Windows 11 → WSL2 Data■Science Environment (DIY Master's in the AI Era)

A step■by■step, reproducible guide to rebuild your full setup from a clean Windows 11 install: WSL2 + Ubuntu, VS Code, Git/SSH, Conda (Py 3.13.5), Docker Desktop, GitHub Actions CI, and a project skeleton in *ds-zero-to-one*.

0) Overview & Assumptions

This document rebuilds the exact environment you used for the curriculum:
• Windows 11 (administrator access), internet available.
• GitHub account available.
• Final state uses: WSL2 (Ubuntu), VS Code + Remote■WSL, Docker Desktop (Linux containers via WSL2), Conda 25.5.1 with Python 3.13.5, and a GitHub Actions workflow that verifies the environment by printing “CI env OK”.
Tip: Commands marked (PowerShell) are run in Windows; (WSL) are run inside the Ubuntu terminal.

1) Install Core Windows Apps (PowerShell / Windows UI)

Install these in any order:

- Docker Desktop (Linux containers only). Enable “Use the WSL 2 based engine”; DO NOT enable Windows Containers.
- Visual Studio Code.
- Windows Terminal (bundled on Win11; install from Microsoft Store if missing).
- Git for Windows (optional; you can use Git entirely in WSL).

2) Enable WSL2 + Install Ubuntu (PowerShell as Administrator)

```
wsl --install -d Ubuntu
```

Reboot when prompted. On first Ubuntu launch, create your UNIX username and password.

3) VS Code + WSL Integration (Windows UI)

Open VS Code → Extensions → install:

- WSL (by Microsoft)
- Python (by Microsoft)
- Jupyter (by Microsoft)
- Docker (by Microsoft)
- YAML (by Red Hat)
- Markdown All in One (Yzhang)

4) Configure Docker Desktop (Windows UI)

Open Docker Desktop → Settings:
• General: ✓ Use the WSL 2 based engine
• Resources → WSL Integration: enable for Ubuntu
• (Optional) Start Docker Desktop on login

5) SSH Keys & Git Identity

5A) Reuse Windows SSH key in WSL (recommended):

```
mkdir -p ~/.ssh
cp /mnt/c/Users/<YourWindowsUser>/.ssh/id_ed25519 ~/.ssh/
cp /mnt/c/Users/<YourWindowsUser>/.ssh/id_ed25519.pub ~/.ssh/
chmod 600 ~/.ssh/id_ed25519
chmod 644 ~/.ssh/id_ed25519.pub
eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_ed25519
ssh -T git@github.com   # success message expected
```

5B) Or generate a new key inside WSL:

```
ssh-keygen -t ed25519 -C "github-wsl"
eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_ed25519
cat ~/.ssh/id_ed25519.pub   # add at GitHub → Settings → SSH and GPG Keys
```

5C) Configure Git identity (use GitHub noreply email):

```
git config --global user.name "Your Name"
git config --global user.email "12345678+yourusername@users.noreply.github.com"
git config --global init.defaultBranch main
```

6) Get Your Repository (WSL)

Clone existing repo that contains the subfolder ds-zero-to-one:

```
mkdir -p ~/projects
cd ~/projects
git clone git@github.com:<your-username>/Masters-level-DIY-Data-Science-Curriculum-ai-Era-.git
cd Masters-level-DIY-Data-Science-Curriculum-ai-Era-
git remote -v   # should show SSH URLs
```

7) Project Structure (inside ds-zero-to-one)

```
cd ~/projects/Masters-level-DIY-Data-Science-Curriculum-ai-Era-
mkdir -p ds-zero-to-one
cd ds-zero-to-one

mkdir -p data/raw data/processed notebooks reports src tests .github/workflows
touch data/.gitkeep data/raw/.gitkeep data/processed/.gitkeep       notebooks/.gitkeep reports/.gitkee

cat > .gitignore <<'EOF'
data/**
```

```
!data/.gitkeep
!data/raw/.gitkeep
!data/processed/.gitkeep
**/.ipynb_checkpoints
__pycache__/
*.pyc
.env
.venv
EOF

cat > .gitattributes <<'EOF'
* text=auto
*.py      text eol=lf
*.ipynb   text eol=lf
*.yml     text eol=lf
*.md      text eol=lf
*.csv     text eol=lf
*.ps1     text eol=crlf
*.bat     text eol=crlf
*.png binary
*.jpg binary
*.pdf binary
EOF

git add .
git commit -m "chore: scaffold project structure"
git push --set-upstream origin main
```

