Self-Hosted AI Development Environment on AWS EC2: A Comprehensive Guide

Compiled by Abdur-Rahman Bilal

Check out a summary here: https://aramb.dev/aws-research/gemini

aramb-dev

Table of Contents:

1. Introduction

This report provides a comprehensive guide for establishing a personal, self-hosted AI development environment on Amazon Web Services (AWS) Elastic Compute Cloud (EC2). The objective is to replicate the convenience of cloud-based AI developer tools (such as Replit Ghostwriter, Vercel v0, or Cursor AI) while maintaining full control over the environment, running models locally, and eliminating recurring Software-as-a-Service (SaaS) subscription fees.

The focus is on creating a powerful, flexible, and cost-effective setup tailored for developers. This involves selecting appropriate EC2 instances capable of running demanding AI models, automating the deployment of AI backends like Ollama or LocalGPT, configuring secure and clean access via the Nginx web server, enabling remote development through a browser-based VS Code interface (Code-Server), ensuring service reliability through automated restarts, and implementing cost management strategies via scheduled instance shutdowns.

This guide covers the analysis of hardware requirements, particularly GPU Video RAM (VRAM), comparisons between relevant EC2 instance types, outlines for automated setup scripts, detailed configuration examples for Nginx and service management, and a thorough monthly cost estimation based on different usage patterns and instance choices. The aim is to deliver a practical roadmap for building a robust, self-managed AI assistant and coding environment hosted on AWS infrastructure.

2. EC2 Instance Selection for AI Workloads

Choosing the right EC2 instance is critical for running AI models effectively. The primary considerations are GPU capabilities (especially VRAM), system RAM, CPU performance, and cost.

2.1. Understanding VRAM Requirements

The amount of available VRAM on the GPU is frequently the most significant bottleneck when running large language models (LLMs). Model size, typically measured in billions of parameters, directly correlates with VRAM consumption. However, techniques like quantization can significantly reduce this footprint.

Quantization involves reducing the numerical precision of the model's parameters (weights). Common precisions include full precision (FP32 - 4 bytes per parameter), half-precision (FP16 or BF16 - 2 bytes per parameter), 8-bit integers (INT8 - 1 byte per parameter), and even lower-bit representations like 4-bit or 5-bit.[1] Lower precision drastically reduces VRAM requirements and can speed up inference, but excessively

aggressive quantization (e.g., below 4-bit or specific low-bit quantizations like 2.x bpw) might negatively impact the model's accuracy, coherence, or ability to follow complex instructions.[2]

For developer-focused AI agents, where tasks often involve code generation or complex reasoning, maintaining model quality is crucial. While quantization is often necessary to fit larger models onto affordable hardware, striking a balance is key. Quantization levels around 4-bit (e.g., Q4_K_M in GGUF format) or 5-bit often provide a good compromise between VRAM savings and performance preservation for many models.[2] However, some reports suggest that even Mixtral quality can degrade below Q4 for coding tasks [4], and Llama 3 70B might show reduced precision below Q3/Q4.[3] This implies that selecting an instance with sufficient VRAM to support at least Q4 or Q5 quantization for the desired models is advisable for a satisfactory developer experience.

The table below provides estimated VRAM requirements for some of the models mentioned in the user query, based on common quantization levels. These are estimates, and actual usage can vary based on the specific quantization method (e.g., GGUF, EXL2, AWQ), context size used during inference, and the inference engine itself (Ollama, llama.cpp, etc.).[9]

Model Name	Quantization Level	Estimated VRAM (GB)	Source(s)
Llama 3 70B Instruct	FP16	~141 - 168	[2]
Llama 3 70B Instruct	Q8_0 (8-bit)	~75	[6]
Llama 3 70B Instruct	Q5_K_M / Q5_0 (5-bit)	~49 - 50	[6]
Llama 3 70B Instruct	Q4_K_M / Q4_0 (4-bit)	~40 - 43	[2]
Llama 3 70B Instruct	Q3_K_L (3-bit)	~37	[6]
Llama 3 70B Instruct	Q2_K (2-bit)	~26	[6]

Mixtral 8x7B Instruct	FP16	~90 - 94	[7]
Mixtral 8x7B Instruct	INT8 (8-bit)	~45 - 52	[7]
Mixtral 8x7B Instruct	4-bit (e.g., Q4_K_M)	~27 - 30	[5]
Mixtral 8x7B Instruct	3.5 bpw (EXL2)	~24	[5]
DeepSeek Coder 6.7B	Q4_K_M (4-bit)	~4.5 - 5.5	(Rule of thumb: ~0.6-0.7GB per B params for Q4 [16], plus context)
Phi-3.5-mini-instruct (3.8B)	Q4_K_M (4-bit)	~2.5 - 3.5	(Rule of thumb: ~0.6-0.7GB per B params for Q4 [16], plus context)
Llama 3 8B Instruct	Q8_0 (8-bit)	~8.5	[6]
Llama 3 8B Instruct	Q4_K_M (4-bit)	~4.9	[6]
Mistral 7B Instruct	Q4_K_M (4-bit)	~4.5 - 5.5	(Similar to Llama 8B Q4)
StarCoder (e.g., 15B)	Q4_K_M (4-bit)	~10 - 12	(Rule of thumb: ~0.6-0.7GB per B params for Q4 [16], plus context)
Llama 3.2 Vision 11B	Default (quantized)	~8+	[17]
Llama 3.2 Vision 90B	Default (quantized)	~64+	[17]

Note: bpw = bits per weight. QX_Y are GGUF quantization levels. Context size significantly impacts VRAM; these estimates assume moderate context.

A key observation is the substantial VRAM requirement for the Llama 3 70B model. Even with 4-bit quantization, it demands around 40-50GB of VRAM.[2] Running this

model comfortably often necessitates instances with multiple GPUs or a single high-VRAM GPU (like an A100 80GB, which is expensive). While it's possible to run such models on GPUs with less VRAM (e.g., 24GB like an RTX 3090/4090 or AWS A10G) by offloading layers to system RAM, this significantly degrades performance (inference speed), potentially yielding only a few tokens per second or less.[4] Achieving a balance between cost and performance for 70B models might involve accepting slower speeds or prioritizing smaller, yet capable, models like Mixtral 8x7B or various 7-13B parameter models that fit more easily within the VRAM of cost-effective single-GPU instances.

System RAM also plays a role, especially if offloading layers from VRAM is necessary. A minimum of 24GB is often cited, but 32GB, 64GB, or even more is recommended for smoother operation, particularly with larger models or when significant offloading occurs.[1] CPU core count is less critical for inference speed once the model is loaded (as GPU handles the main computation), but sufficient cores (e.g., 8+) are needed for general system operation, data handling, and managing the inference server.[1]

2.2. Comparing GPU Instance Families (G4dn vs. G5)

For GPU-accelerated workloads on AWS, the G4dn and G5 instance families are primary candidates.

- **G4dn Instances:** Powered by NVIDIA T4 Tensor Core GPUs (16GB VRAM each). They offer a cost-effective entry point for GPU workloads, suitable for moderate AI inference and smaller model training.[18] They are widely available and have been a standard choice for some time.[20]
- **G5 Instances:** Feature newer NVIDIA A10G Tensor Core GPUs (24GB VRAM each). They deliver significantly better performance for both graphics and machine learning compared to G4dn instances. AWS benchmarks show up to 3x higher ML inference performance and up to 40% better price/performance.[21] The A10G GPUs also have more VRAM (24GB vs 16GB), which is crucial for LLMs. G5 instances are built on the AWS Nitro System, providing near bare-metal performance.[21]

While G4dn instances are cheaper per hour [19], the superior performance and price/performance of G5 instances generally make them the preferred choice for demanding ML inference tasks like running LLMs.[21] The increased VRAM per GPU (24GB) is also a significant advantage, allowing larger models or less aggressive quantization compared to the T4's 16GB. Newer G6 instances with NVIDIA L4 GPUs exist, but LLM inference is often memory-bandwidth bound rather than purely compute-bound, potentially making the A10G in G5 instances a very suitable and

potentially more cost-effective choice still.[21]

Recommendation: Prioritize G5 instances for this use case due to their superior inference performance, better price/performance ratio, and higher VRAM per GPU compared to G4dn instances.[21] G4dn serves as a lower-cost alternative if budget constraints are severe and only smaller models (fitting within 16GB VRAM) are acceptable.

2.3. Recommended Instances

Based on the VRAM requirements and the G5 family preference, here are specific instance recommendations, balancing capability and cost:

1. **Baseline (Smaller Models ~7-13B, Mixtral low-quant):**
    - g5.xlarge: 1x A10G (24GB VRAM), 4 vCPU, 16GB RAM.[21]
    - *Suitability:* Capable of running models up to ~13B comfortably, and potentially Mixtral 8x7B with specific 4-bit or lower quantizations (like 3.5bpw EXL2 needing ~24GB [5], or BNB 4-bit needing ~27GB [14] which requires some RAM offload). A good starting point balancing cost and capability.
2. **Improved Baseline (Smoother Operation):**
    - g5.2xlarge: 1x A10G (24GB VRAM), 8 vCPU, 32GB RAM.[21]
    - *Suitability:* Same VRAM as g5.xlarge, but double the CPU and RAM. Better for multitasking (e.g., running Code-Server alongside Ollama) and provides more headroom if minor layer offloading to RAM is needed.
3. **Comfortable Mid-Range (Mixtral Q4/Q5, Llama 70B low-quant with offload):**
    - g5.4xlarge: 1x A10G (24GB VRAM), 16 vCPU, 64GB RAM.[21]
    - g5.8xlarge: 1x A10G (24GB VRAM), 32 vCPU, 128GB RAM.[21]
    - *Suitability:* Still limited by the 24GB VRAM, but the substantial system RAM allows for more significant layer offloading if attempting to run larger models like Llama 3 70B at lower quantizations (e.g., Q2/Q3 needing 26-37GB [6]) or Mixtral at better quantizations (e.g., Q4 needing ~27-30GB [5]). Performance will still be impacted by offloading.[4]
4. **Running Llama 3 70B (Quantized) Well:**
    - g5.12xlarge: 4x A10G (96GB total VRAM), 48 vCPU, 192GB RAM.[21]
    - *Suitability:* Provides ample VRAM (96GB) to run Llama 3 70B at good quantization levels (e.g., Q4/Q5 needing ~40-50GB [2]) entirely on the GPUs, ensuring much better performance than offloading scenarios. Can also handle Mixtral comfortably at higher precision (e.g., 8-bit needing ~45-52GB [8]). This is the recommended starting point for a good 70B model experience.
5. **High-End (Multiple Large Models / Higher Precision):**
    - g5.24xlarge: 4x A10G (96GB VRAM), 96 vCPU, 384GB RAM.[21]

    - g5.48xlarge: 8x A10G (192GB VRAM), 192 vCPU, 768GB RAM.[21]
    - *Suitability:* For running multiple large models concurrently, using less aggressive quantization (e.g., Llama 3 70B FP16 needing ~140GB+ [2]), or handling very large context windows which also consume significant VRAM.[9]
6. **G4dn Alternative (Budget Focus, Small Models Only):**
    - g4dn.xlarge: 1x T4 (16GB VRAM), 4 vCPU, 16GB RAM.[19]
    - *Suitability:* Lowest cost GPU option. The 16GB VRAM significantly limits model size, generally suitable for 7B/8B models or perhaps very specific 13B quantizations. Not recommended for Mixtral or Llama 70B.

2.4. CPU-Only Considerations

While Ollama and LocalGPT can technically run models using only the CPU, performance for the types of models requested (7B and larger) will be extremely slow, often taking many seconds or even minutes per response.[3] This negates the goal of having a convenient and responsive AI assistant. Therefore, CPU-only instances (like AWS c-series or r-series [32]) are not recommended for this setup.

