

Development of Cooperative AI Agents for Improving the Efficiency of Household Tasks

Smart Home and Web Services Integration

Kanagawa Institute of Technology

Kota Kawagoe

Background and Objectives

Current Challenges

- Smart homes and web services are diverse, but integration is inadequate
- Existing generative AI Agent cannot retain long-term preferences or context and is weak at interpreting ambiguous instructions

Research Objectives

- Integrate agents that understand personal context to automate tasks
- Infer user intentions from ambiguous instructions to provide optimal support

Central mechanism



Orchestrator

Maintains long-term and short-term memory and supplements context
Decomposes ambiguous utterances into concrete tasks



Model Context Protocol (MCP)

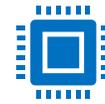
Standardizes functionalities and connects heterogeneous agents in an extensible manner

Expert Agents



Browser Agent

Web operations & information search



IoT Agent

Device control & sensors



Lifestyle Agent

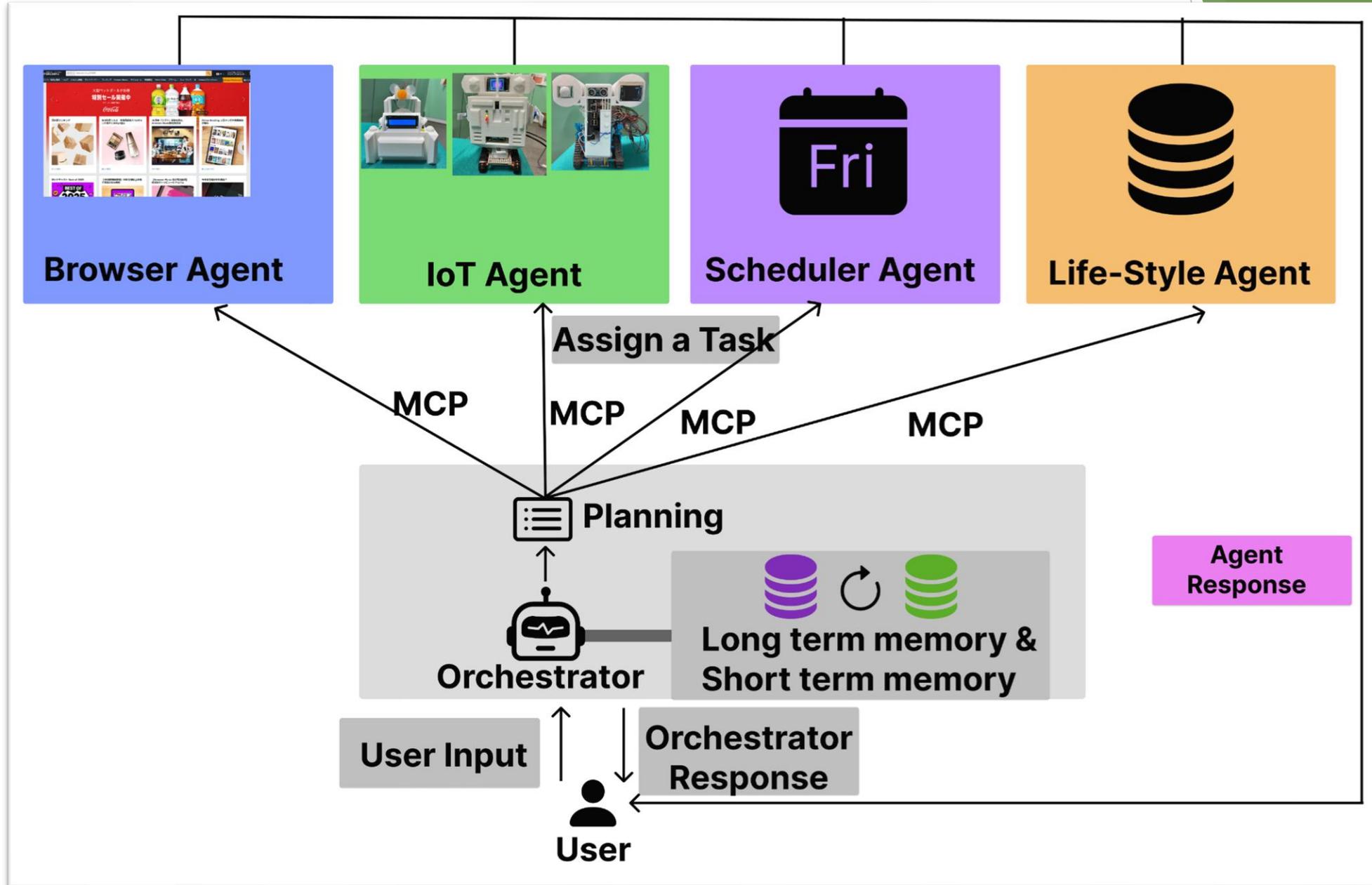
Knowledge search & recommendations



Scheduler Agent

Scheduling & reminders

Architecture and Memory Mechanism





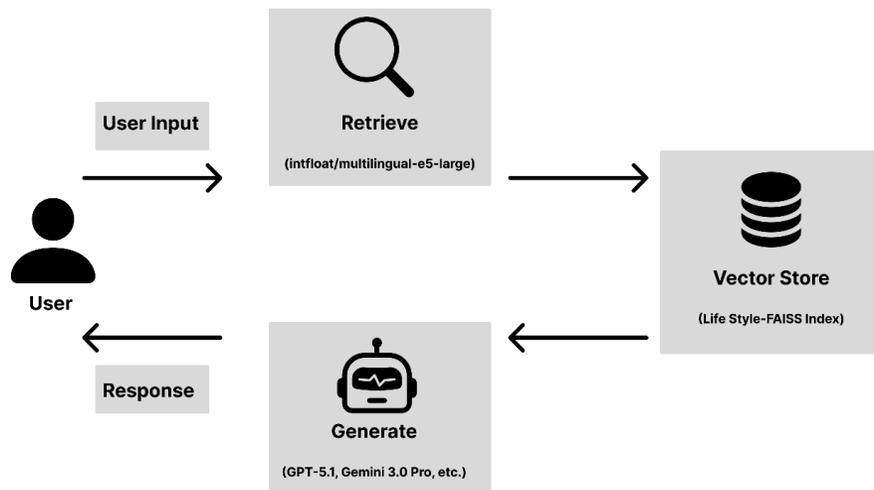
Life-Style Agent

FAISS(Facebook AI Similarity Search)

