

Tampuuri API -käytön perusteet ja integraatioiden suunnittelu

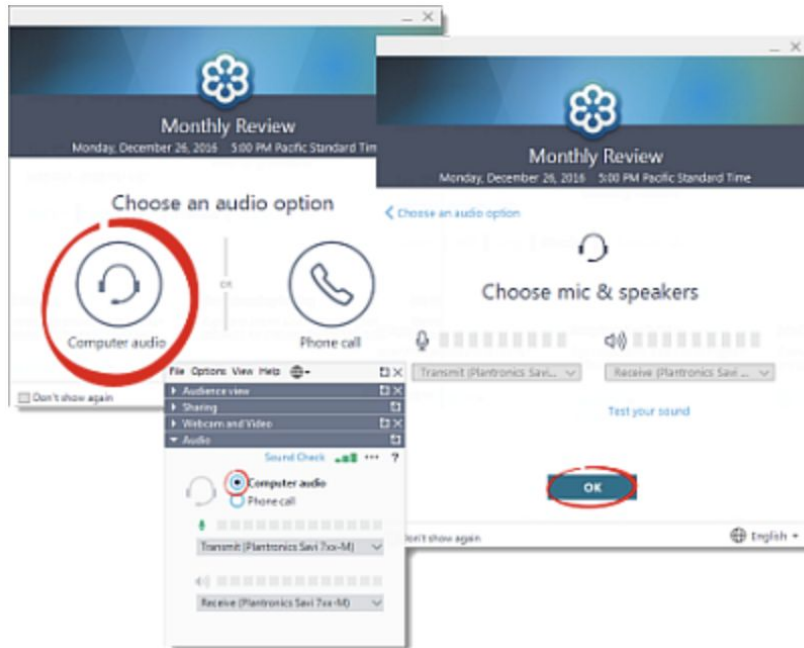
29.10.2021 klo 9-10

Timo Varis, Product Manager

Antti Hartikainen, Sales Engineer



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

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.

nguares

h, Swift,

ossible

la, Dart,

gian, C#,

ish, C++,

C Sharp,

, Kotlin,

Finnish.

Asiantuntijat



Antti Hartikainen

Sales Engineer

Visma Tampuuri



Timo Varis

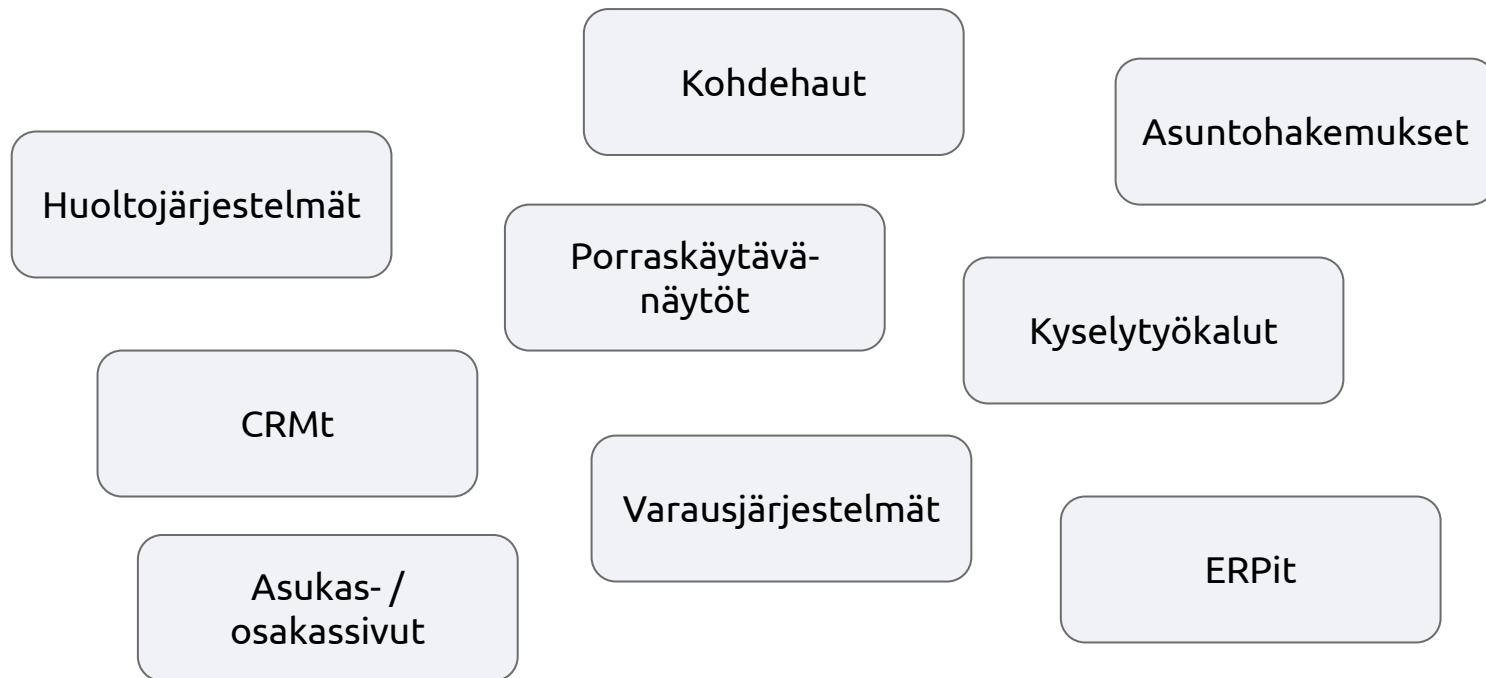
Product Manager

Visma Tampuuri

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 API:n päälle.

