

GEDA: Generative Excel Data Assistant

A Function Calling Excel Agent with Language Models

Welcome to the second half of the class and to the Challenge Generative Excel Data Assistant (GEDA). This document specifies the challenge. It counts to 50% of the course grade. There will be a final written exam contributing to the other half of the grade.

The challenge is kindly contributed by METTLER TOLEDO (MT). The winner of the challenge will receive a prize from MT¹. While the provided data is purely fictitious and is not related to any METTLER TOLEDO business activities, getting insights on data contained in excel files through natural language interaction is an interesting use case for MT and companies in general.

Corporate overview of METTLER TOLEDO

METTLER TOLEDO is a global manufacturer and marketer of precision instruments for use in laboratory, industrial and food retailing applications. The Company has strong worldwide leadership positions. A significant majority of our instrument sales are in segments in which we are the global leader. In addition to a broad product offering, we have one of the largest global sales and service organizations among precision instrument companies.

METTLER TOLEDO

Empowering a World of Opportunities™

We focus on the high value-added segments of our markets by providing innovative instruments that often integrate various technologies including application-specific solutions for customers. We design our instruments not only to gather valuable data, but also to facilitate the processing, analysis, and transfer of this data into customers' management information systems.

More information: [METTLER TOLEDO Balances & Scales for Industry, Lab, Retail - METTLER TOLEDO](#)

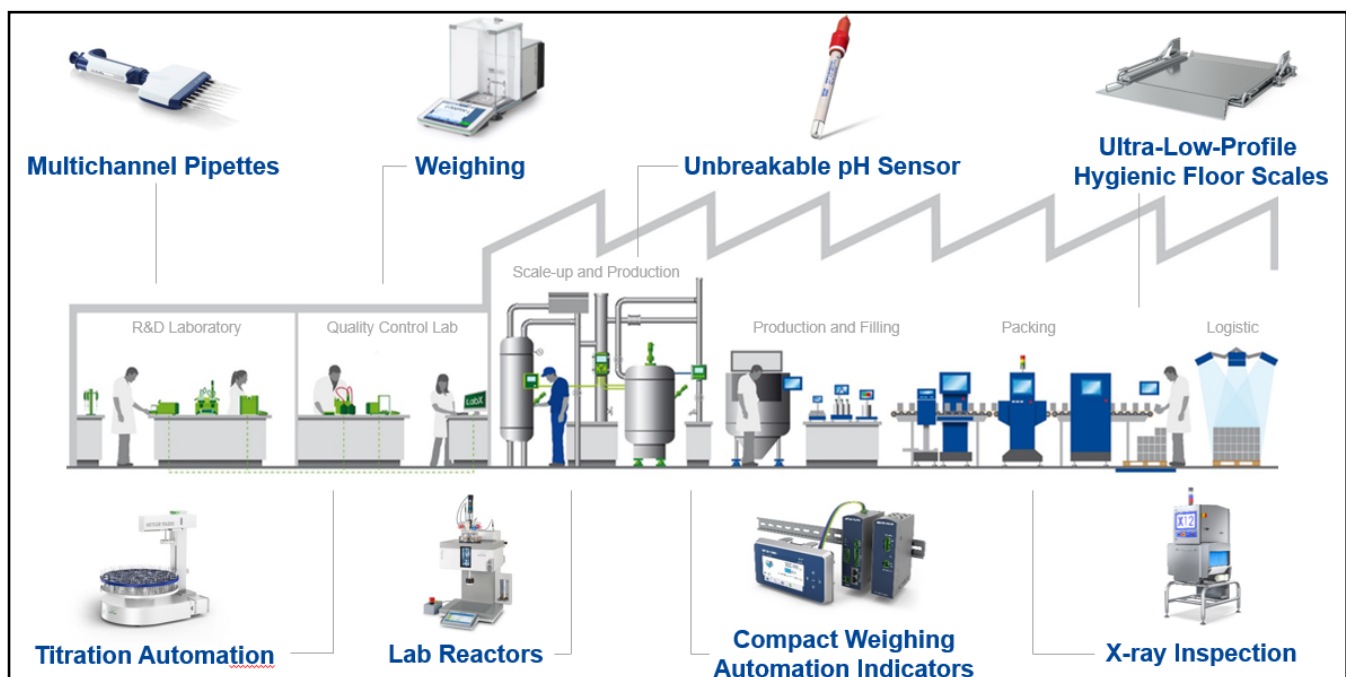


Figure 1: Recent Innovation Examples of METTLER TOLEDO instruments.

