TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Pille Ulmas 176248IDDR

PSD2 API'de erisuste lahendamine kontoteabe teenuse väljatöötamisel

Diplomitöö

Juhendaja: Andres Käver
BSc

Tallinn 2021

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Pille Ulmas

17.05.2021

Annotatsioon

Käesoleva bakalaureusetöö eesmärk on luua võimalikult paindliku arhitektuuriga kontoteabe teenuse prototüüp, mis võimaldaks minimaalse vaevaga uute pankade API'de integreerimist.

Töös kirjeldab autor erinevate API'de dokumentatsioonide põhjal tuvastatud erisusi, seletab olulisemaid prototüübi valmistamises kasutatud tehnilisi vahendeid ja võtteid ning näitab loodud funktsionaalsusi.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 50 leheküljel, 5 peatükki, 17 joonist, 1 tabelit.

Abstract

Developing an Account Information Service – Resolving PSD2 API Differences

The intent of this bachelor's thesis is to create a prototype of an Account Information Service with the most flexible architecture possible, which would enable the integration of APIs of new banks with minimal effort.

In the document, the author describes the differences identified based on the documentation of different APIs, explains the most important technical tools and techniques used in the production of the prototype, and shows the created functionalities.

The thesis is in Estonian and contains 50 pages of text, 5 chapters, 17 figures, 1 table.

Lühendite ja mõistete sõnastik

AISP	*Account Information Service Providers* - teenusepakkujatel on õigus vaadata kliendi maksekonto teavet, kui klient (PSU) on selleks nõusoleku andnud. Enamasti pakuvad AISP-d koondvaate klientide arvepidamistest arvukates pankades koos nende tehingute üksikasjadega.
AIS	*Account Information Service* – Kontoteabe teenus
API	*Application program Interface* – Rakendustarkvara liides
ASPSP	*Account Service and Payment Service Provider* – konto- ja makseteenuse pakkuja on mis tahes finantsasutus, mis pakub veebipõhise juurdepääsuga maksekontot. Enamasti on need pangad ja muud finantsasutused. PSD2 kohustab ASPSP-sid API-de kaudu andma TPP-dele juurdepääsu oma klientide maksekontode kontode ja tehingute andmetele.
Berlin Group	Berliini grupp on 2004 aastal asutatud panganduse standartide loomise ühendus, mis koosneb peaaegu neljakümnest pangast, ühendusest ja makseteenuse pakkujast üle kogu Euroopa Liidu.
EBA	*European Banking Authority* – Euroopa pangandusjärelvalve
eIDAS	*electronic IDentification, Authentication and trust Services* – Euroopa Parlamendi ja nõukogu määrus "e-identimise ja e-tehingute jaoks vajalike usaldusteenuste kohta siseturul ja millega tunnistatakse kehtetuks direktiiv 1999/93/EÜ"
GDPR	*General Data Protection Regulation* – Isikuandmete kaitse üldmäärus
JWK	*JSON (JavaScript Object Notation) Web Key* – andmestruktuur, mis esindab krüptograafilist võtit.
Nõusolek	Nõusolek on peamine osa PSD2 reguleerimisest ja koostööst kolmandate osapoolte pakkujatega. Ainus viis, kuidas TPP-d PSU nimel tegutseda saavad, on see, kui klient on andnud selgesõnalise nõusoleku selliste õiguste kasutamiseks. Teisisõnu, nõusoleku puudumine tähendab volituse puudumist.
OAuth 2.0	OAuth 2.0 on valdkonna standardne autoriseerimisprotokoll. OAuth 2.0 asendab 2006. aastal loodud algse OAuthi protokolli. OAuth 2.0 keskendub kliendi arendajate lihtsusele, pakkudes samas veebirakenduste, töölauarakenduste, mobiiltelefonide ja IoT-seadmete jaoks konkreetseid autoriseerimisvooge. See võimaldab kolmandate osapoolte rakendustel saada piiratud

	juurdepääsu veebiteenusele.
PSD2	*Revised Payment Services Directive* – muudetud Makseteenuste Direktiiv
PSU	*Payment Service User* – Kasutaja, kes annab nõusoleku TPP-le oma kontodele ligipääsu lubamiseks. PSU võib olla nii eraisik kui ka äriisik.
QSEAL	*Qualified Certificate for Seals* – kvalifitseeritud allkirjastamise sertifikaat. Digitaalne turvalist teadete allkirjastamist võimaldav sertifikaat, mis vastab eIDAS regulatsioonidele.
QWAC	*Qualified Website Authentication Certificate* – kvalifitseeritud veebilehe autentimise sertifikaat. Digitaalne turvalist andmevahetust võimaldav sertifikaat, mis on eIDAS regulatsiooniga defineeritud
RTS	*Regulation Technical standards* - Euroopa Komisjon võttis 2017. aasta novembris vastu tehnilised standardid, milles esitatakse erinõuded, et tagada klientide tugev autentimine ja muud turvameetmed, mis peavad selliste tehingute jaoks kehtima. Dokumendis on välja toodud protokollid, mida tuleb rakendada klienditeabe turvalisuse ja konfidentsiaalsuse kaitsmiseks ning turvalise ja avatud suhtluse tagamiseks kogu makseprotsessi vältel.
SCA	*Strong Customer Authentication* - Tugev kliendi autentimine, nagu on määratletud EBA regulatiivsetes tehnilistes standardites, on autentimine, mis põhineb kahe või enama elemendi kasutamisel, mis liigitatakse teadmisteks (mida ainult kasutaja teab [näiteks parool]), valduseks (midagi, mis on ainult kasutajal [näiteks konkreetne mobiiltelefon ja telefoninumber]) ja pärilikkus (midagi, mida kasutaja on [või mis tal on, näiteks sõrmejälg või iirise muster]), mis on sõltumatud, nii et ühe rikkumine ei ohusta muid ja on loodud selliselt, et kaitsta autentimisandmete konfidentsiaalsust.
TLS	*Transport Layer Security* – Transpordikihi turbeprotokoll on krüptograafiline protokoll, mis turvab võrgusuhtlust. TLS krüpteerib võrguühenduste segmente transpordikihis suhtluskanali otspunktides.
Token	Autoriseerimise käigus loodud sõne, mida kasutatakse päringute tegemisel andmetele ligipääsu õiguse kinnitamiseks
TPP	*Third Party Payment Service Providers* – täidab kliendi (PSU) nimel määratletud teenuseid. TPP võib API-liidese kaudu ligi pääseda teiste pankade (ASPSP) poolt hallatavatele PSU konto(de)le.
URL	*Uniform Resource Locator* - veebiaadress

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Koos tehnoloogia ja majanduse arenguga on arenenud ka inimeste vabadus valida ja nende sõltumatus teenusepakkujatest, kuna valik on laiem kui kunagi varem. Järjest enam soovitakse otsuseid teha ennast ühegi konkreetse teenusepakkujaga sidumata. Inimesed on teadlikumad, teevad uurimusi ja valivad omale sobivaima teenuse vastavalt pakkumise sobivusele, mitte enam niivõrd teenusepakkujate vähesusele toetudes.

Ligipääs oma andmetele, nii finantsilistele kui ka muudele, nendega manipuleerimine ja oma soovi järgi kasutamine on saanud uueks normiks. Sellise avanemise ja valikuvabadusega on ühiskonna survel kaasa läinud ka varasemalt väga konservatiivne olnud pangandus. Enam ei ole kliendid kinni eluaegselt ühes ja ainukeses pangas ja tihtipeale opereeritaksegi mitmes erinevas finantsteenuseid osutavas asutuses samaaegselt – näiteks on ühest saadud parimate tingimustega laen, teises paremad võimalused investeerimisteks ning kolmandas mõistlik teha igapäevaseid makseid või kasutada krediitkaarti, mis on just kõige sobivam ja vajadustele vastavaim. Kaubanduskeskustes pensioni fondide pakkujaid on kohanud ilmselt kõik ja võibolla mõned on ka kenasti esitatud jutu ära kuulanud ning fondi üle viimise kaasa teinud.

Globaliseerumise ja avatusega kaasa läinud panganduse uued normid on praeguseks ka seadustesse sisse kirjutatud. Alates 14.09.2019 on Euroopa majanduspiirkonnas asuvatel maksekontode pakkujatel kohustus tagada kliendi soovil ligipääs nende andmetele läbi *Open Banking*'u API'de [1]. Juhindutakse makseteenuse direktiivist (PSD2) [2], mis peab olema Euroopa liidu liikmesriikide seadustesse sisse viidud [3]. Eestis on makseteenuste direktiiv üle võetud järgneva kolme seadusega [4]: Makseasutuste ja e-raha asutuste seadus, Võlaõigusseadus ja Finantsinspektsiooni seadus. Suurbritannias alustati vastavate tegevustega juba paar aastat varem – sealne *Open Banking UK* standard arendati välja 2016-ndal aastal ja avatud panganduse API'sid hakati info edastamiseks kohustuslikus korras reaalselt kasutama 2018-nda aasta jaanuarist [5]. Avatud pangandus tähendab, et pangandusteenuseid, nagu maksete algatamine või kontoinfo kuvamine, saavad lisaks pankadele pakkuda ka teised teenuseosutajad, nt

finantstehnoloogia ettevõtted ja maksete vahendajad. Klientide jaoks tähendab see valikuvõimaluste laienemist läbi suurenenud teenusepakkujate vahelise konkurentsi [4].

Avatud panganduse kaudu saadud informatsioonil on mitmeid erinevaid võimalikke kasutusvaldkondi. PSD2 direktiivi järgi on see esmasena mõeldud kliendile kuvamiseks, kuid kui sellele lisada vastavad kliendi poolt saadud nõusolekud, siis on kasutusvaldkond palju laiem. Näiteks on pangalinkidelt ja *screenscrapingu* meetoditelt maksete tegemiseks aegamööda üle mindud avatud panganduse API'de kaudu maksete algatamisele. Erinevad kontoteabe teenuse pakkujad võidavad kliente sellega, et kogu oma finantsinfot saab vaadata ühes kohas ning osaliselt kasutatakse kättesaadavat infot juba praegu ka näiteks krediidivõimekuse hindamiseks – selle asemel, et paluda kliendil tuua oma teise panga viimase poole aasta väljavõte ja seda käsitsi töödelda, on teatud laenu pakkujad läinud vastava teenuse kasutamise teed. Klient loob kontoteabe teenuse pakkuja abil nõusoleku oma andmete kogumiseks, saadud info põhjal teostatakse kliendi nõusolekul krediidivõimekuse hindamine ja esmane laenuotsus on enamikel juhtudel võimalik automaatselt teha. See loob tugeva eelise kiire vastusega klientide püüdmiseks. Vastavate teenuste pakkujateks on näiteks ForPeeps rakendus ja Krediidiregister.

1.1 Töö käigus lahendatav probleem

Käesoleva töö eesmärk on lahendada probleem, mis tuleneb kasutusel olevatest erinevatest standarditest ja nende lahtisest kirjeldusest ja kohati vähesest rangusest – kõik API'de pakkujad on tõlgendanud kasutusele võetud standardit omale mugavamal viisil ja sealt on tekkinud olulised erisused, millega kontoteabe teenuse pakkujad silmitsi peavad seisma. Probleemi lahendamiseks töötab töö autor läbi erinevate pankade *Open Banking* API'de dokumentatsiooni ja testib läbi nende töövoo ning püüab leida optimaalse meetodi nende ühendamiseks ühtseks teenuseks sel moel, et järgnevate pankade API'de integreerimine toimuks minimaalse vaevaga.

2 Analüüs

Avatud panganduse reeglite alusel loodud rakendustel on võimalik lai kasutusala. Käesolevas töös keskendutakse vaid kontoteabe teenuse pakkumise osale. Vastava teenuse pakkumine loob mitmeid ärilisi võimalusi, kuid antud lahenduse esmane tehniline teostus on kõrge keerukusastmega. Olenemata sellest, et kõik Euroopa pangad on oma API'de loomisel juhindunud Euroopa komisjoni poolt vastu võetud tehnilistest standarditest (PSD2 RTS-st) ja suurem osa neist kasutanud ka *Berlin Group*'i [6] loodud detailset standardit „*Access to Account (XS2A) Open Banking Framework*" [7], on tegelikkuses vastavad standardid jätnud üsna suure osa vabalt tõlgendatavaks ja sellega seoses on erinevatel pakkujatel suurte erisustega töövood nõusoleku loomiseks ja kontoteabe hankimiseks. Samuti on mõningad ASPSP'd ka oma API'de loomisel juhindunud *Open Banking UK* standardist [8] mis on oma olemuselt erinev *Berlin Groupi'*i omast [7].

ASPSP'de esmaseks prioriteediks ei ole oma API'dega ühendumist lihtsamaks teha ja kulutada oma ressursse ühiste kokkulepete järgi ühtseks joondumiseks ja sarnaste API'de ehitamiseks. Seega on kontoteabe teenuse pakkuja jaoks üheks suurima keerukusega ülesandeks lahendada API'de erisused, et kliendi jaoks samast ja lihtsat teenust pakkuda. Enamik silmapaistvamaid erisusi on nõusoleku loomise protsessis. Erinevad on nii pankade nõusolekute loomise töövood kui ka päringute ning vastuste ülesehitus ja samuti ka sisu. Ka konfiguratsioonid ja nõuded päringutele ei ole ühtsed ning päringu turvasertifikaatide kasutused erinevad oluliselt.

Kui kehtiv nõusolek on juba loodud, siis järgnevate päringute tegemisel kontojäägi ja tehingute nimekirja saamiseks on juba rohkem ühiseid jooni, kuigi erisusi leidub ka seal. Näiteks on pangati erinev see, kas kontojäägi saab kätte koos kontode päringuga või on see vaja eraldiseisva päringuga küsida, kui ka see, kui pikaaegset tehingute nimekirja on üldse võimalik pärida.

Käesolevas töös on analüüsitud kontoteabe teenuse pakkuja keerukusi erinevate API'dega ühendumisel. Töö raames on plaanis lahendada API erisused, luues ühtne

prototüüp, mis liidab erinevate pakkujate API'de päringute tulemused ühtseks rakenduseks. Tulemusena on planeeritud valmima rakenduse esmane prototüüp, mis võimaldab kuvada erinevates pankades olevat kontode infot – kontojääki ning tehingute nimekirju. Kuna *live* ühenduse jaoks on vaja olla kvalifitseeritud makseteenuse pakkuja ja omada vastavat sertifikaati, siis on prototüübis ühendumiseks kasutatud mõningate piirangutega ASPSP-de poolt pakutud *sandbox'*ide kasutamise võimalust. Rakenduse edasi arendamine ja produktsiooni vastu ühenduste tegemine peaks teoorias olema juba lihtsam, kuna päringud on prototüübis läbi proovitud ja *sandbox*'is valideeritud.