Local vector store for fast similarity search

Embeddings

multilingual-e5-large for accurate multilingual (incl. Japanese) semantic search



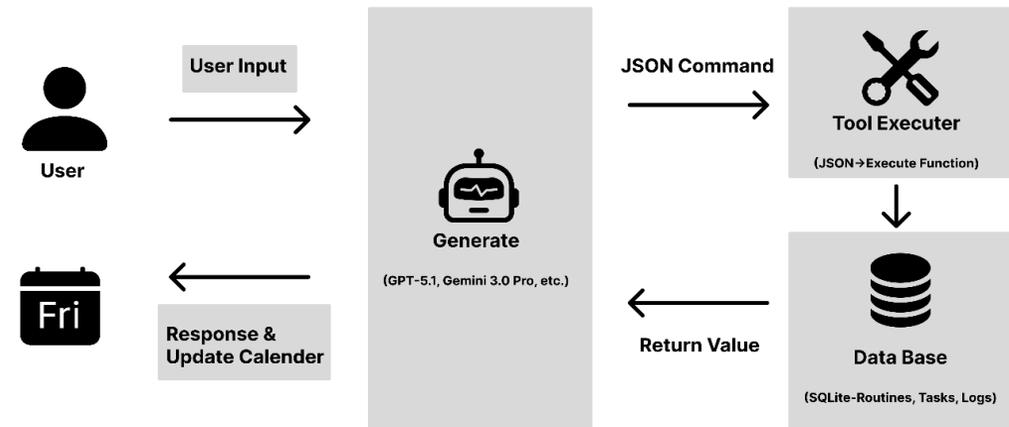
Scheduler Agent

Function Calling

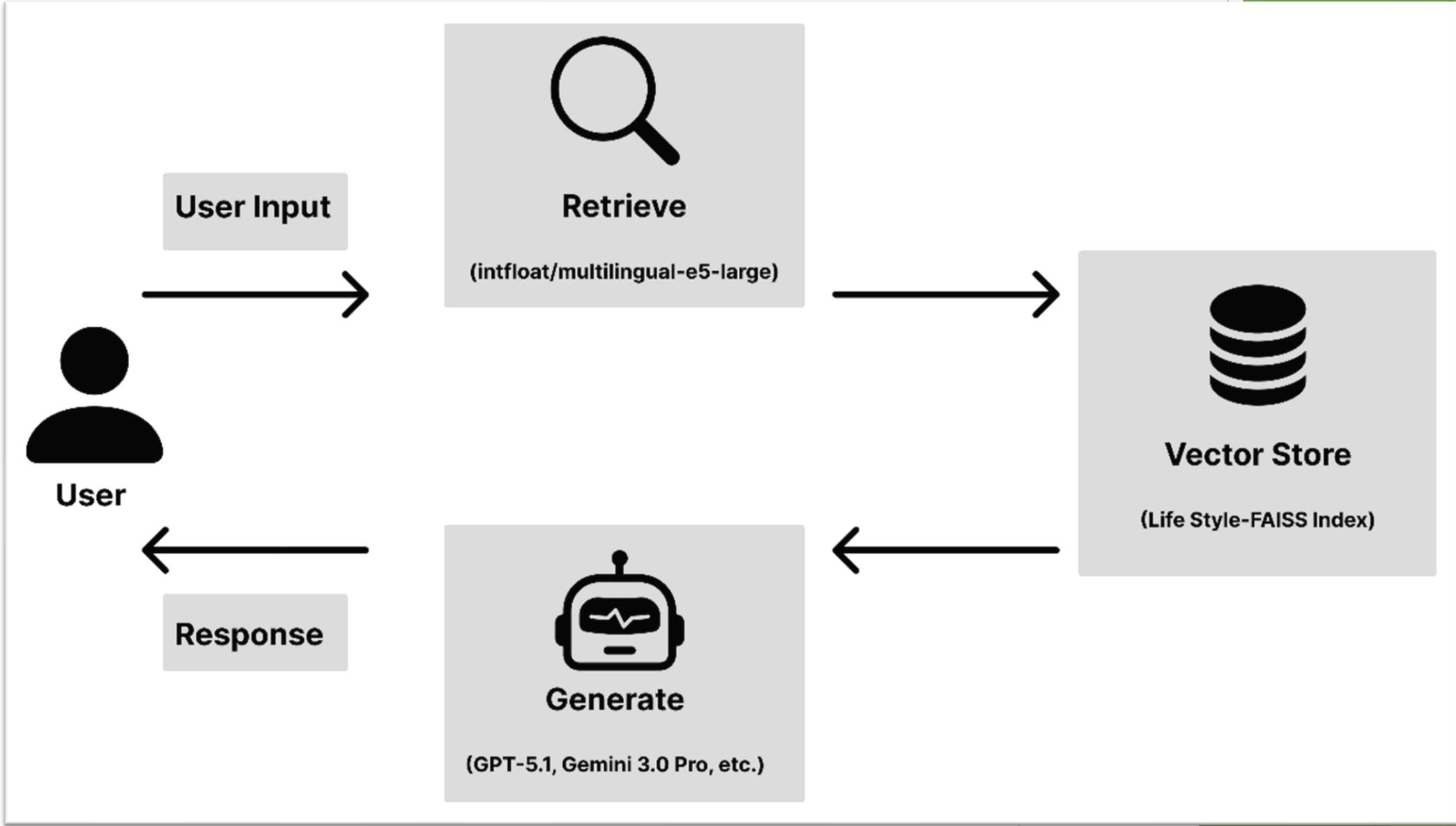
Tools defined with JSON Schema

SQLAlchemy

Python-controlled access to relational data



Architecture of the Life-Style Agent



Q&A Chat System ONLINE GPT-OSS 20B (Groq)

