Slovenská technická univerzita Fakulta informatiky a informačných technológii
Ilkovičova 3, 812 19 Bratislava

Inžinierske dielo

Tím 09: Zig-Zag Group

Akademický rok: 2020/2021
Predmet: Tímový projekt

Vedúci tímu: Ing. Rastislav Bencel, PhD.
Členovia tímu: Matej Holý
Adrián Libiak
Andrej Šulavík
Daniel Minárik
Kamil Macek
Matej Petráš
Tomáš Babjak

Slovník pojmov

BC	Blockchain
BE	Backend
Cargo	Označuje skupinu zásielok zoskupených za účelom prepravy
FE	Frontend
ID	identifikátor
MS	Management Story
PR	Pull Request
Produkt	Jedná sa o všeobecný popis nejakého tovaru, ktorý výrobca vyrába
S-Chain	Projekt, na ktorom ako tím spoločne pracujeme
US	User Story
Výrobok	Jedinečný, konkrétny kus výrobku, ktorý výrobca vyrobil
VzV	Výrobok z výrobkov. Jedná sa o kompozíciu výrobkov za účelom vytvorenia nového výrobku (napríklad motor, kolesá a karoséria vytvoria auto).
Zásielka	Označuje skupinu výrobkov zoskupených za účelom prepravy
Zig-Zag Group	Názov nášho tímu

Obsah

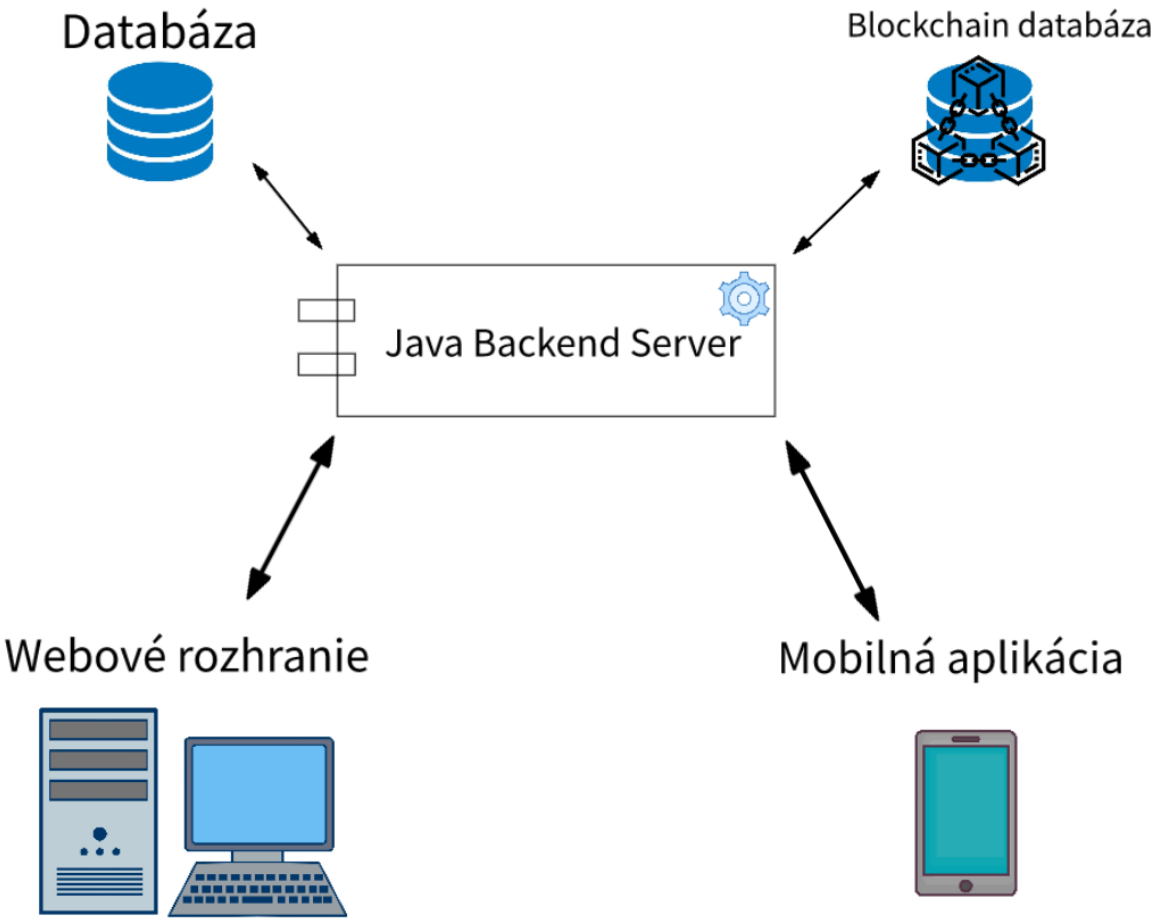
Big Picture

Úvod

V tomto dokumente sa nachádza opis realizovaného projektu S-Chain vytvoreného v rámci predmetu Tímový projekt v rokoch 2020-2021.
Dokument obsahuje technické detaily a opisuje výhradne projekt. Nachádza sa tu pohľad na štruktúru projektu, použité technológie a spôsob ich aplikácie. Na úvode popisujeme celkový prehľad systému a ciele projektu. Neskôr pokračuje analytická časť, za ktorou nasleduje návrh a implementácia. Dokument je písaný tak, aby sa prezentoval informácie úplne ale plynulo.

Celkový pohľad na systém

V tejto krátkej kapitole predstavíme naše riešenie v technických detailoch, lenže iba v základnom prehľade. Detailnejší opis bude v neskorších kapitolách zobrazený a popísaný 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 v zjednodušenej podobe.



Naše riešenie má za úlohu poskytnúť informácie o procesoch výroby a prepravy, ktorými si výrobok prechádza predtým, ako pristane na pultoch v predajniach. Výrobca si bude môcť evidovať svoje produkty a následne vytvárať výrobky, ktoré sa budú môcť pomocou jedinečného identifikátora aktualizovať[1]. Všetky aktualizácie so sebou prinesú dodatočné informácie ako pre výrobcu, tak neskôr aj pre zákazníka.
Výrobca si zároveň môže svoj vyrobený a expedovaný tovar sledovať, než sa dopraví na miesto určenia. Dopravca, ktorý bude takisto využívať našu aplikáciu, bude zaznamenávať vykladanie, nakladanie a inú manipuláciu s tovarom. Tieto záznamy spolu s informáciami z výrobného procesu sa nakoniec zobrazia na požiadanie zákazníkovi po naskenovaní jedinečného identifikátora. Zákazník uvidí, čím si tovar v jeho rukách prechádzal, celou cestou od výroby až po doručenie do obchodu. Záznamy budú okrem dátumov a časov aktualizácií obsahovať aj krátky popis, ktorý bude odhaľovať bližšie informácie o záznamoch.
Zákazník a takisto aj samotný výrobca môže vidieť zmeny, ktorými si teda tovar prechádzal. Lenže takéto zobrazovanie zmien nič neznamená, ak sa ani jeden z používateľov systému nemôže spoľahnúť na dôveryhodnosť poskytovaných informácií. Ako je možné zabezpečiť, aby sa nemohlo neoprávneným spôsobom zasahovať do záznamov a klamať 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 zmeny.
Pri rozoberaní štruktúry systému sme prišli na jednotlivé komponenty. Základom je serverové sídlo, ktoré je vytvorené pomocou Java Spring Boot rámca a nástroja Gradle. Tento server spravuje PostgreSQL databázu, v ktorej sa ukladajú dáta, ktoré nie sú nevyhnutné na zaručenie transparentnosti (napríklad prihlasovacie údaje). Táto databáza bude synchronizovaná podľa potreby s blockchain databázou. Rozhodnutie využitia databázy PostgreSQL a neukladania všetkých údajov do blockchain je z dôvodu zabezpečenia efektivity[2].
Blockchain je vytvorený pomocou IBM Hyperledger rámca (konkrétne ide o Hyperledger Fabric), ktorý ponúka prístup do blockchain pomocou autentifikácie. Nie je vhodné použiť verejný blockchain, pretože sa budú evidovať informácie aj počas výroby a prepravy výrobkov, preto je nutné umožniť mieru súkromia pri prístupe a aktualizácii dát.
Na server sa budú pripájať zariadenia pomocou prehliadača, preto je nutné vytvoriť webové rozhranie pre náš produkt. Z pohľadu spracovania webovej verzie je použitý rámec Vue.js 2.6.12 s využitím Bootstrap-vue 2.17.3 (všetky závislosti a presné verzie sú uvedené v **prílohe B**). Predpokladáme aj použitie mobilného zariadenia, preto je identifikovaná Android mobilná aplikácia vytvorená pomocou jazyka Kotlin.

Ciele projektu na zimný semester

Jedným z primárnych cieľov je detailné oboznámenie sa s problémovou oblasťou dodávateľského reťazca. Spolupráca s externou spoločnosťou nám objasní spôsoby prepravy a umožní prehľad v biznis oblasti. Počas zimného semestra je zároveň našim cieľom sa zoznámiť bližšie s technológiou blockchain a s jej využitím je naša priorita vytvoriť

[1] Bližšie informácie k terminológii sú v časti Návrh.
[2] Blockchain je pomalší ako relačné databázy a zároveň sa nie vždy vyskytuje potreba zabezpečovať transparentnosť niektorých typov dát (napr. prihlasovacie údaje).

jednoduchý prototyp, ktorý podporuje základnú funkcionalitu nášho produktu. Dôležité je pre nás mať po zimnom semestri konkrétnu predstavu o:

- Architektúre nášho riešenia: Systematickým pristupovaním k analýze problémovej oblasti je hlavné vytvoriť takú architektúru systému, aby sme vedeli pokryť požiadavky projektu. Zároveň je nevyhnutné zabezpečiť škálovateľnosť riešenia, pretože sa ráta s vysokým nárastom položiek v prípade využitia projektu. Medzi zvýšené počty položiek môžeme považovať nové a nové výrobky, novo-registrované spoločnosti a takisto rôzne množstvo záznamov pri manipulácii s výrobkami pred ich doručením na konečné miesto,
- Biznis doméne: Predstava o tom, ako funguje výroba a akým spôsobom sa s tovarom manipuluje pri preprave je nevyhnutná na vytvorenie nielen používateľsky prívetivého prostredia, ale zároveň aj funkčného. Priorita nášho projektu je transparentnosť, preto je podstatné túto vlastnosť zabezpečiť.
- Celkovom riešení projektu: Zohľadnením nadobudnutých vedomostí počas semestra by sme mali byť schopní ako tím teoretizovať o naplnení projektu a vedieť určiť spôsob jeho vytvorenia. Realizácia prototypu nám výrazne pomôže zistiť výhody a obmedzenia nami zvolených technológií.

Ciele projektu na letný semester

Počas zimného semestra sa podarilo vytvoriť základnú verziu nášho riešenia. Naše riešenie preto počas letného semestra potrebujeme obohatiť o ďalšiu funkcionalitu a vytvoriť mobilnú aplikáciu. Budeme komunikovať s Oltis Group na to, aby sme zabezpečili správnosť nášho riešenia pomocou rozhrania, ktoré nám Oltis Group vystaví. Na tomto rozhraní budeme dynamicky testovať našu aplikáciu a zachytávať nedostatky simulovaním reálneho využitia nášho riešenia.

Ďalším z cieľov je dostať riešenie do stavu, v ktorom bude dostatočne reprezentatívne na zúčastnenie sa súťaže TP Cup. V rámci súťaže pripravíme prezentačné materiály na to, aby sme opodstatnili potrebu riešenia a aplikovanosť na trhu.

Dôležitou súčasťou letného semestra je ukončenie semestra, kedy skontrolujeme okrem riešenia aj vytvorenú dokumentáciu a ostatné súvisiace materiály. Podstatné bude zanechať dobrý podklad pre pokračovanie iných tímov na našom projekte neskôr. Na záverečnú revíziu zohľadníme:

- Prioritne funkcionalitu a pokrytie celého nášho riešenia
- Dokumentáciu, aktuálnosť diagramov a úplnosť textu
- Webstránku - funkčnosť odkazov, dostupnosť všetkých zdrojov
- Podklady pre TP Cup (video, prezentácia a podobne)

Moduly systému

Na zachytenie systému v modeloch sme zvolili jazyk UML. Týmto spôsobom budeme môcť vytvárať konzistentné diagramy a budeme môcť svoju prácu efektívne zdieľať. **Veľmi silno odporúčame** použiť nástroj **Enterprise Architect** (ideálne verzia *15.2 alebo novšia*) minimálne na analýzu diagramov, ktoré sú prezentované v tomto dokumente. Všetky diagramy sú v ňom navrhnuté a je vhodné si okrem tohto dokumentu prehliadnuť diagramy "ručne" v spomínanom nástroji. Dôvodom je, že vzhľadom na veľkosť sa nedá diagramy vždy efektívne a prehľadne vložiť do tohto dokumentu.

Analýza

V rámci analýzy prebehlo viacero prieskumov doménových oblastí. Prvá nevyhnutná kontrola prebehla v rámci blockchain sietí a ich ponuky na trhu. Náš tím prešiel viaceré alternatívy a z nich vybral IBM Hyperledger Fabric ako vhodnú možnosť. Dôvodov bolo viacero[3]:

- Nedostupnosť verejnosti (tzv. *permissioned*): údaje v rámci blockchain nie sú prístupné verejnosti a účastníci transakcií sa navzájom poznajú (sú rozoznateľní). Z tohto dôvodu nie je možné pristupovať k dátam, ktoré ešte nemajú byť prístupné, jednoducho z verejnosti a takisto je dohodnuté (samotným používaním tohto blockchainu), že členovia budú dodržiavať stanovené pravidlá transakcií.
- Výkonnosť: Nie je potreba mať perfektný, bezchybný konsenzus, pokiaľ sa predpokladá, že sa blockchain používa pod dôveryhodnou autoritou (napríklad v rámci jednej firmy). Toto umožňuje efektívnejší prístup pri realizovaní transakcií a je takisto možné konsenzus dobre prispôsobovať.
- Bezpečnosť: Hyperledger Fabric umožňuje vytvorenie privátnych dát. Vytvorením podsiete sa dajú potom v rámci blockchain posielať dáta medzi niektorými špecifickými klientmi súkromne, a tým pádom sa zvyšuje bezpečnosť komunikácie dôveryhodných klientov.

Ako ďalší krok sa v rámci našej analýzy vykonával prieskum existujúcich riešení. Zatiaľ čo blockchain riešení v oblasti dodávateľského reťazca sme našli veľmi málo, inšpirovali sme sa aj riešeniami ohľadom sledovania zásielok pri bežnej preprave.

Existujúce riešenia

Momentálne sú na trhu dostupné riešenia, ktoré sa zaoberajú problematikou dodávateľského reťazca. Jedno riešenie sa nazýva Supply Chain Solutions[4]. Tento projekt ponúka efektivitu a zníženie nákladov pre biznis. Okrem toho ponúkajú spracovanie a manažment transportu. Nevýhodou práve tohto riešenia je, že neberie do úvahy koncového zákazníka. Samozrejme je benefitom pre koncového zákazníka napríklad rýchlejšie doručenie, lenže zákazník nie je prioritou pri tomto riešení.
Inou nevýhodou je, že nie je spomenutý spôsob výmeny informácií a predovšetkým nie je spomenuté to, ako je komunikácia zabezpečená a transparentná. V dokumentoch projektu

[3] Informácie boli získané z online zdroja Hyperledger Fabric: https://hyperledger-fabric.readthedocs.io/en/latest/ [citované 24. Nov. 2020]
[4] Dostupné na adrese: https://www.scsolutionsinc.com [citované 1. Máj 2021]

sa nespomína blockchain a jeho využitie jeho schopností zabezpečiť dôveryhodnú a transparentnú komunikáciu všetkých zúčastnených aktérov v procese dodávateľského reťazca.

Ďalšie prezentované riešenie v rámci dodávateľského reťazca sa nazýva MPO Customer Chain Control[5]. Táto platforma, podobne ako v predošlom príklade, sa sústredí na zefektívnenie prepravného procesu. Ako je uvedené priamo na ich stránke: "*service options, faster delivery, reliable performance, and lower costs*". V rámci našej analýzy nás najviac zaujíma, že poskytujú obojstranný prehľad a kontrolu v reálnom čase (preložené z "*real-time end-to-end visibility and control*"). Hlavným záujmom je, že výkon počas doručovania tak efektívny a cenovo zvýhodnený oproti iným prístupom ako sa dá. Nevidíme však žiaden zámer obohatenia a dôležitosti postavenia koncového používateľa ale skôr výpomoc výrobcom poskytnutím kontroly nad doručovacím procesom.

