Présentation du projet 3

Thomas BAUDIN CDNT21

1) Technologies

- Django : Framework en langage python. Peut-être utilisé pour réaliser
  - L'API
  - Le back-office
  - Le front-end
- Bootstrap : Framework CSS. Utilisé pour
  - Le front-end
- React-Native : Framework Javascript utilisé pour
  - L'application mobile
- MySQL et MySQL Workbench
  - La base de donnée

2) Concept

A) Basique

Ce site aura pour but de permettre le partage de code informatique. Les utilisateurs pourront :

- Uploader du code
- Lire (et donc télécharger) le code des autres utilisateurs
- Commenter, noter le code des autres utilisateurs

B) Pour aller plus loin

I) Résumé

Le site devrait implémenter un système de « Combat de bot » à la manière de Codingame (https://www.codingame.com/multiplayer/bot-programming/tron-battle).

Les utilisateurs posterait du code (un Bot) qui réagirait à des inputs (via l'entrée standard), et enverrait des outputs (via la sortie standard). L'api piocherait des Bots aléatoirement à intervalle régulier, et les ferait s'affronter, ajustant leurs score en fonction de leurs performances.

Tout cela se ferait idéalement en Python.

II) Sécurité

Faire s'exécuter du code fournit par une personne tierce représente un risque de sécurité non négligeable (rm -r ~…) Il faudra donc faire se lancer le code dans un conteneur isolé (comme **Docker**).

III) Fonctionnement

On peut utiliser la méthode **Popen** du module **subprocess** pour lancer un programme Python depuis un autre programme Python.

On peut ensuite lancer des threads (**threading.Thread**) qui lanceront des méthodes qui prendront en argument la sortie standards des programmes pour pouvoir lire leurs réponses de manière asynchrone.

On peut également envoyer des informations aux programmes via leurs entrée standard en faisant **process.stdin.write**, où process est le nom du programme.

On n'a plus qu'a lancer une boucle while où on exécute les actions suivantes :

1. Envoyer les informations courantes aux programmes
2. Lire leurs réponses
3. Calculer le nouvel état du 'jeu' en fonction de leurs réponses. Si un programme met trop de temps à répondre, il perd automatiquement.
4. Dès qu'on programme gagne, on termine le programme principale, et on ajuste le score des morceaux de codes.

C) Pour aller encore plus loin

Il serait judicieux d'implémenter une interface graphique en front pour que les utilisateurs puissent voir le dernier match de leurs bot. Cela rendrait le tout ludique, et également plus fair-play.

Le plus logique pour implémenter ça serait de charger toute les données du match en front, puis d'utiliser une bibliothèque graphique pour afficher le déroulement.

3) Partie mobile

La partie mobile fera la même chose, minus l'envoi de code (et peut-être l'interface graphique le cas échéant).