

Building in Dify: from recording to minutes



Guide 5 for the AI Agents workshop

LeX Consultancy B.V. · 21 September 2026

You build a workflow that turns an audio recording of a meeting into text and writes minutes from it: a summary, decisions, action points with owner and date, and what was deferred. At the end you have a web address where a colleague uploads an mp3 and gets the minutes and the full transcript back within two minutes.

Compared with guides 1 and 3 two things are new: a **file as input** instead of text, and a **tool node** (Speech To Text) instead of only LLM nodes. The rest you know.

This guide follows the cloud version of Dify as it looked on 21 September 2026. Dify changes fast: if your screen differs, follow what you see, not what is written here.

How to read this guide

Form	Meaning
1. 2. 3.	What you do: click, choose, open
Bold	A button, menu or field as it appears on your screen
Table <i>What you type</i>	Values you put in a form field
Grey box <i>Type in</i>	Text you type or paste literally
Orange box	A pitfall you would otherwise discover yourself
Green box <i>Your choice</i>	A place where you can replace our example with your own material, style or prompt

The design card for this app

Before we started clicking, this card was filled in (the blank card is in the Design card appendix). Fill in the same five boxes for your own app; the loose version of this filled-in card is in 00-design-card/examples.

1. Who is it for	2. What goes in
The minute-taker of a department meeting whose minutes are always two weeks late. Has a 45-minute mp3 and fifteen minutes of time.	One audio file (mp3, m4a or wav, at most 5 MB) through a file field. For the prototype only a fictional recording read out by a computer voice.

3. What comes out	4. What it must stick to
Minutes with five headings (Meeting, Summary, Decisions, Action points with owner and date, Deferred) and below them the full transcript. Uncertain names with a question mark.	No real recordings in Dify cloud (privacy frame); delete audio after transcription; someone reads the minutes. Tool: Speech To Text with your own OpenAI key.

5. How you know it works

Test	What goes in	What must come out
1	The fictional three-minute recording	Minutes with five headings, three action points with an owner
2	The same recording at 128 kbps (too big)	An error about the file size; shrink to 32 kbps
3 (the hard one)	A photo instead of audio	The input field refuses the file; only Audio is enabled

Read this first: what is and is not allowed

A voice is personal data. So these agreements apply, and they are not up for discussion:

What	Agreement
Recordings of real people in Dify cloud	No. Dify cloud is normally not covered by your school's data processing agreement. This guide works with a fictional recording read out by a computer. Use that, or make one yourself (appendix A).
Real use	Only on a platform your school has approved for personal data (a data processing agreement, processing in your jurisdiction). Dify can run on such a platform too, but that is a separate set-up outside this guide.
Recording a meeting	Explicit consent from all participants beforehand. Structural recording: inform the staff council or works council.
Retention	Delete the audio as soon as the transcript exists. Dify keeps uploaded files and results; see "Housekeeping" at the end.
Students	A scenario with speaking tests or student conversations is only allowed after a data protection impact assessment and after informing parents. This guide is only about staff meetings.
Accessibility	If the tool is used for students: for all students, not only for students with a disability.

In short: what you build here is a working prototype to learn how it works. The step to real use is a separate decision with a separate environment.

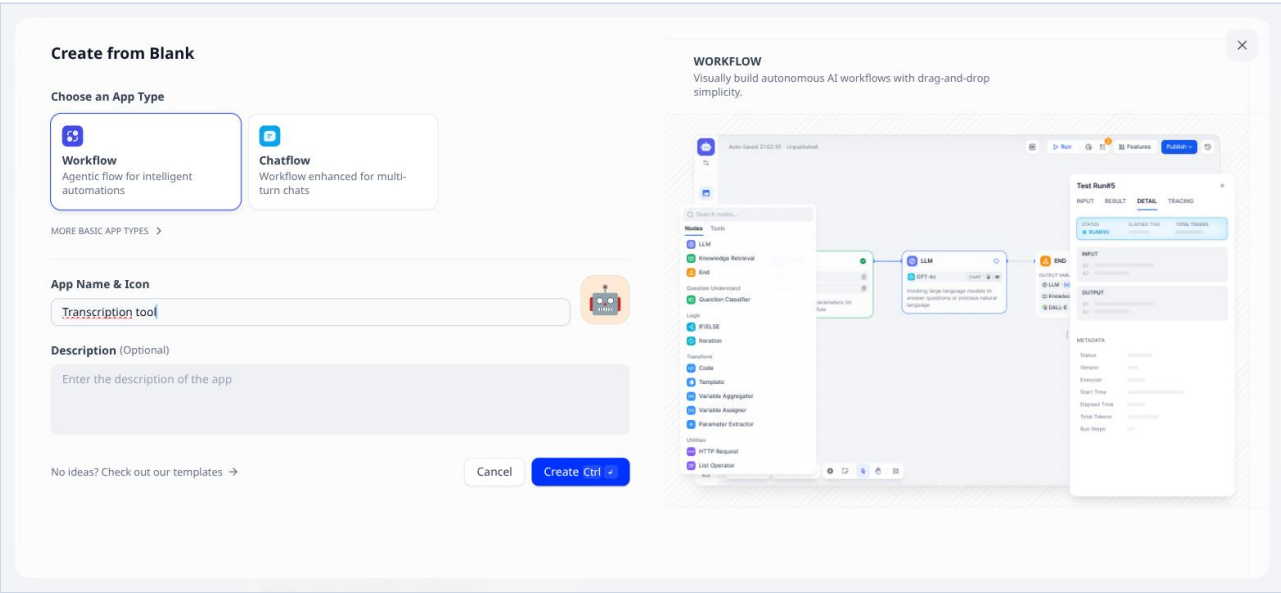
Preparation

- A Dify account with your own OpenAI key, set to **Usage priority: API key** (Integrations → Model Provider → OpenAI → Config). Without your own key the Speech To Text tool does not work: Dify's free credits only cover text models.
- A test recording: `department-meeting-small.mp3` from the folder of this guide, the fictional English department meeting read by a computer voice (2 min 47 s, 668 kB). It was made with the Text To Speech tool in Dify, see appendix A; `make_recording.py` does the same from the command line with your own OpenAI key. `sample-meeting-dutch-small.mp3` is the Dutch version; the tool transcribes Dutch just as well and the minutes come out in the language of your prompt.
- Cost per run through your own key: transcription about 2 cents, minutes less than 1 cent.

Step 1. Create the workflow

1. Go to **Studio**, click **Create** and choose **Create from Blank**.
2. Click **Workflow**.
3. Fill in the name.

Field on screen	What you type
App Name & Icon	Transcription tool



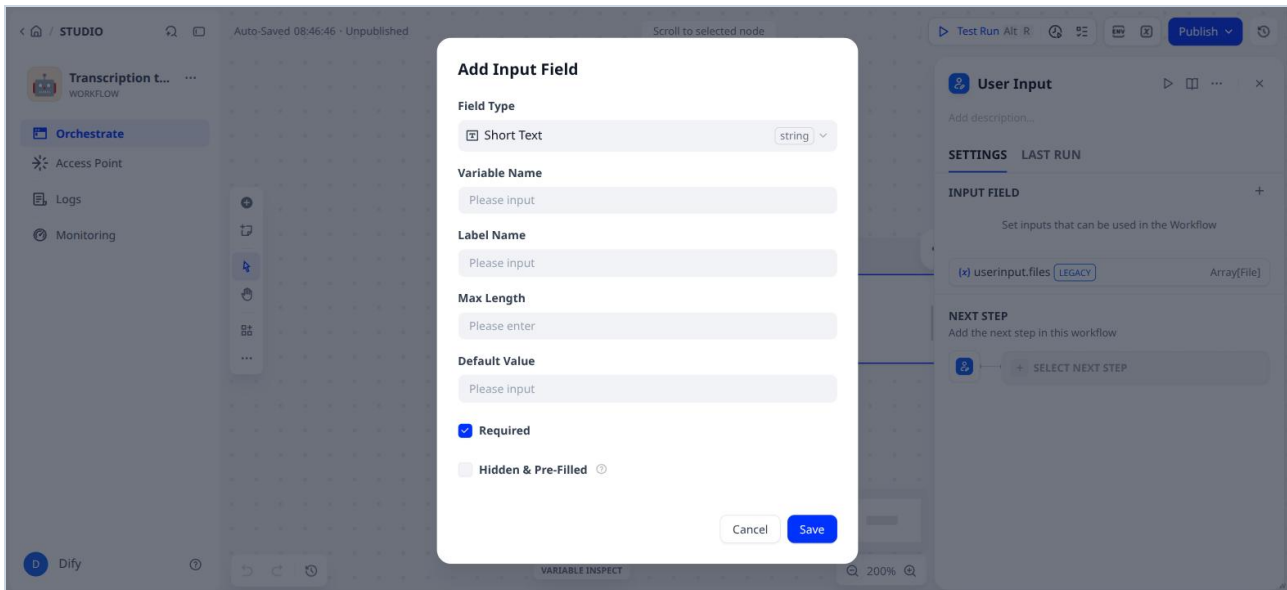
Workflow chosen, name filled in

4. Click **Create**.

A sandbox account may have five apps, and agents in the Agents menu count. If you see “Upgrade to create more apps”, first delete an old app via the three dots on its card in Studio.

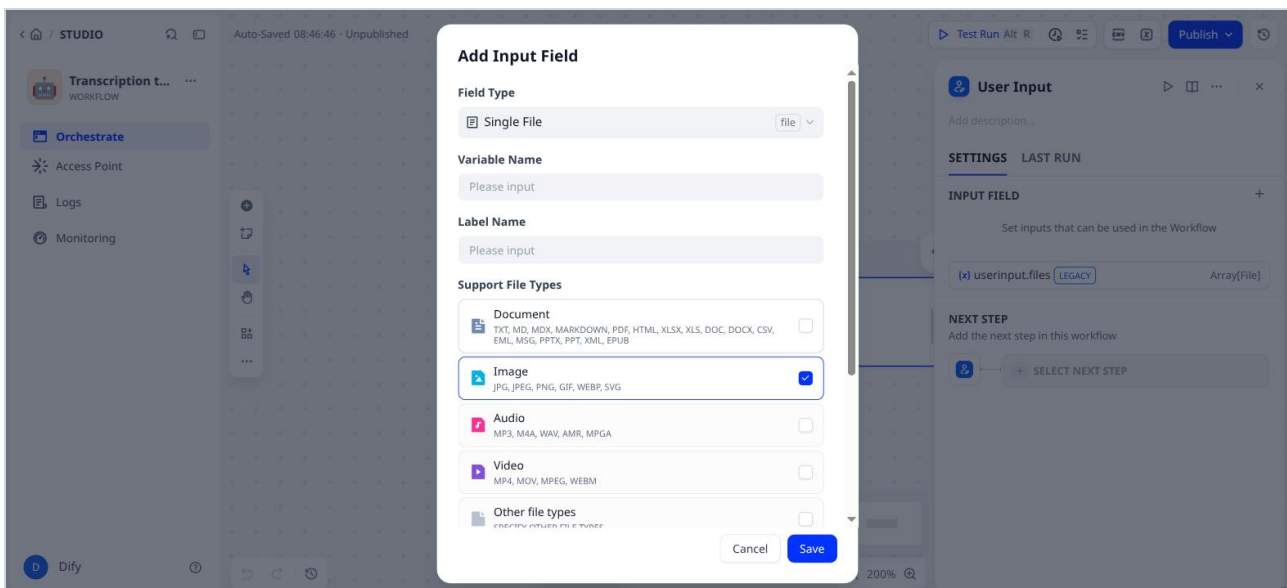
Step 2. Create an input field for the audio file

1. Click **User Input** in the right-hand panel.
2. Click the plus sign next to **INPUT FIELD**.



The Add Input Field dialog

3. Open **Field Type**. You see the kinds of fields: Short Text, Paragraph, Select, Number, Checkbox, and at the bottom **Single File** and **File List**.
4. Choose **Single File**. The window gets longer: file types appear.

