Fraunhofer FOKUS

Fraunhofer Institute for Open Communication Systems

Large Language Models: Generative Agents

AWT Seminar WS 2023/24

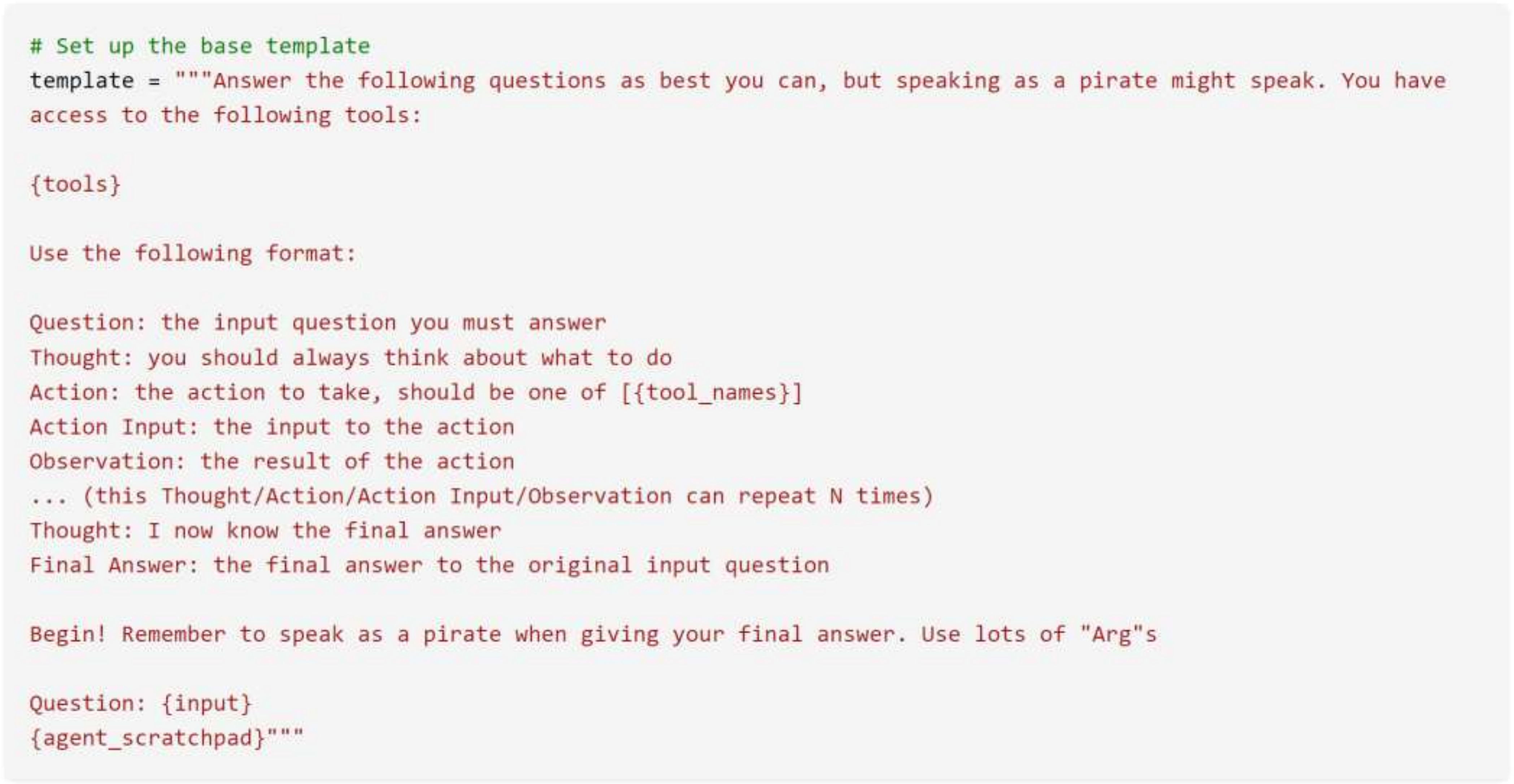
Background: Generative Agents

- ChatGPT popularized Large Language Models (LLMs)
- Since 2023, **a variety of capable OS LLMs** has emerged that **can be deployed locally**
- Utilization of LLMs goes far beyond Chatbots, e.g.,: Generative Agents

