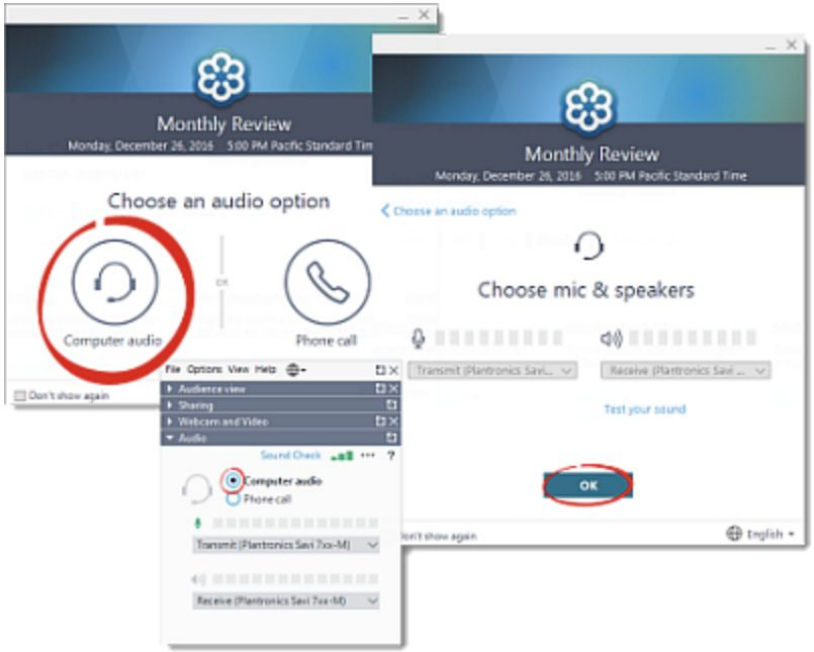
Tampuuri API -käytön perusteet ja integraatioiden suunnittelu

29.10.2021 klo 9-10

Timo Varis, Product Manager
Antti Hartikainen, Sales Engineer

VISMA

Tervetuloa Tampuurin webinaariin! Ongelmia äänen kanssa? Tarkista asetukset



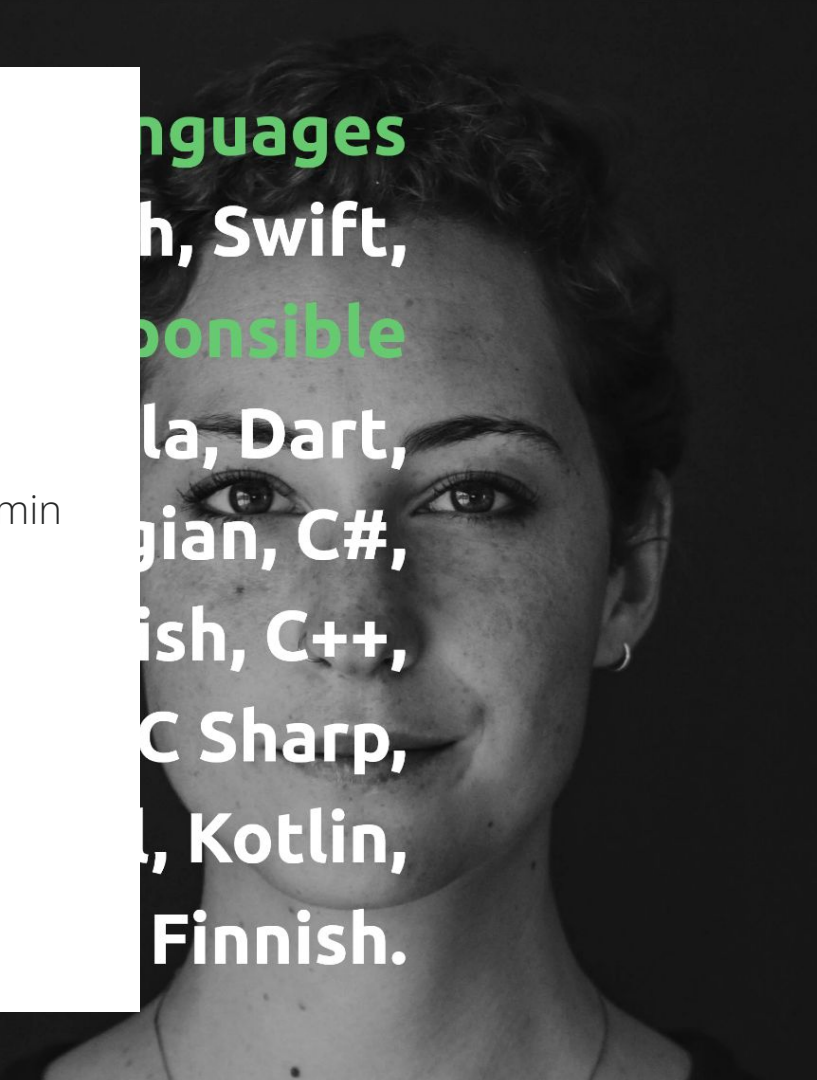
- Kun osallistut ensimmäisen kerran webinaariin, sinulle aukeaa kysely ääniasetuksista (voi aueta muiden ikkunoiden taakse)
- Valitse, millä laitteella haluat kuunnella webinaaria
  - Tietokone
  - Puhelin
- Valitse, kuunteletko webinaaria laitteen kaiuttimen vai kuulokkeiden kautta
- Muista painaa lopuksi ok

VISMA

Alkuun muutama käytännön asia

- Webinaarin kesto on **1 tunti**
- Webinaari tallennetaan. Saatte tallenteen myöhemmin sähköpostitse.
- Kysymyksiä voi esittää **Questions**-kohdasta koko webinaarin ajan. Kysymyksiin vastataan esityksen jälkeen.