Millaisia integraatioita APilla on tehty



Tampuuri API ja Webhookit

- Tampuuri API:n on saatavana lisäpalveluna myös tuki tapahtumapohjaisille toteutuksille, eli ns. Webhookeille
- Webhookit mahdollistavat Tampuuri API:n integroituille 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 API:n 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

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.

Miten Tampuuri API otetaan käyttöön?

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

Tampuuri API:n käyttäminen

- Tampuuri API:n 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 ympäristökohtainen
 - 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`

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 tarkoitettut 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

```
Send Request
POST /ilmoitukset/selaa
X-TampuuriAvain: [{"TampuuriAvain": ""}]
X-AsiakasNimi: [{"AsiakasNimi": ""}]
X-Ohjelmistoversio: [{"Ohjelmistoversio": ""}]
X-TiliNimi: [{"TiliNimi": ""}]
X-Kayttajatunnus: [{"Kayttajatunnus": ""}]
X-Salanasana: [{"Salanasana": ""}]
Content-Type: [{"Content-Type": ""}]

{
  "TyonsuorittajaId": 459102,
  "VainPäätasonIlmoitukset": true
}
```

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.

```
451 {  
452   "KohdeMetatyytit": [  
453     "Kustannuspaikka"  
454   ],  
455   "Tilat": [  
456     1,  
457     2,  
458     3,  
459     4,  
460     5,  
461     6,  
462     7,  
463     8,  
464     9,  
465     10,  
466     11,  
467     12,  
468     13,  
469     14,  
470     16,  
471     50,  
472     100,  
473     110  
474   ],  
475   "KustannuspaikkaId": 1609493,  
476   "AlkaenVuodesta": 1980,  
477   "PaattynVuoteen": 2050  
478 }
```

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, **nyrkisääntönä 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

API-kutsun palauttama tieto

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

application/json, text/json

Sample:

```
{
  "Id": "sample string 1",
  "Alkaa": "2021-10-26T13:20:55.1824125+03:00",
  "Loppuu": "2021-10-26T13:20:55.1824125+03:00",
  "Kuitattu": true,
  "TyonsuorittajaYritys": {
    "Id": 1,
    "YritysId": 1,
    "Osapuolityyppi": 0,
    "Nimi": "sample string 2",
    "Etunimi": "sample string 3",
    "Sukunimi": "sample string 4",
    "Katuosoite": "sample string 5",
    "Postinumero": "sample string 6",
    "Postitoimipaikka": "sample string 7",
    "Puhelin": "sample string 8",
    "Matkapuhelin": "sample string 9",
    "Sahkoposti": "sample string 10",
    "Tunnus": "sample string 11",
    "Turvakielto": true,
    "Tunnistautuminen": {
      "Tyyppi": "Tupas",
      "Tunnistettu": "2021-10-26T13:20:55.1824125+03:00"
    },
    "Arkistoitu": "2021-10-26T13:20:55.1824125+03:00"
  },
  "TyonsuorittajaHenkilo": {
    "Td": 1.
```


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)

```
{  
  "Items": [],  
  "Metadata": {  
    "Offset": 0,  
    "Limit": 50,  
    "Total": 0  
  }  
}
```

