MAXIME SIMARD
Baccalauréat en génie informatique

LABORATOIRE 4

Travail
présenté à

Monsieur Jérémy Bouchard
dans le cadre du cours
6GEI311 - Architecture des logiciels

Université du Québec à Chicoutimi
Le 11 novembre 2021

Composants

Base de donnée

La base de données du logiciel est l'endroit où les données recherchées sont sauvegardées. Elle permet donc d'enregistrer les tweets retournés lorsqu'on effectue des requêtes à l'API de Twitter. Avant d'ajouter un nouveau tweet à la base de données, une vérification s'effectue afin de s'assurer que le tweet retourné est dans un format valide (sous forme de dictionnaire). Lorsqu'on tente d'accéder à la base de données, si cette dernière est invalide, par exemple quelqu'un a changé son format accidentellement, la base de données retournée sera une base de données vide afin de s'assurer de retourner un format valide.

Serveur

Le serveur est la partie du logiciel qui gère les différentes requêtes des utilisateurs. Lorsqu'il reçoit une requête, si l'url demandé est invalide, on redirige l'utilisateur vers la page Search.html. Sinon, lorsque l'url est valide, on lui retourne la page souhaitée. Avant d'effectuer une demande de recherche à l'API de twitter, on vérifie que la demande de l'utilisateur n'est pas vide. Si elle est vide, on renvoie la page de résultats de recherche sans effectuer de recherche. Sinon, on retourne la page avec la recherche effectuée.

Twitter API

La classe TwitterAPI est l'interface qui permet de communiquer avec l'API de Twitter. C'est dans cette classe qu'on crée toutes les informations liées aux requêtes. Par exemple: le header, les paramètres de la requête, la destination (url). Afin d'éviter de faire des requêtes inutiles à l'API de Twitter, plusieurs vérifications sont faites avant d'envoyer une requête. On vérifie que toutes les parties de la requête sont valides. Exemple de vérification:

- La présence d'un token dans le header
- Le nombre de tweets demandé est valide (entre 10 et 100)
- Le mot clé recherché n'est pas vide

Tests

Base de données

Le but de ces tests est de s’assurer de toujours retourner une base de données valide, tout en empêchant d'ajouter des données invalides

Load tweets returns empty list on error

Test que la base de donnée retourne une liste vide si une erreur est détectée lorsqu’on y accède. Exemple d’erreur: la base de données n’est pas de type List.

Can load tweets default, Can load tweets mocked db

Test qu’une base de données valide peut être retournée.

Can save tweets

Test qu’on peut sauvegarder des tweets valides dans la base de données.

Save invalid tweets

Test que les tweets de format invalide ne sont pas sauvegardés dans la base de données.

Serveur

Le but de ces tests est de s’assurer que la navigation et l’utilisation de l’application ne retourne pas d’erreur à l’utilisateur.

Route search, Route display, Route invalid path

Test que les requêtes retoure les urls demandés, et un url par défaut si la requête est invalide.

Invalid json data returned, Search empty query

Test que la page de résultat est tout de même retournée même s’il y a une erreur dans la requête.

TwitterAPI

Tous les tests du TwitterAPI retourne un message d’erreur dans les cas invalide. Le but de ces tests est d’éviter de faire des requêtes inutiles à l’API.

Request no header, Header is dictionary

Test que le header de requête existe et est dans le bon format.

Header no authorization, Header authorization is string, Header empty bearer token

Test que l'autorisation du header existe et est valide.

No url, Url is string, Empty url

Test que l’url fournis à la requête existe et est valide.

No query, Query is string, Empty query

Test que la query dans les paramètres de la requête existe et est valide.

No max results, Max results is int, Request less than 10 max results, Request more than 100 max results

Test que le max results dans les paramètres de la requête existe et est valide (entre 10 et 100).