

Documentation d'Installation TaskFlow



1. Installation classique	3
1.1. Prérequis	3
• Installation Python 3.9+	3
• Installation Miniconda	3
• Installation de PyCharm	3
• Installation de Git	4
1.2. Étapes d'installation	4
• PowerShell	4
• Cloner le dépôt :	4
• Ouvrir le projet dans PyCharm	5
• Configurer l'interpréteur Python dans PyCharm	5
• Installer les dépendances Python supplémentaires	6
• environnement virtuel Conda	7
• Ouvrir le projet dans PyCharm	8
2. Installation de TaskFlow par docker sur linux	9
1.1 Prérequis	9
• Docker et Docker Compose installés :	9
• Pour installer Docker Compose :	9
• Pour installer Xhost et les outils X11	9
2. Installation de TaskFlow	9
• Cloner le dépôt	10
• Exécution du script d'installation	10
Que fait le script d'installation :	10
• Lancer l'application	10
• Désinstaller le projet	11
3 Points importants	12
X11 et affichage graphique :	12
Droits utilisateur :	12
Dépannage :	12

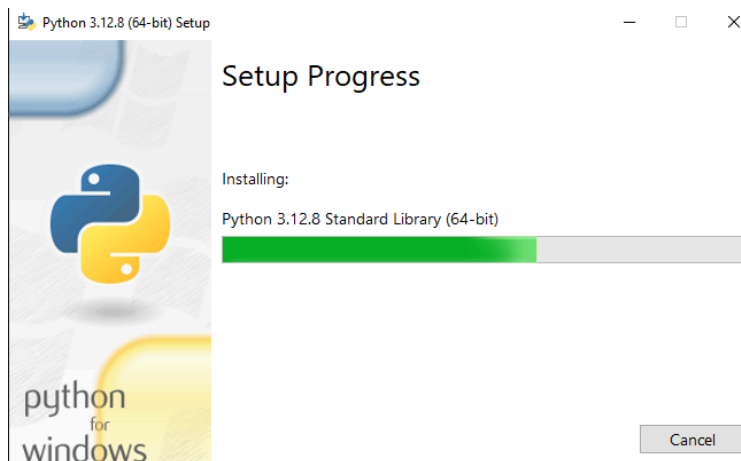
1. Installation classique

1.1. Prérequis

- **Installation Python 3.9+**

Téléchargez et installez depuis python.org.

Pourquoi ? Python est nécessaire pour exécuter le projet TaskFlow et gérer les dépendances.



Une fois installé, vérifiez la version avec la commande suivante pour s'assurer qu'il est correctement installé : `python --version`

- **Installation Miniconda.**

Téléchargez et installez depuis conda.io.

Pourquoi ? Miniconda est utile pour gérer les environnements virtuels et les dépendances du projet de manière isolée.

Vérifiez l'installation avec la commande : `conda --version`

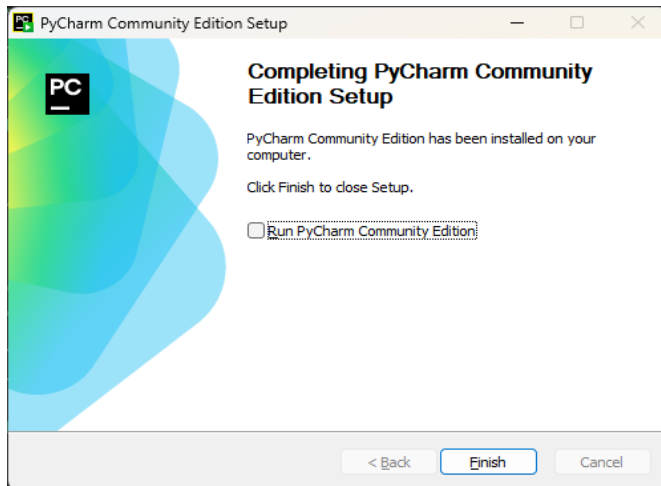
- **Installation de PyCharm**

Téléchargez **PyCharm Community Edition**, un IDE (Environnement de Développement Intégré), depuis jetbrains.com/pycharm.

Pourquoi ? PyCharm offre un environnement convivial pour éditer, tester et exécuter le code du projet TaskFlow.

Suivez les instructions d'installation spécifiques à votre système d'exploitation :

Windows : Lancez l'exécutable téléchargé.



- **Installation de Git**

Téléchargez et installez **Git** depuis git-scm.com.

Pourquoi ? Git est utilisé pour cloner le dépôt TaskFlow depuis GitHub.

