# Pilotaż wdrożenia AI w PARP

Warszawa, luty 2026

**AUTORZY:**

Andrzej Jędrzejowski, Wojciech Teichert, Paweł Fedoruk

Wszelkie wnioski i rekomendacje sformułowane w raporcie stanowią opinię autorów.

# Spis treści

Celem artykułu jest przedstawienie doświadczeń Polskiej Agencji Rozwoju Przedsiębiorczości (PARP) z pilotażu wdrożenia agenta AI, przeprowadzonego w sierpniu 2025 r. Opracowane narzędzie wykorzystywało model LLM w metodzie Retrieval-Augmented Generation (RAG) i zostało uruchomione na infrastrukturze lokalnej.

Opis pilotażu, przedstawiony w rozdziale 2, został poprzedzony omówieniem trzech podstawowych zagadnień, które pozwalają lepiej zrozumieć przedmiot i kontekst pilotażu:

- Jakie jest podstawowe zadanie modelu LLM?
- Czym jest Retrieval-Augmented Generation (RAG)?
- Jakie są kluczowe aspekty działania agentów AI na infrastrukturze lokalnej w porównaniu do rozwiązań chmurowych?

W ostatnim rozdziale przedstawiono podsumowanie oraz plany PARP związane z dalszym wdrażaniem agentów AI w organizacji.

# 1. Podstawowe zagadnienia

## 1.1. Jakie jest podstawowe zadanie modelu LLM?

Podstawowym zadaniem dużych modeli językowych (Large Language Models, LLM) jest **generowanie odpowiedzi na pytanie** zadane przez użytkownika[1]. Modele te nie wyszukują informacji (np. w zewnętrznych bazach danych lub wyszukiwarkach). Ich działanie polega na statystycznym przewidywaniu kolejnych tokenów[2] na podstawie wzorców poznanych w trakcie uczenia się modelu.

Model językowy zawsze wygeneruje odpowiedź na zadane pytanie, przy czym ważne jest, aby zrozumieć co się dzieje, gdy zadane pytanie wykracza poza „wiedzę" modelu[3] (zakres danych, na których model był uczony). W zależności od metody treningu, dany model może:

- **przyznać się do niewiedzy** - wygenerować odpowiedź, w której napisze np., że nie posiada wiedzy na dany temat;
- lub **próbować „zgadnąć"** poprawną odpowiedź – wygenerować odpowiedź, która jest językowo poprawna i brzmi wiarygodnie, lecz może być błędna (zjawisko to określane jest mianem **halucynacji**).

To jak się zachowa zależy od tego jak był uczony – które z powyższych zachowań było nagradzane (czy lepiej zgadywać, czy przyznać się do braku wiedzy?).

Wiedza, na której opierają się odpowiedzi modelu językowego, pochodzi z dwóch głównych źródeł:

1. **Danych treningowych**
   Model jest uczony na bardzo dużym zbiorze tekstów, obejmującym książki, artykuły, strony internetowe, dokumenty techniczne. Wiedza zawarta w tym zbiorze:

1 Jest to celowe uproszczenie mające za zadnie przedstawienie idei. W praktyce, zapytanie (prompt) wysłane do modelu nie musi mieć formy pytania – jest to po prostu tekst (lub inna informacja – np. obraz), w odpowiedzi na który model generuje odpowiedź.
2 Token to najmniejsza jednostka tekstu przetwarzana przez model językowy — może to być całe słowo, jego fragment, znak interpunkcyjny lub symbol.
3 Technicznie rzecz biorąc, model nie przechowuje wiedzy wprost, tylko reprezentacje statystyczne wzorców językowych.

- jest statyczna (nie aktualizuje się samoczynnie – jest aktualna według stanu na moment zakończenia treningu modelu),
- ma charakter ogólny,
- nie obejmuje wewnętrznych dokumentów, procedur czy kontekstu konkretnej firmy.

2. **Treści promptu**

   Prompt, oprócz pytania, może zawierać również dodatkowe informacje – np. fragmenty tekstów wybranych publikacji branżowych. Dostarczony w prompcie tekst, model traktuje jako **kontekst** – informacje, na których powinien oprzeć swoją odpowiedź.

Jeżeli kontekst zawiera informacje umożliwiające poprawną odpowiedź na postawione przez użytkownika pytanie, ryzyko braku odpowiedzi lub halucynacji może zostać znacząco ograniczone. Warto podkreślić, **że im bardziej precyzyjne, aktualne i jednoznaczne są treści (kontekst) zawarte w naszym prompcie, tym większe jest prawdopodobieństwo uzyskania poprawnej odpowiedzi**. Samodzielne wyszukiwanie takich treści oraz ich ręczne wklejanie do promptów jest jednak niepraktyczne i czasochłonne – proces ten można jednak zautomatyzować wykorzystując opisaną poniżej metodę Retrieval-Augmented Generation (RAG).

