VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ

FACULTY OF BUSINESS AND MANAGEMENT

ÚSTAV INFORMATIKY

DEPARTMENT OF INSTITUTE OF INFORMATICS

TVORBA A VÝVOJ NOVÉHO PROJEKT MANAGEMENT SYSTÉMU JUSTIMIO

CREATING AND DEVELOPMENT OF NEW PROJECT MANAGEMENT SYSTEM JUSTIMIO

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — LUKÁŠ HUVAR

AUTHOR

VEDOUCÍ PRÁCE — Ing. PETR DYDOWICZ, Ph.D.

SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA PODNIKATELSKÁ

Zadání diplomové práce

Ústav:	Ústav informatiky
Student:	**Bc. Lukáš Huvar**
Studijní program:	Systémové inženýrství a informatika
Studijní obor:	Informační management
Vedoucí práce:	**Ing. Petr Dydowicz, Ph.D.**
Akademický rok:	2016/17

Ředitel ústavu Vám v souladu se zákonem č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů a se Studijním a zkušebním řádem VUT v Brně zadává diplomovou práci s názvem:

Tvorba a vývoj nového projekt management systému Justimio

Charakteristika problematiky úkolu:

Úvod
Vymezení problému a cíle práce
Teoretická východiska práce
Analýza problému a současné situace
Vlastní návrh řešení, přínos práce
Závěr
Seznam použité literatury

Cíle, kterých má být dosaženo:

Cílem práce je popsat tvorbu a monetizaci nového projekt/time management systému Justimio. Tento systém bude sloužit pro organizaci harmonogramu a řízení projektů pomocí časové osy a task managementu.

Základní literární prameny:

BASL, Josef a Roman BLAŽÍČEK. Podnikové informační systémy. Podnik v informační společnosti. 1. vyd. Praha: Grada, 2008. 283 s. ISBN 978-80-247-2279-5.

MOLNÁR, Zdeněk. Automatizované informační systémy. 1. vyd. Praha: Strojní fakulta ČVUT, 2000. 126 s. ISBN 80-01-02269-2.

MOLNÁR, Zdeněk. Efektivnost informačních systémů. 1. vyd. Praha: Grada Publishing, 2000. 142 s. ISBN 80-7169-410-X.

ŘEPA, Václav. Analýza a návrh informačních systémů. 1. vyd. Praha: Ekopress, 1999. 403 s. ISBN 80-86119-13-0.

SODOMKA, Petr a Hana KLČOVÁ. Informační systémy v podnikové praxi. 2. aktualiz. a rozš. vyd. Brno: Computer Press, 2010. 501 s. ISBN 978-80-251-2878-7.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2016/17.

V Brně, dne 28. 2. 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA PODNIKATELSKÁ
14

L. S.

doc. RNDr. Bedřich Půža, CSc.
ředitel

doc. Ing. et Ing. Stanislav Škapa, Ph.D.
děkan

Abstrakt

Tato práce se zabývá analýzou, návrhem a implementací informačního systému Justimio pro společnost Justmighty s.r.o., která se pohybuje na trhu digitálních agentur a podniká v internetovém marketingu. Diplomová práce je rozdělena na tři části. První část práce se zabývá teoretickými východisky práce. Druhá část se zabývá analýzou současného stavu, kde za použití odborných metod dojde k analýze webové aplikace v prostředí nejbližších konkurentů. Pozornost je v této kapitole věnována i společnosti Justmighty, pro kterou je aplikace vyvíjena. Za pomocí základních analytických modelů si odpovídáme, zda je společnost schopna realizovat implementaci aplikace s ohledem na její konkurenceschopnost a jak aplikace dokáže společnosti s konkurenceschopností pomoci. Poslední část je věnována návrhu a implementaci informačního systému Justimio.

Abstract

The main purpose of this report is to analyse and suggest suitable implementation of information system Justimio, which would be used by Justmighty, a company operataning on the Czech digital market, mainly oriented on internet marketing. The master thesis is divided into three parts. The first part is focused on the analyse of the theoretical knowledge and its related possibilities. Due to the usage of professional methods, the current situation can be analysed and the web applications position in the environment of the closest competitors can be recorded. In second chapter, a focus is also given to the company Justmighty, for which the application is made for. A company's ability to implement this application is checked by using a basic analytical methods, whereas also can be analysed how this application would be helpful, when comes to the competitors and current market. The last part is focused on possible solutions and implementation of information system Justimio.

Klíčová slova

informační systém, aplikace, návrh, implementace, data, databáze, React, GraphQl, SLEPT, Porter, SWOT

Keywords

information system, application, design, implementation, data, database, React, GraphQl, SLEPT, Porter, SWOT

Biblografická citace

HUVAR, L. *Tvorba a vývoj nového projekt management systému Justimio*. Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2017. 67 s. Vedoucí práce Ing. Dydowicz Petr, Ph.D..

Prohlášení

Prohlašuji, že předložená diplomová práce je původní a zpracoval jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem ve své práci neporušil autorská práva (ve smyslu Zákona č. 121/2000 Sb., o právu autorském a o právech souvisejících s právem autorským).

. . . . . .

Lukáš Huvar

24. května 2017

Poděkování

Tímto bych poděkoval Ing. Petru Dydowiczovi, Ph.D. za odobrnné vedení a cenné rady, které mi pomohly k vytvoření této práce. Dále bych chtěl poděkovat společnost Justmighty s.r.o a všem jejím zaměstnancům za poskytnuté informace a pomoc při tvorbě této práce. Nakonec bych rád poděkoval všem kamarádům a rodině, kteří mi jakýmkoliv způsobem pomohli a podíleli se na vypracování této práce.

Obsah

Úvod

V dnešní hektické a uspěchané době je organizace času důležitou součástí běžného života jedince i základem fungování úspěšné firmy. Každodenně se setkáváme s problémem dodělávání zakázek na poslední chvíli, z důvodu nastavení nedodržitelných termínů. Bohužel v této době neexistuje pro menší a střední firmy aplikace, která by pomáhala řídit, hlídat a organizovat projekty s důrazem na časové hledisko. Mohlo by se zdát, že vytvoření takovéto aplikace bude pro organizaci spíše plýtvání časem. Praxe ale ukazuje, že s dodržováním termínů mají podniky opakované a časté problémy a aplikace by tedy mohla mít v dlouhodobém horizontu velký užitek a přínos. Pro klienta by hlavní benefit plynul ve splnění zakázky ve stanoveném termínu. Firmy by potom profitovaly zejména z vyšší efektivity jednotlivých procesů. Díky včasnému splnění projektů by zároveň došlo i k finanční úspoře a stejně tak i ke snížení stresové zátěže zaměstnanců.

Práce je členěná na dvě části – teoretickou a praktickou. V teoretické části nejdříve vysvětlíme technické pojmy, spojené s tvorbou informačního systému. Zároveň se zaměříme na definování potřeb společnosti Justmighty, pro kterou webová aplikace vznikne. Dále budou popsány metody pro analýzu vnitřního a vnějšího prostředí společnosti, včetně SWOT analýzy konkurenčních aplikací, aktuálně dostupných na trhu.

V praktické části budou následně aplikovány metody definované v teoretické části. Stěžejním výstupem a zároveň i hlavním cílem této práce je potom návrh a popis implementace samotného informačního systému Justimio. Očekávaný přínos plyne zejména pro vedení a zaměstnance společnosti Justmighty.

1. Vymezení problému a cíle práce

Cílem diplomové práce bylo navrhnout a implementovat informační systém Justimio pro společnost Justmighty s.r.o.. Tento informační systém bude implementován jako webová aplikace, která umožní uživatelům přehledně pracovat s daty. Aplikace pomůže společnosti při plánování harmonogramu jednotlivých úkolů pro klientskou i interní práci. Požadavek na tento informační systém vznikl z důsledku nedodržování termínů zakázek a obecně špatné organizace práce ve společnosti Justmighty.

Veškeré aplikace na trhu nesplňovaly požadavky společnosti Justmighty na funkčnost systému a uživatelskou přívětivost, proto jsme se rozhodli vytvořit vlastní řešení. Webová aplikace bude využívat nejmodernější technologie, aby nedošlo v blízké době k zestárnutí systému a problémům s udržováním informačního systému. Tyto moderní technologie také umožní rychlou implementaci jako nativní aplikaci pro mobilní zařízení a stolní počítače. Celá aplikace bude zároveň navržena s ohledem na možný budoucí růst a škálování systému.

2. Teoretická východiska práce

V následující kapitole probereme základní teoretické pojmy nutné ke správné tvorbě webové aplikace. Výsledkem práce má být webová aplikace, proto si nejdříve teoreticky zakotvíme technické pojmy týkající se informačního systému. Následně popíšeme všechny technologie, které budou použity k implementaci tohoto informačního systému.

2.1 Data

V kontextu informačních technologií se data používají pro označení čísla, textu zvuku, obrazu nebo jiných vjemů ve vhodné podobě pro zpracování počítačem. Při práci s daty je můžeme rozdělovat na:

- **Strukturovaná data** - zachycují fakta, atributy, objekty atd. Typickým příkladem jsou uložená data v databázových systémech, kdy víme, která data co znamenají a k čemu je můžeme použít.
- **Nestrukturovaná data** - jsou vyjádřena jako posloupnost bytů bez dalšího rozlišení. Jedná se především o videozáznamy, obrázky a textové dokumenty.

Data slouží pro reprezentaci faktů, atributů, obrazu dějů a věcí. Například **60210** může reprezentovat něco v realném světě, ale bez kontextu data nedávají smysl. Je to pouze posloupnost náhodných čísel.[1]

2.2 Informace

Informace jsou data v kontextu, které jsou použitelné a srozumitelné. Pokud vezmeme minulý příklad s číslem **60210** a uvedeme, že se jedná o číslo popisné. Můžeme konstatovat, že se jedná o číslo popisné městské části Brno-Střed. V tomto případě se jedná o informaci. Data, která popisují data, se nazývají metada. Pokud tedy data popíšeme pomocí metadat, stávají se z nich informace, s kterými nadále můžeme pracovat.

Informace jsou v dnešním světe jednou z nejdůležitějších výrobních komodit a také cenným artiklem, se kterým můžeme obchodovat. Proto je vždy velmi důležité, aby se s nimi odpovědně nakládalo, řídilo jejich zpracování a musí se klást důraz, jak se s kterými informacemi zachází. Pěkným příkladem toho, jak jsou informace důležité je např.

informace o stavu měnového kurzu. V některých momentech může mít taková informace nesrovnatelně větší hodnotu než např. cena luxusního automobilu.[1]

2.3 Systém

Systém je množina prvků, které mají mezi sebou vazby. Tyto vazby určují vlastnosti tohoto systému. Kdykoliv dojde ke změně jednoho prvku systému, má to vliv na všechny ostatní prvky systému. Většina systémů má vstupní a výstupní vazby, které určují jejich chování k okolí.[2]

2.4 Informační systém

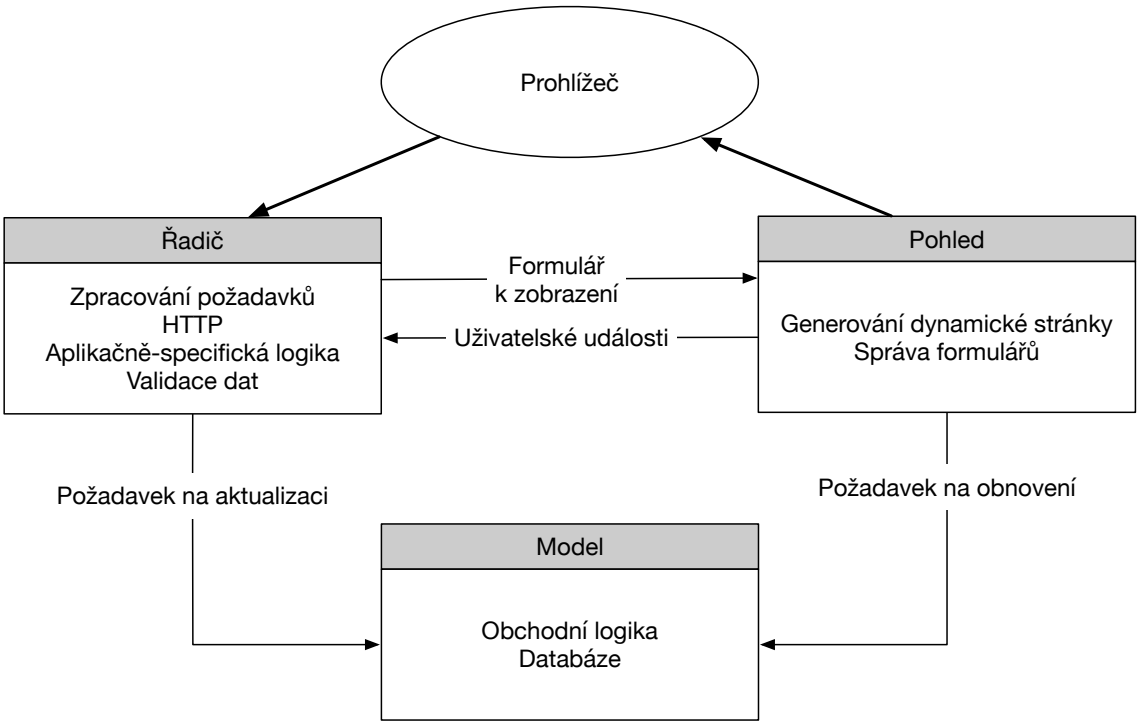
Pomocí výše popsaných informací a systému můžeme definovat informační systém. Je to soubor lidí, technických prostředků a metod, který zabezpečuje sběr, přenos, uschování a zpracování dat za účelem tvorby a prezentace informací pro potřeby uživatelů v systému. Informační systémy vždy pracují se sdílenou databází a umožňují kontrolovaný přístup k velké bázi informací. V současné době se většina informačních systémů přesouvá na web, kde jsou přístupné pomocí webového prohlížeče.[1]

2.5 Architektonické vzory

Architektonickými vzory myslíme stylizované abstraktní popisy optimálního postupu, který byl vyzkoušen a otestován v různých systémech a prostředích. Architektonický vzor popisuje organizaci systému, který byl v minulosti úspěšný. Typický architektonický vzor by měl jasně definovat, kdy je vhodné daný vzor použít a kdy se mu naopak vyhnout. Nedílnou součásti popisu jsou i jeho silné a slabé stránky.[2]

2.5.1 Model-view-controller

Model-view-controller je návrhový vzor, který odděluje prezentaci a interakci od systémových dat. Systém je strukturalizován do tří oddělených komponent, které mezi sebou reagují. Komponenta **Model** se stará o správu dat a operace s nimi. Komponenta **View** (Podhled) definuje, jak se budou data zobrazovat uživateli. **Controller** (Řadič) se stará o uživatelskou interakci (např. stisknutí kláves, pohyb myší) a předává tyto informace ostatním komponentám. Používá se v případech, kdy existuje více způsobu zobrazení dat a interakcí s nimi. Například architektura webového aplikačního systému:

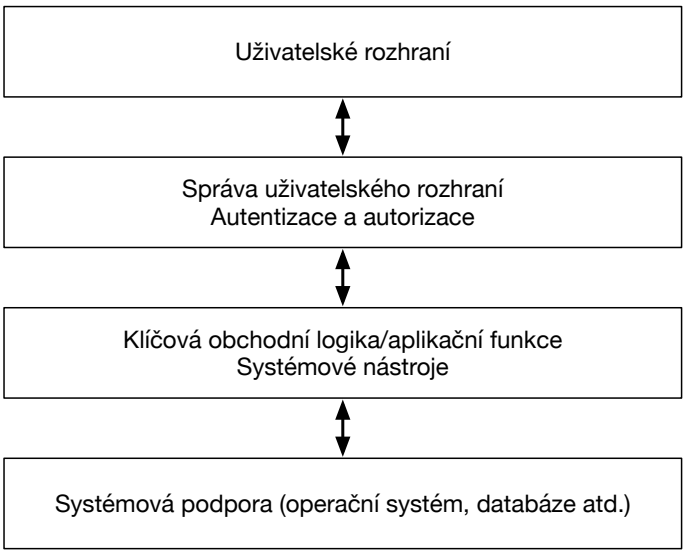


Obrázek 2.1: Architektura webové aplikace využívající MVC (Zdroj: vlastní)

Mezi jeho hlavní výhody patří změna dat nezávisle na jejich prezentaci. Umožňuje zobrazit stejná data různými způsoby a každá změna se promítne ve všech prezentacích. Nevýhoda tohoto vzoru je přílišná složitost v případech jednoduchého datového modelu a interakcí.[2]

