Projet 7 - Créez GrandPy Bot, le papy-robot

Parcours OpenClassrooms - Développeur d'application Python

Etudiant : Guillaume OJARDIAS Mentor : Erwan KERIBIN
Mentor évaluateur : Aurélien MASSE

I. Contexte

Le but du projet est de créer un chatbot permettant à l'utilisateur de trouver l'adresse d'un lieu ainsi que d'avoir plus d'informations au sujet de ce lieu grâce à extrait d'article Wikipédia.

Le code source du projet est disponible à l'adresse suivante : => https://github.com/GuillaumeOj/YodaPy

L'application est déployée sur la plateforme Heroku : => https://yoda-py.herokuapp.com/

Pour finir, les users stories utilisées pour l'élaboration de l'application sont disponibles ici : => https://trello.com/b/P17ksldE/yodapy

Cette application web, nous permet de mettre en œuvre un certain nombre de connaissances :

- l'utilisation d'API,
- l'utilisation d'un framework (FLASK),
- l'utilisation du JavaScript (y compris AJAX),
- le déploiement d'une application en ligne,
- la réalisation d'une interface responsive (HTML5 + CSS3),
- la mise en place de tests (module Pytest) pour un projet en mode TDD,
- et bien entendu utilisation du langage Python.

II. Démarche

L'application suit le fonctionnement suivant.

1. L'utilisateur écrit un message, puis l'envoi (touche `<Enter>` ou bouton `Envoyer`)
2. Un script JavaScript récupère le contenu du message, l'affiche dans un fil de discussion et l'envoi (en AJAX) à une page `/process` de l'application.
3. Sur la page `/process`, l'entrée utilisateur est envoyée à un `parser` qui va "normaliser" le message de la manière suivante :
   a. texte en minuscule, suppression des caractères accentués,
   b. séparation du texte en liste de phrases,
   c. cherche la phrase contenant le lieu cherché par l'utilisateur,
   d. suppression des mots "inutiles" (stopwords).
4. Si le parser retourne une valeur (mots clefs), ceux-ci sont envoyés à un module permettant la recherche de coordonnées géographique d'un lieu. Ici, le module fait appel à l'API fournie par MapBox.
   a. envoi d'une requête à l'API précisant les mots clefs de la recherche et les paramètres (langue du résultat, token utilisateur, paramètres supplémentaires),
   b. le module fait un choix (parmi la liste retournée par l'API) en se basant sur l'indice `relevance` fourni dans le résultat.
5. Si le module de recherche géographique renvoi un résultat, les coordonnées géographiques sont envoyées au module permettant la recherche d'article Wikipédia.
   a. envoi d'une requête à l'API précisant les coordonnées à utiliser pour la recherche ainsi que les paramètres précisant les informations souhaitées (extrait d'article, titre de l'article),
   b. le module choisie l'article avec l'index le plus petit,
   c. une URL vers l'article est reconstituée (non fournie dans le résultat)
6. Pour les étapes 4 et 5, on fait appel au module `Bot` permettant de retourner des phrases selon le contexte (adresse trouvée, article trouvé, demande non comprise, erreur grave, etc.)

7. Toute ces informations sont retournées à la page principale sous forme de JSON.
8. Le script JavaScript reçoit le résultat de la requête AJAX et affiche dans le fil des messages sous forme de différents posts les informations retournées.

III. Bilan

1. Les tests

Ce projet nous permet de mettre en œuvre nos premiers tests unitaires sur un projet. Pour cela, j'ai utilisé le package Pytest. Ce que j'ai appris, c'est que dans son principe de base, un test est très facile à mettre en place et à construire.

Cependant, quand il entre dans une logique de méthodologie de travail, notamment d'un projet en mode Test Driven Development (TDD), les choses se compliquent. En effet, il ne s'agit pas seulement d'écrire un test, mais de le penser et le concevoir de façon à ce qu'il soit utile et sûr.