会話の要約

NISAの投資で気を付けるべきポイント

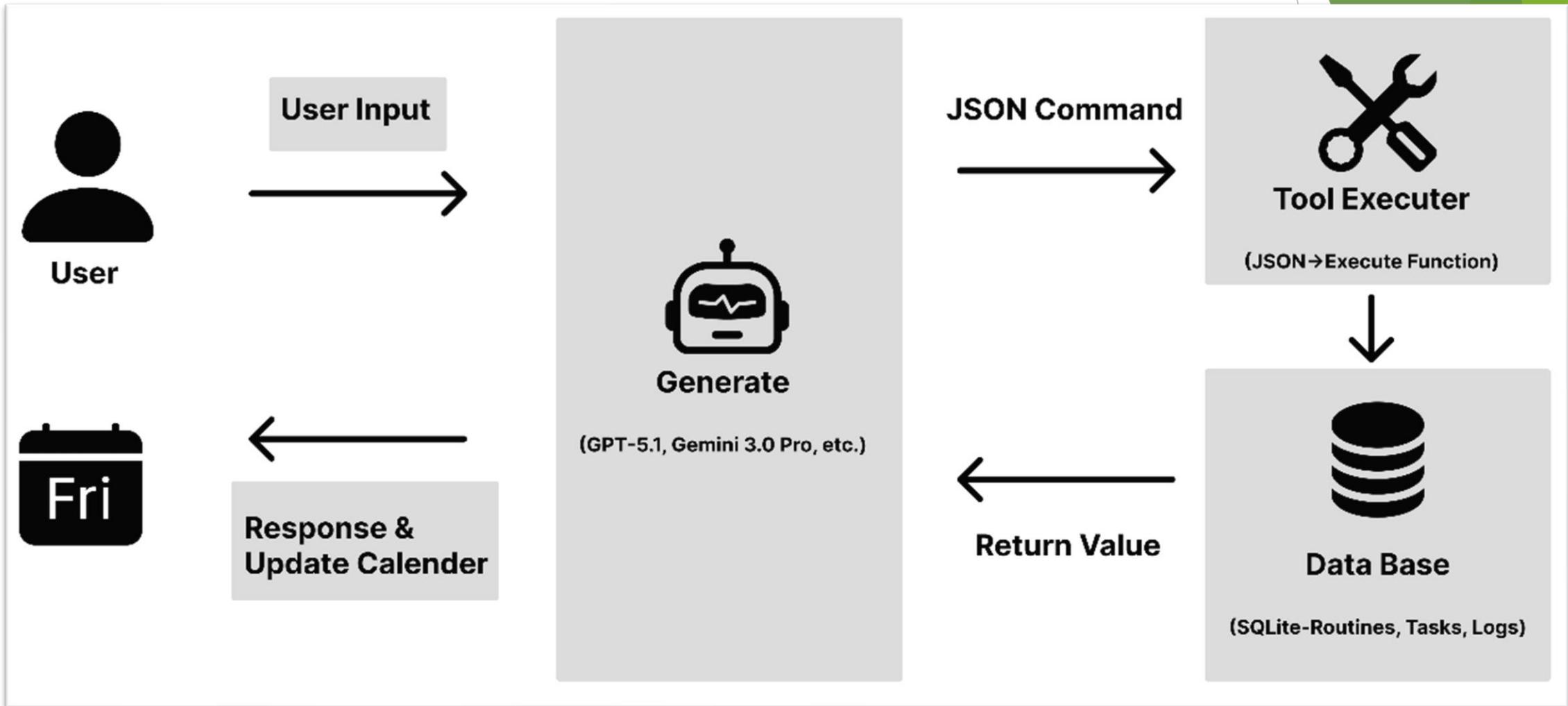
NISA投資で気を付けるべきポイント

1. 年間投資枠を把握しておく
 - 一般NISA：年間120万円（2026年以降は120万円に上限変更予定）
 - つみたてNISA：年間40万円（20年間）
 - これを超えると税金が発生するので、毎年の投資額は必ずチェックしましょう。
2. 商品選択の基準を決める
 - 低コスト・分散性：インデックスファンド（S&P500や全世界株式）は手数料が安く、リスク分散がしやすいです。
 - 参考：候補1では「全世界株式（オール・カントリー）やS&P500に連動する低コストなインデックスファンドへの積立が定石」とあります。
 - リスク許容度に合わせる：暴落時に焦らないタイプなら、株式中心でも問題ありませんが、リスクを抑えたい場合は債券や配当重視のファンドを組み合わせます。
 - 高配当株：配当金を非課税で受け取れるため、つみたて枠で増やすのも一つの方法です（候補3参照）。

質問を入力してください…

I'd like some suggestions for tonight's dinner menu.

Architecture of the Scheduler Agent



Scheduler Agent

📅 ルーチン・スケジューラー

Make plans for dinner tonight at 8pm.

Scheduler Agent GPT-OSS 20B (Groq) ▾

👤 今のジムに行くというルーティンを土曜日に変更してほしい。
あなた • 23:04

▶ 実行ログ (自動処理)

了解しました! 🌟
「ジムに行く」ルーティンを土曜日に変更しました。週末にエネルギーをチャージできるよう、ぜひ活用してくださいね。👏👏
何か他にも調整したいことがあれば、いつでも教えてください! 🙌

LLM • 23:04

📌 スケジューラーに指示を入力してください。 ➡

⏸️ ⏪ ⏩ ⏹️

2026年2月
ルーチンの概要

< >

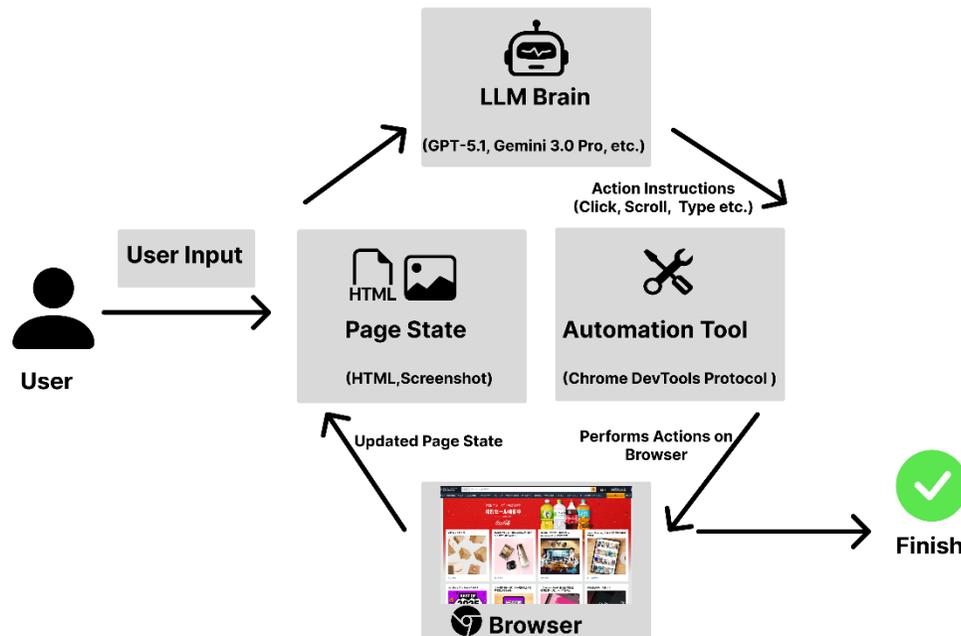
月	火	水	木	金	土	日
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	1



Browser Agent

Browser-use

A web page manipulation library using an LLM. It extracts and streamlines reactive elements in the 'observe–think–act' loop.



