INFO 834 TP1 Redis

Dans ce TP nous avons développé une application web qui utilise un serveur wamp, une base de données MySQL et Redis pour gérer et limiter les tentatives de connexion des utilisateurs. L’objectif est d’empêcher un utilisateur de se connecter plus de 10 fois en 10 minutes, afin d’éviter les abus et d’améliorer la sécurité.

Fonctionnalités :

Lorsque l’utilisateur tente de se connecter un script python est appelé pour :

1. Une image contenant texte, capture d’écran, Police

   Le contenu généré par l’IA peut être incorrect.Vérification des identifiants
   * L’application interroge une base de données MySQL pour vérifier que l’email et le mot de passe sont corrects.
2. Suivi des connexions avec Redis
   * Redis stocke un compteur des tentatives de connexion pour chaque utilisateur.
   * Ce compteur est associé à une clé unique.
   * Chaque fois que l’utilisateur se connecte, ce compteur est incrémenté.
3. Temps de connexion
   * Une image contenant texte, capture d’écran, Police

     Le contenu généré par l’IA peut être incorrect.A chaque fois qu’une connexion est créée une variable stocke l’heure de la connexion.
   * Cela permet de vérifier le délai depuis la dernière connexion.
4. Blocage des connexions si la limite est atteinte
   * Une image contenant texte, capture d’écran, Police, ligne

     Le contenu généré par l’IA peut être incorrect.Si un utilisateur dépasse 10 tentatives de connexion en moins de 10 minutes, Redis empêche toute nouvelle tentative.
   * L’application affiche alors un message indiquant que l’accès est temporairement refusé.

Un deuxième script python sert quand a lui a afficher les différentes statistiques :

1. Récupérer les utilisateurs récents :

* Liste les derniers utilisateurs ayant effectué une action dans le système.

1. Obtenir les utilisateurs les plus actifs :

* Identifie les utilisateurs ayant effectué le plus de connexions.

1. Statistiques d’utilisation des services :

* Une image contenant texte, Police, capture d’écran, algèbre

  Le contenu généré par l’IA peut être incorrect.Fournit des informations sur les services les plus utilisés et les utilisateurs les moins actifs.

Utilisation de Redis et des clés utilisées

L'utilisation de Redis permet de gérer les connexions et de suivre l'activité des utilisateurs. Les clés Redis suivantes sont utilisées dans cette application pour stocker et manipuler les données liées aux utilisateurs et à leurs connexions :

**1. redis\_key = f"user:{email}:services"**

* Rôle : Stocke la liste des services utilisés par un utilisateur. Chaque élément de la liste contient un timestamp et un type de service.
* Utilisation : À chaque fois qu'un utilisateur utilise un service, l'élément correspondant est ajouté à la liste.
* Commande associée : *r.rpush(redis\_key, f"{current\_time}:{service\_type}")*

**2. redis\_key = f"user:{email}:connections"**

* Rôle : Suivre les connexions d'un utilisateur. Chaque élément dans la liste représente une connexion, avec un timestamp.
* Utilisation : Chaque fois qu'un utilisateur se connecte, un timestamp est ajouté à cette liste.
* Commande associée : *r.rpush(redis\_key, current\_time)*

**3. redis\_key\_connections\_count = f"user:{email}:connections\_count"**

* Rôle : Compter le nombre total de connexions d'un utilisateur.
* Utilisation : Chaque fois qu'une connexion est enregistrée, ce compteur est incrémenté de 1.
* Commande associée : *r.incr(redis\_key\_connections\_count)*

**4. redis\_key\_recent\_users = "recent\_users"**

* Rôle : Liste des derniers utilisateurs ayant effectué une action dans le système. La liste est limitée aux 10 derniers utilisateurs pour ne pas surcharger Redis.
* Utilisation : Chaque fois qu'un utilisateur effectue une action (connexion ou utilisation de service), son email est ajouté à la liste des utilisateurs récents.
* Commande associée : *r.rpush(redis\_key\_recent\_users, email)* et *r.ltrim(redis\_key\_recent\_users, -10, -1)* pour garder seulement les 10 derniers.