🧩 Lab Setup Guide – Windows & macOS

🎯 Objective

Prepare a unified development environment that runs production automation workloads — including APIs, databases, dashboards, and monitoring tools — consistently on **Windows 10/11** and **macOS**.

⚙️ 1. System Requirements

Component	Minimum	Recommended
CPU	4 cores	8 cores
RAM	8 GB	16 GB
Storage	30 GB free	60 GB free
OS	Windows 10/11 (Pro) or macOS 12+	Latest stable release
Internet	Required for package installations	Broadband (20 Mbps+)

🧰 2. Tool Installation

🪟 For Windows Users

Step 1: Install Python 3.10+

- Download from https://www.python.org/downloads/windows
- During setup, check ✅ "Add Python to PATH"
- Verify installation:

```
python --version
pip install pandas flask requests sqlalchemy matplotlib psycopg2
```

Step 2: Install Docker Desktop

- Download: https://www.docker.com/products/docker-desktop
- Enable **WSL 2 Backend** (Docker will prompt)
- Verify:

```
docker --version
docker-compose --version
```

Step 3: Install PostgreSQL

- Download installer: https://www.postgresql.org/download/windows/
- Default credentials:
  Username: `postgres`
  Password: `admin123`

Step 4: Install VS Code

- Download from https://code.visualstudio.com/
- Extensions:
  - Python
  - Docker
  - REST Client
  - SQLTools (for DB integration)

Step 5: Install Git & Postman

- Git: https://git-scm.com/download/win
- Postman: https://www.postman.com/downloads/
- Verify Git:

```
git --version
```

🧱 Creating a Python Virtual Environment

A **virtual environment (venv)** helps isolate dependencies per project so your production applications don't conflict with system Python packages.

🪟 For Windows Users

Step 1 – Check Python Installation

Open **Command Prompt** or **PowerShell**:

```
python --version
```

✅ You should see a version like:
`Python 3.10.12`

If not, install Python from https://www.python.org/downloads/windows/
Make sure to **check the box** "Add Python to PATH" during installation.

Step 2 – Create the Virtual Environment

Navigate to your project folder:

```
cd C:\Users\<yourname>\production-automation-starter
```

Now create the environment:

```
python -m venv venv
```

This creates a folder named `venv/` inside your project containing a standalone Python setup.

Step 3 – Activate the Environment

In Command Prompt:

```
venv\Scripts\activate
```

You'll notice your terminal prompt changes to:

```
(venv) C:\Users\<yourname>\production-automation-starter>
```

That means your virtual environment is **active**.
Any package you install now will go *only* into this environment.

Step 4 – Install Required Packages

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

If you don't have a `requirements.txt` file yet, manually install key modules:

```
pip install flask pandas requests sqlalchemy matplotlib psycopg2
```

Step 5 – Verify Packages

Check that your packages installed successfully:

```
pip list
```

Step 6 – Deactivate Environment

When done working:

```
deactivate
```

🍏 For macOS / Linux Users

Step 1 – Verify Python

```
python3 --version
```

✅ Example Output: `Python 3.10.14`

If not installed, use:

```
brew install python3
```

Step 2 – Create Virtual Environment

Navigate to your project directory:

```
cd ~/production-automation-starter
python3 -m venv venv
```

Step 3 – Activate Environment

```
source venv/bin/activate
```

Your terminal prompt should change to:

```
(venv) MacBook-Pro:production-automation-starter venkatesh$
```

Step 4 – Install Dependencies

```
pip3 install -r requirements.txt
```

or manually:

```
pip3 install flask pandas requests sqlalchemy matplotlib psycopg2
```

Step 5 – Confirm Installation

```
pip3 list
```

Step 6 – Deactivate Environment

deactivate

🧩 Troubleshooting Tips

Problem	Solution
"python not recognized"	Add Python to PATH during install
"venv not found"	Install venv module: `pip install virtualenv`
Permission denied (macOS)	Use `sudo chown -R $USER:$USER project-folder`
Wrong Python version	Use `python3` instead of `python` on macOS/Linux

Day 1: Python Foundations + Banking/Trading AI Agent, here's a **complete pip package list** based on your agenda. I've categorized them by purpose so it's easier to manage.

1. Core Python Essentials

pip install pandas # Dataframes, CSV/JSON handling

pip install numpy # Numeric operations

pip install python-dotenv # Environment variable management

2. Advanced Python / Utilities

pip install loguru # Logging decorator / structured logging

pip install pydantic # Data validation for classes/API

pip install typer # Optional, CLI app support for testing scripts

3. API / Async / Fetching Live Data

```
pip install requests      # Synchronous HTTP requests

pip install aiohttp       # Async HTTP requests for live prices
```

4. AI / Agent Tools

```
pip install langchain      # Core LangChain library

pip install openai         # OpenAI SDK for GPT models

pip install semantic-kernel # Semantic Kernel SDK
```

5. Optional / Visualization / Demo

```
pip install matplotlib     # For plotting data if needed

pip install plotly         # Interactive plots (optional)
```

6. Testing / Debugging

```
pip install pytest         # Unit tests for classes and functions
```

7. Quick Install Command (All-in-One)

```
pip install pandas numpy python-dotenv loguru pydantic typer requests aiohttp langchain openai
semantic-kernel matplotlib plotly pytest
```

Day 2: Advanced Agents + Energy/Environment Case Studies, here’s a **complete pip package list** based on your agenda. I’ve categorized them for clarity.

