, přidělená rizika, přiřazení dotazníku a informace o tom, jak uživatelé na

tento dotazník odpovídali. Uživatel `USER` se zobrazí pouze karty projektů, ke kterým je přihlášen (a na rozdíl od dashboardu zde se zobrazí i neaktivní projekty). `MANAGER` může vidět a zobrazovat všechny projekty. `ProjectCards` také obsahují pole po vyhledávání zadaného textu v kartách projektu. Na rozdíl od tabulek, kde tuto funkčnost automaticky poskytuje knihovna `material-table`, tak v případě tohoto pole se jeho tvorba inspirovala vzhledem políčka v tabulkách, jeho funkčnost pak obstarává funkce `handleSearchField` v `ProjectCards.js`.

K vytvoření projektu se lze dostat dvojným způsobem – přejít na adresu `/app/project/addForm`, která je přístupná pouze administrátorovi nebo při zobrazení karet všech projektu, kliknout na ikonku přidání nového záznamu, který se nachází napravo od vyhledávacího pole (toto tlačítko je opět viditelné pouze administrátorovi). Jedná se o několika krokový formulář, který je vytvořen v komponentě `AddProjectForm` v adresáři `addProject`. Podobně jako ve všech formulářích i tady zodpovídá za jeho správu knihovna `Formik`, která z jednotlivých polí získává data a pomocí knihovny `yup` je validuje. Tento formulář je ale rozdílný tím, že každým krokem se mění jeho validační schéma. Všechny části schématu lze vidět ve zdrojovém kódu 5.7.

The screenshot shows a web interface for managing project data. At the top, there are navigation tabs: 'INFORMACE O PROJEKTU' (selected), 'RIZIKA PROJEKTU', 'SWOT ANALÝZA', and 'DOTAZNÍK'. Below the tabs is a form with several sections:

- Název projektu:** A text input field containing 'Informační systém' with a blue edit icon on the right.
- Aktivita projektu:** A card showing 'Aktivní' with a blue edit icon.
- Počet rizik:** A card showing the number '3'.
- Počet členů týmu:** A card showing the number '3'.
- Dnů do ukončení:** A card showing the number '10'.
- Popis projektu:** A text area containing a placeholder text starting with 'Projekt obecného informačního systému...' and a blue edit icon.
- Datumy projektu:** A section with two date pickers: 'Začátek' (01. 05. 2019) and 'Konec' (03. 06. 2020), each with a blue edit icon.
- Manažer projektu:** A text input field containing 'Alan Riger' with a blue edit icon.
- Řešitelé projektu:** A list of names: 'Diana Laris' and 'Casper Novak', with a blue edit icon.

At the bottom right of the form is a button labeled '← PŘEHLED'.

Obrázek 5.3: Správa základních údajů projektu [zdroj vlastní]

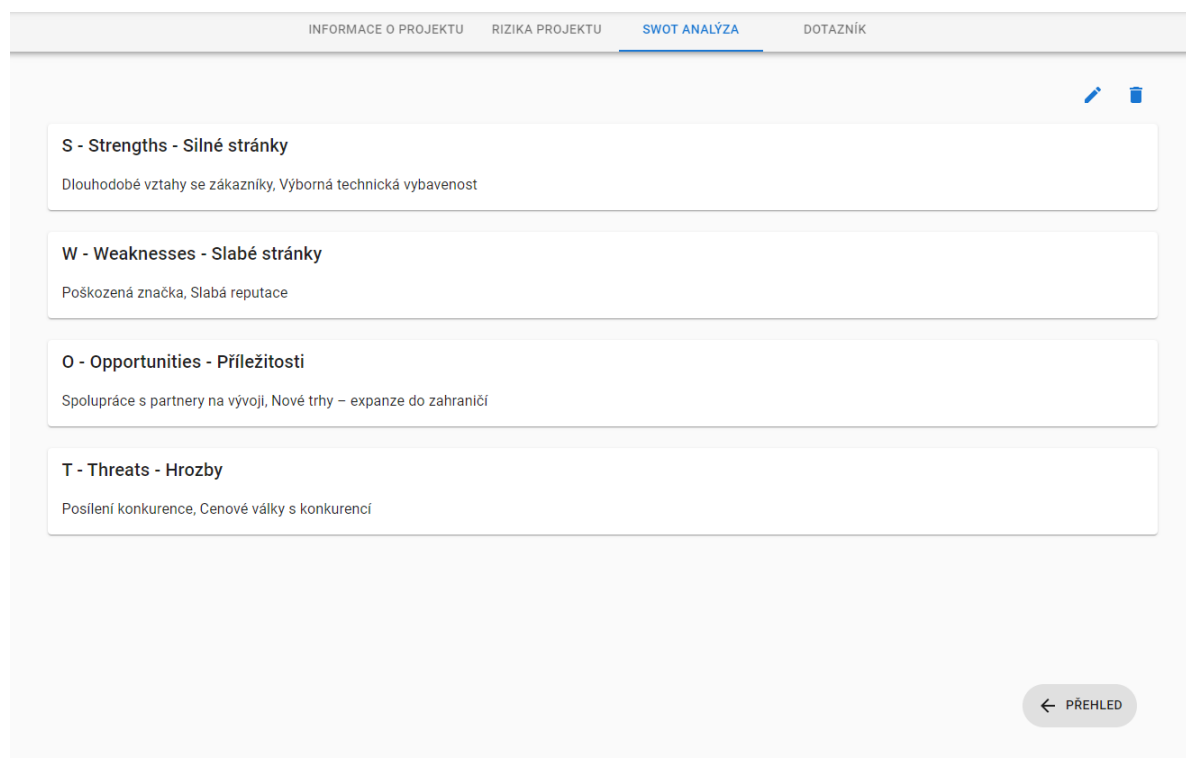
Ve formuláři jsou následující části – název a popis projektu, datum začátku a datum předpokládaného konce, přiřazení manažera, přiřazení řešitelů. `Formik` nedovolí přejít k dalšímu kroku, pokud validační schéma označuje, že se vyskytuje chyba. Přiřazení řešitelů je jako jediná nepovinná položka. Po odeslání dat na server bude uživatel přesměrován na stránku s údaji právě vytvořeného projektu, kde může provádět jeho celkovou správu.

Samotná správa projektů se pak nachází na adrese `/app/project/id`, kde `id` je id projektu, které mu přidělil server. Je rozdělena na čtyři části – **základní údaje**, **rizika**, **swot**

a **dotazník**. Hlavní komponentou je `ProjectPageID`, která načte všechna data projektu a uloží je do Redux store. Části pak již načítají potřebná data ze store. Manažeři, kteří nejsou k projektu přiřazeni si mohou prohlížet jeho údaje, ale nemůžou nic měnit.

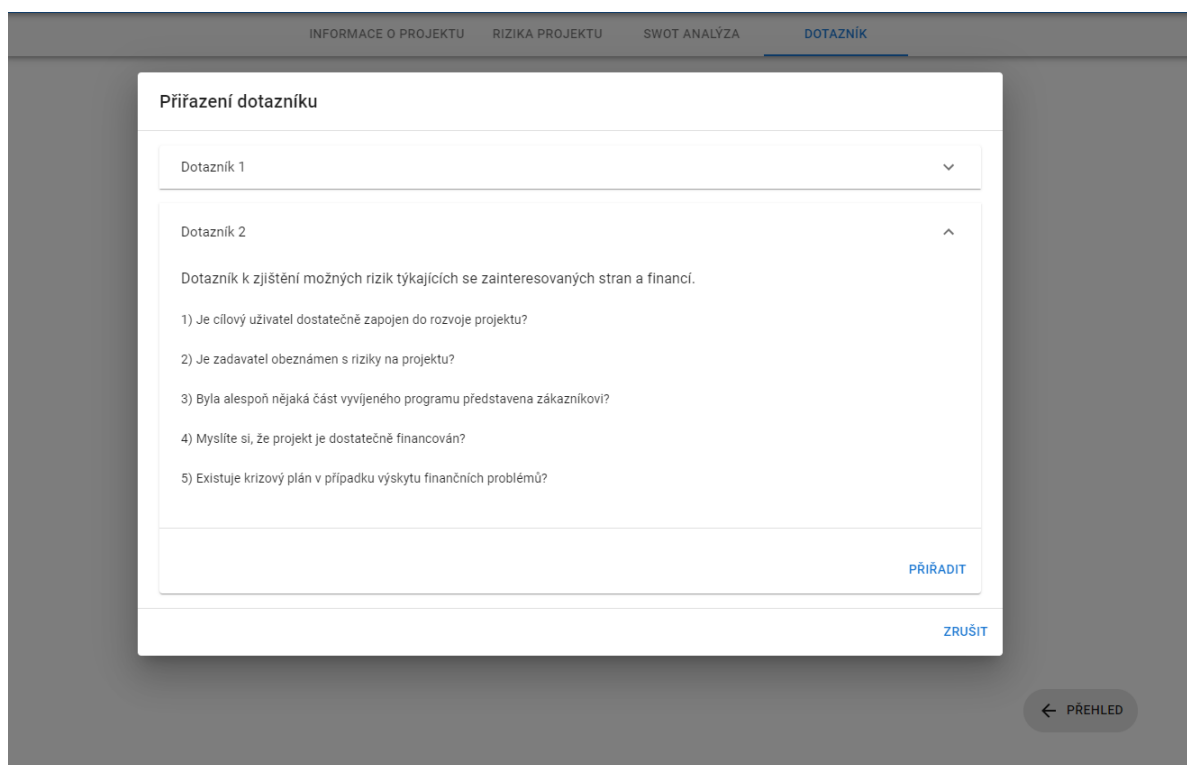
`ProjectInfoTab` zobrazuje komponenty, které se používají pro správu základních údajů projektu (obrázek 5.3). Většina se nachází v adresáři `projectInfo`. Tlačítko pro editaci údajů jako název, popis a datумы se zobrazí pouze přiřazeným členům řešitelského týmu. Přiřazovat a odhlašovat řešitelé může pouze manažer projektu a manažera pak může změnit pouze administrátor, který zároveň může editovat i všechny předcházející prvky. Pro základní údaje se ve skutečnosti používají dvě komponenty, jedna pro obvyčejné zobrazování textu (většinou se jedná o `Typography` z `Material-UI`) a druhá pro editaci. Tyto komponenty po potvrzení editace údaje, použijí stažená projektová data ze store a odešlou na server novou verzi projektu. Následně se ještě upraví i data v Redux store. Dodatečně se zobrazují i doplňkové informace o velikosti řešitelského týmu, zda je projekt stále aktivní, o počtu přiřazených rizik a kolik dnů zbývá do ukončení projektu.

