

How to make the WeiseEule app run on Linux/Mac?

This document provides step-by-step instructions for setting up WeiseEule as a localhost application on your machine. The source code along with this guide is available in the following GitHub repository: <https://github.com/wasimaftab/WeiseEule-LocalHost>. As the app continues to evolve, the guidelines may change; therefore, users are encouraged to check the GitHub repo for up-to-date instructions. If any step is not clear or does not yield desired result, please report these in the issue section of the repository. We are committed to addressing any issues promptly.

```
.
├── grobid_client_python-master
├── index.html
├── index.mjs
├── Local_DB
├── package.json
├── Pickled_vecs
├── public
├── pycodes
├── qa_env.yml
├── server.mjs
├── vite.config.js
└── yarn.lock
```

Fig 1: App directory structure – The WeiseEule app directory contains the source code of the app and software documentation to set it up. Directories are shown in bold blue colored font.

1. Setting up JavaScript (JS) environment

Install `yarn` package manager by running the following commands

- Update package list by running the following command to ensure the most recent versions of packages and their dependencies are downloaded:

```
sudo apt update
```

- Install Node.js. If Node.js is not installed, then install it by running:

```
sudo apt install nodejs npm
```

- Install Yarn using npm (Node Package Manager) by running:

```
npm install --global yarn
```

- Verify the installation by checking the version of Yarn installed:

```
yarn --version
```

2. Setting up Python environment

- Install Conda by following instructions given in the official Anaconda documentation

<https://docs.anaconda.com/free/anaconda/install/linux/>

- Go to the app directory as given in Fig 1 and create Python environment by running `conda`

```
env create -f qa_env.yml
```

3. Get the API keys

- Obtain ENTREZ API KEY by following instructions given in the NCBI E-utilities page

<https://ncbiinsights.ncbi.nlm.nih.gov/2017/11/02/new-api-keys-for-the-e-utilities/>

- Obtain PINECONE API KEY by following the guidelines from Fig 2.

- Get an OpenAI API KEY by following these steps [https://maisieai.com/help/how-to-get-](https://maisieai.com/help/how-to-get-an-openai-api-key-for-chatgpt)

[an-openai-api-key-for-chatgpt](https://maisieai.com/help/how-to-get-an-openai-api-key-for-chatgpt)

4. Setting up API keys

Open .bashrc file located at your home directory and create environment variable to hold API keys as shown below:

```
export OPENAI_API_KEY='your-openai-key-here'

export ENTREZ_API_KEY='your-entrez-key-here'

export PINECONE_API_KEY='your-pinecone-key-here'
```

Note, keep the names of the environment variables for API keys as mentioned in this doc. If you would like to change the names, then feel free to edit the WeiseEule source code where they appear.

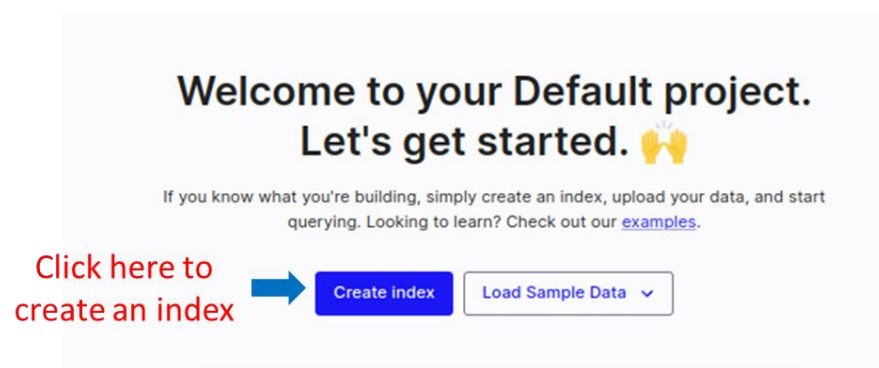
5. Setting up PINECONE environment

- Login to your account by registering at <https://pinecone.io>. See Fig 2 for more details.
- Open .bashrc file located in your home directory and set the pinecone index, region variables as shown below

```
export PINECONE_INDEX="your-pinecone-index-name"

export PINECONE_REGION="your-pinecone-region-name"
```

The index name and region are the one which you set during registration and index creation in <https://pinecone.io>



(A)

Create a new index

Default / ← Enter an index name

Configuration
The dimensions and metric depend on the model you select.