1. Core / Data Handling

```
pip install pandas        # CSV, JSON, tabular data

pip install numpy         # Numeric operations

pip install python-dotenv # Environment variable management
```

2. API / HTTP / IoT Integration

```
pip install requests      # REST API calls

pip install aiohttp      # Async HTTP requests
```

3. Database / Storage

```
pip install sqlalchemy        # ORM for database access

pip install psycopg2-binary  # PostgreSQL driver

pip install influxdb-client   # Optional: time-series DB for IoT sensors
```

4. AI / Agent Frameworks

```
pip install langchain          # LangChain for agents, tools, chains

pip install openai            # OpenAI GPT models

pip install semantic-kernel    # Semantic Kernel advanced functions

pip install faiss-cpu          # Vector store / embeddings

pip install sentence-transformers # Embeddings for context & RAG
```

5. RAG / Retrieval Pipelines

```
pip install chromadb          # Vector DB (alternative to FAISS)

pip install weaviate-client    # Optional vector search DB client
```

6. Visualization & Dashboards

```
pip install matplotlib        # Plotting energy usage, carbon footprint

pip install seaborn           # Enhanced visualizations

pip install plotly            # Interactive charts
```

7. Scheduling / Automation

```
pip install apscheduler       # For scheduling periodic tasks

pip install schedule          # Lightweight scheduler alternative
```

8. Logging / Utilities

```
pip install loguru            # Pretty logging

pip install structlog         # Structured logging

pip install pydantic          # Data validation for agent payloads
```

9. Testing / Debugging

pip install pytest   # Unit tests for modules & agents

10. Quick Install Command (All-in-One)

pip install pandas numpy python-dotenv requests aiohttp sqlalchemy psycopg2-binary influxdb-client langchain openai semantic-kernel faiss-cpu sentence-transformers chromadb weaviate-client matplotlib seaborn plotly apscheduler schedule loguru structlog pydantic pytest

Day 3: Multi-Agent Systems + Capstone, here's a **complete pip package list** based on your agenda. Since Day 3 is essentially an **end-to-end combination of Day 1 + Day 2 plus advanced multi-agent orchestration**, this includes all necessary packages.

1. Core / Data Handling

pip install pandas   # CSV, JSON, tabular data

pip install numpy   # Numeric operations

pip install python-dotenv # Environment variable management

2. API / HTTP / IoT Integration

pip install requests   # REST API calls

pip install aiohttp   # Async HTTP requests

3. Database / Storage

pip install sqlalchemy   # ORM for database access

pip install psycopg2-binary # PostgreSQL driver

pip install influxdb-client   # Optional: time-series DB for IoT sensors

4. AI / Agent Frameworks

```
pip install langchain          # LangChain agents, tools, chains

pip install openai             # OpenAI GPT models

pip install semantic-kernel    # Semantic Kernel advanced functions

pip install faiss-cpu          # Vector store / embeddings

pip install sentence-transformers # Embeddings for context & RAG
```

5. RAG / Retrieval Pipelines

```
pip install chromadb           # Vector DB (alternative to FAISS)

pip install weaviate-client    # Optional vector search DB client
```

6. Multi-Agent Orchestration / Scheduling

```
pip install apscheduler        # For scheduling periodic tasks

pip install schedule           # Lightweight scheduler alternative

pip install ray                # Multi-agent / distributed orchestration
```

7. Visualization & Dashboards

```
pip install matplotlib         # Plotting energy usage, carbon footprint

pip install seaborn            # Enhanced visualizations

pip install plotly             # Interactive charts and dashboards
```

8. Logging / Observability / Governance

```
pip install loguru           # Pretty logging
pip install structlog         # Structured logging
pip install pydantic          # Data validation
```

9. Testing / Debugging

```
pip install pytest           # Unit tests for modules & agents
```

10. Quick Install Command (All-in-One)

```
pip install pandas numpy python-dotenv requests aiohttp sqlalchemy psycopg2-binary
influxdb-client langchain openai semantic-kernel faiss-cpu sentence-transformers chromadb
weaviate-client apscheduler schedule ray matplotlib seaborn plotly loguru structlog pydantic
pytest
```

```
pip install pandas numpy python-dotenv requests aiohttp sqlalchemy psycopg2-binary
influxdb-client langchain openai semantic-kernel faiss-cpu sentence-transformers chromadb
weaviate-client apscheduler schedule ray matplotlib seaborn plotly loguru structlog pydantic
pytest typer
```

pandas

numpy

python-dotenv

requests

aiohttp

sqlalchemy

psycopg2-binary

influxdb-client

langchain

openai

semantic-kernel

faiss-cpu

sentence-transformers

chromadb

weaviate-client

apscheduler

schedule

ray

matplotlib

seaborn

plotly

loguru

structlog

pydantic

pytest

typer

🍏 For macOS Users

Step 1: Install Homebrew (if not installed)

```
/bin/bash -c "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

Step 2: Install Core Packages

```
brew install python3 git postgresql docker docker-compose
pip3 install pandas flask requests sqlalchemy matplotlib psycopg2
```

Step 3: Install VS Code

- Download: https://code.visualstudio.com/
- Extensions: Python, Docker, REST Client, SQLTools

Step 4: Start PostgreSQL

```
brew services start postgresql
createdb productiondb
```

Step 5: Start Docker Desktop

- Download for mac: https://www.docker.com/products/docker-desktop
- Launch and ensure **containers can run**.