`ProjectSwotTab` umožňuje správu a zobrazování swot tabulky. Komponenta, kterou lze vidět na obrázku 5.4 se nachází v adresáři `projectSWOT`. V případě, že projekt nemá swot vytvořené, tak je o tom uživatel informován a po stisku tlačítka pro tvorbu je mu zobrazena kostra tabulek v editačním módu. Při pokusu o smazání dat tabulky, musí akci uživatel ještě potvrdit v dialogu. Základní komponentou každé položky je `TextFieldEdit`, která vnitřně používá `Typography` pro zobrazení a `InputBase` při editaci. Obě jsou součástí knihovny uživatelských prvků `Material-UI`. Podobně jako u správy základních údajů i tady můžou provádět akce spojené s tabulkou pouze administrátor a uživatelé přiřazení k projektu.



Obrázek 5.4: Správa swot tabulky projektu [zdroj vlastní]

ProjectSurveyTab je hlavní dotazníkovou komponentou projektu. Společně se svými podpůrnými komponentami se nachází v adresáři projectSurvey. Podobně jako u swot tabulky i tady pokud ke projektu není přiřazen žádný dotazník, tak je o tom, uživatel informován. Tlačítko pro přiřazení je ovšem dostupné pouze manažerovi projektu a administrátorovi. Výběr dotazníků se provádí v dialogu, který lze vidět na obrázku 5.5.



Obrázek 5.5: Výběr dotazníku [zdroj vlastní]

Uživatelovi se v něm vždy zobrazí jméno dotazníku, jeho popis a jaké obsahuje otázky. Pokud projekt již má přiřazený dotazník, tak se opět zobrazuje jeho jméno, popis a dále také kolik otázek obsahuje a zda již uživatel na dotazník odpověděl. Dále stránka umožňuje dotazník zodpovědět a zobrazit statistiku odpovědí. Řešitelé projektu musí nejprve dotazník zodpovědět, až po tom jim je umožněno statistiku zobrazit. Manažer projektu a administrátor ji můžou zobrazovat ještě před zodpovězením dotazníku. Statistika se zobrazuje jako procentní hodnocení — jaké procento uživatelů (z celkového počtu co dotazník již zodpovědělo) odpovědělo na danou otázku touto odpovědí. Výsledná hodnota je zaokrouhlena na dvě desetinná čísla a nejvíce používaná odpověď je tučně zvýrazněná. Procenta pro každou odpověď počítá funkce `countPercentage` z `ProjectSurveyTab`.

`ProjectRiskTab` zobrazuje komponenty, které se používají při znázorňování rizik identifikovaných na projektu a které se nacházejí v adresáři `projectRisks`. Stránka je rozdělena na čtyři části. První pouze zobrazuje počet přidělených rizik. Druhá pak tvoří skupinu statistických grafů – druhy kategorií přidělených rizik a počet rizik v daných kategoriích (jedná se o sloupcový graf), dále pak koláčové grafy znázorňující stav zpracování rizika, pravděpodobnost výskytu, míru dopadů a prioritu při zpracovávání. Následující tabulka (5.1) ukazuje mapování slovních popisů atributů rizika používaných v klientské části s čí-

selnými hodnotami, které se ukládají do databáze (toto mapování lze také vidět v definici konstantních proměnných v adresáři constants).

Atribut	Hodnota v databázi				
	1	2	3	4	5
Stav	Nové	Rozpracováno	Hotovo		
Pravděpodobnost	Velmi nízká	Nízká	Střední	Vysoká	Velmi vysoká
Dopad	Velmi malý	Malý	Střední	Vysoký	Velmi vysoký
Priorita	Malá	Střední	Vysoká		

Tabulka 5.1: Mapování slovních popisů klienta na číselné hodnoty z databáze [zdroj vlastní]

Poslední dvě komponenty pak tvoří základ správy rizik na projektu. Jedná se o matici pravděpodobností a dopadů a tabulku s přidělenými riziky. Základem matice je modifikovaný HeatMap graf z Nivo knihovny.

Tento graf původně zobrazuje barvu políčka podle toho, jak velká je hodnota v dané buňce. Nivo ve výchozím nastavení nabízí dva tvary – čtverec a kruh, ale umožňuje si také vytvořit vlastní tvar. Tento vlastní objekt, v programu pojmenovaný jako `MyShape`¹, dostane již od Nivo přidělenou barvu (jak již bylo řečeno, podle hodnoty políčka), ale je možné ještě dodatečně vypočítat novou barvu a tou nahradit původně přidělenou. Barvu políčka vypočítává funkce `calculateColor`, která využívá vynásobenou číselnou hodnotu pravděpodobnosti a dopadu. Komponenta matice se nazývá `RiskMatrix`. Ve svých buňkách ukazuje počet rizik, které v atributech mají danou kombinaci hodnoty pravděpodobnosti a dopadu. Při najetí myši dané políčko, se zobrazí názvy rizik, které se tam nacházejí. Matici pravděpodobností a dopadů lze vidět na obrázku 5.6.

Poslední komponentou je tedy tabulka rizik přiřazených k projektu. Lze ji vidět na obrázku 5.7 a tvoří ji hlavně komponenta `ProjectRiskTable`.

Tato tabulka se mírně liší od ostatních tím, že je možné jednotlivé záznamy rozkliknout a nechat si tak zobrazit podrobnosti. Přidat riziko k projektu lze dvěma způsoby, buď se vybere již nějaké vytvořené z centrálního registru rizik nebo se vytvoří nové, které se při ukládání na server automaticky uloží i do centrálního registru. V případě, že uživatel se pokusí přiřadit riziko, které se již v tabulce nachází, tak na to bude upozorněn a pokud ho i tak uloží, tak nově uložené hodnoty přepíšou ty původní.

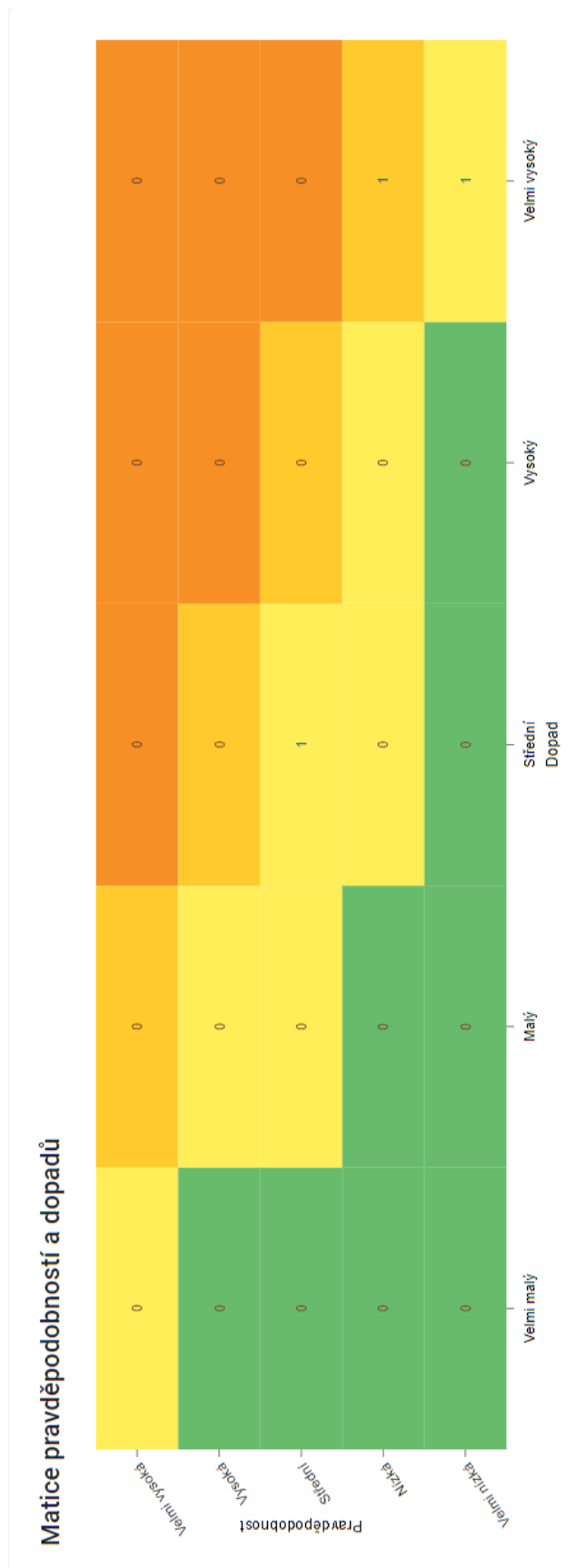
Centrální registr rizik

Registr rizik je dostupný všem uživatelům a zobrazuje se na adrese `/app/register`. Komponenty se nacházejí v adresáři `riskRegister`, přičemž `MainRiskRegisterPage` pak po stránce rozmísťuje následující čtyři komponenty.

