Swavo — guida per chi integra (API e avvisi)

Versione 1.0 · 25/11/2026 · Per chi sviluppa un gestionale, un CRM o un programma su misura e vuole collegarlo a Swavo.

Swavo è la segretaria AI per le piccole e medie imprese. **Giulia**, l'assistente vocale di Swavo, risponde al telefono dell'azienda: riconosce chi chiama, raccoglie la richiesta, fissa gli appuntamenti e manda i promemoria su WhatsApp. Questa guida spiega come il vostro programma parla con Swavo: dal vostro programma a Swavo con l'**API** (anagrafica, richieste, agenda), da Swavo al vostro programma con gli **avvisi** (una chiamata al vostro server ogni volta che succede qualcosa).

La guida basta per scrivere il collegamento senza chiederci le cose di base. **Ogni esempio è una chiamata vera**, fatta il 25/11/2026 sul nostro ambiente di produzione (`https://portal.swavo.ai`) su un'azienda di prova: le risposte sono copiate così come sono arrivate. La chiave è sempre mascherata (`sk_live_…`).

Per cominciare andate al §1 «Come si parte»: in cinque minuti fate la prima chiamata. Per sviluppare senza aspettare noi ci sono: avvisi di prova a comando, il registro delle consegne e il rimando, il registro delle vostre chiamate con il motivo di ogni errore, un'azienda di prova da svuotare quando volete e la descrizione OpenAPI da importare in Postman (§8.7). Questa guida è anche online, sempre aggiornata: https://portal.swavo.ai/sviluppatori/guida.

---

0. Casi tipici

Cosa vuole l'azienda	Direzione	Come
Giulia riconosce chi chiama, con i dati del vostro gestionale	Vostro programma → Swavo	Mandate l'anagrafica e tenetela allineata con **`POST /api/v1/customers/upsert`**: il cliente si riconosce dal telefono, rilanciare non crea doppioni (§3.1).
Le richieste raccolte al telefono entrano nel vostro programma	Swavo → vostro programma	**Avvisi firmati** `ticket.created` e `lead.created` verso un vostro indirizzo https (§8), oppure lettura con **`GET /api/v1/requests`** (§4.3). I dati raccolti da Giulia sono nel campo `details` (§4.1).
Le agende restano nel vostro programma, ma i clienti ricevono il promemoria WhatsApp	Vostro programma → Swavo, poi Swavo → vostro programma	Copiate gli appuntamenti in Swavo (**`POST /api/v1/appointments`**, poi **`PATCH /api/v1/appointments/:id`** per spostamenti e annullamenti, §5). Il giorno prima Swavo manda il promemoria; se l'azienda chiede la conferma, la risposta del cliente torna a voi con l'avviso **`appointment.status_changed`** (§6).
Giulia fissa gli appuntamenti sull'agenda di Swavo	Swavo → vostro programma	Giulia usa gli orari di lavoro, le ferie e le assenze di Swavo (§5.2, §5.6); voi ricevete ogni appuntamento con l'avviso **`appointment.created`**.
Sapere chi ha chiamato e cosa voleva	Swavo → vostro programma	Avvisi **`call.ended`** (con il riassunto di Giulia) e **`call.missed`** (chiamata persa), oppure **`GET /api/v1/calls`** (§7).
Collegare senza scrivere codice	—	L'app Swavo per **Zapier e Make** e le ricette per **n8n**: https://portal.swavo.ai/sviluppatori. Questa guida è per chi scrive il collegamento.

Parole che usiamo: **porta** = un indirizzo dell'API (es. `POST /api/v1/customers/upsert`) · **richiesta** = un intervento da gestire, quello che Giulia apre a fine telefonata (in altre parti dell'API si chiama *ticket*) · **avviso** = una chiamata che Swavo fa a un vostro indirizzo https quando succede qualcosa (in inglese *webhook*) · **operatore** = la persona che ha un'agenda (tecnico, medico, consulente…).

Porte di settore: quelle dei ristoranti (`/menu`, `/orders`) servono solo alle aziende di ristorazione e sono descritte su https://portal.swavo.ai/sviluppatori. Per le richieste usate `/requests`, non la porta generica `/tickets` (§4.4).

---

1. Come si parte

Questa sezione basta per fare la prima chiamata in cinque minuti. Le sezioni dopo (§2-§8) sono il riferimento completo, da consultare quando serve.

1.1 Cosa serve e cosa non serve

- **Non serve registrarsi su Swavo.** Niente account, niente password, niente accesso al portale web di Swavo: per l'integrazione non dovete mai entrarci. Tutto avviene dal vostro codice, con richieste HTTPS.
- **Serve solo una chiave** (comincia con `sk_live_`). Ogni chiave vale per **una sola azienda** e vede solo i suoi dati.
- Indirizzo di base: **`https://portal.swavo.ai/api/v1/`**. Tutte le porte (gli indirizzi dell'API) stanno lì sotto. Si parla solo in JSON (UTF-8), solo in HTTPS.
- **Descrizione completa per le macchine**: https://portal.swavo.ai/api/v1/openapi.json (OpenAPI 3.1, si apre senza chiave). In Postman o Insomnia: «Import → Link» e avete tutte le porte pronte da provare; con openapi-generator o NSwag (.NET) generate il client nel vostro linguaggio.
- Per qualunque dubbio: info@swavo.ai. Se una chiamata vi sembra sbagliata, mandateci l'intestazione `x-vercel-id` della risposta: ci porta dritti a quella chiamata (con curl la vedete aggiungendo `-i` al comando).

Come si ottiene la chiave.

Chi siete	Chiave	Come
State sviluppando o volete provare	**Chiave di prova**, su un'azienda di prova tutta vostra	Chiedetela a info@swavo.ai con oggetto «Chiave di prova API»: diteci chi siete e cosa volete collegare. La mandiamo noi, in genere in giornata. L'azienda di prova è vuota (due operatori con orari 8-20, nessun cliente, nessun numero di telefono né WhatsApp) e potete svuotarla quando volete (§8.7).
Collegate l'azienda di un cliente Swavo	**Chiave dell'azienda vera**	La crea il titolare dal portale, con il piano Pro: *Impostazioni → Integrazioni → API Keys → Nuova chiave*. Per iscrivervi agli avvisi da programma (§8.1) va spuntato «Userò questa chiave con Zapier o Make». Si vede una volta sola: il titolare ve la passa.
Siete già clienti Swavo con il piano Pro	La vostra	Come sopra, dal vostro portale.

Il codice è lo stesso per la prova e per l'azienda vera: cambia solo la chiave.

1.2 Dove va la chiave

La chiave va **nell'intestazione HTTP** `x-api-key` (un'intestazione è una riga che accompagna la richiesta, prima del corpo), **su ogni richiesta**.

- **Non** nell'indirizzo (`?api_key=…`): finirebbe nei registri dei server.
- **Non** in `Authorization: Bearer …`: viene ignorata e la risposta è `401` (provato).
- **Non** nel corpo JSON.

Una richiesta completa, così come viaggia:

```
POST /api/v1/customers/upsert HTTP/1.1
Host: portal.swavo.ai
x-api-key: sk_live_…la vostra chiave…
Content-Type: application/json

{"phone": "333 123 4567", "name": "Mario Rossi"}
```

`Content-Type: application/json` serve solo quando mandate un corpo (`POST`, `PATCH`).

Esito: tutto ciò che comincia con 2 è un successo. `200` per letture e modifiche, `201` **quando si crea qualcosa** (cliente nuovo, richiesta, appuntamento, iscrizione; per un'iscrizione anche quando quell'indirizzo era già iscritto). Controllate «2xx», non solo `200`. La risposta ha sempre anche `"success": true` o `false`.

1.3 Primo passo: verifica in un minuto

Copiate il comando in un terminale e mettete la vostra chiave al posto di `sk_live_…la_vostra_chiave…`.

Linux, macOS, Git Bash:

```
curl -s https://portal.swavo.ai/api/v1/me \
  -H "x-api-key: sk_live_…la_vostra_chiave…"
```

Windows, PowerShell (scrivete `curl.exe`, non `curl`: in PowerShell `curl` è un altro comando):

```
curl.exe -s https://portal.swavo.ai/api/v1/me `
  -H "x-api-key: sk_live_…la_vostra_chiave…"
```

Risposta ricevuta il 25/11/2026 (identica da bash, PowerShell 5.1 e PowerShell 7):

```
{
  "success": true,
  "data": {
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "company_name": "Azienda di prova API",
    "sector": "generic",
    "auth_method": "api_key",
    "sandbox": true,
    "can_subscribe_webhooks": true
  }
}
```

Se vedete `"success": true` e il nome della vostra azienda (qui `"Azienda di prova API"`), siete collegati. Se no, guardate la tabella al §1.5. `"sandbox": true` vuol dire che è un'azienda di prova, che potete svuotare quando volete (§8.7); con la chiave di un'azienda vera è `false`.

Per i comandi che seguono (§1.5 e §1.7), mettete una volta sola chiave e indirizzo in due variabili, nello stesso terminale.

Linux, macOS, Git Bash:

```
KEY=sk_live_…la_vostra_chiave…
API=https://portal.swavo.ai/api/v1
```

PowerShell:

```
$KEY = "sk_live_…la_vostra_chiave…"
$API = "https://portal.swavo.ai/api/v1"
```

1.4 La stessa chiamata dal vostro codice

Tutti e quattro provati: rispondono `200` con lo stesso JSON del §1.3. La chiave la leggono dalla variabile d'ambiente `SWAVO_API_KEY` (in produzione non scrivetela nel codice).

C# / .NET (va su .NET Framework 4.x e su .NET 6 e successivi):

```
using System;
using System.Net;
using System.Net.Http;

class Program
{
    static void Main()
    {
        // Solo .NET Framework 4.x: senza questa riga la connessione HTTPS fallisce
        ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
        var chiave = Environment.GetEnvironmentVariable("SWAVO_API_KEY");
        using (var http = new HttpClient())
        {
            http.DefaultRequestHeaders.Add("x-api-key", chiave);
            var url = "https://portal.swavo.ai/api/v1/me";
            var risposta = http.GetAsync(url).GetAwaiter().GetResult();
            var testo = risposta.Content.ReadAsStringAsync().GetAwaiter().GetResult();
            Console.WriteLine((int)risposta.StatusCode + " " + testo);
        }
    }
}
```

PHP (estensione `curl` attiva):

```
<?php
$chiave = getenv('SWAVO_API_KEY');
$ch = curl_init('https://portal.swavo.ai/api/v1/me');
curl_setopt($ch, CURLOPT_HTTPHEADER, ['x-api-key: ' . $chiave]);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$testo = curl_exec($ch);
if ($testo === false) { die('Errore di connessione: ' . curl_error($ch) . PHP_EOL); }
$stato = curl_getinfo($ch, CURLINFO_HTTP_CODE);
echo $stato . ' ' . $testo . PHP_EOL;
```

Su Windows, se esce `SSL certificate … unable to get local issuer certificate`, a PHP manca l'elenco dei certificati: scaricate `cacert.pem` da https://curl.se/ca/cacert.pem e indicatelo in `php.ini` con `curl.cainfo = "C:\percorso\cacert.pem"` (provato: senza dà questo errore, con risponde `200`).

Node.js (18 o successivo):

```
// Node.js 18 o successivo (fetch già incluso)
const risposta = await fetch('https://portal.swavo.ai/api/v1/me', {
  headers: { 'x-api-key': process.env.SWAVO_API_KEY },
});
console.log(risposta.status, await risposta.text());
```

Python 3 (nessuna libreria da installare):

```
import os, urllib.request

richiesta = urllib.request.Request(
    "https://portal.swavo.ai/api/v1/me",
    headers={"x-api-key": os.environ["SWAVO_API_KEY"]},
)
with urllib.request.urlopen(richiesta) as risposta:
    print(risposta.status, risposta.read().decode("utf-8"))
```

1.5 Cosa può andare storto

Risposta	Causa più probabile	Cosa fare
`401 "Authentication required: send the x-api-key header"`	La chiave non è arrivata: manca l'intestazione, ha un altro nome, oppure è in `Authorization: Bearer`	Mettetela in `x-api-key` (§1.2)
`401 "Invalid API key"`	Chiave scritta male (spazio o carattere in più, copiata a metà), revocata o scaduta	Ricopiatela intera; se persiste, scriveteci
`403 "API e webhook sono disponibili dal piano Pro"`	L'azienda della chiave non ha l'API attiva (serve il piano Pro). Con la chiave di prova non succede	Il titolare dell'azienda passa al piano Pro, oppure scriveteci
`404 "Questo percorso non esiste. L'elenco delle porte: GET /api/v1/openapi.json"`	Porta inesistente (es. `/api/v1/clienti` invece di `/api/v1/customers`)	Controllate il nome della porta in questa guida
`404 "Customer not found"` (o simile) in JSON	L'id non esiste, o è di un'altra azienda	Controllate l'id: è quello che vi abbiamo restituito noi?
`307`, rimando a `…/login`	Manca `/api/v1/` nel percorso: state chiamando il portale web, non l'API	Ogni indirizzo comincia con `https://portal.swavo.ai/api/v1/`
`400 "phone: Invalid input"`	Corpo non valido: il messaggio dice **quale campo** (qui: telefono mancante)	Correggete quel campo (tabelle dei campi dal §3)
`400 "Invalid input"` senza nome di campo	JSON rotto (es. virgola in fondo) o `Content-Type` mancante	Controllate il JSON e l'intestazione `Content-Type: application/json`
`429 "API rate limit exceeded"`	Più di 1000 chiamate nell'ora	Aspettate i secondi indicati nell'intestazione `Retry-After`, poi riprendete

Le risposte qui sopra sono state ottenute davvero. Due esempi:

```
curl -s -w "\nHTTP %{http_code}\n" https://portal.swavo.ai/api/v1/me \
  -H "Authorization: Bearer $KEY"
```

```
{"success":false,"error":{"code":"UNAUTHORIZED","message":"Authentication required: send the x-api-key header"}}
HTTP 401
```

```
curl -s -w "\nHTTP %{http_code}\n" -X POST $API/customers/upsert \
  -H "x-api-key: $KEY" -H "Content-Type: application/json" \
  -d '{"name": "Cliente senza telefono"}'
```

```
{"success":false,"error":{"code":"VALIDATION_ERROR","message":"phone: Invalid input"}}
HTTP 400
```

`403` e `429` non si possono provocare con la chiave di prova: li abbiamo provati con due chiavi di collaudo (una su un'azienda senza piano Pro, una con un limite di 2 chiamate l'ora) e le risposte sono al §2.6. Un errore `500` è sempre un guasto nostro: ritentate più tardi e, se continua, scriveteci.

