PROJECT OVERVIEW · AI WORKFLOW BUILT IN N8N

Instant Translator — French ↔ Hungarian

Speak or write naturally; AI translates automatically, both ways, inside Telegram — so French and Hungarian relatives talk without a language barrier.

Built by Gauvain van Ghele — Product Builder **Engine** n8n (ported from Make in ~20 min) **Surface** Telegram (text & voice)

01 The idea

A simple, personal 0→1 build: let two sides of a family — French speakers and Hungarian speakers — keep talking naturally, each in their own language. No new app to learn: everything happens inside a Telegram chat they already use. You send a message — typed or spoken — and the other side receives it translated, as text or as a natural voice note.

02 How it works

TEXT PATH

A Telegram message arrives → a router detects it's text → Claude detects the language and translates in context (FR↔HU) → the translation is sent straight back into the chat.

VOICE PATH

A voice note arrives → the file is downloaded → Whisper transcribes it → Claude translates the text → OpenAI TTS speaks it back in a natural voice → it's sent as a voice note.

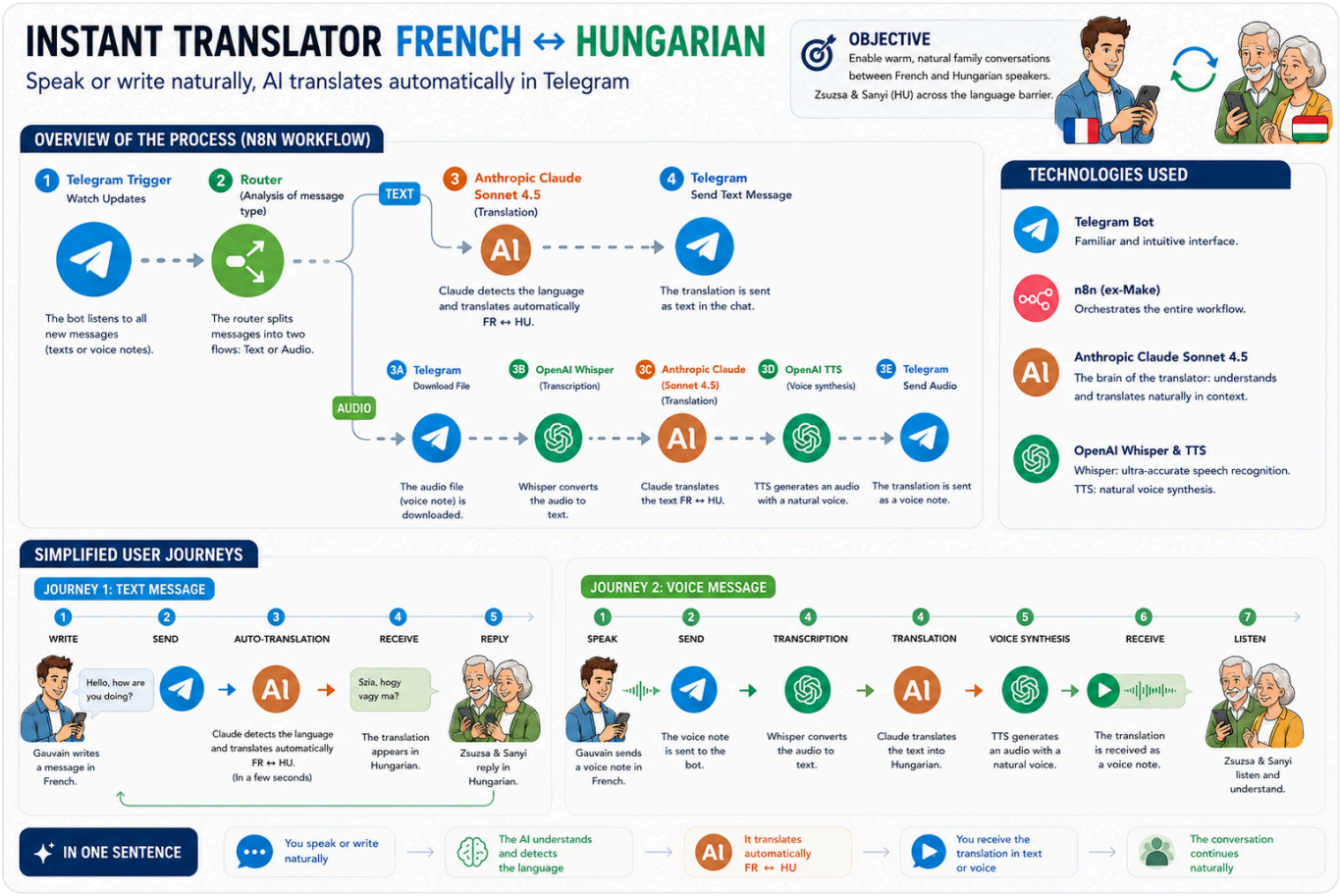

Fig. 1 — End-to-end flow and the two user journeys (text and voice), with the technologies used at each step.

03 Inside the n8n editor

The whole thing is a single n8n workflow: a Telegram trigger, a router that splits text from voice, and two branches that call Claude, Whisper and OpenAI TTS over HTTP before sending the result back to Telegram. I prototyped it on Make first, then rebuilt it natively in n8n in about twenty minutes.

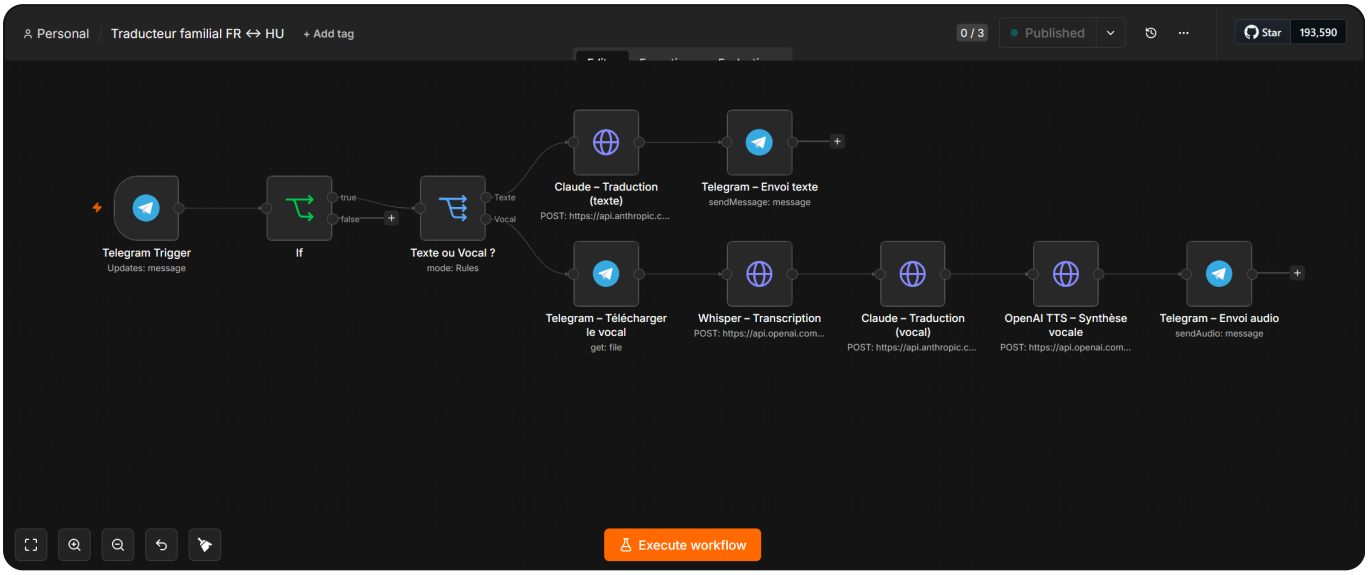

Fig. 2 — The workflow in n8n: Telegram trigger → router (text / voice) → Claude translation for text, and the audio path (download → Whisper → Claude → OpenAI TTS → Telegram). Published workflow.

`n8n · orchestration` `Telegram Bot` `Claude · translation` `OpenAI Whisper · transcription` `OpenAI TTS · voice` `HTTP · APIs`

Why it matters as a build: a small, real AI product — multi-step, multi-model, bidirectional, text and voice — wired end to end and actually used by a family. The same orchestration patterns scale to far larger systems; here they solve a human problem in a few nodes.