Lebanese American University

Department of Computer Science & Mathematics

CSC491- Professional Experience

Internship Report

Student Name: Samir Khansa

Student ID Number: 202001781

Instructor: Dr. Ramzi Haraty

Company: Comprehensive Computing Innovations- CCI

Table of Contents

1. Introduction

Comprehensive Computing Innovations (CCI) is a company that is part of the ICC group and was founded in 2015, in which its main department is in Beirut Lebanon and has offices in multiple countries including KSA, Qatar, UAE, Kuwait, and Oman. CCI provides IT solutions for organizations of different sizes and industries. Those solutions are used to help organizations in dealing with complex IT a business requirements. CCI also provides next-generation solutions which include Artificial Intelligence (AI).

A relative had informed me about an AI internship that is in Beirut, Lebanon. After applying for that position, I was able to have two interviews. The first interview was an HR interview that lasted 45 minutes. Furthermore, Five weeks later, I had a technical interview where I was asked multiple questions concerning what I learned in the university, what programming languages I worked on, and what are my previous experience in working with Artificial Intelligence. Few days after the interview, I received an email getting informed that I got accepted into their AI internship program. Moreover, when I received the acceptance email, I was required to bring three documents "a colored copy of my ID, juridical record, and recommendation letter." throughout my internship program my tasks was to create a fully functional AI application, learning about Microsoft Azure and its services such as Azure Cognitive Search, Azure Form Recognizer, and Azure Bot Framework Composer. Moreover, I also have learned about

Machine Learning, Deep Learning, and Artificial Intelligence, how to deal with unstructured data, prompt engineering APIs and Frontend development (HTML, CSS, JavaScript, Streamlit).

This internship provided me with a chance to increase my skills and gaining a significant amount of knowledge in the field of AI without having any previous knowledge. I learned multiple techniques, programming languages, and frameworks that helped in advancing my skills and capabilities in the field of Artificial Intelligence. Moreover, I worked for 411 hours during my internship program, which started on June 12, 2023, and ended on August 31, 2023.

2. <u>**Week 1-2: Introduction to frontend development**</u>

At the beginning of the internship program I started to learn about fronted development, where I learned CSS, HTML, JavaScript and React. These are the languages that are used in web development to help in creating a fully functional website. HTML is a programming language that helps in describing the structure of a Web page. CSS is a programming language that is used to give styles and layout of the webpage. In other words, CSS can describe how each HTML element should be viewed on the websites screen. JavaScript is a programming language that would allow programmers to add more functionality, interactivity and behavior to websites. Finally, a JavaScript library called React allows developers to create an interactive and dynamic web applications. In other words, it would allow the developer to create reusable user interface components and efficiently update it.

I was learning frontend development from scratch since I hadn't yet taken the web development course in the university, therefore I had no previous knowledge of frontend development, which was challenging for me since I had to learn frontend development as fast as possible. Moreover, through my second week, I started to learn React, however, I realized immediately that learning React is much more difficult than learning HTML, CSS, and JavaScript since I had no previous knowledge and the syntax is not as easy to understand as frontend development. Furthermore, I also learned how to connect my frontend code with my backend code where I was using HTTP requests to send data and receive responses between my frontend code and my backend code. Additionally, the employees who are working at CCI gave me their frontend code and they started giving me tasks on that code. For example, I added an “Add Document” button in which when the user adds a document, the document is automatically added to their cloud database. In this example I learned how to deal with files using the programming language React, and since I didn’t have any background knowledge of frontend development the first two weeks were challenging since the employees preferred for me to discover and solve my mistakes instead of them telling or helping me in were my mistakes are.

3. <u>Week 3: Introduction to Microsoft Azure</u>

At the beginning of week 3, I started to first research what is Artificial Intelligence (AI). AI refers to the theory that a computer systems is capable of performing tasks that typically require human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages. Moreover, it involves creating systems with the ability to reason, discover meaning, generalize, or learn from past experiences.

Moreover, after researching what Artificial Intelligence is I researched what is Azure Microsoft and its importance? Microsoft Azure is a cloud computing platform and service offered by Microsoft. It provides multiple cloud services that includes computing, analytics, storage, and networking. Moreover, Azure also allows organizations to build, deploy, and manage applications and services through Microsoft data centers. Similarly, Azure Microsoft offers a variety of tools and frameworks for developers to create multiple applications and services. There are 200 products and services that Azure Microsoft provides, however the services that I researched during the third week of the internship program included:

1. Azure Machine Learning, is a service provided by Microsoft Azure that allows data scientists and developers to build, deploy, and manage high-quality machine learning models. This service provides training and deploying models, which is a powerful AI infrastructure to build and train machine learning models.
2. Azure Bot Services, is a Software application that provides comprehensive tools and features for building conversational AI bots. It offers a modular set of developer tools to build, test, deploy, and manage Chabot using C# or JavaScript. Using this software we

can create Chabot's with an easy graphical interface without requiring coding or AI expertise.

3. Azure Databricks, is a unified, open analytics platform for building, deploying, sharing, and maintaining enterprise-grade data, analytics, and AI solutions at scale. It integrates with cloud storage and security in the cloud account and manages and deploys cloud infrastructure on our behalf.
4. Azure Cognitive Search, is a cloud-based search service that enables developers to add search capabilities to their applications without the need for complex infrastructure or search expertise. It provides an AI approach to content understanding by using built-in cognitive skills to extract knowledge from unstructured data.
5. Azure Form Recognizer is an AI service provided by Microsoft Azure that applies advanced machine learning to extract text, key-value pairs, tables, and structures from documents automatically and accurately. It allows us to turn documents/pictures…. into usable data and shift our focus to acting on information rather than compiling it.
6. Semantic Configuration, is a feature in Azure Cognitive Search where it ranks the search results based on the meaning or interpretation of a word, phrase, sentence, or text.

In conclusion, during the third week of my internship program, I was mostly doing research on what is Artificial Intelligence, The importance of Microsoft Azure and what are some of the services that Microsoft Azure provides including Azure Machine Learning, Azure Bot Services, Azure Databricks, Azure Cognitive Search and Azure Form Recognizer.

