FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Mapa skladu – dokončení
Student:	Jan Šuráň
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 2023/2024

Pokyny pro vypracování

Cílem této práce je navázat na rozpracovaný projekt mapy skladu, který realizoval v rámci své bakalářské práce Bc. Vojtěch Kopecký.

Postupujte v těchto krocích:
1. Analyzujte současný stav projektu od Bc. Vojty Kopeckého
2. Na základě analýzy a závěrů předešlé práce navrhněte vhodné kroky k reálné integraci mapy do skladového systému Atlantis.
3. Navržené kroky řádně realizujte včetně potřebné implementace na klientské a serverové části. Zajistěte řádné ukládání a mapování na reálné skladové umístění.
4. Výsledné řešení řádně otestujte k dosažení potřebné použitelnosti.
5. Z dat ze skladového systému se pokuste sestavit heatmapu vytížení skladových pozic, anonymizovaná data si vyžádejte u vedoucího práce.
6. Shrňte dosažené výsledky

Elektronicky schválil/a Ing. Michal Valenta, Ph.D. dne 27. září 2022 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Mapa skladu – dokončení

Jan Šuráň

Katedra softwarového inženýrství
Vedoucí práce: Ing. Jiří Hunka

28. června 2023

Poděkování

Rád bych poděkoval vedoucímu mé bakalářské práce Ing. Jiřímu Hunkovi za jeho rady, podporu, organizační zajištění a možnost podílet se na vývoji aplikace, která se využívá v praxi. Pracovníkům společnosti Jagu s.r.o. patří pak velký dík především za množství rad a konzultací, které mi poskytli. Také bych rád poděkoval testerům, kteří věnovali svůj čas účasti v testování použitelnosti výsledného řešení, a v neposlední řadě mé rodině, která mě při realizaci této práce podporovala.

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 28. června 2023 . . . . . .

Odkaz na tuto práci

Šuráň, Jan. *Mapa skladu – dokončení*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2023.

Abstrakt

Bakalářská práce se věnuje dokončení projektu mapy skladu ve skladovém systému Atlantis. Nejprve analyzuje problematiku a současný stav projektu mapy skladu. Dále se věnuje návrhu úprav a rozšíření frontendové části editoru mapy a také návrhu tvorby backendu mapy skladu. Navržené úpravy a rozšíření frontendu jsou implementovány za využití frameworku Vue.js a knihoven Vuex a Vuetify. Backend mapy skladu je implementován v jazyce PHP za využití frameworku Symfony a knihovny Doctrine. Je vhodně začleněn do současného backendu skladového systému. Práce se také zabývá implementací zobrazení heatmapy vytíženosti skladových pozic. Vytvořené řešení je podrobeno důkladnému testování použitelnosti tak, aby bylo možné mapu skladu reálně integrovat do skladového systému. Výsledkem práce je funkční editor mapy skladu, ve kterém lze zaznamenat veškerá data o skladu a jeho objektech, oblastech a umístěních. Na tuto práci lze navázat vývojem optimalizací skladových procesů, které by využily data mapy.

Klíčová slova skladový systém, mapa skladu, správa skladu, frontend, backend, Vue.js, Vuetify, Vuex, PHP, Symfony, Doctrine

Abstract

This bachelor's thesis focuses on completing the warehouse map project in the Atlantis warehouse system. First, it analyzes the issues and the current state of the warehouse map project. Then it deals with the design of the map editor's front-end modifications and expansion, as well as the creation of the warehouse map's back-end. The proposed front-end modifications and extensions are implemented using the Vue.js framework and the Vuex and Vuetify libraries. The warehouse map's back-end is implemented in PHP using the Symfony framework and the Doctrine library and it is conveniently integrated into the current warehouse system back-end. The thesis also deals with implementing the display of the warehouse positions utilization heatmap. The created solution is subjected to thorough usability testing so that the warehouse map can be realistically integrated into the warehouse system. The result of the thesis is a functional warehouse map editor, in which all data about the warehouse and its objects, areas, and locations can be recorded. This thesis can be followed up by developing warehouse process optimizations that would use the map data.

Keywords warehouse system, warehouse map, warehouse management, frontend, backend, Vue.js, Vuetify, Vuex, PHP, Symfony, Doctrine

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Skladování je v dnešní době jednou z nejdůležitějších součástí moderní logistiky. Jedná se o spojovací článek mezi výrobními subjekty a odběrateli, který hraje klíčovou roli v zajištění potřebné úrovně dodavatelského servisu. [1] Vzhledem k tomu, že sklady zastávají v tomto systému tak důležitou roli, je třeba je efektivně spravovat a evidovat zboží a materiál, který těmito sklady prochází. K těmto účelům se zde využívají informační systémy, které evidují všechna data, která jsou důležitá ke správě skladu a optimalizují využití skladových prostor k co nejefektivnějšímu provozu při co možná nejnižších nákladech. Takovým systémem je také skladový systém Atlantis od společnosti Jagu s.r.o.

Systém Atlantis v současné době podporuje optimalizace skladových procesů, které v něm byly zavedeny na základě bakalářských prací „Optimalizace nejčastějších procesů ve skladovém systému“ [2], „Frontend optimalizace skladových procesů systému Atlantis“ [3] a „Backend optimalizace skladových procesů systému Atlantis“ [4]. Systém optimalizuje pohyb a pozici zboží, které se nachází ve skladě na určených skladových umístěních. Eviduje se vytíženost, dostupnost umístění, položky, které se zde nachází, a další důležité údaje. Na základě vyhodnocení těchto dat jsou navrhovány vhodné změny, které zefektivňují chod skladu.

Při hledání dalších možností ke zlepšení provozu skladu však narážíme na to, že v systému není evidována přesná poloha umístění a tedy i zboží ve skladě. Pokud bychom chtěli například seřadit skladníkovi umístění, na kterých má pracovat v rámci jednoho úkolu za sebou tak, aby ušlá vzdálenost mezi umístěními a tedy i doba, za kterou úkol splní, byla co nejnižší, nejsme toho schopni. Díky znalosti pozic skladových umístění bychom také mohli shlukovat úkoly ve skladu podle části skladu, ve které se skladník pohybuje, nebo například umisťovat produkty, které se nejčastěji expedují na nejbližší a nejlépe dostupná místa.

Tuto problematiku jsme se rozhodli řešit v rámci projektového týmu v před-

mětech BI-SP1 a BI-SP2[1]. Na to následně navázal jeden ze členů našeho týmu Bc. Vojtěch Kopecký, který připravil grafický návrh a prototyp editoru mapy skladu, ve kterém se data o poloze umístění budou zaznamenávat [5]. Na práci Bc. Kopeckého navazuje tato bakalářská práce, která si klade za cíl dokončit projekt mapy skladu.

Cíle práce

Cílem práce je navázat na rozpracovaný projekt a mapu skladu dokončit. Dále potom pomocí dat poskytnutých mapou skladu sestavit heatmapu vytíženosti skladových pozic.

Nejdříve analyzuji současný stav skladového systému a projektu mapy skladu od Bc. Vojtěcha Kopeckého. Dále dle této analýzy stanovím postup dalšího vývoje a navrhnu kroky, které povedou k reálné integraci mapy skladu do skladového systému. V této části bude třeba dobře promyslet a zajistit řádné mapování umístění mapy skladu na umístění již definovaná ve skladu. Navržené řešení následné implementuji a řádně otestuji.

[1] Softwarový týmový projekt 1 a 2 – předměty vyučované na FIT ČVUT

Kapitola 1

Analýza

Dříve, než bude možné stanovit kroky, které povedou k dokončení projektu mapy skladu, je třeba provést důkladnou analýzu současného stavu systému Atlantis, problematiky mapy skladu a řešení vytvořeného Bc. Kopeckým.

Na začátku této kapitoly představím skladový systém a definuji, jaké objekty se ve skladu nachází, což je důležité pro stanovení toho, co by měla mapa skladu zachycovat. Dále představím projekt mapy skladu a zanalyzuji současný stav, ve kterém se projekt nachází. Poté zformuluji požadavky, které jsou kladeny na mapu skladu a vytvořím model případů užití.

1.1 Skladový systém Atlantis

Základní stavební kameny Skladového systému Atlantis vyvíjeného společností Jagu s.r.o. vznikly v roce 2019 v diplomových pracích Ing. Oldřicha Malce [6] a Ing. Pavla Kováře [7]. Jedná se o nástupce systému Sysel od stejné společnosti a je určen především pro správu skladových prostor, evidenci zboží a procesů, které ve skladech probíhají. Je realizován formou webové aplikace s architekturou klient-server a skládá se ze čtyř komponent. V backendu je to autorizační server s názvem Angler a aplikace Octopus, na frontendu poté aplikace Swordfish, na kterou je navázana mobilní aplikace. [7]

1.1.1 Položky a skladová umístění

Ve skladech jsou v systému evidovány skladové položky (někdy také zboží), tedy předměty, které se ve skladu uchovávají. Ty se ve skladu nachází na takzvaných skladových umístěních.

Skladové umístění je konkrétní místo ve skladě, které má nějaký název a kód. V běžných skladech se nachází někdy i tisíce takových skladových umístění. Většina z nich je nejčastěji uspořádána v regálech, objektech členěných horizontálně sloupci a vertikálně patry na jednotlivé buňky, které obsahují daná umístění. Entita regálu byla v systému Atlantis zavedena s první

verzí editoru mapy skladu. Umístění jsou zde systematicky pojmenovávána za pomoci jmenovacích systémů. Přesný popis těchto systémů je možné najít v bakalářské práci Bc. Kopeckého [5].

Ve skladu také existují umístění, která se nenachází v regálech. Může se například jednat o umístění, na kterých se zboží nachází při balení, nebo paletová umístění, kde zboží čeká před odesláním či po jeho přijetí do skladu.

1.2 Mapa skladu

Na rozšíření skladového systému Atlantis, které obsahuje i nástroj mapy skladu, jsme začali pracovat v rámci předmětů BI-SP1 a BI-SP2 v roce 2021. Zadáním tohoto projektu bylo navrhnout a analyzovat rozšíření, které by se staralo o optimalizace skladových procesů, přesněji o shlukování úkolů podle jejich polohy, trasování skladníka a řazení položek uvnitř úkolů. V rámci tohoto předmětu jsme identifikovali potřebu evidence pozice umístění a dalších důležitých a význačných bodů ve skladu. Tak vznikla myšlenka mapy skladu, která by evidovala přesné souřadnice skladového umístění a umožnila by realizaci potřebných optimalizací.

Na tuto analýzu následně navázal Bc. Vojtěch Kopecký svou bakalářskou prací, ve které částečně navrhl a implementoval uživatelské rozhraní editoru mapy skladu.

1.3 Současný stav projektu

V současné chvíli je v projektu mapy skladu implementovaná část uživatelského rozhraní – editor mapy. Ten je implementován ve frontendu skladového systému – aplikaci Swordfish a umožňuje základní operace při zobrazení mapy, tedy editaci rozměrů, přiblížení atp. Dále je implementována funkce tvorby, editace a smazání regálu a umístění v něm. Editor však zatím neumožňuje uložení dat mapy skladu ani namapovaní umístění zaznamenaných v mapě na umístění, která jsou v daném skladu již evidována.

Aktuálně navržený editor mapy se nachází v systému v zobrazení detailu skladu v sekci sklady. Jak lze vidět na obrázku 1.1, editor je rozdělen do čtyř sekcí – panel zobrazení, panel nástrojů, okno s mapou a postranní panely.

1.3.1 Panel zobrazení

Panel zobrazení umožňuje uživateli přepínat mezi pohledem na mapu a pohledem na jednotlivý regál. Pohled na regál je možné otevřít pouze v případě, je-li v mapě skladu zvolen regál, jehož detail chce uživatel zobrazit (obr. 1.2). Kliknutím na položku detail regálu přepne uživatel okno mapy na zobrazení mapy regálu (obr. 1.4).

1.3.2 Panel nástrojů

Panel nástrojů je zobrazen přímo nad oknem mapy a je s ním graficky úzce spjat tak, aby bylo jednoznačné, že se jedná o panel poskytující nástroje pro editaci mapy zobrazené v okně mapy. Je-li však zobrazena mapa regálu, není panel nástrojů viditelný.

První položkou, která se v panelu nachází, je ovládání zobrazení mapy – přiblížení a oddálení. Tato funkce je realizována posuvníkem, který je obklopen dvěma ikonami lup identifikujícími směr přibližování (obr. 1.1).

Panel nástrojů dále umožňuje v základním nastavení uživateli přidávat skladové objekty. Aktuálně je implementována pouze možnost přidat regál (obr. 1.1 a 1.3). Je-li vybrán v mapě konkrétní objekt, zmizí možnosti přidávání nových objektů a naopak se zobrazí ikony nástrojů umožňující práci s vybraným objektem (obr. 1.2). Jedná se o následující nástroje:

Pohyb objektu: Tento nástroj umožňuje uživateli přesunout objekt na mapě na jinou pozici. Uživatel klikne na místo na mapě, na které chce objekt přesunout. Pokud by přesun nezpůsobil kolizi s jiným objektem, je daný objekt přesunut a zobrazen uživateli na nové pozici.

Rotace objektu: Objekt je otočen ve skladu z vertikální do horizontální polohy a naopak.

Kopírovat objekt: Tento nástroj umožňuje duplikovat vybraný objekt. Uživatel klikne na pozici na mapě, na kterou chce objekt zduplikovat. Pokud by nový objekt nezpůsobil kolizi s jiným objektem ve skladu, je na dané pozici vytvořen nový objekt, který má stejné vlastnosti, jako objekt předchozí. U regálu se však inkrementuje poslední symbol názvu regálu a změní se tedy i kódy všech umístění v regálu.

V panelu nástrojů chybí možnost odstranění objektu. Domnívám se, že tuto funkci by zde bylo vhodné doplnit.

1.3.3 Okno mapy

Na panel nástrojů je navázáno okno mapy skladu. Jedná se o čtvercovou síť. Světle šedé buňky reprezentují průchozí prostory skladu. Do této sítě jsou umisťovány objekty – aktuálně pouze regály (obr. 1.3). Klikne-li uživatel na daný objekt, je objekt vybrán a v panelu nástrojů se zobrazí nabídka možností práce s ním.

1.3.4 Postranní panely

Tato část rozhraní se skládá ze tří samostatných karet. Karty se nachází v pravé části editoru.

1.3.4.1 Seznam skladových objektů

První kartou je seznam objektů skladu viditelný například na obrázku 1.1. V textové podobě je zobrazen seznam objektů, které se nachází na mapě (v aktuální chvíli se zde nachází pouze záložka pro regály). Klikne-li uživatel na danou položku seznamu, je objekt vybrán a pozadí položky v seznamu se vybarví šedou barvou tak, aby bylo zvýrazněno, která položka byla vybrána.

Tato karta není zobrazena, když je v panelu nástrojů aktivní detail regálu a okno mapy zobrazuje mapu regálu.

1.3.4.2 Detail objektu

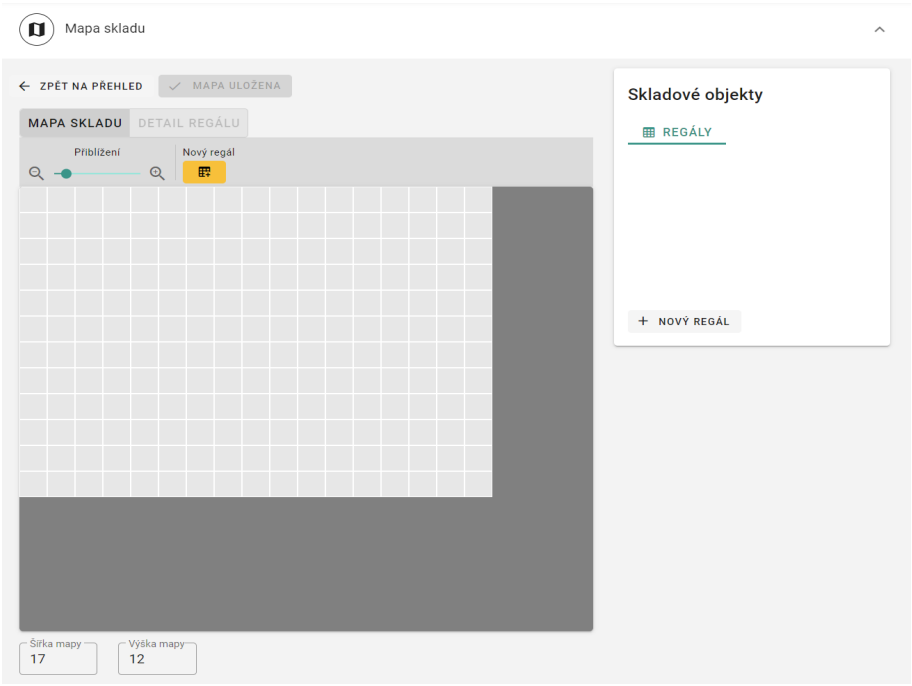
Karta detailu objektu se zobrazí vždy při vybrání objektu v mapě (obr. 1.2). Umožňuje například změnu názvu objektu či editaci jeho rozměrů. Jsou zde také tlačítka pro odstranění objektu a zobrazení detailu regálu.

1.3.4.3 Editace detailu regálu

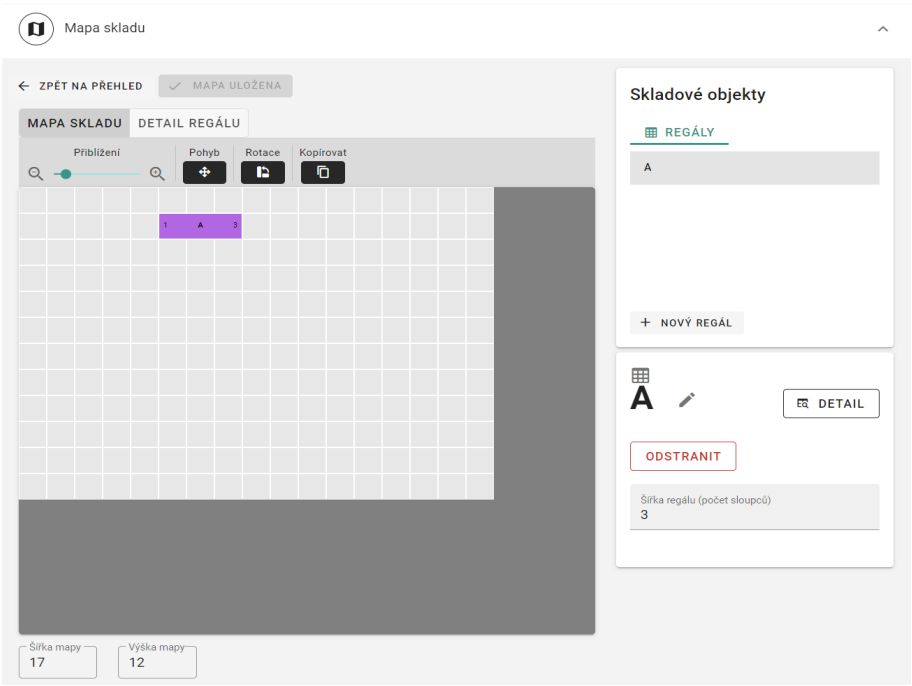
Karta editace detailu regálu je zobrazena zároveň s mapou regálu a umožňuje změnu jeho vlastností a editaci rozměrů umístění v něm (obr. 1.4).

Na kartě obecné lze měnit počet pater regálu a systém, podle kterého jsou umístění v regálu pojmenovávána. V polích počáteční symbol patra a sloupce je možné měnit, jakým symbolem bude jmenná řada začínat. Vedle těchto polí lze dále volit možnost, jakým směrem se budou umístěním v regálu přidělovat jména. Určení těchto vlastností umožňuje automatické generování kódů umístění v regálu. Uživatel tak nemusí pro každé umístění vypisovat jeho kód. Systém automaticky kódy umístění v regálech generuje.

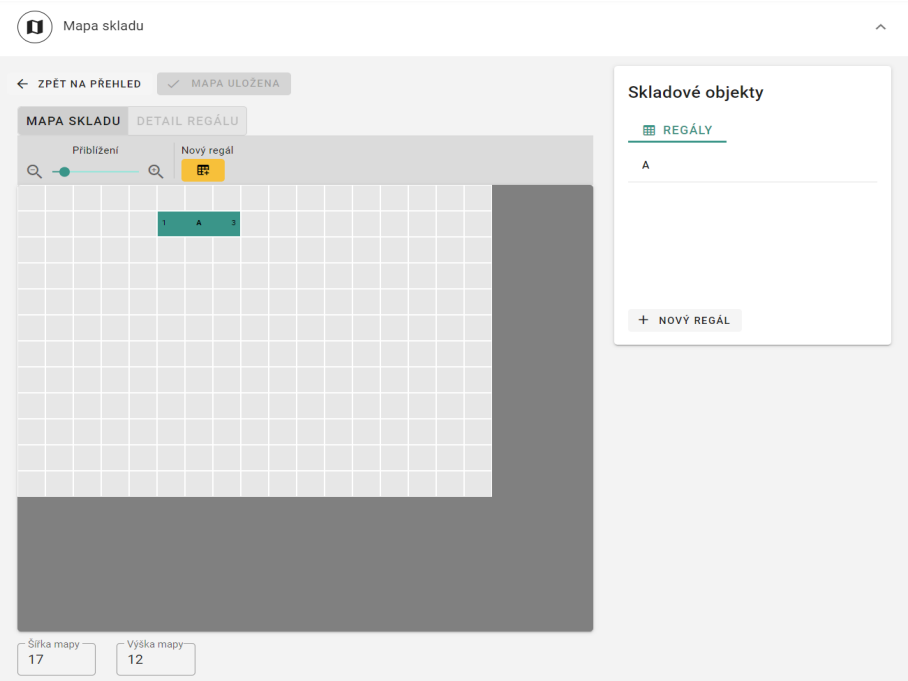
Po přechodu na kartu rozměry je možné pro každou buňku regálu měnit velikost umístění, která se v ní nachází. Můžeme měnit šířku, výšku a hloubku umístění a přiřadit těmto rozměrům také barvu, kterou bude daná buňka zvýrazněna tak, aby se uživateli zjednodušila orientace v mapě regálu.



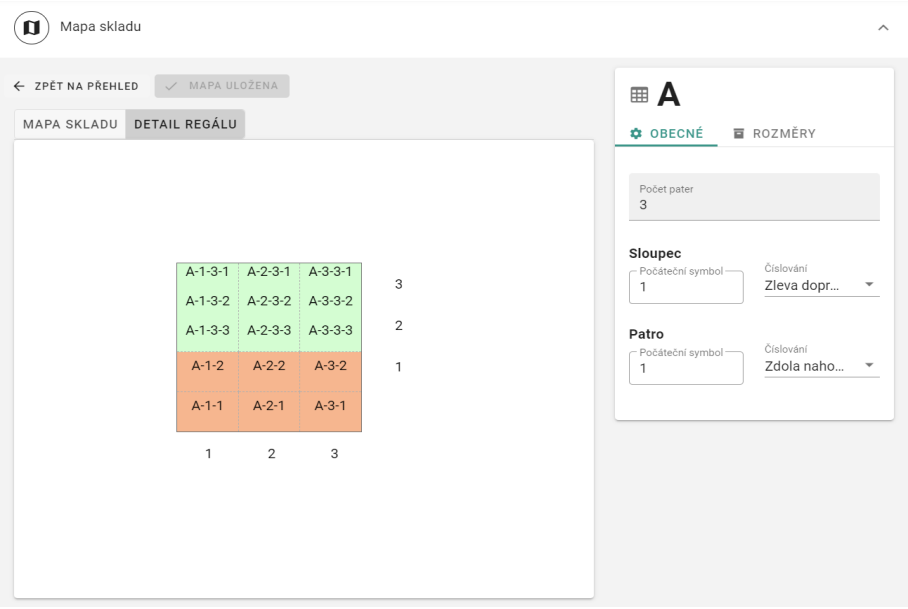
Obrázek 1.1: Editor mapy skladu od Bc. Kopeckého



Obrázek 1.2: Editor mapy skladu od Bc. Kopeckého – zobrazení při zvolení regálu



Obrázek 1.3: Editor mapy skladu od Bc. Kopeckého – regál umístěný v mapě



Obrázek 1.4: Editor mapy skladu od Bc. Kopeckého – detail regálu

1.4 Formulace požadavků

V této kapitole formuluji požadavky, které jsou kladeny na editor mapy skladu a roztřídím je dle systému klasifikace požadavků FURPS+.

Systémové požadavky jsou popisy funkčnosti a vlastností systému. Obecně je rozdělujeme na požadavky funkční a nefunkční. Funkční požadavky se zabývají činnostmi a chováním systému, nefunkční pak určují vlastnosti systému a omezení na něj kladená. [8]

Dle systému FURPS dále rozdělujeme požadavky pokrývající funkčnost (functionality), použitelnost (usability), spolehlivost (reliability), výkon (prerformance) a zabezpečení (security) systému. Kategorie požadavků na funkčnost odpovídá předchozí definici funkčních požadavků. Ostatní kategorie poté pokrývají nefunkční požadavky. Tento systém ještě rozšiřuje klasifikace FURPS+, která přidává kategorie požadavků na implementaci, rozhraní a podporovatelnost systému nebo také kategorie návrhových a fyzických omezení. [9]

1.4.1 Požadavky na funkčnost

FR1 Evidence mapových dat

Systém bude moci pro každý sklad evidovat mapu skladu. Bude umožněno, aby pro každý sklad mohlo být evidováno více map. Správa více map pro jeden sklad však nemusí být implementována v editoru mapy. Stačí, aby editor mapy umožňoval tvorbu a úpravu jedné mapy pro každý sklad.