Prototüüp on kirjutatud C# keeles, kuna see programmeerimiskeel on töö autorile kõige tuttavam ja uue keele omandamine piisaval tasemel oleks liiga ajamahukas. Andmete salvestamise maht on prototüübi tarbeks minimaalne – salvestatakse vaid nõusoleku detailid ning kontode nimekiri. Nii kontojääkide kui ka tehingute nimekirja päringuid tehakse vaid siis, kui klient reaalselt selleks soovi avaldab, ehk rakenduses sisse logituna on ja vastavat nuppu vajutab või lehekülge külastab. Selline lähenemine on ka kõige rohkem kooskõlas Makseasutuste ja e-raha asutuste seaduse § 63^2 lõige 2-ga, mis ütleb: „Kontoteabe teenuse pakkuja ei kasuta, ei püüa saada ega säilita kliendi andmeid muul eesmärgil kui kliendi poolt selgesõnaliselt taotletud kontoteabe teenuse osutamiseks kooskõlas andmekaitse normidega" [9].

Järgnevas analüüsis on välja toodud erinevate teenusepakkujate erisusi kui ka sarnasusi.

2.1 Nõusolekute erisused

Berlin Group'i standardi järgi saab luua 3-tüüpi nõusolekuid – „*Detailed consent*", „*Bank offered consent*" ja „*Global consent*". Põhiline erisus nende puhul on, et detailse nõusoleku puhul kuvatakse kliendile peale kontoteenust pakkuvas pangas autentimist kõigi teenuses võimalike kättesaadavate kontode andmed, mille seast klient teeb valiku, milliseid nõusolekusse lisada soovib ning nõusolek luuakse vaid nende kontode andmete kuvamise jaoks. Panga poolt pakutud nõusoleku puhul autendib klient ennast kontoteenust pakkuvas pangas ning saab allkirjastada panga poolt pakutud kontode valikuga nõusoleku, millele eelnevalt on võimalik ASPSP poolel muuta nõusolekusse kuuluvate kontode valikut. Globaalne nõusolek sisaldab kõiki saadaolevaid kontosid, mis vastavad kontoteabe teenuse tingimustele ASPSP poolt.

Vaatluse all olevad pangad kasutavad erinevaid nõusolekute tüüpe – näiteks SEB-l [10], Swedbank'il [11], Coop'il [12] ja Luminoril [13] on kasutusel detailne nõusolek, LHV [14] kasutab panga poolt pakutud nõusoleku versiooni ning Citadele [15] ja Šiaulių [16] klientidel on võimalus luua globaalne nõusolek, mis sisaldab kõiki saadaval olevaid kontosid. Revolut [17] järgib *Open Banking UK* standardit, mis erineb *Berlin Group'*i standardist juba rohkema kui vaid nõusoleku tüübi poolest.

2.2 Autentimine ja autoriseerimine

Analüüsi käigus uuritud pangad on oma API'de väljatöötamisel võtnud väga erinevaid seisukohti autentimise ja autoriseerimise teostamisel. Kui mõningatel juhtudel on kasutatud OAuth 2.0 autoriseerimise meetmeid, siis teistel puhkudel on autoriseerimise asemel kasutatud näiteks POST päringute allkirjastamist sertifikaadiga (Šiaulių bankas puhul kasutades QSEAL sertifikaati) või autoriseerimine üldse ära jäetud ja loodud nõusoleku loomise soovi kinnitav nupp, mis selle nõusoleku aktiveerib (näiteks Citadele puhul).

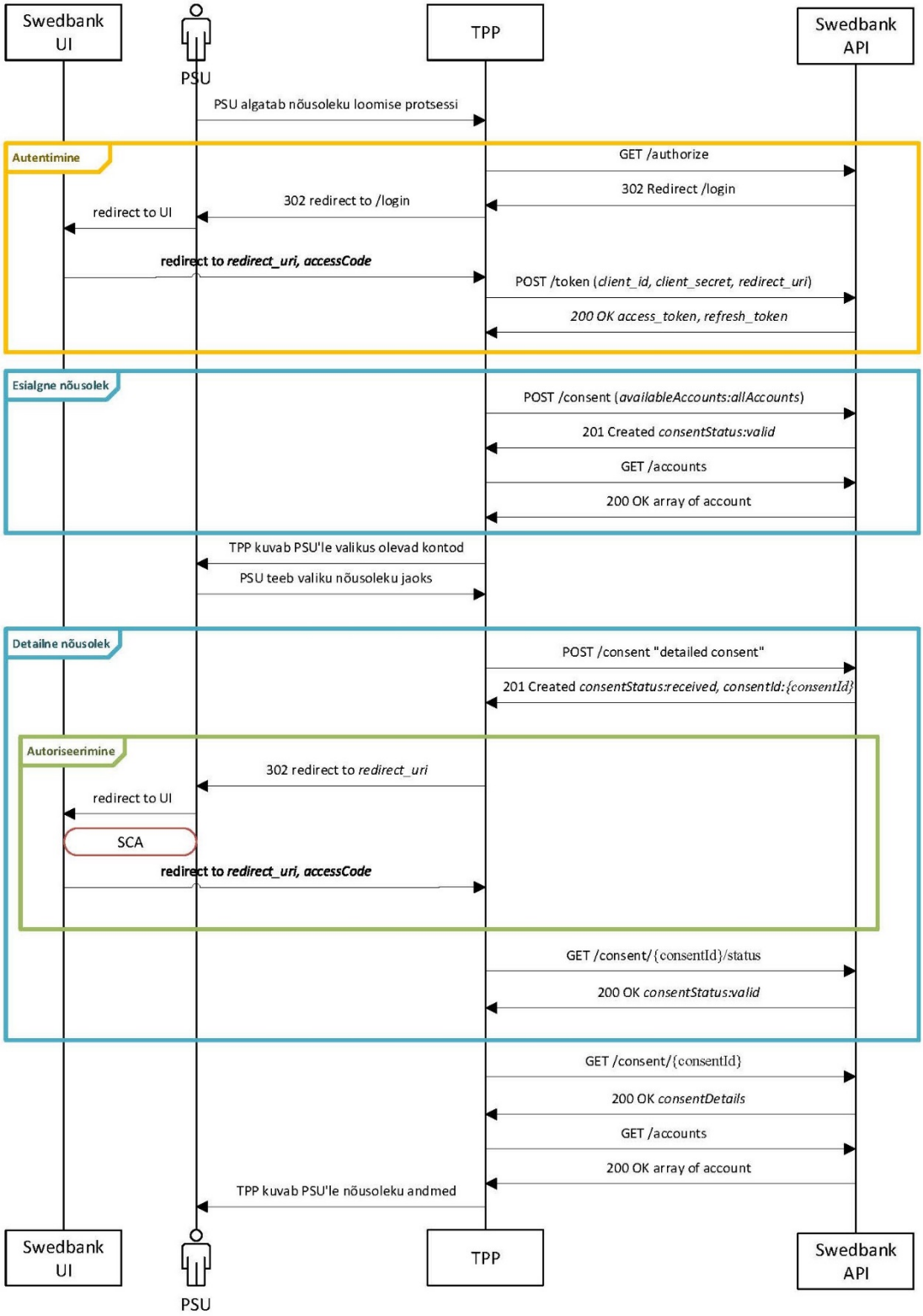
Samuti on ka autentimise ajaline asukoht erinevates töövoogudes üsna erinevates etappides. Kui Swedbank'i ja ka mitmete teiste pankade puhul alustatakse nõusoleku loomise voogu kliendi autentimisest enne isegi n.ö. esialgse nõusoleku loomist ja kontode nimekirja saamist ja autoriseerimist kasutatakse alles detailset nõusolekut kinnitades, kui skoop ja nõusolekus sisalduvad kontod juba valitud, siis Revolut'i ja Šiaulių bankas puhul alustatakse hoopis nõusoleku loomisest ja kliendi autentimisega tegeletakse peale seda, kui ASPSP poolt on juba tulnud edukas vastus esialgse tühja nõusoleku loomise kohta. Sealjuures on ka erisusi selles, mida ja mil moel mingi pank autoriseerimise tulemusel tagastab.

2.3 Nõusolekute loomise vood

2.3.1 Swedbank

Olulised detailid: maksimaalselt 90-päevane detailne nõusolek, mille loomise protsessi kuulub kaks korda POST /consent päringu tegemine – esmalt esialgse nõusoleku saamiseks kontode nimekirja pärimiseks ning teistkordselt koos juba olemasolevate kontode hulgast valitud andmetega detailse nõusoleku loomiseks. Nõusoleku loomise

protsessis luuakse *access_token* ja *refresh_token*, mida on võimalik TPP-l nõusolekus sisalduvate andmetele ligipääsuks kasutada 90-päeva jooksul nõusoleku loomisest alates.



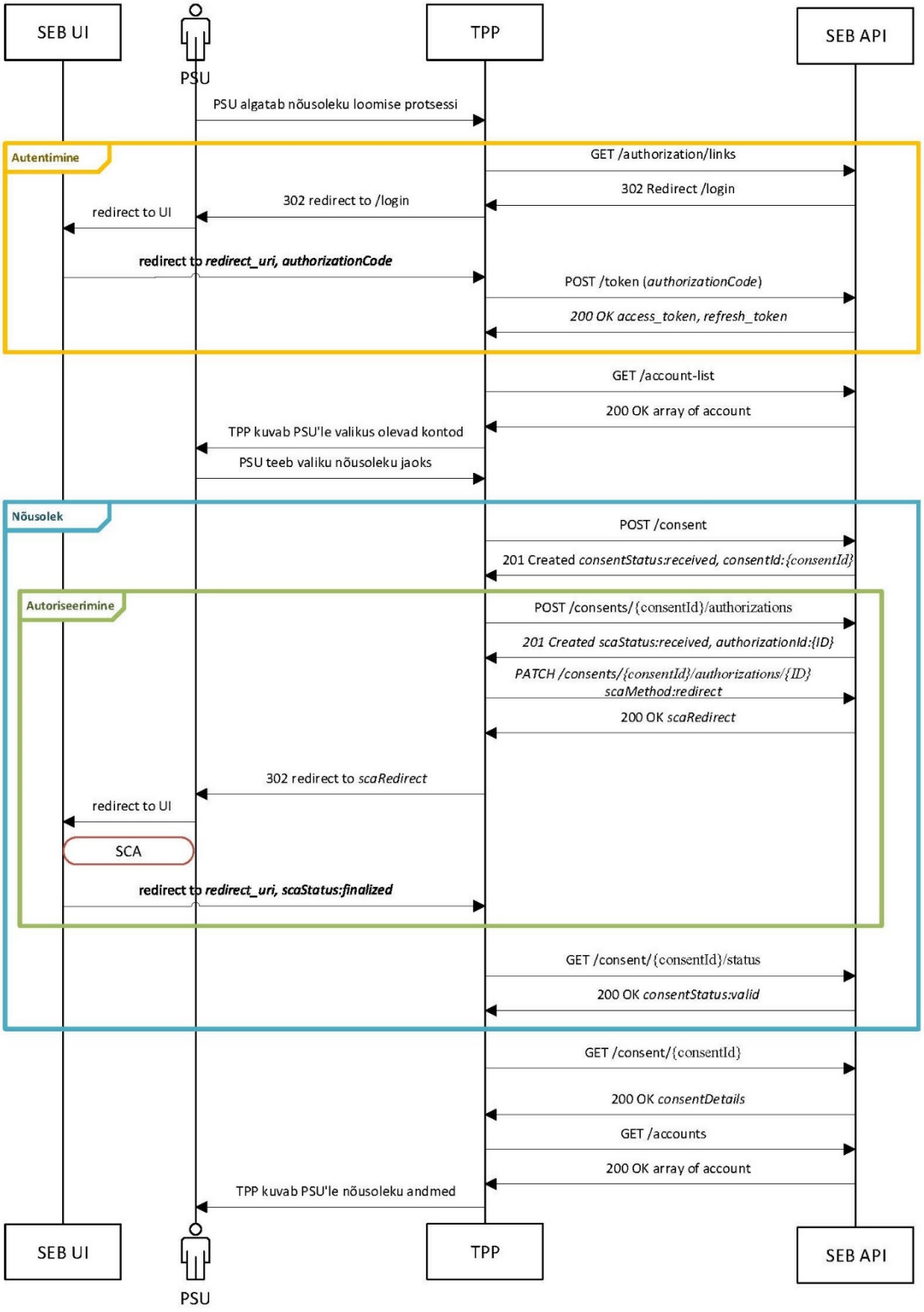
Joonis 1. Nõusoleku loomise töövoog Swedbank'is

- PSU algatab TPP rakenduse kliendiliideses nõusoleku loomise protsessi
- TPP algatab serveri poolel OAuth 2.0 päringu GET /authorize
- ASPSP tagastab HTTP koodi 302 ja redirect lingi vastuse päises
- TPP suunab kliendi ASPSP lehele ennast autentima
- PSU autendib ennast ASPSP lehel
- ASPSP suunab kasutaja TPP *redirect_uri* lehele koos *access_code* ja *state* parameetritega
- TPP teeb päringu POST /token endpointile ja vahetab *access_code* OAuth 2.0 *tokeni* vastu
- ASPSP saadab päringu vastusega *access_tokeni* ja *refresh_tokeni*
- TPP salvestab PSU *token*'id edaspidiseks kasutuseks ja teeb POST /consent päringu esialgse nõusoleku loomiseks
- ASPSP tagastab esialgse nõusoleku andmed (consentId)
- TPP teeb GET /accounts päringu
- ASPSP tagastab kontode nimekirja
- TPP suunab kliendi lehele, kus kliendile kuvatakse nimekiri kontodest, mida on võimalik nõusolekusse lisada
- PSU valib kontod ja nõusoleku aegumistähtaja
- TPP teeb POST /consent päringu detailse nõusoleku loomiseks
- ASPSP tagastab nõusoleku andmed ja staatuse (*received*) ning SCA päringu andmed
- TPP suunab PSU scaRedirect lingile nõusolekut autoriseerima

- PSU autoriseerib nõusoleku ASPSP lehel
- ASPSP suunab PSU tagasi TPP *redirect_uri* lingile ja saadab TPP-le nõusoleku andmed
- TPP salvestab nõusoleku andmed ja kontod ning teeb GET /consent/{consentId}/status päringuid seni, kuni nõusoleku staatuse vastuseks tuleb *valid*.
- TPP saadab GET /consent/{ID} päringu
- ASPSP saadab nõusoleku detailid
- TPP salvestab nõusoleku detailid ja teeb päringu GET /accounts
- ASPSP tagastab nõusolekus olevate kontode andmed.
- TPP salvestab kontode andmed. Peale seda on võimalik algatada kontojäägi ja tehingute pärimise protsessi.