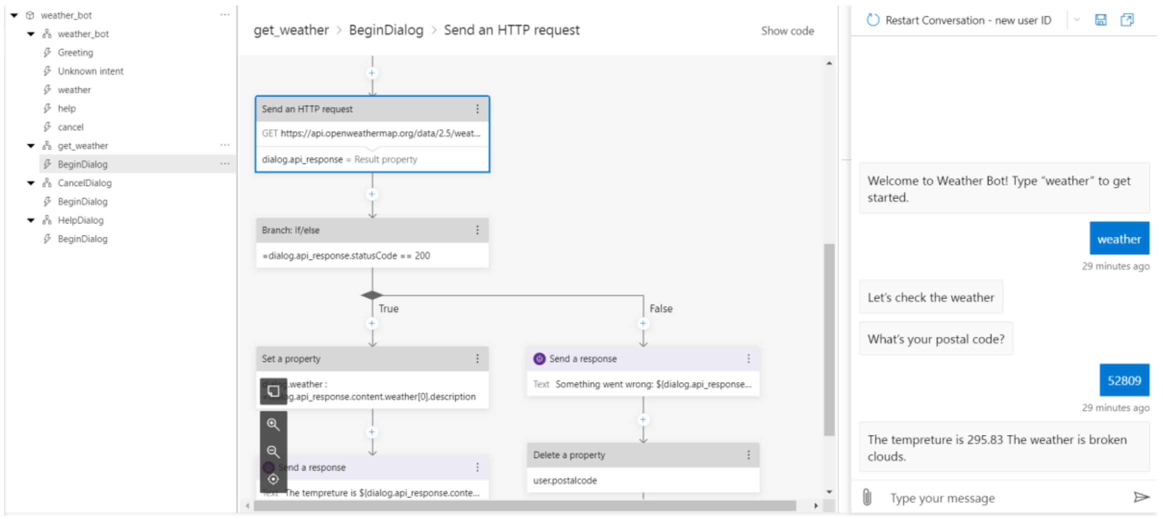
4. <u>**Week 4: Azure bot framework composer, API**</u>

In week 4 I downloaded a software called Azure bot framework composer and this software as I explained before is an easy tool that helps developers in building Chabot's using the language C#. In other words, it helps developers build a bot without needing any previous knowledge in the fields of C#, Machine learning, or AI. When I first began working with this software I was having difficulties in understanding how it works since there is only the documentation of Azure bot framework composer concerning how to use this software. However, after a few time trying to understand how to use this software, I learned a few vocabularies that I will be using throughout this week.

1. **Dialog:** A Dialog is a conversational flow that guides the user through a series of steps to accomplish a specific task.

2. **Trigger:** Each Dialog will execute when it gets triggered either by the user or when the code reaches a specific place in the dialog. There are different kinds of triggers. The type of trigger that I used in my bot was the **Intent recognized**. This trigger is launched when the bot recognizes a user's intent based on their input.
3. **Application Programming Interface (API):** APIs are rules that are defined by each application separately to allow multiple applications to communicate with each other.

After a day of researching and understanding how to use this software, I was able to build my first bot which was a weather bot. This bot would begin in the greetings dialog which only has a single response that is "Welcome to Weather Bot! Type weather to get started". If the user enters something other than "Weather", a trigger would get activated which is called "Unknown intent" and this trigger would reply to the user "Please type Weather to get started". After the user types weather, the trigger called weather begins a new dialog called get_weather in which the bot asks what the postal code for your geographic area is. After the user enters the postal code I used the API of a website called https://openweathermap.org/ in which I put the postal code that the user entered in the API and I would get in Jason format all the information that the website has in this geographic region.


get_weather > BeginDialog > Send an HTTP request

Send an HTTP request
GET https://api.openweathermap.org/data/2.5/weat...
dialog.api_response = Result property

Branch: If/else
=dialog.api_response.statusCode == 200

True:
Set a property
dialog.weather : =dialog.api_response.content.weather[0].description
Send a response
Text The tempreture is ${dialog.api_response.conte...

False:
Send a response
Text Something went wrong: ${dialog.api_response...
Delete a property
user.postalcode

Welcome to Weather Bot! Type "weather" to get started.
weather
29 minutes ago
Let's check the weather
What's your postal code?
52809
29 minutes ago
The tempreture is 295.83 The weather is broken clouds.

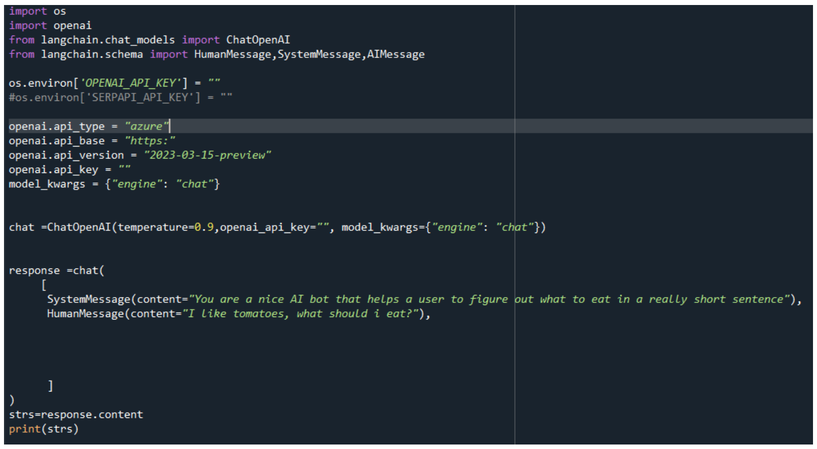
After creating the weather bot I created a Calculator bot and I also created a bot that would connect to the AI that the company CCI is using. Azure bot framework composer is a new software and there are not a lot of people who know how to use this software. In conclusion, Azure bot framework composer is a helpful interface that helps developers create their own bots without the need for any knowledge in the programming language C#. However, if the developer wants to change or look at the code he is able to alter the code, which is useful since it helps developers in understanding how the code is being done.

5. **Week 5-6: LangChain, OpenAI**

When I was taking the course Software Engineering at the University of LAU (Lebanese American University), I learned that software reuse is the finest way to construct a new software system. Developers think about how they can assemble their software from pre-existing software components and systems instead of starting from scratch. Moreover, the employees at the company CCI are developing their AI website based on another software which is ChatGPT. According to them they already have a subscription with Microsoft and since Microsoft purchased OpenAI in 2019, it would be cheaper for them to use this software instead of building a new software that is similar to ChatGPT. OpenAI is an organization that focuses on creating safe and beneficial artificial general intelligence (AGI), the developers who work at this organization are responsible for building ChatGPT. Hence, I was able to download a library called OpenAI which was used in my Python backend code.

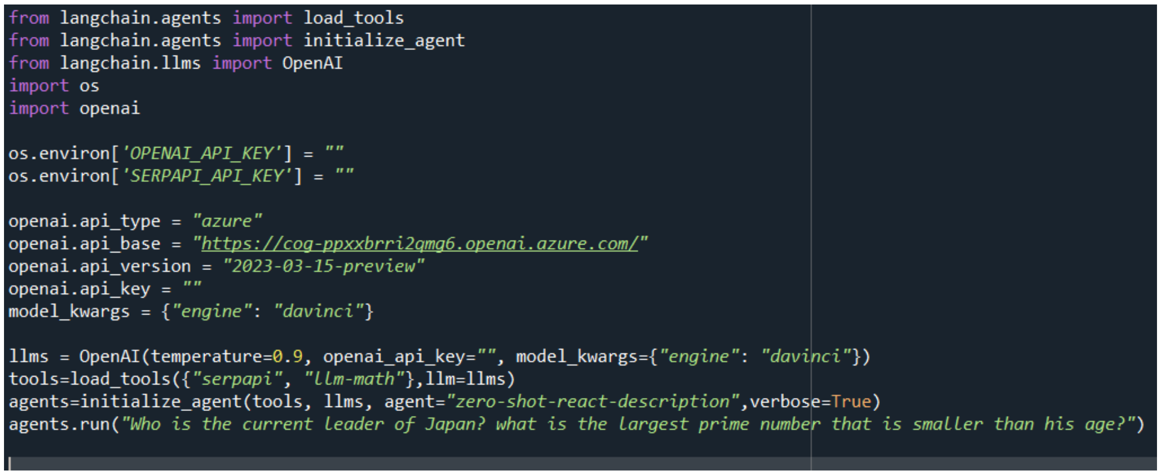
Furthermore, LangChain is a framework for developing applications powered by language models, it enables data-aware applications, allowing a language model to connect to other sources of data, hence enabling it to interact with its environment. Therefore, I also used this library in my backend code.

