🎬 Video 2: Production Serving with vLLM

To achieve high-speed inference, we are replacing standard Python environments with **Conda**. Conda is the "Industry Standard" for AI because it manages complex math and GPU libraries (like CUDA) much better than regular Python.

Goal: Deploy a professional-grade inference server that mimics the OpenAI API—meaning any app designed for ChatGPT can now talk to **your** private engine! 🛡️

Step 1: Connect & Prepare Your Engine 🔗

First, let's get back into our server. SSH to your VM:

```
# 1. Jump into your Sovereign AI Engine
# This command uses Google's secure tunnel to give you command-line
access.
gcloud compute ssh sovereign-ai-engine --zone=us-east1-b
```

Step 2: Install the Miniconda "Toolbox" 🛠️

Since we are on a fresh, clean Debian VM, we need to install **Miniconda**. Think of this as the "System Architect" that will manage all our AI software versions for us.

Bash

```
# 1. Switch to the root user to ensure we have full permissions for
the install
sudo su
cd /

# 2. Grab the latest Miniconda installer from the official source
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-
x86_64.sh

# 3. Run the installer script
# The -b flag means 'batch' (don't ask me questions) and -p sets the
install path.
bash Miniconda3-latest-Linux-x86_64.sh -b -p /miniconda3

# 4. Activate Conda and initialize it for your terminal
# This ensures that every time you log in, the 'conda' command is
ready to go.
source /miniconda3/bin/activate
conda init bash
source ~/.bashrc
```

Step 3: Build Your Dedicated AI Environment 🏗️

Now we create a specialized "room" (a virtual environment) just for our vLLM server. This keeps our project clean and prevents different software versions from fighting with each other. 🥊

Bash

```
# 1. Accept the terms of service for the official Anaconda channels
# This ensures we are using authorized, stable software repositories.
conda tos accept --override-channels --channel
https://repo.anaconda.com/pkgs/main
conda tos accept --override-channels --channel
https://repo.anaconda.com/pkgs/r

# 2. Create the 'vllm-env' environment using Python 3.10
# We use 3.10 because it's the 'Sweet Spot' for stability in the AI
world right now.
conda create -n vllm-env python=3.10 -y

# 3. Enter your new environment
# You will see (vllm-env) appear next to your username in the
terminal!
conda activate vllm-env
```

Why this matters for your Students 🎓

- **Conda vs. Venv:** In your labs, students will find that installing GPU-heavy libraries like `vLLM` often fails in standard Venvs because of missing C++ compilers. Conda avoids this by shipping the pre-compiled tools directly to the VM.
- **Production Ready:** By the end of this video, your students won't just have a "script" running—they'll have a **Server** capable of powering enterprise chatbots.
