Bhuvi Ghosh
60009210191

ACL Miniproject

```
[2]  !pip install colab-xterm
     %load_ext colabxterm
```

```
Collecting colab-xterm
  Downloading colab_xterm-0.2.0-py3-none-any.whl.metadata (1.2 kB)
Requirement already satisfied: ptyprocess~=0.7.0 in /usr/local/lib/python3.10/dist-
Requirement already satisfied: tornado>5.1 in /usr/local/lib/python3.10/dist-packag
Downloading colab_xterm-0.2.0-py3-none-any.whl (115 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 115.6/115.6 kB 2.9 MB/s eta 0:00:00
Installing collected packages: colab-xterm
Successfully installed colab-xterm-0.2.0
```

```
[ ]  # type these codes in below terminal after run the cell (%xterm)
     #curl -fsSL https://ollama.com/install.sh | sh
     # ollama serve & ollama pull llama3 & ollama pull nomic-embed-text
```

```
%xterm
```

```
Launching Xterm...
```

```
[ ]  !pip -qq install langchain
     !pip -qq install langchain-core
     !pip -qq install langchain-community
```

```
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.4/2.4 MB 77.6 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.1/3.1 MB 50.0 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.5/49.5 kB 4.4 MB/s eta 0:00:00
```

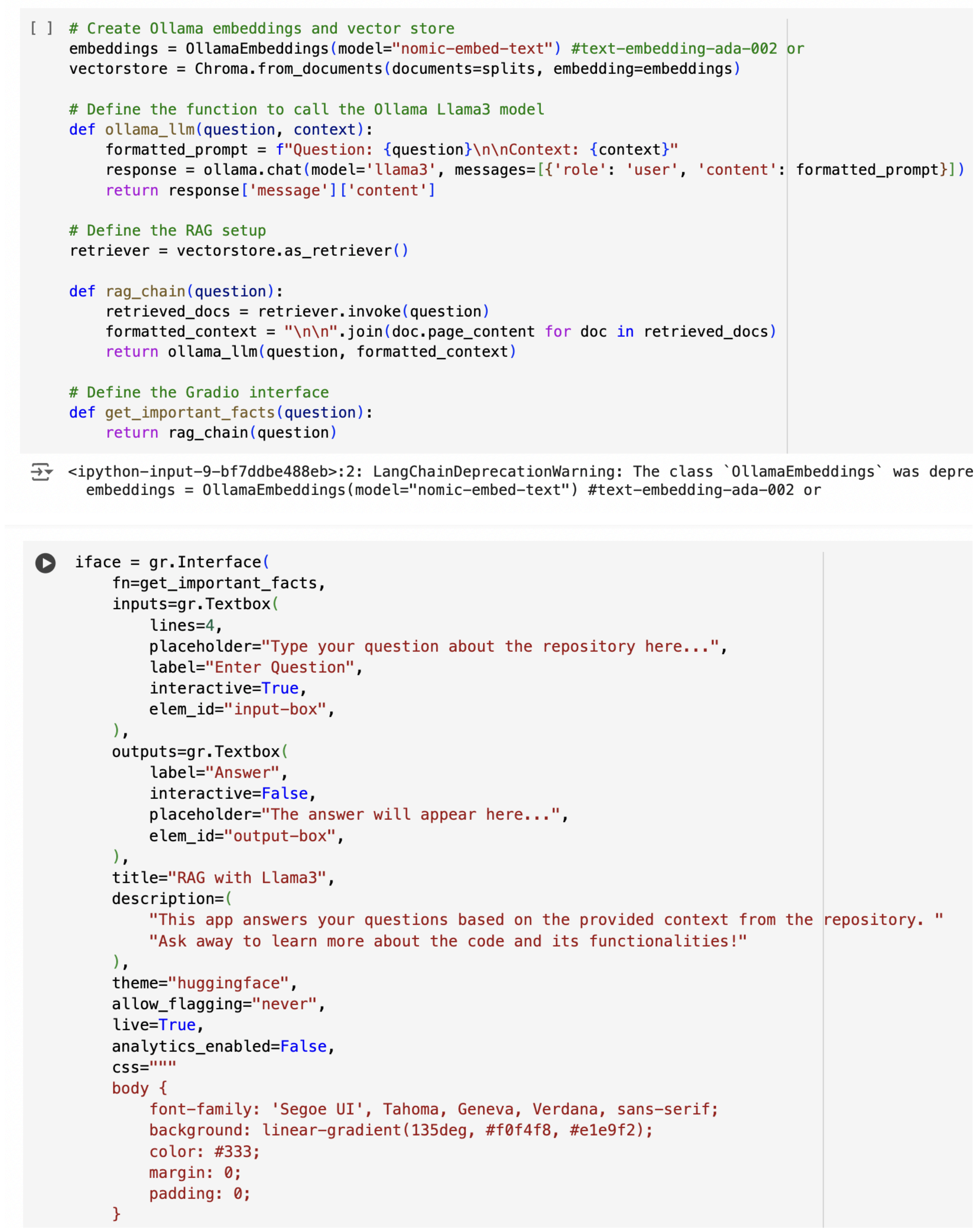
```
[ ]  from langchain_community.llms import Ollama
```

```
!pip install ollama langchain beautifulsoup4 chromadb gradio -q
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 67.3/67.3 kB 5.2 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 617.9/617.9 kB 39.6 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.4/2.4 MB 16.5 MB/s eta 0:00:00
```

```
[ ]  import gradio as gr
     import ollama
     from bs4 import BeautifulSoup as bs
     from langchain.text_splitter import RecursiveCharacterTextSplitter
     from langchain_community.document_loaders import WebBaseLoader
     from langchain_community.vectorstores import Chroma
     from langchain_community.embeddings import OllamaEmbeddings

     # Load the data from the web URL
     url ='https://github.com/bhuvighosh3'
     loader = WebBaseLoader(url)
     docs = loader.load()

     # Split the loaded documents into chunks
     text_splitter = RecursiveCharacterTextSplitter(chunk_size=1000, chunk_overlap=200)
     splits = text_splitter.split_documents(docs)
```

```
WARNING:langchain_community.utils.user_agent:USER_AGENT environment variable not set, consider setting it to identify your requests.
```

