0.2 SETUP `simple-ollama-environment`

In this section we'll get your **local LLM sandbox** running: Python 3.11, Jupyter, and Ollama packaged together in a reproducible way.

You can choose either:

- A **Docker-first** setup (recommended for consistency), or
- A **local virtual environment** using your host's Python 3.11 and an existing Ollama install.

GOAL

By the end of this lab you will:

- Have the **`simple-ollama-environment`** repo cloned.
- Be able to launch a Jupyter Notebook.
- Run `notebooks/ollama_quickstart.ipynb` and chat with a local LLM (e.g., `qwen2.5:0.5b-instruct` or `llama3.2:1b`).

REPOSITORY OVERVIEW

Once cloned, you'll see something like:

```
simple-ollama-environment/
├── Dockerfile
├── Makefile
├── pyproject.toml
├── README.md
├── assets/
│   └── screenshot.png (example)
└── notebooks/
    └── ollama_quickstart.ipynb
```

Key pieces:

- **`Dockerfile`** Builds a container image that bundles:
  - Python 3.11
  - Jupyter
  - Ollama (server + CLI)
  - A small pre-pulled model (configurable)
- **`Makefile`** Cross-platform shortcuts for:
  - `make install` – local venv + kernel.
  - `make build-container` – Docker image.
  - `make run-container` – run image with ports & volumes.

STEP 1 – CLONE THE REPOSITORY

Pick or create a parent folder for all workshop repos:

```
mkdir -p ~/projects/watsonx-workshop
cd ~/projects/watsonx-workshop
```

Clone:

```
git clone https://github.com/ruslanmv/simple-ollama-environment.git
cd simple-ollama-environment
```

You should now be inside the repo root.

STEP 2 – CHOOSE SETUP PATH

You have two main options.

OPTION A – DOCKER (RECOMMENDED)

Best if:

- You want minimal local setup.
- You're happy to let Docker handle Python + Ollama.

A.1 BUILD THE CONTAINER IMAGE

From the repo root:

```
make build-container
```

Under the hood this runs `docker build` and creates an image (for example `simple-ollama-environment:latest`) that includes:

- Python 3.11 + dependencies.
- Jupyter.
- Ollama server + client.
- A tiny pre-pulled model (configurable via `PREPULL` build arg).

A.2 RUN THE CONTAINER

STEP 3 – INSTALL & CONFIGURE OLLAMA MODELS

If you’re using the Docker image with `OLLAMA_PREPULL`, some models may already be present. Otherwise, you can pull them yourself.

PULL A SMALL MODEL

Examples:

```
# From host or inside container:
ollama pull qwen2.5:0.5b-instruct
ollama pull llama3.2:1b
```

These are small enough to work well on most laptops.

QUICK HEALTH CHECK

With the container running, you can test:

```
curl http://localhost:11434/api/tags
```

You should see JSON listing available models.

STEP 4 – RUN `ollama_quickstart.ipynb`

Now let’s test end-to-end.

1. Open **Jupyter** (either inside the container or local).
2. Navigate to `notebooks/`.
3. Open `ollama_quickstart.ipynb`.
4. Run the cells top to bottom.

You should see something along the lines of:

```
import ollama

response = ollama.chat(
    model="qwen2.5:0.5b-instruct",
    messages=[{"role": "user", "content": "Tell me a joke about AI and coffee."}],
)
print(response["message"]["content"])
```

If everything is wired correctly, you’ll get a text response from the model.

HOW THIS RELATES TO RAG & THE ACCELERATOR

Right now, you're just sending plain prompts to a local model, but the same patterns will be used later when you:

- Implement a **local RAG notebook** (`rag_local_ollama.ipynb`).
- Compare local RAG vs watsonx.ai RAG.
- Treat local LLMs and watsonx.ai as interchangeable "generation backends" in the **accelerator**.

What you're learning here:

- How to:
  - Talk to Ollama's HTTP API / Python client.
  - Run notebooks in a controlled environment.
- Will directly transfer to:
  - Calling watsonx.ai in the other repo.
  - Plugging a watsonx.ai LLM into the `accelerator/rag/pipeline.py`.

TROUBLESHOOTING

OLLAMA NOT REACHABLE

- Make sure the container is running (`docker ps`) or the desktop app/service is started.
- Check that `curl http://localhost:11434/api/tags` returns JSON.
- In Docker: ensure you mapped `-p 11434:11434`.

JUPYTER TOKEN ISSUES

- If the browser asks for a token:

```
docker logs simple-ollama-env | grep "http://127.0.0.1"
```

Copy the URL with the token.

MODEL TOO BIG / OUT OF MEMORY

- If 7B or 13B models crash:
  - Use smaller models (0.5B–1B).
  - Close other applications.
  - Reduce concurrency.

PORTS ALREADY IN USE

CHECKLIST

Before moving on:

- ✅ Repo cloned (`simple-ollama-environment`)
- ✅ Dependencies installed (Docker image or venv)
- ✅ Jupyter starts successfully
- ✅ `ollama_quickstart.ipynb` runs a model and prints a response

If all green: you're ready to set up `simple-watsonx-enviroment` next.