Step 1: Install Dependencies

Before installing `pyenv`, you need to ensure that your system has all the required dependencies. For a typical Ubuntu system on AWS, you can install these dependencies with:

bash
Copy code
```
sudo apt update
sudo apt install -y build-essential libssl-dev zlib1g-dev libbz2-dev \
libreadline-dev libsqlite3-dev wget curl llvm libncurses5-dev
libncursesw5-dev \
xz-utils tk-dev libffi-dev liblzma-dev python-openssl git
```

Step 2: Install `pyenv`

You can install `pyenv` using its installer script or manually from the GitHub repository. Here's how to do it with the installer script:

bash
Copy code
```
curl https://pyenv.run | bash
```

This script will clone `pyenv` and its two commonly used plugins (`pyenv-virtualenv` and `pyenv-update`) into the `~/.pyenv` directory.

Step 3: Configure Environment Variables

After installation, add `pyenv` to your shell startup file (`.bashrc`, `.zshrc`, etc.):

bash
Copy code
```
echo 'export PYENV_ROOT="$HOME/.pyenv"' >> ~/.bashrc
echo 'export PATH="$PYENV_ROOT/bin:$PATH"' >> ~/.bashrc
echo 'eval "$(pyenv init --path)"' >> ~/.bashrc
echo 'eval "$(pyenv virtualenv-init -)"' >> ~/.bashrc

# Reload the shell configuration
source ~/.bashrc
```

For other shells, you might need to modify the configuration files accordingly.

Step 4: Install Python Versions

Now, use `pyenv` to install the Python versions you need:

bash
Copy code
```
pyenv install 3.10.6  # Install Python 3.10.6
pyenv install 3.12.0  # Install Python 3.12.0
```

Step 5: Set Global and Local Python Versions

Set the system-wide (global) Python version:

bash
Copy code
```
pyenv global 3.12.0
```

For a specific project that requires Python 3.10:

bash
Copy code
```
# Navigate to your project directory
cd /path/to/your/project

# Set the local Python version for this project
pyenv local 3.10.6
```

Step 6: Verify Python Versions

Verify the Python version in use with:

bash
Copy code
```
python --version  # Should return Python 3.12.0 globally or 3.10.6 in
the project directory
```

You can also check which version is active in any directory:

bash
Copy code
```
pyenv version
```

This setup allows you to manage multiple Python environments efficiently on a single AWS instance, switching between them as required for different projects.