 **Informaticien/-ne CFC**

Travail pratique individuel 2017 (TPI)

Travail Pratique Individuel 2017

Candidat N°: 122599

Amazon Reader

Nom du candidat : Onur Oktay Dogan

Classe du candidat : 304541

4 juin 2017

Table des matières

[1 Analyse de l’état actuel 3](#_Toc484352142)

[1.1 Analyse de l’état désiré 3](#_Toc484352143)

[2 Exigences de l’application 3](#_Toc484352144)

[2.1 L’import des Articles 3](#_Toc484352145)

[2.2 Sauvegarde des articles dans une base de données 4](#_Toc484352146)

[2.3 Serveur Web 4](#_Toc484352147)

[2.4 La sécurité 4](#_Toc484352148)

[2.5 Client Web 5](#_Toc484352149)

[3 Architecture d’application 5](#_Toc484352150)

[4 Choix des technologies 6](#_Toc484352151)

[4.1 Front-end 6](#_Toc484352152)

[5 Choix du back-end 6](#_Toc484352153)

[5.1 Pourquoi Elasticsearch 6](#_Toc484352154)

[6 User stories 7](#_Toc484352155)

[7 APIs de l’application 8](#_Toc484352156)

[8 Bibliographie 9](#_Toc484352157)

# Analyse de l’état actuel

Il s’agit d’un nouveau développement. Il n’y a pas d’état actuel.

## Analyse de l’état désiré

L’objectif de ce projet est de réaliser une application web qui permet la recherche et la visualisation de plusieurs millions d’articles.

# Exigences de l’application

## L’import des Articles

Pour réaliser ce projet j’ai besoin d’importer un fichier json (de 10 GB) qui contient quelques millions d’articles Amazon avec leur titre, leur prix, un lien vers une image, une liste de catégories.

L’obtention des données est conditionnelle à la citation des documents SIGIR et KDD ci-dessous.

**Citation**

Please cite one or both of the following if you use the data in any way:

**Image-based recommendations on styles and substitutes**  
J. McAuley, C. Targett, J. Shi, A. van den Hengel  
SIGIR, 2015  
[pdf](http://cseweb.ucsd.edu/~jmcauley/pdfs/sigir15.pdf)

**Inferring networks of substitutable and complementary products**  
J. McAuley, R. Pandey, J. Leskovec  
Knowledge Discovery and Data Mining, 2015  
[pdf](http://cseweb.ucsd.edu/~jmcauley/pdfs/kdd15.pdf)

L’importation des données devra se faire de 2 manières différents :

* Directement depuis le client web (package GO)
* Depuis une application desktop (en ligne de commande, en mentionnant le chemin du fichier à importer).

## Sauvegarde des articles dans une base de données

Les données doivent être sauvegardée dans une base de données.

* Les performances en écriture doivent être très bonne (importation en minutes)
* Les performances en lecture qui devront excellentes afin de supporter des recherches complexes.
* Les données devront être sauvegardées dans la base de données.

## Serveur Web

Le serveur web doit offrir des APIs sur lesquelles le client pourra s’appuyer pour récupérer les données.

Il faut s’assurer que les données qui transitent entre le client et la base de données soient cryptées.

Écrire les tests unitaires.

## La sécurité

* Les utilisateurs doivent s’authentifier
* L’application se compose de 3 niveaux d’autorisations prédéfinis (rôles).
  + Les utilisateurs peuvent effectuer des recherches et afficher les articles
  + Les rédacteurs disposent des droits des utilisateurs et peuvent gérer les articles (modification et suppression).
  + Les administrateurs disposent des droits des rédacteurs et peuvent gérer les comptes des utilisateurs (création, modification et suppression).
* Les mots de passe doivent être cryptés et salés.
* Le serveur génère un jeton jwt ne contenant que l’identifiant de l’utilisateur. Le jeton circule avec les requêtes, encapsulé dans un cookie qui ne devra pas pouvoir être visible par le client.
* Le cookie doit avoir une durée de vie limitée à :
  + 1 jour pour les administrateurs
  + 1 semaine pour les rédacteurs
  + 1 mois pour les utilisateurs
* Il faudra générer un certificat (auto-signé).
* La connexion entre le serveur et la base de données doit être également protégée.
* Une protection contre les attaques de type injection sql doit être implémentée.
* La minification du code javascript

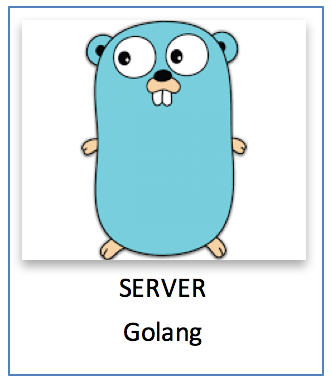
## Client Web

Une librairie permettant d’assurer une qualité à l’interface utilisateur.

