1 Environnements

1.1 PyCharm

PyCharm est un environnement de développement intégré (IDE en anglais, pour *integrated development environment*) pour la programmation en *Python*. Il est gratuit sous sa forme *community edition* et payant (sauf sous certaines conditions, comme être étudiant) sous sa forme *professional edition*. Ce logiciel est facile à utiliser notamment grâce à son interface ergonomique. Il contient les meilleurs éléments des IDEs modernes avec la complétion de code, la coloration des mots clés, l'analyse du code en temps réel pour trouver certaines erreurs de syntaxe, un débogueur, un profileur, etc.

1.1.1 Installation - Windows

L'installation de PyCharm se fait assez facilement. Il suffit en premier temps d'aller sur le site de PyCharm et d'installer la version *community* ou *professional*. Ces versions sont disponibles au lien suivant : `https://www.jetbrains.com/pycharm/download/#section=windows`. Une fois l'installation lancée, nous recommandons d'activer les options

- ☑ Create Desktop Shortcut/64-bit luncher ;
- ☑ Update context menu/Add "Open Folder as Project" ;
- ☑ Create Associations/.py ;

puisque celles-ci facilitent grandement l'ouverture de projet dans l'explorateur de fichiers.

1.1.2 Création d'un nouveau projet

Il existe essentiellement deux façons de créer un projet avec PyCharm. La première consiste à cliquer sur "new project" à l'ouverture de l'application ou dans le menu "File" si l'application est déjà ouverte sur un autre projet. Une fois fait, il faut entrer les paramètres du nouveau projet. Avec l'aide de la figure 1.1, on voit qu'il faut préalablement indiquer la localisation du projet ainsi que le nom de celui-ci.

Pour ce qui est de l'environnement virtuel, celui-ci permet d'avoir un interpréteur propre au projet. En effet, différents projets vont demander différents prérequis en termes de version de python ou de version de différents modules. Il est donc important d'avoir un environnement virtuel différent pour chacun des projets. Grossièrement, un environnement virtuel est une copie du python de base, donc il est recommandé d'avoir une telle copie pour chaque projet afin que celle-ci soit spécifique au projet courant. Pour des projets plus avancés, il est même requis de faire plusieurs environnements virtuels pour un seul projet. Bref, pour cet environnement virtuel, il faut lui donner un nom qui est normalement tout simplement "venv". Finalement, il faut choisir la version de python qui sera utilisée dans le projet. Si c'est le premier projet et qu'aucun interpréteur python n'a encore été installé, il faut préalablement à cette étape aller installer la version de python voulue à `https://www.python.org/downloads/` [1].

1. Il est recommandé de télécharger l'avant-dernière version de python, car celle-ci est très souvent la plus stable. Alors par exemple, si la dernière version de python est 3.10.1, il est recommandé de prendre la version 3.9.X.

