TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Markus Leemet

Mikroteenuse loomine ettevõtte 1oT andmete haldamiseks

Bakalaureusetöö (9 EAP)

Juhendaja(d): Helle Hein

Tartu 2023

Mikroteenuse loomine ettevõtte 1oT andmete haldamiseks

Lühikokkuvõte:

Käesoleva bakalaureusetöö eesmärgiks on luua ettevõttele 1oT andmehalduse mikroteenus, mis kogub kokku ja töötleb ettevõtte jaoks olulisi andmeid. Andmehalduse mikroteenuse sisendandmed pärinevad eksisteerivatest 1oT teenustest ja 1oT partneritelt. Mikroteenus viib andmed sobivale kujule ja edastab teenusess Power BI. Kirjalik töö annab ülevaate nii töödeldavatest andmetest kui ka andmehalduse mikroteenuse arhitektuurist. Töö lõplikuks väljundiks on anda ettevõttele 1oT ülevaade oma klientidest ja teenuste tarbimistest, et oleks võimalik langetada andmetel põhinevaid ärilisi otsuseid.

Võtmesõnad:

Mikroteenus, Spring Boot, Java, Power BI, 1oT

CERCS: P175 Informaatika, süsteemiteooria

Creation of Microservice for Managing 1oT Data

Abstract:

The aim of the bachelor’s thesis is to create a data management microservice for the 1oT. The created microservice collects the company's data from other microservices and 1oT's partners. Collected data is then processed and converted into the required structure and forwarded to Power BI. The written work provides an overview of the managed data and created microservice architecture. The overall objective is to give an overview to the 1oT about its customers and sold services so that it would be possible to make data drive business decisions.

Keywords:

Microservice, Spring Boot, Java, 1oT

CERCS: P175 Informatics, systems theory

Sisukord

1 Sissejuhatus

Ettevõte 1oT, mis asutati 2016 aastal, pakub asjade interneti seadmetele ühenduvust üle maailma. 1oT kliendid, kes tegelevad asjade interneti seadmete tootmise ja müümisega, ei pea vaevama end sellega, kuidas nende seadmed saavad eri riikides andmesidet kasutada, kuna ühe SIM-kaardiga saab võrku ühenduda igas maailma riigis. Lisaks on kõikide SIM-kaartide haldamiseks üks platvorm ja iga kuu väljastatakse kliendile üks kõikehõlmav arve [1].

Suur väärtus, mida 1oT oma klientidele pakub, on 1oT Terminal. Tegu on iseteeninduslatvormiga, mis laseb klientidel monitoorida ja hallata oma kasutuses olevaid SIM-kaarte. Kliendid saavad oma SIM-kaarte aktiveerida ja deaktiveerida vastavalt sellele, kas SIM-kaart peaks saama andmesidet tarbida või mitte. Lisaks pakub platvorm detailset SIM-kaardi põhist infot: kui palju mingil perioodil andmesidet tarbiti, mis oli viimane SIM-kaardi asukoht, mis oli viimane mobiilimast, kuhu SIM-kaart oli ühendunud, ja palju muud. Kõige selle saavutamiseks vajab 1oT Terminal andmeid välistelt partneritelt iga SIM-kaardi andmeside kasutuse kohta. 1oT kodulehel seisab, et platvormil on 1.4 miljonit SIM-kaarti [2]. Selleks, et kõikide kasutuses olevate SIM-kaartide puhul näidata täpset sessioonide põhist infot ja ka ajaloolist infot, tuleb 1oT-l hallata suures koguses andmeid.

Kogu selle andmehulga majandamise juures on täna peamiseks probleemiks, et neist puudub ammendav ülevaade. Andmetes peituvad teadmised, kuid kui need on struktureerimata ja töötlemata, ei saa neid kasutada kaalutletud otsuste langetamisel ja see on täna raisatud potentsiaal. Kuna inimene on suurte andmemahtude töötlemisel ebaefektiivne, kulukas ja kaldub tegema vigu, siis on vaja andmete haldamiseks luua tehniline lahendus. Loodava mikroteenuse eesmärk on andmete kogumine ja töötlemine, et oleks võimalik langetada kaalutletud ja andmetel põhinevaid ärilisi otsuseid.

Käesoleva töö käigus luuakse ettevõttele 1oT andmehalduse mikroteenus, mille eesmärgiks on viia algandmed kujule, mis on inimesele arusaadavad ja mille põhjal saab langetada kaalutletud otsuseid. Andmehalduse mikroteenus on eraldiseisev mikroteenus, mis suhtleb teiste 1oT mikroteenustega, saamaks vajalikud sisendandmed. Andmed töödeldakse ning edastatakse Microsofti teenusesse Power BI, mis võimaldab andmeid graafiliselt kuvada.

Kirjaliku töö esimene ja teine peatükk annavad ülevaate kasutatud mõistetest ja andmetest, millega andmehalduse mikroteenus tegeleb. Peatükid 4 – 8 kirjeldavad andmehalduse

mikroteenust ja selle arhitektuuri. Üheksandas peatükis kirjeldatakse mikroteenuse publitseerimist ning kümnendas peatükis on välja toodud tagasiside loodud mikroteenusele.

2 Mõisted ja terminid

1oT Terminal on ettevõtte 1oT poolt loodud SIM-kaartide haldus platvorm, mis lihtsustab asjade interneti seadmete haldust [1].

CDR (ingl *Call Detail Record*) on raamatupidamiseesmärkideks mõeldud fail, mis annab infot, mis teenuseid ja kui palju on tarbinud võrku ühendunud seade. CDR failist saab infot nii telefonikõnede, SMS-ide kui ka andmesidekasutuse kohta. CDR failidel ei ole ühtset formaati, ning nende sisu võib erineda [3].

Cron-lause võimaldab defineerida ajahetked erinevate funktsionaalsuste jooksutamiseks [4].

ICCID (ingl *Integrated Circuit Card Identification*) on SIM-kaardi unikaalne number, mis koosneb 18 - 22 numbrist. ICCID-d kasutatakse otsustamaks, millistesse võrkudesse seade saab ühenduda [5].

IMSI (ingl *International Mobile Subscriber Identity*) on unikaalne number, mida MNO-d kasutavad, et tuvastada võrgu kasutajaid ning on oluline osa SIM-kaardist. IMSI on tavaliselt 15 ühiku pikkune ja koosneb kolmest osast. Esimene osa on MCC (ingl *Mobile Country Code*), mis määrab riigi, kus võrgu kasutaja peamiselt asub. Teine osa on MNC (ingl *Mobile Network Code*), mis määrab MNO, millega võrgutarbija seotud on. Ning viimane osa on identifitseerimisnumber, mis on iga võrgutarbija puhul unikaalne [6].

Juurdepääsutasu (ingl *access fee*) on tasu, mida telekommunikatsiooniettevõtted võivad küsida, et seade saaks nende võrku ühenduda [7].

Mikroteenuste arhitektuur on rakenduse arhitektuur, mille puhul rakendus on jagatud mitmeteks eraldiseisvateks mooduliteks ehk mikroteenusteks. Kõik mikroteenused on eraldiseisvad protsessid, mis on võimelised omavahel erinevate protokollidega suhtlema. Mikroteenuste arhitektuuri vastandiks on monoliitne arhitektuur, mille puhul kogu rakendus on üks suur üksus. Mikroteenuste eelis monoliitse rakenduse ees on skaleeritavus ja võimalus erinevaid teenuseid publitseerida, ilma kogu rakendust publitseerimata [8].

MNO (ingl *Mobile Network Operator*) on telekommunikatsiooniettevõte, kes pakub oma klientidele võimalust üle õhu andmete tarbimiseks ja helistamiseks [9].

MSISDN (ingl *Mobile Station Integrated Services Digital Network*) on unikaalne number, mis võimaldab võrgus olevaid seadmeid tuvastada. Seda kasutatakse näiteks, et telefonikõne jõuaks õige seadmeni ja et vastuvõtja teaks, kellelt saabuv kõne pärineb. MSISDN on

maksimaalselt 15 kohaline number, mis koosneb riigikoodist, riigi võrgukoodist ja unikaalsest numbrist eristamaks SIM kaarte [10].

PLMN (ingl *Public Land Mobile Network*) on igale võrgule omane unikaalne 5 – 6 numbri pikkune kood. PLMN koosneb MCC-st (ingl *Mobile Country Code*) ja MNC-st (ingl *Mobile Network Code*) [11].

SIM-kaart (ingl *Subscriber Identity Module*) on sisseehitatud kiibiga kaart, mis hoiustab identifitseerimiseks vajalikku informatsiooni. See informatsioon on vajalik, et telekommunikatsioonieettevõte oleks võimeline seostama seadet kliendiga ja et otsustada, kas seade peaks saama võrguga ühenduda või mitte. Igal SIM-kaardil on oma IMSI ja ICCID, mida nendeks toiminguteks kasutatakse [12].

SMS MO (ingl *Mobile Originated*) on seadmelt saadetud SMS [13].

SMS MT (ingl *Mobile Terminated*) on seadmele saadetud SMS [13].

TADIG (ingl *Transferred Account Data Interchange Group*) on unikaalne kood, millega saab tuvastad MNO-d [14].

Võrk (ingl *network*) on vähemalt kahest seadmest koosnev grupp, mis on võimelised omavahel suhtlema. Seadmed võivad olla omavahel ühendatud füüsiliselt, kuid suhtlus võib toimuda ka üle õhu [15].

3 Taustainfo

Andmehalduse mikroteenus tegeleb kolme tüüpi andmetega: SIM-kaartide müümise tulude ja kuludega, andmete tarbimise kulude ja tuludega ning 1oT Terminali lisateenustega seotud tuludega. Nii SIM-kaartide müümisega kui ka SIM-kaartide tarbimisega kaasneb 1oT jaoks nii rahaline kulu kui ka tulu. Ainuke andmekategooria, millega ei ole seotud otseseid kulusid ettevõtte jaoks, on 1oT Terminali lisateenuste pakkumine. Käesolevas peatükis kirjeldatakse igat valdkonda, andmaks ülevaate valdkonnast endast, mis andmetega valdkonnas tegeletakse ning mis andmetega mikroteenus tegelema hakkab.

3.1 SIM-kaardid

Ettevõte 1oT pakub oma klientide asjade interneti seadmetele ühenduvust. Et seadmed saaksid võrku ühenduda, on neil vaja SIM-kaarti, mis määrab millistesse võrkudesse seade saab ühenduda. Kuna 1oT ei tooda ise SIM-kaarte, siis ostetakse need välistelt partneritelt sisse ja müüakse oma klientidele edasi. Ehk SIM-kaartide klientidele müümise puhul toimib 1oT kui vahendaja.

Siin on oluline ettevõtte jaoks jälgida, mis hinnaga ostetakse erinevate partnerite SIM-kaarte ja mis hindadega neid edasi müüakse, et teada kas tegu on kasumliku tegevusega. Lisaks tuleb jälgida SIM-kaartide klientidele saatmisega seotud tulusid. Oluline on ka jälgida trendi, mis vormiteguri ja telekommunikatsiooni ettevõtte SIM-kaarte enim müüakse, et saaks arvestada lao varudega. Saamaks ülevaate SIM-kaartide müümisega seotud infost, kogub andmehalduse mikroteenus kokku kõik klientide SIM-kaartide tellimused koos relevantsete andmetega.

3.2 Andmete tarbimine

Teine ja suurem kulude ja tulude allikas 1oT jaoks on klientide andmetarbimine. Ehk kui palju ja millistel tingimustel tarbivad klientide seadmed andmeid, saadavad SMS-e ning kasutavad helistamise funktsionaalsust. Et saada täielik ülevaade SIM-kaardi tasemel tarbimisest, on vaja andmeid: kui palju SIM-kaart erinevaid andmeid tarbis, kui palju 1oT selle tarbimise eest maksis ning kui palju 1oT selle tarbimise eest teenis. Tarbimise ja tulude andmed pärinevad 1oT enda teistest mikroteenustest. Kulude andmed pärinevad 1oT partnerite sisend CDR failidest.

