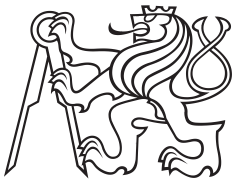
Bakalářská práce



České
vysoké
učení technické
v Praze

FEL

Katedra počítačů

Administrační rozhraní aplikace Pingl

Filip Štěrba

Vedoucí práce: Ing. Filip Molčík
Květen 2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Štěrba** Jméno: **Filip** Osobní číslo: **466066**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Administrační rozhraní aplikace Pingl

Název bakalářské práce anglicky:

Admin interface for Pingl application

Pokyny pro vypracování:

1) Proveďte rešerši současného řešení a současných procesů ve vybrané společnosti.
2) Vypracujte analýzu uživatelských požadavků.
3) Na základě analýzy nadefinujte očekávanou funkcionalitu administračního rozhraní, které umožní správu současných a nových klientů a zobrazení statistiky nad existujícími daty.
4) Srovnejte technologie pro tvorbu single page aplikací a vyberte takovou, která bude nejvhodnější pro navrhovanou aplikaci.
5) Vytvořte návrh uživatelského rozhraní.
6) Naimplementujte administraci jako single-page aplikaci v předem definováném rozsahu za pomocí vámi vybraných technologií.
7) Ověřte funkčnost a uživatelské rozhraní formou uživatelských testů.

Seznam doporučené literatury:

[1] R. Wieruch, The Road to learn React: Your journey to master plain yet pragmatic React.js, 2018
[2] M. Mikowski, J. Powell, Single Page Web Applications: JavaScript end-to-end, Manning Publications, 2013
[3] J. J. Garrett, The Elements of User Experience: User-Centered Design for the Web and Beyond, New Riders, 2010

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Filip Molčík, KOALA42

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **14.02.2020** Termín odevzdání bakalářské práce: **22.05.2020**

Platnost zadání bakalářské práce: **30.09.2021**

Ing. Filip Molčík podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. | Datum převzetí zadání

Podpis studenta

Poděkování

V první řadě bych chtěl poděkovat mému vedoucímu Ing. Filipu Molčíkovi za pomoci při vypracování této bakalářské práce. Dále bych chtěl poděkovat kolegům z Pingl s.r.o., Vojtěchu Rychnovskému a Alexanderovi Mensákovi, za jejich součinnost při sběru požadavků a testování. V neposlední řadě děkuji rodině a všem, kteří mě ve studiu podporovali.

Prohlášení

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

V Praze, 22. května 2020

Abstrakt

Cílem této práce je nejprve zanalyzovat potřebu uživatelů administrace a důvody pro vznik projektu. Následně od relevantních uživatelů sesbírat požadavky a určit rozsah práce. Důkladná analýza současné situace a požadavků uživatelů bude následně využita při porovnávání technologií a výběru té nejvhodnější. Nelze totiž bez znalosti celkového rozsahu projektu a požadavků jednoznačně vybrat vhodnou technologii pro implementaci. Při výběru technologie budou analyzovány a porovnány dva přístupy pro tvorbu webových aplikací a následně zvolen nejvhodnější postup. Na základě sesbíraných informací bude učiněn návrh podoby uživatelského rozhraní a provedena implementace systému. Po dokončení implementace proběhne testování, které ověří výsledek implementace.

Klíčová slova: administrace, single-page aplikace, web, návrh, React

Vedoucí práce: Ing. Filip Molčík

Abstract

The goal of this semestral thesis is to analyze the needs of administration users and reasons for project development. Next, collect requirements from stakeholders and define the scope of the project. Results of the current situation analysis and user requirements will be used during technologies comparison and selection of the most relevant one. Without knowledge of project scope and all requirements, it would not be possible to choose the best technology for implementation. Two principles of web application development will be compared and the best for our case will be selected. Based on requirements will be designed user interface and created a simple prototype of a single-page application with selected technologies.

Keywords: administration, single-page application, web, design, React

Title translation: Admin interface for Pingl application

Obsah

Obrázky

Tabulky

Kapitola 1
Úvod

Pingl je webová platforma, která umožňuje zákazníkům gastro podniků objednávat a platit bez čekání pomocí mobilního telefonu. Jejím hlavním cílem je zlepšit zážitek z jídla zákazníkům. Umožňuje tak okamžitou objednávku bez čekání na číšníka, na menu nebo na zaplacení.

Platformu začíná využívat více a více podniků, které musí firma Pingl s.r.o. spravovat. Na základě rostoucí poptávky, která zapříčinila nemožnost manuálního spravování partnerů (podniků) využívající platformu Pingl, vznikla potřeba vytvořit administraci. Administraci, přes kterou bude mít management firmy přehled nad všemi klienty a jejich daty. Bude tak zajištěna centralizovaná správa celé platformy, která usnadní proces přidávání nového klienta, umožní datovou analýzu nad všemi podniky v platformě a extrémně zjednoduší některé současné firemní procesy.

Aby výsledná práce byla skutečně užitečná pro konkrétní použití, bude potřeba vytvořit analýzu, která definuje potřeby a přání samotných koncových uživatelů platformy - spolumajitelů společnosti Pingl s.r.o. V práci se tak nachází analýza požadavků od zainteresovaných stran a návrh uživatelského rozhraní.

Zároveň existuje několik postupů, jak kýženého výsledku dosáhnout. Je tedy potřeba vypracovat analýzu, která rozebere moderní přístupy k tvorbě administrací a vybrat ten nejlepší pro tento konkrétní případ a využít jej k implementaci. Výběr přístupu a technologie bude také zásadní pro budoucí údržbu a rozvoj administrace.

Kapitola 2

Rešerše současného stavu

V úvodu samotné práce je potřeba zmínka o platformě Pingl, pro jejíž potřeby se bude administrace navrhovat a následně vyvíjet. Tato kapitola stručně shrnuje dosavadní podobu služby, jak po byznysové, tak technologické stránce. Kapitola tak odpovídá na otázku "Proč?", která by před zahájením jakéhokoliv projektu nebo snažení měla být vyjasněna a definuje procesy, které chce firma pomocí administračního rozhraní řešit.

2.1 Popis platformy Pingl

Jak bylo zmíněno v úvodu, platforma Pingl umožňuje zákazníkům gastro podniků objednávat a platit bez čekání. Na této platformě společně s kolegy pracuji přes rok a úspěšně ji využívá několik podniku v Praze. Pro účely této práce je důležité alespoň částečně představit její části, protože poznání platformy Pingl bude stěžejní při získávání požadavků, návrhu funkcionalit i samotné implementaci.

Platforma se sestává ze tří částí:

2.1.1 Aplikace pro zákazníky

Webová aplikace pro koncové zákazníky, kteří si mohou objednávat v restauracích, kavárnách a bistrech, která jsou součástí platformy Pingl. Aplikace umožňuje objednávku a platbu zákazníkům přímo od stolu, nebo na určitý čas k vyzvednutí s sebou. Při objednávání od stolu je na každém stole v podniku umístěný QR kód, který zákazník naskenuje v aplikaci, je tím okamžitě propojen s podnikem, vidí jeho menu a může začít objednávat.

2.1.2 Rozhraní pro personál

Aplikace pro obsluhu podniku, ve které personál vidí příchozí objednávky a může je následně odbavovat. Tuto část aplikace má obsluha k dispozici v tabletu, který je umístěn na baru nebo v místě, kde může obsluha tablet kontrolovat. Rozhraní také umožňuje připojit bonovací tiskárnu, pro případ, že je obsluha zvyklá s bony pracovat a znesnadnily by se tím tak zažité procesy v kuchyni.

2.1.3 Administrace pro provozní

Webová administrace, která provoznímu nebo majiteli podniku dává přehled o objednávkách přes Pingla v daném podniku. Majitelům nebo provozním podniku je zřízen unikátní účet s heslem, přes který se do administrace přihlásí. Následně mohou naplnit a upravovat menu restaurace, nastavovat slevy a různé akce, mají detailní přehled o všech objednávkách. Lze také nastavit otevírací hodiny, podle kterých se možnost objednávek automaticky vypíná, nastavovat časové omezení jednotlivých položek a konfigurovat celou platformu v rámci podniku.

2.2 Technologie

Všechny tři části platformy Pingl popsané výše jsou vyvinuty jako webové aplikace. Administrace, aplikace pro zákazníky a rozhraní pro personál je postaveno na javascriptovém frameworku React.

Aplikace pro zákazníky má uživatelské rozhraní navržené na míru a nevyužívá žádné externí knihovny. Rozhraní pro personál využívá části knihovny pro tvorbu UI - Material Components, například pro výběr času, nebo tlačítka pro kontextové menu. Administrace pro provozní podniku má uživatelské rozhraní postaveno za pomocí knihovny Ant Design.

Primární backend platformy je naprogramován v jazyku Kotlin. Celá platforma disponuje také druhým backendem postavený na Node. Ten se využívá například pro zpracovávání transakcí. Všechna data jsou ukládána do NoSQL databáze MongoDB.

Komunikace mezi backendem a frontendem probíhá přes GraphQL API.

2.3 Procesy

V tento moment jsou všechny aktivity, které mají být centralizovány do jedné administrace prováděny na několika místech.

2.3.1 Přidání nového podniku

Funkcionalita, která je již na backendu implementována, ale chybí pro ní uživatelské rozhraní, přes které by se požadavek odeslal. Při vytváření nového podniku do platformy je potřeba onen požadavek, který entitu restaurace vytvoří, volat z nástroje Insomnia, nebo jiného nástroje umožňujícího provádět REST / GraphQL dotazy. Navíc, onen dotaz neobsahuje vše potřebné, takže se některé změny musí provádět přímo v databázi správcem serveru. Tento proces je jednak neefektivní, není uživatelsky přívětivý a představuje jisté bezpečnostní riziko.

2.3.2 Vytváření účtů

Dalším příkladem procesu, který má administrační rozhraní značně usnadnit je vytváření uživatelských účtů pro provozní a personál nového podniku. Ty se musí opět vytvářet přes nástroj Insomnia a následně se musí uživatelům ručně nastavit práva k dané restauraci přes MongoDB Compass, nástroj umožňující přímý vstup do MongoDB. Opět se jedná o bezpečnostní riziko a proces, který může provádět jen člověk znalý zmiňovaného nástroje. Cílem však je proces zjednodušit a umožnit i dalším pověřeným osobám účty vytvářet.

2.3.3 Analytika

Pro zobrazení přehledů nad daty v platformě je zde opět cesta, která již v současné době při současné velikosti platformy nevyhovuje a znesnadňuje tak administrátorům platformy Pingl přístup k důležitým datům a statistikám. Jediná možnost zobrazení dat je přihlášení se do administrace daného podniku a zobrazit si data pouze o něm, která jsou přístupná i majitelům podniků. Tato data jsou jednak neúplná a postrádají informace, které jsou klientům skryty. Zároveň neexistuje možnost, jak si zobrazit data z více podniků najednou. Proto je tento proces nedokonalý a znemožňuje dělat rychlá rozhodnutí na základě dat.

Kapitola 3
Analýza

Kapitola se zaměřuje na analýzu vstupních požadavků a procesů, které se následně promítnou do funkcí administrace. Budou zde určeni uživatelé platformy, proběhne sběr požadavků od zainteresovaných osob, jejich prioritizace. Na základě přiřazení priorit budou vybrány požadavky, které budou součástí výsledné implementace a v kapitole se tak definuje rozsah projektu.

3.1 Zainteresované strany

Zainteresované strany nebo osoby jsou vedoucími orgány s rozhodovací pravomocí, které jsou odpovědné za části organizace, která bude ovlivněna konečným strategickým směřováním produktu.[1]

Před zahájením sběru požadavků bude důležité si definovat všechny tyto osoby a strany, které mohou mít na jejich definici vliv. Obecně se jedná například o koncové zákazníky, vedoucího projektu, management firmy, která projekt zajišťuje a další [2]. Identifikace těchto osob a institucí je důležitá, protože úspěch projektu na nich přímo závisí. Pokud nejsou s výsledkem spokojeni, projekt není úspěšný, prodlužuje se, nebo je úplně zrušen. Je tedy důležité znát požadavky všech zúčastněných stran.

V našem konkrétním případě se jedná o tři osoby, spolumajitele firmy, kteří budou mít do administrace plný přístup. V současné chvíli žádní další uživatelé, nebo osoby, které by měly vliv na definici požadavků nejsou.

Nelze však opomenout skutečnost, že v případě růstu firmy bude třeba administraci zpřístupnit i dalším osobám, které nebudou mít plná práva. Systém se tedy musí navrhovat tak, aby bylo možné jeho snadné rozšíření v budoucnu.

Seznam zainteresovaných osob

Seznam zainteresovaných osob pomáhá stanovit jednak jejich počet, ale i jejich důležitost a očekávání od projektu. V případě, kdy je zainteresovaných osob větší počet, můžeme se soustředit pouze na ty, které mají na projekt největší vliv, případně vybrat zástupce za skupinu, například uživatelé.[3]

Jméno	Role	Vliv
Alexander	Spolumajitel, uživatel	Velký
Vojtěch	Spolumajitel, uživatel	Velký
Filip	Spolumajitel, uživatel	Velký

Protože se v tomto případě jedná o celkem tři osoby, které budou systém využívat a mají k němu také stejná práva, bude sběr požadavků obsahovat odpovědi od těchto třech osob. Osoby, které prozatím budou jedinými uživateli systému jsou také jeho zadavateli. Soustředění se na tuto skupinu zaručí, že žádná důležitá zainteresovaná osoba nebude opomenuta a bude předejito vzniku dodatečných požadavků v průběhu implementace.

3.2 Analýza požadavků

Analýza požadavků je proces definování očekávání zainteresovaných osob od aplikace, která má být vyvinuta nebo modifikována [4].

Existuje několik technik, kterými lze požadavky získávat, kde každá z technik se hodí pro různé typy projektů a vyžaduje některé prerekvizity. Protože je projekt v začátcích a neexistuje žádná projektová analýza, prototyp, ani další podklady, můžeme vyřadit metody, které zmíněné podklady vyžadují. Díky předem provedené analýze zainteresovaných osob také vidíme, že jejich počet není velký. Nedává tedy smysl dotazník, který se využívá zejména při větším množství zúčastněných stran, od kterých potřebujeme požadavky sesbírat [5].

3.2.1 Brainstorming, Skupinové diskuze

V naší počáteční fázi chceme jednak sesbírat nové nápady a požadavky na nově vznikající systém, tak si ověřit současná řešení a procesy, které se pro konkrétní činnosti aplikují. Proto se nejvíce nabízí následující dvě metody.

První metodou je brainstorming. Během brainstormingu má každý z účastníků čas mluvit, není přerušován ani napomínán, pokud nemluví přímo k věci. Cílem této metody je totiž získat i netradiční pohled na řešený problém [6]. Metoda byla zvolena, protože je projekt v počátcích a bylo potřeba získat co nejvíce podnětů a požadavků pro nový systém.

Druhá metoda se jmenuje "Focus groups", v překladu Skupinové diskuze [7]. Během skupinových diskuzí je naopak zapotřebí, aby účastníci měli hlubší znalost problematiky a spíše validovali již existující nápady. Protože byla naše skupina zainteresovaných osob zároveň uživateli systému a zadavateli řešení, byl po brainstromingu vyhrazen prostor nad zamyšlením se nad současným řešením, propojení s existující administrací pro provozní podniků a validací současných řešení.

Při sbírání požadavků bylo využito kombinace obou metod [8]. Jak bylo popsáno, nejprve následoval brainstorming pro získání všech požadavků a následně byly požadavky porovnávány s existujícími procesy.

Všechny požadavky byly zaznamenány, nejprve v surové podobě a následně rozřazeny do kategorií a sekcí, aby mohla proběhnout jejich prioritizace.

3.2.2 Prioritizace požadavků

V předchozí kapitole byly za pomocí dvou zmiňovaných metod nasbírány požadavky na výsledný systém od zainteresovaných osob. Požadavky byly následně převedeny do formátu, ve kterém jsou čitelné, nejsou duplicitní a lze nad nimi provést další iteraci, kterou je prioritizace požadavků.

Všichni účastníci sběru a získávání požadavků následně obdrželi kompletní seznam všech požadavků a byli požádáni o přidělení priorit. Stupnice byla nastavena od 1 do 5, kde 1 představuje požadavek, který je nejméně důležitý, 5 naopak ten, který je nepostradatelný a systém by měl danou funkcionalitu rozhodně mít.

Každý z účastníků prováděl prioritizaci nezávisle na ostatních, přičemž neměl k dispozici výsledné hodnocení ostatních. Tím bylo dosaženo nezávislosti a zamezeno jakémukoliv ovlivňování. Po rozdělení priorit byly dokumenty posbírány a ohodnocení sečteno. Požadavky, které měly hodnoty mezi 11-15 byly zařazeny do první implementační fáze. Dále podle stejné stupnice byly zařazeny mezi další fáze, které jsou však mimo rozsah této práce. Při udělování priorit byly použity prvky metody zvané Priority Poker [9].

3.3 Rozdělení požadavků

V předešlých kapitolách a v úvodu docházelo ke sběru požadavků a přání od zainteresovaných osob. Sesbírané informace rozdělíme do následujících skupin: Business požadavky, Funkční požadavky a Nefunkční (kvalitativní) požadavky [10]. Každá ze skupin obsahuje jinou sadu požadavků a popisuje systém z jiného pohledu.

3.3.1 Business požadavky

Business požadavky popisují důvody, proč má být změna provedena a definují, co se má změnit [11]. Tyto požadavky vychází z potřeb majitelů platformy Pingl a uživatelů administrace. Odůvodňují a vysvětlují potřebu administraci vlastnit.

