# Configuration Docker (avec redémarrage automatique) :

## 1) Installation

### **Installer WSL**

Ouvre **PowerShell (Admin)** et lance :   
 wsl –install  
 wsl --set-default-version 2

### **1.2 Installer Docker Engine dans Ubuntu (WSL)**

Dans la **distro Ubuntu** (terminal WSL) :

**1) Préparer le dépôt officiel Docker**

sudo apt-get update

sudo apt-get install -y ca-certificates curl gnupg lsb-release

sudo install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg \

| sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] \

https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable" \

| sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

**2) Installer**

sudo apt-get update

sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-compose-plugin

* Pour plus d’infos concernant ces commandes, se référer à la documentation officielle : [Docker\_Documenatation](https://docs.docker.com/engine/install/ubuntu/)
* **Remarque** : pour éviter de taper à chaque fois sudo pour utiliser les commandes de docker, exécutez: sudo usermod -aG docker $USER && exec su - $USER  
  Notez que le groupe docker donne des privilèges élevés. Activez-le seulement si besoin.

## 2) Configuration de redémarrage automatique

### **2.1 Schéma (flux global)**

**[Windows démarré]  
 │**

**│ Étape 1: Planificateur de tâches Windows** exécute wsl.exe

**│   
 ▼**

**[WSL2 Ubuntu démarré]  
 │**

**│ Étape 2: systemd activé** au démarrage de WSL etlance **│ docker.service**

**▼**

**[Docker Engine démarré]**

**│**

**│ Étape 3:** démarre tous les conteneurs gérés par la politique **│** **restart: unless-stopped** (ou **always**)

**▼**

**[Conteneurs démarrés]**

**Explication :**

Quand Windows démarre, la distribution WSL Ubuntu ne se lance pas automatiquement. Pour résoudre cela, on configure le **Planificateur de tâches Windows** afin qu’il exécute wsl.exe au démarrage, ce qui “réveille” la distro WSL (**=Étape 1**).

Une fois Ubuntu démarré, **systemd** (activé via /etc/wsl.conf) s’exécute comme gestionnaire de services et lance automatiquement le service docker.service (=**Étape 2**).

Ce service démarre le moteur **Docker Engine**, qui restaure ensuite tous les conteneurs configurés avec la politique de redémarrage restart: unless-stopped (ou always) (=**Étape 3**). Ces conteneurs sont alors relancés à chaque démarrage, sauf s’ils ont été arrêtés volontairement par l’utilisateur.

### **Implémentation**

#### **Étape 1 : Planificateur de tâches Windows exécute WSL**

Par défaut, WSL démarre uniquement “à la demande” et pas automatiquement au redémarrage. Pour résoudre ça, on implémente une **tâche planifiée** qui exécute wsl.exe au démarrage de Windows pour initialiser la distro Ubuntu immédiatement.

Pour ce faire :

1. Ouvre **Planificateur de tâches** → Créer une tâche.
2. **Onglet Général**
   * Entrer le nom de la tâche (exemple: Docker AutoStart)
   * Cocher*:* « Exécuter avec les autorisations maximales »
   * Cocher l’une des deux options:  
     - «Exécuter uniquement si l’utilisateur est connecté»   
     - «Exécuter que l’utilisateur soit connecté ou non» (dans ce cas, entrer le mot de passe qui sera demandé à la fin)
3. **Onglet Déclencheurs**
   * Clique sur Nouveau, puis choisis: Au démarrage ou à l’ouverture de session (si connexion d’utilisateur requis)*.*
   * (Optionnel) coche : Retarder la tâche de **30 secondes (ou 1min)**→ ça laisse le temps au réseau et aux services de se charger, plus le temps de connexion au compte si connexion requis.
   * Valide.
4. **Onglet Actions**
   * Clique sur Nouveau, puis choisis : Action: Démarrer un programme.
   * Programme/script : wsl.exe
   * Arguments : -d Ubuntu-24.04 -u root --exec /bin/true  
     (Si votre distro a un autre nom que “Ubuntu-24.04”, remplacez celui-ci par ce nom. Vérifiez avec *wsl -l -q*.)
5. **Onglet Paramètres**
   * Sélectionne: Si la tâche est déjà en cours, ne pas démarrer une nouvelle instance.
6. **Valide**, puis teste : clic droit → Exécuter.

#### **Étape 2 : systemd activé au démarrage de WSL et active docker.service**

**Systemd**, le gestionnaire des services, doit être activé pour que le service docker.service soit démarré automatiquement au lancement d’Ubuntu dans WSL. Sans systemd, pas d’auto-start propre des services.