1.6 Regole sulla chiave

- È una password: **non** va in repository pubblici, né nel codice che gira sul computer o nel browser di un utente, né negli indirizzi (finirebbe nei registri dei server).
- Se pensate che sia uscita, ditecelo: la revochiamo e ve ne diamo una nuova in pochi minuti (la vecchia smette subito di funzionare).
- Quella di prova **scade dopo 60 giorni** (la data ve la diciamo quando ve la mandiamo). Se il lavoro continua, la rinnoviamo.
- Limite: **1000 chiamate l'ora** per chiave. Per un import iniziale grande (un'anagrafica di migliaia di clienti) lo alziamo: diteci quanti sono.

1.7 Il primo giorno: percorso in 6 passi

Alla fine di questi sei passi avete toccato le quattro cose che servono quasi sempre: anagrafica, richieste, agenda e avvisi. Ogni comando è stato eseguito il 25/11/2026 sull'azienda di prova; sotto c'è la risposta vera (riformattata su più righe per leggerla meglio). Gli id negli esempi (cliente, iscrizione, appuntamento…) sono quelli delle nostre prove: i vostri saranno diversi.

Usate le variabili `KEY` e `API` definite alla fine del §1.3.

In PowerShell il corpo JSON si mette in una variabile e si passa a `curl.exe` con `--data-binary '@-'`: è il modo che funziona uguale su PowerShell 5.1 e 7 (scritto dentro al comando, PowerShell 5.1 toglie le virgolette e il JSON arriva rotto). In PowerShell 5.1 le lettere accentate nei corpi arrivano come `?` (provato: «Niccolò» diventa «Niccol?»; in PowerShell 7 arrivano giuste): per le prove da 5.1 usate testo senza accenti. Dal vostro codice (§1.4) nessuno dei due problemi esiste.

Passo 1 — Verifico la chiave (`GET /me`)

```
curl -s $API/me -H "x-api-key: $KEY"
```

```
curl.exe -s "$API/me" -H "x-api-key: $KEY"
```


```
{
  "success": true,
  "data": {
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "company_name": "Azienda di prova API",
    "sector": "generic",
    "auth_method": "api_key",
    "sandbox": true,
    "can_subscribe_webhooks": true
  }
}
```


Passo 2 — Creo un cliente (`POST /customers/upsert`)

«Upsert» vuol dire: se il telefono non c'è, crea il cliente; se c'è, lo aggiorna. Rilanciare lo stesso comando non crea doppioni. Il telefono si manda come lo scrive una persona: Swavo lo porta a `+39…`.

```
curl -s -X POST $API/customers/upsert \
  -H "x-api-key: $KEY" -H "Content-Type: application/json" \
  -d '{"phone": "333 123 4567", "name": "Mario Rossi",
       "address": "Via Roma 1, Genova", "notes": "Codice gestionale C-1003"}'
```

```
$body = '{"phone": "333 123 4567", "name": "Mario Rossi",
  "address": "Via Roma 1, Genova", "notes": "Codice gestionale C-1003"}'
$body | curl.exe -s -X POST "$API/customers/upsert" `
  -H "x-api-key: $KEY" -H "Content-Type: application/json" --data-binary '@-'
```


```
{
  "success": true,
  "data": {
    "id": "9d7ffc12-745c-43ae-acfe-02d43b80cfe9",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "name": "Mario Rossi",
    "phone_number": "+393331234567",
    "email": null,
    "address": "Via Roma 1, Genova",
    "notes": "Codice gestionale C-1003",
    "marketing_consent": false,
    "source": "api",
    "created_at": "2026-11-25T21:29:29.128047+00:00",
    "updated_at": "2026-11-25T21:29:29.128047+00:00",
    "created": true
  }
}
```


`"created": true`: il cliente è nuovo. Rilanciando lo stesso comando esce `"created": false` e lo stesso `id`.

Passo 3 — Lo ritrovo dal telefono (`GET /customers/find`)

È quello che fa Giulia quando squilla il telefono dell'azienda: riconosce il cliente dal numero.

```
curl -s "$API/customers/find?phone=3331234567" -H "x-api-key: $KEY"
```

```
curl.exe -s "$API/customers/find?phone=3331234567" -H "x-api-key: $KEY"
```


```
{
  "success": true,
  "data": [
    {
      "id": "9d7ffc12-745c-43ae-acfe-02d43b80cfe9",
      "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
      "name": "Mario Rossi",
      "phone_number": "+393331234567",
      "email": null,
      "address": "Via Roma 1, Genova",
      "notes": "Codice gestionale C-1003",
      "marketing_consent": false,
      "source": "api",
      "created_at": "2026-11-25T21:29:29.128047+00:00",
      "updated_at": "2026-11-25T21:29:29.128047+00:00"
    }
  ]
}
```

Nessun cliente con quel numero = `"data": []` (lista vuota), non un errore.

Passo 4 — Apro una richiesta (`POST /requests`)

Una «richiesta» è un intervento da gestire: è la stessa cosa che Giulia crea quando chiude una telefonata (§4).

```
curl -s -X POST $API/requests \
  -H "x-api-key: $KEY" -H "Content-Type: application/json" \
  -d '{"phone": "3331234567", "type": "service",
       "description": "Perdita sotto il lavello, chiede un sopralluogo"}'
```

```
$body = '{"phone": "3331234567", "type": "service",
  "description": "Perdita sotto il lavello, chiede un sopralluogo"}'
$body | curl.exe -s -X POST "$API/requests" `
  -H "x-api-key: $KEY" -H "Content-Type: application/json" --data-binary '@-'
```


```
{
  "success": true,
  "data": {
    "id": "04bfbdfd-56a3-4023-8154-fe5419b4e8aa",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "9d7ffc12-745c-43ae-acfe-02d43b80cfe9",
    "issue_type": "service",
    "description": "Perdita sotto il lavello, chiede un sopralluogo",
    "priority": "normal",
    "status": "new",
    "assigned_technician_id": null,
    "customer_name": "Mario Rossi",
    "customer_phone": "+393331234567",
    "customer_address": "Via Roma 1, Genova",
    "is_emergency": false,
    "source": "api",
    "lead_status": "lead",
    "resolution_notes": null,
    "created_at": "2026-11-25T21:29:30.466668+00:00",
    "updated_at": "2026-11-25T21:29:30.466668+00:00",
    "completed_at": null,
    "details": {},
    "customer_created": false
  }
}
```


`lead_status: "lead"`: in questa azienda ogni richiesta è anche un contatto da richiamare (dipende dalle impostazioni dell'azienda; altrimenti è `null`). `details` è vuoto perché la richiesta non viene da una telefonata: in quelle di Giulia contiene i dati raccolti al telefono (§4.1).

Passo 5 — Leggo gli operatori e gli orari liberi (`GET /technicians`, `GET /appointments/slots`)

```
curl -s $API/technicians -H "x-api-key: $KEY"
curl -s "$API/appointments/slots?from=2026-12-02&to=2026-12-02&duration_minutes=60&limit=3" \
  -H "x-api-key: $KEY"
```

```
curl.exe -s "$API/technicians" -H "x-api-key: $KEY"
curl.exe -s "$API/appointments/slots?from=2026-12-02&to=2026-12-02&duration_minutes=60&limit=3" `
  -H "x-api-key: $KEY"
```

```
{
  "success": true,
  "data": {
    "data": [
      {
        "id": "2c31dc6b-df38-430e-ae8a-94af38f5c2bc",
        "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
        "name": "Laura",
        "phone": null,
        "email": null,
        "is_active": true,
        "color": null,
        "notes": null,
        "created_at": "2026-11-25T20:59:16.411654+00:00",
        "specialization": null
      },
      {
        "id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
        "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
        "name": "Titolare",
        "phone": null,
        "email": null,
        "is_active": true,
        "color": null,
        "notes": "Agenda di base creata dal settore: rinomina l'operatore e sistema orari e giorni.",
        "created_at": "2026-11-25T20:58:52.155119+00:00",
        "specialization": null
      }
    ],
    "total": 2,
    "limit": 20,
    "offset": 0
  }
}
```



```
{
  "success": true,
  "data": [
    {
      "date": "2026-12-02",
      "time": "08:00",
      "end_time": "09:00",
      "duration_minutes": 60,
      "technician_id": "2c31dc6b-df38-430e-ae8a-94af38f5c2bc",
      "technician_name": "Laura"
    },
    {
      "date": "2026-12-02",
      "time": "08:00",
      "end_time": "09:00",
      "duration_minutes": 60,
      "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
      "technician_name": "Titolare"
    },
    {
      "date": "2026-12-02",
      "time": "09:00",
      "end_time": "10:00",
      "duration_minutes": 60,
      "technician_id": "2c31dc6b-df38-430e-ae8a-94af38f5c2bc",
      "technician_name": "Laura"
    }
  ]
}
```


Se quel giorno è già passato, mettete un giorno feriale futuro (in `from` e `to`, formato `AAAA-MM-GG`, ora italiana).

Passo 6 — Fisso un appuntamento e ricevo il mio primo avviso

Un **avviso** (webhook) è una chiamata che Swavo fa **a voi**, verso un vostro indirizzo https, quando succede qualcosa (un appuntamento creato, una richiesta di Giulia…). Ogni avviso è **firmato**: una firma calcolata con un segreto che conoscete solo voi, per essere sicuri che venga da noi (§8.3).

6a. Un indirizzo dove ricevere, senza avere ancora un server. Aprite https://webhook.site nel browser: vi dà subito un indirizzo personale (`https://webhook.site/…`, «Your unique URL») e mostra in diretta tutto quello che riceve. Senza browser, lo stesso indirizzo si ottiene da terminale (il vostro indirizzo sarà `https://webhook.site/` seguito dal campo `uuid` della risposta; se webhook.site risponde «rate limited», aspettate i secondi che indica e riprovate):

```
curl -s -X POST https://webhook.site/token -H "Accept: application/json"
```

```
curl.exe -s -X POST https://webhook.site/token -H "Accept: application/json"
```

È gratuito, senza registrazione, adatto solo a dati di prova (chi conosce l'indirizzo vede cosa arriva; nella versione gratuita può smettere di ricevere dopo una cinquantina di richieste: in quel caso createne un altro). In alternativa potete esporre un vostro programma locale con un tunnel https (ngrok, Cloudflare Tunnel): `http://` semplice è rifiutato.

6b. Mi iscrivo all'evento `appointment.created`. Sostituite l'indirizzo con il vostro di webhook.site:

```
curl -s -X POST $API/hooks \
  -H "x-api-key: $KEY" -H "Content-Type: application/json" \
  -d '{"url": "https://webhook.site/e1019b6d-d28f-405a-b160-97f5ca8ab3aa",
       "event": "appointment.created"}'
```

```
$body = '{"url": "https://webhook.site/e1019b6d-d28f-405a-b160-97f5ca8ab3aa",
  "event": "appointment.created"}'
$body | curl.exe -s -X POST "$API/hooks" `
  -H "x-api-key: $KEY" -H "Content-Type: application/json" --data-binary '@-'
```


```
{
  "success": true,
  "data": {
    "id": "e763e622-1a6c-4a63-b248-facf0acc2f0c",
    "event": "appointment.created",
    "app": "altro",
    "secret": "whsec_124f01…"
  }
}
```


Conservate `secret` (qui accorciato): serve a verificare la firma degli avvisi. Se lo perdete, rifate la stessa chiamata con lo stesso indirizzo: la risposta ridà lo stesso segreto. Conservate anche `id`: serve per cancellare l'iscrizione.

