Lab Activity: Hosting an App on Streamlit

Introduction

Streamlit is a powerful framework for creating web applications with Python. This lab activity will, you will go through the process of hosting an Information Retrieval (IR) app using document embeddings on Streamlit. The app will allow users to enter a query and retrieve the top K most relevant documents.

Prerequisites

Before starting, ensure you have:

- Python installed (Python 3.7+ recommended)
- `pip` installed for package management
- Precomputed document embeddings stored as a NumPy array
- A text-based dataset with corresponding documents

Step 1: Install Required Libraries

To get started, install Streamlit and other necessary dependencies:

```
pip install streamlit numpy sklearn
```

Step 2: Load Document Embeddings and Define a Similarity Function

Create a Python script `app.py` and add the following code to load the document embeddings and define a similarity function:

```python
import streamlit as st
import numpy as np
from sklearn.metrics.pairwise import cosine_similarity

# Load precomputed document embeddings (Assuming embeddings.npy and documents.txt exist)
embeddings = np.load("embeddings.npy")
with open("documents.txt", "r", encoding="utf-8") as f:
    documents = f.readlines()

def retrieve_top_k(query_embedding, embeddings, k=10):
    """Retrieve top-k most similar documents using cosine similarity."""
    similarities = cosine_similarity(query_embedding.reshape(1, -1), embeddings)[0]
    top_k_indices = similarities.argsort()[-k:][::-1]
    return [(documents[i], similarities[i]) for i in top_k_indices]
```

Step 3: Create the Streamlit App Interface

Now, define the Streamlit UI components:

```
# Streamlit UI
st.title("Information Retrieval using Document Embeddings")

# Input query
query = st.text_input("Enter your query:")

# Load or compute query embedding (Placeholder: Replace with actual embedding
model)
def get_query_embedding(query):
    return np.random.rand(embeddings.shape[1])  # Replace with actual embedding
function

if st.button("Search"):
    query_embedding = get_query_embedding(query)
    results = retrieve_top_k(query_embedding, embeddings)

    # Display results
    st.write("### Top 10 Relevant Documents:")
    for doc, score in results:
        st.write(f"- **{doc.strip()}** (Score: {score:.4f})")
```

Step 4: Run the Streamlit App

To run your application locally, use the command:

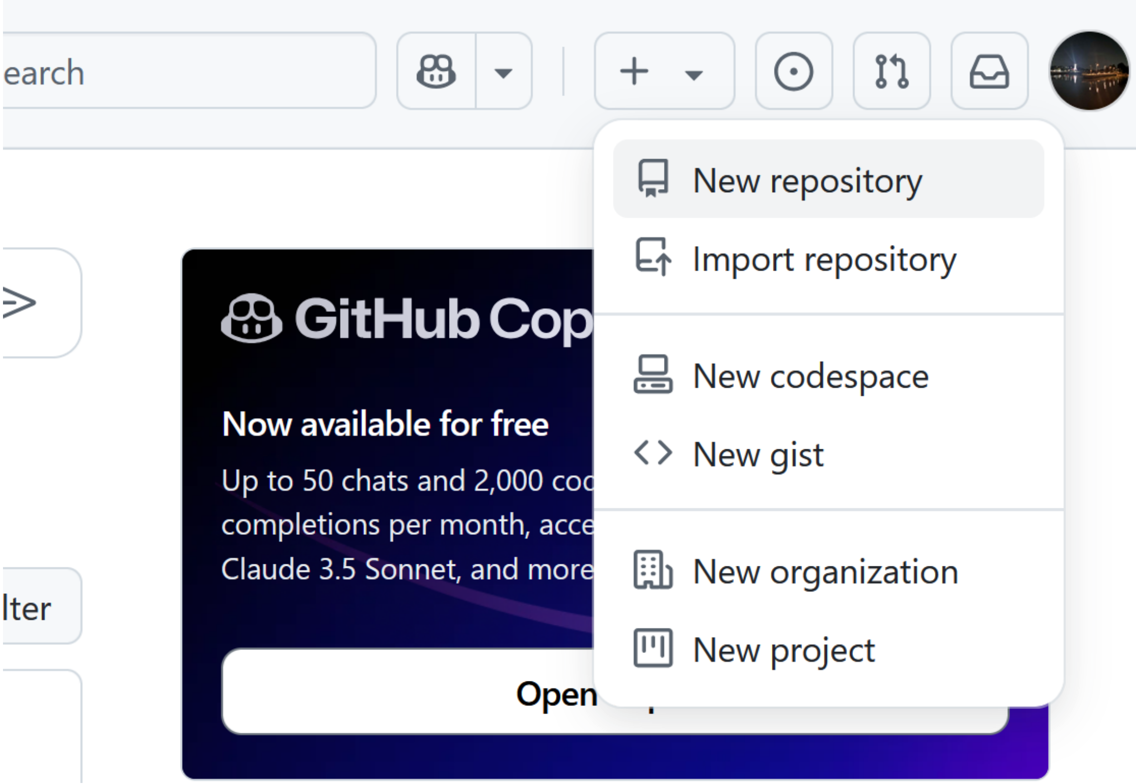
```
streamlit run app.py
```

This will launch a local web server, and you can access the app in your browser at `http://localhost:8501`.

