Technische
Universität
Berlin

WiSe 25/26

Project Advanced Media Technologies: „Middleware for GenAI“

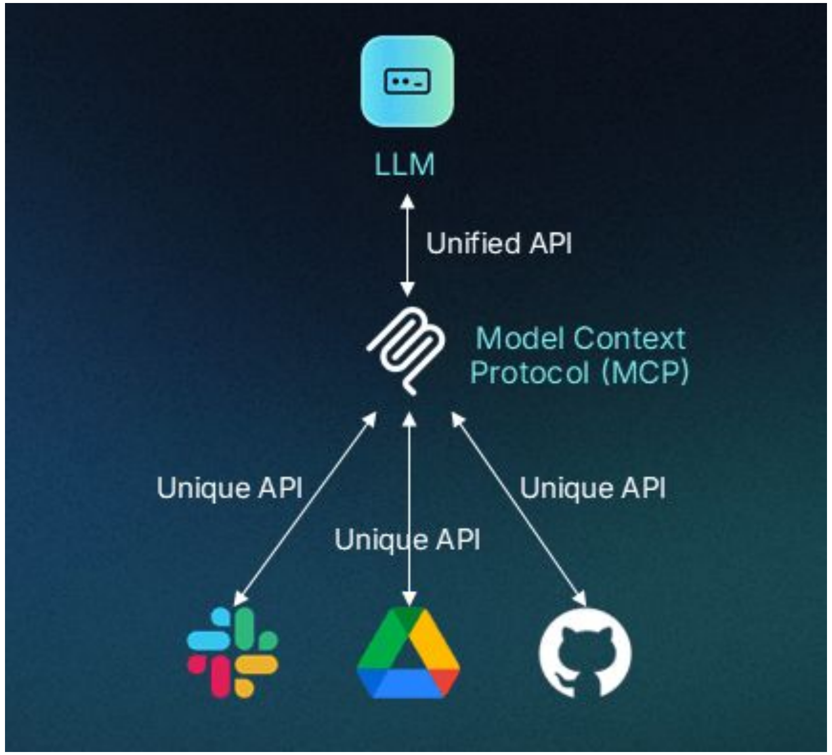
Arno Bock (arnobock_1@campus.tu-berlin.de), Shady Kadry (kadry@campus.tu-berlin.de), George Badour (george.badour@campus.tu-berlin.de)

Terminology and Technology

- What is **GenAI?**
  - Generative AI (e.g. GPT, Claude) that creates text, code, etc.
  - Powers chatbots, copilots, content tools.

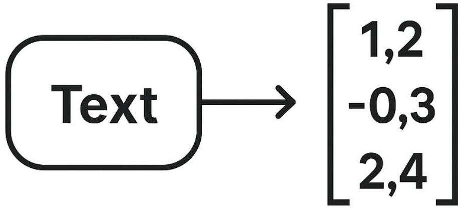
- What is **MCP?** *(Model Context Protocol)*
  - Standard for connecting LLMs to tools.
  - Separates clients (chatbots) from servers (tools).

- What is **RAG?** *(Retrieval-Augmented Generation)*
  - LLM retrieves relevant documents before answering.
  - Makes responses factual, grounded and domain-aware.



Terminology and Technology - Part 2

- What is an **embedding pipeline?**
  - Transforms text or data into vector representations *(embeddings)* that capture meaning.
  - Used for search, retrieval and context building in GenAI systems.



- What is **Policy Control?**
  - Manages access rights and data usage based on user identity or role.
  - Ensures security, compliance and responsible AI behavior across apps

- **What do we need all of this for?**
  → To build reliable, reusable and policy-compliant GenAI systems.

