FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

# ZADÁNÍ DIPLOMOVÉ PRÁCE

| | |
|---|---|
| **Název:** | Mobilní aplikace pro webovou SaaS aplikaci Productboard |
| **Student:** | Bc. Igor Kuľka |
| **Vedoucí:** | Ing. Jiří Hunka |
| **Studijní program:** | Informatika |
| **Studijní obor:** | Webové a softwarové inženýrství |
| **Katedra:** | Katedra softwarového inženýrství |
| **Platnost zadání:** | Do konce zimního semestru 2019/20 |

## Pokyny pro vypracování

Tématem práce je analýza, návrh a vytvoření aplikace pro mobilní zařízení pro již existující webovou SaaS aplikaci Productboard, která slouží jako platforma pro podporu produktového managementu.
Postupujte v těchto krocích:

1. Analyzujte potřeby stávajících zákazníků v kontextu návrhu mobilní aplikace.
2. Definujte funkční a nefunkční požadavky a klíčové vlastnosti navrhované mobilní aplikace.
3. Identifikujte a proveďte rozbor částí uživatelského rozhraní stávající webové aplikace Productboard, které odpovídají definovaným požadavkům.
4. Navrhněte základní prototyp mobilní aplikace s ohledem na výhody a omezení mobilních zařízení.
5. Aplikujte vylepšení prototypu na základě uživatelského testování.
6. Vyberte vhodné technologie a implementujte aplikaci pro mobilní zařízení.
7. Proveďte expertní i uživatelské testování mobilní aplikace.
8. Zhodnoťte testování a aktuální stav aplikace.

## Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

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

V Praze dne 19. února 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práca

# Mobilní aplikace pro webovou SaaS aplikaci Productboard

***Bc. Igor Kuľka***

Katedra softwarového inženýrství
Vedúci práce: Ing. Jiří Hunka

15. februára 2019

# Poďakovanie

V prvom rade by som chcel poďakovať vedúcemu tejto diplomovej práce Ing. Jiřímu Hunkovi za ochotu viesť prácu so zadaním, ktoré sám navrhol a chcel realizovať, ako aj za jeho dobre mienené rady a nápady. Taktiež chcem poďakovať Danielovi Hejlovi za to, že mi poskytol príležitosť a možnosti na realizáciu práce pre spoločnosť productboard, kde vedie vývojový tím. Veľká vďaka patrí aj Karlovi Škopkovi, ktorý bol jedným z hlavných programátorov aplikácie na strane servera, bez ktorej by táto aplikácia takmer nevynikla. Moju veľkú vďaku majú aj Geda Juškaitė, Winston Christie-Blick, Juraj Chrappa a Marián Moravčík za cenné rady, podnety a pomoc pri príprave a organizácii zákazníckeho výskumu. V neposlednom rade moja vďaka patrí aj dizajnérom súčasnej webovej aplikácie productboard za možnosť konzultovať jednotlivé štádiá návrhu aplikácie, za ich cenné rady a postrehy. Ďakujem aj všetkým participantom, ktorí mi venovali svoj čas a zúčastnili sa ktorejkoľvek fázy užívateľského výskumu. Určite ďakujem aj svojej priateľke a rodine za pomoc a podporu, ktorú som dostával nie len počas písania tejto práce, ale aj počas celého môjho štúdia.

# Prehlásenie

Prehlasujem, že som predloženú prácu vypracoval(a) samostatne a že som uviedol(uviedla) všetky informačné zdroje v súlade s Metodickým pokynom o etickej príprave vysokoškolských záverečných prác.

Beriem na vedomie, že sa na moju prácu vzťahujú práva a povinnosti vyplývajúce zo zákona č. 121/2000 Sb., autorského zákona, v znení neskorších predpisov, a skutočnosť, že České vysoké učení technické v Praze má právo na uzavrenie licenčnej zmluvy o použití tejto práce ako školského diela podľa § 60 odst. 1 autorského zákona.

V Praze 15. februára 2019 .....

### Odkaz na túto prácu

Kuľka, Igor. *Mobilní aplikace pro webovou SaaS aplikaci Productboard.* Diplomová práca. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2019.

# Abstrakt

Táto diplomová práca sa zaoberá analýzou, návrhom a implementáciou aplikácie productboard pre mobilné zariadenia. Táto aplikácia slúži ako platforma na podporu produktového manažmentu - pomáha systematicky zhromažďovať informácie o potrebách používateľov, strategicky určovať priority v procese tvorby produktu a zjednocovať členov spoločnosti nad spôsobom realizácie produktu. Na základe analýzy súčasného webového riešenia aplikácie productboard a potrieb užívateľov v kontexte mobilnej aplikácie som s použitím technológie React Native aplikáciu implementoval a podrobil automatizovanému a užívateľskému testovaniu. Zistené údaje sú zdrojom podnetov pre jej budúci vývoj, vylepšovanie a využitie v praxi.

**Klíčová slova** mobilná aplikácia, produktový manažment, React Native, Redux, GraphQL

# Abstract

This diploma thesis deals with the analysis, design and implementation of the application productboard for mobile devices. It is a platform for the support of product management processes - it aids the company in systematic gathering of information about users‘ needs, strategic prioritisation in product creation process and aligning the organisation around the way of product creation. Based on the analysis of current web application and the research of users‘ requirements in the context of mobile application, I implemented the mobile application using the React Native framework. The application was subsequently subjected to the automated as well as user testing. The identified data suggest ideas for the future development, improvement and use in practice.

**Keywords** mobile application, product management, React Native, Redux, GraphQL

# Obsah

# Zoznam obrázkov

# Zoznam tabuliek

# Úvod

Žijeme v dobe, kedy sme zo všetkých strán obklopení produktmi a službami rôzneho druhu. Či už ide o produkty fyzické alebo tie, ktorých charakter je striktne virtuálny, všetky existujú z dvoch hlavných dôvodov. Prvým dôvodom je to, že majú svojou existenciou naplniť potreby alebo túžby zákazníkov, robia každodenné úkony svojich odberateľov jednoduchšími, pohodlnejšími a príjemnejšími. Taktiež môžu šetriť čas a poskytnúť tým možnosť dodávať výsledky práce rýchlejšie, prípadne napomáhajú v ich adekvátnej prezentácii. V neposlednom rade môžu slúžiť aj ako zdroj zábavy či spríjemnenia voľných chvíľ. Druhým a nie menej podstatným dôvodom je fakt, že ich predaj prináša finančný zisk jednotlivcom alebo spoločnostiam, ktoré tieto produkty a služby uviedli na trh. V súčasnej dobe môže mať jeden druh produktu veľké množstvo poskytovateľov či výrobcov, ktorých podiel na trhu však nemusí byť priamo úmerný ich počtu. Niektorí dosiahnu pozíciu, kedy sú vo svojej kategórii jedným z najväčších a najzvučnejších mien a získajú si majoritnú časť trhu, iní naopak obsiahnu iba malý zlomok odberateľov. Existuje samozrejme obrovské množstvo faktorov, ktoré ovplyvňujú podiel na trhu, no jedným z najdôležitejších je spokojnosť užívateľov. Dovolím si tvrdiť, že neexistuje žiaden produkt či služba, ktorá by bola dlhodobo úspešná a predávaná aj napriek faktu, že s ňou zákazníci nie sú spokojní. Konkurencieschopnosť trhu nebola ešte nikdy rýchlejšia a pohotovejšia ako je to v dnešnej dobe, a práve preto hraje spokojnosť odberateľov neprekonateľnú rolu v tom, aký je produkt úspešný a aký podiel na trhu si získa.

Veľmi zriedka sa taktiež stáva, že prvotná myšlienka výrobcu a jej následná realizácia je natoľko dokonalá, že sa bez ďalšieho výskumu, vývoja a zdokonaľovania stane dlhodobo úspešnou a obľúbenou v rámci svojej kategórie. Bez toho, aby výrobca poznal a naplnil potreby svojich zákazníkov, je takmer nemožné predbehnúť konkurenciu a získať prvenstvo. Tento fakt je jedným z dôležitých rozdielov, ktorý odlišuje malých hráčov od veľkých a silných producentov, ktorých postavenie na trhu je majoritné. Väčšina z nich totiž už

dávno pochopila, že znalosť problémov, potrieb a túžob svojich súčasných, ale aj budúcich zákazníkov, je pomyselným základným kameňom úspechu. Nie je to ale ani zďaleka jediný faktor, ktorý by predurčoval spoločnosti k ovládnutiu trhu. Rovnako dôležité ako znalosť potrieb zákazníkov je vedieť na základe týchto znalostí robiť správne rozhodnutia tak, aby daný produkt prosperoval a získaval si čoraz väčšiu priazeň súčasných a nových zákazníkov. Nie vždy má totiž aj veľmi početná požiadavka určitého segmentu zákazníkov rovnakú váhu ako iná, síce menej početná požiadavka iného segmentu, ktorý je ale strategicky dôležitejší. Taktiež majú rôzne požiadavky rôznu náročnosť z hľadiska času, ceny, dostupnosti zdrojov a ďalších. Na to, aby kompetentní ľudia dokázali robiť strategicky správne rozhodnutia ohľadom ďalšieho rozvoja produktu, často potrebujú nemalé množstvo údajov, metrík, štatistík a iných dát, na základe ktorých dokážu naozaj urobiť rozhodnutia, ktoré sú racionálne, správne a vedú k prosperite produktu.

Ruka v ruke s tvorbou správnych a strategických produktových rozhodnutí ide aj stanovenie správnych cieľov. Veľmi ťažko sa dá rozhodnúť správne, ak nie je jasné, aký cieľ má byť na základe tohto rozhodnutia dosiahnutý. Pre dosiahnutie úspešného a obľúbeného produktu je teda veľmi dôležité definovať si správne ciele, rovnako ako aj časový horizont, kedy majú byť naplnené.

Často sa stáva, a to hlavne vo väčších spoločnostiach, že tieto strategické rozhodnutia má na starosti človek produktový manažér alebo dokonca celý produktový tím. Či už ide o jednotlivca alebo celý tím, je dôležité, aby mali kompetentní ľudia neustály prístup k vyššie spomenutým dôležitým informáciám, aby mohli správne a efektívne vykonávať svoju prácu. Čím viac ľudí je zahrnutých do procesu rozhodovania o napredovaní produktu, tým väčšia je potreba uchovávať všetky dôležité informácie konzistentne, prehľadne a hlavne dostupne pre všetkých zúčastnených, a to ideálne na jednom mieste.

A presne na podporu celého procesu tvorby excelentného produktu vznikla webová aplikácia productboard, ktorá podáva pomocnú ruku ľuďom a tímom, ktoré sa snažia o vytvorenie neobyčajných produktov a prevádza ich celým procesom od poznávania svojich zákazníkov, ich potrieb, cez definíciu krátkodobých a dlhodobých cieľov, zhromažďovania a vyhodnocovania údajov potrebných pre správne strategické rozhodnutia, až po dodanie finálnych produktov, ktoré čo najlepšie pokrývajú potreby trhu a tým získavajú silnú konkurenčnú výhodu.

Práca produktového manažéra, ale aj ostatných členov produktového tímu, so sebou okrem kancelárskej práce prináša aj cestovanie, stretnutia so zákazníkmi a rôzne iné aktivity, pri ktorých z rozličných dôvodov nie je možné alebo nie je pohodlné používať laptop ako primárny pracovný nástroj. Pri komplexnosti a vysokej informačnej hustote aplikácie productboard nebolo možné a zároveň ani efektívne, aby bola celá webová aplikácia uspôsobená pre využitie na prenosných zariadeniach s malou zobrazovacou plochou, a to práve kvôli spomínanej vysokej informačnej hustote. Napriek tomu, že obrazovka mobilného telefónu prípadne tabletu nedokáže obsiahnuť rovnaké množstvo informácií ako monitor počítača, existuje niekoľko situácií, kedy je využitie mo-

bilného telefónu jednoduchšie alebo dokonca jediné možné riešenie. Tento fakt, spolu s dopytom užívateľov aplikácie po mobilnom variante, mi dal v spolupráci so spoločnosťou productboard priestor zamyslieť sa nad podmnožinou funkcionalít webovej aplikácie, ktorá je pre užívateľov dôležitá práve v situáciach, keď potrebujú plniť úlohy v teréne. A práve návrh užívateľského rozhrania tejto mobilnej aplikácie na základe dát získaných z užívateľského výskumu a následná implementácia tohto návrhu je predmetom tejto diplomovej práce.

Čo sa týka štruktúry tejto práce, v prvej kapitole som sformuloval hlavný cieľ, ktorý chcem v rámci tejto práce naplniť, v druhej kapitole som vymedzil základné pojmy dôležité pre správne pochopenie nasledujúcich kapitol. Tretiu kapitolu som venoval analýze súčasného webového riešenia aplikácie productboard, užívateľskému výskumu, analýze rozdielov pri návrhu pre webovú a mobilnú platformu a definícii funkčných a nefunkčných požiadaviek na mobilnú aplikáciu. Tretia kapitola pozostáva z návrhu kostry aplikácie, jej jednotlivých obrazoviek, návrhu informačnej štruktúry a návrhu užívateľských interakcií. Predposledná kapitola sa zaoberá analýzou vhodných technológií, procesom realizácie mobilnej aplikácie, popisuje štruktúru zdrojového kódu, postupy, technológie a riešenia, ktoré som v rámci implementácie zvolil. Na kapitolu realizácie naväzuje kapitola, ktorá sa zaoberá testovaním, a to jednak automatizovaným testovaním, ale aj fyzickým testovaním v rámci testovania aplikácie s užívateľmi a expertným testovaním. V závere práce hodnotím naplnenie cieľov a uvádzam možnosti pre budúce vylepšenie alebo rozšírenie mobilnej aplikácie.

Kapitola 1

# Cieľ práce

Cieľom tejto diplomovej práce je vhodne zvoleným spôsobom vyhotoviť dôkladnú analýzu potrieb súčasných zákazníkov webovej SaaS služby productboard v kontexte mobilnej aplikácie, ktorá poskytne všetky potrebné informácie a vedomosti o nedostatkoch webového riešenia, ktoré majú vplyv na každodenné pracovné úkony, a tým komplikujú alebo znemožňujú naplnenie týchto pracovných úkonov. Vychádzajúc z tejto analýzy identifikujem problémy, ktoré sú zapríčinené povahou webovej platformy a zároveň je možné ich odstrániť práve využitím rozdielnych charakteristík, ktoré ponúka platforma mobilná. Zoznam a definícia týchto problémov bude slúžiť ako vstup pre vyhotovenie návrhu užívateľského rozhrania mobilnej aplikácie, ktorého informačná architektúra, štruktúra a návrh užívateľských interakcií bude vhodným spôsobom riešiť prekážky spôsobené webovou platformou a bude viesť k úspešnému, jednoduchému a pohodlnému naplneniu potrebných pracovných úkonov. Výberom vhodných technológií a moderných postupov v oblasti vývoja mobilných aplikácií navrhnuté užívateľské rozhranie implementujem, navrhnem a implementujem automatizované testy, ktorými následne implementovanú aplikáciou otestujem. Taktiež podrobím aplikáciu užívateľskému testovaniu.

Kapitola 2

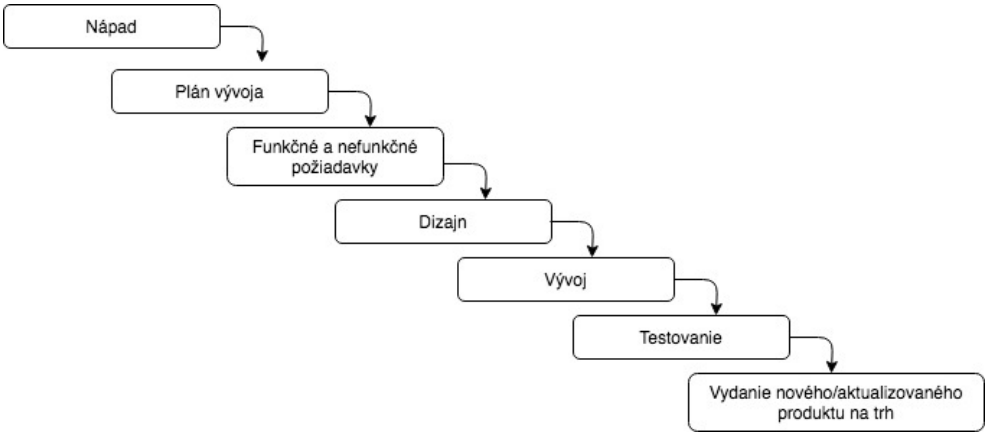
# Úvod do problematiky a vymedzenie pojmov

Hneď v úvode tejto práce je pre správne pochopenie nasledujúcich kapitol veľmi dôležité definovať, čo je produkt a k čomu slúži, ako funguje produktový manažment a prečo fundamentálne ovplyvňuje úspešnosť produktu na trhu, kto je produktový manažér a aké úlohy v spoločnosti plní a aké postupy a techniky sú v súčastnosti považované za osvedčené v kontexte tvorby, distribúcie a predaja produktov.

## 2.1 Produkt

Produkt je tovar, ktorý je ponúkaný zákazníkom za účelom predaja. Môže ísť o fyzický objekt, ale rovnako môže ísť aj o službu, ktorú obchodník poskytuje svojím zákazníkom za úplatu. Forma takéhoto produktu môže byť fyzická, ale taktiež môže nadobúdať formu virtuálnu, ktorá začala dávať najväčší zmysel príchodom a rozšírením internetu. Každý produkt vyžaduje pre svoj vznik nejaké počiatočné náklady a následne je predávaný za určitú cenu, pričom táto cena býva vždy vyššia ako počiatočné náklady a rozdiel medzi predajnou cenou a počiatočnými nákladmi tvorí zisk. Táto cena je taktiež závislá na trhu, na ktorom je produkt predávaný, na segmente zákazníkov, ktorým je tento produkt predávaný, na jeho kvalite, na marketingu, ktorý tento produkt sprevádza a prípadne na ďalších faktoroch špecifických pre daný produkt. Každý produkt má taktiež svoju životnosť a po jej skončení je potreba tento produkt nahradiť iným.

Ak budem v kontexte tejto diplomovej práce hovoriť o produkte, tak mám vo väčšine prípadov na mysli digitálny produkt.




Obr. 2.1: Diagram vývojovej metódy Waterfall popisujúci jednotlivé fázy danej metódy a ich vzájomné poradie

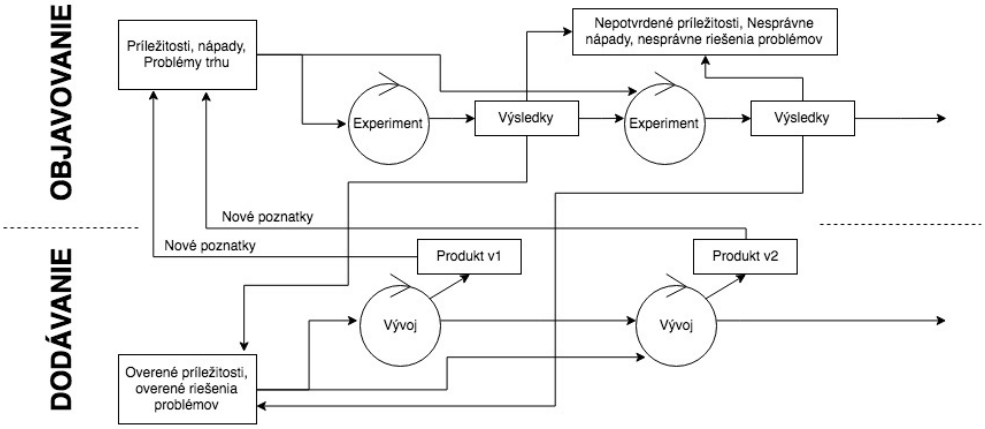
## 2.2 Metódy vývoja produktu

Produkt v rámci svojho životného cyklu prechádza viacerými fázami počínajúc prvotnou myšlienkou, cez stanovenie produktovej vízie a stratégie, dizajn, vývoj, vydanie produktu na trh až po jeho zánik. Samozrejme, existuje mnoho metód a postupov, ktoré určujú, ako budú jednotlivé fázy za sebou postupovať, no medzi tie najznámejšie patrí tzv. metóda "Waterfall", ktorá bola populárna a veľmi rozšírená v minulosti, no kvôli jej nedostatkom, ktoré popisujem v sekcii nižšie, ju čím ďalej tým viac nahrádza metóda kontinuálneho objavovania a dodávania.

### 2.2.1 Metóda Waterfall

Na obrázku 2.1 je zobrazený diagram popisujúci jednotlivé fázy metódy Waterfall a postupnosť, v ktorej sú vykonané. Na začiatku sú nápady súvisiace s produktom, ktoré buď prichádzajú od vedenia alebo zakladateľov spoločnosti, prípadne od produktového manažéra, alebo môžu taktiež pochádzať od aktuálnych alebo potenciálnych zákazníkov. V druhom kroku je definovaný plán vývoja, ktorý obsahuje zoznam funkcionalít, ktoré má produkt obsahovať, a tie sú zvyčajne zoradené podľa priority a termínu či obdobia, v ktorom by mali byť hotové. Následne sa definujú konkrétne funkčné a nefunkčné požiadavky, ktoré neskôr slúžia ako vstupné informácie vo fáze dizajnu a vývoja samotného produktu. Finálnemu vydaniu produktu na trh ešte predchádza dôkladné otestovanie, či produkt správne funguje a či spĺňa všetky náležitosti definované v predchádzajúcich fázach.

Táto metóda bohužiaľ trpí viacerými nedostatkami, no tým najzávažnejším nedostatkom je fakt, že produkt prejde celým procesom od nápadu až po vytvorenie samotného produktu v produkčnej kvalite a ku skutočnému zákazníkovi




Obr. 2.2: Diagram vývojovej metódy kontinuálneho objavovania a dodávania popisujúci jednotlivé fázy danej metódy a ich vzájomné vzťahy

sa dostane až na úplnom konci. Bohužiaľ, v praxi sa často stáva, že viac ako polovica nápadov v konečnom dôsledku nefunguje a zvyšok nápadov častokrát potrebuje niekoľko iterácií na to, aby sa z nápadu stala naozaj hodnotná súčasť produktu. A keďže o tom, či je produkt hodnotný a či bude medzi zákazníkmi obľúbený a používaný, rozhodujú v konečnom dôsledku samotní zákazníci, veľmi často sa stáva, že pri zvolení tejto metódy sa k zákazníkom dostane produkt, ktorý pre nich nie je úplne vyhovujúci, nerieši problém trhu vhodným spôsobom alebo neprináša očakávanú hodnotu. Bohužiaľ, táto spätná väzba zákazníkov prichádza veľmi neskoro, čo má za následok to, že sa zahadzuje veľké množstvo už vykonanej a nákladnej práce a rovnako úprava produktu do podoby, ktorá bude zákazníkom vyhovovať, môže byť taktiež veľmi nákladná a časovo náročná. V niektorých prípadoch sa dokonca môže stať to, že všetky zdroje, ktoré boli k dispozícii na vývoj produktu, sa vyčerpali a neúspech takéhoto produktu privedie spoločnosť do krachu.

### 2.2.2 Metóda kontinuálneho objavovania a dodávania

Obrázok 2.2 zobrazuje diagram kontinuálneho objavovania a dodávania. Ako je z obrázku zjavné, táto metóda je založená na dvoch paralelne prebiehajúcich procesoch, ktorými sú objavovanie a dodávanie.

Vstupom pre proces objavovania je zoznam príležitostí, ktoré sa na trhu vyskytujú, zoznam nápadov na funkcionality, ktoré by produkt mohol obsahovať a v neposlednom rade zoznam problémov, ktoré boli na trhu identifikované a daný produkt ich má za úlohu vyriešiť. Pre každý z týchto vstupov je vykonaný experiment, pri ktorom sú iteratívne navrhované riešenia, z týchto riešení je následne vytvorený prototyp, ktorý je následne validovaný s vybranými súčasnými alebo potenciálnymi zákazníkmi. Výsledok takéhoto

experimentu môže dopadnúť troma spôsobmi. Buď zistíme, že určité riešenie zákazníkom nevyhovuje, neprináša im očakávanú hodnotu alebo ho z iného dôvodu nechcú používať, prípadne zistíme, že adekvátne riešenie v produkčnej kvalite by bolo s dostupnými zdrojmi nerealizovateľné alebo príliš nákladné na to, aby spoločnosti generovalo zisk. Takéto riešenie sa neosvedčilo a nebude obsiahnuté vo finálne dodanom produkte. Rovnako sa ale po niekoľkých iteráciách dá dopracovať k riešeniu, ktoré je vyhovujúce, a v tom prípade má zmysel investovať zdroje do vývoja tohto riešenia v produkčnej kvalite. Poslednou možnosťou je zistenie, že daný nápad alebo problém úzko súvisí s iným problémom, a preto sa výsledky experimentu použijú ako vstupné dáta pre experiment súvisiaceho problému.

Proces dodávania má na vstupe zoznam riešení, ktoré boli v procese objavovania overené, a ide teda spravidla o optimálne riešenia, u ktorých je rozumné a žiadúce investovať zdroje do vývoja konkrétneho riešenia v produkčnej kvalite. Výstupom každého vývojového cyklu je verzia produktu produkčnej kvality, ktorá je vypustená na trh medzi zákazníkov. Každá takáto verzia by mala obsahovať analytické nástroje, ktoré budú zbierať rôznorodé údaje o tom, ako zákazníci tento produkt používajú, alebo údaje o prípadných chybách a problémoch s danou verziou produktu. Taktiež je veľmi dôležité poskytnúť zákazníkom nejakú formu podpory, kde môžu získať odpovede na svoje otázky ohľadom produktu alebo nahlásiť prípadné problémy. Tieto analytické údaje spolu s informáciami zo zákazníckej podpory sú pravidelne analyzované a pridávané na zoznam príležitostí a problémov do procesu objavovania.

Z popisu tejto metódy je zjavné, že kľúčovou výhodou tejto metódy oproti metóde Waterfall je včasná eliminácia nevyhovujúcich riešení, čím sa výrazne zvyšuje pravdepodobnosť priaznivej adopcie produktu na trhu a zároveň sa tým výrazne znižuje plytvanie zdrojmi.

## 2.3 Produktový manažment

Produktový manažment ma za úlohu podporovať a riadiť všetky aktivity spojené s plánovaním, vývojom, marketingom, uvedením produktu na trh a jeho predajom, a to počas celého životného cyklu produktu od prvotného nápadu až po samotný zánik tak, aby v čo najväčšej miere prispel k naplneniu obchodných cieľov, ktorými sú typicky generovanie zisku a maximalizácia podielu na trhu. Tieto ciele je možné naplniť iba dokonalou znalosťou trhu, jeho potrieb a problémov, výberom správnych techník a postupov a zostavením produktovej stratégie, ktorá bude viesť k návrhu správneho riešenia pre identifikovaný problém na trhu. Takto navrhnuté riešenie by malo spĺňať niekoľko atribútov na to, aby ho bolo možné považovať za naozaj správne.

V prvom rade musí ísť o riešenie, ktoré svojím zákazníkom bude prinášať hodnotu. Samotné vyriešenie problému totiž nie je dostačujúce, ak riešenie svojim zákazníkom neprinesie žiadnu hodnotu. V takom prípade je veľmi pravde-

podobné, že za takéto riešenie zákazníci nebudú ochotní zaplatiť a dokonca ho ani používať. Druhým a nemenej podstatným atribútom je použiteľnosť. Ak navrhneme produkt, ktorý rieši nejaký problém a prináša svojim zákazníkom hodnotu, ale jeho zákazníci nie sú schopní tento produkt používať, pretože je produkt zložitý, neintuitívny a náročný na porozumenie a používanie, tak si veľmi pravdepodobne tento produkt taktiež nevyberú a nebudú používať. Ďalším dôležitým atribútom je uskutočniteľnosť riešenia. Pokojne sa môže stať, že navrhnuté riešenie spĺňa predchádzajúce dva atribúty ale takéto riešenie nie je možné z rôznych dôvodov realizovať. Buď môže existovať technologický limit, kvôli ktorému nebude riešenie realizovateľné, alebo bude realizácia vyžadovať zdroje, ktoré v danom čase pre realizačný tím nie sú dostupné. Nech už je dôvod akýkoľvek, ak je riešenie nerealizovateľné, tak sa nedostane k svojim zákazníkom, a tak nevygeneruje žiaden zisk. Posledným významným atribútom je rentabilita. Aj napriek tomu, že existuje riešenie, ktoré vytvára hodnotu, je použiteľné a zároveň uskutočniteľné a pravdepodobne by bolo na trhu výborne adoptované, nemusí ešte ísť o správne riešenie, nakoľko realizácia takéhoto riešenia môže byť príliš nákladná a tieto náklady môžu presiahnuť očakávaný zisk, čo je v rozpore s obchodným cieľom, aby daný produkt zisk generoval.

U malých a začínajúcich spoločností úlohy produktového manažmentu na seba typicky preberá jeden zo zakladateľov spoločnosti, u spoločností v pokročilejšom štádiu väčšinou nájdeme človeka na pozícii produktového manažéra a v prípade etablovaných spoločností, ktoré vytvárajú viacero produktov alebo je ich produkt príliš komplexný, sú tieto povinnosti rozdelené medzi viacerých produktových manažérov, ktorí spolu s ostatnými pozíciami podieľajúcimi sa na tvorbe produktu tvoria produktový tím. U týchto hlboko zakorenených spoločností je spravidla tento produktový tím rozdelený na dve časti, pričom jedna sa zaoberá vývojom produktu a druhá sa zaoberá marketingovými povinnosťami. Pri komplexnom produkte alebo multiproduktových spoločnostiach býva zvykom, že medzi produktovými manažérmi existuje hierarchická štruktúra a produktoví manažéri na nižších úrovniach sú zodpovední za jeden z mnohých produktov alebo určitú časť komplexného produktu a svoju prácu a výsledky reportujú seniornejším produktovým manažérom, ktorí majú dohliadať na celý produkt alebo produktový rad.

## 2.4 Produktový manažér

Existujú malé rozdiely v definícii pozície produktového manažéra a jeho pracovnej náplne v závislosti na štádiu životného cyklu, v ktorom sa konkrétna spoločnosť nachádza, ale medzi hlavné náplne produktového manažéra patria:

- participuje na tvorbe produktovej vízie a vykonáva dohľad nad tým, aby produkt túto víziu dlhodobo nasledoval

- pozoruje trh a vyhľadáva nové príležitostí pre vylepšenie alebo rozšírenie produktu
- plánuje a pripravuje produktovú stratégiu
- sleduje a vyhodnocuje analytické údaje týkajuce sa produktu
- vedie alebo participuje na rozhovoroch so zákazníkmi
- prijíma hlavné rozhodnutia ohľadom funkcionality, ktorú bude produkt obsahovať a schvaľuje všetky dizajnové a funkčné zmeny, ktoré budú do produktu premietnuté
- riadi a koordinuje produktový tím tak, aby čo najefektívnejšou cestou dostal na trh produkt, ktorý bude zodpovedať produktovej vízii, nasledovať produktovú stratégiu, bude svojim zákazníkom prinášať hodnotu a bude napĺňať obchodné ciele spoločnosti
- udržuje, spravuje a prioritizuje zoznam príležitostí, problémov na trhu a nápadov na novú funkcionalitu produktu alebo jeho úpravy
- prioritizuje overené riešenia a zostavuje plán a poradie, v akom budú vybrané riešenia zakomponované do produktu
- reportuje aktuálny stav, rovnako ako aj plánované zmeny akcionárom spoločnosti a ostatným oddeleniam, ktoré vyžadujú tieto informácie pre výkon svojej funkcie, ako sú napríklad marketingové či obchodné oddelenie

## 2.5 Techniky a postupy produktového manažmentu

Pre všetky z vyššie uvedených povinností produktového manažéra vzniklo postupom času veľké množstvo techník a postupov, ako sa s uvedenými úlohami dá vysporiadať a v tejto sekcii sa zameriam na vysvetlenie princípov niektorých z nich, a to konkrétne tých, ktoré sú v súčasnosti považované za overené a zaužívané. Väčšina zo spomenutých techník a postupov funguje výborne s metódou kontinuálneho objavovania a dodávania, no niektoré z nich sú taktiež aplikovateľné aj na metódu Waterfall. Ide hlavne o techniky používané v procese objavovania, nakoľko vo fáze dodávania má produktový manažér skôr funkciu koordinovať a dohliadať na produktový tím realizujúci pripravenú množinu funkcionalít a zmien.

### 2.5.1 Technika OKR

Názov tejto techniky pozostáva z anglickej skratky OKR pre celý názov *Objectives and key results*, čo v preklade do slovenského jazyka znamená *Ciele a kľúčové výsledky*. Ide o veľmi jednoduchú, no zároveň veľmi účinnú techniku pre riadenie, zameranie sa na najdôležitejšie problémy a zosúladenie cieľov naprieč celou spoločnosťou. V rámci tejto techniky sú najprv definované ciele, ktoré majú byť dosiahnuté za konštantné množstvo času (typicky ide o jeden kvartál) a následne sú definované kľúčové výsledky, ktoré sú jednoznačné, kvantitatívne a merateľné. Na základe týchto kľúčových výsledkov sa po uplynutí zvoleného konštantného množstva času určí, či boli jednotlivé ciele naplnené alebo nie. Táto technika býva často používaná nielen na úrovni celej spoločnosti, ale aj na úrovni jednotlivých produktových tímov a v niektorých prípadoch dokonca aj na úrovni jednotlivcov. Vždy je ale dôležité dodržiavať túto hierarchiu, aby priorita OKR spoločnosti bola vyššia ako priorita OKR tímu a jednotlivca.

### 2.5.2 Rozhovory so zákazníkmi

Ide o jednu z najzákladnejších a najdôležitejších techník, ktorá by nemala chýbať v žiadnej produktovej firme a v ideálnom prípade by túto techniku mal každý produktový manažér dokonale ovládať. Rozhovory so zákazníkmi môžu mať rôzne formy. Niektoré sú viac a iné menej formálne, niektoré nasledujú metodiku používateľského výskumu a iné sa len jednoducho snažia získať nové poznatky a náhľady na to, aké problémy zákazníka sužujú a ako ich v súčasnej dobe rieši. Pri každej takejto interakcii so zákazníkom je možné získať veľmi hodnotné, častokrát až kľúčové poznatky a postrehy. Ako som spomenul, každý rozhovor so zákazníkom môže nadobúdať inú podobu a môže byť vedený na rôzne témy, no existuje niekoľko otázok, na ktoré je dobré poznať odpoveď bez ohľadu na formu alebo tému rozhovoru.

- Zodpovedá zákazník predstave spoločnosti z hľadiska rozdelenia zákazníkov podľa špecifických charakteristík?
- Naozaj má zákazník problémy, ktoré si spoločnosť myslí, že má?
- Ako tieto problémy zákazník rieši v súčasnej dobe?
- Čo zákazník vyžaduje aby bol ochotný opustiť súčasné riešenie a prejsť na riešenie (produkt) spoločnosti?

### 2.5.3 Prototypovacie techniky

Prototypy sú používané ako nástroj k overeniu hypotéz a získaniu nových poznatkov o konkrétnom riešení. Jednou z hlavných charakteristík je, že na jeho vytvorenie je potrebné rádovo menšie množstvo zdrojov a času, než by

si vyžadovala realizácia všetkých možných riešení v produkčnej kvalite. Na rozdiel od textovej špecifikácie ponúka možnosť zamýšľať sa nad riešením na výrazne hlbšej úrovni, a tým napomáha odhaliť mnoho problémov hneď v ranom štádiu produktu. Prototyp je taktiež vhodným nástrojom na zjednotenie pochopenia určitého problému naprieč tímom a na jednoduchšiu tímovú kooperáciu.

