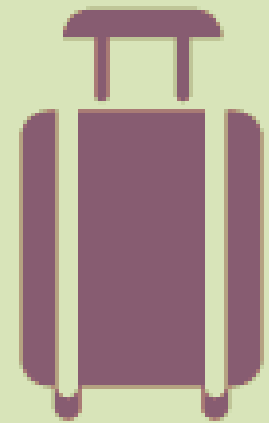


/01



FLY
ME

**Développez un
chatbot pour
réserver des
vacances**

Sommaire

Introduction (Context, Objective)
Les données
Entraînement et tests d'un modèle
d'analyse sémantique: LUIS
Creation, Deployment et Tests du
chatbot
Application Insight
Architecture Actuelle et Cible
Presentation d'application (demo)
Conclusions





Introduction

Context

- FlyMe: entreprise qui propose des voyage au professioneles et particuliers
- Creation d'une chatbot pour les colaborateurs de l'entreprise FlyMe

Objective

- Construire une MVP pour aider les employés de FlyMe à réserver des voyages pour leurs vacances. Pour ça on va utilisé des données disponible en ligne.
- Creation des tests unitaires pour testé le chat.
- Identifié dans la demande d'une utilisateur:
 - Ville de départ
 - Ville de destination
 - Date aller souhaitée du vol
 - Date retour souhaitée du vol
 - Budget maximum pour le prix total des billets
- Suivre et d'analyser l'activité du chatbot en production et lever une alerte lorsque le chatbot donnera plusieurs mauvaises réponses à des utilisateurs



