--- English ---

Complete Deployment Guide for the Orchestrator on a VPS (Ubuntu 22.04)

This guide provides a comprehensive, step-by-step process for installing and configuring your orchestrator server on a public VPS.

Step 0: Prerequisites

A VPS Server: A VPS running Ubuntu 22.04. You will need its public IP address and root access (or a user with sudo privileges).

A Domain Name (Recommended): While you can use the IP, a domain is more professional. Point your domain (e.g., your-domain.com) or a subdomain (e.g., api.your-domain.com) to your VPS's public IP address via your domain provider's DNS records.

The orchestrator.py file: Have the latest version of the orchestrator code ready on your local computer.

Step 1: Connect and Update Your Server

Connect to your VPS via SSH, replacing your_user and your_public_ip.

```
ssh your_user@your_public_ip
```

Once logged in, update the system's package list:

```
sudo apt update && sudo apt upgrade -y
```

Step 2: Install System Dependencies

Install Python, its package manager pip, the virtual environment tool, and the NGINX web server.

```
sudo apt install python3 python3-pip python3-venv nginx -y
```

Step 3: Prepare the Application Environment

We will create a dedicated folder for the project and a Python virtual environment to keep dependencies isolated.

Create the project directory:

```
sudo mkdir -p /var/www/helios_orchestrator
```

Create a virtual environment:

```
sudo python3 -m venv /var/www/helios_orchestrator/venv
```

Set permissions for your user to manage the files:

```
sudo chown -R $USER:$USER /var/www/helios_orchestrator
```

Upload your orchestrator.py file:

From your local computer, use the scp command.

```
# Run this on your local machine, not the VPS
scp /path/to/your/orchestrator.py your_user@your_public_ip:/var/www/helios_orchestrator/
```

Create the uploads folder inside the project directory on the VPS:

```
# Back on your VPS terminal
mkdir /var/www/helios_orchestrator/uploads
```

Step 4: Install Python Dependencies

Now, we will install the necessary Python libraries inside the virtual environment.

Activate the virtual environment:

```
source /var/www/helios_orchestrator/venv/bin/activate
```

(You will see (venv) at the beginning of your terminal prompt).

Install the libraries:

```
pip install "fastapi[all]" gunicorn
```

Deactivate the environment for now:

```
deactivate
```

Step 5: Create the systemd Service

This will run our orchestrator as a system service, ensuring it starts on boot and restarts if it fails.

Create and edit the service file:

```
sudo nano /etc/systemd/system/helios-orchestrator.service
```

Copy and paste the entire content of the helios-orchestrator.service file provided in this guide. Save (Ctrl+O, Enter) and exit (Ctrl+X).

Start and enable the service:

```
sudo systemctl start helios-orchestrator
sudo systemctl enable helios-orchestrator
```

Check its status:

```
sudo systemctl status helios-orchestrator
```

(You should see a green "active (running)" message).

Step 6: Configure NGINX as a Reverse Proxy

NGINX will handle traffic from the internet and forward it securely to our application.

Create and edit the NGINX configuration file:

```
sudo nano /etc/nginx/sites-available/helios-orchestrator
```

Copy and paste the entire content of the helios-orchestrator.nginx file provided in this guide. Important: Replace your_domain.com with your actual domain name. If you don't have a domain, use your public IP address.

Enable the configuration by creating a symbolic link:

```
sudo ln -s /etc/nginx/sites-available/helios-orchestrator /etc/nginx/sites-enabled
```

Test the NGINX configuration for errors:

```
sudo nginx -t
```

(It should say "syntax is ok" and "test is successful").

Restart NGINX to apply the changes:

```
sudo systemctl restart nginx
```

Step 7: Configure the Firewall

Allow NGINX to receive traffic.

```
sudo ufw allow 'Nginx Full'
sudo ufw enable
```

At this point, you should be able to access your application at http://your_domain.com.

Step 8: (Highly Recommended) Secure with HTTPS

This final step encrypts all communication to and from your server.

Install Certbot, the Let's Encrypt client:

```
sudo apt install certbot python3-certbot-nginx -y
```

Obtain and install the SSL certificate. Replace your_domain.com with your domain.

```
sudo certbot --nginx -d your_domain.com
```

Certbot will ask for your email and for you to agree to the terms. It will also ask if you want to redirect HTTP traffic to HTTPS. Choose option 2 (Redirect).

Congratulations! Your orchestrator is now securely deployed and running 24/7 at https://your_domain.com. Remember to update the ORCHESTRATOR_PUBLIC_URL in your worker.py file before distributing it.