#### 2.5.3.1 Prototyp uskutočniteľnosťi

Ide o prototyp realizovaný tímom, ktorý je zodpovedný za technickú realizáciu produktu a má za úlohu odhaliť technické problémy s uskutočniteľnosťou konkrétneho riešenia. Používa sa najčastejšie v prípadoch, ak riešenie zahŕňa použitie novej technológie alebo algoritmu. Obsahuje čo najmenšiu podmnožinu produkčného riešenia, na ktorej je možné s veľkou presnosťou definovať potenciálne problémové úseky.

#### 2.5.3.2 Používateľský prototyp

Tento prototyp simuluje reálny produkt pre potreby používateľského výskumu. V rôznych fázach používateľského výskumu sa používajú rôzne úrovne vernosti prototypu. V počiatočných fázach sa často používa iba málo verný prototyp, ktorý u virtuálneho produktu môže pripomínať veľmi hrubý nákres obrazovky, v neskorších fázach býva vernosť prototypu už pomerne vysoká a používateľ len ťažko príde na to, že sa nejedná o reálny produkt.

#### 2.5.3.3 Prototyp s reálnymi dátami

Existujú prípady, v ktorých nám nestačí otestovať, či zákazník rozumie informačnej architektúre alebo návrhu interakcií, ale potrebujeme overiť riešenie, ktoré vykonáva nejaké operácie s reálnymi dátami a na základe týchto dát dáva konkrétny výstup. V takom prípade je potrebné zabezpečiť tok reálnych dát do prototypu a vytvoriť prototyp, ktorý relevantne preukáže funkčnosť, prípadne nefunkčnosť konkrétneho riešenia za použitia reálnych dát. U tohto prototypu je veľmi dôležité dodať, že ide iba o najmenšiu nutnú podmnožinu produkčného riešenia, ktorá nemusí zohľadňovať všetky možné scenáre a neobsahuje žiadne optimalizácie z hľadiska bezpečnosti, výkonnosti či spoľahlivosti. Napriek tomu, že už ide o vysoko verný prototyp schopný pracovať s reálnymi dátami, nikdy nesmie byť považovaný za hotové riešenie a nasadený do produkčného prostredia.

#### 2.5.3.4 Hybridný prototyp

V praxi sa občas môžeme stretnúť aj s prípadom, kedy budeme potrebovať v rámci jedného prototypu skombinovať vlastnosti niektorých z vyššie uvedených prototypov a k tomu slúži hybridný prototyp.

### 2.5.4 Techniky testovania

Medzi významné techniky, ktoré sú v priebehu vývoja produktu používané, patria aj techniky testovania. V rámci objavovacej fázy produktu je snaha o rýchlu separáciu dobrých nápadov a funkčných riešení od tých, ktoré naopak nefungujú, správna. Existujú štyri hlavné faktory, ktoré je potrebné otestovať predtým, ako je určité riešenie predané do fázy vývoja, kde sa z neho má stať riešenie produkčnej kvality. Týmito faktormi sú hodnota pre zákazníka použiteľnosť a uskutočniteľnosť.

#### 2.5.4.1 Test použiteľnosti

Testovanie použiteľnosti je asi najznámejšia z týchto testovacích techník a vo väčšine prípadov ide o prvú techniku, ktorá príde na um každému pri slovnom spojení používateľské testovanie. Túto techniku najčastejšie používajú produktoví dizajnéri a jej úlohou je otestovať tie návrhy, kde má produktový dizajnér obavy, že informačná architektúra alebo návrh interakcie nemusí byť zo strany zákazníkov správne pochopený. Mnoho produktov má totiž komplexné postupy, ako sa dostať k žiadanému výsledku a úlohou dizajnéra je uistiť sa, že navrhnutá interakcia dáva používateľom zmysel a taktiež v predstihu identifikovať problémové miesta a navrhnúť vhodnejšie riešenie [1]. Táto technika je typicky praktikovaná s vybranou množinou zákazníkov alebo vybranou množinou používateľov, ktorí zodpovedajú charakteristikám segmentu trhu, na ktorý sa produkt má dostať. Toto testovanie má svojho moderátora, ktorý používateľa prevádza sadou testovacích scenárov, pričom priebeh celého testovania je buď pozorovaný výskumníkom, ktorý okamžite analyzuje situáciu a zaznamenáva poznatky dôležité pre ďalšiu iteráciu nad dizajnom, alebo je z testovania vytvorený audiovizuálny záznam, ktorý je neskôr rovnako spracovaný kompetentným výskumníkom.

#### 2.5.4.2 Test hodnoty

Je známe, že zákazníci si nekúpia produkt alebo nebudú používať určitú funkcionalitu, pokiaľ nebudú vnímať, že im tento produkt alebo táto funkcionalita prinesie hodnotu. Ak dokonca zákazník aktuálne už používa nejaký produkt na riešenie svojho problému, tak produkt, ktorý chceme, aby začal používať, musí zákazník vnímať ako podstatne lepší, aby bol ochotný investovať čas a peniaze na prechod k inému produktu. A ako už názov napovedá, práve tento test slúži k testovaniu hodnoty produktu pre zákazníka.

Pri testovaní hodnoty je potrebné zamerať sa na tri aspekty:

**Dopyt**
: Niekedy nie je úplne jasné, či po plánovanom produkte bude dostatočný dopyt. Preto je pred jeho realizáciou potrebné položiť si nasledujúce

otázky: Zaujíma sa vôbec zákazník o problém, ktorý by mal tento produkt vyriešiť? A ak áno, zaujíma sa oň natoľko, že by bol ochotný prejsť na tento produkt a zakúpiť si ho? [1]. Jednou z efektívnych techník pre zisťovanie dopytu po produkte alebo funkcionalite je technika falošnej vstupnej stránky v prípade zisťovania dopytu po novom produkte alebo falošného tlačidla v prípade dopytu po novej funkcionalite. Nakoľko princíp je u oboch techník do vysokej miery rovnaký, tak ho popíšem pre techniku falošnej vstupnej stránky. Technika spočíva vo vytvorení vstupnej stránky, ktorá bude ponúkať tento nový produkt, bude obsahovať všetky náležitosti, ktoré by obsahovala reálna stránka propagujúca nový produkt, s jediným podstatným rozdielom, a to takým, že pri objednaní tohto produktu by potenciálny zákazník bol namiesto objednávkového formulára presmerovaný na špeciálnu stránku vysvetľujúcu, že ide o prieskum dopytu po danom produkte. Taktiež môže obsahovať možnosť pre potenciálneho zákazníka prihlásiť sa k dobrovoľnej spolupráci a získavať testovacie verzie produktu. Táto falošná stránka bude samozrejme zbierať údaje o počte prístupov a taktiež vyhodnocovať konverzný pomer.

**Zákaznícke reakcie**

Účelom je pomocou kvalitatívnych techník rozlíšiť, kedy používateľ reaguje na testovaný produkt kladne preto, že to vychádza z jeho prirodzenej povahy a nechce uraziť moderátora, alebo preto, že je naozaj nadšený zo samotného produktu a vidí v ňom hodnotu. Tento test je založený na predpoklade, že ak je používateľ naozaj fascinovaný produktom a považuje ho za hodnotnú voľbu pre riešenie problému, ktorý ho trápi, bude ochotný za produkt zaplatiť, odporučiť ho svojim známym alebo investovať svoj čas na podporu rozvoja tohto produktu. Takže v tomto teste požiadame používateľa, aby si produkt zakúpil, zdieľal tento produkt verejne na sociálnych sieťach alebo poskytol referenčné kontakty na svojich priateľov či spolupracovníkov, prípadne ho požiadame o nezanedbateľné množstvo času, ktoré by bez nároku na honorár venoval úkonom spojeným s vylepšovaním produktu. Počas toho zároveň sledujeme reakciu používateľa. Samozrejme, v rámci testu nie je žiaduce, aby používateľ naozaj za produkt zaplatil alebo splnil inú z požadovaných možností, ale jeho ochota vyhovieť alebo naopak nevyhovieť požiadavke vypovedá o jeho reálnom záujme o tento produkt.

**Účinnosť**

Na rozdiel od reakcie zákazníka sú v tomto prípade dôležité kvantitatívne údaje, pretože účelom je zistiť, ako dokonalé je dané riešenie identifikovaného problému a či dostatočne pokrýva celý problém. Ak pre problém existuje viacero možných riešení, tak tzv. A/B testovanie môže veľmi napomôcť pri zisťovaní, ktoré z riešení je účinnejšie a vhodnejšie. Ide

o metódu, kedy sú používatelia rovnomerne rozdelení do takého počtu skupín, aký je počet riešení, a každej skupine je odprezentované iba jedno z riešení, pričom pre každé riešenie sa zaznamenávajú údaje o používaní. Na konci testu sa porovnajú údaje jednotlivých riešení a je vybrané jedno, ktoré dosiahlo najlepšie výsledky.

#### 2.5.4.3 Test uskutočniteľnosťi

Cieľom tohto testovania je, aby bol realizačný tím schopný zodpovedať nasledujúce otázky:

- Vie realizačný tím, ako zrealizovať dané riešenie?
- Sú v tíme zastúpené všetky potrebné schopnosti pre realizáciu?
- Je riešenie realizovateľné v danom časovom rámci?
- Vyžaduje si riešenie nejaké architektonické zmeny aktuálneho systému?
- Sú k dispozícii všetky zdroje potrebné pre realizáciu?
- Sú známe všetky závislosti súvisiace s realizáciou?
- Je možné realizovať riešenie s akceptovateľnou výkonnosťou a škálovateľnosťou?

Vo väčšine prípadov dokáže zodpovedný člen realizačného tímu zodpovedať tieto otázky okamžite, no ak to tak nie je, je potreba požiadať realizačný tím o zhotovenie prototypu uskutočniteľnosti, na základe ktorého bude možné všetky vyššie uvedené otázky zodpovedať.

### 2.5.5 Prioritizácia

Potreba prioritizácie vychádza z jednoduchého faktu, že spoločnosť nemá dostatok zdrojov na to, aby mohla pracovať na všetkých plánovaných riešeniach a funkcionalitách produktu naraz. U digitálnych produktov je často na trh vydaný tzv. MVP, čo je skratkou pre *Minimum Viable Product* a ide o produkt s minimálnou množinou funkcionalít, ktorý je ale plne funkčný a uspokojí potreby prvých zákazníkov [2]. Tento produkt je potom kontinuálne rozvíjaný a vylepšovaný. No nie každá funkcionalita produktu je rovnako hodnotná a dôležitá. Rôzne riešenia si vyžadujú rôzny čas na realizáciu a rôzne množstvo zdrojov. Preto je potrebný proces, ktorý stanoví poradie tých riešení a funkcionalít, ktoré by mal produkt obsahovať, aby dodal svojim zákazníkom čo najväčšiu hodnotu pri každom vydaní novej verzie produktu. Medzi najpoužívanejšie techniky používané pre prioritizáciu nápadov a riešení patria nasledujúce:

**Pomer hodnoty a náročnosti realizácie**

Tento model dáva do pomeru hodnotu, ktorú dané riešenie zákazníkovi prinesie, k náročnosti realizácie tohto riešenia. Typicky je každý z faktorov vynesený na jednu os karteziánskej sústavy, aby bol vzťah týchto faktorov jednoduchšie viditeľný [3].

**Vážené skóre**

Pri tejto metóde sú každému riešeniu priradené hodnoty, ktoré popisujú, akou mierou dané riešenie prispeje k naplneniu jednotlivých cieľov, ktoré si spoločnosť v procese vývoja produktu definovala a každému z týchto cieľov je priradená jeho váha. Nech $O$ je značka pre cieľ, $O_w$ je váha nejakého cieľa a $O_c$ je miera príspevku riešenia k naplneniu daného cieľa, tak potom pre množinu cieľov o veľkosti $n$ sa výsledné skóre spočíta ako $\sum_{i=1}^{n} O_{c_i}/O_{w_i}$ [3].

**Kano model**

Tento model dáva do pomeru pôžitok zákazníka z riešenia k potenciálnej investícii do vytvorenia riešenia alebo úpravy súčasného riešenia. Typicky je rovnako ako u prvej spomínanej metódy každý z faktorov vynesený na jednu os karteziánskej sústavy, aby bol vzťah týchto faktorov jednoduchšie viditeľný [3].

**RICE Skóre**

Ide o skratku pre štyri faktory, ktoré do výpočtu skóre vstupujú a sú to *reach, impact, confidence, effort* v preklade *dosah (koľko zákazníkov daná funkcionalita ovplyvní v definovanom časovom rámci), dopad (ako veľký dopad to na každého zákazníka bude mať, pričom jednotlivým stupňom dopadu sú pridelené číselné hodnoty), istota (percentuálne vyjadrenie istoty, nakoľko sú odhady predchádzajúcich hodnôt správne) a úsilie (v jednotkách času, ktorý strávi jeden človek realizáciou riešenia).* Vo vzorci pre výpočet tohto skóre sú medzi sebou vynásobené faktory dosah, dopad a istota a výsledok je vydelený úsilím [4].

**Škála MoSCoW**

V tomto prípade taktiež veľké písmená v názve tejto metódy pomenúvajú hodnoty, ktoré môže v rámci prioritizácie konkrétne riešenie nadobudnúť. Ide o hodnoty *Must have, Should have, Could have, Will not have* vo voľnom preklade *nevyhnutné, dôležité, vhodné, nevhodné* [5].

Kapitola 3

# Analýza

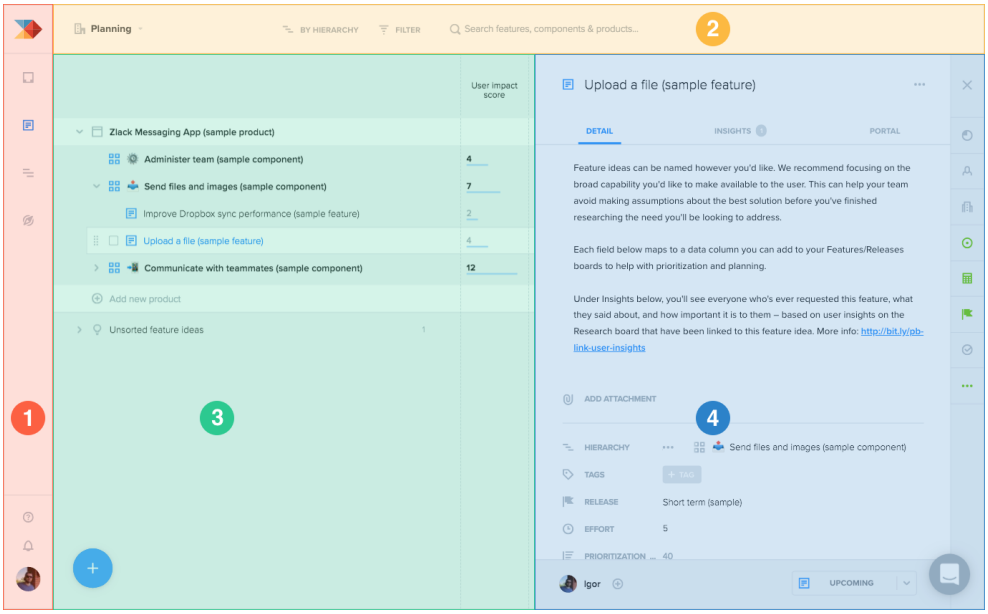
V tejto kapitole popíšem a zanalyzujem súčasnú webovú aplikáciu a rozoberiem jej jednotlivé časti a ich prínos pre produktový manažment, následne rozoberiem jednotlivé metódy zákaznického výskumu a ich výsledky, ďalej budem pokračovať analýzou rozdielov medzi jednotlivými platformami a kapitolu zakončím špecifikáciou funkčných a nefunkčných požiadaviek, ktoré sú dôležitým vstupom pre samotný návrh aplikácie.

## 3.1 Analýza súčasného webového riešenia

productboard je platforma pre podporu produktového manažmentu, ktorá pomáha produktovým manažérom porozumieť potrebám užívateľov, prioritizovať nápady a riešenia, ktoré sa majú v produkte ocitnúť a zjednotiť všetkých ľudí podieľajúcich sa na produkte okolo jasne definovaného plánu, ako bude produkt napredovať. S productboardom môžu produktové tímy zhromažďovať užívateľské vstupy prichádzajúce z rôznych zdrojov a definovať jasné strategické ciele pre fázu prioritizácie. productboard portál prináša produktovým manažérom možnosť validovať svoje nápady a zdieľať so svojimi aktuálnymi aj budúcimi používateľmi plánované zmeny a rozšírenia.

Webová aplikácia productboard sa skladá z množstva rôznych obrazoviek, no vo väčšine prípadov ide o rôzne variácie piatich hlavných obrazoviek, ktorými sú:

- Insights
- Features
- Roadmap
- Portal
- Settings

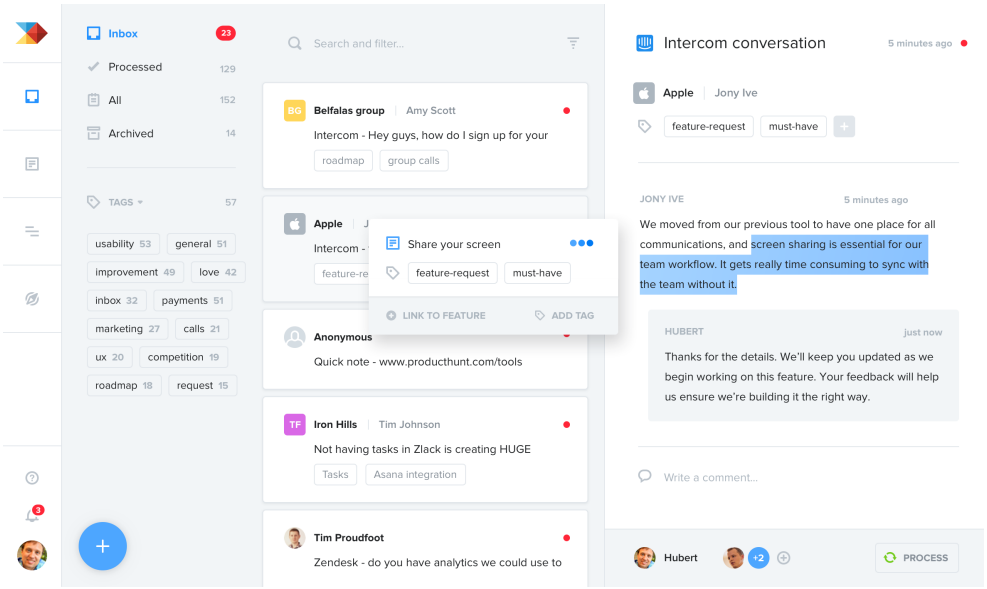



Obr. 3.1: Rozloženie hlavných prvkov webovej aplikácie Productboard

S výnimkou poslednej menovanej, každá z týchto obrazoviek reprezentuje jeden úsek procesu tvorby produktu, v ktorom sa z nápadov stávajú plnohodnotné súčasti produktu. Na obrázku 3.1 je možné vidieť základné rozloženie prvkov, ktoré je rovnaké naprieč väčšinou obrazoviek aplikácie. V ľavej časti pod číslom 1 nájdeme hlavnú navigáciu aplikácie, ktorá pozostáva z loga, štyroch ikon reprezentujúcich odkazy na spomínané štyri hlavné obrazovky a v spodnej časti nájdeme dve ikony, pričom prvá slúži na zobrazenie možnosti nápovedy a druhá na zobrazenie notifikácií. Posledným elementom je náhľad užívateľskej fotografie, ktorý po kliknutí zobrazí menu s odkazmi na profil užívateľa a ďalšie sekcie v nastaveniach, možnosť odhlásenia a prepínač menných priestorov. Pod číslom 2 je na obrázku zobrazený panel, ktorý typicky obsahuje ovládacie prvky pre zmenu pohľadu na dáta, ovládanie filtrovania a vyhľadávacie pole. Číslom 3 je označená hlavná plocha aplikácie, kde sú zobrazené samotné údaje. Forma zobrazenia týchto údajov je závislá na konkrétnom type obrazovky a v rámci tejto sekcie nižšie popisujem detailne každú z nich. Poslednou časťou je bočný panel pod číslom 4, ktorý slúži na správu a zobrazenie detailných údajov o konkrétnej entite v závislosti na zvolenom type obrazovky.

### 3.1.1 Obrazovka Insights

Cieľom tejto časti aplikácie je zhromažďovanie všetkých užívateľských vstupov, ako aj vyhodnotenie a kategorizácia dôležitých poznatkov z týchto údajov.




Obr. 3.2: Snímok obrazovky Isights

Môže ísť o nápady na novú funkcionalitu, pripomienky k aktuálnemu produktu, upozornenia na chyby v produkte alebo akékoľvek ďalšie vyjadrenia užívateľov, ktoré v sebe obsahujú hodnotné informácie pre budúci vývoj produktu. Tieto informácie sú agregované buď automaticky zo zdrojov, ako napríklad používateľský portál, alebo zákaznícky email na definovanú emailovú adresu, alebo poloautomaticky pomocou integrácií s vybranými aplikáciami tretích strán určených na komunikáciu s používateľmi, ako sú napríklad *Intercom*, *Zendesk* alebo *Slack*, pričom používateľ productboardu vyberie konverzáciu alebo jej časť a pomocou integrácie alebo *Productboard rozšírenia pre webové prehliadače* dostane tieto údaje z aplikácií tretích strán alebo iných webových stránok priamo do productboardu. Poslednou možnosťou je vytvorenie záznamu manuálne priamo v aplikácii, čo je výhodné, ak produktový manažér alebo iný člen tímu komunikuje s užívateľom priamo a manuálne zaznamenáva dôležité informácie z tohto rozhovoru.

Ako je možné vidieť na obrázku 3.2, táto obrazovka pripomína svojím rozvrhnutím užívateľského rozhrania na tri stĺpce emailového klienta, a to práve preto, že zhromažďované údaje svojou formou pripomínajú práve emailovú správu alebo konverzáciu. Samotná aplikácia pomenúva jednotlivé záznamy ako *notes*, čiže v preklade *poznámky*. V ľavom stĺpci sa nachádzajú záložky, do ktorých sú jednotlivé poznámky zaradené na základe ich stavu a ďalej sa tam nachádza zoznam štítkov, ktoré sú jednotlivým poznámkam priradené. Uprostred sa nachádza zoznam všetkých poznámok vo vybranej záložke, ktorému predchádza vyhľadávacie pole a ikona pre expandovanie pokročilých možností filtrovania aktuálne zobrazenej množiny poznámok. Na pravej strane

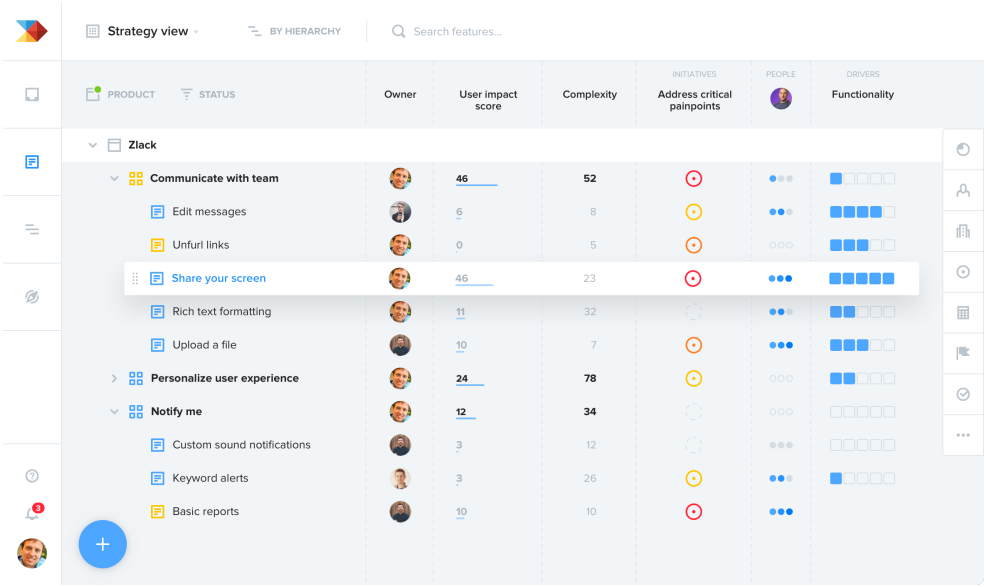
sa nachádza bočný panel, ktorý reprezentuje detail vybranej poznámky.

V hornej časti tohto panelu sa nachádza názov poznámky, prípadne generický názov konverzácie, meno používateľa a prípadne spoločnosť, pre ktorú pracuje a zoznam štítkov, ktoré sú priradené k poznámke. Potom nasleduje samotný obsah poznámky. Pod obsahom je priestor na internú konverzáciu produktového tímu týkajúcu sa danej poznámky. V zápätí tohto panelu sa v ľavej časti nachádza niekoľko náhľadov fotografií. Vedľa prvého náhľadu sa nachádza aj meno. Ide o užívateľa aplikácie, ktorý túto konkrétnu pripomienku vytvoril, alebo ktorý bol priradený ako vlastník tejto poznámky. Po kliknutí na tento element je možné zo zobrazeného zoznamu užívateľov vybrať iného užívateľa a tým zmeniť vlastníka poznámky. Ďalšie fotografie označujú užívateľov, ktorí boli priradení, aby danú poznámku sledovali. Pri veľkom počte poznámok tak môžu používatelia sledovať práve tie poznámky, ktoré ich zaujímajú. Kliknutím na ikonu plus je možné priradiť k poznámke ďalšieho sledujúceho. Na pravej strane zápätia sa nachádza tlačidlo, pomocou ktorého je možné zmeniť stav poznámky z nespracovanej na spracovanú, čo úzko súvisí s nasledujúcim procesom.

Ako je na obrázku možné vidieť, časť obsahu je podfarbená modrou farbou a vľavo od tohto vyznačeného textu je vyskakovacie okno. Ide o jednu z kľúčových funkcionalít tejto aplikácie. Produktový manažér alebo člen produktového tímu zodpovedný za vyhodnotenie a spracovanie poznámok takýmto spôsobom označuje dôležité postrehy ohľadom nejakej konkrétnej témy alebo funkcie z poznámky alebo zákazníckej konverzácie a vytvára spojenie medzi týmto dôležitým postrehom a záznamom o nápade na novú funkciu produktu. Práve tento proces má za výsledok to, že v ideálnom prípade každý nápad na novú funkcionalitu obsahuje informácie o tom, ktorí zákazníci sa k tejto funkcionalite vyjadrili, ako sa o tejto funkcionalite vyjadrili a dôležitosť tohto vyjadrenia určená spracovateľom poznámky. Vďaka tomu je možné v každom štádiu návrhu riešenia pre danú funkcionalitu validovať tento návrh oproti pôvodným potrebám a túžbam užívateľov. Po preskúmaní celej poznámky a spracovaní všetkých dôležitých postrehov spracovateľ označí poznámku za spracovanú alebo môže túto poznámku archivovať.

### 3.1.2 Obrazovka Features

Pri zjednodušenom pohľade na obrázok 3.3 sa dá túto obrazovku definovať ako tabuľku, kde riadky tvoria jednotlivé záznamy reprezentujúce jednotlivé funkcie produktu v rôznych štádiách životného cyklu od nápadov, cez funkcie vo fáze objavovania, funkcie, ktoré boli zamietnuté alebo naopak sa stali kandidátmi na vývoj v produkčnej kvalite, až po úspešne realizované funkcie, ktoré boli vydané medzi zákazníkov. Naopak stĺpce tejto tabuľky reprezentujú rôzne významné informácie, údaje a metriky týkajúce sa jednotlivých záznamov, ktoré sú pre projektového manažéra dôležité pri procese tvorby úspešného produktu. Konkrétne môže ísť o nasledujúce údaje:




Obr. 3.3: Snímka obrazovky Features

**Používatelia**

Ide o stĺpec zobrazujúci dôležitosť konkrétnej funkcie pre daného používateľa vyjadrenú na škále takmer totožnej so škálou MoSCoW popísanej v predošlej kapitole. Táto škála taktiež nadobúda štyri hodnoty s významovo podobným, no nie rovnakým názvoslovím. Túto hodnotu definuje projektový manažér alebo zodpovedný člen produktového tímu pri priraďovaní nejakého dôležitého postrehu z konverzácie s používateľom k danej funkcionalite.

**Spoločnosti**

Tento stĺpec poskytuje agregované údaje o dôležitosti danej funkcie pre používateľov patriacich do konkrétnej spoločnosti. Ide o prehľad, koľko užívateľov danej spoločnosti vyjadrilo dôležitosť funkcie pre jednotlivé hodnoty upravenej MoSCoW škály.

**Segmenty**

Každý trh je možné rozdeliť na menšie koherentné celky, ktoré sú v aplikácii pomenované ako segmenty. Tento stĺpec teda vyjadruje dôležitosť určitej funkcionality pre konkrétny segment na trhu. V tomto prípade je taktiež použitá upravená MoSCoW škála.

**Iniciatívy**

Iniciatívy sú ciele definované na krátke až stredne dlhé obdobie. Každá iniciatíva pokrýva nejaký problém trhu, ktorý je potrebné vyriešiť a obsahuje riešenia (súbor funkcií), ktoré sú kandidátmi na vyriešenie daného

problému. Po nájdení adekvátneho riešenia a jeho implementácii je táto iniciatíva považovaná za hotovú. Tento stĺpec obsahuje hodnotu na upravenej MoSCoW škále, ktorá reprezentuje, ako veľmi konkrétna funkcia prispieva k naplneniu iniciatívy.

***Drivers***
Ide o podobný koncept ako u iniciatív s tým rozdielom, že ovládače sú dlhodobo definované ciele, ktoré sa spoločnosť dlhodobo snaží vylepšovať, ale nie je možné presne definovať množinu funkcionalít, ktorá bude viesť k absolútnemu naplneniu tohto cieľa. V tomto prípade nie je použitá upravená MoSCoW škála, ale číselná škála s rozmedzím 0 až 5, pričom 0 znamená, že daná funkcia žiadnym spôsobom nenapomáha naplneniu tohoto ovládača a 5 znamená že daná funkcia je neodmysliteľnou súčasťou pre vylepšenie a priblíženie sa k naplneniu tohto cieľa.

**Vydania**
Tento stĺpec obsahuje informáciu o tom, či je daná funkcia súčasťou konkrétneho produkčného vydania danej verzie produktu alebo nie.

**Úlohy**
V rámci realizácie nejakej funkcie existujú rôzne úlohy, ktoré je potrebné naplniť, aby bolo možné prehlásiť realizáciu za hotovú a ukončenú. Typicky ide o úlohy ako dizajn, vývoj, vytvorenie dokumentácie a mnohé ďalšie. Každá z týchto úloh zároveň má určitý aktuálny status, napríklad, že je naplánovaná, aktuálne v procese alebo ukončená. Tento stĺpec teda reprezentuje status konkrétnej úlohy vo vzťahu k danej funkcionalite.

**Vlastníci**
Každá funkcia má svojho vlastníka, či už ide o používateľa, ktorý túto funkciu v aplikácii vytvoril, alebo k nej bol priradený. Obsahom tohto stĺpca je vlastník konkrétnej funkcie.

**Štítky**
Rovnako ako k poznámkam, aj k funkciám môžu byť priradené rôzne štítky. Tento stĺpec obsahuje zoznam štítkov priradených ku konkrétnej funkcii.

**Úsilie**
Vývoj každej funkcie si vyžaduje určité úsilie. Tento stĺpec obsahuje hodnotu, ktorá reprezentuje, koľko pracovných dní jedného človeka je potrebných na realizáciu danej funkcie. Odporúčané hodnoty si užívateľ môže vybrať z pseudo-fibbonaciho postupnosti počínajúcej hodnotou 0,5 a končiacej hodnotou 100. Ide ale iba o odporúčané hodnoty a prípustné sú ľubovoľné kladné hodnoty.

**Skóre dopadu na užívateľa**
V tomto prípade, rovnako ako u stĺpca spoločnosti, ide o agregované

údaje o dôležitosti funkcie pre užívateľov, no na rozdiel od stĺpca spoločnosti tento stĺpec obsahuje údaje všetkých používateľov, ktorých dôležité vyjadrenia boli priradené k danej funkcionalite.

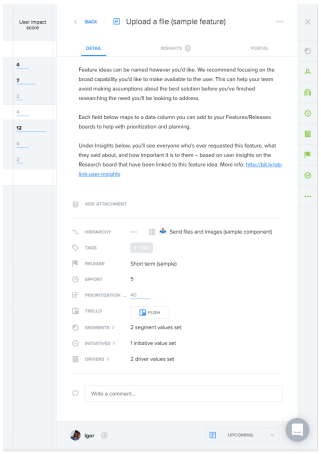
**Prioritizačné skóre**
Užívateľ aplikácie si môže ľubovoľne zostaviť prioritizačné skóre na základe vážených hodnôt vybraných ovládačov. Výsledná hodnota takto zvoleného skóre mu následne pomáha pri procese prioritizácie jednotlivých funkcií. Tento stĺpec obsahuje hodnotu danej funkcionality vo vzťahu k užívateľom definovanému prioritizačnému skóre.

V rozdielnych fázach vývoja produktu potrebuje produktový manažér vidieť rôzne z vyššie uvedených údajov, a preto je možné jednotlivé stĺpce ľubovoľne vypínať a zapínať. Taktiež sa stáva, že produktový manažér potrebuje vidieť informácie o nejakej podmnožine funkcií. A presne pre tento účel existuje pre každý stĺpec možnosť filtrovania na základe hodnôt konkrétneho stĺpca a taktiež aj radenie týchto hodnôt. Taktiež je v hornej časti k dispozícii vyhľadávacie pole, ak je potrebné zobraziť nejakú konkrétnu funkciu. Aby bolo možné sa k takto prefiltrovaným údajom znova vrátiť alebo ich zdieľať s kolegami v rámci produktového tímu, môže produktový manažér takto filtrované údaje uložiť ako pomenovaný pohľad. Tento pohľad môže byť buď privátny, a teda dostupný iba pre užívateľa, ktorý ho vytvoril, môže byť zdieľaný naprieč produktovým tímom alebo môže byť verejný, a teda viditeľný pre ľubovoľného užívateľa aplikácie z rovnakej spoločnosti.

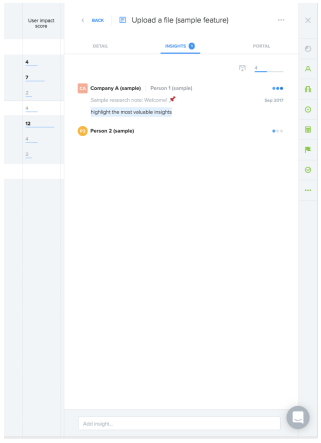
Funkcie na riadkoch tejto tabuľky taktiež môžu byť usporiadané podľa rozdielnych zoskupení. Na obrázku 3.3 je možné vidieť hierarchické zoskupenie, kde existujú tri entity - produkt, komponent a funkcia. Jeden produkt môže obsahovať ľubovoľný počet komponentov, ale taktiež môže priamo obsahovať ľubovoľný počet funkcií. Komponent môže taktiež obsahovať ľubovoľný počet ďalších komponentov, ktoré budú v hierarchii nižšie ako tento rodičovský komponent a taktiež môže obsahovať ľubovoľný počet funkcií. Funkcia je v tomto prípade elementárny prvok a už sa ďalej nedelí na žiadne menšie celky. Ďalšími možnými zoskupeniami sú ploché zoskupenie, ktoré obsahuje jednoduchý zoznam funkcií bez ohľadu na ich miesto v hierarchii, zoskupenie podľa stavu funkcií a zoskupenie podľa termínu vydania, v ktorom sa jednotlivé funkcie nachádzajú. Rovnako ako u zoznamu poznámok na obrazovke Insights, aj na obrazovke Features sa po kliknutí na konkrétnu funkciu zobrazí bočný panel s detailom konkrétnej funkcie.