`RiskCategoryTable` a `RiskRegisterTable` jsou funkčností a vzhledem obdobné s tabulkami ve správě uživatelů. První slouží pro tvorbu, editaci a smazání možných kategorií rizik. Druhá tabulka představuje centrální registr rizik, ze kterého se k projektům přidělují rizika. Každá z nich používá odpovídající dialog pro tvorbu a editaci záznamů.

`CategoryBarChar` a `RiskInProjectChart` jsou dodatečné statistické sloupcové grafy. První zobrazuje graf Kategorie rizik, kde se ukazuje počet rizik v jednotlivých kategoriích.

¹Je vhodné ještě zmínit, že vytvořený objekt `MyShape` je převzatý z knihovny Nivo a ve výchozím nastavení se používá pro zobrazení čtvercového tvaru, dodatečně se v něm ale ještě jednou vypočítává barva, kterou se má vykreslit.



Obrázek 5.6: Matice pravděpodobností a dopadů projektu [zdroj vlastní]

Registr rizik							Q Hledat	×	+
Název	Stav	Prevděpodobnost	Dopad	Priorita	Správce	Kategorie	Akce		
> Překročení rozpočtu	Nové	Nízká	Velmi vysoký	Vysoká	Alan Riger	Finance			
> Nízká výsledná kvalita	Hotovo	Velmi nízká	Velmi vysoký	Střední	Casper Novak	Kvalita			
∨ Špatná komunikace se zákazníkem	Rozpracováno	Střední	Střední	Malá	Diana Laris	Komunikace			

Popis
Komunikace se zákazníkem je nedostatečná, může vést k nepřesnostem na návrhu či snížené kvalitě

Popis dopadu
Zpoždění projektu, nespokojenost zákazníka, napomenutí zaměstnance

Plán řešení
Napomenou zaměstnance, domluvit přesčas na nadrobení zameškané práce

[← PŘEHLED](#)

Obrázek 5.7: Registr rizik projektu s otevřeným zobrazením podrobností u prvního rizika [zdroj vlastní]

Z grafu `RiskInProjectChart` lze pak vyčíst kolikrát se dané riziko již k nějakému projektu přiřadilo.

Správa dotazníků

Rozdělení správy dotazníků je podobné správě projektů. Také se jedná o tři části – zobrazení shrnujících karet, přidávání a editace dotazníků. Opět se jedná o část nástroje, ke které má přístup pouze administrátor. Komponenty používané v dotaznících se většinou nacházejí v adresáři `survey`.

Zobrazení karet dotazníků provádí `MainSurveyPage` a jeho podpůrná komponenta `SurveyCards`. Lze k nim přistoupit na adrese `/app/survey`. Na kartách se zobrazuje název dotazníku, jaký počet otázek obsahuje a zda je přiřazen k nějakému projektu. Dále také obsahují odkaz na přesměrování na stránku s podrobnými daty dotazníku a tlačítko na jeho smazání. Na stránce `MainSurveyPage` se nachází i tlačítko na vytvoření dotazníku.

Podrobné údaje dotazníku lze zobrazit na stránce `/app/survey/id`, kde `id` je id dotazníku, které mu přidělil server. Hlavními komponentami jsou `SurveyInfoPage` a `SurveyInfo` z podadresáře `surveyInfo`. Na této stránce může administrátor editovat název, popis, přidávat a odebírat otázky. Výběr otázky probíhá pomocí dialogu, kde lze filtrovat otázky podle oblastí.

Tvorba dotazníku probíhá na adrese `/app/survey/addSurvey` a využívá stejné komponenty, které se používají při editaci dotazníku.

Správa otázek a odpovědí

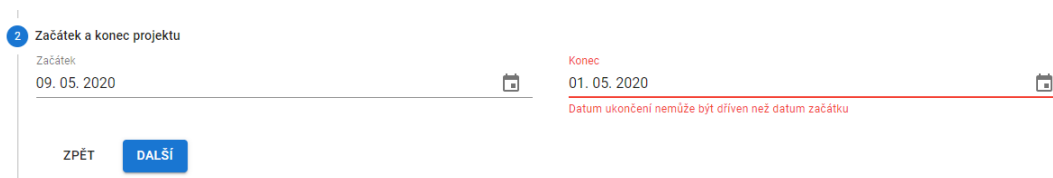
Stránka správy otázek a odpovědí, která se nachází na adrese `/app/question`, je tvořena třemi tabulkami, které umožňují zobrazování a správu odpovědí, oblastí otázek a samotných otázek. Hlavní komponentou, která zobrazuje již zmíněné tři části je `MainQA` nacházející se v adresáři `questionAnswer`. Stránka je přístupná pouze administrátorovi.

Tabulky Odpovědi (`AnswersTable`) a Oblastí otázek (`QuestionAreaTable`) fungují na stejném principu jako již předem představené tabulky např. tabulka Role ve správě uživatelů či tabulka Kategorie v centrálním registru rizik.

Tabulka Otázky (`QuestionTable`) umožňuje u jednotlivých záznamů si nechat zobrazit jaké obsahují odpovědi. Při tvorbě otázky lze dynamicky přidávat a odebírat přiřazené odpovědi, vždy ale musí být minimálně dvě a nesmí se opakovat.

Zobrazení chyb

Nástroj umožňuje zobrazovat několik druhů chyb. První druh se zobrazuje při vyplňování formulářů, které jsou spravovány pomocí knihovny Formik a validace probíhá pomocí Yup. Pokud zadaná hodnota nesplňuje validační schéma (nebo je hodnota povinná, ale uživatel nic nezadal), tak Formik, k danému políčku propaguje chybu. Vstupní prvky Material-UI mají výchozí podporu zobrazování chyb, kdy se políčko zbarví do červené a pod něj se vypíše chybová zpráva. Příklad lze vidět na obrázku 5.8, kdy uživatel při tvorbě projektu zadal datum ukončení před datem začátku projektu.



Obrázek 5.8: Indikace chyby při vyplňování formuláře [zdroj vlastní]

Druhým druhem chyb jsou chybové stavy, které se vrátí ze serveru. Jedná se hlavně o chybové kódy **401 Neautentizováno** a **404 Nenalezeno**. Neautentizováno se vrátí v případě, že server z nějakého důvodu odhlásí uživatele, velmi často z důvodu neaktivity. Nenalezeno vrátí, pokud uživatel chce získat zdroj, který neexistuje, v případě tohoto nástroje se může jednat o pokus o zobrazení neexistujícího projektu či dotazníků. Pro tento druh chyb je vytvořena komponenta `MyErrorHandler`, adresář `error`, která „obaluje“ celou aplikaci (aplikace je tedy jejím potomkem). Komponenta sleduje `error` stav v Redux store, který se nastaví na chybový kód, pokud některý axios požadavek se vrátí jako neúspěšný s daným chybovým stavem. V případě, že je `error` stav nastaven, tak se zobrazí odpovídající chybová stránka. Při 401 se uživateli vysvětlí, že byl serverem odhlášen a je mu zpřístupněn odkaz na přihlašovací stránku. Při 404 se pak na chybové stránce nachází odkaz na dashboard uživatele. Pokud ale error stav nastavený nebude, tak `MyErrorHandler` zobrazí svého potomka neboli aktuální komponentu aplikace, která se má zobrazit.

Tmavý vzhled

Pokud uživatel preferuje tmavý vzhled aplikace, tak hlavní panel aplikace obsahuje tlačítko pro přepnutí do tmavého režimu. Material-UI definuje styl jednotlivých prvků pomocí tzv.

theme a pokud chce vývojář tento objekt modifikovat a změny propagovat dále do aplikace, tak musí použít komponentu `ThemeProvider` z knihovny `Material-UI`, kterou „obalí“ aplikaci. Theme je ve výchozí konfiguraci nastavena na světlou verzi stylu, ale velice jednoduše lze přepnout do tmavé verze a to přiřazením řetězce `dark` do proměnné `palette.type`. Tato modifikace způsobí automatickou změnu několika dalších proměnných, které se týkají barev používaných v aplikaci. Jako indikace právě používaného módu se používá proměnná `appTheme` v Redux store.

Kapitola 6

Testování

Testování softwaru se provádí vždy za účelem něco zjistit. Ať už se jedná o zjištění kvality vytvořeného nástroje nebo ověření stavu databáze po provedení určité akce. Testy lze také automatizovat, kdy se spustí celé sady testů, jejichž výsledek lze předpovědět. Výsledky těchto testů se pak porovnávají se předpovězenými hodnotami a tím se určí jejich úspěch či neúspěch.

V této kapitole se popisuje průběh a způsob testování nástroje. Je rozdělena na tři části, kde v první se popisují automatizované testy prováděné na serveru. Ve druhé se krátce popíše práce s programem Postman a laděním API serveru. Třetí část pak popisuje testovací scénáře použití aplikace, které se prováděly při uživatelském testování klienta.

6.1 Automatizované testování

Spring Boot umožňuje vytvářet automatizované testy a podobně jako při programování jednotlivých modulů i zde lze testy vytvářet různými způsoby. Zároveň lze také vytvářet oddělené testy pro jednotlivé části aplikace.

Automatizované testy byly vytvořené pro serverovou část jako integrační testy, které mají určit, zda nově přidané moduly správně spolupracují s ostatními a netvoří neočekávané chyby. Testy se tvořily již v průběhu programování jednotlivých částí serveru, kdy nejprve se vytvořil test a následně se doprogramovaly jednotlivé potřebné moduly. Spojují části service, repository a domain. Soubory s testovacími metodami se nacházejí v adresáři test, kde každá metoda je anotovaná s `@Test`. Každý test je samostatnou jednotkou a neovlivňuje další testy, což znamená, že ačkoliv test může změnit databázi, tak po jeho ukončení jsou všechny změny vráceny zpět. V rámci této sekce testování se vytvořilo 70 integračních testů.

