OHJEMISTOSUUNNITTELUDOKUMENTTI

Matkalippusovellus

Tekijät: Otto Karpoja, Matias Laukka, Irina Salonen, Konsta Turunen, Ville Vierros

Versiohallinta:

Versio	Päiväys	Tehdyt muutokset	Hyväksytty
0.1	20.1.2021	Dokumenttipohja	20.1.2021 /KLa
1.0	18.3.2021	Dokumentin täydennys kohdasta 3.3	18.3.2021 / KTu
1.1	25.3.2021	3.1.1-kohtaa täydennetty	19.4.2021/ISa
1.2	30.3.2021	Dokumentin tarkistus ja täydennys tiimissä, kohdat 1-3	19.4.2021/ISa
1.3	8.4.2021	4.2.1 työstö aloitettu: alustava tilakaavio lisätty, yms.	19.4.2021/ISa
1.4	21.04.2021	2.1 Lisätty tekstiä ja 4 osioon lisätty moduuleita.	22.4.2021/KTu
1.5	22.4.2021	4.1 ja kohdan 1 työstämistä	22.4.2021/KTu
1.6	29.4.2021	Dokumentin täydennystä tiimissä	4.5.2021/ISa
1.7	3.5.2021	4.5, 6.0 ja 7.0 Lisätty puuttuvia tietoja	4.5.2021/ISa
1.8	4.5.2021	Dokumentin viimeistely	4.5.2021/ISa

Sisällys

1. JOHDANTO

1.1 Tarkoitus ja kattavuus

Suunnitteludokumentti liittyy Laurea ammattikorkeakoulun tietojenkäsittelyn opintoihin ja opintojaksoon Ohjelmistotuotteen määrittely ja suunnittelu. Dokumentin tuottamiseen ovat osallistuneet ryhmän 3 (matkalippusovellus) jäsenet, johon kuuluvat Otto Karpoja, Matias Laukka, Irina Salonen, Konsta Turunen ja Ville Vierros. Jokaisella ryhmäläisellä on ollut oma vastuualueensa raportin kirjoittamisessa, mutta suunnitteludokumenttia on työstetty yhteisesti myös ryhmän tapaamisissa. Ryhmätyötä on tehty Microsoft Teams sovelluksessa jaettuna Word dokumenttina käyttäen apuna määrittelyvaiheen dokumentteja

Tämän dokumentin on tarkoitus ohjata ja opastaa ohjelmoijaa luomaan suunnittelu- ja määrittelydokumentin mukaisen Matkalippusovelluksen. Määrittelydokumentissa kohdan 7. Suunnittelurajoitteiden mukaisesti versioon 1.0 pyritään toteuttamaan seuraavat vaatimukset:

- Sovellus toimii pilvipalvelussa
- Sovelluksella on mahdollista ostaa lippu (TV1)
- Sovelluksella on mahdollista tarkastella aikatauluja (TV6)
- Sovellus tukee karttapalvelua (TV10)
- Sovellukseen ei tarvitse rekisteröityä (TV9)
- Ostaminen tulee vaatimaan luottokortin
- Sovellukseen on mahdollista kirjautua (TV4)
    - Sallii maksutietojen tallentamisen
    - Sovelluksella voi poistaa omat asiakastiedot
- Tietoturva riittävä (ETV2)
    - Sovelluksen kehittämisessä noudatetaan yleisiä, hyväksi havaittuja käytäntöjä tietoturvan toteutumiseksi
- Tuettu Android 8, iOS 13.7 ja uudemmat (ETV3)
    - Aloitetaan kehitys käyttöjärjestelmien uusimmista versioista ja varmistetaan yhteensopivuus taaksepäin myöhemmissä versioissa
- Valuutta euro (ETV4)
- Asiakkaalta veloitetaan aina euroja, jolloin asiakas vastaa valuuttaheilahtelujen vaikutuksesta hintoihin
- Lipuntarkastajan käyttöliittymää ei ole tarjolla ensimmäisessä versiossa
- Sovellus voidaan tarjota rajatulle joukolle käyttökokemustestaukseen ensimmäisessä vaiheessa, kunnes lipuntarkastajan käyttöliittymä valmistuu myöhemmin.
- Sovelluksella voi lähettää matkalipun toiselle henkilölle

Versio 1.0 – vaatimusten ominaisuuksien tarkemmat kuvaukset vaatimusluettelossa. Suunnittelun avuksi toimitetaan määrittely- ja vaatimusmäärittelydokumentti.

1.2 Tuote ja ympäristö

Tässä suunnitteludokumentissa kuvataan matkalippusovellusta, joka toimii joukkoliikenteen asiakkaan lipunoston tukena, sekä tarjoaa sovelluksen käyttäjälle muita hyödyllisiä palveluja. Matkalippusovellus on suunniteltu siten että se mahdollistaa käyttäjälle julkisten kulkuvälineiden aikataulujen, reittien ja mahdollisten häiriötilanteiden tarkastelun reaaliaikaisesti. Sovellusta käyttävä asiakas voi tallentaa sovellukseen eri maksuvälineitä.

Tuote on matkalippusovellus, jonka käyttäjä voi ladata puhelimeen AppStore tai Google Play sovelluskaupasta. Matkalippusovelluksella käyttäjä voi ostaa erilaisia matkalippuja (aikuinen, opiskelija, päivän lippu, yhden kerran lippu) joukkoliikennevälineisiin ja osoittaa lippujensa ostohistorian. Matkalippusovelluksen tavoitteena on tarjota joukkoliikenneasiakkaan matkalipun ostoon sekä matkareittien selailuun liittyvän helppokäyttöisen palvelun. Lisäksi matkalippusovellus tarjoaa palvelun ylläpitäjille helpon käyttöympäristön hallita palvelun käyttämiä tietoja.

Palvelu on suunniteltu toteutettavaksi mobiililaitteille, joten sitä pitää pystyä käyttämään liikkuessa ilman häiriöitä.

1.3 Määritelmät, merkintätavat ja lyhenteet

Amazon Amplify Flutter: Toimii sovelluksen kehitysympäristönä. https://aws.amazon.com/blogs/aws/amplify-flutter-is-now-generally-available-build-beautiful-cross-platform-apps/

Dart: Toimii ohjelmointikielen kääntäjänä. https://en.wikipedia.org/wiki/Dart_(programming_language)

Flutter: Toimii sovelluksen kehitysalustana https://aws.amazon.com/blogs/aws/amplify-flutter-is-now-generally-available-build-beautiful-cross-platform-apps/

NAP: Liikkumispalveluiden tietojen avoin yhteyspiste, josta saadaan palveluun esimerkiksi aikataulut ja reittitiedot. https://finap.fi/#/

1.4 Viitteet

Tässä suunnitteludokumentissa tullaan viittamaan ulkoisiin dokumentteihin, jotka ovat listattu alla. Nämä alla olevat dokumentit toimitetaan erillisinä tiedostoina tämän dokumentin yhteydessä.

- Käyttötapausluettelo versio 1.5, 10.3.2021
- Määrittelydokumentti versio 1.6, 10.3.2021
- Vaatimusluettelo versio 1.4, 9.3.2021

2. JÄRJESTELMÄN YLEISKUVAUS

Sovelluksen tarkoitus on mahdollistaa erilaisten joukkoliikenteessä matkustamiseen liittyvien suoritteiden tekeminen ennen joukkoliikennevälineeseen nousemista. Palvelu on suunniteltu ensimmäisessä versiossa joukkoliikenteen asiakkaan ja myöhemmin myös lipuntarkastajan tarpeet huomioon ottaen, eli palvelu mahdollistaa reittien ja aikataulujen tarkastelun, matkalipun ostamisen, sekä tarvittaessa ostohistorian tarkastelun ostettujen matkalippujen käytön jälkeen. Palvelun käyttäjäryhmä on siis laaja, joten palvelun suunnittelussa pitää ottaa huomioon palvelun esteettömyys.

2.1 Sovellusalueen kuvaus

Matkalippusovellus toimii älypuhelin ympäristössä, jossa tarjotaan käyttäjälle mahdollisuutta kätevästi ja yksinkertaisesti ostaa lippu moniin eli julkisen liikenteen kulkuvälineisiin. Sovellus on saatavilla IOS ja Android puhelimille ja käyttää puhelimissa olevaa GPS ja tietoliikenneyhteyksiä ja mahdollisesti käyttäjästä riippuen myös tunnistautumisessa sormenjälkitunnistinta.

Matkalippusovellus pyörii Amazonin pilvipalvelussa, joka tunnetaan nimellä Amazon Web Services eli AWS. AWS ylläpitää sovellukselle tärkeää tietokantaa.

2.2 Järjestelmän liittyminen ympäristöönsä

Mobiilisovellus on käytössä asiakkaalla ja myöhemmässä vaiheessa lipuntarkastajalla. Sovellus toimii Android 8, iOS 13 ja niitä uudemmissa käyttöjärjestelmissä. Sovellus yhdistää asiakkaalle tietoja Matkalippusovellustietokannasta ja API-rajapintojen välityksellä, sekä mahdollistaa myöhemmissä versioissa lipuntarkastuksen. Palvelun ylläpitäjällä on pääsy tietokantoihin ja muuttamaan, tai lisäämään API-rajapintoja AWS:n avulla. Palvelun ylläpitäjällä voi olla eri rooleja, joilla on eri oikeudet toimintoihin.

- Lipputietokanta (hakee lipputyypin ja hinnat)
- Asiakasrekisteri (hakee asiakkaan tiedot)
- Open Street Map -karttapalvelu (karttanäkymä)
- NAP-liikkumispalvelukatalogi (Eri liikennöitsijöiden reittien haku API-rajapintaa hyödyntäen)
- MapBox kokonaisreittien lisäämiseksi kartalle
- Nets maksupalveluntarjoajana
- Pilvipalveluyhteydet tietokannan ylläpitoon ja muuten ylläpitäjän käyttöön
- Sovellus ladattavissa Google Play ja AppStore
- Puhelimen JTN
    - GPS
    - Sormenjälkitunnistin
    - Tietoliikenneyhteydet

2.3 Laitteistoympäristö

Matkalippusovelluksen laitteistoympäristönä toimii Android tai IOS puhelinkäyttöjärjestelmät. Matkalippusovellus tukee Android 8, iOS 13.7 ja sitä uudempia käyttöjärjestelmien versioita. Sovelluksen kehitysympäristönä toimii Android 10 ja iOS 13 versiot. Sovellus voidaan ladata, joko Google Play tai AppStoresta. Sovellus tarvitsee puhelimesta GPS yhteyden sovelluksen karttapalveluiden toiminnallisuuden vuoksi ja tietoliikenneyhteyden internettiin, jotta sovellus voi lähettää internet protokollan eli IP:n avulla olla yhteydessä MongoDB:llä toteutettuihin tietokantoihin, joita hostaa ulkoinen palveluntarjoaja Amazon Web Services.

Ylläpitäjä voi hallita sovellusta ja MongoDB-tietokantoja Amazon Web Services -pilvipalvelun kautta. AWS on saatavilla selaimen välityksellä. Ylläpitäjän rooleja ovat asiakaspalvelu ja admin eli järjestelmän valvoja. Asiakaspalvelijat voivat ylläpitäjän käyttöliittymästä hallinnoida matkalippusovellustietokantaa rajoitetuin oikeuksin. Järjestelmänvalvoja voi taas lisätä ja poistaa esimerkiksi tietoja tietokannasta. Tulevissa versioissa matkalippusovellusta voidaan moderoida ylläpitäjän käyttöliittymästä, joka toimii Google Chrome ja Safari -selaimissa.

Matkalippu sovellus on myös yhteydessä ulkoisiin rajapintoihin, kuten NAP, Open Street Map, Nets ja Facebook/Google tunnistautuminen. Ulkoisiin rajapintoihin sovellus on yhteydessä internetin IP protokollan avulla ja toimii wifi tai mobiili tietoliikenne yhteyksillä.

2.4 Ohjelmistoympäristö

Kääntäjä	Dart	2.12.2
Tietokantaohjelmisto	MongoDB	3.6
WWW-selain	Firefox Chrome Safari	86.0.1 89.0.4389.90 14.0.3
Karttaohjelmisto	Mapbox OpenStreetMap	v2.2.0-beta.1 v0.6
Liikennetiedot	mmtis-national-access-point	v2.09
Käyttöjärjestelmä	Android, iOS	v10, v13.7
Pilvipalvelut	Amazon Web Services	
Kehitysympäristö	Amazon Amplify-Flutter	V0.1.0
Kehitysalusta	Flutter	1.21
Kirjautumisintegraatio	OAuth	2.0
Maksupalvelu	Nets	

2.5 Toteutuksen keskeiset reunaehdot

Käyttäjän laitteistona toimii käyttöjärjestelmänään Android 8 ja iOS 13.7 tai uudemmat käyttävät mobiililaitteet. Ylläpitäjä voi hoitaa tehtäviään ylläpitäjän käyttöliittymästä, joka on ensimmäisessä versiossa toteutettu Amazon Web Services -ylläpitonäkymän kautta. Myöhemmin ylläpitoa varten saatetaan rakentaa oma webbipohjainen ylläpitonäkymä, jota voi käyttää Safari tai Chrome –selaimilla. Sovellus tulostaa kolmea kieltä - suomea, englantia ja ruotsia, joista suomi on käytössä alustavasti. Ohjelmistokielenä toimii Dart, kehitysalustana Flutter ja kehitysympäristönä Amazon Amplify-Flutter. Ohjelmiston toimintaa tuetaan MongoDB-tietokantaohjelmistolla ja Amazon Web Services –pilvipalvelulla.

2.6 Sopimukset ja standardit

Henkilötietoja sisältävää tietokantaa ylläpidettäessä on noudatettava EU:n yleistä tietosuoja-asetusta (GDPR).