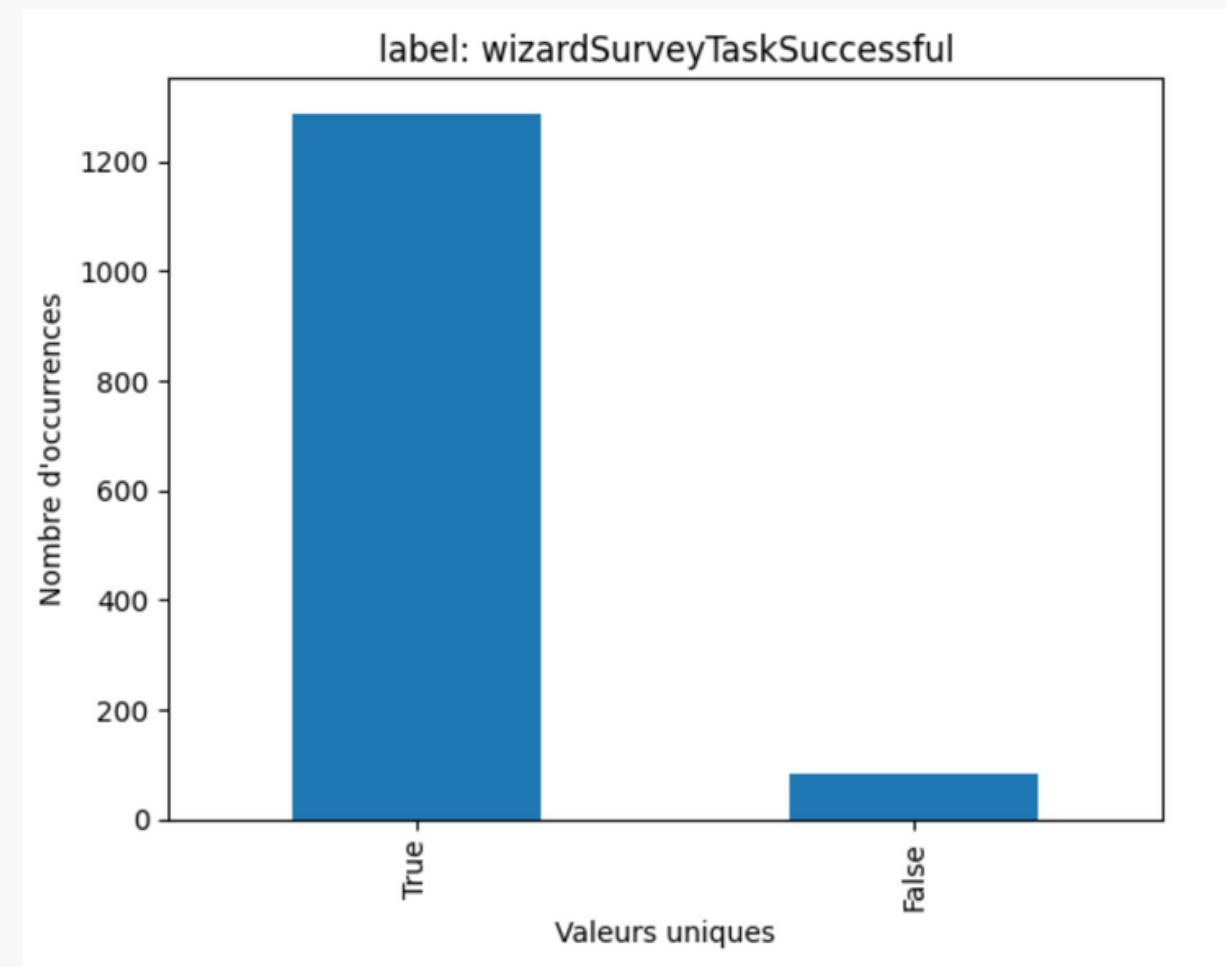
Les données

Frames Dataset

<https://www.microsoft.com/en-us/research/project/frames-dataset/download/>



- Contient un historique d'échange entre un utilisateur et le Bot composés de plusieurs tours.
- Les données sont dans une format json.
- Le dialogue est entre un robot et un utilisateur essayant de réserver un vol.



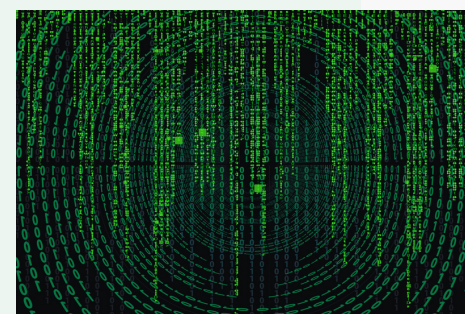
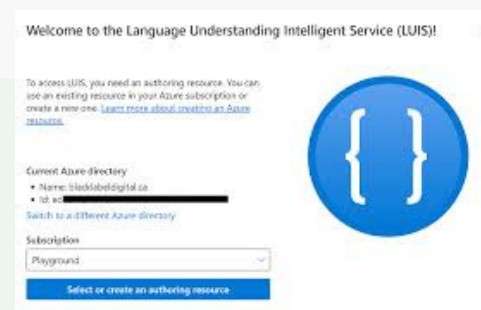
- L'ensemble de données est composé de **1369 dialogues annotés**. 82 communication avec le chatbot n'a pas aider l'utilisateur.

/05

MODÈLE D'ANALYSE SÉMANTIQUE: LUIS

Première étape du projet

- En utilisant le service cognitif Azure LUIS (Language Understanding) - entraîner et deployer une modèle d'analyse sémantique



Étape 1

Creation de ressource LUIS.

Étape 2

Ajouter les intents, entities, prebuilt entities et rendre les données compatible avec LUIS

Étape 3

Ajouter les exemples au modèle.

```
{'text': 'leave whenever. i should be back for September 11',
'intent_name': 'BookFlight',
'entity_labels': [{'entity_name': 'end_date',
'start_char_index': 37,
'end_char_index': 49}]}
```

Étape 4

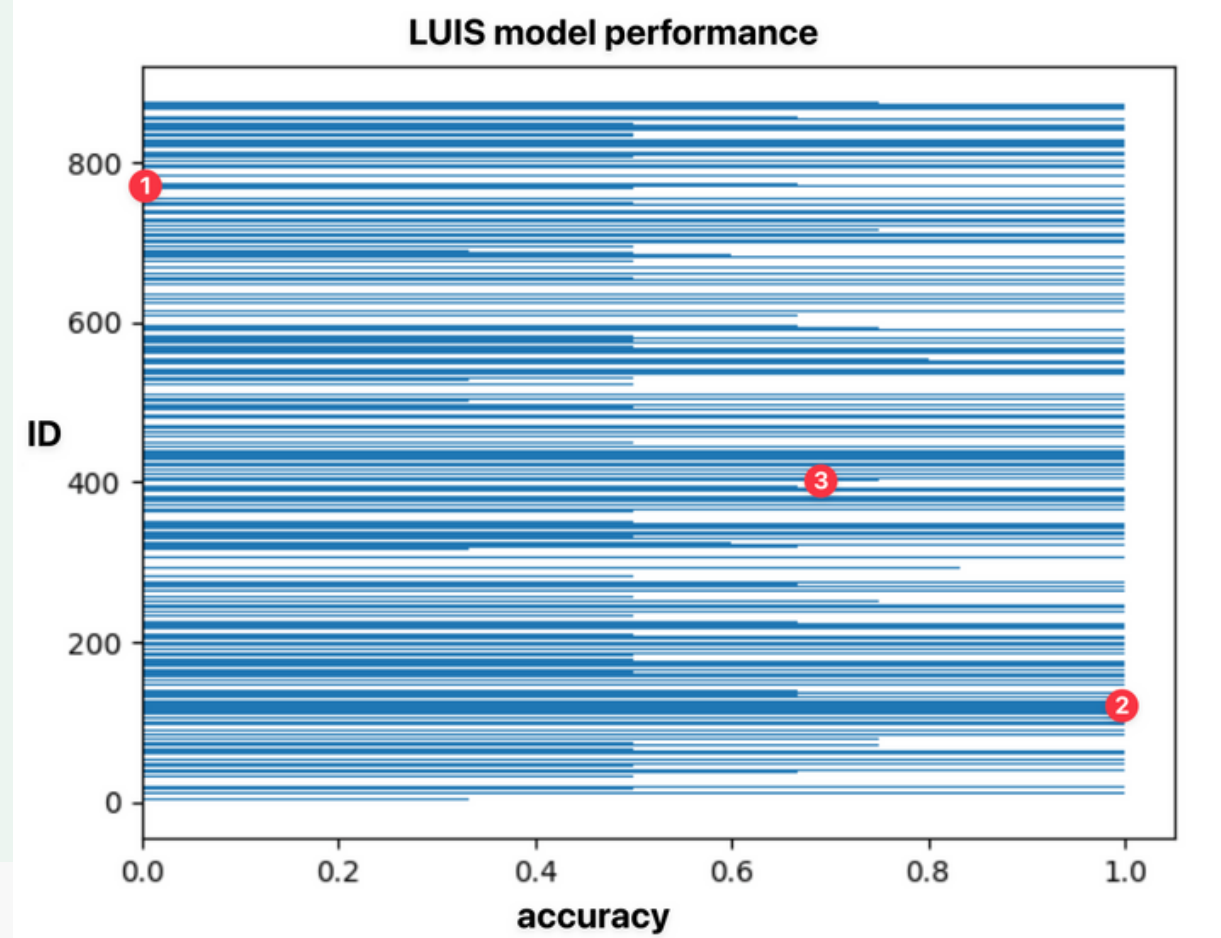
Entraîner et déployer (publier) le modèle

