FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Webová aplikace na podporu práce s úkoly malého týmu
Student:	Bc. Pavel Švagr
Vedoucí:	Ing. Martin Půlpitel
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce zimního semestru 2021/22

Pokyny pro vypracování

Zpracujte analýzu zadávaní úkolů v malé až středně velké firmě agenturního typu (softwarová firma, design studio). Na základě analýzy navrhněte a implementujte webovou aplikaci pro práci s úkoly, která bude komunikovat s tiketovacím open source systémem Redmine přes jeho API.
V práci se zaměřte zejména na:
- analýzu procesů zadávání úkolů
- stanovování priorit úkolů
- efektivitu vytěžování zaměstnanců
- analýzu problémů účastníků procesu
- optimalizaci procesu
- podporu práce s úkoly, které jsou zadávány přes Redmine
- aplikaci otestujte v reálném provozu

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 28. února 2020

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Webová aplikace na podporu práce s úkoly malého týmu

Bc. Pavel Švagr

Katedra softwarového inženýrství
Vedoucí práce: Ing. Martin Půlpitel

6. ledna 2021

Poděkování

Chtěl bych poděkovat své rodině, přítelkyni a přátelům za podporu při studiu na vysoké škole a tvorbě této práce. Mé díky také patří všem, kteří se podíleli na informacích, které sloužili jako podklad pro diplomovou práci – všem respondetům v rozhovorech a účastníkům v uživatelském testování. Děkuji také Ing. Martinu Půlpitlovi za domluvení firem pro rozhovory, poskytnutí Redmine instance firmy Ackee pro testování a konzultace ohledně finální implementované aplikace.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principu při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisu. V souladu s ust. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisu, tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programu, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli zpusobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelum). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu) licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným zpusobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 6. ledna 2021 . . . . . .

Odkaz na tuto práci

Švagr, Pavel. *Webová aplikace na podporu práce s úkoly malého týmu.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2021.

Abstrakt

Diplomová práce se zabývá procesem zadávání úkolů v malém týmu. Cílem je navrhnout, implementovat a otestovat moderní webovou aplikaci pro podporu zadávání úloh napojenou na systém Redmine, která bude reflektovat aktuální potřeby firem zaměřených na tvorbu softwarového či grafického produktu. Za tímto účelem je kromě analýzy procesu na teoretické úrovni proveden průzkum problémů a slabin při zadávání úkolů ve vybraných podnicích doplněn o zhodnocení již existujících podpůrných nástrojů.

Klíčová slova úkoly, zadávání úkolů, správa úkolů, efektivita, webová aplikace, Redmine, JavaScript

Abstract

The diploma thesis deals with the process of task management in a small team. The aim is to design, implement, and test modern web application supporting task assignment procedure which is connected to the Redmine system and reflects the current needs of companies focused on creating software or graphic products. In addition to the analysis of the process at the theoretical level, a survey of difficulties and shortcomings of task assignment in selected companies is carried out along with an evaluation of existing support tools.

Keywords tasks, task assignment, task management, efficiency, web application, Redmine, JavaScript

Obsah

Úvod **1**
Motivace . . . 1
Cíle práce . . . 2

1 Analýza procesu zadávání úloh **3**
1.1 Distribuce úkolů a určování závislostí . . . 3
1.2 Tvorba zadání a dokumentace . . . 4
1.3 Odhady termínů a nacenění . . . 5
1.4 Určování priorit . . . 5
1.4.1 MoSCoW metoda . . . 6
1.4.2 Eisenhowerova metoda . . . 6
1.4.3 ABCDE metoda . . . 7
1.4.4 Pravidlo 1-3-5 . . . 7
1.4.5 Stodolarová metoda . . . 8
1.4.6 Kano model . . . 8
1.4.6.1 Backlog . . . 10
1.5 Přiřazení řešitele . . . 10
1.6 Komunikace . . . 10
1.7 Stavy úkolů . . . 11
1.8 Ovlivňující faktory . . . 12
1.8.1 Projektové řízení . . . 12
1.8.1.1 Waterfall . . . 13
1.8.1.2 Rational Unified Process . . . 13
1.8.1.3 Scrum . . . 14
1.8.1.4 Extrémní programování . . . 14
1.8.2 Velikost týmů a počet zaměstnanců . . . 15
1.8.3 Volba podpůrných systémů . . . 15

2 Analýza zadávání úloh ve vybraných firmách **17**

2.1 Vybrané firmy . . . 17
2.2 Témata a způsob dotazování . . . 18
2.3 Velikosti firem a rozložení týmů . . . 19
2.4 Projektové řízení firem . . . 19
2.5 Zadávání a správa úkolů . . . 20
2.5.1 Distribuce a delegace úloh . . . 20
2.5.2 Určování priorit úkolů . . . 20
2.6 Role zaměstnanců zapojených do správy úkolů . . . 21
2.7 Používané podpůrné systémy a nástroje . . . 22
2.7.1 Chybějící funkce . . . 22
2.7.2 Špatně zvolené systémy . . . 23
2.8 Vyhodnocení průzkumu . . . 23
2.8.1 Problémové oblasti . . . 23
2.8.2 Procesy založené na schůzkách . . . 24

3 Analýza podpůrných systémů 25
3.1 Sledovaná kritéria . . . 25
3.2 Trello . . . 26
3.2.1 Funkce pro správu úkolů . . . 27
3.2.2 Uživatelské rozhraní a přístupnost . . . 28
3.2.3 Shrnutí . . . 28
3.3 Asana . . . 29
3.3.1 Zobrazení a definice úkolů . . . 29
3.3.2 Sledování stavu úkolů . . . 30
3.3.3 Podpůrné funkce projektového řízení . . . 30
3.3.4 Uživatelské rozhraní . . . 30
3.3.5 Shrnutí . . . 30
3.4 Notion . . . 31
3.4.1 Nástroje pro správu úkolů . . . 31
3.4.2 UI, přístupnost a možnosti napojení . . . 31
3.4.3 Shrnutí . . . 32
3.5 Redmine . . . 32
3.5.1 Funkce pro správu úkolů . . . 32
3.5.2 Atributy a priority úkolů . . . 33
3.5.3 Sledování stavu projektu . . . 33
3.5.4 Napojení na externí systémy . . . 33
3.5.5 Uživatelské rozhraní a přístupnost . . . 34
3.5.6 Shrnutí . . . 34
3.6 Jira . . . 34
3.6.1 Konkurenční výhody . . . 35
3.6.2 Uživatelské rozhraní . . . 35
3.6.3 Shrnutí . . . 36
3.7 Azure DevOps . . . 37
3.7.1 Podporované funkce . . . 37

3.7.2 UI, přístupnost a napojení na další systémy 37
3.7.3 Shrnutí 37
3.8 Vyhodnocení 38

4 Návrh podpůrné aplikace **39**
4.1 Cíl projektu 39
4.2 Navržené řešení 40
4.2.1 Úprava Ackee tabulky 40
4.2.2 Podpůrné nástroje 40
4.2.3 Využití aplikace 41
4.3 Požadavky na aplikaci 41
4.3.1 Nefunkční požadavky 41
4.3.2 Funkční požadavky 42
4.4 Případy užití 43
4.4.1 Uživatel 43
4.4.2 Moderátor 43
4.4.3 Administrátor 44
4.4.4 Aktéři pro tabulky 45
4.5 Návrh uživatelského rozhraní 45
4.5.1 Navigace 45
4.5.2 Prvky pro výčet entit 46
4.5.3 Zobrazení tabule 46
4.5.4 Zobrazování dodatečných informací 47
4.5.5 Nastavení tabule 48

5 Implementace aplikace **49**
5.1 Vybrané technologie 49
5.1.1 Klientská část 50
5.1.2 Serverová část 51
5.1.3 Sdílené technologie 52
5.1.4 Externí nástroje 52
5.2 Databázový model 53
5.3 Komunikace 54
5.3.1 REST API 55
5.3.2 Upozornění pomocí SSE a SendGrid 56
5.3.3 Nahrání uživatelů z Redmine 58
5.4 Zabezpečení a přihlášení 58
5.5 Struktura aplikace 59

6 Nasazení a testování **61**
6.1 Testování v aplikaci 61
6.2 Nasazení na Heroku 62
6.3 Testování v provozu 62
6.3.1 Prezentace 62

6.3.2 Uživatelské testování 63
6.3.2.1 Scénáře pro testování 63
6.3.2.2 Výsledky a úpravy aplikace 64
6.3.2.3 Vyhodnocení 66
6.4 Možná budoucí rozšíření 66

Závěr **67**

Literatura **69**

A Seznam použitých zkratek **79**

B Instalační příručka **81**
B.1 Demo aplikace 81
B.2 Lokální spuštění 81
B.2.1 Založení účtů v externích aplikacích (Prosinec 2020) . . 81
B.2.1.1 MongoDB 82
B.2.1.2 Google Identity 82
B.2.1.3 Amazon S3 83
B.2.1.4 SendGrid (volitelné) 84
B.2.2 Konfigurace a spuštění aplikace 84
B.2.3 Testovací data 85
B.2.4 Spuštění testů 85
B.2.5 Vyprázdnění databáze 85

C Souhlasy respondentů s pořízením rozhovoru **87**

D Přepsané rozhovory z firem **93**
D.1 Ackee (Ing. Martin Půlpitel) 93
D.2 Qest (Ing. Martin Koperniech) 97
D.3 Digital Ant (Ing. Štěpán Heller a Lukáš Obdržálek) 101
D.4 2FRESH (Lenka Kůrková) 106

E Wireframy pro návrh uživatelského rozhraní **113**

Seznam obrázků

1.1 Ukázka Eisenhowerovy matice 7
1.2 Graf Kano modelu 9
1.3 Ukázka Kanbanovské tabulky 12

2.1 Podpůrná tabulka pro priority používaná ve firmě Ackee 21

3.1 Ukázka Kanbanovské tabulky v aplikaci Trello 27
3.2 Ukázka problému řazení v časové řadě v systému Asana. 29
3.3 Ukázka použití backlogu v aplikaci Jira 36

4.1 Diagram případů užití 44
4.2 Navržené logo pro aplikaci TaskBoard. 45
4.3 Wireframy pro seznam tabulí 46
4.4 Wireframy pro editorský detail tabule. 47
4.5 Wireframe pro nastavení tabule. 48

5.1 Finální schéma databázového modelu 54
5.2 Ukázka zaslaného emailového upozornění aplikace TaskBoard. . . . 57

Úvod

Projektové řízení je nedílnou součástí fungování nejen vývojářských firem. Klient v dnešní době neplatí pouze za kreativní a technickou práci odvedenou specialisty, ale také za manažerskou činnost na pozadí. Právě cena často rozhoduje o tom, jakou firmu si klient pro zpracování svého projektu zvolí. Proto je cílem firem projektové řízení zefektivnit a náklady s ním spojené minimalizovat. Zadávání úkolů je proces při řízení projektu, který probíhá v každé firmě nehledě na její velikost nebo zaměření a je s ním spojená velká část pracovní náplně projektového manažera.

Motivace

Během procesu zadávání úloh může docházet k řadě problémů jako jsou opožděná delegace, nedostatečná komunikace či špatné určení priorit a závislostí. Následkem může být zdržení dodávky produktu, prodražení projektu, nesplnění klientského očekávání, ale i špatná pracovní morálka zaměstnanců v týmu. Efektivita a přehled v zadávání úloh jsou proto důležitými faktory pro zkvalitnění projektového řízení firmy.

V dnešní době je sepsáno mnoho článků založených na zkušenostech v určitých podnicích. Problémem pro firmy ovšem bývá jejich reálná aplikace v praxi. Každý klient má své specifické potřeby a projekty mohou mít různý tvůrčí proces. Existuje také mnoho podpůrných systémů s nástroji, které se snaží tento proces zjednodušit a urychlit. Často jsou ale placené a v ceně bývají sady funkcí, které podnik nevyžaduje nebo v nich naopak potřebné funkce chybí.

Cíle práce

Cílem rešeršní části práce je nastudovat proces zadávání úloh v různě projektově řízených týmech. Dále na základě průzkumu z firem zaměřených na tvorbu grafického či softwarového produktu identifikovat společné problémy a chyby, které při tomto procesu vznikají. Následně je nutné prozkoumat existující nástroje pro podporu zadávání úkolů a způsoby, kterými se jednotlivé problémy snaží vyřešit.

V praktické části práce je cílem navrhnout řešení, které by mohlo zamezit tvorbě nejčastěji nalezených problémů. Toto řešení by mělo být implementováno jako webová aplikace s napojením na existující tiketovací systém Redmine. Na závěr je potřeba aplikaci otestovat, vyhodnotit přínosy vytvořeného řešení a možná rozšíření do budoucna.

Kapitola 1

Analýza procesu zadávání úloh

Procesem zadávání úloh se myslí kompletní přenos informací ohledně zpracování určitého požadavku od zadavatele k řešiteli. Výsledkem je úloha (nazýváno také jako „tiket“), kterou má jedinec či určitá jednotka v rámci organizace vykonat. Výkonné akce, které úkol vyžaduje, jsou řízeny znalostmi jednotlivých členů organizace a spějí k určitému společnému cíli. Ten je dosažitelný často mnoha způsoby. Způsob řešení je tak vybírán na základě dostupných zdrojů a okolností za jakých zadání vzniká. [1]

Na zadávání úkolů ve většině firem dnes slouží podpůrné systémy. Při jejich špatném využití může dojít k mnoha problémům bránícím výkonu akce. Důsledkem je neefektivní plnění úkolů, které vede ke ztrátě investovaných zdrojů [2]. Obecně je problém zadávání a správy úkolů označován jako „task management“ (dále jen TM). Požadavky na výkonný tým lze rozdělit na dva typy, které mívají mírně odlišné postupy zadávání a správy:

- tvůrčí úkol (dále v textu nazýváno jako úkol typu „feature“),
- opravení nedostatku (v IT známé jako „bug“, dále v testu pouze „issue“).

Při TM dochází k mnoha úkonům, které musí řešit jak zadavatel, tak výkonný pracovník. V této kapitole shrnu nejčastější postupy a principy používané při správě úkolů a uvedu podpůrné nástroje či praktiky, které slouží k zvýšení efektivnosti procesu.

1.1 Distribuce úkolů a určování závislostí

Práce v týmu vyžaduje rozdělení úkolů mezi jednotlivé výkonné členy. To je i základním postupem při plánování projektu. Prvotním krokem je stanovení základního seznamu úloh, jejichž splněním vznikne finální produkt. Tento seznam se nazývá „work breakdown structure“ (dále jen WBS). [3]

Základní seznam úloh může být dál dělen. Obecně uznávanou praktikou při tvorbě WBS je rozdělovat úkoly na menší celky, čímž se ze seznamu stává strom, kde listy jsou nejmenší dílčí úlohy a kořenem stromu je finální projekt. Dělení úkolů můžeme rozlišit na dva následující typy. [4, 5]

Dělení založené na objektu (object based) vytváří úkoly podle jednotlivých částí produktu, které je možné vytvářet nezávisle. Pro produkt webové stránky tak vznikají například úkoly pro tvorbu jednotlivých prvků na stránce (navigace, tlačítka, formuláře, ...).

Dělení založené na aktivitě (activity based) je zaměřeno na vytváření úkolů podle činností, které nejsou konfliktní (tvorba designu, implementace backend části, ...). Tento typ dělení je typickou součástí standardizace projektového řízení dle návodu „Project Management Body of Knowledge“, známém pod zkratkou PMBOK [6].

Každý tým či typ projektu může vyžadovat jiný styl dělení. V praxi je častá i kombinace obou typů, vždy by ale výsledné úkoly měly být:

- měřitelné v pokroku ke splnění,
- dobře časově odhadnutelné,
- přiřazené jedné zodpovědné výkonné jednotce,
- založené na dodání určitého výsledku a
- splnitelné v rozumném čase. [7]

Důležitou součástí distribuce úkolů je určení závislostí mezi nimi. Některé požadavky mohou být vzájemně ovlivňující. Jeden úkol tak není možné splnit, dokud není dokončena část jiného úkolu, na kterém závisí. Důsledkem je prodloužení tvorby, což může vést i k nestihnutí termínu dodání. [8]

Rozdělení úkolů je tak nejčastěji děláno na základě zkušeností s předchozími projekty nebo expertizy v dané činnosti. V některých případech jsou vytvořeny šablony procesu, které určují rozdělení úloh. Jako další pomocné nástroje se používají například seznamy členů týmů, tabulka zodpovědností a grafy zobrazující časový plán a tok projektu. Typickým zástupcem těchto vizuálních nástrojů je pak Ganttův diagram nebo Roadmapa (plán projektu). [9, 10]

1.2 Tvorba zadání a dokumentace

Při tvorbě úkolu je důležité výkonné jednotce předat přesné informace o tom, co má být cílem její činnosti. Nutnost obsažnosti zadání je závislá na typu projektu. Obzvlášť významné jsou popisy u softwarových projektů. Typickým příkladem jsou úkoly typu issue. Při realizaci takového požadavku je

často efektivní opakování postupu vedoucího k chybě v ladícím režimu aplikace. Programátor tak může snadno nalézt část kódu, která se chybu způsobuje. Je proto vhodné v těchto případech podat přesný popis, jak byla chyba nalezena. [11]

Při softwarovém vývoji se využívají tiketovací systémy, které v sobě nesou informace o úkolech. Pro výkonné pracovníky i manažery je pak častou aktivitou hledání určitého úkolu či projektu. Může tak hrát velkou roli i označení či název úkolu. Zavádějí se proto například firemní pravidla pro pojmenovávání jednotlivých požadavků. [12]

Podpůrným nástrojem mohou být šablony struktury ukládaného úkolu, které často umožňují tiketovací systémy vytvořit. Zadavatelé jsou pak povinni vyplnit informace, které mohou být v dalších procesech nebo budoucímu reportování užitečné. Dalšími nástroji mohou být například přiložené soubory s diagramy, obrázky či textem. Pro tento účel mohou být využity také dokumentové systémy, kde je průběžně udržováno aktuální zadání projektu a jeho dílčích částí.

1.3 Odhady termínů a nacenění

Stanovování termínů dokončení je z jednou z největších výzev projektového manažera. Při špatném odhadu náročnosti je buďto úkol nadhodnocen, čímž může docházet k nevytíženosti týmu, nebo je podhodnocen a projekt se může ocitnout v časové tísni. Při nestíhání termínu dodání pak musí přijít řešení, která nemusí být přijatelná jako např. prodloužení termínu, práce přesčas, získání externí pomoci, nebo zúžení zaměření projektu. [13]

Pro určování termínů jsou důležité i předešlé zkušenosti. Používány jsou proto ve firmách různé nástroje na sledování stráveného času na úkolech. Odhady často probíhají při plánování projektu a bývají stanoveny na základě seznamu úkolů, jejich závislostí a priorit. Používají se proto stejné nástroje jako u distribuce úkolů, tedy např. WBS, či Ganttův diagram. [14]

1.4 Určování priorit

Přiřazení priorit k úkolům je další z řady typických výzev při procesu správy úloh. Priority vyjadřují míru důležitosti požadavků v daný moment a určují, který úkol by měl být vypracován nejdříve. Pro vykonavatele je v případě více zadaných úkolů důležité vědět, který z nich má vykonat první, aby byl dodržen termín dodání, nebo aby se do příštího vydávaného prototypu produktu dostaly správné funkce a části.

Proces určování priorit často probíhá v iteracích a je prováděn kontinuálně s tvorbou projektu. Typicky ke změně priorit dochází u agilních metodik řízení projektu, kde se během času může měnit důležitost požadavků na základě komunikace s klientem. [15]

Určení priorit je také silně spjato se závislostmi úloh. Pokud existuje úkol, který blokuje vykonávání jiného, může dojít k nedokončení závislého úkolu včas. Těmto požadavkům se obecně říká „blokující úkoly“ a mívají proto vysokou prioritu.

Konvenční nástroje často umožňují různé způsoby, jak priority přiřazovat. Obecně rozšířenou metodou je označování úkolů různými štítky, které definují stupeň priority, případně seřazení do seznamu od nejdůležitějšího po nejméně důležitý. V podpůrných systémech si lze často štítky a druhy priorit definovat. Důležité tak je proto zvolit správnou techniku pro stanovení priorit, na základě které pak bude i systém přizpůsoben.

1.4.1 MoSCoW metoda

Metoda MoSCoW je založena na ujasnění základních potřeb projektu v daný okamžik. Využívá se, pokud lze pro tým přesně definovat úkoly a není pro ně primárně důležitý termín dodání. Typicky se metoda aplikuje při řazení úloh v agilních metodikách. Rozdělení priorit je podle následujících jednoduchých kritérií. [16]

M Co musí projekt obsahovat (must have).

S Co by měl projekt obsahovat (shoud have).

C Co by projekt mohl obsahovat (could have).

W Co by projekt mohl mít, ale není to aktuálně potřeba (won't have this time).

1.4.2 Eisenhowerova metoda

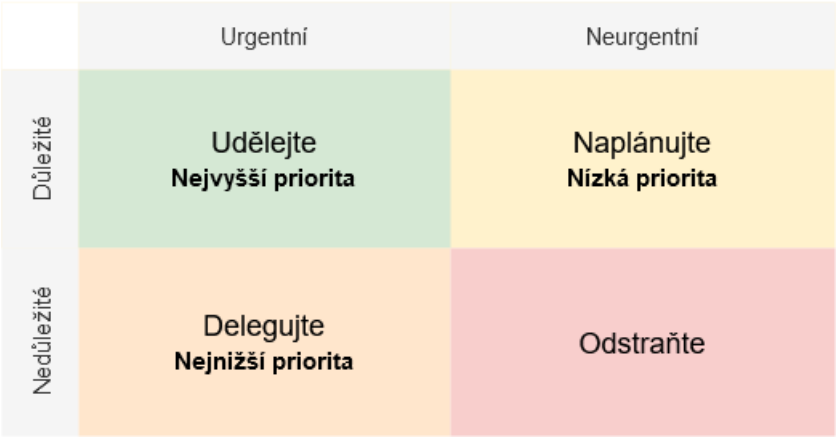
Další základní metodou určení priorit je Eisenhowerova metoda (nebo také Eisenhowerova matice). Její vizualizace je ukázána na obrázku 1.1. Jedná se o obecnou praktiku pro rozdělení priorit nejen týmových úkolů. Metoda bere v úvahu dvě měřítka: důležitost a urgentnost úkolu. [17]

Důležité úkoly a urgentní (DU) určují nejvyšší prioritu. Tyto úkoly by měly být okamžitě vykonány.

Důležité a neurgentní úkoly (DNU) mají druhou nejvyšší prioritu a mělo by být naplánováno, kdy se budou vykonávat.

Urgentní, ale nedůležité (NDU) úkoly nesou nejmenší prioritu a měly být dál delegovány na jednotku, která je může udělat za nás.

Nedůležité a neurgentní (NDNU) úkoly by neměly nést žádnou prioritu, protože pro projekt nejsou důležité a měly by být odstraněny.



Obrázek 1.1: Ukázka Eisenhowerovy matice nakreslena dle článku [17].

1.4.3 ABCDE metoda

Podobnou metodou Eisenhowerově je metoda ABCDE. Principem je přiřazování písmen anglické abecedy od A do E dle priority úkolu. Oproti Eisenhowerově matici třídí NDU úkoly dále na dvě další priority B a C dle jejich urgentnosti. Jednotlivá písmena odpovídají následujícím prioritám. [18]

A – Nejvyšší priorita (odpovídá DU).

B – Střední priorita.

C – Nízká priorita.

D – Delegace úkolu (odpovídá NDU).

E – Eliminace úkolu (odpovídá NDNU).

1.4.4 Pravidlo 1-3-5

Metodou pro krátkodobé určení priorit je takzvané pravidlo 1-3-5. Uvažuje časové možnosti výkonného pracovníka, urgentnost úkolů a optimální rozpoložení zátěže. Jejím cílem je mít přesně určené, co je pro zaměstnance v daný časový úsek (nejčastěji jeden den) to nejdůležitější. Pravidlo říká, že každý zaměstnanec by měl mít jeden úkol s největší, tři úkoly se střední a pět úkolů s nízkou prioritou, kterým by se daný časový úsek měl věnovat. [19]

Logikou skrytou za tímto rozložením je princip, který představil v knize „Eat the Frog“ Bryan Tracy. Jeho doporučení, jak se vyhnout negativním efektům jako prokrastinaci či upřednostňováním menších nedůležitých úkolů, je dělat vždy nejdříve ty nejdůležitější a největší úkoly. [20].

Toto pravidlo je spíše obecným principem. Může sloužit k udržování informovanosti a přehledu o aktuálních aktivitách v týmu nebo v případě nemožnosti definovat dostatek úkolů také indikátorem nevytíženosti určité výkonné jednotky. Metoda je upravitelná v závislosti na průměrném množství požadavků, které řešitelé mají, například snížením počtu jednotlivých úloh na dané prioritě.

1.4.5 Stodolarová metoda

Stodolarová metoda[1] je opět založena na urgentnosti a důležitosti jednotlivých úkolů, kterým je přiřazována bodová hodnota (dolary). Na začátku je určení bodového přídělu pro danou skupinu úkolů (v základní verzi metody se jedná o 100 bodů). Z různých pohledů jako například důležitost pro cílovou skupinu, nákladnost, či termín dodání, jsou přiřazovány body. Čím významnější pro dané kritérium úkol je, tím více bodů by měl dostat. Součet všech bodů se na konci vždy musí rovnat původnímu bodovému přídělu. Úkoly jsou pak seřazeny podle získaných bodů sestupně. Čím výše je úkol umístěn, tím větší má prioritu. [21]

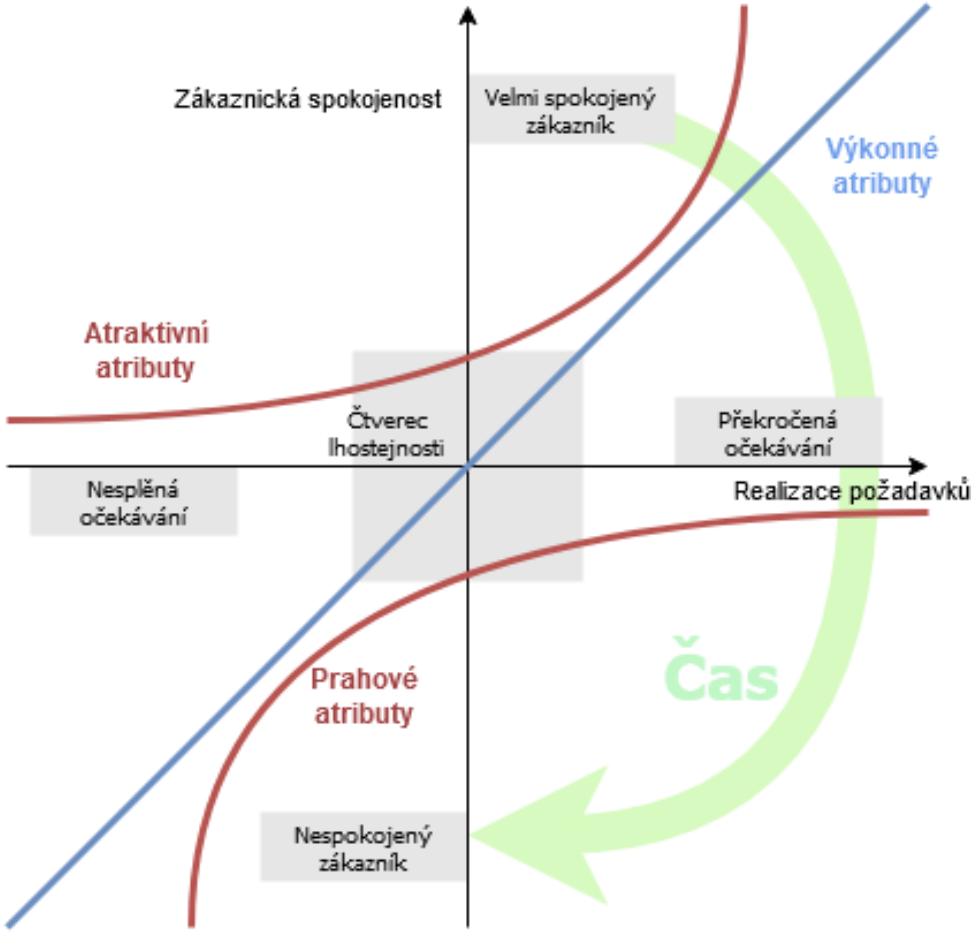
1.4.6 Kano model

Pomocnou metodou pro plánování projektu a určování priorit je i Kano model. Jedná se o graf ukazující spokojenost klienta v závislosti na vývoji produktu a je možné ho vidět na obrázku 1.2. Na ose x je vyjádřena míra naplnění požadavků zákazníka a na ose y je ukázána míra jeho spokojenosti. Kano model dále rozlišuje tři typy atributů projektu (a tedy i úkolů), které mají určitý vliv na zákazníkovu spokojenost. [22]

Prahové atributy (threshold attributes) nebo také „základy“ jsou požadavky, které výsledný produkt obsahovat. V případě, že tyto požadavky nejsou splněné, dochází k silné nespokojenosti zákazníka. Úkoly spojené s prahovými atributy by tedy měly být nejvíce prioritní. Pro objednávkový systém to může být například možnost objednat si produkt.

Výkonné atributy (performance attributes) nebo také „spokojenosti“ jsou požadavky, které jsou známé a mají v přímé úměře vliv na spokojenost zákazníka. Pokud dané atributy výsledný produkt má, zákazník je o to spokojenější. Pokud je nemá, zákazník bude nespokojený. Pro objednávkový systém může být takovým požadavkem rychlost objednávky.

[1] Také známá jako „Stobodová metoda“.



Obrázek 1.2: Graf Kano modelu překreslen na základě interaktivního obrázku v článku na mitroTOOL.de [23].

Atraktivní atributy (excitment attributes) nebo také „radosti“ jsou takové požadavky, které pro produkt nutné nejsou, ale jejich existence zvyšuje míru spokojenosti zákazníka. V praxi se může jednat o určité atributy, které klient chce, ale nepovažuje je za důležité či nejsou součástí původního zadání, proto je jejich vytvoření nejméně prioritní.

V některých případech se postupem času z atraktivních atributů se stávají výkonné a z výkonných zase prahové. Typickým případem je právě iterativní vývoj softwarových produktů v agilních metodikách, kde klient přichází stále s novými požadavky. Tento jev v grafu vyznačuje kruhová šipka s označením času. Pokud jsou atributy dostatečně naplněny, zákazník začíná být přiměřeně spokojený. Tuto situaci vyjadřuje čtverec lhostejnosti (indifference zone) umístěný ve středu grafu. [24]

1.4.6.1 Backlog

Typickým nástrojem pro přehled důležitých úkolů při agilním vývoji v moderních podpůrných systémech bývá řazení požadavků do seznamu nazývaného backlog. Z backlogu výkonní pracovníci, v případě že je předchozí práce hotová, postupně odebírají úkoly. Vždy by si měl pracovník vzít nejvýše umístěný požadavek, který může aktuálně splnit, čímž dochází k plnění nejdůležitějších úkolů co nejdříve. [25]

V backlogu se často ale nachází také úkoly, které nejsou aktuálně zpracovávané. Na nejvyšších pozicích jsou jasně definované úkoly, které jsou okamžitě vykonávané. Čím níže se v seznamu posouváme, tím méně podrobněji jsou v něm úkoly definovány. Backlog tak vyžaduje neustálou správu a dodefinovávání. K určení priorit v backlogu pak mohou sloužit již zmíněné metody.

1.5 Přiřazení řešitele

Po vytvoření a rozdělení úkolů je důležité je přiřadit správnému řešiteli. Výběr je primárně postaven na schopnostech výkonných pracovníků, dále rozhoduje jejich vytíženost a časová náročnost úkolu. Zbylé rozhodování je v rukou projektového manažera. Špatný výběr může mít např. negativní sociální důsledky [26]. Jednou z možných strategií, jak rozdělovat úkoly, je také nechat pracovníky vybrat si, které úkoly chtějí dělat. Může tak být zvýšena morální spokojenost spojená s lepšími výsledky zaměstnanců [27].

Podpůrné aplikace pro TM nabízí určitou podporu pro přiřazování řešitelů. Většinou tak u tiketovacích systémů dochází pomocí změny řešitele přiřazením do seznamu úloh daného uživatele. Po delegaci úkolu pak běžně následuje upozornění prostřednictvím nějakého komunikačního kanálu např. email.

1.6 Komunikace

Pravidelné předávání informací o plnění a stavu úkolů je běžnou činností při kolektivní tvorbě produktu. Obzvlášť v agilně řízených týmech je komunikace z důvodu změn požadavků od klienta a doručování software v iteracích klíčovým faktorem [28]. Pokud je informace o požadavcích a jejich zadávání předávána mezi mnoha stranami, může dojít k negativnímu „efektu tiché pošty“, kdy je informace od zadavatele k řešiteli postupně každým člověkem deformována [29]. Komunikace lze rozdělit dle několika parametrů jako jsou:

- médium komunikace (z očí do očí, textová, ...),
- uchovatelnost (zpětně dohledatelná, nedohledatelná),
- komunikující strany (uvnitř jednoho týmů, napříč různými týmy, ...).

Pro každý druh projektu může být různý styl komunikace jinak efektivní. Obecně bývá z pohledu budoucí kontroly nutná uchovatelnost získaných informací. To může představovat problém v případě videohovoru, který lze nahrát, ale není možné v něm informace efektivně hledat. Problém může nastat i u komunikace z očí do očí, která nemusí být uchovávána vůbec. Proto se běžně používají různé zápisy či shrnutí ze schůzek.

Podpůrné systémy pro TM často poskytují uchovatelnou formu komunikace. Typicky umožňují přidávání komentářů u jednotlivých úkolů pro sledování stavu, čímž udržují zároveň veškeré informace o úkolu zapouzdřené na jednom místě. Dále jsou využívány obvyklé nástroje pro komunikaci jako chatovací aplikace, email či systémy umožňující videohovory.

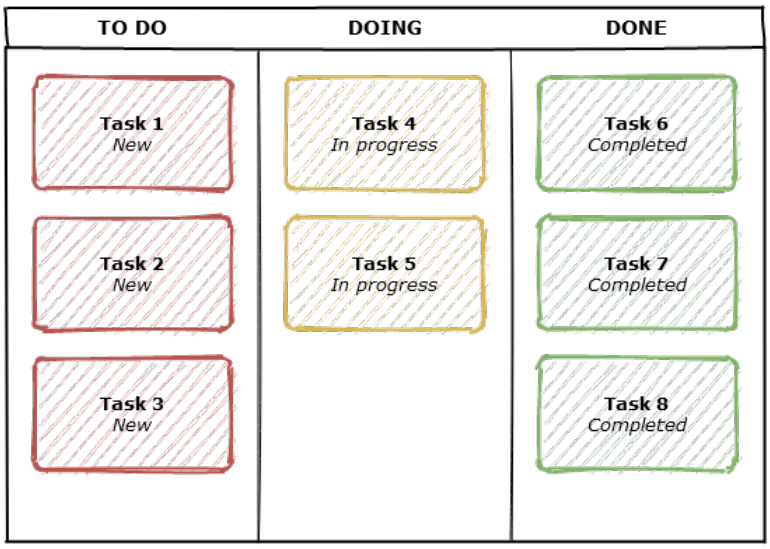
1.7 Stavy úkolů

Dalším atributem úkolů, který se při TM běžně definuje pro přehlednost pracovního toku, je jejich stav. Pro vyjádření se může použít orientovaný graf, kde stavy jsou množina vrcholů a hrana určuje možný přechod mezi nimi. Typicky jsou používána tři základní označení:

- k vyřešení (to do)
- v řešení (doing),
- vyřešený (done).

Dle typu výkonné činnosti se přidávají další stavy, do kterých se úkol může dostat. Například pro podniky dodávající softwarový produkt bývá časté zařazení stavů spojených s připraveností produktu jako jsou: „k otestování“, „nasazen na testovací server“, „nasazený na produkční server“.

Nástrojem běžně používaným pro definici stavů je Kanbanovská tabulka jejíž příklad je zobrazen na obrázku 1.3. Jedná se o zleva doprava seřazené sloupce pojmenované podle definovaných stavů. V každém sloupci se nachází seznam úkolů v tomto stavu. Úlohy se pak přesouvají dle pracovního toku zleva doprava. Tabulka je často využívána i ve fyzické podobě s tabulí a lístečky. Nalézt lze i specifická přizpůsobení pro různé metodiky projektového řízení. [30]



Obrázek 1.3: Načrtnutá ukázka Kanbanovské tabulky.

1.8 Ovlivňující faktory

Příčinou problémů může být jak lidský faktor, tak špatně nastavený proces TM. V obecných měřítkách se tak jedná o individuální problém každého výkonného týmu. Jakým způsobem jsou úkoly zadávány a spravovány ovlivňují kromě výkonné činnosti firmy také faktory zmíněné v následujících sekcích.

1.8.1 Projektové řízení

Styl projektového řízení má zásadní vliv na TM. Používané procesy a metodiky často definují, kdy dochází ke stanovení cílů projektu (a tedy i vzniku zadání úkolů), jak se mohou cíle projektu měnit, obsažnost projektové dokumentace, či označení stavů úkolů. V dnešní době společnosti z důvodu efektivity využívají zavedené standardy projektového řízení a metodiky. Metodiky můžeme rozdělit na tradiční a agilní. [31]

Tradiční metodiky bývají použity převážně pro velké zakázky, u kterých je jasně daný cíl a u nichž nedochází ke změně požadavků během tvorby. Nedochází k výrazným změnám cílů a plánů projektu. Proto je běžně nutné dopředu znát kompletní scénář případů užití a požadavky na funkčnost, což vyžaduje podrobnou dokumentaci. Mezi neznámější metodiky patří Waterfall či Rational Unified Process. [32]

Agilní metodiky jsou oblíbené u menších a středních podniků a u týmů využívající průběžnou dodávku produktu (contignous delivery). Přístup je méně striktní na určitost požadavků při zahájení projektu a zakládá si na iterativním vývoji a průběžném prototypování. Úkoly bývají zadávány na základě aktuálních klientských požadavků pro danou iteraci a jsou ovlivňovány předchozími zkušenostmi. Běžně se neklade důraz na rozsáhlou dokumentaci a plán projektu, jelikož k detailnějšímu upřesnění dochází v průběhu tvorby.

Typickým nástrojem pro dělení úkolů v agilním řízení jsou „epics“, neboli seznamy úkolů, které se dají dělit v menší vykonavatelné úlohy nazývané „user stories“. Mezi nejznámější metodiky agilního přístupu patří Scrum, Extrémní programování, Lean software development, Crystal, Feature driven development a Dynamic system development. [33]

1.8.1.1 Waterfall

Příkladem tradiční metodiky je Waterfall, neboli „Vodopádový model“. Je založen na lineární tvorbě produktu, jejíž fáze se rozdělují na:

- určení všech klientských požadavků,
- design a návrh řešení,
- tvorba produktu,
- schválení vytvořeného produktu klientem,
- údržba projektu. [34]

Kvůli nutnosti znát důkladně požadavky před samotnou tvorbou řešení dochází k složitému procesu plánování. Nejdříve je sestaven kompletní projektový plán, který se konzultuje s osobou zodpovědnou za projekt na straně klienta (stakeholder). Následně se přechází k výzkumné části. Na základě požadavků dojde k vytvoření tvůrčího týmu a jsou mu předány na schůzce, které se říká kick-off. Úkoly jsou tvořeny z původního plánu, který je neměnný. V dalším kroku tvorby se nepokračuje, pokud ten předchozí není kompletně dokončen. Úlohy tak bývají vzájemně blokující. [35]

1.8.1.2 Rational Unified Process

Další používanou tradiční metodikou je Rational Unified Process (dále jen RUP). Oproti Waterfallu funguje v iteracích, na jejichž konci jsou vydávány verze výsledného produktu. V rámci jednotlivých iterací je důležitá dokumentace zadání, ke které se používá Unified Modeling Language (UML). Na začátku je definovaný rozsah projektu (scope), který udává cíle. V rámci iterací může dojít k vyjasnění rozsahu, nikoli však ke změnám prvotního zadání.

