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 SPRÁVU A VIZUALIZACI RIZIK V MANAGEMENTU PROJEKTŮ

PROJECT MANAGEMENT AND RISKS VISUALIZATION SUPPORT TOOL

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. KATEŘINA PŘIBYLOVÁ**
AUTHOR

VEDOUCÍ PRÁCE **doc. RNDr. JITKA KRESLÍKOVÁ, CSc.**
SUPERVISOR

BRNO 2018

Abstrakt

Tato diplomová práce se zabývá problematikou řízení rizik a vysvětluje jeho důležitost v rámci projektového managementu v oblasti IT. Postupně popisuje všechny fáze životního cyklu řízení rizik a také metody a postupy používané během těchto fází. Dále se práce zaměřuje na rozhodovací analýzu, konkrétně na metody rozhodovacích stromů a simulace Monte Carlo. Následující část obsahuje vypracovaný návrh systému pro správu a vizualizaci rizik, na základě kterého byl zhotoven prototyp. Výsledná aplikace byla naprogramována v PHP frameworku Laravel. Implementací a testováním aplikace se zabývá další část této práce. Na závěr je uvedeno zhodnocení, kde jsou diskutována další možná rozšíření výsledné aplikace.

Abstract

This master thesis deals with the topic of risk management and explains its importance during project management in IT projects. It descibes every phase of the risk management life cycle and also methods and procedures used in each phase. After that the thesis focuses on decision analysis, mainly on decision trees and Monte Carlo simulation. The last part contains the design of an application for risk management and visualization. Prototype of this system has been implemented based on this design in PHP framework Laravel. Details of the implementation and testing are in the next part of this thesis. In the end there is an assessment with discussion of possible expansions.

Klíčová slova

rizika, rizikový management, projektový management, rozhodovací analýza, rozhodovací stromy, simulace Monte Carlo, PHP, Laravel framework, webová aplikace

Keywords

risks, risk management, project management, decision analysis, decision trees, simulation Monte Carlo, PHP, Laravel framework, web application

Citace

PŘIBYLOVÁ, Kateřina. *Nástroj pro správu a vizualizaci rizik v managementu projektů*. Brno, 2018. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. RNDr. Jitka Kreslíková, CSc.

Nástroj pro správu a vizualizaci rizik v managementu projektů

Prohlášení

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

.....
Kateřina Přibylová
15. května 2018

Poděkování

Ráda bych poděkovala vedoucí mé práce doc. RNDr. Jitky Kreslíkové, CSc. za odborné vedení, konzultace a cenné rady, které mi pomohly při řešení této diplomové práce. Dále bych chtěla poděkovat všem kolegům a odborníkům z praxe, kteří byli nápomocni při testování a poskytli mi hodnotnou zpětnou vazbu.

Obsah

Kapitola 1

Úvod

Existuje něco důležitějšího pro úspěšnost projektu, než dobrá rozhodnutí? Rozhodovat se v problémových situacích a předcházet těmto situacím je jednou z klíčových činností každého projektového manažera. Ve své diplomové práci jsem se proto rozhodla zabývat riziky a s nimi související rozhodovací analýzou, se kterou jsem se již setkala na zahraničním studijním pobytu. Každý z nás denně dělá spousty rozhodnutí, z nichž některá s sebou nesou menší či větší rizika. Naštěstí pravděpodobnost se statistikou nám jsou schopny ulehčit rozhodovací proces a vybrat nejlepší řešení.

Nejprve je v kapitole 2 čtenář seznámen s riziky, pojmem řízení rizik a jeho souvislostí s projektovým řízením. Konkrétněji se poté kapitola zaměřuje na jednotlivé fáze životního cyklu řízení rizik, tedy na plánování, identifikaci a s ní související klasifikaci, analýzu, reakci na rizika a jejich kontrolu.

V další kapitole 3 je věnována pozornost rozhodovací analýze. Jsou zde rozebrány nejčastěji používané modely nejenom při analýze rizik, konkrétně rozhodovací stromy, statistické početní metody, simulace Monte Carlo a analýzy citlivosti spolu s bodovacím modelem.

Následující kapitola 4 se zabývá analýzou existujících řešení, kde byly vybrány tři nejlépe hodnocené systémy. Dále je zde zadefinována specifikace požadavků pro navrhovaný systém a návrh uživatelského rozhraní. Návrh staví na znalostech získaných během předchozího studia.

Předchozí kapitoly byly vypracovány v rámci semestrálního projektu, na který tato diplomová práce navazuje.

Kapitola 5 se zaměřuje na samotnou realizaci prototypu navrženého systému. Nejprve je proveden rozbor aktuálně nejpoužívanějších PHP aplikačních rámců a je zde zdůvodněno, proč byl pro implementaci vybrán právě framework Laravel. Je zde rovněž objasněno, jak tento framework funguje a jaké jsou jeho hlavní výhody. Poté je pozornost věnována již samotné implementaci aplikace. Implementační část je rozdělena do ucelených částí, které kopírují postupný průchod aplikace uživatelem.

Testováním aplikace se zabývá kapitola 6. Zde je popsáno, jak probíhalo testování výsledného prototypu, a to nejenom testování funkčnosti samotného prototypu, testování použitelnosti, ale i uživatelské testování zaměřené na možnost využití aplikace v odvětví projektového managementu.

Poslední kapitola 7 hodnotí výsledek celé práce a diskutuje další možnosti budoucího rozšíření v případě komerčního využití.

Kapitola 2

Rizika v managementu projektů

Rozhodování v projektovém managementu neznamená pouze rozhodnutí, zda projekt uskutečnit, či ho zastavit hned v začátku. Rozhodování provází celý životní cyklus daného projektu a ve velké míře ovlivňuje jeho cenu, čas a kvalitu. Nejprve je třeba si zadefinovat základní pojmy týkající se projektového managementu. Následně je blíže specifikována problematika řízení rizik. Poslední část se zabývá detailně jednotlivými částmi cyklu řízení rizik, tedy plánováním, identifikací, analýzou, zvládnutím a kontrolou rizik.

2.1 Projektový management

Řízení rizik je jednou ze znalostních oblastí projektového managementu, neboli řízení projektů. Z tohoto důvodu je mu v následující sekci věnována pozornost a jsou definovány dále používané pojmy.

Projekt je časově ohraničená a ucelená sada činností a procesů, jejímž cílem je zavedení, vytvoření nebo změna něčeho konkrétního v požadované kvalitě, za danou cenu a v daném čase [12]. Projekt je třeba řídit a je charakterizován typickými znaky:

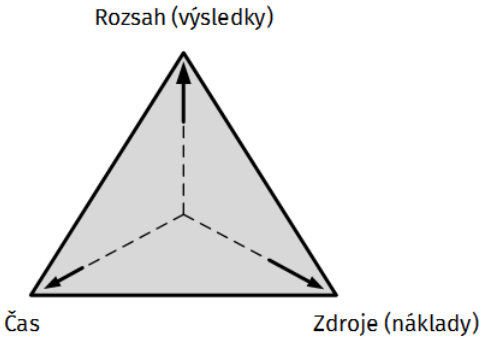
- Projekt musí mít jasný cíl, výsledek či užitek, tedy něco, co má realizovat, vytvořit či změnit.
- Projekt je časově omezený sled činností, obvykle v řádu měsíců.
- Projekt je jedinečný. Jedná se o neopakovatelný, unikátní sled činností, který vyžaduje specifický způsob řízení.

Z předchozí definice vyplývá, že projekt má dočasný charakter, tedy i lidské, materiální a finanční zdroje jsou vymezeny začátkem a koncem projektu. Nejznámější mezinárodní metodiky, a certifikace, pro řízení projektů jsou PMI (Project Management Institute), IPMA (International Project Management Association) a PRINCE2 (AXELOS Limited). Standard, který se vztahuje k projektovému řízení je definován v ISO 21500 Management projektu. Všechny se ale zhruba shodují na základních fázích životního cyklu projektu:

1. zahájení,
2. plánování,
3. koordinace,
4. kontrola,

5. uzavření.

Účelem **projektového řízení** je zajistit efektivní řízení činností tak, aby přinesly předpokládaný výsledek v předpokládaném čase a za předpokládané náklady (tzv. trojimperativ viz 2.1). Výsledkem projektového řízení je dokončený projekt.



Obrázek 2.1: Trojimperativ[1]

2.1.1 Agilní přístup

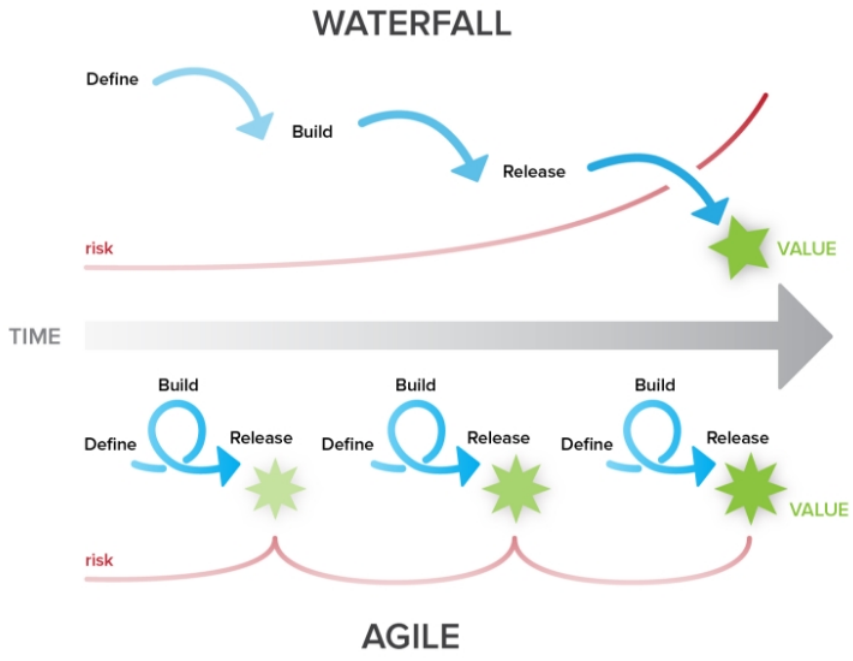
Pro projekty, kde je výstupem software, je situace komplikovanější. Nelze se pevně držet tradiční koncepce zaměřené na úkoly, jejímž motivem je posloupnost konkrétních úkolů. Tato koncepce nachází uplatnění u projektů s nízkými riziky a dobře známým návrhem. Zatímco protichůdný přístup zaměřený na hodnotu se namísto plnění daných úkolů soustřeďuje na zvyšování hodnoty výsledného produktu. Vstupem projektu není pevná zásoba úkolů, ale proměnlivý tok [9]. Z tohoto důvodu se během posledních let využívají různé agilní metodiky, a to nejenom pro vývoj, ale pro celé řízení projektů. Agilním řízení projektů se zaměřuje na včasné dodání prototypů, průběžné vylepšování (continuous improvement) produktu i procesů a flexibilitu rozsahu projektu, tedy pozitivní přístup ke změnám specifikace a ke komunikaci se zákazníkem. Tím je zajištěno postupné navyšování obchodní hodnoty projektu. Dále se řízení orientuje na tým a komunikaci v týmu a v neposlední řadě na dodání řádně otestovaného produktu, který reflektuje požadavky zákazníka [12]. Rozdíl mezi agilním a klasickým vodopádovým přístupem lze vidět na obrázku 2.2.

2.2 Rizika

Dle C.R. Pandiana [15] riziko, ačkoliv tradičně vnímáno jako negativní, neznamená nutně potenciální problém. Riziko je nejistota v projektu, která může mít záporný nebo kladný vliv na splnění cílů daného projektu [17]. V projektovém managementu pozitivní rizika vnímáme jako příležitosti a negativní naopak jako hrozby. O toto rozdělení se rovněž opírá SWOT analýza 2.3.3, která bude později probrána. Riziko je tedy pravděpodobnost utrpení ztráty kvůli nepředvídatelným faktorům za účelem splnění cílů [15]. Riziko vyjadřuje určitou míru nejistoty, tedy pravděpodobnost dosažení výsledku, který je rozdílný od očekávaného [14].

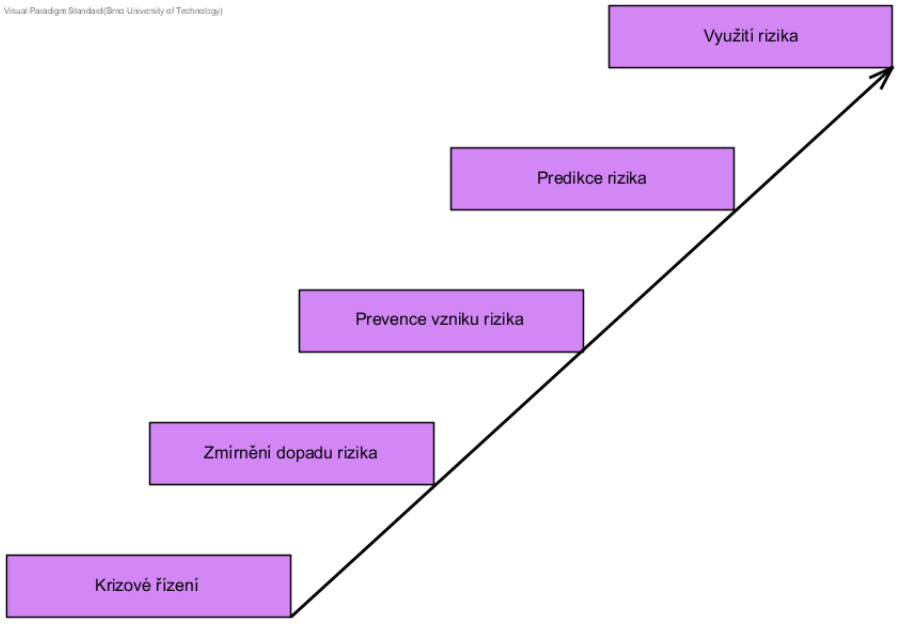
[1] Převzato z http://www.pmconsulting.cz/pm-wiki/trojimperativ-projektu/

[2] Převzato z https://www.cirdangroup.com/cirdan-blog/2017/8/30/waterfall-vs-agile-some-differences-to-keep-in-mind



Obrázek 2.2: Rozdíl mezi vodopádovým a agilním řízením[2]

Je důležité si pojem rizik neplést s podobnými pojmy. Závady a chyby jsou výsledkem pochybení a jsou nalezeny inspekcí a testováním, zatímco problémy odráží rozdílný výsledek od plánu. Všechny tyto pojmy se týkají minulosti a uskutečněných skutečností. Naopak rizika řeší budoucnost a jejich řešením klesá počet výsledných chyb a problémů.



Obrázek 2.3: Fáze zralosti rizikového řízení [Inspirováno v [15]]

V obrázku 2.3 jsou zobrazeny fáze zralosti rizikového řízení. V nejhorším případě je rizikové řízení z projektového řízení vynecháno a řeší se až nastalé problémy, tato varianta může mít katastrofické následky pro projekt. Ideálním stavem je prevence a predikce rizik. Pomyslným nejvyšším bodem je poté umět rizika obrátit v příležitosti a využít je v náš prospěch.

Nejčastější zdroje rizik v projektech z informačních technologií se dají rozdělit do následujících kategorií [17].

Tržní rizika

Do této kategorie spadají rizika, která se zabývají tím, zda bude vyvíjená služba, či produkt přínosný, zda bude mít šanci se uplatnit na trhu u potenciálních zákazníků, nebo do jaké míry ho ohrožuje konkurence.

Finanční rizika

Mezi finanční rizika patří nedostatečné finanční zajištění projektu, špatně provedené finanční odhady, nedostatečná návratnost investic a doba návratnosti.

Technologická rizika

Při identifikaci rizik z této kategorie je třeba se ptát, zda je projekt technologicky zvládnutelný, zda budou použity moderní technologie, zda budou softwarové, hardwarové a síťové komponenty v projektu správně fungovat nebo zda technologie v průběhu vývoje nezastará.

Lidská rizika

Dostatek pracovníků s odpovídajícími schopnostmi, dostatečné manažerské a odborné schopnosti, podpora vyššího vedení nebo nedostatečná znalost zákazníka a špatné vztahy s ním, to jsou témata, která ovlivňují lidská rizika.

Strukturální a procesní rizika

Sem spadají rizika, která řeší změny v obchodních a organizačních procesech, nebo počet dalších systémů, se kterými musí nový systém komunikovat.

2.3 Řízení rizik

Řízení rizik je dnes důležitou součástí životního cyklu softwarového projektu, bez které by projekt nemusel uspět. Protože každý projekt dnes provází rizika a o to více v případě softwarových projektů, které jsou na rizika mnohem více náchylnější. Při každé definici našich cílů si musíme určit i rizika, které tento cíl obnáší. Pak záleží už na rozhodnutí projektového manažera, zda je projekt dostatečně ziskový, aby si mohl dovolit více riskovat, či nikoliv. Cílem řízení rizik není pouze rizika eliminovat, ale odhalit je a snížit co nejvíce škody, které mohou způsobit. Provádí se v průběhu celého životního cyklu projektu.

Řízení rizik je oblast řízení zaměřující se na analýzu a snížení rizika, pomocí různých metod a technik prevence rizik, které eliminují existující nebo odhalují budoucí faktory zvyšující riziko. Je to soustavná, opakující se sada navzájem provázaných činností, jejichž cílem je řídit potenciální rizika, a tím omezit pravděpodobnost jejich výskytu nebo snížit jejich dopad. Účelem je předejití problémům a vyhnutí se krizovému řízení [14].

Rozlišujeme dva základní modely řízení rizik, jedním je dle C. R. Pandiana [15] Organic risk management process, tedy neformálně přeloženo přírodní řízení rizik, kdy je hlavním stavebním kamenem lidská intuice. Nervový systém lidského těla je schopen během setin sekundy analyzovat riziko a naše instinkty jsou rovněž velmi rychle schopny situaci vyhodnotit a patřičně zareagovat. Jedná se o SAT (Sense-Act-Take) metodu. Organic risk management využívá lidského svědomí a kreativity k tomu, aby odhalil rizika a rychle na ně zareagoval. Tato metoda je velmi vhodná při rozhodování o nenadálých katastrofických rizicích.

Druhým modelem je Goal Selection, tedy výběr cíle. Tato metoda je založena na redefinici cílů tak, aby znamenaly pro projekt co nejmenší riziko. Redefinice probíhá na všech úrovních, kde jednou z používaných metod jsou milníky. Čím více milníků se v projektu ustanoví, tím se sníží počet rizik a zlepší se obecná průhlednost celého projektu.

Na základě těchto dvou základních modelů je odvozen nespočet dalších. Liší se především ve specifikaci jednotlivých kroků v rámci řízení a v míře schopnosti rychle reagovat. Příkladem je minimální rizikové řízení, kdy se po odhalení rizika tento fakt pouze oznámí zainteresovaným stranám a tím je delegována zodpovědnost za rozhodování a reakce na jiný tým.

Základní fáze řízení rizik se liší v závislosti na modelu, respektive standardu, který pro řízení vybereme. Obecně by se ovšem dal životní cyklus řízení rizik shrnout do následujících pěti bodů:

1. plánování řízení rizik,
2. identifikace rizik,
3. kvantitativní a kvalitativní analýza rizik,
4. zvládnutí/zmírnění rizik a plánování reakcí na rizika,
5. sledování a kontrola rizik.

2.3.1 Vliv agilních metodologií na rizikové řízení

Jaký vliv má na řízení rizik změna z klasických na agilní metodiky? V agilních projektech nemusí řízení rizik obsahovat obsáhlou dokumentaci rizik a speciální mítinky. Agilní framework[3] namísto toho přímo obsahuje řízení rizik. Díky agilním principům je zajištěno snížení rizik, která často vznikala během klasického řízení. O jakých principech je řeč? Například, že nejvyšší prioritou je uspokojit zákazníka včasným a průběžným dodáváním hodnotného softwaru, vítat změny v specifikaci požadavků, které vedou k vyšší spokojenosti zákazníka, dodávat zákazníkovi průběžný výsledek často a zaměřovat se hlavně na komunikaci, a to nejenom týmovou, ale i s ostatními zainteresovanými stranami [12].

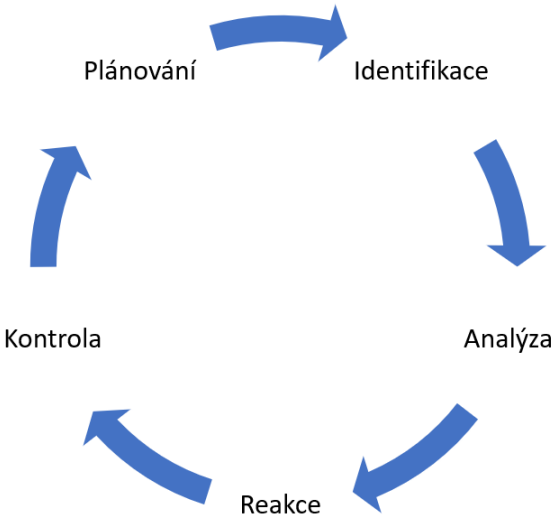
Jak je vidět z obrázku 2.5, tak s postupujícím vývojem projektu klesá pravděpodobnost rizik. Obzvlášť se eliminuje katastrofální riziko, že projekt nebude splňovat zákazníkovy požadavky. Vývoj ve sprintech zajišťuje krátký časový rozdíl mezi investicí a důkazem, že projekt funguje. Agilní mítinky (hlavně review[5] a retrospective[6]) zajišťují časté doručení

[3]Framework je softwarová struktura, která slouží jako podpora při programování, vývoji a organizaci jiných softwarových projektů. Může obsahovat podpůrné programy, knihovny API (Application Programming Interface), podporu pro návrhové vzory nebo doporučené postupy při vývoji.