2.3.2 SEB

Olulised detailid: maksimaalselt 90-päevane detailne nõusolek, mille loomise käigus luuakse *access_token* ja *refresh_token*, mida TPP saab hilisemalt nõusolekus sisalduvate andmete ligipääsuks 90-päeva jooksul kasutada. Kontode loend detailse nõusoleku jaoks saadakse ilma eraldi nõusolekut loomata kui on kasutusel spetsiaalne skoop *accounts.lists*.



Joonis 2. Nõusoleku loomise töövoog SEB's

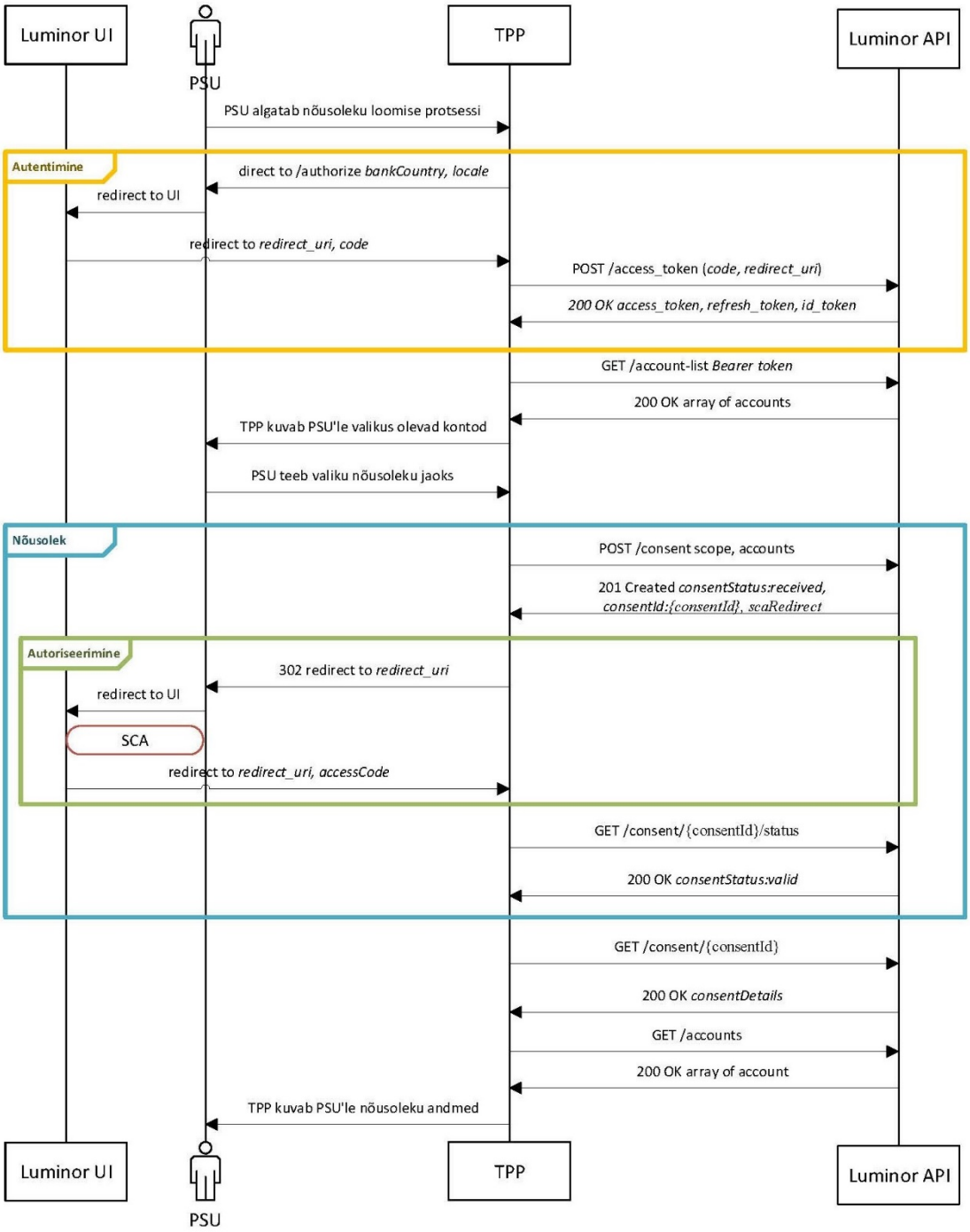
- PSU algatab TPP rakenduse kliendiliideses nõusoleku loomise protsessi

- TPP saadab GET /authorization/links päringu
- ASPSP tagastab *redirect* lingi
- TPP suunab kliendi ASPSP lehele ennast autentima
- PSU autendib ennast ASPSP lehel ja kinnitab nõusoleku skoobid
- ASPSP suunab tagasi TPP redirect_uri'le, lisades *authorization code*
- TPP teeb päringu POST /token endpointile, saates kaasa *authorization code*
- ASPSP tagastab *access token*'i ja *refresh token*'i
- TPP salvestab PSU *token*'id edaspidiseks kasutuseks ja teeb GET /account-list päringu
- ASPSP tagastab PSU kontode nimekirja
- TPP suunab kliendi lehele, kus PSU'le kuvatakse nimekiri kontodest, mida on võimalik nõusolekusse lisada
- PSU valib kontod ja nõusoleku aegumistähtaja
- TPP teeb POST /consent päringu nõusoleku loomiseks
- ASPSP tagastab consentId
- TPP teeb päringu POST consents/{ID}/authorizations, algatamaks autoriseerimist
- ASPSP tagastab autoriseerimise id, staatuse ja võimalikud meetodid
- TPP valib redirect meetodi ja teeb päringu PATCH /consents/{ID}/authorizations/{ID} (SCA method: redirect)
- ASPSP saadab staatuse ja redirect URL'i
- TPP suunab kliendi ASPSP redirect lehele autoriseerima
- PSU autoriseerib nõusoleku ASPSP lehel

- ASPSP saadab PSU TPP redirect-uri lehele ja saadab TPP-le nõusoleku andmed
- TPP salvestab nõusoleku andmed ja kontod ning teeb päringu GET /consents/{ID}/authorizations/{ID} senikaua, kuni vastuseks tuleb status: finalized
- TPP teeb päringu GET /consents/{ID}
- ASPSP saadab vastuse status: valid. Peale seda on võimalik algatada kontojäägi ja tehingute pärimise protsessi.

2.3.3 Luminor

Olulised detailid: detailne nõusolek, mille loomise käigus luuakse *access_token*, *id_token* ning *refresh_token*, mida saab hiljem kasutada nõusoleku andmetele ligi pääsemiseks. Dokumentatsioonist selgub, et *id_token*'it kasutatakse *access_token'*it värskendades *cookie*'s, mille sarnast tegevust ühegi teise analüüsitud panga töövoos ei esine.



Joonis 3. Nõusoleku loomise töövoog Luminor'is

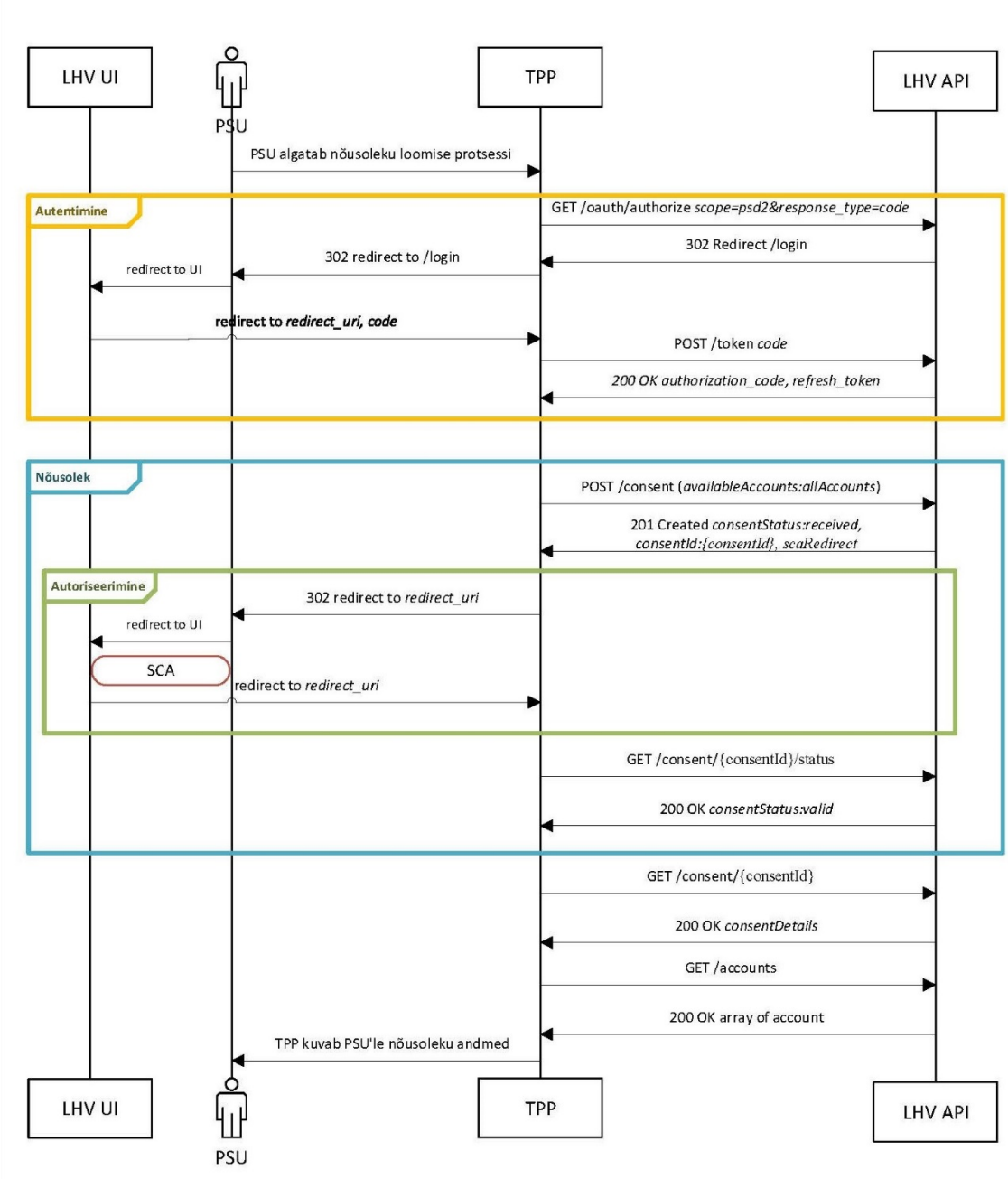
- PSU algatab TPP rakenduse kliendiliideses nõusoleku loomise protsessi
- TPP suunab PSU ASPSP autentimise lehele /am/oauth2/authorize, kasutades kaasa andes riigi ning keele valikuid

- PSU autendib end ASPSP lehel
- ASPSP suunab peale õnnestunud autentimist PSU redirect_uri'le ning lisab parameetri *code*
- TPP saadab POST /access_token endpointile päringu, lisades parameetritesse eelmise päringu vastusena saadud *code* ning „*grant_type=authorization_code*"
- ASPSP tagastab vastuse sisus *access_tokeni*, *id_tokeni* ning *refresh_tokeni* ning suunab PSU tagasi redirect_uri lehele
- TPP salvestab *token*'id edaspidiseks kasutamiseks ning saadab päringu GET /account-list
- ASPSP tagastab PSU valikus olevad kontod
- TPP kuvab kliendile saadud kontod oma kasutajaliideses
- PSU valib kontod ja nõusoleku lõputähtaja
- TPP teeb POST /consent päringu, lisades sisuks PSU poolt valitud kontod
- ASPSP saadab vastuses nõusoleku ID, staatuse: *received* ja lingi *scaRedirect*
- TPP suunab PSU *scaRedirect* lingile nõusolekut allkirjastama
- ASPSP suunab kliendi peale allkirjastamist tagasi TPP *redirectUrl'*i lehele
- TPP teeb GET consent/{ID}/status päringu senikaua, kuni vastuseks tuleb *status: valid*
- ASPSP saadab vastuse *status: valid*
- TPP saadab GET /consents/{ID} päringu
- ASPSP saadab nõusoleku andmed
- TPP salvestab nõusoleku detailid ning teeb päringu GET /accounts
- ASPSP saadab nõusolekus sisalduvate kontode andmed ja lingid

- TPP salvestab kontode andmed. Peale seda on võimalik algatada kontojäägi ja tehingute pärimise protsess.

2.3.4 LHV

Olulised detailid: maksimum 90-päevane panga poolt pakutud nõusolek, millesse kuuluvaid maksekontosid saab klient enne allkirjastamist ASPSP lehel valida. Süsteem kasutab sama kliendipõhist ligipääsu *token*'it kõigile erinevatele pakkujatele, mistõttu võib juhtuda, et *token* aegub palju varem kui lubatud nõusoleku 90 maksimaalset päeva. *Token'*i värskendamine on võimalik, kuid vajab uut SCA-d.