Problem Statement

- **Problems with state-of-the-art orchestration**
  - Fragmented components
  - Repeated custom integrations
  - Policy control challenges

- **Problem statement:** Low reusability of existing GenAI applications due to individual orchestration of:
  - MCP host/server registration
  - Embedding pipelines
  - Database access
  - Policy control mechanisms

Literature Review

Existing frameworks or tools with partial solutions:

- **LangChain**[1] (**No** Cross-app standardization, **No** Robust access control)
- **LlamaIndex**[2] (embedding manager incl. pipelines)

[1] H. Chase et al., *LangChain: A framework for developing applications powered by large language models*. LangChain, 2025. Available online: https://python.langchain.com/docs/introduction/

[2] J. J. Liu et al., *LlamaIndex: The framework for context-augmented LLM applications*. LlamaIndex, 2025. Available online: https://docs.llamaindex.ai/

Literature Review

LangChain

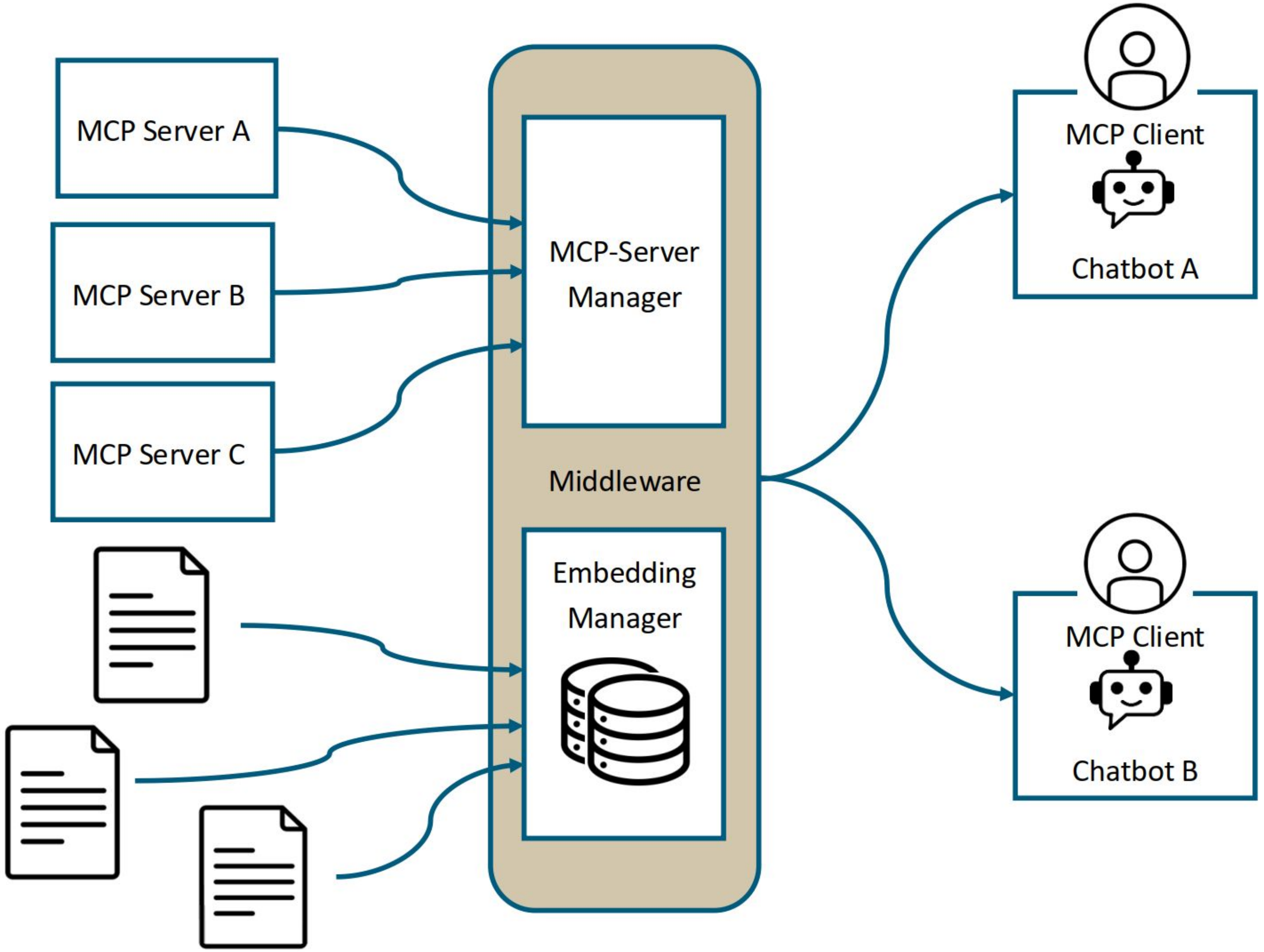
- Strong abstractions: prompts, retrievers, tools, agents, LCEL/Runnables.
- Supports many integrations:
  - vector databases
  - LLMs
  - APIs
  - developer tooling
  - Database access
- Build and orchestrate pipelines within a single app
- No support for MCP server/client registration or tool/resource discovery
- No built-in policy engine (RBAC/ABAC); access control handled externally or ad hoc