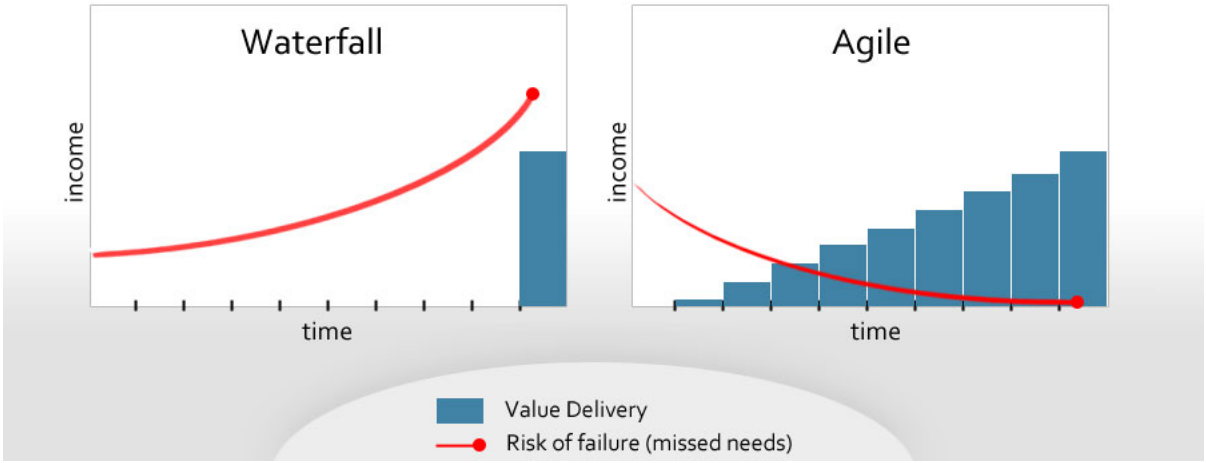
[4]Převzato z https://www.scrum.as/academy.php?show=2&chapter=4

[5]Zhodnocení sprintu.

[6]Ohlédnutí se a navrhnutí změn.



Obrázek 2.4: Cyklus řízení rizik [zdroj vlastní]



Obrázek 2.5: Rozdíl mezi vodopádovým a agilním řízením [4]

zpětné vazby vývojářskému týmu. Tato zpětná vazba působí jako preventivní opatření proti odchylkám ze zákazníkových představ. Mezi tři významné faktory, které ovlivňují redukci rizik patří Definition of Done, rychlé selhání a self-funding projekty [12].

Definition of Done

Tato metoda se využívá jako potvrzení, že je požadavek splněn a může být předveden zákazníkovi. Metoda je založená na tom, že požadavek nejprve musí splňovat týmem nastavené DoD, tedy definici splnění. Většinou tato definice zahrnuje to, že je požadavek

plně naprogramován, otestován, integrován (funguje správně s ostatními komponentami) a zdokumentován. Zároveň s tím je vypracován i seznam akceptovatelných rizik, například může být nadbytečné po každém sprintu dělat výkonnostní testy. Díky této metodě dostává zákazník každý sprint funkční prototyp produktu a tím se výrazně sníží rizika.

Self-funding projekty

Agilní projekty mohou do značné míry snížit i finanční rizika, příkladem jsou například projekty, kdy se produkt vydává postupně a může začít generovat zisk ještě v průběhu svého vývoje a tím získávat kapitál na svůj další vývoj. Tento princip má i několik dalších výhod, hodí se v případech, kdy není dostatečný počáteční kapitál na financování projektu, a je rovněž velmi užitečný, jako zábrana proti zrušení projektu při nějakých krizových situacích.

Rychlé selhání

Poslední z agilních principů, které ovlivňují rizika je rychlé selhání (Failing fast). Díky častému testování je odhalení chyb včasné, v řádu několika sprintů, a to včetně chyb v specifikaci, které mohou u normálních modelů vést k velikým ztrátám a k neúspěšnému ukončení projektu. Tento princip není nutně použitelný pouze na testování softwaru, ale funguje i jako test, zda by měl daný produkt na trhu odbyt. Prototyp, který vznikne po prvních dvou iteracích se nabídne potenciálním zákazníkům a v případě neúspěchu jsou ztráty na zrušení projektu několikanásobně nižší, než kdyby se na tuto prvotní chybu ve vizi projektu přišlo až po jeho naplánovaném ukončení.

Nástroje pro správu rizik v agilním prostředí

Ačkoliv je v agilním prostředí pravděpodobnost vzniku rizika nižší, tak není dobré tuto skutečnost zanedbávat a řízení rizik by stále mělo být součástí celého projektového řízení. Protože agilní metodiky propagují samosprávné týmy, tak jsou týmy zodpovědné za identifikaci rizik, za snahu zabránit jejich uskutečnění a za patřičnou reakci v návaznosti na jejich uskutečnění. V agilním prostředí by měly vždy mít prioritu ty nejvíce hodnototvorné a nejvíce rizikové požadavky. Namísto nákladné identifikace a dokumentace možných rizik se využívá již existujících agilních nástrojů, jako jsou vize projektu, plány (roadmaps), produktový a sprintový seznam nevyřízených úkolů (backlog), tabulce úkolů (task board) a scrum ceremonie, tedy plánování, zhodnocení (review), stand-up a retrospektivy. Nejdůležitějším prvkem je ovšem pravidelná komunikace v týmu, bez které by všechny předchozí body nefungovaly.

Z výše uvedeného vyplývá, že řízení rizik je třeba integrovat do ostatních částí řízení projektu, neměl by to být jeden samostatný proces, ale řízení rizik by mělo být součástí všech procesů. Tato integrace by měla být postupná a co nejjednodušší.

2.3.2 Plánování rizik

V rámci plánování volíme nejlepší strategii pro vybraný projekt či firmu. Jak již bylo zmíněno, možných modelů řízení existuje několik, proto bychom vždy měli najít ten nejvhodnější model pro naši situaci. Během této fáze specifikujeme, jak bude náš konkrétní proces řízení rizik fungovat a v případě agilního vývoje plánujeme integraci tohoto procesu do ostatních procesů. Je rovněž třeba určit, jakým způsobem budeme rizika identifikovat a analyzovat, jakou klasifikaci pro rizika vybereme a jak je budeme sledovat. Při plánování

provádí tým revize stanovení rozsahu projektu, plánu řízení projektu a organizačních aktiv organizace [17]. Hlavním výstupem je plán řízení rizik.

2.3.3 Identifikace rizik

Detekce a identifikování má dva základní kroky, prvně je potřeba rizika najít a poté je rozpoznat, klasifikovat a odhadnout jejich možné důsledky. Výstupem identifikace rizik je registr rizik. Registr rizik by měl být jednoduchý mechanismus pro zachycení všech dat souvisejících s riziky [11]. Obvyklými zdroji rizik jsou změny požadavků, nedostatek personálu, špatné odhady, nerealistický rozpočet, špatná funkcionalita, nevyhovující uživatelské rozhraní, špatně definované odpovědnosti nebo chyby v dokumentaci projektu. Pro identifikace rizik existují následující metody [18]:

- Techniky založené na způsobu myšlení:
  - mind mapping,
  - brainstorming,
  - myšlení Out of the box.
- Techniky založené na analýze:
  - analýza pěti sil 5F,
  - PESTLE analýza,
  - SWOT analýza,
  - VRIO analýza,
  - SMART návrh cílů.
- Další techniky:
  - top 10 rizik,
  - kontrolní seznamy rizik,
  - dotazníky,
  - technika scénářů.

Mind mapping

Tato metoda využívá toho, že mysl mapuje známé symptomy na budoucí problémy. Této metodě nelze přiřadit nějaký lineární logický rámec, protože je založena čistě na lidském faktoru. Využívá se intuice a zkušeností projektového manažera. Díky jeho intuici a zkušenostem je schopen včas odhalit, nebo předvídat možné problémy. Výstupem je forma diagramu, který reprezentuje myšlenky, aktivity, rizika a další věci související s hlavním tématem.

Brainstorming

Nejdůležitějším specifikem této metody je skupinové myšlení. Tato metoda je založená na skupinové diskuzi, je to kreativní technika, během které se generují nápady na dané téma. V případě identifikace rizik se například vede diskuze nad plánovacími dokumenty projektu. Důležitými podmínkami této metody je, že si jsou všichni účastníci rovni, že veškeré nápady jsou vítány a že v brainstormingu není prostor pro kritiku. Výstupem může být rovněž mind mapa.

Myšlení Out of the Box

Tato metoda využívá techniky zvané v překladu jako „myšlení mimo rámec“. Je potřeba se na danou věc podívat z jiného úhlu a odklonit se od svých zvyků. Stereotypy a určitá familiárnost mají ve zvyku člověku zkreslit jeho pohled a i zřejmé problémy mohou být poté pro člověka neviditelné.

Analýza pěti sil

Porterova analýza pěti sil (5F) je způsob analýzy odvětví a jeho rizik. Prognózuje se vývoj stávajících a potenciálních konkurentů, dodavatelů, zákazníků a substitutů. Rizika identifikovaná touto metodou budou ve většině případů externí.

PESTLE analýza

PESTLE analýza je technika používaná pro strategickou analýzu okolního prostředí. Vnější faktory, které zkoumá, jsou politické, ekonomické, sociální, technologické, legislativní a ekologické. Analýzou rovněž identifikujeme externí rizika, které ovlivňují celou organizaci.

SWOT analýza

Analýza SWOT je technika zaměřená na zhodnocení vnitřních a vnějších faktorů ovlivňujících úspěšnost organizace. Analyzuje silné a slabé stránky firmy a zároveň její příležitosti a možné hrozby.

VRIO analýza

VRIO je analytická technika pro hodnocení zdrojů firmy, a to nejenom finančních zdrojů, ale i lidských a materiálních. Dimenzemi hodnocení jsou hodnota, vzácnost, napodobitelnost a organizace zdroje.

SMART

Poslední analytickou technikou je SMART. Tato technika slouží pro navrhování cílů během plánování. Jednotlivé cíle musí splňovat SMART podmínky, tedy musí být specifické, měřitelné, dosažitelné, realistické a časově specifické.

Top 10 rizik

Tato metoda staví na historických zkušenostech. Proč „vynalézat kolo“, když již bylo vynalezeno. Existuje několik seznamů, které obsahují nejčastější rizika, případně oblasti, na které se zaměřit. Projitím těchto seznamů bychom měli být schopni zkontrolovat konkrétní části našeho projektu a vyvarovat se nejčastějším rizikům.

- Autoři známých seznamů:
  - Caper Johnn
  - Rex Black
  - Brian A. Will
  - Barry Boehm

– Chester Summer

V příloze A je jako příklad uvedena šablona pro identifikaci rizik v softwarových projektech.

Kontrolní seznamy a dotazníky

S předchozí metodou úzce souvisí kontrolní seznam rizik. Kontrolní seznam je složen z nápověd, symptomů nebo z názvů rizik a slouží jako průvodce při hledání rizik. Seznam může být doprovázen dotazníky, které pomocí otázek pomáhají s identifikací rizik.

Technika scénářů

Podstatou této techniky je promyšlení možných scénářů pro případ určité události. Při budování těchto scénářů lze identifikovat možná rizika dané situace a zároveň vymyslet patřičná opatření.

2.3.4 Klasifikace rizik

Pro přehlednost a jednoduchost je zapotřebí identifikovaná rizika klasifikovat a určit jejich atributy, tedy jejich charakteristické vlastnosti. Každé riziko má název, svoje unikátní identifikační číslo, pravděpodobnost(P) s kterou může nastat, míru dopadu(I), pokud riziko nastane, míra vystavení riziku, tedy kombinaci pravděpodobnosti a míry dopadu, tzv. P-I index, původ rizika, kategorii a vlastníka rizika, který odpovídá za proces, které toto riziko ohrožuje.

Příklad klasifikace rizik:

- identifikační číslo,
- míra pravděpodobnosti:
  - téměř jistá,
  - velmi pravděpodobná,
  - pravděpodobná,
  - málo pravděpodobná,
  - nepravděpodobná.
- míra dopadu,
- povaha – viz obrázek 2.6:
  - Katastrofická – sem spadají rizika, která mohou nejvíce uškodit.
  - Omezující – neboli překážky, tato rizika s 100% pravděpodobnstí nastanou.
  - Nominální – standardní míra vystavení riziku.
  - Triviální – nepodstatná rizika.
- vystavení riziku,
- typ rizika - technické, obchodní, rizika řízení projektu, ...

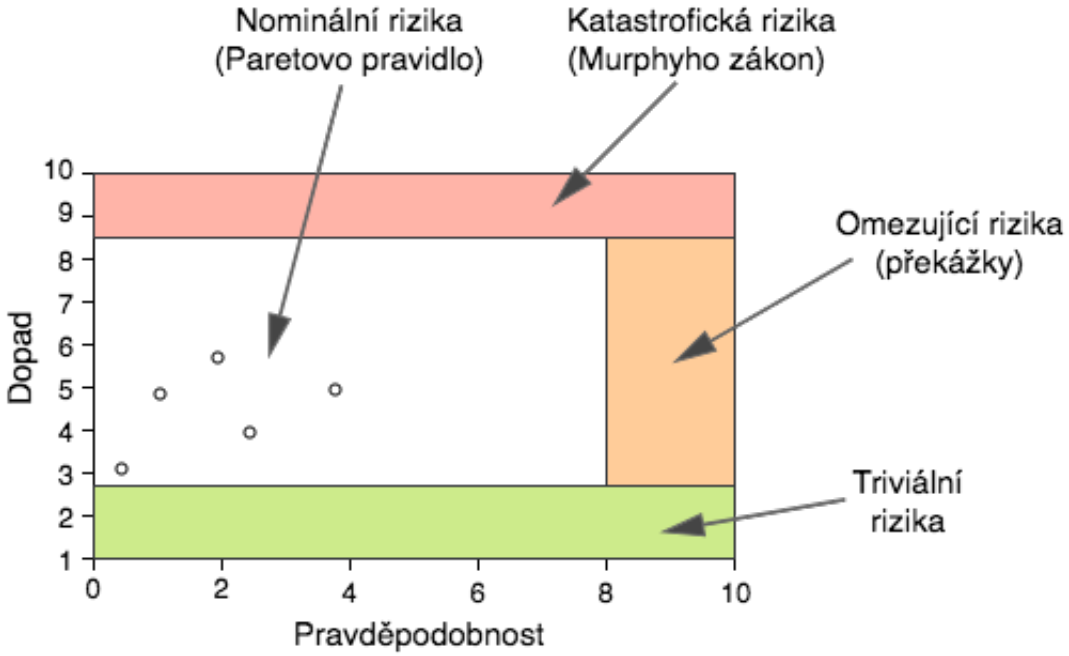
- míra ovlivnitelnosti rizika,
- míra akceptovatelnosti:
  - nezbytná,
  - únosná,
  - neúsnosná.
- vztah k organizaci:
  - externí,
  - interní.
- zasažená úroveň:
  - projekt,
  - program/produkt,
  - SBU (strategická obchodní jednotka),
  - společnost.
- Postižené oblasti – například která oddělení byla zasažena.
- Kategorie specifické k danému projektu, například míra ovlivnění jednotlivých cílů.
- Doba projevení dopadu, případně rychlost šíření.
- ...

Není třeba využívat všechny tyto kategorie, naopak je vhodné nalézt optimální množství a složení atributů specificky pro každý projekt.

2.3.5 Analýza rizik

Zásadní pro řízení rizik je jejich analýza. Pomocí analýzy zjišťujeme míru nebezpečí, zranitelnost (jak vysoká je pravděpodobnost, že hrozba nastane), jaký dopad může riziko mít. V tomto i v následujících krocích je důležité k rizikům přistupovat dle jejich určené priority. Priorita není pouze na základě pravděpodobnosti a dopadu rizika, ale může ohrožovat konkrétní cíle projektu, nebo část systému, která je pro výsledný projekt zásadní. Manažer by tedy měl vždy pečlivě zvážit priority jednotlivých rizik.

Analýzu rizik rozlišujeme na kvalitativní a kvantitativní. Během kvalitativní analýzy jsou rozebrána hrozící rizika, která jsou následně seřazena dle jejich závažnosti, na základě jejich pravděpodobnosti a dopadu případného vzniku. Následuje analýza kvantitativní, během které se provádí číselné odhady dopadů rizik na cíle projektu. Výstupem obou fází je aktualizovaný registr rizik.



Obrázek 2.6: Mapa povahy rizik [Inspirováno v [15]]

Analýza First-Order

Jedná se o metodu, která jednoduše zmapuje rizika a pomůže nám rychle identifikovat nejkritičtější rizika. Pro nejvíce kritická rizika se využívá FTA a ETA (fault/event tree analysis), kdy se snažíme dostat až k příčinám daného rizika. V případě těchto rizik aplikujeme Murphyho zákon, tedy že pokud se něco může pokazit, tak se to pokazí.

Jak již bylo nastíněno na obrázku 2.6, tak další kategorií jsou překážky. O ty je postaráno separátně, protože máme jistotu, že nastanou a musíme je tedy vyřešit. Další a největší kategorií jsou nominální rizika. Zde se naopak využívá Paretovo pravidlo, kdy se 20% rizik podílí na 80% vzniklých problémů [15]. Pro lepší rozdělení této kategorie se využívá vizualizace pomocí matice pravděpodobností a dopadů 2.3.5.

Triviální rizika jsou většinou pouze dále monitorována, aby nedošlo k jejich gradaci. V rámci této metody se rovněž využívá výše zmíněné metody Top 10 rizik 2.3.3.

Distribuční analýza

Pro lepší znázornění lze rizika vizualizovat do grafů rozdělením za pomoci jejich klasifikace. Takto lze rizika rozdělit dle vztahu k organizaci, úrovně jakou riziko zasahuje, doby projevení apod. Například u doby projevení je důležité, aby kromě důvodů vzniku byl analyzován i spouštěč daného rizika, podle kterého poznáme, že nastalo.

Analýza stromu událostí a poruch

Analýza stromu události, neboli ETA, a analýza stromu poruch, neboli FTA, jsou kauzální analytické techniky pro vyhodnocení průběhu procesu a jeho událostí vedoucích k možnému problému [14]. Analýza rozebírá sekvence činností a událostí v procesu a zobrazuje je pomocí

grafického modelu. Výsledkem jsou různé scénáře analyzovaného rizika. Zatímco ETA je preventivní metoda, FTA lze použít i při řešení existujícího problému.

Metoda Delphi

Delphi se využívá pro předpovídání budoucího vývoje pomocí skupiny anonymních expertů. Jedná se vlastně o druh brainstormingu, ovšem s jasně danými pravidly. Každá odpověď experta by měla být zdůvodněná a celý postup by měl mít několik kol. Díky zpětné vazbě se odhad zpřesňuje.

What-if analýza

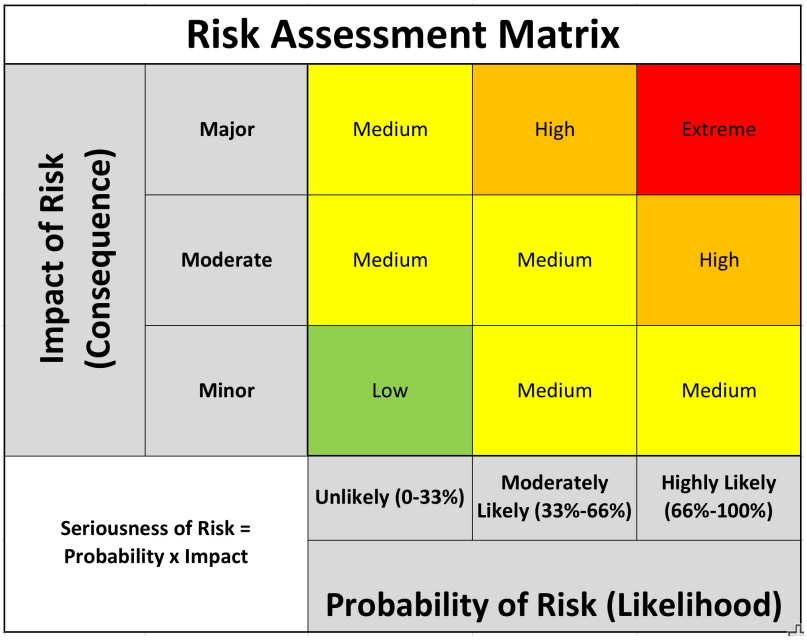
Jak z názvu vyplývá, tak princip metody What-if spočívá v hledání možných dopadů rizik. Využívá principy brainstormingu. Generují se scénáře, které se liší různými podmínkami.

Matice pravděpodobností a dopadů

Matice pravděpodobností a dopadů (Risk assessment matrix) umožňuje kategorizaci rizik podle dvou pamametrů:

- Pravděpodobnost vzniku rizika
- Dopady rizika pro projekt

Jak je znázorněno na obrázku 2.7 na jedné straně matice uvádí relativní pravděpodobnost vzniku rizika a na druhé relativní dopad rizika. Každé riziko je dle těchto atributů zařazeno na vypočtenou pozici v matici a díky tomu lze určit odpovídající priorita.



