

Projet 10 : Fly Me

Challenge : Développez un chatbot pour réserver des vacances

https://github.com/blanchonnicolas/IA_Project10_Openclassrooms_Chatbot

Agenda

01

INTRO

Contexte et Objectifs
Présentation du jeu de données
Préparation des données

02

LUIS

Principes d'utilisation
Modélisation
Performance

03

CHATBOT

Spécification fonctionnelle
Test Unitaire
Intégration continue
Monitoring

04

DEMO

Démo
Conclusions

Introduction

Contexte et Objectifs

FlyMe

Fly Me est une **agence** qui propose des **voyages clé en main** pour les particuliers ou les professionnels.

Fly Me a lancé un projet ambitieux de **développement d'un chatbot** pour aider les utilisateurs à choisir une offre de voyage.

Chatbot MVP

La première étape de ce projet est de **construire un MVP** permettant de **réserver un billet d'avion**.

Le Chatbot devra pouvoir poser les **questions pertinentes (en anglais)** à l'utilisateur pour comprendre sa demande.

Le MVP est destiné aux collaborateurs internes de Fly Me afin de tester le concept.

Objectifs de la mission



Construire une solution de Chatbot capable de reformuler la demande de l'utilisateur et lui demander de valider sa compréhension.



Identifier les villes de départ et de destination, les dates aller et retour souhaitées du vol, ainsi que le budget maximum pour le total des billets



Jeu de données Frames.json
Microsoft Bot, reposant sur le Framework SDK v4.
Service cognitif LUIS d'Azure
Web App sur Cloud Azure
Bot Framework Emulator



Automatiser la chaîne de traitement d'intégration et de déploiement, incluant l'exécution de tests unitaires..
Evaluer la performance du modèle LUIS
Analyser l'activité du chatbot en production

Présentation du jeu de données

Compréhension du jeu de données

Fly Me



Fichier frames.json de 67.9Mo

- ☐ 1369 échanges
- ☐ 5 champs principaux

- ☐ Intention recherchée: Book flight
- ☐ Entités à prendre en charge par le Chabot:
 - From
 - To
 - str_date
 - end_date
 - budget

Historique d'échanges entre un chatbot et un utilisateur

Plus riche que ce dont nous avons besoin pour la V1, donc nécessité d'isoler les éléments essentiels au MVP.

Etude exploratoire des données présentes dans le notebook EDA.ipynb

Chiffres clés du jeu de données

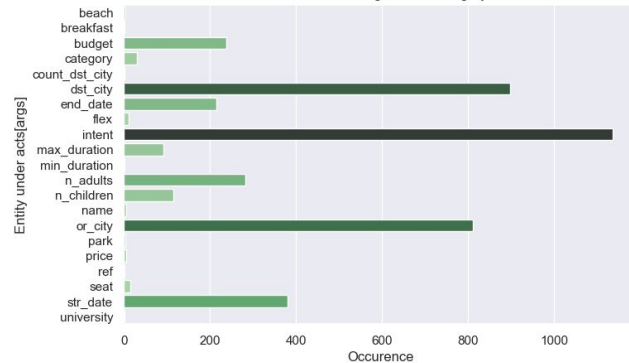
Number of User/Wizard authoring text from dialogue



Name in acts = Intentions

inform 2252
greeting 405
request 15
thankyou 1

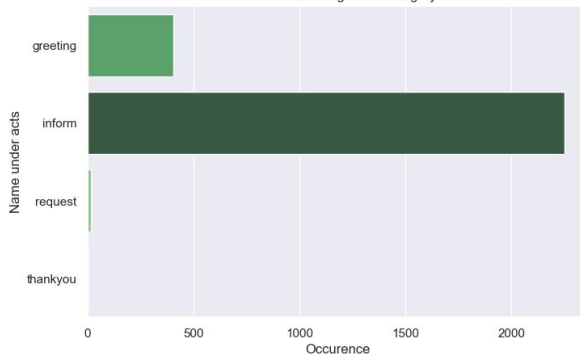
Number of Dialog Entities category



Echanges :