Složitost: vysoká

Priorita: vysoká

FR2 Zobrazení mapových dat

Systém bude umožňovat zobrazit mapu skladu. V ní budou obsažené všechny informace, které jsou u mapy evidovány. Především se jedná o objekty, oblasti a umístění, která jsou v mapě zaznamenána.

Složitost: vysoká

Priorita: vysoká

FR3 Zaznamenání skladových prostor, objektů, oblastí

Uživatel bude moci definovat velikost skladu a do jeho mapy zanést a následně upravovat objekty, které se v něm nachází. Jedná se především o regály, skupiny samostatných umístění či překážky (například sloup, zeď atp.).

Dále bude možné do mapy zaznamenávat oblasti skladu, které jsou důležité pro provoz skladu. Jedná se především o prostory, ve kterých dochází k balení

produktů, jejich příjmu a výdeji, nebo například parkování vysokozvižných vozíků.

Složitost: vysoká

Priorita: vysoká

FR4 Zaznamenání polohy skladových umístění v mapě

Do mapy bude možné zaznamenat umístění, která se v daném skladu nachází. U každého umístění musí být evidována jeho horizontální poloha ve skladě. Dále u umístění, které se nachází v regále, je třeba evidovat jeho pozici v rámci tohoto regálu – tedy police a řada, v jaké se nachází.

Složitost: vysoká

Priorita: vysoká

FR5 Zaznamenání rozměrů umístění v regálech

V regálu bude možné zaznamenat velikost umístění, která se nachází v určité buňce regálu. Nejčastěji je rozměr umístění definován jako tři veličiny – výška, šířka a hloubka.

Složitost: vysoká

Priorita: střední

FR6 Synchronizace skladu dle mapy

Systém bude umožňovat úpravu stávajících skladových umístění dle evidované mapy skladu. Pokud se v mapě budou nacházet umístění, která nejsou evidována ve skladě, systém je ve skladě vytvoří. Pokud se naopak v mapě nebudou nacházet některá skladová umístění, systém je ze skladu odstraní.

Složitost: střední

Priorita: střední

1.4.2 Požadavky na použitelnost

UR1 Automatická tvorba umístění

Není žádoucí, aby uživatel manuálně zadával do systému každé umístění zvlášť. Sklad, který jsme například v rámci předmětu BI-SP1 navštívili, obsahoval několik tisíců umístění. Tvorba mapy pro takový sklad by zabrala přílišné množství času a uživatele by odradila od využití systému. Ve skladech jsou umístění systematicky pojmenována v regálech. Systém by měl tato umístění automaticky v mapě vytvářet.

Složitost: vysoká

Priorita: vysoká

1.4.3 Požadavky na implementaci

IR1 Využití technologií užitých ve frontendu a backendu skladového systému

Je třeba navázat na technologie využité v projektu mapy skladu, frontendu a backendu skladového systému tak, aby výsledné řešení bylo se systémem kompatibilní.

Složitost: střední

Priorita: vysoká

1.4.4 Požadavky na podporovatelnost

SR1 Návaznost optimalizací

Data mapy skladu musí být v systému uložena tak, aby byly údaje o poloze skladového umístění dostupné pro části systému, které budou na jejich základě zavádět optimalizace skladových procesů. Na mapu skladu bude pak možné navázat implementací těchto optimalizačních algoritmů.

Složitost: nízká

Priorita: vysoká

1.4.5 Fyzická omezení

PC1 Responsivita

Editor mapy skladu je určen především pro užití na osobním počítači. Z práce Bc. Kopeckého a diskuse s vedoucím práce vyplynulo, že se jedná se o součást systému, která nebude určena pro mobilní zařízení s menší obrazovkou. Přesto by mělo být na těchto zařízeních alespoň možné mapu skladu zobrazit.

Složitost: střední

Priorita: vysoká

1.5 Model případů užití

V této kapitole vytvořím model případů užití pro editor mapy skladu. Nejdříve se zaměřím na aktéry, kteří interagují s mapou skladu, a poté popíši samotné případy užití. Nakonec také shrnu, jaké funkcionality specifikované případy užití již současné řešení editoru mapy skladu pokrývá a které naopak v řešení schází.

1.5.1 Aktéři

Aktér neboli účastník modeluje typ role, kterou hraje entita (například uživatel, jiný systém či kus hardware), vykonávající nějaké chování nebo akci vedoucí k interakci s daným systémem. Vůči systému se však tato entita jeví jako externí. Aktéři nemusí nutně představovat konkrétní fyzické entity, ale pouze konkrétní aspekty některých entit (role), které jsou relevantní pro specifikaci souvisejících případů užití.

Primárním aktérem je aktér, který zahajuje interakci se systémem a vyzývá systém, aby poskytl jednu ze svých služeb. Primární aktér je často aktér, který spouští případ užití. Sekundární aktér naopak službu systému poskytuje. Systém v tomto případě zahajuje interakci s tímto aktérem. U cílů sekundárního aktéra se neočekává, že budou splněny případem užití. [10] [11]

Jak již definoval ve své práci Bc. Kopecký, mapu skladu bude v systému vytvářet a spravovat zaměstnanec skladu, který je zodpovědný za jeho správu. V alternativním případě pak bude s mapou pracovat firemní expert ze společnosti Jagu s.r.o. V obou případech se jedná o uživatele, který má v systému roli vedoucího skladu. V rámci editoru mapy skladu se bude jednat o jediného primárního aktéra.

1.5.2 Případy užití

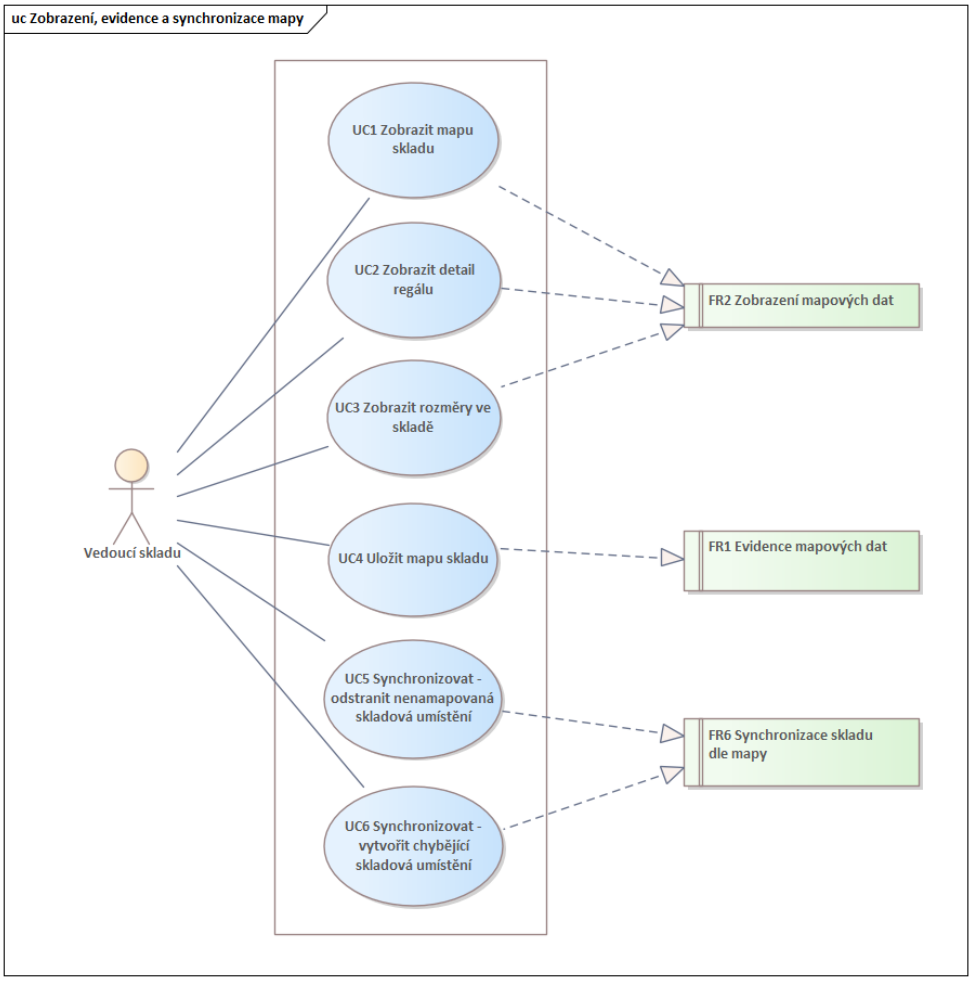
Model případů užití je detailní specifikací funkčních požadavků z pohledu uživatele. Typicky se jednotlivé požadavky rozpadají na několik případů užití, které popisují způsob, jakým aktéři interagují se systémem. [12] V této části popíši případy užití, které jsem v projektu identifikoval.

1.5.2.1 Zobrazení, evidence a synchronizace mapy

V první části se věnuji případům užití spojených se zobrazením a evidencí mapy a také synchronizací umístění mezi mapou a skladem. Diagram těchto případů užití je zaznamenán na obrázku 1.5.

UC1 Zobrazit mapu skladu

Vedoucí skladu může na stránce *Detail skladu* v kartě *Mapa skladu* zobrazit mapu daného skladu. V zobrazené mapě je vykreslen sklad s příslušnými ob-



Obrázek 1.5: Diagram případů užití – Zobrazení, evidence a synchronizace mapy

jekty a oblastmi. U objektů typu regál je zobrazen jeho název a názvy krajních sloupců.

UC2 Zobrazit detail regálu

Systém umožňuje uživateli s rolí vedoucího skladu u regálů v mapě zobrazit jejich detail. Uživatel vybere v mapě regál, pro který chce detail zobrazit a stiskne příslušné tlačítko. Systém zobrazí uživateli boční pohled, ve kterém jsou znázorněny buňky regálu s umístěními, která se v nich nachází. Jednotlivé buňky jsou v tomto pohledu obarveny dle rozměrů, které jsou jim přiřazeny.

UC3 Zobrazit rozměry ve skladě

Vedoucí může zobrazit rozměry umístění zaznamenané pro daný sklad v mapě. Systém zobrazí u každého rozměru jeho hodnoty, tedy výšku, šířku a hloubku umístění. Dále také zobrazí barvu, kterou budou v detailu regálu obarveny buňky, jejichž umístění mají dané rozměry.

UC4 Uložit mapu skladu

Uživatel s rolí vedoucího může uložit zaznamenaná mapová data. Po stisknutí příslušného tlačítka systém uloží všechna data mapy, která uživatel upravoval v editoru.

UC5 Synchronizovat – odstranit nenamapovaná skladová umístění

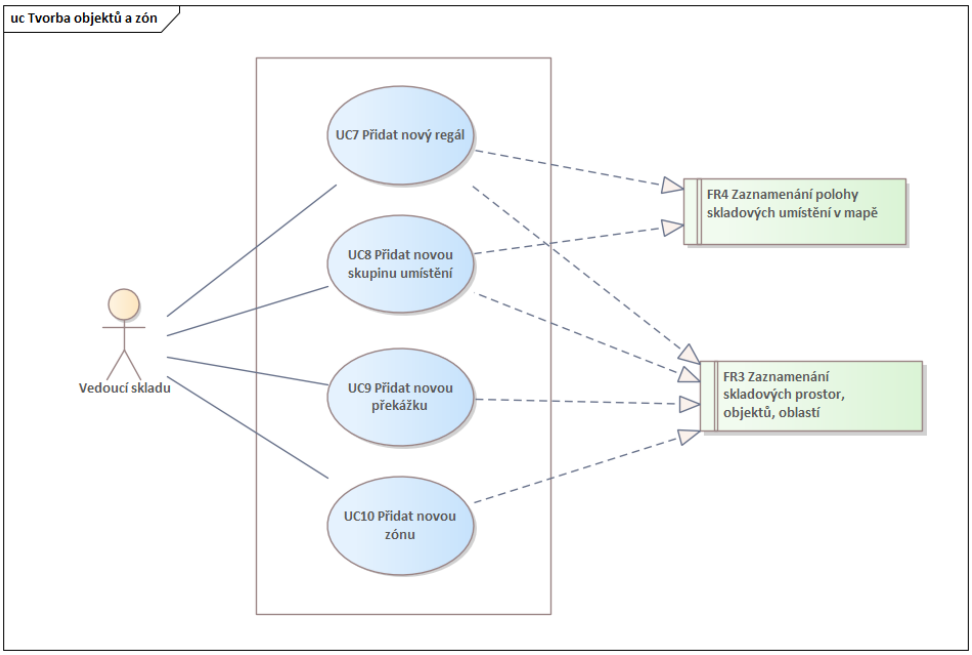
Systém umožňuje vedoucímu skladu automaticky odstranit ze skladu nenamapovaná skladová umístění. Jakmile uživatel provede volbu tohoto typu synchronizace, systém odstraní ze skladu všechna umístění, která nejsou evidována v mapě skladu tak, aby každé umístění ve skladě mělo odpovídající umístění v mapě.

UC6 Synchronizovat – vytvořit chybějící skladová umístění

Vedoucímu umožňuje systém automaticky vytvořit ve skladě chybějící umístění z mapy. Jakmile provede volbu uvedeného typu synchronizace, systém vytvoří ve skladě umístění, která se nachází v mapě, ale ve skladě nemají evidována odpovídající skladová umístění. Každé umístění v mapě tak bude po dokončení synchronizace mít odpovídající umístění ve skladě.

1.5.2.2 Tvorba objektů a zón

V této části se věnuji případům užití spojených s tvorbou nových objektů a zón v mapě skladu. Diagram těchto případů užití je patrný na obrázku 1.6.



Obrázek 1.6: Diagram případů užití – Tvorba objektů a zón

UC7 Přidat nový regál

Vedoucí skladu může do mapy skladu přidat nový regál. Po stisknutí příslušného tlačítka systém vyzve uživatele, aby v mapě vybral pozici, na kterou nový regál umístí. Na pozici, kterou uživatel zvolí, se nesmí nacházet jiný objekt. Jakmile je vybrána vhodná pozice, zobrazí systém formulář, ve kterém uživatel zadá název vytvářeného regálu. Po potvrzení formuláře vytvoří systém v mapě nový regál.

UC8 Přidat novou skupinu umístění

Uživatel s rolí vedoucího může v mapě zaznamenat novou skupinu umístění. Nejdříve stiskne v editoru mapy tlačítko na vytvoření nové skupiny umístění. Následně uživatel vybere pozici v mapě, na které se nenachází žádný objekt a na kterou si přeje novou skupinu umístit. Následně systém zobrazí formulář, ve kterém může uživatel vložit do skupiny nová umístění. Po potvrzení formuláře systém vytvoří na vybrané pozici skupinu umístění.

UC9 Přidat novou překážku

Systém umožňuje vedoucímu přidat do mapy skladu novou překážku. Vedoucí nejdříve zvolí v editoru mapy tlačítko pro přidání překážky. Uživatel

vybere v mapě místo, na kterém se nenachází jiný objekt a na které chce překážku přidat. Poté systém zobrazí formulář pro specifikaci názvu překážky a po zadání jména vytvoří na specifikované pozici novou překážku.

UC10 Přidat novou zónu

Vedoucí skladu může v mapě vyznačit oblast (tzv. zónu), která je důležitá pro chod skladu. Po stisknutí příslušného tlačítka systém vyzve uživatele, aby v mapě vybral pozici, na které chce novou zónu vytvořit. Na této pozici se nesmí nacházet jiná zóna. Povoleny jsou naopak pozice, které jsou v mapě volné, nebo na kterých se nachází pouze objekt. Jakmile je vybrána vhodná pozice, zobrazí systém formulář, ve kterém uživatel zadá název vytvářené zóny a případně vybere jeden či více typů zón. Po potvrzení formuláře systém vytvoří na vybrané pozici danou zónu.

Typy zón mohou být následující:

- Příjem
- Výdej
- Balicí místo
- Parkování retraků[2]

1.5.2.3 Úprava mapy, objektů a zón

V následující části se věnuji případům užití spojených s úpravou mapy, zón a objektů (tedy regálů, skupin umístění nebo překážek), které jsou v ní zaznamenány. Diagram těchto případů užití je patrný na obrázku 1.7.

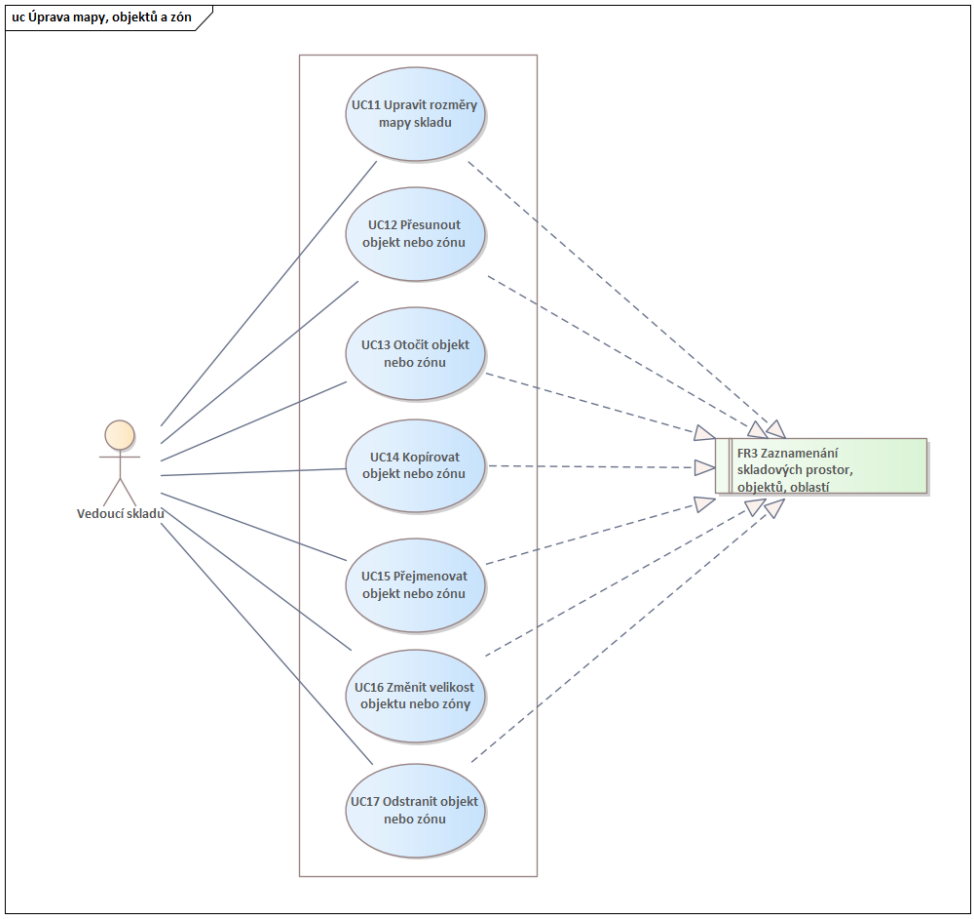
UC11 Upravit rozměry mapy skladu

Vedoucí může upravit rozměry mapy skladu. Uživatel s rolí vedoucí skladu změní v editoru mapy šířku nebo výšku mapy v příslušném formuláři. Systém provede kontrolu, zda bylo zadáno kladné celé číslo. V případě, že uživatel snižuje hodnotu rozměru, systém také zkotroluje, zda se v oblasti mapy, která bude úpravou rozměru odstraněna, nenachází žádné objekty či oblasti. V případě, že jsou splněny předchozí podmínky, systém upraví velikost mapy.

UC12 Přesunout objekt nebo zónu

Systém umožňuje uživateli s rolí vedoucího skladu přesunout v mapě objekt (regál, skupinu umístění, překážku) nebo zónu na jinou pozici. Uživatel vybere v mapě objekt nebo zónu a zvolí její přesun. Systém vyzve uživatele

[2]Retrak je typ vysokozdvyžného vozíku užívaného ve skladech



Obrázek 1.7: Diagram případů užití – Úprava mapy, objektů a zón

k výběru nové pozice objektu nebo zóny v mapě. Jakmile uživatel vybere pozici, na kterou chce objekt nebo zónu přemístit, systém zkontroluje, zda umístění objektu nezpůsobí kolizi s jiným objektem, případně zda se dvě zóny nebudou překrývat. Pokud jsou splněny předchozí podmínky, systém přesune v mapě objekt nebo zónu na novou pozici.

UC13 Otočit regál, překážku nebo zónu

V systému může vedoucí skladu otočit regál, překážku nebo zónu. Uživatel vybere v mapě, který regál, překážku či zónu chce otočit. Po stisknutí příslušného tlačítka systém otočí vybraný objekt či zónu o 90°.

UC14 Kopírovat regál, překážku nebo zónu

Vedoucí může v mapě zkopírovat již existující regál, překážku nebo zónu. Nejdříve vybere příslušný regál, překážku nebo zónu a zvolí její kopírování. Následně vyzve systém uživatele, aby vybral pozici objektu nebo zóny vytvořené zkopírováním. Jakmile uživatel vybere pozici, na kterou chce kopii umístit, systém zkontroluje, zda nový objekt nezpůsobí kolizi s jiným objektem, případně zda se dvě zóny nebudou překrývat. Pokud jsou splněny předchozí podmínky, systém vytvoří na vybrané pozici kopii objektu nebo zóny. Je-li kopírován regál, je změněn jeho název a název umístění v něm tak, aby nebyla v mapě vytvořena duplicitní umístění.

UC15 Přejmenovat regál, překážku nebo zónu

Uživatel s rolí vedoucího skladu může změnit název regálu, překážky nebo zóny. Nejdříve uživatel vybere regál, překážku nebo zónu a vybere možnost přejmenování objektu. Systém zobrazí formulář pro změnu názvu a uživatel v něm název upraví a formulář potvrdí. Systém následně změní název vybraného regálu, překážky nebo zóny.

UC16 Změnit velikost regálu, překážky nebo zóny

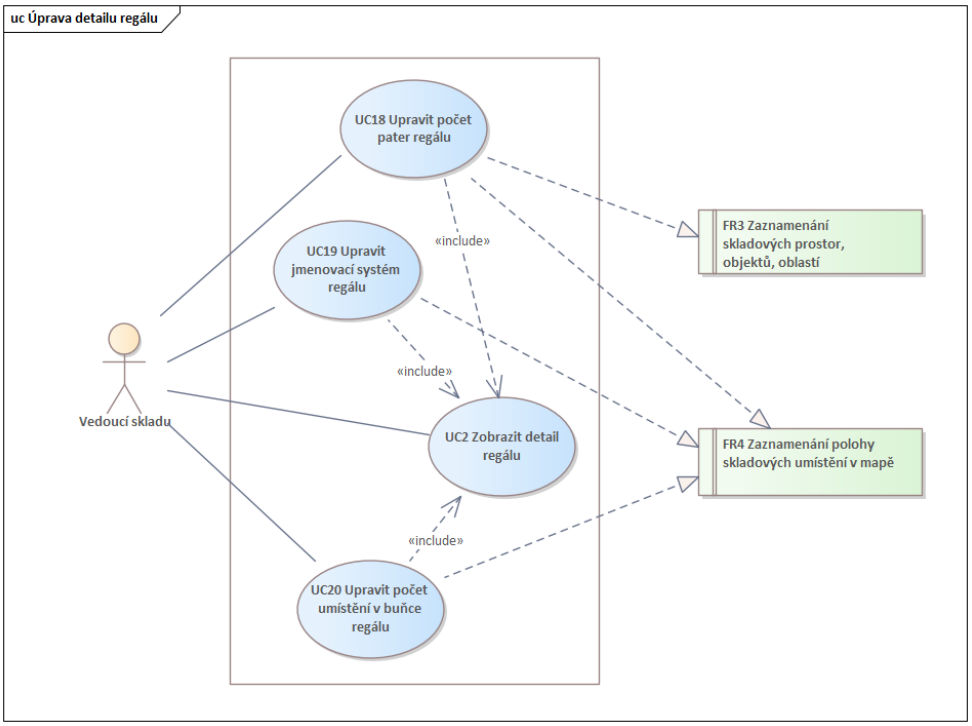
Systém umožňuje vedoucímu změnit délku regálu a výšku a šířku překážky nebo zóny. Vedoucí vybere regál, překážku nebo zónu, u které chce změnit rozměry. Systém zobrazí formulář, ve kterém zadá vedoucí nové hodnoty rozměrů. V případě regálu může vedoucí měnit hodnotu jeho délky. Délka regálu reprezentuje výšku regálu, je-li regál ve vertikální poloze a šířku regálu, je-li regál do mapy zanesen vertikálně. V případě překážky a zóny umožňuje systém zadání nové šířky a výšky. Systém vyhodnotí, zda nově zadané hodnoty nezpůsobí kolizi dvou objektů, případně zón a pokud je vše v pořádku, změní rozměry objektu nebo zóny.