#### 3.1.2.1 Detail funkcie

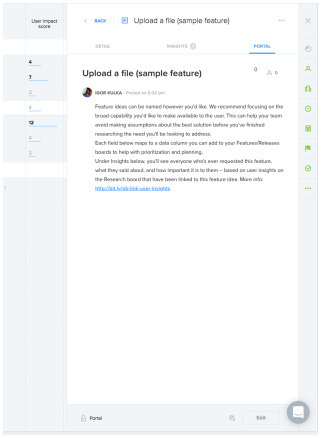
Bočný panel s detailom konkrétnej funkcionality v hornej časti obsahuje záhlavie s názvom funkcionality a kontextovým menu s rýchlymi akciami, ako sú expandovanie bočného panelu do zobrazovacieho módu na celej obrazovke, duplikácia danej funkcionality, spojenie záznamu s iným redundantným záznamom,



Obr. 3.4: Záložka obsahujúca detailné informácie o vybranej funkcionalite



Obr. 3.5: Záložka obsahujúca vyjadrenia zákazníkov o vybranej funkcionalite



Obr. 3.6: Záložka obsahujúca detail portálovej karty k vybranej funkcionalite

archivácia a zmazanie. Ako je možné vidieť na obrázkoch 3.4 až 3.6, zvyšok informácií je rozdelený do troch záložiek, ktorými sú detail, insights a portal.

Záložka detail obsahuje detailný popis funkcionality, definovaný produktovým manažérom tak, aby dostatočne výstižne popisoval problém, ktorý má táto funkcionalita vyriešiť, a spôsob, akým má byť tento problém vyriešený. Tento popis by mal slúžiť ako jediný zdroj pravdy pre celý produktový tím a celú spoločnosť. Ďalej táto záložka obsahuje prílohy, ktoré môžu rozširovať a dovysvetľovať popis, alebo ide o materiály potrebné pre niektorú z fáz realizácie tejto funkcionality. Pod zoznamom príloh sa nachádzajú zopakované údaje jednotlivých stĺpcov tabuľky spojené s konkrétnou funkcionalitou z dôvodu uľahčenia prístupu k týmto údajom. Rovnako ako to bolo u detailu poznámky, aj tu sa nachádza priestor na internú komunikáciu produktového tímu v kontexte tejto funkcionality. Pod spomenutou sekciou komentárov nájdeme zoznam posledných modifikácií a manipulácie s detailom funkcie. Zápätie tejto záložky rovnako ako u poznámky tvorí zoznam vlastníkov a sledovateľov funkcie na ľavej strane a tlačidlo pre zmenu stavu funkcie na strane pravej.

Záložka insights obsahuje zoznam kľúčových vyjadrení zákazníkov k tejto funkcionalite. Ide presne o tie vyjadrenia a postrehy, ktoré produktový manažér alebo iný zodpovedný člen produktového tímu identifikuje v zákazníckych konverzáciách na obrazovke insights a prepojí tieto vyjadrenia s konkrétnou funkcionalitou.

Poslednou záložkou je záložka portál. Tá obsahuje informácie k danej funkcionalite zobrazované na verejnom portáli. Význam týchto informácií, ako aj popis toho, čo zákaznícky portál je a k čomu slúži, popíšem nižšie v rámci

obrazovky portál.

### 3.1.3 Obrazovka Roadmap

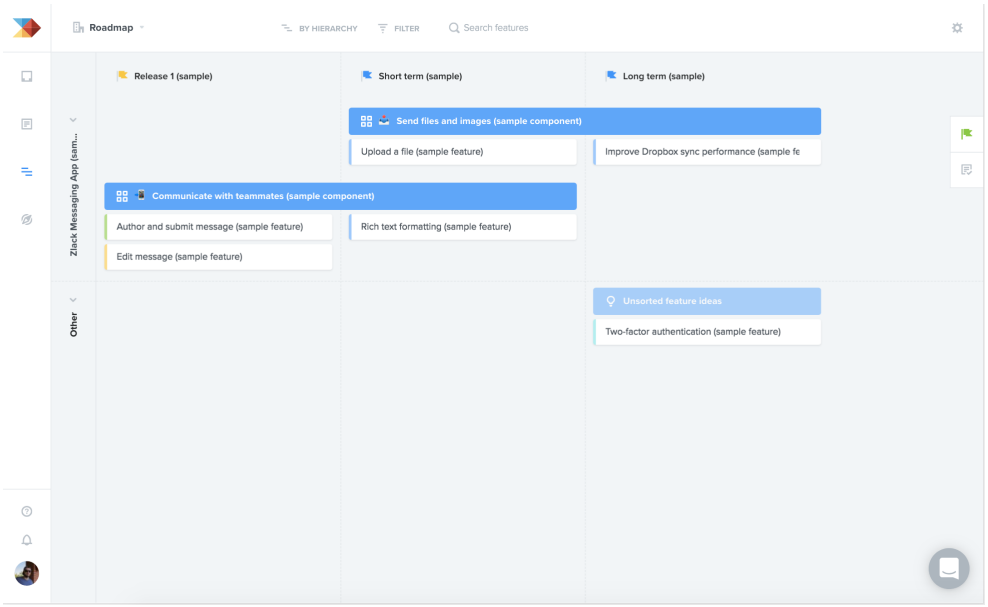
Vo voľnom preklade slovo *roadmap* znamená plán cesty. V kontexte produktového manažmentu teda ide o spôsob vizualizácie výsledku strategického plánovania krátkodobých a dlhodobých cieľov pri vývoji produktu a skladá sa z jednotlivých štádií a míľnikov, ktorými produkt prešiel alebo bude prechádzať v budúcnosti. Tento produktový plán je zdieľaný zdroj pravdy, ktorý popisuje víziu, smer, priority a progres produktu v priebehu času. Je to plán akcií, ktorého hlavnou úlohou je zjednotiť spoločnosť nad krátkodobými a dlhodobými cieľmi produktu a nad spôsobom ich naplnenia [6]. Existuje niekoľko rôznych verzií tohto plánu, ktoré obsahujú iné údaje vzhľadom na to, pre koho tento plán slúži.

**Interný plán pre vývojový tím**
: Tento typ produktového plánu slúži hlavne na zjednotenie produktového tímu nad tým, ako bude postupovať vývoj produktu. S ohľadom na pracovné postupy konkrétneho tímu môže obsahovať jednotlivé míľniky, presné dátumy vydania jednotlivých funkcionalít alebo u tímov pracujúcich agilnými metodikami, ako napríklad Scrum, môže obsahovať jednotlivé šprinty.

**Interný plán pre vedenie spoločnosti**
: Tento plán zdôrazňuje, ako tímová práca prispieva k naplneniu cieľov spoločnosti. Často býva organizovaný po mesiacoch alebo kvartáloch a vizualizuje progres jednotlivých tímov pri naplnení daných cieľov. Všeobecne poskytuje menšiu granularitu úloh, na ktorých sa pracuje a neobsahuje detaily, ktoré sú dôležité pre vývojový tím.

**Interný plán pre obchodné oddelenie**
: Tento plán sa zameriava na informácie o nových funkcionalitách a výhody pre zákazníkov, ktoré produkt obsahuje. Tieto informácie sú veľmi dôležité pre obchodné oddelenie pri rozhovoroch so zákazníkmi za účelom predaja produktu. Táto verzia často neobsahuje žiadne konkrétne dátumy vydania, aby obchodníci zbytočne neprisľúbili termín dodania, ktorý sa nakoniec ukáže ako nedosiahnuteľný.

**Externý plán**
: Tento typ plánu slúži na prezentáciu zákazníkom o zmenách, ktoré sa v blízkej budúcnosti v produkte chystajú. Keďže ide o verejnú prezentáciu aktuálnym aj potenciálnym zákazníkom, tak sa často vyžaduje, aby táto verzia bola vizuálne prívetivá a jednoducho pochopiteľná, aby vzbudila v zákazníkoch záujem o pripravované zmeny.



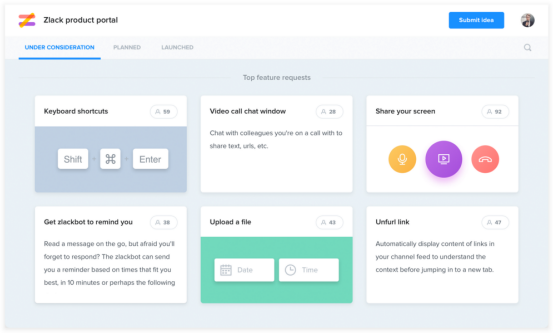

Obr. 3.7: Snímok obrazovky Roadmap

Webová aplikácia productboard sa snaží pokryť všetky spomenuté typy produktového plánu. Na obrázku 3.7 je možné vidieť obrazovku Roadmap, ktorá ponúka užívateľom rozmanité možnosti nastavení tak, aby bolo možné pokryť všetky varianty interného produktového plánu. Externý produktový plán obstaráva obrazovka Portal, ako bude vysvetlené v nasledujúcej sekcii.

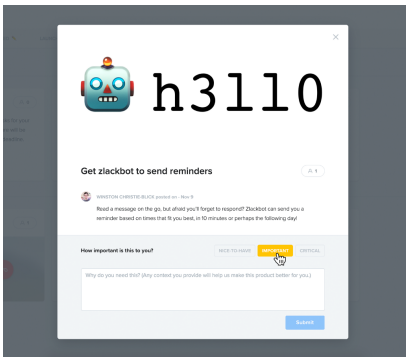
Hlavná zobrazovacia plocha je rozdelená na vertikálne bloky. Tieto bloky môžu reprezentovať jednotlivé míľniky alebo statusy, v ktorých sa daná funkcionalita nachádza. Prepínať medzi reprezentáciami týchto vertikálnych blokov je možné pomocou ikon na pravej strane obrazovky. Jednotlivé míľniky si produktový manažér vytvára sám ľubovoľne podľa potreby, takže si sám môže určiť, či pôjde o bloky reprezentujúce mesiace, prípadne kvartály pre potrebu vedenia spoločnosti, alebo či pôjde o bloky reprezentujúce šprinty pre agilne pracujúci produktový tím. Zobrazenie podľa statusu zase najlepšie vyhovuje plánu pre obchodné oddelenie. Jednotlivé bloky obsahujú náležité zoznamy funkcionalít. Rovnako ako to bolo aj u obrazovky Features, aj tu je možné jednotlivé funkcionality filtrovať, vyhľadávať medzi nimi a prefiltrované informácie ukladať v podobe pomenovaných pohľadov. Taktiež tu podobne ako na obrazovke Features existujú zoskupenia. Najjednoduchším zoskupením je ploché, pri ktorom vertikálne bloky obsahujú iba jednoduché zoznamy funkcií. Hierarchické zoskupenie pridáva pre jednotlivé iniciatívy horizontálne oddeľovače reprezentujúce komponenty hierarchickej štruktúry, ktoré môžu prechádzať viacerými vertikálnymi blokmi a pod týmito oddeľovačmi sa v zodpovedajúcich horizontálnych blokoch nachádzajú zoznamy funkcionalít,

ktoré sú potomkami daného komponentu. Tieto oddeľovače sú vizualizované obdĺžnikmi s modrým pozadím, ako je možné vidieť na obrázku. Každý tento komponent je možné rozbaliť alebo zbaliť, a tým podľa potreby zobraziť alebo naopak skryť hierarchiu nižšej úrovne. Hierarchické zoskupenie je na zvislej ose rozdelené na horizontálne bloky, pričom každý blok reprezentuje jeden produkt. Posledným zoskupením je zoskupenie podľa iniciatív. Toto zoskupenie je vizualizované podobným spôsobom ako hierarchické a v tomto prípade horizontálne oddeľovače reprezentujú jednotlivé iniciatívy. Všetky spomenuté entity sú klikateľné a po kliknutí zobrazia svoj detail vo vysúvacom bočnom paneli rovnako ako to bolo na obrazovke Features.

### 3.1.4 Obrazovka Portal




Obr. 3.8: Verejný portál určený pre zákazníkov a používateľov klientskeho produktu

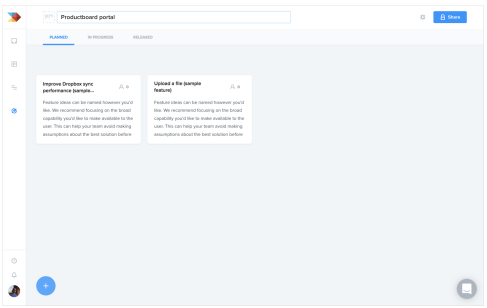



Obr. 3.9: Detail konkrétnej karty s popisom konkrétnej funkcionality

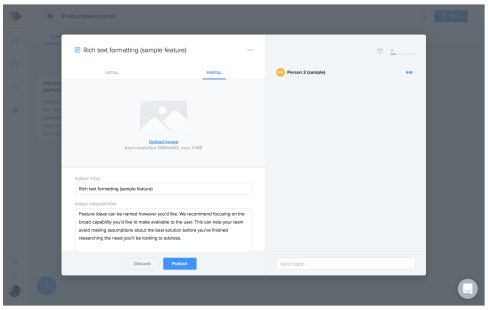
productboard portál plní dve úlohy. V prvom rade ide o ďalší zdroj získavania informácií od používateľov, no zároveň ide taktiež o nástroj na zdieľanie externého plánu so zákazníkmi. Portál pozostáva z dvoch oddelených častí. Prvou je verejná časť pre zákazníkov a používateľov spoločnosti používajúcej productboard, ktorá je úplne vyčlenená z webovej aplikácie productboard a druhou je administračná časť, ktorá je reprezentovaná obrazovkou Portal v rámci webovej aplikácie.

Verejný portál je jednoduchá verejne dostupná stránka, ktorá obsahuje zoznam kartičiek reprezentujúcich jednotlivé funkcionality, ako je možné vidieť na obrázku 3.8. Užívateľ si na stránke môže prezerať jednotlivé kartičky, otvoriť ich detailný popis (pozri obrázok 3.9) alebo po prihlásení pomocou e-mailovej adresy môže za jednotlivé funkcionality hlasovať a tým zvýšiť šancu, že sa daná funkcionalita dostane do produkčnej aplikácie rýchlejšie. Pri hlasovaní je používateľ vyzvaný zadať dôležitosť, ktorú pre neho daná funkcionalita má a zároveň dôvod, prečo je táto funkcionalita dôležitá a ako prispeje

ku skvalitneniu a zefektívneniu pracovného postupu daného užívateľa. Pre výber dôležitosti je opäť použitá upravená škála MoSCoW, pričom negatívna hodnota tejto škály nie je vizualizovaná, ale je implicitne ukrytá v rozhodnutí užívateľa za konkrétnu funkcionalitu nehlasovať. Tento verejný portál môže byť pre prehľadnosť rozdelený do niekoľkých záložiek a v rámci každej záložky môže existovať viacero sekcií. Ak by predsa len užívateľ na portále nenašiel kartičku funkcionality, ktorú práve potrebuje, môže tento nápad na novú funkcionalitu produktovému tímu navrhnúť kliknutím na tlačidlo *Submit idea* a vyplnením obdobného formulára ako pri hlasovaní za už existujúcu kartičku. Obsah tohto formulára je aj pri hlasovaní za existujúcu kartičku aj pri návrhu nového nápadu automaticky pridaný ako nová poznámka na obrazovke insights s náležitými údajmi ako je zvolená dôležitosť a taktiež, že zdrojom konkrétnej poznámky je klientsky portál.




Obr. 3.10: Obrazovka administrácie klientskeho portálu

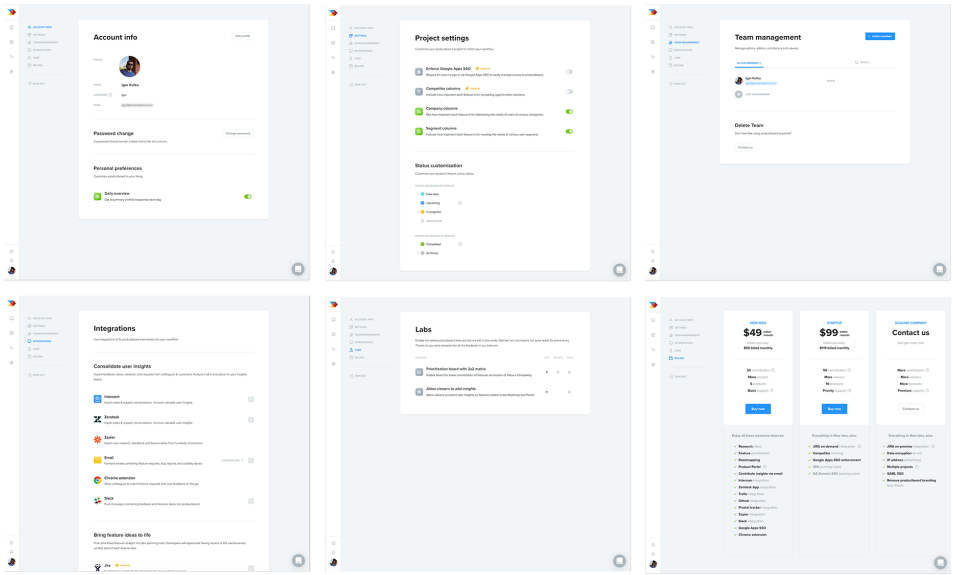



Obr. 3.11: Detail editačného formulára portálovej karty

Administračná obrazovka Portal obsiahnutá vo webovej aplikácii slúži na tvorbu a úpravu obsahu verejného klientskeho portálu a taktiež na úpravu jeho nastavení. Obrázok 3.10 zobrazuje túto administračnú obrazovku a ako je možné vidieť, táto obrazovka je do vysokej miery podobná samotnému klientskemu portálu s tým rozdielom, že väčšina prvkov ponúka možnosť jednotlivé časti upravovať. V záhlaví vľavo sa nachádza element umožňujúci nahrávanie alebo zmenu loga pre klientsky portál a hneď vedľa nasleduje pole pre úpravu názvu portálu. V pravej časti záhlavia sa nachádza ikona zobrazujúca možné nastavenia portálu, ako napríklad zmena primárnej farby. Kliknutím na tlačítko *Share* sa nastavujú jednotlivé možnosti zdieľania klientskeho portálu. V spodnej časti záhlavia sa nachádza lišta, ktorá slúži na vytváranie, úpravu a zmenu poradia jednotlivých portálových záložiek. Hlavná obsahová plocha obsahuje, rovnako ako samotný klientsky portál, kartičky k jednotlivým funkcionalitám. Kartičky je možné na portál pridať pomocou veľkého modrého tlačidla situovaného v ľavom dolnom rohu obrazovky. Po rozkliknutí kartičky sa zobrazí detail kartičky, ako je vidno na obrázku 3.11. Na rozdiel od klientskeho detailu kartičky ide o interný detail vybranej funkcionality, rovnako ako sa zobrazuje do bočného panelu na obrazovke Features alebo Roadmap.

V tomto prípade je ale v expandovanom stave a predvybraná je záložka Portal, ktorá obsahuje formulár, v ktorom je možné editovať náhľadový obrázok, názov a obsah budúcej kartičky na klientskom portáli. Užívateľ sa v prípade potreby pri tvorbe obsahu môže bez problémov pozrieť do záložky Detail pre potrebné informácie, alebo si môže znova prezrieť vyjadrenia zákazníkov k danej funkcionalite, ktoré nájde v pravej časti detailového okna.

### 3.1.5 Obrazovka Settings




Obr. 3.12: Prehľad jednotlivých sekcií v nastaveniach

Táto obrazovka zoskupuje jednotlivé sekcie nastavení, ktorými je možné webovú aplikáciu prispôsobiť tak, aby čo najviac vyhovovala nielen projektovému manažérovi, ale aj všetkým ostatným zamestnancom spoločnosti, ktorí s touto aplikáciu prichádzajú do kontaktu. Obrazovka pozostáva z nasledujúcich šiestich sekcií:

**Užívateľský účet**
: Sekcia slúžiaca na nastavenie základných informácií o aktuálnom užívateľovi, ako fotka, meno alebo heslo pre vstup do aplikácie.

**Projekt**
: V tejto sekcii sa nachádzajú všeobecné nastavenia pre projekt, ako napríklad stavy, ktoré môžu jednotlivé funkcionality nadobúdať.

**Správa tímu**
: Táto sekcia umožňuje produktovému manažérovi do aplikácie prizvať

ďalších používateľov a priradiť im jednu zo štyroch používateľských rolí a tým riadiť prístupové práva jednotlivých používateľov.

**Integrácie**
Nastavenia všetkých integrácií so službami tretích strán sa nachádzajú práve v tejto sekcii.

**Labs**
Táto sekcia poskytuje užívateľom možnosť zapnúť si pripravované funkcionality, ktoré realizačný tím productboardu implementoval z dôvodu otestovania konceptu riešenia s dobrovoľníkmi z radov zákazníkov.

**Platby a plány**
Táto sekcia ponúka meniť voľbu aktuálneho predplatného a platobných údajov.

## 3.2 Analýza charakteristík mobilnej platformy

V súčasnej dobe sa chytré telefóny čím ďalej tým viac približujú osobným počítačom a laptopom, a to nie len svojim závratne sa zvyšujúcim výkonom, ale aj spôsobom ich použitia. Nie je to tak dávno, keď bol chytrý telefón nazývaný chytrým len z dôvodu, že na rozdiel od ostatných vtedajších telefónov bolo do neho možné nainštalovať zopár pomerne jednoduchých aplikácií, ktoré najčastejšie slúžili na zábavu. Situácia sa zmenila a dnešné chytré telefóny dokážu bez problémov nahradiť osobné počítače a laptopy aj v každodenných pracovných úkonoch. Dokonca vďaka stúpajúcemu výkonu a vďaka výhodám ako je výborná prenosnosť týchto zariadení, každodenne pribúda viac a viac ľudí, ktorí nahrádzajú v každodennom pracovnom živote laptop za chytrý telefón.

Softvér, ktorý na týchto mobilných zariadeniach beží, musí ale rešpektovať rozdiely a limity, ktoré mobilná platforma oproti bežným osobným počítačom má. Tieto rozdiely a limity významne ovplyvňujú spôsob, akým je potrebné pristúpiť k návrhu mobilných aplikácií a taktiež k ich realizácii. V tejto časti sa pokúsim zhrnúť jednotlivé rozdiely mobilnej platformy oproti bežným osobným počítačom (keďže v súčasnej dobe môžu používatelia aplikácie productboard tento produkt pohodlne používať prostredníctvom webu iba na zariadeniach ako je osobný počítač) a taktiež vplyv týchto rozdielov na prenos aplikácie z prostredia webu na mobilnú platformu.

**Prenosnosť**
Malé rozmery a nízka hmotnosť prinášajú možnosť použitia mobilných zariadení aj na miestach a v situáciách, v ktorých by bolo použitie osobného počítača nemožné, nekomfortné alebo nevhodné. To so sebou prináša množstvo nových možností pre rozmanitejšie spôsoby, akými sa dá aplikácia v daných situáciách použiť.

**Nestabilita pripojenia**

Aj keď v tomto smere už doba výrazne pokročila, tak na mobilných zariadeniach oveľa viac ako na osobných počítačoch bojujeme s nestabilitou internetového pripojenia. Tento fakt treba brať do úvahy a navrhnúť aplikáciu pre telefón tak, aby sa s touto nestabilitou dokázala vyrovnať, a to hlavne aby kvôli tejto nestabilite nevznikla dátová nekonzistencia a v ideálnom prípade aby výpadok pripojenia neovplyvnil činnosť užívateľa, ak aktuálne nevykonáva akciu, pre ktorú je internetové pripojenie nevyhnutné.

**Výkon**

Aj v tejto oblasti sa mobilné zariadenia čím ďalej tým viac približujú výkonu bežných osobných počítačov, ale to platí hlavne o zariadeniach vo vyššej cenovej triede. Priemerné mobilné zariadenia majú stále nižší výkon ako je to u priemerného osobného počítača. S týmto faktom treba počítať pri implementácii výpočtovo náročnej logiky a taktiež pri práci s veľkým objemom dát.

**Veľkosť zobrazovacej plochy**

Ide o asi najmarkantnejší rozdiel oproti bežným pracovným počítačom, a to hlavne ak berieme do úvahy chytré telefóny. U tabletov rozdiel vo veľkosti zobrazovacej plochy nebýva taký priepastný. Pri návrhu aplikácie pre malé zobrazovacie plochy ako je displej chytrého telefónu treba byť veľmi obozretný. Ak je zvolená vysoká informačná hustota na úkor veľkosti textu, tak sa bude aplikácia veľmi zle používať. Ak je zachovaná rozumná veľkosť textu a elementov, s ktorými užívateľ interaguje, a zároveň je informačná hustota neprimerane vysoká, tak sa aplikácia stáva neprehľadnou. Pri zvolení rozumnej veľkosti textu a rozumného množstva informácií pre danú veľkosť zobrazovacej plochy taktiež môže nastať problém, že informácie, ktoré spolu úzko súvisia, nemusia byť zobrazené spolu, čím sa taktiež môže aplikácia stať neprehľadnou a nepoužiteľnou.

**Vstupné zariadenia**

Na rozdiel od bežných osobných počítačov, u ktorých sú vstupné zariadenia myš a klávesnica, vstupným zariadením pre mobilné zariadenia je vo väčšine prípadov dotykový displej. Z toho vyplýva, že k návrhu užívateľských interakcií pre mobilné aplikácie sa musí pri návrhu aplikácie pristúpiť úplne inak ako k návrhu webovej aplikácie používanej na osobnom počítači. Mobilné zariadenia využívajú množstvo gest, ktorými bežný osobný počítač nedisponuje, a naopak mobilné zariadenia neobsahujú typické akcie myši ako je napríklad kliknutie pravým tlačidlom alebo prechod kurzorom nad určitý element.

**Ovládanie jednou rukou**

Väčšina používateľov chytrých telefónov používa telefón orientovaný prevažne na výšku a veľmi často jednou rukou. S ohľadom na fakt, že odhadovaný celosvetový podiel ľudí s dominantnou pravou rukou je 85-90% oproti 10-15% ľudí, ktorých dominantná ruka je ľavá [7], je vhodné dôležité alebo často používané akcie umiestniť na dosah palca pravej ruky.

**Vzorce užívateľských interakcií**

Rozdiel vo vstupných zariadeniach má za následok rozdiely v zaužívaných spôsoboch interakcie používateľa s mobilným zariadením. Interakcia s dotykovým displejom mobilného zariadenia prebieha pomocou gest. Okrem základných gest ako je ťuknutie, dvojité ťuknutie a posunutie do strán sa na chytrých zariadeniach stretneme s dlhým stiskom, ťahaním, približovaním alebo vzďaľovaním dvoch prstov, rotáciou dvoch prstov a mnohými ďalšími gestami, ktoré sú kombináciou spomenutých. Každé z týchto gest je zaužívané a používa sa pri určitých typoch akcií a na to, aby boli tieto gestá použité správne, je taktiež dôležité myslieť pri návrhu aplikácie a interakcií v nej.

**Prístup k aplikácii a aktualizácie**

Na rozdiel od webovej aplikácie, ku ktorej sa pristupuje na konkrétnu URL adresu a zdrojový kód, ktorý sa vráti do internetového prehliadača a je okamžite v tomto prehliadači interpretovaný, mobilné aplikácie sú distribuované v podobe skompilovaného binárneho súboru prostredníctvom tzv. virtuálnych obchodov, pričom užívateľ si musí takúto aplikáciu stiahnuť zo svojho zariadenia a nainštalovať. Tento fakt úzko súvisí so zmenami a aktualizáciami aplikácie. Zatiaľ čo u webovej aplikácie je možné pomerne jednoducho a rýchlo nasadiť upravený zdrojový kód do produkčného prostredia a pri ďalšom prístupe na danú URL aplikácie je automaticky interpretovaná nová verzia kódu, u mobilnej aplikácie si vyžadujú úpravy zdrojového kódu výrazne väčšiu mieru plánovania, pretože sa upravený zdrojový kód musí znova skompilovať a nahrať do virtuálneho obchodu, a táto aktualizovaná verzia aplikácie musí byť znova nainštalovaná na zariadenie používateľa.

**Rozmanitejšie API**

Mobilné zariadenia dávajú možnosť využiť rozmanité množstvo API, vďaka ktorým je možné v aplikáciách využívať dáta z hardvérových zariadení ako je fotoaparát, GPS modul, polohový senzor alebo infračervený port a iné.

**Orientácia zariadenia**

Väčšina osobných počítačov má obrazovku, ktorá je orientovaná na šírku s pomerom strán typicky 16:9 alebo 16:10. Veľmi zriedka sa stáva, že sa orientácia displeja zmení a takáto situácia je ešte menej častá, ak

uvažujeme o tom, že by sa orientácia obrazovky osobného počítača menila s častým intervalom. U mobilných zariadení je situácia iná. Vďaka svojím rozmerom ponúkajú jednoduchú možnosť zmeny orientácie displeja a aj napriek faktu, že je orientácia na výšku častejšie používaná, nie je až tak ojedinelé, aby užívatelia využívali zariadenie v orientácii na šírku. Preto je nutné, aby bola aplikácia pre mobilné zariadenie navrhnutá tak, aby dokázala pružne reagovať na časté zmeny orientácie zobrazovacej plochy.

## 3.3 Analýza užívateľov a ich potrieb v kontexte mobilnej aplikácie

V tejto sekcii sa budem venovať užívateľskému výskumu v kontexte mobilnej aplikácie. Najprv definujem jednotlivé užívateľské role, ktoré sa v aplikácii vyskytujú a aké úkony v aplikácii vykonávajú, následne vyhodnotím doterajšie vyjadrenia zákazníkov k potrebe mobilnej aplikácie a na základe týchto informácií zvolím a aplikujem vhodné postupy kvalitatívneho a kvantitatívneho výskumu pre prehĺbenie pochopenia problematiky. Nakoniec vyhodnotím a zhrniem výsledky výskumu.

### 3.3.1 Užívateľské role

Produktové tímy sa naprieč rôznymi spoločnosťami môžu odlišovať. Dôvodom je nielen individuálne nastavenie spoločnosti, ale aj samotná veľkosť a vyspelosť spoločnosti. V začínajúcich a menších spoločnostiach, kde sú produktové tímy relatívne malé, bývajú do diskusií a rozhodnutí ohľadom produktu zainteresovaní takmer všetci alebo úplne všetci členovia. V korporátnom prostredí sa situácia výrazne mení a existuje požiadavka na to, aby určité skupiny užívateľov mohli vidieť iba určité časti aplikácie. Rovnako sú v spoločnostiach určité typy pozícií rozdelené na viaceré menšie podtypy často hierarchickej štruktúry. Keďže takéto detailné rozdelenie nie je pre túto prácu dôležité, uvádzam nasledujúce všeobecnejšie skupiny pozícií, ktoré prichádzajú s produktom do kontaktu.

**Produktový manažér**
: Nakoľko ide o hlavného predstaviteľa produktového manažmentu v spoločnosti, tak je viac než jasné, že produktový manažér portrebuje mať prístup ku všetkým informáciám, a teda ku všetkým častiam aplikácie.

**Vývojári**
: Najčastejšie potrebujú prístup k zoznamu funkcionalít, na ktorých majú pracovať, k ich detailom a taktiež k internému plánu.

**Dizajnéri**
: Ako skupina používateľov sú dizajnéri podobní produktovým manažérom,

nakoľko sa zúčastňujú aj fázy objavovania problémov a riešení, ale rovnako sa podieľajú aj na tvorbe riešení. Potrebujú teda prístup ku všetkým častiam aplikácie.

**Zákaznícka podpora**
Ide o ľudí, ktorí sú v denno-dennom nepretržitom kontakte so zákazníkmi a prostredníctvom nich sa pravdepodobne dostane do časti Insights najviac poznámok a konverzácií. Rovnako potrebujú mať prehľad o tom, v akom stave je vývoj funkcionalít, takže zároveň vyžadujú prístup k internému plánu.

**Marketingové oddelenie**
Táto skupina ľudí sa nachádza aj na konci vývojového cyklu nejakej funkcionality, zaujíma ich, aká funkcionalita sa pripravuje, aká sa vyvíja a ktoré riešenia budú v najbližšom čase vydané. Mali by si teda vystačiť s interným plánom.

**Obchodné oddelenie**
Rovnako ako u marketingu, aj obchodné oddelenie potrebuje mať prehľad o funkcionalite, ktorá sa pripravuje, vyvíja alebo už je hotová. Keďže sú obchodní zástupcovia taktiež v priamom kontakte so zákazníkmi, rovnako prispievajú poznámkami a konverzáciami do časti Insights.

**Zakladatelia a iní akcionári**
Aj táto skupina ľudí je v pravidelnom kontakte so zákazníkmi a prispieva do časti Insights. Zároveň vyžadujú prehľad o tom, na ktorých veciach sa pracuje a aká funkcionalita bola dodaná zákazníkom. Tieto informácie sa nachádzajú v internom pláne.

### 3.3.2 Vyhodnotenie insights

Vďaka faktu, že tím podieľajúci sa na tvorbe webovej aplikácie productboard zároveň túto aplikáciu sám využíva, som už hneď v úvode získal počiatočné dáta obsahujúce vyjadrenia zákazníkov productboardu k požadovanej funkcionalite, ktorou je mobilná aplikácia. Prešiel som všetky spomenuté klientske vyjadrenia, pripomienky a požiadavky s cieľom zistiť, aké sú najväčšie problémy pri používaní webovej aplikácie na prenosných dotykových zariadeniach ako tablet alebo chytrý telefón, v akých situáciách a za akých okolností títo užívatelia preferujú využitie takéhoto zariadenia oproti štandardnému využitiu počítača, či existujú špecifické potreby zákazníkov na funkcionalitu, ktorú im webová aplikácia pre charakteristiky platformy nemôže doručiť, a taktiež aké prenosné dotykové zariadenia majú užívatelia najviac záujem používať, keďže tablet a chytrý telefón poskytujú rozdielny komfort pri používaní. Taktiež som zisťoval, ktoré zo základných častí webovej aplikácie potrebujú používatelia na spomínaných zariadeniach používať a aké pracovné postupy potrebujú vykonať.

Našiel som 45 veľmi prínosných a hodnotných vyjadrení z celkového počtu 93 a u každého som si poznamenal, akú časť webovej aplikácie konkrétny používateľ požadoval a či bolo požadované zariadenie tablet alebo chytrý telefón. Vyhodnotenie je možné vidieť v tabuľke 3.1.

Tabuľka 3.1: Tabuľka zobrazujúca vyčíslenie zákazníckych požiadaviek na jednotlivé časti aplikácie a prenosné zariadenia

| Časť aplikácie | Počet zákazníkov | Tablet | Telefón | Nešpecifikované |
|---|---|---|---|---|
| Insights | 18 | 5 | 9 | 4 |
| Features | 15 | 2 | 12 | 2 |
| Roadmap | 6 | 2 | 3 | 1 |
| Portal (admin) | 1 | 0 | 1 | 0 |
| Verejný portál | 0 | 0 | 0 | 0 |
| Celá aplikácia | 4 | 1 | 3 | 0 |
| Nešpecifikovaná časť | 4 | 2 | 2 | 0 |

Z výsledkov je možné hneď vidieť, že najžiadanejšími časťami produktu sú Insights a Features. Taktiež sa zdá, že zákazníci majú záujem využívať mobilnú aplikáciu viac na chytrom telefóne ako na tablete. K týmto informáciám pridávam aj presnejší prehľad akcií a postupov, ktoré z jednotlivých častí aplikácie užívatelia požadovali v mobilnej aplikácii.