turns = 1369
user = 10407
wizard = 9579

Number of Dialog intent category

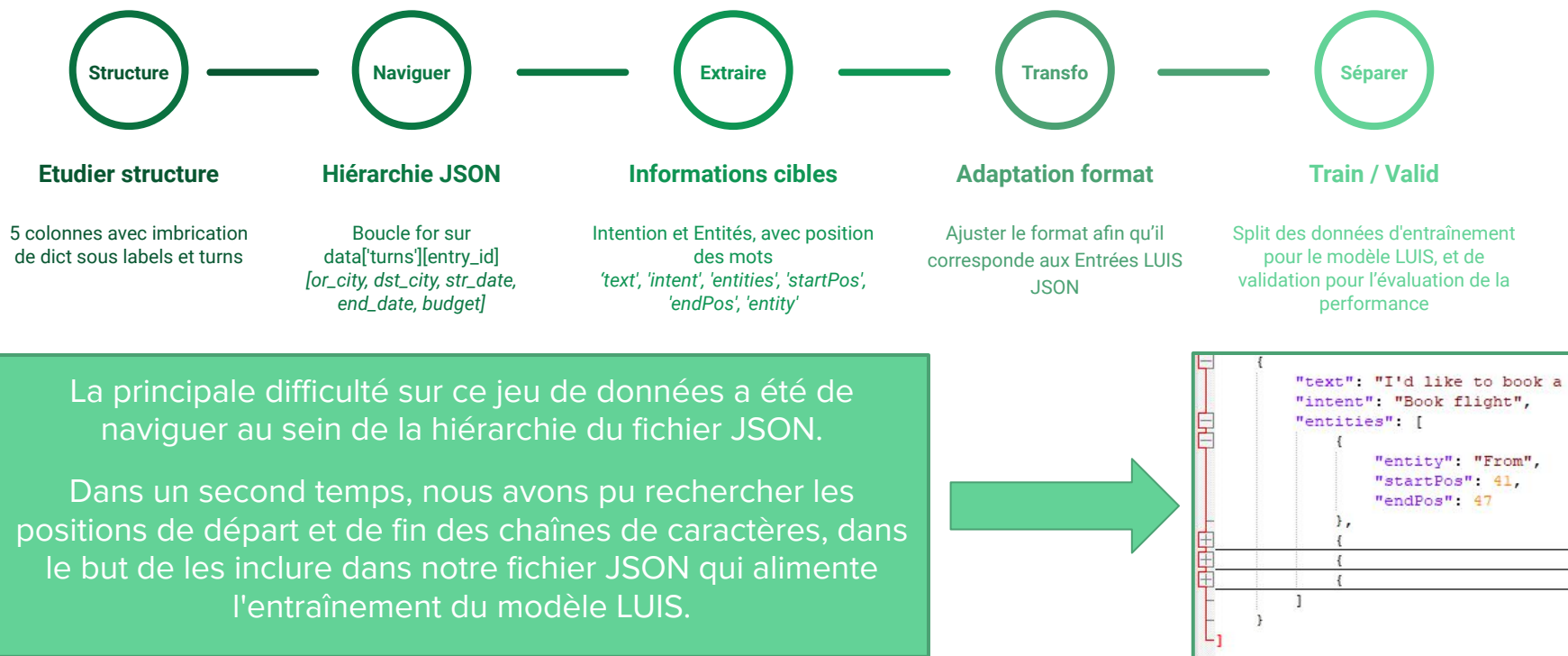


Entités

or_city 811; dst_city 897
str_date 381 ; end_date 215
budget 238
n_adults 282 ; n_children 114

Préparation des données

Processus de préparation des données



LUIS (Language Understanding)

