O'REILLY®

Helion

Interfejsy API w AI i Data Science

Programowanie w Pythonie z użyciem FastAPI

Ryan Day

Tytuł oryginału: Hands-On APIs for AI and Data Science: Python Development with FastAPI

Tłumaczenie: Radosław Meryk

ISBN: 978-83-289-2985-2

© 2025 Helion S.A.

Authorized Polish translation of the English edition of *Hands-On APIs for AI and Data Science* ISBN 9781098164416 © 2025 Ryan Day

This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiejkolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz wydawca dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz wydawca nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
helion.pl/user/opinie/inapiw
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:
https://ftp.helion.pl/przyklady/inapiw.zip

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *helion.pl* (księgarnia internetowa, katalog książek)

Printed in Poland.

- Kup książkę
- Poleć książkę
- Oceń książkę

- Księgarnia internetowa
- Lubię to! » Nasza społeczność

Spis treści

Przedmowa **15**

Część I. Tworzenie interfejsów API do wykorzystania w danologii 21

1. Tworzenie API, które pokochają danolodzy **23**
W jaki sposób danolodzy korzystają z API? 23
Jakich narzędzi używają danolodzy? 24
Projektowanie interfejsów API dla danologów 24
Wprowadzenie do projektu portfolio z części I 26
Każdy interfejs API ma swoją historię 26
Poznaj swoją firmę: SportsWorldCentral 26
SWC potrzebuje interfejsu API 29
Wybór pierwszych produktów API 30
Identyfikacja potencjalnych użytkowników 30
Tworzenie historyjek użytkownika 31
Dodatkowe zasoby 32
Podsumowanie 32

2. Wybór architektury API **34**
Style architektoniczne interfejsu API 34
REST 35
GraphQL 35
gRPC 36
Twój wybór: REST 36
Architektura techniczna 38
Oprogramowanie używane w tym rozdziale 40
Python 40
GitHub 40

Kup książkę Poleć książkę

Pierwsze kroki z GitHub Codespaces 41
Tworzenie konta w serwisie GitHub 41
Klonowanie repozytorium części I 41
Uruchamianie środowiska GitHub Codespaces 42
Przeglądanie nowej przestrzeni kodu 42
Tworzenie pierwszego commita 43
Dodatkowe zasoby 46
Podsumowanie 46

3. Utworzenie bazy danych 47
Składniki Twojego API 47
Oprogramowanie używane w tym rozdziale 48
SQLite 48
SQLAlchemy 49
pytest 49
Tworzenie bazy danych SQLite 50
Tworzenie tabel w bazie danych 50
Struktura tabel 52
Wczytywanie danych 53
Dostęp do danych w Pythonie 54
Instalowanie SQLAlchemy w swoim środowisku 55
Tworzenie plików Pythona na potrzeby obsługi dostępu do bazy danych 56
Tworzenie pliku konfiguracyjnego bazy danych 60
Tworzenie funkcji pomocniczych SQLAlchemy 61
Instalacja pytest 65
Testowanie kodu SQLAlchemy 65
Dodatkowe zasoby 68
Podsumowanie 69

4. Tworzenie kodu FastAPI 70
Rozbudowa projektu portfolio 70
Oprogramowanie używane w tym rozdziale 71
FastAPI 71
HTTPX 72
Pydantic 72
Uvicorn 73
Kopiowanie plików z rozdziału 3. 73
Instalowanie nowych bibliotek w środowisku programistycznym 74
Tworzenie plików Pythona dla interfejsu API 74
Tworzenie schematów Pydantic 74
Tworzenie kontrolera FastAPI 78

Kup książkę Poleć książkę

Testowanie interfejsu API 84
Uruchamianie interfejsu API 87
Dodatkowe zasoby 89
Podsumowanie 89

5. Dokumentowanie interfejsu API 91
Wysyłanie sygnału zaufania 91
Tworzenie świetnych dokumentacji interfejsów API 92
Elementy kluczowe 92
Elementy dodatkowe 93
Przeglądanie przykładów dokumentacji API 94
Aplikacja Sleeper 94
MyFantasyLeague 95
Yahoo! Fantasy Football 96
Przeglądanie wbudowanej dokumentacji API 97
Kopiowanie plików z rozdziału 4. 97
Opcja dokumentacji 1. Swagger UI 98
Opcja dokumentacji 2. Redoc 104
Praca ze specyfikacją OpenAPI 105
Rozbudowa projektu portfolio 107
Dodawanie szczegółów do obiektu info w OAS 108
Dodawanie tagów do kategoryzacji ścieżek 109
Rozszerzanie informacji o poszczególnych punktach końcowych 110
Dodawanie opisów parametrów 110
Wyświetlanie zmian w interfejsie Swagger UI 111
Testy regresji API 112
Aktualizacja pliku README.md 112
Dodatkowe zasoby 114
Podsumowanie 115

6. Wdrażanie interfejsu API w chmurze 116
Korzyści i obowiązki związane z wdrażaniem rozwiązań chmurowych 116
Korzyści 116
Obowiązki 117
Wybór dostawcy chmury dla Twojego projektu 118
Konfigurowanie katalogu projektów 119
Korzystanie z GitHub Codespaces jako hosta w chmurze 119
Wdrażanie w usłudze Render 120
Rejestracja w usłudze Render 121
Tworzenie nowej usługi sieciowej 121
Automatyczne wdrażanie zmian w interfejsie API 123

