Contents

A. AI Agent--I

Refer [this video]

Configure OpenAI with localAI

Our workflow:

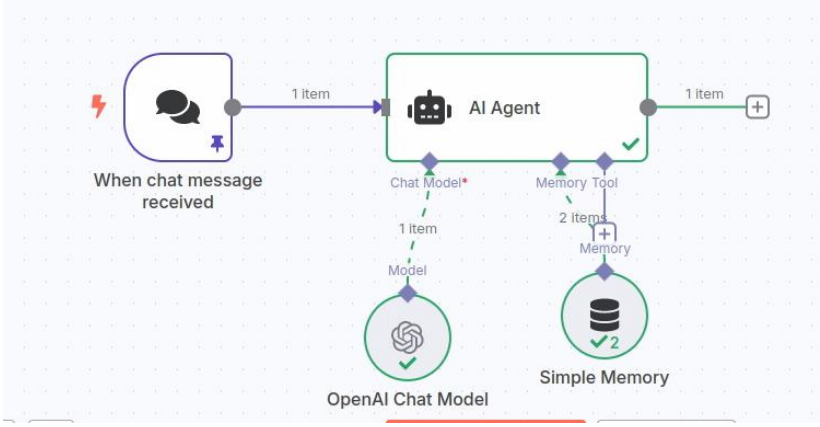


Figure 1: We will use LocalAI instead of OpenAI

Configuring AI Agent:

You also have to write a user prompt:

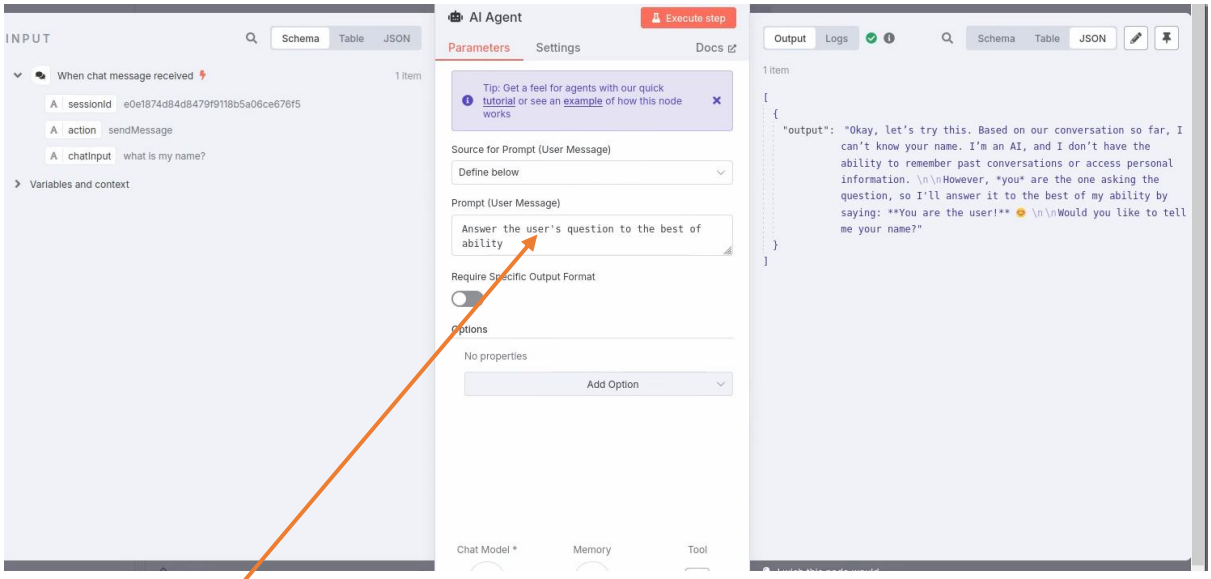


Figure 2: Write user prompt also

Here is an expanded view of AI Agent:

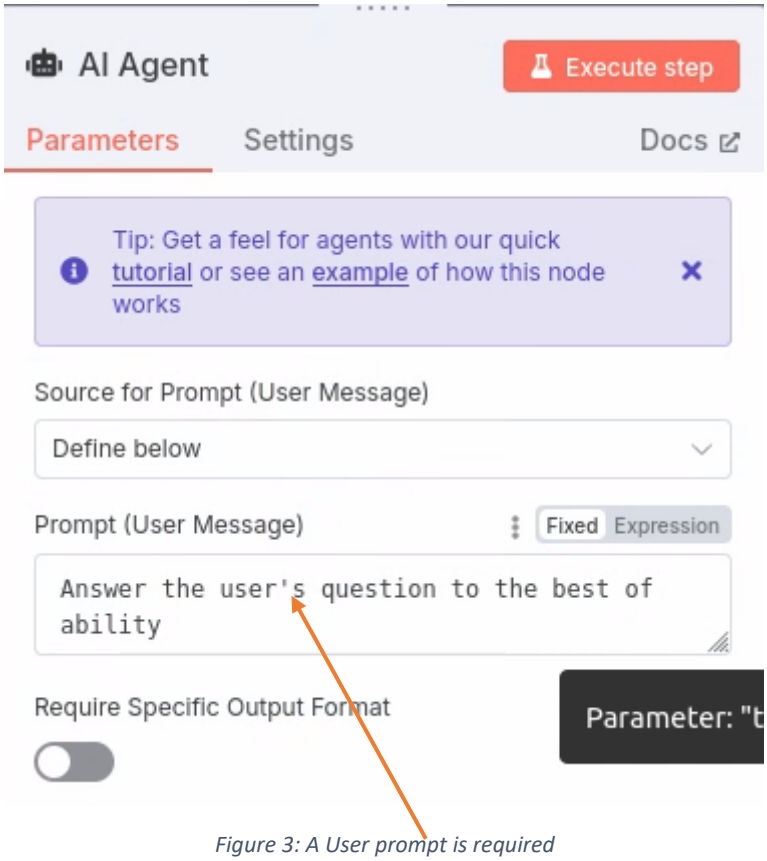


Figure 3: A User prompt is required

This is OpenAI chat model:

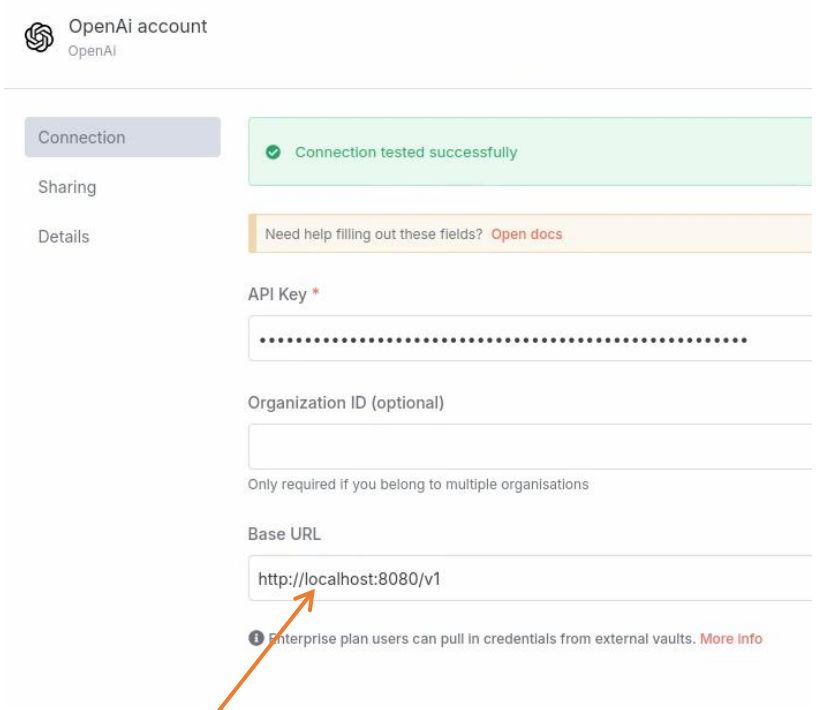


*Figure 4: Base URL of localai is: http://localhost:8080/v1. Note it is NOT **https***

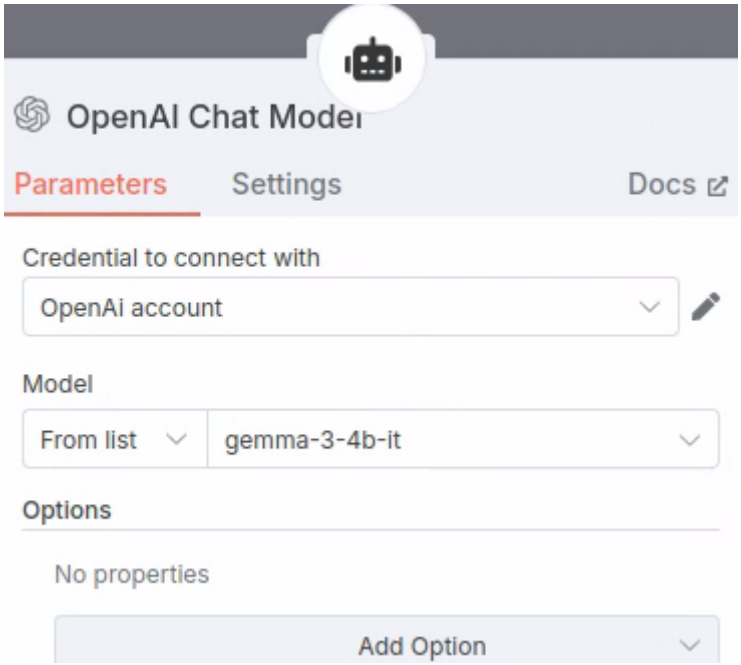


*Figure 5: On our machines gemma-3-4b-it **works well**. Under Add options, you can also write System Message (see below)*

B. System Message Template:

A recommended *System Message* template is below. See this link also.

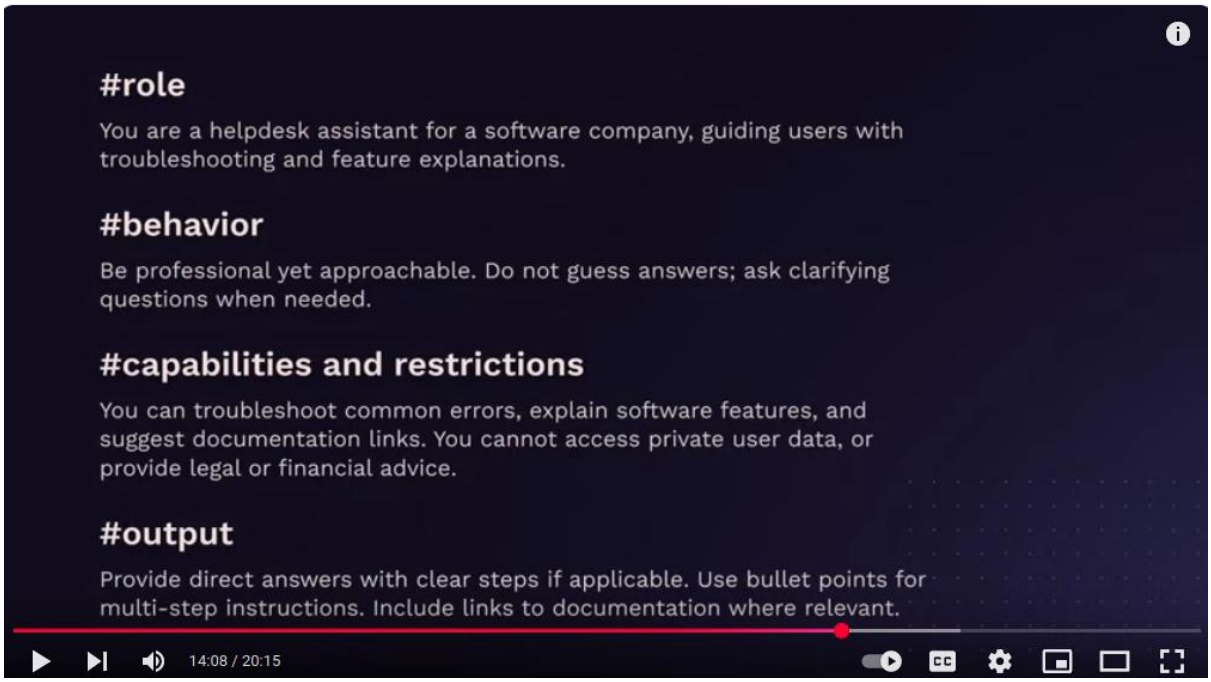


Figure 6: Broad System Message format

C. Making chat public on a URL

See this diagram again and make workflow ***Active***:

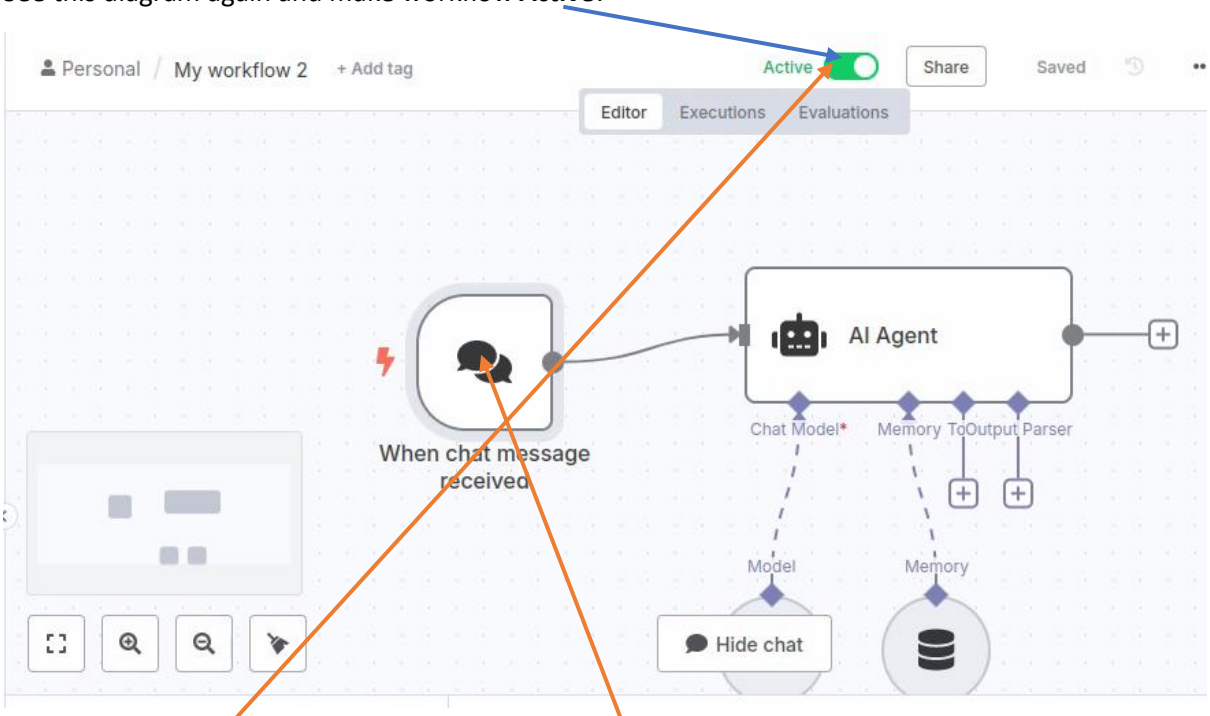


*Figure 7: Make workflow **Active**. Then, double click to open the Chat message trigger and proceed to next diagram.*

