Deloitte.

Machine Learning Prague

A hands-on guide to AI Agents

Philipp Wendland, Ole Wegeleben
Deloitte AI Institute Germany

April 28th, 2025

Philipp Wendland
Deloitte AI Institute, Germany

Ole Wegeleben
Deloitte AI Institute, Germany

Agenda

Chapter 1

Introduction to AI Agents

Evolution from Chatbots to AI Agents

Chatbot

Stand-alone chat

Chat

Copilot

Providing support and enhancing productivity

Integration

Flexibility

Interaction

AI-Agent

Advanced AI system designed to perform specific tasks autonomously

Specialisation

Autonomy

Execution

Deloitte.

Now decides next:
Generating a new future

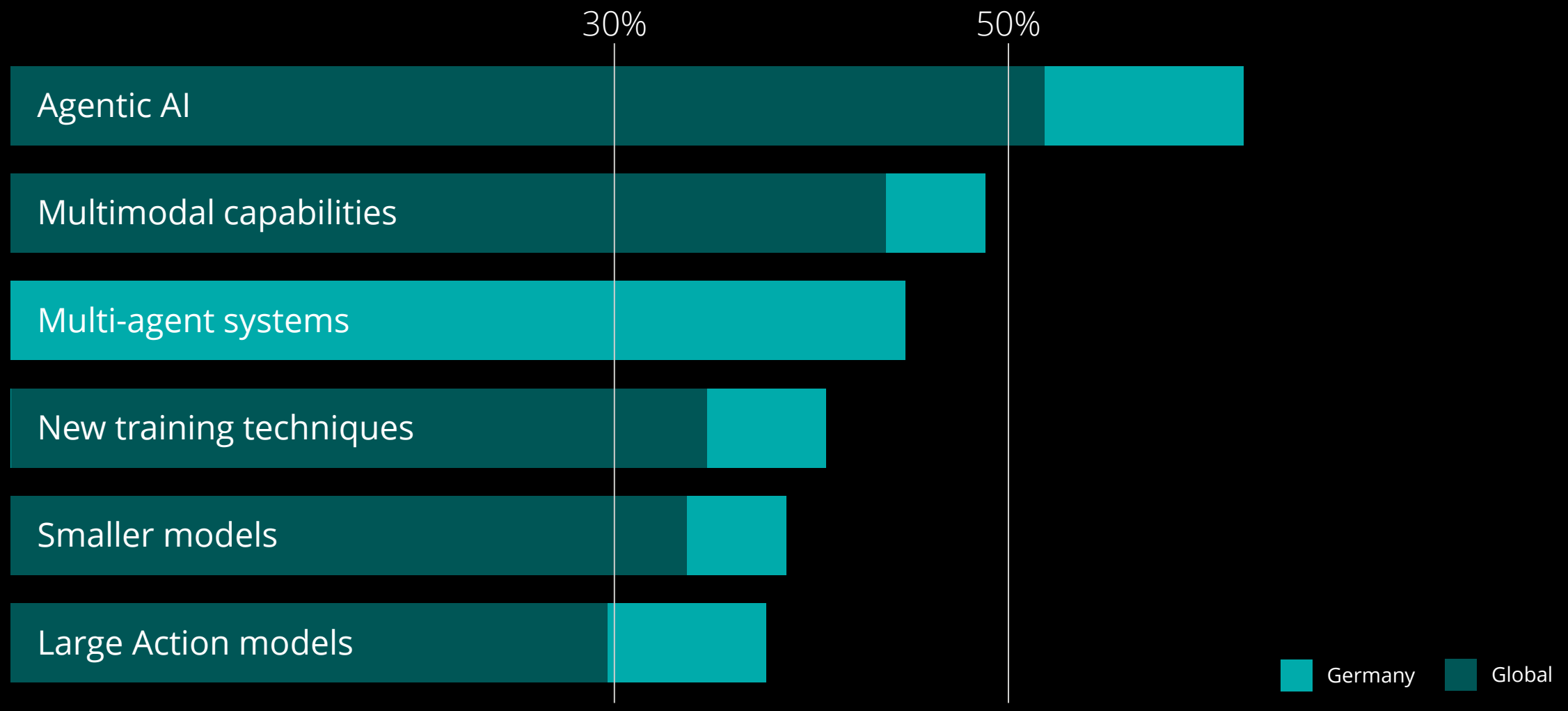
Deloitte's State of Generative AI in the Enterprise
Quarter four report – German Cut