Étape 5

Tester.
 query = 'I would like to book a flight from Lyon to Paris tomorrow at 11pm with a budget of 100 euros'

```
{'or_city': 'Lyon',
'dst_city': 'Paris',
'geographyV2': {'value': 'Paris', 'type': 'city'},
'datetimeV2': {'type': 'datetime',
'values': [{'timex': '2023-02-23T23',
'resolution': [{'value': '2023-02-23 23:00:00'}]}]},
'number': 100,
'budget': '100 euros'}
```

59,71 %



A partir des données de test, on obtient une précision de 59,71% entre le modèle prévu et la libéralisation effective des données originales.

Chaque énoncé a fait l'objet d'une évaluation.

- Un certain nombre d'énoncés ont été évalués comme étant **entièrement exacts**.
- Bon nombre d'entre eux **ne prévoient rien**
- D'autres ont un niveau de **précision moyen**.

On peut examiner ces trois types d'énoncés en les observant.

```
test_data[0]
```

```
{'text': 'how about if I leave from Beijing?',  
'intent_name': 'BookFlight',  
'entity_labels': [{'entity_name': 'or_city',  
  'start_char_index': 26,  
  'end_char_index': 33}]}
```

```
evaluator.y_pred[0]
```

```
{'geographyV2': {'value': 'Beijing', 'type': 'city'}, 'or_city': 'Beijing?'}
```

```
evaluator.y_true[0]
```

```
{'or_city': 'Beijing'}
```

Tester LUIS

/07

CHATBOT

Seconde étape du projet

- On doit construire une chatbot en utilisant Microsoft Bot Framework SDK (python)
- Et le déployé sur Azure Web App



Étape 1

Clonner et utilisé <https://github.com/microsoft/BotBuilder-Samples/tree/main/samples/python/21.corebot-app-insights>



Étape 2

Connecter le Bot au service LUIS pour semantiser les messages d'utilisateur et recuperer les 5 unité demandé.



Étape 3

Implementer les dialogues: demande d'information, annulation, confirmation



Étape 4

Implementer les tests unitaire: luis et dialogues.



Étape 5

Déployé sur Azure Web App.



