# 簡単なインストールガイド

CVATを使用する前に、インストールする必要があります。以下のドキュメントには、最も一般的なオペレーティングシステム向けの手順が含まれています。お使いのシステムがこのドキュメントでカバーされていない場合でも、以下の手順を他のシステムに適用するのは比較的簡単なはずです。

プロキシサーバーを経由している場合は、以下の手順を修正する必要があるかもしれません。プロキシは高度なトピックであり、このガイドでは扱っていません。

中国からのアクセスについては、[中国ユーザー向けの情報源](#sources-for-users-from-china)セクションをお読みください。

## Ubuntu 22.04/20.04 (x86\_64/amd64)

* 端末ウィンドウを開きます。Ubuntuで端末ウィンドウを開く方法がわからない場合は、[こちらの回答](https://askubuntu.com/questions/183775/how-do-i-open-a-terminal)をお読みください。
* 以下のコマンドを端末ウィンドウに入力して、DockerとDocker Composeをインストールします。詳細な手順は[こちら](https://docs.docker.com/install/linux/docker-ce/ubuntu/)で確認できます。
* sudo apt-get update  
  sudo apt-get --no-install-recommends install -y \  
   apt-transport-https \  
   ca-certificates \  
   curl \  
   gnupg-agent \  
   software-properties-common  
  curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -  
  sudo add-apt-repository \  
   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \  
   $(lsb\_release -cs) \  
   stable"  
  sudo apt-get update  
  sudo apt-get --no-install-recommends install -y \  
   docker-ce docker-ce-cli containerd.io docker-compose-plugin
* (オプション) Dockerコマンドの前にsudoを付けるのを避けるには、[インストール後の手順](https://docs.docker.com/install/linux/linux-postinstall/)を実行できます。これには、dockerという名前のUnixグループを作成し、現在のユーザーをこのグループに追加することが含まれます。
* sudo groupadd docker  
  sudo usermod -aG docker $USER
* ログアウトして再度ログイン（または再起動）し、グループメンバーシップが再評価されるようにします。その後、端末ウィンドウでgroupsコマンドを入力し、出力にdockerグループが含まれているか確認できます。
* Gitを使用して[GitHubリポジトリ](https://github.com/cvat-ai/cvat)から\_CVAT\_のソースコードをクローンします。
* 以下のコマンドは最新のdevelopブランチをクローンします：
* git clone https://github.com/cvat-ai/cvat  
  cd cvat
* リリースバージョンの一つをダウンロードしたい場合や、wgetまたはcurlツールを使用したい場合は、[代替手段](#how-to-get-cvat-source-code)を参照してください。
* ネットワーク経由または異なるシステムからCVATにアクセスするには、CVAT\_HOST環境変数をエクスポートします。
* export CVAT\_HOST=FQDNまたはあなたのIPアドレス
* Dockerコンテナを実行します。最新のCVATおよびその他の必要なイメージ（postgres、redisなど）をダウンロードし、コンテナを開始するには時間がかかります。
* docker compose up -d
* (オプション) CVAT\_VERSION環境変数を使用して、インストールしたいCVATの特定のバージョン（例：v2.1.0、dev）を指定します。デフォルトの動作：developブランチにはdevイメージが、リリースバージョンには対応するリリースイメージがプルされます。
* CVAT\_VERSION=dev docker compose up -d
* 代替：未リリースの変更を含むイメージをローカルでビルドしたい場合は、[CVATイメージのプル/ビルド/更新方法](#how-to-pullbuildupdate-cvat-images)セクションを参照してください。
* ユーザーを登録できますが、デフォルトではタスクのリストを表示する権限すらありません。したがって、スーパーユーザーを作成する必要があります。スーパーユーザーは管理パネルを使用して、ユーザーに適切なグループを割り当てることができます。以下のコマンドを使用してください：
* docker exec -it cvat\_server bash -ic 'python3 ~/manage.py createsuperuser'
* 管理者アカウントのユーザー名とパスワードを選択してください。詳細については、[Djangoドキュメント](https://docs.djangoproject.com/en/2.2/ref/django-admin/#createsuperuser)をお読みください。
* Google ChromeはCVATがサポートする唯一のブラウザです。これもインストールする必要があります。以下のコマンドを端末ウィンドウに入力してください：
* curl https://dl-ssl.google.com/linux/linux\_signing\_key.pub | sudo apt-key add -  
  sudo sh -c 'echo "deb [arch=amd64] http://dl.google.com/linux/chrome/deb/ stable main" >> /etc/apt/sources.list.d/google-chrome.list'  
  sudo apt-get update  
  sudo apt-get --no-install-recommends install -y google-chrome-stable
* インストールされたGoogle Chromeブラウザを開き、[localhost:8080](http://localhost:8080)にアクセスします。ログインページでスーパーユーザーのログイン情報/パスワードを入力し、\_Login\_ボタンを押します。これで新しいアノテーションタスクを作成できるようになります。詳細については、{{< ilink “/docs/manual” “CVATマニュアル” >}}をお読みください。

