Career Navigator | Anna Wolańska, Arkadiusz Jaciubek

Raport stanu prac (24.05.2024)

Link do repozytorium:

https://github.com/ajaciubek/issi_proj

Opis projektu

Projekt "Career Navigator" to aplikacja oparta na FastAPI, która umożliwia przewidywanie ról zawodowych na podstawie podanych umiejętności. Wykorzystuje modele uczenia maszynowego do analizy danych i generowania prognoz. Projekt jest skonfigurowany do pracy w środowisku Python 3.12 i integruje się z narzędziami takimi jak DVC, AWS S3 oraz pre-commit.

Struktura projektu

Kluczowe foldery i pliki:

- <u>api</u>: Zawiera logikę aplikacji FastAPI.
 - app.py: Główna aplikacja FastAPI.
 - model.py: Definicje modeli danych dla żądań i odpowiedzi.
- <u>ml</u>: Moduły związane z uczeniem maszynowym.
 - <u>predictor.py</u>: Funkcja <u>predict_job_role</u> do przewidywania ról zawodowych.
 - <u>settings.py</u>: Konfiguracja ścieżek do modeli i innych zasobów.
- data: Przechowuje dane i modele.
 - programming_skills.json: Lista umiejętności programistycznych.
 - Pliki <u>.dvc</u> (np. <u>job_model.keras.dvc</u>): Śledzenie modeli i danych za pomocą DVC.
- config: Pliki konfiguracyjne.
 - .ml-env: Zmienna środowiskowa z konfiguracją modeli.
- <u>test</u>: Testy aplikacji.
 - <u>test_app.http</u>: Testy HTTP dla endpointów FastAPI.

Funkcjonalności

- Endpointy API:
 - GET /: Zwraca komunikat powitalny. W następnym etapie zostanie przygotowany do zwracania aplikacji frontendowej.
 - POST /predict: Przyjmuje listę umiejętności i limit wyników, zwraca przewidywane role zawodowe z prawdopodobieństwami.
- 2. Uczenie maszynowe:
 - Model TensorFlow do przewidywania ról zawodowych.
 - SentenceTransformer do enkodowania tekstu.
 - LabelEncoder do dekodowania etykiet ról zawodowych.
- Zarządzanie danymi:

- DVC do śledzenia wersji danych i modeli.
- AWS S3 jako zdalne repozytorium danych.

Postęp prac

- L Kod aplikacji:
 - Główne funkcjonalności API są zaimplementowane i działają.
 - Moduły uczenia maszynowego są skonfigurowane i zintegrowane z aplikacją.
- 2. Dane i modele:

 - Pliki <u>.dvc</u> wskazują na poprawną konfigurację zdalnego repozytorium S3.
- 3. Testy:
 - Testy HTTP dla endpointów są przygotowane w test_app.http.
- Konfiguracja:
 - Plik <u>pyproject.toml</u> definiuje zależności projektu.
 - Pre-commit hooks są skonfigurowane do analizy kodu (Ruff, Xenon).
- Dokumentacja:
 - Plik <u>README.md</u> zawiera podstawowe instrukcje uruchamiania projektu.

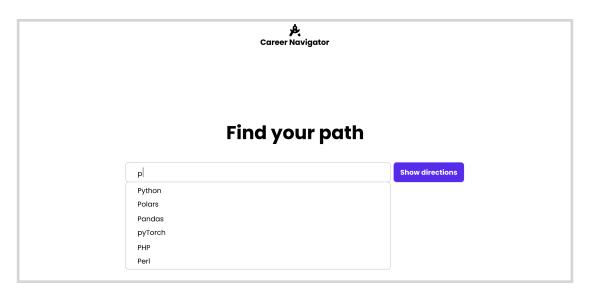
Wymagania systemowe

- Python: Wersja 3.12.
- Narzędzia: Git, AWS CLI, DVC, Uvicorn.

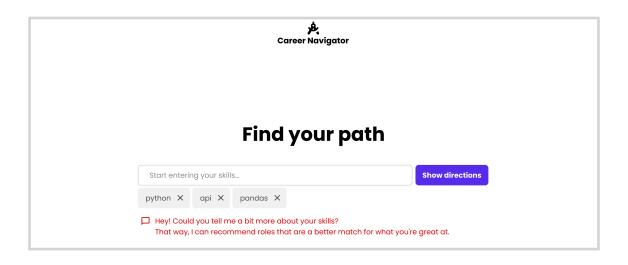
Prace w trakcie

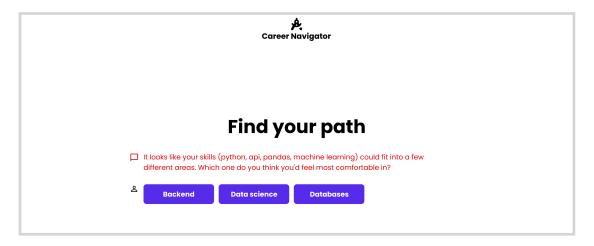
- Backend:
 - Uzgodnienie komunikacji za pośrednictwem API i ostatecznej wersji endpointów.
- 2. Dane i modele:
 - Nowa baza danych oparta na agregatorze ofert pracy Jooble została oczyszczona (po oczyszczeniu dataset zawiera 100k rekordów).
 - Rekordom zostały przypisane etykiety za pomocą LLM.
 - Umiejętności są podzielone na kilka kategorii, każda ma przypisanych kilka pozycji zawodowych.
 - Został przygotowany nowy model, ale konieczny jest fine-tuning dla poprawy rezultatów, eksperymentujemy z pre-processingiem w celu poprawy wyników.
- 3. Frontend:
 - Makiety serwisu zostały zaprojektowane przy użyciu programu Figma. <u>Są dostępne pod linkiem.</u>

^o Została rozpoczęta praca nad frontendową częścią aplikacji.

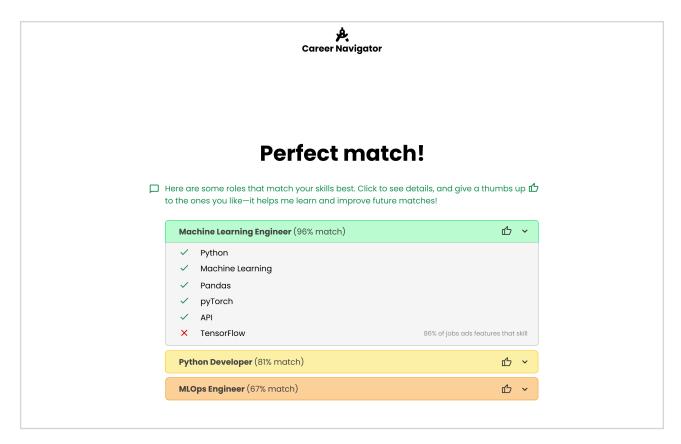


Możliwość wprowadzenia umiejętności (wybór z listy rozwijanej).





Pozyskiwanie informacji od użytkownika w sposób przypominający rozmowę z doradcą zawodowym.



Przedstawienie najlepszych dopasowań wraz z brakującymi umiejętnościami oraz ich popularnością w ofertach.

Następne kroki

- 1. Rozszerzenie testów:
 - Dodanie testów jednostkowych i integracyjnych.
- 2. Optymalizacja modeli:
 - Sprawdzenie wydajności modelu TensorFlow i SentenceTransformer.
- 3. Rozbudowa dokumentacji:
 - Szczegółowe instrukcje konfiguracji i wdrożenia.
- 4. Integracja CI/CD:
 - Rozszerzenie workflow GitHub Actions o testy i wdrożenie.