Literature Review

LlamaIndex

- Offers robust ingestion: document → node conversion, chunking, metadata, embedding
- Provides retriever and query engine components usable in flexible pipelines
- Supports multiple vector store backends; strong for in-project content workflows
- Does not include MCP support *(no server/client discovery or routing)*
- Policy control limited to metadata filters; lacks centralized policy enforcement

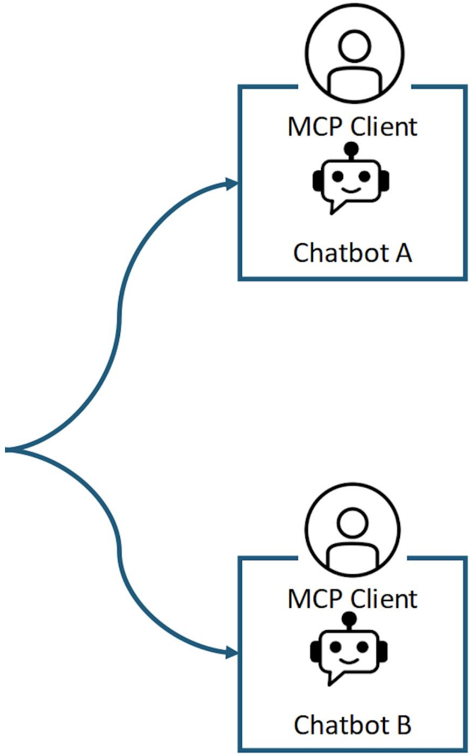
Potential Solution: Middleware Concept



- Eliminates need for **MCP-Clients** to implement:
  - Authorization control
  - MCP host
  - Embedding pipelines
  - Database access/registration

Middleware Concept - Requirements

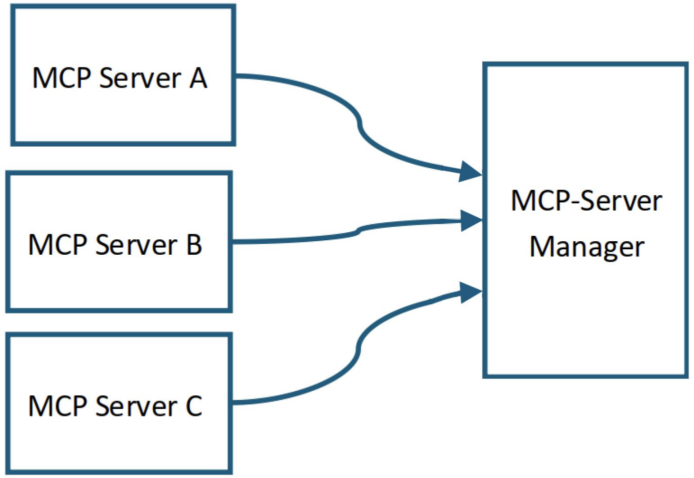
- **MCP-Clients** *(LLM chatbots)*
    - Receive prompts through user interface
    - Pass to middleware and await response
    - Answer based on retrieved information