Iný projekt, nazvaný Hillebrand[6], takisto poskytuje riešenie v rámci dodávateľského reťazca. Znovu sa jedná o podobné riešenie ako v oboch predošlých prípadoch. Rozdielom je, že zameranie je na prepravu a konkrétne prepravu predovšetkým tekutého tovaru. Poskytuje sledovací systém[7] pre rôzne typy prepravy. Ich riešenie poskytuje sledovanie prepravovaného tovaru a je takisto dostupné na mobilných telefónoch, čo považujeme za výhodu. Na druhú stranu zase vidíme problém toho, že sa projekt nezameriava na koncového zákazníka a neposkytuje informáciu o tom, že by ich riešenie bolo podporené transparentnosťou blockchain technológie.

Zhodnotenie existujúcich riešení

Na záver analyzovania niekoľkých dostupných riešení na trhu sme prišli na to, že riešenia sa sústredia hlavne na iniciatívu zo strany výrobcu. Ich hlavnou prioritou býva zaobstaranie dopravy tovaru tam, kam výrobca požaduje a vylepšenie dostupných riešení väčšinou ležia v možnosti kontroly nad týmto procesom. Výhodou riešení vnímame viditeľnosť celého prepravného procesu, no je však otázkou, akú veľký má výrobca vplyv na túto prepravu a ako veľa informácií sa mu poskytuje.
Veľkou nevýhodou vidíme práve to, že sa riešenia zameriavajú len na výrobcu. Očakávajú jednoduchý záujem o zlepšenie prepravného procesu (ušetrením nákladov a zvýšením prehľadu o aktivitách), no neberú vôbec ohľad na koncového zákazníka. Výhody pre koncového zákazníka vnímame skôr ako vedľajší efekt toho, čo spoločnosti svojimi riešeniami pokrývajú.
Ďalšiu nevýhodu, ktorú sme si v riešeniach všimli, je nedostatok takej komunikácie, ktorá by sa dala označiť ako vysoko dôveryhodná. Zatiaľ čo systémy môžu byť zabezpečené, nemáme záruku, ktorú by nám poskytlo riešenie so zapojením blockchain technológie. Preto považujeme za nevyhnutné v našom projekte tieto aspekty a nedostatky zohľadniť v návrhu a implementácii.

---

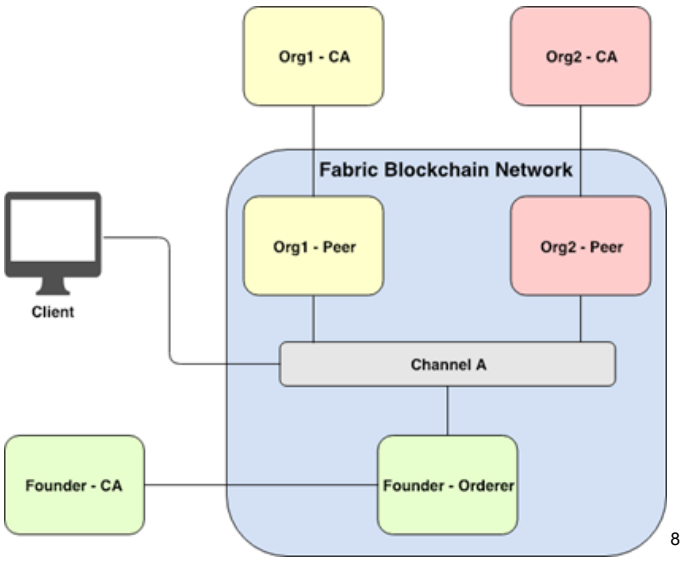
[5] Dostupné na adrese: https://www.scsolutionsinc.com [citované 1. Máj 2021]
[6] Dostupné na adrese: https://hillebrand.com/solutions/supply-chain/supply-chain-management [citované 1. Máj 2021]
[7] Sledovací systém dostupný na adrese: https://www.hillebrand.com/tools/track-a-shipment [citované 1. Máj 2021]

Blockchain

Samozrejmou súčasťou analýzy bolo získanie základného prehľadu o blockchain technológii Hyperledger Fabric. V tejto sekcii sa pozrieme na základnú terminológiu a vysvetlíme dôležité princípy blockchain, ktoré budú figurovať v našom riešení.

Architektúra Hyperledger Fabric

Na diagrame nižšie vidíme zobrazenú architektúru.



[8]

Architektúra obsahuje nasledovné časti:

- Peer – uzol v blockchaine, ktorý ukladá všetky transakcie na channeli. Peery sú vlastnené a spravované organizáciami.
- Channel – súkromná vrstva komunikácie, „podsieť“, medzi viacerými organizáciami, ktoré sú naň pripojené umožňujúca im vytvárať vlastný ledger pre ich transakcie
- Orderer – služba zodpovedná za správu transakcií, vytváranie nových blokov pre dané transakcie a distribuovanie nových blokov na všetky Peery na channeli.
- CA (Certifikačná autorita) – je zodpovedná za správu používateľských certifikátov. Používateľ je schopný dopytovať sa alebo vyvolať akúkoľvek transakciu v ľubovoľnom channeli na základe povolení, rolí a atribútov, ktoré vlastní.
- Client – aplikácia

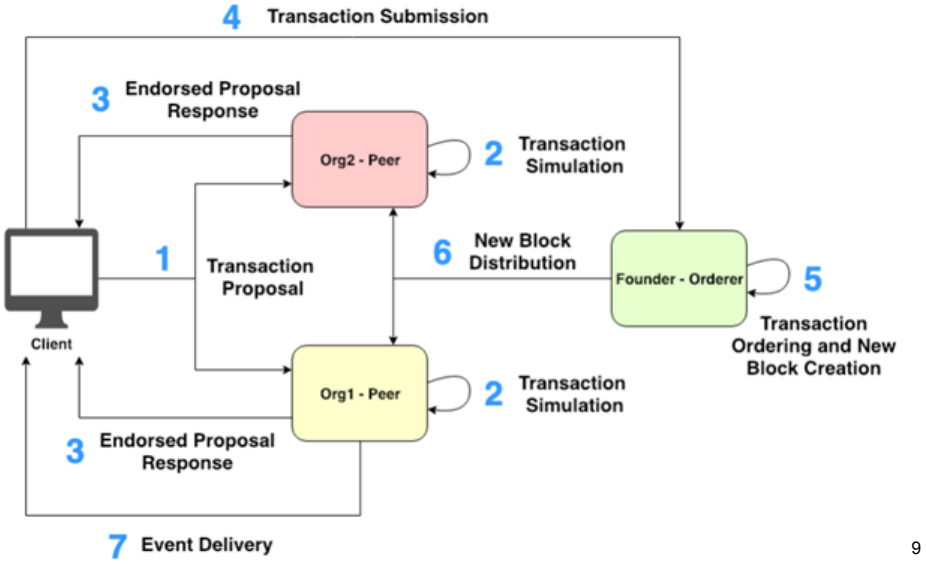
Chaincode

Chaincode je definovaný ako:

- štruktúra smart kontraktov
- neobsahuje iba logiku za vykonávaním smart kontraktov, ale aj validáciu a identifikáciu používateľov, ktorí chcú vykonávať smart kontrakty

[8] Prevzaté zo zdroja https://medium.com/coinmonks/demystifying-hyperledger-fabric-1-3-fabric-architecture-a2fdb587f6cb

Konsenzus



[9]

Konsenzus v Hyperledger Fabric by sa skladá z týchto šiestich krokov[10]:

1. Client vytvára žiadosť o transakciu, ktorú podpíše svojim certifikátom a pošle ju schvaľovacím Peerom na channeli.

2. Každý z týchto Peerov overí identitu a authorizáciu používateľa a ak má správne oprávnenia, Peer simuluje transakciu, generuje odpoveď a podpíše ju svojim certifikátom ale ledger zatiaľ neukladá žiadne zmeny vyvolané transakciou.

3. Klient skontroluje odpovede od všetkých schvaľovacích peerov. Pokiaľ aplikácia odoslala query transakciu a očakáva teda iba query response tak nesubmituje transakciu na Orderer. Pokiaľ ide o transakciu, ktorá aktualizuje ledger, tak sa pred poslaním na Orderer kontroluje, či bola splnená endorsement policy.

4. Klient posiela transakciu spolu s podpísanými odpoveďami na Orderer.

5. Orderer prijme novú transakciu, vygeneruje pre ňu nový blok a podpisuje ho svojim certifikátom.

6. Orderer vysiela blok s transakciou na všetky peery (nielen schvaľovacie) daného channelu, kde peery porovnajú svoj world-state s transakciou a pokiaľ je verifikácia v poriadku, aktualizujú ho podľa nej.

[9] Zdroj:https://medium.com/coinmonks/demystifying-hyperledger-fabric-1-3-fabric-architecture-a2fdb587f6cb

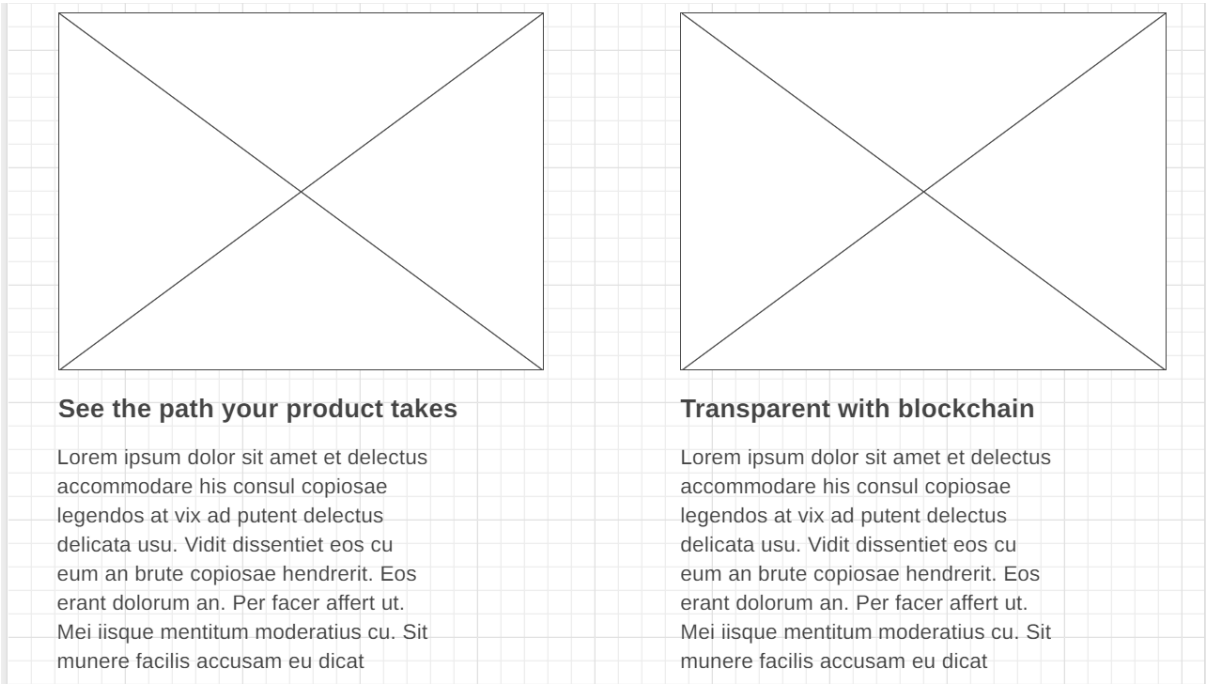
[10] Zdroj: https://hyperledger-fabric.readthedocs.io/en/release-2.2/txflow.html

Grafické návrhy

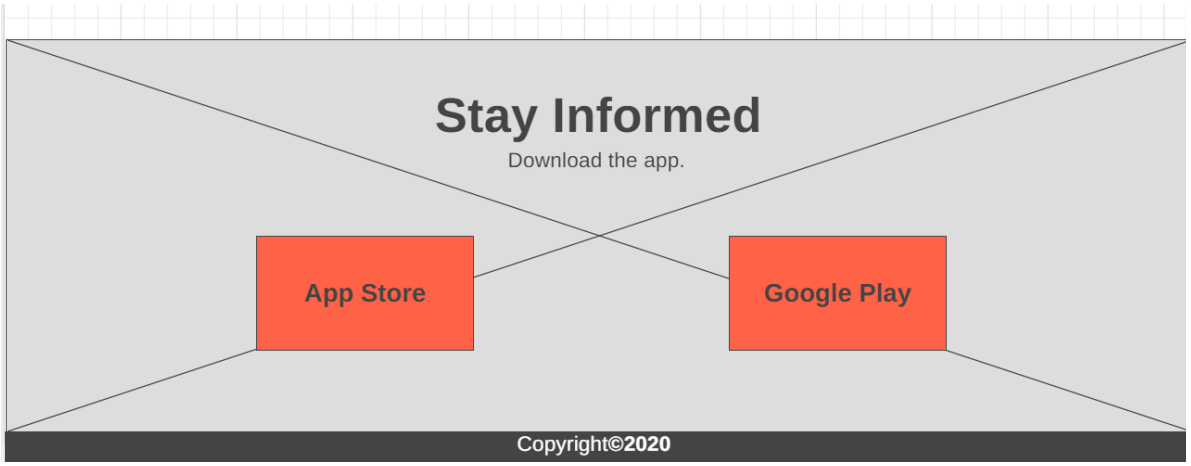
Wireframe webového rozhrania

V prvých fázach projektu začali vznikať potrebné grafické návrhy aplikácie ako podklad pre analýzu. Vznikol jednoduchý wireframe prototyp, ktorý mal reprezentovať úvodný prehľad o vzhľade a poskytnutej funkcionalite webu.
Na obrázku nižšie vidíme hornú časť obrazovky webovej aplikácie. Zistili sme, že budeme musieť prehľadným a jasným spôsobom informovať používateľa o možnosti skenovať kód výrobku, alebo mať možnosť zaregistrovať svoju spoločnosť a zúčastniť sa tak pri vytváraní blockchain riešenia.

Ďalšou nutnou súčasťou webovej stránky je informovanie používateľa o jej cieľoch a fungovaní. Bežný používateľ nevie, čo vlastne cieľom nášho projektu a je a čo je jeho pridaná hodnota. Preto by mali byť tieto informácie dostatočne viditeľné v úvodných častiach webového rozhrania. Tieto informácie vidíme na nasledujúcom obrázku.



V neposlednom rade je nutné poukázať na existenciu mobilnej aplikácie, ktorú by možno používateľ preferoval oproti webovému prehliadaču. Preto je na úvodnej stránke viditeľný odkaz na stiahnutie spomenutej aplikácie, ktorého dizajn vidíme na nasledujúcom obrázku.



Interaktívne prototypy

Za účelom bádania možností ako pre používateľa, tak pre možného výrobcu spoločnosti sme vypracovali interaktívny prototyp pomocou nástroja Figma. Prototyp slúži na overenie potrebnej funkcionality, ktorú by náš produkt mal ponúkať. Vedeli sme si týmto spôsobom overiť, či má používateľ dostupnú všetku nevyhnutnú funkcionalitu. Takisto sme aj overili, čo bude potrebné pre výrobcu používajúceho náš produkt. Prototyp webového rozhrania je dostupný pomocou tohto odkazu.

Za účelom preskúmania možností ponúkaných pre používateľa bol takisto vypracovaný prototyp rozhrania pre mobilné zariadenie. Tento prototyp overil, akým spôsobom prebehne interakcia bežného používateľa s našim produktom. Takisto sa overilo aj to, čo by mohlo byť súčasťou produktu a zlepšilo by používateľský zážitok. Tento prototyp je dostupný pomocou tohto odkazu.

Návrh

V návrhu zohľadníme analýzu a načrtneme prvky, ktoré sa v riešení budú vyskytovať. Začneme tým, že popíšeme jednotlivé objekty v riešení a budeme pokračovať tým, že definujeme interakciu týchto objektov pomocou diagramov prípadov použitia.

Identifikované non-user entity

Spoločnosť

Vzhľadom na to, že naše riešenie pokrýva výrobcov a dopravcov, potrebujeme definovať jednotný objekt spoločnosti. Tento objekt reprezentuje spoločnosť podľa názvu a prislúchajú mu jednotliví používatelia (používatelia sú opísaní nižšie v časti Identifikovaní používatelia). Spoločnosť sa v našom riešení delí na dva typy, ktoré sa označujú pojmami “MANUFACTURER_COMPANY” (spoločnosť výrobcu) a “CARRIER_COMPANY” (spoločnosť dopravcu).