6c. Fisso l'appuntamento (con l'`id` dell'operatore letto al passo 5 e un orario libero):

```
curl -s -X POST $API/appointments \
  -H "x-api-key: $KEY" -H "Content-Type: application/json" \
  -d '{"phone": "3331234567", "date": "2026-12-02", "time": "08:00",
       "duration_minutes": 60, "title": "Sopralluogo perdita",
       "technician_id": "2c31dc6b-df38-430e-ae8a-94af38f5c2bc"}'
```

```
$body = '{"phone": "3331234567", "date": "2026-12-02", "time": "08:00",
  "duration_minutes": 60, "title": "Sopralluogo perdita",
  "technician_id": "2c31dc6b-df38-430e-ae8a-94af38f5c2bc"}'
$body | curl.exe -s -X POST "$API/appointments" `
  -H "x-api-key: $KEY" -H "Content-Type: application/json" --data-binary '@-'
```


```
{
  "success": true,
  "data": {
    "id": "80246c2b-abfd-4ef6-8cfa-952b5723af49",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "9d7ffc12-745c-43ae-acfe-02d43b80cfe9",
    "technician_id": "2c31dc6b-df38-430e-ae8a-94af38f5c2bc",
    "title": "Sopralluogo perdita",
    "notes": null,
    "scheduled_date": "2026-12-02",
    "start_time": "08:00:00",
    "end_time": "09:00:00",
    "estimated_duration_minutes": 60,
    "status": "scheduled",
    "source": "manual",
    "created_at": "2026-11-25T21:29:32.37567+00:00",
    "customer": {
      "id": "9d7ffc12-745c-43ae-acfe-02d43b80cfe9",
      "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
      "name": "Mario Rossi",
      "phone_number": "+393331234567",
      "email": null,
      "address": "Via Roma 1, Genova",
      "notes": "Codice gestionale C-1003",
      "marketing_consent": false,
      "source": "api",
      "created_at": "2026-11-25T21:29:29.128047+00:00",
      "updated_at": "2026-11-25T21:29:32.132+00:00"
    },
    "customer_created": false
  }
}
```


6d. L'avviso arriva. Dopo circa un secondo, sulla pagina di webhook.site compare questa richiesta (ricevuta davvero; firma accorciata):

```
POST /e1019b6d-d28f-405a-b160-97f5ca8ab3aa
Content-Type: application/json
User-Agent: Swavo-Webhooks/1.0
Swavo-Event: appointment.created
Swavo-Delivery: 74e274da-ac24-47b5-85ee-a1efd97ed411
Swavo-Signature: t=1795642172,v1=455515a1…
Idempotency-Key: appointment.created:80246c2b-abfd-4ef6-8cfa-952b5723af49:e763e622-1a6c-4a63-b248-facf0acc2f0c

{
  "id": "74e274da-ac24-47b5-85ee-a1efd97ed411",
  "event": "appointment.created",
  "created_at": "2026-11-25T21:29:32.993Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "data": {
    "id": "80246c2b-abfd-4ef6-8cfa-952b5723af49",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "9d7ffc12-745c-43ae-acfe-02d43b80cfe9",
    "ticket_id": null,
    "technician_id": "2c31dc6b-df38-430e-ae8a-94af38f5c2bc",
    "service_id": null,
    "title": "Sopralluogo perdita",
    "reason": null,
    "notes": null,
    "scheduled_date": "2026-12-02",
    "start_time": "08:00:00",
    "end_time": "09:00:00",
    "status": "scheduled",
    "priority": "normal",
    "address": null,
    "location_type": "on_site",
    "estimated_cost": null,
    "currency": "EUR",
    "source": "manual",
    "created_at": "2026-11-25T21:29:32.37567+00:00",
    "updated_at": "2026-11-25T21:29:32.37567+00:00",
    "customer_name": "Mario Rossi",
    "customer_phone": "+393331234567",
    "customer_email": null
  }
}
```

Senza browser, quello che è arrivato si legge da terminale (al posto di `IL-VOSTRO-UUID` il codice del vostro indirizzo). Nella risposta di webhook.site l'avviso più recente è il primo di `data`: il corpo, così come è arrivato, è nel campo `content` (testo); le intestazioni sono in `headers`, con il nome in minuscolo e il valore dentro una lista (es. `headers["swavo-signature"][0]`). Per verificare la firma (§8.3) usate `content` così com'è:

```
curl -s "https://webhook.site/token/IL-VOSTRO-UUID/requests?sorting=newest" \
  -H "Accept: application/json"
```

```
curl.exe -s "https://webhook.site/token/IL-VOSTRO-UUID/requests?sorting=newest" `
  -H "Accept: application/json"
```

Se non arriva niente: controllate di aver usato `https://` e l'indirizzo giusto. Lo stesso fatto si può leggere anche senza avviso: `GET /api/v1/events/recent?event=appointment.created` (§8.6).

6e. Mi faccio mandare un avviso che nasce dal telefono. Chiamate perse, telefonate finite e richieste raccolte da Giulia, sull'azienda di prova, non nascono mai: non ha un numero di telefono. Ve le facciamo arrivare a comando, con dati finti ma nella forma esatta di quelle vere. Al posto di `IL-VOSTRO-HOOK-ID` mettete l'`id` del passo 6b:

```
curl -s -X POST $API/hooks/test \
  -H "x-api-key: $KEY" -H "Content-Type: application/json" \
  -d '{"event": "call.missed", "hook_id": "IL-VOSTRO-HOOK-ID"}'
```

```
$body = '{"event": "call.missed", "hook_id": "IL-VOSTRO-HOOK-ID"}'
$body | curl.exe -s -X POST "$API/hooks/test" `
  -H "x-api-key: $KEY" -H "Content-Type: application/json" --data-binary '@-'
```

```
{
  "success": true,
  "data": {
    "event": "call.missed",
    "test": true,
    "deliveries": [
      {
        "hook_id": "e763e622-1a6c-4a63-b248-facf0acc2f0c",
        "url": "https://webhook.site/…b3aa",
        "delivery_id": "49937a0c-f657-4ce2-a0be-2313bcf9d954",
        "delivered": true,
        "response_status": 200,
        "duration_ms": 123,
        "error": null
      }
    ],
    "body": {
      "id": "49937a0c-f657-4ce2-a0be-2313bcf9d954",
      "event": "call.missed",
      "created_at": "2026-11-25T21:29:32.855Z",
      "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
      "attempt": 1,
      "test": true,
      "data": {
        "id": "00000000-0000-4000-8000-000000000006",
        "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
        "customer_id": "00000000-0000-4000-8000-000000000001",
        "caller_number": "+393330000001",
        "called_number": "+390100000000",
        "call_timestamp": "2026-11-25T21:27:32.853Z",
        "ring_duration_seconds": 25,
        "disconnect_reason": null,
        "recovery_status": "pending",
        "created_at": "2026-11-25T21:27:32.853Z",
        "customer_name": "Mario Rossi (prova)",
        "customer_phone": "+393330000001",
        "customer_email": "mario.rossi@example.com"
      }
    }
  }
}
```


`delivered: true` e `response_status: 200` dicono cosa ha risposto il vostro server; in `body` c'è il corpo spedito. Sull'avviso arrivato ci sono in più l'intestazione `Swavo-Test: true` e `"test": true` nel corpo: nel vostro codice scartateli in produzione. Funziona per tutti gli eventi (elenco completo al §8.1). Senza `hook_id` va a tutte le vostre iscrizioni a quell'evento; se non ne avete nessuna risponde `404` e ve lo dice. Tutto il resto (registro delle consegne, rimando, svuotare l'azienda di prova) è al §8.7.

Per ricevere anche altri eventi (spostamenti, annullamenti, richieste…) rifate la chiamata del passo 6b con lo stesso indirizzo e un altro `event`: un evento per chiamata, stessa iscrizione, stesso segreto (§8.1).

Fatto. Da qui: tutti i campi dei clienti al §3, le richieste al §4, spostare e annullare appuntamenti al §5.4, gli avvisi e la verifica della firma al §8.

1.8 Domande che vi farete

Domanda	Risposta
Devo registrarmi su Swavo?	No. Serve solo la chiave.
Devo entrare nel portale di Swavo?	No, mai. Tutto passa dall'API.
Posso vedere con i miei occhi i dati che creo?	Sì, rileggendoli dall'API: `GET /api/v1/customers`, `GET /api/v1/requests`, `GET /api/v1/appointments` (§3-§5). Se vi serve vederli anche a schermo, chiedetecelo.
Posso cancellare quello che ho creato?	In gran parte sì: un'iscrizione agli avvisi con `DELETE /api/v1/hooks/:id`; una richiesta con `DELETE /api/v1/tickets/:id`; un appuntamento si annulla con `PATCH /api/v1/appointments/:id` e `{"status": "cancelled"}`; un cliente con `DELETE /api/v1/customers/:id`, **solo se non ha appuntamenti, richieste o telefonate collegate** (altrimenti `409`: lo storico non si perde). Tutte provate. Per ripartire da zero c'è `POST /api/v1/sandbox/reset` (§8.7): svuota l'azienda di prova in un colpo.
Questo ambiente è separato dalla produzione?	È lo **stesso server** di produzione (`portal.swavo.ai`), ma su un'**azienda separata**: ogni chiave vede solo i dati della sua azienda. Quello che fate qui non tocca nessun'altra azienda e non parte nessun WhatsApp (l'azienda di prova non ha un numero).
Ieri una chiamata mi dava errore: come la ritrovo?	`GET /api/v1/logs?status=error`: l'elenco delle vostre chiamate degli ultimi 30 giorni, con l'esito, la durata e il codice e il messaggio dell'errore ricevuto (§8.7). Non contiene mai i corpi; telefoni ed email nella query sono mascherati.
Un avviso non mi è arrivato: come capisco perché?	`GET /api/v1/hooks/deliveries`: per ogni avviso dice se è partito, cosa ha risposto il vostro server, quando ritentiamo e perché è fallito. Con `POST /api/v1/hooks/deliveries/:id/retry` ve lo rimandiamo subito (§8.7).

Domanda	Risposta
Possiamo lavorarci in due con la stessa chiave?	Sì, ma vedete e toccate gli stessi dati: usate numeri di telefono diversi, se no il secondo aggiorna il cliente del primo.
La chiave scade?	Quella di prova sì, dopo 60 giorni; se vi serve ancora la rinnoviamo. Quella di un'azienda vera la decide il titolare quando la crea.
Chi mi dà la chiave dell'azienda vera?	Il titolare dell'azienda, che la crea dal portale (§1.1). Il vostro codice non cambia: cambia solo la chiave. Prima di partire guardate il §9.
Che succede se sbaglio e creo dati sporchi?	Niente: sono dati di prova. Cancellateli (sopra) o svuotate l'azienda con `POST /api/v1/sandbox/reset`.
C'è un limite di chiamate?	1000 l'ora per chiave; oltre si riceve `429` e si aspetta quanto dice `Retry-After`. Per un import iniziale grande lo alziamo.

---

2. Regole comuni

2.1 Forma delle risposte

Sempre la stessa busta:

```
{ "success": true,  "data": … }
{ "success": false, "error": { "code": "VALIDATION_ERROR", "message": "…" } }
```

2.2 Date, ore, fuso orario

Dato	Formato	Fuso
Giorno (`date`, `scheduled_date`, `from`, `to`, `start_date`, `end_date`)	`AAAA-MM-GG` (es. `2026-12-01`). Un giorno inesistente (31 febbraio) è rifiutato con `400`.	**Ora italiana** (Europe/Rome). `from`/`to` sono **compresi** e valgono dalla mezzanotte di Roma.
Ora di un appuntamento (`time`, `start_time`, `end_time`)	`HH:MM` in ingresso (`HH:MM:SS` accettato); in uscita `HH:MM:SS` (negli orari liberi `HH:MM`).	**Ora italiana**, senza fuso: è l'ora sul calendario dell'agenda.
Istanti (`created_at`, `updated_at`, `completed_at`, `started_at`…)	ISO 8601 con fuso, es. `2026-11-23T18:32:12.327206+00:00`	UTC (lo dice il suffisso).

2.3 Telefoni

Si possono mandare come li scrive una persona: `3331234567`, `333 123 4567`, `+39 333…`, `0039…`. Swavo li porta sempre alla forma `+39…` (E.164) e li restituisce così. Un numero italiano senza prefisso diventa `+39…`. Meno di 8 cifre = `400`. **Il telefono è la chiave del cliente**: in un'azienda non possono esistere due clienti con lo stesso numero (`409`).

2.4 Elenchi

Due famiglie, entrambe provate:

Porta	Parametri	Default	Massimo	Totale
`GET /customers`, `GET /tickets`, `GET /technicians` (elenchi paginati)	`limit`, `offset`, `search`, `sort_by`, `sort_order` (`asc` predefinito, `desc`)	`limit=20`, `offset=0`	`limit=100` (oltre: ridotto a 100)	`data.total` (numero di righe che rispondono ai filtri)
`GET /requests`, `GET /appointments`, `GET /calls`, `GET /appointments/slots`, `GET /customers/find` (ricerche)	filtri della porta + `limit`	`limit=10` (orari liberi: 20)	50 (orari liberi: 100)	nessun totale: `data` è la lista

`sort_by` ammessi — clienti: `name` (predefinito), `phone_number`, `email`, `created_at`, `updated_at`, `contract_expires_at` · richieste (`/tickets`): `created_at` (predefinito), `updated_at`, `priority`, `status`, `completed_at` · operatori: `name` (predefinito), `phone`, `created_at`, `is_active`. **Un valore non ammesso non dà errore: si torna all'ordinamento predefinito.** `search` cerca «contiene» su nome, telefono, email (clienti) o descrizione, nome, telefono (richieste).