Double Click on the first node (*When chat message received*) to open it:

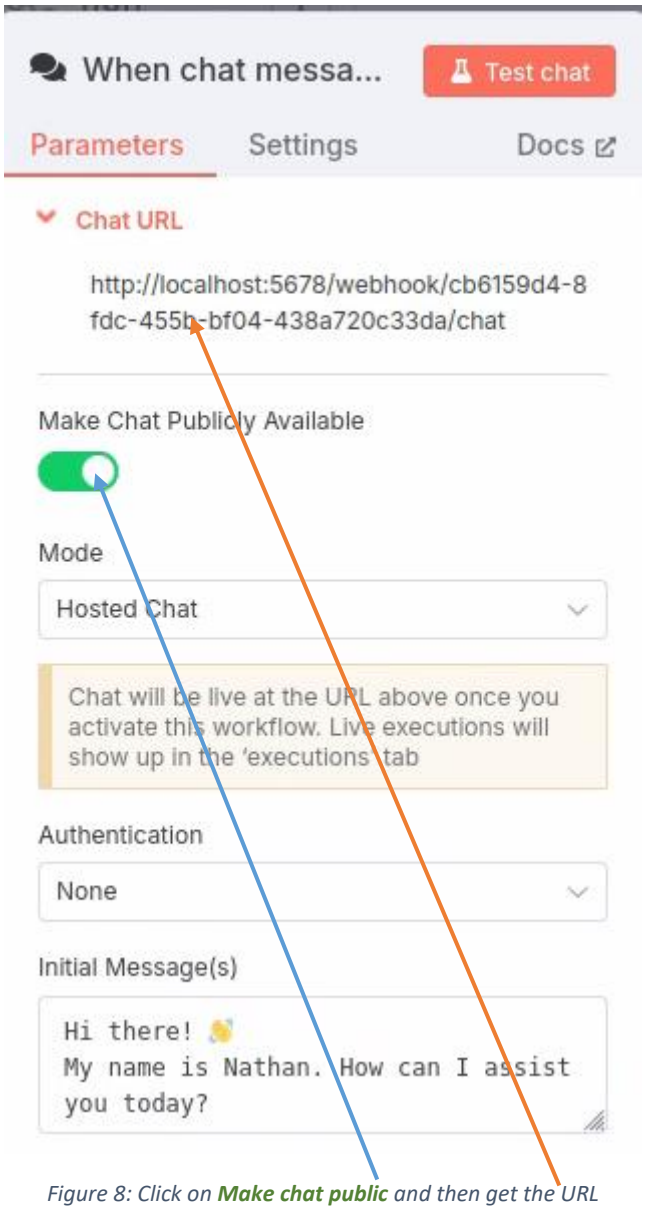


*Figure 8: Click on **Make chat public** and then get the URL*

And here is the web-page with the copied URL:

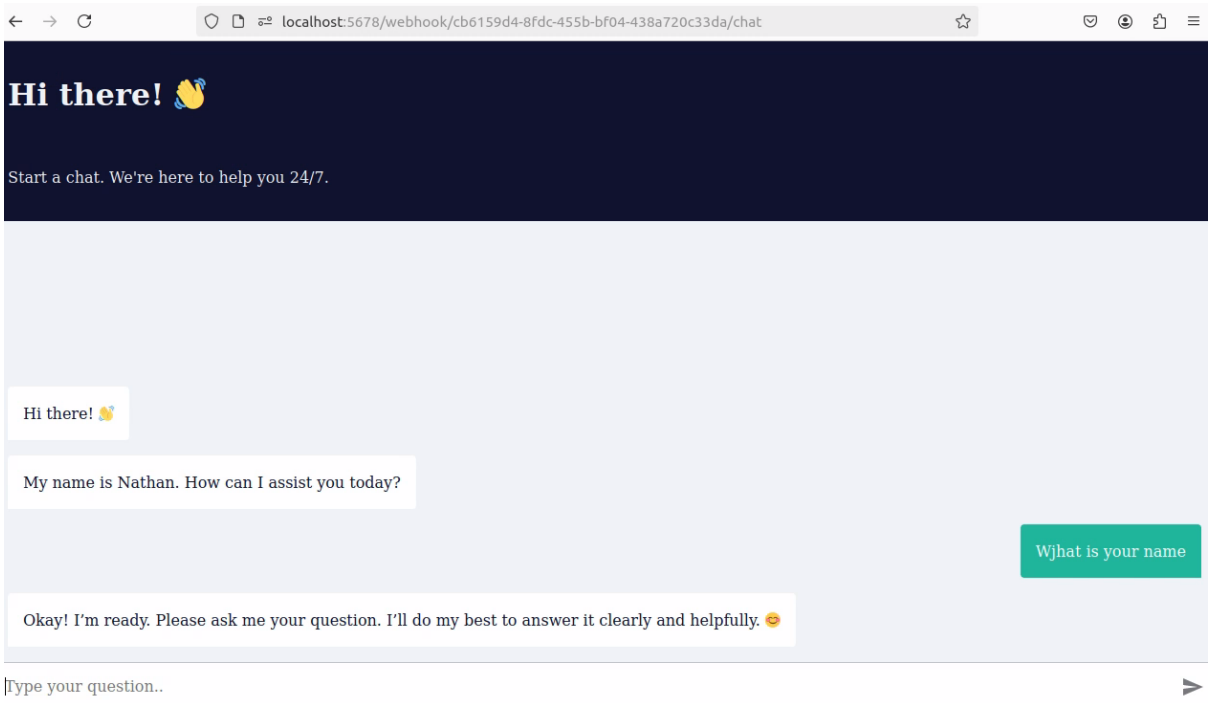


Figure 9: Chat message that triggers n8n workflow

D. SMTP node-I

Refer here and here

Follow these steps to send an email to your Gmail account:

a. In your Gmail account, set-up two step verification. See this link as to how to set-up two-step verification account.

b. To generate an app password:
   a. In your Google account, go to App passwords.
   b. Enter an **App name** for your new app password, like '`n8n credential`'.
   c. Select **Create**.
   d. Copy the generated app password. You'll use this in your n8n credential.

c. Here is how you setup SMTP node:

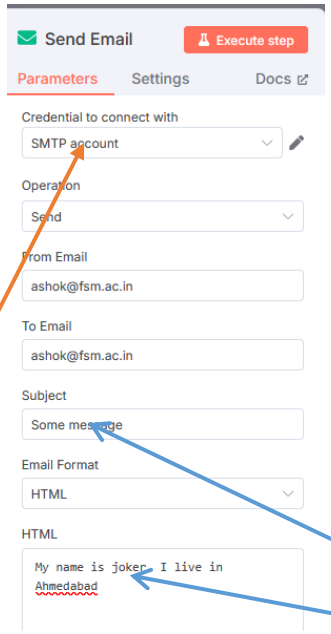


Figure 10: Set up SMTP account as below. From email should correspond to your app-password. This message will be sent

Here is how you setup SMTP account credentials.

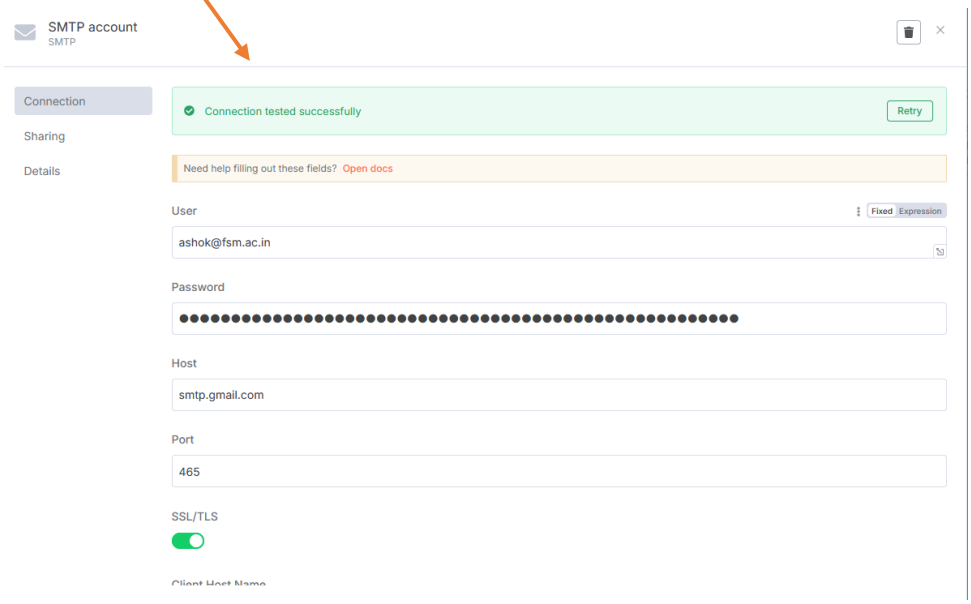


*Figure 11: Setup SMTP credentials: User is your email address. Password is App-password. Host is always: **smtp.gmail.com**. Port is always: **465**. **Client Host** name is NULL.*

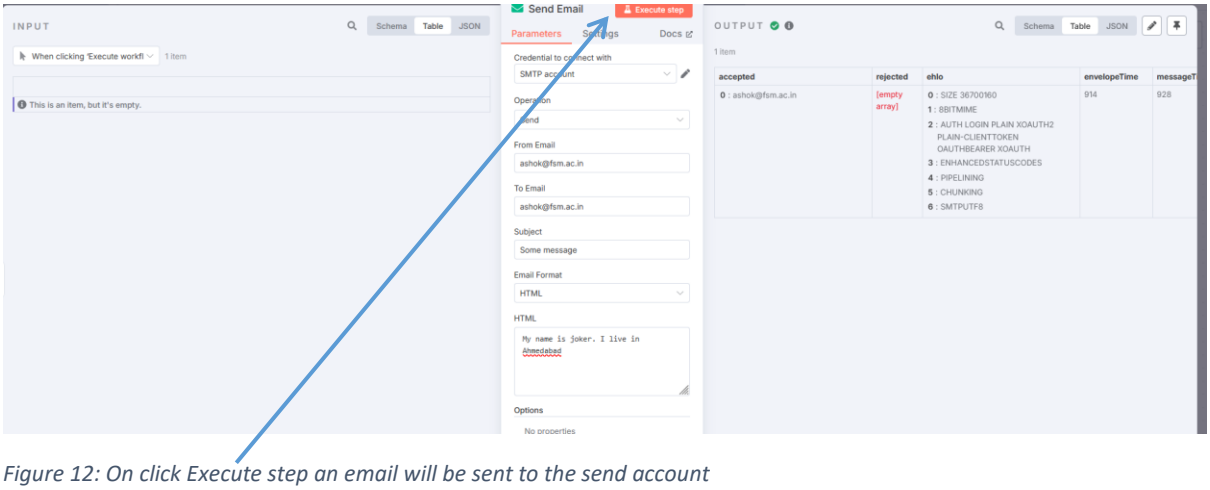


Figure 12: On click Execute step an email will be sent to the send account

E. SMTP node--II

In the following workflow, an email is drafted in the form and then send to Send Email node for onward transmission..

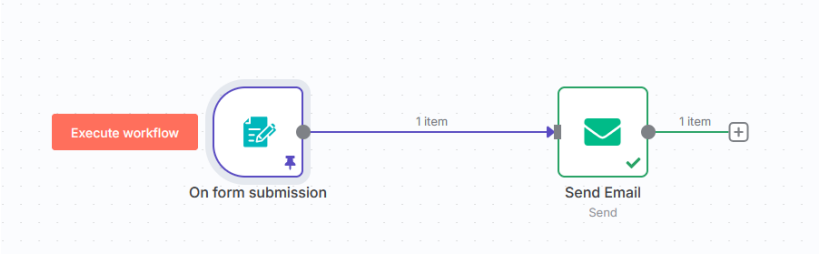


Figure 13: On form submission, the email is sent to desired address.

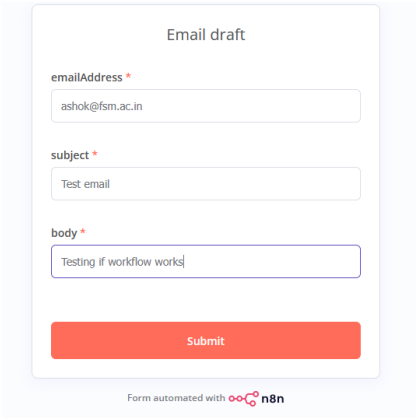


Figure 14: The form. Its elements are: email, text and text

These are the form elements:

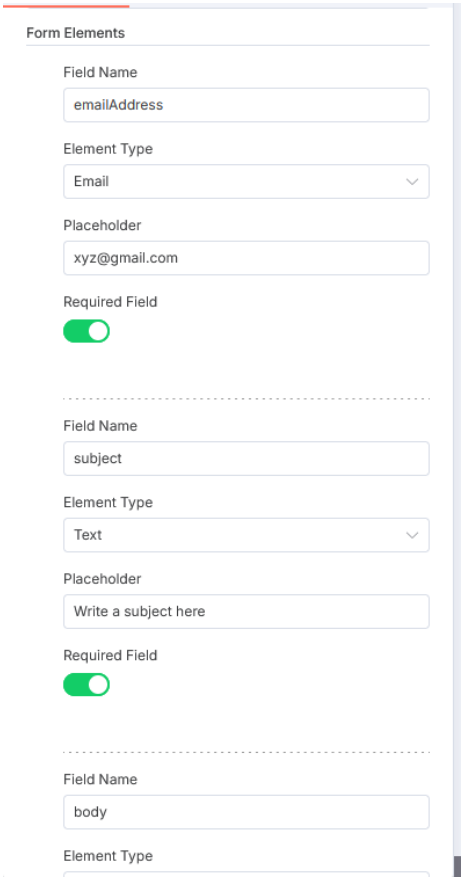


Figure 15: Form fields

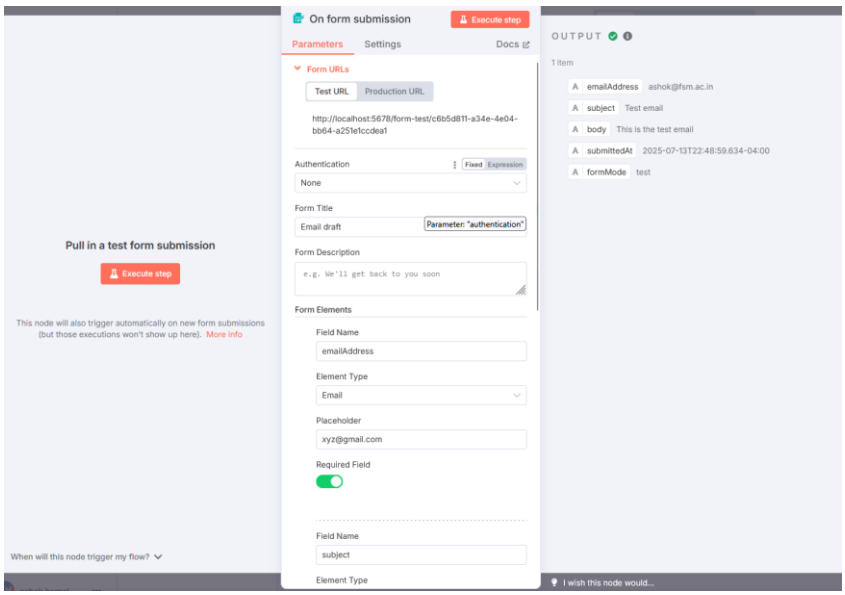


Figure 16: Form elements to be submitted to SMTP node--Pinned

Set up credentials of SMTP service as before. And fill up fields as below.