Obrázek 2.7: Příklad matice pravděpodobností a dopadů[7]

[7]Převzato z http://www.theprojectmanagementblueprint.com/wp-content/uploads/2015/09/Risk-Matrix-3x31.jpg

Pro výpočet pozice v matici je možné využít P-I skóre, které umožní přehledněji znázornit prioritu identifikovaných rizik dle závažnosti. Každému stupni pravděpodobnosti a důsledku vzniku přiřadíme skóre, kde nejvyšší číslo náleží nejpravděpodobnějším či nejhorším rizikům. Následně vypočteme závažnost rizika.

$$S = P + I \tag{2.1}$$

S závažnost rizika

P pravděpodobnost vzniku rizika

I dopad vzniku rizika

V případě, že riziko má několik různých typů dopadu, jako dopad na kvalitu, zpoždění, cenu, nebo reputaci, pak lze vypočítat jednotného skóre závažnosti následovně [19].

$$S = \log 10 \left(\sum_{i=1}^{k} 10^{P_i + l_i} \right) \tag{2.2}$$

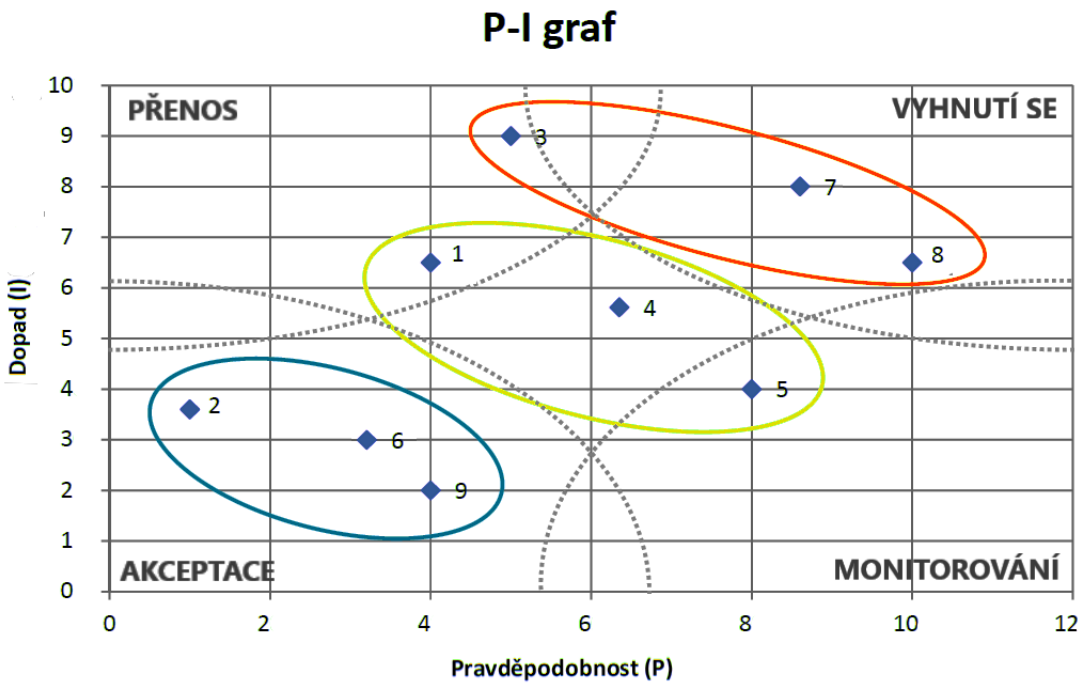
S závažnost rizika

P pravděpodobnost vzniku rizika

I dopad vzniku rizika

l typ dopadu vzniku rizika

Mimo matici lze rizika a jejich P-I skóre vizualizovat v grafu. Příklad takového grafu lze vidět na obrázku 2.8.



Obrázek 2.8: Příklad P-I grafu [Inspirováno v [19]]

2.3.6 Reakce na rizika

Poté, co úspěšně identifikujeme a analyzujeme rizika, je třeba na ně patřičně zareagovat. Prvním krokem je uvědomění všech zainteresovaných stran, poté nalezneme řešení a následně toto řešení implementujeme. Pomocí analýzy, především pak stromové analýzy, byly identifikovány příčiny rizika. Prvním krokem je tedy odstranění příčiny, druhou je posilnění a vylepšení procesu, ve kterém riziko vzniklo. Řešení je implementováno jako akční plán, který obsahuje datum začátku a konce řešení a seznam úkolů k jeho vyřešení. Cílem této fáze by mělo být provedení takových kroků, které vylepší příležitosti a sníží možné hrozby.

Vyhnutí se riziku

Zastavíme související úkoly a zrušíme plány, tím se vyhneme riziku. Tato metoda lze aplikovat pouze pro úkoly s nízkým ziskem či hodnotou.

Přenos rizika

Pomocí změn v týmu či projektu se přesune riziko pod osoby, které jsou kompetentnější v jeho řešení. Často se jedná o třetí stranu, tedy úplně cizí subjekt. Tímto přesunem se mohou výrazně změnit atributy rizika.

Akceptace rizika

V případě akceptace rizika jsou analyzovány důsledky a je nutno počítat s reakcí na vzniklé riziko. Lze vytvořit pohotovostní plán s případným řešením vzniklé situace.

Monitorování rizika

Pokud má riziko nízkou prioritu, tak je vhodným řešením jeho monitorování. Je nutno identifikovat spouštěče rizika a reakce na případný vznik.

Vyřešení rizika

V případě vyřešení rizika je třeba se soustředit jak na snížení pravděpodobnosti uskutečnění rizika, tak na snížení možného dopadu. Je tedy třeba vyřešit příčiny rizika, ale i posílit infikovaný proces, produkt či část projektu.

2.3.7 Kontrola rizik

Rizika je třeba sledovat, k tomuto účelu máme rizika přehledně zanesená do systému pro řízení rizik. Uchováváme informace o jejich atributech, kde především atributy jako pravděpodobnost a dopad sledujeme v čase, protože tyto hodnoty se mohou měnit. Dále uchováváme informace o možných spouštěčích rizik a pohotovostních akčních plánech. Je třeba rizika monitorovat, co nejdříve identifikovat nová rizika a reagovat na vzniklé problémy za pomocí vyhotovených akčních plánů. Obzvlášť důležitým aspektem, hlavně pro softwarové projekty, je vyvození důsledků a využití nově získaných zkušeností pro vylepšení. Je třeba posoudit celý proces, poučit se z chyb a pomocí zpětné vazby navrhnout a implementovat možné změny a vylepšení. Výstupem této fáze jsou mimo jiné nápravná a preventivní opatření.

Kapitola 3

Rozhodovací analýza

Analýza rizik je součástí rozhodovací analýzy. Rozhodovací analýza je disciplína, která pomáhá se chytře rozhodnout v nejistých situacích [16]. Je to framework metod a nástrojů, které podporují kreativitu a pomáhají lidem se lépe rozhodovat [18]. A ačkoliv se problémy, které řeší, liší a každý je unikátní, tak proces, kterým je rozhodujeme, může být stejný. Rozhodovací analýza je aplikovatelná a přizpůsobitelná jakémukoliv typu rozhodování v rámci projektového řízení.

Obecně jde k rozhodování přistupovat třemi odlišnými přístupy.

- Intuitivní přístup na základě pocitů manažera.
- Týmový přístup, kdy si manažer nechá vypracovat hodnocení od týmu. Tato metoda ale opět záleží na intuici, protože manažer může hodnocení odmítnout a nechat ho přepracovat čistě na základě svých pocitů.
- Při rozhodovací analýze je výběr možností více na základě hodnocení výsledků analýz, než na intuici.

Je dokázáno, že intuice není spolehlivá metoda, protože z psychologického hlediska má člověk sklony k určitým předsudkům a sklonům (tzv. biases), je to rozdíl mezi něčím úsudkem a realitou. Tomuto tématu se rovněž věnují Lev Virine a Michael Trumper v knize Project decisions: the art and science [18].

Rozhodování by mělo být transparentní, konzistentní, komplexní a kontinuální. Podobné problémy by se měly řešit standardizovaně, měli bychom vždy zajistit dostatek informací a řádně analyzovat danou situaci a rozhodnutí je třeba sledovat a upravovat dle potřeby.

Rozhodovací analýza se dělí na čtyři fáze. První fáze je formulování rozhodnutí nebo strukturování problému. Během této fáze bychom měli rovněž identifikovat příležitosti. Druhou fází je modelování alternativ. Třetí fáze se poté zabývá analýzou, během které je třeba vytvořit rozhodovací model, kvantifikovat odhady pravděpodobnosti, vytvořit hodnotící model a vypočítat očekávanou hodnotu pro každou z alternativ. V poslední fázi se rozhodnutí implementují, monitorují a přezkoumávají. Protože je tato práce zaměřena na tematiku rizikového řízení, tak lze definovat, že první dvě fáze odpovídají fázím plánování a identifikace rizik a poslední fáze odpovídá kontrole rizik. Z tohoto důvodu se nyní zaměříme na analýzu rozhodnutí a tvorbu modelů, tedy na vizualizaci rozhodnutí a rizik.

3.0.1 Tabulkové modely

Do této kategorie spadá matice pravděpodobností a dopadů, která již byla rozebrána v sekci 2.3.5. Další vhodná porovnání mohou být požadavky vůči rizikům, cíle vůči rizikům a nebo příčiny vůči rizikům.

3.0.2 Stromové modely

Mezi stromové modely patří již zmíněné ETA a FTA stromy. Zvláštním případem stromového modelu, kterému se dále budeme věnovat, je rozhodovací strom.

Rozhodovací strom je graf, který reprezentuje problém, o kterém rozhodujeme. Jedná se o techniku grafové analýzy, která napomáhá při výběru nejlepšího postupu v situacích s nejistými výsledky [17]. Rozhodovací stromy jsou nejčastěji používány pro výpočet očekávané hodnoty nebo očekávané užitečnosti, podle jejíž hodnot je následně uskutečněno rozhodnutí.

Očekávaná hodnota

V případě, že je rozhodování založeno na výběru z několika alternativ, tak se využívá výpočtu očekávané hodnoty (expected value). Nejčastěji hovoříme o očekávané peněžní hodnotě. Jedná se o pravděpodobnostně vážený průměr všech výstupů. Výpočet očekávané hodnoty je následující:

$$EV = \sum (p_n * v_n) \tag{3.1}$$

EV očekávaná hodnota

p_n pravděpodobnost alternativy

v_n hodnota rizikové události

n počet alternativ daného řešení

Příklad výpočtů pro názornost je zobrazen na obrázku 3.1. Čtverec značí rozhodnutí (decision node), kolečko značí nejistotu neboli možné riziko (chance node), v této větvi se pak uvádí pravděpodobnost tohoto jevu. Trojúhelník značí konec větve (end node), každý konec větve obsahuje výpočet očekávané hodnoty. Od koncových uzlů k počátečnímu se poté počítá celková očekávaná hodnota pro každý uzel nejistoty.

Očekávaná doba

Pro časové odhady se hodí znát techniku PERT (Program Evaluation and Review Technique) [18], pomocí které lze spočítat očekávanou dobu.

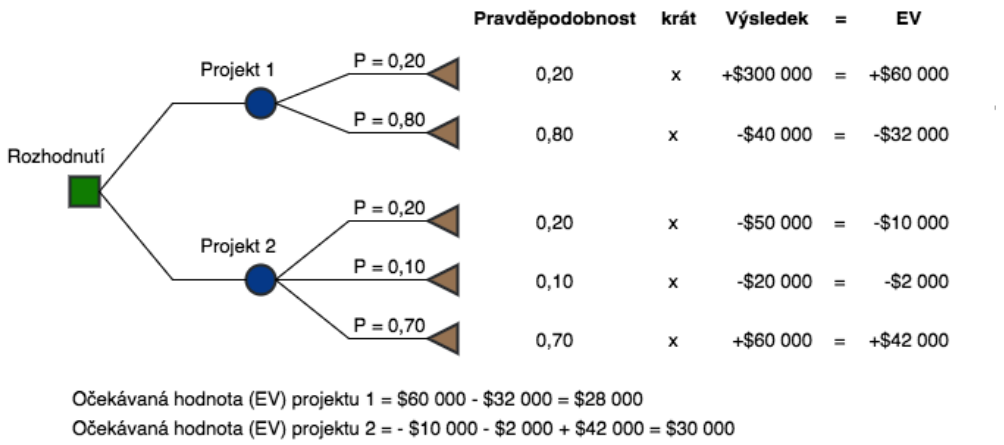
$$t = \frac{(t_{opt} + t_{real} * 4 + t_{pes})}{6} \tag{3.2}$$

t očekávaná doba

t_{opt} optimistický odhad času

t_{real} nejvíce pravděpodobný odhad času

t_{pes} pesimistický odhad času



Obrázek 3.1: Příklad rozhodovacího stromu s výpočty [Inspirováno v [17]]

3.0.3 Statistické metody

Místo modelování se lze spolehnout na matematické výpočty. V této části jsou shrnuty nejdůležitější informace, které se využívají při statistických výpočtech. Nejprve je třeba zadefinovat pojem náhodné veličiny.

$(\Omega, \mathcal{F}, P)$ je pravděpodobnostní prostor; to znamená, že

- Ω je libovolná neprázdná množina (množina elementárních jevů),
- $\mathcal{F}$ je libovolný systém podmnožin Ω , který tvoří σ -algebru, a
- P je pravděpodobnost, čili míra na $(\Omega, \mathcal{F})$, která je normovaná tak, že $P(\Omega) = 1$

$(R, \mathcal{B})$ je množina všech reálných čísel s borelovskou σ -algebrou podmnožin $\mathcal{B}$;
Náhodnou veličinou pak nazýváme každé zobrazení přiřazující elementárnímu jevu reálné číslo, tj. $X : \Omega \to R$, pokud je měřitelné, t.j. pokud pro každou množinu $B \in \mathcal{B}$ platí, že $\{\omega; \omega \in \Omega, X(\omega) \in B\} \in \mathcal{F}$[1].

Histogram frekvence znázorňuje předem zvolené intervaly určité náhodné veličiny (například odhad ceny projektu) a četnost odhadů spadajících do každého z intervalů. Při dostatečném množství intervalů a dat, pak nejvyšší body histogramu frekvence aproximují funkci hustoty rozdělení pravděpodobnosti.

Je-li $\rho(x)$ hustota pravděpodobnosti spojité náhodné veličiny X, pak platí $\rho(x)\mathrm{d}x = 1$, kde Ω je definiční obor veličiny X. Pro hodnoty x mimo definiční obor Ω je hustota pravděpodobnosti nulová, tzn. $\rho(x) = 0$ pro $x \notin \Omega$.

Z grafu této funkce lze vyčíst modus, neboli nejoptimističtější hodnotu, a střední hodnotu.

Modus spojité náhodné veličiny je taková hodnota $\hat{x}$, která pro všechny hodnoty x_i náhodné veličiny X splňuje podmínku $P[X = \hat{x}] \geq P[X = x_i]$ s podmínkou $f(\hat{x}) \geq f(x)$, kde f je hustota pravděpodobnosti náhodné veličiny X.

Výpočet střední hodnoty je v rovnici 3.1. Pojmy očekávaná hodnota, střední hodnota a vážený průměr daného rozdělení jsou synonyma. Značí se rovněž řeckým písmenem μ.

[1] https://cs.wikipedia.org/wiki/Náhodná_veličina

Pro spojitou náhodnou veličinu s hustotou pravděpodobnosti $\rho(x)$ lze distribuční funkci spočítat podle vztahu:

$$F(x) = \int_{-\infty}^{x} \rho(t)\mathrm{d}t \tag{3.3}$$

Druhou nejčastěji používanou statistickou funkcí po střední hodnotě je směrodatná odchylka [16]. Směrodatná odchylka je definována jako odmocnina z rozptylu náhodné veličiny, tedy $\sigma = \sqrt{D(X)}$. Pro spojitou náhodnou veličinu definujeme rozptyl vztahem:

$$\sigma^2 = \int_{-\infty}^{\infty} x^2 f(x)\mathrm{d}x - [\mathrm{E}(X)]^2, \tag{3.4}$$

Obecně řečeno je střední odchylka parametr měřící, jak široko rozložené hodnoty v distribuci jsou.

Dále jsou uvedeny nejzákladnější pravděpodobnostní rozdělení. V projektovém managementu jsou jevy spojené s nejistotou nejčastěji spojité – čas, cena, kvalita. Z tohoto důvodu jsou blíže představeny spojitá rozdělení pravděpodobnosti. V jednodušších systémech, nebo pro speciální události, jako je počet výpadků nebo počet dostupných zaměstnanců, se využívají diskrétní rozdělení, nejčastěji rozdělení binomické a Poissonovo [8] [7].

Rovnoměrné rozložení

Rovnoměrné rozložení reprezentuje rovnocennou pravděpodobnost, že parametr bude z určitého rozsahu. Je pouze známo, že náhodná veličina leží mezi a a b a tedy všechny hodnoty mezi minimem a maximem mají stejnou pravděpodobnost.

Hustota pravděpodobnosti	$f(x) = \frac{1}{max-min}$
Distribuční funkce	$F(x) = \frac{x-min}{max-min}$
Omezení parametrů	$min \neq max$
Obor hodnot	$min < x < max$
Střední hodnota	$\frac{min+max}{2}$
Rozptyl	$\frac{(max-min)^2)}{12}$

Exponenciální rozdělení

Toto rozdělení se nejčastěji používá při měření času do prvního výskytu dané události.

Hustota pravděpodobnosti	$f(x) = \frac{\exp(-\frac{x}{\beta})}{\beta}$
Distribuční funkce	$F(x) = 1 - \exp(-\frac{x}{\beta})$
Omezení parametrů	$\beta > 0$
Obor hodnot	$x > 0$
Střední hodnota	β
Rozptyl	β^2

Trojúhelníkové

Tato distribuce využívá odhad parametru použitím jeho minimálního, maximálního a nejpravděpodobnějšího odhadu. Je často používaná pro její jednoduchost, ale pro reálný svět je příliš zjednodušená.

Hustota pravděpodobnosti	pro $min \leq x \leq modus$ platí $f(x) = \frac{2(x-min)}{(modus-min)(max-min)}$ pro $modus < x \leq max$ platí $f(x) = \frac{2(max-x)}{(max-min)(max-modus)}$
Distribuční funkce	pro $x < min$ platí $F(x) = 0$ pro $min \leq x \leq modus$ platí $F(x) = \frac{(x-min)^2}{(modus-min)*(max-min))}$ pro $modus < x \leq max$ platí $F(x) = 1 - \frac{(max-x)^2}{(max-min)*(max-modus)}$ pro $max < x$ platí $F(x) = 1$
Omezení parametrů	$min \leq modus \leq max$, $min < max$
Obor hodnot	$min < x < max$
Střední hodnota	$\frac{min+modus+max}{3}$
Rozptyl	$\frac{min^2+modus^2+max^2-min*modus-min*max-modus*max}{18}$

Normální rozdělení

V normálním (též Gaussově) rozdělení jsou hodnoty distribuovány symetricky okolo střední hodnoty. Toto rozložení nastává velmi často jak v přírodě, tak ve světě obchodu. Toto rozložení je často používané například při znázornění distribuce chyb. Často se využívá symetričnosti a faktu, že toto rozdělení za určitých podmínek dobře aproximuje řadu jiných pravděpodobnostních rozdělení.

Hustota pravděpodobnosti	$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(x-\mu)^2)}{2\sigma^2})$
Distribuční funkce	$--$
Omezení parametrů	$\sigma > 0$
Obor hodnot	$-\infty < x < +\infty$
Střední hodnota	μ
Rozptyl	σ^2

Další často používané rozdělení jsou například Beta, Gamma, χ^2, Weibull, Lognormální (používané například pro fyzický objem produktu) a spousta dalších.

3.0.4 Simulace Monte Carlo

Analýza Monte Carlo je technika pro kvantifikaci rizik, která opakovaně simuluje výsledky modelu a zjišťuje tak statistické rozdělení vypočtených výsledků. Jedná se o statistické vzorkování. Matematická metoda Monte Carlo se využívá k aproximaci distribuce potenciálních

výsledků na základě pravděpodobnosti. Pro každý pokus je vygenerován náhodný vzorek hodnot pro každou proměnnou z její definované distribuční funkce. Tyto vstupní hodnoty jsou poté využity pro výpočet výsledků. Postup se poté opakuje dokud výsledek nedosáhne požadované přesnosti. Přesnost se mění na základě relativního rozdílu mezi směrodatnou odchylkou nebo očekávanou hodnotou pro dva po sobě jdoucí pokusy.

Základní kroky simulace Monte Carlo [17]:

1. Shromáždíme data o nejpravděpodobnějším, optimistickém a pesimistickém odhadu hodnot veličin ve zvoleném modelu.

2. Stanovíme distribuci neboli rozdělení pravděpodobnosti jednotlivých veličin.

3. Provedeme vzorkování, tedy pro každou z veličin vybereme náhodnou hodnotu ze stanoveného rozdělení pravděpodobnosti.

4. Nad kombinací hodnot spustíme deterministickou analýzu, tedy jeden průchod modelem.

5. Kroky 3 a 4 mnohokrát zopakujeme, abychom docílili věrohodných výsledků simulace. Optimální počet iterací se pohybuje mezi 100 a 1000.

Při generování náhodných vzorků se využívá inverzní funkce $G(F(x))$ k distribuční funkci. Vygeneruje se náhodné číslo r v intervalu $< 0,1 >$ a poté se dosadí do rovnice inverzní funkce $G(r) = x$. Tím získáme vygenerovanou hodnotu pro zvolenou distribuční funkci.