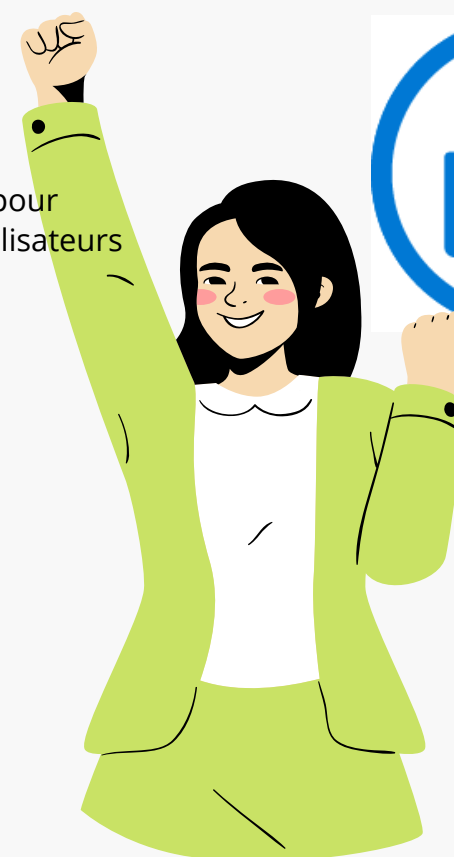
Étape 6

Utiliser Azure App Insights pour l'analyse de l'activité des utilisateurs



Étape 7

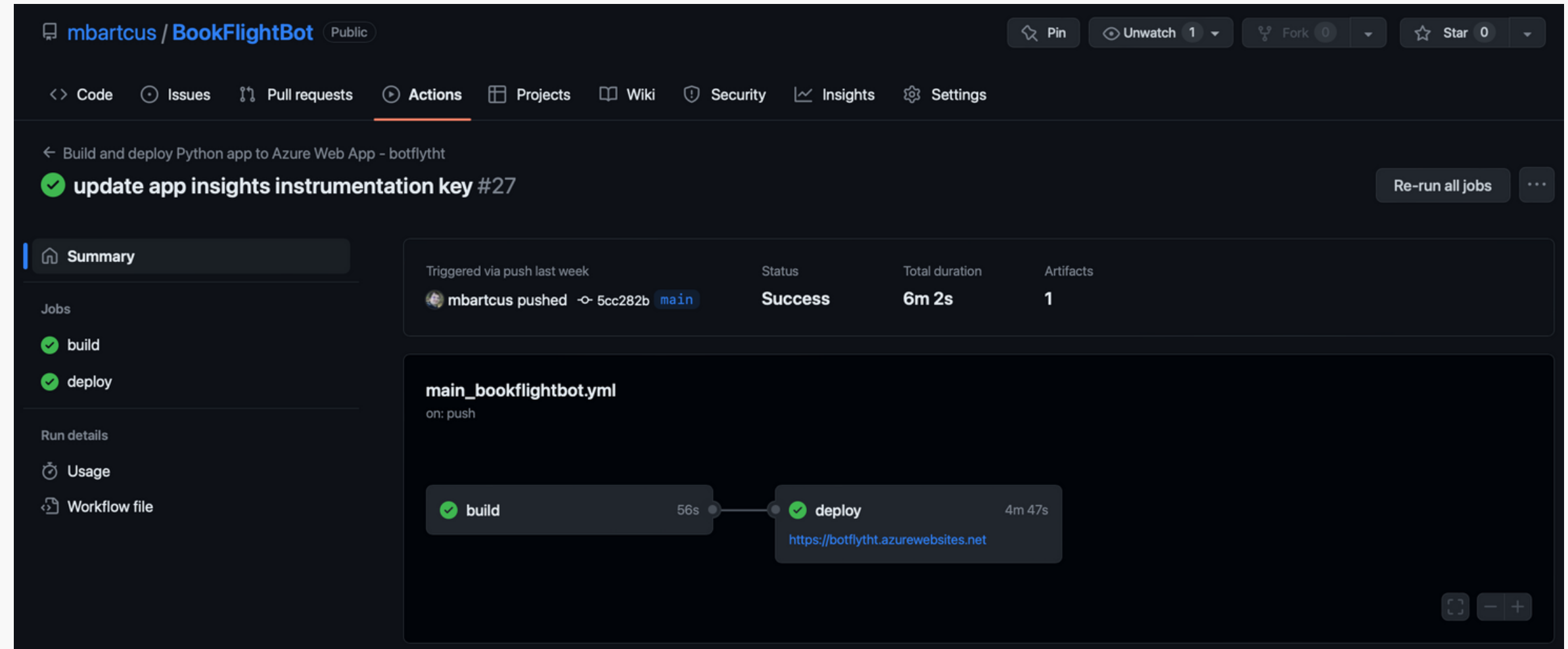
Utiliser Bot Emulator en local pour tester la communication avec le chatbot



/08

GitHub Actions

Service d'automatisation de workflows qui permet aux développeurs de créer, tester et déployer du code de manière continue directement depuis leur dépôt Github.

