n8n Automation for Psychology Practice

1.Overview

This n8n workflow automatically sends a reminder email 48 hours before a patient appointment listed in Google Calendar. It:

Scans upcoming events, detects those within the next 48 hours

Extracts patient name, email (from attendees or description), phone (from description)

Formats datetime in Italian (it-IT) human format

Sends a personalized HTML email via Gmail with Dr. Hadassa de Castro's signature

Marks the calendar event with a \n\n[REMINDER_SENT] <timestamp> tag in the description to avoid re-sends

Runs frequently (cron) to operate 24/7

Target audience for this doc: developer who will import the JSON into n8n and an operations/clinic contact who needs to understand behaviour.

2) What this workflow does (short)

Trigger: Cron (every minute in current JSON — configure to every 15 minutes to reduce quota use)

Listing: Google Calendar: List events between now and now + 48h

For each event: Filter & Extract (Function) returns structured JSON with eventId, summary, formattedDateItalian, patientEmail, phone if present

Sends an HTML email via Gmail

Updates the same calendar event to add [REMINDER_SENT] <timestamp> to the description

3) Files included / importable

My workflow 14 — the full JSON you provided. Import it into n8n via Settings → Workflows → Import from file.

Important: The JSON contains credential references (IDs). After import, open each credential in the nodes and re-select or re-create the correct OAuth2 credentials for your instance (Google Calendar, Gmail).

4) Prerequisites & credentials

n8n instance (Self-hosted or SaaS) with adequate execution time and OAuth support.

Google Calendar OAuth2 credential with scopes:

https://www.googleapis.com/auth/calendar.events.readonly (LIST) and https://www.googleapis.com/auth/calendar.events (UPDATE) — update scope required to write description.

Gmail OAuth2 credential with https://mail.google.com/ scope or https://www.googleapis.com/auth/gmail.send (recommended).

The calendar used in the workflow should be accessible by the OAuth user used by n8n.

Security: Use least-privilege scopes and a dedicated service account / OAuth client where possible. Store credentials securely in n8n credentials.

5) Node-by-node documentation

Below each node name you will find the main parameters and notes.

Cron (every 15m)

Type: n8n-nodes-base.cron

Current parameters: everyMinute (you may change to every 15 minutes)

Purpose: Poll Google Calendar continuously.

Recommendation: Use every 15 minutes to avoid hitting API quotas and reduce duplicate execution risk.

Google Calendar: List events

Type: n8n-nodes-base.googleCalendar (operation: list)

Parameters:

calendar: skimtiaz.prof@gmail.com (replace with client calendar)

start: ={{ new Date().toISOString() }}

end: ={{ new Date(new Date().getTime() + 1000*60*60*48).toISOString() }}

Purpose: Return all events from now until 48 hours ahead.

Notes:

This node returns full event objects; Filter & Extract expects fields such as start.dateTime, summary, description, attendees, organizer.

SplitInBatches (per event)

Type: splitInBatches

Purpose: Process each event individually so errors in one event don’t block others.

Filter & Extract (Function)

Type: function

Purpose: Core logic. Key behaviour:

Reads items[0].json (the event object)

Extracts start from e.start.dateTime or e.start.date

Parses startDate and calculates hours difference from now

Skips events with description including [REMINDER_SENT]

Extracts patientEmail from attendees[] or from description via regex fallback

Extracts phone with a regex fallback

Builds formattedDateItalian using Intl.DateTimeFormat('it-IT', { year, month, day, hour, minute })

Returns structured object: { eventId, calendarId, summary, startDateTime, formattedDateItalian, description, patientEmail, phone, rawEvent }

Important behaviour: The current function has a testing comment that disables the 48-hour filter (line with // TEST MODE - allow all events). Before production, re-enable the filter by uncommenting or removing the TEST MODE bypass so only events within 0–48 hours are processed:

```
if (diffHours < 0 || diffHours > 48) return [];
```

Edge cases to consider:

All-day events use start.date — function handles both, but timezone interpretation differs.

Recurring events return recurrence info; ensure that start.dateTime is present for occurrences.