Požadavky vydefinujeme pomocí následující šablony:

> Jako <role> potřebuji <cíl/přání>, protože <přínos>

1. Jako majitel potřebuji přidávat nové podniky do platformy, protože neustále získáváme nové klienty.

2. Jako majitel potřebuji upravovat informace o existujícím podniku, protože se informace o podniku mění a aktualizují.

3. Jako majitel nepotřebuji upravovat menu jednotlivé restaurace, protože tuto aktivitu mohu udělat po přihlášení do administrace konkrétního podniku.

4. Jako majitel potřebuji vytvářet nové uživatele, protože po vytvoření podniku je potřeba založit účty pro administrátora a obsluhu.

5. Jako majitel potřebuji mít přehled nad uživateli v platformě, protože mi přehled pomůže predikovat tempo růstu platformy.

6. Jako majitel potřebuji mít přehled nad všemi objednávkami napříč podniky a v jednotlivých podnicích, protože tak mohu odhalit, že systém v podniku z nějakého důvodu nefunguje podle představ a pomocí analýzy dat zlepšovat platformu.

3.3.2 Funkční požadavky

Funkční požadavky (v citované knize využitá terminologie Funkční Specifikace) v tomto případě představují požadavky na začátku projektu, které popisují, co by měl systém dělat, jaké funkce by měl splňovat.[1].

Jejich získáváním, použitých metodách a následnou prioritizací se zabývaly kapitoly výše a jsou k dispozici v příloze B.

Dokument obsahuje celkem 62 unikátních funkčních požadavků. Z nich se po udělení priorit všemi zúčastněnými osobami 22 označilo jako důležité a budou zařazeny do první iterace vývoje. Dalších 10 funkčních požadavků po sečtení priorit bylo zařazeno do druhé fáze. Ostatní požadavky nebyly shledány jako důležité a jejich vývoj bude uskutečněn pouze v případě, bude dostatek zdrojů, v našem případě časových, na jejich implementaci.

3.3.3 Nefunkční požadavky

Nefunkční požadavky, nebo také kvalitativní požadavky jsou druhou částí systémových požadavků. Popisují, jak se má systém chovat a nastavují omezení funkcionalitám. Mezi skupiny, které vymezují nefunkční požadavky, patří například použitelnost, bezpečnost, spolehlivost, výkon, dostupnost, škálovatelnost a další [12].

Protože zde zpracováváme pouze frontend administrace pro již existující backend, požadavky na výkon lze naplnit jen z části, kterou může ovlivnit frontendová část. Jedná se tak například o načítání velkého množství dat.

Také u administrace není nutností, aby byla bleskově rychlá, protože pomalejší načítání určitě neodradí uživatele v jejím používání. Požadavky na zátěž při mnoha uživatelích také nemusí být specifikovány, počet uživatelů pravděpodobně nikdy nepřekročí nižší desítky. Požadavky na dostupnost jsou důležitým specifikem, v našem případně se však jedná o uzavřený systém pro konkrétní osoby, nemusíme tak řešit například přístupnost pro slabozraké nebo jinak postižené uživatele. I tak je vhodné, aby měl administrátor přístup k administraci ze všech svých zařízení, jako je mobilní telefon a počítač.

1. Systém musí být přístupný z moderních prohlížečů, kterými jsou Mozilla Firefox, Google Chrome, Safari Web Browser a Microsoft Edge.
2. Systém musí být responzivní - stránky si lze přehledně zobrazit na osobním počítači, tabletu i mobilním zařízení.
3. Systém je přístupný pouze s kombinací jména a hesla s právy hlavního administrátora, které přiděluje správce serveru.
4. Každý akce v systému vyvolaná uživatelem končí buď úspěchem nebo neúspěchem a systém viditelně informuje uživatele o chybě nebo úspěchu.

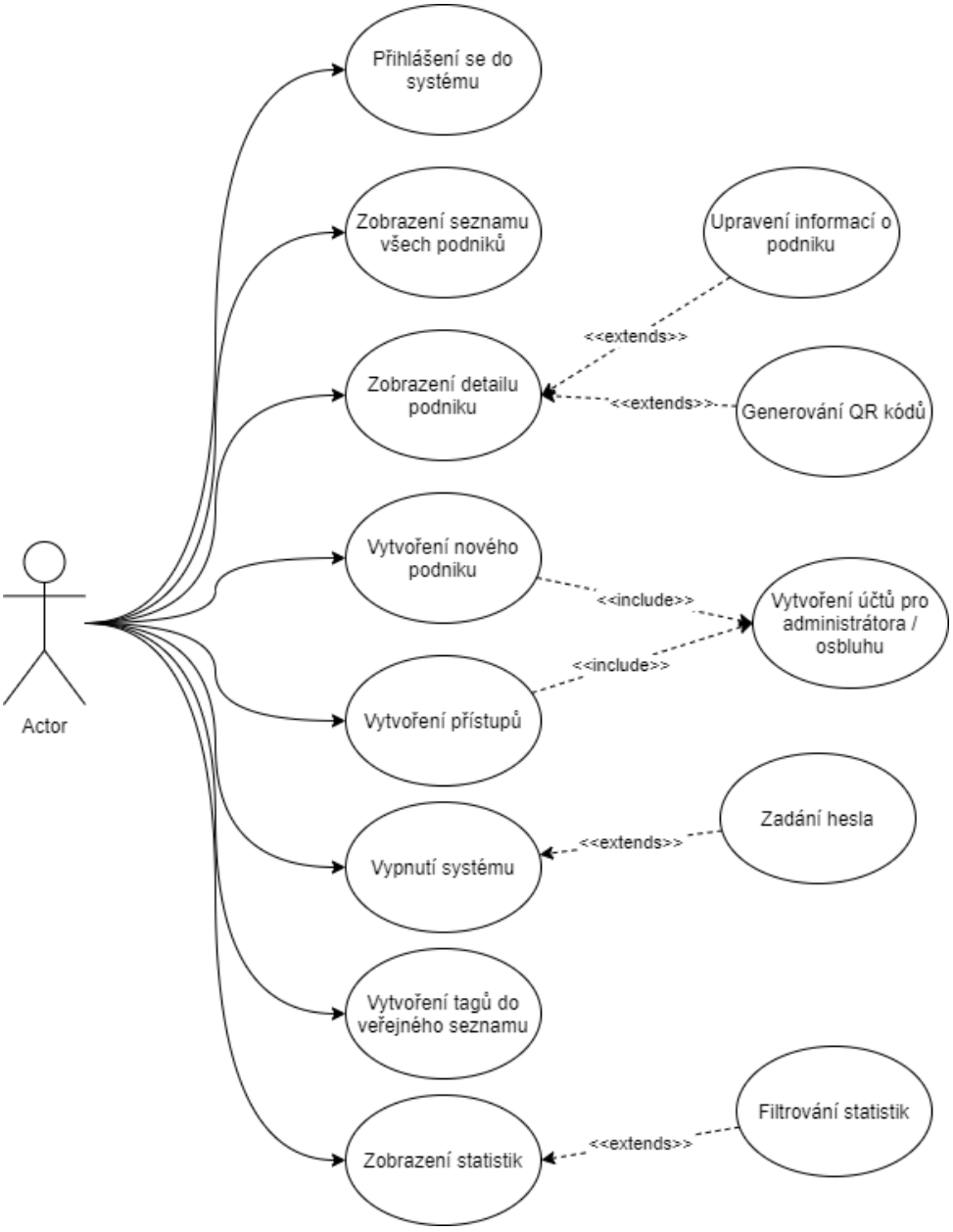
Při tvorbě frontendového rozhraní pro administraci bude tedy nejdůležitějším nefunkčním požadavkem bezpečnost, protože systém bude obsahovat všechna důležitá a citlivá data, stejně tak ovládací prvky, které budou provádět změny v produkční databázi.

3.4 Scénáře případů užití

Celkem u šesti funkcionalit byla po sečtení zaznamenána priorita 14 nebo 15 z maxima 15 bodů. Proto jsou tyto funkce považovány za stěžejní a budou na ně vypracovány scénáře případů užití, které detailněji popisují jejich využití v systému. Scénáře jsou z důvodu kompaktnosti práce v příloze C.

3.5 Diagram případů užití

Výsledné sesbírané požadavky, které budou v první fázi implementovány jsou zaznamenány v následujícím diagramu případů užití.



Obrázek 3.1: Diagram případu užití

Kapitola 4
Technologie

Kapitola probírá možnosti implementace požadovaného systému a využitelných technologií. Budou porovnány dva existující přístupy ke tvorbě webových aplikací a vybrán ten, který je pro tento konkrétní případ nejvhodnější.

4.1 Přístupy k tvorbě webových aplikací

Dnes můžeme webové stránky a aplikace rozdělit do následujících dvou skupin, které představují přístupy ke tvorbě webů. Z anglického Single-page Application (SPA) a Z anglického Multi-page Application (MPA).

Oba přístupy mají několik výhod a nevýhod. Při volbě, jaký přístup zvolit, je tedy důležité zvážit, jaký typ webové stránky nebo webové aplikace chceme budovat, jak má s uživatelem interagovat a jaký má být její obsah. Znalost těchto faktů pomůže vybrat vhodný postup pro toto konkrétní řešení.

4.1.1 Multi-page aplikace

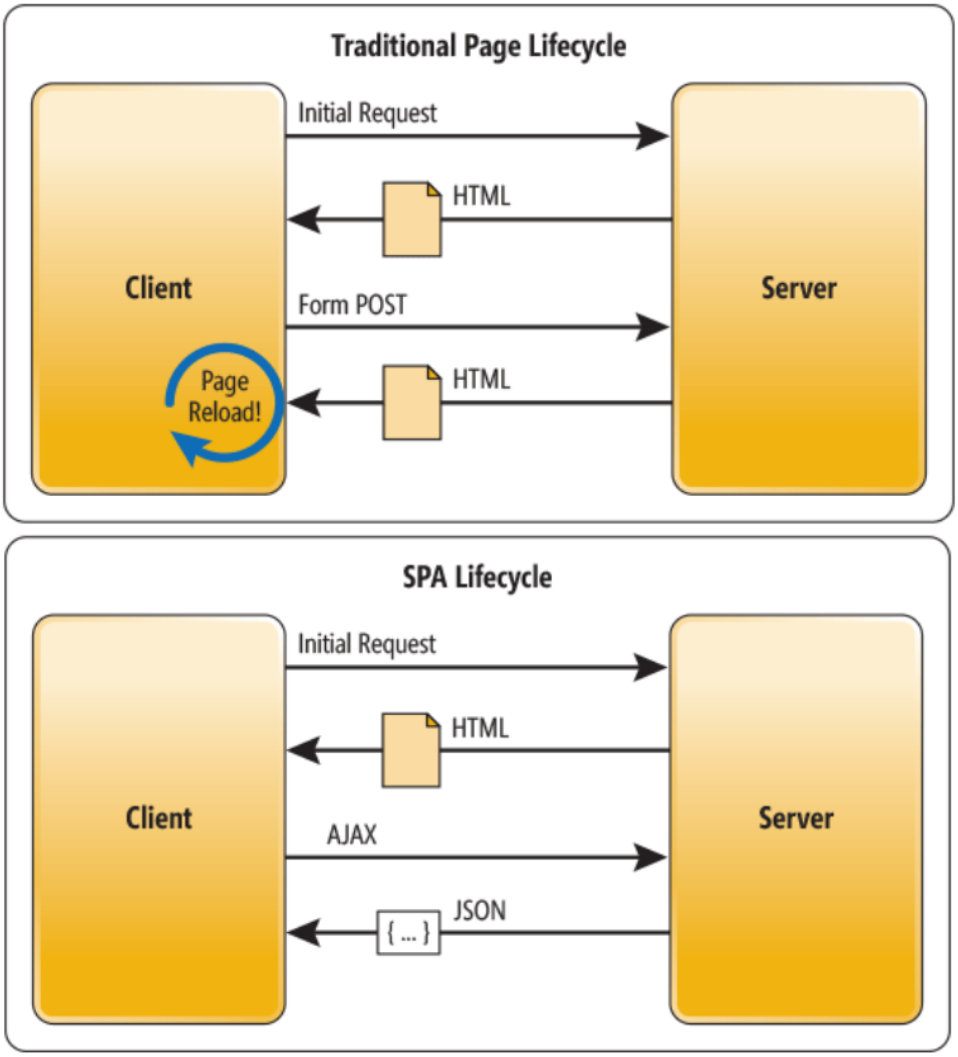
První přístup je historicky starší a proto většina webu stále na tomto principu funguje. Tento přístup využívá například většina známých open-source CMS, jako je Drupal nebo Wordpress, na kterém běží 35% všech webových stránek [13].

U tohoto přístupu je při každém dotazu na serveru vykreslena kompletní HTML stránka i s obsahem, která je vrácena klientovi. Pokud uživatel klikne na odkaz na stránce, nebo vyvolá nějakou změnu, která způsobí odeslání požadavku, dochází opět k překreslení celé stránky. Veškerá logika a vykreslování se tak odehrává na serveru, klient poté slouží jako "zobrazovač"statických stránek.

4.1.2 Single-page aplikace

"SPA je aplikace doručená do prohlížeče, který nemusí znovu načítat stránku během používání. Jako všechny aplikace, má za cíl pomoci uživateli dokončit nějaký úkol, jako "napsat dokument"nebo "spravovat webový server". Nad SPA můžeme přemýšlet jako objemném kódu na klientovi který je načten ze serveru."[14]

Tomu je docíleno díky AJAX dotazům a Web Socketům. Tyto technologie umožňují asynchronní získávání dát do stránky, dnes převážně v podobě JSON odpovědí ze serveru. Na základě nově příchozích dat je pomocí javascriptu upravován DOM a tím se úpravy projeví ve stránce. Vše se tak děje na klientovi, kterým je prohlížeč uživatele. Server zde na rozdíl od přístupu MPA vrací pouze surová data, zajišťuje autorizaci, ale samotné vykreslení obsahu zajišťuje klient.



Obrázek 4.1: MPA vs SPA - životní cyklus[15]

4.1.3 Výhody SPA

Obecně lze říct, že největší výhodou takových aplikací je jejich dopad na UX. Zároveň snižují náklady na hardware serveru, protože velká část výpočtů přesouvá na zařízení jednotlivých klientů.

Rychlost načítání

Většina zdrojů (HTML, CSS a Javascript) je načtena při prvním požadavku, jedou za celý životní cyklus aplikace. Následně se přenáší jenom data v textové podobě, která jsou značně menší než celá stránka. To přispívá rychlosti načítání [16].

Uživatelský dojem

Každá interakce s aplikací je pro uživatele příjemnější, protože nedochází k přesměrováním, uživatel tak nemusí čekat a všechny změny se dějí téměř okamžitě. Často se také načítá pouze část dat, proto uživatel zůstává na stránce a mění se jen odpovídající prvky, což opět aplikaci činí konzistentní a zlepšuje uživatelský dojem [17].

Offline fungování

Díky skutečnosti, že téměř všechny zdroje jsou načteny při prvním požadavku, lze je pomocí Cache Storage API uložit do cache prohlížeče a umožnit tak uživateli aplikace používat v omezeném množství i bez přístupu k internetu. Uživatel sice nebude dostávat aktuální data, může však aplikaci nadále používat a jakmile se jeho internetové připojení vrátí, lze data synchronizovat tak, aby uživatel nemusel aplikaci opouštět.[18]

Usnadněný vývoj

Tento bod je v některých starších zdrojích uváděn spíše jako nevýhoda a pravdou je, že záleží na sadě nástrojů, které je vývojář schopný využívat.

Na jednu stranu při vývoji SPA nepotřebujeme server. Vývoj můžeme začít okamžitě za použití localhostu. Nemusíme tak psát kód, který bude stránku vykreslovat na serveru. Při debugování můžeme celý kód zobrazit v nástroji prohlížeče, u Chrome například v Developer tools. U aplikacích renderovaných na serveru vidíme vždy jen tu část, kterou server vrátí [16]. Na druhé straně stojí znalost frontendových technologií. I přesto, že vytvořit jednoduchou single-page aplikaci jenom pomocí javascriptu lze, vývojář postupem času zjistí, že technologie a nástroje jako Webpack, Express, React a další jsou téměř nutnou součástí, kterou se musí také naučit [17].

4.1.4 Nevýhody SPA

Naproti všem zmíněným výhodám single-page aplikací stojí i několik nevýhod, které musíme brát v potaz při výběru přístupu. Protože je princip tvorby SPA stále více oblíbený, i díky populárním a nově vznikajícím javascriptovým frameworkům, některé nevýhody začínají být překonané.

Úvodní rychlost načítání

Negativní důsledek výše zmíněné výhody o rychlosti načítání v průběhu používání webové aplikace. Protože se celý kód načte ze serveru hned při prvním požadavku, při použití rozsáhlých frontendových frameworků může výsledný kód rychle bobtnat. Po stažení souborů je následně na prohlížeči, aby je vykreslil, což také nějaký čas trvá.

Tomuto problému můžeme u větších aplikací zabránit metodou zvanou Code splitting, v překladu rozdělení kódu. Tento přístup umožňuje rozdělit kód do menších částí, které následně můžeme načítat nezávisle na sobě a snížit tím tak objem kódu načítaného v úvodu [19].

Optimalizace pro vyhledávače

Internetové vyhledávače zjednodušeně fungují tak, že automaticky prochází webové stránky, zaznamenávají a ohodnocují jejich obsah a následně je zobrazují na relevantní hledané dotazy od uživatelů.

Algoritmy, které weby procházejí byly navrženy na čtení HTML souborů, které tradičně server vracel, ale neumějí už provolávat požadavky z javascriptu a získávat tak obsah, jak tomu je u single-page aplikací.

I tento problém lze řešit pomocí vynuceného vykreslení kódu na straně serveru, což umožňují některé javascriptové frameworky. Zároveň se některé algoritmy, například Googlebot od Google, naučili javascipt číst a zpracovávat [20].