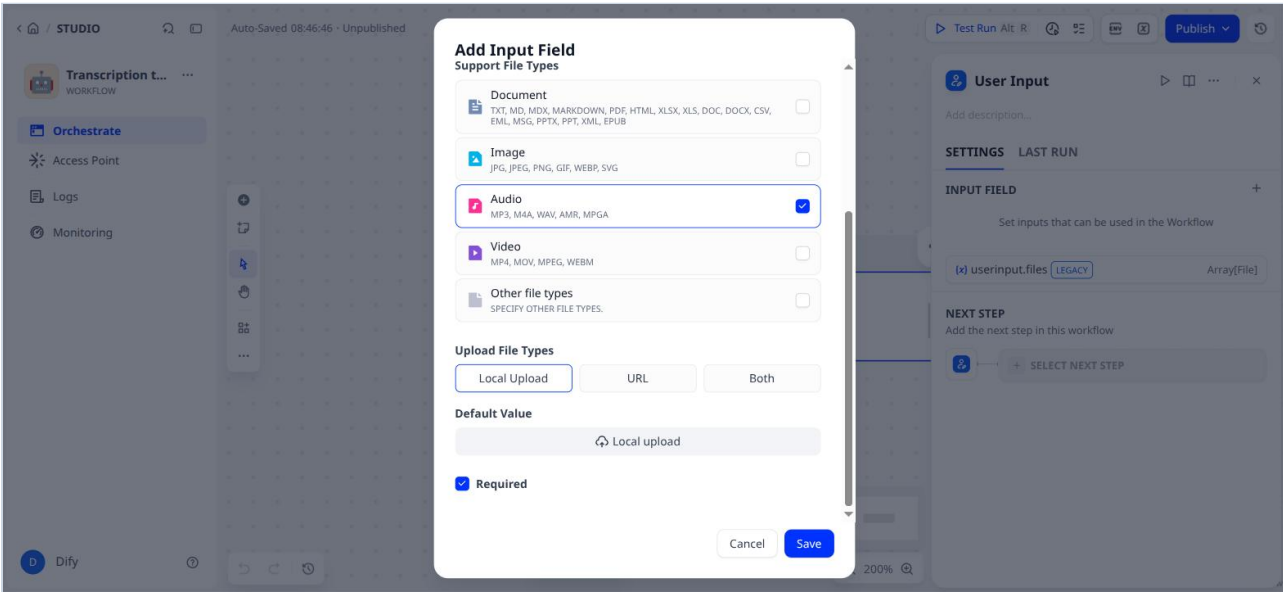


Single File: choosing file types

5. Fill in the fields.

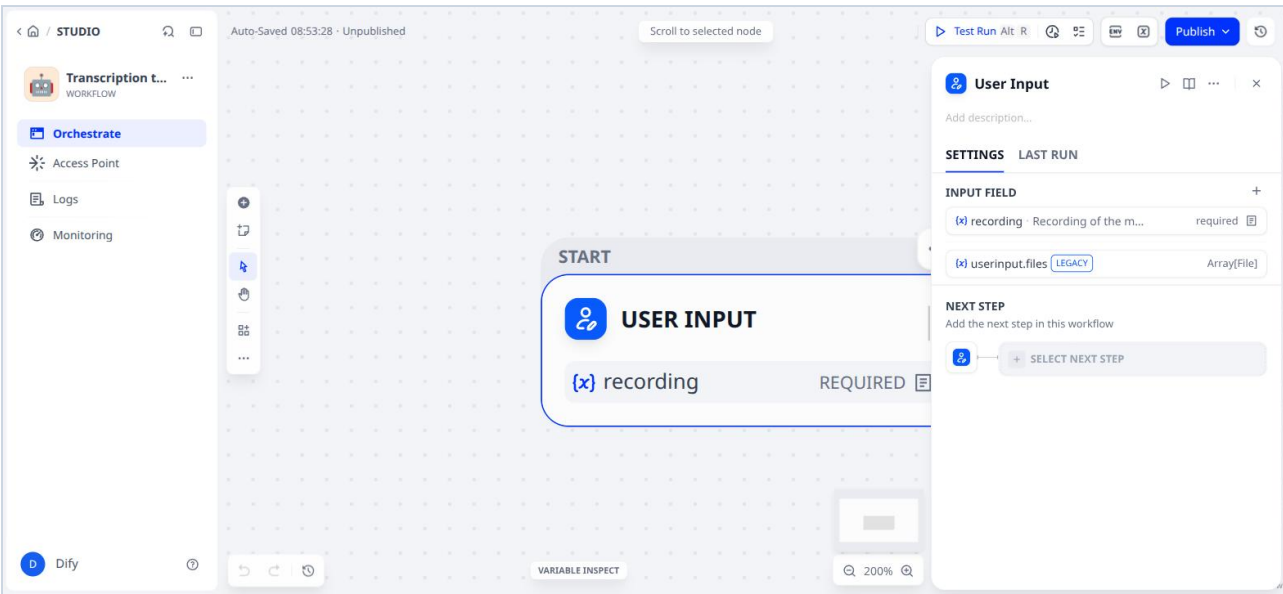
Field on screen	What you type or choose
Variable Name	recording
Label Name	Recording of the meeting (mp3, m4a or wav)
Support File Types	switch Image off, switch Audio on
Upload File Types	Local Upload

Image is on by default. Switch it off, otherwise someone can upload a photo later and the workflow gets stuck at the transcription step.



Field recording, only Audio ticked

6. Click **Save**.

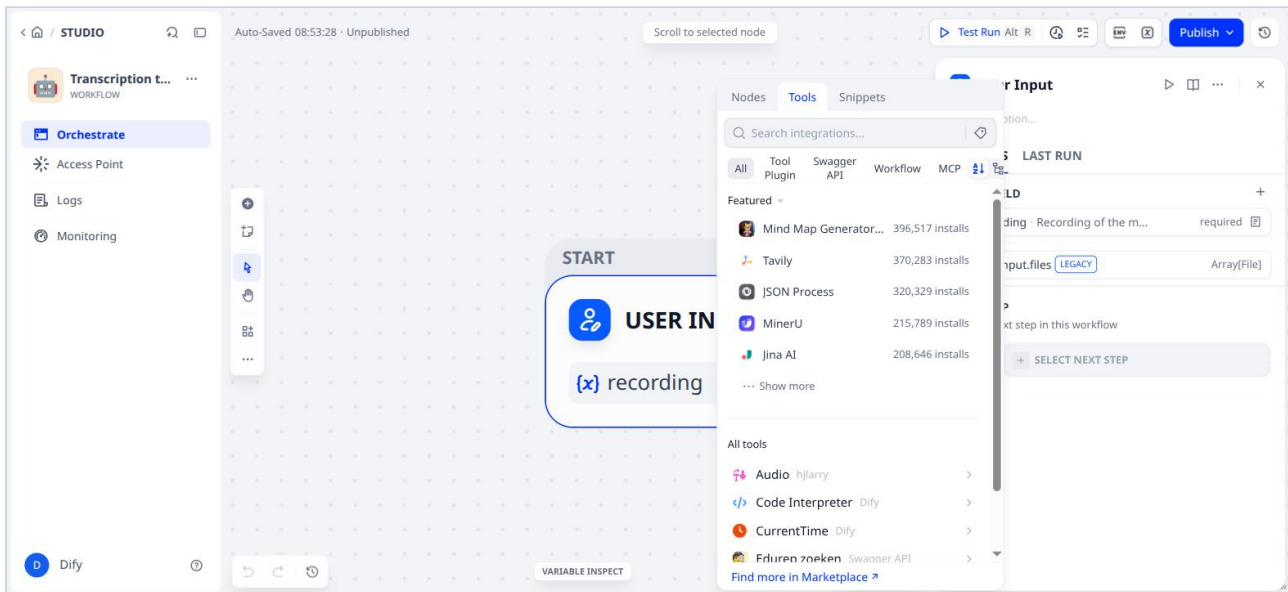


The input field is in the node

Step 3. Add the transcription step

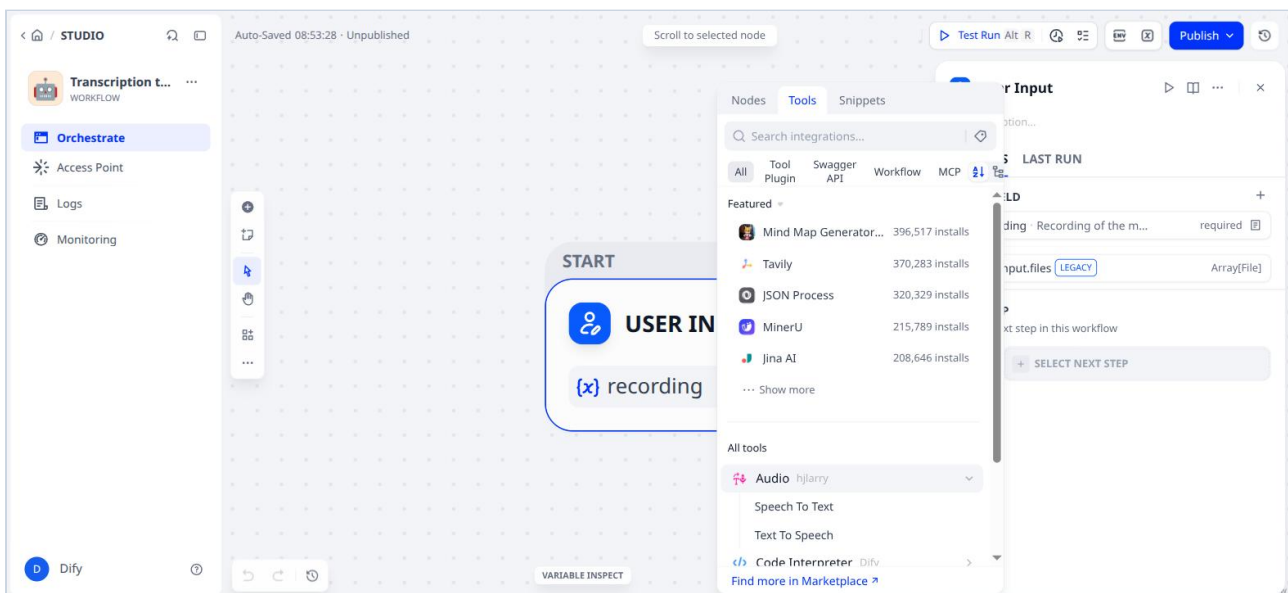
This is the new building block: not an LLM node but a **tool**. Dify has a built-in Audio tool that turns speech into text through the speech model of your provider.

1. Under **NEXT STEP** click **SELECT NEXT STEP**.
2. At the top of the list click the **Tools** tab.



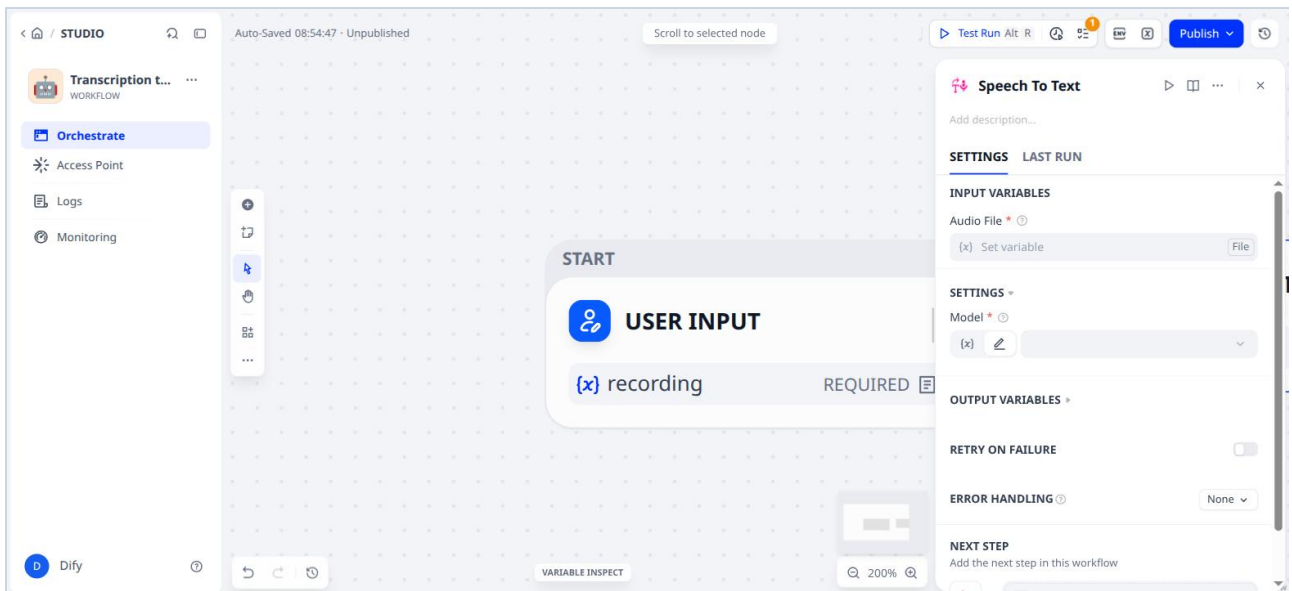
The Tools tab

3. Under **All tools** click **Audio**. Two options unfold.



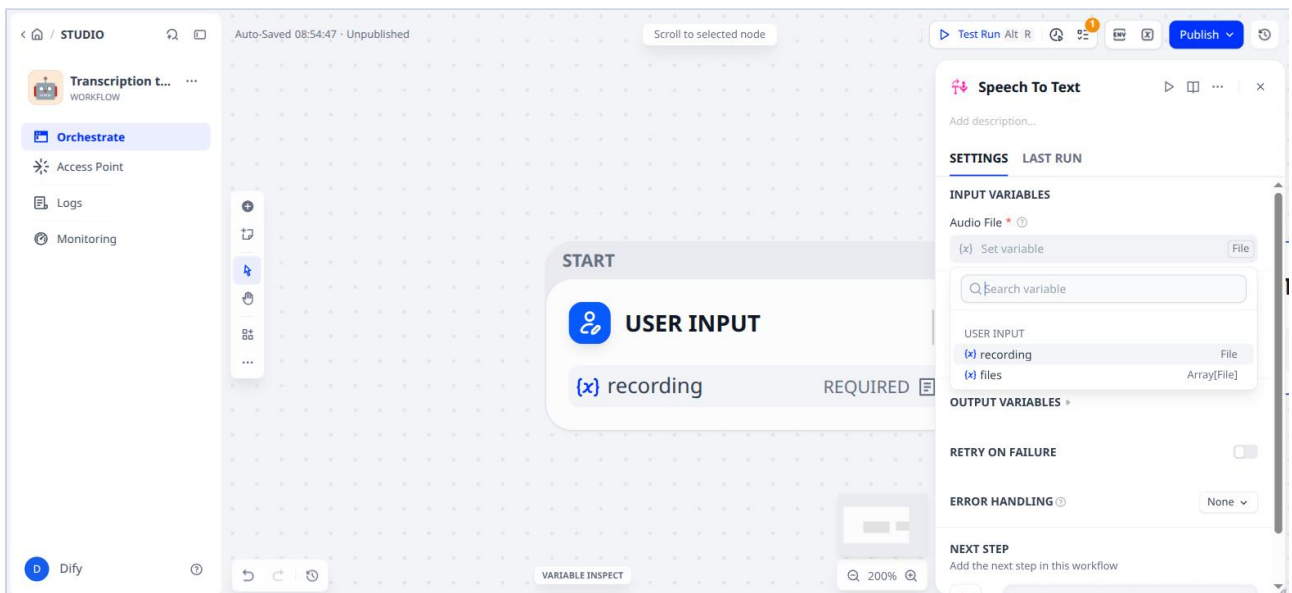
Audio: Speech To Text and Text To Speech