Lisäksi määrityksiin mahdollisesti vaikuttavat sopimukset:

- Maksupalvelu / Nets
- AppStore
- Google PlayStore
- Liikennöitsijät / NAP
- AWS

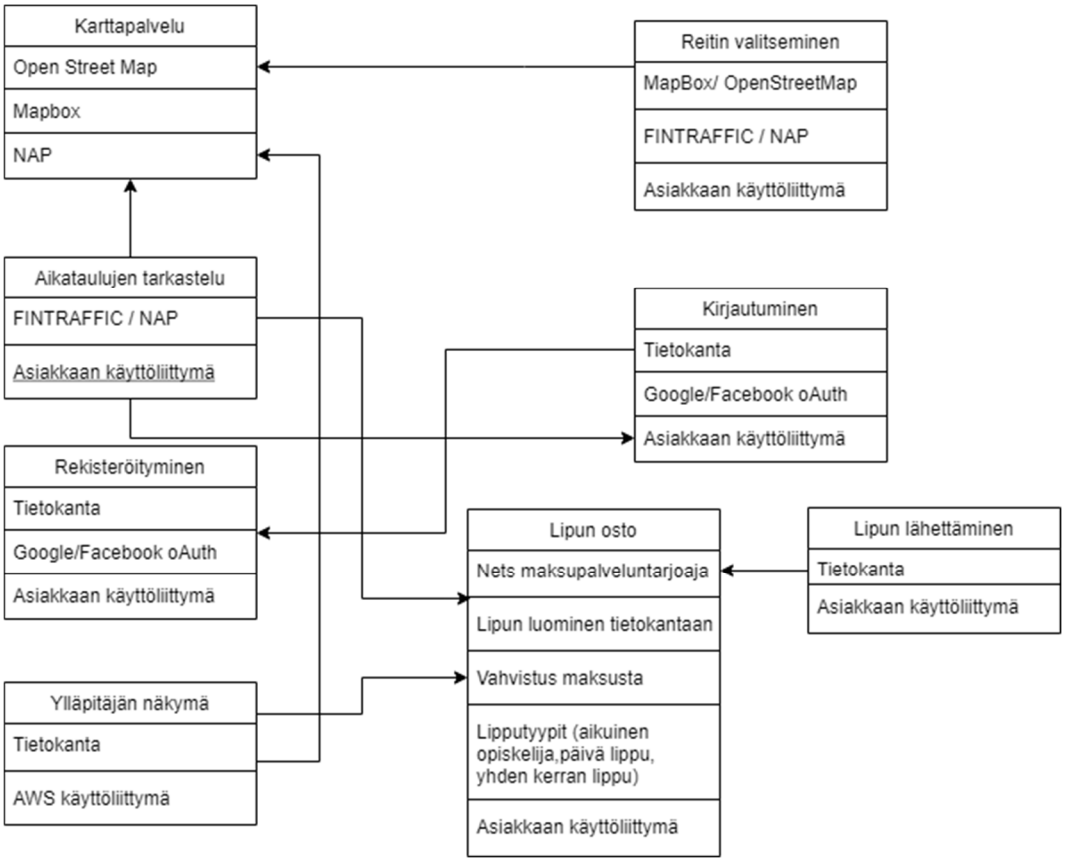
3. ARKKITEHTUURIN KUVAUS

3.1 Suunnitteluperiaatteet

Suunnittelussa on pyritty minimoimaan ylläpitäjän työtä. Karttapalvelu saadaan Mapboxin avulla, karttana toimii OpenStreetMap. Liikennöitsijöiden tiedot, kuten reitit, saadaan NAP-palvelun API-rajapintojen kautta. Maksupalveluntarjoajaksi on valittu Nets ja tunnistautuminen mahdollistetaan OAuth-rajapinnan avulla. Käytössä on Amazon Web Servicesin pilvipalvelu ja MongoDB-tietokanta. Tarkemmat kuvaukset ulkoisten toimijoiden komponenteista löytyy Suunnitteludokumentista kohdasta 5.

3.2 Ohjelmistoarkkitehtuuri, moduulit ja prosessit

Tässä kuvataan moduulit sisältöineen, sekä eri moduulien väliset suhteet.



Reitin valitseminen

- MapBox / OpenStreetMap
- FINTRAFFIC / NAP
- Asiakkaan käyttöliittymä
- Missä menee raja reitin valitsemisen ja lipun oston välillä?

Lipun osto

- Maksupalveluntarjoaja Nets (vain korttimaksu)
- Lipun luominen tietokantaan
- Vahvistus maksusta
- Lipputyypit (aikuinen, opiskelija, päivän lippu, yhden kerran lippu)
- Käyttäjän näkymä

Kirjautuminen

- Tietokanta (asiakasrekisteri)
- Google/Facebook kirjautuminen
- Käyttäjän näkymä

Aikataulujen tarkastelu

- Käyttäjän näkymä
- FINTRAFFIC / NAP

Rekisteröityminen

- Käyttäjän näkymä
- Google/Facebook puuttuvan tiedon tarkastelu (jos puutuu esimerkiksi kaupunki)
- Tunnuksen luominen nimi kaupunki sähköposti salasana TAI Google/Facebook

Karttapalvelu

- Open Street Map
- Mapbox (reitti sis. jalankulku)
- NAP – julkisen kulkuvälineen reitti

Lipun lähettäminen

- Tietokanta (lippu) (onko relevantti tieto, että kenellä lippu on?)
- Käyttäjän näkymä?

Ylläpitäjän näkymä

- Asiakaspalvelu
    - Rajatut oikeudet tietokantaan
        - Asiakasrekisteri
        - Virhelokien seuraaminen
        - Asiakastietojen poisto
    - Yhteys tekniseen tukeen
        - NAP
        - MapBox
        - OpenStreetMap

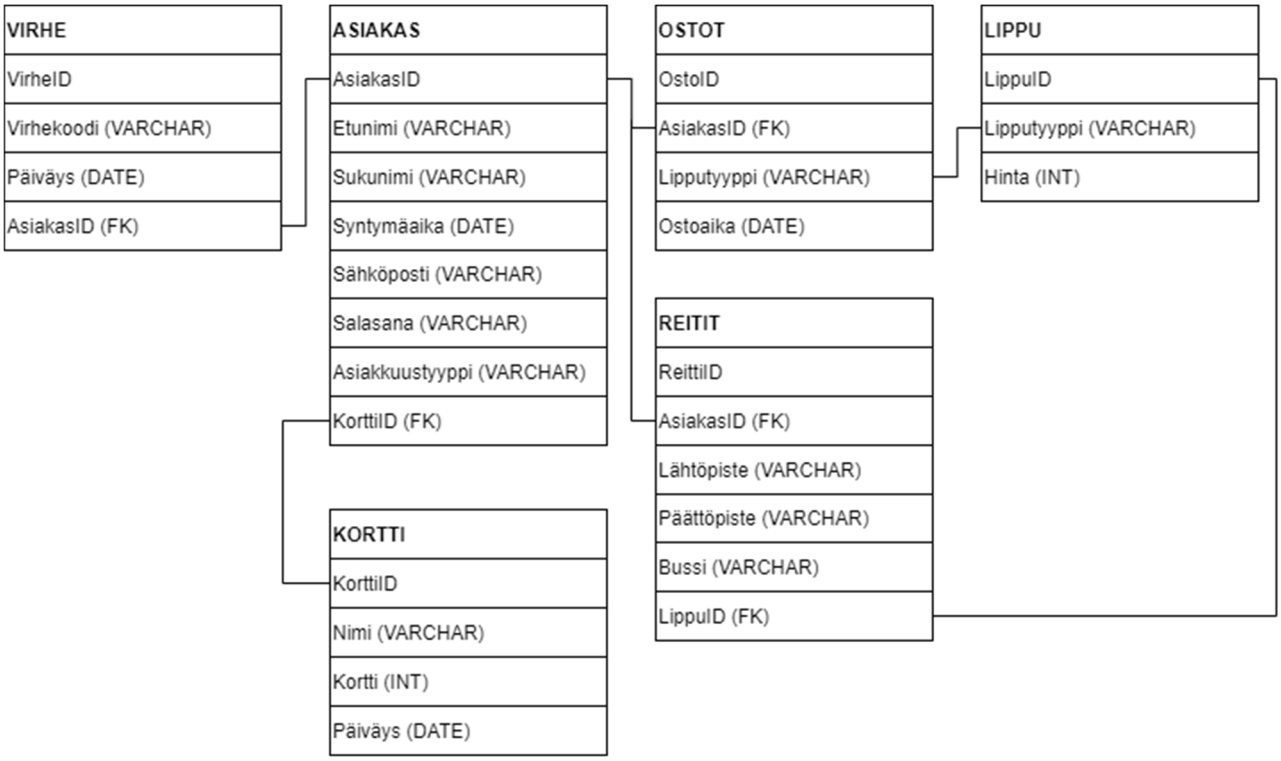
- Admin / järjestelmänvalvoja
    - Tietokanta
        - Asiakasrekisteri
        - Hintojen muokkaaminen
        - Lipputyyppien muokkaaminen
        - Asiakastyyppien muokkaaminen
        - Virhelokien seuraaminen
    - NAP / MAPBOX
        - Uusien reittien päivittäminen NAP:sta
        - MapBox:n päivittäminen

3.3 Tietokanta-arkkitehtuuri

Tietokantamme on toteutettu MongoDB:llä. MongoDB ei ole relaatiotietokanta vaan dokumenttitietokanta eli käytännössä se on rakenteeltaan hieman erilainen kuin esimerkiksi MySQL-tietokanta. MongoDB:n ominaisuuksista huolimatta kuvaamme tietokannan toteutuksen tauluilla, jotta sitä olisi yksinkertaisempi ymmärtää.

MongoDB-tietokantamme on vahvasti yhteydessä NAP:n API-palveluun, josta saamme helposti eri julkisen liikenteen palveluntarjoajien bussiyhteydet, aikataulut, hinnat ym. Ohjelmistomme NAP-yhteyden takia emme lähteneet luomaan omaa taulukkoa reittiaikatauluja varten.

Yksikään ryhmän jäsen ei valitettavasti ollut työskennellyt MongoDB:n kanssa ennen tätä kurssia, joten ymmärryksemme dokumenttitietokannoista oli puutteellinen. Projektin laajuuden takia oli mahdotonta perehtyä kaikkiin aiheen yksityiskohtiin, mutta teimme parhaamme ja toivomme, että seuraava kuvaus tietokannan rakenteesta antaa riittävän hyvän kuvan siitä, millaista rakennetta lähdimme suunnitteluvaiheessa hakemaan.



Lipputietokannassa on kuusi taulua, joissa kaikissa on molemmin suuntainen suhde.

- **Asiakas**
  - AsiakasID (PK)
  - Etunimi (VARCHAR 20)
  - Sukunimi (VARCHAR 20)
  - Syntymäaika (DATE)
  - Sähköposti (VARCHAR 50)
  - Salasana (VARCHAR 20)
  - Asiakkuustyyppi (VARCHAR 1)
  - KorttiID (FK)
- **Kortti**
  - KorttiID (PK)
  - Nimi (VARCHAR 40)
  - Kortti (INT 16)
  - Päiväys (DATE)
- **Ostot**
  - OstoID (PK)
  - AsiakasID (FK)
  - Lipputyyppi (VARCHAR 1)
  - Ostoaika (DATE)
- **Lippu**
  - LippuID (PK)
  - Lipputyyppi (VARCHAR 1)
  - Hinta (NUMBERDECIMAL 6)
- **Reitit**
  - ReittiID (PK)
  - AsiakasID (FK)
  - Lähtöpiste (VARCHAR 20)
  - Päättöpiste (VARCHAR 20)
  - Bussi (VARCHAR 5)
  - LippuID (FK)
- **Virhe**
  - VirheID (PK)
  - Virhekoodi (INT 3)
  - Päiväys (DATE)
  - AsiakasID (FK)

Tietokannan Asiakas-taulukko säilyttää tärkeintä käyttäjätietoa kuten kirjautumiseen tarvittavaa sähköpostia / salasanaa. Lisäksi taulukko sisältää muun yleisen käyttäjätiedon kuten käyttäjän nimen, syntymäajan, asiakkuustyypin ym. Asiakas-taulukko on myös yhteydessä Kortti-taulukkoon, joka säilyttää asiakkaan korttitietoja.

Asiakkaan etsiessä reittiä sen tiedot haetaan NAP:n API:sta. Asiakkaalla on myös mahdollisuus sisään kirjauduttuaan tallentaa usein käytettyjä reittejä. Reitit-taulukko säilyttää näitä reittejä ja on yhteydessä Asiakas-tauluun, jotta ohjelmisto osaa yhdistää asiakkaan omiin reitteihinsä. Tallennetut reitit koostuvat yksinkertaisesti osoitteista, joiden avulla ohjelmisto hakee NAP:sta mahdollisia yhteyksiä.

Asiakkaan ostaessa lippua ohjelmisto pyytää oikeaa lipputyyppiä asiakkaan asiakkuustyypin mukaan Lippu-taulukosta. Lippu-taulukossa on lista erilaisia lipputyyppejä ja niiden hintoja mm. palveluntarjoajan, reitin sekä asiakkuustyypin mukaan ja ohjelmisto hakee oikeaa lipputyyppiä asiakkaan syöttämän reitin mukaan tietokannasta.

Maksuun siirtyessä ohjelmisto muistaa Asiakas-taulukkoon yhteydessä olevasta Kortti-taulukosta oleelliset maksutiedot. Kortti-taulukko säilyttää ainoastaan maksukortin tietoja ja tämän takia taulukossa on erillinen 'Nimi'-kenttä, sillä kortissa oleva nimi voi joissain tilanteissa olla eri kuin asiakkaan. Tämän kaltaisia tilanteita tulee vastaan mm. lapsen lainatessa vanhemman korttitietoja, mikäli ei omista vielä omaa maksukorttia.

Ostot-taulukko säilyttää listaa ostetuista lipuista. Tämä taulukko säilyttää jokaisen ostetun lipun lipputyypin, ajankohdan ja on yhteydessä Asiakas-tauluun, josta saadaan myös ostajan tiedot. Taulukosta käyttäjä näkee omat aiemmin ostetut liput ja hänellä on myös mahdollisuus poistaa nämä tiedot kuten myös muut asiakastiedot GDPR:n mukaan.

Virhe-taulukko säilyttää lokia virheilmoituksista ongelmanratkaisua varten. Tämä on erityisen tärkeä ominaisuus ylläpitäjälle, sillä käyttäjä voi halutessaan lähettää virheilmoituksen ylläpidolle käsiteltäväksi. Ylläpitäjä voi myös päivittää lippujen hintoja, lipputyyppejä sekä asiakastyyppejä, mikäli niiden toimintaan tulee jonkinlaisia muutoksia.

Osa taulukoiden sisällöistä on mahdollisesti tarpeettomia tässä vaiheessa projektia. Esimerkkejä tästä ovat esim. reitit-taulun 'bussi' ja 'lippuID' -osuudet, joilla ei välttämättä ole vielä 1.0-versiossa käyttöä, mutta päätimme lisää ne tietokantaan joka tapauksessa tulevaisuuden suunnitelmia varten.

3.4 Virhe- ja poikkeusmenettelyt

Virheet tallennetaan automaattisesti lokiin ja niistä saadaan raportti ylläpitäjälle.