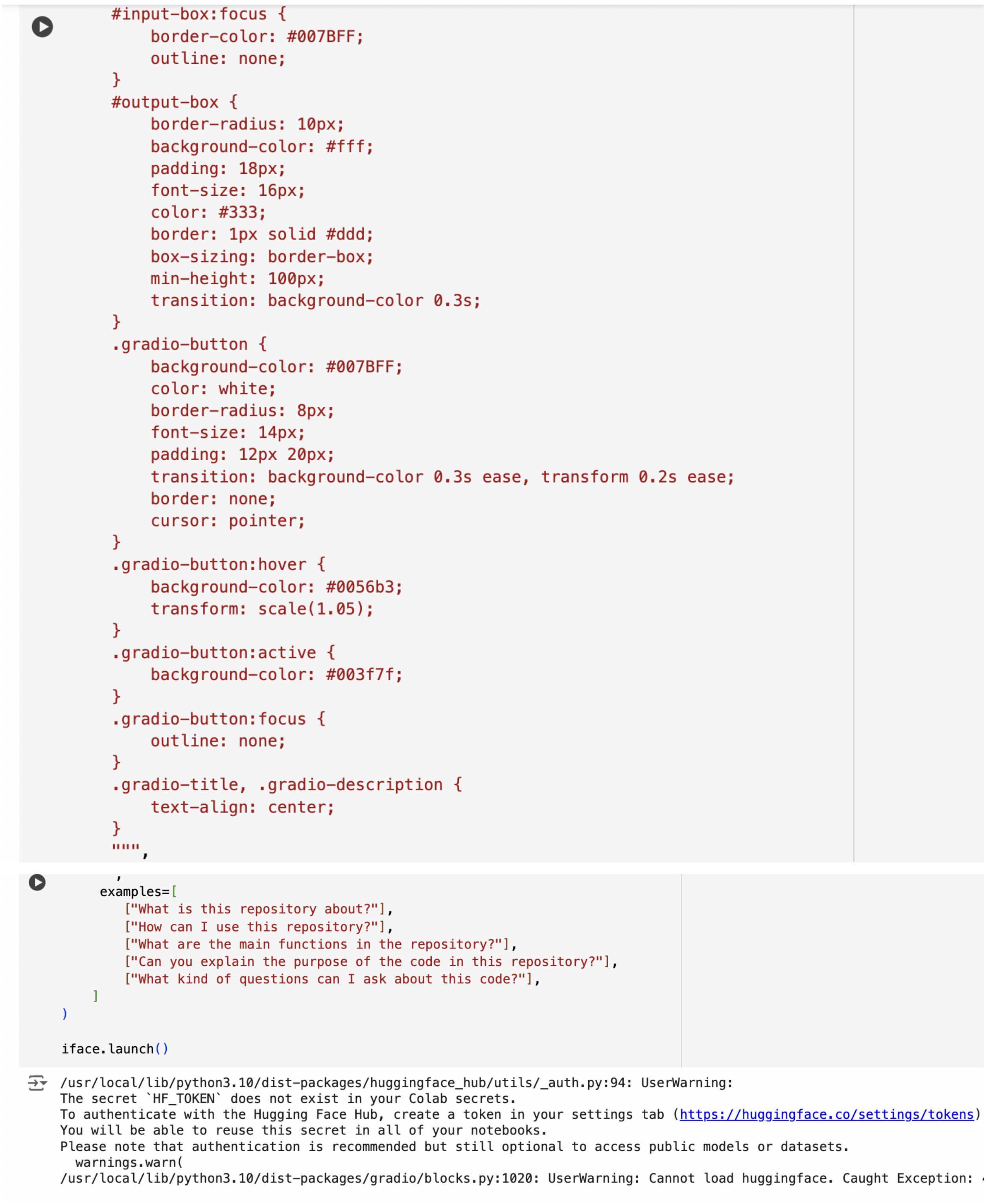


```
[ ] # Create Ollama embeddings and vector store
    embeddings = OllamaEmbeddings(model="nomic-embed-text") #text-embedding-ada-002 or
    vectorstore = Chroma.from_documents(documents=splits, embedding=embeddings)

    # Define the function to call the Ollama Llama3 model
    def ollama_llm(question, context):
        formatted_prompt = f"Question: {question}\n\nContext: {context}"
        response = ollama.chat(model='llama3', messages=[{'role': 'user', 'content': formatted_prompt}])
        return response['message']['content']

    # Define the RAG setup
    retriever = vectorstore.as_retriever()

    def rag_chain(question):
        retrieved_docs = retriever.invoke(question)
        formatted_context = "\n\n".join(doc.page_content for doc in retrieved_docs)
        return ollama_llm(question, formatted_context)

    # Define the Gradio interface
    def get_important_facts(question):
        return rag_chain(question)
```

```
<ipython-input-9-bf7ddbe488eb>:2: LangChainDeprecationWarning: The class `OllamaEmbeddings` was depre
  embeddings = OllamaEmbeddings(model="nomic-embed-text") #text-embedding-ada-002 or
```

```
iface = gr.Interface(
    fn=get_important_facts,
    inputs=gr.Textbox(
        lines=4,
        placeholder="Type your question about the repository here...",
        label="Enter Question",
        interactive=True,
        elem_id="input-box",
    ),
    outputs=gr.Textbox(
        label="Answer",
        interactive=False,
        placeholder="The answer will appear here...",
        elem_id="output-box",
    ),
    title="RAG with Llama3",
    description=(
        "This app answers your questions based on the provided context from the repository. "
        "Ask away to learn more about the code and its functionalities!"
    ),
    theme="huggingface",
    allow_flagging="never",
    live=True,
    analytics_enabled=False,
    css="""
    body {
        font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
        background: linear-gradient(135deg, #f0f4f8, #e1e9f2);
        color: #333;
        margin: 0;
        padding: 0;
    }
```

```
        .gradio-container {
            background-color: #f7f7f7;
            border-radius: 15px;
            padding: 20px;
            max-width: 600px;
            margin: auto;
            box-shadow: 0 8px 24px rgba(0, 0, 0, 0.1);
        }
        .gradio-interface {
            background-color: #ffffff;
            border-radius: 15px;
            padding: 30px;