2.5.2 Vrstevnatá architektura

Vrstevnatá architektura organizuje systém na vrstvy, které mají souvislé funkce. Jedná se o systém tzv. vrstev, kde jedna vrstva poskytuje služby vrstvě nad ní. Z toho vyplývá, že nejnižší vrstvy patří mezi klíčové služby, které jsou používány v celém systému. Vrstevnatá architektura se často používá při budování nových funkcí už nad existujícími systémy. Jiným příkladem využití vrstevnaté architektury je např. systém na sdílení dokumentů s autorskými právy v knihovnách, kde tato architektura umožňuje práci rozdělit na více týmů:

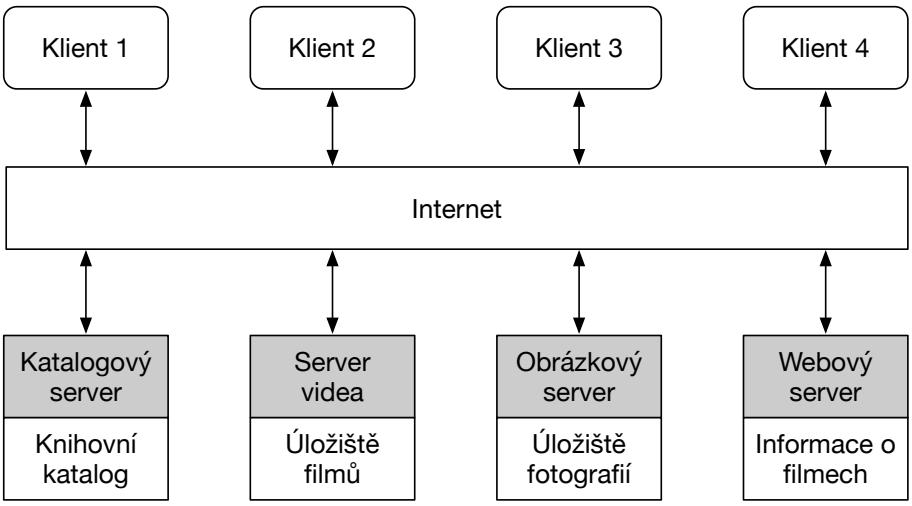


Obrázek 2.2: Generická vrstevnatá architektura (Zdroj: vlastní)

Tento vzor umožňuje nahrazovat celé vrstvy, pokud zachováme stejné rozhraní. Na každé vrstvě je možné poskytnout redundantní funkce, což zvyšuje spolehlivost systému. Mezi nevýhody patří oddělení vrstev, aby například funkcionality z nejvyšší vrstvy nezáležely na nejnižší vrstvě. Problémem může být například požadavek na službu, která je zpracovávaná více vrstvami, protože se tím snižuje její výkon.[2]

2.5.3 Architektura klient-server

Architektura klient-server rozděluje funkčnost do služeb, kdy každou službu poskytuje samostatný server. Klienti jsou uživatelé těchto služeb a přistupují k serverům, aby mohli tyto služby využívat. Využívá se při potřebě přistupovat ke sdílené databázi z různých umístění. Tato architektura může být například použita na knihovnu filmů a videodisků DVD:



Obrázek 2.3: Architektura klient/server pro filmovou knihovnu (Zdroj: vlastní)

Největší výhodou tohoto modelu je možnost distribuce serverů v síti. Nevýhodou pak spatřujeme především v tom, že každá služba představuje kriticky zranitelné místo. Model je často vystaven rizikům a to ve formě útoků nebo selhání serverů.[2]

2.6 Webové systémy

Webové systémy sloužily jako univerzální dostupné úložiště informací. Následně se webové systémy začaly vyvíjet novým směrem a prohlížeče dostávaly stále více nových funkcí. To umožnilo vyvinout složitější systémy, které byly dostupné pomocí webového prohlížeče. Tyto systémy dokázaly přes web poskytnout inovativní služby, a tím přilákat čím dál větší množství zákazníků.

Díky webovým prohlížečům bylo mnohem levnější měnit a aktualizovat systémy, protože nebylo potřeba aktualizovat systémy v každém počítači. Uvedený přístup snížil náklady na vývoj a výzkum.

Další fází webových systémů bylo rozdělení na služby. Toto rozdělení nám umožnilo přeměnit velké monolitické podnikové systémy na malé služby. Každá část systému má rozdílnou funkčnost, a proto je potřeba definovat rozhraní této služby. Rozhraní webových systémů slouží ke komunikaci s okolním světem. Rozhraní webových systémů můžeme dělit na uživatelské a aplikační.[2]

2.6.1 Uživatelské rozhraní

Uživatelské rozhraní, je koncová část aplikace, kde jsou uživateli zobrazovány data a může s nimi pracovat. Jedná se o klíčovou složku uživatelského požitku (user experi-

ence) z informačního systému. Tedy snahu o maximalizaci užitku z používání informačního systému s co nejjednodušší a nejpřívětivější ovládání. Dnešním trendem je minimalizace všech rušivých prvků jako jsou stíny, vzory, barevné přechody. Tento přístup se nazývá **Flat design**, jedná se o minimalistický designový styl. Využívá se zejména v oblasti designu uživatelských rozhraní, a nenapoduje 3D prvky designu, jak tomu bylo v minulosti.[3]

2.6.2 Aplikační rozhraní

Aplikační rozhraní (API - Application Programing Interface) slouží jako komunikační bod mezi různými aplikacemi. Tvoří ho soubor procedur, funkcí, tříd nějaké knihovny, které se mohou využívat. Toto rozhraní může sloužit ke komunikaci s aplikacemi třetích stran. Využívá se ke komunikaci mezi webovými systémy. Může se jednat například o **REST** api, které je založeno na přenosu prostředků ze serveru ke klientovi. Zdroje jsou popsány adresami **URL** (Uniform Resource Locator), které nám určují, o který zdroj se jedná.[4]

2.7 Cloud computing

Je model umožňující nabízet uživatelům sdílené a konfigurovatelné výpočetní prostředky (síť, výkon počítače, úložiště dat, aplikace atd.) Aby byl zajištěn všudypřítomný, rychlý a uživatelsky pohodlný přístup spojený s minimální komunikací s poskytovatelem prostředků. Cloud computing si zakládá na monitorování a měření využívání služeb. Uživatelé platí pouze za služby, které využili. Tento systém umožňuje elasticky škálovat a přizpůsobit výkon dle aktuální zátěže.[5] Cloud computing je nejčastěji dělen na tři modely služeb:

- **Poskytnutí software jako služby (SaaS)** - V tomto modelu jsou poskytovány zákazníkovi aplikace, které jsou provozované a poskytované na jeho pronajaté platformě či infrastruktuře. Zákazník u této kategorie služeb nenastavuje ani nekontroluje platformu, na které to funguje, ale může si upravovat samotnou aplikaci. Mezi nejznámější patří: **GoodData BI**, **NetSuite ERP**, **Google Apps**, **Facebook**, **Youtube**.

- **Poskytnutí výpočetní platformy jako služby (Paas)** - Tento model umožňuje zákazníkům vyvinout aplikaci a zároveň ji provozovat, a nebo může nasadit apli-

kace třetích stran. Mezi nejznámější patří: **Microsoft Azure**, **Google AppEngine**, **Oracle PasS**.

- **Poskytnutí infrastruktury jako služby (IaaS)** - Zákazníkovi jsou poskytnuty přímo výpočetní zdroje, na které si může nasadit libovolný software, včetně operačního systému. Mezi nejznámější patří: **Amazon Elastic Compute Cloud**, **IBM Smart Cloud**, **Oracle IaaS**.

2.8 Webové technologie

Webové technologie nám umožňují vytvářet uživatelské rozhraní ve webových prohlížečích. Často se tento obor nazývá **"frontend"**, a zjednodušeně můžeme říci, že je to "to co vidíme". Využití webových technologií s sebou nese mnoho výhod. Největšími výhodami je přenositelnost nebo dostupnost z jak různých zařízení, tak i míst přístupu. Toto odvětví se rychle vyvíjí a umožňuje nám vytvářet různé kombinace aplikací a služeb.[4]

2.8.1 HTML

Nebo také Hyper Text Markup Language je značkovací jazyk, který vznikl v 90. letech minulého století a slouží pro vytváření dokumentů, které obsahují hypertextové odkazy. O standardizaci HTML se stará organizace W3C, která je vedená Timem Berners-Leem, který je vynálezcem jazyku HTML. Aktuální verze jazyka HTML je 5.1, kterou se řídí všechny webové prohlížeče.

Pro definici dokumentu HTML využívá značky. Tyto značky můžou být párové - například `<p>Odstavec</p>` pro odstavec - a nebo `<hr>`, která je nepárová značka a používá se k vykreslení vodorovné čáry. Pomocí těchto značek se vytvoří základní struktura dokumentu, která poté slouží k zobrazovaní pro uživatele.

Nejčastěji se pro zobrazování dokumentů používají webové prohlížeče, které tyto dokumenty transformují do Document Object Model, zkráceně DOM. K dosažení stejného vzhledu na všech prohlížečích je potřeba mít validní HTML. Validní HTML zaručuje rychlejší zobrazování, jelikož prohlížeče nemusí upravovat nevalidní kód a rovnou ho začít vykreslovat. Validní kód nám také zaručí fungování v budoucnosti bez nutných změn a tím i zpětnou kompatibilitu.[4]

2.8.2 Css

První verze jazyka Css se objevila v roce 1996 a stála za ním organizace W3C. Css nebo Cascading Style Sheets (v češtině kaskádové styly) slouží ke stylování dokument pomocí pravidel, které se aplikují na HTML elementy.

Stylování pomocí Css nám šetří čas, protože nemusíme stylovat všechny elementy. Stačí styl definovat jen jednou, pojmenovat ho a následně ke každému elementu přidat jméno našeho stylu.

S rychlým vývojem webových technologií se rozšiřují vlastnosti, které můžeme stylovat a modifikovat. S tímto souvisí problém, že všechny webové prohlížeče neznají všechny vlastnosti, proto je dnes nutné jim přidávat tzv. prefixy. Tyto prefixy umožňují využití nových funkcionalit Css.

Pro další zjednodušení práce s Css vznikly nadstavby nad Css (sass, less, stylus atd.), které nám umožňují využívat proměnné, funkce a jednoduché matematické operace. Následně je potřeba přeložit tyto nadstavby zpátky do Css ,abychom jsme zaručili fungování ve webových prohlížečích.[4]

2.8.3 Javascript

Javascript je multiplatformní programovací jazyk, který byl napsán Brendnem Eichem v roce 1995. Javascript je prototypovací jazyk s podporou objektovo-orientovaného, imperativního a funkcionálního programovacího stylu. Sloužil převážně pro manipulaci se strukturou DOM a úpravě vlastností HTML tagů. Umožňoval jednoduché úpravy webových dokumentů ve webových prohlížečích a tím pomáhal rozpohybovat statické webové stránky.

S rychlým vývojem webových technologií se funkčnost javascriptu rozšiřovala a vznikly různé knihovny, které umožňovaly jednodušší práci s DOM strukturou, vykreslováním grafů atd. Následně vznikl Node.js, který umožňoval fungování javascriptu mimo webový prohlížeč a umožňuje vytvořit například webový server. Z tohoto důvodu můžeme psát jediný kód, který funguje jak na serveru, tak ve webových prohlížečích. To umožnilo javascriptu se rozšířit na ostatní zařízení jako jsou mobilní telefony nebo tablety. Proto se začal používat výraz **Universal javascript**[6], který se používá pro aplikace se sdíleným kódem pro rozdílné architektury. Tento přístup umožnil snížení nákladů na vývoj a javascript se díky tomu stal velice populární.

2.9 GraphQL

GraphQL je dotazovací jazyk na API, který je zaměřen na klienta. Není omezen na žádný programovací jazyk a proto může fungovat na všech platformách. Je založen na schématech a typech, které přesně definují strukturu tohoto API. GraphQL je považován za následovníka úspěšného RESTu. GraphQl poskytuje výkonné vývojářské nástroje, které nám usnadňují vývoj API v průběhu času.[7] Mezi hlavní výhody GraphQl patří:

- **Pouze potřebná data** - Při posílání dotazů získáme pouze data, o které jsme žádali. Klient předem definuje, které data potřebuje a server nemusí posílat nechtěná data. Omezuje se tím velikost přenesených dat a také rychlost přenosu s tím spojena.
- **Omezení počtu dotazů** - GraphQl nám umožňuje spojovat zdroje do jednoho dotazu. Zatímco typické API rozhraní REST vyžaduje načítání z více adres URL (více dotazů), GraphQL získá všechny údaje, které aplikace potřebuje v jediném dotazu. Aplikace pomocí GraphQL mohou být rychlé i na pomalých mobilních sítích.
- **Verzování API** - Pokud je potřeba rozšířit stávající API, nijak to neovlivní už stávající funkčnost. Toto nám umožní udržovat přehlednější a čistší zdrojový kód. Jelikož nás GraphQl neovlivňuje, můžeme inkrementálně rozšiřovat funkčnost aplikací.

2.9.1 Schéma a typy

Schéma slouží k popisu struktury dat, které můžeme získat od GraphQl API. Data jsou popsána jako objekty, které mají určité pole. Schéma slouží k dotazům a jejich mutacím. Slouží jako kontrola, jestli dotaz je validní a neobsahuje chyby.

Typy slouží k popisu polí. Typy mají více druhů od klasických řetězců, čísel a výčtů. Mohou obsahovat vztahy mezi ostatními typy nebo polemi typů. Můžeme určit, zda jsou to povinné nebo volitelné pole dat. Všechny tyto vlastnosti nám slouží k popisu celé struktury našeho GraphQl API. Takto vytvořené schéma nám tvoří strukturu podobnou grafům, proto máme v názvu Graph.

2.9.2 Dotazy a mutace

Každý GraphQl servis má dotazy a mutace. Dotaz slouží pro získávání dat ze serveru. Dotazy jsou definovány pomocí polí objektu, které jsou definovány pomocí schémat. Dotazy mohou obsahovat argumenty, které nám umožňují získat pouze data, které chceme (např. všechny červené auta). Pomocí argumentů je data možné řadit a agregovat.

Mutace slouží k úpravě dat. Při každém dotazu můžeme data upravovat a následná odpověď nám už vrátí data upravená. Mutace proto slouží k přidávání nových dat a úpravě stávajících.

2.10 React

Je javascriptová knihovna pro tvorbu uživatelského rozhraní. Tato knihovna patří mezi open source knihovny a stojí za ní především vývojáři z Facebooku a open source komunita.[8] React je založen na třech základních pilířích a jmenovitě:

- **Deklarativní** - každý stav aplikace má vždy definovaný výsledný vzhled. React se stará, aby se vždy jen potřebné komponenty překreslily.
- **Komponentově založena** - každá komponenta je zapouzdřena sama o sobě, a tím pádem dovedeme jednoduše skládat celé uživatelské rozhraní. Toto nám umožní komponenty používat naskrz celou aplikací s různou funkčností.
- **"Naučte se jen jednou"** - dá se používat na všech platformách. React se může používat na mobilních zařízení jako React Native, nebo vykreslovat přímo na serveru.

2.10.1 Princip fungování

Pro vykreslování komponent React využívá virtuálního DOMu, který porovnává s aktuální DOM strukturou stránky. Následně se vyhodnotí nejmenší počet potřebných změn, které musí React vykonat, abychom z našeho stavu dosáhli výsledného vzhledu. Existuje mnoho generických řešení tohoto problému, například **State of the art algorithms**. Má složitost $O(n)^3$, kde n je počet elementů ve stromu.[9] Pokud bychom měli 1000 elementů, museli by jsme udělat miliardu porovnání. Proto React využívá heuristický algoritmus o složitosti $O(n)$, který je založen na dvou předpokladech:

- Dva elementy rozdílného typu vytvoří dva různé stromy.
- Vývojář může naznačit, který podřízený element může být stabilní pomocí klíče.

Tento systém řeší výkonnostně náročné překreslování webových stránek, a tím šetří cenné zdroje koncovým uživatelům.

2.10.2 Komponenty

V reactu se komponenty dělí na komponenty "se stavem"a komponenty "bez stavu". Komponenty "bez stavu"jsou čisté funkce a slouží pouze k zobrazování uživatelského rozhraní. Stav v komponentách uchovává data, pokud dojde k jeho změně. React porovná nový virtuální DOM s aktuální DOM strukturou a vykoná potřebné změny.