2.5 Errori

Codice HTTP	`error.code`	Quando	Esempio vero
400	`VALIDATION_ERROR`	corpo non valido, campo mancante, formato sbagliato, JSON rotto; `message` dice quale campo	§2.6
401	`UNAUTHORIZED`	chiave mancante, inventata, revocata o scaduta	§2.6

Codice HTTP	`error.code`	Quando	Esempio vero
403	`FORBIDDEN`	piano senza API; chiave senza permesso per gli avvisi	§2.6
404	`NOT_FOUND`	id inesistente **o di un'altra azienda** (non diciamo quale dei due)	§2.6
409	`CONFLICT`	telefono già usato da un altro cliente; orario già occupato (operatore o cliente); tetto di appuntamenti futuri del cliente	§2.6 e §5
429	`RATE_LIMITED`	troppe chiamate nell'ora	§2.6
500	`INTERNAL_ERROR`	guasto nostro; il messaggio è sempre generico. Ritentate più tardi.	—
502	`WHATSAPP_ERROR`	solo `send-template`: Meta ha rifiutato l'invio; `message` riporta il motivo di Meta	—

2.6 Esempi veri di errore

```
POST /api/v1/customers
x-api-key: sk_live_…
Content-Type: application/json

{
  "phone_number": "+393330231209"
}
```

Risposta **400**:

```
{
  "success": false,
  "error": {
    "code": "VALIDATION_ERROR",
    "message": "name: Invalid input"
  }
}
```

```
GET /api/v1/me
```

Risposta **401**:

```
{
  "success": false,
  "error": {
    "code": "UNAUTHORIZED",
    "message": "Authentication required: send the x-api-key header"
  }
}
```

```
GET /api/v1/me
x-api-key: sk_live_…(una chiave che non esiste)
```

Risposta **401**:

```
{
  "success": false,
  "error": {
    "code": "UNAUTHORIZED",
    "message": "Invalid API key"
  }
}
```

```
GET /api/v1/me
x-api-key: sk_live_…
```

Risposta **403**:

```
{
  "success": false,
  "error": {
    "code": "FORBIDDEN",
    "message": "API e webhook sono disponibili dal piano Pro"
  }
}
```

```
POST /api/v1/hooks
x-api-key: sk_live_…
Content-Type: application/json

{
  "url": "https://webhook.site/…",
  "event": "customer.created"
}
```

Risposta **403**:

```
{
  "success": false,
  "error": {
    "code": "FORBIDDEN",
    "message": "Questa chiave non e abilitata alle app di automazione: crea una chiave nuova spuntando «Userò questa chiave con Zapier o Make»"
  }
}
```

```
GET /api/v1/customers/aaaaaaaa-bbbb-4ccc-8ddd-eeeeeeeeeeee
x-api-key: sk_live_…
```

Risposta **404**:

```
{
  "success": false,
  "error": {
    "code": "NOT_FOUND",
    "message": "Customer not found"
  }
}
```

```
POST /api/v1/customers
x-api-key: sk_live_…
Content-Type: application/json

{
  "name": "Doppione",
  "phone_number": "+393330231201"
}
```

Risposta **409**:

```
{
  "success": false,
  "error": {
    "code": "CONFLICT",
    "message": "Esiste gia un cliente con questo telefono: usa /api/v1/customers/upsert per aggiornarlo"
  }
}
```

```
GET /api/v1/me
x-api-key: sk_live_…
```

Risposta **429**, con le intestazioni `Retry-After` (secondi da aspettare) e `X-RateLimit-Reset` (quando riparte il conteggio):

```
{
  "success": false,
  "error": {
    "code": "RATE_LIMITED",
    "message": "API rate limit exceeded"
  }
}
```

---

3. Clienti (anagrafica)

3.1 `POST /api/v1/customers/upsert` — crea o aggiorna dal telefono ⭐ consigliata per l'anagrafica

Campo	Tipo	Obbl.	Note
`phone`	string	**sì**	chiave del cliente, normalizzato `+39…`
`name`	string ≤200	no	
`email`	string (email)	no	salvata in minuscolo; `""` = non cambiare

Campo	Tipo	Obbl.	Note
`address`	string ≤500	no	indirizzo in una riga
`notes`	string ≤4000	no	**si aggiunge in fondo** alle note esistenti (mai sovrascritte; la stessa nota due volte non si ripete)
`marketing_consent`	boolean	no	

Comportamento: se il telefono esiste si aggiornano **solo i campi mandati** (un campo assente o vuoto resta com'è) e risponde `200` con `created: false`; se non esiste lo crea e risponde `201` con `created: true`. **Mandare due volte lo stesso cliente non crea doppioni**: è la porta da usare per l'allineamento periodico dall'anagrafica del vostro gestionale. Il vostro codice cliente si può tenere nelle note (es. «Codice gestionale C-1003»): oggi non c'è un campo dedicato.

```
POST /api/v1/customers/upsert
x-api-key: sk_live_…
Content-Type: application/json

{
  "phone": "3330231203",
  "name": "Laura Verdi",
  "address": "Corso Garibaldi 10, Imperia",
  "notes": "Codice gestionale C-1004"
}
```

Risposta **201**:

```
{
  "success": true,
  "data": {
    "id": "1aa9283d-0270-4959-bbdd-a2e28cc9e3ee",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "name": "Laura Verdi",
    "phone_number": "+393330231203",
    "email": null,
    "address": "Corso Garibaldi 10, Imperia",
    "notes": "Codice gestionale C-1004",
    "marketing_consent": false,
    "source": "api",
    "created_at": "2026-11-25T21:29:39.290727+00:00",
    "updated_at": "2026-11-25T21:29:39.290727+00:00",
    "created": true
  }
}
```

```
POST /api/v1/customers/upsert
x-api-key: sk_live_…
Content-Type: application/json

{
  "phone": "3330231203",
  "name": "Laura Verdi",
  "address": "Corso Garibaldi 10, Imperia",
  "notes": "Codice gestionale C-1004"
}
```

Risposta **200**:

```
{
  "success": true,
  "data": {
    "id": "1aa9283d-0270-4959-bbdd-a2e28cc9e3ee",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "name": "Laura Verdi",
    "phone_number": "+393330231203",
    "email": null,
    "address": "Corso Garibaldi 10, Imperia",
    "notes": "Codice gestionale C-1004",
    "marketing_consent": false,
    "source": "api",
    "created_at": "2026-11-25T21:29:39.290727+00:00",
    "updated_at": "2026-11-25T21:29:39.886+00:00",
    "created": false
  }
}
```

```
POST /api/v1/customers/upsert
x-api-key: sk_live_…
Content-Type: application/json

{
  "phone": "+39 333 023 1203",
  "email": "laura.verdi@example.com"
}
```

Risposta **200**:

```
{
  "success": true,
  "data": {
    "id": "1aa9283d-0270-4959-bbdd-a2e28cc9e3ee",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "name": "Laura Verdi",
    "phone_number": "+393330231203",
    "email": "laura.verdi@example.com",
    "address": "Corso Garibaldi 10, Imperia",
    "notes": "Codice gestionale C-1004",
    "marketing_consent": false,
    "source": "api",
    "created_at": "2026-11-25T21:29:39.290727+00:00",
    "updated_at": "2026-11-25T21:29:40.464+00:00",
    "created": false
  }
}
```

3.2 `GET /api/v1/customers/find?phone=…` (oppure `?email=…`)

Ricerca **esatta** (non «contiene») sul telefono normalizzato, o sull'email senza distinzione di maiuscole. Risponde sempre una lista (vuota se nessuno), al massimo 5, dalla più recente.

```
GET /api/v1/customers/find?phone=0039%20333%20023%201201
x-api-key: sk_live_…
```

Risposta **200**:

```
{
  "success": true,
  "data": [
    {
      "id": "6994cc2d-f230-4e7a-b9c2-06cbf5b46fc4",
      "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
      "name": "Paolo Neri",
      "phone_number": "+393330231201",
      "email": "paolo.neri@example.com",
      "address": "Via Roma 1, 18038 Sanremo IM",
      "notes": "Cliente creato dall'API",
      "marketing_consent": false,
      "source": "manual",
      "created_at": "2026-11-25T21:29:37.357725+00:00",
      "updated_at": "2026-11-25T21:29:37.357725+00:00"
    }
  ]
}
```

```
GET /api/v1/customers/find?phone=%2B393339999999
x-api-key: sk_live_…
```

Risposta **200**:

```
{
  "success": true,
  "data": []
}
```

3.3 `GET /api/v1/customers` (elenco paginato) · `GET /api/v1/customers/:id` · `POST /api/v1/customers` · `PATCH /api/v1/customers/:id`

`POST /customers` campi: `name` (string, **obbl.**), `phone_number` (string, **obbl.**, normalizzato), `email`, `address`, `notes`, `marketing_consent` (bool), `contract_type` (`none·single·annual·premium`), `contract_expires_at` (`AAAA-MM-GG`), più tre campi tecnici che servono solo al settore caldaie. Se il telefono esiste già: `409` (usate l'upsert). `PATCH` accetta gli stessi campi, tutti facoltativi.

```
GET /api/v1/customers?limit=2&offset=0&sort_by=created_at&sort_order=desc
x-api-key: sk_live_…
```

Risposta **200**:

```
{
  "success": true,
  "data": {
    "data": [
      {
        "id": "1aa9283d-0270-4959-bbdd-a2e28cc9e3ee",
        "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
        "name": "Laura Verdi",
        "phone_number": "+393330231203",
        "email": "laura.verdi@example.com",
        "address": "Corso Garibaldi 10, Imperia",
        "contract_type": null,
        "contract_expires_at": null,
        "notes": "Codice gestionale C-1004",
        "total_interventions": 0,
        "last_intervention_date": null,
        "created_at": "2026-11-25T21:29:39.290727+00:00",
        "updated_at": "2026-11-25T21:29:40.464+00:00"
      },
      {
        "id": "be9e808f-0952-43db-a3b5-41284ac3028c",
        "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
        "name": "Anna Bianchi",
        "phone_number": "+393330231202",
        "email": null,
        "address": null,
        "contract_type": null,
        "contract_expires_at": null,
        "notes": null,
        "total_interventions": 0,
        "last_intervention_date": null,
        "created_at": "2026-11-25T21:29:37.718486+00:00",
        "updated_at": "2026-11-25T21:29:37.718486+00:00"
      }
    ],
    "total": 4,
    "limit": 2,
    "offset": 0
  }
}
```


---

4. Richieste (interventi raccolti da Giulia)

4.1 Come arrivano al vostro programma

Quando Giulia chiude una telefonata con una richiesta, Swavo crea una **richiesta** e manda gli avvisi `ticket.created` (sempre) e `lead.created` (se è un contatto da richiamare, ad esempio un preventivo). In `details` ci sono **i dati che Giulia ha raccolto al telefono**, con i nomi di campo della scheda che l'azienda ha preparato per Giulia (cambiano da azienda ad azienda e non tutti sono presenti in ogni richiesta). Esempio di un'azienda di assistenza tecnica (dati inventati):

```
"details": {
  "nome_cognome": "Mario Rossi",
  "telefono": "+393330000000",
  "indirizzo": "Via Esempio 8, Cremona",
  "problema": "Caldaia in blocco, niente acqua calda",
  "urgenza": "entro domani",
  "modello": "a condensazione, 24 kW"
}
```

Le chiavi possono crescere se l'azienda aggiunge domande alla scheda: **leggetele come coppie chiave → testo**, non come campi fissi. Senza dati raccolti, `details` è `{}` (mai `null`).

4.2 Valori ammessi

Campo	Valori
`status`	`new` (nuova) · `assigned` (assegnata) · `in_progress` (in lavorazione) · `pending_parts` (in attesa materiale) · `completed` (chiusa) · `cancelled` (annullata)
`priority`	`critical` · `high` · `normal` · `low`
`issue_type` (in uscita)	quello che ha scelto Giulia o chi l'ha creata: `service`, `repair`, `maintenance`, `quote` (preventivo), `callback` (richiamata), `info`, `complaint`, `lead`, `escalation`… Trattatelo come testo.
`type` (in ingresso su POST)	`service` (predefinito) · `repair` · `maintenance` · `quote` · `callback` · `info` · `complaint`
`source`	`phone_ai` (Giulia al telefono) · `whatsapp` · `portal` · `api` · `manual`
`lead_status`	`null` se non è un contatto da richiamare; altrimenti la fase (es. `new`, `lead`)

4.3 `POST /api/v1/requests` · `GET /api/v1/requests` · `PATCH /api/v1/requests/:id`

POST — campi: `phone` (**obbl.**; il cliente si trova o si crea), `description` (**obbl.**, ≤4000), `name`, `email`, `address` (≤500), `type` (sopra), `priority` (predefinito `normal`), `is_emergency` (bool; se `true` la priorità diventa `critical`), `lead` (bool: è anche un contatto da richiamare). La risposta ha `customer_created`. I dettagli (misure, modello, urgenza…) vanno nella `description`.

GET — filtri: `phone`, `status`, `limit` (10, max 50). Dalla più recente.