V časti Insights zákazníci požadovali nasledujúce akcie:

- čítanie poznámok a konverzácií,
- pridávanie poznámok,
- možnosť označenia dôležitých častí konverzácie a prepojenia s existujúcimi záznamami o funkcionalite.

V časti Features boli požadované tieto akcie:

- čítanie a popis funkcionality,
- čítanie a komentovanie v internej komunikácii na detaile funkcionality,
- pridávanie nových nápadov na funkcionalitu,
- zisťovanie a zmena stavu funkcionality.

V časti Roadmap chceli zákazníci vidieť také informácie v internom pláne, aby dokázali svojím kolegom, akcionárom, prípadne svojím zákazníkom rýchlo zodpovedať na otázky typu “Aké sú plány vo vývoji produktu?” alebo otázky na to, kedy sa nejaká konkrétna funkcionalita dostane do vývoja, prípadne do produkčného vydania aplikácie. Pre ďalšie časti aplikácie som v údajoch nenašiel žiadne špecifické požiadavky.

Taktiež je dôležité spomenúť, že naprieč vyjadreniami zákazníkov o potrebe mobilnej aplikácie sa vyskytovali aj informácie o tom, v akých situáciách zákazníci preferujú využitie chytrého telefónu alebo tabletu oproti použitiu bežného osobného počítača alebo laptopu. Viaceré vyjadrenia sa zhodovali na nasledujúcich štyroch situáciách:

1. V prípade, že sú na stretnutí u klienta alebo mimo kanceláriu a potrebujú si rýchlo poznamenať nejaký dôležitý výrok, nápad alebo požiadavku klienta, prípadne sa klient dožaduje informácií o nejakej konkrétnej funkcionalite a jej stave.
2. Počas cestovania do práce alebo za pracovnými povinnosťami nemajú prístup k počítaču alebo je v daných okolnostiach nepohodlné použiť počítač.
3. Ak sú mimo pracovné miesto s počítačom, ale chcú pridať komentár k zákazníckej konverzácii alebo detailu nejakej funkcionality.
4. Využívajú tablet ako plnohodnotný pracovný nástroj zastupujúci počítač.

### 3.3.3 Kvalitatívny výskum

Informácie nadobudnuté analýzou dát z predchádzajúcej sekcie som sa rozhodol preskúmať hlbšie za pomoci techník kvalitatívneho výskumu. Kvalitatívny výskum vyžíva induktívne formy vedeckých metód, hĺbkové štúdium jednotlivých prípadov, najrôznejšie formy rozhovorov a kvalitatívnych pozorovaní. Cieľom je získať popis zvláštnosti prípadov, generovať hypotézy a rozvíjať teórie o fenoménoch. Kvalitatívny výskum je orientovaný na bádanie a prebieha najčastejšie v prirodzených podmienkach sociálneho prostredia. Plán výskumu má pružný charakter. To znamená, že plán výskumu sa z daného základu rozvíja, premieňa a prispôsobuje podľa okolností a získaných výsledkov [8]. V tejto oblasti výskumu existuje viacero metód získavania informácií, no pre moju situáciu som zvolil pološtrukturovaný rozhovor. Niekde medzi dvoma extrémami, medzi rozhovormi štrukturovanými a neštrukturovanými, sú rozhovory pološtrukturované. Tento typ má vopred daný súbor tém a voľne pridružených otázok, ale ich poradie, voľba slov a formulácia môže byť pozmenená, prípadne môže byt niečo dovysvetlené. Konkrétne otázky, ktoré sa zdajú anketárovi nevhodné, môžu byť dokonca i vynechané, iné naopak môžu byť pridané. Pološtrukturované rozhovory sú flexibilnejšie a voľnejšie než štrukturované, ale sú organizovanejšie a systematickejšie než neštrukturované rozhovory [9].

#### 3.3.3.1 Príprava a priebeh pološtrukturovaných rozhovorov

Predtým, než som vôbec začal s prvým rozhovorom, som si najprv pripravil témy, ktoré som chcel v rámci rozhovoru prebrať. Následne som pre jednotlivé

témy vytvoril zoznam otázok a nakoniec som si pripravil štruktúru samotných rozhovorov. Otázky som podľa obsahu a dôležitosti rozdelil do nasledujúcich kategórií:

**Zahrievacie otázky**
Tie spolu s úvodom rozhovoru slúžia na navodenie príjemnej atmosféry rozhovoru a vytvorením dôvery medzi respondentom a výskumníkom.

**Všeobecné otázky**
Ide o otázky smerované na zistenie, aké problémy zákazníci zažívajú pri snahe použiť webovú aplikáciu na mobilných zariadeniach, aké sú ich potreby a aké majú očakávania od mobilnej aplikácie.

**Špecifické otázky**
Vďaka informáciám zisteným analýzou dát popísanou v predošlej sekcii tejto práce som spoznal základné problémy a situácie, ktoré zákazníci popísali a sformuloval som otázky týkajúce sa najpožadovanejších funkcionalít a najčastejších situácií, v ktorých zákazníci požadovali mobilnú aplikáciu.

Vynímajúc zahrievacie otázky som v každej kategórii mal definované hlavné otázky na určitú tému a k danej hlavnej otázke možné doplňujúce otázky. Taktiež som mal najdôležitejšie z hlavných otázok označené červenou farbou. Otázky som navrhol tak, aby neboli uzavreté a vyžadovali rozvinuté odpovede a zároveň tak, aby neboli navádzajúce, aby som vedome alebo nevedome neviedol respondenta k mnou očakávanej odpovedi. Dokument obsahujúci prípravu a otázky zvolené pre tieto rozhovory je priložený v prílohách tejto práce.

Po definícii otázok a štruktúry rozhovoru som požiadal o rozhovor dvoch interných produktových manažérov spoločnosti productboard a päť produktových manažérov z radov zákazníkov produktu productboard a naplánoval som si s nimi termíny rozhovorov.

Čo sa týka štruktúry rozhovoru, na začiatok stretnutia som si pripravil úvod, v ktorom som poďakoval respondentovi za ochotu zúčastniť sa výskumu, vysvetlil som respondentovi, ako bude rozhovor prebiehať a čoho sa bude týkať a vypýtal som si súhlas s vyhotovením audio-vizuálneho záznamu tohto rozhovoru pre neskoršie spracovanie a vyhodnotenie poznatkov. Ako pokračovanie som zvolil jednoduché zahrievacie otázky pre vybudovanie dôvery a navodenie príjemnej atmosféry a pozvoľna som prešiel k otázkam, ktoré sa týkali predmetu výskumu. Nakoniec som respondentovi opätovne poďakoval za čas a ochotu zúčastniť sa, ponúkol mu možnosť participovať na neskorších testovaniach prototypov aplikácie a ukončil rozhovor.

#### 3.3.3.2 Vyhodnotenie rozhovorov

Po opätovnom prehratí záznamov z jednotlivých rozhovorov a ich vyhodnotení som vytvoril zoznam zistení a hypotéz, ktoré sú podporené prekladom

doslovných citácií respondentov, nakoľko boli jednotlivé rozhovory vedené v anglickom jazyku, pretože oslovení zákazníci pochádzali z rôznych častí sveta. Z dôvodu, že sa v rozhovoroch nachádzali konkrétne mená spoločností, pracovníkov a produktov a taktiež z dôvodu, že som si v rámci rozhovorov nevyžiadal od respondentov súhlas k publikovaniu celého obsahu jednotlivých rozhovorov, nie sú prepisy obsahu rozhovorov súčasťou tejto práce.

Pri otázkach na situácie, v ktorých participanti preferujú využívanie mobilného zariadenia, sa rovnako ako v predošlej sekcii najčastejšie objavovali situácie, kedy participant cestuje do práce a rád by tento čas využil produktívne, ale v dopravnom prostriedku nemá dostatok priestoru na to, aby mohol pohodlne využiť prenosný počítač.

"Áno, cestovanie do práce je určite dôležitá situácia, hlavne tu na pobreží, kde je verejná doprava tak preplnená, že sa jednoducho nikdy nedostanem k tomu, aby som si sadol, a tak pol hodinu iba stojím. Počas celej cesty však mám celkom dobré internetové pripojenie, takže by som mohol vykonať veľa užitočnej práce. Ale takto strávim hodinu času denne v dopravnom prostriedku, a to je hodina, ktorú by som mohol stráviť produktívne, ale nemôžem ju stráviť v productboarde."

Veľký počet stretnutí, ktorý participanti denne absolvujú, robí ich prácu ešte náročnejšou, ak so sebou musia nosiť na každé stretnutie osobný počítač. Na rozdiel od počítača, mobilný telefón majú pri sebe stále a ak by dokázali vykonávať potrebné činnosti pomocou neho, bolo by pre nich komfortné nechať počítač v kancelárii.

"Medzi stretnutiami strávim veľké množstvo času chodením naprieč kampusom. Chodím z jedného stretnutia na druhé a medzi jednotlivými stretnutiami používam telefón, pretože je to najrýchlejší spôsob, ako môžem získať aktuálne informácie o dianí."

V rámci svojej výskumnej práce sa vyskytujú v situáciách, kedy potrebujú za pochodu zapisovať pozorované údaje alebo vyhľadávať potrebné informácie.

"A tiež keď sa stretávam so zákazníkmi, je dobré byť schopný rýchlo si niečo skontrolovať. Hlavne so zákazníkmi, s ktorými často pracujeme, sme často na ich pracovisku, takže máme možnosť pozorovať ich prácu. Ale s počítačom je to veľmi ťažké. A mať možnosť použiť mobilné zariadenie, aby som sa mohol na niečo pozrieť alebo si urobiť nejaké rýchle poznámky, môže byť naozaj hodnotné."

Vyššie citované tvrdenie poukazuje na fakt, že schopnosť vytvárať rýchle poznámky v priebehu rozhovorov s klientmi, pozorovaní ich pri práci ale aj ďalšej výskumnej činnosti, je momentálne veľmi závažný problém, ktorým aktuálne webové riešenie trpí, nakoľko pridávanie poznámok pri použití webovej aplikácie na mobilnom telefóne je takmer nemožné. Tento problém sa vyskytoval v takmer všetkých rozhovoroch a okolnosti boli vždy veľmi podobné. Častokrát participanti spomínali, že na zapisovanie používajú nejakú z aplikácií na zapisovanie poznámok, ale v niektorých prípadoch sa stávalo, že tieto poznámky zabudli následne skopírovať do productboardu, čím sa z ich

práce vytratila hodnota zdieľania vedomostí o klientskych problémoch naprieč celým tímom, prípadne celou spoločnosťou. V jednom prípade participant poznamenal, že mu na zaznamenávanie poznámok vyhovuje aplikácia Evernote a má v záujme ju aj naďalej používať, ale veľmi by ocenil možnosť okamžite tieto poznámky zdieľať do mobilnej aplikácie productboard.

Často sa počas rozhovorov spomenula téma rýchleho prístupu k informáciám o funkcionalitách plánovaných na najbližšie vydanie alebo o stave konkrétnej funkcionality. Produktoví manažéri často pri stretnutiach s akcionármi spoločnosti alebo kolegami z iných oddelení, ako je napríklad marketingové oddelenie alebo oddelenie predaja, musia často odpovedať na otázky ohľadom interného plánu vývoja. Nie vždy však majú tieto informácie uložené v pamäti a rovnako nie vždy majú práve prístup k počítaču. Preto bolo viackrát v rozhovoroch spomenuté, že by bolo veľmi prínosné, ak by sa k týmto informáciám dokázali rýchlo dostať pomocou mobilného telefónu. V niektorých prípadoch dokonca participanti požadovali možnosť presúvať funkcionalitu medzi jednotlivými vydaniami.

"akcionári sa na stretnutiach pýtajú, na kedy je nejaká funkcionalita naplánovaná alebo napríklad na čom sa plánuje pracovať v najbližšom týždni"

"mám v telefóne uložený interný plán vývoja ako PDF, ale práca s ním je nepohodlná, pretože musím stále približovať a oddiaľovať"

"možnosť presunúť funkcionalitu z jedného kvartálu do iného kvartálu by bol pre mňa obrovský prínos"

Veľmi častou témou v rozhovoroch boli aj notifikácie upozorňujúce na rôzne aktuálne udalosti, ktoré sa v aplikácii productboard udiali. Najčastejšie však išlo o možnosť byť informovaný o aktuálne prebiehajúcich interných konverzáciách v detailoch poznámok alebo funkcionalít, o ktoré má participant záujem alebo do nich aktívne prispieval v minulosti. Notifikácie by im dávali možnosť rýchlo a pružne reagovať v konverzáciách, čím by sa mohol skrátiť rozhodovací proces pri riešení nejakého problému alebo definícii nejakej funkcionality. V niekoľkých prípadoch by participant považoval za hodnotné dostávať pravidelnú notifikáciu, ktorá by ho presmerovala na zhrnutie zmien, ktoré v aplikácii Productboard nastali za určité obdobie, pričom najčastejšie bol spomenutý jeden deň.

### 3.3.4 Kvantitatívny výskum

Metódy kvantitatívneho výskumu sú výskumné metódy, ktoré využívajú čísla a systematické spôsoby merania pri skúmaní javov a ich vzájomných vzťahov. Používajú sa na zodpovedanie otázok vzťahov medzi merateľnými premennými so zámerom vysvetliť, predpovedať a kontrolovať merané javy. Celá kvantitatívna štúdia zvyčajne končí potvrdením alebo vyvrátením testovanej hypotézy [10]. Existuje niekoľko typov kvantitatívneho výskumu. Napríklad môže byť klasifikovaný ako:

1. Prieskum
2. Korelačný výskum
3. Experimentálny výskum
4. Kauzálno-porovnávací výskum

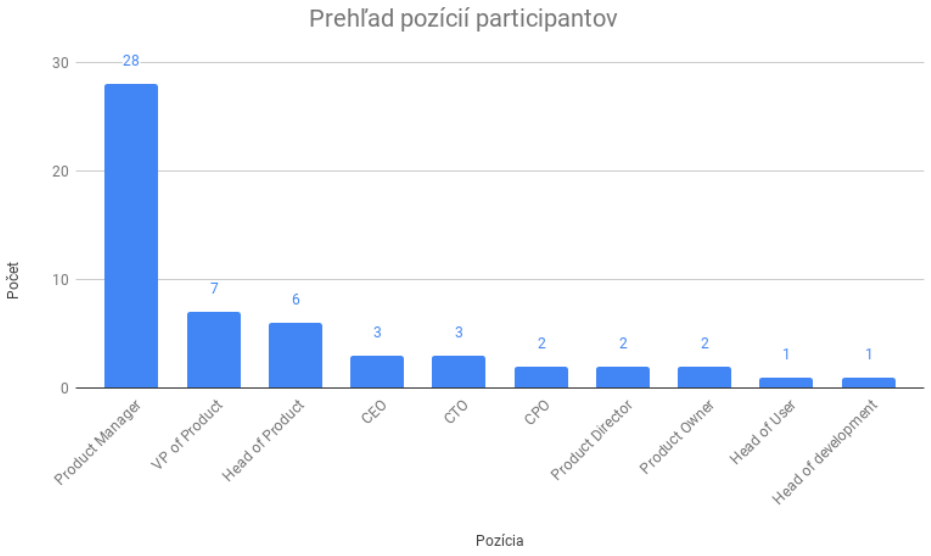
Prieskum využíva systematický výber vzoriek a návrh dotazníka na meranie charakteristík populácie so štatistickou presnosťou [11]. Namerané hodnoty z určitej vzorky populácie poskytujú odhad pre celú populáciu s určitým stupňom presnosti. Tento stupeň presnosti je závislý na tom, ako dobre vybraná vzorka reprezentuje samotnú populáciu. Pre potvrdenie mojich hypotéz o potrebách používateľov webovej aplikácie Productboard v situáciách, kedy nemajú prístup k osobnému počítaču, som si vybral práve metódu prieskumu a vybranú vzorku používateľov aplikácie som požiadal o vyplnenie prieskumu, ktorého otázky som navrhol tak, aby prieskum potvrdil alebo vyvrátil zistenia a hypotézy z predchádzajúceho výskumu.

#### 3.3.4.1 Príprava

Na základe predošlých vedomostí a poznatkov zo zákazníckych vyjadrení som vyhotovil prieskum vo forme dotazníka obsahujúci prevažne uzavreté otázky s výberom jednej alebo viacerých možností, otázky na zoradenie vybraných možností podľa dôležitosti pre konkrétneho participanta a taktiež zopár otvorených otázok s krátkou textovou odpoveďou. Tieto otázky mi majú poskytnúť kvantitatívne údaje o dôležitosti jednotlivých funkcionalít webovej aplikácie pri práci v teréne, o okolnostiach tejto práce v teréne a pre potvrdenie alebo vyvrátenie informácií a hypotéz z predchádzajúcich fáz výskumu. Celkový počet otázok v dotazníku je 18, no dotazník obsahuje aj otázky, ktorých položenie je podmienené výberom definovanej podmnožiny odpovedí nejakej z predošlých otázok. To znamená, že participanta zbytočne neobťažujem doplňujúcimi otázkami, ak v predošlej otázke označil, že je daná funkcionalita nepotrebná. Dotazník som distribuoval dvoma rozdielnymi spôsobmi. Prvým spôsobom bol e-mail s odkazom na prieskum, ktorý som poslal vybranej podmnožine aktuálnych zákazníkov aplikácie productboard. Prevažne išlo o zákazníkov, ktorý sa už v minulosti dožadovali možnosti použiť aplikáciu na mobilnom zariadení. Okrem e-mailov som odkaz na prieskum vložil do verejného kanálu prostredníctvom aplikácie *Slack*, kde tím productboardu pravidelne komunikuje s komunitou ľudí zaoberajúcich sa tématikou produktového manažmentu.

#### 3.3.4.2 Vyhodnotenie

Do tohto prieskumu sa zapojilo 55 participantov z radov súčasných zákazníkov webovej aplikácie productboard. Keďže prieskum som šíril aj pomocou ve-




Obr. 3.13: Graf prehľadu pozícií paticipantov prieskumu

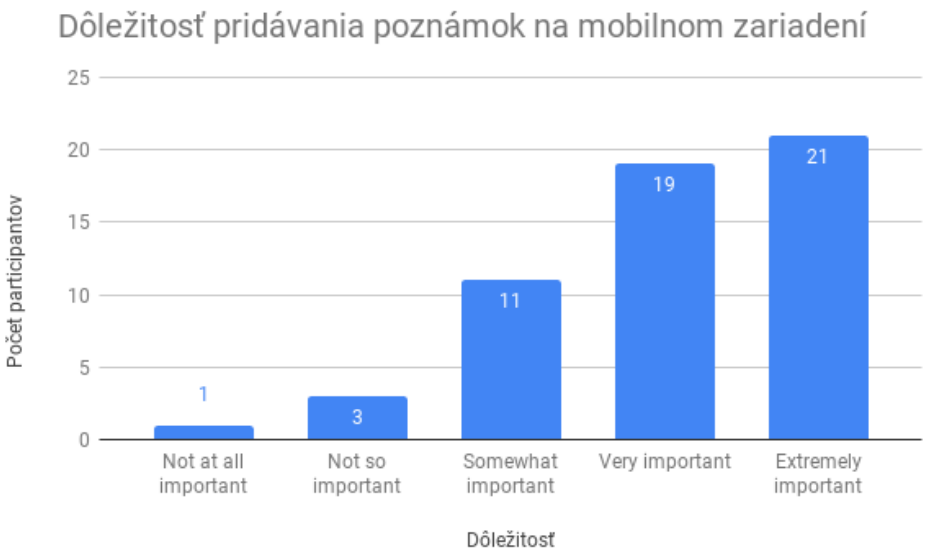
rejného kanálu, kde mohli odpovedať ľudia s rôznymi pozíciami, v prvej otázke som sa pýtal na pracovnú pozíciu participanta, pretože najdôležitejšie sú pre mňa odpovede primárnych používateľov aplikácie productboard, čiže tých, ktorí sa podieľajú na celom životnom cykle produktu.

Graf na obrázku 3.13 jasne ukazuje, že takmer všetci participanti, ktorí vyplnili prieskum, pracujú na pozícii, v ktorej sa podieľajú na všetkých alebo minimálne väčšine fáz životného cyklu produktu. V nasledujúcej otázke ma zaujímalo, aké zariadenia si participanti so sebou berú na stretnutia, a to nie len so zákazníkmi, ale aj na stretnutia s akcionármi, prípadne kolegami. Až 85,5% odpovedalo, že si na stretnutia berú laptop, 74,5% respondentov má pri sebe chytrý telefón a 27,3% si so sebou vezme tablet. Medzi participantmi, ktorí označili, že si so sebou na stretnutia berú tablet, je až 66,6% takých, ktorí na otázku, či používajú tablet ako primárne zariadenie, odpovedali kladne. V otvorenej otázke, kde som dal participantom možnosť spomenúť ďalšie potreby na mobilnom zariadení, ktoré neboli pokryté v prieskume, traja participanti, ktorí zároveň označili, že používajú tablet ako primárne zariadenie, vyjadrili potrebu celej funkcionality webovej aplikácie na tablete. Keďže percento užívateľov tabletu nie je také vysoké a zároveň väčšina týchto užívateľov chce používať tablet ako primárne zariadenie, a teda eventuálne pristupovať ku všetkým funkciám súčasnej aplikácie, považujem lepšie prispôsobenie webovej aplikácie pre použitie na tablete ako vhodné riešenie pre túto minoritnú skupinu, a mobilnú aplikáciu teda nebudem primárne navrhovať pre použitie na tablete, ale zameriam sa na použitie na chytrom telefóne.

V ďalšej otázke som sa zaujímal o to, aké je zastúpenie jednotlivých platforiem, ktoré sú v súčasnosti používané na mobilných zariadeniach. Výsledkom je, že až 78,2% opýtaných používa platformu iOS a zvyšok opýtaných používa platformu Android. Žiadna ďalšia platforma sa vo výsledkoch neobjavila. Kvôli silnému zastúpeniu platformy iOS u zákazníkov aplikácie productboard budem prvú verziu tejto aplikácie navrhovať práve pre túto platformu.

Nasledujúci súbor otázok mal za účel potvrdiť predpoklad, že väčšina respondentov naozaj pracuje počas cesty do práce. Na otázku, či participant zvyčajne pracuje na cestách, až 76,4% opýtaných odpovedalo kladne, ďalších 14,5% odpovedalo záporne, zvyšok označil odpoveď iné a popísali situácie, kedy počas cestovania pracujú a kedy nie. Naväzujúc na predošlú otázku ma zaujímalo, koľko času strávia na cestách počas priemerného pracovného dňa. Na základe odpovedí je možné participantov rozdeliť do troch skupín. Prvú skupinu tvoria tí, ktorí na cestách strávia menej než hodinu, ďalšia skupina cestuje zhruba jednu až dve hodiny a poslednou skupinou sú tí, ktorí strávia na cestách priemerne viac ako 2 hodiny. Zastúpenie participantov v jednotlivých skupinách bolo takmer rovnomerne rozložené. Na základe týchto informácií a informácií z osobných rozhovorov sa môžem domnievať, že mnoho z užívateľov, ktorí zvyknú počas cestovania pracovať, by mohlo siahnuť práve po mobilnej aplikácii práve z dôvodu naplnenia rýchlych a jednoduchých úloh, ak v daných podmienkach nie je možné alebo pohodlné použiť osobný počítač. Aby som predošlé tvrdenie mohol vysloviť s určitou mierou pravdepodobnosti, potreboval by som stráviť výrazne viac času skúmaním tejto konkrétnej situácie za pomoci rozmanitejšieho množstva výskumných techník.

Veľmi častou témou v osobných rozhovoroch bola práve možnosť poznamenávania si poznámok pomocou mobilného telefónu. Preto som sa v nasledujúcej otázke zaujímal o to, či je participant v pravidelnom kontakte so zákazníkmi produktu, za ktorý má zodpovednosť a aká je podstata tohto pravidelného kontaktu. V tejto otázke sa nenašiel ani jeden participant, ktorý by odpovedal, že nie je v žiadnom kontakte so zákazníkmi. Vzhľadom na to, na akých pozíciách participanti pracujú, nie je tento poznatok nijako nezvyčajný. Až 92,7% opýtaných je v kontakte so zákazníkmi pri rôznych formách výskumu. Druhým najpočetnejším spôsobom kontaktu so zákazníkmi boli obchodné stretnutia alebo telefonáty, ktoré uviedlo 63,6%, veľmi podobné množstvo 61,8% je v kontakte so zákazníkmi prostredníctvom zákazníckej podpory, 34,5% sa pravidelne stretáva so zákazníkmi na konferenciách a iných vzdelávacích udalostiach. Na základe faktu, že všetci opýtaní sú v pravidelnom kontakte so zákazníkmi, je pravdepodobné, že si z týchto stretnutí poznamenávajú poznámky. Preto som sa v jednej z nasledujúcich otázok pýtal na dôležitosť vykonávať akcie spojené so spätnou väzbou na mobilnom telefóne. Ku každej z možností mal participant vybrať, akú dôležitosť pre neho daná akcia má na 5-stupňovej škále od hodnoty *not at all important* (absolútne nedôležité), cez hodnoty *not so important* (nie veľmi dôležité), *somewhat important* (trochu dôležité) a *very important* (veľmi dôležité), až po hodnotu



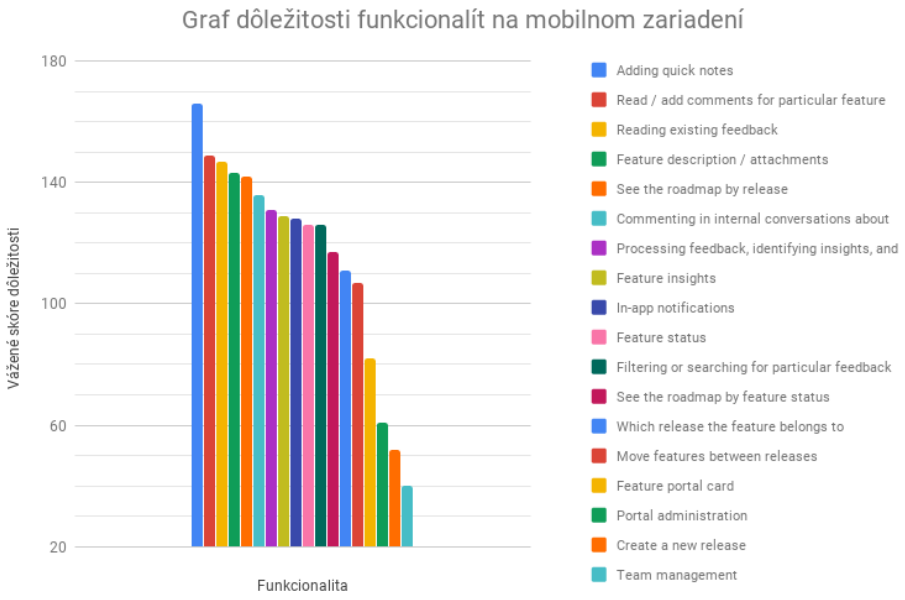

Obr. 3.14: Graf dôležitosti pridávania poznámok na mobilnom zariadení

*extremely important* (extrémne dôležité).

Graf na obrázku 3.14 ukazuje, že väčšina participantov označila túto funkcionalitu za veľmi dôležitú alebo extrémne dôležitú. Ako bude možné vidieť na grafe 3.15, ide dokonca o funkcionalitu, ktorá vyšla z celého prieskumu ako najžiadanejšia, a preto jednoznačne nesmie v navrhovanej aplikácii chýbať.

Okrem otázky na dôležitosť funkcionalít súvisiacich so spätnou väzbou som položil otázku rovnakého typu aj pre iné oblasti produktu. Participanti mali jednotlivé funkcionality z niekoľkých oblastí, ktoré súčasná webová aplikácia poskytuje (interný plán, administrácia portálu, tímový manažment, notifikácie a detailné informácie o funkcionalite), ohodnotiť pomocou rovnakej škály. Každej z týchto textových hodnôt pre dôležitosť $I$ som následne definoval váhu $WI$ v rozmedzí 0 až 4, pričom absolútnej nedôležitosti som priradil hodnotu 0 a ostatným možnostiam 1 až 4 postupne so zvyšujúcou sa dôležitosťou. Pre každú funkcionalitu som následne spočítal vážené skóre použitím vzorca $\sum_{i=1}^{5} WI_i * NI_i$, pričom $NI_i$ je počet participantov, ktorí zvolili danú dôležitosť. Z týchto výsledkov som následne zostavil graf 3.15, ktorý zobrazuje dosiahnutú hodnotu váženého skóre dôležitosti jednotlivých funkcionalít pre participantov na mobilnom zariadení.

Keďže som predpokladal, že možnosť prijímať notifikácie o udalostiach v aplikácii productboard sa umiestni v hodnotení pomerne vysoko, položil som respondentom otázku, v ktorej si mali vybrať jednotlivé udalosti, na ktoré by chceli byť upozornení. Je mi jasné, že nastavenie notifikácií je pomerne individuálna záležitosť a taktiež považujem za potrebné ponúknuť uživateľovi ap-




Obr. 3.15: Graf dôležitosti funkcionalít na mobilnom zariadení

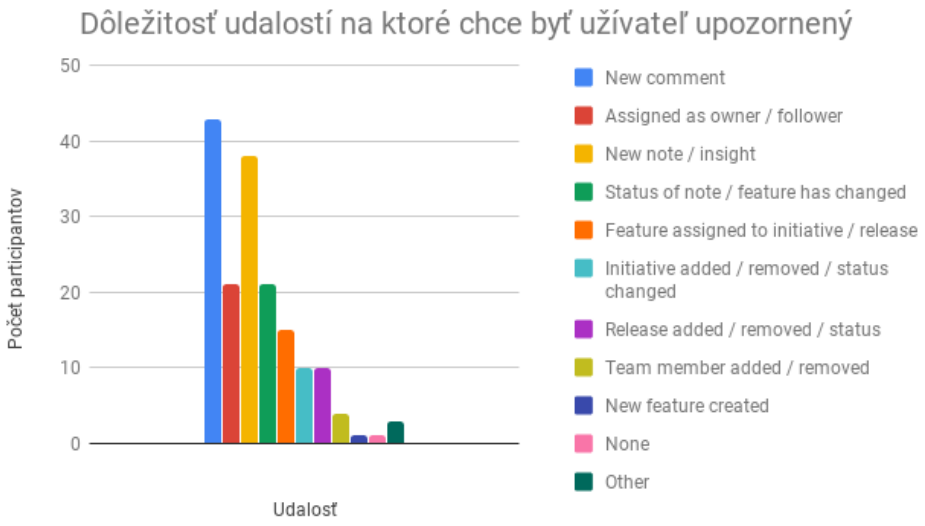
likácie možnosť podrobne nastaviť notifikácie, ktoré chce alebo naopak nechce dostávať, ale informácie získané v tejto otázke mi dajú prehľad o tom, ktoré udalosti sú najžiadanejšie, a najčastejšie odpovede použijem pre východiskové nastavenie aktivovaných udalostí.

Graf 3.16 ukazuje, ako žiadané sú jednotlivé udalosti. Udalosti, ktoré dosiahli nadpolovičný počet respondentov, použijem ako zoznam udalostí, ktoré budú aktívne vo východiskovom nastavení.

Medzi posledné otázky som zaradil vyššie spomenutú otvorenú otázku, kde participant mohol bez obmedzení spomenúť potrebu alebo funkcionalitu, ktorú by vyžadoval na mobilnom zariadení, ale nebola pokrytá v rámci predchádzajúcich otázok. Okrem vyššie spomenutých požiadaviek na funkčnosť celej aplikácie na tablete sa v odpovediach nenachádzali žiadne odpovede, ktoré by sa často opakovali, a teda nepovažujem za potrebné akúkoľvek zo spomenutých potrieb pridať na zoznam funkčných požiadaviek pre prvú verziu mobilnej aplikácie, keďže ide o požiadavky jednotlivcov a nie väčšej skupiny oslovených.

## 3.4 Analýza požiadaviek

Vďaka výsledkom z analýzy súčasných užívateľov webovej aplikácie productboard a ich potrieb v kontexte používania mobilnej aplikácie na prenosných zariadeniach je možné jednoducho a presne definovať funkčné a nefunkčné




Obr. 3.16: Graf dôležitosti udalostí, na ktoré chce byť užívateľ upozornený

požiadavky na mobilnú aplikáciu, ktorej návrhom sa budem zaoberať v nasledujúcej kapitole. Taktiež pridávam niekoľko požiadaviek, ktoré neboli spomenuté užívateľmi v rámci rozhovorov ani v prieskume, ale zo skúseností s návrhom a vývojom užívateľských rozhraní aplikácií považujem tieto požiadavky za nevyhnutné pre správne porozumenie a pohodlné používanie aplikácie, ktorá má byť výstupom tejto diplomovej práce.

### 3.4.1 Funkčné požiadavky

- Prihlásenie
  - Prihlásenie pomocou kombinácie emailu a hesla
  - Prihlásenie pomocou Google Single Sign-On
- Poznámky a zákaznícke konverzácie
  - Zoznam jednotlivých poznámok obsahujúcich zákaznícke konverzácie
  - Detail jednotlivých poznámok
  - Pridávanie nových poznámok
  - Priradenie dôležitých poznatkov zo zákazníckych konverzácií k náležitým funkcionalitám
  - Prehľad internej komunikácie týkajúcej sa konkrétnej poznámky
  - Možnosť prispievať do internej komunikácie týkajúcej sa konkrétnej poznámky

  - Vyhľadávanie poznámok
  - Zmena stavu poznámky
- Záznamy o funkcionalite
  - Zoznam jednotlivých záznamov o funkcionalite
  - Detailný popis konkrétnej funkcionality
  - Indikácia aktuálneho stavu konkrétnej funkcionality
  - Prehľad internej komunikácie týkajúcej sa konkrétnej funkcionality
  - Možnosť prispievať do internej komunikácie týkajúcej sa konkrétnej funkcionality
  - Zmena stavu konkrétneho záznamu
- Interný plán
  - Zoznamy funkcionalít rozdelené podľa náležitosti k definovaným míľnikom interného plánu alebo plánovaným vydaniam
  - Výber zobrazených míľnikov / vydaní
  - Presúvanie funkcionalít medzi míľnikmi alebo vydaniami
- Aktuality
  - Zoznam najnovších poznámok
  - Zoznam najnovších komentárov
- Notifikácie
  - Notifikácie o nových komentároch v interných komunikáciách
  - Notifikácie o nových poznámkach alebo dôležitých zákazníckych vyjadreniach
  - Notifikácie o označení užívateľa ako vlastníka alebo sledovateľa poznámky / funkcionality
  - Zmena stavu poznámky alebo funkcionality
- Nastavenia notifikácií
- Užívateľský profil
- Úvodné oboznámenie s funkciami

### 3.4.2 Nefunkčné požiadavky

- Cieľová platforma - chytré telefóny s operačným systémom iOS
- Výkon
  - Okamžitá odozva
  - Plynulé animácie
  - Adekvátny výkon pri veľkom množstve dát
- Režim off-line
  - Schopnosť aplikácie vykonávať akcie v režime off-line
  - Schopnosť udržať konzistenciu dát pri prechodoch medzi režimom on-line a off-line
  - Automatické získanie aktuálnych dát po opätovnom pripojení
- Modularita
  - Aplikácia musí byť jednoducho rozšíriteľná o novú požadovanú funkcionalitu
- Spoľahlivosť
  - Aplikácia nesmie zničiť a znehodnotiť dáta pochádzajúce z webovej aplikácie
  - Aplikácia musí obsahovať unit testy
  - Aplikácia musí obsahovať e2e testy
- Bezpečnosť
  - Prenos dát medzi aplikáciou a serverom musí prebiehať pomocou zabezpečeného a šifrovaného protokolu

Kapitola 4

# Návrh

Rozdiel medzi dobrou a zlou aplikáciou je často kvalita zážitku, ktorý aplikácia svojmu používateľovi poskytne. Komfortný pocit pri používaní je atribút, ktorý odlišuje úspešné aplikácie od neúspešných. V dnešnej dobe majú používatelia na aplikácie vysoké nároky. Vyžadujú, aby sa aplikácia rýchlo načítala, aby obsahovala jednoduché a príjemné interakcie, aby bolo jednoduché sa v aplikácii vyznať a aby bola na pohľad prívetivá. Ak má byť aplikácia úspešná, používateľský prežitok musí byť nevyhnutnou súčasťou nielen plánovaného dizajnu aplikácie, ale aj celkovej produktovej stratégie [12].