## 1.2. Czym jest metoda RAG?

Metodę Retrieval-Augmented Generation (RAG) najlepiej wytłumaczyć na poniższej analogii.

*Nauczyciel zadał uczniowi pytanie: „Co jadły dinozaury?" Uczeń w tym momencie nie miał informacji, dzięki którym mógłby udzielić poprawnej odpowiedzi. Mógłby co prawda zaryzykować i zgadywać – ale odpowiedź mogłaby być błędna. Postanowił więc, że pójdzie do biblioteki.*

*W bibliotece uczeń powiedział bibliotekarzowi, że szuka książek w których mógłby znaleźć informacje o tym co jadły dinozaury. Bibliotekarz nie znał treści wszystkich książek na pamięć, ale miał bardzo dobrze przygotowany katalog, w którym każda książka była opisana w sposób oddający jej treść.*

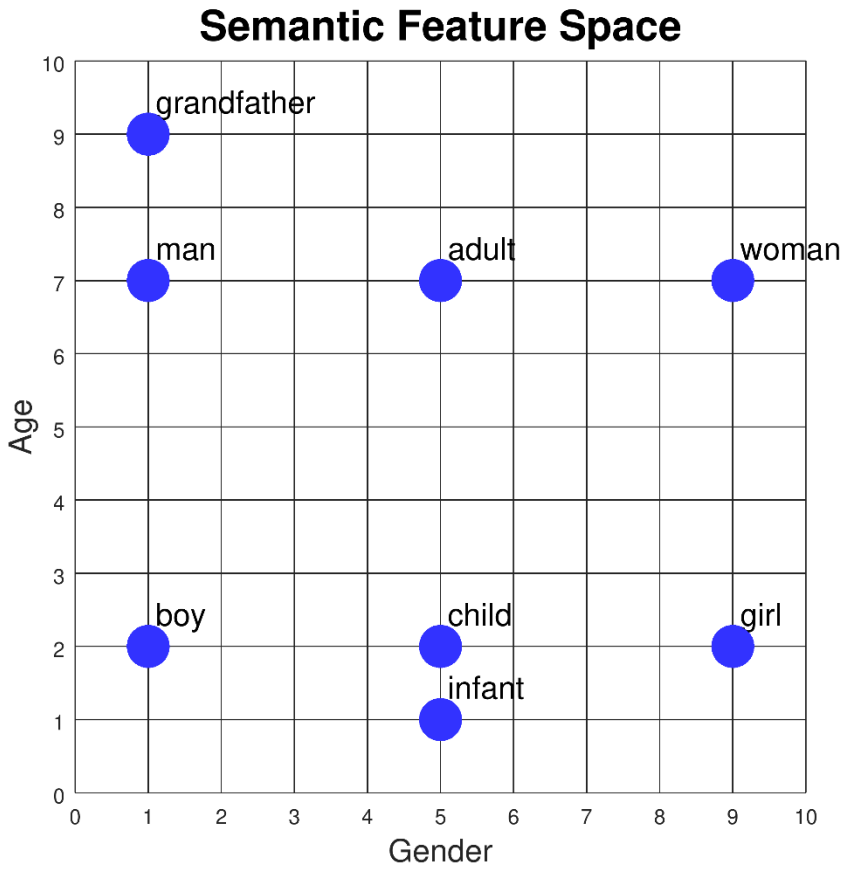
*Na podstawie pytania ucznia bibliotekarz uznał, że poszukiwane informacje mogą być związane z tematami takimi jak: dinozaury, gady mezozoiczne i prehistoria. Bibliotekarz wyszukał w katalogu 5 książek najbardziej związanych z tymi tematami i przyniósł je uczniowi. Po ich przeczytaniu, uczeń udzielił poprawnej odpowiedzi na zadane mu pytanie.*

Powyższa analogia pozwala intuicyjnie zrozumieć ideę RAG. Poniżej opisano, jak ten mechanizm realizowany jest w praktyce. W przedstawionej historii, **nauczyciel** jest użytkownikiem, a **uczeń** jest modelem językowym. Model mógłby spróbować wygenerować odpowiedź na postawione pytanie, jednak bez odpowiedniego kontekstu, jego odpowiedź mogłaby być niepoprawna (odpowiedź mogłaby być halucynacją). W metodzie RAG, kluczowe jest dołączenie do pytania odpowiedniego kontekstu – informacji, które umożliwiłyby udzielenie odpowiedzi na nasze pytanie.