Kup książkę Poleć książkę

Wdrażanie aplikacji w kontenerze Docker 124
Weryfikowanie instalacji platformy Docker 124
Tworzenie pliku Dockerfile 125
Tworzenie pliku .dockerignore 126
Tworzenie obrazu kontenera 126
Uruchamianie obrazu kontenera lokalnie 127
Wdrażanie do chmury AWS 128
Tworzenie usługi kontenerowej Lightsail 128
Instalowanie interfejsu wiersza poleceń platformy AWS 129
Instalacja wtyczki Amazon Lightsail Container Services 129
Konfiguracja danych logowania 129
Przesyłanie obrazu kontenera do usługi Lightsail 130
Tworzenie wdrożenia w Lightsail 131
Aktualizowanie dokumentacji API 133
Dodatkowe zasoby 134
Podsumowanie 134

7. Baterie w zestawie: tworzenie zestawu SDK języka Python 135
SDK niwelują różnice 136
Wybór języka dla Twojego SDK 139
Zacznij od minimalnej wersji pakietu SDK 140
Wskazówka eksperta. Jak ułatwić instalację pakietu SDK? 140
Wskazówka eksperta. Jak zapewnić spójność i idiomatyczność pakietu SDK? 141
Tworzenie zaawansowanego zestawu SDK 144
Wskazówka eksperta. Używanie rozsądnych ustawień domyślnych 144
Wskazówka eksperta. Zapewnianie rozbudowanej funkcjonalności 146
Wskazówka eksperta. Rejestrowanie zdarzeń 150
Wskazówka eksperta. Ukrywanie skomplikowanych szczegółów interfejsu API 153
Wskazówka eksperta. Obsługa pobierania w trybie masowym 154
Wskazówka eksperta. Dokumentowanie SDK 157
Testowanie pakietu SDK 159
Wskazówka eksperta. Implementacja wszystkich zadań obsługiwanych przez API 162
Dokończenie projektu portfolio w części I 163
Dodatkowe zasoby 165
Podsumowanie 166

Część II. Wykorzystanie API w projektach Data Science 167

8. Co o interfejsach API powinien wiedzieć danolog? 169
Korzystanie z różnych stylów interfejsu API 169
Podstawy protokołu HTTP 171
Jak odpowiedzialnie korzystać z interfejsów API? 172

Kup książkę Poleć książkę

Podział odpowiedzialności. Korzystanie z SDK lub tworzenie klientów API 174
Jak tworzyć interfejsy API? 176
Jak testować interfejsy API? 176
Wdrażanie i konteneryzacja interfejsów API 177
Stosowanie kontroli wersji 177
Wprowadzenie do projektu portfolio — część II 178
Pierwsze kroki z GitHub Codespaces 178
Klonowanie repozytorium części II 178
Uruchamianie środowiska GitHub Codespaces 179
Lokalne uruchomienie API SportsWorldCentral (SWC) 180
Dodatkowe zasoby 181
Podsumowanie 182

9. Korzystanie z interfejsów API do analizy danych 183
Niestandardowe metryki do analityki sportowej 183
Wykorzystanie interfejsów API jako źródeł danych do tworzenia niestandardowych metryk w ligach fantasy 184
Tworzenie niestandardowej metryki. Shark League Score 186
Oprogramowanie wykorzystane w tym rozdziale 187
httpx 187
Notatniki Jupyter 187
pandas 188
Instalowanie nowych bibliotek w środowisku Github Codespaces 188
Uruchamianie interfejsu API w środowisku Codespaces 188
Tworzenie pliku klienta interfejsu API 189
Tworzenie notatnika Jupyter 190
Dodawanie ogólnej konfiguracji do notatnika 191
Praca z danymi API 193
Obliczanie współczynnika zrównoważenia ligi 195
Obliczanie współczynnika soczystości ligi 196
Obliczanie współczynnika ligi rekinów (SLS) 197
Dodatkowe zasoby 198
Podsumowanie 199

10. Używanie interfejsów API w potokach danych 200
Typy źródeł danych dla potoków przetwarzania danych 201
Planowanie potoku danych 201
Orkiestracja potoku danych z użyciem Apache Airflow 202
Instalacja Apache Airflow w środowisku GitHub Codespaces 203
Tworzenie lokalnej analitycznej bazy danych 207
Uruchamianie interfejsu API w środowisku Codespaces 208

Kup książkę Poleć książkę

Konfiguracja połączeń Airflow 208
Tworzenie pierwszego grafu DAG 208
Tworzenie funkcji wspólnej 212
Uruchamianie grafu DAG 213
Podsumowanie 215

11. Wykorzystanie API w aplikacjach danych Streamlit 216
Angażowanie użytkowników za pomocą interaktywnych wizualizacji 216
Oprogramowanie wykorzystane w tym rozdziale 217
nfl_data_py 218
Streamlit 218
Instalacja Streamlit i nfl_data_py 218
Uruchamianie interfejsu API w Codespaces 218
Ponowne wykorzystanie pliku klienta API z rozdziału 9. 219
Tworzenie aplikacji Streamlit 219
Aktualizacja pliku startowego 220
Uruchamianie aplikacji Streamlit 220
Tworzenie strony ze składami drużyn 221
Tworzenie strony ze statystykami zespołów 224
Wdrażanie aplikacji Streamlit 227
Uzupełnianie projektu portfolio — część II 228
Dodatkowe zasoby 229
Podsumowanie 229

Część III. Wykorzystanie interfejsów API w sztucznej inteligencji 231

12. Wykorzystanie API w sztucznej inteligencji 233
Połączenie sztucznej inteligencji i interfejsów API 233
Projektowanie interfejsów API do wykorzystania z generatywną sztuczną inteligencją i modelami językowymi 235
Czym jest sztuczna inteligencja? 237
Generatywna sztuczna inteligencja i duże modele językowe 238
Tworzenie aplikacji wykorzystujących inteligentne agenty AI 238
Wprowadzenie do projektu portfolio — część III 240
Pierwsze kroki z GitHub Codespaces 240
Klonowanie repozytorium części III 240
Uruchamianie środowiska GitHub Codespaces 241
Dodatkowe zasoby 242
Podsumowanie 242