Andmehalduse mikroteenus peab suutma need andmed kokku koguda ja omavahel kombineerida. Oluline on nende andmete puhul jälgida, millised kliendid kui palju andmeid tarbivad, millistes võrkudes on suurim tarbimine ja mis riigid on kõige kasumlikumad.

3.3 1oT Terminali lisateenused

1oT Terminal pakub lisaks oma baasfunktsionaalsusele klientidele võimalust kasutada tasulisi lisateenuseid. Neid lisateenuseid saab klient oma äranägemise ja vajaduse järgi nii aktiveerida kui ka deaktiveerida. See võimaldab igal kliendil kasutada platvormi talle kõige otstarbekamal viisil. Näiteks saab klient aktiveerida API lisateenuse ja selle asemel, et oma SIM-kaarte hallata läbi 1oT Terminali kasutajaliidese, saab ta 1oT Terminali API-i ühendada oma süsteemiga. Tabelis 1 on välja toodud kõik klientidele pakutavad lisateenused kategooriate kaupa.

Tabel 1. 1oT Terminali lisateenused kategooria kaupa.

Andmeanalüüs	**Integratsioonid**	**Töövoog**
Diagnostika	Libelium	Töövoo automatiseerimine
Intelligentne ühenduvus	TeamViewer	Tootlikkuse tööriistakomplekt
Analüütika	MS Teams	API
KPI aruanne	Slack	Kliendivaade

Lisateenuste müümise puhul ei ole otseseid kulusid, kuna kaudseid tegureid nagu tööjõukulu ja infrastruktuuri kulusid ei arvestata sisse ühegi kategooria puhul. Lisateenustega seotud andmete puhul on oluline koguda kokku kõik lisateenustega seotud tulud, et oleks ülevaade, kui palju kliendid mingeid lisateenuseid kasutavad, saamaks ülevaadet, mis on klientidele olulised teenused. Samas annab see ka ülevaate, mis lisateenuseid ei kasutata, ning kas nendele oleks vaja pöörata rohkem tehnilist tähelepanu või ei ole hind vastavuses klientide ootustega.

4 Tehnilised nõuded

Kuna andmehalduse mikroteenus luuakse ettevõttele, siis tuleb selle loomisel arvestada 1oT nõudeid nii tehnoloogia valiku, publitseerimise kui ka funktsionaalsuste puhul. Tabelis 2 on välja toodud nii funktsionaalsed nõuded, tehnoloogilised nõuded kui ka stiilinõuded.

Tabel 2. Nõuded andmehalduse mikroteenusele.

Funktsionaalsed nõuded	Tehnoloogilised nõuded	Stiilinõuded
Andmehalduse mikroteenus peab töötama iga päev 24 tundi.	Kasutada tuleb samu tehnoloogiaid, mida kasutavad teised 1oT mikroteenused.	Koodibaas peab olema inglise keeles.
Andmed tuleb edastada ettenähtud struktuuriga Power BI teenuse voogedastus andmestikkudesse kasutades HTTP päringuid.	Publitseerimiseks tuleb kasutada AWS EC2 teenust.	Koodibaas peab järgima DRY, Curly's Law ja KISS põhimõtteid.

Tehnoloogiate valiku puhul lähtutakse põhimõttest, et kõik mikroteenused peaksid kasutama samu tehnoloogiaid ja arhitektuuri, et nende edasine arendus ja haldamine oleks ettevõttele võimalikult lihtne. Mikroteenuse publitseerimise puhul kasutatakse samuti juba kasutusel olevat teenust.

Andmehalduse mikroteenus peab kasutama samu tehnoloogiad, mida kasutavad teised 1oT mikroteenused. See tähendab, et mikroteenus peab kasutama Spring Boot raamistikku ja Java programmeerimiskeelt. Rakenduse ehitamiseks tuleb kasutada Gradle tööriista. Andmebaasina tuleb kasutada PostgreSQL-i ja selle versioonihalduseks FlyWay'd.

Nii loodav mikroteenus kui ka selle poolt kasutuses olev andmebaas peavad jooksma AWS EC2 (ingl *Elastic Compute Cloud*) teenuses. EC2 on pilvemajutusteenus, mis pakub laia valikut erinevaid servereid oma rakenduste jooksutamiseks. Kuna andmehalduse mikroteenus peab töös olema iga päev 24 tundi, et vastu võtta sõnumeid teistelt mikroteenustelt, tuleb publitseerimiseks valida selline server, millega on minimaalsed kulud, kuid mis suudab rahuldada andmehalduse mikroteenuse ressursi nõudeid. See tähendab, et mikroteenus peab

töötama nii ratsionaalselt kui võimalik, et hoida ettevõtte jaoks publitseerimiskulud võimalikult madalad.

Lisaks tehnilistele nõuetele on andmehalduse mikroteenusele ka stiili nõuded. Esiteks peab kogu koodibaas olema inglise keeles. Teiseks peab kood järgima peamiseid programmeerimise printsiipe nagu DRY (ingl *Don't Repeat Yourself*), Curly's Law ja KISS (ingl *Keep it Simple, Stupid!*). Lisaks peab kood olema kergesti loetav ning kommentaaride asemel tuleks kasutada kirjeldavaid meetodi, muutuja ja klassi nimesid.

Funktsionaalsete nõuete koha pealt peab andmehalduse mikroteenus olema võimeline koguma kokku andmed ja edastama need Microsofti teenusesse Power BI. Andmed tuleb edastada kategooriate põhiselt Power BI andmevoogudesse. Saadetavad andmed peavad vastama etteantud struktuuridele. Andmed tuleb edastada teenusesse Power BI kasutades selle andmete voogedastuse funktsionaalsust, mis võimaldab andmeid saata HTTP päringutega. Andmehalduse mikroteenus peab olema kaetud testidega veendumaks, et uute funktsionaalsuste lisamisel jääks olemasolev kood töötama.

5 Kasutatud tehnoloogiad

Andmehalduse mikroteenus on 1oT mikroteenus, mis kasutab sarnaseid tehnoloogiaid teistele 1oT mikroteenustele. Järgnevalt on välja toodud peamised tehnoloogiad, mis on olulised loodava mikroteenuse ülesehituse puhul.

5.1 Spring Boot

Spring Boot on avatud lähtekoodiga Java raamistik, mis loodi aastal 2014, et lihtsustada Java rakenduste publitseerimist. Spring Boot põhineb Spring raamistikul, mis aitab Java rakenduste loomist ja jooksutamist. Spring Boot raamistik koosneb Spring raamistikust ja sisseehitatud serverist. Kuna Spring Boot põhineb enam konventsioonidel kui konfiguratsioonidel, ei eelda see XML konfiguratsioonifaile, mida eeldab Spring raamistik. Kuna arendajad ei pea ise konfiguratsioonifaile looma, on rakenduse loomine ja publitseerimine lihtsustatud [16].

5.2 Gradle

Gradle on tööriist rakenduste ehitamise automatiseerimiseks. Gradle on tuntud oma paindlikkuse poolest ning selle ehitusprotsess hõlmab endas nii koodi kompileerimist, testimist, linkimist kui ka tarkvarapakettide loomist. Gradle'it kasutatakse näiteks Java, Scala, C, C++ kui ka Groovi rakenduste ehitamiseks. Gradle hõlmab endas alternatiivsete ehitamistööriistade Maven ja Ant positiivseid omadusi ning on kõrvale jätnud nende miinused [17].

5.3 Java

Java on programmeerimiskeel, mis loodi aastal 1995 ettevõtte Sun Microsystems poolt ning omandati 2009 aastal Oracle Corporation'i poolt [18]. Tänaseks päevaks on Javast saanud laialdaselt kasutatud programmeerimiskeel. Java on olemuselt objektorienteeritud, platvormide ülene ja võrgukeskne keel. Javat kasutatakse palju tema kiiruse, turvalisuse ja usaldusväärsuse tõttu ning sellega saab luua mitmeid erinevaid rakendusi, kaasa arvatud mobiilirakendusi, taustsüsteeme ja rakendusliideseid. Kuigi Java on olemas olnud pikalt, leiab see endiselt kasutust, kuna sellele leidub palju õppematerjale, sellel on aktiivne kogukonna toetus ja olemas on ka kvaliteetsed arendustööriistad [19].

5.4 PostgreSQL

PostgreSQL on Michael Stonebraker'i poolt 1986 aastal loodud avatud lähtekoodiga objekt relatsiooniline andmebaasi süsteem, mis põhineb SQL keelel. PostgreSQL-i saab kasutada kõigil suurematel operatsioonisüsteemidel. PostgreSQL on laialdaselt kasutatud, kuna see on kiire ja skaleeritav. See pakub samaaegsete andmete lugemise ja kirjutamise võimalust, mis võimaldab mitmel protsessil korraga andmeid lugeda ja kirjutada. Lisaks on PostgreSQL toetatud paljude populaarsete programmeerimiskeelte poolt, sealhulgas Java, Python, JavaScript, C, C++, Rubi ja teiste [20].

5.5 Hibernate

Hibernate on avatud lähtekoodiga objekt-relatsioonivastendus tööriist Javas. See on kergekaaluline alternatiiv JDBC-le. Hibernate ei ole andmebaasi spetsiifiline, ehk sama lahendust saab kasutada mitmete erinevate andmebaasidega ja Hibernate tegeleb sisemiselt andmebaasi spetsiifiliste ühenduste loomisega ja andmebaasipäringute koostamisega. Toetatud on kõik enim kasutatud andmebaaside tehnoloogiad, sealhulgas PostgreSQL ning MYSQL. Hibernate'i peamised tugevused alternatiivide ees on andmebaasi sõltumatus, parem koodi loetavus ja tänu vahemälu kasutamisele parem jõudlus [21].

5.6 FlyWay

FlyWay on avatud lähtekoodiga tööriist andmebaaside migratsioonide implementeerimiseks. See võimaldab andmebaasi muudatused defineerida kas SQL skriptidena või Java koodina ja neid loodud migratsioone kas käsurealt või programmi ehitamise käigus jooksutada. FlyWay peamine väärtus on, et arendajad ei pea teadma, mis staatuses on mingi konkreetne andmebaas, vaid FlyWay peab ise arvet, mis staatuses konkreetne andmebaas on ja jooksutab ise vajalikud migratsioonid, et andmebaas oleks uuendatud. Et hoida järge andmebaasi versiooni ja jooksutatud migratsioonide üle, kasutab FlyWay andmebaasis enda meta tabelit *SCHEMA_VERSION* [22].

5.7 Redis

Redis (ingl *Remote dictionary Server*) on avatud lähtekoodiga andmekogu, mida kasutatakse peamiselt vahemäluna. Redis võimaldab andmeid salvestada erinevate andmestruktuuridena, mis võimaldavad kiiret andmete kättesaamist. Redis on tuntud kui NoSQL andmebaas, mis salvestab väärtuseid võti/väärtus paaridena ning hoiab oma andmeid mälus,

mitte kõvakettal. Redise vahemälu kasutamise eelis tuleneb sellest, et andmete küsimisel ei tule pöörduda kõvaketta poole, mis võib tekitada rakenduse kiirusele pudelikaela. Efektiivseks kasutuseks tuleks Redise vahemälus hoida andmeid, mida päritakse rakenduse toimimiseks tihti. Populaarne kasutus Redise vahemälule on sessioonide hoiustamine [23].