VISMA



Asiantuntijat

Antti Hartikainen

Sales Engineer

Visma Tampuuri

Timo Varis

Product Manager

Visma Tampuuri

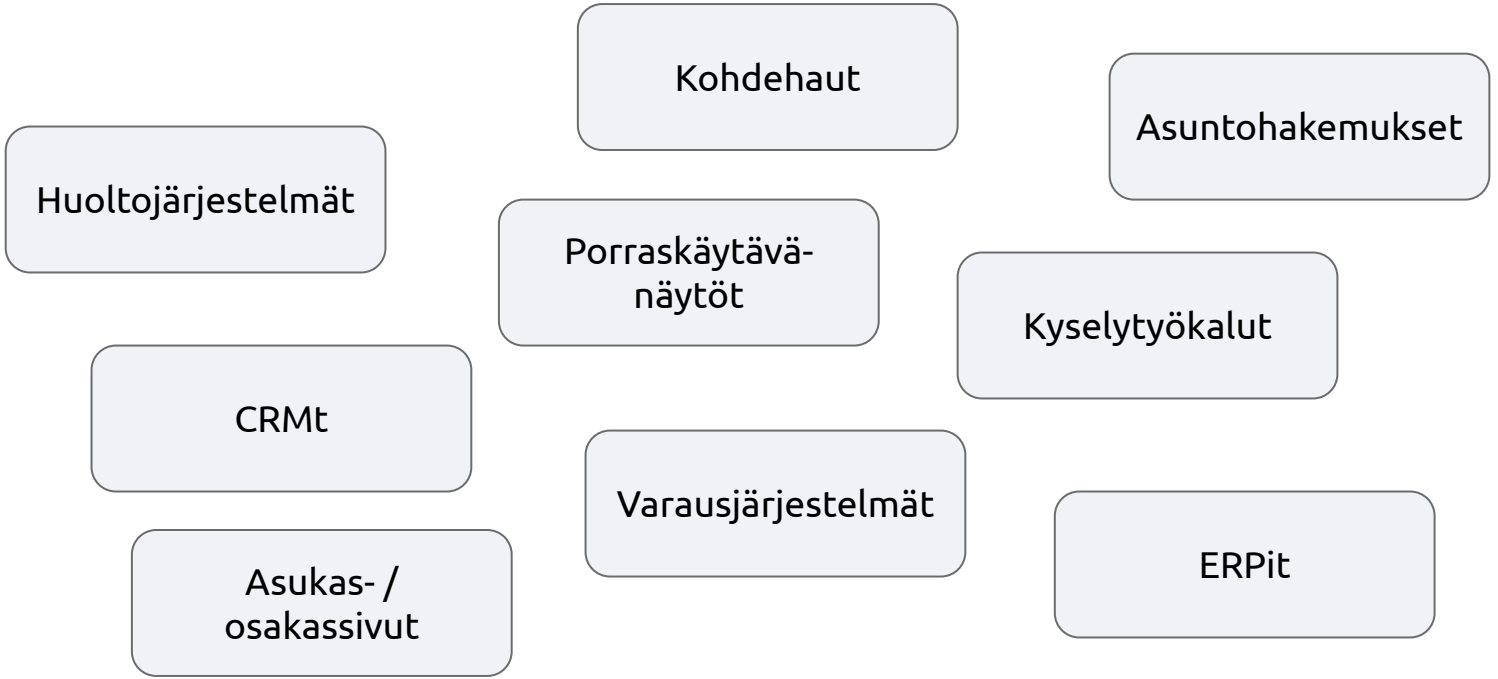
VISMA

Mikä on Tampuuri API?

- Visma Tampuurin rajapintakokonaisuus, joka on
  - REST -tyylinen ohjelmointirajapinta
  - Käyttää tiedonsiirtoon HTTP -protokollaa
  - Tiedon formaattina JSON -notaatiota
- Mahdollistaa Tampuurin tietojen viemisen ulkopuoliseen järjestelmään
- Mahdollistaa Tampuurin toimintojen ajamisen osana monen järjestelmän läpi kulkevia prosesseja → esimerkkinä vikailmoitusprosessi
- Tampuuri API on käytössä kymmenissä integraatioissa sekä Tampuurilla että Asiakkailla ja kumppaneilla
  - Mm. Tampuurin omat Asukassivut ja Kenttätyökalut on rakennettu täysin Tampuuri APIn päälle.

VISMA

Millaisia integraatioita APIlla on tehty



Tampuuri API ja Webhookit

- Tampuuri APIin on saatavana lisäpalveluna myös tuki tapahtumapohjaisille toteutuksille, eli ns. Webhookeille
- Webhookit mahdollistavat Tampuuri APIin integroituville sovelluksille kuuntelijat, jotka saavat tiedon Tampuurin sisällä tapahtuvista asioista
    - Esimerkiksi kohteen tietojen päivittymisestä voidaan nostaa tapahtuma ulkopuolisen sovelluksen tiedoksi, jolloin tämä voi tehdä tietojen virkistyksen Tampuuri APIa käyttäen

Miten Tampuuri API otetaan käyttöön?

- Tampuuri-asiakkaan polku
  - Tampuuri APIn käyttöönotto lähtee liikkeelle sopimuksesta
  - Saatavilla on kolme erilaista pakettiratkaisua, joiden hinnoittelu perustuu käyttötarpeeseen
  - Sopimuksen solmimisen jälkeen nimetään tekninen yhteyshenkilö ja luovutetaan käyttöön tarvittavat avaimet → Kehitys voi alkaa

VISMA

Miten Tampuuri API otetaan käyttöön?

- Mitä voin tehdä, jos en ole asiakas, eikä minulla ole tiedossa pilottiasiakkuutta? Vastaus on Tampuurin kumppanuusohjelma.

Yhteistyöyritys

Tuotteistetut rajapinnat Tampuurin asiakkaille.

Kumppani käyttää Tampuuri API-rajapintaa ja sen tuomaa dataa liiketoimintansa edistämiseksi.

Yhteistyökumppani

Yhteistyökumppanin tarjoama tuote/palvelu täydentää Tampuurin tuoteportfoliota.

Yhteistyökumppani vastaa itse sovellusten tuotannosta/myynnistä ja niitä myydään yhteistyökumppanin omalla brändillä.

Sovelluskumppani

Yhteistyökumppani tuottaa sovelluksen, jota myydään osana Tampuuria.

Sovellus myydään Tampuuri-brändin nimissä, sen toimesta ja hinnoittelemana.

VISMA

Miten Tampuuri API otetaan käyttöön?

- Sopimuksen solmimisen ja Tampuuri APIn käyttöönoton jälkeen asiakkaalle toimitetaan tarvittavat header-tiedot APIn käyttämiseen
  - **X-TampuuriAvain: XXXXXXXX-YYYY-ZZZZ**
  - **X-AsiakasNimi: Visma Tampuuri Oy**
  - **X-Ohjelmistoversio: Integraatio X**
  - **X-TiliNimi: Visma Tampuuri**
  - **X-Kayttajatunnus: kayttajatunnus**
  - **X-Salasana: **********
- APIin tehdyt kutsut autentikoidaan näiden headerien avulla ja ne pitää toimittaa jokaisen http-kutsun yhteydessä

VISMA

Tampuuri APIn käyttäminen

- Tampuuri APIn dokumentaatio löytyy osoitteesta https://api.tampuuri.fi/v10/api/Help
  - Sisältää listan saatavilla olevista operaatioista, sekä niiden mallit
  - Vaatii jonkin verran Tampuuri-termistön tuntemista
- Kun autentikointiin vaadittavat headerit on saatu, on hyvä kokeilla että saadaan yhteys toimimaan
  - Hyväksi havaittuja välineitä testaamiseen:
    - JMeter
    - Insomnia
    - Postman
    - VSCode + REST Client (by Huancao Mao)

API-kutsun rakenne

- API-kutsu koostuu neljästä osasta
  - URI → https://api.tampuuri.fi/v10/api/sopimukset/{sopimusid}
  - Verbi → GET, POST, PUT, PATCH, DELETE
  - Kutsun header-tiedot
  - Kutsun body-tiedot (paitsi GET ja DELETE -verbeissä)
- URI koostuu kiinteästä ja merkitsevästä osasta
  - Kiinteä osuus on ympäristökohteinen
    - Tuotantoympäristöt → https://api.tampuuri.fi/v10/api/
    - Testiympäristöt → https://qa-api.tampuuri.fi/v10/api/
  - Merkitsevä osuus lisätään kiinteän perään ja se koostuu halutun kutsun tiedoista ja mahdollisista URI-parametreista
    - Esim. GET https://api.tampuuri.fi/v10/api/**sopimukset/442766**

VISMA

API-kutsun rakenne

- HTTP-verbi kertoo yleisesti ottaen kutsun tarkoituksesta
  - GET = tiedon noutamiseen tarkoitettu kutsu
  - POST, PUT, PATCH = tiedon lisäämiseen / päivittämiseen tarkoitettu kutsu
  - DELETE = tiedon poistamiseen tarkoitettu kutsu
- Poikkeuksina sääntöön ovat tiedon noutamiseen tarkoitetut POST-kutsut, joiden URI päättyy termiin “selaa”
  - Esim. **POST ilmoitukset/selaa**
  - Näissä on päädytty POST-verbiin, koska näin parametrit voidaan antaa URI:n sijaan kutsun body-elementissä

API-kutsun rakenne

- Kutsun header-tiedot autentikoivat kutsun ja ohjaavat sen oikeaan Tampuuri-tiliin → Samalla avaimella kumppanin on mahdollista päästä moneen Tampuuri-tiliin, mutta jokainen tili tarvitsee oman käyttäjätunnuksen ja salasanan.
- Kutsun mahdollinen body-elementti sisältää kutsun tarvitsemat parametrit JSON-muodossa.