Kup książkę Poleć książkę

13. Wdrażanie interfejsu API na potrzeby uczenia maszynowego 243
Szkolenie modeli uczenia maszynowego 244
Oprogramowanie wykorzystane w tym rozdziale 245
Środowisko wykonawcze ONNX 246
scikit-learn 246
sklearn-onnx 246
Instalowanie nowych bibliotek w środowisku Codespaces 246
Zastosowanie procesu CRISP-DM 247
Zrozumienie potrzeb biznesowych 248
Zrozumienie danych 248
Przygotowanie danych 250
Modelowanie 250
Ocena 253
Wdrażanie 253
Dodatkowe zasoby 260
Podsumowanie 261

14. Korzystanie z interfejsów API z LangChain 262
Wykorzystanie sztucznej inteligencji poprzez API (za pomocą LangChain) 263
Tworzenie agenta LangGraph 264
Rejestracja w usłudze Anthropic 265
Uruchamianie środowiska GitHub Codespaces 266
Instalowanie nowych bibliotek w środowisku GitHub Codespaces 267
Tworzenie notatnika Jupyter 267
Czat z agentem LangGraph 270
Uruchomienie API SportsWorldCentral (SWC) lokalnie 272
Instalacja pakietu SDK biblioteki swcpy 272
Tworzenie zestawu narzędzi LangChain 273
Wywoływanie API z użyciem sztucznej inteligencji (z LangGraph) 276
Rozmowa z Twoim agentem (z użyciem narzędzi) 278
Dodatkowe zasoby 280
Podsumowanie 280

15. Wykorzystanie ChatGPT do wywoływania API 281
Architektura aplikacji 281
Pierwsze kroki z ChatGPT 282
Tworzenie niestandardowego modelu GPT 282
Uruchamianie środowiska GitHub Codespace 286
Uruchamianie interfejsu API SportsWorldCentral (SWC) w środowisku GitHub Codespace 286

Kup książkę Poleć książkę

Dodawanie opisu serwerów do pliku OAS 287
Tworzenie działania GPT 288
Testowanie interfejsów API w Twoim modelu GPT 290
Rozmowy z własnym GPT 292
Uzupełnianie projektu portfolio części III 294
Podsumowanie 295

Kup książkę Poleć książkę

ROZDZIAŁ 12.

Wykorzystanie API w sztucznej inteligencji

Więcej sztucznej inteligencji oznacza więcej interfejsów API.

— Frank Kilcommins, SmartBear

Środowisko osób związanych z techniką czasami postrzega sztuczną inteligencję (AI) i interfejsy programowania aplikacji (API) jako odrębne dziedziny. W gruncie rzeczy są one jednak ściśle ze sobą powiązane, a ich relacja staje się coraz bliższa. W tym rozdziale poznasz obszary, w których AI i API się przenikają, oraz dowiesz się, jakie umiejętności warto rozwijać i jak tworzyć API zgodne z AI. Następnie przygotujesz projekt portfolio do części III, który wykorzystasz w kolejnych rozdziałach.

Połączenie sztucznej inteligencji i interfejsów API

Na początek warto zaznaczyć, że interfejsy API — obok baz danych i plików — są ważnymi źródłami danych wykorzystywanymi do szkolenia modeli sztucznej inteligencji. Powszechnie stosowaną metodą udostępniania modelu użytkownikom po jego przeszkoleniu jest API REST. W rozdziale 13. nauczysz się szkolić model uczenia maszynowego, a następnie wdrożysz go za pomocą API REST.

Narzędzia AI dostępne w chmurze to zaawansowane modele uczenia maszynowego, które są udostępniane poprzez interfejsy API. Narzędzia AI, takie jak mechanizmy generatywnej sztucznej inteligencji, systemy przetwarzania języka naturalnego i inne, często są hostowane w chmurze i udostępniane za pośrednictwem API. W rozdziale 14. nauczysz się, jak za pomocą interfejsu API REST wywołać duży model językowy (LLM) Anthropic.

Nowym obszarem, w którym sztuczna inteligencja łączy się z interfejsami API, jest bezpośrednie wywoływanie API z aplikacji *generatywnej AI*. Aplikacje te powstają na fundamencie modeli LLM i komunikują się z użytkownikami za pomocą języka naturalnego. Jednym z typów takich narzędzi są aplikacje RAG (ang. *retrieval augmented generation* — dosłownie: generowanie uzupełnione wyszukiwaniem). W aplikacji RAG program korzysta z API i innych źródeł danych, a następnie przekazuje uzyskane informacje do modelu LLM razem z zapytaniem użytkownika. Dzięki temu można zniwelować ograniczenia wiedzy modelu LLM, który dysponuje wyłącznie informacjami pochodzącymi z etapu szkolenia.

Kup książkę Poleć książkę

Innym rodzajem aplikacji wykorzystujących generatywną sztuczną inteligencję są rozwiązania wykorzystujące modele LLM do wyszukiwania punktów końcowych API, których należy użyć. W tej książce będziemy je nazywać *aplikacjami agenckimi* (ang. *agentic applications*). Modele LLM podejmują te decyzje w wyniku interpretacji definicji z plików OAS, kodu Pythona lub dokumentacji API. Aplikacje agenckie AI wywołujące interfejsy API z użyciem biblioteki `LangChain` stworzysz w rozdziale 14., a w rozdziale 15. zrobisz to samo z wykorzystaniem ChatGPT.

