Vertti Ahlstén

Ajanvarauksen automatisointi Frends-integraatioilla

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintätekniikka

Insinöörityö

24.11.2022

Tiivistelmä

Tekijä:	Vertti Ahlstén
Otsikko:	Ajanvarauksen automatisointi Frends-integraatioilla
Sivumäärä:	36 sivua + 1 liite
Aika:	24.11.2022
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Tieto- ja viestintätekniikka
Ammatillinen pääaine:	Ohjelmistotuotanto
Ohjaajat:	Lehtori Simo Silander Diplomi-insinööri Ossi Galkin

---

Insinöörityön tavoitteena oli luoda järjestelmäintegraatio Hubspot-palvelun ja Microsoft Teams -viestintäpalvelun välille. Integraation tehtävänä oli automatisoida ajanvarauksen käsittely ja vähentää käyttäjien työtaakkaa. Frends-alustalle toteutettujen integraatioiden tuli käsitellä ajanvarauslomakkeella annetut tiedot ja lähettää ne interaktiivisen kortin muodossa Teams-kanavalle. Sen tuli myös valvoa ajanvarausten tilaa ja lähettää niistä käyttäjille muistutukset, mikäli varauksia ei käsitelty tietyn ajan puitteissa.

Työ aloitettiin suunnittelemalla toteutus ja sen jälkeen valitsemalla sekä tutustumalla työvälineisiin. Työhön valikoitui Microsoft Power Automate -alusta korttien lähettämiseksi Teams-kanavalle ja Graph API -rajapinta korttien seurantaan kanavalla. Tämä loi hyvän pohjan toteutuksen tikettijärjestelmän rakentamiseksi, mikä käsitteli ja seurasi ajanvarausten tilaa. Kun Frends-integraatio saatiin rakennettua, siihen yhdistettiin ajanvarauslomake käyttämällä Hubspot-palvelun prosesseja. Lomakkeen lähetys määritettiin käynnistämään tapahtumakulku, joka lähetti lomakkeen tiedot POST-tyyppisenä HTTP-kutsuna Frends-liittymälle. Lopuksi työhön rakennettiin virheenhallinta, jonka tarkoitus oli vähentää ohjelmiston ylläpidosta johtuvaa työtä varautumalla ennakkoon mahdollisiin virhetilanteisiin.

Lopputuotteena insinöörityöstä syntyi kahden järjestelmän välinen integraatio, joka muodostui viidestä prosessista. Integraatio hyväksyttiin työn ensimmäisenä versiona ja otettiin käyttöön työvälineenä helpottamaan ajanvarauksen käsittelyprosessia. Työn kehitystä jatkettiin paranneltuun toiseen versioon, joka sisältää lisätoiminnallisuuksia ja nykyisten menetelmien optimointia.

Avainsanat: Integraatio, rajapinnat, automaatio, Frends

Abstract

Author:	Vertti Ahlstén
Title:	Automatisation of Appointment Process with Frends iPaaS
Number of Pages:	36 pages + 1 appendice
Date:	24 November 2022
Degree:	Bachelor of Engineering
Degree Programme:	Information and Communication Technology
Professional Major:	Software Development
Supervisors:	Simo Silander, Senior Lecturer
	Ossi Galkin, Master of Science (Engineering)

The purpose of the study was to create an integration with Frends iPaaS between the Hubspot CRM platform and Microsoft Teams communication platform. The goal of integration was to automate the appointment process to ease the workload of the users. The Frends process was meant to create and manage appointment tickets, then send them to the Teams-channel as adaptive cards. It was also meant to monitor the tickets and send reminders of appointments not handled in the intended schedule.

The study began with planning the structure of the software and choosing suitable tools. Microsoft Power Automate was chosen for sending the adaptive cards to the Teams channel and the Graph API interface was picked for monitoring cards in the channel. This created a good base for building the ticket system which managed and took care of monitoring the status of the tickets. Once the Frends integration had been built, the Hubspot meeting tool was connected to it using a workflow. The workflow was activated when the form was submitted to the system, then sending the information given by the client to the Frends process with a POST rest request. As the final touch, error management was built into the software to reduce the work required by the system maintenance.

The outcome of the study is the integration between two software built by five processes in total. The results were accepted as the first version of the software, and it was introduced to the users as a tool to ease their workload. The development was carried on to a second version that offers additional features and optimisation of the current version.

Keywords: Integration, interface, automation, Frends

Sisällys

Lyhenteet

ERP: *Enterprise Resource Planning*. Yrityksen toiminnanohjausjärjestelmien integraatio.

CRM: *Customer Relationship Management*. Asiakashallintajärjestelmien integraatio.

API: *Application Programming Interface*. Ohjelmistorajapinta, joka mahdollistaa kahden ohjelmiston välisen tiedon välityksen.

BPMN: *Business Process Model and Notation*. Liiketoimintaprosessien graafinen mallintaminen kaavioilla.

HA: *A High Availability Environment*. Korkean saatavuuden ympäristö, eli vakaa järjestelmäympäristö, jossa yksittäiset virheet eivät estä ohjelmistojen toimintaa.

VPN: *Virtual Private Network*. Virtuaalinen erillisverkko, joka salaa käyttäjän tietoliikenteen, IP-osoitteen sekä sijainnin.

1 Johdanto

Tämän insinöörityön tavoitteena on luoda ohjelmistointegraatio, joka tulee helpottamaan loppukäyttäjän työntekoa vähentämällä käyttäjän käsin tekemää työtä. Ohjelmistointegraatiot ovat kahden tai useamman sovelluksen välissä toimivia toteutuksia, jotka mahdollistavat tiedon kulun sovellusten välillä. Työn tavoite on automatisoida ajanvarauksen tiedonkulku käyttäjälle ja mahdollistaa ajanvarauksen vaativien toimintojen toteutus vain käyttäjän kuittauksella. Työn on tilannut ohjelmistotalo HiQ Finland Oy, joka tarjoaa ja tuottaa konsulttipalveluita muun muassa integraatioiden parissa (1). Insinöörityössä tullaan käsittelemään ja läpikäymään toteutuksen suunnittelu, toteutus ja tulokset. Työn lopputuotteena on valmis integraatiokokonaisuus kolmen rajapinnan välillä. Integraatio koostuu pienemmistä osista, jotka yhdessä muodostavat toimivan ja luotettavan kokonaisuuden.

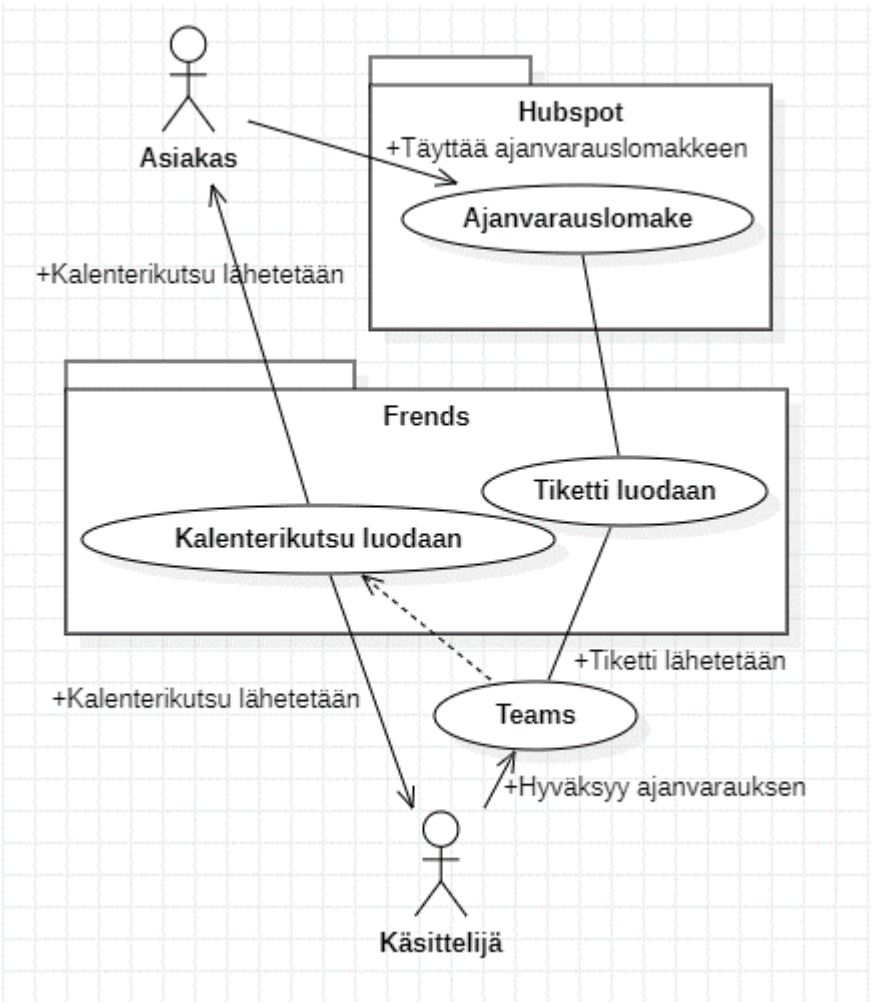
Ajanvarauksen käsittelyprosessi valikoitui aiheeksi, sillä sitä ei vielä ole automatisoitu, ja tarve toteutuksella on kasvanut. Asiakasmäärien kasvaessa yhteydenotot ja ajanvaraukset tuote-esittelyille ovat lisääntyneet merkittävästi. On tärkeää, että käyttäjien työvälineet pystyvät tarjoamaan helpotusta kasvavaan työkuormaan. Tapahtumakulku nykyisellään koostuu käyttäjien välittämästä tiedosta, kuten vastaanotetun ajanvarauksen poimiminen sähköpostista, varauksen lisäämisestä kalenteriin ja keskinäisestä kommunikaatiosta muiden käyttäjien kanssa. Käsin tapahtuva tiedonsiirto eri alustoilta toisille jättää mahdollisuuksia virheille. Tällä on vaikutusta työn laatuun ja nopeuteen. Nykyisen prosessin epävarmuus ja kannattamattomuus kasvavien asiakasmäärien edessä toi tarpeen ajanvarauksen automatisoinnista esille.

HiQ Finland Oy, joka on HiQ Internationalin tytäryhtiö, on alkujaan ruotsalainen ohjelmistotalo, joka myy järjestelmäintegraatioita ja niiden toteutuksia (2). HiQ on palvellut vuodesta 1987, ja sen nimi juontuu sanoista “High Quality”. Yritys palvelee kansainvälisten markkinoiden ohjelmistotarpeita yli 2000 asiantuntijan ja neljän eri maan toimistojen voimin. (1.) HiQ tarjoaa integraatiokonsultaation lisäksi projektitoimituksia, palveluiden ylläpitoa ja jatkokehitystä sekä digitaalista

suunnittelua ja palvelumuotoilua (3). Nimensä mukaisesti yritys pyrkii tarjoamaan laadukkaita ja käyttäjäystävällisiä ratkaisuja. Yrityksellä on omistuksessaan integraatioalusta Frends ja Great Apes -brändi (2).

2 Vaatimukset

Tilaaja on ohjeistanut, että työn tulee välittää tietoa asiakasrajapinnan Hubspot-palvelun ja käyttäjän Microsoft Teamsin välillä (kuva 1). Hubspot-rajapintaa käytetään asiakkaan tuote-esittelyn ajanvarauspyynnön vastaanottamiseen ja käsiteltäväksi lähettämiseen esittelyn järjestävälle taholle eli myyjille. Microsoft Teams on käyttäjien valitsema alusta, joka varmistaa, että varaukset saadaan käsiteltyä nopeammalla aikataululla. Ajanvarauksen tiedon välittämiseen, käsittelyyn ja monitorointiin tullaan käyttämään Frends-integraatioalustaa. Prosessin tulee seurata käyttäjälle lähetettyjä ajanvarauksia ja muistuttaa niistä, mikäli niihin ei reagoida tietyn ajan sisällä. Lisäksi käyttäjän tulee saada tieto hyväksytyistä ajanvarauksista ja ohjelmiston on suoritettava kalenterikutsun luonti ja lähetys varauksen tehneelle asiakkaalle sekä sen käsittelijälle.



Kuva 1. Ajanvarauksen tapahtumakulusta havainnollistava käyttötapauskaavio.

Toteutuksen tulee olla laadukkaasti tehty ja sen tulee aiheuttaa vähemmän työtä käyttäjälleen kuin tapahtumakulku nykyisellään. Siksi virheiden hallinta ja niistä palautuminen on tärkeä osa toteutusta. Prosessin tulee pyrkiä vähentämään käyttäjänsä työtaakkaa ja siksi sen tulee palautua virheistä mahdollisimman hyvin ilman käyttäjän valvontaa.

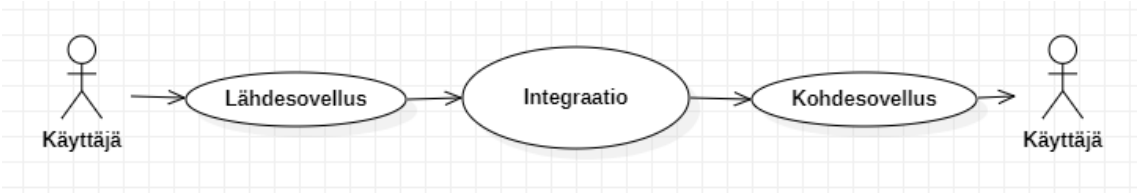
3 Frends integraatioiden työkaluna

Työ toteutetaan Frends-integraatioalustalla, sillä se on helppokäyttöinen ja joustava työkalu integraatioiden toteuttamiseksi. Ajanvarauksen automatisointi vaatii

paljon loogisia kokonaisuuksia toimiakseen, ja Frends mahdollistaa nämä pienellä vaivalla. Tässä luvussa käsittelemme tarkemmin, mitä integraatiot pitävät sisällään ja miten integraatioalusta Frends toimii näiden toteuttamiseen työkaluna.

3.1 Ohjelmistointegraatiot

Ohjelmistointegraatiot ovat sovellusten välillä toimivia ohjelmia, jotka mahdollistavat tiedon liikkumisen sovellukselta toiselle (kuva 2). Tiedon lähteitä ja kohteita integraatioilla voi olla useampia. (4.) Tiedonvälitys ei ole integraatioiden ainoa tavoite, vaan ne voivat mahdollistaa jopa toiminnallisuuksien jakamisen sovellusten välillä. Integraatioita voidaan käyttää esimerkiksi tilanteessa, jossa yritys haluaa päivittää asiakas- ja laskutustietokantansa, jotta asiakkaiden laskutustiedot päivittyisivät automaattisesti henkilötietojen päivittyessä. (5.) Tämänlainen integraatio saattaa olla yksinkertainen tietoja käsittelevä ja siirtävä toteutus, tai se voi sisältää loogisia toiminnallisuuksia. Integraation ansiosta ohjelmat saavat ajankohtaista tietoa tietokannoista, joihin niillä ei muutoin olisi pääsyä. Tämä myös vähentää virheiden määrää, joita vanhentunut data voi aiheuttaa. (6.)



Kuva 2. Havainnollistava kuva yhdensuuntaisesta integraatiototeutuksesta, joka välittää tiedon lähteestä kohdesovellukseen.