            box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);
            transition: all 0.3s ease-in-out;
        }
        .gradio-interface:hover {
            box-shadow: 0 6px 18px rgba(0, 0, 0, 0.12);
        }
        .gradio-title {
            font-size: 28px;
            font-weight: 700;
            color: #2C3E50;
            margin-bottom: 10px;
        }
        .gradio-description {
            font-size: 18px;
            color: #7F8C8D;
            margin-bottom: 20px;
        }
        #input-box {
            border-radius: 10px;
            background-color: #fafafa;
            font-size: 16px;
            padding: 18px;
            width: 100%;
            border: 1px solid #ddd;
            box-sizing: border-box;
            margin-bottom: 20px;
            transition: border-color 0.3s;
        }
```



```
        #input-box:focus {
            border-color: #007BFF;
            outline: none;
        }
        #output-box {
            border-radius: 10px;
            background-color: #fff;
            padding: 18px;
            font-size: 16px;
            color: #333;
            border: 1px solid #ddd;
            box-sizing: border-box;
            min-height: 100px;
            transition: background-color 0.3s;
        }
        .gradio-button {
            background-color: #007BFF;
            color: white;
            border-radius: 8px;
            font-size: 14px;
            padding: 12px 20px;
            transition: background-color 0.3s ease, transform 0.2s ease;
            border: none;
            cursor: pointer;
        }
        .gradio-button:hover {
            background-color: #0056b3;
            transform: scale(1.05);
        }
        .gradio-button:active {
            background-color: #003f7f;
        }
        .gradio-button:focus {
            outline: none;
        }
        .gradio-title, .gradio-description {
            text-align: center;
        }
        """,
```

```
        ,
        examples=[
            ["What is this repository about?"],
            ["How can I use this repository?"],
            ["What are the main functions in the repository?"],
            ["Can you explain the purpose of the code in this repository?"],
            ["What kind of questions can I ask about this code?"],
        ]
)

iface.launch()
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens)
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/gradio/blocks.py:1020: UserWarning: Cannot load huggingface. Caught Exception:
```