¹ To be eligible for the prize, groups must make their GEDA source available to Mettler Toledo. If you choose to opt out, you will receive a fair grade for your work but won't be considered for the prize.

1. Important Dates:

Start: As soon as the challenge is uploaded on the moodle
Live Evaluation: **20th December 2024 during the class**
Report submission: **Monday, January 6th 2025 (upload your report before 23:59)**

2. Administrative rules of the challenge

- This is a solution-oriented project. Section 3 provides a loose description of the problem. The description gives you a skeleton; how you bring it to life is up to your creativity, interests, and skills.
- You are asked to implement a solution in a team. If you have problems, we encourage you to discuss it on the moodle forum to get help from your colleagues. ChatGPT can also be used for brain storming. We are also available for help.
- Work in a team of two to three. Larger teams are strictly not allowed.
- Team members will get the same grade. Note that fraud will not be taken lightly and may have severe consequences. In case of doubt, we reserve the right to interview the suspects. Such interviews are unfortunate, and certainly a nightmare on our side.
- Please submit the report on time. Else, a grade 1.0 will be awarded. If you are unable to meet the deadline for an important reason (illness, military service, etc.) the deadline can be discussed. Contact us as early as possible!
- There will be an assignment called "GEDA" on Moodle (in the week 7 section). **All team members will have to upload the (same) PDF File.** Do not upload the code.

3. Technical details of the challenge

Excel isn't a traditional programming language, but it does have some characteristics of one. It provides powerful functions, formulas, and even logic through conditional statements, allowing complex calculations, manipulations, and automation for storing, analyzing, and visualizing data. Python is particularly strong in scientific computing and is a natural fit in combination with Excel. This, in turn implies it is easy to turn the Python-powered Excel prototype into a full-fledged web application that too with natural language support.

In this Generative AI challenge, you are asked to build GEDA, an intelligent Python-based Excel assistant that uses open-source language models to interact with Excel file(s). The user uploads one or more Excel files to GEDA. The user then communicates with GEDA via natural language queries, aka, prompts. GEDA understands the intent of the user and carries out the intended tasks e.g., querying data, manipulating data, complex data analysis and data visualization etc.

METTLER TOLEDO and OST will provide you the following inputs:

- a) Definition of MVP (Section 5)
- b) Sample Excel files
 - You'll receive a set of Excel files containing data like sales, costs, and material data etc.
Important notice: The data therein is artificially generated for this challenge and is purely fictitious, any similarities are therefore purely coincidental. The data and content are **not** related to any METTLER TOLEDO products, services, customers, or actual sales data, and it does not reflect any business activities at METTLER TOLEDO. Therefore, these are **not** confidential / proprietary datasets, and you are free to use these as your GEDA demands.
- c) Sample prompts

- To give an idea of how GEDA accepts inputs from the user, you are given a set of natural language prompts/instructions (Section 9).
- GEDA will interpret the prompts (e.g., "Compare sales trends for Q1 2023 with Q1 2022" or "Plot material costs in EUR") and produce the required results, e.g., the content of the queried cell, a plot of material costs or changing the value of a particular cell in the Excel etc.

d) Development environment:

The challenge offers you the following options for implementing and deploy GEDA.

- LLMHub: An open source LLM server hosted at OST. The list of deployed LLMs can be found here: <https://llmhub.infs.ch/models/>. We can deploy new LLMs on your request.
- Microsoft Azure: You also have access to the Azure ML <https://ml.azure.com/> on which some LLMs can be accessed. Please do not spin a VM making the budget go quickly out of hand. If you forget to shut down the VM, the cost will accumulate. Hence, do not spawn a VM.
- Jupyterhub: You may also use the Jupyterhub available at <https://gpu.jupyter.ost.ch>. The Jupyterhub offers you a workspace along with one A4000 16GB GPU. It is also possible to combine the Jupyterhub with LLMHub. As a possibility, GEDA can run on Jupyterhub and can send requests to the LLMHub.