2.5. Cost Optimization: On-Demand vs. Spot Instances

AWS offers different pricing models for EC2 instances:

- **On-Demand:** Pay a fixed rate per hour (or second) with no long-term commitment. Predictable but highest cost.[22]
- **Reserved Instances (RIs) / Savings Plans:** Commit to 1 or 3 years of usage for significant discounts (up to 72%) compared to On-Demand.[19] Less flexible.
- **Spot Instances:** Bid on unused EC2 capacity, offering substantial savings (often 70-90% off On-Demand prices).[22] The major drawback is that AWS can reclaim the instance with only a two-minute warning if the capacity is needed elsewhere or your bid price is exceeded.

For a personal development server intended for use during specific hours and automatically shut down otherwise, **Spot Instances represent a highly effective cost-saving strategy.** The risk of interruption during active use hours is relatively low for many instance types/regions, and the daily shutdown minimizes the impact of potential overnight interruptions. The significant cost savings align well with the user's goal of a cost-effective solution. On-Demand remains an option for guaranteed availability if interruptions cannot be tolerated.

2.6. Table: Recommended EC2 Instance Comparison (us-east-1, Linux)

The following table summarizes the key specifications and estimated hourly pricing for the recommended instances in the us-east-1 (N. Virginia) region. Spot prices

fluctuate based on demand and are estimates (often ~70% less than On-Demand).

Instance Type	GPU Type & Count	Total VRAM (GB)	vCPUs	System RAM (GiB)	Instance Storage (GB)	Network (Gbps)	On-Demand Price/hr*	Est. Spot Price/hr*
g4dn.xlarge	1 x NVIDIA T4	16	4	16	1 x 125 NVMe SSD	Up to 25	$0.526	~$0.16
g5.xlarge	**1 x A10G**	**24**	**4**	**16**	**1 x 250 NVMe SSD**	**Up to 10**	**$1.006**	**~$0.30**
g5.2xlarge	1 x A10G	24	8	32	1 x 450 NVMe SSD	Up to 10	$1.212	~$0.36
g5.4xlarge	1 x A10G	24	16	64	1 x 600 NVMe SSD	Up to 25	$1.624	~$0.49
g5.12xlarge	**4 x A10G**	**96**	**48**	**192**	**1 x 3800 NVMe SSD**	**40**	**$5.672**	**~$1.70**
g5.24xlarge	4 x A10G	96	96	384	1 x 3800 NVMe SSD	50	$8.144	~$2.44
g5.48xlarge	8 x A10G	192	192	768	2 x 3800 NVMe SSD	100	$16.288	~$4.89

Prices as of late 2024/early 2025, subject to change. Spot prices are estimates and fluctuate. Check current AWS pricing. Sources: [19]

3. Automated Deployment and Setup

This section outlines the process for deploying the chosen AI backend and necessary supporting software onto the selected EC2 instance, aiming for automation where possible.

3.1. Choosing Your AI Backend (Ollama vs. LocalGPT)

Two primary options were mentioned for running the local AI models: Ollama and LocalGPT.

- **Ollama:** This platform focuses on simplifying the process of downloading, running, and managing LLMs locally.[34] It provides a command-line interface and an API server (running by default on http://127.0.0.1:11434 [36]) that can be easily integrated with various frontends and tools, including popular web UIs like Open WebUI.[34] Ollama has strong community support and frequently adds new models.[40] Its installation is generally straightforward, often involving a single script.[41] Docker images are also readily available.[39]
- **LocalGPT:** This project is specifically designed for interacting with local documents using Retrieval-Augmented Generation (RAG).[47] It involves setting up a Python environment, installing specific dependencies (including potentially complex ones like llama-cpp-python with GPU bindings [48]), ingesting documents into a vector database, and then running query scripts or a UI.[48] While powerful for its specific use case, the setup is generally more involved than Ollama.[47] Docker support exists but might require careful configuration.[48]

Recommendation: For a general-purpose AI agent aimed at developer assistance, similar to the cloud services mentioned, **Ollama is the recommended starting point.** Its ease of installation, robust API, wide model support, and integration with user interfaces align better with the goals of convenience and broad utility. LocalGPT is a better fit if the primary objective shifts specifically to querying personal document collections.

3.2. Prerequisites

Before starting the deployment, ensure the following are in place:

- **AWS Account:** An active AWS account with permissions to manage EC2 instances, IAM roles and policies, Security Groups, and potentially Route 53 (for domains), Lambda, and EventBridge (for auto-shutdown).[55] It is best practice to use an IAM user with specific permissions rather than the root account.
- **AWS CLI:** The AWS Command Line Interface installed and configured on your local machine can be helpful for interacting with AWS services during setup or troubleshooting.[56]

- **Domain Name (Optional):** A registered domain name is required if you want to access the services via a user-friendly URL (e.g., ai.yourdomain.com) and easily secure it with a standard Let's Encrypt SSL certificate.[55] Access via IP address is possible but less convenient and harder to secure with trusted SSL.
- **EC2 Instance:** An EC2 instance launched based on the recommendations in Section 2. Ensure the security group associated with the instance allows inbound traffic on port 22 (SSH) from your IP address for setup, and potentially ports 80 (HTTP) and 443 (HTTPS) from anywhere if using Nginx and a domain name.[18]

3.3. Core Dependencies & Environment Setup

Setting up the instance correctly involves installing several key components:

- **Operating System: Ubuntu 22.04 LTS** or **Ubuntu 24.04 LTS** are excellent choices due to their widespread use, extensive documentation, and compatibility with necessary tools.[47] Alternatively, using an **AWS Deep Learning AMI (DLAMI)** based on Ubuntu or Amazon Linux 2 is highly recommended.[18]
- **NVIDIA Drivers & CUDA Toolkit:** These are essential for leveraging the GPU.
  - **Using DLAMI (Highly Recommended):** The simplest approach is to launch an EC2 instance using a recent AWS Deep Learning AMI (e.g., "Deep Learning AMI GPU PyTorch" or "Deep Learning Base GPU AMI" for Ubuntu). These AMIs come pre-packaged with compatible NVIDIA drivers, CUDA toolkit, cuDNN, and often other useful tools like Docker, saving significant setup time and potential compatibility headaches.[18] Verify the installation post-launch using nvidia-smi.
  - **Manual Installation (If not using DLAMI):**
    1. Install drivers: Use the ubuntu-drivers utility: sudo apt update && sudo apt install -y ubuntu-drivers-common && sudo ubuntu-drivers install && sudo reboot.[59] Verify with nvidia-smi.
    2. Install CUDA Toolkit: Add the NVIDIA CUDA repository and install the toolkit version compatible with your driver and Ollama/PyTorch requirements (Ollama generally works with CUDA 11.x and 12.x).[41] Follow official NVIDIA instructions carefully.[64]
- **Docker & Docker Compose:** Containerization is recommended for running Ollama and Code-Server. It isolates dependencies, simplifies updates, and makes management cleaner.[39]
  - Install Docker Engine following the official Docker documentation for Ubuntu.
  - Install the Docker Compose plugin (usually included with recent Docker Engine installs or installed separately via apt).
- **NVIDIA Container Toolkit:** This allows Docker containers to access the host's NVIDIA GPU resources.[45]

- Installation involves adding the NVIDIA container toolkit repository and installing the nvidia-container-toolkit package, followed by restarting the Docker service.[46] Commands typically look like:
  Bash
  ```
  curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \
  && curl -s -L https://nvidia.github.io/libnvidia-container/stable/deb/nvidia-container-toolkit.list | \
  sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-toolkit-keyring.gpg] https://#g' | \
  sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
  sudo apt-get update
  sudo apt-get install -y nvidia-container-toolkit
  sudo nvidia-ctk runtime configure --runtime=docker
  sudo systemctl restart docker
  ```
  (Commands adapted from [46])

The combination of using an AWS Deep Learning AMI and Docker offers a synergistic advantage. The DLAMI handles the complex and often error-prone low-level GPU driver and CUDA installation [18], while Docker provides a clean, isolated environment for the applications (Ollama, Code-Server), preventing conflicts with system libraries or other installed software.[44] This approach balances ease of setup with maintainable application deployment.

3.4. Unified Setup Script Outline (setup.sh)

To automate the provisioning process on a fresh EC2 instance (ideally based on a recent Ubuntu DLAMI), a setup script can be used. Below is an outline of the steps such a script would perform. Note that this is a template and requires customization (passwords, domain names, specific model choices, etc.).