Historie prohlížeče a navigace

V surové podobě má single-page aplikace pouze jednu URL. To znamená, že neexistují podstránky a není tak zaznamenávána historie změn. V konečném důsledku tak nefunguje historie v prohlížeči, nefungují tlačítka zpět a vpřed, nemůžeme si aplikaci v aktuálním stavu uložit mezi oblíbené záložky.

Tyto problémy mají v současné chvíli také řešení. Díky History API, které přišlo s HTML 5 můžeme programově měnit historii prohlížeče a simulovat tak různé chování na různých podstránkách [18].

4.1.5 Použití principu na náš případ

Výsledkem této práce bude administrační rozhraní pro platformu Pingl. U administrace optimalizaci pro prohlížeče řešit nebudeme, protože není důvod, proč by se odkazy na administraci měly zobrazovat ve vyhledávání. Úvodní rychlost načítání pro nás také není problém, o uživatele tím webová aplikace přicházet nebude.

S administrací bude uživatel interagovat ve velké míře. Často bude data upravovat, zobrazovat a přidávat. Proto by bylo velmi nežádoucí, aby se po každé změně musela celá stránka znovu načítat.

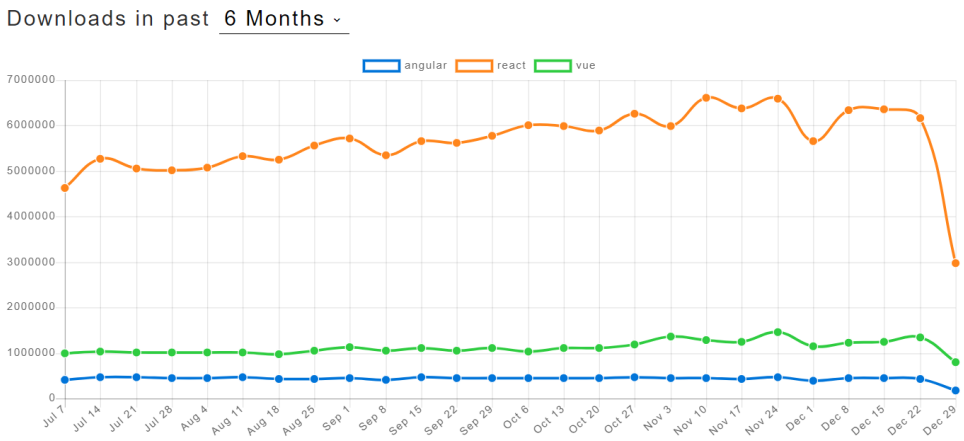
Po porovnání tak vidíme, že vyvíjet administraci jako SPA dává určitě větší smysl, než volit druhý zmiňovaný postup.

4.2 Javascriptové Frameworky

V předešlé sekci bylo rozhodnuto, že pro implementaci administrace bude využit princip single-page aplikace. Při jejich tvorbě v dnešní době pomáhá řada nástrojů a frameworků, které tvorbu tohoto typu webových aplikaci značně usnadňují.

Pomáhají například vyřešit zmiňované problémy, které se u single-page aplikací vyskytují. Jedná se třeba o routování a History API, které frameworky a knihovny umí řešit po svém a značně zjednodušeně. Umožňují tak vyhnout se znovu vynalézání kola [21]. A pomáhají mimo jiné i s výkonovou stránkou single-page aplikací.

Javascriptových frameworků existuje celá řada, nejčastěji jsou však zmiňovány a využívány tyto tři. Angular, Vue a React.



Obrázek 4.2: NPM trends - Angular vs React vs Vue

4.2.1 Angular

Angular je původně javascriptový, později v Typescriptu přepsaný framework vyvíjený společností Google, který byl oficiálně spuštěn v roce 2010. Za tu dobu prošel Angular několika změnami, nejvíce však z verze 1.3, která byla poslední stabilní před nově vydanou verzí 2.0. Ta, mimo ostatních změn, programátory nutila používat tehdy ne moc populární jazyk Typescript.

Popularita

Vývojářská komunita: V dnešní době ve verzi 8.2. je Angular stabilním a jedním z nejvíce používaných frameworků. Podle dotazníku Stack Overflow obsadil Angular třetí místo mezi webovými frameworky, když dostal hlas od 30.7% respondentů [22]. Na zmiňovaném portálu pro vývojáře má Angular téměř 260 tisíc vláken s dotazy[23], což je mnohem víc než ostatní dvě zde porovnávané možnosti. To je však zapříčiněné i stářím Angularu a počtem jeho verzí. Na GitHubu zatím tento framework získal od uživatelů 55.9 tisíc

hvězdiček [24].

Pracovní pozice: Množství otevřených pracovních pozic také potvrzuje jeho oblíbenost. Pracovní portál Indeed uvádí 12 343 [25] otevřených pracovních pozic na americkém trhu práce. LinkedIn Jobs celosvětově nabízí 37 826 pracovních [26] míst.

Firmy: Jelikož Angular je spravován společností Google, není překvapením, že ho využívají na svých službách a projektech, jako například u YouTube. Dále Angular používá například Microsoft, Cisco, Udemy, Paypal, Apple, nebo Amazon.

Výhody

Kompletní framework Angular je kompletní balíček nástrojů, které vývojář potřebuje pro jeho práci. Díky tomu se tak v průběhu vývoje nemusí instalovat další nástroje a učit nové technologie.

Komunita a dokumentovanost: Díky devítileté historii má za sebou Angular obrovskou komunitu vývojářů a nabízí také velké množství knihoven třetích stran. Je také skvěle dokumentovaný a díky skutečnosti, že framework spravuje společnost Google, je pravděpodobné, že bude ještě pár let existovat [27].

Založený na komponentách: Od verze 2, Angular přešel z MVC na architekturu založenou na komponentách. Aplikace je tak rozdělena do nezávislých komponent, která zodpovídají za jednotlivé funkcionality. Tyto komponenty tak mohou být snadno znovu využity v jiných částech aplikace, nebo při jiných projektech a urychlují tak vývoj.

Ivy Renderer: Ivy Renderer je Angularový překladač, který komponenty a šablony překládá do javascriptu a HTML, které lze zobrazit v prohlížeči. Hlavní jeho výhodou je vlastnost zvaná "tree shaking", která při překladu kódu odebere části, které výsledná aplikace nevyužívá. Díky tomu je výsledný kód menší a webová aplikace se tak rychleji načítá [28].

Vkládání závislostí: Další vlastností Angularu, která je však spekulativní a někdy také uváděna jako nevýhoda, je Dependency injection (v překladu vkládání závislostí). Tyto závislosti definují, jak jsou jednotlivé soubory propojeny a ukazují, jak změna v jedné komponentě ovlivní další. Na jednu stranu pomáhá udržet kód lépe dokumentovaný, udržitelný a usnadňuje psaní testů. Na druhou stranu může být vkládání závislostí časově náročnější a další věcí, co se musí vývojář naučit [27].

Nevýhody

Velikost: Skutečnost, že je Angular kompletní webový framework spolu však nese i negativní aspekty. Prvním je jeho velikost. Přímo není uvedena, ale odhaduje se, že je okolo 500KB, což je několikanásobně více, než u porovnávaných konkurentů [29].

Učící křivka: Druhou nevýhodou spojenou s robustností Angularu je jeho učící křivka. Vývojář se totiž musí naučit pracovat se všemi jeho částmi a aspekty. Systém založený na komponentách od verze 2 je sice jistým zjednodušením, ale například zmiňované Dependency injection je stále časově náročná záležitost.

Zpětná kompatibilita: Tento problém byl u Angularu zmiňován primárně s přechodem z Angularu 1.3 na Angular 2.0. Tyto verze mezi sebou vývojář kombinovat nemůže, proto se na starší verzi odkazujeme jako na AngularJS a na novější jako Angular. Je však potřeba říci, že od zmiňované verze 2.0 tento framework zpětnou kompatibilitu zajišťuje [27].

4.2.2 React

React je spíše než framework javascriptová knihovna oficálně publikovaná v roce 2013 společností Facebook, která React i nadále vyvíjí a spravuje [29]. Na oficálních stránkách s dokumentací k Reactu je napsáno, že React je knihovnou pro tvorbu uživatelských rozhraní. Na rozdíl od Angularu se tedy v přirovnání ke známe MVC architektuře soustředí pouze část View (v překladu "pohled"). Ostatní části si vývojář musí obstarat a doplnit mimo React [30].

Popularita

Vývojářská komunita: React se mezi vývojáři těší veliké oblibě, která stále stoupá. V průzkumu prováděným serverem Stack Overflow obsadil React s 31.3% druhé místo, hned za knihovnou JQuery, která je stále z historických důvodů nejrozšířenější [22]. K průzkumu je také uvedeno, že tento rok více vývojářů řeklo, že používají React.js spíše než Angular, na rozdíl od minulého roku.

Na GitHubu tato knihovna má v současné chvíli 142 tisíc hvězdiček [31], což je v porovnání s Angularem více než dvojnásobek. Co se počtu vyhledávaných dotazů na StackOverflow týká, je React s téměř 180 tisíci dotazy za Angularem, což může mít za důsledek zmiňované stáři prvního z frameworků [32].

Pracovní pozice: Podle portálu Indeed je k dispozici 13,874 pracovních míst na pozici React vývojáře [33]. Na pracovním portálu LinkedIn Jobs je uvedeno přes 37 tisíc volných míst [34]. Obě tato čísla jsou srovnatelná s předchozími u frameworku Angular, což potvrzuje jejich oblíbenost.

Firmy: Mimo Facebook a Instagram, u kterých není React překvapením, knihovnu využívají také firmy jako Airbnb, Uber, Reddit nebo Twitter [35].

Výhody

Učící křivka: React má velice dobře zpracovanou dokumentaci s odkazem na veliké množství návodů a tutoriálů. Díky tomu a skutečnosti, že jeho jádro obsahuje jen to nejnutnější, nový vývojář se s ním naučí relativně rychle pracovat.

Komunita a historie: Podobně jako u Angularu, i React má vedle sebe obrovskou komunitu lidí a vývojářů, kteří jej využívají. Díky tomu tak existuje nejen spousta návodů, ale i knihoven třetích stran, které může vývojář následně využívat a usnadnit si tak práci.

Zpětná kompatibilita: Knihovna se samozřejmě neustále vyvíjí. S verzí 16.8 se objevil nový přístup psaní komponent zvaný Hooks. Ty umožňují psát komponenty v Reactu bez nutnosti deklarace tříd. Celý kód se tak zjednodušuje a je lehčí pro naučení pro nově příchozí. React však tuto možnost nabízí a pokud vývojáři, kteří se naučili psát kód postaru, ho nechtějí přepisovat, tak Hooky využívat nemusí. Díky tomu tak zůstává komunita kolem Reactu stabilní .

Virtual DOM: Virtual DOM je koncept v programování, při kterém se drží virtuální reprezentaci uživatelského rozhraní v paměti a je synchronizována s "reálným"DOM v prohlížeči knihovnou ReactDOM [36].

Díky porovnávání Virtual DOM s reálným DOM umí React měnit pouze ty části, které jsou potřeba a nepřekreslovat celou stromovou strukturu pří každé změně, jak tomu bylo předtím. Tento přístup má obrovský pozitivní dopad na výkon [37].

Velikost knihovny: Protože React obsahuje jen nejnutnější části, má v základní podobě pouze 6.4kB [38]. Společně s ReactDOM knihovnou pak okolo 120kB [39]. A velikost knihoven má za následek rychlejší načítání celé výsledné aplikace.

Nevýhody

Neúplnost knihovny: Se zmiňovanou výhodou kompaktnosti Reactu přichází i nevýhoda. Ta se začne projevovat, když se aplikace psaná s pomocí této knihovny začne rozrůstat. Vývojář tak zjistí, že mu jenom React nestačí a musí knihovnu doplnit o další, jako je například Redux, který pomáhá spravovat stav aplikace, naučit se pracovat s nástroji jako Webpack, Babel nebo knihovnou Next. Tyto faktory nakonec zapříčiní, že se učící křivka Reactu zvedá a je komplikovanější s ním pracovat.

Existují však řešení, která poskytují nakonfigurované celé prostředí. Je jím například řešení přímo od Reactu zvané Create React App. Vývojáři, který se s Reactem učí, tak umožní začít bez nutnosti všechny ostatní nástroje znát.

Jazyk JSX: Další překážkou, která stojí novým vývojářům v cestě naučit se React, je syntaxe zvaná JSX. Jedná se o kombinaci javascriptu a HTML, která se v Reactu využívá k psaní komponent. Tento přístup přináší i některé výhody, pro nového vývojáře se však opět jedná o další novinku nutnou k naučení. [40] Ne však nezbytně, React JSX nevyžaduje a knihovnu lze využívat i s javascriptem, jak uvádí ve své dokumentaci [41]. Jedná se však o značně nepohodlný způsob a drtivá většina vývojářů využívá první přístup.

4.2.3 Vue

Ze zmíněných frameworků je nejmladší. V roce 2014 byl zveřejněn bývalým zaměstnancem Googlu Evanem You. Framework je stále více populární i přesto, že za sebou nemá na rozdíl od jeho porovnávaných soků velkou společnost, která by jej zaštiťovala.

Popularita

Vývojářská komunita: I přes mládí frameworku a konkurenci Rectu a Angularu má na GitHubu Vue 155 tisíc hvězdiček, nejvíce z trojice porovnávaných. [42]. Podle očekávání má však s číslem 47 360 [43] nejméně otázek na StackOverflow a v průzkumu javascriptových technologií, který tento portál v roce 2019 prováděl obsadil 7. místo s 15% hlasy [22].

Pracovní pozice: Na trhu práce v porovnání s Reactem a Angularem, Vue zaostává. Důvod, proč si méně firem tento framework vybírá může být ten, že za technologií nestojí žádná z velkých firem a obávají se tak o jeho dlouhodobou existenci a udržitelnost. Na pracovním portálu Indeed se pro americký trh nabízí 1 825 pozic[44], LinkedIn Jobs celosvětově nabízí 18 680 míst [45].

Firmy: Mezi nejznámější firmu, která se rozhodla věřit mladému frameworku, patří čínský gigant Alibaba. Dále se k vývoji na některé projekty rozhodli Vue použít firmy jako Xiaomi, Adobe nebo Gitlab [46].

Výhody

Jednoduchý na naučení: Podobně jako Angular, Vue je kompletní webový framework, který obsahuje většinu toho, co vývojář potřebuje, včetně Vue CLI nebo Vue UI. Začít jej využívat je tak otázka pár minut a hodí se tak pro tvorbu jednodušších projektu nebo prototypů, které mají být rychle vyvinuty [47]. Má tedy vestavěný celý MVC model, což usnadňuje počáteční konfiguraci [48].

Dobře dokumentovaný: V dotazníku, který prováděla softwarová společnost Monterail společně se zakladatelem Vue Evanem You a členem Vue

týmu Chris Fritz je druhou největší výhodou frameworku Vue jeho dobrá dokumentovanost. 94% respondentů ve zmiňovaném průzkumu také uvádí, že při učení se frameworku používali oficiální dokumentaci [49].

Výkon: I přes skutečnost, že se jedná o kompletní framework, je velikost Vue ve verzi 2.6.11 okolo 63kB, což s porovnání s Angularem je značný rozdíl [50]. Díky tomu jsou výsledné webové aplikace menší a snižuje se tím čas načtení. Vue, stejně jako React využívá principu Virtual DOM a díky tomu dosahuje lepších výsledků při vykreslování změn v reálném čase [48].

Nevýhody

Nedostatek zkušených vývojářů Protože je Vue novým frameworkem, vývojáři se s ním učí pracovat a není příliš velká komunita, která by s ním měla mnohaleté zkušenosti. Z toho důvodu hrozí, že při snaže Vue využívat při tvorbě rozsáhlých projektů vývojáři narazí na problém, který tento framework neřeší, ale jeho starší konkurenti už ano.

Více možností vývoje: Jednoduchost a flexibilita Vue je dvousečná. Protože umožňuje vývojářům více cest, jak řešit ten samý problém, může tak vznikat špatný kód, ve kterém se těžko hledají chyby a je náročnější na testování [49].

4.3 Výběr a zdůvodnění

Po porovnání třech v současnosti nejvíce používaných javascriptových frameworků a knihoven nemůžeme určit, že by některý jednoznačně převyšoval ostatní. Navíc, jednotlivé výhody a nevýhody, které v minulosti určovaly oblíbenost těchto nástrojů na tvorbu single-page aplikací se časem ztrácí, protože organizace a vývojáři na nich pracující čerpají inspiraci od svých konkurentů a využívají podobných postupů, se kterými jiný framework uspěl a lidé se jej tak díky tomu oblíbili.

Z výše uvedených důvodů by se tak dal vyloučit pouze framework Vue, kvůli jeho nevyzrálosti a prozatím malému, i když vzrůstajícímu zájmu.

Z dvojice Angular - React bude v tomto projektu využita knihovna React. Hlavním důvodem je její využívanost ve firmě Pingl s.r.o., pro kterou je administrace vyvíjena, i na ostatních projektech. To je velice důležitý faktor, který byl zmíněn v úvodu do problematiky a neměl by být opomenut. Firma tak nebude využívat různé technologie a nebude muset hledat nové vývojáře s jinou sadou znalostí. Ostatní vývojáři z firmy Pingl s.r.o. tak díky znalosti Reactu budou moci projekt později rozšiřovat a udržovat.

Není to však jediný důvod, výhod Reactu je více. Jak bylo zmíněno výše, je v současnosti nejpopulárnějším knihovnou pro tvorbu single-page aplikací, je dokumentován, spravován velkou firmou a jeho existence je tak na dalších několik let zaručena. Proto by byl volbou i v případě, že by soubor technologií ve firmě zatím nebyl stanoven.

