Slovenská technická univerzita

Fakulta informatiky a informačných technológií

Ilkovičova 2, 842 16 Bratislava 4

Inžinierske dielo

Tím 13

Predmet:	Tímový projekt
Akademický rok:	2021/2022
Vedúci tímu:	Ing. Rastislav Bencel, PhD.
Členovia tímu:	Bc. Adam Gajdoš Bc. Marián Jarník Bc. Ivan Jatz Bc. Jozef Juraško Bc. Róbert Karpiel Bc. Samuel Škultéty

Slovník pojmov

Heslo	**Popis**
Balík (package)	Súhrnné označenie pre zásielky a kargá
ID	Identifikátor
Kargo (cargo)	Skupina zásielok zoskupených za účelom prepravy
Manipulant (handler)	Výrobca/dopravca
Podvýrobok (subitem)	Výrobok, ktorý je súčasťou iného väčšieho výrobku (napr. podvýrobky motor, kolesá a karoséria vytvoria auto)
Produkt (product)	Všeobecný popis nejakého tovaru, ktorý výrobca vyrába
S-Chain	Projekt, na ktorom ako tím spoločne pracujeme
Výrobok (item)	Jedinečný, konkrétny kus výrobku, ktorý výrobca vyrobil
Zákazník (client)	Koncový používateľ (spotrebiteľ)
Zásielka (shipment)	Skupina výrobkov zoskupených za účelom prepravy

Obsah

1 Big Picture

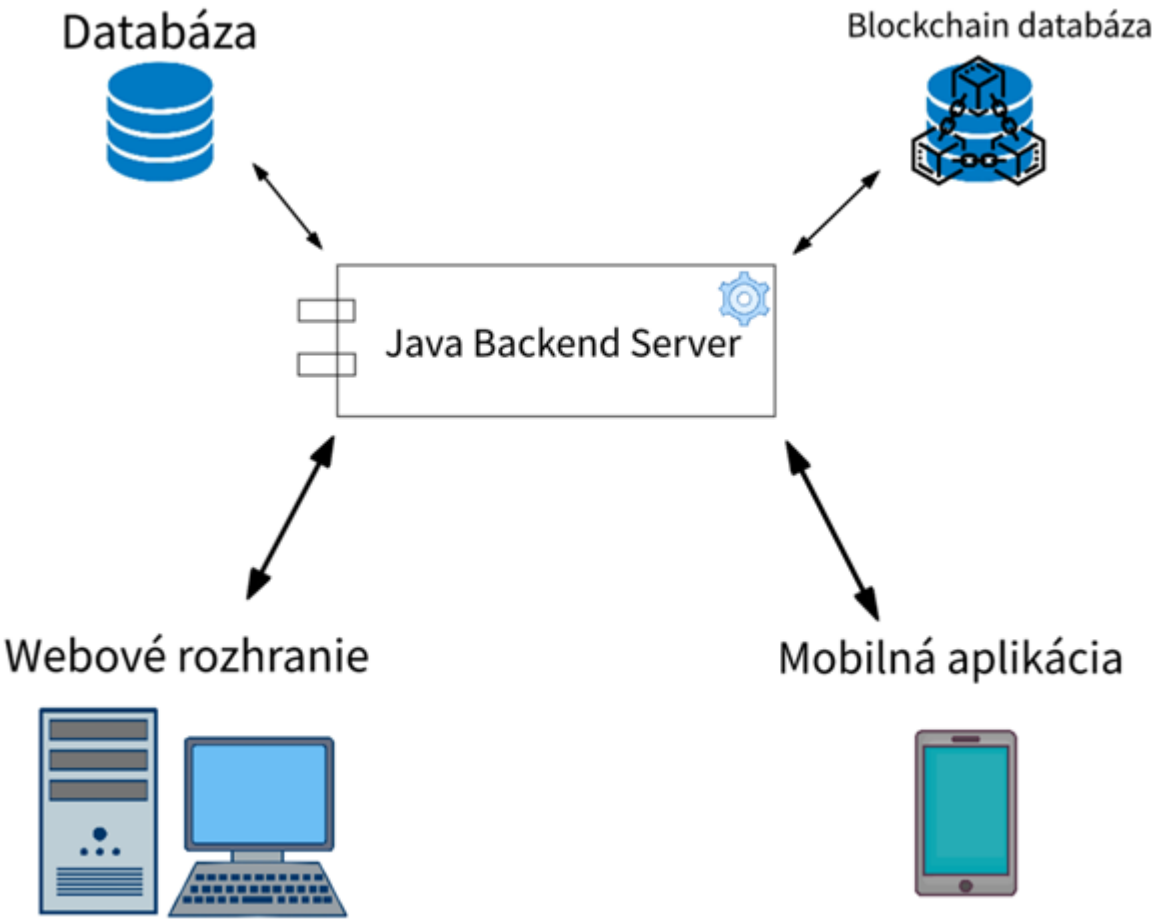
1.1 Úvod

V tomto dokumente sa nachádza opis realizovaného projektu S-Chain vytvoreného v rámci predmetu Tímový projekt v rokoch 2021 – 2022. Tento projekt je pokračovaním tímového projektu z roku 2020 – 2021 (1).

Dokument obsahuje technické detaily a opisuje výhradne projekt. Nachádza sa tu pohľad na štruktúru projektu, použité technológie a spôsob ich aplikácie. Na úvod popisujeme celkový prehľad systému a ciele projektu. Na tieto kapitoly nadväzuje analytická časť, nasledovaná návrhom a implementáciou. Dokument je koncipovaný tak, aby boli v ňom obsiahnuté informácie prezentované úplne, ale zároveň plynulo.

1.2 Celkový pohľad na systém

V tejto krátkej kapitole predstavíme technické detaily nášho riešenia v základnom prehľade. Detailnejší opis bude v neskorších kapitolách prezentovaný v diagramoch, ktoré lepšie ukážu a vyzdvihnú vybrané časti systému. V tejto časti uvedieme a popíšeme rozdelenie častí systému, ktoré je zobrazené na obrázku nižšie (obrázok 1) v zjednodušenej podobe.



Obrázok 1 – Návrh prepojenia komponentov architektúry

Naše riešenie má za úlohu poskytnúť informácie o procesoch výroby a prepravy, ktorými výrobok prechádza predtým, než dorazí na pulty v predajniach. Výrobcovi umožňuje evidovať svoje produkty a následne vytvárať výrobky, ku ktorým je možné pristupovať pomocou jedinečného identifikátora a aktualizovať ich. Všetky aktualizácie so sebou nesú dodatočné informácie ako pre výrobcu, tak (neskôr) aj pre zákazníka.

Výrobca zároveň môže svoj vyrobený a expedovaný tovar sledovať, než sa dopraví na miesto určenia. Dopravca, ktorý bude takisto využívať našu aplikáciu, zaznamenáva dôležité udalosti súvisiace s manipuláciou s tovarom, napr. jeho naloženie alebo vyloženie. Tieto záznamy sa spolu s informáciami z výrobného procesu nakoniec na požiadanie zobrazia

koncovému zákazníkovi po naskenovaní jedinečného identifikátora. Zákazník uvidí, akou cestou od výroby až po doručenie do obchodu prešiel tovar, ktorý má práve v rukách. Záznamy budú okrem dátumov a časov aktualizácií obsahovať aj krátky popis, ktorý o nich bude odhaľovať bližšie informácie.

Zákazník i samotný výrobca môžu vidieť zmeny, ktorými si teda tovar prechádzal. Lenže takéto zobrazovanie zmien nič neznamená, ak sa niektorý z používateľov systému nemôže spoľahnúť na dôveryhodnosť poskytovaných informácií. Ako je možné predísť pochybnostiam o neoprávnených zásahoch do záznamov či zavádzaniu o tom, čo sa s tovarom dialo a akými miestami prechádzal? Riešenie na tento problém poskytuje blockchain. Ten ako špeciálna databáza umožňuje ukladanie záznamov bez možnosti ich neskoršej zmeny.

Z analýzy štruktúry systému vytvoreného predchádzajúcim tímom sme zistili, že sa skladá z viacerých komponentov. Základom je serverové sídlo, ktoré je vytvorené pomocou rámca Java Spring Boot a nástroja Gradle. Tento server spravuje PostgreSQL databázu, v ktorej sa ukladajú dáta, ktoré nie sú nevyhnutné na zaručenie transparentnosti (napríklad prihlasovacie údaje jednotlivých používateľov). Táto databáza bude podľa potreby synchronizovaná s blockchain databázou. K rozhodnutiu o využití databázy PostgreSQL namiesto ukladania všetkých údajov do blockchainu sa pristúpilo primárne z dôvodu zabezpečenia efektivity, nakoľko v databáze tohto typu nevieme prepisovať už existujúce uložené záznamy, čo môže databázu rozšíriť do veľkých rozmerov. Blockchain sieť sa navyše skladá z viacerých uzlov, ktoré sa musia pri povolení zápisu zhodnúť. To znamená, že zápis môže trvať omnoho dlhšie než zápis do klasickej SQL databázy.

Zvolenou implementáciou technológie blockchain je rámec IBM Hyperledger (konkrétne Hyperledger Fabric), ktorý ponúka prístup do blockchain databázy pomocou autentifikácie. Nie je vhodné použiť verejný blockchain, pretože v ňom budú evidované aj informácie týkajúce sa výroby a prepravy výrobkov. Je teda nutné umožniť pri prístupe k dátam a ich aktualizáciách určitú mieru súkromia.

Na server sa rozličné zariadenia používateľov (najmä koncových zákazníkov) budú pripájať primárne pomocou prehliadača, preto je nutné vytvoriť pre náš produkt webové rozhranie. Z pohľadu spracovania webovej verzie je použitý rámec Vue.js 2.6.12 s využitím Bootstrap-vue vo verzii 2.17.3 (všetky závislosti a ich presné verzie sú uvedené v prílohe A). Pokračujeme aj vo vývoji mobilnej aplikácie v jazyku Kotlin, ktorú vytvoril predchádzajúci tím. Vzhľadom na zistené rozličné možnosti využitia mobilnej aplikácie rôznymi skupinami používateľov je vytvorená aj druhá, samostatná mobilná aplikácia určená hlavne pre používateľov zainteresovaných v preprave produktu – zamestnancov výrobných a dopravných spoločností.

1.3 Ciele projektu na zimný semester

Keďže pokračujeme v práci na existujúcom projekte (1), ktorého základná funkcionalita už je implementovaná, naším prvým cieľom je zoznámenie sa s existujúcim kódom, aby sme s ním dokázali v rámci nášho projektu efektívne pracovať. Medzi ďalšie ciele určené na obdobie zimného semestra patrí napríklad vylepšenie funkcionality skenovania QR kódov, ktorými sú označené sledované produkty. Takisto sme sa po analýze existujúcej verzie projektu rozhodli vylepšiť aj proces registrácie, do ktorého implementujeme overovanie e-mailových adries pre autentifikáciu registrácie. Pre uľahčenie našej práce sme sa takisto rozhodli upraviť kód serveru tak, aby ho bolo možné využiť technológiu Docker a zefektívniť tak nasadzovanie jednotlivých častí projektu.

Stanovili sme si tiež niekoľko dlhodobých cieľov, primárne na základe pripomienok od externého partnera projektu, spoločnosti OLTIS. Medzi tieto ciele patrí napríklad zmena systému sledovania produktov, ktoré momentálne používa výhradne GPS dáta, ktoré chceme doplniť kódmi SR 70 a UN/LOCODE. Tieto kódy sú zaužívaným štandardom v železničnej doprave.

Ďalším z cieľov na zimný semester je internacionalizácia aplikácií, ktorú si vyžiadala spoločnosť OLTIS, keďže budú využívané prepravcami po celom svete.

K dlhodobejším cieľom projektu patrí aj vytvorenie samostatnej aplikácie pre manipulantov, v ktorej bude možné vytvárať nové produkty, výrobky a zásielky, aktualizovať ich stav a pozíciu a vykonávať ďalšie úkony týkajúce sa výroby a prepravy tovaru. Implementovanie tohto cieľa bude pokračovať aj v letnom semestri.

1.4 Ciele projektu na letný semester

Pre letný semester boli stanovené nasledujúce ciele:

- sledovanie výrobkov a jeho distribútorov, resp. subjekty, ktoré prichádzajú s výrobkom do styku. Týmto chceme zabezpečiť vyššiu ochranu voči falšovaniu
- spracovanie kategorizácie výrobkov, ktorá bude slúžiť na odhaľovanie nezhody vo výrobkoch
- kontrola zacyklenia produktového stromu pri pridávaní podproduktov; produkt nemôže byť svojím vlastným podproduktom, a to ani prostredníctvom iného podproduktu (cyklus typu $A \rightarrow B \rightarrow A$)
- vytvorenie vernostného systému, ktorý bude odmeňovať koncového zákazníka za používanie aplikácie
- rozšírenie spracovania QR kódov o nové druhy týchto kódov pre zásielky a kargá
- optimalizácia dopytov na blockchainovú databázu, ktorá by slúžila k zníženiu časovej náročnosti tvorby nových výrobkov
- spracovanie programovej dokumentácie; po splnení všetkých cieľov je potrebné zdokumentovať projekt, podľa ktorého bude následne používaný

2 Moduly systému

2.1 Analýza

Ako už bolo spomenuté vyššie, naša práca je pokračovaním minuloročného projektu (1), preto pri voľbe používaných programovacích jazykov a softvérových nástrojov vychádzame z existujúcich základov. Z tohto dôvodu ich nebudeme v našej práci podrobnejšie analyzovať, ale zameriame sa hlavne na analýzu existujúceho riešenia, primárne častí, do ktorých budeme implementovať novú funkcionalitu.

2.1.1 Existujúce riešenie

Existujúce riešenie má podobu síce funkčného, no neúplného softvéru na sledovanie polohy produktov. Pozostáva z piatich modulov: server, blockchainová databáza, PostgreSQL databáza, webové rozhranie a mobilná aplikácia pre zákazníkov. Server je naprogramovaný v jazyku Java, pričom sa používa framework Java Spring Boot. Na spravovanie blockchainu je využitý IBM Hyperledger Fabric. Webové rozhranie je napísané v JavaScripte pomocou frameworku Vue.js. Mobilná aplikácia je vytvorená v jazyku Kotlin.

Viaceré časti systému obsahujú technické, dizajnové a iné nedostatky, ktoré zväčša opíšeme v kapitole Návrh a následne aj navrhneme riešenie. Do analýzy sme sa rozhodli uviesť len zmenu sledovania polohy výrobkov. V súčasnosti sa na to používajú GPS súradnice, avšak z dôvodu, že náš projekt je primárne určený pre železničnú dopravu, je potrebná transformácia na kódy SR 70, prípadne UN/LOCODE.

2.1.2 SR 70

Predpis SR 70 je číselník dopravných bodov železničnej siete Slovenskej a Českej republiky. Jednotlivé krajiny majú odlišné metodiky kódovania týchto bodov, avšak pre obe platí, že stanice, zástavky, nákladiská a iné dopravne sú číslované šesťmiestnym číslom. Tieto čísla sú

unikátne, neexistujú teda dve stanice s rovnakým číselným kódom. Prihraničné stanice majú rovnaký číselný kód na Slovensku aj v Česku, avšak tu ide o rovnakú stanicu (2) (3).

Zostavenie evidenčného čísla má na Slovensku a v Česku odlišné metodiky, avšak posledná číslica vyjadruje v oboch to isté. Ide o kontrolnú číslicu, ktorá sa vypočíta na základe predchádzajúcich piatich čísel výpočtom, ktorý zodpovedá medzinárodne platnej vyhláške UIC 913 (4).

2.1.3 UN/LOCODE

UN/LOCODE je kódovacia schéma pre obchod a dopravu. Prideľuje kódy rozličným miestam používaných pri doprave tovaru – prístavom, železničným staniciam, cestným terminálom, letiskám, poštám a podobne (5).

Štruktúra kódu pozostáva z piatich znakov, pričom prvé dva označujú krajinu a ďalšie tri miesto v konkrétnej krajine. Kódy zväčša pozostávajú z písmen, avšak v prípade potreby väčšieho množstva kombinácií je možné použiť aj čísla 2 až 9. Čísla 0 a 1 sú zakázané, aby nedošlo k zámene s písmenami O a I.

Schéma UN/LOCODE má oproti SR 70 viacero výhod. Predovšetkým ide o medzinárodné kódovanie, a teda nie je obmedzené len na použitie v stredoeurópskej oblasti. Taktiež sú tieto kódy používané vo všetkých oblastiach dopravy, na rozdiel od SR 70 používaných len v rámci železníc.

Nakoľko je však náš projekt primárne zameraný na železničnú dopravu v Českej a Slovenskej republike, v ktorých je číselník SR 70 používaný častejšie, do nášho systému budeme implementovať sledovanie polohy predovšetkým pomocou tohto číselníka.

2.2 Návrh

V tejto kapitole opíšeme návrh nových funkcionalít, ktoré vyplynuli z minuloročného projektu (1). Primárne chceme implementovať pripomienky z minulého roku a aktuálne požiadavky firmy OLTIS. Medzi naše hlavné priority patria:

- vytvorenie dvoch mobilných aplikácií – pre koncových používateľov (zákazníkov) a manipulantov
- pridanie vernostného systému za účelom predĺženia času používateľa stráveného v aplikácii a odmenenia stálych používateľov
- implementácia lokalizácie zásielok pomocou SR 70 a UN/LOCODE kódov
- internacionalizácia systému – podpora viacjazyčnosti v mobilnej a webovej aplikácii
- úprava registrácie – umožnenie registrácie cez mobilnú aplikáciu, overenie e-mailových adries registrovaných používateľov
- úprava generovania a spracovania QR kódov – spracovanie rôznych verzií špecifikácie QR kódov, vizuálne odlíšenie QR kódov systému S-Chain
- úprava webovej stránky a zákazníckej mobilnej aplikácie – zlepšenie prehľadnosti, dostupnosti a funkčnosti pomocou dizajnových zmien
- optimalizácia blockchain databázy – zrýchlenie zápisu nových záznamov a načítanie existujúcich záznamov

2.2.1 Vernostný systém

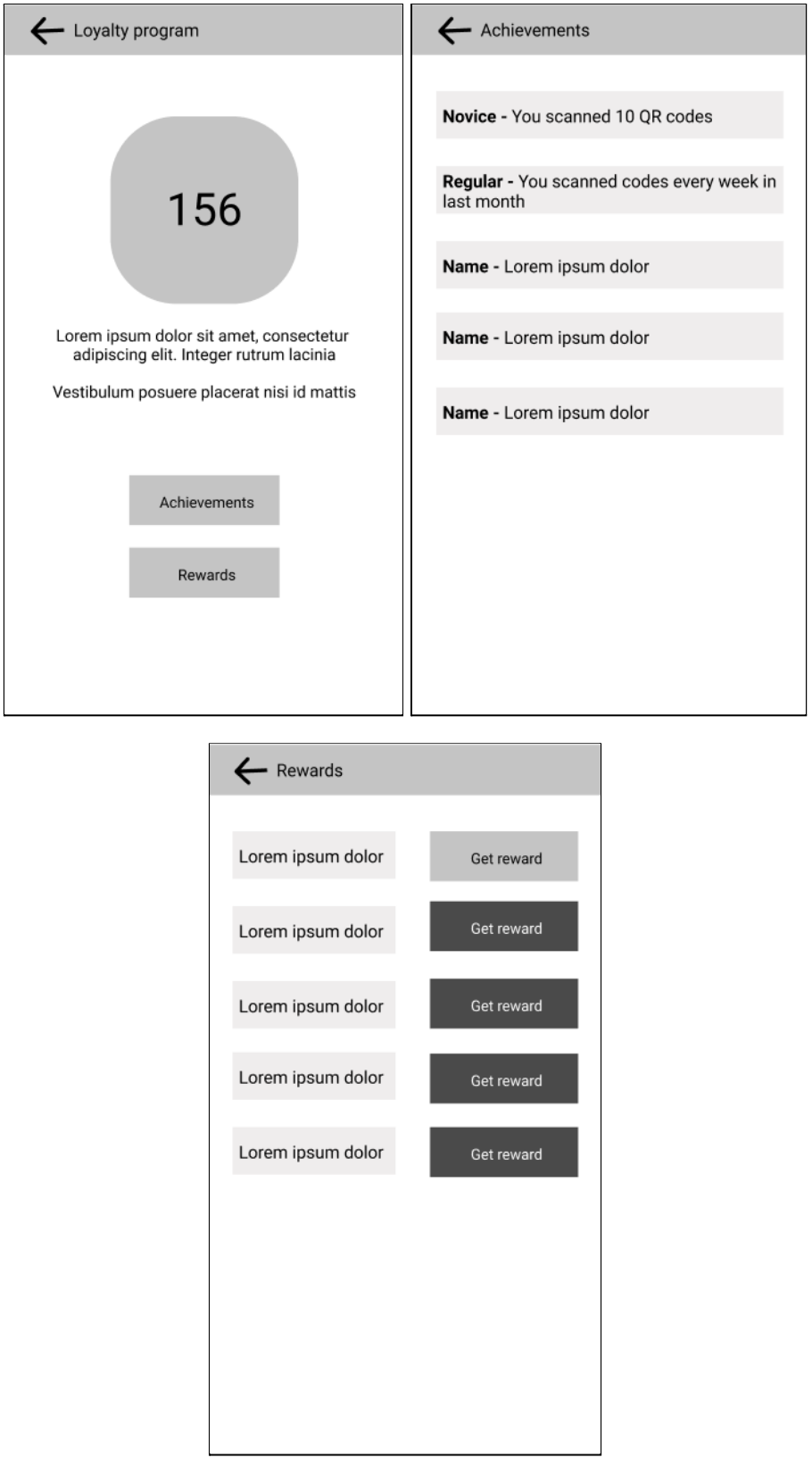
Na základe existujúcich požiadaviek chceme v systéme podporovať vernostný program. Ten bude slúžiť najmä koncovým zákazníkom (používateľská rola client), ktorí budú za aktívne používanie mobilnej aplikácie odmeňovaní formou bodov, ktoré bude následne možné vymeniť za rôzne odmeny. Týmto modelom sa budeme snažiť prilákať nových používateľov a tiež motivovať existujúcu komunitu k pravidelnému používaniu aplikácie.