Bash

```
#!/bin/bash
set -e # Exit immediately if a command exits with a non-zero status.
```

```
echo ">>> Starting AI Server Setup Script <<<"

# --- 1. System Update & Essential Tools ---
echo "Updating packages and installing essentials..."
sudo apt-get update && sudo apt-get upgrade -y
sudo apt-get install -y git curl wget nano nginx python3-pip tree # Install Nginx early

# --- 2. Docker & Docker Compose (If not pre-installed on DLAMI) ---
# Check if Docker is installed, install if necessary
if! command -v docker &> /dev/null
then
    echo "Docker not found. Installing Docker Engine..."
    # Add official Docker install steps here
    # Example (verify official steps for current Ubuntu version):
    sudo apt-get install -y ca-certificates curl
    sudo install -m 0755 -d /etc/apt/keyrings
    sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc
    sudo chmod a+r /etc/apt/keyrings/docker.asc
    echo \
      "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu \
     $(. /etc/os-release && echo "$VERSION_CODENAME") 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-buildx-plugin
docker-compose-plugin
    sudo usermod -aG docker $USER # Add current user to docker group
    echo "Docker installed. Please log out and log back in for group changes to take effect, or run
'newgrp docker'."
else
    echo "Docker already installed."
fi

# --- 3. NVIDIA Container Toolkit (If not pre-installed on DLAMI) ---
# Check if nvidia-container-toolkit is installed, install if necessary
if! dpkg -l | grep -q nvidia-container-toolkit
then
    echo "NVIDIA Container Toolkit not found. Installing..."
    # Add NVIDIA Container Toolkit install steps here (verify official steps)
    curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --dearmor
```

```
-o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \
  && curl -s -L
https://nvidia.github.io/libnvidia-container/stable/deb/nvidia-container-toolkit.list | \
  sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-toolkit-keyring.gpg]
https://#g' | \
  sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list > /dev/null
  sudo apt-get update
  sudo apt-get install -y nvidia-container-toolkit
  sudo nvidia-ctk runtime configure --runtime=docker
  sudo systemctl restart docker
  echo "NVIDIA Container Toolkit installed."
else
  echo "NVIDIA Container Toolkit already installed."
fi
# Verify GPU access within Docker
sudo docker run --rm --gpus all nvidia/cuda:12.0.0-base-ubuntu22.04 nvidia-smi

# --- 4. Create Project Directories ---
echo "Creating project directories..."
mkdir -p ~/ai-server/ollama_data
mkdir -p ~/ai-server/code-server_data
cd ~/ai-server

# --- 5. Create docker-compose.yml ---
echo "Creating docker-compose.yml..."
cat << EOF > docker-compose.yml
version: '3.8'
services:
 ollama:
  image: ollama/ollama:latest
  container_name: ollama
  ports:
   - "127.0.0.1:11434:11434" # Bind only to localhost for security
  volumes:
   -./ollama_data:/root/.ollama
  deploy:
   resources:
    reservations:
     devices:
      - driver: nvidia
        count: all # Use all available GPUs
        capabilities: [gpu]
  restart: unless-stopped
```

```
  code-server:
    image: codercom/code-server:latest
    container_name: code-server
    user: "\$(id -u):\$(id -g)" # Run as host user to avoid permission issues
    ports:
      - "127.0.0.1:8080:8080" # Bind only to localhost
    volumes:
      -./code-server_data:/home/coder/.local/share/code-server
      - \$HOME:/home/coder/project # Mount home directory or specific project dir
    environment:
      - PASSWORD=YourStrongPasswordHere # CHANGE THIS
    restart: unless-stopped

volumes:
  ollama_data:
  code-server_data:

EOF
echo "Remember to set a strong PASSWORD in docker-compose.yml!"

# --- 6. Create Nginx Configuration ---
echo "Creating Nginx site configuration..."
# Placeholder - Actual config will be more detailed (see Section 4)
# User needs to replace 'your_domain.com'
sudo tee /etc/nginx/sites-available/ai-server.conf > /dev/null <<EOF
server {
    listen 80;
    server_name your_domain.com; # CHANGE THIS

    # Redirect HTTP to HTTPS (Certbot often handles this)
    location / {
        return 301 https://\$host\$request_uri;
    }
}

server {
    listen 443 ssl http2;
    server_name your_domain.com; # CHANGE THIS

    # SSL Configuration (Managed by Certbot)
    ssl_certificate /etc/letsencrypt/live/your_domain.com/fullchain.pem; # CHANGE THIS
    ssl_certificate_key /etc/letsencrypt/live/your_domain.com/privkey.pem; # CHANGE THIS
    include /etc/letsencrypt/options-ssl-nginx.conf;
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;

    # Security Headers (Add more here - see Section 4)
```

```
    add_header Strict-Transport-Security "max-age=31536000; includeSubDomains; preload" always;
    add_header X-Frame-Options "SAMEORIGIN" always;
    add_header X-Content-Type-Options "nosniff" always;

    location / {
        # Proxy to Code-Server
        proxy_pass http://127.0.0.1:8080;
        proxy_set_header Host \$host;
        proxy_set_header Upgrade \$http_upgrade;
        proxy_set_header Connection upgrade;
        proxy_set_header Accept-Encoding gzip;
        proxy_set_header X-Real-IP \$remote_addr;
        proxy_set_header X-Forwarded-For \$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto \$scheme;
    }

    location /ollama/ {
        # Proxy to Ollama API
        proxy_pass http://127.0.0.1:11434/;
        proxy_set_header Host \$host;
        proxy_set_header X-Real-IP \$remote_addr;
        proxy_set_header X-Forwarded-For \$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto \$scheme;

        # Optional: Add authentication or rate limiting here if needed
    }
}
EOF
sudo ln -sfn /etc/nginx/sites-available/ai-server.conf
/etc/nginx/sites-enabled/ai-server.conf
sudo nginx -t # Test configuration
sudo systemctl reload nginx

# --- 7. Install Certbot (Optional, if using domain) ---
echo "Installing Certbot (optional)..."
# Using snap is recommended by Certbot
if! command -v certbot &> /dev/null
then
    sudo snap install core; sudo snap refresh core
    sudo snap install --classic certbot
    sudo ln -s /snap/bin/certbot /usr/bin/certbot
    echo "Certbot installed via snap."
    echo "Run 'sudo certbot --nginx -d your_domain.com' after DNS propagates."
else
    echo "Certbot already installed or install manually if needed."
```

```
fi

# --- 8. Start Services ---
echo "Starting services via Docker Compose..."
cd ~/ai-server
sudo docker compose up -d

# --- 9. Pull Initial Ollama Models ---
echo "Pulling initial Ollama models (this may take time)..."
# Add desired models here
sudo docker exec ollama ollama pull llama3:8b
sudo docker exec ollama ollama pull mistral
# Add other models like deepseek-coder, phi-3, etc.
# Example: sudo docker exec ollama ollama pull deepseek-coder:6.7b

echo ">>> Setup Script Finished <<<"
echo "Access Code-Server at https://your_domain.com (after running Certbot and DNS setup)"
echo "Access Ollama API via https://your_domain.com/ollama/"
echo "Manage services with 'sudo docker compose stop/start/logs' in ~/ai-server"
```

This script provides a solid foundation. The user must replace placeholders like your_domain.com and YourStrongPasswordHere, potentially adjust mounted volumes, and select the initial Ollama models to download. The auto-shutdown mechanism (Section 7) needs separate setup (Lambda/EventBridge or cron).

4. Configuring Nginx for Secure Access

Nginx serves as a robust and efficient web server and reverse proxy. In this setup, it acts as the secure front door to the backend AI services (Ollama API) and the remote development environment (Code-Server).

4.1. Nginx Installation

Nginx is typically installed via the system's package manager. The setup script outline in Section 3 includes sudo apt install -y nginx.

4.2. Basic Reverse Proxy Concepts

A reverse proxy sits in front of one or more backend servers (like the Ollama API server running on port 11434 or Code-Server on port 8080). It receives incoming requests from clients (e.g., a web browser or an API client) and forwards them to the appropriate backend service.[67] After the backend service processes the request, it

sends the response back to Nginx, which then relays it to the original client.[67]

This architecture offers several advantages:

- **Single Point of Access:** Clients connect to a single domain/IP address managed by Nginx, simplifying access.[68]
- **SSL/TLS Termination:** Nginx can handle HTTPS encryption and decryption, offloading this task from the backend applications.[57]
- **Security Layer:** It hides the backend services' direct ports and allows for centralized security policies, firewall rules, and potentially access control.[36]
- **Load Balancing:** While not needed for this single-instance setup, Nginx can distribute traffic across multiple backend servers.
- **Serving Static Files:** Nginx is highly efficient at serving static assets directly.

4.3. Nginx Configuration File

Nginx configurations are typically managed through site-specific files located in /etc/nginx/sites-available/. These are then enabled by creating symbolic links to them in the /etc/nginx/sites-enabled/ directory.[57] The setup script outline creates /etc/nginx/sites-available/ai-server.conf and links it.

4.4. Reverse Proxy for Ollama API

Ollama's API server runs on 127.0.0.1:11434 by default.[36] Directly exposing this port to the network (e.g., by setting OLLAMA_HOST=0.0.0.0 [36]) is **highly discouraged** as the API lacks built-in authentication. An exposed Ollama API could allow unauthorized users to consume significant GPU resources.[70]

Nginx provides a secure way to expose the API. A location block in the Nginx configuration forwards requests to Ollama. It's common practice to place the API under a specific path, like /ollama/.

Nginx

```
# Inside the 'server' block of /etc/nginx/sites-available/ai-server.conf

location /ollama/ {
    # Note the trailing slash on proxy_pass, important for path handling
    proxy_pass http://127.0.0.1:11434/;
```

```
    proxy_set_header Host              $host; # Forward original host header
    proxy_set_header X-Real-IP         $remote_addr; # Forward client IP
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for; # Forward proxy chain
    proxy_set_header X-Forwarded-Proto $scheme; # Forward original protocol (http/https)

    # Optional: Increase timeouts if needed for long-running model requests
    # proxy_connect_timeout       600;
    # proxy_send_timeout          600;
    # proxy_read_timeout          600;
    # send_timeout                600;
}
```

This configuration takes requests made to https://your_domain.com/ollama/ and forwards them to the Ollama server running locally on port 11434. The proxy_set_header directives ensure the backend service receives relevant information about the original request.[36]

4.5. Reverse Proxy for Code-Server

Code-Server typically runs on 127.0.0.1:8080.[60] It relies heavily on WebSockets for real-time communication between the browser and the server-side VS Code instance. Therefore, the Nginx configuration must include specific headers to support WebSocket connections.

Nginx

```
# Inside the 'server' block of /etc/nginx/sites-available/ai-server.conf

# Proxying Code-Server to the root path '/'
location / {
    proxy_pass http://127.0.0.1:8080/; # Forward to Code-Server

    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
```

```
    proxy_set_header X-Forwarded-Proto $scheme;

    # WebSocket support headers - CRITICAL for Code-Server
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";

    # Optional: Disable buffering for potentially better responsiveness
    # proxy_buffering off;

    # Optional: Increase timeouts if experiencing disconnects
    # proxy_connect_timeout       600;
    # proxy_send_timeout          600;
    # proxy_read_timeout          600;
    # send_timeout                600;
}

# Alternatively, proxying Code-Server to a subpath like '/code/'
# location /code/ {
#     proxy_pass http://127.0.0.1:8080/;
#     proxy_set_header Host $host;
#     proxy_set_header X-Real-IP $remote_addr;
#     proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
#     proxy_set_header X-Forwarded-Proto $scheme;
#     proxy_http_version 1.1;
#     proxy_set_header Upgrade $http_upgrade;
#     proxy_set_header Connection "upgrade";
#
#     # Rewrite the path (remove /code/) before sending to backend
#     rewrite ^/code/(.*)$ /$1 break;
#     # Ensure redirects from code-server maintain the subpath (might need tweaking)
#     proxy_redirect http://127.0.0.1:8080/ /code/;
# }
```

The key additions here are proxy_http_version 1.1, proxy_set_header Upgrade $http_upgrade, and proxy_set_header Connection "upgrade" which enable the protocol switch required for WebSockets.[57] Proxying to a subpath (/code/) is possible but often requires more complex configuration (rewrite, proxy_redirect) to handle internal paths correctly.[73] Proxying to the root (/) is generally simpler.

4.6. Implementing Security Headers

Adding HTTP security headers instructs browsers to enable protective mechanisms, significantly enhancing the application's security posture against common web attacks.[78] These headers should generally be added within the server block that handles HTTPS traffic.

- **Strict-Transport-Security (HSTS):** Forces browsers to use HTTPS for all future connections to the domain, preventing protocol downgrade attacks and cookie hijacking over HTTP.[78]
  Nginx
  `add_header Strict-Transport-Security "max-age=31536000; includeSubDomains; preload" always;`
  (max-age is one year; includeSubDomains applies it to subdomains; preload allows inclusion in browser preload lists).
- **X-Frame-Options:** Prevents the site from being embedded in <iframe> or <frame> elements on other domains, mitigating clickjacking attacks.[78]
  SAMEORIGIN allows framing only by pages from the same origin.
  Nginx
  `add_header X-Frame-Options "SAMEORIGIN" always;`

- **X-Content-Type-Options:** Prevents browsers from trying to guess (MIME-sniff) the content type of a resource if it differs from the declared Content-Type header. This stops attacks where malicious files might be disguised as harmless types.[79]
  Nginx
  `add_header X-Content-Type-Options "nosniff" always;`

- **Referrer-Policy:** Controls how much referrer information (the URL the user came from) is included with requests. strict-origin-when-cross-origin sends the full URL for same-origin requests but only the origin for cross-origin requests, balancing utility and privacy.[79]
  Nginx
  `add_header Referrer-Policy "strict-origin-when-cross-origin" always;`

- **Content-Security-Policy (CSP):** A powerful header that defines allowed sources for various content types (scripts, styles, images, etc.), effectively preventing Cross-Site Scripting (XSS) and data injection attacks.[78] CSP requires careful configuration based on the specific resources loaded by Code-Server and any custom frontends. A starting point might be:
  Nginx

```
# Example CSP - Needs refinement based on actual application needs!
add_header Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-inline' 'unsafe-eval'; style-src 'self' 'unsafe-inline'; img-src 'self' data:; font-src 'self'; connect-src 'self'
```

wss:;" always;
Note: 'unsafe-inline' and 'unsafe-eval' are often needed for VS Code extensions but reduce security. Refine this policy by identifying specific sources if possible.