Výrobok

Konkrétna “živá” inštancia produktu (spomenutý nižšie). Ide o entitu, ktorá sa v reálnom svete vyrobila, zaevidovala sa do nášho systému a následne sa prepravila do cieľovej destinácie. Cieľovou destináciu vždy rozumieme akýkoľvek typ fyzického predajného miesta (maloobchod, veľkoobchod, stánok, …).
Podlieha zmenám počas výroby a prepravy, kedy jednotliví používatelia zaznamenávajú zmeny buď v stave alebo lokalite výrobku. Pre predstavu si môžeme uviesť príklady:

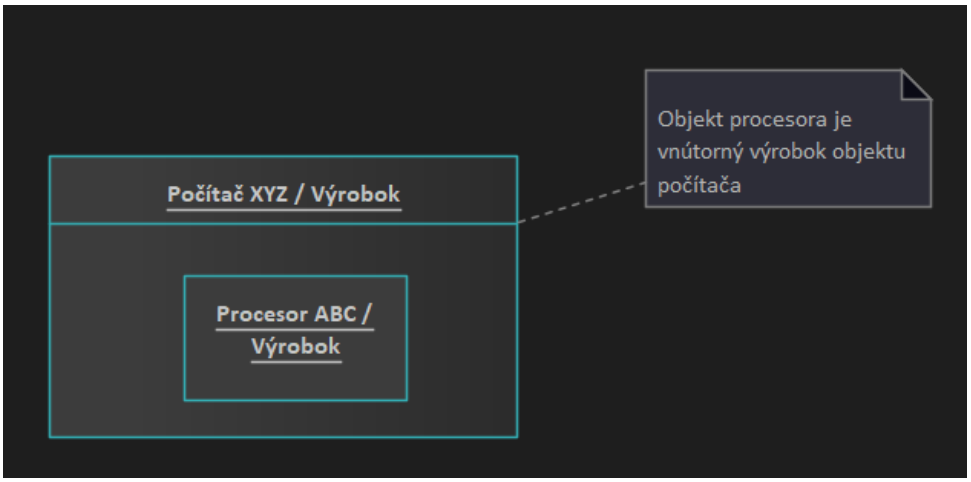
- Výrobok prešiel cez hranicu, dorazil do cieľovej stanice (dôležitá zmena jeho polohy)
- Výrobok ukončil cestu a použije sa ako súčasť iného výrobku (napríklad procesor sa zabuduje do počítača, bližšie popísané nižšie)
- Výrobok sa počas prepravy stratil, poškodil alebo mu bolo inak znemožnené pokračovať v ceste (dôležitá informácia o zmene stavu)

V aplikácii sa označuje pojmom “Item”.

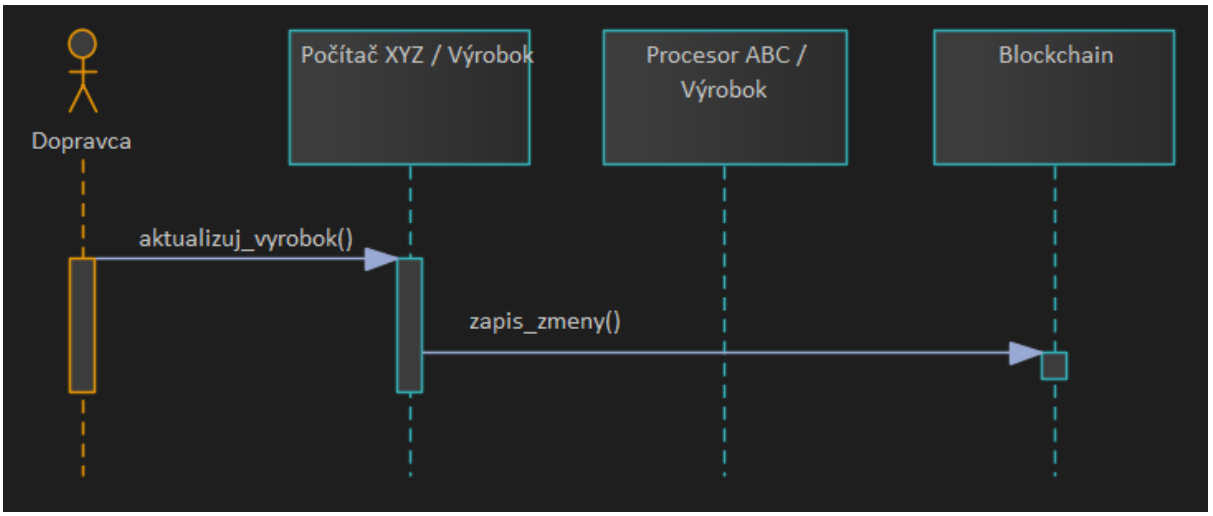
Výrobok z výrobkov

Výrobok z výrobkov (VzV) označuje prípad, kedy sa výrobok používa ako medziprodukt. Pre zrejmosť budeme označovať túto hierarchickú kompozíciu pomocou výrazov vonkajší a vnútorný výrobok. Pre ozrejmenie pojmov je treba dať do pozornosti, že sa táto väzba môže kľudne nazývať “Produkt z produktov” a nezmení sa tým logický význam, ktorý sa v tejto podčasti opisuje. Táto terminológia sa teda môže používať ľubovoľne, v tomto dokumente je zvolený pojem VzV.
Ukážku pre jednoduchú štruktúru VzV môžeme vidieť na obrázku nižšie. VzV mení spôsob toho, ako sa správa aktualizácia výrobku v rámci aplikácie.

Počas života výrobku sa evidujú jeho zmeny. Tieto zmeny sú priamo viazané na ID výrobku. Ak sa výrobok použije ako komponent pre iný výrobok, stáva sa z neho **vnútorný výrobok**. Znamená to teda, že je súčasťou iného výrobku. Aktualizácia tohto výrobku už nie je možná priamo, ale práve pomocou jeho **vonkajšieho výrobku**. Postup vyzerá potom tak, že sa vnútorný výrobok *automaticky* aktualizuje vždy, keď sa aktualizuje jeho zodpovedajúci vonkajší výrobok.
Následne si ukážeme zjednodušený príklad aktualizácie výrobku, ktorý sa skladá z jedného vnútorného výrobku (príklad počítača a procesora). Najprv si túto štruktúru ukážeme vizuálne.



Teraz si odhalíme, čo sa udeje pri aktualizácii vyššie uvedeného počítača.



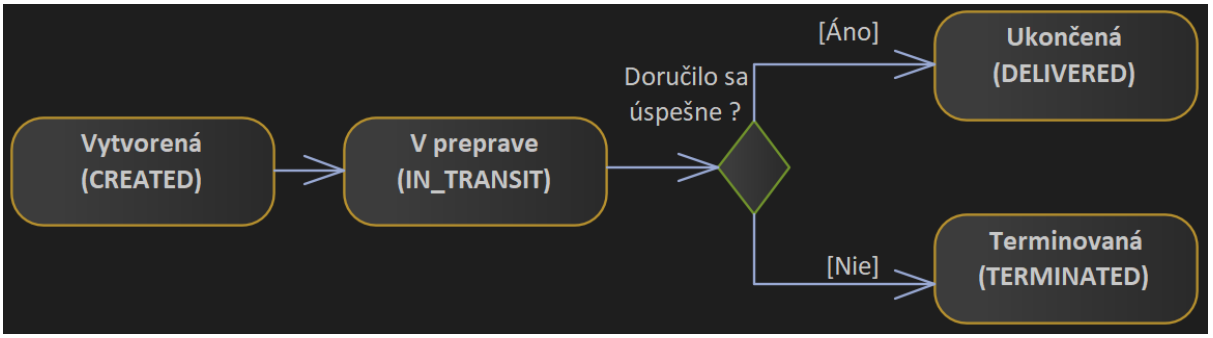
Musíme si tu všimnúť dôležitý detail - procesor sa ako samotný výrobok neaktualizuje. Aktualizuje sa len jeho vonkajší výrobok - počítač. V prípade, ak by počítač neskôr použil iný výrobca ako vnútorný produkt (povedzme, že napríklad predáva superpočítač zložený z iných počítačov a ďalších extra súčiastok), tak by sa tento počítač tiež prestal aktualizovať a vždy by sa aktualizoval len jeho vonkajší výrobok.

Produkt

Produkt slúži v našom systéme ako abstraktná položka, ktorá zoskupuje viaceré špecifikácie výrobku. Dôležitý je predovšetkým názov produktu, vedľajšie sú napríklad jeho parametre (cena, typ, a pod.).
Predpoklad pri vytváraní produktu je, že ešte neexistuje žiaden výrobok, ktorý by prislúchal danému produktu. Po vytvorení produktu je možné vytvárať jeho reálne výrobky. Pre bežného používateľa slúži produkt ako prehľad informácii o výrobku a výrobok má každý unikátne vlastnosti ako napríklad polohu alebo stav. Pomocou produktu je takisto možné sledovať základné štatistiky ako je počet výrobkov alebo priamo pristupovať cez produkty ku výrobkom.
V aplikácii sa označuje pojmom “Product”.

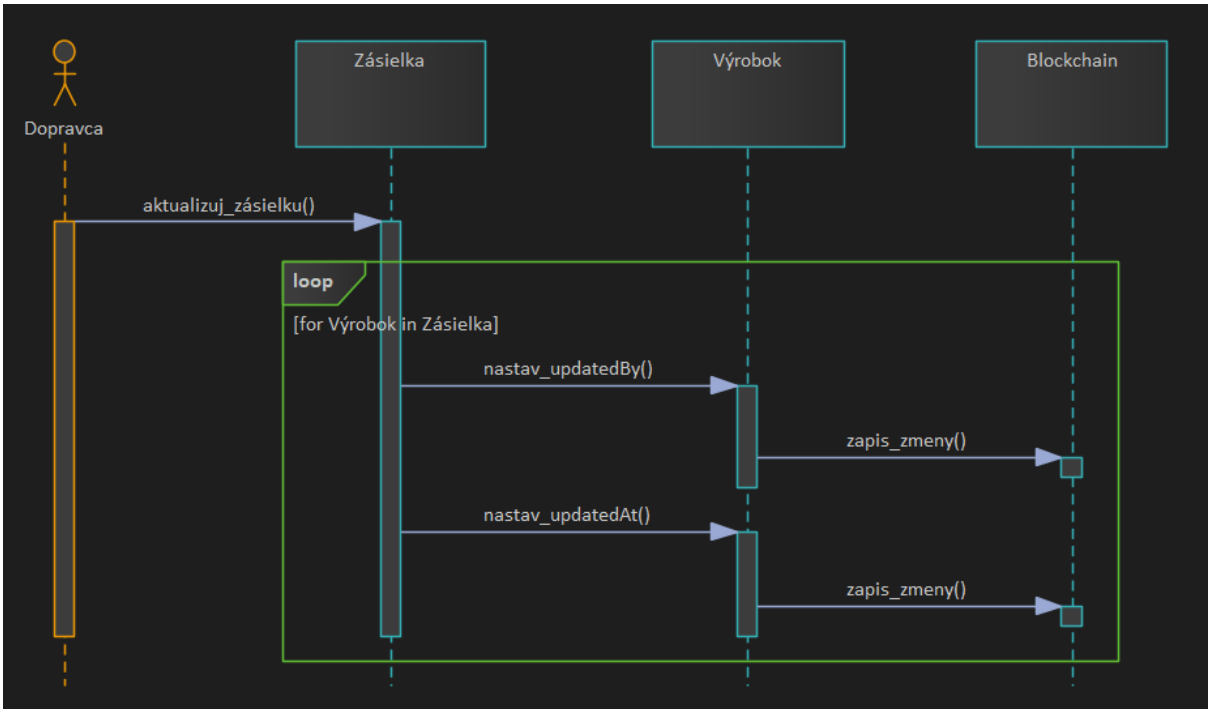
Zásielka a Cargo

Pre efektivitu sledovania a manipulácie s entitami definovanými vyššie sme vytvorili združovací mechanizmus, ktorý sme nazvali *zásielka* a *cargo*. Dané položky slúžia najmä dopravcom, ale sú tiež určené v prípade potreby pre výrobcu (opísané v časti Identifikovaný používatelia nižšie).
Zásielka má za úlohu zoskupovať výrobky do jedného celku. V analógii s reálnym svetom môže ísť napríklad o jednu paletu výrobkov, na ktorej sa nachádza 50 kusov výrobku A a 10 kusov výrobku B. Zásielka je určená výrobcom (na prípravu tovaru na prepravu) alebo dopravcom.
Cargo na druhú stranu slúži na zoskupovanie zásielok. Preto môže byť obsah carga reprezentovaný zásielkami 1241, 6214 a 6322[11]. Cargo je určené **výhradne pre dopravcu**.
V aplikácii sa označuje zásielka pojmom “Shipment” a cargo pojmom “Cargo”. Na to, aby sme sa vedeli odkazovať na akúkoľvek z týchto dvoch prepráv referencovať, používame v implementácii označenie “Package”. Na diagrame nižšie môžeme vidieť prechod stavmi, v ktorých sa typ “Package” (a teda zásielka alebo cargo) môže nachádzať.



Aby sme si bližšie ukázali, ako sa vykonávajú aktualizácie, pozrime sa na nasledujúci diagram. V ňom môžeme vidieť, ako sa pri aktualizovaní zásielky aktualizujú a zapíšu zmeny výrobkov do blockchain databázy.

[11] Dané čísla reprezentujú ID zásielok.



Identifikovaní používatelia

Výrobca

Reprezentant svojej firmy alebo organizácie, ktorý do procesu vstupuje ako záujemca o sledovanie svojho tovaru pomocou blockchain technológie. Má možnosť zaregistrovať sa v našom systéme. Následne po prihlásení vytvára produkty a pre tieto produkty eviduje vyrobené výrobky.
Ďalej má možnosť vytvárať aj zásielky, ktoré doručí na prepravu dopravcovi alebo si ich prepraví sám.
V aplikácii sa označuje pojmom “MANUFACTURER”.

Dopravca

Dopravca má možnosť registrácie a po prihlásení môže pristupovať k zásielkam či cargám a zobrazovať alebo aktualizovať ich stav. Predpoklad je, že si naskenuje alebo zadá identifikátor zásielky alebo carga. Takisto má možnosť vytvárať zásielky alebo cargá na to, aby mohol s prepravovaným tovarom jednoduchšie manipulovať. Jednoduchosť je poskytnutá tým, že manipulovať s cargom alebo so zásielkou môže analogicky ako s jedným samostatným výrobkom. Zobrazenie alebo aktualizovanie zásielky sa odrazí na všetkých výrobkoch, ktoré sa v nej prepravujú. Podobne sa pri aktualizácii carga zmení stav všetkých zásielok v danom cargu.
V aplikácii sa označuje pojmom “CARRIER”.

Zákazník

Predstavuje koncového zákazníka, ktorý si pomocou mobilného zariadenia alebo webového rozhrania naskenuje QR kódy výrobkov a vidí, akými zmenami si výrobky prešli. Z jeho strany je obohatený o nové informácie, ktoré sú podložené overením pomocou blockchain

technológie. Môže sa takisto registrovať a prihlásiť do našej aplikácie aby si ukladal históriu skenovaných výrobkov.
V aplikácii sa označuje ako “CLIENT”.