UC17 Odstranit objekt nebo zónu

Vedoucí skladu může v systému odstranit objekt nebo zónu. Nejdříve vybere objekt nebo zónu, kterou chce odstranit. Následně kliknutím na příslušné tlačítko vybere možnost odstranění objektu. Systém odstraní zvolený objekt nebo zónu z mapy.

1.5.2.4 Úprava detailu regálu

Tato část se věnuje případům užití spojených s úpravou dat, která jsou k dispozici v detailu regálu. Diagram těchto případů užití je patrný na obrázku 1.8.



Obrázek 1.8: Diagram případů užití – Úprava detailu regálu

UC18 Upravit počet pater regálu

Uživatel s rolí vedoucího skladu může upravit počet pater regálu. Nejdříve uživatel dle již uvedeného případu užití UC2 zobrazí detail regálu, u kterého chce změnit počet pater. Systém zobrazí formulář, ve kterém uživatel upraví počet pater, které daný regál obsahuje, a systém dle nové hodnoty upraví regál a umístění, která obsahuje.

UC19 Upravit jmenovací systém regálu

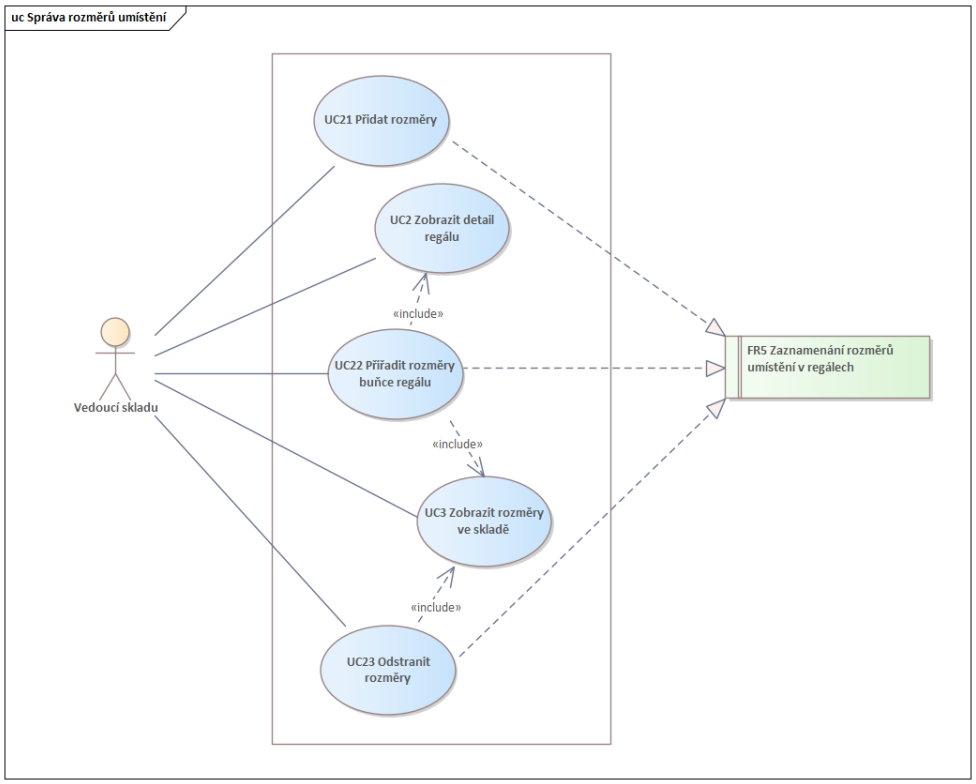
Systém umožňuje vedoucímu upravit jmenovací systém, podle kterého jsou automaticky vytvářeny názvy umístění v regálech. Dle scénáře UC2 zobrazí vedoucí detail regálu. Systém zobrazí formulář, ve kterém může uživatel upravovat počáteční symboly a směr pojmenovávání sloupců a pater a dále také oddělovače, které jsou v názvech použity. Po změně hodnot ve formuláři systém upraví v regále názvy umístění dle zadaného systému.

UC20 Upravit počet umístění v buňce regálu

Vedoucímu skladu je umožněno upravovat počet umístění, která se nachází v konkrétní buňce regálu. Nejdříve uživatel dle případu užití UC2 zobrazí detail regálu, u kterého chce provádět úpravu. Následně uživatel vybere v zobrazeném bočním pohledu na regál buňku, ve které chce změnit počet umístění. Systém zobrazí formulář, ve kterém uživatel může měnit hodnotu počtu umístění. Systém dle hodnoty, kterou uživatel zadal, ubere či přidá příslušná umístění do buňky regálu.

1.5.2.5 Správa rozměrů umístění

Následující část se věnuje případům užití spojených se správou rozměrů umístění, která se v mapě skladu nachází. Diagram těchto případů užití je zaznamenán na obrázku 1.9.



Obrázek 1.9: Diagram případů užití – Správa rozměrů umístění

UC21 Přidat rozměry

Uživatel s právy vedoucího skladu může do mapy přidat nové rozměry umístění ve skladě. Po stisknutí příslušného tlačítka systém zobrazí uživateli formulář, ve kterém vedoucí nastaví hodnoty výšky, šířky a hloubky umístění a barvu, jakou mají být označeny buňky, které obsahují umístění s daným rozměrem. Systém zkontroluje, zda jsou příslušné hodnoty validní a po potvrzení formuláře vytvoří příslušný rozměr.

UC22 Přiřadit rozměry buňce regálu

Vedoucí může přiřadit rozměry umístěním, která se nachází v buňce regálu. Dle již uvedených případů užití UC2 a UC3 zobrazí uživatel umístění ve skladě a detail regálu, ve kterém chce rozměry přiřadit. Poté uživatel vybere v zobrazeném bočním pohledu na regál buňku, ve které chce umístěním přiřadit jejich rozměry. Ze zobrazených rozměrů vybere rozměr, který chce daným umístěním přiřadit a stiskne příslušné tlačítko. Systém změní dle vybraných hodnot rozměr umístění v buňce.

UC23 Odstranit rozměry

Systém umožňuje vedoucímu skladu odstranit zaznamenané rozměry umístění z mapy skladu. Dle případu užití UC3 uživatel zobrazí rozměry ve skladě. Následně vybere rozměr, který chce ostranit, a stiskne příslušné tlačítko. Systém odstraní z mapy vybraný rozměr.

1.5.2.6 Úprava umístění ve skupině, úprava typů zón

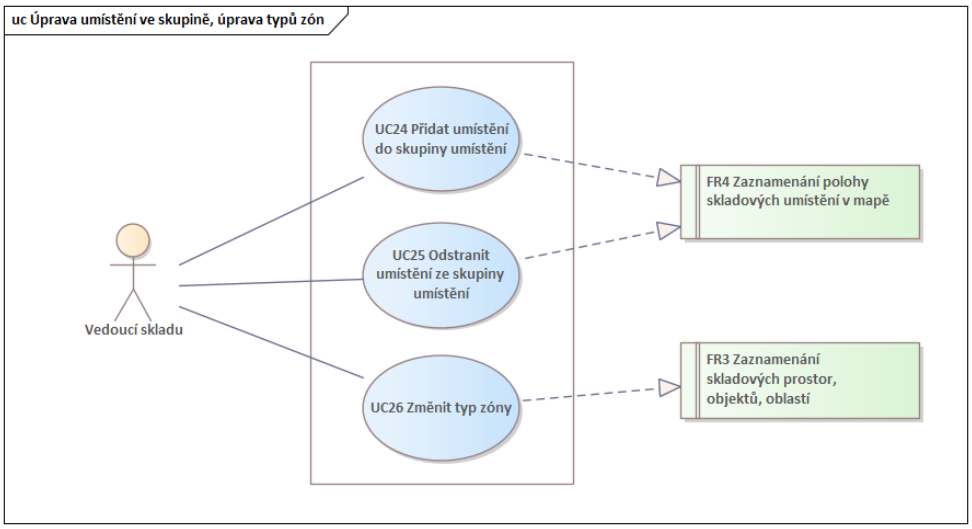
V poslední části se věnuji případům užití spojených s úpravou umístění ve skupině umístění a úpravou typů zón. Diagram těchto případů užití je patrný na obrázku 1.10.

UC24 Přidat umístění do skupiny umístění

Uživatel může do skupiny umístění přidat nové umístění. V mapě uživatel vybere skupinu umístění, do které chce přidat nové umístění. Systém zobrazí formulář, ve kterém uživatel vyplní název nového umístění a formulář potvrdí. Systém přidá do skupiny umístění nové umístění.

UC25 Odstranit umístění ze skupiny umístění

Vedoucí skladu má možnost odstranit umístění ze skupiny umístění. Nejdříve provede v mapě výběr skupiny, ze které chce umístění odstranit. Systém zobrazí formulář, ve kterém vedoucí vybere umístění, které chce ze skupiny odstranit. Po stisknutí příslušného tlačítka odstraní systém umístění ze skupiny umístění.



Obrázek 1.10: Diagram případů užití – Úprava umístění ve skupině, úprava zón

UC26 Změnit typ zóny

Systém umožňuje vedoucímu změnit typ zóny v mapě. Vedoucí provede výběr zóny, u které chce změnit její typ. Systém zobrazí formulář, ve kterém uživatel změní typ dané zóny. Systém následně zaznamená změnu v mapě.

1.5.3 Pokrytí funkcionality definované případy užití

Současné řešení plně pokrývá funkcionalitu definovanou následujícími případy užití:

- UC1 Zobrazit mapu skladu
- UC2 Zobrazit detail regálu
- UC3 Zobrazit rozměry ve skladě
- UC7 Přidat nový regál
- UC11 Upravit rozměry mapy skladu
- UC18 Upravit počet pater regálu
- UC20 Upravit počet umístění v buňce regálu
- UC21 Přidat rozměry
- UC22 Přiřadit rozměry buňce regálu

- UC23 Odstranit rozměry

Funkcionalitu vyplývající z následujících případů užití současné řešení pokrývá částečně:

- UC12 Přesunout objekt nebo zónu
- UC13 Otočit regál, překážku nebo zónu
- UC14 Kopírovat regál, překážku nebo zónu
- UC15 Přejmenovat regál, překážku nebo zónu
- UC16 Změnit velikost regálu, překážky nebo zóny
- UC17 Odstranit objekt nebo zónu
- UC19 Upravit jmenovací systém regálu

Funkcionalita definovaná následujícími případy užití není v současném řešení pokryta:

- UC4 Uložit mapu skladu
- UC5 Synchronizovat – odstranit nenamapovaná skladová umístění
- UC6 Synchronizovat – vytvořit chybějící skladová umístění
- UC8 Přidat novou skupinu umístění
- UC9 Přidat novou překážku
- UC10 Přidat novou zónu
- UC24 Přidat umístění do skupiny umístění
- UC25 Odstranit umístění ze skupiny umístění
- UC26 Změnit typ zóny

Kapitola 2

Návrh

V této kapitole nejdříve navrhnu vhodné kroky, které povedou k dokončení projektu mapy skladu. Dále zvolím metodiku, kterou se při vývoji budu řídit a rozvrhnu a definuji iterace, ve kterých budu postupovat. Také přiblížím technologie využité při vývoji tohoto projektu a nakonec popíši části návrhu, které jsem provedl v jednotlivých iteracích.

2.1 Kroky vedoucí k dokončení projektu

Na základě předešlé analýzy jsem identifikoval a navrhnul následující kroky, které je nutné realizovat k dokončení projektu.

2.1.1 Přidání nových nástrojů pro tvorbu a úpravu skupin umístění, překážek a zón

V tomto kroku budou v editoru mapy přidány nové nástroje, které umožní tvorbu a úpravu skupin umístění, překážek a zón. Jedná se o vývoj funkcionality na klientské straně systému – v aplikaci Swordfish. V tomto kroku bude pokryta funkcionalita vyplývající z následujících případů užití:

- UC8 Přidat novou skupinu umístění
- UC9 Přidat novou překážku
- UC10 Přidat novou zónu
- UC24 Přidat umístění do skupiny umístění
- UC25 Odstranit umístění ze skupiny umístění
- UC26 Změnit typ zóny

2.1.2 Úprava stávajících nástrojů pro úpravu objektů a oblastí

Jedná se o úpravu nástrojů, které jsou již v editoru mapy implementovány tak, aby bylo možné patřičně upravovat a odstraňovat všechny objekty, které může mapa skladu zaznamenávat. Opět se jedná o úpravu funkcionalit implementovaných na klientské straně systému. Po dokončení úprav bude v editoru mapy dostupná funkcionalita vyplývající z těchto případů užití:

- UC12 Přesunout objekt nebo zónu
- UC13 Otočit regál, překážku nebo zónu
- UC14 Kopírovat regál, překážku nebo zónu
- UC15 Přejmenovat regál, překážku nebo zónu
- UC16 Změnit velikost regálu, překážky nebo zóny
- UC17 Odstranit objekt nebo zónu
- UC19 Upravit jmenovací systém regálu

2.1.3 Uložení, správa a přístup k datům mapy

V tomto kroku bude realizováno ukládání a práce s daty mapy skladu. Ve skladovém systému jsou data o skladech spravována na serverové části aplikací Octopus, která poskytuje přístup k těmto datům pomocí REST API. To je konzumováno na klientské části aplikací Swordfish, ve které je také aktuálně implementován editor mapy skladu. Je tedy logické implementovat evidenci dat mapy skladu stejným způsobem. Tento krok zahrnuje definici API endpointů, pomocí kterých bude realizováno ukládání a načítání mapy, implementaci funkcionality pro správu dat na serverové části a implementaci ukládání a načítání dat mapy skladu na klientské části. Po dokončení tohoto kroku bude pokryta funkcionalita případu užití UC4, který definuje proces uložení mapy.

2.1.4 Mapování a synchronizace umístění

Při ukládání mapy je třeba zajistit, aby se umístění v mapě namapovala na odpovídající skladová umístění, která jsou v systému evidována. Dále je také třeba přidat možnost automatické synchronizace umístění v mapě s umístěními ve skladě. V tomto kroku bude implementována uvedená funkcionalita, která odráží případy užití UC5 a UC6 zaměřující se na synchronizaci umístění.

2.2 Volba metodiky vývoje

Metodika vývoje software je souhrn principů, procesů, praktik, rolí, technik, nástrojů a produktů používaných v rámci celého životního cyklu budování informačního systému, tedy při vývoji, údržbě a provozu. [13]

Metodiky nejčastěji rozdělujeme na tradiční a agilní. Tradiční metodiky se vyznačují především fixním rozsahem, jsou méně pružné a kladou větší důraz na analýzu a návrh projektu. Díky tomu je poté samotná implementace jednodušší a rychlejší. Pokud však zákazník v průběhu realizace změní své požadavky na výsledný produkt, je velmi složité se mu přizpůsobit. Agilní metodiky se naopak zaměřují na samotnou tvorbu produktu a lépe se přizpůsobují změnám požadavků. Cílem je zákazníkovi dodávat postupně fungující části software a získávat jeho zpětnou vatbu. Na jejím základě jsou poté prováděny úpravy a vyvíjí se další části software. [14] [15]

Část projektu je však již zpracována, jsou jasně definovány požadavky na výsledný stav systému a při vývoji se lze opřít o široce zpracovaný návrh a analýzu, která byla provedena v rámci předmětů BI-SP1 a BI-SP2, práce Bc. Kopeckého a kapitoly 1 této bakalářské práce. Z tohoto důvodu by bylo výhodné využít při vývoji jednu z tradičních metodik. Z předchozí podkapitoly 2.1, ve které jsem definoval kroky nutné k realizaci tohoto projektu však plyne, že bude možné systém vyvíjet po částech a ty na sebe potom napojovat tak, aby na konci vznikl funkční celek. Z těchto důvodů by bylo výhodné využít jednu z agilních metodik a vyvíjet tento projekt iterativně.

Rozhodl jsem se, že zvolím inkrementální metodiku vývoje, která v sobě spojuje výhody tradičních a agilních metodik. Tato metodika staví na tradiční metodice zvané Vodopád, ve které je vývoj rozdělen na fáze analýzy, návrhu, implementace, testování a nasazení software. Tyto fáze se provádí sekvenčně v daném pořadí pro celý projekt a není možné se v nich vracet. Inkrementální metodika však rozděluje projekt na menší části, které jsou následně v postupných iteracích vyvíjeny metodikou Vodopád. Jedná se tedy o sérii „mini vodopádů“. [15] [16]

2.3 Postup řešení

Ve vývoji projektu jsem se rozhodl nejprve postupovat ve čtyřech iteracích. V nich jsem naplánoval postupnou realizaci kroků, které jsem definoval v první části této kapitoly (2.1). Nakonec však bylo nutné z důvodů, které popisuji níže přidat dvě další iterace.

V první iteraci jsem se věnoval dopracování nástrojů v editoru mapy skladu. Upravil jsem stávající nástroje pro tvorbu a práci s regály a přidal jsem nové nástroje pro práci se skupinami umístění, překážkami a zónami.

V druhé iteraci jsem nejdříve navrhl API endpointy, pomocí kterých ukládá klientská část aplikace mapová data a následně vytvořil backend mapy skladu.

Ve třetí iteraci došlo k propojení stávajícího frontendu mapy skladu s backendem vyvinutým v druhé iteraci. Ve frontendu mapy jsem zajistil ukládání mapy a její načítání z backendu.

Ve čtvrté iteraci jsem zajistil, aby byla umístění, která uživatel v editoru mapy vytvoří, mapována na skladová umístění, která eviduje systém Atlantis. Dále jsem do frontendu přidal přehledy mapování umístění. V rámci této iterace jsem také v mapě implementoval funkcionalitu synchronizace umístění se skladem.

Po skončení uvedených iterací se ukázalo, že bude nutné přidat ještě dvě nové iterace, se kterými jsem na začátku tohoto projektu nepočítal. V bakalářské práce jsem měl totiž také za úkol z dat skladového systému a mapy skladu sestavit heatmapu vytížení skladových pozic. Když jsem tuto problematiku řešil s vedoucím práce, poukázal nato, že existuje lepší způsob řešení daného problému, než je manuální tvorba heatmapy pro jeden konkrétní sklad. Tím byla implementace funkcionality automatického generování heatmapy do skladového systému. Uživatelé by tak měli možnost pro každý sklad zobrazit vytíženost jeho skladových pozic. Rozhodl jsem se proto pro realizaci tohoto řešení, což bylo cílem páté iterace.

V závěru vývoje jsem mimo jiné provedl také testování, které mělo za cíl ověřit, zda editor dosahuje potřebné použitelnosti. V průběhu tohoto testování jsem narazil na několik problémů, které bylo nutné odstranit, aby bylo možné editor mapy plně využít. Pracovníci společnosti Jagu s.r.o. provedli code review[3], v rámci kterého identifikovali chyby, kterých jsem se v implementaci dopustil. Uvedené problémy a chyby jsem se rozhodl vyřešit v šesté iteraci.

2.4 Použité technologie

V projektu se zabývám dokončením části skladového systému Atlantis – editoru mapy skladu. Pro zajištění kompatibility proto využiji stejných technologií, které jsou v systému využity. Díky tomu také poměrně snadno realizuji požadavky IR1 a SR1 zabývající se využitím technologií a návazností dalších rozšíření na tento projekt.

Frontend skladového systém Sworfish a současný editor mapy skladu jsou napsány v jazyce JavaScript ve frameworku Vue.js. Jsou zde využity především grafická knihovna Vuetify a knihovna Vuex, která zajišťuje správu stavu aplikace.

Backend skladového systému je implementován v jazyce PHP a využívá frameworku Symfony a knihovny Doctrine, která zajišťuje práci s daty v databázi a jejich objektově relační mapování. Data jsou ukládána do databáze PostgreSQL.

[3]Code review, neboli posouzení kódu – proces systematického zkoumání zrojového kódu, jehož cílem je najít chyby, které jeho autor přehlédl [17]

2.5 Nástroje editoru mapy skladu

V této části se budu zabývat návrhem úprav frontendu editoru mapy skladu, který jsem provedl v rámci první iterace. Před tím je však potřebné si pro lepší orientaci a rozlišení mezi umístěními, která jsou zanesena do mapy a umístěními, která eviduje ve skladu systém Atlantis, zavést následující pojmy:

Skladové umístění je reprezentace reálného umístění ve skladu, které je aktuálně evidováno v systému Atlantis.

Mapové umístění je entita v mapě, která slouží k přiřazení souřadnic v mapě skladovému umístění.

2.5.1 Úprava nástrojů editace regálu

Do možností, které nabízí panel nástrojů při výběru regálu (obr. 1.2), bude přidána možnost smazání objektu. Tlačítko, které bude tuto funkci realizovat, bude stejného stylu jako ostatní tlačítka, která se v panelu nachází. Je však nutné odlišit, že oproti všem ostatním tlačítkům nerealizuje funkci, která objekt edituje, ale maže. Proto bude mít tlačítko červenou barvu. Po kliknutí na tlačítko bude vybraný objekt z mapy smazán.

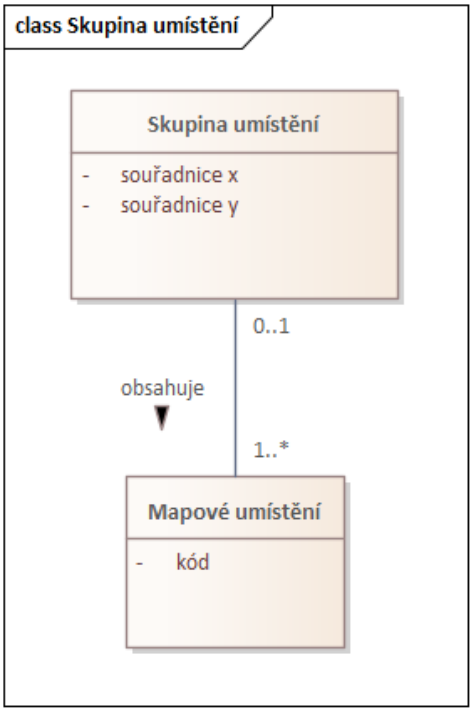
V kartě detailu regálu (obr. 1.4) bude přidána do záložky obecné, která se nachází v postranním panelu, možnost editace oddělovačů – symbolů, které v kódu umístění oddělují název regálu od sloupce, patra a pozice umístění v buňce regálu. To bude realizováno pomocí tří textových polí, která budou tyto oddělovače nastavovat. Při vytvoření regálu budou hodnoty oddělovačů nastaveny na výchozí hodnotu – znak, který byl v editoru užíván doposud – pomlčka.

2.5.2 Nástroje práce s umístěními mimo regál

Umístění se v současném editoru vždy nachází na jednom ze čtverečků tvořících mapu – sloupec regálu odpovídá jednomu čtverečku. Na jednom čtverečku se tedy může nacházet více umístění, tedy všechna umístění, která se nachází v daném sloupci.

Umístění, která se nenachází v regálu, je tedy vhodné realizovat v mapě stejným přístupem. Umístění bude mít vždy dáno svou polohu jedním čtverečkem mapy, na kterém leží. V jednom čtverečku se bude moci nacházet více umístění. Bude se tedy jednat o skupinu umístění. Třída reprezentující tuto skupinu je uvedena v diagramu na obrázku 2.1.

Do panelu nástrojů bude přidáno tlačítko pro přidání nové skupiny umístění. Jeho design bude stejný jako design tlačítka pro nový regál. Po jeho stisknutí bude uživatel vyzván k vybrání čtverečku, na kterém chce skupinu umístění vytvořit. Po kliknutí na tento čtvereček se uživateli zobrazí dialog, ve kterém bude moci přidat do skupiny jednotlivá mapová umístění. Po přidání umístění



Obrázek 2.1: Doménový model – Skupina umístění

do vytvářené skupiny uživatel potvrdí dialog a systém do mapy zaeviduje danou skupinu umístění.

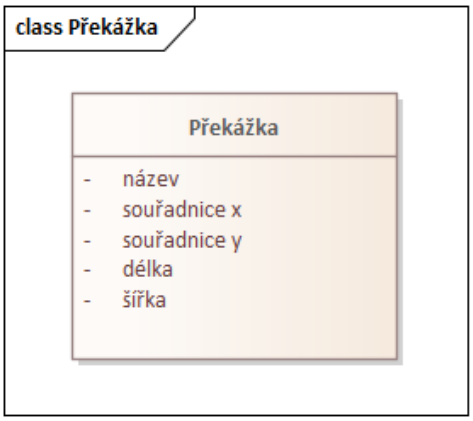
Skupinu umístění bude mapa skladu zobrazovat jako čtvereček se symbolem reprezentujícím umístění. Barva tohoto pole bude stejná jako barva regálu (tmavě zelená) tak, aby byla jasně označena místa v mapě, na kterých se nachází mapová umístění. Klikne-li uživatel na políčko v mapě reprezentující skupinu umístění, políčko změní svou barvu na fialovou a indikuje tak uživateli, že došlo k výběru dané skupiny.

Je-li vybrána skupina umístění, zobrazí se, podobně jako při výběru regálu (obr. 1.2), v postranním panelu karta, která bude umožňovat editaci vlastností skupiny. V dané kartě bude možné přidávat a odstraňovat mapová umístění, která skupina obsahuje.