PATCH — campi (almeno uno): `status`, `priority`, `resolution_notes` (≤4000). Passando a `completed` Swavo mette da solo `completed_at`. Ogni cambio di stato fa partire `ticket.status_changed`.

```
POST /api/v1/requests
x-api-key: sk_live_…
Content-Type: application/json

{
  "phone": "+393330231204",
  "name": "Bar Centrale",
  "address": "Via del Porto 5, 18100 Imperia",
  "type": "service",
  "description": "Il frigorifero del bancone non raffredda. Chiede un tecnico entro la settimana.",
  "priority": "high",
  "is_emergency": false
}
```

Risposta **201**:


```
{
  "success": true,
  "data": {
    "id": "eb556c07-ee8d-49fc-944a-1be0c991ed36",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "cd382d98-de8b-4692-83d6-3c0f241330c0",
    "issue_type": "service",
    "description": "Il frigorifero del bancone non raffredda. Chiede un tecnico entro la settimana.",
    "priority": "high",
    "status": "new",
    "assigned_technician_id": null,
    "customer_name": "Bar Centrale",
    "customer_phone": "+393330231204",
    "customer_address": "Via del Porto 5, 18100 Imperia",
    "is_emergency": false,
    "source": "api",
    "lead_status": "lead",
    "resolution_notes": null,
    "created_at": "2026-11-25T21:29:42.145447+00:00",
    "updated_at": "2026-11-25T21:29:42.145447+00:00",
    "completed_at": null,
    "details": {},
    "customer_created": true
  }
}
```


```
PATCH /api/v1/requests/eb556c07-ee8d-49fc-944a-1be0c991ed36
x-api-key: sk_live_…
Content-Type: application/json

{
  "status": "completed",
  "resolution_notes": "Intervento eseguito, guarnizione sostituita."
}
```

Risposta **200**:

```
{
  "success": true,
  "data": {
    "id": "eb556c07-ee8d-49fc-944a-1be0c991ed36",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "cd382d98-de8b-4692-83d6-3c0f241330c0",
    "issue_type": "service",
    "description": "Il frigorifero del bancone non raffredda. Chiede un tecnico entro la settimana.",
    "priority": "high",
    "status": "completed",
    "assigned_technician_id": null,
    "customer_name": "Bar Centrale",
    "customer_phone": "+393330231204",
    "customer_address": "Via del Porto 5, 18100 Imperia",
    "is_emergency": false,
    "source": "api",
    "lead_status": "lead",
    "resolution_notes": "Intervento eseguito, guarnizione sostituita.",
    "created_at": "2026-11-25T21:29:42.145447+00:00",
    "updated_at": "2026-11-25T21:29:42.536+00:00",
    "completed_at": "2026-11-25T21:29:42.536+00:00",
    "details": {}
  }
}
```


```
GET /api/v1/requests?status=chiusa
x-api-key: sk_live_…
```

Risposta **400**:


```
{
  "success": false,
  "error": {
    "code": "VALIDATION_ERROR",
    "message": "Stato sconosciuto (status). Validi: new, assigned, in_progress, pending_parts, completed, cancelled"
  }
}
```


4.4 `/api/v1/requests` o `/api/v1/tickets`? → `/requests`

Sono la stessa tabella. `/requests` è quella pensata per un gestionale: si lavora col telefono del cliente (niente id da cercare prima), accetta tutti i tipi di richiesta (`quote`, `callback`…), restituisce `details` e `lead_status`. `/tickets` è la porta generica per id, con tipi limitati al settore caldaie (`maintenance`, `first_ignition`, `repair`, `service`: un preventivo verrebbe rifiutato) e senza `details`.

---

5. Agenda

5.1 `GET /api/v1/technicians` — operatori

Elenco paginato. Campi: `id`, `name` (**un campo solo**, non nome e cognome separati), `phone`, `email`, `is_active`, `color`, `notes`, `specialization`, `created_at`. Filtro `is_active=true|false`.

```
GET /api/v1/technicians?limit=10
x-api-key: sk_live_…
```

Risposta **200**:

```
{
  "success": true,
  "data": {
    "data": [
      {
        "id": "2c31dc6b-df38-430e-ae8a-94af38f5c2bc",
        "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
        "name": "Laura",
        "phone": null,
        "email": null,
        "is_active": true,
        "color": null,
        "notes": null,
        "created_at": "2026-11-25T20:59:16.411654+00:00",
        "specialization": null
      },
      {
        "id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
        "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
        "name": "Titolare",
        "phone": null,
        "email": null,
        "is_active": true,
        "color": null,
        "notes": "Agenda di base creata dal settore: rinomina l'operatore e sistema orari e giorni.",
        "created_at": "2026-11-25T20:58:52.155119+00:00",
        "specialization": null
      }
    ],
    "total": 2,
    "limit": 10,
    "offset": 0
  }
}
```


5.2 `GET /api/v1/appointments/slots` — orari liberi

Le stesse regole che usa Giulia: orari di lavoro di ogni operatore, ferie e assenze, appuntamenti già presi, calendari esterni collegati. Parametri: `from`, `to` (`AAAA-MM-GG`, compresi; predefinito oggi → 7 giorni dopo; **massimo 31 giorni**), `duration_minutes` (5-720, predefinito 60), `technician_id` (facoltativo), `limit` (20, max 100). Gli orari di oggi già passati non escono. Ordine: dal più vicino.

```
GET /api/v1/appointments/slots?from=2026-12-03&to=2026-12-03&duration_minutes=60&technician_id=f8d5b614-1628-4a67-86bd-7faa76f71ee2&limit=5
x-api-key: sk_live_…
```

Risposta **200**:


```
{
  "success": true,
  "data": [
    {
      "date": "2026-12-03",
      "time": "08:00",
      "end_time": "09:00",
      "duration_minutes": 60,
      "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
      "technician_name": "Titolare"
    },
    {
      "date": "2026-12-03",
      "time": "09:00",
      "end_time": "10:00",
      "duration_minutes": 60,
      "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
      "technician_name": "Titolare"
    },
    "… altri 3 elementi"
  ]
}
```


5.3 `POST /api/v1/appointments` — crea

Campo	Tipo	Obbl.	Note
`phone`	string	**sì**	il cliente si trova o si crea
`date`	`AAAA-MM-GG`	**sì**	ora italiana
`time`	`HH:MM`	**sì**	ora di inizio
`duration_minutes`	intero 5-720	no	predefinito 60; la fine la calcola Swavo
`technician_id`	uuid	no	operatore della stessa azienda (altrimenti `400`)
`name`, `email`	string	no	per il cliente

Campo	Tipo	Obbl.	Note
`title`	string ≤200	no	es. «Riparazione frigorifero»
`notes`	string ≤4000	no	

Nasce con `status: "scheduled"`. Risposta: l'appuntamento + `customer` (scheda cliente) + `customer_created`. **Salvate il nostro `id`**: serve per spostarlo o annullarlo.

```
POST /api/v1/appointments
x-api-key: sk_live_…
Content-Type: application/json

{
  "phone": "+393330231204",
  "name": "Bar Centrale",
  "date": "2026-12-03",
  "time": "08:00",
  "duration_minutes": 60,
  "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
  "title": "Riparazione frigorifero",
  "notes": "Portare la guarnizione"
}
```

Risposta **201**:


```
{
  "success": true,
  "data": {
    "id": "bd0619b7-123d-4a7a-b01e-b12ea997d5eb",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "cd382d98-de8b-4692-83d6-3c0f241330c0",
    "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
    "title": "Riparazione frigorifero",
    "notes": "Portare la guarnizione",
    "scheduled_date": "2026-12-03",
    "start_time": "08:00:00",
    "end_time": "09:00:00",
    "estimated_duration_minutes": 60,
    "status": "scheduled",
    "source": "manual",
    "created_at": "2026-11-25T21:29:45.410128+00:00",
    "customer": {
      "id": "cd382d98-de8b-4692-83d6-3c0f241330c0",
      "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
      "name": "Bar Centrale",
      "phone_number": "+393330231204",
      "email": null,
      "address": "Via del Porto 5, 18100 Imperia",
      "notes": null,
      "marketing_consent": false,
      "source": "api",
      "created_at": "2026-11-25T21:29:41.935467+00:00",
      "updated_at": "2026-11-25T21:29:45.137+00:00"
    },
    "customer_created": false
  }
}
```


I conflitti (`409`) — due vincoli del nostro database, provati:

- `unique_technician_slot`: lo **stesso operatore** ha già un appuntamento (non annullato) che **inizia alla stessa ora** dello stesso giorno;
- `unique_customer_slot`: lo **stesso cliente** ha già un appuntamento che inizia alla stessa ora;
- **tetto di appuntamenti futuri per cliente**: dipende dal settore e dall'azienda; qui **3** in stato `scheduled` (quelli `confirmed` non contano), il quarto è rifiutato. Nella ristorazione non c'è.

```
POST /api/v1/appointments
x-api-key: sk_live_…
Content-Type: application/json

{
  "phone": "+393330231206",
  "name": "Altro cliente",
  "date": "2026-12-03",
  "time": "08:00",
  "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2"
}
```

Risposta **409**:

```
{
  "success": false,
  "error": {
    "code": "CONFLICT",
    "message": "Orario gia occupato per questo operatore (unique_technician_slot)"
  }
}
```

```
POST /api/v1/appointments
x-api-key: sk_live_…
Content-Type: application/json

{
  "phone": "+393330231204",
  "date": "2026-12-03",
  "time": "08:00",
  "technician_id": "2c31dc6b-df38-430e-ae8a-94af38f5c2bc"
}
```

Risposta **409**:

```
{
  "success": false,
  "error": {
    "code": "CONFLICT",
    "message": "Il cliente ha gia un appuntamento che inizia a quell ora (unique_customer_slot)"
  }
}
```

```
POST /api/v1/appointments
x-api-key: sk_live_…
Content-Type: application/json

{
  "phone": "+393330231208",
  "date": "2026-12-03",
  "time": "13:00",
  "technician_id": "2c31dc6b-df38-430e-ae8a-94af38f5c2bc"
}
```

Risposta **409**:

```
{
  "success": false,
  "error": {
    "code": "CONFLICT",
    "message": "LIMITE_APPUNTAMENTI: cliente ha già 3 appuntamenti su 3 massimi consentiti"
  }
}
```

⚠️ **Da sapere**: la porta **non** rifiuta un appuntamento che si **sovrappone** a un altro con un inizio diverso (provato: 08:00-09:00 e 08:30-09:30 sullo stesso operatore, entrambi accettati). È voluto: se le agende le governa il vostro programma, una sovrapposizione che esiste lì deve poter esistere anche nella copia. Gli orari liberi (§5.2) invece la tengono in conto.

5.4 `PATCH /api/v1/appointments/:id` — sposta, conferma, annulla

Campi (almeno uno): `date`, `time`, `duration_minutes`, `status`, `notes`. Stati ammessi qui: `scheduled` · `confirmed` · `completed` · `cancelled` · `no_show`. La risposta porta anche `customer_name`, `customer_phone`, `customer_email`. Cliente e operatore non si cambiano da qui. Gli esempi: spostare (con `appointment.rescheduled`), annullare (con `appointment.status_changed` e `appointment.cancelled`) e spostare sull'ora in cui lo stesso operatore è già occupato (`409`). Un appuntamento annullato non occupa più il suo orario. Gli avvisi di spostamento e annullamento arrivano solo se il vostro indirizzo è iscritto a quegli eventi (§8.1).

```
PATCH /api/v1/appointments/bd0619b7-123d-4a7a-b01e-b12ea997d5eb
x-api-key: sk_live_…
Content-Type: application/json

{
  "time": "16:00",
  "duration_minutes": 90
}
```

Risposta **200**:

```
{
  "success": true,
  "data": {
    "id": "bd0619b7-123d-4a7a-b01e-b12ea997d5eb",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "cd382d98-de8b-4692-83d6-3c0f241330c0",
    "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
    "title": "Riparazione frigorifero",
    "notes": "Portare la guarnizione",
    "scheduled_date": "2026-12-03",
    "start_time": "16:00:00",
    "end_time": "17:30:00",
    "estimated_duration_minutes": 90,
    "status": "scheduled",
    "source": "manual",
    "created_at": "2026-11-25T21:29:45.410128+00:00",
    "updated_at": "2026-11-25T21:29:50.611+00:00",
    "customer_name": "Bar Centrale",
    "customer_phone": "+393330231204",
    "customer_email": null
  }
}
```


```
PATCH /api/v1/appointments/401c236c-1140-46a1-aa64-3f38fc649b0d
x-api-key: sk_live_…
Content-Type: application/json

{
  "status": "cancelled"
}
```

Risposta **200**:


```
{
  "success": true,
  "data": {
    "id": "401c236c-1140-46a1-aa64-3f38fc649b0d",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "e7ee87c3-c5b6-4df0-8554-925a7772ab10",
    "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
    "title": null,
    "notes": null,
    "scheduled_date": "2026-12-03",
    "start_time": "08:30:00",
    "end_time": "09:30:00",
    "estimated_duration_minutes": 60,
    "status": "cancelled",
    "source": "manual",
    "created_at": "2026-11-25T21:29:47.349836+00:00",
    "updated_at": "2026-11-25T21:29:51.124+00:00",
    "customer_name": "Sovrapposto",
    "customer_phone": "+393330231207",
    "customer_email": null
  }
}
```


```
PATCH /api/v1/appointments/bd6de075-6bfa-4a88-bd29-4aadc8af1c0b
x-api-key: sk_live_…
Content-Type: application/json

{
  "time": "11:00"
}
```

Risposta **409**:


```
{
  "success": false,
  "error": {
    "code": "CONFLICT",
    "message": "Orario gia occupato per questo operatore (unique_technician_slot)"
  }
}
```


5.5 `GET /api/v1/appointments` — cerca

Filtri: `phone`, `status`, `from`, `to`, `limit` (10, max 50). Dal più vicino nel tempo. Solo appuntamenti con un cliente.

Stati di un appuntamento: `scheduled` (preso) · `confirmed` (confermato, anche dal cliente col pulsante WhatsApp) · `in_progress` (intervento in corso) · `completed` · `cancelled` · `no_show` (il cliente non si è presentato).