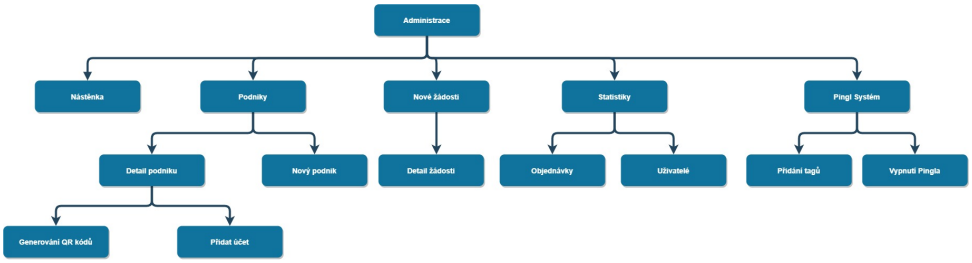
Kapitola 5

Návrh

Na základě předchozích kapitol a sesbíraných požadavků bude učiněn návrh uživatelského rozhraní. Nejprve bude nadefinována struktura stránek webové administrace a následně vytvořen wireframe.

5.1 Mapa stránek

Z požadavků sesbíraných v první části sestavíme mapu stránek. Ta bude sloužit k určení struktury webu a poslouží při návrhu wireframu, pozdějším návrhu uživatelského rozhraní a následné implementaci.



Obrázek 5.1: Mapa stránek

5.2 Wireframy

Wireframy, v překladu "drátěný model", je ilustrace webové stránky, která znázorňuje rozložení obsahu a funkcí webu, se kterými uživatel interaguje. Zobrazují, které elementy budou zobrazeny na kterých konkrétních stránkách.

Umožňují tak vytvořit rychlou vizualizaci navrhovaného webu, kterou může klient posoudit, poskytnout zpětnou vazbu a tím předejít nutným změnám během implementace. Zároveň slouží k následné tvorbě grafického zpracování webové stránky.

Při tvorbě wireframů se nepoužívají barvy, obrázky ani jiné prvky designu. Pro vytvoření wireframů byl využit nástroj Adobe XD. Wireframy jsou součástí přílohy D.

Kapitola 6
Implementace

Na základě předchozích kapitol, připravených wireframů a vybraných technologií bude provedena implementace projektu.

6.1 Příprava

Před zahájením vývoje je důležité nastavit prostředí pro vývoj. Pro vývoj byla využita předpřipravená šablona, kterou pro vývoj projektů v Ping s.r.o. využíváme. Jedná se o nastavení řady nástrojů, které usnadňují vývoj.

6.1.1 Vývojové prostředí

Pro psaní zdrojového kódu bude využit editor VS Code [51]. Díky dlouhodobé zkušenosti s tímto nástrojem a jeho popularitou mezi vývojáři ve vybraném frameworku React je volbou číslo jedna. Velkou výhodou je, že nabízí sadu rozšíření, které pomáhají s psaním lepšího kódu. Při psaní kódu tak budou využity následující rozšíření:

- **ESLint** [52] je rozšíření, které pomáhá detekovat a opravit chyby v JavaScriptu nebo Typescriptu. Druhý zmíněný jazyk bude využit i pro implementaci.
- **Stylelint** [53] je podobný s výše zmíněným ESLintem, soustředí se však na správnost psaní CSS pravidel.
- **Prettier**[54] má na rozdíl od ESLintu na práci pouze formátovat kód, nikoliv opravovat chyby. Toto nastavení se následně sdílí mezi vývojáři pracující na projektu a tím je dosaženo stejného formátování ze všech zdrojů. Díky tomu pak nevznikají problémy v Gitu při porovnávání změn.

6.1.2 Základní šablona

Jak bylo v předchozích kapitolách rozhodnuto, projekt bude postaven jako SPA na technologii React. K zahájení vývoje je současně s přípravou editorů kódu potřeba nastavit zmíněné technologie

Create React App Je již předpřipravená sada nástrojů společně se Reactem od vývojářů z Facebooku. Projekt je také přímo doporučený pro vývoj single-page aplikací v Reactu. Přesně tak splňuje předchozí analýzu a kopíruje předchozí rozhodnutí.

Ostatní doporučená řešení pro vývoj aplikací s knihovnou React v oficiální dokumentaci jsou například Next, nebo Gatsby. První jmenovaná knihovna je vhodná pro aplikace, které vyžadují renderování na straně serveru. Gatsby zase umožňuje vývojářům využívajících React generovat statický obsah. Ani jeden z těchto projektů však nemá pro webovou administraci výhody a proto bude využita zmíněná sada Create React App.[55]

6.1.3 Programovací jazyk a syntax

I přesto, že již proběhla volba knihovny pro tvorbu administrace, kterou vyhrál React, stále je zde jistá svoboda při volbě konkrétního způsobu, jak psát za pomocí této knihovny finální kód.

JavaScript

JavaScript je programovací jazyk, který běží v prohlížeči a do kterého se výsledný kód vytvořený za pomocí Reactu kompiluje. V knihovně tak lze použít přímo tento jazyk a vše bude fungovat jak má. Tento přístup je vhodný pro vývojáře, kteří mají s čistým JavaScriptem bohaté zkušenosti a nechtějí se učit žádnou nadstavbu.

JSX

JavaScript XML (JSX), je rozšířená syntax jazyku JavaScript. React zastává princip Oddělení zodpovědností [56], na rozdíl od tradičnějšího přístupu u webových aplikací - rozdělování podle technologií, jako jsou HTML, CSS, JS soubory. Z tohoto důvodu vznikla i nová syntax JSX, která si dává za cíl zjednodušit psaní kompletních funkčních komponent bez nutnosti rozdělovat technologie. I proto bude tento syntax využit při implementaci.

TypeScript

TypeScript sám sebe nazývá jako typovanou nadmnožinu JavaScriptu, která se přímo do JS kompiluje. Funguje tak v každém prohlížeči a na jakémkoliv operačním systému. Projekt je otevřený všem a je spravován společností Microsoft. [57]

TypeScript podporuje kontrolu chyb a kompilaci JSX souborů přímo do JS. Aby mohl Typescript fungovat s JSX, je potřeba souborům dávat příponu .jsx a povolit jej v nastavení Typescriptu v projektu. [58]

Využití této kombinace je stále častější a populárnější. Přináší s sebou tak řadu výhod a nepřímo se stává standardem pro vývoj webových aplikací v Reactu.

6.2 Architektura

Sekce rozebírá architekturu projektu, rozvržení souborů, vrstev a principy použitých technologií.

6.2.1 Adresářová struktura

V projektu se nachází sada složek a souborů, které mají svou specifickou funkci a obsahují konkrétní sadu souborů. Přehledná adresářová struktura tak usnadňuje orientaci v souborech stávajícím, i nově příchozím vývojářům. Existují tak doporučené principy, které by se v projektech využívajících totožnou sadu technologií měly dodržovat.[59]

- **Public** složka obsahuje veřejné statické soubory, které se málo kdy mění. Jejím nejdůležitějším prvkem je soubor **index.html**, bez kterého by aplikace nefungovala. Ten v sobě totiž obsahuje s id="root", ke kterému se připne celý JavaScriptový soubor obsahující celou aplikaci a ta je tím načtena.

- **Components** složka obsahuje takzvané "hloupé komponenty". Jedná se pouze o ty části, které přijmou data a zobrazí je. V komponentách nacházející se v této složce by nemělo docházet k editaci dat, stahování dat z API atd.

  Ve složce se však také nachází důležitý soubor **App.tsx**. Tento soubor v sobě sdružuje všechny komponenty aplikace, které se následně renderují na zmiňovaný s id="root".

- **Containers** oproti složce Components sdružuje komponenty, které v sobě nesou určitou funkční část. Může se jednat například o formuláře nebo jiné interaktivní prvky, díky kterým dochází k manipulaci s daty.

- **Constants** složka v sobě má konstanty používané napříč projektem. Jedná se například o barvy, typy fotnů nebo chybové hlášky. Díky vytažení těchto na víc místech používaných konstant je projekt přehlednější a v případě nutné změny stačí úpravu provést na jednom místě.

- **Data** je místo, kde se nachází celý vnitřní stav aplikace. V tomto projektu je využita knihovna Redux, která bude představena a rozebrána podrobněji v následující sekci. [60]

- **Pages** pojímá komponenty ze složek Components a Containers a skládá je do funkčních celků - stránek. Každý soubor ve složce tak představuje určitou stránku webové aplikace, kterou může uživatel navštívit.

  Rozdíly mezi stránkami se u single-page aplikace mohou stírat, jelikož se v základu jedná pouze o jednu stránku překreslující komponenty na základě uživatelských akcí. Proto byl do projektu přidán i react-router, který tento problém řeší. Jeho funkce jsou rozebrány v kapitole níže.

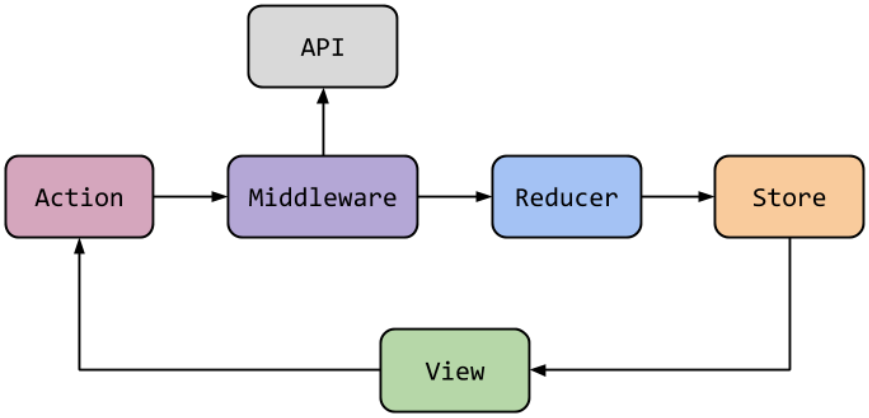
- **client.tsx** soubor obsahuje základní nastavení API klienta, v našem případě knihovny apollo-client, která usnadňuje volání dotazů na GraphQL API, které vystavuje backend a ze kterého aplikace čerpá veškerá data. Tento přístup je opět rozebrán v následující sekci.
- **index.tsx** je hlavním JavaScriptovým souborem, který se stará o připnutí a vykreslení celé aplikace na zmiňovaný hlavní HTML tag. Soubor obsahuje sdružené komponenty - soubou App.tsx. Mimo to zde může vývojář přidat i další knihovny a rozšíření, které pro projekt potřebuje a které "obalí"celou aplikaci a umožní využívání zmíněných knihoven pro všechny komponenty.

6.2.2 Redux

Častou otázkou u robustnějších single-page aplikací je spravování a správný návrh vnitřního stavu aplikace. K tomu slouží knihovna Redux.

Tato knihovna pomáhá javascriptovým aplikacím s psaním konzistentních aplikací. Mezi její výhody patří strukturované uchovávání dat, snadné testování a sada výhod pro vývojáře při tvorbě aplikace jako je možnost měnit stav aplikace za běhu.

Architektura aplikace využívající redux se vyznačuje tím, že data, která jsou v aplikaci uchovávána se neupravují, jsou takzvaně "immutable". [61]



Obrázek 6.1: Redux one way data flow + middleware [62]

Na obrázku je schéma fungování aplikací využívající Redux. Níže jsou rozebrány jeho dílčí části. Adresářová struktura Reduxu zde popsaná přebírá jeden rozšířených postupů zvaný "Rails-style".[60]

- **Store** je centrálním úložištěm dat v aplikaci. Uchovává tak celý její stav. Jeho hlavní vlastností je neměnnost, což znamená, že se data neupravují, ale vždy se vytvoří nový otisk celého stromu dat a použije se místo toho starého. Díky tomu je tak možné během vývoje se "vracet v čase"a posunou stav celé aplikace o krok zpět.

- **Action** "Akce je jediný způsob, jak lze změnit data v hlavním úložišti dat aplikace, ve Store. Je to objekt, který popisuje, jaká změna se má stát".[63]

- **Reducer** Jak se aplikace rozrůstá, není už vhodné vše dávat přímo do hlavního Storu aplikace, ale rozdělit datový model do funkčních celků. Těm se říká reducery a představují malý store pro konkrétní část datového modelu. Následně jsou ve Store spojeny pomoci funkce combineReducers a aplikace na výsledku obsahuje opět jednu stromovou strukturu dat.

- **Middleware** v této práci představuje knihovna redux-thunk. Zajišťuje, že asynchronní volání akcí a dlouho trvající procesy, jako je stahování většího množství dat ze serveru, doběhnou bez vedlejších efektů. Middleware není přímou součástí reduxu, je však velmi vhodné jej použít u rozrůstajících se aplikací s větším počtem volání asynchronních požadavků.

- **Selector** je jednoduchá funkce, která vytahuje data ze Store do komponent.

- **Constants** jsou konstanty, unikátní jmenné identifikátory akcí. Díky ním tak reducer pozná, jaká akce se volá a upraví tak odpovídající informace ve storu.

Tato architektura se u menších aplikací může jevit jako příliš složitá a robustní, protože vyžaduje větší množství kódu. Nicméně pro rostoucí a rozšiřující se aplikace je tento návrh výhodný, protože roste společně s aplikací. Značnou výhodou jsou také zmíněné vývojářské nástroje a zjednodušené testování celé aplikace díky centralizovaným datům.

```
25  const DashboardReducer: Reducer<IDashboardState, IDashboardActions> = (
26    state = initState,
27    action
28  ) => {
29    switch (action.type) {
30      case CONST.LOAD_ALL_USERS_SUCCESS:
31        return { ...state, users: action.payload }
32      case CONST.LOAD_ALL_ORDERS_SUCCESS:
33        return { ...state, orders: action.payload }
34      case CONST.LOAD_ALL_RESTAURANTS_SUCCESS:
35        return { ...state, restaurants: action.payload }
36      case CONST.SAVE_ALL_DEMANDS_SUM:
37        return { ...state, demands: action.payload }
38      default:
39        return state
40    }
41  }
42
43  export default DashboardReducer
```

Obrázek 6.2: Ukázka kódu - reducer

6.2.3 GraphQL

Z důvodu, že připravený backend, na který se celá administrace dotazuje, využívá dotazovacího jazyku GraphQL, bude tato možnost při implementaci využita.

Skutečností však zůstává, že tento přístup přináší sadu výhod. Na rozdíl od REST API má GraphQL pouze jediný endpoint, který obsahuje veškerá veřejně dostupná data. Vývojář si tak může sám definovat, jaká data si chce z daného endpointu stáhnout a nemusí kvůli sadě dat nutně komunikovat s tvůrcem backendu. [64] Požadavky lze volat ve dvou základních podobách:

- **Query** je prvním požadavkem v GraphQL. Query je realizována jako GET požadavek. Klient se dotáže serveru na data a ten mu odpoví. [65]
- **Mutation** umožňuje klientovi, ze kterého je dotaz volán, měnit nebo upravovat data na serveru. Je realizována POST požadavkem. [66]

V projektu bude využit klient pro volání GraphQL dotazů zvaný Apollo.[67] Jedná se o nejrozšířenější implementaci pro knihovnu React, která vývojářům usnadňuje komunikaci s GraphQL API díky předpřipraveným komponentám a funkcím.

Jednou z nesporných výhod je možnost generovat schéma, na které se lze dotazovat. Díky tomu tak získáme kompletní seznam dostupných Queries a Mutations, jejich možní parametry a případné vstupní parametry. S využitím Typescriptu v našem projektu tak obdržíme otypované veškeré odpovědi, což extrémně usnadní vývoj a zamezí chybnému volání dotazů.

Následující příkazy, které je možné v projektu volat zajistí, že se na klienta stáhne otisk všech dostupných dat a následně se podle definovaných dotazů vygenerují jejich typy v jazyce Typescript. Tyto příkazy se využívají napříč projekty ve společnosti Pingl s.r.o.

```
45  "schema:fetch": "apollo schema:download --endpoint=https://dev.backend.pingl.app/graphql graphql-schema.json",
46  "schema:generate": "apollo client:codegen --target typescript --localSchemaFile=graphql-schema.json --globalTypesFile=src/__generate__/globalTypes.ts"
```

Obrázek 6.3: Příkazy pro generování schema

6.2.4 Zabezpečení

Z podstaty projektu a požadavků definovaných na začátku vyplynulo, že aplikaci může využívat jen určitá úzká skupina uživatelů. Z tohoto důvodu je veškerý obsah aplikace schovaný za přihlašovacím formulářem.

Po zadání správného jména a hesla obdrží klient ze serveru autorizační token. Přítomnost tokenu se následně kontroluje a v v momentě, kdy je zaznamenán, dochází k vykreslení zbytku aplikace.

Autorizační token se uloží do local storage a následně je přidáván ke všem požadavkům. Server poté ověřuje platnost tokenu pro každý požadavek, který není veřejný (společné s veřejnou aplikaci, jako například seznam všech restaurací). Tento token má určitou platnost a po jejím vypršení je uživatel odhlášen.

```
27  const App: React.FC = () => {
28    const isAuthenticated = useSelector(selectIsAdmin)
29
30    if (!isAuthenticated) {
31      return (
32        <Router>
33          <GlobalToast />
34          <Route path={routes.index} component={Login} />
35        </Router>
36      )
37    } else {
38      return (
39        <React.Fragment>
```

Obrázek 6.4: Hlavní větev celé aplikace - rozhoduje o zobrazení po přihlášení

```
42  const langLink = setContext((_, { headers }) => {
43    // return the headers to the context so httpLink can read them
44    return {
45      headers: {
46        ...headers,
47        'Accept-Languages': 'en',
48        Authorization: 'Bearer ' + getLocalStorage(LocalStorageKeys.USER_TOKEN),
49      },
50    }
51  })
```

Obrázek 6.5: Autorizační token, který se přidává k požadavkům

Veškeré další validace si následně řeší hotový backend, který podle tokenu a uživatelských rolí vrací buď požadovaná data, nebo chybu.