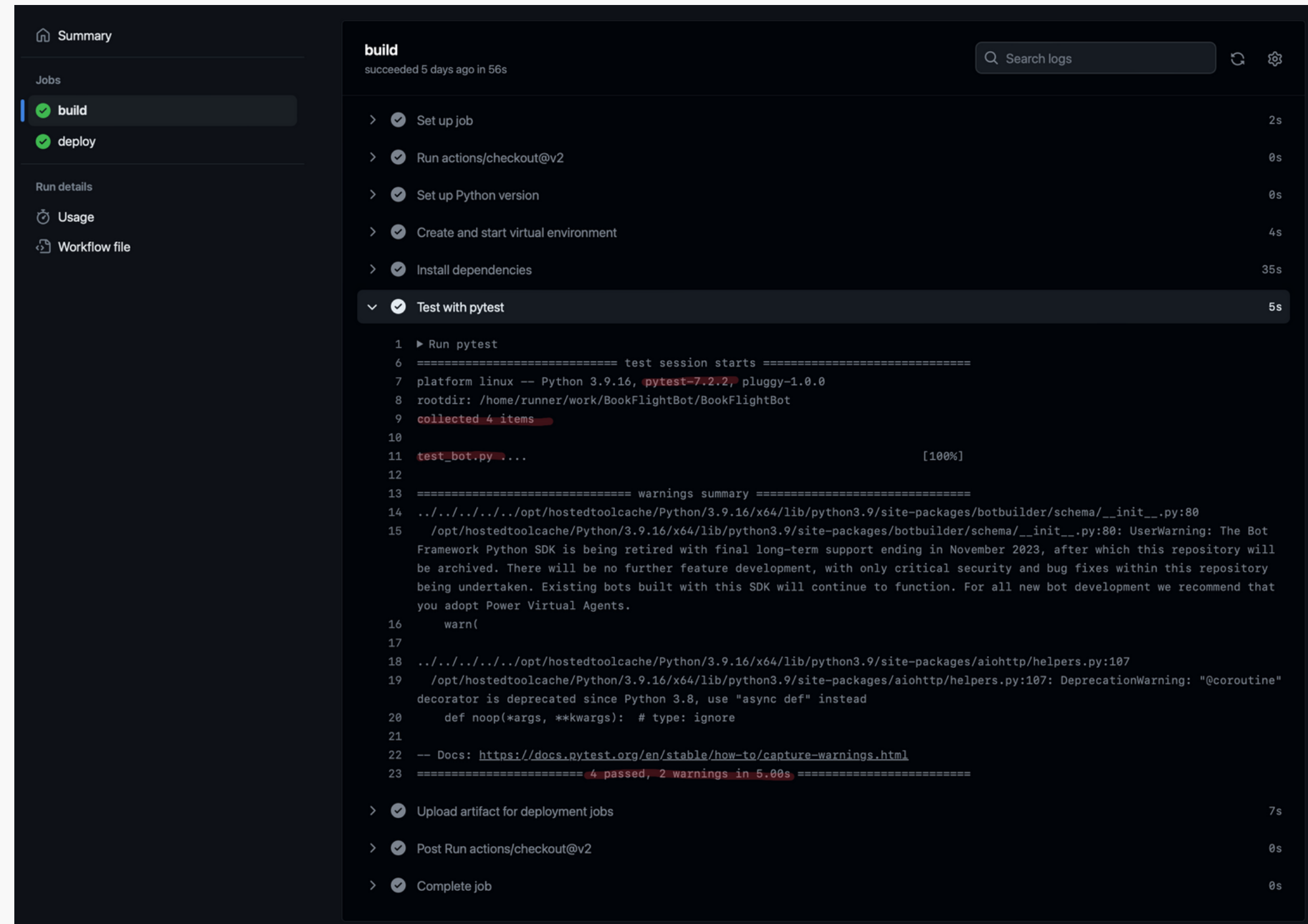


/09

Tests Unitaire

Les tests unitaire sont réalisé pour chaque push sur github et à chaque build.

- Tester LUIS
- Tester une réservation étape par étape
- Tester une annulation de réservation
- Tester une réservation en fournissant toutes les informations en une seule fois



The screenshot displays a GitHub Actions workflow run for a 'build' job. The job is marked as 'succeeded 5 days ago in 56s'. The workflow consists of several steps: 'Set up job' (2s), 'Run actions/checkout@v2' (0s), 'Set up Python version' (0s), 'Create and start virtual environment' (4s), 'Install dependencies' (35s), 'Test with pytest' (5s), 'Upload artifact for deployment jobs' (7s), 'Post Run actions/checkout@v2' (0s), and 'Complete job' (0s). The 'Test with pytest' step is expanded, showing the following output:

```
1 ▶ Run pytest
6 ===== test session starts =====
7 platform linux -- Python 3.9.16, pytest-7.2.2, pluggy-1.0.0
8 rootdir: /home/runner/work/BookFlightBot/BookFlightBot
9 collected 4 items
10
11 test_bot.py ... [100%]
12
13 ===== warnings summary =====
14 ../../../../opt/hostedtoolcache/Python/3.9.16/x64/lib/python3.9/site-packages/botbuilder/schema/__init__.py:80
15 /opt/hostedtoolcache/Python/3.9.16/x64/lib/python3.9/site-packages/botbuilder/schema/__init__.py:80: UserWarning: The Bot
Framework Python SDK is being retired with final long-term support ending in November 2023, after which this repository will
be archived. There will be no further feature development, with only critical security and bug fixes within this repository
being undertaken. Existing bots built with this SDK will continue to function. For all new bot development we recommend that
you adopt Power Virtual Agents.
16     warn(
17
18 ../../../../opt/hostedtoolcache/Python/3.9.16/x64/lib/python3.9/site-packages/aiohttp/helpers.py:107
19 /opt/hostedtoolcache/Python/3.9.16/x64/lib/python3.9/site-packages/aiohttp/helpers.py:107: DeprecationWarning: "@coroutine"
decorator is deprecated since Python 3.8, use "async def" instead
20     def noop(*args, **kwargs): # type: ignore
21
22 -- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
23 ===== 4 passed, 2 warnings in 5.00s =====
```