Käyttäjälle lähtee kaikista luetelluista virhetilanteista virheilmoitus, joka kertoo virheen syyn yksinkertaisesti. Alla on listattuna virhekoodit ja niistä käyttäjälle näytettävä virheilmoitus. Tämän dokumentin muissa osioissa viitataan virhekoodeihin, joiden selvennykset löytyvät tästä

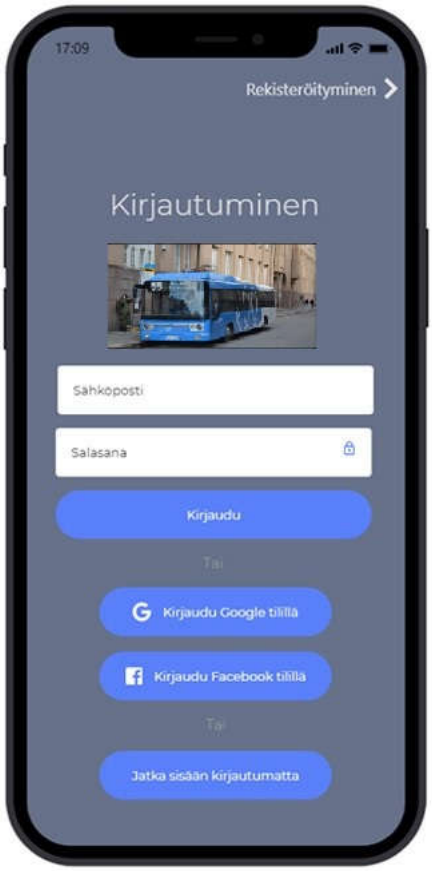
Virhekoodit

- 101 - Internet-yhteys: ”Virhe: Tarkista Internet-yhteys”
- 121 – Karttapalvelut: ”Virhe: Karttapalveluun ei saada yhteyttä”
- 131 - NAP-yhteys: ”Virhe: Liikennöitsijän järjestelmään ei saada yhteyttä”
- 141 – Tietokantayhteys: ”Virhe: Sovelluksen tietokantaan ei saada yhteyttä”
- 151 - Lipun osto: ”Virhe: Matkalipun osto epäonnistui”
- 152 – Nets: ”Virhe: Sovellus ei saanut yhteyttä maksupalveluun”
- 161 – Kirjautuminen: ”Virhe: Virheellinen sähköposti tai salasana”
- 162 – Kirjautuminen: ”Virhe: Sovellus kohtasi virheen ulkoisessa kirjautumispalvelussa”

4. MODUULI /LUOKKA/PROSESSI-KUVAUKSET

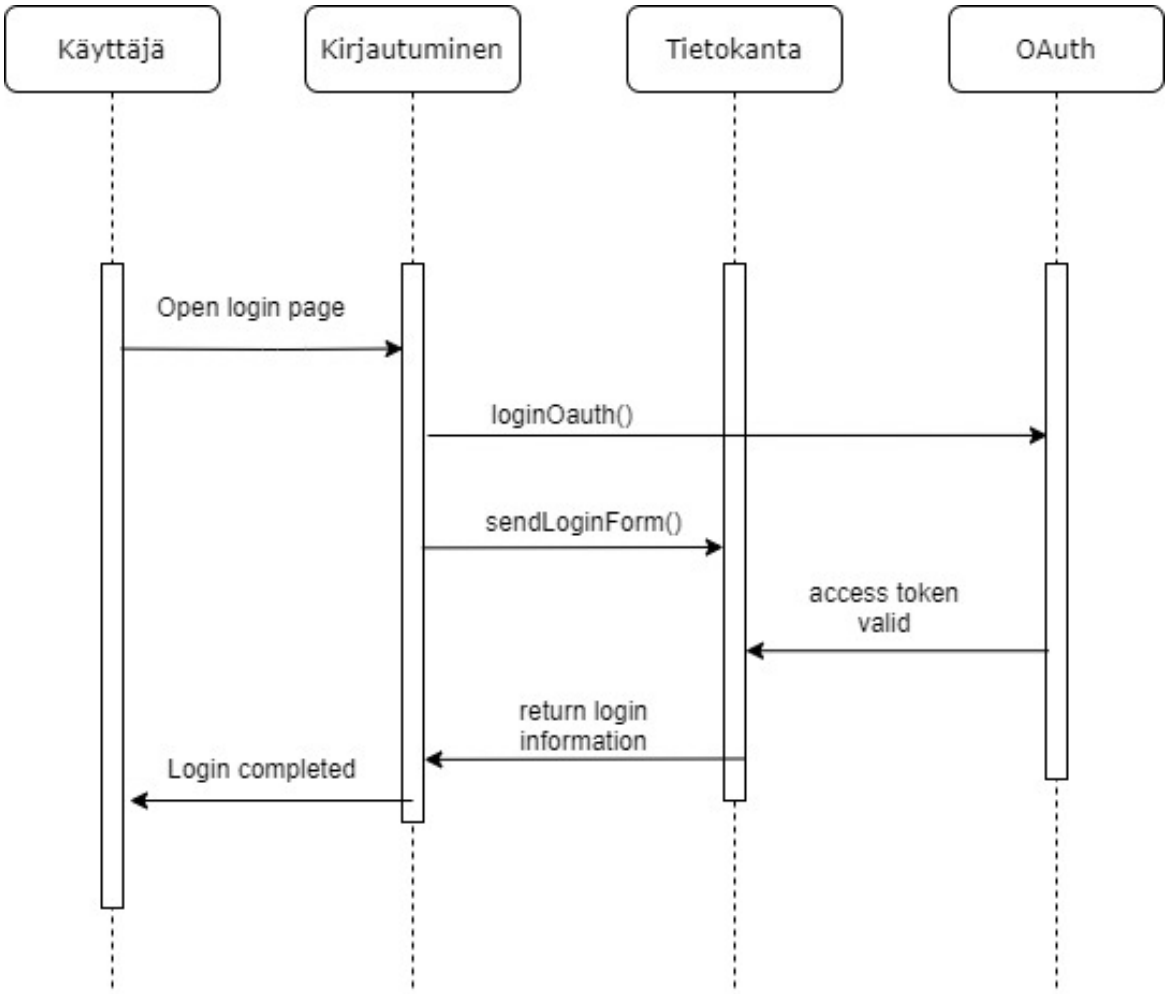
4.1 Kirjautuminen

4. 1.1 Yleiskuvaus



Moduulin nimi:	Kirjautuminen sovellukseen	**Moduulin tyyppi:**	Prosessi
Yleiskuvaus:	Käyttäjä kirjautuu sovellukseen		
Asiakkaat:	Käyttäjän tulee olla rekisteröitynyt sovellukseen, ennen kuin hän voi kirjauta siihen.		
Riippuvuudet ja liitännät:	Moduuli on riippuvainen rekisteröidy-moduulista Kirjautuminen sovellukseen -moduuli vaatii toimiakseen internet-yhteyden, yhteyden tietokantaan. Tarvittaessa sovellus tarvitsee yhteyden ulkopuoliseen palveluun OAuth, jonka kautta mahdollistetaan sovellukseen kirjautuminen ulkoisen palveluntarjoajan kautta (Facebook/Google).		
Suunnittelija ja pvm:	Konsta Turunen 3.5.2021		

4.1.2 Rajapinta yleisesti



Kaavio kuvaa prosessin, jonka käyttäjä käy läpi kirjautuessaan sovellukseen sisälle.

Vaihtoehto 1. Käyttäjä avaa kirjaudu sisään -näytön, jonka jälkeen hän syöttää käyttäjätunnuksen ja salasanan. Tämän jälkeen sovellus tarkistaa edellä mainittujen tietojen vastaavuuden tietokannassa oleviin tietoihin. Jos tiedot ovat oikein, käyttäjälle palautuu tieto siitä ja kirjautuminen on onnistunut.

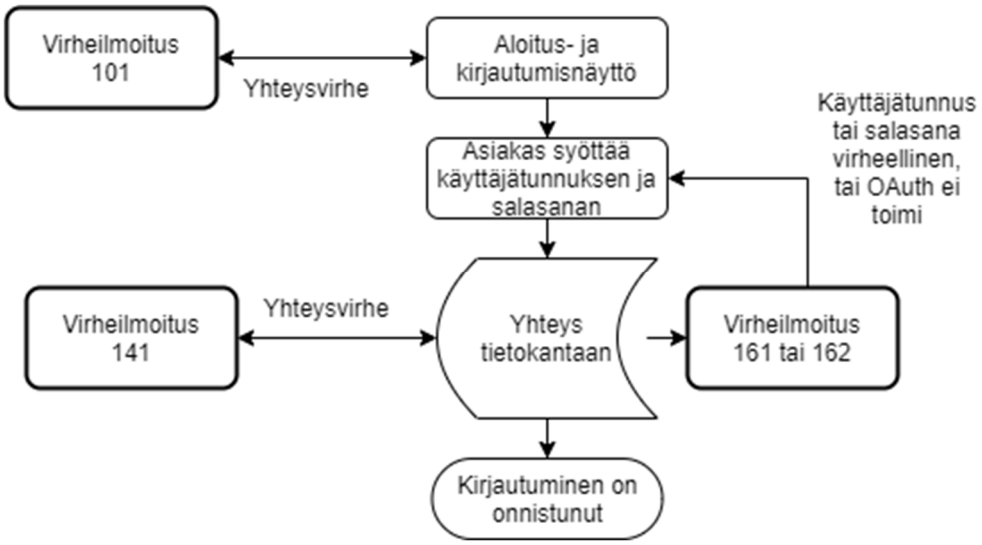
Vaihtoehto 2. Käyttäjä avaa kirjaudu sisään -näytön, jonka jälkeen hän valitsee ulkoisen palvelun kirjautumistavaksi. Hän kirjautuu ulkoiseen palveluun. Tämän jälkeen sovellus tarkistaa ulkoisesta palvelusta tulleiden tietojen vastaavuuden tietokannassa oleviin tietoihin. Jos tiedot ovat oikein, käyttäjälle palautuu tieto siitä ja kirjautuminen on onnistunut.

4.1.3 Rajapintafunktiot

Funktion nimi:	sendLoginForm()	
Toiminto, mitä funktio tekee:	Toiminto vertaa käyttäjän antamia tietoja tietokannassa oleviin tietoihin. Funktio palauttaa tiedon siitä, että vastaako käyttäjän antamat tiedot tietokannassa olevia tietoja	
Funktion parametrit ja paluuarvo:	**Parametri:** • **sähköpostiosoite** • **salasana**	**Paluuarvo:** • **Palauttaa asiakkaan reitin haku -sivulle**
Esiehdot:	Käyttäjän tulee olla rekisteröitynyt palveluun eli tietokantaan on tallentunut etu- ja sukunimi, sähköpostiosoite, kaupunki tai kunta sekä kieli. Hän on myöskin painanut kirjaudu -painiketta, jolloin tarkistetaan tietojen vastaavuus tietokannan tietoihin.	
Jälkiehdot:	Tarkistaa käyttäjätunnuksen ja salasanan tietokannasta, jos tiedot ovat oikein käyttäjä kirjautuu sisään ja siirtyy reittihakuun.	
Virhetilanteet:	Internetyhteys ei toimi, esitetään virhe 101. Yhteys tietokantaan ei toimi esitetään virhe 141. Käyttäjätunnus tai salasana on virheellinen tai puuttuu kokonaan, esitetään virhe 161.	
Suunnittelija ja pvm:	Konsta Turunen 3.5.2021	

Funktion nimi:	loginOauth()	
Toiminto, mitä funktio tekee:	Funktio hakee kirjautumistiedot ulkoisesta järjestelmästä ja vertaa niitä tietokannassa oleviin tietoihin.	
Funktion parametrit ja paluuarvo:	**Parametriä ei ole**	**Paluuarvo:** **Palauttaa asiakkaan reitin haku -sivulle**
Esiehdot:	Käyttäjän tulee olla rekisteröitynyt palveluun eli tietokantaan on tallentunut etu- ja sukunimi, sähköpostiosoite, kaupunki tai kunta sekä kieli. Hän on valinnut kirjautumisen ulkoisessa palvelussa. Nämä tarkistetaan ja niiden vastaavuus tietokannan tietoihin.	
Jälkiehdot:	Jos ulkoiselta palvelun tarjoajalta saapuvat tiedot vastaavat sovelluksen käyttämään tietokantaan käyttäjä kirjautuu sisälle ja siirtyy automaattisesti reittihakuun	
Virhetilanteet:	Internetyhteys ei toimi, esitetään virhe 101. Yhteys tietokantaan ei toimi esitetään virhe 141. Ulkoisen palveluntarjoajan käyttämiseen liittyvä virhe kirjautumisvaiheessa, esitetään virhe 162.	
Suunnittelija ja pvm:	Konsta Turunen 3.5.2021	

4.1.4 Moduulin toteutus



4.1.5 Virhekäsittely

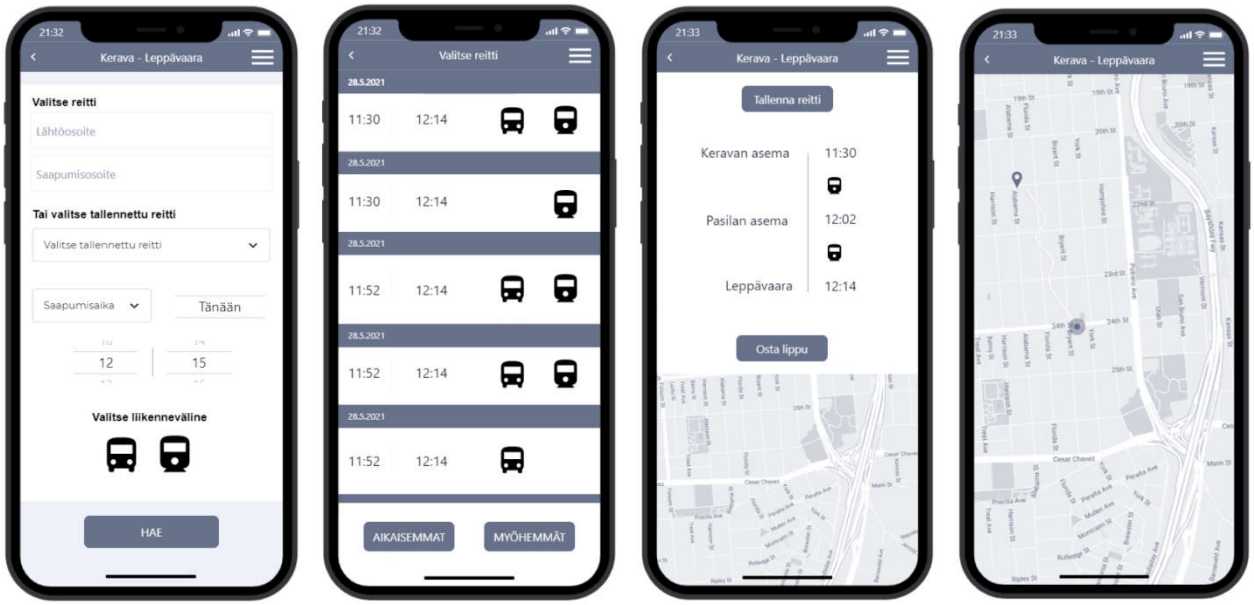
Sovelluksen kirjautumisvaiheessa voi tapahtua useita eri virheitä ja moduulin mukaisesti nämä virheet liittyvät poikkeuksetta joko tietokantaan tai ulkoiseen palveluntarjoajaan. Virhekäsittely tehdään virhekäsittelysuunnitelman mukaisesti. Moduulissa on mahdollista tulla seuraavat virheet:

- 101 Internetyhteys ei toimi
- 141 Tietokantayhteys ei toimi.
- 161 Käyttäjä syöttää väärän käyttäjätunnuksen tai salasanan.
- 162 Ulkoisen palveluntarjoajan käyttämiseen liittyvä virhe kirjautumisvaiheessa

Tarkempi kuvaus Suunnitteludokumentin kohdassa 3.4 Virhe- ja poikkeusmenettelyt

4.2 Reitin valitseminen

Moduulissa Reitin valitseminen kuvataan käyttäjän polku hakuehtojen syötöstä tietyn reitin valitsemiseen. Käyttäjä lisää itse lähtö- ja saapumisosoitteet, tai valitsee ne tallennetuista (vain kirjautuneet). Käyttäjälle esitetään lista mahdollisista reittivaihtoehdoista, joista hän valitsee parhaiten sopivan ja siirtyy tarkastelemaan reittiä. Reitistä annetaan tietona käytetyt liikennevälineet, vaihdot, osoitteet, sekä karttanäkymä, johon reitti piirtyy. Kirjautuneen käyttäjän on myös mahdollista tallentaa valittu reitti, jolloin hän voi jatkossa helposti käyttää sitä ilmoittamatta erikseen lähtö- ja saapumisosoitetta.