Jednotlivé komponenty jsou skládány do složitějších komponent. Tato vývojářská metodika se nazývá **Component-Driven Development**[10]. Je to proces tvorby uživatelského rozhraní "od zdola nahoru". Vytvořené komponenty se skládají až do výsledných stránek aplikace. Tento styl vývoje má několik výhod:

- **Vytvoření knihovny komponent** - která se může používat napříč projekty a ulehčí vývoj.
- **Vizuální testování** - jednoduché testování vizuální stránky komponent a jejich stavů.
- **Zaměřený vývoj** - pouze na jednu komponentu a její chování. Není potřeba měnit stav celé aplikace a pracně s ní manipulovat.
- **Paralérní spolupráce** - možnost práce na stejné obrazovce. Nemusíme se bát, že změnami ohrožujeme práci druhých.

2.11 Zabezpečení informačních systémů

Bezpečnost je jeden ze základních stavebních kamenů informačních systémů. Informační systémy uchovávají velké množství informací, které slouží jako obchodní artikl a cenné know-how společnosti. Některá data podléhají zákonu na ochranu osobních údajů, proto je data nutné dobře zabezpečit, aby nedošlo k úniku informací. Bezpečný informační systém je takový, který je zajištěn po fyzické, administrativní, technické a logické stránce.[11]

- **Hardwarové zabezpečení** - je založeno na fyzickém přístupu k zařízení. Datové servery by měly být na bezpečném místě a měly by být přístupné pouze oprávněným osobám.
- **Softwarové zabezpečení** - jedná se o celkové zabezpečení aplikace a operačního systému. Měl by být povolen pouze autorizovaný přístup a systém by měl být chráněn před škodlivým kódem a hackerskými útoky.

- **Datové zabezpečení** - si zakládá na bezpečném uchování dat, řádném zálohováním a omezeném přístupu. Data by měla být šifrována, aby nedocházelo k jejich zneužití.

2.11.1 Možné hrozby informačních systémů

V současné době existuje velké množství typů útoků na informační systémy. Útoky se liší ve velikosti dopadu a v možnostech jak se jim bránit.[5] Pár příkladů typu útoku:

- **Neoprávněný přístup** - je založen na neautorizovaném přístupu ke zdrojům. Útočník získá přístup k nezabezpečeným zdrojům, které pak následně může využít ke svému prospěchu.

- **Odposlech dat** - je útok směřovaný na zcizení informací, kterou můžou být přihlašovací údaje, číslo kreditní karty, platební příkazy atd. Tyto informace se následně používají k vlastnímu prospěchu. Při odposlechu dochází k narušení soukromí a je narušena důvěryhodnost celého systému. Tento útok se používá při přenosu dat mezi koncovým uživatelem a serverem.

- **Modifikace dat** - jedná se o typ útoku, při kterém se modifikují přenesená data. Může se jednat například o změnu platebního příkazu nebo platebního šeku. Tento útok se nazývá **Mann in the middle**. Útočník je prostředníkem mezi komunikací serveru a koncového uživatele.

2.11.2 Zabezpečená HTTPS komunikace

Nedílnou součástí zabezpečení webových aplikací je šifrovaný protokol HTTPS. Tento protokol využívá protokol HTTP, který je šifrován pomocí protokolu SSL nebo TSL. Před zahájením komunikace je potřeba navzájem prokázat totožnost. Server svojí totožnost prokáže digitálním certifikátem, koncový uživatel se rozhodne jestli bude důvěřovat tomuto serveru. Server s tímto certifikátem posílá veřejný klíč, který slouží k šifrování informací. Následně klient pošle svůj zašifrovaný certifikát pomocí veřejného klíče. Server tímto způsobem získá bezpečně certifikát klienta. Tento certifikát obsahuje veřejný klíč pro šifrování dat pro klienta. Pomocí tohoto klíče server zašifruje posílaná data. Tímto způsobem vznikne šifrovaná komunikace mezi serverem a klientem. Pokud jedna ze stran ukončí přenos informací, celá šifrovaná komunikace se ukončí.

Aby nedocházelo k falšování certifikátů jsou vydávány pouze certifikačními autoritami. Tímto je zamezeno, aby certifikáty nebyly padělány a nebyla prolomena šifrovaná

komunikace. Zabezpečená HTTPS komunikace slouží k ochraně před odposlechem dat a ochraně před modifikací dat.[5]

2.11.3 Autentizace

Jedná se o proces ověření pravosti daného subjektu. Využívá se k zabezpečení a slouží k ochraně před falšováním identity[5]. Pro ověření identity se využívají faktory autentizace mezi které patří:

- **Znalostní** - Jedná se o faktor, který uživatel zná. Může se tedy jednat o heslo, PIN, bezpečnostní otázku, nebo také jejich sérii.
- **Vlastnická** - Tento faktor vyžaduje vlastnictví nějaké identifikace. Může se jednat o náramek, průkaz totožnosti, bezpečnostní klíč, telefon s hardwarovým nebo softwarovým klíčem.
- **Biometrická** - Mezi tyto faktory patří, co uživatel má nebo dělá. Jedná se o otisk prstu, DNA, podpis, hlas nebo biometrické znaky.

Podle typu systému, který je potřeba zabezpečit se vybírá kolik faktorů má tato autentizace mít. Nejbezpečnější třídou autentizace je biometrická, která vyžaduje přítomnost dané osoby. Typy autentizace jsou rozděleny do tří skupin a to:

- **Jednofaktorová autentizace** - Jedná se o nejslabší úroveň autentizace, využívá pouze jednu třídu autentizace. Tento typ autentizace se nedoporučuje pro finanční nebo uživatelsky důležité transakce, které vyžadují vysokou úroveň zabezpečení.
- **Dvoufaktorová autentizace** - Využívá alespoň dvě třídy autentizace k zajištění identity. Může se například jednat o přístup k účtu. Vlastnický faktor je bankovní karta a znalostní faktor je PIN k účtu.
- **Vícefaktorová autentizace** - Tato metoda využívá všechny třídy autentizace a jejich kombinace.

2.11.4 Autorizace

Jedná se o proces získávání souhlasu s provedením operací a přístupům ke zdrojům. Systém autorizace má poskytnout odpověď na otázky:

- Je uživatel X autorizován k přístupu ke zdroji Z?

- Je uživatel X autorizován k provedení operace O?
- Je uživatel X autorizován k provedení operace O na zdroji Z?

K zajištění autorizace potřebujeme zjistit, který subjekt požaduje přístup. Nejdřív potřebujeme provést autentizaci a uživatele označit, abychom jsme věděli o koho se jedná. Pro označení uživatelů v informačních systémech se používají nejčastěji cookies nebo přístupový klíč. Obě tyto metody obsahují informace o uživateli. Podle těchto údajů se systém rozhoduje, jestli uživatel má dostatečné práva k provedení operace. Autorizace a autentizace slouží k ochraně před neoprávněným přístupem k datům a jejich možnému zneužití.

2.12 Open source

"Open source"nebo také "otevřený software"označuje přístup k vývoji software. Kód je volně přístupný a jsou k němu přizváni dobrovolníci. Tento přístup přinesla Free Software Foundation, která doporučuje, aby zdrojový kód byl otevřený a přístupný uživatelům, kteří jej mohou libovolně zkoumat a upravovat.[2] Mezi hlavní výhody open source patří:

- **Spolehlivost** - Systémy s otevřeným kódem bývají zpravidla velmi kvalitně zpracované a spolehlivé, jelikož se na jeho vývoji podílí velká skupina vývojářů. Vývojáři chyby opravují většinou sami a nemusí čekat na nové verze systému. Z toho vyplývá, že chyby se opravují rychleji než bývá u proprietárního software.
- **Zdarma** - Většina open source bývá naprosto zdarma, platí se pouze za podporu, nebo vůbec. Toto má za důsledek ušetření času a financí za vlastní vývoj systémů. Pokud hraje rychlost uvedení produktu na trh klíčovou roli, je to značné plus při jeho vývoji.

Mnoho společností se domnívá, že při využívání open source odhalí důvěryhodné firemní informace jejich konkurentům, takže se open source vyhýbají. Pokud firma nabídne software jako open source, může tím zajistit podporu i poté, co dodavatel ukončí svou činnost. Tento model si nezakládá na prodeji tohoto produktu, ale spíše na prodeji podpory a školení k tomuto produktu.

Při použití open source se musí brát ohled na licenci, pod kterou byl vydán. Zdrojové kódy jsou volně dostupné, ale to neznamená, že s nimi můžeme dělat, co nás napadne. Vývojář kódu (společnost nebo jednotlivec) tento kód z právního hlediska pořád vlastní.[2] Využití tohoto kódu můžeme omezit určitou licencí, která stanoví pravidla, za kterých

může být kód využíván a upravován. Většina licencí open source je založena na jednom ze tří obecných modelů:

- **GPL** - jedná se o licenci "reciproční", což znamená, že pokud využijeme software s licenci GPL, musíme svůj software rovněž poskytnout jako open source.
- **LGPL** - jedná se o variaci licence GPL, která nám umožňuje psát části kódu, který využívá open source bez nutnosti tento kód publikovat. Případné změny licencovaného kódu je potřeba publikovat.
- **BSD** - jde o "nereciproční"licenci, to znamená, že vývojáři nemají povinnost znovu publikovat změny nebo úpravy kterékoliv části licencovaného kódu. Kód jde zahrnout do proprietárních systémů a ten dále prodávat. Musí se pouze odkázat na původního autora kódu.

2.13 Diagram případů užití

Tento diagram zobrazuje chování systému tak, jak ho vidí koncoví uživatelé. Účelem tohoto diagramu je popsat funkcionalitu systému, tedy co od něj očekáváme. Diagram popisuje, co by systém měl umět, ale nezabývá se implementací této funkčnosti. Jedná se o prvotní návrh funkcionality systému, který je graficky přehledný a srozumitelný.[2] Pro modelování se využívají základní komponenty, a to:

- **Případy užití** - jsou činnosti, které vedou ke splnění stanoveného cíle. Může se jednat o přidání komentáře, smazání uživatel atd., definuje tedy jednu funkcionalitu systému, kterou by měl systém obsahovat. Případ užití je zobrazován jako elipsa s popisem činnosti uvnitř.
- **Aktéři** - jsou role přiřazené osobám nebo jiným částem systému. Aktéři komunikují s jednotlivými případy užití a značí se jako postava s názvem.
- **Vztahy** - mezi aktéry a případy užití. Vztah může dělit na dvě kategorie **«include»** a **«extend»**. Include slouží k popisování vztahů mezi samotnými případy užití. Druhý typ extend slouží k rozšíření chování původního případu užití.
- **Ohraničení** - udává hranice systému či subsystému, který popisujeme.

2.14 Konkurenceschopnost

Protože aplikace vzniká pro potřeby konkrétní společnosti, která se pohybuje v konkurenčním prostředí, popíšeme si také základní metody pro analýzu vnějších a vnitřních faktorů. Tyto metody poté využijeme v praktické části práce a při tvorbě samotné webové aplikace, abychom dokázali aplikaci co nejlépe přizpůsobit společnosti, pro kterou vzniká, tedy pro grafické studio Justmighty.

Nejdůležitější hodnotou pro firmy v tržní ekonomice je jejich konkurenceschopnost, která je stavebním kamenem pro úspěšné fungování v hospodářské soutěži s ostatními podniky. Konkurenceschopnost, či konkurenční výhoda je něco hmotného, či nehmotného, co podnik odlišuje od konkurence. Hmotnými zdroji myslíme budovy, stroje či zařízení, nehmotnými pak především lidské zdroje nebo know-how. [16]

Konkurenční výhody dělíme na vnitřní a vnější. Mezi vnější faktory patří např. příležitosti i rizika odvětví podnikání nebo i vlivy politické, sociální či etické. Mezi vnitřní faktory řadíme pak finanční zdroje, technologie, hodnoty a motivace jednotlivých zaměstnanců nebo jejich schopnosti a dovednosti. [15]

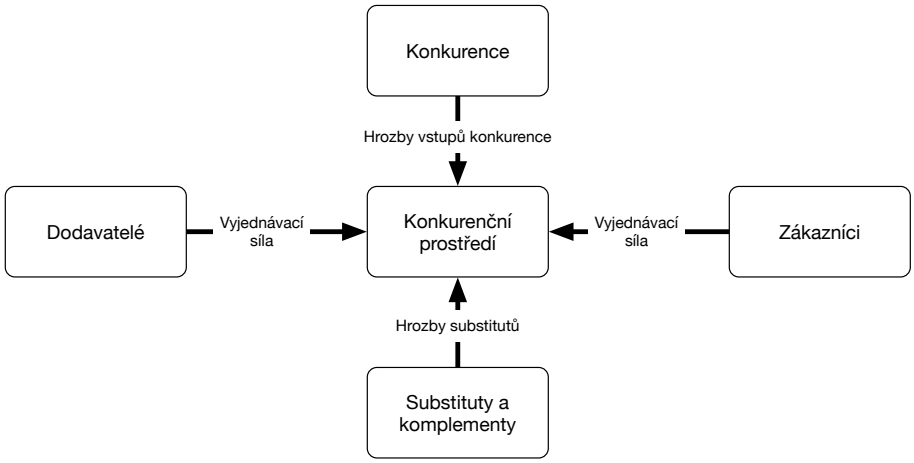
Pro analýzu vnějších hodnot jsme vybrali Porterův model a SLEPT analýzu. K analýze vnitřního prostředí pak model 7s.

2.15 Porterův model konkurenčního prostředí

Porterův model konkurenčního prostředí nebo také sil, slouží k analýze oborového okolí podniku. Model slouží k zmapování konkurenční pozice firmy a velmi často se také využívá i při tvorbě marketingových strategiích. Vychází z předpokladu, že na firmu působí pět vnějších sil, které určují jeho pozici.[14] Jedná se o tyto síly:

- **Vyjednávací síla zákazníků** - závisí především na tom, jestli zákazník může přejít ke konkurenci (srovnatelná cena, kvalita), nebo začít využívat substituty.
- **Vyjednávací síla dodavatelů** - jak moc je firma závislá na svých dodavatelích a zda firma má dostatečné velikosti objednávek, které by mohly ovlivňovat cenu produktů nebo jejich kvalitu.
- **Hrozba vstupu nových konkurentů** - závisí na velikosti fixních nákladů na vstup do odvětví, know-how a licenčních smlouvách.
- **Hrozby substitutů** - pokud je možné vyrábět substituty s nižšími náklady, než je cena současných výrobků, a užitek je srovnatelný.

- **Rivalita firem na daném trhu** - záleží na celkové velikosti trhu, ziskovosti a trendech v odvětví.



Obrázek 2.4: Porterův model konkurenčního prostředí (Zdroj: vlastní)

2.16 SLEPT analýza

Slouží k analýze makrookolí společnosti. Zkoumá, jak silně je podnik ovlivňován okolím, hodně záleží na předmětu podnikání.[16] Jedná se o faktory:

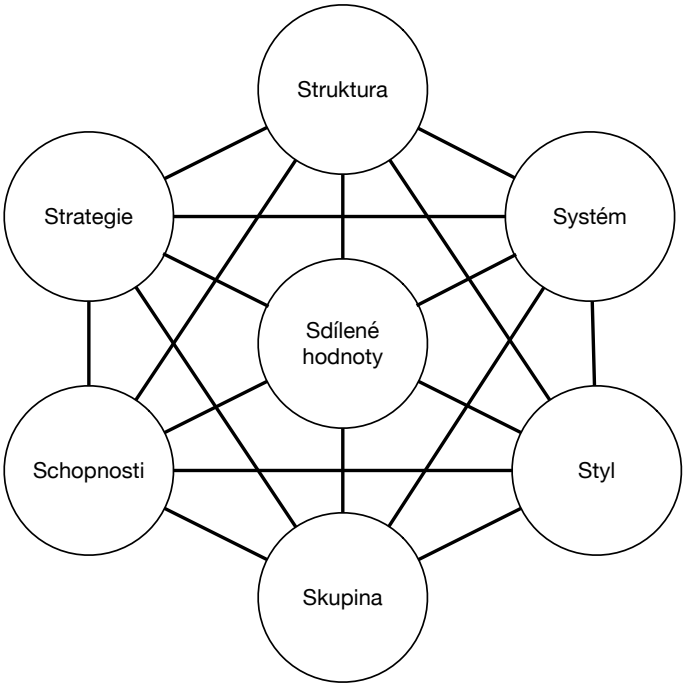
- **Sociální** - změny ve společnosti, náboženství, rodinné hodnoty, vzdělanost obyvatelstva
- **Legislativní** - zákony, vyhlášky, regulace
- **Ekonomické** - úrokové sazby, hospodářský růst, inflace, směnné kurzy
- **Politické** - politický vliv na podnikání, monetární a fiskální politika, podpora zahraničního obchodu
- **Technologické** - postoj k vědě a výzkumu, nové pracovní postupy, metody a technologie

2.17 McKinsey 7S

Slouží k analýze interních faktorů společnosti a jejich změn. Je založen na sedmi elementech, které jsou seřazeny a vzájemně se podporují.[17] Mezi tyto elementy patří:

- **Strategie** - obsahuje misi firmy, její dodržování a dosahování stanoveného cíle.

