



 [Pobierz treść tej lekcji w formacie markdown](#)

 [Pobierz treść tej lekcji w formacie markdown \(EN\)](#)

## Słowem wstępu

Każdy, kto uczestniczył w poprzednich edycjach AI\_devs lub obserwuje mnie na X, Instagramie czy LinkedInie, wie, że "Alice", która pojawia się w treściach lekcji, ma również swoją drugą wersję, dostępną wyłącznie dla mnie. Wynika to z faktu, że poziom personalizacji tego projektu jest tak duży, że trudno było mi go udostępnić...

...do dziś.

Dotychczasowa podróż w AI\_devs doprowadziła nas do miejsca w którym dość zrozumiałe stały się kluczowe aspekty związane z tworzeniem narzędzi, pracą z danymi, różnymi formatami plików czy rozwijania logiki agentów. Teraz dobrze byłoby na to spojrzeć zarówno z szerokiej perspektywy nieco większego projektu, ale też detali.

Istotną zmienną jest tutaj także fakt, że dzięki współpracy z Cursor, nawet jeśli pracujesz w innej technologii lub na co dzień skupiasz się na innych obszarach niż aplikacje backendowe, to i tak będziesz w stanie szczegółowo zapoznać się z moim kodem. Możliwość tą potwierdziły ostatnie tygodnie oraz osoby pracujące w Pythonie, .NET czy po prostu w obszarze front-endu lub innych specjalizacji.

Zatem w ostatniej lekcji AI\_devs 3 skupimy się na omówieniu systemu, który **łączy wszystkie dotychczasowe koncepcje** i który w założeniu powinien pomóc Ci "połączyć kropki" lub przynajmniej pokazać, jak możesz stosować wiedzę z AI\_devs 3 w praktyce.

## AI\_devs 3

Lekcje AI\_devs 3 przeprowadziły nas przez **najważniejsze rzeczy, które aktualnie wiemy na temat modeli, technik interakcji z nimi poprzez API, budowie interfejsów użytkownika, strukturyzowania i przetwarzania treści** oraz **budowaniu narzędzi** z myślą o dużych modelach językowych.

Każdy z przykładów miał formę "modułu" (jest to przenośnia, bo nie spełniają one definicji modułu), który zwykle skupiał się **tylko na jednym zadaniu**. Dzięki temu stopniowo zauważylismy, że możemy łączyć je ze sobą w praktycznie dowolnej kolejności. Natomiast w ostatnich lekcjach dowiedzieliśmy się także w jaki sposób możemy je rozbudować i sprawić, by pełniły rolę agentów współpracujących między sobą.

Powiedziałem jednak, że do budowy zespołu agentów, warto dobrze opanować zbudowanie przynajmniej jednego z nich.

"Alice" w mojej prywatnej wersji to projekt, który rozwijam od grudnia 2022. Sama jego forma stanowi kontynuację mojej koncepcji zespołu 'avatarów', którą rozpoczęłem kilka lat wcześniej, gdy do dyspozycji miałem tylko kod oraz narzędzia no-code.

Przez cały ten czas Alice rozwijała się. Możliwości modeli językowych wzrastały, a nowe narzędzia pojawiały się regularnie. Moje doświadczenie również się powiększało. Uznałem więc, że AI\_devs 3 to doskonała okazja, by nie tylko stworzyć kolejną wersję, ale także udostępnić kod projektu — "Alice AGI".

"AGI" oczywiście należy traktować z przymrużeniem oka i nie brać tego dosłownie, aczkolwiek obecne możliwości projektu to całkiem dobra namiastka tego, jak może wyglądać nasza codzienność za jakiś czas.

# Alice AGI



Alice AGI is an application that uses LLM (GPT-4o) to perform actions based on user queries, while using available tools and long-term memory.

## ⚠️ Important Notice

This is a personal project shared ONLY to demonstrate ideas and concepts. It is not intended to be an open-source assistant for active development or contributions. While discussions and idea exchanges are welcome in the Issues section, please note that this repository is for personal use. Feel free to explore and draw inspiration or create a fork, but **active development will remain personal**.

## Installation

1. Clone the repository
2. Install dependencies using `bun install`
3. Create `.env` file based on `.env.example` and set your own values
4. Run the application using `bun run dev` — but before that, you need to run database migrations and seeds

## Database, migrations & seeds

Alice AGI uses `SQLite` as a database with `DrizzleORM`. To create the database and run migrations, you can use the following commands:

```
bun generate  
bun migrate  
bun seed
```



## Zanim zacznijemy

Projekt "Alice AGI" udostępniam na swoim githubie, ale nie mam w planach rozwijania go w formie Open Source czy też publikowania aktualizacji z zachowaniem wstępnej kompatybilności. Ten projekt nadal będzie mieć zatem prywatny charakter. Co prawda sekcja "Issues" może posłużyć wymianie pomysłów czy sugestii, ale nic poza tym.

Naturalnie w projekcie występuje jeszcze sporo mniejszych i większych błędów, aczkolwiek z głównych funkcjonalności nowej wersji korzystam z powodzeniem od około 4 tygodni.

Nadrzędnym celem tego projektu na potrzeby tej lekcji, jest **przedstawienie tego, w jaki sposób wszystkie omówione koncepcje AI\_devs 3 można połączyć w całość**.

Jest to więc kopalnia wiedzy (przynajmniej według mnie) zarówno na temat tego, o czym mówiłem w lekcjach, ale także tego, o czym powiedzieć nie miałem okazji.

## Koszty i czas reakcji

Z jakiegoś powodu podczas rozmowy z drugą osobą na czacie, oczekивание kilku lub kilkunastu minut na odpowiedź jest dla nas w pełni akceptowalne. Jednak zupełnie inne oczekивание mamy wobec aplikacji i modeli językowych.

Dobrze więc wiedzieć, że w przypadku Alice AGI w obecnej formie, czas odpowiedzi rzadko spada poniżej jednej minuty. Wyjątkiem są zapytania klasyfikowane jako 'fast track', które nie wymagają zaangażowania logiki agenta.

### Trace latencies

Trace Name	50th	90th	95th ▼	99th
X7CeHaXiwBEB0Da7...	2m 14s	2m 14s	2m 14s	2m 14s
iw5w88RIBXzb5PSN...	2m 1s	2m 1s	2m 1s	2m 1s
klyxuXD91ImLDQos...	1m 47s	1m 47s	1m 47s	1m 47s
HNk7NbRwhkF0RhRI...	1m 44s	1m 44s	1m 44s	1m 44s
MFqMDFLSDTzKyInJ...	1m 38s	1m 38s	1m 38s	1m 38s

Podobnie jest także z kosztami, bo aktualnie sięgają one \$0.3 za jedno zapytanie. Oczywiście jest tutaj ogromny potencjał na optymalizację i przejście na modele Claude 3.5 Haiku bądź GPT-4o-mini, co także zmniejszyłoby czas reakcji w przypadku części zadań.

**TRACE ZEUf71C1nKAjvpsGWwgEs**

12/3/2024, 6:26:58 PM

1m 11s

61,640 → 2,714 ( $\Sigma$  64,354)

$\Sigma$  \$0.308844

Pretty ✨

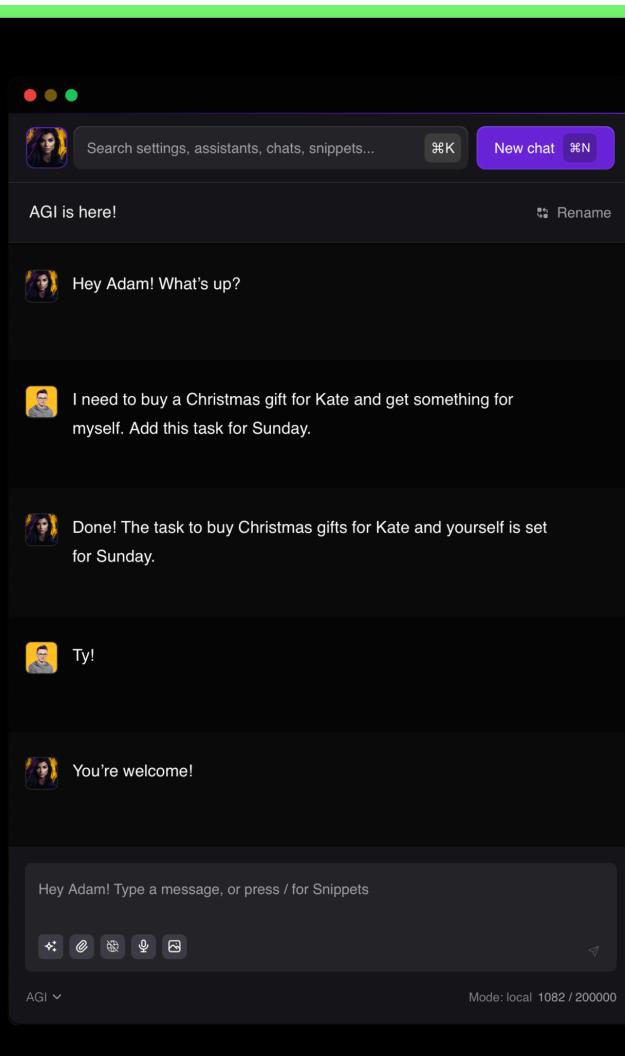
JSON

Warto więc już na wstępie mieć na uwadze fakt, że złożoność systemu przekłada się na zdecydowanie wyższy czas reakcji oraz stosunkowo wysokie koszty. Może się zatem okazać, że korzystanie z takiego rozwiązania będzie dla Ciebie nieopłacalne, albo że **warto zbudować taki, który będzie w dużej mierze opierać się o modele open source**.

