INFO834 – TP1 – Un cache Redis pour EtuServices

Vous venez d'être recruté.e pour effectuer une mission auprès d'une startup *EtuServices* qui propose aux étudiants 2 services : la vente et l'achat d'articles (DVD, livres...). La startup commence à être victime de son succès avec une utilisation de plus en plus soutenue de ses services via son application web. L'objectif de la mission est de limiter le nombre de connexions au site de l'entreprise.

Vous allez pour cela devoir consigner le nombre de connexions par utilisateur enregistré et connecté. La startup considère qu'il est possible de « requêter » leurs services à raison de 10 appels par fenêtre de 10 minutes.

Outils/langages à utiliser : une interface Web (PHP **ou** Nodes.js API REST), MySQL (gestion des utilisateurs), Python avec l'interface Redis (gestion du nombre de connexions et des appels de services).

Livrables (dépôt moodle) **:** Un rapport expliquant vos choix de conception + lien vers votre dépôt Github du projet.

Étapes du projet (pas nécessairement dans cet ordre)

N.B. Bien réfléchir à l'architecture globale avant de commencer à coder. Les différents composants du système doivent être testés individuellement avant de les intégrer.

1. Créer un dépôt Git sur Github pour le projet
2. Cloner votre projet sur votre machine
3. Créer sur le serveur local de votre ordinateur un dossier « EtuServices » qui contiendra les fichiers 3 pages web qui nous intéressent : accueil.php, login.php et services.php.
4. Créer sur votre BD MySQL une table utilisateurs pour stocker leur nom, leur prénom, leur email et leur mot de passe.
5. Lancer le serveur Redis (cf. TD1).
6. En utilisant votre environnement Python préféré (Idle, Spyder...), utiliser la bibliothèque Redis (https://redis-py.readthedocs.io/en/latest/).
7. Écrire le programme Python qui permet de lancer un client Redis pour la gestion des connexions et utilisations des services Vente et Achat. A chaque fois que l'utilisateur se connecte sur la page web de connexion, le serveur web vérifie si l'utilisateur est autorisé à se connecter et à accéder aux services.

8. Si l'utilisateur est connu, des informations de connexion seront fournies à votre programme Python qui à son tour demande à Redis de gérer le nombre de connexions. L'incrémentation ne se fera que si l'utilisateur n'a pas dépassé les 10 connexions dans une fenêtre de 10mn. Dans le cas contraire, la connexion ne sera pas autorisée.
9. Tester par l'intermédiaire de redis-cli et de votre serveur que tout fonctionne correctement.
10. **Pour aller plus loin :**
    - Dans la base de données 1 de Redis : stocker les appels d'un service (Vente ou Achat) par un utilisateur connecté
    - Afficher les statistiques suivantes : les 10 derniers utilisateurs qui se sont connectés ; le top 3 des utilisateurs qui se sont connectés le plus ; les utilisateurs qui ont le moins utilisé les services ; le service le plus utilisé...

Commandes utiles PHP :

- *$output = shell_exec($cmd)* ou *escapeshellcmd($cmd)* pour lancer l'exécution d'une commande externe $cmd (e.g. Script Python).
- *var_dump($output)* pour afficher les sorties de *shell_exec.*