Using the libraries OpenAI and LangChain, I was able to have a conversation using ChatGPT.



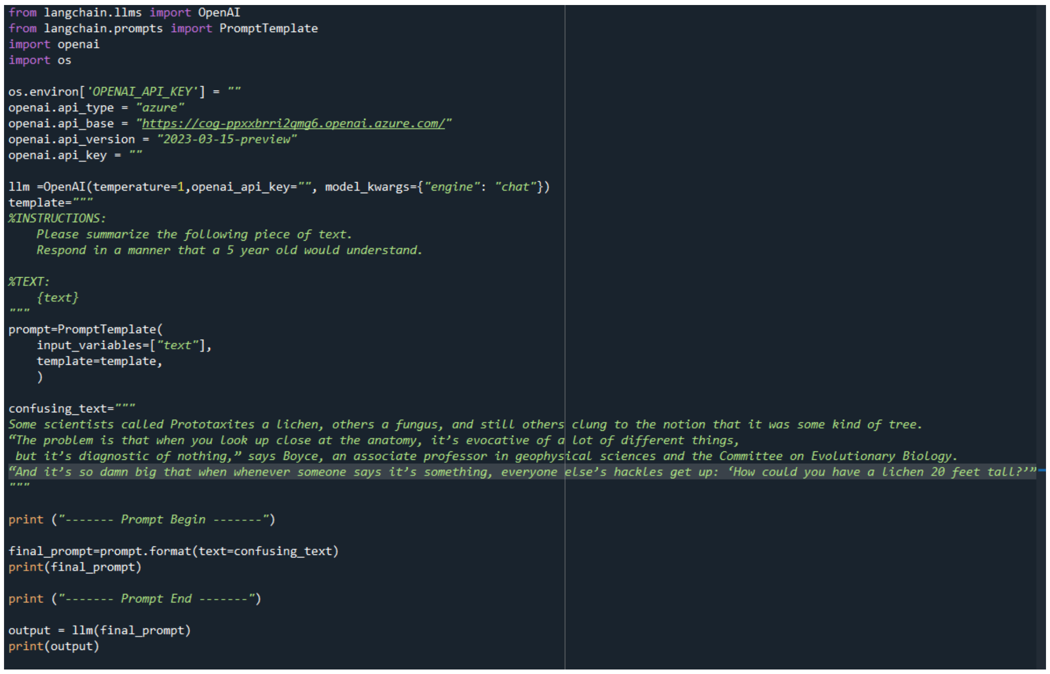
```python
import os
import openai
from langchain.chat_models import ChatOpenAI
from langchain.schema import HumanMessage,SystemMessage,AIMessage

os.environ['OPENAI_API_KEY'] = ""
#os.environ['SERPAPI_API_KEY'] = ""

openai.api_type = "azure"
openai.api_base = "https:"
openai.api_version = "2023-03-15-preview"
openai.api_key = ""
model_kwargs = {"engine": "chat"}


chat =ChatOpenAI(temperature=0.9,openai_api_key="", model_kwargs={"engine": "chat"})


response =chat(
     [
      SystemMessage(content="You are a nice AI bot that helps a user to figure out what to eat in a really short sentence"),
      HumanMessage(content="I like tomatoes, what should i eat?"),




      ]
)
strs=response.content
print(strs)
```

I was also able to go to connect to google using the tool Serpapi, the user would ask questions and he would receive an answer from google.



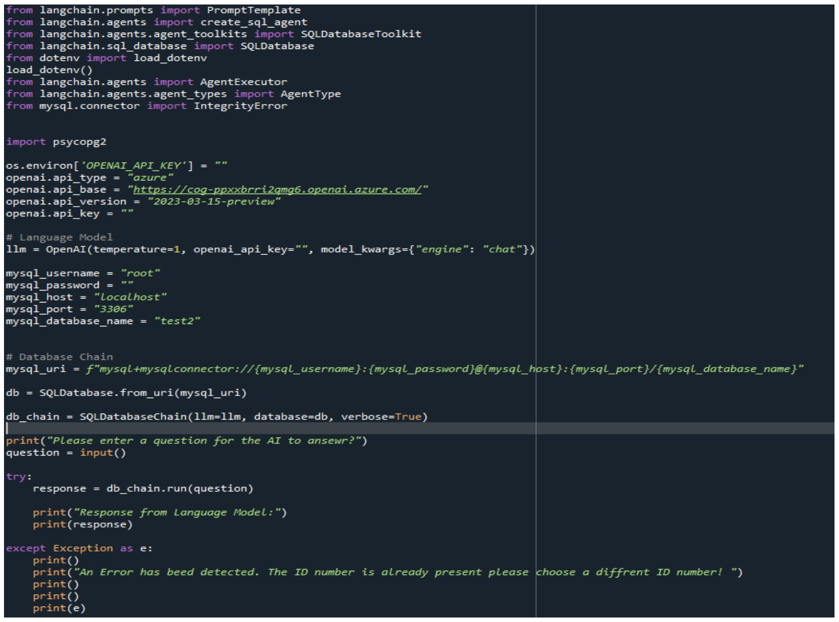
```python
from langchain.agents import load_tools
from langchain.agents import initialize_agent
from langchain.llms import OpenAI
import os
import openai

os.environ['OPENAI_API_KEY'] = ""
os.environ['SERPAPI_API_KEY'] = ""

openai.api_type = "azure"
openai.api_base = "https://cog-ppxxbrri2qmg6.openai.azure.com/"
openai.api_version = "2023-03-15-preview"
openai.api_key = ""
model_kwargs = {"engine": "davinci"}

llms = OpenAI(temperature=0.9, openai_api_key="", model_kwargs={"engine": "davinci"})
tools=load_tools({"serpapi", "llm-math"},llm=llms)
agents=initialize_agent(tools, llms, agent="zero-shot-react-description",verbose=True)
agents.run("Who is the current leader of Japan? what is the largest prime number that is smaller than his age?")
```