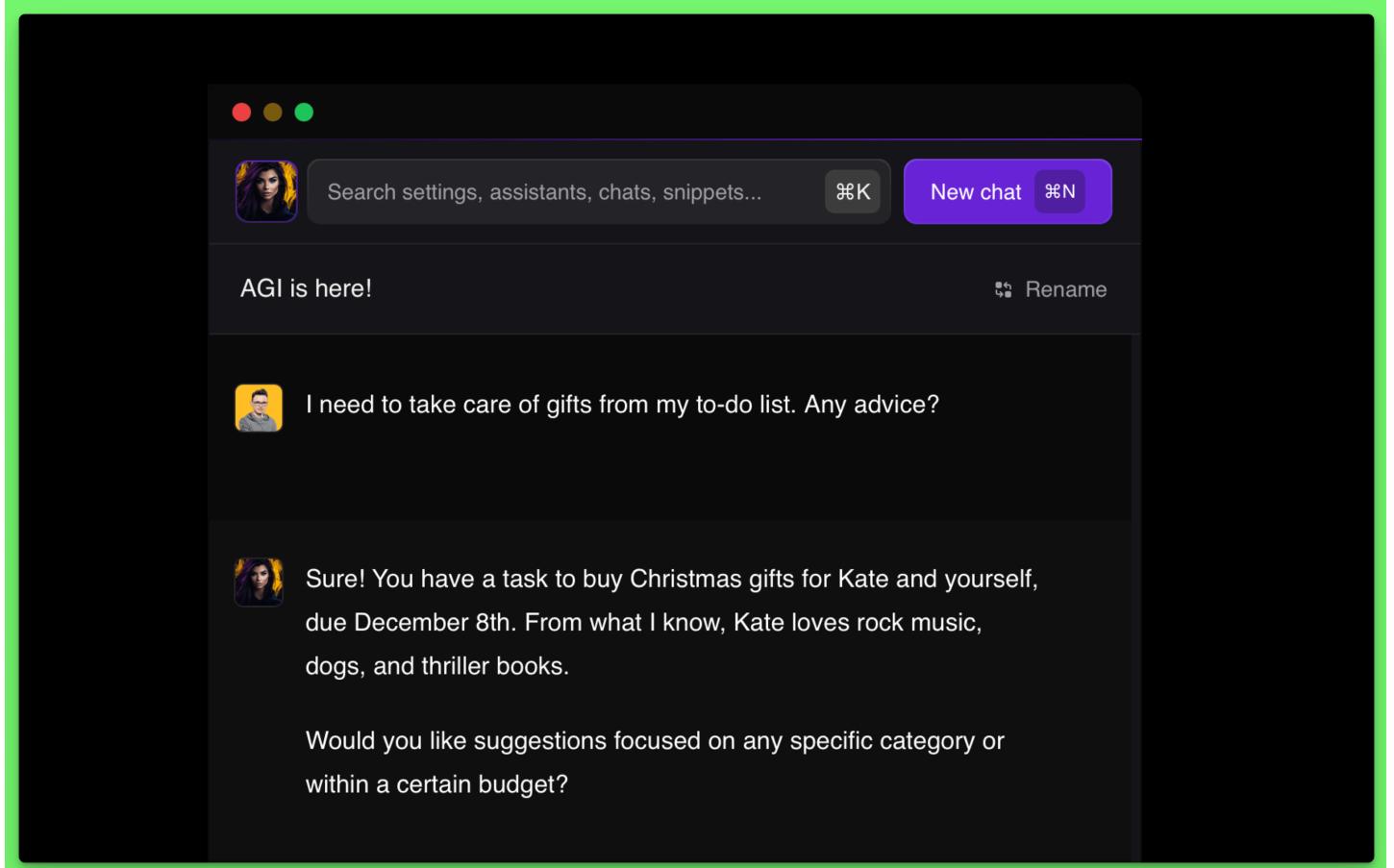
No bo nikt nie powiedział, że każdy agent musi być zbudowany dokładnie w taki sposób, a wszystkie moje założenia są poprawne.

## Główna koncepcja

Koncepcja Alice AGI jest prosta i **polega na dążeniu do możliwie swobodnej rozmowy z aplikacją, która posiada wiedzę na mój temat oraz jest w stanie posługiwać się narzędziami z którymi pracuję**. Poniżej widać przykład takiej rozmowy, w wyniku której do mojej listy zadań został dodany nowy wpis.



W przyszłości mogę wrócić do tego tematu i może mieć to miejsce **w nowej konwersacji**. Historia wcześniejszych wiadomości nie ma tutaj znaczenia, ponieważ potrzebne informacje są wczytane zarówno z mojej listy zadań w Linear, jak i pamięci długoterminowej modelu. Poniżej widzimy, że faktycznie Alice mówi o zakupie prezentów oraz przywołuje zainteresowania Kate.



Powysze pytanie uruchomiło widoczną poniżej logikę w której widzimy nawiązanie połączenia z listą zadań oraz wczytanie informacji z pamięci długoterminowej. To, co widzimy poniżej, to oczywiście [Langfuse](#), który omawialiśmy w pierwszych tygodniach AI\_devs.



Rozmowa oczywiście może być kontynuowana i poniżej widzimy odpowiedź wygenerowaną na podstawie wyszukiwania w Internecie, ponieważ domyślnie model nie posiada informacji na temat produktu "Arc Ultra".



Maybe a Sonos soundbar would be good choice for our living room? I've heard they have something new. Could you check which model is the latest one?



Yes, Sonos recently released the **Arc Ultra** soundbar in October 2024. It features their new Sound Motion™ technology, offering double the bass of its predecessor and a 9.1.4 spatial audio experience. Some highlights include:

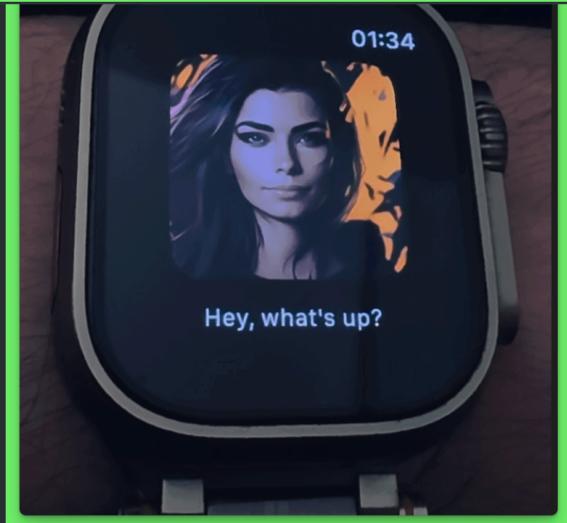
- 14 Sonos-engineered drivers
- Dolby Atmos support
- Advanced Speech Enhancement for clearer dialogue
- Sleek design that fits well with modern TVs
- Simple setup with HDMI eARC
- Bluetooth line-in capability

It's priced at \$999 USD. Want to know more about these features or compare it to older models?

Nie zawsze jednak może być to w pełni swobodna rozmowa, ponieważ wiemy, że model nie jest w stanie domyślić się wszystkiego. Poza tym, jesteśmy ograniczeni dostępnymi narzędziami, czy w ogóle możliwością interakcji z nimi z poziomu kodu — obie te rzeczy omawialiśmy w lekcjach AI\_devs 3.

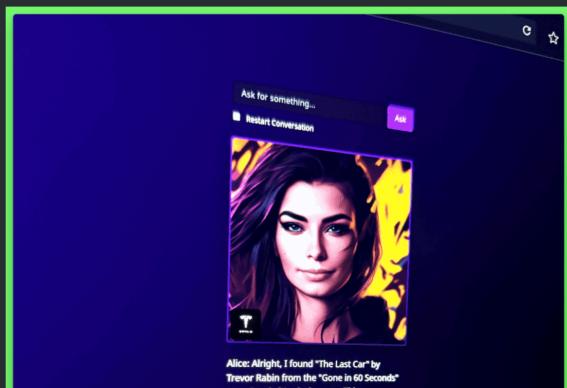
W tej chwili może pojawiać się pytanie: **dlaczego obsługiwać te narzędzia z pomocą czatu, a nie samodzielnie?**

Przede wszystkim, jedno nie wyklucza drugiego. Natomiast możliwość przywołania informacji z kilku źródeł lub dodanie kilku wpisów za pomocą jednej wiadomości jest często prostsze niż przełączanie się między narzędziami. Po drugie, możliwość przesyłania wiadomości jest dostępna także z poziomu automatyzacji czy zewnętrznych usług, o czym również wspomniałem w lekcjach. Co więcej, we wpisie "[Jak pracuję z LLM?](#)" mówiłem także o tym, że dokładnie te same możliwości są dla mnie dostępne na Slacku, w zegarku, czy w samochodzie.



Analogicznie działa w przeglądarce dostępnej w samochodzie. Oznacza to, że w trakcie jazdy mogę o czymś powiedzieć, a następnie wrócić do rozmowy, gdy wrócę do domu. Tutaj jednak największym ograniczeniem w czasie podróży jest dostępność Internetu, ponieważ wciąż mamy obszary bez zasięgu.

Natomiast ogólne funkcjonalności pozostają takie same, tylko interakcja odbywa się przez mikrofon / głośniki samochodu.



Zobaczmy więc, jak to wszystko działa.

## Zestaw narzędzi

Pomimo wzrastającej popularności frameworków, które już dziś pojawiają się w ogłoszeniach o pracę firm takich jak Amazon, Cloudflare czy Apple, zdecydowałem się na pracę z [Vercel AI SDK](#) w celu połączenia z modelami OpenAI oraz Anthropic. Natomiast poniższy zrzut ekranu wskazuje na możliwy wzrost znaczenia frameworków w przyszłości. Warto zatem monitorować ich rozwój, testować nowe wersje, a po zakończeniu AI\_devs rozważyć zapoznanie się z LangGraph, który według opinii niektórych z Was zasługuje na uwagę.

When you know you're onto something huge...

NVIDIA, Amazon, Cloudflare, PwC and Apple, what they have in common? All of them have open positions for CrewAI Engineers!

8:28 PM · Dec 3, 2024 · 20.4K Views

19 28 277 101

Post your reply Reply

**praveen** @tiwariprt · 2h Something that's based of langchain wont be production worthy imo

**João Moura** @joaomdmoura · 2h we are not :D we dropped it as a dependency months ago

**Michael Cunn** @mcunningh · 9h Still remember the "holy shit" moment I had about agents looking at AutoGPT almost 2 years ago

Show replies

Look where we are now

Pomimo tego, że jest to "jedynie" SDK, to i tak zdążyłem już spotkać ograniczenia wymagające pominięcia jej przy obsłudze audio. Natomiast i tak projekt jest bardzo solidnie prowadzony i jak dotąd postrzegam wybór AI SDK jako właściwy. Jednocześnie zaimplementowanie reszty logiki było na tyle proste, że nie widzę powodu, aby na ten moment sięgać po frameworki, które zrobią to za mnie.