March 2025

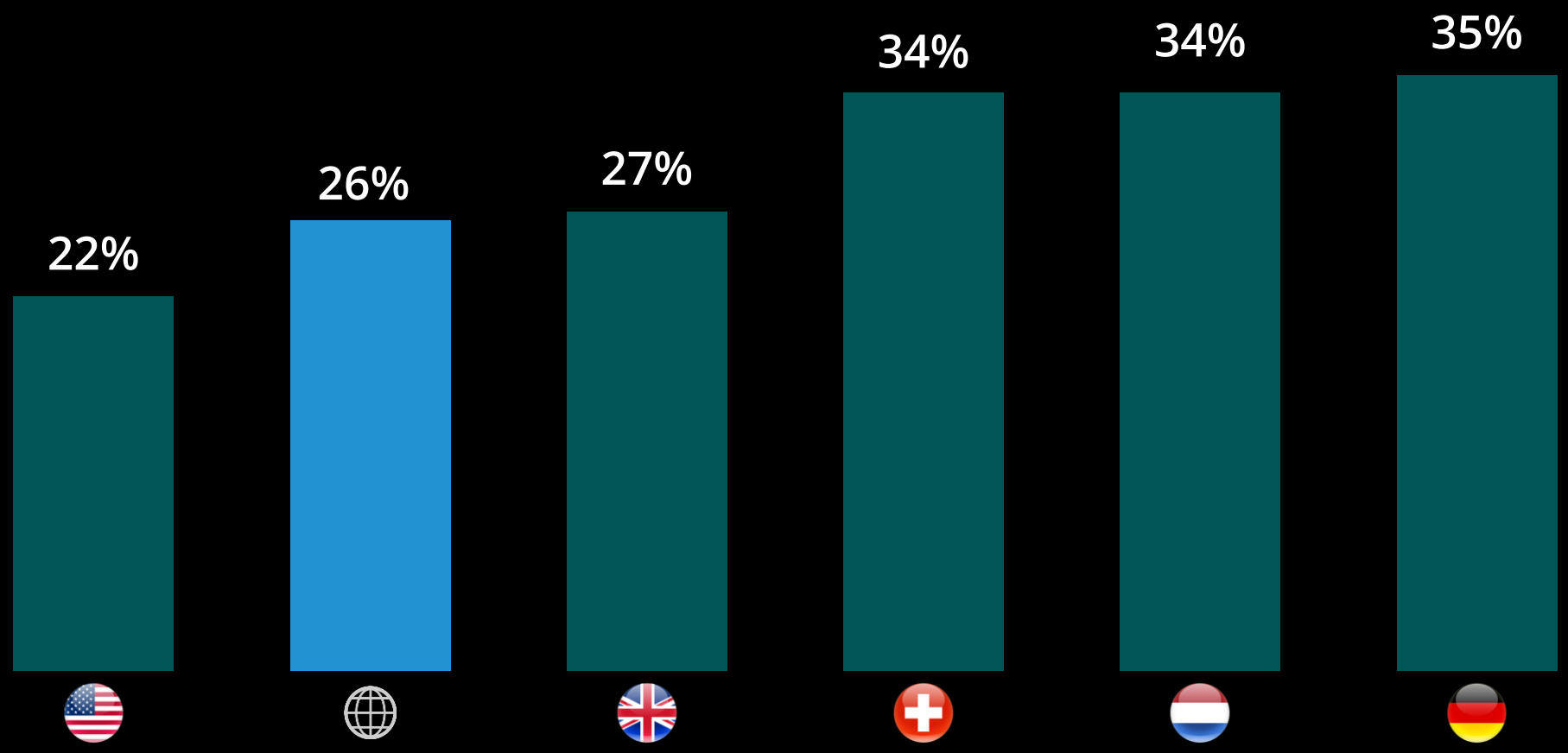
deloitte.com/de/ki-studie

https://www.deloitte.com/de/ki-studie

Organizations are most interested in Agentic AI



Is your organization exploring AI Agents?



Organizations choosing to a “large” or “very large extent”