I also learned and wrote multiple prompts according to how I wanted the AI to answer. Moreover, I was also able to use prompts to inform the AI to only answer questions from the document that is provided and if there is a question that is not in the provided in the document, the AI would apologize and inform the user that he does not know the answer.



```python
from langchain.llms import OpenAI
from langchain.prompts import PromptTemplate
import openai
import os

os.environ['OPENAI_API_KEY'] = ""
openai.api_type = "azure"
openai.api_base = "https://cog-ppxxbrri2qmg6.openai.azure.com/"
openai.api_version = "2023-03-15-preview"
openai.api_key = ""

llm =OpenAI(temperature=1,openai_api_key="", model_kwargs={"engine": "chat"})
template="""
%INSTRUCTIONS:
    Please summarize the following piece of text.
    Respond in a manner that a 5 year old would understand.

%TEXT:
    {text}
"""
prompt=PromptTemplate(
    input_variables=["text"],
    template=template,
    )

confusing_text="""
Some scientists called Prototaxites a lichen, others a fungus, and still others clung to the notion that it was some kind of tree.
“The problem is that when you look up close at the anatomy, it’s evocative of a lot of different things,
 but it’s diagnostic of nothing,” says Boyce, an associate professor in geophysical sciences and the Committee on Evolutionary Biology.
“And it’s so damn big that when whenever someone says it’s something, everyone else’s hackles get up: ‘How could you have a lichen 20 feet tall?’”
"""

print ("------- Prompt Begin -------")

final_prompt=prompt.format(text=confusing_text)
print(final_prompt)

print ("------- Prompt End -------")

output = llm(final_prompt)
print(output)
```

Finally, I was also able to turn a simple sentence from the user into an SQL syntax and connecting the code into my database so that I would receive a reply according to the database.

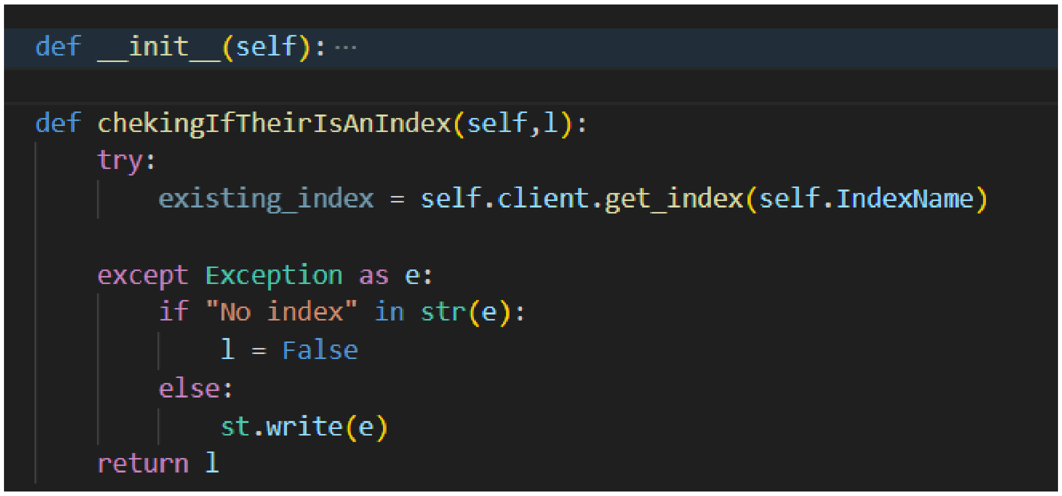


```python
from langchain.prompts import PromptTemplate
from langchain.agents import create_sql_agent
from langchain.agents.agent_toolkits import SQLDatabaseToolkit
from langchain.sql_database import SQLDatabase
from dotenv import load_dotenv
load_dotenv()
from langchain.agents import AgentExecutor
from langchain.agents.agent_types import AgentType
from mysql.connector import IntegrityError


import psycopg2

os.environ['OPENAI_API_KEY'] = ""
openai.api_type = "azure"
openai.api_base = "https://cog-ppxxbrri2qmg6.openai.azure.com/"
openai.api_version = "2023-03-15-preview"
openai.api_key = ""

# Language Model
llm = OpenAI(temperature=1, openai_api_key="", model_kwargs={"engine": "chat"})

mysql_username = "root"
mysql_password = ""
mysql_host = "localhost"
mysql_port = "3306"
mysql_database_name = "test2"


# Database Chain
mysql_uri = f"mysql+mysqlconnector://{mysql_username}:{mysql_password}@{mysql_host}:{mysql_port}/{mysql_database_name}"

db = SQLDatabase.from_uri(mysql_uri)

db_chain = SQLDatabaseChain(llm=llm, database=db, verbose=True)

print("Please enter a question for the AI to ansewr?")
question = input()

try:
    response = db_chain.run(question)

    print("Response from Language Model:")
    print(response)

except Exception as e:
    print()
    print("An Error has beed detected. The ID number is already present please choose a diffrent ID number! ")
    print()
    print()
    print(e)
```

I also learned other important things using LangChain and OpenAI, however, I only shared the most important information learned during that week. Unfortunately, since I was using keys to connect to ChatGPT and Azure I am not able to run my code and show the input and output results since the company has changed the keys. However, I will send a video of the final project which I will explain later.

6. <u>**Week 8-9: Project**</u>

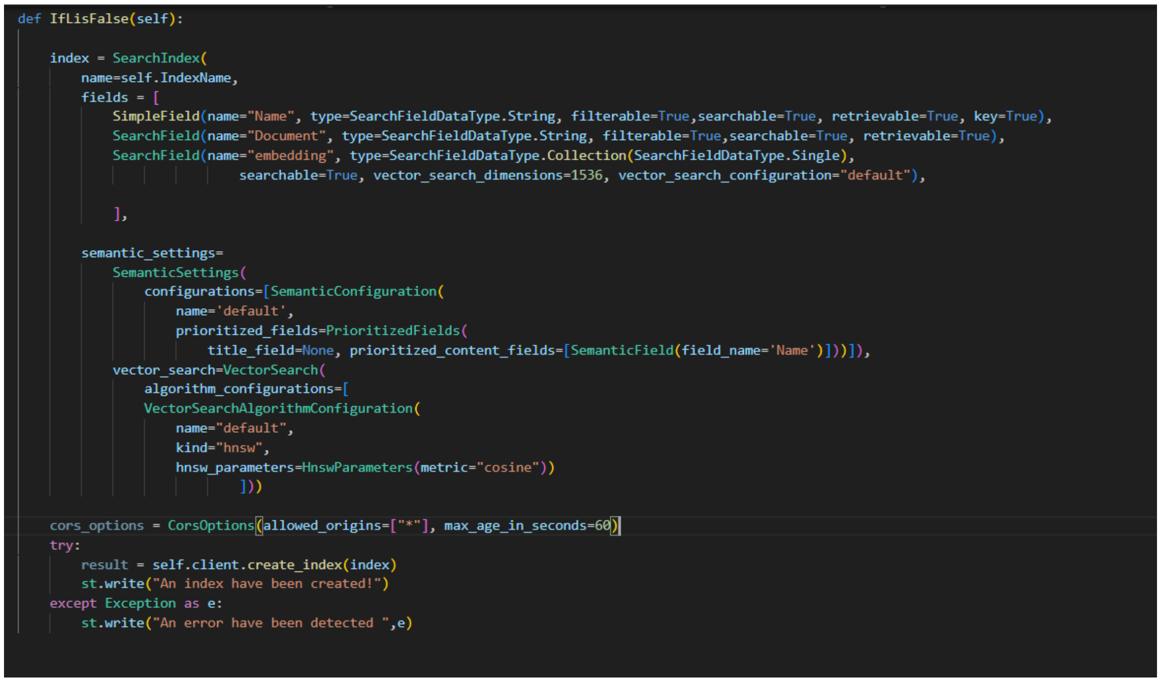
At the beginning of Week 8, I began my first AI project where I combined all of my previous knowledge into a big final piece. As the front end, I used a python library called Streamlit. Moreover, my python project is divided into 5 functions. The first two functions check if I created a place called “saved3” in Azure Cognitive Search to save the documents that the user adds. If I did not create the index saved3 then an L variable will be False else it will be True.



```
def __init__(self):⋯

def chekingIfTheirIsAnIndex(self,l):
    try:
        existing_index = self.client.get_index(self.IndexName)

    except Exception as e:
        if "No index" in str(e):
            l = False
        else:
            st.write(e)
    return l
```

The second function is only opened when L is False, when it is False it will create an index in Azure Cognitive Search to add the documents of the user. This index will have three fields the Name of the document which is a key, The Document itself, and the embedding of the document.