* Si l’utilisateur n’a pas de compte, il peut en créer un lui-même. Par défaut, les nouveaux comptes ont un niveau d’autorisation « utilisateur ».
* Une fois authentifié, l’utilisateur peut effectuer des recherches pour afficher les articles :
  + Par mots ou phrases clés contenus dans le titre
  + Par catégorie
  + Par prix
  + Combinaison de plusieurs de ces choix
* S’il dispose du privilège « rédacteur » il peut également modifier ou supprimer des articles.
* L’administrateur doit pouvoir modifier les privilèges ou supprimer des comptes utilisateurs.
* Écrire les tests unitaires et end-to-end

# Architecture d’application

# Choix des technologies



## Front-end

J'ai passé une année à utiliser Angular.js et 6 mois à utiliser Angular 2. Ce sont des frameworks que je maitrise et que j'apprécie beaucoup. Le choix s'impose donc à moi naturellement.

La version actuelle de Angular est cependant la 4. Par chance, elle est identique à la version 2, je n'aurai donc pas à réapprendre de nouveaux concepts.

En ce qui concerne les éléments graphiques, je vais utiliser la bibliothèque Angular Material 2 que je connais également déjà. Je suis le seul chez Xpert à la connaître car elle est encore en version bêta mais je sais qu'elle est suffisamment stable pour ne pas rencontrer de problèmes.

# Choix du back-end

Mon choix s'est tout de suite porté sur Go car c'est le langage que je maitrise le mieux pour le back-end et également parce qu'il est très performant. Il possède d'autres qualités qui en font un choix idéal (voir annexe 1 🡪 Pourquoi Xpert Technologies utilise Go).

De plus, je vais pouvoir très rapidement développer mon server RESTful sans devoir installer et configurer un serveur http.

## Pourquoi Elasticsearch

C’est un moteur de recherche Full-Text très performant. Il est écrit en Java. Il s’agit d’une base de données NoSQL orientée document.

# User stories

|  |  |
| --- | --- |
| **Titre**: Création d’un compte utilisateur | **N°**: 1 |
| **Description**:  En tant qu’**utilisateur** de l’application, je peux créer mon propre compte.  **Critères d’acceptation :**   * Le compte créé reçoit un rôle minimal, c’est-à-dire celui d’utilisateur | |

|  |  |
| --- | --- |
| **Titre**: Accès à l’application | **N°**: 2 |
| **Description**:  En tant qu’**utilisateur**, **rédacteur** ou **administrateur** je dois m’authentifier afin d’accéder à l’application.  **Critères d’acceptation :**   * L’application n’est disponible que pour les personnes authentifiées, sinon l’application n’affiche que la fenêtre du login * Si l’authentification est incorrecte, l’application affiche la fenêtre du login ainsi qu’un message d’erreur * Le nom de l’utilisateur est affiché de manière visible dans l’application | |

|  |  |
| --- | --- |
| **Titre**: Gestion des utilisateurs | **N°**: 3 |
| **Description**:  En tant qu’**administrateur** je peux accéder à la gestion des utilisateurs.  **Critères d’acceptation :**   * Le menu de gestion des utilisateurs ne doit être visible que pour les administrateurs * Éditer un utilisateur existant afin de lui changer son rôle * Supprimer un utilisateur existant * La modification ou la suppression n’est possible que si au minimum un administrateur reste actif | |

|  |  |
| --- | --- |
| **Titre**: Import des articles | **N°**: 4 |
| **Description**:  En tant qu’administrateur je peux importer les articles d’un fichier json.  **Critères d’acceptation :**   * Le menu d’import ne doit être visible que pour les administrateurs * Le fichier à importer doit être du type json | |

|  |  |
| --- | --- |
| **Titre**: Recherche des articles | **N°**: 5 |
| **Description**:  En tant qu’**utilisateur**, **rédacteur** ou **administrateur** de l’application, je peux effectuer des recherches.  **Critères d’acceptation :**   * Par mot ou phrases clés sur le titre * Par catégorie * Par prix (plage de – à –) * Combinaison de plusieurs de ces choix * L’affichage contient 15 articles par page * Une pagination permet d’afficher les articles suivants ou précédents | |

|  |  |
| --- | --- |
| **Titre**: Gestion des articles | **N°**: 6 |
| **Description**:  En tant que **rédacteur** ou **administrateur** je peux éditer et supprimer un article.  **Critères d’acceptation :**   * Seuls les rédacteurs et administrateurs ont accès à la gestion des articles | |

# APIs de l’application

|  |  |  |  |
| --- | --- | --- | --- |
| Méthode | Accès | URI | Action |
| POST | Utilisateur | /user | Créer un utilisateur |
| GET | Administrateur | /user/id | Obtenir un utilisateur |
| PUT | Administrateur | /user/id/role | Mettre à jour un utilisateur |
| DELETE | Administrateur | /user/id | Supprimer un utilisateur |
| GET | Administrateur  Rédacteur  Utilisateur | /users | Récupérer tous les utilisateurs |
| GET | Administrateur  Rédacteur  Administrateur | /item/id | Obtenir un item |
| PUT | Administrateur  Rédacteur | /item/id | Mettre à jour un article |
| DELETE | Administrateur  Rédacteur | /item/id | Supprimer un article |
| GET | Administrateur  Rédacteur  Administrateur | /items | Récupérer les articles en fonction des critères de recherche données dans le body de la requête |
| GET | Administrateur | /import | Importation des articles |
| POST | Administrateur  Rédacteur  Administrateur | /login | Authentification |

# Bibliographie

<https://www.quora.com/What-is-golang-good-for>