Finalement, j'ai compris la valeur ajouté d'un développement en TDD, mais je pense qu'il va falloir encore beaucoup de pratique avant de réussir à écrire des tests pertinents qui ne se cassent pas à chaque "amélioration" du code.

2. Le JavaScript

De ce que je connaissais du langage (c'est à dire pas grand chose), le JavaScript était un langage uniquement là pour apporter des effets dynamiques à la page web. C'était sans compter sur les évolutions importantes au cours de ces dernières années que je n'avais pas du tout suivie.

Finalement aujourd'hui j'ai, grâce à ce projet, une vision plus large des possibilités offertes par le langage. Cependant, je me rends compte aussi que son apprentissage n'est pas aussi "simple" que celui du langage Python. En effet étant donné l'évolution rapide qu'il a subit ces dernières années, trouver des ressources à jour n'est pas toujours aisée.

J'ai découvert aussi à quel point ce langage peut avoir une forte dépendance du client de l'utilisateur de l'application. Il est donc toujours important de vérifier que telle méthode ou telle fonction est fonctionnelle sur l'ensemble (ou la majorité) des navigateurs du marché.

La possibilité était donnée d'utiliser JQuery sur ce projet (un framework), j'ai délibérément choisi de ne pas en faire l'usage pour me confronter au langage brut sur un projet de petite envergure.

3. Flask

Autre point important abordé lors de ce projet, l'utilisation d'un framework. Plus particulièrement Flask. Le plus dur dans la mise en place de Flask aura était le début. En effet, suite aux cours que j'ai pu lire sur le sujet, il y a tellement de petites choses à mettre en place pour que tout fonctionne, que c'était déroutant au début du projet.

En revanche, une fois que tout est en place, il s'agit là d'un outil tout à fait pertinent sur un projet comme celui-ci apportant un certain nombre de simplification très appréciée (réponse HTTP très simple a réaliser au niveau des "routes", gestion des templates ultra modulaire, débogage aisé grâce au log "live", etc.).

Fait amusant, ce n'est que sur la fin de mon projet que j'ai compris comment faire en sorte de lancer l'application Flask depuis mon ordinateur et de pouvoir consulter l'application directement sur mon téléphone. Et cela tenait seulement à une option et une adresse IP.

4. API

Le cœur du projet s'appuie sur l'utilisation de deux API. Celle fournie par MapBox pour l'utilisation de cartes et permettant de trouver l'adresse d'un lieu, et celle fournie par Wikipédia permettant de retrouver une liste d'articles situés aux alentours de coordonnées géographiques.

En soit, les appels à ces deux API n'ont pas représenté de grandes difficultés En revanche je me suis aperçu qu'il n'est pas toujours évident au premier abord d'obtenir un résultat bien précis tant les possibilités offertes sont vastes.

J'ai donc passé beaucoup de temps à faire des tests de requêtes, lire les documentations fournies. Et finalement à force de persévérance obtenir finalement le résultat souhaité.

IV. Conclusion

Au final, ce projet m'aura apporté énormément de satisfaction. Il s'agit du deuxième projet sur le parcours permettant la mise en place d'une application avec une interface "graphique". Et donc une application un peu "waouh" que l'on ai fier de montrer aux autres (même si les "profanes" ont souvent une réaction du style "oui et alors ?")

Il m'aura surtout appris à bien mettre en œuvre un précepte que tout bon développeur doit suivre dans sa vie professionnelle "Read The Damn Manual" (pour la version polie). Puisque ce projet, au-delà du fait que j'ai du utiliser de nouveau langage nous pousse à utiliser des outils "tierce" que ce soit des API, des packages ou des frameworks. Or les différents tutoriels et cours que l'on peut trouver au grès des recherches internet sont un bon moyen de se mettre le pieds à l'étrier, mais quand il s'agit de pousser l'utilisation plus loin, rien de tel que la documentation officielle !