FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Frontend optimalizace skladových procesů systému Atlantis
Student:	Vít Urban
Vedoucí:	Ing. Jiří Hunka
Studijní program:	Informatika
Obor / specializace:	Webové a softwarové inženýrství, zaměření Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

Cílem práce je tvorba frontendu rozšíření skladového systému Atlantis, věnované optimalizaci náročnosti skladových operací. Práce bude realizována ve spolupráci s Danielou Kováčovou, která v rámci své bakalářské práce realizuje backend stejného rozšíření.

Postupujte v těchto krocích:
1. Analyzujte současný stav skladového systému Atlantis se zaměřením na problematiku časové optimalizace nejčastějších procesů skladníků.
2. Na základě analýzy vhodně navrhněte nejpodstatnější rozšíření frontendové části.
3. Návrh projděte a řádně konzultujte minimálně se svým vedoucím a projektovým manažerem, který má na starost skladový systém Atlantis.
4. Na základě návrhu implementujte výsledné rozšíření.
5. Rozšíření řádně otestujte, minimálně na jednom reálném uživateli skladu.
6. Shrňte dosažené výsledky, navrhněte možná budoucí vylepšení dle získaných zkušeností.

Elektronicky schválil/a Ing. Michal Valenta, Ph.D. dne 17. ledna 2022 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Frontend optimalizace skladových procesů systému Atlantis

Vít Urban

Katedra softwarového inženýrství
Vedoucí práce: Ing. Jiří Hunka

11. května 2022

Poděkování

Rád bych poděkoval vedoucímu své práce Ing. Jiřímu Hunkovi za trpělivost a čas, který mi věnoval. Zároveň bych chtěl poděkovat své rodině a přítelkyni za neustálou podporu během celého studia i tvorby této práce.

Prohlášení

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

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

V Praze dne 11. května 2022 .....

Odkaz na tuto práci

Urban, Vít. *Frontend optimalizace skladových procesů systému Atlantis.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2022.

Abstrakt

Tato práce je zaměřena na analýzu, návrh a implementaci frontendu optimalizace současných skladových procesů systému Atlantis, s důrazem na korektní umístění zboží ve skladu. Tato práce je spjata s bakalářskou prací Daniely Kováčové, která na stejné téma vytvořila backendovou část rozšíření. Hlavním cílem této práce bylo navrhnout a implementovat vhodná rozšíření frontendové části, která by zrychlila časté skladové procesy. Práce analyzuje současný stav systému a zabývá se jeho nedostatky v oblastech umisťování zboží. Navrhuje možná rozšíření frontendové části, která se týkají především přesunů zboží na vhodné umístění. Výsledkem této práce je kromě jednotlivých návrhů i implementovaný a otestovaný frontend rozšíření, který je připraven na propojení s backendovou částí. Z této práce lze vycházet při budoucí tvorbě dodatečných optimalizací systému.

Klíčová slova skladový systém, frontend, uživatelské rozhraní, optimalizace skladových procesů, Vue.js, softwarový proces

Abstract

This thesis is focused on the analysis, design and implementation of fronted with a goal to optimalize current storage processes with emphasis on the correct placement of storage goods. This thesis is connected to the bachelor thesis made by Daniela Kováčová, which focuses on backend implementation of the same topic. The main goal of this thesis was to design and implement a suitable extension of the frontend part of the storage system Atlantis which would speed up common storage processes. This thesis analyzes the current state of the system and focuses on its shortcomings in the correct placement of goods. The thesis designs possible extension of the frontend part which mostly relates to product movements to the correct location. The result of this work is implemented and tested frontend part of the extension and is ready to be connected to its backend part. This thesis can be used for future optimalization of the storage system.

Keywords storage system, frontend, user interface, optimalization of storage processes, Vue.js, software process

Obsah

Úvod **1**

1 Analýza **3**
1.1 Rizika . . . 3
1.2 Modely softwarového procesu . . . 4
1.2.1 Vodopád . . . 4
1.2.2 Agilní modely . . . 5
1.2.3 Naše využití . . . 5
1.3 Metody sběru požadavků . . . 6
1.3.1 Analýza dokumentace . . . 7
1.3.2 Interview . . . 7
1.3.3 Pozorování pracovního prostředí . . . 8
1.3.4 Brainstorming . . . 8
1.4 Analýza současného stavu . . . 8
1.4.1 Informace od zadavatele . . . 8
1.4.2 Průchod systémem a analýza dokumentace . . . 9
1.4.3 Terénní výzkum . . . 10
1.4.4 Nejpodstatnější rozšíření . . . 11
1.5 Požadavky . . . 12
1.5.1 Funkční požadavky . . . 12
1.5.2 Nefunkční požadavky . . . 16

2 Návrh řešení jednotlivých požadavků **17**
2.1 Rozměry umístění . . . 18
2.2 Rozměry produktů . . . 18
2.3 Vytíženost produktů . . . 19
2.4 Dostupnost umístění . . . 20
2.5 Přesuny zboží . . . 21
2.5.1 Přesun zboží . . . 21

2.5.2 Výměna zboží . . . 22
2.5.3 Sloučení zboží . . . 22
2.5.4 Seskupení zboží, které se prodává často společně . . . 22
2.6 Proces naskladnění . . . 22
2.7 Doplnění kusového zboží . . . 23

3 Realizace **25**
3.1 Použité technologie . . . 25
3.1.1 JavaScript . . . 25
3.1.2 Vue.js . . . 26
3.1.3 Mock Service Worker . . . 26
3.1.4 PhpStorm . . . 27
3.2 Implementace návrhů . . . 27
3.2.1 Výskyt a řešení problémů . . . 27
3.2.2 Rozměry produktů . . . 28
3.2.3 Rozměry umístění . . . 29
3.2.4 Vytíženost produktů . . . 30
3.2.5 Dostupnost umístění . . . 30
3.2.6 Návrhy na přesun zboží . . . 31

4 Testování **33**
4.1 Uživatelské testování . . . 33
4.2 Scénáře . . . 34
4.2.1 TS1 - Nové umístění . . . 34
4.2.2 TS2 - Úprava produktu . . . 34
4.2.3 TS3 - Návrh na přesun . . . 34
4.2.4 TS4 - Návrh na sloučení . . . 35
4.2.5 TS5 - Doplnění kusového zboží . . . 35
4.3 Průběh a vyhodnocení testování . . . 35
4.3.1 Průběh . . . 35
4.3.2 Vyhodnocení . . . 36
4.4 Úpravy po testování . . . 37

5 Možnosti navázání **39**

Závěr **41**

Bibliografie **43**

A Seznam použitých zkratek **47**

B Návrhy obrazovek **49**

C Výsledné obrazovky **55**

D Obsah přiloženého CD **59**

Seznam obrázků

1.1 Sprint 6
1.2 Atributy 9

2.1 Návrh tvorby nového umístění 18
2.2 Návrh zobrazení vytíženosti 19
2.3 Návrh zobrazení dostupnosti 20

3.1 Definice zachycení požadavku pomocí Mock Service Worker 26
3.2 Zobrazení rozměrů pro skladové umístění 29
3.3 Možnost úpravy vytíženosti v editaci produktu 30

B.1 Návrh zobrazení rozměrů 49
B.2 Návrh zobrazení rozměrů u produktů 50
B.3 Návrh obrazovky s přesunem 51
B.4 Návrh obrazovky s výměnou 52
B.5 Návrh obrazovky se sloučením 53

C.1 Výsledná obrazovka s návrhem na doplnění kusového zboží 55
C.2 Výsledná tabulka bez zobrazení detailů 56
C.3 Výsledná tabulka s detaily 56
C.4 Výsledná obrazovka s editací produktu 57
C.5 Výsledná obrazovka s editací produktu a detaily 57
C.6 Výsledná obrazovka návrhu na přesun po testování 58

Úvod

Události posledních let ukázaly důležitost fungování internetových obchodů a nákupů v online prostředí. Prakticky žádný e-shop by ale nemohl fungovat bez precizní práce skladu. Tyto sklady ke své činnosti často využívají různé systémy, které jim napomáhají dosáhnout bezchybného běhu všech procesů. Jedním z nich je právě skladový systém Atlantis. Tento systém uchovává veškeré informace o všech procesech, které se ve skladu dějí a zajišťuje jeho bezproblémový chod. V některých aspektech, by systém mohl ulehčovat a zrychlovat různé procesy ještě více. Tomuto problému jsem se rozhodl věnovat v mé bakalářské práci.

Výsledek této práce bude ulehčovat, a především zrychlovat fungování skladů, které využívají skladový systém Atlantis. Mé rozšíření jim napomůže v organizaci zboží a díky tomu i lepšímu časovému rozložení častých skladových procesů.

Hlavním cílem této práce je vytvořit frontend k rozšíření již používaného skladového systému Atlantis, které zlepší časovou režii nejčastějších skladových procesů a umožní tak pracovníkům lépe rozložit čas mezi jednotlivé aktivity. Důležité je zjistit, co přesně by ocenili přímo uživatelé skladového systému, jejich požadavky zanalyzovat a rozhodnout, zdali existuje rozumné řešení, které by je pokrylo. Dalším cílem je tato řešení navrhnout takovým způsobem, aby zapadla do současné implementace skladového systému. Je tak na místě zdůraznit, že se jedná o rozšíření již plně funkčního a stabilního systému. Významným prvkem implementace bude tedy udržení současného stylu, a to jak ve formě kódu, tak i zobrazení jednotlivých rozšíření. Posledním cílem je zajistit, aby rozšíření bylo jednak užitečné z hlediska zrychlení procesů, tak uživatelsky přívětivé. Oba tyto záměry by mělo pokrýt závěrečné uživatelské testování.

Tato práce částečně navazuje na diplomovou práci Ing. Oldřicha Malce, který vytvořil frontend skladového systému Atlantis a na bakalářské práce Bc. Denise Talára a Bc. Jana Cvrčka, kteří ve své práci popisují možnosti

pro optimalizaci systému z různých hledisek. Důležité je zmínit, že analýza vychází z poznatků, které jsme učinili v rámci předmětu Softwarový týmový projekt 1. V týmu, kde jsem spolupracoval s Danielou Kováčovou, Jakubem Volákem, Michalem Kovářem, Janou Karafiátovou a Dominikem Kyjevským, jsme pod dohledem Ing. Jiřího Hunky vytvořili podklady pro možnou optimalizaci systému s důrazem na korektní umístění zboží. Implementační část částečně navazuje na předmět Softwarový týmový projekt 2, kde Janu Karafiátovou a Dominika Kyjevského nahradili Jakub Meinlschmidt a Jiří Folprecht. V tomto předmětu jsem působil jako vedoucí a dohlížel jsem na základy implementace dříve navržených částí. Celá práce je propojená s bakalářskou prací Daniely Kováčové, která vytváří backend k optimalizaci skladových procesů systému Atlantis.

Práce je rozdělená na několik kapitol. Ta první je věnována analýze, kde kromě popisu metod, které jsme v rámci našeho týmu využívali, popisuji především části skladového systému, které by bylo vhodné optimalizovat. Stejně tak se zaměřuji na procesy, které mohou trvat zbytečně dlouhou dobu a na popis jednotlivých požadavků. V návrhu poté popisuji možná řešení těchto nedostatků a představuji návrh obrazovek pro jednotlivé části optimalizace.

Následuje kapitola o realizaci navržených optimalizací. Nejprve se zmiňuji o technologiích, které využiji při realizaci návrhu a které mi napomohou v dosažení mých cílů. Následně objasňuji, jakým způsobem probíhala samotná implementace. Poslední kapitola je věnována uživatelskému testování. Nejprve se v ní zaměřuji na způsob, jakým uživatelské testování funguje a které části jsou důležité a následně popisuji, jak samotné testování probíhalo, jaké chyby odhalilo a jeho celkový přínos.

V závěru zhodnocuji výsledky mé práce a také které cíle a požadavky se podařilo naplnit a představuji různé možnosti pro pokračování a navázání na mou práci. Tedy především jaké části skladového systému by bylo vhodné v budoucnu optimalizovat.

Kapitola 1

Analýza

V této kapitole se zaměřuji především na analýzu současného stavu skladového systému Atlantis. Popisuji možné nedostatky v rámci jednotlivých skladových procesů s důrazem na ukládání zboží. Nejprve se zabývám různými modely, které popisují postupy, jakými lze vyvíjet software a následně přibližuji které a jakým způsobem jsme využili. Zároveň také popisuji metody, které se používají pro sběr požadavků a jak jsme je využili pro odhalení částí systému, které by bylo vhodné optimalizovat. Při zjišťování informací o fungování systému vycházím z diplomových prací Ing. Oldřicha Malce a Ing. Pavla Kováře, kteří tento systém vytvořili [1] [2]. Samotný systém je realizován pomocí dvou samostatných celků, frontendu a backendu, které spolu komunikují pomocí REST API, které je detailně popsáno v interní dokumentaci celého projektu. Při zkoumání jednotlivých procesů a následném návrhu řešení vycházím i z tohoto zdroje.

1.1 Rizika

Jelikož je tato práce realizována ve spolupráci s Danielou Kováčovou, která v rámci své bakalářské práce realizuje backend stejného rozšíření, je zde riziko, že se mohou vyskytnout problémy týkající se synchronizace společné činnosti a průběžného propojování obou celků. V ideálním scénáři bude docházet k pravidelné komunikaci a při implementaci frontendu budu schopný využívat rozpracovanou či již funkční část odpovídající backendové implementace. V tomto případě by bylo možné výsledek naší implementace společně otestovat na reálných uživatelích a výsledek našich prací by byl připraven k nasazení. Je ale nutné předvídat možné problémy, které v rámci spolupráce mohou vzniknout. Je nutné dopředu vědět, jakým způsobem bude prováděna implementace nebo testování, jestliže se implementace jedné či druhé strany prodlouží a nebude tak možné průběžně propojovat obě části. Pokud by nastala situace, při které by byla implementace backendové části opožděna, je důležité mít předem definované API, které bude následně implementováno

nezávisle na obou celcích. Pro implementaci frontendu by to znamenalo nasimulovat odpovědi backendu pomocí takzvaného mocku. To znamená definovat přesné odpovědi backendu bez provedení konkrétních výpočetních operací či komunikace s databází. Nevýhodou tohoto řešení je především nemožnost rychlého nasazení. Pokud by byl frontend realizován tímto způsobem, muselo by být následně provedeno větší množství úprav, než by bylo celé rozšíření připraveno k použití. Výhodou je samozřejmě to, že by realizace i testování mohly probíhat nezávisle na stavu implementace backendové části rozšíření.

1.2 Modely softwarového procesu

Pro zajištění funkčního softwarového vývoje existuje několik modelů, které popisují průběh softwarového procesu. V následujících částech představuji především ty, které se v dnešní době používají nejvíce a které jsme využívali v rámci naší práce. Na konci této sekce popisuji, které z modelů jsme při vývoji použili a z jakého důvodu.