Administrátor

Administrátor nie je v momentálnom stave riešenia významná rola, ale predpokladáme jej využitie v budúcnosti. Pomocou administrátorskej role je možné manipulovať s používateľskými entitami aplikácie, a konkrétne teda vytvárať používateľov a spoločnosti pre systém. Rola administrátora sa delí na tri časti:

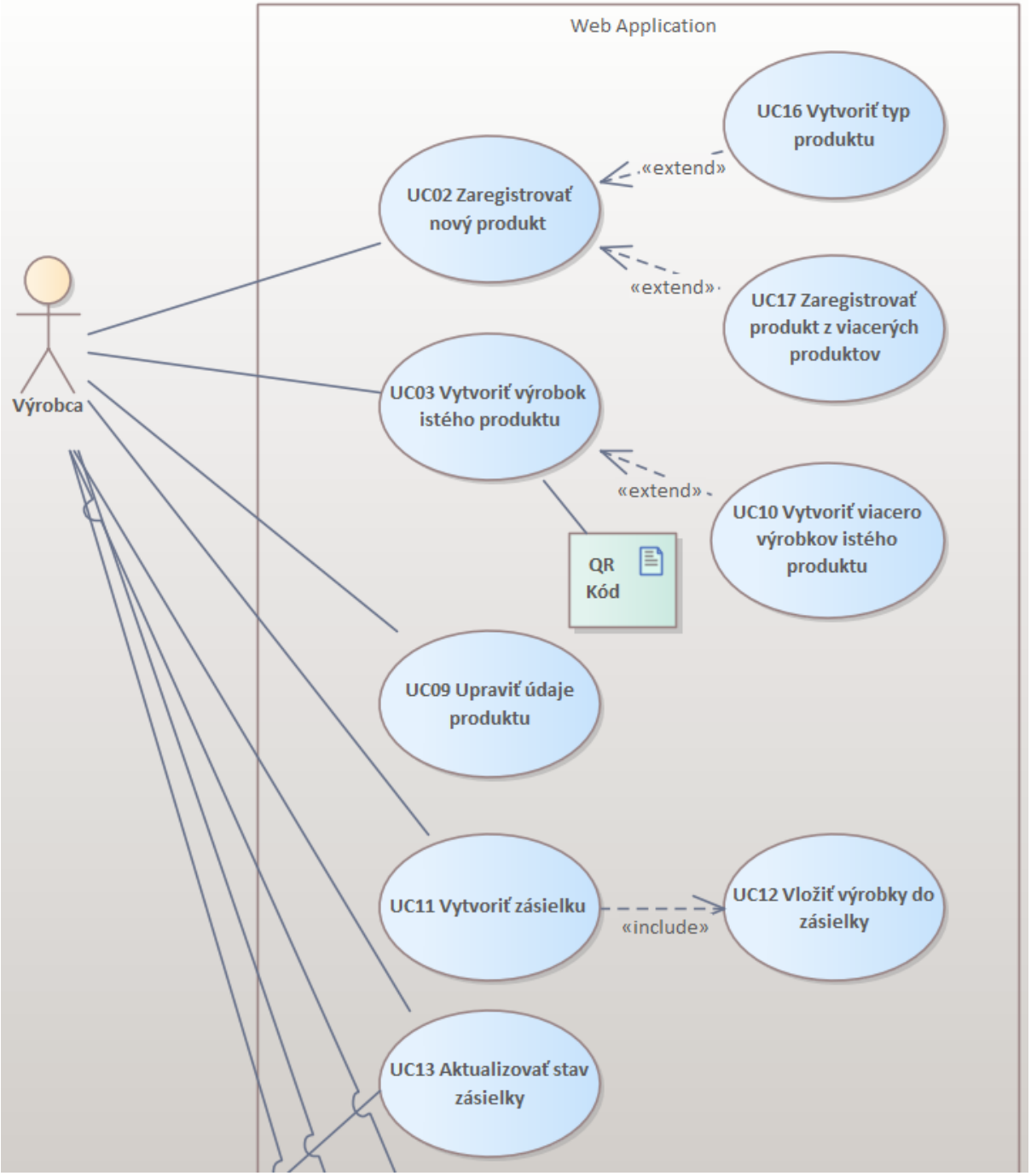
- Administrátor (všeobecne): Považuje sa za celkového správcu systému, ktorý môže registrovať spoločnosti. Proces registrácie sa predpokladá byť formálne dohodnutý mimo aplikáciu tak, že administrátor potom jednoducho vytvorí konto spoločnosti. Na toto konto sa potom pripojí daný správca spoločnosti, ktorý vytvorí účty iným členom v rámci tejto spoločnosti. Označuje sa pojmom “ADMIN” v aplikácii.
- Výrobca-administrátor: Predstavuje entitu výrobcu obohatenú o určité právomoci administrátora. Momentálne si môže výrobca vytvárať iných používateľov svojej spoločnosti podľa potreby. Táto rola má okrem toho rovnaké právomoci a prístup ako rola bežného výrobcu. Označuje sa pojmom “MANUFACTURER_ADMIN” v aplikácii.
- Dopravca-administrátor: Predstavuje entitu dopravcu doplnenú o vytváranie nových používateľov v rámci svojej spoločnosti (podobne ako v prípade výrobcu-administrátora). Momentálne si môže výrobca vytvárať iných používateľov svojej spoločnosti podľa potreby. Táto rola má okrem toho rovnaké právomoci a prístup ako rola bežného výrobcu. Označuje sa pojmom “CARRIER_ADMIN” v aplikácii.

Platformy a použitie aplikácie

Realizovaný systém je rozdelený na mobilnú aplikáciu a webové rozhranie. Predpokladá sa, že používanie mobilného telefónu bude preferované skôr koncovým zákazníkom, zatiaľ čo webové rozhranie bude preferované výrobcami či dopravcami.

Predpokladané využitie webového rozhrania

Počas analýzy problémovej oblasti sme prišli na návrh interakcie používateľa so systémom. Tento návhr môžeme vidieť na diagrame prípadov použitie na obrázku nižšie[12]. Prípady použitia vychádzajú z predpokladu, že jednotliví používatelia systému používajú rozličné typy entít ako výrobok, zásielka alebo cargo.



[12] Pre prehľadnosť je diagram rozdelený na dve časti.

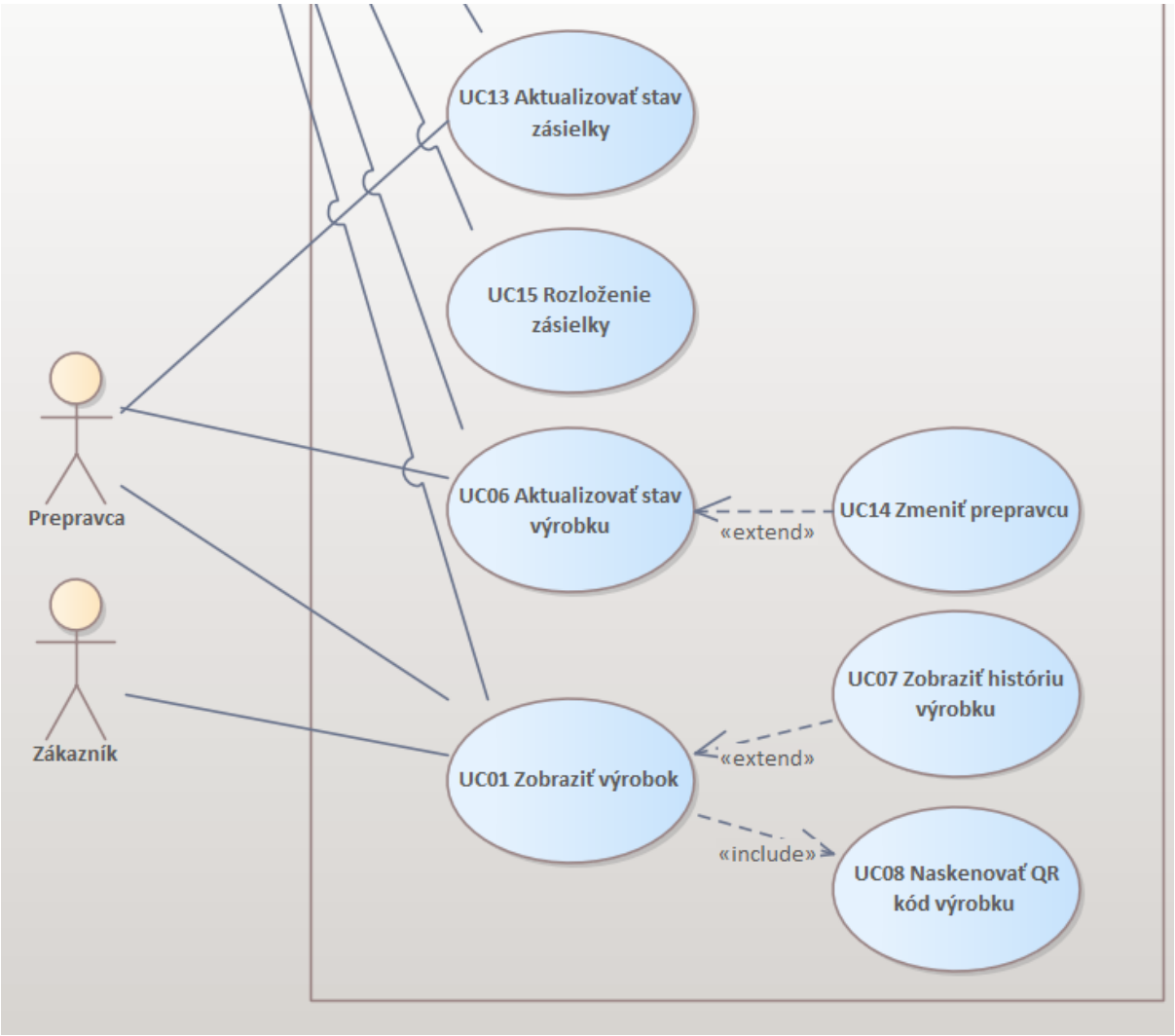
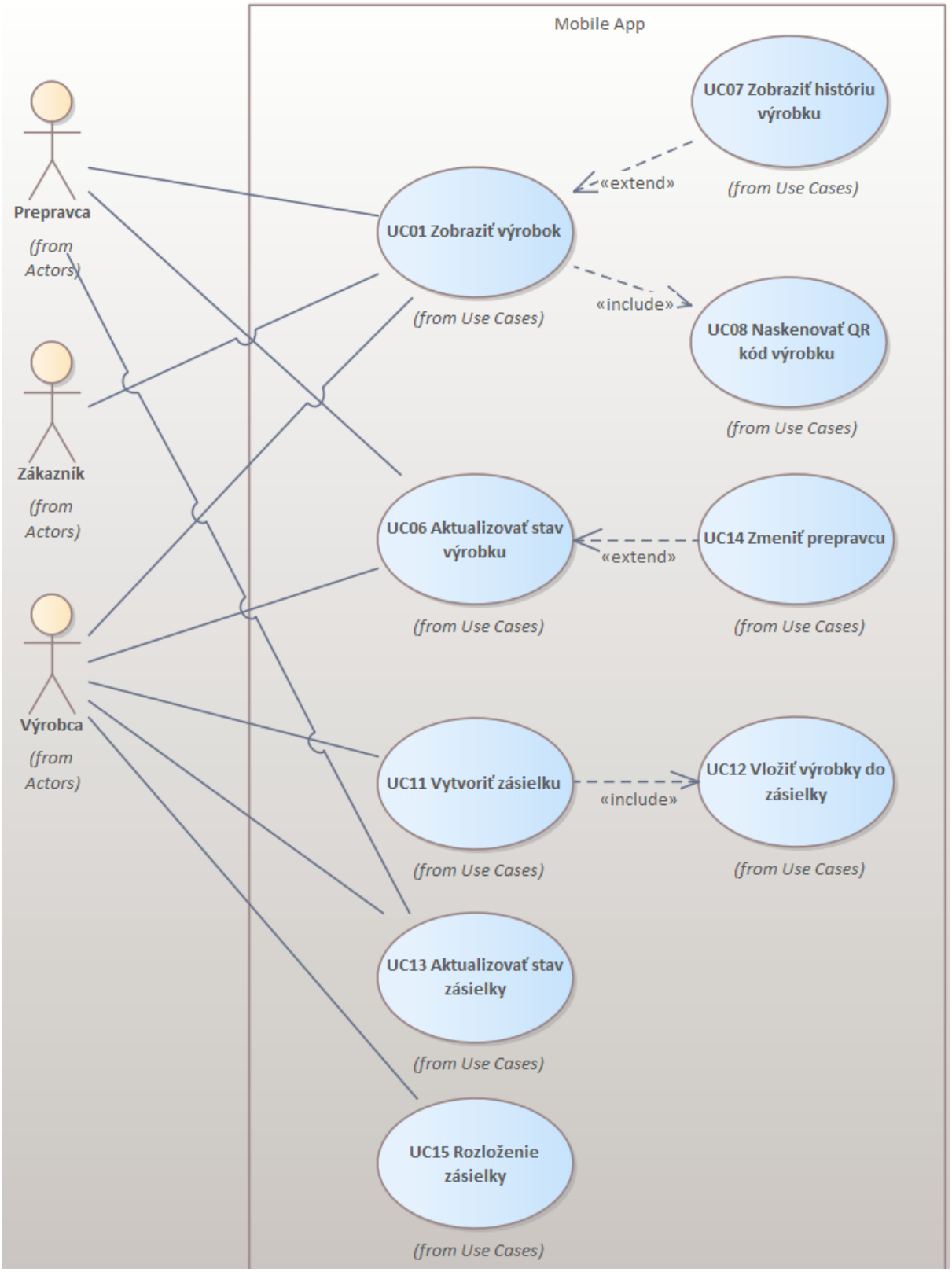

UC13 Aktualizovať stav zásielky
UC15 Rozloženie zásielky
UC06 Aktualizovať stav výrobku
UC14 Zmeniť prepravcu
«extend»
UC07 Zobraziť históriu výrobku
UC01 Zobraziť výrobok
«extend»
«include»
UC08 Naskenovať QR kód výrobku
Prepravca
Zákazník

Diagram prípadov použitia pre mobilnú aplikáciu

Na obrázku nižšie vidíme diagram prípadov použitia analogický tomu v sekcii Predpokladané využitie webového rozhrania. Tento diagram však berie do úvahy možnosti, ktoré predpokladáme, že budú poskytnuté rozhraním mobilného telefónu.



Implementácia

Projekt je rozdelený na niekoľko sub-projektov, ktoré riešia už vyššie spomenuté zamerania systému. Jednotlivé subprojekty sú:

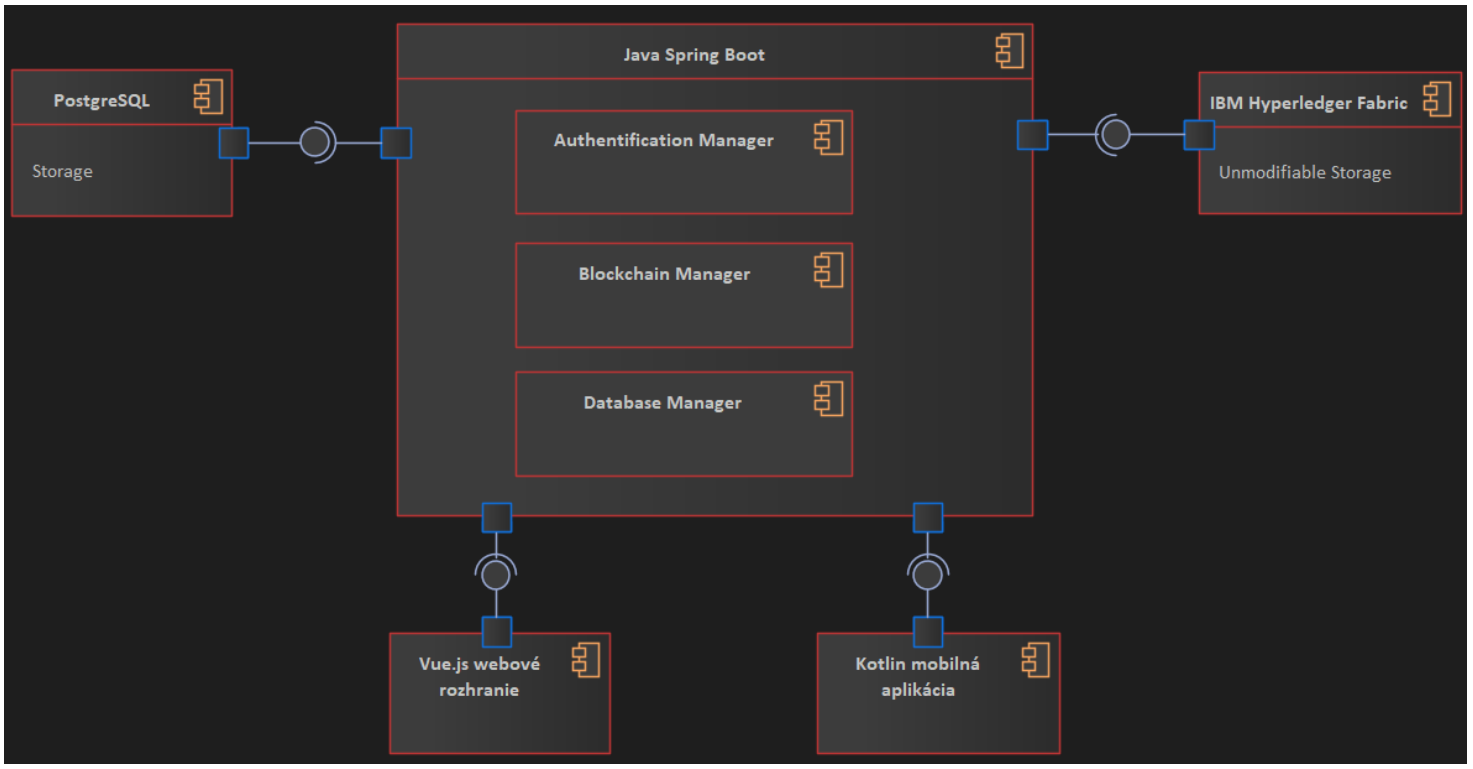
- s-chain-server: Spring-boot server
- s-chain-mobile: Kotlin mobilná aplikácia
- s-chain-web: Webové rozhranie Vue.js
- s-chain-blockchain: IBM Hyperledger blockchain
- s-chain-page: záloha webovej stránky nášho tímu

Pre účely testovania vznikli aj nasledovné účty, ktorých význam je vysvetlený v priebehu tejto kapitoly:

- **Admin**: admin@admin.com
- **Manufacturer**: manufacturer@company.com
- **Manufacturer admin**: manufacturer-admin@company.com
- **Carrier**: carrier@company.com
- **Carrier admin**: carrier-admin@company.com
- **Client**: john.doe@example.com

Heslo pre uvedené účty je **heslo123**

Komponenty identifikované v našej aplikácii sme zachytili v diagrame komponentov na obrázku nižšie. Vidíme, že sa nám podarilo identifikovať základné komponenty, ktoré rozširujú jednoduchý diagram z kapitoly Prehľad o systéme. Komunikácia medzi komponentami je zabezpečená pomocou REST API, čo však lepšie uvidíme na diagrame rozloženia.