5.8 RabbitMQ

RabbitMQ on avatud lähtekoodiga sõnumimaakler, mida kasutatakse erinevate rakenduste vahel suhtlemiseks. Sõnumimaaklerid võimaldavad erinevatel rakendustel üksteistega suhelda ning see omakorda võimaldab rakenduste eri funktsionaalsused üksteistest eraldada, mis on omakorda eelduseks mikroteenuste arhitektuuri implementeerimiseks. RabbitMQ omab ka ajutist sõnumite salvestamise funktsionaalsust, et andmed ei läheks kaotsi. RabbitMQ kasutab järjekordi veendumaks, et sõnumid jõuavad õigesse kohta kohale. Järjekorda saadavad väljastajad (ingl *publisher*) sõnumeid ja tarbijad (ingl *consumer*) võtavad järjekorrast sõnumeid. RabbitMQ on kirjutatud programmeerimiskeeles Erlang ja kasutab sisemiselt AMQP protokolli (ingl *Advanced Message Queuing Protocol*). RabbitMQ vahevara saab kasutada mitmete erinevate programmeerimiskeelte ja operatsioonisüsteemidega. Lisaks pakub RabbitMQ veebipõhist platvormi keskkonna monitoorimiseks ja haldamiseks [24].

5.9 Docker

Docker on avatud lähtekoodiga platvorm, mis võimaldab publitseerida ja hallata rakendusi igas keskkonnas. Docker võimaldab seda tänu konteineritele, mis on standardiseeritud ja käivitatavad komponendid, mida saab jooksutada iga operatsioonisüsteemiga. Kuigi Docker ei ole ainuke võimalus rakenduse konteineriseerimiseks, on see tänapäevaks saanud konteineriseerimise sünonüümiks [25].

5.10 *Common* teegid

Ettevõte 1oT kasutab tarkvaraarenduse puhul mikroteenuste arhitektuuri, seega eraldiseisvad funktsionaalsused on jaotatud eraldiseisvatesse projektidesse. Kuid tihti on nendel mikroteenustel vaja kasutada mingit sarnast funktsionaalsust ning selle asemel, et seda igas projektis duplikeerida, on 1oT kasutusele võtnud *common* teegid. Kõik universaalne loogika, mida on vaja kasutada rohkem kui ühes mikroteenuses, on viidud eraldi teekidesse, mida saab mikroteenusesse importida ning seega järgitakse programmeerimise DRY (ingl *Don't Repeat Yourself*) põhimõtet.

5.11 Quartz

Quartz on avatud lähtekoodiga tööde ajastamise teek, mida saab integreerida Java rakendustega. Tegu on paindliku tööriistaga, mis võimaldab tööde jooksutamist spetsiifilistel ajahetkedel. Defineerimaks, millal konkreetne kood peaks jooksma, kasutab Quartz cron-lauset, mis koosneb 6st kuni 7st tühikuga eraldatud väärtusest. Iga cron-lause sisaldab sekundi, minuti, tunni, kuupäeva, kuu ja nädalapäeva väärtust ning lisaks võib veel sisaldada aasta väärtust [4]. Näide cron-lausest on *1 * * * * ? **, mis tähendab "iga minuti esimesel sekundil".

6 Arhitektuur

Andmehalduse mikroteenuse tehnoloogiate valik põhineb teistel 1oT mikroteenuste tehnoloogiatel. Samade tehnoloogiate valik võimaldab tulevikus minimaalse vaevaga muudatusi teha projektis ka isikutel, kes ei ole eelnevalt koodibaasiga tuttavad. Lisaks on 1oT-l kergem hallata kõiki mikroteenuseid kui need kasutavad samu tehnoloogiaid.

Kaustas *src/main/java/com/oneot/reports/conf* olevad konfiguratsiooni klassid on standardsed 1oT mikroteenuse konfiguratsioonifailid, mis on eelnevalt eksisteerivatelt mikroteenustelt üle kantud. Samade konfiguratsioonifailide kasutamine tagab samuti mikroteenuste sarnase toimimise, mis teeb lihtsamaks tulevikus tiimil mikroteenuse haldamise.

Spring Boot projektis on loodud konfiguratsioonid kahele erinevale profiilile. Nendeks profiilideks on *dev* ja *prod*. Profiili *dev* kasutamine on mõeldud arenduseaegseks kasutamiseks. Kui *dev* profiil on aktiveeritud, kasutab rakendus *application-dev.yml* ja *logback-dev.xml* konfiguratsioonifaile. Profiil *prod* on mõeldud rakenduse publitseerimiseks. Kui profiil *prod* on aktiivne võtab rakendus konfiguratsioonid failidest *application-prod.yml* ja *logback-prod.xml*. Turvalisuse eesmärgil kasutab *application-prod.yml* keskkonna muutujaid. See tagab, et delikaatne info ei oleks defineeritud koodibaasis.

6.1 Vahemälu

Andmehalduse mikroteenus kasutab oma funktsioonide täitmisel vahemälu klasse. Tegu on spetsiaalsete Java klassidega, mis hoiavad mälus kiiresti ja tihti vajaminevat infot. Andmeid hoitakse *HashMap* klassis, mis teeb nende leidmise kiireks ja efektiivseks. Lisaks on nendes vahemälu klassides defineeritud funktsionaalsus andmete sisse lugemiseks, uuendamiseks ja kustutamiseks. Vahemälu klassid saavad andmed andmebaasi tabelitest ning on kasutusel mikroteenuse protsesside kiirendamiseks. Kuna vahemälu klasse kasutades küsitakse andmeid mälust, mitte andmebaasist, on andmete leidmine kiirem ja efektiivsem. Järgnevalt on välja toodud kõik defineeritud vahemälu Java klassid koos nende otsarvetega.

- ***AccessFeeCache.java*** klassi kasutatakse kontrollimaks, kas mingile PLMN-ile on määratud ühendustasu.
- ***KpnRatePlanPricesCache.java*** klassi kasutatakse, et saada KPN-i paketi hind.
- ***NetworkCache.java*** klassi kasutatakse, et saada detailne võrgu info.
- ***SimCache.java*** klassi kasutatakse, et saada detailne SIM-kaardi info.

Peale iga cron-lausega defineeritud protsessi lõppu märgitakse kehtetuks kõik vahemälu klassid. See tähendab, et järgmine kord kui mingi protsess vahemälu klassist andmeid küsib, tuleb andmed uuesti andmebaasist vahemällu sisse lugeda. See tagab, et andmed oleks alati ajakohased.

6.2 Andmebaas

Andmehalduse mikroteenus vajab funktsioneerimiseks andmebaasi, milles hoiustada andmeid, mis ootavad saatmist teenusesse Power BI, kui ka sisse loetud andmete salvestamiseks, mis alles ootavad töötlemist. Selleks otstarbeks kasutab andmehalduse mikroteenus Timescale andmebaasi, mis on täiendatud PostgreSQL andmebaas. Andmebaasi struktuuri muudatuste haldamiseks on kasutusel FlyFay, mis on versioonikontroll andmebaasile.

Kuna 1oT erinevate mikroteenustega töötab korraga mitmeid inimesi ja iga uus funktsionaalsus võib muuta olemasolevat andmebassi struktuuri, siis on 1oT iga mikroteenuse puhul võtnud kasutusele FlyWay, mis tagab, et igas keskkonnas jookseks õige versiooni ja struktuuriga andmebaas. FlyWay lubab defineerida kahte tüüpi migratsioone: migratsioone, mida jooksutatakse vaid üks kord ja migratsioone, mida jooksutatakse igal rakenduse käivitamisel. Ühekordsed migratsioonid järgivad nimemalli *V${aasta}${kuu}${kuupäev}${tund}${minut}__${kirjeldus}.sql$* ja mitmekordsed migratsioonid nimemalli *R__${kirjeldus}.sql*. Näiteks on andmehalduse mikroteenuses loodud FlyWay migratsioon nimega *V202110291138__create_networks_table.sql*, mis loob uue tabeli võrkude info hoiustamiseks. Operatsioonid, mida tuleb ainult üks kord jooksutada, defineeritakse ühekordsetes migratsioonides ning käsklused, mida peaks igal rakenduse käivitamisel jooksutama, on defineeritud mitmekordsetes migratsioonides. Mitmekordsetes migratsioonides tasub defineerida näiteks andmebaasi vaateid ja funktsioone. Andmehalduse mikroteenuses on näiteks mitmekordne migratsioon nimega *R__cost_and_revenue_per_sim.sql*, mis loob olemasolevate tabelite põhjal uue vaate, kus on kombineeritud nii andmetarbmise kulud ja tulud.

Andmehalduse mikroteenuse andmebaasis on nii mikroteenuse enda loodud andmebaasi struktuurid (lisa 1) kui ka 1oT *common* teekide poolt loodud struktuurid (lisa 2), mis on ühised kõikide 1oT mikroteenuste puhul.

6.3 Konfiguratsioonid

Andmehalduse mikroteenus sõltub oma töös välistest parameetritest, mida ei tasu koodi sisse kirjutada, kuna see ei ole turvaline ning on ebapraktiline. Järgnevalt on välja toodud *configuration* andmebaasitabeli väärtused koos seletustega.

- **BILLING_IMPORT_STORAGE** on viide AWS S3 teenuse URL-ile.
- **BACKEND_URL** on viide 1oT *Backend* mikroteenuse URL-ile.
- **POWER_BI_COST_AND_REVENUE_URL** on viide *Cost and revenue* voogedastus andmestikule.
- **POWER_BI_APP_CONSUMPTION_URL** on viide *Addons* voogedastus andmestikule.
- **POWER_BI_SIM_ORDER_URL** on viide *Sim orders* voogedastus andmestikule.

Common-conf teegi poolt loodud *configuration* tabel on mõeldud rakenduse tööks vajalike väliste parameetrite hoiustamiseks. Tabelis tuleb iga väärtuse puhul defineerida nii võti, väärtus kui ka konfiguratsiooni kirjeldus.

6.4 Logimine

Erinevate sündmuste logimine on oluline osa rakendustest. See annab ülevaate rakenduse seisust ning juhul kui ette tuleb ootamatusi, on logide põhjal vigade üles leidmine lihtsustatud. Ka andmehalduse mikroteenus logib erinevaid sündmuseid. Logitakse nii oodatud sündmuseid, nagu näiteks failide edukad sisse lugemised kui ka erindid, mida ei tohiks mikroteenuse töös esineda. Mikroteenus kasutab erinevate sündmuste logimiseks SLF4J (ingl *Simple Logging Facade for Java*) teeki. Tegu on Javale mõeldud logimise raamistikuga. Logimise konfigureerimiseks on projektis failid *logback-dev.xml* ja *logback-prod.xml* ning nendest failidest valitakse see, mis vastab aktiivsele Spring Boot profiilile. Ehk kui kasutatakse *dev* profilli, siis kasutatakse *logback-dev.xml* faili konfiguratsiooni ja kui kasutatakse *prod* profiili kasutatakse *logback-prod.xml* faili konfiguratsiooni.