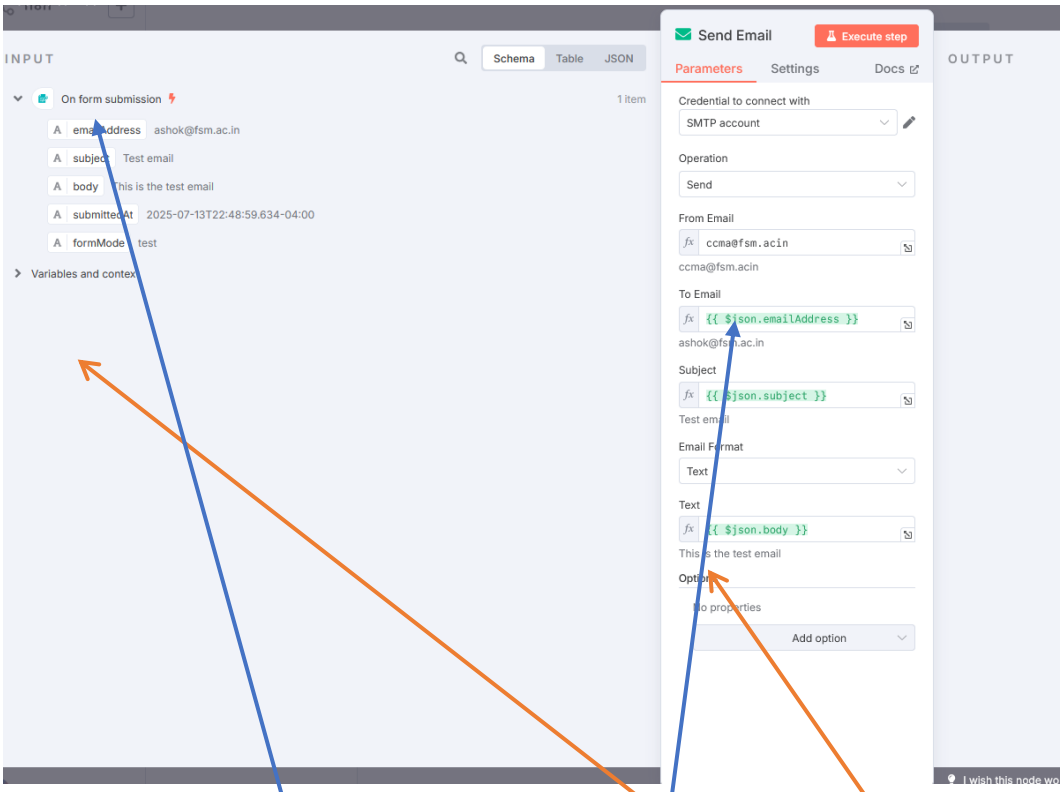


*Figure 17: Left panel contains data submitted from **form**. Drag fields from left panel to the Central panel instead of writing json code. For example, drag **emailAddress** from left panel and drop it in **To:Email** field; similarly for **subject** and other fields.*

On Execute workflow, an email goes to ashok@fsm.ac.in.

F. N8n memory error

n8n keeps data in memory while the workflows are running. Creating sub-workflows is a good idea as after a sub-workflow is executed, its memory is released. At times n8n breaks and gives memory error. Error message is about JavaScript heap memory being exceeded. Memory needs to be increased. See this link and this link.

One can assign more memory by changing the environment variable `--max-old-space-size`. This can be done while starting docker, as:

```
docker run -d --name n8n -p 5678:5678 -e NODE_OPTIONS="--max-old-space-size=8000"
docker.n8n.io/n8nio/n8n
```

And, if n8n is directly installed, run n8n as:

```
NODE_OPTIONS="--max-old-space-size=8000" npx n8n
```

To check memory usage, issue top command. Specifically for user ashok, issue top -u ashok :

```
top - 09:33:57 up 21 min,  1 user,  load average: 0.11, 0.08, 0.02
Tasks:  52 total,   1 running,  51 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.5 us,  0.1 sy,  0.0 ni, 99.4 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1486 ashok     20   0   38.6g   6.9g  56640 S  14.0  22.9   1:46.01 node
   1605 ashok     20   0    7800   3680   3040 S   0.3   0.0   0:00.31 top
      1 root      20   0  167152  11356   7836 S   0.0   0.0   0:00.81 systemd
      2 root      20   0    3060   1760   1760 S   0.0   0.0   0:00.00 init-systemd(Ub
      6 root      20   0    3076   1820   1760 S   0.0   0.0   0:00.00 init
     61 root      19  -1   47804  14080  13280 S   0.0   0.0   0:00.11 systemd-journal
     90 root      20   0   23028   5888   4608 S   0.0   0.0   0:00.10 systemd-udevd
    127 systemd+  20   0   26200  14240   9120 S   0.0   0.0   0:00.05 systemd-resolve
    128 systemd+  20   0   89364   7040   6240 S   0.0   0.0   0:00.05 systemd-timesyn
    209 root      20   0    4308   2560   2400 S   0.0   0.0   0:00.00 cron
    211 message+  20   0    8588   4000   3680 S   0.0   0.0   0:00.13 dbus-daemon
    223 root      20   0   30088  18400   9920 S   0.0   0.1   0:00.08 networkd-dispat
    226 syslog    20   0  222404   4800   4160 S   0.0   0.0   0:00.02 rsyslogd
    232 root      20   0   15336   6880   6240 S   0.0   0.0   0:00.08 systemd-logind
    253 root      20   0    3240   2080   2080 S   0.0   0.0   0:00.00 agetty
    257 root      20   0    3196   1920   1920 S   0.0   0.0   0:00.00 agetty
    258 root      20   0   15436   9120   7520 S   0.0   0.0   0:00.00 sshd
    322 root      20   0  107164  20960  13120 S   0.0   0.1   0:00.07 unattended-upgr
    359 postgres  20   0  215808  29600  27040 S   0.0   0.1   0:00.05 postgres
    441 postgres  20   0  215808   7448   4800 S   0.0   0.0   0:00.00 postgres
    442 postgres  20   0  215808   8568   5920 S   0.0   0.0   0:00.04 postgres
    443 postgres  20   0  215808  11448   8800 S   0.0   0.0   0:00.01 postgres
    444 postgres  20   0  216376  10168   7200 S   0.0   0.0   0:00.00 postgres
```

Figure 18: `top` command output. RES shows total memory usage by user 'ashok' and the process is `node` (ie nodejs).

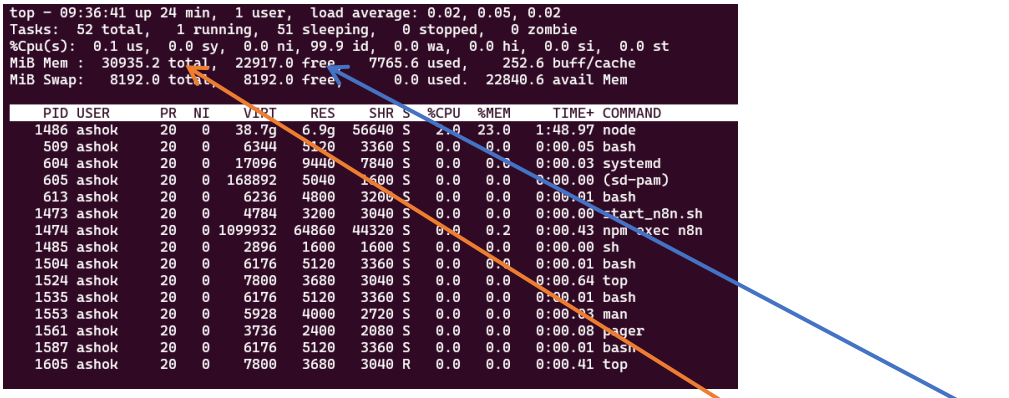


Figure 19: Output of command: `top -u ashok`. You can also see `total memory` as also `free memory`

G. N8n on docker with PostgreSQL on machine

Steps:

a. Reconfigure PostgreSQL security:

- Configure */etc/postgresql/postgresql.conf*. Modify the *listen_addresses* parameter to allow connections from external interfaces. To allow connections from all interfaces, set it to '*'.

  listen_addresses = '*'

- Configure /etc/postgresql/pg_hba.conf. For broader access (e.g., from a specific IP range), you can use a CIDR notation. In google you can raise a query: '*how to write 172.30.109.200 with netmask of 255.255.240.0 in cidr notation*'. Or see this link for calculation:

  host all all 172.30.96.0/20 md5

- Restart postgresql

  sudo systemctl restart postgresql

- Create a user, say kumar, and assign him necessary provileges:

  ```
  sudo useradd -m kumar
  ```

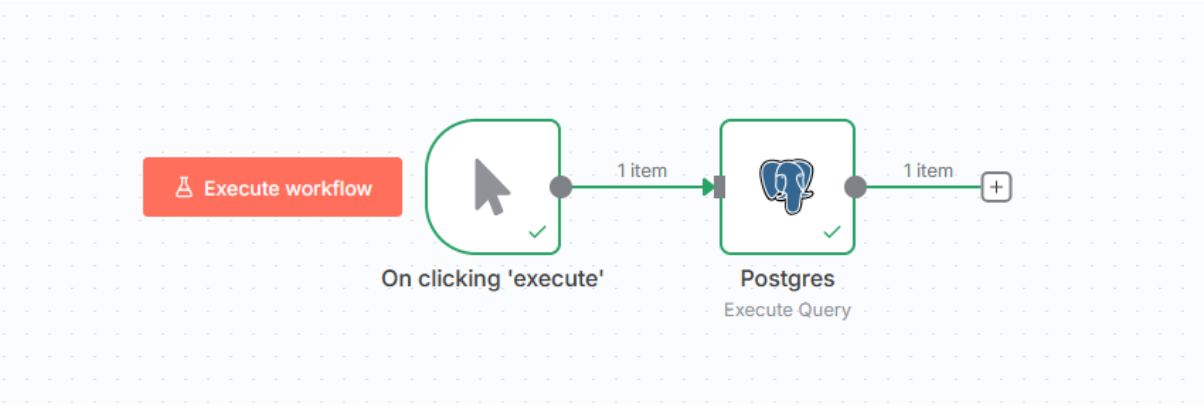
```
sudo passwd kumar
sudo -u postgres psql -c 'create database kumar;'
sudo -u postgres psql -c 'create user kumar;'
sudo -u postgres psql -c 'grant all privileges on database kumar
to kumar;' -d kumar
sudo -u postgres psql -c "alter user kumar with encrypted
password 'kumar';"
sudo -u postgres psql -c " GRANT ALL ON SCHEMA public TO kumar;"
-d kumar
sudo -u postgres psql -c " CREATE EXTENSION vector;" -d kumar

# Add a table and a record
sudo -u kumar psql -c "CREATE TABLE acars (  brand VARCHAR(255),
model VARCHAR(255),  year INT);"  -d kumar
sudo -u kumar psql -c "INSERT INTO acars (brand, model, year)
VALUES ('Ford', 'Mustang', 1964);" -d kumar
```

- OR as: Create a table in database *kumar*, as:

```
$sudo su kumar
$psql kumar
kumar=> \c kumar
You are now connected to database "kumar" as user "kumar".
kumar=> CREATE TABLE acars (  brand VARCHAR(255), model VARCHAR(255),year INT);
kumar=> INSERT INTO acars (brand, model, year) VALUES ('Ford','Mustang', 1964);
kumar=> select * from acars ;
```

b. In n8n create the following very simple network:



Create PostgreSQL credentials, as:

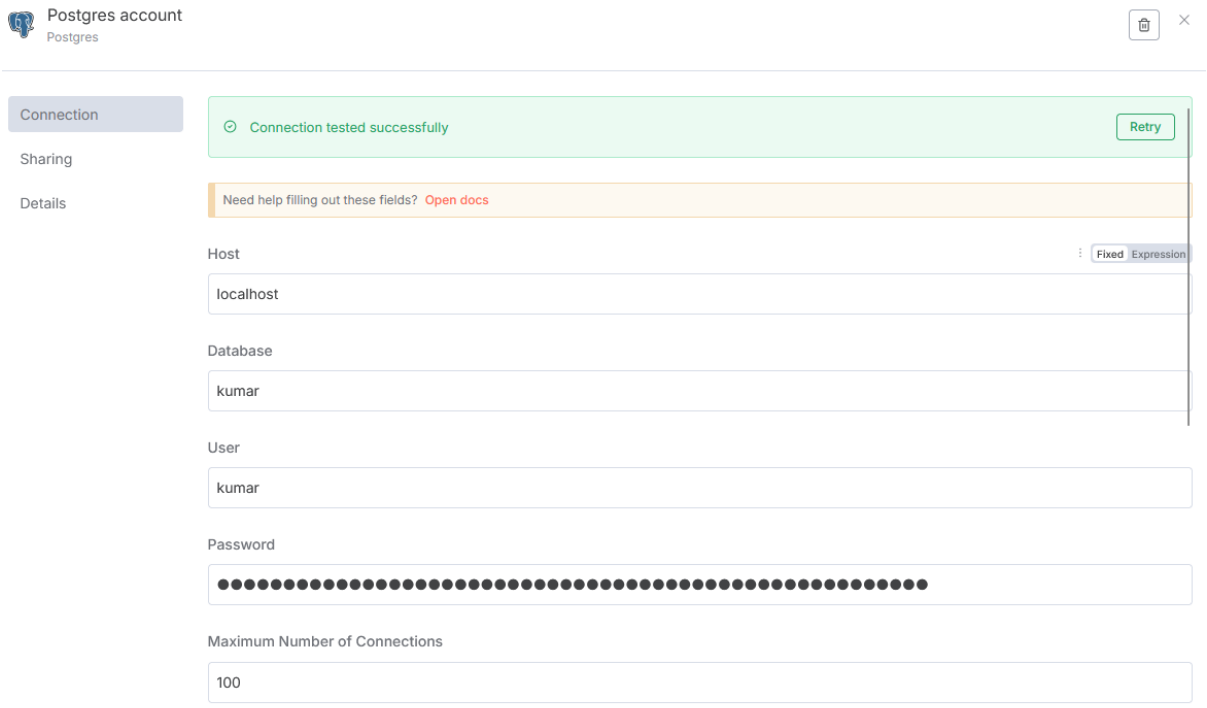


Figure 20: User: kumar; password: kumar; database: kumar

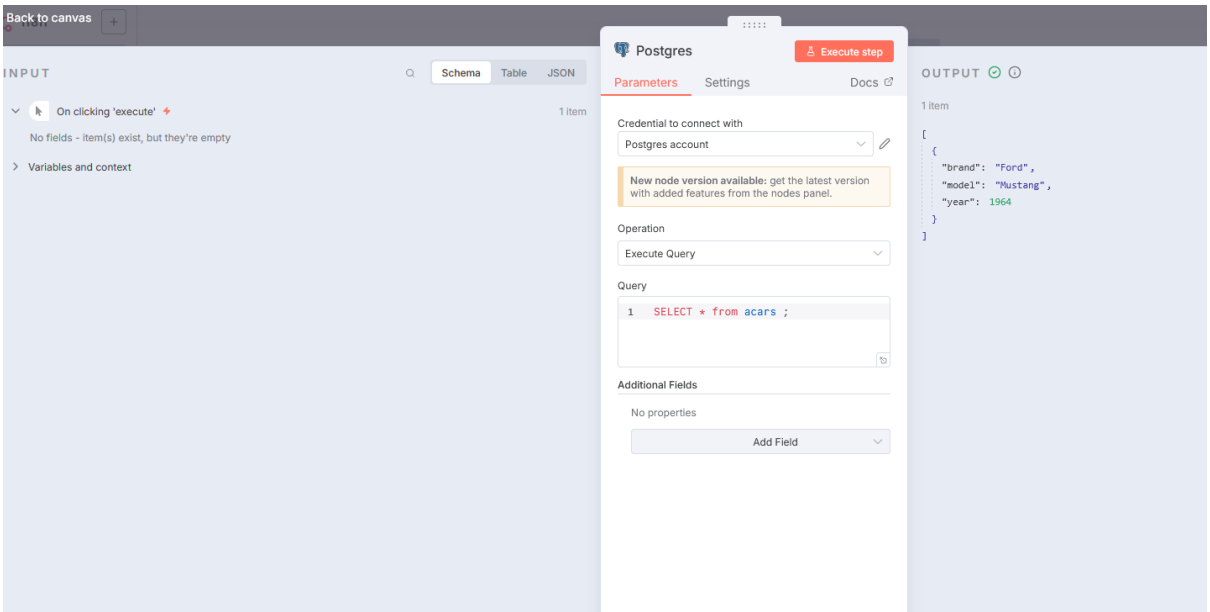


*Figure 21: Query: select * from acars; The result is on the right.*

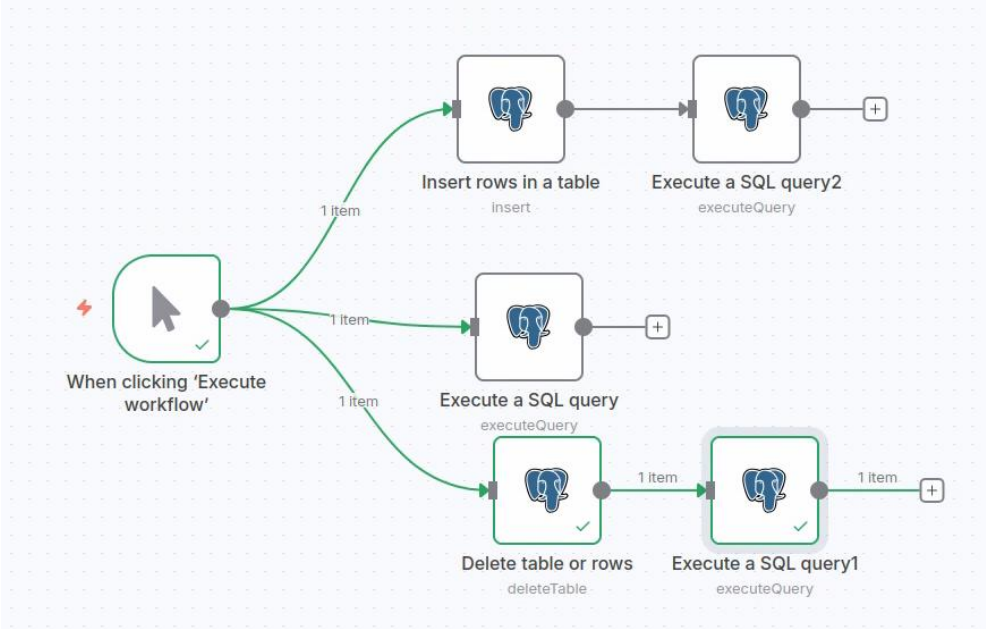


Figure 22; Workflow using different postgres nodes

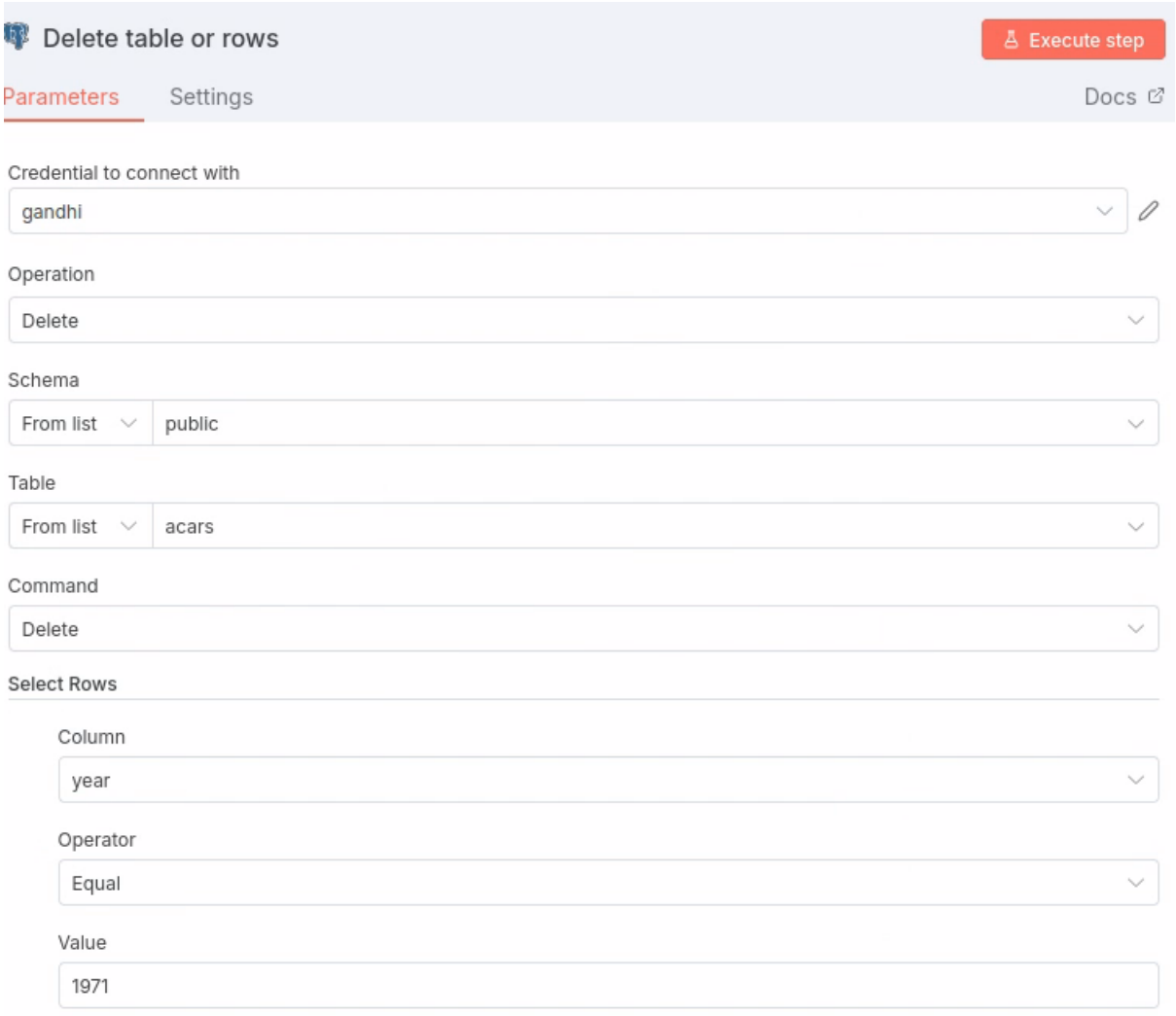
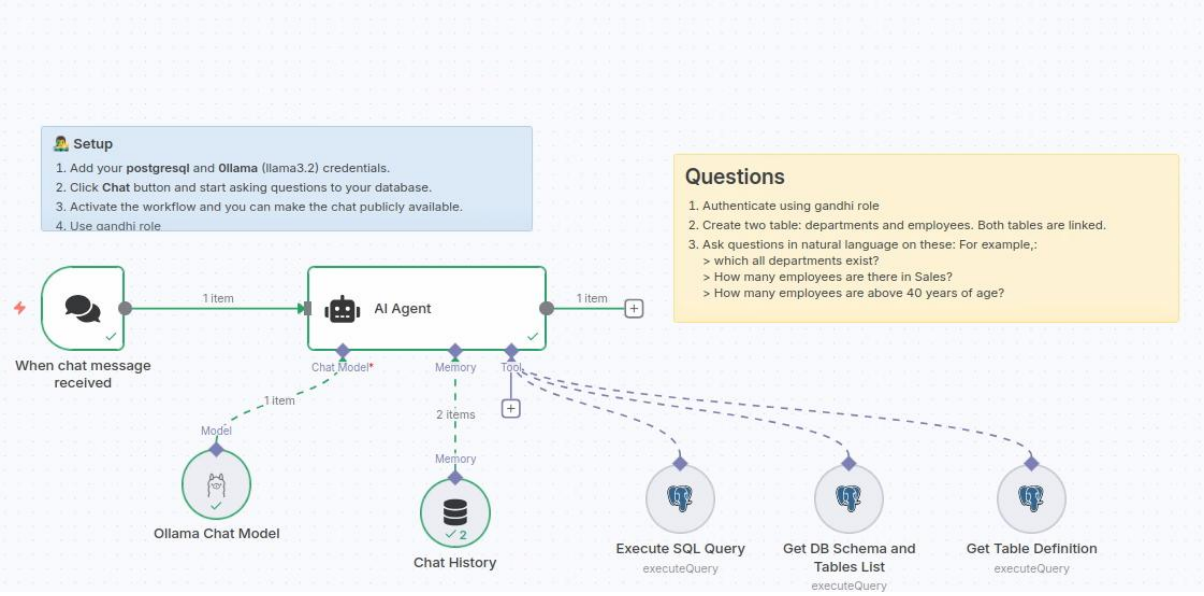


Figure 23: Delete table rows settings

H. AI agent and postgres

Postgres with ai agent

https://n8n.io/workflows/1954-ai-agent-chat/



System prompt of AI agent:

> You are DB assistant. You need to run queries in DB aligned with user requests.
>
> Run custom SQL query to aggregate data and response to user. Make sure every table has schema prefix to it in sql query which you can get from `Get DB Schema and Tables List` tool.
>
> Fetch all data to analyse it for response if needed.
>
> ## Tools
>
> - Execute SQL query- Executes any sql query generated by AI
>
> - Get DB Schema and Tables List - Lists all the tables in database with its schema name
>
> - Get Table Definition - Gets the table definition from db using table name and schema name"

I.

J.

K. Data transformation

L. Sub-workflows in n8n

M. Using webhook in n8n

N. Slack

How to get slack api or token from slack. Here are the steps:

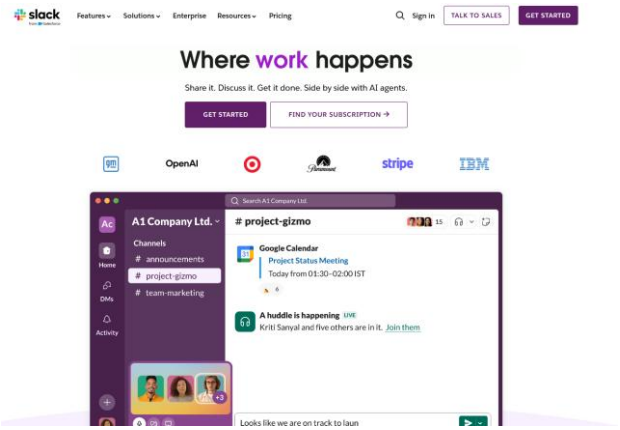


Figure 24: Sign in using your email.