Kvôli implementácii tejto súčasti systému prerobíme generovanie QR kódov pridaním loga S-Chain do samotných kódov a pridaním informácií o verzii QR kódu. Taktiež prenášaný obsah bude ďalej spracovaný na strane backendu (pozri kapitolu 2.2.5).

Pre pripočítanie vernostných bodov je potrebné, aby bol používateľ v systéme registrovaný a prihlásený. Vernostné body získa naskenovaním validného QR kódu produktu. Aplikácia vyhodnotí, či prihlásený, a teda overený, používateľ naskenoval QR kód s platným hashom výrobku. Validácia hashu bude prebiehať na strane backendu. Ak sú všetky podmienky splnené, aplikácia používateľovi priráta vernostné body a uloží ich do jeho "peňaženky".

V rámci implementácie preto musíme rozšíriť tabuľku produktu o jedinečný alfanumerický hash. Potrebujeme tiež upraviť tabuľku account, do ktorej pridáme atribúty potrebné pre evidenciu stavu vernostných bodov získaných koncovými zákazníkmi. V neposlednom rade vytvoríme tabuľku, ktorá bude dokumentovať dosiahnuté úspechy používateľa.

Z pohľadu mobilnej a webovej aplikácie bude potrebné vytvoriť prívetivé používateľské rozhranie, ktorého hlavnou úlohou bude zaujať zákazníka a prehľadne zobraziť súčasnú bilanciu vernostných bodov v jeho virtuálnej peňaženke a dosiahnuté úspechy. Jeho návrhy je možné vidieť na nasledujúcich obrázkoch (obrázok 2, 3, 4).



Obrázok 2, 3, 4 – Low fidelity návrhy obrazoviek mobilnej aplikácie

2.2.2 Úprava lokalizácie zásielok pomocou kódov SR 70 a UN/LOCODE

Jednou z najzávažnejších pripomienok firmy OLTIS bola chýbajúca možnosť lokalizovať zásielky pomocou kódov SR 70, ktoré sme bližšie opísali v kapitole Analýza. Prevzatá implementácia tieto kódy nepodporuje – pri získavaní údajov o aktuálnej polohe zásielky boli v doterajšom riešení použité len GPS súradnice, prípadne konkrétna adresa, ktorú zadal manipulant do webovej aplikácie. Keďže bude systém primárne využívaný na sledovanie zásielok prepravovaných po železničnej sieti, lokalizácia pomocou spomínaných kódov je výhodnejšia.

Našou snahou je využívať hybridný prístup. Ten by spočíval v tom, že v prípade železničnej prepravy využijeme SR 70 kód (na základe ktorého server automaticky doplní GPS súradnice zodpovedajúcej stanice) a v prípade iného druhu prepravy využijeme priamo GPS pozíciu. Transformáciu konkrétneho kódu SR 70 na súradnice zabezpečí samostatná databázová tabuľka, ktorá bude obsahovať tento kód, názov mesta (prípadne konkrétnej stanice) a súradnice.

Pre potreby zadávania týchto dát používateľmi bude potrebné implementovať zmeny na logickej aj grafickej časti frontendu webovej aplikácie, ako aj vytvoriť príslušnú funkcionalitu v mobilnej aplikácii pre manipulantov.

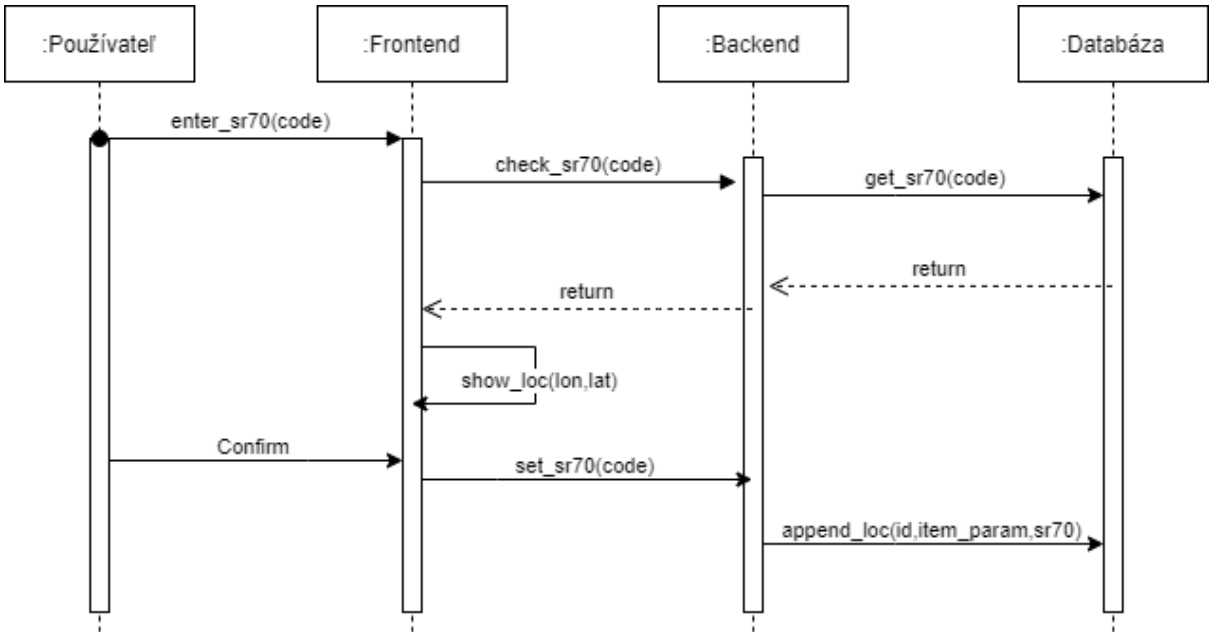
Návrhy zmien týkajúcich sa grafickej časti webovej aplikácie sú načrtnuté na obrazovkách nižšie (obrázok 5). Konkrétne ide o skrytie terajších existujúcich polí na zadávanie GPS súradníc a adresy a pridanie poľa na zadanie kódu SR 70. Ďalej pribudnú aj polia s informáciou o danom mieste (mesto / vlaková stanica / GPS súradnice miesta), ktoré ale nebude možné meniť manuálne – k ich zmene dôjde automaticky po zadaní kódu SR 70. Staré riešenie stále zostane funkčné, ale zobrazí sa a bude použiteľné až po odkliknutí tlačidla určeného na zmenu štýlu zadávania informácií o polohe zásielky. Tento spôsob budú môcť používať najmä koncoví dopravcovia zásielok (využívajúci iné spôsoby dopravy, týmto štandardom nepodporované), prípadne výrobcovia, aby označili výrobný komplex, v ktorom

bola položka vyrobená, respektíve sklad, z ktorého bola expedovaná. Obdobné obrazovky budú vytvorené aj pre novú mobilnú aplikáciu.

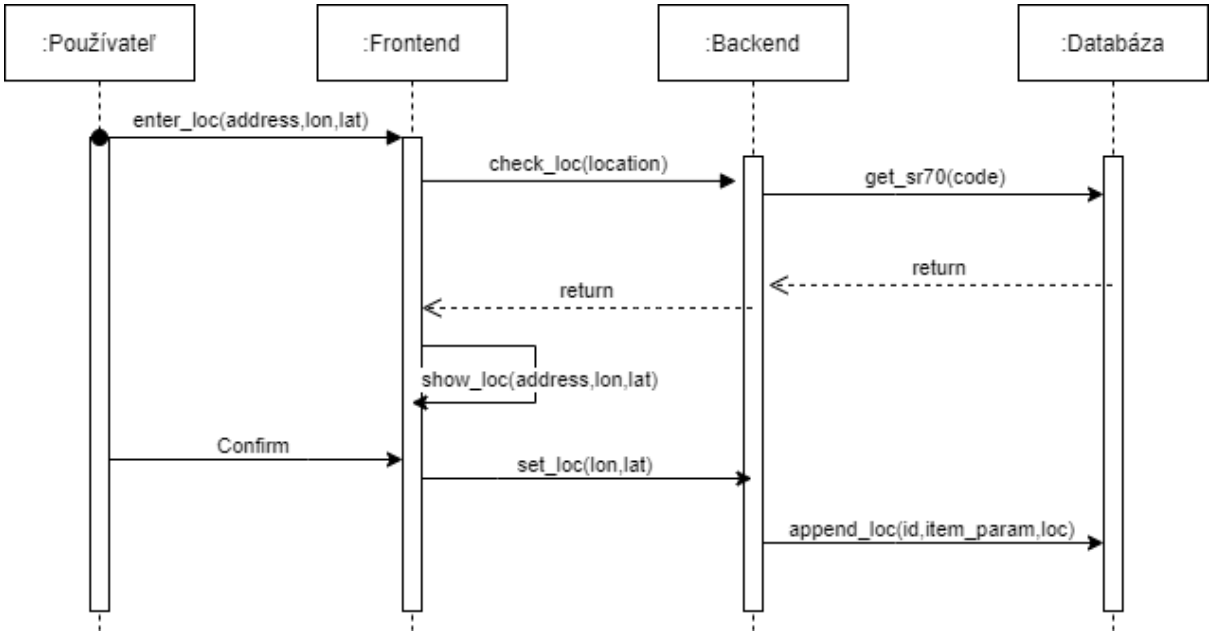
Obrázok 5 – Grafický návrh obrazovky webovej aplikácie

V logickej časti frontendov bude potrebné doimplementovať odosielanie údajov o polohe na server, keďže klient najprv odošle údaje o kóde SR 70 a následne získa od serveru dáta o danej lokalite zo zodpovedajúcej tabuľky (obrázok 6). Pokiaľ sa daný kód v tabuľke nenachádza, vypíše sa chybová hláška. Existujúce riešenie zostane implementované a bude využité v prípadoch, ktoré boli opísané v grafickej časti (obrázok 7).

Pozn.: Obdobné pravidlá ako pre kód SR 70 budú platiť aj pre využívanie štandardu UN/LOCODE, ktorý sa používa pre označovanie dopravných bodov celosvetovo.



Obrázok 6 – Sekvenčný diagram získavania údajov pomocou kódu SR 70



Obrázok 7 – Sekvenčný diagram získavania údajov pomocou zem. šírky a dĺžky

2.2.3 Internacionalizácia

Hoci sa v dnešnej dobe znalosť angličtiny považuje za samozrejmosť, v medzinárodnom priestore sa pohybuje aj významné množstvo potenciálnych používateľov nášho systému, ktorí angličtinu neovládajú. Z dôvodu zlepšenia dostupnosti a použiteľnosti systému preto považujeme za povinnosť implementovať do aplikácie aj lokalizáciu do ďalších vybraných jazykov a zjednodušiť spôsob vytvárania ďalších jazykových mutácií.

Naše riešenie spočíva v zavedení novej tabuľky do existujúcej databázy, ktorej účelom bude držať informácie o podporovanom jazyku a jednotlivých jazykových mutáciách údajov zobrazených vo webovej i mobilnej aplikácii.

V mobilnej aplikácii budú potrebné údaje uchovávané vo forme XML súborov. V rámci používateľského rozhrania teda nebudeme využívať konštantné texty, ale reťazce uložené v XML súboroch, ktoré môžeme neskôr pre ďalšie jazykové mutácie jednoducho dopĺňať a umožňujú nám jednoduchý výber medzi dostupnými jazykovými verziami. V prevzatej verzii zákazníckej aplikácie je implementovaná podpora jediného jazyka (angličtiny).

Vo webovej aplikácii bude internacionalizáciu zabezpečovať knižnica Vue i18n, ktorá slúži na získavanie jednotlivých textov vo zvolenom jazyku zo súboru formátu JSON. Všetky konštantné texty v aplikácii je preto potrebné zapísať do daného súboru a pridať im varianty pre podporované jazyky.

Používateľ s rolou výrobcu bude mať možnosť pridávať k produktom a výrobkom informácie vo viacerých jazykoch, ktoré si určí, pričom anglický jazyk bude povinný. To docielime pridaním ďalšej tabuľky do databázy, ktorá bude slúžiť na uchovávanie jazykových variantov jednotlivých informácií produktu/výrobku.

2.2.4 Úprava registrácie

Pôvodná aplikácia obsahovala funkčne implementovanú možnosť registrácie používateľa systému. Aktuálne systém rozpoznáva nasledovné typy používateľov:

- zákazník (client) – prehľad zoznamu produktov, skenovanie QR kódov produktov
- výrobca (manufacturer) – prehľad zoznamu vyrobených produktov, kontrola doručenia, sledovanie zásielok
- dopravca (carrier) – skenovanie QR kódu produktov, sledovanie zásielok, sledovanie prepravy zásielok, pridanie lokácie produktu
- správca (admin)
    - super admin – pridanie nových spoločností (company), vytváranie nových účtov, udeľovanie manufacturer admin a carrier admin práv
    - manufacturer admin – okrem funkcionalít výrobcov má aj možnosť pridať nových zamestnancov (výrobcov)
    - carrier admin – okrem funkcionalít dopravcu má aj možnosť pridať nových zamestnancov (dopravcov)

Do systému sa dokáže priamo registrovať len zákazník. Výrobcov a dopravcov dokáže registrovať len správca.

Problém aktuálnej implementácie spočíva v tom, že aplikácia umožňuje vytváranie účtov bez akéhokoľvek overenia, čo môže vyústiť do zneužívania cudzích či dokonca neexistujúcich e-mailových adries. Zároveň tento prístup (bez dodatočnej kontroly) dokáže zahltiť databázu falošnými používateľmi s neplatnými registračnými údajmi. Neexistuje tiež žiadna možnosť zrušenia účtu, čo je zjavný nedostatok pri uplatňovaní pravidiel GDPR a ďalších regulácií.

Na overenie registrácie budeme generovať jedinečný kód, ktorý bude používateľovi zaslaný na e-mailovú adresu uvedenú v procese registrácie. Tento kód bude priamo obsiahnutý vo verifikačnom odkaze, pričom po kliknutí na tento odkaz dôjde k automatickému overeniu používateľského účtu. Tento kód bude mať časovo obmedzenú platnosť. Po vypršaní platnosti kódu bez jeho využitia na overenie bude zodpovedajúci neoverený účet automaticky odstránený zo systému. Kód bude tiež v prípade chyby možné na e-mail opätovne zaslať. Neoverený používateľ k aplikácii až do momentu overenia nebude mať prístup nad rámec oprávnení bežného neregistrovaného používateľa.

Ďalšou navrhovanou funkcionalitou bude možnosť vymazania účtu. Používateľ bude mať možnosť účet deaktivovať (soft delete) alebo úplne odstrániť zo systému. Po deaktivácii je účet možné opätovne obnoviť.

2.2.4.1 Webová aplikácia

Pre zlepšenie zážitku z používania webovej stránky je tiež potrebné upraviť registračné obrazovky. Hlavnou zmenou je pridanie overovacích kódov, ktoré budú zaslané používateľovi hneď po registrácii. Preto je dôležité, aby bol používateľ na odoslanie tohto kódu upozornený.

Pre potreby overovania je vhodné vytvoriť novú obrazovku, na ktorú bude používateľ presmerovaný po kliknutí na pridelený potvrdzovací link. Táto obrazovka bude obsahovať informáciu o úspešnom overení registrácie a tlačidlo, ktoré používateľa presunie na hlavnú stránku webu.

Spolu s touto zmenou je potrebné taktiež vytvoriť logiku, ktorá bude zodpovedná za posielanie a prijímanie správ zo serveru týkajúcich sa potvrdzovania registrácie.

Nie všetky plánované zmeny sú ale spojené iba so zvýšením bezpečnosti registrácie. Plánujeme obrazovku registrácie upraviť tiež kvôli zlepšeniu prehľadnosti, nakoľko súčasný dizajn a umiestnenie niektorých elementov považujeme za nevhodné.

Vo webovej aplikácii je taktiež potrebné upraviť aj proces prihlasovania tak, že na prihlasovaciu obrazovku bude pridané políčko, po ktorého zakliknutí bude uložená informácia o tom, že používateľ si želá zostať v danom prehliadači prihlásený.

Na základe rozhodnutia používať SR 70, resp. UN/LOCODE na sledovanie produktov je potrebné upraviť aj niektoré obrazovky webu, kde sa tieto kódy budú zobrazovať alebo zadávať.