## Windows 10

* [こちらの公式ガイド](https://docs.microsoft.com/windows/wsl/install-win10)を参照してWSL2（Linux用Windowsサブシステム）をインストールします。WSL2にはWindows 10、バージョン2004以降が必要です。WSL2をインストールした後、選択したLinuxディストリビューションをインストールします。
* [Docker Desktop for Windows](https://desktop.docker.com/win/main/amd64/Docker%20Desktop%20Installer.exe)をダウンロードしてインストールします。Docker for Windows Installerをダブルクリックしてインストーラーを実行します。詳細な手順は[こちら](https://docs.docker.com/docker-for-windows/install/)で確認できます。Docker WSL2バックエンドの公式ガイドは[こちら](https://docs.docker.com/docker-for-windows/wsl/)で確認できます。注意：Dockerには具体的にWSL2バックエンドを使用していることを確認してください。
* Docker Desktopで、Settings >> Resources >> WSL Integrationに移動し、選択したLinuxディストリビューションとの統合を有効にします。
* [Git for Windows](https://github.com/git-for-windows/git/releases/download/v2.21.0.windows.1/Git-2.21.0-64-bit.exe)をダウンロードしてインストールします。パッケージをインストールする際は、すべてのオプションをデフォルトのままにしてください。パッケージに関する詳細情報は[こちら](https://gitforwindows.org)で確認できます。
* [Google Chrome](https://www.google.com/chrome/)をダウンロードしてインストールします。これはCVATがサポートする唯一のブラウザです。
* Windowsメニューに移動し、インストールしたLinuxディストリビューションを見つけて実行します。端末ウィンドウが表示されるはずです。
* [GitHubリポジトリ](https://github.com/cvat-ai/cvat)から\_CVAT\_のソースコードをクローンします。
* 以下のコマンドは最新のdevelopブランチをクローンします：
* git clone https://github.com/cvat-ai/cvat  
  cd cvat
* リリースバージョンの一つをダウンロードしたい場合は、[代替手段](#how-to-get-cvat-source-code)を参照してください。
* Dockerコンテナを実行します。最新のCVATリリースおよびその他の必要なイメージ（postgres、redisなど）をDockerHubからダウンロードし、コンテナを作成するには時間がかかります。
* docker compose up -d
* (オプション) CVAT\_VERSION環境変数を使用して、インストールしたいCVATの特定のバージョン（例：v2.1.0、dev）を指定します。デフォルトの動作：developブランチにはdevイメージが、リリースバージョンには対応するリリースイメージがプルされます。
* CVAT\_VERSION=dev docker compose up -d
* 代替：未リリースの変更を含むイメージをローカルでビルドしたい場合は、[CVATイメージのプル/ビルド/更新方法](#how-to-pullbuildupdate-cvat-images)セクションを参照してください。
* ユーザーを登録できますが、デフォルトではタスクのリストを表示する権限すらありません。したがって、スーパーユーザーを作成する必要があります。スーパーユーザーは管理パネルを使用して、他のユーザーに適切なグループを割り当てることができます。以下のコマンドを使用してください：
* sudo docker exec -it cvat\_server bash -ic 'python3 ~/manage.py createsuperuser'
* winptyがインストールされていない場合や上記のコマンドが機能しない場合は、以下を試すこともできます：
* # 最初にdockerイメージに入る  
  docker exec -it cvat\_server /bin/bash  
  # 次に実行  
  python3 ~/manage.py createsuperuser
* 管理者アカウントのユーザー名とパスワードを選択してください。詳細については、[Djangoドキュメント](https://docs.djangoproject.com/en/2.2/ref/django-admin/#createsuperuser)をお読みください。
* インストールされたGoogle Chromeブラウザを開き、[localhost:8080](http://localhost:8080)にアクセスします。ログインページでスーパーユーザーのログイン情報/パスワードを入力し、\_Login\_ボタンを押します。これで新しいアノテーションタスクを作成できるようになります。詳細については、{{< ilink “/docs/manual” “CVATマニュアル” >}}をお読みください。