Pour ce faire, il faut d’abord activer systemd au démarrage de Ubuntu (WSL) :  
 - Ouvre le fichier de config : sudo nano /etc/wsl.conf  
 - Puis y colle : [boot]  
 systemd=true  
 Vérification :  
 - Exécute dans Powershell : wsl –shutdown  
 - Relance Ubuntu, et vérifie : systemctl is-system-running  
 Si tout marche bien, vous allez voir : 'running' ou 'degraded'

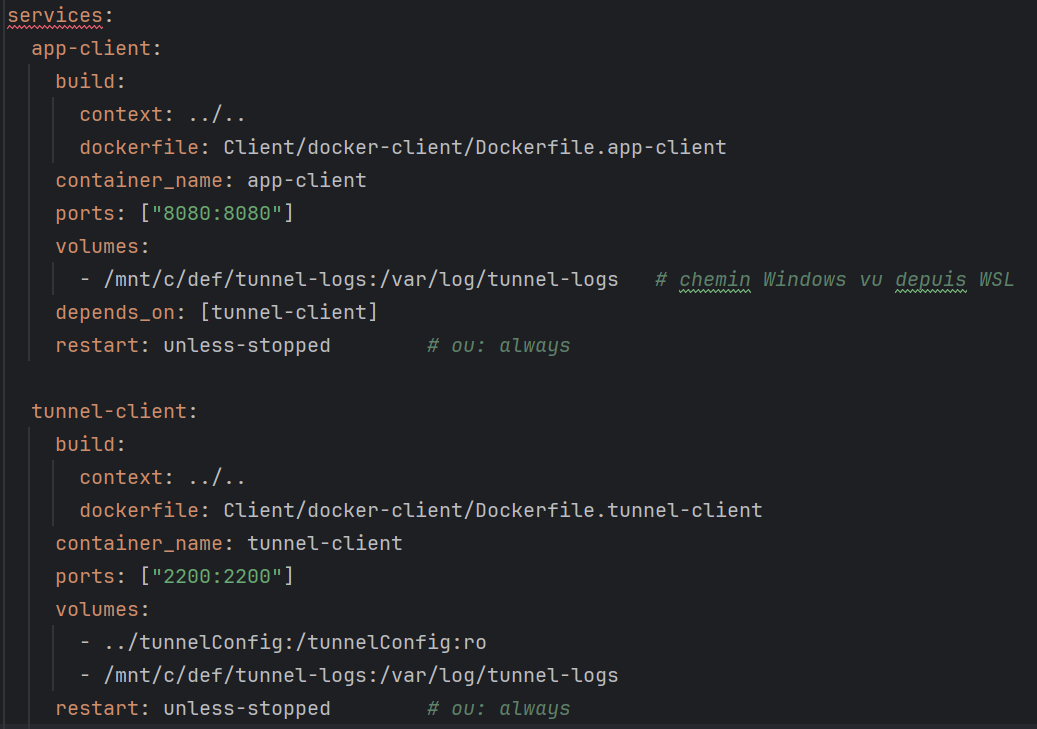
Ensuite, il faut activer le service docker.service pour qu'il soit lancé automatiquement par systemd déjà démarré, et lance ainsi Docker Engine (dockered).  
Pour ce faire, exécute dans **Ubuntu (WSL)** : sudo systemctl enable --now docker

* Vérification : (l’état de docker)  
  systemctl is-active docker **# doit afficher 'active'**

#### **Étape 3 : Relance automatique des conteneurs (restart policy)**

Au démarrage de Docker, **tous les conteneurs existants** portant une restart policy adaptée sont relancés automatiquement :

* restart: always : relance **toujours** (même si stoppé manuellement avant reboot).
* restart: unless-stopped : relance sauf si **l’utilisateur** les as stoppés (par   
  docker stop) avant reboot.

**Pour ce faire, il faut ajouter cette restart policy dans les fichiers docker compose, voici alors un exemple : compose.client.yaml**

**Remarque** : Si vous exécutez docker compose down qui **supprime** les conteneurs (au lieu d’uniquement les stopper par docker stop), il n’y a **plus rien à relancer** car la restart policy ne recrée pas le stack supprimé.

### **Test “réel” après reboot**

1. Redémarre Windows.
2. Dans PowerShell, vérifie :  
    wsl -l -v  
    **-> doit afficher: Running (donc WSL est bien lancé)**  
    wsl -d Ubuntu-24.04 -- systemctl is-active docker  
    **-> doit afficher: active**   
     
    wsl -d Ubuntu-24.04 -- docker ps  
    **-> doit afficher tous les conteneurs actifs «up»**

**Remarque :**  
Si besoin , remplacer dans ces commandes le mot Ubuntu-24.04 par le nom de votre distro WSL actuelle, que vous pouvez obtenir en éxécutant dans PowerShell :  
wsl -l -q

**Optionnel :**

* **Ajouter Healthchecks dans docker compose:**  
  depends\_on uniquement ne garantit pas l’état “prêt”, il faut donc ajoute un healthcheck pour conditionner l’état healthy du service :

tunnel-client:

# ...

healthcheck:

test: ["CMD-SHELL", "bash -c '</dev/tcp/127.0.0.1/2200'"]

interval: 5s

timeout: 2s

retries: 12

start\_period: 10s

app-client:

# ...

depends\_on:

tunnel-client:

condition: service\_healthy

* **Recréation du stack si supprimé** :  
  si tu veux que le stack se **recrée** au boot même après un docker compose down/ prune, il faut ajouter un service systemd “Compose” au lieu du service docker.service

# Dans Ubuntu (root)

sudo tee /etc/systemd/system/tunnel-stack.service >/dev/null <<'EOF'

[Unit]

Description=Tunnel stack (docker compose)

After=docker.service network-online.target

Requires=docker.service

[Service]

Type=oneshot

WorkingDirectory=/home/<user>/projet

ExecStart=/usr/bin/docker compose up -d

ExecStop=/usr/bin/docker compose down

RemainAfterExit=yes

[Install]

WantedBy=multi-user.target

EOF

sudo systemctl daemon-reload

sudo systemctl enable --now tunnel-stack.service

À utiliser uniquement si vous voulez lancer les conteneurs d’une façon permanente, meme si reboot après avoir stoppé volontairement, ce qui ne respecte pas la policy unless-stopped.