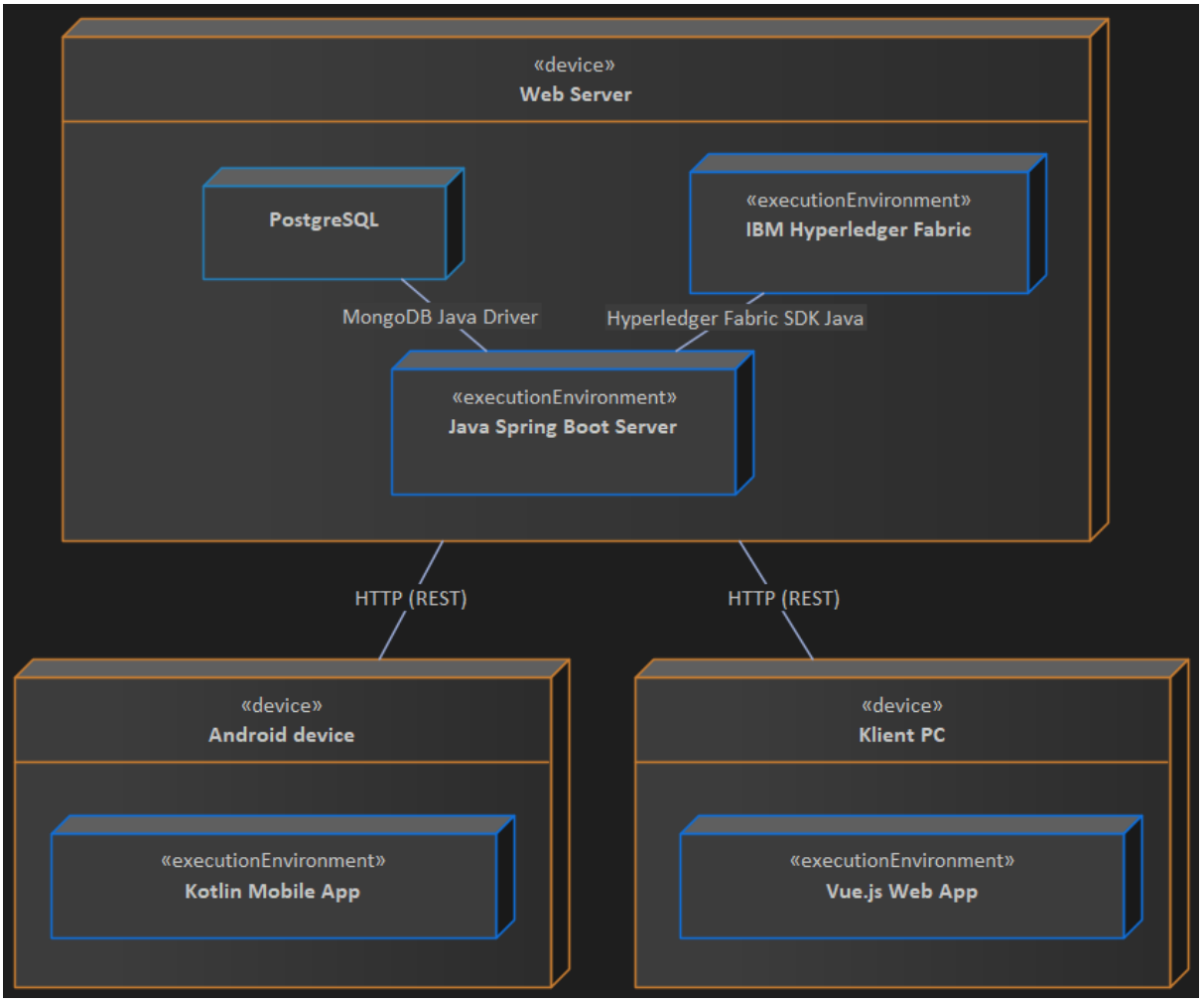


Dôležitá poznámka

Pri ukončovaní projektu sme **na konci semestra** zapracovali pripomienky Oltis Group a doplnili niektoré nové funkcionality do systému. Ale práve vďaka ukončovaniu semestra nebolo možné viac čakať na dokončenie týchto úloh a následné ich zdokumentovanie. Preto sa najaktuálnejšia verzia projektu môže v istej miere líšiť od implementácie v tejto kapitole (hlavne v rámci diagramov). Logická štruktúra riešenia však zostáva zachovaná, zmenili sa len implementačné detaily.

Diagram rozloženia systému

Na obrázku nižšie vidíme, ako sa nachádzajú jednotlivé časti systému na reálnych strojoch. Momentálne máme nasadený server, blockchain a databázu na jednom spoločnom serveri. Mobilná aplikácia a webové rozhranie existuje ako standalone verzia a pripája sa externe na tento server.



Android zariadenie reprezentuje používateľa, ktorý si pomocou mobilného telefónu skenuje a zobrazuje históriu výrobkov.

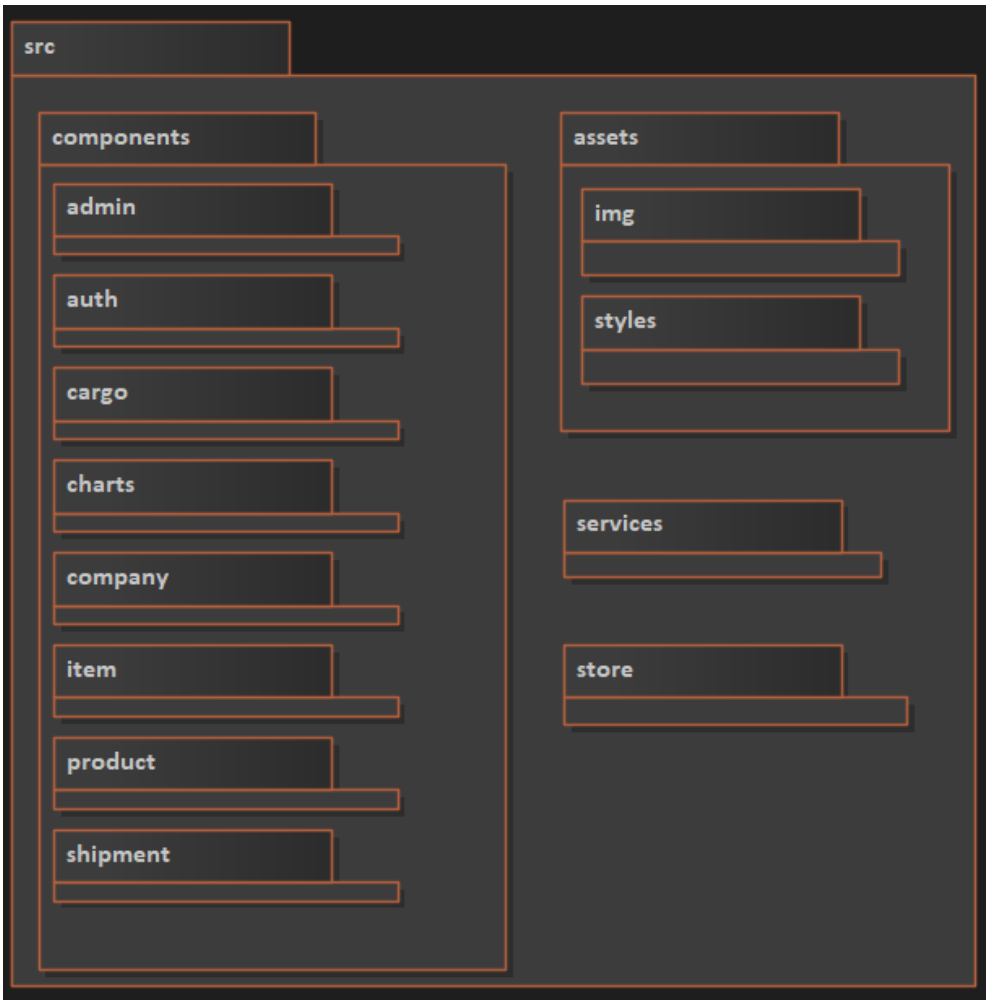
Štruktúra jednotlivých projektov

V tejto časti si zobrazíme štruktúry jednotlivých projektov, ktoré boli definované na začiatku kapitoly “Implementácia”. Štruktúru si zobrazíme pomocou diagramov balíkov, pričom pri nie úplne jasnom účele balíka bude poskytnutý krátky opis jeho cieľov.

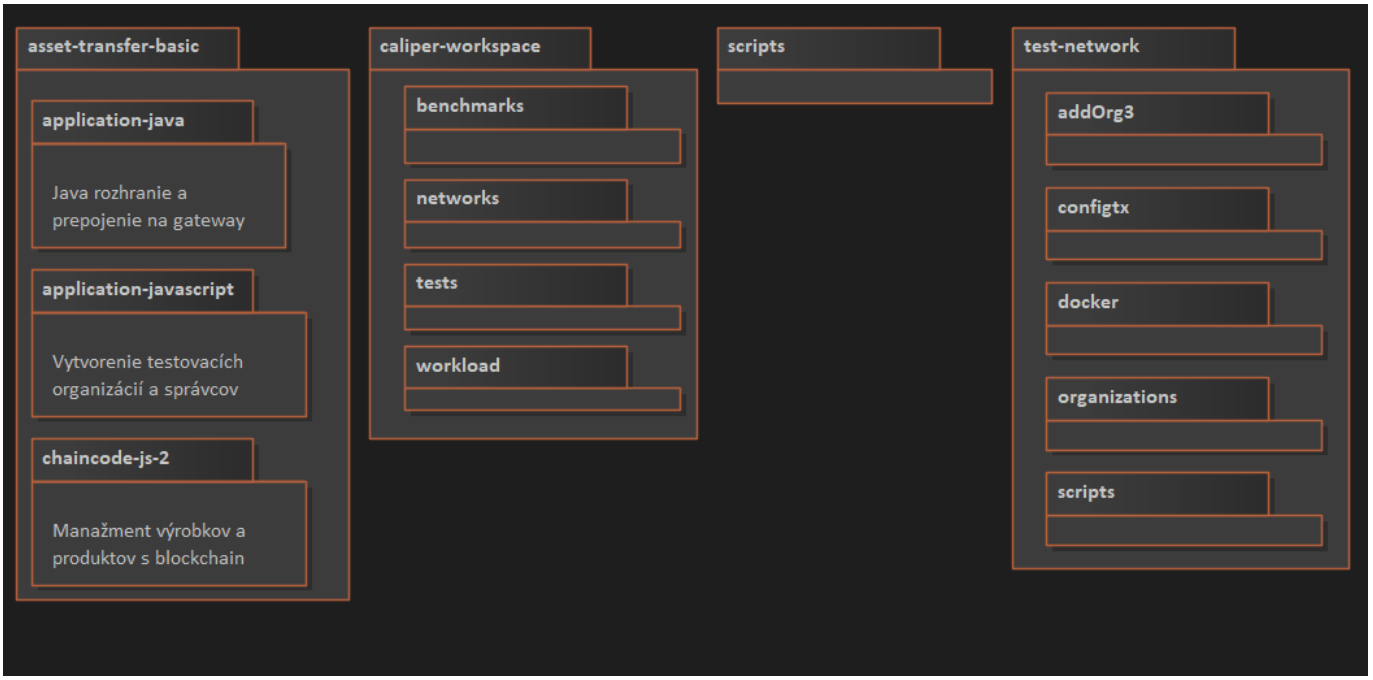
Server


core
account
admin
Podklad pre registráciu spoločností a používateľov.
blockchain
carrier
cargo
company
client
dev
Developer API (endpointy s prefixom "_dev"
item
manufacturer
pack
Manžment balíkov (package)
procut_item
Kontajner pre produkt, výrobok a spoločnosť, ktorej patrí.
product
product_type
scanned_item
shipment
user
company_user
api
swagger
auth
config
filter
token
blockchain
filter
jpa
qrcode
db
utils
value