## Mac OS Mojave

* [Docker for Mac](https://download.docker.com/mac/stable/Docker.dmg)をダウンロードします。Docker.dmgをダブルクリックしてインストーラーを開き、Moby the whale（クジラのアイコン）をApplicationsフォルダにドラッグします。Applicationsフォルダ内のDocker.appをダブルクリックしてDockerを起動します。詳細な手順は[こちら](https://docs.docker.com/v17.12/docker-for-mac/install/#install-and-run-docker-for-mac)で確認できます。
* MacにGitをインストールする方法はいくつかあります。最も簡単なのは、おそらくXcode Command Line Toolsをインストールすることです。Mavericks (10.9)以降では、初めてターミナルからgitを実行しようとすると、これを簡単に行うことができます。
* git --version
* まだインストールされていない場合は、インストールを促すプロンプトが表示されます。詳細な手順は[こちら](https://git-scm.com/book/en/v2/Getting-Started-Installing-Git)で確認できます。
* [Google Chrome](https://www.google.com/chrome/)をダウンロードしてインストールします。これはCVATがサポートする唯一のブラウザです。
* 端末ウィンドウを開きます。ターミナルアプリはApplicationsフォルダ内のUtilitiesフォルダにあります。開くには、Applicationsフォルダを開き、Utilitiesを開いてターミナルをダブルクリックするか、Command - スペースバーを押してSpotlightを起動し、「Terminal」と入力して検索結果をダブルクリックします。
* Gitを使用して[GitHubリポジトリ](https://github.com/cvat-ai/cvat)から\_CVAT\_のソースコードをクローンします。
* 以下のコマンドは最新のdevelopブランチをクローンします：
* git clone https://github.com/cvat-ai/cvat  
  cd cvat
* リリースバージョンの一つをダウンロードしたい場合や、wgetまたはcurlツールを使用したい場合は、[代替手段](#how-to-get-cvat-source-code)を参照してください。
* Dockerコンテナを実行します。最新のCVATリリースおよびその他の必要なイメージ（postgres、redisなど）をDockerHubからダウンロードし、コンテナを作成するには時間がかかります。
* docker compose up -d
* (オプション) CVAT\_VERSION環境変数を使用して、インストールしたいCVATの特定のバージョン（例：v2.1.0、dev）を指定します。デフォルトの動作：developブランチにはdevイメージが、リリースバージョンには対応するリリースイメージがプルされます。
* CVAT\_VERSION=dev docker compose up -d
* 代替：未リリースの変更を含むイメージをローカルでビルドしたい場合は、[CVATイメージのプル/ビルド/更新方法](#how-to-pullbuildupdate-cvat-images)セクションを参照してください。
* ユーザーを登録できますが、デフォルトではタスクのリストを表示する権限すらありません。したがって、スーパーユーザーを作成する必要があります。スーパーユーザーは管理パネルを使用して、他のユーザーに適切なグループを割り当てることができます。以下のコマンドを使用してください：
* docker exec -it cvat\_server bash -ic 'python3 ~/manage.py createsuperuser'
* 管理者アカウントのユーザー名とパスワードを選択してください。詳細については、[Djangoドキュメント](https://docs.djangoproject.com/en/2.2/ref/django-admin/#createsuperuser)をお読みください。
* インストールされたGoogle Chromeブラウザを開き、[localhost:8080](http://localhost:8080)にアクセスします。ログインページでスーパーユーザーのログイン情報/パスワードを入力し、\_Login\_ボタンを押します。これで新しいアノテーションタスクを作成できるようになります。詳細については、{{< ilink “/docs/manual” “CVATマニュアル” >}}をお読みください。