6.2.5 React router

Jak bylo zmíněno v analýze, rozdíl single-page aplikací od tradičních stránek je také v tom, že v základu běží na jedné URL. To však může být u složitějších aplikací problém. Uživatelé jsou zvyklí, že se při navigaci ve stránce mění i URL. A co víc, tato skutečnost se navíc promítá i do navigace na stránce pomocí šipek vpřed a zpět, které kvůli tomu nefungují tak, jak je uživatel zvyklý. [68]

Abychom dali uživatelům funkcionalitu, na kterou jsou zvyklý, musíme stránkování explicitně definovat. K tomu skvěle slouží knihovna react-router. Ačkoliv základní šablona Create React App jí neobsahuje, je React Router svým způsobem standardizovanou knihovnou pro potřeby navigace mezi stránkami v projektech využívajících react.

Díky této knihovně a její zabudované práci s History API lze přepisovat URL a přidávat do historie stránky pro vykreslení. Když následně uživatel použije navigaci prohlížeče, dostane se na stránku zpět a vpřed.

Rozdíl z pohledu implementace a vývoje je následující. Nejprve se musí definovat výčet všech stránek, které uživateli nabízí oddělenou funkcionalitu. Následně se pro navigaci mezi stránkami nepoužívá standardní HTML tag pro odkaz <a>, ale komponenta <Link /> zabudovaná v knihovně. Ta zajistí přesměrování mezi stránkami, aniž by se musela znovu stahovat a vykreslovat

celá aplikace, jak by tomu bylo při použití značky pro odkaz.

```
return (
  <React.Fragment>
    <GlobalToast />
    <Router>
      <Switch>
        <RouteWrapper exact path={routes.index} component={Dashboard} />
        <RouteWrapper exact path={routes.venues} component={Venues} />
        <RouteWrapper path={routes.venueDetail} component={VenueDetail} />
        <RouteWrapper path={routes.createVenue} component={CreateVenue} />
        <RouteWrapper exact path={routes.demands} component={Demands} />
        <RouteWrapper path={routes.demandDetail} component={DemandDetail} />
        <RouteWrapper path={routes.statistics} component={Statistics} />
        <RouteWrapper path={routes.system} component={System} />
        <RouteWrapper component={NoMatch} />
      </Switch>
    </Router>
  </React.Fragment>
)
```

Obrázek 6.6: React router v komponentě App.tsx

6.3 Další funkcionalita

V předchozí kapitole byla rozebrána architektura a struktura aplikace, včetně primárních technologií využitých pro její implementaci. Tato kapitola představuje některé další funkce platformy.

6.3.1 Responzivní zobrazení

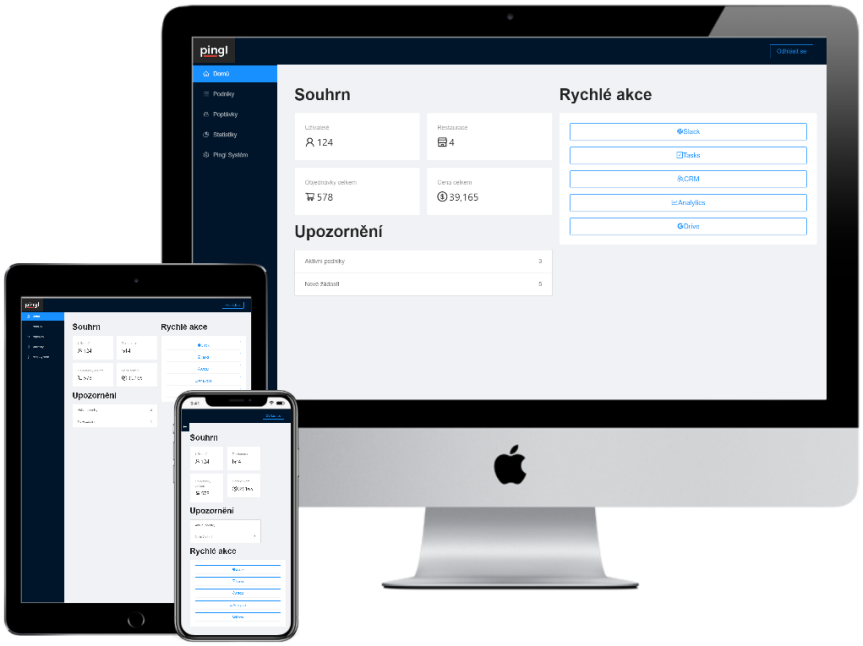
V nefunkčních požadavcích na začátku práce bylo také definováno, že webová administrace musí být responzivní. To znamená, že se musí přizpůsobit velikosti displaye různých zařízení, jako jsou mobilní telefony, tablety a osobní počítače.

Díky využití knihovny pro tvorbu uživatelských rozhraní Ant Design [69] bylo jednodušší tohoto efektu dosáhnout. Knihovna má spoustu komponent a Layout, nebo rozložení stránky, které umožňuje definovat počet sloupečků zobrazujících se v závislosti na velikosti obrazovky daného zařízení.

6.3.2 Podpora více jazyků

Tato funkcionalita nebyla přímo vyžadována, nicméně byla do aplikace implementována z následujícího důvodu. Protože administrace obsahuje spoustu statických textů, jako jsou popisky formulářových polí, menu a další, nebylo z pohledu návrhu a dalšího rozšiřování vhodné tyto texty vkládat přímo do kódu.

Proto byly texty za pomocí knihovny I18next extrahovány do separátních souborů a následně se při kompilaci propíší na správné místo v aplikaci. Tato knihovna primárně slouží k uchovávání více jazykových mutací, které



Obrázek 6.7: Ukázka responzivního zobrazení administrace [70]

se následně podle jazyku prohlížeče automaticky, případně podle preference uživatele manuálně nastavují.

V současné podobě, protože to projekt ani specifikace nevyžadují, nejsou texty přímo přeloženy do jiného jazyka a používají se pouze české. V případě nutnosti je však jejich překlad díky nastavené knihovně a separaci textů velmi snadný.

Kapitola 7

Nasazení

Aby mohli uživatelé přistupovat k administraci, je potřeba jí nahrát na veřejnou adresu. V momentě, kdy chce vývojář aplikaci veřejně nahrát spustí script, který připraví její produkční verzi.

npm run build

Tento příkaz, který je součástí základní šablony Create-React-App, zajistí, že budou všechny soubory převedeny na statické HTML, CSS a JS soubory. Ty jsou komprimovány a minifikovány, aby výsledná aplikace byla co nejmenší.

Uživatel, který navštíví odkaz, na kterém je aplikace zveřejněna si následně stáhne veškerá data k sobě. Ty se následně překreslují podle toho, jak se uživatel v aplikaci pohybuje. A protože se jedná o statická data, můžeme je nahrát na libovolný server. Pro potřeby této práce bude využita služba Firebase.

Firebase je kompletní balík služeb od společnosti Google. Kromě vlastní domény nabízí služba například i datové úložiště nebo jednoduchý způsob ověřování uživatelů. Pro tento projekt je však zajímavá pouze doména a nahrávání souborů na server.[71] Nespornou výhodou je, že firebase hosting automaticky a zdarma nastaví SSL certifikát pro HTTPS připojení.

Podle návodu [72] tak byl do projektu nainstalován a nakonfigurován modul firebase. Následně stačí spustit příkaz **firebase deploy** a nová verze aplikace se automaticky nahraje a zpřístupní na přidělené doméně.

Kapitola 8
Uživatelské testování

Před nasazením systému do produkčního prostředí je téměř nezbytným krokem jeho testování. To lze provádět několika způsoby, ať už se jedná o automatizované testy v průběhu implementace, nebo pomocí uživatelského testování po dokončení systému.

Pro tento případ bylo předem zvoleno uživatelské testování. Jak už bylo zmíněno v analýze na začátku práce, vyvíjený systém budou prozatím využívat celkem tři osoby, společně s autorem práce. Díky tomu tak máme velmi úzkou skupinu uživatelů a zainteresovaných osob, jež mají na projekt vliv.

Nabízí se tedy tuto skupinu využít i pro testování systému. Díky tomu můžeme odhalit odchylky od zadání, nefunkční části systému a další nepříjemnosti, které by mohli při nasazení do produkčního prostředí způsobovat potíže.

Uživatelé, kteří budou zároveň provádět testování jsou zasvěceni do problematiky a vědí, k čemu a proč chtějí tento interní nástroj používat. Na základě těchto skutečností tak bylo rozhodnuto, že uživatelské testování proběhne ve dvou krocích. Prvním bude systémové testování, které odhalí technické nedostatky při implementaci, následovat budou akceptační testy, při kterých dojde ke zmíněnému porovnání výsledného díla s předem definovaným zadáním.[73]

8.1 Systémové testování

Jako první probíhalo systémové testování, které mělo za cíl ověřit konkrétní cesty a jejich funkcionalitu z pohledu systému. [74]

Testování probíhalo vzdáleně a samostatně. Každý z testerů obdržel celkem pět přesně definovaných scénářů. Ty samostatně prošel a poskytl výstupy.

8.1.1 Testovací scénáře

Pro otestování vytvořené administrace bylo sepsáno celkem pět scénářů, které pokrývají primární funkcionalitu aplikace. Každý scénář obsahuje následující parametry:

- **ID scénáře**
- **Popis testu**

- **Autora scénáře**, který scénář vytvořil.
- **Testera**, který scénář provádí.
- **Vstupní podmínky** je sada informací nutná k uskutečnění scénáře. Popisuje stav, ve kterém se musí tester a jeho prostředí nacházet, než začne vykonávat scénář.
- **Testovací data** vstupy, které tester potřebuje znát. Jedná se například o přihlašovací údaje.
- **Kroky scénáře**, přičemž každý má krátký popis, definovaný očekávaný výsledek, volné pole pro skutečný výsledek, který tester zaznamenal a stav, ve kterém se krok nachází.
- **Poznámky testera**, pro zaznamenání dalších poznatků.

8.1.2 Výsledek

Od každého testera bylo obdrženo všech pět scénářů s vyplněnými daty. Všechny scénáře od každého testera jsou v příloze F.

Ze všech deseti scénářů, pět u každého testera, prošlo celkem osm. U dvou bylo zjištěno pochybení. U testování testerem A nebylo možné vypnout platformu Pingl přes nastavení systému.Při testování přidávání podniku testerem B bylo zjištěno, že není možné přidat podnik a uživatel nedostane zpětnou vazbu, proč tomu tak bylo.

Oba problémy byly indentifikovány a odstraněny.

8.2 Akceptační testy

Druhé kolo testování probíhalo pomocí akceptačních testů. Tato část neměla konkrétní scénáře a byla prováděna pomocí samostatných videohovorů. Každý z uživatelů dostal přístupy do systému a využíval jej podle svého uvážení.

Každý z uživatelů sdílel obrazovku a podával průběžný komentář ke svým aktivitám. Komentáře byly zaznamenávány pozorovatelem. Jednalo se tedy o nemoderované a vzdálené testování.

8.2.1 Výsledek

Testování pomohlo dva funkční nedostatky, které byly následně zapracovány do finálního řešení.

Zároveň při akceptačních testech finálními uživateli bylo zjištěno několik dalších požadavků, které by mohly být zapracovány do administračního rozhraní. Bude tak pravděpodobné, že se administrační nástroj bude rozrůstat s rostoucí společností a novými požadavky. Pro implementaci dalších požadavků tak bude opakován postup popsaný v celé této práci.

Kapitola 9

Závěr

V práci byl popsán a shrnut kompletní postup při tvorbě administračního rozhraní pro konkrétní společnost. Na začátku byly sesbírány požadavky od zainteresovaných osob a uživatelů systému, které byly rozděleny do kategorií. Požadavkům byla přidělena priorita a na jejím základě vydefinován rozsah implementace.

V další části byly rozebrány dva hlavní přístupy při tvorbě webových aplikací. Jedná se o takzvané Single-page aplikace a Multi-page aplikace. Oba přístupy mají sadu výhod a nevýhod v závislosti na jejich konkrétním využití. Pro konkrétní případ této práce byl na základě analýzy vybrán princip single-page aplikace.

Následovala kapitola, která měla za cíl porovnat knihovny a frameworky, které tvorbu single-page aplikací usnadňují. Během této analýzy byl kladen mimo technologickou stránku a vyspělost knihovny brán zřetel jednak na současné využívaní technologií ve společnosti, které má administrace sloužit, tak na situaci na trhu práce a další určující prvky, že vybraná technologie má slibnou budoucnost.

Po výběru vhodné technologie následoval návrh uživatelského rozhraní, který kopíroval předem sesbírané a prioritizované požadavky. Vytvořený návrh uživatelského rozhraní v podobě wireframů pak definoval rozložení jednotlivých funkcionalit v rámci administrace.

Dle navržených wireframů byla provedena implementace. V práci je popsána sada technologií, které byly společně s knihovnou React použity pro usnadnění a zkvalitnění vývoje. Byla popsána architektura výsledné aplikace.Projekt byl následně zveřejněn na konkrétní doménu a zpřístupněn tak pro testování.

Pro ověření dosažení definovaných požadavků byly provedeny uživatelské testy. Tyto testy prováděli přímo uživatelé platformy, od kterých byla sesbírána zpětná vazba na systém. Z výsledků testování se ukázalo, že projekt splnil očekávání uživatelů a až na drobné chyby fungoval podle představ, přestože se v budoucnu očekává jeho rozšiřování na základě nově vznikajících požadavků ve firmě Pingl s.r.o.

Bibliografie

[1] Jesse James Garrett. *The Elements of User Experience*. Pearson Education, 2011. ISBN: 9780321683687.

[2] Peter Landau. *What is a Stakeholder?* ProjectManager.com. 13. červ. 2017. URL: https://www.projectmanager.com/blog/what-is-a-stakeholder (cit. 22. 12. 2019).

[3] *Stakeholder Analysis*. ProductPlan. 7. dub. 2019. URL: https://www.productplan.com/glossary/stakeholder-analysis/ (cit. 19. 12. 2019).

[4] ReQtest. *Requirements Analysis – Understanding the Basics*. reqtest.com. 25. říj. 2018. URL: https://reqtest.com/requirements-blog/requirements-analysis/ (cit. 17. 12. 2019).

[5] nancydehra. *10 Essential Business Requirements Gathering Techniques*. brighthubpm.com. URL: https://www.brighthubpm.com/project-planning/60264-techniques-used-in-business-requirements-gathering/ (cit. 21. 12. 2019).

[6] Morgan Masters. *Using the Brainstorming Technique in Business Analysis*. ModernAnalyst.com. 11. říj. 2011. URL: https://www.modernanalyst.com/Resources/Articles/tabid/115/ID/2067/Using-the-Brainstorming-Technique-in-Business-Analysis.aspx (cit. 21. 12. 2019).

[7] GIGI DEVAULT. *Focus groups*. The Balance Small Business. 7. pros. 2019. URL: https://www.thebalancesmb.com/what-is-a-market-research-focus-group-2296907 (cit. 26. 12. 2019).

[8] Frank Van De Ven. *Focus group or brainstorm? Why not both?* 24. říj. 2018. URL: https://uxdesign.cc/focus-group-or-brainstorm-why-not-both-35f4f06%2097%20be908 (cit. 27. 12. 2019).

[9] Brent Johnson. *Product Prioritization: Planning Poker*. Left Travel. 6. zář. 2019. URL: https://medium.com/left-travel/product-prioritization-planning-poker-8eb3f4119516 (cit. 27. 12. 2019).

[10] *Functional and Nonfunctional Requirements: Specification and Types*. Altexsoft. 29. květ. 2018. URL: https://www.altexsoft.com/blog/business/functional-and-non-functional-requirements-specification-and-types/ (cit. 27. 12. 2019).

[11] Martin Komárek. *Business analýza a studie proveditelnosti*. ČVUT, Cactoo Software. 1. ún. 2019. URL: https://moodle.fel.cvut.cz/pluginfile.php/186172/mod_resource/content/2/ByznysAnalyza_StudieProveditelnosti_Prednaska1_LS2016_2017.pdf (cit. 28. 12. 2019).

[12] *Nonfunctional Requirements*. © Scaled Agile, Inc. 18. srp. 2018. URL: https://www.scaledagileframework.com/nonfunctional-requirements/ (cit. 27. 12. 2019).

[13] Brenda Barron. *2019's Most Surprising WordPress Statistics*. 12. pros. 2019. URL: https://www.whoishostingthis.com/compare/wordpress/stats/ (cit. 30. 12. 2019).

[14] Josh Powell a Michael Mikowski. *Single Page Web Applications: JavaScript End-to-end*. Manning Publications Co., 2014. ISBN: 9781617290756.

[15] *SPA vs MPA diagram*. 18. květ. 2020. URL: https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/choose-between-traditional-web-and-single-page-app (cit. 18. 05. 2020).

[16] Daryna Pukha. *Single-Page Apps vs Multiple-Page Web Apps: What to Choose for Web Development*. Yalantis. URL: https://yalantis.com/blog/single-page-apps-vs-multiple-page-apps/ (cit. 31. 12. 2019).

[17] Paweł. *What are Single Page Applications(SPA)?* 12. zář. 2019. URL: https://dev.to/kendyl93/what-are-single-page-applications-spa-32bh (cit. 31. 12. 2019).

[18] Anastasia Z. *What's the Difference Between Single-Page and Multi-Page Apps*. 25. červ. 2018. URL: https://rubygarage.org/blog/single-page-app-vs-multi-page-app (cit. 31. 12. 2019).

[19] *Code Splitting*. webpack.js.org. URL: https://webpack.js.org/guides/code-splitting/ (cit. 31. 12. 2019).

[20] Anastasia Kryzhanovska. *Pros and Cons of Building Single Page Applications in 2019*. Gearheart. 17. červ. 2019. URL: https://gearheart.io/blog/pros-and-cons-building-single-page-applications-2019/ (cit. 31. 12. 2019).