Existuje veľké množstvo vecí, na ktoré je potrebné pri návrhu aplikácie myslieť, no dobrý návrh aplikácie by nemal porušovať žiaden z nasledujúcich šiestich základných dizajnových princípov.

**Štruktúra**
: Návrh by mal účelovo usporiadať používateľské rozhranie zmysluplným a užitočným spôsobom založeným na jasných, konzistentných modeloch, ktoré sú zjavné a rozpoznateľné pre používateľov, spájaním súvisiacich vecí a oddeľovaním odlišných vecí. Štruktúra sa zaoberá celkovou informačnou architektúrou a architektúrou užívateľského rozhrania [13].

**Jednoduchosť**
: Návrh by mal robiť jednoduché, bežné úlohy jednoduchými, komunikovať jasne a jednoducho jazykom, ktorý je užívateľovi dobre známy a poskytovať vhodné skratky tak, aby rozumne a účelne nahrádzali zdĺhavejšie procesy [13].

**Prehľadnosť**
: Návrh by mal umožniť viditeľnosť všetkých potrebných možností a podkladov pre danú úlohu bez toho, aby rušil užívateľa externými alebo nadbytočnými informáciami. Dobrý dizajn neobťažuje používateľov alternatívami a nemätie ich nepotrebnými informáciami [13].

**Odozva**
Návrh by mal používateľov informovať o krokoch, akciách, zmenách stavu alebo o chybách a výnimkách, ktoré sú pre užívateľa relevantné, a to prostredníctvom jasného, stručného a jednoznačného jazyka, ktorý je pre používateľov známy [13].

**Opätovné použitie**
Návrh by mal opätovne používať komponenty, interakcie a vzorce správania, pričom by sa mala zachovať zhoda s pôvodným účelom, čím sa predíde potrebe užívateľa vyhodnocovať a pamätať si chovanie rôznych komponentov a interakcií [13].

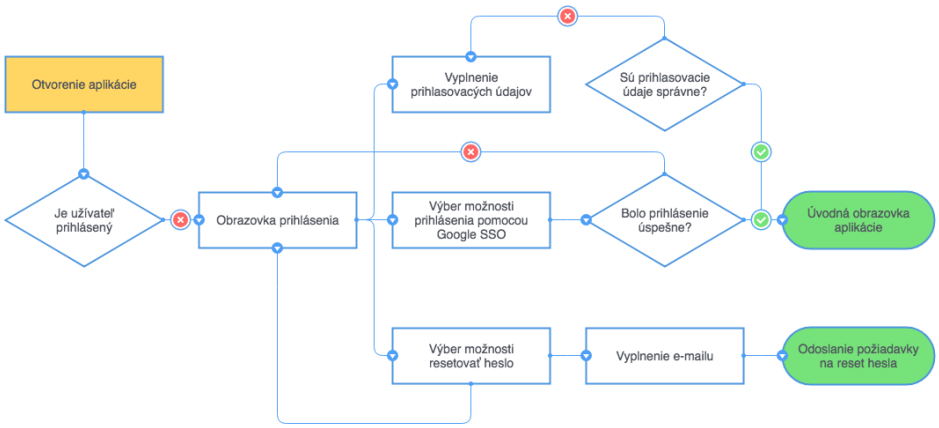
## 4.1 Diagram užívateľskej cesty

Diagram užívateľskej cesty dáva odpoveď na nasledujúce otázky: Čo musí užívateľ urobiť aby dosiahol, čo potrebuje? Aké kroky je potrebné vykonať pre naplnenie požadovaného cieľa [14]? Ide teda o graf, ktorý zobrazuje kroky, ktoré používateľ musí vykonať v rámci produktu alebo služby, aby splnil svoj cieľ [15]. V teórii môže existovať jeden enormný graf, ktorý popisuje celý produkt. V praxi ale dáva väčší zmysel vytvoriť niekoľko diagramov pre najčastejšie ciele používateľa. Cieľom tejto metódy je zamyslieť sa nad tým, čo všetko je potrebné pre naplnenie konkrétneho cieľa, aké obrazovky je potrebné používateľovi ukázať a aké rozhodnutia musí používateľ na svojej ceste učiniť. Diagram môže obsahovať aj vetvenie, ale nie je vhodné, aby toto vetvenie bolo príliš komplexné [14]. Taktiež je z tohto grafu jednoduché odpozorovať, či navrhnutá cesta nie je príliš dlhá a náročná a je tak možné tento problém vyriešiť už v ranom štádiu návrhu užívateľského rozhrania aplikácie.

Podkladom pre tvorbu diagramov užívateľskej cesty mi poslúžili funkčné požiadavky definované v predošlej kapitole. Diagramy uvedené nižšie popisujú vybranú podmnožinu funkčných požiadaviek. Neuvádzam samostatný graf pre každú definovanú funkčnú požiadavku, pretože vybraná podmnožina obsahuje zložitejšie akcie a chýbajúce akcie sú buď obsiahnuté v uvedených grafoch, alebo je rozšírenie uvedených grafov na ďalšie akcie celkom priamočiare.

Uzly grafu podfarbené žltou farbou predstavujú počiatočné uzly grafu. Koncové uzly grafu sú znázornené zaobleným obdĺžnikom, ktorého podfarbenie môže byť zelené, a vtedy ide koncový uzol, ktorý označuje stav, kedy užívateľ úspešne naplnil požadovaný cieľ, alebo je podfarbenie červené, a vtedy ide o neúspech pri naplnení požadovaného cieľa. Kosoštvorce reprezentujú vetvenie v danom mieste a vychádzajú z nich hrany označené červeným krížikom, čo znamená že podmienka nebola splnená, a hrany označené zelenou fajkou naopak znázorňujú vetvu, v ktorej podmienka splnená bola.

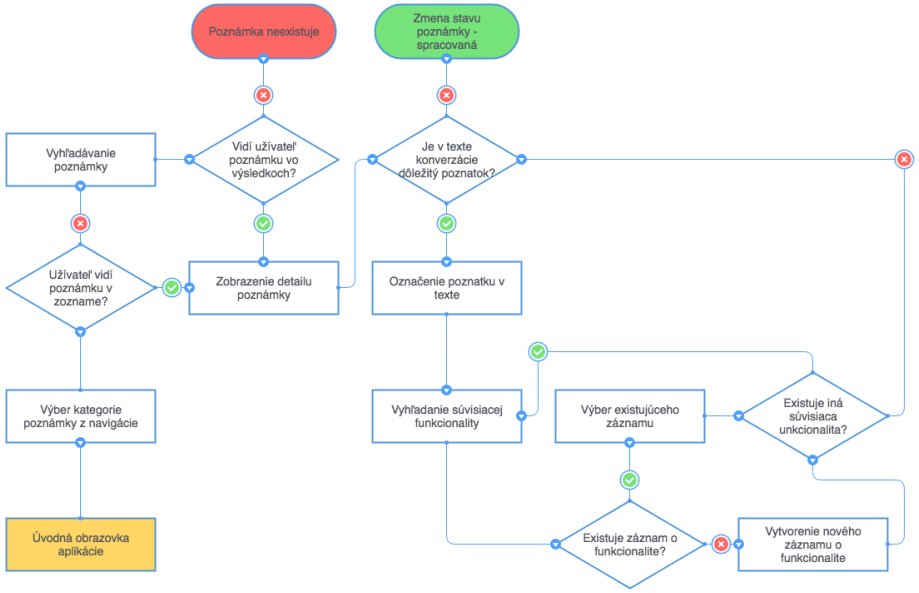
Pre prvý diagram užívateľskej cesty som si vybral akciu prihlásenia sa do aplikácie. Na obrázku 4.1. je možné vidieť, že existujú dva spôsoby, ako je




Obr. 4.1: Diagram užívateľskej cesty popisujúci proces prihlásenia sa do aplikácie

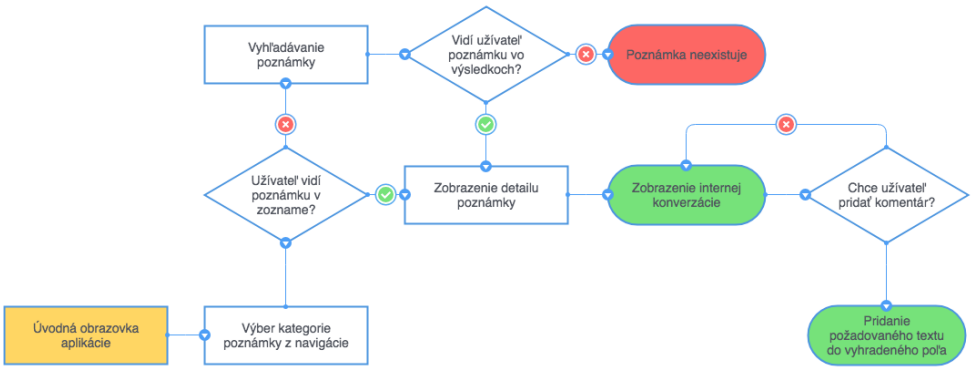
možné sa do aplikácie prihlásiť. Prvý spôsob vyžaduje zadanie správnej kombinácie e-mailovej adresy a hesla a druhý spôsob ponúka užívateľovi možnosť prihlásiť sa použitím jednotného prihlásenia existujúcim účtom do služieb Google. V prípade, že používateľ zabudol svoje heslo, môže vyplnením e-mailovej adresy požiadať o reset aktuálneho hesla a vytvorenie hesla nového. V takom prípade užívateľovi príde na uvedenú e-mailovú adresu správa obsahujúca odkaz na stránku, kde si užívateľ vytvorí nové heslo. Po prihlásení do aplikácie ostane užívateľ v aplikácii prihlásený, až kým sám nevykoná odhlásenie. Tento fakt je premietnutý do nasledujúcich diagramov, ktoré predpokladajú, že sa užívateľ po otvorení aplikácie ocitne na úvodnej obrazovke.

Obrázok 4.2 popisuje proces spracovania konkrétnej poznámky. V prvom rade potrebuje užívateľ nájsť požadovanú poznámku. V niektorých prípadoch môže spracovávať poznámky po poradí a vyberá jednotlivé poznámky zo zoznamu. Inokedy môže užívateľ chcieť spracovať konkrétnu poznámku a pre jej nájdenie využije vyhľadávanie. Ako je z grafu možné vidieť, vyhľadávaná poznámka nemusí z rôznych dôvodov existovať, a to je prípad, kedy užívateľ nemohol naplniť svoj cieľ. Ak užívateľ požadovanú poznámku nájde, otvorí si jej detail, ktorý typicky obsahuje vyjadrenie nejakého zo zákazníkov alebo časť konverzácie medzi zákazníkom a interným zamestnancom spoločnosti. Častokrát sú z tohto obsahu dôležité iba malé časti, ktoré majú súvislosť s určitou funkcionalitou. Užívateľ zodpovedný za spracovanie poznámky označí dôležitú časť textu, aby ho mohol prepojiť so záznamom o funkcionalite. Tento záznam môže, ale nemusí existovať. Ak záznam existuje, tak ho užívateľ vyberie, v opačnom prípade môže vytvoriť nový záznam. Označená časť textu môže byť dôležitá pre viacero funkcionalít, a tak sa proces priradenia opakuje. Taktiež môže text obsahovať viacero informácií, ktoré sú dôležité. Ak je




Obr. 4.2: Diagram užívateľskej cesty popisujúci proces výberu dôležitého poznatku a jeho priradenie k prislúchajúcej funkcionalite

každý dôležitý úryvok obsahu poznámky prepojený so všetkými záznamami o funkcionalite, ku ktorým patrí, je možné poznámke nastaviť stav indikujúci, že poznámka je už spracovaná.

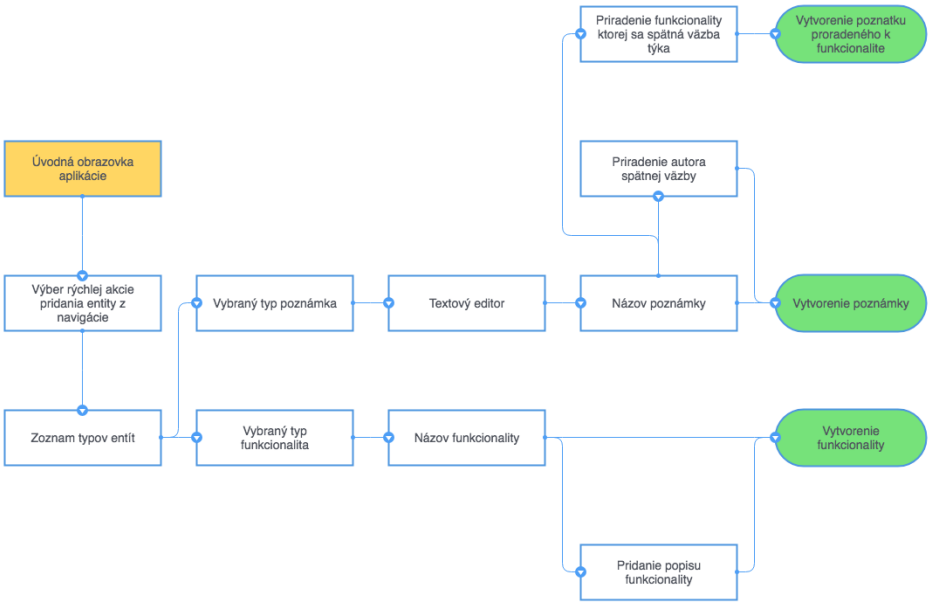



Obr. 4.3: Diagram užívateľskej cesty popisujúci proces pridania komentára k internej konverzácii týkajúcej sa konkrétnej poznámky

Diagram na obrázku 4.3 popisuje proces pridania komentára k internej

komunikácii v detaile vybranej poznámky. Postup, ktorým sa užívateľ dostane k požadovanej poznámke, je úplne zhodný ako v predošlom prípade, takže nie je nutné ho znova popisovať. Na detaile konkrétnej poznámky obsahuje priestor pre internú konverzáciu tímu súvisiacu s vybranou poznámkou. Ak táto konverzácia už existuje, môže si ju používateľ prečítať a prispieť svojím komentárom. Ak konverzácia ešte neexistuje, môže užívateľ túto konverzáciu započať pridaním prvého komentára.

Analogicky môže užívateľ pridať komentár k internej konverzácii alebo ju započať na detaile určitej funkcionality. Postup je takmer zhodný s postupom pre entitu poznámky s tým rozdielom, že vyhľadávanie funkcionality bude prebiehať na obrazovke zobrazujúcej zoznam funkcionalít.




Obr. 4.4: Diagram užívateľskej cesty popisujúci proces pridania novej poznámky alebo nového záznamu o funkcionalite

Poslednou pomerne netriviálnou akciou, ktorú môže užívateľ vykonať, je pridanie novej poznámky alebo pridanie nového záznamu o funkcionalite. V hlavnej navigácii bude existovať položka rýchleho pridania entity. Po jej vybraní dostane užívateľ na výber, či chce vytvoriť poznámku alebo funkcionalitu. V prípade, že si vyberie poznámku, bude mu zobrazený textový editor, v ktorom vytvorí obsah poznámky. V ďalšom kroku bude mať užívateľ možnosť túto poznámku pomenovať a priradiť k tejto poznámke autora spätnej väzby, ktorú táto poznámka obsahuje. Taktiež bude mať možnosť priamo vybrať funkcionalitu, ktorej sa daná poznámka týka. Ak užívateľ vyberie funkcionalitu, ku ktorej sa poznámka viaže, bude obsah poznámky automaticky prepo-

jený s funkcionalitou a poznámka bude označená ako spracovaná. V opačnom prípade sa vytvorí nová poznámka, ktorá pribudne v zozname všetkých nespracovaných poznámok.

V prípade, že užívateľ ako vytváranú entitu zvolí funkcionalitu, v nasledujúcom kroku musí zvoliť názov pre túto funkcionalitu a v textovom editore môže voliteľne vytvoriť popis pre danú funkcionalitu. Po vytvorení bude mať záznam stav *New idea*, v preklade *Nový nápad*, čo je východiskový stav v súčasnej webovej aplikácii, a bude zaradená do predvoleného vydania *Later*, v preklade *neskôr*, rovnako ako to je v súčasnej webovej aplikácii.

## 4.2 Informačná architektúra

Pod pojmom informačná architektúra si častokrát ľudia predstavia mapu stránok, hrubé náčrty užívateľského rozhrania, ktoré ale mnohí z nás poznajú pod anglickým pomenovaním *wireframes*, alebo si pod tým predstavujú návrh štruktúry hlavného menu aplikácie či webovej stránky. Je pravda, že všetko vymenované neodmysliteľne k informačnej architektúre patrí, no nepopisuje to informačnú architektúru ako takú. V literatúre zaoberajúcej sa tématikou informačnej architektúry môžeme nájsť rovno niekoľko definícií toho, čo to vlastne informačná architektúra je.

- Štruktúrny návrh zdieľaného informačného prostredia
- Syntéza organizácie, označovania, vyhľadávania a navigačných systémov v rámci digitálnych, fyzických a zmiešaných ekosystémov
- Umenie a veda o formovaní informačných produktov, podpore použiteľnosti, vyhľadateľnosti a porozumenia
- Rozvíjajúca sa disciplína zameraná na prinášanie princípov dizajnu a architektúry do digitálneho prostredia [16]
- Vytvorenie štruktúry webových stránok, aplikácií alebo iných produktov, ktoré používateľom umožňujú pochopiť, kde sa nachádzajú a kde nájdu požadované informácie vo vzťahu k ich súčasnej pozícii [17].

Informačná architektúra je základom efektívneho návrhu. Dokonca aj najsilnejší dizajn používateľského rozhrania môže ľahko zlyhať bez vhodnej informačnej architektúry [17]. Dobre navrhnutá informačná architektúra výrazným spôsobom prispieva k tomu, aby používatelia našli požadované informácie v čo najkratšom čase a s čo najmenším úsilím. Ak by bol proces hľadania potrebných informácií príliš komplikovaný alebo príliš zdĺhavý, hrozí reálne riziko, že používateľ aplikáciu opustí [17].

Informačná architektúra v sebe zahŕňa metódy z dvoch vedeckých oblastí, ktorými sú knihovníctvo a kognitívna psychológia. Kategorizácia a katalogizácia sú veľmi dôležité a hodnotné metódy z oblasti knihovníctva, používané

pri návrhu informačnej architektúry. Kategorizácia je metóda zoskupovania určitých entít na základe ich podobností, pričom pozorovaný atribút a miera podobnosti v danom atribúte určujú príslušnosť k danej kategórii. Katalogizácia je metóda, ktorá priraďuje entitám alebo obsahu metainformácie, ktoré napomáhajú rýchlejšej vyhľadateľnosti tohto obsahu v budúcnosti. Kognitívna psychológia je štúdium toho, ako ľudia vnímajú, učia sa, pamätajú si a premýšľajú o informáciách [18]. Väčšina pravidiel o návrhu užívateľského rozhrania, užívateľských interakcií a všeobecne všetkých pravidiel z oblasti návrhu digitálneho alebo fyzického produktu s cieľom uspokojiť potreby používateľa vychádza práve z kognitívnej psychológie. Je teda viac než jasné, že aj návrh informačnej architektúry využíva niektoré princípy z oblasti kognitívnej psychológie.

**Gestalt princíp**
: Skúma vizuálne vnímanie prvkov vo vzťahu k sebe navzájom [17]. Ukazuje, že ľudia majú tendenciu zjednocovať vizuálne prvky do skupín podľa ich blízkosti, podobnosti, kontinuity, uzavretia alebo symetrie [18].

**Mentálny model**
: Mentálne modely sú predpoklady, ktoré ľudia majú v mysli predtým, než interagujú s aplikáciou alebo webovou stránkou. Informácie sa ľahšie objavia, keď sú na mieste, ktoré zodpovedá očakávaniam používateľov o tom, kde by malo byť [17].

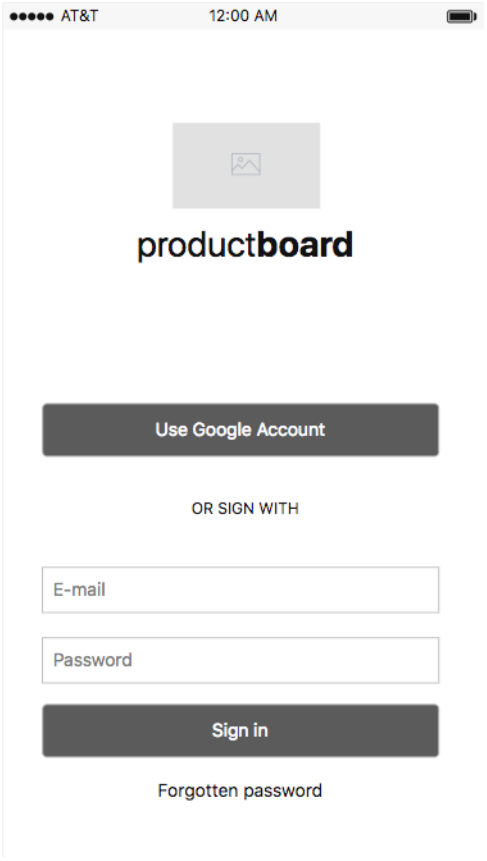
**Kognitívne zaťaženie**
: Kognitívne zaťaženie je množstvo informácií, ktoré dokáže osoba v danom okamihu spracovať. Keď dizajnéri uvažujú o kognitívnom zaťažení používateľa, pomáhajú mu zabrániť tomu, aby bol používateľ preťažený príliš veľkým množstvom informácií naraz [17].

**Vizuálna hierarchia**
: Vizuálna hierarchia priamo súvisí s čitateľnosťou obsahu. Jedným zo základných bodov, ktoré je potrebné zvážiť pri návrhu, sú skenovacie vzory. Pred čítaním stránky ľudia text skenujú, aby získali prehľad o tom, či je text pre nich zaujímavý. Najbežnejšie vzory skenovania sú vzory pripomínajúce písmená F a Z [17].

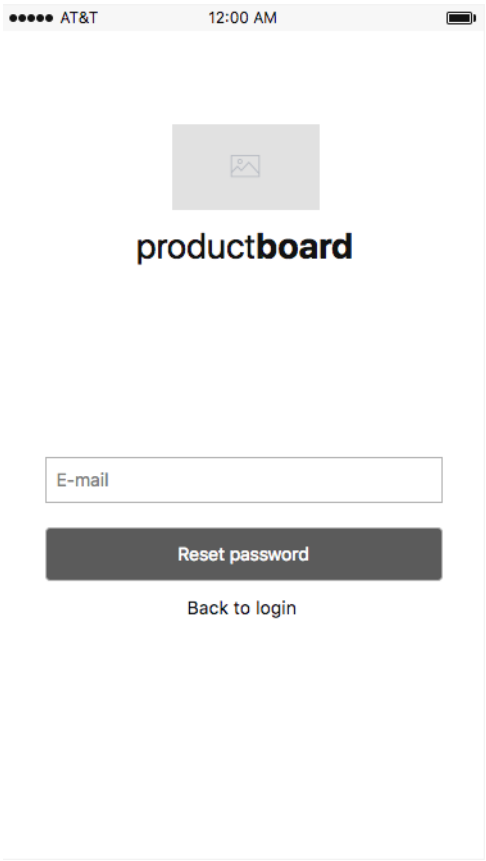
### 4.2.1 Návrh informačnej architektúry pre mobilnú aplikáciu

Diagramy užívateľskej cesty, ako aj ostatné poznatky nadobudnuté analýzou súčasného riešenia a analýzou užívateľov, mi poslúžili ako základ pre návrh informačnej architektúry pre mobilnú aplikáciu productboard. Pre popis informačnej architektúry som zvolil takzvaný *wireframe*. Ide o jednoduchý nákres zobrazujúci štruktúru konkrétnej obrazovky. Typicky býva čierno-biely a obsahuje iba jednoduché elementy a zástupné symboly pre budúce reálne prvky

na obrazovke. Typicky sa používa pri tvorbe rozhodnutí, v akom kontexte a na akom mieste bude umiestnený konkrétny obsahový element na obrazovke. Existuje niekoľko úrovní vernosti tohto nákresu. *Wireframe* s nízkou vernosťou poskytuje rýchly náhľad na to, ako bude obsah na obrazovke usporiadaný, naopak *wireframe* s vysokou vernosťou obsahuje aj detailné informácie a presný pohľad na to, ako bude obsah na obrazovke vyzerať a kde bude umiestnený.




Obr. 4.5: Návrh prihlasovacej obrazovky

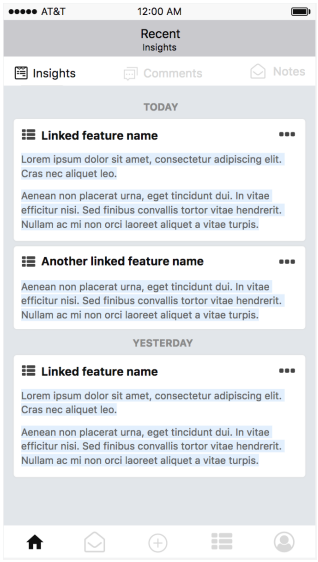



Obr. 4.6: Návrh obrazovky pre reset hesla

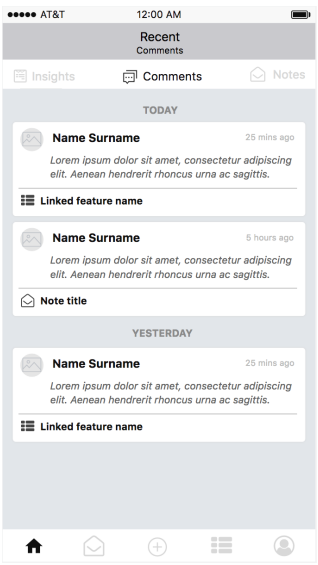
#### 4.2.1.1 Obrazovka prihlásenia

Na obrázku 4.5 je možné vidieť rozmiestnenie jednotlivých prvkov na obrazovke prihlásenia. V hornej časti je umiestnené logo aplikácie, aby užívateľ okamžite rozpoznal aplikáciu, do ktorej sa práve chystá prihlásiť. Nasleduje tlačidlo, ktoré užívateľovi umožňuje použiť jednotné prihlásenie do služieb Google. Následne je použitý textový oddeľovač, ktorý užívateľovi naznačuje, že existuje aj iná možnosť prihlásenia, a to konkrétne prihlásenie pomocou prihlasovacieho formulára, ktorý obsahuje textové pole pre zadanie e-mailovej

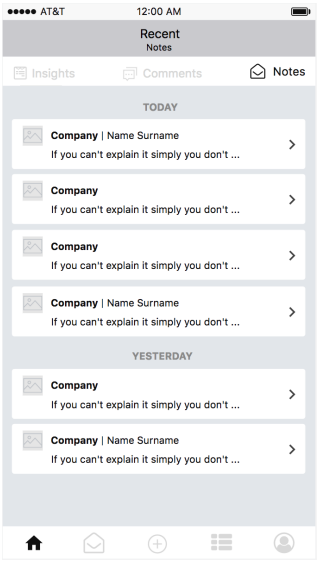
adresy, pod ktorou je účet zaregistrovaný, heslo k účtu a tlačidlo potvrdzujúce prihlásenie. Posledným prvkom je odkaz na obrazovku, kde si užívateľ môže svoje heslo obnoviť. Táto obrazovka je vyobrazená na obrázku 4.6. Taktiež obsahuje logo aplikácie v hornej časti, ďalej nasleduje formulár pre reset hesla obsahujúci textové pole pre zadanie e-mailovej adresy a potvrdzovacie tlačidlo a posledným elementom je odkaz, ktorý vráti užívateľa späť na prihlasovaciu obrazovku v prípade, že si na heslo spomenul alebo sa dostal na túto obrazovku omylom.




Obr. 4.7: Návrh obrazovky nedávnych insights




Obr. 4.8: Návrh obrazovky nedávnych komentárov




Obr. 4.9: Návrh obrazovky nedávnych poznámok

#### 4.2.1.2 Obrazovka nedávnych udalostí

Po úspešnom prihlásení sa užívateľ dostane do samotnej aplikácie, ktorá bude rozdelená na päť hlavných častí, ktoré budú prístupné pomocou hlavnej navigácie umiestnenej na spodnej hrane displeja, ako je to možné vidieť na obrázkoch 4.7 až 4.9. Keďže ide o veľmi dôležitý navigačný prvok, umiestnil som ho na spodnú stranu, čo je oblasť, ktorá je jednoducho obsiahnuteľná palcom jednej ruky v prípade, že užívateľ drží telefón na výšku. Užívateľ sa tak dokáže rýchlo a jednoducho pohybovať medzi hlavnými sekciami aplikácie. Navigácia obsahuje päť ikon reprezentujúcich nasledujúce sekcie:

1. Sekcia s prehľadom posledných udalostí
2. Sekcia poznámok

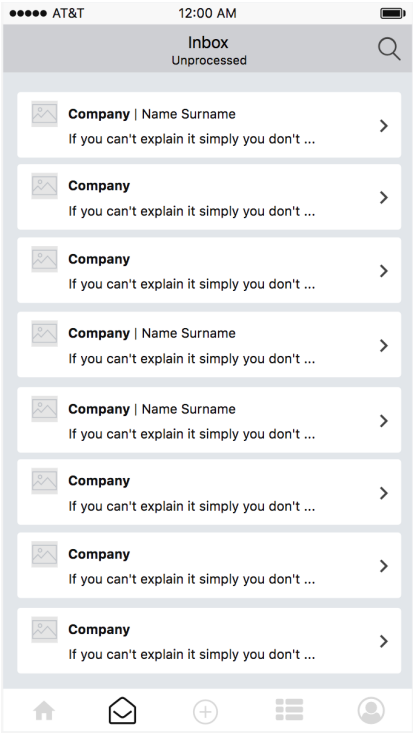
3. Rýchle pridanie novej entity
4. Sekcia týkajúca sa funkcionalít
5. Sekcia s užívateľským profilom a nastaveniami

Zvýraznená ikona na obrázkoch 4.7 až 4.9 signalizuje, že všetky tri obrazovky patria do prvej sekcie, ktorá sa zobrazí užívateľovi ako úvodná obrazovka po prihlásení do aplikácie. Táto sekcia obsahuje prehľad posledných udalostí, ktoré sa v aplikácii productboard udiali. Ako je možné na obrázkoch vidieť, táto sekcia je rozdelená do troch podsekcií reprezentovaných záložkami v hornej časti obrazovky a aktuálne vybraná záložka je vždy zvýraznená. Jednotlivé podsekcie budú obsahovať posledné *insights*, posledné komentáre v interných konverzáciách a naposledy pridané poznámky. Záznamy v obsahovej časti obrazovky budú rozdelené do skupín po dňoch, kedy daná udalosť nastala. Radenie udalostí na obrazovke som zvolil tak, aby najnovšie udalosti boli viditeľné vždy v hornej časti obrazovky a smerom nadol budú čím ďalej tým staršie udalosti. Tým zabezpečím, že najnovšie udalosti budú okamžite viditeľné po príchode do danej podsekcie bez nutnosti posúvania nadol.

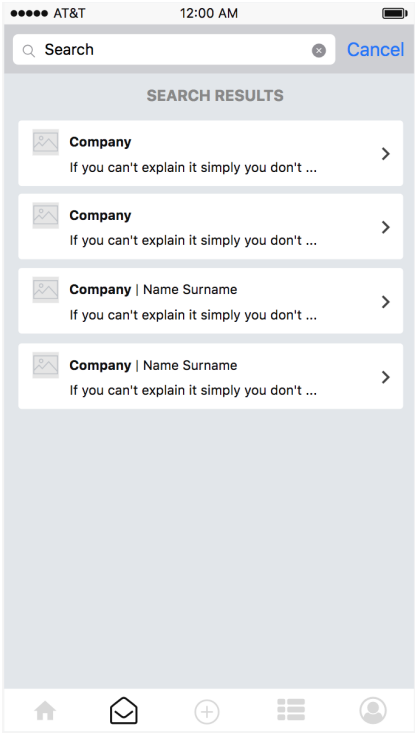
Jednotlivé položky v podsekcii *insights* budú obsahovať názov funkcionality, ku ktorej poznatok patrí, indikátor dôležitosti vyjadrenia a samotný text. V podsekcii komentárov bude štruktúra položky zoznamu nasledujúca: V hornej časti bude uvedený autor komentára a čas uverejnenia, nasledovať bude text komentára a nakoniec bude uvedená entita, ku ktorej bol komentár pridaný. Entita bude obsahovať ikonu, vďaka ktorej dokáže užívateľ okamžite rozlíšiť, či ide o poznámku alebo funkcionalitu, a za ikonou bude nasledovať názov danej entity. Položka v podsekcii poznámok bude obsahovať meno a názov spoločnosti autora spätnej väzby, ktorú poznámka obsahuje, a taktiež malý náhľad obsahu poznámky. Ak nebude meno a názov spoločnosti uvedené, bude poznámka označená ako anonymná.

#### 4.2.1.3 Obrazovka poznámok

Druhá položka hlavného menu reprezentuje sekciu poznámok. Po príchode do tejto sekcie sa užívateľovi zobrazí zoznam nespracovaných poznámok, ako je možné vidieť na obrázku 4.10. Štruktúra položky zoznamu je v tomto prípade úplne rovnaká ako to bolo v prehľade posledných poznámok u predošlej obrazovky. V záhlaví obrazovky sa na pravej strane nachádza ikona lupy, ktorá indikuje možnosť vyhľadávania v medzi jednotlivými poznámkami. Keďže nejde o kľúčovú akciu, ktorá by mala byť často využívaná, je táto ikona umiestnená v hornej časti obrazovky, kde nie je tak jednoducho dosiahnuteľná. Toto umiestnenie je taktiež konzistentné s väčšinou dobre známych a často používaných aplikácií, a z toho dôvodu by nájdenie tejto možnosti vyhľadávania užívateľom nemalo robiť žiadne problémy. Obrázok 4.11 zobrazuje samotnú obrazovku vyhľadávania. Táto obrazovka neobsahuje záhlavie s názvom sekcie, pretože



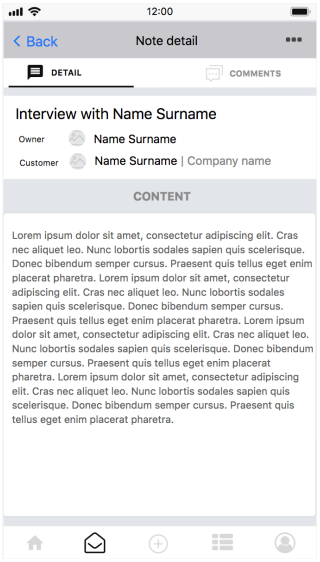

Obr. 4.10: Návrh obrazovky zobrazujúcej zoznam poznámok



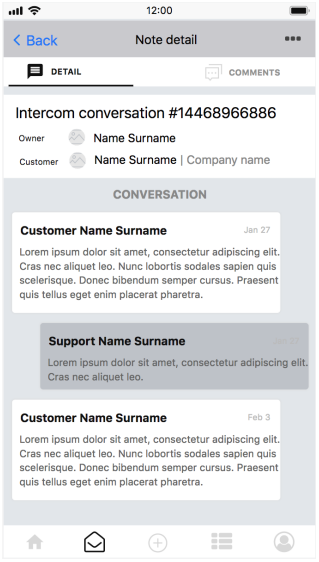

Obr. 4.11: Návrh obrazovky vyhľadávania medzi poznámkami

to považujem za nepotrebné, keďže z kontextu užívateľom vykonanej akcie by malo byť užívateľovi dostatočne jasné, že ide o obrazovku vyhľadávania medzi poznámkami. Zároveň tým ušetrím miesto vo vertikálnom smere obrazovky, čím dosiahnem možnosť zobraziť viac položiek, ktoré sú výsledkom vyhľadávania. Ako vyhľadávacie pole som zvolil jednotné vyhľadávacie pole, ktoré zodpovedá pokynom pre užívateľské rozhrania, ktoré uvádza na svojich webových stránkach samotný Apple ako autor platformy iOS [19], na ktorej bude táto aplikácia používaná. Pod vyhľadávacie pole som umiestnil textový oddeľovač, ktorý má užívateľovi jasne vysvetliť, že sa pod ním nachádza zoznam poznámok, ktoré vyhovujú vyhľadávanému textu. Každá z položiek zoznamu je odkazom na obrazovku obsahujúcu detail vybranej poznámky.