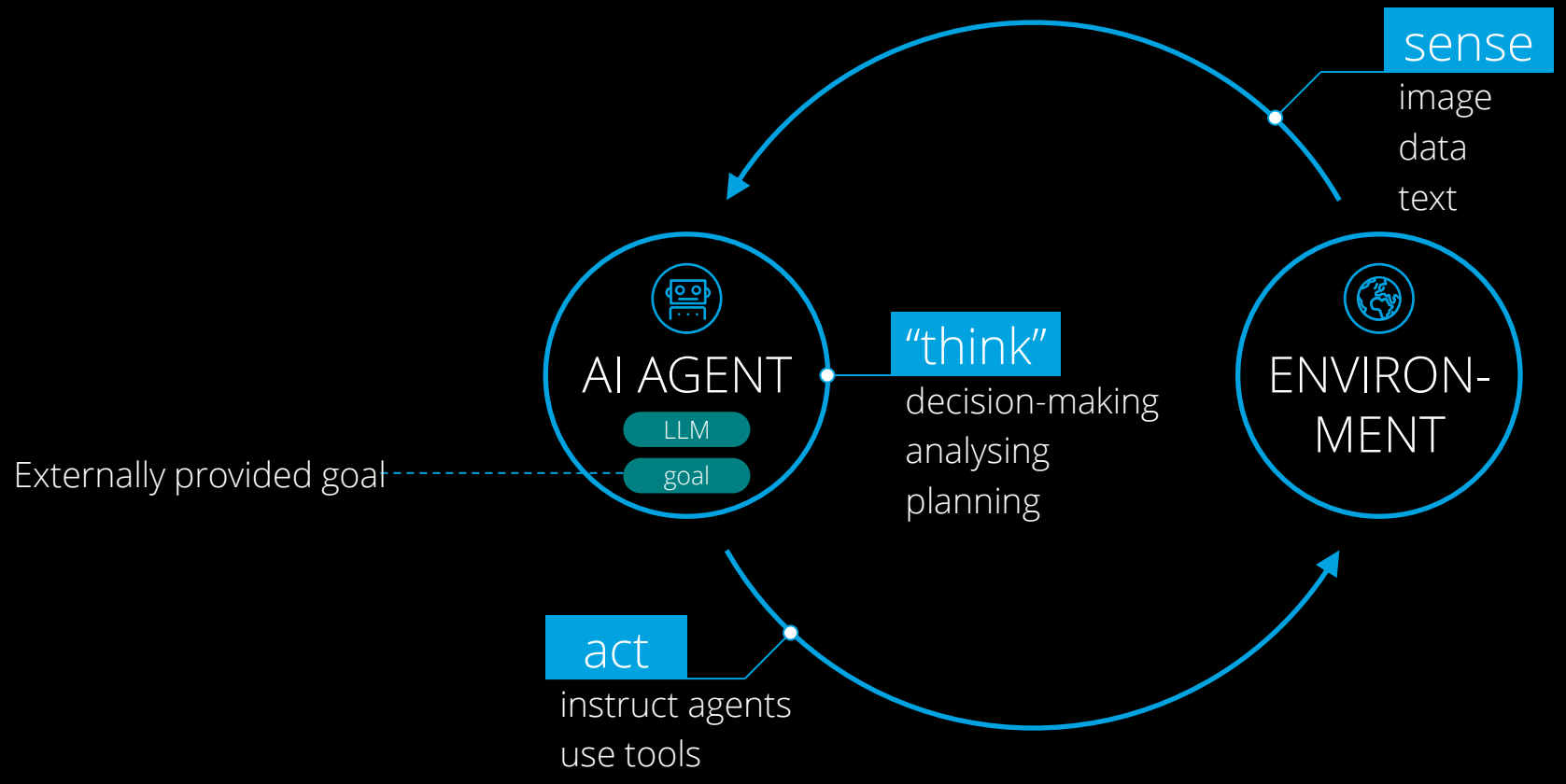
State of Generative AI in The Enterprise - Wave 4

AI AGENT

An AI agent is a software systems that uses AI and has access to tools to accomplish a goal on behalf of a user.

The anatomy of an AI-Agent

AI Agents autonomously decide the control flow and perform tasks using tools



“AI agents will transform the way we interact with technology, making it more natural and intuitive. They will enable us to have more meaningful and productive interactions with computers.”

Fei-Fei Li | Professor Stanford University

AI Agents can be built in a variety of environments

Several factors, incl. the use case and data availability will determine the ideal approach

	Platform integrated	Low/No-code	Developer frameworks
Type	Platform providers are introducing agent capabilities into their existing offerings Most simple to set-up, but only works within the specific platform and limited customization	Drag & drop agent builders allow users to build low-code agents relying on the expertise of established providers Best-practices on AI Agent building integrated	Code-based agent frameworks provide a structure to build advanced, customized multi-agent systems Provides full flexibility and requires expertise in AI agent building
Flexibility	Low	Medium	High
Ease of use	High	Medium	Low

Production-grade agents are part of ecosystems

Data & Integrations

Data Integration & Orchestration using APIs, events, messages & knowledge repositories

Business Process Layer

Conducted and governed by people and business applications

Multi-Agent AI Systems (MAS)

Coordination amongst each other and with humans to accomplish complex tasks

"Humans in Loop" SOPs

Standard Operating procedures (SOPs) leveraging agentic frameworks & prompts with humans in loop

Agents

Plan and act as per role by using its knowledge, memory and tools

Workflows

Orchestrate Agents with Human in the loop to automating processes

Models, Tools & More

Model (AI/ML, LLM, SLM, etc.) training, mgmt. and finetuning; Automation & Integration Tools etc.

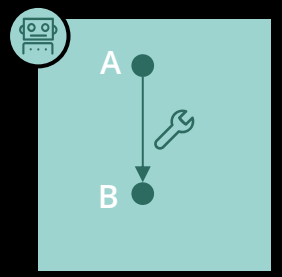
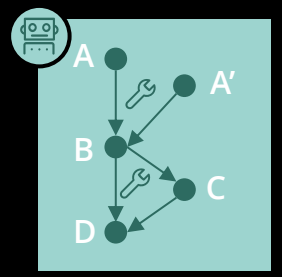
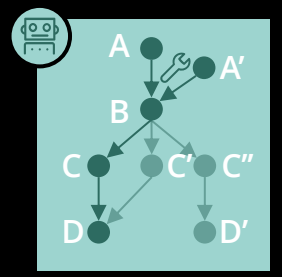
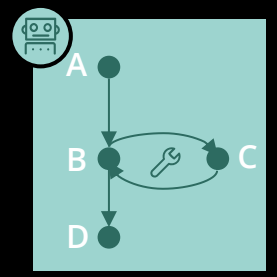
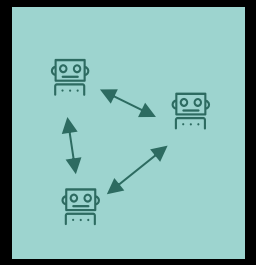
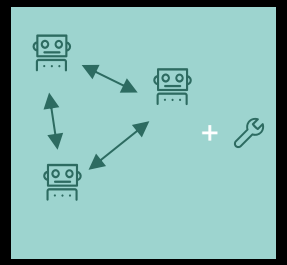
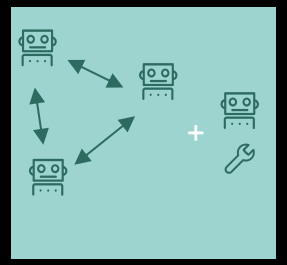
AI & Data Infrastructure, Platforms and Toolkits