Pro vybranou skupinu umístění se v panelu nástrojů zobrazí tlačítka pro pohyb a smazaní objektu. Vzhledem k tomu, že skupina zabírá vždy pouze jeden čtereček, nezobrazí se tlačítko pro rotaci objektu. Dále v panelu nástrojů nebude zobrazena možnost kopírování objektu, která zde není žádoucí. Skupina umístění totiž slouží k evidenci umístění, která nejsou v regálech a nemusí pro ně být definován jmenovací systém. Proto nejsme schopni kvalifikovaně měnit kódy umístění při kopírování skupiny a vytvořením skupiny se stejnými kódy bychom v mapě vytvořili nežádoucí duplicity.

2.5.3 Nástroje práce s překážkami

Překážka bude objekt v mapě překrývající obdélníkový či čtvercový prostor několika čtverečků o určité šířce a délce. Jeho poloha bude určena souřadnicemi levého horního čtverečku. Bude se jednat o neprostupnou oblast, které bude možné přiřadit název tak, aby se dalo v mapě lépe orientovat. Třída, reprezentující překážku, je zobrazena na obrázku 2.2.



Obrázek 2.2: Doménový model – Překážka

Nově se bude v panelu nástrojů nacházet tlačítko pro přidání překážky. Jeho design bude stejný jako design tlačítka pro nový regál a skupinu umístění. Po jeho stisknutí bude uživatel vyzván k vybrání čtverečku, na kterém chce překážku vytvořit. Po kliknutí na tento čtvereček systém do mapy zanese danou překážku o jednotkové šířce a délce. Stejně tak jako u regálu bude možné tyto rozměry při výběru překážky upravovat.

Překážka bude v mapě obdobně jako regál zobrazena jako souvislý objekt, uprostřed kterého se bude nacházet symbol pro blokovaný prostor. Tento objekt bude mít tmavě šedou barvu – stejný odstín, jako pozadí prostoru, na kterém je vykreslena mapa skladu. Klikne-li uživatel na tento objekt, změní se jeho barva na fialovou a indikuje tak uživateli, že došlo k výběru překážky.

Je-li vybrána překážka, zobrazí se v postranním panelu karta umožňující editaci jejích vlastností. V dané kartě bude možné měnit název překážky, rozměry – délku, šířku překážky a překážku z mapy odstranit.

Pro vybranou překážku se v panelu nástrojů zobrazí všechny možnosti práce s objektem, tedy pohyb, rotace, kopírování a mazání překážky. Kopírování vytvoří na uživatelem zvolené pozici kopii vybrané překážky se stejnými vlastnostmi, tedy délkou, šířkou i názvem.

Při tvorbě, rotaci, přesunu a kopírování překážky bude možné tyto úkony vykonávat pouze tak, aby nedošlo ke kolizi s ostatními objekty v systému.

2.5.4 Nástroje práce se zónami

Zóna bude obdélníkový či čtvercový prostor několika čtverečků o určité šířce a délce. Na rozdíl od regálu, skupiny umístění a překážky se nejedná o objekt, ale o oblast. Objekty se v mapě nemohou mezi sebou překrývat. Oblast však může objekt částečně či úplně překrýt. Objekt, nebo jeho část se pak nachází v dané zóně. U každé zóny rozlišujeme její typ. Konkrétní zóna může mít více typů, a nebo také nemusí mít přiřazen typ žádný. Aktuálně budou definovány čtyři typy zón, které jsou patrné v seznamu níže. V budoucnu může dojít k identifikaci více typů zón.

Příjem: Jedná se o zónu, ve které dochází k příjmu zboží do skladu.

Výdej: Jedná se o zónu, ve které dochází k výdeji zboží ze skladu.

Balení: Jde o typ zóny, kde se produkty balí pro potřeby další expedice.

Parkování retraků: Oblast, ve které parkují retraky a jiná ve skladě využívaná technika.

Pro lepší orientaci v mapě skladu bude možné zónu pojmenovat. Třídy reprezentující zónu a její typy jsou zobrazeny na obrázku 2.3.

Posledním tlačítkem, o které bude základní zobrazení panelu nástrojů rozšířeno, bude tlačítko pro přidání zóny. Jeho design bude stejný jako design tlačítka pro nový regál, skupinu umístění a překážku. Po jeho stisknutí bude uživatel vyzván k vybrání čtverečku, na kterém chce překážku vytvořit. Po kliknutí na něj se uživateli zobrazí dialog, ve kterém vybere typy vytvářené zóny. Po výběru uživatel potvrdí dialog a systém do mapy vytvoří na místě, na které uživatel kliknul, zónu o rozměrech jednoho čtverečku.

Zóna bude v mapě zobrazena tak, že políčka, která zakrývá, budou překryta žlutou průhlednou vrstvou tak, aby byly viditelné objekty, které se v zóně nachází. Klikne-li uživatel na čtvereček, na kterém se nachází zóna a na kterém se nenachází žádný jiný objekt, změní se její barva na fialovou a indikuje tak uživateli, že došlo k výběru zóny. Nachází-li se však na čtverečku, který pokrývá zóna, i jiný objekt a uživatel na něj klikne, zobrazí se na této pozici nabídka, ve které uživatel může zvolit, zda si přeje vybrat zónu nebo objekt, který se zde nachází.

Je-li vybrána zóna, zobrazí se v postranním panelu karta umožňující editaci jejích vlastností. V dané kartě bude možné měnit název zóny, její délku a šířku, typy, které jsou zóně přiřazeny, a také zónu smazat.

Pro vybranou zónu se v panelu nástrojů zobrazí všechny možnosti práce s objektem, tedy pohyb, rotace, kopírování a mazání zóny. Kopírování vytvoří na uživatelem zvolené pozici kopii vybrané zóny se stejnými vlastnostmi, tedy novou zónu se stejnou délkou, šířkou, názvem a typem.

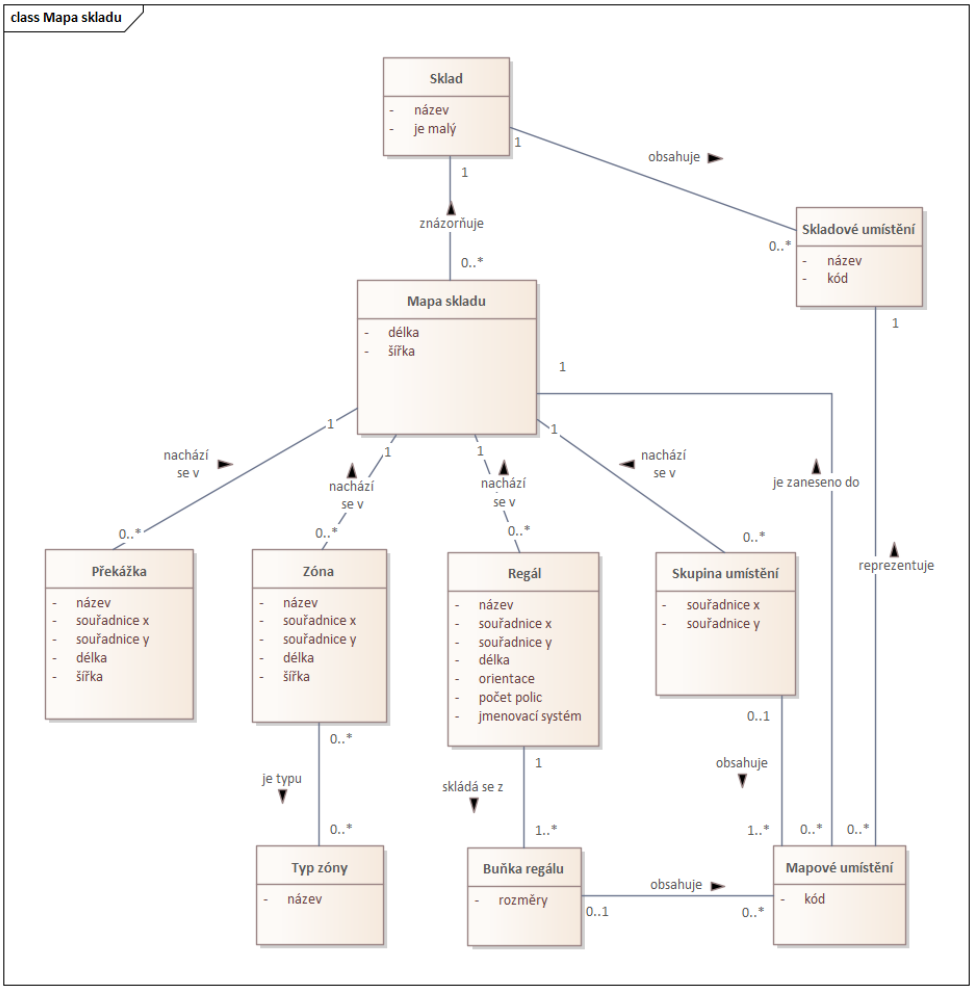
Obrázek 2.3: Doménový model – Zóna

2.6 Backend mapy skladu

V této kapitole se budu věnovat návrhu backendu mapy skladu, který jsem provedl v rámci druhé iterace tohoto projektu. Nejdříve vytvořím doménový model a popíši třídy, které se v této doméně nachází. Dále se budu věnovat návrhu datového modelu, ve kterém popíši entity, se kterými bude systém pracovat. V neposlední řadě navrhnu rozhraní, za pomoci kterého bude frontend s backendem komunikovat.

2.6.1 Doménový model

S využitím znalostí z provedené analýzy a předchozí iterace jsem nejdříve identifikoval třídy týkající se domény mapy skladu. Ty znázorňuje diagram doménového modelu na obrázku 2.4. Dále následuje popis těchto tříd a jejich atributů.



Obrázek 2.4: Doménový model mapy skladu

Sklad

Sklad, který je znázorněn jednou či více mapami a obsahuje skladová umístění.

Název atributu	**Popis**
název	Název skladu
je malý	Vyjadřuje, zda se jedná o malý nebo velký sklad

Tabulka 2.1: Popis atributů třídy Sklad

Skladové umístění

Umístění nacházející se ve skladu. Může být reprezentováno mapovým umístěním.

Název atributu	**Popis**
název	Název skladového umístění
kód	Kód skladového umístění

Tabulka 2.2: Popis atributů třídy Skladové umístění

Mapa skladu

Mapa znázorňující sklad. Mohou se v ní nacházet překážky, zóny, regály a skupiny umístění. Mohou v ní být zanesena mapová umístění.

Název atributu	**Popis**
délka	Velikost skladu zakresleného v mapě v horizontálním směru
šířka	Velikost skladu zakresleného v mapě ve vertikálním směru

Tabulka 2.3: Popis atributů třídy Mapa skladu

Překážka

Překážka nacházející se ve skladě.

Název atributu	**Popis**
název	Název překážky
souřadnice x	Horizontální souřadnice levého horního rohu překážky v mapě
souřadnice y	Vertikální souřadnice levého horního rohu překážky v mapě
délka	Velikost překážky v horizontálním směru
šířka	Velikost překážky ve vertikálním směru

Tabulka 2.4: Popis atributů třídy Překážka

Zóna

Zóna nacházející se ve skladě. Zóna může být několika typů.

Název atributu	**Popis**
název	Název zóny
souřadnice x	Horizontální souřadnice levého horního rohu zóny v mapě
souřadnice y	Vertikální souřadnice levého horního rohu zóny v mapě
délka	Velikost zóny v horizontálním směru
šířka	Velikost zóny ve vertikálním směru

Tabulka 2.5: Popis atributů třídy Zóna

Typ zóny

Typ zóny, který může být přiřazen několika zónám v mapě.

Název atributu	**Popis**
název	Název typu zóny

Tabulka 2.6: Popis atributů třídy Typ zóny

Regál

Regál nacházející se v mapě, skládající se z jedné nebo více buněk.

Název atributu	**Popis**
název	Název regálu
souřadnice x	Horizontální souřadnice levého horního rohu regálu v mapě
souřadnice y	Vertikální souřadnice levého horního rohu regálu v mapě
délka	Délka regálu
orientace	Vyjadřuje, zda je regál orientován horizontálně nebo vertikálně
počet polic	Počet polic, které se v regálu nachází
jmenovací systém	Definuje strategii, jakou jsou umístění v regálu pojmenovávána

Tabulka 2.7: Popis atributů třídy Regál

Buňka regálu

Buňka regálu obsahující mapová umístění. Z několika buněk se skládá regál.

Název atributu	**Popis**
rozměry	Rozměry umístění, která se nachází v buňce

Tabulka 2.8: Popis atributů třídy Buňka regálu

Skupina umístění

Skupina umístění nacházející se v mapě, obsahující jedno nebo více mapových umístění.

Název atributu	Popis
souřadnice x	Horizontální souřadnice skupiny umístění v mapě
souřadnice y	Vertikální souřadnice skupiny umístění v mapě

Tabulka 2.9: Popis atributů třídy Skupina umístění

Mapové umístění

Mapové umístění, které je zaneseno do mapy skladu a obsahuje ho buď buňka regálu, nebo skupina umístění. Reprezentuje skladové umístění.

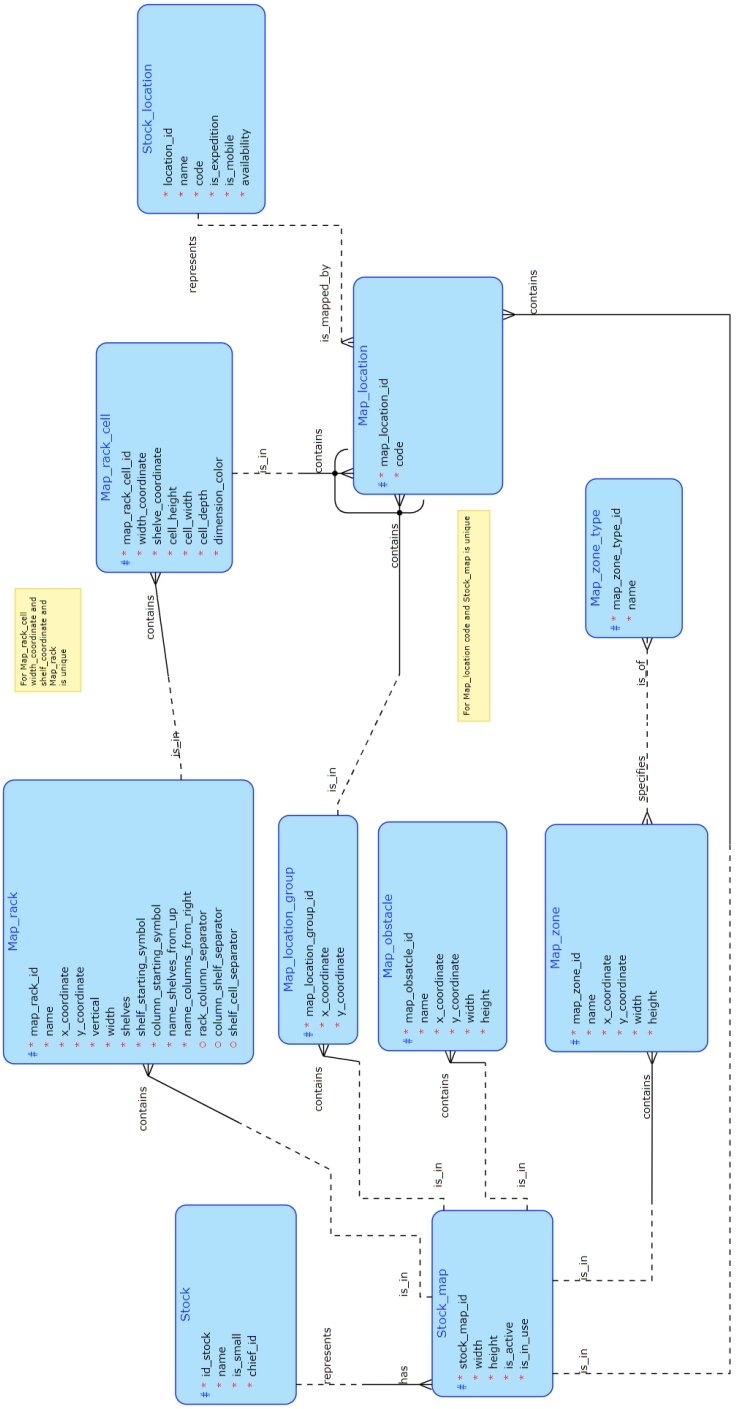
Název atributu	Popis
kód	Kód mapového umístění

Tabulka 2.10: Popis atributů třídy Mapové umístění

2.6.2 Datový model

Konceptuální model dat mapy skladu je zachycen na obrázku 2.5.

Tento model jsem na doporučení vedoucího bakalářské práce navrhl tak, aby se všechny nově vytvořené entity mapy skladu (v obrázku entita *Stock map* a entity začínající slovem *Map*) na systém pouze napojily a nebylo třeba upravovat v systému entity již implementované. Toho jsem docílil tvorbou entity *Map location*, která nejen umožňuje napojení na systém bez zásahu do entity *Stock Location*, ale zároveň také umožňuje ukládat mapu, která obsahuje mapová umístění, jež zatím nejsou namapována na skladová umístění v systému.



Obrázek 2.5: Konceptuální model mapy skladu

2.6.3 Specifikace API

Dále jsem za pomoci nástroje Swagger[4] definoval API[5], díky kterému spolu bude backend s frontendem mapy skladu komunikovat.

Definoval jsem následující endpointy[6]:

Prefix: stocks/{stockId}	
URI	
Metoda	**Popis**
/maps	
GET	Zobrazí všechny mapy v daném skladu
POST	Vytvoří mapu skladu
/maps/{mapId}	
GET	Zobrazí mapu skladu
DELETE	Odstraní mapu skladu
/maps/{mapId}/export	
GET	Zobrazí mapu skladu se všemi objekty, které se v ní nachází
/maps/{mapId}/import	
PUT	Aktualizuje mapu skladu i s objekty, které se v ní nachází

Tabulka 2.11: Specifikace API mapy skladu – 1. verze

Podle tohoto návrhu bude pro sklad nejdříve vytvořena mapa skladu, která nebude obsahovat žádné objekty ani oblasti. Následně ji bude možné upravovat tak, že bude mapa obsahující objekty a oblasti ve formátu JSON předána endpointu *import*, který dle předaných dat provede aktualizaci mapy a všech příslušných entit. Díky endpointu *export* bude poté možné získat veškerá data mapy včetně všech dat příslušných objektů a oblastí.

Nejdříve bylo mým záměrem předávat endpointu *import* data o celé mapě skladu. Z diskuse s pracovníkem společnosti Jagu s.r.o., který se zabývá vývojem aplikace Octopus, však vyplynulo, že množství umístění a množství objektů, které bude obsahovat reálná mapa skladu, jsou příliž velká na to, aby bylo možné zasílat data o celé mapě skladu. Bylo mi doporučeno odesílat pouze

[4]nástroj od společnosti SmartBear umožňující návrh, dokumentaci, testování a nasazení API

[5]aplikační programové rozhraní – slouží k předávání dat mezi dvěma systémy dle předem určené specifikace[18]

[6]koncový bod API, na kterém rozhraní přijímá požadavky a odesílá odpovědi[18]

data, která se v mapě budou měnit. Objekty a oblasti poté zasílat s atributem, který určí, zda je třeba objekt v mapě vytvořit, upravit nebo smazat.

Při testování tohoto řešení se ale ukázalo, že počet objektů, které by mohly být při jednom uložení odeslány, je příliž velký na to, aby byly všechny odesílány najednou na jeden endpoint, který by obsluhoval úpravu mapy. Z hlediska návrhu by také bylo vhodnější, aby každý typ objektu či oblasti měl několik endpointů, přes které by bylo možné objekty zobrazovat, vytvářet, upravovat a mazat. Následně jsem tedy znovu navrhnul API tak, že pro každý typ objektu či oblasti v mapě jsem vytvořil vlastní sadu endpointů. Tento nový návrh API je vyobrazen v tabulce 2.12.

Prefix: stocks/{stockId}	
URI	
Metoda	**Popis**
/maps	
GET	Zobrazí všechny mapy v daném skladu
POST	Vytvoří mapu skladu
/maps/{mapId}	
GET	Zobrazí mapu skladu
PUT	Upraví vlastnosti mapy skladu
DELETE	Odstraní mapu skladu
/maps/{mapId}/location-groups	
GET	Zobrazí skupiny umístění v dané mapě
POST	Vytvoří skupinu umístění
/maps/{mapId}/location-groups/{locationGroupId}	
PUT	Upraví skupinu umístění
DELETE	Odstraní skupinu umístění
/maps/{mapId}/obstacles	
GET	Zobrazí překážky v dané mapě
POST	Vytvoří překážku
/maps/{mapId}/obstacles/{obstacleId}	
PUT	Upraví překážku
DELETE	Odstraní překážku

/maps/{mapId}/racks	
GET	Zobrazí regály v dané mapě
POST	Vytvoří regál
/maps/{mapId}/racks/{rackId}	
PUT	Upraví regál
DELETE	Odstraní regál
/maps/{mapId}/zones	
GET	Zobrazí zóny v dané mapě
POST	Vytvoří zónu
/maps/{mapId}/zones/{zoneId}	
PUT	Upraví zónu
DELETE	Odstraní zónu

Tabulka 2.12: Specifikace API mapy skladu – 2. verze

2.7 Ukládání a načítání mapy skladu

V této části se budu zabývat návrhem, který jsem provedl v rámci třetí iterace. Ten se týkal realizace ukládání a načítání mapy skladu ve frontendu. Jednalo se tedy o propojení editoru s backendovou částí, která byla vyvinuta v druhé iteraci.

V okně editoru mapy skladu se bude v levém horním rohu nacházet tlačítko pro uložení mapy. Toto tlačítko bude aktivní pouze v případě, že v mapě byly provedeny změny, které ještě nejsou evidovány v backendu. V případě, že se v mapě nachází dvě a více mapových umístění se stejným kódem, bude tlačítko zvýrazněno oranžovou barvou a mapu v tomto stavu nebude možné uložit.

2.7.1 Načítání mapy skladu

Při rozbalení karty *Mapa skladu* v okně detailu skladu dojde k načtení aktuálně užívané mapy skladu z backendu. V průběhu načítání není v okně editoru zobrazena žádná mapa a v jeho horní části je zobrazena lišta s probíhajícím indikátorem načítání. Pokud se na backendu pro daný sklad taková mapa nevyskytuje, vytvoří systém mapu novou, kterou následně zobrazí. Po načtení mapy tlačítko pro uložení mapy zešedne a nebude možné na něj kliknout. Aktivuje se poté až v případě, kdy uživatel provede v mapě změny.

2.7.2 Uložení mapy skladu

V případě, že uživatel klikne na tlačítko uložit mapu, v tlačítku se místo jeho popisku zobrazí rotující kolečko indikující, že mapa se ukládá. V daný moment provede systém aktualizaci mapy v backendu podle dat, která uživatel zadal do editoru a znovu načte mapu skladu. Vzhledem k tomu, že jsem se nakonec v minulé iteraci rozhodl data na backend posílat postupně, bude se také v tlačítku během ukládání zobrazovat v procentech poměr počtu vykonaných požadavků ku celkovému počtu požadavků na backend. Uživateli tak bude zdůrazněno, že proces ukládání probíhá a je třeba pouze vyčkat na jeho dokončení. Jakmile se uložení dokončí, tlačítko zešedne a nebude aktivní.

2.8 Mapování a synchromizace umístění

V této části se budu zabývat návrhem, který jsem provedl ve čtvrté iteraci. Jednalo se o návrh funkcionality mapování a synchronizace umístění.

2.8.1 Mapování umístění v backendu

Entita *Map location* vytvořená v backendu systému v předchozích iteracích obsahuje nepovinnou vazbu na entitu *Stock location*, která v systému reprezentuje skladové umístění. Díky tomu je možné v systému ukládat evidenci mapování umístění. Odpovídající instance mapového umístění bude mít vazbu na konkrétní instanci skladového umístění, které reprezentuje. Nebude-li mapové umístění odpovídat svým kódem žádnému skladovému umístění, nebude tato vazba realizována.

2.8.2 Mapování umístění ve frontendu

Editor mapy skladu si bude udržovat seznam kódů a identifikátorů všech skladových umístění, která jsou v daném skladě evidována. Při načtení mapy skladu z backendu se k nim přiřadí mapová umístění se stejným kódem. Následně při každé úpravě mapy, zahrnující změnu názvu, přidání či odebrání mapového umístění, systém nalezne v seznamu potřebnou vazbu a provede její úpravu dle nově zadaných dat. Zároveň také systém nastaví v datech mapy u mapového umístění atribut obsahující identifikátor mapovaného skladového umístění na novou hodnotu. Dojde-li pak k uložení mapy skladu, tato data o mapování se odešlou backendu, kde dojde k jejich uložení.

2.8.3 Zobrazení stavu mapování uživateli

Přehled skladových a mapových umístění a jejich mapování bude uživateli zobrazen pod editorem mapy. Bude se jednat o několik záložek, mezi kterými bude uživatel moci přepínat.

Výčet záložek:

Nenamapovaná umístění: Obsahuje kódy skladových umístění, která jsou evidována pro daný sklad, ale ještě nebyla v mapě zaznamenána.

Namapovaná umístění: Jedná se o výčet kódů mapových umístění, která svými kódy odpovídají skladovým umístěním evidovaným pro daný sklad a jsou tedy mapována.

K vytvoření (neexistující): V této záložce se nachází kódy mapových umístění, ke kterým nejsou ve skladu evidována skladová umístění se stejným kódem.