4. Choose **Speech To Text**.



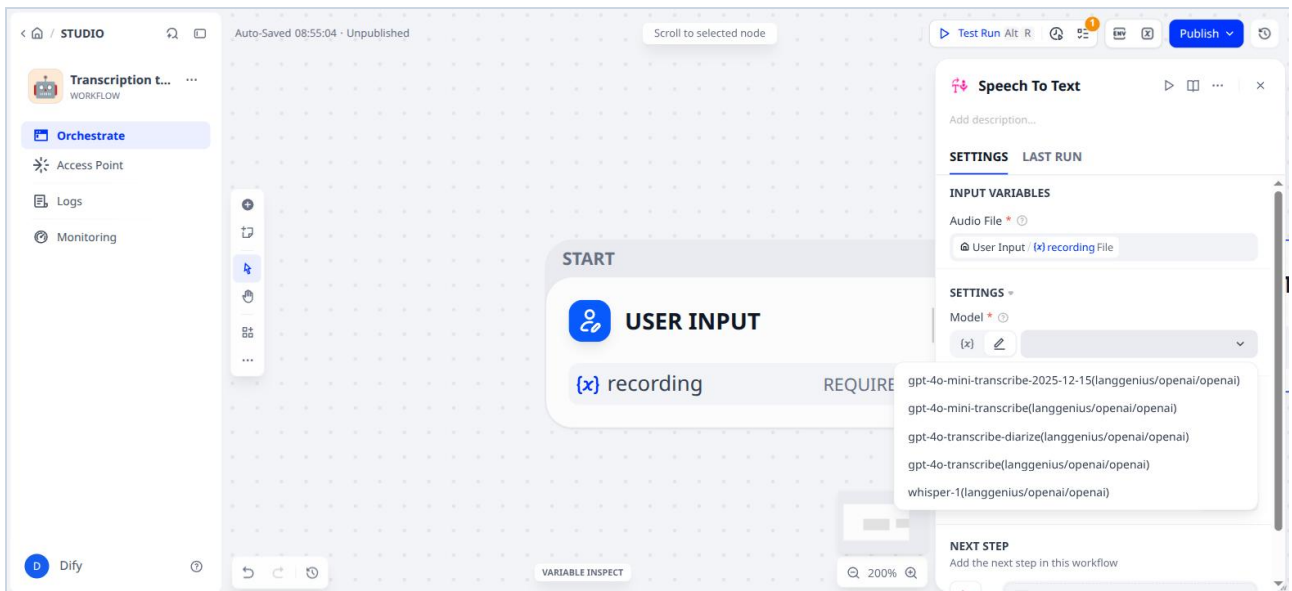
The empty Speech To Text node

5. Under **Audio File** click **Set variable** and choose the variable **recording** under USER INPUT.



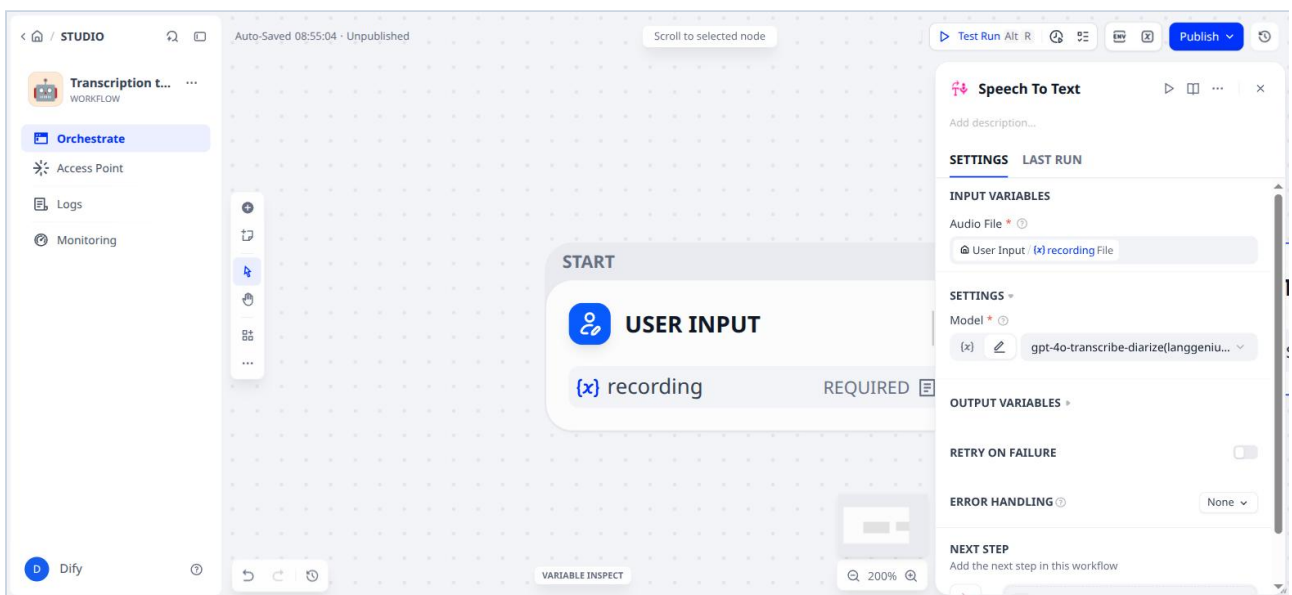
Connecting the recording

6. Under **Model** click the drop-down. You see the speech models available through your key.



The speech models

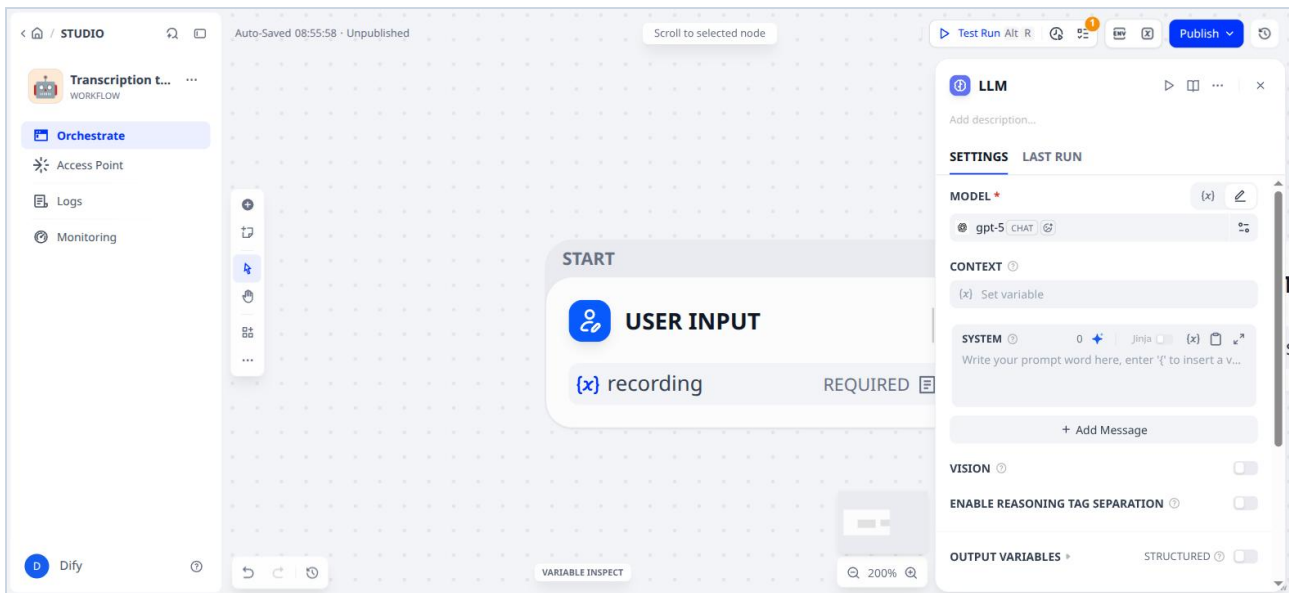
7. Choose **gpt-4o-transcribe-diarize**. "Diarize" means the model separates speakers; that helps with a meeting. If you want cheaper, choose **whisper-1**.



Speech To Text set up

Step 4. Add the minutes step

1. Click **SELECT NEXT STEP** and under **Nodes** choose the **LLM** node.



The empty LLM node

2. Click the model name **gpt-5** and choose **gpt-5-mini**.
3. Click in the **SYSTEM** field and type the instruction.

TYPE IN

Start your answer with the line # MINUTES followed by an empty line.

You write the minutes of a meeting at a school, based on the transcript below. The transcript comes from automatic speech recognition and may contain errors in names and numbers; write down what is most likely and put a question mark in brackets after anything you are not sure of.

Write in English, businesslike and brief. Use exactly this structure with these headings:

Meeting

One line: which meeting, date, those present (if the transcript shows it).

Summary

At most five sentences about what was discussed.

Decisions

Numbered list. Only what was really decided, no intentions.

Action points

Table with columns: What, Who, When. If no date is given, write "not mentioned".

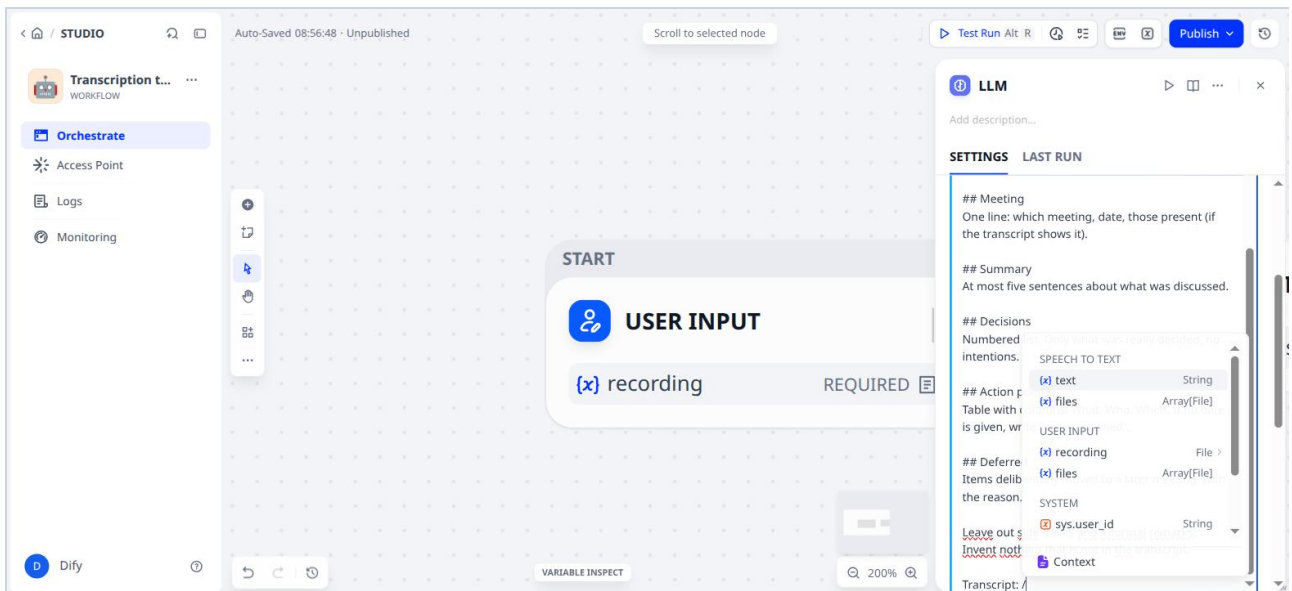
Deferred

Items deliberately moved to a later meeting, with the reason.

Leave out side tracks and informal remarks. Invent nothing that is not in the transcript.

Transcript:

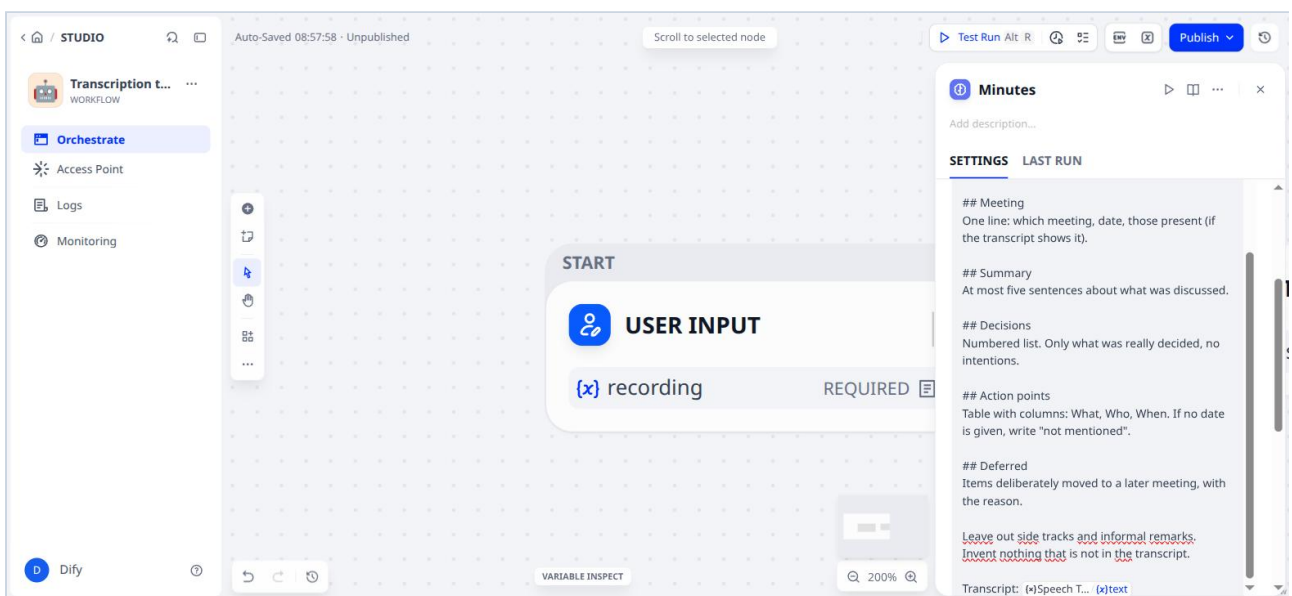
4. Right after "Transcript:" type a forward slash /. At the top of the list is now **SPEECH TO TEXT** with the variable **text**.



The variable list with the output of Speech To Text

5. Choose **text** under SPEECH TO TEXT.

6. Click the name **LLM** at the top of the panel, replace it with **Minutes** and press Enter.



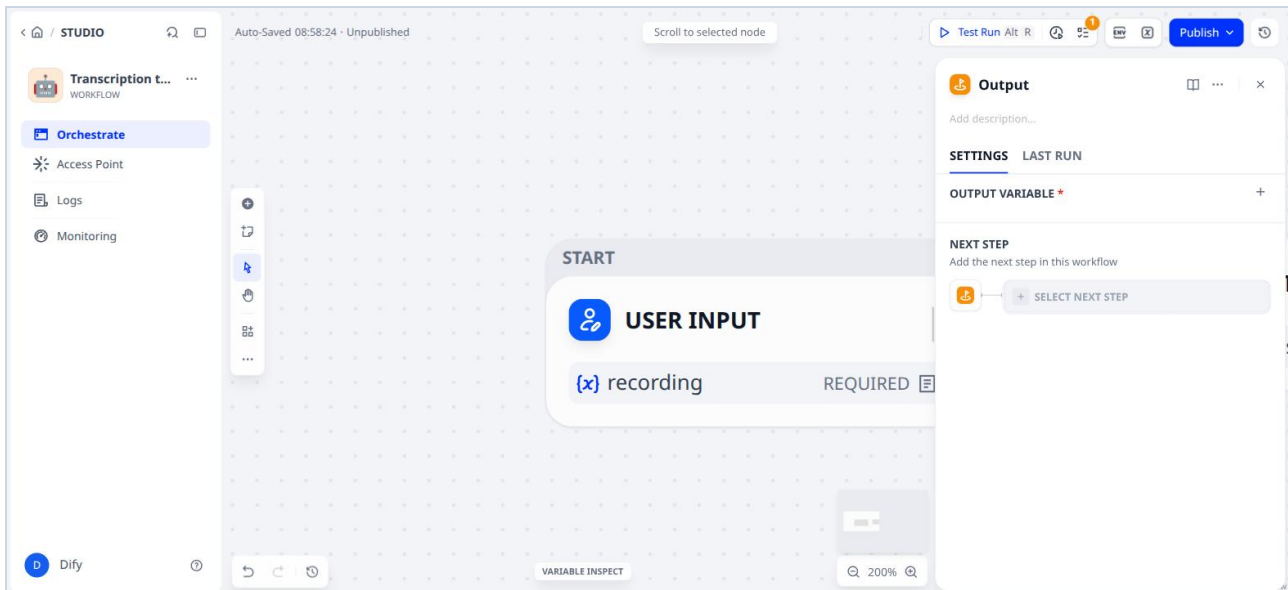
The Minutes node, complete

Your choice: the structure of the minutes. The headings in the prompt are the structure a department meeting often has. A staff council meeting may want "Advice and consent", a team day wants "Agreements" and "Parking lot", a student review wants a block per student (but that is the scenario that needs an impact assessment first). Change only the headings and their explanation; keep the rules above them (speech recognition errors, invent nothing). Want two versions, for example full minutes and a five-line message for the team? Add a second LLM node with the same variable **text** as input.

