Huawei HCCDA-AI certification

Instructor : Fawad Bahadur

3

CORVIT
Reaching end to end

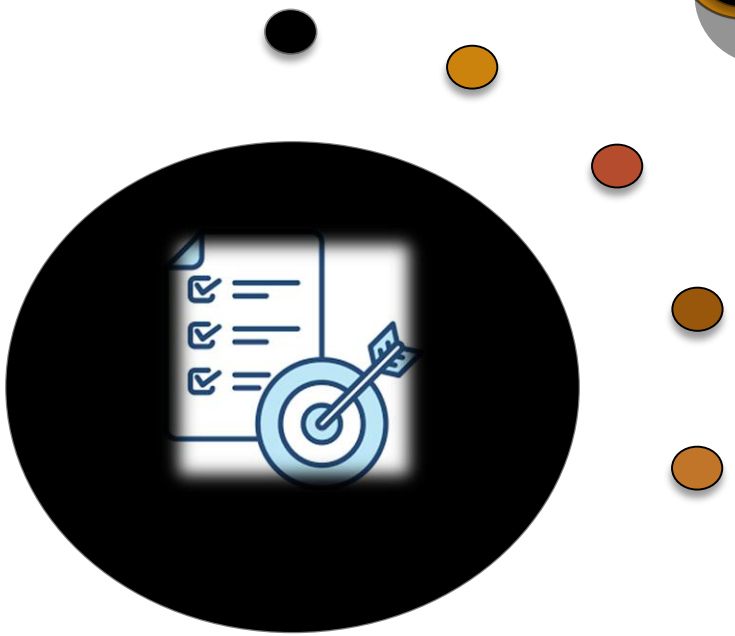
Virtual Environments for AI Projects

From Basic to Expert-Level Workflow

Dated: 14 June 2025

Instructor : Fawad Bahadur

Training Objectives & Outcomes



1. Understand the role and benefits of virtual environments in AI projects
2. Learn how to create, activate, and manage isolated environments
3. Install and manage AI libraries (e.g., NumPy, Pandas, scikit-learn)
4. Integrate environments with Jupyter, IDEs, and automation workflows

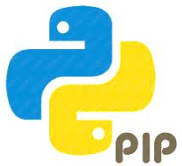
CORVIT
Reaching end to end

Python Basics – Pre-requisite

Why Learn This First?

- Virtual environments depend on basic Python and command-line usage
- You'll use pip, terminal commands, and Python scripts during setup





Tip

Always ensure your Python version is compatible with the AI libraries you plan to use (e.g., 3.8 to 3.11).

Key Requirements

1. Python Installation

Download from: www.python.org

During install: ✔ Add Python to PATH

Verify:

bash

```
python –version
pip –version
```

2. Basic Command-Line Skills

- Navigate: cd folder/
- Create folder: mkdir ai_project
- Run script: python script.py

3. IDE/Text Editor

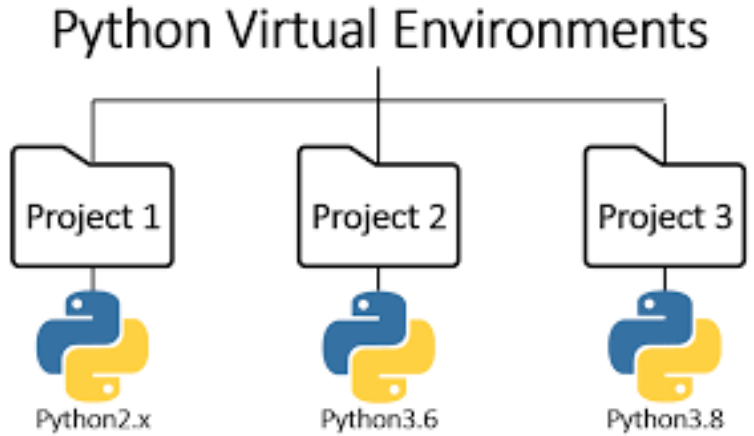
Recommended: VS Code, PyCharm, JupyterLab

Instructor : Fawad Bahadur

What is a Virtual Environment?