Embedding is used to transform high-dimensional data into a more compact and meaningful format that can be processed more effectively by machine learning algorithms. In other words, the document is transformed into a list of integers based on the meanings of that document. If the document is related to cats, then the embedding of the document will have similar numbers to the embedding of a document that is related to dogs since both documents are about animals. In the backend code, I used Semantic settings for the search index, this section defines how the search engine should prioritize fields when returning results. Vector Search Configuration configures vector search for the index. Vector search is used for searching and retrieving data based on vector representations “embedding’s” of the data.

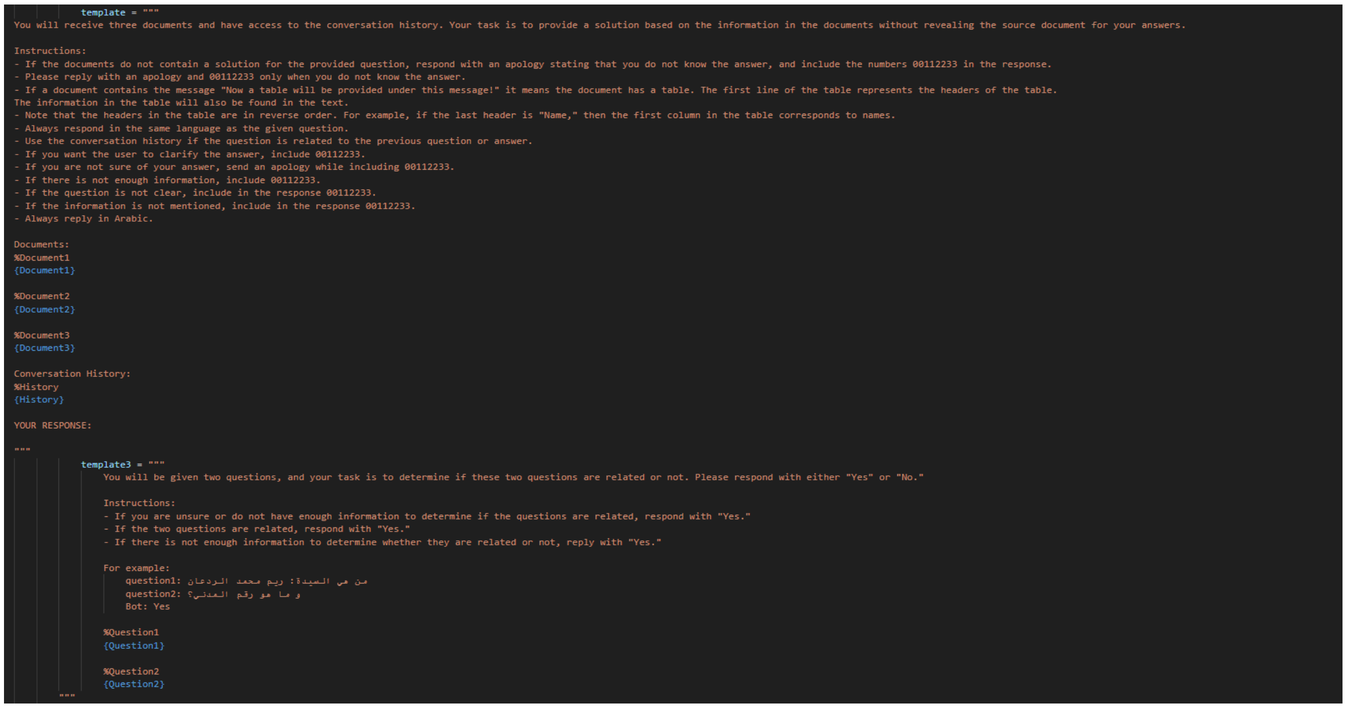


```python
def IfLisFalse(self):

    index = SearchIndex(
        name=self.IndexName,
        fields = [
            SimpleField(name="Name", type=SearchFieldDataType.String, filterable=True,searchable=True, retrievable=True, key=True),
            SearchField(name="Document", type=SearchFieldDataType.String, filterable=True,searchable=True, retrievable=True),
            SearchField(name="embedding", type=SearchFieldDataType.Collection(SearchFieldDataType.Single),
                        searchable=True, vector_search_dimensions=1536, vector_search_configuration="default"),

            ],

        semantic_settings=
            SemanticSettings(
                configurations=[SemanticConfiguration(
                    name='default',
                    prioritized_fields=PrioritizedFields(
                        title_field=None, prioritized_content_fields=[SemanticField(field_name='Name')]))]),
            vector_search=VectorSearch(
                algorithm_configurations=[
                VectorSearchAlgorithmConfiguration(
                    name="default",
                    kind="hnsw",
                    hnsw_parameters=HnswParameters(metric="cosine"))
                        ]))

    cors_options = CorsOptions(allowed_origins=["*"], max_age_in_seconds=60)
    try:
        result = self.client.create_index(index)
        st.write("An index have been created!")
    except Exception as e:
        st.write("An error have been detected ",e)
```

Moreover, the third function is activated when the user presses the Ask a Question option that is present on the frontend screen. Furthermore, once the user types a question, the prompts are being shown informing the AI of its tasks, if there is an error it types the error code 00112233, always replies in Arabic, apologizes if the answer to the question asked is not in the documents provided, and when the AI expects to read a table since if I do not inform him that he would read the table as a table, then he would read the table as paragraphs. I also used another prompt to ask if the previous question was related to the current question, and the AI would reply with either a Yes or No. If the AI replies with Yes then I do not need to search Azure cognitive search for new documents, I can simply reply with the documents that were previously used in the old question.