6.2 Postman a API

Program Postman se využíval při tvorbě řadičů (*controller*), které umožňují reakci na volání API serveru. V Postman lze vytvářet jednotlivé REST požadavky a zasílat je na server. Funguje tedy jako náhrada za klientskou část pro otestování a ladění reakcí na požadavky.

Při tvorbě požadavku lze ovlivňovat celý jeho formát – cookies co se mají poslat, hlavičky, jaká data a v jakém formátu. Po jeho odeslání Postman přijme ze serveru odpověď, ve které lze např. zkoumat její status (HTTP kód odpovědi), přeposlaná data, cookies, hlavičky atd. V případě, že požadavek byl chybný, tak se zobrazí odpovídající chybová zpráva i se svým kódem, kterou server poslal zpět.

Postman umožňuje také vytvářet automatizované testy, které zkoumají a vyhodnocují odpovědi získané ze serveru. Dohromady bylo takto vytvořeno a otestováno 71 REST požadavků.

6.3 Uživatelské testování

Cílem uživatelských testů bylo zjistit použitelnost výsledného nástroje. Bylo vytvořeno několik testovacích scénářů, které postupně měly provést uživatele přes jednotlivé případy použití. Tyto scénáře nejsou klasickými krokovými testy, kdy je uživatel pozorován, jak jednotlivé kroky plní. Místo toho je uživateli popsána situace (scénář), ve které se nachází, přičemž bude na něm, jak se s danou situací vypořádá.

Žádný z testerů se s aplikací během jejího vývoje nesetkal, ale z důvodu aktuální pandemické situace není ani jeden z nich přímo svázán s řízením projektů či s vyhledáváním rizik. Místo toho jsem vybrala testery, kteří mají již trochu rozsáhlejší znalosti práce na počítači nebo jsou v průběhu studia na univerzitě technického zaměření. Toto by mohlo simulovat malý tým pracující na nějakém start-upu. Všechny testy probíhaly na lokálním stroji, na kterém byl nástroj tvořen. Část testerů byla fyzicky přítomná při notebooku s aplikací, druhá část pak ovládala aplikaci vzdáleně pomocí sdílené obrazovky a sdíleném ovládacím myši a klávesnice. Ke sdílení se použila aplikace Teams od společnosti Microsoft.

Před začátkem testování bylo každému uživateli vysvětleno k čemu aplikace slouží, v jaké situaci se nachází a co má za úkol. Při plnění jednotlivých scénářů, které lze vidět v následujícím seznamu, byla snaha uživateli téměř nenapovídat. Podrobnosti k jednotlivým scénářům lze nalézt v příloze D.

1. Firma přijala dva nové zaměstnance – Alana Rigeru na pozici projektového manažera a Betty Rocky jako členku řešitelského týmu. Je nutné jim vytvořit účet, aby mohli aplikaci využívat. Tuto akci může provádět pouze administrátor.
2. Firma získala zakázku na nový projekt se zaměřením na záchranu velkých kočkovitých šelem. Jako administrátor je tvou povinností tento projekt vytvořit v nástroji a tím ho uložit do databáze. Název projektu je Kočkovité šelmy a je popsán jako projekt vytvoření informačního systému pro záchranou stanici velkých kočkovitých šelem. Začátek prací na projektu byl stanoven na 20.5.2020 a konec se předpokládá někdy na druhou polovinu roku 2022. Nově přijatí zaměstnanci (Alan Riger a Betty Rocky) jsou prvními členy řešitelského týmu a jsou přiřazeni ke svým odpovídajícím pozicím.
3. Pro projekt Kočkovité šelmy je nutné vytvořit nový dotazník, který může pomoci řešitelům projektu identifikovat rizika, která se mohou vyskytnout na tomto projektu. Před tvorbou dotazníku je vhodné zkontrolovat, zda v nástroji existují požadované otázky, které má dotazník obsahovat.
4. Jako manažer (Alan Riger) ti byl přiřazen nový projekt, který máš řídit. Projekt má pracovní název Kočkovité šelmy. Nově bylo ustanoveno, že datum ukončení bude 30.9.2022. Zároveň ti bylo sděleno, že Betty Rocky byla přesunuta do jiného projektu a jako náhrada za ní je přidělena Diana Laris. Změň informace na projektu, aby odpovídaly novým skutečnostem.
5. Řešitelský tým projektu Kočkovité šelmy identifikoval dvě nová rizika. Riziko Překročení rozpočtu z kategorie Finance je již známé a je již uloženo v centrálním registru rizik. Projekt je ale dobře financován, takže i když má střední pravděpodobnost

výskytu, tak bude mít na projekt pouze minimální dopad. Druhé riziko tým identifikoval poprvé – Pandemie infekční nemoci. Z důvodu situace ve světě je velká pravděpodobnost, že riziko zastihne i tento projekt, přičemž bude mít nejvyšší úroveň dopadu. Z tohoto důvodu by riziko mělo mít nejvyšší prioritu dalšího řešení. Jelikož se první riziko týká financí je vhodné, aby ho měl na starost manažer projektu. U druhého rizika může být správcem kdokoliv.

6. Jako manažer se chystáš vytvořit SWOT analýzu projektu Kočkovité šelmy. Pro jednodušší práci se můžeš inspirovat jinými projekty v systému.
7. Bylo ti oznámeno, že dotazník k projektu Kočkovité šelmy je připraven. Přiřadit ho k tomuto projektu, aby ostatní řešitelé mohli na něho odpovídat. Tuto akci proved jako manažer daného projektu.
8. Jednou z povinností řádového člena týmu je zodpovědět dotazník, který byl přiřazený k projektu, na kterém daný člen pracuje. Ujisti se, že jsi na něho zodpověděl a pokud ne, tak ho zodpověz.
9. Zajímá tě, jak ostatní uživatelé odpověděli na dotazník. Zjisti si tuto informaci.
10. Jsi ve tmavé místnosti a bílé pozadí aplikace dráždí tvé oči. Pokus se tento problém v aplikaci vyřešit.
11. Dostal jsi za úkol přepsat z papíru několik rizik do centrálního registru – proved to.
12. Projekt Kočkovité šelmy skončil neúspěchem. Firma nechce uchovávat jeho data. Jako administrátor jsi dostal za úkol tento projekt ze systému odstranit.

Testování použitelnosti se provedlo se čtyřmi testery. Každý z nich úspěšně dokončil všechny dříve zmíněné scénáře. Pokud se už nějaký zádrhel vyskytl, tak stačilo pouze trochu jinak formulovat scénář. I testování pomocí Teams proběhlo bez problémů.

Při vykonávání prvního scénáře byli testeři trošku v aplikaci zmateni, to bylo ovšem očekávané, když to bylo poprvé co nástroj viděli a používali. Jeden z testerů očekával tlačítko pro přidání uživatele v jiné podobě. Díky tomu, že všechna tlačítka pro přidávání jakéhokoliv objektu mají stejnou podobu a nachází se na stejné pozici, tak po splnění prvního scénáře již testeři přesně věděli, kde danou funkčnost najít.

Ve druhém a čtvrtém scénáři se manipuluje s přidáváním řešitelů do projektu. Tato akce se provádí pomocí výběru ze seznamu (prvek Select), ze kterého lze vybrat více nových členů týmu bez toho, aniž by se seznam po vybrání jednoho políčka uzavřel. Vybraní uživatelé se indikují zešednutím políčka. Většinou testerům, ale tato funkčnost připadala velice matoucí.

V pátém scénáři se má přiřazovat rizika k projektu. Někteří testeři chvíli tápali, kde můžou tuto akci provést, tabulka se totiž nachází až vespod karty s riziky na projektu a při zobrazení této karty ji nelze vidět.

Ostatní scénáře, pak již testerům, nedělaly žádné potíže.

Uživatelské testování pomohlo odhalit několik nedostatků ve vytvořeném nástroji, jako špatně se zobrazující hodnoty, chybějící popisky prvků či prvky, které se danému uživateli s danou rolí neměly zobrazit. Všechny odhalené nedostatky a některé výhrady testerů byly v aplikaci upraveny.

Kapitola 7

Závěr

Cílem této diplomové práce bylo navrhnout a vytvořit prototyp nástroje pro podporu řízení rizik, který by umožňoval uživateli jednodušší plánování, identifikaci a analýzu rizik. Tento cíl byl splněn.

Úvodem práce jsem musela nastudovat obecnou problematiku znalostních oblastí řízení projektu. Následně jsem se pak musela blíže seznámit s procesy plánování, identifikace a kvalitativní analýzy ze znalostní oblasti řízení rizik. Všechny tyto informace lze nalézt v kapitole 2, které vycházejí z nejnovějšího rozsáhlého standardu o řízení projektu od firmy PMI. Dále bylo také představeno, jak se při agilním řízení projektu nahlíží na jednotlivé znalostní oblasti.

Pomocí získaných znalostí jsem pak specifikovala požadavky na výsledný nástroj. Ten by měl podporovat uživatele při identifikaci a analýze rizik a zároveň by měl umožňovat vizualizaci rizik na projektu. Výsledná aplikace by měla být uživatelsky přívětivá s moderním vzhledem. Tento nástroj jsem následně navrhla jako webovou klient-server aplikaci. Pro realizaci serverové části se použil Java framework Spring Boot a pro klientskou pak JavaScript knihovna React s dodatečnými podpůrnými knihovnami. Jednou z hlavních podpůrných knihoven byla Material-UI, která umožnila tvorbu uživatelského rozhraní podle specifikace Material Design.

Výsledný nástroj splňuje požadavky, které na něho dle specifikace požadavků byly kladeny. Jedná se o klient-server aplikaci, kde ovšem obě části jsou od sebe oddělené, díky čemuž bylo umožněno je vyvíjet více méně nezávisle na sobě. Díky tomuto oddělení, lze také jednu z částí nahradit, přičemž ta druhá by se musela upravovat pouze minimálně.