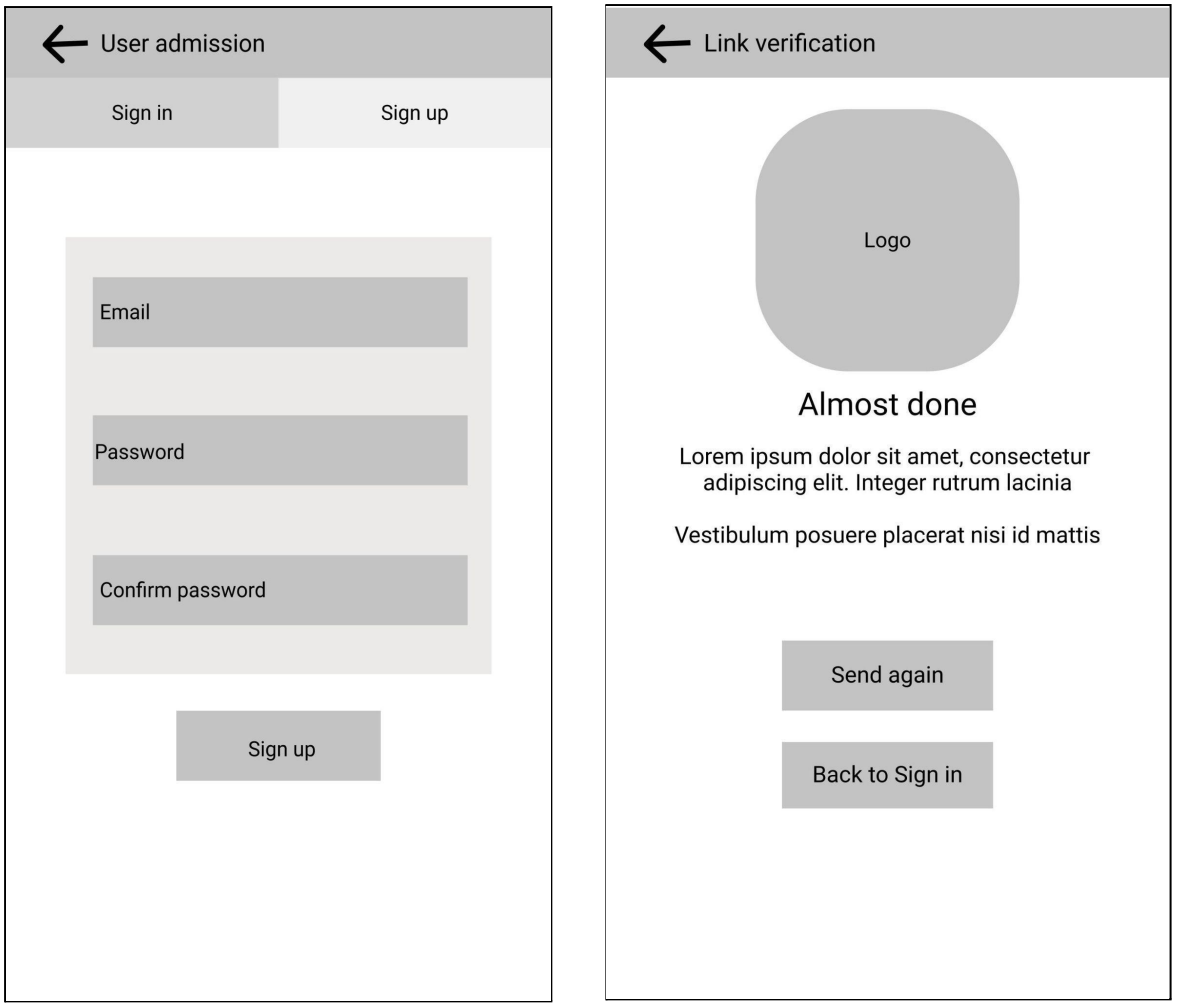
2.2.4.2 Mobilná aplikácia pre zákazníkov

Ako je uvedené v analýze, mobilná aplikácia v súčasnosti priamo neposkytuje možnosť registrácie používateľa. Používateľ sa teda musí pre plnohodnotný prístup k funkcionalite aplikácie registrovať prostredníctvom webového rozhrania systému, čo nám nepripadá ako vhodné riešenie.

Keďže pri registrácii potrebujeme od používateľov iba ich e-mailovú adresu a heslo, rozhodli sme sa na účely registrácie opätovne využiť obrazovku prihlasovania, v ktorej je nutné vykonať niekoľko zmien, napríklad pridať textové pole na potvrdenie zvoleného hesla.

Ako väčšina moderných mobilných aplikácií, aj tá naša bude obsahovať jednoduchú možnosť prechodu medzi obrazovkou prihlásenia a registrácie.

Po registrácii bude vyžadovaná aj validácia e-mailu od používateľa, ktorá z pohľadu mobilnej aplikácie predstavuje jednu informatívnu obrazovku o odoslaní verifikačného linku na poskytnutú e-mailovú adresu. Obrazovka bude taktiež obsahovať tlačidlo na opätovné odoslanie linku v prípade, že pri prvom pokuse nastala chyba (obrázok 8, 9).



Obrázok 8, 9 – Návrhy obrazoviek mobilnej aplikácie

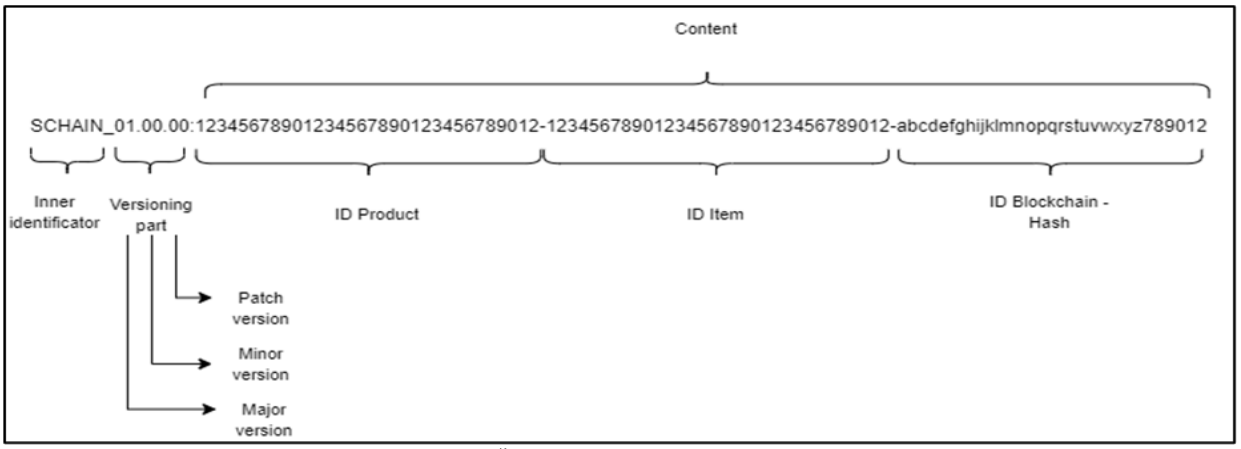
2.2.5 QR kód

2.2.5.1 Úprava generovania QR kódov

Hlavným dôvodom zmeny aktuálnej implementácie je snaha o robustnejší prístup ku generovaniu QR kódov. V pôvodnej verzii sa v procese generovania zohľadňuje iba ID produktu a ID výrobku. Zmenou prenášaných informácií docielime komplexnejšie overenie prenášaného obsahu prostredníctvom QR kódu a tiež možnosť implementovania vernostného systému.

Prvým krokom bude pridanie loga do vygenerovaného QR kódu, aby mohol používateľ jednoduchšie vizuálne identifikovať produkty, ktoré boli spracované pomocou systému S-Chain.

Okrem vonkajšej zmeny plánujeme upraviť aj obsah informácií prenášaných vnútri QR kódu. Nový obsah bude pozostávať z nasledovných prvkov (špecifikácia QR kódu S-Chain v1.0.0):



Obrázok 10 – Špecifikácia QR kódu S-Chain v1.0.0

- vnútorný identifikátor – identifikátor systému, ktorý vygeneroval QR kód
- verzovacia časť – časť umožňujúca kontrolu verzie vygenerovaného QR kódu
- ID produktu – identifikátor produktu (šablóny na výrobu výrobkov)
- ID výrobku – identifikátor konkrétneho (fyzického) výrobku
- blockchain hash – identifikátor umožňujúci neskoršie nasadenie vernostného systému

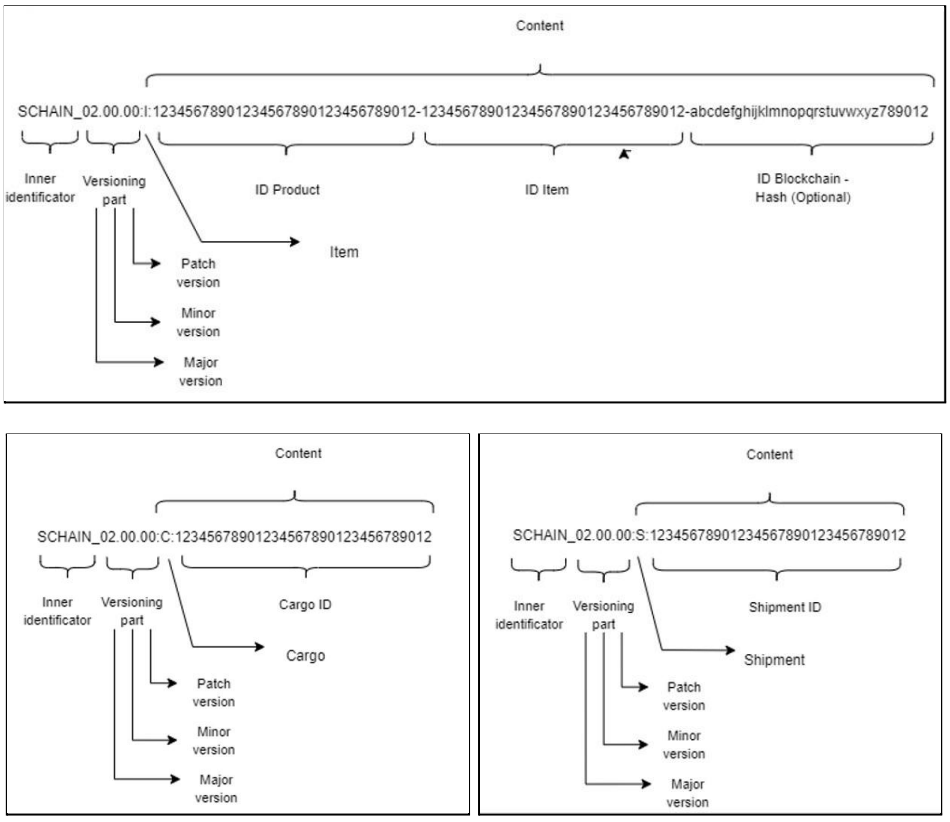
Vnútorný identifikátor chceme pridať pre jednoduchú identifikáciu QR kódu. Mobilná/webová aplikácia tak dokáže sama odhaliť neplatný kód bez potreby jeho zasielania na server. Dôvod pridania tohto identifikátora bolo zvýšenie robustnosti samotného QR kódu.

Ďalšou informáciou bude verzia vytvoreného kódu. Táto verzia bude primárne určená pre spracovanie obsahu kódu serverom, ktorý podľa nej dokáže korektne narábať aj so staršími verziami/tvarmi týchto kódov. Je veľmi dôležité, aby aj kódy, ktoré boli vydané v rámci skoršej špecifikácie boli stále v rámci systému validné a použiteľné a nebolo potrebné opätovne generovať ich novšie verzie.

Následne sme upravili samotnú dvojicu “id_produktu”-”id_výrobku”. Tieto identifikátory budú 32-miestne čísla. Zmena bude implementovaná primárne z dôvodu zachovania uniformity QR kódov.

Poslednou pridanou informáciou je 32-miestny alfanumerický hash, ktorý bude slúžiť pre potreby vernostného systému. Každému výrobku bude priradený unikátny a tajný hash, ktorý bude mimo databázy dostupný výhradne ako súčasť QR kódu na výrobku a jeho naskenovanie bude jednou z podmienok pre pridelenie zodpovedajúcej hodnoty vernostných bodov používateľovi

V priebehu implementácie projektu v letnom semestri sme identifikovali nedostatky v informáciách prenášaných prostredníctvom QR kódov. Nedostatky sa týkajú skutočnosti, že vyššie opísaný návrh podporuje iba QR kódy pre výrobky. Pre umožnenie vytvárania QR kódov aj pre kargá a zásielky sme navrhli QR kód verzie 2.0.0, ktorá prenáša aj informáciu o type QR kódu. Nová verzia má tvar uvedený na obrázkoch 11 až 13 (podľa typu obsahu).



Obrázok 11, 12, 13 – Špecifikácia QR kódu S-Chain v2.0.0

2.2.5.2 Zmeny spracovania QR kódov v rámci mobilnej a webovej aplikácie

Zmeny v súvislosti so zavedením nového formátu QR kódov sa týkajú aj funkčnej stránky frontendov webovej a mobilnej aplikácie. Existujúce riešenia nedisponujú kontrolou načítaného QR kódu, preto ju bude potrebné doplniť. Na kontrolu správneho tvaru QR kódu je vhodné využiť regulárny výraz, ktorý overí načítané dáta podľa vopred určeného vzoru podrobnejšie opísaného v časti 2.2.5.1. Nekontrolujú však formát časti "content", ktorý sa naprieč špecifikáciami môže výrazne líšiť, ale len interný identifikátor, verzovaciu časť a typ QR kódu. Ich predpokladaná frekvencia zmien naprieč špecifikáciami je dostatočne nízka na to, aby si vyžiadala nutnosť aktualizácie aplikácií.

Aplikácie skontrolujú korektnosť údajov jednotlivých častí zakódovaného reťazca (správnosť využitých symbolov, prípadne existenciu vopred daného reťazca – QR kód sa začína pomenovaním SCHAIN, tromi dvojčísliami označujúcimi verziu kódu a jedným znakom na označenie jeho typu, všetko s použitím zodpovedajúcich oddeľovacích znakov), vhodne rozdelia získané dáta do štruktúry JSON (troch polí označujúcich verziu a jedného poľa "content") a odošlú požiadavku na server.

Zmeny zasiahnu okrem načítania QR kódu aj sledovanie zásielky na základe ID. Nakoľko v existujúcom riešení je ID zhodné s QR kódom, bude implementovaná funkcionalita, ktorá zadané ID pred odoslaním na server upraví podľa príslušných pravidiel do formátu JSON a odošle.

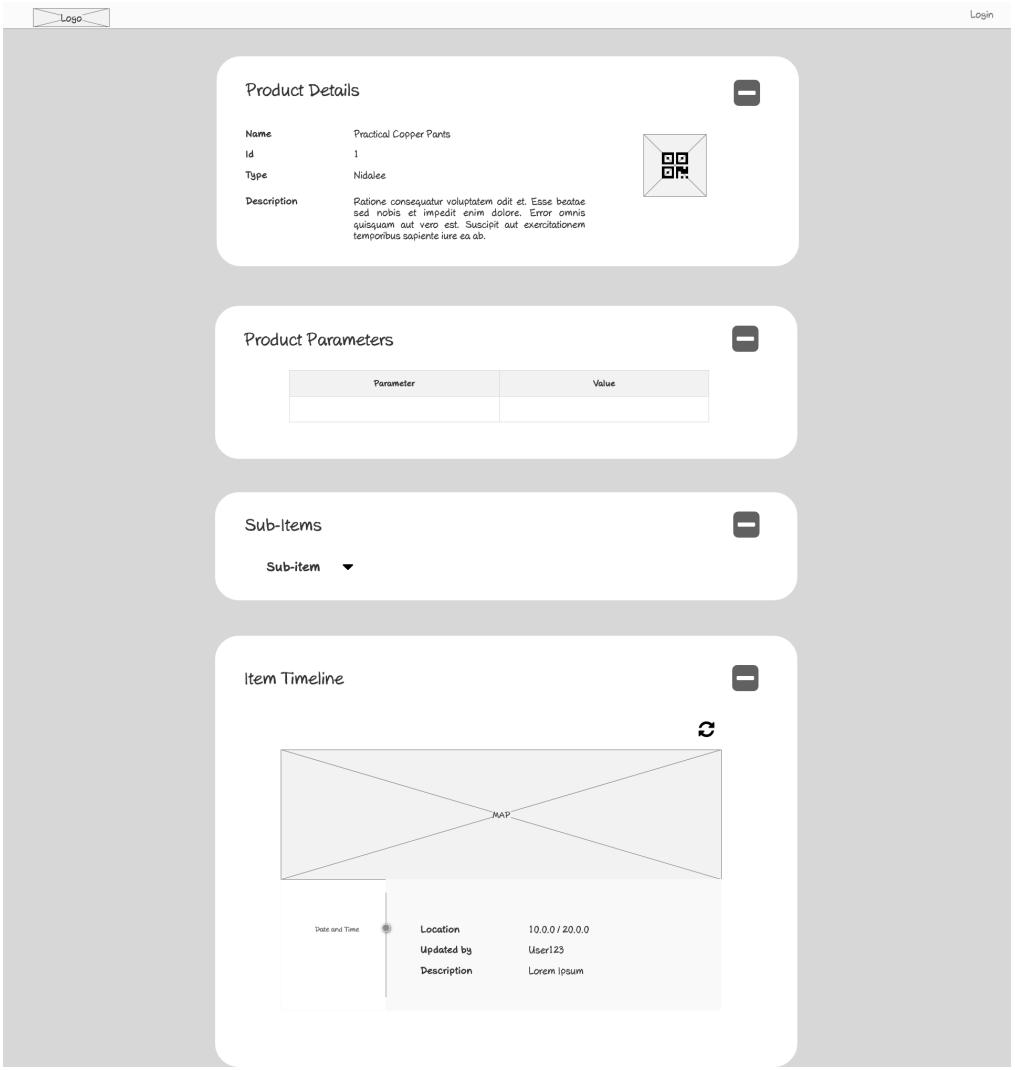
Úpravy sa nebudú týkať presmerovania na podstránku daného výrobku, pre URL sa budú používať rovnaké dáta ako v existujúcom riešení a teda "id_produktu-id_výrobku". Budú sa ale získavať odlišným spôsobom – nie priamo z QR kódu, ale zo štruktúry JSON, ktorá sa získa z načítaného QR kódu.

2.2.6 Úprava designu

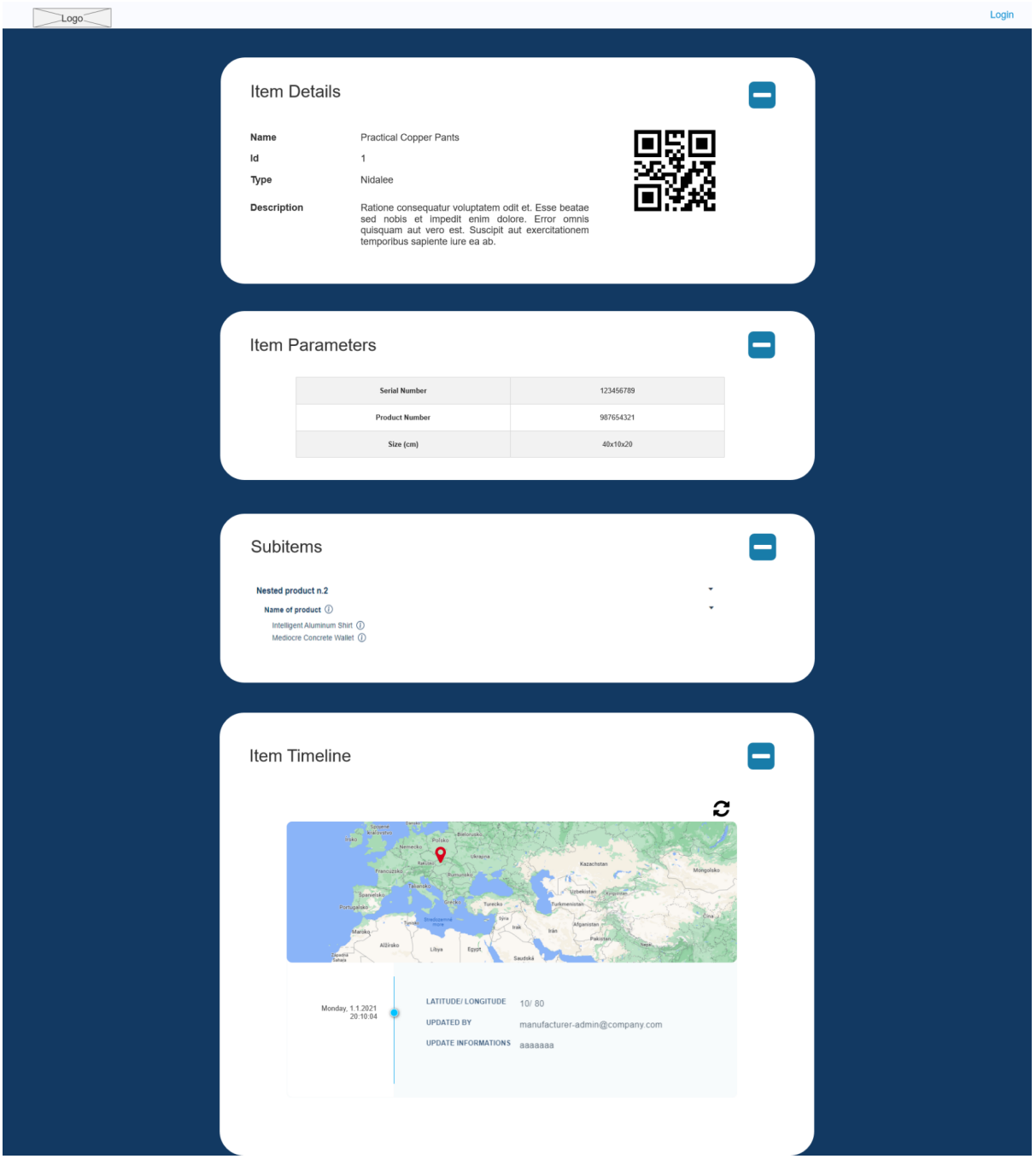
Počas analýzy existujúceho riešenia vznikla požiadavka na úpravu zobrazenia detailu výrobku vo webovej i mobilnej aplikácii.