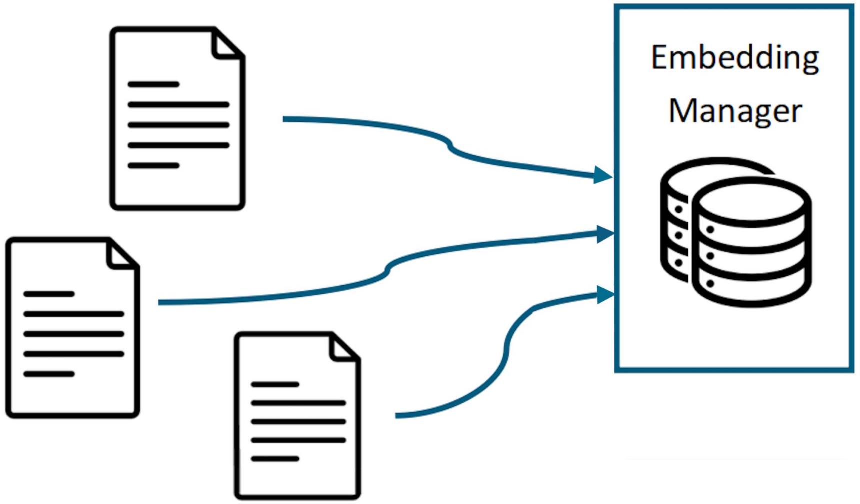
- **Middleware**
    - Authorize user and parse prompt
    - Communicate with MCP-Server manager/Embedding manager
    - Return information back to MCP-Client

Potential Solution - System Design

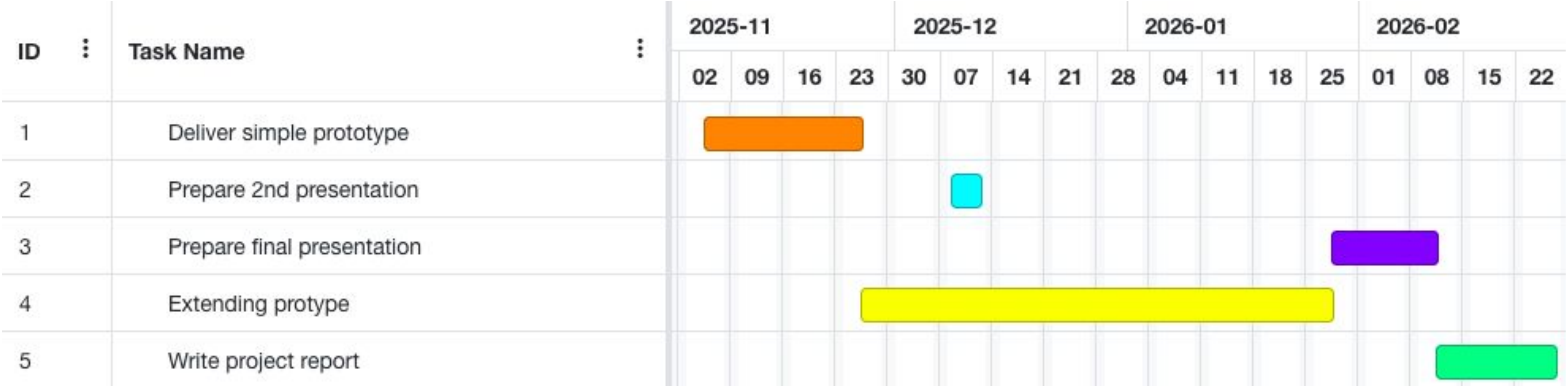
- **MCP-Server Manager**
  - Manage internal MCP-Server registry
  - Route prompts to relevant servers *(capability-based)*
  - Register new servers through admin user only



- **Embedding Manager**
  - Embed prompts based on pipeline registry
  - Upload data through admin user only
  - Registers/Manages different vector databases
  - Performs database session control *(user-based)*



Project Schedule



ID	Task Name
1	Deliver simple prototype
2	Prepare 2nd presentation
3	Prepare final presentation
4	Extending protype
5	Write project report

Next steps:

- **Subtask: “Deliver simple prototype”**
  - Create middleware application
  - Attach LLM-based chatbot
  - Create an MCP-Manager/host
  - Create an embedding manager

- **Coming-up: “Extend prototype”**
- **Coming-up: “Prepare 2nd presentation”**

Technische Universität Berlin

Thanks for your attention!

Any Questions?