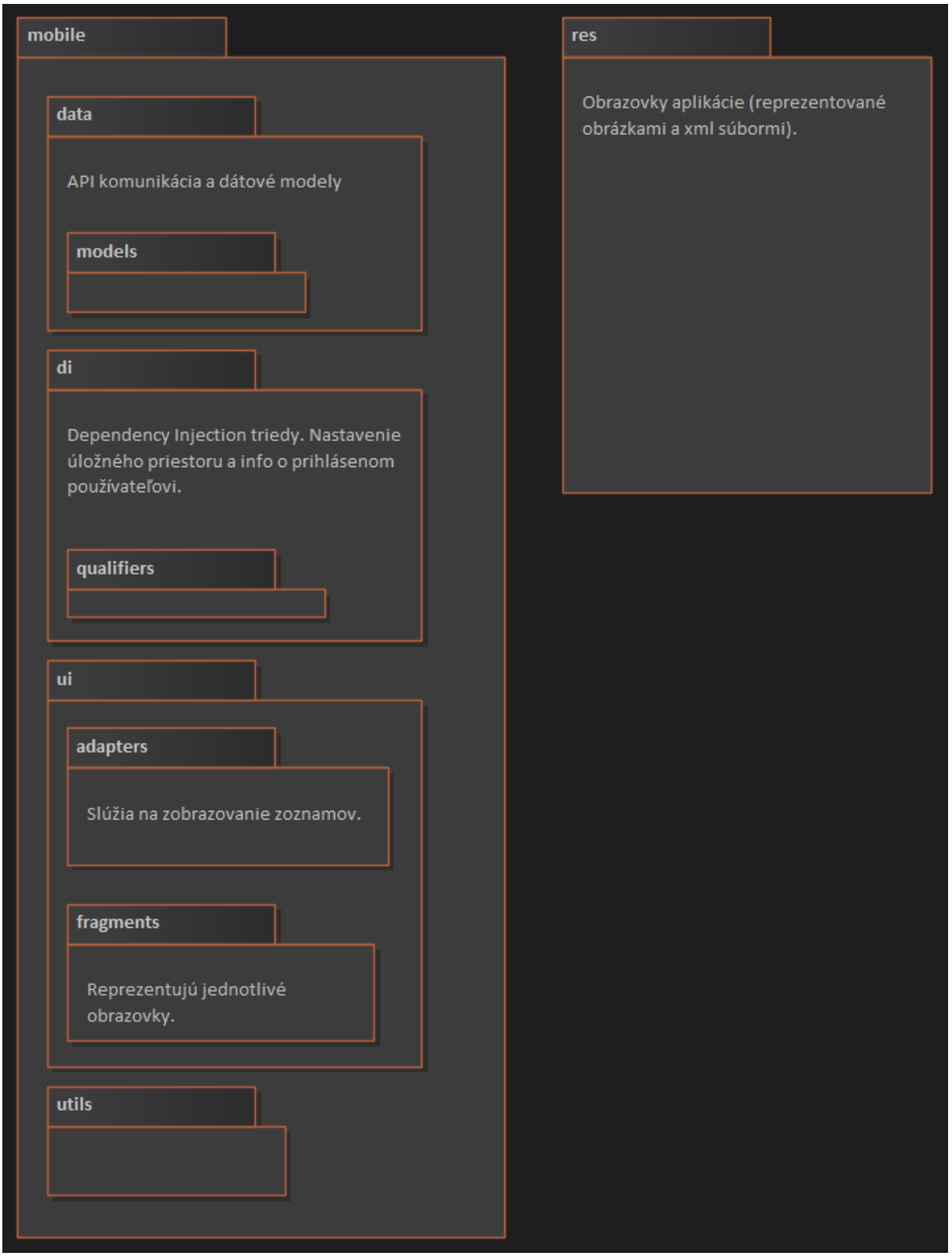
Webové rozhranie



Blockchain



Mobilná aplikácia

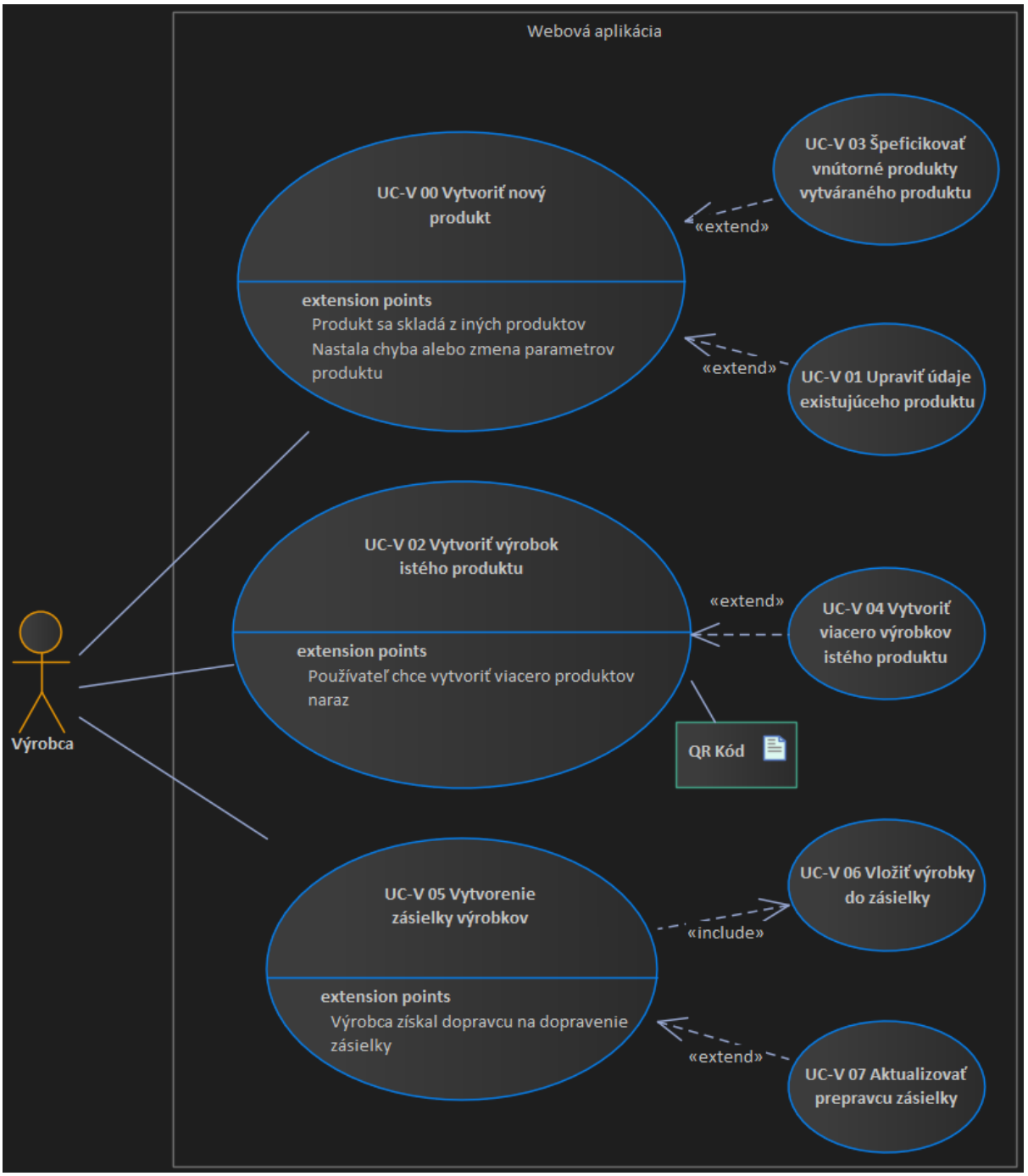


Interakcie so systémov (prípady použitia)

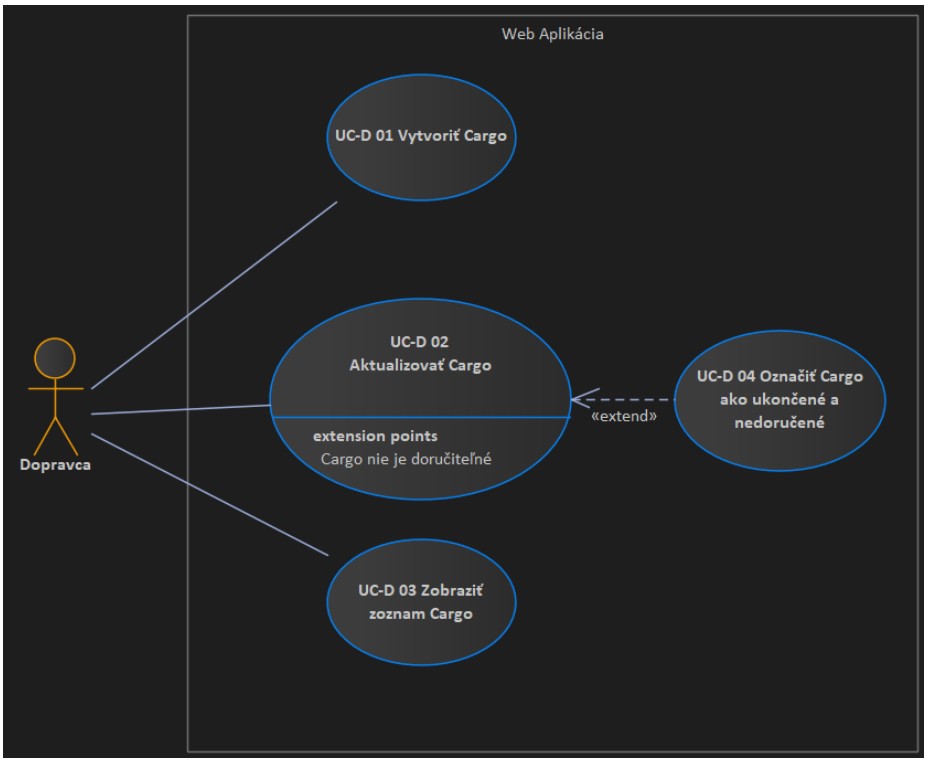
Pre interakciu sme zvolili reprezentáciu pomocou diagramov prípadov použitia. V tejto časti sa budeme venovať teda poskytovanej funkcionalite. Pre každú rolu zobrazíme prípady

použitia separátne kvôli prehľadnosti a neskôr zobrazíme zdieľané prípady použitia pre všetky roly v jednom diagrame. Opäť pripomíname možnosť interakcie s diagramami manuálne pre zabezpečenie lepšej prehľadnosti.

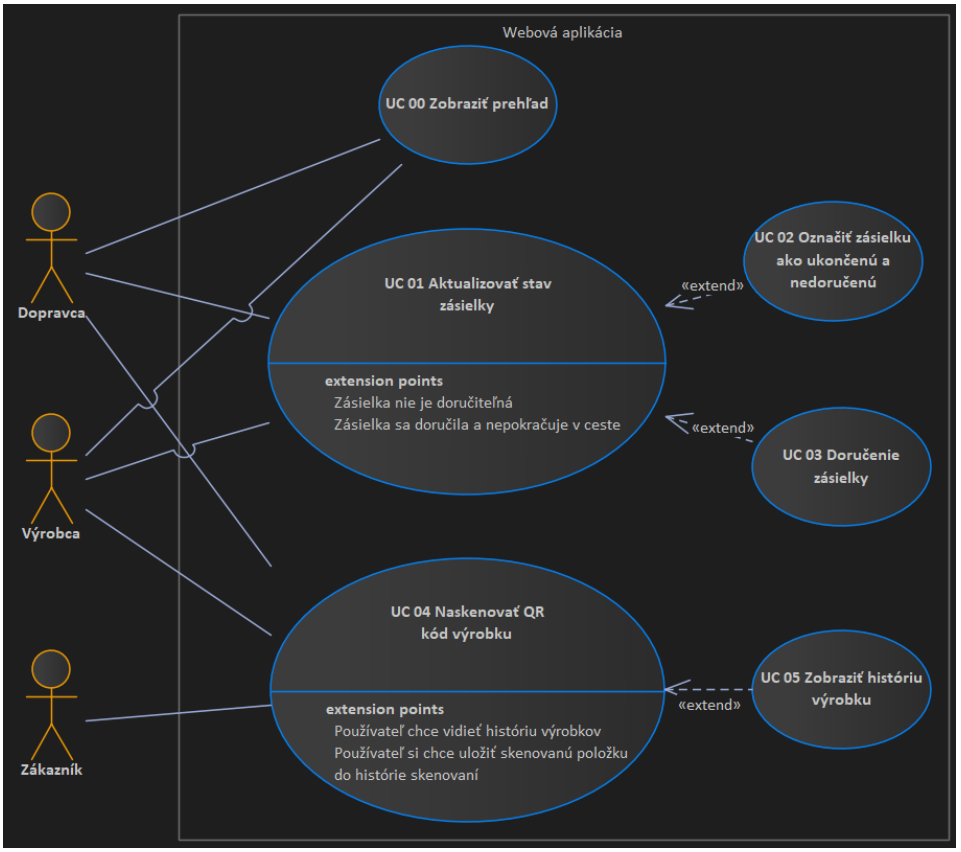
Výrobca - webové rozhranie



Dopravca - webové rozhranie

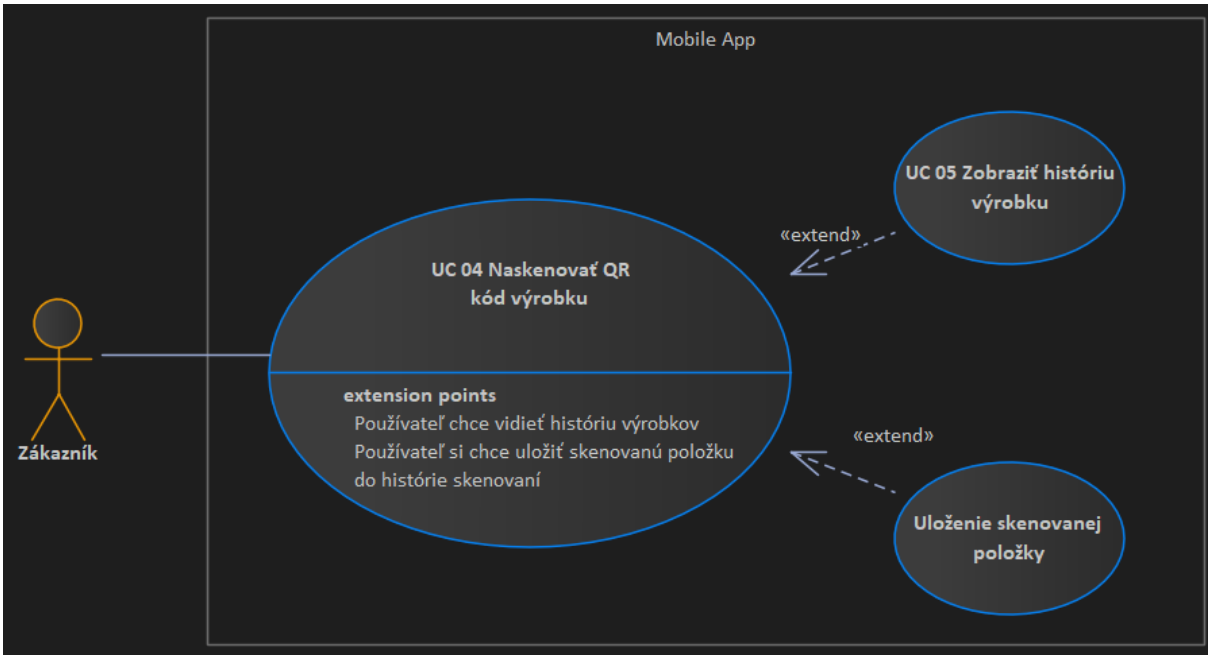


Webová aplikácia - zdieľané prípady použitia



Mobilná aplikácia

S mobilnou aplikáciou je možné interagovať len z pohľadu zákazníka. V prípade, že je zákazník registrovaný, je možné ukladať aj históriu skenovaných výrobkov.

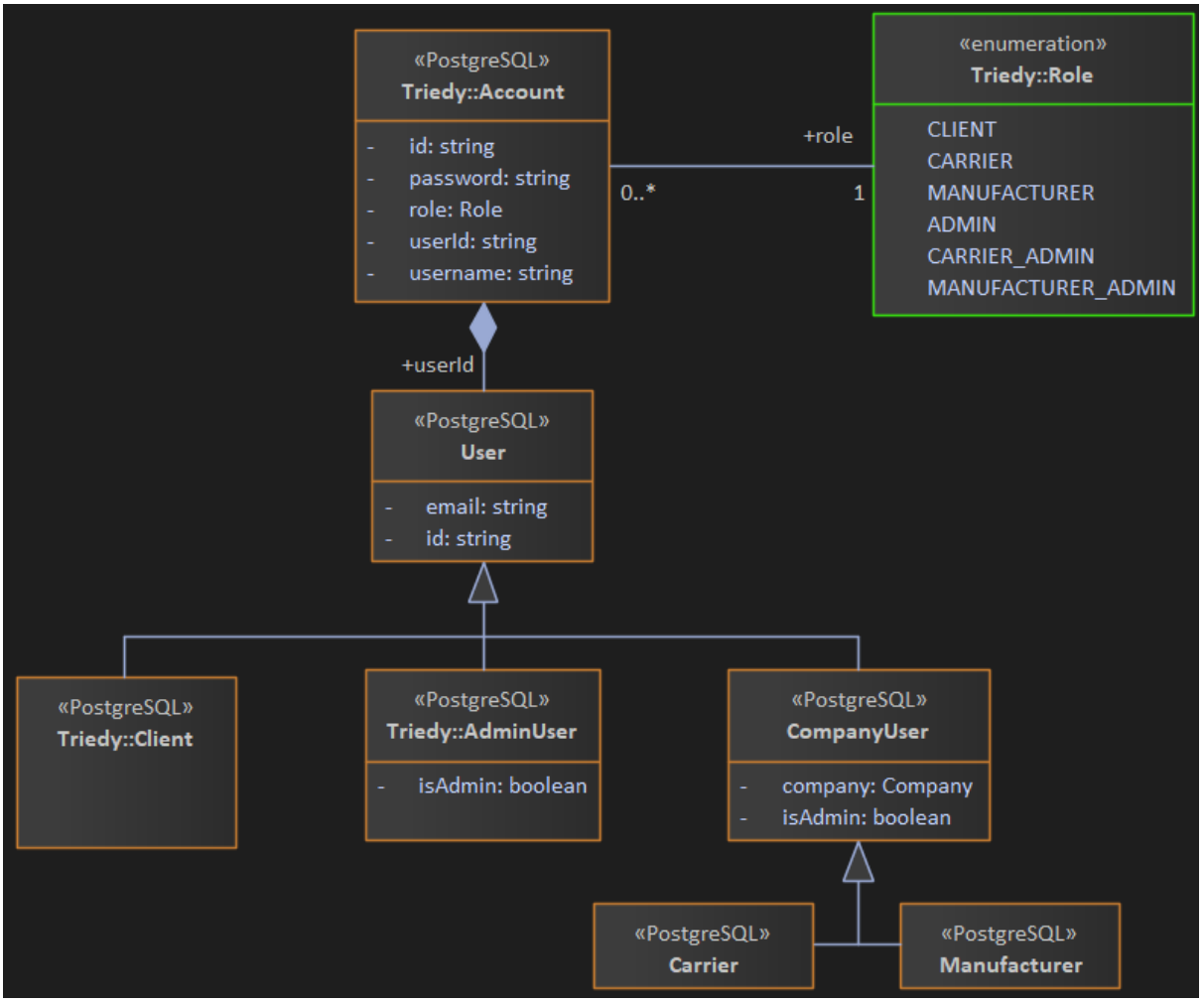


Dátové modely

V tejto časti dokumentu sú prezentované reprezentácie dátových modelov. Diagramy sú prezentované ako diagramy tried čím sa približujú k reálnej implementácií.