2.2.6.1 Webová aplikácia

Údaje na stránke budú tematicky spojené do blokov, ktoré bude možné skrývať a zobrazovať. Zmena sa týka iba dizajnu a nemení obsah zobrazených údajov. Konkrétne zmeny sú vyobrazené v nasledujúcich grafických návrhoch obrazoviek (obrázok 14, 15).



Obrázok 14 – Low fidelity návrh obrazovky



Obrázok 15 – High fidelity návrh obrazovky

Hlavné navrhované zmeny v porovnaní s predošlým riešením:

- údaje budú tematicky rozdelené do 4 blokov (Item Details, Item Parameters, Subitems, Item Timeline)**;** dôvodom zmeny je zlepšenie orientácie používateľa v aplikácii

- pridanie popisov k jednotlivým poliam na obrazovke pre lepšiu informovanosť používateľa
- zlúčenie informácií o produkte s opisom produktu do jedného prvku
- spojenie blokov Item Last State a Item History Timeline do jednej časti obsahujúcej všetky informácie o lokalite položky; táto zmena vychádza najmä z dôvodu zbytočného opakovania jednej informácie na viacerých miestach

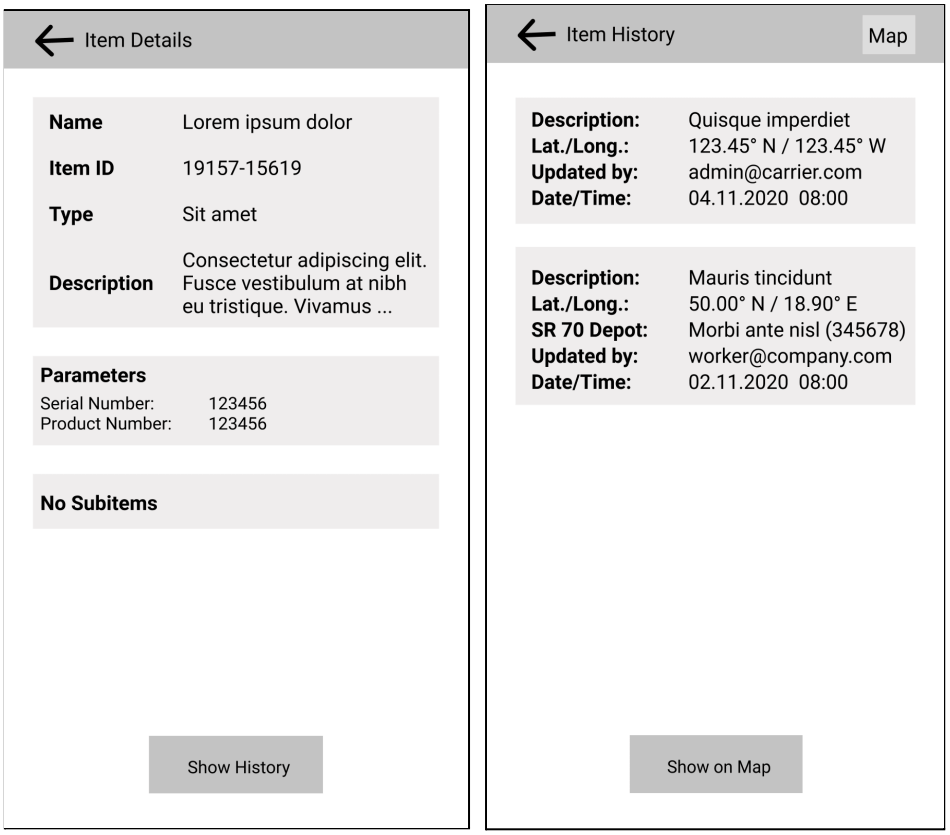
2.2.6.2 Mobilná aplikácia pre zákazníkov

V rámci zákazníckej mobilnej aplikácie dôjde k podobným úpravám, primárne s cieľom zjednotiť hlavné prvky rozhrania mobilnej a webovej aplikácie.

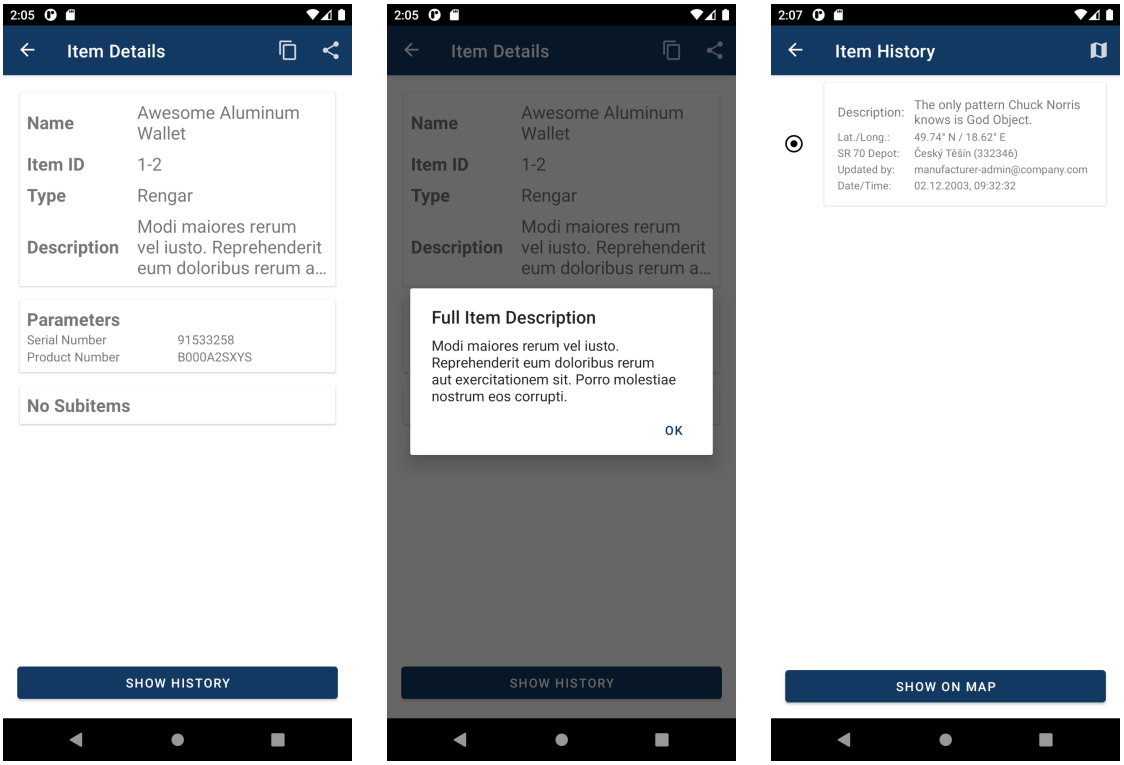
Názov výrobku sa presunie z hlavičky do samostatného poľa v rámci detailov výrobku. Obsah textového poľa s popisom výrobku sa skráti na fixnú maximálnu výšku; namiesto doteraz používanej, no neintuitívnej a nepraktickej možnosti rolovať týmto poľom sa v novom dizajne po rozkliknutí vyvolá dialógové okno obsahujúce kompletný popis.

Dôjde k preusporiadaniu jednotlivých polí a k sprehľadneniu informácií o jednotlivých bodoch v histórii výrobku; pribudnú nové informácie o pozícii podľa kódov SR 70 a/alebo UN/LOCODE (ak boli zaznamenané) a im zodpovedajúce zmeny v dátovom modeli aplikácie, ako aj jednoduchšie pochopiteľné prepočty zemepisnej šírky a dĺžky. Vznikne samostatné, výraznejšie tlačidlo pre grafické zobrazenie histórie (presun na obrazovku mapy), nakoľko používateľská spätná väzba naznačovala v tomto smere nedostatočnú intuitivitu súčasného rozhrania.

Na rozdiel od webového rozhrania ale kvôli lepšej použiteľnosti na mobilných zariadeniach stále zachováme skutočnosť, že ide o tri samostatné, vzájomne navigačne prepojené obrazovky (obrázok 16, 17, resp. 18, 19, 20).



Obrázok 16, 17 – Low fidelity návrhy obrazoviek mobilnej aplikácie



Obrázok 18, 19, 20 – High fidelity návrhy obrazoviek mobilnej aplikácie

2.2.7 Mobilná aplikácia pre manipulantov

Portfólio našich produktov sme sa rozhodli rozšíriť o mobilnú aplikáciu pre manipulantov – dopravcov a výrobcov. Funkcionalitu pre manipulantov sme nechceli pridávať do zákazníckej aplikácie, ktorá bude voľne dostupná aj pre bežných používateľov.

Mobilná aplikácia pre manipulantov bude obsahovať podobnú funkcionalitu ako webové rozhranie, avšak prispôsobenú mobilnému zariadeniu a práci v teréne. Aplikácia nebude umožňovať vytvárať produkty, výrobky, zásielky či kargá, keďže sme neidentifikovali žiadny prípad použitia, pri ktorom by bola táto funkcionalita potrebná priamo v teréne. Bude však možné prezerať si filtrované výrobky, zásielky aj kargá a ich históriu a taktiež aktualizovať s nimi súvisiace informácie.

2.2.7.1 Architektúra

Nová aplikácia by mala mať podobnú architektúru ako zákaznícka aplikácia, teda byť primárne koncipovaná ako single activity app s navigáciou medzi fragmentmi. Oproti zákazníckej aplikácii však bude výrazný rozdiel pri prihlasovaní používateľov. Nová aplikácia neumožní vstup bez prihlásenia a neprihlásený používateľ tak bude vidieť len obrazovku prihlásenia. V aplikácii tiež nebude možná registrácia, keďže dopravcovia ani výrobcovia sa neregistrujú sami, ale sú registrovaní administrátorom vo webovom rozhraní.

2.2.7.2 Hlavná obrazovka

Na hlavnej obrazovke aplikácie bude čítačka QR kódov. Okrem skenovania pomocou čítačky bude možné zadať ID aj manuálne a tiež nahrať obrázok už odfoteného QR kódu podobne ako v zákazníckej aplikácii.

Takto zadané/extrahované ID sa pošle na server a na základe odpovede bude následne používateľ presmerovaný na príslušnú obrazovku výrobku, zásielky alebo karga.

2.2.7.3 Výrobky

Funkcionalita týkajúca sa výrobkov bude oproti zákazníckej aplikácii prevažne nezmenená; prístup k výrobkom prostredníctvom aplikácie pre manipulantov majú len výrobcovia, nakoľko pre dopravcov informácie o konkrétnych výrobkoch, ktoré prepravujú, nie sú relevantné.

2.2.7.4 Zásielky

Aplikácia pre manipulantov bude podporovať všetky základné typy činností, ktoré je primerané očakávať pri využití aplikácie v teréne (napr. v sklade výrobcu či dopravcu alebo počas prepravy). Menovite pôjde hlavne o skenovanie QR kódov zásielok, filtrovanie kompletného zoznamu zásielok, nad ktorými má výrobca alebo dopravca kontrolu, zobrazenie detailu konkrétnej zásielky a možnosť rôzne parametre zásielky aktualizovať (len zamestnanci výrobcu).

2.2.7.4.1 Filtrovanie

V aplikácii pre manipulantov bude možné z pozície výrobcu aj dopravcu pristupovať k filtrovanému zoznamu všetkých zásielok, ktoré prihlásenej spoločnosti patria, resp. jej boli pridelené na prepravu. Filtrovanie bude možné na základe stavu zásielky (vytvorená, v preprave, zrušená, doručená), miesta pôvodu a koncovej destinácie a dátumu vytvorenia zásielky. Po vyfiltrovaní sa zobrazí zoznam obsahujúci stručné informácie o zásielkach, ktoré zodpovedajú zadaným parametrom, s možnosťou priameho prechodu na detail jednotlivých zásielok.

2.2.7.4.2 Detail zásielky

V rámci detailného zobrazenia konkrétnej zásielky budú používateľovi prehľadne zobrazené všetky relevantné údaje o zásielke, napríklad je systémové a sledovacie identifikačné číslo, popis, stav, odhadovaná cena (v USD), miesto pôvodu a cieľ, prepravná spoločnosť, ktorej bola zásielka zverená a informácie o vytvorení a poslednej aktualizácii zásielky. Súčasťou detailu bude aj počet a prehľad výrobkov, ktoré sa v zásielke nachádzajú. Poslednou sekciou

bude prehľad histórie pohybu zásielky (v textovej pozícii i na mape), podobný ako pri histórii výrobku.

2.2.7.4.3 Aktualizovanie informácií

Aplikácia bude ponúkať aktualizovanie rovnakých informácií o zásielkach, ako ponúka existujúce webové rozhranie. Tieto informácie bude môcť aktualizovať iba výrobca.

Výrobca bude môcť aktualizovať opis, cenu, sledovacie ID, stav, pôvod, destináciu a aktuálnu polohu zásielky. Pôvod, destináciu a aktuálnu polohu bude možné aktualizovať pomocou kódov SR 70, UN/LOCODE alebo tiež GPS súradnicami po výbere miesta kliknutím na mapu.

Aktualizovanie informácií bude taktiež zahŕňať pridanie a odstránenie výrobkov z danej zásielky. Pridávanie bude možné na základe naskenovaného QR kódu výrobku.

2.2.7.5 Kargá

V prípade karga pôjde o prakticky identickú funkcionalitu ako pri zásielkach, avšak tentoraz ju bude môcť vykonávať iba dopravca. Samozrejme do, respektíve z karga nebude možné pridávať a odoberať výrobky, ale zásielky. Taktiež sa pri zobrazovaní informácií o kargu nebude zbytočne zobrazovať informácia o prepravnej spoločnosti, nakoľko prístup k zobrazeniu každého karga budú mať výhradne zamestnanci spoločnosti, ktorá ho spravuje.

2.2.8 Blockchain

Optimalizáciu komunikácie medzi serverom a blockchain databázou dosiahneme vytvorením novej funkcie, ktorá zabezpečí vkladanie viacerých záznamov do blockchainu naraz. Bude preto nevyhnutné v backend časti upraviť volanie funkcie, ktorá posiela vkladané záznamy blockchain databáze na uloženie. Rovnako bude tiež vhodné upraviť aj trvanie doby, počas ktorej blockchain databáza čaká na prijatie všetkých posielaných záznamov, a maximálny počet záznamov, ktorý je možný naraz odoslať.

2.3 Implementácia

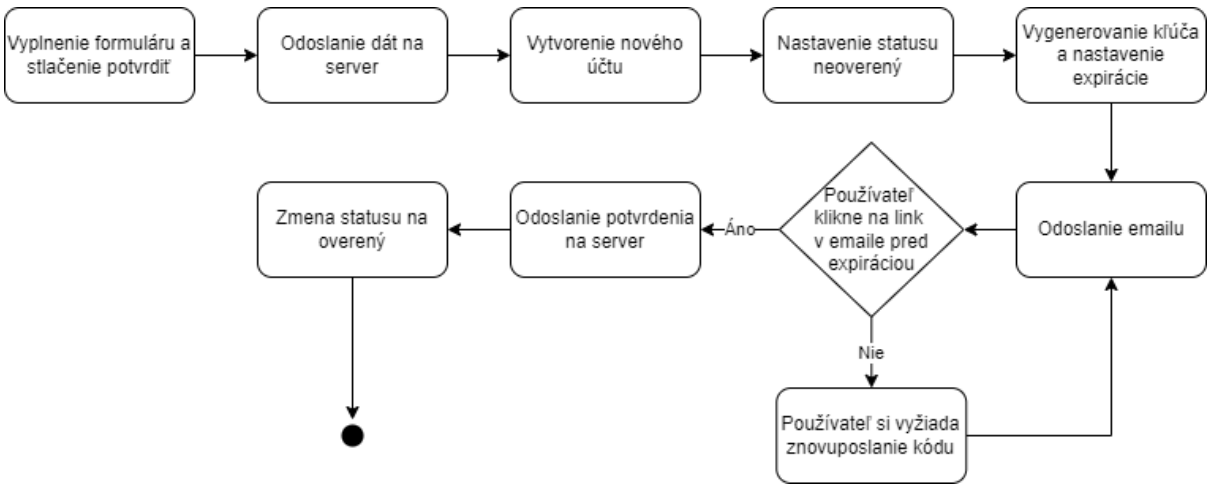
V rámci implementácie vyplývajúcej z kapitoly Návrh boli v projekte implementované nasledujúce zmeny.

2.3.1 Registrácia

2.3.1.1 Backend

Proces registrácie (obrázok 21) bol rozšírený na nasledujúce kroky:

- vytvor nový účet a nastav mu status "neoverený"
- vygeneruj používateľskému účtu overovací kľúč a nastav mu dobu exspirácie
- zašli overovací e-mail používateľovi



Obrázok 21 – Proces registrácie

Používateľ dokáže pomocou otvorenia linku v e-maile aktivovať účet. Pokiaľ nie je účet aktivovaný, pokus o prihlásenie vráti chybu. Táto chyba obsahuje používateľské meno, rolu a informáciu, že účet nie je aktivovaný.

Na generovanie a spracovanie validačného kódu pribudla nová service trieda (RegisterService). Táto trieda má definované rôzne funkcie:

generateValidationHash() – táto funkcia slúži na samotné generovanie validačného kódu. Kód je tvorený ako 32-miestny alfanumerický reťazec.

generateAndSaveValidationCode(String username) – funkcia vygeneruje validačný hash a vloží ho do databázy pre daný účet. Kód je potom vložený do tabuľky registration_code. Funkcia vracia validačný kód.

getVerificationLink(String username) – funkcia vytvorí verifikačný link pre používateľa. Počas generovania zistí, či pre daného používateľa existuje vygenerovaný kód – ak nie, vygeneruje ho. Samotný link je vytvorený pomocou šablóny.

validateRegistrationCode(Validation validation) – funkcia slúži na overenie poskytnutého registračného kódu. Funkcia dokáže identifikovať a vyhodiť tri typy chýb:

- používateľ bol už validovaný
- validačný kód exspiroval
- validačný kód nebol správny

Ak validácia prešla všetkými kontrolnými bodmi, status účtu je nastavený na aktívny a verifikačný kód je z databázy vymazaný.

Pri implementácii bolo tiež potrebné upraviť niektoré koncové body API a vytvoriť niekoľko nových.

/auth/login – koncový bod typu POST, ktorý počas prihlasovania aplikácia kontroluje, či je používateľský účet aktívny.

/auth/register – koncový bod typu POST, ktorý počas registrácie vygeneruje a pošle registračný kód.

/auth/validate?user={}&hash={} – koncový bod typu GET, ktorý je určený na validáciu používateľského účtu.

/resend/{username} – koncový bod typu GET, ktorý je určený na zaslanie verifikačného linku na používateľský e-mail.

Pre potreby registrácie bola implementovaná funkcionalita na zasielanie e-mailov. Na implementáciu sme využili knižnicu spring-boot-starter-mail, ktorá nám poskytla jednoduché API na tvorbu a odosielanie správ.

Samotná správa je tvorená v triede EmailData, ktorá obsahuje všetky dôležité informácie (príjemcov, predmet, telo správy, prílohy). Samotná trieda sa skladá z nasledujúcich funkcií:

- addAttachment(String resourcesPath) – funkcia pridá prílohu podľa cesty
- addInlineResource() – funkcia pridá vložený súbor (pre potreby html šablóny)
- insertDataToTemplate() – funkcia na nahradenie premenných v šablóne

Samotný e-mail je potom zložený a odoslaný v triede EmailSenderService.