- **Struktura** - definuje pravomoci a organizační struktury ve společnosti.
- **Systém** - řízení popisuje metody, postupy, procesy a využívané informační systémy a technologie.
- **Styl** - řízení určuje, jak manažeři komunikují se zaměstnanci. Řeší efektivnost řízení společnosti, správnost a rychlost rozhodování.
- **Skupina** - určuje jejich specializaci, motivaci a možnosti osobního rozvoje. Stanoví možnost jejich uplatnění nebo také případné riziko.
- **Schopnosti** - jedná se o všechny schopnosti, dovednosti, znalosti a zkušenosti zaměstnanců a celé společnosti. Jde tedy o celé know-how společnosti.
- **Sdílené hodnoty** - obsahují a definuje firemní kulturu, základní hodnoty a vizi firmy.



Obrázek 2.5: McKinsey 7S (Zdroj: vlastní)

Před tvorbou samotné aplikace je nutné udělat pečlivou analýzu prostředí, ve kterém samotná webová aplikace vzniká. Nejrozšířenější a základní metodou pro tuto analýzu je tzv. SWOT.

2.18 SWOT analýza

SWOT analýza spočívá ve vytvoření matice se čtyřmi kvadranty, kde je každý kvadrant jeden z faktorů. Faktory jsou:

- **Silné stránky** (strenghts)
- **Slabé stránky** (weaknesses)
- **Příležitosti** (opportunities)
- **Hrozby** (threats)

SWOT analýza tedy slouží k identifikaci silných a slabých stránek, příležitostí a hrozeb firmy.[13] Skládá se z OT analýzy - příležitostí a hrozeb, které přichází z okolního prostředí. Jedná se o makroprostředí (politicky-právní, ekonomické, sociálně kulturní a technické) a mikroprostředí (zákazníci, dodavatelé, odběratelé). Kritické pro správné provedení SWOT analýzy je sestavení její matice, při které je potřeba držet se základních kroků. Musíme si nejdříve stanovit cíl, kterého chceme pomocí SWOT analýzy dosáhnout. Druhým krokem je pak identifikovat a vyhodnotit silné a slabé stránky.

Obrázek 2.6: Swot analýza (Zdroj: vlastní)

3. Analýza problému a současné situace

V následující kapitole budeme hodnotit aktuální stav společnosti Justmighty a její postavení na trhu, proto, abychom webovou aplikace zcela přizpůsobili potřebám společnosti Justmighty a přispěla tak ke zefektivnění procesů uvnitř firmy a tím se zvýšila její konkurenceschopnost. Pomocí analýzy vnějšího a vnitřního prostředí (SLEPT, Porterův model, 7s) budeme zjišťovat, zda je společnost schopná zpracovat novou aplikaci na správu času, a co všechno musí webová aplikace splňovat, aby byla implementace co nejméně náročná.

Pomocí analýzy SWOT také zhodnotíme prostředí a nejbližší konkurenty naší aplikace. Porovnáme silné a slabé stránky, které nám umožní se od nich odlišit a získané informace nám následně poslouží při návrhu řešení.

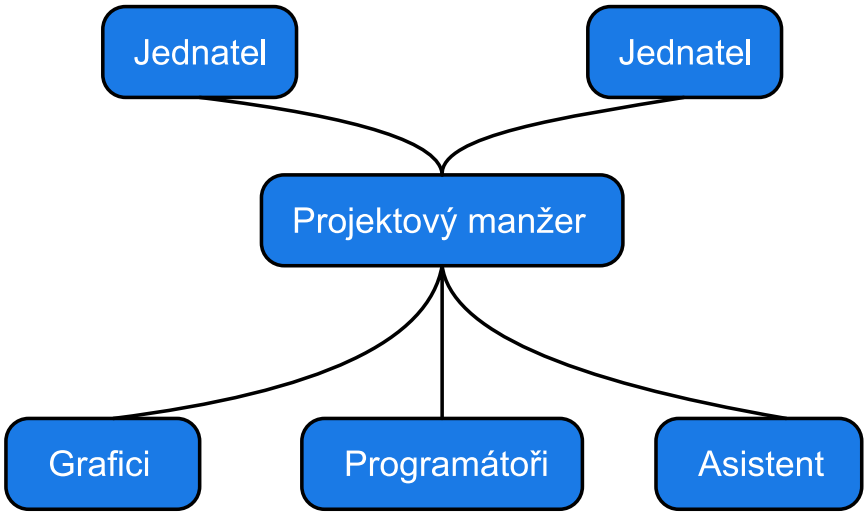
3.1 Společnost Justmighty

Společnost Justmighty byla založena v roce 2014 v Brně. Jedná se o grafické studio, které se specializuje především na tvorbu firemní identity a webových prezentací. Všechny produkty jsou tvořené přesně na míru zákazníka a snaží se dosahovat nejvyšší kvality. S těmito předpoklady je spojeno finanční ohodnocení, za které jsou tyto služby nabízeny.

Další činnosti, které se společnost Justmighty věnuje, je online marketingu. Pomocí on-line marketingu Justmighty svým klientům propaguje produkty, ať placenou reklamou ve formě PPC nebo pomocí organického vyhledávání. Všechny informace, které získá touto formou, se snaží využít ke zlepšení výsledného produktu. Proto je u každé webové prezentace sledováno chování zákazníků a její návštěvnost. Všechny tři hlavní činnosti branding - tvorba webu - on-line marketing na sebe navazují a společnost je tak schopna nabízet ucelený balíček služeb.

3.1.1 Organizační struktura

O vedení společnosti se starají dva jednatelé, kteří jsou současně vlastníci. Jelikož tým se skládá pouze z 8 lidí, jednatelé se zároveň účastní vlastní práce. O řízení projektů se stará projektový manažer, který vede tým programátorů a grafiků. Posledním členem týmu je asistent, který se stará o administrativní činnosti celé firmy. Strategické řízení firmy mají na starosti jednatelé.



Obrázek 3.1: Organizační struktura Justmighty (Zdroj: vlastní)

3.1.2 Analýza 7S

Strategie

Hlavní cíl firmy Justmighty je stát se jednou z nejznámějších digitálních a internetových agentur v České Republice. Zajímá se především o tvorbu webových prezentací a jejich marketing. Snaží se dělat kvalitní produkty, které mají zaujmou na první pohled. Těchto cílů dosahuje s rostoucím počtem klientů a sledovaností na sociálních sítích.

Struktura

Firma Justmighty má dva jednatele, kteří jsou vlastníci firmy. O chod firmy a hladký průběh všech projektů se stará projektový manažer. Ten organizuje veškerou činnost a posloupnost projektů. Firma v současné době má 8 členů, kteří zastávají více pozic najednou. Tento problém vzniká jelikož je firma malá a nedokáže pokrýt všechny potřebné pozice. Všichni zaměstnanci jsou součástí tvorby nových a úpravě stávajících produktů, ale konečné slovo mají vždy jednatelé společnosti.

Systém

Justmighty nepoužívá žádný složitější ERP systém, pro organizaci všech úkolů slouží převážně Office Suite od společnosti Google. Využívá ho na správu dokumentů, tabulek a také všech dat. Pro organizaci času firma využívá momentálně jenom kalendář, jeli-

kož není na trhu žádný produkt, který by ji vyhovoval. Je to jeden z důvodu, proč sáhla k vytvoření aplikace Justimio.

Nejvíce používaná technologie pro tvorbu webových prezentací je Wordpress. Důvod tohoto využití jsou především skoro nulové náklady na vývoj a údržbu. Pro sledování výkonnosti jednotlivých webových prezentací se používá Clicky. Slouží k úpravám webů pro lepší konverze a uspokojení více návštěvníků. Placená propagace webových stránek je především pomocí PPC (Pay Per Click), využívají se služby Google adwords a Facebook ads. Propagace organickou formou je především pomocí optimalizace pro webové prohlížeče nebo SEO (Search Engine Optimization).

Každá tvorba webových stránek prochází analýzou trhu, která zaručí zacílení webu a jeho potřeby. Pomocí těchto podkladů se začíná tvořit grafický návrh webu, který má většinou několik iterací. Po ukončení designu se web předává kodérům kteří ho přetvoří do kódu. Poté nastává fáze spouštění, kdy se začínají sledovat metriky webu a chování návštěvníků. Tyto podklady se poté využívají k úpravě webu lepší výkonost. Následně se vytvoří marketingové kampaně, které zaručí rychlejší přístup návštěvníků a dosažení požadovaných cílů.

Styl

Organizaci práce a komunikaci se zákazníky má na starost projektový manažer. O vedení firmy se stará s pomocí jednatelů, kteří chtějí spíše kreativně tvořit než se starat o vedení společnosti. Rozhodování je rychlé, jelikož konečné slovo mají ve společnosti jednatelé. Firma je vedena v přátelském duchu se stanovenými pravidly, kdy každý zaměstnanec může mít slovo a nebojí se říct svůj názor. Z důvodů nízkého průměrného věku všech zaměstnanců (23 let), firma vystupuje nejen v interní komunikaci, ale i před klienty neformálně. Typickým znakem neformálního vystupování je neexistující dresscode a přátelské vystupování v jak e-mailové, tak v osobní komunikaci.

Skupina

Jelikož firma má malý počet zaměstnanců, musí zaměstnanci zastávat více pozicí ve firmě. Všichni zaměstnanci jsou proto všestranní a chybí jím úplná specializace v určitém oboru. Největší motivací spolupracovníků je možnost osobní realizace a adekvátní finanční ohodnocení.

Schopnosti

Firma má především mladé zaměstnance, proto nemá tolik zkušeností. Velkou výhodou je naopak obrovské know-how v nových trendech a v internetovém marketingu. Mezi hlavní schopnosti patří tvorba moderních webových stránek, kvalitní a funkční design nebo vývoj.

Sdílené hodnoty

Firma si zakládá na velmi přátelské firemní kultuře. Jedná se o především o mladý tým, který se snaží dravě posouvat dopředu a tlačit firmu před sebou. Mezi základní hodnoty patří především kvalitně odvedená práce, za kterou se firma nikdy nemusí stydět. Mezi další hodnoty je kvalitní design, který osloví všechny potencionální zákazníky a návštěvníky. Vizí firmy je stát se uznávanou internetovou agenturou ve světě a odvádět co nejlepší řemeslo v daném oboru. Základním stavebním kamenem pro dosažení tohoto cíle je tým mladých nadšených lidí.

3.2 Požadavky na aplikaci Justimio

S dynamicky rostoucí velikosti firmy (noví zaměstnanci, více zakázek, větší obrat), se objevil problém s organizací času. Jednatelé společnosti přišli s nápadem vytvořit aplikaci na správu času a organizaci jednotlivých úkolů pro klienty, protože žádná aktuální aplikace nesplňovala jejich požadavky.

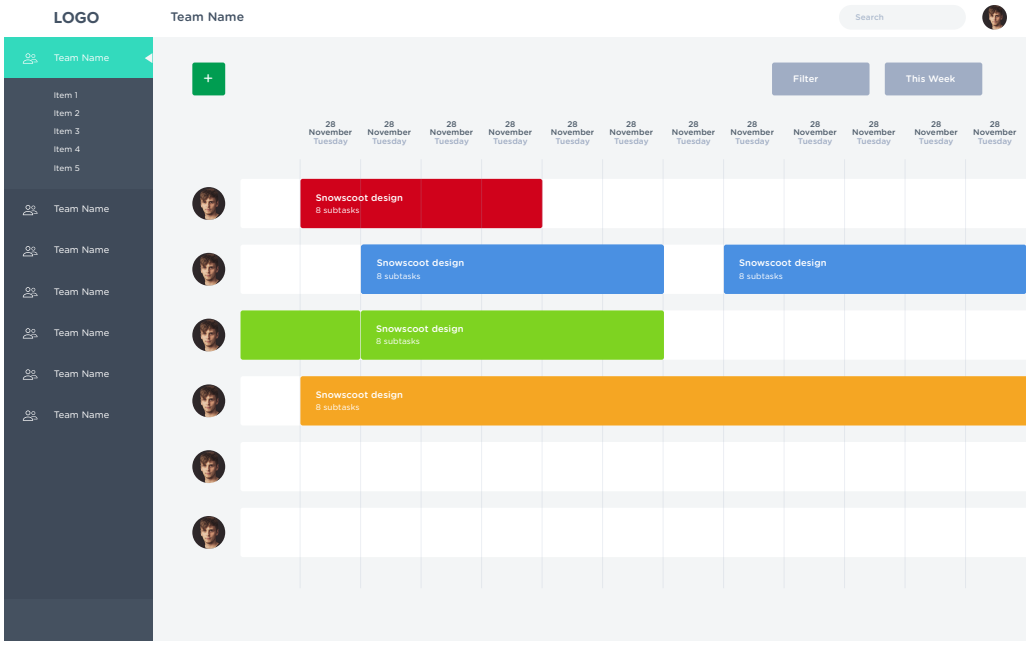
Aplikace by měla sloužit pro malé až středně velké týmy, které současně pracují na více projektech najednou. Současná práce na více projektech vyžaduje složité rozdělování úkolu a vymezování jejich časových rámců. Tato aplikace by měla usnadnit práci projektovým manažerům a vedení společnosti, aby lépe organizovali své zaměstnance. Aplikace by měla přehledně zobrazovat, který člen týmu pracuje na jakém úkolu a kolik mu zbývá času na jeho dokončení.

Proto vznikl návrh aplikace, která by měla v jednoduchém kalendářovém zobrazení umožňovat zobrazovat úkoly na více dnů. Aplikace bude umožňovat následující funkce:

- **Dlouhodobý úkol** - Možnost vytvoření dlouhodobého úkolu, který bude začínat v jednom dni a bude trvat x-dnů. Úkol může být také přerušen na několik dnů.
- **Podúkol** - Každý dlouhodobý úkol v sobě může mít x - malých podúkolů, které bude možné označit po dokončení jako splněné. Krátkodobým úkolů může být přidělen konečný termín a osoba, která na něm pracuje.

- **Kalendářové zobrazení** - Úkoly by měly být zobrazeny v kalendáři pro přehlednost a lepší plánování časových mezer. Kalendářové zobrazení by mělo umožňovat měnit časové rámce zobrazení.
- **Hromadná spolupráce** - Aplikace by měla zvládat více uživatelů a přeřazování úkolů mezi jednotlivými uživateli nebo práci více uživatelů na stejném úkolu.
- **Organizace týmů** - Možnost shlukovat uživatele do skupin, například: vývojáři, grafici, ekonomické oddělení. Lepší organizace ve firmách s vyšším počtem uživatelů.
- **Výstupy** - Možnost získat výstupy a jednoduché přehledy o splněných projektech.

Z těchto předpokladů vznikl původní návrh uživatelského rozhraní aplikace. Aplikace bude rozdělena na tři části. První část je hlavička stránky, kde budou všechny potřebné informace o uživateli a název aktivní společnosti, ve které se nachází. Druhá část tvoří menu, kde si uživatel může vybírat mezi společnostmi, ve kterých je členem. V tomto menu si může vybírat i zobrazení jednotlivých týmů ve společnosti. Třetí a hlavní část tvoří kalendářové zobrazení, kde budou přehledně zobrazeni členové týmu a činnosti, na které právě pracují. Aplikace umožní filtrovat toto zobrazení podle času a dalších kritérií.



Obrázek 3.2: Prvotní návrh uživatelského rozhraní aplikace Justimo (Zdroj: Tomáš Pohl)

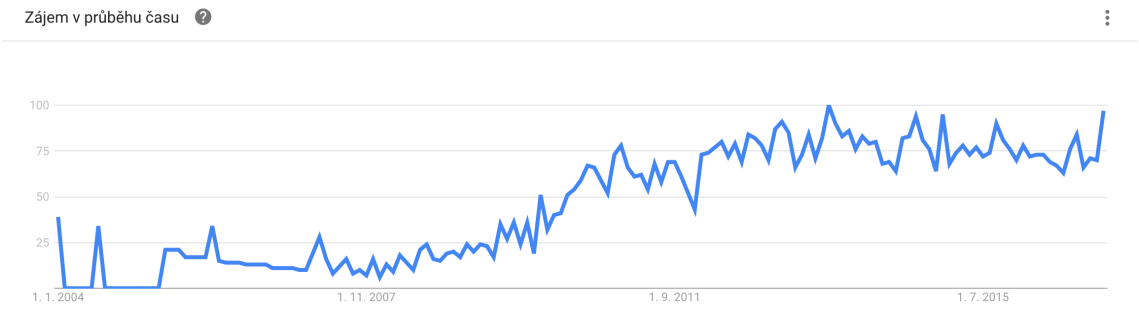
Celá aplikace bude přístupná z jakéhokoliv zařízení, a proto musí být uzpůsobeno její uživatelské rozhraní. Toto uživatelské zobrazení musí být plně responzivní a musí

být kladen důraz na finální design aplikace. Po celkovém otestování funkčnosti aplikace uvnitř firmy by měla být zpřístupněna široké veřejnosti za měsíční poplatek.