Dimensions ← Enter 1536 as dimension


Metric ← Select cosine as metric [Setup by model](#)

Capacity mode

SERVERLESS PODS STARTER


Starter
Free tier with a capacity for 100k records.

Cloud provider

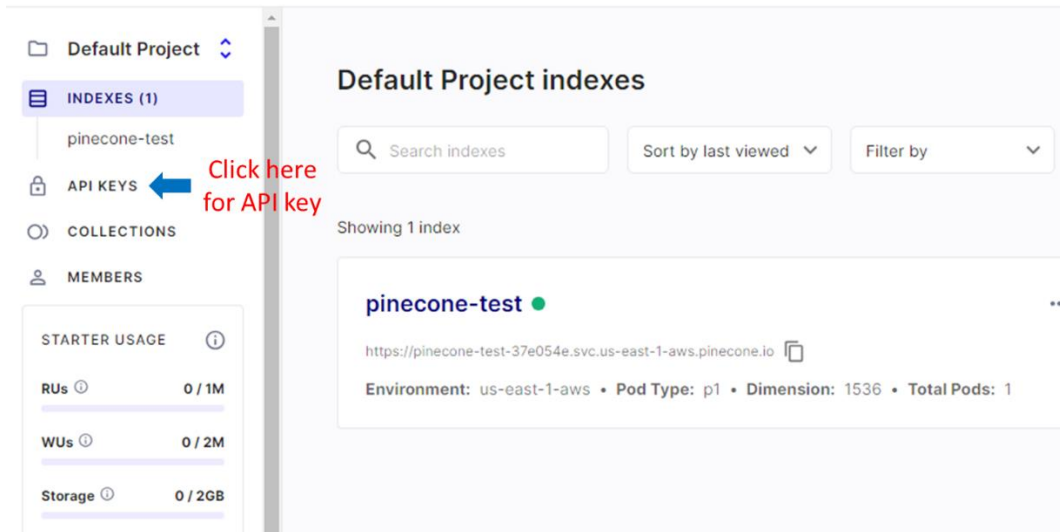


* Upgrade to Standard or Enterprise Plans for full access to all cloud providers

Region [Request a region](#)


Iowa
gcp-starter ← Select a region

(B)



(C)

Fig 2: Pinecone index creation – Register with <https://pinecone.io> to begin **(A)** Open an index creation wizard by clicking on the button as indicated by an arrow **(B)** Then populate necessary fields in the wizard to create an index where namespaces will be stored. Select the free starter plan. **(C)** Obtain the API key by clicking on the button as indicated by an arrow.

6. Launching the app


Using terminal go to the app directory and install the required packages locally by running following command: `yarn install`

After the successful execution of the above command a folder named `node_modules` will be created inside the app directory. Then you can launch the app by running the following command:
`yarn start:both`

This will start the app on <http://localhost:9000/>

7. How to WeiseEule app?

Use the following 3 steps to get answers to your queries. Fig 3 demonstrates this.

- a. Select an LLM from the sidebar under 'Set Parameters' navigation bar (Fig 3, circle 1) .
- b. Select appropriate namespace under 'Set Parameters' navigation bar (Fig 3, circle 2).
- c. Type your question in the chat box on the main window and click the paper plane icon or hit enter (Fig 3, circle 3). If you are unsure about certain parameter just hover your cursor on  symbol which serve as help-point, will pop up an explanatory window.
- d. <optional> Advanced users can play with other params for more sophisticated search experience. See Fig 6 to know how to open 'Advanced param' wizard.