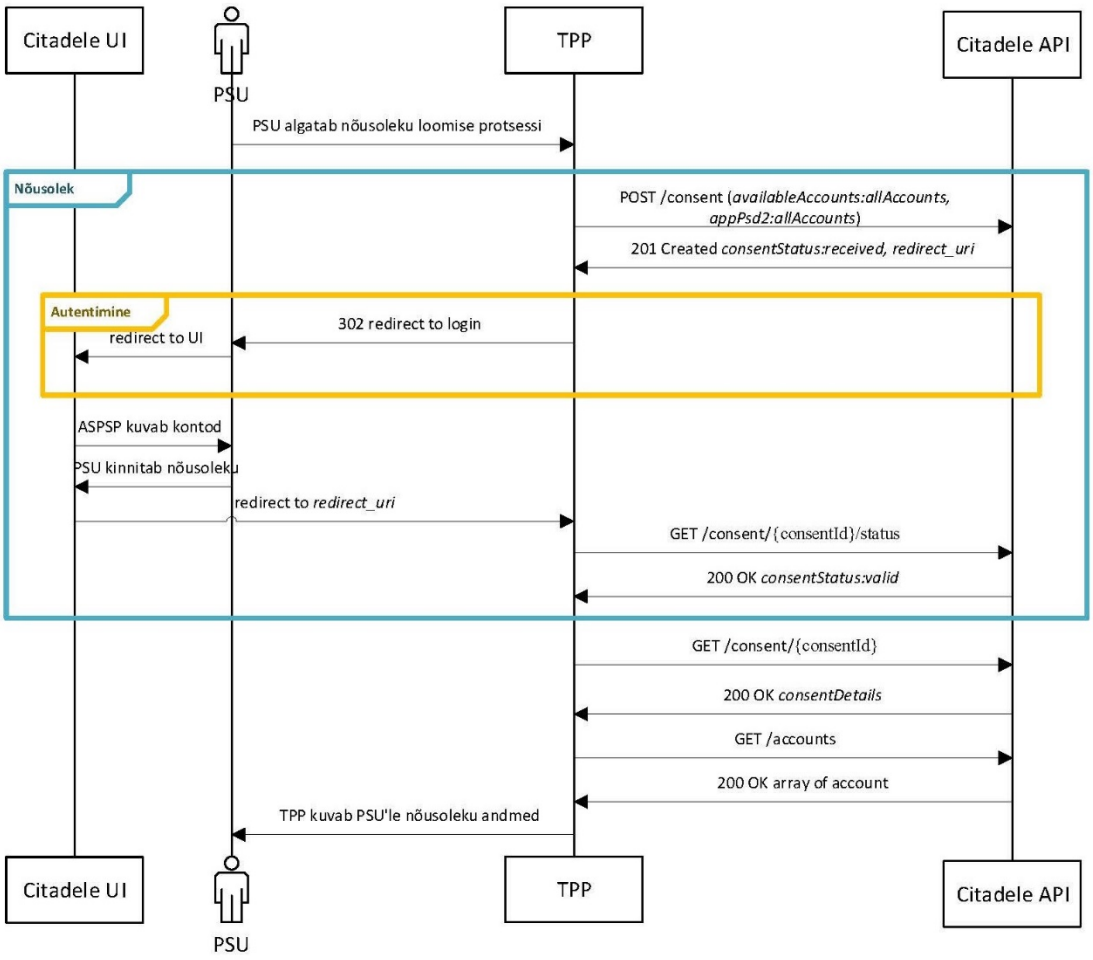


Joonis 4. Nõusoleku loomise töövoog LHV's

- PSU algatab TPP rakenduse kliendiliideses nõusoleku loomise protsessi
- TPP suunab kliendi ASPSP lehele ennast autentima GET /oauth/authorize päringuga, mille parameetriteks on scope=psd2 ja response_type=code
- PSU autendib ennast ASPSP lehel
- ASPSP suunab tagasi TPP redirect_uri'le, lisades päringu parameetritesse *code*, mis kehtib 3600 sekundit
- TPP teeb päringu POST /token endpointile, saates kaasa *code*
- ASPSP tagastab *authorization_code* ja *refresh tokeni*
- TPP salvestab *token'*id edaspidiseks kasutuseks ja teeb päringu POST /consents
- ASPSP suunab kliendi autoriseerimise lehele, kus kuvab kontod ja skoobid
- PSU teeb kontode hulgast valiku ja autoriseerib nõusoleku ASPSP lehel
- ASPSP suunab PSU tagasi TPP redirect_uri lehele
- TPP teeb GET consents/{ID}/status päringu senikaua, kuni vastuseks tuleb *status: valid*
- ASPSP saadab vastuse *status: valid*
- TPP saadab GET /consent/{ID} päringu
- ASPSP saadab nõusoleku detailid
- TPP salvestab nõusoleku detailid ja teeb päringu GET /accounts
- ASPSP tagastab nõusolekus olevate kontode andmed.
- TPP salvestab kontode andmed. Peale seda on võimalik algatada kontojäägi ja tehingute pärimise protsess.

2.3.5 Citadele

Olulised detailid: globaalne nõusolek, millele ligipääsuks ei looda *token*'it, piisab vaid *consentId*-st ning millesse kuuluvad automaatselt kõik kliendi maksekontod. TPP päringud on piiratud 4 peale päevas, olenemata kontode arvust.



Joonis 5. Nõusoleku loomise töövoog Citadele's

- PSU algatab TPP rakenduse kliendiliideses nõusoleku loomise protsessi
- TPP teeb POST /consents päringu, mille sisuks on *„availableaccounts":"allAccounts"* ja *„appPsd2":"allAccounts"*
- ASPSP loob esialgse nõusoleku ja tagastab lingi, kuhu PSU autentimiseks suunata

- TPP suunab PSU ASPSP lehele ennast autentima
- PSU autendib ennast ASPSP lehel
- ASPSP saadab TPP-le autoriseerimata nõusoleku andmed (Id ja staatuse) ning suunab PSU nõusolekut autoriseerima, kuvades talle kliendi kõik maksekontod
- PSU autoriseerib nõusoleku ASPSP lehel
- ASPSP suunab PSU *redirect_uri* lehele. Autoriseerimise ebaõnnestumise korral suunab ASPSP PSU *Nok_redirect_uri*'le
- TPP teeb päringu GET /consents/{ID}/status senikaua, kuni tuleb vastuseks *status: valid*
- ASPSP tagastab nõusoleku staatuse: *valid*
- TPP saadab GET /consents/{ID} päringu
- ASPSP saadab nõusoleku detailid
- TPP salvestab nõusoleku detailid ning teeb päringu GET /accounts
- ASPSP saadab vastuses nõusolekus sisalduvate kontode andmed
- TPP salvestab kontode andmed. Peale seda on võimalik algatada kontojäägi ja tehingute pärimise protsessi

2.3.6 Coop Bank

Olulised detailid: detailne nõusolek, mille loomise protsessi kuulub kaks korda POST /consent päringu tegemine – esmakordselt esialgse nõusoleku saamiseks kontode listi pärimiseks ning teistkordselt koos juba olemasolevate kontode andmetega detailse nõusoleku loomiseks. OAuthi voog on dokumenteerimata. *Sandbox*'ile ligipääsuks on vaja registreerida reaalse makseteenuse osutaja firmaga, seega jääb praeguse töö raamest välja

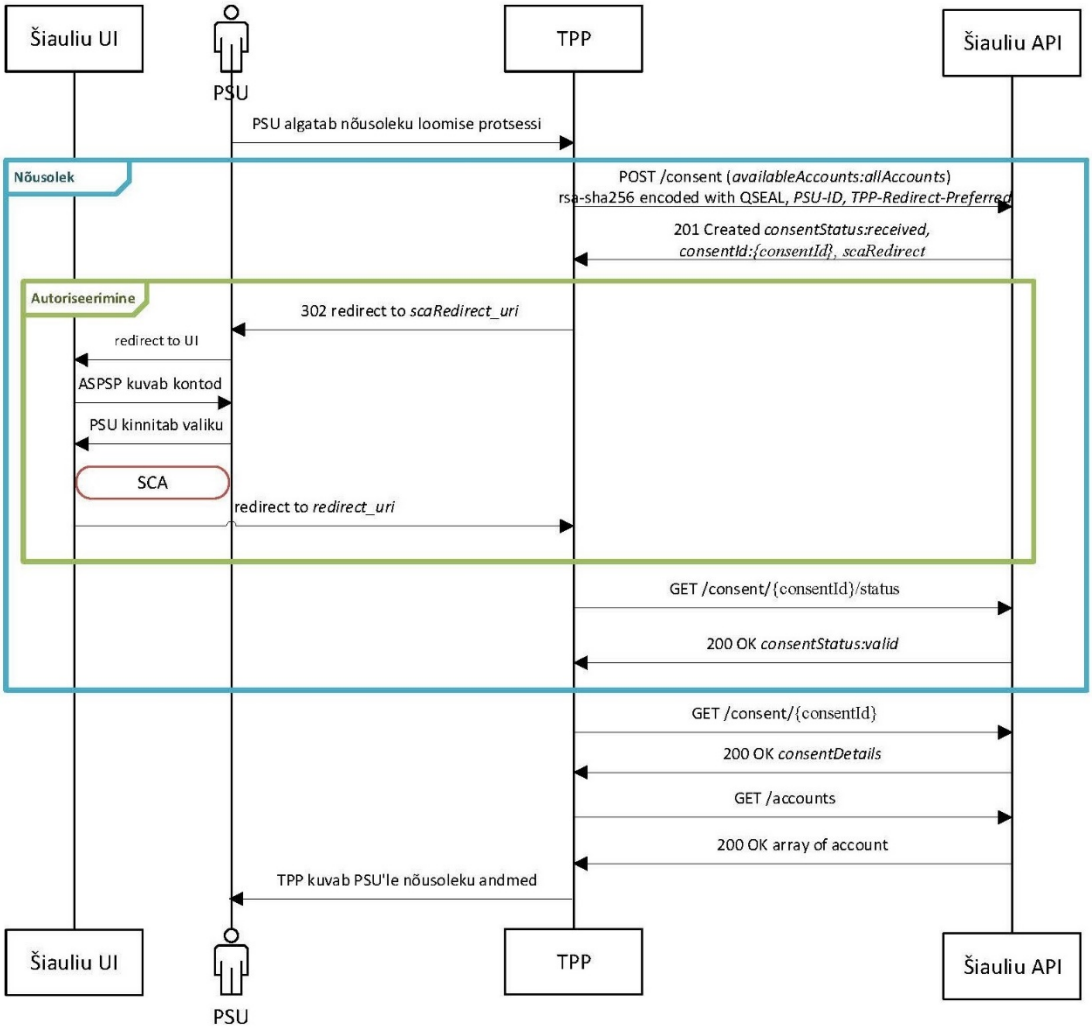
- Dokumentatsioonis kirjeldamata OAuth voog
- TPP teeb POST /consent päringu esialgse nõusoleku loomiseks

- ASPSP tagastab esialgse nõusoleku andmed (consentId)
- TPP teeb GET /accounts päringu
- ASPSP tagastab kontode nimekirja
- TPP suunab kliendi lehele, kus kliendile kuvatakse nimekiri kontodest, mida on võimalik nõusolekusse lisada
- PSU valib kontod ja nõusoleku aegumistähtaja
- TPP teeb POST /consent päringu detailse nõusoleku loomiseks
- ASPSP tagastab nõusoleku andmed ja staatuse (*received*) ning SCA päringu andmed
- TPP suunab PSU scaRedirect lingile nõusolekut autoriseerima
- PSU autoriseerib nõusoleku ASPSP lehel
- ASPSP suunab PSU tagasi TPP *redirect_uri* lingile ja saadab TPP-le nõusoleku andmed
- TPP salvestab nõusoleku andmed ning teeb GET /consent/{consentId}/status päringuid seni, kuni nõusoleku staatuse vastuseks tuleb *valid*.
- TPP saadab GET /consents/{ID} päringu
- ASPSP saadab nõusoleku detailid
- TPP salvestab nõusoleku detailid ning teeb päringu GET /accounts
- ASPSP saadab vastuses nõusolekus sisalduvate kontode andmed
- TPP salvestab kontode andmed. Peale seda on võimalik algatada kontojäägi ja tehingute pärimise protsessi.

2.3.7 Šiaulių bankas

Olulised detailid: nõusoleku algatamise päring on vaja signeerida QSEAL sertifikaadi privaatvõtmega, rsa-sha256 krüpteeringus, ning sama päringuga on vaja kaasa saata ka

PSU ID-kood. *Token*'it ei kasutata, nõusoleku andmetele ligi pääsemiseks piisab *consentId*-st.



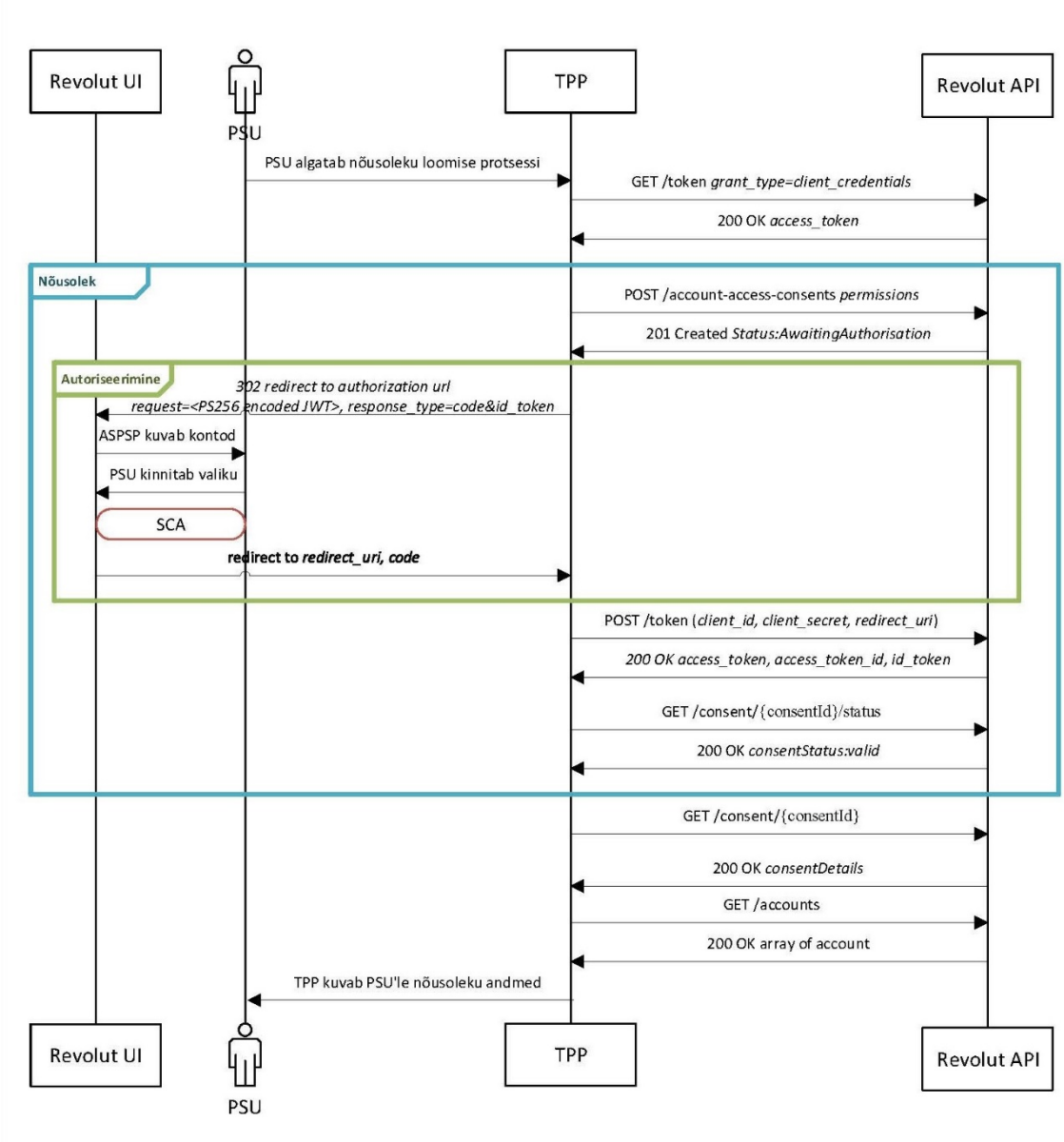
Joonis 6. Nõusoleku loomise töövoog Šiaulių's