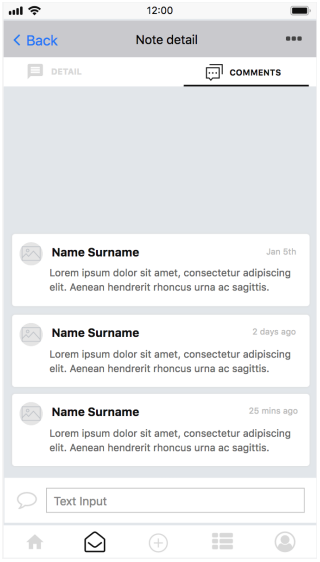
Obrazovka detailu poznámky bude rozdelená do dvoch podsekcií, ktoré sú reprezentované záložkami situovanými hneď po záhlaví obrazovky, ako je zjavné z obrázkov 4.12 až 4.14. Samotné záhlavie obrazovky obsahuje vľavo odkaz na návrat na predošlú obrazovku, rovnako ako to používa väčšina aplikácií na platforme iOS, a vpravo sa nachádza ikona horizontálne umiestnených troch bodiek, ktorá v súlade s dizajnovými pokynmi spoločnosti Apple znázorňuje možnosť zobraziť ďalšie akcie [20], ktoré je možné na obrazovke




Obr. 4.12: Návrh obrazovky detail poznámky - obsah




Obr. 4.13: Návrh obrazovky detail poznámky - klientska konverzácia

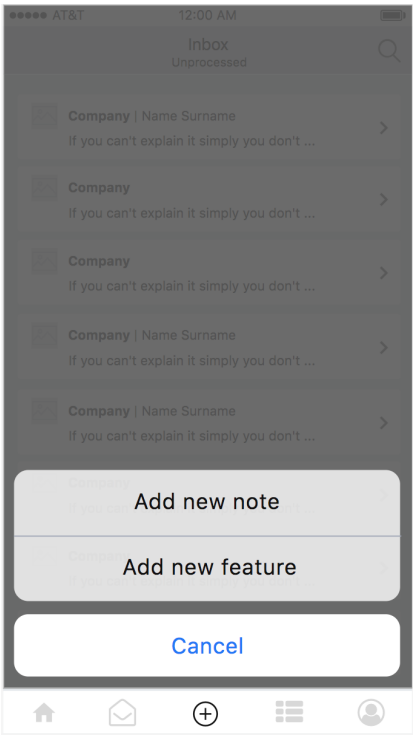



Obr. 4.14: Návrh obrazovky detail poznámky - komentáre

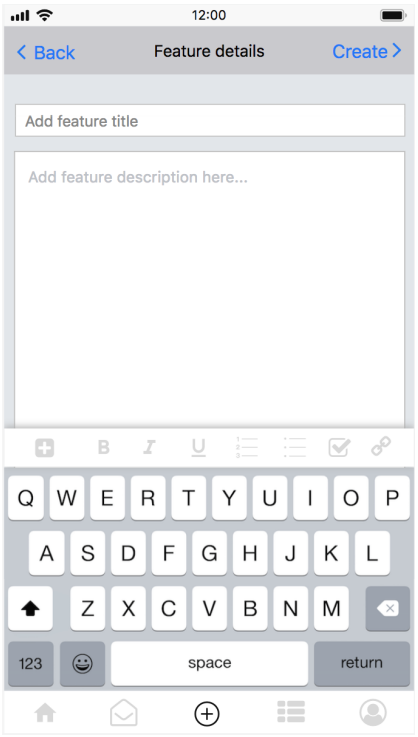
vykonať. Medzi tieto ďalšie akcie bude patriť možnosť označiť poznámku ako spracovanú a taktiež možnosť vymazať poznámku. Keďže ide o akcie, ktoré môžu byť nebezpečné, ak sú vykonané náhodne, je ich umiestnenie zámerne zvolené tak, aby neboli v priamom dosahu palca. Rovnako ako súčasná webová aplikácia productboard, aj mobilná aplikácia bude podporovať dva typy obsahu poznámky. Prvý typ obsahu je možné vidieť na obrázku 4.12 a ide o bežný štrukturovaný text. Druhý typ (obr. 4.13) je vo forme konverzácie, ktorá je typicky do productboardu importovaná prostredníctvom integrácie s aplikáciami tretích strán určenými na komunikáciu so zákazníkmi. V oboch prípadoch tomuto obsahu predchádza blok obsahujúci základné informácie o názve poznámky, autorovi poznámky a autorovi spätnej väzby, ktorú poznámka obsahuje. Druhá záložka obsahuje internú konverzáciu tímu týkajúcu sa konkrétnej poznámky. Jednotlivé položky konverzácie obsahujú meno autora, časový údaj o tom, kedy bol komentár pridaný a samotný text komentára. Jednotlivé komentáre sú zoradené od najnovších smerom zdola nahor. Najnovší komentár sa teda zaradí priamo nad textové pole, v ktorom užívateľ komentár napíše, čo je konzistentné chovanie s majoritou aplikácií, ktoré slúžia ako nástroj na komunikáciu.

#### 4.2.1.4 Obrazovka pridania entity

V poradí treťou položkou v hlavnej navigácii je ikona so symbolom plus, ktorá má symbolizovať možnosť rýchleho pridania poznámky alebo funkcionality. Po

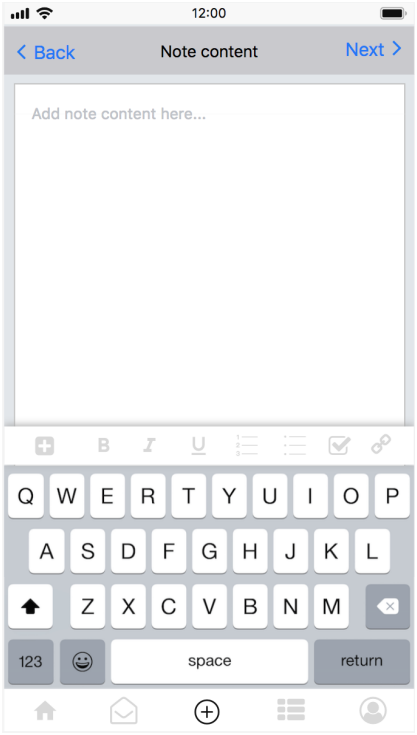



Obr. 4.15: Návrh obrazovky pre pridanie entity

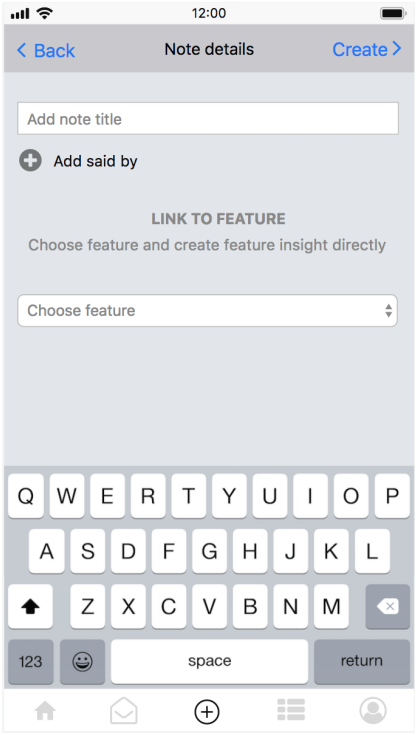



Obr. 4.16: Návrh obrazovky pre pridanie záznamu o funkcionalite

ťuknutí na túto položku sa užívateľovi zobrazí menu akcií (pozri obrázok 4.15), v ktorom si užívateľ vyberie, či chce pridať funkcionalitu alebo poznámku. Okrem jednotlivých možností menu obsahuje aj možnosť zrušiť akciu, čo vyvoláva u užívateľa dôveru, že akcia nie je nezvratná a môže túto akciu kedykoľvek opustiť. Po výbere možnosti pridania funkcionality užívateľ uvidí obrazovku s textovým poľom pre názov funkcionality a textovou oblasťou pre možnosť vložiť formátovaný text ako popis danej funkcionality. Umiestnenie poľa pre názov funkcionality ako prvého na obrazovke je zámerné, pretože ide o povinný údaj potrebný k vytvoreniu záznamu. Z praktickej skúsenosti s používaním aplikácie productboard viem, že užívateľ môže častokrát rýchlo vytvoriť záznam s nápadom na funkcionalitu a samotný popis nemusí byť pri dobre zvolenom pomenovaní potrebný alebo pre užívateľa vyplnenie popisu nie je v danej chvíli až tak dôležité a dostane sa k vyplneniu neskôr v situácii, kedy to bude vhodnejšie. Pre možnosť popis ľubovoľne formátovať bude užívateľovi slúžiť lišta obsahujúca bežné formátovacie akcie, ktorá bude zobrazená hneď nad systémovou klávesnicou. V záhlaví obrazovky je opäť ponúknutá možnosť návratu na predošlú obrazovku, čím zároveň zruší akciu pridania funkcionality. Na pravej strane je naopak akcia, ktorou používateľ potvrdí vytvorenie záznamu. Umiestnenie v hornej časti obrazovky je v tomto prípade zvolené




Obr. 4.17: Návrh obrazovky pre pridanie obsahu poznámky




Obr. 4.18: Návrh obrazovky pre pridanie doplňujúcich údajov k poznámke

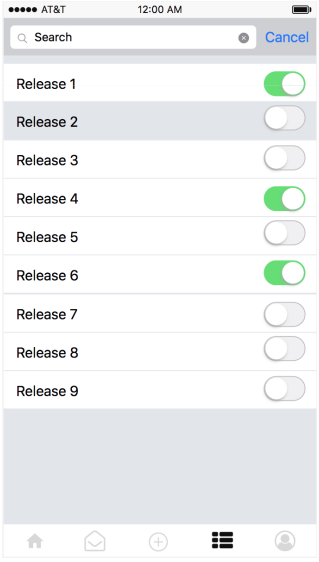
z dôvodu, aby užívateľ omylom akciu nepotvrdil skôr než potrebuje, a keďže pre vykonanie akcie potrebuje natiahnuť prst pravej ruky alebo použiť druhú ruku, tak si je dostatočne vedomý tejto akcie.

V prípade, že si užívateľ v menu akcií vyberie pridanie novej poznámky, uvidí obrazovku, ktorá je na obrázku 4.17. V prípade poznámky je oveľa hodnotnejší samotný obsah poznámky a názov v tomto prípadne nie je až tak dôležitý. Z výskumu vyplýva, že užívateľ si chce častokrát rýchlo poznamenať postreh z aktuálne prebiehajúceho rozhovoru a vyžadovať po ňom v tomto momente akékoľvek iné informácie by bolo zbytočne neefektívne a tento proces by iba zdržiavalo. Aj v tomto prípade môže užívateľ zadávať formátovaný text vrátane odrážkového zoznamu, čo je spôsob, ktorý je pomerne rozšírený pri tvorbe rýchlych poznámok. Po tom, čo si užívateľ zaznamená všetky potrebné poznámky, sa pomocou akcie na pravej strane záhlavia obrazovky dostane na ďalší krok (pozri obrázok 4.18), v ktorom môže ľubovoľne vyplniť doplňujúce údaje, ako je názov poznámky, taktiež môže priradiť meno človeka, od ktorého pochádza spätná väzba v obsahu poznámky, a v neposlednom rade môže výberom náležitej funkcionality priradiť obsah novej poznámky priamo k vybranej funkcionalite, tým sa poznámka automaticky vytvorí so stavom

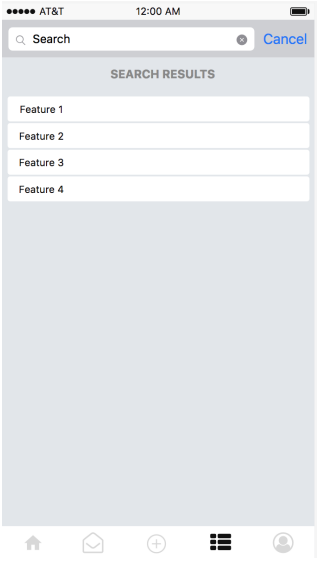
spracovaná.

#### 4.2.1.5 Obrazovka funkcionalít


Obr. 4.19: Návrh obrazovky so zoznamom funkcionalít



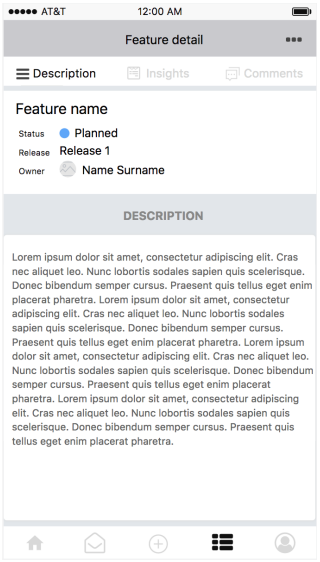

Obr. 4.20: Návrh obrazovky pre výber zobrazených vydaní



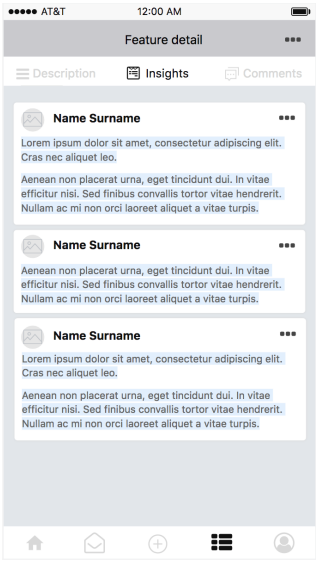

Obr. 4.21: Návrh obrazovky vyhľadávanie funkcionality

Na obrázku 4.19 sa nachádza obrazovka so zoznamom funkcionalít. Tieto záznamy sú zoskupené podľa jednotlivých vydaní, do ktorých patria a každé takéto vydanie má svoju vlastnú záložku. Podľa výsledkov z analýzy je pre užívateľov zoskupenie podľa vydania omnoho dôležitejšie ako zoskupenie podľa jednotlivých stavov, ktoré môže funkcionalita nadobudnúť, a preto sa bude toto zoskupenie zobrazovať ako prednastavené. Pomocou kontextového menu, ktoré je možné vyvolať ťuknutím na ikonu v záhlaví obrazovky, bude možné zoskupiť jednotlivé záznamy podľa stavu, ktorý môžu jednotlivé funkcionality nadobudnúť. Aktuálne zvolené zoskupenie bude zobrazené priamo v záhlaví obrazovky. Taktiež je veľmi pravdepodobné, že užívateľ nepotrebuje vidieť prehľad všetkých vydaní, ktoré boli vytvorené a naplánované, ale s vysokou pravdepodobnosťou budú užívateľa zaujímať iba vydania, ktoré sú pomerne aktuálne. Preto sa užívateľ prostredníctvom kontextového menu bude môcť dostať do nastavení zobrazených vydaní, ktoré je možné vidieť na obrázku 4.20. V týchto nastaveniach môže užívateľ aktivovať alebo deaktivovať vydania, ktoré budú zobrazené, a pre rýchlejšiu možnosť nájsť požadované vydanie bude užívateľovi k dispozícii aj vyhľadávanie medzi vydaniami. Ak má užívateľ nastavené zoskupovanie podľa stavov, bude možné totožným spôsobom vybrať zobrazené stavy. V prípade, že užívateľa zaujíma konkrétna funkcionalita a

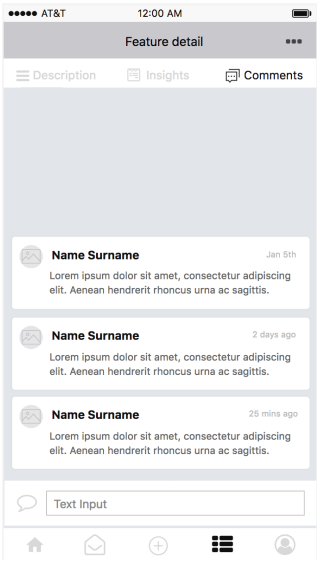
netuší, v akom vydaní alebo pod akým stavom by ju mal hľadať, môže využiť vyhľadávanie (pozri obrázok 4.21) prístupné prostredníctvom ikony lupy v záhlaví, ktoré vyhľadáva výsledky v celej množine dostupných záznamov.




Obr. 4.22: Návrh obrazovky detail funkcionality - popis




Obr. 4.23: Návrh obrazovky detail funkcionality - insights

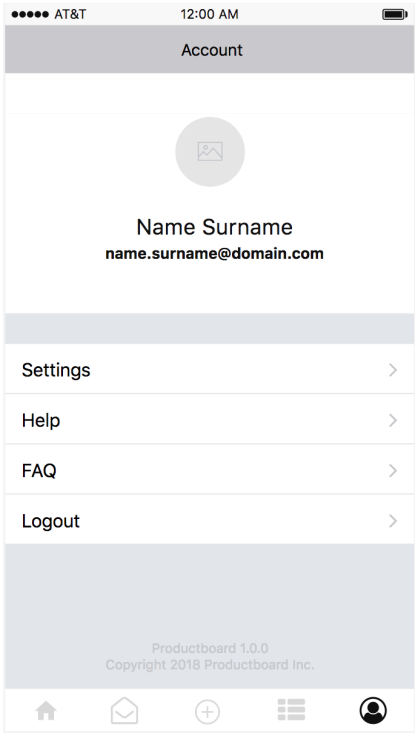



Obr. 4.24: Návrh obrazovky detail funkcionality - komentáre

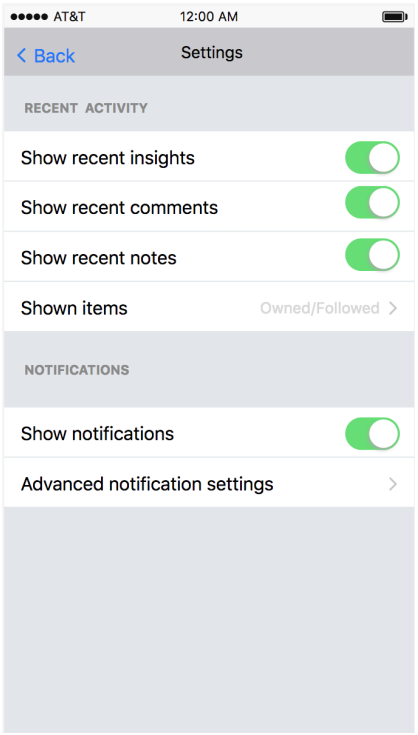
Každý záznam o funkcionalite, či už v záložke konkrétneho vydania alebo v zozname výsledkov vyhľadávania, je odkazom na detail konkrétnej funkcionality. V záhlaví sa opäť nachádza ikona zobrazujúca kontextové menu, ktoré obsahuje rýchle akcie, ako je zmena stavu funkcionality, zmena vydania, do ktorého funkcionalita patrí alebo zmazanie príslušného záznamu o funkcionalite. Detail funkcionality môže obsahovať pomerne veľa informácií, ktoré nie je možné rozumne umiestniť na jednu obrazovku tak, aby bola táto obrazovka dostatočne prehľadná. Preto som identifikoval tri skupiny informácií a každá z týchto troch skupín bude zobrazená v samostatnej záložke v detaile. Prvou skupinou a zároveň prvou záložkou sú základné informácie o funkcionalite a jej popis. Táto záložka je zobrazená na obrázku 4.22 a obsahuje základné informácie o funkcionalite, ako je jej názov, stav, príslušnosť k nejakému vydaniu a meno vlastníka. Po týchto informáciách nasleduje blok obsahujúci popis funkcionality. Druhá záložka obsahuje jednotlivé postrehy a dôležité vyjadrenia zákazníkov, ktoré sa týkajú danej funkcionality. štruktúra záznamu je veľmi podobná ako na obrazovke úvodnej obrazovke s tým rozdielom, že záznam neobsahuje názov funkcionality, ku ktorej je priradený, pretože je to z kontextu zrejmé, a naopak obsahuje meno zákazníka, ktorého vyjadrenie obsahovalo tento obsah. Poslednou záložkou v detaile funkcionality je záložka s internou tímovou konverzáciou k danej funkcionalite. Táto záložka je úplne

zhodná so záložkou komentárov v detaile poznámky.

#### 4.2.1.6 Obrazovka profilu a nastavení




Obr. 4.25: Návrh obrazovky profilu užívateľa




Obr. 4.26: Návrh obrazovky nastavení

Poslednou položkou v hlavnom menu aplikácie je ikona zobrazujúca siluetu človeka, ktorá má symbolizovať profil užívateľa, ktorý je zobrazený na obrázku 4.25. Okrem základných informácií o práve prihlásenom užívateľovi, ako je náhľad fotografie, meno a emailová adresa, sa tu nachádzajú dôležité odkazy, ktoré užívateľa dostanú do nastavení aplikácie, na oficiálne stránky obsahujúce rady a návody na používanie aplikácie, ale aj možnosť napísať správu na zákaznícku podporu produktu productboard. Poslednou možnosťou je možnosť odhlásiť sa z aplikácie. V spodnej časti je možné nájsť informácie o aktuálnej verzii aplikácie.

Platforma iOS používa aplikáciu Nastavenia ako centrálne miesto, kde je možné okrem konfigurácie samotného telefónu meniť aj konfiguráciu pre jednotlivé aplikácie. Keďže užívateľ musí najprv opustiť aktuálnu aplikáciu, aby sa dostal do aplikácie Nastavenia, kde bude mať možnosť konfigurovať nastavenia danej aplikácie, je vhodné poskytnúť užívateľovi v rámci aplikácie odkaz, ktorý ho dostane do aplikácie Nastavenia priamo do sekcie nastavení

týkajúcich sa vybranej aplikácie [21]. Jednotlivé nastavenia, ktoré je možné ovplyvňovať pre mobilnú aplikáciu productboard, je možné vidieť na obrázku 4.26. Tieto nastavenia sú rozdelené do dvoch sekcií, pretože sa týkajú konfigurácie dvoch rozdielnych častí aplikácie. Prvá sekcia umožňuje užívateľovi zvoliť, aké informácie sa mu zobrazia na úvodnej obrazovke. Užívateľ môže vypnúť zobrazenie niektorých záložiek a taktiež môže zvoliť, či sa zobrazované informácie týkajú všetkých záznamov alebo iba záznamov, u ktorých je pridelený ako majiteľ záznamu, prípadne daný záznam sleduje. Druhá sekcia obsahuje nastavenie notifikácii. V tejto sekcii môže užívateľ vypnúť všetky notifikácie alebo sa môže dostať na detailný zoznam jednotlivých notifikácií a tam vypínať a zapínať notifikácie pre vybrané udalosti v aplikácii productboard.

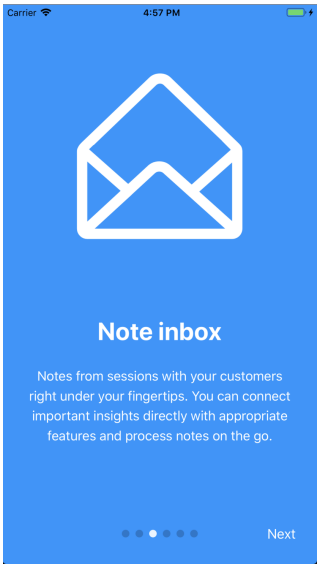
## 4.3 Používateľské interakcie

Návrh interakcií medzi užívateľom a aplikáciou je pomerne komplexná oblasť v tvorbe užívateľského rozhrania. Interakcie napomáhajú užívateľovi čo najlepšou cestou dosiahnuť svoj cieľ. Nie je to ale zďaleka len o tom poskytnúť užívateľovi ovládacie prvky pre akcie, ktoré potrebuje vykonať. Rovnako dôležité je poskytnúť pomocou správnych interakcií užívateľovi prehľad o tom, čo môže pomocou daného užívateľského rozhrania dosiahnuť, a adekvátnym spôsobom udržiavať povedomie užívateľa o tom, aký je súčasný stav akcie alebo cieľa, ktorý potrebuje naplniť a aké ďalšie kroky potrebuje vykonať pre naplnenie požadovaného cieľa. Tieto interakcie by mali byť efektívne, jednoducho zrozumiteľné a konzistentné, aby užívateľ vždy vedel, aký krok má vykonať a čo je výsledkom daného kroku, a pri nových akciách nepotreboval premýšľať nad ich vykonaním, pretože interakcia užívateľského rozhrania sa zhoduje s interakciou, ktorú už užívateľ pozná z predošlého vykonania podobnej akcie.

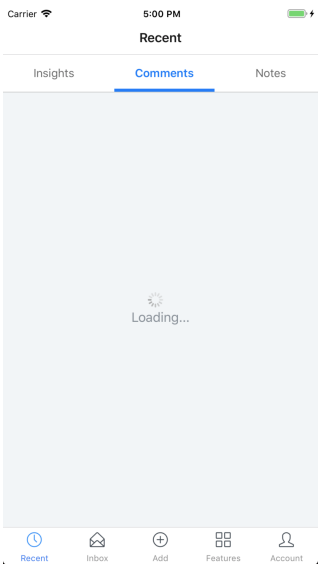
Z hľadiska interakcií som pri návrhu použil niekoľko techník, ako urobiť interakciu medzi užívateľským rozhraním aplikácie a samotným užívateľom čo najjednoduchšiu a najpríjemnejšiu. Hneď po prihlásení do aplikácie som navrhol jednoduchý element (pozri obrázok 4.27), ktorý oboznámi nového užívateľa s možnosťami a funkciami mobilnej aplikácie pomocou jednoduchého posuvníka malého počtu obrazoviek, pričom každá obrazovka v krátkosti popisuje jednu konkrétnu funkciu aplikácie. Týmto spôsobom získa užívateľ jednoducho a rýchlo prehľad o tom, čo môže od aplikácie očakávať a zároveň získa prehľad terminológie, ktoré aplikácia v súvislosti s danými funkciami používa.

Ako som spomenul, je viac ako vhodné, ak užívateľské rozhranie dáva užívateľovi informácie o tom, v akom stave sa nachádza posledná akcia vykonaná užívateľom, a to samozrejme aj v prípade, že akcia nedopadla podľa očakávaní. V prípade mnou navrhovanej aplikácie sa u akcií, ktoré zo svojej

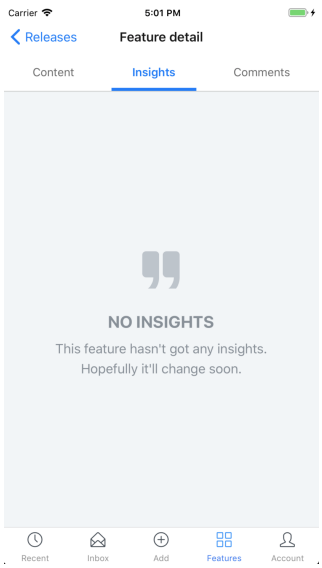
povahy nemôžu byť vykonané okamžite, ako napríklad získavanie dát zo servera, zobrazí element indikujúci prebiehajúce načítavanie dát (obrázok 4.28). Ide o bežne používaný spôsob, ktorým sa užívateľovi dáva najavo, že akcia, ktorú vykonal, stále prebieha. Po tom, čo aplikácia získa potrebné údaje, vykreslí požadovaný obsah. V komunikácii medzi aplikáciou a serverom ale môže dôjsť ku chybe alebo sa v odpovedi žiadne dáta nenachádzajú, pretože žiadne dáta nie sú uložené na serveri. V prípade, že došlo ku chybe v komunikácii, bude o tejto skutočnosti užívateľ oboznámený pomocou dialógového okna, ako je znázornené na obrázku 4.30. Ak sa naopak vráti zo servera odpoveď, ktorá relevantne neobsahuje žiadne dáta, užívateľovi sa na obrazovke zobrazí informácia o tom, že pre danú obrazovku neexistujú žiadne údaje, ktoré je možné vykresliť, ako je znázornené na obrázku 4.29.




Obr. 4.27: Ukážka vstupného prehľadu funkcií aplikácie



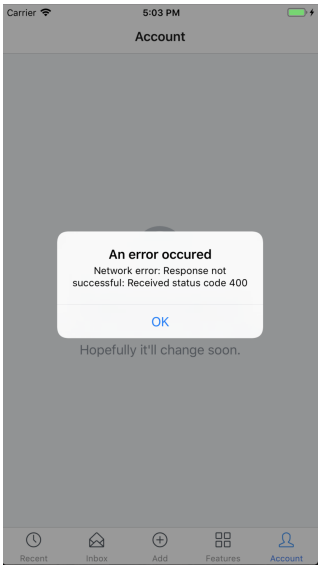

Obr. 4.28: Ukážka interakcie pri priebehu akcie



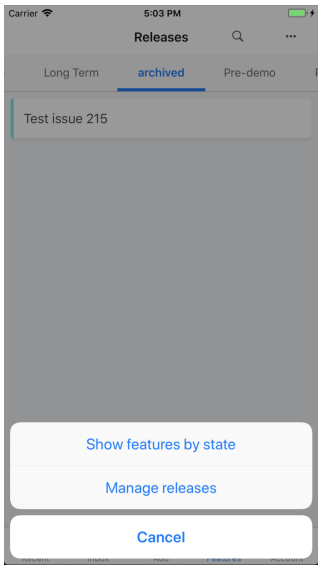

Obr. 4.29: Interakcia obrazovky v prípade chýbajúcich informácií pre vykreslenie

Mobilná platforma poskytuje elementy, ktoré je možné zobraziť priamo nad obsahom aktuálnej obrazovky. Medzi tieto elementy patrí dialógové okno (obrázok 4.30), menu s výberom akcií (obrázok 4.31) a modálne okno (obrázok 4.32). Pre zavedenie lepšej konzistencie interakcií v rámci mnou navrhovanej aplikácie používam tieto elementy pre špecifické typy interakcií s užívateľom, aby užívateľ dokázal už zo samotného použitia elementu rozlíšiť, aké informácie mu aplikácia prezentuje alebo aké kroky sú po užívateľovi vyžadované. Dialógové okno je použité výhradne v prípadoch, kedy je užívateľ informovaný o chybovom stave v aplikácii. Ide o jeden z najznámejších použití dialógového okna

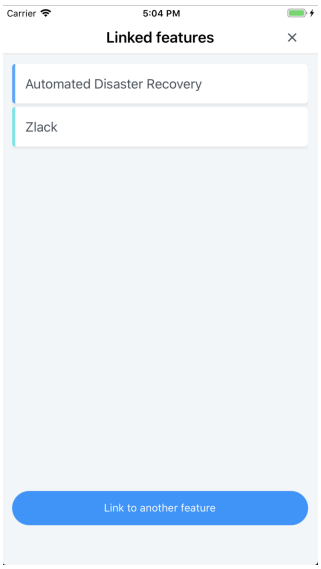
vôbec. Naopak, ak obrazovka ponúka nejaké akcie ukryté pod ikonou troch horizontálnych bodiek v pravej časti záhlavia obrazovky, ťuknutím na ikonu sa vyvolá menu s výberom akcií.




Obr. 4.30: Dialógové okno signalizujúce chybu v aplikácii




Obr. 4.31: Menu pre výber požadovanej akcie




Obr. 4.32: Modálne okno pre vykonanie akcie v kontexte aktuálnej obrazovky

Modálne okno využívam v situáciách, keď je pre naplnenie určitého užívateľovho cieľa potrebné vykonať nejaké ďalšie kroky vedúce k naplneniu cieľa. Napríklad v prípade, že chce užívateľ prepojiť dôležitý poznatok z klientskej konverzácie s relevantnou funkcionalitou, je potrebné, aby užívateľ vybral relevantnú funkcionalitu a taktiež zvolil dôležitosť vybraného poznatku pre autora výroku.

## 4.4 Vizuálny štýl

Vizuálny štýl aplikácie býva vo väčšine prípadov doménou grafika alebo výtvarníka s určitou mierou estetického cítenia, a preto som do tejto oblasti veľmi nezachádzal. Hlavným cieľom z vizuálneho hľadiska pre mňa bolo čo najviac sa priblížiť vizuálnemu štýlu súčasnej webovej aplikácie productboard, a to z dôvodu, aby si užívateľ dokázal túto aplikáciu asociovať s aplikáciou productboard na webe aj bez toho, aby videl v aplikácii logo spoločnosti alebo jej názov. Najväčší priestor pre priblíženie sa webovej aplikácii považujem pri použití rovnakej farebnej škály a aplikovaním tejto farebnej škály na prvky rovnakým alebo podobným spôsobom ako to je u webovej aplikácie. Vo vizualizáciách aplikácie na obrázkoch 4.27 až 4.32 je možné vidieť, že nepoužívam

rovnaké písmo a ikony ako webová aplikácia, čo je na prvý pohľad vizuálne nekonzistentné. Dôvodom je, že tieto elementy nemôžem v rámci tejto práce využiť z dôvodu autorských práv, ale vo finálnej produkčnej aplikácii budú samozrejme použité rovnaké ikony, aby vyvolávali u užívateľov mobilnej aplikácie rovnaké asociácie ako pri využívaní aplikácie v internetovom prehliadači.

Kapitola 5

# Realizácia

V tejto kapitole najprv popíšem a porovnám rôzne prístupy k vývoju mobilnej aplikácie, následne detailnejšie popíšem prístup a technológie, ktoré som zvolil a taktiež aj dôvody, ktoré ma k tomuto rozhodnutiu viedli. Ďalej popíšem techniky, postupy a nástroje, ktoré som zvolil pri implementácii mobilnej aplikácie a nakoniec popíšem samotnú implementáciu.