4. Dos and Don'ts of GEDA

a) What GEDA is not:

- GEDA is not a general python code generator as there is not so much control over the generated code. The generated code is meant to be executed which may have unsafe side effects or outcomes.
- For the same reason, GEDA does not create Excel macros.
- GEDA is definitely not an Excel addin or a plug-in.
- There is also xlwings to build interactive applications where Excel spreadsheets act as the user interface, allowing you to call Python by clicking a button or calling a user-defined function. Again, addins are not welcome. xlwings can be used for direct scripting and execution in a python environment without need for the add-in.

b) What GEDA must:

- be external to Excel, i.e., manipulates Excel from outside.
- have a basic understanding of the Excel files.
- read the sheets, the cells, manipulates the cells, analyses and changes columns, and produces charts.
- Be a function calling/ tool calling agent – GEDA does not produce and execute arbitrary Python code. Executable functions must be whitelisted specifically.

c) What GEDA can:

- be seen as a standalone app that reads Excel File(s).
- be a web application as well.
- only call the chosen functions/tools of the supported framework (e.g., Excel, OpenPyXL, or XlsxWriter, whatever your choice be).

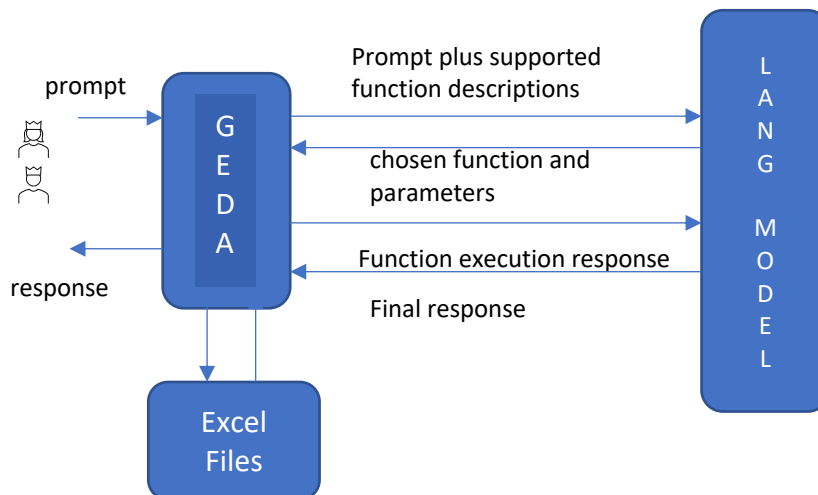
d) What GEDA is not allowed:

- Last but not the least, if you plan to use azure, GEDA must not spin a VM making the budget go quickly out of hand. If you forget to shut down the VM, the cost will accumulate. Hence, do not spawn a VM.

5. Minimum Viable Product

To qualify for a passing grade (4.0), your GEDA must:

1. **File Handling:**
 - 1.1. allow the user to upload or add Excel file(s)
 - 1.2. be able to work with at least one Excel file
2. **Natural language input:** take a user prompt and use a language model to understand the intent
3. **Function Calling:** map the intent to an underlying function call
 - 3.1. Python libraries like Openpyxl, XlsxWriter etc. can be used under the hood for manipulating Excel. You may create higher level Python wrapper functions calling underlying library functions².
 - 3.2. collect the arguments, and present either the function call to the user or execute the function and display the response (The latter is desirable.)
4. **Language model:**
 - 4.1. must support gpt-4omini
 - 4.2. must support an open-source small language model with maximum 8B parameters
5. **Data Analysis and visualization**
 - 5.1. be able to query and manipulate a single cell in an Excel sheet
 - 5.2. plot simple charts like histogram, line plot of certain column
6. **User Interface:**
 - 6.1. have a basic minimal chat-like user interface using Gradio, streamlit, dash, flask/FastAPI etc.



1. User sends an instruction/prompt to GEDA
2. GEDA receives the instruction and sends to the language models along with the supported tools/function definitions.
 1. The supported functions are listed in Section 9.
3. The language model chooses a suitable function
4. The language model returns the chosen function name and arguments to GEDA
5. GEDA can either execute the function or display it to the user
 1. If it executes the function, it returns the response to the model
6. The language model may use the response to determine the final response
7. GEDA presents the final response to the user

Some inspiration can be taken from:

https://docs.mistral.ai/capabilities/function_calling/
<https://gorilla.cs.berkeley.edu/index.html>

² If your GEDA generates Excel functions, display those to the user as a response in the chat window. In this case, the function should work correctly when copy pasted in the Excel file.