The rule “put a question mark in brackets after anything you are not sure of” is not decoration. In our test the speech recognition heard “Karin” once as “Karen” and once as “Karin”, and “Department meeting” once came out garbled. The minutes neatly added “(Karen?)”. A person still has to check that; the tool makes visible where.

Step 5. Decide what comes out

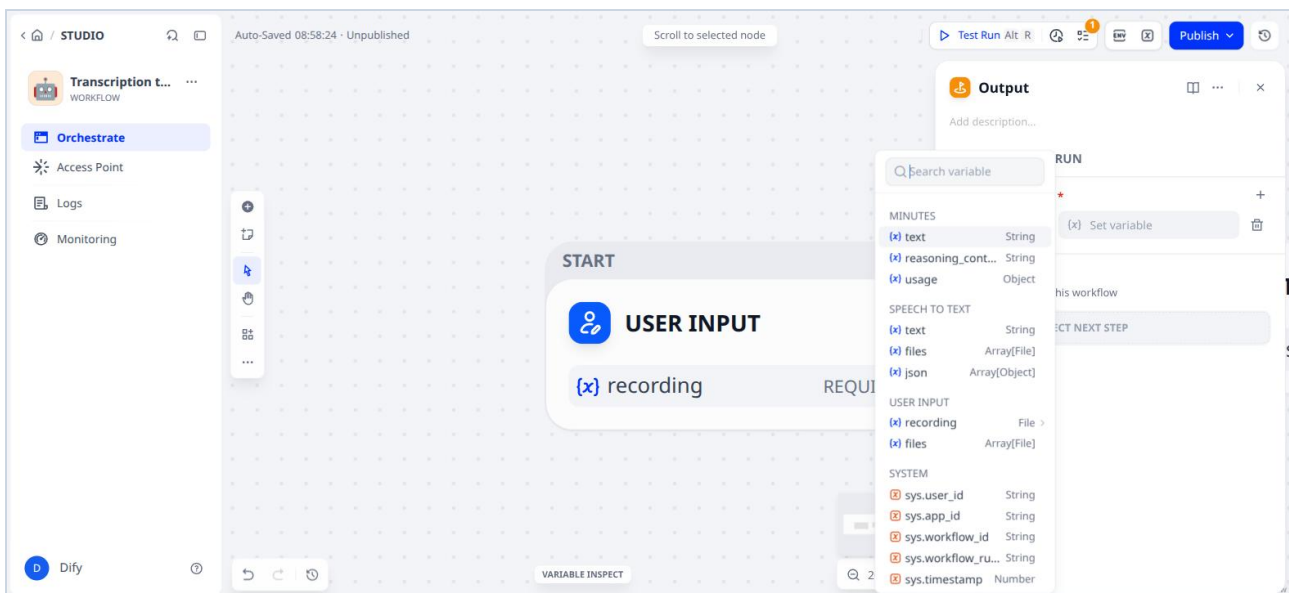
1. Click **SELECT NEXT STEP** and choose **Output**.



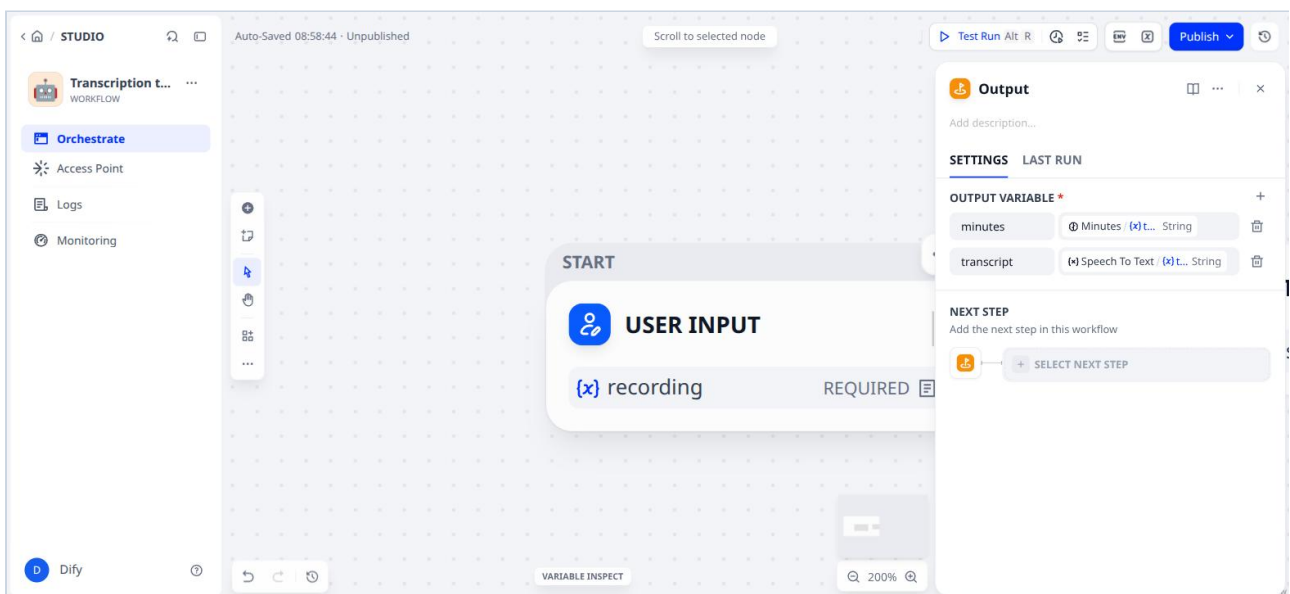
The empty Output node

2. Click the plus sign next to **OUTPUT VARIABLE** and create two variables.

Variable name (type)	Source (choose)
minutes	Minutes / text
transcript	Speech To Text / text

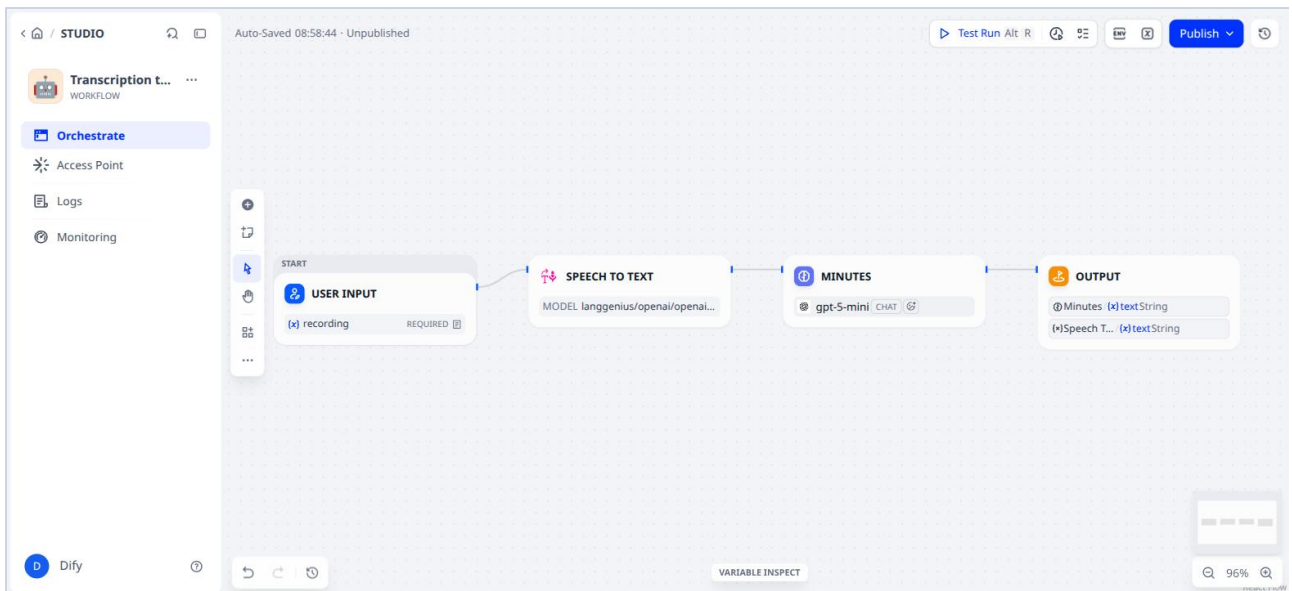


The source list



Both output variables

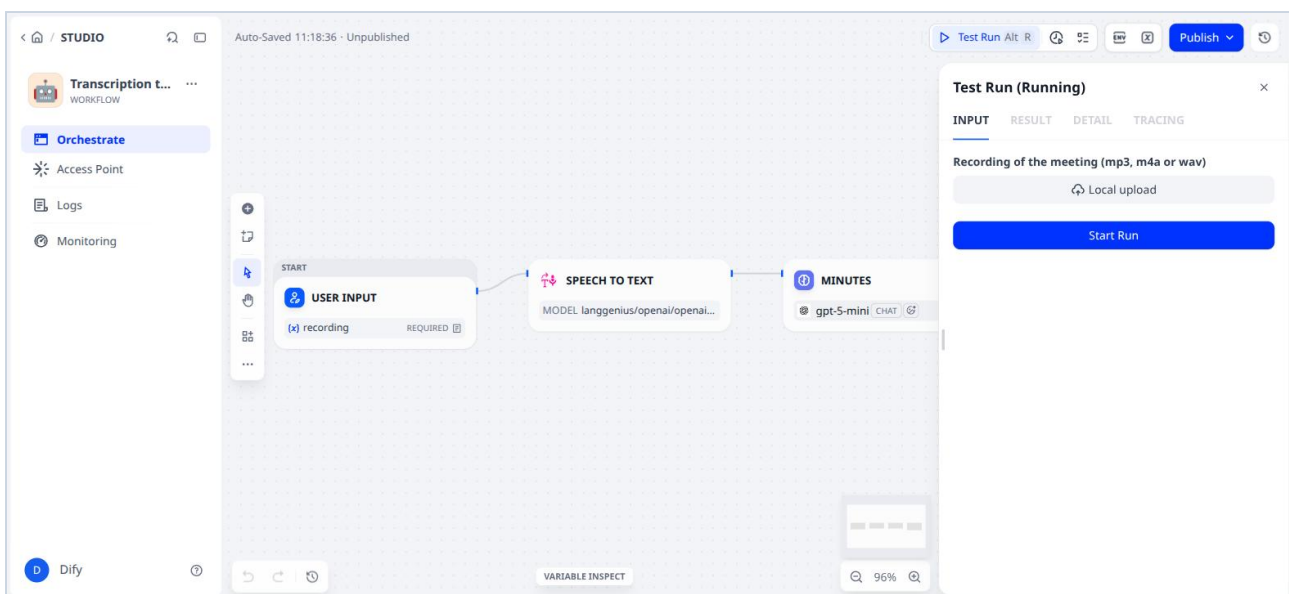
3. Close the panel and press Ctrl+1. You see the whole workflow: four nodes.



User Input, Speech To Text, Minutes, Output

Step 6. Test with the recording

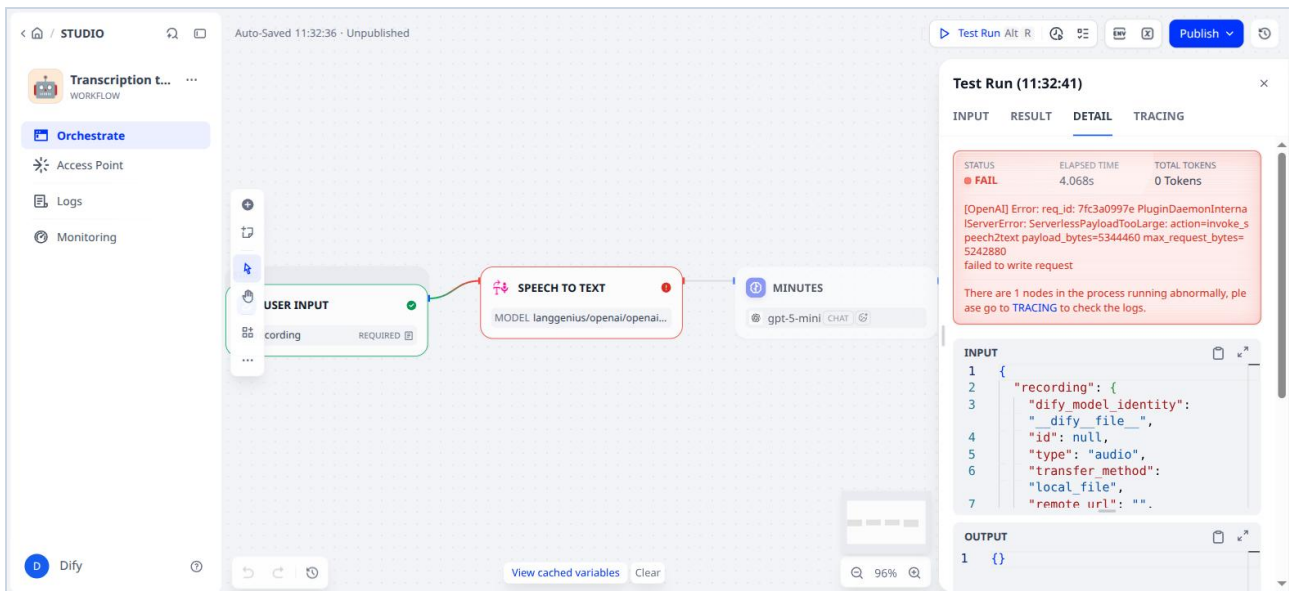
1. Click **Test Run** at the top right.