2.3.1.2 Mobilná aplikácia pre zákazníkov

2.3.1.2.1 Registrácia

Pri implementácii registrácie sme namiesto vytvorenia nového fragmentu opätovne použili fragment pre prihlásenie. V ňom sme upravili používateľské rozhranie nasledovne:

- pridanie TabLayout-u na prepínanie medzi prihlásením a registráciou
- pridanie textového poľa na overenie správnosti hesla, ktorého viditeľnosť sa nastavuje podľa aktuálneho výberu (registrácia, prihlásenie)
- text na tlačidle pre potvrdenie, ktorý sa taktiež nastavuje podľa aktuálneho výberu

Okrem grafickej časti sme museli upraviť aj tú logickú. V tejto časti sa úprava týkala hlavne obsluhy používateľského rozhrania, odoslania požiadavky na server a následného vysporiadania sa s odpoveďou. Jednotlivé kroky zahŕňajú:

- pridanie argumentu pri navigácii z domovskej obrazovky na obrazovku prihlásenia/registrácie, na základe ktorého zistíme, o ktorú z nich ide

- pridanie actionListenerov pre jednotlivé tlačidlá, ktoré budú prepínať medzi prihlásením a registráciou
    - changeRegLog(isRegister: Boolean)
- pridanie funkcie na registráciu
    - získanie a overenie správnosti údajov – getCredentials(isRegister: Boolean), checkRegCredentials()
    - odoslanie požiadavky triede ApiCallHandler, ktorá rieši komunikáciu so serverom, a následný výpis odpovede – register()
- pridanie funkcie na odoslanie požiadavky na server – trieda ApiCallHandler
    - register(Credentials: credentials) → funkcia volá ďalšiu funkciu inej triedy – ItemApi register(Credentials: credentials), ktorá komunikuje so serverom
- pridanie funkcie na vysporiadanie sa s odpoveďou servera – trieda ApiCallHandler
    - handleRegister(response: Response<T>)

2.3.1.2.2 Verifikácia

Okrem registrácie sme v mobilnej aplikácii taktiež implementovali verifikáciu registrovaného e-mailu. Znamenalo to vytvorenie novej informačnej obrazovky, ktorú sme následne museli vložiť do navigácie. Pristúpiť je k nej možné z obrazovky prihlásenia/registrácie. Obrazovka obsahuje logo S-Chain, informáciu o odoslaní verifikačného linku a taktiež dve tlačidlá – pre opätovné odoslanie linku a pre návrat na obrazovku prihlásenia.

Logická časť opäť obsahuje viacero funkcií:

- návrat na stránku prihlásenia (s predvyplneným e-mailom)
- odoslanie linku a následné vypísanie odpovede, ktoré postupuje cez podobné funkcie ako tomu bolo pri registrácii
    - funkcia na spracovanie požiadavky o ktoré sa starajú ďalšie triedy, a vypísanie odpovede
    - funkcia na odoslanie požiadavky na server – trieda ApiCallHandler

- resendLink(Credentials: credentials) → funkcia volá ďalšiu funkciu inej triedy – ItemApi resendLink(Credentials: credentials), ktorá komunikuje so serverom. Následne sa vysporiada s odpoveďou.

2.3.1.3 Webová aplikácia

2.3.1.3.1 Registrácia

Vo webovej verzii aplikácie sme upravili registračný proces. V aplikácii sme pridali informačné obrazovky, ktoré užívateľa usmerňujú počas celého procesu registrácie. Napríklad po registrovaní aplikácie užívateľa informuje o tom, že mu prišiel overovací e-mail.

Takéto informačné obrazovky rovnako pribudli aj pri prihlasovaní už registrovaného používateľa do aplikácie.

2.3.1.3.2 Verifikácia

Po registrácii nového používateľa je vyslaná požiadavka na backend server, ktorá pošle e-mail s verifikačným kódom na e-mailovú adresu novoregistrovaného používateľa. Bez kliknutia na tento link sa používateľ do aplikácie nemôže prihlásiť a je upozornený, že jeho účet nie je overený.

2.3.2 Pridanie refresh tokenov

Pre potreby mobilných a webovej aplikácie sme pridali generovanie refresh tokenov. Tieto tokeny slúžia na obnovenie prihlásenia bez potreby opätovného zadávania prihlasovacích údajov. Refresh tokeny sú platné 30 dní.

2.3.2.1 Backend

Z pohľadu backendu bolo potrebné upraviť proces prihlasovania. V rámci prihlasovania sa okrem autorizačného tokenu vracia aj obnovovací token. Tento token nie je možné použiť na autorizáciu, pretože neobsahuje informácie o rolách používateľa. Jediné jeho využitie je

na vygenerovanie obnoveného autorizačného tokenu. Pre potreby tvorby a obnovovania tokenov bola vytvorená trieda RefreshTokenHandler, ktorá obsahuje nasledujúce metódy:

- encode(AuthToken token) – funkcia zakóduje informácie z autentifikačného tokenu a vytvorí obnovovací token
- decode(String token) – funkcia vytvorí zo Stringu obnovovací token
- refresh(Optional<String> rawToken) – funkcia vytvorí zo zaslaného obnovovacieho tokenu nový autorizačný token.

Na obnovu tokenu bol vytvorený nový koncový bod:

- **auth/token/refresh** – koncový bod očakáva validný obnovovací token, zaslaný v overovacej hlavičke požiadavky

2.3.2.2 Mobilné aplikácie

V mobilnej aplikácii bolo potrebné spraviť viaceré zmeny. V prvom prípade išlo o zmenu modelu prihláseného používateľa. Okrem roly, používateľského mena a autentifikačného tokenu bolo nevyhnutné pridať aj refresh token a funkciu na odoslanie požiadavky na server.

Najväčšia časť funkcionality však pribudla pri spracovávaní požiadaviek. Pribudla tu funkcia setAuthToken(refreshToken: String), ktorá odošle požiadavku na obnovu autentifikačného tokenu na server a následne prijatý token priradí prihlásenému používateľovi. Samozrejmosťou je ošetrenie chybových scenárov.

Ďalej museli byť upravené všetky funkcie komunikujúce so serverom, ktoré používajú autentifikačný token. Tieto funkcie sa musia vysporiadať s možnosťou neplatnosti tohto tokenu, čo znamená zavolať funkciu setAuthToken a následne znova odoslať požiadavku na server.

2.3.2.3 Webová aplikácia

V rámci implementovania refresh tokenov vo webovej aplikácii bolo potrebné doplniť novú funkcionalitu na obnovenie autorizačného tokenu na základe refresh tokenu implementovanú

vo funkcii authHeaderRefreshToken() a rozšíriť logiku prihlasovania o prvotné získanie refresh tokenu, kde bola štruktúra získavaná pri prihlásení rozšírená o toto pole. Ďalej bola upravená funkcia slúžiaca na overovanie a použitie autentifikačného tokenu pri požiadavkách na server, nakoľko táto funkcia musela byť rozšírená o spracovanie refresh tokenov a získanie nového autorizačného tokenu v prípade jeho vypršania počas doby platnosti refresh tokenu.

2.3.3 QR kód

Popis súčasnej implementácie jednotlivých funkcionalít identifikovaných v časti Návrh pre oblasť QR kódov je uvedený v podkapitolách nižšie.

2.3.3.1 Podpora nového formátu QR kódu

Proces overovania obsahu QR kódu sme implementovali v triede “ClientItemApi.java” v metóde “checkSentQR(JsonQR jsn)”. Správanie metódy môžeme popísať v troch krokoch:

- zachytenie POST requestu – na endpointe “/api/client/item/checkQR”. POST request nesie JSON obsahujúci informácie naskenované z QR kódu
- spracovanie prijatého JSON – overenie verzií (major, minor, patch) QR kódu. Okrem verzie obsahuje QR kód aj typ informácie, ktorú nesie. Podporovanými typmi sú výrobok (i), kargo (c) a zásielka (s). Na základe typu kódu a verzie prebehne kontrola prijatého ID (32-miestne čísla s vodiacimi nulami, ktoré po kontrole odstraňujeme). Nakoniec sa kontroluje, či má používateľ prístup k danému zdroju.
- presmerovanie – ak bola kontrola jednotlivých polí JSON objektu úspešná, nasleduje presmerovanie požiadavky na endpoint podľa typu QR kódu. Pre výrobok je presmerovanie na “/api/client/item/{id}”, ktorý zabezpečuje načítanie výrobku z DB. Posielaným parametrom je ID, ktoré má tvar “id_produktu-id_výrobku”, pričom product_item a item_id sú 32-miestne čísla z JSON objektu bez vodiacich núl. Ak kontrola dopadla neúspešne, je vyhodená výnimka. Ak sa používateľ dopytuje na kargo, tak je presmerovaný na konečný bod “/api/carrier/cargo/{id}”. Poslednou možnosťou je presmerovanie na zásielku. Toto presmerovanie závisí od prihláseného

používateľa “/api/{user}/shipment/{id}” (funguje správne len pre výrobcu a dopravcu).

2.3.3.2 Generovanie QR kódu s logom

Pri modifikovaní tohto procesu sme v prvom kroku nahrali logo S-Chain “logo_schain_v3.png” do priečinka resources. Týmto logom budeme prekrývať QR kód. Následne sme upravené generovanie implementovali v triede “ItemCodeGenerator.java” vo funkcii “generateQrCodeBase64Overlay(String id)”.

Metóda funguje nasledovne:

- generovanie QR kódu podľa ID výrobku, ktorý je v tvare “id_produktu-id_výrobku”. Pri generovaní sa používajú nápovedy (hints), ktoré zaručujú, že skenovanie QR kódu bude úspešné aj po vložení nášho obrázka (umožňujú korekciu chybného skenu kódu)
- načítanie a naškálovanie loga S-Chain, ktoré chceme vložiť do vygenerovaného QR kódu
- pridanie popisu pod QR kód v tvare:
    - pre výrobok: “ITEM:id_produktu-id_výrobku”
    - pre kargo: “CARGO:id_karga”
    - pre zásielku: “SHIPMENT:id_zásielky”
- samotné spojenie vygenerovaného QR kódu so spracovaným logom S-Chain a popisom QR kódu. Vo finálnej podobe je logo v strede QR kódu a popis pod QR kódom.

2.3.3.3 Frontend

Vo webovej aplikácii bola v rámci implementácie načítania a spracovania nového QR kódu mierne pozmenená logika načítania stránky pre detail výrobku. V predošlom riešení sa detaily výrobku získavali z backendu až po načítaní stránky pre vybrané ID výrobku. Nakoľko nový QR kód obsahuje aj iné informácie a jeho rôzne verzie môžu mať rôznu formu, nie je možné získať ID výrobku bez dopytu na server. V dôsledku týchto zmien

na backende je pri aktuálnej podobe implementácie potrebné načítať informácie o výrobku už na predošlej stránke a z nich získať ID výrobku.

Nakoľko k presmerovaniu na stránku s detailom výrobku dochádza z rozličných obrazoviek, bolo potrebné túto zmenu zapracovať na viacerých miestach v zdrojovom kóde. Konkrétne ide o zobrazenie výrobku z úvodnej stránky na základe jeho ID (súbor LandingPage.vue), respektíve načítania QR kódu (súbor QRReader.vue). Pri zobrazení detailu výrobku výrobcom ostala zachovaná predošlá implementácia, nakoľko výrobca zobrazuje výrobok podľa jeho ID, nie na základe QR kódu.

Pri načítaní QR kódu bola implementovaná aj základná kontrola správnosti formátu QR kódu. Nový tvar QR kódu (podrobnejšie opísaný v časti QR kód v kapitole Návrh) sa skladá z dvoch hlavných častí – hlavičky a obsahu. V rámci frontendu je kontrolovaná len prvá časť – hlavička, kde sa pomocou regulárneho výrazu (obrázok 22) kontroluje, či obsahuje všetky požadované prvky podľa špecifikácie.

```
/^SCHAIN_[0-9]{2}\.[0-9]{2}\.[0-9]{2}:*/gm
```

Obrázok 22 – Regex na overenie hlavičky QR kódu

2.3.4 Internacionalizácia

Oproti pôvodnému návrhu nebola v rámci implementácie pridaná podpora viacjazyčných popisov výrobkov; výrobca si môže zvoliť ľubovoľný jazyk popisu, ktorý k objektu priradí.

2.3.4.1 Frontend

Vo webovej aplikácii bola internacionalizácia implementovaná s využitím knižnice Vue i18n. V rámci použitia tejto knižnice bol vytvorený súbor i18n.js, ktorý obsahuje štruktúru formátu JSON. V tejto štruktúre sa nachádzajú všetky statické textové reťazce používané v aplikácii preložené do všetkých podporovaných jazykov, dohľadateľné pomocou konkrétnych identifikátorov. Vo zvyšku zdrojového kódu boli všetky zodpovedajúce texty nahradené

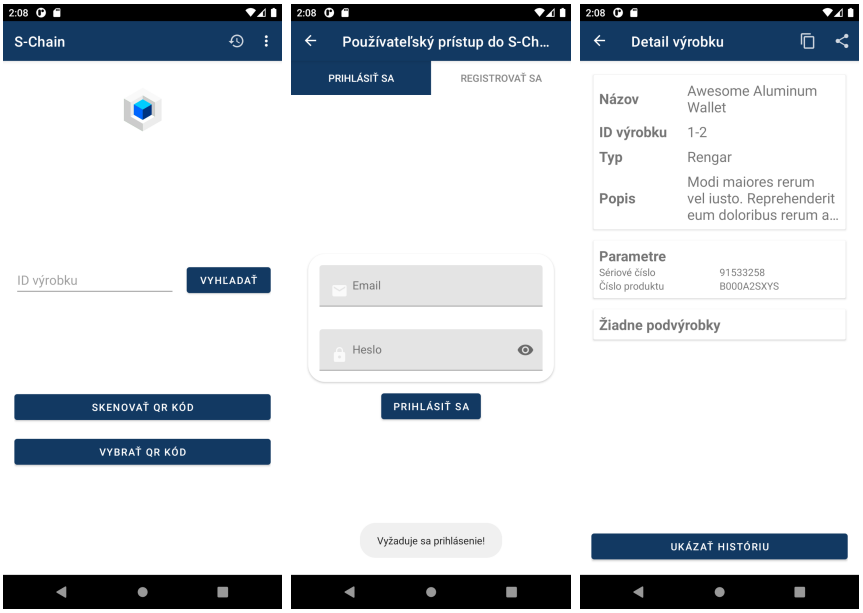
príslušnou premennou zo súboru i18n.js. Podpora ďalších jazykových mutácií bude v budúcnosti teda spočívať iba v rozšírení štruktúry JSON o nový jazyk.

Ďalším implementovaným prvkom webovej aplikácie týkajúcim sa internacionalizácie je výber jazyka v hlavičke stránky. Vizuálne je tento prvok reprezentovaný ako zoznam skratiek podporovaných jazykov.

2.3.4.2 Mobilné aplikácie

Android Studio, ktoré používame pri vývoji mobilnej aplikácie, odporúča využívať na internacionalizáciu XML súbory strings.xml, ktoré sa nachádzajú v priečinkoch "values-{kód lokalizácie}". O tom, z ktorého priečinku sa súbor strings.xml vyberie, rozhoduje nastavená lokalizácia priamo v systéme Android. V prvom rade bolo teda potrebné vytvoriť XML súbor pre slovenskú lokalizáciu. Následne sme museli do týchto XML súborov pridať všetky konštantné textové reťazce zo zdrojového kódu, v ktorom sme nahradili dané reťazce premennými z XML súborov.

Došlo tiež k úprave atribútov viacerých textových polí a tlačidiel s popisom za účelom lepšej automatickej prispôsobiteľnosti rozličným dĺžkam internacionalizovateľných reťazcov, najmä na obrazovke histórie výrobku.



Obrázok 23, 24, 25 – High fidelity návrhy mobilnej aplikácie

2.3.5 Použitie SR 70 a UN/LOCODE

Oproti prvotnému návrhu sme pri implementácii využili aj lokalizačné kódy UN/LOCODE, ktoré sú používané aj mimo územia Slovenskej a Českej republiky a pre väčší rozsah typov dopravy.

2.3.5.1 Backend

V rámci databázy boli na podporu lokalizačných kódov SR 70 a UN/LOCODE vytvorené tabuľky:

- SR70 – tabuľka nesie nasledovné informácie:
    - id : int8
    - station_code : varchar(6)
    - station_name : text
    - gps_location_longitude : float8
    - gps_location_latitude : float8
- UN/LOCODE – tabuľka nesie nasledovné informácie
    - id : int8
    - locode: text
    - country: text
    - location_code : text
    - station_name : text
    - gps_location_longitude : float8
    - gps_location_latitude : float8

Tabuľka Package bola upravená nasledovne:

- destination_coordinates_latitude → destination_coordinates_gps_location_latitude
- destination_coordinates_longtitude → destination_coordinates_gps_location_longtitude
- location_latitude → location_gps_location_latitude
- location_longitude → location_gps_location_longitude

- origin_coordinates_latitude → origin_coordinates_gps_location_latitude
- origin_coordinates_longitude → origin_coordinates_gps_location_longitude

Následne pre možnosť práce s týmito tabuľkami boli implementované ORM triedy:

- SR70, SR70Dto, SR70Repo
- UNLOCODE, UNLOCODEDto, UNLOCODERepo

Taktiež boli pridané triedy:

- CoordinatesWithCodes, ktorá v sebe nesie:
    - gpsLocation : Coordinates
    - sr70_id : long
    - unlocode_id : long
- CoordinatesForCreation:
    - táto trieda v sebe nesie:
        - Coordinates coordinates
        - String sr70
        - String unlocode
        - String info
    - táto trieda je statická a vnorená v triedach:
        - UpdatePackage
        - CreatePackage
        - CreateItem
- CoordinatesForUpdate:
    - táto trieda v sebe nesie:
        - Coordinates gpsLocation
        - String sr70
        - String unlocode
    - táto trieda je statická a vnorená v triede:
        - UpdateState
- LocationService, ktorá v sebe nesie:

- transformCoordinates(Coordinates location, String sr70, String unlocode) – táto metóda slúži na pretransformovanie vstupných parametrov na objekt CoordinatesWithCodes

Nakoniec bolo potrebné pridať funkčné endpointy na vyriešenie prichádzajúcich requestov zo strany web aplikácie a tiež zo strany mobilnej aplikácie. Pre tieto potreby sme implementovali nasledovné endpointy:

- SR70
  - získanie všetkých záznamov z tabuľky SR70:
    - GET /api/location/sr70
  - získanie záznamu z tabuľky SR70 podľa názvu stanice:
    - GET /api/location/sr70/name/{id}
  - získanie záznamu z tabuľky SR70 podľa kódu stanice:
    - GET /api/location/sr70/code/{id}
  - získanie záznamu z tabuľky SR70 podľa GPS súradníc stanice:
    - GET /api/location/sr70/coordinates?lat=””&lon=””
- UNLOCODE
  - získanie všetkých záznamov z tabuľky UNLOCODE:
    - GET /api/location/unlocode
  - získanie záznamu z tabuľky UNLOCODE podľa názvu stanice:
    - GET /api/location/unlocode/name/{id}
  - získanie záznamu z tabuľky UNLOCODE podľa kódu stanice:
    - GET /api/location/unlocode/code/{id}
  - získanie záznamu z tabuľky UNLOCODE podľa GPS súradníc stanice:
    - GET /api/location/unlocode/coordinates?lat=””&lon=””
  - získanie všetkých dostupných krajín, pre ktoré máme UNLOCODE:
    - GET /api/location/unlocode/countries
  - získanie záznamu z tabuľky UNLOCODE podľa kódu lokácie:
    - GET /api/location/unlocode/location/code/{id}
  - získanie záznamu z tabuľky UNLOCODE podľa kódu krajiny:

- GET /api/location/unlocode/country/{id}

Vyššie spomenuté endpointy sú spracované v triedach SR70Api a UNLOCODEApi.