Perspektywy API. Bill Doerrfield o przenikaniu się sztucznej inteligencji i interfejsów API

Bill Doerrfield jest redaktorem naczelnym w Nordic APIs (*https://nordicapis.com*), międzynarodowej społeczności praktyków i entuzjastów interfejsów API. Bill od ponad dekady zajmuje się tematyką ekonomii API, dzięki czemu posiada szeroką wiedzę na temat wpływu sztucznej inteligencji na rozwój interfejsów API.

Jak sztuczna inteligencja wpłynęła w ostatnich latach na dziedzinę interfejsów API?

Uważam, że rozwój sztucznej inteligencji i rozwój interfejsów API są nierozerwalnie ze sobą związane. To zjawisko nie jest nowe i nie ogranicza się jedynie do ery ChatGPT i modeli LLM. Od lat interfejsy API są kluczowym elementem produktów typu *software-as-a-service*, obejmujących rozpoznawanie obrazów, klasyfikację obiektów, przetwarzanie języka naturalnego, zamianę mowy na tekst i tekstu na mowę oraz platformy chatbotów. Ostatnio, gdy API stały się uniwersalnym standardem wymiany danych nowych usług opartych na sztucznej inteligencji, obserwujemy również wzrost zainteresowania samymi interfejsami API.

Jakie przyszłe trendy dostrzegasz na styku interfejsów programistycznych i sztucznej inteligencji?

Agenty sztucznej inteligencji coraz lepiej rozumieją działanie interfejsów API, a ja postrzegam te interfejsy jako element, który w przyszłości będzie spinał w całość wiele skomplikowanych procesów zachodzących w tle, niewidocznych dla użytkownika. Kolejnym wyzwaniem będzie wyposażenie agentów AI nie tylko w możliwość integracji z danymi i wykonywania zapytań typu `POST`, lecz także w dostęp do zastrzeżonych danych i funkcji, które zazwyczaj podlegają monetyzacji, oraz w zdolność do realizowania transakcji w imieniu użytkownika.

Jakie umiejętności powinni rozwijać programiści i projektanci interfejsów API, wziąwszy pod uwagę rosnące znaczenie sztucznej inteligencji?

Projektanci i programiści interfejsów API powinni przyjąć, że wszystko, co jest publicznie dostępne, zostanie wykorzystane przez sztuczną inteligencję. Agenty AI oparte na technologii RAG mogą całkowicie wyeliminować potrzebę korzystania z API. Ich działanie można porównać do zautomatyzowanych procesów z użyciem techniki *screen scrapingu*. Dlatego należy zadbać o to, aby interfejsy API były bardziej dostępne i precyzyjnie zdefiniowane, co umożliwi modelom LLM ich analizę i naukę, a agentom AI integrację z nimi. W przeciwnym razie programiści sięgną po inne metody dostępu do danych i funkcjonalności.

Programiści powinni śledzić najnowsze trendy w obszarze API, takie jak specyfikacja OpenAPI i jej rozszerzenie Arazzo. Należy rozważyć zastosowanie sztucznej inteligencji do zwiększenia wygody pracy użytkowników API tam, gdzie to jest możliwe. Projektanci API powinni również zadbać o to, aby ich usługi były dostępne w systemach, z których korzystają programiści, na przykład w środowiskach IDE lub asystentach AI.

Kup książkę Poleć książkę

Projektowanie interfejsów API do wykorzystania z generatywną sztuczną inteligencją i modelami językowymi

Jak zatem, zgodnie z zaleceniami Doerrfielda, zaprojektować interfejs API, aby mogła go wykorzystać aplikacja oparta na sztucznej inteligencji generatywnej? Ta dziedzina rozwija się bardzo szybko. Oto kilka wstępnych wskazówek, które mają zastosowanie do `LangChain` (rozdział 14.) i ChatGPT (rozdział 15.).

Najpierw zastanów się, czy korzystanie z interfejsu API lub punktu końcowego bez dodatkowych zabezpieczeń jest właściwe w przypadku aplikacji wykorzystującej generatywną sztuczną inteligencję. Dostawcy modeli LLM ostrzegają, że nie należy ich używać „samodzielnie w sytuacjach wysokiego ryzyka" (Anthropic Claude 3) oraz że „mogą czasami dostarczać niedokładnych informacji" (Google NotebookLM). Dokumentacja ChatGPT zaleca po prostu użytkownikowi „sprawdzanie ważnych informacji".

Naukowcy i inżynierowie zajmujący się uczeniem maszynowym badają dodatkowe metody ograniczania ryzyka związanego z wykorzystaniem modeli LLM do wywoływania interfejsów API i wykonywania innych zadań biznesowych. Niektóre potencjalne zabezpieczenia obejmują wymaganie zatwierdzenia przez człowieka zadań sugerowanych przez modele LLM przed ich wykonaniem, łączenie wielu agentów AI w celu weryfikacji zadań przed ich realizacją, analizowanie i filtrowanie danych wejściowych i wyjściowych modeli oraz przeglądanie logów działania systemu. Ponadto, gdy modele LLM korzystają z interfejsów API, konieczne jest — podobnie jak w przypadku konwencjonalnego oprogramowania — stosowanie podstawowych praktyk związanych z zarządzaniem interfejsami API i bezpieczeństwem. Kluczową praktyką bezpieczeństwa jest ograniczenie uprawnień nadawanych systemom wykorzystującym modele LLM.

Oto kilka wskazówek dotyczących projektowania API. Oparłem je na własnych doświadczeniach oraz na artykule Blobr *Is Your API AI-ready? Our Guidelines and Best Practices* (*https://oreil.ly/jxllA*):

Ogranicz rozmiar zwracanych danych.