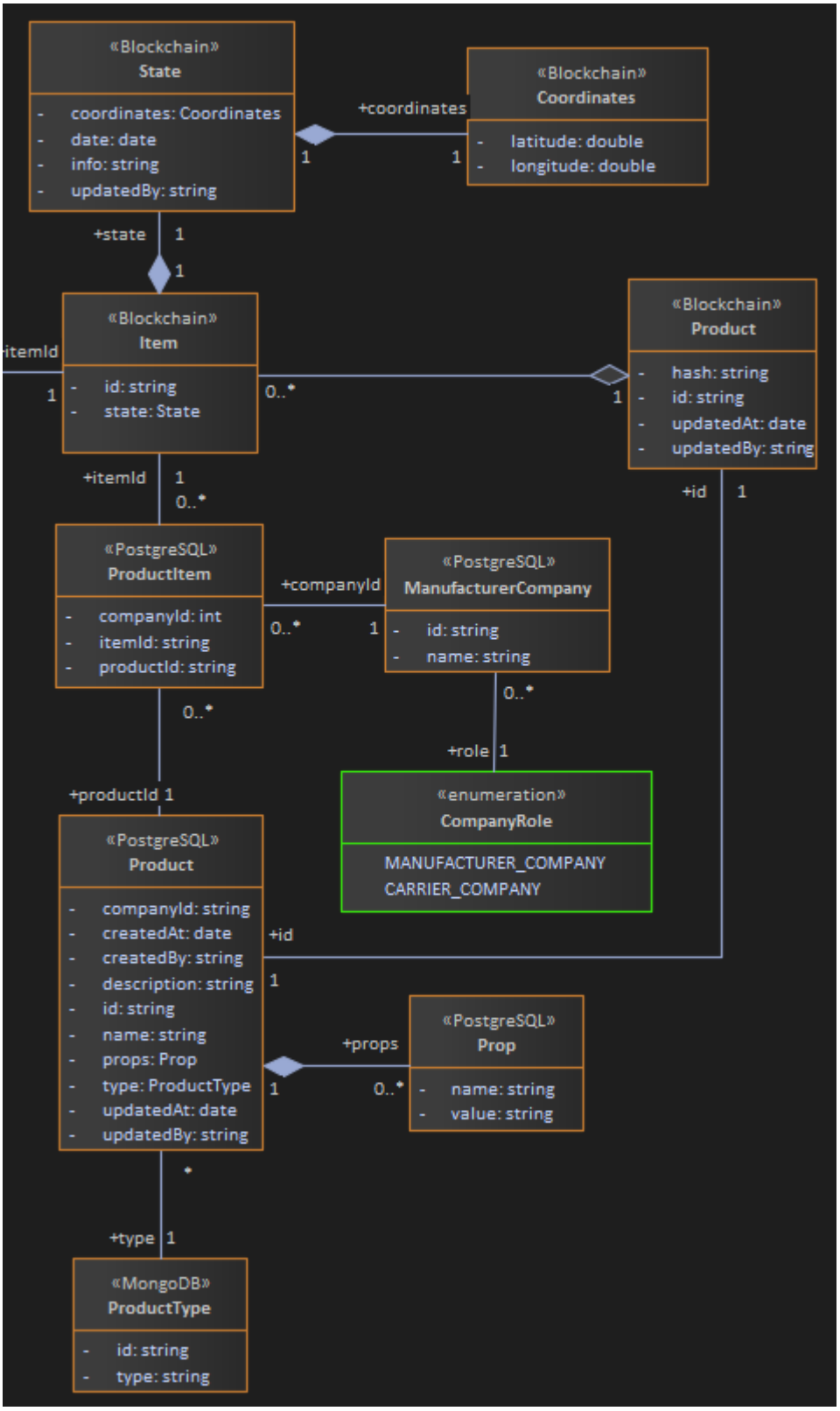
Dátový model účtov a rolí

Na obrázku nižšie vidíme dátový model zobrazujúci používateľa v systéme.

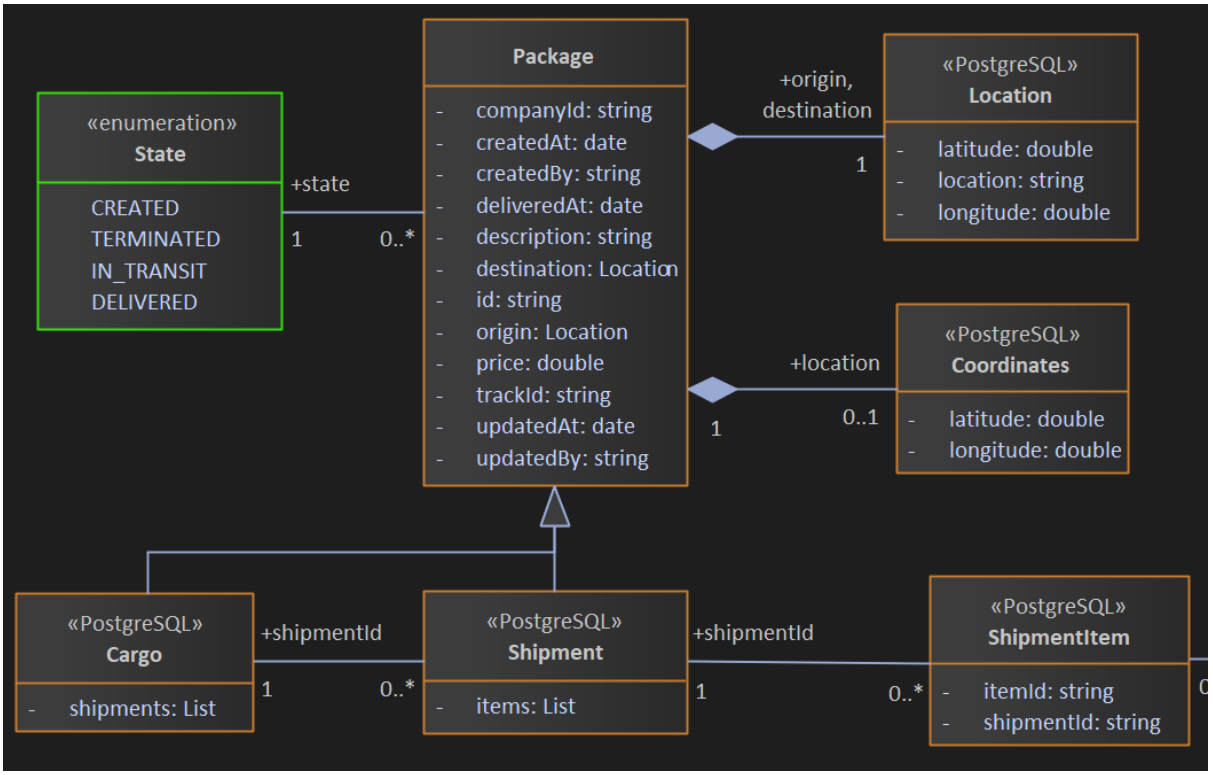


Dátový model produktu, výrobku a zásielky

Na nasledujúcom diagrame vidíme dátový model pre produkty a výrobky tak, ako boli opísané v časti “Návrh”. Pre prehľadnosť je diagram rozdelený na dve časti , predstavuje však jeden celistvý diagram. Vidíme, že výrobky (Item) obsahujú produkty, čo vlastne znamená, že zobrazením výrobku môžeme zobraziť aj informácie o produkte, ku ktorému tento výrobok patrí. Z diagramu takisto vyplýva prepojenie s databázou (postgreSQL alebo blockchain). Entity patriace do blockchain majú zabezpečuje dôvernosť údajov.
Naše rozhodnutie o neukladaní informácií o produkte do blockchain vyplýva z toho dôvodu, že informácie sa môžu meniť a samotný produkt sa realizuje pomocou výrobku, takže produkt nepodlieha žiadnym zmenám (aktualizácie pri výrobe či preprave). Preto nie je problém, ak by sa mali dáta produktu upravovať (chyby, zmeny alebo prispôsobenie opisu). Pre zabezpečenie dôveryhodnosti dát sme sa však rozhodli ukladať do blockchain databázy hash informácií produktu. Takto je možné v prípade neautorizovanej zmeny detekovať problém a zabrániť zmene.



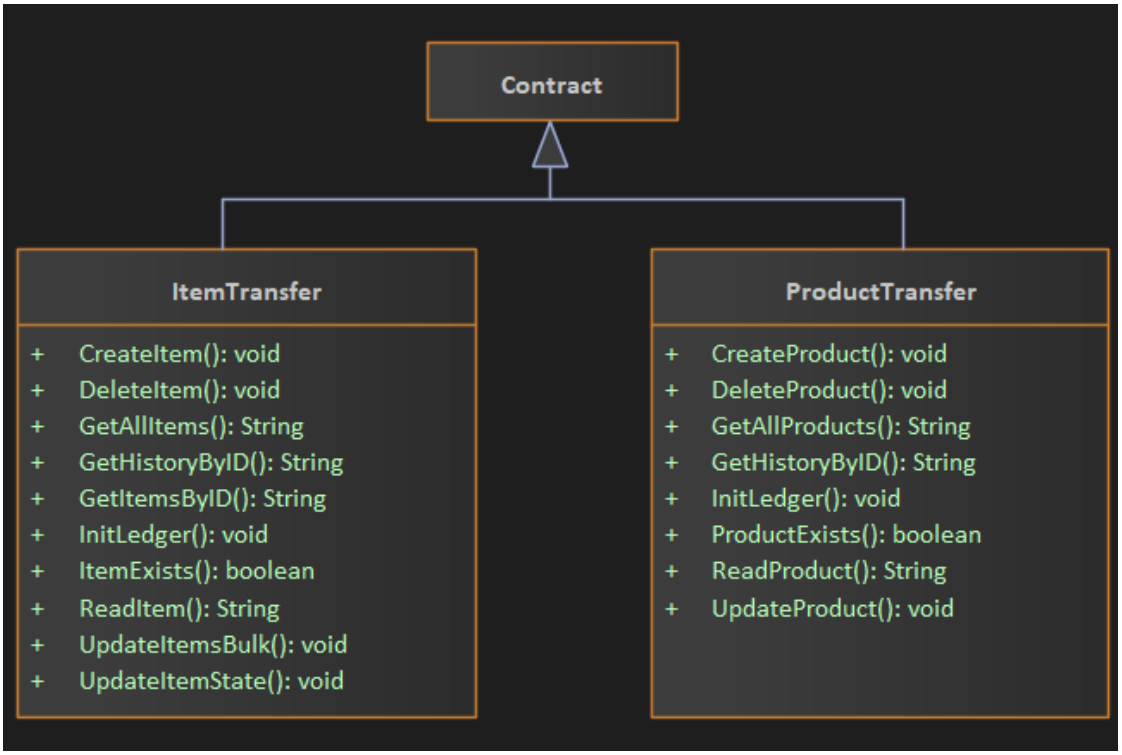
Na tomto diagrame vidíme štruktúru produktov, výrobkov a ich reprezentácie v databázach a vzhľadom ku spoločnosti.



Na diagrame vyššie vidíme zobrazenie zásielky a carga a reprezentáciu ich vlastností.

Blockchain základné funkcie

Pre manipuláciu servera s blockchain databázou existujú jednoduché ovládacie funkcie, ktoré sú zobrazené na diagrame nižšie.



Záver a ďalšie smerovanie

V rámci tohto projektu sme vytvorili základ projektu, ktorý sme nazvali S-chain a nasadili ho na doménu s-chain.tech. Náš projekt podporuje už vyššie spomínané entity a funkcionalitu. Bol overovaný dynamickým testovaním s partnermi z Oltis Group, ktorý nám poskytli RUSim rozhranie, ktoré volalo naše poskytnuté API tak, akoby bolo vyvolávané používateľmi z reálneho sveta. V tejto časti načrtneme ďalšie smerovanie projektu, predpokladané vylepšenia a nedostatky, ktoré nebolo možné dokončiť alebo opraviť.

Testovanie

Naše riešenie sme vyvíjali predtým, ako sa na ne navrhovali testy. V rámci zabezpečenia funkcionality a spoľahlivosti by bolo vhodné doplniť a rozšíriť testy pre každý z projektov.
Dôležitejšie je tiež testovanie s reálnymi používateľmi. Ide primárne o validáciu používateľského rozhrania webovej a mobilnej aplikácie so zameraním na jednoduchosť používania, zmysluplnosť názvov, intuitívnosť a podobne.

Konzultovanie biznis logiky praxe

Ako bolo v predošlej časti spomenuté, bolo by v rámci rozšírenia funkcionality, bolo by vhodné overiť a konzultovať prípady použitia s potenciálnymi používateľmi. Oltis Group nám poskytlo prehľad o tom, ako funguje primárne železničná doprava, lenže nemáme priame vstupy od samotných výrobcov a ani dopravcov. Bolo by vhodné overiť naše riešenie používateľským testovaním s potenciálnymi výrobcami a dopravcami. Poskytnuté prípady použitia v našej aplikácii sú zamerané *všeobecne* a ich využitie sa len predpokladá. Bolo by vhodné overiť, či náš spôsob riešenia vyhovuje aspoň základným potrebám výrobcov či dopravcov.
V rámci rozšírenia aplikácie je vhodné konzultovať biznis procesy hlavne zo strany výrobcov, ktorí pre nás počas nášho vývoja neboli dostupní. Je nutné zistiť, ktoré prípady použitia v našej aplikácii chýbajú a ktoré sú naopak nevhodne navrhnuté pre používanie výrobcom alebo dopravcom. Takisto je odporúčané, aby sa zo strany dopravcov zúčastnili na testovaní ideálne skupiny dopravcov rôznych typov prepravy (cestná, lodná, železničná). Tým pádom sa môžu objaviť možné problémy alebo napríklad diery v zabezpečení zo strany rozhrania dopravcu.

Prístupnosť pre “reálny svet”

Naša aplikácia zatiaľ prevažne používa na vývoj a testovanie lokálne prostredie. Naše riešenie je nasadené a aktualizované na doméne s-chain.tech, lenže všetky organizácie a používatelia sú vytvorení za účelom testovania. Rozhranie pre administrátorov na registrovanie nových používateľov a spoločností je dostupné, no na prispôsobenie sa reálnemu svetu je potrebné ho rozšíriť. Vhodné by bolo napríklad spravovať existujúcich používateľov alebo spoločnosti. Tento problém momentálne nemá až takú vysokú prioritu, ale v rámci úplnosti je dôležité ho tu spomenúť.

Obohatenie mobilného rozhrania

Mobilné rozhranie je v aktuálnej forme použiteľné len pre zákazníka alebo registrovaného zákazníka. Predpokladáme, že mobilné rozhranie radi využijú aj iné typy používateľov, preto by bolo vhodné implementovať mobilnú verziu rozhrania pre týchto používateľov.

Príloha A - Spätná väzba k produktu

Zápis z jednání 28.4.2021

Přítomní za Oltis: Jiří Čáp, Petr Šohajek, Petra Setlíková, Viktor Patras.

Přítomní za partnera:Tibor Vincze, Rastislav Bencel, Tomáš Babjak, Daniel Minárik, Adrián Libiak, Andrej Šulavík, Kamil Macek, Matej Petráš, Matej Holý

Po konzultaci s partnery byla pro komunikaci mezi partnery projektu (respektive aktéry dané simulace) zvolena technologie REST umožňující komunikaci pomocí http požadavků.