## 高度なトピック

### CVATソースコードの取得方法

#### Git (Linux, Mac, Windows)

1. システムにGitがまだインストールされていない場合はインストールします。
   * Ubuntu:

* sudo apt-get --no-install-recommends install -y git
  + Windows: <https://git-scm.com/download/win>の手順に従います。

1. [GitHubリポジトリ](https://github.com/cvat-ai/cvat)から\_CVAT\_のソースコードをクローンします。

* 以下のコマンドはデフォルトブランチ（develop）をクローンします：
* git clone https://github.com/cvat-ai/cvat  
  cd cvat
* 特定のタグ、例えばv2.1.0をクローンするには：
* git clone -b v2.1.0 https://github.com/cvat-ai/cvat  
  cd cvat

#### Wget (Linux, Mac)

最新のdevelopブランチをダウンロードするには：

wget https://github.com/cvat-ai/cvat/archive/refs/heads/develop.zip  
unzip develop.zip && mv cvat-develop cvat  
cd cvat

特定のタグをダウンロードするには：

wget https://github.com/cvat-ai/cvat/archive/refs/tags/v1.7.0.zip  
unzip v1.7.0.zip && mv cvat-1.7.0 cvat  
cd cvat

#### Curl (Linux, Mac)

最新のdevelopブランチをダウンロードするには：

curl -LO https://github.com/cvat-ai/cvat/archive/refs/heads/develop.zip  
unzip develop.zip && mv cvat-develop cvat  
cd cvat

特定のタグをダウンロードするには：

curl -LO https://github.com/cvat-ai/cvat/archive/refs/tags/v1.7.0.zip  
unzip v1.7.0.zip && mv cvat-1.7.0 cvat  
cd cvat

### CVATヘルスチェックコマンド

以下のコマンドを使用すると、CVATコンテナが正常に動作しているかテストできます。

docker exec -t cvat\_server python manage.py health\_check

正常なCVATコンテナの期待される出力：

Cache backend: default ... working  
DatabaseBackend ... working  
DiskUsage ... working  
MemoryUsage ... working  
MigrationsHealthCheck ... working  
OPAHealthCheck ... working

### プロキシの背後でCVATをデプロイする

プロキシの背後でCVATをデプロイし、自動アノテーションのための[サーバーレス関数](#semi-automatic-and-automatic-annotation)を使用しない場合は、エクスポートされた環境変数http\_proxy、https\_proxy、no\_proxyでイメージをビルドするのに十分です。そうでない場合は、コンテナを起動するユーザーのホームディレクトリにある~/.docker/config.jsonファイルを作成または編集し、次のようなJSONを追加してください：

{  
 "proxies": {  
 "default": {  
 "httpProxy": "http://proxy\_server:port",  
 "httpsProxy": "http://proxy\_server:port",  
 "noProxy": "\*.test.example.com,.example2.com"  
 }  
 }  
}

これらの環境変数は、任意のコンテナ内で自動的に設定されます。詳細については、[Dockerドキュメント](https://docs.docker.com/network/proxy/)を参照してください。