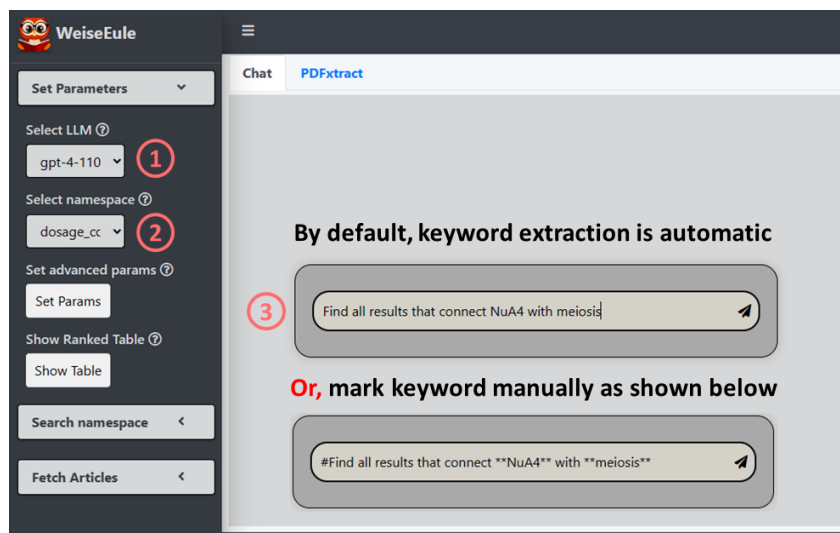


Fig 3: Three essential steps to get answers – Parameters can be set by interacting with the input boxes and menus on the sidebar. A user should type her question in the chat box provided on the main window.

By default, the keywords are extracted from the query automatically. However, sometimes the automatic approach may not select all the desired keywords and you might want to manually extract them. If that is the case, then start the query with a '#' symbol and mark keywords using ** as shown in Fig 3. The idea here is to use the marked keywords from your query and rank

paragraphs/chunks (from research papers) based on the keyword frequency. The rationale is that the chunks with higher frequency of your keyword are more likely to have contexts useful to produce a good quality answer. Let us understand it using an example; suppose the topmost table in Fig 4 is obtained after ranking chunks using cosine similarity-based ranking, where each row corresponds to a chunk and *Nua4*, *meiosis* are the two keywords from our query. If we go by our rationale, clearly the second row should have been ranked highest. To do that, we re-rank the chunks based on their frequencies and only keep chunks where all the keywords are present (See Fig 4).

Before re-ranking

NuA4	meiosis	total_count	rank
0	5	5	1410
1	4	5	759
2	0	2	1399
1	3	4	2080
2	2	4	2954

↓

After re-ranking

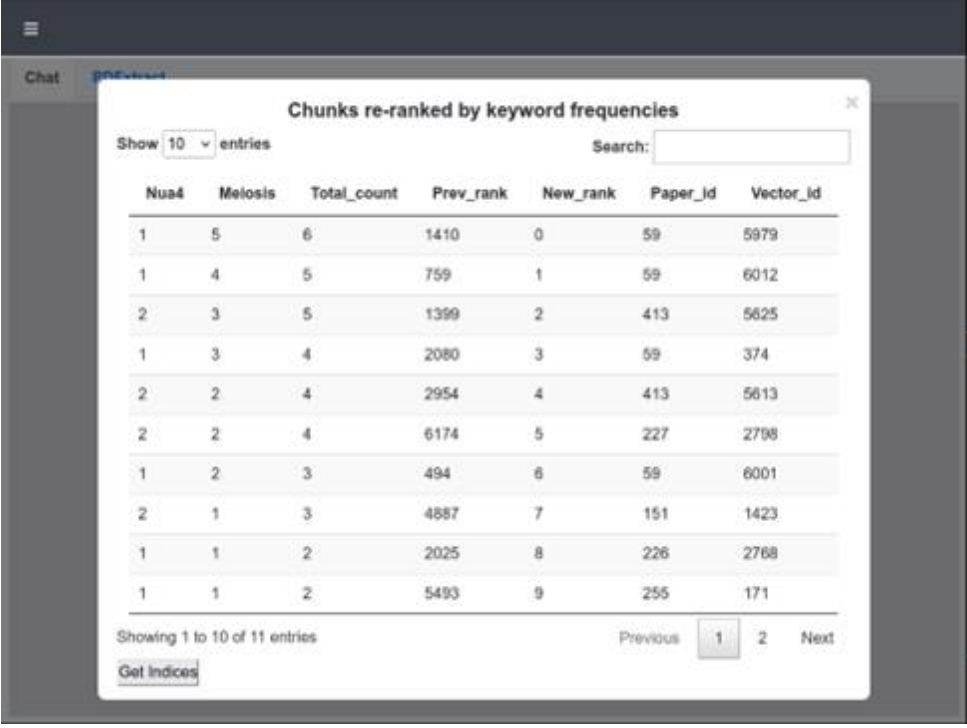
NuA4	meiosis	total_count	prev_rank	new_rank
1	4	5	759	0
1	3	4	2080	1
2	2	4	2954	2

Fig 4: Re-ranking chunks – Elevates the most relevant chunks by analyzing keyword frequencies, ensuring key information surfaces to the top.