W tym celu tworzona jest baza wiedzy w formie bazy danych (przedstawiona w powyższej historii jako **biblioteka**). Stworzenie i utrzymanie bazy danych jest kluczowym zadaniem w budowie agentów AI – model LLM musi mieć dostęp do wartościowej i uporządkowanej bazy wiedzy. Baza powinna zawierać uporządkowaną wiedzę w formie jednostek tekstu – tekst powinien być oczyszczony z niepotrzebnych znaków, podzielony na określone fragmenty (np.

akapity) i logicznie uporządkowany (np. z zachowaną informacją o strukturze rozdziałowej). Tak przygotowane jednostki tekstu należy zwektoryzować – z wykorzystaniem modelu embeddingowego, oblicza się ciąg liczb, które oddają znaczenie każdej kolejnej jednostki tekstu. Modele embeddingowe charakteryzują znaczenie jednostek tekstu w kilkuset lub kilku tysiącach wymiarów (w zależności od zastosowanego modelu). Dla zrozumienia mechanizmu działania takiego modelu, załóżmy, że korzystamy z modelu embedingowego obliczającego wartość tylko dla dwóch wymiarów: gender (płeć) i age (wiek). Wizualizacja efektów pracy takiego modelu widoczna jest na poniższej ilustracji. Wektor dla granfather ma wartość [1,9]; dla child [5,2]. Jest to wyłącznie uproszczony przykład ilustracyjny — w praktyce modele embeddingowe nie operują na tak prostych i nazwanych wymiarach.

**Rysunek 1 Jak embedding koduje znaczenie słów?**




Źródło: https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/tutorial.html

Jak zauważono wcześniej, w praktyce, model embeddingowy ocenia związek zadanej jednostki tekstu z wieloma tysiącami tematów. Informacje te zapisywane są w bazie danych w osobnej kolumnie (w powyższej historii, funkcje tą pełni **katalog**). Co należy podkreślić, praca ta nie jest związana bezpośrednio z pojedynczym pytaniem wysłanym do modelu – baza wiedzy jest przygotowywana i utrzymywana niezależnie od wysyłanych do modelu pytań (baza wiedzy jest niezależna od modelu językowego).

W momencie zadania pytania, zanim trafi ono do modelu językowego, treść pytania jest wektoryzowana (analogicznie, jak jednostki tekstu z bazy wiedzy). Następnie stworzone w tym celu narzędzie (**bibliotekarz**) szuka w bazie danych określoną liczbę wierszy (np. 5),

których wektory są najbardziej zbliżone znaczeniowo do wektoru naszego pytania. W dalszej kolejności, treść jednostek tekstu (**książek**) związanych z tymi wektorami dołączana jest do treści pytania i jest wysyłana do modelu językowego (**ucznia**), dzięki czemu zwiększamy prawdopodobieństwo, że udzielona odpowiedź będzie poprawna.

## 1.3. Jakie są kluczowe aspekty działania agentów AI na infrastrukturze lokalnej w porównaniu do rozwiązań chmurowych?

Stworzenie narzędzia (agenta AI) wykorzystującego model LLM w metodzie Retrieval-Augmented Generation (RAG) wymaga odpowiedniej infrastruktury obliczeniowej. Model językowy, model embeddingowy, baza wiedzy oraz mechanizmy wyszukiwania kontekstu muszą działać w środowisku zapewniającym wystarczającą moc obliczeniową, dostępność oraz bezpieczeństwo danych.

W praktyce organizacje stoją przed podstawowym wyborem:

- **uruchomienie agenta AI na infrastrukturze lokalnej** – zakupionej i utrzymywanej przez organizację, zlokalizowanej w przestrzeni przez nią zarządzanej i kontrolowanej;
- **skorzystanie z rozwiązań chmurowych** - infrastruktury wynajmowanej od zewnętrznego dostawcy, stanowiącej jego własność i zarządzanej w jego środowisku.

Oba podejścia umożliwiają tworzenie agentów AI, jednak warto mieć świadomość uwarunkowań z nimi związanych. W dalszej części rozdziału skupimy się na trzech kluczowych obszarach: koszcie, bezpieczeństwie oraz odpowiedzialności za wynik pracy agenta[4].

### Koszt

W kontekście generatywnej AI - w tym modeli LLM - należy jasno powiedzieć, że: **nie istnieje coś takiego jak darmowa generatywna sztuczna inteligencja**.

Model językowy może być udostępniany bez opłat licencyjnych (np. jako open source), jednak jego działanie zawsze wymaga infrastruktury obliczeniowej oraz energii, których wykorzystanie wiąże się z realnym kosztem. Część modeli LLM dostępna jest wyłącznie w formie usług chmurowych i nie może zostać uruchomiona lokalnie. Jednocześnie liczba modeli open source jest znacząca i stale rośnie.  Na publicznych platformach (np. Hugging Face) dostępna jest bardzo duża liczba modeli dedykowanych generowaniu tekstu, obejmująca zarówno modele bazowe, jak i ich liczne warianty[5].