## 5.1 Analýza technológií

V súčasnej dobe existuje viacero možných prístupov, ako implementovať aplikáciu pre mobilné zariadenie. Tieto prístupy sa od seba pomerne radikálne odlišujú vo vlastnostiach a parametroch finálnej aplikácie a každý z týchto prístupov má svoje výhody a nevýhody, ktoré je potrebné zohľadniť pri výbere správneho prístupu alebo technológie pre konkrétny prípad mobilnej aplikácie. S ohľadom na použité technológie a rozdiely v architektúre sa dajú tieto prístupy rozdeliť do nasledujúcich kategórií:

- Natívne aplikácie
- Kompilované aplikácie pre viacero platformiem
- Interpretované aplikácie pre viacero platformiem
- Hybridné aplikácie
- Progresívne webové aplikácie

### 5.1.1 Natívne aplikácie

Natívna mobilná aplikácie je taká aplikácia, ktorá je implementovaná pre zariadenie so špecifickým operačným systémom v programovacom jazyku, ktorý je kompatibilný s týmto operačným systémom. Medzi najrozšírenejšie operačné systémy pre mobilné zariadenia v dnešnej dobe patrí operačný systém

iOS, ktorý je určený výhradne pre mobilné zariadenia od výrobcu Apple. Podporované programovacie jazyky pre túto platformu sú Objective-C, ktorý je už starší, ale ešte stále je v ňom možné programovať pre platformu iOS, a novším jazykom, ktorý iOS podporuje je jazyk Swift. V prípade, že sa nejedná o zariadenie od výrobcu Apple, je najrozšírenejším a najpoužívanejším operačným systémom pre mobilné zariadenia systém Android vyvinutý spoločnosťou Google. V prípade Androidu sú podporované programovacie jazyky taktiež dva. Prvým a straším podporovaným jazykom je jazyk Java od spoločnosti Oracle a druhým, pomerne novým jazykom je jazyk Kotlin od spoločnosti JetBrains. Ako samotný názov tejto sekcie napovedá, natívne aplikácie sú primárne určené pre mobilné zariadenia, a práve preto majú vo väčšine prípadov oproti ostatným prístupom výrazné výhody. No nájde sa aj zopár nevýhod, ktoré sú v tomto prípade skôr ekonomické než technologické.

#### 5.1.1.1 Výhody

Natívne aplikácie majú na rozdiel od iných prístupov všetky technické výhody a ďalšie výhody vyplývajúce z toho, že tento spôsob vývoja mobilných aplikácií funguje od počiatkov samotných chytrých mobilných zariadení. Medzi hlavné výhody patria:

- Najlepší výkon
- Najpútavejšie používateľské interakcie
- Najvyššia miera bezpečnosti
- Najvyššia spoľahlivosť
- Priama podpora hardvéru a aplikačných rozhraní
- Veľké množstvo dokumentácie a materiálov k štúdiu
- Veľké množstvo knižníc vývojárov tretích strán
- Veľká komunita zaoberajúca sa natívnym vývojom mobilných aplikácií

#### 5.1.1.2 Nevýhody

Ako som už spomenul vyššie, po technickej stránke neexistujú žiadne nevýhody tohto riešenia. V prípade, že sa vyžaduje aplikácia, ktorá má fungovať na oboch najrozšírenejších platformách, vynára sa niekoľko nevýhod práve z dôvodu, že každá z uvedených platforiem poskytuje rozdielne rozhrania a podporuje iné programovacie jazyky. Medzi najväčšie nevýhody patria nasledovné:

- Duplikácia rovnakej logiky v dvoch rozdielnych programovacích jazykoch
- Potreba nezávislých tímov programátorov pre každú platformu

- Vývoj pre obe platformy trvá dvojnásobok času a vyžaduje dvojnásobok prostriedkov
- Komplikovaná synchronizácia verzií naprieč platformami (vývoj pre rozdielne platformy zvyčajne netrvá rovnakú dobu)
- Nutnosť riadiť a testovať aplikáciu zvlášť pre každú platformu

### 5.1.2 Kompilované aplikácie pre viacero platforiem

V tomto prípade ide o aplikácie, ktoré sa vyznačujú tým, že ich je možné naprogramovať v jednom programovacom jazyku a následne je tento zdrojový kód skompilovaný do podoby, ktorú zariadenie pozná a dokáže tento kód vykonať. Zámerne som nespomenul presnú podobu skompilovaného kódu, pretože v tomto prípade sa skompilovaná podoba môže líšiť pre jednotlivé technológie, ale aj pre cieľové platformy. Medzi hlavných predstaviteľov kompilovaných aplikácii pre viacero platforiem patria Xamarin od spoločnosti Microsoft a Flutter od spoločnosti Google.

#### 5.1.2.1 Xamarin

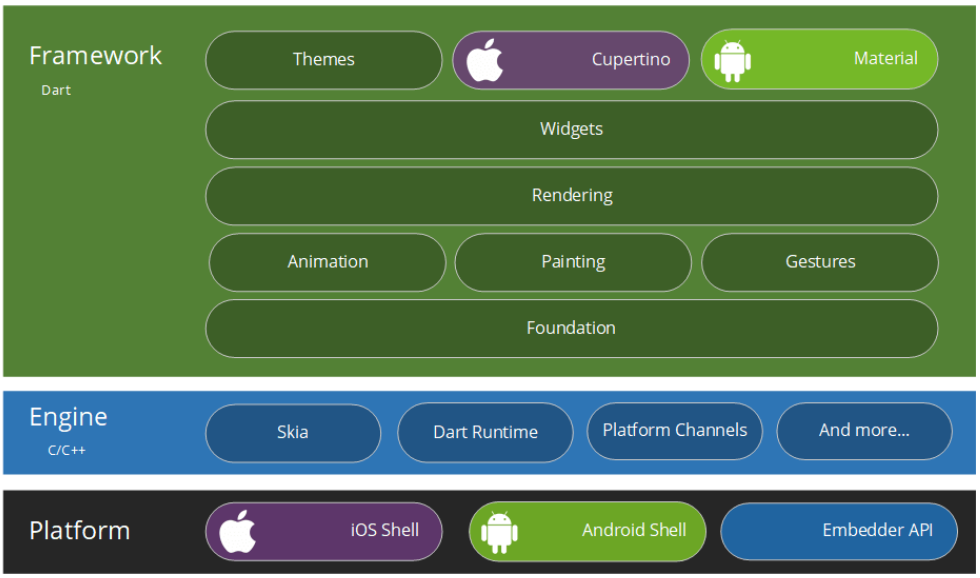
Xamarin umožňuje napísať aplikácie v jazyku C# a zdieľať rovnaký kód na viacerých platformách, pričom ale samotná implementácia v každom systéme je veľmi odlišná [22]. Kompilácia zdrojového kódu a jeho následné vykonanie na cieľovom zariadení sa pre jednotlivé platformy pomerne značne odlišuje.

V prípade platformy iOS je C# prostredníctvom *ahead-of-time* (AOT) kompilácie skompilovaný do nízkoúrovňového jazyka inštrukčnej sady procesoru ARM. Spolu so zdrojovým kódom aplikácie je súčasťou kompilácie aj .NET framework, z ktorého sú počas fázy linkovania odstránené prebytočné a nepoužité triedy, čo napomáha k redukcii dátovej veľkosti výslednej aplikácie. Niektoré konštrukcie jazyka C# ale nie sú v tomto prípade k dispozícii, pretože spoločnosť Apple nepodporuje na platforme iOS generovanie kódu v prostredí *runtime* [22].

Pre platformu Android je situácia značne rozdielna. Jazyk C# je skompilovaný do tzv. *intermediate language* (IL). Ide o nízkoúrovňový objektovo orientovaný jazyk určený pre virtuálny stroj, ktorý je v prostredí *runtime* konvertovaný do binárneho kódu [23]. Spolu so zdrojovým kódom je kompilovaný virtuálny stroj MonoVM a JIT. Rovnako ako u kompilácie pre iOS sú vo fáze linkingu odstránené všetky prebytočné a nepoužité triedy. Takto skompilovaná aplikácia beži súbežne s Java/Android runtime (ART) a interaguje s natívnymi typmi prostredníctvom Java Native Interface (JNI) [22].

#### 5.1.2.2 Flutter

Flutter je open-source SDK pre vývoj mobilných aplikácií, ktorý vyvinula spoločnosť Google. Hlavnou myšlienkou je vývoj vysokokvalitných natívnych




Obr. 5.1: Prehľad architektúry technológie Flutter [24]

aplikácií pre platformy iOS a Android za použitia jediného programovacieho jazyka a zdieľaného kódu naprieč platformami.

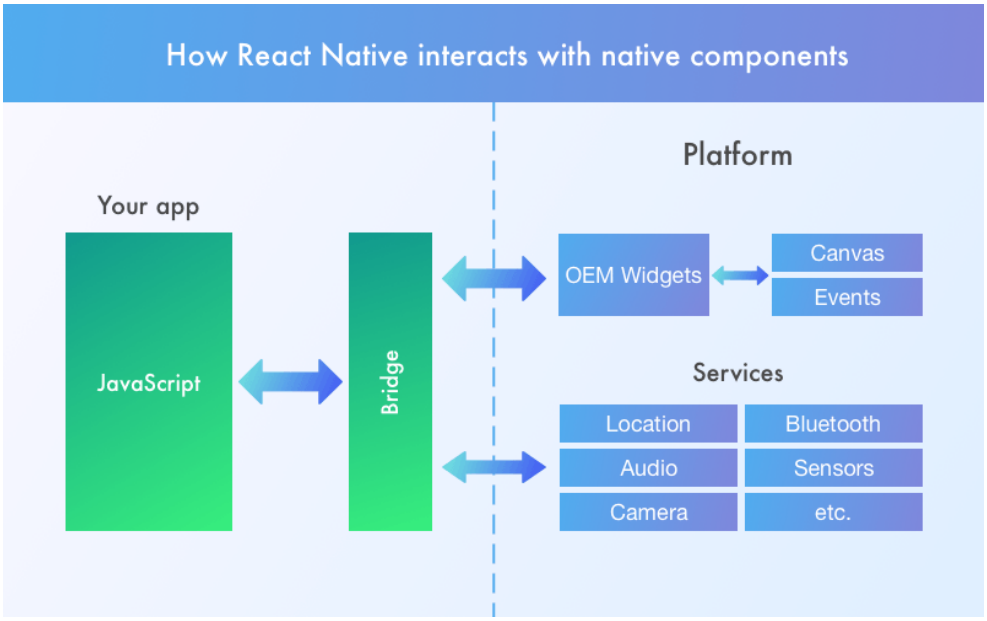
Na obrázku 5.1 je možné vidieť architektúru technológie Flutter. Zahŕňa v sebe moderný framework miestami pripomínajúci React a C/C++ engine pre vykresľovanie 2D grafiky [25]. Framework Flutter podporuje programovací jazyk Dart a jeho základnými stavebnými blokmi sú tzv. *widgets*. Pomocou hierarchickej kompozície týchto blokov je zostavené užívateľské rozhranie aplikácie [25]. Zdrojový kód aplikácie je skompilovaný do podoby IL, ktorá je následne vykreslená pomocou Skia canvas za pomoci C/C++ engine. Tento engine taktiež poskytuje prístup k natívnym rozhraniam platformy [26].

#### 5.1.2.3 Výhody

Medzi najväčšie výhody týchto aplikácií patrí fakt, že je možné vytvoriť aplikáciu pre najrozšírenejšie mobilné platformy za použitia jediného programovacieho jazyka s porovnateľným výkonom, a tým eliminovať nevýhody natívnych aplikácií. Ďalšou výhodou je, že Flutter je možné použiť na vytvorenie celej mobilnej aplikácie, ale rovnako je možné pridať ho do existujúcej natívnej aplikácie a implementovať pomocou nehou iba určitú časť aplikácie.

#### 5.1.2.4 Nevýhody

Najväčšou nevýhodou týchto aplikácií je, že kód nedokáže byť zdieľaný medzi mobilnou aplikáciou a webovou aplikáciou. Nevýhodou Xamarinu je, že napriek možnosti kompilovať pre obe platformy je potrebné napísať veľa dup-




Obr. 5.2: Prehľad architektúry technológie React Native [27]

licitného kódu pre rozdielne platformy, a tým sa zdanlivá výhoda pomerne výrazne stráca. Zároveň je v komunite Xamarinu známe, že oprava chýb v samotnom Xamarine, ako aj prispôsobenie Xamarinu na nové zmeny v natívnych platformách trvá pomerne dlho. Flutter je ešte pomerne nová technológia a napriek tomu, že si získava čim ďalej tým väčšiu popularitu, niektoré časti tohto frameworku sú ešte v pomerne ranom štádiu vývoja a ani komunita vývojárov zatiaľ nie je taká rozsiahla.

### 5.1.3 Interpretované aplikácie pre viacero platforiem

Interpretované aplikácie pre viacero platforiem spájajú technológie a princípy známe z vývoja aplikácií pre web so svetom mobilných aplikácií, ktorých vzhľad, vlastnosti a užívateľské interakcie sú takmer totožné s natívnymi mobilnými aplikáciami. Hlavným podporovaným jazykom v prípade týchto aplikácií býva JavaScript alebo nejaká z jeho nadstavieb, ako napríklad TypeScript. Základným kameňom týchto aplikácií je hierarchická štruktúra jednoduchých komponentov, ktoré neobsahujú žiadnu logiku, ale aj sofistikovanejších komponentov, ktoré dokážu vykonávať určitú logiku a udržiavať si svoj aktuálny stav. Všetky základné komponenty jazyka JavaScript tvoriace užívateľské rozhranie majú svoj ekvivalent na strane natívnej platformy a pomocou prekladača, ktorý funguje v prostredí *runtime*, často nazývaného *bridge*, čo v preklade znamená most, sú tieto komponenty interpretované a mapované na svoje natívne ekvivalenty. Pomocou tohto mosta sú sprístupnené

natívne rozhrania (pozri obrázok 5.2) na stranu JavaScriptovej aplikácie. Celá logika aplikácie sa vykonáva v časti implementovanej v jazyku JavaScript a pomocou mosta sa prenášajú potrebné parametre potrebné pre aktualizáciu natívnych komponentov. Medzi najznámejšie technológie fungujúce na tomto princípe patrí jednoznačne React Native vyvinutý spoločnosťou Facebook a o niečo menej známy NativeScript.

#### 5.1.3.1 Výhody

Medzi hlavné výhody tohto prístupu patria:

- Vďaka faktu, že celá logika sa vykonáva v aplikácii napísanej v jazyku JavaScript, je možné okrem zdieľania kódu pre iOS a Android zdieľať v rámci jednej aplikácie aj kód pre webovú aplikáciu.
- Výkon by teoreticky nemal byť výrazne nižší ako je to u natívnych aplikácií.
- Aplikácie pre mobilné platformy dokáže naprogramovať aj programátor z prostredia webových aplikácií.
- Možnosť využiť širokú škálu nástrojov pre vývoj a testovanie dostupných pre vývoj webových aplikácií.
- Pre React Native existuje obrovské množstvo učebných materiálov, kurzov, videí a diskusií na rôznych fórach.
- V prípade React Native existuje veľká komunita vývojárov, ktorí poskytujú knižnice tretích strán.
- Úpravy v kóde jazyka JavaScript môžu byť distribuované bez zložitého procesu vloženia novej verzie aplikácie do distribučných sietí jednotlivých platforiem.
- Rovnako ako Flutter, aj React Native môže byť použitý na vývoj celej aplikácie, môže byť použitý iba ako súčasť už existujúcej natívnej aplikácie a taktiež dokáže aplikácia v React Native importovať moduly z natívneho jazyka.

#### 5.1.3.2 Nevýhody

Medzi hlavné nevýhody tohto prístupu patrí samotný medzičlánok medzi aplikáciou v jazyku JavaScript a natívnou platformou. Každá zmena v aplikácii prechádza týmto miestom, a preto ide z hľadiska výkonu o úzky profil tohto riešenia. Druhou nevýhodou je, že každá novinka v natívnej platforme musí byť implementovaná v rámci týchto technológií a to typicky trvá niekoľko týždňov alebo mesiacov. Poslednou hlavnou nevýhodou tohto prístupu je, že

ide o pomerne nový prístup a zmeny medzi jednotlivými verziami môžu byť pomerne zásadné a často sa stáva, že aktualizáciou verzie je potrebné riešiť problémy, ktoré nastanú v dôsledku nekompatibility s predošlou verziou.

### 5.1.4 Hybridné aplikácie

Hybridné aplikácie sa v jednoduchosti dajú nazvať webovými aplikáciami, ktoré sú zabalené do prostredia s natívnym zabudovaným webovým prehliadačom prístupným prostredníctvom danej platformy. Tieto hybridné aplikácie sú rovnako ako aplikácie pre web vyvíjané s pomocou jazykov JavaScript, Hyper Text Markup Language (HTML) a Cascading Style Sheets (CSS). Tieto hybridné aplikácie naviac riešia niektoré nedostatky bežných webových aplikácií, napríklad to, že nevyžadujú pre svoj chod internetové pripojenie. Taktiež môžu byť rovnako ako predchádzajúce uvedené typy uverejnené v distribučných sieťach zodpovedajúcich jednotlivým platformám a užívateľ ich môže nájsť a nainštalovať úplne rovnako ako natívne aplikácie [28]. Medzi najznámejších reprezentantov tohto prístupu patria PhoneGap od spoločnosti Adobe, Cordova od spoločnosti Apache a Ionic Framework.

#### 5.1.4.1 Výhody

Hlavné výhody tohto prístupu sú:

- Kód je veľmi dobre zdieľaný nielen medzi jednotlivými mobilnými platformami, ale aj s webovými aplikáciami.
- Pre implementáciu tohto riešenia nie sú potrebné žiadne konkrétne znalosti vývoja mobilných aplikácií a mobilnú aplikáciu týmto spôsobom dokáže vyvinúť ktokoľvek, kto sa vyzná v základných webových technológiách.
- Pri implementácii mobilnej aplikácie je možné využiť ľubovoľné knižnice tretích strán, ktoré sú primárne určené pre vývoj webových aplikácií.
- K dispozícii je väčšina natívnych rozhraní platformy.

#### 5.1.4.2 Nevýhody

Toto riešenie je napriek svojim výhodám známe aj svojimi nedostatkami, ktoré sú nasledovné:

- Najväčšou nevýhodou je, že tieto aplikácie bývajú výrazne pomalšie a menej výkonné v porovnaní s akýmkoľvek z vyššie spomenutých typov.
- Nové funkcie natívnych aplikácií sa do týchto technológií dostávajú oneskorene.

- Tieto aplikácie bývajú ťažšie akceptovateľné do distribučných sietí mobilných platforiem
- Napriek prístupnosti rozhraní natívnych zariadení je táto prístupnosť obmedzená

### 5.1.5 Progresívne webové aplikácie

Progresívne webové aplikácie (PWA) sú webové stránky, ktoré využívajú moderné webové technológie na to, aby vytvorili pre mobilné weby zážitok podobný zážitku pri používaní natívnych mobilných aplikácií. Prostredníctvom odkazu na domovskej obrazovke môžu užívatelia využívať funkcie, ktoré boli u bežných webových stránok nemožné alebo pomerne ťažko dosiahnuteľné. Medzi tieto funkcie patria:

- Možnosť pristupovať k obsahu bez prístupu na internet
- Možnosť získavať upozornenia
- Ukladať obľúbené webové stránky alebo aplikácie ako ikony na domovskú obrazovku

PwA pomohli zvýšiť záujem užívateľov o webové stránky a aplikácie. Zatiaľ čo mobilné zariadenia sú jedným z hlavných motorov technológie, desktop predstavuje rastúci trh. PWA umožňujú bezproblémové používanie pre používateľov mobilných telefónov, tabletov a stolných počítačov, a to bez ohľadu na platformu [29].

#### 5.1.5.1 Výhody

Medzi hlavné výhody PWA patria:

- Náklady na tento typ aplikácie sú v porovnaní s ostatnými typmi výrazne nižšie.
- Obsah v PWA je vyhľadateľný prostredníctvom internetových vyhľadávačov.
- Tieto aplikácie sú neustále aktualizované.
- Majú pridelenú URL, takže je možné na ne odkazovať.

#### 5.1.5.2 Nevýhody

Medzi hlavnú nevýhodu PWA patrí, že nie sú podporované rovnako všetkými mobilnými prehliadačmi. Taktiež nemajú prístup k natívnym rozhraniam zariadenia, na ktorom sú prevádzkované. Tieto aplikácie nie je možné publikovať a distribuovať pomocou štandardných kanálov, ktoré fungujú pre distribúciu vyššie spomenutých typov mobilných aplikácií.

## 5.2 Výber technológie

Po zvážení všetkých okolností, požiadaviek na aplikáciu, výhod a nevýhod jednotlivých možných prístupov som sa v konečnom dôsledku rozhodol aplikáciu implementovať s použitím React Native. K tomuto rozhodnutiu ma viedli nasledujúce dôvody:

**Jeden kód pre viacero platforiem**
Súčasná webová aplikácia productboard na pozadí taktiež používa knižnicu React určenú pre webové aplikácie. Pri správnej abstrakcii logiky a rozumnej implementácii zobrazovacích komponentov je možné dosiahnuť pomerne vysoký podiel kódu, ktorý môže byť zdieľaný medzi webovou aplikáciou a mobilnými aplikáciami pre platformy iOS a Android.

**Možnosť rozšírenia aplikácie použitím natívneho kódu**
React Native stále pomerne nová technológia. V dobe začiatku implementácie aplikácie, ktorá je predmetom tejto práce, som s touto technológiou nemal žiadne praktické skúsenosti. Fakt, že do aplikácie postavenej na tejto technológii je možné pridávať časti aplikácie napísané v natívnom jazyku, mi dáva pocit bezpečia, a to v zmysle, že ak by som v rámci budúceho vývoja narazil na situáciu alebo problém, ktorý v rámci tejto technológie ešte nebol vyriešený, stále mám možnosť siahnuť po natívnom kóde a tento problém vyriešiť mimo React Native.

**Základom je jazyk a postupy známe z vývoja webových aplikácií**
Ďalším pomerne dôležitým faktom je, že som sa osobne nikdy nestretol s postupmi vývoja mobilných aplikácií pre platformu iOS a taktiež neovládam ani jeden z programovacích jazykov Objective-C a Swift. Naopak mám pomerne solídnu znalosť jazyka JavaScript a niekoľkoročné skúsenosti s vývojom webových aplikácií, a to aj s knižnicou React, na ktorej princípoch práve React Native stavia. A nejde iba o mňa. Po tom, čo dokončím túto diplomovú prácu, sa aplikácia bude naďalej vyvíjať a vylepšovať a je pravdepodobné, že k tomuto vývoju sa pridajú aj iní programátori spoločnosti productboard. Keďže táto spoločnosť v súčasnej dobe nedisponuje vývojármi natívnych mobilných aplikácií, ale naopak veľká časť vývojového tímu rovnako ako ja ovláda JavaScript a React, vďaka týmto znalostiam bude pre nich možné pristúpiť k vývoju a vylepšovaniu tejto aplikácie pomerne jednoducho a bez nutnosti naučiť sa nový programovací jazyk alebo nové metódy a postupy vývoja.

**Rozšírenosť a veľká komunita**
V neposlednom rade veľmi dôležitým faktorom pri výbere tejto technológie zohral aj fakt, že práve React Native je jedna z narozšírenejších technológií pre vývoj aplikácií pre viacero platforiem. Existuje obrovské

množstvo materiálov, tutoriálov a videí, z ktorých je možné sa o React Native učiť a taktiež existuje pomerne veľká zásoba knižníc tretích strán, vďaka ktorým nie je nutné implementovať dobre známe koncepty znova od začiatku.

## 5.3 React Native

Ako som už spomenul v predošlých častiach tejto kapitoly, React Native je framework v jazyku JavaScript, ktorý umožňuje vytvárať mobilné aplikácie pre platformy iOS a Android. Jeho základ tvorí knižnica pre tvorbu užívateľských rozhraní React, ktorá sa využíva aj pri tvorbe webových aplikácií. Pre správne pochopenie toho, ako React Native funguje, je najprv potrebné vysvetliť, ako funguje samotný React [30].

V knižnici React existuje takzvaný virtuálny Document Object Model (DOM), v preklade objektový model dokumentu, a tento virtuálny DOM slúži ako vrstva medzi programátorským predpisom toho, ako by dokument mal vyzerať, a reálnym prekreslením DOM v prehliadači. Na to, aby sa udiala zmena interaktívneho užívateľského rozhrania v prehliadači, je potrebné upraviť priamo DOM v tomto prehliadači. Táto akcia je ale pomerne náročná a drahá a časté úpravy DOM majú za následok výrazný dopad na výkon aplikácie. Preto namiesto priamych úprav DOM v prehliadači si React udržiava v pamäti jeho virtuálnu podobu so zmenami, ktoré majú nastať, a následne porovná túto virtuálnu podobu DOM s reálnou podobou DOM a v prípade, že nastali zmeny, prekreslí iba minimálnu potrebnú časť, čiže iba tie časti DOM, v ktorých nastala zmena [30].

V kontexte knižnice React pre vývoj webových aplikácií si veľa vývojárov myslí, že virtuálny DOM primárne slúži na optimalizáciu výkonu webovej aplikácie. Tento virtuálny DOM má určite pozitívny dopad na výkon aplikácie, ale jeho naozajstný potenciál tkvie v sile tejto abstrakcie. Umiestnenie abstraktnej vrstvy medzi kód vývojára a výsledku, ktorý zariadenie vykreslí, ponúka veľké množstvo možností, ako sa dá tento princíp využiť [30].

Tento princíp je samozrejme využitý v React Native, ktorý nepracuje s objektovým modelom dokumentu, ale s natívnymi rozhraniami danej platformy. Po zmene vo virtuálnom DOM sa vyvolá aktualizácia parametrov, prípadne vykreslenie iných elementov užívateľského rozhrania, ktoré ponúka priamo natívna platforma. Predanie informácií o zmenách vo virtuálnom DOM prebieha pomocou medzivrstvy, ktorú autori React Native nazývajú *bridge* a ide o pomyselný most medzi aplikáciou v React Native a rozhraniami, ktoré ponúka natívna platforma. Tento most interpretuje zmenu z jazyka JavaScript, identifikuje komponent, ktorý sa zmenil, nájde ekvivalent tohto komponentu v natívnej platforme a vyvolá zmenu natívneho komponentu.

Základný kameň knižnice React je element užívateľského rozhrania nazývaný komponent. Skladaním viacerých komponentov do hierarchickej štruktúry vzniká

užívateľské rozhranie. V knižnici React existujú dva rozdielne typy komponentov. Prvý typ je jednoduchý komponent reprezentovaný čistou funkciou, ktorý prijíma nejaké vstupné dáta nazývané *props* a na základe týchto dát vykreslí komponent. Tento typ komponentu nemá žiadny vnútorný stav, a dátami, ktoré prišli na vstupe, nijako nemanipuluje. Druhý typ komponentu je komponent, ktorý má vlastný vnútorný stav. Rovnako ako predchádzajúci typ komponentu, aj tento typ prijíma vstupné dáta, ale rozdiel je v tom, že vykreslenie komponentu môže a nemusí nastať ako reakcia na prichádzajúce dáta a taktiež toto vykreslenie môže byť vyvolané zmenou vnútorného stavu. Každý komponent je reprezentovaný triedou a táto trieda obsahuje povinnú metódu s predpisom pre vykreslenie komponentu. Pre popis toho, ako sa má komponent vykresliť sa v knižnici React využíva tzv. JSX syntax. Ide o špeciálne rozšírenie syntaxe jazyka JavaScript, ktorá vzhľadom pripomína značkovacie jazyky, ako je napríklad Extensible Markup Language pre veľa ľudí známy pod skratkou XML a funkčne môže pripomínať nejaký šablónovací jazyk [31]. Táto syntax je odporúčaná pre popis užívateľského rozhrania komponent knižnice React, ale nejedná sa o povinnosť a existuje ekvivalentný zápis pomocou funkcie.

Každý komponent v knižnici React má definovaný svoj životný cyklus. Pri úvodnom vykreslení obrazovky sa po vykreslení rodičovského komponentu vloží komponent potomka do štruktúry DOM a následne sa taktiež vykreslí.Takto prebehne vykreslenie všetkých komponentov hierarchickej štruktúry. Po úvodnom vykreslení až do odobrania komponentu z DOM sa komponent prekreslí iba v prípadoch, kedy je to potrebné. To sú situácie, kedy komponent dostane na vstupe nové dáta alebo sa zmení jeho vnútorný stav. Ak taká udalosť nastane, React vypočíta rozdiel medzi virtuálnym a reálnym DOM a iniciuje prekreslenie tých komponentov, ktoré sú odlišné. Tento životný cyklus je rovnaký aj pre React Native, pričom rozdiel je v akcii prekreslenia. V prípade React Native sa neprekreslí uzol v DOM, ale iniciuje sa prekreslenie elementu užívateľského rozhrania natívnej platformy.

React je knižnica pre tvorbu užívateľských rozhraní založená na hierarchickej kompozícii bezstavových a stavových komponentov. React Native rozširuje React určený pre úpravy DOM v prehliadači o možnosť manipulovať priamo s rozhraniami a elementami užívateľského rozhrania natívnej platformy. Ani jedna z týchto technológií ale nerieši otázku udržiavania a manažovania lokálneho stavu aplikácie a taktiež nerieši tok dát medzi aplikáciou a serverom. Keďže ale mnou implementovaná aplikácia potrebuje riešiť tieto problémy, využívam k tomu v súčasnosti najznámejšie a najpokrokovejšie technológie, ktorými sú knižnica Redux a jazyk pre získavanie dát z tzv. Application programming interface (API) nazývaný GraphQL. Keďže pomerne zásadným spôsobom zasahujú do architektúry a funkčnosti aplikácie, popíšem tieto technológie podrobnejšie v nasledujúcich sekciách.

## 5.4 Redux

Požiadavky na aplikácie v jazyku JavaScript sa všeobecne stávajú čím ďalej tým viac komplikované a programátor musí spravovať viac stavov ako kedykoľvek predtým. Tieto stavy môžu obsahovať odpovede a dáta získané zo servera, ale taktiež lokálne vytvorené dáta. Stav užívateľského rozhrania taktiež naberá na komplexnosti, keďže potrebujeme spravovať veľké množstvo udalostí, ktoré užívateľ v užívateľskom rozhraní vyvolá. Udržiavanie tohto stavu v rámci jednotlivých komponentov je veľmi náročné, a to hlavne v prípade, že komponenty svoj stav vzájomne ovplyvňujú. V tomto stave je veľmi jednoduché stratiť prehľad o tom, kedy, ako a prečo sa stav komponentu zmenil. Nehovoriac o prípade, kedy zmena dát nastáva asynchrónne [32]. A práve takéto prípady pomáha riešiť knižnica Redux. Redux ponúka jednotný spôsob udržiavania stavu celej aplikácie na jednom mieste, pričom komponenty, ktoré pracujú s určitou časťou aplikačného stavu, majú prístup iba k definovanej časti, s ktorou potrebujú pracovať, a nemôžu svojvoľne ovplyvňovať časti aplikačného stavu, ku ktorým nemajú prístup. Tieto zmeny stavu je zároveň možné vykonávať iba presne definovaným spôsobom, čo napomáha jednoduchému a rýchlemu zisteniu, kedy, za akých okolností a prečo sa stav aplikácie zmenil.

Hlavná myšlienka knižnice Redux je postavená na troch konceptoch. Prvý koncept je jednotné úložisko aplikačného stavu, z ktorého môžu komponenty stav iba čítať a nie je možné do neho zapisovať. Autori knižnice nazývajú toto úložisko *store*. V kontexte jazyka JavaScript ide o objekt, ktorého vlastnosti môžu obsahovať dáta jednoduchých dátových typov alebo ďalšie objekty. Druhým konceptom sú tzv. *actions*. Ide o funkcie s presne definovanou štruktúrou argumentov, ktoré tieto funkcie prijímajú. Prvým povinným argumentom je identifikátor typu akcie a druhým argumentom je objekt, obsahujúci nové dáta, ktoré sa majú v aplikačnom stave objaviť. Vyvolanie konkrétnej akcie s objektom obsahujúcim zmeny je jediný možný spôsob, ako vyvolať zmenu aplikačného stavu. To napomáha prehľadnosti, predvídateľnosti a jednoduchej dohľadateľnosti dôvodov, prečo sa aplikačný stav zmenil. Tretím a posledným konceptom sú tzv. *reducers*. Ide o čisté funkcie, ktoré aplikovaním zmien, ktoré sú popísané vo vyvolanej akcii, vytvoria z aktuálneho aplikačného stavu nový stav aplikácie, ktorý je výstupným parametrom týchto funkcií. Tieto stavy aplikácie sa ukladajú a práve vďaka tomu je možné jednoducho a presne vyhľadať, aké zmeny v aplikačnom stave spôsobila konkrétna akcia [33].

## 5.5 GraphQL

GraphQL je vyhľadávací jazyk pre získavanie dát pre klientske aplikácie z API a *runtime* prostredie na strane servera, ktoré zabezpečuje vrátenie adekvátnych

dát pre definované vyhľadávacie požiadavky s použitím existujúcich dát a existujúcej vyhľadávacej logiky. GraphQL poskytuje úplný a zrozumiteľný popis dát poskytovaných prostredníctvom API, poskytuje klientskym aplikáciám možnosť presne definovať požadované dáta a požadovať iba tie dáta, ktoré sú pre aplikáciu naozaj potrebné, uľahčuje vývoj API v priebehu času a poskytuje výkonné vývojárske nástroje [34].

Pre použitie GraphQL som sa rozhodol preto, že prináša množstvo výhod oproti tradičnému *Representational State Transfer* (REST) API. Hlavnou výhodou oproti REST API je možnosť presne definovať dáta, ktoré aplikácia potrebuje. REST API poskytuje prístupový bod, ktorý vráti objekt reprezentujúci určitý dátový model so všetkými vlastnosťami, ktoré sú pre tento model definované a sú uložené v databáze. Vo väčšine prípadov ale klientska aplikácia nepotrebuje hodnoty pre všetky dostupné vlastnosti daného modelu. Pomocou GraphQL aplikácia dostane iba hodnoty pre tie vlastnosti dátového modelu, ktoré boli špecifikované v požiadavke pre získanie dát. Tým sa výrazne zníži množstvo dát, ktoré je potrebné preniesť medzi klientskou aplikáciou a serverom, čo výrazne napomáha rýchlejšej a pohotovejšej komunikácii. Rovnako to dáva možnosť efektívne pracovať s ukladacím priestorom zariadenia, na ktorom klientska aplikácia beží, v prípade, že aplikácia tieto dáta ukladá do pamäte.

Ďalšou dôležitou výhodou GraphQL je, že na rozdiel od REST API je možné získať potrebné dáta z viacerých dátových modelov pomocou jedinej požiadavky. Spomenuté výhody platia samozrejme nielen v prípade potreby dáta získať, ale taktiež aj pri vytváraní nových dát alebo úprave existujúcich dát.

Vďaka tomu, že GraphQL vyžaduje, aby boli všetky dáta na serveri popísané, je pri paralelnom vývoji klientskej a serverovej aplikácie veľmi jednoduché zistiť, aké dáta server dokáže poskytnúť. GraphQL prístupový bod poskytuje GraphQL schému, čo je presný popis všetkých dátových modelov, ich vlastností a dátových typov, ktoré sú pre jednotlivé vlastnosti povolené. Nie je teda potrebné vyhľadávať žiadnu dokumentáciu alebo kontaktovať programátora serverovej aplikácie, aby objasnil štruktúru dát, ako to často býva v prípade REST API.

Keďže samotný jazyk GraphQL je abstraktný a nie je viazaný na žiadny konkrtétny programovací jazyk, použil som v rámci implementácie mojej aplikácie knižnicu Apollo GraphQL Client, ktorá poskytuje všetky výhody GraphQL pre aplikácie naprogramované v jazyku JavaScript.

## 5.6 Nástroje použité pri vývoji

Pre pohodlný proces vývoja som využil niekoľko nástrojov, ktoré mi uľahčovali prácu a zároveň pri programovaní kontrolovali kvalitu kódu a upozorňovali ma

na prípadné nedostatky, vďaka čomu som sa vyhol potenciálnym nepríjemným chybám.

