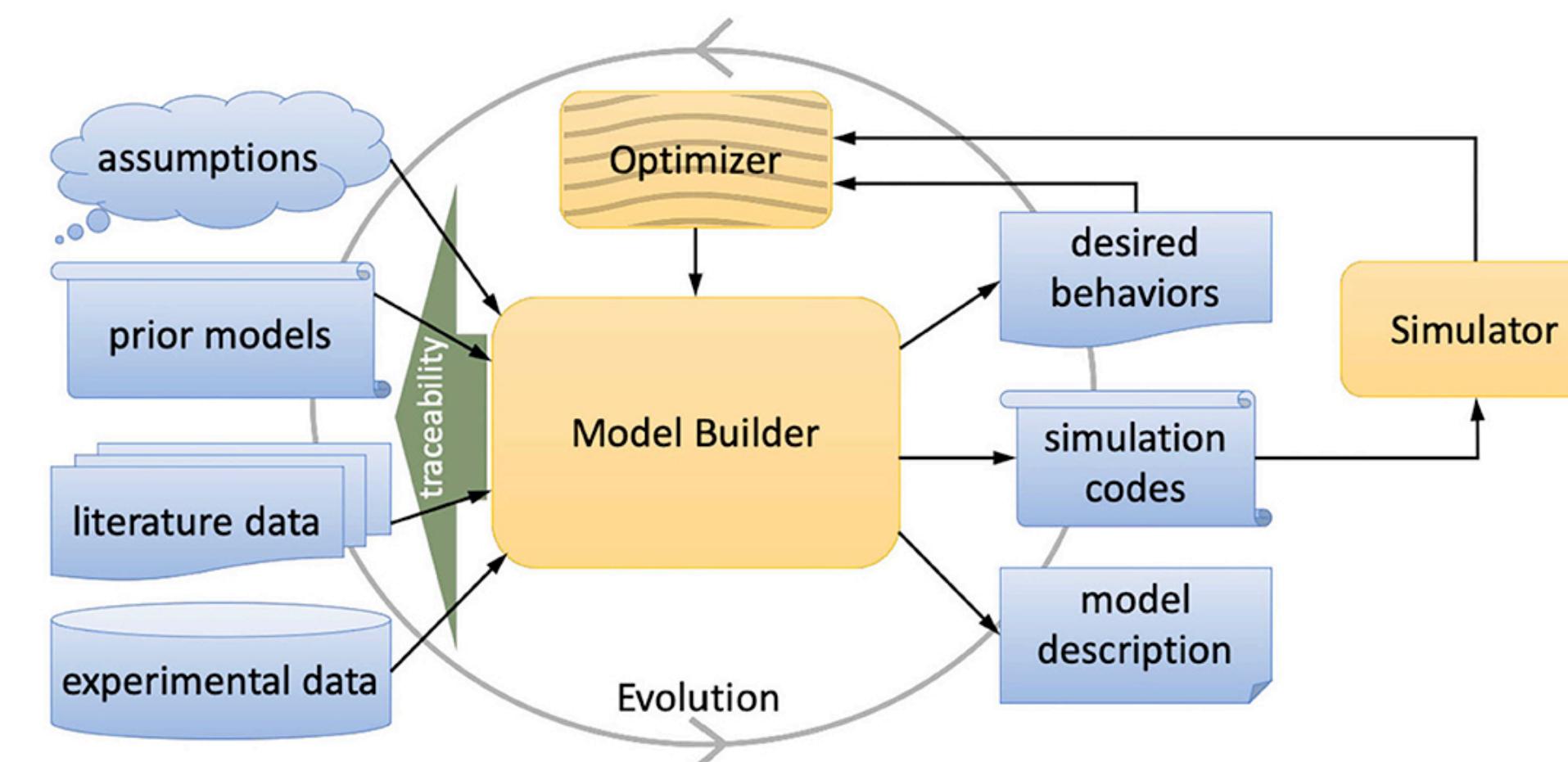


Building BrainModeling Workflows

A proof-of-concept framework



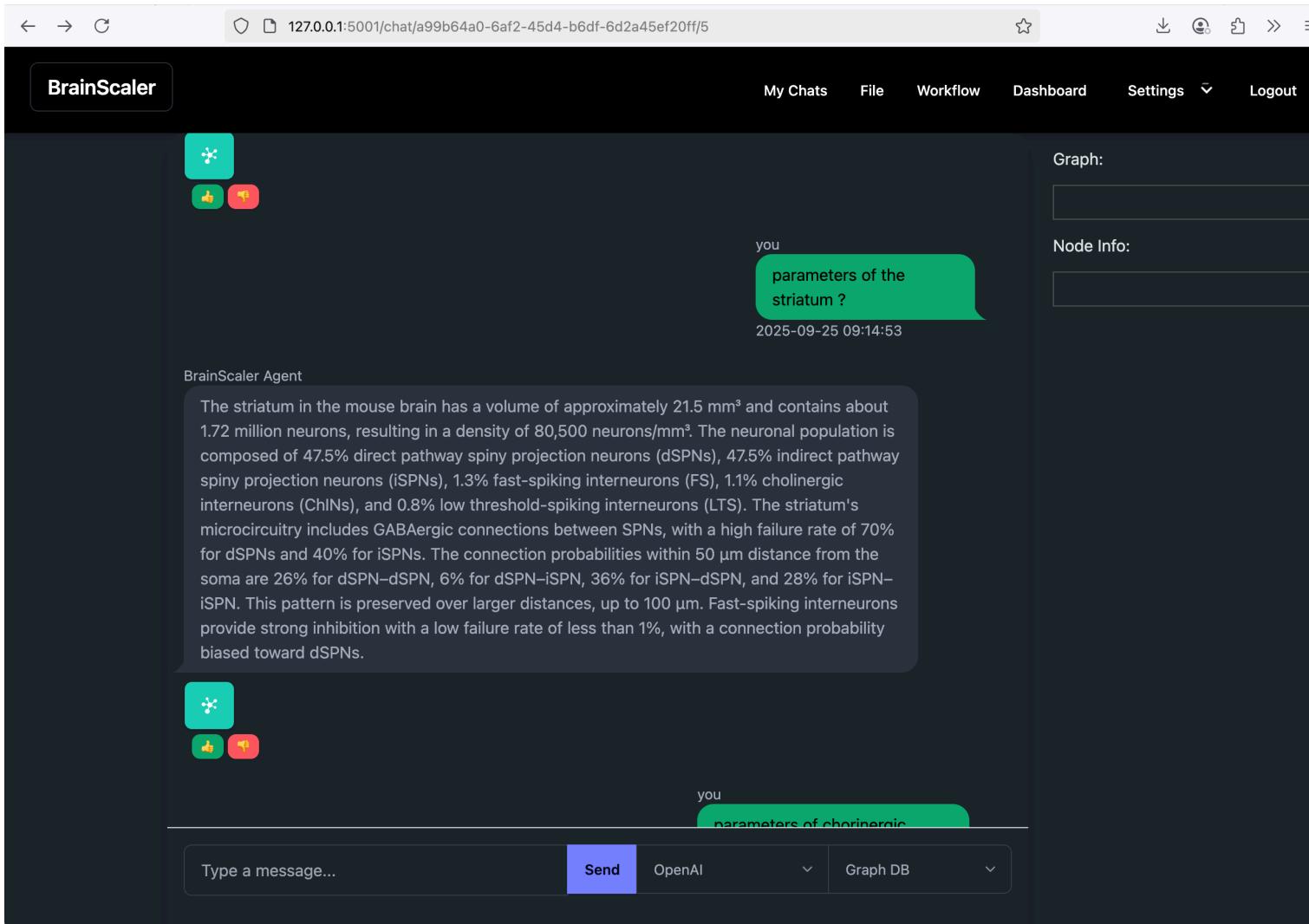
OIST

Carlos Gutierrez, Kenji Doya, Neuro-workflow dev.team

Sept. 28 2025

Prototypes - Proof-of-Concept

Chatbot

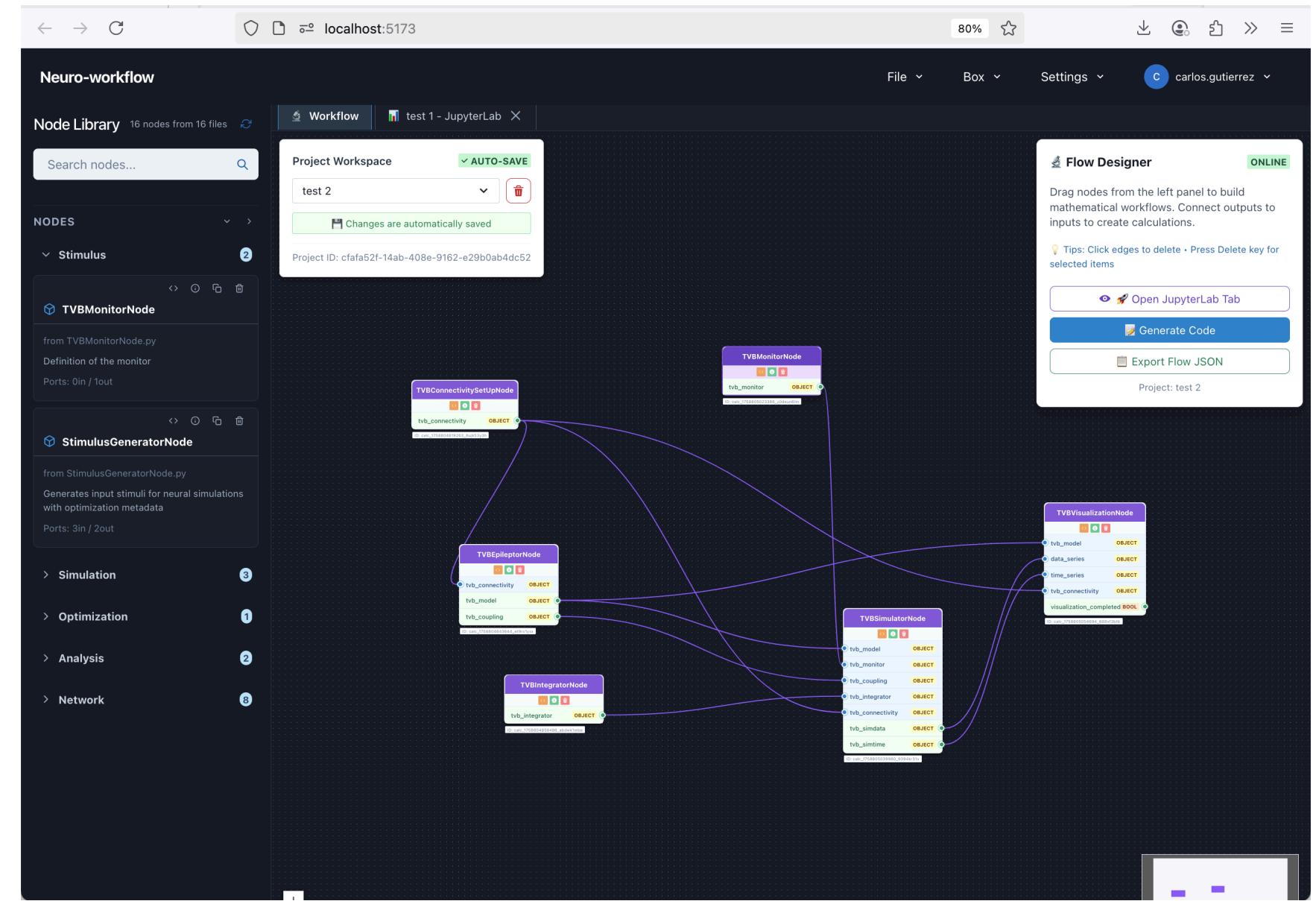


https://github.com/oist/brainscaler_frontend

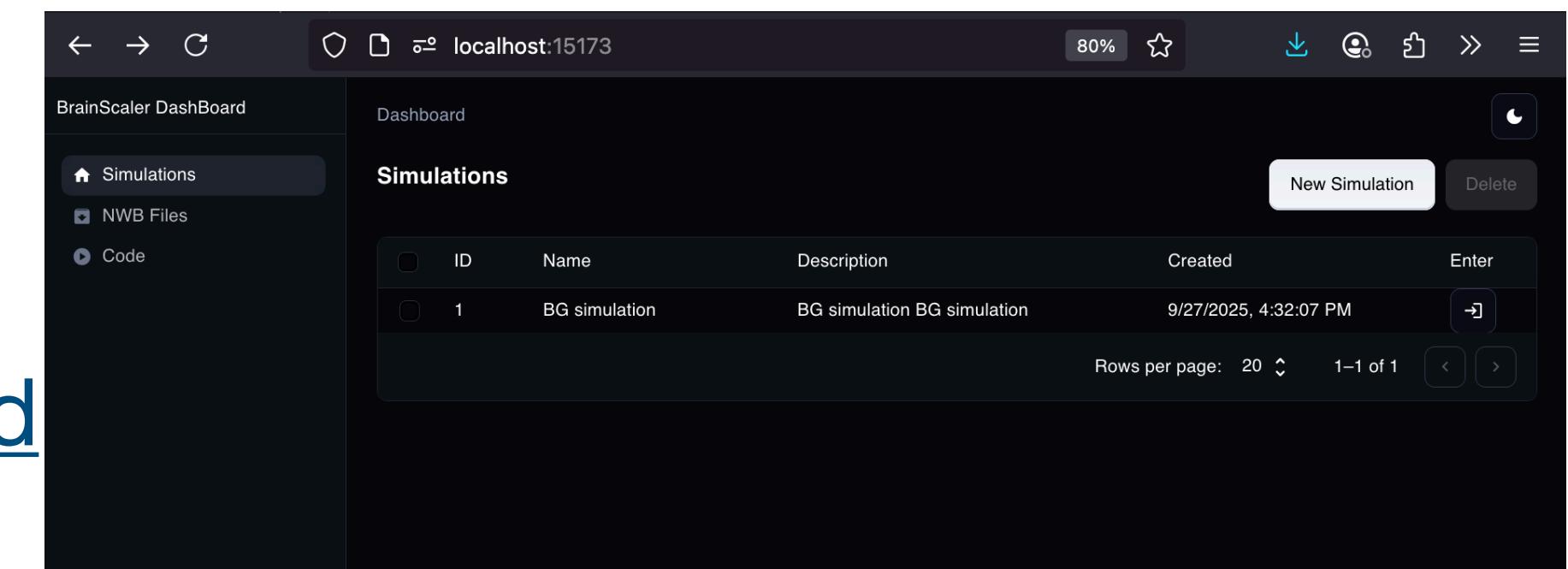
Dashboard

https://github.com/oist/brainscaler_dashboard

Neuro-workflow



<https://github.com/oist/neuro-workflow>



Access: send your GitHub user (or GitHub linked email) to ->> carlos.gutierrez@oist.jp

Prototypes - Proof-of-Concept

1 - Clone the repositories in your local laptop/PC

2 - Install instructions

Chatbot

Check “Installation in Docker”

https://github.com/oist/brainscaler_frontend/blob/main/README.md

NeuroWorkflow

<https://github.com/oist/neuro-workflow/blob/main/gui/README.md>

Dashboard

https://github.com/oist/brainscaler_dashboard/blob/main/README.md

3 - Install instructions include **.env** files set-up.