Ohjelmistointegraatiot vaativat usein toisistaan erilaisia toteutuksia, ja harvoin integraatio on täysin samanlainen toisensa kanssa. Tämän vuoksi integraatioiden toteutuksille usein rakennetaan uniikit ohjelmistot ja työvälineet. (7.) Vaikka integraatiot ovat työläitä, ne vähentävät työkuormaa ja mahdollistavat tehokkaamman työskentelyn jo käyttöönotosta alkaen. Ne tarjoavat myös arvokasta

tietoa yritykselle, paljastamalla tiedoissa esiintyviä kehityssuuntia, jotka muuten voisivat jäädä huomaamatta. (5.)

Integraatioita voidaan jakaa kategorioihin, joita on muun muassa:

- ERP (“Enterprise Resource Planning”), eli yrityksen toiminnanohjauksen integraatio. ERP-integraation muoto on tarkoitettu nimenomaan keskustelemaan yrityksen ydinsovellusten kuten talous-, henkilöstö- ja tuotehallinnan järjestelmien kanssa. Tämä auttaa yrityksiä jakamaan tietoa sovelluksien välillä ja sujuvoittaa näin työntekoa. (5; 8.)
- CRM (“Customer Relationship Management”), eli asiakkuuksien hallintajärjestelmän integraatio. CRM-integraatiot helpottavat asiakkaita koskevan tiedon välittämisessä sovelluksille, kuten laskutus- tai markkinointijärjestelmille. Tämä nopeuttaa asiakastietojen käsittelyä ja lisää asiakastyytyväisyyttä. (5; 9.)
- Markkinoinnin automaation integraatio (“Marketing automation integrations”), eli integraatio, jonka tarkoituksena on automatisoida markkinoinnissa käytettäviä prosesseja. Tällaisissa integraatioissa voidaan yhdistää asiakaspalvelun ja myynnin järjestelmiä. Tämä mahdollistaa sujuvamman myynnin, sekä nostaa asiakastyytyväisyyttä. (5.)

Integraatioissa usein hyödynnetään ohjelmointirajapintoja sovellusten kytkemiseksi toisiinsa. API (“Application Programming Interface”) eli ohjelmointirajapinta mahdollistaa integraatioiden yhdistämisen sovelluksiin. (10.) Rajapinnat vastaanottavat kutsuja toisista ohjelmistoista, jotka joko lähettävät tai hakevat dataa (6). Ne käsittelevät ja todentavat sisääntulevan rajapintakutsun, ja suorittavat määritetyn toiminnallisuuden. Rajapintojen hyöty syntyy niiden nopeasta tavasta liikutella tietoa sovellusten välillä. (7.) Ne ovat myös tarpeeseen helposti mukautuvia, sillä niitä saadaan rakennettua käyttäjän tarpeiden mukaisiksi. Useista uudemmista sovelluksista löytyy avoin rajapinta, joka on asiakkaiden käytettävissä ja hyödynnettävissä (10). Tämä mahdollistaa integraatioiden to-

teutuksen asiakkaan tarvitsemien sovellusten välille. Rajapintojen käyttöä saadaan muun muassa rajattua API-avaimilla, joita käytetään asiakkaan todentamiseen kutsujen yhteydessä. Avain on asiakaskohtaisesti luotu avainkoodi, joka sallii rajapintakutsujen vastaanottamisen. (6.) Rajapintoihin saadaan myös rakennettua tiukempia todentamismenetelmiä, jotka vaativat käyttäjätietojen syötön tai tunnistusavaimen käytön. Kutsujen todentaminen on tarpeellista, kun ohjelmistoista haettava data sisältää arkaluontoista tietoa käyttäjistä tai tiedot omistavista yrityksistä (6). Suojaamaton rajapinta on altis väärinkäytöksille, tiedonkalastelulle sekä palvelunestohyökkäyksille.

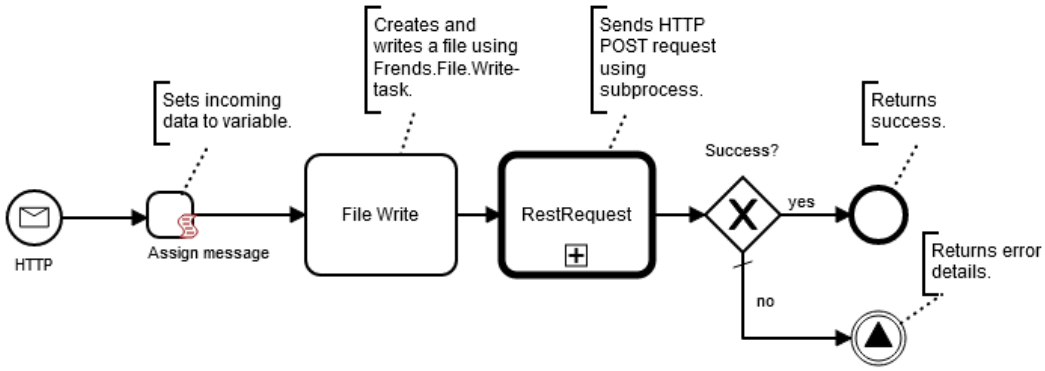
Vaikka rajapinnoista ja integraatioista löytyy paljon samaa, ovat ne toisistaan erillisiä kokonaisuuksia. Rajapintoja käytetään usein integraatioiden ja sovelluksien välisiin yhteyksiin, mutta se ei ole ainut keino välittää tietoa ohjelmistojen välillä. (11.) Integraatio voisi toimia esimerkiksi lokaalin tietokannan kanssa tai seuraamassa tiedostojen päivittymistä hakemistossa, sekä huolehtimassa tietojen päivittämisestä tarpeellisiin kantoihin. (10.) Rajapinnat taas toimivat työkaluna, jolla sovellukset saavat siirrettyä tietoa. Tähän väliin ei välttämättä tarvita integraatiota, mikäli ohjelmiston välittämä tieto on jo sovelluksen ymmärtämässä muodossa tai muutos saadaan tehtyä kohteessa tai lähteessä. (11.) Integraatiot astuvat kuvaan, kun kaksi tai useampaa sovellusta halutaan saada välittämään tietoa toisilleen, ilman sovellusten tai lähdedatan muokkaamista. (7.) Toteutuksen selkeä rakenne ja uudelleenkäytettävyys takaa pitkälle kantavan integraation, joka tukee yritystä tietojenkäsittelyssä.

3.2 Frends

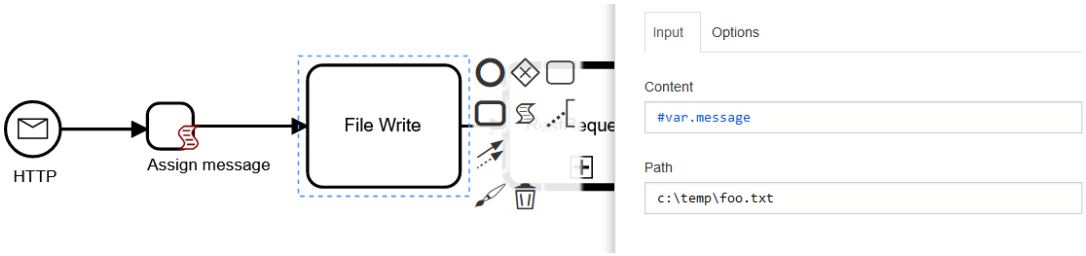
Frends on integraatioalusta, jonka avulla voi toteuttaa ohjelmistointegraation kirjoittamatta riviäkään koodia. Sen käyttöliittymä on rakennettu käyttämään BPMN- (“Business Process Model and Notation”) eli liiketoiminta- ja luokitusmalleja (12; 13) ja ne muistuttavat ohjelmoijille tutuista käyttötapaus-, luokka- ja sekvenssikaaviosta. Tämä mahdollistaa helposti ymmärrettävän lopputuloksen

myös monimutkaisemmista toteutuksista. Frends alustana tukee DevOps-toimintamallin ketteriä menetelmiä. (12.) DevOps eli kehityksen ja operoinnin toimintamalli pyrkii automatisoimaan sähköisten palveluiden kehityksen, testaamisen, sekä ylläpidon (14). Ensimmäisen katsauksen näihin menetelmiin antaa jo alustan graafinen käyttöliittymä. Jo monelle tutut prosessikaaviot luovat intuitiivisen ja sujuvan käyttäjäkokemuksen.

Seuraavaksi käymme läpi Frends-ympäristön toimintaperiaatteita. Ympäristöissä luodaan liittymiä, jotka koostuvat muuttujista, ehdoista, ohjelmatoteutuksista sekä aliliittymistä (15)(kuva 3). Aliliittymiä, eli aliohjelmien tavoin toimivia liittymiä käytetään pienempien kokonaisuuksien rakentamiseksi, jotka auttavat käyttäjiä pilkkomaan suurempia liittymiä pienemmiksi ja uudestaan käytettäviksi kokonaisuuksiksi (16). Nämä liittymät ovat myös muiden prosessien hyödynnettävissä (17). Liittymät aktivoidaan ehdoilla tai tapahtumilla, joita voivat olla esimerkiksi HTTP-kutsun vastaanotto, ajastus tai tapahtuma, kuten hakemistoon lisätty tiedosto. Nämä ehdot ja triggerit määrittävät liittymän käynnistystavan. Esimerkiksi ehto voi olla rajapinnasta palautunut sanoma, mikäli tämä sisältää tietoa, liittymä käynnistetään. (18.) Frends tarjoaa usein tarvitut työkalut integraatioiden rakentamiseksi valmiissa paketeissa. Näitä kutsutaan nimellä “Frends Task”, eli käyttäjille suunnattuja toimintoja, joiden tehtävänä on toimia toteutuspohjina erinäisille toiminnallisuuksille. Nämä ovat C#-ohjelmointikielen toteutuksia, jotka liitetään ohjelmaan aliohjelman tavoin. (19.) Ne saavat syötteenä käyttäjän määrittämät parametrit, jotka ohjailevat sovelluksen toimintaa (kuva 4), eli ne vaativat käyttäjältä vain konfiguraation. Esimerkiksi usein käytetyt taulukkomuotoiset tiedostot eli CSV-tiedostot saadaan käsiteltyä sille tarkoitetulla toiminnolla mutkattomasti. Sen avulla data saadaan käännettyä lista- tai JSON-muotoon ja sillä voidaan luoda CSV-muotoinen tiedosto. Frends-toiminnot ovat avoimen lähdekoodin ohjelmatoteutuksia, eli niiden lähdekoodi on kenen tahansa nähtävissä Github-versionhallintapalvelussa (20) ja muokattavissa omaan toteutukseen sopivaksi. Frends tarjoaa käyttäjälleen mahdollisuuden rakentaa omia tarpeisiinsa sopivia toimintoja ja lisätä niitä luotuihin prosesseihin. (19; 20; 21.)



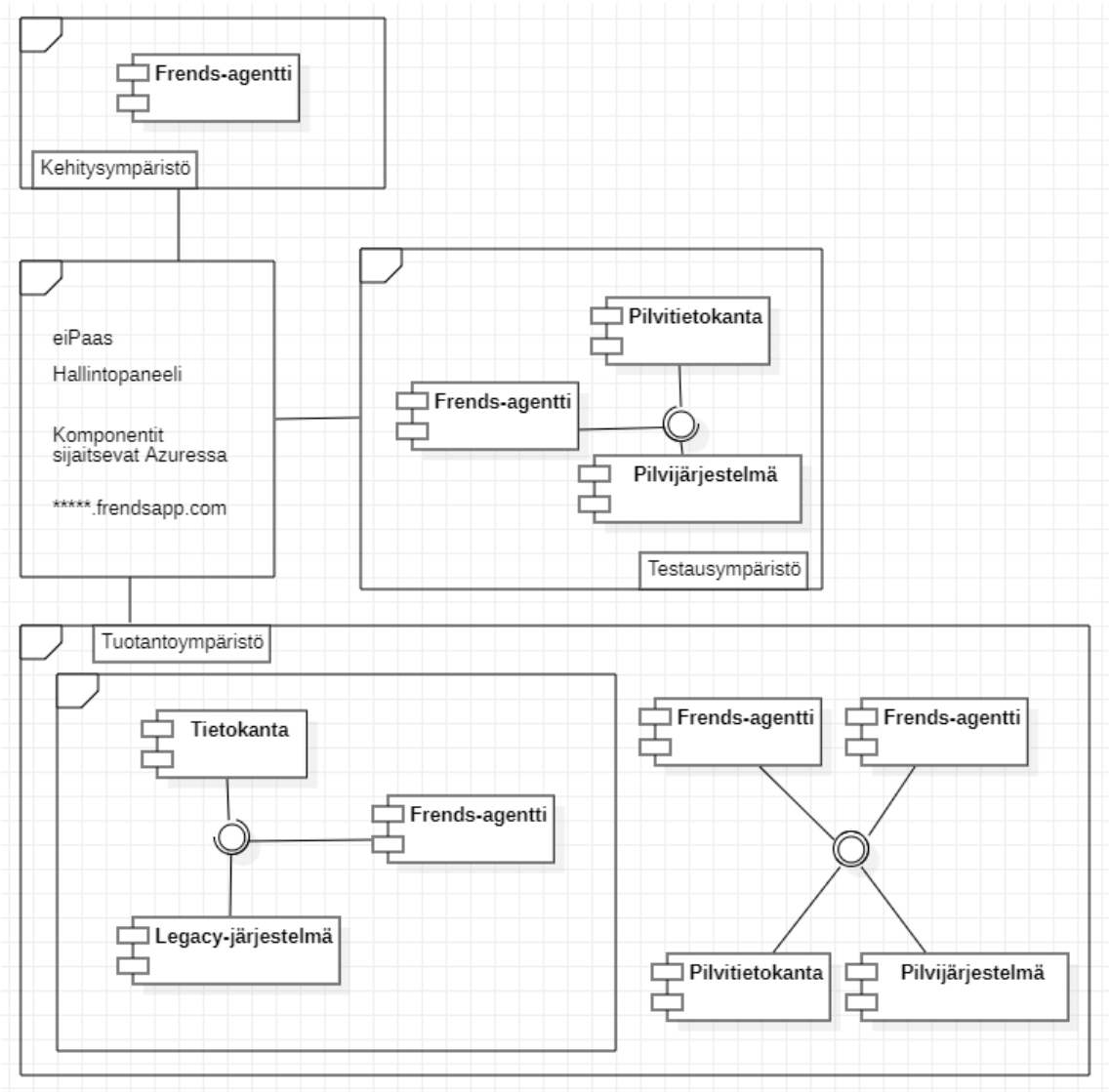
Kuva 3. Esimerkki Frends-prosessista, jossa vastaanotettu sanoma kirjoitetaan tiedostoon ja lähetetään kohteelle aliliittymän avulla. Sanoman lähetyksen onnistuessa liittymä lopettaa suorituksen. Muulloin suoritus päättyy virheeseen, joka on tarkasteltavissa liittymän lokista.



Kuva 4. Kuvakaappaus “File Write” -toiminnon konfiguraatioikkunasta, johon määritykset syötetään.