1.2.1 Vodopád

Mezi tradiční a dříve často používané modely se řadí vodopádový model. Jedná se o sekvenční vývojový model, který postupně prochází jednotlivé části vývoje. První částí je analýza požadavků. V ní se sbírají a analyzují jednotlivé požadavky, které následně definují chování systému. Po analýze těchto požadavků se připraví dokumentace, která následně napomáhá v celém vývoji. Dalším krokem je design. Tato fáze vychází z výsledků té předchozí a zahrnuje například navrhování databázových schémat nebo celkové architektury systému. Po této části přichází implementace, kde je hlavním cílem naprogramovat systém, který splňuje všechny dříve definované požadavky a koresponduje s vytvořenými návrhy. V předposlední fázi se tato implementace testuje. Zde se snažíme zjistit, jestli implementace odpovídá požadavkům a případně najít chyby a nedostatky systému, které následně opravíme. Poslední částí je údržba, ve které se staráme o námi vytvořený a používaný systém. Můžeme přidávat nové úpravy nebo opravovat nově zjištěné chyby.[3]

Mezi hlavní výhody vodopádu patří jasné definování požadavků již na začátku vývoje a díky tomu lze jednoduše odhadnout celková náročnost celého procesu a predikovat tak důležité faktory jako například cenu nebo rozsah celého projektu. Zároveň se jedná o jednoduchý lineární model, který je vcelku snadný na implementaci. Největší nevýhodou je však to, že se celý projekt vytváří z požadavků definovaných na začátku celého projektu a velice špatně tak reaguje na možné změny či úpravy od zadavatele. [4]

1.2.2 Agilní modely

Zatímco tradiční modely se zakládají především na schopnosti predikovat průběh celého projektu, agilní modely používají spíše adaptivní vývojové metody. Tím je myšlené především to, že tým s agilním přístupem velice často komunikuje se zadavatelem a průběžně upravuje jednotlivé požadavky. Software se v takovém případě vyvíjí po menších částech, velice často se testuje a podléhá pravidelné kontrole. Nedílnou součástí jsou také časté schůzky celého týmu, ve kterých se analyzuje, jaký byl posun v jednotlivých úkolech [5]. Největší výhodou agilního přístupu oproti tradičnímu je právě velice rychlá reakce na změnu požadavků a na zpětnou vazbu od zadavatele celého projektu. Aby některý z agilních modelů fungoval, je nezbytně nutné kontinuální zapojení všech členů týmu.

1.2.2.1 Scrum

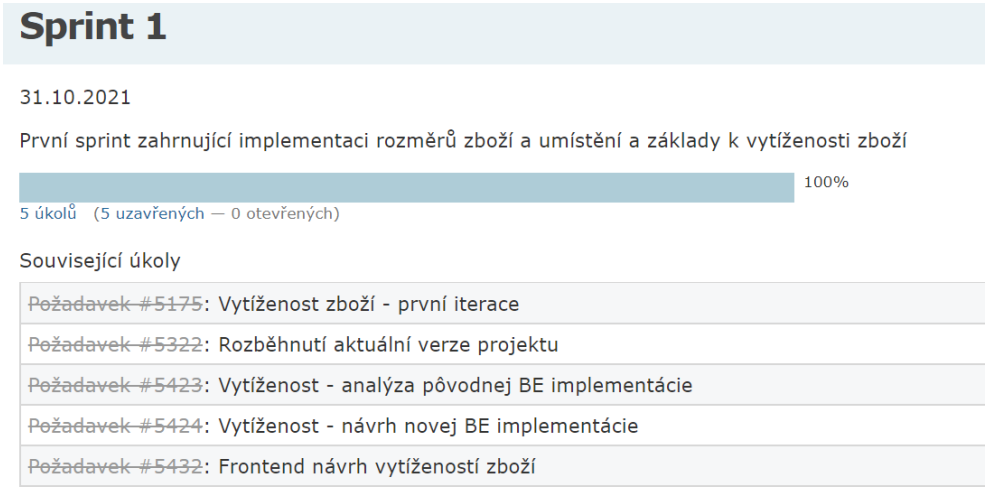
Mezi nejpoužívanější agilní model se řadí scrum. Tento model je charakteristický tím, že pro vývoj používá kratší časové intervaly, takzvané sprinty[6], které trvají maximálně měsíc. Na začátku každého sprintu se určí, které úkoly mají být řešeny a hlavní snahou je je v daném sprintu vyřešit. Během sprintu se každý den pořádají schůzky, kde se diskutuje o pokroku v jednotlivých částech vývoje a upravuje se plán, aby byl sprint úspěšně ukončen. Po konci každého sprintu přichází vyhodnocení a představení výsledků zákazníkům. Podle zpětné vazby se následně vytvoří prioritizovaný seznam úkolů pro další sprint, do kterého se přesouvají i problémy, které se nepodařilo v předchozím sprintu dokončit.[7, chap.1]

Součástí scrum modelu je definice tří hlavních rolí účastníků. Tou první je product owner, jehož hlavní cíl je maximalizovat návratnost investic hlavně tím, že vytváří seznam požadavků pro každý sprint a určuje jejich prioritu. Je také zodpovědný za prezentování pravidelných výsledků a za komunikaci se zákazníky. Druhou rolí je scrum master. Ten je zodpovědný především za bezproblémový chod vývojového týmu. Jeho úkolem je připravit vhodné prostředí pro nerušený vývoj a zároveň se snažit o to, aby tým i product owner dodržovali praktiky scrum. Poslední částí je vývojový tým. Počet jeho členů se většinou pohybuje okolo sedmi a jejich hlavním cílem je vývoj daného projektu. [7, chap.1]

1.2.3 Naše využití

Při práci na optimalizaci skladového systému, jsme v týmu z počátku využívali spíše vodopádový model. Nejprve jsme analyzovali, jak můžeme systém vylepšit. Následně jsme vytvořili seznam požadavků, které jsme zanalyzovali a poté jsme připravili návrhy jak obrazovek, tak algoritmů, které jsme chtěli využívat pro vývoj a které zároveň popisuji v následující kapitole. Po celou dobu jsme ale využívali i některé prvky z agilního vývoje, jelikož jsme vše

pravidelně konzultovali s vedoucím práce, tedy zadavatelem celého projektu a aktivně jsme reagovali na jeho podněty a připomínky. V implementační fázi jsme využili scrum model a dělili jsme práci na jednotlivé sprinty, které trvaly čtrnáct dní. Po každém zakončení sprintu jsme vyhodnotili náš dosavadní postup a rozdělili si úkoly na další sprint. V tomto případě jsem jakožto vedoucí tohoto týmu zastával několik rolí. Byl jsem členem vývojového týmu a podílel jsem se především na implementaci frontendové části. Zároveň jsem působil jako pomyslný product owner, jelikož jsem prezentoval naše výsledky vedoucímu celého projektu a určoval prioritu jednotlivých úkolů. Částečně jsem působil také jako scrum master, jelikož jsem komunikoval se všemi členy týmu a řešil s nimi problémy či nejasnosti. Zároveň jsem dohlížel na dodržování scrum modelu a plnění úkolů v rámci jednotlivých sprintů. Záznam prvního sprintu je možné vidět na obrázku 1.1.



Obrázek 1.1: Sprint

Po zakončení předmětů SP1 a SP2 jsme spolu s Danielou Kováčovou znovu navázali spíše na vodopádový model a implementace a testování probíhaly z dříve definovaných požadavků.

1.3 Metody sběru požadavků

Pro určení možných rozšíření a pro korektní optimalizaci systému je nutné zjistit aktuální nedostatky systému. V tomto případě to může být jak z pohledu pracovníků skladu, kteří mohou vidět problém v určité části fungování systému, tak například z pohledu vývojářů systému Atlantis. V této sekci popisuji různé způsoby sběru požadavků, které nám identifikují možné nedostatky. Zaměřuji se především na metody, které jsme využili v rámci analýzy současného stavu systému.

1.3.1 Analýza dokumentace

Analýza dokumentace spočívá v jednoduchém principu procházení dostupných materiálů, které jsme schopni získat od zadavatele daného projektu. Hlavním účelem je pochopení fungování daného subjektu, pro který chceme vytvářet software. Jednat se může o smlouvy, formuláře nebo také současnou dokumentaci informačního systému, pokud nějaký již existuje. Nevýhodou může být, že se jedná o dokumenty, které odrážejí současnou verzi systému a můžeme tak získat zastaralé a do budoucna nepotřebné informace. Oproti tomu výhodou může být, že si sami vývojáři rozhodují o důležitosti informací. [8]

1.3.2 Interview

Interview můžeme zařadit do konzervativních metod sběru požadavků. Konzervativní metody jsou takové, které využívají přímou komunikaci mezi dvěma a více subjekty. Jedná se o přirozenou cestu, při které se pokládají a zodpovídají otázky a která často vede k jasnému a efektivnímu definování možných problémů a požadavků. [9]

Samotné interview pak spočívá v připraveném rozhovoru s danou osobou, při kterém získáme informace o konkrétních požadavcích. Rozhovor se, ve většině případů, skládá z několika fází.

- **Určení účastníků:** Jako první je nutné určit účastníky interview. Ve většině případech se začíná se zadavatelem celého projektu, který může následně doporučit další vhodné osoby k rozhovoru. [10, chap.4]
- **Příprava:** Následně je nutné připravit si předem otázky a materiály k rozhovoru a dohodnout si s účastníky čas a místo uskutečnění. Je také vhodné si předem zjistit základní informace o daném projektu, například analýzou dokumentace, aby nedocházelo k zbytečnému prodlužování interview. [11]
- **Rozhovor:** Tato fáze začíná představením všech účastníků po kterém tazatel pokládá jednotlivé otázky.
- **Ukončení:** Po ukončení rozhovoru by měla následovat krátká debata nad zjištěnými informacemi, aby se tazatel utvrdil v jejich správnosti. [11]
- **Oponentura:** Poslední fáze spočívá ve vytvoření výsledné zprávy tazatelem, kterou následně zašle všem účastníkům a tím si potvrdí korektnost výsledků rozhovoru. [10, chap.4]

1.3.3 Pozorování pracovního prostředí

Pozorování pracovního prostředí je jedna z metod sběru požadavků, která spočívá v pozorování a zaznamenávání jednotlivých procesů a náplně práce daného subjektu, pro který vytváříme software. Sloužit může k utvrzení informací získaných z jiných metod, například interview nebo analýzy dokumentace, ale také k zjištění nových požadavků. Výhodou je, že získané informace vyobrazují aktuální stav jednotlivých procesů. Nevýhodou může být časová náročnost pozorování nebo to, že procesy, které se neprovádějí tak často, nemusí být zpozorovány. Je tak důležité nepoužívat pro zisk požadavků pouze tuto metodu. [8]

1.3.4 Brainstorming

Brainstorming je metoda, při které se setká skupina lidí a vytváří co největší množství návrhů. Důležité je, aby se na tvorbě podíleli všechny zainteresované strany celého projektu, pro který se vytváří požadavky. Brainstorming se většinou skládá ze dvou částí. V té první je snaha o vygenerování co největšího počtu návrhů, bez jakéhokoliv kritizování či zavrhování a v té druhé by měla nastat diskuze nad jejich relevantností k tématu a jejich celkové zhodnocení. [12]

1.4 Analýza současného stavu

Systém jsme analyzovali několika způsoby. V této sekci popisuji, jaké metody jsme k tomu využili a na konci shrnuji k jakým závěrům jsme dospěli. Samotné požadavky, které z této analýzy vyplývají budou popsány až v následující sekci.

1.4.1 Informace od zadavatele

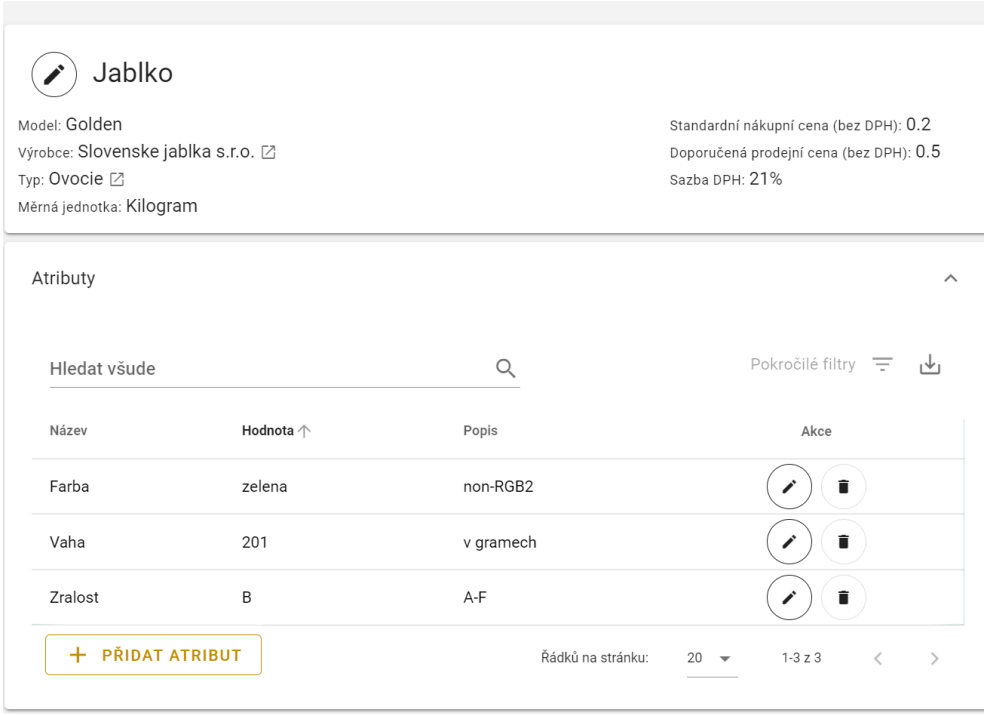
Prvotním zadáním, které jsme obdrželi od našeho vedoucího projektu Ing. Jiřího Hunky, bylo zanalyzovat aktuální stav systému. Dále pak najít možnosti urychlení a optimalizování častých skladových procesů s pomocí informací o rozměrech zboží a umístění, frekvenci přesunů zboží a dostupnosti umístění. Nejprve jsme v rámci týmu použili metodu brainstorming, ve které jsme se snažili zaměřit na různé procesy, které jsou obecně využívané ve většině skladových zařízeních a výsledek jsme konzultovali s vedoucím projektu. Dospěli jsme k tomu, že nejčastější procesy, které se často provádí a bez kterých se neobejde žádný sklad, souvisí především s umísťováním skladových položek, jejich naskladněním, přesouváním či vyskladněním. Takový proces může být velice neefektivní, pokud se jedná o větší sklad a umisťování jednotlivých položek není žádným způsobem řízeno. V tu chvíli by se mohlo stát, že skladník musí chodit velice často pro vyskladňované položky až na konec skladu a s položkami, které má blízko, příliš často nepracuje. Rozhodli jsme se tedy

zaměřit na proces ukládání položek na dostupná umístění podle jejich frekvence vyskladnění. Pro další účely naší analýzy a následné implementace jsme dostali přístup k současné verzi skladového systému Atlantis a také k jeho dokumentaci.

1.4.2 Průchod systémem a analýza dokumentace

1.4.2.1 Rozměry

Při procházení aktuální verze systému jsme se nejprve zaměřili na možnosti uchovávání informací o rozměrech zboží a umístění, jelikož jsou tato data nutná pro doporučení korektního umístění skladových položek. Pro jednotlivé produkty lze uchovávat rozměry pomocí takzvaných atributů. Ke každému produktu lze přiřadit atribut, který se skládá z názvu, hodnoty a popisu, jak lze vidět na obrázku 1.2. Pro přidání nového atributu je nutné vyplnit alespoň název a hodnotu.



Obrázek 1.2: Atributy

Aktuálně se tyto atributy pro uchovávání informací o rozměrech používají pouze zřídka a pro hromadné využití za účelem optimalizace se tato možnost nejeví jako nejideálnější řešení. Pro jednotlivá skladová umístění v tuto chvíli neexistuje možnost, jak rozměry uchovávat.

1.4.2.2 Frekvence vyskladnění jednotlivých položek

Následně jsme se zaměřili na to, jestli existuje způsob, jakým získat data pro určení frekvence vyskladnění jednotlivých položek. V aktuální verzi systému se v databázi uchovávají jednotlivá vyskladnění včetně jeho položek. Pro každý produkt tak můžeme získat informaci o počtu jeho vyskladnění v určitém časovém intervalu a pro naše účely je tato skutečnost naprosto dostačující.

1.4.2.3 Dostupnost umístění

Každý sklad má určitá umístění. Umístění má v současné verzi název a kód. Zároveň se k němu dá připojit informace o tom, zda se jedná o mobilní nebo balící umístění. Informace o jeho poloze lze částečně vyčíst z jeho názvu, který by měl odpovídat jeho umístění ve skladu. Například název *A02-2-3* znamená řada A, blok 2, druhý sloupec, třetí pozice od podlahy. Pro určení dostupnosti však tato informace nestačí.