Principes d'utilisation

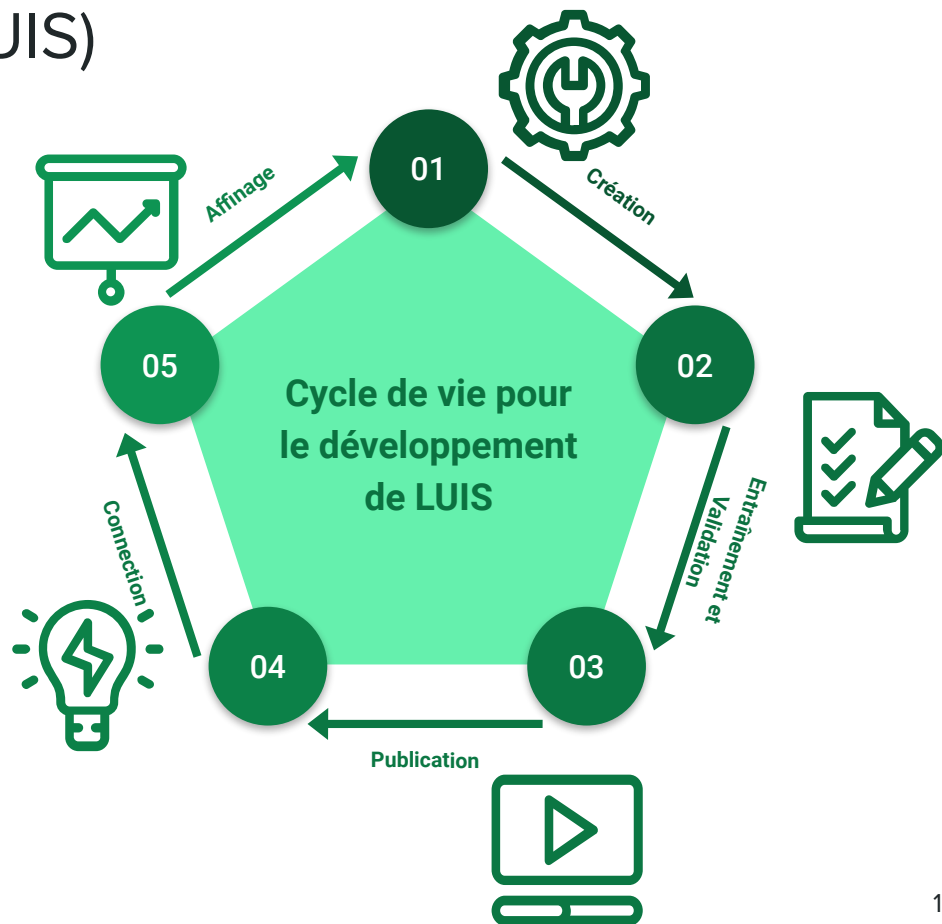
Language Understanding (LUIS)

Description: LUIS est un service d'IA conversationnelle qui applique une intelligence NLP personnalisée afin de prédire le sens général des conversations d'un utilisateur.

Dans le cadre du projet FlyMe, il nous permet d'extraire des informations pertinentes.

Quelques liens:

- [API](#)
- [Se connecter au portail](#)
- [Portail LUIS](#)
- [Bibliothèque de client SDK](#)



Modélisation

Modélisation LUIS

Création

Définir des intentions et des entités.

01

Entraînement et Validation

Ajouter des énoncés d'entraînement pour chaque intention, et tester le modèle avec d'autres énoncés

02

Publication

Déployer l'application pour la prédiction et activer le point de terminaison

03

Ces étapes peuvent être réalisées via le portail LUIS, ou via les bibliothèques de client et API REST (scripts de Modélisation stockés [ici](#))

Le taux d'intention correctement détectée est de = 99.26 %
(voir fin du notebook)

Le taux d'entités détectées alors qu'elles n'auraient pas dû est de = 8.82 %
(voir fin du notebook)

Le taux d'entités non détectées comme espérée = 13.97 %
(voir fin du notebook)

Aperçu de LUIS

Exemples ⓘ

✓ Confirmer toutes les entités  Déplacer vers  Supprimer ...