[21] Luke Joliat. *Do we still need JavaScript frameworks?* freecodecamp. 4. ún. 2019. URL: https://www.freecodecamp.org/news/do-we-still-need-javascript-frameworks-42576735949b/ (cit. 31. 12. 2019).

[22] *Developer Survey Results 2019*. Stack Overflow. 14. ún. 2019. URL: https://insights.stackoverflow.com/survey/2019#technology-_-web-frameworks (cit. 31. 12. 2019).

[23] *Questions tagged [angularjs]*. Stack Overflow. 31. pros. 2019. URL: https://stackoverflow.com/questions/tagged/angularjs (cit. 31. 12. 2019).

[24] Google LLC. *Angular repository on GitHub*. 31. pros. 2019. URL: https://github.com/angular/angular (cit. 31. 12. 2019).

[25] *Indeed - Angular jobs*. Indeed. 31. pros. 2019. URL: https://www.indeed.com/jobs?q=angular+developer&l= (cit. 31. 12. 2019).

[26] *LinkedIn Jobs - Angularjs in Worldwide*. LinkedIn. 30. pros. 2019. URL: https://www.linkedin.com/jobs/search/?geoId=92000000&keywords=angularjs&location=Worldwide (cit. 30. 12. 2019).

[27] Liliia Harkushko. *Angular: Best Use Cases and Reasons To Opt For This Tool*. Yalantis. 31. pros. 2019. URL: https://yalantis.com/blog/when-to-use-angular/ (cit. 31. 12. 2019).

[28] Eliran Eliassy. *All you need to know about Ivy, The new Angular engine!* 10. červ. 2019. URL: https://medium.com/angular-in-depth/all-you-need-to-know-about-ivy-the-new-angular-engine-9cde471f42cf (cit. 31. 12. 2019).

[29] Shaumik Daityari. *Angular vs React vs Vue: Which Framework to Choose in 2020*. 11. pros. 2019. URL: https://www.codeinwp.com/blog/angular-vs-vue-vs-react/ (cit. 31. 12. 2019).

[30] Jens Neuhaus. *Angular vs. React vs. Vue: A 2017 comparison*. 28. srp. 2017. URL: https://medium.com/unicorn-supplies/angular-vs-react-vs-vue-a-2017-comparison-c5c52d620176 (cit. 31. 12. 2019).

[31] Inc. Facebook. *React repository on GitHub*. URL: https://github.com/facebook/react (cit. 31. 12. 2019).

[32] *Questions tagged [reactjs]*. Stack Overflow. 31. pros. 2019. URL: https://stackoverflow.com/questions/tagged/reactjs (cit. 31. 12. 2019).

[33] *Indeed - React jobs*. Indeed. 31. pros. 2019. URL: https://www.indeed.com/jobs?q=react+developer&l= (cit. 31. 12. 2019).

[34] *LinkedIn Jobs - React.js in Worldwide*. LinkedIn. 30. pros. 2019. URL: https://www.linkedin.com/jobs/search/?geoId=92000000&keywords=react.js&location=Worldwide (cit. 30. 12. 2019).

[35] *Who uses React?* URL: https://stackshare.io/react (cit. 31. 12. 2019).

[36] *Virtual DOM and Internals*. Facebook Inc. 30. pros. 2019. URL: https://reactjs.org/docs/faq-internals.html (cit. 30. 12. 2019).

[37] *React: The Virtual DOM*. Codecademy. URL: https://www.codecademy.com/articles/react-virtual-dom (cit. 30. 12. 2019).

[38] Shubham Kanodia. *react@16.12.0*. URL: https://bundlephobia.com/result?p=react@16.12.0 (cit. 31. 12. 2019).

[39] Shubham Kanodia. *react-dom@16.12.0*. URL: https://bundlephobia.com/result?p=react-dom@16.12.0 (cit. 31. 12. 2019).

[40] Aleksandra Dikusar. *The Price of Popularity. Strong and Weak Sides of ReactJS and React Native*. XB Software Ltd. 16. led. 2019. URL: https://xbsoftware.com/blog/price-of-popularity-pros-and-cons-of-reactjs-and-react-native/ (cit. 30. 12. 2019).

[41] *Introducing JSX*. Facebook Inc. 30. pros. 2019. URL: https://reactjs.org/docs/introducing-jsx.html (cit. 30. 12. 2019).

[42] Yuxi (Evan) You. *Vue repository on GitHub*. URL: https://github.com/vuejs/vue (cit. 31. 12. 2019).

[43] *Questions tagged [vue.js]*. Stack Overflow. URL: https://stackoverflow.com/questions/tagged/vue.js (cit. 31. 12. 2019).

[44] *Indeed - Vue js jobs*. Indeed. 31. pros. 2019. URL: https://www.indeed.com/jobs?q=vue+js&l (cit. 31. 12. 2019).

[45] *LinkedIn Jobs - Vue.js in Worldwide*. LinkedIn. 30. pros. 2019. URL: https://www.linkedin.com/jobs/search/?geoId=92000000&keywords=vue.js&location=Worldwide (cit. 30. 12. 2019).

[46] Michał Sajnóg. *13 Top Companies That Have Trusted Vue.js – Examples of Applications*. 21. břez. 2018. URL: https://www.netguru.com/blog/13-top-companies-that-have-trusted-vue.js-examples-of-applications (cit. 30. 12. 2019).

[47] Maja Nowak. *Vue vs. React in 2019: Which Framework to Choose and When*. 8. břez. 2018. URL: https://www.monterail.com/blog/vue-vs-react-2019 (cit. 30. 12. 2019).

[48] Liliia Harkushko. *Single-Page Apps vs Multiple-Page Web Apps: What to Choose for Web Development*. Yalantis. URL: https://yalantis.com/blog/javascript-frameworks-how-to-make-your-choice/ (cit. 31. 12. 2019).

[49] *State of Vue.js 2019*. Monterail. 1. ún. 2019. URL: http://bit.ly/2rSncs4 (cit. 31. 12. 2019).

[50] Shubham Kanodia. *vue@2.6.11*. URL: https://bundlephobia.com/result?p=vue@2.6.11 (cit. 31. 12. 2019).

[51] *Visual Studio Code - Free. Built on open source. Runs everywhere.* Microsoft. 20. břez. 2020. URL: https://code.visualstudio.com/ (cit. 20. 03. 2020).

[52] *ESLint - Find and fix problems in your JavaScript code*. https://openjsf.org/. 20. břez. 2020. URL: https://eslint.org/ (cit. 20. 03. 2020).

[53] *stylelint - A mighty, modern linter that helps you avoid errors and enforce conventions in your styles*. https://openjsf.org/. 20. břez. 2020. URL: https://stylelint.io/ (cit. 20. 03. 2020).

[54] *Prettier*. 20. břez. 2020. URL: https://prettier.io/ (cit. 20. 03. 2020).

[55] *Create a New React App*. Facebook Inc. 22. břez. 2020. URL: https://reactjs.org/docs/create-a-new-react-app.html (cit. 22. 03. 2020).

[56] *Introducing JSX*. Facebook Inc. 22. břez. 2020. URL: https://reactjs.org/docs/introducing-jsx.html (cit. 22. 03. 2020).

[57] *TypeScript is Typed JavaScript at Any Scale*. Microsoft. 22. břez. 2020. URL: https://www.typescriptlang.org/index.html (cit. 22. 03. 2020).

[58] *TypeScript - JSX*. Microsoft. 16. květ. 2020. URL: https://www.typescriptlang.org/docs/handbook/jsx.html#introduction (cit. 17. 05. 2020).

[59] *Create React App - Folder Structure*. Facebook, Inc. 20. dub. 2020. URL: https://create-react-app.dev/docs/folder-structure/ (cit. 20. 04. 2020).

[60] *Redux FAQ: Code Structure*. Dan Abramov a the Redux documentation authors. 20. dub. 2020. URL: https://redux.js.org/faq/code-structure (cit. 20. 04. 2020).

[61] *Redux FAQ: Immutable Data*. Dan Abramov a the Redux documentation authors. 20. dub. 2020. URL: https://redux.js.org/faq/immutable-data (cit. 20. 04. 2020).

[62] *Connect react with redux, redux-saga - diagram*. 18. květ. 2020. URL: https://medium.com/@amitha4g/connect-react-with-redux-redux-saga-41f472b5f6d7 (cit. 18. 05. 2020).

[63] *Redux-thunk repository on GitHub*. 22. dub. 2020. URL: https://github.com/reduxjs/redux-thunk (cit. 22. 04. 2020).

[64] *GraphQL*. The GraphQL Foundation. 22. dub. 2020. URL: https://graphql.org/ (cit. 22. 04. 2020).

[65] *GraphQL - Queries and Mutations*. The GraphQL Foundation. 22. dub. 2020. URL: https://graphql.org/learn/queries/ (cit. 22. 04. 2020).

[66] *GraphQL - Queries and Mutations*. The GraphQL Foundation. 22. dub. 2020. URL: https://graphql.org/learn/queries/#mutations (cit. 22. 04. 2020).

[67] *The Apollo Data Graph Platform*. Meteor Development Group Inc. 22. dub. 2020. URL: https://www.apollographql.com/ (cit. 22. 04. 2020).

[68] Michael Jackson and Ryan Florence. *React router - Quick Start*. 1. květ. 2020. URL: https://reacttraining.com/react-router/web/guides/quick-start (cit. 01. 05. 2020).

[69] *React router - Quick Start*. XTech. 4. květ. 2020. URL: https://ant.design/ (cit. 04. 05. 2020).

[70] *Multi Device Website Mockup Generator - zdroj*. 18. květ. 2020. URL: http://techsini.com/multi-mockup/index.php (cit. 18. 05. 2020).

[71] *Firebase helps mobile and web app teams succeed*. Google. 7. květ. 2020. URL: https://firebase.google.com/ (cit. 07. 05. 2020).

[72] *Firebase - Add Firebase SDKs and initialize Firebase*. Google. 7. květ. 2020. URL: https://firebase.google.com/docs/web/setup#node.js-apps (cit. 07. 05. 2020).

[73] Tomáš Hlava. *Fáze a úrovně provádění testů*. 21. srp. 2011. URL: http://testovanisoftwaru.cz/metodika-testovani/druhy-typy-a-kategorie-testu/faze-testu/ (cit. 12. 05. 2020).

[74] *The best usability testing methods for your projects*. Hotjar Ltd. 14. led. 2020. URL: https://www.hotjar.com/usability-testing/methods/ (cit. 12. 05. 2020).

Nomenklatura

Tato příloha vysvětluje zkratky a pojmy užité v práci. Pojmy, které jsou v textu dále rozvedeny a vysvětleny v rámci práce, stejně jako přípony souborů, zde vysvětleny nejsou.

UI Z anglického User interface

API Z anglického Application programming interface

MPA Z anglického Multi-page Application

SPA Z anglického Single-page Application

AJAX Z anglického Asynchronous JavaScript and XML

DOM Z anglického Document Object Model

JSON Z anglického JavaScript Object Notation

UX Z anglického User Experience

JS JavaScript

HTML Z anglického Hypertext Markup Language

CSS Z anglického Cascading Style Sheets

URL Z anglického Uniform Resource Locator

MVC Z anglického Model–view–controller

JSX JavaScript XML

CMS Z anglického Content management system

SSL Z anglického Secure Sockets Layer

HTTPS Z anglického Hypertext Transfer Protocol Secure

CLI Z anglického Command Line Interface

MongoDB Multiplatformní dokumentová databáze, která využívá dokumenty podobné formátu JSON.

NoSQL Non-Structured Query Language.

React Javascriptová knihovna sloužící pro tvorbu uživatelských rozhraní.

Kotlin Staticky typovaný programovací jazyk běžící nad JVM.

Node Node.js je otevřené, mlutiplatformní javascriptové prostředí vykonávající javascriptový kód mimo webový prohlížeč.

GraphQL Otevřený jazyk pro manipluaci a dotazování nad daty určený pro API.

Compass Grafické rozhraní pro MongoDB.

Webpack Javascriptový modulátor.

Express Webový framework pro Node.

Gatsby Moderní generátor statických stránek pro React.

Next Webový framework pro aplikace podporující Server-side-rendering.

I18next Knihovna pro podporu více jazyků pro javascriptové aplikace

Příloha A

Přihlašovací údaje k administraci

Projekt běží na veřejné doméně. Je napojený na testovací databázi. Pro přístup využijte údajů níže.

Odkaz

- `https://superadmin-87b9e.web.app/`

Přihlašovací údaje

- **Přihlašovací jméno:** cvutdemo@pingl.app
- **Heslo:** CtuPass753

Příloha B

Požadavky

V příloze se nachází list požadavků společně s jejich prioritizací.

Priotity [1]				Funkce								
A	V	F		ID	Název / popis							
					Podniky							**UC**
					Poptávky o založení [2]							
4	4	3	**11**	FR_0001		Zobrazit všechny poptávky						
4	4	3	**11**	FR_0002		Zobrazit detial poptávky						
4	1	2	**7**	FR_0003		Automaticky vytvořit nový podnik						
3	1	2	**6**	FR_0004		Doplnit chybějící informace přímo do poptávky						
2	2	4	**8**	FR_0005		Získat kontaktní údaje na osobu, která poptávku vyplnila						
					Seznam podniků [3]							
						Záznam v tabulce						
3	5	5	**13**	FR_0006			Základní info (jméno, obrázek)					
2	2	1	**5**	FR_0007			Tajné info - poplatek, bankovní účet,...					
5	5	4	**14**	FR_0008			Status podniku - otevřeno, zavřeno, schováno z veřejného listu					1
						Možnost seznam filtrovat						
4	3	1	**8**	FR_0009			Podle statusů					
4	3	1	**8**	FR_0010			Podle typů objednávek					
2	3	1	**6**	FR_0011		Vyhledávání v seznamu podniků						
					Detail podniku [4]							
2	3	5	**10**	FR_0012		Základní informace (jméno, poplatky, adresa, ..)						
1	3	5	**9**	FR_0013		Úprava základních informací						
3	5	4	**12**	FR_0014		Kontaktní informace						
3	5	4	**12**	FR_0015		Úprava kontaktních informací						
5	5	4	**14**	FR_0016		Tajné informace a konfikurace						2
5	5	4	**14**	FR_0017		Úprava tajných informací a konfigurace						2
2	4	1	**7**	FR_0018		Historie aktivit (vypnuto, zapnuto, přihlášení, ..) [5]						
3	1	2	**6**	FR_0019		Zobrazit a tisknout smlouvy s podnikem [6]						
0	3	3	**6**	FR_0020		Ukázat všechny objednávky v podniku						
0	3	3	**6**	FR_0021			ve vybraný časový úsek					
5	5	5	**15**	FR_0022		Možnost nastavit poplatky						3
3	3	2	**8**	FR_0023		Informace o chybách v menu (chybí cena za obaly, možnost obalů)						
					Vytvořit nový podnik							
5	5	5	**15**	FR_0024		Vytvořit podnik s povinnými informacemi (viz. DB)						4
5	4	3	**12**	FR_0025		Automaticky přidat účty pro administrátora						
4	1	2	**7**	FR_0026		Poslat automaticky uvítací email						
					Uživatelé							
					Tvorba uživatelů							
3	5	4	**12**	FR_0027		Možnost vytvořit účty pro admina [7]						
3	5	4	**12**	FR_0028		Možnost vytvořit účty pro obsluhu						
3	4	2	**9**	FR_0029		Poslat automaticky odkaz na reset hesla						
					Seznam uživatelů							
2	2	3	**7**	FR_0030		Ukázat všechny uživatele						

Priotity [1]				Funkce								
A	V	F		ID	Název / popis							
2	3	2	**7**	FR_0031		Filtrovat seznam podle emailů, id						
2	3	1	**6**	FR_0032		Fulltext vyhledávání uživatelů						
					Detail uživatele							
2	3	2	**7**	FR_0033		Ukázat objednávky uživatele						
2	3	1	**6**	FR_0034		Ukázat poslední objednávky, jejich cenu						
2	2	3	**7**	FR_0035		Zobrazit aktivitu uživatele						
2	2	1	**5**	FR_0036		Odstranit uživatelský účet						
2	1	2	**5**	FR_0037		Zobrazit informace o uživateli						
					Pingl system							
5	5	4	**14**	FR_0038		On / Off Pingl						5
5	5	5	**15**	FR_0039		Tags in marketplace						6
					Prints							
					QR generator							
1	3	3	**7**	FR_0040		Zobrazit současný design QR kódu						
5	5	3	**13**	FR_0041		Vybrat restauraci pro generování kódů						
5	5	3	**13**	FR_0042		Vybrat stoly a počty QR kódů						
					Ostatní materiály							
3	1	1	**5**	FR_0043		Promo materiály						
3	1	1	**5**	FR_0044		Konfigurace promo materiálů (loga, ...)						
					Statistiky							
					Uživatelé							
3	3	3	**9**	FR_0045		Zobrazit typy registrace						
3	4	4	**11**	FR_0046		Zobrazit typy noticikací, které má uživatel povolené						
4	2	3	**9**	FR_0047		Zobrazit zemi, ze které uživatel přišel						
3	2	4	**9**	FR_0048		Zobrazit přírůstek uživatelů v čase [8]						
3	1	2	**6**	FR_0049		Zobrazit uživatele podle restaurací [9]						
0	1	4	**5**	FR_0050		Zobrazit konverzi uživatelů - kolik z nich nedokončilo objednávku						
0	1	4	**5**	FR_0051		Zobrazit akviziční kanály - odkud uživatelé chodí						
					Orders and items							
5	5	3	**13**	FR_0052		Zobrazit platební metody a jejich využití						
5	5	3	**13**	FR_0053		Filtrovat objednávky podle restaurace						
5	4	4	**13**	FR_0054		Filtorvat objednávky podle času						
						Grafy (po aplikování filtrů)						
5	4	4	**13**	FR_0056			Počet a cena objednávek za čas					
2	3	2	**7**	FR_0055			Doba čekání na objednávku					
					Podniky							
4	2	1	**7**	FR_0057		Zobrazit přírůstek podniků v čase						
2	1	1	**4**	FR_0058		Zobrazit resaurace podle typu Pingla						