Zjistili jsme ale, že současně pracuje jeden z týmů, také pod vedením Ing. Jiřího Hunky, na možnosti tvorby mapy pro jednotlivé sklady v rámci systému Atlantis. Po společné konzultaci jsme dospěli k závěru, že nám budou schopni poskytnout vzdálenosti dvou či více umístění nebo výšku jakéhokoliv umístění. Tato informace nám postačí k celkovému zjištění dostupnosti každého umístění.

1.4.2.4 Proces naskladnění

Následně jsme zjišťovali, jakým způsobem lze provádět naskladnění. Pokud bychom chtěli korektně umisťovat produkty dle frekventovanosti jejich vyskladnění, bylo by vhodné je korektně umisťovat již při jejich naskladnění, aby nedocházelo k dodatečným přesunům. V současné verzi systému lze uskutečnit naskladnění dvěma způsoby. Tím prvním je naskenování vícero položek, které se nacházejí v jednom naskladnění a následné naskenování určitého umístění a přesunutí položek na toto umístění. Tím druhým je opačný přístup, při kterém se nejprve naskenuje umístění, na kterém chceme produkty uskladnit a až poté se skenují jednotlivé položky.

1.4.3 Terénní výzkum

Pro potvrzení našich dosavadních poznatků a jejich případné rozšíření jsme se dohodli s vedoucím projektu, že spolu s dalším týmem, který má na starost právě implementaci mapy skladu, navštívíme jeden ze skladů, který využívá skladový systém Atlantis, budeme mít možnost zeptat se na několik otázek vedoucího daného skladu a prohlédnout běžné pracovní činnosti skladníků. Jednalo se o kombinaci metod interview a pozorování pracovního prostředí. Pro korektní provedení interview bychom potřebovali více času pouze s vedoucím a pro samostatné pozorování pracovního prostředí by bylo nutné návštěvu několikrát zopakovat. V samotném skladu se nám a našim otázkám věnoval

vedoucí skladu. Rozhovor probíhal přímo na pracovišti a mohli jsme tak během něj pozorovat i práci jednotlivých skladníků. Poté jsme si prohlédli sklad a způsob, jakým se ukládají jednotlivé položky. Výsledkem tohoto pozorování bylo několik poznatků. Jednak jsme se utvrdili v tom, že aktuální způsob umisťování není řízen dle žádného systému a tato optimalizace by byla vítaná. Zároveň nám vedoucí sdělil, že další proces, který se často provádí a trvá delší dobu je každodenní doplnění kusového zboží. Větší balení skladových položek jsou uchovávána na paletách ve vyšších patrech skladu. Kusové produkty jsou uloženy ve spodních patrech, aby skladníci při vyskladnění nemuseli používat vysokozdvižné vozíky, mohli pouze projít spodní patra a připravit dané zboží k vyskladnění. Aktuálně se ale musí každé ráno projít spodní patra, aby se zjistilo, kterého kusového zboží je málo a je ho tak nutné doplnit z vyšších pater. Tento proces by tedy bylo také vhodné optimalizovat.

1.4.4 Nejpodstatnější rozšíření

V rámci našeho týmu jsme dospěli k názoru, že nejzásadnějším rozšířením, které zrychlí časté skladové procesy, je především korektní umístění produktů. Tím se myslí především to, že zboží, které se často prodává, by se mělo nacházet na dostupnějším umístění než zboží, se kterým se tolik nepracuje. Abychom toho dosáhli, je nutné zajistit několik menších částí. Musíme evidovat rozměry jak skladových položek, tak jednotlivých umístění, aby nedocházelo ke kolizím z hlediska velikosti. Dále je nutné vytvořit několik algoritmů, které vypočítají dostupnost jednotlivých umístění a vytíženost jednotlivých produktů. Tyto informace následně použijeme k tomu, abychom doporučili vhodné umístění pro každý produkt. Tím dosáhneme časové úspory procesu vyskladnění i naskladnění, jelikož produkty, se kterými se často manipuluje, budou uloženy blíže umístěním, která se pro tyto procesy využívají.

Další rozšíření, které by ušetřilo čas, by bylo již zmiňované upozornění na docházející kusové zboží. Tento proces by se rapidně zrychlil, pokud by systém každý den připravil seznam docházejících kusových produktů a nemusel by se každé ráno procházet sklad pro získání této informace.

Tyto části popisuji v podobě požadavků v následující sekci, kde jim kromě stručného vysvětlení určuji náročnost a prioritu, se kterou by měly být realizovány.

1.5 Požadavky

Z analýzy současného stavu vyplynulo několik požadavků, kterými se v této sekci zabývám. Nejprve stručně představuji, co je hlavním cílem požadavku a následně ho detailněji popisuji případně vysvětluji z jakého důvodu jsem se rozhodl ho vytvořit. Poté každému požadavku určuji prioritu, s jakou by se měl řešit a odhaduji náročnost jeho realizace. Jelikož je tato práce zaměřená na frontend rozšíření skladového systému, bude náročnost odpovídat odhadu náročnosti implementace uživatelského rozhraní a jednotlivých obrazovek. Neberu tedy příliš v potaz složitost jednotlivých algoritmů, které jsou potřeba pro celkové řešení požadavku.

1.5.1 Funkční požadavky

1.5.1.1 FP1: Rozměry umístění

Systém bude umožňovat uchovávání informací o rozměrech umístění. Tím se konkrétně myslí výška, šířka, délka a nosnost.

Hlavním cílem je optimalizovat umístění skladových položek a doporučit pracovníkům skladu, jaká místa by měla být využita pro uložení daného produktu. Tyto varianty, ale musí být proveditelné, a proto je nezbytně nutné uchovávat si informace o velikosti umístění. V opačném případě by mohl nastat problém, že se na nové umístění zboží nevejde. V aktuální verzi skladového sytému se tyto informace neevidují.

Priorita : Velmi vysoká
Náročnost : Nízká

Priorita je zvolena jako velmi vysoká, jelikož se jedná o základní stavební jednotku celé optimalizace. Bez informace o rozměrech není možné navrhovat jakékoliv změny v umístění zboží, jelikož by mohly nastat kolize v rámci velikosti umístění a daného zboží. Náročnost je naopak nízká, jelikož se jedná o drobné úpravy a přidání několika atributů k již vytvořené entitě.

1.5.1.2 FP2: Rozměry produktu

Systém bude umožňovat uchovávání informací o rozměrech produktu. Tím se konkrétně myslí výška, šířka, délka a hmotnost.

Stejně tak jako u umístění, je i u produktů nutné evidovat jejich rozměry. Místo nosnosti zde hodláme brát v patrnost váhu jednotlivých produktů. Jediná možnost, jak tyto informace nyní zaznamenat je pomocí atributů, o kterých se zmiňuji v minulé sekci. Pro optimalizaci je toto řešení však nepoužitelné, jelikož jsou atributy volitelné a nemusí nutně zaznamenávat pouze

rozměry. Pro rychlé fungování systému by toto mohlo znamenat překážku, tudíž je vhodné zavést rozměry jako samostatnou část. Zároveň jsme se rozhodli tyto informace uchovávat přímo pro čárové kódy jednotlivých produktů, a nikoliv pro produkty samotné. Může se stát, že některé zboží se skladuje v několika různých variantách. Může jich být více v jednom balení, ale zároveň se může uskladňovat po jednotlivých kusech. Každá tato varianta má tím pádem různé rozměry a musíme je evidovat pro každou zvlášť.

Priorita : Velmi vysoká
Náročnost : Nízká

Priorita i náročnost je zvolena stejně jako u předchozího požadavku, jelikož se jedná o velice podobný problém.

1.5.1.3 FP3: Vytíženost produktu

Systém bude evidovat vytíženost produktu. Tedy jak často je vyskladněn v rámci určitého uplynulého časového intervalu.

Abychom dosáhli vhodného umístění produktů, musíme vědět, jak často se s jakým zbožím pracuje. K tomuto nám pomůže právě vytíženost zboží. Tato hodnota bude vázaná k jednotlivým produktům a bude vycházet z počtu jeho vyskladnění za určitý časový interval.

Priorita : Velmi vysoká
Náročnost : Nízká

Stejně jako v předchozích požadavcích se i v tomto případě jedná o velice důležitou součást celé optimalizace. Je nezbytně nutné vědět, jak často se s daným produktem manipuluje, aby mohl systém korektně určit jeho vhodné umístění. Náročnost je nízká, jelikož se znovu jedná o přidání atributu, k již vytvořené entitě.

1.5.1.4 FP4: Dostupnost umístění

Systém bude evidovat dostupnost umístění. Tedy, jak vzdálené je umístění od důležitých míst.

Pro uložení produktů na vhodná umístění, je, kromě vytížení produktů, nutné zjistit, která umístění jsou lépe dostupná a která hůře. Informace by měla vycházet ze vzdálenosti od takzvaných důležitých míst. Tím může být například balící místo, místo určené k vyskladnění nebo naskladnění zboží, ale i místo, odkud pracovníci skladu vycházejí pro jednotlivé produkty.

Priorita : Velmi vysoká
Náročnost : Nízká

Jedná se o podobný požadavek jako FP3, tudíž priorita i náročnost zůstávají stejné.

1.5.1.5 FP5: Přesun zboží

Systém bude navrhovat vhodné přesuny zboží.

Jelikož díky realizaci předchozích požadavků budeme znát informace o tom, se kterými produkty se pracuje často, jak jsou veliké a která umístění jsou dobře dostupná, můžeme doporučit pracovníkům skladu, jak korektně umisťovat jednotlivé produkty. Jelikož se ale jedná o rozšíření skladového systému, který již funguje, je nutné počítat s tím, že produkty už jsou nějakým způsobem rozmístěné. Pro tento případ zavedeme přesuny zboží. Uživateli skladu se zobrazí možné přesuny, které by ušetřily nejvíce času.

Priorita : Velmi vysoká
Náročnost : Vysoká

V tomto případě se jedná o stěžejní rozšíření systému a z toho důvodu je mu přiřazena velmi vysoká priorita. Zároveň už se jedná o poměrně komplexní rozšíření, které zabere více času.

1.5.1.6 FP6: Výměna zboží

Systém bude navrhovat vhodnou výměnu zboží.

Výměna zboží by měla fungovat na velice podobném principu jako klasický přesun, ale s tím rozdílem, že se provedou dvě přeskladnění. Tento typ přesunu se doporučí v případě, že máme špatně umístěné vytížené zboží a zároveň dobře umístěné, ale málo vytížené zboží. V této situaci dojde k jednoduché výměně produktů na daných umístěních.

Priorita : Střední
Náročnost : Střední

Situace, která by vyžadovala výměnu zboží, nenastane příliš často. Bylo by nutné, aby oba produkty odpovídaly rozměrem jejich umístění a zároveň, aby výměna dávala smysl z hlediska ušetřeného času. Po implementaci řešení předchozího požadavku již nebude tak náročné implementovat řešení i tohoto požadavku. Z toho důvodu je priorita i náročnost nastavena na střední.

1.5.1.7 FP7: Sloučení zboží

Systém bude navrhovat sloučení stejného zboží.

Z pozorování jsme zjistili, že ve skladu často nastává situace, při které je stejný produkt rozmístěn na různých umístění. Tento typ přesunů je určen právě pro tento případ, kdy zjistí, kde všude se produkt nachází a jestli by nebylo možné přesunout ho na jedno umístění nebo ho přeskupit alespoň blíže k sobě.

Priorita : Nízká
Náročnost : Střední

Stejně jako u FP6 tato situace nenastane příliš často a neušetří velké množství času. Produkty jsou většinou umístěny na různých umístění z důvodu, že se nevejdou pouze na jedno. Jednalo by se zde spíše o zjednodušení práce skladníků, při větším množství vyskladnění stejného produktu. Pro sklady, které již využívají tento systém, by ale toto rozšíření mohlo znamenat relativně velké množství přesunů. Z těchto důvodů má tento požadavek nízkou prioritu. Náročnost je stejná jako u FP6, jelikož se jedná o podobné rozšíření.

1.5.1.8 FP8: Seskupení zboží, které se prodává často společně

Systém bude navrhovat seskupení zboží, které se prodává často společně.

Pokud zjistíme, že se různé produkty často objevují společně ve stejných objednávkách, je pro časovou úsporu vhodné je umístit blíže k sobě.

Priorita : Střední
Náročnost : Velmi vysoká

Prioritu jsem určil jako střední a náročnost jako velmi vysokou. Jedná se sice o zajímavé rozšíření, které by do budoucna mohlo pomoci v konkrétních obdobích, ale v tuto chvíli je příliš komplexní. Bylo by nutné, aby na tento přesun systém korektně zareagoval a nežádal například ihned další přesun. To by mohlo nastat v případě, kdy po seskupení dvou produktů, které se prodávají společně, vznikl velký rozdíl mezi dostupností a umístěním některého z těchto produktů.

1.5.1.9 FP9: Návrhy při naskladnění

Systém bude navrhovat vhodné umístění produktů při naskladnění.

Proces naskladnění v tuto chvíli vytíženost zboží ani dostupnost zboží nijak nezohledňuje. Pro optimální umístění zboží je tento proces důležitý. Když

se správně umístí právě naskladněné zboží, nebude nutné pro něj v budoucnu vytvářet doporučení na přesun, jelikož bude umístěno dle jeho vytíženosti.

Priorita : Střední
Náročnost : Vysoká

Při implementaci bude nutné vzít v potaz obě dvě možnosti při naskladnění a doporučit vhodné umístění. Jedná se tedy o komplexnější problém, ve kterém budeme muset dbát i na rychlost souvisejících procesů.

1.5.1.10 FP10: Doplnění kusového zboží

Systém bude upozorňovat na nedostatek kusového zboží.

Tento požadavek, který vzešel přímo od vedoucího skladu, je přidat do systému upozornění na docházející kusové zboží. V tuto chvíli musí pracovníci skladu každé ráno obejít všechna umístění a zjistit, na kterých se nachází malý počet kusového zboží, které je potřeba doplnit z větších balení. Tato operace zabírá poměrně velké množství času a pro optimalizaci by bylo ideální, aby na to systém sám upozornil.

Priorita : Vysoká
Náročnost : Střední

Jedná se o požadavek, který znatelně ulehčí každodenní práci pracovníkům skladu, tudíž jsem mu přiřadil vysokou prioritu. Z hlediska implementace bude nutné přidat novou stránku, která tyto produkty bude zobrazovat, ale nemělo by se jednat o příliš komplexní problém. Náročnost je tedy střední.

1.5.2 Nefunkční požadavky

- **NP1: Součást systému** - Rozšíření bude implementováno jako součást stávajícího skladového systému Atlantis.
- **NP2: Frontend v jazyce Vue.js** - Frontend optimalizace bude implementován v jazyce Vue.js. Jelikož se jedná o rozšíření již používaného skladového systému, bude i ono rozšíření implementováno ve stejném jazyce, tudíž Vue.js.
- **NP3: Mobilní aplikace** - Není vyžadována speciální aplikace pro mobilní zařízení.
- **NP4: Responzivita** - Dodržíme stávající zobrazené systému a veškeré součásti rozšíření budou responzivní.

Kapitola 2

Návrh řešení jednotlivých požadavků

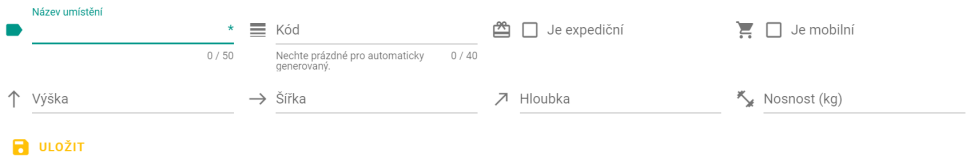
V této kapitole se zabývám navržením řešení jednotlivých požadavků, které jsou popsány v minulé kapitole. Část návrhů obrazovek uživatelského rozhraní vznikla v rámci předmětu BI-SP1 ve spolupráci s Michalem Kovářem a Janou Karafiátovou. Jejich návrhy následně vždy podléhaly debatě, které se účastnili všichni členové týmu a konzultovaly se i s vedoucím celého projektu. Výsledné návrhy větších obrazovek, které jsem částečně upravil, lze nalézt v příloze B.