Vinkki: Oma käyttöoikeusrooli APille

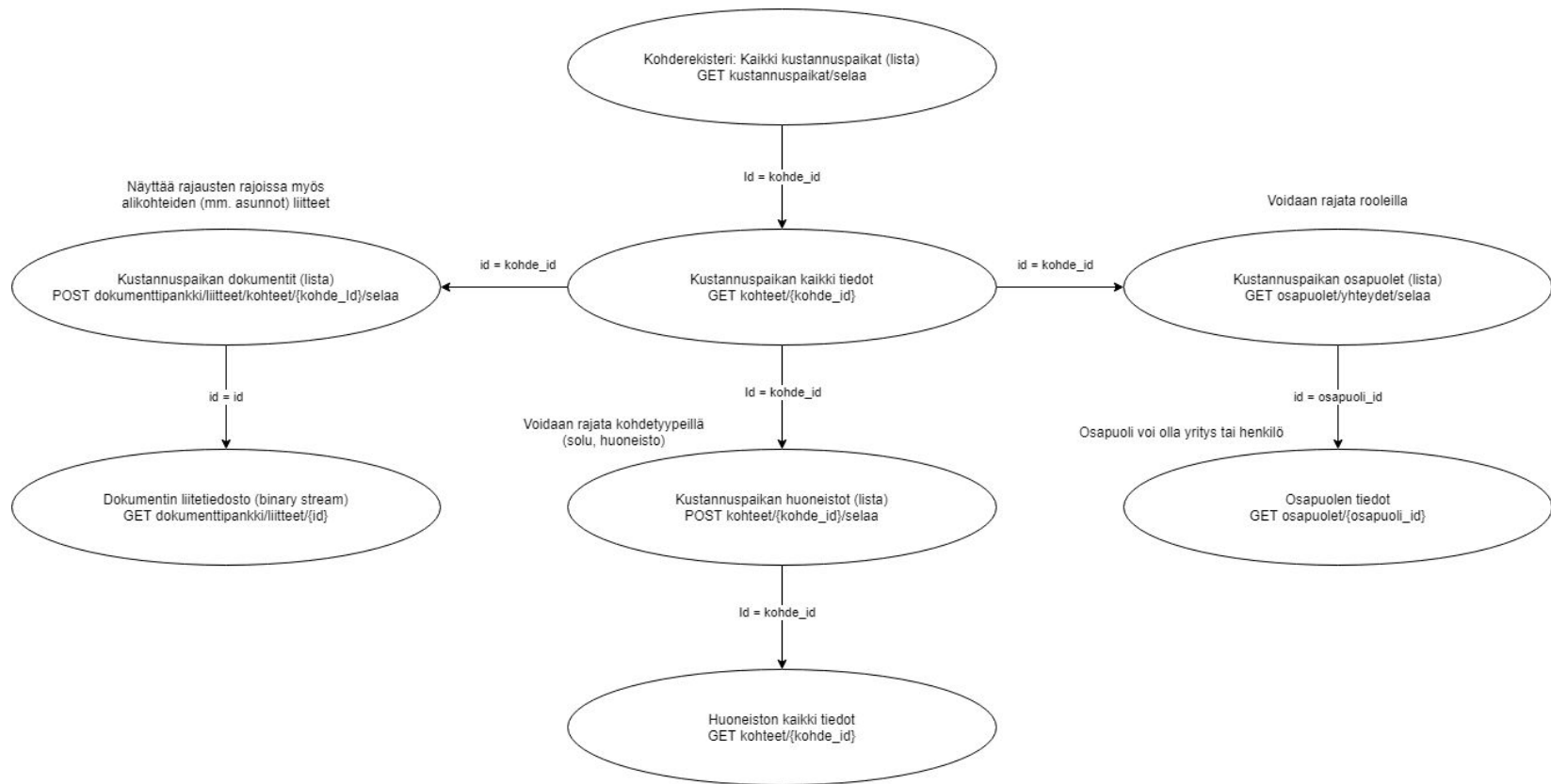
The screenshot shows the Tampuuri user management interface. On the left sidebar, the 'ROOLIHALLINTA' (Role Management) section is expanded, and the 'Käyttöoikeusrooli' (Usage Rights Role) is selected. The main content area displays the configuration for the 'Cloud' role. The 'Nimi' (Name) field is set to 'Cloud', and the 'Kuvaus' (Description) field contains 'Cloudia varten luotu rooli.'. The 'Kohdeoikeus' (Target Rights) dropdown menu is set to 'Kaikki' (All), which is highlighted with a red box. Below this, the 'Vain kohdetiedot' (Only target details) checkbox is checked. The 'Kaikkien roolien' (All roles) checkbox is unchecked, and the 'Tekstiviestivarmenne' (Text message backup) checkbox is also unchecked.

The screenshot shows the Tampuuri user management interface. On the left sidebar, the 'ROOLIHALLINTA' (Role Management) section is expanded, and the 'Käyttöoikeusrooli' (Usage Rights Role) is selected. The main content area displays the configuration for the 'Käyttöoikeusrooli' role. The 'Kohdeoikeus' (Target Rights) dropdown menu is set to 'Kaikki' (All), which is highlighted with a red box. Below this, the 'Vain kohdetiedot' (Only target details) checkbox is checked. The 'Kaikkien roolien' (All roles) checkbox is unchecked, and the 'Tekstiviestivarmenne' (Text message backup) checkbox is also unchecked. The right sidebar shows a list of roles, with 'Käyttöoikeusrooli' highlighted. The bottom right corner shows a list of roles, with 'Käyttöoikeusrooli' highlighted.