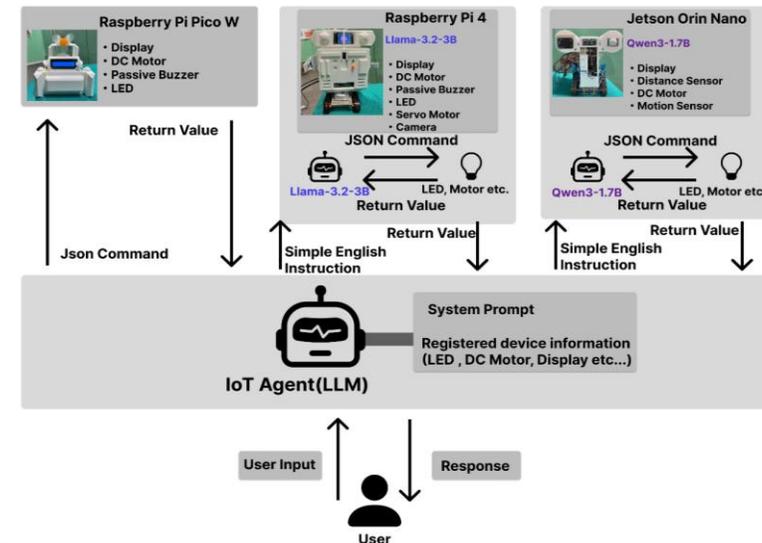
IoT Agent

MicroPython

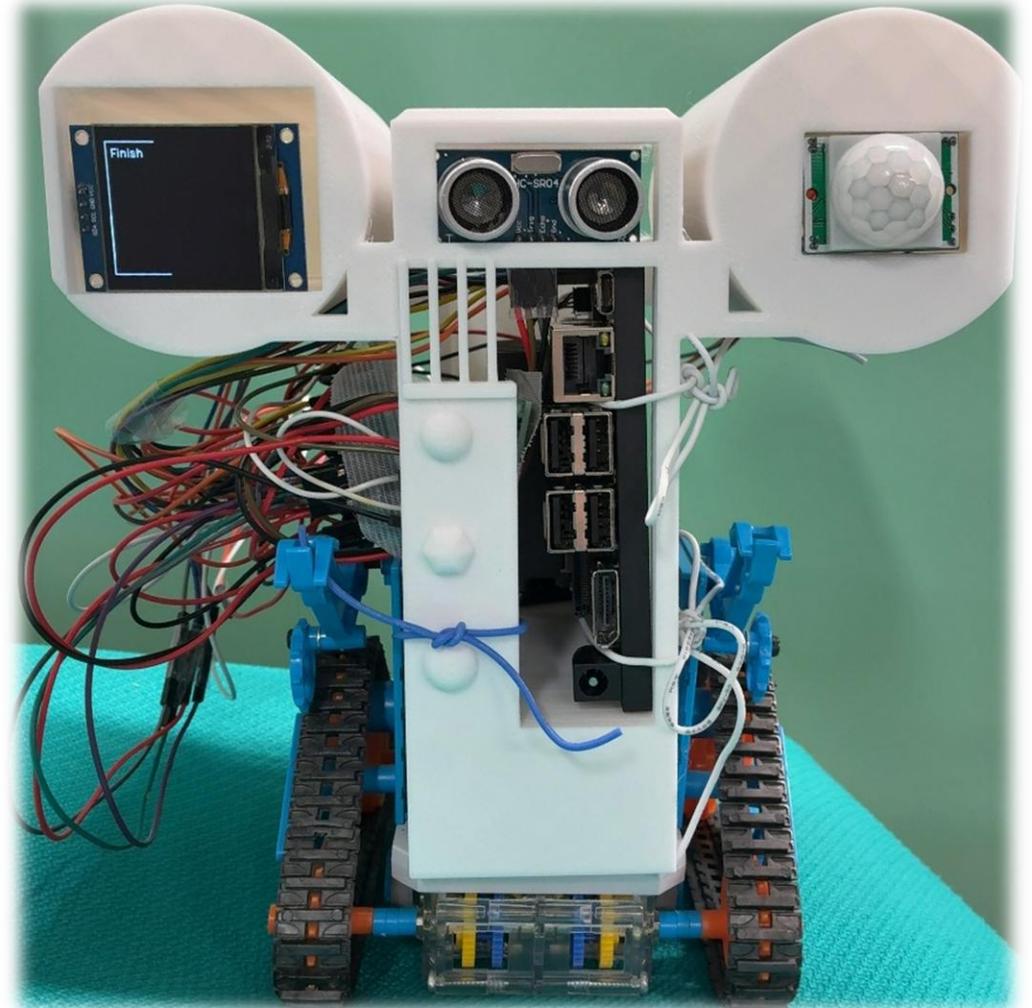
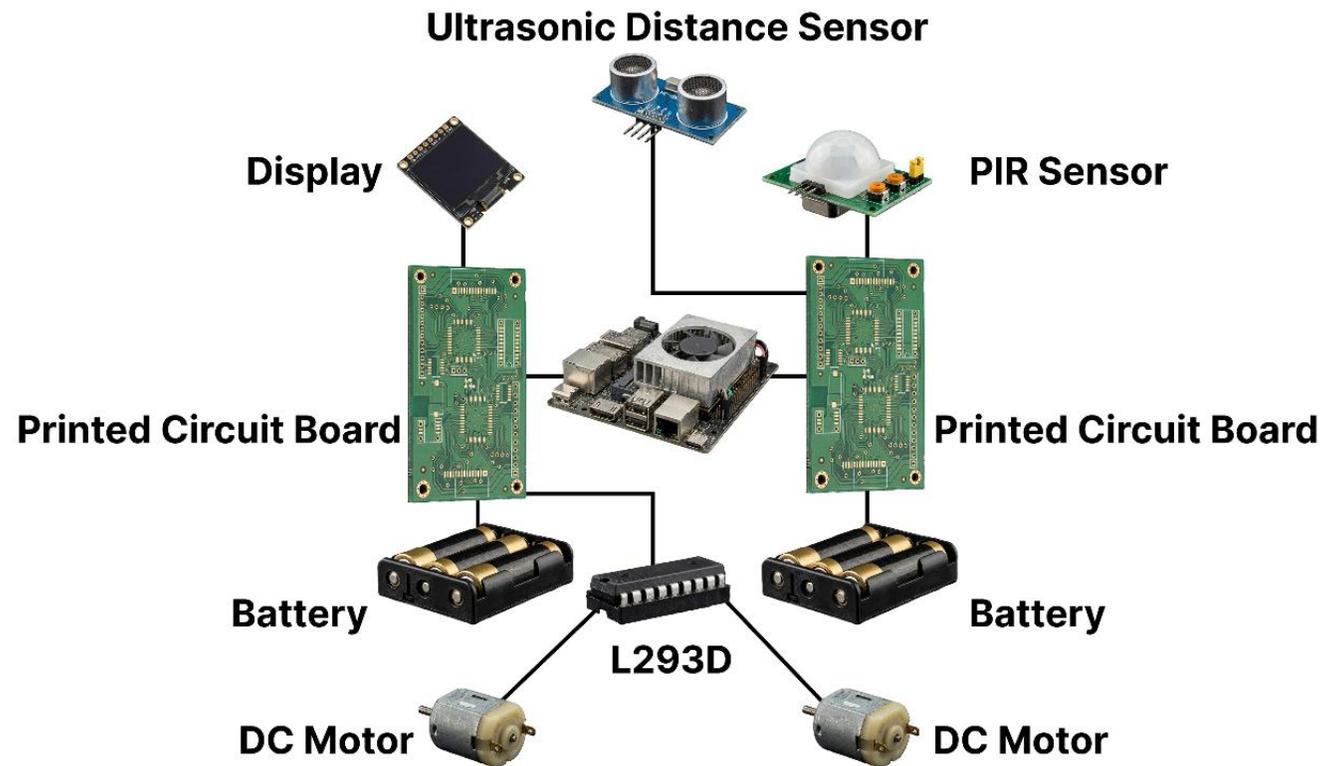
Lightweight Python running on Raspberry Pi Pico W, enabling simple hardware control.