  Options d'affichage ▾ 

☐ Exemple d'entrée utilisateur Score

Tapez un exemple de ce qu'un utilisateur peut dire et appuyez sur Entrée.

☐ hi , i need to go to mos eisley for a wedding , leaving on saturday , august 13 , 2016 and returning on tuesday , august 16 , 2016 . preferably for \$ 3700 . 1.000

To str_date end_date budget
datetimeV2 datetimeV2

☐ hello , i am planning to book a trip to pittsburgh ✓   : ▾ 0.999


To

☐ hello , i would like to book a 2 - week trip leaving from melbourne on august 27 . i would like to go to mannheim . 1.000

datetimeV2 From str_date To
datetimeV2

☐ i ' m looking for a trip to gotham city leaving from kakariko village on saturday , august 13 , 2016 . 3 adults for no more than \$ 2400 usd . 1.000

To From str_date budget
datetimeV2

☐ hey , i want to go to st . louis on the 17th of august 0.996 

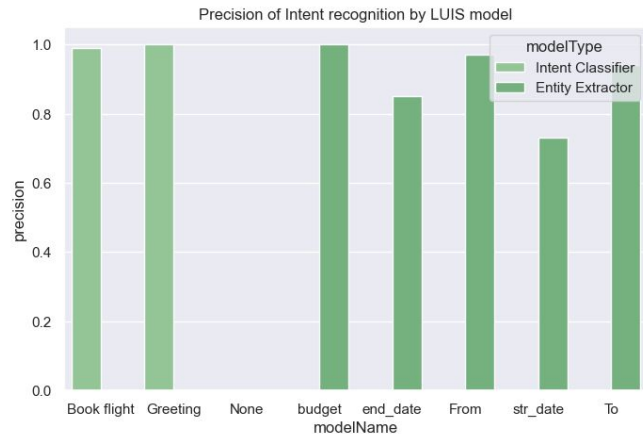
To str_date
datetimeV2

Performance

99.26%

d'intentions correctement détectée

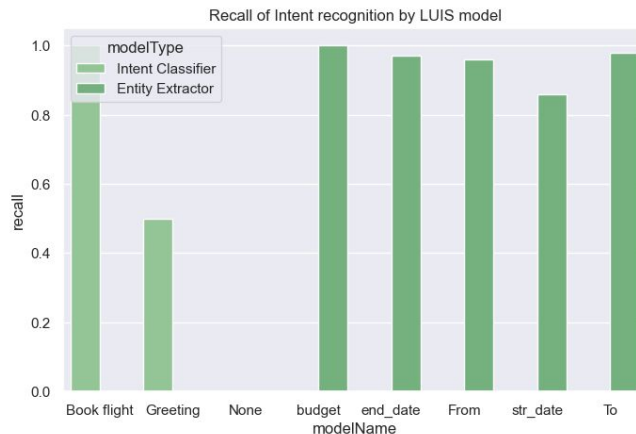
Analyse de la performance



Précision très forte sur les villes de départ et d'arrivée, et plus faible sur les dates

Dans notre cas, nous serons particulièrement attentif à la gestion des faux-positifs, afin d'éviter l'insatisfaction client !
(⇒ Précision)

Performance de rappel réduite sur les intentions "Greeting"



CHATBOT

Spécification fonctionnelle

Eléments de réservation: Intentions et Entités



Texte en anglais, capable de distinguer une intention de réservation et une salutation