Po implementovaní lokalizácie pomocou kódov sme v rámci endpointov /api/manufacturer/, /api/carrier/ urobili nasledovné zmeny, spočívajúce v očakávanom JSON-e v requeste poslanom z webovej aplikácie a aplikácie pre manipulantov:

- /api/manufacturer/item

```
{
 "productId": 0,
 "serialNumber": "string",
 "state": {
  "coordinates": {
   "coordinates": {
    "latitude": 0,
    "longitude": 0
   },
   "sr70": "string",
   "unlocode": "string"
  },
  "info": "string"
 },
 "subItemsIds": [
  "string"
 ]
}
```

- /api/carrier/item/{id}

```
{
 "coordinates": {
  "coordinates": {
   "latitude": 0,
   "longitude": 0
  },
  "sr70": "string",
  "unlocode": "string"
 },
 "info": "string"
}
```

- /api/manufacturer/items

```
[
```

```
  {
    "productId": 0,
    "serialNumber": "string",
    "state": {
      "coordinates": {
        "coordinates": {
          "latitude": 0,
          "longitude": 0
        },
        "sr70": "string",
        "unlocode": "string"
      },
      "info": "string"
    },
    "subItemsIds": [
      "string"
    ]
  }
]
```


- /api/carrier/shipment/track-id/{trackId}/location


```
{
  "coordinates": {
    "coordinates": {
      "latitude": 0,
      "longitude": 0
    },
    "sr70": "string",
    "unlocode": "string"
  },
  "info": "string"
}
```


- /api/carrier/shipment/{id}/location


```
{
  "coordinates": {
    "coordinates": {
      "latitude": 0,
      "longitude": 0
    },
    "sr70": "string",
    "unlocode": "string"
  },
  "info": "string"
}
```

- /api/carrier/cargo


```
{
 "cargo": {
  "carrierCompanyId": 0,
  "createdAt": "2021-12-15T13:51:43.010Z",
  "createdBy": 0,
  "deliveredAt": "2021-12-15T13:51:43.010Z",
  "description": "string",
  "id": 0,
  "price": 0,
  "state": "CREATED",
  "trackId": "string",
  "updatedAt": "2021-12-15T13:51:43.010Z",
  "updatedBy": 0
 },
 "destination_coordinates": {
  "coordinates": {
   "latitude": 0,
   "longitude": 0
  },
  "info": "string",
  "sr70": "string",
  "unlocode": "string"
 },
 "origin_coordinates": {
  "coordinates": {
   "latitude": 0,
   "longitude": 0
  },
  "info": "string",
  "sr70": "string",
  "unlocode": "string"
 },
 "shipmentIds": [
  0
 ]
}
```


- /api/carrier/cargo/{id}


```
{
 "description": "string",
 "destination": {
  "coordinates": {
   "latitude": 0,
   "longitude": 0
```

```
    },
    "info": "string",
    "sr70": "string",
    "unlocode": "string"
  },
  "origin": {
    "coordinates": {
      "latitude": 0,
      "longitude": 0
    },
    "info": "string",
    "sr70": "string",
    "unlocode": "string"
  },
  "price": 0,
  "trackId": "string"
}
```


- /api/carrier/cargo/{id}/location


```
{
  "coordinates": {
    "coordinates": {
      "latitude": 0,
      "longitude": 0
    },
    "sr70": "string",
    "unlocode": "string"
  },
  "info": "string"
}
```


- /api/carrier/cargo/track-id/{trackId}/location


```
{
  "coordinates": {
    "coordinates": {
      "latitude": 0,
      "longitude": 0
    },
    "sr70": "string",
    "unlocode": "string"
  },
  "info": "string"
}
```


- /api/manufacturer/shipment


```
{
```

```
  "destination_coordinates": {
    "coordinates": {
      "latitude": 0,
      "longitude": 0
    },
    "info": "string",
    "sr70": "string",
    "unlocode": "string"
  },
  "itemIds": [
    "string"
  ],
  "origin_coordinates": {
    "coordinates": {
      "latitude": 0,
      "longitude": 0
    },
    "info": "string",
    "sr70": "string",
    "unlocode": "string"
  },
  "shipment": {
    "carrierCompanyId": 0,
    "createdAt": "2021-12-15T14:08:43.736Z",
    "createdBy": 0,
    "deliveredAt": "2021-12-15T14:08:43.736Z",
    "description": "string",
    "id": 0,
    "price": 0,
    "state": "CREATED",
    "trackId": "string",
    "updatedAt": "2021-12-15T14:08:43.737Z",
    "updatedBy": 0
  }
}
```


- /api/manufacturer/shipment/{id}


```
{
  "description": "string",
  "destination": {
    "coordinates": {
      "latitude": 0,
      "longitude": 0
    },
    "info": "string",
    "sr70": "string",
```

```
    "unlocode": "string"
  },
  "origin": {
    "coordinates": {
      "latitude": 0,
      "longitude": 0
    },
    "info": "string",
    "sr70": "string",
    "unlocode": "string"
  },
  "price": 0,
  "trackId": "string"
}
```


- /api/manufacturer/shipment/{id}/location


```
{
  "coordinates": {
    "coordinates": {
      "latitude": 0,
      "longitude": 0
    },
    "sr70": "string",
    "unlocode": "string"
  },
  "info": "string"
}
```


2.3.5.2 Mobilné aplikácie

V rámci logickej časti bolo nevyhnutné prispôsobiť dátový model aplikácie, aby bol schopný evidencie údajov týkajúcich sa kódov SR 70 a UN/LOCODE. K tomu došlo vytvorením dvoch nových dátových tried využívajúcich rozhranie Parcelable (Sr70 a UnLocode) a pridaním nových atribútov týchto dvoch typov do triedy State, ktorá nesie informácie o jednotlivých zaznamenaných okamihoch histórie stavu výrobku.

Príslušné vykonané zmeny v grafickej časti aplikácie potrebné pre ich zobrazenie na patričných obrazovkách sú opísané v kapitole 2.2.6.2 a viditeľné na obrázkoch 16 až 20.

2.3.5.3 Webová aplikácia

Vo webovej aplikácii bolo potrebné upraviť niekoľko obrazoviek, kde sa pridali polia na kódy SR 70 alebo UN/LOCODE. V prípade kódov UN/LOCODE bolo pridané aj pole na výber krajiny a následne bola možnosť výberu kódu dopravného uzla v danej krajine. Rovnako bolo potrebné upraviť aj logickú časť aplikácie, kde sme modifikovali formát pre dáta, ktoré odosielame alebo prijímame z backend časti aplikácie.

2.3.6 Generický endpoint pre update

Za účelom uľahčenia dopytovania sa na backend server mobilnými aplikáciami bol zavedený jednotný endpoint, ktorý generickým spôsobom updatuje jednotlivé atribúty karga (cargo) a zásielky (shipment).V princípe sa na endpoint zasiela string mapa v tele requestu. Endpoint dáta rozparsuje a upraví v príslušnom objekte (kargo/zásielka).

Problematické v našej implementácii je množstvo špeciálnych dátových typov, ktoré nie je možné priamo serializovať do objektu. Pre tieto prípady boli implementované tzv. “Parsers”, ktoré dokážu každý špeciálny dátový typ správne spracovať.

2.3.7 Aplikácia pre manipulantov

Z dôvodu širšej dostupnosti a lepšieho oddelenia logiky systému sme sa rozhodli vytvoriť mobilnú aplikáciu pre manipulantov. Na začiatku sme vytvorili nový projekt v Android Studiu a inicializovali základné vlastnosti, premenné, aktivity a podobne.

2.3.7.1 Architektúra

Nová mobilná aplikácia obsahuje tri aktivity. Pri spustení aplikácie sa automaticky spustí StartupActivity, ktorá skontroluje, či je používateľ (korektne) prihlásený alebo nie. Pokiaľ používateľ prihlásený nie je, prejde sa k LoginActivity, v ktorej sa môže prihlásiť. Pokiaľ je prihlásený, prejde sa k MainActivity, ktorá je koncipovaná ako NavigationActivity a teda umožňuje ďalšiu navigáciu medzi jednotlivými fragmentmi.

2.3.7.2 Prihlásenie

Ako sme uviedli v sekcii vyššie, prihlásenie do aplikácie sa vykonáva v samostatnej aktivite – LoginActivity. Implementácia je podobná zákazníckej aplikácii, je však vytvorená pomocou ViewModel a FormState, ktoré kontrolujú stavy textových polí. Pokiaľ sú vyplnené údaje o e-mailovej adrese a heslo a používateľ klikne na tlačidlo prihlásenia, tieto údaje sa pošlú na server a na základe odpovede zo servera sa buď vypíše chybová hláška alebo je používateľ presmerovaný na MainActivity. Prihlásenie je nutnou podmienkou pre prístup k MainActivity.

2.3.7.3 Hlavná aktivita

Hlavná aktivita aplikácie, MainActivity, obsahuje menu, v ktorom si používateľ môže vybrať, na akú obrazovku sa chce prekliknúť. Na výber má domovskú obrazovku alebo obrazovku filtrovania zásielok, v prípade dopravcu aj filtrovania kárg. Z týchto obrazoviek sa následne pomocou navigácie vie prepínať medzi ďalšími fragmentmi.

Hlavná aktivita taktiež poskytuje možnosť odhlásenia sa z aplikácie, a to bez ohľadu na aktuálny fragment, na ktorom sa používateľ nachádza.

2.3.7.4 Domovská obrazovka

Po úspešnom prihlásení sa používateľovi otvorí domovská obrazovka MainActivity. Na tejto obrazovke sa nachádza čítačka QR kódov. Okrem skenovania pomocou čítačky je možné zadať ID aj manuálne a tiež prečítať už odfotený QR kód z obrázku podobne ako v zákazníckej aplikácii.

Extrahované alebo zadané ID sa pošle na server; na základe odpovede sa následne presunieme na obrazovku výrobku, zásielky alebo karga. Väčšina funkcionality na domovskej obrazovke je prevzatá z pôvodnej zákazníckej aplikácie.

2.3.7.5 Výrobky

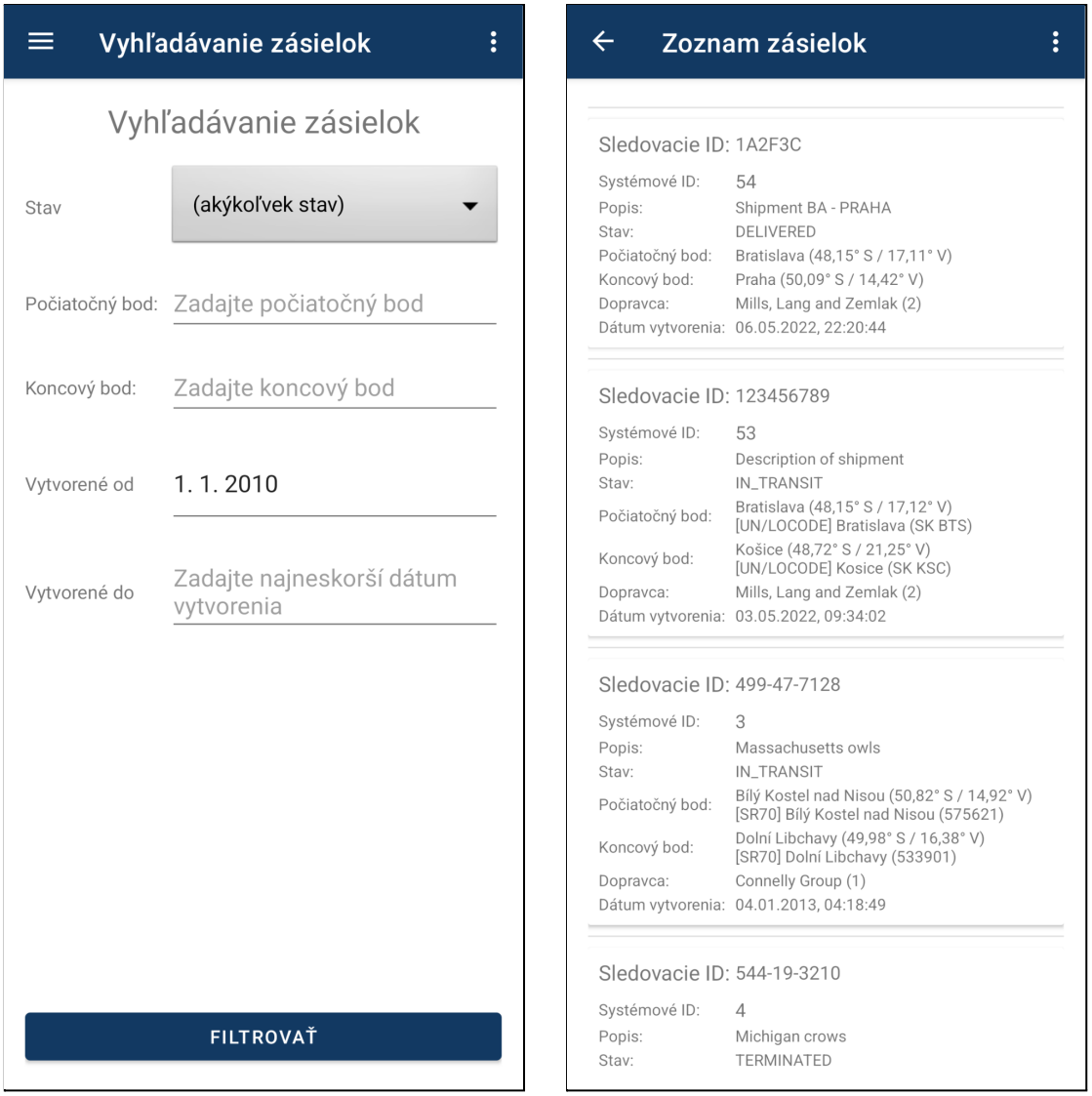
Funkcionalita pri výrobkoch je rovnaká ako v zákazníckej aplikácii. Výrobca sa vie z domovskej obrazovky po zadaní/naskenovaní ID výrobku, prípadne prostredníctvom detailu zásielky obsahujúceho tento výrobok dostať k detailom výrobku. Následne si vie pozrieť jeho históriu v textovej forme i priamo na mape.

2.3.7.6 Zásielky

Na strane aplikácie pre manipulantov sú zásielky implementované spoločne s kargami v rámci jednej triedy Package, keďže rozdiely medzi nimi sú minimálne. Výrobcovia majú prístup ku všetkým zásielkam, ktoré im patria (novovytvorenými, odovzdanými na prepravu, zrušenými i doručenými); dopravcovia len k tým zásielkám, ktoré im boli zverené na prepravu. V prípade dopravcov ide navyše väčšinou len o práva na zobrazenie informácií, nie na ich zmenu. Implementované boli všetky činnosti opísané v súvisiacej kapitole návrhu.

2.3.7.6.1 Filtrovanie

Funkcionalitu filtrovania zabezpečuje fragment PackageFilterFragment, ktorý je zodpovedný za načítanie zoznamu dostupných stavov zásielky zo servera do súvisiaceho dropdown spinneru a následne načítanie všetkých používateľom zadaných filtračných informácií, ich spracovanie a vo forme (redukovanej) stringmapy zaslanie na backend. Následne prebehne automatické navigačné presmerovanie na fragment PackageListFragment, na ktorom sa zobrazí výsledný zoznam zásielok (obrázok 26, 27), pričom po kliknutí na jednotlivé zásielky nastáva navigácia na detail konkrétnej zásielky.



Obrázok 26, 27 – High fidelity návrhy mobilnej aplikácie

2.3.7.6.2 Detail zásielky

Zobrazenie detailu zásielky (obsluhované fragmentom PackageFragment) je z veľkej časti podobné detailu výrobku implementovanému v zákazníckej i manipulantskej aplikácii. Hlavný rozdiel spočíva vo výrazne väčšom počte zobrazovaných informácií (napr. cena, poloha, dopravca, identifikačné čísla a pod.) a fakte, že namiesto zoznamu podvýrobkov, z ktorých sa výrobok skladá, sa na obrazovke nachádza zoznam výrobkov, ktoré sú súčasťou zásielky.

2.3.7.6.3 Aktualizovanie informácií

Aplikácia ponúka možnosť aktualizovania rovnakých informácií o zásielkach ako existujúce webové rozhranie. Tieto informácie môžu byť aktualizované iba výrobcom.

Pri obrazovke prehľadu zásielky môže používateľ kliknúť na tlačidlo aktualizovať zásielku. Zobrazí sa obrazovka, na ktorej sa nachádzajú textové polia na zmenu opisu, ceny a sledovacieho ID. Taktiež sa tu nachádza spinner (dropdown) so stavmi, do ktorých je možné priradiť zásielku.

Pokiaľ chce používateľ zmeniť koordináty – aktuálnu polohu, pôvod alebo destináciu, klikne na príslušné tlačidlo. Následne sa zobrazí obrazovka, na ktorej používateľ zvolí, čo presne chce zmeniť a typ zadávaných informácií – kód SR 70, UN/LOCODE alebo GPS súradnice.

Po zvolení SR 70 aplikácia odošle požiadavku na server, ktorý v odpovedi vráti všetky SR 70, ktoré sa vložia do filtrovateľného spinneru. Textový filter sa začne prejavovať po zadaní minimálne troch znakov kódu alebo názvu stanice. Z takto získaného filtrovaného zoznamu si používateľ vyberie konkrétny dopravný bod.

Pri zvolení UN/LOCODE je nutné najskôr zvoliť si krajinu záujmu a následne prebieha obdobný proces ako pri SR 70. Pokiaľ chce používateľ meniť priamo GPS súradnice, zobrazí sa mapa a používateľ klikne na konkrétne miesto na nej. Na aktualizovanie informácií je použitý nový generický endpoint.

Aktualizovanie taktiež zahŕňa pridanie a odstránenie výrobkov z danej zásielky. Pridávanie je možné na základe naskenovaného QR kódu výrobku.

2.3.7.7 Kargá

Funkcionalita týkajúca sa kárg je implementovaná prakticky identicky ako funkcionalita zásielok, nakoľko sú v rámci kódu reprezentované spoločnou triedou Package a s ňou súvisiacimi fragmentmi; prístup ku kargám je však obmedzený len pre dopravcov. Samozrejme do, respektíve z karga nie je možné pridávať a odoberať výrobky, ale zásielky. Taktiež sa pri zobrazovaní informácií o kargu nezobrazuje nadbytočná informácia

o prepravnej spoločnosti, nakoľko prístup k zobrazeniu každého karga majú výhradne zamestnanci spoločnosti, ktorá ho spravuje.

2.3.8 Vernostný systém

2.3.8.1 Backend

Z podnetov priemyselného partnera bola v záverečných šprintoch implementovaná aj základná funkcionalita pre podporu vernostného systému. Ten v aktuálnej verzii funguje pre registrovaných používateľov (zákazníkov), ktorým bude po naskenovaní QR kódu výrobku pripočítaný vernostný bod do ich zákazníckej peňaženky. Spomínaný QR kód obsahuje jedinečný hash (loyalty code), ktorý môže byť načítaný ľubovoľným počtom zákazníkov, avšak body sa každému z nich pripočítajú len raz. Podrobný priebeh a konkrétne možnosti používateľa týkajúce sa samotného procesu odmeňovania zákazníkov sú však (na základe dohody) v réžii priemyselného partnera.

2.3.8.2 Mobilné aplikácie

V mobilnej aplikácii pre zákazníkov (spotrebiteľov) bol po predošlej dohode s priemyselným partnerom vernostný systém nateraz implementovaný len v podobe informácie o aktuálnom stave "vernostnej peňaženky" prihláseného zákazníka. Nakoľko zatiaľ nie sú bližšie špecifikované biznisové možnosti nasadenia vernostného systému, potenciálna implementácia hlbšej funkcionality (zoznamu úspechov, možnosti výberu odmien priamo v aplikácii a pod.) bola v tejto etape podľa odporúčania priemyselného partnera odložená.

Keďže je vernostný systém primárne zamýšľaný ako dodatočná motivácia pre spotrebiteľov, mobilnej aplikácie pre manipulantov sa jeho implementácia netýka.

2.3.8.3 Webová aplikácia

V rámci zapracovania vernostného systému do webovej aplikácie bolo pre prihlásených používateľov pridané pole, ktoré zobrazuje aktuálne vernostné body daného používateľa. Toto pole sa nachádza v profile používateľa a zobrazí sa po rozkliknutí dropdown tlačidla

v pravej hornej časti obrazovky. Zobrazené body sa získavajú z backendu vo funkcii getLoyaltyPoints(), ktorá je zavolaná pri každom načítaní stránky. Tieto body sú zobrazené len pre koncových zákazníkov, nakoľko slúžia ako motivácia k ich zaregistrovaniu.