Peruskäsitteistö ja periaatteet

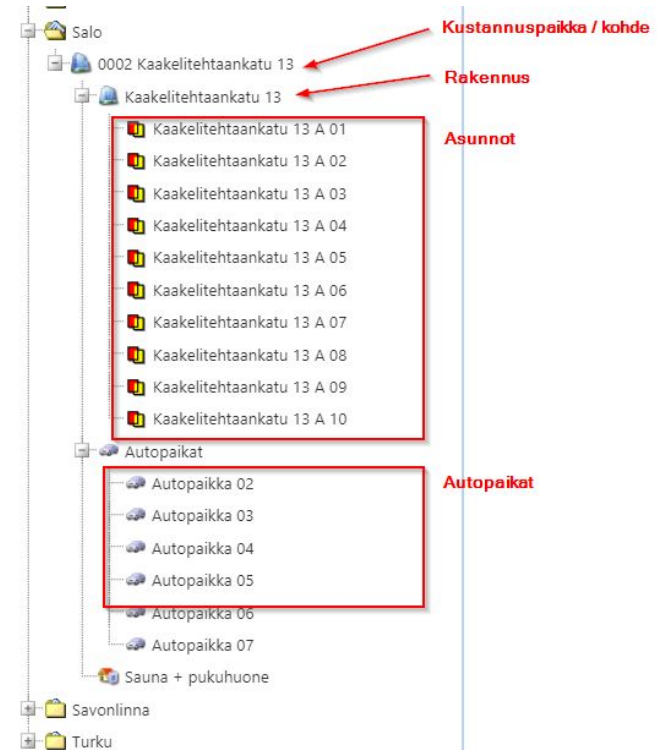
- APIssa eri resursseihin viitataan niiden sisäisellä IDlla, esimerkiksi asunnon, henkilön tai sopimuksen IDlla.
 - Nämä ovat yleensä muodoltaan juoksevia numeroita
 - Esim. GET sopimukset/{sopimus-id} → GET sopimukset/**32892**
 - APIa kutsuvan järjestelmän pitää tuntea nämä ID:t, tai niiden selvittämiseksi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

Esimerkki: Kohderekisteristä liikkeelle

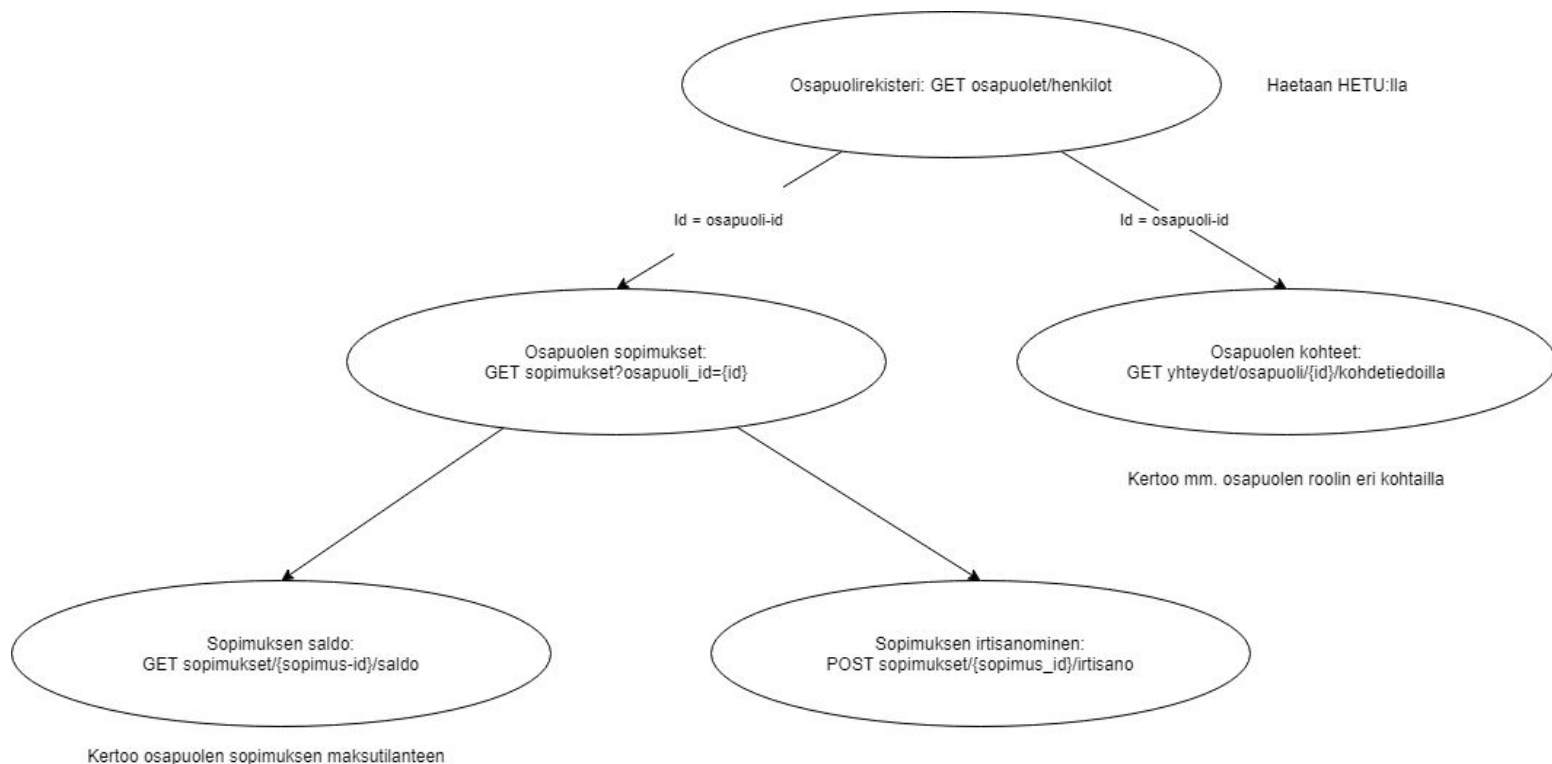


Esimerkki: Kohderekisteristä liikkeelle

- 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

Webhooks


Tapahtumien kuuntelijat

Palvelu vastaa normaalisti.