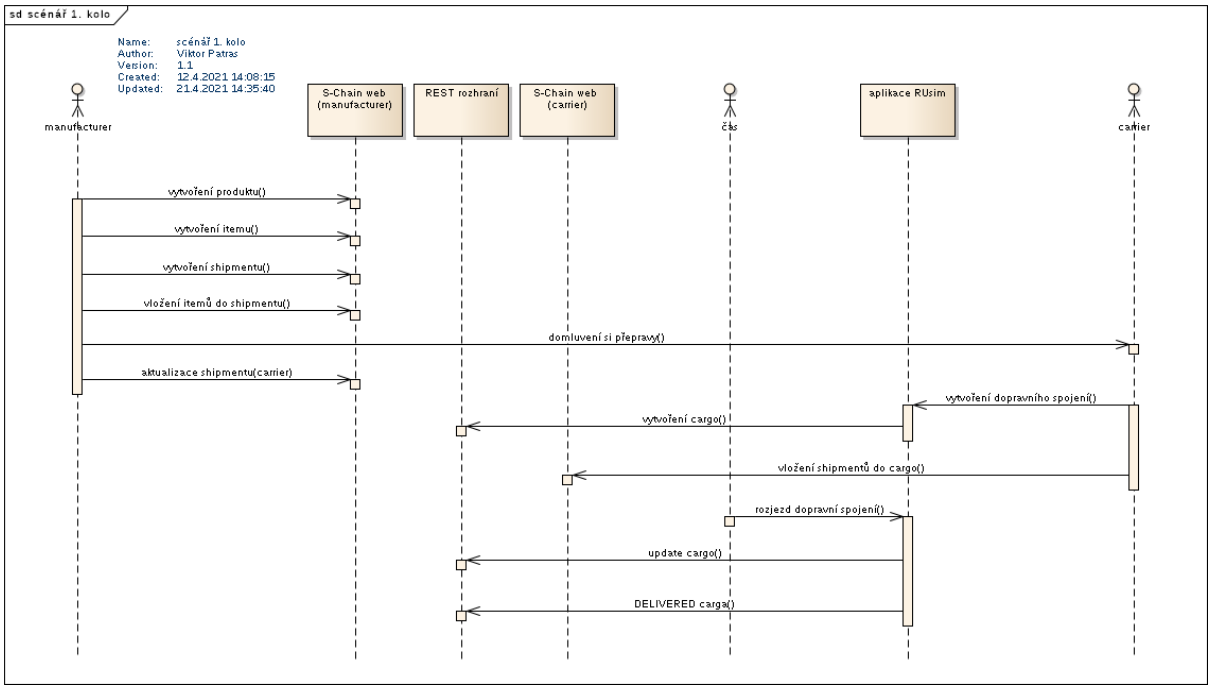
V rámci spolupráce na projektu bylo zpřístupněno testovací prostředí vznikající webové aplikace, dostupné na: https://s-chain.tech/item/1-bc46edd1-8768-4e7d-a3b7-d0ca3ae0055f.

V rámci testování byla dodána také základní dokumentace v open source frameworku Swagger, dostupná na: https://team09-20.studenti.fiit.stuba.sk/app/swagger-ui/?urls.primaryName=carrier#/cargo-api/addShipmentsUsingPOST.

Součástí dokumentace je rámcový dokument, popisující jednotlivé uživatelské role a procesní diagram znázorňující návrh postupu výrobku od výrobce po konečného zákazníka v rámci projektu (Příloha č.1), po dobu přípravných práci projektu dostupný dostupný online na: https://docs.google.com/document/d/1Et8lZJXF5WKL-OnQVILNNMM9gouF91VUJKV8_hcoXaM/edit?pli=1.

Návrh a popis možného scénáře

P5i testování došlo k úpravě sekvenčního diagram (obrázek č.1), který popisuje návrh komunikace mezi výrobcem a dopravcem:



Obrázek 1 - scénář 1.kolo

1) Aktér manufacturer (výrobce) na webu:
    a. založí produkt
    b. vytvoří item
    c. vytvoří shipment
    d. vloží itemy do shipmentu
    e. po domluvení přepravy s dopravcem: aktualizuje shipment - carriera a **trackingId** (dopravcovo číslo přepravy)
2) Aktér carrier (dopravce) datově vytvoří cargo.
3) Aktér carrier na webu vloží shipmenty do carga
4) Aktér čas rozjede dopravní spojení
5) Aktér carrier interně sleduje jízdu dopravního spojení
6) Aktér carrier datově:
    a. během jízdy dopravního spojení průběžně updatuje cargo / shipment o polohu
    b. po příjezdu do cílového bodu změní stav carga na DELIVERED

Pro úplnost jsou pojmy chápány takto:

- product = výrobní program
- item = konkrétní výrobek
- shipment = zásilka, trackingId je číslo konkrétní přepravy (co by to mohlo být?)
- cargo = dopravní prostředek vlak/loď/auto – zde možná úvaha o přejmenování vzhledem k implementaci pro zjednodušení lidského chápaní?

Návrh vystavení rozhraní za Oltis:

Za Oltis bylo na adrese: ***https://pardubice.oltisgroup.cz/rusim*** vystaveno několik zdrojů pro vznášení dotazů pomocí požadavků REST API:

POST /api/v1/connections

Založení nového dopravního spojení (pokud již neexistuje) a vrácení celého detailu. Důležitá je zde položka number, která odpovídá podacímu číslu.

PUT /api/v1/connections/{id}

Aktualizuje (změna stavu z CREATED na MOVING) stav daného dopravního spojení, vygeneruje události na časovou osu a vrátí celý detail.

PUT /timer

Spustí / vypne timer neboli simulační jádro.

GET /api/v1/connections

Vrátí seznam identifikačních čísel dopravních spojení.

GET /api/v1/connections/{id}

Vrátí detail daného dopravního spojení (podle příslušného ID).

GET /api/v1/connections/headers

Vrátí seznam hlaviček dopravních spojení uložených v databázi.

GET/api/v1/events

Vrátí seznam událostí na časové ose.

GET/actuator/health
Vrátí stav simulačního jádra.

GET/actuator/info
Vrátí informace o aplikaci.

GET /timer
Vrátí stav aplikace – hodnota true/false podle stavu timer.

Návrh úprav pro s-chain web
Na internetových stránkách, které se v rámci projektu neustále vyvíjí je cílem pohodlné a smysluplné užívání zákazníkem. Níže jsou uvedeny návrhy změn z pohledu nynějších rolí výrobce a dopravce.

Za roli user: manufacturer@company.com
- https://s-chain.tech/dashboard:
  - oranžový trojúhelník u https certifikace
  - Dávalo by nám logiku mít **Items** pod Produkty z hlediska hierarchie, tedy že **Items** jsou podmnožiny Produktů.
- https://s-chain.tech/products:
  - První sloupec: Product ID
  - Druhý sloupec: OK
- Detail produktu, např.: https://s-chain.tech/product/102
  - předpokládáme, že je to typ produktu, chybí nadpis
  - po stisku „create item“ odkazuje na: https://s-chain.tech/item/new/102, nadpis by měl být stejný jako volba – tedy Create Item, aby to korespondovalo s danou volbou
  - Co je property name?
  - Co je property value?
- https://s-chain.tech/item/new/102: Textbox s nadpisem INFO by se tedy mohl jmenovat něco jako Info about last state.
- https://s-chain.tech/products:
  - Přidat k nadpisu vysvětlení něco jako že to je seznam typů nebo druhů výrobků.
  - Nabízí se varianta jako druhý sloupec použít právě Product category a v něm list s categories.
  - Také nás napadlo, že by zde mohl být přidán sloupec s výrobní cenou.
  - Možná by se nám tu i líbil název katalog výrobků. Otázkou je co na to angličtina. **Catalogue** je asi trochu lepší.
  - V případě, že si můžeme filtrovat produkty podle data (created from, created to, updated from či updated to) pak se nabízí alespoň jednu z těchto položek zobrazovat jako sloupec.
- https://s-chain.tech/items
  - Nápad s klikátkem not in shipment je uživatelsky přívětivý. V souvislosti s tím, filter not in shipment - umožnit zákazníkovi zařadit odtud nějak **item do shipmentu**? Když už to máme ve filtrech?
  - Sloupce tabulky:
    - První sloupec: ID opět postrádá bližší specifikaci, tedy bychom tipovali seriové číslo/číslo šarže, tedy minimálně by tu mělo být uvedeno ID items.

Zvážit, jestli tam nedat popis "serial number". Což v realitě asi více odpovídá.
    - Sloupec č.4 by se mohl přidat do filtru - třídit podle toho, kdo s tím manipulovat naposled.
    - První sloupec je fajn, ale před ním by mohl být ještě sloupec **product id,** aby na první dobrou bylo jasné, z jakého produktu daná **Item** vznikla/pochází.
    - Možná zvážit přidat filtr Items podle katalogového ID (tzn, podle čísla produktu). např. 53 Mushrooms.
    - Přidání sloupců je asi mnoho: např. která směna/linka výrobek vyrobila, resp. který závod (v jaké lokaci). K diskusi/zamyšlení.
- https://s-chain.tech/shipment
    - shipment price jsou náklady na přepravu nebo cena produktu?! Pokud je to hodnota zásilky, tak bychom to viděli jako údaj, který nemá výrobce v zájmu předávat nikam kam to není nutné (krádež apod. Tyto info se předávají pouze v rámci např. celního řízení při platbě cla. Jinak je to interní údaj a myslíme si, že i velmi citlivý údaj).
    - Další možné rozšíření: Sloupec: ETA (estimated time of arrival – užitá zkratka)
- Detail shipmentu, např. https://s-chain.tech/shipment/57:
    - Nešlo nám změnit carrier, ***s chybou 500***.
    - Pokud není ještě carrier přidělen, pak by se mělo zobrazit přiřadit dopravce, nikoli update. Ale může to být i vedle sebe. A na tlačítku pak asi stačí ok.
    - Co je **estimated price**? Je to hodnota zásilky? A nebo cena dovozného?
    - Přidat položky **odjezd+příjezd**: respektive časový údaj, kdy by to mělo odjet z továrny, nebo respektive časový údaj plánované nakládka. A v souvislosti teda i termín **dodání** (nejpozdější).
    - Pro výrobce je důležitější informace, kolik v shipment listu je itemů, tedy by se mohla přesunout více nahoru. Mapka je asi méně důležitá informace, spíš vizuálně hezké, ne tak prioritně potřebné info.
- https://s-chain.tech/shipment:
    - shipment price: suma položek pro konkrétní zásilku, tedy celková hodnota na faktuře zásilky, která s danou zásilkou pojede(včetně packing listů - nezbytné pro clení). Tedy případně lze údaj odesílat EDI na celní zprávu (EORI-hlubší teorie)
    - sloupec ID přejmenovat na shipment ID.
    - vznik sloupce s názvy shipmentů - protože se systémem pracují lidé, nebo alespoň kategorie zásilky nebo něco podobného.
    - vznik sloupce ORIGIN.
    - vznik sloupce DESTINATION.
    - vznik slupce s názvem dopravce, je-li přidělen a pokud nemá tak se zobrazí no carrier.
    - vznik slupce počet kusů položek zásilky, nabízí se také možnost zobrazovat rozměry zásilky (váha, šířka, výška, hloubka - obzvláště důležité v rámci letecké přeprav), nebo údaj, je-li to na paletách nebo v TEU.
    - ? vznik sloupce - cena za přepravu - zde může být uvedena maximální částka, kterou jsem ochoten zaplatit resp. Úvaha, že dopravci většinou nabízí cenu, tedy jaký dopravce jakou cenu nabídl?
    - ? vznik sloupce např. boty na slovensko - pojmenování pro výrobce, tedy analogicky polidštit označení shipmentu. Může to být za id shipmentu.

Za roli user: carrier@company.com

- https://s-chain.tech/dashboard:
  - nefunguje track ID
  - položka **update cargo** je asi zbytečná?
- https://s-chain.tech/shipment
  - vesmes úpravy navrženy jako u pohledu výrobce
  - sloupec ID: přejmenovat na ID shipment
- https://s-chain.tech/cargo:
  - sloupec ID, analogicky přejmenovat na ID CARGO
  - V **Cargo listu**, resp. v tabulce by měla být informace odkud, kam čím co se veze a jak je to velké. Tedy sloupce:
    - origin
    - destination
    - weight (netto/brutto?)
    - rozměr zásilky/carga v*š*h
    - název přepravy (uhlí z lovosic/ tenisky z PUSANU, počítače pro česko etc.)
    - mohl/měl by tu být i příjemce
    - a odesílatel zboží
    - Také by nebylo na škodu a to jen úvaha, vznik sloupce Incoterms = přepravní podmínky (nebezpečné zboží) apod.
- Update cargo tlačítko: https://s-chain.tech/cargo/update/5:
  - new cargo by mohlo být tlačítko nikoli položka menu? Jen nápad.
- https://s-chain.tech/cargo/new:
  - Nadpis, asi spíš teda create new cargo.
  - Kromě origin a destination je potřeba také SR70 a název bodu (Praha hl.n. apod).
  - Sloupce v tabulce add new shipments to cargo rozšířit o jména zásilek tedy.
  - Sloupec ID rozepsat, je to teda shipment id.
  - Proč jsou tam zásilky in-transit? Ty už jedou, tak ty už nebudu chtít přidávat do carga, nebo ano? Jaká je za tím myšlenka? Spojování zásilek, nebo?
  - Plus slupce:
    - Origin
    - Destination
    - name (název přepravy/název obchodního případu) např.uhlí do chvaletic, boty uživatelsky přiblížit.

Body k diskuzi dále

- Proč není možnost požadavku:- GET /api/carrier/{id}/cargo
- Zvážení automatického ukončení přepravy po dosažení endpointu – nemusí se shodovat s praxí, kde při dosažení cílového bodu (mapa s možnou tolerancí polohy) nemusí dojet

k doručení shipmentu – celní závěra/VET kontrola, nutnost manipulace na skladě, procesy apod.

- Na železnici se používají v praxi kódy stanic **SR70**, které určují přesnou polohu vlaku. Také jsme zmiňovali možnost využití v praxi používaných UNLOCODE pro mezinárodní dopravu. **Jak je plánováno implementovat tuto skutečnost?** Viz excel návrhu tras pro testování.

V rámci testování jsme narazili na některé nejasnosti/otázky, dle domluvy zasíláme poznámky viz níže.

- Jak přidat Category produktu?
- V detailu produktu je v přehledu **Subproduct odkaz na sebe sama**.

Součástí dokumentu je soubor importovatelný do služby Postman - rozhranní za Oltis.

Príloha B - Závislosti jednotlivých projektov

Server

Java verzia: 14
Postgres 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

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
- 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
- com.nhaarman:mockito-kotlin:1.6.0
- com.github.javafaker:javafaker:1.0.2
- org.yaml:snakeyaml:1.27

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

Mobilná aplikácia

Kotlin verzia: 1.4.21
Nav verzia: 2.3.0
Lokálna URL: http://192.168.0.108:8080
Lokálna URL (emulátor): http://10.0.2.2:8080

Závislosti zo súboru **build.gradle**:

- com.android.tools.build: gradle: 4.1.3
- 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

Blockchain

Závislosti sú vždy aktualizované spúšťaním blockchainu. Spúšťanie je popísané v časti “Technická dokumentácia - Inštalácia a spustenie”

Technická dokumentácia

Inštalácia a spustenie

Predpoklady:

- Linux stroj (ideálne Ubuntu 18.04) (kvôli blockchainu)
    - Možné alternatívy: Virtuálny stroj
- go, python, NODE, npm, docker, docker-compose
- Mobilná aplikácia:
    - nainstalovane 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. Postupne spustiť blockchain, server a následne Vue.js frontend

Inštalácia mobilnej aplikácie do Android zariadenia je možná pomocou vytvoreného inštalačného súboru **schain.apk**.

Spustenie Blockchain

1. Navigovať sa do blockchain časti projektu a spustiť **beforeFirstStart.sh**
2. Vytvorenie blockchain organizácií (spustenie) pomocou **start.sh**

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

Spustenie servera

1. Navigovať sa do server časti projektu a spustiť **start-dev-first.sh**

- tento proces zároveň spustí server. V prípade opakovaného spustenia je potom možné spustiť server pomocou **start-dev.sh**
- **Možné problémy:**
    - v prípade novej verzie servera/blockchain je pravdepodobne nutné znovu spustiť **start-dev-first.sh**

Spustenie Vue.js frontend

1. Navigovať sa do fronted časti
2. Nainštalovanie potrebných modulov pomocou **npm install**
3. Spustenie webového rozhrania pomocou **npm run serve**