Jest to istotne ze względu na koszty i dokładność. Dostawcy modeli pobierają opłaty za przetwarzanie *tokenów*, czyli fragmentów tekstu. Rozmiar tych tokenów może się różnić, ale zasada pozostaje ta sama: im więcej danych model przetwarza, tym wyższe koszty. Przetwarzanie dużych zbiorów danych oprócz tego, że generuje koszty, może wpływać na dokładność modelu. Programiści korzystający z ChatGPT zauważyli, że model ma trudności z wykonywaniem obliczeń na bardzo dużych zbiorach danych zwracanych przez interfejsy API. Jeśli używasz modelu do wykonywania obliczeń, ograniczenie rozmiaru zwracanych danych poprawia jego dokładność.

Istnieje kilka sposobów na osiągnięcie tego celu. Zamiast zwracać wszystkie pola związane z danym obiektem, można zwrócić tylko te najważniejsze. Jeśli API zwraca rekordy

Kup książkę Poleć książkę

podrzędne w kolekcji (np. `product.orders`), warto je wykluczyć z głównych wyników i udostępnić w osobnym punkcie końcowym. Dobrym rozwiązaniem jest także dodanie parametrów, filtrów i stronicowania, aby zawęzić zakres konkretnych rekordów w wywołaniu API.

Zadbaj o spójność struktur danych w całym API.

Im bardziej przewidywalne jest API, tym dokładniej sztuczna inteligencja może z niego korzystać. Wielokrotne wykorzystywanie schematów w API i definiowanie ich w pliku OAS pomaga modelom LLM zrozumieć, czego mogą się spodziewać w wynikach. W API, które stworzyłeś w części I, użyłeś biblioteki Pydantic, która wymusza stosowanie standardowych schematów i publikuje je w pliku OAS.

Udostępnij zestaw narzędzi programistycznych (SDK).

Udostępnianie SDK to sposób na zapewnienie podzbioru punktów końcowych i dostosowanych wywołań API, które są odpowiednie dla aplikacji wykorzystujących sztuczną inteligencję. SDK może również zawierać szczegółowe objaśnienia wywołań API i parametrów, co pomaga modelom językowym w zrozumieniu ich zastosowania. W rozdziale 14. wykorzystasz SDK `swcpy` wraz z bibliotekami `LangChain` i `LangGraph`.

Dostosuj swoją specyfikację OpenAPI (OAS).

Niektóre metody wykorzystania generatywnej sztucznej inteligencji umożliwiają odczytywanie pliku OAS, aby można było wnioskować o punktach końcowych API do wywołania. Możesz utworzyć spersonalizowany plik OAS z punktami końcowymi odpowiednimi dla AI oraz szczegółowymi opisami każdego punktu i jego parametrów, co pomoże modelowi językowemu zrozumieć ich znaczenie. Punkty końcowe w pliku OAS powinny mieć unikatowe i czytelne identyfikatory operacji.

Udostępnij osobny punkt końcowy dla statystyk podsumowujących.

Gdy użytkownicy proszą sztuczną inteligencję o informacje podsumowujące lub liczniki, model językowy może zachowywać się w sposób nieprzewidywalny. Może próbować przeskanować każdy rekord w API, ograniczyć się do identyfikatorów i na ich podstawie oszacować liczbę elementów albo obrać zupełnie inną strategię. Udostępnienie dedykowanych punktów końcowych pozwala ograniczyć to zgadywanie i zwiększa kontrolę nad działaniem modelu.

Udostępnij punkt końcowy wyszukiwania, który nie opiera się na identyfikatorze rekordu.

Modele językowe (LLM) lepiej radzą sobie z językiem naturalnym niż z liczbami. Preferują wyszukiwanie na podstawie informacji, o które użytkownicy prawdopodobnie będą pytać.

Kup książkę Poleć książkę

Użycie specyfikacji Arazzo do definiowania procesów wieloetapowych

Agenty AI są niedeterministyczne, co utrudnia zapewnienie prawidłowego korzystania z interfejsów API, zwłaszcza gdy realizacja procesu biznesowego wymaga wielu wywołań API. Obecnie, aby zachęcić aplikacje AI do łączenia wielu wywołań API, dodaje się informacje do opisów poszczególnych punktów końcowych w pliku OAS lub w komentarzach zamieszczanych w funkcjach narzędziowych. Opisy te mogą wyjaśniać modelom AI relacje między poszczególnymi punktami końcowymi. Jednak to od modelu zależy, czy zostaną one poprawnie wykorzystane.

Nowym trendem, który według Doerrfielda może pomóc sprostać temu wyzwaniu, jest specyfikacja Arazzo (*https://oreil.ly/qXKCU*). Arazzo określa „sekwencje wywołań i ich zależności", co pozwala na wspólne użycie wielu punktów końcowych API w celu osiągnięcia konkretnego celu biznesowego. Zdefiniowany z użyciem specyfikacji Arazzo zestaw powiązanych wywołań może stanowić deterministyczny blok konstrukcyjny wykorzystywany przez aplikacje AI, co zwiększa niezawodność korzystania z API. Arazzo współpracuje z OpenAPI, więc naturalnie pasuje do aplikacji i narzędzi, które obecnie używają specyfikacji OAS.

Warto śledzić rozwój Arazzo, aby zobaczyć, jak platformy AI będą w przyszłości dodawać formalną obsługę tej specyfikacji. Już teraz warto poeksperymentować z modelami LLM, aby sprawdzić, jak można ją wykorzystać.

Czym jest sztuczna inteligencja?