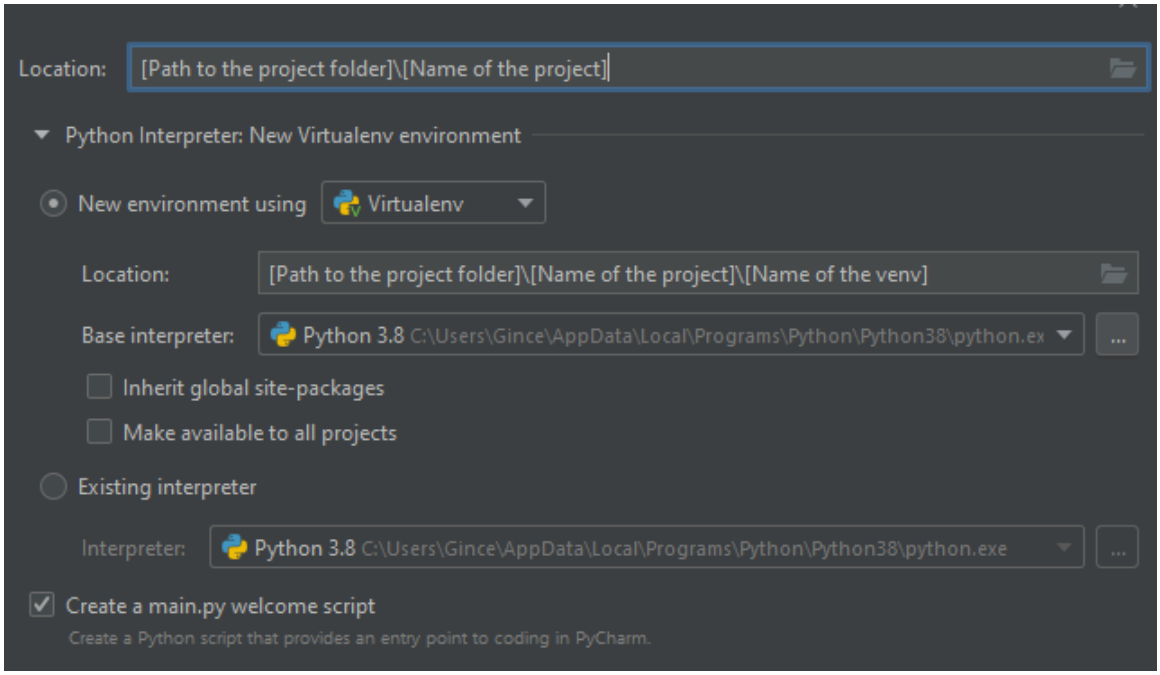


FIGURE 1.1: Fenêtre de création de projet avec PyCharm. Le chemin vers le python de base peut varié mais sa valeur par défaut est montré à la liste 1.1.2.

La seconde façon de créer un projet est la suivante. Dans un premier temps, il faut cloner un répertoire git dans son ordinateur et accéder au dossier créé. Ensuite, il faut faire un clic droit sur le dossier et appuyer sur "Open Folder as a PyCharm Project". Si cette dernière option n'est pas disponible c'est sûrement parce que l'option *Update context menu/Add "Open Folder as Project"* n'a pas été cochée à l'installation de l'application ou bien que l'ordinateur a besoin d'être redémarré. Dans le cas où ce n'est pas possible d'ouvrir le projet de cette façon, il suffit d'ouvrir l'application et aller dans le menu "File/Open" afin d'arriver au même résultat. Par la suite, afin d'ajuster les paramètres de l'interpréteur et de créer l'environnement virtuel, il faut aller dans le menu "File/Settings/Project :Python/Python Interpreter". Il faut ensuite appuyer sur l'icône de paramètres à droite de "Python Interpreter" et appuyer sur "Add". Finalement, il faut créer l'environnement virtuel en allant dans le sous-menu "Virtual Environment". Il sera donc possible d'entrer les mêmes valeurs qu'indiqué à la figure 1.1. Il faut toutefois, prendre en note qu'il arrive que PyCharm ne détecte pas le python de base de l'ordinateur. À ce moment, il faut entrer manuellement le chemin (*path*) vers celui-ci. Ce chemin vous a été donné lors de l'installation de python, mais sa valeur par défaut est énoncé à la liste 1.1.2.

→ (**Windows**) : "C :\Users\[User Name] \AppData\Local \Programs \Python \Python[version] \python.exe" ;

→ (**Mac OS**) : "/usr/bin/python".

Liste 1: *Path* par défaut de python.

1.1.3 Création d'un environnement virtuel

Dans plusieurs cas, vous aurez un projet déjà existant et vous devrez vous préparer à l'utiliser. La première chose à faire est d'ouvrir le dossier content les fichiers du projet avec PyCharm. Pour

ce faire, si vous avez activé l'option *Update context menu/Add "Open Folder as Project"* lors de l'installation de PyCharm, vous pouvez faire clic droit sur le dossier directement et cliquez ensuite sur *"Open Folder as Project"*. Si vous n'avez pas cette option, vous pouvez ouvrir PyCharm et ensuite ouvrir un nouveau projet en sélectionnant ce dossier. Vous avez maintenant le projet ouvert avec PyCharm.