Jelikož jsem v rámci prvního týmového projektu pracoval v celku, který se staral spíše o backend skladového systému Atlantis, umožnilo mi to účastnit se na návrhu jednotlivých algoritmů. To mi pomohlo k bližšímu pochopení jednotlivých procesů a díky tomu mohu některé části lépe popsat. Algoritmy, které jsou nutné k řešení požadavků, detailněji popisuje ve své práci Daniela Kováčová. Pro komplexní obraz problematiky zde popisuji dané algoritmy ve zjednodušené formě.

2.1 Rozměry umístění

Pro realizaci požadavku FP1 je nutné do systému přidat rozměry z důvodů vysvětlených v popisu požadavku. Návrh zahrnuje upravení následujících částí aktuální verze systému:

- **Vytvoření umístění:** Při vytváření nového umístění přidáme možnost definovat jednotlivé rozměry pomocí nových textových polí na obrazovce určené k tvorbě nového umístění. Tyto parametry přidáme jako nepovinnou volbu, ale pokud nebudou vyplněny, nebude se dané místo zohledňovat pro optimalizaci. Návrh této možnosti je možné vidět na obrázku 2.1.
- **Editace stávajícího umístění:** V editaci bude možné upravit jednotlivé rozměry pomocí textových polí, do kterých se vepíše hodnota rozměru stejně jako při vytváření nového umístění.
- **Zobrazení informací o umístění:** Informace o rozměrech se nově zobrazí v detailu každého umístění a bude také přidaná do tabulky s přehledem jednotlivých umístění daného skladu. Návrh lze vidět na obrázku B.1.



Obrázek 2.1: Návrh tvorby nového umístění

2.2 Rozměry produktů

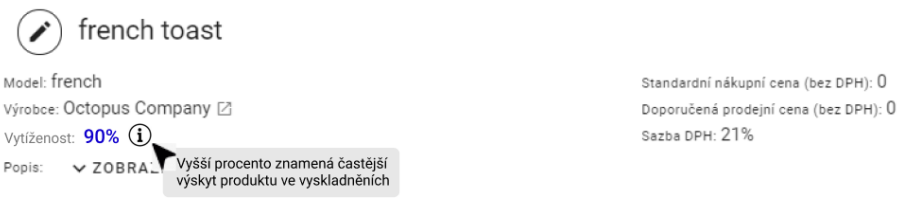
Rozměry produktů budou uchovávány u jednotlivých čárových kódů z důvodů zmiňovaných u FP2. Jednotlivé úpravy budou totožné s úpravami pro rozměry umístění, které popisuji v předešlé sekci. Jediný rozdíl nastane u zobrazení. V detailu produktu se v aktuální verzi zobrazuje seznam jednotlivých čárových kódů. V tomto seznamu nově zobrazíme rozměry, které se vážou k jednotlivým produktovým čárovým kódům. Návrh lze vidět na obrázku B.2.

2.3 Vytíženost produktů

Pro realizaci požadavku FP3 přidáme do systému informaci o vytíženosti produktů. Hlavním prvkem pro výpočet této vytíženosti je počet vyskladnění daného produktu za poslední měsíc. Tato hodnota se bude přepočítávat každý den a bude tak poměrně rychle reagovat na změny. Základní myšlenka algoritmu je, že se každý den zjistí z databáze, které produkty se vyskladňovaly za poslední měsíc. Nejčastěji vyskladňovanému produktu se přiřadí hodnota 100 % a všem ostatním se nastaví procenta dle poměru k tomuto nejčastěji vyskladňovanému. Původní myšlenka byla, že se zobrazí pomocí 3 štítků, ale po konzultaci s vedoucím práce, jsme se rozhodli pro zobrazení této hodnoty pomocí procent. Čím vyšší procento, tím častěji se s daným zbožím pracuje.

Uživatel bude mít možnost s touto hodnotou pracovat. Pokud bude chtít může vytíženost ručně upravit na jím zvolenou hodnotu a tím i zakázat každodenní přepočítávání pro daný produkt. Pro tuto variantu jsme se rozhodli z důvodu možnosti náhlé změny v počtu vyskladnění některého zboží. Uživatel může mít informaci, že v následujících dnech se s některým výrobkem nebude vůbec pracovat, přestože v posledním měsíci byl vyskladňován velice často a může tak ručně zamezit přepočítávání, které by na změnu zareagovalo až po několika dnech. Variantou by bylo manipulovat s počty vyskladnění v kratším časovém horizontu, než je jeden měsíc, ale z konzultace s vedoucím práce vyplynulo, že dlouhodobější data jsou více relevantní. Tato možnost bude dostupná na stránce pro úpravu produktu, kde se přidá možnost uzamknout produkt pro přepočítávání, čímž se zpřístupní textové pole, do kterého bude možné napsat vlastní hodnotu vytíženosti daného produktu. Tento produkt se následně nebude zohledňovat v hromadném přepočítávání i kdyby měl hodnotu 100 %.

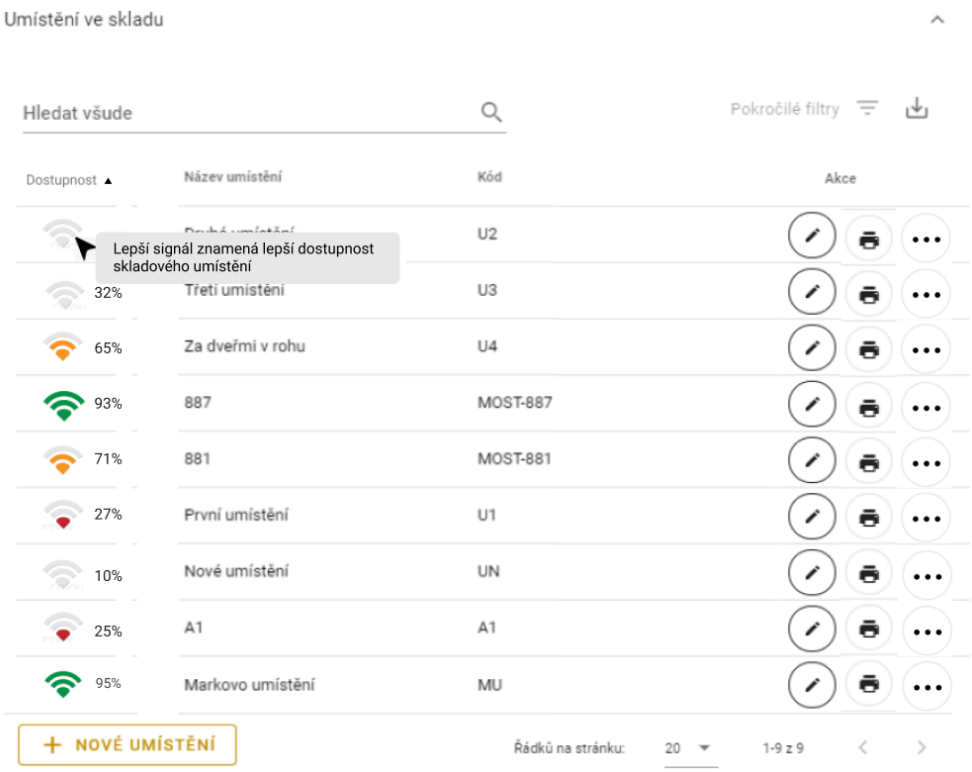
Hodnota vytíženosti se zobrazí v detailu produktu. U této hodnoty se bude nacházet informační ikonka, na kterou když uživatel najede myší, zobrazí stručnou definici vytíženosti. Kromě detailu produktu nalezneme vytíženost i v seznamu produktů. Pro lepší přehlednost se bude vyšší hodnota vytíženosti zobrazovat tučnějším písmem, jak lze vidět na obrázku 2.2.



Obrázek 2.2: Návrh zobrazení vytíženosti

2.4 Dostupnost umístění

Do systému přidáme informaci o dostupnosti jednotlivých umístění. Tato informace bude vycházet ze vzdálenosti daného umístění od důležitých míst definovaných v požadavku FP4. Jednotlivé vzdálenosti nám bude schopen poskytnout spolupracující tým pod vedením Ing. Jiřího Hunky, který má za úkol zpracovat mapu skladu. Vzdálenosti se následně zprůměrují a tato hodnota se porovná s hodnotou ostatních umístění. Následně se podobně jako u vytíženosti nastaví umístění s nejnižší průměrnou vzdáleností hodnota 100 % a všem ostatním se nastaví procenta dle poměru k nejdostupnějšímu umístění.



Obrázek 2.3: Návrh zobrazení dostupnosti

Jelikož by mohl panovat velký nepoměr mezi umístěním, které je sice blízko, ale vysoko a pro přístup k němu se musí použít vysokozdvižný vozík, rozhodli jsme se, že umístění, která nejsou v ručním dosahu označíme jako *vysoká* a jejich dostupnost určíme podle jejich nižších pater. K tomuto kroku jsme se uchýlili z toho důvodu, že ve vysokých umístění jsou nejčastěji velké palety se zbožím, které se většinou nevyskladňují, ale měly by v ideálním případě obsahovat velké množství kusového zboží stejného druhu, které je uskladněné v nižších patrech.

Při výpočtu dostupnosti si nejprve zjistíme, v jaké výšce se nachází dané umístění, což nám bude schopna poskytnout funkcionalita aktuálně vyvíjené mapy skladu a pokud bude umístění výše než 180 cm budeme ho považovat za vysoké a přiřadíme mu dostupnost dle jeho nižších pater.

Stejně jako u vytíženosti bude mít uživatel možnost tuto hodnotu ručně upravit. Na rozdíl od vytíženosti se tato hodnota nebude automaticky měnit, ale k umístění se přiřadí pouze při přidání tohoto rozšíření do systému, případně po přidání nového umístění. Hodnota se zobrazí v detailu umístění a v seznamu jednotlivých umístění daného skladu. Pro lepší přehlednost bude u hodnoty i ikona signálu, kde při vyšší dostupnosti ukáže vyšší signál. Po najetí myší na ikonu se zobrazí definice dostupnosti, jak lze vidět na obrázku 2.3.

2.5 Přesuny zboží

Pro korektní umístění jednotlivých produktů a realizaci požadavku FP5 je nutné vytvořit návrhy na přesun již umístěného zboží. Vedoucímu zobrazíme přesuny, které mu ušetří nejvíce času a když některý z nich potvrdí, vytvoří se nový úkol na přeskladnění. V rámci uživatelského rozhraní se přidá nová položka v menu s názvem *Přesuny zboží*. Po jejím rozkliknutí se zobrazí seznam jednotlivých návrhů na přesun seřazených podle ušetřeného času. V jednotlivých řádcích seznamu budou uvedené i základní informace o přesunu. Na každý přesun se bude dát kliknout a pokud uživatel tak učiní, ukáže se detail daného přesunu. V této fázi bude mít uživatel možnost přesun potvrdit nebo zamítnout. Při potvrzení se zobrazí okno, které uživatele upozorní na skutečnost, že daný přesun ovlivní ostatní zobrazené přesuny a některé z nich již nebude možné potvrdit. Tyto přesuny mohou být různého druhu a každý z nich popisuji v samostatné části.

2.5.1 Přesun zboží

V tomto případě se jedná o klasický přesun jednoho typu produktu na nové umístění. Je to nejjednodušší případ, při kterém se provede pouze jeden přesun. Důležité je zajistit, aby se zboží na nové umístění vešlo bez nutnosti dodatečných přeskladnění. Zajímat nás budou hlavně přesuny vytížených skladových položek, které leží na umístění s nízkou dostupností. V detailu přesunu se zobrazí aktuální informace o produktu včetně jeho umístění a informace o novém umístění. Tento návrh souvisí s požadavkem FP5. Návrh lze vidět na obrázku B.3.

2.5.2 Výměna zboží

Výměna zboží navrhne výměny všech kusů určité skladové položky na daném umístění za jiný produkt. Jednat se bude především o ty s vysokou vytížeností, které se nachází na umístění s nízkou dostupností a není pro ně možný klasický přesun. Tento návrh souvisí s požadavkem FP6. Návrh lze vidět na obrázku B.4.

2.5.3 Sloučení zboží

Při návrhu na sloučení se zobrazí zboží, které je umístěné na více místech. Preferované bude především to s vyšší vytížeností. Tento návrh souvisí s požadavkem FP7. Návrh lze vidět na obrázku B.5.

2.5.4 Seskupení zboží, které se prodává často společně

Návrh na řešení požadavku FP8 je takový, že na samostatné záložce v menu pod návrhy na přesun bude možné zobrazit návrhy na seskupení zboží, které se prodává často společně. Tyto informace bychom měli být schopni získat od spolupracujícího týmu, který se zabývá predikcí objednávek v rámci skladového systému Atlantis. Znovu bude mít uživatel možnost návrh zamítnout nebo potvrdit a vytvořit tak úkol k přeskladnění jednoho z produktů. Hlavní myšlenka algoritmu je taková, že se pro dané produkty prohledávají nejbližší umístění, kam by se ten druhý mohl uložit bez nutnosti dalších přesunů.

2.6 Proces naskladnění

Řešením požadavku FP9 je vhodná optimalizace procesu naskladnění, při kterém se rovnou ukládají naskladňované produkty na vhodná umístění dle jejich vytíženosti. Jak je popsáno v analýze, tak v tuto chvíli existují dvě možnosti, jakými lze provést naskladnění.

V té první, kdy skladník naskenuje nejprve všechny položky, které chce naskladnit a následně je umisťuje, je optimalizace relativně jednoduchá. Kromě míst, kam je může naskladnit, se mu nově zobrazí pro každý produkt ideální místa, kam by měl zboží uložit, aby to odpovídalo jeho vytíženosti. Toto umístění ale nechceme vynucovat, tudíž zboží bude moct být uloženo na jakékoliv volné místo. Pokud by se však jednalo o výrazný rozdíl vytíženosti a dostupnosti, bude na to skladník upozorněn.

V případě druhém, tedy když si skladník naskenuje nejprve místo a následně na něj umisťuje jednotlivé produkty, jsme se rozhodli pro upozornění, pokud se bude jednat o rozdíl mezi dostupností naskenovaného umístění a vytížeností zboží, stejně jako v předchozím případě a až následném doporučení vhodného umístění. Pokud tedy skladník naskenuje umístění s dostupností 90 % a poté

naskenuje produkt s vytížeností 30 %, bude upozorněn, že se jedná o velký rozdíl a že pro tento produkt by bylo vhodnější jiné umístění.

Možností jak zobrazit navrhované umístění je více. V aktuální verzi systému se při naskladnění zobrazuje informace, na která umístění lze naskenovaný produkt uložit. Variantou by tedy bylo upravit tuto implementaci a zobrazit jednotlivá umístění seřazené podle toho, na která z nich je vhodné produkt umístit. Druhou variantou by bylo vytvořit novou implementaci této funkcionality a zobrazit vhodné umístění pro naskladňovaný produkt zvlášť.

2.7 Doplnění kusového zboží

Poslední z požadavků, tedy FP10, se týká upozornění na nedostatek kusového zboží. Tato upozornění se budou nacházet v menu v záložce nazvané *Doplnění kusovek*. Algoritmus bude vycházet z průměrného denního počtu vyskladněných kusů daného zboží a pokud zjistí, že se ve spodních umístění, tedy těch, která nejsou označena jako *Vysoká*, nachází menší počet, zobrazí ho v seznamu. Seznam zobrazí jednotlivá doplnění a uživatel bude mít možnost vybrat ty, pro které chce vytvořit úkol na doplnění nebo je vytisknout.

