**Dokumentacja aplikacji „Interactive Chat with Store Bot” zrealizowanej w ramach zajęć Inżynierii oprogramowania w semestrze zimowym roku akademickiego 2024/2025**

**Start projektu:** 21.09.2024

**Koniec projektu:** 09.01.2025

**Podział pracy:**

Durawa Jakub - Backend, logika aplikacji, przystosowanie modelu językowego do danych

Bilski Marcel – Frontend, weryfikacja jakości kodu, testowanie aplikacji

Fierek Patryk - Przygotowanie i modeling danych, usprawnienie modelu językowego

Czupryński Michał – Dokumentacja projektu, weryfikacja jakości kodu, testowanie aplikacji

**Model biznesowy**: Pay2Go

**Nazwa: Interactive Chat with Store Bot**

**Link do repozytorium: https://github.com/Durawix/Inzynieria\_Oprogramowania?fbclid=IwZXh0bgNhZW0CMTAAAR26ozdeJ3t3RquGiy8JZqp\_\_qCHJ0-4aeJXTOGAAjnN59hUnNUEU-KF0OA\_aem\_E4TLNByab86WO2mZLdseeg**

**Opis aplikacji**

Aplikacja jest prototypem rewolucjonizującym zakupy w sklepie w dobie ukierunkowania na bezosobową obsługę klienta. Jej główną funkcjonalnością jest chatbot z wirtualnym asystentem sklepu pomagający nam znaleźć produkty, lub dopasować produkty do naszych potrzeb. Doszlifowana aplikacja ma szansę w przyszłości stać się niezwykle pomocna widząc obecne trendy wzrostu popularności pośredników zakupowych takich jak Glovo, Uber czy Lisek. Aplikacja rozwiązuje także problem barier językowych między stroną dostawcy/klienta, a stroną pracownika sklepu.

Wirtualny asystent korzysta z zaawansowanego systemu analizującego treść zapytania użytkownika, co pozwala mu w prosty sposób zrozumieć, czego szukamy, i odpowiednio dopasować produkty. Dzięki połączeniu z bazą ChromaDB - Chroma jest otwarto-źródłową (opensource) bazą danych zaprojektowaną do przechowywania i wyszukiwania osadzeń (embeddings) dla aplikacji wykorzystujących sztuczną inteligencję i modele LLM. Dzięki temu aplikacja korzysta z rozwiązania, które jest nie tylko darmowe, ale także łatwe do dalszego rozwijania i dostosowywania do własnych potrzeb. Kod źródłowy Chroma jest dostępny na platformie GitHub na licencji Apache 2.0, co oznacza, że każdy może z niej korzystać, modyfikować ją oraz swobodnie udostępniać. Baza danych ChromaDB działa lokalnie, co zapewnia pełną kontrolę nad danymi użytkownika.

Interfejs aplikacji został stworzony za pomocą narzędzia Streamlit, dzięki czemu jest prosty w obsłudze i dostępny zarówno na komputerach, jak i urządzeniach mobilnych. W przyszłości aplikacja może być rozwijana o nowe funkcje, takie jak rekomendacje produktów na podstawie wcześniejszych wyszukiwań użytkownika lub integracja z systemami sklepów i usługami dostawczymi, co znacznie zwiększy jej użyteczność.

**Funkcjonalności**

1. **Odczyt danych:**
   * Funkcja read\_data wczytuje dane z pliku Store\_Inventory.xlsx, który zawiera informacje o produktach (takie jak nazwa, kategoria, cena, numer alejki i opis).
2. **Generowanie osadzeń**:
   * Dla każdego produktu z wczytanych danych tworzony jest opis zawierający nazwę produktu, kategorię, nazwę alejki oraz opis. Kod wysyła te dane jako zapytanie do modelu mxbai-embed-large za pomocą biblioteki **ollama**, który generuje wektor osadzeń (embedding). Kod tworzy dokumenty JSON zawierające szczegóły każdego produktu, takie jak nazwa, cena, kategoria i lokalizacja w sklepie, każdy dokument jest dodawany do kolekcji ChromaDB wraz z odpowiadającym mu wektorem osadzeń.
3. **Wyświetlanie wyników i Interfejs (UI)**:

* Aplikacja posiada interfejs użytkownika stworzony w **Streamlit,** gdzie użytkownik może wpisać zapytanie o produkt. Jeśli znaleziono dopasowane produkty, aplikacja wyświetla je w formie tabeli zawierającej szczegóły produktów (nazwa, cena, kategoria, alejka itp.). Pytania mogą być bezpośrednio skierowane w sposób pytający o produk np: Gdzie znajdę wodę Jurajska 0.5L, ale także mające na celu uzyskać sugestie dotyczące produkt np. wpisanie „Brzydko pachnę *– I smell*” do chatboxa zwróci nam produkty higieniczne.

1. **Dodatkowe funkcje**

* Czat działa w sposób interaktywny wyświetlając wyniki w sposób zbliżony do konwersacji dla lepszego doświadczenia z apliakcją.
* Funkcja prevent\_duplicate\_response sprawdza, czy bieżące zapytanie użytkownika jest identyczne z poprzednim, aby uniknąć powtarzania odpowiedzi.

**Wymagania systemowe**

* **System operacyjny**: Windows
* Zainstalowany i skonfigurowany wirtualne środowisko Python3.9 lub nowszy
* Zainstalowane wymagane pakiety Python określone w pliku requirements.txt:

Instalacja odbywa się poprzez komendę: pip install -r requirements.txt

* Chroma – uruchamiana lokalnie na porcie 8000, wskazująca ścieżkę do bazy danych ./embeddings\_db
* **Ollama** – narzędzie do pobierania modeli (ollama pull), dostępne do pobrania z ollama.com/download
* **Streamlit** – framework do uruchamiania aplikacji webowej

**Specyfikacje wymagań**

|  |  |
| --- | --- |
| **Id** | 1. |
| **Nazwa** | Wyświetlenie lokalizacji i informacji o produkcie |
| **Opis** | Aplikacja pozwala wyświetlić lokalizację tj. alejkę i półkę, oraz dodatkowe informacje o produkcie takie jak cenę, kategorie produktu oraz informację czego cena dotyczy np. koszt/paczka |
| **Priorytet** | 1 |

|  |  |
| --- | --- |
| **Id** | 2. |
| **Nazwa** | Dopasowanie produktu do zapytania |
| **Opis** | Aplikacja dopasowuje produkty do zapytań związanych z potrzebami klienta, np. Jestem głodny, Jestem spragniony, Brzydko pachnę itp. |
| **Priorytet** | 1 |

|  |  |
| --- | --- |
| **Id** | 3. |
| **Nazwa** | Dostosowanie formy konwersacyjnej |
| **Opis** | Aplikacja posiada przyjazny interfejs w formie konwersacji z wirtualnym asystentem store-bot co poprawia ogólne doświadczenie korzystania z aplikacji |
| **Priorytet** | 2 |

**Prawa autorskie**

**Autorzy:**

Durawa Jakub

Bilski Marcel

Fierek Patryk

Czupryński Michał

**Warunki licencyjne**

Aplikacja jest udostępniana na licencji **Uznanie autorstwa 4.0 Międzynarodowe (CC BY 4.0)**. Licencja ta umożliwia kopiowanie, modyfikowanie, rozpowszechnianie, prezentowanie oraz wykorzystywanie utworu, pod warunkiem, że autor zostanie odpowiednio wskazany.

https://creativecommons.org/licenses/by/4.0/legalcode