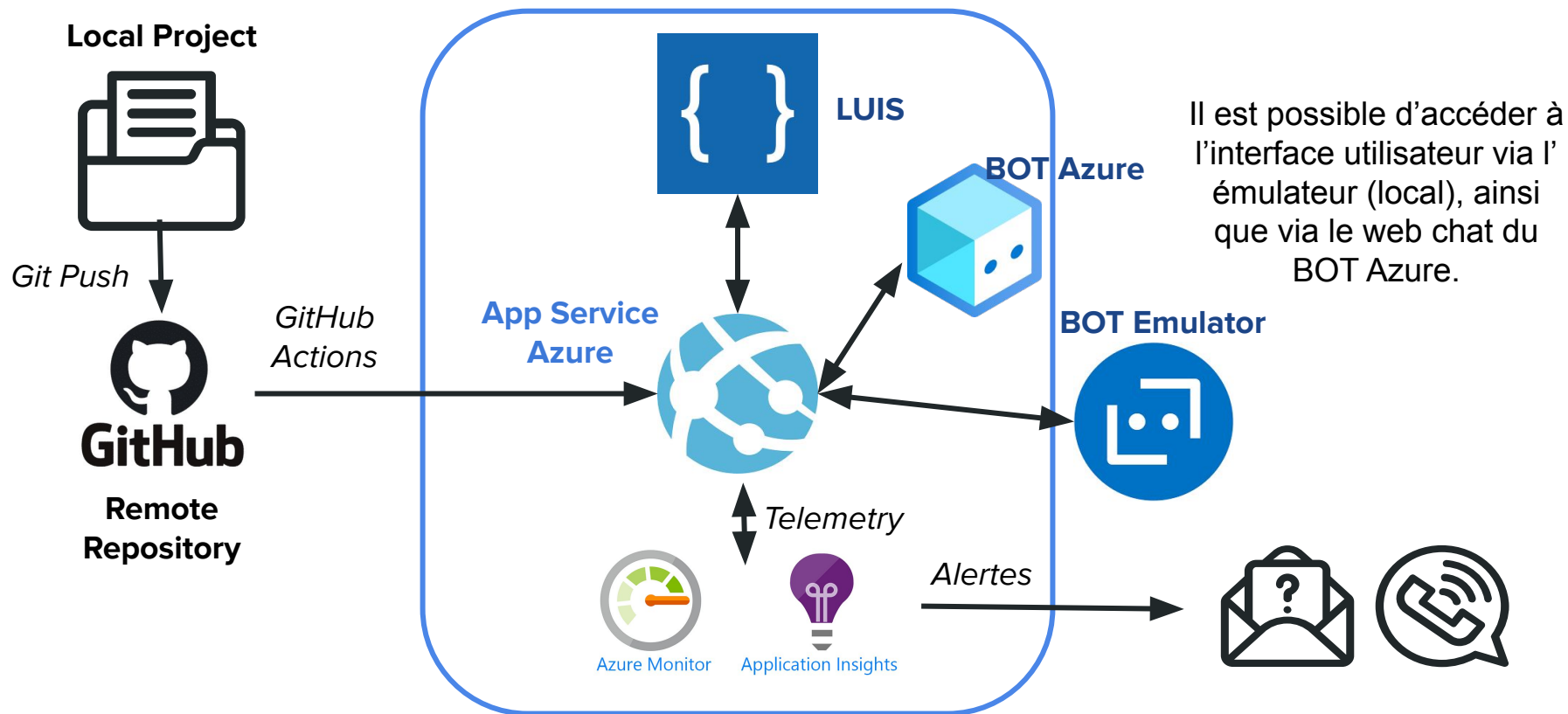
Dialogues depuis diverses interfaces (émulateur & BOT Azure) permettant de capturer les informations nécessaires à la réservation.

Les dates sont capturées en utilisant la fonction timex, en les retournant au format approprié.

L'ensemble des informations sont récapitulées avant validation finale de l'utilisateur.

Spécification technique

Architecture



Tests unitaires

Tests unitaires

01

Test Intention Greeting

02

Test Intention Book
flight

03

Test Intention Score
Greeting

04

Test Intention Score
Book flight



```
Test with pytest
1  ▶ Run pytest unit_test
10 ===== test session starts =====
11 platform linux -- Python 3.9.15, pytest-6.2.3, py-1.11.0, pluggy
12 rootdir: /home/runner/work/IA_Project10_Openclassrooms_chatbot/1
13 collected 9 items
14
15 unit_test/unit_test.py .....
16
17 ===== 9 passed in 4.42s =====
```