8) Install Miniconda (WSL) & Create Env

```
sudo apt update && sudo apt install -y wget curl
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
bash Miniconda3-latest-Linux-x86_64.sh -b -p ~/miniconda3
~/miniconda3/bin/conda init bash
exec bash

conda --version
conda config --add channels conda-forge
conda config --set channel_priority strict

# Create env (or use environment.yml in next step)
conda create -n ds -c conda-forge python=3.13.5 numpy pandas scikit-learn              jupyterlab ipyk
conda activate ds
pip install polars
python -m ipykernel install --user --name ds --display-name "Python (ds)"
```

9) Add environment.yml (preferred)

```
cat > environment.yml <<'YML'
name: ds
channels:
  - conda-forge
channel_priority: strict
dependencies:
  - python=3.13.5
  - numpy
  - pandas
  - matplotlib
  - scikit-learn
  - jupyterlab
  - ipykernel
  - duckdb
  - plotly
  - pip
  - pip:
      - polars
YML

git add environment.yml
git commit -m "chore: add environment.yml"
git push
```

10) JupyterLab Smoke Test

```
cd ~/projects/Masters-level-DIY-Data-Science-Curriculum-ai-Era-/ds-zero-to-one
conda activate ds
jupyter lab --no-browser --ip=0.0.0.0
# In a new notebook, run:
# import pandas as pd, duckdb, polars as pl
# print("Environment OK!", pd.__version__, duckdb.__version__, pl.__version__)
```

11) GitHub Actions CI (repo root)

```
cd ~/projects/Masters-level-DIY-Data-Science-Curriculum-ai-Era-
mkdir -p .github/workflows
cat > .github/workflows/ci.yml <<'CI'
---
name: CI
'on':
  - push
  - pull_request

jobs:
  build:
    runs-on: ubuntu-latest
    defaults:
      run:
        working-directory: ds-zero-to-one
        shell: bash -l {0}
    steps:
      - uses: actions/checkout@v4
      - uses: conda-incubator/setup-miniconda@v3
        with:
          miniforge-version: latest
          use-mamba: true
          auto-activate-base: false
          conda-remove-defaults: true
          channels: conda-forge
          channel-priority: strict
          environment-file: ds-zero-to-one/environment.yml
          activate-environment: ds
      - name: Sanity check
        run: python -c "import pandas, duckdb, polars as pl; print('CI env OK')"
CI

git add .github/workflows/ci.yml
git commit -m "ci: reproducible conda-forge workflow"
git push
```

12) Optional Docker Dev Image

```
cat > ds-zero-to-one/.dockerignore <<'DIGN'
__pycache__/
*.pyc
.ipynb_checkpoints/
.env
.env.*
data/
models/
mlruns/
node_modules/
DIGN

cat > ds-zero-to-one/Dockerfile <<'DF'
FROM mambaorg/micromamba:1.5.10
WORKDIR /workspace
COPY environment.yml /tmp/environment.yml
ARG MAMBA_DOCKERFILE_ACTIVATE=1
RUN micromamba create -y -n ds -f /tmp/environment.yml && micromamba clean --all --yes
SHELL ["/bin/bash", "-lc"]
ENV CONDA_DEFAULT_ENV=ds
EXPOSE 8888
COPY . /workspace
CMD jupyter lab --ip=0.0.0.0 --port=8888 --no-browser --NotebookApp.token="" --NotebookApp.password=""
DF

cd ~/projects/Masters-level-DIY-Data-Science-Curriculum-ai-Era-/ds-zero-to-one
docker build -t ds-env .
docker run --rm -p 8888:8888 -v $(pwd):/workspace ds-env
```

13) Validation Checklist

- SSH: ssh -T git@github.com → success message
- Docker: docker run hello-world → 'Hello from Docker!'
- Conda: conda --version (25.5.x); python --version (3.13.5) in env
- Jupyter: prints versions for pandas/duckdb/polars
- GitHub Actions: Actions → CI → 'CI env OK'

14) Troubleshooting (Quick Fixes)

Polars conda solver conflicts: Install polars via pip (keep conda-forge only).

CI can't import pandas: Ensure activate-environment: ds and defaults.run.shell: 'bash -l {0}', and correct environment-file path.

SSH passphrase prompts each push: eval "$(ssh-agent -s)"; ssh-add ~/.ssh/id_ed25519; add to ~/.bashrc to auto-load.

Docker not found in WSL: Enable WSL integration in Docker Desktop; run 'wsl --shutdown', reopen Ubuntu.

No workflows listed: Workflows must live at repo root .github/workflows/. Use working-directory to target subfolder.