Nimi	Käytävissä olevat tapahtumat.	Kuvaus
Valitse Testitapahtuma	Testitapahtuma	
Valitse AsuntotarkastusUusi	Tapahtuma uudesta asuntotarkastuksesta	
Valitse AsuntotarkastusTilamuutos	Tapahtuma asuntotarkastuksen tilan muuttumisesta	
Valitse IlmoitusTilamuutos	Tapahtuma ilmoituksen tilan muuttumisesta	
Valitse DokumenttipankkiDokumenttiUusi	Tapahtuma dokumenttipankin uudesta liitteestä	
Valitse DokumenttipankkiDokumenttiMuokkaa	Tapahtuma dokumenttipankin liitteen muokkaamisesta	
Valitse DokumenttipankkiDokumenttiPoista	Tapahtuma dokumenttipankin liitteen poistamisesta	
Valitse KohdearakeräysKohdeTietoMuokkaa	Tapahtuma kohdearakeräyksen kohteen tiedon muutoksesta	
Valitse KiinteistöedusteluUusi	Tapahtuma kiinteistöedusteluun	
Valitse OsapuoliUusi	Tapahtuma uuden osapuolen lisäämisestä	
Valitse YritystenkirkkoUusi	Tapahtuma henkilön lisäämisestä yritykselle	
Valitse OsapuoliTietoMuokkaa	Tapahtuma osapuolen tiedon muutoksesta	
Valitse KohdeUusi	Tapahtuma uuden kohteen lisäämisestä	
Valitse OsapuoliPoista	Tapahtuma osapuolen poistamisesta	
Valitse KohdePoista	Tapahtuma kohteen poistamisesta	
Valitse KohdeOsapuoliUusi	Tapahtuma uudesta osapuolen ja kohteen yhteydestä	
Valitse KohdeOsapuoliMuokkaa	Tapahtuma osapuolen ja kohteen yhteyden muokkaamisesta	
Valitse RemonttiUusi	Tapahtuma uuden PTS-remontin lisäämisestä	
Valitse RemonttiMuokkaa	Tapahtuma PTS-remontin kenttien muokkaamisesta	
Valitse YritystenkirkkoPoista	Tapahtuma PTS-remontin yhteyden poistamisesta	
Valitse KäyttöpaikaväijennäköUusi	Tapahtuma kohteen käyttöpaikaväijennän merkinnän luomisesta	
Valitse KohdeOsapuoliPoista	Tapahtuma osapuolen ja kohteen yhteyden poistamisesta	
Valitse LaskutusjärjestelmäUusi	Tapahtuma kirjituksen hyväksymisestä. Tapahtumaan liittyy tapahtuman tyypin	
Valitse MuutosYhteydet	Tapahtuma osakkaan muutosten tilan muutoksesta	
Valitse IlmoitusUusi	Tapahtuma ilmoituksen luomisesta	
Valitse IlmoitusLiiteUusi	Tapahtuma ilmoituksen liitteen lisäämisestä	
Valitse IlmoitusLiitePoista	Tapahtuma ilmoituksen liitteen poistamisesta	
Valitse YhteydenottoUusi	Tapahtuma uudesta yhteydenotosta	
Valitse SopimusUusi	Tapahtuma uuden sopimuksen luomisesta	
Valitse SopimusPoista	Tapahtuma sopimuksen päättämisen asettamisesta	
Valitse IlmoitusMuokkaa	Tapahtuma ilmoituksen kenttien muokkaamisesta	
Valitse AspaTarjousTilaMuokkaa	Tapahtuma tarjouksen tilan muokkaamisesta	
Valitse AspaTarjousVoimassaPvmMuokkaa	Tapahtuma tarjouksen voimassaolon muokkaamisesta	
Valitse *	Listen in on all events.	

