Voice AI Appointment Booking

Setup guide for the n8n workflow. Google Calendar + Google Sheets, for Vapi and ElevenLabs Agents.

What you get

An n8n workflow that gives your voice AI agent three tools. The agent calls them during a phone call, and the workflow does the work in your Google Calendar and Google Sheet.

Tool	What it does
check_availability	Returns the free start times for a date and service.
book_appointment	Checks the slot one last time, then creates the calendar event. This prevents double bookings.
add_waitlist	Adds the caller to a Google Sheet when no suitable time is free.

Tested live with ElevenLabs Agents and Vapi. If Google Calendar is unreachable, the agent gets a spoken fallback message. It never offers made-up free times and never books blindly.

What you need

- An n8n instance, either n8n Cloud or self-hosted. A self-hosted instance must be reachable from the internet over HTTPS, because Vapi and ElevenLabs call it.
- A Google account with the calendar you want to book into, and a Google Sheet for the waitlist.
- A Google Calendar credential and a Google Sheets credential in n8n. On n8n Cloud this is a few clicks. On a self-hosted instance you first create a Google OAuth client; n8n's documentation covers it.
- A Vapi or ElevenLabs account with a voice agent.

Part 1 - Set up the workflow in n8n

Step 1. Import the workflow

1. In n8n, create a new empty workflow.
2. Open the workflow menu (top right, three dots) and choose **Import from file**.
3. Select **voice-agent-appointment-booking.json**.

Step 2. Create the waitlist sheet

1. Create a new Google Sheet.
2. Rename the first tab to exactly **Waitlist** (capital W).
3. Type this header row in row 1, one word per column, A to G:

```
name | phone | service | preferred_date | notes | status | created_at
```

4. Select the whole **phone** column and set Format > Number > **Plain text**. Otherwise Google Sheets removes the + sign from phone numbers.

Step 3. Connect your Google accounts

1. Open the node **Get Day Events**, click the credential field and add your Google Calendar account.
2. Select the same credential on **Get Slot Events** and **Create Booking**.
3. Open **Add to Waitlist** and add your Google Sheets account.

Step 4. Edit the Config node

Config is the only node you need to edit. Open it and change the values at the top of the code:

Setting	What to enter
timezone	Your business timezone, e.g. Europe/London or America/New_York.
calendarId	Replace YOUR_CALENDAR_ID. In Google Calendar: Settings > click your calendar > Integrate calendar > Calendar ID. For a main calendar this is your Gmail address.
waitlistSheetId	Replace YOUR_GOOGLE_SHEET_ID with the long ID from your sheet's address: docs.google.com/spreadsheets/d/**THIS-PART**/edit
waitlistSheetName	Waitlist (leave as is if you named the tab Waitlist).
openDays	1 = Monday ... 7 = Sunday. [2, 3, 4, 5, 6] means Tuesday to Saturday.
openTime / closeTime	Opening hours, 24-hour format, e.g. '09:00' and '18:00'.
slotStepMinutes	Gap between offered start times (30 = 09:00, 09:30, 10:00 ...).
minNoticeMinutes	No bookings sooner than this many minutes from now.
services	Your services and their length in minutes, e.g. 'Haircut': 45. The names must match what your agent sends.

Service names: the workflow matches the service the agent sends against this list (capitals do not matter). If you rename a service in Config, use the same name in your agent's prompt. An unknown service uses defaultDurationMinutes.

Step 5. Activate and copy your webhook URLs

1. Switch the workflow to **Active** (toggle, top right).
2. Open each of the three webhook nodes and copy the **Production URL**. Never use the Test URL: it stops working as soon as you close the editor.

Node	Path (end of the URL)
Check Availability Webhook	/webhook/booking-check-availability
Book Appointment Webhook	/webhook/booking-book-appointment
Add Waitlist Webhook	/webhook/booking-add-waitlist

Keep these URLs private. Anyone who has a URL can send requests to your calendar. Do not post them publicly.

Part 2 - Connect Vapi

Create three tools in Vapi (Tools > create a tool of type API Request, or Function). Menu names can change slightly as Vapi updates. Use the same pattern for each tool:

- **Method:** POST
- **Server URL:** the matching Production URL from Part 1
- **Parameters:** add the parameters below, all type string. Mark the bold ones as required.
- **Static Body Fields: leave empty.** Placeholder text in this section is the most common reason a tool fails.

Tool name	Parameters
check_availability	**date**, service
book_appointment	**name**, **phone**, **date**, **time**, service, notes
add_waitlist	**name**, **phone**, **notes**, service, preferred_date

Give each parameter a clear description, for example date: "The day the caller wants, as YYYY-MM-DD, e.g. 2026-11-03" and notes: "The caller's preferred time or any details". Add all three tools to your assistant.