High-performance hardware, and AI and data platforms (Hybrid Cloud)

Vertical Agent Use Cases

Agent based use cases and solutions for specific functional and domain areas specifically built and customized for an enterprise and its needs

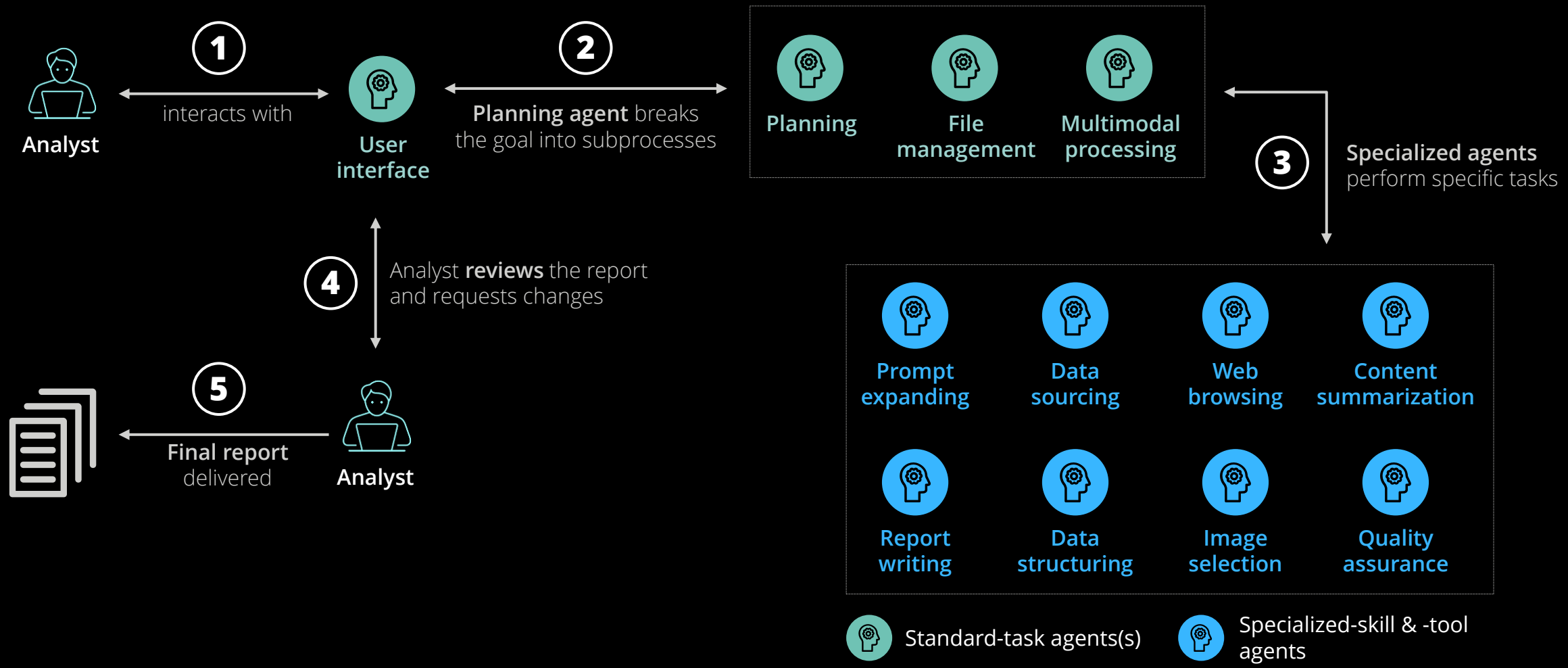
Agents exist on a spectrum from simple to complex

"Agent" type	Prompt	Prompt-Chains	Dynamic prompt-chains	Prompt-Loops	Multi-agent systems (MAS)	MAS + tool choice	Autonomous systems
							
Agentic capability	One prompt generates one output (by e.g., calling a tool)	Prompts are chained, final output is based on several intermediate steps	Prompts are chained, but the agent autono-mously decides which path to take during run-time	Agent has control over workflow execution paths, including cycles (i.e. moving backwards) and loop iterations	MAS is capable of designing complete workflows, given a fixed number of agents and tools	With a fixed number of agents, the MAS can provision new tools (e.g., from existing repositories)	The system can build new tools, create agents, and provision own infrastructure