7.1 An intermediate step before chat response

Once you write your question in the chat box and hit enter (Fig 3, step 3), there will be an intermediate output before you get your answer. First, you will get the keyword frequency table

(as shown in Fig 5) where each row corresponds to a paragraph/chunk from a research paper and these chunks are re-ranked based on the keyword frequencies (as described earlier). Once you click *Get Indices* button, the app will attempt to obtain an answer to your query by fetching chunks to construct prompt for LLM. By default, the construction of prompts for GPT-3.5 and GPT-4 typically involves considering the top five and top ten rows/chunks, respectively.



Nua4	Melosis	Total_count	Prev_rank	New_rank	Paper_Id	Vector_Id
1	5	6	1410	0	59	5979
1	4	5	759	1	59	6012
2	3	5	1399	2	413	5625
1	3	4	2080	3	59	374
2	2	4	2954	4	413	5613
2	2	4	6174	5	227	2798
1	2	3	494	6	59	6001
2	1	3	4887	7	151	1423
1	1	2	2025	8	226	2768
1	1	2	5493	9	255	171

Showing 1 to 10 of 11 entries

Previous 1 2 Next

Get Indices

Fig 5: Chunks are re-ranked by keyword frequencies – Each row points to a chunk. The frequencies of keywords in each chunk, previous rank and the rank after frequency-based re-ranking is also shown.

7.2 Accessible LLMs

Currently we provide access to following four LLMs:

1. `gpt-4`: This is GPT-4 model and often provides better responses than GPT-3.5 models.

It also costs more.

2. `gpt-4-1106-preview`: This is a variant of GPT-4 family and performance wise comparable to `gpt-4` but slightly cheaper than that model. This model is selected by default.
3. `gpt-3.5-turbo`: This is GPT-3.5 model, cost-effective but not precise for complex queries.
4. `gpt-3.5-turbo-1106`: This is an updated variant of GPT-3.5 and may show slightly improved performance over `gpt-3.5-turbo`.

We recommend using `gpt-4-1106-preview`, at least for automated keyword selection.