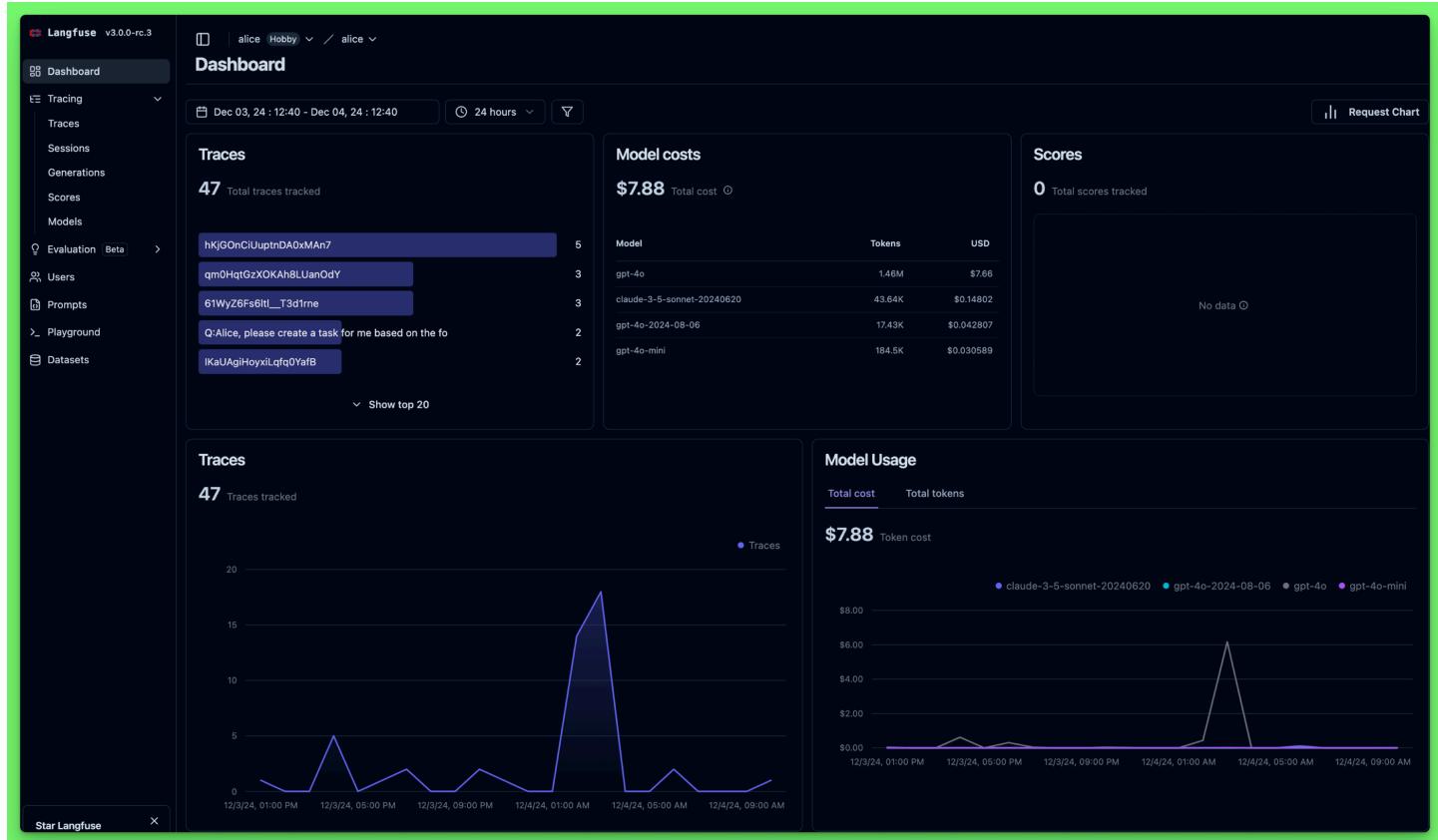
Główny kod aplikacji tworzę w Node.js, wykorzystując framework [hono.dev](#), z którym pracuję po raz pierwszy. Podobnie, debiutuję w pracy z SQLite w połączeniu z [Drizzle ORM](#). Te decyzje skutkują bardzo małą listą zależności projektu, gdzie większość bibliotek jest związana z narzędziami używanymi przez agenta.

Zatem zarówno framework, baza danych, jak i struktura katalogów aplikacji jest dopasowana do małego projektu, ponieważ takie jest też jego założenie. Zależy mi na tym, aby całość była tak prosta, jak to możliwe, ale opierała się o solidne fundamenty (choć te moim zdaniem w tej chwili są jeszcze dyskusyjne).

```
src
  config
  database
    migrations
    db.ts
    migrate.ts
    seed.ts
  dto
  middleware
  prompts
    agent
    tools
  routes
  schema
  services
    agent
    common
    tools
    types
    utils
    app.ts
    index.ts
```

Mamy więc kilka plików konfiguracyjnych, pliki migracji bazy danych, funkcje middleware, katalogi z promptami, ścieżkami, serwisami oraz kilka funkcji pomocniczych. Najwięcej dzieje się naturalnie w akcjach przypisanych do **ścieżek** (katalog routes) oraz poszczególnych **serwisach**. Całkiem istotne są także funkcje **middleware**, które dbają o transformację zapytań z formatów OpenAI/Anthropic na format Vercel AI SDK.

No i oczywiście w temacie rozwoju promptów oraz obserwowania aplikacji, do gry wchodzi tutaj Langfuse oraz Promptfoo (aczkolwiek ten nie jest jeszcze podłączony). Sama integracja odbywa się bezpośrednio przez oficjalne SDK, co daje kontrolę nad monitorowanymi zdarzeniami.



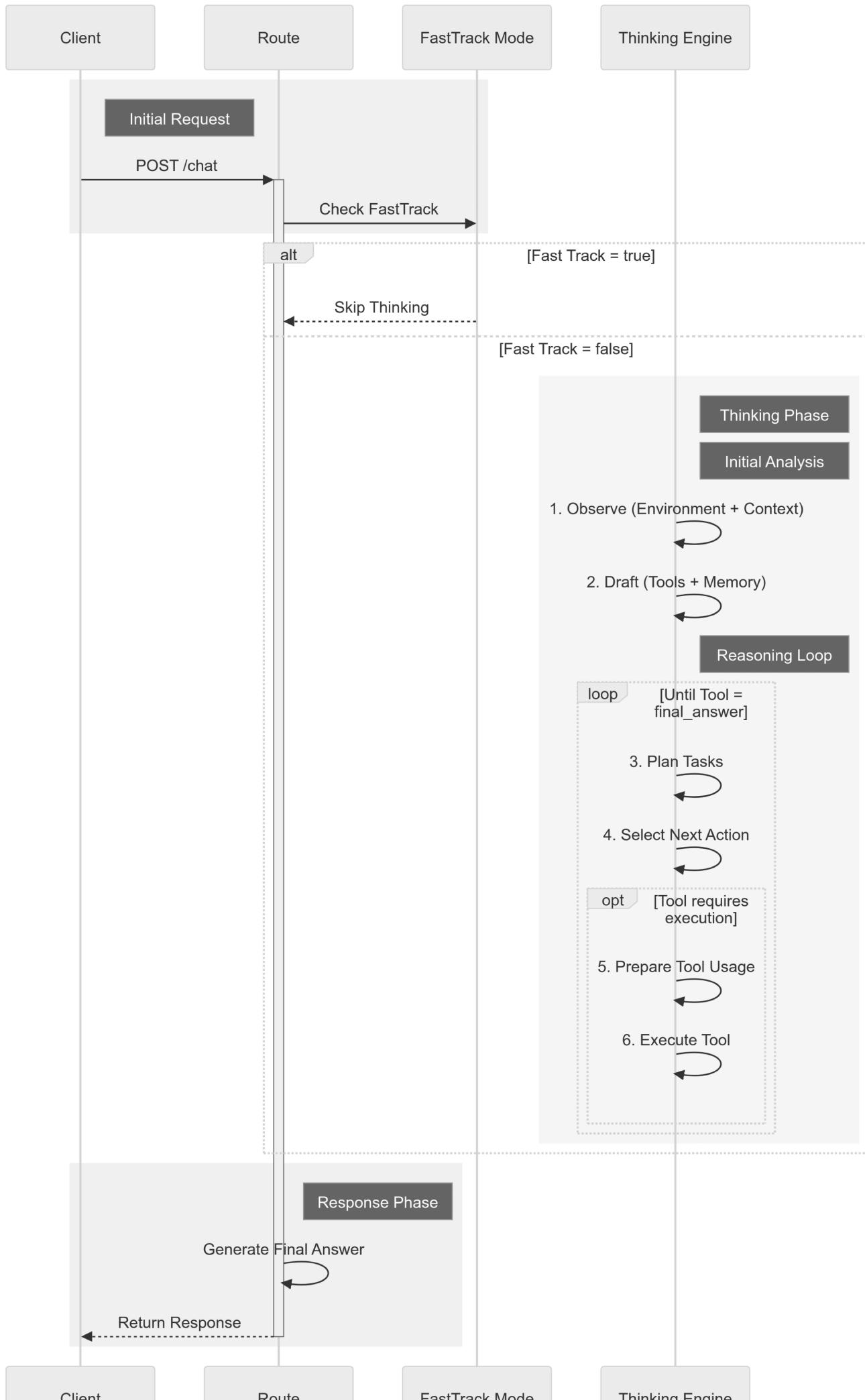
Nie zdecydowałem się jedynie na podłączenie promptów pod Langfuse, ale nie wykluczam tego w przyszłości. Po prostu na ten moment, prompty obecne w plikach projektu wydają się wystarczającym rozwiązaniem.

## Główna logika

Logika Alice AGI to zmodyfikowana wersja przykładu logiki agenta, którą omawialiśmy w kilku ostatnich lekcjach.

Co prawda warto spojrzeć na nią z lekkim dystansem, ponieważ opiera się wyłącznie o moje doświadczenie i już na tym etapie nie mam żadnych publikacji czy przykładów, które są w stanie poprzeć moje założenia.