Please use the files at the link below and rename each of them as **.env**

<https://www.dropbox.com/scl/fo/vytgfc7gsz2kcte0b55n5/AJg4RYR0fjFNyrD2uHCe870?rlkey=1o28ybx93ccrqi3afje9i2fa&st=7wiuw86a&dl=0>

Prototypes - Proof-of-Concept

NeuroWorkflow

IPs addresses

Chatbot

https://github.com/oist/brainscaler_frontend
<http://0.0.0.0:5001>

NeuroWorkFlow

<https://github.com/oist/neuro-workflow/tree/main/gui>
<http://localhost:5173/>

Django project folder
/Users/carlosengutierrez/Downloads/neuro-workflow/gui/workflow_backend/django-project

Dashboard

https://github.com/oist/brainscaler_dashboard/tree/main
<http://localhost:15173/>

Prototypes - Proof-of-Concept

NeuroWorkflow

Available nodes

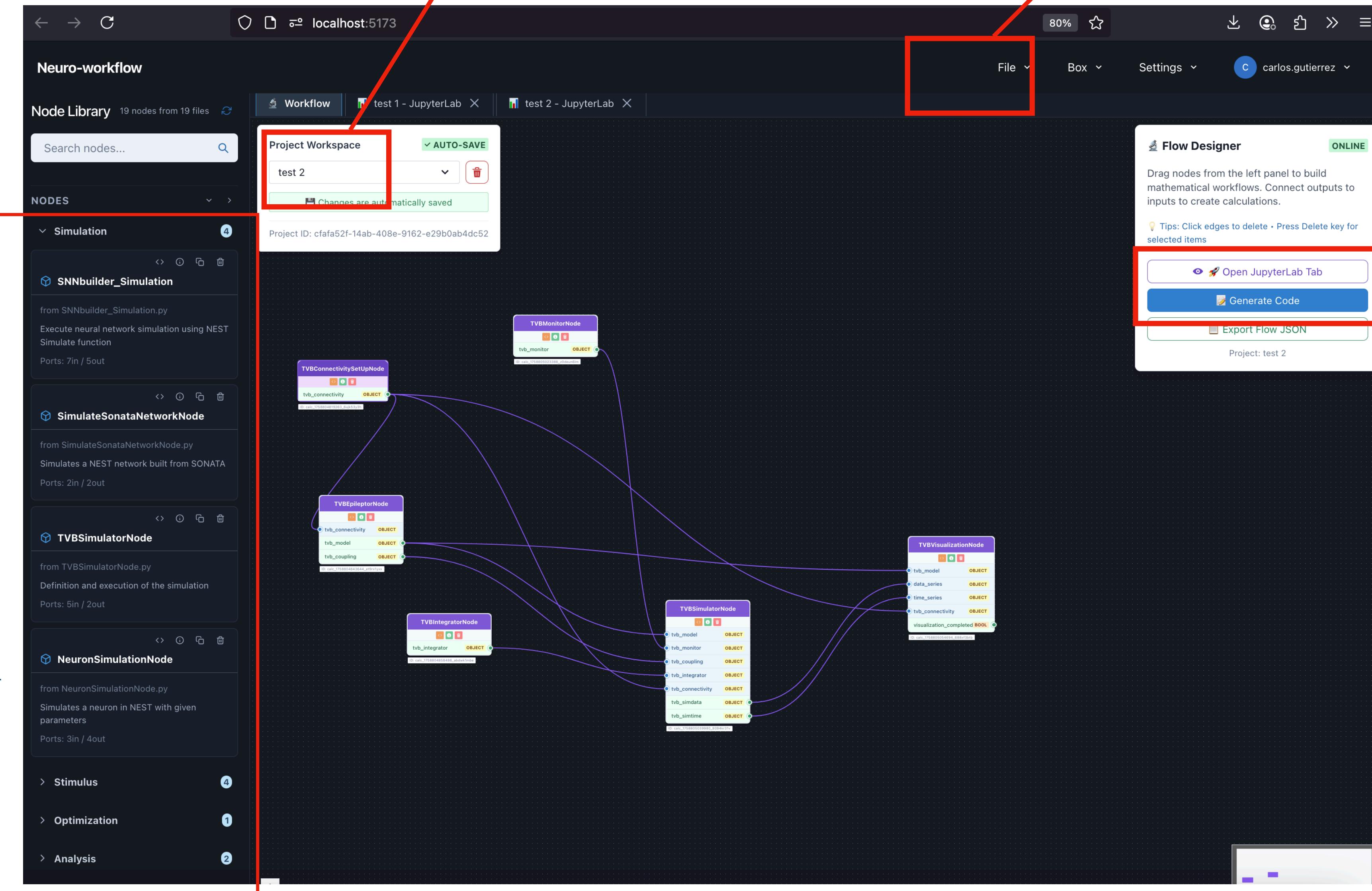
Node Schema:

https://github.com/oist/neuro-workflow/blob/main/NODE_SCHEMA.md

Custom node docs:

https://github.com/oist/neuro-workflow/blob/main/CUSTOM_NODE_TUTORIAL.md

2 - Select your Project



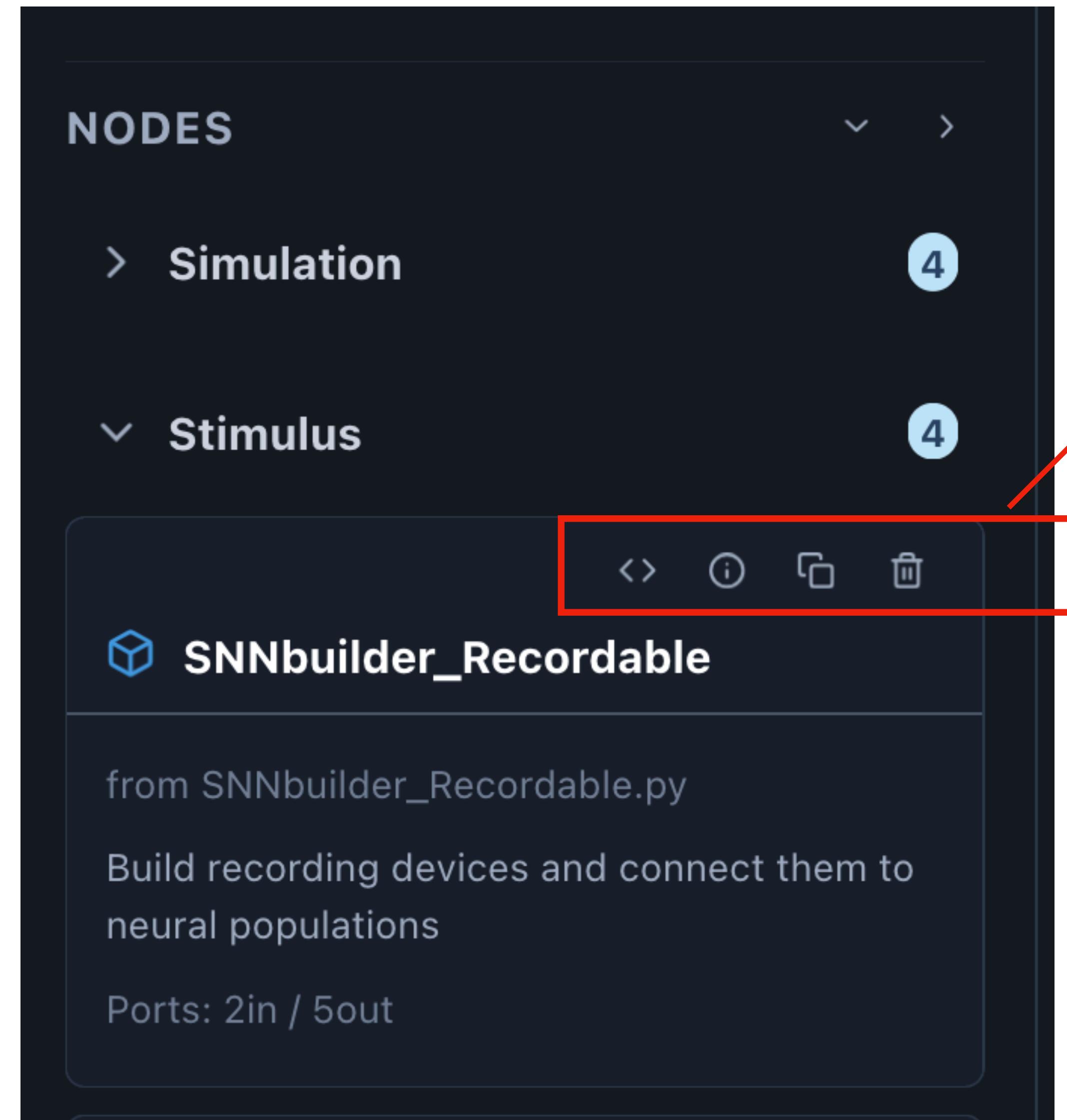
1 - Create a Project (a workflow)

3 - Run a workflow

*First Generate Code
*And next Open Jupyter Lab

Prototypes - Proof-of-Concept

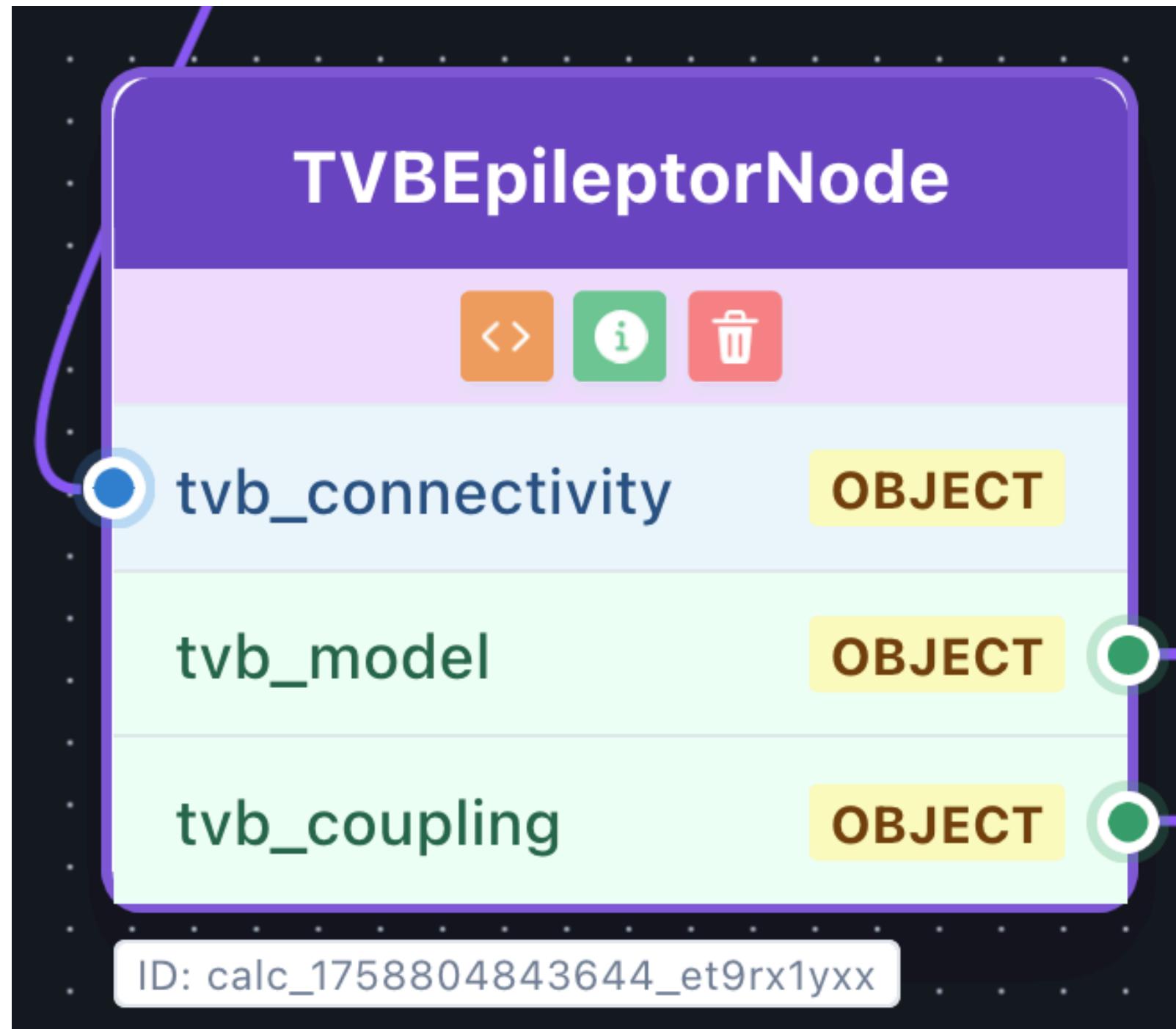
NeuroWorkflow



- Examine the custom node
- The node Information (schema)
- Duplicate the node with a different name (If you want to change or create a new node based on the current node)

Prototypes - Proof-of-Concept

NeuroWorkflow



<> - Examine the custom node

i - The node Information (schema)
Use this to UPDATE parameters of the node (configure)

↓

Node Details: TVBEpileptorNode

TVBEpileptorNode
Node ID: calc_1758804843644_et9rx1yxx

Inputs

tvb_connectivity : object

Outputs

tvb_model : object
tvb_coupling : object

Methods

"model_initialization"
description: Initialize the Epileptor Model based on Heatmap of epileptogenicity and related parameters
inputs: **["tvb_connectivity"]**
outputs: **["tvb_model"]**

"coupling_initialization"
description: Note that the global coupling parameter value for each submodel is already set in the initialisation of the model (see variables Ks and K_rs above), so here we set the value of coupl
outputs: **["tvb_coupling"]**

Parameters

"tau"
description: is this tau_0 ? time scale of.
default_value: **1000**
constraints: **{}**

"K_rs"
description: global coupling of the 3rd population (xrs,yrs),(Jirsa et al., 2014)
default_value: **1**
constraints: **{}**

Update the parameter "tau"

"tau"
description: is this tau_0 ? time scale of.
default_value: **1000**
constraints: **{}**

Prototypes - Proof-of-Concept

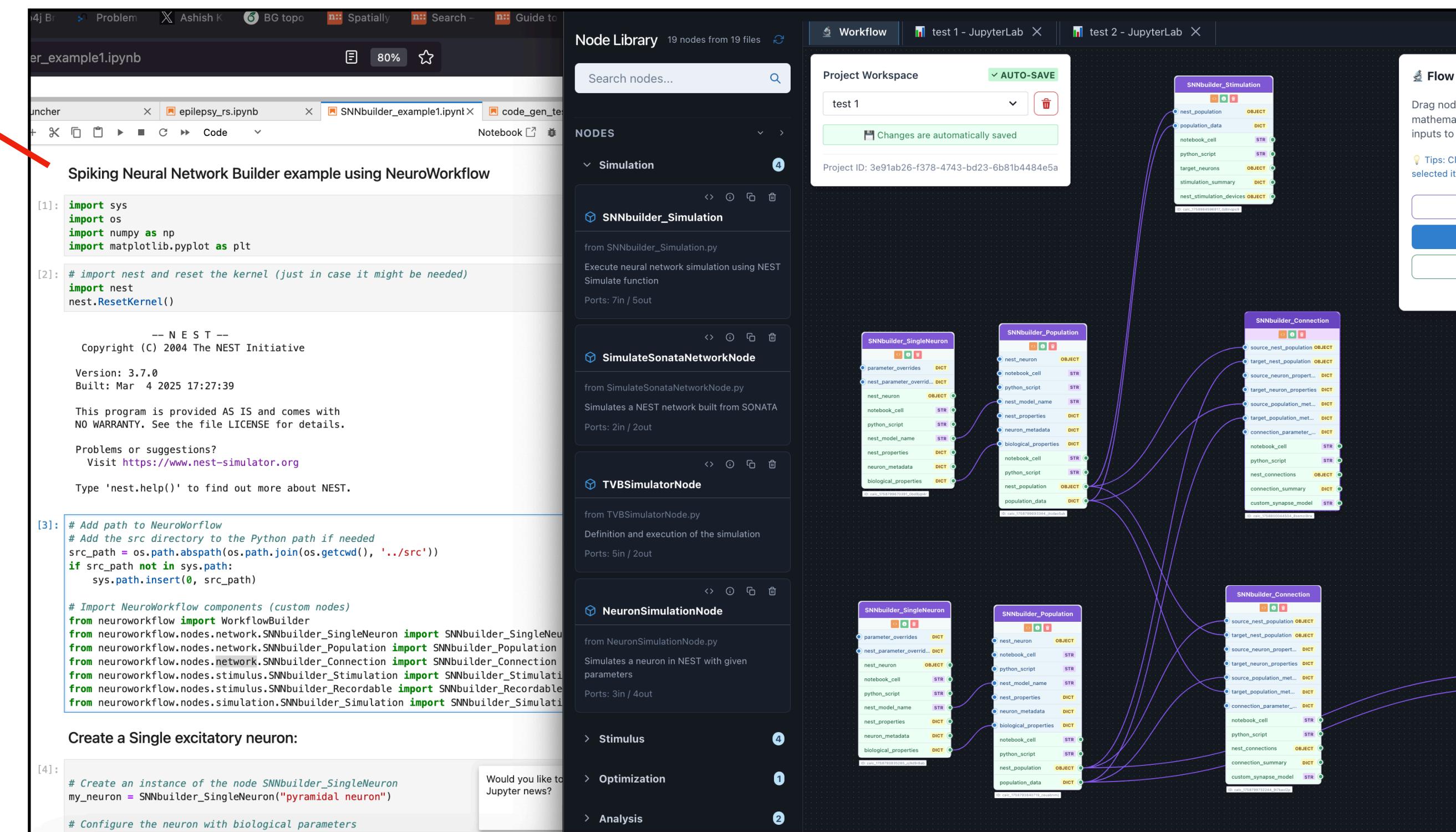
NeuroWorkflow

Examples for the Hackathon at Folder Notebook

<https://github.com/oist/neuro-workflow/blob/main/README.md>

- `epilepsy_rs.ipynb` - Interactive example of epileptic resting state using the virtual brain TVB
- `SNNbuilder_example1.ipynb` - Interactive example of Spiking Neural Network building using SNNbuilder custom nodes

