WIZJA PROJEKTU

MEET*UP*

semestr zimowy 2020/2021

Projekt zespołowy czw. 8.25-11.00

Karol Adamiak 237011

Jakub Cembrzyński 235544

Krzysztof Dziuba 236883

Małgorzata Gardziel 237062

1. Opis projektu

Celem projektu jest stworzenie aplikacji do planowania grupowych spotkań. Umożliwiałaby nie tylko wybór daty spotkania, poprzez określenie dostępnych terminów, ale także zaplanowanie atrakcji filmowych oraz dobór zamawianych dań na wieczór. Idea aplikacji opiera się na uwzględnieniu preferencji nie tylko organizatora spotkania, lecz wszystkich uczestników.

Składać się będzie z interfejsu użytkownika, aplikacji webowej, mikroserwisów z elementami uczenia maszynowego, relacyjnej bazy danych oraz silnika wyszukiwania.

2. Rozwiązania konkurencyjne

Doodle, meetingBird, Calendar – aplikacje pozwalające na ustalenie spotkania oraz podgląd kalendarza spotkań. Szczególnie warta uwagi jest aplikacja doodle, pozwalająca na oddanie głosu na wybraną datę wraz z określeniem, czy jest to preferowana data, czy raczej ostatecznie jest się skłonnym na jej akceptację.

Netflix – serwis streamingowy, wykorzystujący AI do określenia rekomendacji dla użytkownika, na podstawie wcześniej wyświetlanych filmów. Co więcej, nawet ikony reklamujące daną pozycję, są dobierane pod kątem użytkownika.

Pyszne.pl - serwis pozwalający zamawiać jedzenie na dowóz z opcją filtrowania restauracji pod kątem preferencji użytkownika. Serwis jednak nie spełnia wymagań naszego projektu. Pyszne.pl nie wspiera grup użytkowników, a filtrowanie i wybór restauracji jest zwykłym filtrowaniem kolekcji bez modułu decyzyjnego opartego na sztucznej inteligencji.

3. Własność produktu

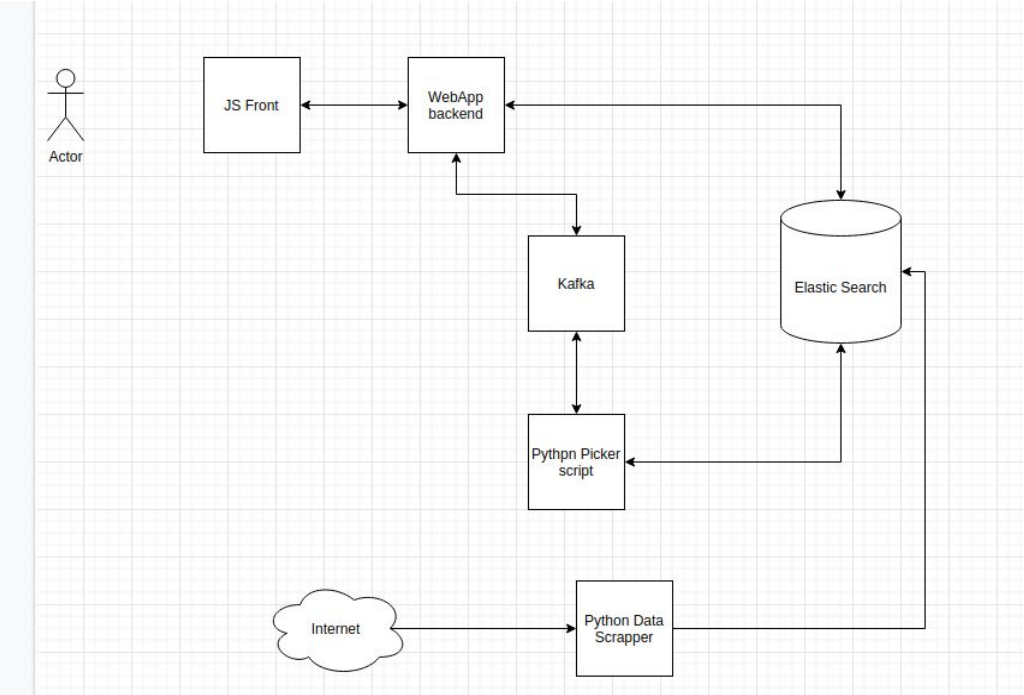
Projektowana usługa, ma oferować jej użytkownikom następujące funkcjonalności:

3.1. Wymagania funkcjonalne

Wymaganie	Problem	Kryterium spełnienia
Proces rejestracji i logowania	Zarządzanie sesją, uprawnieniami i kontekstem.	Użytkownik może założyć konto, zalogować się na nie za pomocą wybranego hasła w trakcie rejestracji. Użytkownik widzi w aplikacji dane, do których powinien dostęp.
Tworzenie grup użytkowników	Spotkanie może być ustalone dla wybranej grupy użytkowników	Użytkownicy mogą tworzyć grupy oraz dodawać do niej nowych członków i usuwać istniejących
Ustalenie daty	Dobór daty spotkania dogodnej dla wszystkich jego uczestników przy różnorodnych preferencjach uczestników	Wybrana data jest dogodna dla jak największej ilości uczestników
Wybór filmu	Wybór filmu przy różnorodnych preferencjach uczestników	Rekomendowane filmy są atrakcyjne dla jak największej ilości uczestników
Wybór dania	Wybór dania na dowóz przy różnorodnych preferencjach uczestników	Zaproponowane dania uwzględniają preferencje użytkowników

3.2. Wymagania niefunkcjonalne

Wymaganie	Problem	Kryterium spełnienia
Dostęp do odpowiednich danych	Podgląd danych innych użytkowników	Użytkownicy mają dostęp tylko do swojego konta oraz do informacji dla nich przeznaczonych
Intuicyjny interfejs	Interfejs użytkownika nie powinien odrzucać potencjalnych klientów trudnością obsługi	Osoba przeciętnie zaznajomiona z aplikacjami webowymi potrafi obsłużyć aplikację bez instrukcji obsługi
Skalowalność	Wzrost użytkowników i obciążenia systemu	System powinien pozwalać skalować się pionowo i poziomo
Mobilność	Użytkownicy korzystający głównie z telefonów i tabletów	Aplikacja nie powinna sprawiać problemów wynikających z korzystania na urządzeniu mobilnym



Rys. 1 - Model ideowy projektowanego systemu

Idea systemu:

- warstwa back-end'u aplikacji stworzona z wykorzystaniem .Net Core, udostępnia REST'owy interfejs, wykorzystywany przez front-end implementowany z wykorzystaniem JavaScript'owego framework'a Vue, w celu komunikacji z użytkownikiem;
- oprogramowanie ElasticSearch (ES) wykorzystywane jest w dwóch celach, pierwszy to przechowywanie danych na temat użytkowników. Back-end, komunikuje się bezpośrednio z ES w procesie tworzenia oraz logowania użytkownika;
- w drugim zastosowaniu wykorzystuje się algorytmy sztucznej inteligencji zawarte w ES w celu pozyskania informacji o filmach oraz restauracjach, jakie może preferować dany użytkownik lub grupa użytkowników. Ten przypadek użycia jest uruchamiany przez skrypt Pythonowy *(Python Picker script, PPs)*, który dobiera odpowiedni model, dodaje parametry i wysyła żądanie wyszukiwania;
- komunikacja między backend'em aplikacji, a PPs przebiega z wykorzystaniem brokera Kafka;
- ostatnim elementem projektowanego systemu jest Python Data Scrapper, który pełni rolę interfejsu pobierającego dane z witryn Webowych, w celu realizacji procesu wyszukiwania prowadzonego przez ES.

4. Harmonogram prac.

- tydzień 1 - stworzenie wizji projektu;
- tydzień 2 - konsultacja z Profesorem, plan podziału pracy oraz szczegółowy harmonogram na 1 etap implementacji;
- tydzień 3 - konfiguracja środowiska developerskiego, stworzenie modelu szkieletowego, stworzenie interfejsów do komunikacji wewnętrznej aplikacji, implementowanie modeli danych;
- tygodnie 4-8 - praca indywidualna członków zespołu, implementacja podstawowych funkcjonalności;
- tydzień 9 - integracja stworzonych algorytmów, testowanie działania;
- tydzień 10 - naniesienie niezbędnych poprawek w celu zagwarantowania poprawnego działania systemu;
- tydzień 11 - dokumentowanie przeprowadzonych prac oraz zastosowań aplikacji;
- tydzień 12 - prezentacja stworzonego serwisu.

5. Podział obowiązków

Moduł	*Architekt rozwiązania*	*Osoby poznające technologię*
Frontend	Jakub Cembrzyński	Krzysztof Dziuba Małgorzata Gardziel
Backend	Małgorzata Gardziel	Karol Adamiak
Machine learning modules	Krzysztof Dziuba	Jakub Cembrzyński
Data scrapper	Karol Adamiak	Krzysztof Dziuba