| | |
|---|---|
| Assignment 1 | Project Summary |
| Course | Intelligent Agents with Generative AI - 2024 |
| GitHub Repository | https://github.com/Kristiyan025/ai-assistant |

| Project author | | |
|---|---|---|
| № | Name | Faculty Number |
| 1 | Kristiyan Garchev | 0MI0800290 |

| | |
|---|---|
| Project name | MyPAI |

## 1. Short project description (Business needs and system features)

Generative AI web-based agents have been adopted by many businesses and individuals as intelligent assistants, automating tasks such as information retrieval, Q&A, task automation, content generation, training, etc. The current project aims to develop a multiagent system for a web-based AI assistant, that can answer general questions, play music from Spotify, execute/automize file related tasks, answer questions regarding a collection of documents, etc. The system has a chat-bot interface, but also can accept voice commands. It has a file workspace that the user can view/modify. Furthermore, the system can create multiple RAGs (abstracted as subagents and user can choose to have conversations with them) and answer questions regarding specific resources. Also, it keeps track of all past conversations. The project will use *LangChain+LangGraph, Llama 3.1/3.2 LLM* with *Ollama and ChatGPT*, capable of supporting person's needs for interactive information retrieval / content modification / task automation, etc. The project is a mix of interactive agents (voice/chat), tool agents (file ops, transformations, Spotify), knowledge/ML agents (LLM + RAG subagents) and orchestration / infra agents (supervisor). The multiagent system will have an open architecture allowing addition of new specialized agents in the future.

The system is called MyPAI (My Personal AI assistant) and will consist of following agents:

1. *Hardware Wrapper Agent* (HWAs) – these agents will perform the function of a software proxy wrappers of the hardware devices used for the project: microphone & speakers. For the microphone, it can perform additional tasks such as reducing noise via another agent.
2. *Voice I/O Agent* (Speech Agent) (VIOA) – the agent will convert user speech ↔ text handle wake words / session audio streaming. It will communicate with the VIOA for input output.
3. *Session Manager Agent* (SMA) – the agent will manage the user sessions, i.e. store/retrieve the data for all text/audio chats with the user.
4. *Task Planner Agent* (TPA) – the agent will receive the query from the user (via the interface agents) and will create an execution plan for the query, i.e. which agents and in what order should be involved to execute the query.
5. *Web Search Agent (*WSA) - the agent will perform web queries to popular search engines (as described in https://www.semrush.com/blog/python-for-google-search/).
6. *File System Agent* (FSA) – this agent is the system's reverse proxy to the file system. It manages mainly the system's workspace, but it can access (in a restricted manner)

other parts of the file system as well. It executes file commands but also enforces policy - cannot delete or modify external files, i.e. only can create/copy external files.

7. *File Transform Agents* (FTAs) – these agents will perform file conversions and transformations (i.e. png→jpeg, rotate PDF page, reorder pages, etc.) using either local tools (ImageMagick, PyPDF2) or external APIs. Also, some queries may involve custom transformations, so there will be one general agent that would write code for their execution.
8. *Oracle Agent* (OA) – this agent is responsible for managing all the knowledge databases in the system.
9. *Music Player Agent* (MPA) – the agent will play mosic from Spotify. This would entail managing playback requests, playlists, authentication with Spotify, play/pause/seek and providing confirmation.
10. *UI Agent* (UIA) – the agent serve chat UI, show past/current conversations, keep track of whether the current convesation is audio/text, stream real-time updates. Additionally, the UI will display the system's workspace, and also allow the user to view the Task Planner's work for the current query even after the response is served.
11. *Orchestrator/Supervisor Agent* (OSA) – the agent will receive the execution plan from the TPA and execute it. It will issue commands to tool agents, track long-running tasks & cancel/retry them, even escalate to human for resolving issues or requesting further information.
12. *Task Dependency Manager Agent* (TDMA) – the agent receives the task execution plan from the TPA and builds dependency graph of the subtasks that it passes to the OSA.

2. ML/Agent System Description using PEAS [https://aima.cs.berkeley.edu/4th-ed/pdfs/newchap02.pdf]

| Agent name | Performance Measure | Environment | Actuators/Outputs | Sensors/Inputs |
|---|---|---|---|---|
| **HWAs** | • Low input/output latency<br>• High audio signal-to-noise ratio (SNR)<br>• Reliability and uptime of hardware drivers | Local machine hardware layer (microphone, speaker devices, OS audio stack) | Provides raw or denoised audio streams to Voice I/O Agent; passes playback audio to speakers | Receives audio capture from microphone; playback audio data from VIOA; hardware status events |
| **VIOA** | • Word Error Rate (WER)<br>• Speech-to-text latency<br>• TTS naturalness<br>• Correct wake-word activation | Acoustic environment; streaming speech data; Text dialog environment | Transcribed text to Session Manager / TPA; synthesized speech audio to HWA; user feedback cues (beeps, visual indicators) | Audio stream from HWA; user voice commands; wake-word triggers; TTS requests from OSA/UIA |
| **SMA** | • Accurate session state | Database or in-memory session | Sends stored dialog context to | Receives text/audio |