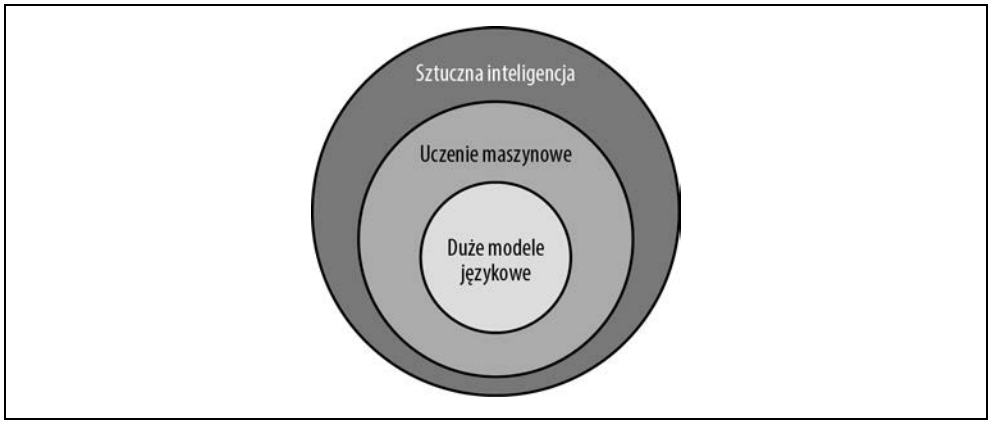
Sztuczna inteligencja to technologia umożliwiająca komputerom i innym urządzeniom naśladowanie ludzkich zdolności, takich jak uczenie się, rozumienie, rozwiązywanie problemów, podejmowanie decyzji, kreatywność i autonomiczne działanie.

> — *What Is Artificial Intelligence (AI)?*, Cole Stryker i Eda Kavlakoglu, IBM Corporation, 2024

Oprócz formalnej definicji sztucznej inteligencji (AI) mamy też współczesną nieformalną, która mówi, że AI to program komputerowy zdolny do prowadzenia rozmów i wykonywania zadań w sposób przypominający człowieka. AI obejmuje *systemy eksperckie*, które istnieją od dziesięcioleci. Są to oparte na regułach złożone systemy, które potrafią wykonywać zadania podobne do ludzkich. Współczesna AI koncentruje się na *uczeniu maszynowym*, w którym badacze kierują procesem szkolenia modeli, ale nie programują ich bezpośrednio. Jednym z głównych zastosowań uczenia maszynowego jest generatywna AI wykorzystująca duże modele językowe (LLM).

Powiązania między tymi pojęciami przedstawiono na rysunku 12.1.

Kup książkę Poleć książkę



Rysunek 12.1. Terminologia związana ze sztuczną inteligencją

Generatywna sztuczna inteligencja i duże modele językowe

Choć uczenie maszynowe znajduje zastosowanie w wielu różnych dziedzinach, w ostatnich latach uwaga opinii publicznej skupiła się głównie na generatywnej sztucznej inteligencji. Zdolność tych aplikacji do tworzenia tekstu, muzyki i filmów na podstawie poleceń tekstowych doprowadziła do szybkiego rozpowszechnienia się takich rozwiązań jak ChatGPT firmy OpenAI, Copilot Microsoftu, Gemini Google oraz wielu dodatków do innych programów.

Mimo imponujących możliwości tych aplikacji używanie generatywnej sztucznej inteligencji wiąże się również z wieloma zagrożeniami i ograniczeniami. Dostawcy niektórych popularnych modeli ostrzegają przed stronniczością, halucynacjami, błędami i szkodliwymi treściami. Są to poważne zagrożenia, które programiści korzystający z tych rozwiązań powinni traktować z należytą uwagą. Więcej szczegółów na temat zagrożeń i ograniczeń modeli przedstawionych w tej części książki zawierają rozdziały 14. i 15.

Tworzenie aplikacji wykorzystujących inteligentne agenty AI

Jak zauważa Doerrfield, w centrum badań i rozwoju sztucznej inteligencji znajdują się agenty AI. Agent to oprogramowanie, które steruje działaniem aplikacji za pomocą modelu językowego (LLM). Im bardziej autonomicznie model językowy steruje systemem, tym bardziej *agencki* staje się ten system.

Tworzenie agentów AI z użyciem dużych modeli językowych to nowa dziedzina, w której pojawiło się wiele narzędzi do budowania pojedynczych agentów lub orkiestracji wielu agentów w celu wykonywania różnych zadań. Kilka otwartych platform do tworzenia agentów i aplikacji opartych na modelach LLM zestawiono w tabeli 12.1.

Kup książkę Poleć książkę

Tabela 12.1. Platformy programistyczne dla agentów AI

Oprogramowanie	Obsługiwane języki programowania
Autogen	Python, .NET
CrewAI	Python
`LangChain/LangGraph`	Python
LlamaIndex	Python, TypeScript
`PydanticAI`	Python
SDK Vercel AI	TypeScript

W rozdziale 14. wykorzystasz LangChain i LangGraph do komunikacji z interfejsami API.

Perspektywy sztucznej inteligencji. Samuel Colvin o agentach AI

Samuel Colvin jest twórcą `Pydantic`, biblioteki open source dla Pythona, którą wykorzystywaliśmy w tej książce do walidacji i serializacji danych. `Pydantic` jest używany w wielu zestawach SDK dostawców modeli oraz w pythonowych frameworkach do tworzenia agentów AI. Rozmawiałem z Colvinem o agentach i jego własnym frameworku `Pydantic` do tworzenia agentów — `PydanticAI`.

Jakie znaczenie mają agenty AI i platformy agenckie w kontekście rozwoju sztucznej inteligencji w najbliższych latach?