Il est maintenant temps de créer votre environnement virtuel. Il y a plusieurs façons d'y arriver, la façon utilisant l'interface de PyCharm et la façon en lignes de commandes. Pour la façon PyCharm, c'est très simple. Il suffit d'aller dans les *settings* du projet, ensuite aller dans *Project :[Nom du projet]/Python Interpreter*. À ce moment vous verrez que vous pouvez choisir l'interpréteur de votre projet, mais vous voulez installer un nouvel environnement virtuel, alors vous allez cliquer sur *Add*. À ce moment, vous allez voir une interface très semblable à la figure 1.1, remplissez les entrées comme décrites dans cette section. Vous avez maintenant un nouvel environnement virtuel, il est temps d'aller l'activer. Pour ce faire, vous allez dans *Terminal* et vous ajoutez un nouvel onglet nommé *Command prompt*, ce qui va ouvrir un terminal avec votre environnement virtuel activé. Pour vous assurer que cela a bien été fait, vous allez voir "([Nom de votre environnement virtuel])" en avant de votre *path* courant. Si vous n'arrivez pas à suivre les étapes d'activation de votre environnement virtuel, veillez suivre les étapes en terminal. Une fois activé, vous allez vouloir installer les "requirements" du projet, s'il y en a. pour ce faire, vous allez dans votre *Command prompt* et vous lancer la commande *pip install -r [path]/requirements.txt*. De même façon, si vous voulez installer de nouveaux modules python, vous lancer la commande *pip install [nom du module]*. Les figures qui suivent montrent les différentes fenêtres pour créer un nouvel environnement virtuel avec PyCharm.

Pour la version terminale, vous commencez par ouvrir votre projet comme mentionné précédemment. Ensuite, vous ouvrez le terminal et vous lancez la commande suivante : *python -m venv [path du nouvel environnement]/[nom du nouvel environnement virtuel]* ou bien *[path vers le python que l'on veut "copier"]/python -m venv [path du nouvel environnement]/[nom du nouvel environnement virtuel]*. Ensuite, pour l'activer, vous lancez la commande *[path du venv]/Scripts/activate*. Une fois activé, vous allez vouloir installer les "requirements" du projet, s'il y en a. Pour ce faire, vous lancez la commande *pip install -r [path]/requirements.txt*. De même façon, si vous voulez installer de nouveaux modules python, vous lancez la commande *pip install [nom du module]*.