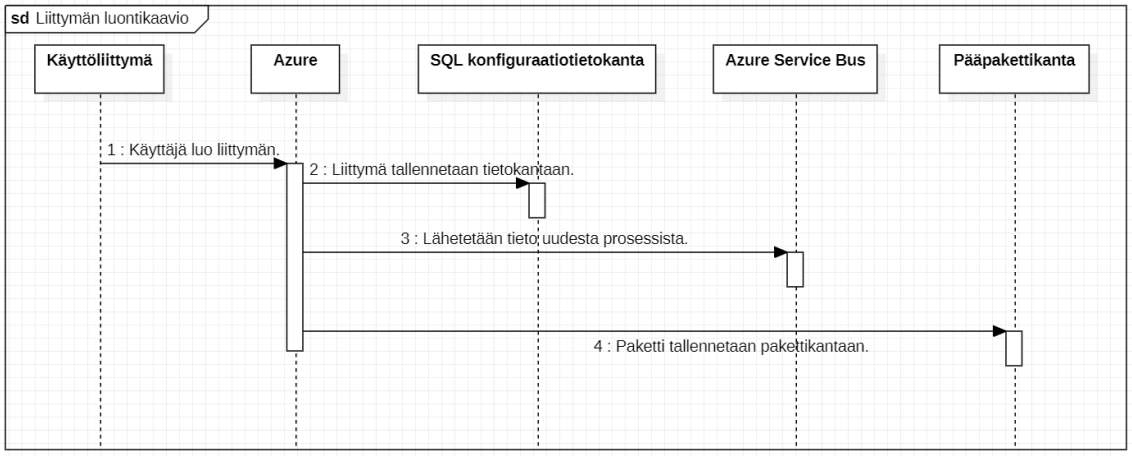
Frends-ympäristöjen tukiranka on Frends-agentit, jotka vastaavat prosessien ja integraatioiden suorittamisesta ympäristöissä. Näitä ympäristöjä ovat kehitys-, testaus- ja tuotantoympäristö ja niillä voi olla yksi tai useampi agentti valvomassa toimintaa. Mikäli agentteja on enemmän kuin yksi, jaetaan agentit agenttiryhmiin. Frends-agenttien määrä kertoo ympäristön tehokkuudesta, sillä agenttiryhmät jakavat työmäärää keskenään. Ne kommunikoivat tietokannan välityksellä tilastaan, jolloin vapaana oleva agentti vastaanottaa liittymän ajokutsun ja näin yksittäisen agentin työkuorma pienenee. Tämä toimintamalli muodostaa HA:n (“a High Availability Environment”), eli korkean saatavuuden ympäristön. (22.) Agentit jaetaan ympäristöjen mukaan, eli kehitys-, testaus- ja tuotantoympäristöillä on käytettävissään omat agenttiryhmät. Agenttiryhmien erottaminen ympäristökohtaiseksi varmistaa, että ympäristössä kulkeva tieto on turvassa sekaannuksilta. Se takaa myös, että muut ympäristöt jatkavat toimintaansa, vaikka

agenttiin menetettäisiin hetkellisesti yhteys. Frends-ympäristöissä voi olla käytössä yksi tai useampi agenttiryhmä. Nämä agenttiryhmät voivat sijaita eri alustoilla, esimerkiksi pilvipalvelussa ja “on-premise” eli paikallisessa datakeskuksessa. Esimerkiksi käyttäjä saattaa tarvita kehitys- ja testausympäristöön yhden agentin, sekä tuotantoympäristöön asennetaan kolme agenttia (kuva 5). Kaksi pilviagenttia muodostaa yhdessä agenttiryhmän ja yksi tuotannon agenteista asennetaan paikallisesti yrityksen tiloihin. Agentti on näin yhteydessä yrityksen käyttämään sisäverkkoon ja sillä on suora pääsy käytettyihin järjestelmiin ilman VPN-yhteyttä tai palomuurin muutoksia. (22; 23.)



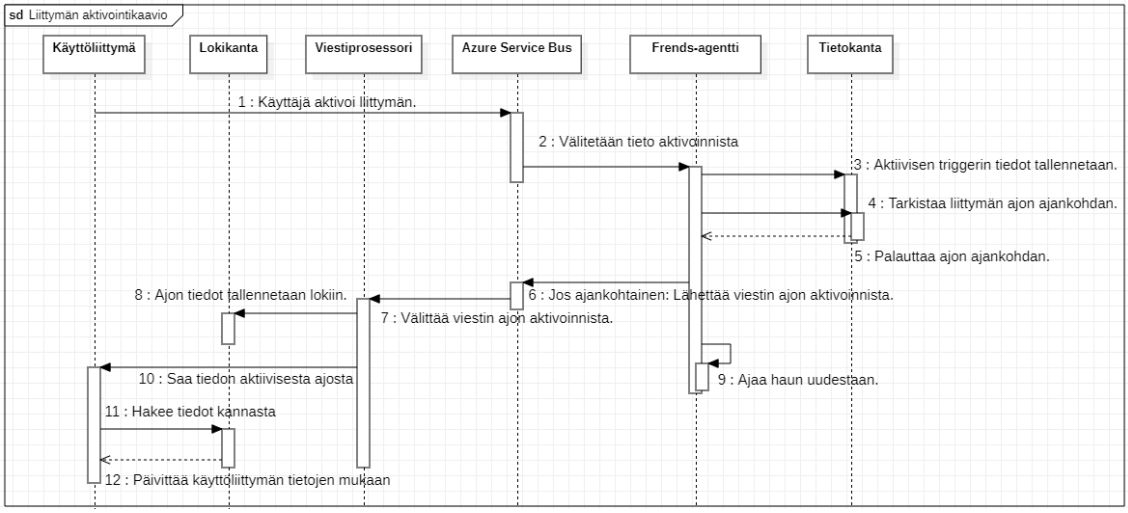
Kuva 5. Havainnollistava kuva esimerkkitapauksen ympäristöjen kokoonpanosta.

Luodut liittymät tallennetaan ensin kehitysympäristöön (kuva 6) ja ne voidaan siirtää muihin kuten testaus- ja tuotantoympäristöön Frends-käyttöliittymän avulla. Frends-pilviagentit käyttävät Azure SQL -tietokantaa ja paikallisen datakeskuksen agentit käyttävät joko paikallista tai palvelimella sijaitsevaa SQL-kantaa, jota ne hyödyntävät ympäristöissä sijaitsevien liittymien tallentamiseen. Kun agentti saa tiedon uudesta liittymästä, sen tiedot haetaan käyttöliittymän konfiguraatiotietokannasta. (23.)



Kuva 6. Sekvenssikaavio liittymän luonnin kulusta.

Kun liittymä aktivoidaan toimintaan (kuva 7), Service Bus lähettää tiedon triggerin toiminnasta agentille. Ajastin-triggerillä toimivan liittymän agentti hakee sekunnin välein tiedon, tuleeko liittymä ajaa. Esimerkiksi, jos liittymä on ajastettu ajettavaksi joka päivä kello 12:00, vertaa agentti haun ajankohtaa liittymälle määritettyyn suoritusajankohtaan. Käynnistetty viestiporsessori poimii ajon tiedot sekä indeksin liittymälokiin. Ajon indeksi toimii suorituksen identiteettitietona, jolla prosessori erottaa suoritukset toisistaan. Nämä tiedot tallennetaan lokitietokantaan. Kun liittymän ajo päättyy, Frends-agentti lähettää tiedon Service Bus -palvelulle, joka välittää sanoman viestiprosessorille sekä käyttöliittymään. Tämä sanoma sisältää tarkemmat tiedot prosessin kulusta, kuten aloitus- ja päättymisajankohdan, onnistumisen sekä mahdollisesti esiintyneet virheet. (23.)



Kuva 7. Sekvenssikaavio liittymän aktivoinnin kulusta.

Frends on hybridi-integraatioalusta, mikä tarkoittaa, että palvelua pystyy hyödyntämään niin pilvialustalla kuin paikallisessa datakeskuksessa (22). Esimerkiksi vanhemmat “legacy”-järjestelmät voivat olla tiukasti suljetuissa paketissa tai ne eivät mahdollista rajapintojen käyttöä, jolloin ne eivät ole helposti integroitavissa uusien järjestelmien kanssa (24). Tällöin voi olla tarpeellista asentaa integraatioalusta samaan ympäristöön, jotta tietojen päivitys voi toimia esimerkiksi jaetun tietokannan kautta.

3.3 Integraatiot Frendsin avulla

Työ toteutetaan Frends-integraatioalustaa käyttäen, sillä sen graafinen käyttöliittymä mahdollistaa monimutkaisten toteutuksien teon ymmärrettävässä muodossa. Toteutus sisältää useampien rajapintojen käytön ja näiden yhdistämisen yhtenäiseksi putkeksi. Haasteena on näiden rajapintojen sovittaminen yhteen, ja tämä vaatii työkalut loogisen sovelluksen rakentamiseksi.

Lähteenä toimii Hubspot, jota käytetään varauksen vastaanottamiseen asiakkaalta web-lomakkeen muodossa. Hubspot on CRM-palvelu, joka tarjoaa markkinoinnin automaatiojärjestelmän, jossa yhdistyvät myynti, markkinointi, asiakaspalvelu sekä asiakasrekisteri yhdeksi kokonaisuudeksi. (25.) Tämä helpot-

taa tietojen analysointia ja myyntitoiminnan kehitystä. Hubspotin toiminnot perustuvat “Inbound”-markkinointiin, eli asiakaskeskeiseen markkinointiin, jossa pyritään auttamaan asiakasta jo ennen ostopäätöstä. Palvelu pyrkii houkuttelemaan web-sivulle vieraita, jotka muunnetaan “liideiksi” eli asiakasprofiileiksi ja lopulta uusiksi asiakkaiksi. (26; 27.)

Kohderajapintana toimii Microsoft Power Automate, jota käytetään automatisaation toteuttamiseksi Microsoft-tuotteisiin. Sillä voidaan yhdistää ja rakentaa automatisoituja kokonaisuuksia, sekä sitä voidaan käyttää osana integraatiota. (28.) Käytämme sitä tässä toteutuksessa viestien lähettämiseen ja päivittämiseen Microsoft Teams -kanavalla.

Toteutuksessa käytämme apurajapintana Microsoft Graph API:a, joka on Microsoftin tukema rajapinta käyttäjäkohtaisiin toimiin, kuten käyttäjätietojen luku, viestien lähetys, kanavien tarkastelu ja paljon muuta. (29). Toteutuksessa hyödynnämme tätä tikettien monitoroinnissa, eli tarkistamme tällä tiketin tilan kanavalta.

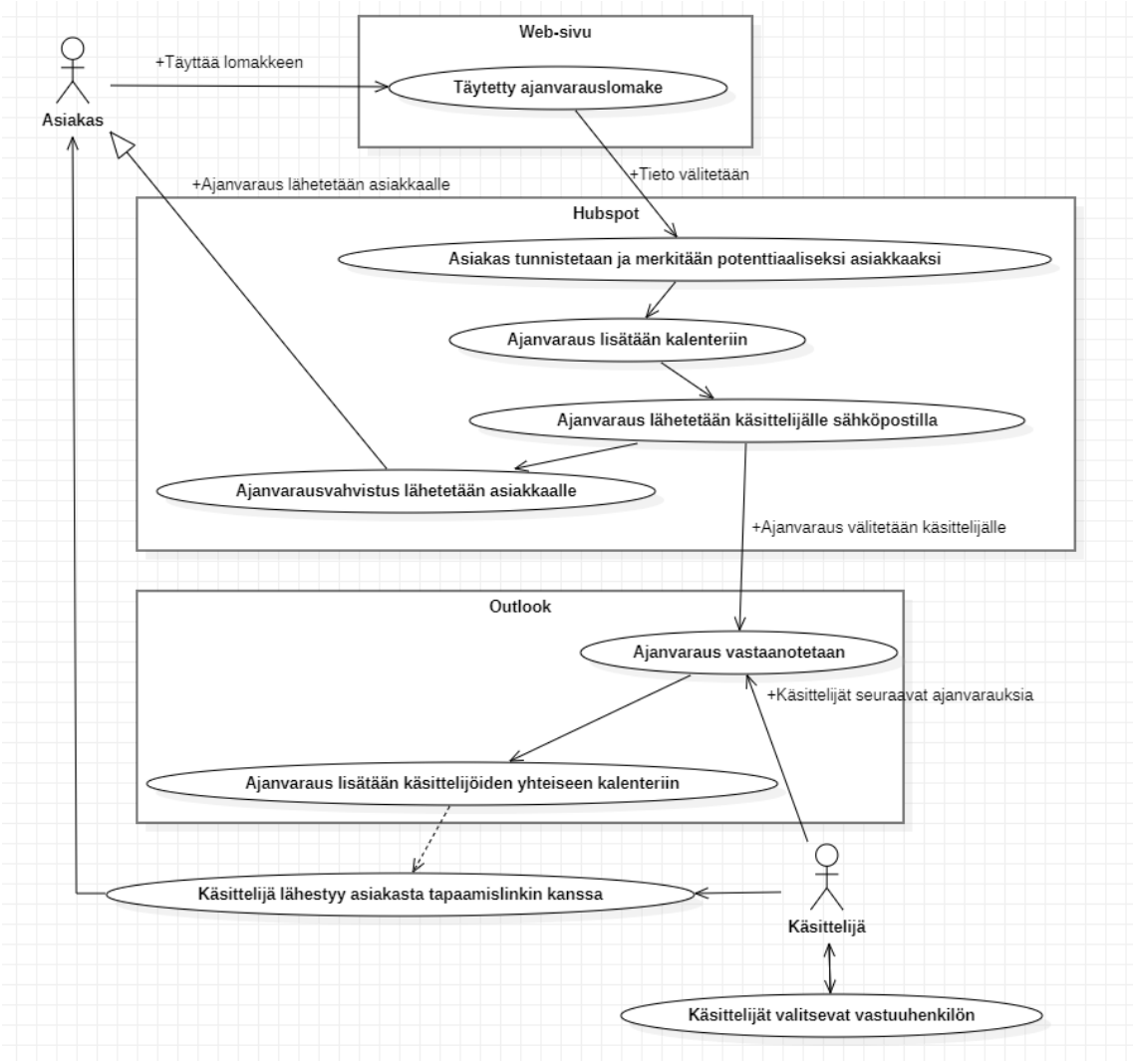
Hubspot- ja Power Automate -rajapinnat on mahdollista yhdistää toisiinsa ilman väli-integraatiota. Tämä olisi mahdollista määrittämällä Power Automate -rajapinta Hubspot-palvelun kohteeksi, mutta se jättäisi paljon vaadittua toiminnallisuutta uupumaan. Toteutus jättäisi monitoroinnin toteuttamatta, eli reagoimattomista viesteistä ei saataisi lähetettyä muistutuksia. Myös virheiden seuranta ja niistä toipuminen vaikeutuisi. Yksittäiset yhteyshäiriöt mahdollisesti estäisivät tiedon saapumisen kohteeseen. Frends ei tarjoa ainoastaan alustaa monitoroinnille, vaan myös virheenhallinnalle, joka palautuu hetkellisistä yhteysvirheistä. Frends määritetään tallentamaan sanoma ja lähettämään se seuraavassa ajossa uudestaan. Tämä vähentää virheiden määrää ja takaa sen, ettei ajanvarauksia pääse katoamaan. Virheenhallinta myös tuo virheen käyttäjän tietoisuuteen, mikäli se estää prosessin toteutumisen. Frends saadaan helposti yhdistettyä näihin rajapintoihin HTTPS-kutsuilla, joita Frends pystyy vastaanottamaan ja lähettämään eteenpäin.