Jestem przekonany, że z czasem ludzie przestaną bezpośrednio korzystać z SDK dostarczanych przez firmy oferujące rozwiązania AI. Pojawią się biblioteki oparte na tych zestawach SDK, które będą podejmować przemyślane decyzje dotyczące tego, czego naprawdę potrzebują użytkownicy. Jeśli spojrzymy na frameworki webowe jako analogię, to technicznie rzecz biorąc, nakładają one pewne ograniczenia na to, co można zrobić (przy tworzeniu aplikacji webowych), ale są to ograniczenia, które wszyscy akceptują, ponieważ stosowanie frameworków pozwala pisać kod na znacznie wyższym poziomie abstrakcji. Mam nadzieję, że `PydanticAI` będzie odgrywać w tym procesie istotną rolę.

Jaką rolę, Twoim zdaniem, może odegrać PydanticAI w tworzeniu aplikacji wykorzystujących sztuczną inteligencję?

Mam nadzieję, że `PydanticAI` stanie się standardowym narzędziem do wdrażania sztucznej inteligencji w środowisku produkcyjnym. Wynika to z naszego większego doświadczenia w Pythonie, ale także z intensywnej pracy nad zwiększeniem wygody pracy programistów i wprowadzeniem niezbędnych mechanizmów kontrolnych przy wdrażaniu takich rozwiązań. Na przykład w przypadku `PydanticAI` włożyliśmy ogromny wysiłek w zapewnienie bezpieczeństwa typów, włączając w to system zależności.

Jakich umiejętności, narzędzi lub frameworków programiści powinni uczyć się już teraz, biorąc pod uwagę erę sztucznej inteligencji, w którą właśnie wkraczamy?

Uważam, że warto zrozumieć intuicyjne aspekty działania modeli językowych oraz niuanse ich specyficznego zachowania. Kluczowe znaczenie mają również solidne podstawy programowania w Pythonie. Może pojawić się pokusa, by zrezygnować z nauki podstawowych technik programistycznych, skoro korzystamy z generatywnej sztucznej inteligencji. W rzeczywistości

Kup książkę Poleć książkę

te umiejętności stają się jeszcze bardziej istotne. Praca z systemem niedeterministycznym wymaga, abyśmy mieli absolutną pewność, że nasze podstawowe testy jednostkowe dla elementów deterministycznych są niezawodne.

Colvin zaleca programistom zapoznanie się z dwoma dodatkowymi standardami: OpenTelemetry (*https://oreil.ly/OTEL*) do monitorowania i rejestrowania danych w aplikacjach AI oraz Model Context Protocol (MCP) (*https://oreil.ly/mcptL*) do dostarczania kontekstu dla dużych modeli językowych. OpenTelemetry umożliwia kompleksowe śledzenie działania aplikacji, podczas gdy MCP pozwala na skuteczne przekazywanie do modeli AI informacji kontekstowych.

Wprowadzenie do projektu portfolio — część III

W tej części książki stworzysz projekt portfolio, który zaprezentuje Twoje umiejętności pracy z interfejsami API i sztuczną inteligencją. Oto przegląd zadań, które Cię czekają:

- rozdział 13. — wdrażanie interfejsu API dla uczenia maszynowego,
- rozdział 14. — korzystanie z interfejsów API z LangChain,
- rozdział 15. — wykorzystanie ChatGPT do wywoływania API,

Każde z tych zadań pozwoli Ci w unikatowy sposób zaprezentować swoje umiejętności w zakresie interfejsów API i sztucznej inteligencji.

Pierwsze kroki z GitHub Codespaces

W trzeciej części książki nadal do tworzenia kodu będziesz używać GitHub Codespaces. Jeśli jeszcze nie masz konta na GitHubie, załóż je teraz.

Klonowanie repozytorium części III

Wszystkie przykłady kodu z części III znajdują się w repozytorium GitHub tej książki (*https://github.com/handsonapibook/api-book-part-three*).

Aby sklonować repozytorium, zaloguj się do GitHuba i przejdź na stronę importowania repozytorium (*https://github.com/new/import*). Na tej stronie wprowadź w odpowiednich polach następujące informacje:

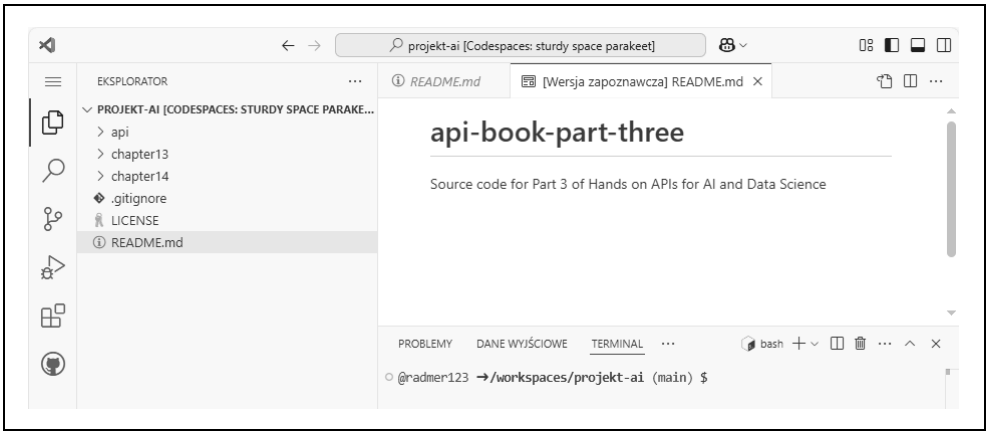
- *The URL for your source repos*: `https://github.com/handsonapibook/api-book-part-three`.
- *Your username for your source code repository*: pozostaw to pole puste.
- *Your access token or password for your source code repository*: pozostaw to pole puste.
- *Repository name*: `projekt-ai`.
- *Public*: zaznacz tę opcję, aby móc udostępnić wyniki swojej pracy.