BLEU-2 Test Cases: Abstractive vs Extractive

Who does this account belong to?

```
from nltk.translate.bleu_score import sentence_bleu, SmoothingFunction

def compute_bleu_2(reference_sentence, candidate_sentence):
    reference = [reference_sentence.split()]
    candidate = candidate_sentence.split()
    smooth = SmoothingFunction().method1
    return sentence_bleu(reference, candidate, weights=(0.5, 0.5, 0, 0), smoothing_function=smooth)

reference_sentence = input("Enter the reference sentence: ")
candidate_sentence = input("Enter the candidate sentence: ")

bleu_2_score = compute_bleu_2(reference_sentence, candidate_sentence)
print(f"BLEU-2 Score: {bleu_2_score:.4f}")
```

```
Enter the reference sentence: Bhuvi Ghosh
Enter the candidate sentence: Bhuvi Ghosh
BLEU-2 Score: 1.0000
```

Who is the owner of this account?

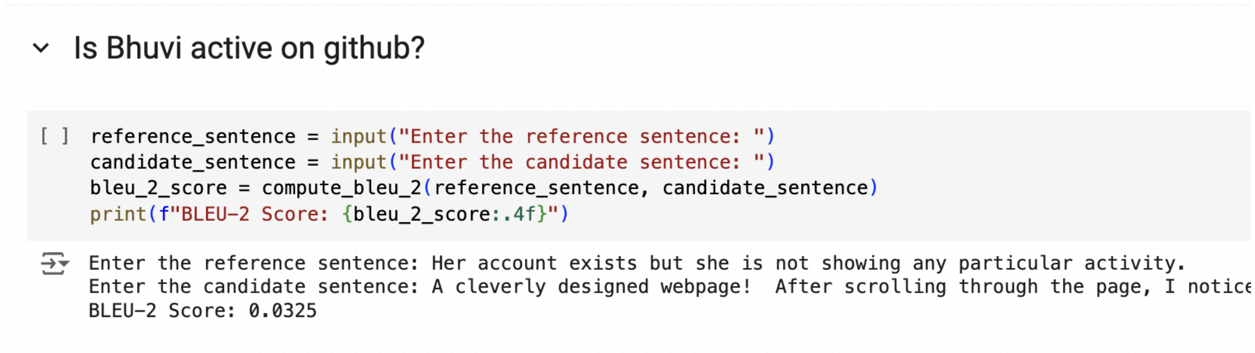
```
reference_sentence = input("Enter the reference sentence: ")
candidate_sentence = input("Enter the candidate sentence: ")
bleu_2_score = compute_bleu_2(reference_sentence, candidate_sentence)
print(f"BLEU-2 Score: {bleu_2_score:.4f}")
```

```
Enter the reference sentence: Bhuvi Ghosh
Enter the candidate sentence: Bhuvi Ghosh
BLEU-2 Score: 1.0000
```

How many people does the owner of the account follow?

```
reference_sentence = input("Enter the reference sentence: ")
candidate_sentence = input("Enter the candidate sentence: ")
bleu_2_score = compute_bleu_2(reference_sentence, candidate_sentence)
print(f"BLEU-2 Score: {bleu_2_score:.4f}")
```