### Traefikダッシュボードの使用

docker-composeファイルをカスタマイズしていて予期しない問題が発生した場合、Traefikダッシュボードを使用すると、問題がTraefikの設定にあるのか、それとも他のサービスにあるのかを確認するのに非常に役立ちます。

docker-compose.ymlから以下の行のコメントを解除することで、Traefikダッシュボードを有効にできます。

services:  
 traefik:  
 # Traefikダッシュボードを取得するにはコメントを解除  
 # - "--entryPoints.dashboard.address=:8090"  
 # - "--api.dashboard=true"  
 # labels:  
 # - traefik.enable=true  
 # - traefik.http.routers.dashboard.entrypoints=dashboard  
 # - traefik.http.routers.dashboard.service=api@internal  
 # - traefik.http.routers.dashboard.rule=Host(`${CVAT\_HOST:-localhost}`)

そして、docker-compose.https.ymlを使用している場合は、これらの行もコメント解除します。

services:  
 traefik:  
 command:  
 # Traefikダッシュボードを取得するにはコメントを解除  
 # - "--entryPoints.dashboard.address=:8090"  
 # - "--api.dashboard=true"

注意：この「安全でない」ダッシュボードは、本番環境（およびインスタンスが公開されている場合）では推奨されません。本番環境でダッシュボードを保持したい場合は、Traefikの[ドキュメント](https://doc.traefik.io/traefik/operations/dashboard/)を読んで適切に保護する方法を確認してください。

### 追加コンポーネント

#### 半自動および自動アノテーション

{{< ilink “/docs/administration/advanced/installation\_automatic\_annotation” “このガイド” >}}に従ってください。

### すべてのコンテナを停止する

以下のコマンドは、upによって作成されたコンテナとネットワークを停止し、削除します。

docker compose down

### 独自のドメインを使用する

localhost以外（別のドメイン）からCVATインスタンスにアクセスしたい場合は、CVAT\_HOST環境変数を次のように指定する必要があります：

export CVAT\_HOST=<YOUR\_DOMAIN>

### 共有パス

タスクを作成する際に、データのアップロードに共有ストレージを使用できます。そのためには、共有ストレージをCVAT dockerコンテナにマウントする必要があります。この目的のためのdocker-compose.override.ymlの例：

services:  
 cvat\_server:  
 volumes:  
 - cvat\_share:/home/django/share:ro  
 cvat\_worker\_import:  
 volumes:  
 - cvat\_share:/home/django/share:ro  
 cvat\_worker\_export:  
 volumes:  
 - cvat\_share:/home/django/share:ro  
 cvat\_worker\_annotation:  
 volumes:  
 - cvat\_share:/home/django/share:ro  
 cvat\_worker\_chunks:  
 volumes:  
 - cvat\_share:/home/django/share:ro  
  
volumes:  
 cvat\_share:  
 driver\_opts:  
 type: none  
 device: /mnt/share  
 o: bind

共有デバイスパスを実際の共有パスに変更できます。

クラウドストレージをFUSEとして{{< ilink “/docs/administration/advanced/mounting\_cloud\_storages” “マウント” >}}し、後で共有として使用できます。

### メール検証

新しく登録されたユーザーに対してメール検証を有効にすることができます。Django allauthを設定してメール検証を有効にするには、[設定ファイル](https://github.com/cvat-ai/cvat/blob/develop/cvat/settings/base.py)でこれらのオプションを指定します（ACCOUNT\_EMAIL\_VERIFICATION = ‘mandatory’）。ユーザーのメールアドレスが検証されるまでアクセスは拒否されます。

ACCOUNT\_AUTHENTICATION\_METHOD = 'username\_email'  
ACCOUNT\_CONFIRM\_EMAIL\_ON\_GET = True  
ACCOUNT\_EMAIL\_REQUIRED = True  
ACCOUNT\_EMAIL\_VERIFICATION = 'mandatory'  
  