Vérifiez que Git est correctement installé avec la commande : `git --version`

1.2. Étapes d'installation

- **PowerShell**

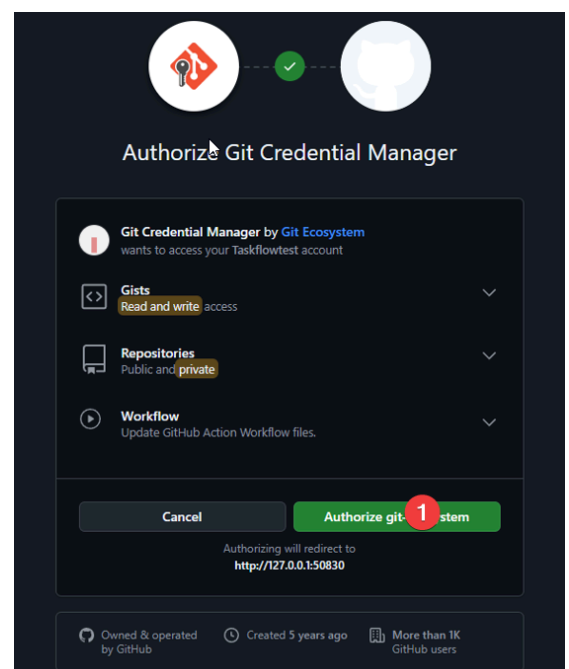
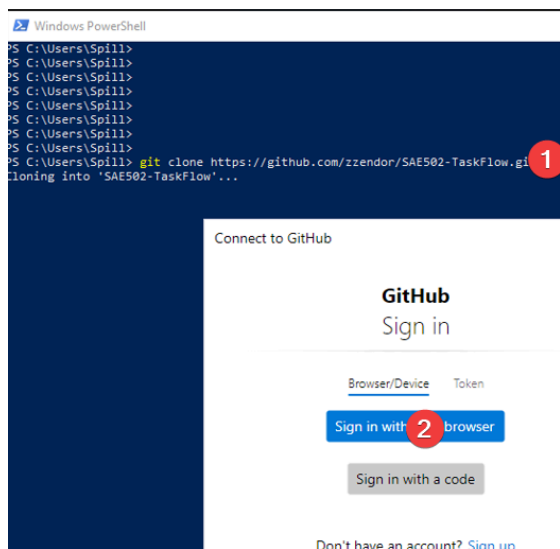
Utilisez un terminal pour exécuter les commandes nécessaires à l'installation.

- **Cloner le dépôt :**

Exécutez la commande suivante pour récupérer le projet depuis GitHub :

`git clone https://github.com/zzendor/SAE502-TaskFlow.git`

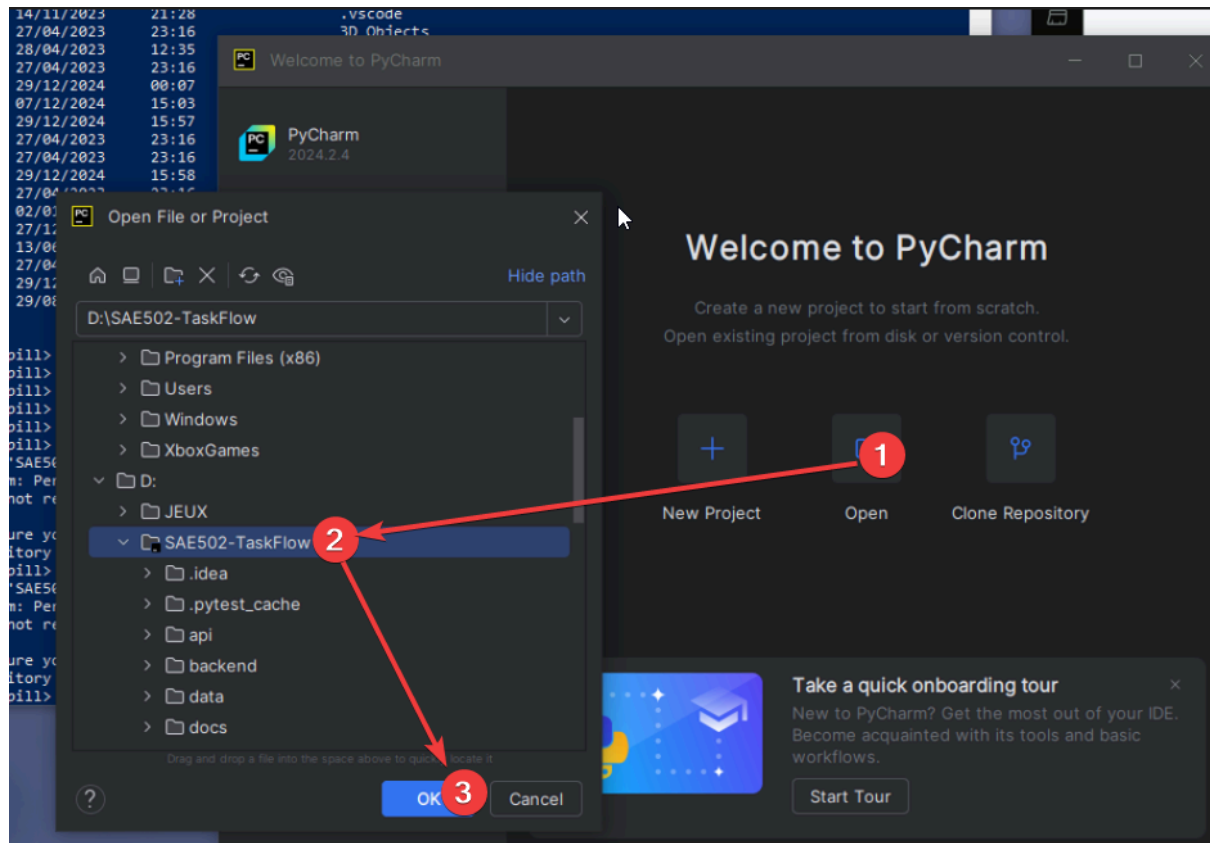
connecter vous avec vos ID github (image)
cd SAE502-TaskFlow