Trudno mi więc na tym etapie określić możliwości tego systemu, choć ostatnie tygodnie pokazały mi, że są one znacznie lepsze od tego, z czym pracowałem do tej pory.





Mamy więc tutaj:

- Główny endpoint chat, który przyjmuje i wysyła dane zgodne z formatem OpenAI. Wspiera także dodatkową właściwość conversation\_uuid, o której mówiłem w kontekście rozpoznawania wątków.
- Etap "fast track" to klasyfikacja zapytania pod kątem tego, czy odpowiedź wymaga uruchamiania pełnej logiki agenta. Ułatwia to prowadzenie rozmowy, ponieważ nie zawsze wiadomości wymagają podejmowania nowych akcji.
- Etap "observe" polega na kompresji informacji pochodzących z otoczenia oraz profilu użytkownika. Dochodzi tutaj także do naszkicowania wstępnego planu działań.
- Etap "reasoning loop" zawiera w sobie niemal tę samą logikę, którą omawiałem w przykładzie assistant. Mamy więc tam planowanie zadań, oraz akcji które muszą być podjęte w związku z nimi.
- Etap "response" to po prostu wygenerowanie odpowiedzi na podstawie wszystkich dostępnych informacji

Jest to stosunkowo prosta logika, która pozwala na połączenie agenta z zestawem narzędzi, które także omawialiśmy w AI\_devs 3. Chciałbym jednak, aby na tym etapie już nic nie pozostawało wątpliwości.

Poniżej mamy nasz główny endpoint. Cała logika sprowadza się tutaj do:

- Ustalenia stanu konwersacji, czyli wczytania wszystkich niezbędnych danych.
- Klasyfikacji zapytania pod kątem tego, czy może zostać zaadresowane bez fazy "myślenia"
- Wygenerowania odpowiedzi albo w formie zwykłej odpowiedzi, albo strumieniowania.

```

15
16 export default new Hono<AppEnv>().post('/chat', async c => {
17   const request = c.get('request');
18   const conversation_id = await setInteractionState(request);
19   const trace = observer.initializeTrace(request.conversation_id || 'general');
20
21   await aiService.fastTrack(trace);
22   let state = stateManager.getState();
23
24   if (!state.config.fast_track) {
25     await aiService.think();
26   }
27
28   state = stateManager.getState();
29
30   const messages: CoreMessage[] = [{role: 'system', content: answerPrompt(state)}, ...request.messages];
31   const final_generation = observer.startGeneration({name: 'final_answer', input: messages});
32   const result = request.stream
33     ? await completion.stream({...request, messages})
34     : await completion.text({...request, messages}, true);
35
36   if (!request.stream && isChatCompletion(result)) {
37     observer.endGeneration(final_generation.id, result);
38
39   const final_task = state.interaction.tasks.find(task => task.type === 'final');
40   const response_content = result.choices[0]?.message?.content || '';
41
42   if (final_task) {
43     await Promise.all([
44       taskService.updateTaskStatus(final_task.uuid, 'completed'),
45       taskService.updateTaskResult(final_task.uuid, response_content),
46       state.config.current_action?.uuid && actionService.updateActionWithResult(state.config.current_action.uuid, 'This turn was completed. ')
47     ]);
48   }
49
50   await setAssistantResponse({conversation_id, response: result});
51 }
52
53 return request.stream && isStreamResponse(result)
54   ? streamResponse(c, result, {
55     traceId: trace.id,
56     generationId: final_generation.id,
57     messages: request.messages,
58     conversation_id
59   })
60   : c.json(result);
61 }
62
63

```

Dodatkowo, zanim jeszcze zostanie uruchomiona powyższa akcja, dane przechodzą przez zestaw funkcji "middleware". Odpowiadają one między innymi za zabezpieczenie ścieżek przed zbyt dużą liczbą zapytań oraz nieautoryzowanym dostępem. Także wszystkie zapytania (poza kilkoma wyjątkami) muszą posiadać nagłówek Authorization ustawiony na Bearer API\_KEY (wartość tego klucza należy ustawić w pliku .env na dowolną wartość).

```

5
6 export const authMiddleware = () => {
7   return async (c: Context, next: Next) => {
8     if (
9       c.req.path.match(/^\api\auth\google(?:\/.*?$/) ||
10      c.req.path.match(/^\api\auth\api\files\api\file)\/[0-9a-f-]+$/i) ||
11      c.req.path.match(/^\api\auth\spotify\/(?:callback|authorize)/)
12    ) {
13      return next();
14    }
15
16    const authHeader = c.req.header('Authorization');
17
18    if (!authHeader?.startsWith('Bearer ')) {
19      return c.json({error: 'Missing or invalid authorization header'}, {status: 401});
20    }
21
22    const token = authHeader.split(' ')[1];
23    const user = await userService.findByToken(token);
24
25    if (!user) [
26      return c.json({error: 'Invalid API key'}, {status: 401});
27    ]
28

```

To, co niebawem ulegnie tutaj zmianie, dotyczy sposobu strumieniowania informacji tak, aby rozpoczęło się ono **od razu**. Nawiązuje to do jednej z ostatnich aktualizacji AI SDK "non-blocking data streams" o której wspominałem w lekcji S05E04.

Takie strumieniowanie, lub przejście na komunikację opartą o zdarzenia jest istotne zarówno ze względu na długi czas odpowiedzi, jak i w ogóle możliwość obserwowania i ewentualnego wpływu na proces.

No bo jeśli spojrzymy sobie na metodę think, to jej wykonanie zajmuje najwięcej czasu i tam możemy mieć potrzebę **przerwania wykonania lub nawiązania kontaktu z użytkownikiem**.

```

57
58     think: async () => {
59       const observingSpan = observer.startSpan('observing', {
60         phase: 'observation_and_drafting'
61       });
62
63       await aiService.observe(observingSpan);
64       await aiService.draft(observingSpan);
65
66       observer.endSpan(observingSpan.id);
67
68       while (shouldContinueThinking()) {
69         const thinkingSpan = observer.startSpan(`thinking ${stateManager.getState().config.step}`, {
70           phase: 'reasoning_loop'
71         });
72
73         await aiService.plan(thinkingSpan);
74         await aiService.next(thinkingSpan);
75
76         const state = stateManager.getState();
77         if (state.config.current_tool?.name === 'final_answer') break;
78
79         const payload = await aiService.use(thinkingSpan);
80         if (payload) {
81           await aiService.act(payload, thinkingSpan);
82         }
83
84         stateManager.updateConfig({
85           step: stateManager.getState().config.step + 1
86         });
87
88         observer.endSpan(thinkingSpan.id);
89       }
90     },
91   },
92 }
```

Skoro już jesteśmy przy fazie thinking, to wewnętrz niej działają następujące zapytania:

- **observe**: wykonywane są tam dwa prompty, które pobierają tylko istotne informacje z **otoczenia** oraz **profilu użytkownika**. Chodzi o to, że z wielu dostępnych danych wydobywamy tylko te, które są kluczowe w danym momencie.
- **draft**: tutaj także wykonywane są dwa prompty. Jeden z nich generuje notatki na temat potencjalnie istotnych obszarów pamięci do przeszukania, a drugi robi to samo z listą narzędzi. Inaczej mówiąc, tutaj agent wstępnie zastanawia się, co zrobić. Jest to istotne, ponieważ taka refleksja zwykle jest poprawna i wzmacnia nacisk na podjęcie tych działań.
- **plan**: etap planowania dąży do wygenerowania **listy zadań** potrzebnych do udzielenia odpowiedzi.
- **next**: tutaj dochodzi do wyboru **kolejnej akcji** w ramach wybranego przez model zadania.

- **use:** polega na wygenerowaniu danych potrzebnych do uruchomienia akcji. W przypadku Function Calling, byłoby to wygenerowanie argumentów potrzebnych do uruchomienia funkcji. W moim przypadku jest to po prostu zwykły obiekt JSON
- **act:** tutaj akcja zostaje wykonana, a jej rezultat zostaje zapisany w bazie danych i wpływa na wykonanie kolejnego obrotu pętli

No i klasycznie, pętla while jest ograniczona programistycznie w taki sposób, że zostanie przerwana, gdy dojdzie do wyboru akcji final\_answer lub gdy zostanie przekroczena dopuszczalna liczba kroków.

Powyższe podejście sprawdza się świetnie w przypadku zadań, których plan można ustalić od razu, ale ma jeszcze problemy ze scenariuszami, które wymagają znaczącej zmiany podejścia w trakcie wykonywania. Są to dość rzadkie scenariusze występujące w najbardziej złożonych zadaniach, ale jednak się zdarzają.

## API

API opiera się przede wszystkim o główny endpoint /api/agı/chat, ale nie jest to jedyny adres, na który możemy kierować zapytania.