4 Nie jest to zamknięta lista aspektów, które należy brać pod uwagę przy projektowaniu agentów AI.
5 Stan na dzień 30.01.2026: 321 071; https://huggingface.co/models?pipeline_tag=text-generation

W przypadku udostępnienia modelu LLM jako open source model taki może pracować zarówno na infrastrukturze lokalnej organizacji, jak również w chmurze. Choć niektóre z modeli mogą być uruchamiane nawet na standardowych komputerach, to ich możliwości są często ograniczone[6]. Tworzenie agentów AI zdolnych do udzielania poprawnych, szczegółowych i użytecznych odpowiedzi w praktycznych zastosowaniach wymaga infrastruktury dedykowanej pracy takich modeli.

W zależności od wybranego podejścia, różni się sposób ponoszenia kosztów:

- **w modelu lokalnym organizacja inwestuje w zakup i utrzymanie infrastruktury** - serwerów (w tym m.in. akceleratory, GPU) oraz energii elektrycznej. Jest to w dużej mierze koszt stały, niezależny od liczby zapytań użytkowników, ale wymagający istotnej inwestycji początkowej. Koszt energii jest kosztem zmiennym, zależnym od intensywności wykorzystania systemu.
- **w modelu chmurowym koszt ma charakter zmienny i jest zwykle uzależniony od zużycia** — w uproszczeniu od liczby przetwarzanych tokenów, a więc liczby i długości zapytań. Rodzi to ryzyko nieprzewidywalnych kosztów, których skala może być trudna do oszacowania. Jednym ze sposobów kontroli wydatków jest wprowadzenie limitów zużycia, co jednak może prowadzić do czasowej niedostępności narzędzia dla użytkowników.

Trudno jest jednoznacznie określić, które podejście jest bardziej opłacalne – koszt w obydwu podejściach jest zależny od wielu zmiennych (m.in. liczby użytkowników modelu w organizacji) i jest zmienny w czasie (m.in. w zależności od zmian cen infrastruktury oraz konkurencji cenowej dostawców usług chmurowych). Warto jednak mieć świadomość, że koszt usług chmurowych, przy dużej liczbie użytkowników, w długiej perspektywie może być znaczący. Zakładając koszt 80 zł za jedną licencję na miesiąc, w przypadku 500 użytkowników, w perspektywie 3 lat łączny koszt wynosiłby 1,44 mln zł.

Należy również podkreślić, że jeżeli organizacja nie ponosi żadnego bezpośredniego kosztu finansowego korzystania z modelu LLM, to w wielu przypadkach ponosi koszt pośredni — w postaci danych, które mogą być wykorzystywane przez dostawcę usługi.

## Bezpieczeństwo

Agent AI wykorzystujący model LLM w metodzie Retrieval-Augmented Generation (RAG), co do zasady, jest narzędziem, któremu udostępniane są dane. W większości przypadków są to dane niepubliczne — informacje, które stanowią własność organizacji lub za które organizacja ponosi odpowiedzialność. Przykładowo, aby model LLM mógł analizować treść umów zawartych przez organizację, musi mieć do nich bezpośredni dostęp.

W tym kontekście kluczowym zagadnieniem staje się bezpieczeństwo danych, w szczególności danych osobowych oraz informacji poufnych. W przypadku modelu LLM działającego na infrastrukturze lokalnej dane pozostają w środowisku organizacji i nie opuszczają jej infrastruktury. Organizacja nie przekazuje wówczas odpowiedzialności za bezpieczeństwo danych podmiotom trzecim.

6 Mały model językowy można uruchomić na zwykłym komputerze bez specjalistycznej karty graficznej, ale generowanie odpowiedzi będzie wtedy wyraźnie wolniejsze. Wbudowany w nowsze komputery układ NPU może to działanie przyspieszyć i zmniejszyć zużycie energii, jednak nie zastąpi wydajnych serwerów przy większych modelach lub większej liczbie użytkowników. Praca mniejszych modeli językowych może też wiązać się z większym ryzykiem halucynacji.

W przypadku rozwiązań chmurowych dane organizacji są przetwarzane na infrastrukturze należącej do zewnętrznego dostawcy, co wiąże się z ograniczeniem bezpośredniej kontroli nad nimi. W szczególności:

- dane mogą być przetwarzane na infrastrukturze zlokalizowanej w innym kraju, w tym poza Unią Europejską;
- organizacja nie ma pełnej kontroli nad sposobem wykorzystania przekazywanych danych — niezależnie od zapisów umownych istnieje ryzyko ich użycia np. do trenowania modeli;
- organizacja nie ma pełnej kontroli nad listą podmiotów mających dostęp do przekazywanych danych, w tym w sytuacjach, gdy dostęp do nich może zostać wymuszony przez organy państwowe kraju, w którym fizycznie zlokalizowana jest infrastruktura.