* **Chemins de volumes Windows sous WSL :**Utilisez les chemins montés /mnt/<lettre>/… (ex. C: → /mnt/c/…) pour accéder aux chemins Windows. Exemple :  
   volumes:  
   - /mnt/c/def/tunnel-logs:/var/log/tunnel-logs

## Annexes :

### **- Annexe A : Option Linux**

#### Schéma

**[Linux (systemd activé par défaut)]**

**│**

**│ Étape 1:** systemd démarre docker.service (au boot)

**│**

**▼**

**[Docker Engine démarré]**

**│**

**│ Étape 2:** relance tous les conteneurs (restart policy)

**│  
 ▼**

**[Conteneurs démarrés]**

Dans Linux, systemd est activé par défaut, il suffit donc d’activer **docker.service** au démarrage via **systemd**. Au lancement du démon, Docker applique les **restart policies** (always / unless-stopped) pour relancer les conteneurs existants.

#### Implémentation (pas à pas)

1. **Installer Docker Engine (ex. Ubuntu) :** la meme commande utilisé pour installer docker dans le cas de Windows avec WSL:

sudo apt-get update

sudo apt-get install -y ca-certificates curl gnupg lsb-release

curl -fsSL https://download.docker.com/linux/ubuntu/gpg \

| sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] \

https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable" \

| sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-compose-plugin

1. **Activer docker.service** :

sudo systemctl enable --now docker

systemctl is-active docker # 'active' attendu

1. **Restart policy** **dans les docker-compose.yaml** :

services:

my-service:

image: ...

restart: unless-stopped # ou: always

always redémarre toujours (y compris après reboot du démon) ; unless-stopped ne relance pas si tu as arrêté manuellement avant.

### **- Annexe B: Option Windows avec Docker Desktop**

#### Schéma :

**[Windows démarré]  
 │**

**│ Étape 1: Planificateur de tâches Windows** au démarrage

│ exécute: docker desktop start   
 **▼**

**[Docker Desktop]**

**│**

**│** **Étape 2:** démarre le deamon relançant les conteneurs définis par

**│ la restart policy  
 ▼**

**[Conteneurs démarrés]**

Docker Desktop n'est pas un vrai service Windows, mais il est possible de contourner ce problème en utilisant le **Planificateur de tâches** de Windows pour automatiser son lancement, tout comme dans le cas de Windows avec WSL.

Ainsi, une fois le daemon lancé, Docker applique les **restart policies** (always/unless-stopped) et **relance** tes conteneurs existants.

#### Implémentation (pas à pas) :

1. **Planificateur de tâches** **pour lancer Docker Desktop :**
2. Ouvre **Planificateur de tâches** → Créer une tâche…
3. **Onglet Général**
   * Entrer le nom de la tâche (exemple: Docker Desktop AutoStart)
   * Cocher*:* « Exécuter avec les autorisations maximales »
   * Cocher l’une des deux options:  
     - «Exécuter uniquement si l’utilisateur est connecté»   
     - «Exécuter que l’utilisateur soit connecté ou non» (dans ce cas, entrer le mot de passe qui sera demandé à la fin)
4. **Onglet Déclencheurs**
   * Clique sur Nouveau, puis choisis: Au démarrage ou à l’ouverture de session (si connexion d’utilisateur requis)*.*
   * (Optionnel) coche : Retarder la tâche de **30 secondes (ou 1min)**→ ça laisse le temps au réseau et aux services Windows de se charger, plus le temps de connexion au compte si connexion requis.
   * Valide.
5. **Onglet Actions**
   * Clique sur Nouveau, puis choisis Action: Démarrer un programme.
   * Dans Programme/script, mets :   
     "C:\Program Files\Docker\Docker\Docker Desktop.exe"  
     (Ou bien "C:\Program Files\Docker\Docker\resources\bin\ docker.exe", avec argument : desktop start)

1. **Onglet Paramètres**
   * Sélectionne: Si la tâche est déjà en cours, ne pas démarrer une nouvelle instance.
2. **Valide**. Pour tester : clic droit → Exécuter.
3. **Restart policy dans les** docker-compose.yaml**:**

services:

my-service:

image: ...

restart: unless-stopped # ou: always

#### Vérification après redémarrage :

1. Redémarre Windows.
2. Dans PowerShell:  
    docker desktop status # "Running" attendu  
    docker ps # les conteneurs actifs