Degree of agency

Adopt | Experiment | Vision

Multi-Agent-System for research and reporting



The game of protocols

Protocols aim to enhance LLM capabilities and agent orchestration

MCP (Model Context Protocol)

Developed as an Open Standard to let LLMs mount data sources and tools

Unified communication between LLM and external systems / tools

Allows quick and simple integration through MCP servers

A2A (Agent to Agent protocol)

Allow agent to agent communication and interoperability across various agent frameworks

Agent card in JSON format

Relies on existing standards (htttp, json)

ACP (Agent communication protocol)

A standard for local multi-agent communication (i.e., agent-to-agent messaging)

Focus on single-environement

MCP: https://www.anthropic.com/news/model-context-protocol, ACP: https://research.ibm.com/blog/multiagent-bee-ai, A2A: https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability/

Risks & limitations of multi-agent systems

Premature termination

Design deficiencies

Coordination complexity

Bias

Performance variability

Exposure of information

Malicious Behavior

Tracing difficulties

Sources: Why do multi-agent LLM systems fail (https://arxiv.org/pdf/2503.13657)

The difference between jobs and skills is key

Understanding where and what AI systems can augment humans is crucial

Work

is the result of harnessing human and tool capabilities

Meeting sales, UX, and customer satisfaction targets

Desired outcomes are and will be defined by humans

Jobs

define the work humans do to achieve outcome

Roles like sales reps, developers, account managers

Not jobs are automated, but tasks and jobs may be redefined based on skills

Tasks

are specific activities within jobs aimed at producing outcome

Identifying sales channels or tailoring product offerings

AI's role lies in task automation and efficiency enhancement

Skills

enable us to complete tasks and attain work outcomes

Problem solving, coding, and data analysis skills

Skills can reside in humans and AI

Crucial for evolving work with Gen AI integration

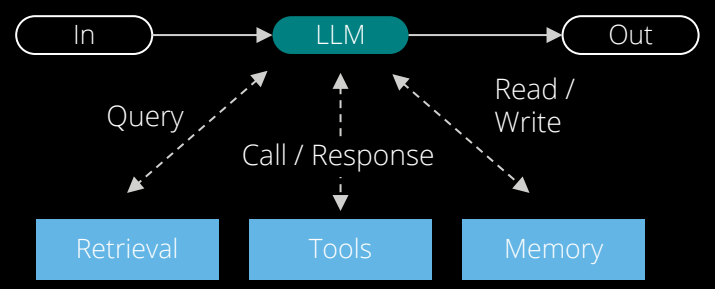
Chapter 2

Exercise Block 1

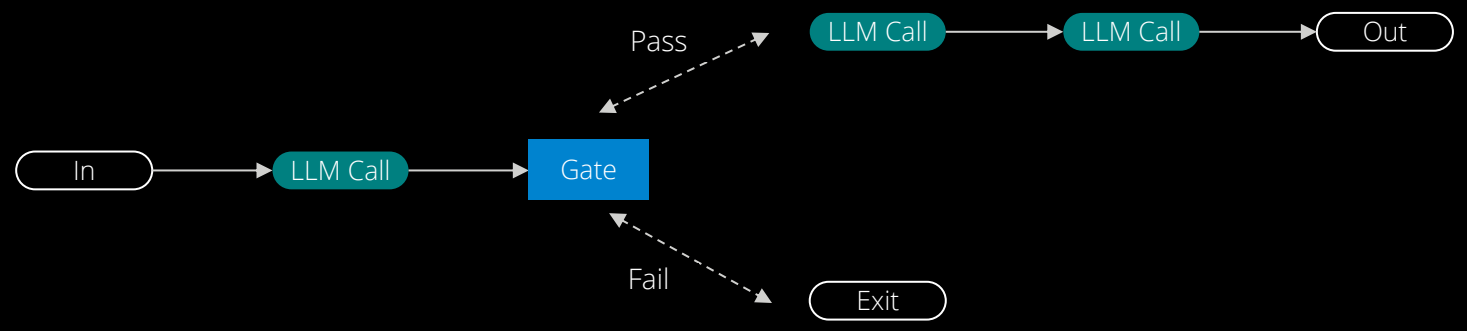
Anthropic on “Building effective agents”

“Successful implementations use simple patterns rather than complex frameworks”

Augmented LLMs



Prompt-Chaining Workflow



https://www.anthropic.com/engineering/building-effective-agents

ML PRAGUE 2025: A HANDS-ON GUIDE TO AI AGENTS

Exercises 1, 2 & 3a

Prompt-chain workflows and basic tool-calling

Exercise 1

- ✓ Setup the environment & install dependencies
- ✓ Adjust the given prompt
- ✓ Happy with the LLM output? Use it as a baseline for the more complex workflows & agents

Exercise 2

- ✓ Use the sequential_chain method
- ✓ Write an additive_chain method
- ✓ Try out your chain with a new use case