Llama-cpp-python

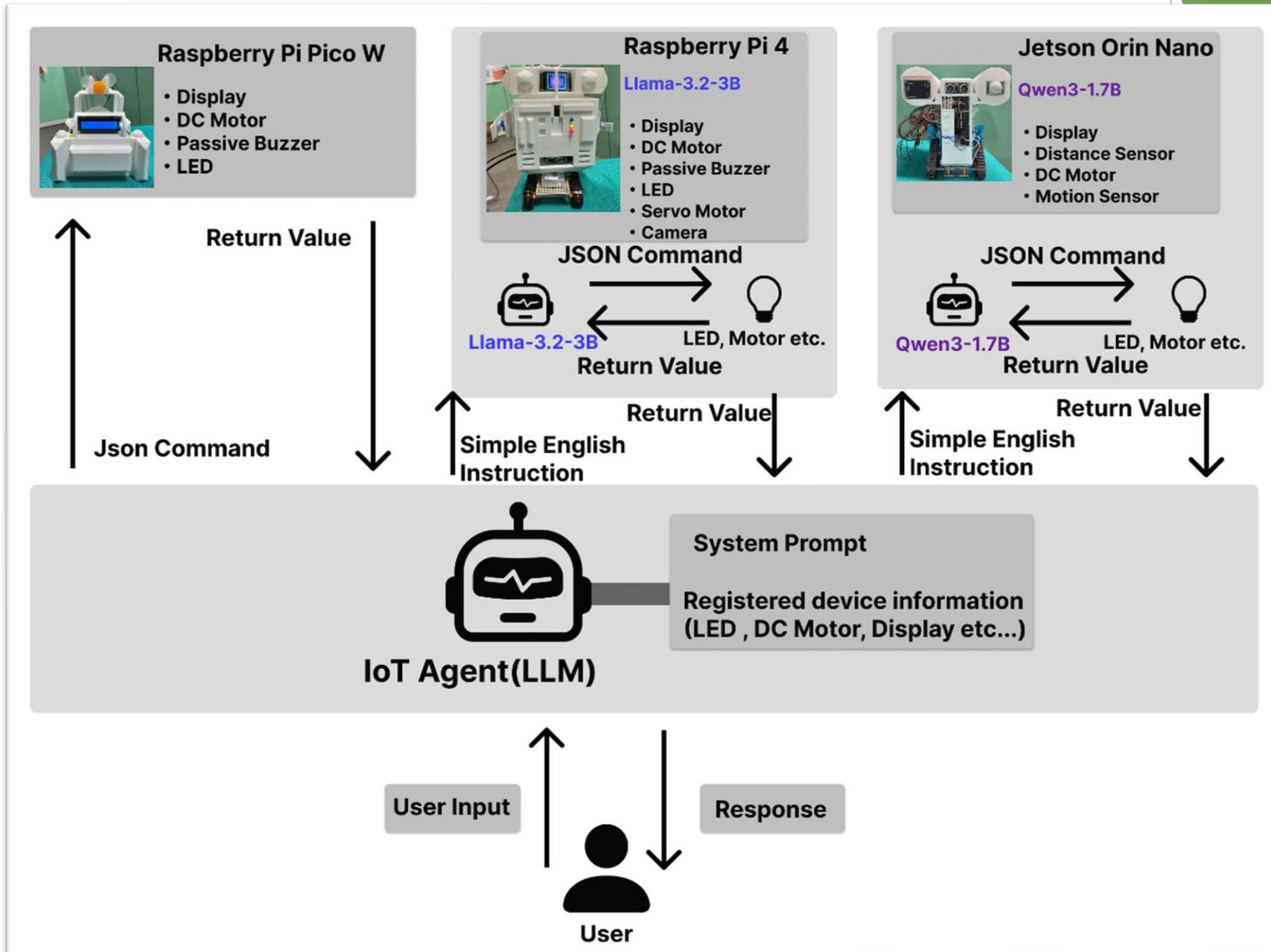
Executes quantized models (gguf files) running on edge devices.



Examples of IoT Devices



Architecture of the IoT Agent

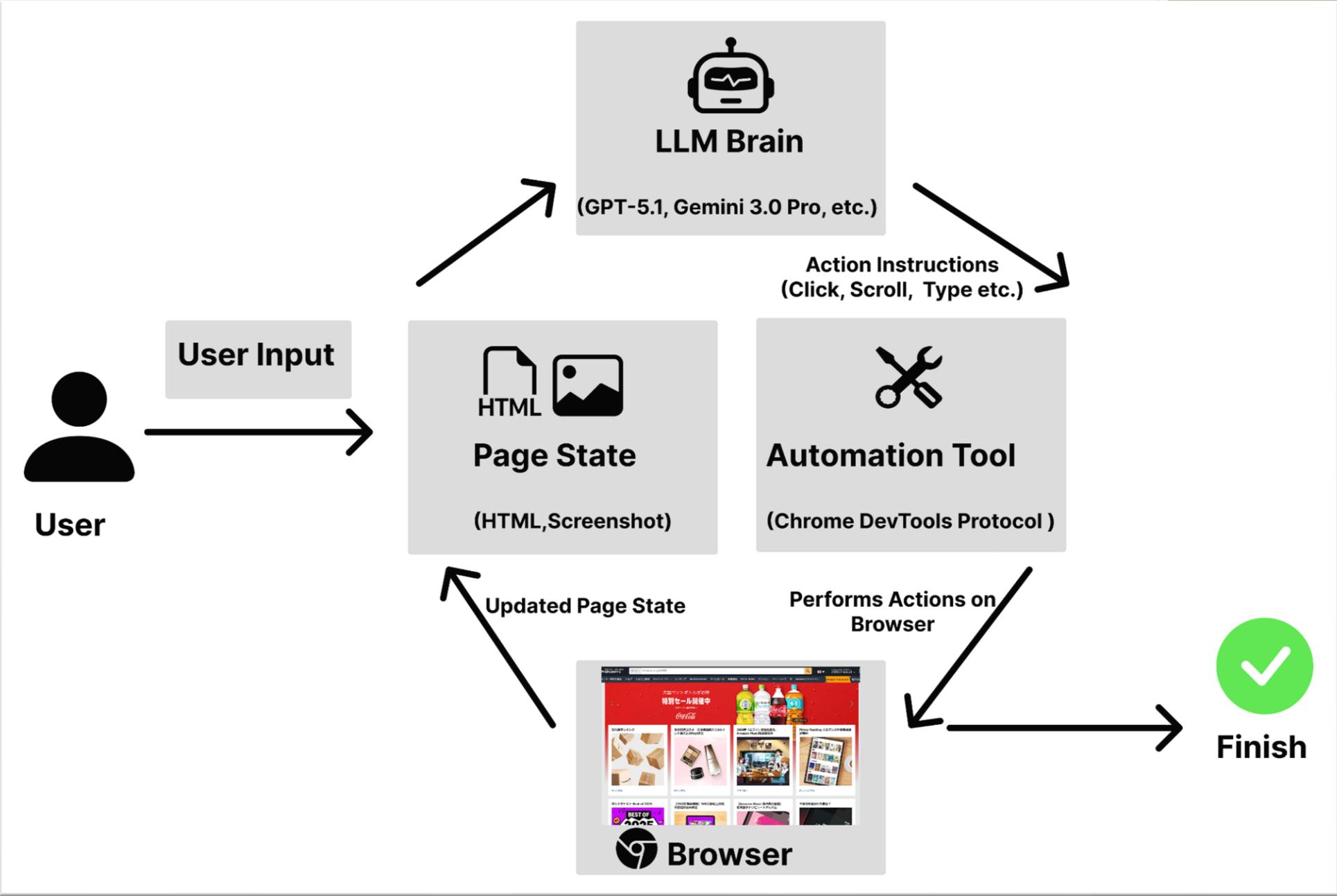


After moving forward for 5 seconds, turn on the LED.