Priotity [1]				Funkce								
A	V	F		ID	Název / popis							
					Platby							
4	2	1	**7**	FR_0059		Procesování plateb u uživatele [10]						
					Převody peněz							
1	2	2	**5**	FR_0060		Zobrazit informace o ziscích						
1	3	2	**6**	FR_0061		Nastavit, jaký časový rámec zobrazit						
					Shrnutí objednávek							
2	1	2	**5**	FR_0062		Generovat CSV pro podnik o prodejích, poplatcích, ..						
							DŮLEŽITÉ	22				
							MÉNĚ DŮLEŽITÍ	10				
							NEDŮLEŽITÉ	30				
							celkem	62				

Příloha C

Případy užití

Use Case 1 - Přidání podniku

Id	UC_001
Název	Přidání nového podniku do systému
Aktéři:	Administrátor
Předpoklady:	Administrátor je přihlášený do systému
Hlavní flow:	1. Případ užití začíná, když administrátor klikne na tlačítko 'přidat nový podnik' 2. Systém zobrazí stránku s formulářem 3. Administrátor vyplní všechny povinné informace a dodatečné informace, které má k dispozici 4. Systém před odesláním označí chybná nebo nevyplněná pole 5. Administrátor klikne na tlačítko 'Přidat podnik', kterým odešle požadavek na vyvtoení podniku 6. Pokud je vše vyplněné správně, systém vypíše potvrzující hlášku a případ užití končí 7. Pokud některé informace chybí, uživatel je vyzván k jejich doplnění 8. Pokud nelze podnik do systému přidat, systém vypíše chybovou hlášku a případ užití tím končí

Use Case 2 - Zobrazení viditelnosti podniků

ID	UC_002
Název	Zobrazení viditelnosti podniků
Aktéři	Administrátor
Předpoklady	Administrátor je přihlášený do systému
Hlavní flow	1. Případ užití začíná, když administrátor klikne na záložku 'seznam podniků' 2. Systém zobrazí list všech podniků v systému, společně s názvem podniku a jeho aktivním statusem 3. Administrátor podle sloupečku 'Viditelné v seznamu'zkontroluje indikátory značící viditelnost v podniku a případ užití tím končí

Use Case 3 - Úprava konfigurace podniku

ID	UC_003
Název	Úprava konfigurace podniku
Aktéři	Administrátor
Předpoklady	Administrátor je přihlášený do systému
Hlavní flow	1. Případ užití začíná, když administrátor klikne na záložku 'seznam podniků' 2. Systém zobrazí seznam všech podniků 3. Administrátor kliknutím vybere podnik, který chce upravovat 4. Systém zobrazí detailní informace o podniku 5. Administrátor klikne na tlačítko 'Upravit informace' 6. Systém zaktivní pole, která lze upravovat 7. Administrátor upraví požadované informace a klikne na tlačítko 'Aktializovat' 8. Pokud je vše vyplněné správně, systém vypíše potvrzující hlášku a případ užití končí 9. Pokud některé informace chybí, nebo jsou chybně zadány, administrátor je vyzván k jejich doplnění 10. Pokud nelze podnik upravit, systém vypíše odpovídající chybovou hlášku a případ užití tím končí

Use Case 4 - Nastavení poplatku daného podniku

ID	UC_004
Název	Nastavení poplatku daného podniku
Aktéři	Administrátor
Předpoklady	Administrátor je přihlášený do systému
Hlavní flow	1. Případ užití začíná, když administrátor zvolí podnik, který chce upravit 2. Systém zobrazí detailní informace o podiku 3. Administrátor klikne na tlačítko 'změna poplatku' 4. Systém zobrazí okno pro nastavení poplatku společně s popisem typu poplatku, take-away nebo od stolu 5. Admnistrátor vyplní poplatek a potvrdí změny 6. Pokud je vše zadáno v pořádku a bez chyb, systém vypíše potvrzující hlášku a případ užití končí 7. Pokud některé informace chybí, nebo jsou chybně zadány, administrátor je vyzván k jejich doplnění 8. Pokud nelze částku zadat, systém vypíše odpovídající chybovou hlášku a případ užití tím končí

Use Case 5 - Zapnutí a vypnutí celého systému Pingl

ID	UC_005
Název	Zapnutí a vypnutí celého systému Pingl
Aktéři	Administrátor
Předpoklady	Administrátor je přihlášený do systému
Hlavní flow	1. Případ užití začíná, když administrátor klikne na záložku "Nastavení systému" 2. Systém zobrazí stránku, na které lze nastavovat globální parametry celé platformy Pingl, včetně přepínače na vypnutí a zapnutí 3. Administrátor klikne na přepínač 4. Systém zobrazí potvrzující dialogové okno se čtyřmístným kódem 5. Administrátor zadá příslušný kód a volbu potvrdí 6. Systém vypne možnost online objednávek přes Pingla ve všech pondicích 7. Administrátor klikne na záložku "Seznam podniků" a zkontroluje, že je online objednávání vypnuto ve všechn podnicích a případ užití končí

Use Case 6 - Přdání tagu do seznamu tagů

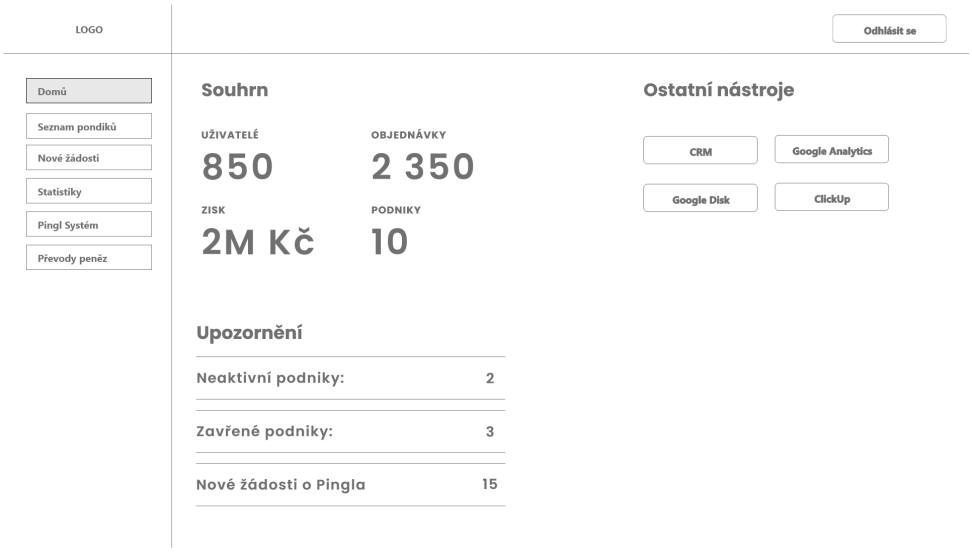
ID	UC_006
Název	Přdání tagu do seznamu tagů
Aktéři	Administrátor
Předpoklady	Administrátor je přihlášený do systému
Hlavní flow	9. Případ užití začíná, když administrátor klikne na záložku “Nastavení systému” 10. Systém zobrazí stránku, na které lze nastavovat globální parametry celé platformy Pingl, včetně seznamu všech tagů, které se přidávají k podnikům 11. Administrátor klikne na ‘Přidat tag’ 12. Systém zobrazí formulář 13. Administrátor napíše název tagu, jeho překlad a potvrdí odeslání 14. Pokud je vše zadáno v pořádku a bez chyb, tag se zařadí do seznamu, systém vypíše potvrzující hlášku a případ užití končí 15. Pokud některé informace chybí, nebo jsou chybně zadány, administrátor je vyzván k jejich doplnění 16. Pokud nelze tag přídatt, systém vypíše odpovídající chybovou hlášku a případ užití tím končí