6.5 Sisendandmed

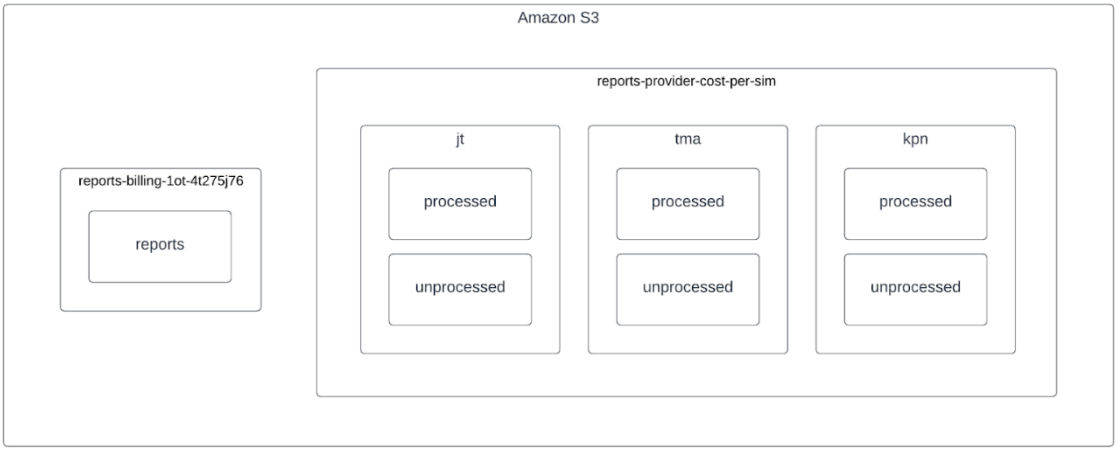
Andmehalduse mikroteenus toetub andmete saamisel projektivälistele failidele, mida kasutatakse, et andmed sisse lugeda ja andmebaasi salvestada. Need failid jagunevad kahte

kategooriasse. Esimeses kategoorias on failid, mis sisaldavad 1oT kulusid ja teises kategoorias on failid, mis sisaldavad tulusid.

Kulude failid pärinevad 1oT partneritelt, kes on telekommunikatsiooniettevõtted. Iga kuu genereeritakse partnerite poolt uus sisendfail eelmise kuu andmete kohta. Ehk iga kuu on vaja läbi töötada kolm uut sisendfaili, mis on seotud kuludega. Partneritelt pärinevad näidisfailid, mis ei sisalda päris andmeid, on välja toodud lisades 3 – 5.

Tulufailid pärinevad 1oT enda teistest mikroteenustest. Tulufaile on kahte tüüpi, esimene sisaldab SIM-kaartidega seotud tulusid ja teine fail sisaldab 1oT Terminali lisateenustega seotud tulusid. SIM-kaartidega seotud tulud asuvad failis nimega *cost_per_sim_2021-12.csv*, kus faili nimes olen aasta ja kuu on asendatud vastava aasta ja kuuga, mille kohta fail andmeid sisaldab. Terminali lisateenustega seotud tulud asuvad failis *app_consumption_2021-12.csv*, kus samuti faili nimes olev aasta ja kuu kajastavad perioodi, mille kohta failis olevad andmed käivad. Lisa 6 näitab Terminali lisateenuste tulude faili struktuuri ja lisa 7 SIM-kaartide tulude faili struktuuri.

Nii kulu- kui ka tulufailid jõuavad iga kuu Amazoni S3 teenusesse, mis on andmete hoiustamiseks mõeldud teenus. Joonisel 1 on näha teenuse S3 kaustade struktuur, kus kulusid kajastavad failid on kaustas *reports-provider-cost-per-sim* ja tulusid kajastavad failid *reports-billing-1ot-4t275j76* alamkaustas *reports*.



Joonis 1. Amazon S3 kaustade struktuur.

Näidistena esitatud failid näitavad failide struktuuri ja millist sorti andmeid need sisaldavad, kuid tegu ei ole päris andmetega, vaid juhuslikkuse funktsioone kasutades saadud andmetega.

6.6 Cron-laused

Protsesside perioodiliseks jooksutamiseks ja nende ajastamiseks kasutab andmehaldus mikroteenus *Common-cron* teeki. See on 1oT poolt loodud teek, mis lihtsustab cron-lausete defineerimise ning see võimaldab protsesse jooksutada andmebaasi konfiguratsioonide põhjal. Selleks tuleb *cronjob* tabelisse teha sissekanne, kus on määratud järgnevad väärtused:

- **Nimi** (*name*) on cron-protsessi tuvastamiseks mõeldud väärtus.
- **Lubatud** (*enabled*) on tõeväärtus, mis määrab ära kas cron-lauset peaks käitama või mitte.
- **Cron-lause** (*expression*) on spetsiaalse struktuuriga sõne, mis määrab ajahetke protsessi käivitamiseks.
- **Klass** (*class*) on Java klass, mis laiendab *Job* abstraktset klassi ja milles on defineeritud *execute* meetod, mida jooksutatakse vastavalt cron-lausele.
- **Parameetrid** (*params*) on lisa parameetrid, mida saab *execute* meetodile kaasa anda.
- **Minimaalne aega käivituste vahel millisekundites** (*min_next_excecution_delay_in_milliseconds*) on minimaalne aeg eelmisest *execute* meetodi jooksutamisest, mis oodatakse enne selle uuesti käitamiseest.

Lisaks nendele kohustuslikele veergudele on *cronjob* tabelis veel info väljad, mis annavad ülevaate cron-lausest ja selle viimasest jooksutamisest.

- **Id** (*id*) on tabeli primaarvõti.
- **Viimati jooksutatud** (*last_executed*) näitab ajatemplina viimast *execute* meetodi käitamise aega.
- **Ajavöönd** (*timezone*) on kohalik ajavöönd.
- **Viimase jooksutamise staatus** (*last_status*) näitab kas viimane *execute* meetodi jooksutamine oli edukas (*success*) või tekkis meetod jooksutamisel viga (*failed*).

- **Viimase jooksutamise kestus millisekundites** (*last_execution_time_in_milliseconds*) näitab, kui pikk oli viimase *execute* meetodi käitamise aeg.

Selleks, et andmehalduse mikroteenus suudaks vajalikud andmed sisse lugeda ja edasi saata, on vaja perioodiliselt jooksutada erinevaid protsesse ning nende protsesside ajastamiseks ja jooksutamiseks kasutataksegi *Common-cron* teeki, mille abil saab defineerida, millal on vaja kindlad protsessid käivitada.

Tabel 3. Cron-lausete konfiguratsioonid.

Nimi	Cron-lause	Klass
costPerSim.parsing	0 51 12 * * ?	com.oneot.reports.cron.CostPerSimParsingJob
costPerSim.reading.jt	0 40 12 * * ?	com.oneot.reports.cron.CostPerSimReadingJob
costPerSim.reading.tma	0 31 12 * * ?	com.oneot.reports.cron.CostPerSimReadingJob
simOrder.pushing	0 0 12 * * ?	com.oneot.reports.cron.SimOrderPushingJob
costPerSim.reading.kpn	0 0 12 * * ?	com.oneot.reports.cron.CostPerSimReadingJob
revenue.parsing	0 25 6 * * ?	com.oneot.reports.cron.RevenueParsingJob
costPerSim.pushing	0 59 9 * * ?	com.oneot.reports.cron.CostPerSimPushingJob
appConsumption.pushing	0 47 6 * * ?	com.oneot.reports.cron.AppConsumptionPushingJob

Tabelis 3 on välja toodud kõik mikroteenuse kasutatavad cron-laused ja nende poolt jooksutatavad protsessid. Kõiki *cronjob* andmebaasi tabeli veerge ei ole eelmainitud tabelisse lisatud tabeli formaadi limiidi tõttu.

6.7 Redis

Redis on ühine vahemälu kõikidele 1oT mikroteenustele, kus hoitakse andmeid, millele on erinevatel mikroteenustel sageli vaja ligi pääseda. Andmehalduse mikroteenus vajab Redist, et saada infot SIM-kaartide, võrkude, riikide kui ka 1oT klientide kohta. Et Java rakendus oleks võimeline suhtlema Redisega, on mõistlik kasutada mõnda valmisolevat teeki.

Andmehalduse mikroteenus kasutab selleks otstarbeks *Maven Central Repository* teeki *org.springframework.boot:spring-boot-starter-data-redis.*

6.8 RabbitMQ

Andmehalduse mikroteenus kasutab RabbitMQ sõnumimaaklerit, et kuulata sõnumeid, mida teised 1oT mikroteenused edastavad. RabbitMQ konfigureerimiseks on kasutatud *Maven Central Repository* teeki *org.springframework.boot:spring-boot-starter-amqp.*

Peamine viis, kuidas erinevad 1oT mikroteenused omavahel suhtlevad, on RabbitMQ sõnumid. Andmehaldus mikroteenusel on vaja saada infot kahte tüüpi sündmuste kohta, mille puhul RabbitMQ kasutamine on optimaalne lahendus. Esimene sündmus, mille kohta mikroteenus infot vajab on klientidele SIM-kaartide saatmine ja teine sündmus on SIM-kaartide andmete tarbimine.

```
public class SimOrderEvent {
   private String form;
   private Long companyId;
   private String orderId;
   private String provider;
   private Integer simCount;
   private YearMonth period;
   private BigDecimal simCost;
   private BigDecimal shippingRevenue;
   private BigDecimal simUnitsTotalRevenue;
   private List<SimCountAndBuyInCost> simCountsAndBuyInPrices;
}
```

Joonis 2. RabbitMQ *SimOrderEvent* sõnumi struktuur.

Joonisel 2 on välja toodud *SimOrderEvent* sõnumi struktuur, mida andmehaldus mikroteenus kuulab. See sõnum edastatakse iga kord, kui kliendile saadetakse välja SIM-kaardid. Sõnum sisaldab infot nii kliendi kohta kui ka SIM-kaartide kohta. Lisaks on sõnumis välja toodud nii transpordikulud kui ka SIM-kaartide ostu ja müügi hinnad.

```
public class CdrBatch {
   private CdrType cdrType;
   private Long simId;
   private Network network;
   private List<Cdr> cdrs;
   private String provider;
}
```

Joonis 3. RabbitMQ *CdrBatch* sõnumi struktuur.

Teine sõnumi tüüp, mida andmehaldus mikroteenus kuulab, on välja toodud joonisel 3. See sõnum sisaldab CDR infot ja annab infot SIM-kaartide andmetarbimise kohta. Sõnumis on info andmeid tarbinud SIM-kaardi kohta, võrgu kohta ja ka sessioonipõhine info tarbimisest. Sessioonipõhine info on *cdrs* muutujas, kus listi iga element vastab üksikule sessioonile.

6.9 Power BI

Andmete lõplikuks visualiseerimiseks kasutatakse Microsofti teenust Power BI (Ingl *Business Intelligence*), mis on tööriist erinevate andmehulkade importimiseks ja visuaalseks kuvamiseks.

Power BI võimaldab mitmeid erinevaid võimalusi andmete importimiseks. Kuid funktsionaalse nõude kohaselt edastab andmehaldus mikroteenus kogutud andmed HTTP POST päringutega. See võimaldab protsessi täielikult automatiseerida ja välistab manuaalse andmete liigutamise. Selle protsessi toimimiseks on vaja defineerida teenuses Power BI voogedastus andmestikud, kuhu saab andmeid saata.

Power BI's voogedastus andmestiku loomiseks on vaja defineerida andmestiku nimi ja andmestruktuur, millel andmestik põhineb. Iga defineeritud välja puhul tuleb defineerida ka selle välja andmetüüp. Kolm andmetüüpi, mille vahel valida saab, on tekst, number ja kuupäev.

Voogedastuse andmestiku loomise järel genereerib Power BI URL-i, mida saab kasutada andmete HTTP POST päringuga saatmiseks. Genereeritav URL on formaadiga *https://api.powerbi.com/beta/${workspace}/datasets/${dataset}/rows?key=${key}* ning see on kogu vaja minev info, et andmeid edastada. See on ka kogu konfiguratsioon, mida on vaja Power BI teenuse poole pealt teha.

1oT lisateenuste Power BI andmestiku struktuur on esitatud joonisel 4. See mikroteenuse Java klass vastab Power BI *Addons* andmestikule, mida kasutatakse 1oT lisateenuste müümise info haldamiseks.

```
public class AppConsumptionRowDto {
   private Long invoiceId;
   private String invoiceNr;
   private YearMonth period;
   private Long companyId;
   private String companyName;
   private Long packageId;
   private String packageName;
   private PriceType priceType;
   private PriceSubtype priceSubtype;
   private BigDecimal quantity;
   private BigDecimal price;
   private BigDecimal total;
}
```

Joonis 4. Power BI *Addons* andmestiku struktuur.