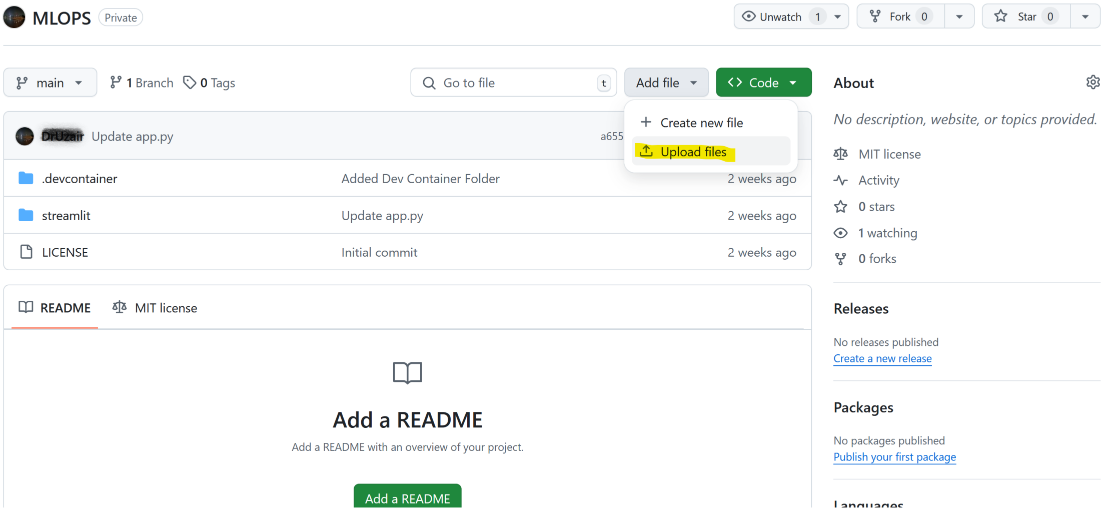
Step 5: Deploying the Streamlit App

Using Streamlit Cloud

1. Create and account on GitHub and create a repository.



2. Upload your code, embeddings file, and documents to the repository.



No description, website, or topics provided.

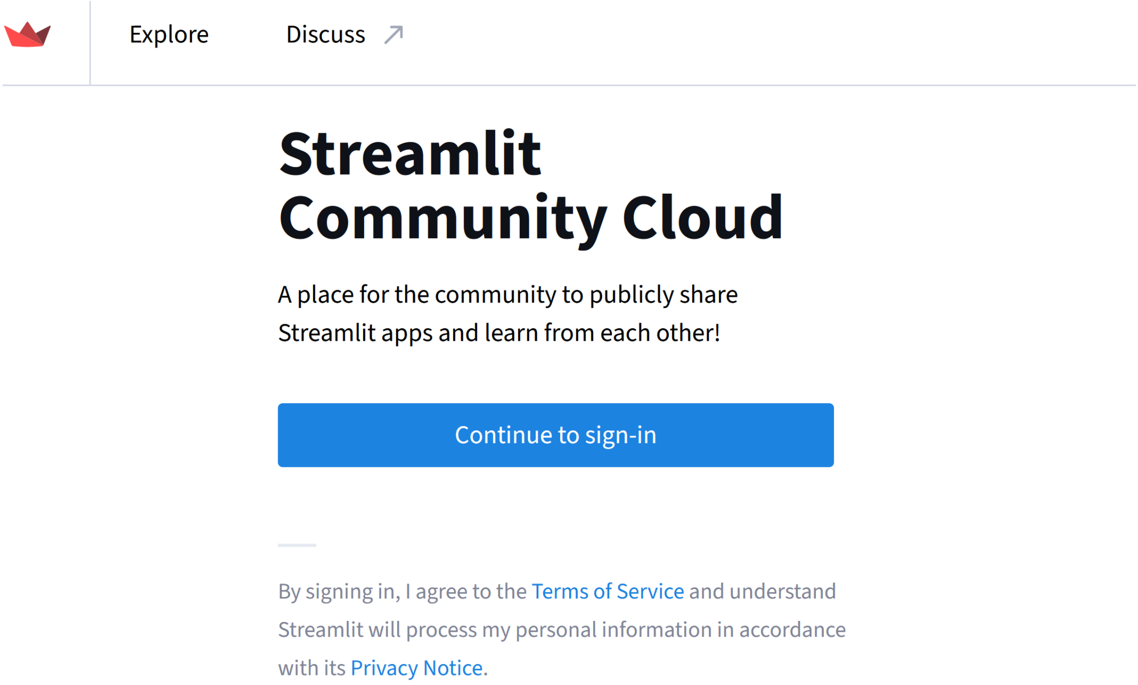
No releases published

No packages published

[redacted] Update app.py		a655
.devcontainer	Added Dev Container Folder	2 weeks ago
streamlit	Update app.py	2 weeks ago
LICENSE	Initial commit	2 weeks ago

Add a README with an overview of your project.

3. Go to Streamlit Cloud , create and account and log in.



Streamlit
Community Cloud

A place for the community to publicly share Streamlit apps and learn from each other!

By signing in, I agree to the Terms of Service and understand Streamlit will process my personal information in accordance with its Privacy Notice.

4. Create a new app, link it to your GitHub repository, and deploy.
5. Test the functionality. Make changes to your code if necessary and update the code in GitHub repository.

Assignment Task

Fine tune your Information Retrieval App and submit the public URL of deployed app.