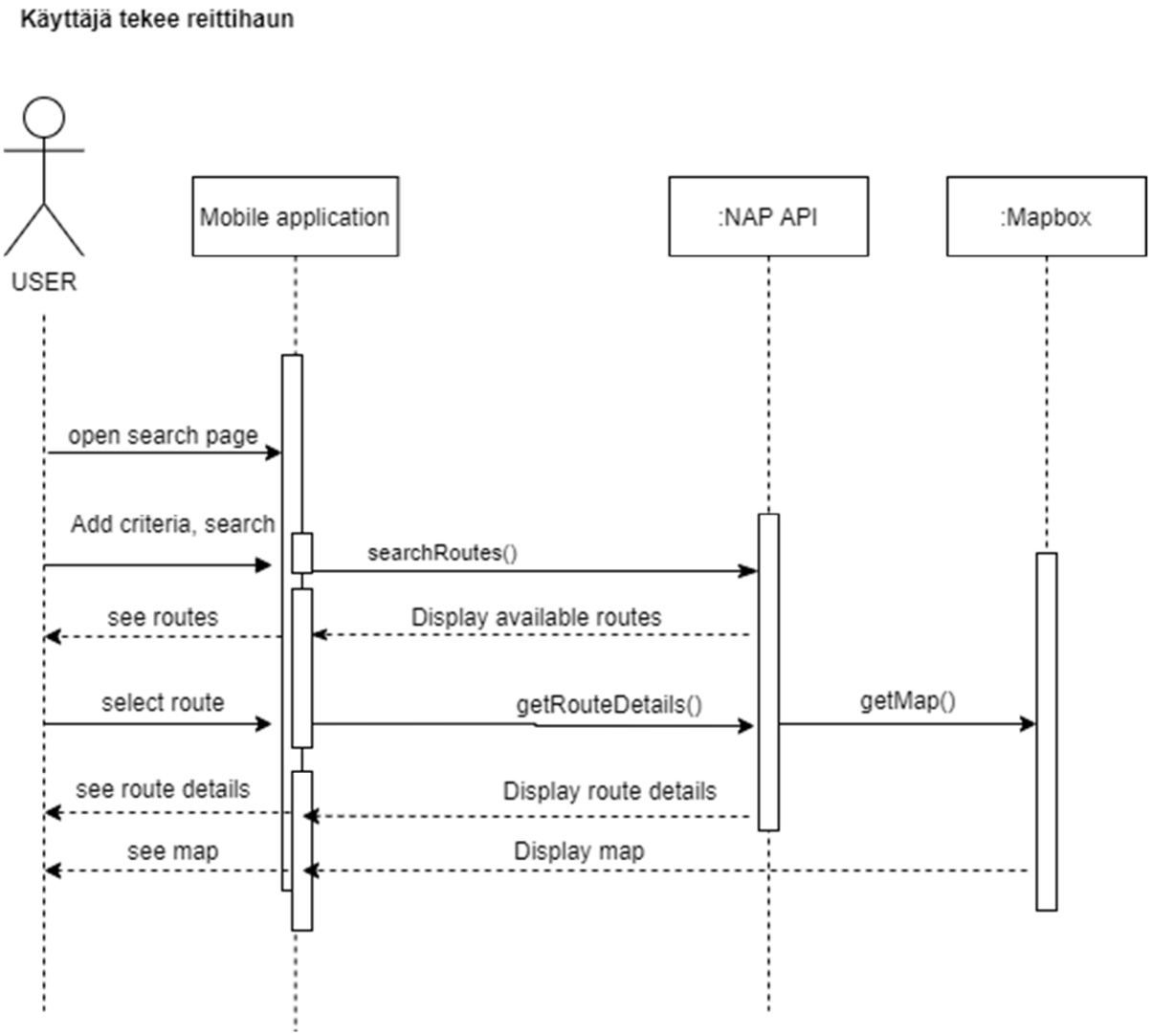
Huom! Mikäli käyttäjä ei ole kirjautunut, ei ruudun 1 ”Valitse tallennettu reitti” -otsikko ja -dropdown näy. Samoin ruudun 3 ”Tallenna reitti” -button ei näy, jos käyttäjä ei ole kirjautunut, tai mikäli kyseinen reitti on jo tallennettu.

4.2.1 Yleiskuvaus

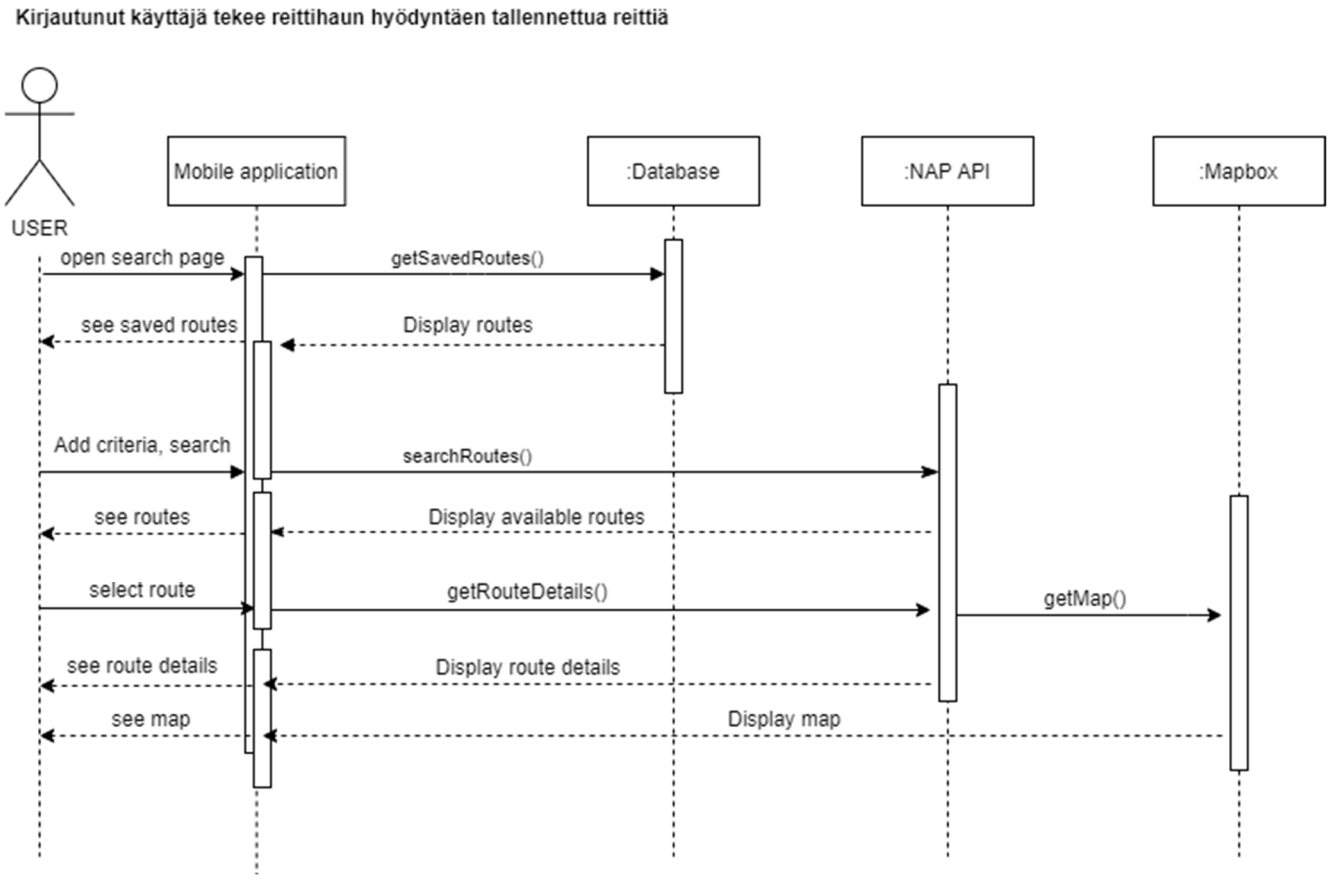
Moduulin nimi:	Reitin valitseminen	**Moduulin tyyppi:**	Prosessi
Yleiskuvaus:	Käyttäjä pystyy valitsemaan sopivan reitin useasta vaihtoehdosta		
Asiakkaat:	Kaikki muut moduulit voivat toimia itsenäisesti, esimerkiksi lipun voi ostaa ilman reitin valitsemista.		
Riippuvuudet ja liitännät:	Toimiakseen Reitin valitseminen -moduuli tarvitsee yhteyden Mapboxin tarjoamaan OpenStreetMapiin, sekä liikennöitsijöiden API-rajapintoihin, joiden kautta kartalle tuodaan reitit.		
Suunnittelija ja pvm:	Irina Salonen 8.4.2021		

4.2.2 Rajapinta yleisesti

Alla on kuvattu Reitin valitseminen -moduulin kaksi polkua sekvenssikaavioina. Virhekäsittely kuvataan kohdan 4.2.3 alla kunkin funktion yhteydessä.



Kaavio kuvaa tilannetta, jossa käyttäjä on reittihakusivulla ja kirjoittaa lähtö- tai saapumisosoitteen ja ajan, sekä klikkaa ”Hae”. Sovellus hakee NAP:n API-rajapintoja hyödyntäen palvaluntarjoajien reitit halutulle matkalle haluttuna ajankohtana ja esittää ne käyttäjälle. Käyttäjä valitsee tietyn reitin, jonka yksityiskohdat haetaan API-rajapinnan kautta palveluntarjoajalta. Samalla haetaan kartta, jolle reitti mallinnetaan. Yksityiskohtaiset reittitiedot vaihtoineen, sekä kartta esitetään käyttäjälle. Käyttäjä voi halutessaan tarkastella vain karttaa klikkaamalla sitä



Kaavio kuvaa tilannetta, jossa sovellukseen kirjautunut käyttäjä on reittihakusivulla ja valitsee ensin tietokantaan tallennetun reitin (lähtö- ja saapumisosoitteet). Sen jälkeen käyttäjän tulee merkitä lähtö- tai saapumisaika ja klikata ”Hae”. Sovellus hakee NAP:n API-rajapintoja hyödyntäen palvaluntarjoajien reitit halutulle matkalle haluttuna ajankohtana ja esittää ne käyttäjälle. Käyttäjä valitsee tietyn reitin, jonka yksityiskohdat haetaan API-rajapinnan kautta palveluntarjoajalta. Samalla haetaan kartta, jolle reitti mallinnetaan. Yksityiskohtaiset reittitiedot vaihtoineen, sekä kartta esitetään käyttäjälle. Käyttäjä voi halutessaan tarkastella vain karttaa klikkaamalla sitä.

4.2.3 Rajapintafunktiot

Funktion nimi:	getSavedRoutes()	
Toiminto, mitä funktio tekee:	Funktio hakee tietokannasta käyttäjän tallentaman reitin (lähtöosoite ja saapumisosoite). Reitti palautetaan sovelluksen hakunäkymään.	
Funktion parametrit ja paluuarvo:	**Parametrit** • AsiakasID	**Paluuarvot** • Lähtöpiste • Päättöpiste
Esiehdot:	Käyttäjän pitää olla reittihakusivulla ja kirjautunut järjestelmään, jotta voi käyttää tallennettuja reittejä.	
Jälkiehdot:	Ohjelma näyttää valitun reitin hakunäkymässä ja käyttäjä voi ajan valittuaan hakea kaikki tarjolla olevat reittivaihtoehdot kyseiselle matkalle.	

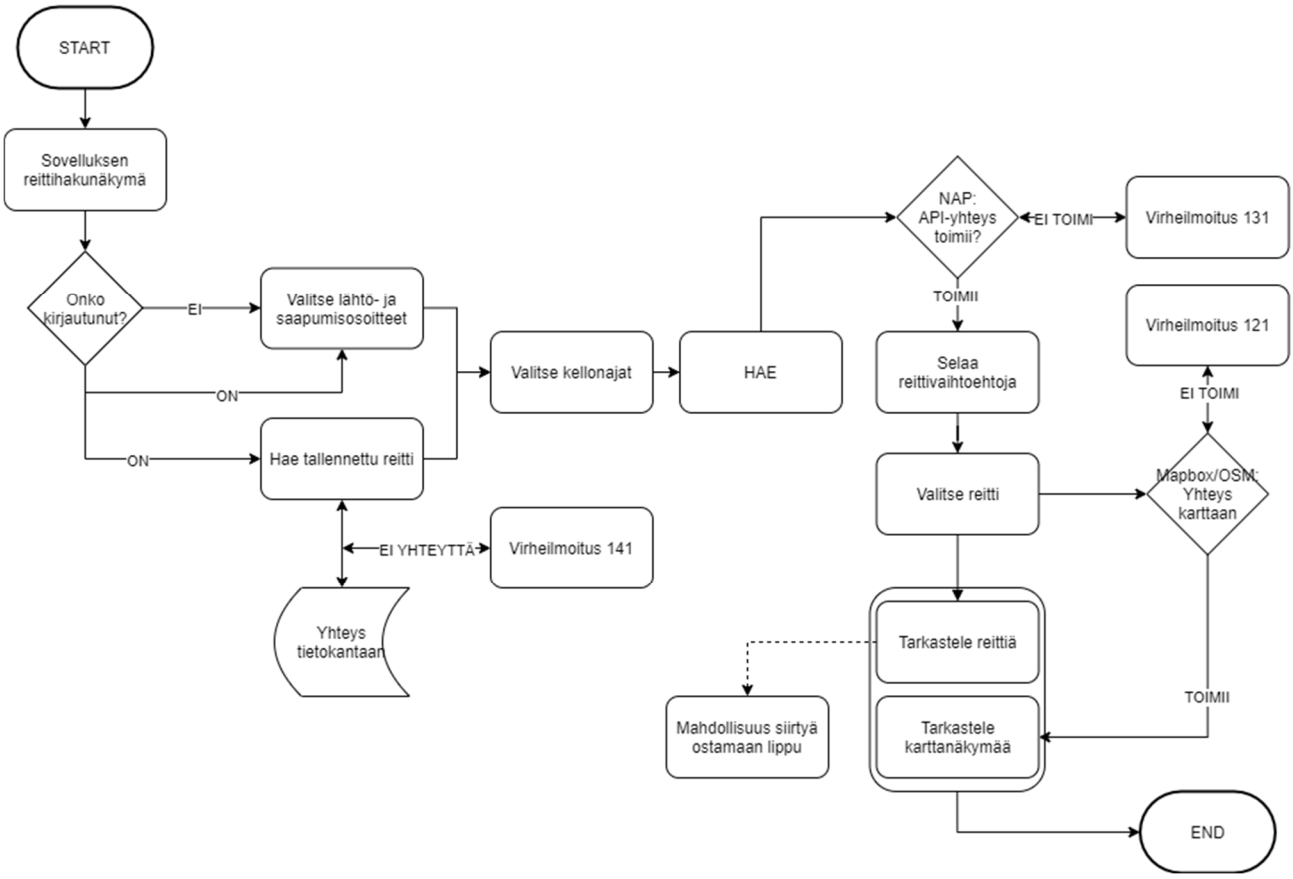
Virhetilanteet:	Jos käyttäjä ei ole kirjautunut sisään, tallennetun reitin valintamahdollisuutta ei näytetä. Jos tietokantaan ei saada yhteyttä, tulee hakutilanteessa virheilmoitus 141.
Suunnittelija ja pvm:	Irina Salonen 28.4.2021

Funktion nimi:	searchRoutes()	
Toiminto, mitä funktio tekee:	Funktio hakee NAP:n API-rajapintojen välityksellä palveluntarjoajien reitit käyttäjän valitsemalle matkalle	
Funktion parametrit ja paluuarvo:	**Parametrit** • Lähtöpiste • Saapumispiste • Lähtö-/saapumisaika	**Paluuarvot** • Saatavilla olevien reittien aikataulut 1h sisällä hakuehdoista
Esiehdot:	Käyttäjän tulee olla valinnut lähtö- ja saapumisosoitteet, sekä saapumis- tai lähtöaika. Käyttäjä on voinut myös valita vain tietyt kulkuvälineet.	
Jälkiehdot:	Ohjelma listaa käyttäjälle reittihakuun sopivat tulokset tunnin aikaväliltä. Ensimmäisenä esitetään parhaiten valittuun aikaan sopiva matka. Klikkaamalla ”hae aiemmat” tai ”hae myöhemmät” käyttäjä voi tehdä uuden haun edellisistä/seuraavista reiteistä. Mikäli valittuna saapumisaika, niin reitit esitetään halutusta saapumisajasta taaksepäin. Mikäli valittuna lähtöaika, niin reitit esitetään halutusta lähtöajasta eteenpäin.	
Virhetilanteet:	Jos API:iin ei saada yhteyttä, käyttäjälle esitetään virhe 131.	
Suunnittelija ja pvm:	Irina Salonen 29.4.2021	