5.6 `POST /api/v1/technicians/:id/absences` — assenza di un operatore

Campi: `start_date` (**obbl.**), `end_date` (predefinito = `start_date`, al massimo un anno), `type` (`vacation` predefinito · `sick_leave` · `personal` · `training` · `other`), `title`. Giorni interi. Effetto provato: quel giorno l'operatore sparisce dagli orari liberi.

```
POST /api/v1/technicians/f8d5b614-1628-4a67-86bd-7faa76f71ee2/absences
x-api-key: sk_live_…
Content-Type: application/json

{
  "start_date": "2026-12-04",
  "type": "sick_leave",
  "title": "Malattia"
}
```

Risposta **201**:

```
{
  "success": true,
  "data": {
    "id": "d00943fc-b0a2-46c1-b562-b472c609182b",
    "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
    "block_type": "sick_leave",
    "title": "Malattia",
    "start_date": "2026-12-04",
    "end_date": "2026-12-04",
    "created_at": "2026-11-25T21:29:52.060211+00:00",
    "technician_name": "Titolare"
  }
}
```

```
GET /api/v1/appointments/slots?from=2026-12-04&to=2026-12-04&technician_id=f8d5b614-1628-4a67-86bd-7faa76f71ee2&limit=3
x-api-key: sk_live_…
```

Risposta **200**:

```
{
  "success": true,
  "data": []
}
```

---

6. Promemoria WhatsApp

Per i promemoria non dovete chiamare WhatsApp. Swavo ha già il messaggio del giorno prima, con due modalità che l'azienda sceglie nelle sue impostazioni su Swavo (non serve niente da parte vostra): **promemoria** semplice, oppure **richiesta di conferma** con i pulsanti «Confermo» / «Devo spostarlo». Parte per **tutti gli appuntamenti `scheduled` o `confirmed` del giorno dopo**, qualunque sia la loro origine: quindi anche per quelli che il vostro programma ha copiato in Swavo con `POST /appointments`. «Confermo» porta l'appuntamento a `confirmed` e voi lo ricevete con l'avviso `appointment.status_changed`; «Devo spostarlo» apre la conversazione con Giulia.

Per un invio libero esistono comunque:

- `GET /api/v1/whatsapp/templates` — i modelli approvati da Meta dell'azienda (`name`, `language`, `body_text`, `params_count`).
- `POST /api/v1/whatsapp/send-template` — `{ "to", "template", "language" (predefinito "it"), "params": [ … ] }`. Risposta `{ "sent": true, "message_id", "template", "to" }`. Errori: `404` modello inesistente o non approvato; `400` dati mancanti; `502` rifiuto di Meta (motivo nel messaggio).

L'azienda di prova non ha un numero WhatsApp: lì potete provare solo i casi d'errore.

```
POST /api/v1/whatsapp/send-template
x-api-key: sk_live_…
Content-Type: application/json

{
  "to": "+393330231204",
  "template": "promemoria_inesistente",
  "params": [
    "Mario",
    "3 dicembre",
    "09:00"
  ]
}
```

Risposta **404**:

```
{
  "success": false,
  "error": {
    "code": "NOT_FOUND",
    "message": "Template \"promemoria_inesistente\" (it) not found for this company"
  }
}
```

---

7. Telefonate

`GET /api/v1/calls` — filtri `phone` (chi ha chiamato), `from`, `to` (giorni di Roma, compresi), `outcome`, `limit` (10, max 50). Dalla più recente. Porta **il riassunto di Giulia**, mai la trascrizione intera. Le telefonate di prova non escono. Sull'azienda di prova l'elenco è vuoto (non ha un numero): la forma di ogni telefonata è quella di `data` nell'avviso `call.ended` (§8.5). `outcome`: `appointment_booked` · `resolved_by_phone` · `info_only` · `transferred` · `cancelled_existing` · `abandoned` · `missed`.

```
GET /api/v1/calls?limit=2
x-api-key: sk_live_…
```

Risposta **200**:

```
{
  "success": true,
  "data": []
}
```

---

8. Avvisi in uscita (webhook)

8.1 Iscrizione da programma, senza passare da noi

`POST /api/v1/hooks` con `{ "url": "https://…", "event": "<evento>" }`. **Un evento per chiamata**: ogni chiamata aggiunge un evento a quell'indirizzo (stesso indirizzo = stessa iscrizione, stesso segreto) e risponde `201` anche se l'indirizzo era già iscritto. Gli eventi: `customer.created`, `lead.created`, `ticket.created`, `ticket.status_changed`, `appointment.created`, `appointment.rescheduled`, `appointment.status_changed`, `appointment.cancelled`, `appointment.rated`, `call.ended`, `call.missed`, `message.received`, `order.created`, `order.status_changed` (gli ultimi due solo per i ristoranti). **La risposta contiene il `secret` (`whsec_…`) con cui firmiamo**: conservatelo. `GET /api/v1/hooks` elenca le iscrizioni (indirizzo mascherato; esempio sotto: questa porta, nata per le app Zapier e Make, ha i nomi dei campi in italiano, e `descrizione` è il testo scritto alla prima iscrizione, che non cambia quando aggiungete eventi). `DELETE /api/v1/hooks/:id?event=<evento>` toglie un evento; senza `event` toglie l'iscrizione. Rifiutati: `http://`, indirizzi di rete privata, eventi sconosciuti.

```
POST /api/v1/hooks
x-api-key: sk_live_…
Content-Type: application/json

{
  "url": "https://webhook.site/e1019b6d-d28f-405a-b160-97f5ca8ab3aa/secondo",
  "event": "appointment.created"
}
```

Risposta **201**:

```
{
  "success": true,
  "data": {
    "id": "dc13f6d2-9e2f-4a5d-b877-cb245f684e0e",
    "event": "appointment.created",
    "app": "altro",
    "secret": "whsec_b4730e…"
  }
}
```

```
POST /api/v1/hooks
x-api-key: sk_live_…
Content-Type: application/json

{
  "url": "http://webhook.site/e1019b6d-d28f-405a-b160-97f5ca8ab3aa",
  "event": "appointment.created"
}
```

Risposta **400**:

```
{
  "success": false,
  "error": {
    "code": "VALIDATION_ERROR",
    "message": "serve https:// — su http i dati viaggerebbero in chiaro"
  }
}
```

```
DELETE /api/v1/hooks/dc13f6d2-9e2f-4a5d-b877-cb245f684e0e
x-api-key: sk_live_…
```

Risposta **200**:

```
{
  "success": true,
  "data": {
    "id": "dc13f6d2-9e2f-4a5d-b877-cb245f684e0e",
    "deleted": true
  }
}
```

```
GET /api/v1/hooks
x-api-key: sk_live_…
```

Risposta **200**:

```
{
  "success": true,
  "data": [
    {
      "id": "e763e622-1a6c-4a63-b248-facf0acc2f0c",
      "url": "https://webhook.site/…b3aa",
      "descrizione": "App · appointment.created (iscritta dall’app)",
      "eventi": [
        "appointment.created",
        "customer.created",
        "lead.created",
        "ticket.created",
        "ticket.status_changed",
        "appointment.rescheduled",
        "appointment.status_changed",
        "appointment.cancelled",
        "call.ended",
        "call.missed"
      ],
      "is_active": true,
      "consegne_ok": 13,
      "consegne_ko": 0,
      "last_success_at": "2026-11-25T21:29:53.131+00:00",
      "last_error": null,
      "created_at": "2026-11-25T21:29:31.660703+00:00"
    }
  ]
}
```

8.2 Cosa riceve il vostro server

```
POST <vostro indirizzo>
Content-Type: application/json
User-Agent: Swavo-Webhooks/1.0
Swavo-Event: ticket.created
Swavo-Delivery: <id della consegna, uguale nei ritentativi>
Swavo-Signature: t=<secondi unix>,v1=<hmac esadecimale>
Idempotency-Key: <evento>:<id riga>:<id iscrizione> (uguale nei ritentativi)

{ "id", "event", "created_at", "company_id", "attempt", "data": { … } }
```

`Swavo-Signature` è la firma (§8.3). `Idempotency-Key` è una chiave che resta uguale se lo stesso avviso arriva due volte: serve a non registrarlo due volte. `data` è lo stesso oggetto che restituisce l'API (stessi nomi di campo). Gli eventi di cambio portano anche `changed_to` (lo stato al momento del cambio) e lo stato di adesso in `status`.

8.3 Firma

`v1 = HMAC-SHA256(segreto, "<t>.<corpo grezzo>")` in esadecimale. Il segreto si usa così com'è, `whsec_` compreso (niente da togliere o decodificare). Verificate sul **corpo così com'è arrivato** e rifiutate se `|ora − t| > 300` secondi. Provato con uno script Python scritto a parte (non il nostro codice) sugli avvisi ricevuti nei nostri collaudi: firma giusta su tutti; corpo alterato → rifiutato; segreto sbagliato → rifiutato; stesso avviso verificato dopo più di 5 minuti → rifiutato.

Le stesse quattro funzioni, provate su un avviso vero con lettere accentate nel corpo: firma giusta → vero; corpo alterato di un carattere → falso; segreto sbagliato → falso.

Python

```
import hmac, hashlib, time

# corpo_grezzo: il corpo così come è arrivato (bytes), PRIMA di json.loads
def verifica(corpo_grezzo: bytes, intestazione: str, segreto: str) -> bool:
    p = dict(x.split("=", 1) for x in intestazione.split(","))
    if abs(time.time() - int(p["t"])) > 300:
        return False
    firmato = f'{p["t"]}.'.encode() + corpo_grezzo
    atteso = hmac.new(segreto.encode(), firmato, hashlib.sha256).hexdigest()
    return hmac.compare_digest(atteso, p["v1"])
```

Node.js

```
const crypto = require('crypto');

// corpoGrezzo: il corpo così come è arrivato (stringa o Buffer), PRIMA di JSON.parse
function verifica(corpoGrezzo, intestazione, segreto) {
  const p = Object.fromEntries(intestazione.split(',').map((x) => x.split('=')));
  if (Math.abs(Date.now() / 1000 - Number(p.t)) > 300) return false;
  const atteso = crypto.createHmac('sha256', segreto)
    .update(`${p.t}.`).update(corpoGrezzo).digest('hex');
  return atteso.length === (p.v1 || '').length
    && crypto.timingSafeEqual(Buffer.from(atteso), Buffer.from(p.v1));
}
```

PHP

```
<?php
// $corpo = file_get_contents('php://input');  // il corpo grezzo, prima di json_decode
// $intestazione = $_SERVER['HTTP_SWAVO_SIGNATURE'];
function verifica(string $corpo, string $intestazione, string $segreto): bool {
    parse_str(str_replace(',', '&', $intestazione), $p);
    if (abs(time() - (int)($p['t'] ?? 0)) > 300) return false;
    $atteso = hash_hmac('sha256', $p['t'] . '.' . $corpo, $segreto);
    return hash_equals($atteso, $p['v1'] ?? '');
}
```

C# / .NET (Framework 4.x e .NET 6+)

```
using System;
using System.Security.Cryptography;
using System.Text;

static class FirmaSwavo
{
    // corpo: il corpo grezzo così come è arrivato, PRIMA di deserializzarlo
    public static bool Verifica(string corpo, string intestazione, string segreto)
    {
        string t = null, v1 = null;
        foreach (var parte in intestazione.Split(','))
        {
            var kv = parte.Split(new[] { '=' }, 2);
            if (kv[0] == "t") t = kv[1];
            if (kv[0] == "v1") v1 = kv[1];
        }
        if (t == null || v1 == null) return false;
        if (Math.Abs(DateTimeOffset.UtcNow.ToUnixTimeSeconds() - long.Parse(t)) > 300) return false;
        using (var h = new HMACSHA256(Encoding.UTF8.GetBytes(segreto)))
        {
            var hash = h.ComputeHash(Encoding.UTF8.GetBytes(t + "." + corpo));
            var atteso = BitConverter.ToString(hash).Replace("-", "").ToLowerInvariant();
            if (atteso.Length != v1.Length) return false;
            int diff = 0;
            for (int i = 0; i < atteso.Length; i++) diff |= atteso[i] ^ v1[i];
            return diff == 0;
        }
    }
}
```

8.4 Consegna e ritentativi (provati)

- Rispondete **2xx entro 10 secondi**. Gli avvisi partono in pochi secondi dal fatto.
- **5xx, 408, 429, nessuna risposta** → ritentiamo dopo 5, 10, 20, 40 minuti (massimo 5 tentativi). Provato: 503 → secondo tentativo dopo 5 minuti, terzo programmato dopo altri 10, stessa `Idempotency-Key`, `attempt` che sale.
- Provato anche un **429** del destinatario: ritentato dopo 5 minuti e consegnato.

- **Altri 4xx** → nessun ritentativo (provato con 404 e 400). **410** → l'iscrizione si spegne da sola (provato: `is_active` passa a falso dopo la prima consegna).
- I redirect non si seguono. L'ordine di arrivo non è garantito: ordinate per `data.updated_at`/`created_at`. Lo stesso fatto può arrivare due volte: scartate le `Idempotency-Key` già viste.

8.5 Gli eventi principali — corpi veri ricevuti

Arrivati davvero sull'azienda di prova il 25/11/2026 (firme accorciate). `call.ended` e `call.missed` sono avvisi di prova (`POST /api/v1/hooks/test`, dati finti): l'azienda di prova non ha un numero di telefono. Gli altri eventi (`message.received`, `appointment.rated` e quelli dei ristoranti, `order.created` e `order.status_changed`) sono descritti su https://portal.swavo.ai/sviluppatori; ve li fate arrivare con `POST /api/v1/hooks/test`.