Úkoly jsou tak zadávány na základě jednotlivých iterací s ohledem na projektové plánování v úvodu s možností doplnění dokumentace během jednotlivých iterací. Projektová dokumentace se vždy dává ke schválení klientovi. Běžnou praktikou při tvoření úkolů je práce s pracovním tokem (workflows). Pro zvýraznění závislostí jednotlivých úloh se používá sekvenční UML diagram. [36]

1.8.1.3 Scrum

Velmi rozsáhlou agilní metodikou je Scrum. Ta je orientovaná na dodávání produktu v krátkých iteracích, které nazýváme sprinty. Sprinty bývají dlouhé jeden týden až měsíc. Požadavky jsou ukládány v backlogu, ze kterého se vždy vybírají požadavky jen pro následující sprint. Scrum také rozděluje roli projektového manažera dle zodpovědností na dvě další role. [37]

- První je „product owner“, který je zodopovědný za úkoly v backlogu.
- Druhou je „scrum master“ zodpovědný za komunikaci a odstraňovat problémů v týmu.

Metodika také definuje několik následujících schůzek v týmu pro podporu vzájemné informovanosti o stavu a plnění úloh v rámci projektu. [38]

Stand-up je každodenní velmi krátká schůzka (15 min) kde členové týmu krátce vysvětlí jaké úkoly splnili předchozí den, jaké splní dnes a jaké mají při plnění překážky.

Plánování sprintu slouží k definování, co bude cílem sprintu následujícího vzhledem k časovým kapacitám týmu.

Hodnocení sprintu je informativní schůzka, kde se ukazuje demo vytvořených funkcionalit z proběhlého sprintu a probíhá jejich hodnocení.

Retrospektiva je schůzka výkonného týmu, kde se probírá, co nefungovalo v proběhlém sprintu a navrhují se možnosti vylepšení procesu.

Přehled o procesu zadání a plnění představuje Kanbanová tabulka této metodiky, v angličtině také nazývaná „Scrumboard“. Správně řízený tok úloh a sledování jejich plnění je tak i jednou z důležitých statistik metodiky. Pokud dochází ke splnění stanovených cílů pro daný sprint, upravuje se práce s požadavky při plánování následujících sprintů.

1.8.1.4 Extrémní programování

Další rozšířenou agilní metodikou obzvlášť v softwarových projektech je extrémní programování (dále jen XP). XP je založeno na aktuálních požadavcích

klienta, které jsou v iteracích vykonávány. Nedochází k dokumentaci požadavků, ale k vytváření uživatelských příběhů (user stories) [39]. Celý proces je postavený na kontrole plnění pomocí testování.

Typickou úlohou při extrémním programování je přepisování či úprava původního řešení (refaktorizace) [40]. Úkoly jsou tak zakládány bez povědomí o možných budoucích požadavcích s možností pozdějšího předělání v případě, že je to potřeba. Priorita vykonávaných úloh je určována termínem dodání.

1.8.2 Velikost týmů a počet zaměstnanců

Velké množství zaměstnanců vyžaduje větší množství úkolů a jejich náročnější správu. Nejen softwarové firmy proto rozdělují zaměstnance do menších týmů z důvodu větší produktivity. Způsob rozdělení týmů a jejich úkolů je pak ovlivněn například následující faktory:

- typ výkonné činnosti,
- klienti,
- kultura firmy,
- role zaměstnanců,
- dovednosti a zkušenosti zaměstnanců. [41]

Projekt je proto rozdělen úkoly, které se dále dělí na úlohy pro týmy či přímo pro jednotlivce. Jednotlivá zadání pak spravuje jeden, či více projektových manažerů.

1.8.3 Volba podpůrných systémů

Organizace pro správu úkolů nejčastěji využívají různé systémy. Moderní podpůrné systémy se zaměřují na řízení pracovního toku a nazýváme je process-aware information systems (PAIS). Jejich cílem bývá zpřehlednit dostupné kapacity v týmu, umožnit plánování, definici a správu úkolů za účelem zefektivnění práce v týmu. [42]

V dnešní době existuje hned několik desítek aplikací s rozdílnými funkcemi a uživatelským rozhraním. Podporovány jsou často funkce pro kompletní proces projektového řízení včetně zadávání úloh. Systémy často poskytují obecné používané nástroje jako např. Ganttův diagram či Kanbanovskou tabulku. Většina systémů s rozsáhlejšími funkcemi jsou často placené. Volba správného tak může výrazně ovlivnit efektivitu TM a náklady s procesem spojené.

Kapitola 2

Analýza zadávání úloh ve vybraných firmách

Tato kapitola se zabývá analýzou procesu zadávání úkolů na základě rozhovorů ve vybraných firmách. Cílem této kapitoly je seznámení se s průběhem procesu v reálném prostředí a identifikace nejčastěji řešených problémů s ohledem na strukturu a projektové řízení firmy. Vyhodnocení průzkumu pak poslouží jako základ pro návrh a implementaci podpůrného nástroje v praktické části.

2.1 Vybrané firmy

Firmy byly vybrány dle zadání této práce na základě jejich velikosti a zaměření. Mělo se jednat o malé až středně velké podniky se zaměřením na tvorbu softwarového nebo grafického produktu (agentury). Pro průzkum byly vedoucím práce domluveny následující čtyři firmy, které souhlasili se spoluprácí na projektu a projevili zájem o dané téma.

Ackee je agentura zaměřená na vývoj mobilních a webových aplikací. Vznikla v roce 2012 jako startup tří absolventů z ČVUT v Praze [43]. Mezi její nejznámější produkty patří aplikace App4Event či aplikace pro Německý spolkový sněm Bundestag app, která získala i mezinárodní ocenění German Brand Award[2] v kategorii Digitální řešení a aplikace [44]. Na otázky odpovídal jeden ze zakladatelů, aktuální Chief executie officer (dále CEO) společnosti a vedoucí této práce Ing. Martin Půlpitel.

Qest automation je firma specializující se na vývoj webových aplikací. Nabízí návrh, implementaci i testování informačních systémů nebo například chytrých domů. Firma vznikla v roce 2013 a stojí za realizací projektu Prezident 21 či tvorbou nového webu Mall.cz [45]. Rozhovoru se zúčastnil zakladatel a CEO společnosti Ing. Martin Koperniech.

[2]Německé prestižní ocenění od German Brand Institute.

2FRESH je společnost se zaměřením na výzkum a design. Nabízí řešení v oblastech UX a UI, reklamní produkce, marketingu či obchodních strategií. Firma vznikla v roce 2006 a zpočátku poskytovala i vývoj software. Mezi známé projekty patří například design O2 TV či implementace nástroje pro sledování cen a profitu s názvem Costlocker. Respondetem pro rozhovor byla Lenka Kůrková, tehdejší Chief operations officer (COO) společnosti. [46]

Digital Ant je agentura se specializací v marketingu převážně pro startup projekty. Funguje přes 8 let a zakládá si na vlastním agilním stylu tvorby projektu. Firma dělala kampaně například pro HobbyGo od ŠKODA AUTO DigiLab či projekt pro studentské bydlení THE FIZZ. Rozhovoru se zúčastnili oba zakladatelé firmy Ing. Štěpán Heller a Lukáš Obdržálek. [47]

2.2 Témata a způsob dotazování

Rozhovory probíhaly osobní schůzkou se zaměstnanci firem, kteří mají přehled o procesu TM v jejich společnosti. Ze schůzky byl pořízen audio záznam, který byl následně přepsán do písemné podoby. Souhlasy dotazovaných s pořízením audio záznamem jsou v příloze C. Přepsané rozhovory je možné nalézt v příloze D. Většina rozhovorů probíhala na podzim v roce 2019, aktuálně tak situace ve firmách nemusí odpovídat tehdejšímu stavu.

Témata byla vybírána na základě informací z analýzy procesu zadávání úloh z kapitoly 1. Každý rozhovor byl přizpůsoben ovlivňujícím faktorům procesu zadávání úkolů, které z odpovědí respondenta vyplynuly. Rámcová schéma rozhovoru vypadalo následovně.

- Úvod:
  - seznámení s respondenty,
  - typ jejich výkonné činnosti.
- Struktura firmy:
  - rozložení jejich výkonných týmů,
  - způsob projektového řízení,
  - role aktérů v procesu.
- Hlavní část:
  - popis procesu zadávání úloh v jejich společnosti,
  - jejich problémy spjaté s TM,
  - možnosti optimalizace procesu,
  - používané podpůrné systémy.

2.3 Velikosti firem a rozložení týmů

Respondenti byli dotázáni na velikost a rozložení týmů v jejich firmě. Přehled odpovědí je znázorněn v tabulce 2.1. Z nich plyne, že všechny firmy přizpůsobují velikost týmu potřebám projektu. Větší společnosti jako Ackee a 2FRESH poskytují větší rozsah služeb a rozdělují tak zaměstnance i do výkonných týmů dle zaměření.

V Ackee jsou to týmy rozložené podle technologií, ve kterých zaměstnanci pracují. Vytížení jednotlivých týmů závisí na požadavcích aktuálně zpracovávaných projektů. V 2FRESH jsou zaměstnanci rozděleni do divizí dle poskytovaných služeb. Každá divize funguje odděleně na svých požadavcích a projektech. V případě velkých zakázek jsou divize propojeny sdílenými úkoly. Pro každý projekt pak je většinou různý počet pracovníků. Firma Qest se snaží týmy udržovat stejně velké a zapouzdřené[3].

Název firmy	# zaměstnanců	Dělení týmů	Velikosti týmů
Ackee	60 až 70	dle technologií a dle projektu	dle projektu
Qest	35	dle projektu	5 až 6 lidí
2FRESH	40	do divizí[4]	dle divize[5]
Digital Ant	12	dle typu kampaně a dle projektu	dle projektu

Tabulka 2.1: Odpovědi respondentů na velikost firmy a rozložení týmů

2.4 Projektové řízení firem

Všichni dotazovaní potvrdili, že jejich společnosti fungují buďto na agilních metodikách projektového řízení, nebo zavádí v různých ohledech vlastní pravidla. Obě softwarové firmy alespoň na některých projektech aplikují metodiku Scrum. Qest se jejích pravidel drží striktně. Vývoj tedy běží v pevně určených iteračních cyklech. V Ackee jsou rozsahy požadavků klientů velmi různorodé. Z toho důvodu dochází k aplikaci i metodiky Waterfall či vlastních pravidel na bázi Scrumu při vývoji menších projektů.

V marketingových firmách je řízení z velké části dáno výkonnou činností. Vzhledem k návaznosti a závislosti úkolů jsou kampaně dělány lineárně. Ve firmě Digital Ant z důvodu malého množství zaměstnanců a velmi podobného schématu rozdělení požadavků nebyla prozatím nutnost aplikovat žádnou

[3]Na projektu pracuje vždy jen jemu přidělený tým.
[4]Dělení dle projektu probíhá jen v reklamní divizi.
[5]V reklamní divizi záleží na projektu. U ostatních divizí ne.

obecně využívanou metodiku. Projektové řízení je tak prováděno na základě úvodního plánování a následného sledování stavu na týdenních schůzkách.

V 2FRESH při tvorbě projektů postupují obecně dle metodiky Waterfall. Velké projekty jsou rozděleny na úkoly pro jednotlivé divize, kde jedna čeká na dokončení úkolů jiné. Každá divize dle Lenky Kůrkové aplikuje mírně odlišné přístupy řízení projektu.

2.5 Zadávání a správa úkolů

Zadávání úkolů je v Ackee, 2FRESH a Qest běžné dle metodik projektového řízení. V Digital Ant je na začátku prvotní schůzka, na základě které jsou vytvořeny seznamy úkolů. Seznam a úlohy bývají často podle určité šablony. Nepracuje se tak s detaily každého úkolu jako v případě softwarových firem, ale pouze s rámcovým zadáním. Správa vytvořených kampaní pak probíhá na základě reakcí od cílové skupiny či klienta.

2.5.1 Distribuce a delegace úloh

Rozdělení úkolů závisí primárně na dělení týmů. Digital Ant a Qest nepracují s úkoly, které je nutné dělit či delegovat napříč různými týmy. U Digital Ant je to kvůli nízkému počtu zaměstnanců, úkoly se tak přidělují přímo zaměstnanci. V Qest je jednoduchost zadávání úkolů způsobená zapouzdřením týmů pro jeden projekt. Požadavky se tak vždy dají rozdělit na úkoly pro jednotlivé členy a nemusí docházet ke složitému předávání napříč týmy. Situace v Ackee a 2FRESH je u velkých zakázek složitější, jelikož dochází k dělení úkolů mezi technickými týmy nebo divizemi a jejich možnému následnému předávání.

2.5.2 Určování priorit úkolů

U marketingových firem probíhá určování priorit v závislosti na dohodnutém termínu dodání. 2FRESH navíc odlišuje klientské a interní projekty. Klienské neodkladné požadavky mají vyšší prioritu než neodkladné úkoly k interním projektům. Obě firmy pro řízení pracovního toku využívají principy Kanbanové tabulky.

Softwarové firmy řeší priority dle metodiky Scrum iteračně. V Qest využívají funkce řazení úkolů v backlogu. Čím výše se tedy úkol nachází, tím dříve je odbaven. V Ackee jsou omezeni možnostmi podpůrného systému Redmine, který neumožňuje číselné priority u úkolů. Požadavkům jsou tak přiděleny štítky z definované množiny. Při stejných označeních u více úkolů tak dochází ke kolizím. Jako pomůcka je proto použita vlastní tabulka v Google Sheets[6] zobrazena na obrázku 2.1.

[6]Nebo také „Google Tabulky“ je aplikace pro tvoření programovatelných tabulek. [48]

	A	BD	BE	BF	BG	BH	BI
1		3. týden 10-2019 (14.10. - 20.10.)			4. týden 10-2019 (21.10. - 27.10.)		
2		1. projekt	2. projekt	3. projekt	1. projekt	2. projekt	3. projekt
9	Jirka Backend	Priorita 1	Priorita 2	Priorita 3			
10	Jirka Backend	Priorita 1	Priorita 2	Priorita 3			
11	Jirka Backend	Priorita 1	Priorita 2	Priorita 3			
12	Jirka Backend	Priorita 1	Priorita 2	Priorita 3			
13	Jirka Backend	Priorita 1	Priorita 2	Priorita 3			
14	Jakub Frontend	Priorita 1	Priorita 2	Priorita 3			
15	Jakub Frontend	Priorita 1	Priorita 2	Priorita 3			
16	Jakub Frontend	Priorita 1	Priorita 2	Priorita 3			
17	Jakub Frontend	Priorita 1	Priorita 2	Priorita 3			
18	Jan iOS	Priorita 1	Priorita 2	Priorita 3			
19	Jan iOS	Priorita 1	Priorita 2	Priorita 3			
20	Jan iOS	Priorita 1	Priorita 2	Priorita 3			
21	Jan iOS	Priorita 1	Priorita 2	Priorita 3			
22	Jan iOS	Priorita 1	Priorita 2	Priorita 3			
23	David Android	Priorita 1	Priorita 2	Priorita 3			
24	David Android	Priorita 1	Priorita 2	Priorita 3			
25	David Android	Priorita 1	Priorita 2	Priorita 3			
26	David Android	Priorita 1	Priorita 2	Priorita 3			

Obrázek 2.1: Ukázka používané tabulky ve firmě Ackee zaslaná Ing. Martinem Půlpitlem.

Tabulka slouží jako přehled tří projektů, na kterých každý týden členové týmů pracují. Pomáhá projektovým manažerům při sledování kapacit a aktuální práci vývojářů. Vývojářům pomáhá s určením aktuálního nejdůležitějšího úkolu. Např. pokud má pracovník dva kolizní úkoly v prioritě nebo termínu dodání ze dvou různých projektů, podívá se do tabulky na jemu přiřazené prioritní projekty, a zpracovává nejdříve úkol, který patří k prioritnějšímu projektu.

Přiřazené projekty nejsou striktní. V případě vzniku důležitějšího úkolu je projektovým manažerem zadán příslušnému zaměstnanci a je splněn nejdříve. Tabulka je vyplňována na společné schůzce, kde se i jednotliví členové týmů mohou vyjádřit, jaký úkol by chtěli dělat, případně kdo z nich je, vzhledem ke specializaci, na danou problematiku ideálním řešitelem.

2.6 Role zaměstnanců zapojených do správy úkolů

Kromě firmy Digital Ant by měl být dle respondentů v ideálním případě za proces zadávání úkolů zodpovědný projektový manažer případně při aplikování metodiky Scrum product owner. Tato osoba by měla na celý proces dohlížet a sledovat, zdali jsou všechny úkoly plněny dle klientských požadavků.

Agentura 2FRESH v době dotazování projektového manažera neměla. Jeho funkci nahrazuje account manažer, jehož primární funkcí by ovšem měla být konzultace se zákazníkem a budování nových obchodních příležitostí [49]. Lenka Kůrková potvrdila, že absence projektového manažera je i jedním z ak-

tuálně největších problémů při zadávání úkolů ve firmě. Oproti tomu v Digital Ant sledují stav úkolů vedoucí firmy společně s product owners.

2.7 Používané podpůrné systémy a nástroje

Všechny firmy v době dotazování používaly různou kombinaci podpůrných nástrojů pro TM a přidružené procesy, jako je sledování strávených hodin či správa dokumentů. Tabulka 2.2 poskytuje jejich přehled. V posledním sloupci se nachází vyjádření spokojenosti firmy s podpůrnými nástroji. Slovní označení odpovídající následujícím stavům.

Vysoká spokojenost značí, že firma nemá k systémům žádné větší výtky a plánuje je v budoucnu používat dál v aktuální konfiguraci nebo určitým způsobem propojit.

Střední spokojenost znamená, že společnost je se systémy spokojena, ale chybí jí v ní určité funkce (nebo je považují za drahé), nelíbí se jí uživatelské rozhraní nebo je nutné používat jiné vlastní nástroje k doplnění funkcionalit.

Nízká spokojenost označuje stav, kdy systémy mohou vyhovovat, ale firma s nimi není zcela spokojená a plánuje do procesu zapojit další nové systémy nebo některé vyřadit.

Název firmy	Systém na TM	Další podpůrné systémy	Spokojenost
Ackee	Redmine	Google documents	Střední
Qest	Azure DevOps	Costlocker, Legito	Vysoká
2FRESH	Notion	EasyProject[7], Costlocker	Nízká
Digital Ant	Asana	Costlocker	Střední

Tabulka 2.2: Používané systémy na podporu TM ve vybraných firmách.

Z tabulky vyplývá, že je obecně velmi důležité pro malé agilně řízené týmy sledovat finanční stavy projektů a úkolů. Vypovídá o tom využívání placeného nástroje Costlocker[8] ve třech ze čtyř vybraných firem. Nástroj ve firmách slouží pro zápis stráveného času (ať už pomocí stopek nebo ručního zápisu), pro statistiky nákladů a výnosů nebo pro uchovávání faktur.

2.7.1 Chybějící funkce

Kromě společnosti Qest nebyla ani jedna firma plně spokojena s funkcemi svých používaných systémů. V Digital Ant byl dříve používán systém Trello.

[7]Firma plánuje přechod na systém Float.
[8]Analytický nástroj pro zápis času od firmy 2FRESH [50].

U něho byl ale problém s nemožností definovat atributy dílčích úloh jako například řešitele. Proto společnost přešla na aplikaci Asana, kde je nyní určitým problémem cena. Při zadávání úloh vedoucím chybí možnost definovat závislé úkoly. Asana nabízí požadované funce, ale jsou dostupné pouze v placené verzi, kterou firma prozatím vyhodnotila jako finančně nevýhodnou. Náhradní řešení zatím v Digital Ant vymyšleno nebylo.

V Ackee jsou se svým systémem spokojeni. Díky architektuře Redmine, postavené na pluginech, mohla být aplikace do velké míry přizpůsobena potřebám firmy. Někteří zaměstnanci, kteří se zaučují, ovšem mívají problém s uživatelským rozhraním a také řešení priorit úloh je pro firmu v systému nedostačující.

2.7.2 Špatně zvolené systémy

V 2FRESH byl respondentem za problém označen kompletně celý systém na plánování projektů EasyProject. Údajně poskytuje mnoho funkcionalit, které zaměstnanci nepoužívají a hodnotí ho jako uživatelsky nepřívětivý. Pro zadávání úkolů ve firmě slouží Notion, který nabízí postačující funkce pro sledování stavu i zadávání úloh. Z pohledu manažera ale neposkytuje dostatečné funkce pro plánování kapacit a zaměstnanci vyžadovali přehledné zobrazení úloh v kalendáři. Všechny problémy se systémy spojené chtěla firma řešit v době dotazování pomocí zavedení nového podpůrného systému Float[9].

2.8 Vyhodnocení průzkumu

Rozhovory poskytly pohled na TM v reálném prostředí. Vzhledem k různým zaměřením, rozdělením týmů, projektovým řízením a používaným podpůrným systémům jsou zdroje problémů v každé firmě odlišné. Společnosti Qest a Digital Ant dokonce neshledávají žádný závažnější problém v aplikovaném procesu. Z odpovědí lze nicméně vyvodit několik kritických oblastí, které jsou i za použití podpůrných systémů stále zdrojem neefektivity.

2.8.1 Problémové oblasti

Všichni respondenti označili některé oblasti v TM za zdroje chyb, či zmínili, že by daná oblast vyžadovala optimalizaci. Četnost zmínění v rozhovorech seřazených dle nejvyššího je znázorněna v tabulce 2.3.

Nejzmiňovanějším problémem je lidský faktor, neboli individuální chyby zaměstnanců, při zadávání nebo plnění požadavků, nesouvisející se zavedeným procesem. Respondenti se také shodují, že eliminace takového problému je prakticky nemožná. Snižovat chybovost se snaží například v Qest pomocí

[9]Systém na plánování kapacit pomocí kalendáře. [51]

Problémová oblast	Četnost zmínění v odpovědích
Lidský faktor	4
Podpůrné systémy	3
Závislosti úkolů	2
Priority úkolů	2

Tabulka 2.3: Četnost zmínění problémových oblastí při zadávání úkolů v rozhovorech

zvyšování informovanosti a přehledu všech zaměstnanců pomocí debat a možností dodefinování zadání na každodenních stand-up schůzkách.

Nedostatečné funkce, cena, nebo špatný výběr podpůrného systému byly druhou nejzmiňovanější kritickou oblastí. Jedinou plně spokojenou firmou byla Qest s využitím Azure DevOps přímo pro potřeby agilního vývoje v metodice Scrum. Faktor projektového řízení může být v tomto ohledu rozhodující. Ostatní firmy aplikují více odlišných metodik, pro které systémy buďto nemusí mít nativní podporu, nebo neposkytují dostatek podpůrných funkcí. Vzhledem k velikosti firem pak komplexnější systémy mohou být finančně nevýhodné. V 2FRESH špatná původní volba systémů vedla až ke změně podpůrného nástroje pro plánování.

Třetí a čtvrtá nejzmiňovanější problémová oblast spolu souvisí, jelikož závislost úkolů je řešitelná pomocí správného nastavených priorit. Prvním zmiňovaným problémem byl nedostatečný přehled v podpůrných systémech o závislých úkolech a prioritách. Druhým byla nutnost rozhodnutí pracovníka v případě kolizí, který úkol bude dělat dříve. Řazení úkolů v backlogu může tento problém vyřešit, jak ukázala firma Qest. Tento postup ovšem není aplikovatelný na všechny firmy a ne všechny systémy obsahují podporu pro backlog.

2.8.2 Procesy založené na schůzkách

Kromě obecných postupů jako změny stavů nebo řešitele, pro jejichž účely používají všechny firmy určité aplikace, si lze na zavedených procesech všimnout nutnosti setkávání. Při zadávání úkolů je ve všech firmách na začátku projektu nebo iterace schůzka, kde se upřesňují detaily s klientem. Na základě těchto schůzek vzniká odhadování náročnosti, termínů, potřebných kapacit a rozdělení úkolů. Kromě Lenky Kůrkové z 2FRESH všichni respondenti také mluvili o pravidelných schůzkách, na kterých dochází buďto ke kontrole stavu řešení úkolů nebo předání informací o nových požadavcích. Na setkáních jsou i v případě firmy Ackee a Qest řešeny jejich kritické oblasti při TM spojené mimo jiné i s lidským faktorem.

Kapitola 3

Analýza podpůrných systémů

Tato kapitola se zabývá analýzou systémů pro podporu TM, které jsou používány v malých a středně velkých firmách. Cílem této kapitoly je identifikace funkcí a nástrojů pro jednotlivé části procesu zadávání úloh, které systémy nabízí, a získání přehledu o již existujících řešeních.

Analýza systémů bude založena na uživatelském otestování funkcionalit spojených se zadáváním úloh autorem této práce. V případě, že nějaký systém či jeho nástroj nebude možné otestovat, bude to v práci výslovně uvedeno. Vzhledem k velkému množství existujících aplikací byly pro podrobnou analýzu vybrány ty, které se vyskytly v odpovědích respondentů z rozhovorů. Zdrojem informací budou také oficiální webové stránky systémů.

3.1 Sledovaná kritéria

Na základě obecných požadavků na software a výsledků průzkumu lze stanovit několik následujících kritérií pro hodnocení a porovnávání podpůrných systémů. U každého z nich bude přihlédnuto k danému typu softwaru a jeho cílové skupině dle oficiálních webových stránek.

Podporované funkce systémem jsou pro firmy nejdůležitějším faktorem při rozhodování o jeho používání. Hodnocen bude rozsah a možnosti poskytovaných funkcí pro jednotlivé procesy v rámci zadávání úkolů. Pro firmy může být také důležité, aby software podporoval funkce spojené s metodikami projektového řízení, které v týmech aplikují. Bude proto sledováno, pro jaké typy projektových řízení se aplikace hodí.

Cena se ukázala být na základě průzkumu pro malé podniky důležitá. Některé aplikace nebo její části mohou být příliš drahé nebo se investice do nich firmě nevyplatí. Cena bude hodnocena dle porovnávání aktuálních nabídek systémů.

Uživatelské rozhraní (dále UI) a jeho použitelnost a grafické zpracování může hrát klíčovou roli při vybírání jakéhokoliv softwarového produktu. Různé typy interakce systému s uživatelem mohou mít vliv na efektivitu používání daného nástroje. Firmy pak raději sahají po systémech, jejichž používání je nejsnadnější a pro zaměstnance zároveň rychle pochopitelné. Hodnocena bude proto praktická přijatelnost řešení z pohledu funkčnosti a efektivnosti práce s nástroji.

Přístupnost může být důležitá v případě, že aktéři v procesu TM využívají různé přístroje pro práci například mobilní telefon či tablet. Umožnění jednoduchého přístupu z těchto zařízení pak může napomoci při sledování a zadávání úkolů v situacích, kde není možný přístup ze zařízení s vysokým rozlišením.

Možnosti propojení s dalšími systémy mohou být dle průzkumu důležité, jelikož jsou využívány i další podpůrné nástroje se zadáváním úloh související. Například napojení na aplikaci pro komunikaci či sledování stráveného času na úlohách.

3.2 Trello

Trello je jednoduchý a v základní verzi zdarma dostupný nástroj pro podporu správy úkolů. Je součástí balíčku pro projektové řízení od firmy Atlassian. V roce 2019 společnost evidovala více než 50 milionů uživatelů této aplikace [52]. Trello je dostupné jako webová aplikace běžící v cloudu. [53]

Autoři poskytují další dvě verze programu za měsíční platbu za každého uživatele s množstevní slevou v případě nejdražší verze. Cena se pohybuje v rozmezí 10 až 17 USD za uživatele pro firmy do 100 zaměstnanců dle zvolené verze. Rozšíření otevírají možnosti většího množství nástěnek a jejich vylepšení (šablony, skupiny, ...) nebo používání týmů a možnosti napojení na více než 100 aplikací jako například Slack[10], Google Hangouts[11], Gmail nebo další aplikace od firmy Atlassian, jako jsou Jira či dokumentový systém Confluence.

Testována byla nejdražší verze ve čtrnáctidenní zkušební době. Aplikace poslouží v této práci pro seznámení se základními nástroji pro zadávání úkolů, které budou u dalších systémů porovnávány.

[10] Komunikační platforma pro týmy [54]

[11] Nástroj pro komunikaci od společnosti Google [55].

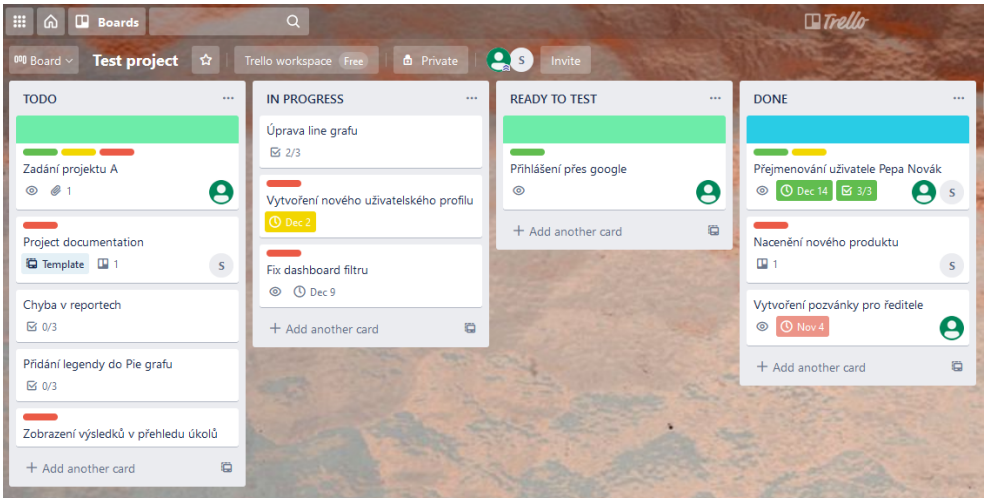
3.2.1 Funkce pro správu úkolů

Trello umožňuje založení nástěnek obsahující seznamy kartiček. Kartičky představují jednotlivé úkoly a seznamy určují stavy ve kterých se nacházejí. Aplikace tak velmi efektivně a jednoduše umožňuje vytvořit nástěnky, které budou fungovat jako Kanbanovská tabulka při agilním vývoji. Vytvořenou Kanbanovskou tabulku při testování je možné vidět na obrázku 3.1.

Každá kartička obsahuje vlastní detail, kde lze přidat další vlastnosti jako jednoduše formátovaný popis, barevné štítky, termín dokončení, seznam nutných kroků pro splnění (checklist) nebo přílohy s limitem velikosti dle používané verze. V aplikaci nebyla nalezena podpora pro tvorbu dílčích úkolů, což může být zásadním problémem při velkých projektech. Náhradou může být seznam s checkboxy v kartičce, nicméně k prvkům seznamu již není možné přiřadit nic dalšího.

Velmi užitečnou funkcí je vytváření šablon pro detaily kartiček. Lze tak snadno rozlišit různé struktury pro jiné typy úloh (feature, issue). Založení daného typu úkolu je pak velmi rychlé pomocí zvolení připravené šablony. Pro podporu sledování stavu úkolů slouží notifikace, u kterých je možné nastavit zasílání do emailu. Zajímavá funkce je možnost na notifikaci odpovědět ihned přes obdržený email.

V detailu se také objevuje podpora komunikace prostřednictvím komentářů. V nich lze přidat další přílohy či označit uživatele. Lze také zobrazit historii změn na kartičce či seznam všech akcí na nástěnce v postranním panelu nástěnky. Kromě zobrazení na nástěnce je možné přepnout sledování úkolů i do kalendáře, kde jsou rozřazené podle termínů možného dokončení.



Obrázek 3.1: Ukázka nastavené Kanbanovské tabulky v aplikaci Trello při testování aplikace.

Problémové pro funkce systému může být velké množství úkolů a členů týmu. Seznamy úkolů totiž není možné řadit ani filtrovat podle priorit ani řešitelů. Při přidávání nových úkolů se tak seznamy stále rozšiřují do spodní části stránky a je nutné nimi dlouho při hledání listovat. Pokud by uživatelé chtěli pro každého zaměstnance pro zmenšení seznamů vytvořit speciální nástěnku, je to možné, jelikož úkoly lze přesouvat i mezi jednotlivými nástěnkami. Bohužel tím, z důvodu nutnosti si vždy rozkliknout nástěnku daného uživatele, může klesat přehlednost pracovního toku. Efektivnost tak záleží na zvoleném přístupu využívání poskytnutých nástrojů.

3.2.2 Uživatelské rozhraní a přístupnost

Trello je díky možné interakci uživatele téměř s každým prvkem na obrazovce rychlé při používání. Umožňuje například přesouvání kartiček i seznamů pomocí jejich přetažení a většina dostupných informací lze upravit pomocí jednoho až dvou kliknutí myši. V aplikci je také podpora několika klávesových zkratek. Všechny úpravy jednoho uživatele jsou ihned propagovány ke všem ostatním, v prohlížeči je tak vždy zobrazen aktuální stav. Nevýhodou při interakci je formátování textu v komentářích, to neprobíhá pomocí WYSIWYG editoru[12], ale pomocí formátových značek.

Aplikace umisťuje všechny prvky uživatelského rozhraní pouze do dvou hlavních oken – nástěnky a detailu kartičky. Užitečné pro informovanost uživatele o stavu úkolu je zobrazování malých ikon nebo štítků se shrnutím z kartičky. Při přidání nových položek k úkolu se tak důležité prvky zobrazují i přímo na nástěnce a uživatel tak nemusí otevírat detail.

Trello je díky responzivnímu designu webového rozhraní dobře přístupné jak z tabletu, tak z mobilního zařízení. Kvůli velkému množství prvků na nástěnkách začíná být na menších obrazovkách rozhraní poněkud nepřehledné. Nabízena je možná i z toho důvodu aplikace pro systémy Android i iOS. Vyzkoušena byla aplikace pro Android [56], která problémy nepřehledného zobrazení řeší.

3.2.3 Shrnutí

Trello je nástroj obsahující sadu všech základních funkcionalit pro správu úkolů a jejich delegaci v malých týmech. Všechny nástroje jsou jednoduše přizpůsobeny systému nástěnek a kartiček. Aplikace tak nenabízí podporu pro další procesy v projektovém řízení, jako je např. plánování či tvorba dokumentace. Přímočarost aplikace sice zlepšuje přehlednost a jednoduchost UI, ale s narůstajícím množstvím úkolů a členů týmu může být nástroj nedostačující. Hodí se tak převážně pro velmi malé týmy s jednotkami zaměstnanců.

[12]Název je zkratkou pro „What you see is what you will get“. Takový editor zobrazuje přímo formátovaný text bez netisknutelných znaků.

3.3 Asana

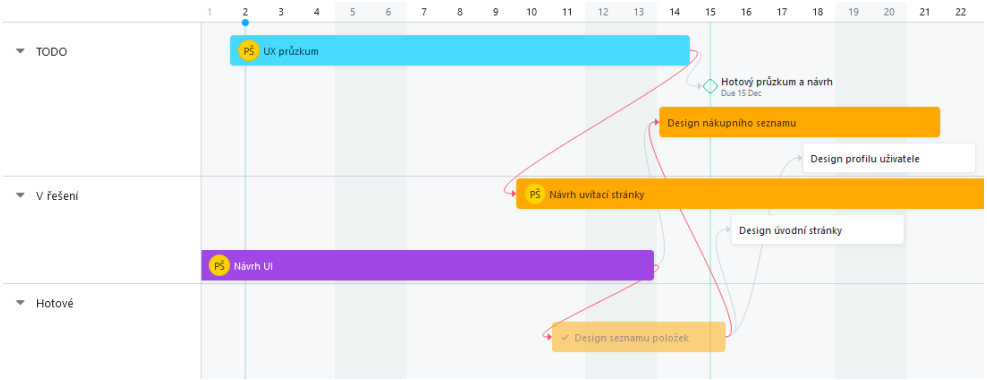
Asana je aplikace se spoustou funkcionalit nezávislých na tom, zdali je tvořen softwarový či jiný produkt. Nabízené funkce jsou dány různě drahými balíčky. Úvodní balíček je zdarma, další ceny balíčků se odvíjí od počtu uživatelů a platí se měsíčně. Ceny uvedené na stránkách jsou 10,99 USD za business verzi a 24,99 USD za premium verzi. Základní verze aplikace umožňuje kolaboraci maximálně patnácti členů týmu a poskytuje jen základní funkce při zakládání úkolů. Pro účely diplomové práce byla testována premium verze ve zkušební době. [57]

3.3.1 Zobrazení a definice úkolů

Funkcionality z neplaceného balíčku jsou velmi podobné aplikaci Trello. Místo nástěnek je zde definice projektů. Tyto projekty pak mají kategorie, do kterých spadají dané úkoly. K úkolům lze oproti Trello definovat v business verzi i úlohy závislé nebo dílčí.

U úloh je možné kromě termínu zvolit časový úsek, ve kterém by se měl vykonávat. Díky tomu Asana otevírá možnost k alternativním zobrazením workflow. Kromě Kanbanovské tabulky a kalendáře se v aplikaci nachází zobrazení v seznamu nebo časové řadě. Časová řada je prakticky ineraktivní Ganttův diagram a je dostupná pouze v premium verzi. Každé zobrazení je efektivní pro různé druhy činností. V seznamu lze efektivně filtrovat, v časové řadě zase efektivně určovat termíny a v tabulce se snadným přetažením mění stavy úkolů.

Otázkou je, zdali je kategorizace úkolů dle stavu, které i defaultně aplikace nastavuje, efektivní i pro ostatní zobrazení. Např. v časové řadě by mohlo být efektivnější mít úkoly tříděné dle typu zaměření. Tento problém je zobrazen na obrázku 3.2. Aplikace neumožňovalo třídění v době testování vypnout.



Obrázek 3.2: Ukázka problému řazení v časové řadě v systému Asana na testovacím projektu. Oranžové úkoly ze stejné kategorie jsou nepřehledně tříděné dle stavu.

3.3.2 Sledování stavu úkolů

Oproti Trello umožňuje Asana velmi snadné filtrování a řazení úkolů dle uživatelsky definovaných kategorií. Každý styl zobrazení se pak hodí pro různé role v týmu. Výkonný pracovník může používat seznam jemu přiřazených úkolů. Projektový manažer při plánování může používat časovou řadu či Kanbanovskou tabulku.

Pro sledování úkolů je v placené verzi dostupný nástroj „dashboard“. Jedná se o stránku s několika grafy, kterým lze měnit sledované atributy úkolů na osách. Nástroj ale v době testování nenabízel moc prostoru pro přizpůsobení, obsahoval pouze základní sadu grafů a je opět dostupný pouze v business verzi aplikace.

3.3.3 Podpůrné funkce projektového řízení

Podpůrnou funkcí pro plánování projektu a pracovního toku jsou formuláře. Ty dle oficiálních stránek mohou sloužit k definici user stories či pro sbírání kolektivních informací o projektu. Mezi další funkce patří definice cílů, u kterých lze nastavit různé metriky, nebo vytváření portfolií, díky kterým lze definovat rozpočty projektů a u projektů nastavit jejich priority.