Aplikace umožňuje správu rizik, přičemž ta je rozdělena na dvě části – centrální registr rizik, ve kterém se nacházejí základní obecné informace o riziku a lokální část, kde lze najít k riziku specifické informace, které se vážou k danému projektu. Vizualizace těchto specifických informací se provádí pomocí grafů, hlavně pak matice pravděpodobností výskytů a míry dopadů na projekt. Dále lze v aplikaci provádět základní správu uživatelů, tvořit a přiřazovat k projektům dotazníky, na které lze odpovídat a také k projektům vytvořit swot tabulky. Data se načítají a ukládají do databáze, která běží v paměti serveru.

Serverová část byla v průběhu programování testována automatizovanými testy, které kontrolovaly, zda jednotlivé části spolupracují a zda se do databáze ukládají správná data. Po dokončení obou částí se výsledný prototyp otestoval na jeho použitelnost. S výsledky bylo možné se seznámit v části Uživatelské testování v kapitole Testování.

7.1 Možná rozšíření

Každou aplikaci lze jistým způsobem upravovat a rozšiřovat. V této části se popisuje několik možných rozšíření, které by se mohly na nástroji provést.

Vylepšené přihlašování

V aktuální verzi nástroje probíhá základní autentizace pomocí formuláře, jehož data se posílají z klienta na server, kde se hodnoty porovnají s hodnotami v databázi a následně se vrátí úspěch či neúspěch. Spring Boot dále umožňuje také podporu OAuth 2.0, díky které se lze přihlásit pomocí Google nebo GitHub účtu. Dále také podporuje autentizaci pomocí LDAP serveru nebo využití JWT tokenů.

Reálné zprovoznění

Ačkoliv obě části aplikace jsou schopné běhu na lokálním stroji, tak pro potřeby reálného používání je toto řešení nedostatečné. Proto by bylo vhodné aplikaci nasadit na skutečné servery, příkladem může být nasazení na Heroku nebo Amazon web services. V úvahu připadá také úprava grafického vzhledu prvků.

Rozšíření správy rizik

U správy rizik by mohlo být uživateli umožněno nadefinovat vlastní atributy rizik, bez toho, aniž by se musel upravovat kód databáze. Další funkcionalitou by mohlo být vygenerování zpráv o rizicích na projektu nebo zpráv, které shrnují všechna rizika z centrálního registru. Dále by se také mohla vytvořit podpora pro tvorbu hierarchických grafů, které byly představeny v části o kvalitativní analýze rizik.

Větším rozšířením by pak mohlo být vytvoření automatizovaného analyzátoru, který by podle poskytnutých dat sám mohl předběžně určit jaká rizika se na projektu mohou vyskytnout.

Rozšíření aplikace

Samotnou aplikaci by se pak mohlo rozšířit, aby podporovala další znalostní oblasti. Znalostní oblasti řízení časového plánu a nákladu jsou velmi blízké oblasti řízení rizik. Proto by jedna z nich mohla být vhodným kandidátem, pro výběr dalších funkcionalit programu.

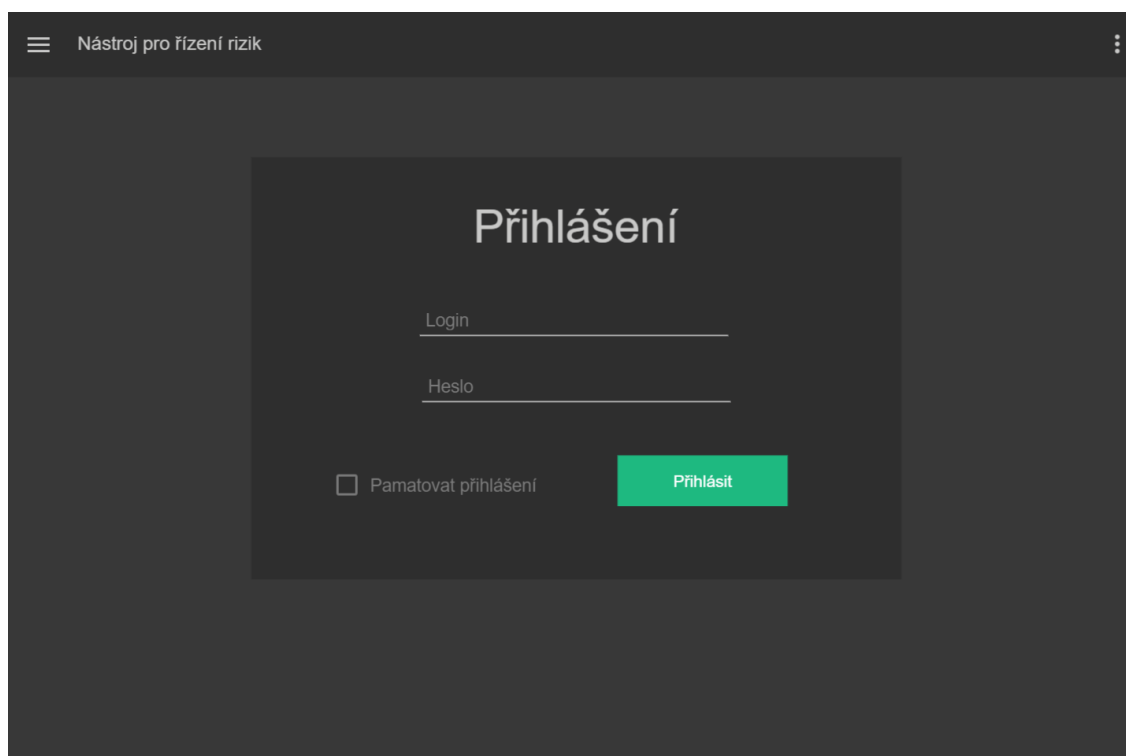
Literatura

- [1] *Agile Practice Guide*. USA: Project Management Institute, Inc, 2017. 167 s. ISBN 978-1-62825-199-9.
- [2] *A guide to the project management body of knowledge (PMBOK guide)*. 6. vyd. USA: Project Management Institute, Inc, 2017. 756 s. ISBN 978-1-62825-184-5.
- [3] DAVIES, A. *Waterfall vs Agile: Which Methodology is Right for Your Project*. [cit. 2019-01-02]. Dostupné z: <https://www.devteam.space>. Path: Blog; Search: Waterfall vs Agile.
- [4] FACEBOOK. *React*. [cit. 2020-04-09]. Dostupné z: <https://reactjs.org>.
- [5] FACEBOOK. Virtual DOM and Internals. *React*. [cit. 2020-04-15]. Dostupné z: <https://reactjs.org>. Path: Docs; FAQ; Virtual DOM and Internals.
- [6] FACEBOOK. Introducing JSX. *React*. [cit. 2020-04-15]. Dostupné z: <https://reactjs.org>. Path: Docs; MAIN CONCEPTS; Introducing JSX.
- [7] FLUID SOFTWARE LTD.. *Fluid UI - Web and mobile prototypes in minutes*. [cit. 2020-04-02]. Dostupné z: <https://www.fluidui.com>.
- [8] GOOGLE. *Material Design*. [cit. 2020-04-02]. Dostupné z: <https://material.io>.
- [9] GOOGLE. Rally. *Material Design*. [cit. 2020-04-02]. Dostupné z: <https://material.io>. Path: Design; Material Studies; Rally.
- [10] GOOGLE. Reply. *Material Design*. [cit. 2020-04-02]. Dostupné z: <https://material.io>. Path: Design; Material Studies; Reply.
- [11] HAT, R. *Hibernate*. [cit. 2020-04-10]. Dostupné z: <http://hibernate.org/>.
- [12] KRESLÍKOVÁ, J. *Základní pojmy projektového řízení*. [Online; navštíveno 10.10.2019]. Dostupné z: https://wis.fit.vutbr.cz/FIT/st/cfs.php?file=%2Fcourse%2FMMPR-IT%2Flectures%2Frok-2018%2F1s_zakladni-pojmy.pdf.
- [13] KRESLÍKOVÁ, J. *Řízení rizik projektu*. [Online; navštíveno 24.10.2019]. Dostupné z: https://wis.fit.vutbr.cz/FIT/st/cfs.php?file=%2Fcourse%2FMMPR-IT%2Flectures%2Frok-2018%2F10s_rizeni+rizik.pdf.
- [14] MAPLE, S. a BINSTOCK, A. JVM Ecosystem report 2018 – About your Platform and Application. *Snyk*, 17. října 2018 [cit. 2020-04-02]. Dostupné z: <https://snyk.io>. Path: Resources; Blog; Ecosystems.