Budowa agentów AI na infrastrukturze lokalnej zapewnia również większą niezależność od decyzji podejmowanych poza organizacją. Lokalny agent AI jest mniej podatny na nagłe zmiany cen usług chmurowych, ograniczenia dostępności czy decyzje regulacyjne i polityczne, które mogą wpływać na ciągłość działania systemu.

## Odpowiedzialność za decyzje

Niezależnie od tego, czy agent AI działa na infrastrukturze lokalnej, czy w chmurze, kluczowym aspektem jego wykorzystania pozostaje odpowiedzialność za decyzje podejmowane na podstawie jego pracy. Agent AI może znacząco przyspieszyć oraz zautomatyzować proces analizy danych, a także dostarczyć wartościowych rekomendacji, jednak odpowiedzialność za ostateczną decyzję zawsze spoczywa na człowieku.

Załóżmy sytuację, w której agent AI ma za zadanie oceniać wnioski o dofinansowanie. Agent może przeprowadzić analizę wniosku oraz zaproponować ocenę wraz z jej uzasadnieniem, jednak sama decyzja powinna pozostać po stronie człowieka. Może on oprzeć się na wyniku pracy agenta, lecz to właśnie człowiek będzie zobowiązany do uzasadnienia decyzji — zarówno wobec wnioskodawcy, jak i instytucji kontrolnych.

W przypadku błędu odpowiedzialność oraz jego konsekwencje zawsze ponosi człowiek lub organizacja, a nie agent AI. Przenoszenie odpowiedzialności na „decyzję algorytmu”, którego działania nie da się w pełni wytłumaczyć, może stanowić istotne ryzyko prawne i organizacyjne. Z tego względu **agenci AI powinni być projektowani jako systemy wspierające proces decyzyjny, a nie jako rozwiązania, które go całkowicie zastępują**.

# 2. Pilotaż

Pilotaż przeprowadzono w sierpniu 2025 r. Prace realizowano na lokalnej infrastrukturze.

## 2.1. Cele

Podstawowym celem naszego pilotażu było zweryfikowanie możliwości stworzenia działającego lokalnie agenta AI wykorzystującego model LLM w metodzie Retrieval-Augmented Generation (RAG). Dodatkowo pilotaż miał na celu:

- ocenę jakości i użyteczności agentów AI w praktycznym zastosowaniu;
- zweryfikowanie wymagań infrastrukturalnych pod kątem przyszłych wdrożeń agentów AI;
- nabycie kompetencji w zakresie projektowania i budowy agentów AI.

## 2.2. Założenia

Założyliśmy stworzenie agenta AI pełniącego jasno określoną, ograniczoną rolę. Zadaniem agenta było udzielanie odpowiedzi na pytania dotyczące informacji zawartych w intranecie PARP, w szczególności w obszarach takich jak:

- sprawy administracyjne;
- sprawy informatyczne;
- sprawy kadrowe (w tym m.in. informacje o organizacji pracy, benefitach, bezpieczeństwie);
- informacje o pracownikach.

Na wybór zadania postawionego przed agentem wpływ miał łatwy dostęp do danych oraz potencjalna duża użyteczność rozwiązania dla szerokiej grupy pracowników PARP (pozwoliło to na zaangażowanie większej liczby użytkowników w testowanie narzędzia).

Założyliśmy, że agent zostanie oparty na modelach dostępnych open source. W pilotażu wykorzystaliśmy model embeddingowy **Qwen3-Embedding-8B**[7] oraz model LLM **Qwen3-32B**[8]. Obydwa modele zostały zainstalowane na lokalnej infrastrukturze.

## 2.3. Baza wiedzy

Baza wiedzy zbudowana na potrzeby pilotażu miała postać pojedynczej tabeli bazy danych, liczącej około 1000 rekordów, z których każdy odpowiadał jednemu artykułowi z intranetu.

Na potrzeby budowy bazy wiedzy utworzono dedykowaną bazę danych **PostgreSQL**. Baza została rozszerzona o dodatek **VectorChord**[9], który umożliwia przechowywanie embeddingów w postaci wektorów oraz wykonywanie operacji wyszukiwania podobieństwa wektorowego. Mechanizm ten pozwala m.in. porównywać wektor odpowiadający pytaniu wysłanemu do modelu LLM z wektorami reprezentującymi jednostki tekstu zapisane w bazie wiedzy. Instalacja odpowiedniego rozszerzenia (takiego jak VectorChord lub inne narzędzie

7 https://huggingface.co/Qwen/Qwen3-Embedding-8B
8 https://huggingface.co/Qwen/Qwen3-32B
9 https://vectorchord.ai/

do obsługi wektorów) umożliwia sprawną realizację operacji wektorowych, które są kluczowe dla wyszukiwania semantycznego w metodzie RAG.