Dalším nástrojem je definice statusů projektu. U nich je možné přidávat přílohy či různě popsat v jakém stavu v daný okamžik projekt je. Pro podporu komunikace je kromě klasických komentářů umožněno zakládat konverzace v týmech. Všechny tyto funkce jsou opět dostupné pouze v business verzi aplikace.

3.3.4 Uživatelské rozhraní

Uživatelské rozhraní je poměrně nesjednocené. Prakticky každá stránka má jiné rozložení, což uživatelům může dělat problém při orientaci. Asana nicméně téměř pro každou funkci poskytuje návod přímo na stránce, případně umožňuje náhled do ukázkového projektu, který je výsledkem dané funkce. Pro přehlednost slouží definice barev jednotlivých atributů úkolu, podle kterých je pak úkol zabarven i v přehledu.

Aplikace nabízí oproti Trello více klávesových zkratek. Webové rozhraní je neresponzivní a pro použití na mobilních zařízeních je doporučena oficiální aplikace. Vyzkoušena byla verze pro Android [58], která ovšem neobsahuje některé placené funkce jako např. zobrazení na časové řadě. Mobilní verze tak slouží převážně pro rychlou kontrolu či zadávání úkolů.

3.3.5 Shrnutí

Asana nabízí podobné funkce jako Trello obohacené o nové nástroje. Může tak sloužit jako jeho přímá alternativa. Poskytované funkce jsou užitečné nejen pro správu úkolu, ale i pro obecné projektové řízení. Díky filtrování, řazení a

podpoře plánování se hodí i pro větší týmy. Za většinu užitečných funkcí, které mohou být při TM potřebné např. definice závislostí úkolů, se ovšem musí připlatit. Cena business verze obsahující nejdůležitější rozšíření aplikace je srovnatelná s Trello, premium verze se všemi funkcionalitami je již podstatně dražší.

3.4 Notion

Aplikace Notion nabízí hned čtyři placené verze programu v rozmezí 4 až 8 USD za uživatele měsíčně. Pro umožnění členění uživatelů do týmů je nutné pořízení nejdražší verze, která byla ve zkušební verzi testována. [59]

3.4.1 Nástroje pro správu úkolů

Notion své funkce u projektů zaobaluje do tzv. pracovních prostorů. Každý pracovní prostor může nést vlastní úkoly nebo je může sdílet s ostatními. Stejně jako Asana umožňuje různé pohledy na přehled úkolů. Kromě Kanbanovské tabulky jsou poskytovány také kalendář, galerie, časová řada či obyčejná tabulka.

Při založení nového projektu aplikace nadefinuje základní strukturu, kde je vidět, jak se jednotlivá zobrazení dají využít jako nástroje při agilním řízení týmu. Pro řízení pracovního toku je určena Kanbanovská tabulka. Pro plánování je použita časová řada. Pro definici epics a user stories je využita obyčejná tabulka, kde lze přidávat či odebírat sloupce a řádky a používat základní matematické formule jako v například v Google Sheets.

V aplikaci nebyla v době testování nalezena podpora pro definici závislých ani dílčích úkolů a zapisování stráveného času. Určování priorit je řešeno stejně jako u Trello nebo Asana. V testovací verzi ovšem není možné v různých pohledech řadit a filtrovat dle vlastních parametrů u úkolů. Oproti ostatním systémům má Notion podporu pro správu dokumentů a tvorbu wiki. Vytvářeny jsou pomocí blokového editoru s množstvím maker např. pro vložení kódu nebo označení členů týmu. Editor je v době testování bez větší podpory formátování textu, nabízí pouze základní makra např. pro nadpisy. Dostupné jsou také exporty do formátů pdf, csv, markdown či HTML.

3.4.2 UI, přístupnost a možnosti napojení

UI je jednoduché a interaktivní. Uživateli je umožněno přesouvat jednotlivé prvky a vše editovat pomocí kliknutí na příslušný prvek v pracovním prostoru. Rozmístění prvků je v některých případech poněkud nekonvenční. Například tlačítko pro odhlašování uživatele se nachází v seznamech projektů. K přístupnosti opět slouží několik klávesových zkratek, které jsou často popsány přímo v aplikaci. Přístup z tabletů a mobilních zařízení je zajištěn pomocí

aplikace pro systémy Android a iOS. Testovaná aplikace pro Android [60] je přehledná a obsahuje všechny funkce webového rozhraní.

Pro integraci s jinými podpůrnými systémy jako Trello či Asana slouží možnost importu projektu ze souboru, který může být z ostatních systémů vyexportován. Jiné formy propojení přímo ze strany Notion nebyly nalezeny. Autoři aplikace aktuálně pracují na aplikačním rozhraní (Application Programming Interface dále API), které umožní napojení do libovolného systému [61].

3.4.3 Shrnutí

Aplikace Notion je díky možnosti definic vlastních pracovních prostorů s tvorbou tabulek a dokumentace univerzální nástroj. Mnoho různých pohledů na úkoly umožňuje jak sledování stavu úkolů, tak plánování. Vzhledem k nízké ceně v porovnání s ostatními systémy se může hodit malým týmům, které nevyžadují pokročilé možnosti při definici úkolů (závislosti úkolů, číselné priority, zápis stráveného času) a nechtějí využívat jiné systémy pro tvorbu dokumentů. Pro sledování stráveného času na úkolech je nutné využít externí systém.

3.5 Redmine

Známou a používanou aplikací je Redmine. Je vyvíjena od roku 2006 a její zdrojový kód je dostupný zdarma na jejích oficiálních stránkách[13]. Aplikace je napsána v jazyku Ruby a byla testována na lokální instanci, v online dostupné demo verzi a s omezenými právy na produkční instanci firmy Ackee. [62]

3.5.1 Funkce pro správu úkolů

Redmine nabízí kompletní sadu nástrojů pro TM a některé přidružené procesy v rámci projektového řízení. K projektům lze psát dokumentace pomocí WYSIWYG editoru. Pro editor lze doinstalovat několik pluginů pro různé typy syntaxe (příklad markdown). Při zakládání projektu je možné zvolit, které funkcionality budou v projektu používány pomocí modulů. Mezi defaultní moduly patří:

- sledování úkolů pomocí notifikací,
- sledování stráveného času na úlohách,
- tvorba dokumentace a wiki,
- nahrávání souborů,
- zobrazení kalendáře,

[13] V době testování byla aplikace chráněna licencí GNU General Public License verze 2.

- Ganttův diagram,
- umožnění tvorby diskuzních fór nebo
- zobrazení novinek v projektu.

3.5.2 Atributy a priority úkolů

Aplikace obsahuje definici úkolů podobnou Trello a Asana. Pro úkoly lze vybrat na rozdíl od Trello vždy jen jednoho řešitele. Pro přiřazení více lidem je nutné vytvořit pracovní skupinu. Redmine také podporuje strukturování projektů a úkolů do stromů. U úkolů je možné nastavit další závislé úkoly. V nastavení projektu lze definovat kategorie úkolů, jejichž přidělením je pak úkolu automaticky přiřazen řešitel z kategorie. Stavy úkolů jsou řešeny přiřazením štítku dle předdefinovaného pracovního toku. Pro změnu definovaných štítků je potřeba použít rozšiřující plugin.

Priority úkolů jsou také řešeny pomocí štítků. Není tak jednoduše možné nastavit číselné priority některým úkolům. Na oficiálních stránkách existuje možné řešení pomocí pluginu pro Scrum řízení týmu [63]. Ten pak také poskytuje nové funkce jako řazení pomocí backlogu.

3.5.3 Sledování stavu projektu

Změny na daných úkolech je možné sledovat pomocí notifikací. Základní verzi je opět možné rozšířit o plugin, který umožňuje nadefinovat různé akce, při kterých je notifikace zaslána na email [64]. Oproti Trello a Asana je v aplikaci komplexněji vyřešeno vyhledávání. Každý uživatel může definovat vlastní sadu filtrů přes všechny nastavené atributy úkolů a je možné podle nich i přehledně úkoly řadit.

Sledování stráveného času na úlohách probíhá pomocí zapsání čísla v hodinách s textovou poznámkou. V detailu úkolu je také ve výchozím stavu aplikace zobrazeno procento dokončení úkolu, které musí být měněno ručně a nereflektuje tedy počet strávených hodin.

Funkcionality pro definici úkolů a jejich filtrování poskytují základ pro všechny typy projektového řízení. Pro podporu plánování je zde již zmíněná podpora Ganttova diagramu a možnost zobrazit úkoly v kalendáři. Pomocí dalších nainstalovaných pluginů lze docílit i dalších funkcionalit, které mohou být velmi užitečné pro specifické řízení projektu.

3.5.4 Napojení na externí systémy

Na stránkách Redmine je nabízen seznam několika externích aplikací, které již na systém napojení obsahují. Jsou mezi nimi například pluginy pro prohlížeče Firefox či Chrome. Oproti Trello a Asana je možné také zdarma využít již

existující API, kterou lze jednoduše používat pomocí klíče vygenerovaného v administraci.

I autentizaci je možné řešit vlastními způsoby. Systém podporuje Lightweight Directory Access Protocol (zkráceně LDAP) s možnou definicí vlastních přístupových bodů. Všechny zmíněná napojení je také možné řešit přes vlastnoručně napsané pluginy.

3.5.5 Uživatelské rozhraní a přístupnost

Redmine oproti ostatním placeným systémům není příliš uživatelsky přívětivý. Téměř každá akce vynutí načtení nové stránky. Grafika aplikace kopíruje funkční styl starších desktopových aplikací. Díky pluginům je možné vzhled aplikace přizpůsobit, nebo pro určité akce díky dostupné API vytvořit aplikaci vlastní. Přizpůsobení aplikace ovšem vyžaduje znalost programování v jazyku Ruby.

Aplikace je v základní verzi neresponzivní, nenabízí tak podporu pro mobilní zařízení a tablety. Existuje ale hned několik řešení v podobně neoficiálních mobilních aplikací pro iOS a Android. Vyzkoušena byla nejstahovanější aplikace Redminer [65], která nabízí UI přizpůsobené Material designu[14]. Problémem těchto externích aplikací je, že neobsahují podporu pro pluginy, které jsou do Redmine nainstalovány. Lze tak přes ně sledovat jen základní funkcionality systému.

3.5.6 Shrnutí

Redmine je funkční nástroj pro správu úkolů, který je možné díky open source licenci a pluginům do velké míry přizpůsobit a obohatit. Jelikož pluginy nejsou oficiální, nemusí tak být podporovány nově vydanými verzemi aplikace nebo mohou způsobovat výkonností problémy. Některé pluginy jsou navíc také zpoplatněny. V základní verzi je aplikace bohužel uživatelsky nepřívětivá a nabízené funkce mají značná omezení. Instalace pluginů či napojení na další aplikace vyžaduje znalosti programování, nehodí se tak pro týmy, které hledají kompletní řešení, které nebudou muset nasazovat na server a spravovat.

3.6 Jira

Další aplikací pro podporu TM od firmy Atlassian je Jira. Cena aplikace je účtována podobně jako u Trello a Asana za uživatele měsíčně. Používání aplikace v cloudu je zpoplatněno od 7 do 14 USD za uživatele dle užívané verze. Při objednání dalších systémů od Atlassian (Trello, Confluence[15], Bitbucket[16]) je cena za každý produkt o něco nižší. [67]

[14] Příručka pro návrh UI od společnosti Google [66].

[15] Aplikace pro tvorbu dokumentace.

[16] Nástroj podporující vývoj pro verzovací nástroje Git a Mercurial.

Jira byla testována ve zkušební době v nejdražší premium verzi. Byly otestovány základní funkcionality spojené s definicí projektu a úloh, tvorbou grafů a základní nástroje pro podporu agilního vývoje.

3.6.1 Konkurenční výhody

Oproti Redmine je dostupná aktivní podpora od Atlassian a stálý vývoj nových funkcionalit. Aplikace může také fungovat v cloudu bez nutnosti nasazení na vlastní server. Základní nastavení pro malé týmy je ulehčeno díky úvodnímu návodu a mnoha šablonám nastavení aplikace pro různě řízené týmy.

Aplikace je postavena na principech agilních metodik. Kromě podpory běžných funkcí, které nabízí Redmine tak má firma k dispozici kompletní balíček vzájemně propojených nástrojů z metodik Scrum nebo Waterfall. Například ukázka použití nástroje backlog je na obrázku 3.3. Většinu funkcí navíc nalezneme i ve verzi zdarma, která je ovšem omezena kapacitou 10 lidí na aplikaci. Uživatelé mohou pro správu úloh využít:

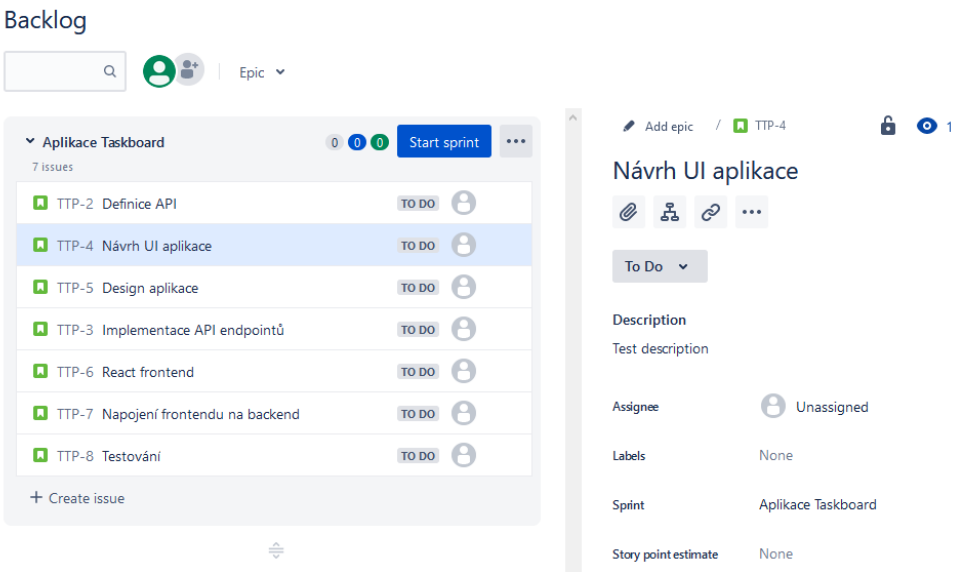
- definici sprintů,
- definici epics,
- vlastní workflow,
- Kanbanové tabulky,
- Roadmapy projektu či
- backlog.

Jira také obsahuje spoustu reportingových nástrojů, jako jsou grafy nebo tabulky se statistikami. Všechny tyto nástroje jsou plně přizpůsobitelné. Mohou tak být sledovány různé metriky plnění úkolů jako např. chybovosti nebo doby trvání. Zaznamenávat strávený čas lze číselným vyjádřením k úkolu či pomocí spuštění stopek přímo v aplikaci. Výsledky stráveného času lze taktéž zanést do reportingových funkcionalit, které lze jednoduše exportovat například do formátů pdf a xls.

Všechny formy napojení na externí systémy jsou dostupné pouze v placených verzích. Je možné využít velkého množství pluginů, případně zabezpečenou API. Jira nabízí také vestavěné napojení na verzovací systémy. Lze tak jednotlivé úkoly napojit např. na větve ve verzovacím systému Git.

3.6.2 Uživatelské rozhraní

UI je závislé na zvolených nástrojích. Stejně jako u Asana má každá stránka jiné rozložení prvků. Opět je pro řazení použito přesouvání prvků např. v Kanbanovských tabulkách nebo Backlogu. Další systémy od Atlassian jako



Obrázek 3.3: Ukázka použití backlogu v aplikaci Jira na testovacím projektu.

Confluence či Bitbucket jsou navrženy ve stejném grafickém stylu a zvykání si na UI je tak pro uživatele jednodušší. Oproti Trello zde chybí podpora propagace aktuálního stavu projektů ke všem uživatelům. Pro zpřístupnění aktuálního stavu je tak nutné aplikaci přenačíst.

Pro přístupnost z mobilních zařízení a tabletů je poskytnuta mobilní aplikace pro Android i iOS. Testována byla verze pro Android [68]. V ní jsou poskytnuty všechny známé funkce z webového rozhraní včetně statistických nástrojů. Jediným zjištěným nedostatkem bylo nezobrazování kompletní historie akcí u úkolů.

3.6.3 Shrnutí

Jira obsahuje velké množství funkcí primárně pro podporu agilních metodik. Díky zabudovaným reportingovým nástrojům je možné sledování chyb v procesu TM. Velká míra přizpůsobení aplikace a kvalitní uživatelské rozhraní činí z aplikace komplexní řešení pro různě řízené týmy. Jira je využívána i ve velkých firmách jako je Samsung, Visa nebo Hitachi [69]. Pro podporu dalších funkcionalit jako např. UML je možné dokoupit pluginy.

Pro agilně řízené vývojářské firmy se jeví jako ideální řešení z aktuálně nabízených produktů. Pro grafická studia je Jira možná příliš komplexním nástrojem, navíc bez pluginů není např. možné zobrazení úkolů v kalendáři nebo použití Ganttova diagramu. Bez zakoupení systému Confluence také není možné tvořit externí dokumentace. Nicméně vzhledem k tomu, že cena nejdražší verze je nižší než u nástrojů Trello a Asana, může být Jira taktéž poten-

cionálně vhodnou podpůrnou aplikací i pro agentury bez zaměření na vývoj software.

3.7 Azure DevOps

Microsoft nabízí stejně jako Atlassian nástroje pro správu úkolů a napojení na verzovací systémy prostřednictvím cloudového řešení Azure DevOps. Kompletní proces TM pak probíhá prostřednictvím služby Azure Boards. Základní verze poskytující kompletní správu úkolů je dostupná za přibližně 6 USD za uživatele měsíčně a byla testována ve zkušební lhůtě. [70]

3.7.1 Podporované funkce

Pomocné funkce jsou stavěné na agilním vývoji softwaru. V systému lze nalézt 5 základních kategorií: seznam úkolů, Kanbanovskou tabulku, definici sprintů a epics, backlog a vlastní filtry. Úkoly obsahují stejné atributy jako Jira. Jejich rozmístění v detailu úkolu je upravitelné. K úkolu je možné také přímo napojit větev z verzovacího nástroje Git. Pro definici závislých a dílčích úkolů slouží odkazy na další úkoly. Kromě vazeb pro závislost a dědičnost jsou zde i další typy přizpůsobené převážně pro použití při vývoji softwaru jako např. vazby pro testování.

Systém obsahuje také statistické nástroje, které si každý uživatel může definovat na vlastní stránce s přehledem. Na rozdíl od systému Jira jsou stránky s přehledy ale definovány vždy pouze pro daný projekt. Na úvodní stránce aplikace je pouze seznam aktuálních úloh ze všech projektů a není tak možné nastavit jednu společnou stránku pro všechny projekty či vytvářet obecné statistické stránky jako u aplikace Jira.

3.7.2 UI, přístupnost a napojení na další systémy

Azure DevOps přebírá vzhled ze všech ostatních nástrojů od firmy Microsoft. Míra možnosti přímé interakce s rozhraním je někde na rozmezí mezi systémy Redmine a Jira. Některé prvky lze měnit pomocí kliknutí a přetahování, jiné vyžadují přesunutí na amdinistrátorská stránku. Webové rozhraní je plně responzivní a z toho důvodu Miscrosoft neposkytuje oficiální DevOps mobilní aplikaci. [71]

Aplikaci lze jednoduše napojit na další služby od firmy Microsoft. Služby od jiných společností je možné napojit prostřednictvím služeb Pipelines, Azure Resource Manager, případně lze využít API. [72]

3.7.3 Shrnutí

Azure DevOps je systém s funkcionalitami velice podobnými systému Jira. Je levnější při použití základních verzí, nenabízí ale tak velkou míru přizpůsobení.

Oproti konkurentům není univerzálním řešením pro různé typy firem, jelikož je přímo orientovaný na agilní vývoj softwaru s hlavní podporou pro nástroje metodiky Scrum. Hodí se tak pro vývojářské firmy, které využívají více aplikací od firmy Microsoft.

3.8 Vyhodnocení

Všechny analyzované systémy nabízí vhodné a často podobné funkce pro zadávání a základní definici úkolů. Rozdíly mezi aplikacemi jsou převážně v jejich cílové skupině uživatelů, čemuž přizpůsobují své nástroje. Další podpůrné aplikace (např. Costlocker či Float) pak kvalitně řeší chybějící funkce pro odhadování kapacit či zápis stráveného času a s ním spjaté sledování finančního stavu projektu. Tím je proces zadávání úkolů plně pokryt.

Typickou nevýhodou všech placených systémů je, že některé užitečné funkce jako definice závislých či dílčích úkolů jsou zpoplatněny. Hlavní rozdíly pak bývají v možnosti určování priorit a uživatelská rozhraní. Určování priorit většinou probíhá pomocí štítků, nebo řazení v backlogu. Ten je nahraditelný například jedním sloupcem v Kanbanovské tabulce, nicméně se tím ztrácí některé pokročilé funkce backlogu jako jsou definice sprintů a epics, které nabízí například Jira a Azure DevOps.

Zatímco placené systémy většinou nabízí kvalitní UI, Redmine v tomto ohledu poměrně zaostává. Nicméně, ač je na trhu v dnešní době velké množství dalších podobných systémů, Redmine je jediný, takto bohatě funkčně založený systém, který je dostupný plně zdarma.

Kapitola 4

Návrh podpůrné aplikace

Z analýzy vyplývá, že na trhu existuje mnoho systémů pro podporu TM, u kterých je většina základních funkcí podobná. Specifické nástroje systémů jsou přizpůsobeny metodám projektového řízení. Průzkum ve vybraných firmách navíc ukázal, že jsou pro podniky z velké části tyto nástroje dostačující a efektivní. Problémovou oblastí se ukázala být nepřehlednost a nedostatečná informovanost zaměstnanců při sledování stavu, závislostí a priorit úkolů. Při návrhu aplikace tak bude kladen důraz převážně na následující oblasti.

- Podpora sledování aktuálního vytížení týmů.
- Možnost určení prioritních úkolů zaměstnanců.
- Jednoduchost a přístupnost uživatelského rozhraní.

4.1 Cíl projektu

Aplikace bude dle zadání podpůrným nástrojem pracujícím s daty systému Redmine. Přizpůsobena bude procesům všech společností z provedeného průzkumu. Nicméně kromě Ing. Martina Půlpitla nikdo z respondentů neuvedl žádný konkrétní problém při zadávání úloh, který by byl přímo aplikačně řešitelný. Hlavním cílem tak bude nahrazení ručního zapisování prioritních projektů do tabulek ve firmě Ackee a přizpůsobení nástroje pro možné využití v ostatních firmách z průzkumu. Snaha bude umožnit nejjednodušší správu a konfiguraci aplikace, aby byla využitelná okamžitě v reálném provozu pro již probíhajícími projekty.

4.2 Navržené řešení

Pro nahrazení tabulky využívané ve firmě Ackee je nutné poskytnout stejné funkcionality. Z pohledu sledování kapacit je taková tabulka přehledným zdrojem informací. Pokud není možné do určité priority zařadit zaměstnanci další projekt, je to pro zadavatele indikátor nevytíženosti. Přiřazování úkolů členům týmu pro časový úsek je pak použitím upraveného 1-3-5 pravidla, které se ve firmě osvědčilo. Navržené řešení proto bude kopírovat rozvržení Google Sheets tabulky v Ackee. Vzhledem k jejímu orientačnímu účelu by aplikace neměla upravovat data v systému Redmine. Veškeré informace nastavené v rozhraní tak budou uchovávány ve vlastní databázi.

4.2.1 Úprava Ackee tabulky

Z pohledu ostatních firem nemusí být takto nastavená tabulka užitečná. První možný problém může být přiřazování projektů. Zaměstnanci mohou pracovat v určitém časovém úseku pouze na několika málo projektech jako např. ve firmě Qest. Určování priorit pro projekty by tak mohlo být zbytečně kontraproduktivní. Řešení proto bude podporovat i přiřazení tiketů.

Dalším problémem mohou být časové úseky a počet priorit. Přiřazování v týdenním intervalu nemusí být pro všechny firmy efektivní. Obzvlášť, pokud v tabulce zvolí přiřazování úkolů namísto projektů, nemusí být zcela pokryta vytíženost pracovníka na celý týden. Poskytnuta proto bude možná změna časového úseku na dny a měsíce a možnost změnit počet sloupců s prioritami.

Poslední úpravou tabulky bude její členění na týmy. Například ve firmě Digital Ant nerozlišují při tvorbě produktu členy do týmů a nutnost seskupovat zaměstnance do skupin je příliš svazující. Členění do týmů tak bude volitelné. Díky zmíněným úpravám bude možné například definovat tabulku pro firmu s přiřazováním projektů na měsíční bázi a pro týmy pak založit tabulky s definicí aktuálně prioritních úkolů pro jiný interval. Upravená tabulka bude v kontextu aplikace nazývaná tabulí (anglicky „board“) a celý projekt ponese název „TaskBoard“.

4.2.2 Podpůrné nástroje

Přiřazování úloh uživateli bude prováděno pomocí seznamu s dostupných tiketů nebo projektů ze systému Redmine. Pomocí textového vyhledávání pak bude možné dohledat přesnou textaci zadání, které by mohlo být uživateli přiřazeno. Úkoly bude možné později měnit či mazat pro určitý časový úsek.

Pro podporu rychlého zadání, v případě velkého množství zaměstnanců či častého opakování stejných tabulí, budou umožněny funkce pro kopírování a vkládání. Poskytnut bude nástroj pro kopírování aktuálních úkolů jednoho uživatele, které budou moci být vložené uživateli druhému. Dále bude podpořeno kopírování tabulí napříč časovými intervaly. Při vložení budou přepsány

jak jednotlivé úkoly členů, tak nastavení pro aktuální časový interval např. počet priorit.

Reflektování nepřítomnosti zaměstnanců pro přehlednost aktuálně dostupných kapacit v daný čas bude deaktivace uživatele v tabuli. Pro daný časový interval nebude možné deaktivovanému uživateli přiřadit žádný úkol. Pro lepší přehled na tabuli bude umožněna i změna jmen členů tabule.

4.2.3 Využití aplikace

Takto navržená aplikace může být užitečná nejen pro firmu Ackee, ale i pro ostatní firmy. Její primární využití po implementaci v této práci by mělo cílit na společné schůzky v procesu.

Pro firmy pracující v metodice Scrum jako Qest může být aplikace pomůckou při pravidelných stand-up schůzkách, kde se mluví o aktuálních úkolech, a které úkoly byly předešlý den splněny. Obzvlášť v softwarových firmách, kde je hojně využívána práce z domova, může být využita při online schůzkách pro přehled aktuálně zpracovávaných úkolů či projektů.

Pro agentury zaměřené na grafický produkt jako jsou Digital Ant a 2FRESH pak může pomoci zapojení nástroje na schůzkách jako jsou statusy či úvodní plánování. Pomocí sledování aktuálně zpracovávaných úkolů a stanovování priorit může dojít k zamezení lidských chyb při odhadech. Zároveň při společné kontrole zpracovávaných úloh může dojít k odhalení závislostí.

4.3 Požadavky na aplikaci

Pro implementaci je důležité stanovit akceptační kritéria v podobě požadavků na aplikaci. Nefunkční požadavky budou definovány na základě analýzy používaných nástrojů ve vybraných firmách z průzkumu a stanovených cílů pro aplikaci. Z navrženého řešení jsou pak odvozeny požadavky funkční.

4.3.1 Nefunkční požadavky

Aplikace bude přístupná přes webové rozhraní v angličtině.
Webové rozhraní v anglickém jazyce je standardem všech analyzovaných podpůrných systémů, které vybrané firmy používají. Proto bude požadováno i od navrženého řešení z důvodu rychlého a jednoduchého zavedení ve společnostech.

Autentizace bude přístupná přes externí službu.
Využívané podpůrné systémy obsahují buďto vlastní autentizaci nebo je možné využít externích služeb. Pro jednoduchost využívání a konfigurace aplikace by tak měla být také poskytnuta možnost přihlášení přes externí službu.

Aplikace bude kvalitně přístupná z mobilních zařízení.
Standardem moderních webových aplikací je responzivní design nebo poskytnutí verze pro systémy Android či iOS. Přístupnost z chytrých telefonů je umožněna i ve všech firmami používaných systémech.

Jednoduché napojení na existující podpůrné systémy.
Aplikace by měla být tvořena s povědomím o budoucím napojení na další podpůrné systémy. Pro potřeby Ackee a dle zadání práce bude systém napojen na systém Redmine. Napojení by mělo být provedeno co možná nejuniverzálněji.

Systém bude dostupný zdarma s možnými úpravami.
Další placené řešení by pro firmy nemuselo být výhodné. Licence pro volné šíření a úpravu kódu zaručí možnost vlastní implementace autentizace, designu či externího napojení.

Serverová část aplikace bude napsána pomocí Node.js.
Obě softwarové firmy z průzkumu používají pro serverovou část svých webových aplikací systém Node.js [73]. Jedná se o moderní, hojně využívanou technologii. Pro snadné nasazení a možnou snadnou vlastní konfiguraci ve firmách by měla být zvolena právě tato technologie.

4.3.2 Funkční požadavky

Uživatelé

- Aplikace umožní přidání nových uživatelů.
- Uživatel se bude moci do aplikace přihlásit.
- Uživatelé budou rozlišeni do rolí podle možnosti editace tabulek.
- Uživatele bude možné deaktivovat a následně aktivovat.

Týmy

- Aplikace umožní vytvoření týmů.
- Aplikace umožní přiřazení uživatelů k týmům.
- Aplikace umožní nastavení barvy týmu.

Správa tabulek

- Aplikace umožní správu přístupových údajů k externím systémům.
- Aplikace umožní vytvoření tabule pro zvolené přístupové údaje.
- Aplikace umožní přiřazení týmů k tabuli.
- Aplikace umožní přiřazení uživatelů k tabuli (dále členové tabule).
- Aplikace umožní načtení členů tabule z externího systému.

Záznamy v tabulce

- Aplikace umožní ruční řazení týmů a členů na tabuli.
- Aplikace umožní vyhledání úkolů a projektů z externího systému.
- Aplikace umožní přiřazení úkolu nebo projektu členovi tabule.
- Aplikace umožní smazání úkolu člena tabule.
- Aplikace umožní deaktivaci uživatele pro časový interval.
- Aplikace umožní listování v časových intervalech tabule.
- Aplikace umožní kopírování a vkládání tabulí.
- Uživatel bude upozorněn v případě změny úkolu na tabuli.

4.4 Případy užití

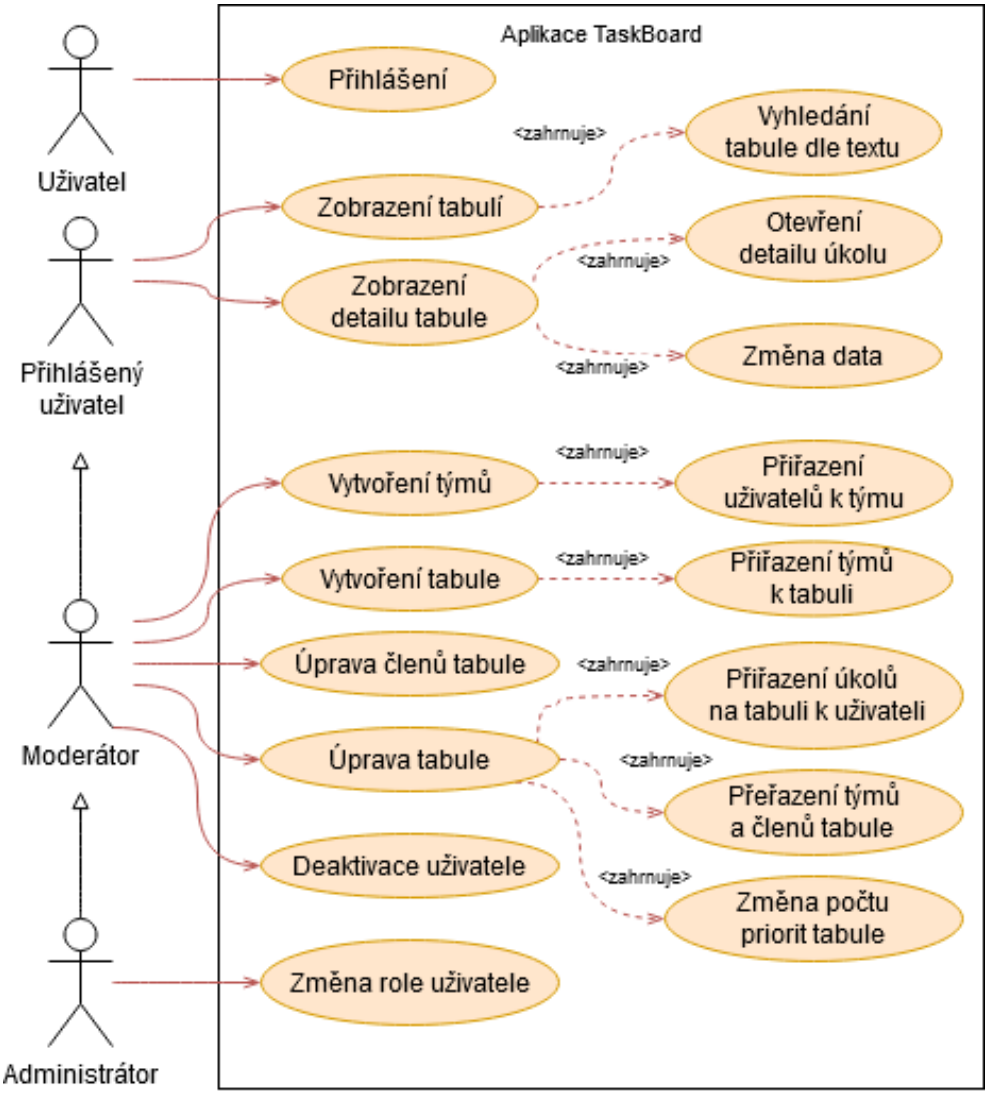
Z navrženého řešení lze také definovat základní případy užití aplikace. Jejich základní přehled vyjadřuje diagram na obrázku 4.1. Aktéry v procesu můžeme rozdělit na editora a běžného uživatele. Editorem se myslí uživatel, který v aplikaci má práva na úpravu tabulí a dalších dat. Tím bude buďto administrátor nebo moderátor.

4.4.1 Uživatel

Běžný uživatel aplikace potřebuje pouze zobrazení informací v aplikaci. Po přihlášení přes externí službu bude uživatel schopen zobrazit seznam existujících týmů a jejich členů. Bude mu umožněno nahlížet do tabulí, kterých je členem. V nich bude schopen vidět seznamy přiřazených úloh či projektů a zobrazit jejich detail v externí podpůrné aplikaci. Nebude mu umožněno žádné informace v aplikaci měnit, či vytvářet.

4.4.2 Moderátor

Moderátor aplikace bude schopen spravovat většinu entit v aplikaci. Po přihlášení bude vytvářet, upravovat a mazat tabule, týmy a jejich členy. Při zobrazení tabule mu bude umožněno upravit a seřadit týmy a členy na tabulce zobrazené a přiřazovat jim úkoly či projekty.



Obrázek 4.1: Diagram případů užití pro globální role.

4.4.3 Administrátor

Administrátor bude mít stejné v aplikaci možnosti jako moderátor. Jeho případy užití se rozšiřují o přidání, úpravu, mazání přístupových údajů a uživatelů v celém systému.

4.4.4 Aktéři pro tabulky

Aplikace dle návrhu umožní definování více tabulek napříč různými týmy. Globální role tak nemusí být dostačující. Systém proto bude odlišovat aktéry na uživatele a editory i pro určité tabulky. Globální pravomoce aktérů pak budou mít přednost před pravomocemi v tabulkách. Rozdělení rolí pro tabulky bude následující.

Člen tabulky nebude schopen provádět změny v tabulce.

Moderátor tabulky bude editor bez možnosti měnit role ostatních členů.

Vlastník tabulky bude editor s možností měnit role ostatních členů.

4.5 Návrh uživatelského rozhraní

Analyzované placené aplikace často poskytovaly zajímavá uživatelská rozhraní, která ovšem nebyla dostupná z menších zařízení a bylo nutné využívat mobilní aplikaci. Při návrhu UI je proto od začátku myšleno na responzivitu jednotlivých prvků a jejich uspořádání na obrazovkách s malým rozlišením. Pro aplikaci bylo autorem práce vytvořeno logo na obrázku 4.2. Návrh UI probíhal pomocí tvoření wireframů v aplikaci Balsamiq [74]. Všechny při návrhu vytvořené wireframy je možné nalézt v příloze E.

Obrázek 4.2: Navržené logo pro aplikaci TaskBoard.

4.5.1 Navigace

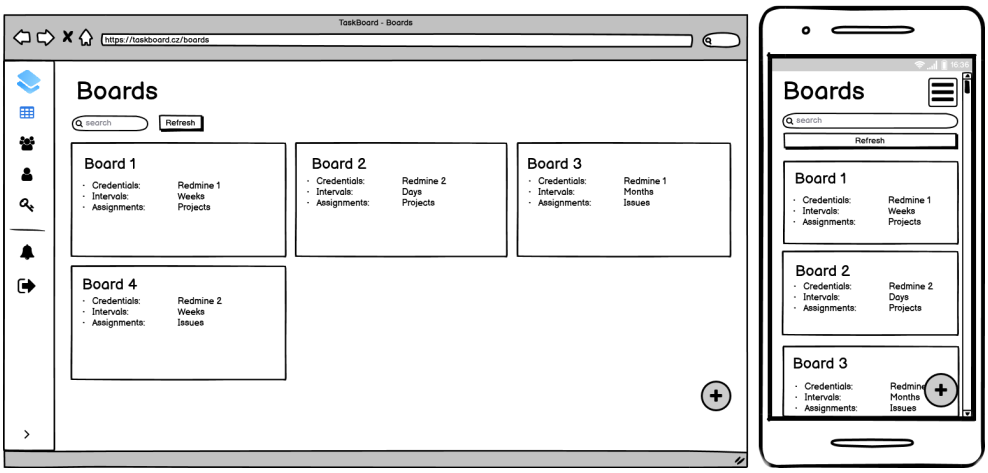
Většina analyzovaných podpůrných systémů obsahuje boční navigaci. Ta je zvolena jako hlavní i pro aplikaci TaskBoard. Horn9 navigace je poskytnuta pro pomocné nástroje při úpravě tabulky. Vzhledem k velkému množství místa, které by navigace mohla na stránce zabírat, jsou zobrazeny jen ikony jednotlivých položek. Uživatel může navigaci pomocí dolního tlačítka rozšířit a vidět tak i názvy akcí. Na malých rozlišeních pak dochází ke kompletnímu skrytí

navigace a zobrazení pouze tlačítka v pravém horním rohu obrazovky, které navigaci zpřístupní. Rozkliknutím je zobrazen plný detail navigace přes celou obrazovku.

4.5.2 Prvky pro výčet entit

Obecnou strukturou pro výčet dat budou tvořit tabulky. Pro přehlednost budou nejdůležitější informace vždy umístěny v levé části. Sloupec umístěný úplně vpravo tabulky pak bude sloužit pro akční tlačítka nad daty jako je smazání či úprava. Responzivita tabulek bude zajištěna přepnutím do bloků. Sloupce budou zobrazeny v řádcích a akční tlačítka zůstanou v pravé části tabulky. U výčtu entit bude zobrazeno akční tlačítko pro přidání v pravém dolním rohu obrazovky dle návodu Meterial Design [75].