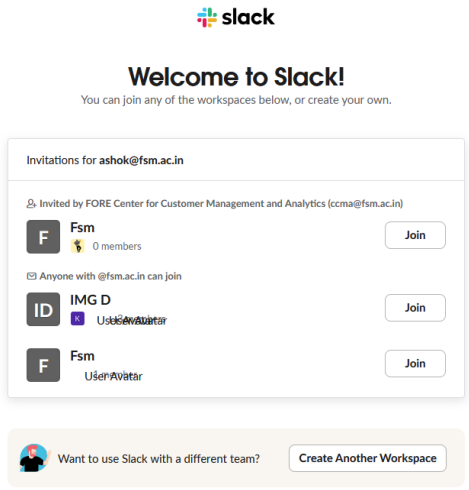


Figure 25: Create a new Workspace of your own where your team will work

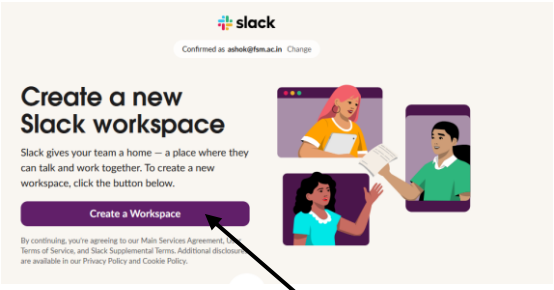


Figure 26: Create a Workspace

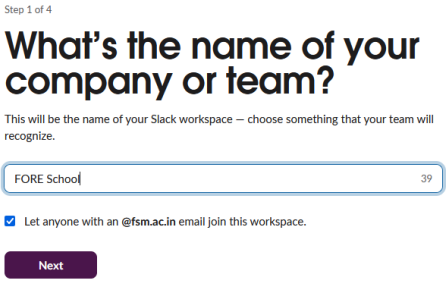


Figure 27: Fill in some details about your workspace. Name your team and click Next

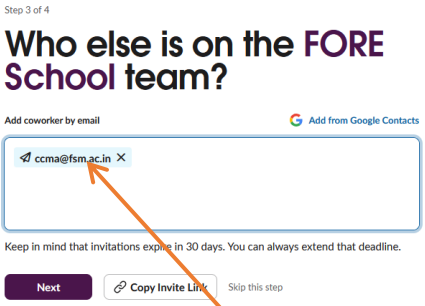


Figure 28: Add team members

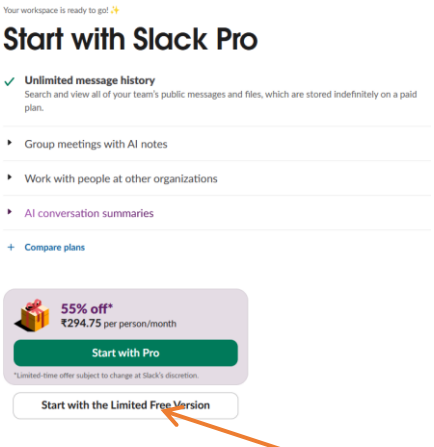


Figure 29: No. Start with a limited free version

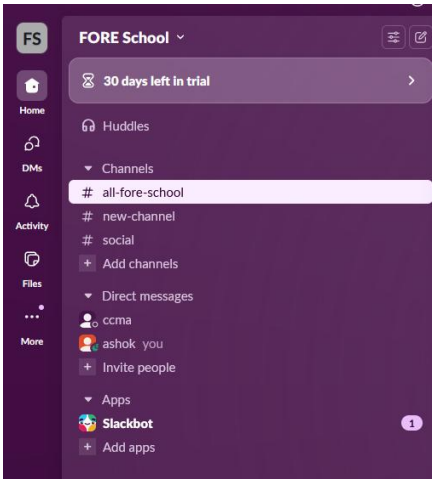


Figure 30: Three channels, one app

What are Slack channels:

A Slack channel is a dedicated space within a Slack workspace where teams can collaborate, communicate, and share information related to a specific topic, project, or team. Channels can be public, allowing anyone in the workspace to join, or private, requiring explicit invitation. Key features of channels are:

Collaboration: Channels facilitate communication, file sharing, and even video/audio calls.

Searchability: Messages and files within a channel are searchable, making it easy to find past conversations and information.

Customization: Channels can be created for various purposes, such as project updates, team discussions, or specific topics.

A company might have a public channel for general announcements (#general), a private channel for sensitive HR matters (#hr-private), and a project-specific channel for a new product launch (#product-launch).

What is an app

Apps connect other software that you use (such as Google Calendar, OneDrive or one of your company's internal tools) to Slack. With all your tools in one place, you can streamline work and help people in your workspace collaborate more effectively.

What you need to know

- There are a few different types of apps that you may see in Slack – built by Slack, third parties or your own team. How an app was built determines how it can be installed and managed in a workspace, as well as where and how you'll be able to interact with it.
- By default, any workspace member can install apps, but owners and admins can choose to restrict this permission. Once an app is installed to a workspace, any member can connect their account to use it.
- Before installing an app from the Slack Marketplace, you can review its privacy policy and security and compliance information (if submitted by the app's developer) from the app page. We recommend only choosing services that you trust when installing apps to Slack.

O. Pinecone vector store:

1. Create a free account (say, using google)
2. Create an API key
3. Create a (vector) index

Understanding Pinecone indexes:

In Pinecone, an index is the primary organizational unit for storing and querying vector data. Think of it as a table in a database, but specifically designed for efficient similarity searches on vectors. Indexes can accept, store, and serve queries on vectors, as well as perform other vector operations. Here's a more detailed breakdown:

Storage and Organization:

Indexes hold the vector embeddings of your data, allowing you to store and manage them in a structured way.

Key Parameters dimension etc:

When creating an index, you'll need to define its name, the dimensionality of the vectors it will store, and the similarity metric (e.g., cosine, Euclidean).

Types of Indexes for scaling out:

Pinecone offers serverless and pod-based indexes, with different options for scaling and performance.

Namespaces:

Within an index, you can further organize data using namespaces, allowing you to isolate queries to specific subsets of your data.

Similarity Search:

Pinecone indexes are built for efficient similarity searches, meaning you can quickly find vectors that are similar to a given query vector.

Highest Level:

It's the top-level container for your vector data within Pinecone, similar to a table in a relational database.

Creation and Management:

Indexes can be created via the Pinecone UI or programmatically using their API.

Metadata:

Pinecone indexes can store associated metadata with each vector, enabling filtering and more complex search conditions.

Metadata

Every record in an index must contain an ID and a vector. In addition, you can include metadata key-value pairs to store additional information or context. When you query the index, you can then include a metadata filter to limit the search to records matching a filter expression. Searches without metadata filters do not consider metadata and search the entire namespace.

See LlamaIndex example for Chromadb as to how metadata filter works.

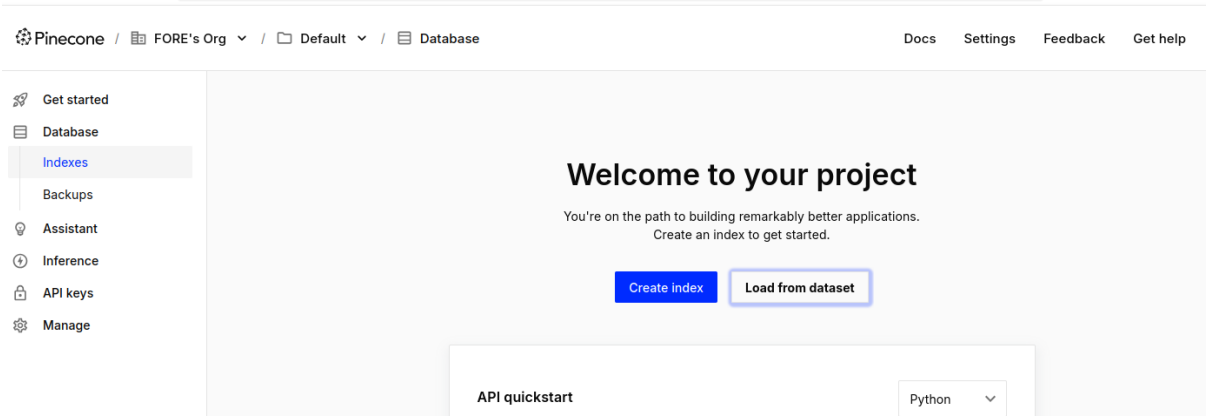


Figure 31: Click Create Index to begin creating an index

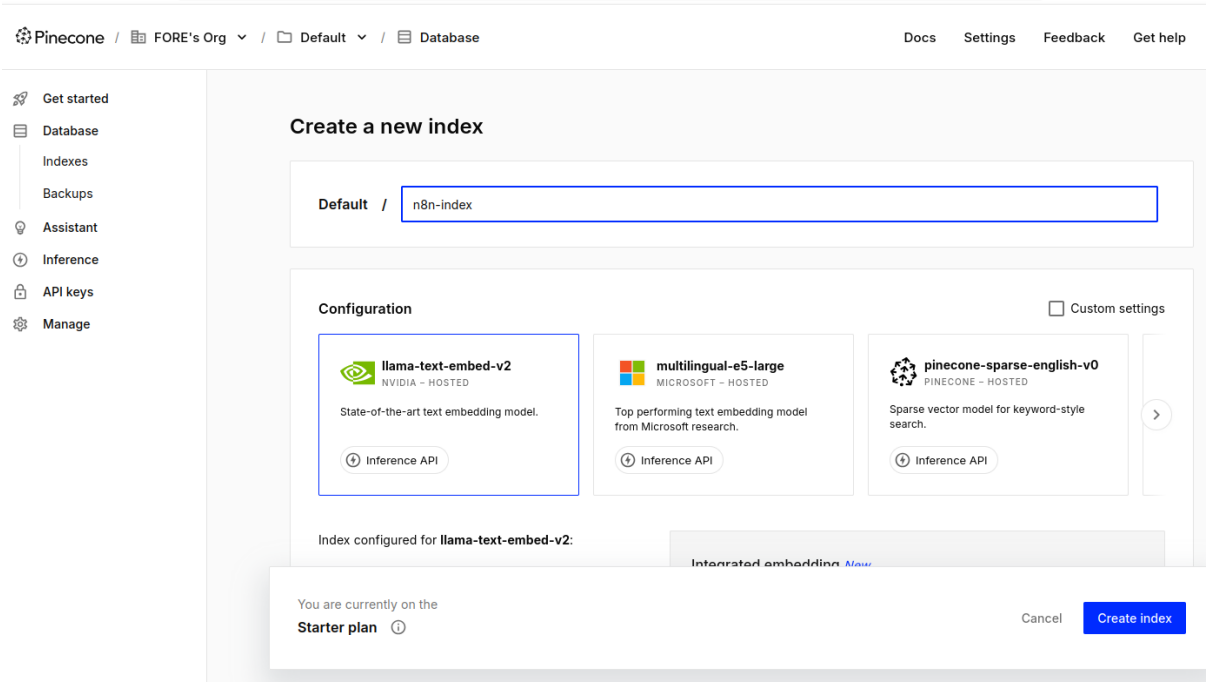


Figure 32: Select an index and select a Configuration for index (such as Vector Dimension: 768, 1024 etc). See figure below

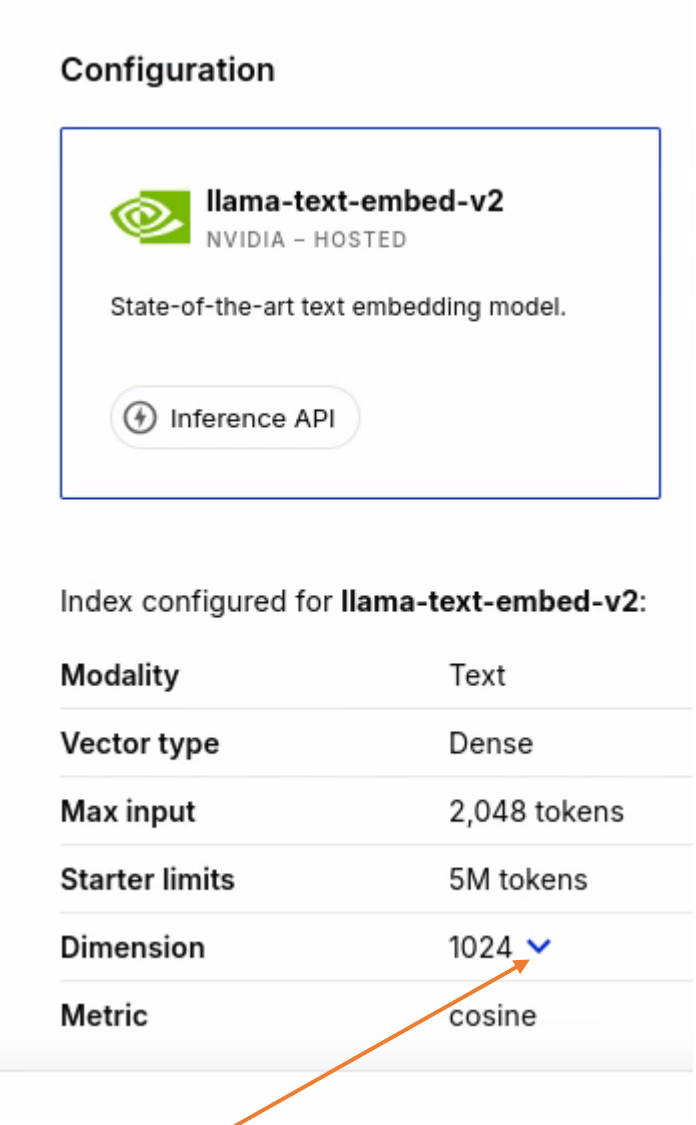


Figure 33: Set vector Dimension as per your embedder (click Down-arrow). For example, nomic-embed-text has a vector dimension of 768 AND NOT of 1024. We have used 768

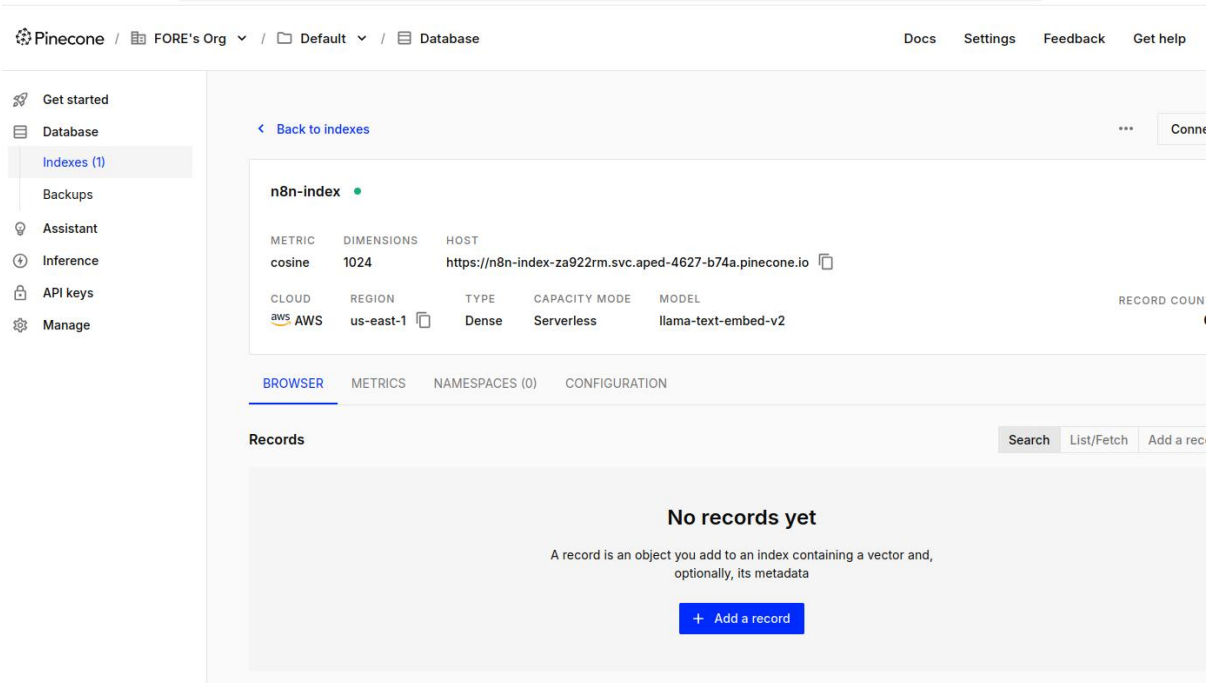


Figure 34: Create index. Records will come from n8n workflow

Note that selected embedder will also decide max input tokens. The character splitter chunk size will affect the amount of **input tokens** you're trying to get. For the embedding model of. *mxbai-embed-large,* max input token limit is 512.