- **X-XSS-Protection:** This header controlled older browser built-in XSS filters. Modern best practice, supported by OWASP and Mozilla, is to disable this header (0) and rely on a strong Content Security Policy, as the browser filter itself could sometimes introduce vulnerabilities.[82] Some older guides might still recommend 1; mode=block.[78] Disabling it is generally preferred now.
  Nginx
  add_header X-XSS-Protection "0" always;

- **server_tokens off;:** This directive (placed in the http or server block, not using add_header) prevents Nginx from sending its version number in error pages and the Server response header, reducing information disclosure.[78]

A potential pitfall with Nginx configuration is that add_header directives defined in a higher-level block (like server) are **not** inherited by location blocks if the location block itself contains any add_header directive.[87] To ensure security headers apply consistently, either place all add_header directives only at the server level (if no location-specific headers are needed) or meticulously repeat all necessary security headers within *every* location block that uses add_header. For this setup, defining them once at the server level (specifically the HTTPS server block) is likely sufficient and less error-prone.

4.7. Setting up SSL with Let's Encrypt (Certbot)

Securing the connection with HTTPS is essential. Let's Encrypt provides free SSL/TLS certificates, and Certbot is a tool that automates the process of obtaining and renewing these certificates.

1. **Install Certbot:** Use the recommended method for Ubuntu, which is often the snap package [58]:
   Bash
   sudo snap install core; sudo snap refresh core
   sudo snap install --classic certbot
   sudo ln -s /snap/bin/certbot /usr/bin/certbot
   Alternatively, use the apt package if preferred (common on older Ubuntu versions or if snaps are avoided) [89]:
   Bash
   sudo apt update
   sudo apt install certbot python3-certbot-nginx

2. **Configure DNS:** Ensure the domain name (e.g., ai.yourdomain.com) has an A record pointing to the public IP address of the EC2 instance.
3. **Configure Firewall:** Allow incoming HTTPS traffic on port 443. If using ufw:
   Bash
   ```
   sudo ufw allow 'Nginx Full' # Allows both HTTP & HTTPS
   sudo ufw delete allow 'Nginx HTTP' # Remove redundant HTTP rule if 'Nginx Full' is used
   sudo ufw status
   ```
   [58]
4. **Run Certbot:** Use the Nginx plugin, which automatically detects the domain from the server_name directive in your Nginx configuration and modifies the configuration file to enable SSL.
   Bash
   ```
   sudo certbot --nginx -d your_domain.com
   ```
   (Replace your_domain.com with the actual domain name configured in /etc/nginx/sites-available/ai-server.conf). Certbot will ask for an email address and agreement to terms. It will then obtain the certificate and update the Nginx config (e.g., adding listen 443 ssl;, ssl_certificate, ssl_certificate_key directives).[58] It typically also sets up HTTP to HTTPS redirection.
5. **Verify Auto-Renewal:** Certbot automatically sets up a scheduled task (systemd timer or cron job) to renew certificates before they expire.[92] Test the renewal process:
   Bash
   ```
   sudo certbot renew --dry-run
   ```

4.8. Example Nginx Configuration (/etc/nginx/sites-available/ai-server.conf)

Nginx

```
# /etc/nginx/sites-available/ai-server.conf
# Remember to replace 'your_domain.com' with your actual domain

# Optional: Redirect HTTP to HTTPS (Certbot usually handles this)
server {
    listen 80;
    listen [::]:80;
```

```
    server_name your_domain.com;

    # For Certbot challenges if needed initially, or redirect
    location /.well-known/acme-challenge/ {
        root /var/www/html; # Or adjust as needed
    }

    location / {
        return 301 https://$host$request_uri;
    }
}

server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    server_name your_domain.com;

    # SSL Configuration - Managed by Certbot
    # These paths will be automatically added/updated by 'sudo certbot --nginx'
    ssl_certificate /etc/letsencrypt/live/your_domain.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/your_domain.com/privkey.pem;
    include /etc/letsencrypt/options-ssl-nginx.conf; # Recommended SSL parameters
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # Diffie-Hellman parameters

    # Basic Security Headers (Add CSP separately if needed)
    add_header Strict-Transport-Security "max-age=31536000; includeSubDomains; preload"
always;
    add_header X-Frame-Options "SAMEORIGIN" always;
    add_header X-Content-Type-Options "nosniff" always;
    add_header Referrer-Policy "strict-origin-when-cross-origin" always;
    add_header X-XSS-Protection "0" always; # Disable browser XSS filter, rely on CSP
    # add_header Content-Security-Policy "..." always; # Add your CSP here

    # Hide Nginx version
    server_tokens off;

    # Proxy pass to Code-Server (assuming it's at the root)
    location / {
        proxy_pass http://127.0.0.1:8080/; # Forward to Code-Server default port
```

```
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;

        # WebSocket support
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";

        # Optional: Increase timeouts if needed
        # proxy_read_timeout 86400s; # Example: 24 hours
        # proxy_send_timeout 86400s;
    }

    # Proxy pass to Ollama API
    location /ollama/ {
        proxy_pass http://127.0.0.1:11434/; # Forward to Ollama default port
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;

        # Optional: Increase timeouts for potentially long Ollama requests
        # proxy_connect_timeout       600;
        # proxy_send_timeout          600;
        # proxy_read_timeout          600;
        # send_timeout                600;
    }

    # Optional: Deny access to hidden files
    location ~ /\. {
        deny all;
    }
}
```

After creating/editing this file, ensure the symlink exists (sudo ln -sfn /etc/nginx/sites-available/ai-server.conf /etc/nginx/sites-enabled/ai-server.conf), test

the configuration (sudo nginx -t), and reload Nginx (sudo systemctl reload nginx).

5. Ensuring Service Reliability (Auto-Start)

To ensure the AI backend (Ollama) and the remote development environment (Code-Server) are available after an EC2 instance reboot (e.g., after maintenance, unexpected shutdown, or manual restart), they must be configured to start automatically. The two primary methods are using the system's systemd init system or leveraging Docker Compose's restart policies.

5.1. Method 1: Using systemd

systemd is the standard init system and service manager on most modern Linux distributions, including Ubuntu.[93] It manages services using unit files (typically ending in .service) located in /etc/systemd/system/.

- **Ollama Service:** If Ollama is installed using the official installation script (curl... | sh), it usually creates and enables a systemd service file automatically at /etc/systemd/system/ollama.service.[41] The file typically contains:

  ```
  Ini, TOML
  [Unit]
  Description=Ollama Service
  After=network-online.target # Start after network is ready


  ExecStart=/usr/bin/ollama serve # Command to start Ollama
  User=ollama # Dedicated user created during install
  Group=ollama
  Restart=always # Or on-failure
  RestartSec=3
  Environment="PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin" # Ensure PATH is set
  # Add other environment variables like OLLAMA_HOST if needed here:
  # Environment="OLLAMA_HOST=127.0.0.1" # Example: Ensure binding to localhost if not default

  [Install]
  WantedBy=multi-user.target # Enable for standard multi-user runlevel
  ```

  (Content adapted from 37)
  To ensure it's enabled: sudo systemctl enable ollama.service.
- **Code-Server Service:** If Code-Server is installed manually or via a package manager that doesn't set up a service (the official script usually does [96]), a unit file needs to be created manually. Create

/etc/systemd/system/code-server.service (or code-server@.service for user-specific instances):

```ini
[Unit]
Description=code-server (VS Code in browser)
After=network.target # Start after network is ready


Type=simple
User=ubuntu # CHANGE THIS to the user running code-server
Group=ubuntu # CHANGE THIS to the user's group
WorkingDirectory=/home/ubuntu # CHANGE THIS to the user's home or project directory

# Path to the code-server executable (adjust if installed differently)
ExecStart=/usr/bin/code-server --bind-addr 127.0.0.1:8080 --user-data-dir /home/ubuntu/.local/share/code-server --auth password

Restart=on-failure # Restart if it crashes
RestartSec=5

# Set the password via environment variable if desired
Environment="PASSWORD=YourStrongPasswordHere" # CHANGE THIS or remove if using config file password

# Increase file limits if needed
LimitNOFILE=65536

[Install]
WantedBy=multi-user.target
```

(Content adapted from 60)
Important: Replace ubuntu with the actual username intended to run Code-Server. Ensure the WorkingDirectory and --user-data-dir paths exist and are accessible by that user. Set a strong password either via the Environment="PASSWORD=..." line or in the ~/.config/code-server/config.yaml file (if using the config file, remove the Environment line here).

- **Managing systemd Services:**
  - After creating/editing a .service file: sudo systemctl daemon-reload.[41]
  - Enable service to start on boot: sudo systemctl enable <service-name>.service.[41]

- Disable service from starting on boot: sudo systemctl disable <service-name>.service.[102]
- Start service now: sudo systemctl start <service-name>.service.[41]
- Stop service now: sudo systemctl stop <service-name>.service.[95]
- Restart service: sudo systemctl restart <service-name>.service.[95]
- Check service status: sudo systemctl status <service-name>.service.[41]
- View service logs: journalctl -u <service-name>.service -f.[103]

5.2. Method 2: Using docker-compose

If Ollama and Code-Server are run as Docker containers managed by docker-compose, ensuring they restart automatically is simpler. The restart policy within the docker-compose.yml file controls this behavior.[39]

Available restart policies include [106]:

- no: Never restart the container (default).
- on-failure: Restart only if the container exits with a non-zero status code (indicating an error). Can optionally specify a maximum retry count (e.g., on-failure:5).
- always: Always restart the container if it stops for any reason (crash, normal exit). If manually stopped (e.g., docker stop), it will restart when the Docker daemon restarts.
- unless-stopped: Similar to always, but it will *not* restart the container if it was explicitly stopped by the user or another process (e.g., via docker stop or docker compose stop), even after the Docker daemon restarts.

Recommendation: Use restart: unless-stopped. This policy provides robustness against crashes and ensures services come back up after a server reboot, while still respecting intentional manual stops.[105] The always policy can sometimes be inconvenient, as it might restart a container that was deliberately stopped for maintenance or configuration changes when the Docker daemon is restarted.[107] on-failure is safer if concerned about restart loops but might not restart a container that exits cleanly but unexpectedly.

5.3. Example docker-compose.yml with Restart Policy

YAML

```yaml
# ~/ai-server/docker-compose.yml
version: '3.8'
services:
  ollama:
    image: ollama/ollama:latest
    container_name: ollama
    ports:
      - "127.0.0.1:11434:11434"
    volumes:
      -./ollama_data:/root/.ollama
    deploy:
      resources:
        reservations:
          devices:
            - driver: nvidia
              count: all
              capabilities: [gpu]
    restart: unless-stopped # <--- Auto-restart policy

  code-server:
    image: codercom/code-server:latest
    container_name: code-server
    user: "$(id -u):$(id -g)"
    ports:
      - "127.0.0.1:8080:8080"
    volumes:
      -./code-server_data:/home/coder/.local/share/code-server
      - $HOME:/home/coder/project
    environment:
      - PASSWORD=YourStrongPasswordHere # CHANGE THIS
    restart: unless-stopped # <--- Auto-restart policy

volumes:
  ollama_data:
  code-server_data:
```

(Adapted from Section 3.4)

With this configuration, running sudo docker compose up -d will start the containers, and the Docker daemon will automatically restart them if they crash or after the EC2 instance reboots, unless they were explicitly stopped using sudo docker compose stop.