Některé entity v aplikaci nemusí obsahovat moc informací, nebo je není nutné při zobrazení seznamu znát. Z toho důvodu bude při zobrazování seznamu přihlašovacích údajů a tabulí přistoupeno rovnou ke zobrazování bloků. V případě tabulí bude blok klikatelný, čímž dojde k přesměrování na detail dané tabulky. Ukázka wireframu pro seznamu tabulí je na obrázku 4.3.



Obrázek 4.3: Wireframy pro seznam tabulí v prohlížeči a na mobilním zařízení.

4.5.3 Zobrazení tabule

Tabule bude obsahovat horní navigaci s názvem tabule, přepínáním aktuálního data a nástroji pro úpravu. Samotná tabule s úkoly pak v každém řádku obsahovat uživatele. V nejlevějším sloupci tabule bude jméno či přezdívka uživatele a jeho profilový obrázek. U něj bude pro editory umístěna ikona s dalšími možnostmi např. pro kompletní kopírování řádku. V dalších sloupcích pak budou přidělené úkoly či projekty pro dané období.

Při třídění uživatelů do týmů budou uživatelé zabarveni do barvy týmu pod kterým jsou přiřazeni a tým bude pomocí kliknutí na jeho název možné skrýt. Změna časového období tabule bude probíhat pomocí selekce ve vyskakovacím okně. Bude vždy přizpůsobena zvolenému intervalu tabule. V případě dní tak bude zobrazen výčet dní, v případě týdnů intervaly 7 dní, a v případě měsíce pak pouze názvy měsíců.

Pohledy jednotlivých rolí se budou lišit počtem akčních tlačítek v horním panelu a také nástroji zobrazenými v řádcích uživatelů. Běžný uživatel u každého úkolu uvidí ikonu oka, která bude indikovat, že při zakliknutí systém přesměruje na detail úkolu. Administrátor pak uvidí buďto znak plus, značící přiřazení nového úkolu, nebo šipku označující zobrazení seznamu dalších úloh, na které lze přiřazení změnit. Editorský pohled je zobrazen na obrázku 4.4

Obrázek 4.4: Wireframy pro editorský detail tabule v prohlížeči a na mobilním zařízení.

4.5.4 Zobrazování dodatečných informací

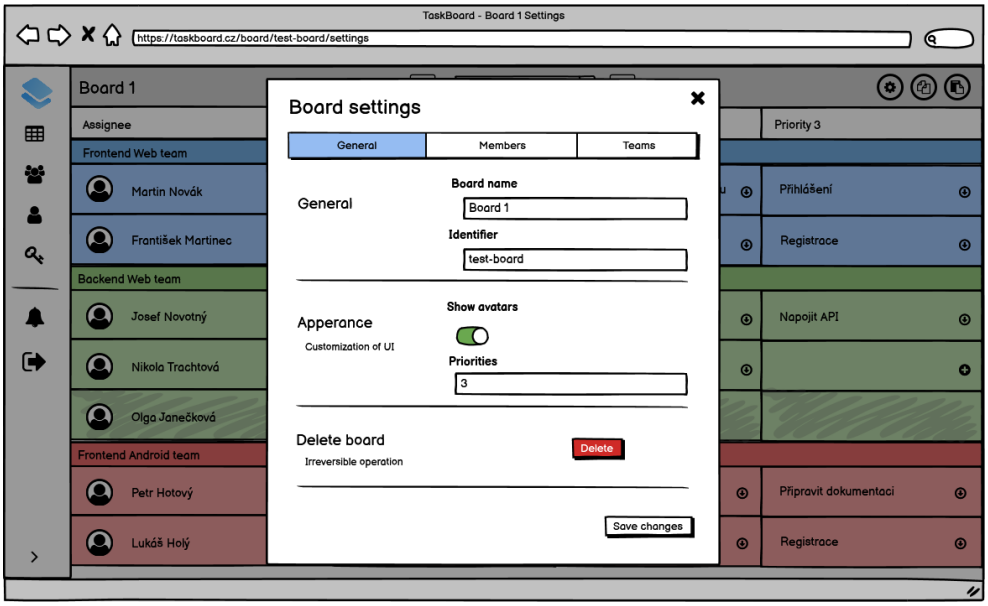
Hlavním problémem analyzovaného uživatelského rozhraní Redmine byla nutnost při každé akci přenačítat stránku. Docházelo tak k neustálému načítání všech informací na stránkách včetně těch, které byly společné i pro stránky předchozí. Z toho důvodu jsou dodatečné informace na daných pohledech pro uživatele, přihlašovací údaje, týmy a tabule vždy zobrazeny ve vyskakovacích oknech nad původním zobrazením. K opětovnému načtení původních seznamů dojde pouze ve chvíli, kdy bude ve vyskakovacím okně vykonána nějaká akce.

4.5.5 Nastavení tabule

Navržené funkce indikují, že tabuli lze výrazně přizpůsobit. Její nastavení proto obsahuje velký počet informací. Zobrazení ve vyskakovacím oknu je rozděleno na kategorie, ke kterým se nastavení vztahuje na:

- obecné nastavení (settings),
- členy tabule (members),
- týmy tabule (teams).

Tento přístup by měl být pro uživatele přehlednější. Vždy je ihned vědět, co z tabule je aktuálně upravováno. Ke každé kategorii je možné připsat další informace o tom, co přesně změna nastavení způsobí. Detail obecného nastavení je zobrazen ve wireframu na obrázku 4.5.



Obrázek 4.5: Wireframe pro nastavení tabule.

Kapitola 5

Implementace aplikace

Webová aplikace lze vždy rozdělit na část spouštěnou v prohlížeči (klient) a část spouštěnou na vzdáleném serveru. Aplikace v této práci cílí na kvalitní a rychlé uživatelské rozhraní. Klientská část tak bude ve výsledku poměrně rozsáhlá. Využívat serverovou technologii pro vykreslování by tak mohlo být pomalé, proto je aplikace TaskBoard rozdělena na dvě oddělené části, které komunikují přes API. Tento přístup přináší například tyto výhody:

- možnost napojení dalších aplikací na serverovou část.
- jednoduchá kompletní výměna klientské části (změna UI, grafiky),
- možnost nasazení částí na rozdílné servery,
- oddělení vykreslování dat a jejich správy,
- po načtení klientské části jsou po síti přenášena pouze data.

Dochází tak k oddělení zodpovědností za proces vykreslování a ukládání dat. Serverová část ukládá data v univerzálním formátu připraveném k využití i pro jiné klientské části či pro externí systémy. Klientská část se pak bude starat o specifické vykreslení dat dle navrženého UI.

5.1 Vybrané technologie

Pro implementaci jsou vybrány moderní technologie pro webové aplikace. Přihlédnuto bylo k faktu, zdali jsou již používané v softwarových firmách z průzkumu, aby pro ně byly snadno v praxi využitelné. Výběr byl také založen na autorovo preferencích, jelikož přednost dostaly technologie, se kterými se chtěl v rámci této práce seznámit.

5.1.1 Klientská část

Na základě oddělení od serverové části a cílení na přívětivé UI je v klientské části implementována jednostránková aplikace (známé jako „Single page appliaction“[17], dále SPA). Mezi moderní populární knihovny pro tvorbu SPA patří Vue, Angular, React, Ember a Backbone [76]. Zvolena je knihovna React [77] z důvodu, že je ze zmiňovaných knihoven aktuálně dle portálu BuiltWith na webu nejpoužívanější [78] a je také využívána firmami Ackee a Qest.

Pro zjednodušení tvorby UI byly uvažovány další knihovny pro práci s vizuální stránkou aplikace. Výběr byl pro jednoduchou integraci zúžen na ty, které již obsahují komponenty pro React. Mezi momentálně nejpopulárnější dle statistik portálu GitHub patří React Bootstrap, Semantic UI React, Blueprint UI, Material-UI a Ant Design [79].

Všechny knihovny by byly v případě využití postačující, jelikož nabízí podporu pro rozložení prvků na stránkách a responzivní design. Přednost byla dána knihovně Ant Design [80]. V porovnání s Material-UI, která je aktuálně nejpoužívanější, poskytuje více komponent a umožňuje jejich okamžité používání bez nutnosti instalace dalších knihoven např. pro zobrazení notifikací, validaci formulářů. Autor této práce Ant Design navíc na rozdíl od Material či Bootstrap dříve nepoužíval.

Na základě volby zmiňovaných technologií pro klientskou část jsou dále přidány následující nástroje a knihovny pro ulehčení implementace.

Create React App (dále CRA) je nástroj pro jednoduché vytváření klientských částí moderních webových aplikací pomocí jednoho příkazu. Vytvořen je balíček knihoven postavených na React, které jsou okamžitě připravené k používání. Pomocí jednoduchých skriptů je možné zařídit jejich připravení pro nasazení na produkci. CRA tak nahrazuje velké množství závislostí pouze jednou. Jádrem je také využívání optimalizačních nástrojů jako ESLint, Webpack a Babel. [81]

Sass je preprocesor CSS. Oproti ostatním preprocesorům je možné jej jednoduše využít při vytvoření aplikace přes CRA. V aplikaci je použit pro centralizaci stylů do oddělených souborů. Umožňuje také využívat rozšířené konstrukty jako jsou cykly, mixiny nebo funkce pro změnu barev. [82]

CRACO je akronymem pro „Create React App Configuration Overwrite“. Jedná se o nástroj pro změnu konfigurace CRA. Pro úpravu nastavení balíčků v CRA je běžně nutné aplikaci rozbalit a spravovat pak závislosti ručně. CRACO umožní změnu bez rozbalování. Nástroj je použit pouze pro úpravu barev v Ant Design a pro možnost v React aplikaci využít i soubory mimo kořenový adresář aplikace. [83]

[17] Aplikace načítá většinu klientského kódu ze serveru při prvním požadavku a mění své UI dynamicky v prohlížeči.

Redux je stavový kontejner pro knihovnu React. Umožňuje oddělit a centralizovat logiku změny stavů komponent do oddělených souborů. Základem je vyvolávaní akcí, které vedou ke změně kontejneru a překreslení naslouchajících komponent. Tím přispívá k možnosti lépe kód strukturovat a testovat. [84]

Redux Thunk je rozšíření Redux pro podporu asynchronních akcí s vyvarováním se nežádoucích efektů [84]. Použit je pro akce, které vyžadují odeslání informací na serverovou část aplikace.

React Router poskytuje sadu navigačních komponent pro řízení aktuálně vykresleného obsahu na stránkách. Součástí je také podpora změny a prasování Uniform Resource Locatoru (dále URL) v prohlížeči, čímž lze simulovat změnu stránky a zpřístupnit tak funkci „zpět“. [85]

React beautiful dnd je React knihovna od firmy Atlassian sloužící k plynulému přesouvání objektů pomocí funkce drag and drop [86]. Využita je pro lepší interakci uživatele s UI aplikace.

5.1.2 Serverová část

Výběr technologie pro serverovou část aplikace byl ovlivněn hlavně nefunkčními požadavky stanovenými při návrhu aplikace. Zvolen tak byl nástroj Node.js s frameworkem Express [87]. Tato kombinace umožňuje jednoduchou a rychlou implementaci webové API v JavaScriptu [88]. Express je navíc i základním stavebním kamenem pro další využívané frameworky postavené na Node.js jako Sails, Kraken či Nest.js [89].

Jako databáze byly uvažovány jak běžné, na SQL založené, systémy tak moderní NoSQL (not only SQL) řešení. Vzhledem k předchozím zvoleným technologiím a povaze ukládaných dat bylo vybráno MongoDB. Databáze ukládá data pomocí dokumentů s volnou strukturou ve formátu JSON. Tímto formátem je ovlivněn i dotazovací jazyk, čímž je databáze jednoduše použitelná v jazyce JavaScript. MongoDB bude využito na oficiálním cloudu. [90]

Pro podporu dalších funkcionalit jsou použity následující knihovny.

Mongoose je knihovna pro práci s MongoDB za pomoci mapování dokumentů do objektů (ODM). Díky ní bude možné definovat striktní schéma databáze pro jednodušší testování a udržitelnost aplikace. [91]

Express validator je sada funkcí pro podporu validace informací z HTTP požadavků (URL a tělo požadavku) prostřednictvím Express middleware. [92]

Passport je knihovna pro umožnění jednoduché autentizace prostřednictvím různých technologií a externích služeb. [93]

Cookie Session je knihovna pro podporu práce se session a cookie v Node.js. Využita bude pro uložení údajů o autentizaci pomocí Passport. [94]

5.1.3 Sdílené technologie

Dříve zmíněná volba klientských a serverových technologií je v komunitě zkráceně označována jako MERN (MongoDB, Express, React, Node.js). Díky ní lze uskutečnit kompletní tok dat pomocí JSON formátu za využití pouze jazyka JavaScript. To otevírá možnost pro využití stejných knihoven v obou částech aplikace. Použity jsou následující.

Axios je HTTP klient pro klientské i Node.js aplikace [95]. Využit bude pro zasílání požadavků na API serverové části a externích systémů.

Moment je knihovna pro usnadnění práce s daty a časy v JavaScriptu [96]. Použita bude z důvodu nativní podpory v Mongoose a Ant Design.

Prettier je nástroj pro formátování JavaScript kódu [97]. Umožňuje definici pravidel, podle kterých se má kód formátovat. Je tak možné udržet stejný styl psaní kódu ve všech částech aplikace bez ruční kontroly.

ESLint je nástroj pro optimalizaci JavaScript kódu [98]. Podle určených pravidel kontroluje závadné či zranitelné části kódu a upozorňuje na ně. Využita upravená konfigurace od firmy Airbnb [99].

Server Sent Events (dále SSE) je technologie pro zasílání zpráv ze serveru na klienta přes HTTP. Využita bude pro zasílání notifikací pro aktuálně přihlášené uživatele.

5.1.4 Externí nástroje

Kromě serverových a klientských technologií jsou využity i externí služby, na které bude aplikace napojena. Bude se jednat převážně o volně dostupné technologie, které budou snadno nahraditelné v reálném provozu vlastními nástroji.

Redmine bude využit pro načítání existujících dat o projektech a úkolech. Aplikace bude napojena přes existující REST API.

SendGrid je služba pro zasílání emailů [100]. Zvolena je vzhledem k možnosti využití zdarma. V aplikaci slouží pro zasílání emailových upozornění.

Google Identity je služba pro identifikaci uživatelů přes jejich účet na Google. Využit je pro autentizaci přes protokol OAuth 2.0. [101]

Amazon S3 je cloudový kontejner pro soubory [102]. Aplikace jej používá pro ukládání dynamických souborů jako např. uživatelských avatarů.

5.2 Databázový model

Oproti tradičnímu SQL přístupu k ukládání dat dle normalizačních forem je v MongoDB umožněno vnoření dokumentů do sebe a ukládání položek do kolekcí (polí). Tyto funkce jsou pak omezeny pouze maximální velikostí jednoho dokumentu, na kterou je potřeba dbát při návrhu [103]. NoSQL přístup také nepracuje s klasickou normalizací dat, je tak běžné data duplikovat v případě, že je to pro dotazy efektivní [104]. Finální schéma databáze je zobrazeno na obrázku 5.1.

Celá aplikace se soustředí na definici a úpravu tabulí. Hlavním dokumentem tak je tabule (*Board*), která v sobě nese referenci na používané přihlašovací údaje do externí aplikace (*Credentials*) či jejího zakladatele. Na tabuli jsou pak navázány dokumenty jako jsou týmy (*Team*) přes dokument *BoardTeam*. Dále jsou k tabuli přiřazeni členové (*Member*), uživatelské úkoly (*BoardItem*), či konkrétní nastavení pro časové intervaly (*BoardSettings*).

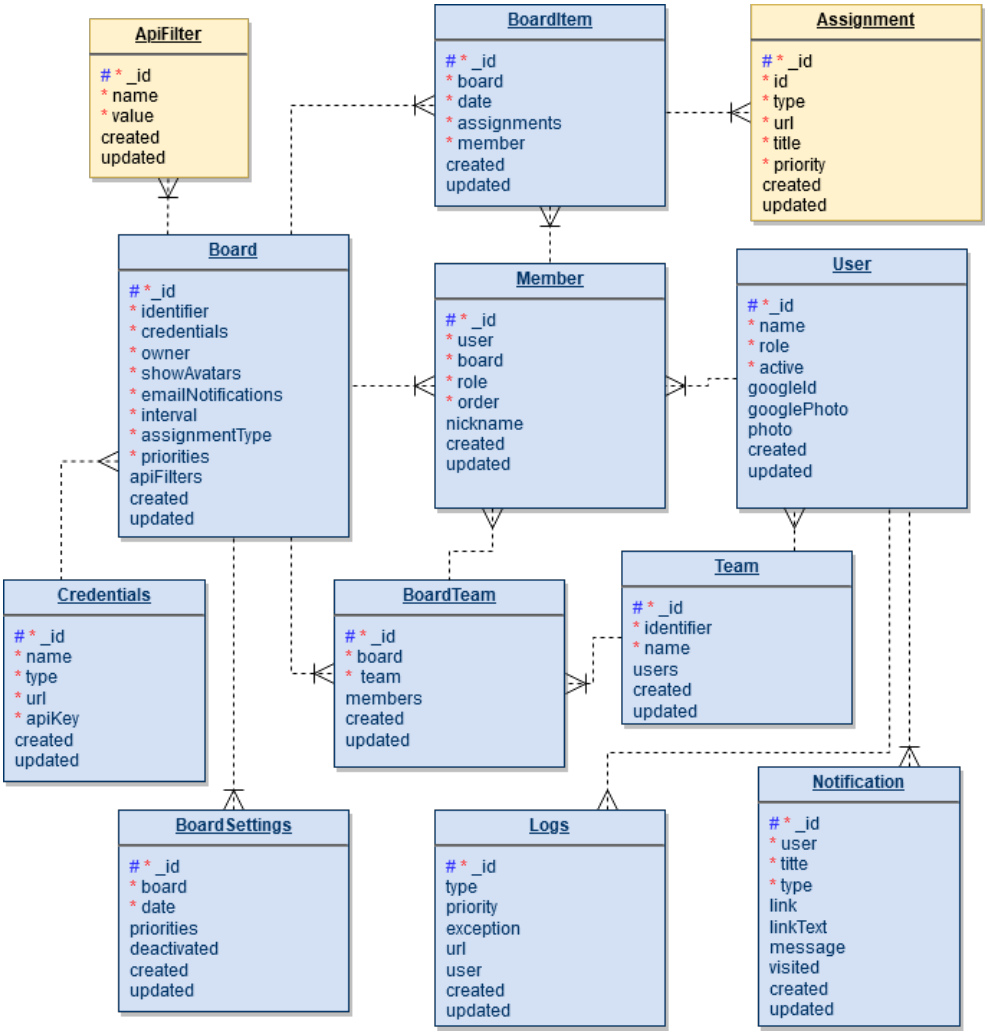
Ve výsledné aplikaci bude často nutné získat seznam týmů tabule a jejich členy. Výpis všech členů týmů je v tomto případě nežádoucí, jelikož tabule bude vždy zobrazovat pouze uživatele, kteří jsou jejími členy, což nemusí být všichni členové přiřazeného týmu. Získávání tohoto výpisu bude nutné při každém načtení tabule, zatímco globální úprava členů týmu nebude tak častá. Z toho důvodu byl využit princip kopírování dat.

Zmíněná entita *BoardTeam* tak obsahuje seznam členů tabule, kteří jsou členové odkazovaného týmu. Vyplývá tím větší náročnost pro úpravu členů týmů v entitě *Team*, kde při úpravě musí dojít ke kontrole všech entit *BoardTeam* a přidání či odstranění příslušných odkazů na dokumenty *Member*. Při dotazování se na týmy tabule, je ale získání jejich členů okamžité.

Dalším řešeným problémem při implementaci bylo ukládání informací o projektech a úkolech. Neustále dotazování na externí API pro získání informací by mohlo být zatěžující a časové nákladné, proto je nutné o nich ukládat i některé informace z Redmine. Duplikování všech dat není pro aplikaci potřebné, jelikož není nutné znát všechny atributy úkolů, které se navíc mohou rychle měnit např. řešitel či stav. Při přiřazení úkolu či projektu jsou proto uloženy pouze základní informace pro výpis na tabuli bez nutnosti dotazu na externí API. Ukládány jsou:

- identifikátory (id),
- názvy (title),
- odkazy na detail do původní aplikace (url).

Aplikace zpětně nekontroluje, zdali došlo ke změnu názvu či odkazu. Toto rozhodnutí bylo učiněno z toho důvodu, že aplikace má primárně sloužit pouze pro aktuální přehled. Čtení historických přiřazení je možné, ale nezaručuje, že v náhledu tabule jsou projekty či úkoly pod správnými názvy Nevalidní přiřazené úkoly je ale vždy možné zpětně přeřadit na aktuální znění.



Obrázek 5.1: Finální schéma databázového modelu. Žlutou barvou jsou vyznačeny vnořené dokumenty.

5.3 Komunikace

Výměna dat mezi klientskou a serverovou částí aplikace probíhá pomocí implementované API serveru a také technologie SSE. Veškeré informace jsou klientem zpracovány do globálního Redux kontejneru, jehož změna pak vyvolá překreslení patřičných komponent. Serverová část aplikace dále komunikuje s externími systémy Redmine, Amazon webovou službou S3 (dále AWS S3) a SendGrid.

5.3.1 REST API

Při implementaci rozhraní serveru bylo myšleno na univerzálnost a možnou znovupoužitelnost pro externí systémy. API proto respektuje základní pravidla Representational state transfer (REST), které definuje Roy Fielding ve své dizertační práci a kterými jsou:

1. bezstavovost,
2. možnost uložení v mezipaměti,
3. nezávislost klienta a serveru,
4. umožnění vícevrstvého systému a
5. univerzální rozhraní. [105]

Využitím HTTP je většina bodů splněna, jelikož se jedná o bezstavový protokol typicky pro komunikaci mezi prohlížečem a serverem. Protokol také umožňuje cachování díky specifickým hlavičkám v požadavcích jako „Cache-Control“, „Last-Modified“ či „ETag“ [106]. Aplikace využívá hlavičky ETag, která je automaticky doplněna frameworkem Express.

Univerzálním rozhraním jsou definice cílových zdrojů, formát reprezentace zdroje, jednotné návratové stavové kódy a užité HTTP metody. Pro reprezentaci dat byl zvolen formát JSON. Zdroje jsou identifikovány jedinečnými URL a odpovídají nejčastěji entitám z databáze či jejich spojení. Například pro entitu tabule je vytvořen zdroj `/boards`, ke kterému je možné přistoupit následovně.

`GET /boards` vrátí seznam všech tabulí.

`POST /boards` vytvoří novou tabuli a vrátí její reprezentaci.

`GET /boards/<id>` vrátí tabuli se zadaným parametrem id.

`PATCH /boards/<id>` upraví atributy tabule a vrátí upravenou reprezentaci.

`DELETE /boards/<id>` odstraní tabuli se zadaným id a vrátí ji.

Pro zdroje navracející kolekce entit, které mohou nabývat velkého množství prvků (např. pro výpis všech tabulí), je implementováno stránkování. Pomocí query parametrů u URL je tak možné řídit aktuální stránku (offset) a počet navrácených prvků (limit). Pokud parametry nejsou specifikovány, je výsledek omezen na přednastavenou hodnotu. Výsledný zdroj je pak obalen v JSON objektu pro stránkování, který obsahuje tyto parametry a výsledek v atributu „data“. Pro přehlednost je také navracen počet všech prvků v kolekci (count). Vizuální ukázka stránkování je zobrazena v kódu 5.1.

Pro usnadnění přístupu přes URL adresy je pro entity týmů a tabulí implementována možnost definovat vlastní identifikátor. Slouží pro dotazování na API tam, kde by jinde byl automaticky vygenerovaný umělý identifikátor z databáze. Např. pokud uživatel definuje u tabule identifikátor `my-board`, může pak k tabuli přistoupit přes URL `/board/my-board`.

Jelikož aplikace primárně cílí na využití společně s klientskou částí a jednotlivé zdroje spolu nemusí být přímo provázány, byl z implementace vynechán jeden z užívaných principů HATEOAS[18]. Kompletní seznam zdrojů API je uveden v dokumentaci u zdrojového kódu aplikace.

```
{
    "data": [
        {
            "identifier": "board-one",
            "name": "Board One",
            ...
        },
        {
            "identifier": "board-two",
            "name": "Board Two",
            ...
        },
    ],
    "limit": 2,
    "offset": 0,
    "count": 12
}
```

Kód 5.1: Ukázka stránkování v reprezentaci API zdroje kolekce tabulí.

5.3.2 Upozornění pomocí SSE a SendGrid

Pro SSE byl vytvořen serverový zdroj `/api/notifications/stream`. Klientem odeslaný požadavek metodou `GET` na tuto adresu je uložen do paměti. Pokud dojde ke změně dat týkající se daného uživatele, je zpět odesláno upozornění. Využitím technologie SSE není spojení po získání odpovědi ukončeno, ale klient čeká na odpovědi dál. Zaslané zprávy jsou ve formátu JSON, ve kterém kořenový objekt obsahuje dva následující atributy.

type – Identifikuje typ dat a odpovídá definovaným zdrojům v API.

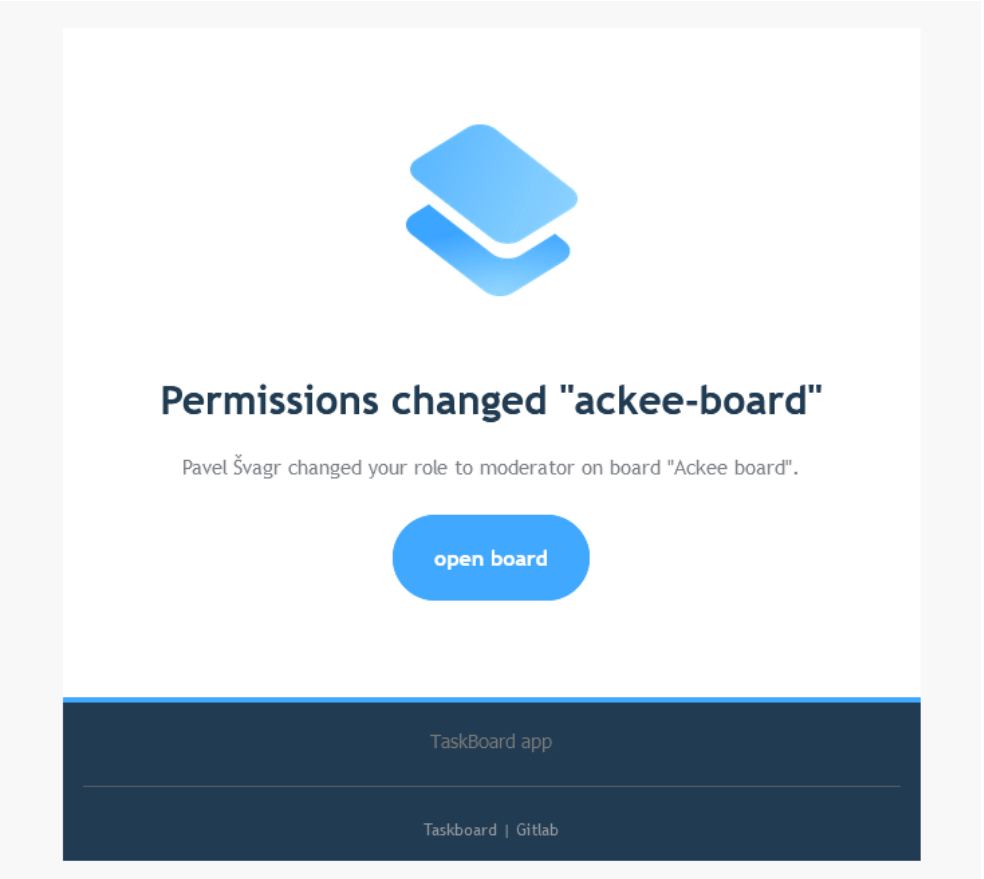
data – Samotná data zaslaná ze serveru.

[18] V překladu „hypermedia jako aplikační stav.“ slouží k průchodu API bez nutnosti předem znát cílové zdroje [107].

Díky takto nadefinované struktuře je možné zaslané objekty na klientské části identifikovat stejným způsobem jako výsledky požadavků na API serveru a stejně k nim i přistupovat. Mohou tak být využity stejné metody pro zpracování, na základě kterých aplikace překresluje UI. Uživatel je tak ihned upozorněn vyskakovacím oknem v pravém rohu obrazovky, když:

- je mu přiřazen nebo změněn úkol,
- je mu změněna role,
- byl přidán k nové tabuli.

Události jsou taktéž ukládány do seznamu uživatelových upozornění a zasílány do emailů prostřednictvím služby SendGrid. Ukázka takové emailové notifikace je na obrázku 5.2. Pomocé tlačítka je vždy možné se ihned přesměrovat na určitou stránku, kde byla notifikace vyvolána. V nastavení tabule je pak možné zasílání do emailu vypnout.



Obrázek 5.2: Ukázka zaslaného emailového upozornění aplikace TaskBoard.

5.3.3 Nahrání uživatelů z Redmine

Pro jednoduché prvotní nastavení projektu je možné při založení tabule zvolit možnost pro nahrání existujících uživatelů ze systému Redmine. Po založení tabule je dotázán seznam uživatelů z Redmine API. V tomto seznamu je hledán email postupně v polích „email“ a „login“. Pokud pod tímto emailem v systému již existuje uživatel, je přiřazen k aktuální tabuli jako člen. Pokud neexistuje, je nejdříve vytvořen s tímto emailem a následně přiřazen jako člen.

Aktuální omezení této funkcionality je nutnost mít zaregistrovaný email v Redmine i pod Google účtem, ze kterého se do aplikace uživatel přihlašuje. Uživatelský email je pak jedině možné změnit v nastavení uživatelů administrátorem. Toto omezení lze do budoucna změnit implementováním dalších možných přihlášení do aplikace.

5.4 Zabezpečení a přihlášení

Přihlášení do aplikace je zabezpečeno pomocí knihovny Passport. Po získání identifikátoru uživatele ze služby Google Identity je upraven jeho profil v aplikaci. Při každém dalším požadavku pak Passport automaticky kontroluje cookie soubory, aby zjistil, zdali je uživatel přihlášen. Při přihlášení přes Google uživatel uděluje svolení s použitím profilových informací, jejíž součástí je i profilový obrázek. Ten je ovšem externím aplikacím přístupný jen po dobu, dokud nevyprší platnost získaného tokenu. Proto v případě, že došlo ke změně URL profilového obrázku do Google, je z ní stažen a nahrán na službu AWS S3 pomocí bezpečnostních klíčů v konfiguraci aplikace, aby byl v aplikaci dostupný napořád.

Obě části aplikace zároveň využívají společný seznam dostupných rolí pro kontrolu přístupů. Klientská část podle aktuální role uživatele skrývá určité prvky na stránce, serverová část pak při vykonávání požadavků kontroluje, zdali má uživatel na dané akce práva.

Serverová část také ukládá přihlašovací údaje k externím aplikacím. Z důvodu bezpečnosti není, kromě prvotního vytvoření údajů, klíč v klientské části aplikace k dispozici. Aby nebyl v případě útoku přímo dostupný z databáze, je před uložením šifrován pomocí algoritmu Advanced Encryption Standard (AES) s klíčem určeným konfigurací aplikace. S dešifrovaným klíčem pak pracuje pouze serverová část aplikace, která pomocí něj získává informace z Redmine o aktuálních úkolech a projektech.

Uživatelem vyplněná data do aplikace jsou validována na straně uživatele pomocí mechanismů v Ant Design, které využívají balíček async-validator [108]. Aplikace tak ukazuje chyby ve vyplněných formulářích okamžitě bez odeslání na server. Klientská část validace nicméně může být případným útočníkem snadno vypnuta. Z toho důvodu jsou všechna data validována i na straně serveru pomocí knihovny Express validator.

5.5 Struktura aplikace

Projekt je rozdělen na dvě hlavní části ve složkách `client` (klientská část) a `server` (serverová část). Klientská část aplikace kopíruje klasický projekt vytvořený za pomoci CRA. Komponenty jsou rozděleny do složek dle jednotlivých pohledů na UI. Například veškeré prvky zobrazené při načtení seznamu tabulí jsou ve složce `boards`. Tento přístup umožní jednoduché implementování nových pohledů. Obecné a sdílené komponenty se nachází ve složce `common`. Globální styly jsou umístěny ve složce `styles` a styly týkající se konkrétních komponent jsou pak umístěny ve složkách s komponentami.

Serverová část kopíruje MVC strukturu s absencí pohledové vrstvy (view), kterou tvoří část klientská. Framework Express poskytuje možnost pracovat s požadavky řetězově. Přes samotným zpracováním tak mohou být s požadavkem vykonány další akce prostřednictvím definovaných middleware. V aplikaci jsou využívány pro všechno zmíněné zabezpečení. Zpracování HTTP požadavku aplikací je v následujícím pořadí:

1. kontrola autentizace,
2. kontrola autorizace
3. validace dat,
4. kontrola, zdali existují v databázi objekty s „id“ uvedeným v URL,
5. zpracování požadavku kontrolery a
6. odchycení případných chyb.

Pokud middleware kontrolující existenci dat objekt najde, uloží jej a předává dál. Následující služby tak již pracují s existující entitou a nemusí se znovu dotazovat na objekt do databáze.

Odchytávání chyb probíhá v poslední fázi zpracování požadavku. Pokud některý z předchozích middleware nalezne chybu, jsou ostatní middleware kontroly a zpracování přeskočeny. Při odchycení chyby, která není vyvolána napsanými JavasScriptovými výjimkami aplikace, dojde k jejímu uložení do databáze pro možný budoucí přehled.

Na základě typu chyby je rozhodnuto, jaký HTTP kód je zaslán zpět klientské části aplikace, která rozhoduje o zobrazení chyby. V případě, že chyba nastala při odeslání formuláře a byly navráceny kódy 404 (špatný požadavek) nebo 422 (nezpracovatelná entita), jsou chyby v klientské části předány aktivním formulářům, které chyby zobrazují u příslušných formulářových prvků.

Pro jednoduchou správu logiky v modelové vrstvě aplikace byl implementován vzor Respository nad namapovanými modely z knihovny Mongoose. Modely jsou nahrány do entitních tříd, které jsou rozšiřitelné o další logiku.

O kompletní úpravu entit v databázi a dotazování pak starají potomci třídy `MongooseRepository`.

Jelikož klientská a serverová část by měly být vzájemně nezávislé, bylo nutné vyřešit umístění sdíleného kódu. Byl proto vytvořen další balíček funkcí ve složce `shared`. Společný kód se tak neduplikuje a je možné ho připojit k oběma vrstvám v případě nasazení na různé servery. Obě části aplikace tak využívají například stejné konstanty a funkce pro chybové stavy, role uživatelů a členů tabulí nebo práci s časem.

Kapitola 6

Nasazení a testování

Součástí této práce je také otestování aplikace v provozu. Za tímto účelem je vyžadováno její nasazení na nějaký server. Z aktuálně nabízených řešení byl vybrán server Heroku. Jedná se o cloudovou platformu, na které lze spustit aplikace napsané ve více jazycích, s možností definice vlastních proměnných prostředí či spouštěním vlastních skriptů. [109]

6.1 Testování v aplikaci

Pro eliminaci chyb před samotným nasazením a uživatelským testováním jsou pro aplikaci napsány lokální testy. Pro možné oddělení do více instancí jsou vytvořeny testy pro všechny implementované části zvlášť (serverová, klientská, sdílená).

Serverová část aplikace je tvořena middleware architekturou a kontrolery, které vždy pracují s požadavky vytvořenými knihovnou Express, proto je pro testování vhodné využít integračních testů. Vybranými nástroji pro testování na serverové a sdílené části jsou Mocha [110] a Chai [111]. Tato kombinace dovoluje testování celé aplikace v podobě simulace HTTP požadavků.

Aplikace je tak testována sadou na sebe navazujících požadavků na cílovou API. Testy kontrolují, zdali je daný zdroj dostupný a zdali jsou zpět vždy navráceny správná data. Pro kontrolu autentizace a autorizace jsou dostupné URL adresy podrobeny přístupu z různých uživatelských rolí a odpovědi jsou podrobeny kontrole návratového stavového kódu.

Pro testování klientské části byl využit nástroj Jest [112], který je přímo součástí CRA. Jedná se o nástroj společnosti Facebook, který poskytuje kromě běžného testování také funkce pro testování React komponent. V aplikaci jsou implementovány jednotkové testy pro pomocné nástroje, dále smoke testy[19] pro hlavní komponenty aplikace a na závěr i jejich snapshot testy. Snapshot testování je moderní technika testů, která spočívá ve vykreslení komponenty

[19]Testy pro ověření, zdali se komponenta vykreslí bez chyby v aplikaci.

do souboru (snapshot). Při dalším spuštění testů se kontroluje, zda je vykreslená komponenta stejná jako dříve vytvořený snapshot.

6.2 Nasazení na Heroku

Vytvořená aplikace obsahuje na serverové části z důvodu napojení na externí systémy několik přístupových údajů. Ty jsou centralizovány do souborů pro vývoj a produkci. Produkční nastavení je bráno z proměnných prostředí, zatímco při vývoji je nastavení vyplněno přímo do konfiguračního souboru.

Pro jednoduché nasazení na Heroku byla klientská i serverová část uvažována jako celek. Aplikace tak funguje společně se sdíleným kódem jako jedna instance, která se připojuje na externí služby pro Redmine instanci firmy Ackee, MongoDB, AWS S3, SendGrid a Google Identity.

Pro testovací účely byla vytvořena také zkušební sada dat. Tu lze jednoduše spustit příkazem, který vygeneruje přístupové údaje do externí aplikace, profil administrátora dle konfiguračních souborů a několik testovacích týmů a tabulek s imaginárními uživateli. Testovací sada dat využívá avatary vytvořené pomocí aplikace Avataaars generator [113].

6.3 Testování v provozu

Testování v reálném provozu ve firmách za dohledu autora práce nebylo kvůli pandemii koronaviru možné uskutečnit. Testování na nasazené instanci aplikace tak proběhlo ve dvou fázích. V první fázi byla aplikace prezentována Ing. Martinu Půlpitlovi pro eliminaci základních chyb, které byly spojeny s návrhem řešení. V druhé fázi byla aplikace otestována uživateli za dohledu autora práce online a zaslána k otestování do firmy Ackee.

6.3.1 Prezentace

Prezentace s Ing. Martinem Půlpitlem probíhala online. Prezentovány byly všechny aktuálně dostupné funkce aplikace. Oceněn byl design aplikace a splnění požadavků z pohledu firmy Ackee. Uvedeny byly dva hlavní problémy při možném reálném použití.

Prvním nedostatečným řešením byl způsob zadávání úkolů, který se zdál vedoucímu práce pomalý. Otevřené okno se seznamem projektů z Redmine vyžadovalo několik kliknutí pro přiřazení projektu uživateli, což by v případě vyplňování pro desítky zaměstnanců nemuselo být efektivní. Z toho důvodu byla do uživatelského rozhraní implementována možnost zapnout okamžitou úpravu[20], kde se po kliknutí do okna v tabuli zobrazí vyhledávací textové pole a uživatel pouze napíše název úkolu a vybere ze seznamu. Pomocí tabulátoru pak lze rychle přepnout do okna dalšího. Funkci je možné vypnout v nastavení

[20] V aplikaci TaskBoard nazýváno jako „inline edit“.

tabule. V případě existence více členů tabule byla pro podporu menší velikosti řádků přidána i možnost skrýt avatary uživatelů.