Connexion GitHub : Si demandé, connectez-vous avec vos identifiants GitHub (utilisateur et mot de passe). Vous pouvez également configurer une clé SSH pour éviter de saisir vos identifiants à chaque fois.

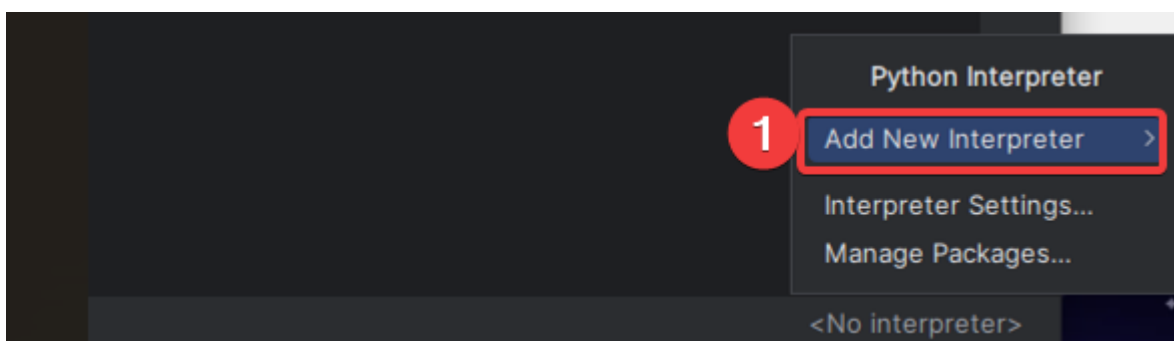
- **Ouvrir le projet dans PyCharm**

Lancez PyCharm et ouvrez le dossier **SAE502-TaskFlow** en allant dans **File > Open**.