6. Setting Up the Remote Development Environment (Code-Server)

Code-Server provides a way to run Visual Studio Code on a remote server and access it through a web browser, enabling a powerful, consistent development experience from any device.[91]

6.1. Installation

Several methods exist for installing Code-Server:

1. **Official Install Script (Recommended):** This is the easiest method for Linux, macOS, and FreeBSD. It attempts to use the system package manager or falls back to a standalone installation. It also typically sets up a systemd service automatically.[96]
   Bash
   ```
   curl -fsSL https://code-server.dev/install.sh | sh
   ```
2. **Package Managers:**
   - **Debian/Ubuntu (.deb):** Download the .deb package from GitHub releases and install with sudo dpkg -i <file>.deb or sudo apt install./<file>.deb.[96] Manual service setup might be needed.
   - **Fedora/CentOS/RHEL (.rpm):** Download the .rpm package and install with sudo dnf install <file>.rpm or sudo yum install <file>.rpm.[96] Manual service setup might be needed.
   - **Snap:** Install via the Snap Store: sudo snap install code-server.[113] Snaps handle updates automatically but run in a more confined environment.
   - **AUR (Arch Linux):** Use an AUR helper or build manually from the AUR package.[96]
3. **Standalone Release:** Download the release archive (.tar.gz) for your architecture from GitHub, unpack it, and run the executable directly (./bin/code-server).[96] Requires manual setup for running as a service and adding to PATH.
4. **Docker (Used in Setup Script):** Use the official Docker image codercom/code-server:latest as shown in the docker-compose.yml in Sections

3.4 and 5.3. This isolates dependencies and integrates well with the Ollama container setup.

The recommended approach for simplicity and integration with systemd (if not using Docker) is the official install script.[96] If using Docker for Ollama, running Code-Server in Docker as well (Method 4) provides consistency.

6.2. Configuration

Code-Server's configuration is primarily managed through command-line flags or a YAML configuration file located at ~/.config/code-server/config.yaml.[73] Key settings when running behind an Nginx reverse proxy:

- bind-addr: Set to 127.0.0.1:8080 to ensure Code-Server only listens on the local interface, accessible only via the Nginx proxy.[60] Do not use 0.0.0.0 if Nginx is handling external access and SSL.
- auth: Set to password (the default) for basic password authentication.[60] none can be used if authentication is handled externally (e.g., SSH tunnel, OAuth proxy), but password auth is recommended for direct exposure via Nginx.[73]
- password: Set a strong password here, or leave it commented out to use the password automatically generated by Code-Server on first run (it will be printed to the log or stored in the config file).[96] If using the Docker image, the password can be set via the PASSWORD environment variable.
- cert: Set to false since Nginx will handle SSL/TLS termination.[73] Code-Server can generate a self-signed certificate if cert: true is set, but using a trusted certificate via Nginx/Certbot is preferred.
- user-data-dir: Specifies where user settings, extensions, etc., are stored. Defaults typically work, but can be customized.[99] Ensure the directory is writable by the user running Code-Server.
- extensions-dir: Specifies where extensions are installed.[99]

Example ~/.config/code-server/config.yaml:

YAML

```
bind-addr: 127.0.0.1:8080
auth: password
```

```
# password: YourStrongPasswordHere # Set explicitly or let it auto-generate
cert: false
# user-data-dir: /path/to/user/data # Optional override
# extensions-dir: /path/to/extensions # Optional override
```

6.3. Running as a Service

To ensure Code-Server runs reliably and starts on boot:

- **If installed via official script:** The script usually creates and enables a systemd user service, typically named code-server@$USER.service. Manage it using systemctl --user commands or sudo systemctl commands targeting the specific user service (e.g., sudo systemctl enable --now code-server@ubuntu).[96]
- **If installed manually or via non-service packages:** Create a systemd service file (/etc/systemd/system/code-server.service) as detailed in Section 5.1, ensuring correct paths, user, and arguments (--bind-addr, --auth, etc.).[60] Then enable and start it using sudo systemctl enable --now code-server.service.
- **If using Docker Compose:** The restart: unless-stopped policy in the docker-compose.yml file (Section 5.3) handles automatic restarts. Manage the service using sudo docker compose up -d, sudo docker compose stop, etc.

6.4. Accessing via Nginx

Once Nginx is configured (Section 4) with the correct proxy_pass directive (pointing to http://127.0.0.1:8080) and WebSocket headers, and Code-Server is running and bound to 127.0.0.1:8080, access the environment by navigating to the configured domain (e.g., https://your_domain.com/ or https://your_domain.com/code/) in a web browser. Nginx will handle the SSL connection and securely proxy the traffic to the Code-Server instance. You will be prompted for the password configured in config.yaml or the PASSWORD environment variable.

7. Automated Cost Management (EC2 Auto-Shutdown)

A significant advantage of self-hosting on EC2 is the ability to control costs by stopping the instance when it's not in use. Since GPU instances are relatively expensive per hour, automating shutdown during off-peak times (e.g., overnight, weekends) can lead to substantial savings, especially when using On-Demand pricing or minimizing Spot instance usage hours.

7.1. Method 1: Instance-based Cron Job

This method involves running a scheduled command directly on the EC2 instance

itself.

- **Requirements:**
  - AWS CLI installed on the EC2 instance: sudo apt update && sudo apt install -y awscli.
  - IAM Role attached to the EC2 instance: This role needs an IAM policy granting permission to stop the specific instance. A minimal policy would look like:

    JSON

    ```
    {
       "Version": "2012-10-17",
       "Statement":,
            "Resource":
    "arn:aws:ec2:YOUR_REGION:YOUR_ACCOUNT_ID:instance/YOUR_INSTANCE_ID",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/Name": "MyAIServer" // Optional: condition on tag
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances" // Often useful for scripts to check state
            ],
            "Resource": "*" // DescribeInstances often needs broader resource scope
        }
    ]
    }
    ```

    Replace placeholders. Attaching this role allows the AWS CLI running on the instance to make authorized calls without needing explicit credentials stored on the machine.[56]
- **Setup:**
  1. Edit the crontab for the user (e.g., ubuntu): crontab -e.
  2. Add a line to schedule the stop command. For example, to stop the instance every day at 2:00 AM server time:

     Code snippet

     ```
     # Stop EC2 instance daily at 2:00 AM
     0 2 * * * /usr/bin/aws ec2 stop-instances --instance-ids YOUR_INSTANCE_ID
     --region YOUR_REGION >> /home/ubuntu/cron_shutdown.log 2>&1
     ```

     Replace YOUR_INSTANCE_ID and YOUR_REGION. The command redirects

output and errors to a log file.[116]

- **Pros:** Conceptually simple; uses standard Linux tools.
- **Cons:** Requires AWS CLI and IAM role setup *on* the instance, potentially increasing its attack surface. Relies on the instance's cron daemon being operational. Less aligned with infrastructure-as-code principles. Permissions granted to the instance might be broader than strictly necessary for its primary function.

7.2. Method 2: AWS Lambda and EventBridge Scheduler (Recommended)

This approach uses AWS-native serverless components, providing a more robust and secure solution.[120]

- **Components:**
  - **AWS Lambda Function:** A small piece of code (e.g., Python using the boto3 SDK) that executes the ec2:StopInstances API call.[120]
  - **IAM Role for Lambda:** A dedicated execution role for the Lambda function, granting only the necessary permissions (ec2:StopInstances for the specific instance ARN, plus basic CloudWatch Logs permissions: logs:CreateLogGroup, logs:CreateLogStream, logs:PutLogEvents).[120] This adheres to the principle of least privilege.
  - **Amazon EventBridge Scheduler (or CloudWatch Events Rule):** A scheduler that triggers the Lambda function based on a defined schedule (e.g., a cron expression).[120]
- **Setup:**
  1. **Create IAM Policy:** Define a policy granting ec2:StopInstances on the target instance ARN and CloudWatch Logs permissions.
  2. **Create IAM Role:** Create a role for Lambda execution and attach the policy created above.[120]
  3. **Create Lambda Function:**
     - Go to the Lambda console, choose "Create function".
     - Select "Author from scratch".
     - Choose a name (e.g., StopMyAIServerInstance).
     - Select a runtime (e.g., Python 3.9 or later).
     - Choose the IAM role created in step 2.
     - Paste the Python code (see example below) into the code editor.
     - Adjust the function timeout (e.g., 10-30 seconds) under Configuration > General configuration.[120]
     - Deploy the function.
  4. **Create EventBridge Schedule:**
     - Go to the EventBridge console, choose "Schedules", "Create schedule".

        - Enter a name (e.g., StopAIServerDaily).
        - Define the schedule pattern. Use a cron expression for specific times. Example for 2:00 AM UTC daily: cron(0 2? * * *).[120] Remember cron expressions are in UTC, adjust accordingly for the desired local time.
        - Select the target: Choose "AWS Lambda" and select the StopMyAIServerInstance function created earlier.[120]
        - Configure other settings as needed (e.g., retry policy) and create the schedule.
- **Pros:** More secure (permissions are tied to the Lambda role, not the EC2 instance), more robust and reliable (uses managed AWS services), better aligns with infrastructure-as-code and serverless patterns, decoupled from the instance's state.
- **Cons:** Slightly more complex initial setup involving multiple AWS services.

The Lambda/EventBridge method is recommended due to its enhanced security and reliability. Granting permissions directly to an EC2 instance for self-management tasks should generally be avoided if a more decoupled, least-privilege approach like Lambda is feasible.

7.3. Python Lambda Function Code Example (Stop Instance)

Python

```
# lambda_function.py
import boto3
import os

# Read environment variables or define directly
REGION = os.environ.get('AWS_REGION', 'us-east-1') # Or replace with your specific region
INSTANCE_IDS = ['i-xxxxxxxxxxxxxxxxx'] # REPLACE with your actual EC2 instance ID(s)

ec2 = boto3.client('ec2', region_name=REGION)

def lambda_handler(event, context):
    """
    Stops the specified EC2 instances.
    """
```

```
    if not INSTANCE_IDS:
        print("No instance IDs provided. Exiting.")
        return {
            'statusCode': 400,
            'body': 'No instance IDs specified'
        }

    try:
        print(f"Attempting to stop instances: {INSTANCE_IDS} in region {REGION}")
        ec2.stop_instances(InstanceIds=INSTANCE_IDS)
        print(f"Successfully sent stop command for instances: {INSTANCE_IDS}")
        return {
            'statusCode': 200,
            'body': f'Successfully initiated stop for instances: {INSTANCE_IDS}'
        }
    except Exception as e:
        print(f"Error stopping instances: {e}")
        return {
            'statusCode': 500,
            'body': f'Error stopping instances: {str(e)}'
        }
```

(Adapted from [120])

This code initializes the boto3 EC2 client for the specified region and calls the stop_instances API for the instance IDs listed in the INSTANCE_IDS variable. Remember to replace the placeholder instance ID and potentially the region. Error handling and logging are included for basic diagnostics via CloudWatch Logs. A similar function can be created for starting instances by replacing ec2.stop_instances with ec2.start_instances.

8. Monthly Cost Estimation

Estimating the monthly cost involves considering the primary cost drivers: EC2 instance runtime, EBS storage, and data transfer.