4 Ajanvarauksen integraatio

Ajanvarauksen automaation integraatio toteutetaan, sillä nykyinen prosessi vaatii ajanvarauksien käsittelijöiltä eli toteutettavan integraation käyttäjiltä käsin tehtyä työtä. Tämä on vältettävissä laajentamalla nykyistä automaatiota, joka mahdollistaa ajanvarauksen lisäämisen käyttäjien yhteiseen kalenteriin. Laajennus vaatii integraation teon, joka välittää ajanvarauksen käyttäjien saataville ja seuraa ajanvarauksen tilaa. Tässä luvussa käsittelemme työn lähtökohtia, suunnittelua, työvaiheita, virheiden hallintaa ja lopulta työstä syntynyttä ohjelmistointegraatiota.

4.1 Lähtökohdat ja suunnittelu

Jotta automaatio voidaan toteuttaa, on meidän ensin tarkasteltava, mitä nykyinen prosessi pitää sisällään. Nykyisellään se koostuu Hubspot-palvelun tarjoamasta internetlomakkeesta, jolla ajanvaraus vastaanotetaan potentiaaliselta asiakkaalta. Tähän käytetään palvelun tarjoamaa ajanvaraustyökalua. (30.) Sitten palvelu lähettää varauksen käyttäjien yhteiseen kalenteriin ja lähettää asiakkaalle ajanvarausvahvistuksen, jossa kerrotaan käyttäjän ottavan myöhemmin yhteyttä tapaamisesta. Nopea vastausaika ajanvarauksen vastaanottamisesta on tärkeää, sillä se lisää asiakastyytyväisyyttä. Tämän vuoksi käyttäjät seuraavat päivittäin yhteistä kalenteria ja sopivat tapaamisista keskenään. Sitten käyttäjä lähestyy asiakasta sähköpostitse tapaamiskutsun kanssa (kuva 8).

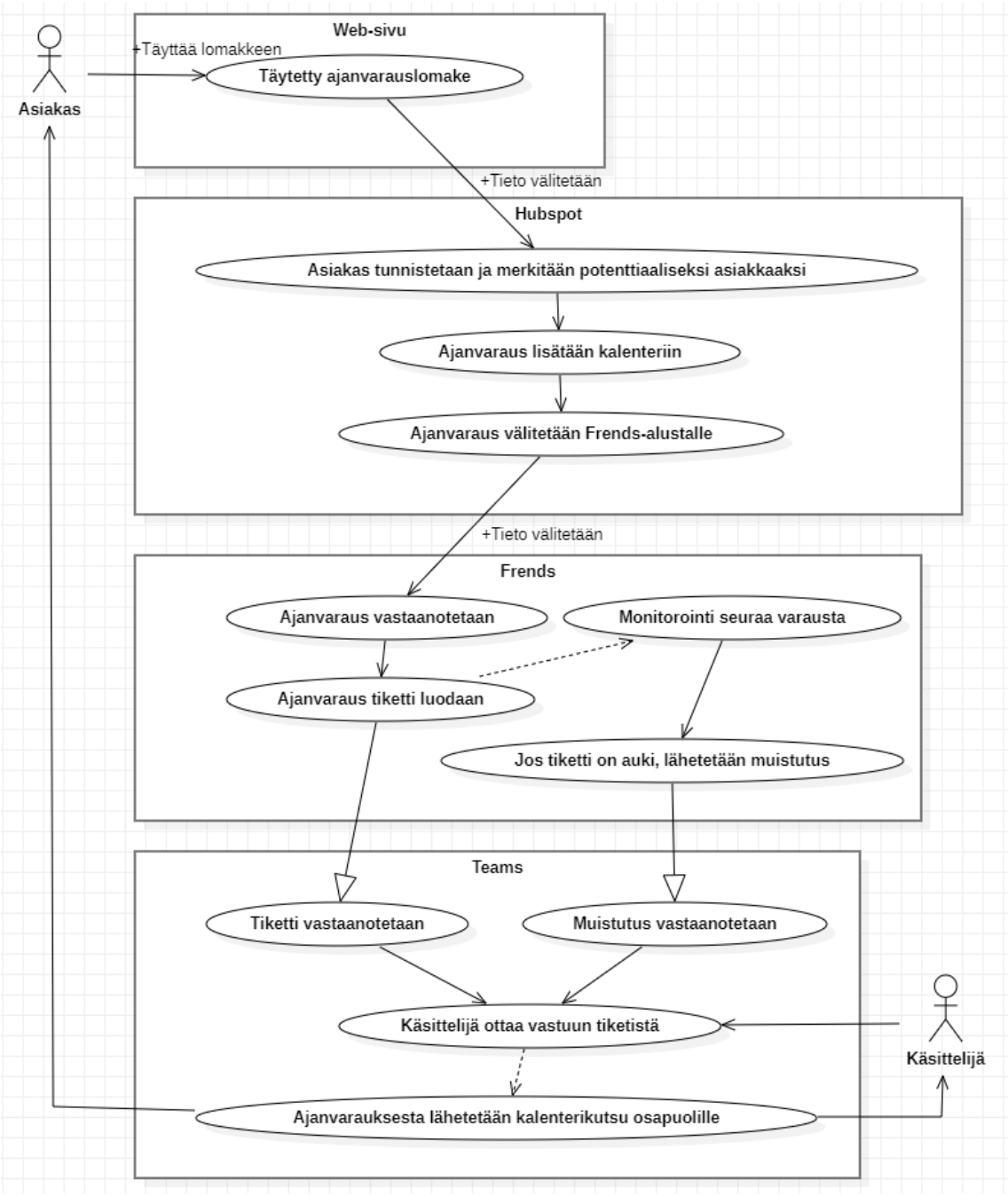


Kuva 8. Havainnollistava kaavio ajanvarauksen tapahtumien kulusta.

Nykyinen tiedonkulku tukeutuu vahvasti ajanvarausten käsittelijöiden väliseen kommunikaatioon ajanvarausten hallinnoinnista ja niiden tilasta. Tämä jättää mahdollisuuden käyttäjävirheille ja sekaannuksille. Esimerkiksi jos useampi käsittelijä ottaisi vastuulleen ajanvarauksen tai ilmoittaisi siihen halukkuutensa, mutta ajanvarauksen vastuuhenkilöä ei valittaisi, saattaa ajanvaraus jäädä käsittelemättä. Jos ajanvarauksia vastaanotetaan odotettua enemmän, ovat varaukset riskissä kadota sähköpostin ja eri kalenterien vuoksi. Nämä riskit voidaan välttää automatisoidulla prosessilla, sillä varaus saadaan tuotua käyttäjien tietoisuuteen reaaliaikaisesti ja tieto ajanvarauksen käsittelystä välitettyä niin asiakkaalle kuin muille käyttäjille. Toteutus tulee helpottamaan ja nopeuttamaan käyttäjien työntekoa, sillä se vähentää sekaannuksia ja varmistuksia siitä, onko

ajanvaraus jo jonkun vastuulla. Integraatio rakennetaan mahdolliset virhetilanteet ennakoiden, sillä useampien rajapintojen toteutus altistaa sen yhteysvirheille. Näihin varaudutaan rakentamalla monitorointi tallentamaan ajanvarauksen tiedot ja lähettämään sen uudestaan myöhemmässä ajossa. Monitoroinnin tulee myös seurata ja muistuttaa käyttäjiä, mikäli ajanvarauksia ei määritetyn ajan sisään käsitellä. Integraation tulee validoida syötetty sanoma, jotta se sisältää kaikki vaaditut tiedot, eivätkä ulkopuoliset pääse hyödyntämään toteutusta lähettääkseen likaista tietoa järjestelmään. Myös käyttäjäkokemuksen tulee olla sujuva ja selkeästi ymmärrettävä. Käyttäjien saama ajanvaraus tulee päivittää kuitatuksi, kun käyttäjä on sen käsitellyt.

Uuden toteutuksen suunnitelmassa hyödynnämme olemassa olevaa Hubspot-lomaketta asiakkaan ajanvarauksen vastaanottamiseen. Lomakkeen täyttäminen käynnistää tapahtumaketjun (kuva 9). Laajennamme Hubspot-toteutusta lähettämään tiedot Webhook-kutsulla, eli POST-tyyppisellä HTTP-kutsulla, Frends-palvelulle. Tämä käynnistää Frends-liittymän, joka luo varaukselle tiketin ja lisää sen monitoroinnin seurantalistalle. Tiketin tiedot välitetään eteenpäin Microsoft Teams -viestipalvelulle, jossa se tuodaan käyttäjän saataville hyödyntämällä Teams-kanavilla käytettyjä “adaptive cards”, eli interaktiivisia kortteja. Tämän interaktiivisen kortin lähettämiseen ja valvomiseen käytämme Power Automate -palvelua, sillä interaktiiviset kortit vaativat taustalla järjestelmän, joka seuraa niiden tilaa ja päivittää ne kuitatuksi. Interaktiivinen kortti sisältää asiakkaan tiedot, ajanvarauksen ajankohdan ja napin, jolla käyttäjä voi lunastaa ajanvarauksen vastuulleen. Samalla monitorointi seuraa tiketin tilaa tunnin välein ja lähettää muistutusviestin, joka sisältää tiketin numeron ja viitteen alkuperäiseen viestiin. Kun käyttäjä on kuitannut varauksen, saadaan käyttäjän ja asiakkaan tiedot interaktiivisen kortin kautta, jolloin ajanvarauksen kalenterikutsu voidaan lähettää tiedoista saatuihin sähköposteihin. Lopulta tiketti poistetaan seurattavien listalta, monitoroinnin seuraavalla ajastetulla ajolla. Tämä päättää ajanvarauksen tapahtumien kulun.



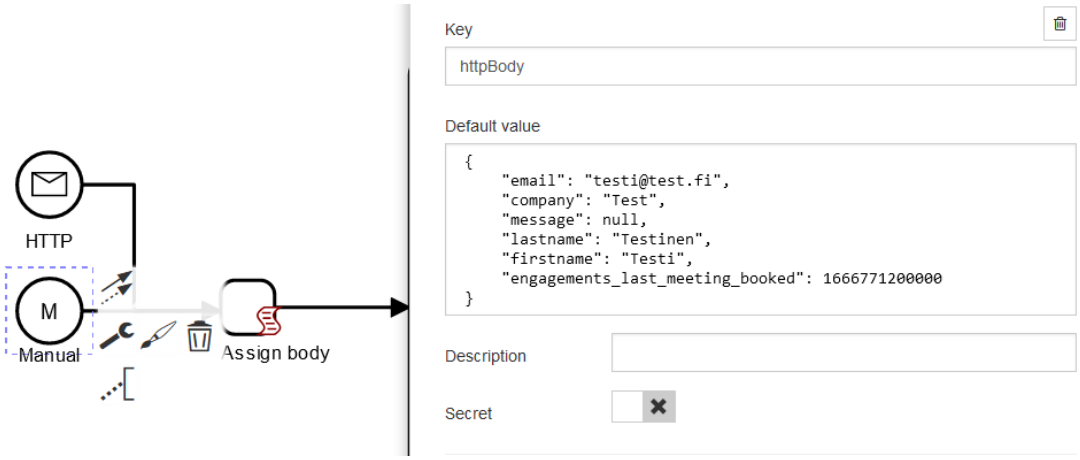
Kuva 9. Havainnollistava kaavio uuden toteutuksen suunnitellusta kulusta.

4.2 Työvaiheet

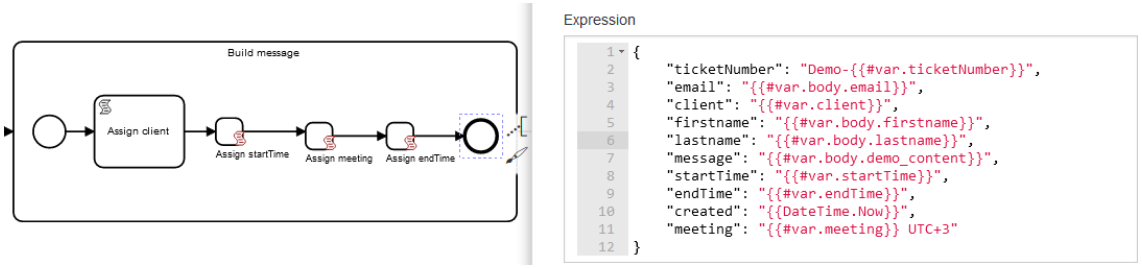
Työn toteutus koostuu viidestä prosessista ja kolmesta käytetystä alustasta. Tarkastelemme toteutuksen rakentumista kolmessa työvaiheessa, jotka on jaettu toiminnallisuuden perusteella:

- Frends-rajapinta, joka vastaanottaa ajanvarauksen Hubspot-palvelulta. Se käsittelee varauksen tiketiksi ja lähetetään Power Automate -prosessille, joka julkaisee interaktiivisen kortin käyttäjien Teams-kanavalle.
- Frends-monitori ja monitoroinnin rakentuminen prosessien välille. Monitori hallitsee tikettien käsittelyä. Se hyödyntää Graph API -rajapintaa tikettien seurannassa ja Power Automate -prosessia muistutusviestien lähettämiseen.
- Hubspot-ajanvaraustyökalun automaation laajennus ajanvarauksen tietojen lähettämiseksi Frends-liittymän käsittelyyn.

Työ aloitetaan Frends-rajapinnan toteutuksesta, koska ilman toteutusta myöhemmät prosessien osat eivät voi käynnistyä. Rajapinnan toteutus voidaan rakentaa ilman Hubspot-toteutusta, sillä liittymän käynnistys voidaan suorittaa Frends-ympäristöstä manuaalisesti tai käyttämällä API-työkalua kuten Postman. Tässä vaiheessa muotoillaan testaamiseen tarkoitettu JSON-muotoinen sanoma, jota hyödynnetään kehityksessä (kuva 10). Sanoman validointi ja sen tarkempi muoto määräytyy Hubspot-prosessin lähettämän sanoman perusteella. Liittymän kehitysvaiheessa testisanoma muutetaan käsiteltävään muotoon, jolloin siitä voidaan poimia lähetettävän viestin tiedot (kuva 11).



Kuva 10. Liittymän manuaaliin käynnistykseen käytettävä testisanoma.



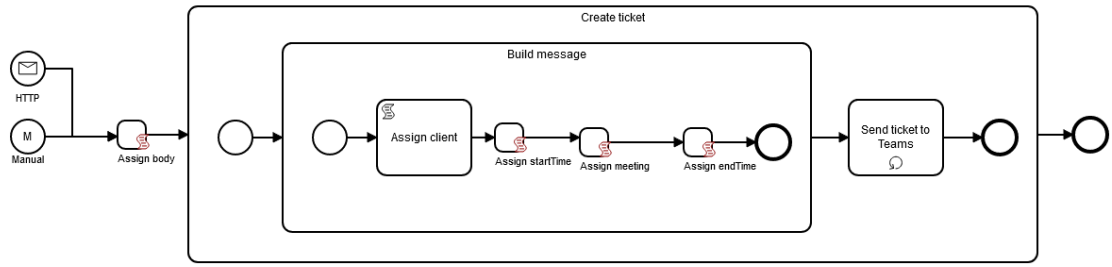
Kuva 11. Viesti rakentuu hakemalla sisään tulleesta JSON-sanomasta tarpeelliset tietueet ja asettamalla ne muuttujiin. Lopuksi muuttujat muodostetaan yhtenäiseksi viestiksi.