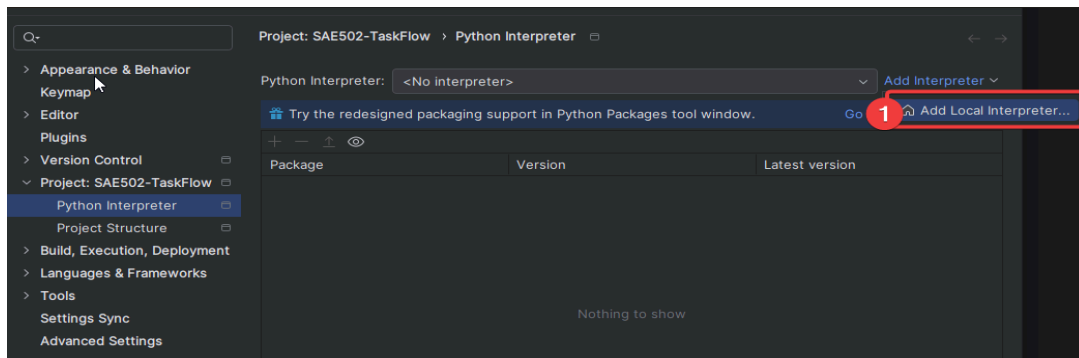


- **Configurer l'interpréteur Python dans PyCharm**

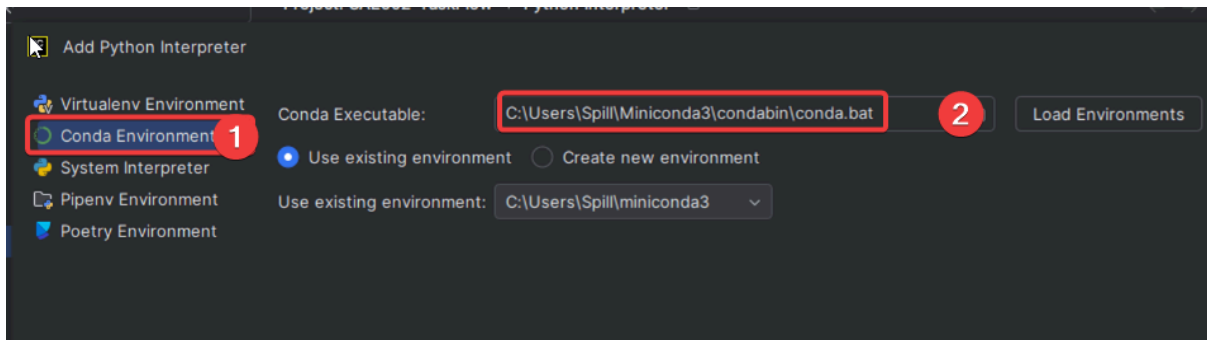
Allez dans **File > Settings > Project > Python Interpreter**.



Cliquez sur **Add Interpreter**.



Sélectionnez **Conda Environment**.



Choisissez l'environnement Conda installé précédemment ou créez-en un nouveau.

Attention : Vérifiez que le chemin de l'interpréteur pointe vers le bon environnement Conda.

- **Installer les dépendances Python supplémentaires**

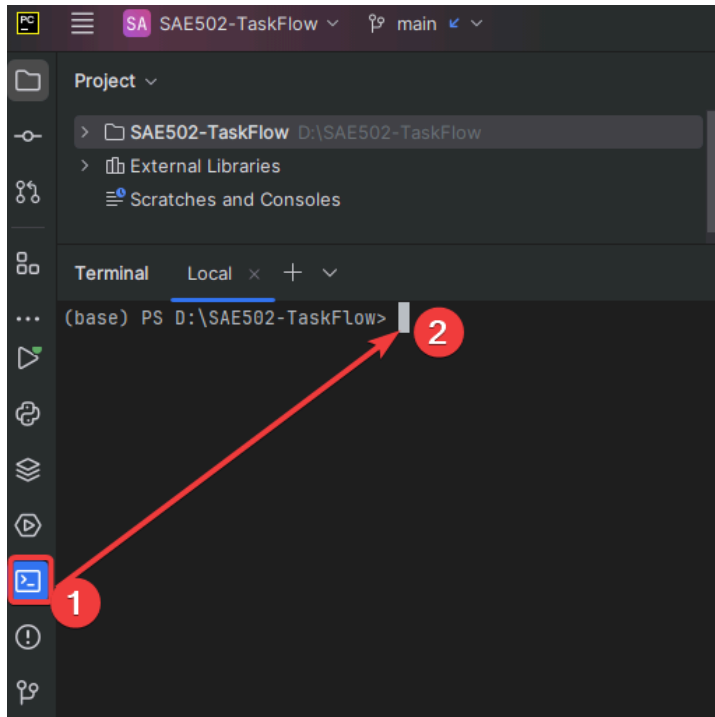
Dans le terminal intégré de PyCharm (ou dans votre terminal) :

```
pip install flask pyotp mysql-connector-python
pip install flask
pip install pyotp
pip install mysql-connector-python
pip install argon2-cffi
pip install zxcvbn
pip install PyQt5
pip install mysql-connector-python
pip install google-auth google-auth-oauthlib google-auth-httpplib2
pip install google-api-python-client
pip show google-auth
```

Pourquoi ? Ces dépendances sont nécessaires pour les fonctionnalités avancées de l'application, comme l'authentification ou la gestion de la base de données.

- **environnement virtuel Conda**

Dans le terminal intégré de PyCharm (ou dans votre terminal) :



entrée ces commandes :

```
conda env create -f conda.yml
```

```
conda env list
```