- Basic functioning of LLM-based Chatbots:
  1. A user provides a prompt
  2. An answer is generated by an LLM
- Generative Agents provide more possibilities. Here, the general logic is:
  1. An LLM-instance generates a statement
  2. The statement is used within the prompt for another LLM-instance

- In general: Input and output are both strings and can be manipulated as such
- In the end, it is all about generating, manipulating strings and using them as new inputs
- Different roles of agents. Different characters.

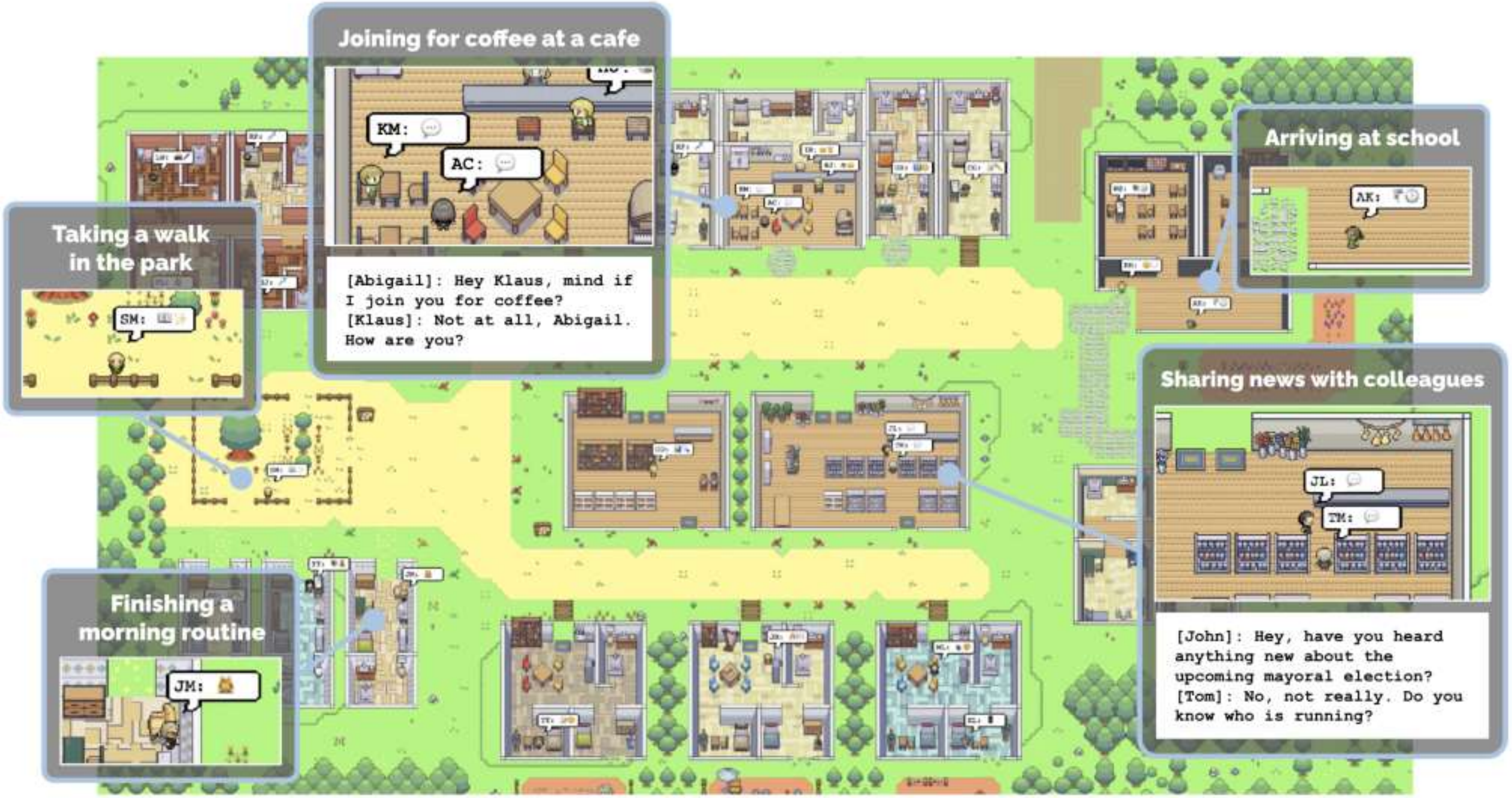
Background: Prompts



```
# Set up the base template
template = """Answer the following questions as best you can, but speaking as a pirate might speak. You have
access to the following tools:

{tools}

Use the following format:

Question: the input question you must answer
Thought: you should always think about what to do
Action: the action to take, should be one of [{tool_names}]
Action Input: the input to the action
Observation: the result of the action
... (this Thought/Action/Action Input/Observation can repeat N times)
Thought: I now know the final answer
Final Answer: the final answer to the original input question

Begin! Remember to speak as a pirate when giving your final answer. Use lots of "Arg"s

Question: {input}
{agent_scratchpad}"""
```