8.1. Cost Components

1. **EC2 Instance Cost:** This is typically the largest cost factor. It depends on:

    - Instance Type and Size (e.g., g5.xlarge, g5.12xlarge).[19]
    - Pricing Model (On-Demand or Spot).[22] Spot instances offer significant savings but can be interrupted.[22]
    - Runtime Hours: The number of hours the instance is in a running state per month. The auto-shutdown mechanism directly impacts this.
2. **EBS Storage Cost:** Cost associated with the Elastic Block Store volumes attached to the instance (root volume and any additional data volumes). It depends on:
    - Volume Type: gp3 (General Purpose SSD) is recommended for a balance of cost and performance. It's priced based on provisioned storage, plus charges for IOPS and throughput provisioned above the free baseline.[130] gp2 is simpler (priced only on size) but generally more expensive than baseline gp3.[130]
    - Provisioned Size (GB): Charged per GB-month.[130] A starting size of 100-250GB is reasonable for the OS and some models, but larger sizes (e.g., 500GB+) might be needed depending on the number and size of models downloaded.[19]
    - Provisioned IOPS/Throughput (for gp3): Costs apply if provisioned beyond the free tier (3,000 IOPS, 125 MB/s).[130] Baseline performance is often sufficient for this use case.
3. **Data Transfer Cost:** Charges for data moving in and out of AWS.
    - **Data Transfer IN (Internet to EC2):** Free.[134]
    - **Data Transfer OUT (EC2 to Internet):** AWS provides a free tier of 100GB per month aggregated across all services and regions (except China/GovCloud). Beyond that, costs are tiered, starting around $0.09/GB in regions like us-east-1.[134] For personal development use, exceeding 100GB/month is unlikely unless transferring very large files or models frequently out of the instance.
    - **Intra-Region Data Transfer:** Transferring data between Availability Zones within the same region typically costs $0.01/GB in each direction.[136] This is generally not applicable to a single-instance setup unless interacting heavily with other AWS services (like S3 via Gateway Endpoints, RDS, etc.) located in different AZs.
4. **Other Potential Costs (Usually Minor):**
    - **EBS Snapshots:** If automated backups are configured, snapshot storage incurs costs (Standard: ~$0.05/GB-month for changed data).[130]
    - **Lambda/EventBridge:** The free tier for both services is generous. A simple daily stop/start schedule is highly unlikely to incur charges.[120]
    - **Elastic IP Address:** A small hourly charge applies if an Elastic IP is allocated but *not* associated with a running instance. Ensure it's associated or released

when the instance is stopped for long periods (though stopping/starting via AWS typically handles this).

8.2. Usage Assumptions for Estimates

To provide concrete estimates, the following assumptions are made:

- **Region:** us-east-1 (N. Virginia) - A major region often with competitive pricing.
- **Operating System:** Linux.
- **Running Hours:**
  - **Scheduled Shutdown:** Assumes usage during typical work/development hours (e.g., 10 hours/day) on weekdays (e.g., 22 days/month), totaling **220 hours/month**. This scenario heavily utilizes the auto-shutdown feature.
  - **24/7 Operation:** Assumes the instance runs continuously, totaling **720 hours/month** (24 hours * 30 days). This serves as a comparison baseline.
- **EBS Volume: 250GB gp3** volume. Assumes baseline IOPS (3,000 free) and throughput (125 MB/s free) are sufficient, incurring only the storage cost (~$0.08/GB-month in us-east-1).[130] Total EBS cost ≈ 250 GB * $0.08/GB-month = **$20/month**.
- **Data Transfer Out:** Assumed to be **less than 100GB/month**, therefore **$0** cost.[138]

8.3. Table: Monthly Cost Estimate Sheet (us-east-1, Linux)

The table below estimates monthly costs for various recommended instances and usage scenarios based on the assumptions above. Spot prices are estimated at approximately 30% of On-Demand rates (a 70% discount), but **actual Spot prices fluctuate and can vary significantly**.

Cost Component	g4dn.xlarge (Spot, 220hr)	g4dn.xlarge (OD, 220hr)	g5.xlarge (Spot, 220hr)	g5.xlarge (OD, 220hr)	g5.12xlarge (Spot, 220hr)	g5.xlarge (Spot, 720hr)	g5.xlarge (OD, 720hr)
EC2 Instance Hourly Rate	~$0.16	$0.526	**~$0.30**	$1.006	**~$1.70**	~$0.30	$1.006
EC2 Instance Monthly Cost	~$35.20	$115.72	**~$66.00**	$221.32	**~$374.00**	~$216.00	$724.32

EBS Storage (250GB gp3)	$20.00	$20.00	**$20.00**	$20.00	**$20.00**	$20.00	$20.00
Data Transfer Out	$0.00	$0.00	**$0.00**	$0.00	**$0.00**	$0.00	$0.00
Total Est. Monthly	**~$55.20**	**$135.72**	**~$86.00**	**$241.32**	**~$394.00**	**~$236.00**	**$744.32**

OD = On-Demand. Spot prices are estimates and vary. Costs are approximate and exclude taxes. Based on pricing from [19] for us-east-1.

This table clearly illustrates the significant cost savings achievable through:

1. **Spot Instances:** Reducing hourly instance costs by potentially 70% or more compared to On-Demand.
2. **Auto-Shutdown:** Drastically cutting down the number of billable hours per month (e.g., 220 vs. 720 hours).

Combining Spot instances with the scheduled shutdown offers the most cost-effective approach for this personal development server use case. Even the powerful g5.12xlarge, suitable for 70B models, becomes considerably more affordable under this model compared to running a smaller instance 24/7 on-demand.

9. Conclusion and Checklist

This report has detailed the steps and considerations for setting up a self-hosted AI development environment on AWS EC2, focusing on balancing performance, convenience, and cost-effectiveness.

The primary recommendations are:

- **Instance Selection:** Prioritize **AWS EC2 G5 instances** (e.g., g5.xlarge for smaller models, g5.12xlarge for comfortably running quantized 70B models) due to their superior ML inference price/performance compared to G4dn. Utilize **Spot Instances** combined with automated shutdown for significant cost savings.
- **AI Backend:** Start with **Ollama** for its ease of use, robust API, and wide model support. Deploy it using **Docker** for clean dependency management.

- **Environment Setup:** Leverage **AWS Deep Learning AMIs (Ubuntu-based)** to simplify NVIDIA driver and CUDA toolkit installation. Install the **NVIDIA Container Toolkit** to enable GPU access for Docker containers. An outline for an automated setup script is provided.
- **Secure Access:** Use **Nginx** as a reverse proxy to provide a single, secure access point. Configure it to proxy requests to Ollama and Code-Server (ensuring WebSocket support for the latter). Implement essential **security headers** and secure the domain with **SSL/TLS certificates** obtained via Let's Encrypt (Certbot).
- **Remote Development:** Install and configure **Code-Server**, accessing it securely through the Nginx proxy.
- **Reliability:** Ensure services auto-start after reboots using Docker Compose's restart: unless-stopped policy or systemd service files if not using containers.
- **Cost Management:** Implement an **automated daily shutdown** during off-peak hours using the recommended **AWS Lambda and EventBridge Scheduler** method for optimal security and reliability.

This configuration provides a powerful, customizable AI development platform that runs locally on cloud hardware, bypassing SaaS fees while retaining convenient access. It allows experimentation with various LLMs, including large models like Llama 3 70B (on appropriate instances), within a controlled and cost-optimized environment.

Potential next steps could include exploring different quantization techniques further, setting up monitoring for the instance and services, experimenting with model fine-tuning (which would require significantly more resources), or integrating LocalGPT for specialized document interaction tasks.

9.1. Checklist Confirmation

The following checklist confirms that all requirements specified in the initial query have been addressed within this report:

Goal	Addressed?	Section(s) Addressed
Best EC2 instance suggestion	✅	2
Full auto setup script	✅ (Outline)	3
Nginx proxy config	✅ (Example)	4

Auto-start services	✅	5
Code-Server setup	✅	6
Auto-shutdown script	✅ (Lambda)	7
Cost estimates sheet	✅	8

Works cited

1. How to Install Llama-3.3 70B Instruct Locally? - NodeShift, accessed April 28, 2025, https://nodeshift.com/blog/how-to-install-llama-3-3-70b-instruct-locally
2. How much GPU VRAM do I need to serve LLama3 70B? - A Simple Solution | Kaggle, accessed April 28, 2025, https://www.kaggle.com/discussions/general/529079
3. Llama-3.3-70B-Instruct - Hacker News, accessed April 28, 2025, https://news.ycombinator.com/item?id=42341388
4. Llama 3 70b instruct works surprisingly well on 24gb VRAM cards : r/LocalLLaMA - Reddit, accessed April 28, 2025, https://www.reddit.com/r/LocalLLaMA/comments/1cj4det/llama_3_70b_instruct_works_surprisingly_well_on/
5. Mixtral-8x7B- do I need 100GB of Vram? : r/LocalLLaMA - Reddit, accessed April 28, 2025, https://www.reddit.com/r/LocalLLaMA/comments/1bwau9s/mixtral8x7b_do_i_need_100gb_of_vram/
6. GPU Requirement Guide for Llama 3 (All Variants) - ApX Machine Learning, accessed April 28, 2025, https://apxml.com/posts/ultimate-system-requirements-llama-3-models
7. Faster Mixtral inference with TensorRT-LLM and quantization | Baseten Blog, accessed April 28, 2025, https://www.baseten.co/blog/faster-mixtral-inference-with-tensorrt-llm-and-quantization/
8. Nvidia-SMI for Mixtral-8x7B-Instruct-v0.1 in case anyone wonders how much VRAM it sucks up (90636MiB) so you need 91GB of RAM - Reddit, accessed April 28, 2025, https://www.reddit.com/r/LocalLLaMA/comments/18pm1m7/nvidiasmi_for_mixtral8x7binstructv01_in_case/
9. Gemma 3 12b uses 24 GB VRAM ??? | Flash Attention | KV Cache Quantization ·

Issue #9730 - GitHub, accessed April 28, 2025, https://github.com/ollama/ollama/issues/9730
10. How can you tell how much memory a model will take, `size` in `ollama ps` vs `ollama list` - Reddit, accessed April 28, 2025, https://www.reddit.com/r/ollama/comments/1hsh75q/how_can_you_tell_how_much_memory_a_model_will/
11. Hardware requirements to run Llama 3 70b on a home server : r/LocalLLaMA - Reddit, accessed April 28, 2025, https://www.reddit.com/r/LocalLLaMA/comments/1eiwnqe/hardware_requirements_to_run_llama_3_70b_on_a/
12. nvidia/Llama-3.1-Nemotron-70B-Instruct-HF · Suitable hardware config for usage this model, accessed April 28, 2025, https://huggingface.co/nvidia/Llama-3.1-Nemotron-70B-Instruct-HF/discussions/22
13. Deep Dive into Mixtral 8x7B - GitHub, accessed April 28, 2025, https://github.com/neobundy/Deep-Dive-Into-AI-With-MLX-PyTorch/blob/master/deep-dives/002-mixtral-8x7b/README.md
14. mistralai/Mixtral-8x7B-Instruct-v0.1 · Min hardware requirements - Hugging Face, accessed April 28, 2025, https://huggingface.co/mistralai/Mixtral-8x7B-Instruct-v0.1/discussions/3
15. Exploring the AI Chatbot Capabilities of the OS LLM Mixtral 8x7b on a 24GB GPU - Liip, accessed April 28, 2025, https://www.liip.ch/en/blog/exploring-the-ai-chatbot-capabilities-of-the-oss-llm-mixtral-8x7b-on-a-24gb-gpu
16. Is there a chart that's shows for xx Billion parameter model, this hardware is needed? : r/ollama - Reddit, accessed April 28, 2025, https://www.reddit.com/r/ollama/comments/1hbleji/is_there_a_chart_thats_shows_for_xx_billion/
17. Llama 3.2 Vision · Ollama Blog, accessed April 28, 2025, https://ollama.com/blog/llama3.2-vision
18. Getting started with LLM in the Cloud with Amazon DLAMI EC2 Instances - Chariot Solutions, accessed April 28, 2025, https://chariotsolutions.com/blog/post/getting-started-with-llm-in-the-cloud-with-amazon-dlami-ec2-instances/
19. Amazon EC2 G4 Instances, accessed April 28, 2025, https://aws.amazon.com/ec2/instance-types/g4/
20. Deploy LLMs in AWS GovCloud (US) Regions using Hugging Face Inference Containers, accessed April 28, 2025, https://aws.amazon.com/blogs/publicsector/deploy-llms-in-aws-govcloud-us-reg