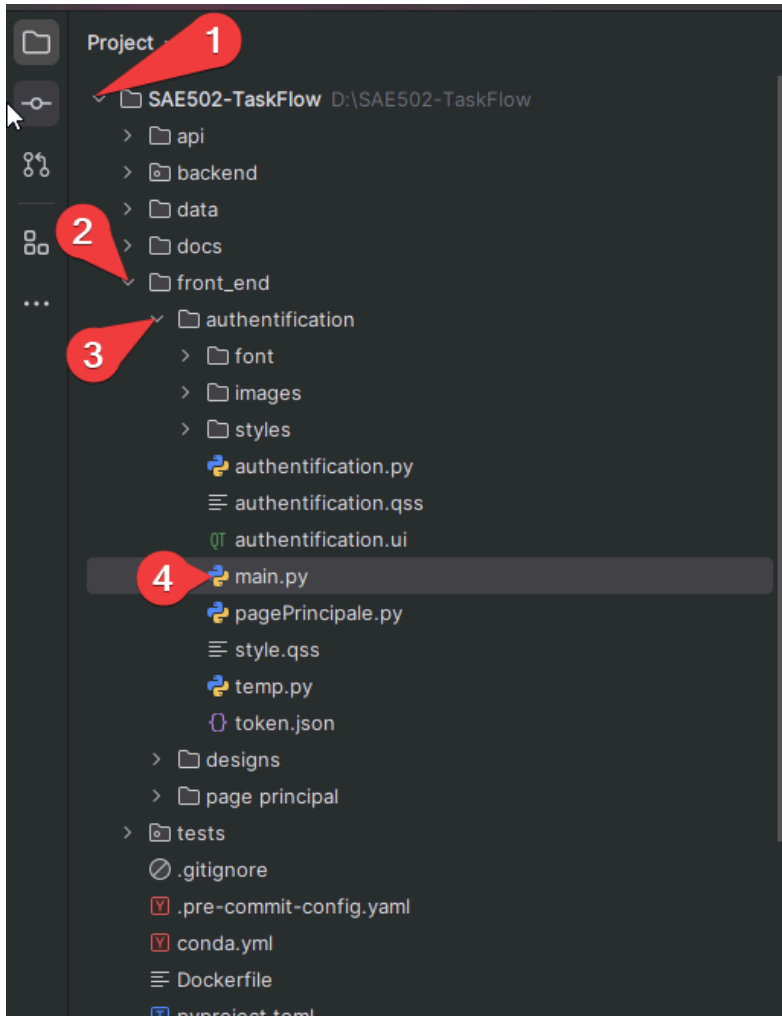
```
conda activate TaskFlow
```

```
conda env export -n TaskFlow | Out-File -Encoding UTF8 conda.yml
```

Pourquoi ? Ces commandes créent un environnement isolé pour le projet et installent automatiquement les dépendances définies dans le fichier conda.yml.

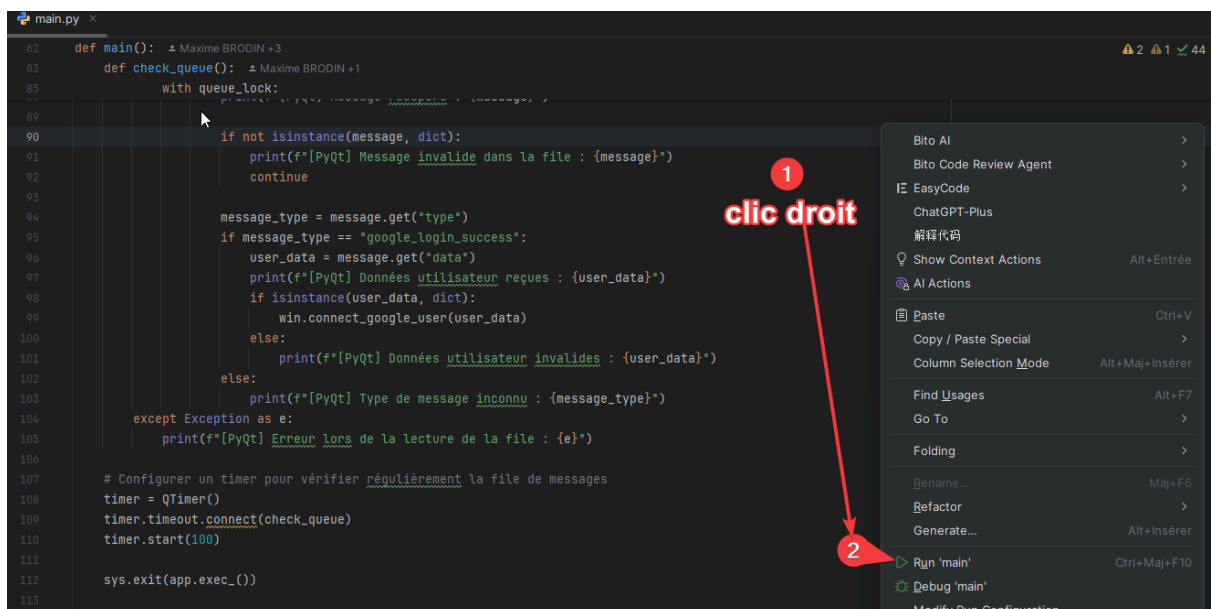
- **Ouvrir le projet dans PyCharm**

Lancez **PyCharm** après l'installation. Naviguez jusqu'au dossier où vous avez cloné le dépôt, généralement nommé **SAE502-TaskFlow**.