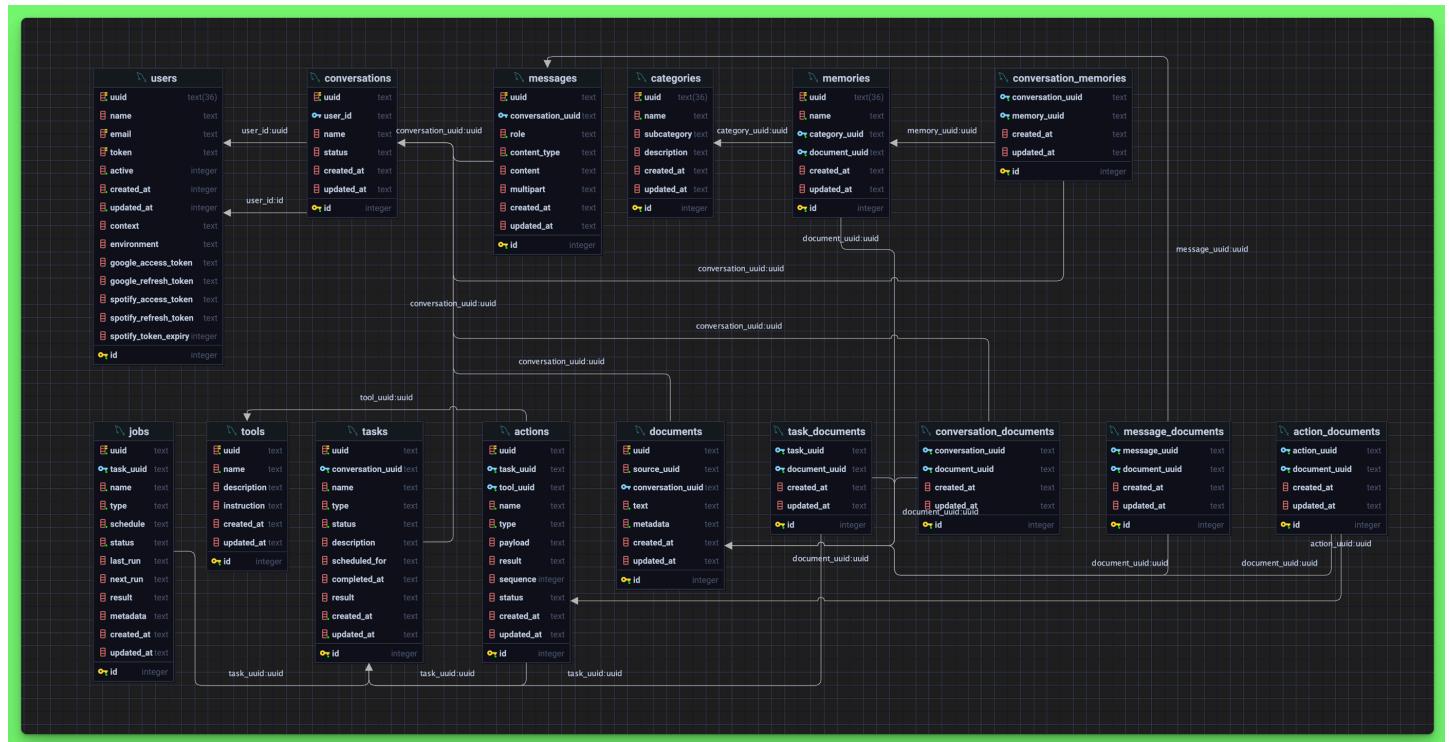
- POST /api/conversation/new: pozwala utworzyć nowy wątek rozmowy i zwraca jego identyfikator. Jest to przydatne w kontekście integracji z zewnętrznym interfejsem, np. Siri Shortcuts
- GET /api/conversation: pobiera konwersację, która miała miejsce w ciągu ostatnich 15 minut. To również jest przydatne w kontekście integracji z zewnętrznym interfejsem. Warunkiem jest tutaj wykluczenie interakcji, które nie zostały uruchomione automatycznie, np. na potrzeby automatyzacji czy harmonogramu zadań
- GET /api/auth spotify/authorize: pozwala na nawiązanie połączenia ze Spotify przez OAuth 2.0
- GET /api/auth google/authorize: pozwala na nawiązanie połączenia z API Google (Calendar / YouTube / Gmail) przez OAuth 2.0
- GET /api/files/:uuid: pozwala na wczytanie treści pliku/dokumentu na podstawie jego UUID
- POST /api/upload: pozwala na wgranie pliku i powiązanie go z konwersacją
- POST /api/tools/memory: pozwala na dodanie dodanie nowego wspomnienia, bez angażowania pełnej logiki agenta
- POST /api/tools/memory/search: pozwala na dostęp do pamięci agenta, bez angażowania jego pełnej logiki
- PATCH /api/tools/memory/:uuid: pozwala na aktualizację wybranego wspomnienia

Część pozostałych narzędzi również posiada własne endpointy, ale nie będę ich tutaj wymieniał, ponieważ nie odgrywa to już zbyt istotnej roli. Kluczowe jest natomiast to, że niektóre z umiejętności asystenta są dostępne bezpośrednio. Także z poziomu automatyzacji czy skryptów, które precyzyjnie dostarczają jakieś informacje, nie ma sensu kierować zapytania na główny endpoint.

Natomiast zapytania ze strony użytkownika, zawsze są kierowane na /api/agj/chat i tam agent podejmuje decyzję o tym, co należy z nimi zrobić.

## Baza danych

Struktura bazy danych także wygląda dość znajomo w porównaniu do tego, co omawialiśmy w lekcjach. Jednak tutaj mamy kilka dodatkowych tabel, co widać na poniższym schemacie (otwórz go w nowej karcie):



- **users**: to klasyczna tabela przechowująca dane użytkowników. Nietypowe są jednak tutaj pola **environment** (JSON) oraz **context** (text) i ich rolą jest przechowywanie aktualnych danych na temat otoczenia oraz ogólnego profilu użytkownika, o których wspomniałem przed chwilą.
- **conversations**: to lista interakcji przypisanych do użytkownika oraz listy wiadomości. Wyodrębnienie tej tabeli jest istotne, bo można rozpoczęć wątek przed interakcją użytkownika. Na przykład, przy wgrywaniu plików do nowej rozmowy, zakładamy wątek, do którego użytkownik będzie mógł później przesyłać wiadomości.
- **messages**: tutaj mamy listę wiadomości, lecz wspierają one także możliwość przechowywania obrazów. Są także połączone z tabelą documents dzięki czemu możliwe jest powiązanie plików z konkretną wiadomością.
- **categories & memories**: to bazowa struktura wspomnień. Każde ze wspomnień również jest powiązane z tabelą documents, której wpisy są (ale nie zawsze) indeksowane w silnikach wyszukiwania.
- **tools**: to lista narzędzi, ich opisów oraz instrukcji wykonania. Narzędzia przypisywane są do akcji.
- **tasks**: to lista zadań, które muszą zostać wykonane w celu udzielenia odpowiedzi. Rezultat zadania ma formę wpisu w tabeli documents

- actions: to lista akcji, które muszą zostać wykonane **w ramach jednego z zadań**. Rezultat akcji ma formę wpisu w tabeli documents
- jobs: to lista zadań wywoływanych według harmonogramu. Tutaj istnieje powiązanie z tabelą tasks oraz nie bezpośrednio z conversations. W ten sposób, gdy nadchodzi odpowiedni moment, system wysyła wiadomość 'sam do siebie' i podejmuje zlecone zadanie.

Widać tutaj wyraźnie, jak istotną rolę odgrywa tabela documents. Jej wpisy pojawiają się praktycznie wszędzie, ale możemy korzystać z nich w jednolity sposób. Nie ma znaczenia, czy "dokumentem" jest plik, wynik działania akcji, czy informacja o błędzie - każdy z nich można przetwarzać według tych samych zasad.

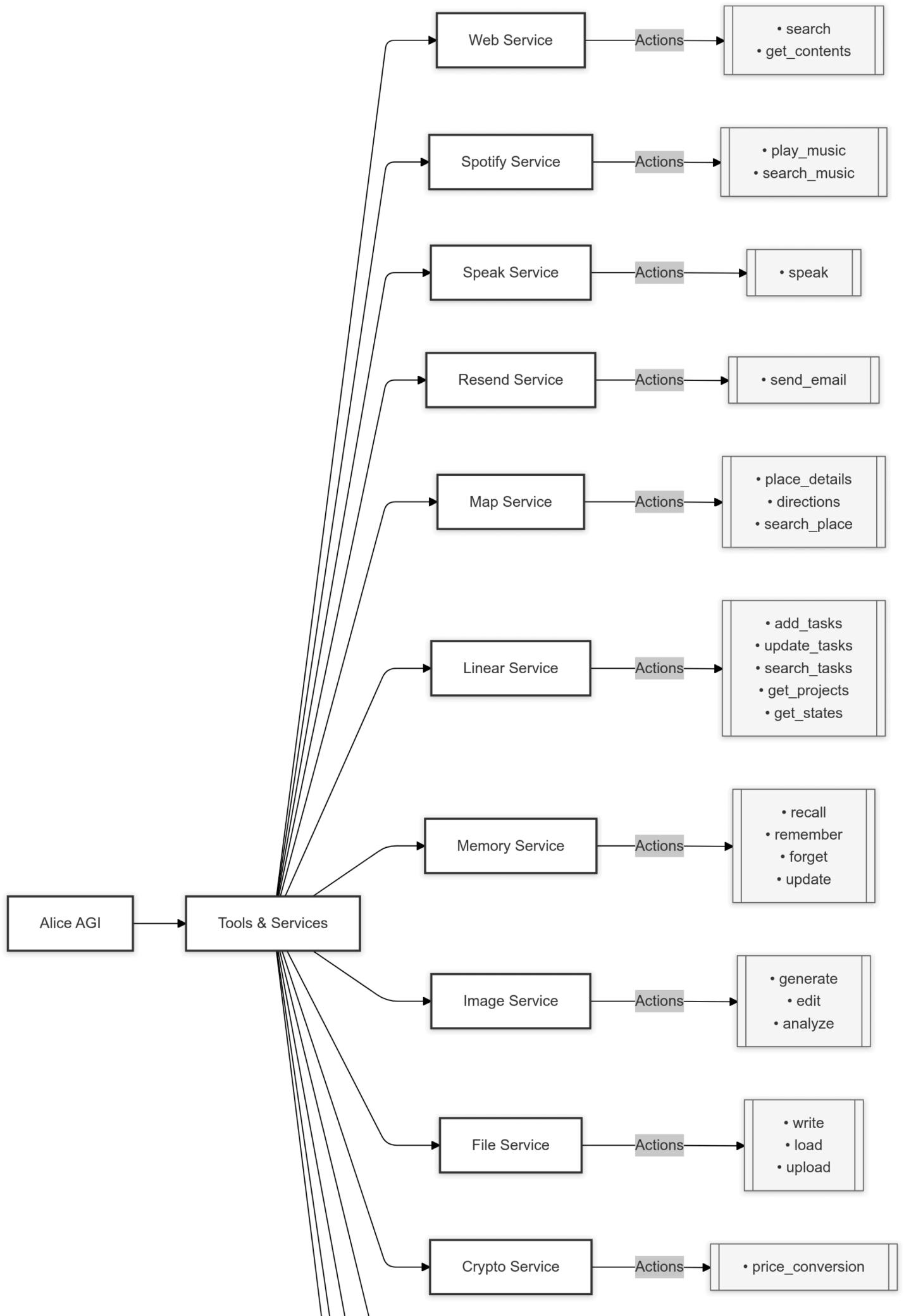
Ogólna struktura bazy danych w jej obecnej formie, ma jednak przynajmniej kilka wad oraz mniejszych nieścisłości.