Duplicitní umístění (vícekrát v mapě): Obsahuje kódy mapových umístění, která jsou v mapě zaznamenána více než jednou.

Umístění budou v jednotlivých záložkách zobrazena pomocí tzv. chipů tak, jak je jejich zobrazení realizováno při tvorbě a editaci skupiny umístění v mapě. Vzhledem k tomu, že počet umístění ve skladu a tedy i v mapě se může řádově pohybovat v tisících, budou chipy s kódy v záložkách stránkovány.

2.8.4 Synchronizace umístění

Vedle tlačítka pro uložení mapy se bude v editoru nacházet tlačítko pro synchronizaci umístění. Toto tlačítko bude zobrazeno pouze, pokud bude mapa skladu uložena a v systému se budou nacházet nějaká nenamapovaná umístění, případně se v mapě budou nacházet umístění, která ve skladě neexistují (k vytvoření).

Po stisknutí tohoto tlačítka se zobrazí dialogové okno, které bude zobrazovat možnosti synchronizace. U každé možnosti bude zobrazen popis akce, která bude v systému vykonána, počet umístění, která budou ovlivněna a tlačítko, které tuto akci spustí.

Možnosti pro synchronizaci umístění budou dvě. První možností bude automatická tvorba umístění ve skladě. Pro mapová umístění, která nemají namapovaná skladová umístění, budou odpovídající skladová umístění vytvořena. Po stisknutí tlačítka této možnosti se pro každé umístění pošle požadavek na příslušný endpoint, který provede tvorbu skladového umístění. Druhou možností bude automatické odstranění skladových umístění, která nejsou v mapě namapována. Po stisknutí tlačítka této možnosti dojde k odeslání požadavků na odstranění daných skladových umístění na příslušný endpoint backendu mapy skladu.

V tlačítku synchronizace bude stejně jako u tlačítka pro uložení mapy v průběhu synchronizace zobrazena v procentech hodnota poměru vykonaných požadavků ku celkovému množství požadavků tak, aby byl uživatel informován o průběhu synchronizace. Po jejím dokončení dojde znovu k načtení mapy skladu z backendu s již aktualizovanými informacemi o mapování umístění.

2.9 Heatmapa vytížení skladových pozic

V této části se věnuji návrhu, který jsem provedl v páté iteraci vývoje projektu. V ní jsem se věnoval funkcionalitě zobrazení heatmapy vytížení skladových pozic.

Aby bylo možné vytvořit heatmapu vytíženosti skladových pozic, bylo nutné znát pro každou pozici na mapě hodnotu počtu pohybů na této pozici normalizovanou v sadě hodnot počtu pohybů všech pozic v mapě mezi 0 a 1. Jedná se tedy o poměr rozdílu množství pohybů na dané pozici a minimálního množství pohybů na jedné pozici ku rozdílu maximálního a minimálního množství pohybů na jedné pozici v mapě. [19] [20] Počet pohybů na jedné pozici je možné spočítat jako součet počtu pohybů na skladových umístěních, která se na dané konkrétní pozici nachází. Výpočet těchto hodnot bude probíhat v backendu mapy skladu. Následně budou pomocí endpointu uvedeného v tabulce 2.13 předány hodnoty poměru pro každou pozici v mapě frontendu.

Prefix: stocks/{stockId}	
URI	
Metoda	**Popis**
/maps/heatmap	
GET	Zobrazí data pro tvorbu heatmapy vytížení skladových pozic

Tabulka 2.13: Specifikace API heatmapy vytížení skladových pozic

Do karty mapa skladu, ve které se nachází editor mapy, bude přidána lišta záložek, ve které bude možné přepínat mezi zobrazením uvedeného editoru mapy skladu a heatmapy vytížení skladových pozic. V zobrazení heatmapy bude vykreslena mapa skladu s tím, že každé políčko v mapě bude obarveno dle hodnoty vypočítané backendem.

Kapitola 3

Implementace

V této kapitole popíši, jakým způsobem jsem v jednotlivých iteracích implementoval řešení navržená v předchozí kapitole. Nejdříve se budu věnovat implementaci nástrojů editoru mapy skladu. Popíši tvorbu backendu mapy a realizaci ukládání a načítání mapy. Zaměřím se na mapování a synchronizaci umístění a také přiblížím implementaci heatmapy vytížení skladových pozic. V závěru této kapitoly se budu věnovat opravě chyb, které byly nalezeny v testování a posouzení kódu.

3.1 Nástroje editoru mapy skladu

V první iteraci jsem v editoru mapy implementoval na frontendu změny týkající se úpravy a záznamu dat do mapy. Data editované mapy skladu jsou v aplikaci uložena ve Vuex store. Store je zjednodušeně kontejner, ve kterém je uložen stav aplikace. Od běžného objektu se liší tím, že nelze napřímo měnit jeho stav, ale změny musí být prováděny pomocí spuštění takzvaných mutací. Díky tomu mohou být změny stavu zaznamenávány. Dále je reaktivní – tedy pokud jsou nějaké Vue komponenty navázány na jeho stav a dojde ke změně, tyto komponenty se reaktivně přizpůsobí daným změnám. [21]

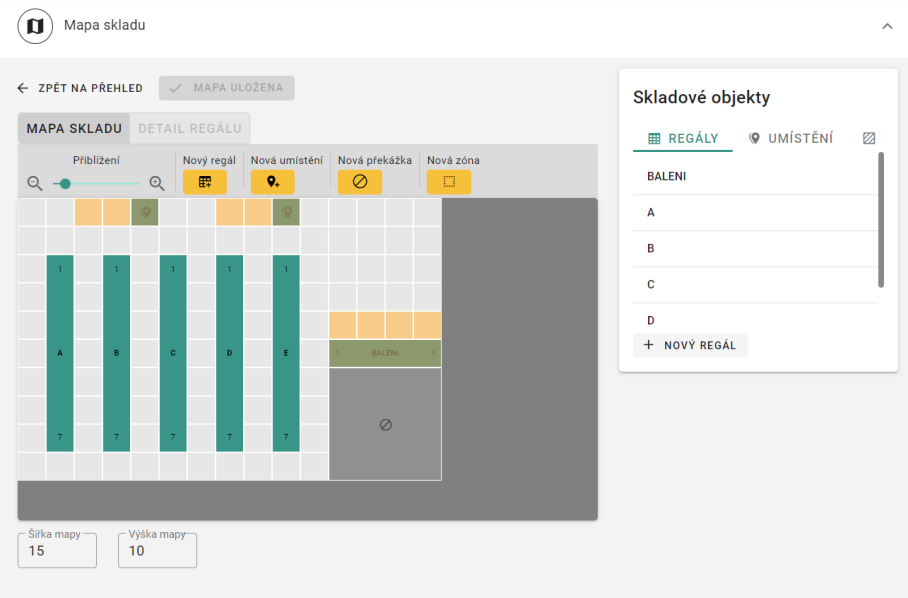
V rámci implementace jsem do mapy nejprve přidal do panelu nástrojů možnost smazat vybraný objekt. Vzhledem k tomu, že mutace mazání objektu byla již v rámci předchozí práce naimplementována, postačovalo do nabídky přidat tlačítko, které realizuje danou funkci.

V komponentě *SidePanel*, která se stará o zobrazení postranního panelu, jsem přidal možnost úpravy oddělovačů v názvech umístění v regálu. Poté jsem přidal do Vuex store modulu mapy skladu mutaci pro jejich úpravu.

Dále jsem do store modulu mapy skladu přidal mutace, které umožňují práci i s jinými objekty, než pouze s regálem. Tyto funkce jsem poté využil při implementaci navržených tlačítek, karet v postranním panelu a samotném zobrazení mapy.

Problém se naskytl u implementace zón. Předchozí implementace komponenty *MapView*, která se stará o vykreslení mapy a způsob uložení dat o mapě v modulu mapy ve store, bránila tomu, aby se na jedné pozici v mapě mohla nacházet zóna a objekt zároveň. Editor byl navržen tak, že se zóny s jinými objekty v mapě nepřekrývají. Z diskuse s vedoucím práce a pracovníky společnosti Jagu s.r.o. vyplynulo, že tato funkcionalita je pro přesné zakreslení mapy nutná. Byla tedy nutná úprava implementace vykreslování mapy.

V rámci první iterace jsem také doimplementoval textové přehledy skladových objektů a oblastí v postranním panelu, které nyní zobrazují všechny objekty a oblasti, které se v mapě skladu nachází. Výsledek implementace první iterace je zachycen na obrázku 3.1.



Obrázek 3.1: Editor mapy skladu po 1. iteraci

3.2 Backend mapy skladu

Ve druhé iteraci jsem implementoval backend editoru mapy skladu. Nejdříve jsem vytvořil v aplikaci Octopus entity, na které budou pomocí knihovny Doctrine mapována data z databáze. Následně jsem vytvořil databázovou migraci, která generuje v PostgreSQL databázi backendu příslušné tabulky. Pro každou entitu jsem implementoval repozitář, který se stará o ukládání a načítání dat z databáze a také třídy services, které pracují s daty z repozitářů.

Následně jsem začal s tvorbou kontrolerů, ve kterých jsou za pomoci Symfony anotací definovány API endpointy. Nejdříve jsem vytvořil endpointy

pro tvorbu a zobrazování mapy skladu bez jejích objektů a oblastí. Následně jsem zpracoval endpoint pro export mapy.

Při implementaci kontroleru pro import mapy jsem zjistil, že bude nejjednodušší vytvořit kotroler pro každý typ objektu a oblasti zvlášť a poté tyto kotrolery využívat v kontroleru pro import k tvorbě, mazání a úpravě příslušných entit. Tento přístup se ukázal jako velmi výhodný. Jak je uvedeno v části 2.6.3, která se věnuje návrhu API endpointů, bylo nakonec nutné místo endpointu import a export pro každý objekt a oblast vytvořit několik samostatných endpointů, které se staraly o jejich zobrazení, tvorbu, úpravu a mazání. V tu chvíli již byly všechny funkce hotové a stačilo k nim pouze přidat správné anotace.

3.3 Ukládání a načítání mapy skladu

V rámci třetí iterace jsem se věnoval implementaci komunikace backendu a frontendu mapy. Jednalo se především o funkcionalitu ukládání a načítání mapy.

Nejdříve jsem upravil objekt *MapAPI*. Ten využívá objektu *BaseAPI*, který v celém frontendu skladového systému zpřístupňuje metody umožňující volání endpointů aplikace Octopus – backendu systému. Do objektu *MapAPI* jsem přidal metody, které se staraly o požadavky zasílané na endpointy vytvořené v backendu v rámci druhé iterace vývoje projektu mapy skladu.

V komponentě editoru mapy jsem definoval metody *loadMap* a *saveMap*. Metoda *loadMap* nejdříve získala z backendu za pomoci objektu *MapAPI* obecné informace o mapě a také data všech objektů, které se v ní nacházely. Poté tato data předala mutaci *importMap*. Tato mutace, vytvořená v store modulu mapy skladu, vymazala data předchozí zpracovávané mapy a vložila do modulu nová data.

Ve store modulu mapy skladu jsem také přidal ke každému objektu nebo zóně v mapě atribut, který určoval, zda byl objekt od posledního načtení mapy vytvořen, upraven, či s ním nebyla provedena žádná akce. Také jsem přidal evidenci vymazaných objektů a zón.

V metodě *saveMap* jsem poté obstaral ukládání mapy. Nejdříve byl v této metodě volán getter store modulu mapy skladu *exportMap*, který vrátil data o upravených údajích o mapě, data vytvořených, upravených a vymazaných objektů. Následně se v *saveMap* zavolaly příslušné metody objektu *MapAPI*, díky kterým se odeslala data o úpravách mapy na backend.

Při otevření editoru mapy se nejdříve ověřilo, že pro daný sklad existuje mapa skladu. Pokud tomu tak nebylo, byl zaslán požadavek na vytvoření mapy. Následně se v komponentě editoru zavolala metoda *loadMap*, která zajistila načtení mapy.

Do editoru jsem dle provedeného návrhu přidal tlačítko, které sloužilo k uložení mapy. Pokud na něj uživatel kliknul, byla zavolána metoda *save-*

Map, která zajistila uložení mapy a následně znovu metoda *loadMap*, která do editoru načetla aktualizovanou mapu.

3.4 Mapování a synchronizace umístění

Ve čtvrté iteraci bylo mým cílem zajistit vhodné mapování a synchronizaci mezi mapovými a skladovými umístěními.

Do store modulu mapy skladu jsem přidal pole *stockLocationMapping*, ve kterém jsem udržoval skladová umístění a k nim počet namapovaných mapových umístění. Nejdříve jsem do funkce *loadMap*, která se starala o načítání mapy, přidal načtení dat skladových umístění upravovaného skladu. Ve funkci *importMap* jsem poté tato umístění uložil do pole *stockLocationMapping* a k nim dle kódů přiřadil počty namapovaných mapových umístění z načítané mapy. Pokud pro mapové umístění neexistovalo skladové umístění s odpovídajícím kódem, bylo v poli vytvořeno nové skladové umístění bez identifikátoru.

Bylo-li přidáno do mapy mapové umístění, zavolala se v modulu akce *mapToStockLocation*, která v poli zvýšila u skladového umístění s odpovídajícím kódem atribut počtu namapovaných mapových umístění a u mapového umístění zaevidovala identifikátor skladového umístění, které reprezentuje. Pokud neexistovalo skladové umístění s odpovídajícím kódem, bylo v poli vytvořeno nové skladové umístění bez identifikátoru.

Jestliže bylo mapové umístění z mapy odebráno, zavolala se v store modulu mapy skladu akce *unmapFromStockLocation*. Ta v poli snížila u skladového umístění s odpovídajícím kódem atribut počtu namapovaných mapových umístění. Pokud se počet snížil na nulu a jednalo o skladové umístění bez identifikátoru, bylo toto umístění odebráno ze seznamu.

Následně jsem dle provedeného návrhu pod editor mapy umístil kartu, ve které byly zobrazeny přehledy mapování. V záložce *Nenamapovaná umístění* jsem zobrazil z pole všechna skladová umístění, která měla nastavený počet namapovaných umístění na hodnotu nula. Záložka *Namapovaná umístění* zobrazovala umístění, u kterých počet odpovídal hodnotě jedna. Umístění, u kterých byl počet větší než jedna, byla umístěna do záložky *Duplicitní umístění*. V záložce *K vytvoření (neexistující umístění)* se poté nacházela umístění, jež v poli neměla evidována identifikátor.

Po implementaci načítání a ukládání mapy ve frontendu jsem provedl změny v backendu skladového systému. Upravil jsem metody pro tvorbu, úpravu a odstranění skladového umístění ve třídě *StockLocationService* tak, aby se provedené změny promítly i do dat mapy skladu.

V případě, že bylo přidáváno nové skladové umístění, zavolala se nově vytvořená metoda *mapStockLocation* definovaná ve třídě *MapLocationService*, která prošla mapu skladu, a pokud existovalo mapové umístění s odpovídajícím kódem, vytvořila vazbu mezi ním a přidávaným skladovým umístěním.

Pokud bylo naopak skladové umístění odstraňováno, zavolala se metoda *unmapStockLocation*. Ta prošla mapu skladu a odstranila vazby, které odstraňované skladové umístění mělo s mapovými umístěními.

Došlo-li k úpravě skladového umístění, bylo toto umístění před provedením úprav zbaveno vazeb na mapu skladu zavoláním metody *unmapStockLocation*. Po dokončení procesu úprav byly vytvořeny dle nově upravených údajů nové vazby voláním metody *mapStockLocation*.

Dle návrhu provedeného v části 2.8.4 jsem do editoru přidal tlačítko a dialogové okno umožňující spuštění synchronizace umístění. Díky způsobu implementace mapování umístění nebylo složité vytvořit samotný mechanismus synchronizace umístění.

Došlo-li v dialogu ke stisknutí tlačítka pro automatickou tvorbu umístění ve skladě, byla spuštěna metoda *startSyncCreateStockLocations*. Ta z pole *stockLocationMapping* vybrala skladová umístění, u kterých nebyl definován identifikátor. Za použití objektu *StockAPI* odeslala na backend pro každé umístění požadavek na vytvoření skladového umístění. Po dokončení tvorby umístění byla pomocí metody *loadMap* znova načtena mapa s aktualizovaným mapováním.

Při spuštění automatického odstranění nenamapovaných umístění, nově vytvořená metoda *startSyncDeleteUnmappedStockLocations* vybrala z pole skladová umístění, u kterých byl počet namapovaných mapových umístění roven nule a za použití objektu *StockAPI* zaslala požadavky na jejich odstranění. Po dokončení tohoto procesu se zavolala metoda *loadMap*, která načetla mapu s aktualizovaným mapováním.

3.5 Heatmapa vytížení skladových pozic

V rámci páté iterace jsem implementoval zobrazení heatmap vytížení skladových pozic v kartě mapy skladu.

Do backendu mapy skladu jsem přidal třídu *HeatMapService*, která prostřednictvím metody *getHeatMap* zajišťovala sestavení dat heatmapy. Tato metoda pro každou pozici v mapě spočítala počet skladových pohybů tak, že sečetla počet pohybů na všech umístěních na této pozici. Počet pohybů pro jedno skladové umístění byl zjištěn za pomoci metody *getMovementAmount*, kterou jsem vytvořil ve třídě *HeatMapService*. Pro každou pozici v mapě byla poté za pomoci těchto hodnot spočítána normalizovaná hodnota mezi 0 a 1 (viz část 2.9). Pole s těmito hodnotami bylo následně předáno třídě *HeatMapController*, ve které byl vytvořen příslušný endpoint.

K implementaci zobrazení heatmapy vytížení skladových pozic ve frontendu jsem využil již vytvořené komponenty *MapView*, která se stará o vykreslení mapy. Zde jsem přidal možnost obarvit pole mapy barvou dle hodnoty, která je tomuto poli v heatmapě přiřazena. Do funkce *loadMap* jsem přidal načítání těchto hodnot z backendu do store modulu mapy skladu.

Do karty *Mapa skladu* jsem poté přidal lištu záložek, ve kterých mohl uživatel přepínat mezi zobrazením editoru mapy a heatmapy vytížení skladových pozic. Toto zobrazení je zachyceno na obrázku 3.6.

3.6 Oprava chyb v implementaci

V šesté iteraci jsem se věnoval opravě chyb v implementaci, které identifikovali zaměstnanci společnosti Jagu s.r.o. v rámci provedeného posouzení kódu. Zároveň jsem v ní také řešil nápravu problémů, které vyplynuly z provedeného uživatelského testování.

Největším problémem, který jsem musel po provedení testování řešit, bylo rozšíření možnosti úpravy systému pojmenovávání umístění v regálu. Zde bylo nutné umožnit uživateli změnu pořadí symbolů, které v názvu umístění reprezentovaly pořadí sloupce a patra, ve kterém se umístění nacházelo a pořadové číslo umístění v buňce. Do formuláře pro úpravu jmenovacího systému jsem proto přidal pro každý symbol pole, ve kterém bylo možné vybrat, o jaký typ symbolu se jedná. Uživatel si tedy poté mohl zvolit, co který symbol v názvu reprezentoval, a nastavit tak téměř jakýkoliv jmenovací systém. Přidal jsem také možnost symbol nevyužít a tím umožnil automatickou tvorbu regálů se jmenovacím systémem s menším počtem symbolů.

Také jsem přidal možnost vkládat do buňky regálu umístění manuálně, což otevřelo možnosti pro tvorbu jakéhokoliv regálu. Využil jsem u toho již implementované komponenty *EditLocationsForm*, která sloužila k manuálnímu vkládání umístění do skupiny umístění. Do ní jsem předal, stejně jako u skupiny umístění, mapová umístění, která obsahovala buňka regálu a následně reagoval na přidání či odebrání umístění do formuláře nově vytvořenými mutacemi store modulu mapy skladu *addNewLocationToSelectedRackCell* a *removeLocationFromSelectedRackCell*.

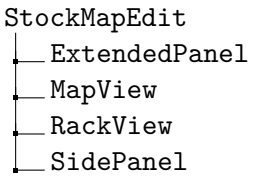
Dalším problémem, který jsem v šesté iteraci řešil, bylo chybné mapování mapových umístění na skladová umístění. Ta byla mapována dle uživatelem zadaného kódu. Bylo však nutné, aby uživatel zadával do mapy název umístění. U mapových umístění jsem tedy přidat na frontendu i backendu atribut názvu a změnil formuláře tak, aby do nich uživatel zadával název umístění. Ten jsem používal při následném mapování. Pokud se ve skladě nacházela umístění, která měla stejný název, zobrazil jsem uživateli dialog, ve kterém se mohl rozhodnout, jaké umístění namapuje.

Do postranního panelu editace regálu jsem také dle výstupů z testování přidal možnost měnit název regálu. U toho jsem použil stejného formuláře, jaký byl použit v postranním panelu detailu objektu v zobrazení mapy.

Rovněž bylo nutné opravit nefunkční formulář pro změnu rozměrů objektu v postranním panelu detailu objektu. Při zneplatnění totiž při dalším zadání nereagoval na nově správně zadaný vstup, což bylo způsobeno tím, že se při validaci kontrolovala špatná proměnná.

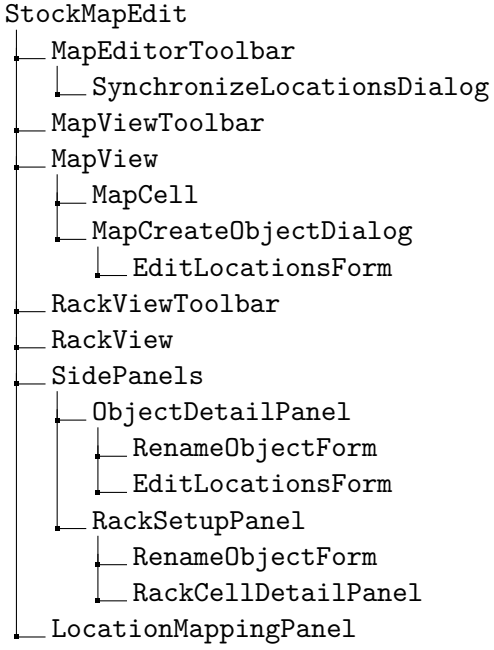
Do dialogového okna, které se zobrazovalo při tvorbě zóny a překážky, jsem přidal do formuláře chybějící pole pro zadání jejich názvu.

Z chyb, které byly nalezeny v posouzení kódu, byla nejzávažnější nepřehlednost kódu frontendu. Komponenty, které jsem implementoval, byly přiliž dlouhé, nepřehledné a kód se často opakoval. Před opravou této chyby se frontend editoru mapy skládal z komponent uvedených v obrázku 3.2.



Obrázek 3.2: Komponenty editoru mapy před 6. iterací

Při úpravách, které jsem v šesté iteraci implementoval, se mi podařilo rozdělit editor do více komponent a zpřehlednit kód frontendu. Výsledná struktura komponent je znázorněna na obrázku 3.3.



Obrázek 3.3: Komponenty editoru mapy po 6. iteraci

Zajímavý problém nastal, když byl frontend mapy spuštěn v produkčním režimu. Při opuštění stránky detailu skladu zůstal díky ukládání do mezipaměti editor mapy daného skladu aktivní. Když byla následně otevřena stránka detailu jiného skladu, byly najednu aktivní dva editory. To zapříčinilo

výrazné zpomalení, častokrát zaseknutí systému. Tento problém jsem odstranil deaktivací editoru mapy při opuštění stránky detailu skladu.

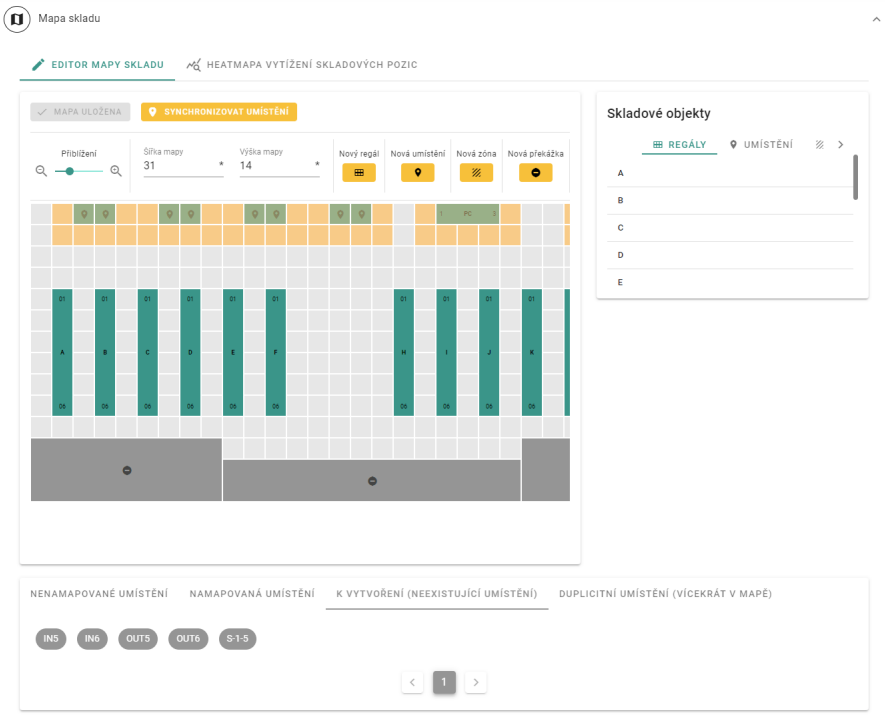
V rámci implementace formulářů jsem na frontendu nevyužíval komponenty *FormFields*. V aplikaci Swordfish byla však pro většinu formulářů tato komponenta použita. S ohledem na zbytek systému jsem pro zajištění konzistence vzhledu i chování formulářů v editoru mapy všechny formuláře upravil tak, aby využívaly uvedenou komponentu.