05

Test Entité From

06

Test Entité To

07

Test Entité Score From

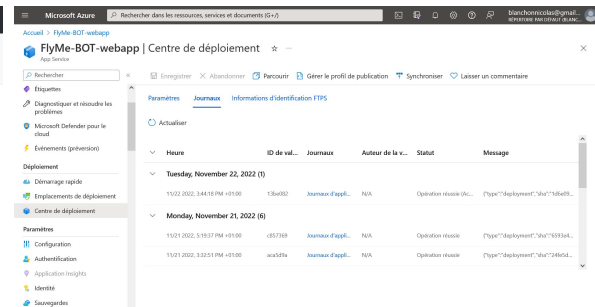
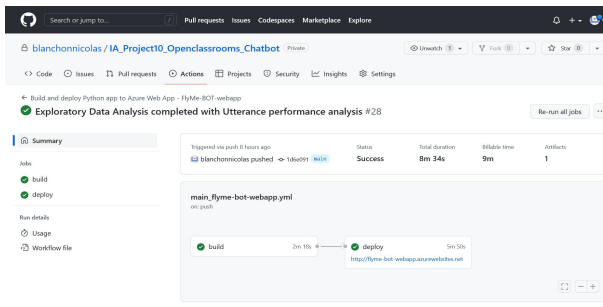
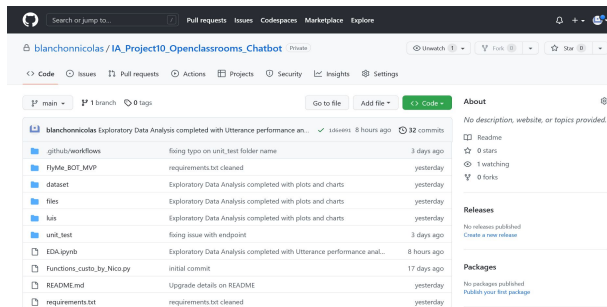
08

Test Entité Score To

8 tests unitaires sont exécutés automatiquement lors du déploiement, via les actions GitHub.

Intégration continu

Stockage GitHub et GitHub Actions



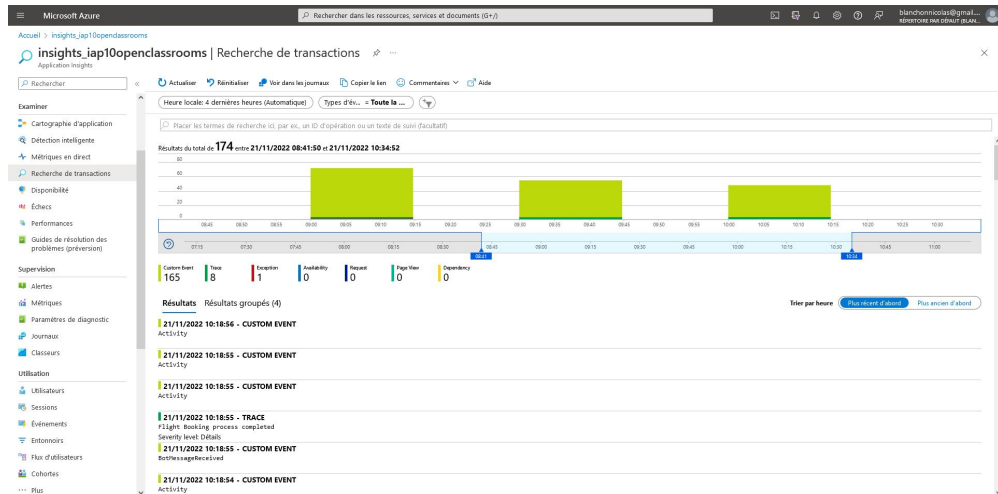
Le code est stocké sur Github, et mis à jour régulièrement via les services Git Push

GitHub Actions déclenche les activités de Build et Déploiement, en se connectant à Azure via le profil de publication

Le code est reçu sur les ressources précédemment créées sur le cloud Azure, et démarre via la commande de démarrage