/10

The screenshot displays a CI/CD pipeline summary for a 'deploy' job. The job is marked as 'succeeded 5 days ago in 4m 47s'. The pipeline consists of three main steps: 'Set up job' (4s), 'Download artifact from build job' (7s), and 'Deploy to Azure Web App' (4m 34s). The 'Deploy to Azure Web App' step is expanded, showing a list of log entries: 1. Run azure/webapps-deploy@v2; 7. Package deployment using ZIP Deploy initiated; 8. Deploy logs can be viewed at <https://botflyttht.scm.azurewebsites.net/api/deployments/57782613-8137-45d2-8ca5-906993b7c48b/log>; 9. Successfully deployed web package to App Service; 10. App Service Application URL: <https://botflyttht.azurewebsites.net>. The final step is 'Complete job' (0s), which includes: 1. Evaluate and set environment url; 2. Evaluated environment url: <https://botflyttht.azurewebsites.net>; 3. Cleaning up orphan processes. The interface includes a search bar for logs and a sidebar with navigation options like Summary, Jobs, Run details, Usage, and Workflow file.

Déploiement

Le déploiement est réalisé automatiquement pour chaque push sur github.

/11

App Insights

- Pour détecter les mauvaises interactions (expérience) entre le chat bot et l'utilisateur.
- On traque deux infos qui remontent au App Insights.
 - INFO (lorsque l'utilisateur est satisfait de la réservation proposée par le chatbot.)
 - ERROR (quand l'utilisateur est insatisfait)

Si l'utilisateur ne valide pas 3 fois la réservation sur une période de 5 minutes, une alerte est déclenchée.

The screenshot displays the Microsoft Azure portal interface for App Insights Alerts. The main view shows a table of alerts for the resource 'botfly'. The table has columns for Name, Severity, Affected resource, Alert condition, User response, and Fire time. Three alerts are listed, all with the name 'not_satisfied' and severity 'Error'. The 'Create an alert rule' dialog box is open, showing configuration options for the alert rule.

Search query *

```
traces  
| where message contains "not satisfied"  
| project message  
| summarize count() by message
```

Alerts Summary:

Total alerts	Critical	Error	Warning	Informational	Verbose
3	0	3	0	0	0

Alerts Table:

Name	Severity	Affected resource	Alert condition	User response	Fire time
not_satisfied	1 - Error	botfly	Fired	New	4/6/2023, 10:36 AM
not_satisfied	1 - Error	botfly	Fired	New	4/6/2023, 10:26 AM
not_satisfied	1 - Error	botfly	Fired	New	3/30/2023, 1:51 PM

Create an alert rule *

Measure: count_

Aggregation type: Total

Aggregation granularity: 5 minutes

Split by dimensions

Use dimensions to monitor specific time series and provide context to the fired alert. Dimensions can be either number or string columns. If you select more than one dimension value, each time series that results from the combination will trigger its own alert and will be charged separately.

Dimension name	Operator	Dimension values	Include all future values
Select dimension	=	0 selected	<input type="checkbox"/>