7.3 Advance parameters setup

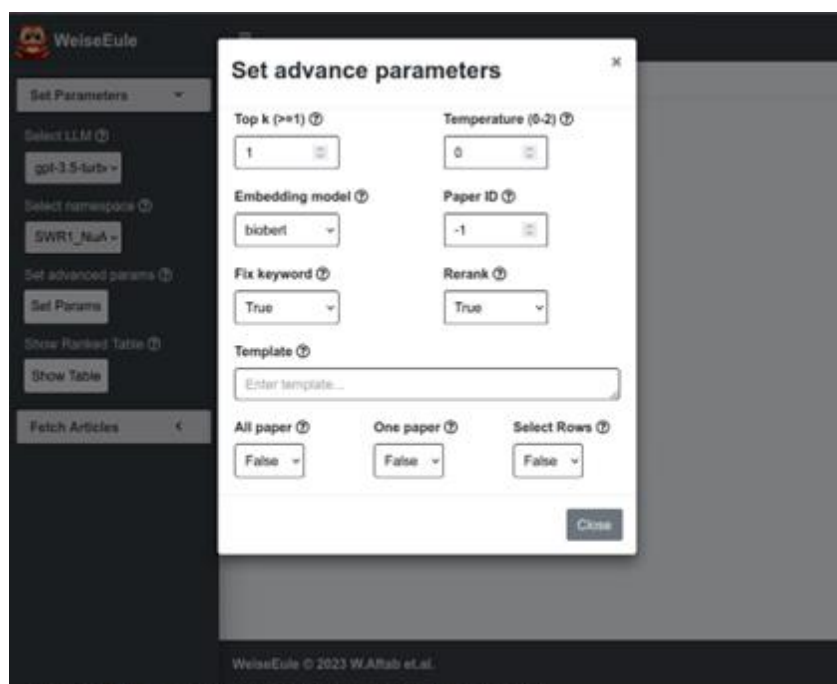
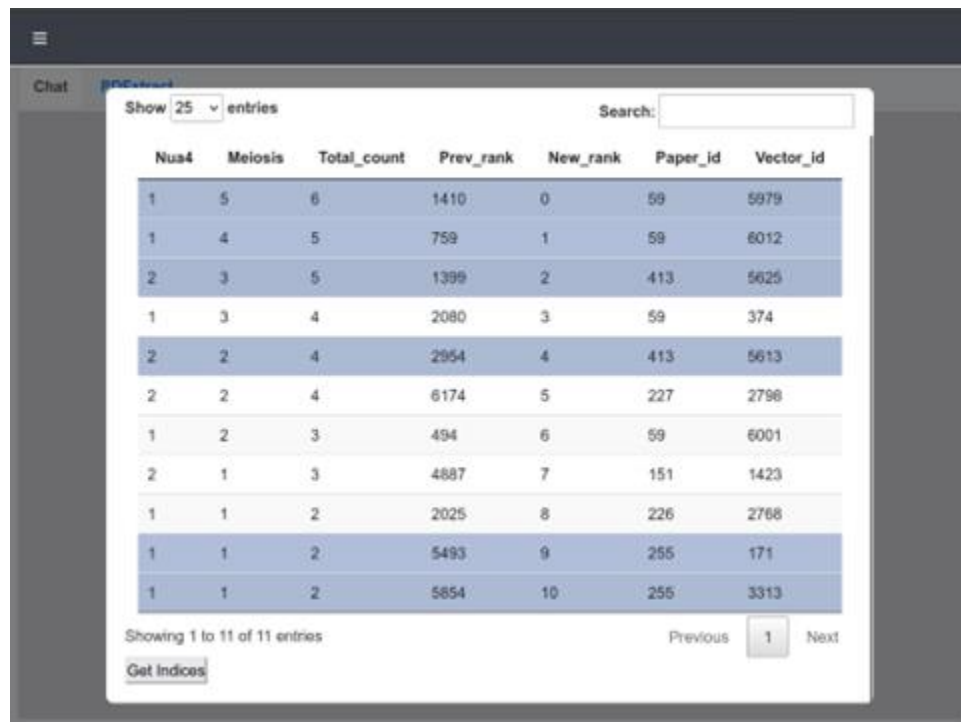


Fig 6: Advance parameters – Use the 'Set Params' modal to configure the app for more sophisticated search.

You can also set advanced params to configure the app for more sophisticated search. To do that, click on *Set Params* button and launch a modal with advance params (see Fig 6). For example, if you want to select custom rows (by default it is top n), then set *Select Rows* to *True* and after that pull up *keyword frequency* table by clicking *Show Table* button. Then you can select custom rows just by clicking on them as depicted in Fig 7 and once you do that, then clicking '*Get Indices*' button will only use the selected rows/chunks to construct prompt. Hover mouse on the question mark symbols (help-points) to understand the meaning of each param and set accordingly. However, remember that this step is optional and most of the time you do not need to change params on this modal.



Nua4	Meiosis	Total_count	Prev_rank	New_rank	Paper_id	Vector_id
1	5	6	1410	0	59	5979
1	4	5	759	1	59	6012
2	3	5	1399	2	413	5625
1	3	4	2080	3	59	374
2	2	4	2954	4	413	5613
2	2	4	6174	5	227	2798
1	2	3	494	6	59	6001
2	1	3	4887	7	151	1423
1	1	2	2025	8	226	2768
1	1	2	5493	9	255	171
1	1	2	5854	10	255	3313