Kun liittymä saadaan käynnistettyä sisään tulevalla sanomalla, seuraavaksi toteutamme sanoman käsittelyn ja viestin lähetyksen Teams-kanavalle. Viestin lähetykseen palvelu tarjoaa muutaman eri vaihtoehdon toteutukselle. Vaihtoehtoisia menetelmiä ovat:

- Teams-kanavien tarjoamaa “Incoming Webhook”-lisäosaa, joka mahdollistaa viestien lähetyksen kanavalle POST-tyyppisillä kutsuilla (31).
- Microsoft Power Automate -automaatio-ohjelma, joka on käyttäjään yhdistetty graafinen liittymä, joka mahdollistaa automatisoitujen prosessien kehittämisen ja hyödyntämisen Microsoftin tuotteissa (28).
- Microsoftin tarjoama Graph API -rajapinta, joka mahdollistaa käyttäjäkohtaisten toimintojen suorittamisen rajapintaa käyttäen (29).

Näistä menetelmistä Power Automate -toteutus valikoitui käyttöön, sillä Webhook-toteutus ei mahdollista viestien päivittämistä kuittauksen jälkeen. Graph API -toteutus taas tukee viestien päivityksen, mutta interaktiivisten korttien nappeja ei saatu rekisteröimään painajan tietoja. Toteutus saatiin lähettämään ilmoituksen kuittauksesta, mutta se olisi vaatinut käyttäjältä erillisen syötteen tämän tunnistamiseksi. Power Automate tarjoaa korttien lähetyksen, päivityksen, seurannan ja käyttäjän tunnistuksen. Palvelun rajoitettujen toimintojen vuoksi ei palvelua voida hyödyntää tietojen välittämiseksi tapahtumista monitoroinnille tai aktivoimaan liittymiä. Monitorointi saadaan Graph API -rajapinnan avulla rakennettua niin, ettei se vaadi sisään tulevaa sanomaa käynnistyäkseen. Power Automate -palvelu valikoitui viestien lähetykseen ja Graph API -rajapinta viestien tilan seuraamiseen. Tämä toteutus ei vaadi käyttäjältä syötteenä tietoja, ja se vähentää erilaisten käyttäjävirheiden, kuten kirjoitusvirheiden,

esiintyvyyttä. Saamme aktivoitua Power Automate -alustalla rakennetun prosessin lähettämällä rajapintaan POST-tyyppisen HTTP-kutsun (kuva 12)(liite 1 kuva 1).

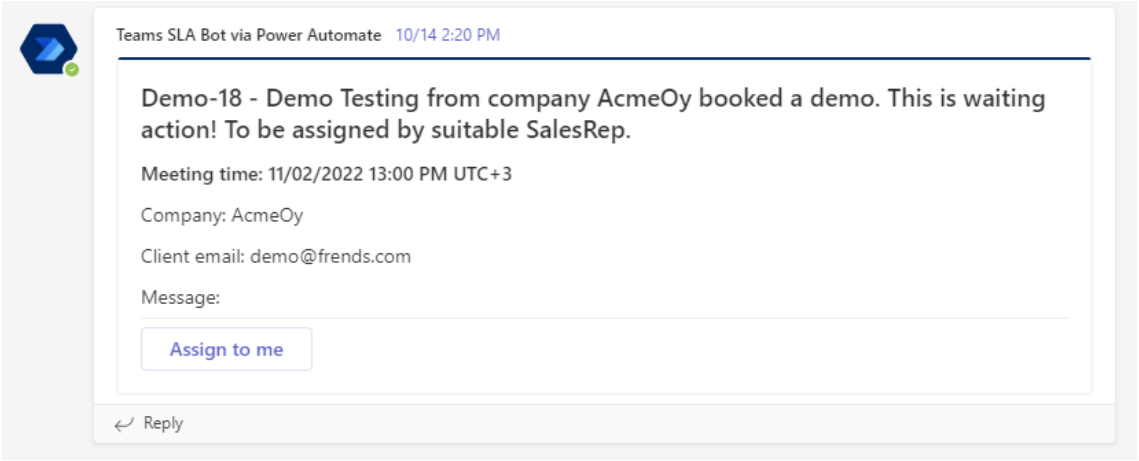


Kuva 12. Liittymässä vastaanotettu sanoma käsitellään viestiksi, joka lähetetään Power Automate -prosessille HTTP POST -kutsuna.

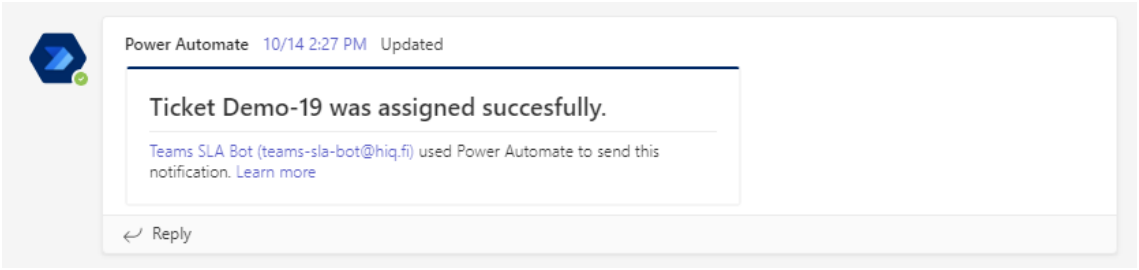
Sanoma vastaanotetaan Power Automate -alustalla, jossa interaktiivisen kortin lähetysprosessi käynnistyy. Sisääntullut sanoma validoidaan ja käsitellään ennen interaktiivisen kortin kokoamista ja lähetystä Teams-kanavalle. Kortin lähetyksen jälkeen prosessi jää odottamaan korttiin kohdistuvia toimintoja. Nämä toiminnot saadaan määriteltyä valmiilla toteutuksilla, jotka vaativat Frends-liittymien tavoin vain konfiguraation (kuva 13).

Kuva 13. Interaktiivisen kortin lähetys ja kuittauksen jälkitoimenpiteet.

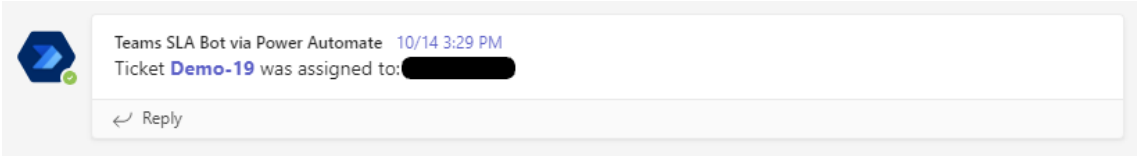
Kun käyttäjä on kuitannut tiketin itselleen painamalla interaktiivisesta kortista löytyvää nappia (kuva 14), prosessi hakee käyttäjän tiedot ja lähettää kalenterikutsun niin tiketistä löytyvään asiakkaan sähköpostiin kuin käyttäjän kuittauksesta saatuun sähköpostiin. Interaktiivinen kortti päivittyy, jonka jälkeen se ei ole enää muiden käyttäjien kuitattavissa (kuva 15). Tästä lähetetään Teams-kanavalle kuittaus (kuva 16), joka kertoo käyttäjille lähetyksen onnistuneen ja ajanvaraukselle määritetyn vastuuhenkilön. Näin käyttäjät pystyvät seuraamaan ja tarkastamaan, kenen vastuulla tiketti on.



Kuva 14. Ajanvarauksesta lähetetty interaktiivinen kortti Teams-kanavalla.

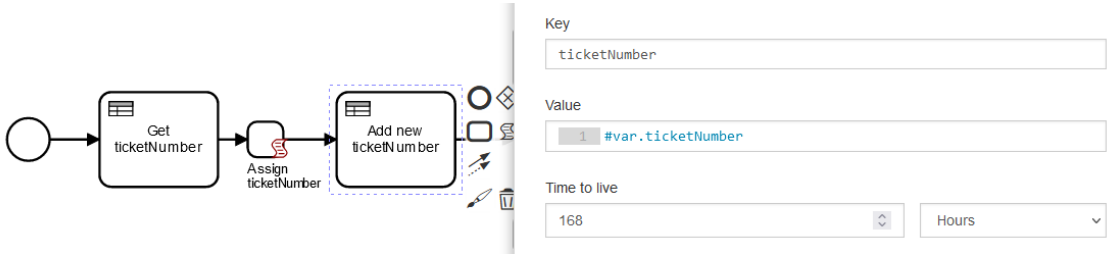


Kuva 15. Kuittauksen jälkeen päivitetty interaktiivinen kortti.



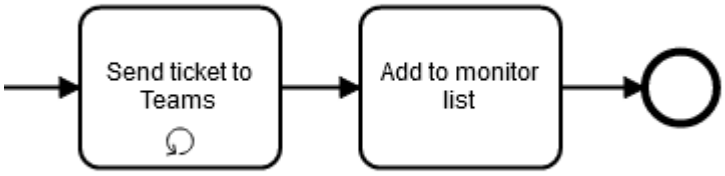
Kuva 16. Teams-kanavalle lähetetty kuittaus tiketin onnistuneesta käsittelystä.

Monitoroinnin toteutus aloitettiin tikettien rakennuksesta, joka vaatii ensin tiketti-numeron luomisen. Tätä varten toteutettiin perinteinen indeksilaskuri, joka jokaisella käyttökerralla kasvattaa tallennettua indeksiä. Numero tallennetaan Frends-alustan tietokantaan käyttämällä “Shared State Task” -toiminnallisuutta (kuva 17). Se tarjoaa mahdollisuuden tallentaa tietoa helposti avainpareina (32). Tiedolle annetaan tunnistusavain, jonka avulla arvo saadaan haettua liittymän seuraavassa ajossa. Indeksiarvoa säilytetään tietokannassa kuukauden ajan, jonka jälkeen arvo poistetaan, mikäli sitä ei tässä ajassa päivitetä. Mikäli arvoa ei löydetä seuraavassa ajossa, indeksilaskuri nollautuu.



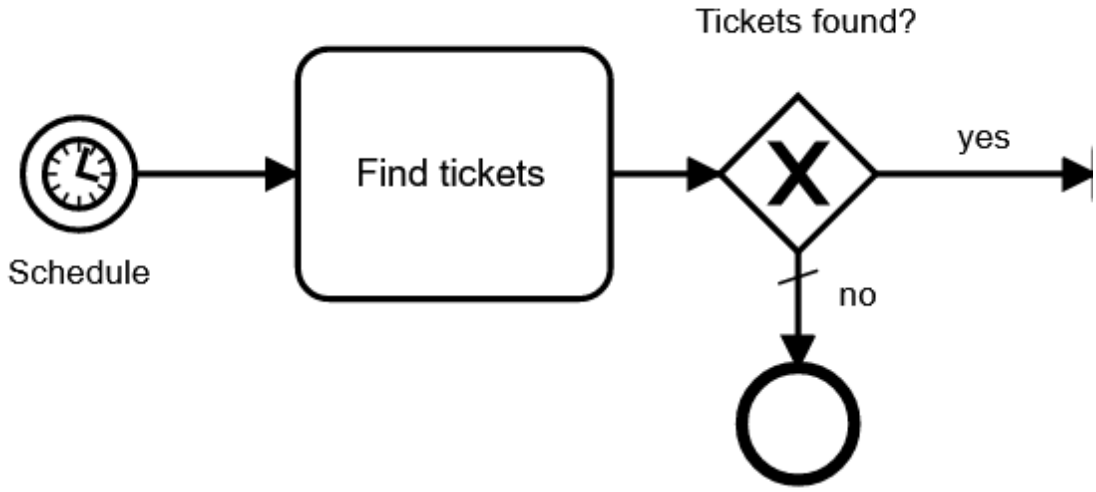
Kuva 17. “Shared State Task” -toiminnollisuuden tietokannasta haetaan tallennettu tikettinumero. Numeroa kasvatatetaan yhdellä ja se lisätään muuttujaan. Lopuksi uusi numero päivitetään haetun arvon tilalle.

Rakennetut tiketit tallennetaan ajon lopuksi monitorin seuraamaan hakemistoon (kuva 18). Tiketit tallennetaan omina tiedostoinaan, jolloin niitä voidaan seurata myös muista liittymistä. Tiedosto sisältää ajanvarauksen tiedot, kuten asiakkaan yhteystiedot, tapaamisajan, tikettinumeron sekä luontiajan. Monitorointi rakennetaan seuraamaan tätä hakemistoa, jotta se ajetaan vain, kun seurattavia tikettejä on.



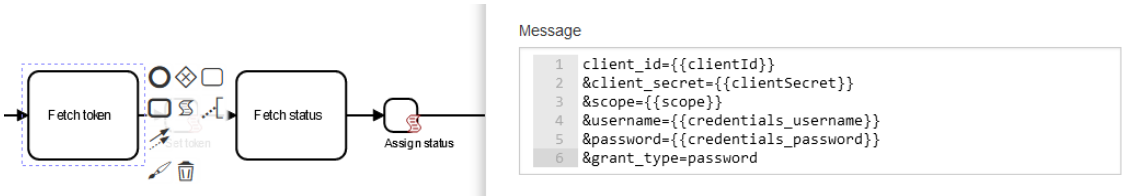
Kuva 18. Tiketin lähetyksen jälkeen se tallennetaan tekstitiedostona paikalliseen hakemistoon.

Monitorointia varten luodaan oma liittymä, jossa tallennetut tiedostot käsitellään (liite 1, kuva 2). Jotta emme suorita monitorointia työajan ulkopuolella, määritetään liittymä käynnistymään ajastuksella. Näin määritämme liittymän käynnistymään tunnin välein, aikavälillä kello 8–17 Helsingin aikaa (kuva 19). Kun liittymä on käynnistynyt, tarkastamme, löytyykö hakemistosta haettuja tiedostoja. Mikäli haulla löytyy hakemistosta monitoroitavia tiedostoja, palauttaa alliliittymä tiedot näistä. Mikäli tiedostoja ei löydy, lopetetaan liittymän suoritus.



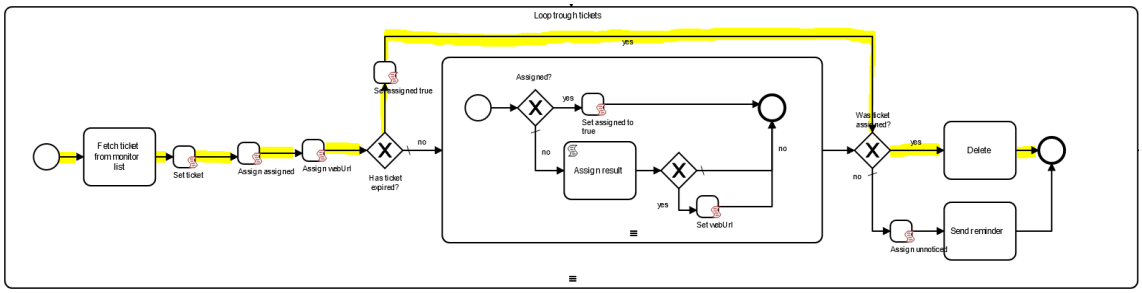
Kuva 19. Liittymä käynnistetään ajastuksella tunnin välein toimistotuntien sisällä. Sen jälkeen haetaan hakemistossa olevat tiketit ja tarkastetaan tulokset. Suoritus lopetetaan, jos hakemistosta ei löydy tiedostoja.