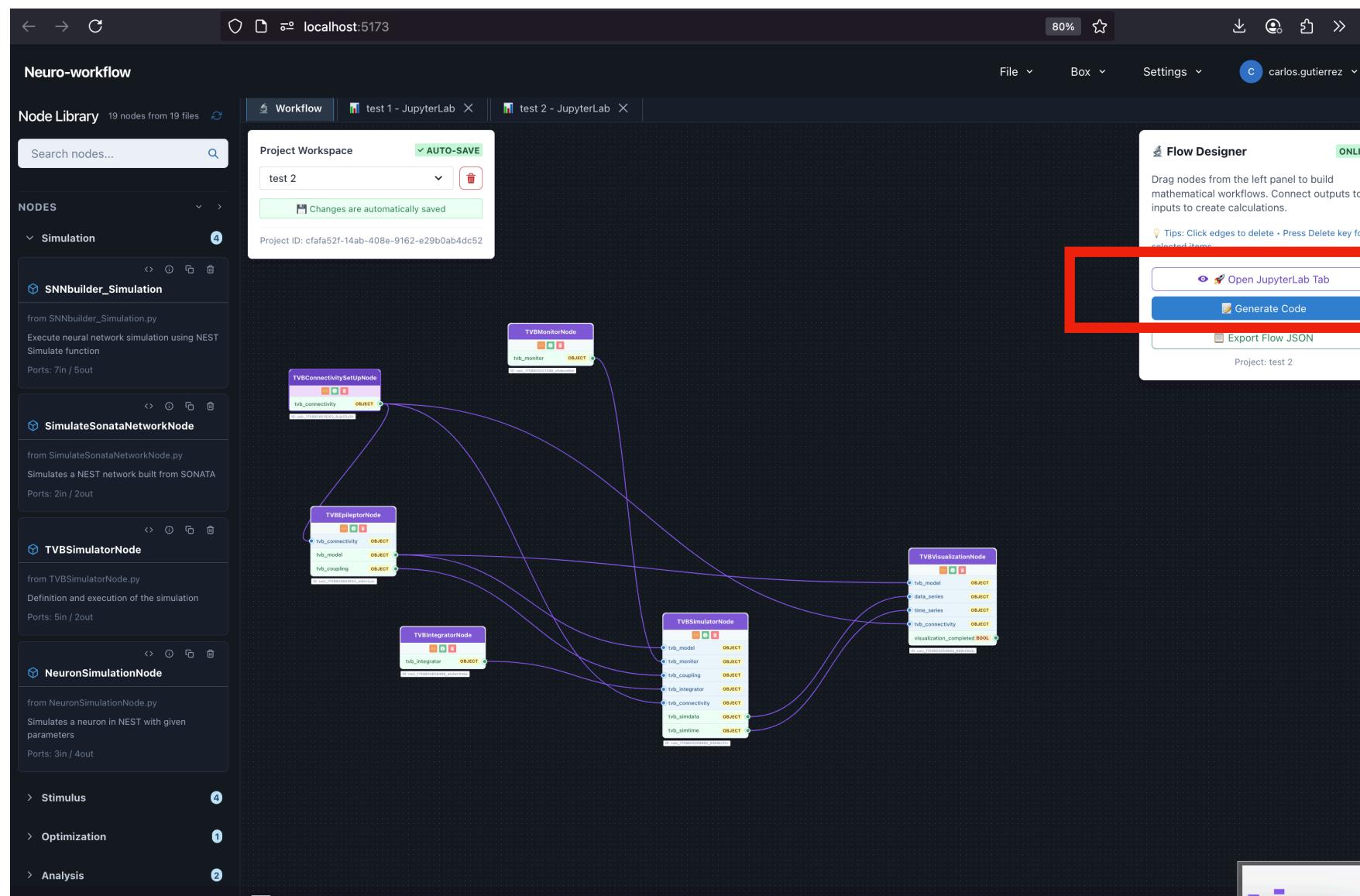
Go through the .ipynb example and build the workflow using the graphical interface.



Prototypes - Proof-of-Concept

NeuroWorkflow

Examples for the Hackathon at Folder Notebook



*First Generate Code
*And next Open Jupyter Lab

The code generation has some bugs.
We prepared “reviewed” generated code
for both examples:

<https://www.dropbox.com/scl/folder/o859x7s580oakliod9h5u/AHULmFzrgvYxcXdlHohDT0o?rlkey=27bh0br5bfhfeqxg9tcna2sxf&st=lyhr4rvl&dl=0>

*SNNbuilder_example1.ipynb → Test1_reviewed.ipynb

*epilepsy_rs.ipynb → Test2_reviewed.ipynb

Jupyter credentials:
user1
password

Prototypes - Proof-of-Concept

NeuroWorkflow

Challenge: build nodes for your model (or any model).

(We are working on LLM based conversion: unstructured python → nodes)

Node Schema:

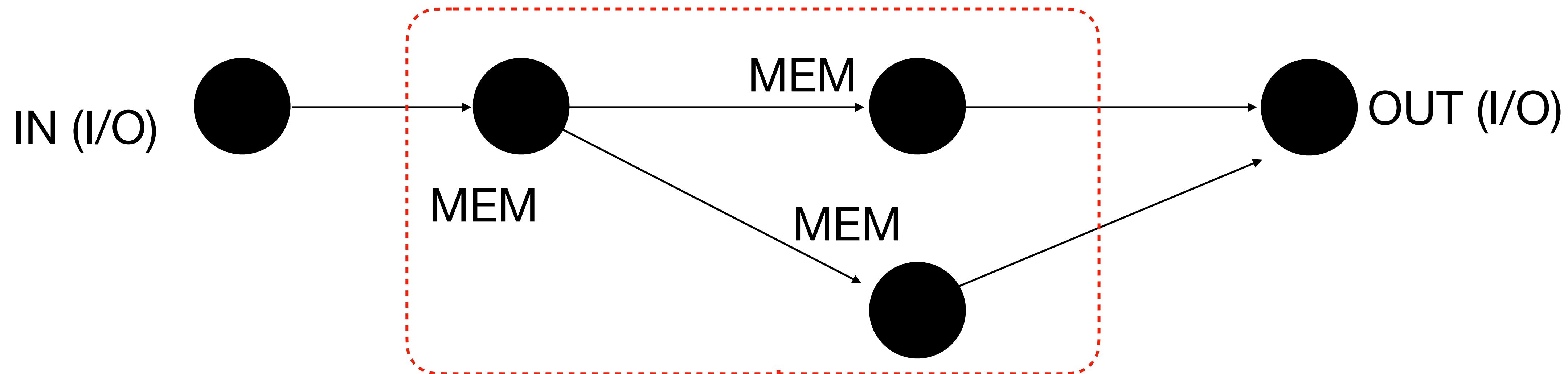
https://github.com/oist/neuro-workflow/blob/main/NODE_SCHEMA.md

Custom node docs:

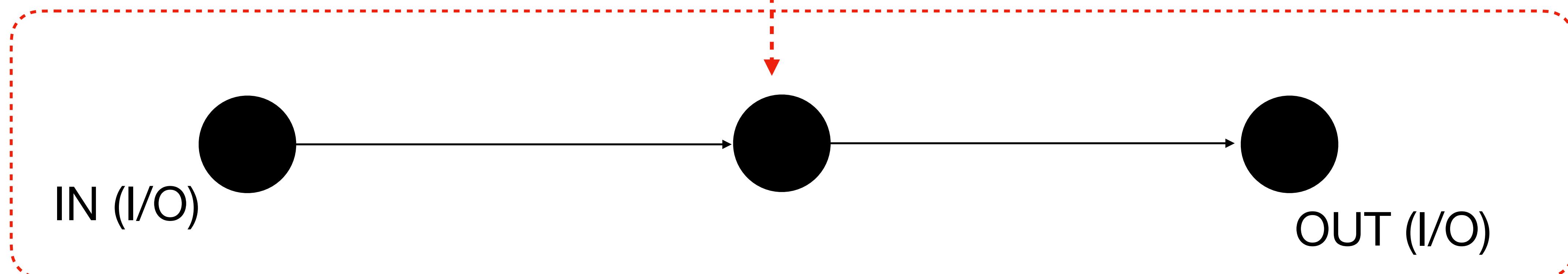
https://github.com/oist/neuro-workflow/blob/main/CUSTOM_NODE_TUTORIAL.md

NeuroWorkflow
(Node - Edges)

Representation for Brain Models Interpretability/Usability and AI-readiness

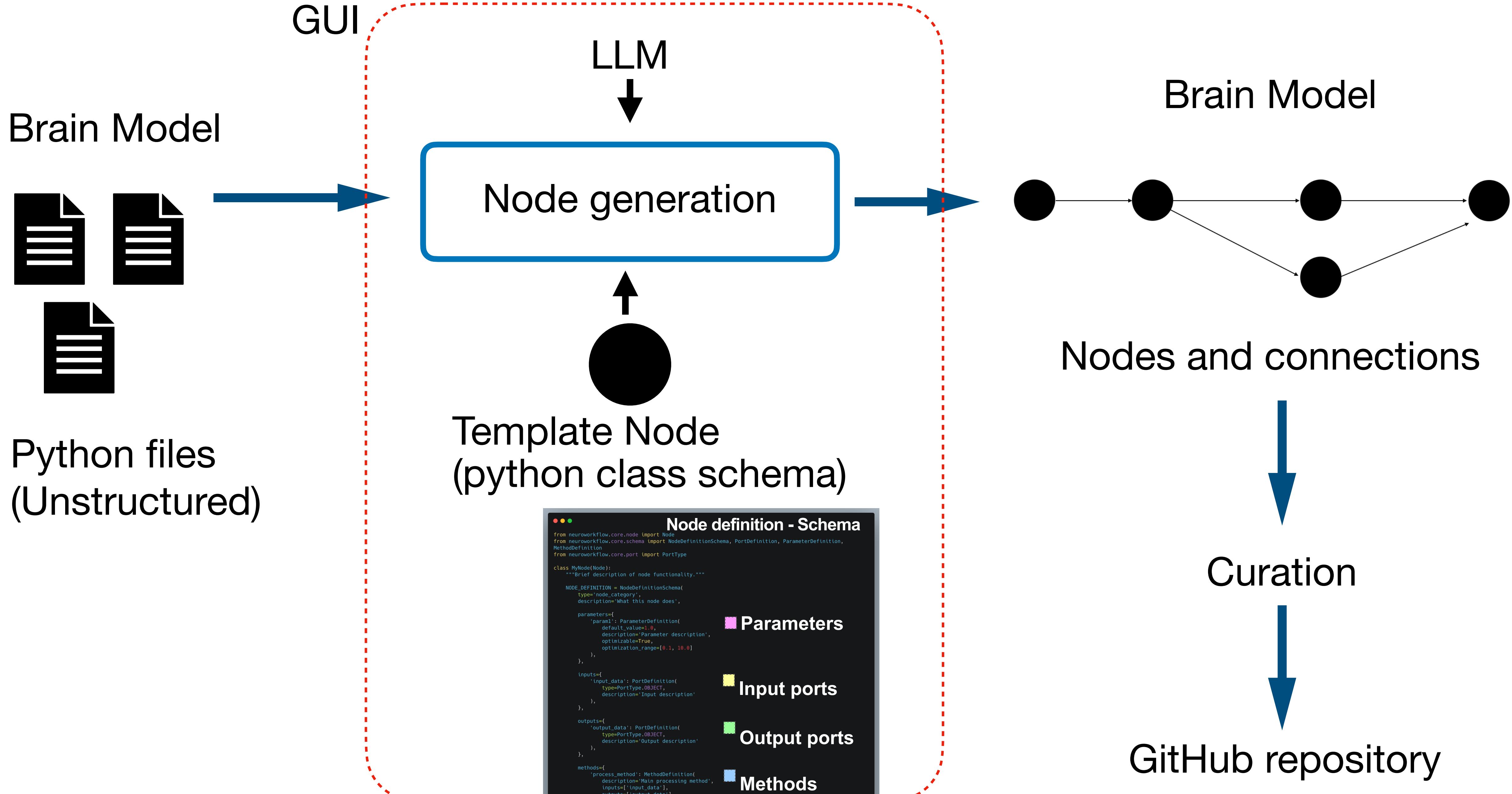


Execution on cluster HPC (i.e: **representation as SnakeMake Rules**)