Kapitola 3

Realizace

V této části práce se zaměřuji především na průběh implementace návrhů definovaných v předchozí kapitole. Nejprve popisuji technologie, které jsem využil při implementaci a následně objasňuji, jak samotný proces probíhal.

3.1 Použité technologie

Jelikož se jedná o již používaný systém, bylo předem dáno, které technologie se mají použít pro toto rozšíření. Backend skladového systému Atlantis, který vytvořil v rámci své diplomové práce Ing. Pavel Kovář [2], je napsán v programovacím jazyce PHP. Oproti tomu frontend, který vytvořil v rámci své diplomové práce Ing. Oldřich Malec [1], je napsán pomocí JavaScriptového frameworku Vue.js. Implementace frontendu jednotlivých návrhů na optimalizaci tedy probíhala taktéž v tomto jazyce. V této sekci se zmiňuji o různých technologiích, které jsem při implementaci využil.

3.1.1 JavaScript

JavaScript je programovací jazyk navržený pro skriptování v různých webových prohlížečích [13]. Vznikl v roce 1995 a v dnešní době patří k nejpoužívanějším programovacím jazykům, které se používají především k implementaci v rámci HTML dokumentů. Díky jeho popularitě vznikl na jeho základě velký počet různých frameworků, což jsou kolekce JavaScriptových knihoven, které poskytují jejich uživatelům předepsaný kód k běžným programovacím úkolům. Při tvorbě aplikace, tak není nutné pokaždé začínat od samého počátku, ale lze využít základy, které poskytují právě tyto frameworky [14]. Mezi nejznámější frameworky patří v dnešní době především Angular, React a Vue.

3.1.2 Vue.js

Vue.js, je JavaScriptový framework, který slouží k tvorbě uživatelského rozhraní [15]. Staví především na základech HTML, CSS a JavaScriptu. V roce 2014 ho vytvořil Evan You a mezi jeho hlavní výhody patří jednoduchost na pochopení a naučení a snadno lze využít pro vývoj nové aplikace. Při implementaci se s ním pracovalo velice dobře i bez větších předchozích zkušeností.

3.1.3 Mock Service Worker

Jak je zmíněno v analýze, frontend a backend skladového systému Atlantis spolu komunikují pomocí REST API. V průběhu implementace se ale vyskytly problémy v synchronizaci se souběžným vývojem backendové části rozšíření, které blíže představuji v kapitole 3.2.1. Z toho důvodu jsem musel přistoupit k simulaci komunikace frontendové a backendové části za pomocí takzvaného mocku. Jedná se o způsob, pomocí kterého se nasimulují odpovědi serveru bez provedení konkrétních výpočetních operací či komunikace s databází. Pro tyto účely jsem zvolil knihovnu Mock Service Worker [16]. Jedná se o knihovnu, která zachytí požadavek a odpoví na něj dle konfigurace provedené vývojářem. Pro zachycení požadavku je nutné definovat jeho URL a následně definovat data, která má vrátit. Jak tato definice vypadá je možné vidět v ukázce kódu 3.1. V něm si můžeme povšimnout, že má zachytit požadavek GET s URL `'https://api.atlantis.jagu.cz/productMovementSuggestions/3'`. Data s informacemi o produktu, který je součástí návrhu na přesun vrací ve formátu JSON.

```
rest.get(
    'https://api.atlantis.jagu.cz/productMovementSuggestions/3',
    (req, res, ctx) => {
        return res(ctx.json({
            id: 3,
            type:ProductMovementSuggestionType.SWAP,
            product: {
                id: 30,
                amount: 5,
                turnover: 88,
                name: "Toastovač",
                code: 123123123
            }));
})
```

Obrázek 3.1: Definice zachycení požadavku pomocí Mock Service Worker

3.1.4 PhpStorm

Jako vývojové prostředí jsem zvolil program PhpStorm, který byl vytvořen společností JetBrains s.r.o. Jedná se o IDE, které podporuje vývoj v několika různých programovacích jazycích a usnadňuje celý proces implementace. V mém případě jsem ho využil především pro snadné vyhledávání v kódu napříč celým projektem a opravu syntaktických chyb, na které samo upozorňuje [17]. Další možností bylo využít program WebStrom od stejné společnosti. Ten je ale určen především pro programování v jazyce JavaScript a jelikož jsem si několikrát potřeboval prohlédnout implementaci backendové části skladového systému Atlantis, která je psaná v jazyce PHP, nebylo pro mě toto IDE optimální.

3.2 Implementace návrhů

V této sekci popisuji, jak probíhala realizace jednotlivých návrhů zmíněných v minulé kapitole. V první řadě se zaměřuji na obecný průběh implementace a následně konkrétně popisuji jednotlivé části.

3.2.1 Výskyt a řešení problémů

Důležité je zmínit, že skladový systém Atlantis je neustále vyvíjející se program, který se velice často upravuje a rozšiřuje. Při implementaci navržených optimalizací jsme se často setkávali s tím, že určitá část systému, kterou jsme upravili v rámci našeho rozšíření, byla již změněna či upravena hlavními vývojáři celého systému. Naštěstí jsme vždy tyto problémy vyřešili komunikací s vedoucím práce a projektovým manažerem skladového systému Atlantis a za pomocí verzovacího nástroje Git, jsme naše řešení upravili tak, aby korespondovalo s provedenými změnami.

Dalším problémem, který se vyskytl, byla časová synchronizace s Danielou Kováčovou, která současně vyvíjela backend navrhovaných rozšíření. Původní plán byl takový, že jednotlivé návrhy budou implementovány ve stejnou chvíli, jak na frontendu, tak na backendu a po dokončení každé části se implementace spojí a otestují, jestli nenastávají nějaké problémy v komunikaci. Vývoj backendu byl ale náročnější, než se původně předpokládalo, a proto jsme přistoupili k řešení, které popisuji v kapitole 1.1, tedy definování API požadavků. Pro tyto požadavky jsem následně připravil mock, jehož princip popisuji v kapitole 3.1.4. Nevýhodou tohoto řešení je, že výsledek mé práce nebude moct být ihned nasazen a použit, ale bude muset čelit následným úpravám, aby zobrazoval reálná data a korektně komunikoval s backendem skladového systému Atlantis. Změny, které jsou nutné pro funkčnost celého rozšíření popisuji v dokumentaci, která se nachází v příloze této práce.

3.2.2 Rozměry produktů

Jak zmiňuji v analýze, část implementace probíhala ještě v rámci předmětů BI-SP1 a BI-SP2. Většina byla však zaměřená především na implementaci algoritmů a procesů na backendu a na frontendu se implementovali pouze základy určitých návrhů. Jedním z nich byly rozměry produktů, respektive jejich čárových kódu. Realizace tohoto návrhu spočívala především v přidání atributů výška, šířka, hloubka a hmotnost k čárovým kódům. Základ této implementace připravil Jakub Volák, který přidal tyto atributy do stránky pro editaci čárových kódů a do tabulky, která zobrazuje informace o čárových kódech. Zároveň také zanesl tyto atributy do souboru s překlady do českého jazyka. Tuto implementaci jsem zkontroloval a dále s ní pracoval. Při konzultaci s vedoucím práce bylo zjištěno, že při zobrazení rozměrů v tabulce s informacemi o čárových kódech, se tabulka stane příliš širokou a nezobrazí se celá, což je pro uživatele důležité. Tabulku jsem tedy upravil tak, aby zobrazovala pouze důležité informace, které ji příliš neroztáhnou a přidal jsem k ní zaškrtávací pole, které při zaškrtnutí zobrazí v tabulce i informace o rozměrech. Rozdíl lze vidět na obrázcích C.2 a C.3.

Dalším problémem, na který jsme spolu s vedoucím práce a projektovým manažerem celého projektu přišli, je, že informace o rozměrech přiřazených k čárovému kódu produktu se nedají získat při skenování skladových položek u naskladnění. Rozhodli jsme se tedy kromě těchto rozměrů uchovávat i základní rozměr jednoho kusu dané položky, který se bude vázat přímo ke konkrétnímu produktu. Při naskenování se totiž dá zjistit počet kusů dané položky v naskenovaném balení a při korektním přidání odchylky, která může vzniknout různou formou balení, bude možné zjistit, kam by se dané balení mělo umístit. Informaci o rozměrech produktu jsem nechal zobrazit na stránce s detaily produktu a také na stránku, kde se produkty vytvářejí či upravují. V tabulce se všemi produkty tyto informace pro přehlednost nezobrazuji.

Po přidání na stránku pro tvorbu nových či úpravu stávajících produktů se ale stránka stala, kvůli velkému množství textových polí nepřehlednou. Po konzultaci s vedoucím mé práce, jsem upravil vzhled tak, aby nejprve ukazoval důležité informace a zároveň přidal rozbalovací panel, který po kliknutí zobrazí další detaily, které se dají upravit. Rozdíl lze vidět na obrázcích C.4 a C.5.

Hodnoty, které se aktuálně zobrazují v systému jsou nastavené přímo v kódu. V tomto případě se totiž nebude volat nový požadavek, ale pouze se přidají data ke stávajícímu, který po zavolání s argumentem `productId` vrátí informace o požadovaném produktu. Pro účely testovaní je tedy lepší nevytvářet mock pro tento požadavek, ale pouze mu nastavit hodnoty u atributů, které se neposlaly. Pro následné nasazení a propojení s funkčním backendem bude nutné tyto hodnoty smazat a zobrazit ty, které se pošlou v rámci odpovědi na požadavek.

3.2.3 Rozměry umístění

U rozměrů umístění byla implementace podobná jako u rozměru produktů akorát již neprobíhala v týmu. V tomto případě ale nebylo nutné přidávat rozbalovací panel, či schovávat podrobnosti, jelikož systém aktuálně neeviduje velké množství detailů ohledně umístění. Kromě samotných třech rozměrů a nosnosti, jsem k zobrazení seznamu umístění, detailu umístění a na stránku s editací stávajícího či tvorbě nového umístění přidal i informaci o tom, zdali se jedná o umístění nacházející se vysoko.

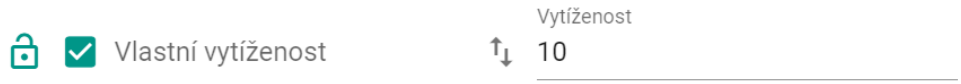
Na obrázku 3.2 je možné vidět jakým způsobem zobrazuji informace o rozměrech určitého umístění. Kód `class="text-caption"` nám určuje, jakým fontem bude zobrazen text. Aby v kódu nebyly české výrazy používají se soubory s překlady. Ty zavoláme pomocí `$t('stocks.locations.width')` a na stránce se zobrazí chtěný výraz v uživateli nastaveném jazyce. Hodnoty jsou stejně jako u rozměrů produktů nastaveny přímo v kódu a například pomocí proměnné `detailsDimensionsWidth`, která je níže v kódu nastavena na vybranou hodnotu.

```
<div>
    <span class="text-caption">
        {{ $t('stocks.locations.width') + ': ' }}
    </span>
    {{ detailsDimensionsWidth }}
</div>
<div>
    <span class="text-caption">
        {{ $t('stocks.locations.height') + ': ' }}
    </span>
    {{ detailsDimensionsHeight }}
</div>
<div>
    <span class="text-caption">
        {{ $t('stocks.locations.depth') + ': ' }}
    </span>
    {{ detailsDimensionsDepth }}
</div>
```

Obrázek 3.2: Zobrazení rozměrů pro skladové umístění

3.2.4 Vytíženost produktů

Implementaci vytíženosti produktů započal Michal Kovář, který ji nechal zobrazit na stránce s detaily produktu včetně stylu formátování, kdy vyšší dostupnost je zobrazena tučněji a tmavší barvou. Tuto implementaci jsem rozšířil o zobrazení nápovědy a přidal zobrazení hodnoty vytíženosti ve stejném formátu i do seznamu produktů. Následně jsem ke stránce pro tvorbu nových či úpravu stávajících produktů přidal checkbox *Vlastní dostupnost*, který při zaškrtnutí zpřístupní možnost editovat textové pole, ve kterém lze ručně upravit stávající vytíženost, jak lze vidět na obrázku 3.3. Poté jsem přidal do souboru s překlady české fráze, aby se vše korektně zobrazovalo.



Obrázek 3.3: Možnost úpravy vytíženosti v editaci produktu

Jelikož se znovu jedná o přidání dat u stávajícího požadavku, je hodnota vytíženosti nastavena přímo v kódu.

3.2.5 Dostupnost umístění

Implementaci dostupnosti umístění jsem realizoval samostatně a zahrnovala podobné části jako u vytíženosti zboží. Dostupnost se nyní zobrazuje v detailu daného umístění, ve výpisu jednotlivých míst určitého skladu a při tvorbě nového, nebo upravení stávajícího skladového umístění. Na poslední zmiňované stránce je přidán checkbox, který po zaškrtnutí zpřístupní možnost upravit hodnotu dostupnosti dle svého uvážení.

Oproti původnímu návrhu, kde dostupnost měla být zobrazena pomocí ikony signálu, jsem nakonec zvolil podobné řešení jako u vytíženosti, a to pomocí barvy, kdy podle hodnoty přechází z černé do zelené, a tloušťky písma, kterým se hodnota zobrazí. Tuto variantu jsem zvolil z toho důvodu, že v systému se používají Material Icons. Jedná se o ikony od společnosti Google, které jsou vytvořeny tak, aby byly dobře čitelné, daly se použít v různých velikostech a zároveň byly designově jednoduché [18]. V této kolekci se však nenacházely ikony signálu, které se použily ve vytvořeném návrhu a ani jim podobné a zároveň jsem nechtěl narušit stávající konvenci používání Material Icons.

Stejně jako u předchozích částí, jsou hodnoty nastaveny přímo v kódu.

3.2.6 Návrhy na přesun zboží

Pro návrhy na přesun jsem vytvořil zcela novou stránku, na kterou se lze dostat přes novou položku ve stávajícím panelu s menu, v části *Přehledy*. Data na této stránce se zobrazí po obdržení odpovědi na požadavek, který jsem definoval v novém souboru `ProductMovementSuggestionAPI.js`. Strukturu jednotlivých požadavků a jejich odpovědí lze nalézt na přiloženém médiu. Odpověď na toto volání jsem nasimuloval pomocí MockServiceWorker.

Tato stránka zobrazuje tabulku možných přesunů, seřazených podle času, který ušetří. Přesuny se vážou vždy k určitému skladu a podskladu. V seznamu lze nalézt jak klasické přesuny zboží z jednoho místa na druhé, tak výměnu zboží, tak návrhy na sloučení zboží, které se nalézá na více umístěních. V přehledu přesunů se zobrazují základní informace o tom, jakého produktu se týká, jak se zlepší jeho dostupnost a kolik ušetří času. Po kliknutí na libovolný řádek se zobrazí bližší informace o přesunu, přičemž pro každý typ se okno s detaily částečně liší.

Kliknutím se zároveň zavolá další požadavek, který si vyžádá, dle id přesunu, jeho bližší informace. Samotný design odpovídá vytvořenému návrhu s několika drobnými úpravami. Přidáno je také dialogové okno, které se objeví při kliknutí na tlačítko potvrzení přesunu a zobrazí varování o změně všech ostatních návrhů, pokud se tento potvrdí.

Zobrazené hodnoty jsou definované v souboru `handlers.js`, ve kterém je připraven mock dohodnutého API. Volání jednotlivých požadavků pro návrhy na přesun je odchyceno a jsou vrácena data, která jsem nadefinoval.

3.2.6.1 Doplnění kusového zboží

Pro informace o doplnění kusového zboží jsem zvolil formu tabulky, ve které se zobrazuje docházející kusové zboží. Konkrétně produkt, jeho umístění, běžný počet kusů a aktuální počet kusů. Uživatel si následně může vybrat, které položky doplnit a seznam vytisknout či zadat úkol k jejich doplnění. Tabulka se zobrazí vždy pro vybraný sklad a podsklad. Tabulku je možné vidět na obrázky C.1.

