Projet

Une bibliothèque d'analyse de données en Java

Ce projet est une opportunité de travailler à plusieurs sur un projet d'envergure. Pour assurer la qualité du logiciel produit, vous mettrez en place une procédure d'intégration continue pour votre logiciel. Le projet s'intéresse à la mise en œuvre d'une bibliothèque de manipulation et d'analyse de données.

1 Informations importantes

- Le projet est à faire par groupe d'**au moins deux personnes** et d'**au plus quatre personnes**.
- **Les solutions ne pourront pas être partagées entre les groupes**.
- **Note** : Ce TP est noté.
- **Date limite** : Le TP est à terminer pour le **16 avril**.

2 Évaluation

Le projet sera noté en fonction de la qualité du code produit, des fonctionnalités mises en œuvre, ainsi que de la maîtrise des outils de développement logiciel.

Veuillez noter que :
- La qualité du code produit sera évaluée notamment au travers du taux de couverture de code par les tests.
- L'évaluation des fonctionnalités tiendra compte du nombre de personnes travaillant sur le projet.

3 Déroulement du TP

3.1 Résultats attendus

Le sujet du TP est très ouvert. Quelques-unes des fonctionnalités que vous pouvez explorer sont brièvement décrites dans la section 5. Cependant cette liste est loin d'être exhaustive. Si d'autres fonctionnalités vous semblent plus intéressantes, vous êtes libre de vous orienter vers celles-ci. Veuillez cependant vérifier auprès des enseignants que la fonctionnalité peut l'intéresser, avant de vous lancer dans sa réalisation.

3.2 Rendu

Le rendu pour ce TP consistera en un simple fichier `txt` qui contiendra l'URL de votre projet sur `Github`, et si applicable, l'URL de votre dépôt sur `Docker Hub`.

Le fichier est à déposer sur Moodle en utilisant le lien prévu à cet effet.

3.3 Contenu de votre dépôt sur Github

Votre dépôt devra contenir au minimum, en plus du code source de votre projet :
- Un fichier `AUTHORS` avec la liste des contributeurs au projet.
- Un ficher `README` qui fournira une documentation aux utilisateurs, c'est-à-dire, une présentation de l'ensemble des fonctionnalités fournies et une explication de leur utilisation. Quelques remarques complémentaires sur le contenu du fichier `README` sont fournies dans la section 4.3.

4 Développement logiciel et intégration continue

Avant de décrire le logiciel que vous allez devoir développer, nous donnons ici les principales instructions à suivre concernant le développement de votre projet.

4.1 Intégration continue

Pour le développement de votre bibliothèque, vous utiliserez une méthode basée sur l'intégration continue. Pour cela, vous devez impérativement utiliser outils suivants :

- Un dépôt `Github` pour stocker votre code source
- `Maven` pour construire les différentes phases de votre projet
- `JUnit` pour les tests unitaires
- Un service d'intégration continue supporté par Github (Travis CI ou CircleCI)

4.2 Qualité du code et des tests

Vous devez utiliser un outil (par exemple, EclEmma) pour évaluer la qualité de vos tests en mesurant la couverture de code. Les rapports de couverture de code devront pouvoir être générés automatiquement en utilisant Maven.

De manière optionnelle, vous pouvez aussi évaluer la qualité de votre code en utilisant l'outil SonarQube [1]. L'évaluation du code avec SonarQube devra elle aussi pouvoir être exécutée en utilisant Maven. Si vous décidez d'intégrer SonarQube, pensez à bien documenter sa configuration et son utilisation dans votre fichier README

4.3 Le fichier README

Le fichier `README` fera office de compte-rendu pour votre TP. Ainsi, il est attendu qu'il contienne au moins les informations suivantes :

- Une description des fonctionnalités fournies par votre service.
- Une description des choix d'outils que vous avez fait.
- Une liste et une courte description des images disponibles sur Docker Hub, le cas échéant.
- Une partie `feedback` dans laquelle vous donnerez votre retour d'expérience sur les différents outils utilisés durant le projet.

4.4 Utilisation de Docker

De manière optionnelle, vous pouvez mettre en pratiques vos connaissances de Docker acquises lors des séances précédentes.

Vous pouvez créer un compte gratuit sur Docker Hub pour avoir votre propre dépôt public d'images Docker. En utilisant le service `Docker Hub`, vous pouvez envisager d'utiliser Docker de différentes manières :

- Vous pouvez utiliser le construction automatique d'images (`automated build`) de Docker Hub pour avoir en permanence une image disponible avec la dernière version de votre bibliothèque pour lesquels les tests unitaires sont prêts à être exécutes.
- Vous pouvez construire une image qui à l'exécution du conteneur déroulera un scénario de démonstration des fonctionnalités de votre bibliothèque.

4.5 Github student developper pack

En tant qu'étudiant à l'université, vous êtes éligibles pour l'offre Student developper pack de Github. Cette offre permet, entre autres, de pouvoir créer des dépôts privés gratuitement sur Github, ou encore d'obtenir des crédits sur la plateforme Cloud d'Amazon (AWS).