Druhou požadovanou funkcí bylo přidání globálního filtrování všech úkolů. Redmine totiž automaticky přes svou API vrací v seznamu projektů i dílčí a uzavřené projekty, které v Ackee zaměstnancům nepřiřazují a není tak pro ně nutné je v seznamu vidět. Redmine bohužel neumožňuje přes API získání seznamu stavů projektů. Není tak možné získat identifikátory stavů a filtrovat je uživatelsky přívětivým způsobem. Do nastavení tabule proto byla přidána možnost definování query parametrů pro všechny požadavky zaslané na Redmine (dokument databáze *ApiFilter*). Nevýhoda tohoto řešení je nutnost dohledání v dokumentaci Redmine, jak dané parametry nastavit. Nicméně řešení je univerzální, jelikož je možné nastavit jakýkoliv filtr, který je aktuálně libovolnou externí API podporován.

6.3.2 Uživatelské testování

Po opravení chyb z prezentace bylo provedeno uživatelské testování pro získání informací o uživatelské přívětivosti a spokojenosti s produktem i v jiných než vybraných firmách. Vybráni byli studenti a podnikatelé z autorova okolí, kteří buďto aktuálně vedou nějaký tým, zadávají úkoly, nebo studují zaměření spojené s projektovým řízením.

Účastník 1: Bc. Lukáš Lojík, 24 let, student magisterského programu Informační management na FIT ČVUT v Praze, Projektový manažer ve společnosti Rox s. r. o.

Účastník 2: Bc. Petra Milcová, 23 let, studentka magisterského programu na VŠE v Praze.

Účastník 3: Marek Hypš, 27 let, Associate director ve společnosti Ipsos s. r. o.

6.3.2.1 Scénáře pro testování

Testování aplikace se soustředilo na základní funkcionality. Nejdříve byla testována prázdná aplikace s předvyplněnými daty pro přihlašovací údaje do Redmine. Na splnění úkolů měli účastníci 20 minut. Cílem bylo zjistit, zdali je uživatelské rozhraní dostatečně pochopitelné pro založení nového projektu a okamžité používání. Testovací scénáře byly následující.

Založení tabule a vyzkoušení nastavení.
Přihlaste se do aplikace a vytvořte novou tabuli pro přiřazování projektů pro jednotlivé dny se čtyřmi prioritami. Tabuli nastavte, aby využívala emailové notifikace a přiřazování úloh fungovalo ve zjednodušeném „inline režimu“.

Vytvoření uživatelů, týmů a přiřazení k tabuli.
Pro aplikaci vytvořte alespoň 3 nové uživatele. Poté založte dva nové týmy a uživatele k nim přiřaďte. Výsledné týmy a uživatele pak přidejte k tabuli.

Úprava tabule, přiřazení úkolů.
Změňte pořadí uživatelů a týmů na tabuli. Zkuste u tabule pro datum 18. 12. 2020 deaktivovat některého z uživatelů a změnit počet sloupců tabule na 3.

Pokročilé funkce tabule.
Zkopírujte vytvořené úkoly a vložte je jinému uživateli. Poté zkopírujte všechny úkoly tabule a vložte je do tabule pro následující den. Vypněte pro tabuli inline režim a zkuste znovu přidělit několik úkolů. Na konec změňte přezdívky uživatelů.

V druhé fázi byla testována aplikace s připravenými testovacími daty. Toto testování účastníkům ukázalo, jak by výsledná aplikace mohla vypadat v reálném provozu s různými nastaveními tabule. Účastníkům bylo pokynuto, aby si vyzkoušeli funkce všech vytvořených tabulí. Poté jim byla vysvětlena technická stránka aplikace, odkud se berou data pro přiřazení úloh a vysvětleny funkce spojené upozorněními. Na konec byli účastníci dotázáni na následující čtyři otázky.

1. Jaké jsou Vaše dojmy z testované aplikace?
2. Pro koho si myslíte, že je aplikace vhodná?
3. Uvažoval/a byste o využití aplikace ve Vaší firmě?
4. Co bylo největší překážkou při používání aplikace a co byste změnil/a?

6.3.2.2 Výsledky a úpravy aplikace

Všichni účastníci úspěšně a poměrně rychle všechny scénáře vyřešili. Jejich odpovědi na otázky po testování byly následující.

Účastník 1

Otázka 1: „Dojmy jsou dobré. Líbil se mi vzhled a UI aplikace.“

Otázka 2: „Asi je vhodná pro kohokoliv, kdo využívá Redmine pro zadávání práce nějakým lidem a chce o tom mít přehled.“

Otázka 3: „Určitě ano, ale potřebovali bychom ho napojit na některý z našich systémů. Obecně ale máme úkoly jasně ve firmě dané, takže bychom to využívali spíš jen jako takový přehled, když máme práce hodně.“

Otázka 4: „Ze začátku mi trvalo pochopit, jak se tabule nastavuje, ale pak už to bylo jednoduché. Určitě bych přesunul změnu přezdívek členů z nastavení tabule někam přímo na tabuli."

Účastník 2

Otázka 1: „Na první pohled mě zaujal vzhled aplikace, který jí navíc činí přehlednou. Používání aplikace bylo snadné a intuitivní."

Otázka 2: „Aplikace se mi jeví jako vhodná téměř pro jakýkoliv tým. Myslím si, že by mohla být i vhodnou alternativou pro spoustu start-upů, které hledají nástroj na určení priorit u úkolů."

Otázka 3: Vynechána, účastník nezadává práci v žádné firmě.

Otázka 4: „Přišlo mi nepřehledné vybírání priorit v okně. Při kliknutí se přeházely priority vybraných úkolů, ale nevěděla jsem, která je ta aktuální."

Účastník 3

Otázka 1: „Aplikace je jednoduchá na použití. Líbí se mi rozložení prvků na stránce – umístění menu, rozložení prvků u tabule. Hned mi bylo jasné, k čemu jednotlivé akce slouží."

Otázka 2: „Hodí se pro každého, kdo přiděluje úlohy v týmu a potřebuje nad tím mít přehled."

Otázka 3: „Myslím si, že klidně ano. Aplikace se hodí, když si lidi neumí vybrat vhodné priority pro jim přidělené úkoly. U nás ve firmě se to občas stává skoro každému, takže kdyby se to napojilo na Jiru, kterou používáme, tak proč ne."

Otázka 4: „Některé vedlejší funkcionality jsem nemohl najít. Například změnu přezdívek členů. U změny priorit jsem nepochopil, že se váže jen pro daný datum. Umístil bych nastavení někam přímo k datu."

Z odpovědí a při sledování počínání uživatelů vyplynuly čtyři úpravy do uživatelského rozhraní. První bylo řazení uživatelů a týmů. To v aplikaci probíhalo pomocí přetažení prvku na požadovanou pozici na tabuli. Všichni účastníci sice na funkci přišli poměrně rychle, ale nejdříve vždy hledali nějaké tlačítko pro zpřístupnění této funkce. Na mobilních zařízeních navíc tato funkce nebyla kvůli skrývání týmů při kliknutí umožněna. Tento problém byl vyřešen přidáním řadícího módu, který je zapnutelný pomocí tlačítka u nástrojů tabule, a který na tabuli zpřístupní tlačítka u týmů a uživatelů, pomocí kterých je lze přesouvat.

Druhým problémem byla změna úloh jednoho uživatele ve vyskakovacím oknu. Při vybrání již zvoleného úkolu docházelo k prohození jeho priority s tou aktuální. Aktuálně zvolený úkol ale neměl ukázanou prioritu, pod kterou

je vybrán, a nebyl barevně od ostatních zvolených úkolů odlišen. Z toho důvodu byl do okna přidán text označující aktuálně volenou prioritu a změněna barevná odlišení pro vybrané úkoly.

Třetí úpravou byla změna počtu priorit (sloupců tabule) pro některý interval. Účastníkům nebylo jasné, že se vztahují k aktuálnímu datu. Dva účastníci tak ihned hledali v nastavení tabule, kde je ovšem nastavení pro globální priority tabule. Jeden z účastníků zmínil, že by bylo dobré tuto funkcionalitu přímo popsat textem pod datem, aby bylo jasné, že se k ní vztahuje. Selekce tak byla přesunuta pod tlačítko k výběru data.

Poslední úpravou byla změna přezdívek. Dva účastníci měli změnu přezdívek v nastavení problém najít. Tato funkcionalita byla přesunuta do seznamu nástrojů ke každému uživateli přímo na tabuli.

6.3.2.3 Vyhodnocení

Všechny poznatky z uživatelského testování umožnily vylepšit přehlednost aplikace. Z odpovědí lze soudit spokojenost s přívětivostí uživatelského rozhraní a poskytovanými funkcemi. Všichni účastníci potvrdili univerzálnost aplikace pro libovolný tým. Navíc dva z účastníků, kteří aktuálně zadávají práci ve svých firmách, by si dokázali i představit využití této aplikace přímo u nich v týmu. Pro vyvozování závěrů ohledně užitečnosti je nicméně nutné aplikaci testovat dlouhodobě na reálných datech.

Aplikace byla následně zaslána i vedoucímu práce jakožto zástupci firmy Ackee. Ten ji otestoval a znovu potvrdil, že všechny funkce odpovídají požadavkům firmy a ocenil i další nástroje, které slouží pro přizpůsobení práce s tabulemi. Neměl žádné další výtky k funkčnosti. Spokojený byl také s rychlostí aplikace.

6.4 Možná budoucí rozšíření

Aplikace nabízí spoustu prostoru pro rozšíření. Primárně je možné napojení na další systémy jako jsou Asana, Jira či Notion. Nabízí se také možná úprava uživatelských profilů (nahrání vlastního avataru a další funkce). Díky univerzálnímu ukládání dat je také možné přidání dalších pohledů na přiřazené úkoly kromě existujícího tabulkového pohledu. Implementovat se mohou také jiné funkce z analyzovaných podpůrných systémů, které jsou běžně placené jako např. definice závislých úkolů.

Po zpřístupnění aplikace veřejnosti a průzkumu, zdali je její využití efektivní i pro další firmy, je možné dodělat funkce pro definici úkolů samotných a odstranit tak nutnost napojení aplikace na externí systém. Aplikace by poté mohla v malých firmách fungovat podobně jako např. jednoduchý nástroj Trello.

Závěr

Cílem diplomové práce bylo provést analýzu procesu zadávání úloh v malé až středně velké firmě. Na jejím základě pak navrhnout, implementovat a otestovat aplikaci komunikující se systémem Redmine přes jeho API.

V teoretické části práce byly nejdříve popsány základní aktivity při procesu zadávání a správy úloh v různě projektově řízených týmech. Uvedeny byly také možná rizika a používané nástroje při odhadování termínů a kapacit, určování prioritních úloh a jejich delegaci a distribuci. Dále byly komentovány důležité faktory, které proces ve firmě ovlivňují.

Ze získaných informací byla vytvořena rámcová témata pro rozhovory ve čtyřech vybraných agenturách. V nich byli respondenti dotazováni na zadávání úloh, určování priorit úkolů, plánování a problémy, které při procesech nastávají. Z průzkumu vyplynulo, že potřeby firem jsou různé a často přímo souvisí se zaměřením výkonné činnosti, počtem zaměstnanců a klientele. Byly proto určeny společné aktivity při správě úkolů a vybrány nejčastěji zmiňované problémové faktory, kterými se ukázaly být lidské chyby, funkce podpůrných systémů, určování priorit úkolů a jejich závislostí.

Následně byly zhodnoceny firmami užívané podpůrné systémy, jejich uživatelské rozhraní a nástroje, pomocí kterých v nich lze vytvářet, upravovat a delegovat úlohy. Z poznatků bylo vyhodnoceno, že nová aplikace by měla být hlavně uživatelsky přívětivá, volně dostupná a poskytovat konkrétní řešení nějakého problému. Aktuální systémy totiž sice často nabízí spoustu funkcí, které by jistě proces zefektivnily, ale bývají placené a firmám se nemusí vyplatit.

V praktické části práce byl proveden návrh nové aplikace, která primárně řeší určování prioritních projektů zaměstnanců ve firmě Ackee. Při návrhu funkčních požadavků a případů užití bylo myšleno i na další podniky a jimi zmíněné problémové oblasti. Navrženy tak byly funkce podporující různé rozložení pracovních týmů, sledování dostupných kapacit a přizpůsobení řešení pro potřeby různých projektů.

Aplikace byla implementována pomocí moderních technologií s napojením

na API systému Redmine, která je využívána pouze pro získání aktuálních seznamů úkolů a projektů. Výsledná aplikace poskytuje rychlou konfiguraci a několik funkcí pro přizpůsobení libovolnému pracovnímu týmu. Řešení bylo nasazeno na server Heroku a otestováno. Při testování se prokázala uživatelská přívětivost implementovaného řešení a možná využitelnost v různých firmách. Na základě získaných poznatků byly vylepšeny funkce aplikace a jejího uživatelského rozhraní.

Do budoucna aplikace nabízí mnoho prostoru pro rozšíření. Je možné ji napojit na další podpůrné systémy či přidat nové funkce spojené s definicí a správou úkolů. V případě zájmu firem a podnikatelů je možné projekt dále vyvíjet pro podporu zadávání úkolů a řešení problémů i v jiných než vybraných firmách v této práci.

Literatura

[1] Riss, U.; Rickayzen, A.; Maus, H.; aj.: Challenges for business process and task management. *Journal of Universal Knowledge Management*, Leden 2005: str. 77.

[2] Entrepreneur: 5 Project Management Mistakes That Can Harm Your Business. *Entrepreneur Europe*, Srpen 2017, [cit. 2020-09-03]. Dostupné z: `https://www.entrepreneur.com/article/299000`

[3] Duke, R.: What is a Work Breakdown Structure? [online], [cit. 2020-09-15]. Dostupné z: `https://www.workbreakdownstructure.com`

[4] Luijbregts, E.: A Project Manager's Guide To Work Breakdown Structure. *The Digital Project Manager*, Duben 2020, [cit. 2020-09-10]. Dostupné z: `https://thedigitalprojectmanager.com/work-breakdown-structure`

[5] Puranam, P.: Dividing Team Tasks: Is There a Better Way? *INSEAD KNOWLEDGE*, Květen 2020, [cit. 2020-10-01]. Dostupné z: `https://knowledge.insead.edu/leadership-organisations/dividing-team-tasks-is-there-a-better-way-4008`

[6] O'Reilly Media: A Guide to the Project Management Body of Knowledge (PMBOK® Guide), 6.2 Define Activities. [online], Leden 2013, [cit. 2020-10-01]. Dostupné z: `https://www.oreilly.com/library/view/a-guide-to/9781935589679/sub6.2.xhtml`

[7] ProjectEngineer: Dividing the Project into Tasks. [online], 2020, [cit. 2020-10-04]. Dostupné z: `https://www.projectengineer.net/tutorials/project-scheduling/dividing-the-project-into-tasks`

[8] Greene, J.; Stellman, A.: *Building the project Schedule*, kapitola Identify dependencies. O'Reilly, první vydání, 2005, ISBN 0596009488, s. 55–56.

[9] Greene, J.; Stellman, A.: *Building the project Schedule*, kapitola Create the Schedule. O'Reilly, první vydání, 2005, ISBN 0596009488, s. 57–58.

[10] Ravat-Farec, C.: Learn to distribute tasks and delegate. [online], 2019, [cit. 2020-11-25]. Dostupné z: https://openclassrooms.com/en/courses/5164326-work-effectively-in-a-team/5483336-learn-to-distribute-tasks-and-delegate

[11] Chaparro, O.; Bernal-Cárdenas, C.; Lu, J.; aj.: Assessing the Quality of the Steps to Reproduce in Bug Reports. ESEC/FSE 2019, New York, NY, USA: Association for Computing Machinery, 2019, ISBN 9781450355728, s. 86—96, doi:10.1145/3338906.3338947. Dostupné z: https://doi.org/10.1145/3338906.3338947

[12] Albrecht, C.: Guidelines for Writing Proper Tickets and Commits. [online], Leden 2018, [cit. 2020-10-06]. Dostupné z: https://www.lullabot.com/articles/guidelines-for-writing-proper-tickets-and-commits

[13] O.J., A.; L.H., D.: *Project management: The Art Of Managing Deadlines*, kapitola Handling missed deadline. Baywood's Technical Communications, Taylor & Francis, 2020, ISBN 9781351864749, s. 34–38, [cit. 2020-10-06]. Dostupné z: https://books.google.cz/books?id=E6QJEAAAQBAJ

[14] Greene, J.; Stellman, A.: *Applied Software Project Management*, kapitola Elements of a Successful Estimate. O'Reilly, první vydání, 2005, ISBN 0596009488, s. 34–38.

[15] Ambler, S. W.: Agile Core Practice: Prioritized Requirements. [online], 2014, [cit. 2020-09-18]. Dostupné z: http://agilemodeling.com/essays/prioritizedRequirements.htm

[16] Lucidchart Content Team: Introduction to MosCoW Prioritization. [online], [cit. 2020-09-18]. Dostupné z: https://www.lucidchart.com/blog/introduction-to-moscow-prioritization

[17] Wilson, F.: What Is the Eisenhower Matrix? How to Use It to Be More Productive? [online], Březen 2019, [cit. 2020-09-18]. Dostupné z: https://www.ntaskmanager.com/blog/eisenhower-matrix

[18] Tracy, B.: The ABCDE Method for Setting Priorities. [online], [cit. 2020-12-03]. Dostupné z: https://www.briantracy.com/blog/personal-success/the-abcde-method-for-setting-priorities

[19] Baer, D.: The 1-3-5 Rule For More Doable To-Do Lists. Fast Company [online], Květen 2013, [cit. 2020-11-10]. Dostupné z: https://www.fastcompany.com/3007885/1-3-5-rule-more-doable-do-lists

[20] Tracy, B.: *Eat That Frog! 21 Great Ways to Stop Procrastinating and Get More Done in Less Time*, kapitola Set the Table. Berrett-Koehler Publishers, druhé vydání, 2007, ISBN 1576754227, s. 16–21.

[21] Widrig, D.; Leffingwell, D.: *Managing Software Requirements: A Use Case Approach*. Leden 2003, ISBN 032112247X.

[22] Krutiš, M.: Kano model. [online], Srpen 2016, [cit. 2020-10-06]. Dostupné z: `https://www.krutis.com/kanuv-model`

[23] microTOOL: The Kano Model. A Means of Analyzing Customer Desires. [online], [cit. 2020-10-06]. Dostupné z: `https://www.microtool.de/en/knowledge-base/what-is-the-kano-model`

[24] Naji, C.: Kano model for product managers: Beginners' guide. [online], Červen 2017, [cit. 2020-10-06]. Dostupné z: `https://www.justinmind.com/blog/kano-model-for-product-managers-beginners-guide`

[25] Dash, S. N.: Product Prioritization Techniques in Agile Development. [online], Srpen 2020, [cit. 2020-09-18]. Dostupné z: `https://www.mpug.com/product-prioritization-techniques-in-agile-development`

[26] Stewart, G.: When Delegation Goes Wrong. [online], Prosinec 2013, [cit. 2020-10-18]. Dostupné z: `https://leadchangegroup.com/when-delegation-goes-wrong/`

[27] Conti, G.: How to Delegate Tasks Effectively (and Why It's Important). [online], Březen 2017, [cit. 2020-10-18]. Dostupné z: `https://www.meistertask.com/blog/delegate-tasks-effectively/`

[28] Hummel, M.; Rosenkranz, C.; Holten, R.: The Role of Communication in Agile Systems Development: An Analysis of the State-of-the-Art. *Business & Information Systems Engineering*, ročník 5, Říjen 2013, doi: 10.1007/s12599-013-0282-4.

[29] Lemon, L. L.: How communicators can avoid the 'telephone game' effect. [online], Říjen 2019, [cit. 2020-12-03]. Dostupné z: `https://www.ragan.com/how-communicators-can-avoid-the-telephone-game-effect`

[30] Rehkopf, M.: What is a kanban board? [online], [cit. 2020-12-20]. Dostupné z: `https://www.atlassian.com/agile/kanban/boards`

[31] Charvat, J.: *Project Management Methodologies: Selecting, Implementing, and Supporting Methodologies and Processes for Projects*, kapitola Project Framework vs Project Methodology. Wiley, 2003, ISBN 9780471221784, s. 18–20.

[32] ManagementMania: Řízení projektů (Project Management). [online], Leden 2016, [cit. 2020-12-03]. Dostupné z: https://managementmania.com/cs/metody-rizeni-projektu

[33] Blueprint: Agile Methodologies. [online], [cit. 2020-09-14]. Dostupné z: https://www.blueprintsys.com/agile-development-101/agile-methodologies

[34] McCormick, M.: Waterfall vs. Agile methodology. *MPCS*, 2012, [cit. 2020-09-14]. Dostupné z: http://mccormickpcs.com/images/Waterfall_vs_Agile_Methodology.pdf

[35] Mahalakshmi, M.; Sundararajan, M.: Traditional SDLC vs scrum methodology–a comparative study. *International Journal of Emerging Technology and Advanced Engineering*, ročník 3, č. 6, 2013: s. 192–196.

[36] IBM: Rational Unified Process: Best practices for software development teams. [online], 2003, [cit. 2020-12-03]. Dostupné z: https://www.ibm.com/developerworks/rational/library/253.html

[37] Denning, S.: Agile: The World's Most Popular Innovation Engine. *Forbes*, Červen 2015, ISSN 0015-6914, [cit. 2020-10-06]. Dostupné z: https://www.forbes.com/sites/stevedenning/2015/07/23/the-worlds-most-popular-innovation-engine/?sh=1fbecd9c7c76

[38] Compuware: Scrum Teams: What's with All the Scrum Meetings? [online], Srpen 2016, [cit. 2020-12-03]. Dostupné z: https://www.compuware.com/scrum-teams-whats-scrum-meetings

[39] Wells, D.: User Stories. [online], 1999, [cit. 2020-11-25]. Dostupné z: http://www.extremeprogramming.org/rules/userstories.html

[40] Wells, D.: Refactor Mercilessly. [online], 1999, [cit. 2020-11-25]. Dostupné z: http://www.extremeprogramming.org/rules/refactor.html

[41] Spalding, H.: What's the Optimal Team Size for Workplace Productivity? [online], Říjen 2015, [cit. 2020-11-25]. Dostupné z: https://www.getflow.com/blog/optimal-team-size-workplace-productivity

[42] Dumas, M.; Van der Aalst, W. M.; Ter Hofstede, A. H.: *Process-aware information systems: bridging people and software through process technology*, kapitola PAIS: Definition and Rationale. John Wiley & Sons, 2005, s. 5–8.

[43] Ackee s. r. o: O nás: Jsme designéři a vývojáři aplikací. [online], 2020, [cit. 2020-11-10]. Dostupné z: https://www.ackee.cz/o-nas

[44] German Brand Institute: Brand Award Winner Bundestag App. [online], 2020, [cit. 2020-11-10]. Dostupné z: https://www.german-brand-award.com/en/the-winners/gallery/detail/31460-bundestag-app.html

[45] Qest: Naše práce. [online], 2020, [cit. 2020-11-10]. Dostupné z: https://qest.cz/references

[46] 2FRESH: Co děláme. [online], 2020, [cit. 2020-11-10]. Dostupné z: https://www.2fresh.cz/sluzby

[47] Digital Ant: Pracujeme pro ambiciózní projekty. [online], 2020, [cit. 2020-11-10]. Dostupné z: https://www.digital-ant.cz/klienti

[48] Google: Tabulky Google – zdarma vytvářejte a upravujte tabulky online. [online], 2020, [cit. 2020-11-10]. Dostupné z: https://www.google.com/sheets/about

[49] Wilmington: Account manager. [online], Listopad 2018, [cit. 2020-12-09]. Dostupné z: https://managementmania.com/cs/account-manager

[50] 2FRESH: Costlocker – Timesheet & time tracking app. [online], 2020, [cit. 2020-11-15]. Dostupné z: https://costlocker.com

[51] Float: The resource planner that keeps projects on track. [online], 2020, [cit. 2020-12-09]. Dostupné z: https://www.float.com/

[52] Wiggers, K.: Trello reaches 50 million users, introduces new automation and template features. Říjen 2019, [cit. 2020-12-09]. Dostupné z: https://venturebeat.com/2019/10/30/trello-reaches-50-million-users-introduces-new-automation-and-template-features

[53] Trello Inc.: Trello. [online], 2020, [cit. 2020-11-25]. Dostupné z: https://trello.com

[54] Slack Technologies: Slack – Welcome to your new HQ. [online], [cit. 2020-11-25]. Dostupné z: https://slack.com/intl/en-cz

[55] Google: Hangouts Google. [online], [cit. 2020-12-25]. Dostupné z: https://hangouts.google.com

[56] Trello Inc.: Trello –- Aplikace na Google Play. [online], 2020, [cit. 2020-11-25]. Dostupné z: https://play.google.com/store/apps/details?id=com.trello

[57] Asana Inc.: Asana – Manage your team's work, projects & tasks online. [online], 2020, [cit. 2020-11-25]. Dostupné z: https://asana.com

[58] Asana Inc.: Asana: Your work manager –- Aplikace na Google Play. [online], 2020, [cit. 2020-11-25]. Dostupné z: https://play.google.com/store/apps/details?id=com.asana.app

[59] Notion Labs, Inc.: Notion – All-in-one workspace. [online], 2020, [cit. 2020-11-25]. Dostupné z: https://www.notion.so

[60] Notion Labs, Inc.: Notion - Notes, Tasks, Wikis –- Aplikace na Google Play. [online], 2020, [cit. 2020-11-25]. Dostupné z: https://play.google.com/store/apps/details?id=notion.id

[61] Notion Labs, Inc.: Does Notion have an API I can use? [online], 2020, [cit. 2020-11-25]. Dostupné z: https://www.notion.so/Does-Notion-have-an-API-I-can-use-4541b07a5caa46dba0026624646118c0

[62] Lang, J.-P.: Alternative (custom) Authentication HowTo. [online], Březen 2010, [cit. 2020-11-25]. Dostupné z: https://www.redmine.org

[63] Montaña, E. G.: Redmine Plugins Drectory: Srum. [online], [cit. 2020-11-25]. Dostupné z: https://www.redmine.org/plugins/scrum-plugin

[64] J, R.: Redmine Plugins Drectory: Event Notification / Email notification. [online], [cit. 2020-11-25]. Dostupné z: https://www.redmine.org/plugins/event_notifications

[65] IDEIL LLC: Redminer: projects and tasks Redmine – Aplikace na Google Play. [online], [cit. 2020-11-25]. Dostupné z: https://play.google.com/store/apps/details?id=com.ideil.redmine

[66] Google: Design – Material Design. [online], 2020, [cit. 2020-12-09]. Dostupné z: https://material.io/design

[67] Atlassian Corporation Plc: Jira – Issue & Project Tracking Software. [online], 2020, [cit. 2020-12-09]. Dostupné z: https://www.atlassian.com/software/jira

[68] Atlassian Corporation Plc: Jira Cloud by Atlassian – Aplikace na Google Play. [online], 2020, [cit. 2020-12-30]. Dostupné z: https://play.google.com/store/apps/details?id=com.atlassian.android.jira.core

[69] Atlassian Corporation Plc: Atlassian – Customers. [online], 2020, [cit. 2020-12-09]. Dostupné z: https://www.atlassian.com/customers

[70] Microsoft Corporation: Azure DevOps Services. [online], 2020, [cit. 2020-11-25]. Dostupné z: https://azure.microsoft.com/cs-cz/services/devops/

[71] Microsoft Corporation: View and update work items via the mobile browser. [online], Únor 2019, [cit. 2020-11-25]. Dostupné z: https://docs.microsoft.com/en-us/azure/devops/project/navigation/mobile-work?view=azure-devops

[72] Batkoski, E.; Chambers, T.; Jenkins, W.; aj.: Azure DevOps Services REST API Reference. [online], 2020, [cit. 2020-11-25]. Dostupné z: https://docs.microsoft.com/en-us/rest/api/azure/devops

[73] OpenJS Foundation: Node.js – Documentation. [online], [cit. 2020-10-11]. Dostupné z: https://nodejs.org/en/docs

[74] Balsamiq Studios: Balsamiq. Rapid, Effective and Fun Wireframing Software. [online], [cit. 2020-12-16]. Dostupné z: https://balsamiq.com

[75] Google: Buttons: floating action button. [online], Prosinec 2020, [cit. 2020-12-21]. Dostupné z: https://material.io/components/buttons-floating-action-button

[76] Sviatoslav, A.: The Best JS Frameworks for Front End. Leden 2020, [cit. 2020-12-16]. Dostupné z: https://rubygarage.org/blog/best-javascript-frameworks-for-front-end

[77] Facebook Inc.: React – A JavaScript library for building user interfaces. [online], [cit. 2020-10-11]. Dostupné z: https://reactjs.org

[78] BuiltWith Pty Ltd: JavaScript Library Usage Distribution on the Entire Internet. [online], Prosinec 2020, [cit. 2020-12-16]. Dostupné z: https://trends.builtwith.com/javascript/javascript-library/traffic/Entire-Internet

[79] Oluka, D.: 5 Most Popular React UI Component Libraries. Září 2020, [cit. 2020-10-10]. Dostupné z: https://hackernoon.com/5-most-popular-react-ui-component-libraries-5sx3t9c

[80] XTech: Ant Design of React. [online], [cit. 2020-10-11]. Dostupné z: https://ant.design/docs/react/introduce

[81] Facebook, Inc.: Create React App. [online], [cit. 2020-10-11]. Dostupné z: https://create-react-app.dev

[82] Catlin, H.; Weizenbaum, N.; Ahn, A.; aj.: Sass. [online], [cit. 2020-10-11]. Dostupné z: https://sass-lang.com

[83] Groupe Sharegate inc.: Craco – Create React App Configuration Override. GitHub repozitář [online], [cit. 11.10.2020]. Dostupné z: https://github.com/gsoft-inc/craco

[84] Abramov, D.: Redux – A Predictable State Container for JS Apps. [online], [cit. 2020-10-11]. Dostupné z: https://redux.js.org

[85] React Training: React Router: Declarative Routing for React.js. [online], [cit. 2020-10-11]. Dostupné z: https://reactrouter.com/

[86] Reardon, A.: Redux Thunk. GitHub repozitář [online], [cit. 2020-10-11]. Dostupné z: https://github.com/atlassian/react-beautiful-dnd

[87] OpenJS Foundation: Express – Node.js web application framework. [online], [cit. 2020-10-11]. Dostupné z: https://expressjs.com

[88] Agarwal, R.: Why use ExpressJS over NodeJS for Server-Side Development? Srpen 2014, [cit. 2020-10-11]. Dostupné z: https://www.algoworks.com/blog/why-use-expressjs-over-nodejs-for-server-side-coding

[89] OpenJS Foundation: Frameworks built on Express. [online], 2017, [cit. 2020-10-11]. Dostupné z: https://expressjs.com/en/resources/frameworks.html

[90] MongoDB, Inc.: The most popular database for modern apps – MongoDB. [online], [cit. 2020-10-11]. Dostupné z: https://www.mongodb.com

[91] Automattic, Inc.: Mongoose ODM. [online], [cit. 2020-10-11]. Dostupné z: https://mongoosejs.com

[92] Tavan, C.; Henke, G.; Ciardi, F.: Express validator: Getting Started. GitHub repozitář [online], [cit. 2020-10-11]. Dostupné z: https://express-validator.github.io/docs

[93] Hanson, J.: Passport.js. [online], [cit. 2020-10-11]. Dostupné z: http://www.passportjs.org

[94] cookie-session. [online], [cit. 2020-10-11]. Dostupné z: https://www.npmjs.com/package/cookie-session

[95] Axios: Promise based HTTP client for the browser and node.js. GitHub repozitář [online], [cit. 2020-10-11]. Dostupné z: https://github.com/axios/axios

[96] Moment.js. [online], [cit. 2020-10-11]. Dostupné z: https://momentjs.com

[97] Prettier Opinionated Code Formatter. [online], [cit. 2020-10-11]. Dostupné z: https://prettier.io

[98] OpenJS Foundation: ESLint - Pluggable JavaScript linter. [online], [cit. 2020-10-11]. Dostupné z: https://eslint.org

[99] Airbnb: Airbnb JavaScript Style Guide(). GitHub repozitář [online], [cit. 2020-10-11]. Dostupné z: https://github.com/airbnb/javascript

[100] SendGrid: Email Delivery Service. [online], [cit. 2020-10-11]. Dostupné z: https://sendgrid.com/

[101] Google: Google Identity. [online], [cit. 2020-10-11]. Dostupné z: https://developers.google.com/identity

[102] Amazon Web Services, Inc.: Amazon S3. [online], [cit. 2020-10-11]. Dostupné z: https://aws.amazon.com/s3

[103] MongoDB, Inc.: *MongoDB Limits and Thresholds*. [cit. 2020-12-05]. Dostupné z: https://docs.mongodb.com/manual/reference/limits

[104] Schaefer, L.: NoSQL vs SQL Databases. [online], [cit. 2020-10-06]. Dostupné z: https://www.mongodb.com/nosql-explained/nosql-vs-sql

[105] Fielding, R. T.; Taylor, R. N.: *Architectural Styles and the Design of Network-Based Software Architectures*. Dizertační práce, 2000, [cit. 2020-12-05]. Dostupné z: https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[106] Mozilla Foundation: HTTP caching. [online], Prosinec 2020, [cit. 2020-12-21]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTTP/Caching

[107] Thijssen, J.: What is HATEOAS and why is it important for my REST API? [online], [cit. 2020-12-16]. Dostupné z: https://restcookbook.com/Basics/hateoas

[108] He, Y.: async-validator. GitHub repozitář [online], [cit. 2020-12-16]. Dostupné z: https://github.com/yiminghe/async-validator

[109] Salesforce: Heroku: Cloud application platform. GitHub repozitář [online], [cit. 2020-12-10]. Dostupné z: https://heroku.com

[110] OpenJS Foundation: Mocha – the fun, simple, flexible JavaScript test framework. [online], [cit. 2020-12-10]. Dostupné z: https://mochajs.org

[111] Luer, J.; Cirkel, K.; da Costa, L. F.; aj.: Chai Assertion Library. [online], [cit. 2020-12-10]. Dostupné z: https://www.chaijs.com

[112] Facebook, Inc.: Jest is a delightful JavaScript Testing. [online], [cit. 2020-12-10]. Dostupné z: https://jestjs.io

[113] Stanley, P.; Lin, F. P.: Avataaars Generator - A free online avatar generator for anyone to make their beautiful personal avatar easily! [online], [cit. 2020-12-10].

Příloha A

Seznam použitých zkratek

AES Advanced Encryption Standard.

API Application Programming Interface.

AWS Amazon Web Services.

CEO Chief executive officer.

COO Chief operations officer.

CRA Create React App.

CRM Customer relationship management.

CSS Cascading Style Sheets.

FIT Fakulta informačních technologií.

HATEOAS Hypermedia as the Engine of Application State.

HTML Hypertext Markup Language.

HTTP Hypertext Transfer Protocol.

IT Informační technologie.

JSON JavaScript Object Notation.

LDAP Lightweight Directory Access Protocol.

MERN MongoDB Express React Node.

MVC Model View Controller.

ODM Object-Ducment Mapping.

PMBOK Project Management Body of Knowledge.

REST Representational state transfer.

RUP Rational Unified Process.

SPA Single-page Appliaction.

SQL Structured Query Language.

SSE Server Sent Events.

TM Task management.

UI User interface.

UML Unified Modeling Language.

URL Uniform Resource Locator.

USD United States Dollar.

UX User experience.

VŠE Vysoká škola ekonomická.

WBS Work breakdown structure.

WYSIWYG What you see is what you get.

XP Extreme programming.

ČVUT České vysoké učení technické.

Příloha B

Instalační příručka

B.1 Demo aplikace

Pro vyzkoušení téměř všech funkcionalit aplikace je dostupné nasazené demo aplikace. Demo je dostupné pro všechny google účty, které využívají email pod doménou fit.cvut.cz. Aplikace využívá testovacích údajů poskytnutých firmou Ackee a není tedy možné upravovat seznamy úloh a projektů, které lze přidělit. Pro přístup do aplikace se stačí přihlásit pod svým účtem na FIT ČVUT. Uživatelům s tímto emailem jsou automaticky přiřazena moderátorská práva. V aplikaci je tak možné otestovat všechny funkce kromě úpravy uživatelů. Demo aplikace je nasazeno na adrese:

`https://taskboard-demo.herokuapp.com`.

B.2 Lokální spuštění

Pro lokální otestování je nutné zprovoznit vlastní instanci Redmine dle návodu na `https://www.redmine.org/projects/redmine/wiki/redmineinstall`. Dále je nutné zprovoznit vlastní účty do všech externích systémů, které aplikace využívá. Pro spuštění je nutné mít nainstalováno:

1. node.js verze >= 14.15.1,
2. npm verze >= 6.14.8.

Zdrojový kód je dostupný z `https://github.com/pavelsvagr/taskboard`.

B.2.1 Založení účtů v externích aplikacích (Prosinec 2020)

Veškerou konfiguraci externích služeb řiďte přes soubor:

`server/config/_dev.js`.

Zde ukládejte pouze informace, které naleznete v návodech v této sekci.

B.2.1.1 MongoDB

Pro zprovoznění je nutné mít založenou vlastní databázi v oficiálním cloudu MongoDB. Toho lze docílit v následujících krocích.

1. Založte si účet na `https://cloud.mongodb.com`.
2. Pro svou organizaci založte nový projekt.
3. V projektu vytvořte nový cluster, vyberte možnost clusteru zdarma a vyberte nejbližšího dostupného poskytovatele.
4. U založeného clusteru klikněte na tlačítko „connect“ a založte přístupové údaje. V případě, že chcete soukromou databázi, zadejte do přístupu pouze svou IP adresu. Uložte si přístupové údaje pro Vámi vytvořeného uživatele.
5. Na další záložce zvolte možnost „Connect your application“ a uložte si connection string.
6. U clusteru nyní klikněte na „Collections“.
7. Na další stránce klikněte na „Add My Own Data“ a vytvořte databázi s názvem „taskboard“ a `collection name` zvolte jako „users“.
8. Do souboru `_dev.js` nyní do hodnoty `mongoURI` vložte connection string z kroku 6. Část řetězce `<password>` nahraďte Vámi nastaveným heslem. Část řetězce `<dbname>` nahraďte názvem databáze: „taskboard“

B.2.1.2 Google Identity

Pro zprovoznění přihlášení pomocí oAuth2 přes službu Google Identity je nutné vytvořit údaje pro vaší aplikaci v Google Cloud. Ten je dostupný například z adresy¨
`https://console.cloud.google.com`.

1. Založte nový projekt pro tuto aplikaci.
2. V levém menu vyberte „APIs & Services“ a „OAuth Consent Screen“.
3. Vyplňte pouze povinné položky a klikněte na „Save“.
4. V levém bočním menu vyberte „Credentials“.
5. Na horním panelu klikněte na „Create“ a vyberte „oAuth Client ID“.
6. Zvolte typ „Web application“.
7. Klikněte na „add URI“ v položce „URIs“ a vyplňte `http://localhost:5000`.

8. Klikněte na „add URI“ v „Redirect URIs“ a zadejte `http://localhost:3000/auth/google/callback`.

9. Klikněte na „Create“.

10. Otevře se Vám okno s přístupovými údaji zkopírujte do souboru `_dev.js`:

    Your Client ID do `googleClientID`

    Your Client Secret do `googleClientSecret`

B.2.1.3 Amazon S3

Pro ukládání avatarů založte účet na `https://aws.amazon.com/s3`.