Hodnoty, které se zobrazují, jsou definované v souboru s připraveným mockem `handlers.js`.

Kapitola 4

Testování

V této kapitole se věnuji testovaní. Nejprve objasňuji, jaký typ testování jsem zvolil a zaměřuji se na jeho nejdůležitější součásti. Následně popisuji, jak samotné testování probíhalo a jaké z něho vzešly poznatky.

4.1 Uživatelské testování

K otestování výsledků implementace jsem se rozhodl využít uživatelské testování, které jsem vyhodnotil jako nejlepší formu k nalezení případných nedostatků ve frontendové části rozšíření. Základní myšlenkou tohoto testování je nechat uživatele plnit různé úkoly, které zahrnují použití nových funkcí systému [19]. Během této činnosti zadavatel pozoruje, jakým způsobem uživatel plní úkoly. Hlavním cílem je tak odhalit, které činnosti trvaly uživateli dlouho, či se kterými úkoly měl nějaký problém. Při uživatelském testování se běžně postupuje v následujících krocích [20]:

- Definice cílů testování
- Výběr vhodných testovacích uživatelů
- Výběr vhodných úkolů, které uživatel bude muset splnit
- Zahrnutí jednotlivých úkolů do scénářů, které jsou pro uživatele snazší pro pochopení.
- Výběr způsobu, kterým se budou zaznamenávat získané informace
- Příprava testovacího prostředí
- Domluva s konkrétním testovacím uživatelem o formě testování
- Provedení samotného testování
- Analýza poznatků a vyvedení důsledků

4.2 Scénáře

Pro účely uživatelského testování jsem vytvořil pět scénářů, pomocí kterých docílím otestování implementovaných funkcionalit.

4.2.1 TS1 - Nové umístění

Ve skladu Prodejna 'Na růžku' vzniklo nové umístění s názvem U08 a kódem N23. Toto umístění je vysoké 50 cm, široké 100 cm, hluboké 65 cm a lze na něj položit až 150 kg. Nejedná se o mobilní ani balící místo, ale je vysoko. Vytvořte toto umístění v systému dle uvedených informací a dostupnost nechte automaticky vygenerovanou systémem.

Pomocí tohoto scénáře ověřím, zdali jsou prvky, které jsem v rámci implementace přidal na stránku s tvorbou nového umístění, čitelné a zobrazené přehledně.

4.2.2 TS2 - Úprava produktu

Víte, že skladová položka Preclík bude v následujících dnech vyskladňována velice často. Zjistěte, jakou má aktuálně nastavenou vytíženost a případně ji nastavte na 80 %. Následně se podívejte, jestli u stejného produktu existuje čárový kód, který představuje produkt s rozměry výška: 10 cm, šířka 20 cm délka 30 cm a hmotnost 1 kg.

Za pomoci tohoto scénáře zjistím, jestli se nově zobrazená vytíženost dá snadno vyhledat a manuálně upravit. Zároveň také zjistím to, jestli forma zobrazení detailů čárových kódů pomocí dodatečného zaškrtávacího pole je vhodná.

4.2.3 TS3 - Návrh na přesun

Máte pocit, že rozmístění produktů ve skladu Prodejna 'Na růžku' není ideální. Prohlédněte si, jestli Vám systém nenabízí návrhy, jak optimálně přesunout zboží. Pokud ano, vyberte ten, který Vám dle systému ušetří nejvíce času, zjistěte, jakého produktu se týká, na kterém umístění aktuálně je a na které by mělo být umístěno a přesun potvrďte.

V tomto případě se jedná o jeden z nejdůležitějších testovacích scénářů. Oproti těm předchozím totiž ověří přehlednost zobrazení nově přidané stránky, kterou uživatelé před tím neviděli. TS1 i TS2 oba testují pouze nově přidané části na stránkách, které uživatelé již dříve využívali a snadno si tak všimnou rozdílu oproti předchozím verzím. V tomto případě se bude jednat pro stávající uživatele o něco zcela nového a pomocí tohoto scénáře tak otestuji, jak rychle

se uživatelé na stránce s návrhy na přesun zorientují a zvládnou na ní vyhledat důležité informace týkající se přesunů.

4.2.4 TS4 - Návrh na sloučení

Víte, že ve skladu Prodejna 'Na růžku' je produkt Kobliha umístěn na dvou různých umístění. Zjistěte, jestli Vám systém nenavrhuje jeho sloučení a pokud ano, zjistěte, na které umístění bude sloučení provedeno a přesun potvrďte.

Jelikož jsou návrhy na přesun důležitou a komplexní součástí implementace, rozhodl jsem se jim věnovat dva scénáře. Tento scénář je tedy podobný jako ten předchozí a má i podobný úkol. Oproti tomu předchozímu však zobrazí trochu jinou stránku s detailem přesunu a může tak odhalit dodatečné nedokonalosti.

4.2.5 TS5 - Doplnění kusového zboží

Víte, že je potřeba doplnit určité kusové zboží umístěné ve skladu Prodejna 'Na růžku', aby bylo připraveno pro případné vyskladnění. Podívejte se v systému, o které konkrétní zboží se jedná a vyberte, které chcete, aby se doplnilo. Následně tento seznam vytiskněte nebo zadejte jako úkol pro doplnění.

Poslední scénář má za úkol odhalit nedokonalosti na také nově přidané stránce s návrhy na doplnění kusového zboží. Oproti návrhům na přesun je však tato stránka jednodušší a není tak nutné ji věnovat více než jeden testovací scénář.

4.3 Průběh a vyhodnocení testování

V této sekci popisuji, jak probíhalo samotné testování, s kým bylo realizováno a jaké přineslo poznatky pro budoucí vývoj celého rozšíření.

4.3.1 Průběh

Uživatelem, který testoval implementaci nových rozšíření, byl Libor Kudrna, který využívá skladový systém Atlantis pro různé účely jak jako vedoucí, tak jako skladník. Výhodou tedy je, že zná dobře různé části systému a je vhodným kandidátem pro tento typ testování.

Testování probíhalo dálkově. Nejprve jsem uživatele seznámil se základními informacemi ohledně našich rozšíření a následně začal pracovat na úkolech definovaných v jednotlivých scénářích. Po dokončení každého scénáře následovala krátká diskuze nad nově přidanými prvky, které byl uživatel nucen využít pro splnění jednotlivých úkolů, jejich přehledností a přínosem pro uživatele.

Po splnění všech scénářů jsme ještě jednou prošli celkový přínos rozšíření a ověřili korektnost zjištěných poznatků.

4.3.2 Vyhodnocení

4.3.2.1 Vyhodnocení TS1

Při provádění prvního testovacího scénáře uživatel nenarazil na žádný zásadní problém. Všechny úkoly provedl bez problému. Jen při tvorbě nového místa zapomněl zakliknout pole, které značí, že se jedná o vysoké umístění. Jednalo se však o nepozornost a uživatel neměl k novým změnám, které v rámci tohoto scénáře objevil, žádné připomínky.

4.3.2.2 Vyhodnocení TS2

Druhý testovací scénář taktéž nebyl pro uživatele velkou překážkou. Jediný problém nastal při ruční úpravě vytíženosti. Uživatel nemohl přijít na způsob, jakým zpřístupnit editaci textového pole, ve kterém lze hodnota upravit. Po následné diskuzi jsme došli k závěru, že by bylo vhodné zaškrtávací pole, které zpřístupňuje editaci vytíženosti zobrazit až za textovým polem s hodnotou vytíženosti nikoliv před ním, jak bylo zobrazeno. Zároveň jsme diskutovali nad rozbalovacím panelem, který schovává dodatečné možnosti editace a který zmiňuji v kapitole 3.2.2. Uživatel zastával názor, že editačních polí není zas takové množství, aby to muselo být skryto, ale že rozbalovací panel nepředstavuje žádný zásadní problém. Nadnesl také myšlenku, že by bylo vhodné přidat do celého systému možnost nastavit pole, která se zobrazí dle individuálního nastavení každého uživatele.

4.3.2.3 Vyhodnocení TS3

Nad plněním úkolů tohoto testovacího scénáře strávil uživatel větší množství času než na předchozích. Hlavním důvodem bylo především to, že se jednalo o zcela novou stránku. Tabulku se seznamem možných přesunů ocenil to, že obsahuje všechny důležité informace a neměnil by její strukturu. Delší debata pak vznikla nad detailním zobrazením návrhu na přesun. Jedním z bodů diskuze byla ikona uprostřed detailu, která v uživateli evokovala tendenci na ni kliknout a přesun tím potvrdit. Dále by ocenil, kdyby v detailu byla možnost dostat se na stránku s detailem produktu či umístění, kterých se přesun týká, aby si zjistil případné další podrobnosti. Úkoly ve scénáři však uživatel bez problému splnil, a kromě výše zmiňovaných poznámek žádné připomínky ke zpracování neměl.

4.3.2.4 Vyhodnocení TS4

I přesto, že tento čtvrtý testovací scénář je podobný jako ten předtím, jeho plnění vyvolalo větší diskuzi. Hlavním předmětem debaty byl samotný návrh

na sloučení a princip, jakým způsobem by měl fungovat. V aktuální verzi je totiž implementováno pouze pro jeden produkt na dvou různých umístění. Dle uživatelových zkušeností je však běžné, že ve větších skladech je jeden produkt rozmístěn na více než dvou umístění a bylo by tedy vhodné doporučit sloučení rovnou ze všech nebo alespoň z více různých míst najednou. Následně jsme řešili, jakým způsobem by tento návrh na sloučení mohl vypadat a jak docílit co nejefektivnějšího zobrazení těchto informací. Samotné provedení detailu pro sloučení z dvou různých umístění přišlo uživateli v pořádku scénář znovu bez problému splnil.

4.3.2.5 Vyhodnocení TS5

Uživatel splnil i všechny úkoly v rámci posledního scénáře, po kterém se ale pozastavil nad celkovou myšlenkou doplnění kusového zboží. Dle jeho zkušeností se ve skladu nejčastěji pracuje s celými paletami zboží a jednotlivá umístění jsou často vytvořená tak, aby se na ně vešla právě jedna paleta. Dle jeho názoru je tak praktičtější vždy vyčkat, dokud kusové zboží nedojde úplně, i kdyby k tomu mělo dojít až během dne a následně doplnit o celou paletu. Varianta, kdy skladník rozdělá paletu, aby doplnil několik kusů uživateli nepřišla jako ideální a reálně proveditelná. Po delší debatě na toto téma jsme dospěli k názoru, že tento problém by měl podlehnout v budoucnu větší analýze a zjistit detailně jakým způsobem toto doplnění aktuálně funguje v různých skladech a jak to zrealizovat co nejoptimálněji. Dle jeho zkušeností tak proces doplnění kusového zboží funguje jiným způsobem, než nám bylo řečeno při návštěvě skladu.

4.4 Úpravy po testování

Po provedení testování jsem učinil drobné změny v implementaci. Tyto změny vycházely především z poznatků učiněných při provádění TS2 a TS3. V editaci produktu jsem prohodil textové pole vytíženosti se zaškrtávacím polem, které jej zpřístupňovalo. Na stránce s návrhy na přesun jsem umožnil potvrzení přesunu pomocí tlačítka uprostřed detailu a přidal tlačítka, která po stisknutí otevírají v novém okně stránku s detailem produktu či umístění. Úpravu je možné vidět na obrázku C.6.

Plnění úkolů definovaných v TS1 žádné nedostatky neobjevilo. Výsledky TS4 bude nutné dále zanalyzovat a rozhodnout jakým způsobem by mohlo být realizováno sloučení produktů z více než dvou umístění najednou. Podobný závěr přinesly i výsledky TS5. U doplnění kusového zboží bude nutná další analýza, která určí nutnost tohoto rozšíření případně i formu. V aktuální verzi jsem ponechal možnost tabulku s návrhy na doplnění kusového zboží vytisknout, a tím splnit požadavky vedoucího navštíveného skladu, který požadoval upozornění na docházející kusové zboží. Zbytek implementace byl ponechán ve stejné formě jako před testováním.

Kapitola 5

Možnosti navázání

Na výsledky mé práce se dá navázat více způsoby. Tím prvním by mělo být propojení implementace frontendu s backendovou částí rozšíření a umožnit nasazení celého rozšíření, které zrychlí časté skladové procesy. Toto propojení vyžaduje funkční backend, implementován dle dohodnutého API a drobné úpravy frontendové části, které popisuji v souboru na přiloženém CD.

Další možností, jak na tuto práci navázat je realizace návrhů, které v rámci této práce nebyly implementovány. Konkrétně se jedná o doporučení vhodného umístění zboží již při jeho naskladnění. Způsob, jakým by tento proces mohl fungovat, je popsán v sekci s návrhy řešení jednotlivých požadavků. Dalším nerealizovaným návrhem je umisťování zboží, které se prodává často společně. K realizaci tohoto návrhu by bylo možné použít predikce pro skladový systém Atlantis, kterými se ve své bakalářské práci zabývá Jan Štipl.

V průběhu testování se přišlo na možné doplnění stávající implementace. Konkrétně se jedná o rozšíření návrhu na sloučení zboží, který je v aktuální verzi realizován pouze pro dvě různá umístění. Tato implementace by se dala rozšířit pomocí přidání možnosti sloučit jeden druh zboží z více různých umístění najednou. Částí, ve které by se také dalo pokračovat je doplnění kusového zboží. V tomto případě by bylo vhodné detailněji analyzovat způsob, jakým aktuálně probíhá doplnění kusového zboží a současnou implementaci upravit či rozšířit například o automatické zadávání úkolů při poklesu kusového zboží na určitém umístění.

Závěr

Hlavním cílem této práce bylo navrhnout a implementovat vhodná rozšíření frontendové části skladového systému Atlantis, která by časově optimalizovala časté skladové procesy. Hlavní důraz byl kladen na dosažení vhodného umístění zboží.

V práci jsem nejprve popsal, jakým způsobem jsme v rámci softwarového týmu prováděli analýzu skladového systému a jaké požadavky jsme určili jako vhodné k vypracování a k řešení. Tyto požadavky se týkaly především rozšíření informací, které se evidují o produktech a skladových umístění a o způsobu jakým docílit vhodného umístění skladových položek. Jedním z výsledků je tedy souhrn těchto poznatků vycházejících z analýzy, které je možné použít pro případnou budoucí optimalizaci.

Následně jsem se zabýval návrhem řešení jednotlivých požadavků, jehož součástí byla tvorba obrazovek, které zobrazují chtěná rozšíření a která odpovídají současné verzi skladového systému Atlantis. Některé návrhy v rámci této práce nebyly realizovány a naskýtá se zde možnost, jak na mou práci navázat právě implementací některého z těchto návrhů.

V rámci realizace se mi podařilo implementovat jednotlivá dílčí rozšíření, která se týkala evidence nových informací u produktů a skladových umístění, ale především jsem vytvořil stránku s návrhy na přesun. Tímto jsem realizoval frontedovou část rozšíření a splnil hlavní cíl celé práce.

Implementaci jsem následně podrobil uživatelskému testování, které zahrnovalo pět scénářů, které pomohly odhalit některé nedostatky a otevřely další prostor budoucímu navázání na tuto práci. Výsledné rozšíření je připraveno na propojení s backendovou částí, kterou v rámci své bakalářské práce realizovala Daniela Kováčová. Toto propojení docílí korektního ukládání skladových položek dle frekvence jejich vyskladnění a tím se zrychlí časté skladové procesy a umožní tak pracovníkům skladu lépe rozložit čas mezi jednotlivé aktivity.

Věřím, že výsledky mé práce budou patřičně využity a napomohou k urychlení častých skladových procesů ve skladech, které využívají pro svůj chod skladový systém Atlantis.