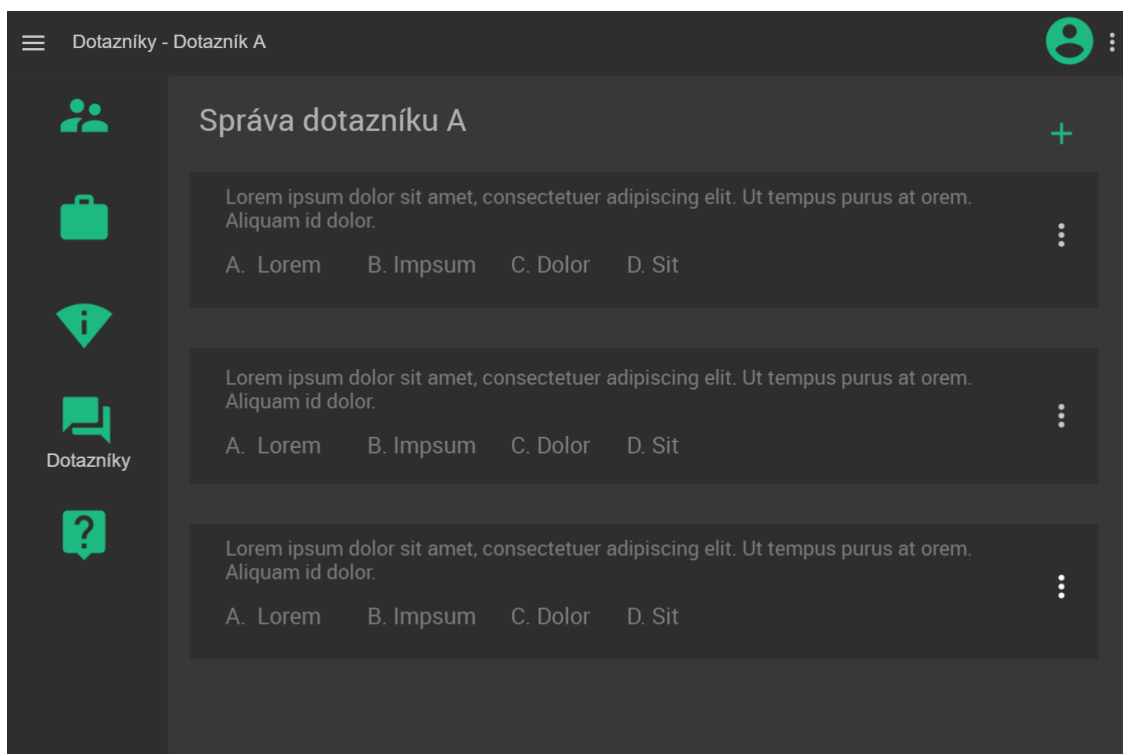
- [15] MUTIMUTEMA, T. 10 of the Most Popular Java Frameworks of 2020. *Stackify*, 24. února 2020 [cit. 2020-04-02]. Dostupné z: <https://stackify.com>. Path: Resources; Blog.
- [16] SCHWALBE, K. *Řízení projektů v IT*. 1. vyd. Brno: Computer Press, a.s., 2007. 720 s. ISBN 978-80-251-1526-8.
- [17] VMWARE. *Spring*. [cit. 2020-04-09]. Dostupné z: <https://spring.io>.

Příloha A

Návrh grafického uživatelského rozhraní



Obrázek A.1: Návrh přihlašovací stránky [zdroj vlastní]



Obrázek A.2: Návrh stránky správy dotazníku [zdroj vlastní]

Příloha B

API serveru

Všechna URI mají předponu `api/nprp`.

B.1 RoleController

`POST /role` Tvorba/editace role.

`GET /secRoles` Získání všech bezpečnostních rolí.

`GET /secRole/id` Získání bezpečnostní role dle jejího id.

`GET /roles` Získání všech rolí.

`GET /role` Získání role aktuálně přihlašeného uživatele.

`GET /role/id` Získání role dle jejího id.

`DELETE /role/id` Smazání role dle jejího id.

B.2 UživatelController

`POST /user` Tvorba/editace uživatele.

`GET /users` Získání všech uživatelů.

`GET /info` Získání informací o aktuálně přihlašeném uživateli.

`GET /users/count` Počet záznamů v tabulce s uživateli.

`GET /user/id` Získání uživatele dle jeho id.

`GET /user` Získání všech dat o aktuálně přihlašeném uživateli.

`DELETE /user/id` Smazání uživatele dle jeho id.

B.3 KategorieController

POST /kategorie Tvorba/editace kategorie rizika.

GET /kategories Získání všech kategorií rizik.

GET /kategorie/id Získání kategorie rizika dle jejího id.

DELETE /kategorie/id Smazání kategorie rizika dle jejího id.

B.4 RegistrRizikController

POST /risk Tvorba/editace rizika v centrálním registru.

GET /risks Získání všech rizik z centrálního registru.

GET /risks/count Počet záznamů v tabulce centrálního registru rizik.

GET /risk/id Získání rizika dle jeho id.

DELETE /risk/id Smazání rizika dle jeho id.

B.5 OblastOtazkyController

POST /questionArea Tvorba/editace oblasti otázky.

GET /questionAreas Získání všech oblastí.

GET /questionArea/id Získání oblasti otázky dle jejího id.

DELETE /questionArea/id Smazání oblasti otázky dle jejího id.

B.6 OdpovedController

POST /answer Tvorba/editace oblasti otázky.

GET /answers Získání všech oblastí.

GET /answer/id Získání oblasti otázky dle jejího id.

DELETE /questionArea/id Smazání oblasti otázky dle jejího id.

B.7 OtazkaController

POST /question Tvorba/editace otázky.

GET /questions Získání všech otázek.

GET /question/id Získání otázky dle jejího id.

DELETE /question/id Smazání otázky dle jejího id.

B.8 DotaznikController

POST /survey Tvorba/editace dotazníku.

GET /surveys Získání všech dotazníků.

GET /surveys/cards Vrátil objekty karet dotazníků, které shrnují jeho základní informace.

GET /survey/id Získání dotazníku dle jeho id.

DELETE /survey/id Smazání dotazníku dle jeho id.

B.9 ProjektController

POST /project Tvorba/editace projektu - základní informace.

POST /project/survey Přiřazení dotazníku ke projektu.

POST /project/idProj/survey/idSurv/user/idUser Uložení do databáze informací o tom, jak uživatel odpověděl na daný dotazník na projektu.

POST /project/swot Tvorba/editace SWOT tabulky projektu.

POST /project/id/manager Přidání/změna manažera projektu.

POST /project/id/users Přidání/změna ostatních řešitelů projektu.

POST /project/id/risk Přiřazení rizika k projektu.

GET /projects/active/user Získání modifikovaných objektů aktivních projektů, na kterých pracuje aktuálně přihlášený uživatel.

GET /projects/active/isActive/manager Získání modifikovaných objektů aktivních či neaktivních projektů, na kterých pracuje aktuálně přihlášený manažer.

GET /projects/active/isActive Získání modifikovaných objektů aktivních či neaktivních projektů.

GET /projects Získání všech projektů.

GET /projects/cards Získání objektů karet všech projektů, které shrnují jeho základní informace.

GET /projects/cards/user/id Získání objektů karet všech projektů, které shrnují jeho základní informace. Uživatel s daným id je aktivním řešitelem nebo manažerem projektu.

GET /project/id Získání projektu s daným id.

GET /project/id/users Získání všech aktivních i neaktivních řešitelů a manažerů projektu.

GET /project/id/survey Získání dat dotazníků, které se týkají projektu – jak uživatelé odpovídali na dotazník.

GET /project/id/risks Získání všech rizik přidělených projektu.

DELETE /project/id Smazání projektu dle jeho id.

DELETE /project/id/swot Smazání SWOT tabulky z projektu dle jeho id.

DELETE /project/id/risk/idRisk Odebrání rizika z projektu.

DELETE /project/id/survey Odebrání dotazníku z projektu.

Příloha C

Příklad dotazníku

Vytvořené dotazníky se mohou (podle otázek, které obsahují) zaměřovat na určité oblasti projektu. Příkladem mohou být oblasti rozpočtu projektu, komunikace se zainteresovanými stranami nebo použitých technologií na projektu.

Na následujících řádcích se nachází příklad obecného dotazníku, který obsahuje otázky z různých oblastí.

1. Rozumíte plně používaným technologiím?

- Ano
- Ne
- Možná
- Nevím

2. Je zadavatel obeznámen s riziky na projektu?

- Ano
- Ne

3. Myslíte si, že projekt je dostatečně financován?

- Ano
- Ne
- Nevím

4. Bylo na identifikaci a analýzu rizik dostatečně času?

- Ano
- Ne

5. Je cílový uživatel dostatečně zapojen do vývoje projektu?

- Ano
- Ne

6. Byla alespoň nějaká část vyvíjeného programu představena zákazníkovi?

- Ano
- Ne

Příloha D

Testovací scénáře použitelnosti aplikace

1. Firma přijala dva nové zaměstnance – Alana Rigeru na pozici projektového manažera a Betty Rocky jako členku řešitelského týmu. Je nutné jim vytvořit účet, aby mohli aplikaci využívat. Tuto akci může provádět pouze administrátor.
2. Firma získala zakázku na nový projekt se zaměřením na záchranu velkých kočkovitých šelem. Jako administrátor je tvou povinností tento projekt vytvořit v nástroji a tím ho uložit do databáze. Název projektu je Kočkovité šelmy a je popsán jako projekt vytvoření informačního systému pro záchranou stanici velkých kočkovitých šelem. Začátek prací na projektu byl stanoven na 20.5.2020 a konec se předpokládá někdy na druhou polovinu roku 2022. Nově přijatí zaměstnanci (Alan Riger a Betty Rocky) jsou prvními členy řešitelského týmu a jsou přiřazeni ke svým odpovídajícím pozicím.
3. Pro projekt Kočkovité šelmy je nutné vytvořit nový dotazník, který může pomoci řešitelům projektu identifikovat rizika, která se mohou vyskytnout na tomto projektu. Před tvorbou dotazníku je vhodné zkontrolovat, zda v nástroji existují požadované otázky, které má dotazník obsahovat.
4. Jako manažer (Alan Riger) ti byl přiřazen nový projekt, který máš řídit. Projekt má pracovní název Kočkovité šelmy. Nově bylo ustanoveno, že datum ukončení bude 30.9.2022. Zároveň ti bylo sděleno, že Betty Rocky byla přesunuta do jiného projektu a jako náhrada za ní je přidělena Diana Laris. Změň informace na projektu, aby odpovídaly novým skutečnostem.
5. Řešitelský tým projektu Kočkovité šelmy identifikoval dvě nová rizika. Riziko Překročení rozpočtu z kategorie Finance je již známé a je již uloženo v centrálním registru rizik. Projekt je ale dobře financován, takže i když má střední pravděpodobnost výskytu, tak bude mít na projekt pouze minimální dopad. Druhé riziko tým identifikoval poprvé – Pandemie infekční nemoci. Z důvodu situace ve světě je velká pravděpodobnost, že riziko zastihne i tento projekt, přičemž bude mít nejvyšší úroveň dopadu. Z tohoto důvodu by riziko mělo mít nejvyšší prioritu dalšího řešení. Jelikož se první riziko týká financí je vhodné, aby ho měl na starost manažer projektu. U druhého rizika může být správcem kdokoliv.
6. Jako manažer se chystáš vytvořit SWOT analýzu projektu Kočkovité šelmy. Pro jednodušší práci se můžeš inspirovat jinými projekty v systému.