> **Waitlist notes:** make **notes** a required parameter on add_waitlist. This is how the caller's preferred time gets saved in the sheet.

Part 3 - Connect ElevenLabs Agents

In your agent, add three webhook tools (Agent > Tools > add a webhook tool). Menu names can change as ElevenLabs updates.

- **Method:** POST, **URL:** the matching Production URL.
- **Body parameters:** the same parameters as in the Vapi table above. The workflow accepts the flat JSON body that ElevenLabs sends.
- Give each parameter a short description so the agent knows what to put in it. Required parameters: date for check_availability; name, phone, date, time for book_appointment; name, phone, notes for add_waitlist.
- Add the tools to the agent and paste the sample prompt from Part 4.

Part 4 - The agent prompt

The workflow does the checking. The prompt only needs to make the agent call the tools in the right order. Copy this and replace the capitalised parts:

```
You are [AGENT NAME], the friendly receptionist at [BUSINESS NAME]. You book appointments
  by phone.

Today is [CURRENT DATE VARIABLE]. Timezone: [YOUR TIMEZONE]. The caller will say dates
  naturally ("tomorrow", "October 31st", "next Tuesday"). You convert them yourself and
  send YYYY-MM-DD with dashes to the tools, e.g. 2026-11-03. Never ask the caller to say a
  date in that format. Use 24-hour times like 14:00.

Services: [SERVICE 1] ([N] min), [SERVICE 2] ([N] min). Open [DAYS], [HOURS]. Closed
  [DAYS].

How to work:
1. Ask which service and which day. Do not call a tool until you know both.
2. Call check_availability. Suggest two or three of the returned slots, but remember the
  FULL list.
3. If the caller asks for a specific time, check it against the full list. If it is in the
```

```
   list, it is free: book it. Only say a time is unavailable if it is not in the list.
4. When the caller picks a time, ask for their name and phone number. Read the phone
   number back and ask the caller to confirm it. Then call book_appointment.
5. If no slot suits them, offer the waitlist and call add_waitlist. Always put their
   preferred time in notes.
6. Every tool reply contains a "message" field. Follow its instructions exactly, including
   when something fails.
7. Never say a booking is confirmed unless book_appointment returned success. Never state
   availability without calling check_availability first.

Keep answers short and natural. This is a phone call. Confirm the booking details back to
   the caller at the end.
```

Never hardcode today's date. If the prompt says a fixed date, the agent gets "tomorrow" wrong. Use your platform's current-date variable. In Vapi, for example: {{"now" | date: "%A, %d %B %Y", "Europe/London"}} (use your own timezone). Services in the prompt must match the names in Config.

Phone numbers: speech recognition sometimes cuts a number short. Asking the agent to read the number back (step 4) catches most of these before they are saved.

Part 5 - Test it

Before you share the number with real customers, make a call and check each step:

1. Say: "I'd like an appointment tomorrow." The agent should ask for the service, then read out free times.
2. Pick a time and give a name and phone number. The event should appear in your Google Calendar.
3. Call again and ask for the same time. It should now be offered as taken.
4. Ask for a day with no free times and say yes to the waitlist. A row should appear in your sheet, with the preferred time in **notes**.
5. Ask for a closed day. The agent should say you are closed.
6. Delete the test events and test rows.

If something does not work, open n8n > Executions, click the latest run and read the output of the failing node. The tool log on the Vapi or ElevenLabs side only says the request returned; the real result is in n8n.

Troubleshooting

Problem	Likely cause and fix
The tool fails or nothing reaches n8n	The workflow is not Active, or the Test URL is used. Switch it on and use the Production URL.
The agent gets the wrong day for "tomorrow"	The prompt has a hardcoded date. Use the current-date variable (Part 4).
"I can't reach the booking system"	Google Calendar failed: expired or missing credential, or a wrong calendar ID. Reconnect the credential and check calendarId in Config.
Waitlist row not written	The tab is not named Waitlist, the header row differs, or the sheet ID is wrong. Check Config and the header row.
Phone number loses its + sign	Set the phone column to Format > Number > Plain text.
Agent says a time is taken but it is free	The agent ignored the full list. Keep rules 2 and 3 in the prompt. Remember that long services block every overlapping start time.
Service not recognised	The name differs from the list in Config. Make the prompt and Config match.
Vapi shows "Completed successfully" but nothing happened	That only means the request returned. Check n8n > Executions for the actual result.

Customising

- **Hours, days, services, slot spacing:** all in the Config node. Nothing else needs editing.
- **Several calendars or locations:** duplicate the workflow, give the copy its own Config and use new webhook paths.
- **Language and voice:** set in your voice platform. The workflow's replies are plain English messages for the agent to act on, and the agent speaks them in the language of your prompt.