Projet : Gestion des Utilisateurs Bannis dans une Application

Objectif

Le but de ce projet est de concevoir et de mettre en œuvre une solution de gestion des utilisateurs bannis dans une application en utilisant Azure Services. L'objectif est de gérer efficacement un grand nombre de requêtes en identifiant les utilisateurs et en renvoyant une erreur pour ceux qui ont été bannis de l'application.

Scénario Réel

Dans une entreprise tierce, il est essentiel de garantir que seuls les utilisateurs autorisés peuvent accéder à une application. Cependant, avec un grand nombre d'utilisateurs, il devient difficile de gérer manuellement la liste des utilisateurs bannis. La solution consiste à automatiser ce processus en utilisant Azure Services.

Architecture de la Solution

La solution est basée sur les principaux services Azure suivants :

Azure Functions : Nous utilisons des fonctions HTTP pour gérer les requêtes des utilisateurs. Ces fonctions sont évolutives et peuvent gérer un grand nombre de demandes simultanées.

Azure Cosmos DB : Nous utilisons une base de données MongoDB hébergée sur Azure Cosmos DB pour stocker les informations sur les utilisateurs, y compris leur statut (banni ou autorisé).

Azure Portal : Nous utilisons le portail Azure pour configurer et déployer nos fonctions, surveiller les performances, et gérer notre base de données.

Implémentation de la Solution

Nous avons implémenté la solution de la manière suivante :

Création d'une Fonction HTTP : Nous avons créé une fonction HTTP Azure qui reçoit les demandes des utilisateurs avec leur nom d'utilisateur en tant que paramètre.

Interrogation de la Base de Données : À l'intérieur de la fonction, nous interrogeons la base de données Azure Cosmos DB pour rechercher l'utilisateur en fonction de son nom d'utilisateur.

Vérification du Statut : Si l'utilisateur est trouvé dans la base de données, nous vérifions son statut. S'il est banni, nous renvoyons une erreur (par exemple, un code d'état HTTP 403). Sinon, nous autorisons l'accès.

Rapport des Résultats : Nous renvoyons une réponse appropriée au client, que ce soit un accès autorisé ou une erreur.

Documentation et Réflexion

Tout au long du projet, nous avons documenté nos décisions de conception, les étapes de mise en œuvre, et les défis rencontrés. Après avoir implémenté la solution, nous avons évalué son efficacité dans la gestion des utilisateurs bannis et réfléchi à d'éventuelles améliorations.

Résultats

Vous pouvez trouver le code source, les fichiers de configuration, les scripts, et d'autres artefacts du projet dans le référentiel GitHub. Vous y trouverez également des captures d'écran et des liens actifs pour démontrer que notre solution fonctionne correctement.

Ce projet démontre comment Azure peut être utilisé pour résoudre un problème du monde réel en automatisant la gestion des utilisateurs bannis dans une application, en garantissant la résilience et la scalabilité de la solution.

Une image contenant texte, diagramme, capture d’écran, ligne

Description générée automatiquement

1. Requête du client pour se connecter.
2. Requête sur la bdd pour récupérer le statut du client
3. Envoie du statut de l’utilisateur
4. Réponse et connexion du client selon son statut
5. Requête de l’admin pour ajouter ou éditer le statut d’un client
6. Requêtes de la azur function sur la bdd pour récupérer ou non le client et modification/ajout de celui-ci
7. Réponse de la bdd (envoie ou non de l’user selon s’il existe)
8. Réponse à l’admin sur l’état du statut mise à jour ou du client ajouté