SIM-kaartide müümisega seotud info kuvamiseks on Power BI's loodud *Sim orders* andmestik, mille struktuur vastab joonisel 5 välja toodud Java klassile.

```
public class SimOrderRowDto {
   private BigDecimal shippingRevenue;
   private BigDecimal revenue;
   private String formFactor;
   private BigDecimal profit;
   private Integer simCount;
   private BigDecimal buyIn;
   private YearMonth period;
   private String provider;
   private String orderId;
   private String company;
}
```

Joonis 5. Power BI *Sim orders* andmestiku struktuur.

Sim orders andmestikule vastav Java klass on välja toodud joonisel 6. See klass vastab Power BI's *Cost and revenue* andmestikule ning see andmestik annab ülevaate SIM-kaartide andmetarbimisega seotud kuludest ja tuludest.

```
public class CostAndRevenueReportRowDto {
   private Long sim;
   private String company;
   private String accountManagerName;
   private PriceType priceType;
   private PriceSubtype priceSubtype;
   private Provider provider;
   private String country;
   private String network;
   private String packageName;
   private BigDecimal buyIn;
   private BigDecimal revenue;
   private BigDecimal profit;
   private YearMonth period;
   private String source;
}
```

Joonis 6. Power BI *Cost and revenue* andmestiku struktuur.

Kuna andmete saatmiseks Power BI teenusesse kasutatakse JSON andmestruktuuri, siis kõik Java andmetüübid peale BigDecimal tüübi muudetakse tekstiks. Andmete vastuvõtmisel oskab Power BI teksti kujul olevad kuupäevad ise teisendada kuupäeva andmetüüpideks. Muud väljad teisendatakse arvudeks või tekstiks vastavalt sellele, mis kujul nad JSON andmestruktuuris on defineeritud.

Power BI on seadnud oma voogedastuse andmestikele ka teatud limiidid, millega tuleb andmete edastamisel arvestada. Microsofti kodulehel on välja toodud järgnevad piirangud, mis on andmehaldus mikroteenuse kontekstis olulised [26].

- Maksimaalselt 10000 rida ühes HTTP päringus
- Maksimaalselt 1000000 rida HTTP päringutega tunnis
- Maksimaalselt 120 HTTP päringut minutis
- Kui tabelis on üle 250000 rea, siis maksimaalselt 120 HTTP päringut tunnis

- 5000000 miljonit rida maksimaalselt tabelis

Nende piirangutega arvestamiseks on andmehalduse mikroteenuses seatud maksimaalseks partii suuruseks 10000 rida. Et arvestada päringute limiidi piiranguga arvestab mikroteenus, et kui Power BI vastab veakoodiga 429, mis viitab limiidi ületamisele, siis oodatakse üks minut ja proovitakse sama päring uuesti edastada. Tabelite suuruse limiit lahendatakse manuaalselt, ehk kui tabel hakkab täis saama, edastatakse andmed lihtsalt uude tabelisse. Selleks on vaja sisestada *configuration* tabelisse uus URL-i väärtus.

7 Optimeerimine ja jõudlus

Andmehalduse mikroteenuse publitseerimiseks kasutatakse AWS EC2 teenust, mis pakub vastavalt rakenduse vajadusele eri võimsuse ja hinnastusega virtuaalseid servereid. Et tagada maksimaalne kuluefektiivsus, peab andmehalduse mikroteenus olema võimalikult efektiivselt optimeeritud. See tagab minimaalse vajaliku süsteemi ressursside tarbimise, mis omakorda võimaldab jooksutada andmehalduse mikroteenust kõige sobivamas serveris. Ehk kui mikroteenus tarbib ainult minimaalselt vajalikke ressursse ja see jookseb virtuaalses serveris, millel on olemas vajalikud ressursid ja mitte rohkem, siis on tagatud maksimaalne kuluefektiivsus.

7.1 Andmebaasi indekseerimine

SQL andmebaasi indekseerimist kasutatakse, et suurtest andmebaasidest leida üles kiiresti vajalikud andmed. Indekseerimine on üks peamine SQL andmebaasi optimeerimise tööriist ja selle kasulikkus avaldub just suurte andmehulkade puhul. Indeks on sisuliselt viide andmetele, mis aitab neid kiiresti tabelist üles leida. Indekseerida saab nii üksikut tabeli veergu kui ka mitmeid veerge korraga. Indekseerimise miinuseks on suurem kõvaketta ruumi kulu, seega ei ole neid mõistlik defineerida väikestel tabelitel [27]. Andmehalduse mikroteenuses luuakse indeks joonisel 7 välja toodud SQL lausega.

```
CREATE INDEX cdr_batch_sim_id_start_timestamp_index ON cdr_batch (start_times-
tamp DESC, sim_id);
```

Joonis 7. Andmebaasi tabeli indeksi loomine.

Indeks on lisatud *start_timestamp* ja *sim_id* veergudele, mis võimaldab neid väärtuseid kasutades leida *cdr_batch* tabelist kiiresti soovitud vasted. Indeksi lisamise vajalikkus *cdr_batch* tabelile seisnes tabeli suuruses ning selles, et vastavast tabelist on vaja tihti andmeid pärida.

7.2 Hibernate'i partiid

Hibernate'i partiide kasutamine võimaldab saata SQL lauseid grupeeritult ühe päringuga SQL serverile. Partiide kasutamine aitab optimeerida nii rakenduse võrgu kui ka mälu

kasutust. Hibernate ei kasuta vaikimisi partiisid ja saadab kõik SQL sisestus- ja uuendus-laused ükshaaval [28].

Kui tabelisse on vaja sisestada miljoneid ridu, siis ükshaaval SQL sisestuslausete saatmine tekitab pudelikaela. Selle vältimiseks ja andmebaasiga suhtluse kiirendamiseks tuleb muuta Hibernate'i vaikimisi seadeid, mille kohaselt partiide kasutamine ei ole lubatud. Selle muutmiseks tuleb muuta Spring Boot projekti *application-property.yml* failis väärtust *spring.jpa.properties.hibernate.jdbc.batch_size*. See väärtus on andmehalduse mikroteenuse projektis 5000 ehk ridu sisestatakse ja muudetakse 5000 kaupa. Lisaks Hibernate'i seadistamisele tuleb seadistada ka PostgreSQL tabelid, kus on tarvis partiisid kasutada. Vastavates tabelites tuleb jooksutada joonisel 8 väljatoodud SQL lause, mis eraldab iga SQL sisestuslause jaoks 5000 identifikaatorit.

```
ALTER SEQUENCE :tabel_id_sequence INCREMENT BY 5000;
```

Joonis 8. Andmebaasi tabeli järjendi muutmine.

Partiide kaupa andmete sisestamine on vajalik vaid tabelite puhul, kuhu salvestatakse palju andmeid korraga. Andmehalduse mikroteenuses on vaja korraga suures koguses andmeid salvestada järgnevatesse tabelitesse:

- app_consumption
- cost_per_sim
- revenue_per_sim
- unprocessed_cost_per_sim

Seega iga eelmainitud tabeli *id_sequnece* väärtus muudeti 5000 peale. Lisaks andmebaasi *id_sequence* väärtustele tuleb määrata ka Java andmebaasi klassis id veerul joonisel 9 välja toodud annotatsioonid.

```
@GeneratedValue(strategy = SEQUENCE, generator = "")
@SequenceGenerator(name = "", sequenceName = "", allocationSize = 5000)
```

Joonis 9. Andmebaasi klassi annotatsioonid.

Partiide kaupa andmete sisestamine on märkimisväärselt kiirem ühekaupa ridade sisestamisest tabelisse. Samas keskkonnas ja samade andmetega katse näitab, et 100000 rea sisestamiseks ühe kaupa kulub 64 sekundit ning 5000 kaupa 6 sekundit. Tulemust mõjutavad nii katse keskkond, andmed kui partiide suurused kuid partiide kasutamine on märkimisväärselt kiirem ühekaupa ridade sisestamisest.

7.3 Hibernate'i vahemälu

Andmebaasi objekti salvestamisel Hibernate'i transaktsioonis hoiustatakse see objekt Hibernate'i vahemälus. Loodud andmebaasi objektid hoiustatakse vahemälus seniks kuni transaktsioon sünkroniseeritakse. See juhtub tavaliselt transaktsiooni lõpus, kuid seda saab ka välja kutsuda *EntityManager.flush()* meetodiga, mis sünkroniseerib Hibernate'i vahemälu andmebaasiga. Peale sünkroniseerimist tuleb andmebaasi objektide Hibernate'i vahemälust eemaldamiseks kasutada meetodi *EntityManager.clear()*, mille tulemusel eemaldatakse objektid vahemälust ja nende poolt kasutatud mälu vabastatakse [28].

Mälu vabastamist partiide sisestamisel tuleks eelkõige kasutada kõige enam mälu nõudvamate protsessidega. Andmehalduse mikroteenuse puhul on nendeks järgnevate cron-lausete poolt käivitatavad protsessid:

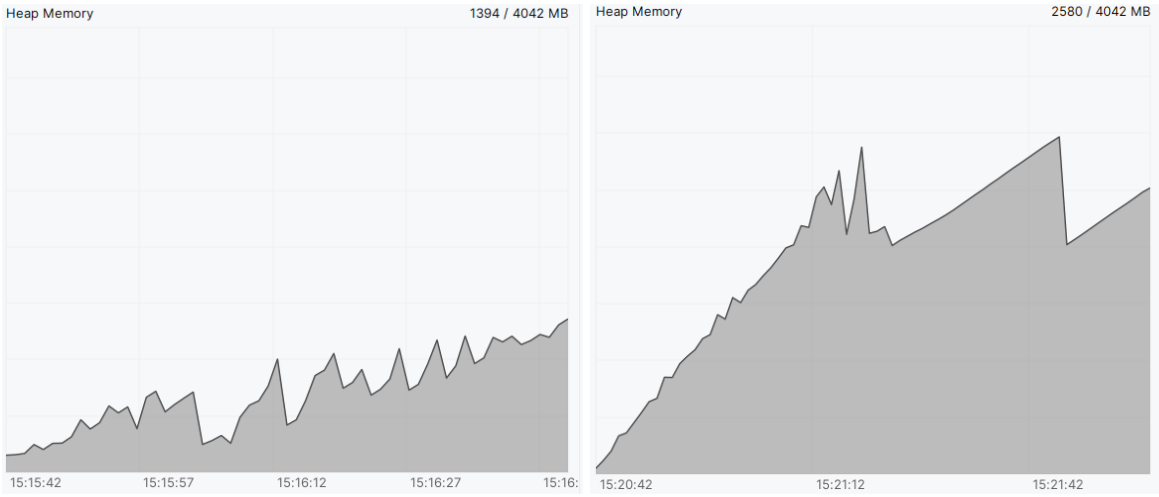
- costPerSim.reading.jt
- costPerSim.reading.tma
- costPerSim.parsing
- costPerSim.reading.kpn

Need protsessid salvestavad suures koguses andmeid andmebaasi ja teevad seda transaktsionaalselt. Seega seniks kuni transaktsionaalne loogika pole lõppenud, hoitakse kõiki loodud andmebaasi objekte mälus. Et vähendada andmehalduse mikroteenuse mälu nõudlikkust, on koodi lisatud loogika, mis iga 100000 objekti sisestamise järel sünkroniseerib ja

tühjendab Hibernate'i vahemälu. Seega ei hoita mälus korraga rohkem kui 100000 andmebaasi objekti. Mälu vabastamiseks käivitatakse iga 100000 rea järel joonisel 10 välja toodud kood.

```
entityManager.flush();
entityManager.clear();
```

Joonis 10. Hibernate'i mälu vabastus käsklused.



