**Step-by-Step with Free Resources**

**1. Data Extraction (Python + OpenWeatherMap)**

* **OpenWeatherMap**: Free tier allows 60 API calls/minute, 1M/month. More than enough for demo.
* **Python**: Use requests for API calls (open-source).

💡 No cost here.

**2. Data Loading & Transformation (PostgreSQL + Python)**

* **PostgreSQL**:
  + Install locally (free, open source).
  + Or use **ElephantSQL** free tier (20MB) / **Supabase free tier** (500MB DB).
* **Python**: Use psycopg2 or SQLAlchemy for loading/transformation.

💡 For demo → local PostgreSQL is enough.

**3. Data Quality Checks (SODA)**

* **Soda Core**: Open-source, free to use locally.
* Run scans with a YAML config, no need for Soda Cloud (paid).

💡 100% free with Soda Core.

**4. Functional & Unit Testing (pytest + unittest + Airflow)**

* **pytest/unittest**: Both free & open source.
* **Airflow**: Open-source, can be installed locally with pip install apache-airflow.
  + For POC, run locally in SequentialExecutor (no need for paid Composer/Managed Airflow).

💡 Entire testing stack is open-source.

**5. CI/CD (GitHub Actions)**

* **GitHub Actions**: Free for public repos; for private repos, 2000 free minutes/month.
* Enough to demo pipelines with tests + SODA scans.

💡 Free tier is sufficient.

**🔹 Optional (Visualization)**

If you want to **show results visually**:

* **Metabase** (free, open-source BI tool).
* **Superset** (free, open-source).
* Or even **Power BI Desktop** (free, though Service is paid).

✅ **Answer:** Yes, you can build your **DataOps Quality Framework** POC with **zero cost**, using free/open-source + free cloud tiers.

dataops-quality-framework/

│── extraction/ # Python scripts for API calls

│ └── fetch\_weather.py

│

│── transform/ # SQL / Python transformations

│ └── transform\_weather.py

│

│── dq\_checks/ # Data Quality checks (Soda configs, rules)

│ └── weather\_checks.yml

│

│── tests/ # Unit + functional tests

│ ├── test\_extraction.py

│ └── test\_transformation.py

│

│── dags/ # Airflow DAGs

│ └── weather\_pipeline\_dag.py

│

│── .github/

│ └── workflows/ # GitHub Actions CI/CD pipelines

│ └── ci.yml

│

│── requirements.txt # Python dependencies

│── README.md # Documentation

# Git Commands to Create This Structure

# 1. Create a new project folder

mkdir dataops-quality-framework

cd dataops-quality-framework

# 2. Initialize git repo

git init

# 3. Create subfolders

$folders = "extraction","transform","dq\_checks","tests","dags",".github",".github\workflows"

foreach ($f in $folders) { mkdir $f }

# 4. Create placeholder files

New-Item extraction\fetch\_weather.py -ItemType File

New-Item transform\transform\_weather.py -ItemType File

New-Item dq\_checks\weather\_checks.yml -ItemType File

New-Item tests\test\_extraction.py -ItemType File

New-Item tests\test\_transformation.py -ItemType File

New-Item dags\weather\_pipeline\_dag.py -ItemType File

New-Item .github\workflows\ci.yml -ItemType File

New-Item requirements.txt -ItemType File

New-Item README.md -ItemType File

5.

mkdir C:\Users\BJeyshree\WeatherDataOps

cd C:\Users\BJeyshree\WeatherDataOps

# 6. Stage & commit

git add .

# 7. Move your project folders inside it

move C:\Users\BJeyshree\extraction C:\Users\BJeyshree\WeatherDataOps\

move C:\Users\BJeyshree\transform C:\Users\BJeyshree\WeatherDataOps\

move C:\Users\BJeyshree\dq\_checks C:\Users\BJeyshree\WeatherDataOps\

move C:\Users\BJeyshree\tests C:\Users\BJeyshree\WeatherDataOps\

move C:\Users\BJeyshree\dags C:\Users\BJeyshree\WeatherDataOps\

move C:\Users\BJeyshree\.github C:\Users\BJeyshree\WeatherDataOps\

move C:\Users\BJeyshree\requirements.txt C:\Users\BJeyshree\WeatherDataOps\

move C:\Users\BJeyshree\README.md C:\Users\BJeyshree\WeatherDataOps\

#8. Add a .gitignore file

# Ignore Windows system files

\*.log

\*.tmp

Thumbs.db

Desktop.ini

# Ignore Python cache

\_\_pycache\_\_/

\*.pyc

# Ignore virtual environments

venv/

.env/

# Ignore VS Code settings

.vscode/

# Ignore sensitive files

\*.key

\*.env