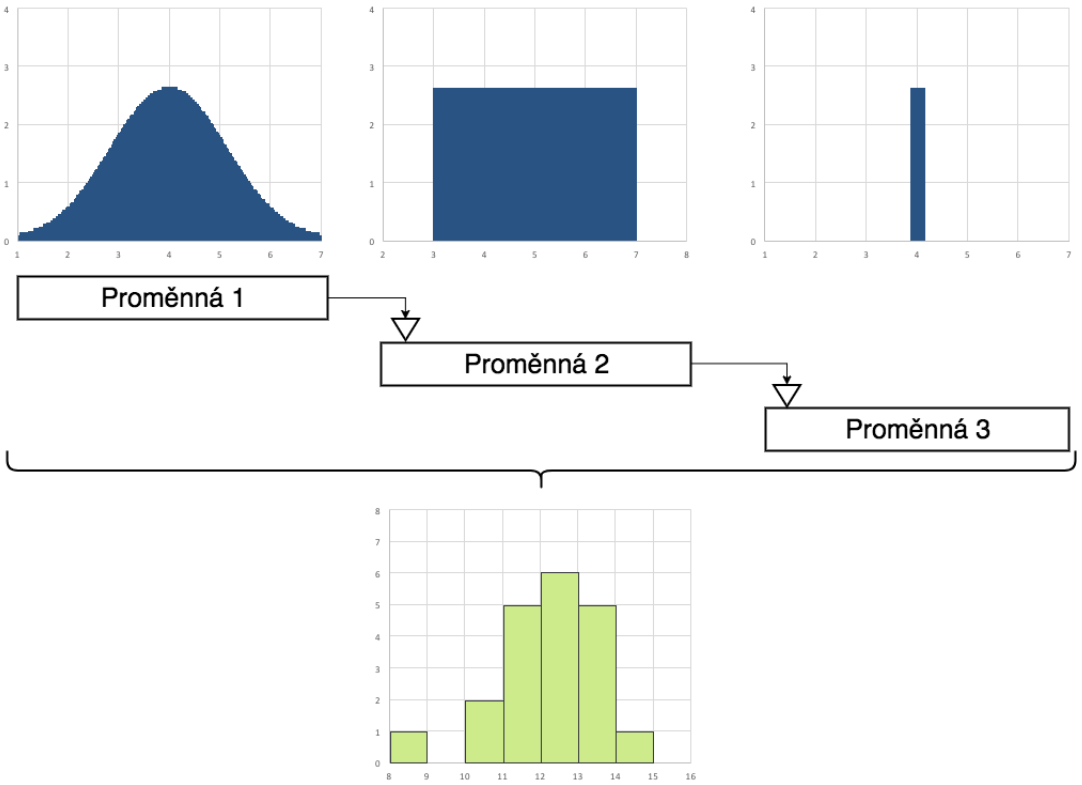
Výsledkem simulace je distribuce pro zvolené parametry, která vznikla z výsledných hodnot každého průchodu. Z tohoto modelu lze poté dopočítat všechny pravděpodobnostní parametry, včetně střední hodnoty, nebo směrodatné odchylky. Princip simulace je znázorněn na obrázku 3.2.

Poslední otázkou zůstává, jak vybrat vhodnou distribuci pravděpodobnosti pro každý z parametrů. Jednou možností je odvodit distribuci z historických dat, ty ale bohužel nejsou vždy k dispozici. Pravděpodobnostní metoda pracuje s odhadem rozsahu hodnot, který je rozdělen do intervalu a za pomoci otázek, jaká je pravděpodobnost, že bude parametr o jednu, dvě či více jednotek, mimo interval. Vzniklými odhady lze poté proložit křivka, která znázorňuje hledané rozdělení pravděpodobnosti.

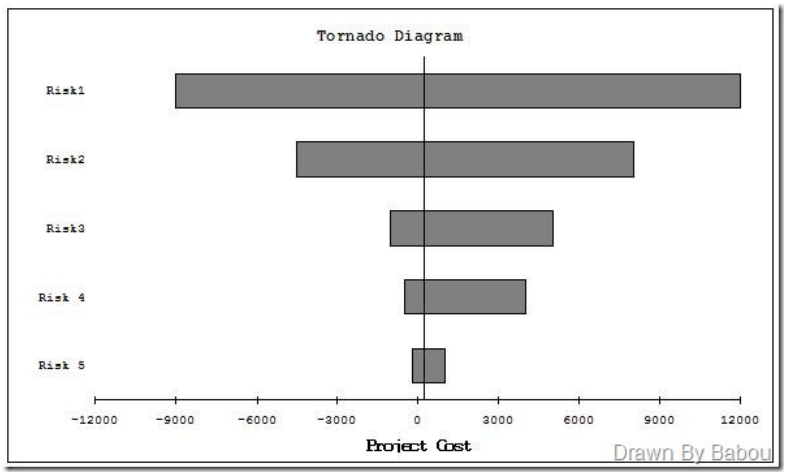
V porovnání s metodou rozhodovacích stromů je simulace vhodnější v případech, kdy existuje velké množství signifikantních nejistot, kdy jsou vyžadovány komplexnější výsledky a nebo kdy je potřeba distribuční funkce pro další studium rizik a možných výsledků. Naopak rozhodovací stromy jsou používány v případech, kdy se řeší sekvence rozhodnutí, nebo u méně prioritních rizik, či tam, kde tento jednodušší model bude jednoduše dostatečný.

3.0.5 Analýza citlivosti

Tato technika zjišťuje do jaké míry se změny jedné nebo více veličin promítají do konečných výsledků, respektive jak jsou výsledky citlivé na změnu vstupních hodnot. Jednou z používaných analýz je například analýza bodu zvratu. Cílem citlivostní analýzy je identifikovat klíčové proměnné, které vyžadují zvýšenou pozornost. Výsledky této analýzy se zakreslují buď do tzv. pavoučích grafů, kde je zobrazena závislost odchylky od základního případu na výsledné hodnotě. Dále se využívají speciální citlivostní schémata, případně tornádové (trychtýřové) diagramy. Příklad tornádového diagramu je znázorněn na obrázku



Obrázek 3.2: Princip simulace Monte Carlo [Inspirováno v [18]]



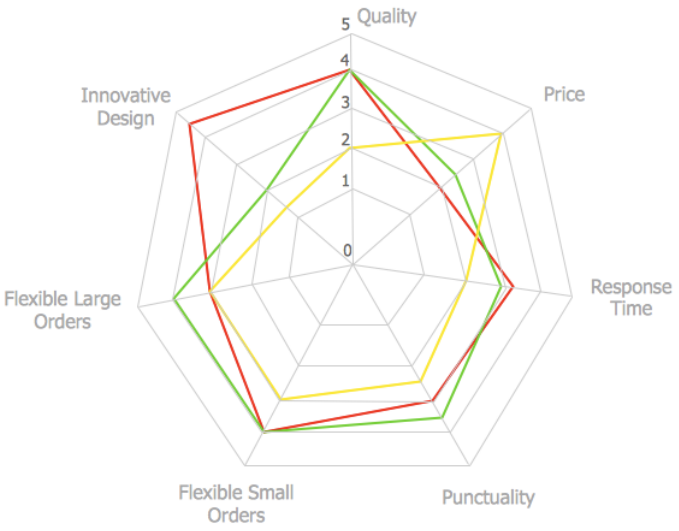
Obrázek 3.3: Příklad tornádového diagramu[2]

3.3, čím delší je sloupec, tím více citlivá tato veličina je. Faktory jsou seřazeny sestupně, tedy nejcitlivější faktor je jako první.

[2]Převzato z https://leadershipchamps.wordpress.com/2009/06/14/find-how-sensitive-is-your-project-against-variables-tornado-diagram/

3.0.6 Bodovací model

Pro rozhodování, ve kterých je třeba zvážit více kritérií je vhodné využít například Bodovací model. Po identifikaci jednotlivých rozhodovacích kritérií je třeba jim přidělit váhu na základě jejich důležitosti, tu můžeme získat například pomocí výše zmíněné analýzy citlivosti 3.0.5. Poté je třeba pro každé alternativní řešení ohodnotit jak splňuje jednotlivá kritéria a vypočítat skóre pro každé kritérium vynásobením váhy a uděleného hodnocení, toto hodnocení se udává v intervalu $< 0, 1 >$. Výsledky se poté dají vizualizovat za pomocí paprskového grafu viz obrázek 3.4, kde je každá alternativa barevně vyznačena.



Obrázek 3.4: Příklad bodovacího modelu[3]

[3]Převzato z http://www.conceptdraw.com/examples/percentage

Kapitola 4

Návrh systému

Tato kapitola se věnuje návrhu systému pro správu a vizualizaci rizik. Nejprve jsou analyzována již existující řešení na trhu. O jejich výsledky se poté opírá specifikace požadavků doplněná o diagram případu použití. Aplikaci by primárně měl používat projektový manažer, který má daný projekt na starosti. Dále může udělit přístup do projektu vybraným uživatelům, kteří budou mít omezené pravomoce.

4.1 Existující řešení

Pro analýzu byly použity nejlépe hodnocené nástroje z webu Capterra[1]. Pokud bylo možné, byla vyzkoušena demo verze daného produktu, jinak byla provedena analýza z dostupných videí či snímků obrazovky z dané aplikace.

4.1.1 ProcessGene GRC Software Suite

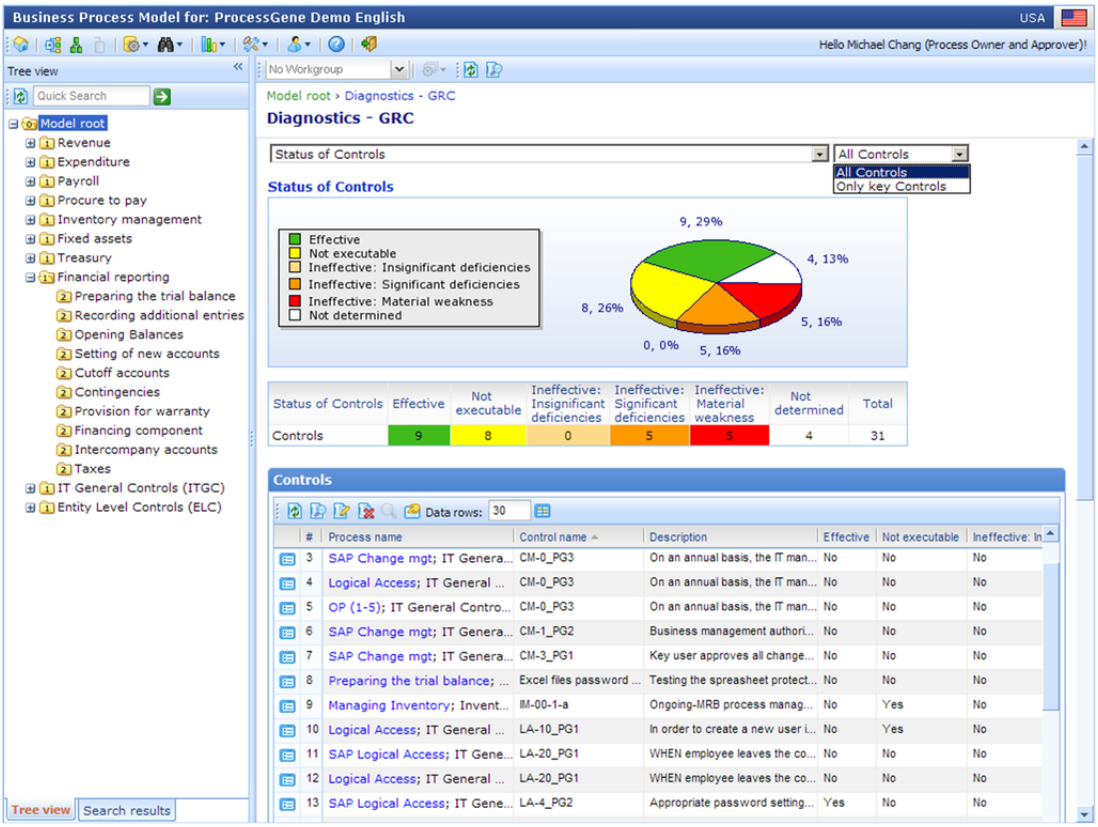
Výrobcem je izraelská firma, založená v roce 2014. Kromě softwaru pro řízení rizik mají mnoho dalších kompatibilních nástrojů pro celkový management projektů, ale hlavně pro různé audity, regulace a politiky. ProcessGene^{TM} Risk Management software[2] je složitější nástroj, který umožňuje firmám identifikovat, analyzovat, řešit a monitorovat rizika. Tento nástroj pokrývá rozsáhlou nabídku rizik, mezi nimi například finanční, operativní, IT rizika nebo rizika týkající se značky či reputace firmy.

Jedná se o desktopovou aplikaci, která je více zaměřena na velké firmy a přináší velmi obsáhlé množství možností. Bohužel její rozsáhlost je spíše na škodu a je značně uživatelsky nepřívětivá a dle recenzí mají uživatelé i špatné zkušenosti s následnou uživatelskou podporou.

4.1.2 MasterControl Risk Analysis

MasterControl[4] je firma založená v roce 1993 v USA. Aplikace MasterControl's Risk Management Software je rovněž součástí rozsáhlého balíku aplikací, které zastřešují systém pro audity, systém pro zákaznickou podporu, řízení kvality, zásob, školící systém a spoustu dalších. Zaměřuje se na dlouhodobé řízení rizik a pokrývá celý cyklus řízení.

[1] https://www.capterra.com/risk-management-software/
[2] http://processgene.com/solutions/grc-software/risk-management/
[3] Převzato z https://assets.getapp.com/
[4] https://www.mastercontrol.com/risk-software/risk_analysis.html



Obrázek 4.1: Snímek obrazovky z aplikace ProcessGene GRC Software Suite[3]

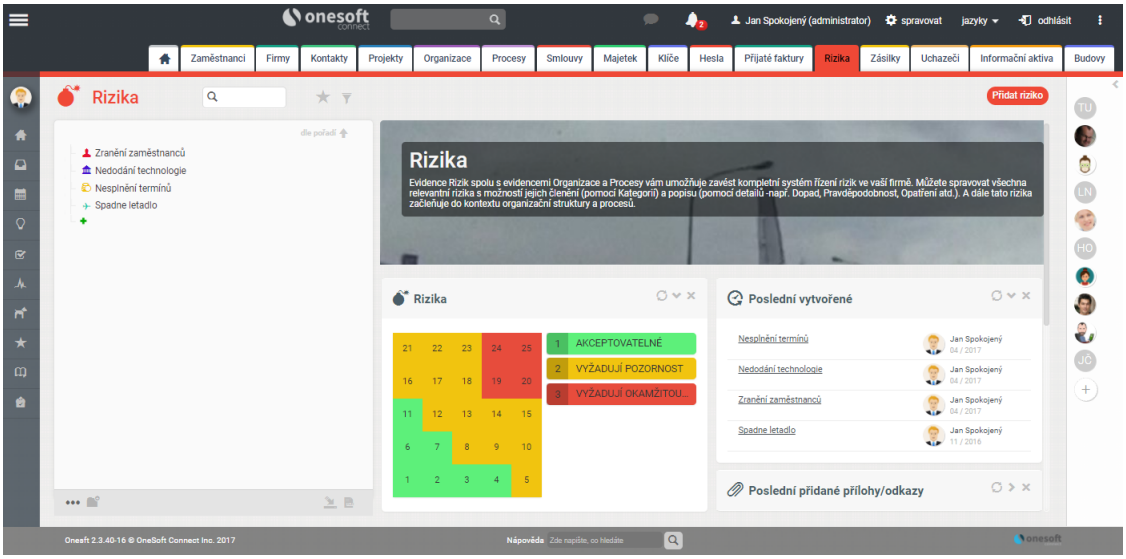
Aplikace má dle recenzí spoustu chyb, které bohužel nejsou opravovány. Jedná se rovněž o značně komplikovaný systém, kde úprava jednoho údaje ovlivní řadu dalších věcí, o kterých nemusel být uživatel informován.

4.1.3 OneSoft Connect

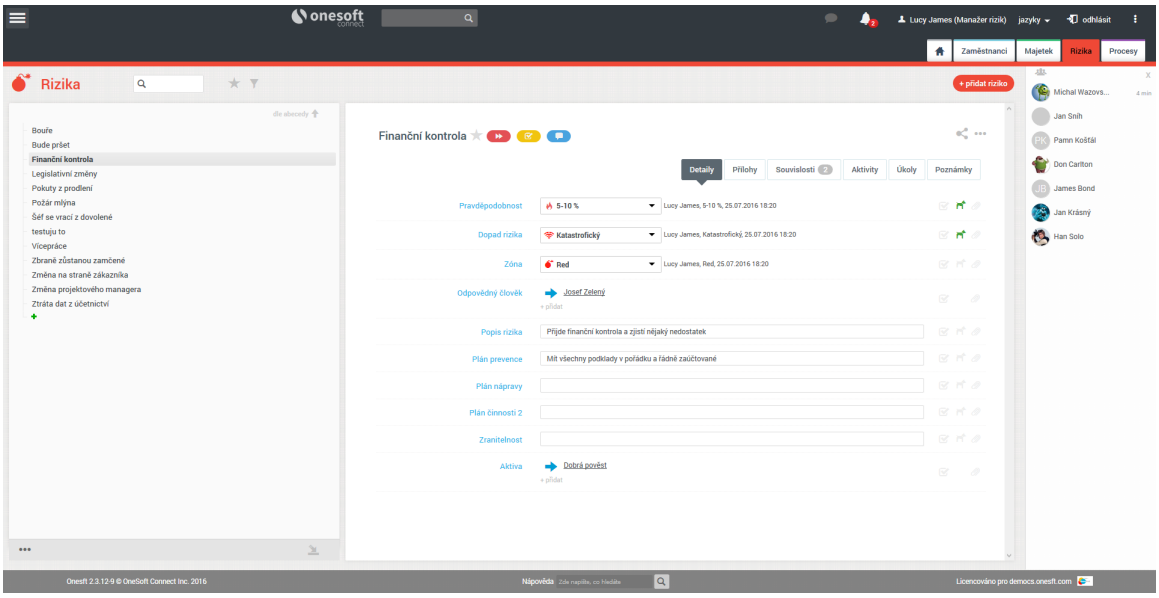
Firma OneSoft Connect[5] byla rovněž založena v USA v roce 2014. Její systém je z pohledu uživatelské použitelnosti a přehlednosti rozhodně nejlepším z výše jmenovaných konkurentů. Mimo hotová řešení poskytuje firma zároveň i možnost přípravy softwaru na míru. Nabízená hotová řešení se mimo řízení rizik zabývají nejenom řízením projektů, ale i správou majetku, lidských zdrojů, obchodu a vztahů se zákazníky s spoustou dalších.

Aplikace je řešena za pomoci webového rozhraní a tedy je kompatibilní napříč různými zařízeními. Je přehledná, upravitelná a uživatelsky přívětivá. Nevýhodou je, že z oblasti vizualizace rizik obsahuje pouze matici pravděpodobností a dopadů. Jako pozitivum a inspirativní nápad hodnotím možnost exportu dat do formátu xls.

[5] http://processgene.com/solutions/grc-software/risk-management/
[6] Převzato z https://www.onesft.com/cs/system-rizeni-rizik
[7] Převzato z https://www.onesft.com/cs/system-rizeni-rizik



Obrázek 4.2: Snímek obrazovky z aplikace OneSoft Connect[6]



Obrázek 4.3: Snímek obrazovky z aplikace OneSoft Connect[7]

4.2 Specifikace požadavků

Výsledná aplikace by měla umožňovat nejenom analýzu rizik, ale měla by pokrývat celý cyklus řízení rizik. Zároveň je mým cílem, aby byla aplikace použitelná a nápomocná pro moderní agilní vývoj, a to nejlépe pro prostředí menších firem a start-upů.

Pro identifikaci rizik budou k dispozici kontrolní seznamy a dotazníky 2.3.3. Klasifikace rizik 2.3.4 se může lišit projekt od projektu. Výběr jednotlivých atributů rizik bude tedy starostí manažera, který je zvolí pro každý ze svých projektů zvlášť. Registr rizik projektu bude vizualizován jako tabulka, ve které bude umožněno filtrovat a vyhledávat. V rámci analýzy rizik bude možné nad vybranými daty zobrazit distribuční analýzu 2.3.5, matici

pravděpodobností a dopadů 2.3.5 a případně další vizualizace. Reakce na rizika budou řešeny atributy stav rizika a plán pro vyřešení rizika.

O rizicích se budou uchovávat následující údaje:

- ID,
- název,
- popis,
- kategorie,
- stav,
- popis možných následků,
- plán pro vyřešení rizika,
- pravděpodobnost,
- dopad,
- závažnost,
- příčina,
- spoušteč.

Dále bude možné vytvořit unikátní sadu atributů rizik pro každý projekt. Na výběr bude z těchto atributů:

- Původ, či zdroj rizika.
- Povaha rizika.
- Dotčené klíčové oblasti, tedy jak ovlivňuje výkon projektu.
- Rychlost následků.
- Čas útoku, tedy jak rychle pocítí firma následky.
- Obchodní úroveň rizika, zda například ovlivňuje pouze projekt, SBU či celou firmu.
- Viditelnost, jak moc viditelné budou následky rizika.
- Dotčené procesní oblasti, umožňuje určit oblasti pro korektivní opatření.