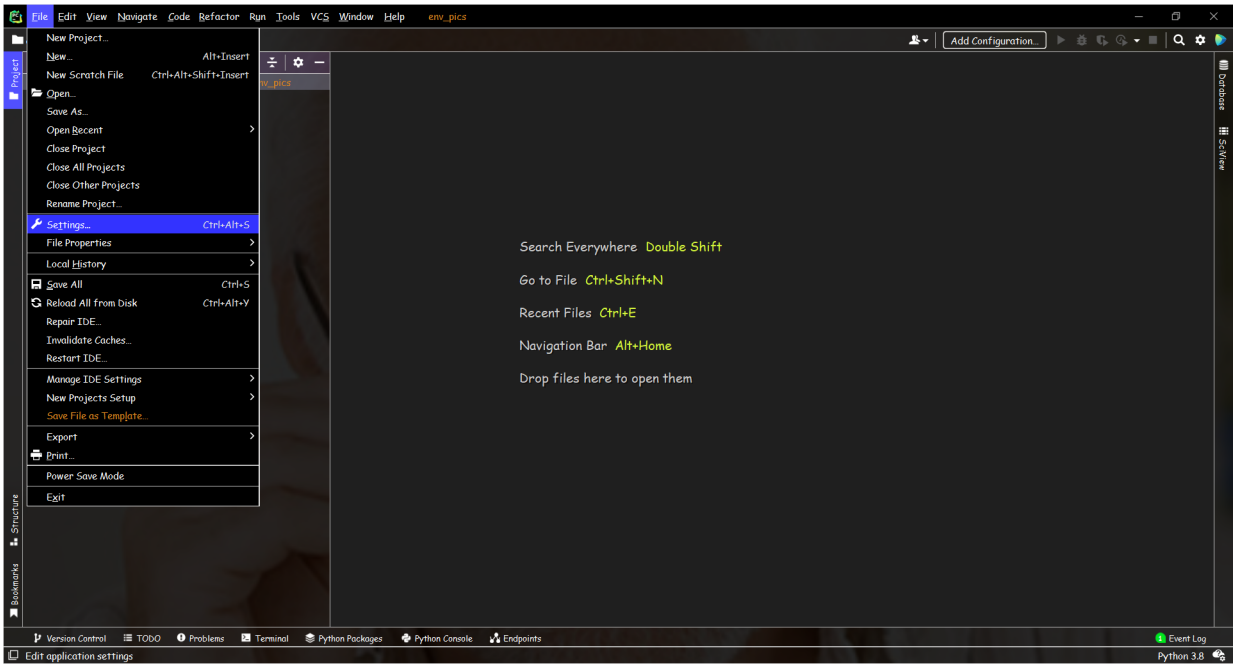


FIGURE 1.2: On appuie sur *File*, puis *Settings*

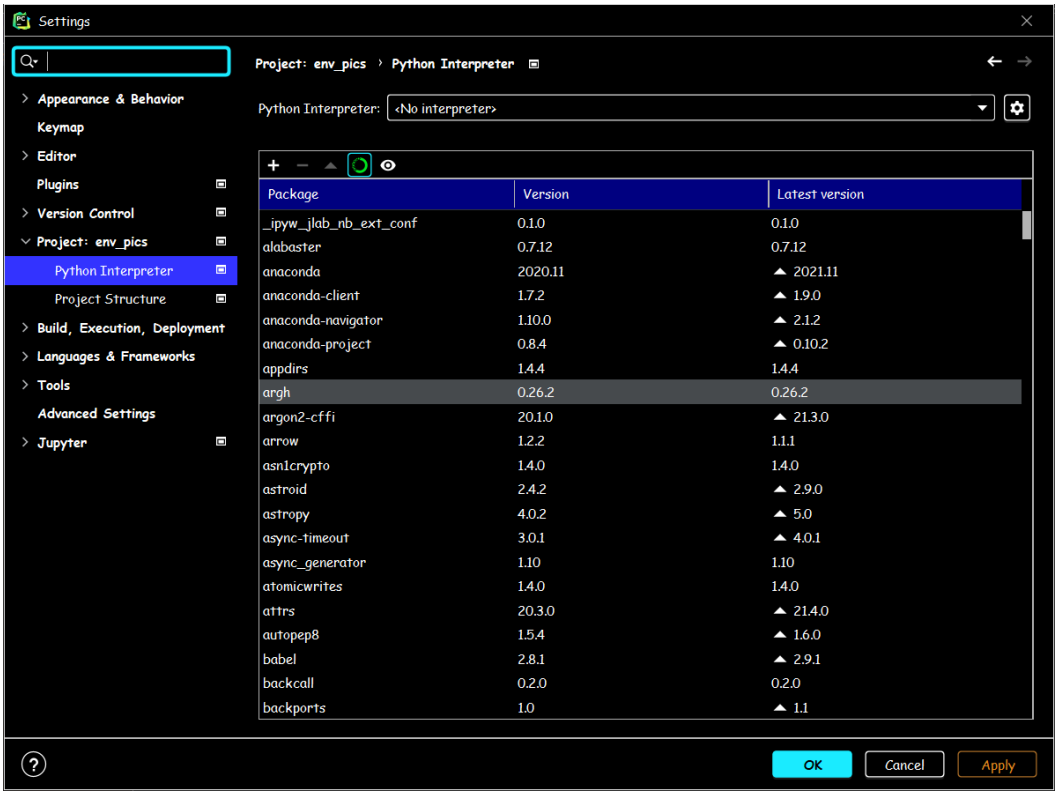


FIGURE 1.3: On appuie sur la petite roue dentée pour ajouter un environnement

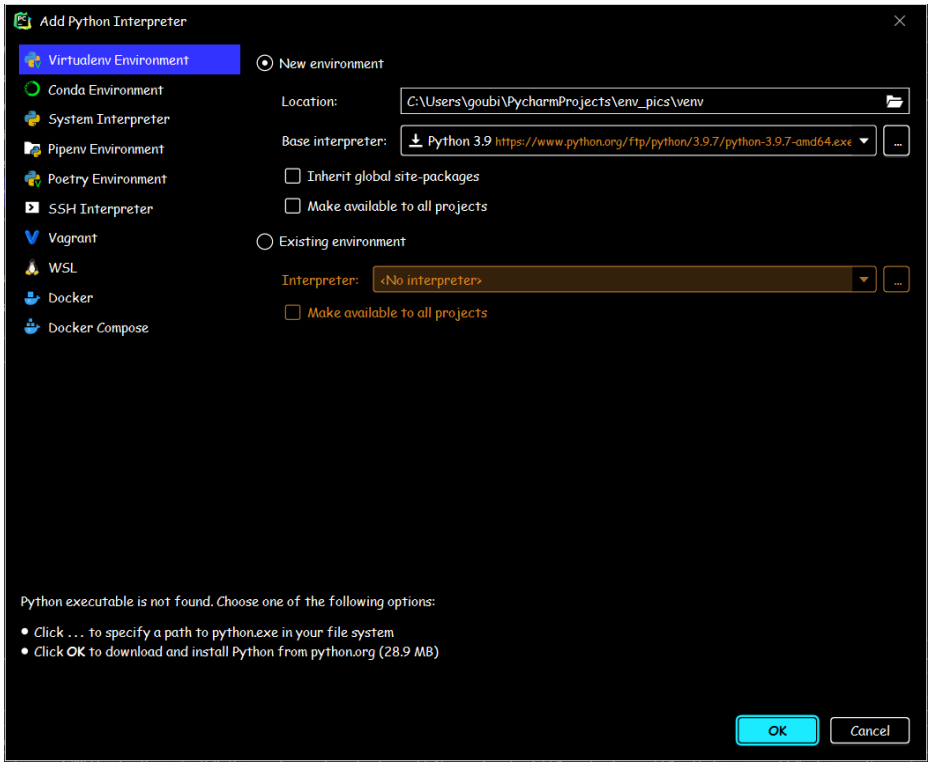


FIGURE 1.4: On trouve une version de *Python* qui nous convient (on peut en télécharger une directement par là). On spécifie où placer l'environnement, si on veut qu'il soit accessible à tous les projets, etc. On peut aussi en prendre un préexistant dans la section du bas.

