Dialogflow to platforma od Google do budowania aplikacji rozumiejących język, jakim posługują się ludzie (czyli język naturalny (NLU – Natural Language Understanding)). Narzędzie to wcześniej nosiło nazwę Api.ai i rozwijane było przez firmę Speaktoit. Po przejęciu firmy przez Google w 2016 roku, nazwę zmieniono na Dialogflow, a narzędzie włączone zostało do oferty Google Cloud Platform. Dialogflow obsługuje obecnie 32 języki i dialekty, przede wszystkim europejskie oraz azjatyckie.

Po wejściu na stronę <https://dialogflow.cloud.google.com> utworzono konto, a następnie po zalogowaniu się stworzono nowego Agenta – BotTest.

Agent to moduł NLP, który rozumie rozległe niuanse ludzkiego języka oraz tłumaczy je na standardowe i uporządkowane treści, które są czytelne dla aplikacji. Następnie udziela informacji zwrotnej. Agent czerpie swoją wiedzę i umiejętności rozumienia dzięki mechanizmom z zakresu uczenia maszynowego takim jak **Intents** i **Entities**

Podczas tworzenia agenta pierwszym ważnym zadaniem jest nauczenie go rozumienia tego, co komunikują nam użytkownicy. Kluczową funkcją DialogFlow służącą do tego jest **Intent** - zamiar lub intencja. To dzięki nim program może skutecznie analizować treść rozmowy i rozpoznać potrzeby drugiej strony.



Rysunek 1

Stworzono intencje (rysunek 1) i do każdej z nich dostarczono przykłady tego, jak potencjalni użytkownicy aplikacji mogą komunikować swoje potrzeby. Dzięki trenowaniu modelu uczenia maszynowego, DialogFlow rozpoznaje podane przykłady oraz inne wyrażenia mające podobny sens i znaczenie. Agent dopasowuje wiadomości użytkownika do zbioru stworzonych intencji, przeszukuje „Training phrases” (rysunek 2) a następnie odpowiada za pomocą odpowiedzi pobranej z Responses (rysunek 3).



Rysunek 2



Rysunek 3

Następnie w zakładce „Fulfillment” włączono Webhooka. Webhook to żądanie HTTP (zazwyczaj typu POST) na zadany przez nas adres URL mające na celu poinformowanie nas o tym, iż zadziała się jakaś akcja, zdarzenie. Jest to sposób, w jaki aplikacje mogą komunikować się ze sobą. Umożliwia przesyłanie danych w czasie rzeczywistym z jednej aplikacji do drugiej w momencie wystąpienia zdarzenia. W naszym projekcie chcemy przechwycić DIAGNOSTIC INFO (rysunek 4) (prawy dolny róg zakładki fulfillment), czyli informacje w formacie json.



Rysunek 4

Użyto **ngrok**, czyli aplikacji, która umożliwia tunelowanie ruchu. Działa w następujący sposób: uruchomiona aplikacja ngrok.exe łączy się z chmurą ngrok, która akceptuje ruch na publicznym adresie. Otrzymany ruch z publicznego adresu przekazuje do procesu ngrok.exe uruchomionego na naszym komputerze, który dalej przekazuje owy ruch do lokalnej aplikacji/serwera. W skrócie – ngrok przekierowuje webhook’a do API uruchomionego lokalnie na komputerze.

Adres do pobrania aplikacji: <https://ngrok.com/download>

(Po zainstalowaniu i rozpakowaniu paczki najlepiej otworzyć lokalizację pliku ngrok.exe, włączyć cmd i dopiero wtedy uruchomić)

Po wpisaniu nazwy pliku należy podać adres localhosta (rysunek 5)



Rysunek 5

Następnie wyświetla się status sesji online, kopiujemy adres https (rysunek 6)



Rysunek 6

Powracamy do zakładki Fulfillment, wklejamy skopiowany adres URL (rysunek 7) i zapisujemy (przycisk SAVE – prawy dolny róg)



Rysunek 7

Kolejnym krokiem jest wykorzystanie Flaska - mikro framework aplikacji webowych napisany w języku Python. Do pisania aplikacji wykorzystano środowisko PyCharm.

Celem jest przechwytywanie „POST” (za każdym razem wpisując coś do dialogflow dostajemy POST). Do zmiennej answer zapisano przechwycone żądanie json a następnie wyodrębniono do zmiennych wypowiedź użytkownika, odpowiedź bota oraz nazwę intent.

Po uruchomieniu programu wyświetla się adres z numerem portu (rysunek 8)



Rysunek 8

Powrócono do DIalogflow w celu przetestowania aplikacji (rysunek 9)



Rysunek 9

Następnie powrócono do PyCharma (rysunek 10). Informacje o Userze, Bocie i Intencie zostały wyświetlone.



Rysunek 10

W terminalu również została wyświetlona informacja o poprawnym wykonaniu polecenia POST



Rysunek 11