Liittymän haettua löydetyt tikettitiedostojen tiedot, haemme autentikointitunnuksen, jotta Teams-kanavalta saadaan haettua tikettiviestit näiden tilan tarkastamiseksi. Tikettiviestit haetaan POST-tyyppisen HTTP-kutsun avulla muuttujaan. Viestien hakuun käytämme Graph API -rajapintaa, joka on Microsoftin tarjoama rajapinta käyttäjäkohtaisten toiminnallisuuksien hallinointiin ja toteutukseen. Rajapinnan hyödyntäminen vaatii käyttäjältään Azure AD -tunnukset ja Azure-sovelluksen rekisteröinnin. Näiden avulla saamme haettua tarvitun autentikaatiotunnuksen ja käytettyä sitä API-rajapinnan kutsun todentamiseksi. Tämä vaatii myös applikaation rekisteröinnistä asetettavat oikeudet, jotka oikeuttavat tunnuksen käyttäjän suorittamaan sallitut toiminnot. (33.) Tunnukset ovat voimassa vain tunnin ajan niiden hakemisesta (34), jonka vuoksi jokaista ajoa varten haetaan oma tunnus.

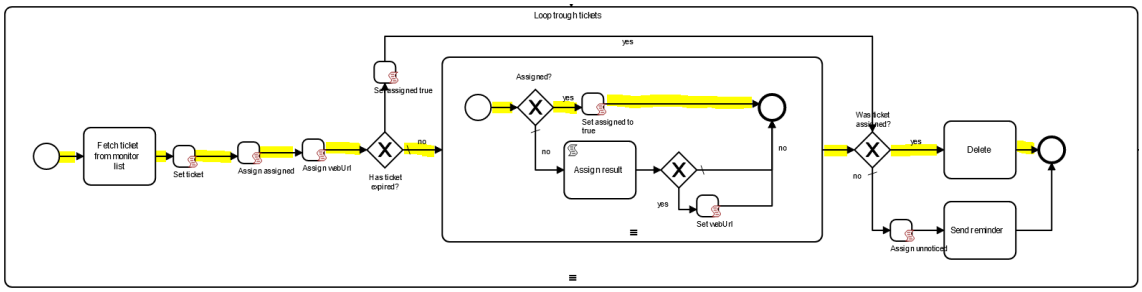


Kuva 20. Ensin haetaan autentikaatiotunnus AD-käyttäjälle rekisteröidystä applikaatiosta POST-tyyppisellä kutsulla. Tämän jälkeen rajapinnalle lähetetty GET-kutsu todennetaan saadulla tunnuksella.

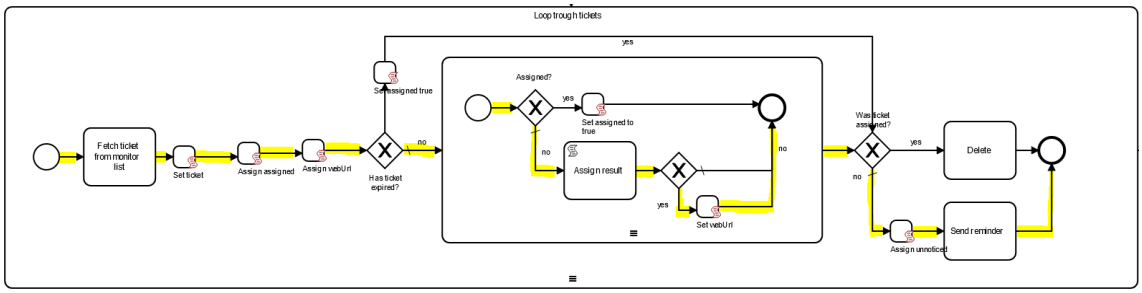
Tämän jälkeen käymme tiedostot läpi käyttämällä iterointia. Jokaisen kierroksen alussa haemme vuorossa olevan tikettitiedoston sisällön tarkasteltavaksi. Ensin tarkistamme tiketistä löytyvän tapaamisen alkamisajan, ja mikäli se on ohitettu, merkitään tiketti vanhentuneeksi, jolloin se poistetaan kierroksen päätteeksi monitorointilistalta (kuva 21). Tämä nopeuttaa kierroksien kulkua, eikä käyttäjien kanava täyty vanhentuneiden ajanvarausten muistutusviesteistä. Kun olemme tarkastaneet tiketin alkamisajankohdan ja todenneet tiketin olevan vielä voimassa, tarkistamme kanavalta haetuista viesteistä, onko tiketti jo kuitattu. Jos tiketin kuittausviesti löytyy, merkitsemme tiketin käsitellyksi. Tällöin tiketti poistetaan monitoroinnin listalta kierroksen päätteeksi (kuva 22). Mikäli kuittausviestiä ei löydy, poimitaan alkuperäisen tikettiviestin linkki ja tallennetaan muuttujaan (kuva 23). Käsittelemättömästä tiketistä lähetetään kierroksen päätteeksi muistutusviesti Power Automate -alustaa hyödyntäen (kuva 24). Muistutusviesti sisältää tallennetun linkin, jolloin käyttäjä pääsee tarkastelemaan alkuperäistä tikettiä vain napin painalluksella (kuva 25).



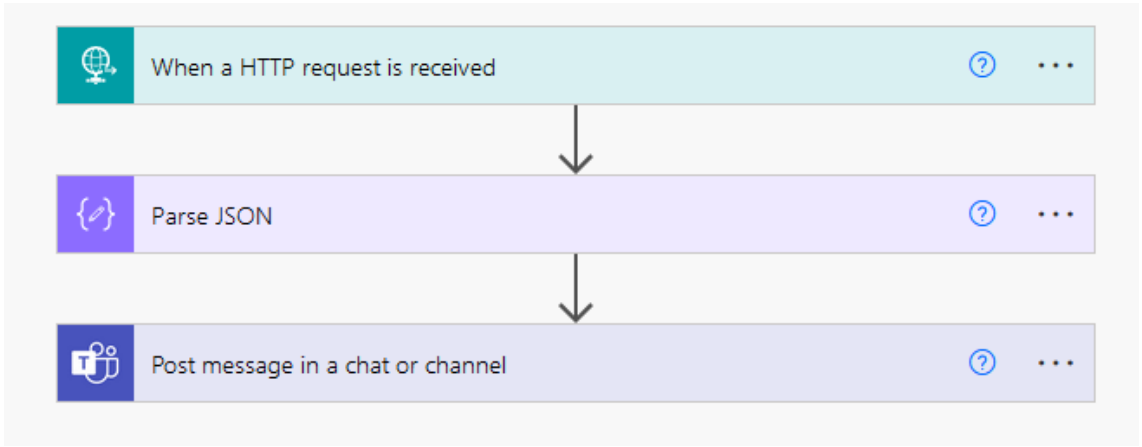
Kuva 21. Kierroksen alkuun tarkastetaan, onko tiketin tapaamisen alkamisajankohta ohitettu. Tällöin ohjelmisto ohjataan kuvan osoittamalle polulle. Tällöin tiketti merkitään vanhentuneeksi ja poistetaan kierroksen päätteeksi monitoroinnin seurannasta.



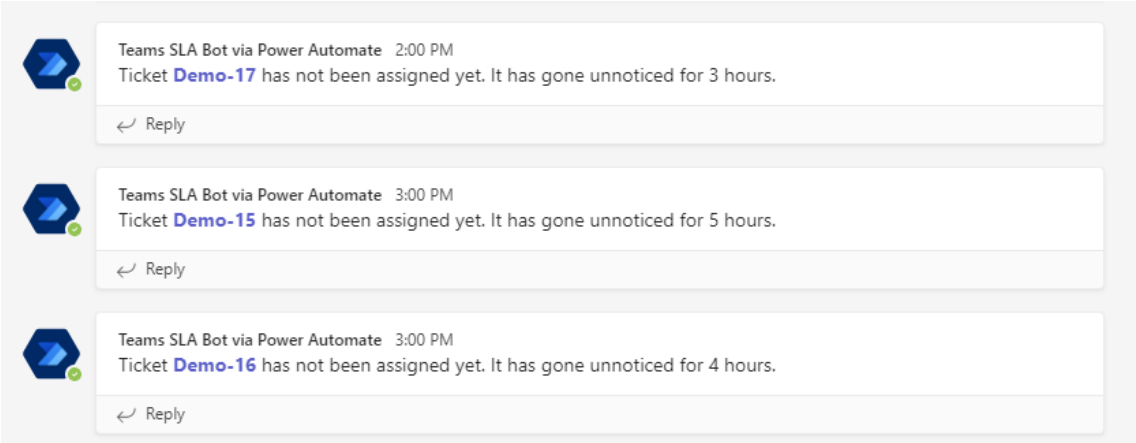
Kuva 22. Teams-kanavalta haetut viestit käydään läpi ja tarkistetaan, löytyykö niistä tiketin kuittausviesti. Kuittauksen löytyessä ohjelmisto ohjataan kuvan osoittamalle polulle ja tiketti merkitään käsitellyksi, jolloin se poistetaan kierroksen lopuksi monitoroinnin seurannasta.



Kuva 23. Kun kuittausviestiä ei löydy Teams-kanavalta haetuista viesteistä, ohjelmisto ohjataan kuvan osoittamalle polulle ja alkuperäisen tiketin viestilinkki otetaan talteen ja lähetetään se kierroksen lopuksi muistutusviestin mukana Power Automate -prosessin käsiteltäväksi.

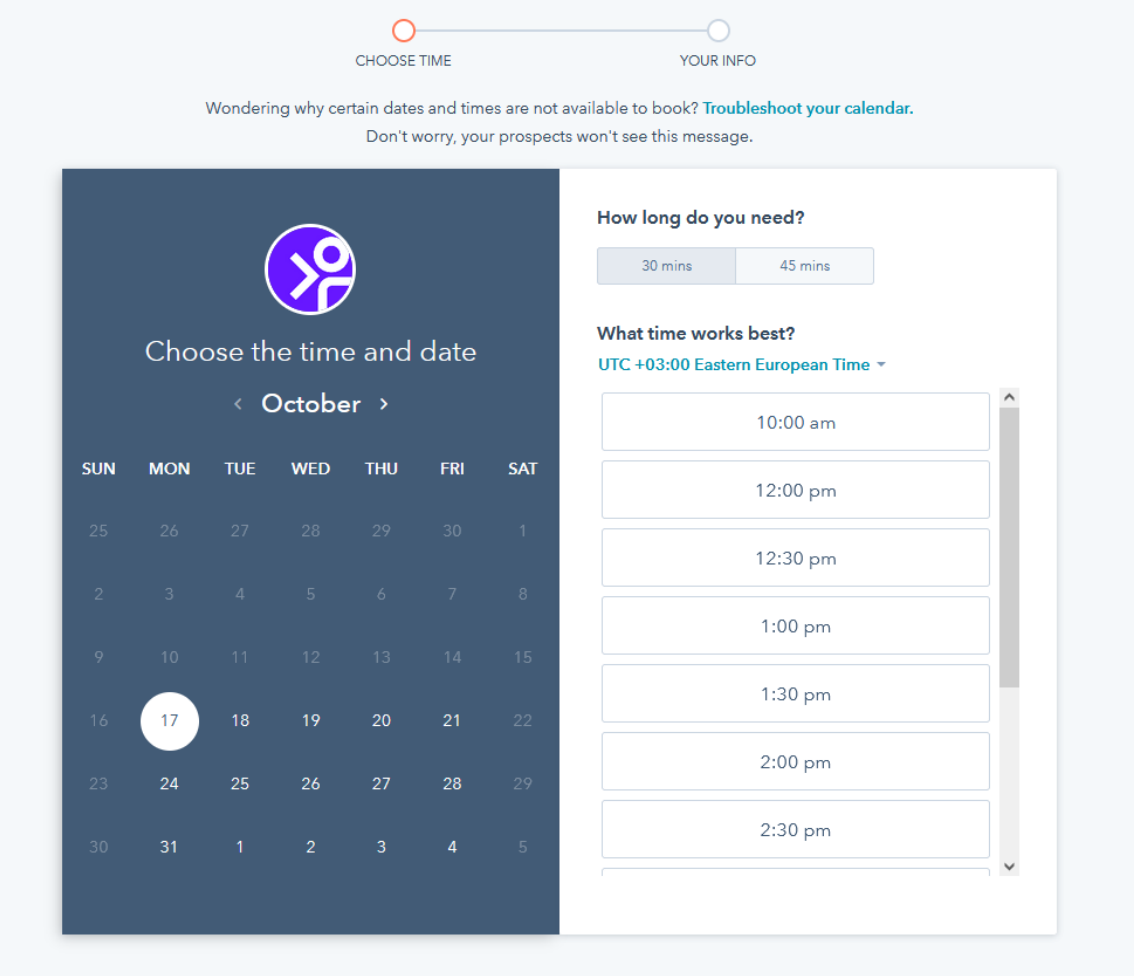


Kuva 24. Muistutusviestin lähetys Power Automate -alustalla. Sisääntuleva sanoma käynnistää prosessin, jossa sanoma käsitellään ja sen sisältämät tiedot lähetetään Teams-kanavalle muistutusviestin muodossa.



Kuva 25. Muistutusviestit Teams-kanavalla.

Kun prosessi on muutoin toiminnassa, voidaan siihen yhdistää käynnistävä prosessi, joka on Hubspot-palvelun ajanvaraustyökalun internetlomake. Tämä Hubspotin tarjoama työkalu automatisoi tietojen käsittelyn sekä kalenterikutsujen lähetyksen sähköpostitse. Sen luoma lomake sisältää ajanvalinnan (kuva 26), asiakkaan tietojen syötön ja lopuksi ajanvaraustyökalu lähettää asiakkaalle vahvistussähköpostin ajanvarauksesta. Varaus lisätään myös Hubspotin omaan kalenteriin, jolloin sama aika ei ole enää muiden varattavissa. Kalenteriin lisätty ajanvaraus välitetään erillisen tapahtumakulun määrityksen avulla myös käyttäjien sähköpostiin ja kalenteriin.

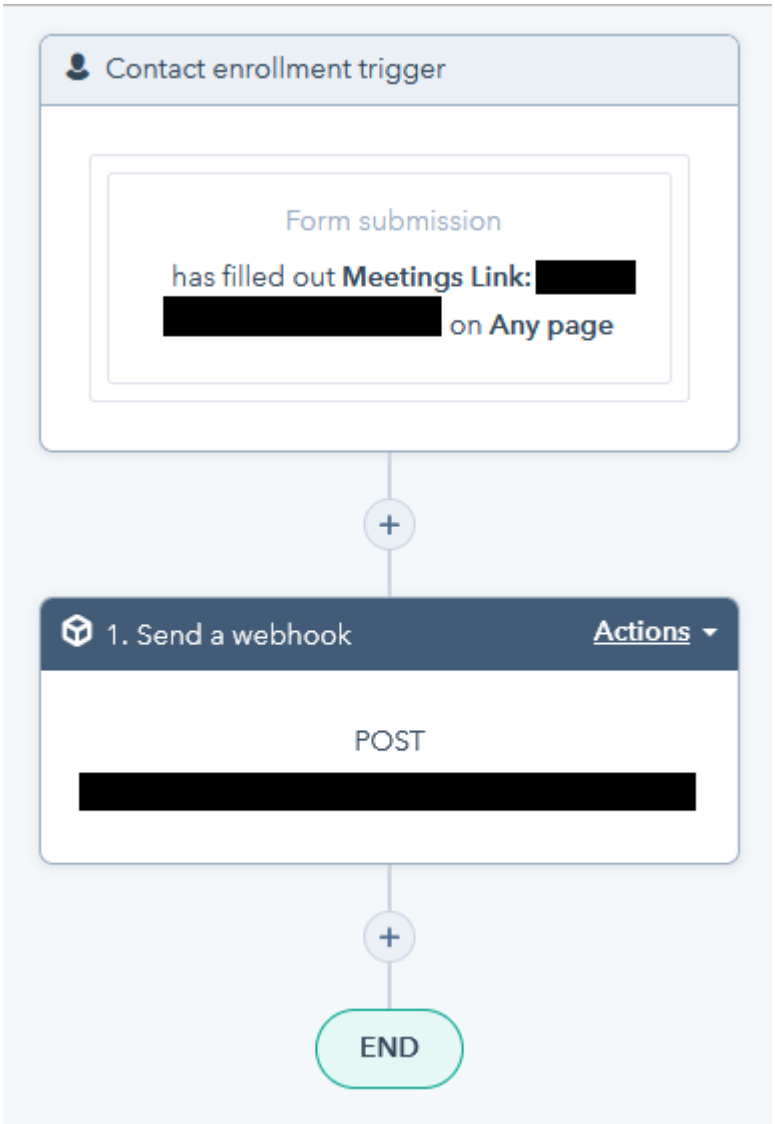


Kuva 26. Hubspot-ajanvarauksen valinta web-lomakkeessa.