3.3 Konkurenční aplikace

Mezi největší konkurenty patří aplikace, které mají webové rozhraní. Tyto aplikace umožňují přístup z jakéhokoliv zařízení a proto je může používat kdokoliv, jak uživatel s telefonem nebo počítačem. Z tohoto důvodu je z konkurence vyřazena například aplikace MS Project od firmy Microsoft, která není bohužel multiplatformní[1] a nedokáže oslovit všechny potenciální zákazníky.

V současné době existuje velké množství organizačních aplikací, které nám umožňují kvalitně plánovat a organizovat čas. Jedná se o aplikace které jsou založeny na systému GTD, nebo využívají pouze klasického kalendáře. Zájem o tento typ aplikací je v dnešní době stále velký, můžeme to vidět na následujícím grafu z Google trendů:



Obrázek 3.3: Celosvětová vyhledávanost aplikací typu todo list ke dni 16.1 2017 (Zdroj: vlastní)

Většina aplikací se snaží zasáhnout co nejširší skupinu cílových uživatelů a prokrýt co největší část trhu. Z tohoto důvodu se snaží implementovat co nejvíce možných funkcí. Z důsledku takového chovaní se tyto aplikace přetvářejí na velké informační systémy. Tyto systémy obsahují datové sklady, umožňují komunikovat uživatelům pomocí real-time chatu a podobných funkcí. Aplikace se z tohoto důvodu stávají nepřehledné a moc komplikované. Původní význam organizace času se pomalu vytrácí.

Hodnocení konkurence provedeme pomocí požadovaných vlastností, které by měla naše aplikace splňovat. Hodnotíme, jestli je konkurence může provést a v jakém mě-

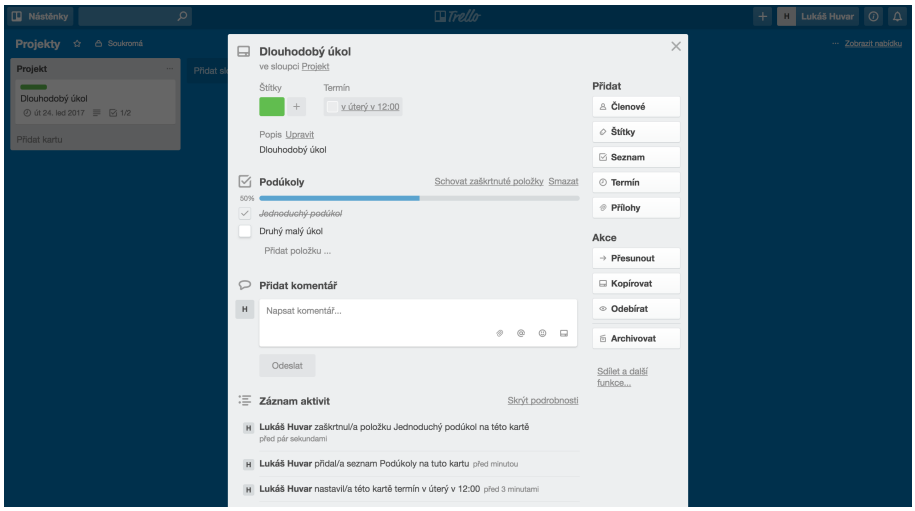
[1]Technické požadavky na MS project: `https://www.microsoftstore.com/store/mseea/cs_CZ/pdp/Project-Professional-2016/productID.324452600`

řítku. V neposlední řadě bude vytvořena SWOT analýza na každou konkurenční aplikaci, abychom zjistili slabé a silné stránky této aplikace.

3.3.1 Trelo

Aplikace Trelo byla spuštěna v roce 2011 jako webová aplikace a následně byla vytvořena aplikace na Iphone. V následujícím roce aplikace dosahuje hranice 500 tisíc aktivních uživatelů a připojuje aplikaci na Android. V roce 2014 její uživatelská základna dosahuje už 4.75 miliónů uživatelů. V roce 2017 společnost kupuje firma Atlantis, která se stará o produkty HipChat, Confluence a JIRA.[18] Zde můžeme vidět, že se aplikace stane součástí těchto velkých informačních systémů.

Aplikace Trelo funguje na jednoduchém principu nástěnek, kde si vytváříme sloupce a mezi nimi můžeme přesouvat úkoly nebo karty. Na následujícím obrázku si můžeme prohlédnout vytváření úkolů a celkový vzhled aplikace:



Obrázek 3.4: Zobrazení vytváření úkolu v aplikaci Trelo (Zdroj: Vlastní)

Chybějící vlastnosti

Aplikace Trello splňuje velké množství požadovaných vlastností, ale chybí jí jeden z hlavních požadavků a to je přehledné kalendářové zobrazení.

- **Dlouhodobý úkol** - Nelze přidělit termín začátku úkolu ani nejde přerušit úkol.
- **Podúkol** - Nejde přidělit termín dokončení ani osobu, která na něm pracuje.
- **Kalendářové zobrazení** - Vidíme úkol jen na posledním dni trvání a nemáme přehled, kdo pracuje na jakém úkolu.

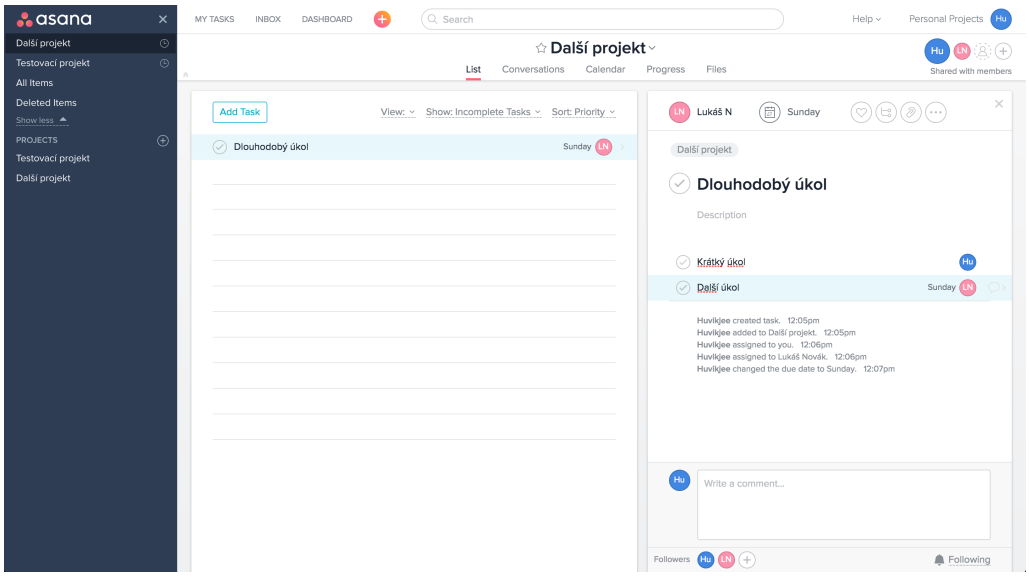
- **Výstupy** - Chybí přehledný výstup, jedinou formou je JSON soubor, který pro zhodnocení splněných a nesplněných úkolů je zbytečný.

SWOT analýza Trelo

- **Silné stránky** - Mezi nejsilnější stránky patří velký počet uživatelů a silné postavení na trhu.
- **Slabé stránky** - Slouží jako záznamník úkolů a není uzpůsobena na organizaci času. Chybí jí kvalitní kalendářové zobrazení.
- **Příležitosti** - Pomocí vylepšení se dají přidělat další funkčnosti této aplikace, tímto způsobem se dále rozšiřuje.
- **Hrozby** - Možnost pohlcení know-how společností Atlantis, následně transformování do vlastní platformy a ukončení fungování této aplikace.

3.3.2 Asana

Tato aplikace vznikla v roce 2008, jako interní nástroj pro organizaci času ve Facebooku. Její autoři se poté se odpojili a vytvořili vlastní firmu Asana, která zastřešila tento nástroj. Tato aplikace slouží převážně jako nástroj pro organizaci úkolů a činností v týmech.



Obrázek 3.5: Vytváření úkolů v aplikaci Asana (Zdroj: Vlastní)

Chybějící vlastnosti

Asana postrádá přehledné kalendářové zobrazení a lepší práci s více uživateli.

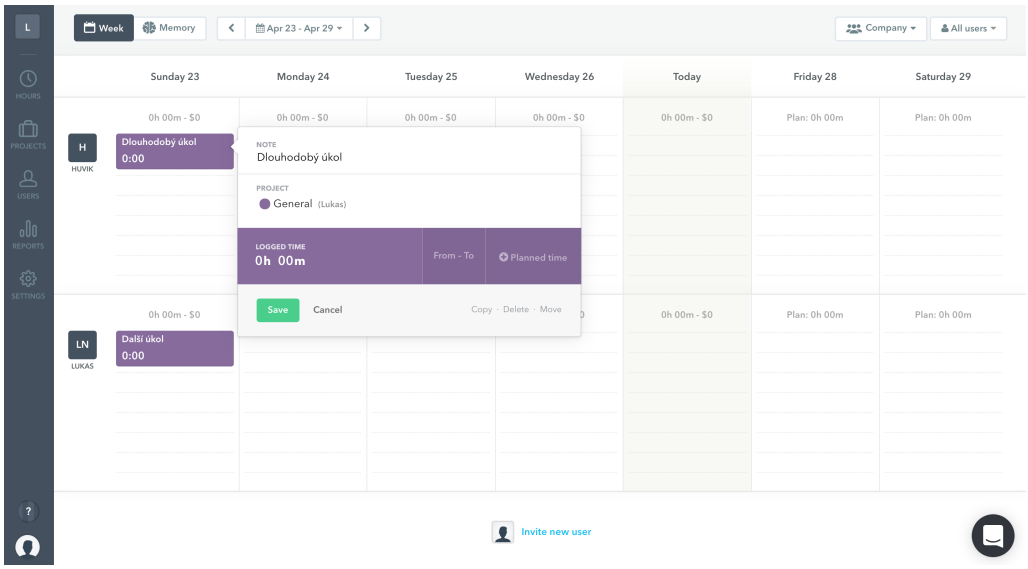
- **Dlouhodobý úkol** - Nelze přidělit termín začátku úkolu ani nejde přerušit úkol.
- **Kalendářové zobrazení** - Úkoly jdou zobrazit v kalendáři, ale pouze jeho konečné datum, není zobrazeno, kdy úkol začal ani délka jeho trvání.
- **Organizace týmů** - Uživatelé nejdou spojovat do týmů, není přehledné, kdo pracuje v kterém týmu a na jakém úkolu .

SWOT analýza Asana

- **Silné stránky** - Know-how, které společnost nabyla za dlouhodobé působení na trhu. Kvalitní design aplikace, který dodává přehlednost aplikace.
- **Slabé stránky** - Tato aplikace je především pro velké týmy a organizace probíhá převážně pomocí projektů a ne týmů. Chybí kalendářové zobrazení.
- **Příležitosti** - Možnost rozšiřovat funkčnost aplikace přes doplňky.
- **Hrozby** - Ztráta určitého počtu uživatelů, kteří potřebují určitou funkcionalitu, kterou vytvoří nová konkurence.

3.3.3 Timely

Tato aplikace se převážné zaměřuje na monitorování činností strávených na projektu. Aplikace vznikla v roce 2013 v Oslu. V roce 2016 získala milion dolarů od investorů.[19]



Obrázek 3.6: Vytváření úkolů v aplikaci Timely a kalendářové zobrazení (Zdroj: Vlastní)

Chybějící vlastnosti

Jelikož je aplikace zaměřená na monitorování času stráveného na úkolech. Nejde monitorovat úkoly, které trvají více než jeden den. Proto chybí smysluplné plánování dlouhodobých úkolů.

- **Dlouhodobý úkol** - Není možné vytvořit úkol delší než 1 den.
- **Podúkol** - Úkoly nemůžou mít podúkoly, špatný přehled o odvedené práci. Nejde přiřadit osobu, která na něm pracuje.
- **Hromadná spolupráce** - Na jednom úkolu nemůže pracovat více než jedna osoba.
- **Organizace týmů** - Nelze tvořit týmy lidí a mít přehled, kdo je členem jakého týmu.

SWOT analýza Timely

- **Silné stránky** - Kalendářové zobrazení, ve kterém zjistíme zařazení člena do týmu a na jakém úkolu pracuje.
- **Slabé stránky** - Slouží převážně k měření času stráveného na úkolu.
- **Příležitosti** - Přidáním dalších funkcionalit by se naskytla možnost získat platící zákazníky od velkých hráčů na trhu.
- **Hrozby** - Za aplikací stojí malá firma, u které může nastat problém s financováním provozu a vývoje aplikace.

3.4 Analýza prostředí aplikace Justimio

V této kapitolo si nejprve analyzujeme vnější prostředí společnosti pomocí Porterovy analýzy a následně pomocí SLEPT analýzy.

3.4.1 Porterův model sil

Vyjednávací sílu zákazníků

Aplikace Justimio bude vstupovat na velice konkurenční trh. Na světovém trhu jsou desítky aplikací na organizaci času, které jsou pro zákazníka na první pohled téměř totožné,

proto při výběru podobné aplikace prochází zákazník komplikovaným rozhodovacím procesem. Až po implementaci konkurenčních aplikací zákazník často zjišťuje, že aplikace není nastavená pro jejich potřeby. Proto je potřeba vytvořit kvalitní aplikaci, která bude použitelná pro co nejvíce firemních prostředí a procesů pro malé a střední firmy. Vzhledem k velkému konkurenčnímu prostředí bude nutné nabízet aplikaci za adekvátní cenu.

Vyjednávací sílu dodavatelů

Jelikož firma ani budoucí aplikace není závislá na datech třetích stran, nebo na nějaké formě dodávaných informací a služeb, můžeme konstatovat, že dodavatelé nemají skoro žádnou sílu při tvorbě a následném prodeji aplikace.

Hrozba vstupu nových konkurentů

Hrozba vstupu nových konkurentů je středně vysoká. Vstup nových konkurentů zahrnuje vysoké znalostní a technologické požadavky. Tvorba podobné aplikace je zároveň velmi náročná na čas a lidské zdroje.

Hrozby substitutů

Jak jsem zmínil v kapitole 3.3 substitutů existuje velké množství, ale žádný se nespecializuje na naší cílovou skupinu. Všechny jsou převážně obecné a snaží se oslovit, co nejvíce zákazníků.

Hrozbou se může stát, pokud se jeden z konkurentů rozhodne zaměřit se na stejnou cílovou skupinu, na jakou cílí aplikace Justimio.

Rivalita firem na daném trh

Trh je velmi koncentrovaný, velký podíl na něm mají společnosti jakými jsou Google, Microsoft nebo Asana. Vedle těchto velkých hráčů existují desítky menších aplikací, které se pro sebe snaží získat každého nového zákazníka. Právě tyto menší firmy a jejich aplikace dynamicky hýbou trhem a jsou pro velké firmy čím dál větším konkurentem.

3.4.2 SLEPT analýza

V následující kapitole se zkoumá dopad makrookolí na aplikaci Justimio. Tyto výsledky jsou brány v potaz při implementaci a návrhu.

Sociální

S neustálým tlakem na kvalitní životní úroveň jsou lidé čím dál více pod časovým stresem. Tento tlak většinou vede ke smysluplnému organizování času, který se snažíme co nejlépe využít.

Proto se používají čím dál častěji diáře a aplikace na organizování času. Bohužel většina těchto způsobů nefunguje pro organizaci více lidí v týmu. Mít tyto informace aktuálně k dispozici je klíčovým faktorem, protože umožňuje rychlejší adaptování se na aktualní situaci a tím šetří čas do budoucna.

Všechny tyto důvody mají pozitivní dopad na vznik aplikace Justimo, jelikož nám usnadní mnoho práce.

Legislativní

S ohledem na legislativu je největším problémem zákon o ochraně osobních údajů. Velký důraz musí být kladen na bezpečnost aplikace, aby nedošlo k úniku citlivých dat. Všechny změny týkající se zákonů, můžou mít důsledek na změněnu aplikace a tím, vznikající náklady na úpravu a servis.