Sélectionnez le dossier et cliquez sur **OK** pour ouvrir le projet.

Une fois ouvert, attendez que PyCharm indexe les fichiers. Cela peut prendre un moment si c'est la première fois



2. Installation de TaskFlow par docker sur linux

1.1 Prérequis

Avant de commencer, assurez-vous que votre système linux est prêt avec :

- **Docker et Docker Compose installés :**

Docker est une plateforme de conteneurisation qui vous permet d'exécuter des applications dans des environnements isolés. Docker Compose facilite la gestion multi-conteneurs.

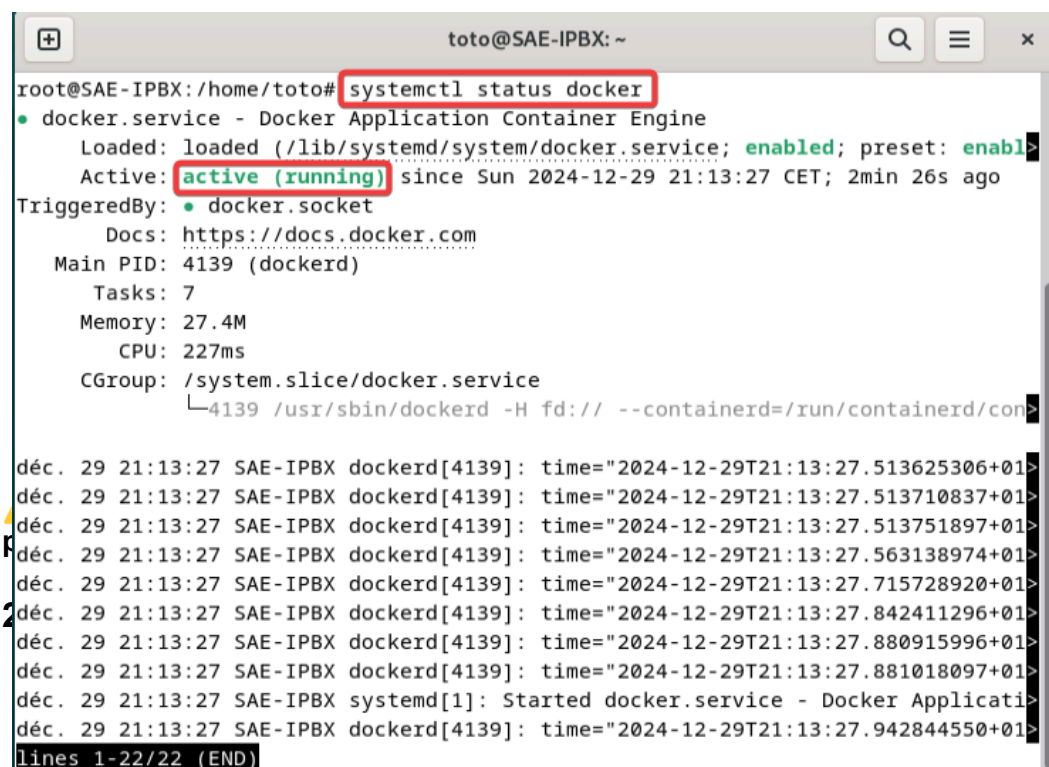
- <https://docs.docker.com/engine/install/debian/>

```
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Paramétrage de libintl-xs-perl (1.33-1) ...
Paramétrage de git (1:2.39.5-0+deb12u1) ...
Paramétrage de criu (3.17.1-2) ...
Traitement des actions différées (« triggers ») pour man-db (2.11.2-2) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.36-9+deb12u4) .
..
root@SAE-IPBX:/home/toto#
```