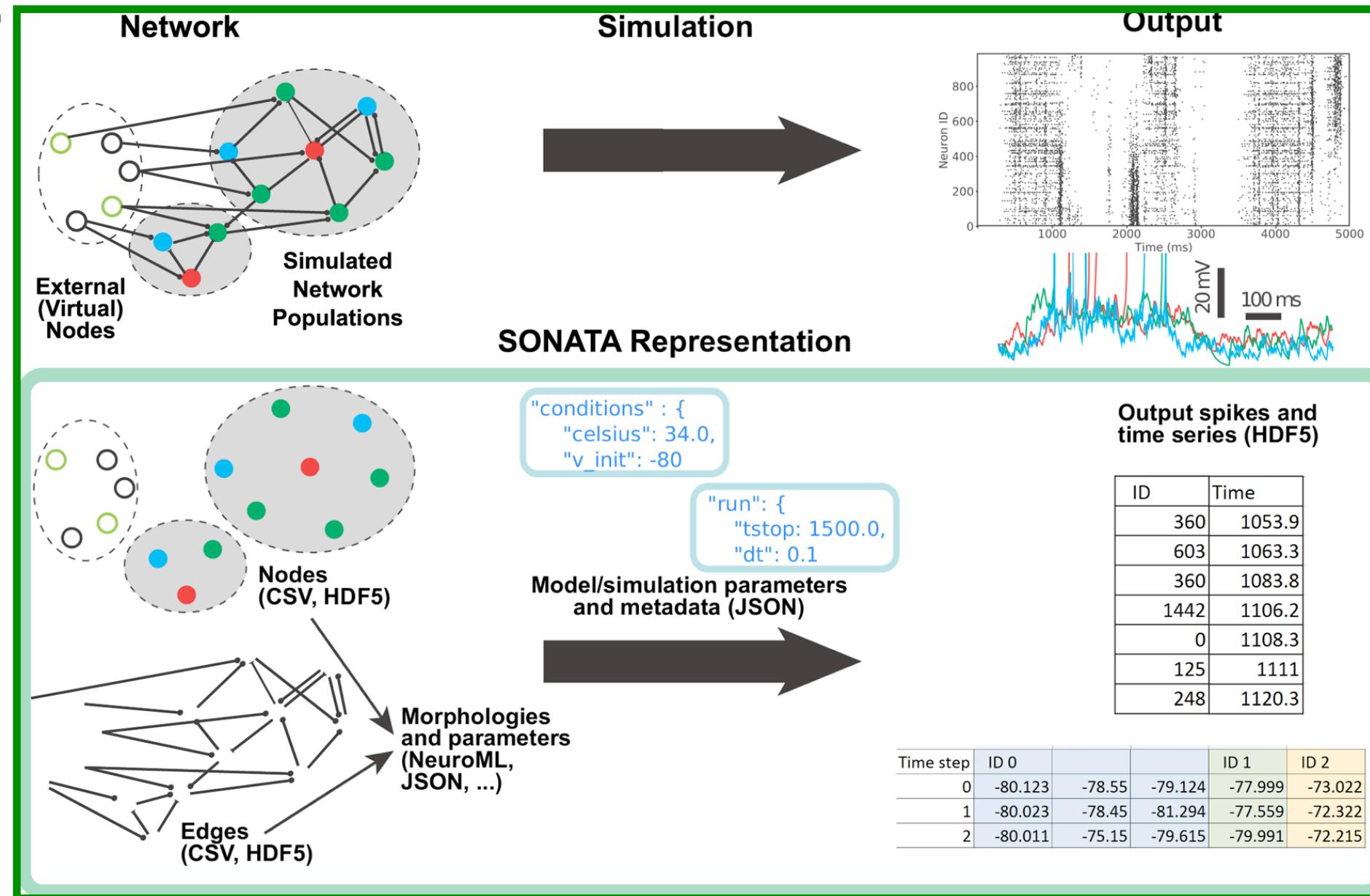
Scaling-up Nodes Generation

NeuroWorkflow



SONATA:

SONATA is a standardized format for large-scale neural network models jointly developed by the Allen Institute for Brain Science and the Blue Brain Project. It's designed for:



Key Components:

- **Internal nodes:** Network neurons with 3D coordinates and properties
- **External nodes:** Input/stimulus sources
- **Internal-internal edges:** Synapses between network neurons
- **External-internal edges:** Input connections
- **Spike trains:** External input data

Example in NEST:

https://github.com/oist/neuro-workflow/blob/main/notebooks/01_Basic_Simulation.ipynb

Basic Neural Simulation with NeuroWorkflow

This notebook demonstrates how to create a simple neural simulation workflow using the NeuroWorkflow library. We'll build a workflow that:

1. Creates a neural network from SONATA format
2. Simulates the network using NEST

Setup

First, let's make sure we can import the NeuroWorkflow library. If you've installed it with pip, you can import it directly. Otherwise, we'll add the source directory to the Python path.

```
import sys
import os
import numpy as np
import matplotlib.pyplot as plt

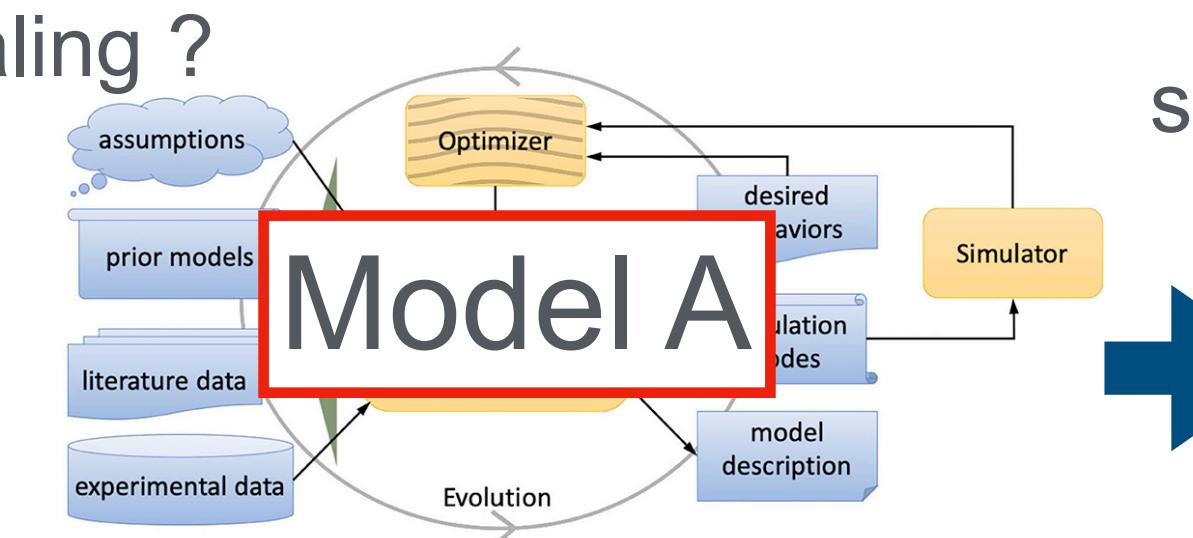
# Add the src directory to the Python path if needed
src_path = os.path.abspath(os.path.join(os.getcwd(), '../src'))
if src_path not in sys.path:
    sys.path.insert(0, src_path)

# Import NeuroWorkflow components
from neuroworkflow import WorkflowBuilder
from neuroworkflow.nodes_network import BuildSonataNetworkNode, BuildSonataNetworkNode
```

SONATA: What we would like to do ?

BM2 modeling platform

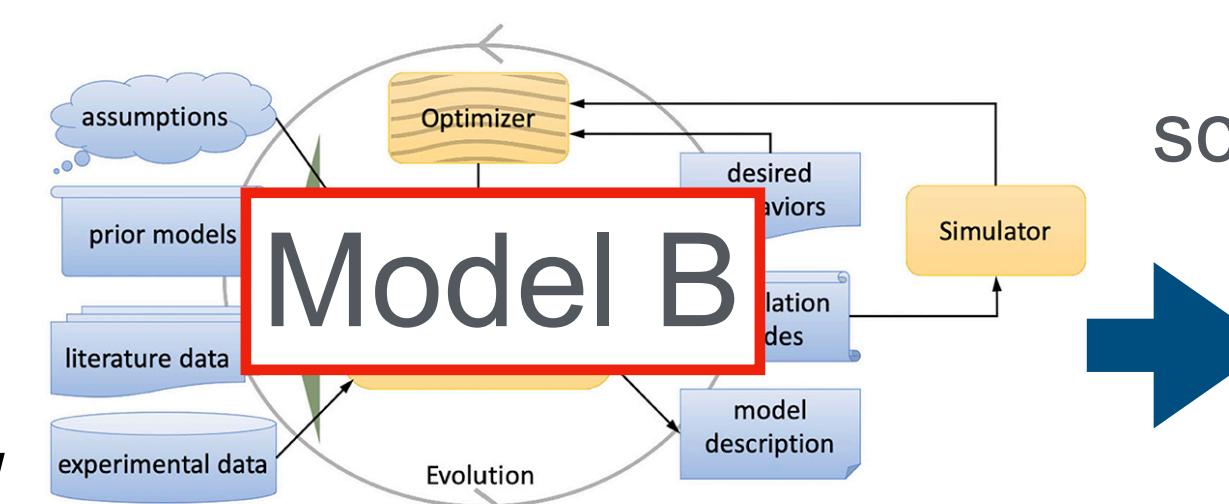
SONATA representation
(**External** Large model
Data / Model A)



A potential solution :
build custom nodes in
neuro-workflow for
BMTK, NEST support
sonata.

Explore, Change Parameter,
Optimize, Experiment,
Change the model, add components,
(Real size, Scaled down size?)

A potential solution : build
custom nodes in neuro-workflow
for **SNNbuilder-SONATA**
conversion



Explore, Change Parameter,
Optimize, Experiment,
Change the model, add components,
(Real size, Scaled down size?)

SONATA representation
(Large model **Model A'**)

scaling up ?

SONATA representation
(Large model **Model B**)

SONATA:

Current State:

- SNNbuilder: Custom workflow-based approach with components like populations, stimuli, connections, recording, simulation
- SONATA: Standardized format with internal/external nodes, edges, configuration files
- Both represent the same underlying concepts but with different structures and semantics

SNNbuilder → SONATA mappings:

- SNNbuilder_Population → SONATA internal nodes
- SNNbuilder_Stimulation → SONATA external nodes + input spike trains
- SNNbuilder_Connection → SONATA internal-internal edges or external-internal edges
- SNNbuilder_Recordable → SONATA reports configuration
- SNNbuilder_Simulation → SONATA simulation configuration

Data Structure Differences: SNNbuilder uses NEST-native objects, SONATA uses HDF5/CSV

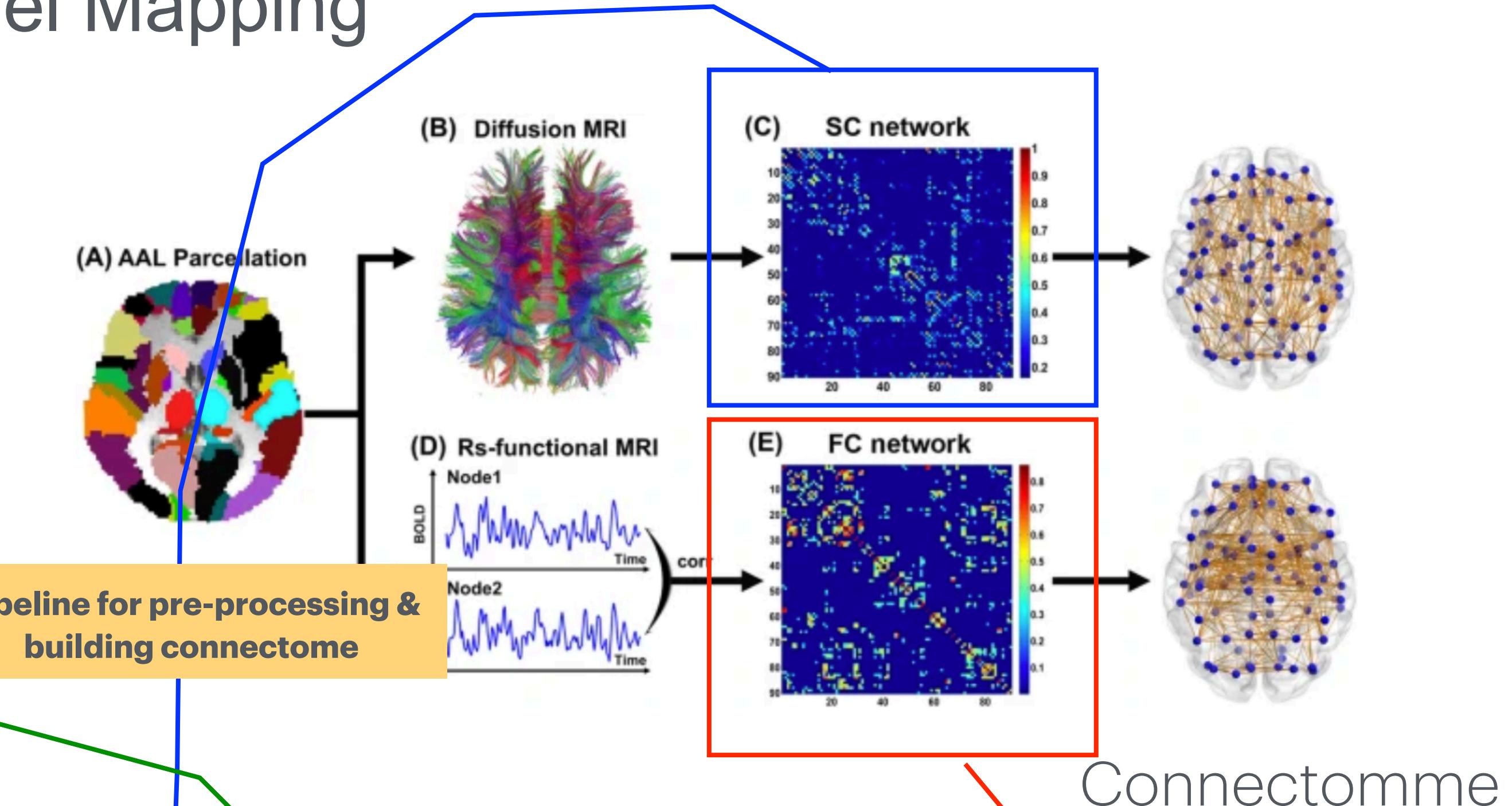
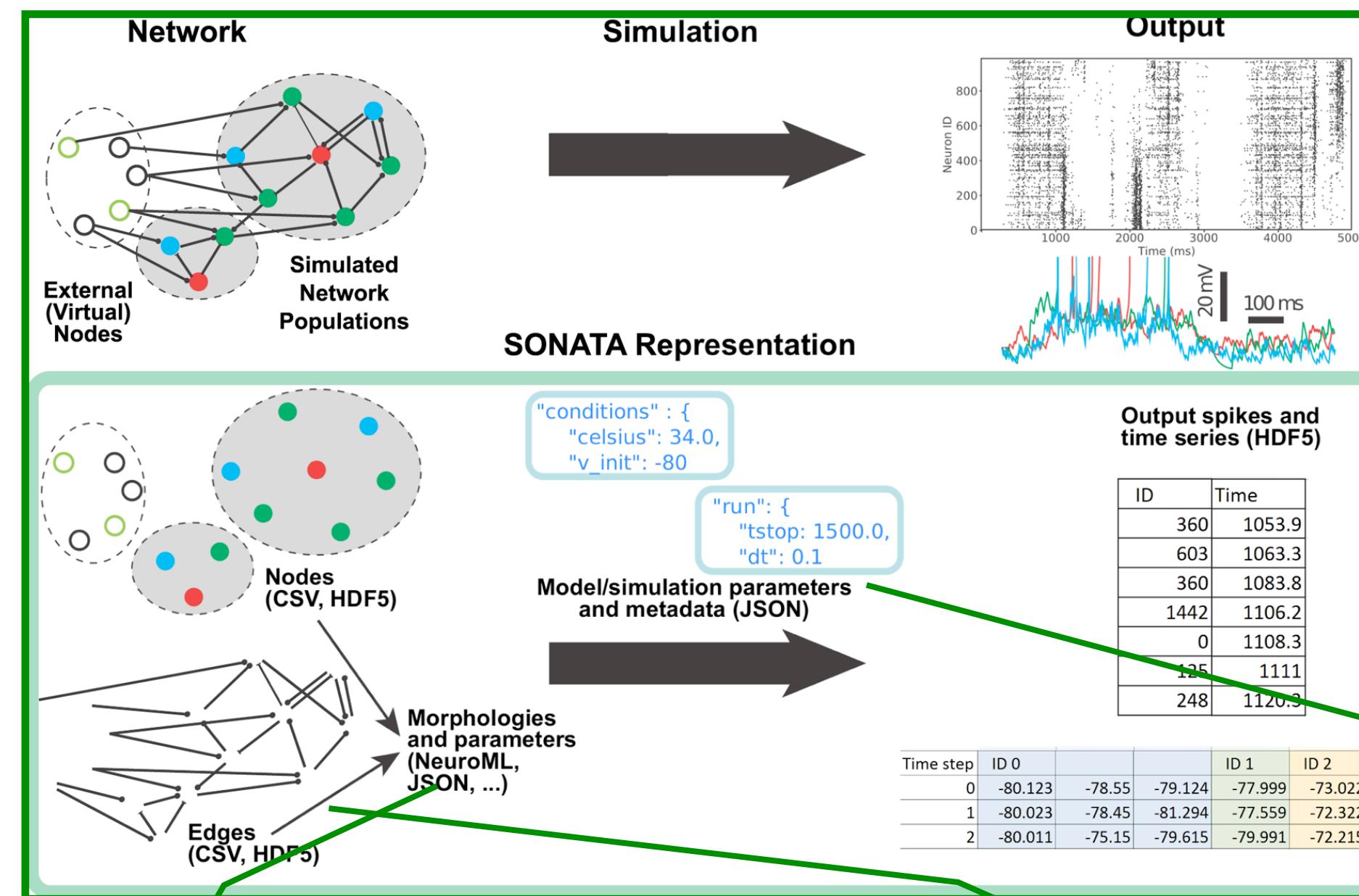
Bidirectional Conversion: Need both SNNbuilder→SONATA and SONATA→SNNbuilder