Joonis 11 Hibernate'i mälu kasutus.

Jooniselt 11 on näha mälu kasutus 2000000 rea sisestamisel andmebaasi 5000 rea suuruste partiide kaupa. Vasakpoolsel graafikul on Hibernate'i vahemälu iga 100000 rea sisestamise järel sünkroniseeritud ja puhastatud ning parempoolsel graafikul mitte. Andmete sisestamisel andmebaasi ilma optimeerimata kasutas Java protsess maksimaalselt 2580 MB mälu ja optimeeritult maksimaalselt 1304 MB mälu.

8 Testimine

Lisaks funktsionaalsetele nõuetele on andmehalduse mikroteenusele ka nõue, et see peab olema kaetud testidega. Koodi kaetavuse protsendiline nõue ei ole fikseeritud, kuid tuumik loogika peab olema testitud. Testide eesmärgiks on veenduda, et kood töötab korrektselt ja ka garanteerida, et tulevikus tehtavad muudatused ei lõhuks ära juba olemasolevat funktsionaalsust. Viimane punkt on eriti oluline, kuna 1oT arendustiimis tegelevad arendajad kõikide mikroteenustega ning tulevikus võib andmehalduse mikroteenuse koodi muuta keegi, kellel puudub sellega varasem kogemus. Loodavad testid tagavad kindluse, et andmehalduse mikroteenuse arendusega saab tulevikus tegeleda ükskõik kes arendustiimist vastavalt vajadusele.

Overall Coverage Summary

Package	Class, %	Method, %	Line, %
all classes	55.2% (95/172)	38.2% (282/738)	36.3% (532/1465)

Coverage Breakdown

Package	Class, %	Method, %	Line, %
com.oneot.reports.service.reading	100% (3/3)	35% (7/20)	10.7% (9/84)
com.oneot.reports.service.pushing	100% (4/4)	47.1% (8/17)	25% (8/32)
com.oneot.reports.service.model	100% (3/3)	57.1% (4/7)	70% (7/10)
com.oneot.reports.redis.repository	100% (4/4)	50% (4/8)	50% (4/8)
com.oneot.reports.integration.powerbi	100% (1/1)	50% (3/6)	35% (7/20)
com.oneot.reports.integration.backend	100% (1/1)	33.3% (1/3)	20% (1/5)
com.oneot.reports.file.file	100% (3/3)	65.2% (15/23)	72.7% (40/55)
com.oneot.reports.event.handler	100% (2/2)	40% (2/5)	28.6% (2/7)
com.oneot.reports.db.model.enums	100% (4/4)	100% (4/4)	100% (31/31)
com.oneot.reports.db.converter	100% (1/1)	33.3% (1/3)	33.3% (1/3)
com.oneot.reports.cache	100% (5/5)	36.4% (16/44)	34.1% (44/129)
com.oneot.reports.service.parsing	88.9% (8/9)	83.1% (64/77)	65% (134/206)
com.oneot.reports.conf	86.7% (13/15)	86.2% (50/58)	66.3% (124/187)
com.oneot.reports.service	77.8% (7/9)	32.1% (17/53)	14.4% (26/180)
com.oneot.reports.file	60% (3/5)	58.6% (17/29)	55.6% (20/36)
com.oneot.reports.redis.model	50% (4/8)	16.7% (8/48)	13.6% (8/59)
com.oneot.reports	50% (1/2)	20% (1/5)	20% (1/5)
com.oneot.reports.db.model	44.1% (26/59)	29.7% (58/195)	25.2% (63/250)
com.oneot.reports.cron	33.3% (2/6)	14.3% (2/14)	12.5% (2/16)
com.oneot.reports.service.exception	0% (0/5)	0% (0/10)	0% (0/10)
com.oneot.reports.integration.powerbi.model	0% (0/9)	0% (0/50)	0% (0/50)
com.oneot.reports.integration.backend.model	0% (0/2)	0% (0/5)	0% (0/5)
com.oneot.reports.file.fileline	0% (0/1)	0% (0/9)	0% (0/15)
com.oneot.reports.event.model	0% (0/11)	0% (0/45)	0% (0/62)

Joonis 12. Andmehalduse mikroteenuse testide raport.

Jooniselt 12 on näha IntelliJ poolt genereeritud testide raport, mis näitab koodibaasi testidega kaetavust. Kõikidest meetoditest 38.2 protsenti ja kõikidest koodiridadest 36.3 on testidega kaetud. Lisaks on eelmainitud jooniselt näha ka täpsemad tulemused projekti kaustade kaupa.

Andmehalduse mikroteenus kasutab koodibaasi testimiseks kahte tüüpi teste. Funktsionaalsuste testimiseks konteksti välistelt kasutatakse üksuste testimise põhimõtet ja laiema funktsionaalsuse testimiseks kasutatakse integratsiooni testimise põhimõtet.

Testimiseks kasutatakse peamiselt JUnit raamistikku. Integratsiooni testideks kasutatakse Spring Booti integratsioonitesti spetsiifilist funktsionaalsust. Lisaks kasutatakse testides, veendumaks, et oodatud tulemus vastab reaalsele tulemusele, Hamcrest raamistikku. Kuna andmehalduse mikroteenus suhtleb väliste rakendutega, tuleb testimisel kasutada ka Mockito raamistikku, mis võimaldab jäljendada meetodite tööd ilma, et tegelikult väliste rakendustega suheldaks.

9 Publitseerimine

Viimane samm andmehalduse mikroteenuse tööks valmis saamise puhul on selle publitseerimine. Andmehalduse mikroteenuse publitseerimiseks kasutatakse AWS EC2 *t2.medium* virtuaalset serverit. Tegu on Ubuntu serveriga, mis kasutab LTS versiooni 20.04.3. Serveril on 4 gibibaiti mälu ja kaks vCPU-d. Selles serveris on vaja jooksutada vaid mikroteenust ennast, kuna selle poolt kasutatav PostgreSQL andmebaas seatakse üles 1oT eraldiseisvasse andmebaasiserverisse.

Andmehalduse mikroteenuse jooksutamiseks tuleb luua serveris mõned eeldused. Esiteks peab serveris olema aktiveeritud Dockeri teenus. Lisaks tuleb serveris defineerida *docker-compose.yml* fail, mida on tarvis Dockeri konteineri loomiseks. Joonisel 13 on näha defineeritud *docker-compse.yml* faili. Selles failis on defineeritud kõik vajalikud andmed konteineri jooksutamiseks.

```
version: "3.7"
services:
  reports:
    image: registry.gitlab.com/1ot/terminal/reports/reports:${versioon}
    ports:
      - "8108:8108"
    env_file:
      - environment.env
    environment:
      - JAVA_OPTS=-Xms256m -Xmx2500m
    stop_grace_period: 30s
    logging:
      driver: "json-file"
      options:
        max-size: "10m"
        max-file: "10"
    deploy:
      resources:
        limits:
          memory: 3500M
          cpus: '1.5'
      replicas: 1
      update_config:
        order: stop-first
```

Joonis 13. *docker-compose.yml* fail.

Jooniselt 13 on lisaks näha, et Dockeri konteiner luuakse Dockeri pildi järgi, mis asub URL-il *registry.gitlab.com/1ot/terminal/reports/reports:${version}*. Sellel URL-il asub GitLabi poolt valmis ehitatud andmehaldus mikroteenuse *jar* faili sisaldav Dockeri pilt.

Et GitLab ehitaks vajaliku Dockeri pildi valmis, tuleb kasutada GitLabi CI/CD funktsionaalsust. Selleks tuleb defineerida koodibaasis *.gitlab-ci.yml* fail, mille põhjal oskab GitLab jooksutada defineeritud käskluseid kindlate sündmuste peale. Andmehalduse mikroteenuse *.gitlab-ci.yml* failis on lisaks teistele käsklustele defineeritud, et koodi lisamisel *Master* harusse ehitatakse valmis Dockeri pilt, mis sisaldab andmehalduse mikroteenuse *jar* faili.

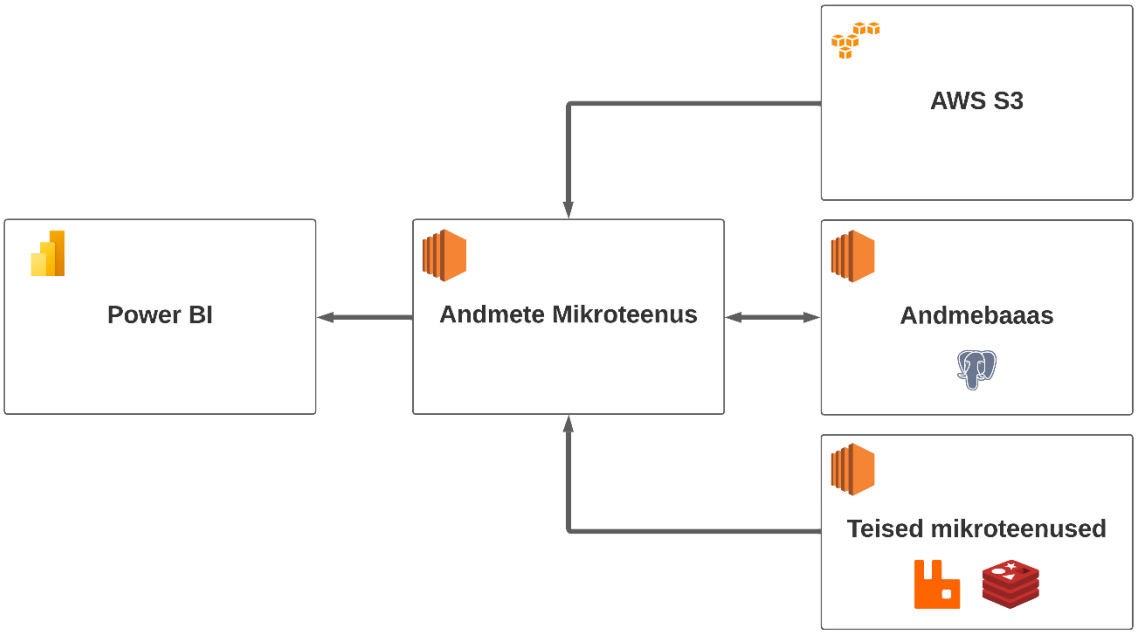
Kui serveris ja GitLabis on vajalikud sammud tehtud, saab serveris andmehalduse mikroteenuse käivitada. Selleks tuleks veenduda kõigepealt, et *docker-compose.yml* failis olev Dockeri pildi viide viitaks viimasele versioonile, mis on GitLabis olemas ning seejärel jooksutada serveris joonisel 14 välja toodud käsklust.

```
docker stack deploy -c docker-compose.yml -with-registry-auth andmete_mikrotee-
nus
```

Joonis 14. Dockeri konteineri jooksutamiskäsklus.

Peale selle käsu jooksutamist peaks andmehalduse mikroteenus serveris töötama. Selle kontrollimiseks võib jooksutada käsklust *docker ps -a,* mis näitab kõiki serveris jooksvaid Dockeri konteinereid ja nende staatust.

Edaspidi andmehalduse mikroteenuse uuendamiseks tuleb koodibaas lükata GitLabi *Master* harusse, mille järel ehitab GitLab automaatselt valmis uue Dockeri pildi. Seejärel tuleb *docker-compse.yml* failis uuendada Dockeri pildi versioon ja jooksutada uuesti käsklust, millega algne konteiner valmis ehitati.



Joonis 15. Andmehalduse mikroteenuse suhtlus teiste rakendustega.

Joonisel 15 on välja toodud andmehalduse mikroteenuse suhtlus teiste teenustega ning kuidas andmed nende vahel liiguvad. Nagu ka eelnevalt välja toodud, pärinevad sisendandmed nii AWS S3 teenusest kui ka teistelt mikroteenustelt ning lõpuks jõuavad andmed Power BI teenusesse.