Funktion nimi:	getRouteDetails()	
Toiminto, mitä funktio tekee:	Funktio hakee NAP:n API-rajapinnan välityksellä reitin yksityiskohdat ja esittää ne käyttöliittymässä käyttäjälle.	
Funktion parametrit ja paluuarvo:	**Parametrit:** • Lähtöpiste • Saapumispiste • Lähtö/saapumisaika	**Paluuarvo:** • Tarkka lähtöpiste • Tarkka saapumispiste • Mahdolliset vaihdot • Liikennevälineet • Mahdollisten vaihtojen aikataulut • Tarkka lähtöaika • Tarkka saapumisaika
Esiehdot:	Käyttäjän on pitänyt hakea mahdolliset reitit (searchRoutes() funktio).	
Jälkiehdot:	Ohjelma esittää käyttäjälle tarkan reitin aikatauluineen ja mahdollisine vaihtoineen.	
Virhetilanteet:	Jos API:iin ei saada yhteyttä, käyttäjälle esitetään virhe 131.	
Suunnittelija ja pvm:	Irina Salonen 29.4.2021	

Funktion nimi:	getMap()	
Toiminto, mitä funktio tekee:	Funktio hakee MapBoxin välityksellä käyttöliittymään esitettäväksi kartan, sekä kartalle piirretyn reitin.	
Funktion parametrit ja paluuarvo:	**Parametrit:** • Reitti	**Paluuarvo:** • Kartta • Reitin kuvaus
Esiehdot:	Käyttäjän on pitänyt valita tietty reitti, jonka tarkat tiedot on saatu NAP:n API-rajapinnan kautta.	
Jälkiehdot:	Ohjelma esittää käyttäjälle käyttöliittymässä kartan, jolle on piirretty valittu reitti	
Virhetilanteet:	Jos karttapalveluun ei saada yhteyttä, näytetään käyttäjälle virheilmoitus 121	
Suunnittelija ja pvm:	Irina Salonen 29.4.2021	

4.2.4 Moduulin toteutus



Tilakaavio: Reitin valitseminen. Virheilmoitusten tarkempi kuvaus Suunnitteludokumentin kohdassa 3.4

4.2.5 Virhekäsittely

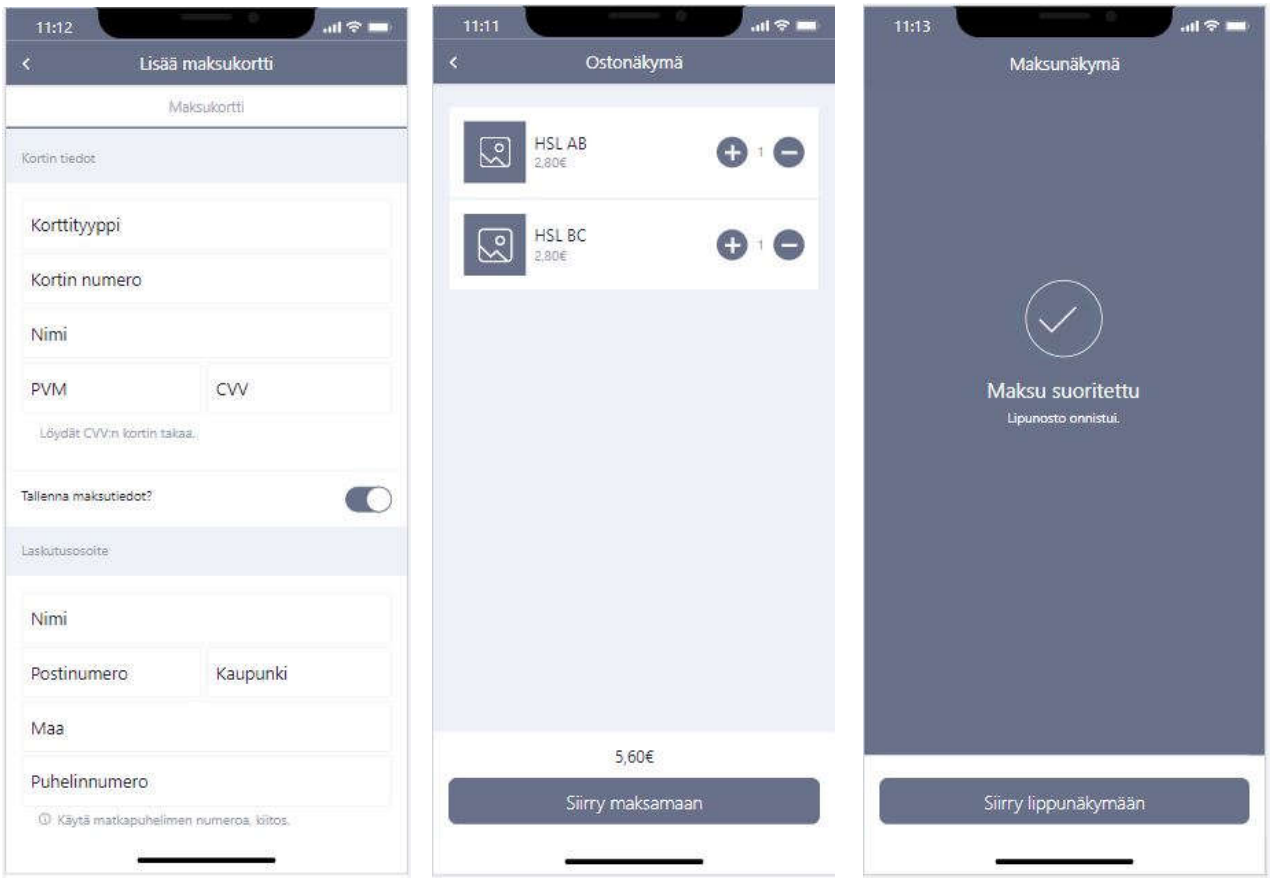
Virhekäsittely tehdään virhekäsittelysuunnitelman mukaisesti. Moduulin mahdolliset virheet (tarkempi kuvaus Suunnitteludokumentin kohdassa 3.4):

- Internet-yhteys 101
- Karttapalvelut 121
- NAP-yhteys 131
- Tietokantayhteys 141

4.3 Lipun ostaminen

Lipun ostaminen –moduuli on kriittinen osa matkalippusovellusta – moduulin kautta asiakas pystyy ostamaan valitun lipun. Moduulin kautta pystytään lisäämään maksukortti, katsomaan valittuja lippuja ja maksamaan lipun ohjaamalla asiakas Nets:n maksupalveluun.

Back-end:ssä lipun ostaminen –moduuli keskustelee lähinnä tietokannan ja Nets:n maksupalvelun kanssa.



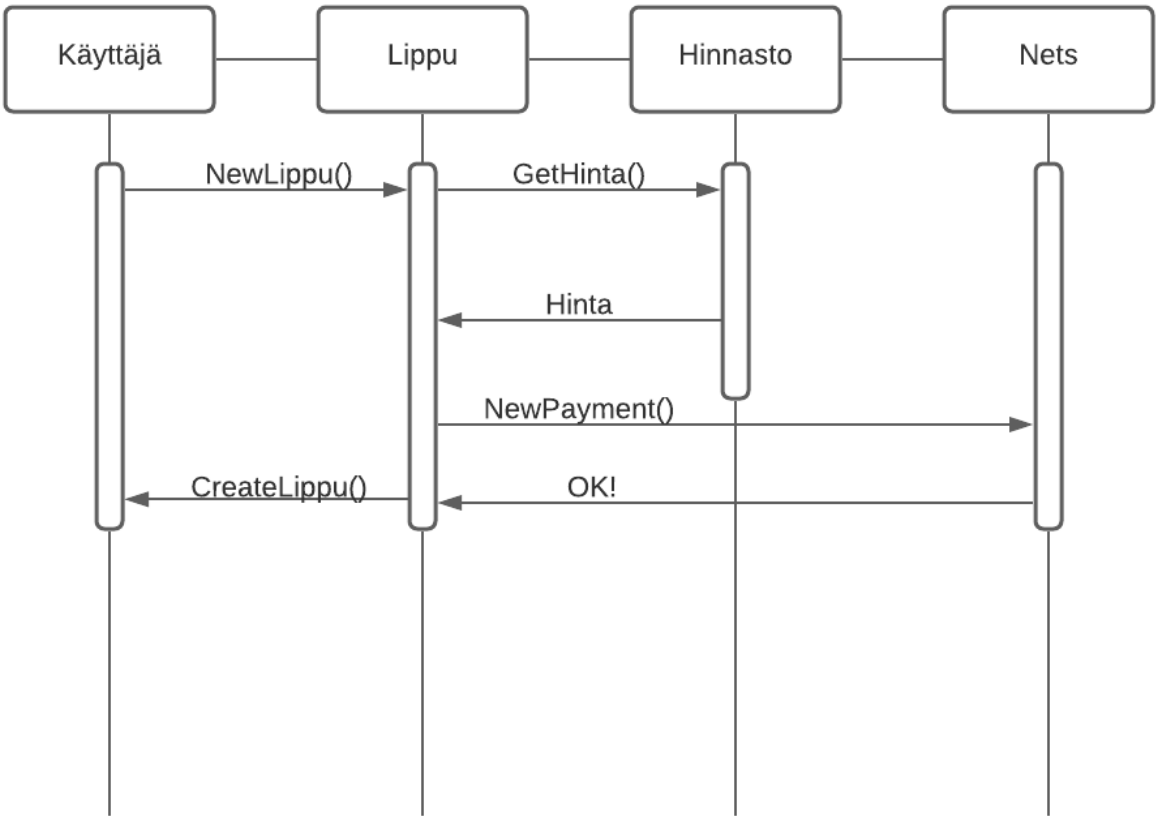
4.3.1 Yleiskuvaus

Moduulin nimi:	Lipun ostaminen	**Moduulin tyyppi:**	Prosessi
Yleiskuvaus:	Asiakas pystyy ostamaan lipun.		
Asiakkaat:	Lipun ostaminen –moduuli seuraa reitin ja lipun valitsemista. Sovellus ei onnistu myymään lippuja ilman tämän moduulin toimintaa.		
Riippuvuudet ja liitännät:	Moduuli on yhteydessä sovelluksen tietokantaan sekä Nets-maksupalveluun. Moduulin yleinen käyttö vaatii myös nettiyhteyttä.		
Suunnittelija ja pvm:	Matias Laukka 14.04.2021		

4.3.2 Rajapinta yleisesti

Alla oleva sekvenssikaavio kuvaa sovelluksen toimintaa back-endissä silloin kun asiakas ostaa lippua. Funktiot tunnistat suluista ja muut linjat ovat palautusarvoja. Alta löydät kuvaukset yksittäisistä funktioista.

Sovellus luo käyttäjälle alustavan lippuobjektin asiakkaan tiedoilla, hakee tietokannasta lipun hinnan ja sen jälkeen yhteyttä Nets:n maksupalveluun korttimaksua varten. Maksun onnistuttua / epäonnistuttua lippuobjekti saa tiedon maksun tilasta. Mikäli maksu epäonnistuu Nets:n ongelman tai matalan tilisaldon takia, asiakas ohjataan ottamaan yhteyttä Nets:iin uudestaan. Maksun onnistuttua lippuobjekti lisää tietokantaan lipun palautusarvot ja asiakas pääsee käyttämään onnistuneesti ostettua lippua.



4.3.3 Rajapintafunktiot

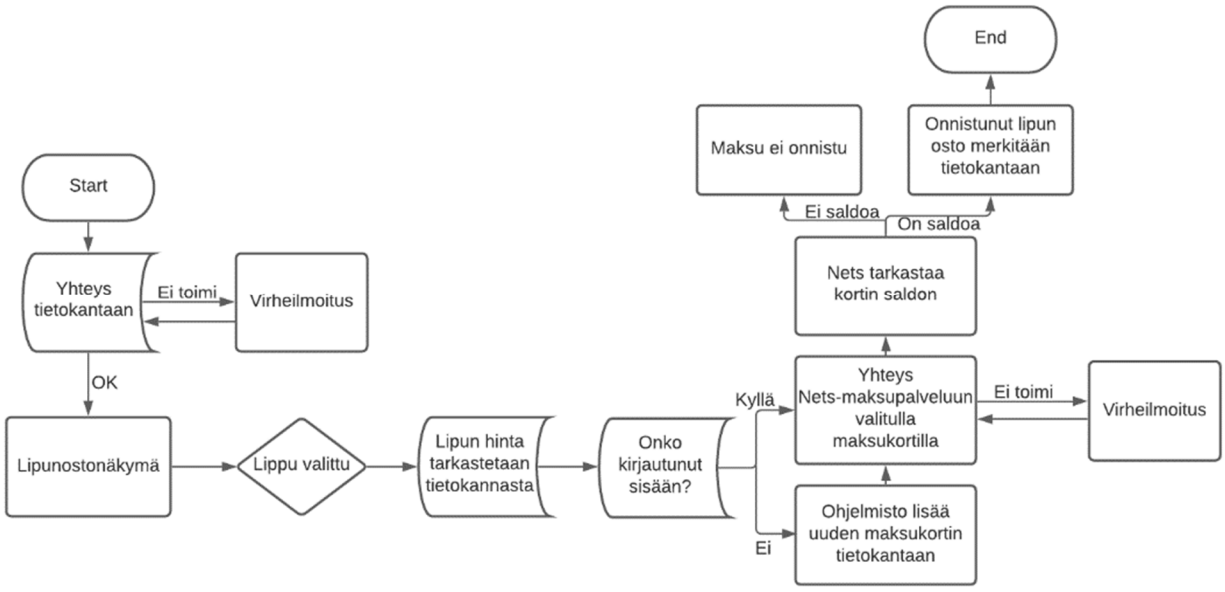
Funktion nimi:	NewLippu()	
Toiminto, mitä funktio tekee:	Ohjelmisto luo käyttäjälle alustavan lippuobjektin.	
Funktion parametrit ja paluuarvo:	**Parametrit** • AsiakasID • Asiakkuustyyppi	**Palautusarvot** • Lipputyyppi • Hinta • Ostoaika • AsiakasID
Esiehdot:	Asiakas on lipunostonäkymässä ja on valinnut lipun.	
Jälkiehdot:	Asiakas on valinnut ja maksanut lipun.	
Virhetilanteet:	Funktio voi keskeytyä, mikäli tietokanta on alhaalla eikä se saa käytettyä tarvittavia parametrejä tai palautusarvoja.	
Suunnittelija ja pvm:	Matias Laukka 29.04.2021	

Funktion nimi:	GetHinta()	
Toiminto, mitä funktio tekee:	NewLippu() -funktion jälkeen lippuobjekti kutsuu lipun hinnan tietokannasta.	
Funktion parametrit ja paluuarvo:	**Parametrit** • AsiakasID • Asiakkuustyyppi • Asiakkaan valitsema lippu	**Palautusarvot** • Lipputyyppi • Hinta • Ostoaika
Esiehdot:	Alustava lippuobjekti on luotu.	
Jälkiehdot:	Lippuobjekti on saanut tietokannasta hinnan lipulle.	
Virhetilanteet:	Funktio voi keskeytyä, mikäli tietokanta on alhaalla eikä se saa käytettyä tarvittavia parametrejä tai palautusarvoja. Funktio voi myös keskeytyä, jos asiakkaalla ei ole nettiyhteyttä.	
Suunnittelija ja pvm:	Matias Laukka 29.04.2021	