- PSU algatab TPP rakenduse kliendiliideses nõusoleku loomise protsessi
- TPP saadab POST /consents päringu, signeerides selle QSEAL sertifikaadiga ja lisades ka PSU ID koodi
- ASPSP tagastab nõusoleku staatuse (*received*), Id ja *scaRedirect* lingi
- TPP suunab PSU *scaRedirect* url'le

- PSU autendib ja autoriseerib nõusoleku ASPSP lehel
- ASPSP suunab PSU tagasi TPP *redirect_uri*'le
- TPP teeb GET consents/{ID}/status päringu seni, kuni saab vastuseks *status: valid*
- ASPSP saadab vastuse nõusoleku staatuse: *valid*
- TPP teeb päringu GET consents/{ID}
- ASPSP saadab vastuses nõusoleku detailid
- TPP salvestab nõusoleku detailid ja teeb GET /accounts päringu
- ASPSP saadab vastuses kontode nimekirja ja kui päringus märgitud, siis ka kontojäägid
- TPP salvestab kontode andmed. Peale seda on võimalik algatada kontojäägi ning tehingute pärimise protsessi

2.3.8 Revolut

Olulised detailid: *Open Banking UK*-le kohaselt on protsessis vajalik JWK-le ligipääs veebist. Seega peab olema loodud JWK ja see üles laetud internetist ligipääsetavasse kohta – aplikatsiooni andmetesse *dev portal*'is lisatakse JWK URL. Nõusoleku loomise protsessi ajal on tarvilik luua JWT ning see allkirjastada PS256 krüpteeritult sama QWAC sertifikaadi privaatvõtmega, mida kasutatakse ka transpordis. Nõusoleku lõpukuupäeva seadmine ei ole vajalik, nõusolek võib olla ka ilma lõpptähtajata, kuid ligipääsu *token* aegub 90 päeva pärast ja siis on vaja nõusoleku jätkamiseks uuesti luua allkirjastatud JWT.



Joonis 7. Nõusoleku loomise töövoog Revolut'is

- PSU algatab TPP rakenduse kliendiliideses nõusoleku loomise protsessi
- TPP teeb päringu POST /token koos parameetriga *grant_type=client_credentials*
- ASPSP saadab TPP-le vastuse sisus access_token'i, mis kehtib 2400 sekundit
- TPP alustab nõusoleku loomise protsessi, saates päringu POST /account-access-consents saates kaasa sisus nõusoleku skoobi ja soovi korral lõpukuupäeva ning tehingute kättesaadavuse ajalised piirangud

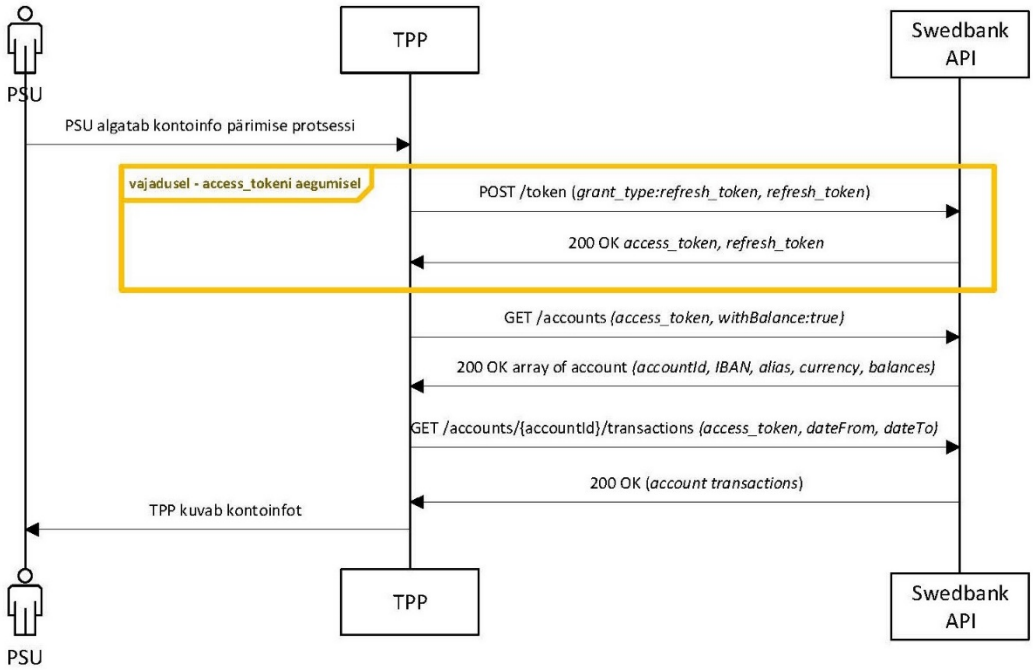
- ASPSP saadab tagasi autoriseerimist ootava nõusoleku andmed (ID ja ajalised piirangud)
- TPP salvestab esialgsed nõusoleku andmed ja loob JWT URL parameetri, mille allkirjastab QWAC sertifikaadiga.
- TPP suunab PSU autoriseerimise URL'le /ui/index.html ASPSP lehel, lisades url'le skoobi, eelmise sammuna loodud JWS'i, ja redirect_uri
- PSU sisestab ASPSP lehel oma telefoni numbri ning pin-koodi.
- ASPSP saadab PSU telefoni aplikatsiooni autoriseerimise jaoks koodi
- PSU sisestab autoriseerimise koodi
- ASPSP kuvab PSU'le tema kontod ja nõusoleku skoobid
- PSU kinnitab nõusoleku loomise soovi vastaval lehel, kus teeb ka valiku nõusolekusse minevate kontode osas.
- ASPSP suunab PSU tagasi TPP *redirect_uri* lehele, pannes parameetritesse lisaks muule *code*, mis on kehtiv kuni kaks minutit
- TPP vahetab autoriseerimise koodi *access_token*'i vastu, tehes päringu POST /token ja saates parameetri *grant_type=authorization_code*
- ASPSP saadab vastuses tagasi *access_token*'i, mis kehtib 90 päeva ja lisaks *access_token_id* ning *id_token*'i
- TPP teeb päringu GET /accounts, kasutades *access-token*'it
- ASPSP tagastab kontode nimekirja.
- TPP salvestab nõusolekus olevat kontod. Peale seda on võimalik algatada kontojäägi ja tehingute pärimise protsessi

2.4 Kontojääkide ja tehingute päringute erisused

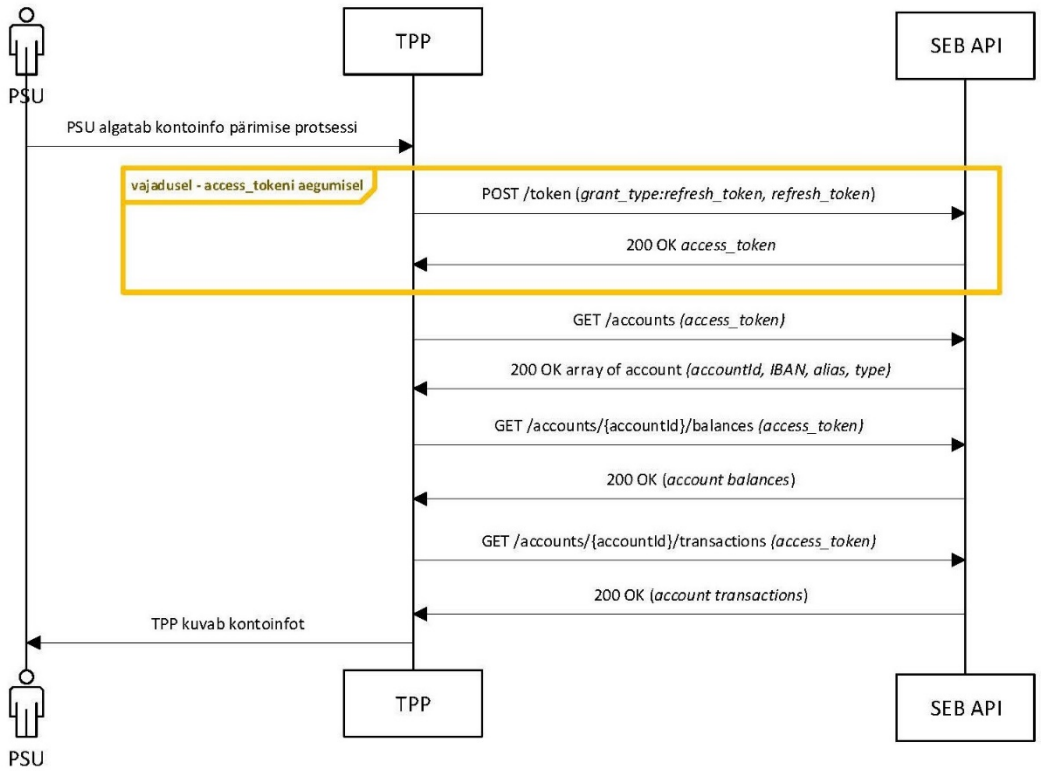
Kuigi kontojääkide ja tehingute päringud on olemuselt rohkem sarnased kui seda on nõusoleku loomine, siis leidub ka siin iseärasusi, millega tuleb arvestada. Oluline on

jälgida *access_token*'i eluiga selle olemasolul. Kui *accsess_token* on enamasti lühikese elueaga (näiteks Swedbank'is ja SEB's tund aega) ja vajab peale selle aja möödumist värskendamist *refresh_token*'iga, siis on tagasi saadavad tulemused mõningaste erinevustega.

Näiteks kehtib nii Swedbank'i kui ka SEB *refresh_token* 90 päeva. Kuid POST /token päringut tehes *token*'i värskendamiseks saab Swedbank'is iga kord lisaks uuele *access_token*'ile ka uue *refresh_token*'i, samas kui SEB värskendamise jaoks vajalikku *refresh_token*'it ei uuenda. Swedbank'i puhul on seega vaja tähelepanu pöörata mitme lõimega paralleelselt päringuid tehes, et sama nõusoleku kohta ei sooritataks päringuid samaaegselt erinevatelt lõimedelt, kuna see võib kaasa tuua olukorra, kus *refresh_token* on äsja uuendatud ja hetk varem välja vahetatud eelneva *token*'iga uut värskendust tehes saab vastuseks veateate.



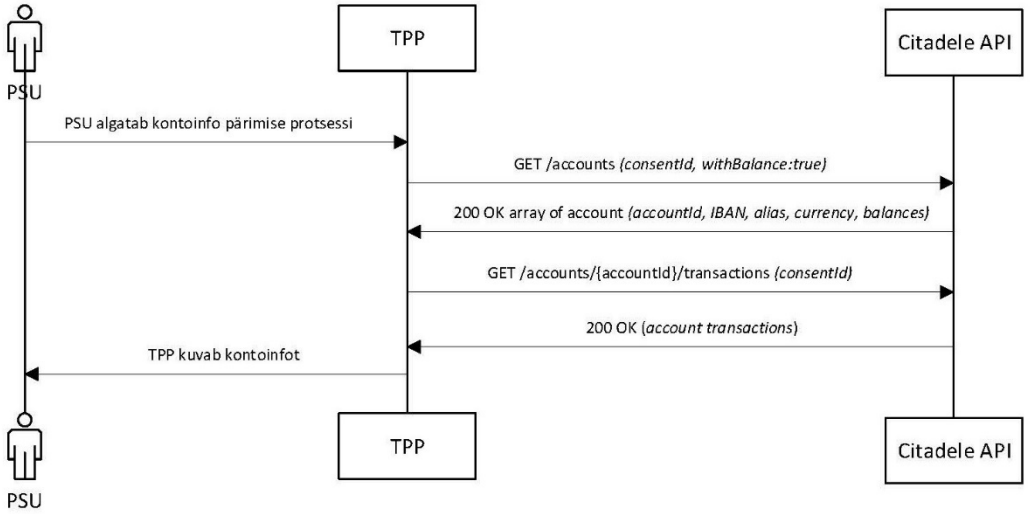
Joonis 9. Kontojääkide ja tehingute pärimine Swedbank'is



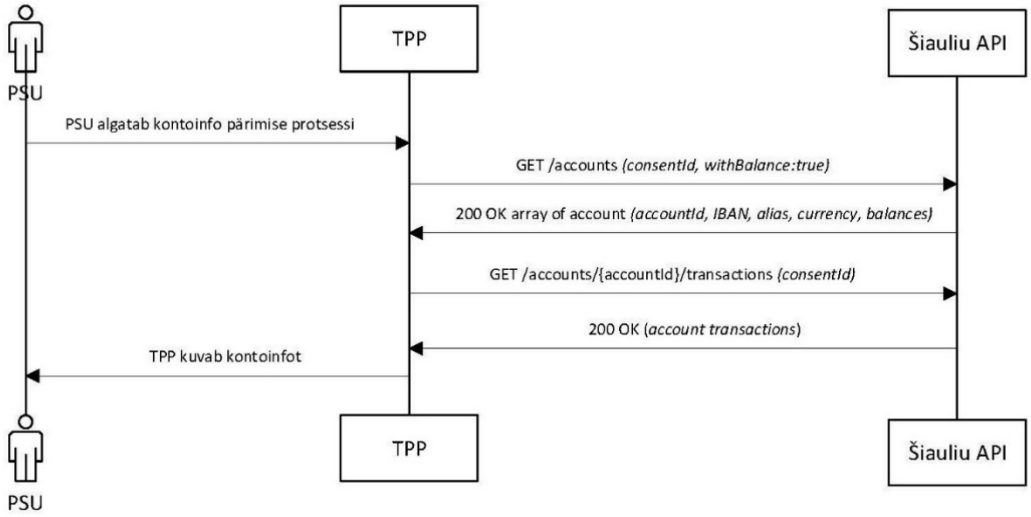
Joonis 8. Kontojääkide ja tehingute pärimine SEB's

Üheks oluliseks erisuseks erinevate teenusepakkujate kontode päringute vahel on võimalus kontojääki kaasa saada GET /accounts päringut tehes. Kui see valik on toetatud, siis vastava päringu lõpp-punkt võtab arvesse päises oleva *withBalance boolean*'i. Seda võimalust kasutades on võimlaik ära jätta eraldi päring kontojääkide saamiseks. Olenevalt nõusolekus sisalduvate kontode arvust saab seega vajadusel minimiseerida vajalike päringute arvu.