Bibliografie

1. MALEC, Oldřich. *FRONTEND SKLADOVÉHO SYSTÉMU* [online]. 2020 [cit. 2022-03-30]. Dostupné z: https://dspace.cvut.cz/handle/10467/86593. Dipl. pr. ČVUT v Praze, Fakulta informačních technologií, Katedra softwarového inženýrství.
2. KOVÁŘ, Pavel. *BACKEND SKLADOVÉHO SYSTÉMU* [online]. 2019 [cit. 2022-03-30]. Dostupné z: https://dspace.cvut.cz/handle/10467/82591. Dipl. pr. ČVUT v Praze, Fakulta informačních technologií, Katedra softwarového inženýrství.
3. ALSHAMRANI, Adel; BAHATTAB, Abdullah. A comparison between three SDLC models waterfall model, spiral model, and Incremental/Iterative model. *International Journal of Computer Science Issues (IJCSI)* [online]. 2015, roč. 12, č. 1, s. 106 [cit. 2022-04-14]. Dostupné z: https://www.academia.edu/download/36637147/SDLC.pdf.
4. BALAJI, Sundramoorthy; MURUGAIYAN, M Sundararajan. Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. *International Journal of Information Technology and Business Management* [online]. 2012, roč. 2, č. 1, s. 26–30 [cit. 2022-04-14]. Dostupné z: https://mediaweb.saintleo.edu/Courses/COM430/M2Readings/WATEERFALLVs%5C%20V-MODEL%5C%20Vs%5C%20AGILE%5C%20A%5C%20COMPARATIVE%5C%20STUDY%5C%20ON%5C%20SDLC.pdf.
5. STOICA, Marian; MIRCEA, Marinela; GHILIC-MICU, Bogdan. Software development: Agile vs. traditional. *Informatica Economica* [online]. 2013, roč. 17, č. 4 [cit. 2022-04-14]. Dostupné z: http://www.revistaie.ase.ro/content/68/06%5C%20-%5C%20Stoica,%5C%20Mircea,%5C%20Ghilic.pdf..
6. RUPARELIA, Nayan B. Software development lifecycle models. *ACM SIGSOFT Software Engineering Notes* [online]. 2010, roč. 35, č. 3, s. 8–

13 [cit. 2022-04-22]. Dostupné z: https://dl.acm.org/doi/abs/10.1145/1764810.1764814.

7. SUTHERLAND, Jeff; SCHWABER, Ken. The scrum papers. *Nuts, Bolts and Origins of an Agile Process* [online]. 2007 [cit. 2022-04-22]. Dostupné z: https://www.academia.edu/download/64196137/scrumpapers.pdf.
8. ŠMÍD, Vladimír. Management informačního systému. [Online]. 2003 [cit. 2022-04-22]. Dostupné z: http://www.fi.muni.cz/~smid/managis.html.
9. NUSEIBEH, Bashar; EASTERBROOK, Steve. Requirements Engineering: A Roadmap. In: *Proceedings of the Conference on The Future of Software Engineering* [online]. Limerick, Ireland: Association for Computing Machinery, 2000, s. 35–46 [cit. 2022-04-14]. ICSE '00. ISBN 1581132530. Dostupné z DOI: 10.1145/336512.336523.
10. KENDALL, Kenneth E.; KENDALL, Julie E. *Systems analysis and design* [online]. Prentice Hall Upper Saddle River, NJ, 2002 [cit. 2022-04-22]. Dostupné z: https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxpdDIzc2FkfGd4OjJmODlhMjNkNTQwOThjOWY.
11. RAGHAVAN, Sridhar; ZELESNIK, Gregory; FORD, Gary. *Lecture notes on requirements elicitation* [online]. 1994 [cit. 2022-04-22]. Tech. zpr. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST. Dostupné z: https://apps.dtic.mil/sti/citations/ADA278536.
12. PAETSCH, F.; EBERLEIN, A.; MAURER, F. Requirements engineering and agile software development. In: *WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003.* [Online]. 2003, s. 308–313 [cit. 2022-04-22]. Dostupné z DOI: 10.1109/ENABL.2003.1231428.
13. ŽÁRA, Ondřej. *JavaScript* [online]. Computer Press, 2015 [cit. 2022-04-29]. Dostupné z: https://books.google.cz/books?hl=cs&lr=&id=BBy2DwAAQBAJ&oi=fnd&pg=PP1&dq=javascript&ots=8yOYdLaHof&sig=fgNgduozcADIG0fn0ZnmnnibwoY&redir_esc=y#v=onepage&q=javascript&f=false.
14. SHAHZAD KHAN. *WHAT IS A JAVASCRIPT FRAMEWORK?* [Online] [cit. 2022-05-01]. Dostupné z: https://generalassemb.ly/blog/what-is-a-javascript-framework/.
15. EVAN YOU. *Introduction* [online] [cit. 2022-04-29]. Dostupné z: https://v2.vuejs.org/v2/guide/.
16. ARTEM ZAKHARCHENKO. *Mock Service Worker* [online] [cit. 2022-04-29]. Dostupné z: https://mswjs.io/docs/.

17. JETBRAINS S.R.O. *PhpStorm* [online] [cit. 2022-04-29]. Dostupné z: `https://www.jetbrains.com/phpstorm/`.

18. GOOGLE LLC. *Material Icons Guide* [online] [cit. 2022-05-01]. Dostupné z: `https://developers.google.com/fonts/docs/material_icons`.

19. LEWIS, James R. Usability testing. *Handbook of human factors and ergonomics*. 2006, roč. 12, e30.

20. BASTIEN, J.M. Christian. Usability testing: a review of some methodological and technical aspects of the method. *International Journal of Medical Informatics*. 2010, roč. 79, č. 4, e18–e23. ISSN 1386-5056. Dostupné z DOI: `https://doi.org/10.1016/j.ijmedinf.2008.12.004`. Human Factors Engineering for Healthcare Applications Special Issue.