2.3.9 Blockchain

Pre optimalizovanie komunikácie s blockchain databázou sme vytvorili novú funkciu CreateItemsBulk(). Táto funkcia je zodpovedná za odosielanie viacerých záznamov pri ich vytvorení ako jeden dopyt na blockchain databázu, ktorá je schopná ich takto spracovať rýchlejšie a efektívnejšie. Rovnako sme upravili aj počet záznamov, ktoré je možné poslať v jednom dopyte a dĺžku času, ako dlho backend časť čaká na vytvorenie novej skupiny záznamov, ktoré sa odošlú na uloženie blockchain databáze.

2.4 Testovanie

2.4.1 Interné testovanie

Implementované riešenia sa vždy pred nasadením na produkčný server patrične otestujú. Konkrétne ide o jednotkové testy jednotlivých implementovaných častí na danom komponente (napr. dôkladné otestovanie novej funkcionality na backende). Toto testovanie vykonáva člen tímu, ktorý danú funkcionalitu implementoval. Je dôležité, aby pokryl všetky možné scenáre. Následne sa pomocou integračných testov otestuje správnosť prepojenia jednotlivých komponentov v danej funkcionalite (prepojenie frontendu/mobilnej aplikácie a backendu).

Testy vykonávajú členovia tímu, ktorí sa danej problematike venovali na konkrétnom komponente systému a prípadné nedostatky konzultujú s členom, ktorý sa venoval súvisiacej náprotivnej časti (napr. vývojár mobilnej aplikácie testuje správnosť prepojenia aplikácie so serverom a nedostatky konzultuje s backend vývojárom). Upravený projekt môže byť nasadený na produkciu len za predpokladu, že všetky tieto testy sú vykonané úspešne.

2.4.2 Implementácia testov

Pre potreby testovania implementácie boli vytvorené unit testy pomocou JUnit5 frameworku. Z dôvodu obmedzeného času bola otestovaná len najdôležitejšia funkcionalita, ktorou je autorizačný modul, modul pre zásielku, modul pre produkt a modul pre výrobok. Testy pre autorizačný modul kontrolujú prihlásenie používateľa, registráciu a overenie registrácie. Testy pre výrobky a produkty overujú schopnosť vytvoriť nové inštancie a upraviť existujúce inštancie. Najkomplexnejšie testy sú implementované pre zásielky, kde sa kontroluje možnosť vytvoriť zásielku, ukončiť zásielku, priradiť ju dopravcovi alebo zmeniť jej polohu. Spolu bolo implementovaných 21 jednotkových testov.

2.4.3 CI/CD

Pre zjednodušenie testovania bola vytvorená konfigurácia pre GitHub Actions. Táto konfigurácia vytvára prostredie, v ktorom je aplikácia zložená, spoločne s databázou spustená a vykonajú sa implementované testy. Tento proces jednoducho a rýchlo odhalí značnú časť chýb, ktoré mohli vzniknúť v aplikácii počas implementácie nových funkcionalít.

2.4.4 Externé testovanie

V rámci spolupráce so spoločnosťou OLTIS bol zo strany partnera vyvinutý simulátor prepravy balíku (package) vlakovou dopravou. Tento simulátor sa dopytuje na backendovú časť nasadenú na adrese https://s-chain.fiit.stuba.sk/schain/. Spôsob dopytovania prebieha tak, ako je uvedené na stránke https://s-chain.fiit.stuba.sk/schain/swagger-ui/.

3 Bibliografia

1. **Holý, Matej, a iní.** *Inžinierske dielo, Tím 09: Zig-Zag Group.* [Online] 2020/2021. [Dátum: 24. november 2021.] https://web.archive.org/web/20211209200903/http://labss2.fiit.stuba.sk/TeamProject/2020/team09/public/main_dokumentacia/s-chain-inzinierske_dielo.pdf.

2. **Správa železnic, státni organizace.** Provozování dráhy. [Online] [Dátum: 24. november 2021.] https://provoz.spravazeleznic.cz/PORTAL/ViewArticle.aspx?oid=34462.

3. **Železnice Slovenskej republiky.** Číselník dopravných bodov železničnej siete SR. [Online] 13. november 2012. http://fpedas.utc.sk/~gasparik/SR_70.pdf.

4. **International Union of Railways.** European Rail Freight Corridors, Regulation 913/2010. [Online] 20. október 2015. https://uic.org/com/enews/nr/469/article/european-rail-freight-corridors.

5. **United Nations Economic Commission for Europe.** UN/LOCODE Code List by Country and Territory. *UNECE.* [Online] júl 2021. [Dátum: 24. november 2021.] https://unece.org/trade/cefact/unlocode-code-list-country-and-territory.

Príloha A Všeobecné informácie a závislosti

A.1 Server

Java verzia: 17

PostgreSQL verzia: 13.2

Lokálna adresa servera: http://localhost:**8080**

Lokálna adresa blockchain: https://localhost:7054

Lokálna adresa webového rozhrania: http://localhost:**8081**

Swagger adresa (API rozhrania): http://localhost:8080/swagger-ui/

Adresa servera: https://s-chain.fiit.stuba.sk/schain/

Závislosti zo súboru **build.gradle**:

- org.springframework.boot:spring-boot-starter-web
- org.springframework.boot:spring-boot-starter-validation
- org.springframework.boot:spring-boot-starter-security
- org.springframework.boot:spring-boot-starter-data-jpa
- org.springframework.boot:spring-boot-starter-mail
- io.jsonwebtoken:jjwt:0.9.1
- org.projectlombok:lombok
- com.google.zxing:core:3.4.1
- com.google.zxing:javase:3.4.1
- io.springfox:springfox-boot-starter:3.0.0
- org.jetbrains.kotlin:kotlin-stdlib-jdk8
- org.jetbrains.kotlin:kotlin-stdlib
- org.postgresql:postgresql:9.4-1203-jdbc4
- com.fasterxml.jackson.datatype:jackson-datatype-hibernate5:2.12.2
- com.querydsl:querydsl-jpa:4.4.0
- com.querydsl:querydsl-apt:4.4.0:jpa
- javax.persistence:javax.persistence-api:2.2
- javax.annotation:javax.annotation-api:1.3.2
- org.hyperledger.fabric:fabric-gateway-java:2.2.0
- com.h2database:h2:1.4.200
- org.springframework.boot:spring-boot-starter-test
- org.assertj:assertj-core:3.17.2

- org.apache.poi:poi:3.9
- org.apache.poi:poi-ooxml:3.9
- org.reflections:reflections:0.10.2
- com.opencsv:5.6
- com.nhaarman:mockito-kotlin:1.6.0
- com.github.javafaker:javafaker:1.0.2
- org.yaml:snakeyaml:1.27

A.2 Webové rozhranie

Závislosti zo súboru **package.json**:

- axios: 0.20.0,
- bootstrap: 4.5.3,
- bootstrap-vue: 2.17.3,
- core-js: 3.6.5,
- qrcode: 1.4.4,
- qrcode.vue: 1.7.0,
- vee-validate: 3.4.5,
- vue: 2.6.12,
- vue-google-charts: 0.3.3,
- vue-qrcode-reader: 2.3.14,
- vue-router: 3.4.7,
- vue-xlsx: 0.2.1,
- vuelidate: 0.7.6,
- vuex: 3.5.1,
- xlsx: 0.16.9

A.3 Mobilná aplikácia pre zákazníkov

Kotlin verzia: 1.4.21

Nav verzia: 2.3.0

Závislosti z projektového súboru **build.gradle**:

- com.android.tools.build: gradle: 7.0.3
- org.jetbrains.kotlin: kotlin-gradle-plugin: 1.4.21
- com.google.dagger: hilt-android-gradle-plugin: 2.28.3-alpha
- androidx.navigation: navigation-safe-args-gradle-plugin: 2.3.0

A.4 Mobilná aplikácia pre manipulantov

Kotlin verzia: 1.4.21

Nav verzia: 2.4.1

Závislosti z projektového súboru **build.gradle**:

- com.android.tools.build: gradle: 7.0.3
- org.jetbrains.kotlin: kotlin-gradle-plugin: 1.6.10
- com.google.dagger: hilt-android-gradle-plugin: 2.38.1
- androidx.navigation: navigation-safe-args-gradle-plugin: 2.4.1

Príloha B Technická dokumentácia

Inštalácia a spustenie

Predpoklady:

- Linux stroj (testované na Ubuntu 18.04)
    - možné alternatívy: Virtuálny stroj
- Node, npm, Docker, Docker-Compose, OpenJDK 17, Gradle 7.2
- Mobilné aplikácie:
    - nainštalované Android Studio
    - nastaviť si Gradle Build Variants – local a local_emulator (lokálny BE), remote (nasadený BE)

Postup pri inštalácii:

1. stiahnuť zdrojové kódy z GitHub repozitárov
2. zabezpečiť, že sú umiestnené podľa podmienok (blockchain a server v jednom adresári)
3. vykonať kroky pre spustenie servera, v rámci ktorých je zabezpečené aj spustenie blockchainu a frontendu

Inštalácia mobilných aplikácií do Android zariadenia je možná pomocou vytvorených inštalačných APK súborov **zigzaggroup.schain.mobile.v1.0.apk** (zákaznícka aplikácia), resp. **zigzaggroup.schain.mobile_handler.v1.0.apk** (aplikácia pre manipulantov) pomocou ľubovoľného nástroja na inštaláciu balíkov. APK súbory sú dostupné v jednotlivých GitHub repozitároch v sekcii Releases.

Spustenie blockchainu

Server vykoná túto operáciu automaticky. Tieto kroky nie sú povinné.

1. navigovať sa do blockchain časti projektu a spustiť beforeFirstStart.sh
2. vytvoriť blockchain organizácie (spustiť) pomocou start.sh

- je možné, že na správne spustenie bude najprv nutné nainštalovať potrebné závislosti pomocou príkazu npm install

Spustenie servera

1. navigovať sa do server časti projektu a spustiť **start-setup.sh**

- tento proces vykoná potrebnú úvodnú inicializáciu a spustí server. V prípade opakovaného spustenia je potom postačujúce spustiť server pomocou **start-dev.sh**
- možné problémy:
    - v prípade aktualizácie servera/blockchainovej časti je pravdepodobne nutné znovu spustiť **start-setup.sh**

Spustenie Vue.js frontend

Server vykoná túto operáciu automaticky. Tieto kroky nie sú povinné.

1. navigovať sa do frontend časti
2. nainštalovať potrebné moduly pomocou príkazu npm install
3. spustiť webové rozhranie pomocou príkazu npm run serve

Príloha C Protokol z testovania

1. OLTIS Group, a. s: Posudek na prototyp týmu č. 13: ZigZag2 Group

1.1. Úvod

V tomto dokumentu členové společnosti Oltis hodnotí a posuzují dosavadní práci S-Chain studijní skupinou, která na základě odsouhlaseného zadání měla za cíl zlepšit systém sledování produktů a obohatit jej o klientskou mobilní aplikaci a mobilní aplikaci pro dopravce. Hodnocení podléhá zvolený postup, kvalita odvedené práce a dosáhnuté výsledky implementace prototypu.

1.2. Hodnocení prototypu

Prototyp se skládá ze tří funkčních celků. Webu S-chain, ke kterému lze přistupovat ve více rolích (manufacturer a carrier) a dvou funkčně menších mobilních aplikací pro klienta a pro dopravce.

Abychom byli schopni poskytnout co nejlepší zpětnou vazbu, vznikl testovací scénář, kde jsou hodnoceny jednotlivé funkční celky z pohledu uživatele S-Chain. Byla použitá vzorová data, tedy je možné, že v praxi uživatelé narazí na komplikace, které jsme nemohli nyní najít/hodnotit či navrhnout jako možná zlepšení.

Součástí hodnocení je testovací protokol v03FP, který vznikl pro poskytnutí zpětné vazby vývojářskému týmu. Testovací protokol prochází případ užití a zároveň poukazuje na úzká místa a možnosti budoucího vývoje. Z pohledu zadavatele je projekt velmi dobře zpracován a poskytuje kvalitní oporu uživateli působícím v supply chain managementu.

1 Web S-chain: manufacturer@company.com

https://s-chain.fiit.stuba.sk v roli: manufacturer@company.com “heslo123”.

- **Product** – z pohledu systému je chápán jako vzor neboli předloha pro výrobek.
- **Item** – položka, věc chápána jako instance produktu, která odkazuje na product ID a má sama své ID.
- **Shipment** – zásilka, z pohledu systému S-chain chápána jako instance naložená např. na paletě.

Napříč dodavatelsko-odběratelského řetězce (supply chain) se předpokládá naskenování příslušného QR kódu a tím ověření kde se aktuálně daný výobek nachází a jaká je jeho historie pohybu.

1.1 Dashboard

1.1.1 Search your shipment or item:

1.1.1.1 Enter ID – po zadání S:53 lze vidět detaily

1.1.2 Shipment details – opravit formát druhé souřadnice (Longitude)

a. Tlačítko Update shipment
   i. Update shipment information
   ii. location: ruční zadání OK , nápady na zlepšení: přidat dialog pro potvrzení, + přidat tlačítko pro vyplnění koordinátů po výběru z mapy (momentálně přes enter v textu, ne vždy fungovalo)

b. Tlačítko show/hide history: funkce tlačítka ok
   i. Ukáže menu lokace – ok
   ii. Opravit formát koordinátů

1.2 QR Code Reader

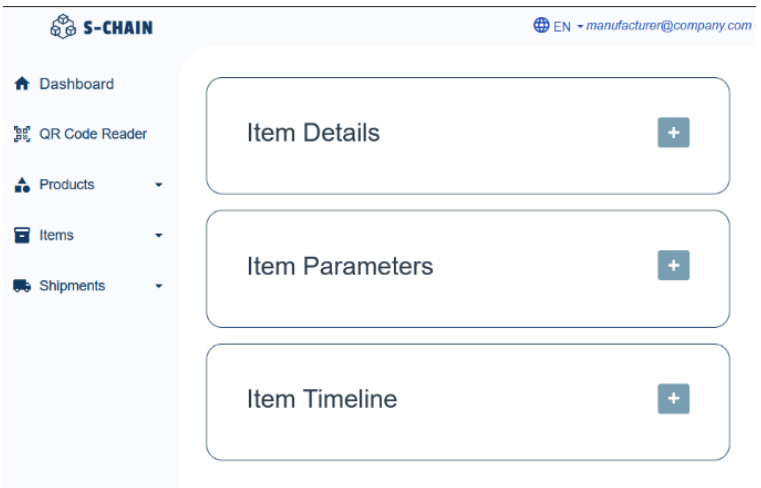
1.2.1 Scan QR Code

Aplikace zapne kameru v notebooku, vzorový QR kód lze naskenovat, následně zobrazí detail.

1.2.2 Load image with QR code

Otevře možnost pro importování obrázku se vzorovým kódem. Po nahrání vzorového QR kódu se otevře detail produktu.

- Item Details + QR code, shipment ID, Name, Product, Category, Description
- Item Parameters + Serial number, Product number.
- Item Timeline + zobrazí lokaci výrobku na mapě, GPS, SR70 číslo a název, update by, update info.



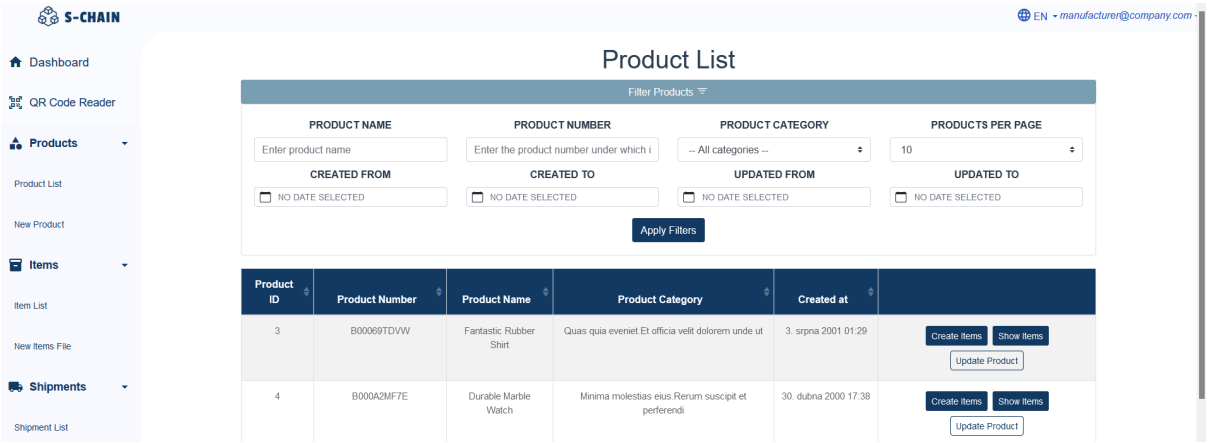
1.3 Products

Z hlediska funkcionality produktu: produkt je chápán jako vzor, ze kterého se vytvářejí jeho jednotlivé instance (items). Produkt může dále pozostávať z ďalších produktov (subproducts).

1.3.1 Product List

Výrobce si může jednotlivé produkty najít pomocí aplikování filtrů, které jsou mu zobrazeny a seřazeny v tabulce. Jednotlivé filtry jsou jmenovány níže. Ve funkcionalitě nebylo zjištěno v testovacím provozu žádných závad.

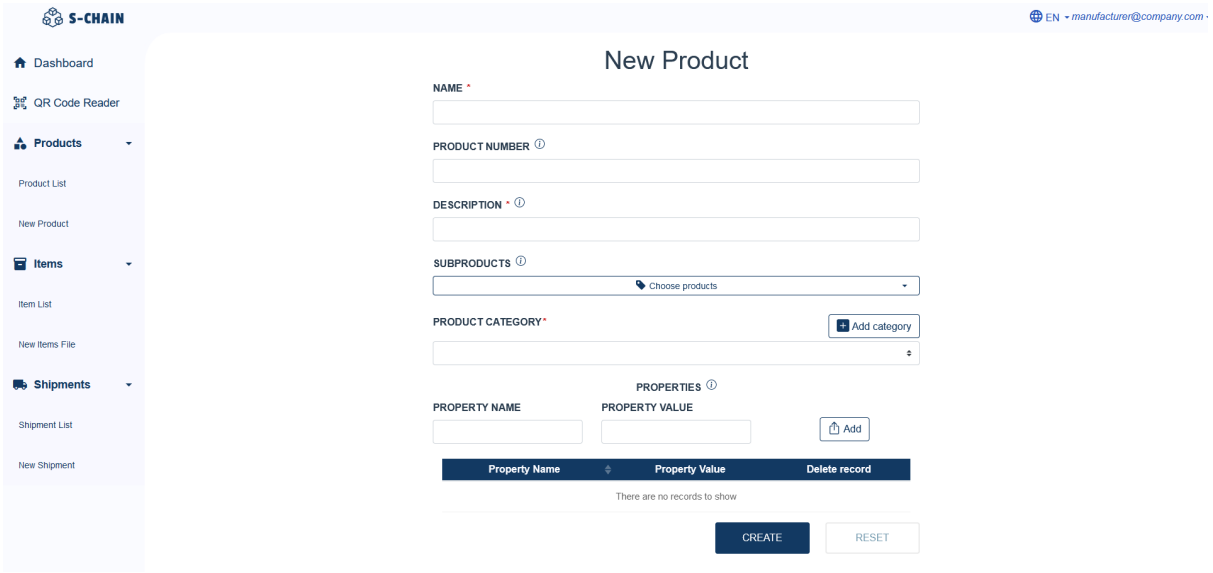
- Product Name
- Product Number
- Product Category
- Products per page
- Created from
- Created to
- Updated from
- Updated to
- Apply Filter:button



Při zadávání nového produktu je povinný název produktu, jeho popis a kategorie. Kategorie lze přidávat. Subprodukty lze vybrat ze seznamu těch, které již jsou v systému zadány.