VISMA

API-kutsun palauttama tieto

- APIa vasten lähetetystä kutsusta seuraa palaute (response)
- Palaute koostuu paluukoodista ja mahdollisesta bodystä
- Paluukoodi kertoo karkealla tasolla kutsun onnistumisesta tai epäonnistumisesta, **nyrkkisääntönä siis:**
  - **200-sarjan** koodit tarkoittavat kutsun onnistuneen
    - Esim. 200 “OK” tarkoittaa, että kutsu onnistui odotetusti
  - **400-sarjan** koodit tarkoittavat, että kutsuva pää on tehnyt virheen
    - Esim. 403 “unauthorized” tarkoittaa tavallisesti, että salasana tai Tampuuri-avain ovat väärin
  - **500-sarjan** koodit tarkoittavat, että kutsuttavan palvelun päässä tapahtui virhe

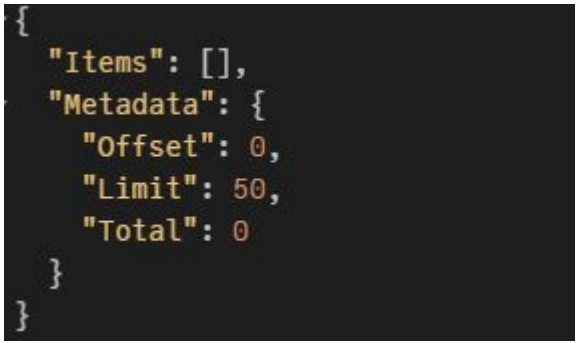
VISMA

API-kutsun palauttama tieto

- Response body sisältää APIn palauttaman tiedon JSON-muodossa
- Kunkin rajapintakutsun palautteen malli on kuvattu APIn dokumentaatiossa
- **HUOM! Tampuuri APIn palauttama data ei huomioi käyttöliittymän kenttäkohtaisia käyttöoikeuksia, vaan käyttää aina dokumentaatiossa kerrottua formaattia. API-käyttäjällä on siis aina täysi näkyvyys palautettaviin resursseihin.**

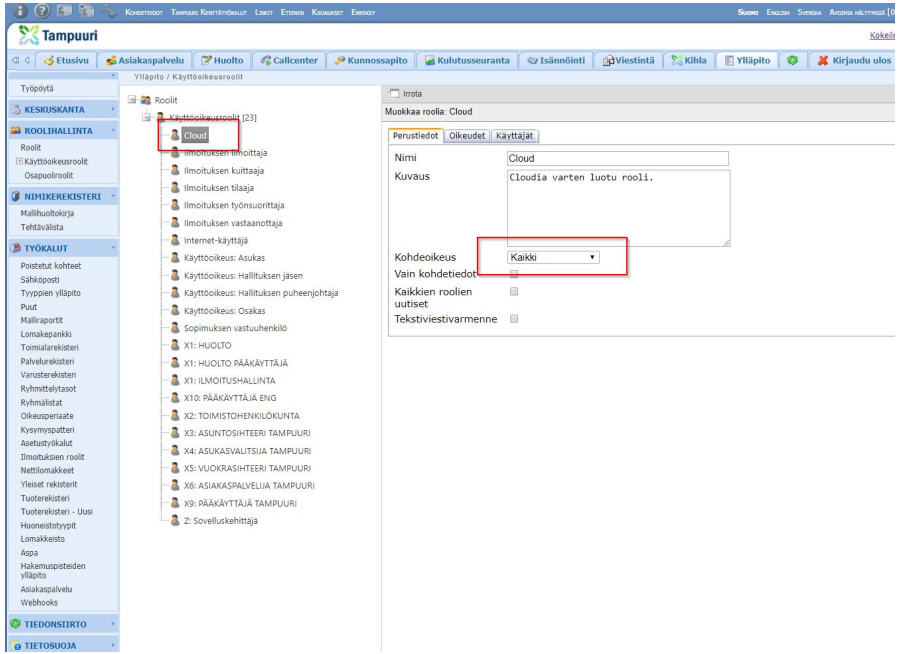
API-kutsun palautteen “sivutus”

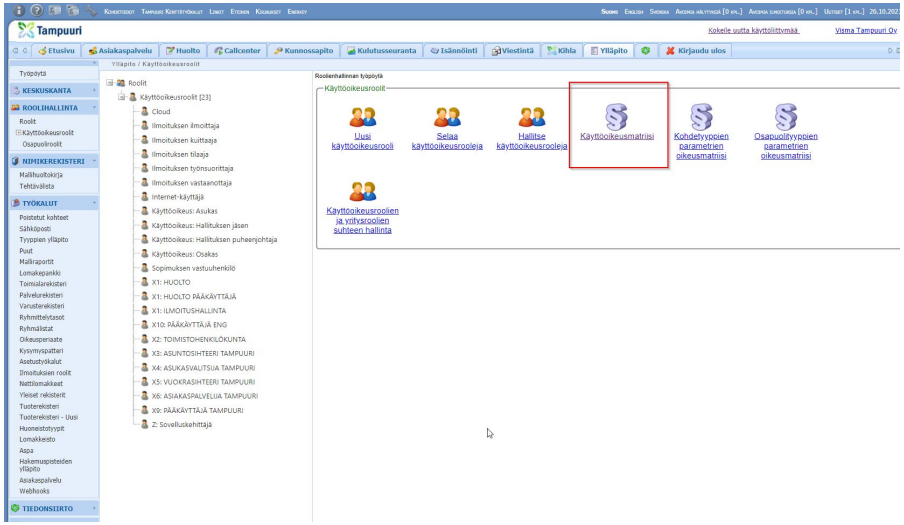
- Joissakin end-pointeissa on potentiaalisesti hyvin suuri response-body palautettujen rivien vuoksi (esim. useissa kohderekisteriin kohdistuvissa kutsuissa)
- Tällaisissa tapauksissa API-kutsu ja response tukevat sivutusta
  - Sivutettujen kutsujen maksimikoko on 250 palautettua objektia (haluttu määrä voidaan antaa kutsun limit-kentässä)
  - Responsen bodyssä annettava metadata-kokoelma kertoo kutsun palauttamien objektien kokonaismäärän → total-kenttä
  - Sivutusta ohjataan antamalla kutsun mukana offset-kenttään yli hypättävien objektien määrä
  - Esim. Jos kutsu palauttaa 275 objektia, voidaan koko massa hakea kahdella kutsulla → Ensimmäisen kutsun limit=250 ja offset =0, toisen kutsun limit=250 ja offset=250 (koska ensimmäiset 250 haettiin ensimmäisellä kutsulla)