Také jsem upravil barvy, které byly v editoru využity tak, aby vzhled editoru lépe navazoval na zbytek systému, a přidal překlady systému do slovenštiny a maďarštiny.

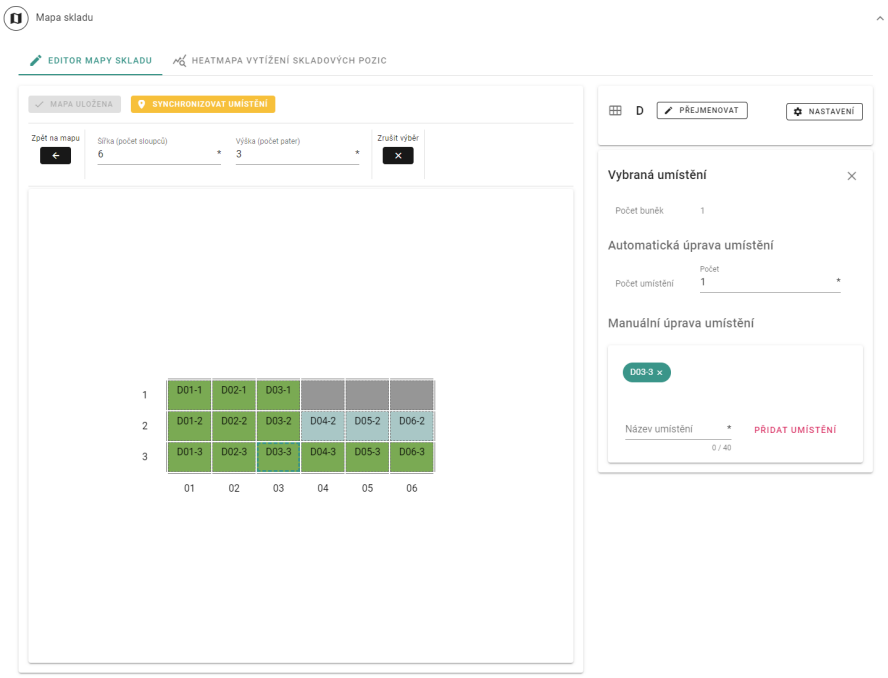
Verze editoru mapy skladu, která byla výsledkem této bakalářské práce, je zachycena na obrázcích 3.4, 3.5 a 3.6. Další snímky obrazovek jsou dostupné v přiloženém médiu v příloze C.

3.7 Zdrojové kódy

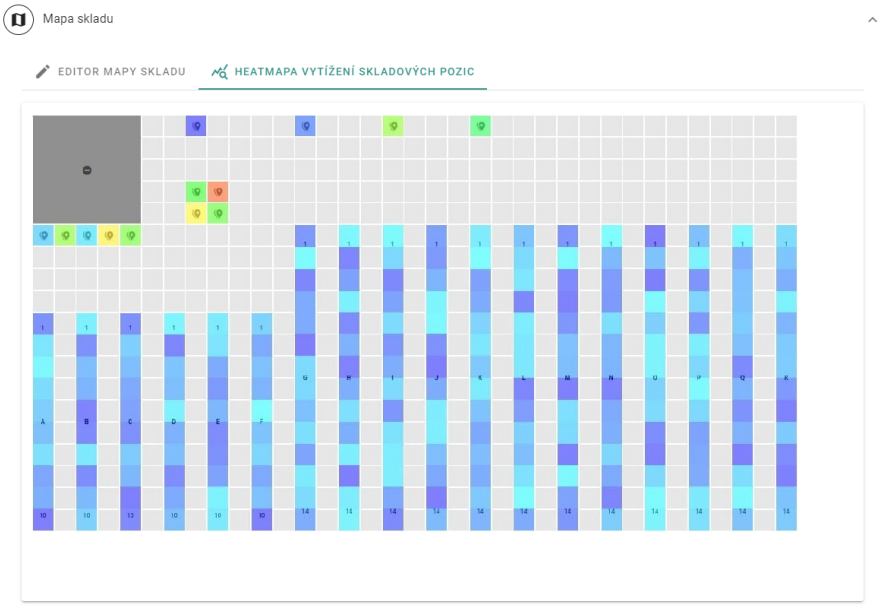
Zdrojové kódy editoru mapy skladu jsou dostupné na GitLabu společnosti Jagu s.r.o. Implementace backendu mapy skladu se nachází v projektu atlantis/Octopus ve větvi storage_map. Frontend editoru mapy lze nalézt v projektu atlantis/Swordfish ve větvi storage_map_completion.



Obrázek 3.4: Editor mapy skladu po 6. iteraci



Obrázek 3.5: Editor mapy skladu po 6. iteraci – detail regálu



Obrázek 3.6: Editor mapy skladu po 6. iteraci – heatmapa

Kapitola 4

Testování

V rámci každé iterace jsem vždy průběžně vytvořené části projektu testoval. Po každé iteraci jsem provedl manuálně funkční testování tak, abych ověřil, že chování systému odpovídá návrhu, který jsem provedl. Nalezené chyby jsem případně vždy ještě v rámci iterace, ve které testování probíhalo, opravil. Při vývoji backendu mapy skladu jsem prováděl statické testy formy a obsahu zdrojového kódu, ve kterých jsem využil automatického testování, které v aplikaci Octopus implementoval a popsal ve své práci Ing. Pavel Kovář [7]. Po dokončení páté iterace také provedli zaměstnanci společnosti Jagu s.r.o. posouzení kódu, takzvané code review, jehož cílem bylo identifikovat chyby, kterých jsem se v implementaci dopustil.

V této kapitole se budu věnovat popisu testování použitelnosti, které jsem také provedl po dokončení páté iterace vývoje a které bylo pro projekt stěžejní. Nejprve vysvětlím pojem použitelnost a připravím plán jejího testování. Následně definuji testovací případy a sestavím scénáře testování. Popíši průběh samotného testování a závěrem shrnu jeho výsledky.

4.1 Použitelnost

Dle Nielsena [22] je použitelnost kvalitativní vlastnost rozhraní hodnotící jednoduchost jeho užívání.

V rámci použitelnosti se například hodnotí, jak jednoduchá je práce s rozhraním bez jeho předešlé znalosti, jak rychle dokáže uživatel po seznámení se s rozhraním plnit složitější úkoly či jak snadno lze obnovit znalosti práce s rozhraním po delší době, kdy uživatel rozhraní nepoužíval. Také se sleduje kolik a jaké chyby uživatelé při práci s rozhraním dělají a jak je jednoduché chyby napravit nebo jak příjemné je pro uživatele užívání rozhraní.

Důležitým pojmem je také užitkovost, která hodnotí, zda dané rozhraní poskytuje uživateli funkcionalitu, kterou potřebuje. Spolu s použitelností určuje užitkovost užitečnost rozhraní.

4.2 Plán testování

Plán testování je základním podkladem obsahujícím veškeré informace týkající se konkrétního procesu testování [23]. V tomto plánu představím cíle testování a metody, které při něm využiji. Dále přiblížím, jak bude testování hodnoceno, představím účastníky, kteří v něm budou figurovat, prostředí, ve kterém bude testování probíhat a naplánuji samotný postup testování.

4.2.1 Cíle testování

Cílem tohoto testování je zjistit, zda řešení, které jsem v rámci bakalářské práce vytvořil, dosahuje dostatečné použitelnosti a je možné ho integrovat do skladového systému Atlantis. Jde tedy o to zjistit, zda je vytvořený editor mapy dostatečně užitečným nástrojem k práci s mapou skladu. Cílem je také identifikovat chyby a problémy, které se v rozhraní nachází a získat potřebná data, která by mohla vést k návrhu nápravy těchto chyb. Zároveň chci také odhalit, jaké jsou příležitosti pro vylepšení daného rozhraní.

4.2.2 Metody testování

Vzhledem k tomu, že cílem těchto testů je ověřit použitelnost daného řešení, bylo logické využít metodu testování použitelnosti.

Testování použitelnosti je metoda založená na pozorování reálného uživatele při vykonávání specifického úkolu v testovaném systému. Používá se pro zjišťování použitelnosti systému a dokáže identifikovat problémy v jeho návrhu, odhaluje příležitosti ke zlepšení a poskytuje informace o chování a preferencích cílových uživatelů. Během testu výzkumník pozoruje chování účastníka při práci se systémem, akce, které provádí, naslouchá jeho zpětné vazbě a případně klade doplňující otázky za účelem získání dalších podrobností. [24] [25]

4.2.3 Hodnocení testování

Testování jsem se rozhodl hodnotit jak kvalitativně, tak i použít kvantitativní metriku.

V rámci kvalitativního hodnocení jsem se v průběhu testování zaměřil na zpětnou vazbu a sledování chování účastníků při interakci s rozhraním. Dále jsem také v průběhu testování kladl účastníkům otázky, které měli za cíl zjišťovat, jaké pocity při práci s rozhraním mají a jak jednoduché je pro ně rozhraní používat. Dále jsem se také při vytváření kvalitativního hodnocení rozhodl zaměřit na chyby, kterých se účastníci při práci s rozhraním dopouštěli a sledovat, jak je pro ně náročné tyto chyby napravit. Důležité pro mne také bylo vzít v potaz to, zda účastník byl schopen s rozhraním pracovat samostatně a efektivně. Zároveň jsem se také po testu rozhodl formou rozhovoru s účastníky prodiskutovat znovu průběh celého testování a jejich pohled na testované rozhraní, abych upřesnil získaná data.

Dále jsem se rozhodl vyhodnotit testování i kvantitativně. Využil jsem velmi populárního dotazníku SUS, který jsem nechal účastníky po skončení testu vyplnit.

System Usability Scale, zkráceně SUS je dotazník od Johna Brookse, který se skládá z 10 tvrzení týkajících se použitelnosti testovaného systému. Pro ně účastník vybírá na škále hodnotu od 1 do 5, která reprezentuje, jak moc s tímto tvrzením souhlasí. Na této škále reprezentuje 1 silný nesouhlas s uvedeným tvrzením, naopak 5 silný souhlas.

Při výpočtu výsledného skóre je od výsledku otázky s lichým pořadovým číslem odečtena 1, výsledek otázky se sudým pořadovým číslem je odečten od 5. Získané hodnoty jsou poté sečteny a tento součet je vynásoben koeficientem 2.5 tak, aby se výsledná hodnota skóre nacházela v rozmezí 0 až 100.

Tato hodnota poté řadí systém na škálu, kde se skóre vyšší než 68 považuje za nadprůměrnou hodnotu, tedy hodnotu, při které je systém dostatečně použitelný. Skóre nad 73 pak řadí systém mezi nejlepších 30% všech testovaných systémů a skóre nad 80 poté mezi vybraných 10%. [26] [27]

Dotazník SUS se zkládá z následujících otázek:

1. Myslím si, že bych systém rád často používal.
2. Systém mi připadal příliš komplexní.
3. Myslím si, že systém byl jednoduchý na použití.
4. Myslím si, že bych potřeboval pomoc technické podpory, abych zvládl systém používat.
5. Přišlo mi, že různé funkce byly v systému dobře propojeny.
6. Připadalo mi, že systém je příliš nekonzistentní.
7. Dokáži si představit, že většina lidí by se naučila systém používat velmi rychle.
8. Užití systému mi přišlo velmi těžkopádné.
9. Při používání systému jsem se cítil velmi sebejistě.
10. Než jsem mohl systém začít používat, musel jsem se naučit spoustu věcí.

4.2.4 Účastníci

Toto testování jsem se rozhodl provést s pěti účastníky což, jak uvádí Nielsen [28], stačí k odhalení zásadních chyb a pokrytí zhruba osmdesáti pěti procent problémů v použitelnosti. Jakmile totiž testujeme s více účastníky, tito další účastníci identifikují v systému stále stejné problémy, které již výzkumník zaznamenal. S každým dalším účastníkem již totiž dle Nielsena výzkumník jen

ztrácí čas opakovaným pozorováním stejného chování a nedozvídá se mnoho nového.

Při výběru účastníků jsem bral především v potaz aktéry, kteří byli pro systém určeni v kapitole 1.5.1. Snažil jsem se testování provádět především se zaměstnanci skladů, kteří pracují na pozici vedoucího a ve svém skladu již používají systém Atlantis. Dále jsem chtěl také zahrnout jednoho z expertů ze společnosti Jagu s.r.o., který se zabývá vývojem systému Atlantis a dané problematice velmi dobře rozumí. Bylo také vhodné testovat uživatele, kteří pracují ve skladech různých velikostí.

Díky vedoucímu mé práce a společnosti Jagu s.r.o. jsem dostal možnost testovat systém s jeho reálnými uživateli. Testu se tak účastnil jeden z expertů z této společnosti (Oldřich) a čtyři vedoucí reálných skladů (Marek, Tomáš, Libor a Veronika). Marek a Tomáš pracovali ve velkých skladech, kde se nacházelo 5000, nejvíce až 8000 skladových umístění. Libor zastupoval střední sklad s řádově 400 až 500 umístěními a Veronika pracovala ve skladě s celkově 50 až 100 umístěními.

4.2.5 Prostředí

Zaměstnanci společnosti Jagu s.r.o. nahráli do testovacího prostředí editor mapy a připravili v něm data, která byla podobná s produkčním prostředím, ve kterém pracovali daní účastníci. Díky tomu bylo možné simulovat podmínky, ve kterých by účastníci za normálních okolností s mapou pracovali. Uživatel pak dostal URL adresu testovacího prostředí a v něm pracoval na svém zařízení se systémem, který ze svého zaměstnání znal.

Testování jsem prováděl dle možností buď přímo na pracovišti účastníka nebo vzdáleně. V obou případech jsem využíval aplikaci Google meet, která umožňuje pořádání online schůzky. Můj a účastníkův počítač byl připojen k jedné schůzce, ve které účastník sdílel zvuk a obrazovku svého počítače a umožnil mi tak nahrávání celého testu. V případě vzdáleného testu jsem také díky této schůzce mohl s účastníkem komunikovat.

4.2.6 Postup testování

Bylo velmi důležité stanovit si jasný průběh a postup testování tak, aby vše probíhalo hladce. Testování jsem se rozhodl rozdělit do tří částí – úvodu, hlavní části a závěru.

V úvodu jsem nejdříve uvítal účastníka a pokusil se navázat přátelskou a příjemnou konverzaci tak, abych účastníka nechal vydechnout, uvolnit a zbavil ho počáteční nervozity a obav. Poté jsem účastníka ubezpečil, že netestujeme jeho schopnosti, ale použitelnost rozhraní. Pokud si tedy v rámci testu nebude vědět rady, není to jeho chyba ale spíše problém nedostatečné intuitivnosti testovaného rozhraní. Následně jsem účastníka seznámil s postupem testu a s připravenými testovacími scénáři.

Poté jsem s účastníkem vyplnil vstupní dotazník, který zjišťoval následující informace:

- Jméno a příjmení
- Věk
- Pracovní pozice ve skladě, jaké činnosti vykonává
- Jak dlouho již pracuje ve skladech
- Jak dlouho již pracuje ve skladě, pro který provádíme testování
- Jak velký je sklad, ve kterém pracuje, kolik umístění řádově obsahuje
- Zda-li již někdy kreslil mapu skladu (na příklad pro vlastní účely, v jiném systému)

Poté jsem přistoupil k hlavní části. Zde jsem s účastníkem prošel příslušné testovací scénáře. Bedlivě jsem sledoval, jak se účastník během testu chová, jaké akce provádí a jakou zpětnou vazbu poskytuje. Vše jsem si řádně zapisoval do poznámkového bloku. Zároveň jsem také nahrával obrazovku počítače účastníka a zvuk pro pozdější vyhodnocení.

V závěru jsem vedl s účastníkem rozhovor, ve kterém jsem se snažil zjistit dodatečné informace, které jsem během hlavní části nezaznamenal a které by mi mohly pomoci při kvalitativním hodnocení testování. Následně jsem nechal účastníka vyplnit dotazník SUS a poděkoval mu za účast na testování.

4.3 Testovací případy

Testovací případ (anglicky test case) je souhrn akcí prováděných s určitou softwarovou komponentou a jejich očekávaných výsledků, díky kterým lze rozpoznat, zda tato komponenta funguje dle požadavků zákazníka či nikoli. Pro návrh takového případu je nejčastěji nutné především definovat název testovacího případu, počáteční podmínky podstatné pro provedení případu, specifikace vstupních dat a kroků, které případ zahrnuje a očekávaný výsledek testovacího scénáře. [29] [30] Níže uvádím seznam testovacích případů, které jsem pro toto testování identifikoval. Detailně jsou tyto případy popsány v příloze B této bakalářské práce.

- TC1 Tvorba nové mapy skladu
- TC2 Úprava rozměrů mapy skladu
- TC3 Tvorba nového regálu
- TC4 Tvorba nové skupiny umístění

- TC5 Tvorba nové zóny
- TC6 Tvorba nové překážky
- TC7 Změna počtu sloupců a pater regálu
- TC8 Změna jmenovacího systému regálu
- TC9 Změna počtu umístění v buňkách regálu
- TC10 Přiřazení nových rozměrů umístění buňkám regálu
- TC11 Odstranění rozměrů umístění z mapy skladu
- TC12 Změna polohy objektu či zóny
- TC13 Rotace objektu či zóny
- TC14 Tvorba kopie objektu či zóny
- TC15 Mazání objektu či zóny
- TC16 Změna názvu objektu či zóny
- TC17 Změna rozměrů objektu či zóny
- TC18 Změna typů zóny
- TC19 Přidání umístění do skupiny umístění
- TC20 Odebrání umístění ze skupiny umístění
- TC21 Uložení mapy skladu
- TC22 Synchronizace umístění – tvorba skladových umístění
- TC23 Synchronizace umístění – odstranění skladových umístění

4.4 Testovací scénáře

Testovací scénář je sada několika testovacích případů, které na sebe logicky navazují a tvoří tak skupinu kroků, jejichž cílem je otestovat vybranou funkčnost aplikace. Cílem je simulovat konkrétní způsob používání aplikace. Nejčastěji se simulují procesy, ke kterým bude aplikace využita u zákazníka. [31] [32]

V rámci tohoto testování jsem připravil několik testovacích scénářů, které jsou specifické pro jednotlivé účastníky a počáteční data o skladu vedená v systému Atlantis.

TS1 Tvorba mapy pro reálný sklad využívající systém Atlantis

Tento scénář simuluje situaci, která nastává v případě, kdy je v systému již definován sklad a skladová umístění v něm, ale ještě není vytvořena mapa skladu. Je určen pro účastníky testování, kteří pracují v reálném skladu využívajícím systém Atlantis – primární persony. Zároveň je také nutné, aby se před začátkem tohoto scénáře v systému nacházel sklad se skladovými umístěními, který byl totožný se skladem, ve kterém daný účastník pracuje.

Tento scénář je připraven tak, aby při jeho ideálním průběhu pokryl testovací případy TC1 až TC10 a dále také případy TC13, TC14, TC16, TC17 a TC21.

Pro pokrytí uvedených testovacích případů v rámci tohoto scénáře jsem se rozhodl vybrat účastníky, kteří pracují ve skladech, pro které platí následující kritéria:

- Ve skladu se nachází regály se skladovými umístěními.
- Ve skladu se nachází skladová umístění, která nejsou součástí regálů.
- Ve skladu se nachází oblasti, kterými není možné při pohybu ve skladě procházet.
- Ve skladu se nachází určené oblasti, ve kterých dochází k příjmu, výdeji případně balení zboží.

Vzhledem k tomu, že tato kritéria splňovala většina běžných skladů, nebyl výběr účastníku příliž omezen.

V tomto scénáři jsem se rozhodl na začátku určit uživateli následující zadání: „Vytvořte a uložte v systému mapu skladu, ve kterém pracujete tak, jak nejlépe zvládnete.“ Poté jsem se rozhodl do průběhu testu příliž nezasahovat a nechat uživatele tvořit mapu a u toho sledovat jeho chování a práci v editoru. Nevěděl-li však účastník v průběhu testu po delší dobu (30 vteřin až jedna minuta), jak dále postupovat, pozastavil jsem test a od účastníka zjistil, jaký další krok chce vykonat a z jakého důvodu mu není jasné, jak postupovat. Poté jsem mu případně poradil správný postup tak, aby mohl v tvorbě mapy pokračovat.

Jakmile účastník ukončil proces tvorby mapy, vyhodnotil jsem, zda byla mapa v systému vytvořena správně a zda byly opravdu pokryty dané testovací případy. Pokud tomu tak nebylo, přistoupil jsem v testování k alternativnímu scénáři TAS1, ve kterém jsem se pokusil uživatele navést ke správnému řešení a případně tak pokrýt zbývající testovací případy.

TS2 Úprava mapy reálného skladu využívajícího systém Atlantis

Tento scénář navazuje na testovací scénář TS1 (případně TAS1), ve kterém byla vytvořena mapa skladu. Cílem scénáře je tuto mapu upravit. Jedná se

o simulaci situace, kdy je sklad přestavěn a jsou přidána a odebrána některá skladová umístění. Tento scénář pokrývá testovací případy TC11, TC12, TC15 a dále také TC18 až TC23.

Vzhledem k tomu, že jsem předem nevěděl, jaká data účastníci v rámci předchozích scénářů do mapy zanesou a tedy jak bude možné sklad přestavět, rozhodl jsem se v tomto scénáři improvizovat. Uživateli jsem se rozhodl v průběhu testu slovně zadávat úkoly, které povedou k úpravě skladu. Tyto úkoly však bylo důležité formulovat tak, aby neobsahovaly příliž mnoho informací o tom, jak je v systému provést, ale zaměřily se spíše na informace o tom, co má být jejich výsledkem. Rozhodl jsem se vybrat následující kategorie úkolů, které jsem v rámci tohoto scénáře s účastníkem prošel:

- Úprava polohy některých z objektů či zón
- Rotace některých objektů a zón
- Změna typu konkrétní zóny
- Přidání nového mapového umístění do jedné ze skupin umístění
- Odebrání skladového umístění z jedné ze skupin umístění
- Mazání rozměrů umístění
- Mazání některých objektů a zón
- Uložení upravené mapy
- Synchronizace umístění

TS3 Tvorba mapy pro nový fiktivní sklad

Tento scénář je určen pro účastníky, kteří v žádném skladu nepracují – sekundární persony. Simuluje situaci, kdy uživatel tvoří mapu skladu dle připraveného plánku. Tento scénář je připraven tak, aby při jeho ideálním průběhu pokryl testovací případy TC1 až TC10 a dále také případy TC13, TC14, TC16, TC17 a TC21.

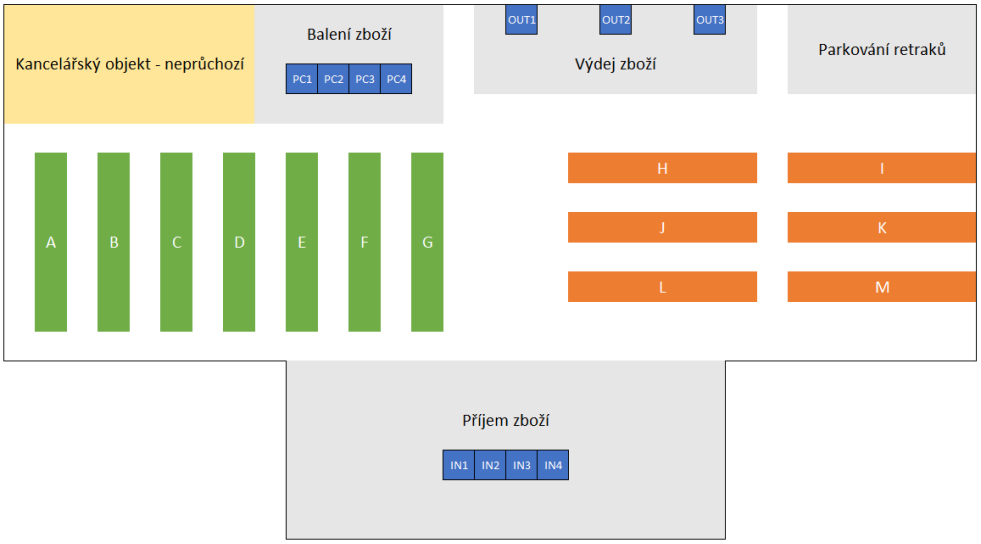
Ve scénáři jsem se rozhodl na začátku určit uživateli následující zadání: „Vytvořte a uložte v systému mapu skladu, která bude reflektovat sklad vyobrazený na poskytnutých podkladech.“ Průběh tohoto scénáře jsem se rozhodl více moderovat a uživateli vysvětlovat, co v plánku vidí tak, aby test nebyl ovlivněn tím, že by uživatel neporozuměl podkladům, ze kterých tvorba mapy vycházela. Nevěděl-li účastník v průběhu testu po delší dobu, jak dále postupovat, pozastavil jsem test a zjistil, jak chce uživatel dále postupovat a proč tak nečiní.