1.1.4 Cloner à partir de PyCharm

Pour cloner à partir de PyCharm, il faut chercher l'option *Get from VCS* dans l'interface de PyCharm. Si aucun projet n'est ouvert, l'option se trouve en haut à droite de l'interface illustrée à la figure 1.5.

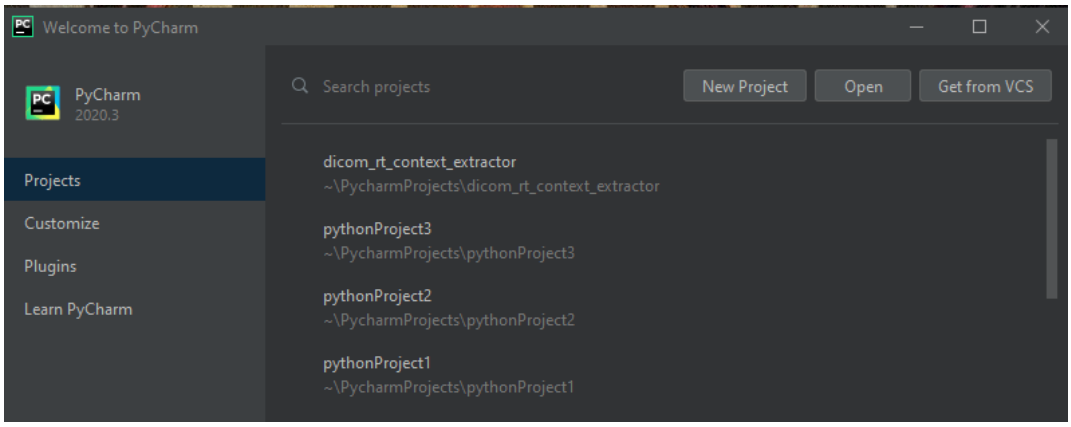


FIGURE 1.5: Interface d'ouverture de projet dans PyCharm

Dans le cas ou un projet est déjà ouvert, il est possible d'aller le chercher dans le menu du haut, voir figure 1.6.