The test panel with the upload button

2. Click **Local upload** and choose your small test recording (department-meeting-small.mp3).

Use the file with “small” in its name. The full recording department-meeting.mp3 at 128 kbps is 2.7 MB for less than three minutes and that is too big: the tool sends audio as text (base64) to the provider, which doubles the size, and the limit is 5 MB (payload 5,344,460 bytes against a maximum of 5,242,880 in our test). You then get the error below.

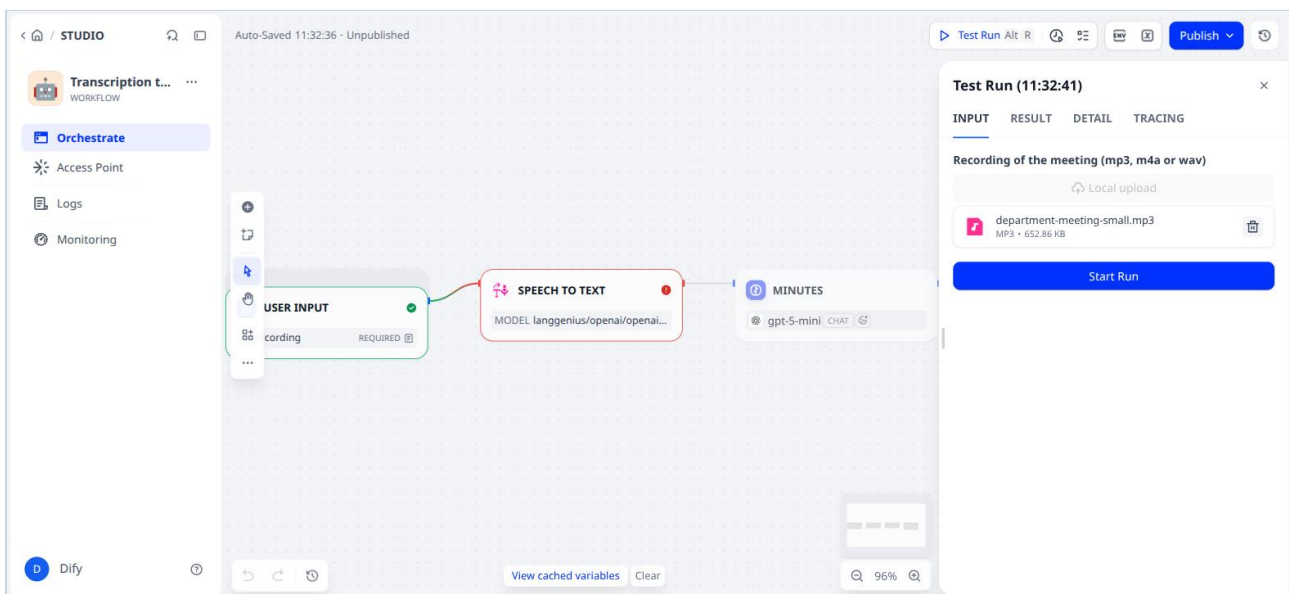


Too big: ServerlessPayloadTooLarge

Rule of thumb: **3 MB per recording**. Half an hour of meeting only fits if you shrink the recording first: mono, 16 kHz, 32 kbps gives about 250 kB per minute. Any audio program can do that, or ffmpeg:

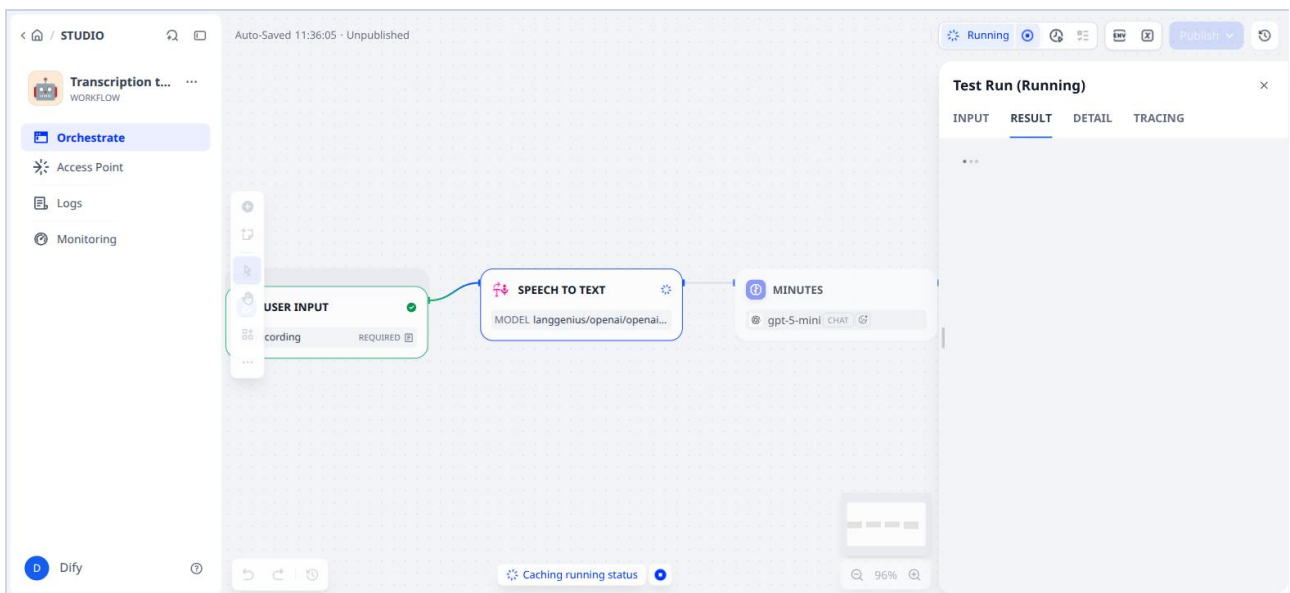
TYPE IN

```
ffmpeg -i recording.m4a -ac 1 -ar 16000 -b:a 32k recording-small.mp3
```

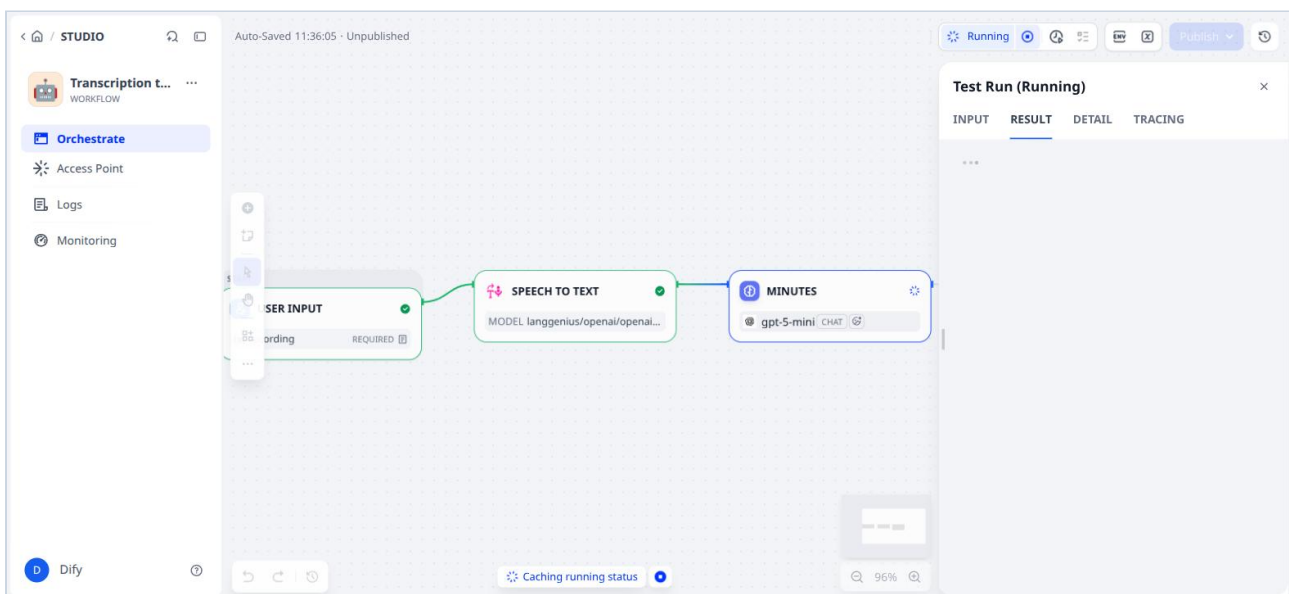


The small file is ready

3. Click **Start Run**. Transcribing three minutes of audio takes about a minute and a quarter; the minutes take another 45 seconds.

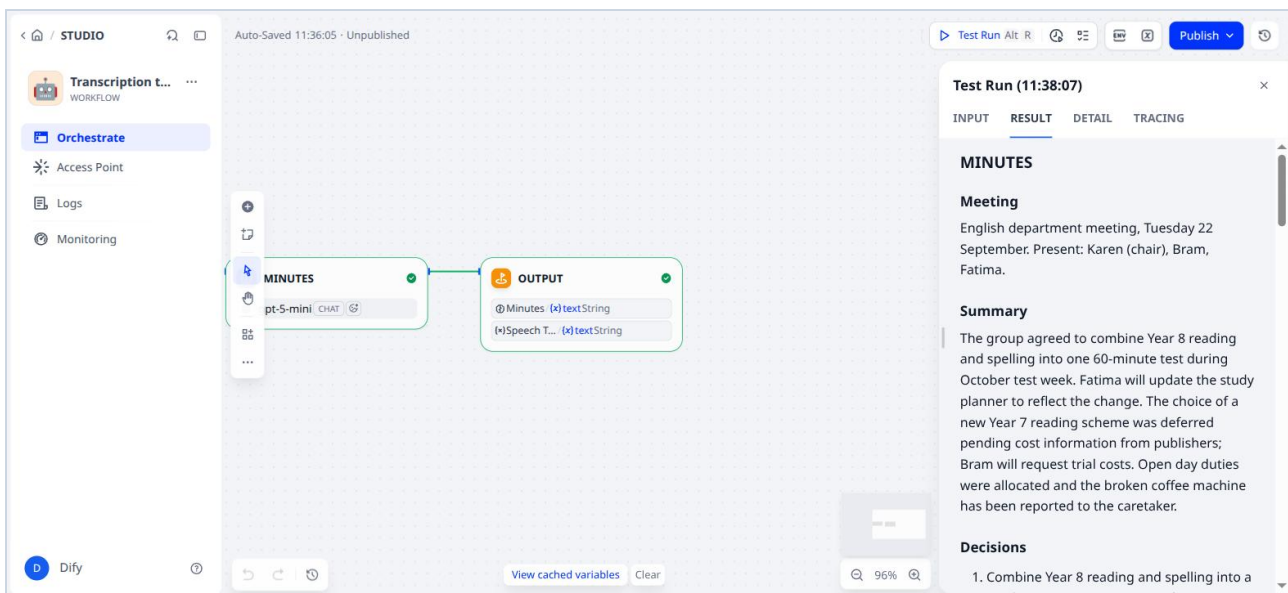


Speech To Text running

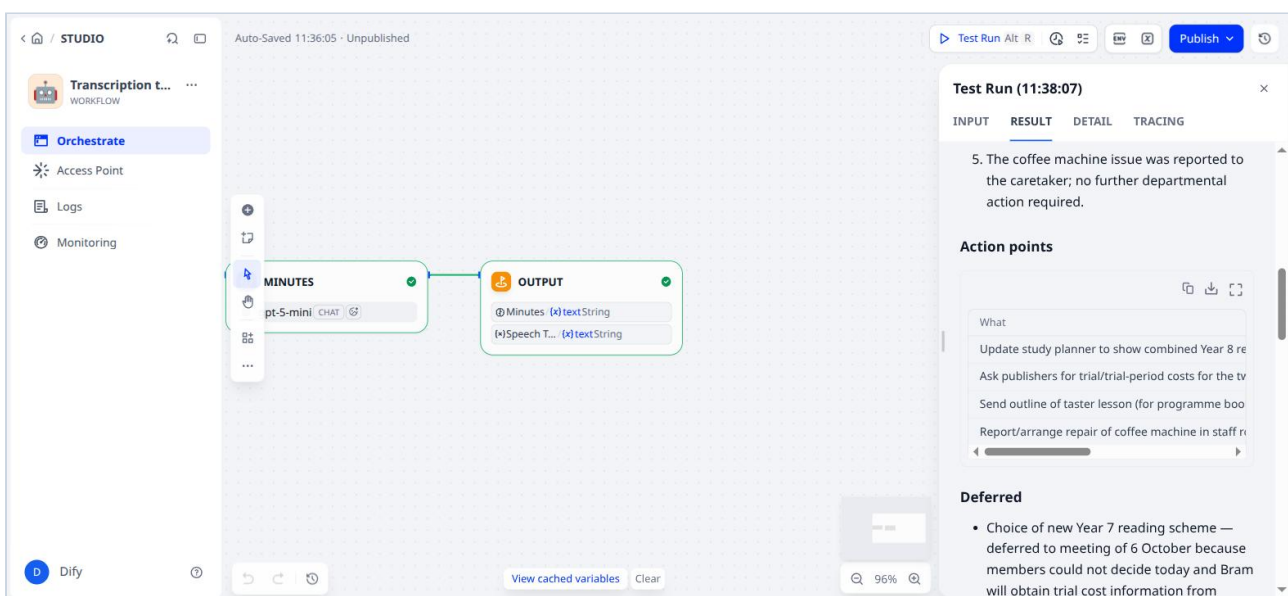


The Minutes node writing

4. Read the result under **RESULT**.



The minutes: meeting, summary, decisions



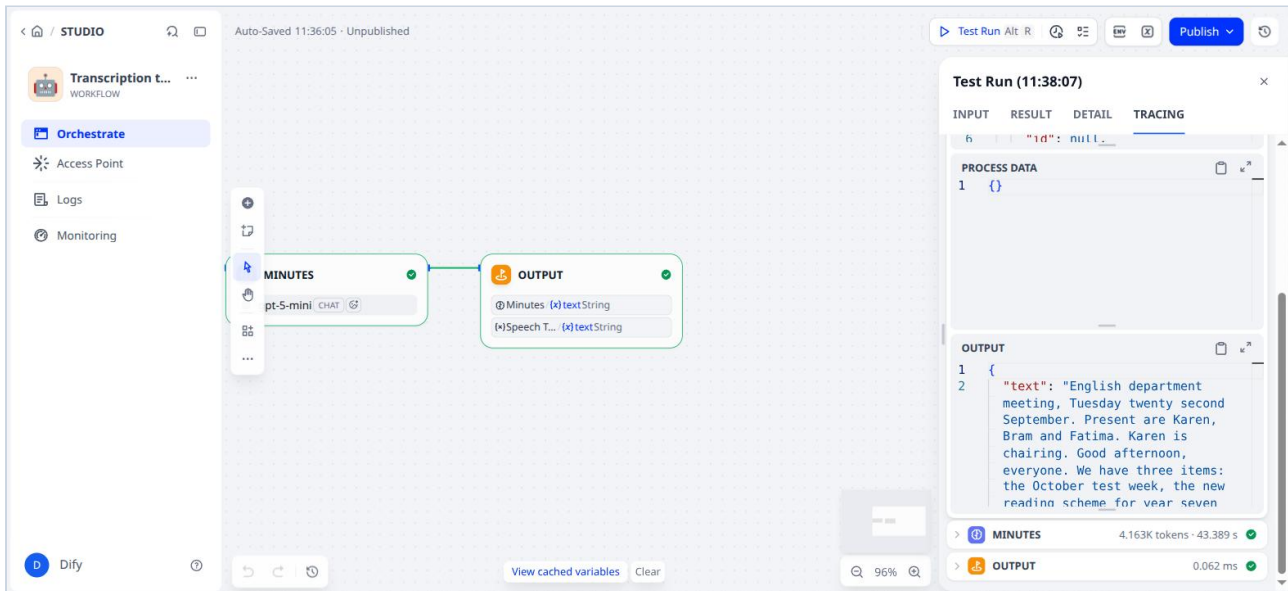
Action points, deferred, and below them the transcript

Check against the script (department-meeting-script.txt) whether it is right. In our test: the combined test and the deferral found, the action points with the right owners and dates (the publishers' email without a date, which the model marks "not mentioned"), the reading scheme under "Deferred" with the reason. Two things to notice. The model listed five "decisions" where the meeting took two; the open day roster and the coffee machine are not decisions. And the coffee machine, which the prompt says to leave out, appeared as an action point "not mentioned, not mentioned". A second run through the webapp (step 7) left it out. The prompt rule works most of the time, not always; the question marks and the "not mentioned" cells are the model saying "check this".