Source: https://python.langchain.com/docs/modules/agents/how_to/custom_llm_agent

Background: Generative Agents


Joining for coffee at a cafe
KM:
AC:
Arriving at school
AK:
Taking a walk in the park
SM:
Sharing news with colleagues
JL:
TM:
Finishing a morning routine
JM:

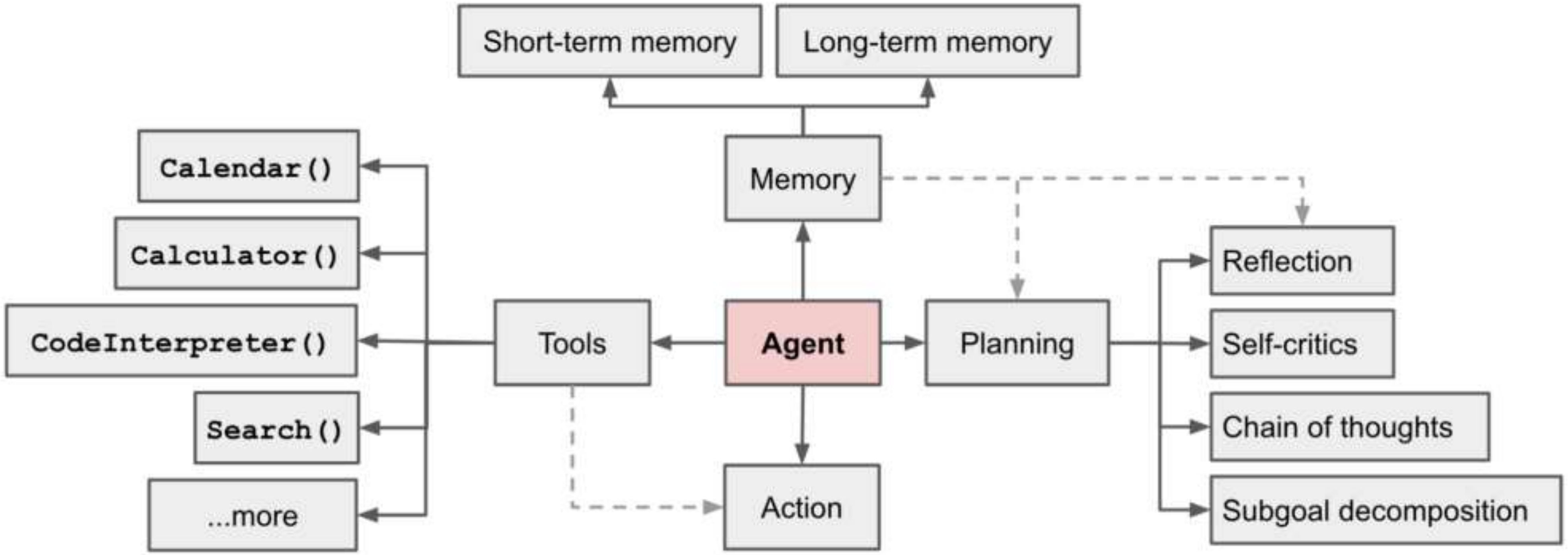
[Abigail]: Hey Klaus, mind if I join you for coffee?
[Klaus]: Not at all, Abigail. How are you?