Citadele ja Šiaulių puhul on märkimisväärne see, et kuna neil ei ole kasutusel *access-token*'it, siis saab nende pankade nõusolekute puhul kontojääkide ja tehingute päringuid teha vaid *consentId*'d kasutades. Selline lähenemine on mõnevõrra vähem turvaline, kuid transpordikihis kasutatakse *live*-ühenduses siiski turvalist ja autentset QWAC sertifikaati, mis vastavat riski maandab. Muus osas on nende kahe panga kontojääkide ja tehingute päringud samuti sarnased.



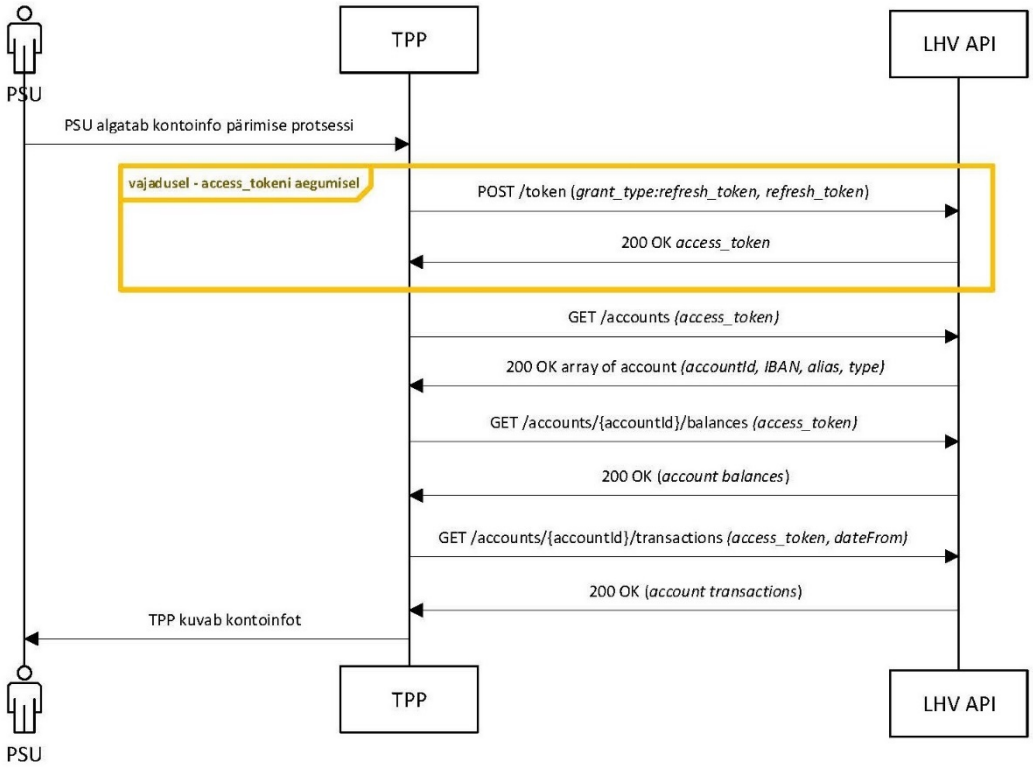
Joonis 10. Kontojääkide ja tehingute pärimine Citadele's



Joonis 11. Kontojääkide ja tehingute pärimine Šiaulių's

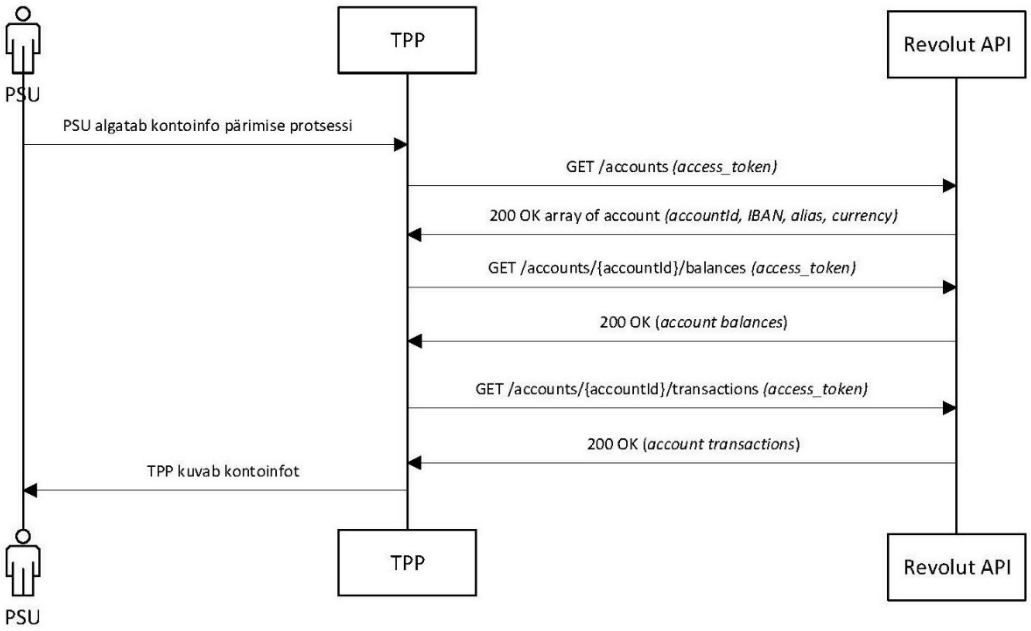
LHV puhul on muule lisaks veel üks keerukust lisav fakt – *refresh_token*'i eluiga on 90 päeva, mis on samaväärne maksimaalse nõusoleku pikkusega, kuid kuna LHV kasutab *live*'s kõigi oma PSD2 API'de poolt pakutavate teenuste jaoks samal kliendil sama *token*'it, siis võib juhtuda, et nõusolek on loodud tunduvalt hiljem kui *refresh_token* ja sellega seoses tekib mõne aja pärast olukord, kus 90-päevase kehtivusajaga *token* aegub

ja päringute tegemiseks on vaja kliendil see uuesti autoriseerida, kuigi nõusolek oli veel kehtiv. Uue *refresh_token*'i tekitamiseks on vaja läbi käia kogu nõusoleku loomise alguses alustatav protsess, kus saadakse peale autentimist *code* mis seejärel POST /token päringuga uue *token*'ite paari vastu vahetatakse.

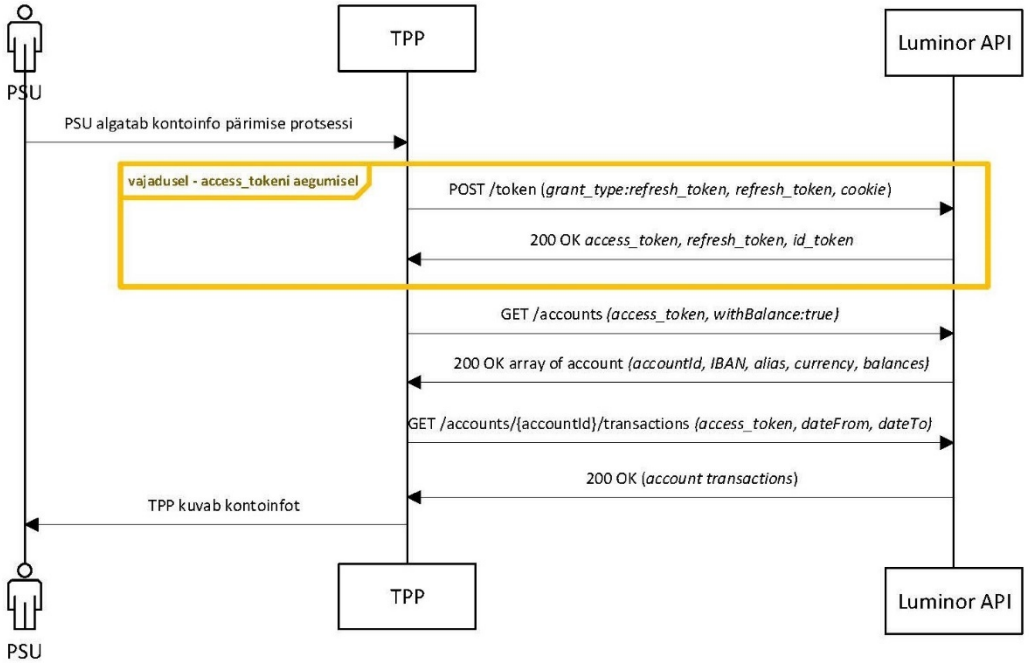


Joonis 12. Kontojääkide ja tehingute pärimine LHV's

Revoluti puhul on olukord lihtsam, nende poolt TPP'le saadetud *access_token*'i eluiga on 90 päeva ja seega seda iga tunni aja tagant värskendada ei ole vajadust. Kuna *Open Banking UK* standardi järgi ei ole nõusolekule lõpukuupäeva vaja märkida, siis tagatakse 90-päevase elueaga *access_token*'iga see, et klient käiks enda nõusoleku sees olevate andmete jätkuvale edastamisele kinnituse andmiseks *token*'it taasloomas.



Joonis 13. Kontojääkide ja tehingute pärimine Revolut'is



Joonis 14. Kontojääkide ja tehingute pärimine Luminor'is

2.5 Tehingute pärimine varasemast perioodist kui 90 päeva

Berlin Group'i standardi [7] järgi on tehingute pärimise võimalik ajaperiood päringu tegemise päevast alates 90 päeva minevikku. Kui klient soovib saada kaugemas ajaloos olevaid tehinguid, on tarvilik vastav päring spetsiaalselt autoriseerida. Erinevad pangad on ka seda nõuet erineval moel tõlgendanud. Mõningatel juhtudel on võimalik nõusolekut andes juba määrata pikema perioodi päringu võimalikkus, mõningal juhul ei ole pikema perioodi pärimine üldse võimalik ja on ka variant, kus eraldiseisvat autoriseerimist tarvis ei ole.

Citadele – dokumentatsioonis ajalisi piiranguid välja toodud ei ole. Päringuid võib teha nii kaugele ajalukku, kui soovid, kuid ühe vastusega saadetakse maksimaalselt 1000 tehingut.

LHV – vanemate kui 90-päeva vanuste tehingute päringud pikaajalises korduvkasutatavas nõusolekus (tavapärane kontoteabe teenuse nõusolek, mida käesolevas töös põhiliselt vaadeldakse) võimalik ei ole. Vastava päringu jaoks on võimalik kasutada lühiajalist nõusolekut. Lühiajaline nõusolek ei tühista küll pikaajalist korduvkasutatavat nõusolekut, kuid GDPR'iga seonduvaid probleeme võib tekkida seoses hilisema saadud andmete kasutamisega, kuna nõusolek on kehtiv vaid pooleks tunniks ja ühekordseks päringuks ning peale seda aegub.

SEB – tehingute pärimise perioodi pikkusele kuupäevalist limiiti ei ole. Küll aga on olemas ühes päringus sisalduv ajaline limiit (maksimaalselt 30 päeva ühe päringuga) ning mahu limiit (500 tehingut ühes vastuses). Kui vastav päring sisaldas koguselt rohkem tehinguid, siis sisaldub vastuses ka link eelnevatele ning järgmistele lehekülgedele. Samas pikema perioodi pärimise jaoks eraldi nõusolekut anda või autoriseerimist teha vaja ei ole, kuna SEB seisukoht on tulenev nõudest, et API kaudu peaks ligipääsu võimaldama samadele andmetele, mida on võimalik kliendil saada läbi otseste kanalite SEB enda keskkondades.

Swedbank – Baltikumis on üle 90-päeva vanuste tehingute pärimiseks vaja iga konto jaoks koostada eraldi päring, mis on tarvis ka kliendil autoriseerida. Peale autoriseerimist on võimalik saadud lingilt alla laadida tehingute väljavõte JSON formaadis. Rootsis on loodud eraldi nõusoleku skoop, mis sisaldab kohe alguses

pikemaajalise tehingute päringu võimalust edasistes päringutes ilma eraldi autoriseerimiseta.

Luminor – käesoleva töö valmimise ajal on Luminor'i pikemaajalise tehingute päringu võimalus alles arendamisel, kuid dokumentatsiooni järgi on peatselt tulemas Swedbank'i lahendusele sarnane lahendus, kus iga konkreetne päring on vaja ka autoriseerida.

Šiaulių bankas – dokumentatsioonis ei ole välja toodud võimalikku päringu perioodi pikkust.

Coop Pank – korduvkasutatavas nõusolekus ei ole võimalik pikema perioodi kui viimase 90 päeva kohta tehingute väljavõtet pärida. Ühekordses nõusolekus on võimalik tehinguid pärida kuni 7 aastat minevikku.

Revolut – varasemaid kui 90-päevaseid tehinguid saab pärida vaid 5-minuti jooksul peale nõusoleku loomist. Peale seda ajaperioodi piiratakse tehingute päringu ajaline pikkus 90-päevani alates päringu tegemise päevast.

2.6 Mitme valuutaga kontod

Asjaolu, millele tasub rakendust disainides tähelepanu pöörata on see, et pangad kasutavad erineval moel mitme valuutaga kontosid. See võib osutuda probleemiks, kui ei ole ette arvestatud esialgse korrektse valuutatähise asemel „XXX" tähise tagastamist või tehingute pärimisel erinevate valuutade vastuvõtmist ja nende oma süsteemides filtreerimist ning kuvamist.

Swedbank'i ja LHV puhul on kasutusel multivaluuta kontod – igal IBAN'il on võimalik omada mitmeid valuutasid kuid iga IBAN'i kohta on vaid üks *resourceId*. Kontojääke ja tehinguid *resourceId* järgi pärides saab vastustest kätte kõigi selle konto valuutade kontojäägid ja tehingud. GET /accounts päringu vastuses olevate kontode valuuta on märgitud tähisega XXX.

SEB – *resourceId* on unikaalne iga IBAN'i ja valuuta kombinatsiooni jaoks. Seega peab iga IBAN'i all oleva valuuta kohta tegema eraldi päringu saamaks kõiki kontojääke ning tehinguid. Peale uue valuuta avamist kontol tekib ka uus *resourceId*, mis peale selle tekkimist GET /accounts päringu vastusesse lisatakse.

2.7 API'de konfiguratsioonid nõusolekutele

Access_token:

- Swedbank, SEB, LHV – 1 tund, Revolut – 90 päeva

Refresh_token:

- Swedbank, SEB, LHV – 90 päeva
- Citadele – *token* puudub, andmeid päritakse *consentId*-d kasutades

Nõusoleku kestvus:

- Swedbank, SEB, Citadele, LHV, Luminor – maksimaalselt 90 päeva
- Revolut – vastavalt Open Banking UK standardile ei ole vajalik nõusoleku aegumise aega määrata, küll aga on ligipääsu *token*'it vaja uuendada 90 päeva möödudes uuesti autoriseerides

Tehingute päringud:

- SEB, Luminor – ühe tehingute nimekirja päringu ajaline kestvus ei tohi ületada 30 päeva.