Jotta pystymme laajentamaan olemassa olevaa toteutusta, on meidän tutustuttava Hubspot-palvelun tarjoamiin “Workflow”-toteutuksiin. Nämä ovat prosesseja, jotka käsittelevät tietoa ja tapahtumakulkua alustalla. Tapahtumakulkuja saadaan aktivoitua tuttuun tapaan erilaisilla tapahtumilla, joita voi olla tiedoissa tapahtuvat muutokset, lomakkeiden täyttäminen sekä kalenterivarausten ajankohtien lähestyminen. (35.) Ajanvarauslomakkeet toimivat prosesseissa samaan tapaan kuin lomakkeet ilman kalenteritapahtumia. Haasteena on saada haettua kaikki tarvittavat tiedot, sillä tapahtumakulut eivät varsinaisesti tue ajanvaraustyökalun tarpeita, sillä se luetaan vain lomakkeeksi. Tämä ilmenee esimerkiksi siten, että valittua aikaa ei saada välitettyä suoraan lomakkeesta vaan tämä vaatii kalenterityökalun käyttöä tietoja hakiessa. Useita tietoja ei saada tapahtumankulussa haetuksi, kuten ajanvarauksen kesto tai lomakkeeseen mää-

ritetyt erikoistietueet, joita ei sisällytetä Hubspotin luomaan “liidiin” eli asiakasprofiiliin. Nämä tiedot voidaan hakea tekemällä erillinen Hubspot API -kysely, joka vaatii autentikaatiosovelluksen eli “Private App” luomisen. Tällä applikaatiolla saadaan luotua kertakäyttöisiä tunnistusavaimia, joilla API-kutsut saadaan todennettua. (36.) Koska tiedot eivät ole täysin välttämättömiä tässä toteutuksessa, ei tätä sovellusta oteta käyttöön, sillä integraation toteutus kokonaisuudessaan sisältää jo paljon rajapintoja. Nämä puuttuvat tiedot saadaan lähetettyä muuntamalla lomakkeen erikoistietue asiakasprofiiliin sisällytettäväksi ja asettamalla tapaamiselle oletuskesto, jonka perusteella kalenterikutsu luodaan. Tällöin näitä tietoja varten emme tarvitse erillistä API-hakua.

Kun olemme tehneet tarvittavat muutokset ajanvarauksen lomakkeeseen, jotta saamme kaikki tarvitsemamme tiedot, määritämme tapahtumakulun (kuva 27). Kulku käynnistetään, kun ajanvarauksen lomakkeemme on täytetty ja lähetetty asiakkaan toimesta. Prosessi hakee lomakkeelle täytetyt tiedot ja tarkistaa kalenterista viimeksi lisätyn ajanvarauksen ajankohdan. Tiedot lähetetään Frendsliittymälle POST-tyyppisenä HTTP-kutsuna. Tämä sanoma sisältää vain ennalta määritellyt tiedot, jolloin sen validointi kohteessa helpottuu. Tapahtumakulun toiminnan saamme varmistettua helposti varaamalla ajanvarauslomakkeesta ajan.



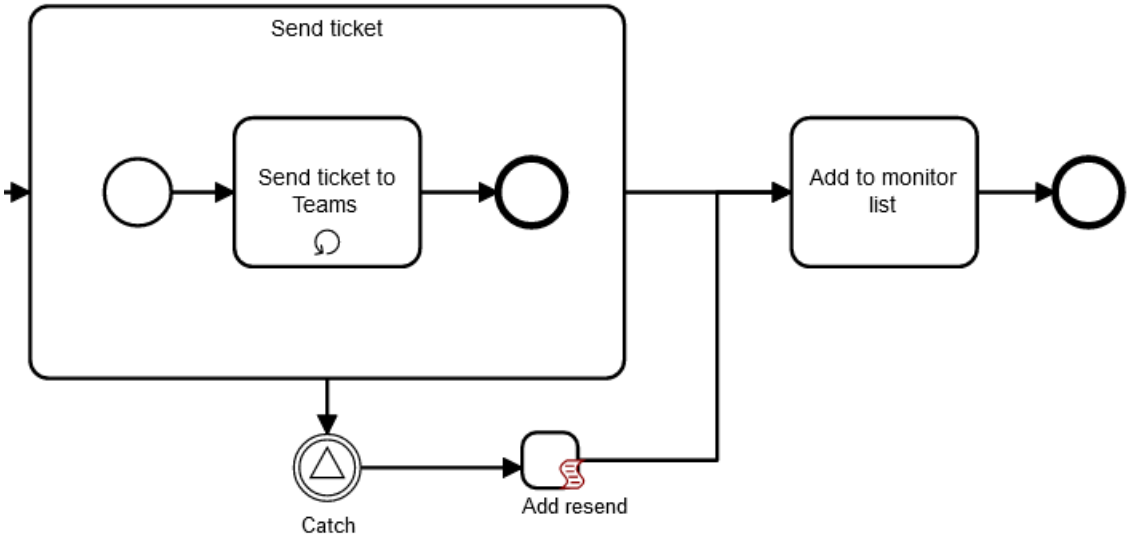
Kuva 27. Hubspot-tapahtumakulku eli “Workflow”, joka koostuu käynnistyselementistä, jossa käynnistyksen ehto määritetään. Kun tapahtumakulku on käynnistynyt ehdon mukaisesti, se suorittaa määritetyt toiminnot.

4.3 Virheenhallinta

Integraatiototeutuksien yleisiä kipupisteitä ovat yhteysvirheet, joita esiintyy erinäisistä syistä, kuten esimerkiksi kun rajapinta ei vastaa yhteydenottoon määrätyn ajan sisällä. Virhetilanteita aiheuttaa myös datan validointi, eli ohjelmistolle syötetty tieto ei vastaa muodoltaan tai sisällöltään ohjelmistoon määritettyä. Jotta ohjelmisto toimii luotettavasti, on näihin virheisiin varauduttava virheenhal-

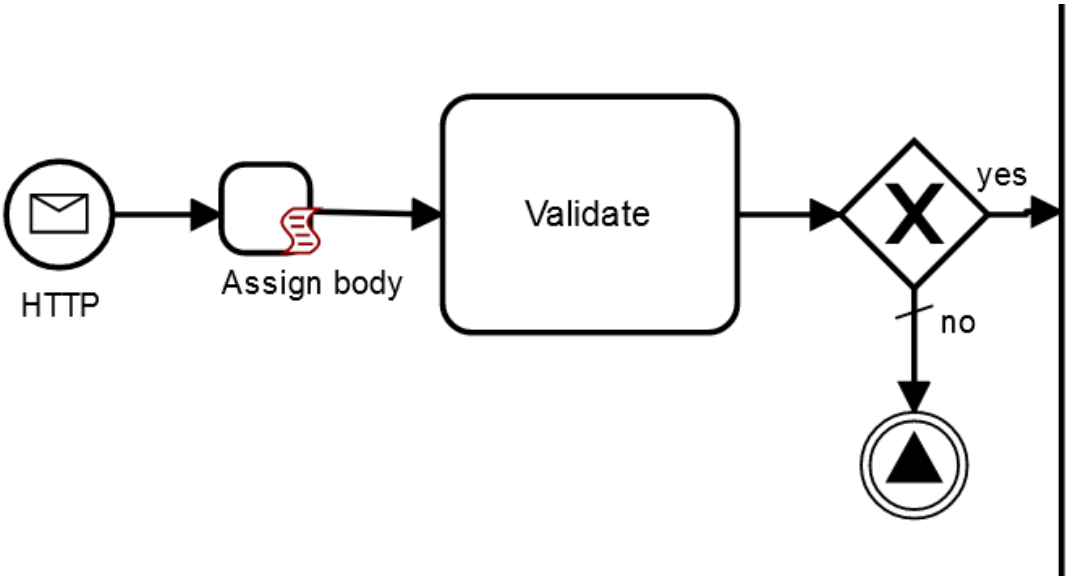
linnalla. Mahdolliset virhetilanteet ennakoidaan, jotta ohjelmisto tietää miten menetellä, kun virhe ilmenee prosessissa. Ohjelmisto määritetään myös käsittelemään virheet, joita ei osata ennakoida. Esimerkiksi tällöin ohjelmisto voi kerätä tiedon sattuneesta virheestä ja lähettää siitä raportin ylläpitäjälle. Virheet tuodaan ylläpitäjän tietoon reaaliaikaisesti, jolloin esiintyneisiin virheisiin voidaan puuttua ja tehdä tarvittavat toimet. Tämä on vakiintunut käytäntö suuremmissa integraatiototeutuksissa, jotka vaativat ylläpitoa. Se myös helpottaa integraation jatkokehitystä, sillä usein ilmeneviin virheisiin voidaan kehittää virheenhallinta suorittamaan tarvittavat toimet, kun jatkokehityksessä voidaan viitata saatuihin raportteihin.

Koska työssämme on käytössä useita rajapintoja, on meidän varauduttava yhteysvirheisiin ja niistä toipumiseen. Ensimmäinen kipukohta on vastaanotettuamme ajanvarauksen ja käsiteltyämme sen muotoon, jossa se lähetetään julkaistavaksi. Tässä on yhteysvirheen vaara, jonka vuoksi lisäämme HTTP-kutsuumme Frends-toiminnon tarjoaman uudelleenlähetyksen. Se tunnistaa rajapinnasta palautuneen virheen ja uudelleen lähettää sanoman. Jos tämäkään lähetys ei onnistu, heittää toiminto virheen. Olemme varautuneet tähän virheenheittoon lisäämällä “Catch”-toiminnon, joka tunnistaa virheen ja ohjaa prosessinkulun käsittelemään tapahtuneen virheen (kuva 28). Tässä haarassa määritämme “_resend”-liitteen lisättäväksi tallennettavan tiedoston nimeen, jolloin monitorointi tunnistaa sen ja lähettää tiedot uudelleen.



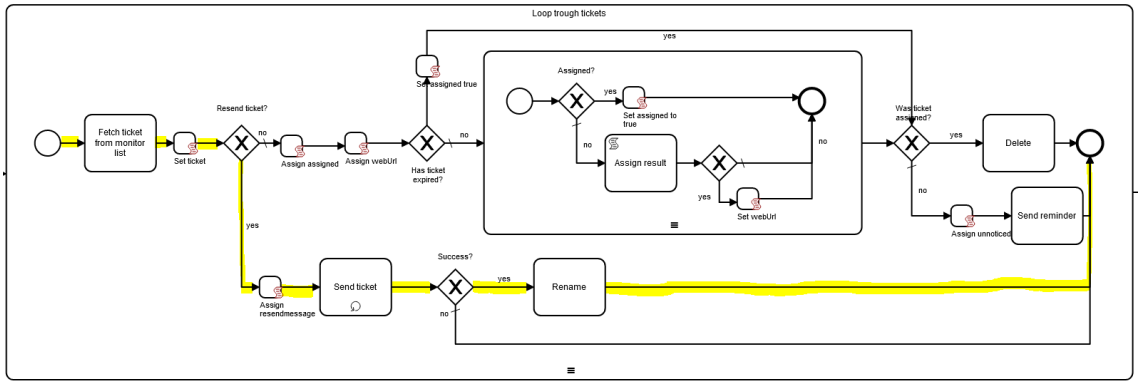
Kuva 28. Mikäli tiketin lähetys Power Automate -alustalle ei onnistu ensi yrittämällä, on toimintoon lisätty “Uudelleenlähetys”-toiminto. Se uudelleenlähettää sanoman, ja jos tämäkin epäonnistuu, toiminto palauttaa rajapinnan palauttaman virheen. “Catch”-toiminto tunnistaa virheen ja ohjaa prosessin suorittamaan toiminnon virheen korjaamiseksi.

Suorituksen alussa tarkistamme sisään tulevan sanoman (kuva 29). Siitä tarkastetaan tiedon olevan JSON-muotoista ja että se sisältää vaaditut tietueet. Tällä varmistetaan, että sanoma sisältää prosessin kannalta tarvitut tiedot ja eikä se sisällä ohjelmistolle haitallista dataa.



Kuva 29. Suorituksen alussa sisään tulleen sanoman formaatti ja tyyppi tarkistetaan. Suoritus lopetetaan virheeseen, mikäli sisään tullut sanoma ei vastaa muodoltaan määritettyä.

Monitoroinnissa lisäämme tiedoston haun jälkeen tarkistuksen, jolla tarkistetaan, löytyykö tiedoston nimestä “_resend”-viite (kuva 30). Se on lisätty tiketin luonnissa, mikäli tikettiä ei saatu lähetettyä, jolloin monitoroinnin tulee uudelleen lähettää tiketti Power Automate -prosessille. Lähetystä ei välitetä tiketin luonnille, sillä silloin meillä olisi kaksi tikettiä samasta varauksesta. Kun tiedot on lähetetty, tarkistetaan lähetyksen onnistuminen. Sen onnistuessa tiketin nimestä poistetaan uudelleenlähetyksen viite; muulloin kierros lopetetaan. Tällöin seuraava suoritus uudelleenlähettää tiketin tiedot, jos niiden lähetys ei tässä suorituksessa onnistunut.



Kuva 30. Tiedoston haun jälkeen tarkistetaan, löytyykö sen nimestä “_resend”-viite. Viitteen löytyessä, liittymä ohjataan kuvan mukaisesti uudelleen lähettämään tiketin tiedot Power Automate -prosessille. Tämän jälkeen tarkistetaan, onnistuuko lähetys. Sen onnistuessa tiedoston nimestä poistetaan viite. Muulloin muutoksia ei tehdä ja suoritus lopetetaan.

Vaikka tekemämme muutokset ovat pieniä, on niillä merkittävä vaikutus liittymien toimintaan tuotannossa. Uudelleenlähetys takaa sen, ettei kenenkään tarvitse valvoa liittymien toimintaa, vaan virhetilanteen sattuessa prosessit palautuvat niistä itsenäisesti. Sanomien validointi taas takaa sen, että prosessit pystyvät suoriutumaan sisään tulevan sanoman käsittelystä. Tämä estää myös likaisen tiedon syötön järjestelmään. Toteutuksessa ei ole kytkettynä tietokantaa, eikä siinä käsitellä arkaluontoisia tietoja, mutta virheidenraportointi on hyvä käytäntö lisätä kaikkiin rajapintatoteutuksiin.