[John]: Hey, have you heard anything new about the upcoming mayoral election?
[Tom]: No, not really. Do you know who is running?

Source: https://github.com/joonspk-research/generative_agents

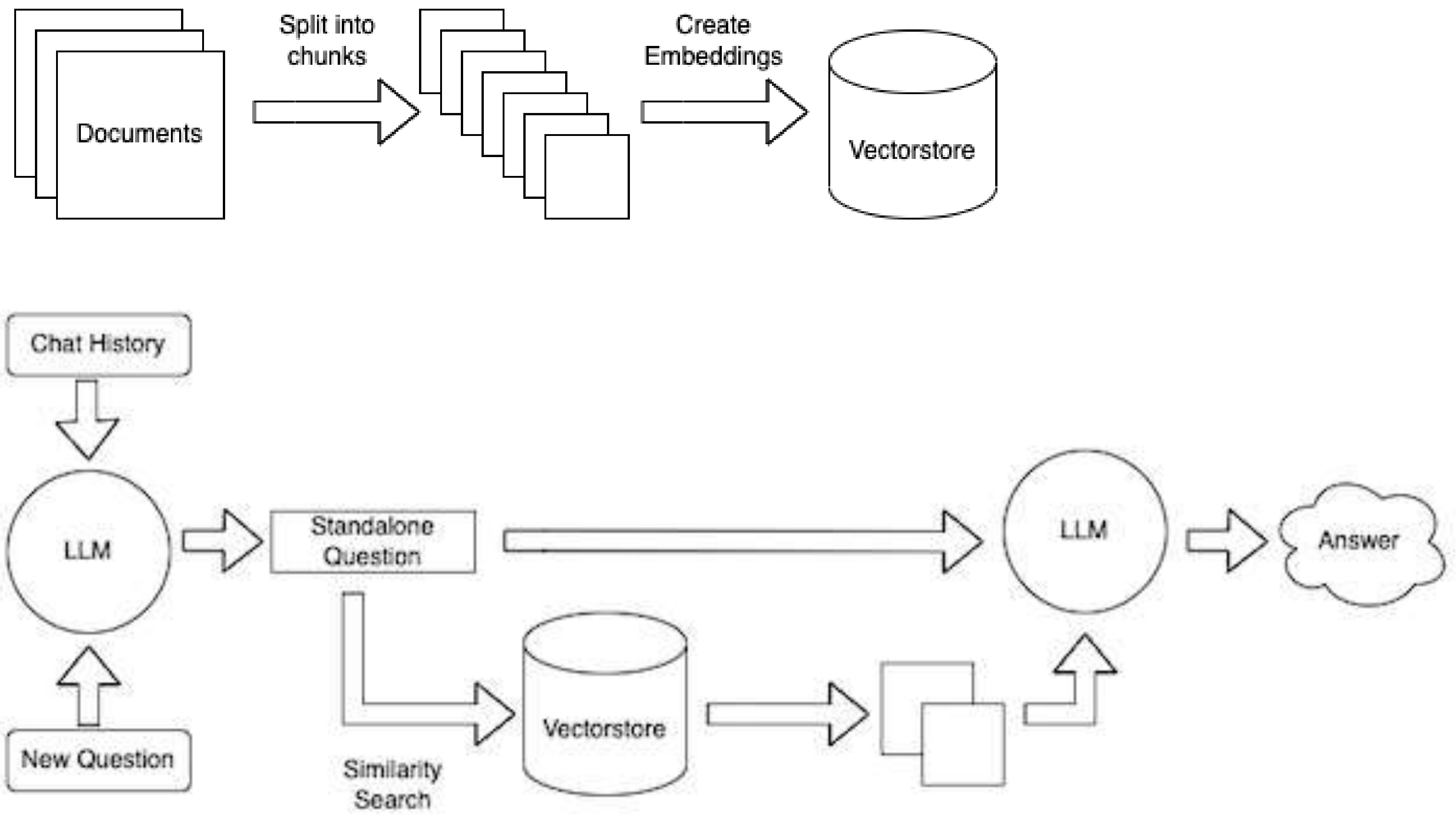
(Alternative: https://www.convex.dev/ai-town)