# Django用のメールバックエンド設定  
EMAIL\_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'

また、メールを送信するためにDjangoのメールバックエンドを設定する必要があります。これは使用しているメールサーバーに依存し、このチュートリアルではカバーされていません。詳細については、[Django SMTPバックエンド設定](https://docs.djangoproject.com/en/3.1/topics/email/#django.core.mail.backends.smtp.EmailBackend)を参照してください。

### ScalewayパブリッククラウドにCVATをデプロイする

[このチュートリアル](https://blog.scaleway.com/smart-data-annotation-for-your-computer-vision-projects-cvat-on-scaleway/)に従って、ScalewayクラウドインスタンスにCVATをインストールし、マウントされたオブジェクトストレージバケット内のデータを使用してリモートアクセスを設定します。

### HTTPSを使用した安全なCVATインスタンスのデプロイ

Traefikを使用すると、Let’s EncryptからドメインのTLS証明書を自動的に取得でき、ウェブサイトへのアクセスにHTTPSプロトコルを使用できるようになります。

これを有効にするには、まずCVAT\_HOST（ウェブサイトのドメイン）とACME\_EMAIL（Let’s Encryptの連絡先メールアドレス）環境変数を設定します：

export CVAT\_HOST=<YOUR\_DOMAIN>  
export ACME\_EMAIL=<YOUR\_EMAIL>

次に、docker-compose.https.ymlファイルを使用して、ベースのdocker-compose.ymlファイルをオーバーライドします：

docker compose -f docker-compose.yml -f docker-compose.https.yml up -d

ファイアウォールでは、ポート80と443が任意の場所からのインバウンド接続に対して開いている必要があります。

すると、CVATインスタンスはあなたのドメインのポート443（HTTPS）と80（HTTP、443にリダイレクト）で利用可能になります。

### 外部データベースを使用したCVATのデプロイ

デフォルトでは、docker compose upはPostgreSQLデータベースサーバーを起動し、CVATのデータを保存するために使用されます。代わりに独自のPostgreSQLインスタンスを使用したい場合は、次のようにします。CVATはdocker-compose.ymlで使用されているPostgreSQLと同じメジャーバージョンのみをサポートすることに注意してください。

まず、データベース接続設定を持つ環境変数を定義します：

export CVAT\_POSTGRES\_HOST=<PostgreSQLホスト名> # 必須  
export CVAT\_POSTGRES\_PORT=<PostgreSQLポート> # デフォルトは5432  
export CVAT\_POSTGRES\_DBNAME=<PostgreSQLデータベース名> # デフォルトは "cvat"  
export CVAT\_POSTGRES\_USER=<PostgreSQLロール名> # デフォルトは "root"  
export CVAT\_POSTGRES\_PASSWORD=<PostgreSQLロールのパスワード> # 必須

次に、docker-compose.external\_db.ymlファイルをdocker compose upコマンドに追加します：

docker compose -f docker-compose.yml -f docker-compose.external\_db.yml up -d

### CVATイメージのプル/ビルド/更新方法

* **CVATバージョンが2.1.0以下の場合**、compose設定は常に最新のイメージタグを指しているため、dockerを使用してイメージをプルする必要があります。例：
* docker pull cvat/server:v1.7.0  
  docker tag cvat/server:v1.7.0 openvino/cvat\_server:latest  
    
  docker pull cvat/ui:v1.7.0  
  docker tag cvat/ui:v1.7.0 openvino/cvat\_ui:latest
* **CVATバージョンがv2.1.0より大きい場合**、CVAT\_VERSION環境変数を使用して、DockerHubから特定のバージョンのビルド済みイメージをプルすることができます（例：dev）：
* CVAT\_VERSION=dev docker compose pull
* イメージを自分でビルドするには、docker-compose.dev.yml compose設定ファイルをdocker composeコマンドに含めます。これは、ソースコードに変更を加えたCVATをビルドする場合に便利です。
* docker compose -f docker-compose.yml -f docker-compose.dev.yml build
* ローカルイメージをlatestまたはdevタグに更新するには、以下を実行します：
* CVAT\_VERSION=dev docker compose pull
* または
* CVAT\_VERSION=latest docker compose pull