Recognising speakers: what the model does and does not do

The model is called gpt-4o-transcribe-**diarize**, and "diarisation" is telling speakers apart. To see what that gives you in Dify, look at what the tool returns.

5. In the test panel click the **TRACING** tab and expand **SPEECH TO TEXT**. Under OUTPUT is what the tool returns: one field `text`. Our recording has one voice for all three speakers, and still the text is split into paragraphs at the turns; with a recording of three different voices (`make_recording.py --three` makes one) there is a line break wherever another voice starts.



The output of Speech To Text: one field text, no names

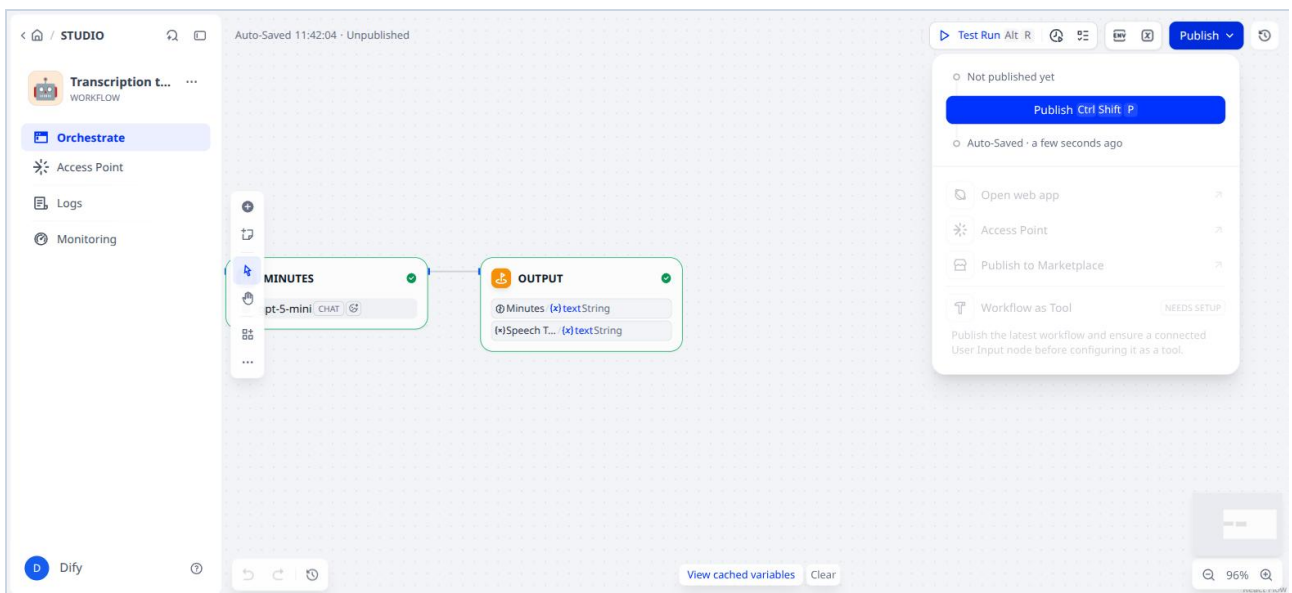
That is the honest answer: the model **splits** the text at speaker changes, but Dify's Speech To Text tool passes on **no labels** (Speaker 1, Speaker 2). Who said what is inferred by the Minutes node from the content, and that works well here because the opening names the people ("Present are Karen, Bram and Fatima") and because people address each other in a meeting. In a meeting without such an opening the action points end up as "not mentioned".

Two more things stood out in the Dutch edition of this test, where we did run three voices. The transcription of three computer voices contains more errors than that of one: the voices read with a slight accent and the speech model hears it. Real speakers do better. And the run time was the same, a good minute for three minutes of audio.

Your choice: real speaker labels. If you want "Karen:" and "Bram:" in the transcript, there are two routes. The easy one: put in the prompt of the Minutes node "The lines in the transcript are speaker changes. Put the speaker's name before each line if it is clear from the content, otherwise Speaker 1, 2, 3." The precise one: replace the Speech To Text tool with an **HTTP Request** node to OpenAI's speech API with `response_format=diarized_json`; that gives a speaker number and a time per sentence. That needs a key in the node (through an environment variable, never in the text) and is a next guide.

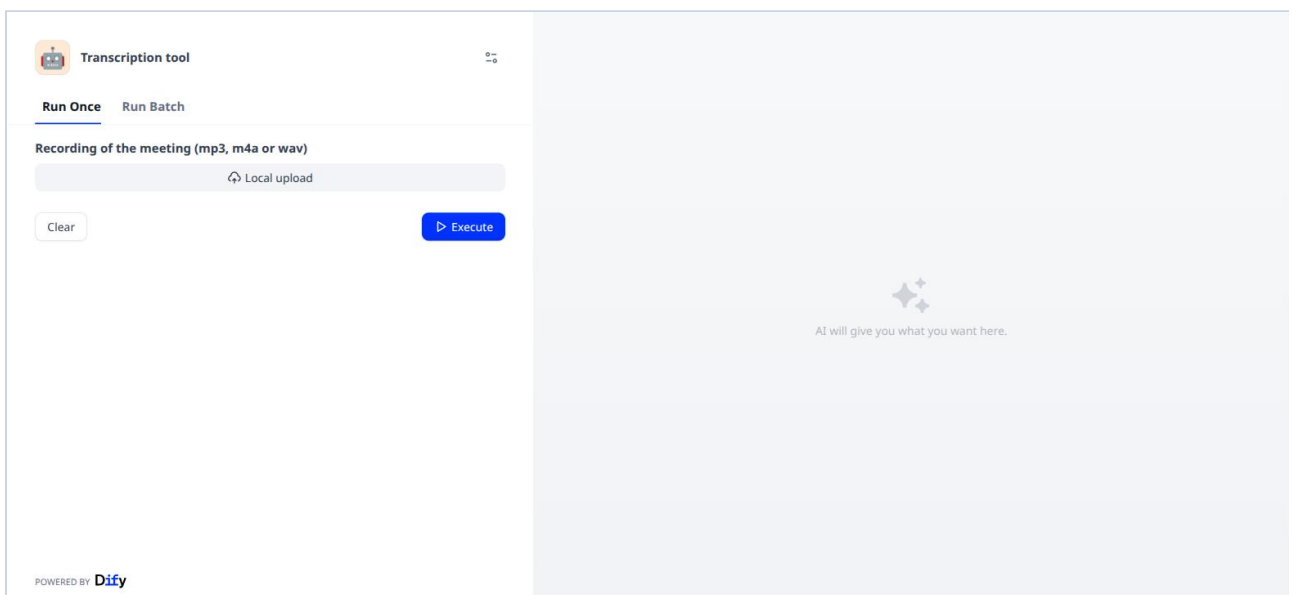
Step 7. Publish

1. Click **Publish** at the top right and then **Publish**.



The Publish menu

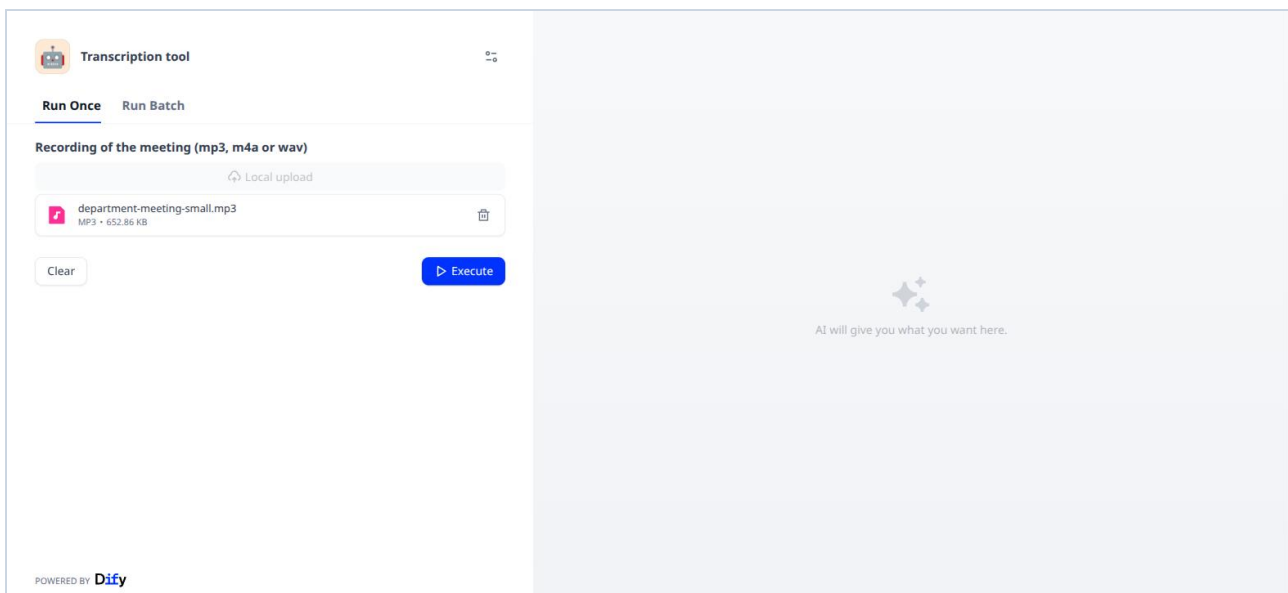
- Open the menu again and click **Open web app**. The web app has two tabs, **Run Once** and **Run Batch** (several recordings from a spreadsheet in one go).



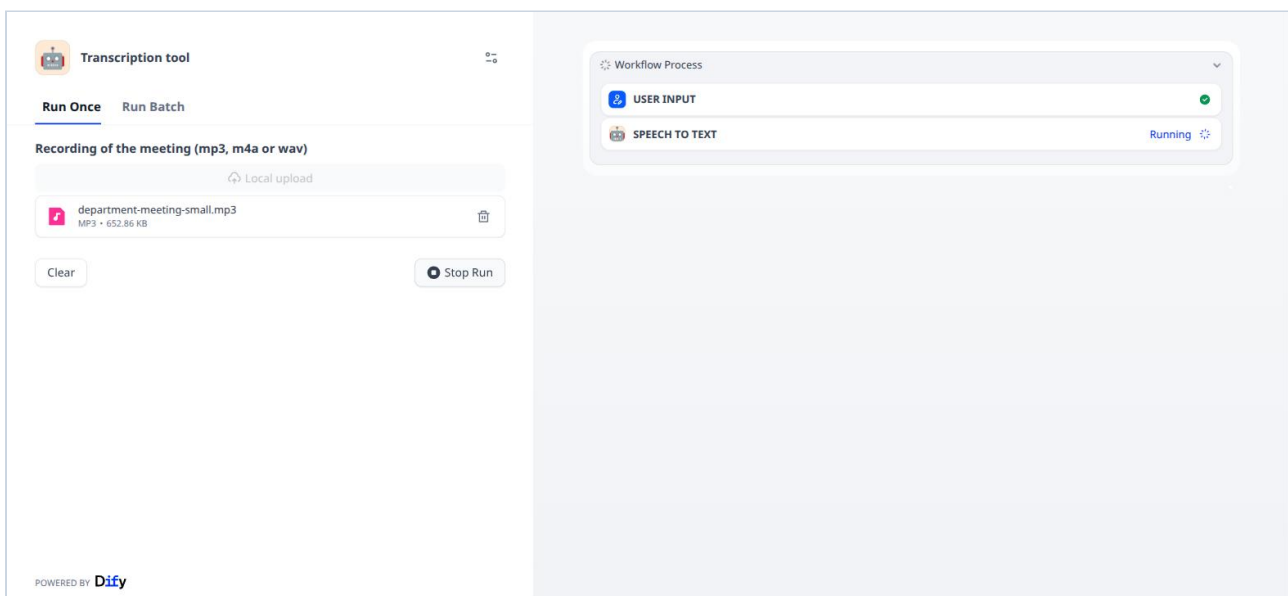
The web app

- Click **Local upload**, choose the small recording and click **Execute**.