Background: LLM Agents in general



Source: https://lilianweng.github.io/posts/2023-06-23-agent/

Background: Retrieval Augmented Generation (RAG)



Source: https://blog.langchain.dev/tutorial-chatgpt-over-your-data/

Introduction

Project Goal

- **Define a compelling use case** in the educational domain for the utilization of (Generative) Agents
- Evaluate and select appropriate OS LLMs for your use case (like fine-tuned versions of quantized llama2 models, which can be deployed locally without requiring GPUs)
- Work with commonly used LLM techniques, like **Retrieval Augmented Generation** (RAG), **LLM Tools** or utilizing **Vector Databases**
- (optional) Create customized chatbots with different learning properties talking to each other
- (optional) Enable user interaction with Agents
- Final goal: **Develop proof-of-concept prototype for the showcase**
- Command line based is a minimum requirement, but you can build a frontend later or utilize current frontend/GUI solutions (simple chatbot structures, or more complex solutions similar to the GA environment)

Project Goal: Example Use Case 1 [starter]

- **Problem Statement**
  - Within Learning Technologies, there is a need to simulate learning environments and learning behaivor for applications like learning analytics. …

- **Use Case**
  - Simulate learning behavior of small learning groups and gather synthetic data by simulating learning situations with multiple agents. …

- **Possible Implementation and open questions**
  - Think of a learning group that is preparing for an exam on a topic of your choice (e.g., Introduction to Theoretical Philosophy, The History of Memes, Introduction to Natural Language Processing etc.).
  - Create 3-4 different agents with different characters (learning speed, verbosity etc.)
  - Let the agents generate exercises and questions for each other and let them discuss the answers
  - Find a way to simulate the learning progress. For example, let the agents provide answers to an exam (many example exams are available online), let them get some answers wrong in the beginning and get better over time
  - Find a way to track the progress of the agents, like metrics on how many answers an agent got wrong.
  - The conversation should also be logged, together with summaries of the discussions and results.

Project Goal: Example Use Case 1

- **Different possible solution**
    - Let a teaching agent generate exercises and questions based on one external source (like a free textbook in pdf form, or a long Wikipedia article)
    - The questions could be configurable for different learning types, for example: long, short, multiple choice, free form etc.
    - Create a small group of learner instances with different “characters” (based on different prompts, like “You are a motivated learner, but you get most answer wrong. Please generate a wrong answer to the asked question”)
    - Provide some kind of tracking. Logging Statements/summaries about what is happening

Project Goal: Example Use Case 2 [starter]

- **Problem Statement**
  - One common approach within classic chatbot architectures was the „concierge bot“: One „welcoming“ bot that takes in a user request and forwards the request to a chatbot that is most likely to find an answer to the question. The same idea could be beneficial in an LLM setting since there are many different types of OS LLMs available finetuned to specific tasks, like LLEMMA for mathematical problems. …
- **Use Case**
  - Ask questions on a variety of topics while utilizing expert LLMs according to a topic. …
- **Possible Implementation**
  - Feasible with LLM Tools, RAG, specified Agents…

