R208 Analyse et traitement de données
BUT R&T : 1A - S2
TD 2 : GitLab - JSON

IUT de Lannion

Préparation du TP

Dans un terminal, déplacez-vous à la racine de votre compte et dans le répertoire *R208/* que vous avez dû créer lors du TD précédent. Placez-vous dans ce dernier répertoire.

1 Découverte de GitLab

1.1 Interface web de GitLab

Un serveur GitLab est disponible sur le réseau de l'IUT, son adresse est : `https://gitlab.iutlan.univ-rennes1.fr/`. Le certificat est auto-signé, il faut donc accepter l'exception de sécurité. Chaque étudiant dispose d'un compte sur ce GitLab et un projet par défaut.

Commencez par vous connecter au serveur GitLab `https://gitlab.iutlan.univ-rennes1.fr/` à l'aide vos identifiants DOMLAN. Vous avez ensuite la possibilité de créer un et un seul projet. Créez un projet (vide) nommé *<login>_ProgRT* où *<login>* est évidemment à remplacer par votre identifiant DOMLAN.

Observez ensuite les différents éléments du projet : branches, membres, etc., et identifiez l'url https de votre projet (copiez-la vous en aurez besoin).

1.2 Accéder à GitLab en local

Depuis un terminal sur votre machine, effectuez les actions suivantes :

- placez-vous dans votre répertoire *~/R208/*,
- clonez votre projet gitlab à l'aide de la commande suivante[1] :
  git -c http.sslVerify=false clone https://gitlab.iutlan.univ-rennes1.fr/.....git

[1] l'option utilisée permet d'éviter l'utilisation de ssl qui n'est pas activé sur le dépôt

- vérifiez la création d'un répertoire contenant le nom de votre projet ainsi que d'un répertoire caché *.git* qui contiendra toutes les meta-données nécessaires à la gestion des versions de votre clone local.

Depuis l'interface web de gitlab :

- créez un nouveau fichier vide nommé *td2.py*.

Retournez sur le terminal sur votre machine, effectuez les actions suivantes :

- vérifiez l'état du clone local vis-à-vis du dépôt gitlab avec la commande *git status* ,
- faites en sorte de mettre à jour votre copie locale pour y intégrer le fichier *td2.py*, git -c http.sslVerify=false pull
- modifiez localement le contenu du fichier *td2.py* (en ajoutant un commentaire avec votre nom), puis mettez à jour le dépôt gitlab avec une nouvelle version de ce fichier (*git -c http.sslVerify=false push*). Retracez sur l'interface web toutes les modifications faites sur le fichier *td2.py*.

2 Manipulation de données JSON

2.1 GitLab

Dans votre répertoire *~/R208/<depotGit>/*, créez les sous-répertoires *TD2/* et *TD2/ExJSON/*. Dans le répertoire *ExJSON/*, copiez le fichier *data.json* disponible sur l'ENT[2] et créez un fichier vide nommé *exjson.py*.

Depuis un terminal en local, faites en sorte que votre répertoire *TD2/* et son contenu soit ajouté à votre dépôt GitLab. Attention votre répertoire *~/R208/<depotGit>/TD2//* est logiquement déjà associé à votre dépôt GitLab, il suffit juste de lui rajouter du contenu (commandes *git add*, *git commit* et *git push*).

2.2 Données JSON

Le fichier *exjson.py* décrit les notes obtenues par des étudiants selon leur groupe de TP. Éditez le fichier *exjson.py* et analyser la structure de ces données au format JSON. Puis, on considérant que nous utilisons le module python *typing* pour définir des types de variable de manière explicite, indiquez dans le cadre ci-dessous le type complet de la variable qui pourra stocker ces données JSON :

.....

[2] `https://foad.univ-rennes1.fr/course/editsection.php?id=187936&sr`

Gestion des tags avec GitLab

Création de tags :

- git tag v1.0

Mise-à-jour du dépôt avec ajout des tags :

- git push origin --tags

2.3 Chargement des données JSON

Dans le fichier *exjson.py*, définissez une fonction qui charge dans une variable le contenu d'un fichier JSON dont le chemin est passé en paramètre. La fonction retourne la variable contenant les données JSON.

En testant avec le fichier *data.json*, vous vérifierez que le type identifié précédemment à partir des données correspond bien à celui de la variable créée.

Versionnez cette version de votre code avec le tag v1.0. Vérifiez sur l'interface graphique de GitLab qu'une nouvelle version du code est bien disponible.

2.4 Traitement des données

Définissez une fonction qui prend en paramètres des données telles que celles chargées précédemment (notes des étudiants des différents groupes), un groupe de tp (grp1a par exemple) et un coefficient (0.9 par exemple). La fonction doit modifier les notes de tous les étudiants du groupe indiqué en paramètre en fonction du coefficient (ici réduction de 10%), puis retourner la structure de données des notes.

Après avoir fait quelques tests en affichant notamment les données retournées par votre fonction, versionnez cette version du code avec le tag v1.1. Vérifiez sur l'interface graphique de GitLab qu'une nouvelle version du code est bien disponible.

2.5 Sérialisation textuelle des données

Définissez une dernière fonction qui prend en paramètre la structure des notes, un nom de fichier et effectue une sérialisation en JSON des données dans le fichier.

Après avoir fait testé cette fonction, versionnez cette version du code avec le tag v1.2. Vérifiez sur l'interface graphique de GitLab qu'une nouvelle version du code est bien disponible.