1920_INFO2_ServicesWeb_Projet

Service d'administration de ChatBot

Dans ce projet, il vous est demandé d'implémenter un service **d'administration** de **ChatBot**. Un ChatBot utilise la technologie Rivescript comme **cerveau**. Le ChatBot peut opérer sur plusieurs **interfaces** : une interface web dédiée, sur Mastodon, Discord, Slack, ...

ChatBot
Un ChatBot n'est pas un pas un poisson d'eau douce (ça, c'est un chabot). C'est un robot conversationnel. En d'autres termes, c'est un logiciel qui propose une interface en langage naturel.

Cerveau (brain rivescript)
Rivescript (https://www.rivescript.com/) est d'abord un langage de description d'interactions « intelligentes » pour des ChatBot basé sur la reconnaissance de schéma (Si tu me dis « Bonjour » alors je te réponds « Salut »). Ces interactions sont décrites dans des fichiers (extension .rive) appelés « brain » (cerveau) car l'ensemble des fichiers .rive donnés à un ChatBot constitue sa « personnalité » (voir https://www.rivescript.com/docs/tutorial#tutorial). Ainsi, vous aller pouvoir créer plusieurs ChaBot ayant chacun sa spécialité/fonction/personnalité, selon les fichiers .rive chargés.

Interface
Rivescript propose plusieurs implémentations (Javascript et Java entre autres). Ceci permet de connecter le cerveau à diverses interfaces textuelles comme le terminal, un dispatcher de sms, ... ou tout simplement le protocole HTTP (protocole textuel question/réponse) et au-delà, toute API basée sur HTTP. Si les fichiers .rive constituent le cerveau de votre ChatBot, ces interfaces sont autant «d'oreilles » et de « bouches ». Dans un premier temps, votre ChatBot sera présent sur une interface Web créée par vous : une simple interface de type SMS.

Administration
Dans ce projet, l'administration des ChatBots se fait à travers une API REST. Administrer un ChatBot, s'est le créer, et le détruire... c'est aussi lui modifier sa personnalité et consulter son état. On peut aussi lui donner une nouvelle interface (lui donner la possibilité d'aller sur Discord par exemple), ou lui enlever.

Use cases
- à travers une interface d'administration, je crée un nouveau ChatBot appelé « Steeve » ayant des fonctionnalités minimales (fichier « standard.rive » de https://www.rivescript.com/try). Je peux alors communiquer avec Steeve à travers une seconde interface (interface de communication, la « bouche »), sur une adresse (et port) propre à Steeve, après m'être identifier avec un login.
- à travers une interface d'administration, je sélectionne un fichier « cerveau » dans une base et je l'ajoute à Steeve. En relançant l'interface de communication, je permets à Steeve de prendre en compte le nouveau fichier. (BONUS : Steeve charge son nouveau cerveau au cours de la conversation).

- Grace à son nouveau cerveau, Steeve peut maintenant garder en mémoire des informations personnelles sur moi en reliant, par exemple, mon login avec ma couleur favorite. (BONUS : stocker le profil collecté sur une base MongoDb en ligne).
- à travers l'interface d'administration, je sélectionne un fichier de paramètres de connexion à Discord (cf https://discordjs.guide) ou Mastodon (utiliser https://botsin.space/api/v1/) et ordonne à Steeve d'y être présent. Je peux alors parler avec une instance de Steeve. (BONUS : L'instance de Steeve qui parle sur botsinspace et celle qui parle sur l'interface de communication partagent les mêmes données collectées.) (BONUS : Je peux aussi choisir une connexion à Slack).
- à travers l'interface d'administration, je supprime l'accès de Steeve à Discord.
- À travers l'interface d'administration, je peux voir l'état (les interfaces actives, a minima) des différents ChatBots (Eude, Hubert, Yann et Adam... tous de la famille Lefrigo). (BONUS : l'interface diffuse les conversations).
- BONUS : Je peux recouper des informations d'un utilisateur recueillies par différents ChatBot.

Cadre du projet

- Ce projet s'effectue en binôme (placement indiqué dans la suite).
- L'API est en NodeJS.
- L'architecture de l'API est REST
- Les livrables sont attendus le 08/06.
- Les livrables sont constitués d'un lien vers un dépôt git et d'un lien vers un screenCast démontrant les fonctionnalités de votre projet.

Liste des binômes

1. Mathieu **Laumonnier** et Elisabeth **Barré**
2. Louis **Le Guen** et Nathan **Collobert**
3. Tony **Chouteau et** Sébastien **Hert**
4. Daniel **Zhu** et Théo **Mingorance**
5. Dejan **Paris** et Adam **Rivière**
6. Manon **Doyelle** et Delphine **Richard**
7. Marc **Vacquant** et Alex **Jobard**
8. Philippe **Martin** et Simon **Jourdan**
9. Divi **De Lacour** et Evan **Le Hellard**
10. Alix **Garot-Adrian** et Cedrine **Croissant**
11. Louis-Maxime **Piton** et Marie **Peter**
12. Thibault **Gaudier** et Simon **Lerin**
13. Anthony **Bokola** et Hala **Elhammoumi**
14. Thomas **Lepercq** et Julie **Boudebs**
15. Yann **Dauvin** et Clement **Van Straaten**
16. Virgile **Petermann** et Pierre **Leroy**
17. Adel **Bendib** et Hugo **Bernat**
18. Celia **Ellmann**
19. Thibaut **Lausecker**
20. Thioba **Ndoye**
21. Maxime **Roques**