SONATA:

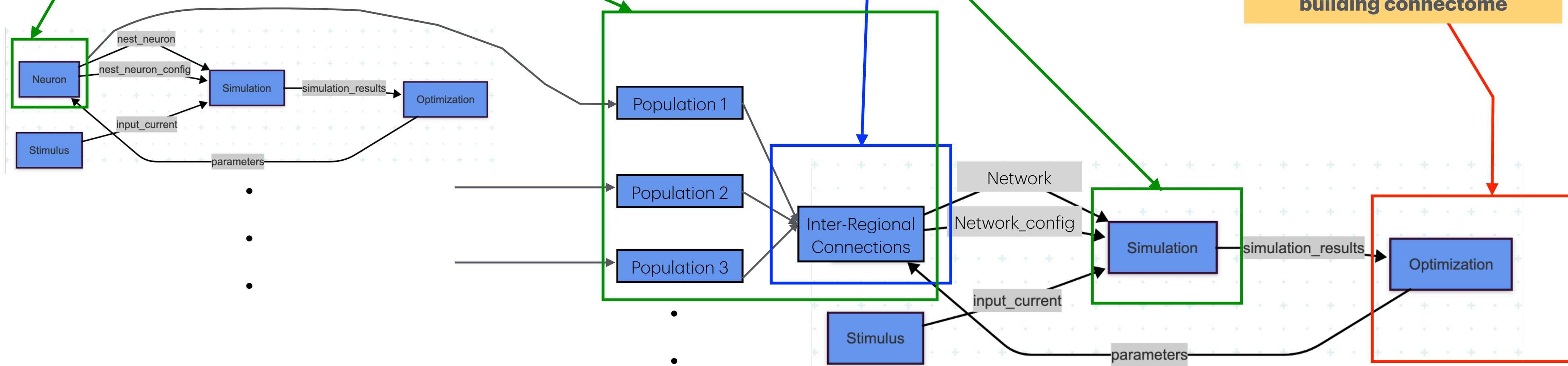
Transformations Nodes from NeuroWorkflow to Sonata:

- **Connection Rules** → Explicit connectivity matrices with proper indexing
- **Parametric Stimuli** → Pre-generated spike trains with timing preservation
- **Spatial Structures** → 3D coordinates with extent calculations
- **Model Parameters** → Comprehensive neuron/synapse parameter mappings

SONATA: Example of Data-to-Model Mapping



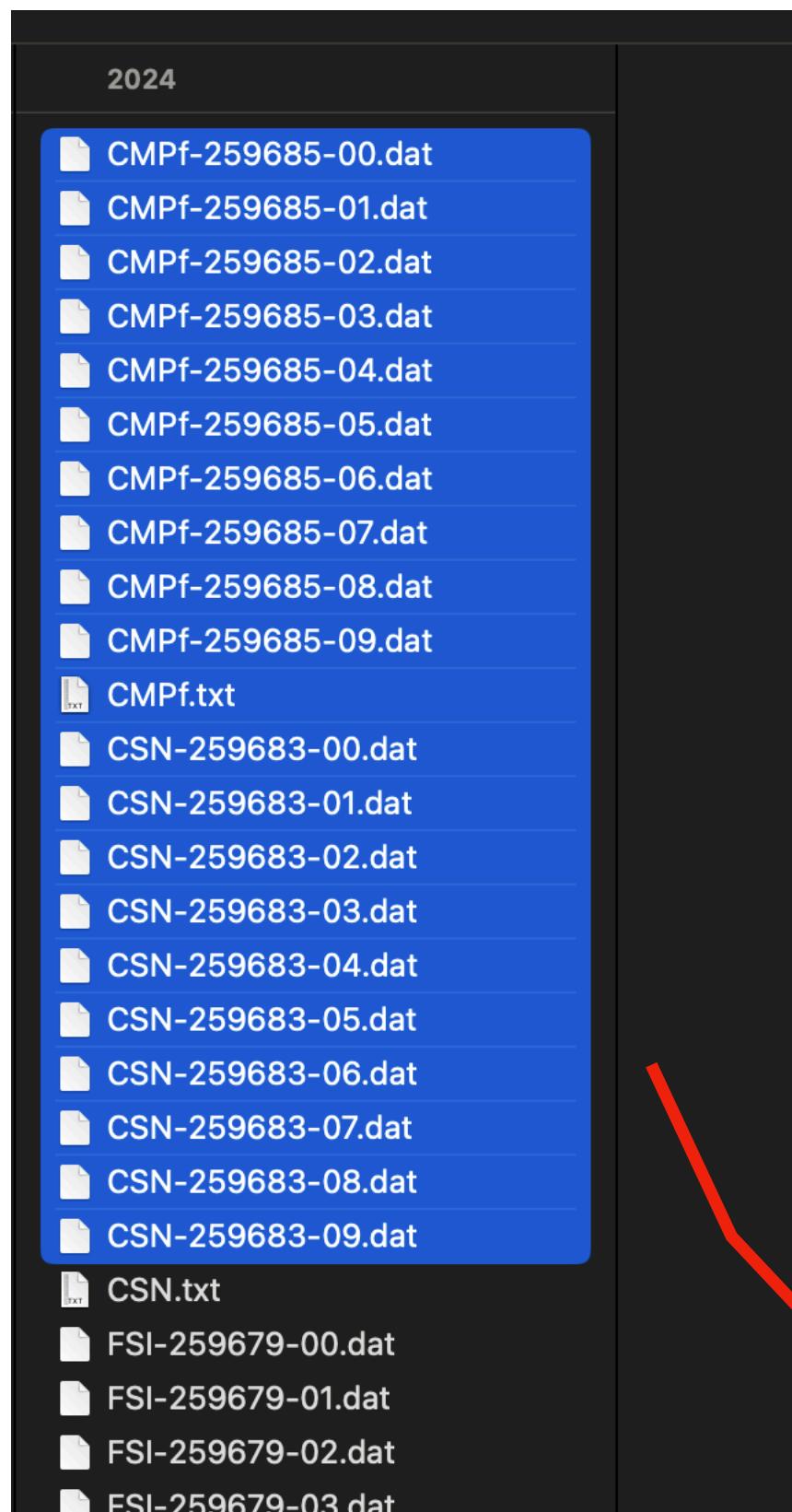
Data in SONATA



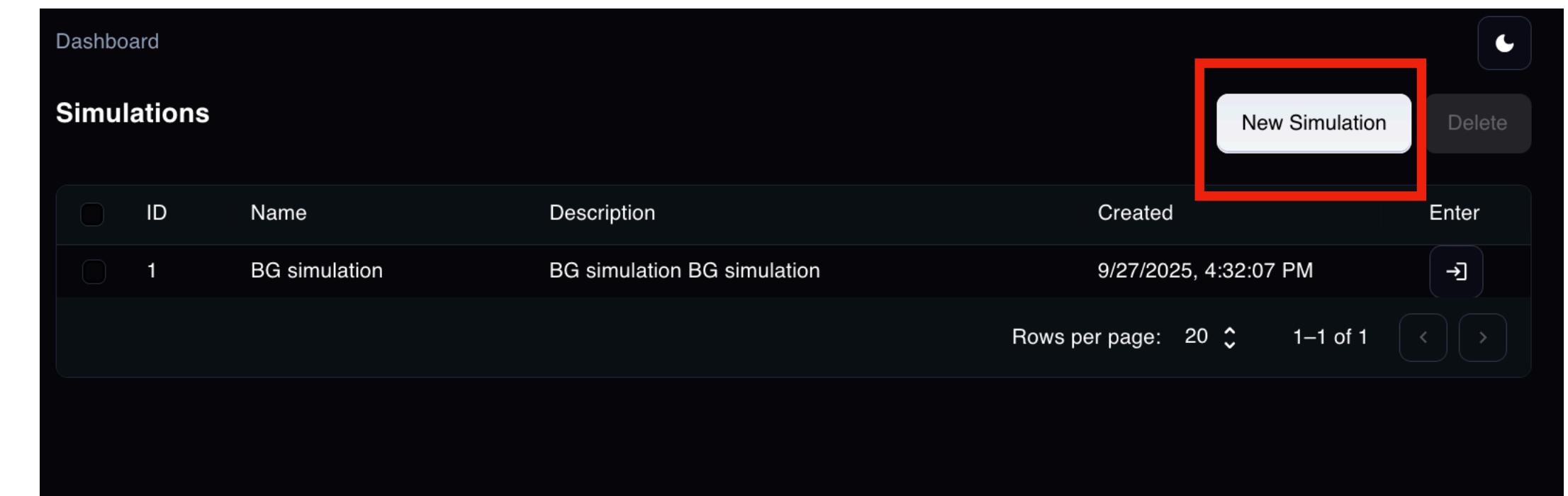
Dashboard

Examples files (NEST simulator output spiking data) here:
(please use only the .DAT files)

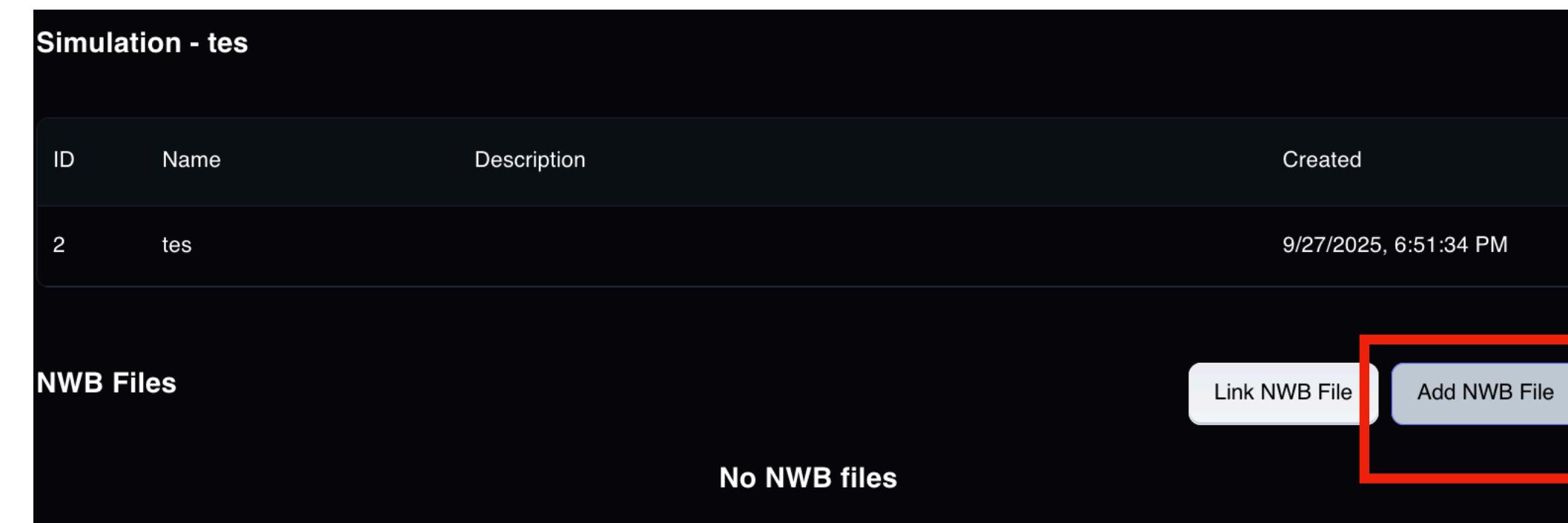
[https://www.dropbox.com/scl/fo/k5psx0lgsw6dhjr2tk2hz/
ADXp6PWMP9Fde7BbJvtKhn0?
rlkey=13b0zy7y1oyerb6r75e37exbs&st=bx157kd6&dl=0](https://www.dropbox.com/scl/fo/k5psx0lgsw6dhjr2tk2hz/ADXp6PWMP9Fde7BbJvtKhn0?rlkey=13b0zy7y1oyerb6r75e37exbs&st=bx157kd6&dl=0)



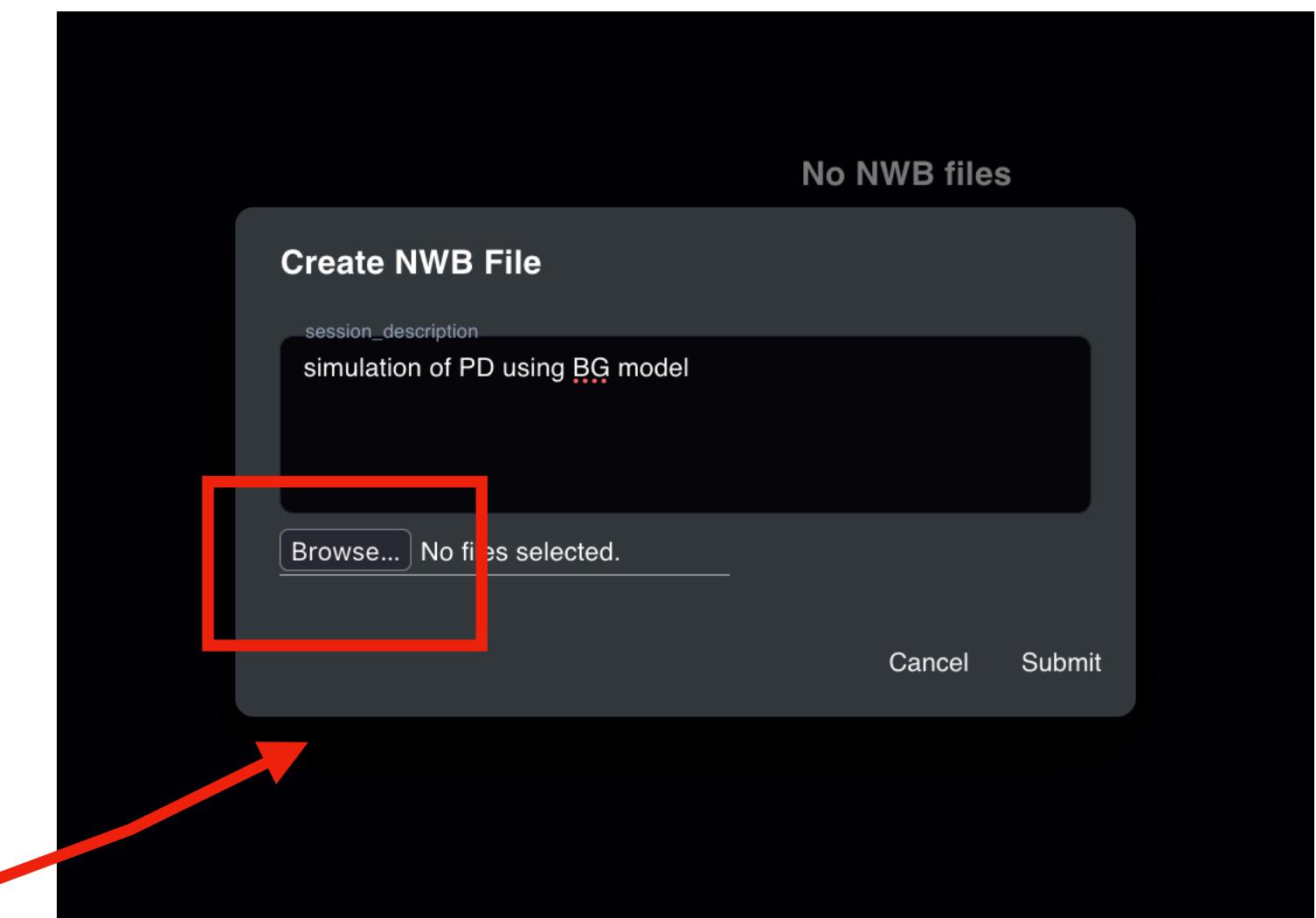
1 - Create a simulation record



2 - Select it and Add NWB file



3 - upload dat files (NEST)