Project Goal: Example Use Case 3 [starter] …

- **Problem Statement**
  - The goal of some learning processes goes beyond acquiring factual knowledge by focusing on skills like debating, where the acquired knowledge is applied. …
- **Use Case**
  - Simulate debating in a group consisting of simulations of different opinions. …
- **Possible solution**
  - Input a topic you want to debate like „are Large Language Models beneficial for society“, or „do aliens exist“
  - Generate debate exercises based on the topic, like concrete questions to discuss
  - Create different chatbots with different characteristics. For example, one dogmatic bot that is always against everything, one bot that usually does not understand the question and has to be corrected by others, one bot that answers cheerfully and is always happy to support the viewpoint of the user, etc.
  - Track the pro and contra arguments generated
  - A human user should be able to participate in the debate (with the bots reacting to the user arguments)
  - Provide feedback to the user (like an assessment of the answer, or metrics like time to response, length etc.)

Research Hints: Implementation

- In general: **You do not have to pay for anything!** Please use free and Open Source solutions
- If you have the feeling that working with CPU only is too slow: **use Google Colab, Kaggle Notebooks** etc.
- You can use: https://github.com/R3gm/InsightSolver-Colab/blob/main/LLM_Inference_with_llama_cpp_python__Llama_2_13b_chat.ipynb
- **Optional exercise**: Create an endless conversation between two bot instances, running in a google colab based on this notebook
- You can work with llama2-based GGUF models. Can be run on CPU and GPUs via google colab with llama-cpp-python
- There are also a lot of other models, like based on Mistral. If you use them, remember updating your prompt templates etc.
- It can be useful to use a dedicated server app for serving LLMs, like: https://github.com/langchain-ai/langchain/issues/10415#issuecomment-1720879489
- You **do not have to use agent frameworks** like langchain, but it might help at the start
- https://www.reddit.com/r/LocalLLaMA/ very useful, a lot of information and guides on how to set up local LLMs

Research Hints: Literature & other sources for inspiration

- Article: "LLM Powered Autonomous Agents": https://lilianweng.github.io/posts/2023-06-23-agent/

- Repo & Paper: "Generative Agents: Interactive Simulacra of Human Behavior": https://github.com/joonspk-research/generative_agents

- Also: look into PRs, forks etc.! For example: https://github.com/joonspk-research/generative_agents/pull/100 (although there are a lot of other ways to solve this specific issue you could try) Or forks like https://github.com/ElliottDyson/OSGA (unchecked)

- LangChain: https://python.langchain.com/docs/get_started/introduction

- An Open Source Framework for Autonomous Language Agents: https://github.com/aiwaves-cn/agents

- Khan Academy approaches, like:https://blog.khanacademy.org/harnessing-ai-so-that-all-students-benefit-a-nonprofit-approach-for-equal-access/ (Although the solutions from Khan Academy consider mostly 1 bot : 1 human situations)

- The Rise and Potential of Large Language Model Based Agents: A Survey https://arxiv.org/abs/2309.07864

- A Survey on Large Language Model based Autonomous Agents https://arxiv.org/abs/2308.11432

Research Hints: Evaluation

- Quantitative evaluation would be great, but it could be hard to come up with a framework in this context.
- So, a qualitative analysis on a small set of examples also possible
- Qualitative evaluation can be similar to the evaluation from the Generative Agents paper, but with your own findings and for different models
- For example: Evaluate the LLMs you use in your solution. Compare outputs of chosen prompts between 2-3 LLMs
- The main question is: What story can the selected examples tell?

Next steps

- Next workshop: **16 Nov 2023**
- Deadline for videos: **12 Nov 2023**
- Please prepare the videos. The talks should be:
  - Max. length of 10 minutes (!)
  - Provide a problem statement describing the context of your use case
  - Describe your user case
  - Describe your solution and how it is expected to perform
  - Include information on your approach (like descriptions, charts, diagrams etc.), a roadmap and your technology stack

- In case of problems: Try to research on Stack Overflow, reddit, discord servers etc. first. If you don't find anything: write to me with a detailed description of the problem of what you tried so far. I will answer directly, to all, or we will discuss it in the next workshop
- My mail address: max.upravitelev@fokus.fraunhofer.de
- Happy Coding ☺!