VISMA

Vinkki: Oma käyttöoikeusrooli APIlle



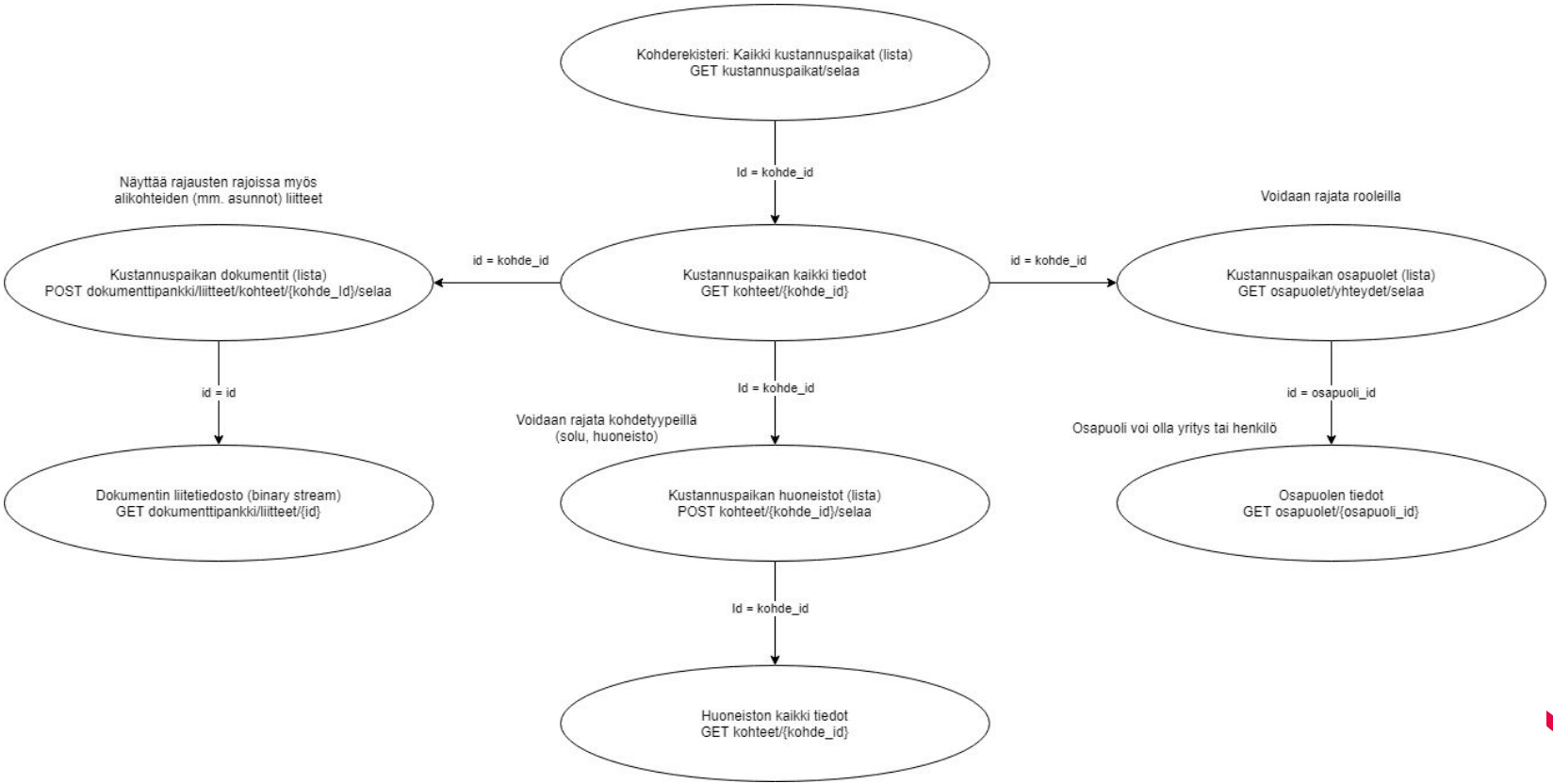


Peruskäsitteistö ja periaatteet

- APIssa eri resursseihin viitataan niiden sisäisellä IDlla, esimerkiksi asunnon, henkilön tai sopimuksen IDlla.
  - Nämä ovat yleensä muodoltaan juoksevia numeroita
    - Esim. GET sopimukset/**{sopimus-id}** → GET sopimukset/**32892**
  - APIa kutsuvan järjestelmän pitää tuntea nämä ID:t, tai niiden selvittämiseksi on rakennettava suunniteltavaan integraatioon jokin keino
- Tampuuri APIn käytössä lähdetään yleensä liikkeelle jommasta kummasta perusrekisteristä
  - Kohderekisteri → kustannuspaikka, rakennus, asunto jne.
  - Osapuolirekisteri → henkilö tai yritys