(Automatic conversion to NWB)

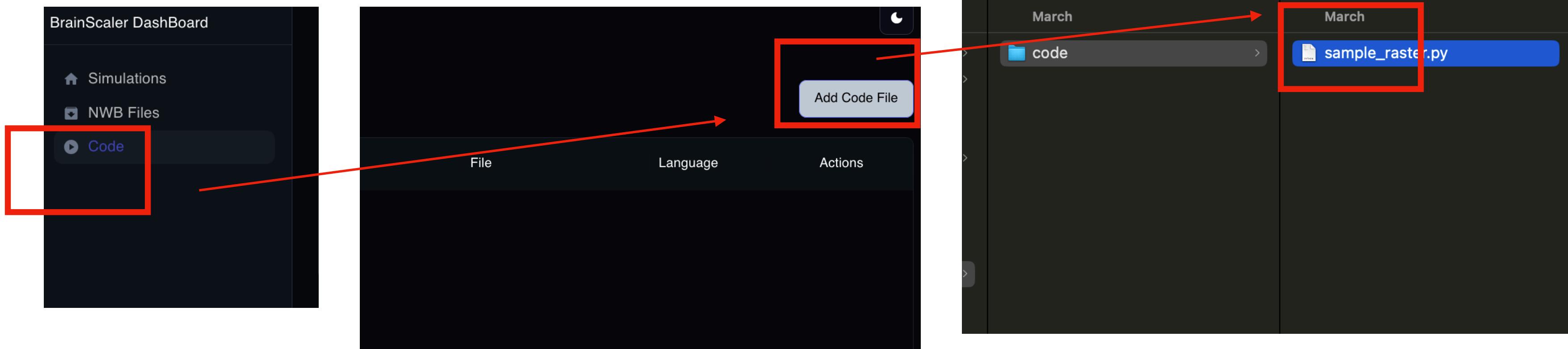
Operation manual—> https://github.com/oist/brainscaler_dashboard/blob/main/doc/operation_manual.pdf

Dashboard

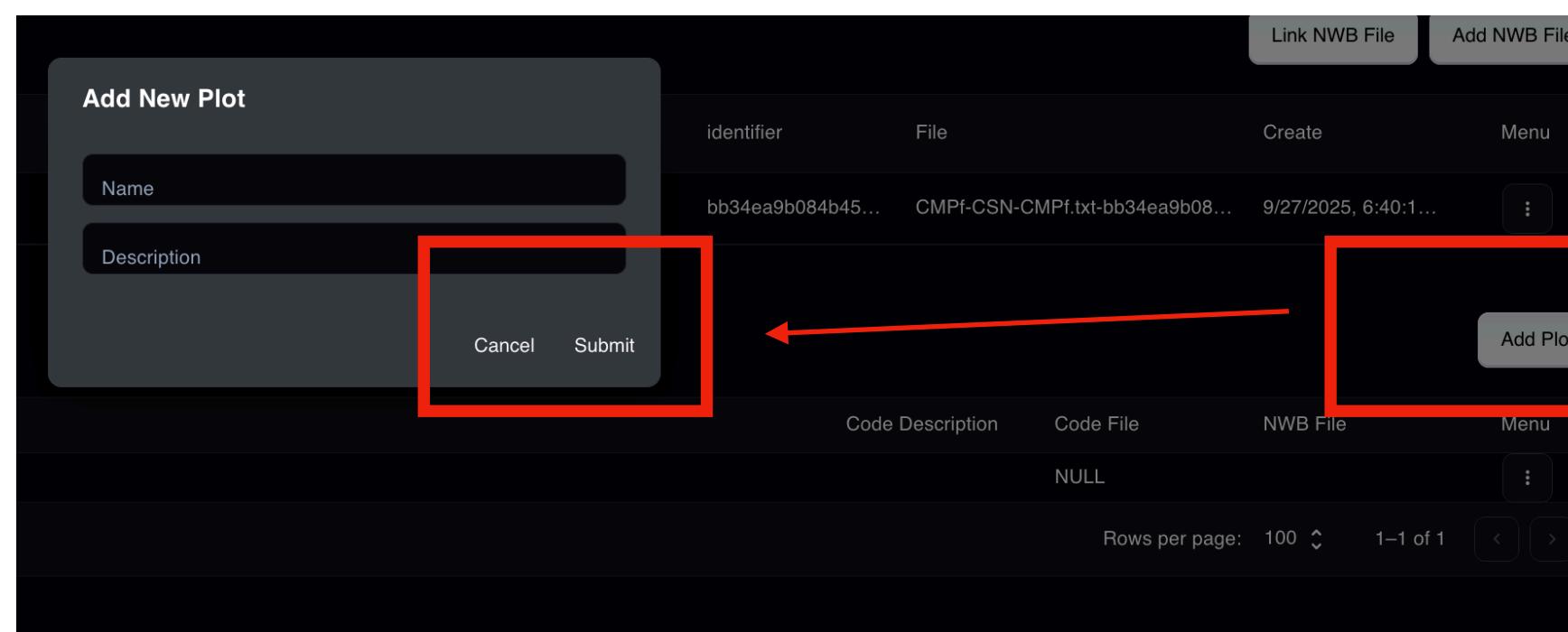
Sample plot code (lets develop more plots !!)

https://github.com/oist/brainscaler_dashboard/blob/main/sample/code/sample_raster.py

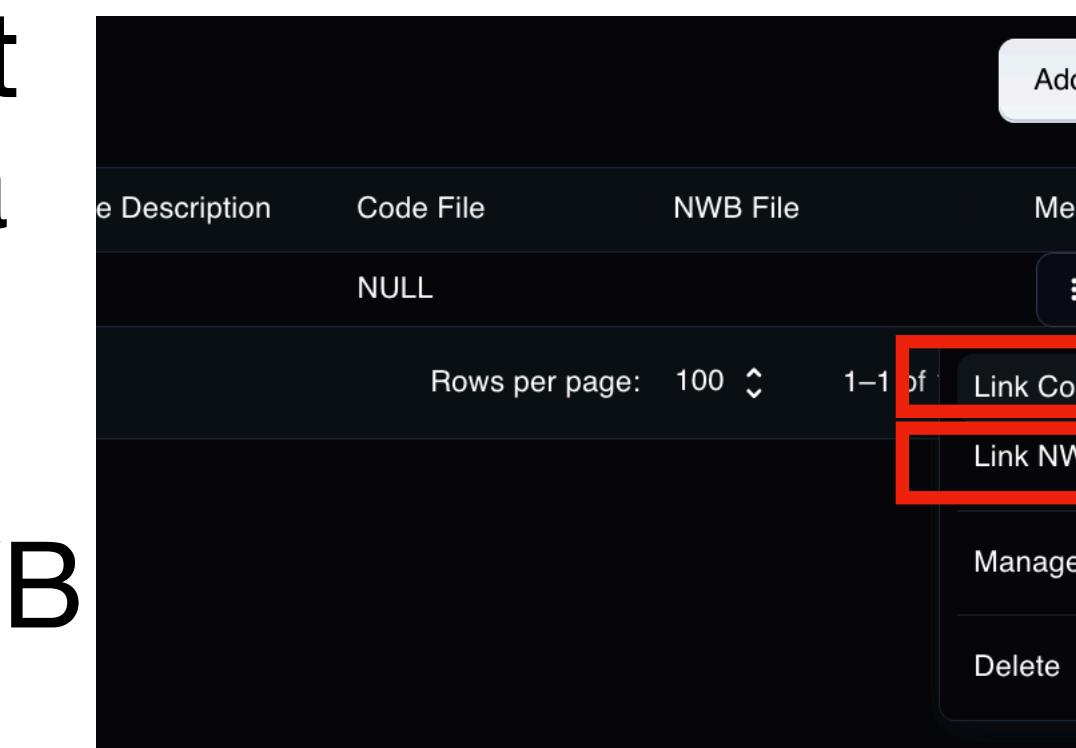
4 - go to code menu to upload a python code of a plot



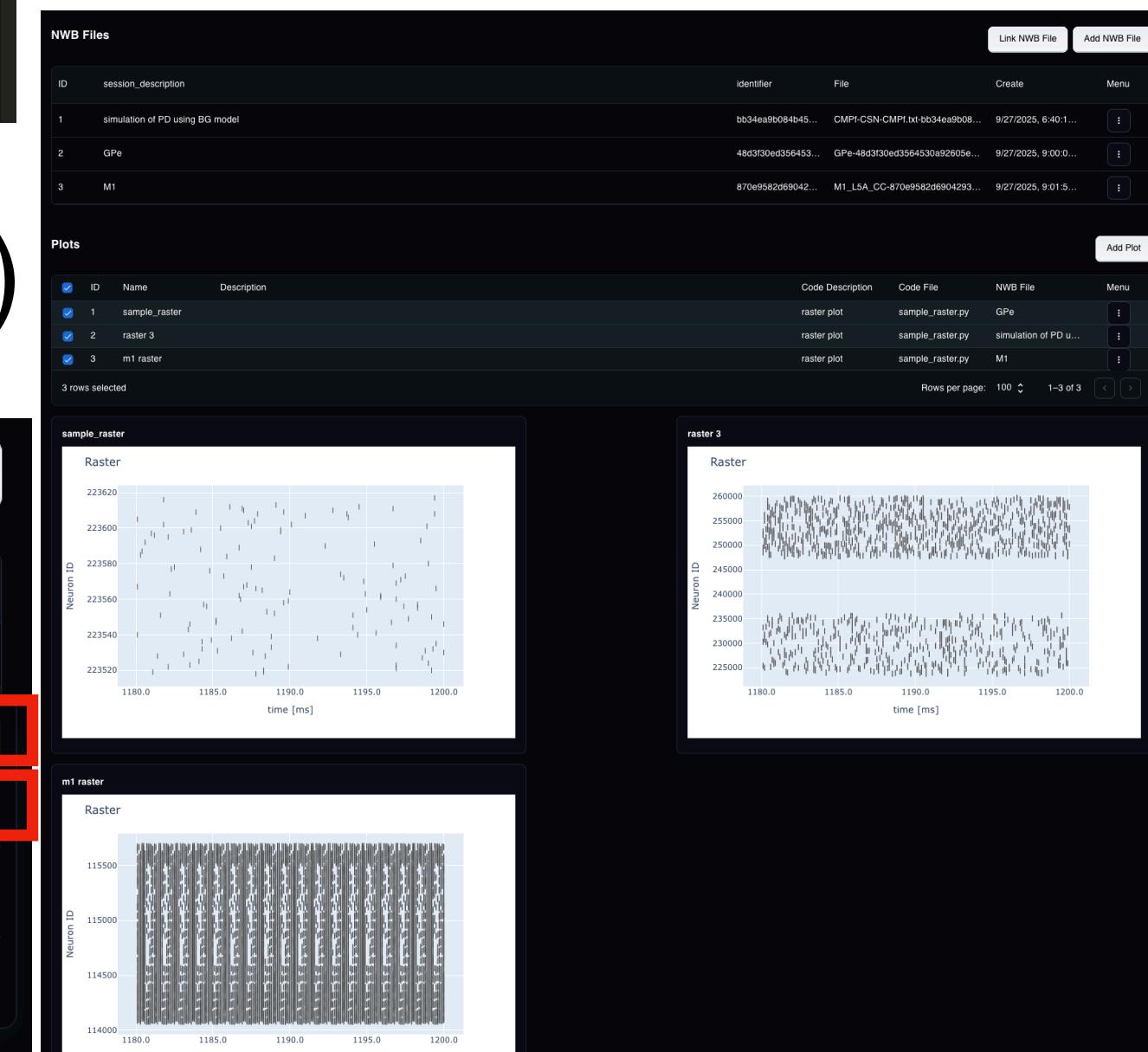
5 - In the simulation record, add a plot (input any description)



6- link that plot with a code, and then with a NWB



7 - you can have many NWB files and plots per simulation (efficient RE-USE of python code for plots)



Dashboard (makes uses of plotly, in theory we can make any interactive plot)



Challenge: Sample plot code (lets develop more plots and upload them !!)

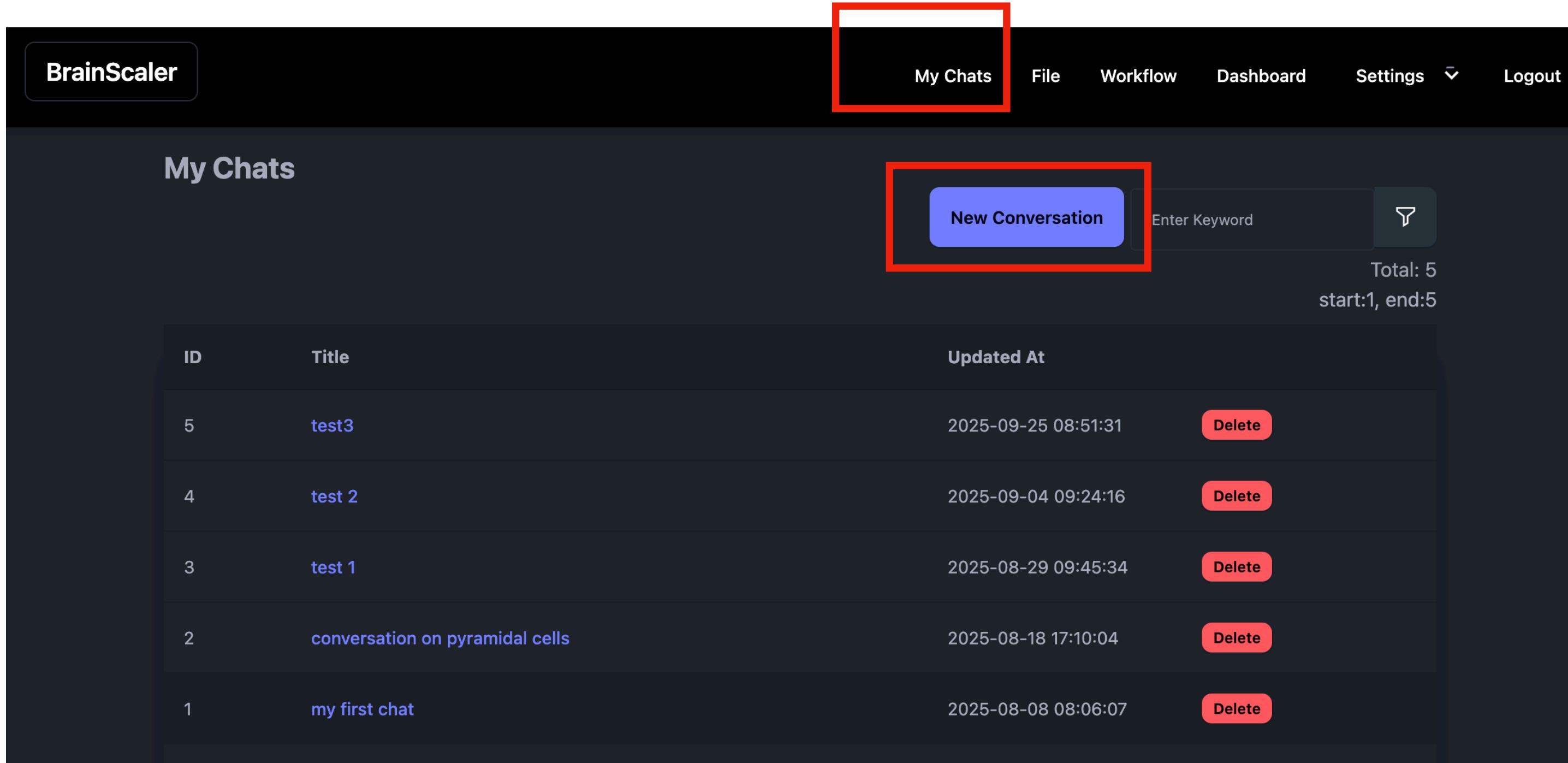
https://github.com/oist/brainscaler_dashboard/blob/main/sample/code/sample_raster.py