Navrhovaný systém, by měl být kompatibilní v rámci zařízení i v rámci různých operačních systémů. Nejvhodnějším řešením je tedy zvolení webové aplikace, která bude mít responzivní design, a tedy bude uživatelsky přívětivá nejenom v desktopové verzi, ale i na mobilních zařízeních, nehledě na operační systém daného zařízení. Manažerům to umožní jednoduchý a neustálý přístup k datům. Navíc se urychlí a zpřehlední komunikace rizik, která je v rámci agilních týmů důležitá.

Jak je vidět na diagramu případu použití 4.4, tak systém bude rozlišovat tři systémové role. Nejdůležitější rolí v systému je manažer, který je vlastníkem projektu, má tedy pravomoc spravovat projekty a uživatele, kteří k daným projektům mají přístup. Uživatel,

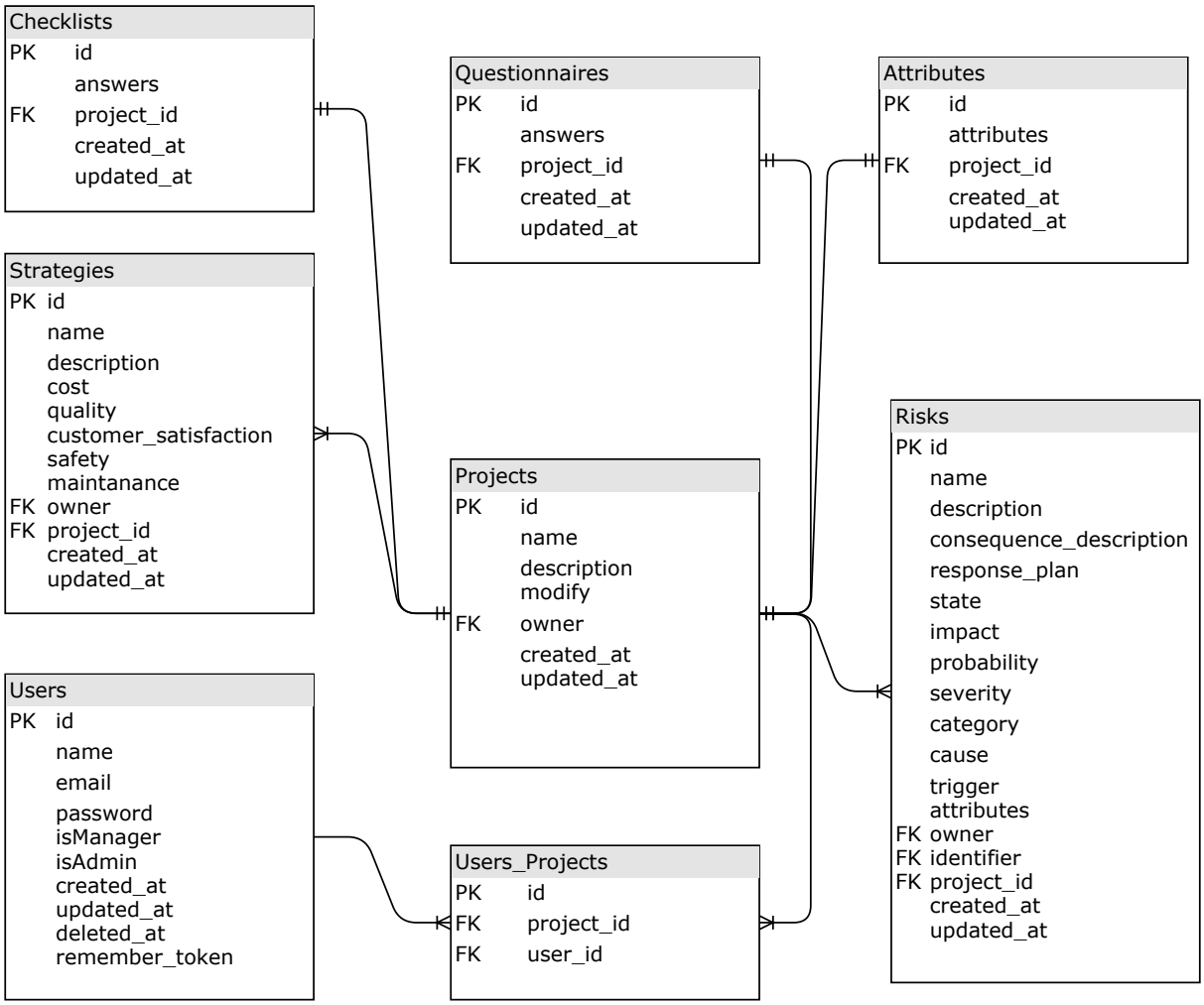
Obrázek 4.4: Diagram případů použití [zdroj vlastní]

který získá přístup k projektu, může přistupovat k registru rizik, kde zvolená rizika může spravovat. Může využívat předpřipravené kontrolní seznamy a dotazníky, které mu umožní snadnější identifikaci rizik. V registru rizik bude možnost si rizika vyfiltrovat dle zvolené klasifikace. Rizika bude rovněž možné vykreslit do vybraných vizualizací. Data bude možno exportovat do formátu pdf a xls. Třetí rolí je administrátor systému, který má na starosti správu všech uživatelů aplikace.

4.3 Návrh uživatelského rozhraní a datové struktury

Drátové modely (wireframes) pro grafický návrh aplikace byly vypracovány pouze na papír, proto nejsou k práci přiloženy. Návrh uživatelského rozhraní se soustředí hlavně na jednoduchost a přehlednost aplikace, bez zbytečných rozptýlení. Po přihlášení bude uživatel přesměrován na rozpis projektů, ke kterým má patřičná oprávnění, či kterých je vlastníkem. Bude zde možnost projekty spravovat a administrátor bude mít možnost přejít do správy uživatelů. Po otevření projektu bude zobrazena nejdůležitější část aplikace a to registr rizik pro daný projekt. Bude zde možné přepínat jednotlivé vizualizace, přejít na vyplnění dotazníků či kontrolních seznamů, spravovat projekt a samozřejmě spravovat samotná rizika.

Pro uložení dat bude použita MySQL databáze. Struktura databáze je vidět na diagramu 4.5.



Obrázek 4.5: Diagram struktury databáze [zdroj vlastní]

Kapitola 5

Realizace

Jak bylo již v návrhu specifikováno, výsledný prototyp aplikace by měl být responzivní a uživatelsky přívětivý. Bylo proto nutné nejprve vybrat vhodný framework, tedy aplikační rámec pro jazyk PHP, který bude nejlépe vyhovovat požadavkům navrženého systému. Nejprve se v této kapitole tedy zaměřím na porovnání nejpoužívanějších aplikačních rámců. Fungování a možnosti vybraného rámce, Laravelu, budou v následující části blíže vysvětleny. Poslední části této kapitoly se zabývá již přímo implementací výsledného prototypu. Zaměřuje se hlavně na nejzajímavější a nejobtížnější části aplikace z implementačního pohledu. Implementační část je rozdělena na celky, které odpovídají průchodu aplikace uživatelem.

5.1 Výběr aplikačního rámce

5.1.1 Proč používat framework?

Framework, neboli aplikační rámec, je ucelený soubor tematicky zaměřených knihoven a ovlivňuje globální chování aplikace, zatímco knihovna je spíše souborem funkcí a tříd, které poskytují okruh služeb, a aby byla využitelná s jakoukoliv aplikací a mohla spolupracovat s dalšími knihovnami, je potřeba, aby naopak nijak neovlivňovala globální stav programu.

Použití existujícího a plně otestovaného aplikačního rámce, tedy komplexního programového řešení, nám nejenom usnadní práci při programování, ale hlavně umožní předejít možným bezpečnostním či návrhovým chybám. Bez jeho použití se často objevují problémy z kategorie tzv. návrhových antivzorů – například špagetový kód, vynalézání kola, kopírování částí kódu apod. Používání aplikačních rámců zajistí čistší kód, snadnější znovupoužitelnost, otestované komponenty a obecně lepší organizaci zdrojových kódů. Většina z nich využívá architektury MVC, případně nadstavbu HMVC. Při výběru jsem vycházela rovněž z informací z absolvovaného kurzu MVC Frameworks for Building PHP Web Applications[1] na platformě LinkedIn Learning.

MVC

Model-view-controller je softwarový architektonický vzor, který rozděluje datový model aplikace, uživatelské rozhraní a řídicí logiku do tří nezávislých komponent tak, že modifikace některé z nich má jen minimální vliv na ostatní. Koncept MVC je nejpoužívanější právě

[1] https://www.linkedin.com/learning/mvc-frameworks-for-building-php-web-applications-2

pro tvorbu webových aplikací a zajišťuje tím flexibilitu a spolehlivost při častých změnách při vývoji. Více o MVC je v části 5.2.

- Model – datová vrstva.
- Controller – řídící vrstva.
- View – prezenční vrstva.

5.1.2 Porovnání aplikačních rámců

Pro porovnání jsem vybrala 8 aktuálně nejpoužívanějších aplikačních rámců [13] pro jazyk PHP. Rozhodujícími prvky pro mě bylo mimo jiné zpracování dat, existující komponenty, modularita, formát UI šablon, organizace kódu, dokumentace, učící křivka a komplexita či mohutnost daného rámce – tedy zda je vhodný spíše pro menší projekty, nebo pro velké firemní aplikace.

Zend

Zend[2] je robustní framework, doporučovaný spíše pro větší projekty. Nabízí velké množství komponent, má rozsáhlou dokumentaci a je i rychlý. Nevýhodami je pomalá učící křivka a komplikovanost.

Symfony

Nejdůležitější výhodou tohoto rámce[3] je rozsáhlá databáze existujících komponent. Tyto komponenty jdou jednoduše nainstalovat s manažerem závislostí, který se nazývá Composer. Má svůj šablonový systém Twig a využívá ORM Doctrine.

CodeIgniter

Tento framework[4] vydaný v roce 2006 není striktně založený na MVC. Je velmi štíhlý, tedy vhodnější spíše pro menší projekty, ale umožňuje přidávat rozšíření třetích stran. Bohužel už dlouho nebyl aktualizován, nemá integrovaný Composer a nenabízí možnosti využití příkazové řádky.

CakePHP

CakePHP[5] poskytuje mnoho zabudovaných bezpečnostních funkcí, má vlastní ovládání příkazové řádky za pomoci Bake, využívá Composer umožňuje přidávat rozšíření třetích stran. Myšlenkově vychází z Ruby on Rails. Velkým plusem rovněž je, že má velmi dobře zpracovanou dokumentaci a dokonce umožňuje stažení učebnice Cookbook zdarma.

[2] https://framework.zend.com/
[3] http://symfony.com/
[4] https://codeigniter.com/
[5] https://cakephp.org/

Yii

Yii[6] je velmi výkonný framework, je rychlejší než všechny ostatní, protože využívá techniku lazy loading. Poskytuje jasně organizovaný a logický kód. Součásti je i generátor kódu Gii. Pro zobrazení používá čisté HTML a je integrován s jQuery.

Laravel

Laravel[7] byl vydán v roce 2011 a je postaven na komponentách Symfony a rovněž využívá Composer. Mezi plusy patří odlehčený šablonový systém Blade, lokální vývojové prostředí Homestead a možnost generování přes příkazovou řádku.

Slim

Slim[8] je tzv. mikro framework. Je velmi minimalistický a je vynikající pro menší aplikace. Jeho nejčastější využití je pro vývoj API a REST služeb.

FuelPHP

Posledním aplikačním rámcem je FuelPHP[9]. Tento framework podporuje i rozvinutou verzi HMVC a přidává novou architektonickou vrstu Presenter, který obsahuje logiku pro generování pohledů.

5.2 Laravel

Tento framework jsem zvolila, protože podporuje Composer, generování přes příkazovou řádku, má jednoduchý šablonový systém, integrované ORM řešení Eloquent a především srozumitelnou dokumentaci, a to nejenom v textové podobě, ale i jako sérii videí. Zároveň není přespříliš komplexní a vyhovuje požadavkům na implementaci menšího projektu.

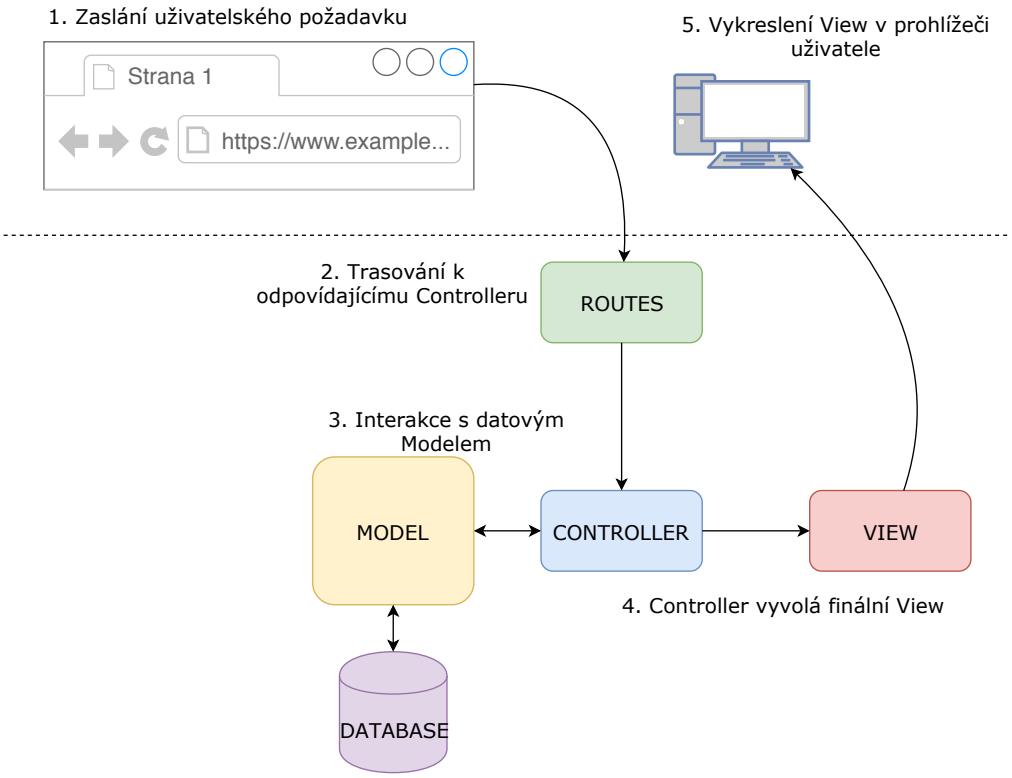
Pro snadnější naučení Laravelu jsem absolvovala dva kurzy na LinkedIn Learning a to MVC Frameworks for Building PHP Web Applications[10] a Laravel 5 Essential Training: 2 Testing, Securing, and Deploying[11].

5.2.1 Zpracování uživatelského požadavku z pohledu Laravelu

Nákres 5.6 objasňuje základní funkcionalitu Laravelu na příkladu uživatelského požadavku.

Uživatelův požadavek [3], tedy například zadání URL adresy či odeslání formuláře kliknutím na tlačítko, vstoupí nejprve do `public/index.php` souboru, kde načte třídy z `autoload.php`, což je soubor generovaný nástrojem Composer. Následně získá instanci aplikaci od Laravelu, protože `bootstrap/start.php` soubor vytvoří aplikaci a nastaví běhové prostředí, a zavolá Kernel. Kernel má na starosti zabezpečení, jako například kontrolu autentifikace uživatelů middleware, nebo kontrolu přítomnosti CSRF tokenu, dále má na starosti také sessions a další konfiguraci. Dále registruje a načte všechny poskytovatele

[6] https://www.yiiframework.com/
[7] https://laravel.com/
[8] https://www.slimframework.com/
[9] https://fuelphp.com/
[10] https://www.linkedin.com/learning/laravel-5-essential-training-1-the-basics
[11] https://www.linkedin.com/learning/laravel-5-essential-training-2-testing-securing-and-deploying



Obrázek 5.1: Cyklus dotazu v Laravelu [zdroj vlastní]

služeb (service providers). Nakonec se požadavek předá do routeru, který najde zadanou cestu. Definované trasy jsou v `routes/web.php`. Jak je vidět v příkladu 5.1, v definici trasy lze využívat klasické HTTP požadavky GET, POST apod. Dále lze využít seskupení tras, v konkrétním případě `middleware` kontroluje, že je uživatel přistupující na stránku přihlášen, nebo zda má administrátorské oprávnění [4]. `Middleware` slouží jako filtr příchozích požadavků. Objekt s dotazem je předán metodě řídící vrstvy, která byla specifikována u dané trasy.

```
Route::middleware('auth')->group(function () {
    Route::group(['middleware' => 'App\Http\Middleware\AdminMiddleware'], function () {
    Route::get('/users', 'UserController@show');
    Route::get('/users/{user_id}/delete', 'UserController@delete'));
    Route::get('/users/{user_id}/restore', 'UserController@restore');
    });
Route::get('/project/new', 'ProjectController@newProject')->name('new_project');
Route::post('/project/new', 'ProjectController@newProject')->name('new_project');
});
```

Ukázka kódu 5.1: Příklad trasování

Controller obsahuje veškerou logiku aplikace. Zpracovává získaný požadavek, interaguje s modelem a spravuje přes něj data v databázi. Po zpracování dat přesměruje aplikaci na finální view, kterému předá i potřebné argumenty s daty. Odpověď se následně zašle klientovi a v jeho prohlížeči se vykreslí daná prezenční vrstva.

5.2.2 Composer

Composer je multiplatformní nástroj pro snadnou správu závislostí v projektu. Závislosti jednotlivých knihoven jsou definovány v souboru `composer.json` v kořenové složce aplikace. Obsažené knihovny lze pak jednoduše přes příkazovou řádku nainstalovat, či aktualizovat jejich verze. Veškeré závislosti jsou uloženy ve složce `vendor`.

5.2.3 Eloquent ORM

Nadstavbou nad databází je Eloquent ORM [2], kde ORM je zkratka z object relation mapper, tedy objektové relační mapování. Jedná se tedy o abstrakci nad databází, která mapuje vztahy objektů v aplikaci. Dalšími příklady ORM nástrojů jsou například Doctrine, používaná spíše pro větší projekty, nebo Propel či RedBean. Eloquent byl speciálně vyvinut pro Laravel, ale je možné ho využít i mimo tento framework. Používá návrhový vzor ActiveRecord, který zajišťuje celkovou správu dat v databázi, tedy jejich čtení, vložení, aktualizaci i smazání. Základem ORM je, že pro každou databázovou tabulku existuje daný model. Dále dokáže Eloquent velmi jednoduše zpracovat nejenom vztahy 1:1 a 1:M, ale i M:N viz příklad 5.2. Zde je vidět použití metod `belongsTo` a `hasMany` či `hasOne`, které určují vazbu mezi danými modely a automaticky přiřazují cizí klíč.

```
class Project extends Model {
    public function owner() {
        return $this->belongsTo('App\User', 'owner', 'id');
    }

    public function users() {
        return $this->belongsToMany('App\User', 'users_projects');
    }

    public function risks() {
        return $this->hasMany('App\Risk');
    }

    public function attributes() {
        return $this->hasOne('App\Attributes');
    }
}
```

Ukázka kódu 5.2: Příklad modelu projektu a jeho vztahů

5.2.4 Blade

Blade [1] je jednoduchý šablonový systém, který na rozdíl od jiných nezakazuje používání čistého PHP kódu v prezenční vrstvě. Naopak všechny views jsou přeloženy do PHP a uloženy v mezipaměti aplikace, dokud nejsou modifikovány. Základní šablona se nachází v `views/layouts/app.blade.php`. Obsahuje hlavičku stránek, tedy veškeré meta tagy, odkazy na externí i interní CSS a JS soubory. Dále může obsahovat například hlavní navigační panel a patičku, které se zobrazují na všech stránkách stejně. Pomocí dvojice příkazů `@yield` a `@section` je možné poté vložit obsah jednotlivých stránek, které používají příkaz `@extends('layouts.app')`.

5.2.5 Homestead prostředí

Laravel Homestead je oficiální předpřipravený Vagrant box, který poskytuje lehce nastavitelné lokální prostředí pro vývoj projektu. Vagrant[12] poskytuje jednoduchý a elegantní způsob, jak spravovat vývojové prostředí na virtuálních strojích bez nutnosti složitého nastavování webového serveru na lokálním počítači a bez nutnosti instalovat další nutný software. Díky použití prostředí Homestead je instalace aplikace pro nového uživatele otázkou několika minut. Podrobný popis instalace je k dispozici v příloze D.

5.2.6 Migrace a seeding

Pomocí migrací lze snadno sdílet a hlavně modifikovat schéma databáze. Schémata zde nejsou pouze uložena, ale při zavolání příkazu migrate je databáze i dle definovaných schémat vybudována. Dále je vhodné pro testovací účely aplikaci naplnit testovacími daty. Z tohoto důvodu jsou implementovány seeder třídy, které jsou stejně jako migrace uloženy ve složce `database`. Jedním ze způsobů naplnění dat je strojové naplnění konkrétními daty, tedy přímé vložení konkrétních dat do databáze. Tohoto principu v aplikaci využívám pro konkrétní testovací uživatelské účty. Druhý způsob využívá Model factories, které umí vygenerovat větší množství náhodně vygenerovaných záznamů. Pořadí volání jednotlivých seeder metod se určuje v souboru `DatabaseSeeder.php`. Příklad volání factory metody `create()` je v příkladu 5.3. Factory jsou definovány v souboru `AppFactory.php` a ukázka definice factory pro projekt je v ukázce 5.4.

```
class ProjectsTableSeeder extends Seeder  {
    public function run()  {
        factory(App\Project::class, 20)->create();
    }
}
```

Ukázka kódu 5.3: Příklad použití factory

```
$factory->define(App\Project::class, function (Faker $faker) {
    return [
        'name' => $faker->word,
        'description' => $faker->text,
        'modify' => 0,
        // created managers are between id 10 to 20
        'owner' => $faker->numberBetween($min = 10, $max = 20)
    ];
});
```

Ukázka kódu 5.4: Příklad factory pro generování instancí modelu Project

5.3 Implementace

Aplikace byla implementována v jazyce PHP s pomocí aplikačního rámce Laravel. Šablonový systém Blade založený na HTML a CSS využívá navíc designový framework Bootstrap. Pro databázi se využívá MySQL a nadstavba ORM Eloquent. V průběhu implementace byl využívám verzovací systém Git a JetBrains PhpStorm IDE.