Bazę wiedzy wypełniono danymi pochodzącymi z intranetu, uzupełnionymi o embeddingi obliczone dla każdej jednostki tekstu (artykułu intranetowego). Surowe dane zostały pobrane i zapisane w postaci pliku .xlsx[10], a następnie zaimportowane do Python. Z wykorzystaniem modelu **Qwen3-Embedding-8B** dla każdej jednostki tekstu obliczono embedding w postaci wektora o 4096 wymiarach, opisującego jej znaczenie semantyczne (każda jednostka tekstu została scharakteryzowana w 4096 wymiarach). Tak przygotowane dane zostały następnie zapisane w bazie danych przy użyciu biblioteki **SQLAlchemy**[11].

Baza wiedzy przygotowana na potrzeby pilotażu była statyczna – dane zostały pobrane jednorazowo i nie były później aktualizowane. Docelowe rozwiązanie powinno zostać uzupełnione o mechanizm weryfikacji i aktualizacji bazy wiedzy (usuwania treści nieaktualnych oraz aktualizacji treści, które się zmieniły).

## 2.4. Narzędzie

Narzędzie zostało przygotowane w dwóch wariantach:

- w postaci skryptu w języku **Python**,
- jako aplikacja udostępniona poprzez **Open WebUI**.

W obu wariantach zastosowano ten sam mechanizm wyszukiwania w bazie wiedzy kontekstu dla zadanego pytania. Mechanizm został opracowany jako funkcja w Pythonie. Dla pytania wprowadzonego przez użytkownika funkcja oblicza jego embedding z wykorzystaniem modelu **Qwen3-Embedding-8B**.

Następnie, przy użyciu biblioteki **SQLAlchemy**, do bazy danych wysyłane jest zapytanie SQL typu SELECT, którego celem jest zwrócenie określonej liczby jednostek tekstu (domyślnie 5), których wektory są najbardziej zbliżone do wektora obliczonego dla pytania. Wyszukane jednostki tekstu stanowią kontekst, który uzupełnia treść pytania wysyłanego do modelu LLM — funkcja zwraca tekst, a nie wektory.

Warto zauważyć, że zapytanie kierowane do bazy danych ma postać standardowego zapytania SQL i może być łatwo dostosowane do własnych potrzeb, na przykład poprzez zmianę liczby zwracanych jednostek tekstu (np. 10 zamiast 5).

### Skrypt w języku Python

W ramach skryptu stworzono funkcję, która dla zadanego pytania:

- wyszukuje odpowiadający mu kontekst (z wykorzystaniem mechanizmu opisanego powyżej),
- przekazuje do modelu LLM Qwen3-32B[12] treść pytania wraz z wyszukanym kontekstem,
- zwraca odpowiedź modelu oraz treść wyszukanego kontekstu.

10 Istnieje możliwość pobrania danych bezpośrednio z bazy danych zawierających treści z intranetu. Wymaga to jednak złożenia i rozpatrzenia wniosku o dostęp. Uwzględniając harmonogram realizacji pilotażu (3 tygodnie w sezonie urlopowym) zdecydowano o samodzielnym pobraniu danych z intranet i zapisaniu ich w pliku .xlsx.
11 https://www.sqlalchemy.org/
12 https://huggingface.co/Qwen/Qwen3-32B

Taka konstrukcja zapewnia, że do modelu językowego trafiają jednocześnie pytanie użytkownika oraz powiązany z nim kontekst. Ponadto możliwe jest wcześniejsze przygotowanie listy pytań i uruchomienie narzędzia w pętli, co pozwala na automatyczne wygenerowanie odpowiedzi dla całego zestawu zapytań.

Rozwiązanie w postaci skryptu oferuje jednak ograniczone możliwości w zakresie pracy wielu użytkowników oraz kontroli dostępu. Brakuje w nim m.in. gotowych mechanizmów uwierzytelniania i zarządzania użytkownikami, co ogranicza jego zastosowanie w środowisku produkcyjnym.

## Aplikacja udostępniona poprzez Open WebUi

Open WebUI[13] to open source'owy interfejs webowy umożliwiający interakcję z modelami językowymi uruchomionymi na infrastrukturze lokalnej lub w chmurze. Użytkownik korzysta z niego w formie znanej z popularnych narzędzi typu ChatGPT — po uruchomieniu przeglądarki internetowej dostępne jest okno czatu, w którym można wpisać prompt, a następnie otrzymać wygenerowaną odpowiedź. W przypadku korzystania z modeli uruchomionych lokalnie żadne dane nie są przesyłane poza infrastrukturę organizacji.