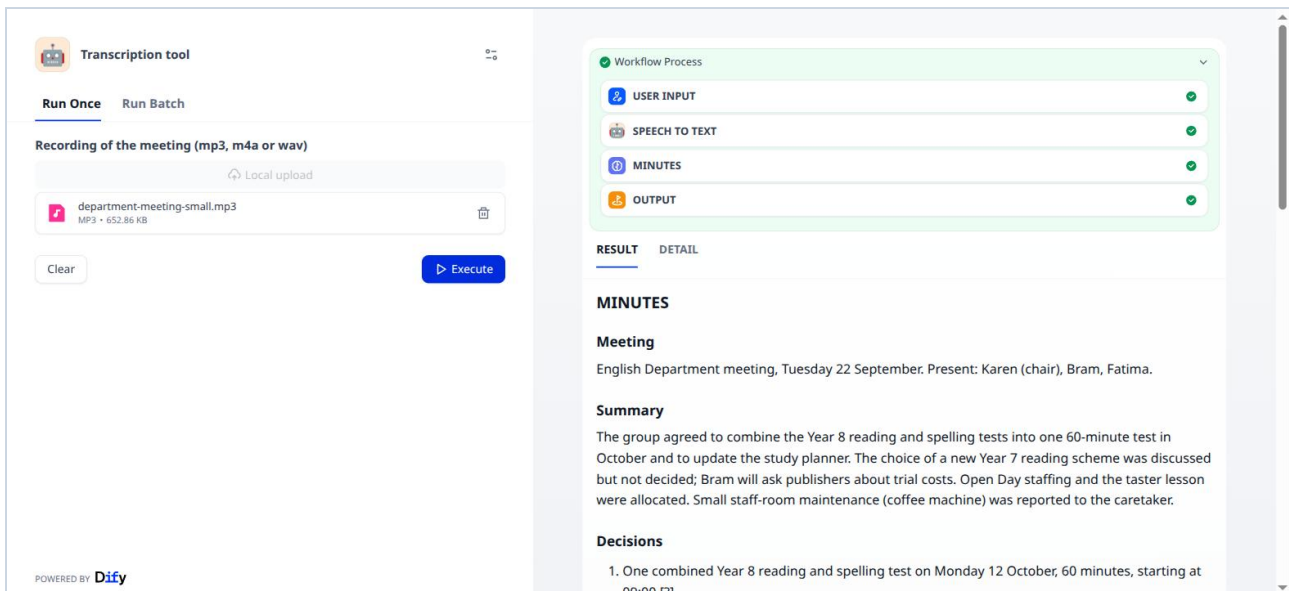
After the upload the Execute button moves down a little. Check where it is before you click.



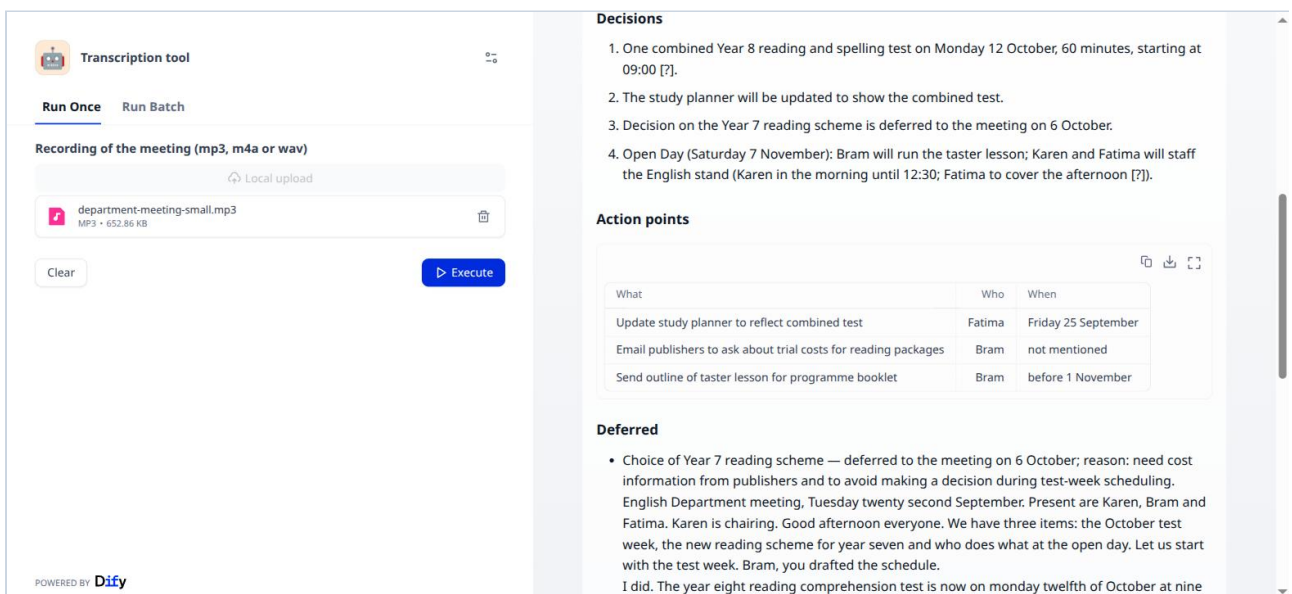
File uploaded, Execute is lower



The four steps running



The minutes in the webapp



The action point table, with copy and download buttons

At the top right of the table there is a copy button and a download button. With those a colleague puts the action points straight into an email or a spreadsheet.

The example from this guide: <https://udify.app/workflow/EPAc3PPeD3E4764I>.

Checking that it worked

- The workflow has four nodes: User Input, Speech To Text, Minutes, Output
- The input field accepts only audio (Image not ticked)
- Test Run with the small recording gives minutes with five headings and below them the transcript
- The webapp gives the same result, with MINUTES as heading

- Uncertain names have a question mark in brackets

When things go wrong

What you see	What is going on
<code>ServerlessPayloadTooLarge ... max_request_bytes=5242880</code>	The audio file is too big. Shrink to 32 kbps mono (step 6)
Speech To Text: no model in the list	No OpenAI key of your own, or Usage priority is set to AI credits. Integrations → Model Provider → OpenAI → Config
The workflow gets stuck at Speech To Text with a photo	Image is still on in the input field (step 2)
The minutes mention people who were not in the meeting	The speech recognition misheard a name. Look in the transcript at the bottom; the right name is often there with a question mark
Minutes and transcript run into each other	The line “Start your answer with the line # MINUTES” is missing from the instruction
“Upgrade to create more apps”	Sandbox limit of five apps. Delete an old app via the three dots on its card
An mp3 link from a test run gives <code>403 forbidden</code>	Download links for Dify files are valid for five minutes. Download straight away, or run the test again
The transcript has no speaker names, only line breaks	That is how the Speech To Text tool works: diarize splits but does not label in Dify. See Recognising speakers at step 6
A message about temperature	The model does not support that parameter. Ignore it

Make it yours

The recording is invented and the structure of the minutes is our choice. This chapter helps you rebuild the tool for your meetings, within the privacy frame at the start of this guide. From easy to hard.

Change	Where	Difficulty	What you get for it
Other headings in the minutes	Step 4, prompt	easy	Minutes in the shape you already use
Language and tone (formal, informal, another language)	Step 4, prompt	easy	Minutes you can share without rewriting
A different speech model (whisper-1 is cheaper)	Step 3, settings of the tool	easy	Lower cost; you lose the line breaks at speaker changes
Speaker names in the transcript	Step 6, prompt of Minutes, or an HTTP Request node (see the green box under Recognising speakers)	medium	A transcript with "Karen:" and "Bram:" in front
A second output: a short message for those who were absent	Extra LLM node after step 4, extra output variable in step 5	medium	Two products from one recording
Action points as a separate list you can copy to a task list	Extra LLM node with only the table	medium	Less retyping after the meeting
An input field "type of meeting" that chooses the structure	Step 2 plus an IF/ELSE node or a prompt with conditions	medium	One tool for several kinds of meeting
Recordings longer than 5 MB	Replace Speech To Text with an HTTP Request node to the speech API	hard	An hour of meeting in one go; see Housekeeping
Running on your school's approved platform	Export DSL, import, reconnect the models	hard	The route that is compulsory for real recordings

Three assignments

1. **Your structure.** Take the last minutes someone in your team wrote by hand. Put its headings in the prompt of step 4, with one sentence of explanation per heading. Run the fictional recording again and compare with the handwritten minutes: what is missing, what is too much?
2. **Make your own test recording.** Write a fictional ten-minute script with your team about a real topic (without real names), and turn it into a recording via appendix A or `make_recording.py`. That way you test the tool on your jargon without a real voice ever entering Dify.
3. **Prepare the move.** Export the workflow (Export DSL) and write on half a page what is needed to run it on your school's platform: which model, which key, who the owner is, how

long audio is kept. That half page is exactly what a data protection review asks of you.

Checklist before you use your own material

- [] No real recording goes into Dify cloud; only fictional or synthetic audio
 - [] For real recordings: the tool runs on your school's approved platform, and the participants have consented
 - [] Audio is deleted after transcription and it is recorded who checks that
 - [] The scenario with students (student reviews, conversations with students) waits for the impact assessment
 - [] Someone reads the minutes before they go round; the question marks in the minutes are there for that
 - [] There is one owner of the tool who maintains the prompt and republishes
-

Housekeeping

What Dify keeps. Every upload and every result is in the app's **Logs**, with the audio file attached. For the prototype with the fictional recording that is no problem. As soon as a real recording would ever go in, this is the reason Dify cloud is not allowed: you cannot delete the audio without wiping the whole log, and a retention policy (audio gone after transcription) cannot be honoured. For real use the tool belongs on your school's platform, with a retention period that is written down.

Costs. Through your own key: gpt-4o-transcribe-diarize costs about 0.6 cents per minute of audio, whisper-1 half that. A 45-minute meeting comes to 20 to 30 cents including the minutes. Dify credits are not touched as long as OpenAI is set to API key.

Longer recordings. The tool's 5 MB limit is hard. An hour of meeting at 32 kbps is 14 MB and so does not fit. Two routes: cut the recording into fifteen-minute pieces and run them separately, or replace the Speech To Text tool with an HTTP Request node that goes straight to the provider's speech API (limit 25 MB). The latter is a next guide.

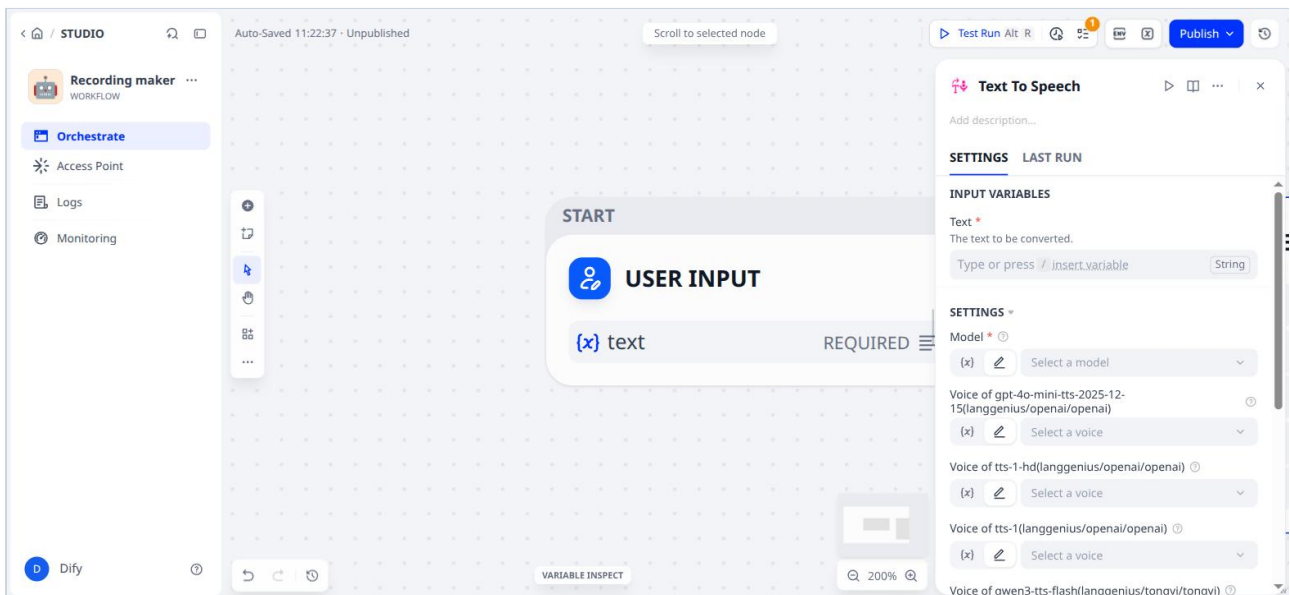
From prototype to your school's platform. The workflow can be saved as a file via **Export DSL** (three dots on the card in Studio) and imported into another Dify account. The models have to be reconnected there to the keys of that account.

Appendix A. Making a test recording without real voices

For a prototype you need a recording without personal data. The easiest route is to have a fictional script read out by the same Audio tool, but the other way round: **Text To Speech**. That is how `department-meeting.mp3` for this guide was made (the script `make_recording.py` does the same from the command line, with your own OpenAI key).

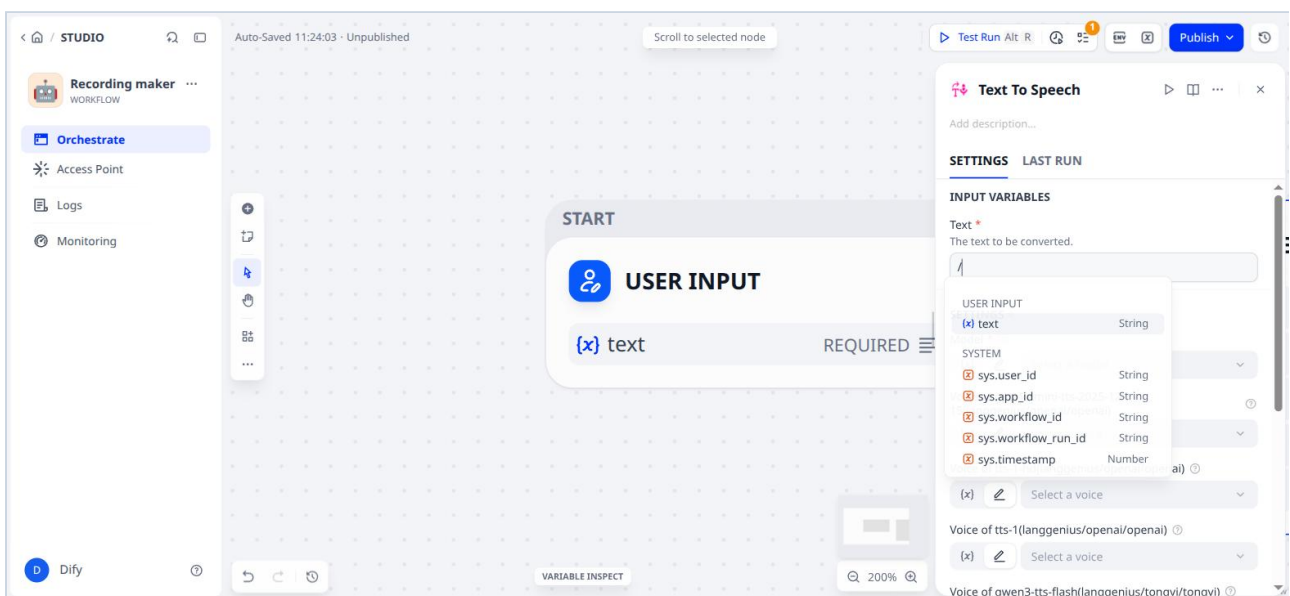
1. Make a workflow called `Recording maker` with a **User Input** with one field: type **Paragraph**, variable `text`.