7. Bylo ti oznámeno, že dotazník k projektu Kočkovité šelmy je připraven. Přiřadit ho k tomuto projektu, aby ostatní řešitelé mohli na něho odpovídat. Tuto akci proved jako manažer daného projektu.
8. Jednou z povinností řádového člena týmu je zodpovědět dotazník, který byl přiřazen k projektu, na kterém daný člen pracuje. Ujisti se, že jsi na něho zodpověděl a pokud ne, tak ho zodpověz.
9. Zajímá tě, jak ostatní uživatelé odpověděli na dotazník. Zjisti si tuto informaci.
10. Jsi ve tmavé místnosti a bílé pozadí aplikace dráždí tvé oči. Pokus se tento problém v aplikaci vyřešit.
11. Dostal jsi za úkol přepsat z papíru několik rizik do centrálního registru – proved to.
12. Projekt Kočkovité šelmy skončil neúspěchem. Firma nechce uchovávat jeho data. Jako administrátor jsi dostal za úkol tento projekt ze systému odstranit.

Číslo scénáře:	1
Oblast:	Správa uživatelů, tvorba uživatelského účtu.
Vstupní podmínky:	1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel není přihlášený.
Očekávaný postup:	1. Přihlásit se jako administrátor. 2. V menu kliknout na položku <i>Uživatelé</i> . 3. V tabulce <i>Uživatelé</i> kliknout na ikonové tlačítko s popiskem <i>Přidat</i> , vyplnit požadované položky v dialogu a stisknutím tlačítka <i>Uložit</i> odeslat data na server. Zopakovat i pro druhého uživatele.
Očekávaný výsledek:	1. V tabulce <i>Uživatelé</i> lze najít nově přidané záznamy.

Tabulka D.1: Podrobnosti pro scénář 1

Číslo scénáře:	2
Oblast:	Tvorba nového projektu.
Vstupní podmínky:	1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený jako administrátor. 3. Byl proveden předcházející scénář.
Očekávaný postup:	1. V menu kliknout na položku <i>Projekty</i> . 2. V pravém horním rohu kliknout na ikonové tlačítko s popiskem <i>Přidat</i> , vyplnit požadované položky a stisknutím tlačítka <i>Odeslat</i> odeslat data na server.
Očekávaný výsledek:	1. Uživatel je po odeslání dat přesměrován na stránku nově vytvořeného projektu. 2. Na stránce projektu lze nalézt informace zadané při tvorbě.

Tabulka D.2: Podrobnosti pro scénář 2

Číslo scénáře:	3
Oblast:	Správa otázek, tvorba otázky, tvorba dotazníku.
Vstupní podmínky:	<ol style="list-style-type: none"> 1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený jako administrátor. 3. Byl proveden předcházející scénář.
Očekávaný postup:	<ol style="list-style-type: none"> 1. V menu kliknout na položku <i>Otázky</i>. 2. V tabulce <i>Otázky</i> pomocí vyhledávacího pole vyhledat požadované otázky. 3. V tabulce <i>Otázky</i> kliknout na ikonové tlačítko s popiskem <i>Přidat</i>, vyplnit požadované položky v dialogu a stisknutím tlačítka <i>Uložit</i> odeslat chybějící otázku na server. 4. V menu kliknout na položku <i>Dotazníky</i>. 5. V pravém horním rohu kliknout na ikonové tlačítko s popiskem <i>Přidat</i> a vyplnit požadované položky. 6. Stisknutím ikonového tlačítka s popiskem <i>Přidat otázky</i> vybrat z dialogu požadované otázky a stisknutím tlačítka <i>Uložit</i> je vložit do dotazníku. 7. Stisknutím ikonového tlačítka s popiskem <i>Potvrdit</i> uložit dotazník na server.
Očekávaný výsledek:	<ol style="list-style-type: none"> 1. V tabulce <i>Otázky</i> lze najít nově přidanou otázku. 2. Po odeslání dotazníku je uživatel přesměrován na stránku nově vytvořeného dotazníku. 3. Na stránce dotazníku lze nalézt informace zadané při tvorbě.

Tabulka D.3: Podrobnosti pro scénář 3

Číslo scénáře:	4
Oblast:	Správa projektu, editace základních informací projektu.
Vstupní podmínky:	<ol style="list-style-type: none"> 1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený jako administrátor. 3. Byl proveden předcházející scénář.
Očekávaný postup:	<ol style="list-style-type: none"> 1. Odhlásit se z aplikace a přihlásit se jako Alan Riger – manažer projektu. 2. Na úvodní <i>Dashboard</i> stránce kliknout na kartě požadovaného projektu na tlačítko <i>Přejít k projektu</i>. 3. Na kartě <i>Informace o projektu</i> změnit požadované informace (<i>Datумы projektu</i> a <i>Řešitelé projektu</i>) stisknutím ikonového tlačítka s popiskem <i>Editovat</i>. 4. Stisknutím ikonového tlačítka s popiskem <i>Potvrdit</i> odeslat nová data na server (každá část je samostatně).
Očekávaný výsledek:	<ol style="list-style-type: none"> 1. Na kartě <i>Informace o projektu</i> lze vidět nově zadaná data.

Tabulka D.4: Podrobnosti pro scénář 4

Číslo scénáře:	5
Oblast:	Správa projektu, přidání nového rizika, přiřazení rizika k projektu.
Vstupní podmínky:	<ol style="list-style-type: none"> 1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený jako Alan Riger. 3. Byl proveden předcházející scénář. 4. Uživatel se nachází na stránce požadovaného projektu.
Očekávaný postup:	<ol style="list-style-type: none"> 1. Přejít na kartu <i>Rizika projektu</i>. 2. V tabulce <i>Registr rizik</i> kliknout na ikonové tlačítko s popiskem <i>Přidat</i>. V dialogu kliknutím na tlačítko <i>Centrální registr</i> načíst rizika z centrálního registru. Vybrat požadované riziko, doplnit informace a stisknutím tlačítka <i>Uložit</i> přidělit riziko k projektu. 3. Opět kliknout na ikonové tlačítko s popiskem <i>Přidat</i>. V dialogu kliknout na tlačítko <i>Nové riziko</i> a doplnit informace nového rizika (tvorba rizika). Stisknutím tlačítka <i>Uložit</i> přidělit riziko k projektu a zároveň uložit do centrálního registru.
Očekávaný výsledek:	<ol style="list-style-type: none"> 1. Na kartě <i>Rizika projektu</i> v tabulce <i>Registr rizik</i> lze vidět přiřazená rizika. <i>Matice pravděpodobností a dopadů</i> zobrazuje rizika. Grafy zobrazují statistické informace. 2. Nově tvořené riziko se nachází v <i>Centrálním registru rizik</i>.

Tabulka D.5: Podrobnosti pro scénář 5

Číslo scénáře:	6
Oblast:	Správa projektu, tvorba SWOT.
Vstupní podmínky:	<ol style="list-style-type: none"> 1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený jako Alan Riger. 3. Byl proveden předcházející scénář. 4. Uživatel se nachází na stránce požadovaného projektu.
Očekávaný postup:	<ol style="list-style-type: none"> 1. Přejít na kartu <i>SWOT analýza</i>. 2. Kliknout na tlačítko <i>Přidat SWOT analýzu</i>. 3. Vyplnit požadované položky a stisknutím ikonového tlačítka s popiskem <i>Potvrdit</i> uložit data na server.
Očekávaný výsledek:	<ol style="list-style-type: none"> 1. Na kartě <i>SWOT analýza</i> lze vidět vytvořenou swot tabulku v zobrazovacím režimu.

Tabulka D.6: Podrobnosti pro scénář 6

Číslo scénáře:	7
Oblast:	Správa projektu, přiřazení dotazníku.
Vstupní podmínky:	<ol style="list-style-type: none"> 1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený jako Alan Riger. 3. Byl proveden předcházející scénář. 4. Uživatel se nachází na stránce požadovaného projektu.
Očekávaný postup:	<ol style="list-style-type: none"> 1. Přejít na kartu <i>Dotazník</i>. 2. Kliknout na tlačítko <i>Přidat dotazník</i>. 3. Ve výběrovém dialogu kliknout na požadovaný dotazník, čímž se zobrazí jeho podrobnosti, a následně kliknutím na tlačítko <i>Přiřadit</i> ho přidat k projektu.
Očekávaný výsledek:	<ol style="list-style-type: none"> 1. Na kartě <i>Dotazník</i> lze vidět základní informace dotazníku. 2. Zobrazily se tlačítka na zodpovězení, zobrazení statistiky a smazání dotazníku.

