Jurnal etapa 1 – Ricu Alexandru Razvan

Cerinte rezolvate

A fost instalat Ollama si s-a ales un model compatibil cu sistemul (gpt-oss:20b).
S-a implementat un server MCP, care ruleaza pe portul 8001 si gestioneaza operatii asupra unui director.
Serverul expune urmatoarele tool-uri:

- list_directory(dir_path) – listeaza fisierele si directoarele dintr-un folder
- get_file_content(file_path) – returneaza continutul unui fisier
- list_directory_recursive(dir_path) – listeaza recursiv toate fisierele si directoarele
- read_files_in_directory(dir_path) – citeste continutul tuturor fisierelor dintr-un folder (non-recursiv)
- read_files_recursively(dir_path) – citeste recursiv toate fisierele
- delete_file(file_path) – sterge un fisier dat
- delete_folder(folder_path) – sterge un folder dat

Modul de rezolvare

Pentru implementare am instalat pachetul Ollama, utilizat pentru rularea modelului local gpt-oss:20b, precum si biblioteca google.adk, care ofera infrastructura pentru definirea agentilor si integrarea cu MCP. De asemenea, am folosit LiteLlm, un wrapper pentru modelele Ollama, care permite comunicarea eficienta intre agent si modelul de limbaj. Biblioteca mcp.server.fastmcp a fost utilizata pentru implementarea unui server MCP capabil sa gestioneze directoare si fisiere.

Serverul OS-MCP-Server ruleaza pe portul 8001 si expune tool-urile cerute (list_directory, get_file_content), alaturi de functionalitati suplimentare (list_directory_recursive, read_files_recursively, delete_file, delete_folder). Fiecare functie este adnotata, documentata si include tratari de erori.

Agentul OS_Agent a fost creat pentru a interactiona exclusiv cu sistemul de fisiere prin intermediul MCP, folosind modelul LiteLlm. Acesta respecta reguli stricte privind utilizarea tool-urilor, confirma actiunile de stergere si gestioneaza erorile in mod controlat. De asemenea, agentul a fost integrat intr-o interfata web locala realizata cu ADK Web, accesibila la localhost:8080, care permite utilizatorului sa comunice direct cu agentul si cu tool-urile expuse.

Testele efectuate confirma functionarea corecta a comunicatiei intre agent si serverul MCP, precum si interactiunea prin interfata web, conform cerintelor etapei 1.

Probleme intalnite si modul de rezolvare

In realizarea cerintelor proiectului, prima dificultate intampinata a fost citirea atenta a documentatiei pentru a intelege structura proiectului, in special rolul fisierului agent.py si al __init__.py. Initial am denumit fisierele altfel, iar ADK nu recunoastea agentul, ceea ce a dus la erori in rulare.

O alta provocare a fost configurarea comunicarii prin Streamable HTTP. La conectarea agentului (agent.py) cu serverul MCP (mcp-server.py) am gresit portul, ceea ce a cauzat imposibilitatea de a stabili conexiunea pana la corectarea parametrilor.

De asemenea, prompt engineering-ul a necesitat atentie: a fost nevoie sa invat modelul ce functii sa foloseasca, cum sa trateze erorile si cum sa respecte regulile stabilite in instructiunile agentului.

Ca imbunatatiri viitoare, ar fi util sa testez agentul pe un model mai mic, deoarece modelul actual (gpt-oss:20b) are nevoie de un timp considerabil pentru a formula raspunsurile, chiar daca acestea sunt corecte.

Concluzii

In aceasta faza a proiectului, am realizat implementarea unui agent AI capabil sa interactioneze cu sistemul de fisiere prin intermediul unui server MCP, folosind un model LLM local (gpt-oss:20b) integrat prin LiteLlm.

Din acest proiect am invatat importanta structurii corecte a proiectului, configurarea precisa a conexiunilor intre agent si server, prompt engineering-ul pentru a instrui modelul sa foloseasca tool-urile corect si, nu in ultimul rand, scrierea de scripturi Python pentru manipularea fisierelor (citire, listare, stergere).