ions-using-hugging-face-inference-containers/
21. Amazon EC2 G5 Instances | Amazon Web Services, accessed April 28, 2025, https://aws.amazon.com/ec2/instance-types/g5/
22. AWS G4 vs G5 Family: A Detailed Comparison of AWS GPU Instances - CloudOptimo, accessed April 28, 2025, https://www.cloudoptimo.com/blog/aws-g4-vs-g5-family-a-detailed-comparison-of-aws-gpu-instances/
23. Achieve four times higher ML inference throughput at three times lower cost per inference with Amazon EC2 G5 instances for NLP and CV PyTorch models | AWS Machine Learning Blog, accessed April 28, 2025, https://aws.amazon.com/blogs/machine-learning/achieve-four-times-higher-ml-inference-throughput-at-three-times-lower-cost-per-inference-with-amazon-ec2-g5-instances-for-nlp-and-cv-pytorch-models/
24. g4dn.xlarge Pricing and Specs: AWS EC2, accessed April 28, 2025, https://costcalc.cloudoptimo.com/aws-pricing-calculator/ec2/g4dn.xlarge
25. [D] Which EC2 instance types do you use for training neural nets? : r/MachineLearning, accessed April 28, 2025, https://www.reddit.com/r/MachineLearning/comments/11f0zs6/d_which_ec2_instance_types_do_you_use_for/
26. Announcing Databricks Support for Amazon EC2 G6 Instances, accessed April 28, 2025, https://www.databricks.com/blog/aws-ec2-g6
27. g5.xlarge Pricing and Specs: AWS EC2, accessed April 28, 2025, https://costcalc.cloudoptimo.com/aws-pricing-calculator/ec2/g5.xlarge
28. g5.2xlarge pricing and specs - Amazon EC2 Instance Comparison - Vantage, accessed April 28, 2025, https://instances.vantage.sh/aws/ec2/g5.2xlarge
29. g5.24xlarge specs and pricing | AWS - CloudPrice, accessed April 28, 2025, https://cloudprice.net/aws/ec2/instances/g5.24xlarge
30. g4dn.xlarge specs and pricing | AWS - CloudPrice, accessed April 28, 2025, https://cloudprice.net/aws/ec2/instances/g4dn.xlarge
31. g4dn.xlarge pricing and specs - Amazon EC2 Instance Comparison - Vantage, accessed April 28, 2025, https://instances.vantage.sh/aws/ec2/g4dn.xlarge
32. Benchmarking Inexpensive AWS Instances : r/LocalLLaMA - Reddit, accessed April 28, 2025, https://www.reddit.com/r/LocalLLaMA/comments/1dclmwt/benchmarking_inexpensive_aws_instances/
33. EC2 Reserved Instance Pricing – Amazon Web Services, accessed April 28, 2025, https://aws.amazon.com/ec2/pricing/reserved-instances/pricing/
34. AWS Marketplace: Ollama with Open WebUI on Ubuntu 24.04, accessed April 28, 2025, https://aws.amazon.com/marketplace/pp/prodview-g36vovk7627cu

35. Run Large Language Models with Ollama and AWS Lightsail for Research, accessed April 28, 2025, https://community.aws/content/2iCmNKQEumqikPPBjA5SEO8Bhzb/run-large-language-models-with-ollama-and-lightsail-for-research
36. Ollama Port Number Explained | Restackio, accessed April 28, 2025, https://www.restack.io/p/ollama-answer-port-number-cat-ai
37. ollama/docs/faq.md at main - GitHub, accessed April 28, 2025, https://github.com/ollama/ollama/blob/main/docs/faq.md
38. How to Deploy Ollama Server on Amazon EC2 with GPU in 10 Minutes | Ligang Yan, accessed April 28, 2025, https://www.ligangyan.com/blog/2024-03-02-how-to-deploy-ollama-server-on-amazon-ec2-with-gpu-in-10-min-zh
39. Setting Up Ollama with Open-WebUI: A Docker Compose Guide - Archy.net, accessed April 28, 2025, https://www.archy.net/setting-up-ollama-with-open-webui-a-docker-compose-guide/
40. Deploy ANY Open-Source LLM with Ollama on an AWS EC2 + GPU in 10 Min (Llama-3.1, Gemma-2 etc.) - YouTube, accessed April 28, 2025, https://m.youtube.com/watch?v=SAhUc9ywIiw
41. ollama/docs/linux.md at main - GitHub, accessed April 28, 2025, https://github.com/ollama/ollama/blob/main/docs/linux.md
42. Ollama Ubuntu GPU Setup Guide | Restackio, accessed April 28, 2025, https://www.restack.io/p/ollama-answer-ubuntu-gpu-setup-cat-ai
43. Download Ollama on Linux, accessed April 28, 2025, https://ollama.com/download
44. Install and run ollama with docker guide - BytePlus, accessed April 28, 2025, https://www.byteplus.com/en/topic/556147
45. Setting Up Ollama With Docker [With NVIDIA GPU] - It's FOSS, accessed April 28, 2025, https://itsfoss.com/ollama-docker/
46. ollama/docs/docker.md at main - GitHub, accessed April 28, 2025, https://github.com/ollama/ollama/blob/main/docs/docker.md
47. PrivateGPT on AWS: Cloud, Secure, Private, Chat with My Docs. - Ron Amosa, accessed April 28, 2025, https://ronamosa.io/docs/engineer/AI/PrivateGPTAWS/
48. PromtEngineer/localGPT: Chat with your documents on your local device using GPT models. No data leaves your device and 100% private. - GitHub, accessed April 28, 2025, https://github.com/PromtEngineer/localGPT
49. LocalGPT on Windows: A Guide to Private AI Implementation - Amilma Digital, accessed April 28, 2025, https://amilma.digital/public/blog/localgpt-on-windows-a-guide-to-private-ai-implementation

50. README.md - PromtEngineer/localGPT - GitHub, accessed April 28, 2025, https://github.com/PromtEngineer/localGPT/blob/main/README.md
51. Local Intelligence: How to set up a local GPT Chat for secure & private document analysis workflow - DEV Community, accessed April 28, 2025, https://dev.to/avatsaev/local-intelligence-how-to-set-up-a-local-gpt-chat-for-secure-private-document-analysis-workflow-1lnm
52. Introducing LocalGPT: Offline ChatBOT for your FILES with GPU - Vicuna : r/LocalLLaMA, accessed April 28, 2025, https://www.reddit.com/r/LocalLLaMA/comments/13vhev0/introducing_localgpt_offline_chatbot_for_your/
53. Can't make llama-cpp-python run with GPU on an AWS EC2 instance! · Issue #856 - GitHub, accessed April 28, 2025, https://github.com/abetlen/llama-cpp-python/issues/856
54. localGPT/Dockerfile at main - GitHub, accessed April 28, 2025, https://github.com/PromtEngineer/localGPT/blob/main/Dockerfile
55. mattporritt/aws_ollama: Set up a test EC2 instane running Ollama in AWS with a publicly available endpoint - GitHub, accessed April 28, 2025, https://github.com/mattporritt/aws_ollama
56. How to Stop/Start an AWS EC2 instance using AWS CLI command line, accessed April 28, 2025, https://www.laurencegellert.com/2023/02/how-to-stop-start-an-aws-ec2-instance-using-aws-cli-command-line/
57. Use NGINX as a Reverse Proxy | Coder Docs, accessed April 28, 2025, https://coder.com/docs/tutorials/reverse-proxy-nginx
58. How To Secure Nginx with Let's Encrypt on Ubuntu - DigitalOcean, accessed April 28, 2025, https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-22-04
59. Install NVIDIA drivers on a GPU-enabled EC2 instance - Ubuntu on AWS documentation, accessed April 28, 2025, https://documentation.ubuntu.com/aws/en/latest/aws-how-to/instances/install-nvidia-drivers/
60. How To Set Up the code-server Cloud IDE Platform on Ubuntu 22.04 - DigitalOcean, accessed April 28, 2025, https://www.digitalocean.com/community/tutorials/how-to-set-up-the-code-server-cloud-ide-platform-on-ubuntu-22-04
61. How to install Nvidia CUDA driver on AWS ec2 instance? - Ask Ubuntu, accessed April 28, 2025, https://askubuntu.com/questions/1397934/how-to-install-nvidia-cuda-driver-on-

aws-ec2-instance
62. Installing Nvidia drivers on a GPU-enabled (Nvidia Tesla) EC2 instance (G4DN) - Tutorials, accessed April 28, 2025, https://discourse.ubuntu.com/t/installing-nvidia-drivers-on-a-gpu-enabled-nvidia-tesla-ec2-instance-g4dn/40610
63. How to Install CUDA on Ubuntu 22.04 | Step-by-Step | Cherry Servers, accessed April 28, 2025, https://www.cherryservers.com/blog/install-cuda-ubuntu
64. How do I install NVIDIA GPU driver, CUDA Toolkit, NVIDIA Container Toolkit on Amazon EC2 instances running Ubuntu Linux? | AWS re:Post, accessed April 28, 2025, https://repost.aws/articles/ARWGxLArMBQ4y1MKoSHTq3gQ/how-do-i-install-nvidia-gpu-driver-cuda-toolkit-nvidia-container-toolkit-on-amazon-ec2-instances-running-ubuntu-linux
65. How can I install CUDA 12.1 on Ubuntu 22.04. I'm going insane here, accessed April 28, 2025, https://forums.developer.nvidia.com/t/how-can-i-install-cuda-12-1-on-ubuntu-22-04-im-going-insane-here/293738
66. A Comprehensive Guide to Ollama Local Installation - Collabnix, accessed April 28, 2025, https://collabnix.com/a-comprehensive-guide-to-ollama-local-installation/
67. NGINX Reverse Proxy | NGINX Documentation, accessed April 28, 2025, https://docs.nginx.com/nginx/admin-guide/web-server/reverse-proxy/
68. How to setup an Nginx reverse proxy server example, accessed April 28, 2025, https://www.theserverside.com/blog/Coffee-Talk-Java-News-Stories-and-Opinions/How-to-setup-Nginx-reverse-proxy-servers-by-example
69. Nginx Reverse Proxy and CODE-Server on two different machines - Collabora Online, accessed April 28, 2025, https://forum.collaboraonline.com/t/nginx-reverse-proxy-and-code-server-on-two-different-machines/1090
70. Your Ollama Servers Are So Open, Even My Grandma Could Use Them! - Reddit, accessed April 28, 2025, https://www.reddit.com/r/ollama/comments/1guwg0w/your_ollama_servers_are_so_open_even_my_grandma/
71. How To Configure Nginx as a Reverse Proxy on Ubuntu 22.04 - DigitalOcean, accessed April 28, 2025, https://www.digitalocean.com/community/tutorials/how-to-configure-nginx-as-a-reverse-proxy-on-ubuntu-22-04
72. Trying to connect to ollama API port 11434 from remote workstation - Reddit, accessed April 28, 2025,