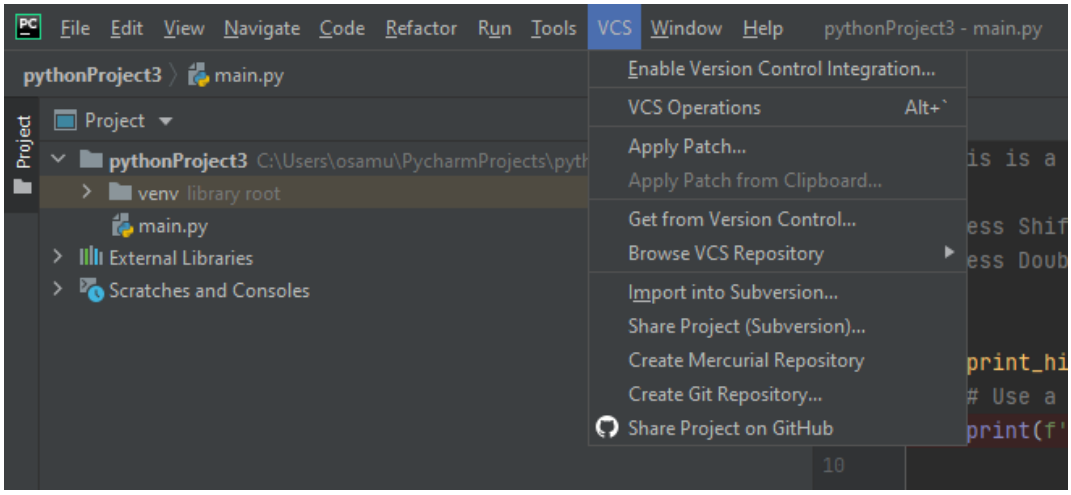
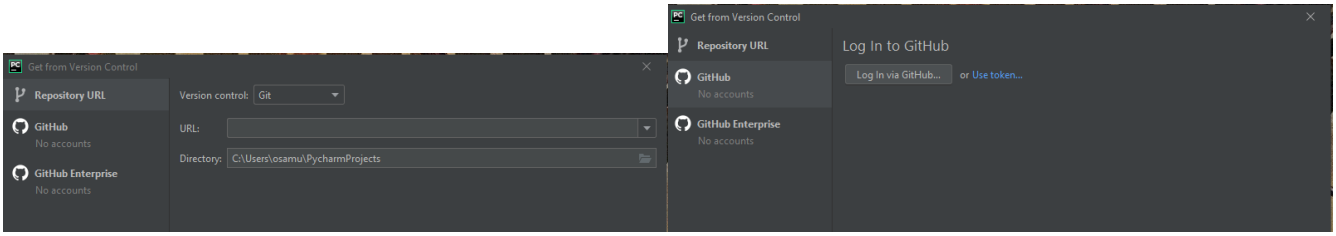


FIGURE 1.6: Interface pour cloner à partir d'un projet ouvert dans PyCharm

Si vous ne trouvez pas l'option lorsqu'un projet est ouvert, vous pouvez cliquer sur *File* en haut à gauche et sélectionner *Close Project*. Aprsès avoir sélectionné l'option, vous aurez deux méthode pour cloner. La premiére en utilisant le url du projet obtenus à partir du serveur git, voir figure 1.7a, la deuxième en vous connectant à Github et en sélectionnant un de vos projets, voir figure 1.7b.



(a) avec l'URL (b) avec la connexion à Github

FIGURE 1.7: Cloner un répertoire git à partir de PyCharm

1.1.5 Le pull, le commit et le push

Lorsque le projet est cloné, les options git devraient apparaître sur l'interface. En haut à droite de l'interface, voir figure 1.8, il y a les raccourcis pour le pull (flèche bleue), le commit (crochet vert) et le push (la flèche verte).

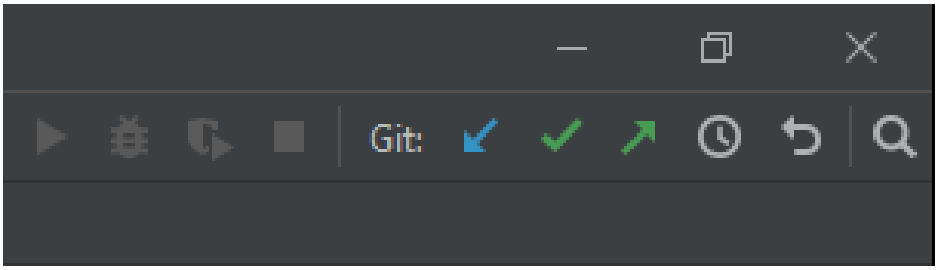


FIGURE 1.8: Accès rapide aux options git dans PyCharm

Ces options se trouvent aussi dans le menu général de git (voir figure 1.9)