Příloha A

Seznam použitých zkratek

API Application Programming Interface

SP1 Softwarový týmový projekt 1

SP2 Softwarový týmový projekt 2

FP Funkční požadavek

HTML Hypertext Markup Language

CSS Cascading Style Sheets

PHP PHP: Hypertext Preprocessor

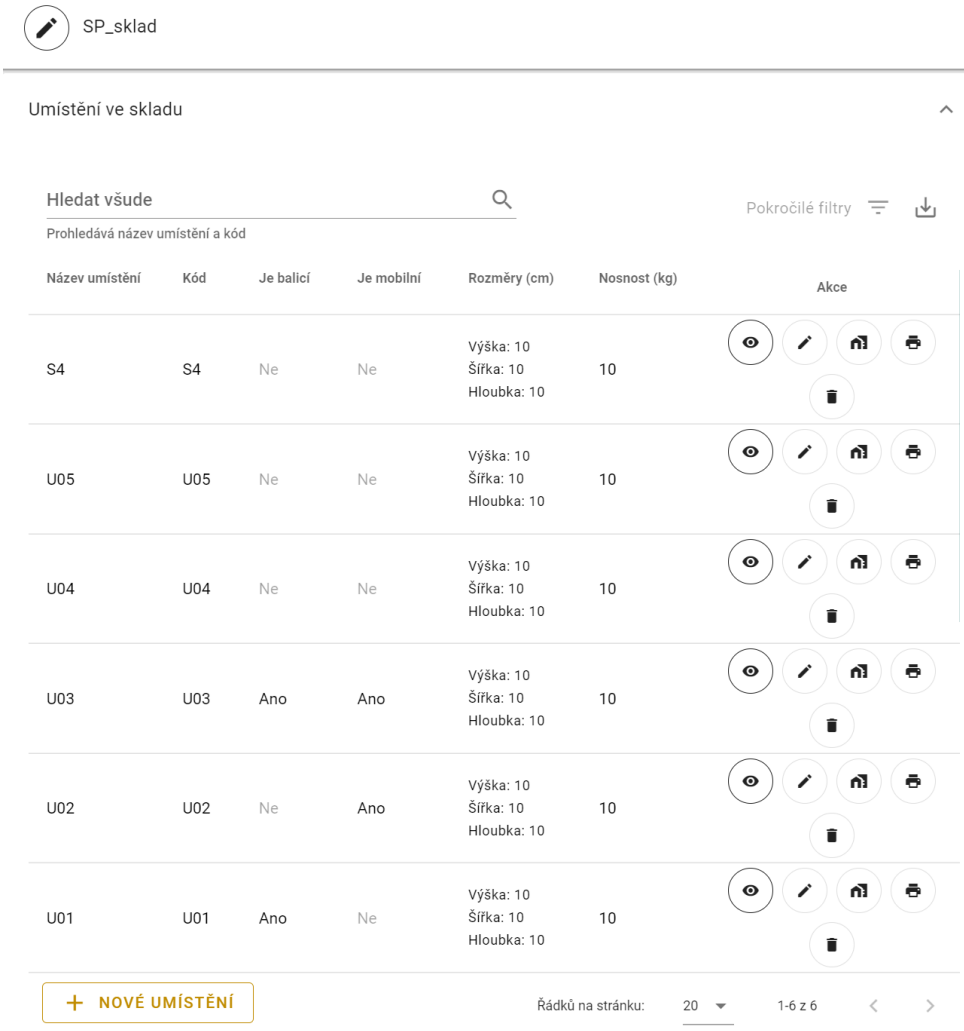
IDE Integrated Development Environment

URL Uniform Resource Locator

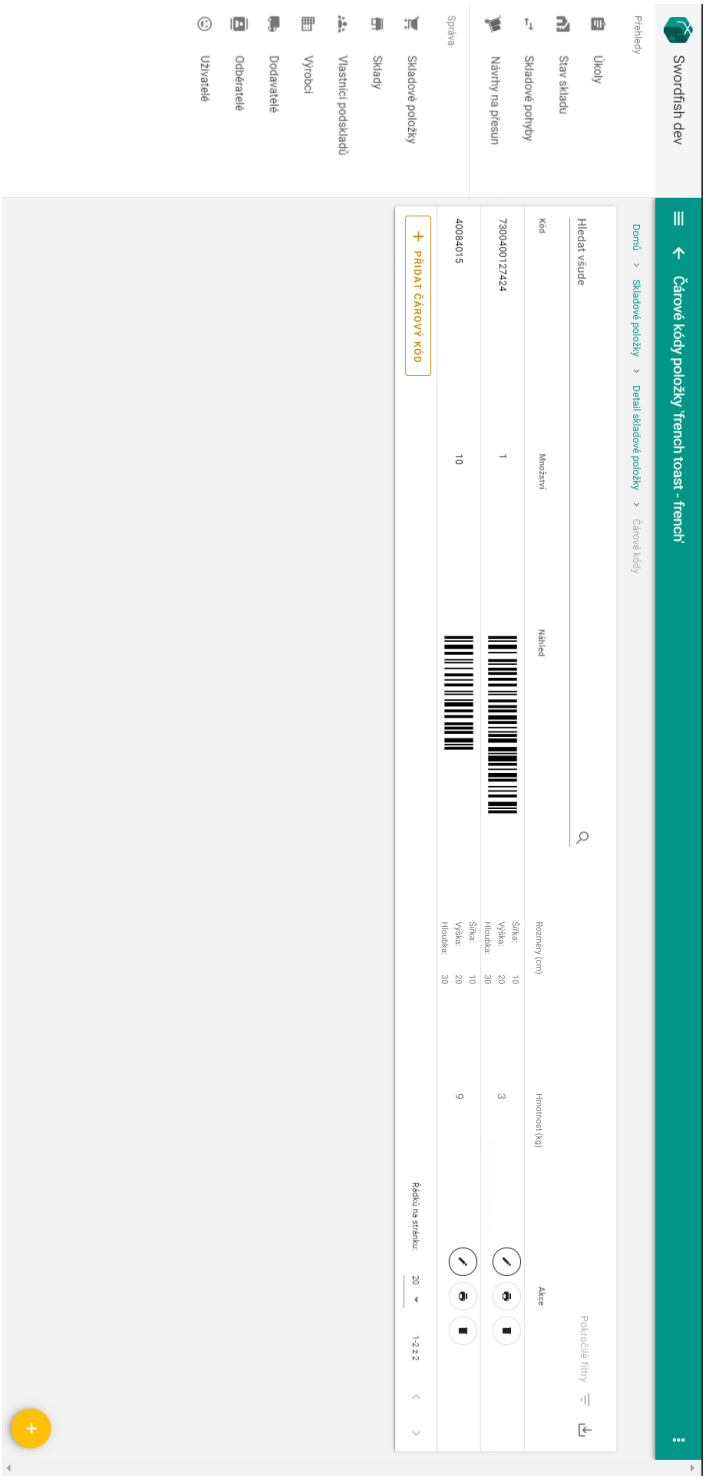
TS Testovací scénář

Příloha B

Návrhy obrazovek

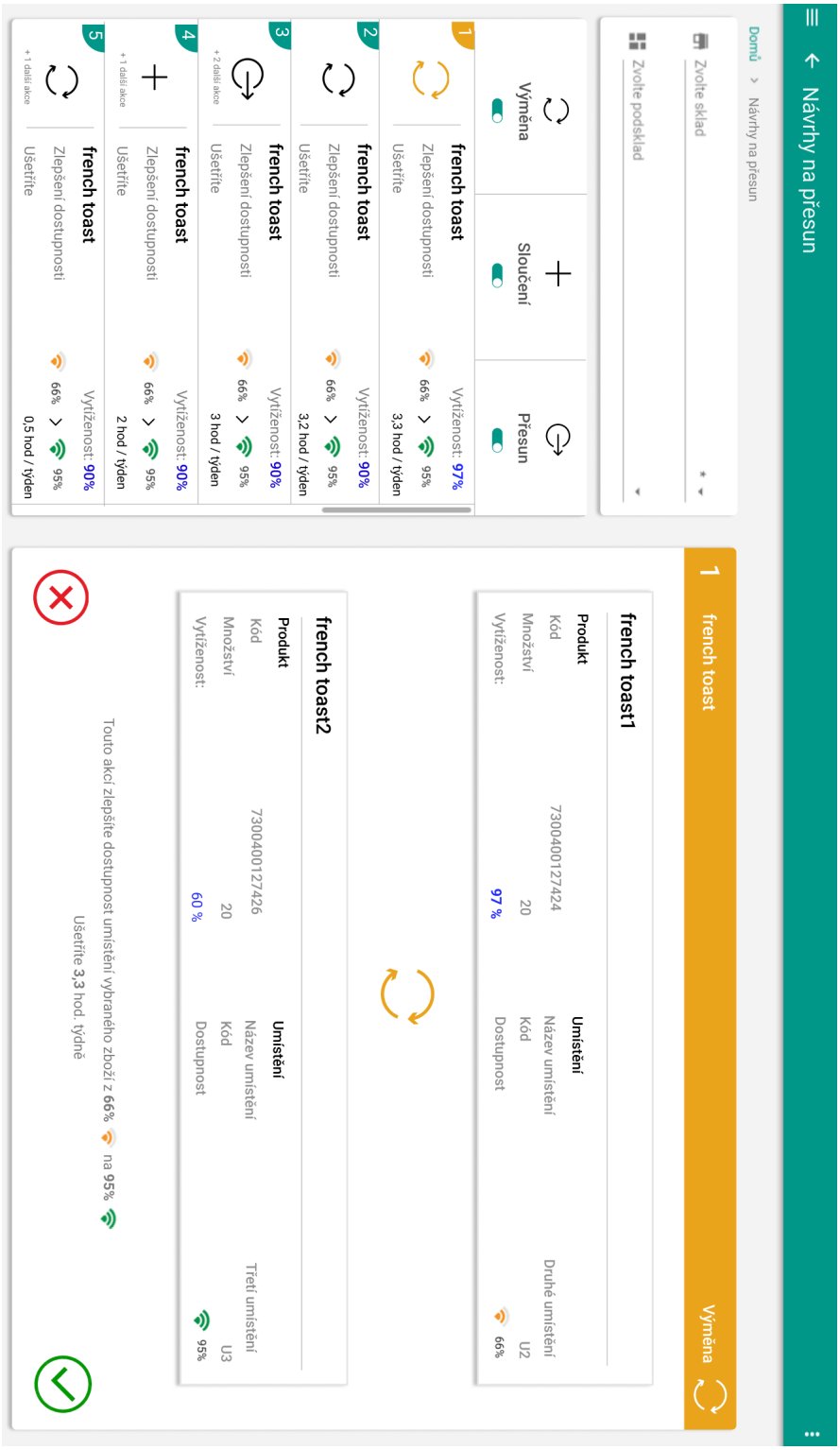


Obrázek B.1: Návrh zobrazení rozměrů umístění

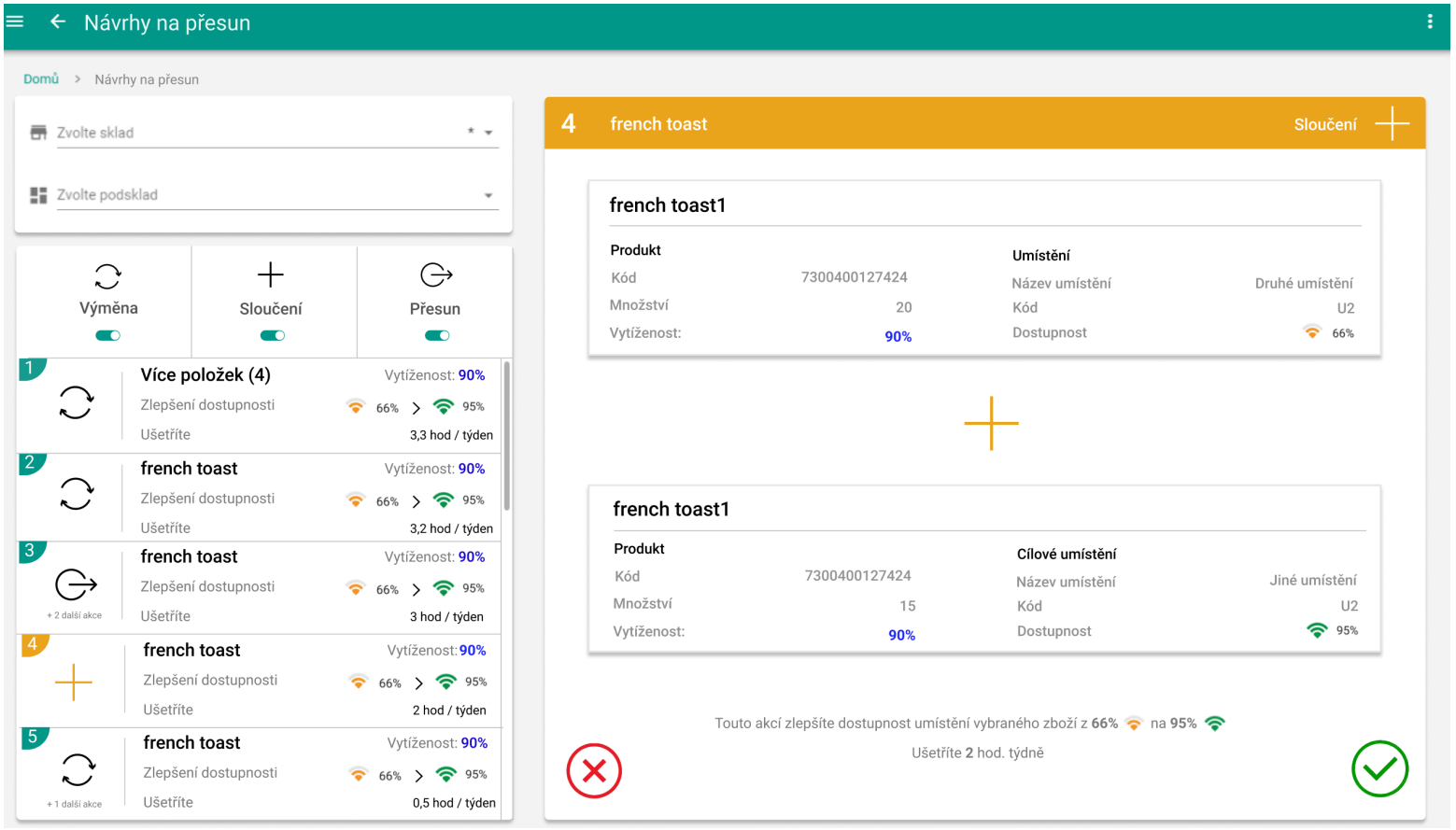


Obrázek B.2: Návrh zobrazení rozměrů u čárových kódů produktu produktů

Obrázek B.3: Návrh obrazovky s přesunem



Obrázek B.4: Návrh obrazovky s výměnou

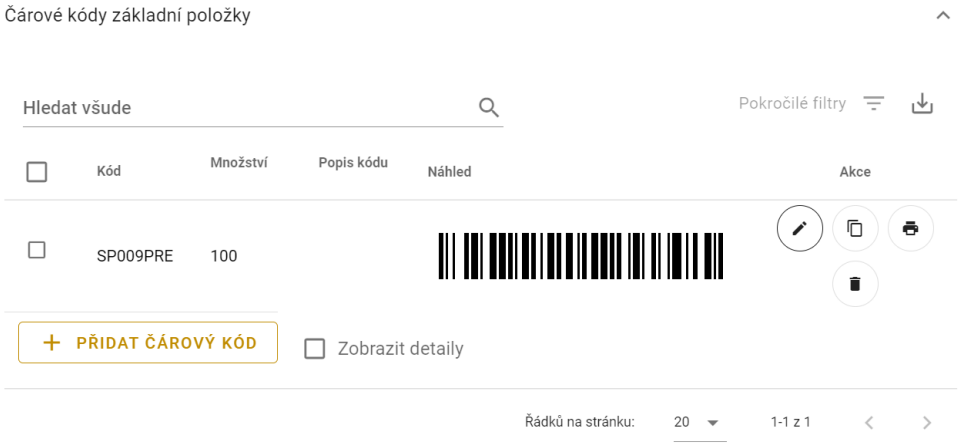


Obrázek B.5: Návrh obrazovky se sloučením

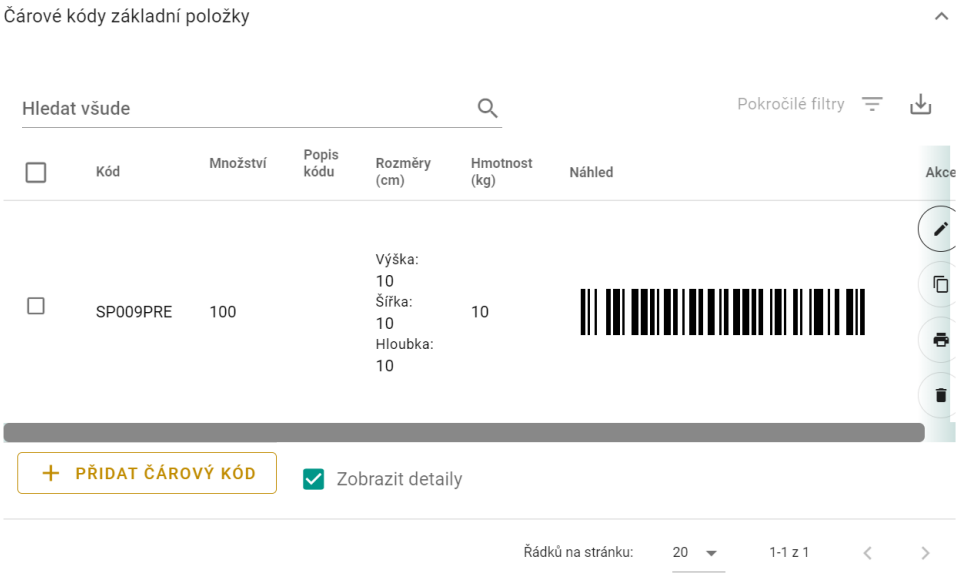
Příloha C

Výsledné obrazovky

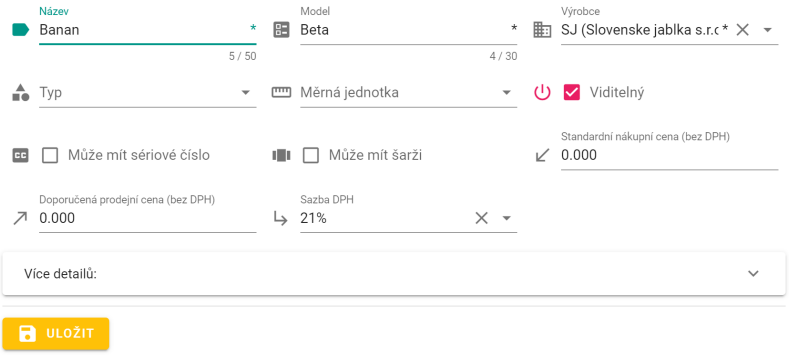
Obrázek C.1: Výsledná obrazovka s návrhem na doplnění kusového zboží



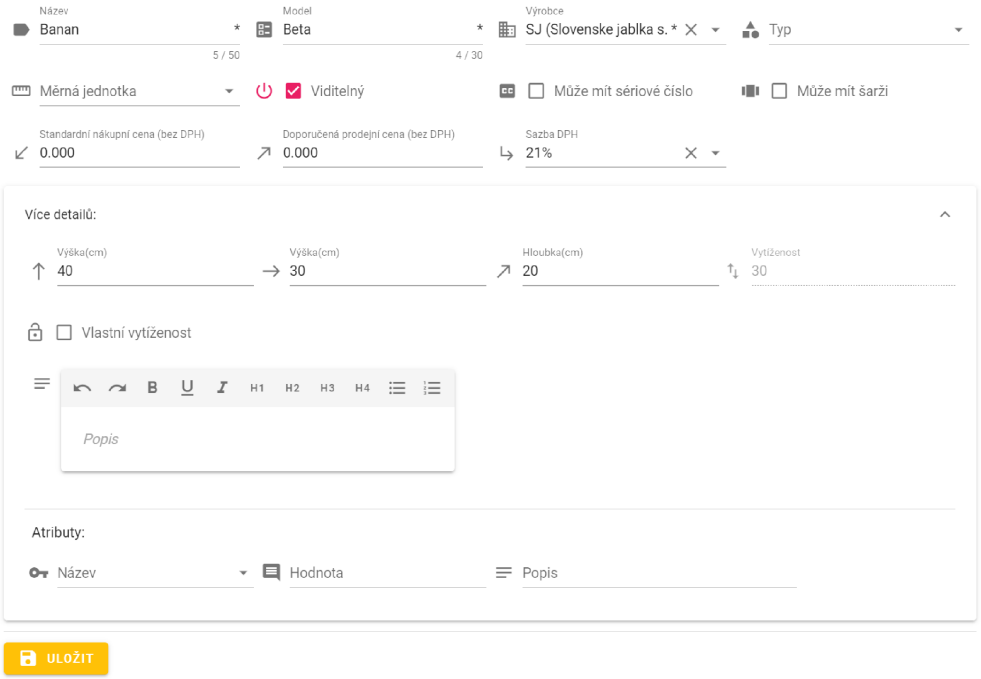
Obrázek C.2: Výsledná tabulka čárových kódů bez zobrazení detailů



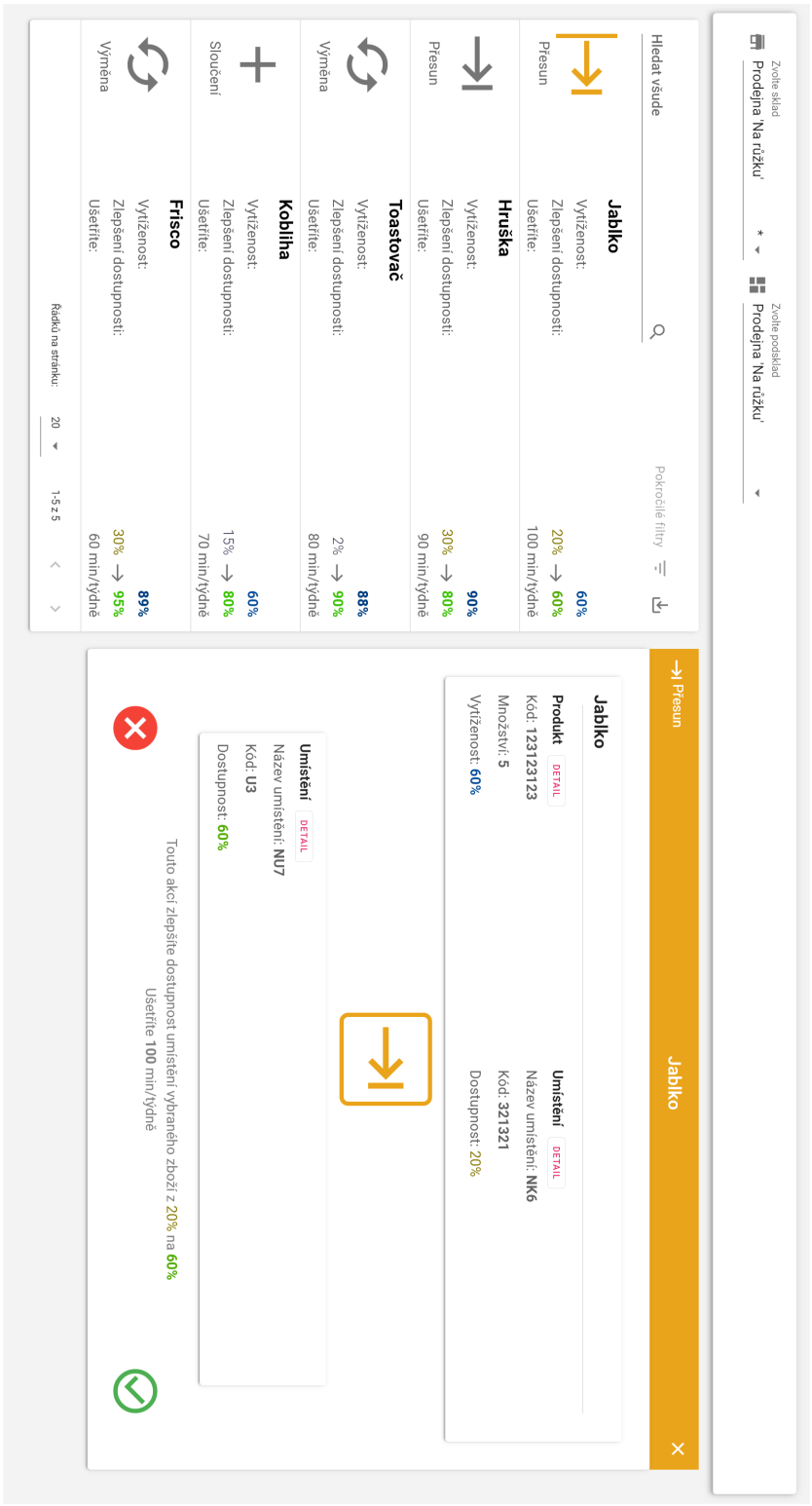
Obrázek C.3: Výsledná tabulka čárových kódů s detaily



Obrázek C.4: Výsledná obrazovka s editací produktu



Obrázek C.5: Výsledná obrazovka s editací produktu a detaily



Obrázek C.6: Výsledná obrazovka návrhu na přesun po testování

Příloha D

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `impl.txt` ..... zdrojové kódy implementace
  - `úpravy pro nasazení.pdf` ..... popis úprav nutných pro nasazení
  - `definice API.yaml` ..... definice API požadavků
- `thesis.pdf` ..... text práce ve formátu PDF
- `thesis.tex` ..... text práce ve formátu Latex
- `scénáře.pdf` ..... testovací scénáře
- `obrazovky`
  - `návrhy.pdf` ..... vytvořené návrhy obrazovek
  - `výsledky.pdf` ..... výsledné obrazovky