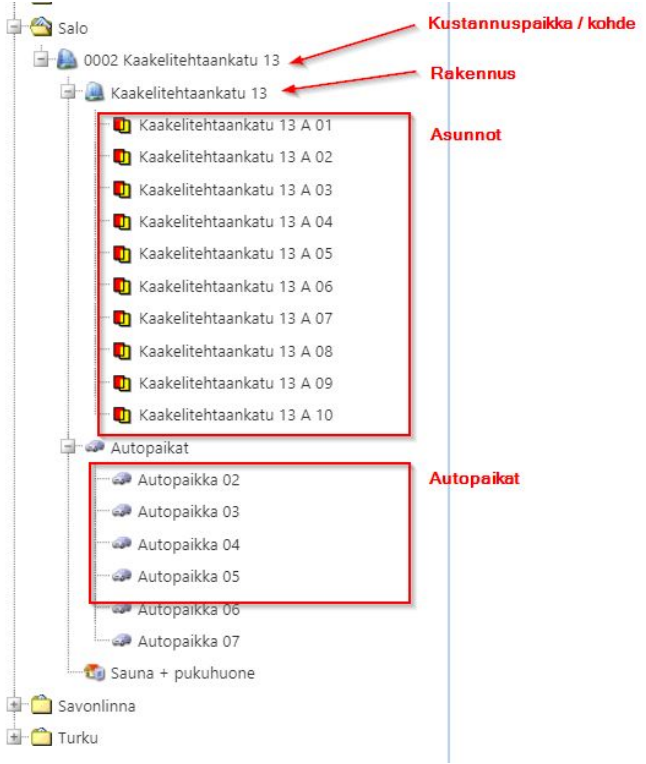
VISMA

Esimerkki: Kohderekisteristä liikkeelle

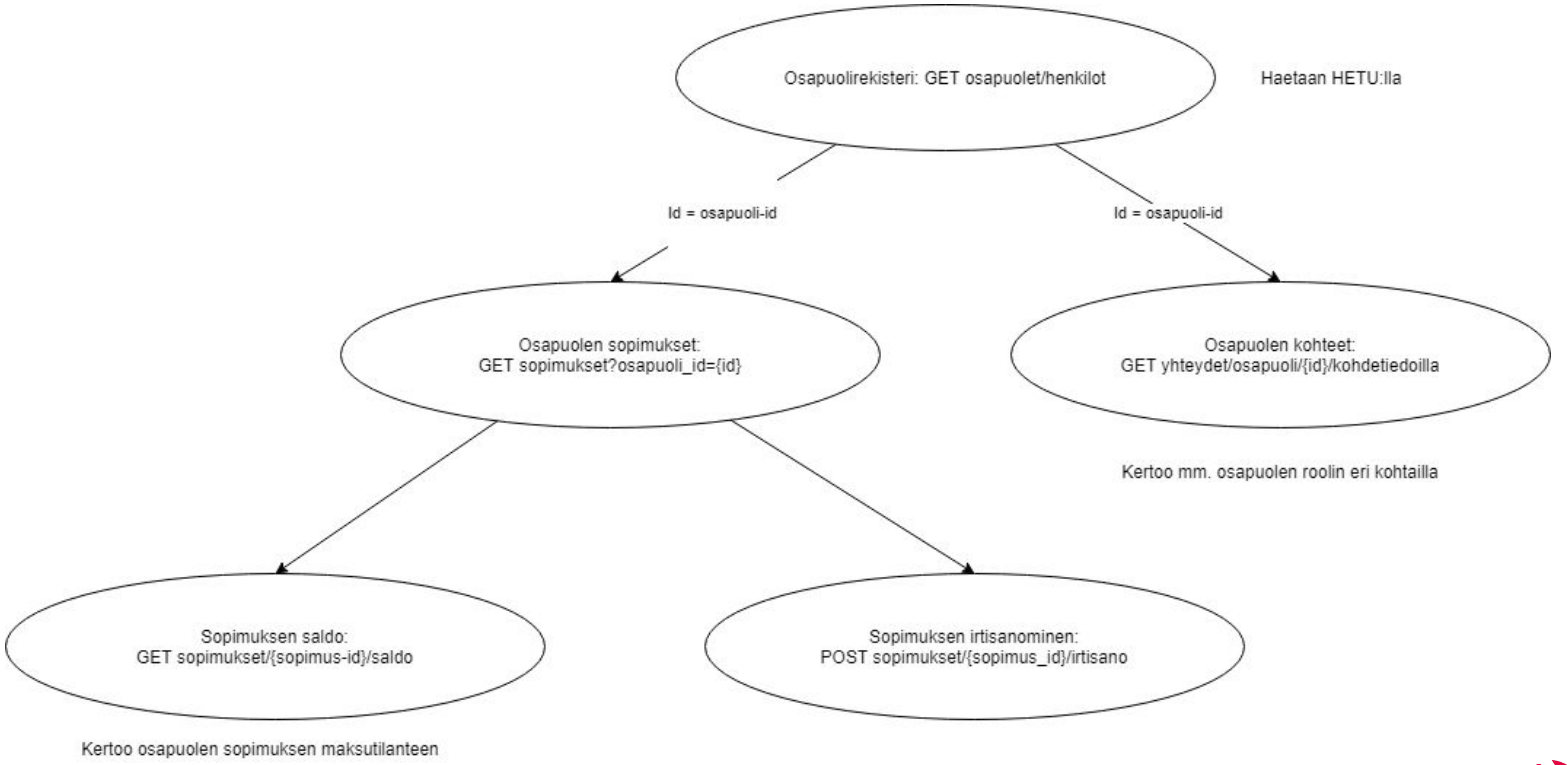


Esimerkki: Kohderekisteristä liikkeelle

- Huomioita:
  - Yleensä asukkaat ja osakkaat kytketään kohdepuun asuntotasolle
  - kun taas huoltoyhtiöt, hallitukset, isännöitsijät yms. kytketään kustannuspaikkatasolle
  - Tampuurin kohderakenne on puumainen ja saattaa vaihdella hieman asiakkaittain (ts. kaikilla asiakkailla ei ole kaikkia tasoja tai kenttiä käytössään)



Esimerkki: Osapuolesta liikkeelle

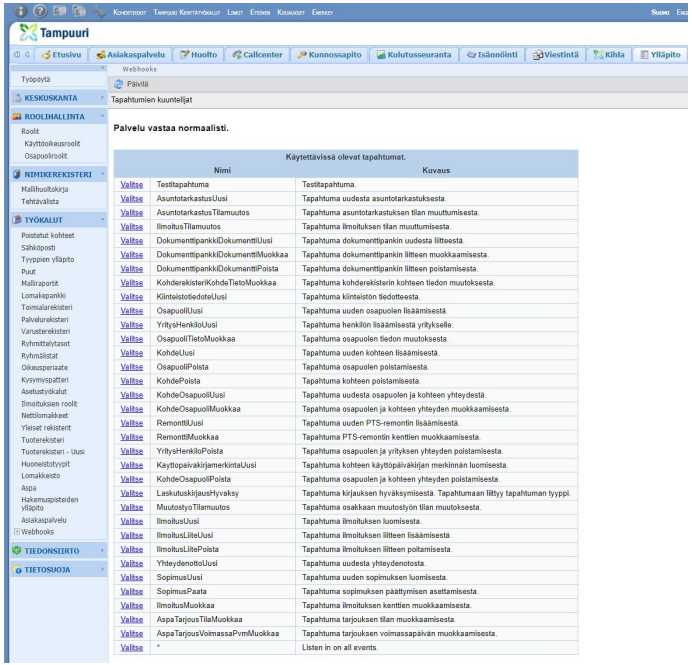


Mitä muuta Tampuuri APIsta löytyy

- Kohde ja osapuolitietojen lisäksi Tampuuri APIsta löytyy satoja endpointteja näihin liittyvien tietojen hakemiseen, mm.
  - Vikailmoitukset, työpyynnöt, huoltokirjan tehtävät
  - Vuokrasopimukset
  - Hakemukset, tarjoukset
  - Asuntotarkastukset
  - Avaimet
  - Dokumenttikansiot ja liitetiedostot
  - Muutostyöilmoitukset

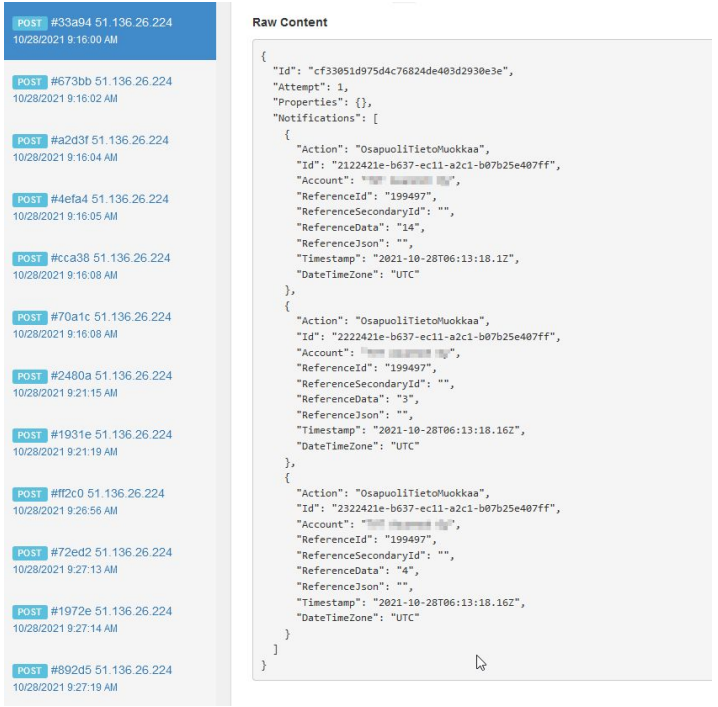
Tampuuri API:n tukemat tapahtumat

- Tampuuri API tukee myös webhookkeja mm. seuraavien tapahtumien osalta
  - Uusi asuntotarkastus / tarkastuksen muutos
  - Uusi vikailmoitus / ilmoituksen muutos
  - Uusi dokumentti / dokumentin muutos
  - Uusi osapuoli / osapuolen muutos / osapuolen poisto
  - Uusi kohde / kohteen muutos / kohteen poisto
  - Uusi remontti / remontin muutos
  - Uusi sopimus / sopimuksen muutos
  - Uusi osapuolen ja kohteen välinen kytkös
- Asiakas voi lisätä kuuntelijan Tapahtumalle itse Tampuurin ylläpito-välilehdeltä
- Tapahtumia voi testata esim. sivuston **webhook.site** kautta
  - Sivustolla voi luoda itse kuuntelijan ja tarkastella siihen tulevia tapahtuma-liipaisuja



Tapahtumailmoitusten formaatti

- Tapahtumista julkaistavat ilmoitukset sisältävät seuraavat kenttätiedot
  - Action = Tapahtumatyyppi
  - Account = Tampuuri-tili, jossa tapahtuma on sattunut
  - ReferenceId = Resurssin ID
  - ReferenceSecondaryId Toissijaisen resurssin ID
  - ReferenceData = Tapahtumaan liittyvä lisätieto
  - ReferenceJson = Lisätieto json-muodossa
  - Timestamp = Aikaleima
- Tapahtuma-väylä niputtaa kuuntelijan pyytämät ilmoitukset toimitushetken mukaan
  - Notifications -kokoelma
  - Nippua koetetaan toimittaa max 5 kertaa (Attempts-property)



Esimerkki integraatiosta: Huoltotehtävien vienti Tampuurista

Käyttötarve

- Asiakkaan yhteistyökumppani (huoltoyhtiö) haluaa tehostaa toimintaansa hallitsemalla kaikki keikat omassa järjestelmässään.
- Asiakas käyttää Tampuurin ilmoitushallintaa vikailmoitusten pääasiallisena talletuspaikkana ja lähteenä maksujen hyväksynnälle.
- Asiakkaalla on käytössään Tampuurin asukassivut, joten on tärkeää, että myös asukas tietää missä tilassa hänen tekemänsä vikailmoitukset ovat.