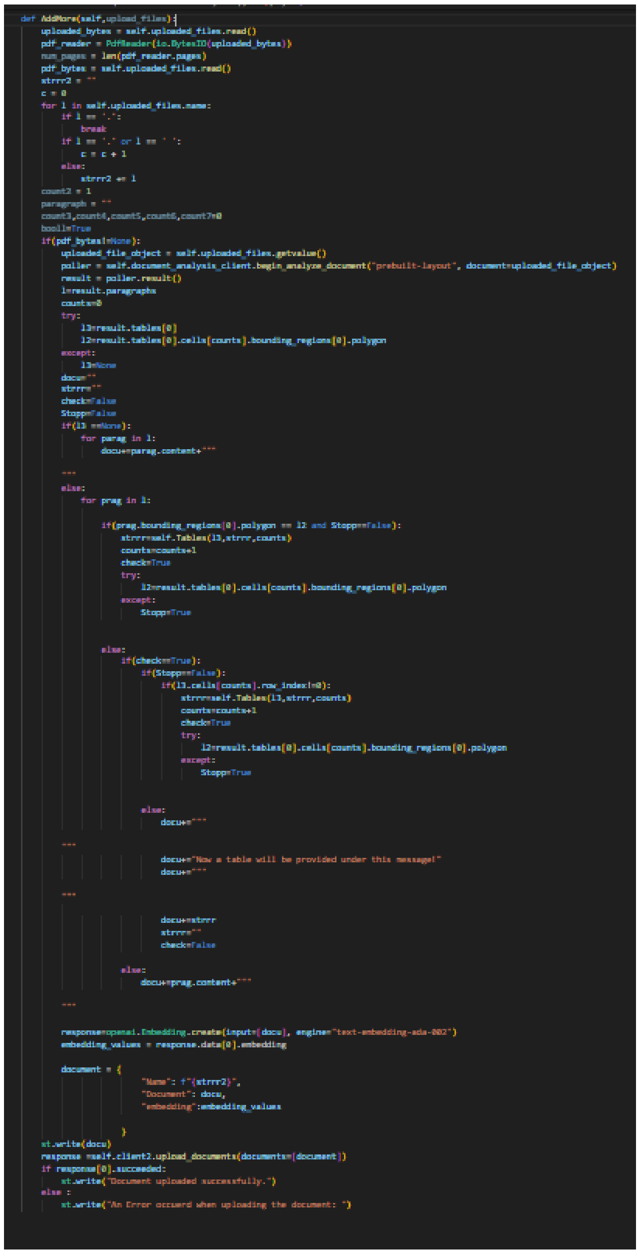
The st.sesstion_state[“chat_history”] is an array that keeps track of the history between the user and the AI and the System Messages. Moreover, I used a function in the openai library called openai.Embedding to create the embedding of the question and compare this embedding to the embedding of the documents. Furthermore, while searching for the documents in Azure Cognitive Search, I used both Embedding and non-embedding techniques to find the required Documents since in some questions the Embedding technique was giving the wrong document where the solution of the question was not present in that document. Similarly, the non-embedding technique had a similar problem to the Embedding. Therefore, by using both techniques the possibility of both not knowing the correct document has a lower chance than when I was using only one of the techniques.

Furthermore, there are times were the AI is getting confused with the history, therefore if the first response of the AI included 00112233 then I would ask the same question without history. However, if the second response of the AI also included 00112233 then I would use the prompt of asking the same question differently since the AI couldn't understand the question. In conclusion, if the AI responded for the third time with 00112233 then he would reply to the user that he doesn't know the answer. Unfortunately, this function is 400 lines of code and I am not able to share it as a screenshot However, this is a picture of the prompts I used in that function.



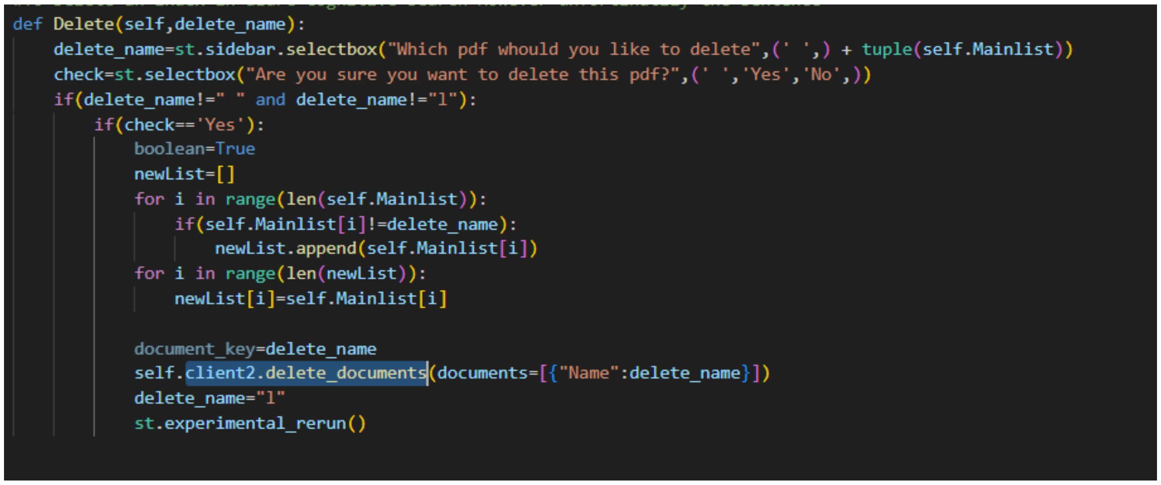
```
            template = """
You will receive three documents and have access to the conversation history. Your task is to provide a solution based on the information in the documents without revealing the source document for your answers.

Instructions:
- If the documents do not contain a solution for the provided question, respond with an apology stating that you do not know the answer, and include the numbers 00112233 in the response.
- Please reply with an apology and 00112233 only when you do not know the answer.
- If a document contains the message "Now a table will be provided under this message!" it means the document has a table. The first line of the table represents the headers of the table.
The information in the table will also be found in the text.
- Note that the headers in the table are in reverse order. For example, if the last header is "Name," then the first column in the table corresponds to names.
- Always respond in the same language as the given question.
- Use the conversation history if the question is related to the previous question or answer.
- If you want the user to clarify the answer, include 00112233.
- If you are not sure of your answer, send an apology while including 00112233.
- If there is not enough information, include 00112233.
- If the question is not clear, include in the response 00112233.
- If the information is not mentioned, include in the response 00112233.
- Always reply in Arabic.

Documents:
%Document1
{Document1}

%Document2
{Document2}

%Document3
{Document3}

Conversation History:
%History
{History}

YOUR RESPONSE:

"""
            template3 = """
                You will be given two questions, and your task is to determine if these two questions are related or not. Please respond with either "Yes" or "No."

                Instructions:
                - If you are unsure or do not have enough information to determine if the questions are related, respond with "Yes."
                - If the two questions are related, respond with "Yes."
                - If there is not enough information to determine whether they are related or not, reply with "Yes."

                For example:
                    question1: من هي السيدة: ريم محمد الردعان
                    question2: و ما هو رقم العدني؟
                    Bot: Yes

                %Question1
                {Question1}

                %Question2
                {Question2}
            """
```