`sudo systemctl start docker`

`sudo systemctl enable docker`

`sudo usermod -aG docker $USER`



```
toto@SAE-IPBX: ~
root@SAE-IPBX:/home/toto# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; preset: enabled)
   Active: active (running) since Sun 2024-12-29 21:13:27 CET; 2min 26s ago
     TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
    Main PID: 4139 (dockerd)
      Tasks: 7
     Memory: 27.4M
        CPU: 227ms
    CGroup: /system.slice/docker.service
            └─4139 /usr/sbin/dockerd -H fd:// --containerd=/run/containerd/cont...

déc. 29 21:13:27 SAE-IPBX dockerd[4139]: time="2024-12-29T21:13:27.513625306+01>
déc. 29 21:13:27 SAE-IPBX dockerd[4139]: time="2024-12-29T21:13:27.513710837+01>
déc. 29 21:13:27 SAE-IPBX dockerd[4139]: time="2024-12-29T21:13:27.513751897+01>
déc. 29 21:13:27 SAE-IPBX dockerd[4139]: time="2024-12-29T21:13:27.563138974+01>
déc. 29 21:13:27 SAE-IPBX dockerd[4139]: time="2024-12-29T21:13:27.715728920+01>
déc. 29 21:13:27 SAE-IPBX dockerd[4139]: time="2024-12-29T21:13:27.842411296+01>
déc. 29 21:13:27 SAE-IPBX dockerd[4139]: time="2024-12-29T21:13:27.880915996+01>
déc. 29 21:13:27 SAE-IPBX dockerd[4139]: time="2024-12-29T21:13:27.881018097+01>
déc. 29 21:13:27 SAE-IPBX systemd[1]: Started docker.service - Docker Applicati>
déc. 29 21:13:27 SAE-IPBX dockerd[4139]: time="2024-12-29T21:13:27.942844550+01>
lines 1-22/22 (END)
```

ement

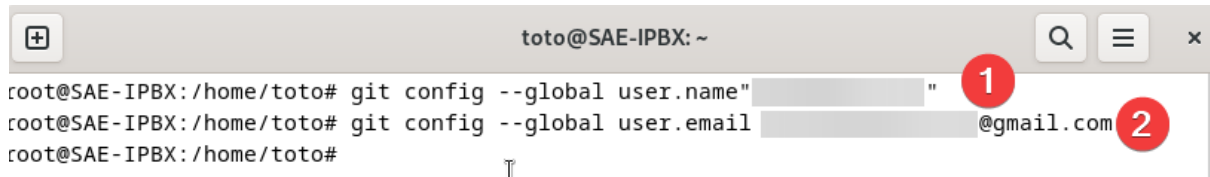
- **Cloner le dépôt**

Téléchargez le code source de l'application TaskFlow en clonant le dépôt Git.

Bash: `git clone https://github.com/zzendor/SAE502-TaskFlow.git`

`cd SAE502-TaskFlow`

Cette commande télécharge le code source depuis le dépôt GitHub et place le projet dans un dossier local.

A screenshot of a terminal window titled 'toto@SAE-IPBX: ~'. The terminal shows three lines of commands: 'git config --global user.name"', 'git config --global user.email @gmail.com', and a prompt. Red circles with numbers '1' and '2' highlight the input fields for the name and email respectively. The terminal window has a search icon, a menu icon, and a close icon in the top right corner.

```
toto@SAE-IPBX: ~  
root@SAE-IPBX:/home/toto# git config --global user.name"  
root@SAE-IPBX:/home/toto# git config --global user.email @gmail.com  
root@SAE-IPBX:/home/toto#
```

- **Exécution du script d'installation**

Lancer le script d'installation : Le script `installation.sh` configure automatiquement l'environnement Docker et crée un raccourci pour l'application TaskFlow.

`chmod +x installation.sh`

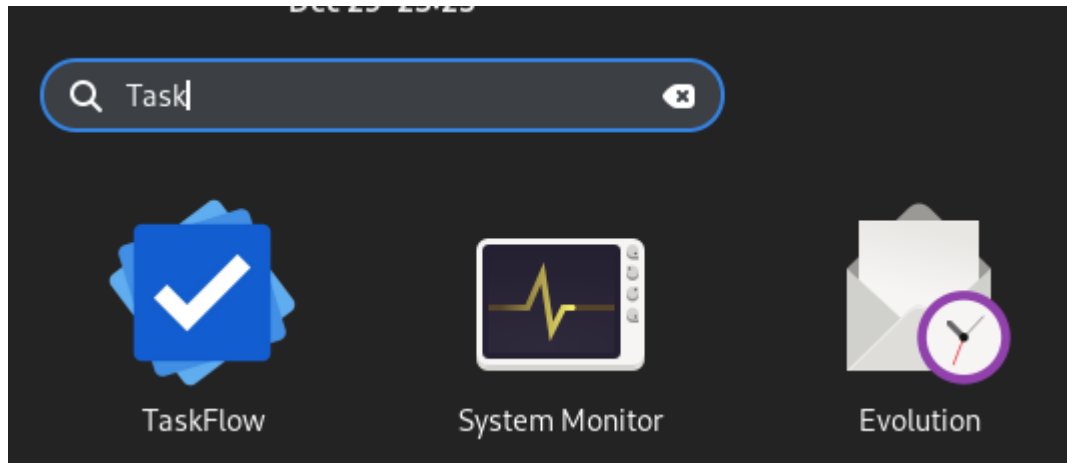
`./installation.sh`

Que fait le script d'installation :

- Il met à jour le système.
- Il vérifie et installe toutes les dépendances nécessaires.
- Il construit l'image Docker avec les dépendances spécifiques de l'application.
- Il crée un raccourci dans les Applications Linux pour exécuter TaskFlow.