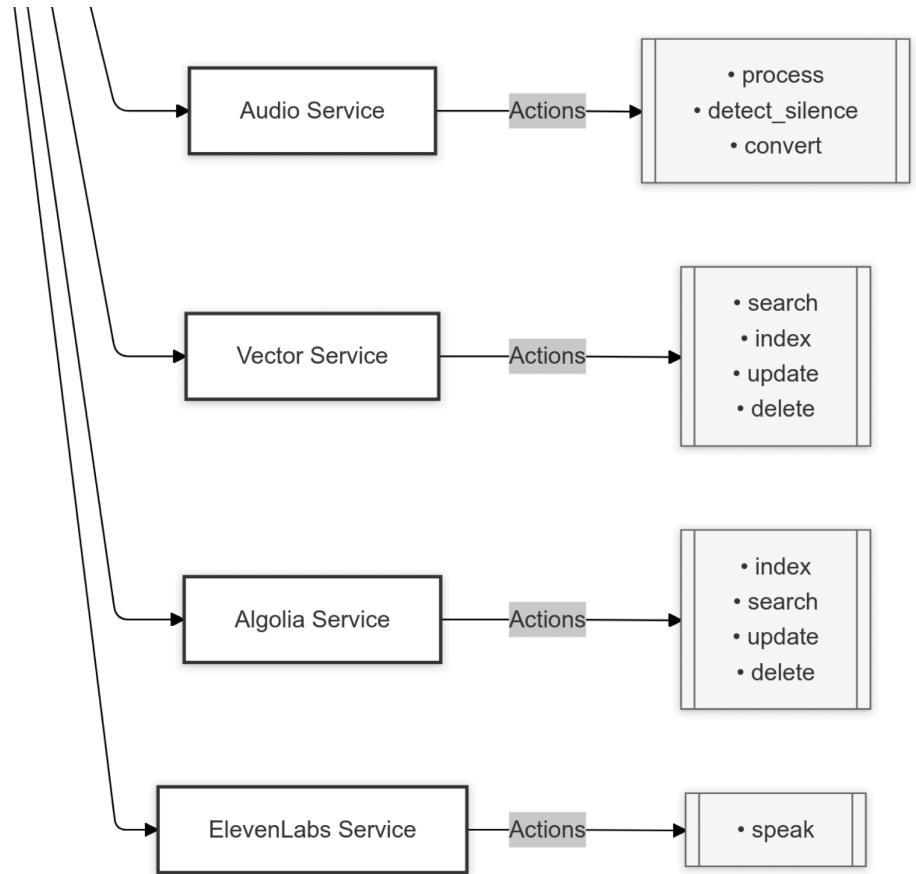
Dla przykładu mam wątpliwość czy tasks powinno mieć powiązanie z konwersacją, czy z messages. Powodem jest fakt, że bez tego nie jestem w stanie odtworzyć pełnej historii interakcji. Obecnie nie stanowi to problemu, ale może się nim stać w przypadku najbardziej złożonych zadań. Nie jestem także przekonany do struktury dokumentów, ale ponownie, obecnie nie mam z tym problemu.

Wniosek jest taki, że obecna struktura bazy danych jest bardzo elastyczna, a zarazem stosunkowo prosta i generyczna. Może stanowić fundament dla różnego rodzaju agentów, choć sama w sobie w obecnej formie nie przewiduje interakcji agent - agent.

## Dostępne narzędzia

Na ten moment Alice AGI posiada dostęp do 14 narzędzi i 33 umiejętności (konkretnie jest ich 37, ale mamy dwa rodzaje wyszukiwania). W skład narzędzi wchodzą te najbardziej podstawowe, takie jak **pamięć, dostęp do Internetu, listy zadań czy kalendarza**. Jest także możliwość przetwarzania plików oraz dostęp do usług pozwalających na wysłanie maila czy sterowanie muzyką.





Jeśli chodzi o narzędzia, opieramy się dokładnie o te same koncepcje, które omawialiśmy. Poza drobnymi porządkami, jest to kod pochodzący wprost z repozytorium przykładów AI\_devs 3.

Podłączenie nowego narzędzia, opiera się tutaj o kilka kroków.

Utworzenia serwisu w katalogu src/services na wzór tych istniejących. Kluczową funkcją jest tam `execute`, która przyjmuje nazwę akcji oraz payload. W pliku `spotify.service.ts` znajduje się dobry przykład.

```

543
544     execute: async (action: string, payload: any, span?: LangfuseSpanClient) => {
545       const state = stateManager.getState();
546
547       span?.event({
548         name: 'spotify_tool',
549         input: { action, query: payload.query, conversation_uuid: state.config.conversation_uuid },
550         output: { success: true, action_executed: action }
551       });
552
553       if (action === 'play_music') {
554         return spotifyService.playMusic(payload.query, state.config.conversation_uuid ?? 'unknown', span);
555       }
556       if (action === 'search_music') {
557         return spotifyService.searchMusic(payload.query, state.config.conversation_uuid ?? 'unknown', span);
558       }
559
560       throw new Error(`Unknown action: ${action}`);
561     }
562   };
563
  
```

Następnie serwis musi zostać dodany w pliku `config/tools.config.ts`, bo dzięki temu będzie dostępny w metodzie `act` z pliku `ai.service.ts`

```
16  export const toolsMap: Record<string, ToolService> = {  
17    spotify: spotifyService,  
18    memory: memoryService,  
19    resend: resendService,  
20    files: fileService,  
21    speak: speakService,  
22    linear: linearService,  
23    maps: mapService,  
24    crypto: cryptoService,  
25    google: webService,  
26    calendar: calendarService  
27  } as const;  
28  
29  export type ToolName = keyof typeof toolsMap;
```

W bazie danych musi zostać dodany wpis do tabeli tools, również na wzór tych istniejących. Wiemy też, że tam — nazwa musi być krótka i unikatowa, opis musi wyjaśniać kiedy z tego narzędzia należy skorzystać, a instrukcja opisują sposób wygenerowania danych do jej uruchomienia.

No i właściwie nic więcej nie jest potrzebne do dodawania nowych narzędzi. Oczywiście w ramach jednego narzędzia można mieć wiele akcji, co jest bardzo wskazane. Warto więc zapoznać się z istniejącymi już narzędziami, aby na ich podstawie podjąć decyzję o dalszych krokach.

Także podsumowując kwestię głównej logiki, **Alice AGI to tak naprawdę połączenie dwóch rzeczy — narzędzi, które budowaliśmy w lekcjach, oraz logiki agenta, którą omawialiśmy w tym tygodniu.**

## Uruchomienie projektu

Szczegóły na temat uruchomienia projektu zostały opisane przeze mnie w pliku [README.md](#). W skrócie należy:

- pobrać repozytorium projektu: [pobierz](#)
- zainstalować zależności: bun install
- przeprowadzić migrację bazy danych: bun migrate
- wypełnić ją początkowymi danymi: bun seed (można je dostosować w pliku seed.ts)
- skopiować plik .env-example i zmienić jego nazwę na .env
- wewnątrz pliku .env należy ustawić API\_KEY na dowolną wartość, oraz klucze API dla OpenAI, Anthropic i wszystkie zmienne dla Langfuse, Qdrant, Algolia. Pozostałe zmienne są opcjonalne, ponieważ są bezpośrednio powiązane z narzędziami, które i tak trzeba dopasować do własnych potrzeb, lub stworzyć własne.

## AI\_devs – Alice AGI Config



Zatem samo uruchomienie projektu, nie jest szczególnie skomplikowane. Dobrze jednak zapoznać się ze strukturą narzędzi oraz logiki agenta. Nie bez powodu dołączam zestaw przykładowych integracji, ponieważ są one dobrym punktem odniesienia.

Po uruchomieniu projektu, potrzebny nam będzie jeszcze graficzny interfejs. Tutaj możliwe jest połączenie pod aplikację Alice (3-miesięczny trial dla AI\_devs 3 dostępny jest [tutaj](#)) lub poprzez szablon aplikacji Tauri, który publikowałem [tutaj](#).

Na filmie poniżej pokazuję możliwości Alice AGI w zakresie posługiwania się narzędziami. Widać tutaj także wyraźnie aktualne ograniczenia modeli językowych (np. szybkość działania), ale także możliwości związane z przetwarzaniem poleceń zapisanych w naturalnym języku.

## AI\_devs – Alice AGI



Alice AGI jasno pokazuje to, że obecnie zbudowanie wyspecjalizowanego agenta, poruszającego się po określonym obszarze, jest już w pełni możliwa. Podobnie wygląda to w zastosowaniach produkcyjnych, choć tutaj ilość ograniczeń, które będziemy chcieli nałożyć na system, będzie odpowiednio większa, ponieważ ilość zmiennych wpływających na jego działanie, rośnie bardzo szybko.

## Alice AGI w praktyce

Alice AGI to projekt-piaskownica. Można więc go wykorzystać do nauki i eksplorowania możliwości modeli, albo z góry określić jego zakres, wdrożyć i po prostu patrzeć jak dla nas pracuje. W moim przypadku są to oba scenariusze, ponieważ projekt nieustannie ewoluje, ale też coraz większy zakres funkcjonalności pozostaje ze mną na stałe.

W moim przypadku na stałe są ze mną:

- Głośowe dodawanie zadań, szczególnie gdy jestem poza domem. Możliwość podyktowania wiadomości, która zamienia się na zadania, wciąż robi na mnie wrażenie.
- Dodawanie wydarzeń w kalendarzu przez wiadomość na Slacku, gdy nie mam możliwości swobodnie mówić ze względu na hałas bądź wprost przeciwnie, ciszę, ale w otoczeniu innych.
- Pamięć długoterminowa — tutaj jest trochę prywatny (ale bez przesady). Wpisy na temat przeczytanych przeze mnie książek, prowadzonych projektów, testów osobowości czy luźnych notatek pozwalają na prowadzenie jakościowej dyskusji sprawiającej wrażenie, że po drugiej stronie jest ktoś, kto mnie zna. Choć omówienie niektórych tematów nadal sprawia wrażenie rozmowy z botem, tak nierzadko wśród wymienionych wiadomości pojawiają się sugestie na które bym nie wpadł. Aktualnie moduł pamięci jest przeze mnie modyfikowany ze względu na inspirację projektem AGI Memory

- Automatyzacje działające według harmonogramu lub zdarzeń. To akcje podłączone do mojej listy zadań, katalogów na Google drive czy nawet chwili w której bluetooth łączy się z moim samochodem. W zależności od kontekstu otrzymuję wiadomość lub po prostu uruchamia się jakaś akcja. Przykładem może być współpraca z osobą odpowiedzialną za montaż części z moich filmów — tam komunikacja, zadaniami oraz rozliczeniem, w pełni zarządza Alice.
- No i oczywiście fun. Nie bez powodu tak dużo w AI\_devs było Spotify, zapalania światła czy sterowania akcjami w samochodzie przez API. Uważam, że właśnie te najbardziej absurdalne akcje, które z pozoru są bezużyteczne, uczyły mnie najwięcej.