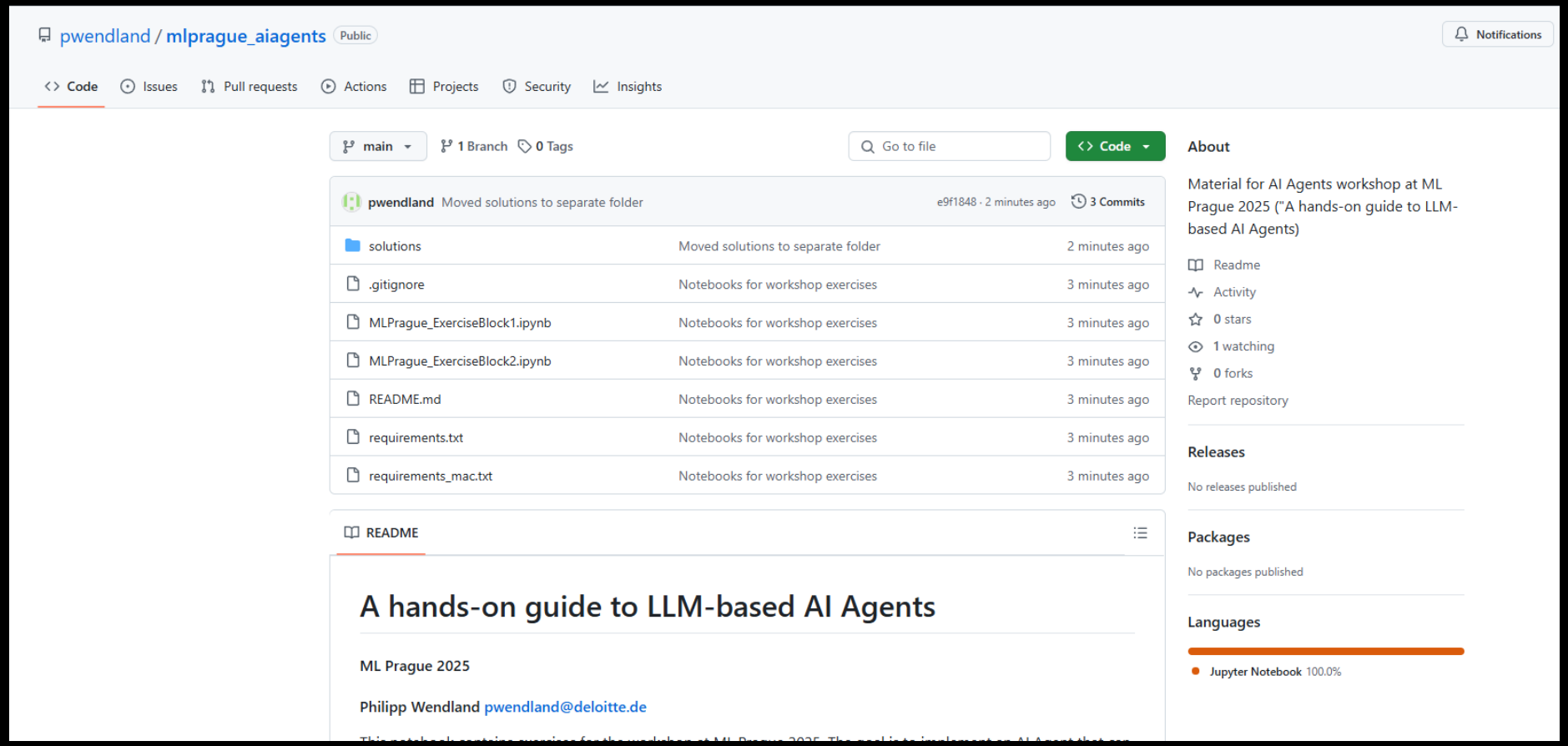
Exercise 3a

- ✓ Extend the previous functions to accept a tools parameter
- ✓ Adjust the prompt chain to include a tool call

ML PRAGUE 2025: A HANDS-ON GUIDE TO AI AGENTS

Getting ready for Exercise Block 1

https://github.com/pwendland/mlprague_aiagents



Coffee Break: Be back by 16:00

Clarify any setup-issues to be ready for smolagents

Chapter 3

Intro to Exercise Block 2

Myriads of agent frameworks have emerged

We'll take a closer look at smolagents today

smolagents

- Lightweight modular agents
- Self-contained agent classes with basic tools and memory

LangGraph

- Build multi-agent workflows as directed graphs
- Offers fine-grained control over agent interactions and state management

Implementation varies depending on “Agency Level”

Agency Level	Description	How that's called	Example Pattern
☆☆☆	LLM output has no impact on program flow	Simple processor	`process_llm_output(llm_response)`
★☆☆	LLM output determines basic control flow	Router	`if llm_decision(): path_a() else: path_b()`
★★☆	LLM output determines function execution	Tool call	`run_function(llm_chosen_tool, llm_chosen_args)`
★★★	LLM output controls iteration and program continuation	Multi-step Agent	`while llm_should_continue(): execute_next_step()`
★★★	One agentic workflow can start another agentic workflow	Multi-Agent	`if llm_trigger(): execute_agent()`

https://huggingface.co/blog/smolagents

Basic smolagents architecture

smolagents consists of several agent types

A CodeAgent is the main type of agent introduced by the smolagents framework

```
from smolagents import CodeAgent, DuckDuckGoSearchTool, HfApiModel

agent = CodeAgent(tools=[DuckDuckGoSearchTool()], model=HfApiModel())

agent.run("Search for the best music recommendations for a party at the Wayne's mansion.")
```

```
from smolagents import ToolCallingAgent, DuckDuckGoSearchTool, HfApiModel

agent = ToolCallingAgent(tools=[DuckDuckGoSearchTool()], model=HfApiModel())

agent.run("Search for the best music recommendations for a party at the Wayne's mansion.")
```

https://huggingface.co/learn/agents-course/en/unit2/smolagents/tool_calling_agents