`customer.created`

```
POST <vostro indirizzo>
Swavo-Event: customer.created
Swavo-Delivery: b9137cf6-737b-4583-8aba-e2bb7a61aff4
Swavo-Signature: t=1795642180,v1=ac8deecc…
Idempotency-Key: customer.created:1aa9283d-0270-4959-bbdd-a2e28cc9e3ee:e763e622-1a6c-4a63-b248-facf0acc2f0c

{
  "id": "b9137cf6-737b-4583-8aba-e2bb7a61aff4",
  "event": "customer.created",
  "created_at": "2026-11-25T21:29:40.277Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "data": {
    "id": "1aa9283d-0270-4959-bbdd-a2e28cc9e3ee",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "name": "Laura Verdi",
    "phone_number": "+393330231203",
    "email": null,
    "address": "Corso Garibaldi 10, Imperia",
    "marketing_consent": false,
    "source": "api",
    "notes": "Codice gestionale C-1004",
    "created_at": "2026-11-25T21:29:39.290727+00:00"
  }
}
```

`lead.created`

```
POST <vostro indirizzo>
Swavo-Event: lead.created
Swavo-Delivery: 92dba536-c174-42fa-99c5-935e4da3b774
Swavo-Signature: t=1795642184,v1=96bc2548…
Idempotency-Key: lead.created:5145ed90-37c5-4a9e-983c-fab83fcfd38b:e763e622-1a6c-4a63-b248-facf0acc2f0c

{
  "id": "92dba536-c174-42fa-99c5-935e4da3b774",
  "event": "lead.created",
  "created_at": "2026-11-25T21:29:44.342Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "data": {
    "ticket_id": "5145ed90-37c5-4a9e-983c-fab83fcfd38b",
    "customer_id": "1fa8d549-03f0-49c6-a384-02b91467778b",
    "name": "Elena Gatti",
    "phone": "+393330231251",
    "reason": "Preventivo per la manutenzione annuale di due impianti",
    "channel": "sito",
    "lead_status": "new",
    "details": {},
    "priority": "normal",
    "status": "assigned",
    "source": "api",
    "created_at": "2026-11-25T21:29:43.505854+00:00"
  }
}
```

ticket.created

```
POST <vostro indirizzo>
Swavo-Event: ticket.created
Swavo-Delivery: 82bddb16-9939-40dd-98c7-99cab9de456b
Swavo-Signature: t=1795642184,v1=7b909b8e…
Idempotency-Key: ticket.created:5145ed90-37c5-4a9e-983c-fab83fcfd38b:e763e622-1a6c-4a63-b248-facf0acc2f0c

{
  "id": "82bddb16-9939-40dd-98c7-99cab9de456b",
  "event": "ticket.created",
  "created_at": "2026-11-25T21:29:44.367Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "data": {
    "id": "5145ed90-37c5-4a9e-983c-fab83fcfd38b",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "1fa8d549-03f0-49c6-a384-02b91467778b",
    "issue_type": "lead",
    "description": "Preventivo per la manutenzione annuale di due impianti",
    "priority": "normal",
    "status": "assigned",
    "assigned_technician_id": null,
    "customer_name": "Elena Gatti",
    "customer_phone": "+393330231251",
    "customer_address": null,
    "is_emergency": false,
    "source": "api",
    "lead_status": "new",
    "contacted_at": null,
    "created_at": "2026-11-25T21:29:43.505854+00:00",
    "updated_at": "2026-11-25T21:29:43.935+00:00",
    "completed_at": null,
    "details": {}
  }
}
```

ticket.status_changed

```
POST <vostro indirizzo>
Swavo-Event: ticket.status_changed
Swavo-Delivery: 5d449e40-39fb-4a99-913f-74bc32d9939e
Swavo-Signature: t=1795642185,v1=d3b9627…
Idempotency-Key: ticket.status_changed:5145ed90-37c5-4a9e-983c-fab83fcfd38b:assigned@1795642183995:e763e622-1a6c-4a63-b248-facf0acc2f0c

{
  "id": "5d449e40-39fb-4a99-913f-74bc32d9939e",
  "event": "ticket.status_changed",
  "created_at": "2026-11-25T21:29:45.430Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "data": {
    "id": "5145ed90-37c5-4a9e-983c-fab83fcfd38b",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "1fa8d549-03f0-49c6-a384-02b91467778b",
    "issue_type": "lead",
    "description": "Preventivo per la manutenzione annuale di due impianti",
    "priority": "normal",
    "status": "assigned",
    "assigned_technician_id": null,
    "customer_name": "Elena Gatti",
    "customer_phone": "+393330231251",
    "customer_address": null,
    "is_emergency": false,
    "source": "api",
    "lead_status": "new",
    "contacted_at": null,
    "created_at": "2026-11-25T21:29:43.505854+00:00",
    "updated_at": "2026-11-25T21:29:43.935+00:00",
    "completed_at": null,
    "details": {},
    "changed_to": "assigned"
  }
}
```

`appointment.created`

```
POST <vostro indirizzo>
Swavo-Event: appointment.created
Swavo-Delivery: 573ea17a-91c0-45c2-b858-968edc93c0bc
Swavo-Signature: t=1795642187,v1=448ac63a…
Idempotency-Key: appointment.created:bd0619b7-123d-4a7a-b01e-b12ea997d5eb:e763e622-1a6c-4a63-b248-facf0acc2f0c

{
  "id": "573ea17a-91c0-45c2-b858-968edc93c0bc",
  "event": "appointment.created",
  "created_at": "2026-11-25T21:29:47.194Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "data": {
    "id": "bd0619b7-123d-4a7a-b01e-b12ea997d5eb",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "cd382d98-de8b-4692-83d6-3c0f241330c0",
    "ticket_id": null,
    "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
    "service_id": null,
    "title": "Riparazione frigorifero",
    "reason": null,
    "notes": "Portare la guarnizione",
    "scheduled_date": "2026-12-03",
    "start_time": "08:00:00",
    "end_time": "09:00:00",
    "status": "scheduled",
    "priority": "normal",
    "address": null,
    "location_type": "on_site",
    "estimated_cost": null,
    "currency": "EUR",
    "source": "manual",
    "created_at": "2026-11-25T21:29:45.410128+00:00",
    "updated_at": "2026-11-25T21:29:45.410128+00:00",
    "customer_name": "Bar Centrale",
    "customer_phone": "+393330231204",
    "customer_email": null
  }
}
```

`appointment.rescheduled`

```
POST <vostro indirizzo>
Swavo-Event: appointment.rescheduled
Swavo-Delivery: ed0a811d-a22b-4108-9eff-a7ba0c6c4ca5
Swavo-Signature: t=1795642191,v1=b373194d…
Idempotency-Key: appointment.rescheduled:bd0619b7-123d-4a7a-b01e-b12ea997d5eb:2026-12-03T16:00:00@1795642190654:e763e622-1a6c-4a63-b248-facf0acc2f0c

{
  "id": "ed0a811d-a22b-4108-9eff-a7ba0c6c4ca5",
  "event": "appointment.rescheduled",
  "created_at": "2026-11-25T21:29:51.264Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "data": {
    "id": "bd0619b7-123d-4a7a-b01e-b12ea997d5eb",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "cd382d98-de8b-4692-83d6-3c0f241330c0",
    "ticket_id": null,
    "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
    "service_id": null,
    "title": "Riparazione frigorifero",
    "reason": null,
    "notes": "Portare la guarnizione",
    "scheduled_date": "2026-12-03",
    "start_time": "16:00:00",
    "end_time": "17:30:00",
    "status": "scheduled",
    "priority": "normal",
    "address": null,
    "location_type": "on_site",
    "estimated_cost": null,
    "currency": "EUR",
    "source": "manual",
    "created_at": "2026-11-25T21:29:45.410128+00:00",
    "updated_at": "2026-11-25T21:29:50.611+00:00",
    "customer_name": "Bar Centrale",
    "customer_phone": "+393330231204",
    "customer_email": null
  }
}
```

`appointment.status_changed`

```
POST <vostro indirizzo>
Swavo-Event: appointment.status_changed
Swavo-Delivery: 24fcb7ad-999f-4913-9f08-019846c804c4
Swavo-Signature: t=1795642192,v1=18457144…
Idempotency-Key: appointment.status_changed:401c236c-1140-46a1-aa64-3f38fc649b0d:cancelled@1795642191189:e763e622-1a6c-4a63-b248-facf0acc2f0c

{
  "id": "24fcb7ad-999f-4913-9f08-019846c804c4",
  "event": "appointment.status_changed",
  "created_at": "2026-11-25T21:29:52.932Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "data": {
    "id": "401c236c-1140-46a1-aa64-3f38fc649b0d",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "e7ee87c3-c5b6-4df0-8554-925a7772ab10",
    "ticket_id": null,
    "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
    "service_id": null,
    "title": null,
    "reason": null,
    "notes": null,
    "scheduled_date": "2026-12-03",
    "start_time": "08:30:00",
    "end_time": "09:30:00",
    "status": "cancelled",
    "priority": "normal",
    "address": null,
    "location_type": "on_site",
    "estimated_cost": null,
    "currency": "EUR",
    "source": "manual",
    "created_at": "2026-11-25T21:29:47.349836+00:00",
    "updated_at": "2026-11-25T21:29:51.124+00:00",
    "customer_name": "Sovrapposto",
    "customer_phone": "+393330231207",
    "customer_email": null,
    "changed_to": "cancelled"
  }
}
```

`appointment.cancelled`

```
POST <vostro indirizzo>
Swavo-Event: appointment.cancelled
Swavo-Delivery: 3b989b9d-3a37-4fb6-a006-68f5f90ab4c4
Swavo-Signature: t=1795642193,v1=b2a90fbd…
Idempotency-Key: appointment.cancelled:401c236c-1140-46a1-aa64-3f38fc649b0d:cancelled@1795642191190:e763e622-1a6c-4a63-b248-facf0acc2f0c

{
  "id": "3b989b9d-3a37-4fb6-a006-68f5f90ab4c4",
  "event": "appointment.cancelled",
  "created_at": "2026-11-25T21:29:53.058Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "data": {
    "id": "401c236c-1140-46a1-aa64-3f38fc649b0d",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "e7ee87c3-c5b6-4df0-8554-925a7772ab10",
    "ticket_id": null,
    "technician_id": "f8d5b614-1628-4a67-86bd-7faa76f71ee2",
    "service_id": null,
    "title": null,
    "reason": null,
    "notes": null,
    "scheduled_date": "2026-12-03",
    "start_time": "08:30:00",
    "end_time": "09:30:00",
    "status": "cancelled",
    "priority": "normal",
    "address": null,
    "location_type": "on_site",
    "estimated_cost": null,
    "currency": "EUR",
    "source": "manual",
    "created_at": "2026-11-25T21:29:47.349836+00:00",
    "updated_at": "2026-11-25T21:29:51.124+00:00",
    "customer_name": "Sovrapposto",
    "customer_phone": "+393330231207",
    "customer_email": null
  }
}
```

`call.ended`

```
POST <vostro indirizzo>
Swavo-Event: call.ended
Swavo-Delivery: 152cae0b-aff3-416b-9536-223d4d8f85fa
Swavo-Signature: t=1795642195,v1=a6324e7e…
Idempotency-Key: test:call.ended:152cae0b-aff3-416b-9536-223d4d8f85fa
Swavo-Test: true

{
  "id": "152cae0b-aff3-416b-9536-223d4d8f85fa",
  "event": "call.ended",
  "created_at": "2026-11-25T21:29:55.862Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "test": true,
  "data": {
    "id": "00000000-0000-4000-8000-000000000005",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "00000000-0000-4000-8000-000000000001",
    "ticket_id": "00000000-0000-4000-8000-000000000002",
    "caller_number": "+393330000001",
    "direction": "inbound",
    "started_at": "2026-11-25T21:24:55.861Z",
    "ended_at": "2026-11-25T21:26:55.861Z",
    "duration_seconds": 118,
    "outcome": "appointment_booked",
    "summary": "Il cliente ha blatte in cucina: fissato un sopralluogo per domani alle 9 (avviso di prova).",
    "reason_text": "Disinfestazione blatte",
    "sentiment_final": "positive",
    "follow_up_required": false,
    "created_at": "2026-11-25T21:26:55.861Z",
    "customer_name": "Mario Rossi (prova)",
    "customer_phone": "+393330000001",
    "customer_email": "mario.rossi@example.com"
  }
}
```

`call.missed`

```
POST <vostro indirizzo>
Swavo-Event: call.missed
Swavo-Delivery: 9c68e76a-4f67-44d8-a03c-2eb780d98546
Swavo-Signature: t=1795642196,v1=7c6a885d…
Idempotency-Key: test:call.missed:9c68e76a-4f67-44d8-a03c-2eb780d98546
Swavo-Test: true

{
  "id": "9c68e76a-4f67-44d8-a03c-2eb780d98546",
  "event": "call.missed",
  "created_at": "2026-11-25T21:29:56.191Z",
  "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
  "attempt": 1,
  "test": true,
  "data": {
    "id": "00000000-0000-4000-8000-000000000006",
    "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
    "customer_id": "00000000-0000-4000-8000-000000000001",
    "caller_number": "+393330000001",
    "called_number": "+390100000000",
    "call_timestamp": "2026-11-25T21:27:56.190Z",
    "ring_duration_seconds": 25,
    "disconnect_reason": null,
    "recovery_status": "pending",
    "created_at": "2026-11-25T21:27:56.190Z",
    "customer_name": "Mario Rossi (prova)",
    "customer_phone": "+393330000001",
    "customer_email": "mario.rossi@example.com"
  }
}
```

8.6 Alternativa senza webhook: `GET /api/v1/events/recent?event=<evento>&limit=<n>`

Gli ultimi fatti di un evento, **nella stessa forma di `data`** dell'avviso, il più recente per primo, con un `id` stabile (max 50). Utile per il primo allineamento o se il vostro server è stato giù.

```
GET /api/v1/events/recent?event=ticket.created&limit=1
x-api-key: sk_live_…
```