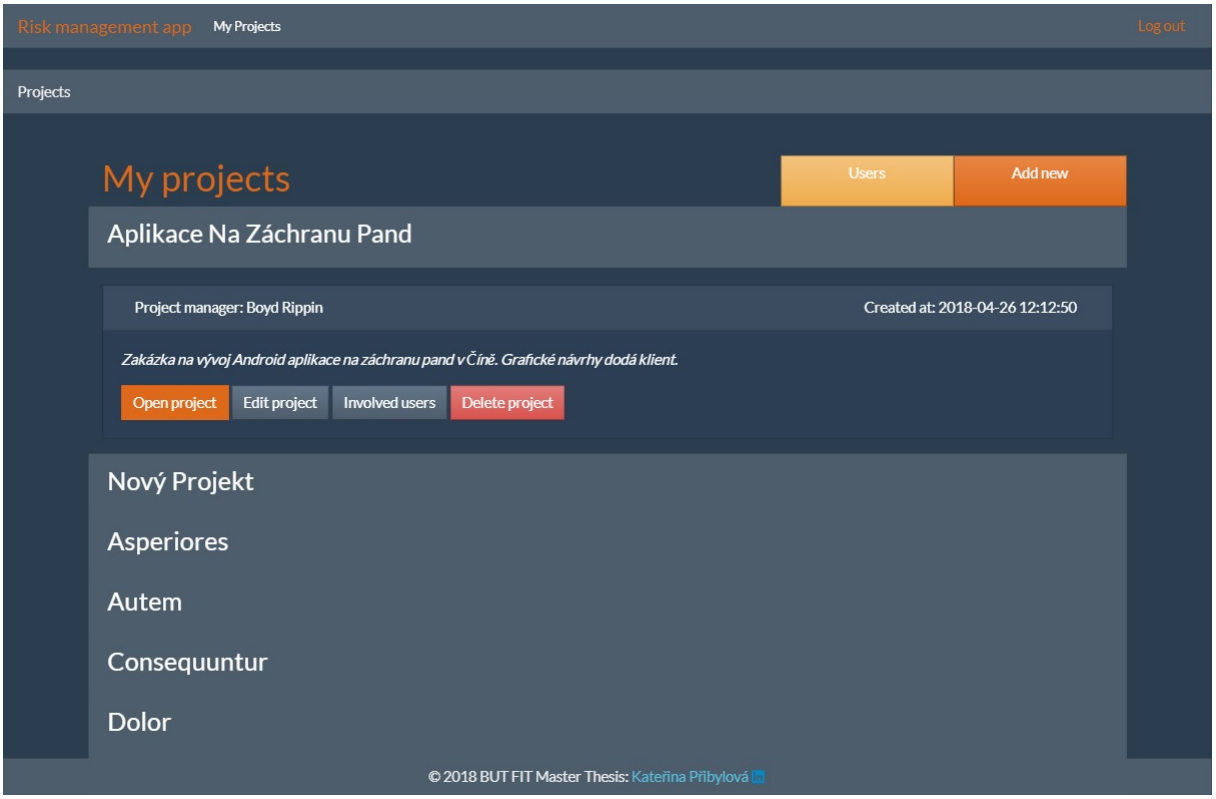
[12] https://www.vagrantup.com/

V následující části práce jsou uvedeny detaily implementace, rozdělení vychází z případů použití a simuluje uživatelův průchod aplikací. Nejprve se práce zaměřuje na přihlášení, registraci a správu uživatelů. Poté je rozvedeno zpracování správy projektů a rizik včetně registru rizik a vizualizací. Poslední část se věnuje provedenému rozšíření aplikace o projektové strategie.

5.3.1 Přihlášení a správa uživatelů

Přihlášení a registraci uživatelů řeší elegantním a jednoduchým způsobem sám Laravel. Pomocí příkazu `php artisan make:auth` se vytvoří předpřipravené controllery, views a do routes se přidají související trasy. Zároveň je vyřešeno i zapomenutí a resetování hesla. Přes fasádu `Auth` lze poté přistupovat ke konkrétní instanci přihlášeného uživatele a tím například hlídat, že je přihlášený.

Po přihlášení se zavolá metoda `mainScreen` z `ProjectController`, která vyhledá a odešle data o projektech, které uživatel buď řídí, pokud je manažer, nebo do kterých je přiřazen. Zobrazí se view `project/mainScreen` s výpisem projektů nebo s oznámením, že uživatel nemá žádný projekt. Pokud je uživatel administrátor, vypíší se mu všechny existující projekty a zároveň je zobrazeno tlačítko vedoucí do správy uživatelů. Tuto obrazovku lze vidět na obrázku 5.2. Možnost projekt editovat, přidávat zainteresované uživatele a projekt smazat má pouze manažer daného projektu a administrátor. Uživatel nebo manažer, který není vlastníkem, mají oprávnění pouze projekt zobrazit.



Obrázek 5.2: Ukázka výpisu projektů pro roli Administrátor [snímek obrazovky aplikace]

Správa uživatelů se řídí pomocí metod v `UserController`. Administrátorovi se zobrazí jména, e-mailové adresy a datum registrace všech uživatelů. Dále je zde rozlišeno, jaké má

uživatel role a zda byl smazán ze systému. Uživatel totiž není přímo smazán z databáze, ale využívá se Eloquent rozhraní `SoftDeletes`, které přidá atribut `deleted_at` a používá funkci `restore()` pro obnovení zvoleného uživatele. Pomocí `withTrahed` lze volat dotaz nad databází, který zahrne i smazané uživatele, jinak jsou standardně z dotazů automaticky vynecháni. Dále je zde rovněž možné měnit uživatelské role.

5.3.2 Správa projektů

Při tvorbě nového projektu se volá metoda `NewProject`, která je rozdělena na dvě části. První část se použije v případě GET požadavku, a ta pouze načte stránku s formulářem. Druhá část se vykonává pouze, pokud jde o požadavek POST. Při odeslání formuláře je nejprve potřeba odeslaná data zvalidovat. K tomuto účelu slouží Laravel metoda `validate()`, která umožňuje jednotlivé validační atributy zanořovat. Ukázka validace je v příkladu 5.5. Poté se vytvoří nová instance třídy konkrétního modelu, v tomto případě projektu, a ta se naplní daty z formuláře. Pomocí Eloquent je následně vše uloženo do databáze. Stejný princip je použit pro vytváření i modifikaci rizik a strategií.

```
$this->validate(
    $request,
    [
        'name' => 'required|max:50',
        'description' => 'required|min:20|max:1600'
    ]
);
```

Ukázka kódu 5.5: Příklad validace formuláře

Během tvorby projektu se určuje, které unikátní atributy bude možné u rizik v daném projektu určovat. Pole těchto atributů, ze kterých projekt manažer vybírá, je ve formátu JSON uloženo v třídě `AttributeList`, která rozšiřuje třídu `ReadOnlyBase`. `ReadOnlyBase` je třída, která implementuje metody, které načtou hodnoty z konkrétních podtříd a předají jejich hodnoty. Kromě pole atributů pracuje třída `ReadOnlyBase` i s daty pro dotazník a kontrolní seznam. Tímto způsobem není nutné tvořit speciální tabulky v databázi a je tedy ušetřeno místo v databázi, ale zároveň jsou tato statická data přehledně uložena a jsou snadno dostupná pro případné změny. V případě atributů projektu jsou konkrétní zvolené atributy zakódovány a uloženy ve tvaru JSON do databáze jako atribut `attributes` v tabulce projektů. Při každém načítání je potřeba atributy nejprve tedy dekódovat.

Při přidávání uživatelů k projektu se manažerovi zobrazí tabulka s uživateli, kteří již do projektu mají přístup. Je zde rozděleno, kteří z nich jsou uživatelé a kteří v dané firmě vystupují jako manažeři, díky tomu lze snadno celý projekt předat pod správu jiného manažera. Sám sebe ovšem manažer v tabulce nevidí. Lze zde uživatele z projektu samozřejmě i odstranit. Pro jejich rychlejší přidání je zde zakomponován několikanásobný select, tedy lze přidat i více uživatelů najednou.

5.3.3 Detail projektu a registr rizik

Detail projetu je zobrazen na snímku obrazovky 5.3. Stránka je rozdělena na 3 sektory. V levé části je kontrolní menu, v pravé horní části jsou zobrazovány vizualizace a v pravé dolní části je registr rizik. Nejprve se budeme věnovat rizikům. V registru rizik je možnost kompletní správy rizik, tedy jak jejich modifikace, tak smazání. Při tvorbě rizika jsou ve formuláři zobrazeny všechny atributy definované ve specifikaci požadavků 4.2. Rovněž

jsou zde vyžadovány atributy, které byly unikátně definovány pro konkrétní projekt. Pro pravděpodobnost, dopad a závažnost jsou implementovány posuvníky, které automaticky přepočítají hodnotu závažnosti při změně jedné ze závislých hodnot, jak je vidět v příkladu 5.6. Unikátní atributy rizik jsou rovněž ukládány ve formátu JSON.

```
var slider_impact = document.getElementById("range_impact");
var output_impact = document.getElementById("output_impact");
output_impact.innerHTML = slider_impact.value;
slider_impact.oninput = function() {
    output_impact.innerHTML = this.value;
    output_severity.innerHTML = this.value*slider_probability.value;
};
```

Ukázka kódu 5.6: Posuvník

Protože unikátní atributy projektu se mohou měnit i po vytvoření projektu, tedy i po vytvoření rizik, bylo nutné ošetřit všechny možné stavy, které mohou nastat. První situací je, že se některý atribut již nevyžaduje. V tomto případě se atribut u rizik nesmaže z databáze, ale pouze se přestane zobrazovat. V případě, že tento atribut bude znovu platný, budou tak hodnoty u těchto rizik stále vyplněné. Druhým scénářem je, když se přidá atribut nový, jehož hodnotu ovšem již existující rizika nemají zadanou. V tomto případě bude v registru rizik hodnota nevyplněna, a při první modifikaci bude zobrazena primární hodnota.

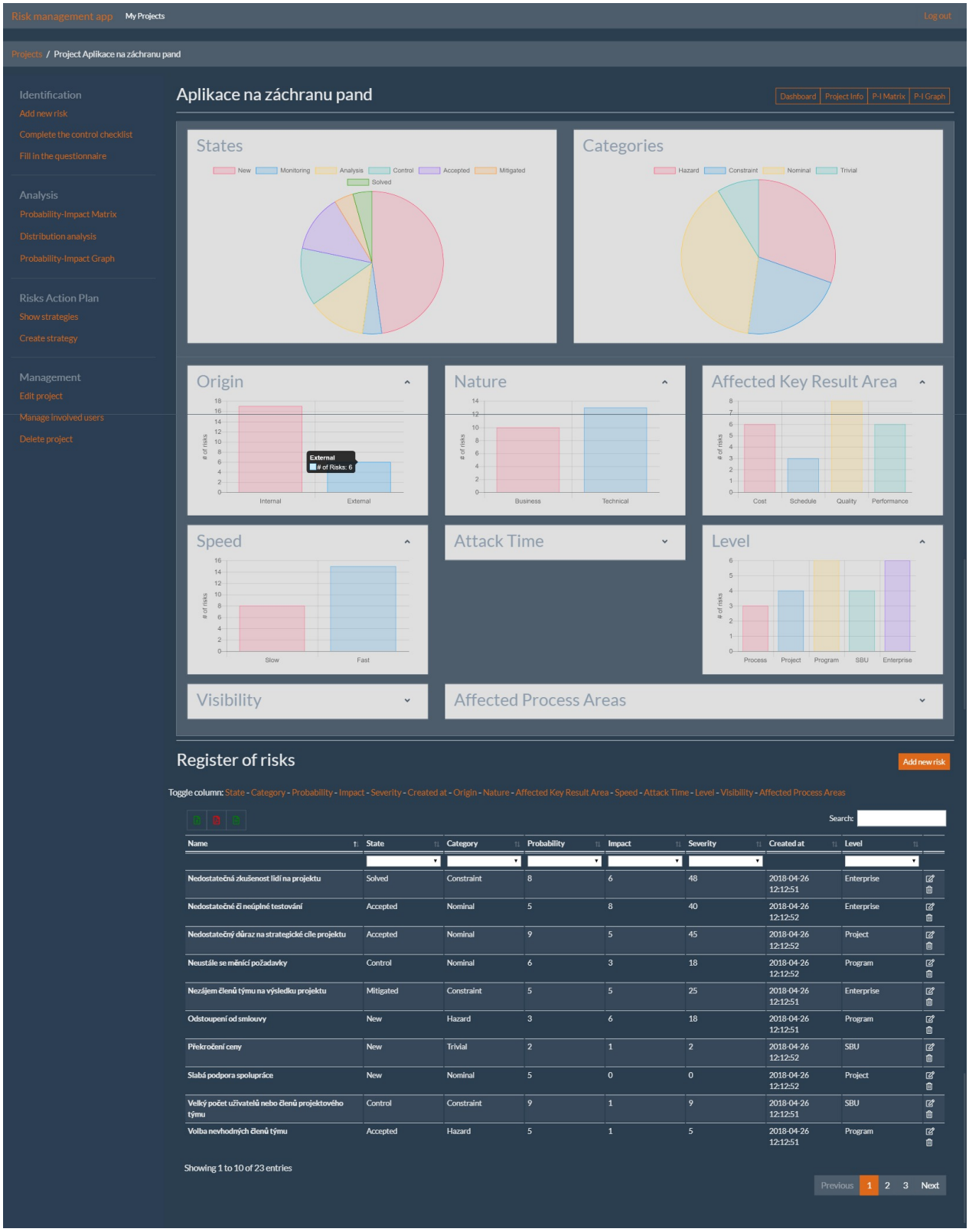
Pro registr rizik byl použit jQuery plugin `DataTables`[13]. Ten zajišťuje filtraci, vyhledávání, řazení a stránkování v registru. Dále je zde možnost schovat či zobrazit sloupce pro větší praktičnost. Standardně jsou skryty sloupce obsahující unikátní atributy. Implementace filtrace a vyhledávání ve sloupci je v příkladu 5.7. Zároveň je přes tento plugin implementován i export do PDF, XLS a CSV.

```
this.api().columns('.filter').every( function () {
  var column = this;
  var select = $('<select class="my_select"><option value=""></option></select>')
    .appendTo( $(column.footer()).empty() )
    .on( 'change', function () {
        var val = $.fn.dataTable.util.escapeRegex(
            $(this).val()
        );

        column
            .search( val ? '^'+val+'$' : '', true, false )
            .draw();
    } );

    column.data().unique().sort().each( function ( d, j ) {
    select.append( '<option value="'+d+'">'+d+'</option>' )
  } );
} );
```

Ukázka kódu 5.7: Vyhledávání a filtrace sloupce

Zobrazení vizualizací je řešeno za pomoci dynamických komponent knihovny `Vue.js`. Vue.js využívá HTML tagy <template> a <component>. V šabloně je HTML obsah dané části stránky a na místo tagu <component> je při zavolání funkce vložena konkrétní šablona. Dále se využívá tagu <keep-alive>, který uchovává v mezipaměti obsah dané šablony, což je velmi užitečné například při vykreslování grafů. Implementace pomocí `Vue.js` je v příkladu 5.8.

[13] https://datatables.net/



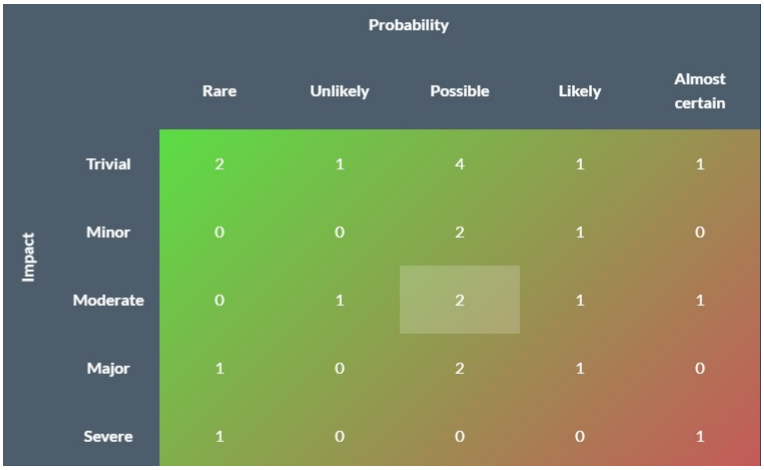
Obrázek 5.3: Detail projektu [snímek obrazovky aplikace]

```
Vue.component('projectInfo', {
    template: '#projectInfo'
});
Vue.component('matrix', {
    template: '#matrix'
});
new Vue({
    el: '#vue',
    data: {
        current: "dashboard"
    },
    methods: {
        switchToMatrix: function () {
            this.current = "matrix"
        },
        switchToProjectInfo: function () {
            this.current = "projectInfo"
        }
    },
    components: ['projectInfo', 'matrix']
})
```

Ukázka kódu 5.8: Dynamické komponenty ve Vue.js

5.3.4 Vizualizace analýzy rizik

Jedna z komponent zobrazuje matici pravděpodobností a dopadů 5.4. Ta ukazuje počet rizik v odstupňovaných kategoriích viz následující tabulka 5.1. Po kliknutí na pole v tabulce je zobrazeno modální okno s výpisem základních informací o rizicích v těchto kategoriích.



Obrázek 5.4: Matice pravděpodobností a dopadů [snímek obrazovky aplikace]

Obsahem komponent jsou rovněž grafy, konkrétně tedy P-I graf a distribuční analýza pro jednotlivá rizika. Grafy jsou vykreslovány pomocí knihovny `Chart.js`[14]. Skript, který generuje daný graf ho vykreslí do HTML elementu s tagem `<canvas>` a konkrétním identifikátorem. Nejobtížnějším krokem je vytvoření datových struktur přesně v definovaném tvaru. Příklad získávání dat pro grafy distribuční analýzy je v příkladu 5.9.