Funktion nimi:	NewPayment()	
Toiminto, mitä funktio tekee:	Ohjelmisto yhdistää käyttäjän Nets:n maksupalveluun.	
Funktion parametrit ja paluuarvo:	**Parametrit** • Hinta	**Palautusarvot** • Boolean (*Maksu läpi)*
Esiehdot:	Alustava lippuobjekti on luotu ja se on saanut tietokannasta hinnan. Asiakas on siirtynyt maksuvaiheeseen.	
Jälkiehdot:	Lippuobjekti saa varmistuksen onnistuneesta tai epäonnistuneesta maksusta.	
Virhetilanteet:	Funktio voi keskeytyä, mikäli Nets-maksupalvelu tai tietokanta ovat alhaalla eikä se saa käytettyä tarvittavia parametrejä. Funktio voi myös keskeytyä, jos asiakkaalla ei ole nettiyhteyttä.	
Suunnittelija ja pvm:	Matias Laukka 29.04.2021	

Funktion nimi:	CreateLippu()	
Toiminto, mitä funktio tekee:	Funktio päivittää maksetun lipun tiedot tietokantaan.	
Funktion parametrit ja paluuarvo:	**Parametrit** • AsiakasID • Lipputyyppi	**Palautusarvot** • OstoID • Ostoaika • Lipputyyppi • AsiakasID
Esiehdot:	Alustava lippuobjekti on luotu, se on saanut tietokannasta hinnan ja lippu on maksettu Nets:n kautta.	
Jälkiehdot:	Onnistuneesti ostetun lipun tiedot on päivitetty tietokantaan. Käyttäjä on valmis käyttämään lippua.	
Virhetilanteet:	Funktio voi keskeytyä, mikäli ei saa yhteyttä tietokantaan. Funktio ei aja ollenkaan, jos missään aiemmassa alifunktiossa oli ongelmia esimerkiksi, jos NewPayment() -funktio palauttaa epäonnistuneen maksun. Funktio voi myös keskeytyä, jos asiakkaalla ei ole nettiyhteyttä.	
Suunnittelija ja pvm:	Matias Laukka 29.04.2021	

4.3.4 Moduulin toteutus



4.3.5 Virhekäsittely

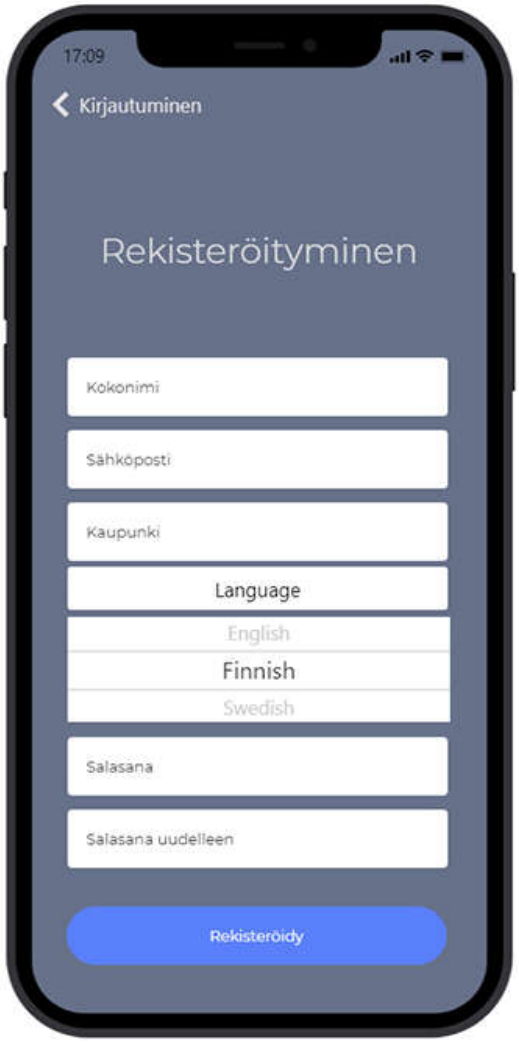
Lipunostovaiheessa voi tapahtua useita ostotilannetta häiritseviä virheitä. Moduulin tehtävän luonteen takia suurin osa virheistä liittyy maksupalveluihin. Virheilmoituksia maksusta voi aiheuttaa kaksi asiaa – ongelma kolmannen osapuolen maksupalvelun kanssa tai ongelma sovelluksen päässä. Sovellus voi epäonnistua maksun suorittamisessa useista syistä, esimerkiksi nettiongelman takia. Pahin uhka lipun ostamiselle on tietokantaongelma, joka estää lähes kaikki sovelluksen toiminnot.

Virhekoodit

- 101 - Internet-yhteys: ”Virhe: Tarkista Internet-yhteys”
- 141 – Tietokantayhteys: ”Virhe: Sovelluksen tietokantaan ei saada yhteyttä”
- 151 - Lipun osto: ”Virhe: Matkalipun osto epäonnistui”
- 152 – Nets: ”Virhe: Sovellus ei saanut yhteyttä maksupalveluun”

(Tarkempi kuvaus Suunnitteludokumentin kohdassa 3.4)

4.4 Rekisteröityminen



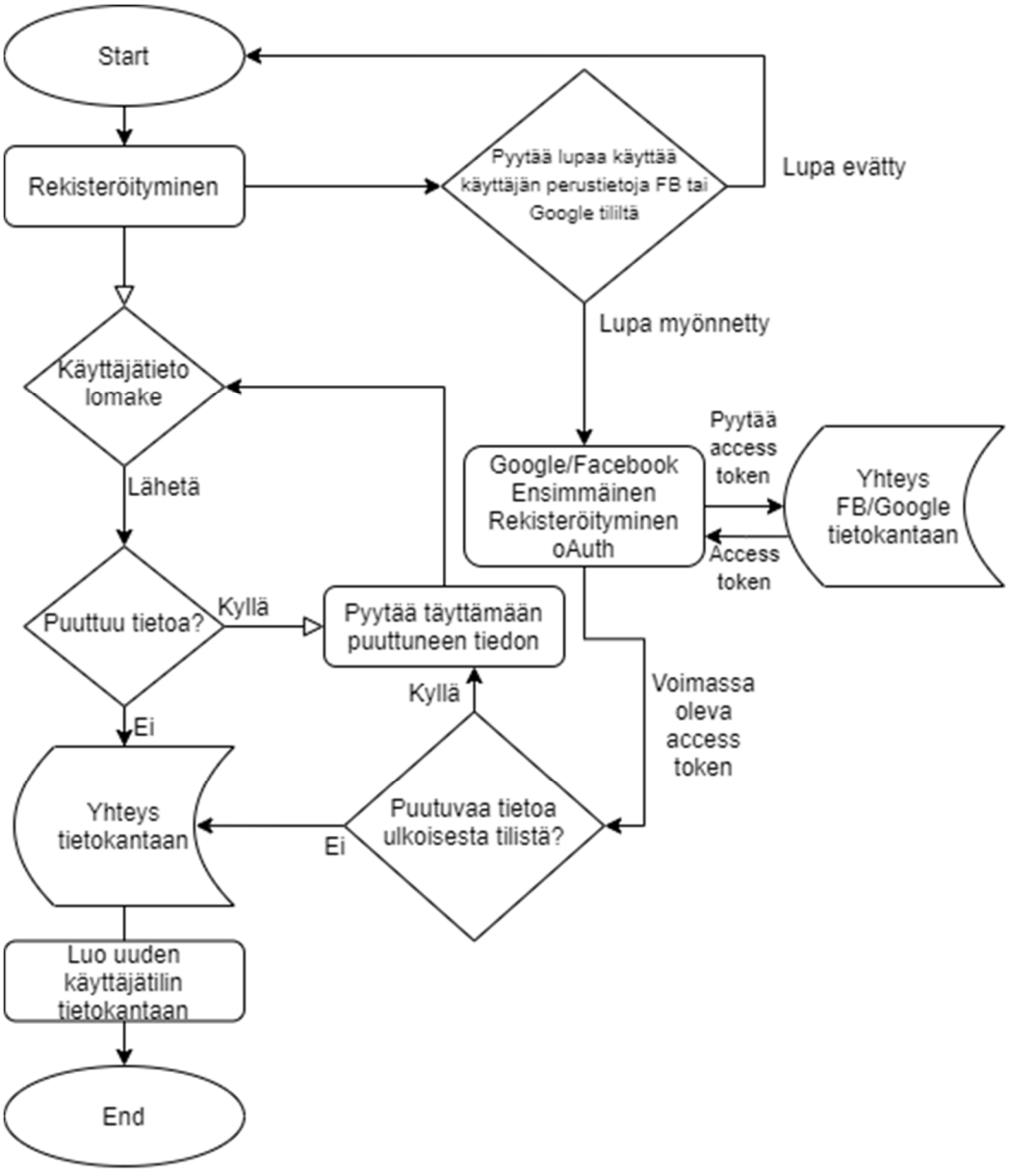
Rekisteröityminen ei ole pakollista sovelluksen tärkeiden toimintojen kannalta, kuten lipun ostamisen. Rekisteröityminen ja sisäänkirjautumisella sovellukseen voidaan tallentaa käyttäjän dataa, kuten tallennettuja reittejä.

4.4.1 Yleiskuvaus

Moduulin nimi:	Rekisteröityminen	**Moduulin tyyppi:**	Prosessi
Yleiskuvaus:	Asiakasprofiilin luonti tietokantaan		
Asiakkaat:	Yleiskäyttöinen.		
Riippuvuudet ja liitännät:	Rekisteröitymisen moduuli on riippuvainen tietokanta ja internet yhteydestä ja sekä mahdollisesti myös ulkoisen palveluntarjoajan kuten Google tai Facebook API tietokantojen yhteydestä ja toimivuudesta.		
Suunnittelija ja pvm:	Ville Vierros 21.4.2021		

4.4.2 Rajapinta yleisesti

Rekisteröityminen voidaan suorittaa vain sovellukseen sisään rakennetulta asiakastietolomakkeelta. Lomakkeessa pyydetään käyttäjää täyttämään seuraavat kentät: kokonimi, sähköposti, kaupunki, kieli, salasana, salasana uudelleen. Kaikki kentät ovat pakollisia ja niihin liittyy erilaisia vaatimuksia, kuten kokonimi kenttään kelpaa vain ASCII merkeistä kirjainmerkit a-ö. Sähköposti kenttä tarkistettaisiin flutterissa toimivassa regex:in koodin avulla. Kaupunki kentässä kelpaa vain ASCII merkeistä kirjainmerkit a-ö ja salasana kentässä vaaditaan erikoismerkki ja isokirjain.



UML kaavio rekisteröitymisestä, jossa näkyy oAuth rekisteröityminen ja mobiili sovelluksessa tapahtuvan käyttäjätietolomakkeen kautta tapahtuva rekisteröityminen. OAuth on pääsääntöisesti kirjautumiseen, mutta ensimmäistä kertaa oAuthia käytettäessä luodaan tietokantaan käyttäjätili, johon voidaan tallentaa tietoa, kuten reittejä, ostohistoriaa ja perusvalinta kielelle.

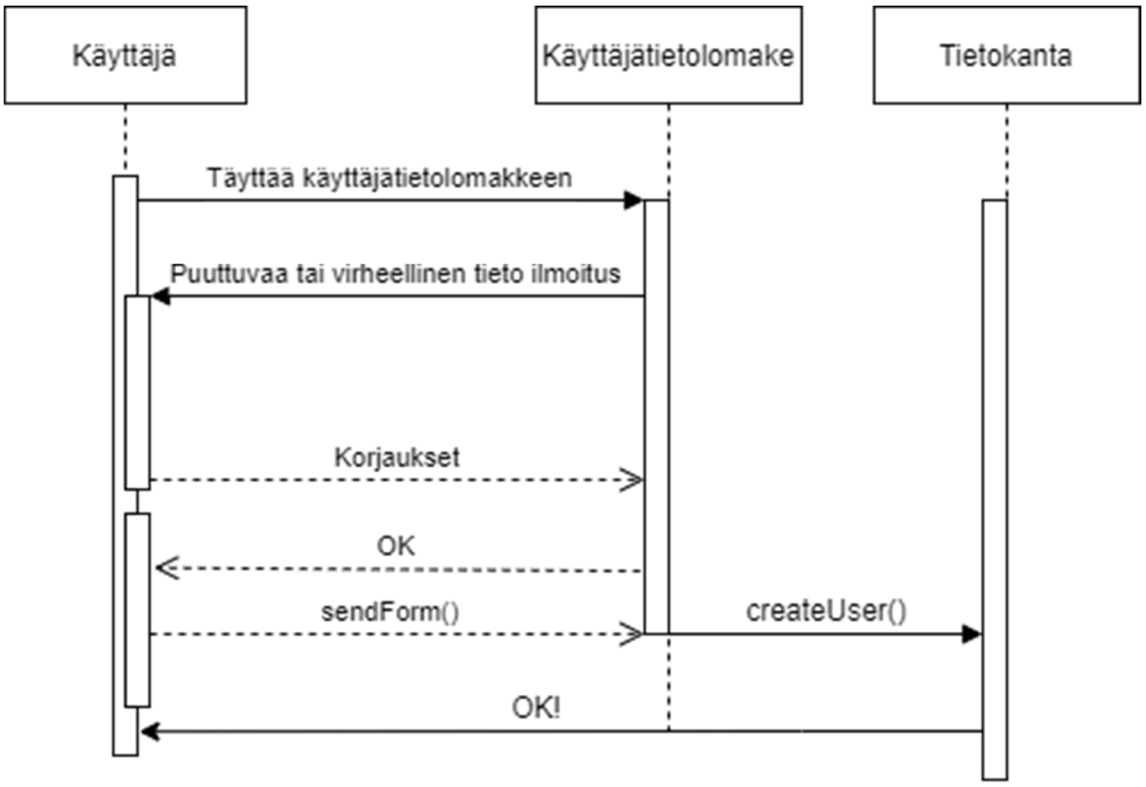
4.4.3 Rajapintafunktiot

Funktion nimi:	SendForm()	
Toiminto, mitä funktio tekee:	Lähettää käyttäjätietolomakkeen nappulaa painettaessa.	
Funktion parametrit ja paluuarvo:	**Parametrit:** Lomake	**Palautusarvot:** CreateUser()
Esiehdot:	Tekstikentät: etu- ja sukunimi, sähköposti, kaupunki, salasana, salasanan vahvistaminen ja kieli tulevat olla oikeassa muodossa.	
Jälkiehdot:	Lähettää käyttäjätietolomakkeen tietokantaan ja siirtää käyttäjän kirjautumis-ruudulle sovelluksessa. Lähettää käyttäjälomake tiedon TCP/IP kautta JSON muodossa.	
Virhetilanteet:	Virhe tapahtuu, jos lomakkeessa oleva tieto on virheellisessä muodossa. Virhe voi tapahtua, jos ei saada internet yhteyttä tietokantaan.	
Suunnittelija ja pvm:	Ville Vierros 21.04.2021	