	Nõusoleku kestvus	***Access_token*'i kestvus**	***Refresh_-token*'i kestvus**	**Ühekordne tehingute päring**	**Tehingute ajalugu**
Swedbank	Kuni 90 päeva	1 tund	90 päeva, ühekordseks kasutamiseks	Piiranguteta	Pikem kui 90 päeva vajab eraldi SCA'd
SEB	Kuni 90 päeva	1 tund	90 päeva	Kuni 30 päeva ja 500 tehingut	Ei ole piiratud
Luminor	Kuni 90 päeva	1 tund	Dokumentat-sioonis täpsustamata	Kuni 30 päeva	Päringu tegemisest 90 päeva, pikem periood arendamisel eraldi SCA'ga
LHV	Kuni 90 päeva	1 tund	90 päeva, jagatud teiste teenustega, võib kehtivuse kaotada ka varem	Piiranguteta	Päringu tegemisest 90 päeva, varasema perioodi jaoks vaja luua eraldi ühekordne nõusolek
Citadele	Kuni 90 päeva	puudub	puudub	1000 tehingut	Ei ole piiratud

Coop Bank	Kuni 90 päeva	Dokumentat-sioonis täpsustamata	Dokumentat-sioonis täpsustamata	Dokumentatsioonis täpsustamata	Päringu tegemisest 90 päeva, varasema, kuni 7-aastase perioodi jaoks, vaja luua eraldi ühekordne nõusolek
Šiaulių bankas	Kuni 90 päeva	puudub	puudub	Dokumentatsioonis täpsustamata	Dokumentatsioonis täpsustamata
Revolut	Ei pea määrama	90 päeva	Puudub	Piiranguteta	5 minuti jooksul nõusoleku loomisest piiramata, edasi päringu tegemisest 90 päeva tagasi

Tabel 1. API'de konfiguratsioonid

2.8 Päringute allkirjastamine

Kõik Euroopa pankade vahelised PSD2 API päringud kasutavad TLS sertifikaati. *Live*-ühenduste puhul on kontoteabe teenuse jaoks transpordi sertifikaadiks kasutusel Euroopa liidus PSD2 eIDAS'e QWAC sertifikaat. Lisaks on mõnede pankade puhul vajadus kasutada ka sertifikaate konkreetsete päringute või päringu osade allkirjastamiseks nõusoleku loomise protsessis.

Revoluti puhul on nõusoleku loomise käigus vaja allkirjastada muidu transpordiks kasutatava QWAC sertifikaadiga kindlate väljadega loodud JWT ja kodeerimiseks tuleb kasutada PS256 algoritmi. Allkirjastamist on vaja küll vaid nõusoleku loomise protsessis ja edasised päringud seda ei vaja, kuid oma rakendust üles ehitades tuleb arvestada sellega, et transpordil kasutatav privaatvõti peab olema kasutatav ja kättesaadav ka allkirjastamiseks. Analüüsi käigus uuritud teised pangad seda ei nõua.

Leedu panga Šiaulių bankas nõusoleku loomise protsessis on vaja allkirjastada POST /consents päring QSEAL sertifikaadiga kasutades rsa-sha256 krüpteerimist ja lisada päringuga kaasa ka vastav avalik võti. Live-ühenduse jaoks tähendab see aga seda, et AISP teenuse pakkumiseks vastavas pangas on vaja omada ka teist sertifikaati, mida ülejäänud analüüsis käsitletud pangad ei nõua. Seega on selle integratsiooni tegemine reaalne lisakulu sertifikaadi maksumuse näol.

Testkeskkondade jaoks on sertifikaatide loomiseks pakutud erinevaid võimalusi. Revoluti *developer portal´*is on võimalik üles laadida nende poolt antud käsu järgi oma

süsteemides loodud CSR fail, mille põhjal nad kasutamiseks sertifikaadi loovad ja ühendujale tagastavad. SEB's ja LHV's saab *developer portal*'is genereerida vastavalt kasutajale testkeskkonna sertifikaadi, Šiaulių panga *developer portal*'ist on võimalik vajalikud testkeskkonna QWAC'ile ja QSEAL'ile vastavad sertifikaadid alla laadida ning mõningad pankad on testkeskkonnas sertifikaadi kasutamise vajaduse üldse ära jätnud.

2.9 Erinevused live-lahenduste ja *sandboxi*'de vahel

Testkeskkondades *sandbox*ide vastu ühendusi tehes saab vajalikud transpordi sertifikaadid reeglina genereerida koostöös testitava panga ja/või nende *developer portal'*iga. Variante selleks on erinevaid. Mõningail juhtudel ei ole transpordi sertifikaati testkeskkonnas kasutusele võetud ja seega on loodud erisus reaalsuse ja testkeskkonna vahel. Sellisteks näideteks analüüsis olnud pankade valikus on Swedbank, Citadele ja Luminor, milledest viimasel on *sandbox'*is kasutusel eraldi päringu päis sisuga *client_secret.*

Mõned teenusepakkujatest on otsustanud testkeskkonnas vahele jätta autoriseerimiseks vajaliku OAuth etapi. Citadele pakub autentimise asemel valmis loodud *consentId* saatmist e-posti teel, mida hiljem päringute tegemisel kasutada. Seega on testkeskkonnas vahele jäetud kõik nõusoleku loomise protsessi etapid ja päringud. LHV on ette valmistanud kaks kasutatavat *Bearer token*'it, millega on võimalik nõusolek luua ja järgnevaid päringuid kontojääkide ja tehingute saamiseks teha. Luminoril seisuga 11.05.2021 OAuth endiselt korrektselt ei toimi ja *token* nõusoleku loomise tarbeks ja edasiste päringute õnnestumiseks tuleb genereerida *developer portal*'is.

Erisusi testkeskkondade ja *live*-vahel on ka edaspidises protsessis. Näiteks ei vasta tehingute vastuseid alati päringute parameetritele – mõningatel juhtudel sisaldavad viimase 90 päeva tehingute päringu vastused paari aasta taguste dateeringutega tehinguid (Citadele) kui ka paariteise aasta taguste dateeringutega tehinguid (Swedbank), kohati on tehingute päringute vastused lihtsalt tühjad listid (Revolut) ja mõnel puhul ei ole hilisemaid kui paari aasta taguseid tehinguid valikusse lisatud (SEB), nii et lõplik rakenduse häälestamine ei saa reaalselt toimuda enne live-keskkonda jõudmist ja päris andmete kasutamist, kuna vastavad vastused testkeskkonnast ei peegelda tegelikkust.

2.10 Rakenduse ehitamise otsuste tegemine

Dokumentatsioonide analüüsi põhjal on saanud selgeks, et Swedbank, Luminor ja dokumentatsiooni järgi ka Coop pank (eraisikuna ja ilma vastava makseteenuse pakkuja sertifikaadita ei ole kahjuks selle panga *sandbox*'i vastu ühenduste loomine võimalik) on oma nõusoleku loomise voo poolest üsna sarnased. Nendest pisut erinev on SEB. Teine grupp omavahel sarnase vooga pankasid on LHV ja Citadele. Ka Šiaulių panga puhul on võimalik kasutada globaalset nõusolekut, mis teeks voo sarnaseks Citadele'ga, kuid lisaks Citadele's kasutusel olevale on Šiaulių panga puhul vaja ka lisaks nõusoleku loomise päringut QSEAL sertifikaadiga allkirjastada, millist teiste pankade ühenduste puhul üldse ei kasutata. Revoluti *Open Banking UK* regulatsiooni järgi jõustatud ühendus aga omab täiesti teistlaadset lähenemist.

Kuna erinevate ühenduste puhul on näha mõningaid sarnaseid jooni, siis on eeldatavalt võimalik luua osaliselt ühtsed baasmeetodid, mida saab vastavalt erisustele ja konfiguratsioonidele kasutada. Peale nõusoleku loomist on erinevatel teostustel rohkem ühisusi kontojääkide ja tehingute pärimisel, seega selles osas peaks olema lihtsam ühtsust saavutada.

Rakenduse siseselt salvestatakse nõusoleku andmed üldisesse vormi ja kuvatakse kliendile ühtselt välja. Kontojääkide ja tehingute päringute tulemuste kuvamiseks kasutatakse samuti kõigi erinevate pankade puhul ühtset väljundit, mille tarbeks kaardistatakse vastustest saadud väljad sarnasesse vormistusse.

3 Teostus

Prototüübi loomise protsess koosnes erinevatest etappidest, millega liiguti sammhaaval terviklikuma lahenduse poole.

3.1 Tehnoloogiad

Rakendust on arendatud C# ja .NET Core tehnoloogiate baasil. Antud tehnoloogiline valik on tehtud töö autori pädevusi silmas pidades. Mõne teise programmeerimiskeele vajaliku mahuga selgeks tegemine oleks olnud liiga ajakulukas. Lisaks on C# rangelt tüübitud objektorienteeritud keel, mis pakub rakenduse konfigureerimiseks kasutajasõbralikke vahendeid.

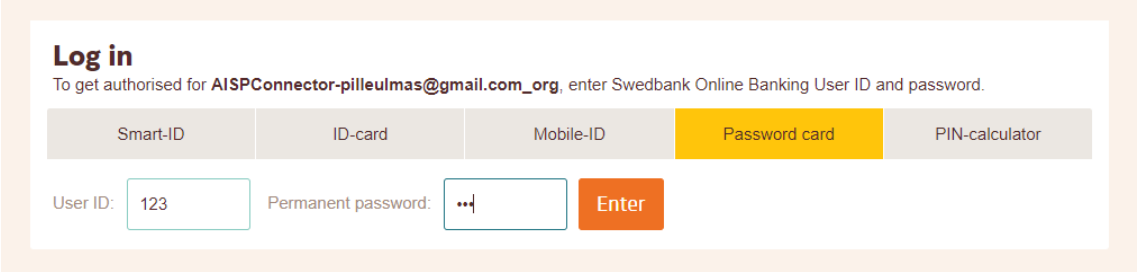
Rakendus teeb päringuid erinevate REST API'de suunas ning suhtluskihiks on HTTP. Arendusprotsessi kiirendamiseks ja prototüübi sujuvaks testimiseks on rakendus seadistatud kasutama mälupõhist andmebaasi (*in-memory database*).

3.2 Arenduse protsess

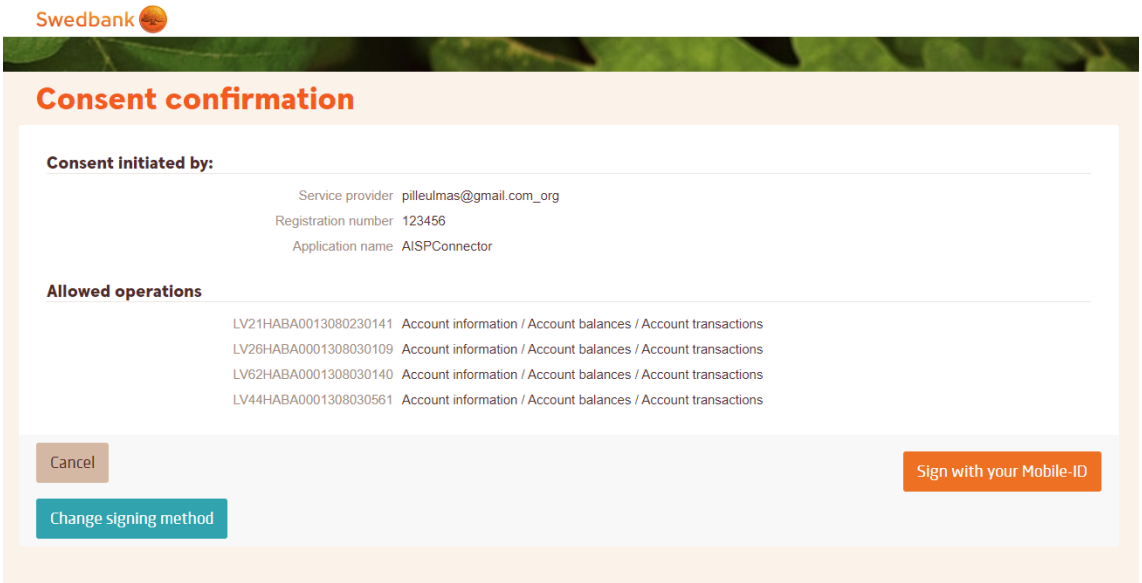
Prototüübi valmimise protsessis liiguti etappide kaupa. Esmalt tehti põhjalik dokumentatsioonide analüüs ja *Postman*'i kollektsioonidega ning erinevate pankade poolt arendatud *sandbox*- lahenduste poolt pakutava veebipõhise päringusüsteemi kaudu (*swagger*) päringute proovimine. Järgmiseks valmis pankhaaval konsoolirakendus, mille manuaalse testimise käigus said teostuseks valitud pankade *sandbox*'ide ühendused järele proovitud ja läbi testitud nende toimimine, sertifikaatide kasutamine ja nendega allkirjastamine ning andmete API'dest kätte saamine ja deserialiseerimine.