1. Přihlašte se a běžte na stránku `https://s3.console.aws.amazon.com`.

2. Klikněte na „Crete new bucket“ a pojmenujte ho „taskboard“.

3. Pro testování odklikněte „Block all public access“ a odešlete.

4. V seznamu vyberte váš nový bucket a běžte do záložky „Permissions“.

5. Zde sjeďte dolů na „Edit cross-origin resource sharing (CORS)“, klikněte na „edit“ a vložte následující json a uložte.

```
[
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "GET",
            "HEAD",
            "POST",
            "PUT"
        ],
        "AllowedOrigins": [
            "*"
        ],
        "ExposeHeaders": []
    }
]
```

6. Klikněte na své jméno v pravém horním rohu a vyberte „My Security Credentials“.

7. Zde rozklikněte záložku „Access keys (access key ID and secret access key)“.

8. Klikněte na „Create New Acces Key“ a zkopírujte údaje do souboru `_dev.js` následovně:.

   Access Key ID do `awsKeyId`

   Secret Access Key do `awsKeySecret`

9. V souboru `_dev.js` vyplňte do pole `awsBucketName` jméno vytvořeného bucketu.

B.2.1.4 SendGrid (volitelné)

V případě, že chcete využít funkce emailových notifikací, založte si účet na `https://app.sendgrid.com` a ve svém profilu na záložce „API keys“ vytvořte přístupové údaje. Do `_dev.js` souboru poté zkopírujte:

API key do `sendGridKey`.

Adresu vyplněnou při zakládání účtu do `fromEmail`

`http://localhost:3000` do `redirectDomain`

B.2.2 Konfigurace a spuštění aplikace

Do souboru `_dev.js` doplňte náhodné textové řetězce pro položky:

- `cookieKey`,
- `encryptPassphrase`.

Soubor `_dev.js` následně přejmenujte na `dev.js`. Spusťte příkaz v kořenovém adresáři projektu:

`npm run install`.

Měly by se vám nanstalovat všechny balíčky potřebné pro spuštění aplikace. Pro úspěšné přihlášení do aplikace máte následující možnosti.

1. Do `dev.js` do pole `whitelistedEmailsAdmin` zadejte doménu vašeho emailu na Google účtu (např. gmail.com). Tím bude každé přihlášení s emailem pod touto doménou úspěšné a bude Vám přiřazena role administrátora.

2. Nebo se držte sekce B.2.3 pro vytvoření testovacích dat jejichž součástí je i administrátorský účet pro testování.

Nyní spusťe příkaz `npm run dev`. Po dokončení sestavování aplikace by se Vám automaticky mělo otevřít okno prohlížeče. Pokud neotevře, mělo by být možné k aplikaci přistoupit přes adresu: `http://localhost:3000/`. Pokud to tak není, zkontrolujte si, že Vám na portech 5000 a 3000 neběží žádná jiná aplikace.

B.2.3 Testovací data

V aplikaci je možné zprovoznit sadu testovacích dat. Dle návodu na Redmine wiki[21] zprovozněte na Vaší instanci přístup přes API. Do `dev.js` vložte následujícíc údaje:

URL pro přístup k instanci Redmine do `testCredentialsRedmineUrl`

API klíč pro Redmine do `testCredentialsRedmineApiKey`

Email od vašeho Google účtu do `testUserAdminGmail`

Nyní spusťte příkaz:

```
npm run seed --prefix server
```

Tím bude v databázi vytvořeno několik testovacích dat a uživatel s administrátorskými právy pro vámi zadaný účet.

B.2.4 Spuštění testů

Pro spuštění testů spusťte v kořenovém adresáři projektu příkaz:

```
npm run test
```

B.2.5 Vyprázdnění databáze

Pro vyprázdnění databáze pro nové testování můžete využít příkaz:

```
npm run truncate --prefix server
```

[21] `https://www.redmine.org/projects/redmine/wiki/rest_api`

Příloha C

Souhlasy respondentů s pořízením rozhovoru

Respondenti při poskytování rozhovoru souhlasili s jeho zveřejněním a přepsáním do textu. Odkaz na výslednou aplikaci i text rozhovoru byl všem respondentům po vyhotovení zaslán. Název diplomové práce na informovaných souhlasech nesouhlasí s aktuálním, jelikož byl ještě po pořízení rozhovorů měněn.

Informovaný souhlas

s poskytnutím rozhovoru a s využitím informací získaných ze zaznamenaného rozhovoru pro účely tvorby diplomové práce:

„Analýza procesu zadávání úkolu malému týmu a webová aplikace na podporu práce s úkoly"

Cílem diplomové práce je získat a analyzovat problémy při zadávání úkolů v malém a středním týmu s ohledem na organizační strukturu firmy a využívané metodiky. Autorem diplomové práce je Pavel Švagr, vedoucím diplomové práce je Ing. Martin Pulpitel.
Z rozhovoru bude pořízen digitální audiozáznam a jeho přepis. Získané informace, záznam a přepis budou využity k vypracování diplomové práce a implementaci prototypu aplikace, který bude zpětně poskytnut informantovi k testování.

Souhlas

Podle zákona č.101/2000 Sb., o ochraně osobních údajů, ve znění pozdějších předpisů, uděluji Pavlovi Švagrovi souhlas se zpracováním svých osobních a citlivých údajů ke studijním účelům, poskytnutých v rozhovoru v rámci diplomové práce na fakultě Informačních technologií ČVUT.

Souhlasím také, že s písemným přepisem svého rozhovoru budu před publikováním diplomové práce seznámen/a, abych se k němu mohl/a vyjádřit (autorizace):

ano ne

Souhlasím také se zveřejněním přepisu rozhovoru ostatním respondentům:

ano ne

Jméno, příjmení, podpis:

V Praze **dne:** 20.12.2020 Martin Půlpitel

Informovaný souhlas

s poskytnutím rozhovoru a s využitím informací získaných ze zaznamenaného rozhovoru pro účely tvorby diplomové práce:

„Analýza procesu zadávání úkolu malému týmu a webová aplikace na podporu práce s úkoly“

Cílem diplomové práce je získat a analyzovat problémy při zadávání úkolů v malém a středním týmu s ohledem na organizační strukturu firmy a využívané metodiky. Autorem diplomové práce je Pavel Švagr, vedoucím diplomové práce je Ing. Martin Půlpitel.
Z rozhovoru bude pořízen digitální audiozáznam a jeho přepis. Získané informace, záznam a přepis budou využity k vypracování diplomové práce a implementaci prototypu aplikace, který bude zpětně poskytnut informantovi k testování.

Souhlas

Podle zákona č.101/2000 Sb., o ochraně osobních údajů, ve znění pozdějších předpisů, uděluji Pavlovi Švagrovi souhlas se zpracováním svých osobních a citlivých údajů ke studijním účelům, poskytnutých v rozhovoru v rámci diplomové práce na fakultě Informačních technologií ČVUT.

Souhlasím také, že s písemným přepisem svého rozhovoru budu před publikováním diplomové práce seznámen/a, abych se k němu mohl/a vyjádřit (autorizace):

ano (ne)

Souhlasím také se zveřejněním přepisu rozhovoru ostatním respondentům:

(ano) ne

Jméno, příjmení, podpis:

V PRAZE dne: 11. 9. 2019

Marku Kopřivník

Informovaný souhlas

s poskytnutím rozhovoru a s využitím informací získaných ze zaznamenaného rozhovoru pro účely tvorby diplomové práce:

„Analýza procesu zadávání úkolu malému týmu a webová aplikace na podporu práce s úkoly"

Cílem diplomové práce je získat a analyzovat problémy při zadávání úkolů v malém a středním týmu s ohledem na organizační strukturu firmy a využívané metodiky. Autorem diplomové práce je Pavel Švagr, vedoucím diplomové práce je Ing. Martin Půlpitel.
Z rozhovoru bude pořízen digitální audiozáznam a jeho přepis. Získané informace, záznam a přepis budou využity k vypracování diplomové práce a implementaci prototypu aplikace, který bude zpětně poskytnut informantovi k testování.

Souhlas

Podle zákona č.101/2000 Sb., o ochraně osobních údajů, ve znění pozdějších předpisů, uděluji Pavlovi Švagrovi souhlas se zpracováním svých osobních a citlivých údajů ke studijním účelům, poskytnutých v rozhovoru v rámci diplomové práce na fakultě Informačních technologií ČVUT.

Souhlasím také, že s písemným přepisem svého rozhovoru budu před publikováním diplomové práce seznámen/a, abych se k němu mohl/a vyjádřit (autorizace):

ano ne

Souhlasím také se zveřejněním přepisu rozhovoru ostatním respondentům:

ano ne

V 27.9.2019 dne: Praze

Jméno, příjmení, podpis: Kůrková

Informovaný souhlas

s poskytnutím rozhovoru a s využitím informací získaných ze zaznamenaného rozhovoru pro účely tvorby diplomové práce:

„Analýza procesu zadávání úkolu malému týmu a webová aplikace na podporu práce s úkoly“

Cílem diplomové práce je získat a analyzovat problémy při zadávání úkolů v malém a středním týmu s ohledem na organizační strukturu firmy a využívané metodiky. Autorem diplomové práce je Pavel Švagr, vedoucím diplomové práce je Ing. Martin Půlpitel.
Z rozhovoru bude pořízen digitální audiozáznam a jeho přepis. Získané informace, záznam a přepis budou využity k vypracování diplomové práce a implementaci prototypu aplikace, který bude zpětně poskytnut informantovi k testování.

Souhlas

Podle zákona č.101/2000 Sb., o ochraně osobních údajů, ve znění pozdějších předpisů, uděluji Pavlovi Švagrovi souhlas se zpracováním svých osobních a citlivých údajů ke studijním účelům, poskytnutých v rozhovoru v rámci diplomové práce na fakultě Informačních technologií ČVUT.

Souhlasím také, že s písemným přepisem svého rozhovoru budu před publikováním diplomové práce seznámen/a, abych se k němu mohl/a vyjádřit (autorizace):

(ano) ne

Souhlasím také se zveřejněním přepisu rozhovoru ostatním respondentům:

(ano) ne

V Praze dne: 25.9.2019

Jméno, příjmení, podpis: Štěpán Hrubý

Příloha D

Přepsané rozhovory z firem

D.1 Ackee (Ing. Martin Půlpitel)

Moderátor: Mohl byste se krátce představit a zmínit jakou zastáváte pozici?

Ing. Martin Půlpitel: Jmenuji se Martin Půlpitel a jsem CEO ve firmě Ackee.

Moderátor: Co vaše firma dělá? Na co se specializuje? V čem jste nejlepší?

Ing. Martin Půlpitel: Jsme agentura nabízející vývoj mobilních a webových aplikací. Pro naše klienty děláme kompletní řešení od návrhu UI a UX, přes design až samotný vývoj a údržbu. Mobilní aplikace děláme aplikací pro iOS a Android a weby píšeme v Node.js. Už to děláme poměrně dlouho a máme za sebou spoustu úspěšných projektů.

Moderátor: Kolik má vaše firma zaměstnanců?

Ing. Martin Půlpitel: Je to šedesát až sedmdesát. Průběžně se to mění, ale pravidelně se blížíme k té sedmdesátce.

Moderátor: Jak velké jsou týmy, které typicky pracují na vašich projektech?

Ing. Martin Půlpitel: To je různé. Ty nejmenší týmy jsou o dvou až třech lidech. Ty největší mají třeba i patnáct členů.

Moderátor: Jak jednotlivé týmy rozdělujete?

Ing. Martin Půlpitel: Jelikož děláme pro různé technologie, tak máme výkonné týmy primárně rozdělené technologicky. Máme třeba týmy pro Android, iOS a tak dále. Tyto týmy mají svého team leadera a mívají velikost přibližně do osmi lidí.

Pak ale skládáme týmy pro specifické projekty a ty jsou často složené napříč těmi technickými. Takže typicky tam jsou jeden nebo dva lidi pro Android,

jeden dva lidi pro iOS, někdo z backendu, lidi pro UX, design, projektový manažer a další. Tento tým pak dělá na určitém projektu, který se skládá ze spousty malých úkolů, které dostává zadaný konkrétní člověk. Někdy jsou to i dva lidi, ale typicky pouze jeden.

Moderátor: Držíte se některých metodik projektového řízení?

Ing. Martin Půlpitel: Na některých projektech, typicky těch větších, jedeme agilně podle Scrumu. Na méně důležitých projektech, které trvají jen pár měsíců, je to buď klasický vodopád nebo modifikovaný Scrum. Člověk už si to tak trochu určuje podle zkušeností a není tam striktní projektové řízení, protože by to pro nás byl overkill[22].

Moderátor: Mohl byste pospat proces rozdělování a zadávání těchto úloh?

Ing. Martin Půlpitel: Úlohy zadává projektový manažer, který ví jaké lidi z technologických týmů má v projektu. Má i přehled o tom, kdo z členů týmu je v dané funkci seniornější a tak jim zadává spíše ty obtížnější úkoly a těm juniornějším členům týmu zadává spíše ty jednodušší úkoly. Navzájem se pak dělají merge requesty v gitu.

Všechny úkoly zadává v tiketovacím systému Redmine, což je dost podobný systém například komerčnímu řešení Jira, jen je open source. Každý si v tom systému filtruje svoje úkoly a centrálně ukládáme všechno tam. Čistě pro nějaký přehled a reporting pak slouží tabulka v google sheets. Jelikož je pro nás těžké sledovat, kdo na čem zrovna dělá, máme v této tabulce zanesené informace o tom, kdo na čem zrovna dělá ten daný týden. Není to žádné dogma, je to čistě jen pro orientaci a přehled. Například Pepa pracuje tento týden na projektu android 1 a Franta dělá napůl na projektu web 1 a web 2. Každý se pak ale dívá hlavně do Redmine na konkrétní úkoly. Pokud jich tam má víc, podívá se do tabulky, zjistí, že má prioritně dělat například na web 1, tak bere nejdříve úkoly z projektu web 1. Tabulku vyplňujeme společně, takže se tam mohou i lidi vyjádřit, který úkol by chtěli dělat, kdo je lepší v čem a podobně, takže nám to hodně vyhovuje.

Moderátor: Jak taková tabulka vypadá?

Ing. Martin Půlpitel: Jsou tam rozepsané do řádků technologické týmy, pod každým názvem týmu vždy všichni jeho členové. Máme tam tři sloupečky pro projekty, což se ukázalo, že většinou stačí. V každém sloupečku pak máme projekt, na kterém ten daný týden člen týmu pracuje. Určili jsme si, že první sloupeček je nejprioritnější a odpovídá zhruba práci na 3 dny. Druhý sloupeček je pak nějaká střední priorita s prací na 1,5 dne a třetí sloupeček je nízká priorita s vytížením třeba 0,5 dne.

[22]Přehnaně drahé nebo náročné řešení situace.

Je to opravdu orientační. Pokud přijde třeba nějaká chyba nebo něco prioritnějšího, tak si to samozřejmě projektový manažer převezme a konzultuje to pak s dotyčnými osobami, které to budou řešit, takže je možné, že se třeba k některým projektům nedostanou. Testovali jsme i nějaké konkurenční systémy, kde se do podobné tabulky skládají přímo úkoly a podle náročnosti úkolu se člověku ukrajuje čas. To bylo ale hrozně nepřesné, protože to nikdy neodpovídalo realitě a pro nás je prostě lepší mít jeden systém na centralizaci úkolů, a pak mít jen přehled projektů. Skládat takhle přesně časové odhady a říkat, že máme na 100 procent vytíženou firmu tento týden pro nás postrádalo smysl, protože tak to prostě v realitě nefunguje.

Moderátor: Za proces zadávání úkolů je tedy zodpovědný projektový manažer?

Ing. Martin Půlpitel: Ano, je to tak.

Moderátor: Jste s procesem zadávání úloh spokojený? Chtěl byste na něm něco změnit?

Ing. Martin Půlpitel: Zdá se nám hodně efektivní. Co jsme to tak porovnávali s agenturami z našeho okolí, tak máme utilizaci toho procesu velice dobrou. Díky tomu, že máme přehled o aktivitách našich zaměstnanců, tak umíme jejich čas efektivně využít a každý vždy ví, kdy co má dělat. Samozřejmě jsou tam výjimky, ale to budou vždycky. Takhle se nám to zdá dobré.

Moderátor: Jste si vědom nějaké kritické oblasti, kde dochází ke zdržování procesu nebo častým chybám?

Ing. Martin Půlpitel: Asi vím o dvou oblastech. První je, když má na sobě přes Redmine daný člověk přiřazeno hodně tiketů, tak Redmine neumí moc dobře určovat priority. Je tam vlastně předdefinováno pár priorit, myslím, že jsou to „low“, „medium“, „high“ a „superhigh“. Pak si tam můžete dodefinovat nějakou vlastní paletu, ale není to prostě od jedné do padesáti, takže si to člověk musí pak často určovat sám nebo to pak musí zjišťovat na základě schůzek jako každodenního stand upu.

Někdy pak prioritní úkol určí špatně nebo se mu ten úkol nechce dělat a tak dělá ten další, což ale asi neodstraní žádný nástroj na světě, to je vyloženě lidský faktor. Občas tomu pomůže prioritu určovat přes termín, ale to není spásné, protože ty termíny se zase mohou překrývat.

Druhý zdroj neefektivity je zpoždění zadání nebo člověku dojde práce a pak se ozývá, že nemá co dělat, tak dochází k nevytíženosti. Tam to pak musí projektový manažer rychle řešit třeba tím, že rychle vypíše nějaký jiný úkol. To jsou takové typické chyby.

Moderátor: Jak udržují projektoví manažeři přehled o plnění stavu úkolů?

Ing. Martin Půlpitel: Všechno sledují v Redmine. My ten systémy používáme již 8 let, máme tam spoustu vlastních úprav a nastavenou vlastí workflow. Programátor když začne pracovat na daném úkolu, tak ho dává do stavu In progress, může tam vyplňovat procenta stavu plnění úkolu a to se reportuje autorovi tiketu, což je tedy vždy projektový manažer.

Když úkol jde na merge request nebo při vyřešení úkolu, když se dostává do testovacího stavu a jde na to někdo z QA[23], tak se o tom pomocí nějaké notifikace vždy projektový manažer dozví. Když nějakou informaci manažer nemá, tak si jí díky té centralizaci v Redmine může prostě jednoduše vyfiltrovat. Takže ten přehled má téměř dokonalý.

Moderátor: Používáte nějaký systém na dokumenty a dokumentaci?

Ing. Martin Půlpitel: Zadání vždy píšeme přímo d Redmine. Dokumentace o projektech se pak vždy odvíjí od požadavků zákazníka. Na dokumenty používáme nástroje od googlu. Pro dokumentaci produktu pak vždycky typicky dokumentujeme přímo kód a píšeme ho přehledně a to většině zákazníků stačí. Když má klient nějaký jiný požadavek na dokumentaci, tak se to řeší individuálně s ním.

Moderátor: Zdá se Vám využívaný podporovaný sytém Redmine přehledný?

Ing. Martin Půlpitel: To je taková věčná debata. Ono je to tak, že když přijde někdo, kdo je zvyklý na systém Jira, tak Redmine samozřejmě nenávidí, ale když zas někdo Redmine dlouho používá, tak nechce nic jiného. Je to open source, takže to samozřejmě nikdo moc neřeší, ale určitě by to mohlo být lepší. Když se s tím člověk nějakým způsobem naučí, tak tam najde všechno co potřebuje. Máme léta diskuze, jeslti nepřejdeme na Jiru, ale ono by to asi nebylo o moc lepší.

Moderátor: Používáte Redmine i pro komunikaci?

Ing. Martin Půlpitel: Používáme ho pro komunikaci ohledně úkolů, když se úkol hodně předává přes týmy nebo více lidí, aby každý měl přehled o tom, co se v úkolu dělo, ale jinak samozřejmě pro nějakou rychlou komunikaci používáme Slack, nebo video, případně pak minimálně email. Nejvíce ale preferujeme komunikaci face to face, která je pro nás nejefektivnější a nejlepší.

Moderátor: Redmine je založený na pluginech, jsou tam přesto nějaké funkce, které nepoužíváte?

Ing. Martin Půlpitel: Je to opravdu tak, že všechno co používáme, tam máme nějakým způsobem natažené přes pluginy. Typicky ale nepoužíváme

[23]Quality assurance, v překladu „zajištění jakosti“, je proces kontroly kvality produktu.

z těch základních věcí třeba modul na wiki, protože wiki si prostě nemáme potřebu psát. Na plánování zas nepoužíváme ten Ganttův diagram, jelikož fungujeme většinou agilně a nejsme například stavební firma, kde si to všechno lze takhle jednoduše rozvrhnout.

Moderátor: Jak řešíte logování času?

Ing. Martin Půlpitel: V Redmine jsou funkce pro zapsání stráveného času. Lidi většinou používají nějaký vlastní nástroj třeba Toggl[24], kde si měří, kolik na tom času strávili a pak to tam prostě zapíšou. Přes některé systémy jako třeba Zapier[25] mohou ten čas pak přímo exportovat do Redmine. To už je pak na těch konkrétních lidech, co jim více vyhovuje. Z Redmine přímo i čas vykazujeme, jsme v tomhle transparentní, takže každý z klientů má přístup k našim úkolům a může tam přes rozhraní i hlásit chyby nebo sledovat stav toho projektu.

D.2 Qest (Ing. Martin Koperniech)

Moderátor: Mohl byste se krátce představit a zmínit jakou zastáváte pozici?

Ing. Martin Koperniech: Volám sa Martin Koperniech a pracujem v spoločnosti Qest ako výkonný riaditeľ.

Moderátor: Co vaše firma dělá? Na co se specializuje?

Ing. Martin Koperniech: Firma prešla určitou genéziou, každopádne dnes tomu, čo robíme, hovoríme „Team as a service“. Náš zákazník má svoj biznis, môže, ale nemusí mať vlastné vývojové kapacity, no primárne má biznis, súčasťou ktorého je software a my sme tí, ktorí dodávajú tím. Nefungujeme ako body shop – máme vlastnú infraštruktúru, tím testerov, automatizované nástroje a to aj ponúkame nášmu klientovi.

Klient je ten, kto určuje požiadavky, ale my ich agilne spravujeme. Máme teda určitý presah do produktu, avšak principiálne sme s klientom partneri. Nerobíme teda nič špecifického oproti iným firmám, akurát sa snažíme udržať tímy čo najviac zapuzdrené a u nás, bez závislosti na iných firmách.

Zapuzdrené myslím v tom zmysle, že každý tím je zložený z podobne skúsených ľudí a snažíme sa u nich budovať vzťahy. Zároveň sa snažíme, aby každý tím prijal daný produkt za svoj, aby bol s investormi ako jedna duša a o to ide – aby si ľudia neprišli odsedieť 8 hodín do práce, ale aby za ten produkt naozaj kopali. Robíme primárne frontend i backend, nemáme designerov ani UX tím, to berieme priamo od investora, čo sa nám overilo, alebo máme externé skupiny, ktoré nám s tým pomáhajú. Sme teda zameraní hlavne na vývoj,

[24]Nástroj pro sledování času dostupný z `https://toggl.com`.
[25]Systém pro automatizované sdílení informací dostupný na `https://zapier.com`.

testing a product management. Máme tu produkťákov, testerov a pomedzi to vývojový tím. Všetko tvoríme v JavaScripte, aby mali čo najužší stack.

Moderátor: Kolik má vaše firma změstnanců?

Ing. Martin Koperniech: Dnes je to tridsaťpäť . Bolo nás cca štyridsaťpäť, mali sme 10 mobilných vývojárov, ale to sme si povedali, že nie je úplne cesta, ktorou sa chceme vydať a tak si chlapci založili vlastnú firmu a fungujeme spoločne na rovnakých zákazkách, takže de facto je nás viac, ale v tom našom JavaScriptovom core tíme je nás tridsaťpäť.

Moderátor: Jakým způsobem máte rozdělené týmy? Kolik členů obvykle mají?

Ing. Martin Koperniech: U nás máme primárne rozdelené tímy podľa projektov. Každý tím obsahuje ľudí podľa potreby na frontend a ľudí na backend, vlastného product ownera, takže máme tímy vyložene uzatvorené. Nerobíme veľmi veľké projekty, takže väčšinou mávame tímy rozdelené na 5 až 6 ľudí a projekt beží najčastejšie 5 až 6 mesiacov.

Moderátor: Používáte některé z metodik projektového řízení?

Ing. Martin Koperniech: Áno, tímy bežia v metodike Scrum, takže sa všetky projekty snažíme riadiť agilne. Robíme to tak, aby i investor chápal, čo agile je, čiže v prvej fáze investora zaškoľujeme a vysvetľujeme mu ako proces funguje. Používame na to TFS, dnes už sa to volá Devops, kde si nastavujeme kompletne celý proces. Máme tam dva boardy - jeden produktový, viac businessový, ktorý vidia všetci a druhý je iteračný v rámci iterácie. Snažíme sa to robiť tak ako to Scrum predpisuje, to znamená že všetky tímy by mali byť vyrovnané a ľudia si berú úlohy z backlogu podľa priorít.

Moderátor: Jak priority úloh řešíte?

Ing. Martin Koperniech: O to sa stará produkťák. Každý si z backlogu berie svoje úlohy podľa priorít a produkťák sa stará o to, aby boli úlohy dobre zoradené. Nemáme žiadny zoznam priorít, v Devops v backlogu čím vyššie úloha je, tým má väčšiu prioritu. Akonáhle teda niekto niečo dokončí, berie si ďalšiu úlohu z backlogu. Takže napríklad pokiaľ som frontenďák, je stand up a viem, že som dokončil svoju úlohu, tak beriem prvú vec, čo je najvyššie na frontende a pokračujem. Samozrejme tam môže byť určitá závislosť, v tom prípade to produkťák prehadzuje do blocked a na stand upe sa rieši, ktoré veci sú konfliktné a podobne.

Takže jedno nastavovanie priorít je behom groomingu, kde sa snažíme, aby tam bol i niekto od investorov, zadávateľ, stakeholder zo strany zákazníka. Tam sú zoradené backlog úlohy v prvom rade pre danú iteráciu a v druhom rade podľa priority - od feature po bugy v celom kontexte cez všetky boardy. Postupom

času tá priorita klesá, ale prvých pár úloh v backlogu sú vždy zoradené podľa priority, preto sú na groomingu prítomní i vývojári, a keď príde požiadavka veľmi technická a dôležitá, tak produkťák si potom musí prioritu odôvodniť.

Moderátor: Jako podpůrný systém tedy používáte Azure Devops? Jste s ním spokojení?

Ing. Martin Koperniech: Áno, je to v podstate alternatíva Jiry. My sme na to zvyknutí, pretože sme predtým robili v C#, kde sme to prvýkrát vyskúšali a k tomu používame git, ktorý na to máme plne napojený, máme v tom postavené celé buildovacie pipeliny od alfa beta až po produkciu. Hneď po pushnutí do gitu ide commit automaticky na pull request, potom sa automaticky vytvoria buildy, na ktoré sa púšťajú automatizované testy. Potom build prechádza procesom schvaľovania od alfy do bety a následne stakeholder testuje produkciu, a tak ďalej. Je to relatívne veľký proces, v ktorom nám funguje z veľkej časti automatizácia.

Snažíme sa to stále zlepšovať, avšak sme na náš postup a proces hrdí. Devops používame, pretože má v sebe všetky spomenuté veci a sú automatizované. Často naviac naše tímy majú okrem jedného projektu ešte nejaký sekundárny projekt. V Devops umožňujú práve nielen pohľad na jednotlivý projekt, ale i pohľad na celý tím, čo nám vyhovuje a navyše má skvelo urobené štatistiky, čiže máme spätnú väzbu ako projekt a celý proces zdokonaľovať.

I nedávno sme vďaka tomu vyriešili problém. Stalo sa nám, že stakeholder sa pasoval do role produkťáka, čo nie je dobrá forma spolupráce. Čísla nás potom upozornili na to, že to nebolo v poriadku a stakeholder nebol dostatočne pružný prijímať výstupy na konci iterácií, takže nakoniec táto spolupráca nefungovala.

Moderátor: Jaké řešíte nejčastější problémy při zadávání úloh?

Ing. Martin Koperniech: V podstate nenarážame skoro na nič. Maximálne to môže byť problém so spomenutými kolíziami, závislosťami, kde nevieme, čo sa má riešiť skôr. Prípadne je niečo nesprávne popísané, ale to sa stáva málokedy, pretože máme vždy pred tým pomerne tvrdý grooming, následne sú subplaningy a technické planingy, takže sa to často nestáva. Ak sa to náhodou prihodí, riešime to tým, že máme na stand upe ”dogrooming”alebo doinformovanie. Problémy s predávaním úloh nemáme, pretože tímy máme striktne rozdelené a úlohy mimo konkrétny tím neskáču.

Moderátor: Takže s Vašmi zavedeným procesem jste spokojení a necítíte, že by nějaký proces potřeboval optimalizovat?

Ing. Martin Koperniech: Sme spokojní, jediná problémová vec je ľudský faktor. To znamená, že sa niečo zle popíše alebo zle pochopí. My do toho

behu máme zavedený tím testerov. Testing je súčasťou už i groomingu, pretože produkťák píše akceptačné kritéria, na základe ktorých sa potom píšu testy. Zároveň do toho vstupuje tester a kritéria upravuje. Produkťák a tester sú pre nás dva mantinely z oboch strán. Na konci každej iterácie musí byť výsledok a práve tester je tam od toho, aby vymýšľal a skúšal testovacie scenáre na základe use casov, čo nám pomáha a funguje to.

Takže jediné čo riešime je ako byť efektívnejší v odhadovaní alebo technickom plánovaní, ale priamo v procese problémy nebývajú. Samozrejme stane sa, že niekto nastaví, že úloha je dokončená, tester mu to potom vracia späť a stráca sa na tom čas, alebo si niekto vezme úlohu zo sprintu, ktorý nie je aktuálny, ale je nacenený, avšak to nie sú problémy procesu, ale naozaj tej ľudskej stránky. Problémy so zadávaním sme mali pokým sme mali ešte aj mobilný vývoj, ale to sme sa zhodli, že tým smerom ani ísť nechceme.

Moderátor: Kdo u vás ručí za úplnost a plnění úkolů?

Ing. Martin Koperniech: Záleží ako ktorá úloha. Máme feature, backlog items a potom sú tasks. Za task je zodpovedný ten, kto ho robí. Za backlog items je zodpovedný tester, ale samozrejme čiastočne i priamo vývojár. Za celý backlog produkťák. Nemáme vyložene scrum mastera, ale stará sa o to na každom projekte práve produkťák. On je zodpovedný za popis, zoradenie, plnenie úloh, za grooming a potom celý proces sleduje. Vývojári sú zodpovední za vývoj, testeri za to, že všetko funguje ako má a že je výsledok dobre otestovaný. Produkťák si to následne od testerov vezme späť a urobí ešte jeden build, na ktorom bežia regresné testy. Od toho sú potom na konci iterácie schôdzky, kde sa rieši, čo sa podarilo, čo nie a prečo. Každopádne tá zodpovednosť je teda taká „komunistická“, „socialistická“, ale najväčší tlak má určite produkťák, aby všetko fungovalo podľa požiadaviek a je ten, kto sa za to postaví.

Moderátor: Takže produkťák je ten, kdo sleduje průběh plnění úkolů?

Ing. Martin Koperniech: No áno i nie. Sleduje plnenie úloh a čaká až mu tester v backlogu označí, že je niečo pripravené. Určite to nie je vývojár a nevidí do vývoja. Na druhej strane každý deň sa s vývojovým tímom stretáva a baví sa s nimi, takže zároveň vie ako sa to programuje a ako sa to robí, takže tým získava know-how. Máme tu troch produkťákov, čo neboli vývojári a je u nich cítiť, že im tieto schôdzky a náhľad na vývoj pomáhajú sa vzdelávať.

Moderátor: Jak u Vás dochází ke sledování strávených hodin na úkolu?

Ing. Martin Koperniech: To práve Devops nemá, takže to nerobíme priamo v ňom, možno že aj má nejaký plugin, ale nevieme o ňom. Každopádne používáme Costlocker od ľudí z 2FRESH. Je na ňom kompletné trackovanie financií zákaziek. Pre vývojárov je to samozrejme trochu pain, že majú Costlocker aj Devops, na druhú stranu robia väčšinou len jeden projekt, takže ráno

pustia trackovanie, potom idú na obed a po ňom to zase spustia, takže robia len dva logy denne, čo nie je toľko práce. Costlocker robí aj naozaj krásne štatistiky, vidíme faktury a marže projektu, takže sme s ním spokojní. Používame ho i na interné projekty a porady, vďaka čomu vieme koľko nás stoja schôdzky a podobne.

Moderátor: Oba systémy máte nějak propojené?

Ing. Martin Koperniech: Zatiaľ nie. Costlocker má nejaké API, aktuálne ho máme napojený na slack, určite by to šlo napojiť aj na DevOps, ale doteraz sme tú potrebu nemali. Costlocker je český produkt, sme s tým fakt spokojní. Má to aj aplikáciu pre Apple, plugin do Chrome a ľudia sa s tým naučili pracovať, takže potrebu prepojenia oboch systémov nemáme.

Moderátor: Používáte nějaké další systémy? Nechtěli byste některé z nich spojit?

Ing. Martin Koperniech: Aktuálne ich ani veľa nemáme. Čo ďalej ešte používame je CRM pre schôdzky, pripomenie mi ich, trackuje ako dlho na akom meetingu sme a podobne. K tomu si pomáham excelom na financovanie, využívam kontingenčné tabuľky, ale žiadny veľký finančný systém na tento biznis nepotrebujeme.

Z CRM nám tiež vychádza mnoho štatistík a dá sa z toho veľa vecí vyčítať. Vyzerá to tam podobne ako v backlogu v Scrum. Vo chvíli, kedy sa dostaneme do nejakej fázy, tak to zadávame do Costlocker, kde to je zdĺhavé, pretože to robíme manuálne, ale zároveň jednoduché, čiže nie je nutná automatizácia. Na dokumenty používame Legito, ale to stojí trochu bokom. Určite sa tam dá zapracovať na automatizácii, ale všetky tieto systémy majú alerty, vyskakujú nám všade hlášky, takže všetko máme pod kontrolou.

D.3 Digital Ant (Ing. Štěpán Heller a Lukáš Obdržálek)

Moderátor: Mohli byste se krátce představit a zmínit jakou zastáváte pozici?

Ing. Štěpán Heller: Já jsem Štěpán Heller, tohle je Lukáš Obdržálek, společně jsme založili Digital Ant, takže jsme majitelé a zároveň firmu řídíme. Máme na starosti klientské projekty, řízení týmu a nějaké horečnosti, které jsou spojené s vedením firmy.

Lukáš Obdržálek: Štěpán se, pokud mluvíme o řízení firmy, více věnuje financím a právu. Já mám více na starost komunikaci a vývoj businessu.

Moderátor: Co vaše firma dělá? Na co se specializuje?

Lukáš Obdržálek: Jsme marketingová firma, zaměřujeme se na start upy, které vstupují na trh s něčím novým. Většinou začínáme tím, že si projekt ověřujeme na trhu přes nějakou ověřovací kampaň a pak agilně pracujeme na kampani abychom projekt, tedy většinou prodej produktu, škálovali na trhu.

Většina agentur se zaměřuje na zajeté firmy takže u nás je unikátní, že se zaměřujeme na projekty, které prochází nějakou změnou nebo na trh vstupují. Naše klientela jsou z velké části už dlouhodobě fungující firmy, ale přichází s něčím novým.

Moderátor: Jak je vaše firma velká. Kolik má zaměstnanců?

Lukáš Obdržálek: Máme tu 12 lidí včetně externí grafičky.

Moderátor: Jak máte u vás rozdělené týmy a kolik mají členů?

Lukáš Obdržálek: Týmy dva. Jeden se věnuje výkonností kampani, druhý se věnuje té kreativní stránce, tedy obsahu jako takovému. Dále se pak týmy rozdělují podle projektů. Vždy jsou tedy dva týmy v rámci každého projektu. Dále jsme tu pak já, Štěpán a Kristýna, která nám pomáhá s řízením projektů, takže my vlastně nejsme v žádném týmu.

Moderátor: Týmy máte takto striktně rozdělené nebo je více dělíte i podle projektu?

Ing. Štěpán Heller: Dělíme i podle projektů. Pokud přijmeme projekt, tak nejdříve sestavíme tým podle dovedností, které daný projekt potřebuje, pak podle kapacit. Takže ne, je to vždy podle toho, co daný projekt potřebuje.

Moderátor: Mohli byste popsat proces zadávání úkolů ve vaší firmě?

Lukáš Obdržálek: Na začátku je vždy briefing, kde je vždy přítomný člověk, který projekt řídí, člověk který má na starosti kampaně (to je člověk z výkonnostního týmu) a člověk z kreativního týmu, který má pak na starosti, co ta kampaň sděluje. Snažíme se tam pozvat i klienta, aby k tomu také mohl něco říct a abychom byli transparentní. Představujeme tam projekt, proč se dělá, stanovují se cíle, termíny, probíhá tam diskuze o tom, jak budeme kampaň dělat. Z toho pak odchází project lead, tedy projekťák s úkolem udělat brief, což je dokument, který obsahuje všechny zmíněné informace a ze kterého potom vycházejí ostatní členové týmu.

Potom se začne připravovat struktura kampaně na základě brainstormingu, která v tom týmu proběhne. Řeší se, co ta kampaň má sdělit, v jakých formátech, na jakých kanálech se má vytvářet. Také se řeší s jakým cílem se má kampaň dělat a na základě toho se připraví reklamy. Například uděláme reklamní příspěvky na facebooku, které se pak schvalují s klientem na dalším setkání nebo elektronicky. Pak se vše dle feedbacku dopracuje, nastaví se do kampaní a ty pak spustí.

Moderátor: Takže hlavní úkoly vychází přímo už z první schůzky?

Lukáš Obdržálek: Spíše jen plán projektu jako hlavní úkol pro projekťáka, který pak přiděluje dílčí úkoly týmu. My ten proces máme nějakým způsobem nadefinovaný, takže většinou všichni hned vědí, co mají dělat. Tím pádem tu nemáme úplně nějaké individuálně speciální úkoly, většinou ten proces je podobný nebo dokonce úplně stejný.

Moderátor: a jaké typy těch dílčích úloh vznikají?

Lukáš Obdržálek: Typy úkolů tu úplně moc nemáme. Samozřejmě se úkoly dají dát do nějaké kategorie. Část úkolů spadá do technické přípravy té kampaně, to dělá ten výkonnostní tým, to jsou úlohy jako například udělat media plán, tedy jaké kanály budeme používat. Na to pak navazují ty kreativní úkoly nebo projektové úkoly.

Projektové úkoly jsou například: projetí, zdali ten plán odpovídá tomu, co projekt potřebuje z produktového pohledu a projednat to s klientem. Kreativní tým pak zase řeší, že mám dané kanály a formáty a na základě toho musí vymyslet myšlenku té kampaně, rozpracovat to, pak to zase projednat s výkonnostním týmem a s projekťákem. Jsou to tedy tři hlavní okruhy a ty jsou přesně rozdělené do těch týmů.

Moderátor: Kdo je u vás za proces zadávání úkolů zodpovědný?

Lukáš Obdržálek: Samozřejmě za každý úkol je zodpovědný ten, kdo úkol dělá, takže když něco k úkolu neví, měl by se doptat, ale zodpovědnost za plnění celé kampaně před klientem má ten produkt lead.