Alert logic

Operator: Greater than or equal to

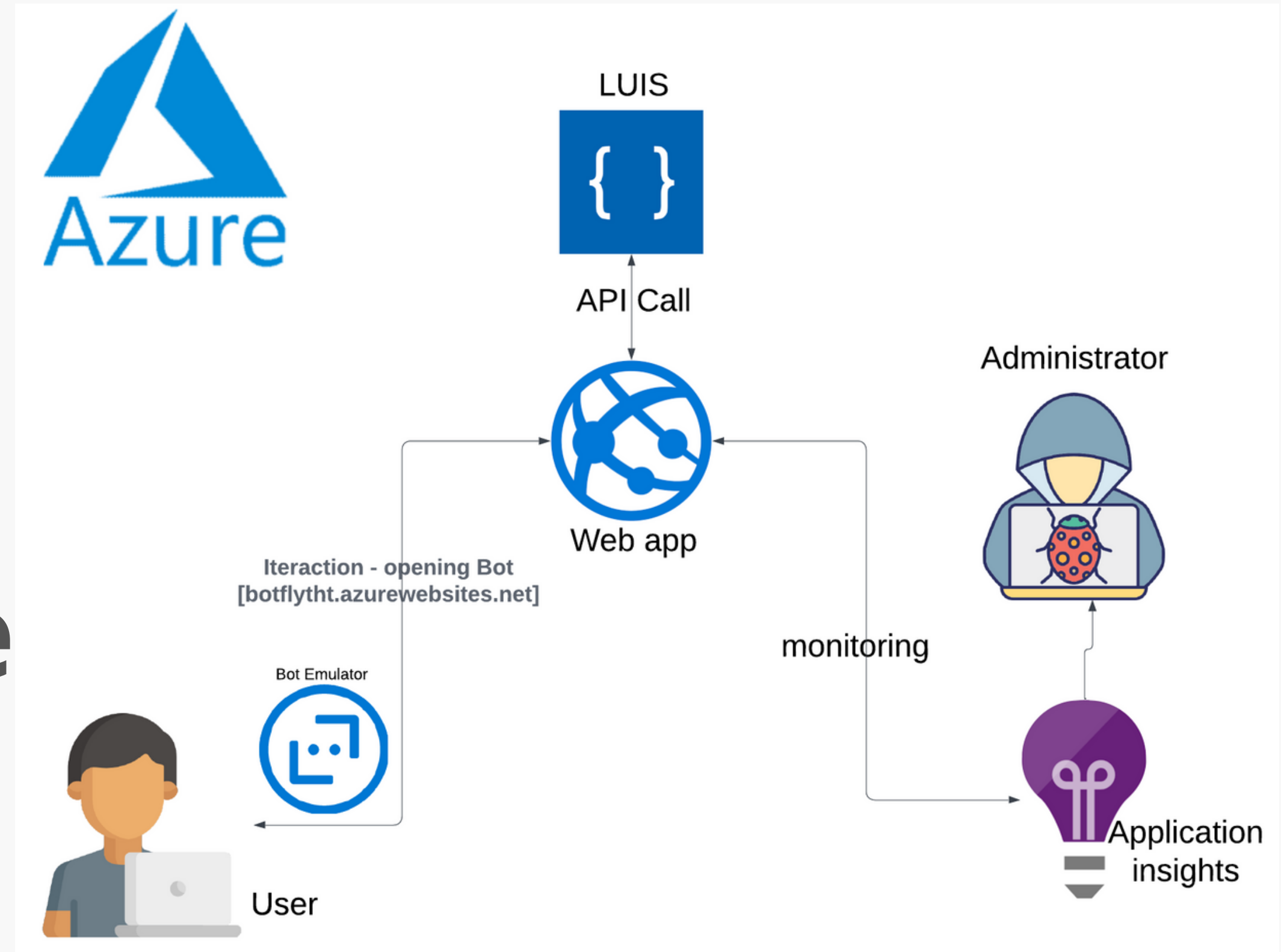
Threshold value: 3

Frequency of evaluation: 5 minutes

Estimated monthly cost \$1.50 (USD)