[14] http://www.chartjs.org/

Hodnoty	Probability	Impact
0-1	Rare	Trivial
2-3	Unlikely	Minor
4-6	Possible	Moderate
7-8	Likely	Major
9-10	Almost certain	Severe

Tabulka 5.1: Tabulka s hodnotami pro pravděpodobnost a dopad

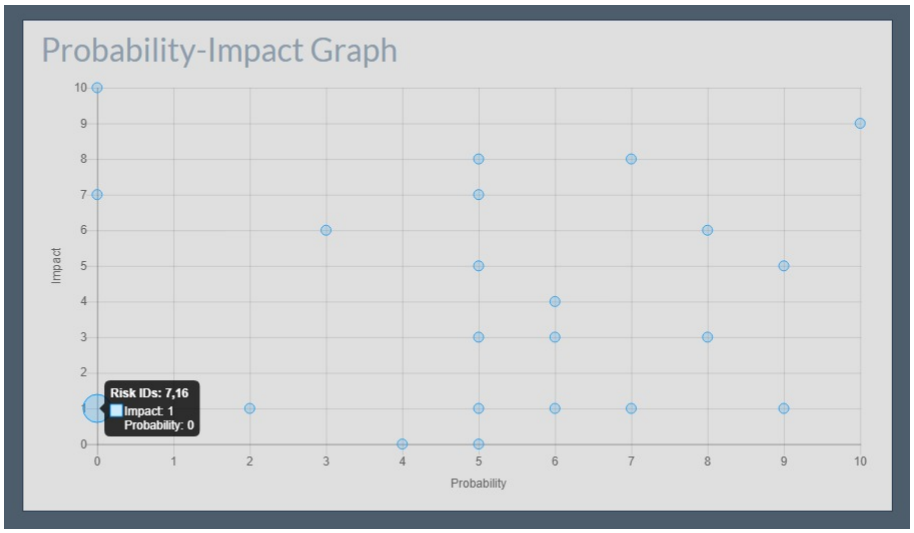
```
// attributes of project
        foreach ($data['attribute_names'] as $project_key => $project_attribute) {
            $index = 0;
            $option_array = [];
            foreach ($project_attribute['options'] as $option) {
                $count = 0;
                // attributes of risks
                foreach ($data['attributes'] as $attribute) {
                    if (array_key_exists($project_key, $attribute)) {
                        if ($attribute[$project_key]['value'] == $option) {
                            $count++;
                        }
                    }

                }
                $option_array[] = $count;
            }
            $attribute_labels[$project_key] = json_encode($project_attribute['options']);
            $attribute_array[$project_key] = json_encode(['data' => $option_array,
                'label' => '# of Risks',
                'backgroundColor' => $backgroundColor,
                'borderColor' => $borderColor,
                'borderWidth' => 1,

            ]);
            $index++;
        }
```

Ukázka kódu 5.9: Příklad získávání dat pro grafy distribuční analýzy

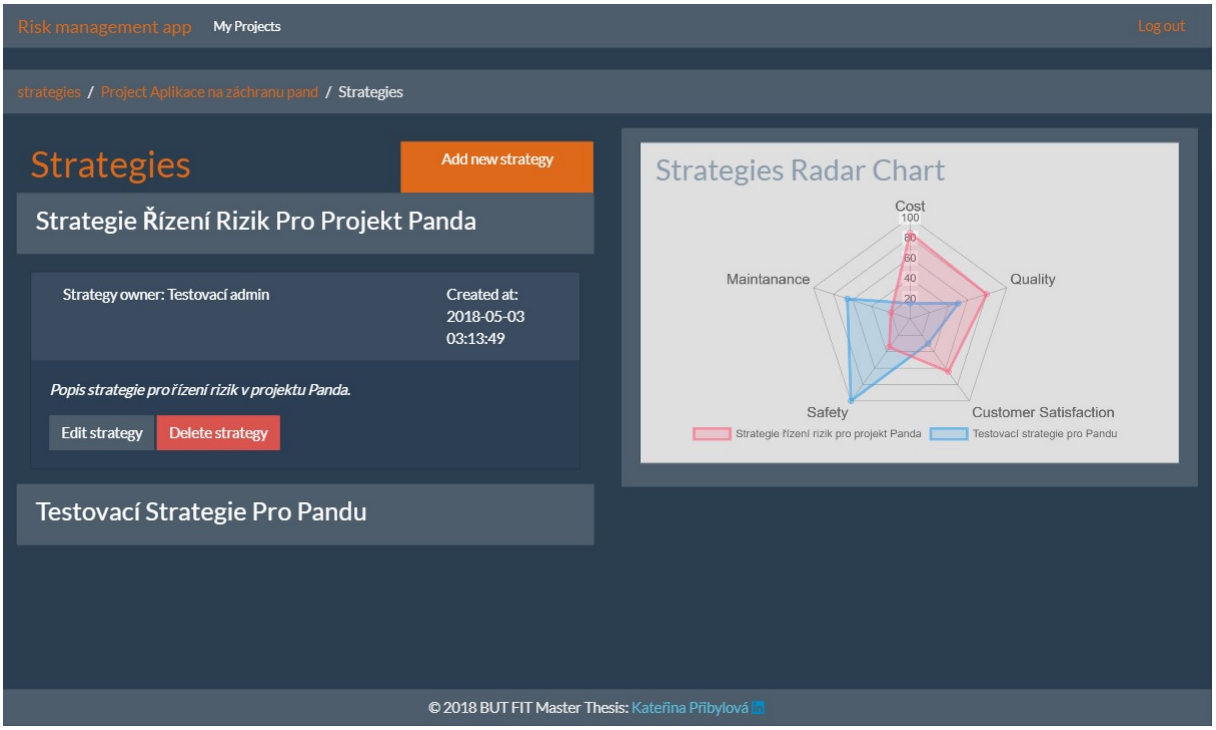
Poslední součástí aplikace jsou notifikace, ty se vykreslí při chybách, či při úspěšném vytvoření, či editaci. Při přesměrování se zároveň odešlou hodnoty, které se uloží do relace (sessions). Následně se podle této hodnoty vykreslí správná barva notifikace včetně předané zprávy. Notifikace jsou definovány v `notification.blade.php`.

Jako rozšíření oproti návrhu byly naprogramovány projektové strategie. Atributy strategií jsou založeny na rozšířeném trojimperativu, tedy cena, kvalita a místo času, jsou zde atributy pro bezpečnost, zákaznickou spokojenost a údržbu. Uživatelé mohou definovat strategie postupu při plánování reakcí a řízení rizik a atributy jednotlivých strategií jsou poté porovnávány bodovacím modelem 3.0.6 v paprskovém grafu.

Celková třídní struktura je zobrazena na diagramech B.2 pro controllery a B.1 pro modely v příloze B.



Obrázek 5.5: Graf pravděpodobností a dopadů [snímek obrazovky aplikace]



Obrázek 5.6:

Kapitola 6

Testování

Před samotným uživatelským akceptačním testováním bylo potřeba, aby byla aplikace velmi komplexně otestována. Po implementaci každého většího celku byly provedeny regresní integrační testy, které zkontrolovaly, že je aplikace funkční, a případné chyby byly ihned opravovány. Regresní testy [10] se využívají při opětovném testování funkcí a vlastností aplikace. Jejich smyslem je ověření, že provedené změny či implementace nových vlastností v aplikaci nemělo žádný vliv na stávající funkce a vlastnosti. Tedy především na oblasti, které zůstaly v programovém kódu nezměněny.

6.1 End-to-end testování

Je obecně známo, že člověk, který zná systém nazpaměť má větší tendence chyby přehlížet. Z tohoto důvodu jsem se rozhodla, že poprosím o pomoc svého bývalého kolegu Ing. Dalibora Klučku, Quality Assurance Test Engineera ze společnosti Avast Software s.r.o. Mým cílem totiž bylo, aby aplikace obsahovala minimální množství chyb. Bylo tedy provedeno end-to-end testování [6], což je testovací metoda, která využívá reálné testovací scénáře od začátku do konce, tedy od přihlášení, přes vykonání akce, až po odhlášení, a testuje, zda aplikace splňuje specifikované požadavky. Testovací sada, vypracovaná pro účely tohoto testování, je v příloze C. Testovací případy pokrývají 100% případů použití. Kromě těchto testů byly provedeny i negativní a bezpečnostní testy. Součástí negativních testů je například zadávání nevalidního tvaru e-mailu, nesprávných či prázdných hodnot do polí formuláře, zadávání nevalidních znaků nebo naopak zadávání hodnot s nadměrným množstvím znaků. Z bezpečnostního hlediska byla aplikace otestována na XSS (cross site scripting) a SQL injection. Výstupem z testování byl rozsáhlý bug report, na jehož základě byly provedeny potřebné opravy.

6.2 Uživatelské akceptační testování

Poté, co byly opraveny všechny nalezené chyby a aplikace byla znovu přetestována, byla připravená pro uživatelské akceptační testování. UAT [5], neboli uživatelské akceptační testování, je poslední fází testování softwaru. Během této fáze reální uživatelé zkouší používat aplikaci a rozhodují, zda aplikace splňuje jejich vize a požadavky. Aplikace byla předvedena třem odborníkům v oboru. Zajímal mě především jejich názor na používaní aplikace s ohledem na jejich zkušenosti v řízení projektů. Níže uvádím jejich recenze.

„Z pozice project managera ve společnosti Dactyl Group zabývající se vývojem softwaru hodnotím Risk management app jako užitečný nástroj, který by se dal v praxi využít pro přehlednou evidenci a analýzu rizik na jednotlivých projektech. Kladně hodnotím možnost vydefinovat si širokou škálu parametrů, které souvisí s riziky a jejich následné analyzování v grafech. Velmi také oceňuji technické zpracování projektu, které je na dobré úrovni a přispívá tak k celkové použitelnosti aplikace. Zároveň bych ale také ocenil, kdyby celá aplikace mohla být jako modul do zavedených nástrojů na řízení projektů, jako jsou například Jira nebo Easy Redmine, aby se dala lépe a rychleji integrovat do firemních procesů. Do budoucna by se mi líbila možnost předdefinovat si rizika a následně je pouze vybírat a přiřazovat k jednotlivým projektům. Dosáhli bychom tak úspory času a zároveň by tato funkce mohla sloužit jako prevence možného opomenutí rizik."

—Ing. Milan Doubek, Project Manager ve firmě Dactyl Group s.r.o.

„Věnuji se koordinaci a řízení malých a středních softwarových projektů v agilním prostředí, kde má správa rizik specifická kritéria. Projekty totiž nejsou dostatečně uniformní pro globální správu a mění se velice rychle. Proto mi produkty dostupné na trhu spíše přidělávaly problémy, než aby mi s riziky pomohly. Na webovém řešení RiskManagement právě oceňuji důraz na agilní vývoj a jednoduchý přístup k rizikům, kde aplikuji základní metody a modely pro jejich analýzu. Lze předpokládat, že by to mohlo narazit v korporátním prostředí, ale pro malé a střední projekty je toto řešení dostačující a účinné. Dobře patrná je i provedená analýza stávajících produktů a vyvarování se nalezených chyb, či naopak inspirace dobrými příklady. Obecně můžu říct, že RiskManagement je dobrý vstupní bod pro řízení rizik při vývoji softwaru."

–Ing. Jiří Semmler, bývalý Project Manager ve firmách Mathesio s.r.o. a Dactyl Group s.r.o.

„Z môjho pohľadu sa jedná o prehľadný a jednoduchý nástroj, ktorý je efektívnou pomôckou pri riadení rizík a jeho vizualizácii. Rozhranie ponúka dobré možnosti pre definovanie a kvantifikovanie rizík, pričom umožňuje adekvátne sledovať ich stav a vlastnosti naprieč celým životným cyklom. To celé je podporené vhodne zvolenou vizualizáciou dát, ktorá urýchľuje orientáciu medzi uloženými informáciami a tým pádom vedie k ich efektívnejšiemu spracovaniu. Z týchto dôvodov považujem nástroj za veľmi vhodný na použitie pri riadení rizík v malej firme."

–Bc. Denis Galajda, Project Manager ve firmě Dactyl Group s.r.o. a Public Relations Manager v IAESTE Czech Republic

6.3 Uživatelské testování použitelnosti

Protože mým cílem bylo, aby aplikace byla snadno použitelná a přehledná, rozhodla jsem se podrobit ji i testování použitelnosti. Vybrala jsem nejzákladnější metodu, tedy sledování uživatele při plnění jednoduchých úkolů. Účastník se mnou zároveň sdílel jeho myšlenky a pocity, při procházení aplikace. Výstupem tohoto testování byly poznámky se seznamem chyb, které vznikly pozorováním subjektů. Poznatky a chyby z tohoto testování byly rovněž patřičně opraveny ve finální verzi aplikace.

Kapitola 7

Zhodnocení a možnosti rozšíření

Cílem diplomové práce bylo navrhnout a vytvořit systém pro správu rizik, který bude vhodný hlavně pro agilní přístup a spíše pro menší až střední firmy, pro které jsou současná řešení na trhu zbytečně komplikovaná. Výsledná aplikace splňuje požadavky zadefinované ve specifikaci, je plně otestována a je připravená pro případně použití v praxi. Aplikace působí přehledně a není zbytečně komplikovaná. Z hlediska použití v rámci projektového řízení splňuje představy projektových manažerů, kteří působí v agilním prostředí v menších až středních firmách.

Aplikace poskytuje jednoduchý způsob, jak spravovat rizika v průběhu celého cyklu jejich řízení. Napomáhá uživatelům rizika plánovat pomocí strategií a bodovacího modelu, napomáhá je definovat pomocí kontrolních seznamů a dotazníků a napomáhá i jejich analýze pomocí vizualizací, tedy distribuční analýzy, matice a grafu pravděpodobností a dopadů. Dále napomáhá i v reakci a kontrole rizik, kdy je možné plánovat odezvu pomocí plánů a mít o rizicích přehled díky praktickému registru rizik.

7.1 Rozšíření

Pro reálné využití aplikace by bylo potřeba odstranit některé z jejích nedostatků. Řešení nedostatků a možností dalšího rozšíření bych rozdělila do několika následujících kategorií.

Rozšíření projektová

Jen ojediněle se stává, že má firma pouze pár projektů. Většinou má jeden manažer na starosti několik projektů, které spolu mohou či nemusí souviset, ale mají například společný rozpočet. Z tohoto důvodu se nabízí rozšíření aplikace o vrstvu projektového portfolia (sada všech projektů v organizaci, nikoliv nutně provázaných) a programu (soubor vzájemně souvisejících projektů). Dále by bylo vhodné zohlednit firemní misi a strategii do proměnných projektu, protože tyto hodnoty by se vždy měly odrážet v jeho řízení. Šlo by rovněž nastavit parametry jako metodika řízení rizik, specifické role a povinnosti jednotlivých uživatelů apod.

Rozšíření riziková

Co se rozšíření možností práce s riziky týče, tak největším rozšířením by bylo přidání dalších metod analýzy a jejich vizualizace, například rozhodovací analýzy či simulace Monte Carlo, které by mohly určovat nejvýhodnější definovanou strategii. Pro identifikaci rizik by bylo

velmi žádané rozšíření propojení s aplikací, která umožňuje snadnou práci s mind mapami. Na základě uživatelského testování by bylo rovněž vhodné doplnit možnost si předdefinovat rizika, která se v projektech nejčastěji opakují.

Ostatní rozšíření

V případě reálného nasazení systému by bylo vhodné zajistit profesionální grafický návrh. Rovněž by bylo vhodné doplnit automatizované testy, které by usnadnili testování. Například pomocí Laravel Dusk[1], který nepotřebuje instalovat ani tradiční Selenium a používá se pro jednoduché automatizované testy fungující přímo v prohlížeči. Z uživatelského testování vyplývá, že by rovněž bylo zajímavé upravit aplikaci na modul, který by mohl spolupracovat s existujícími systémy pro řízení projektů.

[1] https://laravel.com/docs/5.6/dusk

Kapitola 8

Závěr

Cílem mé diplomové práce bylo na základě nastudovaných informací vytvořit návrh a následně prototyp aplikace pro správu a vizualizaci rizik. První důležitým bodem bylo seznámení se s problematikou managementu rizik při vývoji softwarových projektů. Konkrétně se zaměřením na metody, kterými se rizika identifikují, analyzují a jak je možno je vizualizovat. Tomuto tématu se věnuje kapitola 2. V této kapitole byl čtenáři vysvětlen význam řízení rizik a jeho pozice v rámci projektového managementu. Dále byly podrobně představeny jednotlivé fáze řízení rizik a metody, které se při jednotlivých fázích používají. Práce navíc obsahuje i průnik s velmi aktuálním tématem agilních metodik, tedy jak je vhodné nahlížet na řízení rizik tak, aby dodržovalo základní zásady agilního vývoje a naopak napomáhalo zvyšovat hodnotu a usnadňovalo týmovou práci na projektu.

V kapitole 3 jsem se zabývala konkrétními metodami, které se využívají nejenom při analýze rizik, ale obecně při jakémkoliv rozhodování během řízení projektů. Jednalo se o rozhodovací stromy, analýzu citlivosti, bodovací model pro rozhodování za použití více kritérií a o simulaci Monte Carlo a s ní spojené statistické funkce a pravděpodobnostní rozdělení.

Další část se v kapitole 4 zabývá specifikací a návrhem požadavků na systém, který by pomáhal s řešením problematiky správy rizik a umožňoval by jejich vizualizaci. Systém by měl být uživatelsky přívětivý, jednoduchý na pochopení a měl by být multiplatformní, tedy použitelný na většině obvyklých zařízení. Měl by korespondovat s názory agilních metodik a doplňovat jejich nástroje. Výsledná aplikace by měla napomáhat nejenom při analýze rizik, ale měla by pomáhat i s identifikací a kontrolou rizik.

Kapitola 5 navazuje na tento návrh a popisuje průběh implementace prototypu navrženého systému. Nejprve byly představeny jednotlivé PHP aplikační rámce a poté byl blíže popsán vybraný PHP framework Laravel. V následující části je aplikace postupně procházena a jsou prezentovány zajímavější implementační detaily.

V kapitole 6 je specifikováno, jak probíhalo testování výsledné aplikace. Bylo provedeno nejenom podrobné funkcionální testování, ale aplikace byla předvedena i několika odborníkům z praxe, kteří aplikaci následně ohodnotili. Zároveň s tím proběhlo i jednoduché uživatelské testování použitelnosti. Veškeré opravy chyb byly zapracovány do finální verze aplikace.

V poslední kapitole 7 je uvedeno hodnocení práce, kde jsou shrnuty požadavky, které práce splňuje a rovněž jsou zde diskutovány další možnosti rozšíření. Tato rozšíření by bylo vhodné realizovat za předpokladu nasazení aplikace do komerčního prostředí.

Literatura

[1] *Blade Templates*. [Online; navštíveno 03.05.2018].
URL https://laravel.com/docs/5.6/blade

[2] *Eloquent*. [Online; navštíveno 03.05.2018].
URL https://laravel.com/docs/5.6/eloquent

[3] *Request Lifecycle*. [Online; navštíveno 03.05.2018].
URL https://laravel.com/docs/5.6/lifecycle

[4] *Routing*. [Online; navštíveno 03.05.2018].
URL https://laravel.com/docs/5.6/routing

[5] *User Acceptance Testing (UAT)*. [Online; navštíveno 04.05.2018].
URL
https://www.techopedia.com/definition/3887/user-acceptance-testing-uat

[6] *Why End to End Testing is Necessary and How to Perform It?* Únor 2018, [Online; navštíveno 04.05.2018].
URL https://www.softwaretestinghelp.com/what-is-end-to-end-testing/

[7] Clemen, R. T.: *Making hard decisions: an introduction to decision analysis*. Duxbury Press, 1996, ISBN 0-534-26034-9.

[8] Garlick, A. R.: *Estimating risk: a management approach*. Gower, 2007, ISBN 978-0-566-08776-9.

[9] Guckenheimer, S.; Perez, J. J.: *Efektivní softwarové projekty*. Zoner Press, 2007, ISBN 978-80-86815-62-6.

[10] Hlava, T.: *Regresní testy*. Srpen 2011, [Online; navštíveno 04.05.2018].
URL http://testovanisoftwaru.cz/tag/regresni-testy/

[11] Koller, G.: *Risk assessment and decision making in business and industry: a practical guide*. Chapman and Hall/CRC, 2005, ISBN 1-58488-477-0.

[12] Layton, M.: *Agile project management for dummies*. Wiley, 2012, ISBN 9781118235850.

[13] Machač, M.: *10 nejlepších PHP frameworků pro vývojáře*. Říjen 2015, [Online; navštíveno 02.05.2018].
URL https:
//www.interval.cz/clanky/10-nejlepsich-php-frameworku-pro-vyvojare/

[14] *Řízení rizik (Risk Management)*. Leden 2016, [Online; navštíveno 25.11.2017]. URL https://managementmania.com/cs/rizeni-rizik

[15] Pandian, C. R.: *Applied software risk management: a guide for software project managers*. Auerbach/Taylor&Francis, 2007, ISBN 9780849305245.

[16] Schuyler, J.: *Risk and decision analysis*. Project Management Institute, 2001, ISBN 188041028-1.

[17] Schwalbe, K.: *Řízení projektů v IT*. Computer Press, 2007, ISBN 978-80-251-1526-8.

[18] Virine, L.; Trumper, M.: *Project decisions: the art and science*. Management Concepts, 2008, ISBN 978-1-56726-217-9.

[19] Vose, D.: *Risk analysis: a quantitative guide*. Wiley, 2008, ISBN 978-0-470-51284-5.