Definition

A virtual environment is an **isolated workspace** in Python where you can **install packages and dependencies** without affecting the **global Python installation.**



Why It Matters in AI Projects

- Prevents version conflicts between projects
- Keeps your system clean and organized
- Allows different projects to use different library versions
- Essential for reproducibility and collaboration

Without Virtual Env	With Virtual Env
Global installs	Isolated installs
Risk of conflicts	Safe experiments
Hard to replicate	Easy sharing via requirements.txt

Tip

Never install AI/ML libraries globally — always use a virtual environment.

CORVIT
Reaching end to end

Instructor : Fawad Bahadur

Virtual Environment Tools Overview

Common Tools for Creating Environments

1. **venv**

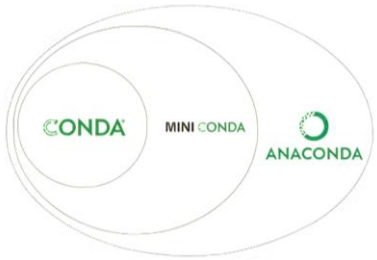
- Built-in with Python 3.3+
- Lightweight and simple
- Ideal for basic AI projects

2. **virtualenv**

- More features than venv
- Works with older Python versions
- Better for flexibility

3. **conda (Anaconda/Miniconda)**

- Used widely in data science & AI
- Manages packages and environments
- Great for GPU-based projects and mixed language dependencies



Feature	venv	virtualenv	conda
Built-in	✔	✘	✘
Lightweight	✔ ✔	✔	✘
Package mgmt	pip	pip	conda/pip
Cross-platform	✔	✔	✔

Tip

Use venv or conda depending on whether you prioritize simplicity or full-stack AI setups.

CORVIT
Reaching end to end

Instructor : Fawad Bahadur

Creating an AI Environment using venv

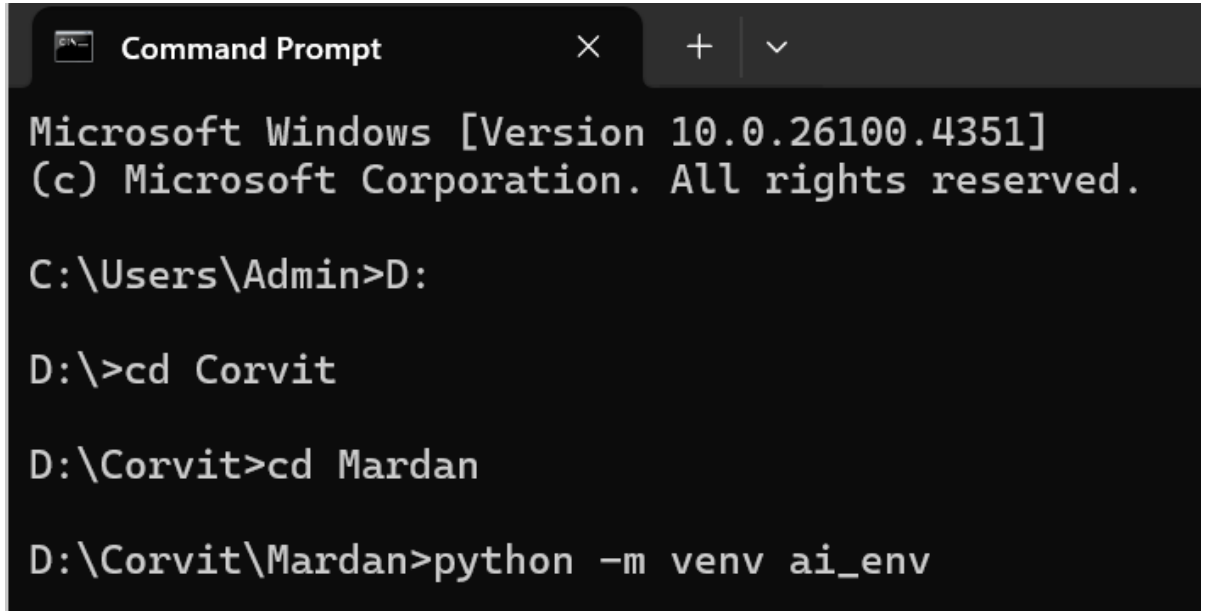
Step-by-Step Guide

1. Open your terminal or command prompt
2. Navigate to your project folder

   `cd path/to/your/project`

3. Create the environment

   `python -m venv ai_env`



This creates a folder named ai_env with isolated Python.