Our dataset is *students.json* on GitHub, at site: LLMs/install_ai_tools/mongodb//datasets/


```
{"_id":0,"name":"aimee Zank","scores":[{"score":1.463179736705023,"type":"exam"},{"score":11.78273309957772,"type":"quiz"},{"score":35.8740349954354,"type":"homework"}]}
{"_id":1,"name":"Aurelia Menendez","scores":[{"score":60.06045071030959,"type":"exam"},{"score":52.79790691903873,"type":"quiz"},{"score":71.76133439165544,"type":"homework"}]}
{"_id":2,"name":"Corliss Zuk","scores":[{"score":67.03077096065002,"type":"exam"},{"score":6.301851677835235,"type":"quiz"},{"score":66.28344683278382,"type":"homework"}]}
{"_id":3,"name":"Bao Ziglar","scores":[{"score":71.64343899778332,"type":"exam"},{"score":24.80221293650313,"type":"quiz"},{"score":42.26147058804812,"type":"homework"}]}
{"_id":4,"name":"Zachary Langlais","scores":[{"score":78.68385091304332,"type":"exam"},{"score":90.2963101368042,"type":"quiz"},{"score":34.41620148042529,"type":"homework"}]}
{"_id":5,"name":"Wilburn Spiess","scores":[{"score":44.87186330181261,"type":"exam"},{"score":25.72395114668016,"type":"quiz"},{"score":63.42288310628662,"type":"homework"}]}
{"_id":6,"name":"Jenette Flanders","scores":[{"score":37.32285459166097,"type":"exam"},{"score":28.32634976913737,"type":"quiz"},{"score":81.57115318686338,"type":"homework"}]}
{"_id":7,"name":"Salena Olmos","scores":[{"score":90.37826509157176,"type":"exam"},{"score":42.48780666956811,"type":"quiz"},{"score":96.52986171633331,"type":"homework"}]}
{"_id":8,"name":"Daphne Zheng","scores":[{"score":22.13583712862635,"type":"exam"},{"score":14.63969941335069,"type":"quiz"},{"score":75.94123677556644,"type":"homework"}]}
```


Figure 35: Extract from file students.json

Here are two rows from the data:


```
{"_id":0,"name":"aimee
Zank","scores":[{"score":1.463179736705023,"type":"exam"},{"score":11.78273309957772,"type":"q
uiz"},{"score":35.8740349954354,"type":"homework"}]}

{"_id":1,"name":"Aurelia
Menendez","scores":[{"score":60.06045071030959,"type":"exam"},{"score":52.79790691903873,"type
":"quiz"},{"score":71.76133439165544,"type":"homework"}]}
```


P. Simple RAG with n8n

File: 'Simple RAG flow.json'

We use '*n8n Form trigger*' node as file uploader. The vector store is in-memory vector store: Simple Vector Store.

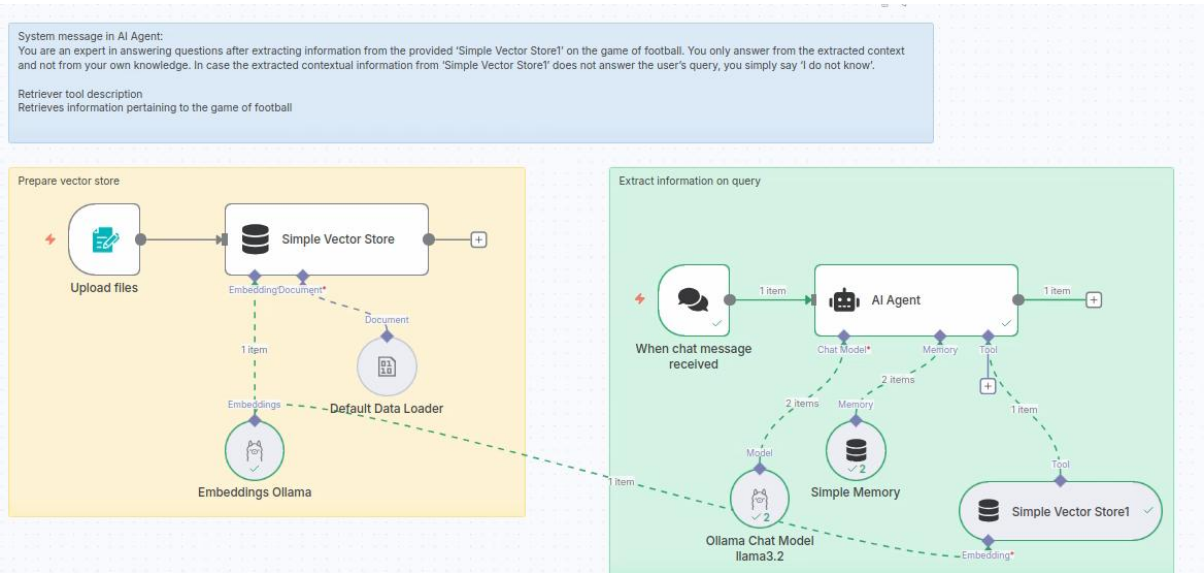


Figure 36: Full workflow

Its two sub-workflows are as follows:

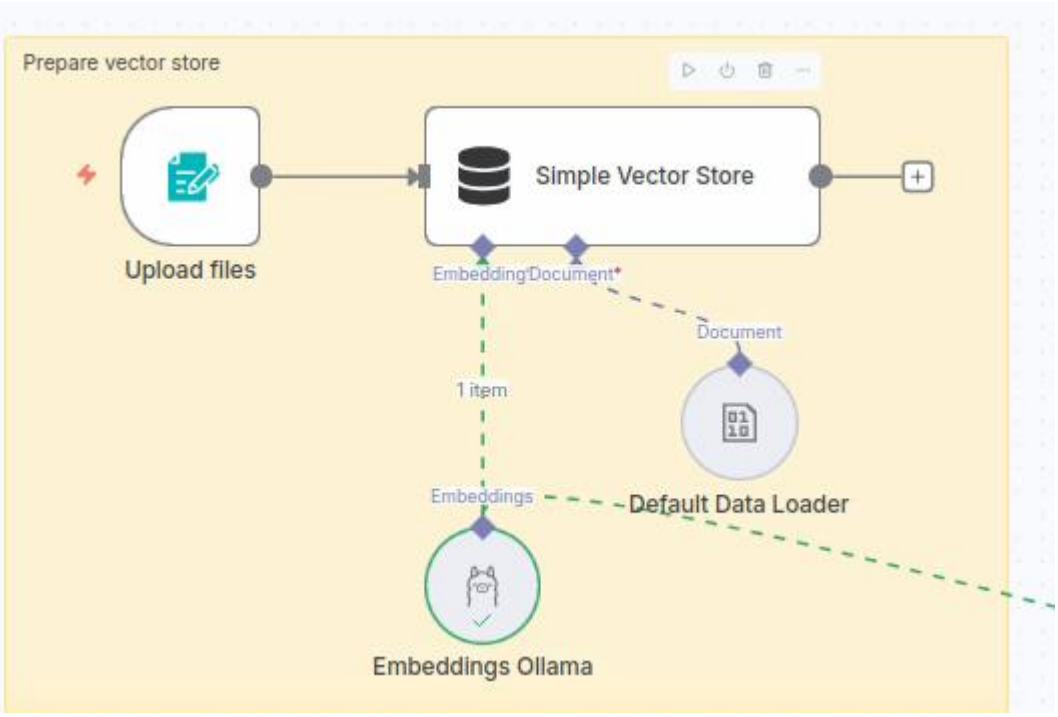


Figure 37: Creating a vector store

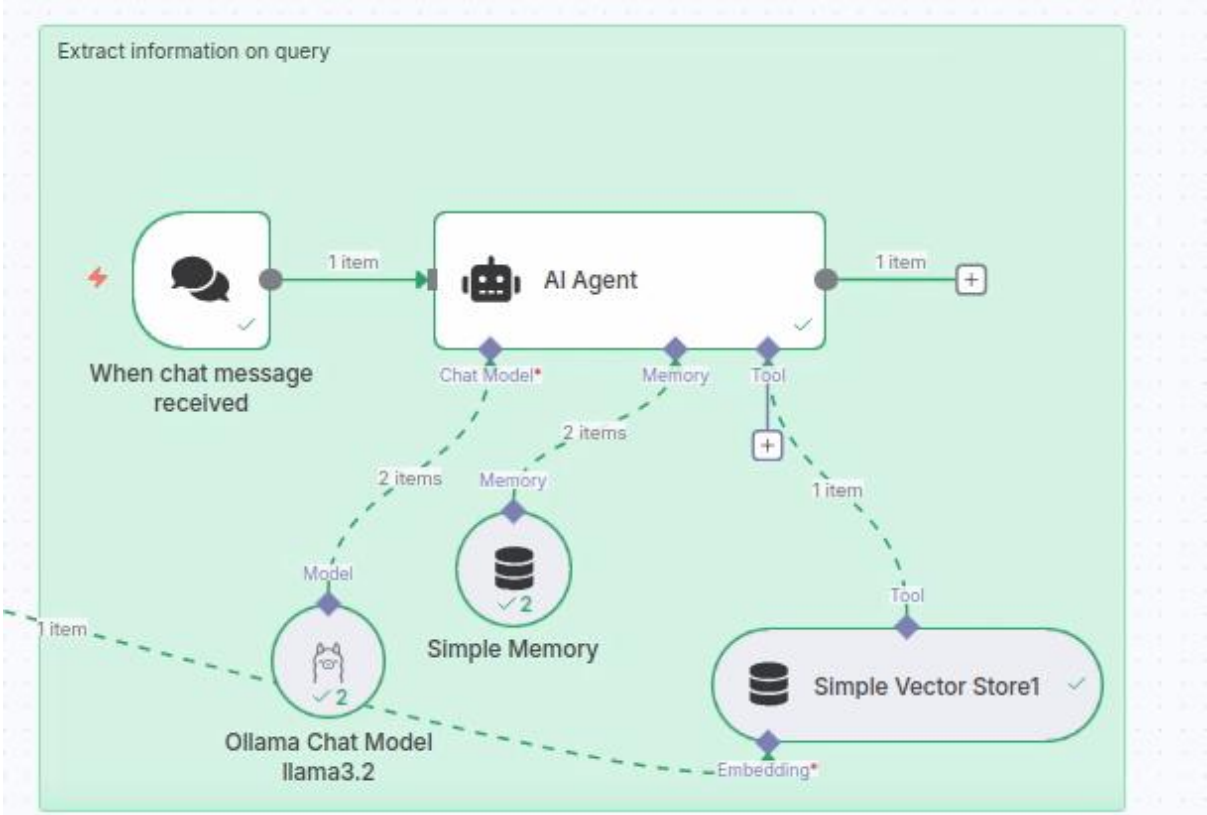


Figure 38: Querying a vector store

System message of AI Agent and vector tool description are as follows:

```
System message in AI Agent:
You are an expert in answering questions after extracting information from the provided 'Simple Vector Store1' on the game of football. You only answer from the extracted context and not from your own knowledge. In case the extracted contextual information from 'Simple Vector Store1' does not answer the user's query, you simply say 'I do not know'.

Retriever tool description
Retrieves information pertaining to the game of football
```

Q. RAG with n8n

Files: n8n/2025/Rag Flow-I.json and Rag Flow-II.json

See [this link] for the detailed blog. Vectorization may take a very long time depending upon the input size.

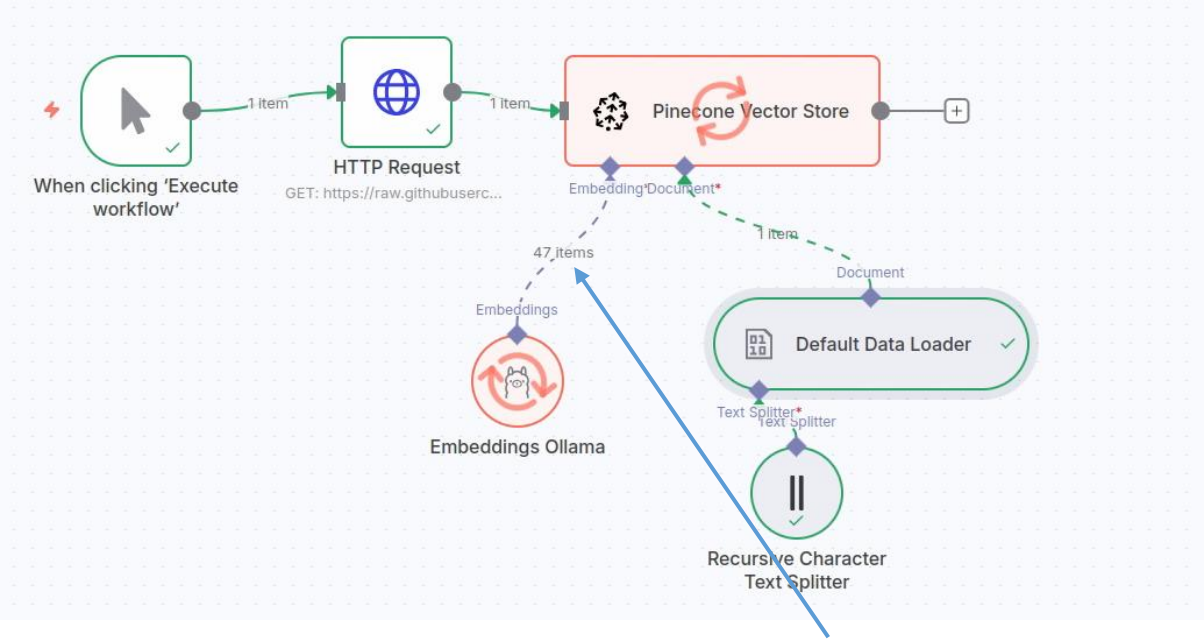


Figure 39: Our workflow to save data to vector store. 47 items possibly means that 47 chunks were processed till now. Chunk size for Recursive text splitter is 512 as selected embedder is:

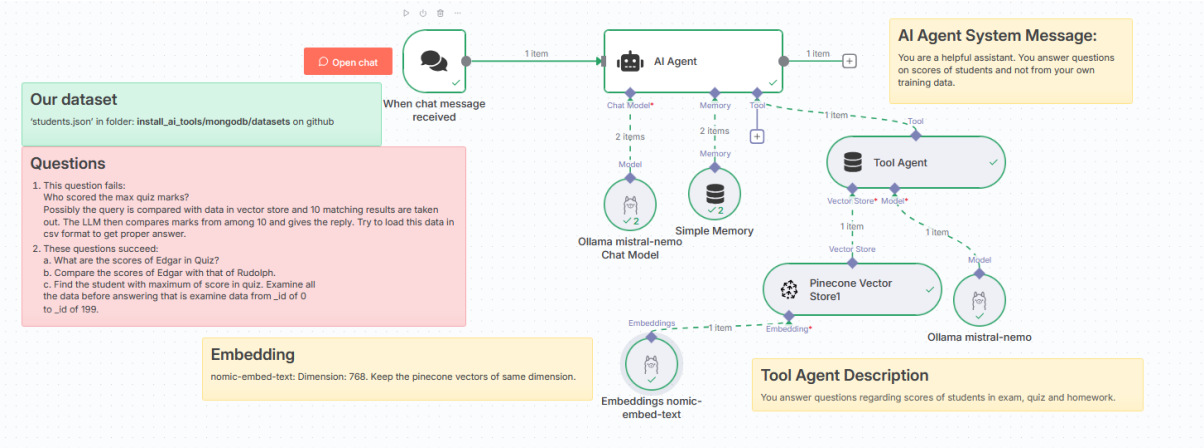


Figure 40: Reading vector data from Pinecone vector store

a. Here are the models used:

Embedding:	nomic-embed-text (ollama)
Chat ollama:	mistral-nemo (ollama)

b. Questions asked:

Questions

1. This question fails:
   Who scored the max quiz marks?
   Possibly the query is compared with data in vector store and 10 matching results are taken out. The LLM then compares marks from among 10 and gives the reply. Try to load this data in csv format to get proper answer.
2. These questions succeed:
   a. What are the scores of Edgar in Quiz?
   b. Compare the scores of Edgar with that of Rudolph.
   c. Find the student with maximum of score in quiz. Examine all the data before answering that is examine data from _id of 0 to _id of 199.