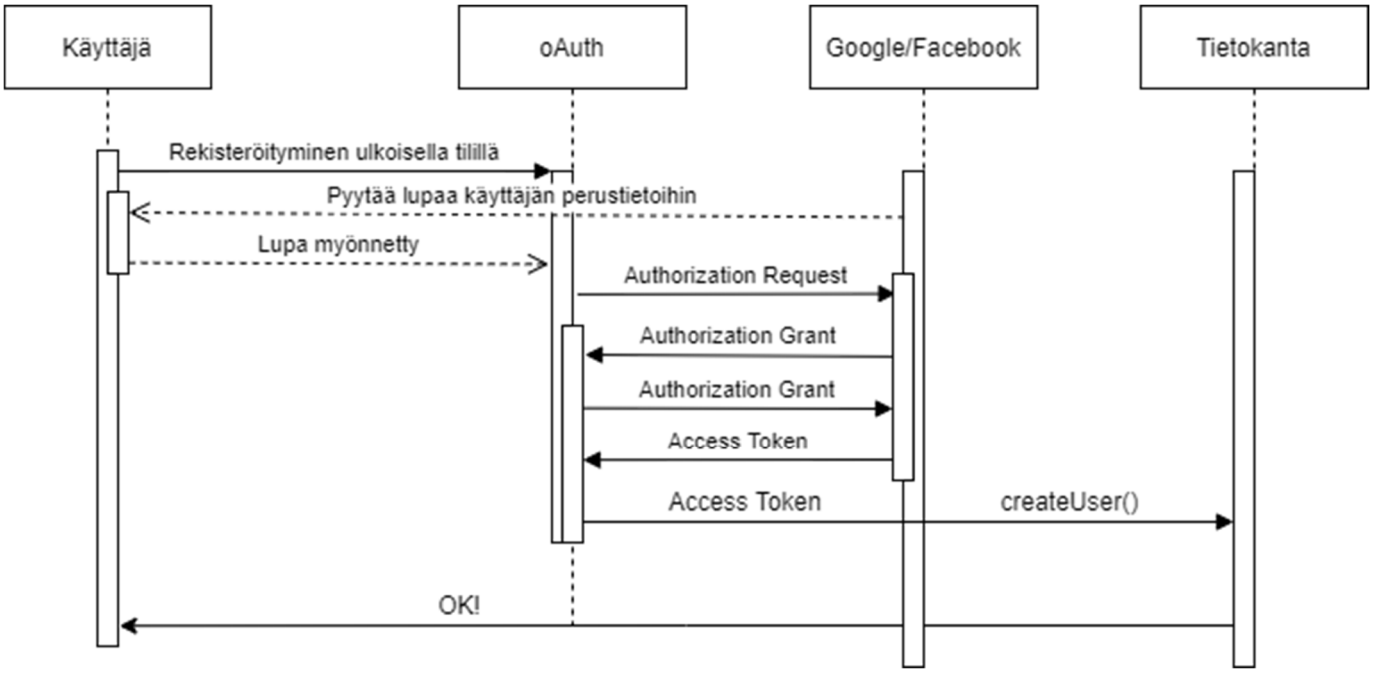
Funktion nimi:	CreateUser()	
Toiminto, mitä funktio tekee:	Luo käyttäjätilin tietokantaan.	
Funktion parametrit ja paluuarvo:	**Parametrit:** SendForm()	**Palautusarvot:** OK ShowLoginScreen()
Esiehdot:	Luo käyttäjän tietokantaan JSON tiedosta.	
Jälkiehdot:	Luonut tietokantaan käyttäjän ja lähettää funktion ShowLoginScreen() ja OK palautusarvoina	
Virhetilanteet:	Virhetilanne tapahtuu, jos tietokanta yhteys katkeaa tai virhe tapahtuu tietokannassa JSON tiedon käsittelystä.	
Suunnittelija ja pvm:	Ville Vierros 28.04.2021	

Funktion nimi:	ShowLoginScreen()	
Toiminto, mitä funktio tekee:	Siirtää käyttäjän kirjautumisruudulle	
Funktion parametrit ja paluuarvo:	**Parametrit:** Kutsuttu funktio: ShowLoginScreen() SendForm():missa	**Palautusarvot:** OK ShowLoginScreen()
Esiehdot:	OK viesti SendForm() ja ShowLoginScreen() funktio kutsuttu	
Jälkiehdot:	Suorittaa matkalippusovelluksessa käyttäjä näkymän muutoksen siirtämällä käyttäjän kirjautumisruudulle.	
Virhetilanteet:	Virhetilanne tapahtuu, jos tietokanta yhteys katkeaa	
Suunnittelija ja pvm:	Ville Vierros 04.05.2021	

4.4.4 Moduulin toteutus



Rekisteröityminen käyttäjätietolomakkeen kautta



Rekisteröityminen oAuthin avulla

4.4.5 Virhekäsittely

Jos käyttäjä syöttää käyttäjätietolomakkeessa tekstikenttiin väärässä muodossa olevaa tiedon, kuten salasanan varmistus tekstikenttään, jossa salasanan tulee vastata ensimmäiseksi kirjoitettua salasanaa, niin sovellus ilmoittaa virheilmoituksen, jossa kuvataan virhe tekstikentässä ja maalaa virheellisen tekstikentän punaiseksi.

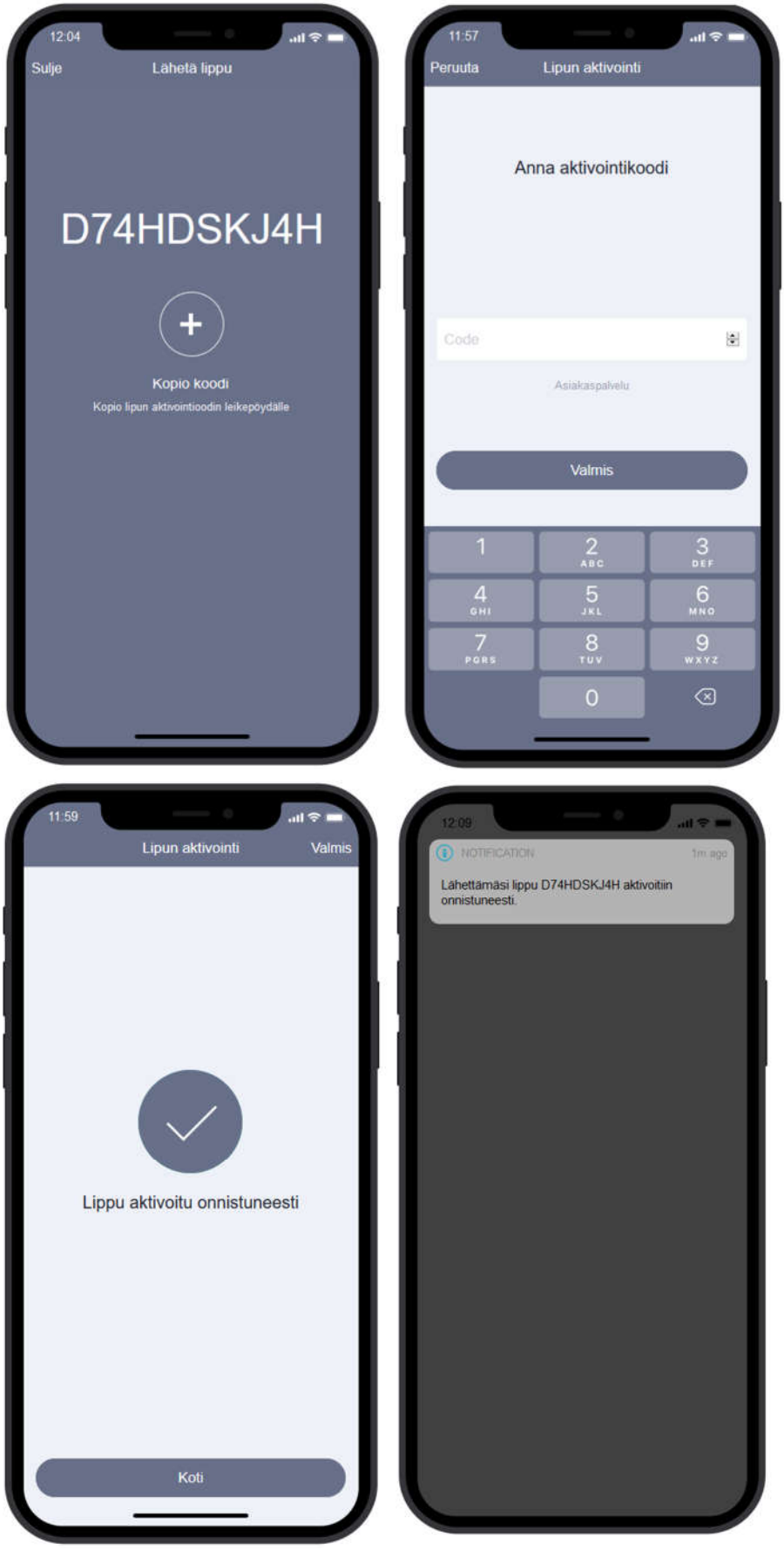
Virhe jos rekisteröitymisen yhteydessä katoaa nettiyhteys:

- 101 - Internet-yhteys: ”Virhe: Tarkista Internet-yhteys”

Virhe jos yrittää rekisteröityä oAuthin kautta ja Google/Facebook palvelimilla on käyttökatkos:

- 162 – Kirjautuminen: ”Virhe: Sovellus kohtasi virheen ulkoisessa kirjautumispalvelussa”

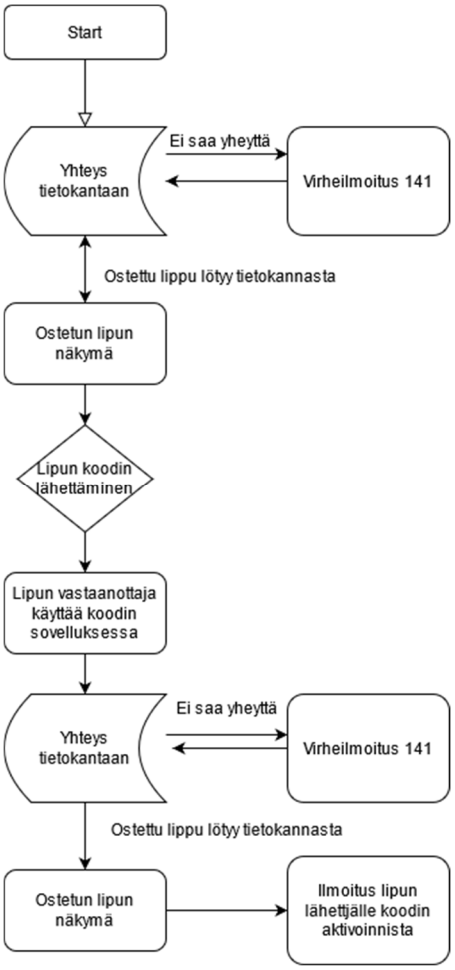
4.5 Lipun lähettäminen



4.5.1 Yleiskuvaus

Moduulin nimi:	Lipunlähettäminen	**Moduulin tyyppi:**	Prosessi
Yleiskuvaus:	Moduuli antaa koodin ostetusta lipusta, jonka käyttäjä voi lähettää toiselle käyttäjälle mitkä he voivat sitten käyttää.		
Asiakkaat:	Yleiskäyttöinen		
Riippuvuudet ja liitännät:	Moduuli on riippuvainen yhteydestä tietokantaan. Moduuli on riippuvainen lipunostaminen moduulista.		
Suunnittelija ja pvm:	Otto Karpoja 14.4.2021		

4.5.2 Rajapinta yleisesti



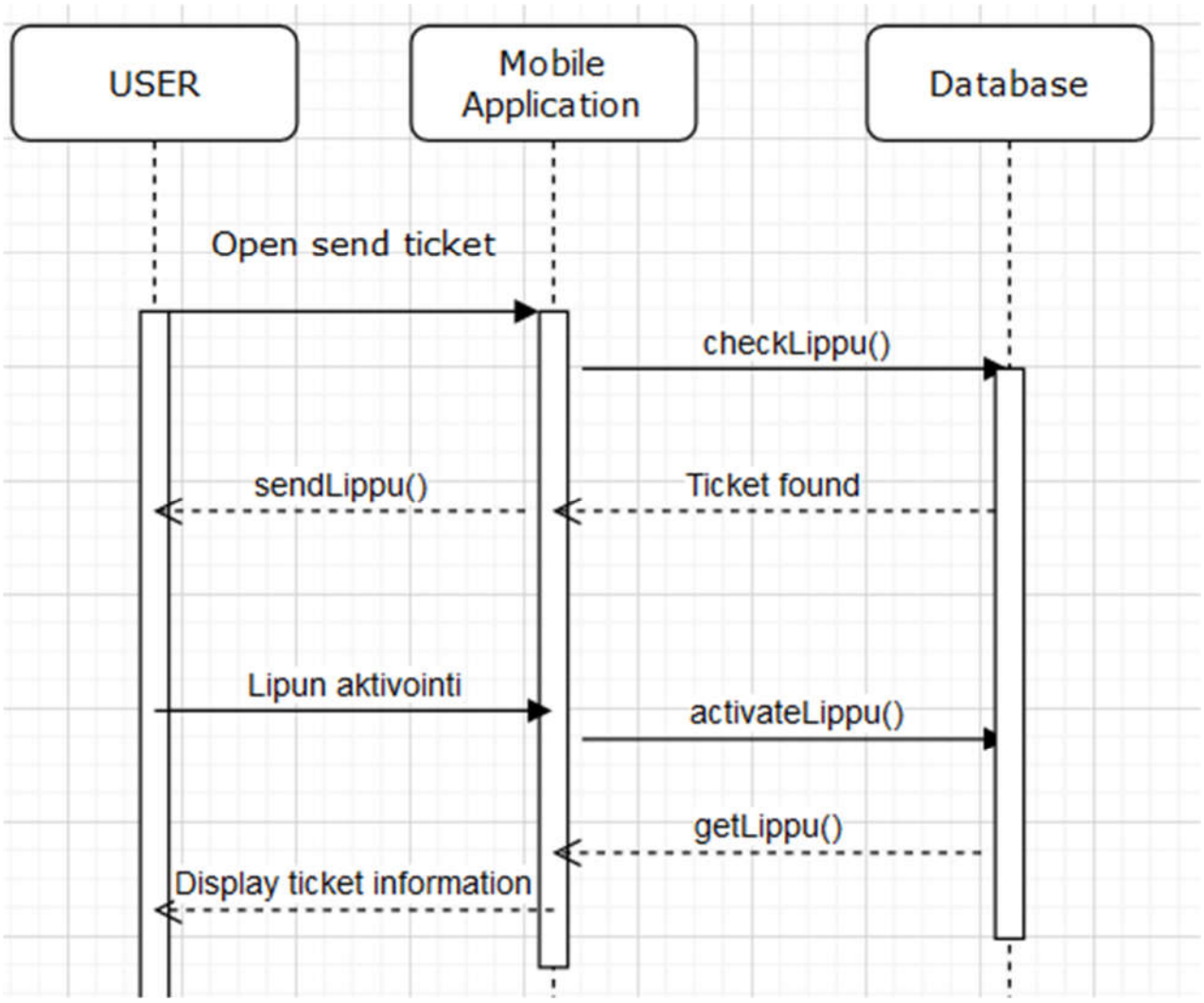
4.5.3 Rajapintafunktiot

Funktion nimi:	getLippu()				
Toiminto, mitä funktio tekee:	Funktio hakee tietokannasta lipuntiedot ja esittää ne käyttäjälle.				
Funktion parametrit ja paluuarvo:	**Parametrit:** Funktio käyttää getRouteDetails() hakemaan lipun reittitiedot.		**Paluuvarvot:** funktio palauttaa getRouteDetails():n avulla: Tarkka lähtopiste,Tarkka saapumispiste, Mahdolliset vaihdot, Liikennevälineet, Mahdollisten vaihtojen aikataulut, Tarkka lähtöaika, Tarkka saapumisaika,		
Esiehdot:	Ostettu lippu pitää löytyä tietokannasta ennen kuin funktiota voidaan käyttää.				
Jälkiehdot:	Funktio tuo asiakkaalle lipun tiedot ja ilmoittaa lipun lähettäjälle sen aktivoinnista.				
Virhetilanteet:	• 141 – Tietokantayhteys: ”VIRHE: Sovelluksen tietokantaan ei saada yhteyttä” • Internetyhteys ei toimi, esitetään virhe 101.				
Suunnittelija ja pvm:	Otto Karpoja 25.4.2021				