Showing 1 to 11 of 11 entries

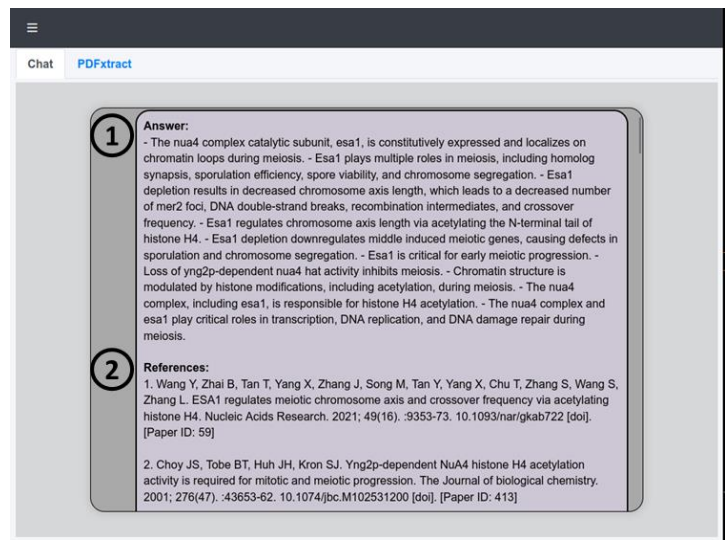
Previous 1 Next

Get Indices

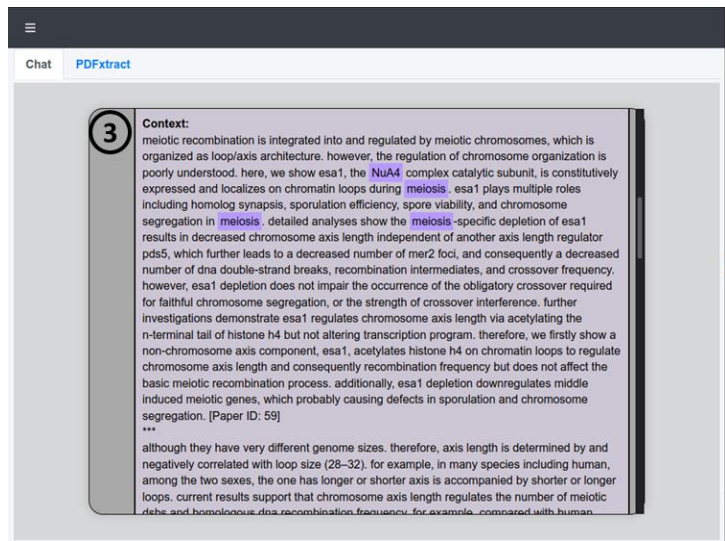
Fig 7: Select custom rows by clicking on them – Upon selection a row background changes to blue color. After that when *Get Indices* is clicked, only these selected rows will be considered for the construction of prompt.

7.4 Chat response

The response contains 3 parts: answer, reference and context as depicted in Fig 8.



(A)



(B)

Fig 8: Illustrates a chat response composed of three segments – (A) The answer is followed by references, each marked with its respective paper ID. **(B)** Below the references lies the context,

where keywords are highlighted, and each context section is identified by the corresponding paper ID. Contexts are separated from one another by the '***' symbol.

7.5 Look-up papers in namespace

It is also possible to search whether a paper is present in a selected namespace as shown in Fig 9. The search result not only reveals the paper's presence/absence but also indicates whether the full text or just the abstract is accessible for the specified PubMed id in that namespace. This is useful when judging the quality of a response or investigating why a certain paper is not included in the reference.