What's Inside ai_env/

Scripts/ → Activation and execution files
Lib/ → Installed libraries
Include/ → C headers (for compiling)
pyvenv.cfg → Config file with interpreter path

Best Practice

Keep your virtual environment in the root of your project folder (but don't push it to GitHub)

CORVIT
Reaching end to end

Instructor : Fawad Bahadur

Activating & Deactivating the Environment

Activation Commands (OS Specific)

- **Windows (CMD):**
  `ai_env\Scripts\activate`
- **Windows (PowerShell):**
  `\ai_env\Scripts\Activate.ps1`
- **Mac/Linux:**
  `source ai_env/bin/activate`

Visual Cue

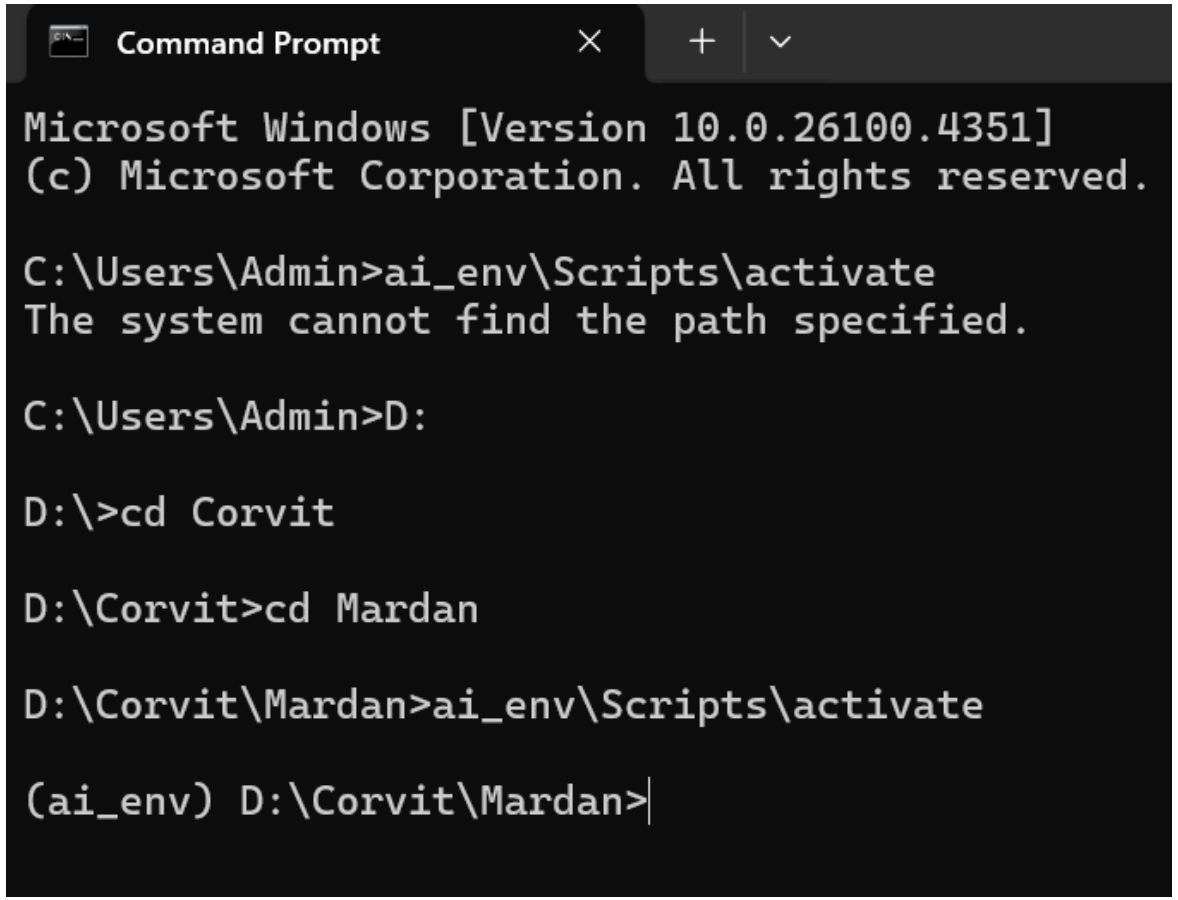
Once activated, your terminal prompt changes:

`(ai_env) C:\Users\YourName\project>`

Deactivate Environment

To exit the virtual environment:

`deactivate`



Tip

- Always activate before running or installing anything
- Keep one terminal open per environment to avoid confusion