Prototypes - Proof-of-Concept

Chatbot

After installation, the Knowledge graph is small by default (only 1 paper, we will update it during hackathon)

1 - Go to My chats and create a new conversation



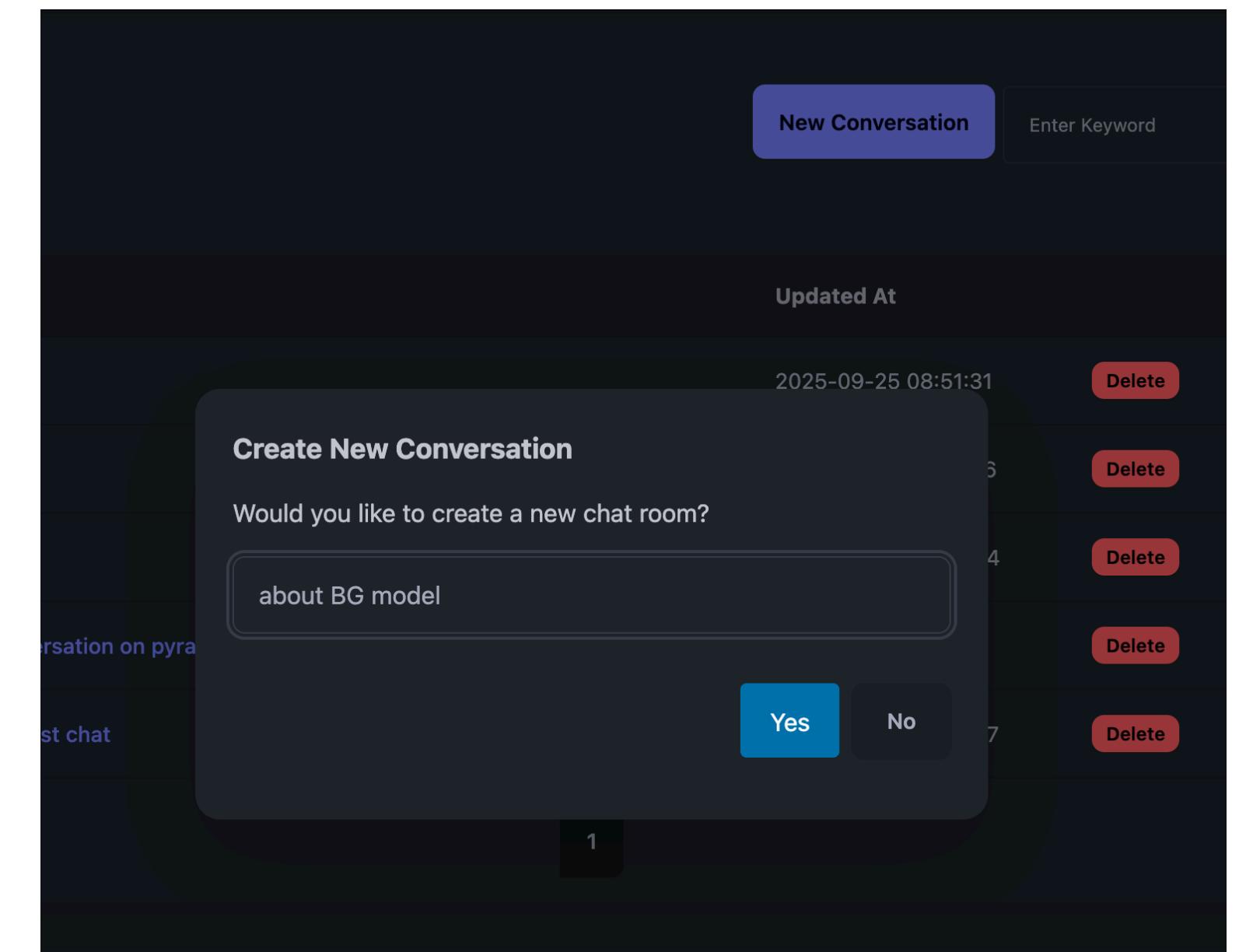
BrainScaler

My Chats

New Conversation

ID	Title	Updated At
5	test3	2025-09-25 08:51:31
4	test 2	2025-09-04 09:24:16
3	test 1	2025-08-29 09:45:34
2	conversation on pyramidal cells	2025-08-18 17:10:04
1	my first chat	2025-08-08 08:06:07

2 - Add conversation theme / name



New Conversation

Enter Keyword

Updated At

2025-09-25 08:51:31

Delete

Create New Conversation

Would you like to create a new chat room?

about BG model

Yes

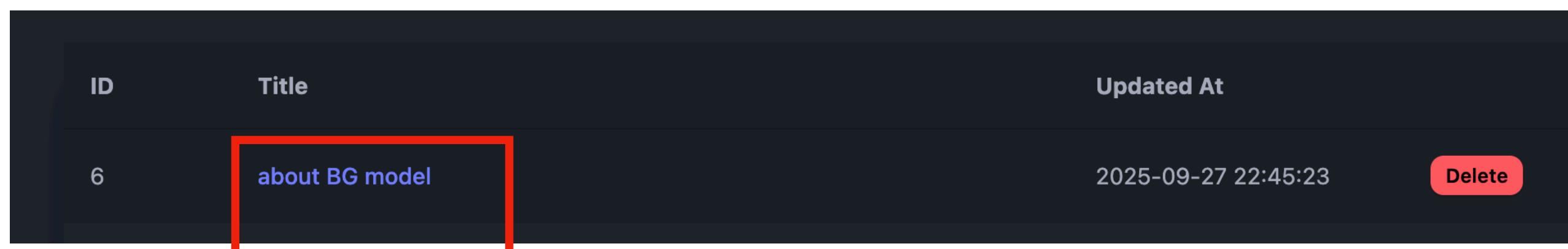
No

Prototypes - Proof-of-Concept

Chatbot

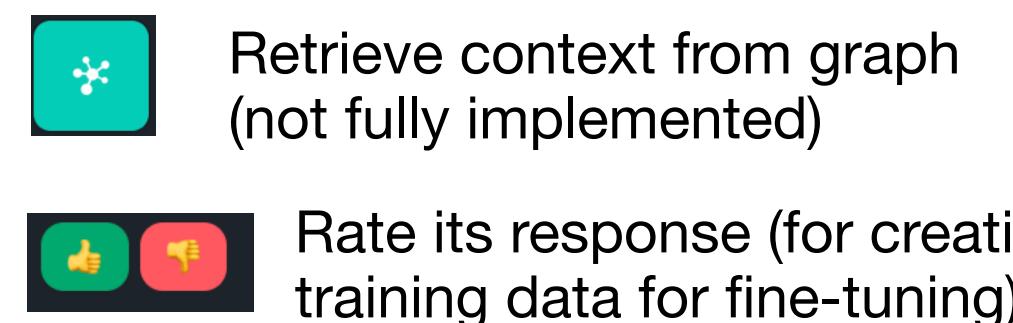
After installation, the Knowledge graph is small by default (only 1 paper, we will update it during hackathon)

3 - Select the conversation

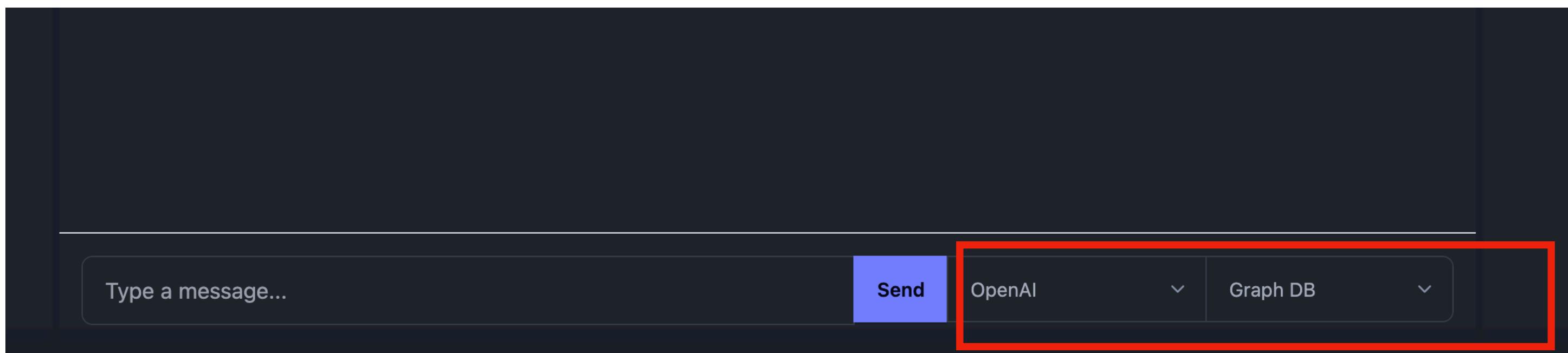


ID	Title	Updated At	Actions
6	about BG model	2025-09-27 22:45:23	<button>Delete</button>

5 - Responses add 3 icons



4 - Select [OpenAI](#) (there are bugs for other LLMs). Choose [Graph DB](#) for hybrid RAG (KG query + vector search) or choose [Vector DB](#) for conventional RAG (vector search)

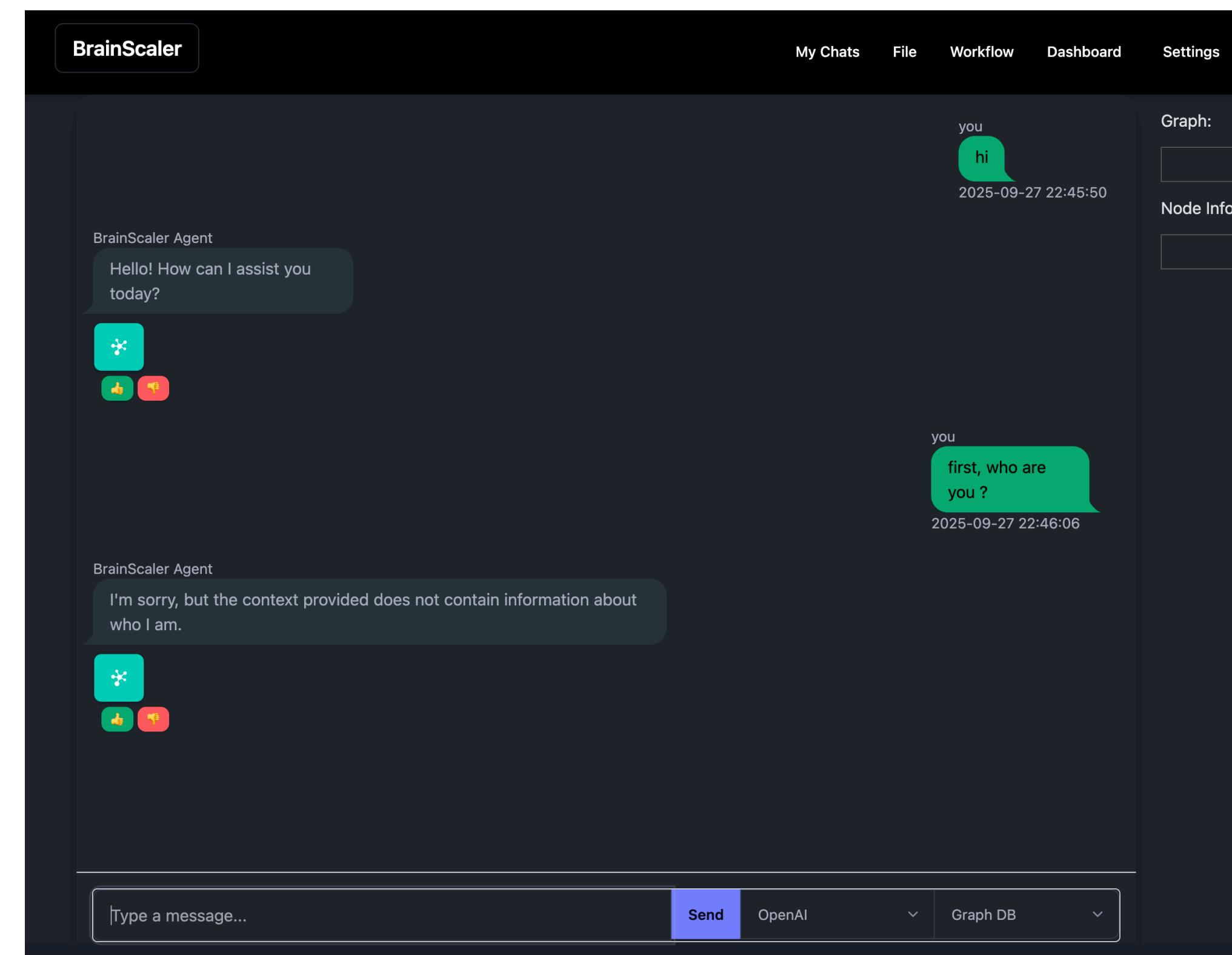


Type a message...

Send

OpenAI

Graph DB



BrainScaler

My Chats File Workflow Dashboard Settings

Graph:

Node Info:

you hi 2025-09-27 22:45:50

you first, who are you ? 2025-09-27 22:46:06

BrainScaler Agent

Hello! How can I assist you today?

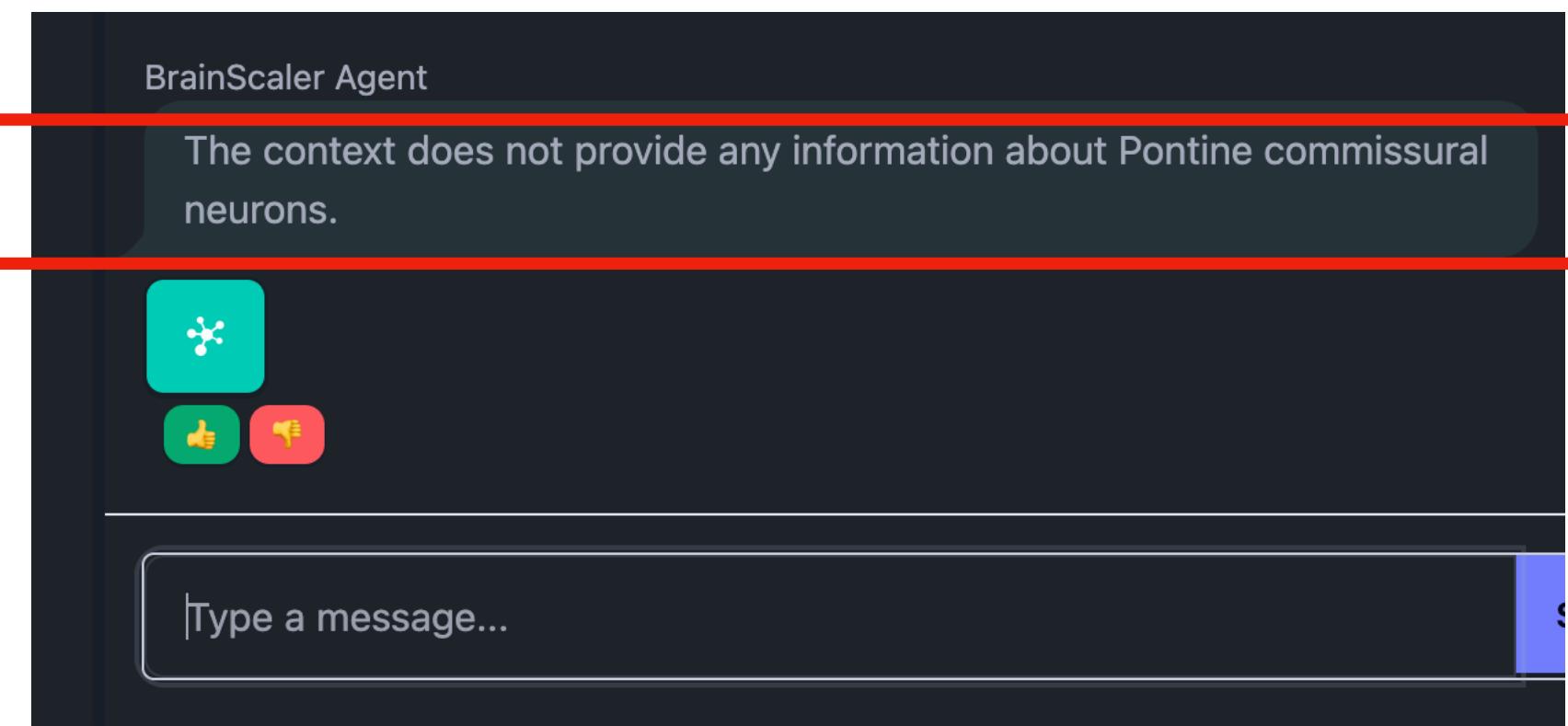
BrainScaler Agent

I'm sorry, but the context provided does not contain information about who I am.