Kup książkę Poleć książkę

Kliknij *Begin import*. Proces importu rozpocznie się i zobaczysz komunikat *Preparing your new repository*. Po kilku minutach otrzymasz wiadomość e-mail z powiadomieniem o zakończeniu importu. Kliknij link w tej wiadomości, aby przejść do swojego nowo sklonowanego repozytorium.

Uruchamianie środowiska GitHub Codespaces

Na stronie repozytorium kliknij przycisk *Code* i wybierz zakładkę *Codespaces*. Następnie kliknij *Create Codespace on main*. Wyświetli się strona ze statusem *Setting up your codespace*. W trakcie konfiguracji otworzy się okno Twojej przestrzeni Codespace. Po zakończeniu procesu konfiguracji wyświetli się strona podobna do pokazanej na rysunku 12.2.



Rysunek 12.2. Środowisko GitHub Codespaces dla części III

Twoja przestrzeń Codespace została utworzona wraz ze sklonowanym repozytorium. To środowisko będzie wykorzystywane w części III tej książki. Otwórz stronę GitHub Codespaces (*https://oreil.ly/nLbqH*), aby wprowadzić kilka zmian. Przewiń stronę w dół, aby znaleźć nowo utworzoną przestrzeń Codespace, kliknij wielokropek po prawej stronie nazwy i wybierz opcję *Rename*. Wpisz nazwę **`Przestrzeń projektu portfolio części III`** i kliknij *Save*. Powinieneś zobaczyć komunikat *Your codespace Projekt ai has been updated*. Kliknij ponownie wielokropek, a następnie, aby wyłączyć automatyczne usuwanie, kliknij ikonę obok opcji *Auto-delete Codespace*.

Aby zaoszczędzić miejsce na ekranie, w przykładach skróciłem listę katalogów wyświetlaną w wierszu poleceń Codespace. Możesz to zrobić w swoim środowisku Codespace przez edycję pliku */home/codespace/.bashrc* w VS Code. Znajdź wiersz z `export PROMPT_DIRTRIM` i ustaw ją na `export PROMPT_DIRTRIM=1`. Aby załadować nowe ustawienia po raz pierwszy, wykonaj w terminalu polecenie: `source ~/.bashrc`.

Kup książkę Poleć książkę

Dodatkowe zasoby

Aby zapoznać się ze stupunktową kartą oceny kompatybilności API z systemami AI, przeczytaj artykuł Blobr zatytułowany *Is Your API AI-ready? Our Guidelines and Best Practices* (*https://oreil.ly/JLaK1*).

Aby zobaczyć przykład obejścia ograniczeń modelu GPT, przeczytaj artykuł Kade'a Halabuzy'ego *Syntax Sunday: Custom API Wrapper for GPTs* (*https://oreil.ly/khh0J*).

Aby dowiedzieć się więcej o możliwych przyszłych zastosowaniach sztucznej inteligencji i interfejsów API, przeczytaj artykuł Petera Schroedera *AI + APIs — What 12 Experts Think The Future Holds* (*https://oreil.ly/t2lxH*).

Podsumowanie

Ten rozdział zawiera wprowadzenie do podstawowych zagadnień w dziedzinie sztucznej inteligencji i wyjaśnia, jak odnoszą się one do interfejsów API.

W rozdziale 13. stworzysz model uczenia maszynowego i wdrożysz go z użyciem FastAPI.

Kup książkę Poleć książkę

PROGRAM
PARTNERSKI
GRUPY HELION
1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ
Zmień swoją stronę WWW w działający bankomat!
Dowiedz się więcej i dołącz już dzisiaj!
http://program-partnerski.helion.pl
GRUPA
Helion

To lektura obowiązkowa dla każdego, kto chce poszerzyć kompetencje w zakresie danologii!

Eric Eager, wiceprezes do spraw analiz piłkarskich, Carolina Panthers

Jeśli chcesz projektować aplikacje oparte na sztucznej inteligencji, tworzyć dobry kod czy analizować dane, musisz opanować zasady pracy z interfejsami API. To już nie tylko kwestia wysłania prostego zapytania do REST API — niezawodność, bezpieczeństwo i skalowalność wymagają znacznie więcej. Łatwo dojść do wniosku, że zakres potrzebnej wiedzy jest przytłaczający.

Ten przystępny podręcznik jest skierowany do analityków danych i programistów chcących zdobyć doświadczenie w tworzeniu i wdrażaniu interfejsów API z wykorzystaniem Pythona, a także frameworków takich jak FastAPI i Streamlit. Dzięki tej książce nauczysz się budować API od podstaw, integrować je z przepływami danych i używać dużych modeli językowych (LLM) do efektywnej interakcji z interfejsami. W efekcie zyskasz niezbędne przygotowanie, by projektować solidne, nowoczesne rozwiązania API dla branży AI i Data Science.

To nieocenione źródło praktycznej wiedzy o API w Data Science i sztucznej inteligencji — dziedzinach, w których interfejsy API odgrywają coraz większą rolę!

James Gough, wybitny inżynier w Morgan Stanley

W książce:

- zasady projektowania API na potrzeby aplikacji analitycznych
- wdrażanie API z użyciem Pythona i FastAPI
- integracja API z przepływami danych
- wdrażanie modeli uczenia maszynowego jako API
- użycie dużych modeli językowych (LLM) w architekturach API

Ryan Day jest ekspertem danologii w Conference of State Bank Supervisor (CSBS) i doświadczonym programistą. Aktywnie uczestniczy w projekcie FastAPI. Ma doświadczenie w przetwarzaniu w chmurze i tworzeniu API w sektorze federalnym.

Helion
helion.pl
HELION S.A.
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

KOD KORZYŚCI
Sięgnij po więcej!

ISBN 978-83-289-2985-2
9 788328 929852

Cena: 79,00 zł