Jakmile účastník ukončil proces tvorby mapy, vyhodnotil jsem, zda byla mapa v systému vytvořena správně a zda byly opravdu pokryty dané testovací

případy. Pokud tomu tak nebylo, přistoupil jsem v testování k alternativnímu scénáři TAS1, ve kterém jsem se pokusil uživatele navést ke správnému řešení a případně tak pokrýt zbývající testovací případy.

Plánek fiktivního skladu, který jsem pro testování použil je patrný na obrázcích 4.1 a 4.2.

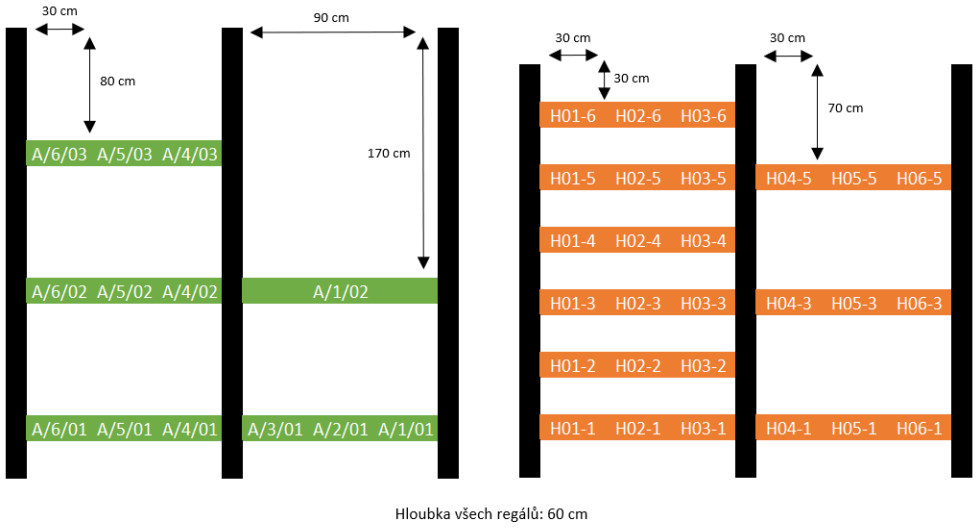
Při návrhu těchto podkladů pro mě bylo prioritou sestavit fiktivní sklad tak, aby scénář pokrýval uvedené testovací případy. Ve fiktivním skladu se proto nachází různé typy zón, které jsou v plánku zaneseny šedivou barvou. V těchto zónách se nachází několik samostatných umístění reprezentovaných modrými čtverečky. Kancelářský prostor, který je v plánku vyznačen žlutě má představovat neprostupnou překážku. Dále se zde nachází dva různé typy regálů tak, aby bylo také možno otestovat možnosti tvorby a uložení různých kofigurací regálů. Rovněž je patrné, že sklad nemá obdélníkové hranice.



Obrázek 4.1: Plán fiktivního skladu 1

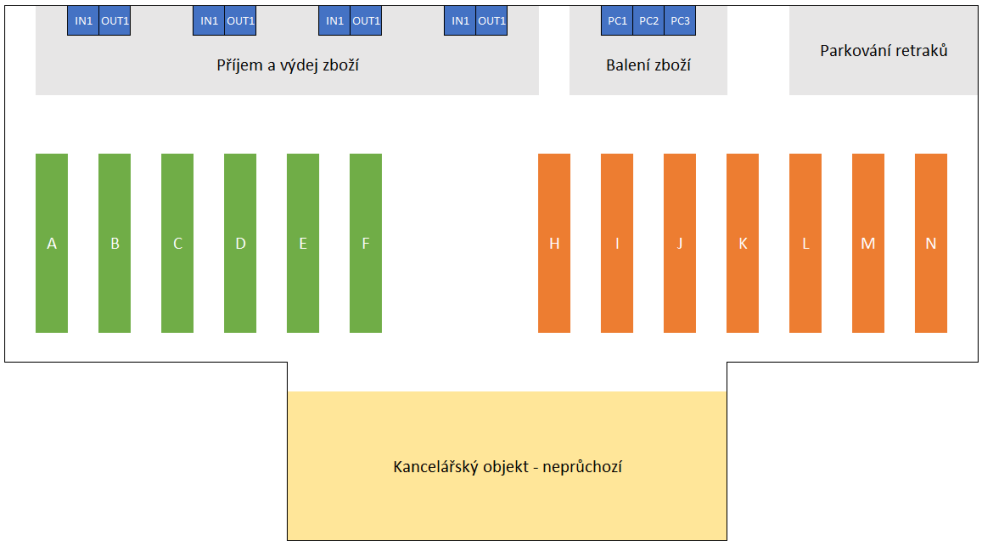
TS4 Editace existující mapy fiktivního skladu

Tento scénář navazuje na testovací scénář TS3 (případně TAS1), ve kterém byla vytvořena mapa fiktivního skladu. Cílem scénáře je tuto mapu upravit tak, aby odpovídala plánku fiktivního skladu 2 vyobrazeného na obrázcích 4.3 a 4.4. Jedná se o simulaci situace, kdy je sklad upraven a je nutné dle nového rozložení upravit mapu skladu. Cílem tohoto scénáře je pokrýt především funkcionalitu, která nebyla otestována v předchozím scénáři. Proto tento scénář pokrývá i následující testovací případy: TC11, TC12, TC15 a dále také TC18 až TC23.

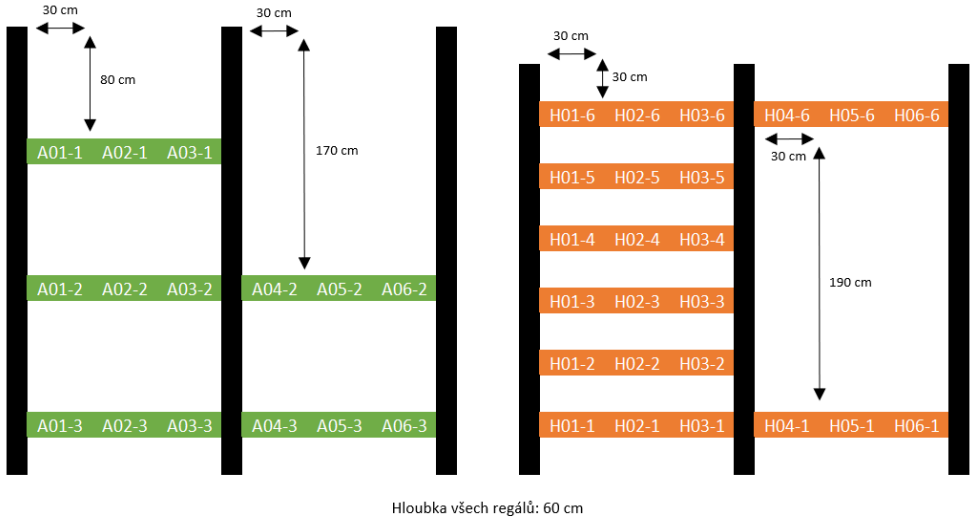


Obrázek 4.2: Plán fiktivního skladu 1 – detail regálů

V rámci scénáře jsem uživateli určil následující zadání: „Upravte vytvořenou mapu skladu tak, aby odpovídala novým podkladům – plánku fiktivního skladu 2.“ Očekával jsem, že účastník se bude vzhledem k absolvování předcházejícího testovacího scénáře v editoru a podkladech lépe orientovat, a proto jsem se rozhodl test příliž nemoderovat.



Obrázek 4.3: Plán fiktivního skladu 2



Obrázek 4.4: Plán fiktivního skladu 2 – detail regálů

TAS1 Alternativní scénář tvorby mapy skladu pro reálný sklad

Tento scénář nastává ve chvíli, kdy uživatel ve scénáři TS1, TS3 nebo TS4 nevytvoří zcela přesně mapu skladu. Cílem scénáře je dokončení nesprávně zakreslené mapy a otestování veškerých funkcionalit, které má TS1 pokrývat.

Před začátkem tohoto scénáře jsem vyhodnotil, jakých chyb se účastník při tvorbě mapy dopustil a identifikoval testovací případy, které nebyly v předchozím scénáři pokryty.

Následně jsem účastníkovi slovně zadával menší úkoly, které vedly k úpravě mapy do cílového stavu. Tyto úkoly však bylo důležité formulovat tak, aby neobsahovaly příliž mnoho informací o tom, jak je v systému provést, ale zaměřily se spíše na informace o tom, co má být výsledkem daného úkolu.

4.5 Průběh testování

V této části popíši průběh testování, chování a akce uživatelů při něm. Budu se snažit vybrat především věci důležité z hlediska použitelnosti a práce s rozhraním, případně chyby, které jsem zaznamenal.

Funkci tvorby mapy skladu se podařilo poměrně jednoduše najít všem účastníkům. Jedinou výjimkou byl Marek. Editor mapy skladu se v systému nacházel na stránce detail regálu. Na této stránce se však před ním nacházely dvě karty se seznamy podskladů a umístění ve skladě. Pokud se v těchto seznamech nachází více položek, není karta s editorem mapy skladu v prohlížeči

vidět a je nutné skrolovat na stránce níže, kde se karta nachází. Proto je překvapující, že většina uživatelů tuto kartu poměrně jednoduše nalezla. Marek nejdříve přešel na stránku *Detail skladu*. Zde však neviděl editor mapy skladu, a tak odešel na stránky *Stav skladu* a *Vlastníci podskladů*. Když zjistil, že ani zde se editor mapy skladu nenachází, vrátil se na stránku detail skladu, kde se pokusil pokračovat přes tlačítka *Nový podsklad* a *Nový vlastník* podskladu. Když ani zde nebyl úspěšný, byl by tvorbu mapy vzdal.

První akcí, kterou učinil Libor, bylo to, že kliknul na tlačítko Synchronizovat umístění. V detailu synchronizace zvolil možnost „Automaticky odstranit nenamapovaná umístění ve skladě“. Vzhledem k tomu, že v mapě ještě nebyla zakreslena žádná umístění, odstranil tím všechna umístění vedená v daném skladu. V následné diskusi Libor upřesnil, že na toto tlačítko kliknul, protože ho nic nevarovalo, co tím může způsobit. Shodli jsme se, že by bylo vhodné přidat ještě jedno dialogové okno, které by uživatele upozornilo, že maže ze skladu daná umístění a vyžadovalo by jeho potvrzení. Ostatní účastníci si před začátkem procesu tvorby mapy tlačítka pro synchronizaci umístění příliš nevšímali.

Všichni uživatelé rychle nalezli možnost tvorby objektů a zón a velmi rychle do skladu přidali nový regál. Následně docela snadno zobrazili detail regálu.

Jako první jsem prováděl test pro sklad, ve kterém byl vedoucím Marek. Při tomto testu jsme narazili na problém, kvůli kterému nebylo možné vytvořit regál s umístěními, která se ve skladu nacházela. Jmenovací systém regálů, které se v tomto skladu nacházely, totiž neodpovídal systému, který v editoru mapy definoval Bc. Kopecký. V daném skladu byla umístění v regálech pojmenovávána řetězcem, ve kterém byl nejdříve název regálu, poté číslo sloupce, pořadí umístění v buňce a nakonec číslo patra. Systém však podporoval pouze jmenovací systém s pořadím název regálu, číslo sloupce, číslo patra a pořadí umístění v buňce. Do regálu také nebylo možné přidávat umístění s jiným kódem, než který odpovídal jmenovacímu systému.

Před dalšími testy jsem se tedy rozhodl přidat funkcionalitu manuální úpravy umístění v buňkách regálu tak, aby bylo možné tvořit jakékoliv regály bez ohledu na to, jaký jmenovací systém sklad využívá. Díky tomu bylo možné další testy provést bez problémů a uživatel mohl namapovat v regálech všechna umístění, která ve skladu měl. Z uvedeného problému také vyplynulo, že by bylo vhodné přidat možnost úpravy pořadí symbolů v jmenovacím systému tak, aby bylo možné automaticky generovat umístění i v regálech, ve kterých jsou umístění pojmenovávána jiným způsobem. Tuto funkcionalitu jsem v systému implementoval před posledním testem, který jsem prováděl s Oldřichem.

V dalších testech se ukázalo, že manuální úprava umístění je pro uživatele velmi srozumitelná a dostatečně pochopitelná. Jakmile se účastníci dostali do detailu regálu, okamžitě začali umístění manuálně upravovat. Pokud bylo moc pracné celý regál vytvářet manuálně, začali zkoumat automatickou úpravu umístění. Při úpravě jmenovacího systému bylo pro Marka složité pocho-

pit, co které pole ve formuláři znamená. Všem ostatním účastníkům to však nedělalo problém.

Jediným malým nedostatkem, na který upozornil Libor při tvorbě jednoho z regálů, bylo to, že nebylo možné jako počáteční symbol sloupce nebo patra nastavit symbol menší než 1. Sloupce regálů ve skladu, ve kterém pracoval, totiž začínaly symbolem 0. Tento problém jsem také odstranil před posledním testem v rámci úpravy funkcionality automatické generace umístění v regálu.

Libor i Veronika dále zmínili, že by pro ně bylo příjemné, pokud by mohli změnit název regálu přímo v detailu regálu a nemuseli by se vracet kvůli tomuto úkonu do mapy. Libor také uvedl, že by bylo vhodné, kdyby byl při zobrazení detailu regálu k dispozici panel s náhledem, ve kterém by viděl, náhled regálu v mapě a mohl se tak lépe orientovat.

Oldřich upozornil na to, že při kopírování objektu či oblasti není v mapě zvýrazněn čtvereček, ve kterém se aktuálně nachází kurzorem myši a na který by byl objekt zkopírován tak, jako tomu bylo například při vkládání nového objektu. Také uvedl, že by bylo vhodné do textových seznamů objektů a oblastí, které jsou zobrazeny v postranním panelu, přidat možnost prohledávat a filtrovat objekty ve skladu.

Všichni účastníci rychle přišli na to, že je možné regály kopírovat a ušetřit si tak práci při vytváření podobných regálů. Tuto funkci ocenili především účastníci, kteří pracovali ve velkých skladech. Stejně tak neměl žádný z účastníků problém s přidáním a úpravou rozměrů umístění v regálech.

Uživatelé v mapě našli možnost změny rozměrů objektu nebo zóny. Při úpravě velikosti za pomoci šipek, které jsou na pravé straně pole formuláře pro úpravu rozměru, upravovaný objekt či zóna se automaticky měnila. Jakmile účastník smazal hodnotu udávající rozměr pomocí klávesnice, pole se zneplatnilo, což bylo očekávané chování vzhledem k tomu, že nebyla zadána hodnota. Napsal-li však do pole účastník novou správnou hodnotu, pole nereagovalo a rozměr objektu či zóny se již neupravil. Účastník byl poté velmi zmaten a chvíli si nebyl jist, jak postupovat. Jednalo se o závažnou chybu v implementaci, kterou bylo nutné odstranit.

Při tvorbě překážky či zóny bylo pro uživatele matoucí, že se těmto objektům automaticky přiřadil název a až po jejich vytvoření ho mohli editovat. Libor uvedl, že by uvítal, kdyby se dal název změnit již při tvorbě objektu, jako tomu bylo u regálu.

Jinak nebyl pro účastníky problém vytvářet, upravovat a mazat překážky a zóny a práce s nimi byla srozumitelná. Dále také dobře pochopili funkcionalitu práce se skupinami umístění a lehce přidali do svých skladů umístění, která se nenachází v regálech.

S Oldřichem testování probíhalo dle scénářů TS3 a TS4. Těmito scénáři prošel bez větších problémů. Je však důležité dodat, že Oldřich byl obeznámen s rozhraním editoru mapy skladu, protože již prováděl jeho testování v rámci bakalářské práce Bc. Kopeckého a během implementace také připomínkoval nedostatky v mém řešení.

Marek, Tomáš, Libor a Veronika pracovali v reálných skladech, a proto testování probíhalo dle testovacího scénáře TS1. S Markem jsme navíc po tomto scénáři pokračovali scénářem TAS1. Vzhledem k tomu, že uživatelé během prvního scénáře často chybovali a museli své chyby napravovat, pokryli v rámci tohoto scénáře většinu testovacích případů. Tento test navíc zabral více času, než jsem předpokládal. Nemělo tak smysl pokračovat scénářem TS2, a proto jsem testování ukončil.

Obrazové a zvukové záznamy z testování nejsou, vzhledem k povaze osobních údajů, které obsahují (záznam hlasu, jméno, příjmení, věk účastníka atp.), součástí této práce a jsou dostupné na vyžádání u autora práce.

4.6 Výsledky testování

V této části shrnu výsledky, kterých jsem v provedeném testování použitelnosti dosáhl. Nejprve se zaměřím na data získaná z úvodního dotazníku, následně shrnu problémy v rozhraní, které jsem díky testování identifikoval, a nakonec se zaměřím na vyhodnocení testování a dotazníku SUS.

4.6.1 Úvodní dotazník

Z dat úvodního dotazníku vyplynulo, že všichni účastníci, kteří pracovali v reálných skladech, v těchto skladech působili nejen v roli vedoucího, ale také skladníka. Ve skladě, kde jsme test prováděli, pracoval účastník vždy více než půl roku. U třech účastníků to bylo dokonce více než dva roky, u jednoho z nich pak více než pět let.

Jeden z účastníků, který se zúčastnil testování, byl mladší než 30 let, tři uživatelé pak spadali do rozmezí 30–40 let a pouze jeden účastník byl starší 40 let.

Zajímavé bylo, že tři účastníci již v minulosti kreslili mapu skladu v programu Microsoft Excel. Jednalo se o účastníky, kteří pracovali ve středních a velkých skladech a mapa pro ně byla užitečná při orientaci ve skladě, případně plánování přestavby skladu. V jiných systémech se účastníci s mapou skladu nesetkali. Oldřich a Veronika měli s editorem mapy jistou zkušenost, protože testovali řešení, které v rámci své práce vytvořil Bc. Kopecký.

4.6.2 Problémy v rozhraní

V následujícím textu uvádím problémy, chyby a možnosti zlepšení použitelnosti rozhraní, které jsem v rámci testování identifikoval. Zmiňuji u nich také jejich závažnost a zda jsem v rámci této bakalářské práce uvedený problém vyřešil.

Variabilita úpravy jmenovacího systému

V editoru nebylo možné automaticky generovat názvy umístění v regálech, ve kterých měl jmenovací systém odlišné pořadí symbolů sloupce, patra a čísla v buňce. Tento problém byl vyřešen již v průběhu testování. To umožnilo ještě před samotným koncem testování ověřit, že jsou provedené úpravy správné a pro uživatele srozumitelné.

Závažnost: vysoká

Vyřešeno: ano, v šesté iteraci vývoje

Manuální úprava umístění v buňce regálu

V editoru nebylo možné přidávat do buňky regálu umístění s názvem, který neodpovídal jmenovacímu systému regálu. Je žádoucí, aby bylo možné do buňky regálu přidat umístění s jakýmkoliv názvem a tak umožnit tvorbu jakéhokoliv regálu. Tento problém byl vyřešen již v průběhu testování. To umožnilo ještě před samotným koncem testování ověřit, že jsou provedené úpravy správné a pro uživatele srozumitelné.

Závažnost: vysoká

Vyřešeno: ano, v šesté iteraci vývoje

Nefunkčnost formuláře pro změnu rozměrů objektu

Při úpravě rozměrů objektu nebo zóny nebylo při předchozím zneplatnění formuláře možné zadat nové hodnoty rozměrů. Po zneplatnění formuláře nereagovalo pole na první validní vstupní hodnotu a rozměr objektu či zóny se neupravil.

Závažnost: vysoká

Vyřešeno: ano, v šesté iteraci vývoje

Možnost změny názvu regálu v detailu regálu

Pro zvýšení použitelnosti rozhraní by bylo vhodné, aby v postranním panelu v zobrazení detailu regálu bylo možné změnit název regálu.

Závažnost: střední

Vyřešeno: ano, v šesté iteraci vývoje

Možnost zadání názvu při tvorbě překážky a zóny

Při tvorbě nové překážky a zóny nebylo možné nastavit jejich název. Místo toho musel uživatel nejprve vytvořit překážku či zónu a následně v postranním panelu detailu objektu změnit automaticky vygenerovaný název. Bylo by vhodnější negenerovat název automaticky, ale nechat uživatele nastavit vlastní název v procesu tvorby překážky nebo zóny.

Závažnost: střední

Vyřešeno: ano, v šesté iteraci vývoje

Zvýraznění buněk při kopírování objektů a zón

Při kopírovaní objektů a zón se v mapě nezvýrazňovala buňka, nad kterou se uživatel nacházel kurzorem myši, jako tomu bylo u všech podobných akcí (tvorba nového objektu, pohyb objektu).

Závažnost: střední

Vyřešeno: ano, v šesté iteraci vývoje

Dialogové okno potvrzení synchronizace umístění

Pro zvýšení použitelnosti rozhraní by bylo vhodné do procesu synchronizace umístění přidat dialogové okno, ve kterém by uživatel potvrdil, že chce skutečně vykonat vybranou akci. Důvodem je nevratnost akcí, které synchronizace provádí.

Závažnost: střední

Vyřešeno: ne

Zobrazení náhledu pozice regálu v detailu regálu

Pro zlepšení orientace při úpravě regálu by bylo vhodné, kdyby byl v zobrazení detailu regálu k dispozici panel s náhledem, ve kterém by byl zobrazen náhled pozice regálu v mapě.

Závažnost: nízká

Vyřešeno: ne

Prohledávání a filtrování textových přehledů objektů a oblastí

Pro zvýšení použitelnosti rozhraní by bylo vhodné přidat do textových přehledů objektů a oblastí, které se nachází v postranním panelu editoru, možnost prohledávat a filtrovat v seznamech objektů a oblastí.

Závažnost: nízká

Vyřešeno: ne

4.6.3 Vyhodnocení testování a dotazníku SUS

Jak uvádím v kapitole 4.2.3, testování jsem se rozhodl hodnotit jak kvalitativně, tak kvantitativně. Při závěrečném rozhovoru s účastníky většina z nich konstatovala, že rozhraní nebylo složité na použití a že se v něm velmi rychle dokázali zorientovat. Tomu také odpovídal průběh testování, během kterého se většina účastníků rychle naučila s editorem pracovat a vykonávat i složitější úkony jim nedělo příliš velký problém.

Tomu odpovídá také výsledek SUS dotazníku, ve kterém v průměru rozhraní dosáhlo skóre 75. Nejnižšího skóre 62.5 dosáhlo rozhraní u Marka. To se však dalo předpokládat, protože tento test probíhal jako první a některé chyby s vysokou závažností ještě nebyly v rozhraní opraveny. Nejvyššího skóre naopak bylo dosaženo u Veroniky, kde bylo dosaženo hodnoty 85.

Souhrnné dosažené skóre je vysoce nadprůměrné a potvrzuje, že je rozhraní dostatečně použitelné. Použitelnost jsem dále ještě zlepšil – v rámci šesté iterace jsem odstranil uvedené problémy identifikované v tomto testování.

Závěr

V rámci bakalářské práce jsem dokončil projekt mapy skladu ve skladovém systému Atlantis. Rovněž jsem do systému přidal možnost zobrazení heatmap vytížení skladových pozic v jednotlivých skladech.

Nejprve jsem provedl analýzu problematiky a rozpracovaného řešení, které v rámci své bakalářské práce realizoval Bc. Kopecký. Poté jsem zformuloval požadavky, které jsou na mapu skladu kladeny, a vytvořil model případů užití.

V rámci návrhu jsem nejdříve vybral vhodné kroky, které vedly k úspěšnému dokončení projektu. Zvolil jsem inkrementální metodiku vývoje a postupoval v šesti iteracích.

V rámci těchto iterací jsem postupně navrhl změny ve frontendu mapy skladu, které jsem následně také implementoval a otestoval. Jednalo se především o funkcionalitu spojenou s tvorbou a úpravou regálů, skupin umístění, překážek, zón a umístění v mapě. Věnoval jsem se ukládání a načítání mapy, zajistil řádné mapování umístění v mapě na skladová umístění a synchronizaci umístění mezi mapou a skladem. Zabýval jsem se automatickou generací heatmap pro zobrazení vytíženosti skladových pozic.

V rámci druhé iterace jsem vytvořil backend mapy skladu. Zde jsem provedl návrh doménového modelu a následně také vytvořil datový model a specifikaci API, které umožňovalo napojení na frontend.

Výsledné řešení jsem podrobil testování použitelnosti, v rámci kterého jsem identifikoval několik problémů v rozhraní, které jsem poté v poslední iteraci vývoje projektu odstranil. Zjistil jsem, že bylo dosaženo potřebné použitelnosti pro integraci mapy do systému Atlantis.

Výsledkem mé bakalářské práce je plně funkční editor mapy skladu, díky kterému mohou být v systému uložena veškerá data o pozici objektů, oblastí a umístění ve skladu. Věřím, že realizovaný editor bude mít vysoké praktické využití. Uvedená data lze totiž velmi dobře použít při optimalizaci důležitých procesů ve skladu. Na bakalářskou práci lze navázat tvorbou algoritmů, které by tyto optimalizace realizovaly.