Monitoring

Surveillance du fonctionnement



La méthode track_trace configurée au travers des fonctions Telemetry du BOT SDK, permettent de suivre le bon fonctionnement du BOT

Des alertes peuvent être créées afin d'informer les responsables opérationnels d'éventuels problèmes de fonctionnement

Microsoft Azure

Fired:Sev2 Azure Monitor Alert User validation rejected on insights_iap10openclassrooms (microsoft.insights/components) at 11/23/2022 9:56:51 AM

[View the alert in Azure Monitor >](#)

Summary

Alert name	User validation rejected
Severity	Sev2
Monitor condition	Fired
Affected resource	insights_iap10openclassrooms
Resource type	microsoft.insights/components
Resource group	iap10openclassrooms
Subscription	Azure subscription 1
Description	User has not rejected the proposed flight booking options, more than 3 times over the last 30 minutes
Monitoring service	Log Alerts V2

DEMO

Démo

https://flyme-bot-webapp.azurewebsites....

SERVICES

Welcome to FlyMe Bot

FlyMe BOT offers new user experience, to help you book your next trip

App Service BOT URL

App Service Messaging API

Openclassrooms Project 10 summary

Welcome to FlyMe Chatbot, How can I help you to book your next trip ?

I want to go in Fortaleza today, from Paris and come back on December 12th 2022 , for a budget of 50 \$

Just now

Please confirm, I have you traveling to: Fortaleza taking off from: Paris leaving on: 2022-11-21 coming back on: 2022-12-12 for a total trip duration of: 21 days within a budget of : \$50

Yes

Just now

I have you booked a flight from Paris to Fortaleza leaving on 2022-11-21, and coming back on 2022-12-12 for a very cheap price of \$ 50

What else can I do for you?

Type your message

JSON

```
root:
  type: "conversationUpdate"
  membersAdded:
  membersRemoved:
  channelId: "emulator"
  conversation:
    id: "f148bef0-699a-11ed-8b63-11ea457a1853"
    localTimestamp: "2022-11-21T13:49:31+01:00"
  recipient:
    timestamp: "2022-11-21T12:49:31.231Z"
  from:
    locale: "en-US"
    serviceUrl: "https://2d72-149-34-245-10.ngrok.io"
```

```
[13:49:31] Connecting to bot on https://flyme-bot-webapp.azurewebsites.
[13:49:31] Emulator listening on http://[::]:56268
[13:49:31] ngrok listening on https://2d72-149-34-245-10.ngrok.io
[13:49:31] ngrok traffic inspector: http://127.0.0.1:4040
[13:49:31] Will bypass ngrok for local addresses
[13:49:31] -> conversationUpdate
[13:50:28] -> message Hello
[13:51:05] <- message application/vnd.microsoft.card.adaptive
[13:51:06] <- message Welcome to FlyMe Chatbot, How can I help you to b
[13:51:06] POST 200 directline/conversations/<conversationId>/activiti
[13:51:06] <- trace Bot State
[13:51:06] POST 200 directline/conversations/<conversationId>/activiti
[13:55:42] -> message I want to go in Fortaleza today, from Paris and c
[13:55:53] <- trace Luis Trace
[13:55:53] <- message Please confirm, I have you traveling to: Fortalei
[13:55:53] <- trace Bot State
[13:55:53] POST 200 directline/conversations/<conversationId>/activiti
[13:56:02] -> message Yes
[13:56:03] <- message I have you booked a flight from Paris to Fortalei
[13:56:03] <- message What else can I do for you?
[13:56:03] <- trace Bot State
[13:56:03] POST 200 directline/conversations/<conversationId>/activiti
```

Conclusions



Modèle de chatbot évolutif pour la réservation de voyage en utilisant les services Microsoft Azure



Intégration continue via les services Git et GitHub Actions



Evaluation de la performance du modèle en production, et suivi en temps réel de l'utilisation via Application Insights et les alertes customisées



Mise à jour du modèle avec la prise en charge d'autres entités et intentions. Recherche d'amélioration pour le traitement des dates et budget.