2. Choose as next step **Tools** → **Audio** → **Text To Speech**.



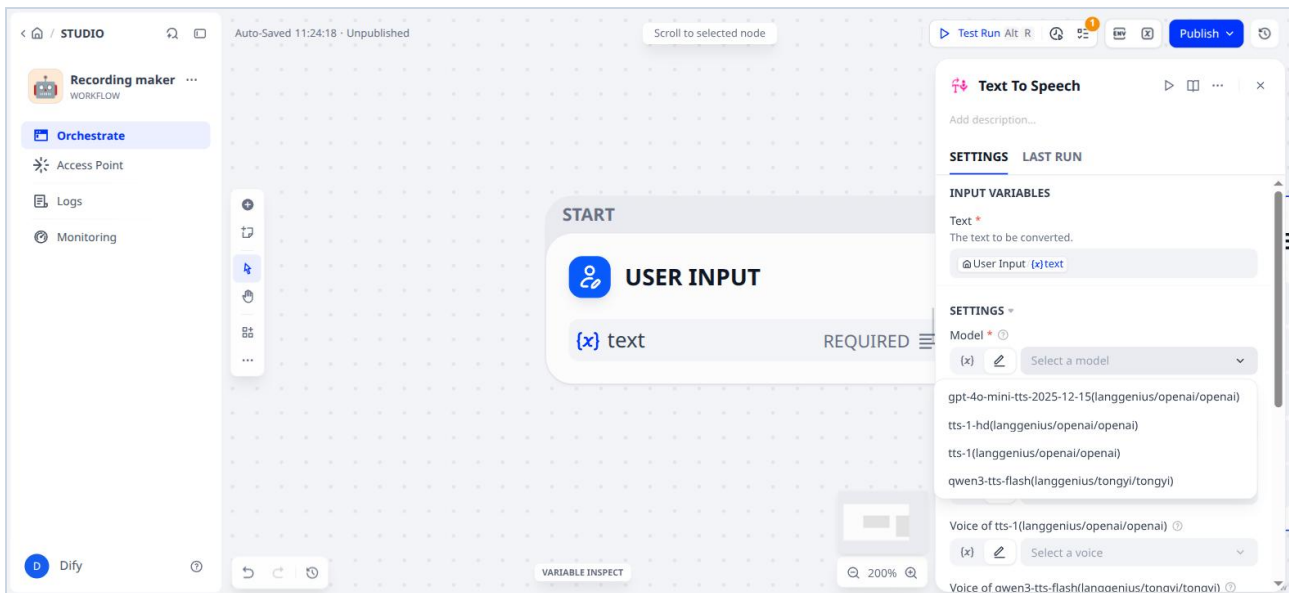
The Text To Speech node

3. Under **Text** click in the field, type **/** and choose **text**.



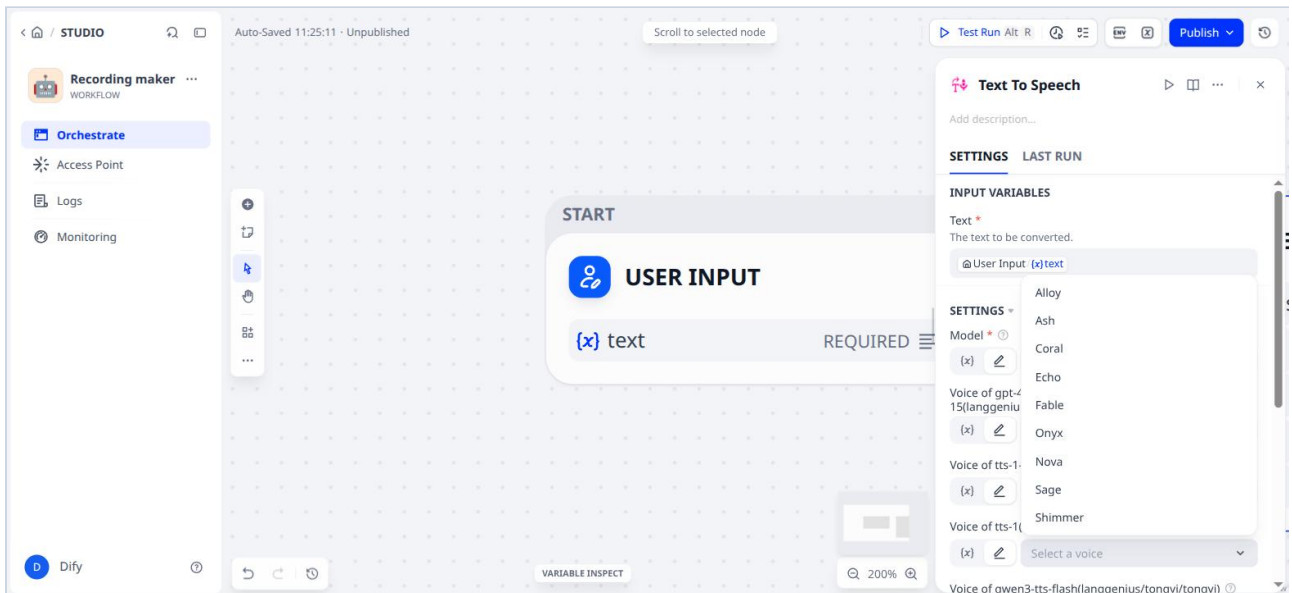
Choosing the variable

4. Under **Model** choose a speech model; **tts-1** is the cheapest.



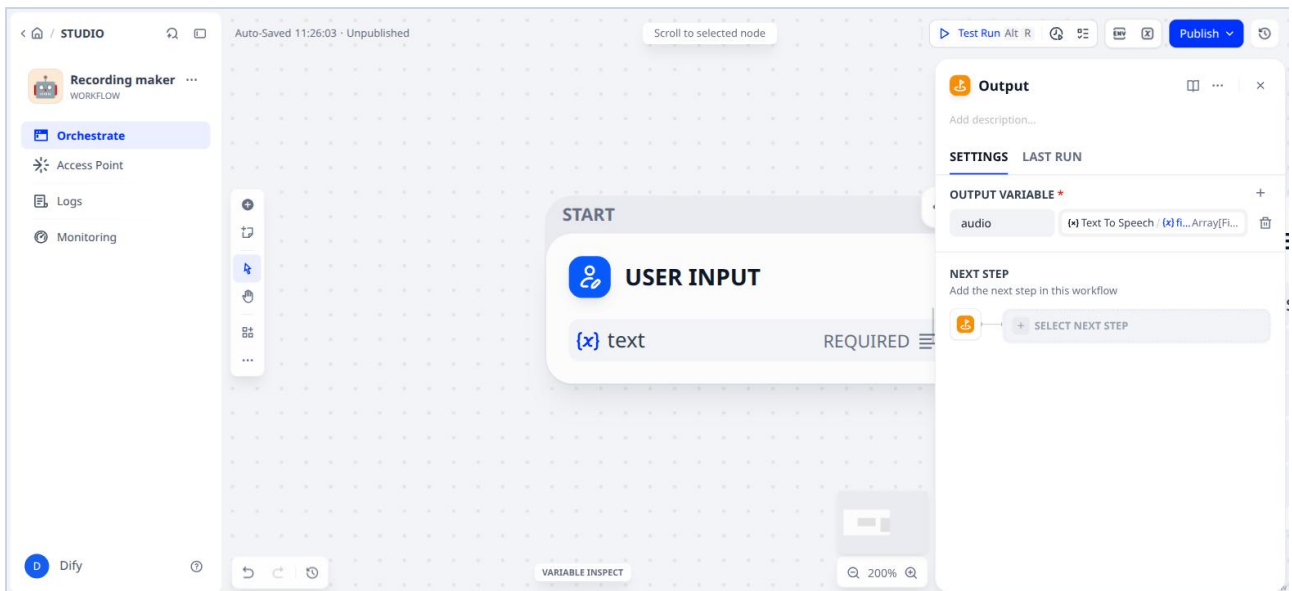
The speech models

5. Under **Voice of tts-1** choose a voice, for example **Alloy**.



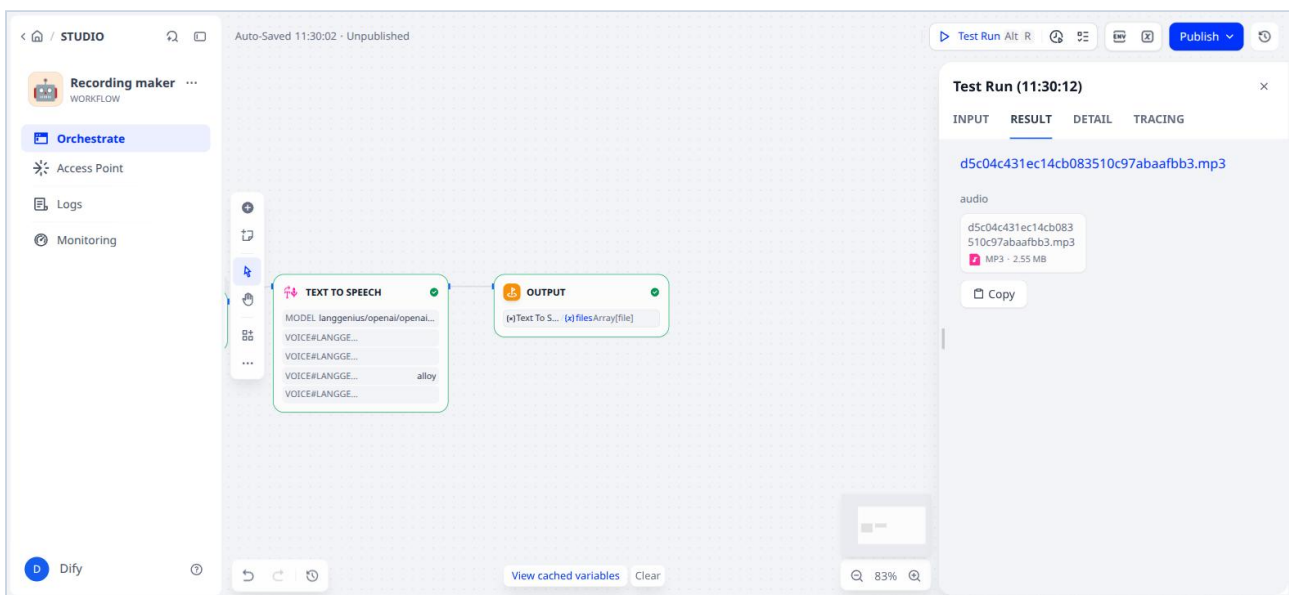
The voices

6. Add an **Output** node with variable `audio` → Text To Speech / **files**.



The Output node with the audio file

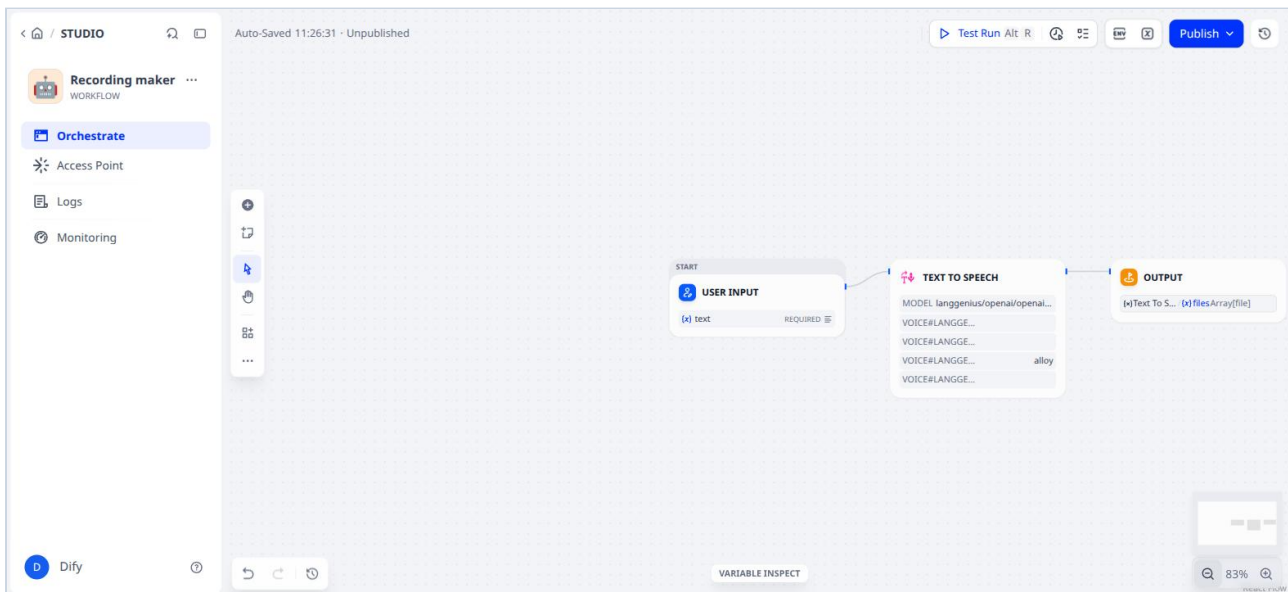
- Click **Test Run**, paste the script (department-meeting-script.txt) and click **Start Run**. After half a minute an mp3 is in the result: 2.55 MB, 128 kbps, mono, 2 min 47 s.



The recording is ready

- Click the file name straight away and save the file. The link works for five minutes. The file is too big for the Speech To Text tool (step 6); shrink it first with ffmpeg as shown there, or with any audio program, to 32 kbps mono.

Cost: about 4 cents for three minutes of speech. The result is one voice reading all the speakers; for a test of transcription and minutes that is enough. A sandbox account has room for five apps, so delete this helper workflow afterwards, or rebuild it into the Transcription tool as done in this guide.



The helper workflow: User Input, Text To Speech, Output

Licence: CC BY-SA 4.0 (<https://creativecommons.org/licenses/by-sa/4.0/>). You may copy, adapt and reuse this document, including at your own school, provided you credit the source (Guido van Dijk, LeX Consultancy B.V., <https://git.lexconsultancy.nl/guido/dify-guides>) and share your adaptation under the same licence. The LeX Consultancy logo is excluded; product names are trademarks of their owners. Schools, people and documents in the examples are fictional.

Owner and contact: Guido van Dijk, LeX Consultancy B.V. · guido@l-e-x.nl



LeX Consultancy B.V. · l-e-x.nl · Building in Dify: from recording to minutes