VISMA

Esimerkki integraatiosta: Huoltotehtävien vienti Tampuurista

Integraation prosessi Tampuuri API:n avulla

- Asukas / Tampuuri-käyttäjä kirjaa vikailmoituksen
- Huoltoyhtiön integraatio pollaa Tampuuria 15 minuutin välein
  - Kutsu POST ilmoitukset/selaa
  - Parametreina → työnsuorittaja-id = huoltoyhtiön osapuoli-id, ilmoituksen tila = avoin
- Api palauttaa määritellylle työn suorittajalle merkityt avoimet keikat
- Huoltoyhtiön integraatio tallettaa keikat omaan huoltojärjestelmään

VISMA

Esimerkki integraatiosta: Huoltotehtävien vienti Tampuurista

Integraation prosessi Tampuuri API:n avulla

- Huoltoyhtiö aloittaa huoltotoimenpiteet
  - Kutsu POST ilmoitukset/{ilmoitus_id}/vastaanota
  - **Huom: Tampuurissa vastaanotettu-tila indikoi, että tehtävä on otettu suoritettavaksi** - toiset toimijat merkitsevät kaikki noudetut keikat heti vastaanotetuiksi, toiset taas vasta kun huoltomies ottaa tehtävän työn alle.

VISMA

Esimerkki integraatiosta: Huoltotehtävien vienti Tampuurista

- Huoltoyhtiö suorittaa tehtävän
  - Mikäli tehtävä vaatii väli kommentointia tai siirtämistä väliaikaisesti odottavaan tilaan (jolloin viiveestä on hyvä ilmoittaa Tampuurin kautta asukkaalle) integraatio tekee tämän APIn kautta
    - Kutsu POST ilmoitukset/{ilmoitus_id}/odotukseen → tila = odottaa
  - Mikäli tehtävä poikii uusia työtilauksia / lisäilmoituksia, integraatio tekee uudet ilmoitukset APIn kautta
    - Kutsu POST ilmoitukset/base64 → uusi ilmoitus, tila = avoin
- Kun tehtävä on valmis, huoltoyhtiö kuittaa sen suoritetuksi APIn kautta
  - Kutsu POST ilmoitukset/{ilmoitus_id}/kuittaatehdyksi

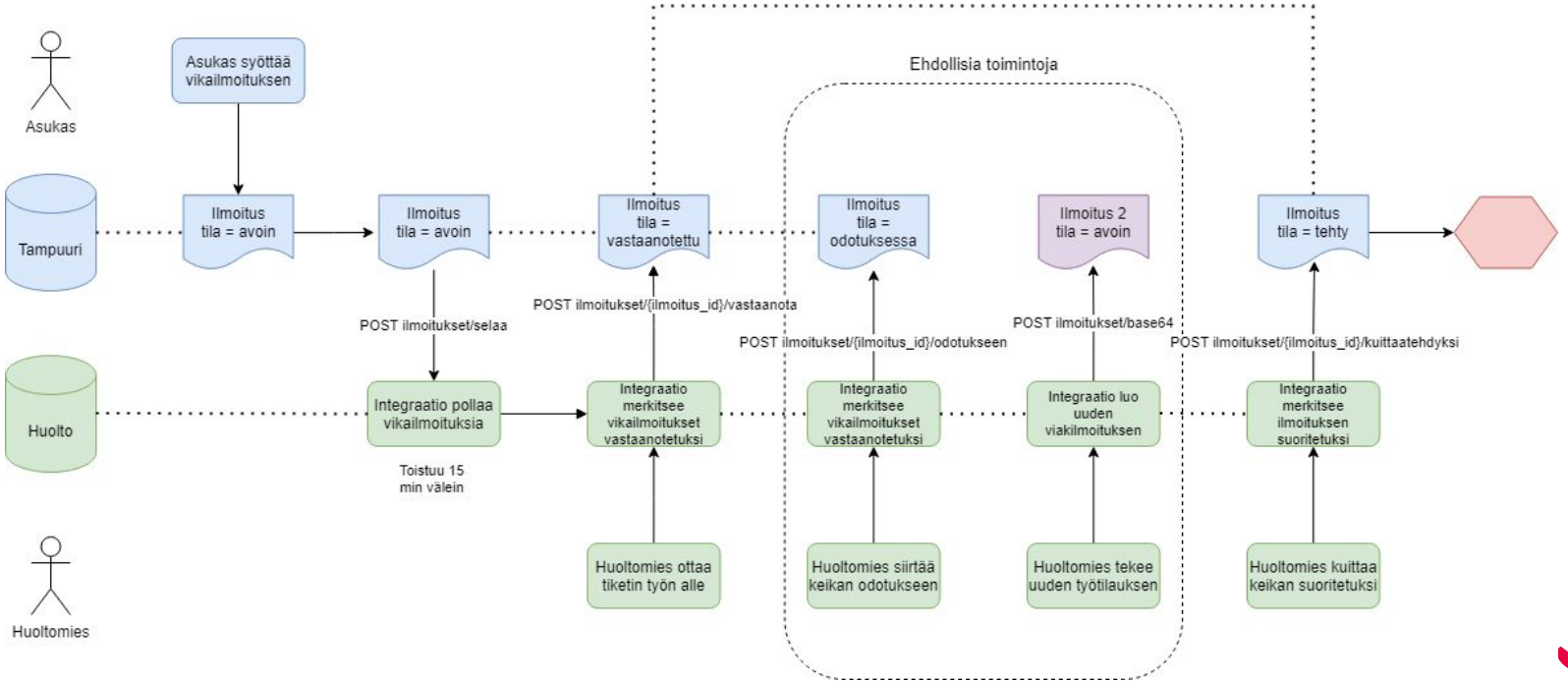
VISMA

Esimerkki integraatiosta: Huoltotehtävien vienti Tampuurista

- Kun tehtävä on suoritettu, prosessi jatkuu Tampuurin puolella vielä mahdollisesti laskutusvaiheen kautta tilaan “valmis”.
- Mikäli tehtävästä tuli jatkotehtäviä, on ne suoritettava ensin loppuun, jonka jälkeen alkuperäinen tehtävä voidaan kuitata valmiiksi.
    - Prosessi varmistaa, että asukassivujen käyttäjä pysyy koko ajan perillä tekemänsä ilmoituksen etenemisestä.