Risposta **200**:

```
{
  "success": true,
  "data": [
    {
      "id": "5145ed90-37c5-4a9e-983c-fab83fcfd38b",
      "event": "ticket.created",
      "company_id": "b4fc2dae-43ea-449f-ac03-e8d993817bcf",
      "customer_id": "1fa8d549-03f0-49c6-a384-02b91467778b",
      "issue_type": "lead",
      "description": "Preventivo per la manutenzione annuale di due impianti",
      "priority": "normal",
      "status": "assigned",
      "assigned_technician_id": null,
      "customer_name": "Elena Gatti",
      "customer_phone": "+393330231251",
      "customer_address": null,
      "is_emergency": false,
      "source": "api",
      "lead_status": "new",
      "contacted_at": null,
      "created_at": "2026-11-25T21:29:43.505854+00:00",
      "updated_at": "2026-11-25T21:29:43.935+00:00",
      "completed_at": null,
      "details": {}
    }
  ]
}
```


8.7 Strumenti per lo sviluppo

Avviso di prova di qualunque evento — `POST /api/v1/hooks/test` con `{ "event": "<evento>", "hook_id": "<facoltativo>" }`. Spedisce subito l'avviso alle vostre iscrizioni a quell'evento (o solo a `hook_id`, per qualunque evento), firmato come uno vero, con dati finti (id con tanti zeri, «Mario Rossi (prova)») nella forma esatta dell'avviso vero. In più: intestazione `Swavo-Test: true`, `"test": true` nel corpo e una `Idempotency-Key` che comincia con `test:`. Non passa dalla coda e non si ritenta: la risposta dice cosa ha risposto il vostro server e contiene il corpo spedito. Esempio al §1.7, passo 6e. Provato su `call.missed`, `lead.created`, `call.ended`, `appointment.status_changed`, `ticket.created`, `message.received`. Le chiavi dei campioni sono state confrontate con quelle degli avvisi veri: identiche, nello stesso ordine.

Registro delle consegne — `GET /api/v1/hooks/deliveries` (filtri `event`, `status` = `queued`|`sent`|`failed`|`skipped`, `hook_id`, `limit` 20/max 100). Per ogni avviso vero: stato (`sending` mentre parte), codice con cui ha risposto il vostro server, tentativi, prossimo tentativo, motivo in chiaro. `id` = la `Swavo-Delivery` che avete ricevuto. Una consegna compare nel registro appena il fatto succede (in pochi secondi è già `sent` o `failed`). Gli avvisi di prova non compaiono.

```
GET /api/v1/hooks/deliveries?event=appointment.created&limit=1
x-api-key: sk_live_…
```

Risposta **200**:


```
{
  "success": true,
  "data": [
    {
      "id": "d8c27ece-8a95-4516-8149-2f0b51b058c8",
      "event": "appointment.created",
      "hook_id": "e763e622-1a6c-4a63-b248-facf0acc2f0c",
      "related_id": "a0ef7770-8b93-4291-a62a-9426f146b8fc",
      "status": "sent",
      "attempts": 1,
      "response_status": 200,
      "duration_ms": 27,
      "error": null,
      "next_attempt_at": null,
      "idempotency_key": "appointment.created:a0ef7770-8b93-4291-a62a-9426f146b8fc:e763e622-1a6c-4a63-b248-facf0acc2f0c",
      "created_at": "2026-11-25T21:29:49.393477+00:00",
      "sent_at": "2026-11-25T21:29:51.204+00:00",
      "updated_at": "2026-11-25T21:29:51.204+00:00",
      "hook_url": "https://webhook.site/…b3aa"
    }
  ]
}
```


Rimando a comando — `POST /api/v1/hooks/deliveries/:id/retry`. Rimanda subito una consegna già conclusa, con il corpo ricomposto adesso, la stessa `Idempotency-Key` e la stessa `Swavo-Delivery`; `attempt` sale di uno. La risposta è una voce del registro, come sopra. Una consegna ancora in coda risponde `409` (partirà da sola). Provato: l'avviso è arrivato una seconda volta, identico nell'identità, con `"attempt": 2`.

```
POST /api/v1/hooks/deliveries/d8c27ece-8a95-4516-8149-2f0b51b058c8/retry
x-api-key: sk_live_…
```

Risposta **200**:

```
{
  "success": true,
  "data": {
    "id": "d8c27ece-8a95-4516-8149-2f0b51b058c8",
    "event": "appointment.created",
    "hook_id": "e763e622-1a6c-4a63-b248-facf0acc2f0c",
    "related_id": "a0ef7770-8b93-4291-a62a-9426f146b8fc",
    "status": "sent",
    "attempts": 2,
    "response_status": 200,
    "duration_ms": 50,
    "error": null,
    "next_attempt_at": null,
    "idempotency_key": "appointment.created:a0ef7770-8b93-4291-a62a-9426f146b8fc:e763e622-1a6c-4a63-b248-facf0acc2f0c",
    "created_at": "2026-11-25T21:29:49.393477+00:00",
    "sent_at": "2026-11-25T21:30:05.782+00:00",
    "updated_at": "2026-11-25T21:30:05.782+00:00",
    "hook_url": "https://webhook.site/…b3aa"
  }
}
```


Ripartire da zero — `POST /api/v1/sandbox/reset`, solo sulle aziende di prova (`GET /api/v1/me` → `"sandbox": true`, come quella che vi diamo per le prove). Cancella clienti, richieste, appuntamenti, assenze, telefonate e registro delle consegne; restano l'azienda, gli operatori con i loro orari e le chiavi. Le iscrizioni agli avvisi restano, salvo `{ "include_hooks": true }`. Tutto o niente. Su un'azienda vera risponde `403` e non tocca niente (provato). Il registro delle vostre chiamate (`/logs`, sotto) invece non si svuota: tiene le righe per 30 giorni.

```
POST /api/v1/sandbox/reset
x-api-key: sk_live_…
Content-Type: application/json

{
  "include_hooks": true
}
```

Risposta **200**:


```
{
  "success": true,
  "data": {
    "reset": true,
    "deleted": {
      "calls": 0,
      "hooks": 1,
      "absences": 1,
      "requests": 3,
      "customers": 9,
      "deliveries": 23,
      "appointments": 6,
      "missed_calls": 0
    }
  }
}
```


Registro delle vostre chiamate — `GET /api/v1/logs`. Ogni chiamata fatta con le chiavi della vostra azienda, dalla più recente: istante, metodo, percorso, esito HTTP, durata sul nostro server e, se è andata male, **il codice e il messaggio dell'errore che avete ricevuto**. Serve a rispondere da soli a «ieri mi dava errore: quale chiamata, e perché?». Una chiamata compare qualche istante dopo la risposta; le righe si conservano **30 giorni**.

Filtro	Valori	Esempio
`status`	`error` (da 400 in su), `ok` (sotto 400), una classe (`4xx`, `5xx`) o un codice	`status=error`, `status=404`
`path`	il percorso comincia così; `/customers` e `/api/v1/customers` sono equivalenti	`path=/customers`
`method`	`GET`, `POST`, `PUT`, `PATCH`, `DELETE`	`method=POST`
`from`, `to`	istante ISO 8601 o data `YYYY-MM-DD` (mezzanotte UTC)	`from=2026-11-24T08:00:00Z`
`request_id`	l'intestazione `x-vercel-id` di una risposta, intera o solo l'ultima parte	`request_id=fra1::fra1::5k5ss-…`
`limit`	da 1 a 100, predefinito 20	`limit=50`

Cosa **non** c'è, per scelta: il corpo della richiesta e della risposta, mai. Nella `query` restano leggibili solo id, numeri, date e parole dell'API; telefoni, email, ricerche e ogni altro valore diventano `***` (di un telefono restano le ultime 3 cifre: `phone=***404`). Lo stesso vale per il percorso e per il messaggio d'errore.

Cosa **non compare**: le chiamate senza `x-api-key` e quelle con una chiave sbagliata (non sappiamo di chi sono). Un `401` compare solo se la chiave è vostra ma scaduta o disattivata. Gli errori ripetuti `401` e `429` si registrano fino a 60 l'ora per chiave.

```
GET /api/v1/logs?status=error&limit=3
x-api-key: sk_live_…
```

Risposta **200** (tre chiamate sbagliate di proposito: un cliente già esistente, un percorso scritto male, un corpo senza telefono):


```
{
  "success": true,
  "data": [
    {
      "id": "2cfacd83-7ed0-420d-a5eb-1c19f8076326",
      "created_at": "2026-11-25T21:30:06.7+00:00",
      "method": "POST",
      "path": "/api/v1/customers/upsert",
      "query": null,
      "status": 400,
      "duration_ms": 201,
      "error": {
        "code": "VALIDATION_ERROR",
        "message": "phone: Invalid input"
      },
      "request_id": "jf72f-1795642206652-a69532699cbb",
      "key": "sk_live_xxxx..."
    },
    {
      "id": "9ab4b164-c520-40d8-916d-21b492ac96f5",
      "created_at": "2026-11-25T21:30:06.424+00:00",
      "method": "GET",
      "path": "/api/v1/customer",
      "query": null,
      "status": 404,
      "duration_ms": 197,
      "error": {
        "code": "NOT_FOUND",
        "message": "Questo percorso non esiste. L'elenco delle porte: GET /api/v1/openapi.json"
      },
      "request_id": "lvmqm-1795642206359-9fb603c56493",
      "key": "sk_live_xxxx..."
    },
    {
      "id": "8af0bd75-eeca-4a11-baee-f041fdeba121",
      "created_at": "2026-11-25T21:30:05.962+00:00",
      "method": "POST",
      "path": "/api/v1/customers",
      "query": null,
      "status": 409,
      "duration_ms": 313,
      "error": {
        "code": "CONFLICT",
        "message": "Esiste gia un cliente con questo telefono: usa /api/v1/customers/upsert per aggiornarlo"
      },
      "request_id": "zjpql-1795642205918-8451861b2ec3",
      "key": "sk_live_xxxx..."
    }
  ]
}
```


Una chiamata riuscita ha `"error": null`; un telefono cercato si vede così: `"path": "/api/v1/customers/find"`, `"query": "phone=***404"`. `key` è l'inizio della chiave usata: se in azienda avete più chiavi, dice quale. `request_id` è l'ultima parte dell'intestazione `x-vercel-id` della risposta: se ci scrivete per un problema, mandateci quella.

Provato in produzione: 35 chiamate fra giuste e sbagliate (`200`, `201`, `400`, `401`, `404`, `409`), tutte ritrovate nel registro con il codice e il messaggio giusti; nessun telefono, nessun nome e niente dei corpi nel database. Il registro non rallenta l'API (mediana di 20 chiamate sulla stessa porta: 387 ms prima, 374 ms dopo).

Un percorso scritto male (`/api/v1/customer` al posto di `/api/v1/customers`) risponde `404` in JSON come ogni altro errore e finisce nel registro.

Descrizione OpenAPI — https://portal.swavo.ai/api/v1/openapi.json, senza chiave. Tutte le porte di questa guida, con campi, tipi, valori ammessi ed errori. Le risposte vere di questa guida sono confrontate con lei ogni volta che la rigeneriamo: 70 risposte vere, 70 conformi.

---

9. Andare in produzione

Quando il collegamento funziona sull'azienda di prova, per passare all'azienda vera:

1. **La chiave**: il titolare dell'azienda la crea dal portale (§1.1) e ve la passa. Mettetela al posto di quella di prova, in una variabile d'ambiente o in un archivio di segreti, mai nel codice.
2. **Verifica**: `GET /api/v1/me` deve rispondere con il nome dell'azienda vera e `"sandbox": false`. `POST /api/v1/sandbox/reset` lì risponde `403`: non si può svuotare un'azienda vera.

3. **Avvisi**: iscrivete l'indirizzo https vero (`POST /api/v1/hooks`, §8.1) e conservate il `secret`. Gli avvisi dell'azienda di prova non passano all'azienda vera: ogni chiave vede solo la sua azienda.
4. **Il vostro server**: risponde `2xx` entro 10 secondi, verifica la firma sul corpo grezzo (§8.3), scarta le `Idempotency-Key` già viste e gli avvisi con `"test": true` (§8.4).
5. **Import iniziale**: un'anagrafica grande va mandata con `POST /api/v1/customers/upsert`, un cliente per chiamata, rispettando le 1000 chiamate l'ora; per andare più veloci chiedeteci di alzare il limite.
6. **Se qualcosa non va**: `GET /api/v1/logs?status=error` per le vostre chiamate, `GET /api/v1/hooks/deliveries?status=failed` per gli avvisi (§8.7). Se scrivete a info@swavo.ai, mandateci il `request_id` o l'intestazione `x-vercel-id`.

---

10. Dati di prova

L'azienda di prova che vi diamo è vuota: due operatori («Titolare» e «Laura», orari 8-20 dal lunedì al sabato), nessun cliente, nessun numero di telefono né WhatsApp. Potete creare quello che volete e cancellarlo (come, al §1.8): non tocca nessun'altra azienda. Per ripartire da zero: `POST /api/v1/sandbox/reset` (§8.7). Se vi serve un insieme di dati realistico (operatori, clienti, appuntamenti), ve lo prepariamo.

Gli esempi di questa guida sono stati rigenerati il 25/11/2026 con 71 chiamate vere e 27 avvisi ricevuti sull'azienda di prova «Azienda di prova API».