Literatura

[1] Klabusayová, N.: *Logistika.* Vovcr.cz [online], 2019, ISBN 978-80-88418-15-3, [cit. 2023-03-09]. Dostupné z: https://www.vovcr.cz/odz/ekon/409

[2] Talár, D.: *Optimalizace nejčastějších procesů ve skladovém systému.* [online], Bakalářská práce, České Vysoké Učení Technické V Praze, Fakulta informačních technologií, 2020, [cit. 2023-03-09]. Dostupné z: https://dspace.cvut.cz/handle/10467/88290

[3] Urban, V.: *Frontend optimalizace skladových procesů systému Atlantis.* [online], Bakalářská práce, České Vysoké Učení Technické V Praze, Fakulta informačních technologií, 2022, [cit. 2023-03-09]. Dostupné z: https://dspace.cvut.cz/handle/10467/101593

[4] Kováčová, D.: *Backend optimalizace skladových procesů systému Atlantis.* [online], Bakalářská práce, České Vysoké Učení Technické V Praze, Fakulta informačních technologií, 2022, [cit. 2023-03-09]. Dostupné z: https://dspace.cvut.cz/handle/10467/101934

[5] Kopecký, V.: *Mapa skladu – frontend.* [online], Bakalářská práce, České Vysoké Učení Technické V Praze, Fakulta informačních technologií, 2020, [cit. 2023-03-09]. Dostupné z: https://dspace.cvut.cz/handle/10467/101655

[6] Malec, O.: *Frontend skladového systému.* [online], Diplomová práce, České Vysoké Učení Technické V Praze, Fakulta informačních technologií, 2020, [cit. 2023-03-09]. Dostupné z: https://dspace.cvut.cz/handle/10467/86593

[7] Kovář, P.: *Backend skladového systému.* [online], Diplomová práce, České Vysoké Učení Technické V Praze, Fakulta informačních technologií, 2019, [cit. 2023-03-09]. Dostupné z: https://dspace.cvut.cz/handle/10467/82591

[8] *Software Requirements*. tutorialspoint [online], [cit. 2023-06-14]. Dostupné z: https://www.tutorialspoint.com/software_engineering/software_requirements.htm

[9] Satzinger J. W., R. B. Jackson, S. D. Burd: *Systems Analysis and Design in a Changing World*. Cengage Learning, 2015, ISBN 978-13-05465-26-8, [cit. 2023-06-14].

[10] *Use Case Analysis: How to Identify Actors?* Visual Paradigm [online], 2022, [cit. 2023-06-04]. Dostupné z: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/how-to-identify-actors/

[11] Daly, N.: *What Is a Use Case?* Wrike [online], 2022, [cit. 2023-06-14]. Dostupné z: https://www.wrike.com/blog/what-is-a-use-case/

[12] Mlejnek, J.: *BI-SI1, Přednáška 3 - Analýza a sběr požadavků*. [online], České Vysoké Učení Technické V Praze, Fakulta informačních technologií, 2022, [cit. 2023-06-14]. Dostupné z: https://moodle-vyuka.cvut.cz/pluginfile.php/532096/mod_resource/content/7/03.prednaska.pdf

[13] Alena, B.: *Metodiky vývoje a údržby informačních systémů*. Praha: Grada, 2005, ISBN 80-247-1075-7, [cit. 2023-03-21]. Dostupné z: https://www.researchgate.net/publication/47042409_Metodiky_vyvoje_a_udrzby_informacnich_systemu_kategorizace_agilni_metodiky_vzory_pro_navrh_metodiky

[14] Kodytek, S.: *Úvod do metodologie vývoje softwaru*. itnetwork.cz [online], [cit. 2023-03-21]. Dostupné z: https://www.itnetwork.cz/navrh/metodiky/uvod-do-metodologie-vyvoje-softwaru

[15] Mlejnek, J.: *BI-SI1, Přednáška 11 - Metodiky vývoje*. [online], České Vysoké Učení Technické V Praze, Fakulta informačních technologií, 2022, [cit. 2023-03-21]. Dostupné z: https://moodle-vyuka.cvut.cz/pluginfile.php/532132/mod_resource/content/4/11.prednaska.pdf

[16] Centers for Medicare & Medicaid Services, O. o. I. S.: *Selecting a development approach*. [online], 2008, [cit. 2023-03-21]. Dostupné z: https://web.archive.org/web/20100331173931/http://www.cms.hhs.gov/SystemLifecycleFramework/Downloads/SelectingDevelopmentApproach.pdf

[17] Kitner, R.: *Posouzení kódu*. KITNER [online], 2021, [cit. 2023-06-28]. Dostupné z: https://kitner.cz/slovnik/posouzeni-kodu/

[18] *API Endpoints - What Are They? Why Do They Matter?* SmartBear [online], 2023, [cit. 2023-06-28]. Dostupné z: https://smartbear.com/learn/performance-monitoring/api-endpoints/

[19] Kassambara, A.: *How to Normalize and Standardize Data in R for Great Heatmap Visualization*. Data novia [online], 2023, [cit. 2023-06-27]. Dostupné z: https://www.datanovia.com/en/blog/how-to-normalize-and-standardize-data-in-r-for-great-heatmap-visualization/

[20] Zach: *How to Normalize Data Between 0 and 1*. Statology [online], 2021, [cit. 2023-06-27]. Dostupné z: https://www.statology.org/normalize-data-between-0-and-1/

[21] You, E.: *Vuex - Getting started*. [online], 2014, [cit. 2023-03-25]. Dostupné z: https://vuex.vuejs.org/guide/

[22] Nielsen, J.: *Usability 101: Introduction to Usability*. Nielsen Norman Group [online], 2012, [cit. 2023-06-03]. Dostupné z: https://www.nngroup.com/articles/usability-101-introduction-to-usability/

[23] Hlava, T.: *Dokumentace v testování: Testovací strategie – test plán*. Testování softwaru [online], 2011, [cit. 2023-06-03]. Dostupné z: http://testovanisoftwaru.cz/dokumentace-v-testovani/test-plan/

[24] *Usability testing: your 101 introduction*. hotjar [online], 2023, [cit. 2023-06-04]. Dostupné z: https://www.hotjar.com/usability-testing/

[25] Moran, K.: *Usability Testing 101*. Nielsen Norman Group [online], 2019, [cit. 2023-06-03]. Dostupné z: https://www.nngroup.com/articles/usability-testing-101/

[26] Sauro, J.: *Measuring Usability with the System Usability Scale (SUS)*. Measuring U [online], 2011, [cit. 2023-06-04]. Dostupné z: https://measuringu.com/sus/

[27] Laubheimer, P.: *Beyond the NPS: Measuring Perceived Usability with the SUS, NASA-TLX, and the Single Ease Question After Tasks and Usability Tests*. Nielsen Norman Group [online], 2018, [cit. 2023-06-04]. Dostupné z: https://www.nngroup.com/articles/measuring-perceived-usability/

[28] Nielsen, J.: *Why You Only Need to Test with 5 Users*. Nielsen Norman Group [online], 2000, [cit. 2023-06-04]. Dostupné z: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/

[29] Hlava, T.: *Dokumentace v testování: Test Case – Testovací případ*. Testování softwaru [online], 2011, [cit. 2023-06-03]. Dostupné z: http://testovanisoftwaru.cz/dokumentace-v-testovani/test-case/

[30] *Test Case*. java T point [online], [cit. 2023-06-03]. Dostupné z: https://www.javatpoint.com/test-case

[31] *Testovací dokumentace - plán, scénář, případ*. Testování software [online], [cit. 2023-06-03]. Dostupné z: http://test.swtestovani.cz/index.php?option=com_content&view=article&id=15:testovaci-dokumentace-plan-scena-pipad

[32] Hlava, T.: *Dokumentace v testování: Test Script – testovací scénář*. Testování softwaru [online], 2011, [cit. 2023-06-03]. Dostupné z: http://testovanisoftwaru.cz/dokumentace-v-testovani/test-script-testovaci-scenar/

Příloha A

Seznam použitých zkratek

API Application Programming Interface

BI-SP1 Softwarový týmový projekt 1

BI-SP2 Softwarový týmový projekt 2

FR Functional requirement

FURPS Functionality, Usability, Reliability, Performance, Security

IR Implementation requirement

PHP Hypertext Preprocessor

REST Representational State Transfer

SQL Structured Query Language

SR Supportability requirement

SUS System Usability Scale

TC Test case

TS Test scenario

UC Use case

UR Usability requirement

URI Uniform Resource Identifier

Příloha B

Testovací případy pro testování použitelnosti mapy skladu

TC1 Tvorba nové mapy skladu

Popis případu:	Uživatel v systému vytvoří pro zvolený sklad novou mapu skladu.
Počáteční podmínky:	Uživatel je přihlášen v systému Atlantis a má práva vedoucího skladu, pro který bude mapa vytvořena.
Kroky:	1. Uživatel přejde na stránku „Sklady“ zobrazující seznam skladů. 2. Uživatel vybere ze seznamu příslušný sklad a zobrazí jeho detail. 3. Uživatel na stránce „Detail skladu“ otevře kartu skladu.
Očekávaný výsledek:	Systém vytvoří mapu skladu a uživateli zobrazí editor mapy s prázdnou mapou.

TC2 Úprava rozměrů mapy skladu

Popis případu:	Uživatel upraví rozměry mapy skladu.
Počáteční podmínky:	Uživatel se nachází v editoru mapy skladu.
Vstupní data:	- nové rozměry mapy skladu
Kroky:	1. Uživatel upraví šířku a výšku mapy v příslušných polích panelu nástrojů.

Očekávaný výsledek: Systém změní rozměry mapy skladu a zobrazí upravenou mapu.

TC3 Tvorba nového regálu

Popis případu: Uživatel v mapě vytvoří nový regál.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu.

Vstupní data:
- pozice nového regálu v mapě
- název nového regálu

Kroky:
1. Uživatel v panelu nástrojů klikne na tlačítko „Nový regál“.
2. Uživatel vybere v mapě volnou pozici, na které se má regál nacházet.
3. Uživatel vyplní název nového regálu a potvrdí vytvoření regálu.

Očekávaný výsledek: Systém vytvoří regál s daným názvem na příslušné pozici a zobrazí ho v mapě skladu.

TC4 Tvorba nové skupiny umístění

Popis případu: Uživatel v mapě vytvoří novou skupinu umístění.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu.

Vstupní data:
- pozice nových umístění v mapě
- kódy nových umístění

Kroky:
1. Uživatel v panelu nástrojů klikne na tlačítko „Nová umístění“.
2. Uživatel vybere v mapě volnou pozici, na které se mají vytvořit nová umístění.
3. Uživatel přidá kódy umístění a potvrdí jejich vytvoření.

Očekávaný výsledek: Systém vytvoří na uživatelem zvolené pozici skupinu obsahující umístění s příslušnými kódy a zobrazí ji v mapě skladu.

TC5 Tvorba nové zóny

Popis případu: Uživatel v mapě vytvoří novou zónu.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu.

Vstupní data:
- pozice nové zóny v mapě
- typy nové zóny

Kroky:
1. Uživatel v panelu nástrojů klikne na tlačítko „Nová zóna“.
2. Uživatel vybere v mapě volnou pozici, na které se má zóna nacházet.
3. Uživatel vybere typy nové zóny a potvrdí vytvoření zóny.

Očekávaný výsledek: Systém vytvoří zónu s danými typy na příslušné pozici a zobrazí ji v mapě skladu.

TC6 Tvorba nové překážky

Popis případu: Uživatel v mapě vytvoří novou překážku.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu.

Vstupní data:
- pozice nové překážky v mapě

Kroky:
1. Uživatel v panelu nástrojů klikne na tlačítko „Nová překážka“.
2. Uživatel vybere v mapě volnou pozici, na které se má překážka nacházet.

Očekávaný výsledek: Systém vytvoří na zadané pozici překážku a zobrazí ji v mapě skladu.

TC7 Změna počtu sloupců a pater regálu

Popis případu: Uživatel v mapě změní, kolik sloupců a pater se nachází v určitém regálu.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu, ve které se vyskytuje alespoň jeden regál.

Vstupní data:
- název upravovaného regálu
- nový počet sloupců upravovaného regálu

- nový počet pater upravovaného regálu

Kroky:
1. Uživatel v mapě klikne na příslušný regál a tím provede jeho výběr.
2. Uživatel v panelu nástrojů klikne na tlačítko „Editovat“ a zobrazí tím detail regálu.
3. Uživatel upraví počet sloupců a pater regálu v příslušných polích panelu nástrojů

Očekávaný výsledek: Systém upraví počet sloupců a pater v příslušném regálu a zobrazí upravený regál v okně detailu regálu.

TC8 Změna jmenovacího systému regálu

Popis případu: Uživatel v mapě změní způsob, jakým jsou v určitém regálu generovány kódy umístění.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jeden regál.

Vstupní data:
- název upravovaného regálu
- počáteční symbol a směr pojmenovávání sloupců
- počáteční symbol a směr pojmenovávání pater
- oddělovače užité v kódu umístění

Kroky:
1. Uživatel v mapě klikne na příslušný regál a tím provede jeho výběr.
2. Uživatel v panelu nástrojů klikne na tlačítko „Editovat“ a zobrazí tím detail regálu.
3. Uživatel v postranním panelu upraví na kartě „Jmenovací systém“ v příslušných polích způsob vytváření kódů umístění.

Očekávaný výsledek: Systém upraví kódy umístění v příslušném regálu a zobrazí je v okně detailu regálu.

TC9 Změna počtu umístění v buňkách regálu

Popis případu: Uživatel v mapě změní počet umístění v několika buňkách, které se nachází v určitém regálu.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jeden regál.

Vstupní data:
- název upravovaného regálu

- pozice buněk, ve kterých má být změněn počet umístění a hodnoty, na které má být počet nastaven

Kroky:

1. Uživatel v mapě klikne na příslušný regál a tím provede jeho výběr.
2. Uživatel v panelu nástrojů klikne na tlačítko „Editovat“ a zobrazí tím detail regálu.
3. Uživatel v okně detailu regálu vybere buňky regálu, ve kterých má upravit počet umístění, která obsahují.
4. Uživatel nastaví v příslušném poli v postranním panelu hodnotu počtu umístění na buňku.

Očekávaný výsledek: Systém upraví počty umístění v buňkách příslušného regálu a v okně detailu regálu zobrazí upravený regál.

TC10 Přiřazení nových rozměrů umístění buňkám regálu

Popis případu: Uživatel v mapě vytvoří nové rozměry umístění a přiřadí je několika buňkám, které se nachází v určitém regálu.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jeden regál.

Vstupní data:

- název upravovaného regálu
- nové rozměry umístění – výška, šířka, délka

Kroky:

1. Uživatel v mapě klikne na příslušný regál a tím provede jeho výběr.
2. Uživatel v panelu nástrojů klikne na tlačítko „Editovat“ a zobrazí tím detail regálu.
3. Uživatel v okně detailu regálu vybere buňky regálu, ve kterých má upravit počet umístění, která obsahují.
4. Uživatel v postranním panelu klikne na kartu „Rozměry“ a zobrazí tak danou kartu.
5. Uživatel klikne v kartě rozměry na tlačítko „Nový rozměr“.
6. Uživatel v postranním panelu upraví v příslušných polích parametry nového rozměru umístění.
7. Uživatel v postranním panelu upraví dle svého uvážení barvu, jakou budou rozměry reprezentovány.
8. Uživatel v postranním panelu v seznamu rozměrů klikne u nově vytvořeného rozměru na tlačítko ”Přiřadit”.

Očekávaný výsledek: Systém upraví rozměry umístění v buňkách příslušného regálu a v okně detailu regálu zobrazí upravený regál.

TC11 Odstranění rozměrů umístění z mapy skladu

Popis případu:	Uživatel v mapě odstraní konkrétní rozměry umístění.
Počáteční podmínky:	Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jeden regál.
Vstupní data:	- rozměry umístění k odstranění – výška, šířka, délka, barva
Kroky:	1. Uživatel v mapě klikne na jeden z regálů a tím provede jeho výběr. 2. Uživatel v panelu nástrojů klikne na tlačítko „Editovat“ a zobrazí tím detail regálu. 3. Uživatel v postranním panelu klikne na kartu „Rozměry“ a zobrazí tak danou kartu. 4. Uživatel v postranním panelu v seznamu rozměrů klikne u rozměrů, které mají být odstraněny, na červené tlačítko s ikonou koše.
Očekávaný výsledek:	Systém odstraní rozměry z mapy skladu a nahradí je v buňkách regálů, ke kterým byly přiřazeny, výchozími rozměry.

TC12 Změna polohy objektu či zóny

Popis případu:	Uživatel v mapě přesune regál, skupinu umístění, překážku nebo zónu na jinou pozici, než na které se aktuálně nachází.
Počáteční podmínky:	Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jeden objekt nebo zóna.
Vstupní data:	- nová pozice objektu či zóny
Kroky:	1. Uživatel v mapě klikne na jeden z objektů či zón. 2. Uživatel v panelu nástrojů klikne na tlačítko „Pohyb“. 3. Uživatel vybere volnou pozici v mapě, na kterou se má vybraný objekt či zóna přesunout.
Očekávaný výsledek:	Systém přesune objekt či zónu na novou pozici a korektně zobrazí přesunutý objekt v mapě skladu.

TC13 Rotace objektu či zóny

Popis případu:	Uživatel v mapě otočí regál, překážku nebo zónu z vertikální do horizontální polohy či naopak.

Počáteční podmínky:	Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jeden regál, překážka nebo zóna.
Vstupní data:	- název regálu, překážky či zóny, která má být otočena
Kroky:	1. Uživatel v mapě klikne na regál, překážku či zónu a dojde tak k jejímu výběru. 2. Uživatel v panelu nástrojů klikne na tlačítko „Rotace“.
Očekávaný výsledek:	Systém otočí vybraný objekt nebo zónu o 90°a zobrazí otočený objekt v mapě skladu.

TC14 Tvorba kopie objektu či zóny

Popis případu:	Uživatel v mapě vytvoří kopii regálu, překážky nebo zóny.
Počáteční podmínky:	Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jeden regál, překážka nebo zóna.
Vstupní data:	- název regálu, překážky či zóny, která má být zkopírována - pozice nově vytvořené kopie objektu či zóny v mapě
Kroky:	1. Uživatel v mapě klikne na regál, překážku či zónu a dojde tak k jejímu výběru. 2. Uživatel v panelu nástrojů klikne na tlačítko „Kopírovat“. 3. Uživatel vybere volnou pozici v mapě, na kterou se má vybraný objekt či zóna zkopírovat.
Očekávaný výsledek:	Systém vytvoří v mapě kopii vybraného objektu nebo zóny a přidá ji do mapy na definovanou pozici, kde ji také zobrazí.

TC15 Mazání objektu či zóny

Popis případu:	Uživatel v mapě smaže regál, skupinu umístění, překážku nebo zónu.
Počáteční podmínky:	Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jeden regál, skupina umístění, překážka nebo zóna.
Vstupní data:	- název objektu či zóny, která má být smazána
Kroky:	1. Uživatel v mapě klikne na regál, překážku či zónu a dojde tak k jejímu výběru. 2. Uživatel v panelu nástrojů klikne na tlačítko „Smazat“.
Očekávaný výsledek:	Systém smaže z mapy daný objekt či zónu a zobrazí mapu skladu, ve které se daný objekt nenachází.

TC16 Změna názvu objektu či zóny

Popis případu: Uživatel změní název regálu, překážky nebo zóny.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jeden regál, překážka nebo zóna.

Vstupní data:
- název regálu, překážky či zóny, která má být přejmenována
- nový název

Kroky:
1. Uživatel v mapě klikne na regál, překážku či zónu a dojde tak k jejímu výběru.
2. Uživatel v postranním panelu klikne na tlačítko „Přejmenovat“.
3. Uživatel vybere volnou pozici v mapě, na kterou se má vybraný objekt či zóna zkopírovat.
4. V zobrazeném poli uživatel změní název objektu či zóny a změnu názvu potvrdí.

Očekávaný výsledek: Systém přejmenuje objekt či zónu a zobrazí nový název v postranním panelu. V případě, že byl upraven název regálu, zobrazí systém nový název také v mapě a změní také názvy umístění v tomto regálu.

TC17 Změna rozměrů objektu či zóny

Popis případu: Uživatel změní rozměry regálu, překážky nebo zóny.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jeden regál, překážka nebo zóna.

Vstupní data:
- název regálu, překážky či zóny, u které mají být změněny rozměry
- nové rozměry objektu či zóny

Kroky:
1. Uživatel v mapě klikne na regál, překážku či zónu a dojde tak k jejímu výběru.
2. Uživatel upraví rozměry vybraného objektu nebo zóny v příslušných polích postranního panelu.

Očekávaný výsledek: Systém změní rozměry objektu či zóny a zobrazí upravený objekt v mapě skladu.

TC18 Změna typů zóny

Popis případu:	Uživatel změní typy konkrétní zóny.
Počáteční podmínky:	Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jedna zóna.
Vstupní data:	- název zóny, u které mají být upraveny typy - nové typy zóny
Kroky:	1. Uživatel v mapě klikne příslušnou zónu a dojde tak k jejímu výběru. 2. Uživatel upraví typy zóny v příslušných polích postranního panelu.
Očekávaný výsledek:	Systém změní typy zóny a zobrazí upravené typy v postranním panelu.

TC19 Přidání umístění do skupiny umístění

Popis případu:	Uživatel přidá do skupiny umístění nové umístění.
Počáteční podmínky:	Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jedna skupina umístění.
Vstupní data:	- pozice skupiny umístění, do které má být přidáno nové umístění - kód umístění, které má být do skupiny přidáno
Kroky:	1. Uživatel v mapě klikne na příslušnou skupinu umístění a dojde tak k jejímu výběru. 2. Uživatel zadá do příslušného pole v postranním panelu kód vytvářeného umístění a potvrdí jeho přidání.
Očekávaný výsledek:	Systém přidá do skupiny umístění nové umístění a zobrazí přidané umístění ve výpisu v postranním panelu.

TC20 Odebrání umístění ze skupiny umístění

Popis případu:	Uživatel odebere ze skupiny umístění konkrétní umístění.
Počáteční podmínky:	Uživatel se nachází v editoru mapy skladu, ve které se nachází alespoň jedna skupina umístění obsahující alespoň jedno umístění.

Vstupní data:
- pozice skupiny umístění, ze které má být umístění odebráno
- kód umístění, které má být ze skupiny odebráno

Kroky:
1. Uživatel v mapě klikne na příslušnou skupinu umístění a dojde tak k jejímu výběru.
2. Uživatel klikne ve výpisu umístění skupiny v postranním panelu u umístění, které chce odebrat, na ikonu křížku.

Očekávaný výsledek: Systém odebere ze skupiny umístění dané umístění a odebrané umístění se již nenachází ve výpisu v postranním panelu.

TC21 Uložení mapy skladu

Popis případu: Uživatel uloží mapu skladu.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu, ve kterém se nachází mapa s neuloženými změnami.

Kroky:
1. Uživatel klikne na tlačítko „Uložit“.

Očekávaný výsledek: Systém uloží mapu skladu a indikuje dokončení tohoto úkolu změnou stavu tlačítka „Uložit“ na „Mapa uložena“.

TC22 Synchronizace umístění – tvorba skladových umístění

Popis případu: Uživatel provede automatickou tvorbu skladových umístění, která mají vytvořená odpovídající mapová umístění, ale v systému ještě nejsou evidována.

Počáteční podmínky: Uživatel se nachází v editoru mapy skladu, ve kterém je zobrazena mapa neobsahující neuložené změny. V této mapě se nachází mapová umístění, která nemají ve skladu evidována odpovídající skladová umístění.

Kroky:
1. Uživatel klikne na tlačítko „Synchronizovat umístění“.
2. Uživatel klikne na tlačítko „Automaticky vytvořit odpovídající umístění ve skladě“.

Očekávaný výsledek: Systém vytvoří ve skladě skladová umístění odpovídající mapovým umístěním, která do této chvíle nebyla mapována. V mapě se nenachází mapová umístění, kterým by neodpovídalo žádné skladové umístění.

TC23 Synchronizace umístění – odstranění skladových umístění

Popis případu:	Uživatel provede automatické mazání skladových umístění, která nemají v mapě vytvořená odpovídající mapová umístění.
Počáteční podmínky:	Uživatel se nachází v editoru mapy skladu, ve kterém je zobrazena mapa neobsahující neuložené změny. V této mapě nejsou reprezentována všechna skladová umístění.
Kroky:	1. Uživatel klikne na tlačítko „Synchronizovat umístění“. 2. Uživatel klikne na tlačítko „Automaticky odstranit nenamapovaná umístění ve skladě“.
Očekávaný výsledek:	Systém odstraní ve skladě skladová umístění, která nemají v mapě příslušná mapová umístění. Ve skladě se nenachází skladová umístění, kterým by neodpovídalo žádné mapové umístění.

Příloha C

Obsah přiloženého média

- `readme.txt` ..... stručný popis obsahu média
- `text`
  - `thesis.pdf` ..... text práce ve formátu PDF
  - `thesis.tex` ..... text práce ve formátu LaTeX
- `screenshots` ..... snímky obrazovek výsledného řešení