Příloha D

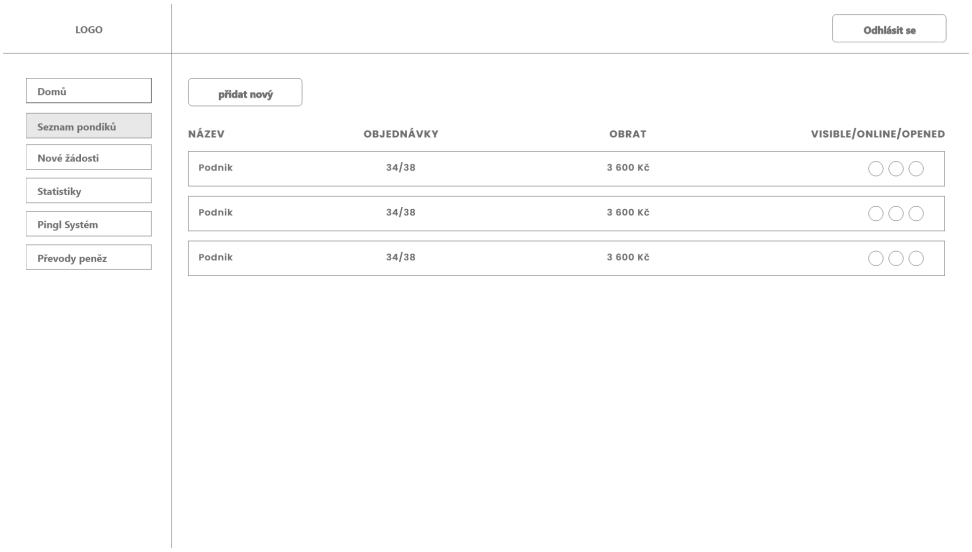
Wireframy

Přihlášení

Domovská stránka

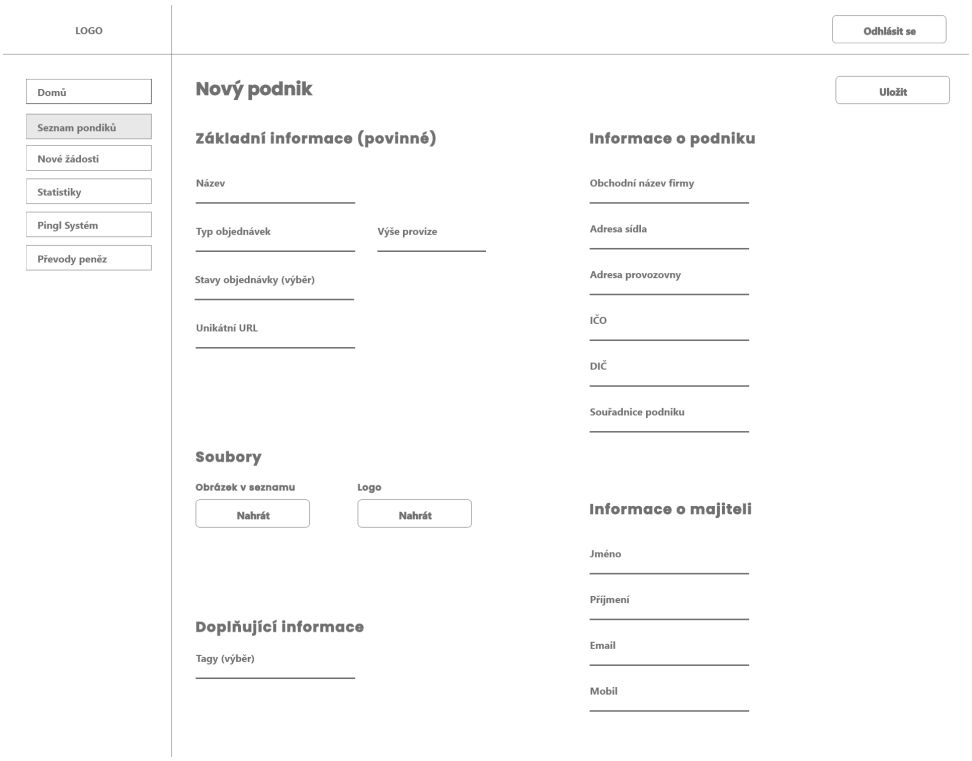


Seznam podniků

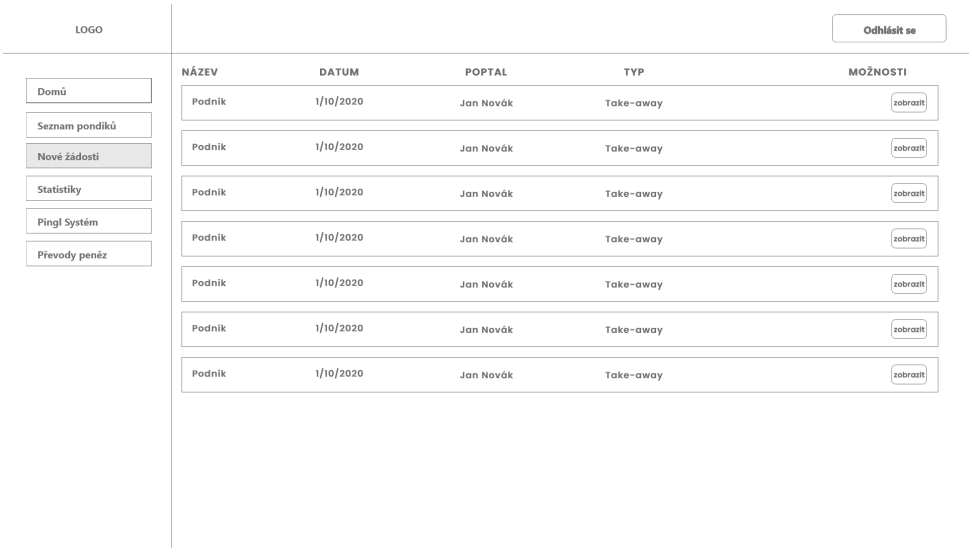


Detail podniků

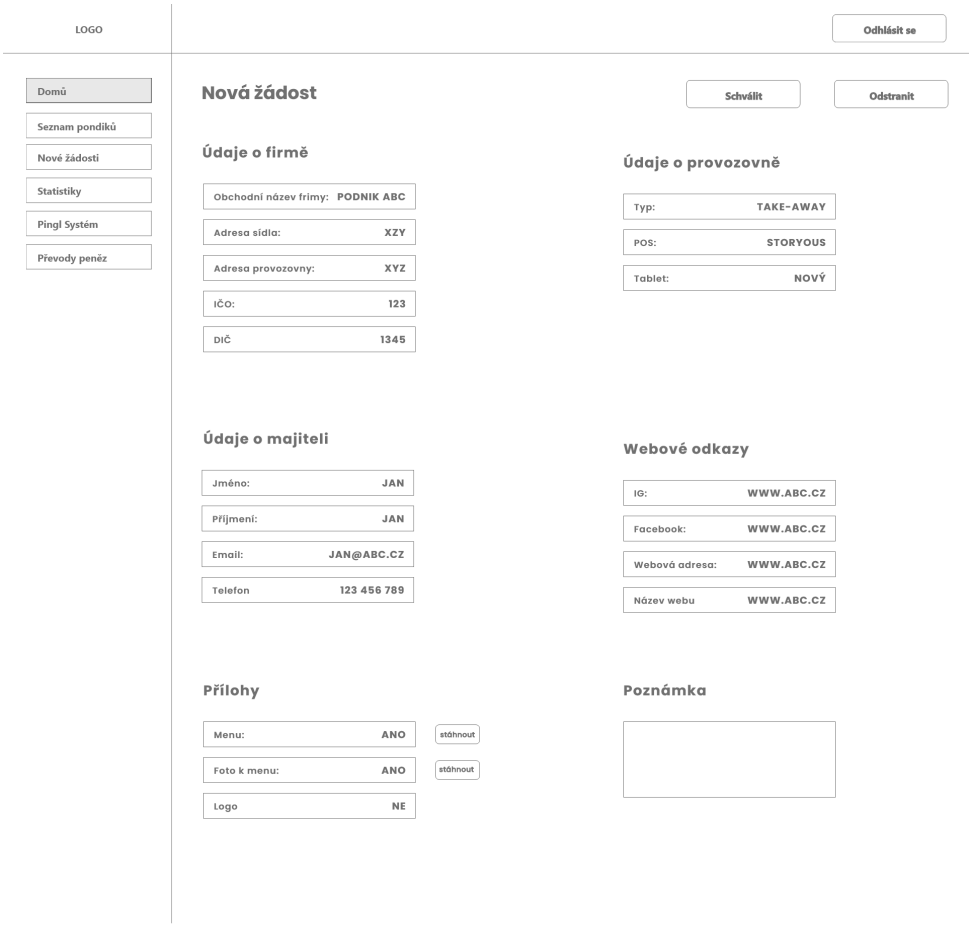
Nový podnik



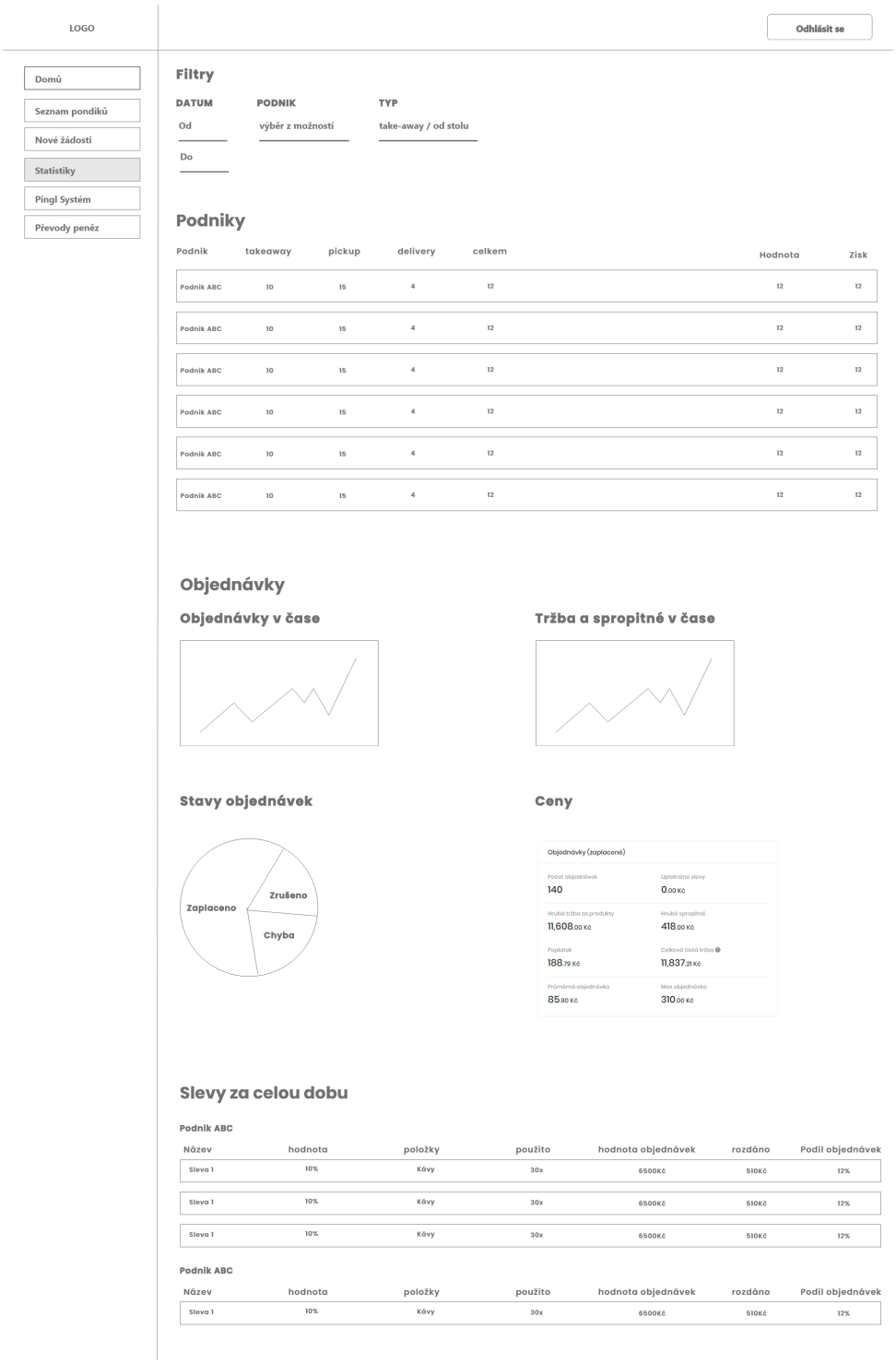
Nové žádosti



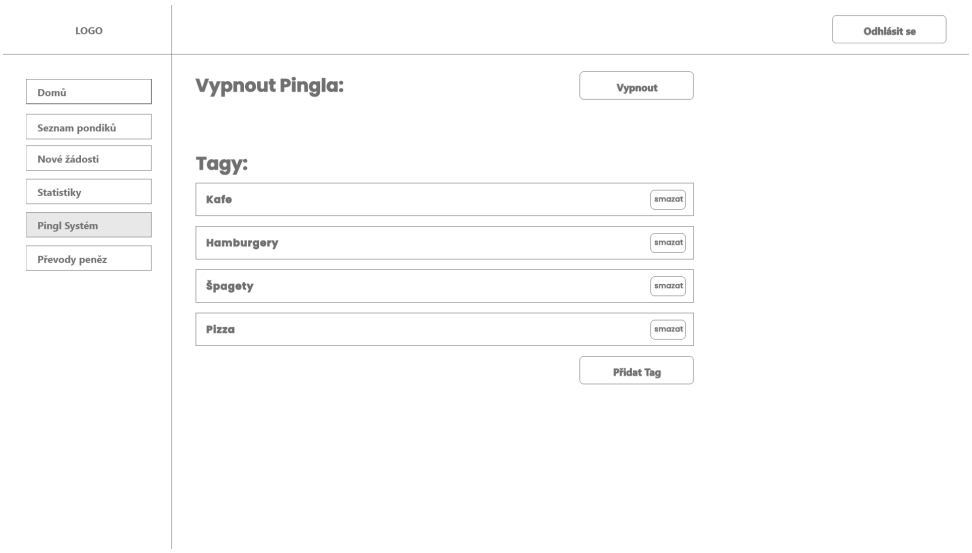
Detail žádosti



Statistiky


LOGO
Odhlásit se
Domů
Seznam pondiků
Nové žádosti
Statistiky
Pingl Systém
Převody peněz
Filtry
DATUM
Od
Do
PODNIK
výběr z možností
TYP
take-away / od stolu
Podniky
Podnik
takeaway
pickup
delivery
celkem
Hodnota
Zisk
Podnik ABC 10 15 4 12 12 12
Podnik ABC 10 15 4 12 12 12
Podnik ABC 10 15 4 12 12 12
Podnik ABC 10 15 4 12 12 12
Podnik ABC 10 15 4 12 12 12
Podnik ABC 10 15 4 12 12 12
Objednávky
Objednávky v čase
Tržba a spropitné v čase
Stavy objednávek
Zaplaceno
Zrušeno
Chyba
Ceny
Objednávky (zaplacené)
Počet objednávek
140
Uplatněné slevy
0.00 Kč
Hrubá tržba za produkty
11,608.00 Kč
Hrubé spropitné
418.00 Kč
Poplatek
188.79 Kč
Celková čistá tržba
11,837.21 Kč
Průměrná objednávka
85.90 Kč
Max objednávka
310.00 Kč
Slevy za celou dobu
Podnik ABC
Název
hodnota
položky
použito
hodnota objednávek
rozdáno
Podíl objednávek
Sleva 1 10% Kávy 30x 6500Kč 510Kč 12%
Sleva 1 10% Kávy 30x 6500Kč 510Kč 12%
Sleva 1 10% Kávy 30x 6500Kč 510Kč 12%
Podnik ABC
Název
hodnota
položky
použito
hodnota objednávek
rozdáno
Podíl objednávek
Sleva 1 10% Kávy 30x 6500Kč 510Kč 12%

Pingl Systém



Příloha E

Testovací scénáře

E.1 Tester A

Testovací scénáře a výsledky testování od prvního testera.

TS 1 - Přihlášení

ID scénáře	TS_001	Popis testu	Přihlášení do admonistračního rozhraní a odhlášení		
Autor	Filip Štěrba	Obdržel	Vojta	Verze	1.00

#	Vstupní podmínky	#	Testovací data	
1	Přístup k internetu	1	Jméno	cvutdemo@pingl.app
2	Prohlížeč Chrome, Mozilla, Firefox nebo Safari	2	Heslo	CtuPass753
		3	URL:	https://superadmin-87b9e.web.app

Krok	Popis	Očekávaný výsledek	Skutečný výsledek	Stav
1	Otevřete odkaz k administraci	Zobrazí se přihlašovací okno		Prošel
2	Zadejte jméno a heslo	Vyplňuje se pole		Prošel
3	Klikněte na přihlásit se	Zobrazení hlavní stránky s daty		Prošel
4	Klikněte na odhlásit se (pravý horní roh)	Zobrazí se přihlašovací okno		Prošel

Poznámky testera	/

TS 2 - Vytvoření podniku

ID scénáře	TS_002	Popis testu	Vytvoření nového podniku v administraci		
Autor	Filip Štěrba	Obdržel	Vojta	Verze	1.00

#	Vstupní podmínky	#	Testovací data	
1	Přístup k internetu	1	Jméno	cvutdemo@pingl.app
2	Prohlížeč Chrome, Mozilla, Firefox nebo Safari	2	Heslo	CtuPass753
3	Přihlášení uživatel	3	URL:	https://superadmin-87b9e.web.app/

Krok	Popis	Očekávaný výsledek	Skutečný výsledek	Stav
1	Přihlaště se do administrace	Zobrazí se úvodní obrazovka		Prošel
2	Kliknte v menu na položku Podniky	Zobrazí se senzam podniků		Prošel
3	Klikněte na talčítko Přidat podnik	Zobrazí se formulář		Prošel
4	Vyplňte povinné údaje	/		Prošel
5	Odešlete formulář	Zobrazí se potvrzení a formulář na		Prošel

Poznámky testera	Přidat tlačítka předchozí a další pro kroky formuláře, dát zpětnou vazbu o chybě

TS 3 - Vypnutí systému

ID scénáře	TS_003	Popis testu	Vypnutí celého systému objednávek pro všechny podniky		
Autor	Filip Štěrba	Obdržel	Vojta	Verze	1.00

#	Vstupní podmínky	#	Testovací data	
1	Přístup k internetu	1	Jméno	cvutdemo@pingl.app
2	Prohlížeč Chrome, Mozilla, Firefox nebo Safari	2	Heslo	CtuPass753
3	Přihlášený uživatel	3	URL:	https://superadmin-87b9e.web.app/
		4	Heslo k vypnutí	C7dk38chsk6jebvs7s

Krok	Popis	Očekávaný výsledek	Skutečný výsledek	Stav
1	Přihlaště se do administrace	Zobrazí se úvodní obrazovka		Prošel
2	Kliknte v menu na položku Systém	Zobrazí se obrazovka s nastavením		Prošel
3	Klikněte na tlačítko Vypnout	Zobrazí se okno s heslem		Prošel
4	Zadejte heslo pro vypnutí	Zobrazí se hláška, že systém byl vypnu	Hláška "Only chosen one can touch	Neprošel

Poznámky tester	Nelze vypnout všechny podniky pomocí postupu a zadaného hesla - výsledkem je error.

TS 4 - Zobrazení statistik

ID scénáře	TS_004	Popis testu	Zobrazení statistik dvou volitelných podniků za poslední týden		
Autor	Filip Štěrba	Obdržel	Vojta	Verze	1.00

#	Vstupní podmínky	#	Testovací data	
1	Přístup k internetu	1	Jméno	cvutdemo@pingl.app
2	Prohlížeč Chrome, Mozilla, Firefox nebo Safari	2	Heslo	CtuPass753
3	Přihlášený uživatel	3	URL:	https://superadmin-87b9e.web.app

Krok	Popis	Očekávaný výsledek	Skutečný výsledek	Stav
1	Přihlaště se do administrace	Zobrazí se úvodní obrazovka		Prošel
2	Kliknte v menu na položku Statistiky	Zobrazí se obrazovka se statistikami		Prošel
3	V horním panelu vyberte:			Prošel
3-a	- libovolné 2 podniky			
3-b	- období za poslední týden			
4	Klikněte na tlačítko Filtrovat	Statistiky se změní		Prošel

Poznámky tester	

TS 5 - Upravení lokace

ID scénáře	TS_005	Popis testu	Upravení lokace existujícího podniku		
Autor	Filip Štěrba	Obdržel	Vojta	Verze	1.00

#	Vstupní podmínky	#	Testovací data	
1	Přístup k internetu	1	Jméno	cvutdemo@pingl.app
2	Prohlížeč Chrome, Mozilla, Firefox nebo Safari	2	Heslo	CtuPass753
3	Přihlášený uživatel	3	URL:	https://superadmin-87b9e.web.app

Krok	Popis	Očekávaný výsledek	Skutečný výsledek	Stav
1	Přihlaště se do administrace	Zobrazí se úvodní obrazovka		Prošel
2	V menu zvolte položku Podniky	Zobrazí se seznam podniků		Prošel
3	Vyberte podnik Yumeee	Zobrazí se detail podniku		Prošel
4	Klikněte na záložku Informace	Zobrazí se výběr informací o podniku		Prošel
5	Klikněte na tlačítko Zvolit na mapě	Zobrazí se mapa s lokací podniku		Prošel
6	Posuňte mapu na jiné místo a potvrdt	Souřadnice se změní		Prošel

Poznámky tester	

E.2 Tester B

Testovací scénáře a výsledky testování od druhého testera.

TS 1 - Přihlášení

ID scénáře	TS_001	Popis testu	Přihlášení do admonistračního rozhraní a odhlášení		
Autor	Filip Štěrba	Obdržel	Alex	Verze	1.00

#	Vstupní podmínky
1	Přístup k internetu
2	Prohlížeč Chrome, Mozilla, Firefox nebo Safari

#	Testovací data	
1	Jméno	cvutdemo@pingl.app
2	Heslo	CtuPass753
3	URL:	https://superadmin-87b9e.web.ap

Krok	Popis	Očekávaný výsledek	Skutečný výsledek	Stav
1	Otevřete odkaz k administraci	Zobrazí se přihlašovací okno		Prošel
2	Zadejte jméno a heslo	Vyplňuje se pole		Prošel
3	Klikněte na přihlásit se	Zobrazení hlavní stránky s daty		Prošel
4	Klikněte na odhlásit se (pravý horní roh)	Zobrazí se přihlašovací okno		Prošel

Poznámky testera	V tabulce testování dát heslo a přihlašovací jméno do zvláštního sloupečku, aby se dalo pohodlně kopírovat.

TS 2 - Vytvoření podniku

ID scénáře	TS_002	Popis testu	Vytvoření nového podniku v administraci		
Autor	Filip Štěrba	Obdržel	Alex	Verze	1.00

#	Vstupní podmínky
1	Přístup k internetu
2	Prohlížeč Chrome, Mozilla, Firefox nebo Safari
3	Přihlášení uživatel

#	Testovací data	
1	Jméno	cvutdemo@pingl.app
2	Heslo	CtuPass753
3	URL:	https://superadmin-87b9e.web.ap

Krok	Popis	Očekávaný výsledek	Skutečný výsledek	Stav
1	Přihlaště se do administrace	Zobrazí se úvodní obrazovka		Prošel
2	Kliknte v menu na položku Podniky	Zobrazí se senzam podniků		Prošel
3	Klikněte na talčítko Přidat podnik	Zobrazí se formulář		Prošel
4	Vyplňte povinné údaje	/		Prošel
5	Odešlete formulář	Zobrazí se potvrzení a formulář na		Neprošel

Poznámky tester	Při odeslání formuláře se nic nestalo.

TS 3 - Vypnutí systému

ID scénáře	TS_003	Popis testu	Vypnutí celého systému objednávek pro všechny podniky		
Autor	Filip Štěrba	Obdržel	Alex	Verze	1.00

#	Vstupní podmínky
1	Přístup k internetu
2	Prohlížeč Chrome, Mozilla, Firefox nebo Safari
3	Přihlášený uživatel

#	Testovací data	
1	Jméno	cvutdemo@pingl.app
2	Heslo	CtuPass753
3	URL:	https://superadmin-87b9e.web.app/
4	Heslo k vypnutí	C7dk38chsk6jebvs7s

Krok	Popis	Očekávaný výsledek	Skutečný výsledek	Stav
1	Přihlaště se do administrace	Zobrazí se úvodní obrazovka		Prošel
2	Kliknte v menu na položku Systém	Zobrazí se obrazovka s nastavením		Prošel
3	Klikněte na tlačítko Vypnout	Zobrazí se okno s heslem		Prošel
4	Zadejte heslo pro vypnutí	Zobrazí se hláška, že systém byl vypnut		Prošel

Poznámky tester	/

TS 4 - Zobrazení statistik

ID scénáře	TS_004	Popis testu	Zobrazení statistik dvou volitelných podniků za poslední týden		
Autor	Filip Štěrba	Obdržel	Alex	Verze	1.00

#	Vstupní podmínky	#	Testovací data	
1	Přístup k internetu	1	Jméno	cvutdemo@pingl.app
2	Prohlížeč Chrome, Mozilla, Firefox nebo Safari	2	Heslo	CtuPass753
3	Přihlášený uživatel	3	URL: https://superadmin-87b9e.web.app/	

Krok	Popis	Očekávaný výsledek	Skutečný výsledek	Stav
1	Přihlaště se do administrace	Zobrazí se úvodní obrazovka		Prošel
2	Kliknte v menu na položku Statistiky	Zobrazí se obrazovka se statistikami		Prošel
3	V horním panelu vyberte:			Prošel
3-a	- libovolné 2 podniky			
3-b	- období za poslední týden			
4	Klikněte na tlačítko Filtrovat	Statistiky se změní		Prošel

Poznámky tester	Graf se stavy objednávek je jen v 1/4 obrazovky.

TS 5 - Upravení lokace

ID scénáře	TS_005	Popis testu	Upravení lokace existujícího podniku		
Autor	Filip Štěrba	Obdržel		Verze	1.00

#	Vstupní podmínky	#	Testovací data	
1	Přístup k internetu	1	Jméno	cvutdemo@pingl.app
2	Prohlížeč Chrome, Mozilla, Firefox nebo Safari	2	Heslo	CtuPass753
3	Přihlášený uživatel	3	URL: https://superadmin-87b9e.web.app	

Krok	Popis	Očekávaný výsledek	Skutečný výsledek	Stav
1	Přihlaště se do administrace	Zobrazí se úvodní obrazovka		Prošel
2	V menu zvolte položku Podniky	Zobrazí se seznam podniků		Prošel
3	Vyberte podnik Yumeee	Zobrazí se detail podniku		Prošel
4	Klikněte na záložku Informace	Zobrazí se výběr informací o podniku		Prošel
5	Klikněte na tlačítko Zvolit na mapě	Zobrazí se mapa s lokací podniku		Prošel
6	Posuňte mapu na jiné místo a potvrdt	Souřadnice se změní		Prošel

Poznámky testera	Mapa se zobrazuje v dev verzi.

Příloha F

Obsah přiloženého souboru

Přiložený soubor obsahuje zdrojový kód systému společně s popisem, jak jej spustit.

readme.txt

Soubor, který popisuje obsah přiloženého souboru.

BAKALRASKA-PRACE.pdf

Text práce ve formátu PDF.

/administrace

Zdrojové kódy k práci.