Tabulka D.7: Podrobnosti pro scénář 7

Číslo scénáře:	8
Oblast:	Správa projektu, zodpovězení dotazníku.
Vstupní podmínky:	<ol style="list-style-type: none"> 1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený jako Alan Riger. 3. Byl proveden předcházející scénář.
Očekávaný postup:	<ol style="list-style-type: none"> 1. Odhlásit se z aplikace a přihlásit se jako Diana Laris – člen týmu. 2. Na úvodní <i>Dashboard</i> stránce kliknout na kartě požadovaného projektu na tlačítko <i>Přejít k projektu</i>. 3. Přejít na kartu <i>Dotazník</i>. 4. Kliknutím na tlačítko <i>Zodpovědět</i> zodpovědět na otázky, které se zobrazily v dialogu. 5. Po zodpovězení všech otázek kliknout na tlačítko <i>Odeslat</i>, čímž se odpovědi odešlou na server.
Očekávaný výsledek:	<ol style="list-style-type: none"> 1. Na kartě <i>Dotazník</i> se v části <i>Zodpovězený dotazník</i> změní informace, že uživatel zodpověděl na dotazník. 2. Tlačítko <i>Výsledky</i> je viditelné.

Tabulka D.8: Podrobnosti pro scénář 8

Číslo scénáře:	9
Oblast:	Správa projektu, zobrazení statistiky dotazníku.
Vstupní podmínky:	<ol style="list-style-type: none"> 1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený jako Diana Laris. 3. Byl proveden předcházející scénář. 4. Uživatel se nachází na stránce požadovaného projektu.
Očekávaný postup:	<ol style="list-style-type: none"> 1. Přejít na kartu <i>Dotazník</i>. 2. Kliknutím na tlačítko <i>Výsledky</i> zobrazit statistiku.
Očekávaný výsledek:	<ol style="list-style-type: none"> 1. Na kartě <i>Dotazník</i> je zobrazena celková statistika, jak zodpovědní uživatelé odpovídali na dotazník.

Tabulka D.9: Podrobnosti pro scénář 9

Číslo scénáře:	10
Oblast:	Přepnutí do tmavého režimu.
Vstupní podmínky:	1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený.
Očekávaný postup:	1. V hlavním aplikačním rámci kliknout na ikonové tlačítko s popiskem <i>Tmavý režim</i> .
Očekávaný výsledek:	1. Aplikace se nachází v tmavém režimu.

Tabulka D.10: Podrobnosti pro scénář 10

Číslo scénáře:	11
Oblast:	Správa centrálního registru rizik, tvorba nového rizika.
Vstupní podmínky:	1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený.
Očekávaný postup:	1. V menu kliknout na položku <i>Registr rizik</i> . 2. V tabulce <i>Registr rizik</i> kliknout na ikonové tlačítko s popiskem <i>Přidat</i> , vyplnit požadované položky v dialogu a stisknutím tlačítka <i>Uložit</i> odeslat data na server. V případě potřeby zopakovat.
Očekávaný výsledek:	1. V tabulce <i>Registr rizik</i> lze nalézt nově přidané záznamy.

Tabulka D.11: Podrobnosti pro scénář 11

Číslo scénáře:	12
Oblast:	Správa projektu, smazání projektu.
Vstupní podmínky:	1. Server běží na pozadí, klientská část je načtená v prohlížeči. 2. Uživatel je přihlášený jako administrátor. 3. Byl proveden scénář 10.
Očekávaný postup:	1. V menu kliknout na položku <i>Projekty</i> . 2. V kartě požadovaného projektu kliknout na tlačítko <i>Smazat</i> . 3. V potvrzovacím dialogu kliknout na tlačítko <i>Potvrdit</i> .
Očekávaný výsledek:	1. V přehledu projektů se nenachází karta smazaného projektu 2. Pokus o přístup na adresu projektu skončí chybovým hlášením 404.

Tabulka D.12: Podrobnosti pro scénář 12

Příloha E

Použité knihovny při tvorbě klientské části

Tento seznam, včetně verzí daných knihoven, lze najít v souboru `package.json`.

date-fns, **date-io/date-fns** Date-fns zpřístupňuje metody pro manipulaci s kalendářními daty. Date-io/date-fns pak tyto metody abstrahuje a díky tomu je můžou používat i jiné knihovny.

material-ui/core, **material-ui/icons**, **material-ui/lab**, **material-ui/pickers** Jedná se o různé části z knihovny Material-UI. Core zpřístupňuje všechny komponenty, které tato knihovna obsahuje. Z icons se pak přidávají Material Design ikony od společnosti Google. Lab obsahuje komponenty, které jsou ještě ve fázi testování a z tohoto důvodu ještě nejsou součástí balíčku core. Pickers se používají jako komponenty kalendáře a hodin.

nivo/bar, **nivo/heatmap**, **nivo/pie** Grafové komponenty z knihovny Nivo. Nivo umožňuje stáhnout každý graf jako samostatnou knihovnu, aby výsledný balíček byl co nejmenší. Bar představuje sloupcový graf, pie je koláčovým grafem a modifikovaný heatmap se používá pro zobrazení matice pravděpodobnosti a dopadu.

axios Knihovna, která hlavně umožňuje tvořit, posílat a přijímat HTTP požadavky.

formik Podpůrná knihovna pro jednodušší správu formulářů. Pro validaci používá knihovnu yup

yup Knihovna používá pro validaci dat v nástroji. Lze použít pro validaci formulářů nebo i samostatných dat.

material-table Tato knihovna tvoří komponenty tabulek, které jsou vytvořené ve stylu Material Design. Vnitřně používá komponenty z Material-UI

randomcolor Knihovna pro vygenerování náhodných barev. Používá se pro vybarvení grafů.

react, **react-dom** Jsou to balíčky z knihovny React. Obsahují všechny potřebné komponenty pro vytvoření React aplikace.

react-scripts Jde o balíček skriptů a konfiguračních souborů, které používá Create React App. Create React App se používá pro rychlé a automatické vytvoření výchozí React aplikace.

react-draggable Tvoří komponentu, díky které lze její potomky tzv. uchopit a přenášet. Používá se u některých dialogů.

redux, react-redux Redux umožňuje vytvářet a spravovat globální stav aplikace, react-redux pak zpřístupňuje propojení mezi Reactem a Reduxem.

redux-auth-wrapper Tato knihovna umožňuje definovat přístup k jednotlivým částem aplikace podle role a oprávnění uživatele. Podobně pak omezuje přístup a viditelnost komponent. Není součástí systému Redux, ale využívá informace uložené v globálním stavu.

react-router-dom Propojuje DOM s komponenty knihovny React Router. Díky tomuto lze v aplikaci provádět směrování.

react-promise-tracker Je to knihovna, která se používá pro sledování slibů (*promise*) v aplikaci. Díky tomuto je možné zobrazovat načítací obrázek v době, kdy se načítají data ze serveru.

react-swipeable-views Knihovna zpřístupňující komponentu pro plynulejší posouvání mezi stránkami na stejné úrovni. Je použita při odpovídání na dotazník projektu.

Příloha F

Návod ke zprovoznění aplikace

Všechny příkazy, pro jednodušší kopírování, lze také najít v souboru README. Předpokládá se spuštění na systému Windows 10 či nějakém ekvivalentním operačním systému.

F.1 Server

Server potřebuje pro svůj běh mít nainstalovanou Javu. Serverová část byla primárně tvořena na verzi 1.8.0_231. Java se musí nacházet v PATH proměnné.

1. V příkazovém řádku najít zdrojový adresář serveru – adresář **server**.
2. Překlad a vytvoření jar souboru
 - (a) Překlad a spuštění automatizovaných testů před vytvořením jar souboru:

```
mvnw.cmd clean package
```

- (b) Vytvoření jar souboru bez překladu testů:

```
mvnw.cmd -Dmaven.test.skip clean package
```

- (c) Překlad zdrojových souborů i testů, ale přeskočení spuštění testů před vytvořením jar souboru aplikace:

```
mvnw.cmd -DskipTests clean package
```

3. Jar soubor aplikace se nachází v adresáři target pod názvem **spravarizik-1.0.jar**. Spuštěná aplikace poběží na **localhost:8080**. Ze zdrojového adresáře lze spustit s:

```
java -Dfile.encoding=UTF-8 -jar target\spravarizik-1.0.jar
```

4. Ukončení aplikace: **ctrl + c**.

F.2 Klient

Pro běh klienta na lokálním stroji je nutné mít nainstalovaný Node.js, ten v sobě již obsahuje správce balíčků npm, který je potřeba pro stažení závislostí a spuštění klienta. Klientská část byla primárně tvořena na node ve verzi 12.15.0 s npm ve verzi 6.13.7.

1. V příkazovém řádku najít zdrojový adresář klientské části – adresář **klient**.
2. Stažení potřebných závislostí: `npm i`.
3. Spuštěná aplikace poběží v prohlížeči na adrese **localhost:3000**. Je vhodné zde zmínit, že klient očekává, že server poběží na portu 8080 a podobně server očekává klientskou část na portu 3000. Spustit ji lze pomocí: `npm start`.
4. Vytvoření balíčku, které lze následně nahrát na nějaký webový server: `npm build`.
5. Ukončení aplikace: po stisknutí `ctrl + c` se zobrazí nabídka k ukončení.

F.3 Přihlašovací údaje

ROLE	LOGIN	HESLO	JMÉNO
Administrátor	xlogin00	qwe	Login Login
Manažer	xrigger00	abc	Alan Riger
Manažer	xhodso00	abc	Bianca Hodson
Člen týmu	xlaris00	123	Diana Laris
Člen týmu	xnovak00	123	Casper Novak

Tabulka F.1: Přihlašovací údaje pro vytvořené účty

Příloha G

Obsah paměťového média

- `README` - Základní popis včetně návodu na zprovoznění aplikace
- `source` - adresář se zdrojovými kódy
- `source/client` - adresář se zdrojovými kódy klientské části
- `source/server` - adresář se zdrojovými kódy serverové části
- `xskuto00.pdf` - tento dokument
- `latex` - adresář se zdrojovými $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ soubory tohoto dokumentu