### 5.6.1 TypeScript

V prvom rade som siahol po statickom analyzátore kódu pre jazyk JavaScript s názvom TypeScript. JavaScript patrí medzi dynamicky typované jazyky, čo v praxi znamená, že kontrola správnosti dátových typov prebieha až pri samotnom vykonávaní kódu. Tento prístup pomerne výrazne napomáha tomu, aby sa akákoľvek nepozornosť programátora prejavila až chybami priamo v produkčnej aplikácii. Častokrát ide o chyby, ktoré sa dajú jednoducho odhaliť statickou analýzou zdrojového kódu, a tým predísť zbytočne zdĺhavému ošetrovaniu vstupov alebo chybám v produkčnom kóde. A práve TypeScript pomáha chyby súvisiace s nekompatibilnými dátovými typmi odhaľovať pomocou už spomínanej statickej analýzy kódu.

### 5.6.2 TSLint

Spolu s TypeScript som si nakonfiguroval aj ďalší nástroj na statickú analýzu kvality kódu na základe vopred definovaných pravidiel. Keďže JavaScript patrí medzi jazyky, ktoré sú interpretované, prípadné syntaktické chyby v kóde sú taktiež viditeľné až pri vykonávaní kódu. TSLint pomocou statickej analýzy kontroluje správnosť syntaxe kódu a upozorňuje na prípadné chyby okamžite priamo v editore. Tento nástroj je konfigurovateľný a pri správnej konfigurácii prináša okrem zobrazovania syntaktických chýb aj pravidlá, ktoré programátora nútia písať kód prehľadne, aby bol jednoducho čitateľný a udržiavateľný.

### 5.6.3 Prettier

Prettier je nástroj na automatické formátovanie kódu. Programátor absolútne nemusí rozmýšľať nad správnym formátovaním kódu, ktorý aktuálne píše, pretože tento nástroj automaticky naformátuje kód v momente, keď je súbor obsahujúci zdrojový kód uložený. Vďaka tomuto jednoduchému nástroju som ušetril množstvo času, ktorý som mohol venovať produktívnej práci.

### 5.6.4 GraphQL Code Generator

Ako už bolo naznačené v sekcii venujúcej sa GraphQL, implementovaná aplikácia komunikuje so serverovou stranou, ktorá poskytuje GraphQL prístupový bod, pomocou ktorého si klientska aplikácia môže vyžiadať dáta. Keďže GraphQL striktne vyžaduje dátové typy, je pre každý dátový model na strane servera pripravená GraphQL schéma obsahujúca jednotlivé vlastnosti dátového modelu a taktiež dátové typy, ktoré jednotlivým vlastnostiam zodpovedajú. Pre správne fungovanie musia byť na klientskej strane vyžadované dáta s

rovnakou štruktúrou a dátovými typmi ako sú definované na strane servera. Vďaka nástroju GraphQL Code Generator je možné jednoducho vygenerovať dátové typy jednotlivých dátových modelov na strane klienta z GraphQL schémy, čím je možné ušetriť čas potrebný pre manuálne definovanie typov na strane klienta a ich prípadnú aktualizáciu, ak na strane servera nastane zmena.

## 5.7 Obmedzenia zo strany servera

V úvode implementácie mobilnej aplikácie som potreboval zistiť, aké technológie používa súčasná aplikácia, ktorá beží na strane servera a poskytuje pre webovú aplikáciu rozhranie pre výmenu dát. Bohužiaľ som zistil, že súčasná aplikácia na strane servera používa pre komunikáciu REST API. A to nebol jediný a najmenší problém. Počas implementácie prvej verzie mobilnej aplikácie som narazil na niekoľko pomerne závažných problémov s aplikáciou na strane servera. Tá bola naprogramovaná prístupom, ktorý je adekvátny v situácii, ak existuje jediná klientska aplikácia, ktorá funguje vo webovom prehliadači. Bohužiaľ ale nebola pripravená na iné klientske aplikácie, ktoré môžu mať aj inú podobu ako je webová *Single page application* (SPA).

Prvým a hneď pomerne zásadným problémom bol spôsob, akým bola zabezpečená autentifikácia a autorizácia. Prevedenie autentifikácie bolo realizované pomocou *cookies*, čo je koncept, ktorý výborne funguje pre webovú klientsku aplikáciu, ale mobilná aplikácia nepozná pojem *cookies* a autentifikácia v týchto aplikáciách je založená na výmene tzv. *tokens*. Pre možnosť vykonávať akékoľvek zmeny prostredníctvom REST API serverová aplikácia ukladala z dôvodu bezpečnej autorizácie *CSRF token* do meta hlavičiek dokumentu *Hypertext Markup Language* (HTML). Na rozdiel od webovej aplikácie mobilná aplikácia štandardne nedisponuje nástrojmi na analýzu a získavanie dát z HTML dokumentu.

Ďalším problémom, na ktorý som narazil, je fakt, že pri počiatočnom načítaní webovej aplikácie sa načítava obrovské množstvo počiatočných dát, z ktorých veľká časť vôbec nie je potrebná pre mobilnú aplikáciu a ďalšia veľká časť týchto dát nie je potrebná pre počiatočné spustenie aplikácie. Pri pokuse vyžiadať si tieto dáta pomocou REST API som zistil, že serverová aplikácia pre jednotlivé modely vráti všetky záznamy, ktoré sa daného dátového modelu týkajú, čo v niektorých prípadoch znamenalo aj desaťtisíce záznamov. V niektorých situáciách mobilná aplikácia zároveň odosielala požiadavky pre dáta pre niekoľko na sebe závislých dátových modelov, zároveň spracovávala veľké množstvo prichádzajúcich dát a ešte k tomu vykresľovala užívateľské rozhranie. To často viedlo k výraznému poklesu výkonu aplikácie hlavne na starších modeloch chytrých telefónov. Na strane servera nie je implementovaný žiaden stránkovací algoritmus, a preto aj tento problém bol pomerne závažný.

Posledným závažným problémom bola možnosť poskytovať dáta pre noti-

fikácie. Súčasná webová aplikácia síce disponuje určitým mechanizmom synchronizácie dát medzi klientskymi aplikáciami v reálnom čase, ale tento mechanizmus je bohužiaľ nekompatibilný s potrebami notifikačnej služby platformy iOS.

Vyššie uvedené problémy viedli k situácii, že spolu s ďalšími členmi vývojového tímu sme sa rozhodli naprogramovať aplikáciu fungujúcu na strane servera, ktorá vytvorí GraphQL nadstavbu nad REST API súčasnej produkčnej serverovej aplikácie, implementuje stránkovacie algoritmy a algoritmy pre možnosť vyhľadávania v dátach na strane servera a vyrieši problém autentifikácie a autorizácie použitím riešenia založeného na výmene *tokens* medzi klientskou stranou a stranou servera. Táto serverová aplikácia ale nie je súčasťou zadania tejto diplomovej práce, preto nebudem ďalej rozoberať jej detaily.

## 5.8 Implementácia

Po vyriešení problémov spomenutých v predchádzajúcej sekcii som prvú verziu mobilnej aplikácie, ktorá bola založená na komunikácii so súčasnou produkčnou serverovou aplikáciou rozhodol nepoužiť a začal som odznova implementáciou druhej verzie mobilnej aplikácie, ktorá je na strane servera podporená novo vyvinutou aplikáciou. V tejto sekcii podrobnejšie rozoberiem túto druhú verziu implementácie mobilnej aplikácie.

### 5.8.1 Štruktúra kódu

Zdrojový kód aplikácie pozostáva z väčšieho počtu súborov, ktoré som rozdelil do niekoľkých kategórií podľa funkcionality, typu a príslušnosti k určitej časti aplikácie. Tejto kategorizácii zároveň zodpovedá aj adresárová štruktúra v zložke so zdrojovým kódom.

**actions**
: Tento adresár obsahuje súbory, ktoré definujú Redux akcie pre jednotlivé časti aplikácie.

**components**
: V tomto adresári sú umiestnené komponenty, ktoré sú zobrazované na jednotlivých obrazovkách. Tie, ktoré sú špecifické pre konkrétnu obrazovku, sú umiestnené v podadresári nesúcom názov danej obrazovky, a komponenty, ktoré sú dostatočne generické, nie sú viazané na konkrétnu obrazovku sú umiestnené v adresári *ui*.

**config**
: V tomto adresári sa nachádzajú súbory, ktoré obsahujú nejaké konfiguračné informácie, ako napríklad globálne premenné pre farby, veľkosti písma a iné premenné týkajúce sa vzhľadu, alebo konfiguráciu pre nastavenie URL prístupových bodov na server.

**constants**
Tento adresár obsahuje súbory s konštantami a enumerátormi súvisiacimi s určitou logikou.

**reducers**
Súbory, ktoré sa nachádzajú v tomto adresári, popisujú Redux reducers.

**screens**
V tomto adresári sa nachádza konfigurácia navigácie a jednotlivé komponenty, ktoré reprezentujú celé obrazovky.

**services**
Ide o súbory s logikou poskytujúcou služby prístupné z celej aplikácie, ako napríklad odhlásenie užívateľa.

**stores**
Tento adresár obsahuje súbory popisujúce jednotlivé časti, z ktorých je zložený Redux store.

**utils**
V tomto adresári sa nachádzajú súbory obsahujúce pomocné funkcie, ktoré sú využívané napríklad počas výpočtov alebo spracovávania dát.

### 5.8.2 Logické a prezentačné komponenty

Ako som už spomínal vyššie, hlavná stavebná jednotka celej aplikácie je komponent. Existuje niekoľko rôznych prístupov k tomu, ako je možné jednotlivé komponenty naprogramovať, no ja som zvolil spôsob rozdeliť komponenty na také, ktoré plnia buď logickú alebo prezentačnú funkciu. Logické komponenty majú za úlohu získavať dáta a spracovávať dáta potrebné pre prezentačné komponenty a vykonávať logiku aplikácie. Tieto komponenty typicky ovplyvňujú aplikačný stav a obsahujú iné komponenty ako potomkov. Najtypickejším príkladom takéhoto logického komponentu v rámci mojej aplikácie sú komponenty reprezentujúce jednotlivé obrazovky. Konkrétne to môže vyzerať takto:

```
import * as React from 'react';

....

type Props = {} & NavigationProps;

class ReleasesScreen extends React.PureComponent<Props> {

  ....

  onFeaturePress = (id: Feature['id']) => {
    const { navigation } = this.props;

    this.onSearchClose();
    navigation.navigate('FeatureDetail', { featureId: id });
  };

  onSearchClose = () => {
    const { navigation } = this.props;

    navigation.setParams({ searchModalVisible: false });
  };

  renderFeature = (item: any) => (
    <FeatureItem
      feature={item.node}
      onPressHandler={() => this.onFeaturePress(item.node.id)}
    />
  );

  render() {
    const {
      navigation: {
        state: { params }
      }
    } = this.props;

    return (
      <React.Fragment>
        <DataQuery
          dataQuery={GET_NOTES}
          SuccessComponent={ReleasesList}
        />
        <SearchModalDataContainer
          visible={params && params.searchModalVisible}
          searchableModelType={SEARCHABLE_MODEL_TYPES.FEATURE}
          renderItem={this.renderFeature}
          onCancel={this.onSearchClose}
        />
      </React.Fragment>
    );
  }
}

export default ReleasesScreen;
```

Listing 5.1: Ukážka kódu logického komponentu ReleasesScreen

V ukážke kódu 5.1 sa nachádza komponent reprezentujúci obrazovku, ktorá zobrazuje zoznamy funkcionalít zoskupené podľa jednotlivých vydaní. Na riadkoch 11 až 21 sú definované funkcie s logikou, ktorá sa má vykonať pri určitých udalostiach, ktoré nastanú v prezentačnom komponente `<SearchModalDataContainer />`, ktorý je umiestnený v predpise metódy zabezpečujúcej vykreslenie komponentov na riadkoch 44 až 49. Na riadkoch 24 aŽ 29 je definovaná funkcia, ktorá pre spomínaný komponent definuje, ako sa majú vykresliť jednotlivé záznamy. Komponent `<ReleasesScreen />` sám o sebe neprezentuje žiadne údaje priamo na obrazovku zariadenia.

```
1  import * as React from 'react';
2
3  ....
4
5  type Theme = {
6    container?: string;
7    text?: string;
8  };
9
10 type Props = {
11    ....
12 };
13
14 export type NativeBaseButton = Omit<Props, ['text', 'children']>;
15
16 export default class Button extends React.PureComponent<Props> {
17   render() {
18     const { text, theme, children, ...NBProps } = this.props;
19
20     return (
21       <NativeButton
22         style={[
23           styles.button,
24           NBProps.primary && styles.primary,
25           theme && theme.container
26         ]}
27         {...NBProps}
28       >
29         {text && (
30           <ButtonText
31             theme={theme && theme.text}
32             text={text} {...NBProps}
33           />
34         )}
35         {children}
36       </NativeButton>
37     );
38   }
39 }
```

Listing 5.2: Ukážka kódu prezentačného komponentu Button

Na rozdiel od logických komponentov, prezentačné komponenty prijímajú od logických komponentov argumenty a na základe dát z týchto argumentov vykreslia elementy natívneho rozhrania platformy s určitými parametrami. Tieto komponenty typicky nemajú žiadnu inú úlohu a neovplyvňujú aplikačný stav. Najlepším príkladom jednoduchého prezentačného komponentu je napríklad tlačidlo, ktorého kód je v ukážke 5.2. Tento komponent obsahuje iba jedinú metódu, a to je práve povinná metóda, ktorá slúži pre vykreslenie elementov. Aj prezentačné komponenty môžu obsahovať ďalších potomkov, ale typicky ide o ďalšie prezentačné komponenty. V ukážke kódu ide napríklad o komponent `<ButtonText />` na riadkoch 30 až 33.

Kapitola 6

# Testovanie

V dnešnom svete sú požiadavky na kvalitu a spoľahlivosť softvéru náročnejšie ako kedykoľvek predtým. Konkurencia spoločností, ktoré vyvíjajú softvér, je veľká, a to vytvára tlak na vývojárov, aby dodávali kvalitný, výkonný a spoľahlivý softvér, inak hrozí, že sa zákazník rozhodne pre iného dodávateľa. Práve preto je testovanie softvéru neoddeliteľnou a často aj nevyhnutnou súčasťou vývoja. Slúži primárne na vyhľadávanie chýb v implementácii, kontrolu toho, či softvér dosahuje dostatočnú kvalitu v parametroch ako je bezpečnosť, spoľahlivosť, výkonnosť alebo použiteľnosť. Testovanie by nemalo byť posledným štádiom vývoja softvéru pred tým, než je program alebo služba vydaná do produkčného prostredia, ale naopak by testovanie malo prebiehať kontinuálne počas celého životného cyklu softvérového projektu. Medzi spoločnosťami zaoberajúcimi sa vývojom a predajom softvéru je dobre známe, že cena za opravu chýb pri vývoji softvéru výrazne rastie so štádiom vývoja.

Existuje viacero kritérií, na základe ktorých sa dá testovanie rozdeliť na menšie kategórie. Z hľadiska veľkosti testovaných celkov môžme rozdeliť testovanie na textovanie funkčných jednotiek, testovanie na úrovni modulov, integračné testy, funkčné testy, systémové testy a akceptačné testy. Z hľadiska spôsobu realizácie to môžu byť manuálne, automatizované alebo exploratívne testy.

Súčasťou implementácie mobilnej aplikácie v rámci tejto diplomovej práce bola implementácia sady automatizovaných testov, ktoré testujú funkčnosť jednotlivých funkčných jednotiek implementovaného kódu a taktiež celkový bezproblémový priechod hlavnými scenármi, ktoré aplikácia ponúka.

## 6.1 Automatizované testovanie

V rámci vývoja aplikácie som pravidelne testoval vyvíjanú aplikáciu automatizovaným testovaním funkčných jednotiek a end-to-end testovaním, aby som sa ubezpečil, že som úpravami kódu nezaviedol do predošlej funkcionality nejakú chybu. Taktiež som pravidelne pridával nové testy pre nový kód pridaný

pri vývoji novej funkcionality. Tieto testy som počas vývoja spúšťal manuálne na lokálnom počítači, no v budúcnosti by mohli testy prebehnúť automaticky pri zahrnutí kódu do hlavnej vetvy verzovacieho systému Git.

### 6.1.1 Testovanie funkčných jednotiek

Testovanie funkčnej jednotky, alebo ako to mnohí poznajú pod anglickým názvom *unit test*, je kus kódu napísaný vývojárom, ktorý podrobuje testovaniu veľmi malú špecifickú oblasť funkčnosti v kóde. Zvyčajne sa testy funkčných jednotiek využívajú pre testovanie konkrétnych funkcií alebo metód, ktoré vykonávajú nejakú malú časť celkového algoritmu [35].

Jednoducho sa totiž môže stať, že programátor urobí chybu v pomerne malej a jednoduchej funkcii, či už z nepozornosti alebo nedostatočným zamyslením sa nad všetkými možnými vstupmi a prípadmi, ktoré môžu v rámci určitej logiky nastať. Malá chyba v jednoduchej funkcii sa dá jednoducho prehliadnuť, ale ak je na tejto funkcii postavený zložitý algoritmus, ktorý ďalej pracuje s výsledkom tejto malej časti logiky, môže to mať na výsledný algoritmus rozsiahle následky. Preto je vhodné použiť testovanie funkčných jednotiek na izolované otestovanie správnej funkčnosti malých častí kódu, z ktorých sa softvér skladá.

V mojom prípade som na testovanie funkčných jednotiek použil nástroj Jest. Ide o jednoduchý *framework* určený pre testovanie aplikácií napísaných v jazyku JavaScript. Tento nástroj som si vybral hlavne kvôli jeho jednoduchej konfigurácii, jednoduchému a priamočiaremu spôsobu písania testov a možnosti paralelného behu jednotlivých testov.

### 6.1.2 End-to-End testy

End-to-End test alebo skrátene E2E test je metodika používaná na overenie, či tok aplikácie funguje tak, ako je navrhnutý od začiatku až do konca. Účelom vykonania skúšok typu end-to-end je identifikácia závislostí systémov a zabezpečenie správnej informácie medzi rôznymi systémovými komponentmi a systémami.

Testovanie typu end-to-end zahŕňa zabezpečenie toho, aby integrované komponenty aplikácie fungovali podľa očakávania. Celá aplikácia je testovaná v reálnom scenári, ako napríklad komunikácia s databázou, sieťou, hardvérom a inými aplikáciami.

Pomocou nástroja pre tvorbu end-to-end testov Detox som vytvoril testy pre všetky funkcionality definované ako funkčné požiadavky v kapitole zaoberajúcej sa analýzou a popísané v diagramoch užívateľskej cesty, okrem funkcie notifikácií.

## 6.2 Testovanie použiteľnosti

### 6.2.1 Heuristická analýza

Heuristická analýza je metóda, ktorej cieľom je nájsť problémy v použiteľnosti užívateľského rozhrania, a preto môže byť zaradená do procesu návrhu užívateľského rozhrania. Vyžaduje si skupinu expertov, ktorí predmetné užívateľské rozhranie preskúmajú a posúdia dodržiavanie známych princípov použiteľnosti [36].

Pri testovaní užívateľského rozhrania, ktoré som navrhol, som použil desaťbodovú heuristickú analýzu Jakoba Nielsena. K tomuto testovaniu som využil expertných testerov, ktorí sa zaoberajú návrhom a vývojom aplikácií a zároveň sa v rámci svojho profesného života stretávajú s webovou aplikáciou productboard ako jej zákazníci. Na každej z obrazoviek aplikácie, ktorú som navrhol a implementoval, títo experti testovali súlad nasledujúcimi pravidlami použiteľnosti:

1. viditeľnosť stavu aplikácie,
2. zhoda v terminológii medzi systémom a reálnym svetom,
3. možnosť užívateľa pohodlne dostať sa z nechcenej situácie,
4. konzistencia a naplnenie štandardov,
5. prevencia chýb,
6. viditeľnosť a prístupnosť prvkov bez nutnosti si ich pamätať,
7. estetický a minimalistický dizajn,
8. zrozumiteľnosť chybových správ a ich schopnosť naviesť užívateľa k správnemu riešeniu,
9. prítomnosť stránky s pomocou alebo dokumentáciou [37].

Výsledkom tohto testovania je, že žiadny z expertov neodhalil pomocou heuristickej analýzy žiadne výrazné chyby v návrhu užívateľského rozhrania.

### 6.2.2 Kognitívny priechod

Kognitívny priechod je metóda testovania použiteľnosti, pri ktorej skupina expertov prechádza sériou úloh a z pohľadu užívateľa si kladú otázky:

1. Má užívateľ všetko potrebné k vykonaniu požadovanej akcie?
2. Je odozva na užívateľskú akciu zrozumiteľná a je čas odozvy primeraný okolnostiam?

V tomto prípade som expertných testerov požiadal o otestovanie všetkých prípadov možného použitia aplikácie, ktoré som uviedol v kapitole o návrhu. Ani v tomto prípade experti nenarazili na žiadny závažný problém s použiteľnosťou.

### 6.2.3 Užívateľské testovanie použiteľnosti

Význam a popis toho, čo vlastne testovanie použiteľnosti je, som už uviedol v druhej kapitole tejto práce. V tejto sekcii sa budem venovať samotnému používateľskému testovaniu použiteľnosti, ktorému som podrobil implementovanú aplikáciu. V predošlej kapitole som uviedol, že pre správnu komunikáciu medzi mobilnou aplikáciou a serverom musela vzniknúť aplikácia na strane servera rozširujúca možnosti súčasnej produkčnej aplikácie na serveri. Táto novo vzniknutá aplikácia v čase plánovaného užívateľského testovania neabsolvovala bezpečnostnú previerku a penetračné testovanie, a z toho dôvodu nebolo možné túto aplikáciu nasadiť do stavu, kedy by bola verejne dostupná, a preto mohlo užívateľské testovanie prebehnúť iba v internom tíme spoločnosti productboard. Pre testovanie použiteľnosti som si vybral troch produktových manažérov a dvoch dizajnérov súčasnej webovej aplikácie. Aby testovanie nebolo zavádzajúce, zvolil som participantov, ktorí túto mobilnú aplikáciu ani jej návrhy nikdy predtým nevideli.

V rámci prípravy na užívateľské testovanie použiteľnosti som pripravil testovací scenár, v ktorom som systematicky definoval a popísal jednotlivé úlohy, ktoré som neskôr v priebehu testovania zadával participantom. Tieto úlohy pokrývajú zoznam požadovaných funkčných požiadaviek na aplikáciu z tretej kapitoly a pri ich zostavovaní som sa pridržiaval diagramov užívateľskej cesty, ktoré som vytvoril v rámci kapitoly o návrhu. Štandardnou súčasťou testovania použiteľnosti by mal byť vstupný dotazník, ktorý participant vyplní pred samotným testovaním a obsahuje prevažne otázky, na základe ktorých je možné participanta zaradiť do nejakej zo skupiny užívateľov testovanej aplikácie. Keďže v tomto prípade prebehlo testovanie s participantmi, u ktorých sú mi tieto údaje známe, rozhodol som sa vstupný dotazník vynechať. Každopádne pri ďalšom užívateľskom testovaní, ktoré bude prebiehať so zákazníkmi produktu, tento dotazník vypracujem a požiadam participantov o jeho vyplnenie. Vypracoval som ale výstupný dotazník, ktorého účelom je získať subjektívne pocity participanta z použiteľnosti aplikácie a celkový dojem z nej.

Výsledkom testovania je, že všetci participanti dokázali absolvovať všetky úlohy stanovené v testovacom scenári bez výrazných zdržaní a problémov. Medzi najproblematickejšie úlohy patrila úprava zobrazených informácií na domovskej obrazovke, pretože nastavenie nie je priamo viditeľné a prístupné zo samotnej obrazovky. Druhou problematickou úlohou bolo priradenie funkcionality do iného vydania. Dôvodom je, že táto interakcia nie je priamo viditeľná v užívateľskom rozhraní v zoskupení funkcionalít podľa vydania, ale ponuka akcií sa zobrazí po dlhom podržaní konkrétneho záznamu. Pri úlohe na vytvorenie novej poznámky a priradenie autora hodnotenia k tejto

poznámke taktiež niekoľkokrát padla relevantná pripomienka k tomu, že aplikácia neumožňuje pridať novú osobu alebo spoločnosť. Táto požiadavka sa v analýze potrieb užívateľov síce neobjavila, ale pri testovaní reálnej implementácie bolo jasné, že táto možnosť v aplikácii nesmie chýbať, aby aplikácia svojim užívateľom prinášala hodnotu. Ohliadnuc od týchto nedostatkov hodnotili participanti aplikáciu v rámci výstupného dotazníka pomerne kladne, ako z hľadiska použiteľnosti, tak s hľadiska celkového dojmu z aplikácie.

# Záver

V závere tejto práce by som chcel zhodnotiť, ako sa mi podarilo naplniť zadanie tejto diplomovej práce a ciele, ktoré som si stanovil v úvode tejto práce.

V tejto práci sa mi podarilo vypracovať výskum potrieb súčasných zákazníkov webovej aplikácie productboard v kontexte používania aplikácie productboard na mobilných zariadeniach, v rámci ktorej som zistil najdôležitejšie problémy zákazníkov, ktoré v súčasnej dobe majú pri používaní aplikácie na mobilnom zariadení, definoval som množinu funkcionalít, ktoré z výskumu vyšli ako najžiadanejšie a najdôležitejšie pre prvú verziu mobilnej aplikácie productboard.

V súlade s výsledkami výskumu som definoval funkčné a nefunkčné požiadavky, ktoré v neskoršej fáze tejto práce slúžili ako vstup pre návrh informačnej architektúry a interakcií užívateľského rozhrania pre mobilnú aplikáciu. Návrh bol vypracovaný s ohľadom na jednoduchosť riešenia, konzistenciu so súčasnou webovou podobou aplikácie, intuitívnosť užívateľského rozhrania a taktiež s ohľadom na použiteľnosť aplikácie.

Tento návrh som následne implementoval pomocou technológií, ktoré som zvolil, pretože verím, že vybrané technológie najlepšie vystihujú potreby implementovanej aplikácie a ich užívateľov. Taktiež som pri ich výbere bral do úvahy budúci rozvoj aplikácie a jej ďalší vývoj, na ktorom sa bude podieľať vývojový tím spoločnosti productboard.

Implementovanú aplikáciu som riadne otestoval a taktiež som ju podrobil expertnému a užívateľskému testovaniu použiteľnosti, ktorého výsledky sú uvedené v kapitole 6 Testovanie.

Vzhľadom k vyššie uvedeným faktom konštatujem, že sa mi podarilo naplniť ciele, ktoré som si v úvode tejto práce vytýčil, a tým som zároveň naplnil aj samotné zadanie tejto diplomovej práce.

Do budúceho rozvoja aplikácie jednoznačne odporúčam opravu nedostatkov zistených užívateľským testovaním použiteľnosti, následne opakované testovanie priamo so zákazníkmi produktu productboard a neskôr podrobenie aplikácie testovaniu výkonnosti a bezpečnosti. V prípade, že aplikácia v týchto

testovaniach uspeje, odporúčam sprístupniť aplikáciu ako alfa verziu pre vybranú podmnožinu zákazníkov. Ďalej odporúčam úpravy serverovej strany a implementácie mobilnej aplikácie tak, aby aplikácia ponúkla užívateľom notifikácie, ktoré boli zákazníkmi požadované, ale v súčasnej podobe serverovej aplikácie nerealizovateľné.

# Literatúra


[1] Cagan, M.: *INSPIRED: How to Create Tech Products Customers Love*. Wiley, 2017, ISBN 1119387507, str. 173, preklad vlastný.

[2] Ries, E.: Minimum Viable Product: a guide. `http://www.startuplessonslearned.com/2009/08/minimum-viable-product-guide.html`, 2009, preklad vlastný. [Online; cit. 7.10.2018].

[3] Semick, J.: 7 Strategies to Choose the Best Features for Your Product. `https://www.productplan.com/strategies-prioritize-product-features/`, 2018, preklad vlastný. [Online; cit. 7.10.2018].

[4] McBride, S.: RICE: Simple prioritization for product managers. `https://www.intercom.com/blog/rice-simple-prioritization-for-product-managers/`, 2018, preklad vlastný. [Online; cit. 20.10.2018].

[5] Zacarias, D.: 20 Product Prioritization Techniques: A Map and Guided Tour. `https://foldingburritos.com/product-prioritization-techniques/`, 2018, preklad vlastný. [Online; cit. 20.10.2018].

[6] Davies, B.: Product Roadmaps. `https://www.atlassian.com/agile/product-management/product-roadmaps`, 2018, preklad vlastný. [Online; cit. 20.10.2018].

[7] Porac, C.: *Laterality: Exploring the Enigma of Left-Handedness*. Academic Press, 2015, ISBN 978-0-12-801239-0, str. 2, preklad vlastný.

[8] Hendl, J.: *Kvalitativní výzkum*. Portál, 2005, ISBN 80-7367-040-2, str. 63, preklad vlastný.

[9] Wildemuth, B. M.: *Applications of Social Research Methods to Questions in Information and Library Science*. Libraries Unlimited, 2009, ISBN 9781591585039, s. 222–231, preklad vlastný.

[10] Perumal, T.: Quantitative Research Methods. https://www.tankonyvtar.hu/en/tartalom/tamop412A/2011-0021_22_research_methodology/adatok.html, 2014, preklad vlastný. [Online; cit. 13.11.2018].

[11] Sukamolson, S.: Fundamentals of Quantitative Research. http://www.culi.chula.ac.th/research/e-journal/bod/suphat%20sukamolson.pdf, 2018, preklad vlastný. [Online; cit. 14.11.2018].

[12] Babich, N.: A Comprehensive Guide To Mobile App Design. https://www.smashingmagazine.com/2018/02/comprehensive-guide-to-mobile-app-design/, 2018, preklad vlastný. [Online; cit. 30.11.2018].

[13] Bieller, E.: How To Design A Mobile App User Interface Like A Pro. https://careerfoundry.com/en/blog/ui-design/how-to-design-a-mobile-app-using-user-interface-design-principles/, 2018, preklad vlastný. [Online; cit. 4.12.2018].

[14] Mathis, L.: *Designed for Use: Create Usable Interfaces for Applications and the Web*. Pragmatic Bookshelf, 2011, ISBN 1934356751, preklad vlastný.

[15] Komninos, A.: 7 UX Deliverables: What will I be making as a UX designer? https://www.interaction-design.org/literature/article/7-ux-deliverables-what-will-i-be-making-as-a-ux-designer, 2019, preklad vlastný. [Online; cit. 5.2.2019].

[16] Rosenfeld, L.; Morville, P.; Arango, J.: *Information Architecture: For the Web and Beyond*. O'Reilly Media, 2015, ISBN 1491911689, str. 24, preklad vlastný.

[17] Babich, N.: A Beginner's Guide to Information Architecture for UX Designers. https://theblog.adobe.com/a-beginners-guide-to-information-architecture-for-ux-designers/, 2017, preklad vlastný. [Online; cit. 10.12.2018].

[18] Sternberg, R. J.: *Cognitive Psychology*. Cengage Learning, 2011, ISBN 1133313914, preklad vlastný.

[19] Human Interface Guidelines. https://developer.apple.com/design/human-interface-guidelines/ios/bars/search-bars/, 2018, preklad vlastný. [Online; cit. 20.12.2018].

[20] Human interface Guidelines. https://developer.apple.com/design/human-interface-guidelines/ios/icons-and-images/system-icons/, 2018, preklad vlastný. [Online; cit. 20.12.2018].

[21] Human Interface Guidelines. https://developer.apple.com/design/human-interface-guidelines/ios/app-architecture/settings/, 2018, preklad vlastný. [Online; cit. 20.12.2018].

[22] Part 1 – Understanding the Xamarin Mobile Platform. https://docs.microsoft.com/en-us/xamarin/cross-platform/app-fundamentals/building-cross-platform-applications/understanding-the-xamarin-mobile-platform, 2017, preklad vlastný. [Online; cit. 20.12.2018].

[23] Bres, Y.; Serpette, B. P.; Serrano, M.: Compiling scheme programs to .NET common intermediate language. *.NET Technologies 2004*, 2004: str. 25, preklad vlastný.

[24] Pedley, A.: How Flutter Works. https://buildflutter.com/how-flutter-works/, 2018, preklad vlastný. [Online; cit. 20.12.2018].

[25] Technical Overview. https://flutter.io/docs/resources/technical-overview, 2018, preklad vlastný. [Online; cit. 20.12.2018].

[26] Pedley, A.: How Flutter Works. https://buildflutter.com/how-flutter-works/, 2018, preklad vlastný. [Online; cit. 20.12.2018].

[27] Kuprenko, V.: Flutter vs. React Native: Choosing the Best Hybrid Framework for Mobile Development. https://dzone.com/articles/flutter-vs-react-native-how-to-choose-the-best-hyb, 2018, preklad vlastný. [Online; cit. 28.12.2018].

[28] ElHady, H.: Your Guide to Cross-Platform Mobile App Development Tools. https://instabug.com/blog/cross-platform-development/, 2018, preklad vlastný. [Online; cit. 30.12.2018].

[29] E-book: "Progressive Web Apps: the future of the mobile web". https://www.thinkwithgoogle.com/intl/en-gb/advertising-channels/mobile/announcing-progressive-web-apps-future-mobile-web/, 2018, preklad vlastný. [Online; cit. 5.1.2019].

[30] Eisenman, B.: *Learning React Native: Building Native Mobile Apps with JavaScript*. O'Reilly Media, 2017, ISBN 978-1-491-98914-2, preklad vlastný.

[31] Introducing JSX. https://reactjs.org/docs/introducing-jsx.html, 2019, preklad vlastný. [Online; cit. 5.1.2019].

[32] Abramov, D.: Motivation. https://redux.js.org/introduction/motivation, 2018, preklad vlastný. [Online; cit. 5.1.2019].

[33] Ighodaro, N.: Why use Redux? Reasons with clear examples. https://blog.logrocket.com/why-use-redux-reasons-with-clear-examples-d21bffd5835, 2018, preklad vlastný. [Online; cit. 5.1.2019].

[34] A query language for your API. https://graphql.org/, 2019, preklad vlastný. [Online; cit. 5.1.2019].

[35] Hunt, A.: *Pragmatic Unit Testing in C with NUnit, 2nd Edition (Pragmatic Starter Kit Series, Vol. 2)*. The Pragmatic Bookshelf, 2007, ISBN 0977616673, str. 3, preklad vlastný.

[36] Nielsen, J.: How to Conduct a Heuristic Evaluation. https://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation/, 1994, preklad vlastný. [Online; cit. 8.1.2019].

[37] Nielsen, J.: 10 Usability Heuristics for User Interface Design. https://www.nngroup.com/articles/ten-usability-heuristics/, 1994, preklad vlastný. [Online; cit. 8.1.2019].

Dodatok **A**

# Zoznam použitých skratiek

**RTF** Rich Text Format

**AOT** Ahead-of-time

**IL** Intermediate language

**HTML** Hyper Text Markup Language

**CSS** Cascading Style Sheets

**PWA** Progresívne webové aplikácie

**DOM** Document Object Model

**API** Application programming interface

**REST** Representational State Transfer

**SPA** Single page application

**HTML** Hypertext Markup Language

**E2E** End-to-End

**MVP** Minimum Viable Product

**URL** Uniform Resource Locator

Dodatok B

# Obsah priloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `impl` ..... zdrojové kódy implementácie
  - `thesis` ..... zdrojová forma práce vo formáte LaTeX
- `text` ..... text práce
  - `research` ..... materiály z užívateľského výskumu
  - `test` ..... materiály z užívateľského testovania použiteľnosti
  - `thesis.pdf` ..... text práce vo formáte PDF
  - `thesis.ps` ..... text práce ve formátu PS