VISMA

Esimerkki integraatiosta: Huoltotehtävien vienti Tampuurista



Esimerkki integraatiosta: Huoltotehtävien vienti Tampuurista

Integraation jatkokehitys ja reaaliaikaisuuden tehostaminen

- Jos huomataan, että huoltokeikat eivät tule tasaisesti, jolloin pelkkä uusien ilmoitusten pollaaminen aiheuttaa vain turhia kutsuja, voidaan integraatiota kehittää tapahtumalähtöisempään suuntaan
  - Sen sijaan, että integraatio kävisi pollaamassa Tampuuri API:sta uusia keikkoja 15 minuutin välein, rekisteröidään integraatiolle kuuntelija, joka reagoi uusiin ilmoituksiin
    - Kuuntelija **IlmoitusUusi** -tapahtumalle
  - Kuuntelijan rekisteröinnin jälkeen Tampuuri tekee POST-kutsun kuuntelijaan jokaisesta uudesta ilmoituksesta

VISMA

Esimerkki integraatiosta: Vuokralaisten hallinta CRM:n kautta

Käyttötarve

- Asiakkaan vuokralaistiedon master-järjestelmä on heidän CRM:nsa
  - Asiakastiedot, sopimustiedot
- Tampuurin rooli on hoitaa vuokrareskontra, sekä kiinteistön ylläpitoon liittyvä prosessikokonaisuus (vikailmoitukset, talonkirja, huoltokalenteri)
  - Ajantasainen vuokralaistieto (osapuolirekisteri) on prosessien kannalta elintärkeä
- Asiakkaalla käytössään Tampuurin vuokrasopimuksen sähköinen irtisanominen, jonka pitäisi heijastua myös CRM:aan

VISMA

Esimerkki integraatiosta: Vuokralaisten hallinta CRM:n kautta

Integraation prosessi Tampuuri APIn avulla

- Henkilö tekee vuokrasopimuksen ja hänet ja hänen sopimuksensa kirjataan CRM:aan
- Integraatio selvittää hetun perusteella, onko osapuoli jo Tampuurissa
  - Kutsu GET osapuolet/henkilot?Hakusana={Hetu}
  - Parametrina henkilön hetu
  - Jos henkilö löytyy, paluuarvon mukana tulee osapuolen ID
- Mikäli henkilöä ei löydy, integraatio luo hänelle uuden osapuolen Tampuuriin
  - Kutsu POST osapuolet/henkilot
  - Parametreina kaikki tarvittavat henkilötiedot
  - Paluuarvona saadaan uuden osapuolen ID

VISMA

Esimerkki integraatiosta: Vuokralaisten hallinta CRM:n kautta

Integraation prosessi Tampuuri APIn avulla

- Kun henkilön osapuoli-ID on tiedossa, integraatio luo hänelle vuokrasopimuksen Tampuuriin
  - Kutsu POST sopimukset
  - Parametreina osapuolen ID, sopimuksen tiedot (esim. alkupvm), sekä asunnon kohde-ID
  - Paluuarvona saadaan uuden sopimuksen ID, joka talletetaan CRM:aan
- **Huom! CRM:n tulee tavalla tai toisella tuntea Tampuurin kohteet, tai pystyä selvittämään ne organisaation master data -järjestelmästä**

VISMA

Esimerkki integraatiosta: Vuokralaisten hallinta CRM:n kautta

Asumisen aikaiset päivitykset integraatiossa

- Sopimuksen voimassaolon aikana CRM pystyy integraation kautta päivittämään osapuolen tietoja
  - Kutsu PUT osapuolet/{osapuoli-id}
- Myös sopimuksen tietoja voidaan muokata integraatiosta monin eri tavoin
  - Esimerkki: Kutsu PATCH sopimukset/{sopimus-id} ← sopimuksen perustietojen muokkaus
  - Esimerkki: 2: Kutsu POST sopimukset/{sopimus_id}/maksulajit/lisaa ← lisää sopimukselle uuden maksulajin, esimerkiksi jos asukas varaa maksullisen saunavuoron tai autopaikan

VISMA

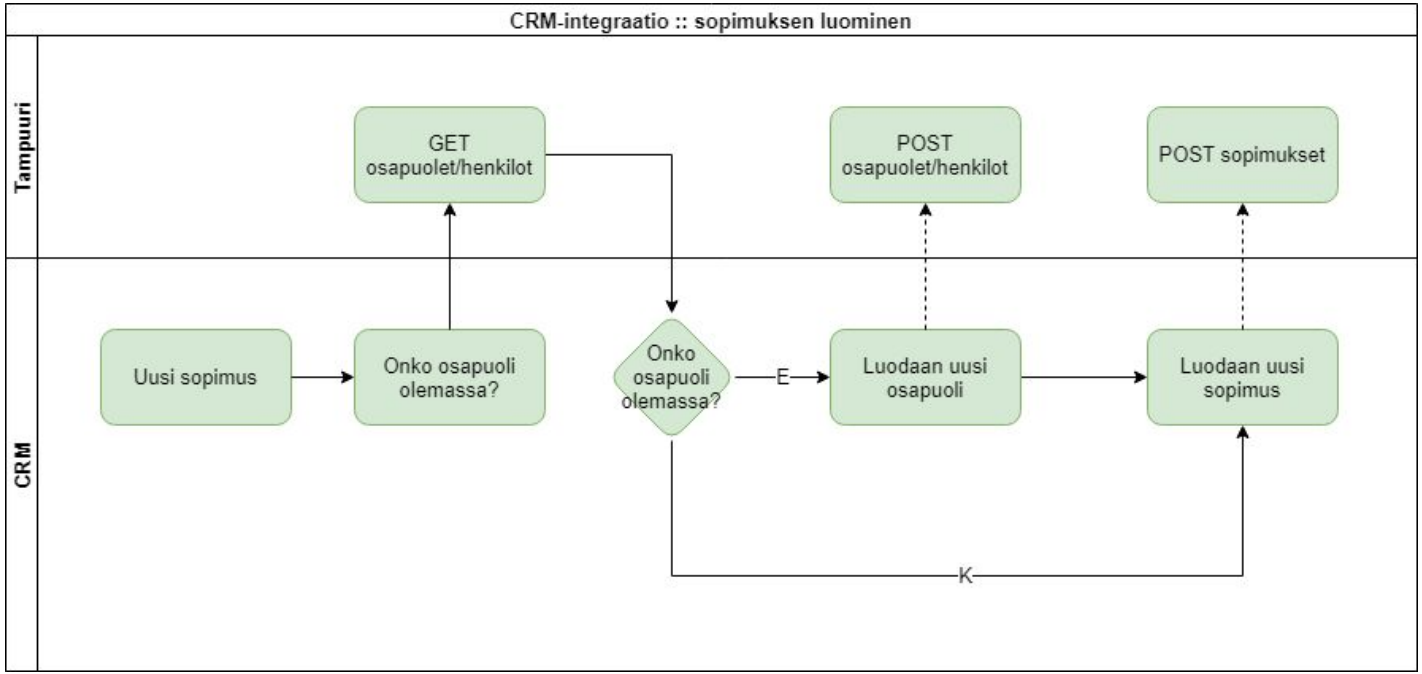
Esimerkki integraatiosta: Vuokralaisten hallinta CRM:n kautta

Sopimuksen päättäminen sähköisesti tai CRM:n kautta

- Kun asukas aikanaan irtisanoo vuokrasopimuksensa Tampuurin sähköisen irtisanomisen kautta, pitää tieto virkistää myös CRM:aan
  - Integraatio voi pollata sopimustietojen muutoksia viikoittain hakemalla kaikkien voimassaolevien sopimusten tiedot ja varmistamalla, onko niiden osalle merkitty **päättymispvm**
    - Kutsu GET sopimukset/{sopimus-id}
- Mikäli asukas irtisanoo vuokrasopimuksen olemalla yhteydessä asiakaspalveluun, se voidaan päättää myös CRM:n päässä ja viedä tieto integraation kautta Tampuuriin
  - Kutsu POST sopimukset/{sopimus_id}/paatasopimus

VISMA

Esimerkki integraatiosta: Vuokralaisten hallinta CRM:n kautta



Esimerkki integraatiosta: Vuokralaisten hallinta CRM:n kautta

Integraation jatkokehitys

- Jos sopimustietojen pollaamisesta haluttaisiin luopua ja saada sopimusten sähköisestä irtisanomisesta reaaliaikainen tieto, voidaan integraatioon ottaa mukaan webhookit
  - Sopimustietojen pollaamisen sijaan voidaan rekisteröidä kuuntelija, joka reagoi päättyviin sopimuksiin
    - Kuuntelija **SopimusPaata** -tapahtumalle
  - Kuuntelijan rekisteröinnin jälkeen Tampuuri tekee POST-kutsun kuuntelijaan, kun sopimukselle lisätään **päättymispvm**