Příloha A

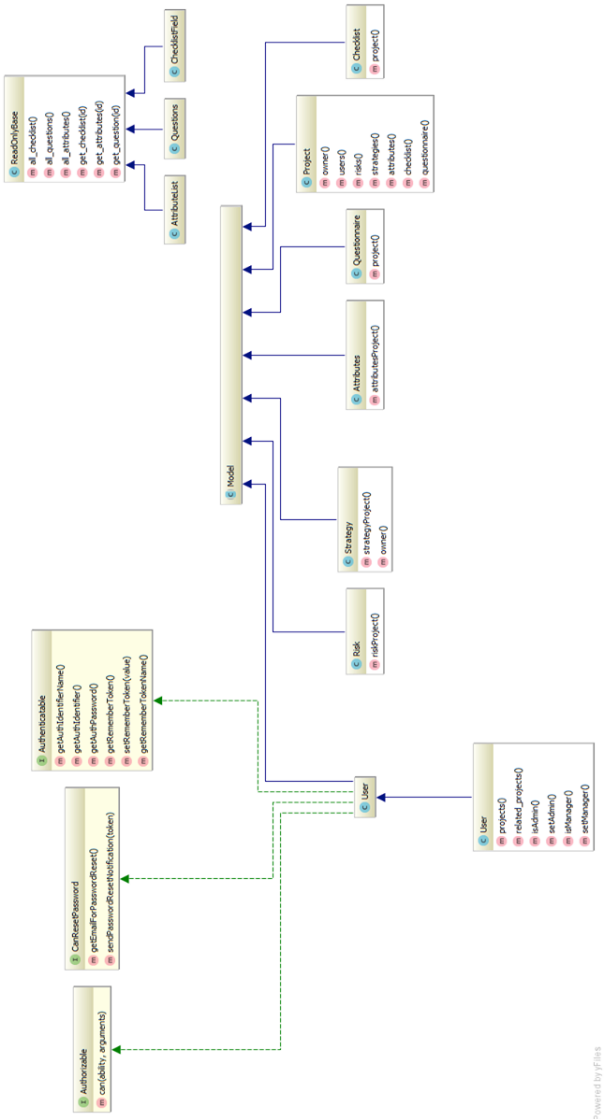
Šablona pro identifikaci rizik v softwarových projektech

- Byznys modelování a požadavky:
  - zřejmé podnikatelské cíle,
  - sběr požadavků,
  - posouzení požadavků,
  - změny požadavků,
  - akceptace požadavků,
  - smlouva.
- Analýza a design:
  - architektura,
  - technologické možnosti,
  - nové technologie,
  - interpretace a analýza požadavků,
  - design.
- Implementace:
  - tvorba kódu,
  - unit testování,
  - integrace,
  - modifikace.
- Kontrola kvality:
  - hodnocení,
  - testování,
  - akceptační testování.
- Nasazení a údržba:

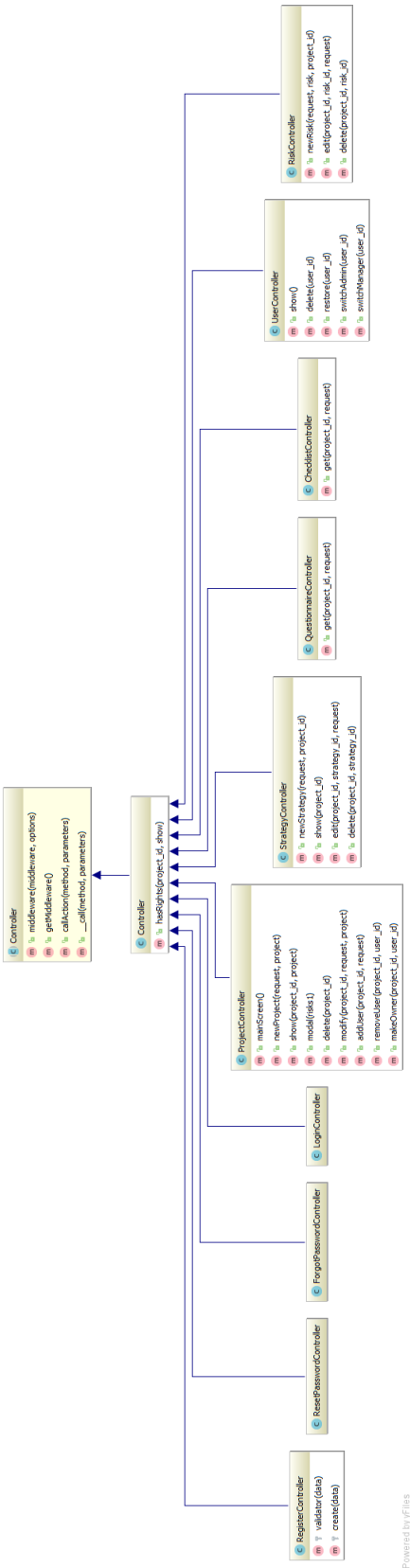
  - nasazení,
  - údržba,
  - instalace,
  - vylepšení a růst.
- Konfigurace a řízení změn:
  - řízení konfigurací včetně procesu kompilování nové verze,
  - řízení změn,
  - změna rozsahu a cílů.
- Projektové řízení:
  - rozvoj řízení projektu,
  - závazky vedení,
  - účast klienta,
  - technická výkonnost,
  - řízení výdajů.
- Prostředí:
  - Vývojářské prostředí:
    * software a nástroje,
    * hardware.
  - Prostředí organizace:
    * zkušenosti manažera,
    * stabilita firmy,
    * zkušenosti organizace v konkrétním projektu,
    * externí vztahy,
    * outsourcing.
  - Zdroje:
    * dostupnost zdrojů,
    * použitelnost zdrojů,
    * výkonnost zdrojů,
    * obrat zdrojů.
  - Ostatní:
    * přírodní prostředí,
    * politické a právní prostředí,
    * kulturní a sociální prostředí,
    * problémy v závislosti na poloze - infrastruktura a zařízení.

Příloha B

Diagramy tříd



Obrázek B.1: Diagram tříd modelů



Obrázek B.2: Diagram tříd controllerů

Příloha C

Testovací sada

	Krok	Akce	Vstupní data	Očekávaný výsledek
	1.	Přihlas se s administrátorským účtem	e-mail: admin@riskmanagement.com heslo: admin123	Zobrazí se stránka se seznamem projektů
oprávnění uživatelů	2.	Klikni na tlačítko "Users" pro správu všech uživatelů registrovaných v systému		Zobrazí se seznam všech registrovaných uživatelů. U každého uživatele je zobrazeno celé jméno, e-mailová adresa zadaná při registraci, datum a čas registrace a role uživatele.
	3.	Nastav uživateli s běžným oprávnění manažerská práva kliknutím na ikonu manažera na konci řádku uživatele	Uživatel: Mr. Garrick Ferry e-mail: watsica.brett@example.net	Role uživatele úspěšně změněna. U uživatele je nyní označena role manažera.
	4.	Odeber uživateli z předchozího kroku manažerská práva kliknutím na ikonu manažera na konci řádku uživatele	Uživatel: Mr. Garrick Ferry e-mail: watsica.brett@example.net	Role uživatele úspěšně změněna. U uživatele není označena role manažera a ani žádná jiná role
	5.	Nastav uživateli s běžným oprávnění administrátorksá práva kliknutím na ikonu administrátora na konci řádku uživatele	Uživatel: Mr. Garrick Ferry e-mail: watsica.brett@example.net	Role uživatele úspěšně změněna. U uživatele je nyní označena role administrátora
	6.	Odeber uživateli z předchozího kroku administrátorská práva kliknutím na ikonu administrátora na konci řádku uživatele	Uživatel: Mr. Garrick Ferry e-mail: watsica.brett@example.net	Role uživatele úspěšně změněna. U uživatele není označena role administrátora a ani žádná jiná role
	7.	Smaž uživatele kliknutím na ikonu smazání.	Uživatel: Mr. Garrick Ferry e-mail: watsica.brett@example.net	Zobrazí se vyskakovací okno vyžadující potvrzení smazání uživatele.
	8.	Potvrď smazání uživatele.		Uživatel byl smazán. U uživatele je nyní označen sloupec "Deleted"
	9.	Obnov uživatele kliknutím na ikonu přidání uživatele		Uživatel byl úspěšně obnoven.
	10.	Vrať se zpět na seznam projektů		
uživatlské role v projektu	11.	Klikni na libovolný projekt a pak na "Involved users"		Zobrazí se stránka se seznam všech uživatelů a tabulkou s uživateli zapojenými do projektu
	12.	Přidej do projektu nového uživatele označením jej v seznamu a kliknutím na "Add person"	Jméno: Marielle O'Keefe e-mail: noble62@example.org	Uživatel je přidán do projektu. Jeho jména a e-mail je zobrazen v tabulce uživatelů zapojených do projektu
	13.	Odeber uživatele z projektu kliknutím na ikonu "Remove user" u uživatele přidaného v předchozím kroce.		Zobrazí se vyskakovací okno vyžadující potvrzení.
	14.	Potvrď odebrání uživatele z projektu		Uživatel byl úspěšně odebrán. Není zobrazen v tabulce zapojených uživatelů.
	15.	Přidej do projektu nového manažera vybráním jej ze seznamu a kliknutím na "Add person"	Jméno: Julio Gleason PhD e-mail: qflatley@example.org Jméno: Rosetta Jakubowski PhD e-mail: okon.eldora@example.net	Uživatel je přidán do projektu. Jeho jméno a e-mail je zobrazen v tabulce uživatelů zapojených do projektu
	16.	U nově přidaného manažera klikni na ikonu "Set manager" a změň mu roli na manažera projektu		Zobrazí se vyskakovací okno vyžadující potvrzení.
	17.	Potvrď změnu manažera projektu		Role manažera projektu byla úspěšně změněna
	18.	Odeber předchozího manažera projektu kliknutím na ikonu odebrání uživatele		Zobrazí se vyskakovací okno vyžadující potvrzení.
	19.	Potvrď odebrání uživatele z projektu		Uživatel byl úspěšně odebrán. Není zobrazen v tabulce zapojených uživatelů.
	20.	Odhlas se kliknutím na "Log out"		Uživatel je odhlášen. Homepage stránka je zobrazena.
	21.	Přihlas se s manažerským účtem přidaným v kroku 6.	Jméno: Julio Gleason PhD e-mail: qflatley@example.org	Uživatel je přihlášen. V seznamu projektů vidí projekt, do kterého byl přidán v bodě 6.
	22.	Opakuj body 11. až 19		

Tabulka C.1: Testovací případ pro registraci a přihlášení

	Krok	Akce	Vstupní data	Očekávaný výsledek
registrace	1.	Jdi na homepage a klikni na "Create new account"		Zobrazí se Registrační formulář
	2.	Vyplň svůj validní e-mail a heslo a klikni na tlačítko "Register"	e-mail: risk.management.test@gmail.com heslo: Heslo_123	Uživatel je úspěšně přihlášen. Stránka Projects je zobrazena (žádné projekty nejsou v seznamu, jelikož užvatel ještě nebyl k žádnému přiřazen)
	3.	Odhlas se kliknutím na "Log out"		Zobrazí se homepage, uživatel je odhlášen
přihlášení	4.	Přihlas se vyplněním registrovaného e-mailu a hesla do přihlašovacího formuláře a klikni na "Login" tlačítko	e-mail: risk.management.test@gmail.com heslo: Heslo_123	Uživatel je úspěšně přihlášen. Stránka Projects je zobrazena (žádné projekty nejsou v seznamu, jelikož uživatel ještě nebyl k žádnému přiřazen)
	5.	Odhlas se kliknutím na "Log out"		Zobrazí se homepage, uživatel je odhlášen
reset hesla	6.	Obnov heslo k účtu kliknutím na "Forgot password".		Zobrazí se formulář pro obnovu hesla
	7.	Zadej registrovaný e-mail z předchozích kroků a klikni na "Send password reset link"	e-mail: risk.management.test@gmail.com	Zobrazí se notifikace s informací, že odkaz na změnu hesla byl zaslán na e-mail
	8.	Jdi do mailboxu a najdi Reset password e-mail		E-mail by úspěšně přijat. Obsahuje odkaz na změnu hesla.
	9.	V e-mailu klikni na "Reset password"		Uživatel je přesměrován na Reset password stránku
	10.	Vyplň e-mail z předchozích kroků a nové heslo a klikni na "Reset password"	e-mail: risk.management.test@gmail.com heslo: Nové_heslo_123	Uživatel je úspěšně přihlášen. Stránka Projects je zobrazena (žádné projekty nejsou v seznamu, jelikož uživatel ještě nebyl k žádnému přiřazen)
	11.	Odhlas se kliknutím na "Log out"		Zobrazí se homepage, uživatel je odhlášen
	12.	Přihlas se s novým heslem	e-mail: risk.management.test@gmail.com heslo: Nové_heslo_123	Uživatel je úspěšně přihlášen. Stránka Projects je zobrazena (žádné projekty nejsou v seznamu, jelikož uživatel ještě nebyl k žádnému přiřazen)

Tabulka C.2: Testovací případ pro správu uživatelů

	Krok	Akce	Vstupní data	Očekávaný výsledek
	1.	Přihlas se s manažerským účtem	e-mail: manager@riskmanagement.com heslo: heslo123	Zobrazí se stránka se seznamem projektů
projekt	2.	Vytvoř nový projekt kliknutím na "Add new project"		Zobrazí se stránka s formulářem pro vytvoření nového projektu
	3.	Vyplň název projektu, popis a libovolně zvolené attributy a potvrď	Libovolný text a hodnoty	Projekt byl úspěšně vytvořen. Je zobrazena stránka s detailem projektu. Název projektu odpovídá názvu vyplněném v předchozím kroku. Na stránce jsou vykresleny grafy pro všechny zadané atributy rizik.
	4.	Klikni na "Edit project" pro editaci projektu vytvořeném v předchozím kroce		Zobrazí se stránka s formulářem pro editaci projektu Hodnoty ve formuláři jsou předvyplněné z předchozího kroku.
	5.	Edituj název, popis a změň nastavení atributů a potvrď		Projekt byl úspěšně editován. Je zobrazena stránka s detailem projektu. Název projektu odpovídá změněné hodnotě. V tabulce rizik jsou nyní zobrazený jen atributy uložené v předchozím kroce.
rizika	6.	Klikni na "Add new risk" pro přídání nového rizika do projektu		Zobrazí se stránka s formulářem pro vytvoření nového rizika.
	7.	Vyplň všechny požadované pole ve formuláři, nastav pravděpodobnost, dopad a závažnost a potvrď	Libovolný text a hodnoty	Riziko bylo úspěšně vytvořeno. Je zobrazena stránka s detailem projektu. Vytvořené riziko je nyní uvedeno v seznamu rizik. Název a parametry rizika odpovídají vyplněným hodnotám v předchozím kroku. V jednotlivých grafech jsou vykresleny hodnoty odpovídající zadaným hodnotam při vytváření rizika.
	8.	Klikni na ikonu editace rizika v tabulce rizik v řádku rizika vytvořeném v předchozím kroce		Zobrazí se stránka s formulářem pro editaci rizika. Hodnoty ve formuláři jsou předvyplněné.
	9.	Edituj název, popisy a další parametry rizika a potvrď		Riziko bylo úspěšně editováno. Je zobrazena stránka s detailem projektu. Editované rizika je uvedeno v seznamu rizika jeho parametry odpovídají provedeným změnám.
	10.	Smaž riziko kliknutím na ikonu smazání rizika v tabulce rizik		Vyskočí okno pro potvrzení smazání.
	11.	Potvrď smazání rizika		Riziko bylo úspěšně smazáno. Nenachází se v seznamu rizik a jeho hodnoty nejsou vykereselny v grafech.
strategie	12.	Klikni na "Add new strategy" pro vytvoření nové strategie		Zobrazí se stránka s formulářem pro vytvoření nové strategie.
	13.	Vyplň jméno a popis strategie a všechny další požadované parametry a potvrď	Libovolný text a hodnoty	Strategie byla úspěšně vytvořena. Je zobrazena stránka se seznamen strategií. Vytvořená strategie je uvedena v seznamu se správným názvem a popisem. Na paprskovém grafu jsou vykresleny hodnoty odpovídající zadaným parametrům při vytváření strategie.
	14.	Klikni na "Edit strategy"		Zobrazí se stránka s formulářem pro editaci strategie. Hodnoty ve formuláři jsou předvyplněné.
	15.	Edituj název, popis a parametry strategie a potvrď.		Strategia byla špěšně editována. Je zobrazena stránka se seznamem strategií. Název, popis a parametry editované strategie odpovídají provedeným změnám.
	16.	Smaž strategii kliknutím na "Delete strategy"		Vyskočí okno pro potvrzení smazání.
	17.	Potvrď smazání strategie		Strategie byla úspěšně smazána. Nenachází se na seznamu strategií ani ve vykresleném grafu.
uživatelé	18.	Přidej do projektu nového uživatele kliknutím na "Manage involved users" v levém menu		Stránka se seznamem uživatelů zapojených do projektu je zobrazena
	19.	Vyber uživatele a klikni na "Add person"		Uživatel přidán do projektu. Je zobrazen v tabulce uživatelů zpaojených do projektu.
	20.	Odeber uživatele z projektu kliknutím na ikonu odebrání		Vyskočí okno pro potvrzení smazání.
	21.	Potvrď odebrání uživatele		Uživatel byl odebrán. Není uveden v tabulce uživatelů zpaojených do projektu.
projekt	22.	Vrať se na detail projektu a smaž projekt kliknutím na "Delete project"		Vyskočí okno pro potvrzení smazání.
	23.	Potvrď smazání projektu		Projekt byl úspěšně smazán. Je zobrazena stránka se seznamem projektů- smazaný projekt není na seznamu.

Tabulka C.3: Testovací případ pro správu projektu

	Krok	Akce	Vstupní data	Očekávaný výsledek
	1.	Přihlas se s běžným uživatelem (bez administerských nebo manažerských práv), který je zapojen minimálně do jednoho projektu.	e-mail: risk.management.test@gmail.com heslo: Heslo_123	Uživatel je přihlášen. Je ztobrazena stránka se seznamem projektů, do kterých je zapojen.
	2.	Jdi do detailu projektu rozkliknutím panelu s projektem a kliknutím na "Open project"		Zobrazí se stránka s detailem projektu. Na stránce jsou vykresleny grafy ke všem rizikům projektu a je zobrazena tabulka se všemi riziky daného projektu.
matice a grafy	3.	Zobraz matici pravděpodobností a dopadů kliknutím na "P-I Matrix"		Stránka s maticí pravděpodobností a dopadů je zobrazena. Počet rizik v matici odpovídá počtům rizik v tabulce rizik.
	4.	Zobraz seznam rizik pod kategorií v P-I matici kliknutím na libovolnou buňku s číslem větším jak 0.		Zobrazí se modální okno se seznamem rizik spadajících pod danou kategorii.
	5.	Zobra graf pravděpodobností a dopadů kliknutím na "P-I Graph"		Stránka s grafem pravděpodobností a dopadů je zobrazena. Při najetí myší na body v grafu je zobrazen počet rizik pomocí jejich ID.
	6.	V tabulce rizik vyfiltruj rizika se zvolenou pravděpodobností a dopadem a ověř jejich počet v grafu.	Pravděpodobnost: 2 Dopad: 3 (v případě 0 rizik zvol libovolné jiné hodnoty)	Počet vyfiltrovaných rizik odpovídá počtu ID zobrazených na bodu v grafu umístěného na souřadnici zvolených hodnot pravděpodobnosti a dopadu.
riziko	7.	Klikni na "Add new" risk pro přídání nového rizika do projektu		Zobrazí se stránka s formulářem pro vytvoření nového rizika.
	8.	Vyplň všechny požadované pole ve formuláři, nastav pravděpodobnost, dopad a závažnost a potvrď	Libovolný text a hodnoty	Riziko bylo úspěšně vytvořeno. Je zobrazena stránka s detailem projektu.
	9.	Zkotroluj vytvořené riziko. V tabulce rizik použij funkci Search a napiš název rizika vytvořeného v předchozím kroku.		Vytvořené riziko je uvedeno v seznamu rizik. Název a parametry rizika odpovídají vyplněným hodnotám v předchozím kroku.
export do souboru	10.	Exportuj rizika do XLSX souboru kliknutím na ikonu označující Excel a zkontroluj soubor		XLSX soubor stažen. V souboru se nachází seznam rizik shodný s tabulkou "Register of risks"
	11.	Exportuj rizika do PDF souboru kliknutím na ikonu označující PDF		PDF soubor stažen. V souboru se nachází seznam rizik shodný s tabulkou "Register of risks"
	12.	Exportuj rizika do CSV souboru kliknutím na ikonu označující CSV		CSV soubor stažen. V souboru se nachází seznam rizik shodný s tabulkou "Register of risks"
strategie	13.	Klini na "Show strategies" pro zobrazení strategií.		Je zobrazena stránka se seznamen strategií a grafem vykreslujícím parametry jednotlivých strategií.
	14.	Klikni na "Add new strategy" pro vytvoření nové strategie		Zobrazí se stránka s formulářem pro vytvoření nové strategie.
	15.	Vyplň jméno a popis strategie a všechny další požadované parametry a potvrď	Libovolný text a hodnoty	Strategie byla úspěšně vytvořena. Je zobrazena stránka se seznamen strategií. Vytvořená strategie je uvedena v seznamu se správným názvem a popisem. Na paprskovém grafu jsou vykresleny hodnoty odpovídající zadaným parametrům při vytváření strategie.
	16.	Jdi zpět do detailu projektu.		Je zobrazena stránka s detailem projektu.
kontrolní seznam	17.	Klikni na "Complete the control checklist" pro zobrazení kontrolního seznamu		Stránka s kontrolním seznamem je zobrazena. Již provedené kontroly jsou označeny zaškrtnutým políčkem.
	18.	Zaškrtni/odškrtni libovolné položky a potvrď kliknutím na Submit		Kontrolní seznam byl uložen. Je zobrazena stránka s detailem projektu.
	19.	Klikni znovu na "Complete the control checklist" pro zobrazení kontrolního seznamu a zkontroluj zašrtnuté položky z předchozího kroku		Stránka s kontrolním seznamem je zobrazena. Všechny položky uložené v předchozím kroku jsou správně označeny.
dotazník	20.	Klikni na "Fill in the questionnaire" pro zobrazení dotazníku		Stránka s dotazníkem je zobrazena. Dříve vyplněné odpovědi jsou zobrazeny.
	21.	Vyplň pole dotazníku a potvrď kliknutím na Submit	Libovolný text	Kontrolní seznam byl uložen. Je zobrazena stránka s detailem projektu.
	22.	Klikni znovu na "Fill in the questionnaire" pro zobrazení dotazníku a zkontroluj vyplněné odpovědi z předchozího kroku		Stránka s dotazníkem je zobrazena. Všechny odpovědi uložené v předchozím kroku jsou správně vyplněny.

Tabulka C.4: Testovací případ pro uživatelské akce

Příloha D

Manuál

Prerekvizity

- stažený projekt,
- PHP 7.1.3 – https://windows.php.net/download/,
- Composer – https://getcomposer.org/download/,
- Vagrant – https://www.vagrantup.com/downloads.html,
- VirtualBox – https://www.virtualbox.org/wiki/Downloads.

Návod k přípravě Homestead prostředí

1. Ujistěte se, že je PHP správně nastaveno v systémové proměnné PATH.
2. V kořenové složce projektu spusťte v CMD příkaz: `composer install`.
3. Spusťte v CMD příkaz: `vagrant plugin install vagrant-hostsupdater`.
4. V kořenové složce projektu spusťte v CMD příkaz: `vagrant up`.
5. Ve vašem prohlížeči jděte na URL `http://risk.management`.

Předpřipravené testovací účty:

- admin@riskmanagement.com/admin123
- manager@riskmanagement.com/heslo123
- user@riskmanagement.com/heslo123

Příloha E

Obsah přiloženého CD

- složka `Aplikace` obsahující zdrojové kódy aplikace,
- technická zpráva,
- testovací případy,
- zdrojové kódy technické zprávy ve složce `TZ`.