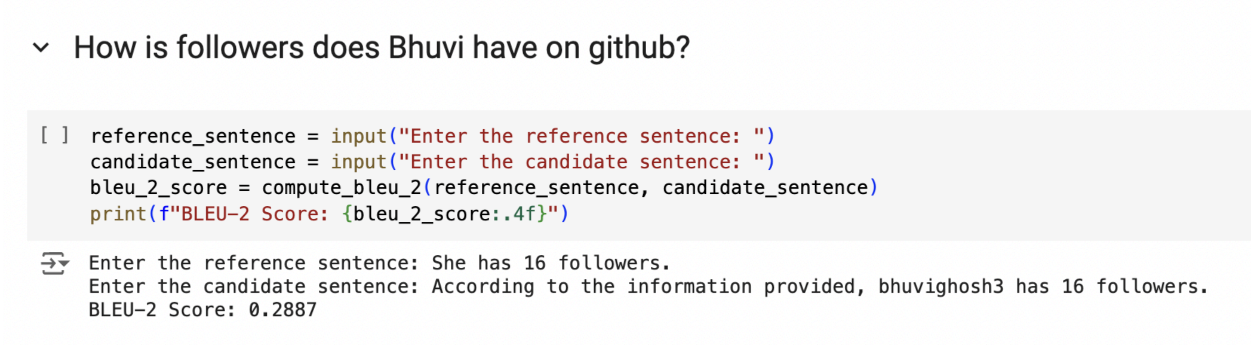
```
Enter the reference sentence: She is following 14 profiles
Enter the candidate sentence: According to the context, Bhuvi is following 14 people.
BLEU-2 Score: 0.2887
```



Is Bhuvi active on github?

```
[ ] reference_sentence = input("Enter the reference sentence: ")
    candidate_sentence = input("Enter the candidate sentence: ")
    bleu_2_score = compute_bleu_2(reference_sentence, candidate_sentence)
    print(f"BLEU-2 Score: {bleu_2_score:.4f}")
```

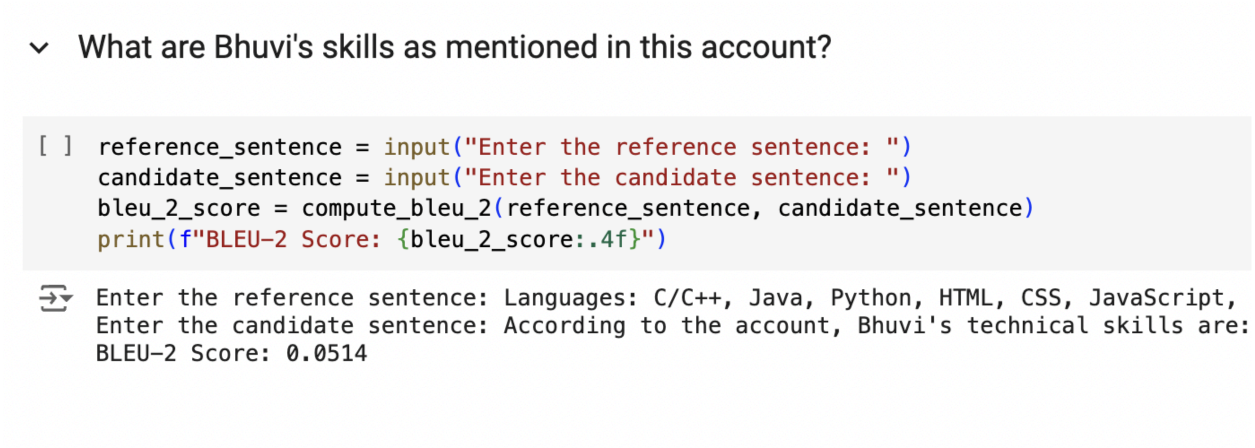
```
Enter the reference sentence: Her account exists but she is not showing any particular activity.
Enter the candidate sentence: A cleverly designed webpage!  After scrolling through the page, I notice
BLEU-2 Score: 0.0325
```



How is followers does Bhuvi have on github?

```
[ ] reference_sentence = input("Enter the reference sentence: ")
    candidate_sentence = input("Enter the candidate sentence: ")
    bleu_2_score = compute_bleu_2(reference_sentence, candidate_sentence)
    print(f"BLEU-2 Score: {bleu_2_score:.4f}")
```

```
Enter the reference sentence: She has 16 followers.
Enter the candidate sentence: According to the information provided, bhuvighosh3 has 16 followers.
BLEU-2 Score: 0.2887
```



What are Bhuvi's skills as mentioned in this account?

```
[ ] reference_sentence = input("Enter the reference sentence: ")
    candidate_sentence = input("Enter the candidate sentence: ")
    bleu_2_score = compute_bleu_2(reference_sentence, candidate_sentence)
    print(f"BLEU-2 Score: {bleu_2_score:.4f}")
```

```
Enter the reference sentence: Languages: C/C++, Java, Python, HTML, CSS, JavaScript,
Enter the candidate sentence: According to the account, Bhuvi's technical skills are:
BLEU-2 Score: 0.0514
```