10 Tagasiside

Andmehalduse mikroteenus oli 1oT esimene katse andmetest parema ülevaate saamiseks ning see täitis sellele seatud eesmärki. See tähendab, et andmed jõuavad ettenähtud kujul teenusesse Power BI, kus neid saab põhjalikumalt analüüsida.

Selle aja jooksul, kui andmehalduse mikroteenus on jõudnud töötada, on välja tulnud ka mõned puudused. Ühe puudusena on välja toodud, Power BI voogedastus andmestiku maksimaalne tabeli suurus. Kuna eelmainitud tabel lubab maksimaalselt 5 miljonit sissekannet, siis tuleb tabeli täitumisel andmed suunata uude voogedastus andmestikku. See eeldab manuaalset sekkumist ning ei ole ka võimallik näidata kõikide olemasolevate perioodide andmeid ühel graafikul.

Teine teadaolev puudus on seotud sisendfailide haldamisega. Praegu eeldab ka see protsess manuaalset sekkumist, kuna pole automaatset protsessi, kuidas partnerite CDR failid jõuaksid AWS S3 teenusesse, kust saab andmehalduse mikroteenus juba nendega automaatselt edasi tegeleda.

11 Kokkuvõte

Bakalaureusetöö eesmärgiks oli luua ettevõttele 1oT süsteem, mis suudaks koondada ettevõtte jaoks olulised andmed, viia need nõutud kujule ja saata need edasi teenusesse Power BI. Olulised andmekategooriad, millega loodud teenus tegeleb, on SIM-kaartide müümine, andmetarbimine ja 1oT lisateenuste müümine. Need on kolm ettevõtte jaoks olulist valdkonda, millest eelnevalt puudus selge ülevaade.

Bakalaureusetöö raames valmis andmehalduse mikroteenus, mis on eraldiseisev mikroteenus, mis suhtleb teiste 1oT mikroteenustega. Andmehalduse mikroteenus koondab nõutud andmed kokku erinevatest allikatest, viib need eeldatud kujule ja edasta teenusesse Power BI. Loodud mikroteenus on operatiivne ja see publitseeriti selle jaoks loodud spetsiaalsesse serverisse.

Kirjalikus töös seletati lahti kasutatud mõisted, kirjeldati erinevaid andmekategooriad, millega mikroteenus tegeleb ja toodi välja tehnilised nõuded loodud mikroteenusele. Lisaks kirjeldab töö põhjalikult loodud mikroteenuse arhitektuuri ning annab ülevaate kasutatud tehnoloogiatest. Viimased peatükid annavad lisaks ülevaate mikroteenuse optimeerimisest, selle testimisest ja ka selle publitseerimisest

Andmehalduse mikroteenus töötab ning tegeleb iga päev taustal andmete kogumisega ja nende andmete töötlemisega. Edasiarendused ja tulevikuplaanid seoses andmehalduse mikroteenusega on ebaselged ja sõltuvad konkreetsetest vajadustest, mis võivad tulevikus ette tulla. Andmehalduse mikroteenuse lähtekood on koodihoidla õiguste olemasolul kättesaadav lingilt https://gitlab.com/1oT/terminal/reports.

12 Viidatud kirjandus

[1] 1oT platvormi litsentseerimine. https://1ot.mobi/licensing/terminal (01.03.2023)

[2] 1oT ülevaade. https://1ot.mobi/company/about (01.03.2023)

[3] Mis on CDR. https://www.ghtinc.com/what-is-cdr-and-how-does-it-can-help-to-win-the-battle-against-covid-19/ (07.03.2023)

[4] Cron-päästiku õpetus. http://www.quartz-scheduler.org/documentation/quartz-2.3.0/tutorials/crontrigger.html#format (12.03.2023)

[5] Mis on ICCID number? https://www.emnify.com/iot-glossary/iccid-number (22.02.2023)

[6] Mis on rahvusvaheline võrgutarbija identiteet (IMSI)? https://www.emnify.com/iot-glossary/imsi (13.03.2023)

[7] Juurdepääsutasu. https://www.cybertelecom.org/ci/access.htm (07.03.2023)

[8] Lühidalt mikroteenustest. https://www.thoughtworks.com/insights/blog/microservices-nutshell (01.03.2023)

[9] MNO. https://ec.europa.eu/eurostat/cros/content/Glossary:Mobile_network_operator_(MNO)#:~:text=A%20mobile%20network%20operator%2C%20abbreviated,for%20its%20subscribed%20mobile%20users. (22.03.2023)

[10] Mis on MSISDN? https://www.wirelesslogic.com/iot-glossary/what-is-msisdn/ (07.03.2023)

[11] GPRS võrgu identifikaatorid. https://mobilepacketcore.com/plmn-lac-rac-gprs/ (26.04.2023)

[12] Mis on SIM-kaart ja kuidas see töötab? https://builtin.com/hardware/what-is-a-sim-card (22.03.2023)

[13] Mis on SMS MO ja MT? https://thesmsworks.co.uk/blog/mo-and-mt-sms/ (23.03.2023)

[14] Terminite sõnastik. https://docs.monogoto.io/tips-and-tutorials/glossary-of-terms (22.03.2023)

[15] Võrk. https://www.techopedia.com/definition/5537/network#:~:text=A%20network%2C%20in%20computing%2C%20is,components%2C%20and%20they%20are%20connected. (26.04.2023)

[16] Mis on Spring Boot? https://www.educative.io/answers/what-is-spring-boot (01.03.2023)

[17] Mis on Gradle? https://www.simplilearn.com/tutorials/gradle-tutorial/what-is-gradle (01.03.2023)

[18] Mis on Java? https://www.java.com/en/download/help/whatis_java.html (01.03.2023)

[19] Mis on Java? https://aws.amazon.com/what-is/java/ (01.03.2023)

[20] Mis on PostgreSQL? https://www.ibm.com/topics/postgresql (01.03.2023)

[21] Mis on Hibernate Javas ja miks seda vaja on? https://www.edureka.co/blog/what-is-hibernate-in-java/ (05.03.2023)

[22] FlyWay ja versioonipõhise andmebaasi migratsiooniga alustamine. https://thorben-janssen.com/flyway-getting-started/ (05.03.2023)

[23] Redise vahemälu: kuidas see töötab ja miks seda kasutada. https://linuxiac.com/redis-as-cache/ (01.03.2023)

[24] RabbitMQ järjekordade ja sõnumite saatmise ABC. https://hevodata.com/learn/rabbitmq-queue/ (01.03.2023)

[25] Mis on Docker? https://www.ibm.com/topics/docker (23.02.2023)

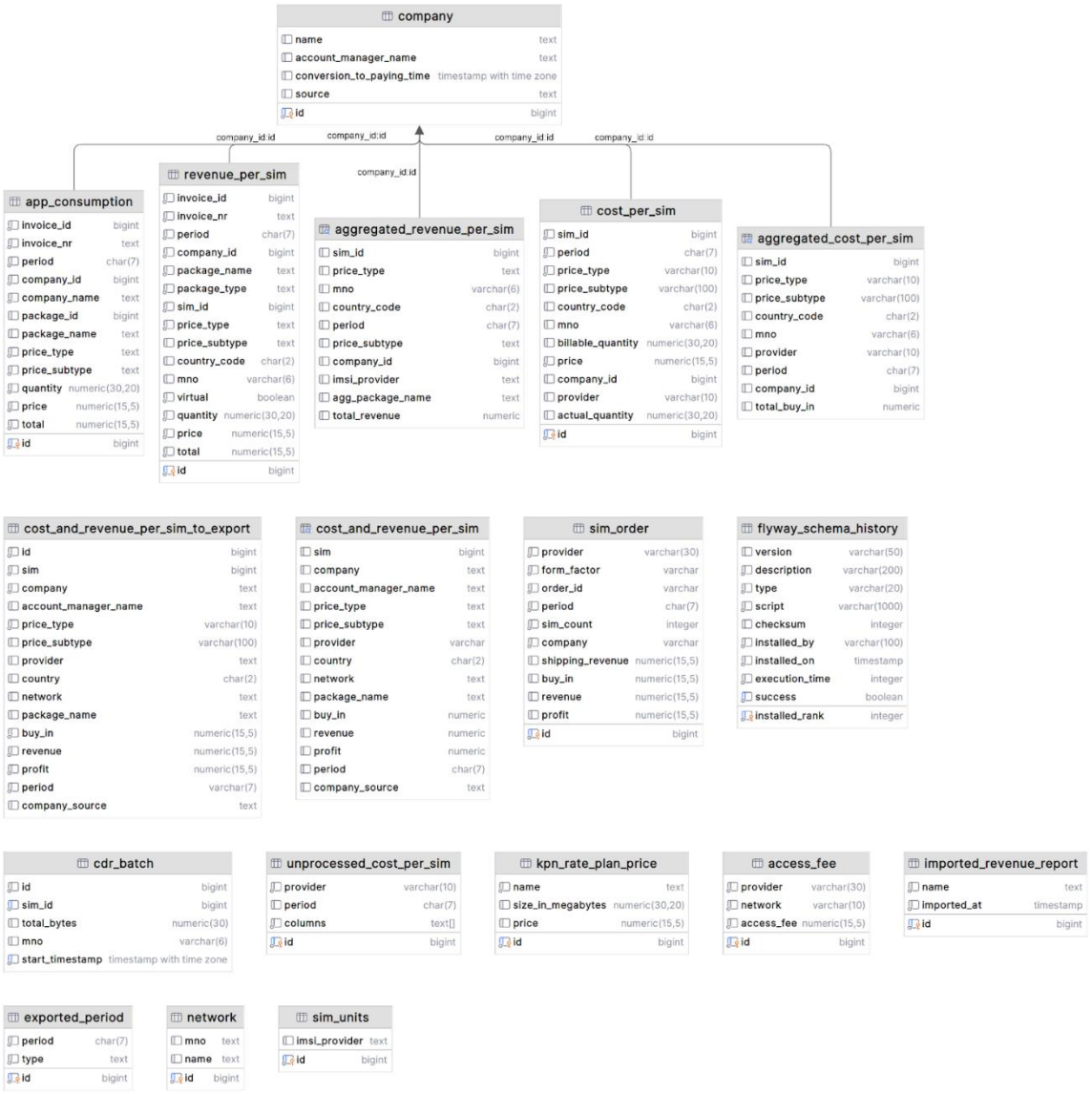
[26] Voogedastus andmetike piirangud. https://learn.microsoft.com/en-us/power-bi/developer/embedded/push-datasets-limitations (02.04.2023)

[27] SQL indeks. https://www.simplilearn.com/tutorials/sql-tutorial/index-in-sql (22.04.2023)

