RAG_Local_HF_Weaviate_v3 (Draft)

Introduction & How to Use This Manual

Welcome to the Local RAG System Project

This manual walks you through creating a **Local Retrieval-Augmented Generation (RAG)** system using:

- **Weaviate** (vector database)
- **Hugging Face** (pre-trained AI models)
- **Python** (the glue that ties everything together)
- **Docker** (coming soon in Step 2)

A Companion to the TicTec Series

We're pairing this manual with the **TicTec** article series, which gives you context, storytelling, and daily how-to guides. The **manual** is your lab workbook—where you'll find the **exact steps** to follow along.

Quick Links to TicTec Articles (So Far)

- **Monday (Dec 23):** *When the Going Gets Weird, the Weird Turn Pro* — Big-picture intro to TicTec and the RAG project.
- **Tuesday (Dec 24):** *Don't Panic — Building Your First Local AI System* — Overview of the RAG architecture and your initial environment checklist.
- **Wednesday (Dec 25):** *Forks, Prefects, and GitHub Basics (Updated)* — Setting up GitHub, creating repositories, and version control fundamentals.

> **Note:** We will reference these articles throughout the manual. For example, if you get stuck with GitHub basics, see **Wednesday's** piece for more detailed tips.

Step 1: Setting Up Your Environment

In this step, you'll ensure your local machine is ready to handle the code and tools needed for a RAG system. Before you spin up Docker or integrate Weaviate, you want to be 100% sure you've got the basics down: **Python**, **Git**, and **GitHub**.

1.1 Install & Verify Python

1. **Download Python 3.9+**
    - Visit python.org/downloads and choose the latest stable version (3.9 or higher).
    - On Windows, select “Add Python to PATH” during installation.
    - Mac/Linux users typically follow their OS-specific installation instructions.
2. **Verify Your Installation**

Open a terminal or command prompt and type:
bash
Copy code
```
python --version
```

    -
    - You should see something like `Python 3.9.x`.
3. **(Optional) Create a Test Script**

If you’re new to Python, create a file called `hello.py` with:
python
Copy code
```
print("Hello, TicTec!")
```

    -

Run it with:
bash
Copy code
```
python hello.py
```

    -
    - If it prints “Hello, TicTec!”, you’re golden.

TicTec Tie-In

- **Tuesday (Dec 24) article** introduced why Python is a critical part of building a local AI system and how you’ll use it to tie together retrieval and generation.
- Next week’s TicTec content will dig deeper into Python libraries, but for now, ensure Python runs smoothly.

1.2 Set Up Git & GitHub

GitHub is essential for version control and collaboration in this project—like your mission control center for all code and config files. If you followed **Wednesday’s TicTec article**, you’ve likely done this, but let’s restate the basics:

1. **Install Git** (if not already installed)
    - Visit git-scm.com and download for your OS.

Verify by typing:
bash
Copy code
```
git --version
```

    -
    - You should see something like `git version 2.42.0`.
2. **Create a GitHub Account & Repository**
    - Go to GitHub.com → **Sign Up** (if you haven't already).
    - Create a **new repository** named `local_rag_project` (or similar).
    - Optionally initialize it with a README.
3. **Clone or Initialize Locally**

Option A: Clone the Repo
bash
Copy code
```
git clone https://github.com/<YOUR-USERNAME>/local_rag_project.git

cd local_rag_project
```

    -

Option B: Initialize a Folder Locally
bash
Copy code
```
mkdir local_rag_project

cd local_rag_project

git init
```

Then connect it to GitHub:
bash
Copy code
```
git remote add origin
https://github.com/<YOUR-USERNAME>/local_rag_project.git

git push -u origin main
```

    -
4. **Make Your First Commit**

Add a simple file (e.g., `readme.md` or `hello.py`):
bash
Copy code

```
echo "TicTec RAG Project" > readme.md

git add .

git commit -m "Initial commit"

git push origin main
```

-

TicTec Tie-In

- **Wednesday (Dec 25):** *Forks, Prefects, and GitHub Basics* explains branching, pull requests, and collaboration. If you prefer a UI-based approach, that article walks you through it step-by-step.

1.3 Optional (But Recommended): Virtual Environments

Since you'll be installing Python libraries (like **transformers** and **weaviate-client**), a virtual environment keeps dependencies clean and contained.

Create a Virtual Environment
bash
Copy code

```
python -m venv venv
```

1.
2. **Activate It**
   - **Windows**: `venv\Scripts\activate`
   - **Mac/Linux**: `source venv/bin/activate`
3. **Document It**
   - Add a short line to your `readme.md` or project docs indicating how to activate the venv.
   - Consider adding a `.gitignore` entry for `venv` if you don't want to commit environment files to GitHub.

Why Bother?

- Avoid dependency conflicts (especially if you have multiple Python projects).

- Keep your RAG project self-contained, making it easier to replicate or move to another machine later.

Next Steps: A Teaser for Docker (Step 2)

You've laid the groundwork for your local RAG system. The next big piece is **Docker**, which helps containerize Weaviate (and potentially other services) so everything runs smoothly on your local machine—no more "works on my computer" problems.

- **TicTec will cover Docker** in upcoming articles (Week 2), walking you through installation, commands, and debugging.
- **This Manual** will detail Step 2 (Running Weaviate with Docker) after you're comfortable with the basics in the TicTec Docker week.

  Sneak Peek
  In Step 2, you'll:

  - Pull the Weaviate Docker image
  - Run it locally on port 8080
  - Possibly experiment with `docker-compose.yml` to run multiple containers

Conclusion (Up to Docker)

Congratulations on completing **Step 1** of the *RAG_Local_HF_Weaviate_v3* manual! You have:

1. **Installed Python** and can run scripts locally.
2. **Set Up Git & GitHub** so you can track changes and collaborate.
3. (Optionally) **Created a Virtual Environment** for neat, conflict-free library installs.

When you're ready for more, we'll delve into **Docker** (Step 2) and how to containerize Weaviate. Until then, keep an eye on the next TicTec articles to reinforce these steps with real-world scenarios and hands-on GitHub workflows.

Quick Reference

- **Git Commands**
  - `git clone <URL>`
  - `git status`

    - `git add .`
    - `git commit -m "Message"`
    - `git push origin main`
- **Python Basics**
    - `python --version`
    - `python -m venv venv`
    - `pip install -r requirements.txt`
- **Where to Learn More**
    - **TicTec Monday & Tuesday**: Big-picture overview of local RAG systems.
    - **TicTec Wednesday**: GitHub fundamentals, branching, and collaboration.
    - **TicTec Next**: Thursday & Friday deeper GitHub workflows, then into Docker next week.

End of “Up to Docker” Section
*(We’ll update again once we’re ready to integrate Docker specifics in **Step 2**.)*