1.3.2 New Products

- Name
- Product Number
- Description
- Subpproducts
- Product Category + Add Category
- Property Name
- Property Value
- Add:button
- Reate:Button
- Reset:button



1.4 Items

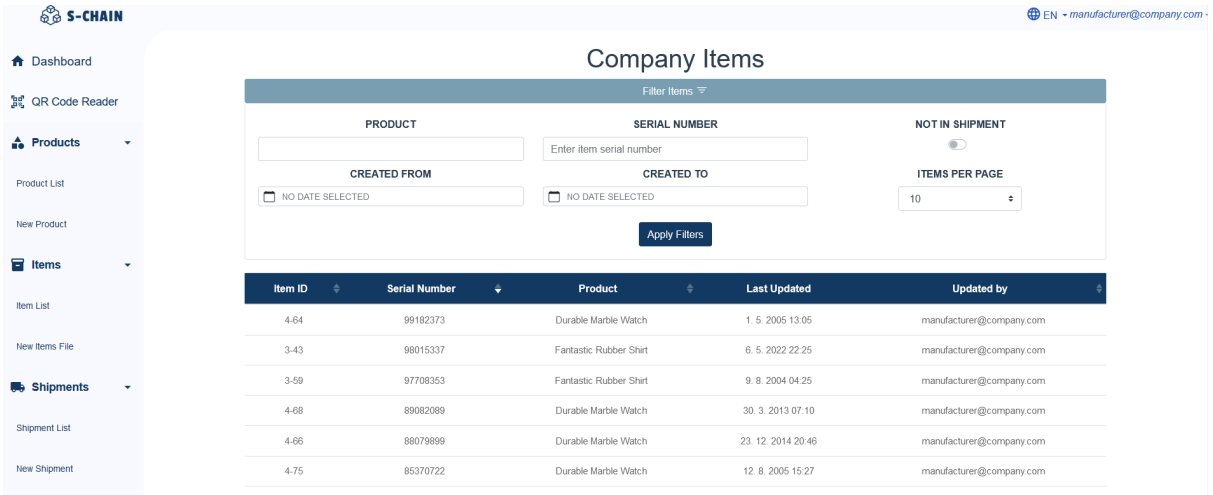
Instance, výrobku s jeho pruduktovým číslem (vazba na product ID).

1.4.1 Item List – Filter items

V záhlaví s názvem Company Items je k dispozici výrobci několi filtrů viz níže. V aktuálním nastavení jsou všechny potřebné a funkční. Po zvolení parametrů k filtrování uživatel volí aplikování a jsou mu zobrazena potřebná data v přehledné tabulce, která lze vzestupně či sestupně řadit.

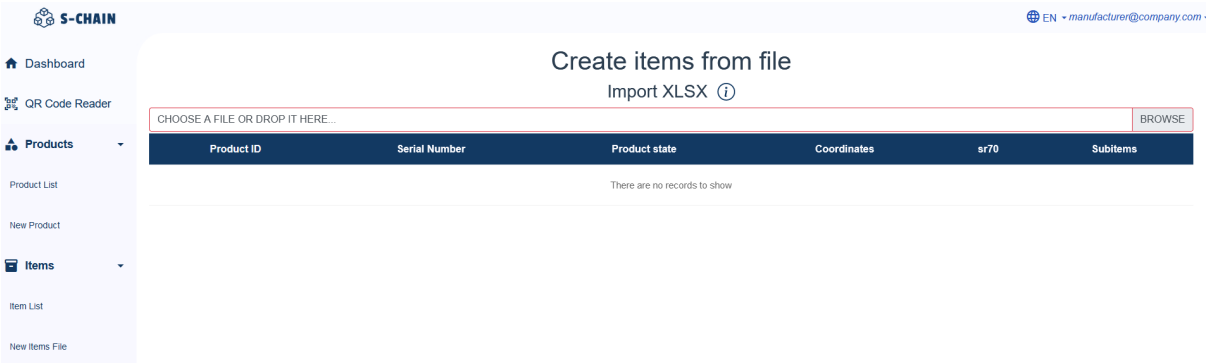
- Product
- Serial Number
- NOT in shipment – bolean
- Created from
- Created to
- Items per page
- Apply filters: button

Návrh na zlepšení: V menu vlevo by mohlo být vidět, která možnost byl vybrána.



1.4.2 New items File

Přidat vzorovou šablonu ke stáhnutí pro uživatele. V tomto stavu nelze importovat data.



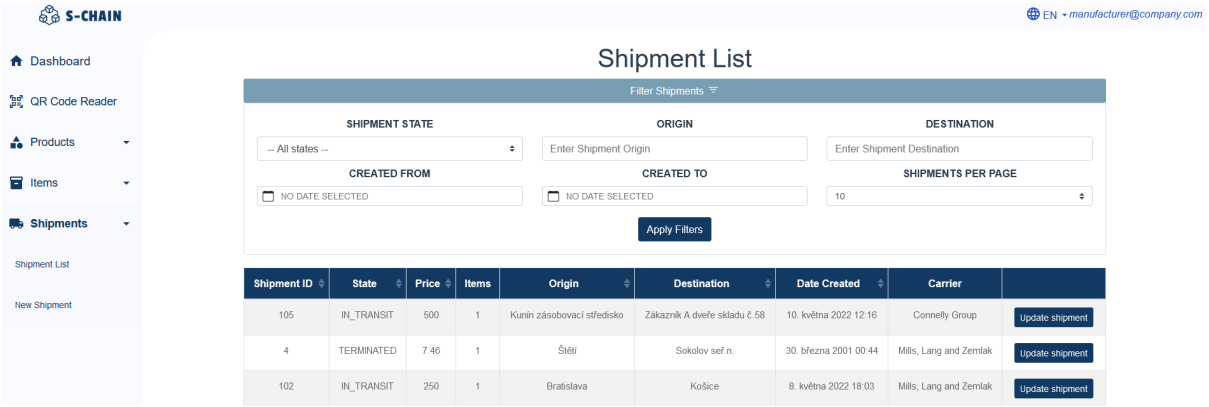
Create Item – také při vytváření nové item např. z produktu by bylo vhodné doplnit GPS souřadnice z mapy, na které uživatel může origin location najít. Resp. nemožné vbrat např. SR70!

1.5 Shipments

Zásilka neboli konkrétní výrobek (který putuje k zákazníkovi s jejím vlastním ID).

1.5.1 Shipment List – Filter Shipments

- Shipment state
- Origin
- Destination
- Created from
- Created to
- Shipmentss per page
- Apply filters: button



1.5.2 New Shipment

- Description:
- Estimated price - měla by být uvedena měna nebo dát na výběr z měn, které lze zvolit, když už takový parametr existuje.
- Carrier Company: Hue, Kerluke and Kerluke
- Origin location: změnit na „Destcription of origin location“
    - GPS – při výběru z mapy nefunkční, v ideálním případě přidat generované GPS do polí k tomu určených

        - SR70 – při výběru této varianty ideálně zobrazit nejen SR70 kód ale také lokaci
        - UNLOCODE - ok
- Destination Location: změnit na „Destcription of destination location“
    - GPS při výběru z mapy nefunkční, v ideálním případě přidat generované GPS do polí k tomu určených
    - SR70 při výběru této varianty ideálně zobrazit nejen SR70 kód ale také lokaci
    - UNLOCODE - ok
- Search by product, dva filtry

2 S-chain web: carrier@company.com

Testovací protokol 10.května 2022. Testováno na: https://s-chain.fiit.stuba.sk v roli: “heslo123”.

- **Shipment** – zásilka, z pohledu systému S-chain chápána jako instance naložená např. na paletě.
- **Cargo** – náklad obsahuje zásilky

2.1 Dashboard

2.1.1 Search your shipment or item:

2.1.1.1 Enter ID

2.1.1.2 Cargo details – opravit formát souřadnic

a. Tlačítko Update Cargo
    i. Update Cargo information
    ii. Update Cargo state – created/terminated
    iii. Update Cargo location
    iv. Update track ID – umožní update sledovací číslo
    v. Deliver Cargo – nastaví na doručeno

2.2 QR Code Reader

2.2.1 Scan QR Code

Aplikace zapne kameru v notebooku, vzorový QR kód lze naskenovat, následně zobrazí detail.

2.3 Load image with QR code

Otevře možnost pro importování obrázku se vzorovým kódem. Po nahrání vzorového QR kódu se otevře detail produktu.

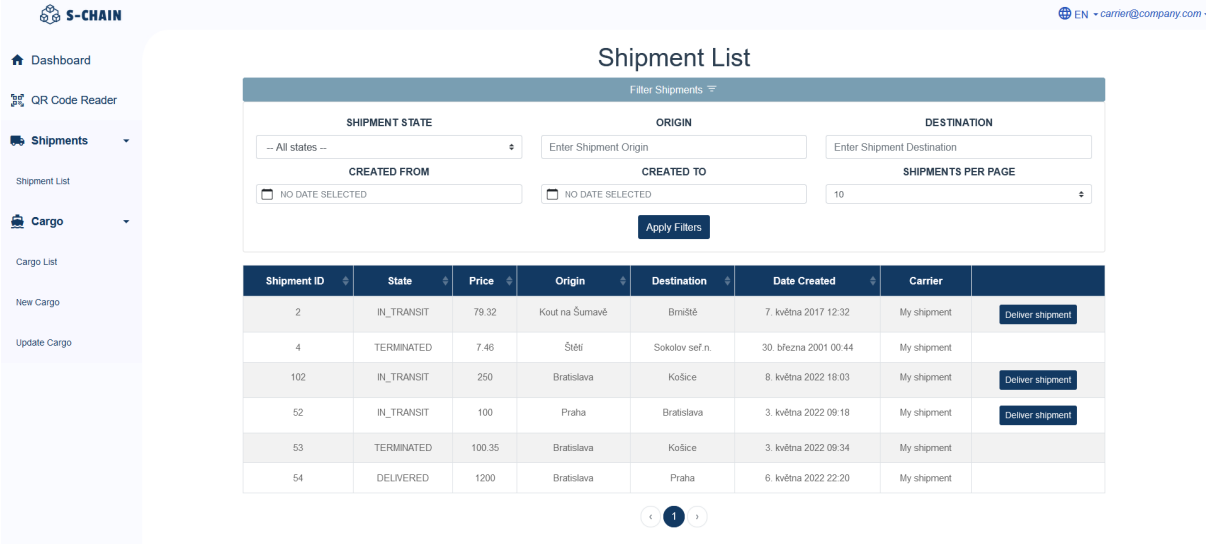
2.4 Shipments

Zásilky mohou být sdružovány do nákladu neboli CargoProduct List

Výrobce si může jednotlivé zásilky najít pomocí aplikování filtrů, které jsou mu zobrazeny a seřazeny v tabulce. Jednotlivé filtry jsou jmenovány níže. Ve funkcionalitě nebylo zjištěno v testovacím provozu žádných závad.

- Shipment state
- Origin

- Destination
- Created from
- Created to
- Shipments per page



Při zadávání nového produktu je povinný název produktu, jeho popis a kategorie. Kategorie lze přidávat. Subprodukty lze vybrat ze seznamu těch, které již jsou v systému zadány.

2.5 Cargo

Instance, výrobku s jeho pruduktovým číslem (vazba na product ID).

2.5.1 Cargo List

V záhlaví s názvem Company List je k dispozici dopravci několik filtrů viz níže. V aktuálním nastavení jsou všechny potřebné a funkční. Po zvolení parametrů k filtrování uživatel volí aplikování a jsou mu zobrazena potřebná data v přehledné tabulce, která lze vzestupně či sestupně řadit.

- Cargo state
- Origin
- Destination
- Created from
- Created to
- Cargos per page

Návrh na zlepšení: V levém menu by mohlo být vidět, která možnost byla vybrána. Kde se nyní nacházíme.

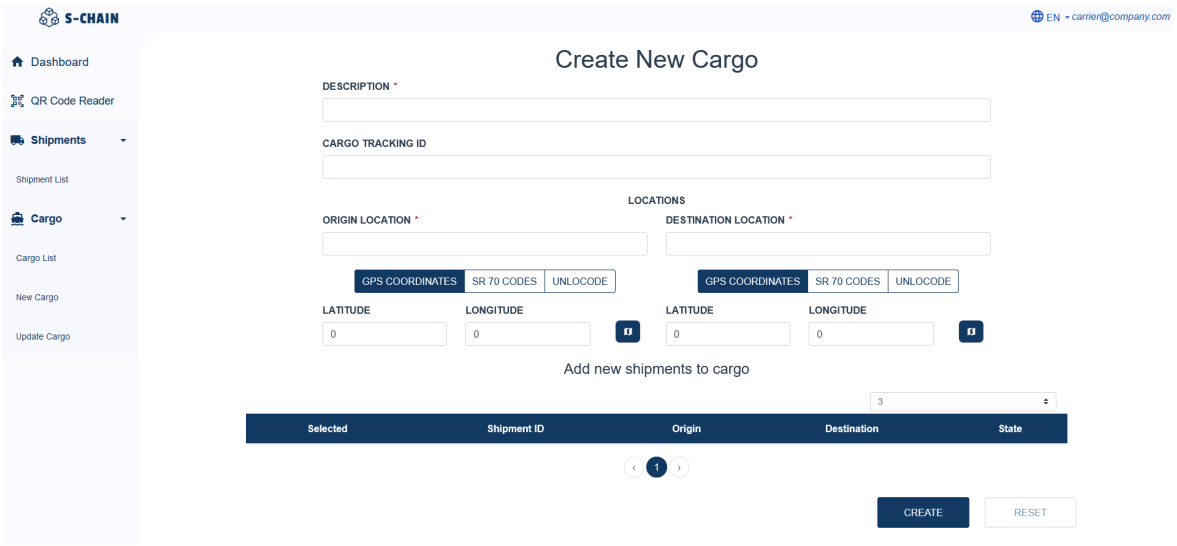
2.5.2 New Cargo

Vytváření nového nákladu

- Description:
- Cargo tracking ID
- Origin location: změnit na „Destcription of origin location“
  - GPS – při výběru z mapy nefunkční, v ideálním případě přidat generované GPS do polí k tomu určených
  - SR70 – při výběru této varianty ideálně zobrazit nejen SR70 kód ale také lokaci

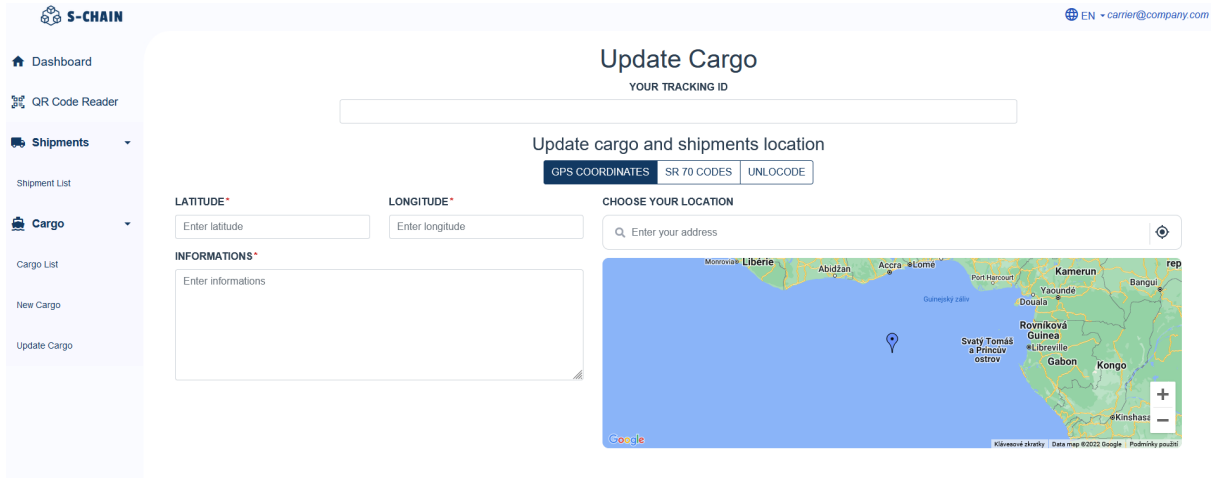
- UNLOCODE - ok

- Destination Location: změnit na „Destcription of destination location“
  - GPS při výběru z mapy nefunkční, v ideálním případě přidat generované GPS do polí k tomu určených
  - SR70 při výběru této varianty ideálně zobrazit nejen SR70 kód ale také lokaci
  - UNLOCODE - ok



2.6 Update Cargo

Pro zadání aktuální polohy.



3 Mobilní aplikace – mobile master verze pro klienta

3.1 Úvodní obrazovka aplikace nabízí možnosti vyhledání výrobku, přihlášení nebo registraci ke službě.

3.1.1 Search – vyhledání podle kódu,

3.1.2 Scan QR code – sken OR kódu

3.1.3 Pick QR code – vložení QR kódu výrobku ze souboru v telefonu

3.2 Item details

Pokud se výrobek v systému dohledá, zobrazí se Item details neboli podrobnosti o výrobku jako jsou Name, Item ID, Type, Description, Parameters a Subitems, pokud jsou u výrobku.

3.3 Item history

Z detailu výrobku se dá planule přejít do historie jeho cesty a podívat se, kde se lokace nachází na mapě.

4 Mobilní aplikace – mobile handler – přihlášení pro dopravce

4.1 Úvodní obrazovka

Dopravce se musí přihlásit (Login), čímž splní nutnou podmínku autentizace a následné authorizace.

4.2 Home

V domovské obrazovce pak dopravce vybírá, se kterým nákladem bude manipulovat – applikace může služit zaměstnancům dopravce ve skladě či jinde po cstě, kde se děje nějaká překládka či jiná významná událost na cestě zboží. Po zadání potřebného identifikačního čísla či naskenování jeho QR kódu na místě či vyhledání QR kódu ze souboru v přístroji sedotavá dopravce do další části aplikace. Je zde také možnost odhlásit se a přihlásit k jinému účtu v případě potřeby.

4.3 Cargo details

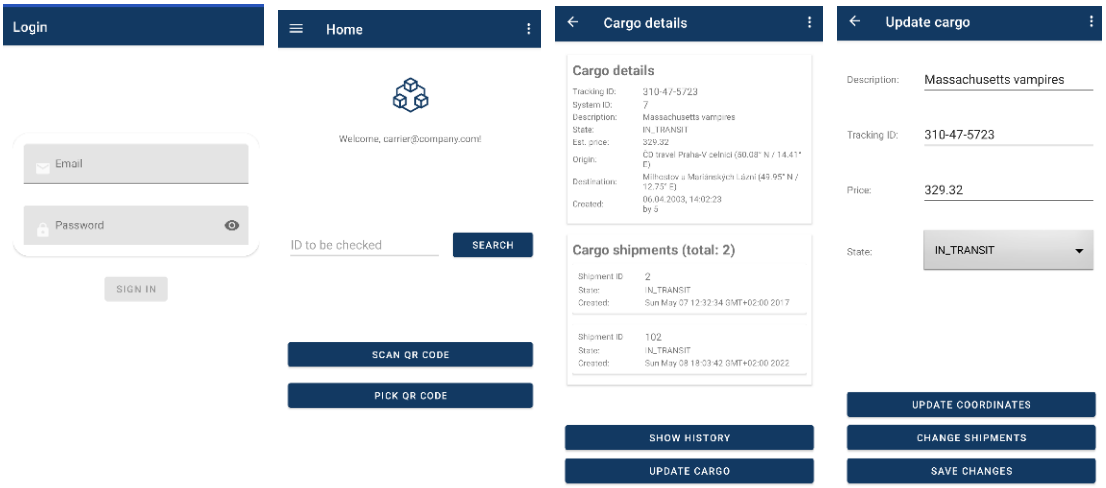
4.3.1 Cargo details

- Tracking ID
- System ID
- Description
- State: - v testovacím provozu IN_TRANZIT, TERMINATED, DELIVERED
- Est. price – cena nákladu
- Origin
- Destination
- Created
- Updated

4.3.2 Cargo shipments (počet Shipments v Cargo)

Ke každé jednotlivé zásilce pak údaje:

- Shipment ID
- State
- Created



4.3.3 Show history

Jak je vidět na výřezu z aplikace jednotlivé pohledy umožní také ukázat historii pohybu např. průjezd jednotlivými nácestnými body/stanice a také kdo v dané lokaci se zásilkou manipuloval.

4.3.4 Update Cargo

Aplikace umožní přechod mezi jednotlivými stavy nákladu (IN_TRANZIT, TERMINATED, DELIVERED) dle požadavků a situace dopravce. Je zde umožněno změnit aktuální souřadnice nákladu dle jeho aktuální polohy, změnit zásilky, které náklad obsahuje a uložit dnaé změny.