Funktion nimi:	sendLippu()				
Toiminto, mitä funktio tekee:	Funktio luo uniikin koodin minkä käyttäjä lähettää toiselle asiakkaalle lipun aktivoimiseksi.				
Funktion parametrit ja paluuarvo:	**Parametrit:** String		**Paluuvarvot:** Funktio palauttaa var Random() käyttämällä 10 merkkiä pitkän String variablen ja jättää sen myös tietokantaa checkLippu() varten.		
Esiehdot:	Ostettu lippu pitää löytyä tietokannasta ennen kuin funktiota voidaan käyttää.				
Jälkiehdot:	Funktio luo asiakkaalle helposti toimitettavan kirjain yhdistelmän, jonka hän voi sitten lähettää toiselle asiakkaalle.				
Virhetilanteet:	• 141 – Tietokantayhteys: ”VIRHE: Sovelluksen tietokantaan ei saada yhteyttä” • Internetyhteys ei toimi, esitetään virhe 101.				
Suunnittelija ja pvm:	Otto Karpoja 25.4.2021				

Funktion nimi:	activateLippu()				
Toiminto, mitä funktio tekee:	Funktio tarkistaa lipun olemassaolon käyttämällä käyttäjän antaamaan merkkijonoa.				
Funktion parametrit ja paluuarvo:	**Parametrit:** String		**Paluuvarvot:** Boolean		
Esiehdot:	Käyttäjällä on merkkijono minkä hän on käyttänyt aktivoidakseen lipun.				
Jälkiehdot:	Funktio vertaa käyttäjän merkkijonoa tietokannassa oleviin merkkijonoihin mitkä ovat luoto sinne sendLippu() funktiolla ja palauttaa boolean true tai false sovellukselle, minkä jälkeen sovellus tietää onko lippua olemassa.				
Virhetilanteet:	• 141 – Tietokantayhteys: ”VIRHE: Sovelluksen tietokantaan ei saada yhteyttä” • Internetyhteys ei toimi, esitetään virhe 101.				
Suunnittelija ja pvm:	Otto Karpoja 25.4.2021				

4.5.4 Moduulin toteutus



4.5.5 Virhekäsittely

Tarkempi kuvaus virhekäsittelystä suunnitteludokumentin kohdassa 3.4.

- 141 – Tietokantayhteys: ”VIRHE: Sovelluksen tietokantaan ei saada yhteyttä”
- Internetyhteys ei toimi, esitetään virhe 101.

5. VALMISOSAT JA ERITYISET TEKNISET RATKAISUT

5.1 Karttapalvelu, Open Street Map

Karttapalveluna käytetään avoimen lähdekoodin Open Street Mapia (OSM). OSM on saatavilla Android- ja iOS-sovelluksiin Mapbox Maps Software Development Kit -kirjastojen avulla.

- Mapbox Maps SDK for Android
- Mapbox Maps SDK for iOS

5.2 NAP

NAP-liikkumispalvelukatalogi on avoin kansallinen yhteyspiste (National Access Point, NAP) johon jokaisen liikkumispalveluja tuottavan on toimitettava tietoja digitaalisista olennaisten tietojen koneluettavista rajapinnoistaan. Tämän rajapinnan avulla sovelluksen kehittäjä saa haettua suunnittelemaamme sovellukseen seuraavat tiedot:

- palvelualue
- reitit
- aikataulut
- hinnat
- palvelun saatavuus
- esteettömyys

5.3 AWS ja MongoDB-tietokannat

Sovelluksen kehitysalustana käytetään Googlen Flutter SDK:ta, mikä tekee Amazonin Amplify Flutter ympäristöstä helposti yhdistettävän. Amazonin Amplify Flutter antaa meille mahdollisuuden rakentaa fullstack ohjelmisto Android ja iOS käyttöjärjestelmille. MongoDB tietokanta yhdistetään Flutter alustaan mongo_dart kirjastoa käyttäen. Amazonin Amplify Flutter CLI yhdistetään Amazon Web Servicen pilvipalveluun.

5.4 Nets

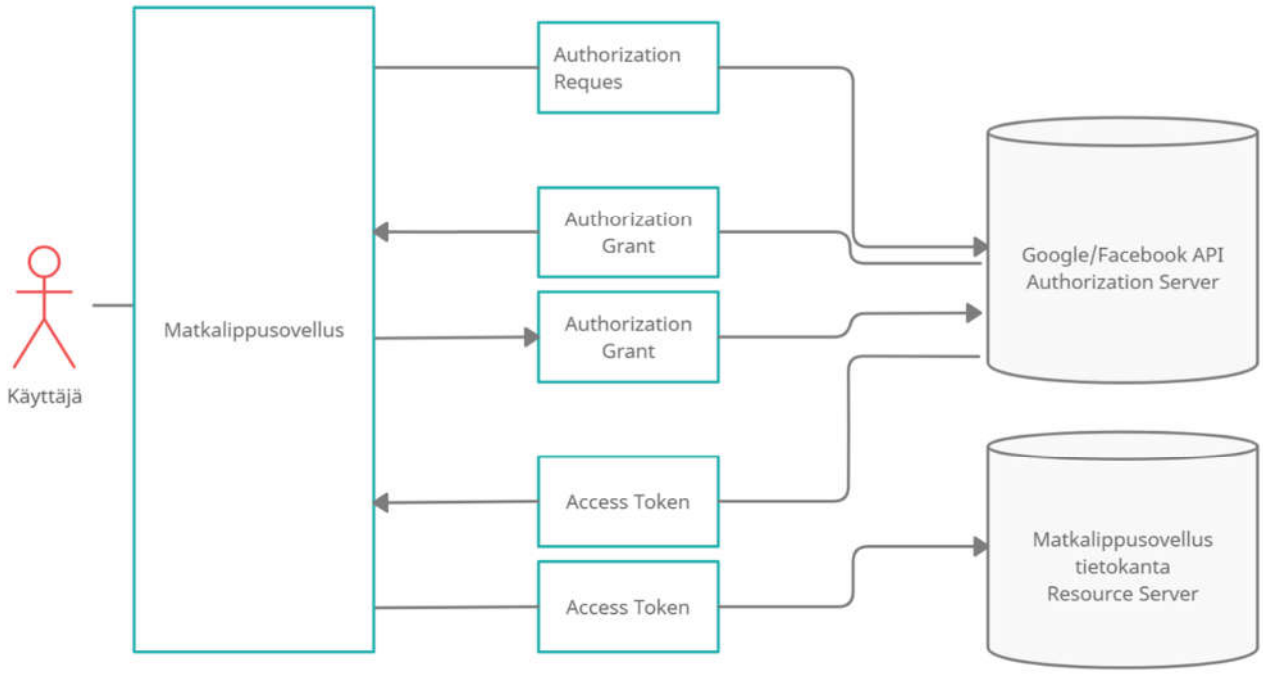
Nets on kolmannen osapuolen tarjoama maksupalvelu, joka sallii korttimaksun verkkokaupassa. Palvelun tarkka toiminta on salattua, joten emme voi kuin spekuloida sen toimintaa ilman yhteydenottoa yrityksen henkilökuntaan - todennäköisesti yhteydenottokaan ei auttaisi tässä tilanteessa.

Hypoteettisesti maksujärjestelmä toimii niin, että käyttäjä ohjataan Netsin maksupalveluun tai korttimaksu tapahtuu täysin back-endissä. Mikäli maksu tapahtuu täysin back-endissä niin sovellus ottaa yhteyttä Netsin palvelimelle ja Netsin palvelin ensin tarkastaa korttitiedot Visalta / Mastercardilta. Sen jälkeen Netsin palvelin ilmoittaa korttitietojen olevan kunnossa. Tämän jälkeen siirrymme itse maksuun ja sovellus ilmoittaa Netsin palvelimelle veloitettavan hinnan ja korttitiedot, jonka jälkeen Nets välittää API-kutsun eteenpäin Visalle / Mastercardille hoitaen veloituksen kortilta. Visa / Mastercard tarkistaa sen jälkeen pankin palvelimilta, että tilillä on katetta, pankin palvelin sanoo, että on katetta / ei ole katetta - mikäli tilillä on katetta niin samalla

kutsulla tehdään myös veloitus. Sitten Visa / Mastercard vastaa Nets:n API-kutsuun, että kaikki on kunnossa ja, että veloitus on tapahtunut. Lopuksi Nets vastaa matkalippupalvelimen API-kutsuun, että veloitus on tehty ja matkalippu kirjataan tietokantaan. Sovelluspalvelin hakee matkalipun tiedot matkalipputietokannasta ja hyväksytyn maksun varmennus johtaa lipun ruudulle tulostamiseen.

5.5 Tunnistautuminen

OAuth 2.0 rajapinta mahdollistaa sisäänkirjautumisen ja rekisteröitymisen sovellukseen ulkoiselta palveluntarjoilijalta, kuten Facebook ja Google. Käyttäjän kirjautuessa sisään oAuth pyytää Facebook tai Google API:lta perustiedot kuten käyttäjänimi ja profiilikuva. OAuth ei kumminkaan, koskaan jaa Facbook tai Google salasanaa sovellukselle eteenpäin vaan toimii vain tunnistautumisessa sovellukseen.



Kuva 1. Kirjautuminen Matkalippusovellukseen oAuthilla

Kun matkalippu sovellus pyytää lupaa eli Authorization Request:tia Googlen/Facebookin API tulee käyttäjän hyväksyä pyyntö, jotta Matkalippu sovellus voi käyttää Google/Facobook tililtä käyttäjän myöntämiä oikeuksia eli matkalippusovelluksen tapauksessa käyttäjänimi, sähköposti ja mahdollisesti kaupunki. Kun matkalippusovellus on saanut oikeuden eli Access Tokenin Googlen/Facebook API:ltä niin voi käyttäjä käyttää matkalippusovellustietokantaa ja sitä kautta päästä käsiksi sovelluksen kirjatumista vaativiin palveluihin.

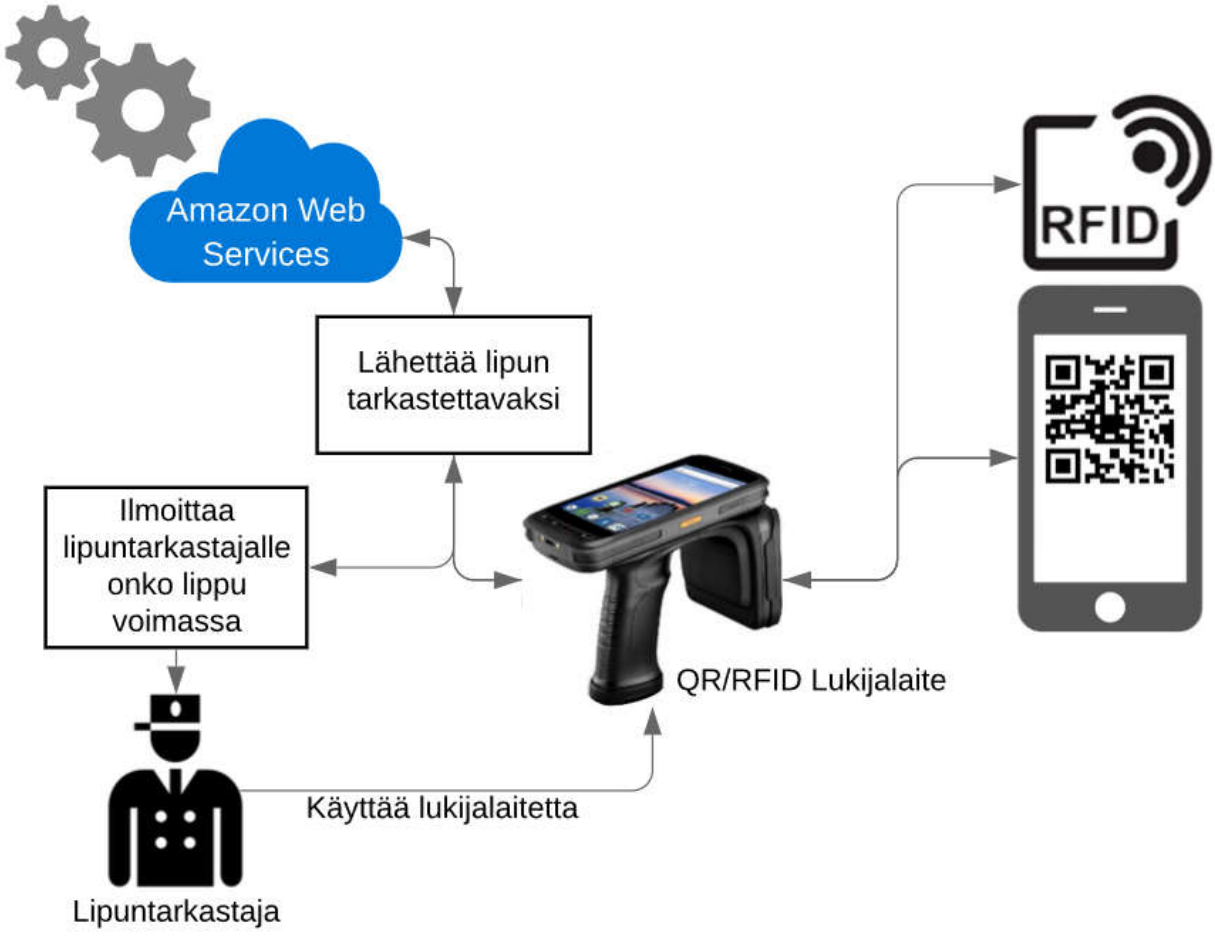
6. HYLÄTYT RATKAISUVAIHTOEHDOT

Komponentin nimi:	Tietokanta numero 2.	**Hylkääjä sekä pvm:**	KTu/29.4.2021
Hylkäyksen peruste:	Päätettiin, että käytetään sovelluksessa ainoastaan yhtä tietokantaa selkeyden vuoksi. MongöDB mahdollistaa tämän ratkaisun		

7. JATKOKEHITYSAJATUKSIA

Jatkoajatuksen ID:	TV12	**Ehdottaja / pvm:**	1.5.2021
Jatkokehitysajatus:	Lipunmyyntipisteiden sijainnin hakeminen		
Jatkoajatuksen ID:	ETV1.3	**Ehdottaja / pvm:**	15.4.2021
Jatkokehitysajatus:	Huomioidaan värit (mm.yötila ja värisokeus) seuraavassa kehitysvaiheessa		
Jatkoajatuksen ID:	TV14	**Ehdottaja / pvm:**	8.4.2021
Jatkokehitysajatus:	Lipuntarkastajan pitää voida tarkistaa sovelluksesta lippu		

Lipuntarkastajan näkymä:



8. VIELÄ AVOIMET ASIAT

Avoimiksi asioiksi suunnittelu dokumentista jää moduuleista funktioita, joita emme kuvanneet:

- checkLippu() -funktiö jäi kuvaamatta moduulissa 4.5

Lisäksi moduulikuvaus tarvitaan vielä seuraavista osa-alueista:

- Aikataulujen tarkastelu
- Karttapalvelu
- Ylläpitäjän näkymä