Oczywiście nie wszystko zawsze jest idealne, ale największe problemy są związane z trzema obszarami — niedostępnością API modeli i/lub zewnętrznych usług, niedostępnością Internetu w podróży lub w niektórych pomieszczeniach, np. w garażu, brakującymi funkcjonalnościami lub ograniczeniami tych istniejących.

Same ograniczenia modeli językowych praktycznie już nie występują, ale mowa tutaj o sytuacji, w której sam jestem użytkownikiem tego systemu. Przekazanie go w ręce innych osób ogranicza się tylko do pojedynczych akcji lub umiejętności (np. tłumaczenia bądź korekty dokumentu na podstawie adresu URL w Notion). Powodem jest fakt, że tak "otwarty" system nadal trudno dopasować do potrzeb dowolnego użytkownika. Nawet integracja z kalendarzem w przypadku osoby pracującej na kilku kontach to już zupełnie inny temat niż integracja z pojedynczym kalendarzem.

Zarówno współpraca z dotychczasowymi wersjami Alice AGI, jak i sama budowa, jasno pokazują mi, że:

- rozwój modeli nadal trwa i bynajmniej nie widzę tutaj spowolnienia
- większość wskaźników takich jak większy kontekst, czy spadające ceny, mają mniejsze znaczenie, niż twierdzą komunikaty marketingowe
- nadal nie wiemy praktycznie nic na temat natury modeli językowych. Zdarza mi się chodzić nietypowymi ścieżkami i nieustannie zaskakiwać się tym, co jest możliwe lub wprost przeciwnie
- warto obserwować ludzi stojących za tą technologią, narzędziami i badaniami. Jednak bezwarunkowo należy pamiętać, że są to tylko ludzie, a na przestrzeni nawet dwóch ostatnich lat duża część powszechnie uznawanych 'prawd' została podważona lub okazała się błędna od samego początku
- nic nie daje większej wartości, niż samodzielna eksploracja i budowanie, połączone z wymianą pomysłów w gronie osób, które uważają tak samo

Ostatecznie nie wiem, czy projekt Alice AGI lub choć sama idea, pozostaną z Tobą na dłużej. Mam jednak nadzieję, że ostatnie tygodnie były dla Ciebie wartościowe i znajdziesz praktyczne zastosowanie tej wiedzy i umiejętności w Twojej codzienności.

## Kolejny rozdział

Patrząc na ostatnie dwa lata rozwoju generatywnego AI, widzimy, że początkowo mogliśmy pytać o odpowiedzi, później o działania, a następnie serie działań.

Dziś możemy dość swobodnie rozmawiać z modelem, a nawet stawiać przed nim cele, które samodzielnie jest w stanie osiągnąć. Wymaga to jednak znajomości natury modeli, a także programistycznej wiedzy. Pozwala to na zaplanowanie systemu w taki sposób, aby mógł spełniać swoje zadanie w sposób przewidywalny lub na tyle skuteczny, aby jego zastosowanie miało sens.

AI\_devs 3 właśnie dobiegło końca. Oznacza to, że w tej chwili rozpoczyna się kolejny rozdział historii współpracy ze Sztuczną Inteligencją, którego część ... napiszesz Ty.

Myślę, że teraz już każdy zgodzi się ze mną, że generatywne AI jest już nieodłączną częścią naszej pracy. Całkiem jasne jest także to, że przynajmniej na ten moment, budowanie kompleksowych systemów nadal możliwe jest tylko poprzez **połączenie naszego doświadczenia z możliwościami nowych technologii**. I nawet jeśli dziś spotykamy jeszcze bariery wynikające bezpośrednio z umiejętności obecnych modeli, to nawet wzorajsza premiera o1 sugeruje, że część z nich niebawem zniknie, ale niewiele wskazuje na to, że wydarzy się to "jutro".

---

Z całego serca dziękuję Ci za udział w AI\_devs 3 ❤️ Dla mnie prowadzenie tego szkolenia to zaszczyt, szczególnie biorąc pod uwagę to, że Twoja obecność tutaj to oznaka zaufania wobec mnie, mojej wiedzy, umiejętności oraz doświadczeń za które jestem Ci wdzięczny.

Zdaję sobie sprawę z tego, że w tej edycji nie wszystko było idealne oraz, że tu i tam pojawiały się mniejsze lub większe trudności. Jednak fakt, że czytasz te słowa sugeruje, że materiał był na tyle interesujący, aby było warto przez niego przejść.

Na koniec mam więc prośbę o dopisanie się do naszej "Ściany Miłości" z pomocą [tego formularza](#). Jeśli chcesz też zostawić sugestie i/lub feedback na temat szkolenia, to niebawem prześlemy także dedykowaną ankietę w tej sprawie.

Nigdy, ale to nigdy tak dobrego kursu nie przchodziłem. Czuje się jakiejś trójcy czuwało nieustannie. I wiele można wrzucić tylko zadania, super wy tłumaczyć i bawić, natomiast u was to było coś więcej. Budowane poczucia bycia w grupie, rozwij community oraz motywacji sprawły że zostało ostatni tydzień do nadrobienia i kurs skonczony. Nie jak zawsze - na półkę wiecznego backlogu. Może i żona, chce rozrodu, a dziecko mówi že zniszczyłem jej czasy dzieciństwa... No ale coś za coś - było warte. ~Masz memika Mateusz <3



**Mateusz Mietelski**  
Manager IT

Ten kurs to złoto! Świetna i przyjemna w odbiorze forma, a w szczególności zadania praktyczne, które zapewniły mi rozrywkę na wiele długich wieczorów. DŁatwo nie było, ale satysfakcja ze zdobyczy wiedzy jest ogromna. Do tej pory chyba żaden kurs nie sprawił, że od razu po jego ukończeniu, jestem w stanie przejść do praktyki z poczuciem, że wiem, co robię. Aktualnie myślenie o wykorzystaniu Gen AI to już nie tylko teoretyzowanie, ale pełna świadomość możliwości i przede wszystkim tego, jak je wdrożyć. Wielkie dzięki i do zobaczenia w kolejnej odsłonie kursu :)

**Kasia**  
Automation Developer

Zdecydowanie najlepsze szkolenie na jakim byłem.

Programuję w technologii, która jest słaba w użytkowaniu API, więc bałam się czy czasowo ogarnę uczenie się typeScript/pythona + przyswajanie wiedzy z AI z myślą dzieci w domu. Jednak narzędzia no-code przyszły z pomocą i udało się zaangażować w pełni w szkolenie bez zaniedbywania pracy i rodziny z czego bardzo się cieszę :)

Pozostaje pozytywna energia w głowie oraz chęć do dalszego zgłębiania tematów poruszonych w szkoleniu.

Bardzo polecam!!!

**Paweł Stryjewski**  
Programista IBM RPG

Nie ma co ściniać, że "to jest jedno z najlepszych szkoleń na jakich byłem". To po prostu jest najlepsze szkolenie na jakim można być. **Masa wiedzy dozwolona stopniowa sprawia, że po pierwszym tygodniu cekasz na kolejną lekcję bardziej niż na Świętego Mikołaja.** Zadania praktyczne, mimo że każde inne to nagle odkrywasz jak wszystko łączy się w całość. **Kurs AI\_Devs daje szansę, że sztuczna inteligencja nie będzie madrzejza niż Twoja własna inteligencja.**

**Amadeusz Walkiewicz**  
Backend dev / DevOps

**Wspaniały kurs dla zabiętych, ale jednak technicznych od urodzenia i stęsknionych za nowościami.** To nie tylko kurs o generatywnym AI, ale też wrota do nowych pomysłów i idei. Dokładnie można zrozumieć, jak działają modele LLM i czego można od nich oczekwać, a czego nie. Kurs lagodzi niepokój, że nie wiemy, co się na świecie dzieje, a my nie nadajemy. Po tym kursie: nadajamy i jest zdecydowanie poczucie bycie blisko awangardy. **Zaczynam rozumieć, że ludzie lubią brać udział w kolejnych edycjach :)**

**Dariusz Charmuszko**

Świetne szkolenie, mocne merytoryczne z fajnymi zadaniami praktycznymi. **Człopaki udzielają rewelacyjnych rad, które skutkują tym, że nauka z nimi to kawał dobrej programistycznej zabawy, która z powodzeniem może się przekladać później na płaszczyźnie projektowej.** Powodzenie z następnymi edycjami. Zdecydowanie polecam.

**Michał Furmaniak**  
Senior Full-stack Developer/PO/SM/Company Owner - www.linkedin.com/in/mikefurmaniak/

Rewelacyjny kurs, poszerzający (lub budujący) wiedzę o sztucznej inteligencji i związanych z nią zagadnienniach, **bardzo głęboko omawiający warstwę "jak"**, ale również "jakimi narzędziami", a przede wszystkim - "dlaczego?"

Kurs opłaca się chyba najbardziej programistom, ale ja skorzystałem na nim prawdopodobnie jeszcze bardziej, ponieważ zagadnienia są prezentowane zrozumiale i osoba bez wiedzy technicznej jest w stanie sobie z nimi poradzić. Również zadania są zrozumiałe, a wartość dodana są nie tylko

**Wyjątkowy kurs! Połączenie najnowszych zdobytych AI z automatyzacją, potężnymi narzędziami no-code/low-code i programowaniem.**

Prowadzący Adam Gospodarczyk i Jakub Mrugalski dawali z siebie wszystko żeby nauczyć, pokazać i wytrenować uczestników w wyciskaniu pełnych możliwości z obecnych modeli LLM.