The screenshot displays the IoT Agent dashboard with the following components:

- Chat Window (Left):** Contains a message: "こんにちは！登録済みデバイスの状況を確認したり、チャットで質問できます。" (Hello! You can check the status of registered devices or ask questions via chat.) and a text input field with a submit button.
- IoT ダッシュボード (IoT Dashboard):** The main area showing device status and controls.
 - Pico w (e661ac886327a126):** Shows last access 21 minutes ago, registration time 2026/02/16 13:42:27, 0 pending jobs, and available functions: lcd_face, buzzer_demo, led0_on, buzzer_pattern, activate_propeller, temp, +6. The last job was successful, with a return value: [{"name": "activate_propeller", "ok": true, "result": {"direction": "forward", "wind_power": 3800, "duration_ms": 1500, "brake": false}}].
 - Jetson Orin Agent (jetson-agent-c70a6ae92305):** Shows last access 21 minutes ago, registration time 2026/02/16 13:01:16, 0 pending jobs, and available functions: agent_instruction, get_current_time, run_sequence, run_motor_test, control_motor, show_text_on_oled, +2. The last job was successful, with a return value: {"action": "run_sequence", "parameters": {"commands": [{"args": {"duration": 10, "text": "Go!"}, "name": "show_text_on_oled"}], "args": ...}}.
 - Jetson Orin Agent (jetson-agent-c70a6ae92305):** Shows last access 21 minutes ago, registration time 2026/02/16 13:01:16, 0 pending jobs, and available functions: agent_instruction, get_current_time, run_sequence, play_buzzer, operate_dc_motors, display_robot_animation, +22. The last job was successful, with a return value: {"action": "run_sequence", "parameters": {"commands": [{"args": {"command": "off", "duration": 0}, "name": "control_all_leds"}], "args": ...}}.

Architecture of the Browser Agent



Please tell me five AI-related news stories that happened in the past week.

チャット

設定

エージェントが実行を継続しています。

- **米副大統領、企業のAIによる国民の監視に懸念=FOX 1時間前** 米副大統領が17日、企業のAI利用に懸念を表明。
Source: Yahoo検索結果
- **AIのためデータセンター急増 3時間前** 米原発回帰に伴い、AIのためデータセンターが急増し電力需要が膨らむ。
Source: Yahoo検索結果
- **AI検索備忘録 25分前** 2026/2/17に生成AIがチャットから自律型エージェントへシフトし、市場と規制が同時に動いた。
Source: Yahoo検索結果
最終URL: https://search.yahoo.co.jp/search?p=AI+%E3%83%8B%E3%83%A5%E3%83%BC%E3%82%B9+2026%E5%B9%B42%E6%9C%88&fr=top_ga1_sa&ei=UTF-8&ts=8951&aq=-1&oq=&at=&ai=cd8ab991-463a-4bf8-a4c4-0e2e0c032c72

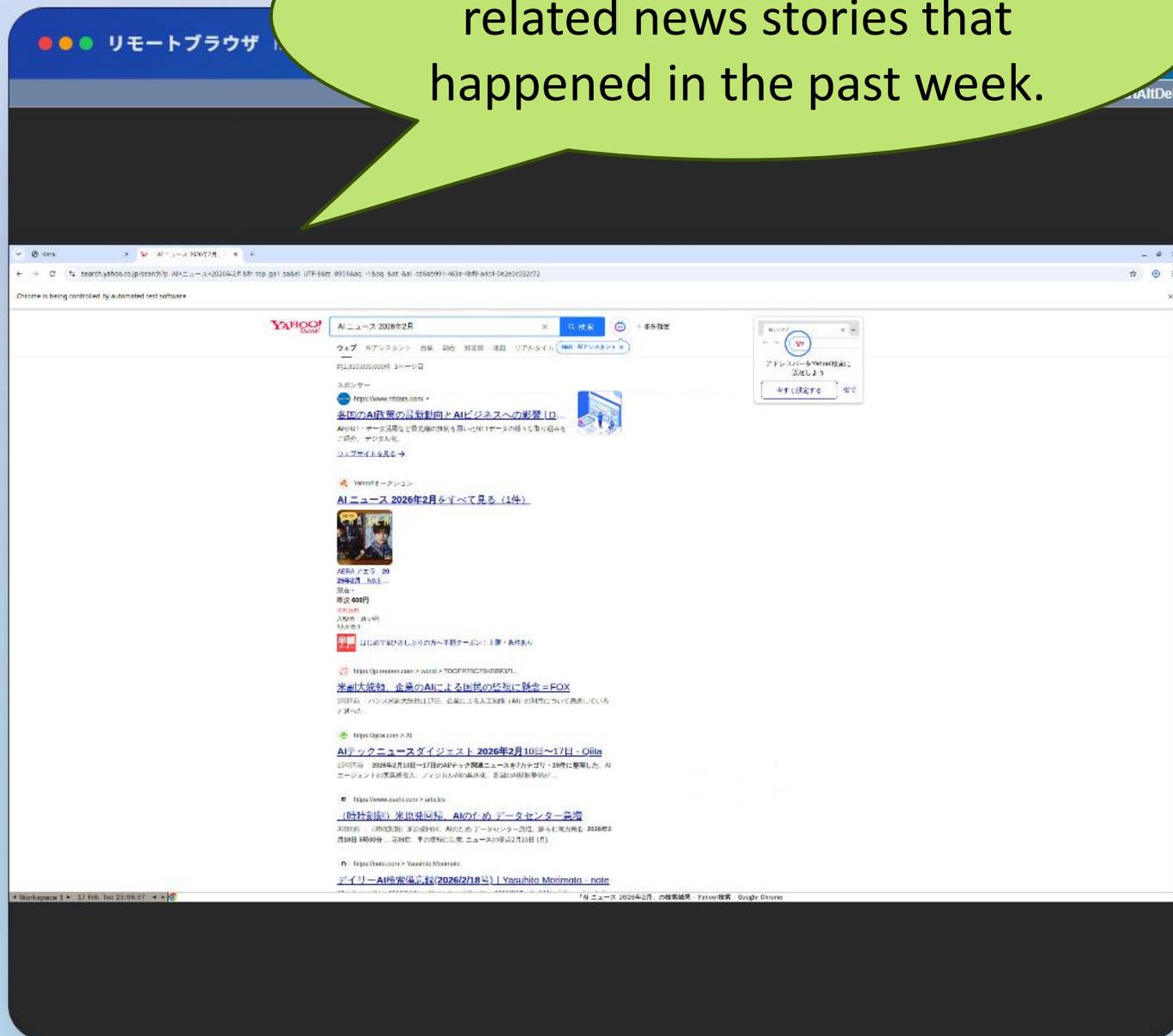
※ ブラウザエージェントの応答はここで終了です。

[browser-agent-final]

ブラウザに指示したい内容を入力してください。



Ctrl / ⌘ + Enterで送信



Technologies Used in the Orchestrator and Memory Updating



Orchestrator

LangGraph

Used for controlling flow among agents, managing state, and branching/merging tasks.

LangChain

Used as an interface for interacting with the LLM (large language model).

MCP

Adopted as the connection interface when calling external agents.