Gmail: Send Email

Type: n8n-nodes-base.gmail

Parameters:

resource: message

subject: Promemoria Appuntamento: {{$json["formattedDateItalian"]}}

includeHtml: true with provided HTML template (Italian copy)

toList: currently ={{ $json.rawEvent.creator.email }}

Important fix to deliver to patient: Currently the node sends to the event creator (rawEvent.creator.email). To send to the patient, change toList to use {{$json.patientEmail}} or falling back to rawEvent.creator.email, e.g.:

```
={{ $json.patientEmail || $json.rawEvent.creator.email }}
```

HTML Template: professional Italian template included. Replace studio contact info and reply-to if desired.

Google Calendar: Update event (mark sent)

Type: n8n-nodes-base.googleCalendar (operation: update)

Parameters:

calendar: skimtiaz.prof@gmail.com

eventId: ={{$json["eventId"]}}

description: ={{ ($json["rawEvent"].description || "") + "\n\n[REMINDER_SENT] " + $moment().toISOString() }}

sendUpdates: none

Purpose: Appends a timestamped [REMINDER_SENT] tag to the description to prevent re-sending.

Note: Using description is a safe approach but if the practice prefers a dedicated custom field, you can use event extendedProperties instead.

Finish batch

No-op to mark completion.

6) Data mapping (inputs → outputs)

Input (Google event object) → Output (structure used by Gmail & Update node):

e.id → eventId

e.organizer.email → calendarId (fallback to primary)

e.summary → summary (patient name)

e.start.dateTime → startDateTime (ISO)

formattedDateItalian (string) → used in email subject & body

patientEmail → used as email recipient (change Gmail node as noted)

phone → displayed in email if present

7) How the workflow avoids duplicate reminders

The Function node checks for the string [REMINDER_SENT] in the event description; if found it skips the event.

After sending email, the Google Calendar Update node appends [REMINDER_SENT] <timestamp> to the description.

Alternative (recommended for robustness): Use extendedProperties.private (Google Calendar extended properties) with a boolean flag key such as reminder_sent=true. This avoids cluttering descriptions and is programmatic.

Example update payload for extended properties (concept):

```
{"extendedProperties": {"private": {"reminder_sent": "true"}}}
```

8) Testing checklist & troubleshooting

Before enabling production

Re-import the workflow into a staging n8n instance.

Reconnect credentials (Google Calendar & Gmail) — re-authorize if needed.

Edit Filter & Extract function to re-enable the 48-hour filter (remove test bypass).

Change Gmail toList to ={{ $json.patientEmail || $json.rawEvent.creator.email }}.

Set Cron to every 15 minutes.

Create test events on the calendar with:

event title = Test Patient Name

attendee email or include testpatient@example.com in description

include a phone number in description to test extraction

event start time = now + 30 minutes (or within 48h)

Run the workflow manually or wait for cron to trigger.

Confirm patient receives the email and that the calendar event shows [REMINDER_SENT] with timestamp.

Common failures

No emails sent: check Gmail credential scopes and that Gmail isn't blocking access. Check n8n execution logs and node input/output.

patientEmail is empty: ensure attendees include an email or description contains the email. Consider requiring patient email during booking.

Timezone issues: confirm server timezone and calendar event timezone. Use event start.timeZone or normalize times using moment/timezone if needed.

9) Modularity — Adding WhatsApp later

The workflow is already modular (SplitInBatches per event) so adding another branch is simple:

Add a new node after Filter & Extract for the WhatsApp provider (e.g., Twilio, MessageBird, or WhatsApp Cloud API).

Use the same formattedDateItalian, patientEmail, and phone fields to send SMS/WhatsApp messages.

Ensure both Gmail and WhatsApp success lead to updating the calendar event; or update only once both channels succeed depending on business rules.

Design note: If you plan to send multi-channel reminders, prefer to use extendedProperties storing reminder_sent_email=true and reminder_sent_whatsapp=true so you can track per-channel status.

10) Suggested pricing & delivery estimate (for your client quote)

These are guideline figures — adjust for local rates, complexity, testing, and support expectations.

Development (build + staging test): 4–7 hours — Suggested $200–$450

Create workflow, configure OAuth, implement robust parsing, add extendedProperties option.

QA & Documentation: 1–2 hours — Suggested $50–$125

Optional extras (WhatsApp integration, multi-calendar support, retries, dashboard): additional $100–$400 depending on provider and testing.

Total estimate: $250–$975 depending on extras and hourly rate. Provide a fixed-price quote or hourly (e.g., $50–$75/hr) depending on your preference.

Delivery: Basic working version: 1–3 business days. With polishing, robust testing and client sign-off: 3–7 business days. (Adjust based on your availability.)