4.4 Tulokset

Lopputuotteena työstä syntyi viiden prosessin integraatiototeutus. Näitä prosesseja ovat Hubspotin “Workflow”-toteutus, kaksi Frends-liittymä toteutusta, sekä kaksi Power Automate -prosessia. Frends toimii integraationa näiden kahden alustan välillä mahdollistaen tikettien luonnin ja seurannan sekä kommunikaation sovellusten välillä.

Työn toiminnallisuus ja vaatimuksien täyttäminen varmennettiin testitapauksien avulla. Nämä testitapaukset keskittyvät käyttäjärajapinnassa tapahtuviin toimiin, sillä se on suoraan yhteydessä käyttäjäkokemukseen. Testitapauksia olivat:

1. Asiakas tekee ajanvarauksen ja käyttäjä käsittelee varauksen.
2. Asiakas tekee ajanvarauksen ylihuomiselle ja käyttäjä ei käsittele varausta päivän aikana. Seuraavana päivänä käyttäjä käsittelee varauksen.
3. Asiakas tekee ajanvarauksen 24 tunnin päähän ja käyttäjä ei käsittele varausta määräaikaan mennessä.

Taulukko 1. Testitapauksien tulokset.

Testitapaus	**Tiketti lähetetty**	**Muistutus lähetetty kahdesti päivän aikana**	**Muistutus lähetetty kahden tunnin välein**	**Kuittausviesti lähetetty**
Testitapaus 1	x			x
Testitapaus 2	x	x	x	x
Testitapaus 3	x		x	

Työ selviytyi testitapauksista ja työn vaatimuksista. Tieto saadaan tuotua ajanvarauslomakkeelta kortin muodossa käyttäjän nähtäville Teams-kanavalle. Käsittelemättömistä varauksista lähetetään muistutukset ja käsitellyistä tiketeistä saadaan kortti päivitettyä, jotta muut käyttäjät näkevät sen kuitatuksi. Asiakas on hyväksynyt toteutuksen ensimmäisenä versiona ja se otetaan käyttöön. Teams-kortin lähetystapa on esitetty muutettavaksi “Webhook”-pohjaiseksi seuraavaan paranneltuun versioon, sillä työn edetessä osoittautui, että kortteja saadaan hallittua myös tällä menetelmällä, ja tällöin voidaan jättää Power Automate -järjestelmä pois toteutuksesta. Työ toi esille ajanvarauslomakkeeseen liittyviä puuttuvia toiminnallisuuksia kuten ilmaisten sähköpostien pois karsimisen, joita tulee tarkastella paremmin uuden version toteutuksessa. Toiseen versioon on myös esitetty lisättäväksi uusia toiminnallisuuksia, jotka mahdollistavat esimerkiksi Hubspot-kalenteriin vastuuhenkilön päivityksen.

5 Yhteenveto

Insinöörityön lopputuotteena on integraatio kahden palvelun välillä, ja tämä kokonaisuus koostuu viidestä liittymästä. Frends-integraation avulla saadaan Hubspot-ajanvarauslomakkeesta välitettyä varauksen tiedot Microsoft Teams -viestipalvelun kanavalle käyttämällä Power Automate -alustaa. Työ aloitettiin määrittämällä työssä käytettävät työvälineet, mikä osoittautui yhdeksi työn vaikeimmista osuuksista. Siinä missä Teams tarjosi paljon eri vaihtoehtoja toteutuksen tekemiseksi, Hubspotin tarjonta oli paljon suppeampi. Toteutuksen kehityksessä monet ongelmat johtuivat rajapintojen rajoituksista, jotka selvisivät vasta toteutusta tehtäessä. Tiketin lähetyksessä rajoitukset osoittautuivat haasteeksi kaikkien rajapintavaihtoehtojen kohdalla. Myös monitoroinnin toimintamalli riippui tiketin lähetyksen toiminnallisuudesta ja siitä, miten tiketti saatiin merkittyä käsitellyksi, jolloin monitoroinnin rakentaminen aloitettiin alusta, kun tiketin lähetyksen toimintamalli muuttui. Kun työ edistyi, myös asiakkaan tarpeet muuttuivat aloituksesta selkeämmiksi. Tämä kosketti eniten Frends-liittymien toimintalogiikkaa esimerkiksi muistutusten suhteen. Alusta mukautui vaatimuksiin ongelmitta, eikä asiakkaalle tarvinnut kertoa, etteikö toiminnallisuutta pystyttäisi toteuttamaan, mikäli se on Frends-alustasta kiinni.

Työ koostui ohjelmistoarkkitehtinä työn määrittelystä ja integraationasiantuntijana toimimisesta. Nämä työtehtävät mahdollistivat olemassa olevan osaamisen hyödyntämisen projektissa sekä itsenäisen tutkimuksen harjoittamisen. Projekti antoi myös mahdollisuuden tutustua tarkemmin Frends-, Hubspot-, Teams-, Power Automate -alustojen ja Graph API -rajapinnan toimintaan ja näin tarjosi paremman ymmärryksen alustojen toimintamenetelmistä. Projekti oli opettavainen kokemus niin asiakaskohtaamisista kuin projektin hallinnollisista näkökulmista. Projektissa kävimme läpi ohjelmistoille tyypillisen kehityskaaren, kuinka ideasta tehdään suunnitelma ja miten tämä suunnitelma rakennetaan valmiiksi tuotteeksi.

Insinöörityöstä syntynyt lopputuote helpottaa ja nopeuttaa ajanvarauksen työn kulkua. Se vähentää käyttäjien työkuormaa sekä varausten riskiä jäädä käsittelemättä ja poistaa käyttäjien välillä tapahtuvat kyselyt käsittelyn toteutumisesta. Työ hyväksyttiin tilaajan toimesta ja otetaan käyttöön tuotteen ensimmäisenä versiona. Kehitystyötä jatketaan uuden version parissa nykyistä toteutusta optimoiden ja uusia toiminnallisuuksia lisäten. Tuotetta tullaan käyttämään muiden automatisoitavien prosessien pohjana.

Lähteet

1 Yritys HiQ Finland. Verkkoaineisto. https://hiq.fi/yritys/. Luettu 22.09.2022.

2 HiQ Finland. Verkkoaineisto. https://fi.wikipedia.org/wiki/HiQ_Finland. Luettu 22.09.2022.

3 HiQ Asiantuntijapalvelut. Verkkoaineisto. https://hiq.fi/palvelut/yllapida/asiantuntijapalvelut/. Luettu 22.09.2022.

4 System integration. Verkkoaineisto. https://en.wikipedia.org/wiki/System_integration. Luettu 26.09.2022.

5 Helling, Brett. 2022. A Beginner's Guide To Software Integrations. Verkkoaineisto. https://bloggingtips.com/software-integration/. Luettu 26.09.2022.

6 Mitä integraatio, rajapinta ja api tarkoittavat? 2019. Verkkoaineisto. https://valjas.fi/opi/blogi/mita-integraatio-rajapinta-ja-api-tarkoittavat/. Luettu 27.09.2022.

7 Jones, Thomas. 2021. What is an API Integration? (A guide for non-technical people). Verkkoaineisto. https://www.gend.co/blog/what-is-api-integration-a-guide-for-non-technical-people. Luettu 26.09.2022.

8 ERP-integraatiot. Verkkoaineisto. https://www.solita.fi/erp-integraatiot/. Luettu 28.09.2022.

9 Paloniitty, Ari. 2019. 12 Tärkeintä CRM-integraatiota. Verkkoaineisto. https://www.blinkhelsinki.fi/blogi/12-tarkeinta-crm-integraatiota. Luettu 28.09.2022.

10 Tiedosto vai rajapinta? Verkkoaineisto. https://hri.fi/fi/ohjeet/datan-avaajalle/tiedosto-vai-rajapinta/. Luettu 27.09.2022.

11 Mikä on ohjelmistointegraatio ja mihin sitä tarvitaan? Verkkoaineisto. https://www.itewiki.fi/p/mika-on-ohjelmistointegraatio-ja-mihin-sita-tarvitaan . Luettu 27.09.2022.

12 Galkin, Ossi. 2022. What is Frends? Verkkoaineisto. https://docs.frends.com/en/articles/2188944-what-is-frends. Luettu 29.09.2022.

13 Liiketoimintamallien malli ja merkinnät. Verkkoaineisto. https://fi.frwiki.wiki/wiki/Business_process_model_and_notation. Luettu 29.09.2022.

14 Devops. Verkkoaineisto. https://fi.wikipedia.org/wiki/Devops. Luettu 29.09.2022.

15 Galkin, Ossi. 2022. Verkkoaineisto. https://docs.frends.com/en/articles/2189208-how-to-use-the-frends-ui. Luettu 29.09.2022.

16 Aliohjelma. Verkkoaineisto. https://fi.wikipedia.org/wiki/Aliohjelma. Luettu 29.09.2022.

17 Galkin, Ossi. 2022. Process. Verkkoaineisto. https://docs.frends.com/en/articles/2236816-process. Luettu 29.09.2022.

18 Galkin, Ossi. 2022. Quick start for those who don’t read documentation. Verkkoaineisto. https://docs.frends.com/en/articles/2324585-quick-start-for-those-who-don-t-read-documentation. Luettu 29.09.2022.

19 Galkin, Ossi. 2022. Tasks. Verkkoaineisto. https://docs.frends.com/en/articles/2211481-tasks. Luettu 29.09.2022.

20 Frends EiPaas. Verkkoaineisto. https://github.com/FrendsPlatform.

21 Galkin, Ossi. 2022. Custom Tasks. Verkkoaineisto. https://docs.frends.com/en/articles/2206746-custom-tasks. Luettu 29.09.2022.

22 Galkin, Ossi. 2022. Agent Architecture. Verkkoaineisto. https://docs.frends.com/en/articles/2824943-agent-architecture. Luettu 05.10.2022.

23 Galkin, Ossi. 2022. How Frends create and run processes. Verkkoaineisto. https://docs.frends.com/en/articles/2189265-how-frends-create-and-run-processes. Luettu 01.10.2022.

24 Frends-integraatioalusta. Verkkoaineisto. https://www.vaimo.com/fi/osaaminen/frends-integraatioalusta/. Luettu 29.09.2022.

25 Monsanto, Claudia Martinez. 2020. Software Integrations: A Beginner's guide. Verkkoaineisto. https://blog.hubspot.com/website/software-integration. Luettu 26.09.2022.

26 Mikä on Hubspot ja miten se tuo sinulle menestystä? Verkkoaineisto. https://pitkospuu.fi/mika-on-hubspot/. Luettu 26.09.2022.

27 Hokkanen, Samuli. 2017. Mitä on Inbound-markkinointi ja mitä hyötyä siitä on? Verkkoaineisto. https://www.crasman.fi/blogi/mit%C3%A4-on-inbound-markkinointi. Luettu 04.10.2022.

28 Get Started with Power Automate. 2022. Verkkoaineisto. https://learn.microsoft.com/en-us/power-automate/getting-started. Luettu 05.10.2022.

29 Overview of Microsoft Graph. 2022. Verkkoaineisto. https://learn.microsoft.com/en-us/graph/overview. Luettu 06.10.2022.

30 Free Meeting Scheduler. Verkkoaineisto. https://www.hubspot.com/products/sales/schedule-meeting. Luettu 28.10.2022.

31 Webhooks and connectors. 2022. Verkkoaineisto. https://learn.microsoft.com/en-us/microsoftteams/platform/webhooks-and-connectors/what-are-webhooks-and-connectors. Luettu 26.09.2022.

32 Galkin, Ossi. 2022. Shared State Task. Verkkoaineisto. https://docs.frends.com/en/articles/5270990-shared-state-task. Luettu 17.10.2022.

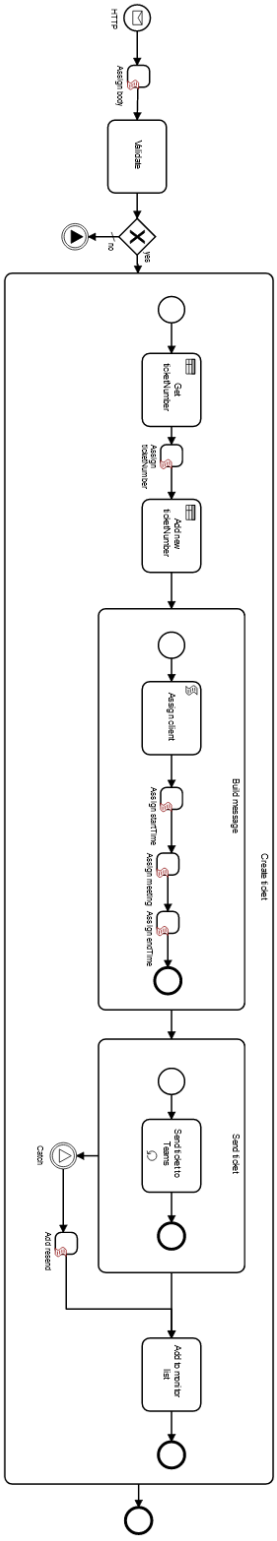
33 Authentication and authorization basics. 2022. Verkkoaineisto. https://learn.microsoft.com/en-us/graph/auth/auth-concepts. Luettu 27.09.2022.

34 Configurable token lifetimes in the Microsoft identity platform. 2022. Verkkoaineisto. https://learn.microsoft.com/en-us/azure/active-directory/develop/active-directory-configurable-token-lifetimes. Luettu 27.09.2022.

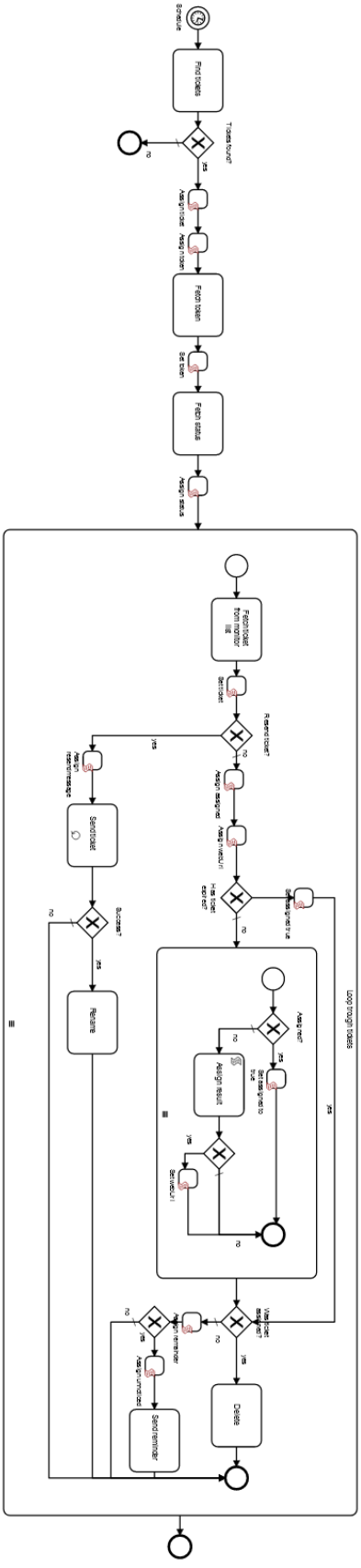
35 Create Workflows. 2022. Verkkoaineisto. https://knowledge.hubspot.com/workflows/create-workflows. Luettu 5.10.2022.

36 Private Apps. 2022. Verkkoaineisto. https://developers.hubspot.com/docs/api/private-apps. Luettu 17.10.2022.

Frends-liittymä kaaviot



Kuva 1. Tiketin luonnin liittymäkaavio.



Kuva 2. Monitoroinnin liittymäkaavio.