Moderátor: Obecně se tedy asi nedá říct, že se držíte některých z agilních metodik?

Lukáš Obdržálek: Ne, to ne. My moc nevěříme na ty dlouhodobé kampaně, které se spustí a pak se vyhodnocuje, jestli to něco dělá nebo nedělá. My postupujeme po menších částech a postupně ten marketing škálujeme. Takže stále postupujeme dopředu, někdy je tam slepá ulička, musí se něco předělat a tak dál, to je to, čemu říkáme agilní marketing. V tom projektovém řízení to ale nespadá do žádné metodologie, to určitě ne. Na trhu fungujeme už 7 let, takže jsme si za tu dobu ověřili, co nám funguje, a co ne a máme na to takhle vlastní proces.

Moderátor: Jak project lead sleduje plnění úkolů?

Lukáš Obdržálek: On vlastně rozdá úkoly a pak si musí pohlídat, že jsou úlohy splněné. Primárně nesleduje tým, jak na úkolech dělá, ale máme každý týden schůzky o statusu toho, co se děje, takže tam se dozví, jak na tom kdo je.

Ing. Štěpán Heller: Zároveň taky máme podpůrný systém Asana. Tam projekťák všechny úlohy zadá a pak vidí, jestli jsou hotové, jestli nejsou po deadlinu nebo zdali na nich někdo začal dělat. Asana posílá notifikace, takže přehled o tom má právě z Asany.

Moderátor: Jak řešíte priority úloh?

Ing. Štěpán Heller: U nás má každý projekt vlastně lineárně danou posloupnost úkolů. Takže priority můžeme určovat až ve chvíli, kdy máme projektů víc. Většinou je to podle významnosti klienta, sekundárně je to pak podle termínu. Jelikož nás není moc, tak to není tak složité.

Moderátor: Jelikož jste na procesu pracovali 7 let, mohli byste vyjmenovat nějaké největší problémy či zdroje neefektivity, se kterými jste se při zadávání úloh setkali?

Ing. Štěpán Heller: Problémy určitě byly u těch komplexnějších úkolů. My jsme dříve používali Trello, kde když se úkol rozpadá do mnoho podúkolů a má různé vlastníky, tak tam už dobře nefunguje to sledování termínů a kvantifikace. Takže se často stávalo, že si člověk nevšiml, že má přidělený úkol, nebo že ten úkol má nějaký deadline. To jsme vyřešili přechodem na Asanu.

Lukáš Obdržálek: Je pravda, že se tou Asanou byl opravdu přelom. Ono do chvíle, kdy jsme byli tři, tak se tohle dalo snadno uhlídat v tom Trello, ale pak už to bylo mnohem těžší. Šlo o to, že jsme tam měli kartu, ta karta mohla mít termín, kdy má být dokončena, nějaké stavy a přiřazeného člověka, ale pod tím byly úkoly, které tohle už nemohly mít. Takže člověk se pak nemohl podívat zároveň na všechny svoje úkoly, to tam neexistovalo, takže si to musel proklikat.

Moderátor: Řešíte nějaké problémy i dnes?

Lukáš Obdržálek: No typicky se nám stává, že někdo má přidělený úkol, a ten má due date. Když ti ten úkol někdo přidělí a ty se na něj nepodíváš, tak si ho všimneš většinou až ve chvíli, když se blíží ten due date, nebo když už nastal a v tu chvíli zjistíš, že je ten úkol mnohem složitější než to vyplývalo z názvu. Tam pak vzniká samozřejmě problém. Ona to řeší placená verze Asany, kterou nemáme a má tam na to nějaké možnosti.

Moderátor: Takže časové odhady do úkolů nezahrnujete?

Lukáš Obdržálek: Ne, ty tam nedáváme. Ona to Asana v základní verzi ani nemá, jde to obejít přes nějaké tagy, ale to úplně není ono.

Moderátor: Jak tedy celkově řešíte sledování času u úkolů a s tím spojené financování projektu?

Lukáš Obdržálek: No, my ty časové odhady samozřejmě máme, ale pouze interně. Kromě Asana pak používáme Costlocker, který nám slouží k tomu, abychom ten projekt uřídili finančně. To znamená, že tam dáváme odhady na typy projektu na nějaké úkoly. Ale není to o jednotlivých úkolech. Prostě interně víme, že třeba připravit kampaň bude trvat 25 hodin, ale neřešíme to, zdali analýza klíčových slov bude trvat 2 hodiny nebo 5 hodin. Takhle nám to aktuálně z finančního pohledu stačí. Pak je tu právě projekťák od toho, aby hlídal, jak dlouho týmu trvá zpracování daného úkolu a finančně za to pak ručí on. Každopádně do Costlockeru má přístup každý, takže pak vidí, kolik je na jeho činnost přiřazeno času, takže by pak měl i on sám upozornit projekťáka, že to nezvládne za daný počet hodin. Nemáme tam žádnou striktní hranici jako třeba že pokud se blížíš 80%, tak to musíš říct.

Moderátor: Je na podpůrném systému Asana něco, co vám na ní chybí, co se vám na ní nelíbí?

Lukáš Obdržálek: Vše co potřebujeme, to má. Jsou tam samozřejmě ty placené věci, které jsou nice to have, ale vyloženě je nepotřebujeme. To je například ta náročnost úkolů, kterou jsme dřív obcházeli psaním do názvu toho úkolu, ale to není ono. Případně nějaký očekávaný start date úkolu, aby ten, kdo se na úkol podívá, věděl, kdy by na úkolu měl začít dělat, aby se vše stihlo v pohodě.

Moderátor: Zapisování hodin probíhá tedy přímo přes Costlocker?

Ing. Štěpán Heller: Ano, tam to funguje stejně třeba jako v Toggl. Člověk se přihlásí, má tam konkrétní aktivity, a když na tom začne dělat, vybere si aktivitu a začne si trackovat čas.

Moderátor: Costlocker máte s Asanou nějakou propojenou?

Lukáš Obdržálek: Ne, máme to separé. Ono popravdě to u nás není potřeba, protože bychom pak v Asaně museli mít 100% všech aktivit, což u nás úplně neděláme. Bylo by dle nás zbytečný si každou aktivitu, která je ad-hock dávat do Asany. Takže tuhle potřebu jsme zatím neměli.

Moderátor: Využíváte v Asaně všechny funkce, které tam jsou? Jste se systémem spokojení?

Ing. Štěpán Heller: Všechny věci určitě ne, ono tam toho zase tolik není, protože nemáme tu placenou verzi. Nám to takhle stačí a jsme s tím spokojení. Nevyužíváme moc štítky. Co nám tam chybí a co je v té placené verzi jsou závislosti. To se u nás docela hodí, protože máme ty úkoly lineárně řazené, takže se by bylo fajn mít možnost říct, že je nějaký úkol závislý na nějakém jiném.

Lukáš Obdržálek: Hezký by byl systém, který jsem nedávno někde viděl. Fungovalo to na základě nějaké umělé inteligence, která vyhodnocovala, jaký úkol mám dnes dělat. Vlastně že se to přizpůsobuje jak deadlinům, rychlosti práce, tak vlastně i chuti lidí. Ale pokud vím, tak ten systém je strašně drahý a nevím jak moc velký přínos to má.

D.4 2FRESH (Lenka Kůrková)

Respondentovi bylo v rozhovoru tykáno na jeho žádost.

Moderátor: Mohla bys ses krátce představit a zmínit jakou zastáváš pozici?

Lenka Kůrková: Jmenuji se Lenka Kůrková a do 2FRESH jsem nastupovala jako šéf projekťáků. To znamená, že jsem měla na starosti nejen ten projektový tým, ale i proces odbavování zakázek v 2FRESH. Nyní jsem na pozici chief operations officer, takže mám na starosti takový běžný chod agentury, zároveň mi ale zůstal proces řízení zakázek. Tím že se teď 2FRESH rozdělil na 5 divizí tak jsou to primárně cross-divizní zakázky.

Moderátor: Proč jste se rozdělili?

Lenka Kůrková: Byli jsme klasicky agentura, která měla reklamní část a produktovou část. Reklamní část klasicky zaměřenou na komunikaci a produktová část na vývoj produktu. Webovky, aplikace, redesign, bankovnictví, dělali jsme cokoliv digitálního.

Problém byl ten, že celá firma jela klasickým vodopádem, to znamená proces jel od a přes několik desítek písmenek, aniž by na konci bylo Z. To začalo vadit u velkých projektů, které potřebovaly hodně lidí.

U jiných agentur (dodavatelů) to bylo tak, že se na začátku udělal nějaký výzkum a UX, následně se udělal UI, klasicky design a pak se udělal klasicky vývoj. Tady to fungovalo trochu jinak. Nebyl tu vývoj a my jsme spolupracovali s jinými firmami. Nejdelší dobu u nás typicky trvá UX, celý výzkum, dotazníky, rozhovory s respondenty a tak dále. Vyšla z toho velká dokumentace, popis produktu, wireframes a podobně. No a ostatní lidi kvůli tomu hrozně dlouho čekali, než se k nim tenhle dokument dostane. Nebyl tu žádný Scrum nic podobného, aby spolu ti lidi mohli spolupracovat. Cílem rozdělení do divizí tedy bylo, aby každá ta jednotka (UX, UI) mohla pracovat sama, mohla přijímat vlastní zakázky.

Moderátor: Co tedy vaše firma dělá a kolik má zmastnanců?

Lenka Kůrková: Z těch divizí máme reklamu, která byla oddělena dříve, pak UX, UI, design consulting a pak to byl i dev, který teď už není. a to je vlastně i odpověď na to, co naše firma poskytuje za služby. Zaměstnance jsme nedávno počítali a je jich přesně 40.

Moderátor: Divize pro vás znamená i rozdělení týmů?

Lenka Kůrková: No, reklamu asi oddělím, protože ta jde trochu bokem a je to taková pomalu vlastní firma s velkým množstvím malých týmů, sice taky řeší projektové řízení, ale je to dost specifické. Ale dále v UI je to 5 lidí, v UX je to asi 8 lidí, takže tam to tak je.

Moderátor: Nějaké další členění v rámci těchto týmů nemáte?

Lenka Kůrková: V UI je to poměrně jednoduché, je tam business unit director, který zastává spoustu funkcí na tak malý tým, protože ostatní v týmu jsou designeři. Tenhle člověk zastřešuje celou divizi – řeší obchodní část, kde shání zakázky, následně se z něj stává account ve chvíli, kdy zakázku sežene, řeší s klientem business a pak je z něj vlastně i project manager. No a do toho ještě řeší HR věci a tak dále, takže to je místo, kde bychom se rozhodně měli rozrůst, protože tam chybí ruce. V tuhle chvíli sháníme do každé divize právě projektoví manažeři, kteří tu dřív byli, ale pak se zrušili, protože zakázek nebylo dostatek, měli vlastní autonomii a nemalé cíle. Teď už je to ale zase tak, že zakázek je spousta a chybí nám ty ruce, protože business unit director pak nezvládá rozjet business. Hledáme například projektového manažera do reklamy, který bude ale zároveň řešit cross-divizní zakázky.

V UX týmu je to trošku jinak. Nemají tam klasickou hierarchii, je tam tým UX designerů a jsou schopní si ten projekt odvést úplně sami. Například si představ, že pracují na redesignu pro Škodovku, třeba mobilní aplikace. Zaštituje jim to opět business unit director, ale ve chvíli, kdy ten projekt přijde sem, tak si ho převezme ten designer a řeší zpracování nabídky, jak budeme postupovat při zpracování té nabídky a má k dispozici případně ještě jednoho designéra, který mu s tím pomáhá. Nemají tam teda projekťáka, ale spolupracují s ním více jako partneři.

Proces v UX a výzkum je opravdu dlouhý. Je u toho vždy i klient aby pochopil, proč se jaký krok dělá a tak dále. Takže tam sice taky sháníme projektového manažera, ale spíš juniorního, který bude pomáhat s alokacemi lidí – tedy kdy bude každý dělat co, fakturace a administrativa než řízení projektů.

Moderátor: Takže u vás nemáte rozdělené týmy na nějaké menší jednotky například dle projektu?

Lenka Kůrková: Ne, nemáme tu nic jako je třeba ve Scrumu. Například v reklamě to tak funguje. Když přijde projekt, sestaví se tým, ale v těch ostatních divizích to tak úplně není. Je to tím, že ta reklama je opravdu velká a mají tam velké strategie. Jsou tam kreativci, co vymýšlí myšlenku, texty, pak nějaký social stratég a nakonec je tam produkce, lidi od videa, přes fotografy nebo nějaké designéry. V reklamě ale taky mají daleko větší vytížení, zatímco v UX a UI se projekty táhnou delší dobu, tak u nich to je day-to-day.

Moderátor: Mohla bys popsat proces zadávání úloh?

Lenka Kůrková: Zkusím to vzít hodně obecně. Přijde nějaká poptávka, například nabídka v tendru, tak přijde člověk a řekne pro jakou divizi to zhruba je. Napadá mě třeba typický příklad redesign je typicky spolupráce UI a reklama, protože to nám chodí nejvíc.

V tu chvíli si určíme, co přesně pro to potřebujeme a co to obnáší. Tyhle cross-divizní projekty jsou pak právě špatné, protože máme více projektových nástrojů, kde odhadujeme zdroje, ale to odbočuji. Takže je potřeba určit člověka, který ten projekt odvede. V ideálním světě by to měl být ten interní projektový manažer a určí se, kde je zodpovědný za tu zakázku a v jaké fázi. Dokud děláme výzkum, jak se značka vnímá, tak je to v režii reklamní divize a ve chvíli, kdy se začíná rozkreslovat a řešit design, už je to v režii UI. Ideální by bylo, aby tam byl jeden account, který hlídá finance a domlouvá podmínky a pak je tam projektový manažer a řeší kolik potřebujeme času, do kdy to stihneme a proč. Řídí si nejen alokace, ale i podklady pro lidi, kteří něco potřebují, ale zároveň je to i parťák v týmu, který nečeká, kdy mu někdo něco pošle, ale i pomáhá sám od sebe.

To bylo v ideálním světě, ale my nemáme ty projektové manažery. Takže většinou tu roli projektového manažera zastává právě ten Business Unit Director, což není ideální, protože mají dost jiné práce a pak se ta práce zadrhává.

Moderátor: Takže zodpovědnost za celý projekt je na tomto člověku?

Lenka Kůrková: Ano, bohužel ano. On je opravdu ze začátku takový „otloukánek“, protože na něm leží všechno, od toho jestli se projekt stíhá, tedy řeší kapacity, kolik na to potřebuje kdo času, přes plnění a finance. Zároveň je v úzkém kontaktu s klientem i se specialisty, aby věděl jestli se stíhá nebo ne.

Tím pádem je zodpovědný i za celou delivery, takže když se něco pokazí, tak se dostatečně neptal specialistů nebo klienta. Pak nejhorší jsou samozřejmě technické projekty, když selže technika, to je věc, která se neovlivní, ale je potřeba pak v projektovém plánu počítat s nějakým buffrem, kdyby se to stalo.

Moderátor: Takže z metodik využíváte Waterfall?

Lenka Kůrková: No, takhle. Ono se tu teprve nedávno vůbec zavedl nějaký proces. Dřív to bylo tak, že se tu prostě jen rozhazovala práce jako seno, ale ve chvíli, kdy nás začalo být víc, tak se potřeboval zavést proces. Přišli jsme s klasikou – Waterfallem, protože nám to tak dávalo smysl a byli jsme rádi alespoň za to. Zároveň přistupovat na nějaká striktnější pravidla jako třeba ve Scrumu pro nás tehdy bylo nemyslitelné, když nebyla firma původně vůbec projektově řízená.

Asi záleží na typu projektu. Třeba v UX se docela svým procesem Scrumu podobají, v UI si myslím, že to zase moc smysl nedává, protože pracuje většinou

jen jeden designér a není potřeba nějaké větší spolupráce. Problémové jsou určitě ty větší projekty, ale ty začaly prakticky teprve až teď a já si myslím, že je pro přechod na něco jiného nejdřív mít tu potřebu a ta tu aktuálně úplně není, takže jsme rádi za Waterfall, ale možná časem přejdeme na něco jiného.

Moderátor: Jak probíhá nacenění projektů?

Lenka Kůrková: Hodně velkou otázkou pro nás byly rozpočty, protože zrovna u nás byl problém klientovi vysvětlit, proč musí platit projekťáky, manažery a podobně, když po nás chce jen reklamu nebo něco naprogramovat, ale to už se nám docela povedlo.

Nacenění probíhá na základě projektového plánu. Na začátku každého projektu je kick-off. Máme tam brief od klienta, který si každý přečte. Sedneme si a jsme tam všichni, kteří na daném projektu pracují a bavíme se o jejich přístupu, co bychom měli udělat a jak k tomu přistoupíme. V tuhle chvíli je tam už tedy nějaký ten projekťák, který to zaznamenává, připraví si poznatky, učeše si to podle toho, jak potřebuje a pak si obejde ty jednotlivé specialisty a začne vytvářet ten projektový plán.

Na začátku je vždy nějaká analýza, co přesně budeme zkoumat. Budeme se koukat do analytik, jestli budeme mluvit s lidmi a tak dále. Tam je důležité, že máme už zajeté některé postupy, všechny projekty jsou jiné, ale často je ten postup podobný.

Někdy jsou ty přístupy lehce odlišné, ale často jsme schopni si ty projektové plány nastřelit v hlavě. Například v reklamě máme udělanou tabulku v Excelu, kde vybíráme, co přesně budeme dělat za kroky.

Moderátor: Tím tedy řešíte problém se stanovením postupu?

Lenka Kůrková: Nemyslím si, že by to byl běžně úplný problém, spíš zrovna v reklamě jsou často mladí juniornější lidi, kteří ty procesy nemají ještě tolik v hlavě, spousta lidí ty typy projektů tolik nezná, neumějí tu práci úplně dobře odhadnout a spousta věcí pro ně byla nových a tohle jim pomáhá si to ustanovit.

Dřív ten proces byl totiž hrozně složitý a veliký a jel se u nás hrozně mikromanažerský přístup, kde každý do toho mohl mluvit a každý hlídal každého, nebyly dány priority, role, což jsme vyřešili ustanovením procesů, kde nám pomáhá ten kick-off a zároveň je tam ten chybějící článek toho projektového manažera.

Moderátor: Jak jste stanovení procesu řešili?

Lenka Kůrková: Prostě jsme si uvědomili, že to tak dál nejde, tak jsme si sedli a vymýšlely, co jsou ty nejnutnější kroky k tomu, aby se ten projekt dokázal odvést, tím jsme si vymezili například to, že kreativci potřebují čas na přemýšlení, který do toho moc zahrnutý nebyl a podobně. To nám pomáhalo

právě i při těch rozpočtech, protože díky tomu i klient může pak přesně vidět, co si zaplatil a nezapomínáme na žádné kroky.

Moderátor: Jak u vás řešíte priority jednotlivých úkolů?

Lenka Kůrková: To je těžké. Záleží jak u čeho. Když je ten projektový plán, tak je jasné, že dokud se nestane tohle, tak se nemůže stát tohle. U takových projektů to prostě musí jít chronologicky, prostě jedeme klasický Waterfall. Ale když se bavíme takhle s team leaderama a tak dále ohledně priorit a odbavování jejich práce (některým team leaderům ani nesledujeme čas, protože dělají spoustu práce, která se moc neplánuje), tak tam řešíme priority podle deadlinu. Úkoly, které mají deadline a jsou pro klienta jsou priorita číslo 1, úkoly, které jsou interní a mají deadline jsou priorita číslo 2, klientské úkoly bez deadlinu priorita 3 a interní věci bez deadlinu priorita číslo 4, ale to se moc nestává, protože my moc bez deadlinů nepracujeme.

Moderátor: Jak do toho zapadají úkoly typu issue nebo bug pokud něco takového máte?

Lenka Kůrková: Jo, to samozřejmě máme, to se ale často prioritizuje s klienty. Samozřejmě pokud je to třeba že otevřu web a ten nefunguje, tak je jasné, že je to okamžitě prioritní. Bereme to tak, že máme stanové cíle a pokud je nějaká překážka v tom, aby ten cíl byl naplněn, tak je to vždycky odbavené s prioritou číslo 1. Naopak, pokud nám přijde, že někde nesedí rozlišení nebo je tam nějaká malá grafická chyba, že to není pixel-perfect, tak to zas má prioritu úplně nejmenší.

Moderátor: To je docela dost pravidel. Tohle řeší každý sám, nebo na to máte nějaký systém, kde ty priority spravujete?

Lenka Kůrková: No, například v tom devu, který už teď nebude fungovat, jsme neměli toho projektového manažera a tam právě chyběl on, který by tohle přesně dělal, a pak to byl velký problém. Projektoval si to pak opravdu vývojář a bylo to peklo. Jako systém mám pocit, že používali Jiru.

Na řízení ostatních projektů používáme Notion, což prakticky ani není žádný nástroj na řízení. Je to prakticky takový dokumentový systém, který je fajn a má spoustu udělátek. Ten nám v těch dokumentech s řízením pomáhá hodně. Jedna taková typická funkce je takové klasické Trello. Máme popsané „Brief“, „In progress“, „Done“, „Ke kontrole“ a posouváme si projekty. Projektový manažer pak řeší přesun těch úloh.

Moderátor: Jak u vás probíhá trackování hodin?

Lenka Kůrková: Je to strašný peklo. Řešíme to neustále a je to poměrně čerstvé téma. Původně jsme používali náš systém – Costlocker, který měl stopky. No a my jsme si teda opravdu pomocí stopek trackovali čas na minuty.

Když šel člověk na kafe nebo na záchod, tak to vypínal a pak u něj nebylo skoro možný za den vykázat 8 hodin práce.

To se naštěstí všechno změnilo. Máme alokované lidi podle projektového plánu, lidi si nezapisují čas po minutách. Dostali jsme ale k tomu, že si všichni zapisovali jen bi-level projekty, což taky bylo špatně, protože pak nám ve statistikách vycházeli, že jsou jakoby „nevytížení“ takže jsme je pak museli donutit psát si to i třeba na tendry nebo interní projekty nebo na další věci jako „učím se“. To jsme zavřeli a nechali jsme jen bi-level projekty, interní projekt a pak už jen dovolené a waiting time (= nemám práci). Nejtěžší na tom bylo lidi přinutit, aby si to zapisovali na to waiting time, protože se báli, že je za to vyhodíme.

Moderátor: Co jste tedy podle tebe řešili za největší problém s procesem zadávání úloh? Co je podle tebe zdrojem neefektivity?

Lenka Kůrková: Nejvíc nás pálí asi ta absence projektového manažera, to je samozřejmě problém, ale z pohledu procesu asi vůbec to, aby lidi aktuálně ten proces přijali za svůj. Velký problém bylo i to trackování času, to jsme řešili dlouho a stálo nás to peníze a velký problém je pořád v lidech. Snažíme se jim třeba vštípit nějaký mindset a říct, proč je to důležité a často se stane, že třeba řeknou, že to tak prostě dělat nebudou, protože jim to nedává smysl. No a poslední problém je asi projektový nástroj – EasyProject. Ten systém je hrozně předimenzovaný, byla tu spousta lidí, která když to viděla, tak zase z firmy odešla. Ten systém se tu zavedl, aniž by se někoho zeptali, co vlastně potřebujeme za nástroje a pak vznikl tenhle problém.

Moderátor: Jak byste si představovali ideální nástroj?

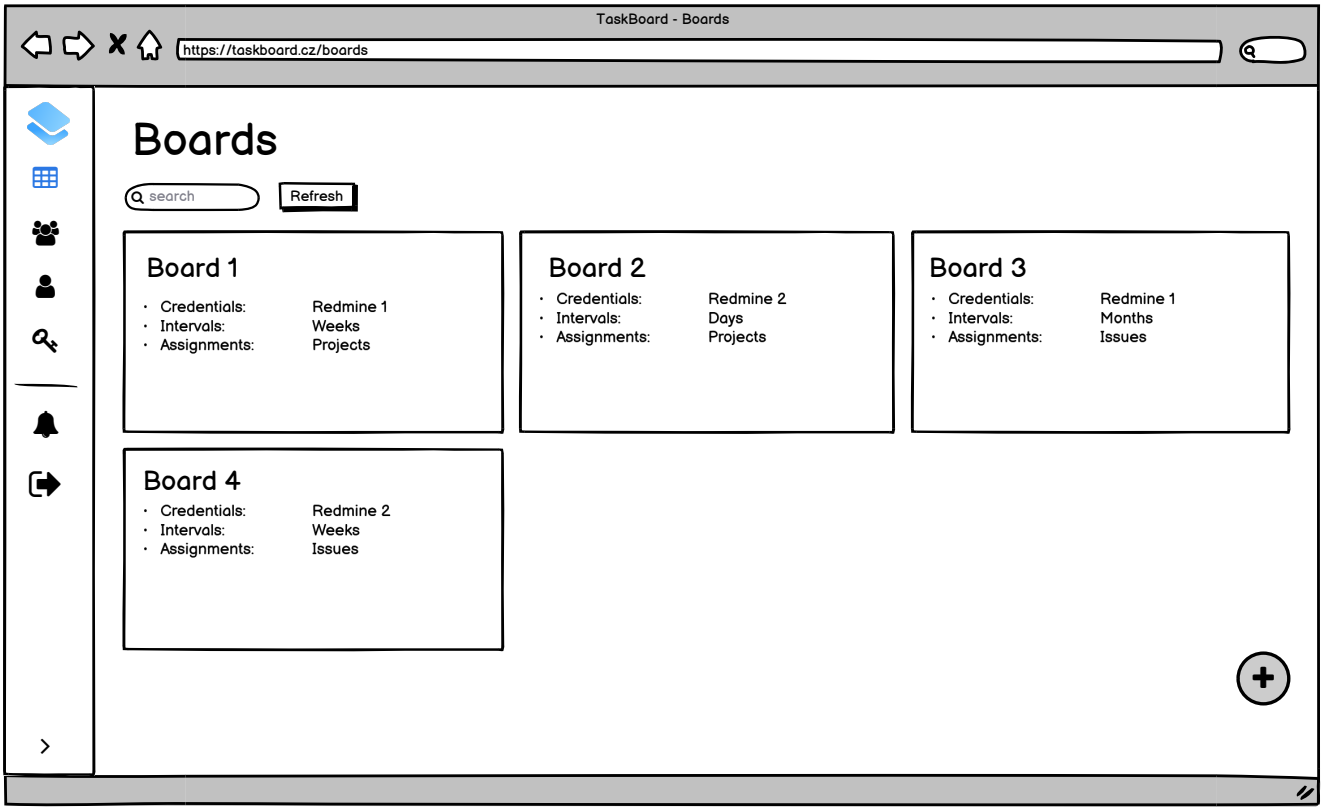
Lenka Kůrková: My vlastně potřebujeme jen pár věcí: nástroj na alokaci zdrojů, hlídání kapacit a z toho se odkazovat do toho Notion, kde lidi vidí, co mají dělat. No a taky jsme se ptali lidí z firmy, jak by to chtěli a všichni nám řekli, že jako dřív, aby měli práci ukázanou v kalendáři včetně schůzek. Takže jsme na to našli nedávno nový software Float, který chceme všude zavést a ideální stav, kam chceme dojít je, že account má Costlocker, kde vidí finanční stav, projektový manažer má Float, kde si lidi rezervuje v čase a vidí, kdy se má co stát a specialisté vidí svoje úkoly v kalendáři a pro víc informací si otevřou Notion.

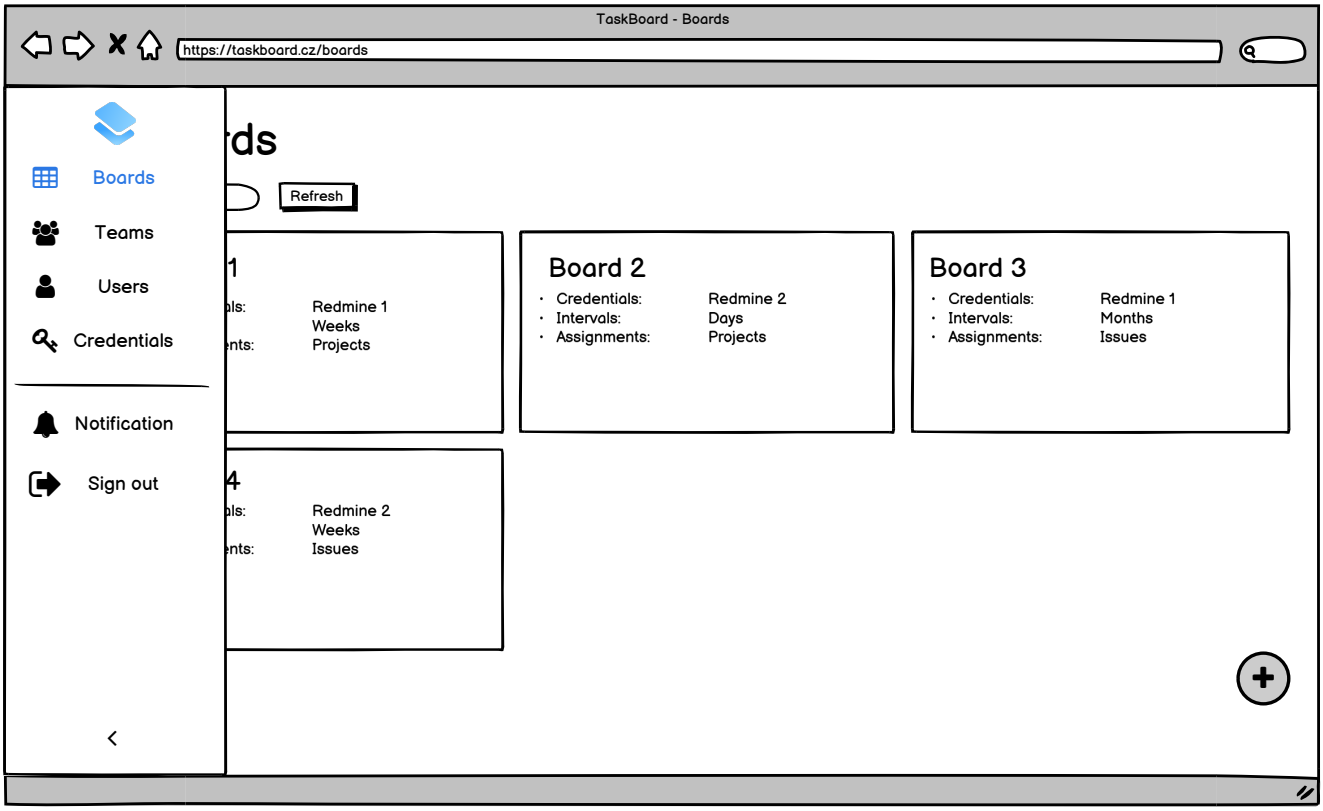
Moderátor: Daly by se nějaké tyhle systémy spojit?

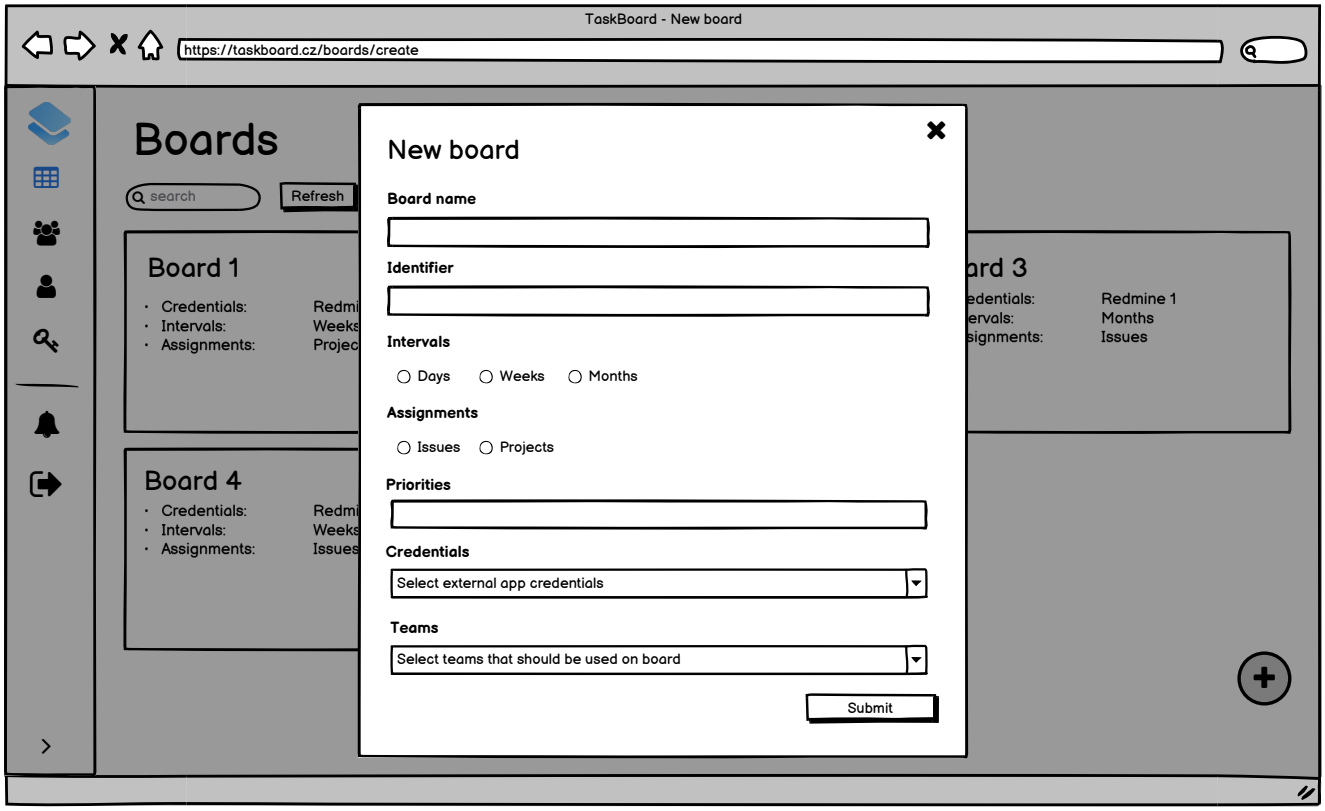
Lenka Kůrková: Pro mě by bylo skvělé, aby se ten booking zdrojů dával přímo do Costlockeru a bylo by to úplně skvělé, ale ač je to náš produkt, tak ho chtějí nechat tak jak je, protože má sloužit pouze k tomu jednomu účelu, co má teď.