| | | | | |
|---|---|---|---|---|
| | persistence<br>• Fast retrieval<br>• Data integrity<br>• Efficient context hand-off | store; file system logs | TPA/OSA/UIA; logs conversation metadata. | transcripts, metadata from VIOA, UIA, and results from OSA; user identifiers and timestamps |
| **TPA** | • Plan correctness and completeness<br>• Minimal number of steps<br>• Task success rate<br>• Planning time | Logical environment of available agents and tools; symbolic representation of user goal | Task execution plan (sequence or DAG of subtasks) to TDMA and OSA. | Receives parsed query and context from SMA/UIA; agent capabilities registry; system knowledge from OA |
| **WSA** | • Search recall and precision<br>• Response latency<br>• Reliability (API uptime) | Internet search engines (Google, Bing, DuckDuckGo, etc.); Web APIs | Returns structured search results, snippets, URLs, or summaries to OA or OSA | Receives search queries from TPA/OSA; optional filters and constraints |
| **FSA** | • File operation success rate<br>• Policy compliance (no unauthorized writes)<br>• Throughput<br>• Data integrity (checksum match) | Local and workspace file systems; OS permissions; sandbox container | Performs copy/move/create within workspace; produces file metadata reports; sends confirmation or error messages | Receives file operation requests from OSA/TPA; monitors workspace directory; system path events |
| **FTA** | • Conversion success rate<br>• Average processing time<br>• Output file quality/accuracy | Local tools (ImageMagick, PyPDF2) or external conversion APIs; local workspace | Generates transformed files; returns job status, logs, and output file URIs | Receives transformation tasks and parameters from OSA/TDMA; input file references; API responses |
| **OA** | • Retrieval precision@k<br>• Latency<br>• Relevance/faithfulness of responses | Vector databases, knowledge graphs, document corpora, embeddings models | Provides factual responses or context embeddings to TPA/OSA; maintains and updates specialized sub-agents or RAG indexes | Receives user or system queries; raw documents for ingestion; new data from WSA or user uploads |
| **MPA** | • Playback reliability<br>• Latency to | Spotify API, Web Playback SDK, OAuth2 services | Sends playback commands to Spotify; provides | Receives play/pause/seek |

| | | | | |
|---|---|---|---|---|
| | play/pause<br>• Correct authentication refresh<br>• User satisfaction (song match accuracy) | | playback status updates and metadata to OSA/UIA | commands from OSA/TPA/UIA; Spotify playback state events; OAuth tokens |
| **UIA** | • Response latency<br>• Usability (user satisfaction)<br>• Low error rate in command interpretation<br>• Session continuity | Web browser or mobile environment; user interface components | Displays chat/voice UI, task plans, file workspace view, progress updates | Receives user text/voice input, events from OSA/SMA, and visualization data from FSA/TPA |
| **OSA** | • Task success/failure rate<br>• Average workflow completion time<br>• Fault recovery rate<br>• Policy enforcement accuracy | Internal multi-agent environment (TPA, TDMA, tool agents); task queue system | Issues commands to operational agents (FSA, FTA, MPA, WSA, OA); sends progress and results to UIA/SMA | Receives execution plan and dependency graph from TPA/TDMA; agent status updates and error reports |
| **TDMA** | • Correctness of dependency graph<br>• Minimal inter-task latency<br>• Absence of deadlocks<br>• Scalable task scheduling | Internal planning environment; task graphs; orchestrator message queue | Builds and outputs dependency graph (DAG) of subtasks to OSA; provides progress coordination data | Receives task plans from TPA; monitors completion events from OSA and tool agents |

## 3. Main Use Cases / Scenarios

| Use case name | Brief Descriptions | Actors Involved |
|---|---|---|
| **3.1. Start Voice Session** | The *User* can activate and deactivate the system using either voice or web UI (*UIA*). | User, HWAs, VIOA, SMA |
| **3.2. Text Chat Session** | The user types a message in the chat UI, and the system processes and replies in text. | User, UIA, SMA, TPA, OSA |