W ramach pilotażu, Open WebUI było dostępne dla pracowników PARP po zalogowaniu się z wykorzystaniem standardowych poświadczeń. Użytkownicy mogli zadawać pytania do modelu językowego **Qwen3-32B**, bez konieczności znajomości technicznych szczegółów jego działania ani sposobu wyszukiwania kontekstu w bazie wiedzy.

W rozwiązaniu zastosowano **Model Context Protocol (MCP)** [14]. MCP umożliwia określenie, z jakich narzędzi model może korzystać w trakcie przetwarzania zapytania użytkownika (np. czy może korzystać z przygotowanych narzędzi do przeszukiwania wskazanych baz danych lub przeglądarki internetowej). MCP można porównać do regulaminu współpracy między pracownikiem a działami wspierającymi. Określa on, do kogo pracownik może się zwrócić po informacje i w jaki sposób powinien to zrobić. Sam nie dostarcza wiedzy, ale umożliwia jej pozyskanie w uporządkowany sposób. Model językowy, na podstawie treści promptu, identyfikował potrzebę pozyskania dodatkowego kontekstu i korzystał z (opisanego powyżej) narzędzia do wyszukiwania kontekstu, które zostało mu udostępnione za pośrednictwem MCP.

Należy jednak zaznaczyć, że mechanizm ten działał poprawnie w większości przypadków, lecz nie zawsze. Zdarzały się sytuacje, w których model językowy nie rozpoznawał potrzeby wyszukania kontekstu i generował odpowiedź wyłącznie na podstawie własnej wiedzy. Aby mieć pewność, że odpowiedź zostanie wygenerowana w oparciu o bazę wiedzy, konieczne było jednoznaczne wskazanie tego w treści promptu (np. „Ekspres do kawy nie działa. Co zrobić? Odpowiedz na podstawie kontekstu z intranetu.").

13 https://openwebui.com/
14 https://modelcontextprotocol.io

## 2.5. Efekty pracy

Treść udzielanych przez agenta odpowiedzi była merytorycznie poprawna. Agent odpowiadał po Polsku, przy czym widoczne były drobne błędy językowe. Czas odpowiedzi na każde pytanie wynosił ok 1-2 minuty.

Poniżej przedstawione są odpowiedzi agenta na dwa przykładowe pytania:

- Ekspres do kawy nie działa. Co zrobić?
- Jak otrzymać zwrot za okulary?

### Pytanie: Ekspres do kawy nie działa. Co zrobić?

W przypadku usterki ekspresu do kawy należy:

1. Nadpisać wiadomość e-mail na adres ***[15], kontaktując się z *** *** (tel. ***, pokój ***).

2. Zarejestrować zgłoszenie w rejestrze usterek dostępna pod adresem: `***\REJESTR_USTEREK\REJESTR_USTEREK_EKSPRESÓW_DO_KAWY.xlsx`.

*** *** odpowiada za obsługę ekspresów do kawy i koordynację działań w takich przypadkach. Inne osoby z kontekstu (np. *** ***, *** ***) nie są odpowiedzialne za te usterki.

Jeśli problem dotyczy innych urządzeń lub systemów IT (np. komputera, sieci), należy kontaktować się z HelpDesk PAR* (tel. ***).

### Pytanie: Jak otrzymać zwrot za okulary?

Aby uzyskać zwrot kosztów zakupu okularów korekcyjnych lub soczewek kontaktowych w PARP, należy postępować zgodnie z poniższymi zasadami:

Warunki przysługiwania refundacji:

1. Eligibilność:
   - Pracownik musi być zatrudniony na stanowisku wyposażonym w monitor ekranowy, który stosuje co najmniej połowę dobowego czasu pracy (np. 4 godziny dziennie przy 8-godzinnym wymiarze).
   - Refundacja przysługuje raz na dwa lata kalendarzowe (24 miesiące od daty akceptacji poprzedniego wniosku).

2. Dokumentacja medyczna:

   - Wymagane są badania okulistyczne przeprowadzone w ramach badań profilaktycznych.
   - Lekarz okulista musi potwierdzić w orzeczeniu lekarskim, że stosowanie okularów lub soczewek jest konieczne do pracy przy monitorze ekranowym.
   - W przypadku pogorszenia wzroku, wymiana okularów wymaga decyzji lekarza okulisty (podczas badań medycyny pracy).

15 Część odpowiedzi udzielonych przez agenta została na potrzeby artykułu zanonimizowana. W oryginalnej odpowiedzi od agenta, w miejscu *** widoczne były poprawne dane.