https://www.reddit.com/r/ollama/comments/1g7zz69/trying_to_connect_to_ollama_api_port_11434_from/
73. Securely Access & Expose code-server - Coder, accessed April 28, 2025, https://coder.com/docs/code-server/guide
74. Code-Server behind Nginx reverse proxy : r/codeserver - Reddit, accessed April 28, 2025, https://www.reddit.com/r/codeserver/comments/fx12q6/codeserver_behind_nginx_reverse_proxy/
75. Running Code server on subdomain with nginx reverse-proxy, accessed April 28, 2025, https://serverfault.com/questions/962509/running-code-server-on-subdomain-with-nginx-reverse-proxy
76. How to Proxy Multiple Code-Servers via Nginx? #7036 - GitHub, accessed April 28, 2025, https://github.com/coder/code-server/discussions/7036
77. Use code server behind nginx #5125 - GitHub, accessed April 28, 2025, https://github.com/coder/code-server/discussions/5125
78. Nginx Security Hardening Guide - SecOps® Solution, accessed April 28, 2025, https://www.secopsolution.com/blog/nginx-security-hardening-guide
79. Enhancing Website Security: Implementing Effective HTTP Security Headers in Nginx - LoadForge Guides, accessed April 28, 2025, https://loadforge.com/guides/enhancing-website-security-implementing-effective-security-headers-in-nginx
80. How to Configure Security Headers in Nginx - LinuxCapable, accessed April 28, 2025, https://linuxcapable.com/how-to-configure-security-headers-in-nginx/
81. Configure Security Headers in Nginx and Apache - Webdock.io, accessed April 28, 2025, https://webdock.io/en/docs/how-guides/security-guides/how-to-configure-security-headers-in-nginx-and-apache
82. An Overview of Best Practices for Security Headers | Okta Developer, accessed April 28, 2025, https://developer.okta.com/blog/2021/10/18/security-headers-best-practices
83. Best nginx configuration for improved security(and performance) - GitHub Gist, accessed April 28, 2025, https://gist.github.com/plentz/6737338
84. Nginx Security Hardening for Running WordPress in 2024 - SpinupWP, accessed April 28, 2025, https://spinupwp.com/hosting-wordpress-yourself-nginx-security-tweaks-woocommerce-caching-auto-server-updates/
85. NGINX Security Headers, the right way - GetPageSpeed, accessed April 28, 2025, https://www.getpagespeed.com/server-setup/nginx-security-headers-the-right-

way
86. HTTP Headers - OWASP Cheat Sheet Series, accessed April 28, 2025, https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Headers_Cheat_Sheet.html
87. How to Set Security Headers in Nginx Conf - Stack Overflow, accessed April 28, 2025, https://stackoverflow.com/questions/63500506/how-to-set-security-headers-in-nginx-conf
88. How To Use Certbot Standalone Mode to Retrieve Let's Encrypt SSL Certificates on Ubuntu 20.04 | DigitalOcean, accessed April 28, 2025, https://www.digitalocean.com/community/tutorials/how-to-use-certbot-standalone-mode-to-retrieve-let-s-encrypt-ssl-certificates-on-ubuntu-20-04
89. How To Secure Nginx with Let's Encrypt on Ubuntu 20.04 - DigitalOcean, accessed April 28, 2025, https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-20-04
90. How To Secure Nginx with Let's Encrypt on Ubuntu 18.04 | DigitalOcean, accessed April 28, 2025, https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-18-04
91. How To Install Code-server On Ubuntu 18.04 - UpCloud, accessed April 28, 2025, https://upcloud.com/resources/tutorials/install-code-server-ubuntu-18-04
92. How to Install Let's Encrypt Nginx plugin (DigitalOcean) - Easiest IT tutorial Website, accessed April 28, 2025, https://easiestsoft.com/how-to/it/letsencrypt-nginx-plugin.html
93. Running your Node.js app with systemd, accessed April 28, 2025, https://assets.ctfassets.net/hspc7zpa5cvq/5geZKiU4GFWgxtVLvxKqnB/e8c8f4ddc7d44e751dd8de8614007715/running-nodejs-apps-systemd.pdf
94. Running Node.js on Linux with systemd - CloudBees, accessed April 28, 2025, https://www.cloudbees.com/blog/running-node-js-linux-systemd
95. Running Your Node.js App With Systemd - Part 1 - NodeSource, accessed April 28, 2025, https://nodesource.com/blog/running-your-node-js-app-with-systemd-part-1
96. Install code-server: OS Instructions for VS Code - Coder, accessed April 28, 2025, https://coder.com/docs/code-server/install
97. Install Code-Server on Ubuntu 18.04 LTS - Vultr Docs, accessed April 28, 2025, https://docs.vultr.com/install-code-server-on-a-ubuntu-18-04-lts-vps
98. Running code-server as systemd service on Ubuntu #3652 - GitHub, accessed April 28, 2025, https://github.com/coder/code-server/discussions/3652

99. Setup a Custom Systemd Service for code-server on Ubuntu 18.0.4 · Issue #758 - GitHub, accessed April 28, 2025, https://github.com/cdr/code-server/issues/758
100. VS code-server starting service issue - Stack Overflow, accessed April 28, 2025, https://stackoverflow.com/questions/76267352/vs-code-server-starting-service-issue
101. Create a systemd service script for running Gunicorn to serve your application, accessed April 28, 2025, https://dev.to/tkirwa/create-a-systemd-service-script-for-running-gunicorn-to-serve-your-application-5aea
102. Deploy a Node.js Application as a Systemd Service - YouTube, accessed April 28, 2025, https://www.youtube.com/watch?v=vCcNu2A2bhQ
103. Run a node.js app with systemd - DEV Community, accessed April 28, 2025, https://dev.to/angeloscle/run-nodejs-app-with-systemd-3g6b
104. Running Node.Js as a systemd service | r0b blog, accessed April 28, 2025, https://blog.r0b.io/post/running-node-js-as-a-systemd-service/
105. Docker Compose Restart Policies | Baeldung on Ops, accessed April 28, 2025, https://www.baeldung.com/ops/docker-compose-restart-policies
106. www.digitalocean.com, accessed April 28, 2025, https://www.digitalocean.com/community/questions/how-to-set-up-docker-container-auto-restart-on-failure-in-a-digitalocean-droplet#:~:text=on%2Dfailure%20%3A%20Restarts%20the%20container,unless%20you%20manually%20stop%20it.
107. Start containers automatically - Docker Docs, accessed April 28, 2025, https://docs.docker.com/engine/containers/start-containers-automatically/
108. How to Set Up Docker Container Auto-Restart on Failure in a DigitalOcean Droplet?, accessed April 28, 2025, https://www.digitalocean.com/community/questions/how-to-set-up-docker-container-auto-restart-on-failure-in-a-digitalocean-droplet
109. Difference in docker restart policy between on-failure and unless-stopped? - Stack Overflow, accessed April 28, 2025, https://stackoverflow.com/questions/61725195/difference-in-docker-restart-policy-between-on-failure-and-unless-stopped
110. Could do with some help troubleshooting restart policies : r/docker - Reddit, accessed April 28, 2025, https://www.reddit.com/r/docker/comments/1auwkbq/could_do_with_some_help_troubleshooting_restart/
111. Docker compose "restart: always" - Reddit, accessed April 28, 2025, https://www.reddit.com/r/docker/comments/c7xvzy/docker_compose_restart_alw

ays/
112. code-server Docs: Run VS Code Anywhere - Coder, accessed April 28, 2025, https://coder.com/docs/code-server
113. Visual Studio Code on Linux, accessed April 28, 2025, https://code.visualstudio.com/docs/setup/linux
114. Install code-server on Ubuntu using the Snap Store - Snapcraft, accessed April 28, 2025, https://snapcraft.io/install/code-server/ubuntu
115. How to run as user "systemctl --user enable --now code-server" : r/codeserver - Reddit, accessed April 28, 2025, https://www.reddit.com/r/codeserver/comments/hahmsd/how_to_run_as_user_systemctl_user_enable_now/
116. To stop the EC2 instance after the execution of a script - Stack Overflow, accessed April 28, 2025, https://stackoverflow.com/questions/7610557/to-stop-the-ec2-instance-after-the-execution-of-a-script
117. EC2 Start and Stop Policy for user | AWS re:Post, accessed April 28, 2025, https://repost.aws/questions/QU2cH6k9vhR82GOMEYYmG5pw/ec2-start-and-stop-policy-for-user
118. Project advice needed - AWS CLI and Lambda : r/aws - Reddit, accessed April 28, 2025, https://www.reddit.com/r/aws/comments/1gpp2l9/project_advice_needed_aws_cli_and_lambda/
119. How to stop ec2 with crontab at @reboot? - Stack Overflow, accessed April 28, 2025, https://stackoverflow.com/questions/71591751/how-to-stop-ec2-with-crontab-at-reboot
120. Stop and start EC2 instances at intervals with a Lambda function | AWS re:Post, accessed April 28, 2025, https://repost.aws/knowledge-center/start-stop-lambda-eventbridge
121. Automate EC2 Instances with Lambda - AWS, accessed April 28, 2025, https://aws.amazon.com/awstv/watch/ca0bbcb2333/
122. Automatically Stopping an EC2 Instance Using AWS Lambda and EventBridge, accessed April 28, 2025, https://community.aws/content/2oLgxvS7uReYerwBAYkKn8ii0gf/automatically-stopping-an-ec2-instance-using-aws-lambda-and-eventbridge
123. Tutorial: Create an EventBridge scheduled rule for AWS Lambda functions, accessed April 28, 2025, https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-run-lambda-schedule.html

124. AWS Serverless: How to Stop EC2 using Event Bridge and Lambda, accessed April 28, 2025, https://community.aws/content/2muS34cXUidGfdzpd5EkpCcphLc/aws-serverless-how-to-stop-ec2-using-event-bridge-and-lambda
125. Automate Start/Stop AWS EC2 Instance Using Lambda: Step-by-Step Guide - K21Academy, accessed April 28, 2025, https://k21academy.com/amazon-web-services/automate-start-stop-aws-ec2-instance/
126. Scheduling EC2 start and stop using Eventbridge and Lambda (part 1 of 2), accessed April 28, 2025, https://dev.to/aws-builders/scheduling-ec2-start-and-stop-using-eventbridge-and-lambda-part-1-of-2-30he
127. Automating EC2 Instance Management with AWS EventBridge and Lambda - InterWorks, accessed April 28, 2025, https://interworks.com/blog/2024/04/02/automating-ec2-instance-management-with-aws-eventbridge-and-lambda/
128. How to start and stop AWS EC2 instance based on a time based schedule - Server Fault, accessed April 28, 2025, https://serverfault.com/questions/867642/how-to-start-and-stop-aws-ec2-instance-based-on-a-time-based-schedule
129. Amazon EC2 – Secure and resizable compute capacity - AWS, accessed April 28, 2025, https://aws.amazon.com/ec2/pricing/
130. High-Performance Block Storage– Amazon EBS Pricing - AWS, accessed April 28, 2025, https://aws.amazon.com/ebs/pricing/
131. AWS EBS Pricing | GeeksforGeeks, accessed April 28, 2025, https://www.geeksforgeeks.org/aws-ebs-pricing/
132. The Guide to AWS EBS Pricing | CloudBolt Software, accessed April 28, 2025, https://www.cloudbolt.io/guide-to-aws-cost-optimization/aws-ebs-pricing/
133. EBS Pricing Explained: A Guide For 2025 - CloudZero, accessed April 28, 2025, https://www.cloudzero.com/blog/ebs-pricing/
134. What is AWS data transfer pricing? | AWS bandwidth pricing - Cloudflare, accessed April 28, 2025, https://www.cloudflare.com/learning/cloud/what-is-aws-data-transfer-pricing/
135. The Ultimate Guide to AWS Data Transfer Pricing | CloudForecast, accessed April 28, 2025, https://www.cloudforecast.io/guides/aws-data-transfer-cost/
136. AWS Data Transfer Pricing: Hidden Network Transfer Costs and What to Do About Them, accessed April 28, 2025, https://bluexp.netapp.com/blog/aws-cvo-blg-aws-data-transfer-costs-solving-hidden-network-transfer-costs

137. Overview of Data Transfer Costs for Common Architectures - AWS, accessed April 28, 2025, https://aws.amazon.com/blogs/architecture/overview-of-data-transfer-costs-for-common-architectures/
138. EC2 On-Demand Instance Pricing – Amazon Web Services, accessed April 28, 2025, https://aws.amazon.com/ec2/pricing/on-demand/
139. Pricing and billing for archiving Amazon EBS snapshots, accessed April 28, 2025, https://docs.aws.amazon.com/ebs/latest/userguide/snapshot-archive-pricing.html