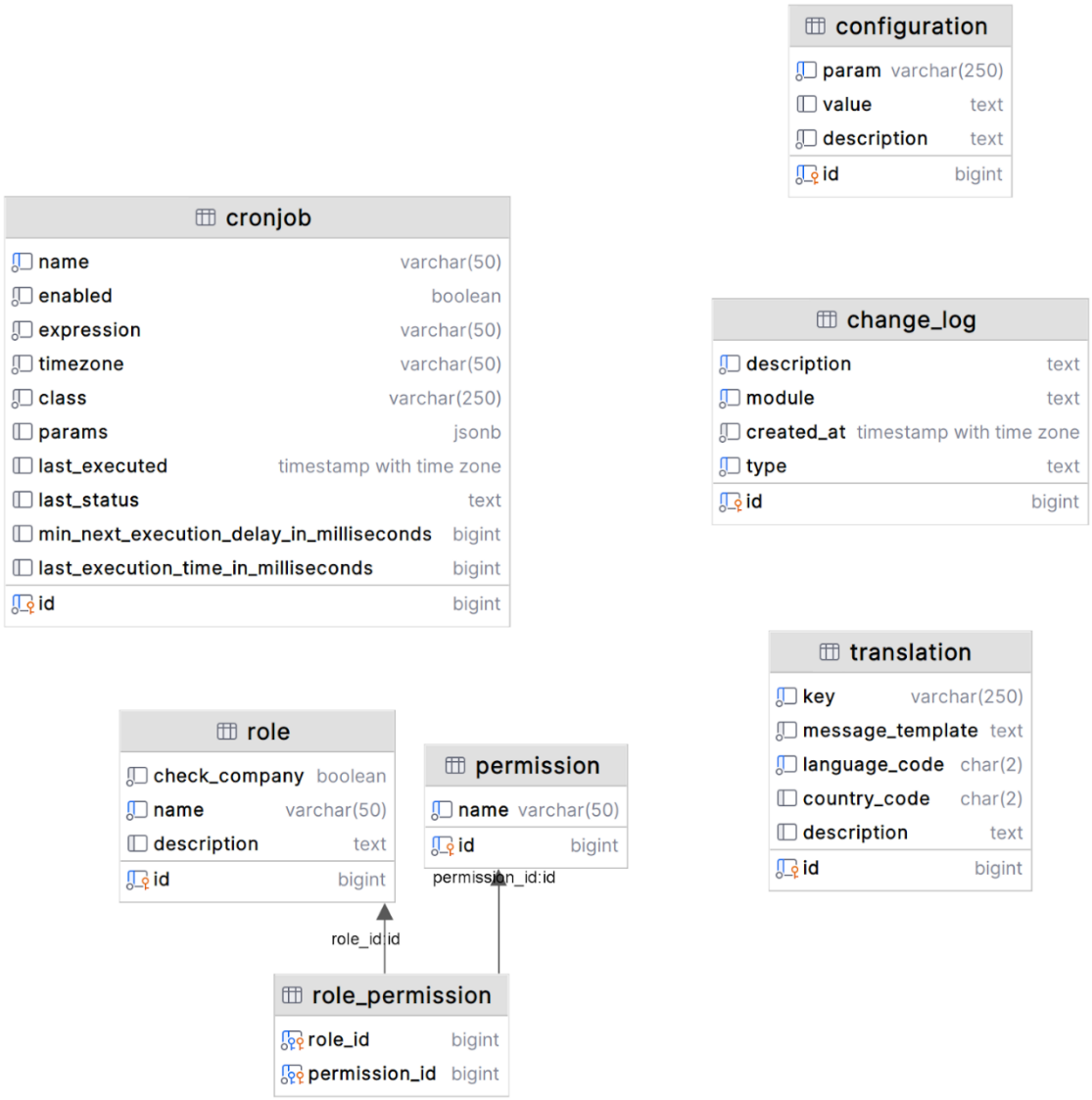
[28] Hibernate'i partiid. https://www.baeldung.com/jpa-hibernate-batch-insert-update (22.04.2023)

13 Lisad

1. Andmehaldus mikroteenuse andmebaasi tabelid



2. *Common* teekide andmebaasi tabelid



3. TMA faili näidisandmed

```
# Report ./cxn_generic_call_by_bill_no.sql produced at Fri Apr 01 07:46:53 UTC
2022 with parameters:
# BILL NO: B1-526523   ACCOUNT NO: 23000089  CALL_TYPE: usage

"CALLTYPE","CONTRACT ID","SUBSCRIPTION TYPE","CALL TIME","UTC OFFSET","A-NUM-
BER","B-NUMBER","DIRECTION","ZONE","APN","ORIGINATING COUNTRY","ORIGINATING NET-
WORK","DESTINATION COUNTRY","DESTINATION NETWORK","IMSI","CALL SET-UP
FEE","UNIT","UNIT PRICE","BILLABLE QUANTITY","ACTUAL QUANTITY","AMOUNT","VOLUME
SENT","VOLUME RECEIVED"
"gprs"," SCRAMBLED_CONTRACT_ID","PP12345445","2022-03-01
01:24:07","+0000","00436761721212218","","MOC","In Group J","test.gma.iot","Mex-
ico","net-
work1","","","232031221207519","0","1048576",".015","1024","899",".000014648","4
97","402"
"gprs"," SCRAMBLED_CONTRACT_ID","PP12345445","2022-03-01
01:24:38","+0000","00436761722823459","","MOC","In Group J","test.gma.iot","Mex-
ico","net-
work1","","","232032321565253","0","1048576",".015","1024","550",".000014648","4
22","128"
"gprs"," SCRAMBLED_CONTRACT_ID","PP12345445","2022-03-01
01:19:41","+0000","00436761721334111","","MOC","In Group J","test.gma.iot","Mex-
ico","net-
work2","","","232031723480688","0","1048576",".015","3072","2629",".000043945","
1819","810"
"gprs"," SCRAMBLED_CONTRACT_ID","PP12345445","2022-03-01
01:22:46","+0000","00436761722445256","","MOC","In Group J","test.gma.iot","Mex-
ico","net-
work2","","","232031724534813","0","1048576",".015","5120","5036",".000073242","
2550","2486"
"gprs"," SCRAMBLED_CONTRACT_ID","PP12345445","2022-03-01
01:01:11","+0000","00436761722456474","","MOC","In Group J","test.gma.iot","Mex-
ico","net-
work1","","","232031562004228","0","1048576",".015","2048","1560",".000029297","
900","660"
"gprs","SCRAMBLED_CONTRACT_ID","PP12345445","2022-03-01
01:23:40","+0000","00436761722467297","","MOC","In Group J","test.gma.iot","Mex-
ico","net-
work1","","","232031767034854","0","1048576",".015","1024","816",".000014648","4
06","410"
```

4. JT faili näidisandmed

```
IMSI,ICCID,Product,Start Date,End Date,Quantity,Cost,Use Case,Custom1,Cus-
tom2,Custom3,Custom4,Custom5,Custom6,Custom7,Custom8,Custom9
,,,,,,,,,,,,,,,
237100003038004,8937803011170380042,"TEST PRODUCT
1",50:04.0,48:06.0,9.067,181.34,,,,,,,,,
237100003038004,8937803011170380042,"TEST PRODUCT
2",37:47.0,42:33.0,10.325,123.9,,,,,,,,,
237100003038004,8937803011170380042,Trigger Mrc,00:00.0,00:00.0,1,140,,,,,,,,,
237100003092004,8944503016770380141,"TEST PRODUCT
1",10:18.0,44:51.0,1.218,24.36,,,,,,,,,
237100003092004,8944503016770380141,Trigger Mrc,00:00.0,00:00.0,1,140,,,,,,,,,
414500003038043,8944503011170380254,"TEST PRODUCT
1",01:55.0,48:17.0,0.616,12.32,,,,,,,,,
414500003038043,8944503011170380254,"TEST PRODUCT
2",37:56.0,55:25.0,13.804,165.648,,,,,,,,,
414500003038043,8944503011170380254,Trigger Mrc,00:00.0,00:00.0,1,140,,,,,,,,,
234512003038034,7944443011170380312,"TEST PRODUCT
1",42:53.0,25:12.0,13.833,276.66,,,,,,,,,
234512003038034,7944443011170380312,Trigger Mrc,00:00.0,00:00.0,1,140,,,,,,,,,
234512003038034,7944443011170380312,Trigger Mrc,39:34.0,00:00.0,0.056,-
7.84,,,,,,,,,
8345000563038046,8914503012170380462,"TEST PRODUCT
1",44:03.0,34:08.0,0.553,11.06,,,,,,,,,
8345000563038046,8914503012170380462,"TEST PRODUCT
2",25:15.0,56:44.0,0.76,9.12,,,,,,,,,
```

5. KNP faili näidisandmed

```
ICCID,Device ID,Customer,Monthly Rate Plan,Prepaid Terms Charged,Standard Rate
Plan,Subscriber Status,Subscription Charge,Fixed Pool Charge,Data Volume
(MB),SMS Volume (msg),SMS MO Volume (msg),SMS MT Volume (msg),Voice Volume
(m:ss),Voice MO Volume (m:ss),Voice MT Volume (m:ss)
8912012420239073476,,,TEST SERVICE 1,,TEST SERVICE 6,Ac-
tive,0.35,0,0,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 2,,TEST SERVICE 5,1,0,0,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 2,,TEST SERVICE 5,1,0,0,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 2,,TEST SERVICE 5,1,0,0,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 2,,TEST SERVICE 5,1,0,0,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 3,,TEST SERVICE 3,Ac-
tive,0.28,0,17.93,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 4,,TEST SERVICE 4,Ac-
tive,0.2,0,0.024,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 4,,TEST SERVICE 4,Ac-
tive,0.2,0,5.141,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 4,,TEST SERVICE 4,Ac-
tive,0.2,0,0.466,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 4,,TEST SERVICE 4,Ac-
tive,0.2,0,0,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 3,,TEST SERVICE 3,Ac-
tive,0.28,0,1.866,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 4,,TEST SERVICE 4,Ac-
tive,0.2,0,2.678,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 4,,TEST SERVICE 4,Ac-
tive,0.2,0,3.649,0,0,0,0:00,0:00,0:00
8912012420239073476,,,TEST SERVICE 3,,TEST SERVICE 3,Ac-
tive,0.28,0,0,0,0,0,0:00,0:00,0:00
```

6. 1oT Terminali lisateenuste näidisandmed

```
Invoice ID,Invoice Nr,Period,Company ID,Company name,Package ID,Package
name,Price type,Price sub type,Quantity,Price,Total
6712161,1oT/2022-03/00009,2022-03,7850,Company6,,,APP,PRODUCTIVITY_TOOL-
KIT,332,0.4,132.8
14046892,1oT/2022-03/00003,2022-03,42486,Company9,,,APP,NOTIFICA-
TIONS,338,0.5,169
79250956,1oT/2022-03/00005,2022-03,41481,Company9,,,APP,LOCALAPI,524,0.6,314.4
40885679,1oT/2022-03/00006,2022-03,92403,Company6,,,APP,KPI,14,0.7,9.8
76286389,1oT/2022-03/00002,2022-03,8123,Company7,,,APP,ANALYTICS,427,0.8,341.6
45827756,1oT/2022-03/00002,2022-03,40034,Company6,,,APP,DIAGNOS-
TICS,976,0.9,878.4
61864146,1oT/2022-03/00002,2022-03,39183,Company7,,,APP,FU-
TURE_EVENTS,149,1.1,163.9
2340310,1oT/2022-03/00006,2022-03,3793,Company4,,,APP,SLACK,551,1,551
178514,1oT/2022-03/00004,2022-03,95110,Company6,,,APP,ANALYTICS,768,0.8,614.4
9767328,1oT/2022-03/00005,2022-03,37201,Company5,,,APP,NOTIFICATIONS,946,0.5,473
```

7. Tulude faili näidisandmed

```
Invoice ID,Invoice Nr,Period,Company ID,Package name,Package type,Sim ID,Price
type,Price sub type,Country code,MNO,Virtual,Quantity,Price,Total
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1791047,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1791045,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1791043,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1791041,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1791039,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1791037,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1790735,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1790733,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1790731,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1790729,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1790727,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1790725,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1790723,SIM,STATUS_LIVE,,,false,1,5.75,5.75
17636359,1oT/2021-12/00001,2021-12,75045,Test Data package,IN-
CLUDED,1790721,SIM,STATUS_LIVE,,,false,1,5.75,5.75
```

8. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

1. Mina, Markus Leemet, annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose „Mikroteenuse loomine ettevõtte 1oT andmete haldamiseks“, mille juhendaja on Helle Hein, reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.
2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.
3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.
4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Markus Leemet

06.05.2023