Structure of a tool in smolagents

```python
@tool
def get_weather(latitude: float, longitude: float) -> dict:
    """
        Obtains the current weather based on a given location
        Args:
            latitude: float of latitude of the given location
            longitude: float of longitude of the given location
        Returns:
            dict containing
                "current_temp": current temperature in celsius
                "cloud_cover": current cloud cover in percent
                "percipitation": current percipitation in mm
    """

    try:
        response = requests.get(f"https://api.open-meteo.com/v1/forecast?latitude={latitude}&longitude=
{longitude}&current=temperature_2m,cloud_cover,precipitation&hourly=temperature_2m,precipitation_probability,cloud_c
over")

    except:
        return "Weather data not available"

    response_data = response.json()


    return {
        "current_temp": response_data['current']['temperature_2m'],
        "cloud_cover": response_data['current']['cloud_cover'],
        "precipitation": response_data['current']['precipitation']
    }
```

Exercises 3b & 4

Prompt-chain workflows and basic tool-calling

Exercise 3b

- ✓ Setup a CodeAgent & examine the system prompt
- ✓ Create an itinerary using a single agent with multiple tools

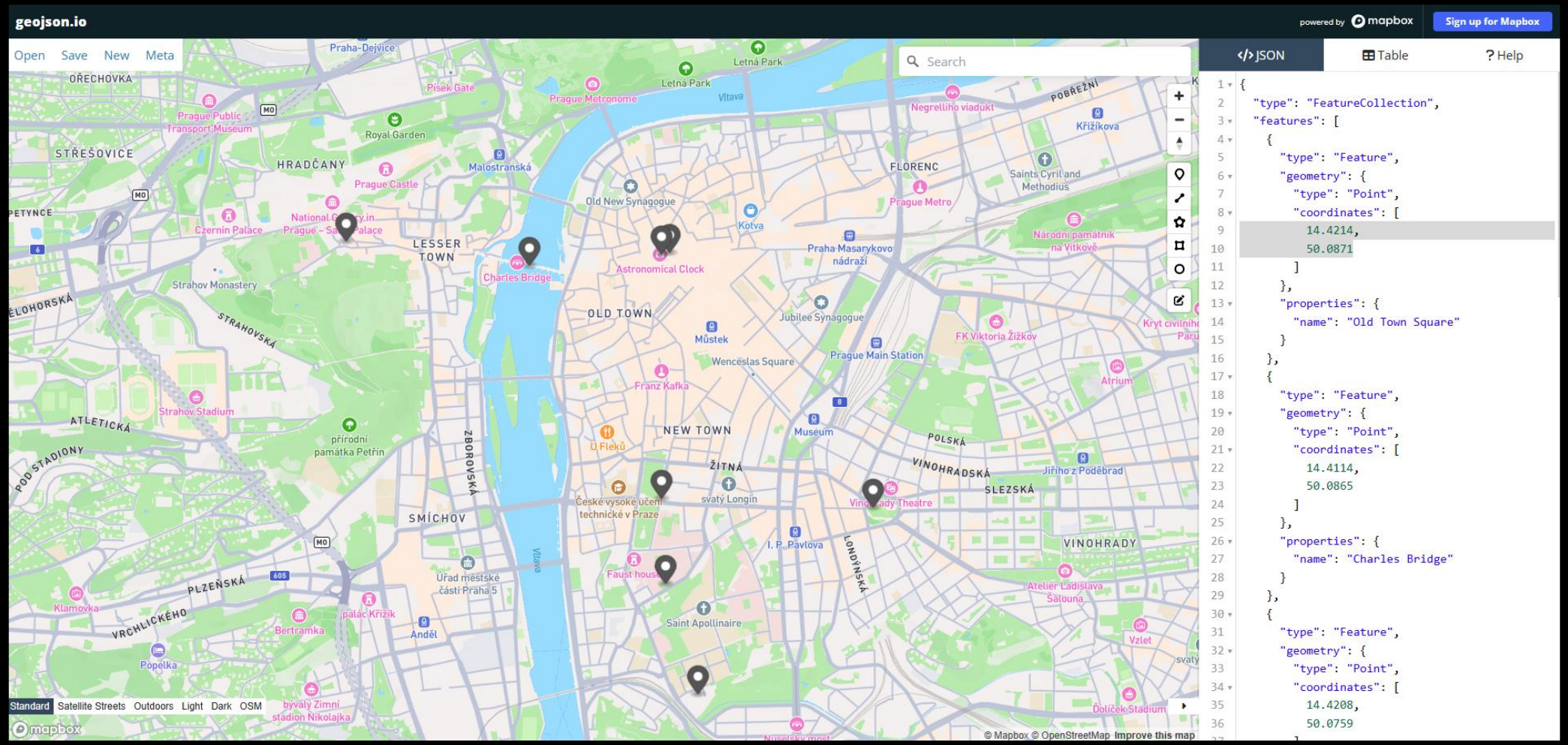
Exercise 4

- ✓ Create a multi-agent system
- ✓ Implement a new tool, create an agent and add the agent to the multi-agent system (e.g.,