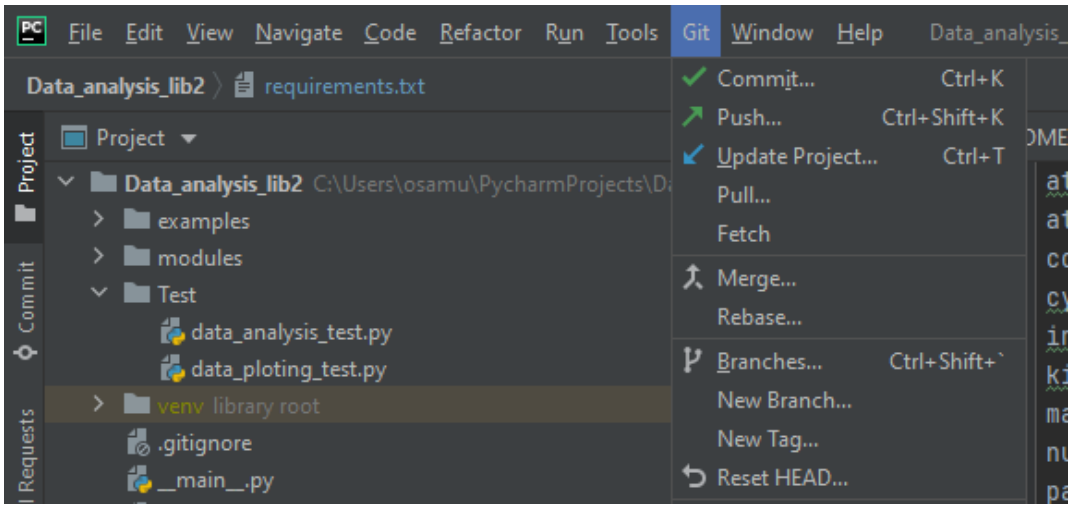


FIGURE 1.9: Liste déroulante des options git dans PyCharm

En cliquant sur commit, le menu suivant apparait, vous permettant de sélectionner les changements à ajouter dans le commit ainsi que le message associé.

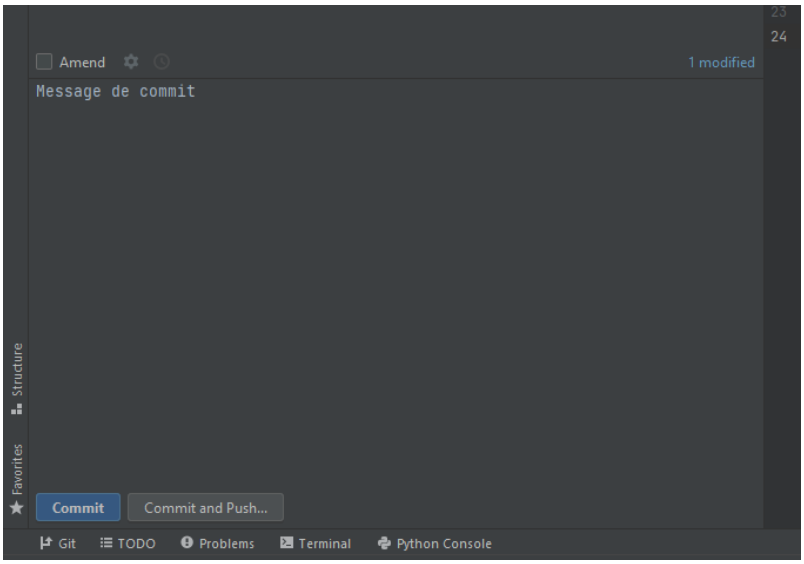


FIGURE 1.10: Interface dans PyCharm pour effectuer le commit

Pour terminer, il reste simplement à faire le bouton push pour envoyer le commit sur le serveur git.

1.1.6 La gestion des branches

Dans le menu de git (figure 1.9), il est possible de voir l'option *branch*. En cliquant sur cette option, il sera possible de choisir sur quelle branche travailler et créer une branche au besoin, voir figure 1.11.