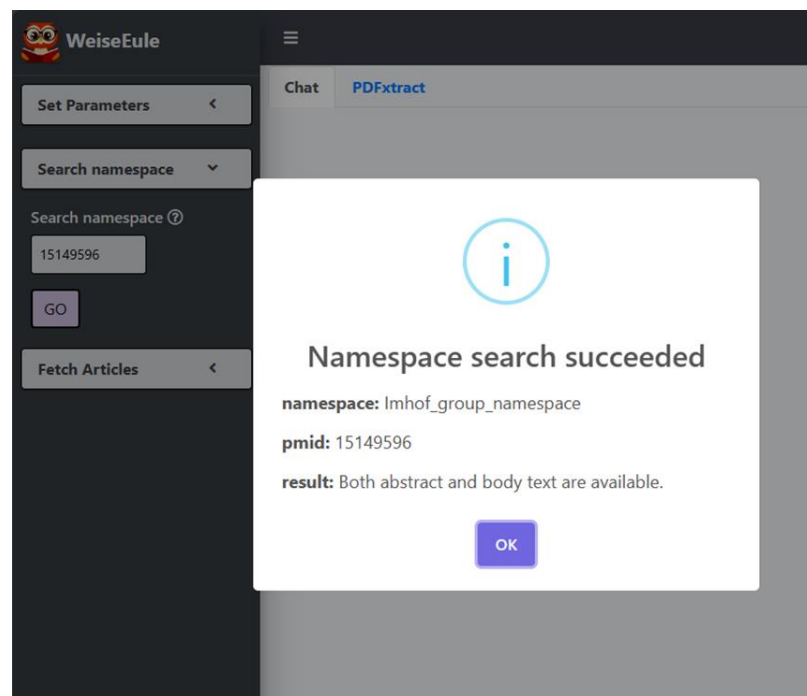


Fig 9: Illustrates search namespace feature – Input a valid PubMed id and hit *GO* button.

7.6 Exception/Error messages

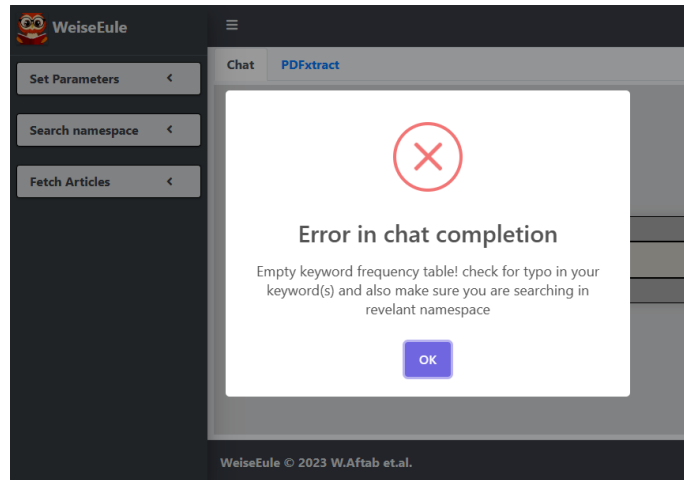


Fig 10: An error message when chat completion fails.

The error messages come with an explanation. For example, when you see a message as shown in Fig 10, it means that the keywords could not be found in any of the chunks in your namespace. There could be several reasons:

- a. You have selected a namespace that is not relevant to your query.
- b. There are typos in your keywords.
- c. Or sometimes you have a paper in mind which you expect to show up, but that paper may not be part of your namespace. In that case, it is recommended to confirm if the paper is indeed part of the selected namespace by using ‘search namespace’ feature of the app (See Fig 9). Sometimes only abstract will be available and it is possible that the keywords are not mentioned there.

8. Namespace creation using GUI

Using WeiseEule GUI, articles can be obtained from PubMed as illustrated in Fig 11. The namespace “name” is determined based on the entered keyword(s) to make the naming relevant. For example, if the entered keyword is “dosage compensation,” then the namespace is named as

“dosage_compensation_c2000”, where “c2000” part at the end implies that in this namespace, articles are chunked into 2000 characters chunks. In the future, we will provide separate fields on the GUI to change the chunk size and naming a namespace. Currently, you can achieve that by modifying the `chunk_size` and `namespace` variables in the `pycodes/fetch_articles.py` located in the app directory (see Fig 1).

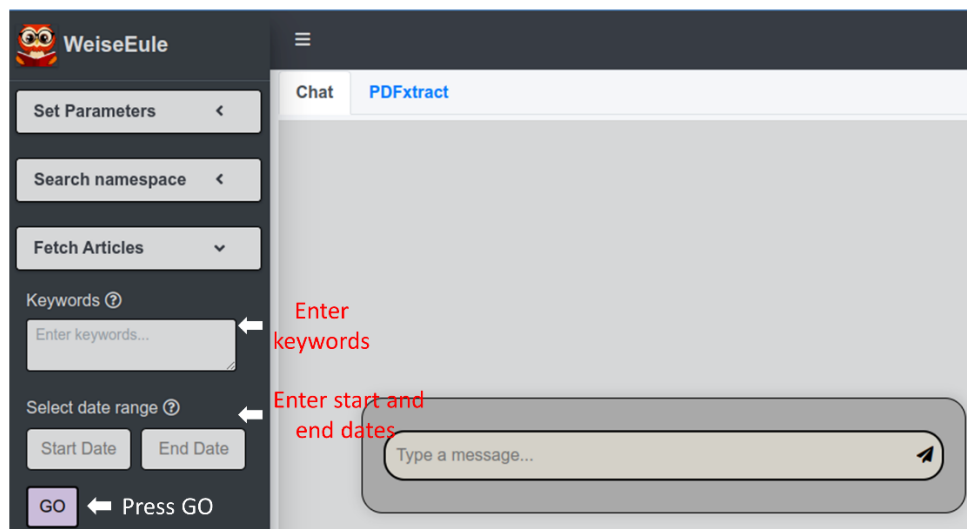


Fig 11: Article accumulation – Only those papers that contain keywords either on the title or abstract are considered for fetching.