Ekonomické

Z ekonomického hlediska patří mezi hlavní problémy časová náročnost tvorby a nedostatek lidských zdrojů. Nulová nezaměstnanost v oboru informačních technologií zvedá neustále platové ohodnocení IT specialistů. Aktuálním problémem v oblasti personálních zdrojů u IT specialistů je i tzv. head hunting (lovení hlav), a tedy přetahování specialistů z jedné společnosti do druhé. Z odchodem zaměstnance ztrácí společnost nejen lidskou sílu ale i know-how. Zde může nastat největší problém, že by společnost nemusela vývoj aplikace dokončit nebo by mohly by nastat prodlevy ve vývoji.

Celý vývoj aplikace bude společnost platit ze svých prostředků a nevyužije žádné externí financování. Všechny využité technologie jsou zdarma, náklady tedy budou plynout z platů zaměstnanců, provozu serverů a s tím spojených služeb.

Politické

Politická situace v oblasti informačních technologií není příliš regulovaná. Internet je celkově svobodný a neexistuje v podstatě skoro žádná cenzura, nebo další negativní politické opatření.

Technologické

Celé odvětví okolo informačních technologií je nejrychleji technologicky rostoucím odvětvím. Jakékoliv sebemenší zaostání za konkurencí může mít fatální následky. Proto je nutné sledovat aktuální trendy a reagovat na změnu trhu. Využitím nových technologií můžeme získat velkou převahu nad konkurencí a získat nové potencionální zákazníky.

3.5 SWOT analýza Justimio

Silné stránky

Silnou stránkou aplikace Justimio bude převážně kalendářové zobrazení, které zaručí přehled, kdo pracuje na jaké činnosti. V kalendářovém zobrazení bude možnost filtrovat členy podle jména a také týmu, ve kterém pracuje. Další silnou stránkou této aplikace bude jednoduché a přehledné uživatelské rozhraní, které bude plně responzivní. Celá aplikace bude fungovat ve webovém prohlížeči, a proto není potřebná instalace této aplikace. Bude fungovat na všech zařízeních, které mají webový prohlížeč.

V neposlední řadě je silnou stránky této aplikace i zacílení na přesně danou skupinu malých a středních firem (do 100 zaměstnanců). Toto nám umožní se odlišit od konkurence a přilákat nové uživatele.

Slabé stránky

Slabou stránkou aplikace se může stát omezené technologické a finanční zázemí malé agentury Justmighty, kde aplikace vzniká. Protože se bude aplikace vyvíjet pro potřeby specifické cílové skupiny, neexistuje žádné detailnější know-how a nastavené procesy, o které se lze při vývoji opřít.

Příležitosti

Justimio bude cílit na menší a zaměřený trh, který by mohl přilákat uživatele od známějších aplikací. Pokud by si aplikace našla oblibu mezi uživateli vybrané cílové skupiny, mohl by z ní plynout velký finanční příjem. Mezi další příležitosti patří vstup investora, který by výrazně urychlil další vývoj a propagaci aplikace.

Hrozby

Největší hrozba je slabá poptávka cílové skupiny po aplikaci Justimio. Hrozbou může být i špatně navržené uživatelské rozhraní, které by mohlo odradit potencionální klienty.

Mezi hrozby se zcela určitě také řadí konkurence, která sice, jak jsme si řekli výše, zatím nenabízí podobnou aplikaci, ale disponuje často obrovským technologickým zázemím a know-how a na svým produktech neustále pracuje. Mezi další hrozby patří prolomení zabezpečení aplikace a s tím spojený zákon o ochraně osobních údajů, který by mohl stát firmu nemalé finanční prostředky. Poslední hrozbou jsou omezené zdroje společnost Justmighty, které můžou znamenat prodlevu v celkovém vývoji aplikace.

3.6 Souhrn analýzy současného stavu

Po provedení všech analýz jsem dospěl k závěru, že zkoumána společnost má dostatečné předpoklady pro vytvoření nové aplikace Justimio.

Hrozby a rizika při tvorbě zde existují, ale nepřevažují nad příležitostmi, které může tato aplikace v budoucnu přinést. Jedná se jak o zisk finančních prostředků, tak získání většího povědomí ze strany konkurence.

Celý návrh aplikace by se měl odlišovat od konkurence a nabízet novou funkcionalitu, kterou konkurence nemá. Aplikace by měla vycházet z vize firmy Justmighty, která zní: "kvalita nad kvantitou"a je tedy založena na kvalitně odvedené práci. Takový přístup k vývoji aplikace by se měl projevit na inovativním a zajímavém designu aplikace, který dokáže přivést a udržet nové zákazníky.

Z analýzy můžeme usoudit, že systém má velký potenciál. V následující kapitole se budeme zabývat samotným návrhem a popisovat implementaci aplikace Justimio.

4. Vlastní návrh řešení, přínos práce

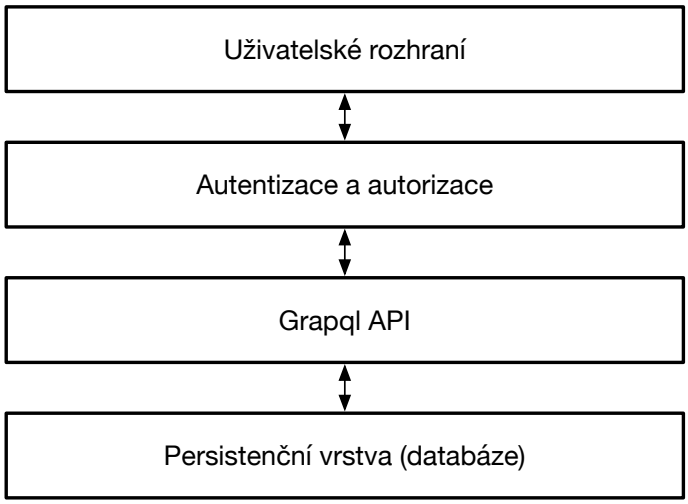
V následující kapitole se pokusíme navrhnout technické řešení aplikace Justimio. Cílem je navrhnout aplikaci, která bude co nejvíce robustní a zároveň co nejsnáze škálovatelná. Tato vlastnost je zásadní, protože dnes nedokážeme určit konečných počet uživatelů.

Celá aplikace bude postavena na moderních technologiích, čímž zaručíme technologickou vyspělost, která bude vždy konkurenceschopná. Zároveň se budeme snažit využít co nejvíce knihoven s otevřeným kódem, abychom ušetřili náklady na vývoji a údržbě informačního systému.

Při návrhu celé aplikace budeme také klást důraz na moderní a přehledný design. Design musí být uživatelsky příjemný, musí být odlišný od konkurence a sledovat aktuální trendy. Mimo jiné i proto bude aplikace navržena jako tzv. "Single page application"(vše na jedné straně), což zaručí co nejpřívětivější ovládání této aplikace.

4.1 Návrh architektury

Navrhovaná architektura bude založena na vrstvené architektuře, abychom oddělili funkcionalitu jednotlivých vrstev. Tento návrh nám umožní pracovat na jednotlivých vrstvách aplikace nezávisle a umožní nám v případě problému některou z vrstev vyměnit aniž bychom tím omezili funkčnost celé aplikace.



Obrázek 4.1: Návrh architektury systému Justimio (Zdroj: vlastní)

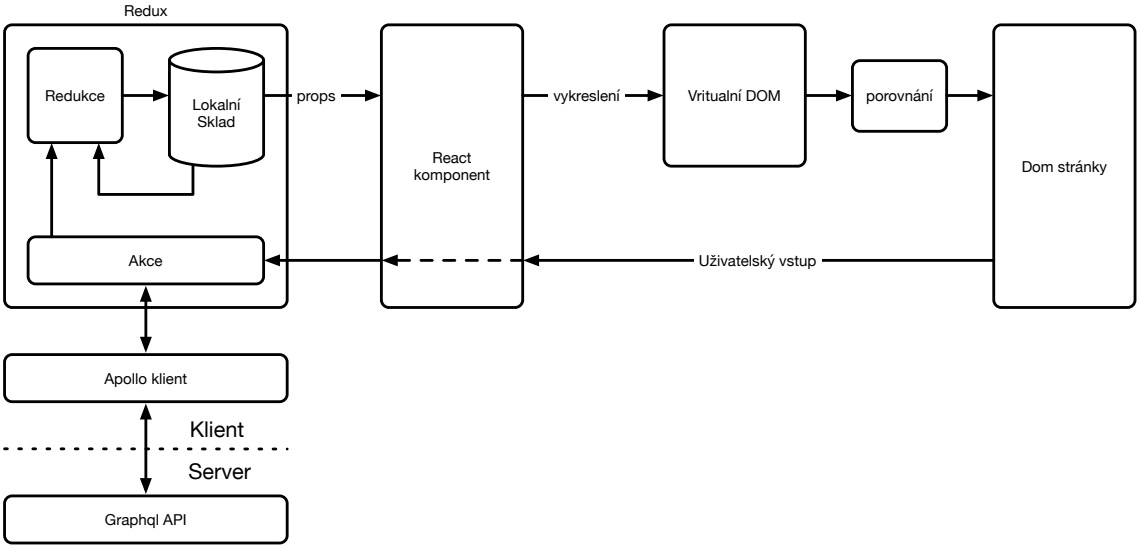
Aplikaci rozdělíme na čtyři vrstvy. První vrstvu bude tvořit klientská část aplikace. Tato část aplikace bude mít webové rozhraní, které je multiplatformní a umožňuje přístup z co největšího počtu zařízení. Pro vytvoření uživatelského rozhraní využijeme knihovnu React, která nám zaručí co nejefektivnější vykreslování a také nejlepší uživatelský požitek.

Druhá vrstva se nám bude starat o bezpečnost celé aplikace. Autentizace a autorizace aplikace zajistí, že k přístupu k API budou mít pouze oprávnění uživatelé.

Třetí vrstva bude tvořena GraphQl API, která bude sloužit jako datový zdroj. Čtvrtou vrstvu bude tvořit databáze, která bude sloužit jako persistenční vrstva pro uchování dat.

4.2 Architektura klientské části aplikace

Klientskou aplikaci jsme rozdělili na tři části. Bude se starat o správné zobrazování dat pro uživatele a také interakci s GraphQl API. Její architektura bude vycházet z modelu MVC, ale řadič a model budou spojené dohromady a tím se od modelu liší. Tvorba podhledů bude v režii Reactu, který se stará o správné zobrazování dat a vytváření odpovídajících komponent. React se bude také starat o uživatelské vstupy, které dále deleguje kontroléru.



Obrázek 4.2: Návrh klientské části aplikace (Zdroj: vlastní)

Model a kontroler bude tvořen knihovnou **Redux**[20], která funguje jako předvídatelný stavový kontejner. Tento kontejner si ukládá stav aplikace v lokálním stavu. Uživatelský vstup nebo GraphQl dotaz vytvoří novou akci. Pomocí této akce a aktuálního stavu aplikace získáme nový stav aplikace. Pro získání nového stavu se používá redukce, která

definuje, co se stane s aktuálním stavem za použití definované akce. Proto vždy s aktuálním vstupem a novou akcí víme, jaký nastane nový stav aplikace. Toto nám zaručí vždy stabilní stav aplikace a nebude docházet k nepředvídatelným situacím.

Pro komunikaci s GraphQl API budeme využívat **Apollo klient**[21], který bude mít na starosti asynchronní operace. Tento klient bude spojovat GraphQl dotazy, pokud více akce vytvoří více dotazů v určitém časovém měřítku. Apollo klient je spojí do jednoho dotazu a budou poslány v jednom velkém kombinovaném dotazu. Tento způsob fungování zvýší celkovou odezvu od aplikace a zpříjemní používaní aplikace koncovému uživateli.

4.3 Infrastruktura

Klientská část aplikace bude fungovat jako javascriptová aplikace, všechna data budou poskytované pomocí jednoduchého webového serveru. Servery poběží na operačním systému Linux (Ubuntu 16.08). Pro dostatečnou dostupnost a prostupnost této aplikace bude zátěž na tyto servery rozprostřena pomocí load balancer serveru.

Třetí a čtvrtá vrstva bude sloužit k uchování dat v databázi. Tyto vrstvy musí být rychle škálovatelné a dosahovat velké propustnosti dat. Jelikož vytvářet kompletní a škálovatelné databáze je velký problém a vyžaduje obsáhlé znalosti v tomto oboru, rozhodli jsme se využít služby **Graphcool**. Tato služba poskytuje databázi jako Saas. Data jsou uložena v clusteru AWS Aurora databázi od společnosti Amazon a využívají Redis caching.[22]

4.4 Zabezpečení aplikace

Jelikož aplikace bude obsahovat citlivá data, veškerý přenos dat bude probíhat po zabezpečeném **HTTPS** protokolu. Veškeré data budou šifrována a nemůže tedy dojít k úniky citlivých informací, jako jsou například přihlašovací údaje do informačního systému.

Pro získání bezpečného certifikátu využijeme službu **Let's Encrypt**[23], která slouží jako certifikační autorita a vydává zabezpečené certifikáty zdarma. Tyto certifikáty jsou validní pouze 90 dní, proto je potřeba aby server každých 60 dnů žádal o nový certifikát, abychom předešli bezpečnostním rizikům a možnému úniku citlivých informací.

4.4.1 Autentizace uživatelů

Autentizace uživatelů se bude provádět pomocí zadání e-mailu a hesla, nebo využitím autentizace služeb třetích stran například Google. Po úspěšném přihlášení se vygeneruje přístupový token. Tento token se následně použije při autorizaci přístupů ke zdrojům.

4.4.2 Autorizace uživatelů

Každý požadavek musí být autorizován, aby nedocházelo ke zneužívání dat a také zbytečnému zatěžování serverů. K autorizaci se využívají přístupové tokeny, které jsou vytvořeny při autentizaci uživatelů. Tento token je posílán v autentizační hlavičce každého požadavku. Token je typu **Bearer**, který ověří, jestli je validní. Tento token uchovává informace o uživateli.

Pomocí tohoto tokenu zjistíme, o kterého uživatele se jedná a jaká má práva k přístupu. Přístupová práva ke zdrojům mají čtyři typy a to:

- **Read** - umožňuje uživatelům číst vytvořené záznamy.
- **Create** - umožňuje uživatelům vytvářet nové záznamy.
- **Update** - umožňuje modifikovat už vytvořené záznamy.
- **Delete** - umožňuje mazat vytvořené záznamy.

Podle typu zdroje jsou přiřazena určitá práva, která uživateli umožňují jeho využívat. Některé záznamy budou omezené přímo na uživatele, který je vytvořil nebo také na skupinu, do které patří.

4.5 Uživatelské role

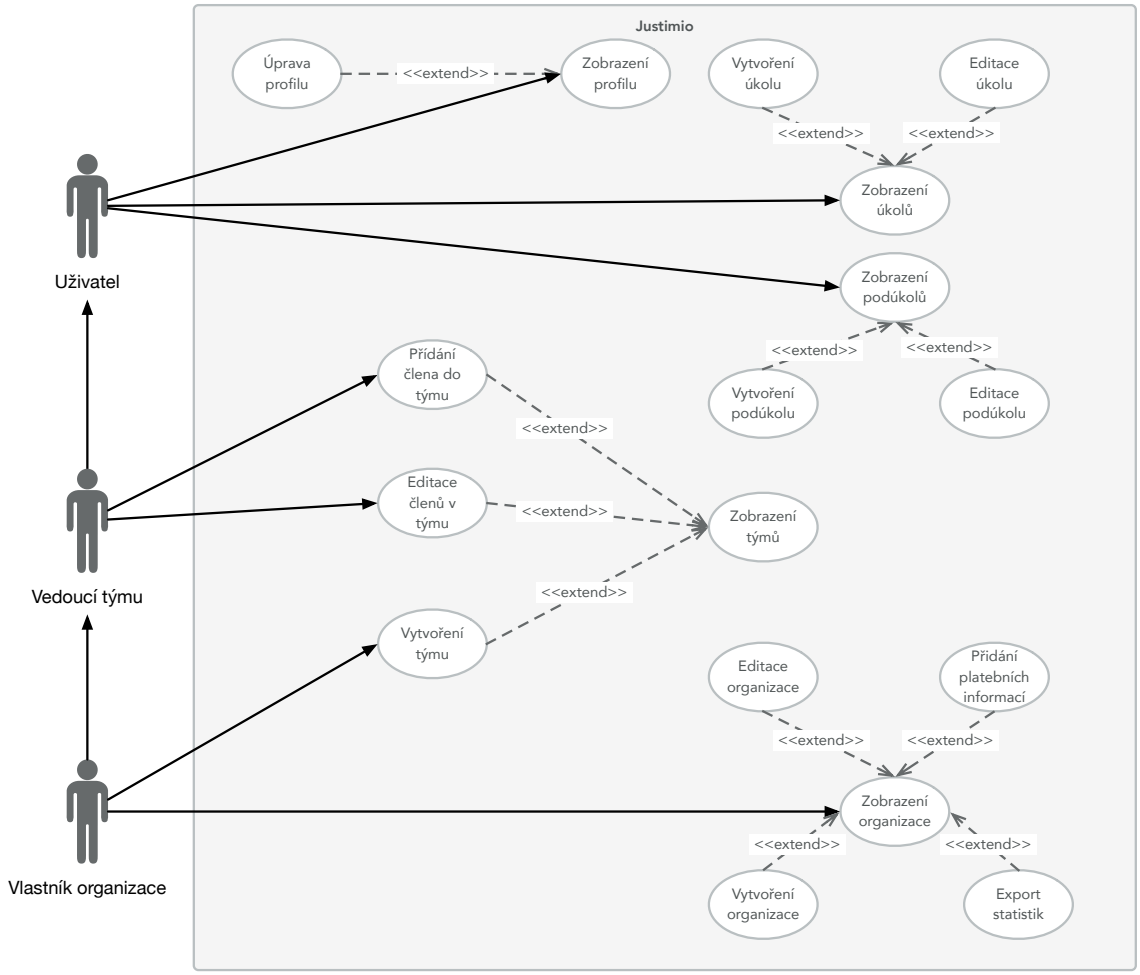
Z předem definované funkčnosti aplikace jsme došli k závěru, že je v systému potřeba mít čtyři druhy uživatelů. Každý druh uživatelů bude mít rozdílné možnosti a jiný případ použití.