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)



POST #33a04	51.136.26.224	10/28/2021 9:16:00 AM
POST #673db	51.136.26.224	10/28/2021 9:16:02 AM
POST #a2d3f	51.136.26.224	10/28/2021 9:16:04 AM
POST #4efa4	51.136.26.224	10/28/2021 9:16:05 AM
POST #cca38	51.136.26.224	10/28/2021 9:16:06 AM
POST #70a1c	51.136.26.224	10/28/2021 9:16:08 AM
POST #2480a	51.136.26.224	10/28/2021 9:21:15 AM
POST #1931e	51.136.26.224	10/28/2021 9:21:19 AM
POST #ff2c0	51.136.26.224	10/28/2021 9:28:56 AM
POST #72ed2	51.136.26.224	10/28/2021 9:27:13 AM
POST #1972e	51.136.26.224	10/28/2021 9:27:14 AM
POST #892d5	51.136.26.224	10/28/2021 9:27:19 AM

Raw Content

```
{
  "Id": "cf33051d975d4c76824de403d2930e3e",
  "Attempt": 1,
  "Properties": {},
  "Notifications": [
    {
      "Action": "OsapuoliTieto/uokkaa",
      "Id": "2122421e-b637-ec11-a2c1-b07b25e407ff",
      "Account": "*****",
      "ReferenceId": "199497",
      "ReferenceSecondaryId": "",
      "ReferenceData": "14",
      "ReferenceJson": "",
      "Timestamp": "2021-10-28T06:13:18.12",
      "DateTimeZone": "UTC"
    },
    {
      "Action": "OsapuoliTieto/uokkaa",
      "Id": "2222421e-b637-ec11-a2c1-b07b25e407ff",
      "Account": "*****",
      "ReferenceId": "199497",
      "ReferenceSecondaryId": "",
      "ReferenceData": "3",
      "ReferenceJson": "",
      "Timestamp": "2021-10-28T06:13:18.162",
      "DateTimeZone": "UTC"
    },
    {
      "Action": "OsapuoliTieto/uokkaa",
      "Id": "2322421e-b637-ec11-a2c1-b07b25e407ff",
      "Account": "*****",
      "ReferenceId": "199497",
      "ReferenceSecondaryId": "",
      "ReferenceData": "4",
      "ReferenceJson": "",
      "Timestamp": "2021-10-28T06:13:18.162",
      "DateTimeZone": "UTC"
    }
  ]
}
```

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.

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

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.

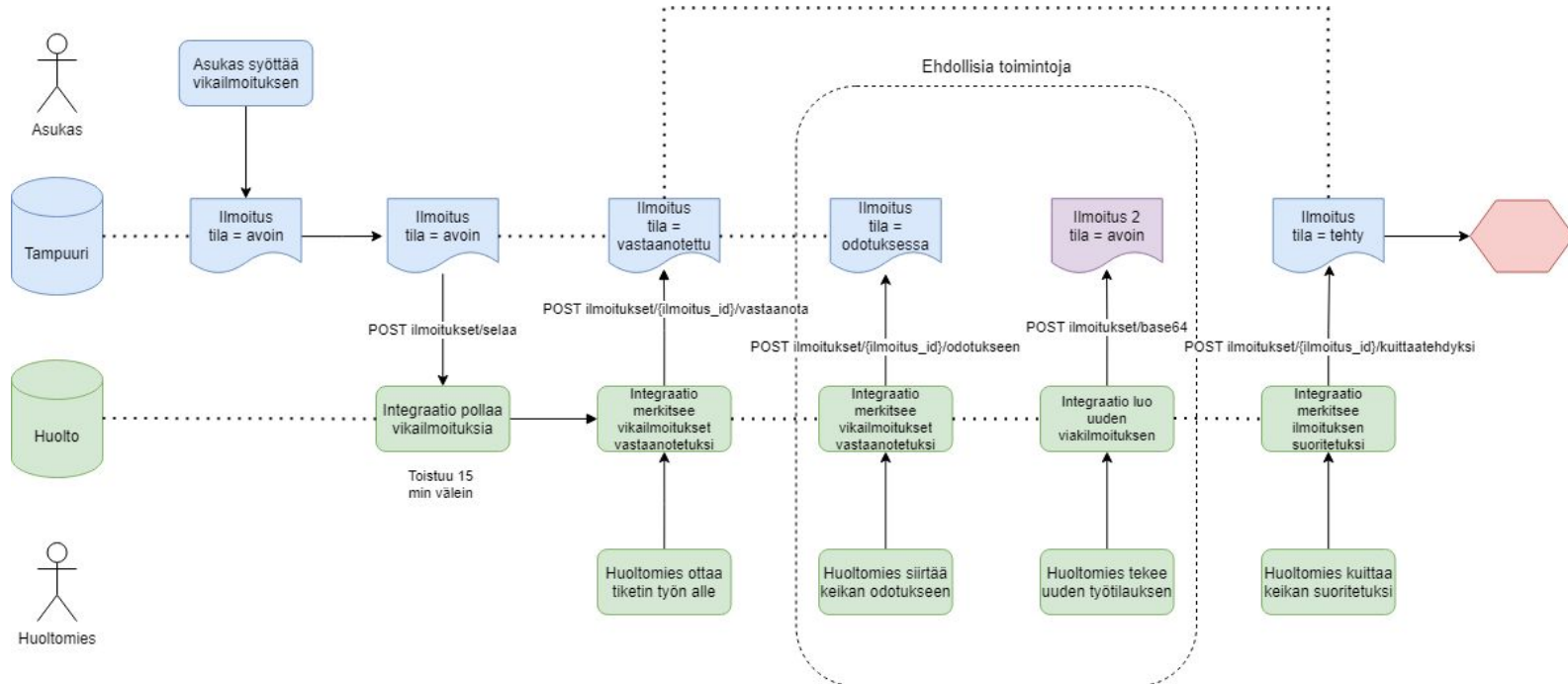
Esimerkki integraatiosta: Huoltotehtävien vienti Tampuurista

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

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.