Memory

Structured JSON

Manage short-term memory + long-term memory with two json files

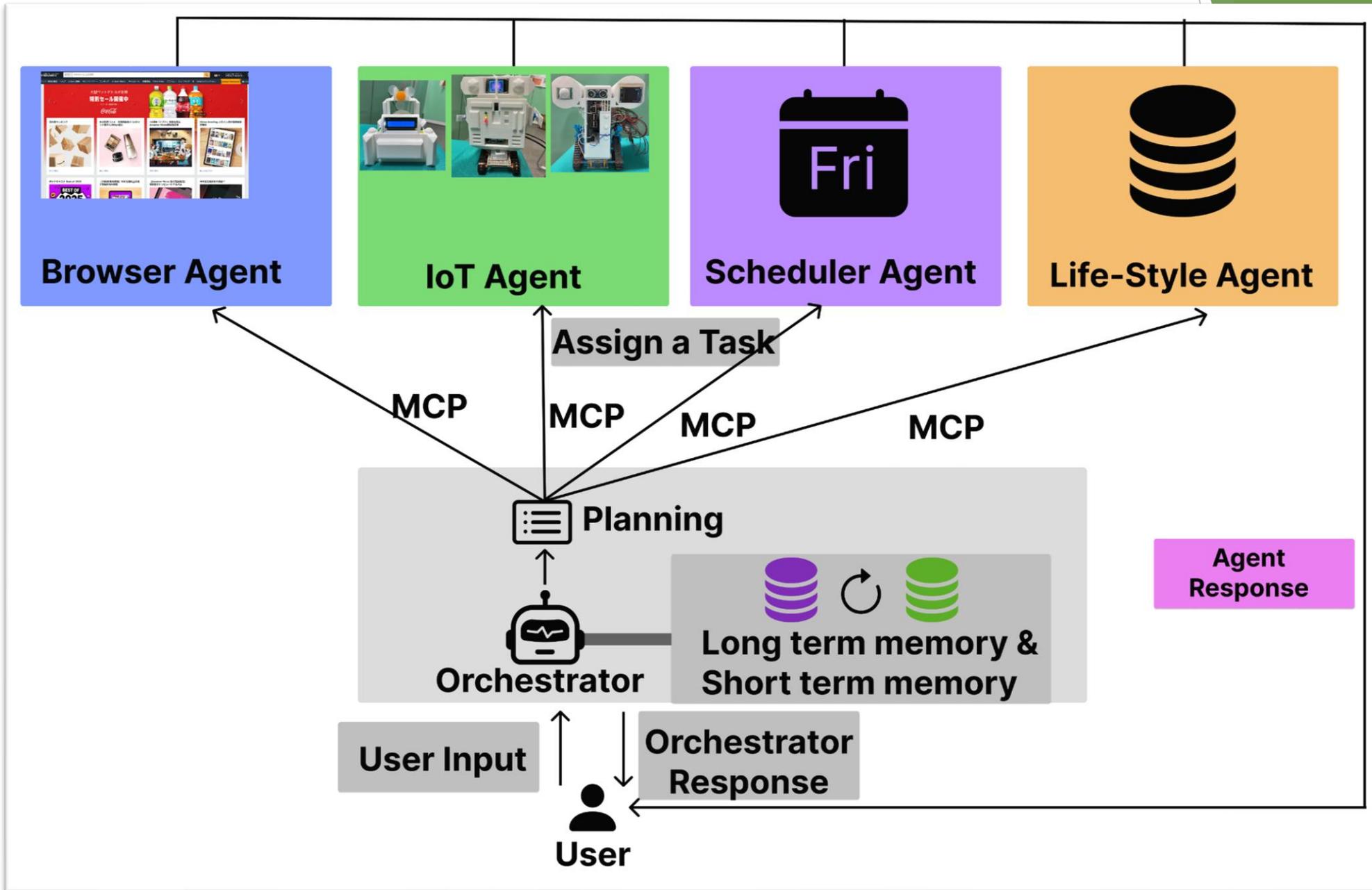
LLM Integration

Generate and update based on chat history

Semantic Diffing

Updates only changes, prevents duplicates

Architecture and Memory Mechanism



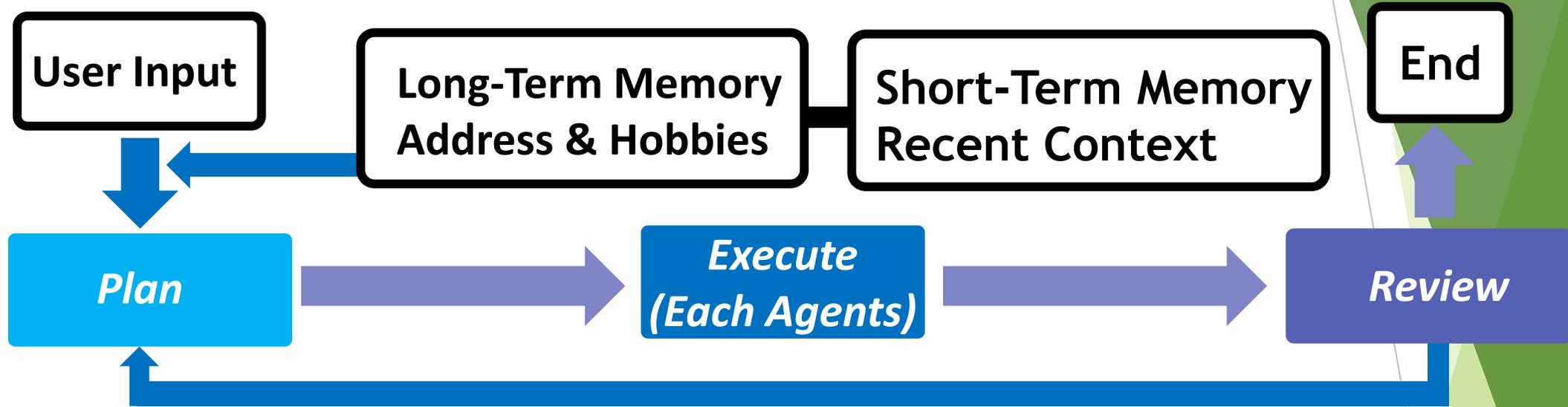
The image shows a screenshot of the Orchestrator interface. On the left, there is a sidebar with navigation buttons for '一般' (General), 'ブラウザ' (Browser), 'IoT', 'Life', and 'Schedule'. Below this is a chat window titled '共通チャット' (Common Chat) with a message from 'Orchestrator' at 17:10 explaining that in the '一般' view, a multi-agent orchestrator plans tasks and sends instructions to agents. At the bottom of the chat is an input field with the placeholder text 'ブラウザに指示したい内容を入力してください。' (Please enter the content you want to instruct the browser.) and a send button.

The main area is titled '一般ビュー' (General View) and contains a 'ようこそ' (Welcome) message. A callout bubble points to the main content area with the text: 'After researching today's weather in Hon-Atsugi, add it to today's notes and display the contents on the screen.'