- **Uživatel** - jedná se o běžného člena týmu, který může zobrazovat všechny úkoly. Tento uživatel nemá skoro žádné práva. Uživatel může pouze označovat jestli dokončil své dlouhodobé a krátkodobé úkoly.
- **Vedoucí týmu** - může vytvářet nové úkoly pro členy týmu. Každému úkolu přiřadí uživatele, který je zodpovědný za dokončení úkolu. Může měnit délku trvání úkolů a může přerušit vykonávání úkolu. Vedoucí týmu může přidávat nové členy do organizace a přiřazovat je svému týmu.
- **Vlastník organizace** - spravuje celou organizaci. Může přidávat nové členy do týmu. Může vytvářet týmy v organizaci a přidělovat jim vedoucí.

- **Administrátor** - se stará o nastavování cenové politiky systému. Může sledovat statistiky o systému, a to například aktuální počet platících týmů, kolik týmů využívá jaký balíček a celkové informace o informačním systému.

4.6 Případy použití

Pomocí modelu případu užití jsou zachyceny další požadavky na navrhovaný informační systém. Informační systém bude obsahovat 4 aktéry a všechny případy užití vyžadují autentizaci a autorizaci uživatelů. Proto autentizace a autorizace nebude v modelu zobrazena, ale budeme s ní vždy počítat. Po přihlášení do systému budou uživatele v základním rozcestníku nebo taky **Dashboard** 4.7.5.



Obrázek 4.3: Model případu užití (Zdroj: vlastní)

4.6.1 Aktér uživatel

Jedná se o základního člena systému, může pouze pracovat s vlastními daty. Mezi jeho úlohy patří úprava vlastního profilu, práce s úkoly a podúkoly. Může přidávat nové úkoly a editovat. Může pracovat pouze se svými úkoly.

4.6.2 Aktér vedoucí týmu

Tento aktér automaticky dědí veškeré činnosti od uživatele. Jeho úlohy se rozšiřují o možnosti práce s týmem, tedy o přidávání nových členů a jejich editaci. Vedoucí týmu může upravovat a přidávat úkoly dalším uživatelům v týmu.

4.6.3 Aktér vlastník organizace

Vlastník organizace automaticky dědí všechna práva od vedoucího týmu. Vlastník organizace může vytvářet nové týmy v organizaci. Může spravovat celou organizaci (editaci jejich vlastností, změna platebních informací atd.). Vlastník organizace může exportovat statistiky o celé organizaci. Mezi tyto statistiky patří např. počet aktivních členů, počet dokončených úkolů, počet aktuálních úkolu atd. Vlastník organizace může vytvářet další nové organizace.

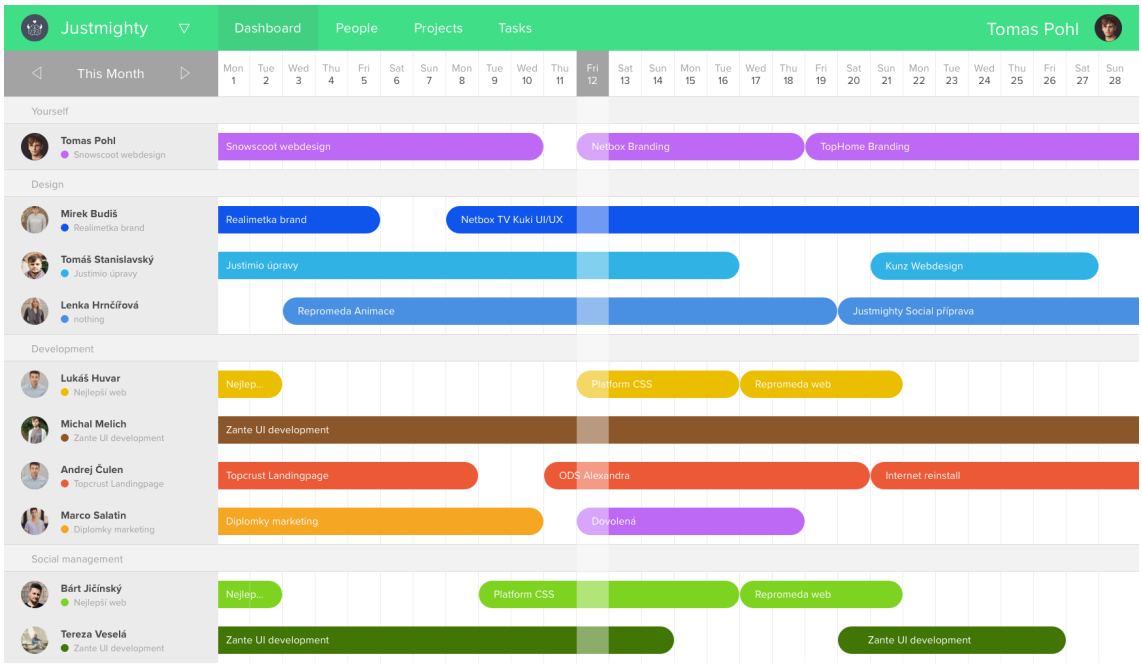
4.6.4 Aktér administrátor

Administrátor dědí všechna práva od vlastníka organizace a má přístupnou sekci administrace, kde začíná jeho model. Tato sekce je přístupná pouze administrátorovi, protože dovoluje pracovat se všemi daty systému. Administrátor může vytvářet nové organizace, a také upravovat stávající nebo je smazat. Pro organizace může vytvářet export pro přehlednější zobrazení. Mezi další případy užití patří zobrazení plateb v systému a jejich možná správa. Administrátor má přístup k editaci všech uživatelů v systému. Může je blokovat, smazat a vytvářet nové.

Obrázek 4.4: Případ použití administrátora (Zdroj: vlastní)

4.7 Návrh uživatelského rozhraní

Uživatelské rozhraní bude složeno z jednoduchých prvků bez rušivých elementů jako jsou stíny, barevné přechody apod. Cílem je uživatelům usnadnit orientaci v aplikaci a aby aplikace odpovídala moderním trendům v oblasti designu uživatelských rozhraní.



Obrázek 4.5: Dashboard aplikace (Zdroj: Tomáš Pohl)

Navigace se bude odehrávat v horní části aplikace, kde bude možné přepínat mezi týmy, navštívit sekci Dashboard, People, Projects, Tasks, vyhledávat napříč aplikací nebo vstoupit do sekce My Profile. Více o jednotlivých sekcích níže.

4.7.1 Organizace

Každý uživatel musí být součástí nějaké organizace a jeden uživatel může být součástí více organizací. Mezi těmito organizacemi může uživatel libovolně přepínat a snadno při práci přecházet od jedné skupiny ke druhé. Tato obrazovka slouží převážně pro zaměstnance s polovičním úvazkem nebo uživateli, kteří pracují pro více společností najednou.

4.7.2 People

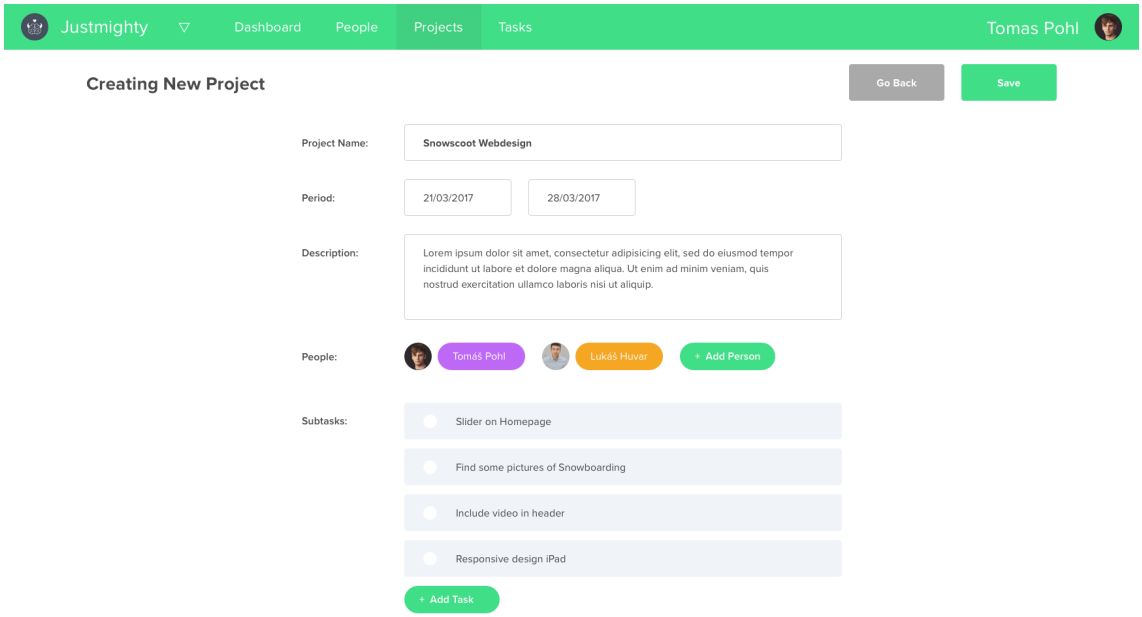
Uživatelé budou základním elementem v aplikaci. Každý projekt musí být přiřazen konkrétní osobě a každá osoba má svá přihlašovací práva. V aplikaci je možné osoby přidávat či odebírat a přiřazovat je k jednotlivým týmům. Každý projekt je pak definován příslušností k alespoň jedné osobě, která bude danou činnost vykonávat.

Každá osoba bude definována jménem, příslušností k týmu (např. design, development apod) a barvou, kterou se vizuálně odlišují od ostatních, celý harmonogram se tím zpřehledňuje. Jednotlivé osoby budou mít různá uživatelská práva. Někteří mohou úkoly vytvářet a editovat v rámci celého týmu, někteří mohou upravovat pouze své úkoly.

V aplikaci bude přístupný seznam všech osob v týmu, kde lze jednotlivé uživatele rychle editovat nebo sledovat, kolik mají právě přidělených projektů či úkolů a kolik jich již dokončili. Vlastník organizace v tomto rozhraní může snadno přidělovat práva konkrétním uživatelům.

4.7.3 Project

Projekt je prvek, který je definován názvem, popisem, datem začátku a datem konce, alespoň jednou osobou, která je za něj zodpovědná a součástí projektu mohou být drobné podúkoly. Projekt je jiné pojmenování pro dlouhodobý úkol. Tyto projekty se zobrazí na časové ose a tvoří harmonogram.



Obrázek 4.6: Vytváření projektů (Zdroj: Tomáš Pohl)

V záložce Projekty bude přístupný seznam všech projektů, kde je bude možné snadno editovat, mazat či mezi nimi filtrovat. Bude zobrazeno, kolik podúkolů projekt obsahuje, v jaké stavu se momentálně nachází (dokončen, rozpracován, čeká), a kdo je za projekt zodpovědný. Lze provádět hromadné změny jako například odstranění vybraných položek.

4.7.4 Tasks

Tasks neboli podúkoly budou podsložkou projektů. Každý projekt může a nemusí obsahovat sadu drobnějších úkolů k danému projektu. Například v případě designu webových

stránek může být podúkolem design homepage, design stránky kontaktů apod. Uživatel tak může snadno evidovat práci na daném projektu.

Úkoly lze vytvořit již během založení projektu nebo kdykoliv během práce s aplikací, a to jak v záložce Task nebo v Dashboardu na časové ose.

4.7.5 Dashboard

. Dasboard bude srdcem celé aplikace. Je to místo, kde lze sledovat celý harmonogram. Každá osoba má jeden řádek, ve kterém má na časové osnově rozložené všechny projekty. Vidí také projekty všech svých kolegů, jak na sebe činnosti navazují, a ve kterém dni na časové ose se právě nachází. Může si projekty otevřít, přečíst si o nich více informací, nebo si k nim přidat drobné podúkoly, tyto podúkoly označit za splněné.

Řádek evidující projekty přihlášeného uživatele bude umístěn v samostatné sekci v horní části aplikace, aby byl vždy dobře viditelný. Dále budou jednotlivé osoby rozřazeny do kategorií odpovídajících týmům uvnitř organizace, tedy například designéři apod.

Také bude možné měnit rozsahy zobrazených dnů pomocí filtru v levé horní části a to buď na přednastavené rozsahy tento týden, minulý týden, příští týden, tento měsíc, minulý měsíc, příští měsíc a nebo přímo prostřednictvím kalendáře zadat požadovaný rozsah až do výše jednoho celého roku.

4.7.6 My profile

V této sekci si bude moci uživatel měnit své základní údaje a upravit své nastavení. Vlastníci organizace zde budou moci spravovat svůj účet z finanční stránky.

4.7.7 Administration

Tato sekce bude sloužit k administraci celého systému. Přístup do následující sekce bude pouze pro administrátory, a proto není zobrazena v navigaci aplikace. Administrátor bude moci vybrat ze 3 základních sekcí.

První sekce bude věnována administraci organizací. V přehledném listu lze pomocí vyhledávání a filtrace vybírat konkrétní organizaci. U každé z organizací je uveden počet členů, projektů, úkolů apod. V této části administrace bude možné vytvářet statistické přehledy o organizacích a jejich údajích.

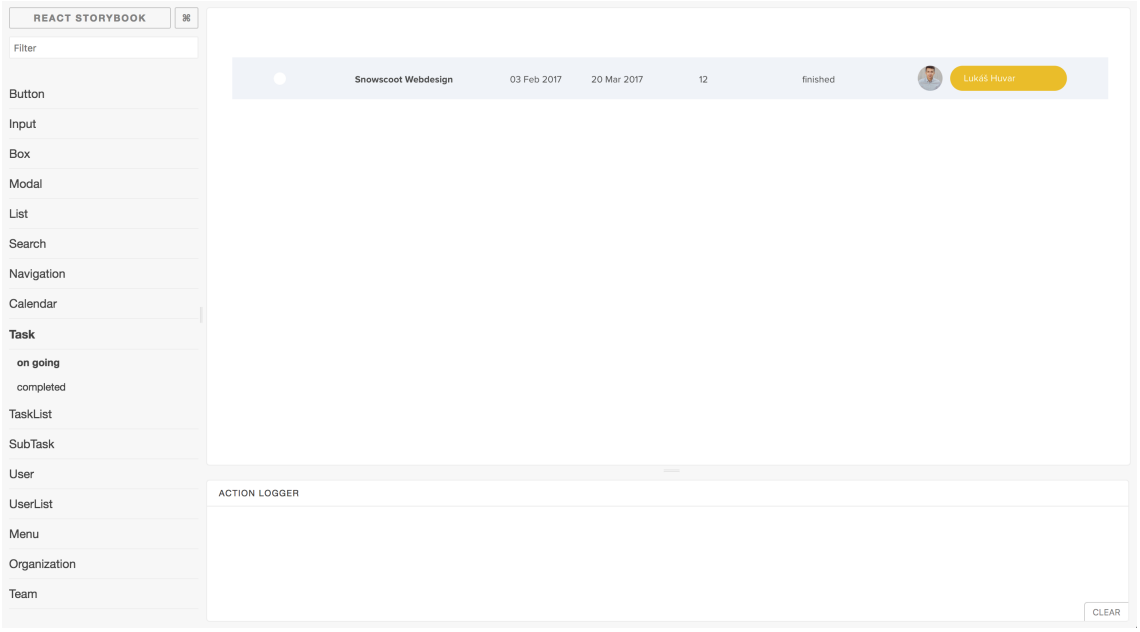
Druhá část administrace nabízí kompletní data o přehledu plateb. V listu je indikováno, zda platba již proběhla, zda byla úspěšná, eventuálně kdy proběhne další platba.