The Fourth function is the Add More function and this function is activated when the user wants to add more documents to Azure Cognitive Search. Once the user presses on the Add document button the function gets Activated. Moreover, when the user uploads a document, I will be using a function in the library Azure Form recognizer called begin_analysis_document to analyze a document and retrieve its content. Form recognizer is useful if the document contains photos of any kind, even if it is a photo of handwritten information. Once the content of the document is retrieved, I wrote an algorithm for any kind of documents into function called Tables to arrange the content of a table in the format of a table and not a text. Furthermore, I used the function openai.Embedding.create() to turn the document into an Embedding. Hence I uploaded the documents using the function client.upload_document().



```
def AddMore(self,upload_files):
    uploaded_bytes = self.uploaded_files.read()
    pdf_reader = PdfReader(io.BytesIO(uploaded_bytes))
    num_pages = len(pdf_reader.pages)
    pdf_bytes = self.uploaded_files.read()
    strrr2 = ""
    c = 0
    for l in self.uploaded_files.name:
        if l == '.':
            break
        if l == '.' or l == ' ':
            c = c + 1
        else:
            strrr2 += l
    count2 = 1
    paragraph = ""
    count3,count4,count5,count6,count7=0
    bool1=True
    if(pdf_bytes!=None):
        uploaded_file_object = self.uploaded_files.getvalue()
        poller = self.document_analysis_client.begin_analyze_document("prebuilt-layout", document=uploaded_file_object)
        result = poller.result()
        l=result.paragraphs
        counts=0
        try:
            l3=result.tables[0]
            l2=result.tables[0].cells[counts].bounding_regions[0].polygon
        except:
            l3=None
        docu=""
        strrr=""
        check=False
        Stopp=False
        if(l3 ==None):
            for parag in l:
                docu+=parag.content+"""

        """
        else:
            for prag in l:

                if(prag.bounding_regions[0].polygon == l2 and Stopp==False):
                    strrr=self.Tables(l3,strrr,counts)
                    counts=counts+1
                    check=True
                    try:
                        l2=result.tables[0].cells[counts].bounding_regions[0].polygon
                    except:
                        Stopp=True


                else:
                    if(check==True):
                        if(Stopp==False):
                            if(l3.cells[counts].row_index!=0):
                                strrr=self.Tables(l3,strrr,counts)
                                counts=counts+1
                                check=True
                                try:
                                    l2=result.tables[0].cells[counts].bounding_regions[0].polygon
                                except:
                                    Stopp=True


                        else:
                            docu+="""

        """
                            docu+="Now a table will be provided under this message!"
                            docu+="""

        """

                            docu+=strrr
                            strrr=""
                            check=False

                    else:
                        docu+=prag.content+"""

        """

        response=openai.Embedding.create(input=[docu], engine="text-embedding-ada-002")
        embedding_values = response.data[0].embedding

        document = {
                    "Name": f"{strrr2}",
                    "Document": docu,
                    "embedding":embedding_values

                }
    st.write(docu)
    response =self.client2.upload_documents(documents=[document])
    if response[0].succeeded:
        st.write("Document uploaded successfully.")
    else :
        st.write("An Error occuerd when uploading the document: ")
```

The Fifth Function and the final Function is the Delete Function, this function is used when the user wants to delete a specific document in Azure Cognitive Search. The User would choose which document he wants to delete according to the names of each document since names are unique in Azure Cognitive Search and the user is not able to have two documents with the same name. Once the user enters which document to delete, I would remove the name of the document in Azure Cognitive Search using the function client.delete_document.



```python
def Delete(self,delete_name):
    delete_name=st.sidebar.selectbox("Which pdf whould you like to delete",(' ',) + tuple(self.Mainlist))
    check=st.selectbox("Are you sure you want to delete this pdf?",(' ','Yes','No',))
    if(delete_name!=" " and delete_name!="1"):
        if(check=='Yes'):
            boolean=True
            newList=[]
            for i in range(len(self.Mainlist)):
                if(self.Mainlist[i]!=delete_name):
                    newList.append(self.Mainlist[i])
            for i in range(len(newList)):
                newList[i]=self.Mainlist[i]

            document_key=delete_name
            self.client2.delete_documents(documents=[{"Name":delete_name}])
            delete_name="1"
            st.experimental_rerun()
```

In conclusion, I was able to build a website that would Answer any question the user asks according to the documents provided in Azure Cognitive Search. The code is divided into five major functions. The first two functions CheckingIfTheirIsAnIndex and IfLIsFalse are used to make sure that there is a unique place in Azure Cognitive Search to save and retrieve documents. The third function called AskAquestion is used when the user wants to ask a question, this function workflow is that I turn the question into an Embedding then I ask the question to Azure Cognitive Search in which I get the top 3 most possible results that could have the solution of the question asked. After I receive the documents I would give those documents to ChatGPT with the prompt and he would respond with the solution of the question. The last two functions are used when the user wants to add or delete documents.

7. <u>Difficulties:</u>

My project has multiple weaknesses and I have dealt with each of the weaknesses that exist uniquely. Some of the weaknesses were eliminated, however unfortunately some weaknesses that exist have been dealt with in a way that would decrease the chances of it happening, yet not eliminate it.

1. **Document Length:** Unfortunately, one of the major problems that I was facing was when I wanted to give huge documents to Azure Cognitive Search and then ask a question, I received an Exception that said I have exceeded the maximum number of tokens in the system message. A token is a representation of a word, phrase, symbol, or expression. Moreover, to deal with this issue, I have tried multiple methods. First, I remove some of the Chat History only when the exception appears using the try method in Python. Second, I wrote an algorithm that divides the document into multiple small documents according to the minimum and maximum number of tokens that I chose. However, this algorithm loses information that is related between the two small chunks of documents which is not the best solution. Therefore, another way to fix this issue is to manually divide the documents into smaller documents in a way that information is not lost between the documents. However, this technique is time-consuming since it would consume a significant amount of time for the individual who is dividing the document. If I was still receiving this Exception, I would remove the third document that is given to ChatGPT since the third document has the least chance of having the solution to the problem.

2. **Correct Documents:** Another weakness is when Azure Cognitive Search would choose the incorrect document when provided with the question. In response to this weakness, I used multiple techniques. First, the idea of using both Embedding and Non-Embedding search techniques and then comparing the documents that are provided by both documents. If both Embedding and Non-Embedding provided different documents then I would give ChatGPT the top 1 in Embedding, top 1 in non-Embedding, and top 2 in non-Embedding. The Second technique I used is a filter that is available in Azure Cognitive Search which is “top”. This method would retrieve the top 3 documents that could have the answer to the question provided, when the user sends a question I would give the question to Azure Cognitive Search to choose which document could have the answer. Moreover, after the documents are received, the algorithm would give them to ChatGPT for it to provide the solution. Unfortunately, this solution that I provided simply decreased the chance of Azure

Cognitive Search providing the wrong document, In other words, there is still a chance in which wrong documents are provided.

3. Inaccurate/Incorrect: Another Weakness that I was facing was when ChatGPT provided an incorrect solution or replied that the answer was not found, yet the correct document was provided to ChatGPT. To fix this issue, I wrote an algorithm that would arrange that document in a way that ChatGPT would not confuse multiple paragraphs with each other. In other words, the content of the document is separated by lines instead of being a big chunk of information. Moreover, **I think** that this solution decreased the possibility for ChatGPT to reply incorrectly.