De manière optionnelle, vous pouvez envisager de démarrer des conteneurs sur AWS pour tester le bon fonctionnement des images Docker que vous avez créé.

Dans ce cas, détaillez votre manière de procéder dans votre compte-rendu.

1. `https://www.sonarqube.org/`

4.6 A propos des collaborations sur Github

Voici comment procéder pour travailler à plusieurs sur un projet sur Github :

1. Chaque membre du groupe doit se créer un compte sur Github
2. Un des membres du groupe crée le projet (*repository*) sur son compte Github. Il devient l'administrateur du projet.
3. Il ajoute ensuite les autres membres du groupe en tant que collaborateurs au projet (Sur la page principale du *repository*, cliquez sur l'onglet *Settings*, puis sur l'onglet *Collaborators*).
4. Tous les membres du groupe peuvent maintenant participer au projet, mais seul l'administrateur peut associer le projet à un compte `Travis CI` ou `Docker Hub`.

5 Bibliothèque d'analyse de données

La capacité de générer de la connaissance, et donc de la valeur, à partir d'un ensemble de données brutes est centrale dans notre monde *connecté* où des quantités gigantesques de données sont produites à chaque instant. Dans ce contexte, les langages et bibliothèques permettant de traiter simplement et efficacement de grandes quantités de données deviennent très populaires.

Parmi ces outils pour le traitement de grande quantités de données, le langage Python et plus particulièrement le package `Pandas` est une solution très appréciée des informaticiens. Il n'existe pas de solution équivalente au package `Pandas` pour le langage Java. Durant ce projet nous vous proposons de mettre en oeuvre une petite sous-partie des fonctionnalités offertes par `Pandas` en Java.

L'objectif sera de développer votre propre bibliothèque d'analyse de données en Java en suivant les consignes et recommandations décrites dans la section 4 pour assurer la qualité du code produit.

Dans la suite, nous décrivons quelques fonctionnalités qu'il nous semble intéressant d'intégrer à votre bibliothèques. Vous êtes bien entendu autorisés et invités à en intégrer d'autres.

5.1 Fonctionnalités obligatoires

Dans cette partie, nous décrivons quelques fonctions que vous avez l'obligation d'intégrer à votre bibliothèque dans le contexte de ce projet.

Dataframe : Le type d'objet principal manipulé par Pandas sont les Dataframe. Les Dataframes sont des tableaux en deux dimensions où chaque colonne est identifié par un label et chaque ligne par un index. Chaque colonne stocke des donnes d'un seul type. Cependant deux colonnes différentes peuvent stocker des types différents.

Votre bibliothèque devra supporter la création de dataframes selon au moins 2 méthodes différents :

- A partir d'un constructeur prenant en paramètre le contenu de chaque colonne sous forme d'une structure de données simple (par exemple un tableau).
- A partir d'un constructeur prenant en paramètre le nom d'un fichier `CSV`[2] qui devra être *parsé* au sein de ce constructeur. Dans ce cas, vous devrez déduire du format des données dans le fichier CSV, le type de données à créer pour chaque colonne.

Affichage d'un dataframe Vous devrez fournir à l'utilisateur des méthodes pour afficher un dataframe avec plusieurs variantes :

- Afficher tout le dataframe
- Afficher seulement les premières lignes
- Afficher seulement les dernières lignes

2. A propos du format CSV, voir `https://fr.wikipedia.org/wiki/Comma-separated_values`

Sélection dans un dataframe Vous devez fournir à l'utilisateur des méthodes pour créer un nouveau dataframe en sélectionnant un sous-ensemble des données d'un dataframe existant. Vous devez au moins permettre de sélectionner :

- Un sous-ensemble de lignes à partir de leur index
- Un sous-ensemble de colonnes en utilisant les labels

Statistiques sur un dataframe Vous devez fournir à l'utilisateur la possibilité d'exécuter des calculs de statistiques de base sur les colonnes d'un dataframe, tels que calculer la moyenne des valeurs, trouver le minimum/maximum, etc.

5.2 Fonctionnalités optionnelles

Vous trouverez ci-dessous quelques suggestions concernant d'autres fonctionnalités à ajouter à votre bibliothèque.

- Pour toutes les catégories de fonctionnalités décrites dans la section 5.1, vous avez bien sûr la liberté d'ajouter d'autres fonctionnalités que celle déjà mentionnées si elles vous semblent intéressantes.
- Pour une analyse avancée des données, il est intéressant de pouvoir regrouper les données selon des critères pour ensuite appliquer des opération sur ces groupes. Dans le package Pandas, la fonction `GroupBy` permet de regrouper les données d'un Dataframe selon un critère. La fonction `Aggregate` permet ensuite d'appliquer une opération sur les éléments de chacun des groupes créés. Travailler sur ces manipulations avancées de Dataframes peut être une direction de travail pour l'extension de votre bibliothèque.
- Plus généralement, vous pouvez vous référer à la documentation du package Python `Pandas` pour trouver d'autres fonctionnalités que vous aimeriez mettre en œuvre.