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

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

Esimerkki integraatiosta: Vuokralaisten hallinta CRM:n kautta

Integraation prosessi Tampuuri API:n 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

Esimerkki integraatiosta: Vuokralaisten hallinta CRM:n kautta

Integraation prosessi Tampuuri API:n 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**

Esimerkki integraatiosta: Vuokralaisten hallinta CRM:n kautta

Asumisen aikaiset päivitykset integraatiossa

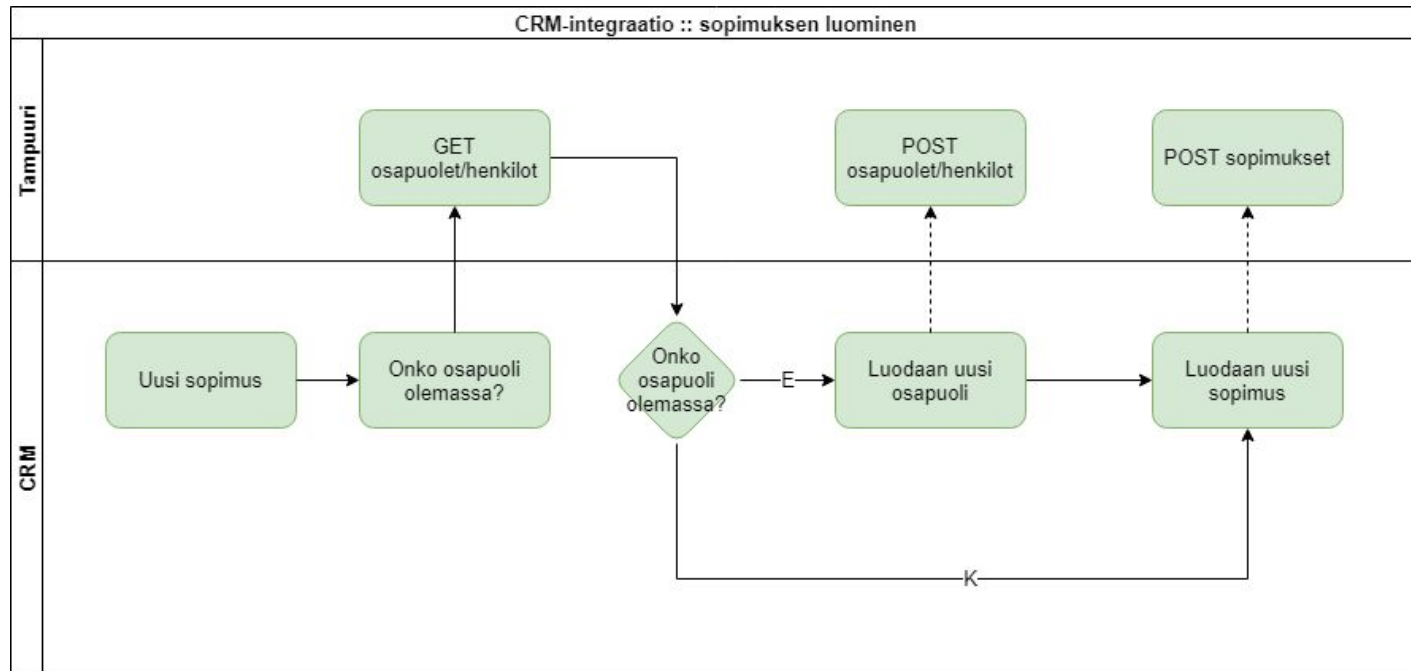
- Sopimuksen voimassaolon aikana CRM pystyy integraation kautta päivittämään osapuolen tietoja
 - Kutsu PUT osapuolet/{osapuoli-id}
- Myös sopimuksen tietoja voidaan muokata integraatiosta monin eri tavoin
 - Esimerkki: Kutsu PATCH sopimukset/{sopimus-id} ← sopimuksen perustietojen muokkaus
 - Esimerkki: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

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 **päätymispvm**
 - 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

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 **päätymispvm**

Yleisiä huomioita integraatioiden suunnitteluun

- Jos integraation vasteaikavaatimukset ovat suuret, kannattaa harkita webhookeja 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 API:n endpointit ja kutsut on aina piilotettava palvelun back-endiin
 - Esim. verkkopalvelun front-end ei saa tehdä suoria kutsuja Tampuuri API:iin siten, että kutsujen tiedot on kopioitavissa ja toistettavissa vaikkapa selaimen kehittäjätyökalujen avulla
- **Jos / kun integraatioissa käsitellään Tampuuri API:n kautta noudettavia henkilötietoja, pitää niiden käytössä huomioida GDPR:n vaatimukset**