Systém rovněž umožňuje statistické přehledy včetně grafických výstupů o platbách. Uživatel si může vybrat časové období a další filtry, které mu umožní vyexportovat vždy ta data, která ho právě zajímají.

Třetí část zobrazuje všechny uživatele aplikace. Veškerá jejich data (žádná z nich nejsou citlivá), příslušnost k organizaci aj. Administrace umožňuje uživatele smazat, blokovat nebo měnit jeho práva.

4.8 Component driven development

Celá aplikace je vyvíjena ve stylu Component driven developmentu, proto je celé uživatelské rozhraní rozdělené na malé komponenty. Tyto komponenty poté tvoří složitější obrazovky. Každý komponent má předem definované stavy, kterých může nabývat. V projektu používáme **React Story Book**[24], kterým tvoříme knihovnu našich komponent.



Obrázek 4.7: Komponenta Task ve Story Booku (Zdroj: vlastní)

Pro každý komponent je vytvořen příběh, který popisuje jeho chování. Tento způsob vývoje nám umožňuje tvořit komplexnější uživatelské rozhraní v kratším časovém měřítku. Další důvod využití tohoto stylu vývoje je kratší potřeba testování aplikace, jelikož otestujeme funkčnost jednotlivých komponent před celkovým dokončením aplikace.

4.9 GraphQl schéma

Podle funkčnosti aplikace jsme si vymodelovali GraphQl schéma, které nám zaručí veškerou funkcionalitu aplikace. Schéma si můžeme prohlédnout na obrázku 4.8 a podrobnější popis typů máme níže.

- **File** - slouží k uchování informací o všech souborech, které jsou v systému. Jedná se převážně o fotky uživatelů a organizací.
- **User** - tento typ obsahuje veškeré informace o uživateli. Obsahuje identifikaci uživatele a jeho přihlašovací údaje.
- **Organization** - slouží k uložení informací o organizaci. Jedná se o název organizace, fotografii a její popis. Je napojena na většinu typů ve schématu, jelikož potřebuje uchovávat informace o členech, platbách, aktivním plánu atd.
- **Membership** - funguje jako vztah mezi **Organization** a **User**. Určuje jaký uživatel má jakou roli v organizaci a také slouží pro jedinečné označení uživatele v organizaci.
- **Team** - slouží k uchování týmů v organizaci, a rozdělení uživatelů do určitých týmů.
- **Project** - obsahuje veškeré informace úkole. Kdo za úkol zodpovídá, termín dokončení a začátku.
- **Task** - uchovává informace o podúkolech. Kdo na úkolu pracuje, do kdy se má dokončit a jeho popis.
- **Pause** - slouží k uchování informací o přerušení úkolu.
- **Plan** - obsahuje typy předplatných a informace s nimi spojené. Každá **Organization** musí mít plán, který určuje kolik může mít aktivních členů a jakou formou platí předplatné.
- **Billing** - slouží k uchování informací o všech proběhlých platbách v systému.

4.10 Graphcool

Pro vytvoření GraphQl API využijeme službu Graphcool, kterou jsme zmínili v návrhu architektury 4.1. Tato služba vytvoří pomocí GraphQl schémat optimalizovanou databázi.

Obrázek 4.8: Návrh graphql schématu (Zdroj: vlastní)

Tato služba nám poskytuje informace o využití našeho API a monitoruje aktuální zátěž celého systému. Pro bezpečnost celé aplikace jsou data automaticky zálohovaná.

4.10.1 Serverless function

Graphcool nám umožňuje vykonávat akce bez nutnosti mít vlastní server. Tato funkcionalita se využívá pro účtování měsíčních plateb za používání aplikace. Každý první den v měsíci se vyvolá funkce, která všem organizacím vyúčtuje aktuální měsíc, pokud mají měsíční předplatné. Pro roční předplatné se tato akce provádí pouze jednou v roce a to v termínu, kdy jsem ji po prvé zaplatili. Pro placení se využívá platební platforma **Stripe**[25]. Stripe se stará o zabezpečené platby a zabraňuje možnému zneužití platební karty. Systém je pomocí Stripe ochráněn proti zneužití ukradených karet a možným finančním únikům.

4.11 Ekonomické zhodnocení

Celý vývoj aplikace byl iniciován nedostupností vhodné aplikace pro správu času. Stupňující se neefektivnosti všech procesů ve firmě vedly k nižší konkurenceschopnosti společnosti Justmighty. To vedlo k nutnosti implementovat aplikaci na organizaci lidských zdrojů. Ročně se vynakládají zbytečné finanční prostředky na aplikace od konkurence, které přitom nesplňovaly požadavky této společnosti a nedokázaly pokrýt její potřeby (byly příliš složité a neohebné).

4.11.1 Náklady na tvorbu aplikace

Náklady na tvorbu aplikace jsou zejména hodiny lidské práce v rámci společnosti Justmighty. Cenová kalkulace vychází z hodinové sazby Justmighty a to 1000 Kč/ hodinu odborných zaměstnanců a z odhadovaného času na realizaci celého projektu. Celý projekt se dá rozdělit na několik fází viz tabulka níže.

Činnost	Odhadovaný počet hodin	Cena
Průzkum trhu	80	80 000 Kč
UI prototyp a testování	100	100 000 Kč
UI design	70	70 000 Kč
Kódování Front-end	100	100 000 Kč
Implementace	50	50 000 Kč
Testování	80	80 000 Kč
Marketing		500 000 Kč
Provoz na 1 rok	200	200 000 Kč
Cena celkem		1 290 000 Kč

Tabulka 4.1: Tabulka odhadovaných nákladů (Zdroj: vlastní)

4.11.2 Zpeněžení aplikace

Projekt byl primárně vytvořen pro potřeby studia Justmighty, ale aplikace bude vyvíjena tak, aby bylo možné ji implementovat do dalších společností. Není proto vázaná na konkrétní firmu nebo oborové odvětví a lze ho nabídnout k volnému prodeji.

Zvolili jsme model zpoplatnění na základě měsíčního paušálu a počtu uživatelů. Veškerá funkcionalita je vždy dostupná všem uživatelům. Zařadili jsme možnost zkušební verze, tedy časově omezeného používání aplikace omezeným počtem uživatelů, kteří mají možnost si aplikaci vyzkoušet.

Produkt jsme rozdělili do 4 balíčků podle počtu uživatelů, aby si mohli zákazníci vybrat model, který nejvíce vyhovuje jejich podnikání. Se zvyšujícím se počtem uživatelů klesá cena za jednoho uživatele. Cenová hladina vyplývá z analýzy trhu konkurence.

- **Free** - do 3 uživatelů je používání aplikace zcela zdarma. Je ovšem zapotřebí zadat platební údaje a při překročení hranice 3 osob se balíček transformuje na balíček Small a dochází k platbě.
- **Small** - do 10 uživatelů je cena za jednoho uživatele stanovena na 9 $ za 1 měsíc používání při platbě na 1 rok dopředu.
- **Medium** - do 40 uživatelů je cena za jednoho uživatele stanovena na 7 $ za 1 měsíc používání při platbě na 1 rok dopředu.
- **Large** - 40 a více uživatelů - cena je za jednoho uživatele stanovena na 5 $ za 1 měsíc používání při platbě na 1 rok dopředu.

4.11.3 Návratnost projektu

Celkové odhadované náklady na realizaci projektu včetně nákladů na marketing jsou na 1 290 000 Kč. Uvažujeme průměrnou měsíční tržbu za jednoho platícího uživatele 7 $ s DPH, tedy 5,8 $ bez DPH. Což při kurzu 26 Kč/USD činí 151 Kč. To odpovídá roční tržbě na jednoho platícího uživatele rovné 1 812 Kč bez DPH.

Pokud náklady projektu včetně marketingu a náklad na jeden rok provozu vydělíme průměrnou roční tržbou za jednoho platícího uživatele, získáme výsledek odpovídající 712 aktivním uživatelům. To je zlomový počet platících uživatelů, při kterém se začne tvořit zisk. Pokud se podaří získat alespoň 713 platících uživatelů do jednoho roku po spuštění projektu, jsou umořeny všechny odhadované náklady a projekt se dostává do zisku.

Při nákladech 500 000 Kč na marketing, který by byl investován zejména do PPC reklam je očekávaný náklad na konverzi jednoho uživatele 700 Kč. Pokud by byla kampaň úspěšná a efektivní, dá se očekávat náklad na jednu konverzi cca 400 Kč. Takto úspěšná kampaň by přinesla 1 250 uživatelů. To odpovídá odhadovaným tržbám ve výši 2 265 000 Kč bez DPH, a tedy potenciálnímu zisku po jednom roce provozu od spuštění projektu 975 000 Kč. Jedná se optimisticky reálnou variantu.

4.12 Přínos aplikace

Aplikace Justimio naopak nabízí jednoduché a úzce soustředěné řešení na problematiku organizace projektů v čase a jejich zařazení na časové ose. Managementu firmy tak nabízí okamžitý pohled na vytíženost kapacity osob ve firmě a lze z ní snadno vyčíst termíny dokončení jednotlivých projektů, a jak na sebe různé činnosti navazují.

Justimio pomáhá eliminovat chyby v řízení projektů tak, aby nedocházelo k překrývání činností jedné osoby. Z grafického rozhraní aplikace lze jednoduše vypozorovat, že v daném termínu pracuje osoba již na jiném projektu a není tedy možné, aby pracovala na dvou úkolech zároveň. Důsledkem je přesnější a reálnější odhadování finálních termínů dokončení projektů a eliminace stresu z časového tlaku, který je často managementem vyvíjen.

Produkt také napomáhá k efektivnímu a plynulému předávání práce mezi osobami, které přebírají rozpracované projekty od svých kolegů. Na časové ose lze vypozorovat, jak na sebe dané projekty navazují, kdy končí práce jednoho zaměstnance a začíná práce druhého. Dále nám aplikace pomůže vyvarovat se dlouhým prodlevám mezi jednotlivými fázemi projektu.

Aplikace velmi usnadní práci zejména projektovému managementu, který může jednak snadno sledovat průběh prací, ale také aktivně zadávat úkoly jednotlivým osobám, čímž se usnadňuje komunikace se středním managementem a zároveň střední management získává lepší kontrolu nad harmonogramem a maximálním využitím lidských kapacit.

Lze také zpětně vyhodnocovat, jak byla naplněna očekávání o včasném dokončení projektu. Jinými slovy, zda byl odhad zanesený do harmonogramu přesný, podhodnocený či nadhodnocený. Výhodou těchto dat je možnost vyvození důsledků pro další podobný projekt. Odhady termínu se tak v čase budou zpřesňovat.

Závěr

Cílem diplomové práce bylo vytvoření nového informačního systému, který by měl umožnit efektivnější organizaci lidských zdrojů v malých a středních podnicích. Tento informační systém by měl primárně sloužit pro společnost Justmighty. Sekundární využití aplikace je její následný prodej vybrané cílové skupině.

Před samotnou implementací systému byla provedena analýza současného stavu. Zároveň bylo nutné analyzovat konkurenční prostředí a trh, ve kterém se nachází společnost Justmighty, a na jaký trh se řadí aplikace Justimio. Pomocí interních a externích analýz jsme došli k závěru, že společnost Justmighty má dostatečné předpoklady pro vytvoření a implementaci aplikace Justimio. Zároveň má potenciál pro splnění sekundárního cíle a tím je vstup na trh s aplikacemi pro zefektivnění organizace času.

Pomocí informací získaných v analýze a požadavků od vedení společnosti jsme navrhnuli architekturu systému a jeho funkcionalitu. Další část práce byla věnována implementaci jednotlivých části systému a jeho celkové funkčnosti. V návrhu řešení byla také popsána nutná investice do vývoje tohoto systému a jeho možná návratnost. Aplikace Justimio má potenciál pro zisk finančních prostředků z měsíčních poplatků, a to při využití správné marketingové strategie a výběru cílové skupiny uživatelů.

Literatura

[1] MOLNÁR, Zdeněk. *Efektivnost informačních systémů*. Praha: Grada, 2000. Systémová integrace. ISBN 80-716-9410-X.

[2] SOMMERVILLE, Ian. *Softwarové inženýrství*. Brno: Computer Press, 2013. ISBN 9788025138267.

[3] DAWSON, Alexander. *Výjimečný webdesign: jak tvořit osobité, přitažlivé, použitelné weby*. Brno: Computer Press, 2012. ISBN 9788025137192.

[4] CASTRO, Elizabeth a Bruce HYSLOP. *HTML5 a CSS3: názorný průvodce tvorbou WWW stránek*. Brno: Computer Press, 2012. ISBN 978-80-251-3733-8.

[5] GÁLA, Libor, Jan POUR a Zuzana ŠEDIVÁ. *Podniková informatika: počítačové aplikace v podnikové a mezipodnikové praxi*. 3., aktualizované vydání. Praha: Grada Publishing, 2015. Management v informační společnosti. ISBN 9788024754574.

[6] *Universal JavaScript* [online]. San Francisco: Michael Jackson, 2015 [cit. 2017-05-17]. Dostupné z: https://medium.com/@mjackson/universal-javascript-4761051b7ae9

[7] *GraphQl* [online]. San Francisco: Facebook, 2017 [cit. 2017-05-17]. Dostupné z: http://graphql.org/

[8] *React* [online]. San Francisco: Facebook, 2017 [cit. 2017-05-17]. Dostupné z: https://facebook.github.io/react/

[9] Reconciliation. *React* [online]. [cit. 2017-04-26]. Dostupné z: https://facebook.github.io/react/docs/reconciliation.html

[10] *Component-Driven Development* [online]. [cit. 2017-04-26]. Dostupné z: https://blog.hichroma.com/component-driven-development-ce1109d56c8e

[11] Dydowicz, Petr. *Zabezpečení IS pomocí integrace SW a HW prvků* Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2014.

[12] SKLENÁK, Vilém. *Data, informace, znalosti a Internet*. Praha: C.H. Beck, 2001. C.H. Beck pro praxi. ISBN 8071794090.

[13] JAKUBÍKOVÁ, Dagmar. *Strategický marketing*. Praha: Grada, 2008. Expert (Grada). ISBN 978-80-247-2690-8.

[14] HANZELKOVÁ, Alena. *Strategický marketing: teorie pro praxi*. Praha: C.H. Beck, 2009. C.H. Beck pro praxi. ISBN 978-80-7400-120-8.

[15] PORTER, Michael E. *Konkurenční strategie: Metody pro analýzu odvětví a konkurentů*. Praha: Victoria Publishing, 1994. ISBN 8085605112.

[16] SEDLÁČKOVÁ, Helena. *Strategická analýza*. 2. přeprac. a dopl. vyd. Praha: C. H. Beck, 2006. ISBN 9788071793670.

[17] Management Mania. *McKinsey 7s* [online]. [cit 2017-01-13]. Dostupné z : https://managementmania.com/cs/mckinsey-7s

[18] About Trello. *Trello* [online]. [cit. 2017-04-27]. Dostupné z: https://trello.com/about

[19] *We unlock the 25th hour* [online]. Oslo: Timely, 2017 [cit. 2017-04-27]. Dostupné z: https://timelyapp.com/about

[20] *Redux* [online]. San Francisco: Github, 2017 [cit. 2017-05-17]. Dostupné z: https://github.com/reactjs/redux

[21] *React Apollo* [online]. San Francisco: Appolo, 2017 [cit. 2017-05-17]. Dostupné z: http://dev.apollodata.com/react/

[22] *Graphcool* [online]. Berlin: Graphcool, 2017 [cit. 2017-05-17]. Dostupné z: https://www.graph.cool/

[23] *Let's Encrypt* [online]. San Francisco: Let's Encrypt, 2017 [cit. 2017-05-17]. Dostupné z: https://letsencrypt.org/

[24] *StoryBook* [online]. San Francisco: Github, 2017 [cit. 2017-05-17]. Dostupné z: https://github.com/storybooks/storybook

[25] *Stripe* [online]. San Franciso: Stripe, 2017 [cit. 2017-05-17]. Dostupné z: https://stripe.com/

Seznam obrázků

Seznam tabulek