Model lekcji do indywidualnej nauki umożliwia przekształcenie kursu we własnym tempie. Jednocześnie aktywność prowadzących na forach i czatach, a także codzienne spotkania bardzo motywowały aby się spinacić i być z materiałem na bieżąco. **Wszelkie wątpliwości można było rozwiewać w najbardziej wygodny dla siebie sposób.**

Dodatekowo **stworzenie społeczności wokół kursu pozwalało na wzajemną pomoc między uczestnikami co dawało wszystkim dodatkowe korzyści.**

Jako uczestnicy otrzymaliśmy mnóstwo bonusów w postaci gotówek do własnego wykorzystania. Niesamowite było też dostosowanie treści do najnowszych informacji i osiągnięć z dziedziny AI pojawiających się w jego trakcie.

Zarówno wiedza pozykana na kursie, jak i samo doświadczenie uczestnictwa w nim były w 300% warte jego ceny.

Polecam uczestnictwo w kursach Adama i Jakuba! Nie ma opcji żeby żałować poświęconych ba nie pieniędzy czy czasu. Super sprawa!

**Jakub Mazurkiewicz**  
IT Manager - https://www.linkedin.com/in/jakub-m

Chyba najlepszy kurs w jakim miałem przyjemność uczestniczyć. **Świetna forma pozwalająca na "nie zmuszanie się do kursu, tylko robienia go z przyjemnością.** Ogromna ilość wiedzy i zadań praktycznych. Cieszę się, że nie skupiło się na samym "AI" ale na całym procesie tworzenia np. własnej aplikacji, dzięki czemu **nauczyłem się też technologii, integracji i procesów poza AI o których nic nie wiedziałem.** Czekam na 3.0!

P.S Dajcie więcej merchu, jak wszyscy założymy koszulki I <3 AI (Devs) to może roboty nas oszczędzają :)

**Bartosz Lipski**

Świetnie przygotowane i przeprowadzone szkolenie. Dzięki systemowemu zadań każda lekcja to nie tylko naukę, ale również małe wypoczynek. **Unikalne szkolenie**.

## Specjalne podziękowania kieruję także do:

- Jakuba oraz Mateusza — za opracowanie fabuły oraz systemu zadań oraz niezwykły klimat przy organizacji największego szkolenia online w mojej historii.
- Pawła Dulaka — za niezwykłe zaangażowanie oraz wsparcie przy obsłudze szkolenia oraz testowania zadań
- Pauli Skrzypeckiej — za merytoryczne i zabawne spotkanie, które rzuciło światło na odpowiedzialne wdrażanie systemów w których pojawia się GenAI
- Michałowi Furmankiewiczowi — za świetne spotkanie, które pokazało, że systemy GenAI już teraz są wdrażane w dużych organizacjach
- Mariuszowi Korzekwie, Dominikowi Fudziekiwieczowi oraz Pawłowi Manowieckiemu — za inicjatywy w postaci nieoficjalnego Discorda oraz meetupów OpenAI Devs
- Mateuszowi Szmigielowi, Mateuszowi Tomkiewiczowi, Bartkowi Karalus, Piotrowi Brzyskiemu — za inicjatywy spotkań AI\_devs w różnych miastach
- Zespołu brave.courses — za bezbłędną organizację szkolenia i zadbanie nawet o najmniejsze detale związane z promocją, zewnętrznymi współpracami, wydarzeniami, oprawą graficzną oraz wszystkimi

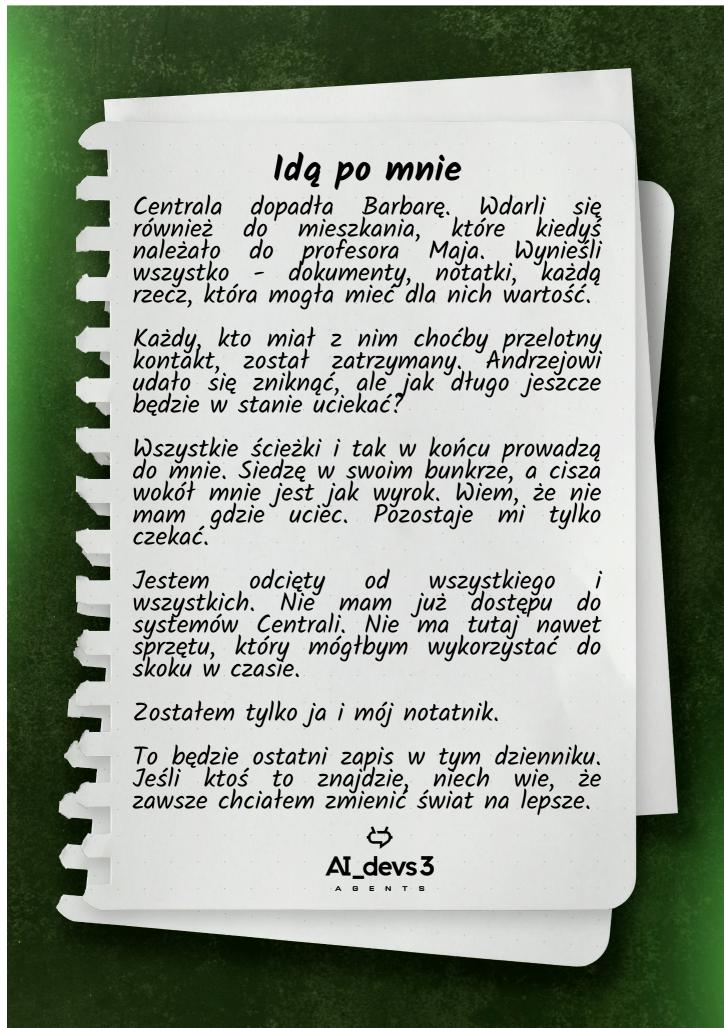
formalnościami

- Zespołu easy.tools — za obsługę płatności oraz pomoc w zakresie integracji naszego systemu zadań
- Alice — za burze mózgów i pomoc przy rozwiązywaniu problemów z przykładami, które wydawały się niemożliwe do rozwiązania.
- oraz przede wszystkim — Katarzynie Gospodarczyk i Grzegorzowi Rogowi — za ich obecność.

Do zobaczenia  

Adam

## NOTATKA ZYGFRYDA



## ZADANIE

Historia dobiegła końca. Azazel mógł już powrócić do swoich czasów, a Centrala po zatruciu systemu robotów rozpoczęła czystki. Uszkodzone maszyny nie stawiają oporu. Większym problemem są za to ludzie, którzy pośrednio i bezpośrednio współpracowali z Zygfrydem. Dokonano wielu zatrzymań, ale profesorowi Majowi udało się uciec.

Ktoś, kogo do tej pory poszukiwano w celu ratowania jego życia i zdrowia, wylądował na liście osób poszukiwanych i podejrzanych o możliwy udział w spisku przeciwko ludzkości. Czy profesor Maj odtworzy brakujące wyniki swoich badań i ostatecznie zbuduje AGI? Tego nie wiemy. Bez pomocy swoich współpracowników i Zygfryda z przeszłości, są na to nikt szanse.

Numerze piąty. Tak jak ostrzegaliśmy, gdy wyruszałeś na tę misję, w roku 2024 nie mamy sprzętu, aby cofnąć Cię do czasów, z których przybyłeś. Pozostaniesz tutaj. Pamiętaj jednak, że jako wysłannik z przeszłości, który opanował sztukę posługiwania się technologiami LLM, ciąży na Ciebie pewien obowiązek. Jesteś jedyną osobą, która na własne oczy widziała, do czego może doprowadzić niekontrolowany i chaotyczny rozwój GenAI. Musisz zmienić ten świat na lepsze. Wykorzystaj swoją wiedzę i działaj.

Centrala ma do Ciebie jedną, ostatnią już prośbę. Pomóż nam ułożyć całą historię i chronologię zdarzeń, które się wydarzyły od Twojego przybycia.

Odpowiedz na pytania centrali:

<https://centrala.ag3nts.org/data/TWOJ-KLUCZ/story.json>

Odpowiedzi prześlij do centrali w standardowy sposób jako tablicę JSON (dokładnie w formie, w jakiej sformułowano pytania) do zadani o nazwie "**story**"

Oczekiwany format odpowiedzi:

```
[  
  "odpowiedź na pierwsze pytanie",  
  "odpowiedź na drugie pytanie",  
  "odpowiedź na trzecie pytanie",  
  "(...)",  
  "odpowiedź na N-te pytanie",  
]
```

Materiały źródłowe, które **mogą** być przydatne przy odpowiedzi na pytania:

- [https://centrala.ag3nts.org/dane/pliki\\_z\\_fabryki.zip](https://centrala.ag3nts.org/dane/pliki_z_fabryki.zip)
- <https://centrala.ag3nts.org/dane/przesluchania.zip>
- [https://centrala.ag3nts.org/dane/zygfryd\\_notatnik.zip](https://centrala.ag3nts.org/dane/zygfryd_notatnik.zip)
- [centrala.ag3nts.org/dane/arxiv-draft.html](https://centrala.ag3nts.org/dane/arxiv-draft.html)
- <https://softo.ag3nts.org/>
- Blog z wpisami Rafała z S05E03
- Transkrypcja rozmów <https://centrala.ag3nts.org/data/TWOJ-KLUCZ/phone.json>



Warto bazę wiedzy zaindeksować sobie w bazie danych (relacyjnej lub wektorowej, a może i obu zsynchonizowanych?), ponieważ ogrom danych uniemożliwi Ci trzymanie ich w ramach kontekstu modelu. Nawet jeśli dane po zoptymalizowaniu zmieszcza się w oknie kontekstowym, to jakość tego rozwiązania będzie naprawdę kiepska.

### Co należy wykonać w zadaniu?

Pobierz listę pytań od centrali

Wykorzystaj wszelkie materiały, z którymi zetknąłeś się wcześniej, aby na nie odpowiedzieć (centrala dostarczyła Ci ich kolekcję)

Odpowiedz na pytania w formacie podanym wyżej i prześlij je do zadania “**story**”

Wykorzystując swoją wiedzę na temat LLM uczyń świat lepszym miejscem, ale nie idź w ślady Zygfryda ;)

Do zobaczenia w przyszłości 