Tampuuri API:n kehityisperiaatteet

- 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 API:n perustuvien Integraatioiden tulee selvittää datamallin laajentamisesta**
 - **Koskee sekä API:n responseja, että webhookkien sisältöä**
- Uusia end-pointteja tulee erilaisten kehitysprojektien seurauksena
 - Esimerkiksi sähköisen asiointin palvelut ja kenttätyökalut pohjautuvat 100 % Tampuuri APIiin
 - Myös asiakasprojektit kasvattavat API:n kattavuutta → esimerkiksi sopimushallinnan end-pointit ovat seurausta isoista CRM-hankkeista

Tampuuri API:n kustannukset

- Tampuuri API:n 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

Tampuuri API:n 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 API:n maksaa kiinteät 500 € / kk

Yhteenveto

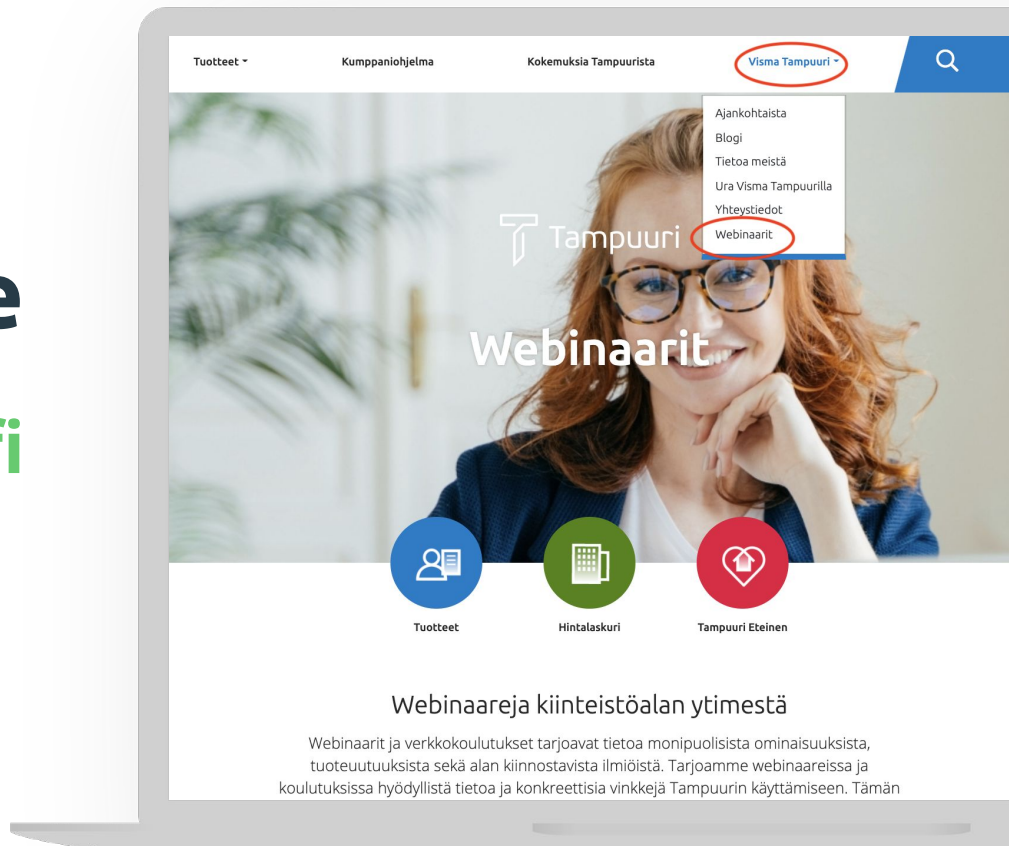
- Tampuuri API lyhyesti
 - Rajapinta Tampuurin ja ulkopuolisten järjestelmien integrointiin
 - Tyypä: REST
 - Tiedonsiirtoformaatti: JSON
 - Laajuus tällä hetkellä ~400 operaatiota, mukana mm.
 - Vikailmoitukset
 - Vuokrasopimukset
 - Dokumenttipankki
 - Osapuoli- ja kohderekisterit
 - Tukee tapahtumapohjaista 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 API:n kutsumiseen



Kysymykset

Tutustu myös muihin webinaareihimme

Osoitteessa **tampuuri.fi**



Seuraa meitä somessa



@vismatampuuri



@vismatampuuri



@vismatampuuri



@vismafinland



@vismafinland

A photograph of a man and a woman in an office hallway. The man, on the left, is wearing a light blue button-down shirt and dark trousers, smiling at the woman. The woman, on the right, is wearing a black top and holding a grey folder, smiling back at him. The hallway has wooden doors on the left and glass-walled offices on the right. The word "Kiitos!" is overlaid in large green letters in the center of the image.

Kiitos!