3. Limit refundacji:

- Maksymalna kwota zwrotu to 700 zł na zakup okularów lub soczewek kontaktowych.

Procedura wnioskowania o refundację:

1. Złożenie wniosku:

- Wypełnij formularz "Wniosek o refundację zakupu okularów korekcyjnych / soczewek kontaktowych" (link do dokumentu: [https://***.parp.gov.pl/...](https://***.parp.gov.pl/images/Aktualnosci/2024/Wniosek_o_refundacj_okularw_korekcyjnych_lub_soczewek_kontaktowych.doc)).
- Załącz paragon za zakup okularów/soczewek oraz orzeczenie lekarza okulisty.

2. Zatwierdzenie przez BZK:

- Wniosek przesyłany jest do Biura Zarządzania Kadrami (BZK), gdzie decyduje dyrektor o akceptacji rozliczenia.

3. Wygoda w przypadku pogorszenia wzroku:

- Jeśli lekarz stwierdzi pogorszenie wzroku w związku z pracą przy ekranie, refundacja może być przyznana nawet przed upływem 24 miesięcy od poprzedniego wniosku.

Kontakt i pomoc:

W sprawach związanych z refundacją okularów należy kontaktować się z:

- *** ***, Specjalistą BZK (pokój ***, tel. ***).

Dla pytań administracyjnych lub problemów z wnioskiem (np. zwrot środków) można skorzystać z pomocy ekspertów Departamentu Kontroli (DK), np.:

- *** *** (tel. ***),
- *** *** (tel. ***),
- *** *** (tel. ***),
- *** *** (tel. ***).

Uwagi dodatkowe:

- W przypadku wymiany szkieł bez zmiany oprawy (np. tylko szkła), refundacja może być przyznana nawet jeśli nie minęły 24 miesiące od poprzedniego wniosku.
- Refundacja obejmuje koszty zakupu, nie pokrywa innych wydatków (np. koszt konsultacji lekarskiej).

# 3. Podsumowanie

**W ramach pilotażu udało się stworzyć działającego lokalnie agenta AI wykorzystującego model LLM w metodzie Retrieval-Augmented Generation (RAG).** Agent korzystał z zadanego modelu LLM i wyszukiwał informacji we wskazanej bazie danych. **Agent AI był dostępny w formie skryptu python oraz poprzez Open WebUi**. Obydwa rozwiązania działały poprawnie, przy czym pierwsze z nich dawało większą kontrolę nad mechanizmem wyszukiwania kontekstu. Narzędzie bazujące na Open WebUi, było bardziej intuicyjne dla użytkowników, jednak zdarzały się sytuacje, w których odpowiedź była generowana bez wcześniejszego wyszukania kontekstu w bazie wiedzy.

W stworzonym agencie AI, zarówno model LLM jak i baza danych mają charakter modułowy – agent działa na wskazanym modelu i wskazanej bazie danych, które można zmieniać. Oznacza to, że ten sam agent może korzystać z innego modelu, jak również może wyszukiwać informacji w innej bazie danych. **Kluczowe znaczenie ma tu przygotowanie oraz utrzymanie bazy danych. Model LLM musi mieć dostęp do wartościowej i uporządkowanej bazy wiedzy**. Bez niej odpowiedzi nie będą związane z kontekstem – istnieje ryzyko, że udzielone odpowiedzi będą niepoprawne.

**W ramach PARP podjęliśmy decyzje o kontynuowaniu prac nad tworzeniem i wdrażaniem agentów AI dedykowanych konkretnym, określonym zadaniom**. Potrzeba tworzenia agenta będzie zgłaszana oddolnie – tak, aby tworzenie nowych agentów AI wynikało z realnych potrzeb pracowników (przed rozpoczęciem prac wdrożeniowych, zgłoszone pomysły będą weryfikowane pod kątem możliwości i zasadności ich realizacji). Uwzględniając modułowość stworzonego rozwiązania, kluczowym jest opracowanie i utrzymanie baz wiedzy na potrzeby danego agenta. Zadanie to będzie realizowane przez poszczególne departamenty merytoryczne, z wykorzystaniem przygotowanych w tym celu narzędzi informatycznych (zapewnienie jakości i aktualności treści w bazach danych będzie zadaniem przypisanym poszczególnym departamentom merytorycznym).

W celu budowy, utrzymania i rozwoju agentów AI, **PARP zakupił dedykowaną infrastrukturę**. Dokumentacja związana ze zrealizowanym zamówieniem jest dostępna pod adresem: https://parp.eb2b.com.pl/open-preview-auction.html/492758/budowa-srodowiska-do-trenowania-modeli-ai-zakup-serwera-gpu