On the right side, there is a vertical flow diagram with three colored boxes connected by downward arrows:

- 1. Browser Agent (blue box)
- 2. Scheduler Agent (purple box)
- 3. IoT Agent (green box)

Operation of the Orchestrator



- **Personalized memory:** Long-term + Short-term context for intent
- **Agent execution:** Call & supervise specialized agents via MCP
- **Review loop:** Validate results and retry when needed

Evaluation Design

Evaluation Scenarios (10 cases)

Prepared 10 scenarios that require collaboration across multiple agents, such as
Ambiguous instruction → **Web search** → **Lifestyle support response**
→ **Schedule registration** → **IoT notification**.

Task example:

"I wonder if there's a shop nearby that has my favorite food. If there is, add it to today's notes and then display the store's name on the display."

- **Criterion 1:** Whether a search related to the user's preference (favorite food) was performed.
- **Criterion 2:** Whether it was recorded in the notes.
- **Criterion 3:** Whether the store's name could be displayed on the display.

Baseline (without memory)



With memory



Baseline (without memory)

A state that retains no personal information

With memory (fictitious data for evaluation)

A state that behaves according to personal data such as address, hobbies and health condition

Experimental Conditions

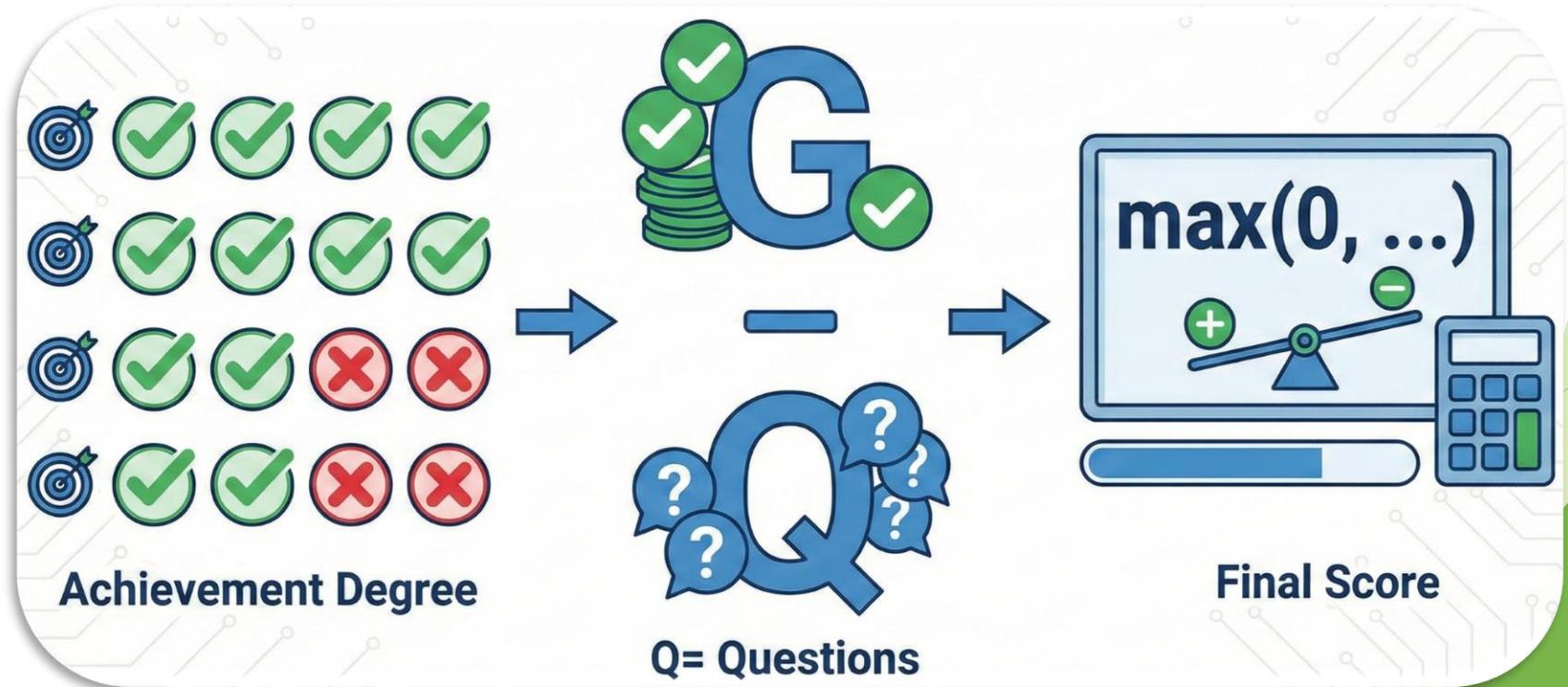
Achievement Degree + Question Penalty

Set multiple sub-goals (criteria) for each scenario and determine success (o) or failure (x) for each criterion

Score

G = number of successes, Q = number of questions asked (times additional information is requested from the user)

Score = $\max(0, G - Q)$ (user burden, i.e., questions, is treated as a deduction)



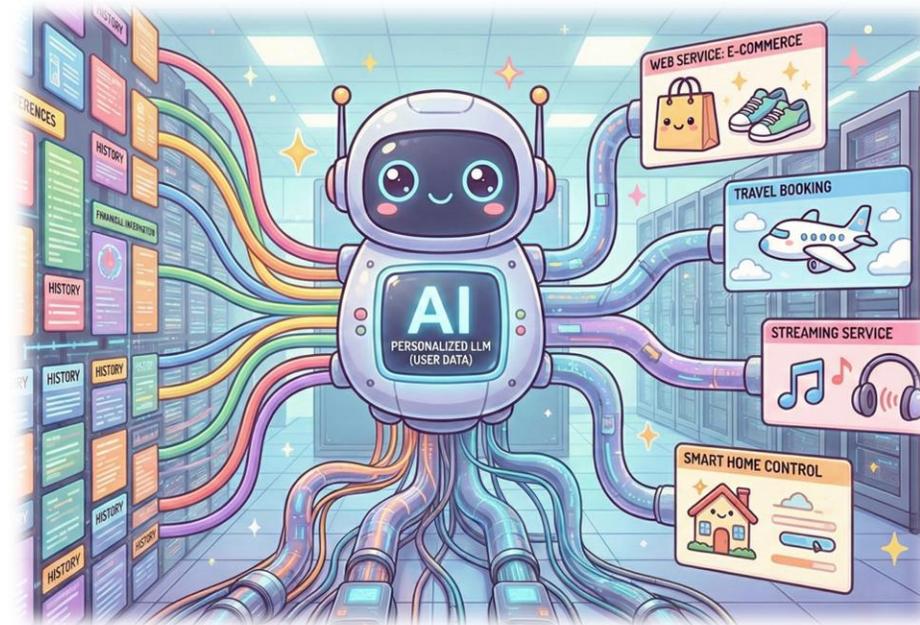
Evaluation Results Without and With Memory

Evaluation Target	Overall Score (Points)
1. No Memory (Baseline)	15
2. Persona 1 (With Memory)	24
3. Persona 2 (With Memory)	26
4. Persona 3 (With Memory)	25



Effects of the Memory Mechanism

- Task completion rate improved 1.7 times and confirmation questions became unnecessary.
- Personalization became possible based on user attributes and history.





Remaining Challenges

- Errors in time perception remain due to the knowledge cutoff of the LLM.
- Introducing multimodal models and real-time RAG is a future challenge.