- **Lancer l'application**

Depuis le raccourci : Une fois l'installation terminée, vous pouvez lancer l'application TaskFlow depuis le menu Applications. Recherchez "TaskFlow" dans la barre de recherche ou dans le menu.



Manuellement via Docker : Si vous souhaitez lancer l'application manuellement, utilisez les commandes suivantes :

```
xhost +local:
```

```
docker-compose up
```

Cette commande démarre l'application et configure l'affichage graphique.

- **Désinstaller le projet**

Si vous souhaitez désinstaller TaskFlow, utilisez le script `uninstall.sh`. Ce script supprimera les conteneurs Docker associés et le raccourci de l'application.

Lancer le script de désinstallation :

```
chmod +x uninstall.sh
```

```
./uninstall.sh
```

Que fait le script de désinstallation :

Il arrête et supprime les conteneurs Docker associés à TaskFlow.

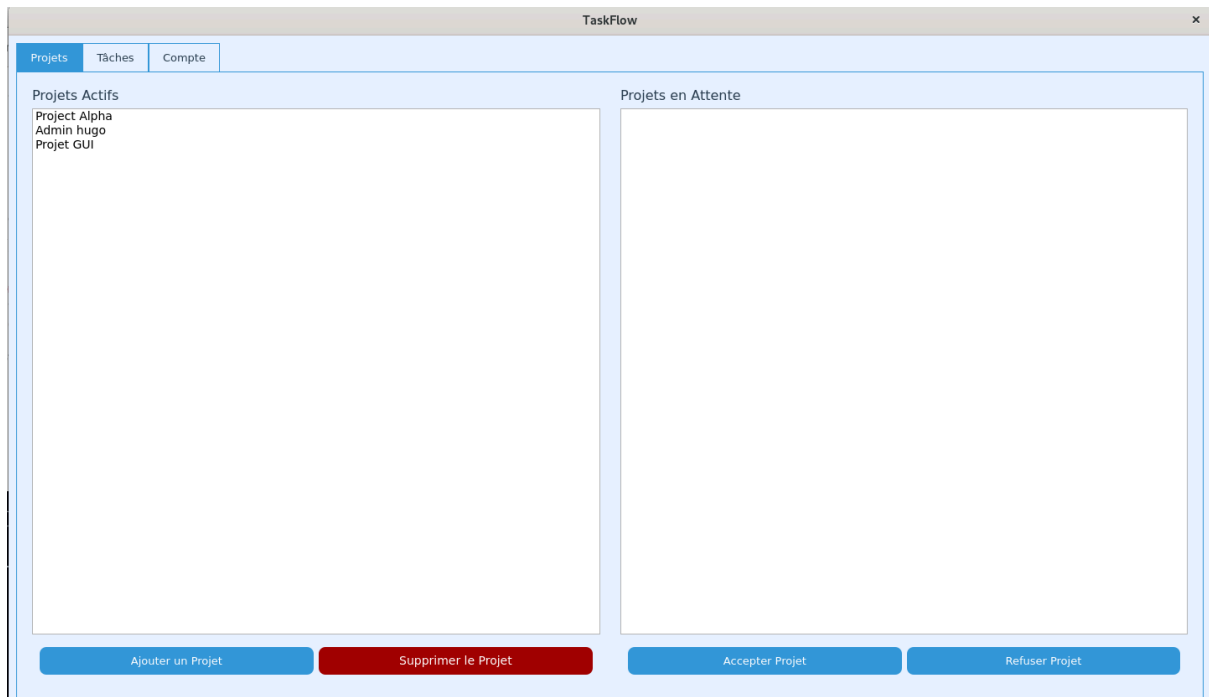
Il supprime le raccourci TaskFlow dans les Applications.

3 Points importants

X11 et affichage graphique :

Le script utilise `xhost` pour autoriser le conteneur Docker à utiliser l'affichage graphique de l'hôte. Si l'affichage ne fonctionne pas, assurez-vous que `xhost` est bien installé et que la commande suivante est exécutée avant de lancer l'application :

`xhost +local:`



Droits utilisateur :

Assurez-vous que votre utilisateur est dans le groupe Docker pour éviter des problèmes de permissions.

Dépannage :

Si l'application ne démarre pas ou plante, exécutez cette commande pour diagnostiquer le problème dans le conteneur Docker :

```
docker run -it --entrypoint=bash sae502-taskflow-main-taskflow-app
```

Cela vous permettra d'exécuter des commandes directement dans le conteneur pour vérifier les erreurs.