## トラブルシューティング

### 中国ユーザー向けの情報源

中国にお住まいの場合、インストールには以下の情報源を上書きする必要があります。

* apt updateを使用するために：
* [Ubuntuミラーリングヘルプ](https://mirrors.tuna.tsinghua.edu.cn/help/ubuntu/)
* コンパイル済みパッケージ：
* deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal main restricted universe multiverse  
  deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-updates main restricted universe multiverse  
  deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-backports main restricted universe multiverse  
  deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-security main restricted universe multiverse
* またはソースパッケージ：
* deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal main restricted universe multiverse  
  deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-updates main restricted universe multiverse  
  deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-backports main restricted universe multiverse  
  deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal-security main restricted universe multiverse
* [Dockerミラーステーション](https://www.daocloud.io/mirror)
* daemon.jsonファイルにレジストリミラーを追加します：
* {  
   "registry-mirrors": [  
   "http://f1361db2.m.daocloud.io",  
   "https://docker.mirrors.ustc.edu.cn",  
   "https://hub-mirror.c.163.com",  
   "https://mirror.ccs.tencentyun.com"  
   ]  
  }
* pipを使用するために：
* [PyPIミラーリングヘルプ](https://mirrors.tuna.tsinghua.edu.cn/help/pypi/)
* pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple
* npmを使用するために：
* [npmミラーリングヘルプ](https://npmmirror.com/)
* npm config set registry https://registry.npm.taobao.org/
* gitの代わりに[gitee](https://gitee.com/)を使用：
* [gitee.com上のCVATリポジトリ](https://gitee.com/monkeycc/cvat)
* 高速化ソースdocker.comを置き換えるために実行：
* curl -fsSL https://mirrors.aliyun.com/docker-ce/linux/ubuntu/gpg | sudo apt-key add -  
  sudo add-apt-repository \  
   "deb [arch=amd64] https://mirrors.aliyun.com/docker-ce/linux/ubuntu \  
   $(lsb\_release -cs) stable"
* *(注: download.docker.comを一般的な中国のミラーであるaliyunに置き換えました。必要に応じて調整してください。)*
* 高速化ソースgoogle.comを置き換えるために実行： *(注: google.com GPGキーの直接的な代替は、ローカルネットワーク設定によっては難しい場合があります。多くの場合、中国のミラーで設定されたパッケージマネージャーはこれを暗黙的に処理します。dl-ssl.google.comへの直接アクセスがブロックされている場合、ユーザーはキーを取得する別の方法を見つけるか、キーをキャッシュしているミラーを利用する必要があるかもしれません。元のコマンドは参考のために残していますが、中国のユーザーは状況に応じて対応する必要があります。)*
* curl https://dl-ssl.google.com/linux/linux\_signing\_key.pub | sudo apt-key add -

### 証明書が原因でHTTPSが機能しない

SSL接続に問題がある場合、原因を特定するには、次のコマンドを実行してtraefikからログを取得する必要があります：

docker logs traefik

ログは問題を見つけるのに役立ちます。

エラーがファイアウォールに関連している場合：

* ポート80と443を任意の場所からのインバウンド接続に対して開きます。
* acme.jsonを削除します。場所はおそらく次のようになります：/var/lib/docker/volumes/cvat\_cvat\_letsencrypt/\_data/acme.json。

acme.jsonが削除された後、すべてのcvat dockerコンテナを停止します：

docker compose -f docker-compose.yml -f docker-compose.https.yml down

変数が（あなたの値で）設定されていることを確認します：

export CVAT\_HOST=<YOUR\_DOMAIN>  
export ACME\_EMAIL=<YOUR\_EMAIL>

そしてdockerを再起動します：

docker compose -f docker-compose.yml -f docker-compose.https.yml up -d