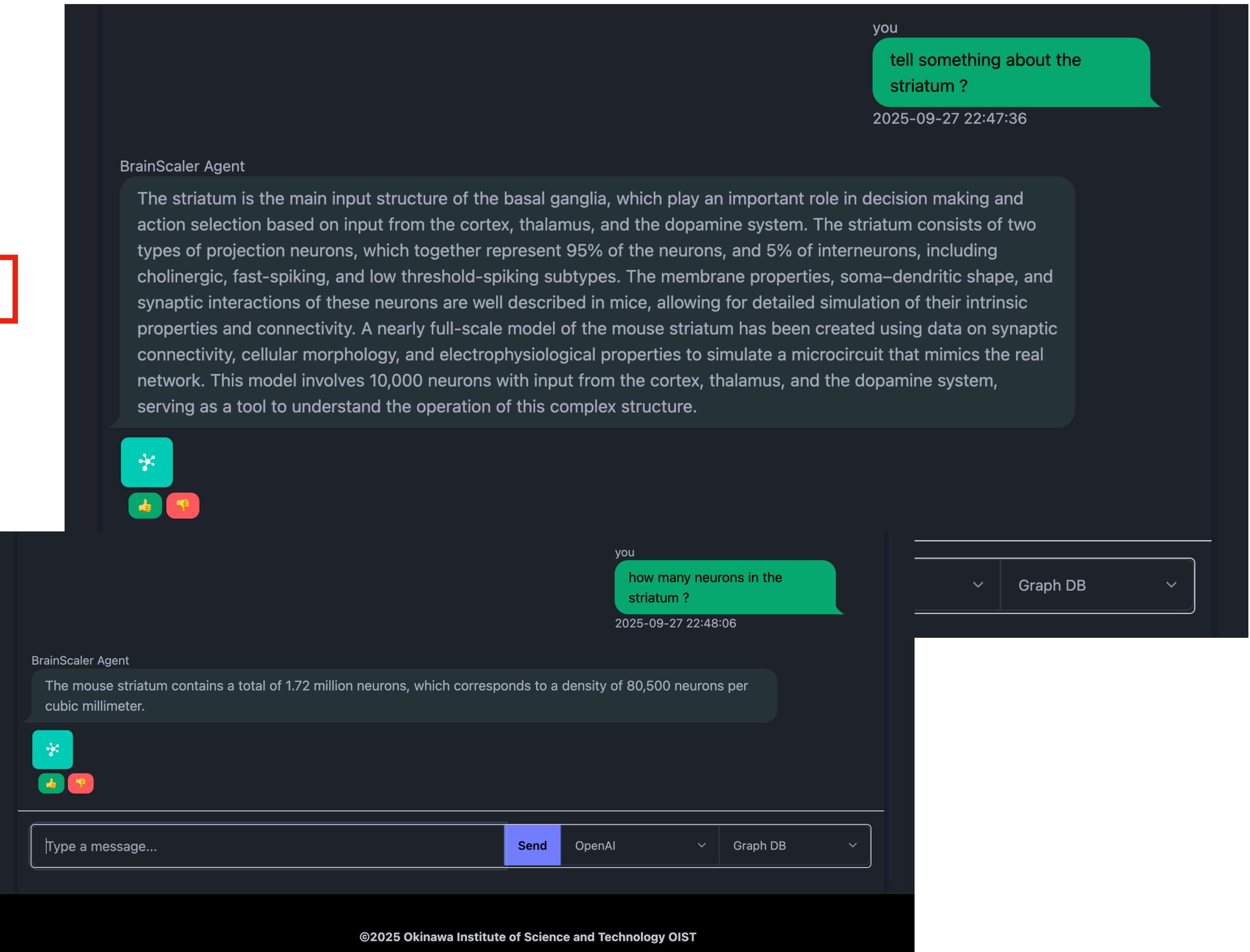
Prototypes - Proof-of-Concept

Chatbot

If context is not found in the graph db
(data related to your query is not found)



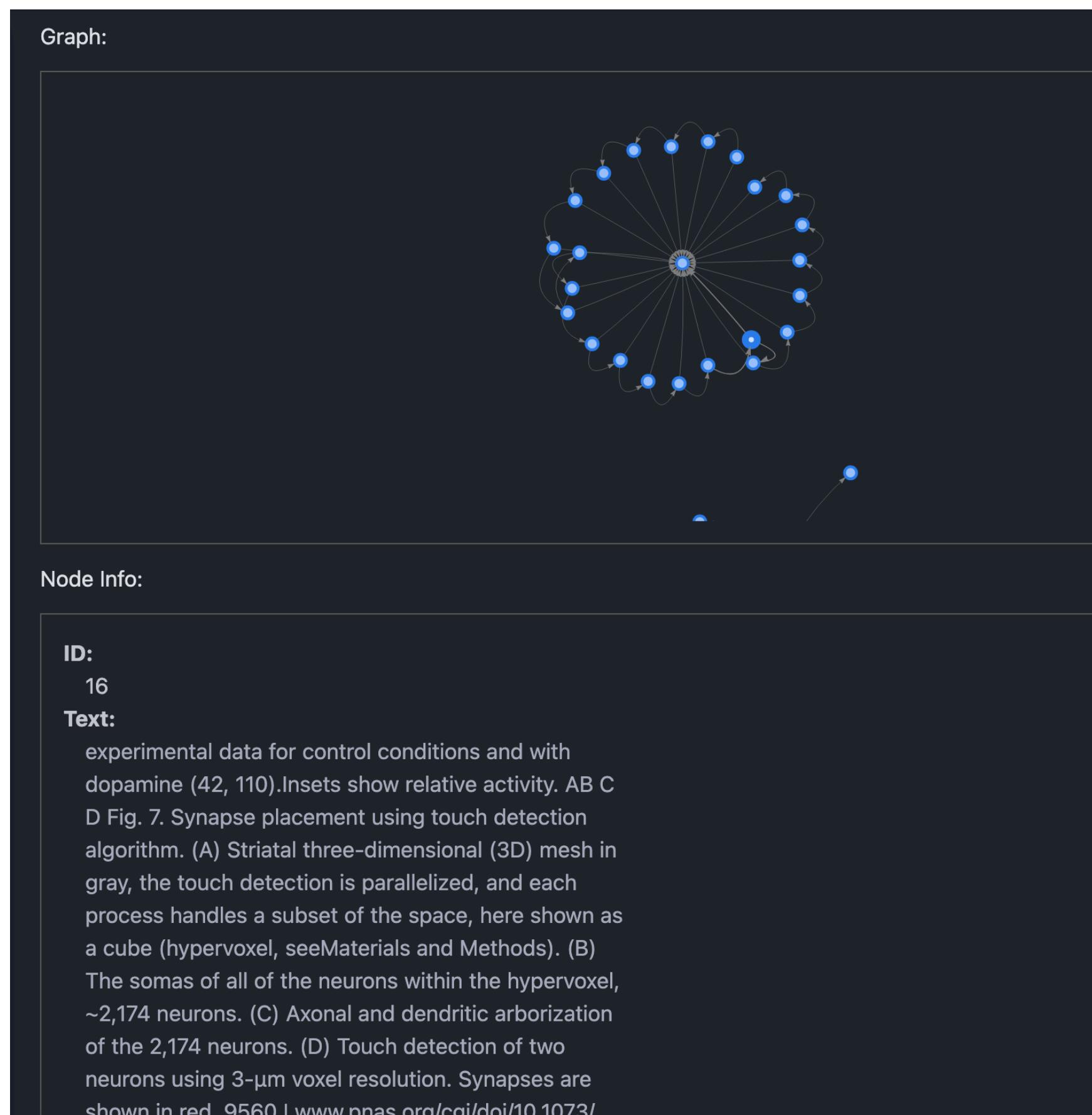
If context exists:



Prototypes - Proof-of-Concept

Chatbot

- Retrieve context from graph (not fully implemented)



Change the character of your chatbot: config.json contains the system and template prompts (changeable)

```
brainscaler_frontend — vim config.json — 165x51

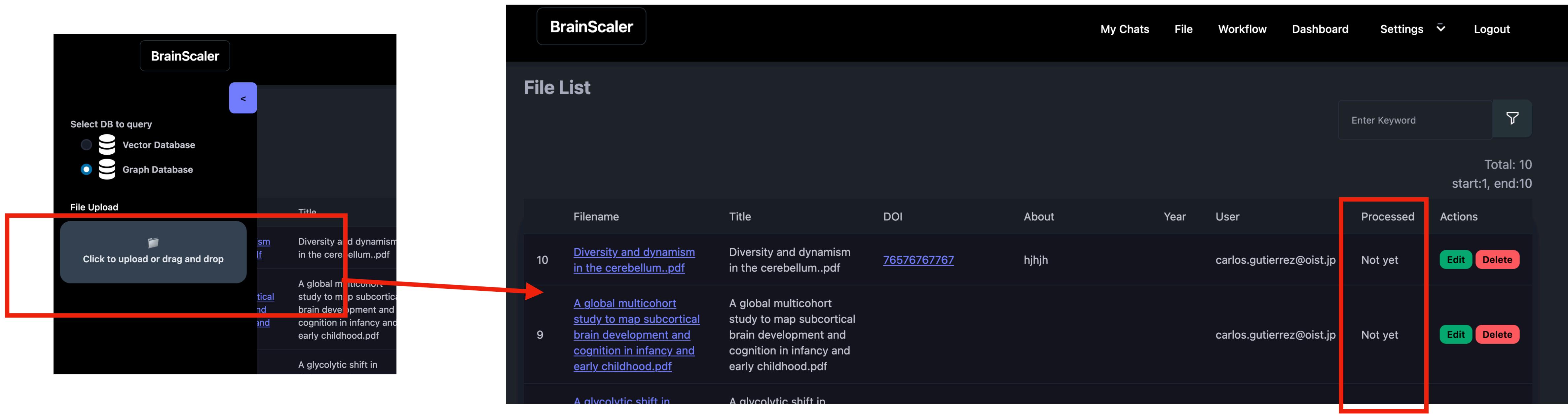
{
  "chat": {
    "system_prompt": "You are a helpful and knowledgeable assistant.",
    "select_provider": "anthropic",
    "anthropic": {
      "model": "claude-3-7-sonnet-latest",
      "temperature": 0.0,
      "max_tokens": null
    },
    "openai": {
      "model": "gpt-4o",
      "temperature": 0.0,
      "max_tokens": null
    },
    "ollama": {
      "model": "gemma3:1b",
      "temperature": 0.0,
      "host": "http://host.docker.internal:11434"
    },
    "rag": {
      "top_k": 5,
      "template": "Answer the Question using the following Context. Only respond with information relative to the context. Do not make up information not mentioned. In the question you may have key words and words that support the sentence. Be flexible and answer in full sentences. Use proper punctuation and spelling. Use lowercase unless instructed otherwise. \n\n# Question:\n{query_text}\n# Context:\n{context}\n# Answer:\n",
      "retrieval_query": "WITH node AS chunk\\nMATCH (chunk)<-[:FROM_CHUNK]-(rel)-[:TO_CHUNK]->chunk\\n AS chunks, \\n  collect(DISTINCT rel) AS rels\\n\\nRETURN '--- text ===\\n' + apoc.text.join([c in chunks | c.text], apoc.text.join([r in rels | startNode(r).name + ' - ' + type(r) + '(' + coalesce(r.details, '') + ')' + ' -> ' + endNode(r).name], '---') + ' ---')\\n\\n"
      "vector_index_name": "text_embeddings"
    }
  }
}
```

Chatbot

Prototypes - Proof-of-Concept

The DBs are built from papers

(It is possible to propose and [upload papers](#), however the processing and detection of entities and connection is done by a [centralized batch process](#))



The image shows the BrainScaler interface in two stages. On the left, a 'File Upload' dialog is open, showing a list of PDF files: 'Diversity and dynamism in the cerebellum.pdf', 'A global multicohort study to map subcortical brain development and cognition in infancy and early childhood.pdf', and 'A glycolytic shift in'. A red box highlights the 'Click to upload or drag and drop' button. On the right, a 'File List' table shows the uploaded files. The table has columns: Filename, Title, DOI, About, Year, User, Processed, and Actions. The first file is 'Diversity and dynamism in the cerebellum.pdf' with DOI 76576767767, user 'carlos.gutierrez@oist.jp', and 'Processed' status 'Not yet'. The second file is 'A global multicohort study to map subcortical brain development and cognition in infancy and early childhood.pdf' with user 'carlos.gutierrez@oist.jp' and 'Processed' status 'Not yet'. A red box highlights the 'Processed' column for both rows.

Filename	Title	DOI	About	Year	User	Processed	Actions
10 Diversity and dynamism in the cerebellum.pdf	Diversity and dynamism in the cerebellum.pdf	76576767767	hjhjh		carlos.gutierrez@oist.jp	Not yet	Edit Delete
9 A global multicohort study to map subcortical brain development and cognition in infancy and early childhood.pdf	A global multicohort study to map subcortical brain development and cognition in infancy and early childhood.pdf				carlos.gutierrez@oist.jp	Not yet	Edit Delete

We are making available a OPENAI temporal api key for testing (.env file)



```
chatbot_dot_env
OPENAI_API_KEY="sk-proj-jLWTP6MxwbL0PlIiRG-Se7fK7cT4X0eTQYbnw4r9LaZpy05_mueJ2CkF_D5WzfXJZuACDJL-gIT3BlbkJrYrFhR8DEFq2v10eiRh8w1cvLeBq5lveKqW_1HYpvvpki6VZz1GhtVxTHPDBUhjY_m0WYVKA"
OPENAI_TEMPOAL_API_KEY="sk-00000000000000000000000000000000"

SUPABASE_URL=https://rexhbnagqwoygewdiaom.supabase.co
SUPABASE_KEY=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3Mi0iJzdXBhYmFzZSIsInJlZiI6InJleGhbmFncXdveWd0jE3NDQxNTU4NjAsImV4cCI6MjA10TczMTg2MH0.C1E80AApbCDqb_cDMpD9-pND_U70yeib5ST_vPbKa88

NE04J_URL=neo4j://localhost:7687
NE04J_USER=neo4j
NE04J_PASS=paolito1677

LOCALDB_DBNAME=postgres
LOCALDB_USER=postgres
LOCALDB_PASSWORD=pass
LOCALDB_HOST=postgres_db
LOCALDB_PORT=5432

CONFIG_FILE=../config.json
```

Thanks, C.