| | | |
|---|---|---|
| **3.3. Ask a General Knowledge Question** | The user asks a factual or reasoning-based question (e.g., “What is quantum entanglement?”). The system uses the Oracle Agent and Web Search Agent to retrieve and synthesize an answer. | User, VIOA/UIA, TPA, WSA, OA, OSA |
| **3.4. Play music from Spotify** | The user requests to play a specific song or playlist from Spotify. The system authenticates with Spotify, starts playback, and manages the player state. | User, MPA, TPA, OSA |
| **3.5. Copy file to workspace** | The user instructs: “Copy file A from folder B to your workspace.” The File System Agent performs the copy operation respecting the safety constraints. | User, FSA, OSA |
| **3.6. Move files within workspace** | The user commands: “Move all files from A directory to B directory in your workspace.” The system validates and executes the move. | User, FSA, OSA, TDMA |
| **3.7. Convert image format** | The user requests: “Convert image.png to image.jpg.” The File Transform Agent uses an online or local API to convert the file. | User, FTA, OSA |
| **3.8. Transform PDF pages** | The user requests: “Rotate page 3 clockwise” or “Move first page to last.” The File Transform Agent manipulates the file as requested. | User, FTA, OSA, TDMA |
| **3.9. Create knowledge database from uploaded resources** | The user uploads resources (PDFs, documents, links), and the Oracle Agent indexes them to build a custom knowledge base (e.g., “AI Research Papers DB”). | User, OA, TPA, OSA |
| **3.10. Query expert knowledge base** | The user asks questions within a specific expert domain (e.g., “Based on my AI Research DB, summarize transformer architecture.”). The Oracle Agent retrieves domain-specific information. | User, OA, TPA, OSA |
| **3.11. Retrieve and summarize web information** | The assistant autonomously fetches and summarizes information from the web, e.g., “Find today’s top tech news.” | User, WSA, TPA, OSA |
| **3.12. Manage conversation sessions** | The Session Manager Agent stores the full interaction history and retrieves context from past sessions for continuity. | SMA, UIA, User |
| **3.13. Display live task planning visualization** | The system shows the Task Planner’s execution plan in the UI in real-time, even as the response is generated. | UIA, TPA, OSA |
| **3.14. Execute compound** | The user issues a compound command (e.g., “Search for all AI conference papers from 2024, summarize them, and save summaries to workspace”). The | OSA, TPA, TDMA, Tool Agents |

| | | |
|---|---|---|
| **multi-agent command** | system decomposes it into subtasks and executes via multiple agents. | |
| **3.15. Handle task failure or clarification** | If a tool/API call fails, the Orchestrator retries, requests clarification from the user, or escalates the issue. | OSA, User, TPA |
| **3.16. End user session** | The user ends the interaction. The session is closed and stored for future retrieval. | User, SMA, UIA |

| 4. API Resources (REST/SSE/WebSocket Backend) | | |
|---|---|---|
| **View name** | **Brief Descriptions** | **URI** |
| **1.1. Voice Input Stream API** | Accepts live audio input streams from microphone; forwards to Speech-to-Text (STT) service. | */api/v1/audio/stream* |
| **1.2. Speech Recognition API** | Converts audio → text using Whisper or Google STT. Returns transcription & confidence. | */api/v1/audio/transcribe* |
| **1.3. Text-to-Speech API** | Converts assistant responses to speech; returns stream for playback. | */api/v1/audio/speak* |
| **1.4. Session Management API** | Stores, retrieves, and lists past user sessions; supports pagination & metadata filtering. | */api/v1/session/{id}* |
| **1.5. Conversation Memory API** | Maintains context for current session (e.g., recent queries, agent decisions). | */internal/session/context* |
| **1.6. Task Planning API** | Receives user intent (natural language or structured); returns execution plan and dependency graph. | */internal/task/plan* |
| **1.7. Task Execution API** | Takes execution plan and orchestrates tool agents via OSA; tracks progress & errors. | */internal/task/execute* |
| **1.8. Web Search API** | Performs web queries (Google, DuckDuckGo) and returns summarized results (via LLM summarizer). | */api/v1/search?q={query}* |
| **1.9. File Access API** | Lists directories, reads metadata, and enforces restricted workspace policy. | */internal/filesystem/access* |
| **1.10. File Transfer API** | Copies/moves files between user space and workspace under policy constraints. | */api/v1/files/transfer* |

| | | |
|---|---|---|
| **1.11. File Transformation API** | Executes file format conversions (e.g., PNG→JPEG) and PDF manipulations using external APIs. | */api/v1/files/transform* |
| **1.12. Knowledge Base API** | Creates and manages user-provided knowledge databases; exposes semantic search interface. | */api/v1/knowledge/{db_name}* |
| **1.13. Query Oracle API** | Answers factual or domain-specific questions using corresponding expert sub-agent. | */internal/oracle/query* |
| **1.14. Music Player API** | Handles Spotify playback, playlists, and authentication flow. | */api/v1/music/{command}* |
| **1.15. UI State API** | Provides front-end with real-time updates of system status, task graph, and workspace view. | */api/v1/ui/state* |
| **1.16. Agent Metrics API** | Reports each agent's health, uptime, task success rates, and latency. | */internal/monitor/metrics* |
| **1.17. Error/Recovery API** | Handles retries, user clarifications, and escalation of failed tasks. | */internal/error/resolve* |
| **1.18. Auth API** | Manages tokens for Spotify, online APIs, and user sessions. | */api/v1/auth/{service}* |
| **1.19. Orchestrator Control API** | Controls orchestration lifecycle — start, pause, cancel running multi-agent tasks. | */internal/orchestrator/control* |
| **1.20. Shutdown/Restart API** | Safe system shutdown, agent restart, and state persistence. | */internal/system/shutdown* |