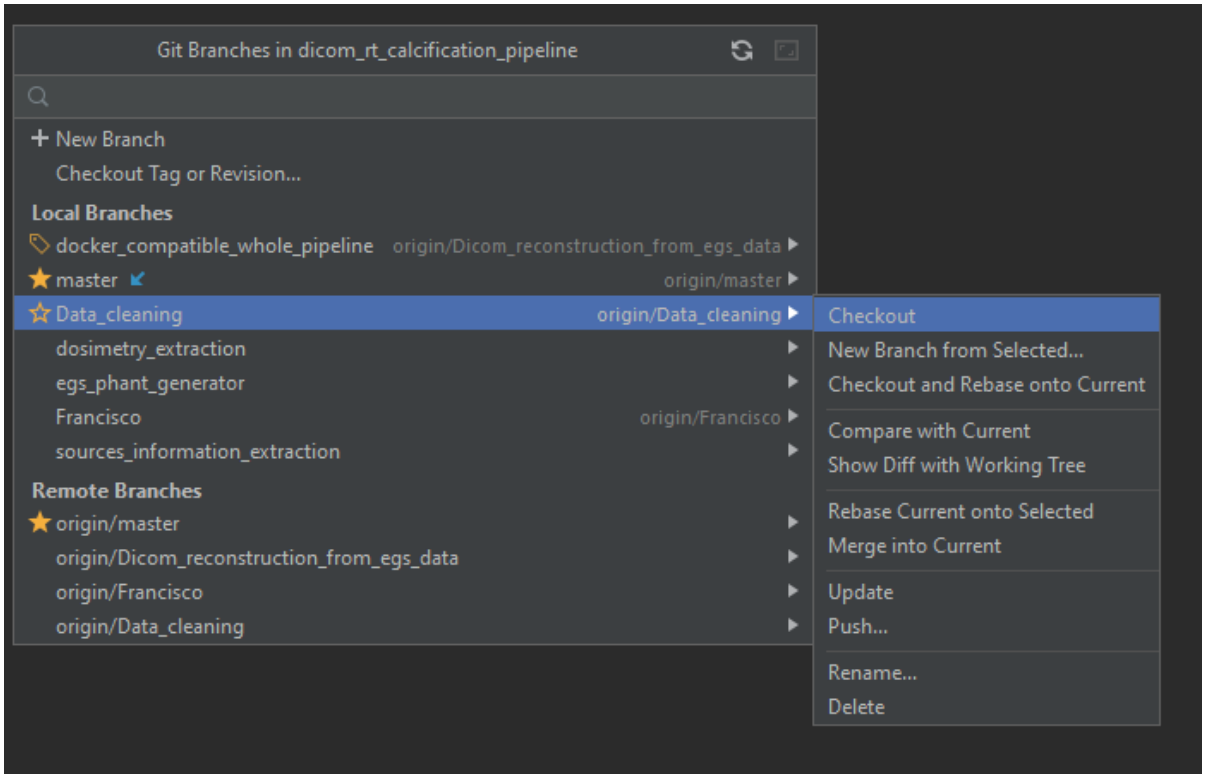


FIGURE 1.11: Section du menu dans PyCharm pour faire la gestion des branches

Il est possible de remarquer qu'il y a plus de branches dans *Local Branches* que dans *Remote Branches*. Dans *Local Branches*, il s'agit des branches présentes dans le répertoire local tandis que *Remote Branches* contient les branches dans le répertoire sur le serveur. Le nombre de branches locales et en ligne peuvent être différent si certaines branches n'ont pas été push vers le serveur en ligne. En effectuant un checkout sur une branche locale, cela sélectionne l'état de la branche locale en question. Il est cependant possible de checkout directement la même branche, mais en ligne. Cela aura pour effet de mettre à jour votre branche locale, que vous le vouliez ou non. Essentiellement, il faut garder en tête que ce qui se passe localement ne se reflète pas automatiquement sur ce qu'il y a en ligne et vice-versa. Une fois que le développement de la branche est terminé, il faut la fusionner dans une autre branche, souvent master. Pour ce faire, il faut utiliser l'option *merge into current* présent dans la figure 1.11. Attention ! Cela fusionne la branche sur laquelle l'option merge est appelée dans la branche sur laquelle vous travailler actuellement. Pour merger dans master, il faut simplement checkout la branche master avant d'effectuer le merge.