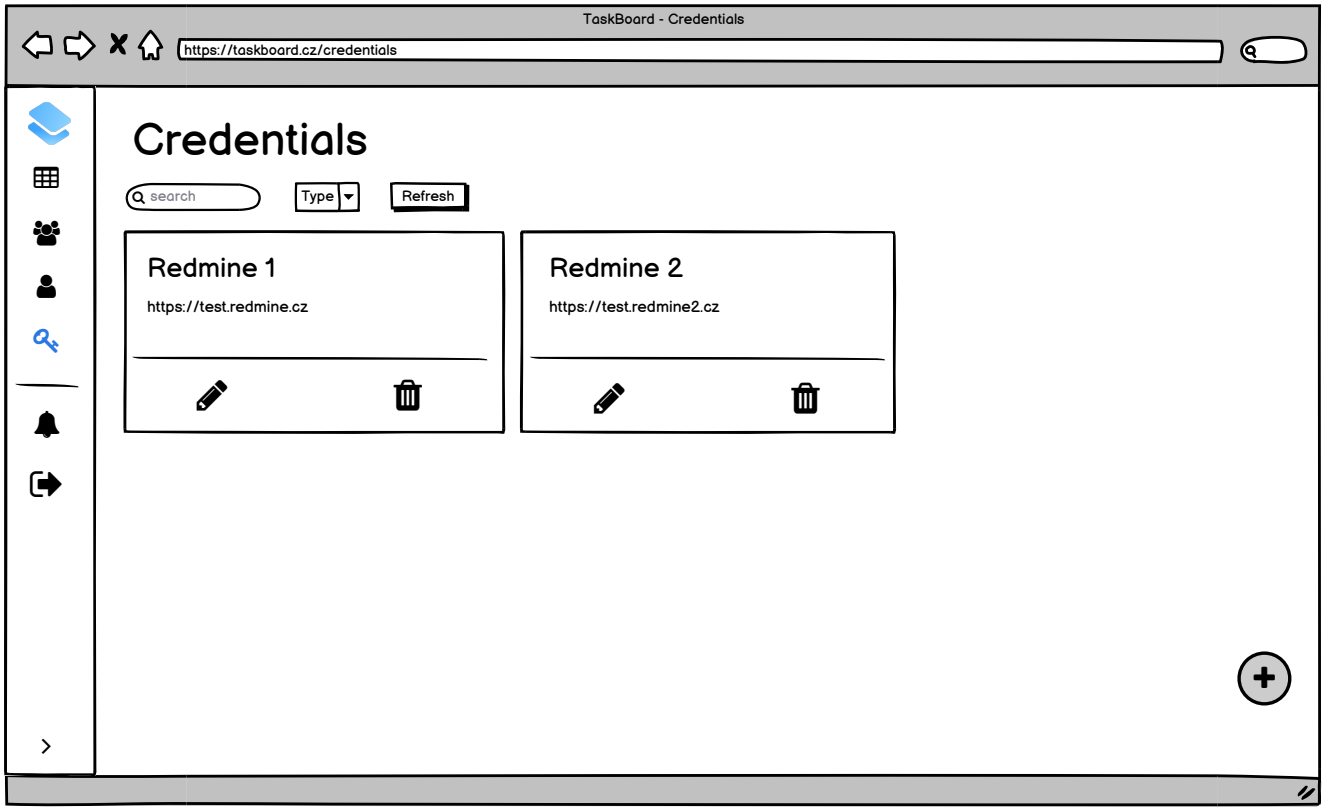
Příloha E

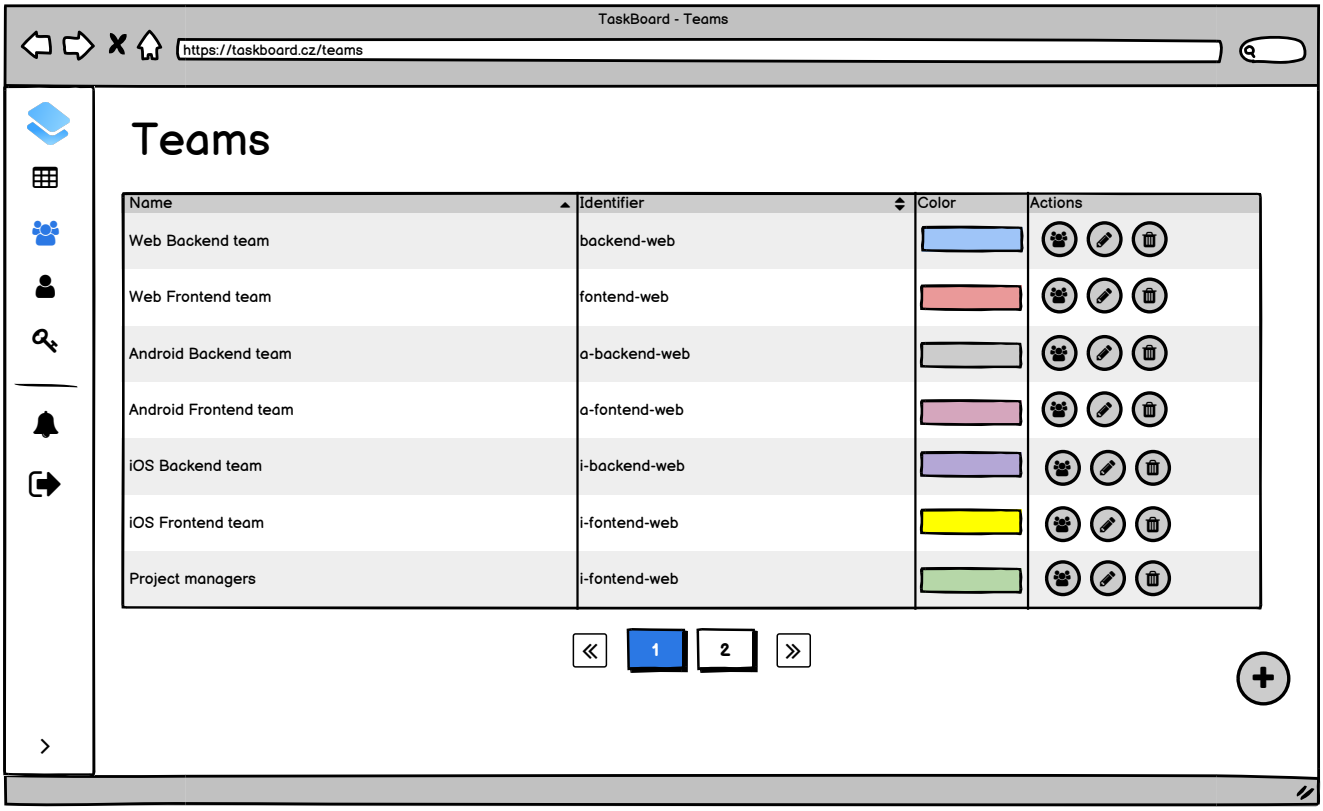
Wireframy pro návrh uživatelského rozhraní

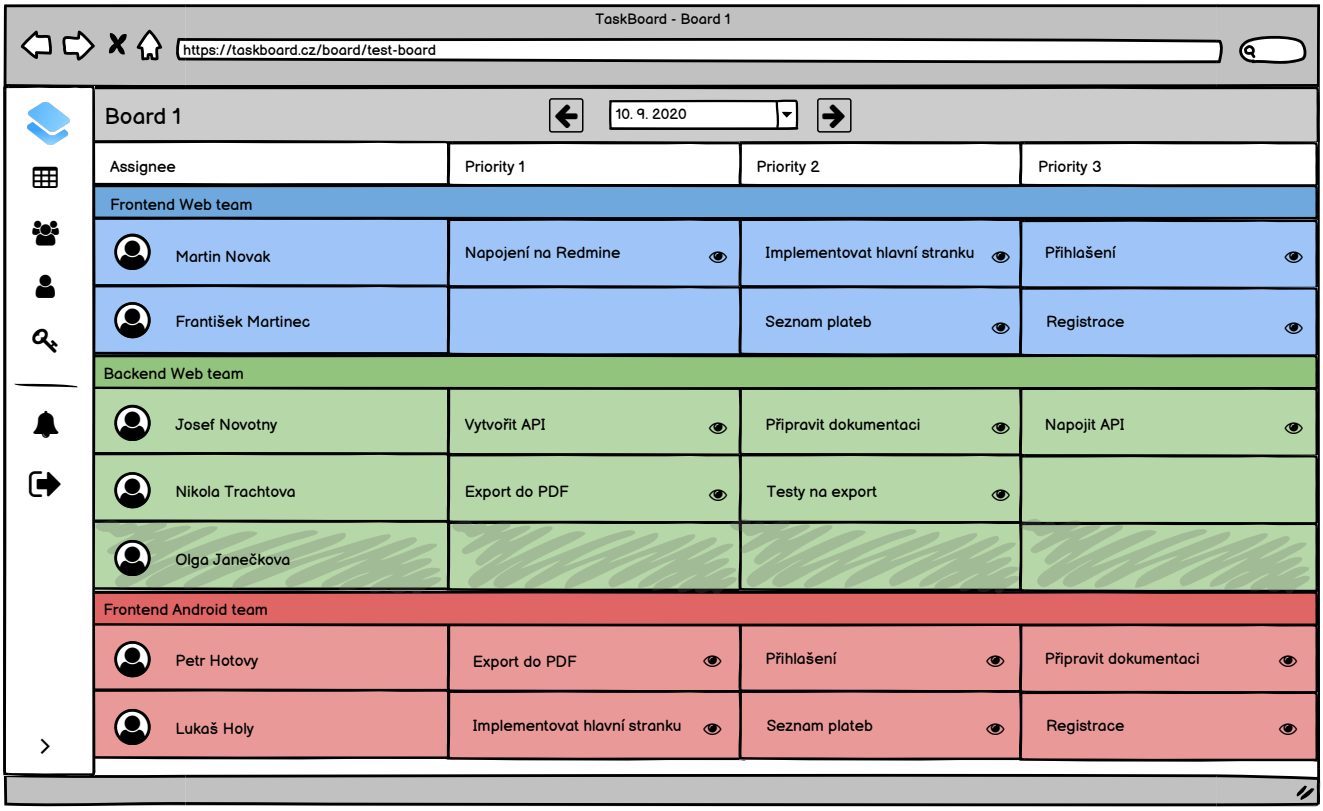

TaskBoard - Boards
https://taskboard.cz/boards
Boards
search
Refresh
Board 1
Credentials: Redmine 1
Intervals: Weeks
Assignments: Projects
Board 2
Credentials: Redmine 2
Intervals: Days
Assignments: Projects
Board 3
Credentials: Redmine 1
Intervals: Months
Assignments: Issues
Board 4
Credentials: Redmine 2
Intervals: Weeks
Assignments: Issues

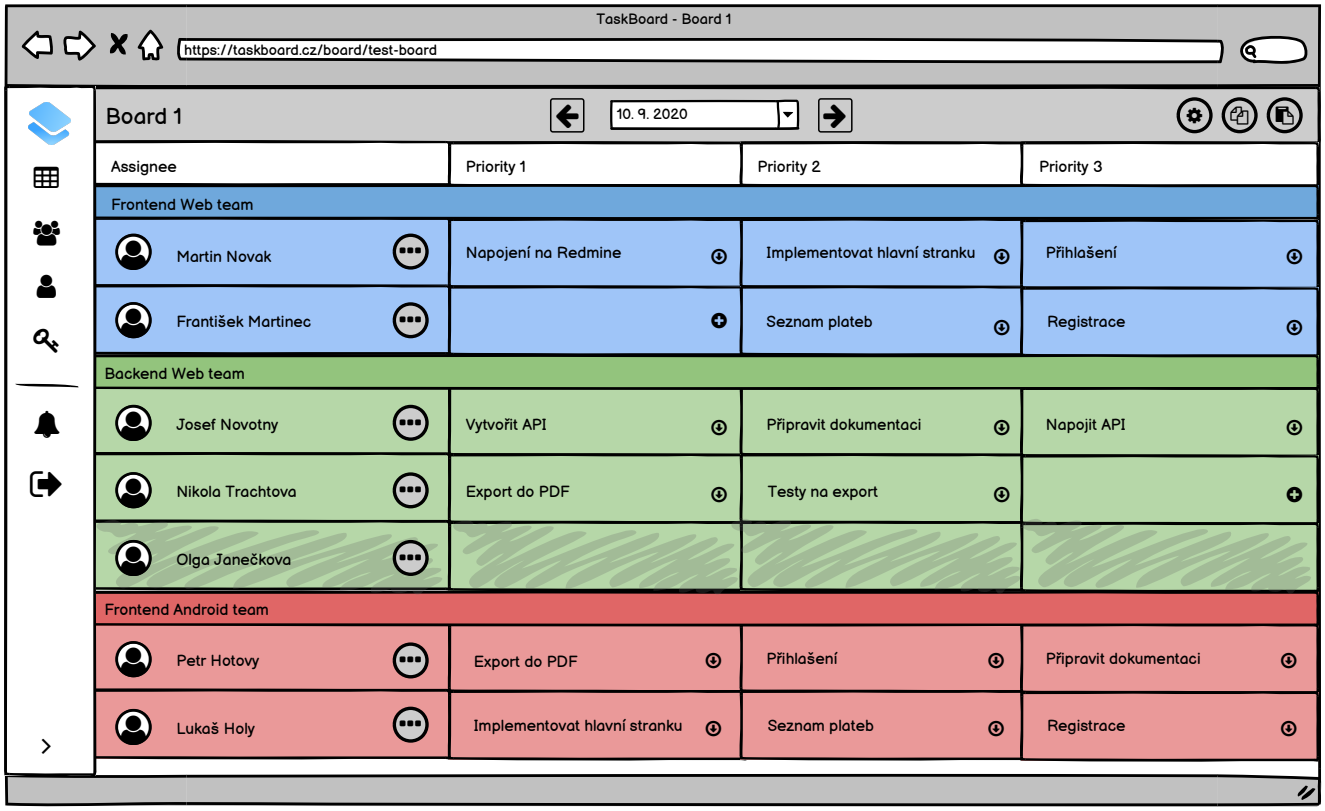

TaskBoard - Boards
https://taskboard.cz/boards
Boards
Teams
Users
Credentials
Notification
Sign out
Refresh
Board 2
Credentials: Redmine 2
Intervals: Days
Assignments: Projects
Board 3
Credentials: Redmine 1
Intervals: Months
Assignments: Issues

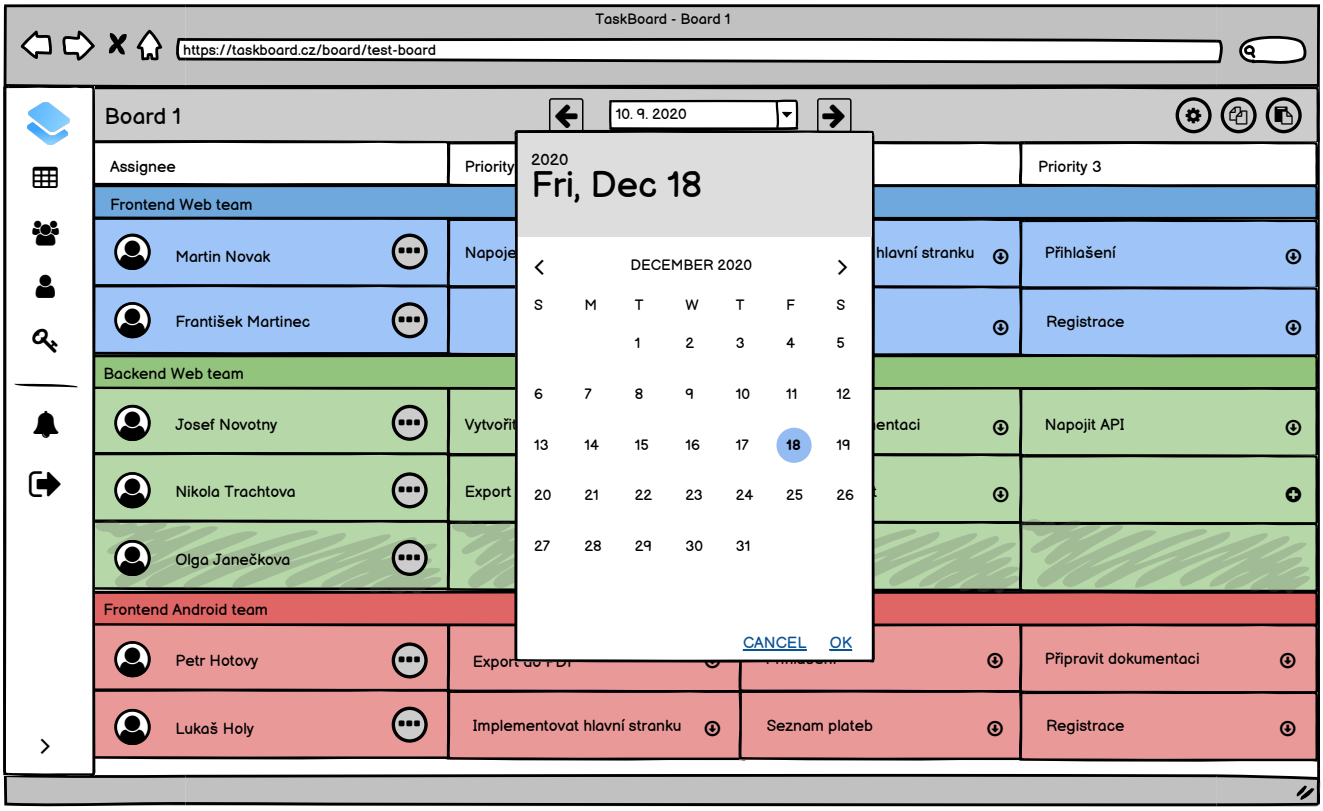

TaskBoard - New board
https://taskboard.cz/boards/create
Boards
search
Refresh
Board 1
Credentials:
Intervals:
Assignments:
Board 4
Credentials:
Intervals:
Assignments:
Issues
Redmine 1
Months
Issues
New board
Board name
Identifier
Intervals
Days
Weeks
Months
Assignments
Issues
Projects
Priorities
Credentials
Select external app credentials
Teams
Select teams that should be used on board
Submit

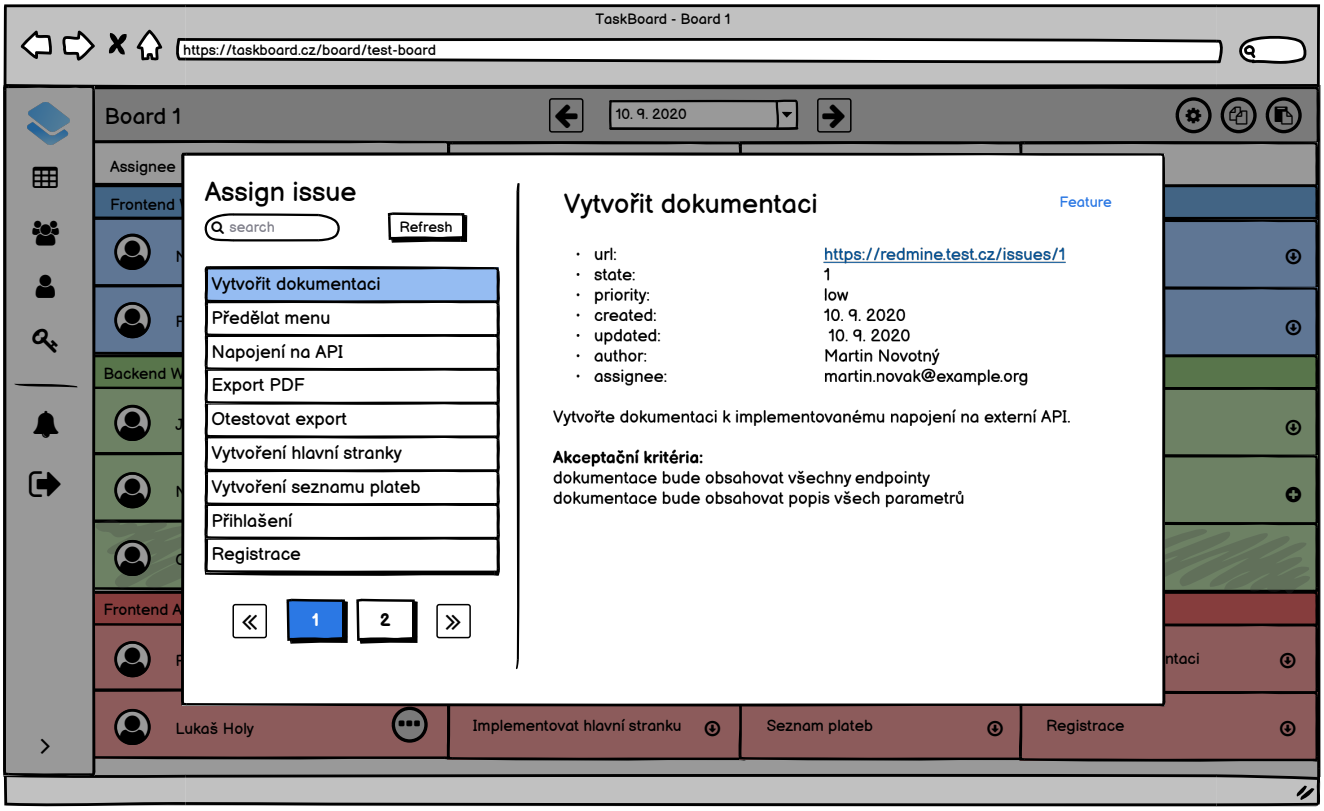

TaskBoard - Credentials
https://taskboard.cz/credentials
Credentials
search
Type
Refresh
Redmine 1
https://test.redmine.cz
Redmine 2
https://test.redmine2.cz

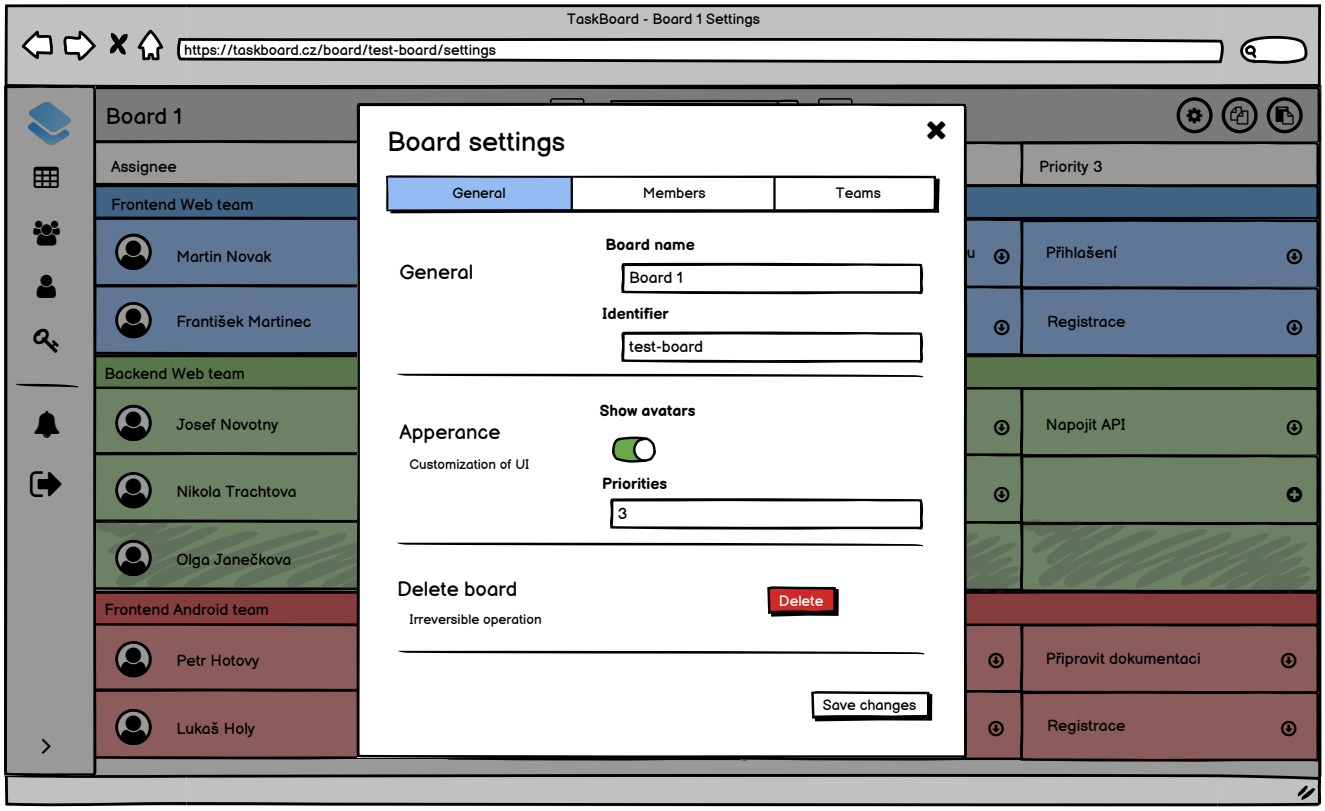

TaskBoard - Teams
https://taskboard.cz/teams
Teams
Name
Identifier
Color
Actions
Web Backend team
backend-web
Web Frontend team
fontend-web
Android Backend team
a-backend-web
Android Frontend team
a-fontend-web
iOS Backend team
i-backend-web
iOS Frontend team
i-fontend-web
Project managers
i-fontend-web
1
2

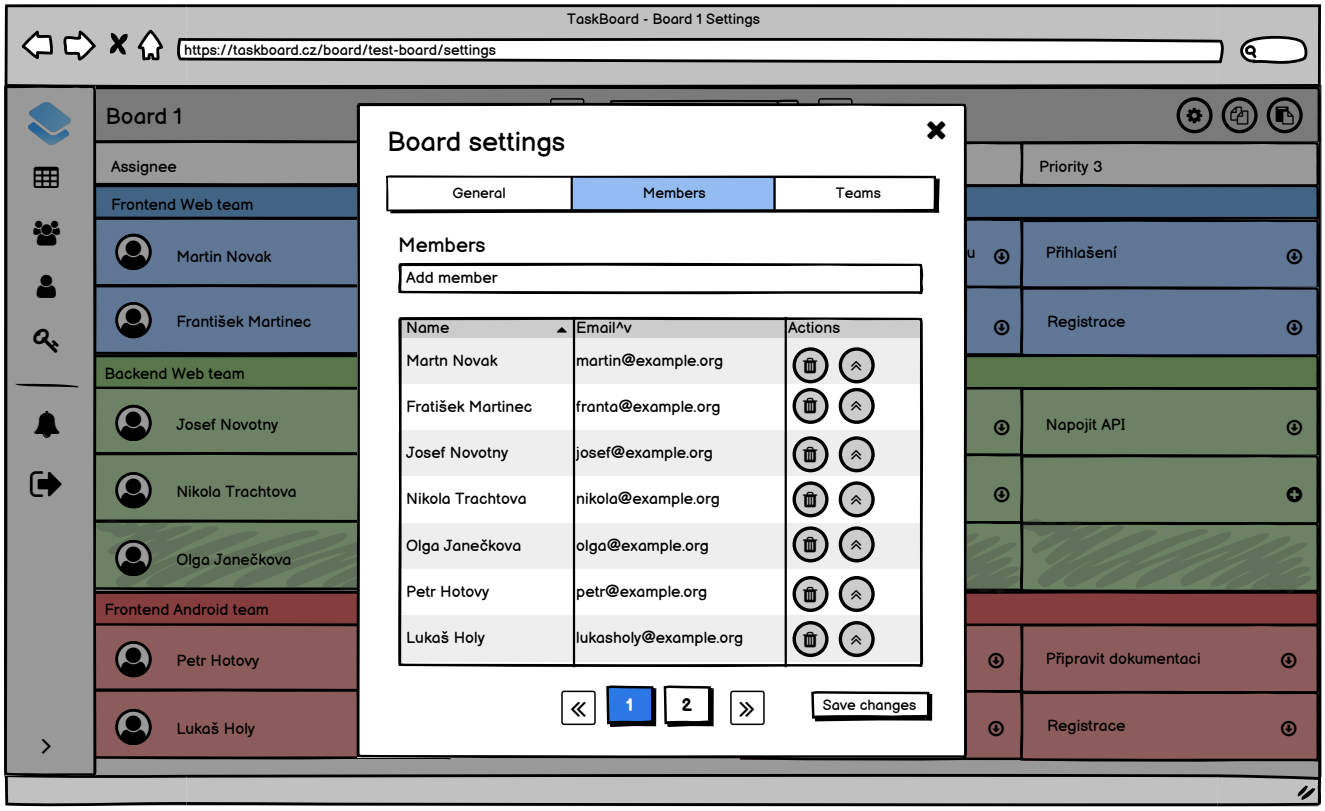

TaskBoard - Board 1
https://taskboard.cz/board/test-board
Board 1
10. 9. 2020
Assignee
Priority 1
Priority 2
Priority 3
Frontend Web team
Martin Novak
Napojení na Redmine
Implementovat hlavní stranku
Přihlašení
František Martinec
Seznam plateb
Registrace
Backend Web team
Josef Novotny
Vytvořit API
Připravit dokumentaci
Napojit API
Nikola Trachtova
Export do PDF
Testy na export
Olga Janečkova
Frontend Android team
Petr Hotovy
Export do PDF
Přihlašení
Připravit dokumentaci
Lukaš Holy
Implementovat hlavní stranku
Seznam plateb
Registrace

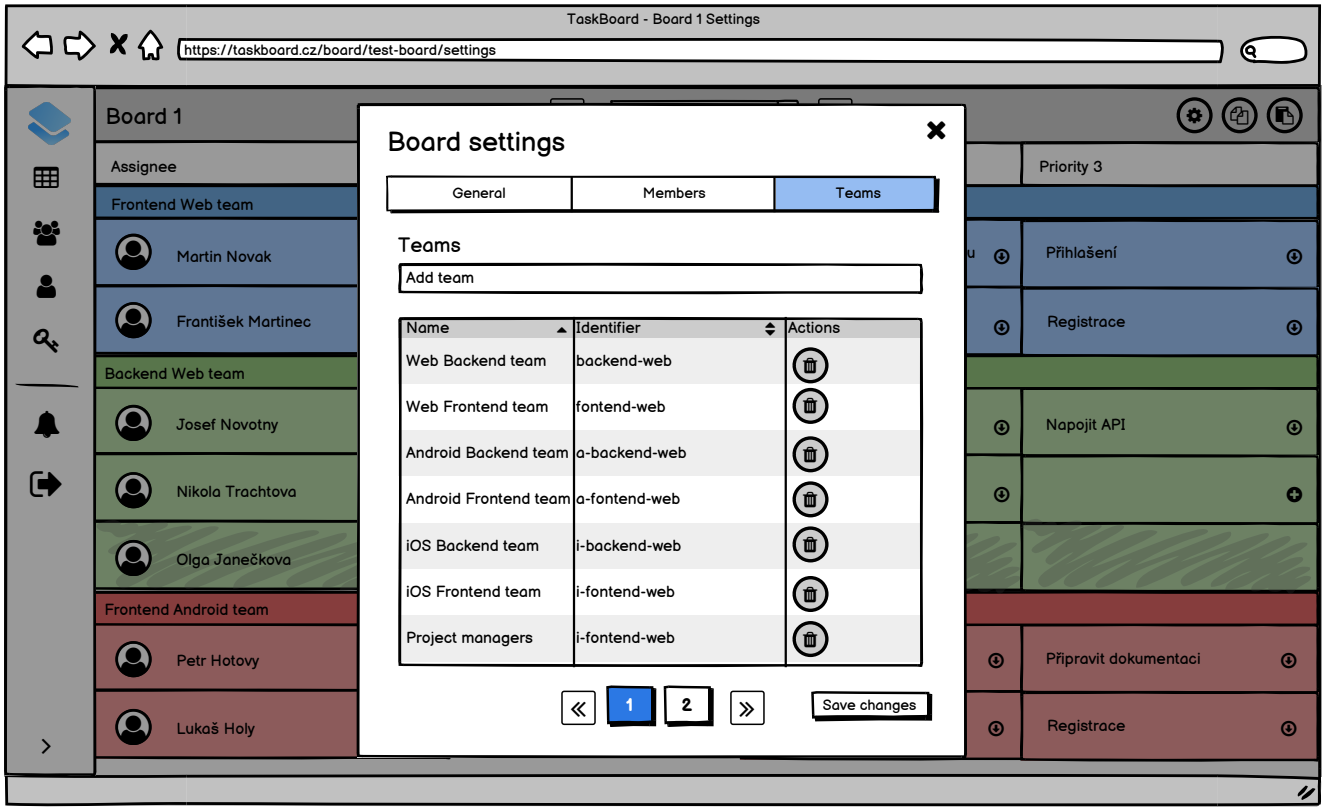

TaskBoard - Board 1
https://taskboard.cz/board/test-board
Board 1
10. 9. 2020
Assignee
Priority 1
Priority 2
Priority 3
Frontend Web team
Martin Novak
Napojení na Redmine
Implementovat hlavní stranku
Přihlašení
František Martinec
Seznam plateb
Registrace
Backend Web team
Josef Novotny
Vytvořit API
Připravit dokumentaci
Napojit API
Nikola Trachtova
Export do PDF
Testy na export
Olga Janečkova
Frontend Android team
Petr Hotovy
Export do PDF
Přihlašení
Připravit dokumentaci
Lukaš Holy
Implementovat hlavní stranku
Seznam plateb
Registrace


TaskBoard - Board 1
https://taskboard.cz/board/test-board
Board 1
10. 9. 2020
Assignee
Priority
Priority 3
Frontend Web team
Martin Novak
František Martinec
Přihlašení
Registrace
Backend Web team
Josef Novotny
Napojit API
Nikola Trachtova
Olga Janečkova
Frontend Android team
Petr Hotovy
Připravit dokumentaci
Lukaš Holy
Implementovat hlavní stranku
Seznam plateb
Registrace
2020
Fri, Dec 18
DECEMBER 2020
S M T W T F S
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
CANCEL
OK

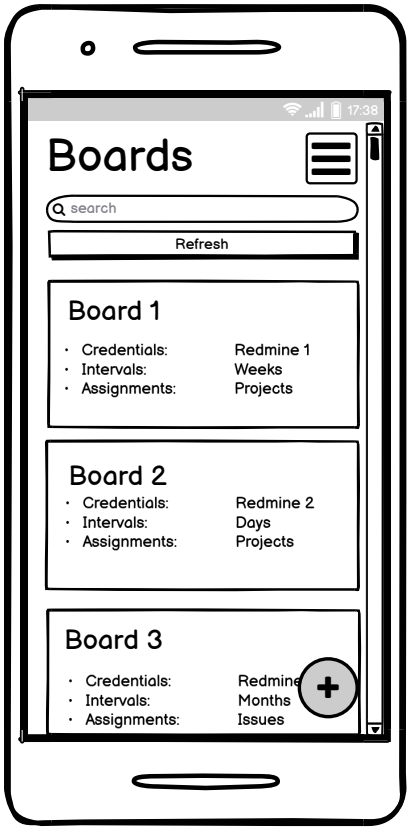

TaskBoard - Board 1
https://taskboard.cz/board/test-board
Board 1
10. 9. 2020
Assignee
Frontend
Backend
Frontend A
Assign issue
search
Refresh
Vytvořit dokumentaci
Předělat menu
Napojení na API
Export PDF
Otestovat export
Vytvoření hlavní stranky
Vytvoření seznamu plateb
Přihlašení
Registrace
1
2
Vytvořit dokumentaci
Feature
url: https://redmine.test.cz/issues/1
state: 1
priority: low
created: 10. 9. 2020
updated: 10. 9. 2020
author: Martin Novotný
assignee: martin.novak@example.org
Vytvořte dokumentaci k implementovanému napojení na externí API.
Akceptační kritéria:
dokumentace bude obsahovat všechny endpointy
dokumentace bude obsahovat popis všech parametrů
Lukaš Holy
Implementovat hlavní stranku
Seznam plateb
Registrace

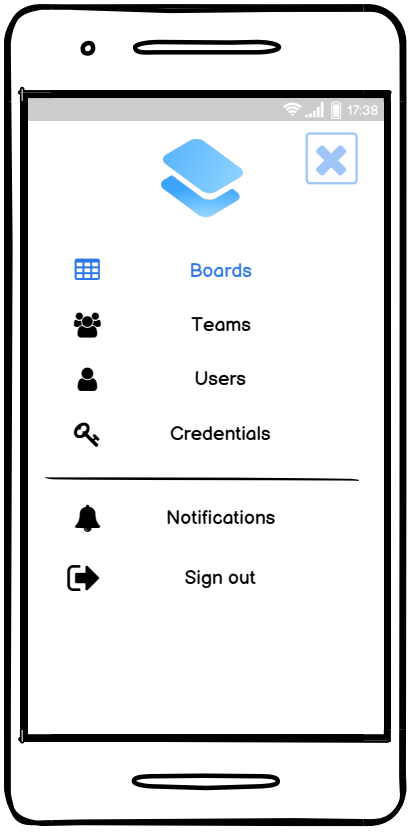

TaskBoard - Board 1 Settings
https://taskboard.cz/board/test-board/settings
Board 1
Assignee
Frontend Web team
Martin Novak
František Martinec
Backend Web team
Josef Novotny
Nikola Trachtova
Olga Janečkova
Frontend Android team
Petr Hotovy
Lukaš Holy
Priority 3
Přihlašení
Registrace
Napojit API
Připravit dokumentaci
Registrace
Board settings
General
Members
Teams
General
Board name
Board 1
Identifier
test-board
Apperance
Customization of UI
Show avatars
Priorities
3
Delete board
Irreversible operation
Delete
Save changes

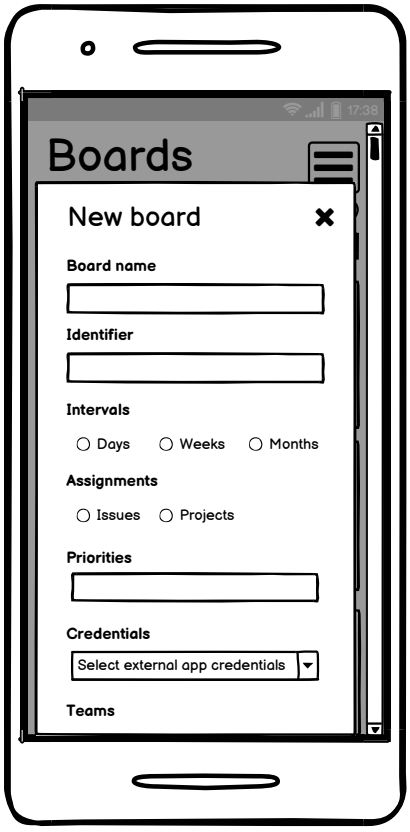

TaskBoard - Board 1 Settings
https://taskboard.cz/board/test-board/settings
Board 1
Assignee
Frontend Web team
Martin Novak
František Martinec
Backend Web team
Josef Novotny
Nikola Trachtova
Olga Janečkova
Frontend Android team
Petr Hotovy
Lukaš Holy
Priority 3
Přihlášení
Registrace
Napojit API
Připravit dokumentaci
Registrace
Board settings
General
Members
Teams
Members
Add member
Name
Email^v
Actions
Martn Novak
martin@example.org
Fratišek Martinec
franta@example.org
Josef Novotny
josef@example.org
Nikola Trachtova
nikola@example.org
Olga Janečkova
olga@example.org
Petr Hotovy
petr@example.org
Lukaš Holy
lukasholy@example.org
1
2
Save changes

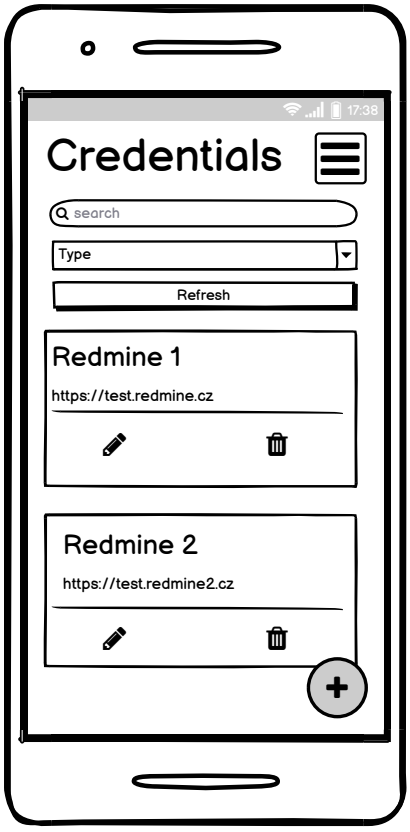

TaskBoard - Board 1 Settings
https://taskboard.cz/board/test-board/settings
Board 1
Assignee
Frontend Web team
Martin Novak
František Martinec
Backend Web team
Josef Novotny
Nikola Trachtova
Olga Janečkova
Frontend Android team
Petr Hotovy
Lukaš Holy
Priority 3
Přihlašení
Registrace
Napojit API
Připravit dokumentaci
Registrace
Board settings
General
Members
Teams
Teams
Add team
Name
Identifier
Actions
Web Backend team
backend-web
Web Frontend team
fontend-web
Android Backend team
a-backend-web
Android Frontend team
a-fontend-web
iOS Backend team
i-backend-web
iOS Frontend team
i-fontend-web
Project managers
i-fontend-web
1
2
Save changes

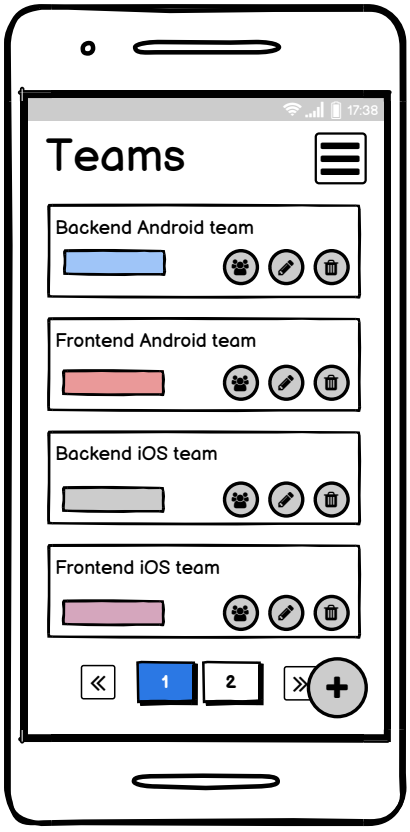

17:40
TaskBoard
Log In With Google

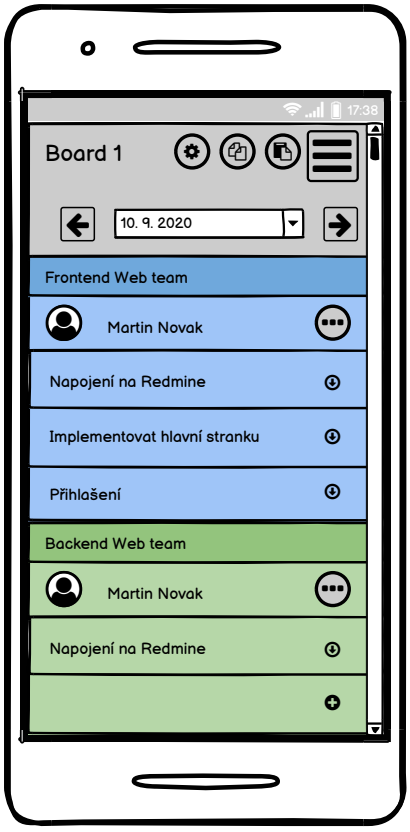

17:38
Boards
search
Refresh
Board 1
Credentials: Redmine 1
Intervals: Weeks
Assignments: Projects
Board 2
Credentials: Redmine 2
Intervals: Days
Assignments: Projects
Board 3
Credentials: Redmine
Intervals: Months
Assignments: Issues


17:38
Boards
Teams
Users
Credentials
Notifications
Sign out


Boards
New board
Board name
Identifier
Intervals
Days
Weeks
Months
Assignments
Issues
Projects
Priorities
Credentials
Select external app credentials
Teams


17:38
Credentials
search
Type
Refresh
Redmine 1
https://test.redmine.cz
Redmine 2
https://test.redmine2.cz


17:38
Teams
Backend Android team
Frontend Android team
Backend iOS team
Frontend iOS team
1
2


17:38
Board 1
10. 9. 2020
Frontend Web team
Martin Novak
Napojení na Redmine
Implementovat hlavní stranku
Přihlašení
Backend Web team
Martin Novak
Napojení na Redmine