Peale õnnestunud konsoolirakendust ja selle abil edukat manuaalset voogude teostust/testimist loodi esialgne veebirakendus, kus kaasatulevate andmete valimine on mõnevõrra dünaamilisem võrreldes eelneva konsoolirakendusega. Rakenduse ehitamise käigus sai päringutest tulevate vastuste andmestruktuure võimalikul määral üldistatud kuid lõpuni see siiski võimalik ei ole seoses pankade vaheliste erisustega. Esialgses

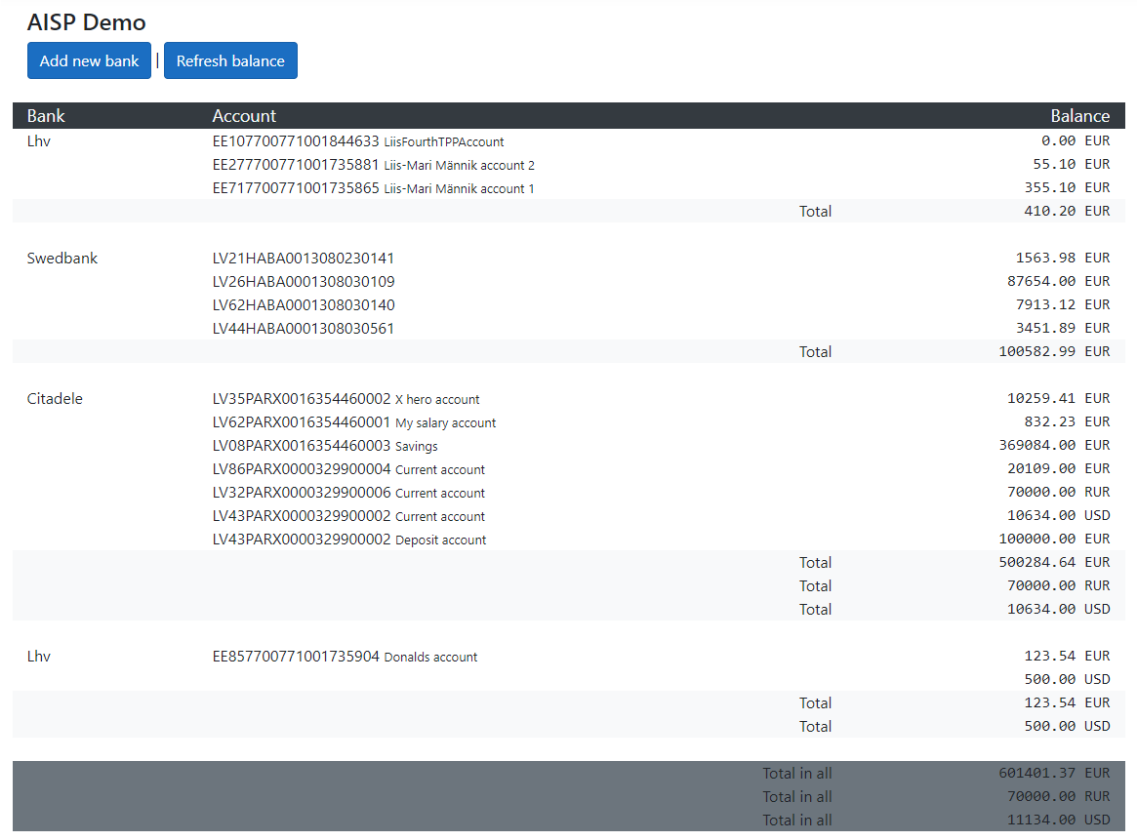
rakenduses on tegelemata jäetud tehingute nimekirjaga, kuigi vastavad päringud on läbi proovitud, vastused *sandbox*'idest kätte saadud ja ka deserialiseeritud vastavateks objektideks. Aeg-ajalt esineb päringute vastuste kättesaamise juures ka partnerite poolseid tõrkeid ja ootamatuid veateateid kuid testkeskkondade olemasolu on suureks abiks rakenduse arendamisel ja päringute korrektsuse valideerimisel ja integratsiooni testimisel.



Joonis 15. Autentimine Swedbank'i *sandbox*'is



Joonis 16. Autoriseerimine Swedbank'i *sandbox*'is



Joonis 17. Kuvatõmmis prototüübi kontojääkide vaatest

Esialgsel prototüübil sai lahendatud kontojääkide kuvamine ja nende valuutapõhiselt kokku arvutamine. Edasi arendamise võimalusi on sellise teenuse juures mitmeid. Kontojääkide põhiselt on näiteks võimalik koostada graafikuid, mis kuvavad visuaalselt erinevate kontode osakaalu kogusummast. Ka tehingute nimekirja kuvamiseks on mitmeid võimalusi. Näiteks tavapärase tehingute väljavõtte kujul kasutaja soovide järgi erinevatel moodustel tehinguid filtreerides või siis ka tehingute põhjal ajaloolist kontojääki või igapäevaseid väljaminekuid kalkuleerides graafikuid visualiseerides.

4 Tulemused

Töö tulemusel valmis kontoteabe teenuse prototüüp, mille ühendused on loodud kättesaadavate *sandbox*'ide vastu. Kuna mitme panga testkeskkonnas on olulisi kitsendusi ja puudujääke võrreldes *live*-ühendustega. siis need on rakenduse lähtekoodis nähtavalt välja toodud ja kirja pandud. Prototüübi loomise käigus sai selgeks, et päris ühtset protsessi ei ole võimalik kõigi pankade ühenduste jaoks tekitada, kuna erisusi erinevates töövoogudes on liiga palju. Diplomitöö kaitsmisperioodil saab prototüüpi testida aadressil https://aisp-proto.azurewebsites.net/.

Analüüsi ja integratsioonide jõustamise käigus sai selgeks, et üldiselt integratsiooni protsessi lihtsustamiseks oleks vaja tunduvalt parandada API'de dokumentatsioone ja mõningatel juhtudel ka *sandbox*'ide päringuid ja vastuseid. Hilisemat *live*-ühenduse loomist teeks hõlpsamaks see kui testkeskkonnad vastaksid maksimaalselt võimalikult produktsioonile ja kõik erinevad kasutusel olevad API päringud oleks võimalik läbi proovida ja reaalsusele vastavad vastused saada. Vastav nõue testkeskkondadele on küll PSD2 direktiivis olemas, kuid selle täitmine puudulik.

5 Kokkuvõte

Töö eesmärgiks oli lahendada erinevate pankade PSD2 API'de erisusi läbi kontoteabe teenuse prototüübi loomise. Prototüübi loomise käigus selgus, et ühise platvormi tekitamiseks on vaja kõik erisused läbi töötada ja neid võimalikult universaalselt/ühtselt teostada. Analüüsi ja teostuse jooksul jõuti järeldusele, et ühtset lahendust kõigi proovitud pankadega rakenduse liidestamiseks ei ole, kuid peale mõningast erinevate dokumentatsioonide läbi töötamist ja integratsioonide tegemist on võimalik märgata sarnaseid vooge ja neid ka osaliselt taaskasutada ning sellega uute ühenduste tekitamise protsessi kiirendada.

Töö tulemusel valmis *sandbox*'ide vastu ühendatud prototüüp, mida on võimalik vastava litsentsi olemasolul ka *live*-keskkonda edasi arendada ning selle põhjal üles ehitada rakendus, milles on võimalik oma erinevates pankades olevat kontoteavet lihtsal moel kuvada või siis sellele lisa funktsionaalsusi juurde ehitada. Kui erinevate pankade ühendused on juba loodud ja andmete jaoks ühine formaat tekitatud, siis on võimalus pakkuda ka API'dega ühendamise teenust. Sarnaseid teenusepakkujaid on turul küll juba praegugi olemas, kuid teenuse hinnatase on suhteliselt kõrge ja sellise teenusepakkuja kasutamisega jäädakse ilma võimalusest teenust ise edasi arendada või selle abil saadud andmete muul moel kasutamisest ilma lisakulutusteta. Seega on võimaluse ja oskuste olemasolul siiski kasumlikum vastav teenus ise välja töötada.

Töö käigus loodud prototüübi näidiskood võib olla abiks alustavale kontoteabe teenuse pakkujale, kuna esialgne erinevates formaatides dokumentatsioonides orienteerumine on üsna keerukas ning vaid ühe näidise põhjal tehtud ühendus võib anda ebaselge ja kallutatud pildi tegelikkusest. Erinevate teenusepakkujate ühise agregaatori näidis on suureks abiks reaalse projekti loomisel.

Kasutatud kirjandus

[1] Uus seadus toob kaasa täiendavaid nõudeid makseteenuse pakkujatele [Võrgumaterjal]. Loetud aadressil: https://www.fi.ee/et/uudised/uus-seadus-toob-kaasa-taiendavaid-noudeid-makseteenuse-pakkujatele. Kasutatud 28.04.2021.

[2] Läbivaadatud eeskirjad makseteenuste kohta ELis [Võrgumaterjal]. Loetud aadressil: https://eur-lex.europa.eu/legal-content/ET/ALL/?uri=LEGISSUM%3A2404020302_1. Kasutatud 28.04.2021.

[3] Implementation by EU countries [Võrgumaterjal]. Loetud aadressil: https://ec.europa.eu/info/law/payment-services-psd-2-directive-eu-2015-2366/implementation/implementation-eu-countries_en. Kasutatud 28.04.2021.

[4] Mis on avatud pangandus? [Võrgumaterjal]. Loetud aadressil: https://pangaliit.ee/arveldused/avatud-pangandus. Kasutatud 28.04.2021.

[5] Payment Services Directive 2 and Open Banking [Võrgumaterjal]. Loetud aadressil: https://www.ukfinance.org.uk/guidance/payment-services-directive-2-and-open-banking. Kasutatud 28.04.2021.

[6] About [Võrgumaterjal]. Loetud aadressil: https://www.berlin-group.org/. Kasutatud 28.04.2021.

[7] NextGenPSD2 XS2A Framework Implementation Guidelines [Võrgumaterjal]. Loetud aadressil: https://77cb457b-3353-4bdc-8ab6-ff6bb2ccdc98.filesusr.com/ugd/c2914b_a7164685fc584703abe39faf60542040.pdf. Kasutatud 28.04.2021.

[8] Welcome to the Open Banking Standard [Võrgumaterjal]. Loetud aadressil: https://standards.openbanking.org.uk/. Kasutatud 28.04.2021.

[9] Makseasutuste ja e-raha asutuste seadus [Võrgumaterjal] Loetud aadressil: https://www.riigiteataja.ee/akt/MERAS. Kasutatud 28.04.2021.

[10] SEB Developer Portal [Võrgumaterjal]. Loetud aadressil:
https://developer.baltics.sebgroup.com/landing. Kasutatud 28.04.2021.

[11] Swedbank Open Banking [Võrgumaterjal]. Loetud aadressil:
https://www.swedbank.com/openbanking.html. Kasutatud 28.04.2021.

[12] Coop Pank API documentation [Võrgumaterjal]. Loetud aadressil:
https://openbanking.cooppank.ee/documentation. Kasutatud 28.04.2021.

[13] Luminor developer portal [Võrgumaterjal]. Loetud aadressil:
https://developer.luminoropenbanking.com/#/. Kasutatud 28.04.2021.

[14] LHV Open banking [Võrgumaterjal]. Loetud aadressil:
https://www.lhv.ee/en/open-banking. Kasutatud 28.04.2021.

[15] Developers portal [Võrgumaterjal]. Loetud aadressil:
https://developer.citadele.lv/en/portal/. Kasutatud 28.04.2021.

[16] Šiaulių bankas Open banking [Võrgumaterjal]. Loetud aadressil:
https://openbanking.siauliubankas.lt/. Kasutatud 28.04.2021.

[17] Introduction to the Open Banking API [Võrgumaterjal]. Loetud aadressil:
https://developer.revolut.com/docs/build-banking-apps/#introduction-to-the-open-banking-api. Kasutatud 28.04.2021.

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Pille Ulmas

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „PSD2 API'de erisuste lahendamine kontoteabe teenuse väljatöötamisel", mille juhendaja on Andres Käver
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

17.05.2021

1 Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 – Koodinäidis: Swedbank'i detailse nõusoleku loomine

```
public static async Task<DetailedConsent> GetDetailedConsentAsync(HttpClient client,
string bearer,string bic, ICollection<App.BLL.Swed.DTO.Account> accounts,string
oauthRedirectUrl)
{
    // add all accounts into consent request
    var body = new DetailedConsentRequest();
    foreach (var account in accounts)
    {
        var iban = new AccountReference {Iban = account.Iban};
        body.Access.Accounts.Add(iban);
        body.Access.Balances.Add(iban);
        body.Access.Transactions.Add(iban);
    }

    string requestQuery = "?bic=" + bic;
    var httpRequestMessage = new HttpRequestMessage
    {
        Method = HttpMethod.Post,
        RequestUri = new Uri("https://psd2.api.swedbank.com:443/sandbox/v3/consents" +
requestQuery),
        Headers =
        {
            {HttpRequestHeader.Authorization.ToString(), "Bearer " + bearer},
            {HttpRequestHeader.Accept.ToString(), "application/json"},
            {"TPP-Redirect-Preferred", "true"},
            {"TPP-Redirect-URI", oauthRedirectUrl},
            {"X-Request-ID", Guid.NewGuid().ToString()},
            {"PSU-IP-Address", "1.2.3.4"},
            {"PSU-User-Agent", "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36"},
            {"Date", DateTime.Now.ToString("r")},
        },

        Content = new StringContent(JsonSerializer.Serialize(body))
    };
    httpRequestMessage.Content.Headers.ContentType = new
MediaTypeHeaderValue("application/json");

    var response = await client.SendAsync(httpRequestMessage);

    var content = await response.Content.ReadAsStringAsync();
    var detailedConsent =
        JsonSerializer.Deserialize<DetailedConsent>(content);

    return detailedConsent!;
}
```

Lisa 3 – Koodinäidis: Swedbank'i kontojäägi päring

```
public static async Task<AccountBalances> GetBalancesAsync(HttpClient client, string bearer, string bic, string consentId, string resourceId)
{
    string requestQuery = "?bic="+bic;
    var httpRequestMessage = new HttpRequestMessage
    {
        Method = HttpMethod.Get,
        RequestUri = new Uri("https://psd2.api.swedbank.com:443/sandbox/v3/accounts/"+resourceId+"/balances"+requestQuery),
        Headers =
        {
            {HttpRequestHeader.Authorization.ToString(), "Bearer " + bearer},
            {"X-Request-ID", Guid.NewGuid().ToString()},
            {"Consent-ID", consentId},
            {"PSU-IP-Address", "1.2.3.4"},
            {"PSU-IP-Port", "80"},
            {"PSU-User-Agent", "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36"},
            {"PSU-Http-Method", "GET"},
            {"Date", DateTime.Now.ToString("r")},
                }
    };

    var response = await client.SendAsync(httpRequestMessage);
    var content = await response.Content.ReadAsStringAsync();
    var balance = JsonSerializer.Deserialize<AccountBalances>(content);

    return balance!;
}
```

Lisa 4 – Koodinäidis: Swedbank'i kontojäägi andmestruktuur

```
public class AccountBalances
{
    [JsonPropertyName("account")] public AccountReference? AccountReference { get; set;
}

    [JsonPropertyName("balances")] public ICollection<Balance>? Balances { get; set; }

}

public class AccountReference
{
    [JsonPropertyName("iban")] public string Iban { get; set; } = default!;
}

public class Balance
{
    [JsonPropertyName("balanceType")] public string BalanceType { get; set; } =
default!;

    [JsonPropertyName("balanceAmount")] public Amount BalanceAmount { get; set; } =
default!;

    [JsonPropertyName("referenceDate")] public string? ReferenceDate { get; set; }
}

public class Amount
{
    [JsonPropertyName("currency")] public string Currency { get; set; } = default!;

    [JsonPropertyName("amount")] public string AmountOfMoney { get; set; } = default!;
}
```

Lisa 5 – Esialgse prototüübi ERD-skeem