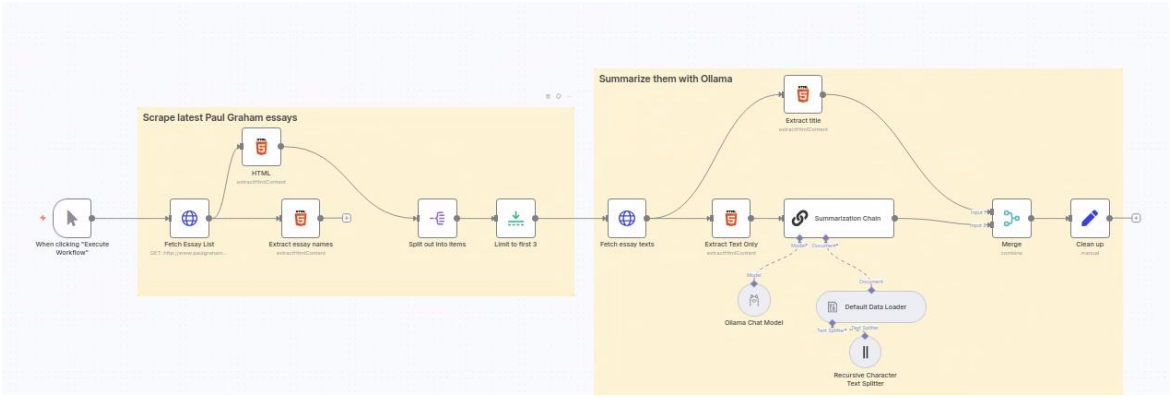
c. AI Agent System message:

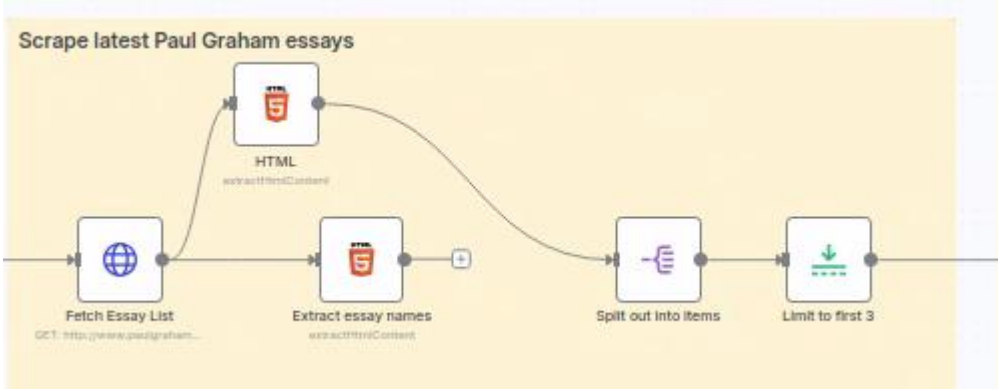
You are a helpful assistant. You answer questions on scores of students and not from your own training data.

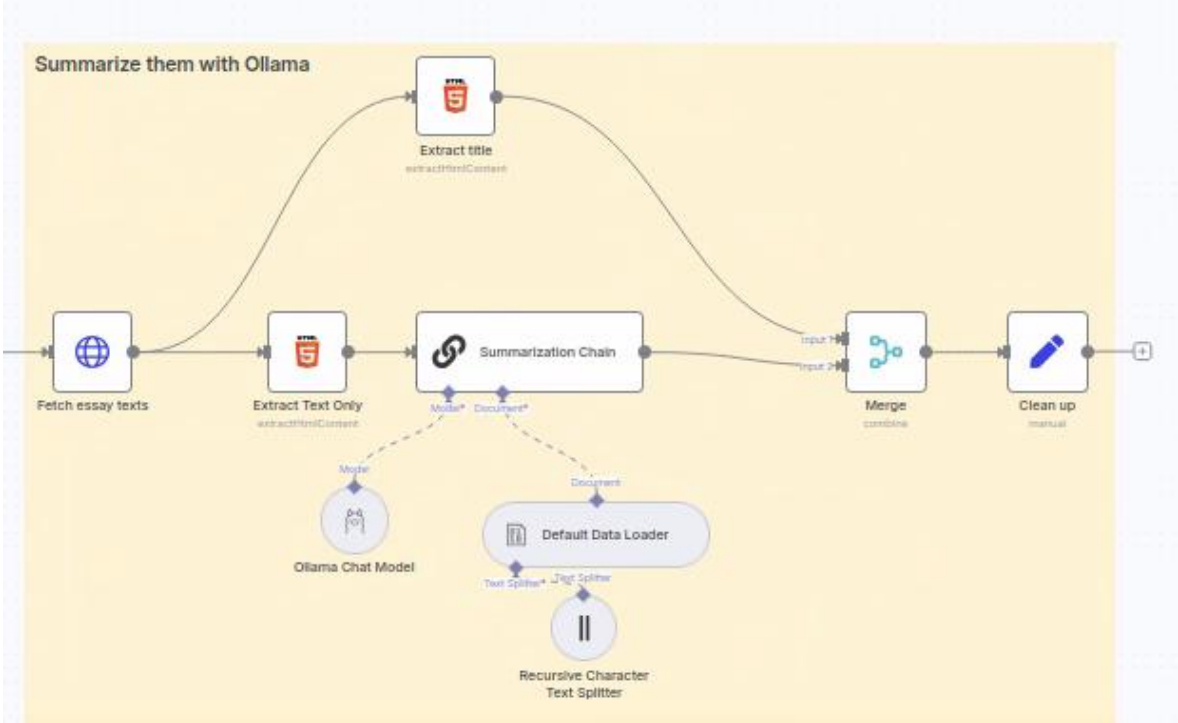
d. Tool Agent Description

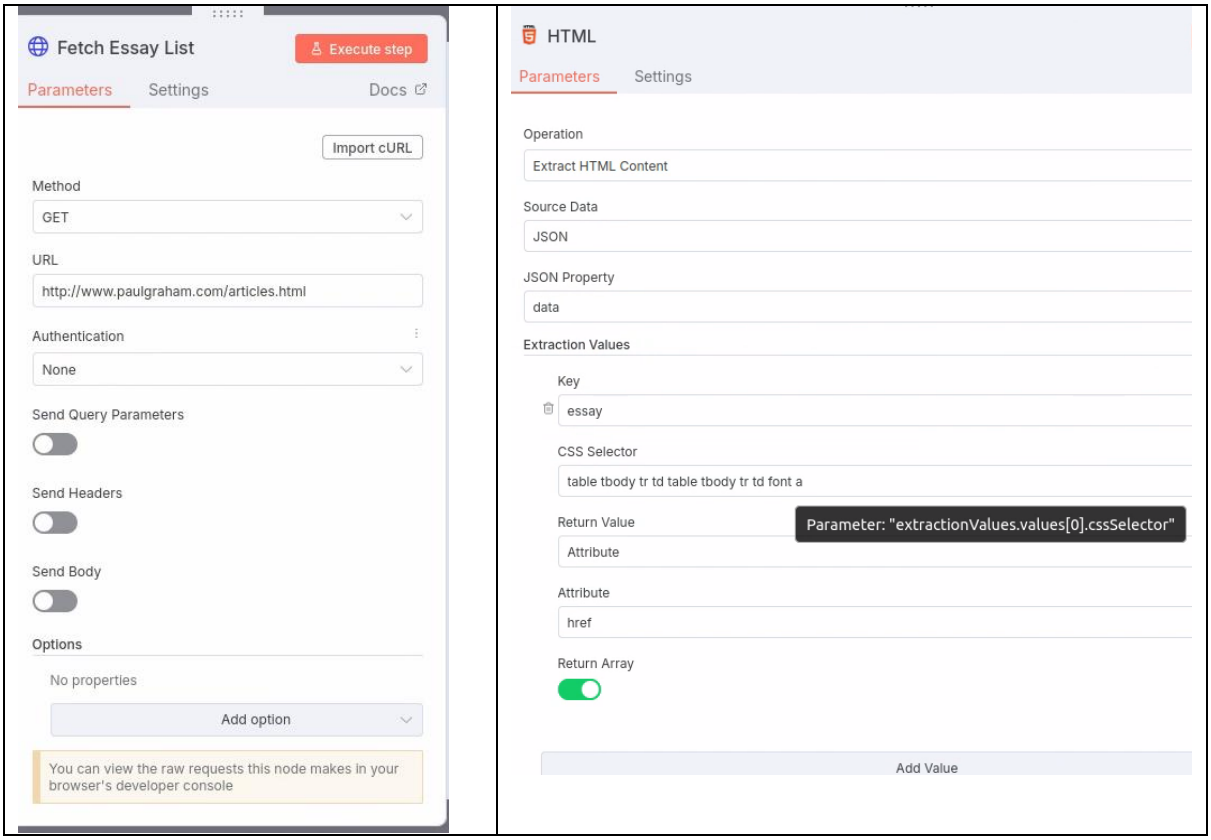
You answer questions regarding scores of students in exam, quiz and homework.

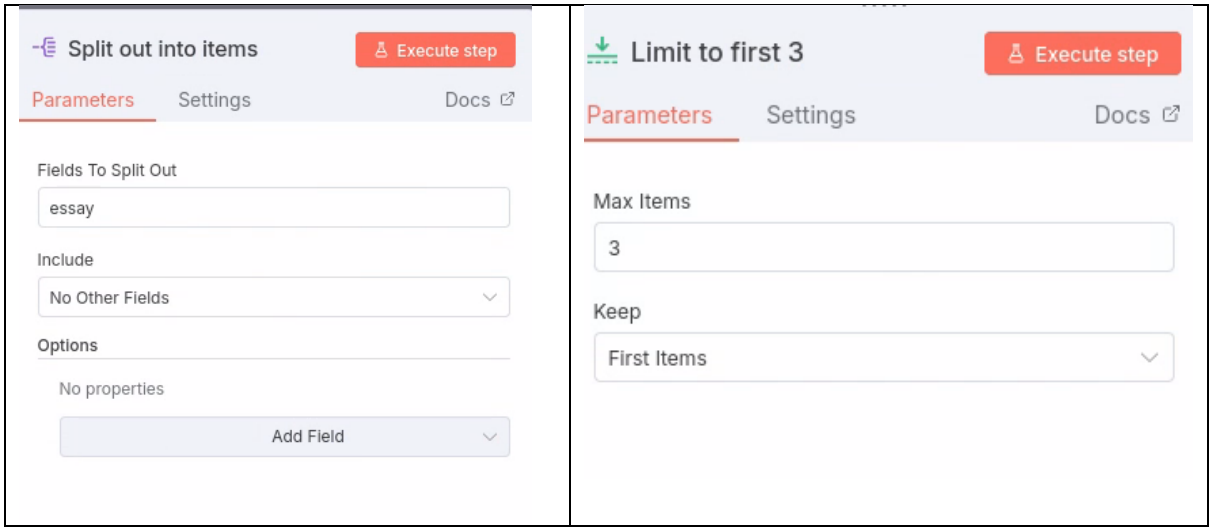
Scrapping web-pages and summarization






Summarize them with Ollama
Extract title
Fetch essay texts
Extract Text Only
Summarization Chain
Merge
Clean up
Ollama Chat Model
Default Data Loader
Recursive Character
Text Splitter


Fetch Essay List
Execute step
Parameters
Settings
Docs
Import cURL
Method
GET
URL
http://www.paulgraham.com/articles.html
Authentication
None
Send Query Parameters
Send Headers
Send Body
Options
No properties
Add option
You can view the raw requests this node makes in your browser's developer console
HTML
Parameters
Settings
Operation
Extract HTML Content
Source Data
JSON
JSON Property
data
Extraction Values
Key
essay
CSS Selector
table tbody tr td table tbody tr td font a
Return Value
Parameter: "extractionValues.values[0].cssSelector"
Attribute
Attribute
href
Return Array
Add Value



n8n how to decide css selector in html node
Thinking
Kicking off 7 searches
Looking at 105 sites
Putting it all together
Deciding on the correct CSS selector in n8n's HTML node (formerly HTML Extract node) is crucial for accurately extracting the data you need from a web page. Here's a breakdown of how to approach this, including recommended practices and tools:
1. Using Browser Developer Tools (Inspect Element)
The easiest and most common way to identify the correct CSS selector is by using your web browser's developer tools.

- **Navigate to the web page:** Open the web page containing the data you want to scrape in your browser.
- **Inspect the element:** Right-click on the specific element (text, image, link, etc.) you want to extract and select "Inspect" or "Inspect Element". This will open the browser's developer tools and highlight the element's HTML code.
- **Copy the CSS Selector/Path:** In the developer tools (usually under the "Elements" tab), right-click on the highlighted HTML element and choose "Copy" > "Selector". This will copy the CSS selector to your clipboard

Using the CSS selector in the n8n HTML node

- **Add an HTML node to your workflow:** According to n8n Docs and, the node is called "HTML" (it replaced the older "HTML Extract" node).
- **Set Source Data:** Choose "JSON" if the HTML content is in the JSON output of a previous node (like an HTTP Request node), or "Binary" if the HTML is in a binary file.
- **Specify JSON Property (if applicable):** Enter the name of the JSON property containing the HTML.
- **Add Extraction Value:** Click "Add Value" to define what you want to extract.
- **Key:** Enter a name for the extracted value (e.g., "title", "price").
- **CSS Selector:** Paste the CSS selector you identified in the previous steps.