CORVIT
Reaching end to end

Installing Essential AI Libraries

Installing Packages with pip

After activating your environment, install core AI/data libraries:

pip install numpy pandas scikit-learn matplotlib seaborn

```
D:\Corvit\Mardan>ai_env\Scripts\activate

(ai_env) D:\Corvit\Mardan>pip install numpy pandas scikit-learn matplotlib seaborn
```

What These Libraries Do

- **NumPy:** Numerical computing with arrays
- **Pandas:** Data manipulation and analysis
- **scikit-learn:** Machine learning algorithms
- **Matplotlib & Seaborn:** Data visualization

Verifying Installation

Check installed packages and versions:

pip list

Tip

Use a requirements.txt file to save dependencies for reproducibility (covered in next slides).

Adding Jupyter Notebook Support

Why Jupyter?

- Interactive coding environment
- Popular for AI research, data exploration, and demos

Installing Jupyter & Kernel

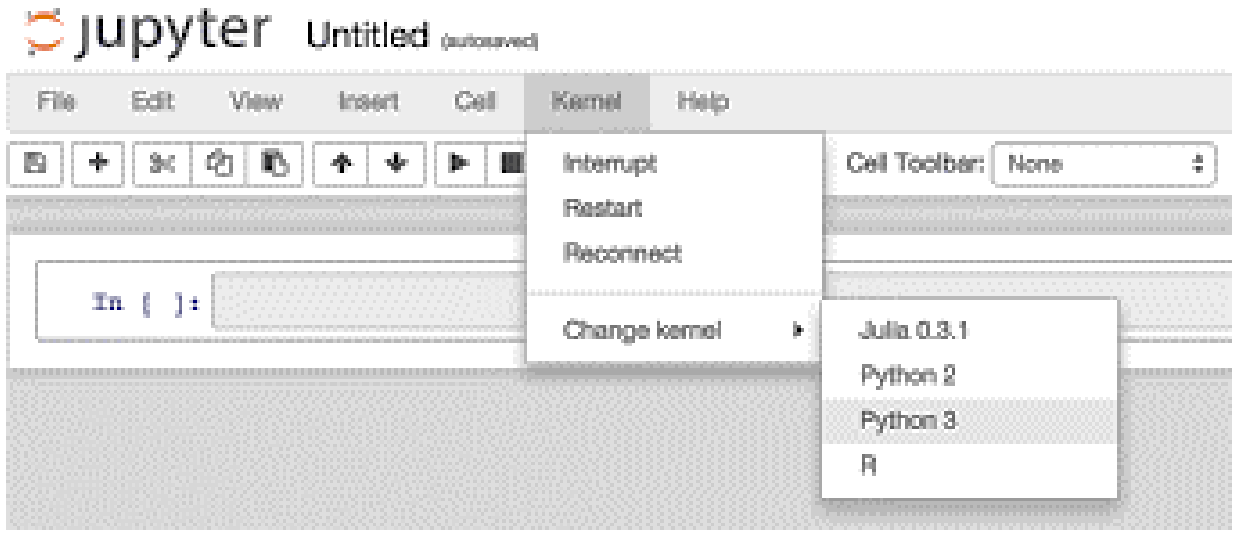
```
pip install jupyter ipykernel
python -m ipykernel install
--user --name=ai_env --display-name "Python (ai_env)"
```

Using Jupyter

Start Jupyter Notebook:

```
bash      jupyter notebook
```

Select your virtual environment kernel named Python (ai_env) for your notebooks



Tip

Use JupyterLab (pip install jupyterlab) for an enhanced interface.

Managing Dependencies with requirements.txt

Why Save Dependencies?

- Share exact package versions
- Reproduce the same environment on any machine
- Essential for collaboration and deployment

Creating requirements.txt

After installing your packages:

```bash
pip freeze > requirements.txt
```

Installing from requirements.txt

To recreate environment elsewhere:

```bash
pip install -r requirements.txt
```

requirements.txt

```
pip==22.3.1
wheel==0.38.4
Pillow==9.5.0
setuptools==65.5.1
packaging==23.1
numpy==1.26.0
```

Tip

Update requirements.txt regularly after adding or updating packages.

CORVIT
Reaching end to end

Managing Environments with Conda

What is Conda?