Note, this approach can create a namespace by fetching full text of those articles that are available (in xml format) on the PMC server and for others it only fetches the abstract from PubMed. However, for many articles in PMC the full text is not available on the PMC in xml format. In those cases, we have a python script called `PDF2NAMEPSACE_with_PMIDs.py` to download those articles as PDFs and then extract texts to augment the existing namespace. Run it either in a terminal or in an IDE such as Spyder.

9. Namespace creation for custom PDFs

In some cases, a user might already have a bunch of PDFs in a local folder which he/she wants to use as a knowledge base. In that case, namespace creation is only possible by running the `PDF2NAMEPSACE_NO_PMIDS.py` file either in a terminal or in an IDE such as Spyder. In future we will also support this feature via GUI. Once the namespace is generated querying it can be done as explained earlier (see Fig 3).

10. Downloading all the metadata from vector DB for PES2

It is important to note that in order to use PES2, one must download all the metadata from the corresponding namespace in the vector DB. To do that run the following python code provided in the app directory: `pycodes/extract_all_meta_pinecone.py` either in a terminal or in an IDE such as Spyder. The code will prompt you to enter a namespace and you must enter a valid namespace hosted in the pinecone server. Without this step, PES2 will not work.

11. Parameter settings used to evaluate PES1

To configure the WeiseEule app to work in PES1 mode, first select `gpt-4-1106-preview` as LLM. Then launch the *Advance parameters* modal (See 7.3 Advance parameters setup) and set the `Rerank` and `Top k` parameters as `False` and `10` respectively as shown in Fig 12.

×

Set advance parameters

Top k (>=1) ?

10

Use top 10 chunks

Temperature (0-2) ?

0

Embedding model ?

biobert

Paper ID ?

-1

Rerank ?

False

To evaluate PES1, set this to 'False'

Fix keyword ?

True

Template ?

Enter template...

All paper ?

False

One paper ?

False

Select Rows ?

False

Close

Fig 12: Parameter settings used to evaluate PES1 – Launch the ‘Set Params’ modal and set the Rerank and Top k parameters.

12. Parameter settings used to evaluate PES2

To configure the WeiseEule app to work in PES2 mode, you must first select `gpt-4-1106-preview` as the LLM. Then set the Rerank parameter to `True` as shown in Fig 13. Upon proceeding, an intermediate table with chunks re-ranked using keyword frequency will appear (See 7.1 An intermediate step before chat response). After that when you click on *Get Indices* button, by default, top 10 chunks is selected as context to answer the query.

Set advance parameters [X]

Top k (>=1) ? <input type="text" value="1"/>	Temperature (0-2) ? <input type="text" value="0"/>	
Embedding model ? <input type="text" value="biobert"/>	Paper ID ? <input type="text" value="-1"/>	
Rerank ? <input type="text" value="True"/>	Fix keyword ? <input type="text" value="True"/>	
Template ? <input type="text" value="Enter template..."/>		
All paper ? <input type="text" value="False"/>	One paper ? <input type="text" value="False"/>	Select Rows ? <input type="text" value="False"/>

[Close]

Fig 13: Parameter settings used to evaluate PES2 – Launch the `Set Params` modal and set the Rerank parameter.

Note:

1. Guidelines in this documentation is tested on Ubuntu 22.04 Linux OS, and it should also work on Mac OS. Windows users can also set up the app environment by following these guidelines with some adjustments, particularly during installation of packages and exporting environment variables as these processes differs between Windows and Linux/Mac systems.
2. In future, we will add documentation on how to set up WeiseEule on Windows. For future updates, users are requested to check the GitHub repo. Additionally, users can contribute by requesting features or submitting issues on the GitHub page.