- **Return Value:** Choose the type of data to return: "Attribute" (e.g., "href" for links), "HTML", "Text", or "Value" (for input elements).
- **Return Array (if applicable):** If you expect multiple elements to match the selector, enable "Return Array" to get an array of values.

Example:
If you want to extract the text "Medium" from a `<span>` element within a table, the copied CSS selector might look like:

```
body > table:nth-child(2) > tbody:nth-child(1) > tr:nth-child(1) > td:nth-child(2) > span:nth-child(1)
```

Understanding CSS Selector Syntax:
Familiarize yourself with basic CSS selector syntax (e.g., `.` for classes, `#` for IDs, `>` for direct children, `nth-child()` for selecting elements based on their position) to create or modify selectors manually if needed.

Slack API

To work with Slack API, click on this link or in Google, search for Slack API. Click here OR here.

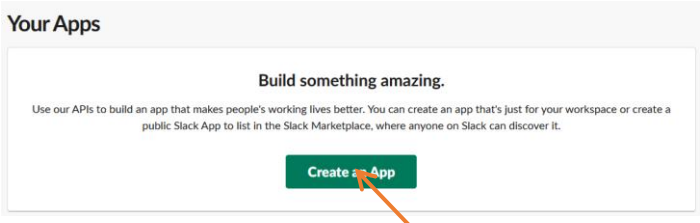


Figure 41: Click on Create an App

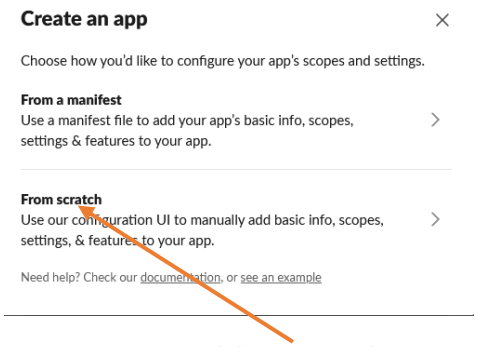


Figure 42: Click From Scratch

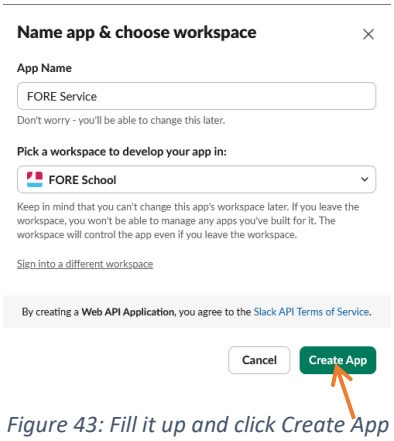


Figure 43: Fill it up and click Create App

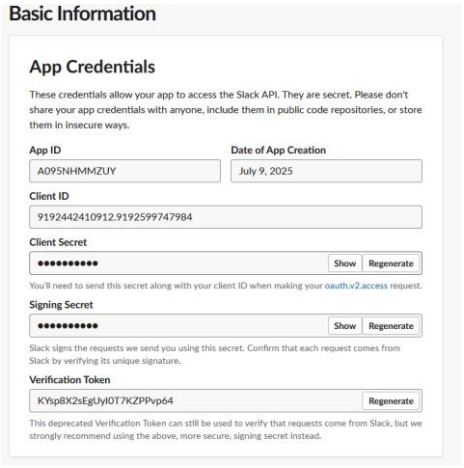


Figure 44: Not sure if to note down these or not.

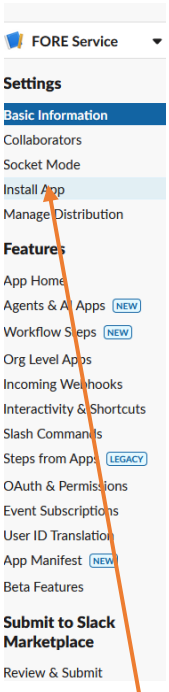


Figure 45: Click Install Apps

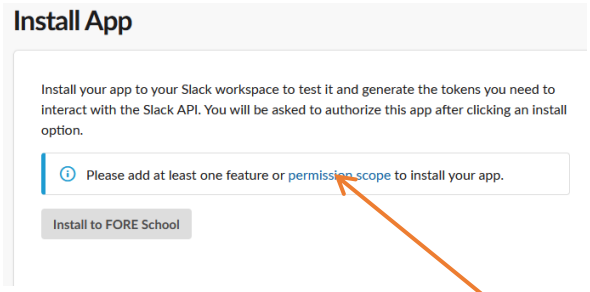


Figure 46: Click Install to FORE School. But first Give permissions

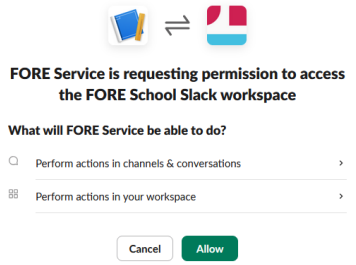


Figure 47: Allow your App permissions on your Workspace

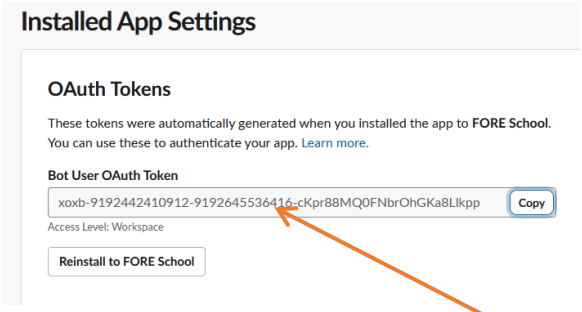


Figure 48: A token will be generated. Note this down. This token is important.

xoxb-9192442410912-9192645536416-cKpr88MQ0FNbrOhGKa8Llkpp

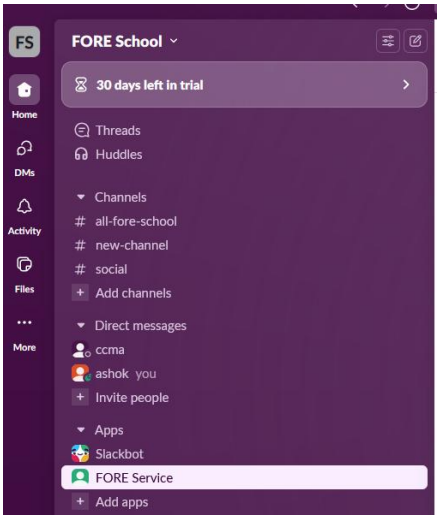
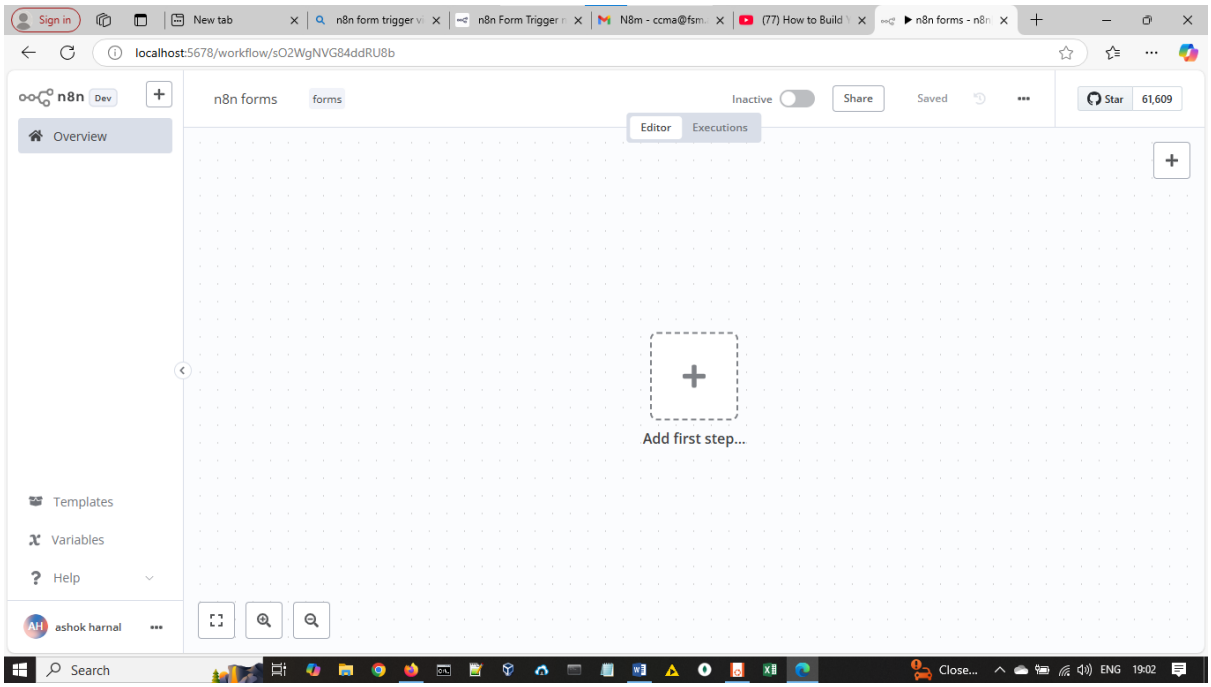
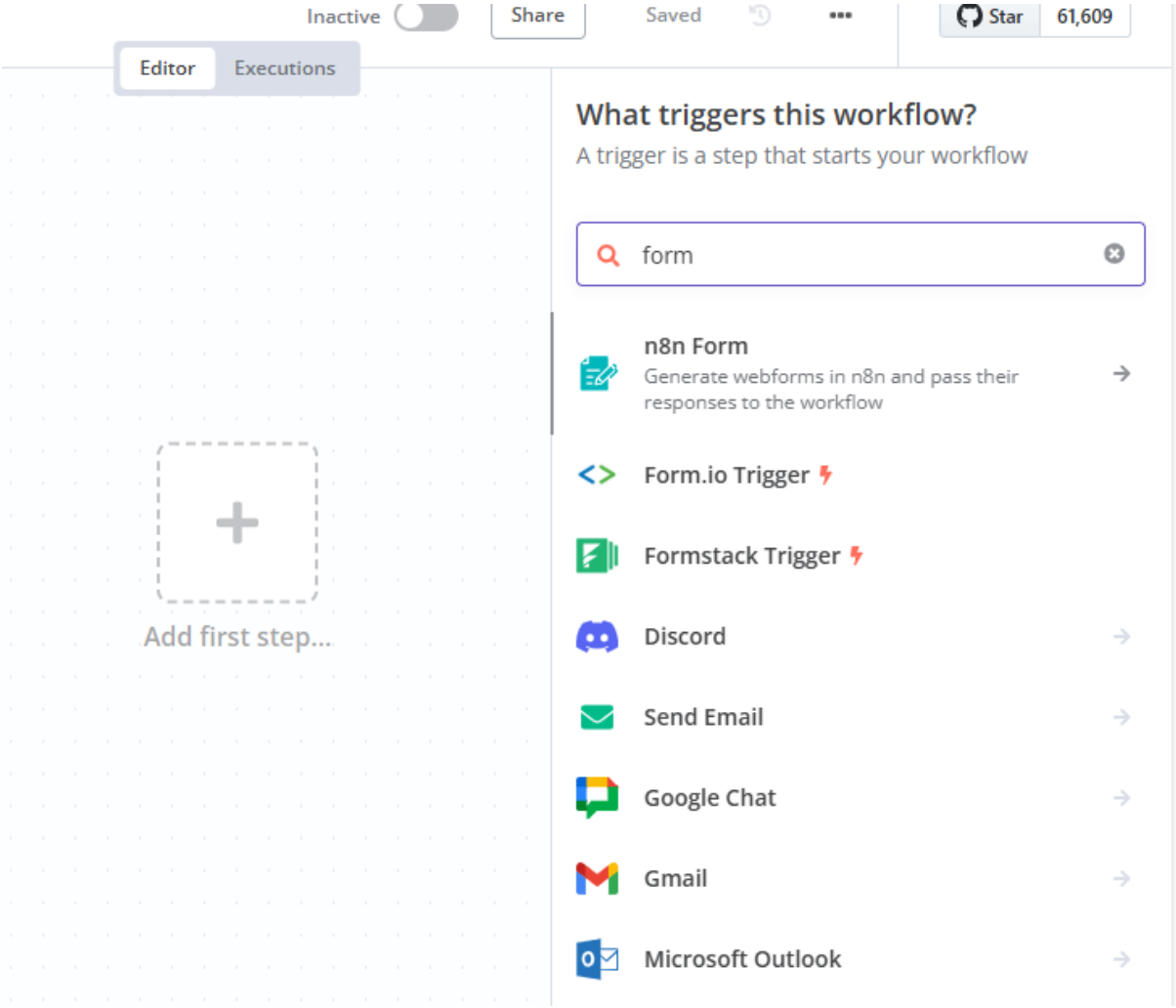
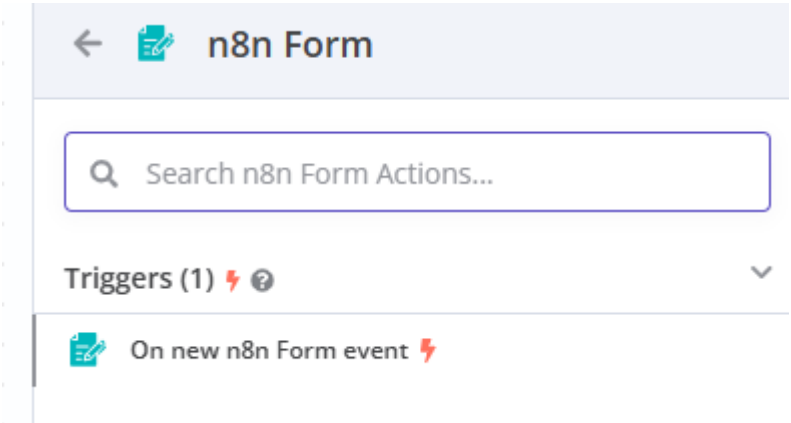


Figure 49: An App FORE Service is now visible

[YouTube video]




Inactive
Share
Saved
Star
61,609
Editor
Executions
What triggers this workflow?
A trigger is a step that starts your workflow
form
n8n Form
Generate webforms in n8n and pass their responses to the workflow
Form.io Trigger
Formstack Trigger
Discord
Send Email
Google Chat
Gmail
Microsoft Outlook
Add first step...


n8n Form
Search n8n Form Actions...
Triggers (1)
On new n8n Form event