6. Deliverables

1. working source code in the GitLab repository. Give us access.
2. You must write a short report (PDF file) describing your agent's architecture, your approach to implementing the function calling ability, the concepts you used, the features you implemented, and whatever you consider interesting. Make a few screenshots that clarify the usage of your client. Document both, the backend, and the client.
 - a) write in English.
 - b) Have a title page with all relevant information (team members, title, ...)
 - c) Use an appropriate diagram to visualize the architecture of your application,
 - d) **Model Choice and Justification:** Evaluate the choice of open-source language models, including a comparison of their performance and how effectively they perform compared to gpt-4omini.
 - e) Structure your report. Write at least four sections:
 - a. **Introduction** where you give some context and explain the goal of the project.
 - b. Documentation of your **GEDA**: What kind of prompts your GEDA supports. Screenshots of the working GEDA and its features
 - c. Some details about the **GEDA backend**
 - d. **Discussion** of results and learnings

7. Live Demo

Present your solution live, with real-time queries and analysis. The best solution, evaluated live with our industry partner METTLER TOLEDO, will be rewarded, based on the following points.

1. **Accuracy:** Correctness of responses, data retrieval, and analysis.
2. **Performance of Language Models:** How well the open-source language model interprets and handles natural language queries compared to the gpt-4o.
3. **Usability:** How well the system interacts with the user, handling ambiguous queries, and providing interactive feedback.

8. Optional features of GEDA

We do not specify a detailed, exhaustive list of optional features, but we mention some desirable features.

- support of multiple Excel files, i.e., instructions encompassing data over multiple files. Automatically locate and process relevant Excel files based on the user query
- support to intelligently identify and fetch the appropriate Excel files based on the user's query.
- support for **multi-step query interactions**, allowing users to ask follow-up questions and refine their queries
- support for external web APIs like price conversion using some external service for currency conversion rates
- Handle complex tasks such as timeseries comparison, generating custom formulas, or performing currency conversion.
- Perform tasks like sales comparisons, and cost analysis etc.

9. Examples of natural language inputs

Some examples of natural language inputs are presented in the table here. This list is not exhaustive and aims to give you an idea of prompts that could be expected from a business user.

Tasks	Examples for natural language query
Finding information and the relevant files	<ul style="list-style-type: none">• Where can I find the cost for wood in the year 2023 and what is it?• What supplier is selling Copper in 2023?• Give me the file names that cover US sales data.
Modify or create new columns in excel file including formulas (from a single file. multiple file support a bonus)	<ul style="list-style-type: none">• For the Sales data of the US in 2022 add a column that indicates the price per unit.• Give me an excel formula with which I can calculate the price per unit in the Sales data for the US in 2022.• To the sales data for Switzerland in 2023, add a column of the supplier for each product. The supplier data is mentioned in another file for each product.
Timeseries data and related questions	<ul style="list-style-type: none">• Can you compare the total sales in Germany for May and June 2023?• Return the total Sales of Switzerland for each month in 2023.
Plotting of timeseries or other kind of data	<ul style="list-style-type: none">• Plot the evolution of sales of Steel for every month in Switzerland for 2023.• Provide a line plot of the cost as a function of time for the materials sold by "Elemental Resources". Only consider data of 2022 and 2023.• Calculate the total Sales of Switzerland for each month in 2023 and plot it as a bar plot. The x-axis should be time in months and the y-axis the monthly sales.
Currency conversion of the sales data (via API call, or other)	<ul style="list-style-type: none">• State the sales of Brass in Switzerland for January 2023, in Swiss Francs (CHF).• Convert the sales column for the Swiss sales in 2023 to CHF and add it as an additional column in the respective file.
Combining information from multiple files	<ul style="list-style-type: none">• What is the revenue for May 2023?• What is the revenue generated with Graphene in Switzerland in 2023?• Compare the total sales of Switzerland and the US in 2023. State the relative difference.

10. Useful Links

- i. Azure Links:
 - a. <https://techcommunity.microsoft.com/t5/azure-infrastructure-blog/get-started-with-azure-ai-services-open-ai-and-deployment-models/ba-p/4138583>
 - b. <https://learn.microsoft.com/en-us/azure/ai-services/openai/concepts/models?tabs=python-secure%2Cglobal-standard%2Cstandard-chat-completions>
 - c. <https://ml.azure.com/model/catalog>
- ii. Jupyter
 - a. <https://gpu.jupyter.ost.ch/hub/spawn>
 - b. <https://github.com/jupyterhub/jupyter-server-proxy>
 - c. <https://jupyter-server-proxy.readthedocs.io/en/latest/>
- iii. LLMHub
 - a. <https://llmhub.infs.ch/>