Bonus

- ✓ Download a tool from the hub or MCP servers

Create a map of famous sights in Prague




```
{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "geometry": {
        "type": "Point",
        "coordinates": [
          14.4214,
          50.0871
        ]
      },
      "properties": {
        "name": "Old Town Square"
      }
    },
    {
      "type": "Feature",
      "geometry": {
        "type": "Point",
        "coordinates": [
          14.4114,
          50.0865
        ]
      },
      "properties": {
        "name": "Charles Bridge"
      }
    },
    {
      "type": "Feature",
      "geometry": {
        "type": "Point",
        "coordinates": [
          14.4208,
          50.0759
```

Chapter 5

Outlook

Considerations beyond experimentation

For productive use cases using AI Agents additional aspects need to be considered

Testing

Domain & Process knowledge

Business value (human-in-the-loop)

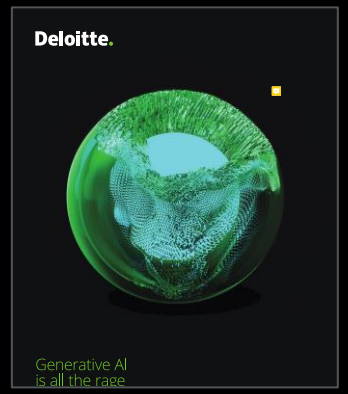
Governance & ethics

SELECTION

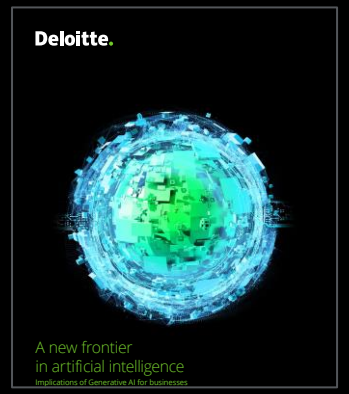
Further reading on Generative AI

Our thought leadership on (Generative) AI is continuously expanding

Generative AI is all the Rage



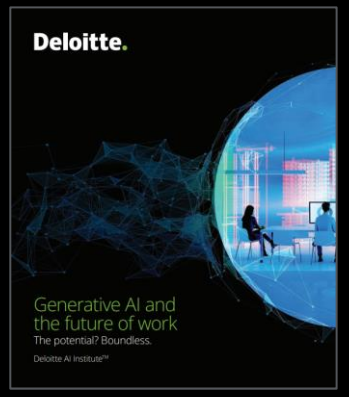
A new frontier in artificial intelligence



How AI agents are reshaping the future of work

The legal implications of Generative AI

Generative AI and the future of work



Proactive risk management in Generative AI

State of Generative AI 2024

State of Generative AI, German Cut

Generative AI Dossier

Artificial Intelligence Act

Deloitte.
Digital

Thank you.

Philipp Wendland
Senior Consultant | AI Institute
pwendland@deloitte.de
Rosenheimer Platz 4
81669 Munich, Germany
Deloitte Consulting GmbH

Ole Wegeleben
Senior Consultant | AI Institute
olwegeleben@deloitte.de
Dammtorstraße 12,
20354 Hamburg, Germany
Deloitte Consulting GmbH

Sources

Deloitte
- Prompting for action: https://www2.deloitte.com/content/dam/Deloitte/us/Documents/consulting/us-ai-institute-generative-ai-agents-multiagent-systems.pdf
- GenAI and the future of work: https://www2.deloitte.com/content/dam/Deloitte/us/Documents/consulting/us-ai-institute-generative-ai-and-the-future-of-work.pdf

Anthropic
- Anthropic cookbook (https://github.com/anthropics/anthropic-cookbook/tree/main)
- Building effective Agents: https://www.anthropic.com/engineering/building-effective-agents
- MCP: https://www.anthropic.com/news/model-context-protocol

Huggingface
- Introducing smolagents: https://huggingface.co/blog/smolagents
- AI Agent course: https://huggingface.co/learn/agents-course/unit0/introduction
- Smolagents documentation: https://huggingface.co/docs/smolagents/v1.14.0/en/index

Agent Communication protocol: https://research.ibm.com/blog/multiagent-bee-ai

Agent to agent protocol: https://research.ibm.com/blog/multiagent-bee-ai

M. Cemri et al. Why do multi-agent systems Fail (https://arxiv.org/pdf/2503.13657)

OpenAI Documentation: https://platform.openai.com/docs/guides/function-calling

LiteLLM Documentation: https://docs.litellm.ai/

Resources

https://github.com/pwendland/mlprague_aiagents