VISMA

Yleisiä huomioita integraatioiden suunnitteluun

- Jos integraation vasteaikavaatimukset ovat suuret, kannattaa harkita webhookkeja verrattuna rajapinnan jatkuvaan pollaamiseen → usein pollaamisessa ~ 99 % liikenteestä ja transaktioista on turhia.
- Onko kyseessä julkinen verkkopalvelu vai muutamien asiantuntijoiden käyttämä järjestelmä?
  - Tampuuri API on operatiivisen järjestelmän selkäranka ja sen kuormitus heijastuu Tampuurin suorituskykyyn
  - Tampuuri API ei ole julkisen verkkopalvelun valmis back-end
  - Tampuuri API ei sovellu kuvapankiksi tai julkisen verkkopalvelun dokumenttivarastoksi
- Tampuuri APIn endpointit ja kutsut on aina piilotettava palvelun back-endiin
  - Esim. verkkopalvelun front-end ei saa tehdä suoria kutsuja Tampuuri APIiin siten, että kutsujen tiedot on kopioitavissa ja toistettavissa vaikkapa selaimen kehittäjätyökalujen avulla
- **Jos / kun integraatioissa käsitellään Tampuuri APIn kautta noudettavia henkilötietoja, pitää niiden käytössä huomioida GDPRn vaatimukset**

VISMA

Tampuuri APIn kehitysperiaatteet

- Tampuuri APIa päivitetään ns. Open / Closed -periaatteen mukaisesti
  - Version sisällä vältetään taaksepäin yhteensopivuuden rikkovia päivityksiä
    - Taaksepäin “rikkovat” päivitykset hoidetaan uusilla end-pointeilla tai laajentamalla tietomallia
    - Uuden pääversion julkaisun yhteydessä vanhentuneeksi merkityt end-pointit poistuvat
  - **HUOM! Tampuuri APIin perustuvien Integraatioiden tulee selvitä datamallin laajentamisesta**
    - **Koskee sekä apin responseja, että webhookkien sisältöä**
- Uusia end-pointteja tulee erilaisten kehitysprojektien seurauksena
  - Esimerkiksi sähköisen asioinnin palvelut ja kenttätyökalut pohjautuvat 100 % Tampuuri APIiin
  - Myös asiakasprojektit kasvattavat APIn kattavuutta → esimerkiksi sopimushallinnan end-pointit ovat seurausta isoista CRM-hankkeista

VISMA

Tampuuri APIn kustannukset

- Tampuuri APIn laskutus koostuu sekä API-kutsujen määrästä, että kiinteästä osasta, esimerkki:
  - Normal paketin kiinteä osa on 240 € / kk
  - Paketti sisältää 40 000 api-kutsua / kk ja tämän määrän ylittäviltä kutsuilta laskutetaan 0,009 € / kpl
  - Asiakas, jonka integraatiot käyttävät 50 000 kutsua kuukaudessa, maksaisi tästä paketista 330 € kuukaudessa

Standard
0 €/kk, 2 000 ilmaista kutsua, 0,011 €/kutsu

Normal
240 €/kk, 40 000 ilmaista kutsua, 0,009 €/kutsu

Business
400 €/kk, 100 000 ilmaista kutsua, 0,007 €/kutsu

VISMA

Tampuuri APIn kustannukset

- Yksi API-kutsu on mikä tahansa Tampuuri API:lle lähetetty komento
  - Esimerkiksi yksi GET kohteet/{kohde-id}
- Huom. Tampuuri API on saatavissa myös asiakkaan testi-tiliin integraation rakentamista varten
  - Testitiliin liitetystä API:sta ei veloiteta erikseen testiympäristön kustannusten lisäksi
- Webhook-tuki Tampuuri APIin maksaa kiinteät 500 € / kk

VISMA

Yhteenveto

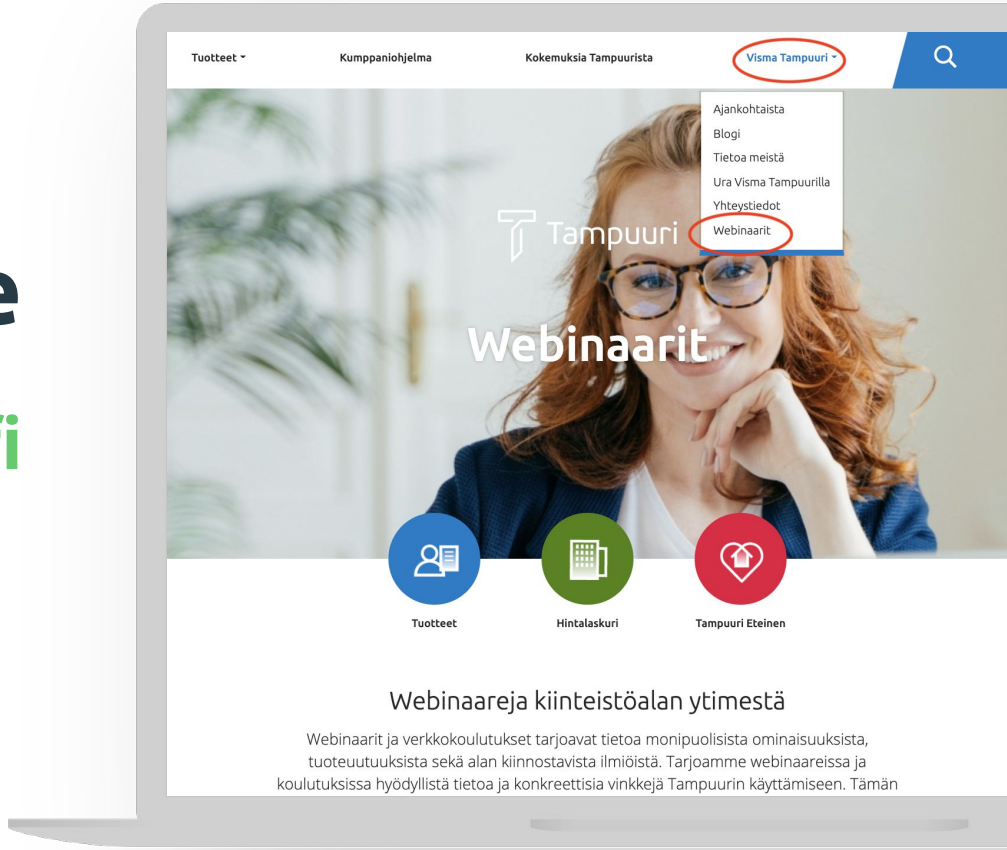
- Tampuuri API lyhyesti
  - Rajapinta Tampuurin ja ulkopuolisten järjestelmien integrointiin
    - Tyyppi: REST
    - Tiedonsiirtoformaatti: JSON
  - Laajuus tällä hetkellä ~400 operaatiota, mukana mm.
    - Vikailmoitukset
    - Vuokrasopimukset
    - Dokumenttipankki
    - Osapuoli- ja kohderekisterit
  - Tukee tapahtumapohjaista lähestymistapaa integraatioihin (webhookit)
- Käyttöönotto vaatii API-sopimuksen
  - Käyttöönotto lähtee liikkeelle ottamalla yhteyttä Tampuurin myyntiin -> **myyntipalvelu.tampuuri@visma.com**
  - Sopimuksen solmimisen jälkeen asiakas saa käyttöönsä tarvittavat tiedot APIn kutsumiseen

VISMA

Kysymykset

Tutustu myös muihin webinaareihimme

Osoitteessa **tampuuri.fi**



VISMA

Seuraa meitä somessa

@vismatampuuri

@vismatampuuri

@vismatampuuri

@vismafinland

@vismafinland

VISMA


Kiitos!