- An open-source package & environment manager
- Supports Python & other languages
- Popular in AI/data science for easy handling of complex dependencies

Activating Conda Environment

Bash `conda activate ai_env`

Installing Conda

- Download & install Anaconda or Miniconda from https://conda.io
- Use conda commands in terminal after installation

Installing Packages

Bash `conda install numpy pandas scikit-learn matplotlib seaborn`

Creating a Conda Environment

Bash `conda create -n ai_env python=3.9`

Tip

Conda manages both packages and environments, ideal for GPU-based AI projects.

Instructor : Fawad Bahadur

Best Practices for Virtual Environments

One Environment = One Project

- Avoid using the same environment for multiple projects
- Keeps dependencies clean and isolated

Never Push Env to Git

Add this to your .gitignore:

```
ai_env/
```

Pin Package Versions

- Prevent future compatibility issues

Example:

bash

```
pip install scikit-learn==1.3.2
```

Document Your Environment

Include setup steps in README.md

Example:

```
python -m venv ai_env    source ai_env/bin/activate
pip install -r requirements.txt
```

Use requirements.txt or environment.yml

- Always include with your code for reproducibility
- Automate project setup for collaborators or deployment

Tip

Transition to Dask gradually by converting Pandas/NumPy code.

CORVIT
Reaching end to end

Instructor : Fawad Bahadur

Using Virtual Environments in IDEs

Why Integrate with IDEs?

- Enables features like code suggestions, debugging, linting
- Ensures your environment is correctly used while developing

Tip

Always double-check the interpreter path matches your virtual environment.

Jupyter Notebooks

Run:

Bash

```
python -m ipykernel install --user --name=ai_env
```

- Use kernel in notebook dropdown

PyCharm

Add Interpreter:

- Go to Settings > Project > Python Interpreter
- Click gear → Add → Existing environment
- Also works with Conda environments

VS Code

Select Interpreter:

- Open Command Palette → Python: Select Interpreter
- Choose your env:

```
.venv/bin/python or
ai_env\Scripts\python.exe
```

- Auto-activates env in terminal

Automating Virtual Environment Setup

Why Automate?

- Saves time in team setups
- Ensures consistent environments
- Ideal for onboarding, CI/CD, or deployment

Conda Environment YAML

Create YAML:

```
conda env export > environment.yml
```

Recreate Env:

```
conda env create -f environment.yml
```

Setup Script (venv + pip)

setup_env.sh (Linux/macOS):

Bash

```
python -m venv ai_env
source ai_env/bin/activate
pip install -r requirements.txt
```

setup_env.bat (Windows):

```
python -m venv ai_env
call ai_env\Scripts\activate
pip install -r requirements.txt
```

Tip

Add script or YAML instructions to your README.md.

CORVIT
Reaching end to end

Instructor : Fawad Bahadur

Virtual Environments vs Docker for AI Projects

Virtual Environments

- Lightweight, quick to set up
- Isolate Python packages
- Great for development and small projects

Docker

- Isolates the entire system (OS + Python + libs)
- Ideal for deployment, scaling, and GPU compatibility
- Works the same on every machine

Feature	Virtual Env	Docker
Scope	Python packages	Full environment
Portability	Medium	High
Reproducibility	Good	Excellent
Setup Complexity	Low	Moderate/High
Best Use	Dev & prototyping	Deployment & CI/CD

Try

Start with virtual envs, move to Docker for production or collaboration-heavy AI systems.

Real-World Virtual Env Setup for AI Project

Project Goal: Predict Student Dropout Using AI

Folder Structure

student-dropout-ai/	
ai_env/	Virtual environment (not in Git)
data/	Raw & processed datasets
notebooks/	Jupyter notebooks
src/	Python scripts
outputs/	Models, logs, results
requirements.txt	Dependencies
README.md	Setup + usage instructions

Setup Steps

Bash

```
python -m venv ai_env
source ai_env/bin/activate
pip install -r requirements.txt
```

OR (Conda)

Bash

```
conda create -n ai_env python=3.10
conda activate ai_env
conda install numpy pandas scikit-learn jupyter
```

Tip

Use Dask-ML to scale your ML pipelines effortlessly.

CORVIT
Reaching end to end

Instructor : Fawad Bahadur

Q & A

CORVIT
Reaching end to end

Instructor : Fawad Bahadur

THANK YOU

CORVIT
Reaching end to end

Instructor : Fawad Bahadur