/12

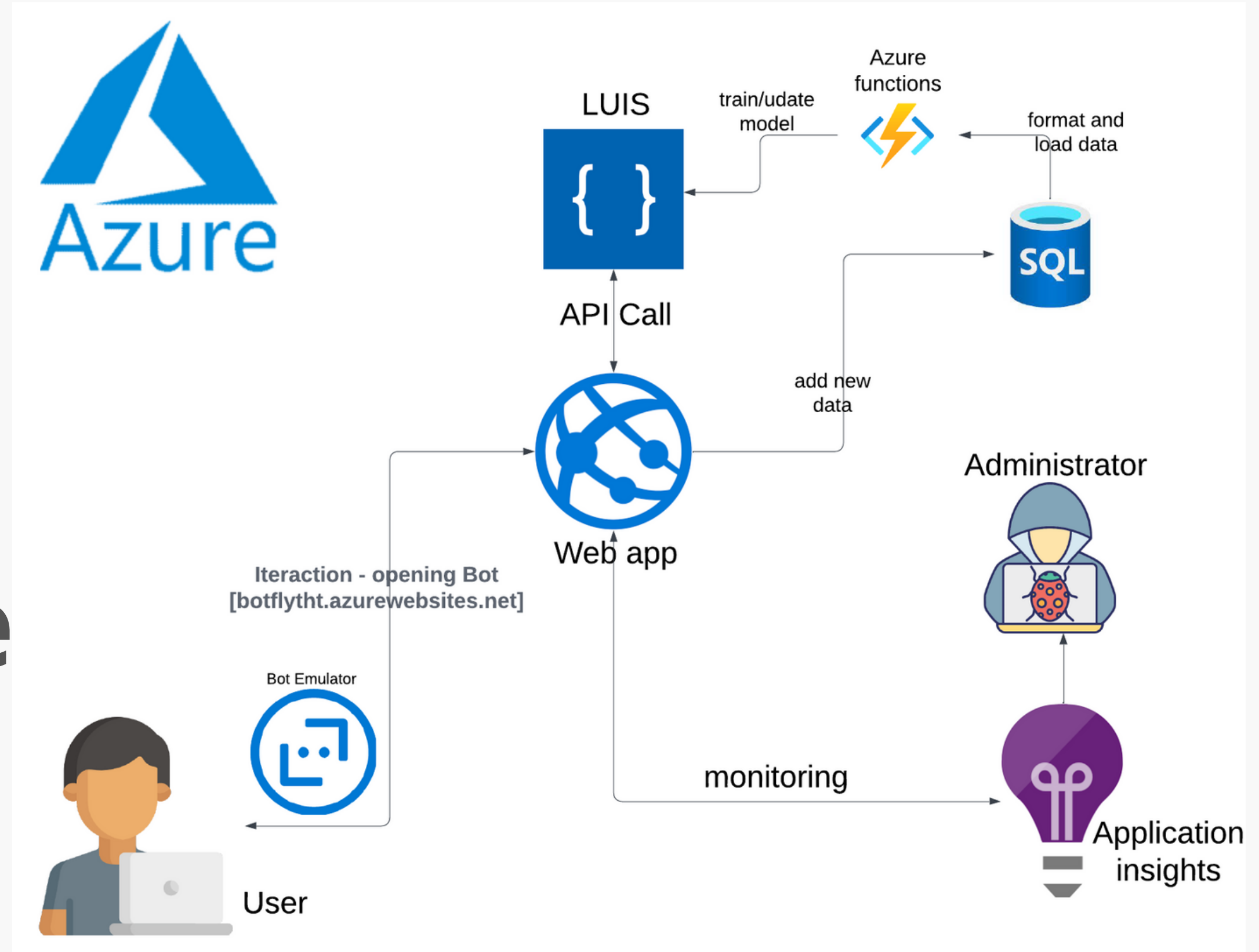
Architecture Actuelle



Cette architecture est destinée au MVP et n'est pas définitive. Une prochaine version pourra enrichir notre modèle en fonction des nouvelles données.

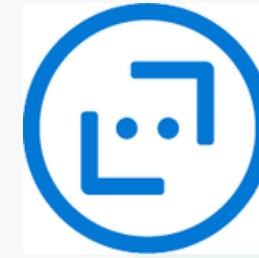
/13

Architecture Cible



Une architecture d'une prochaine version

- Récupération des échanges entre les employés de l'entreprise et le Bot afin d'enrichir les données.
- Sur une fréquence d'un mois apprenait un nouveau modèle sur les nouvelles données recueillies et si ses résultats sont mieux, déployés un nouveau modèle.



On parle avec le Bot

On doit installer ngrok pour tester en local le chatbot déployé

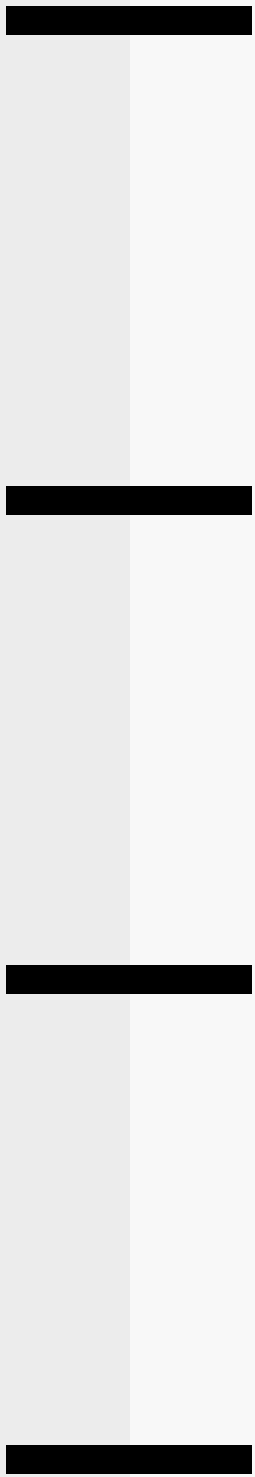
```
ngrok -version  
ngrok version 2.3.41
```

Service

ngrok is network tunneling software. The Bot Framework Emulator works with ngrok to communicate with bots hosted remotely. Read the [wiki page](#) to learn more about using ngrok and how to download it.

Path to ngrok

The screenshot shows the Bot Framework Emulator interface. On the left, there's a 'BOT EXPLORER' sidebar with 'ENDPOINT' and 'SERVICES' sections. The main area displays a chat window with a 'Welcome' message, a 'Live Chat' button, and a 'Restart Conversation' option. The chat history shows a user saying 'hy', the bot replying 'What can I help you with today?', the user asking to book a trip to Chisinau, and the bot providing details. A 'New bot configuration' dialog is open, showing fields for Bot name (bottes), Endpoint URL (https://botflyth.azurewebsites.net/api/messages), Microsoft App ID, and Microsoft App password. The dialog also has checkboxes for 'Azure for US Government' and 'Encrypt keys stored in your bot configuration'. At the bottom right, there's a log panel showing the JSON responses from the bot.



Ce chatbot est un MVP. Des versions ultérieures doivent être développées.

Il faudra mettre en place l'intégration continue comme dans l'architecture cible.

De nouvelles entités peuvent être ajoutées (l'aéroport, le nombre de passagers et leurs informations, les services supplémentaires, etc.)

Ajouté d'autres surveillances comme par exemple en observant le temps de réponse de l'utilisateur.

Conclusions



Merci !