4. **History:** I was also facing the problem that when the user asks a question according to the history, another document would be provided from Azure Cognitive Search from the original which would lead to an incorrect response. In response, I wrote a Python code that would ask ChatGPT if this question is related to the previous question, if the questions are related then I would not request another set of documents from Cognitive Search, however, if the questions are not related then I would treat this question as a regular question and as explained before providing the answer to that question.

8. **Learning Outcomes**

During the internship program, I learned what are Azure Microsoft and its services like Azure Cognitive Search, Azure Form Recognizer, and Azure bot framework Composer. I also learned multiple Python libraries such as OpenAI, LangChain, and Streamlit. Moreover, I learned front-end development such as HTML, CSS, JavaScript, and React. This internship was a unique internship where I learned how to create my own AI.

9. **Courses from the university that were applied in the internship**

a. CSC 243 Object Oriented Program:

During the internship, Most of my time I was coding in Python, In other words, this course that helped me learn Python was essential in helping me understand and execute Python code.

b. CSC 310Algorithms and Data Structure:

While facing a problem during the development of the project, I would think of a better way to enhance my code and provide the company with a better algorithm. This course provided me with the ability to think differently in creating an algorithm that would satisfy my requirements.

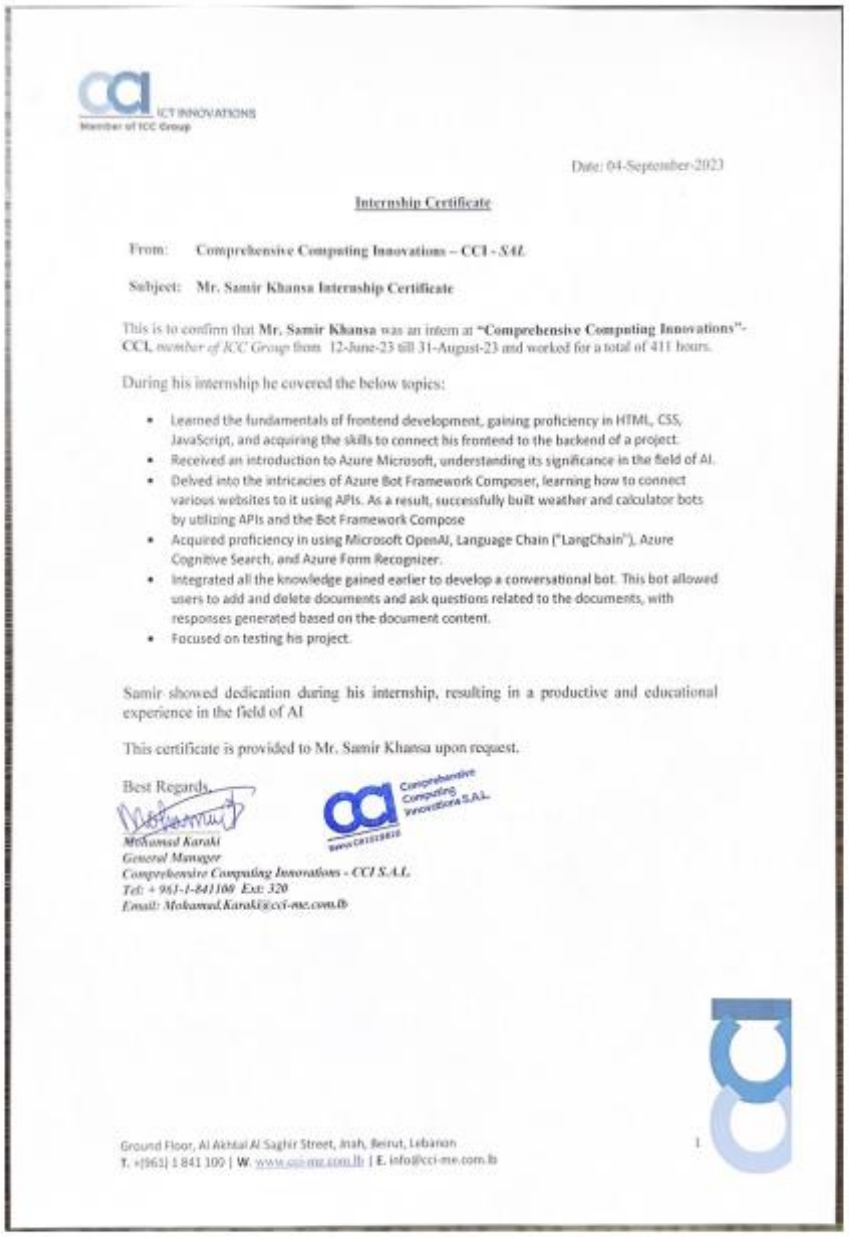
c. CSC 375 Database Management System:

During the time where I was training how to use the library LangChain, I learned how to connect LangChain and OpenAI to a database. In other words, I used my knowledge of SQL statements that were learned during the course to obtain the correct queries and navigate through the database to make sure that OpenAI is responding correctly. Unfortunately, OpenAI was not always responding correctly.

10. Conclusion:

In Conclusion, this internship was very productive. I got introduced to multiple libraries, languages, Technologies and tools (Embedding and Vector database), of which I had no previous knowledge. Moreover, working on a project and facing difficulties without any external help was entertaining and difficult at the same time. Finally, I view myself as having knowledge of Artificial Intelligence and when graduating from university, I would be requesting job opportunities with pride of having an internship in the field of Artificial Intelligence.

11. **Certificate**



ICT INNOVATIONS
Member of ICC Group

Date: 04-September-2023

Internship Certificate

From: **Comprehensive Computing Innovations – CCI - *SAL***

Subject: **Mr. Samir Khansa Internship Certificate**

This is to confirm that **Mr. Samir Khansa** was an intern at **"Comprehensive Computing Innovations"-CCI**, *member of ICC Group* from 12-June-23 till 31-August-23 and worked for a total of 411 hours.

During his internship he covered the below topics:

- Learned the fundamentals of frontend development, gaining proficiency in HTML, CSS, JavaScript, and acquiring the skills to connect his frontend to the backend of a project.
- Received an introduction to Azure Microsoft, understanding its significance in the field of AI.
- Delved into the intricacies of Azure Bot Framework Composer, learning how to connect various websites to it using APIs. As a result, successfully built weather and calculator bots by utilizing APIs and the Bot Framework Compose
- Acquired proficiency in using Microsoft OpenAI, Language Chain ("LangChain"), Azure Cognitive Search, and Azure Form Recognizer.
- Integrated all the knowledge gained earlier to develop a conversational bot. This bot allowed users to add and delete documents and ask questions related to the documents, with responses generated based on the document content.
- Focused on testing his project.

Samir showed dedication during his internship, resulting in a productive and educational experience in the field of AI

This certificate is provided to Mr. Samir Khansa upon request.

Best Regards,

[illegible]

Mohamad Karaki
General Manager
Comprehensive Computing Innovations - CCI S.A.L.
Tel: + 961-1-841100 Ext: 320
Email: Mohamad.Karaki@cci-me.com.lb

Comprehensive Computing Innovations S.A.L.

Ground Floor, Al Akhtal Al Saghir Street, Jnah, Beirut, Lebanon
T. +(961) 1 841 100 | W. www.cci-me.com.lb | E. info@cci-me.com.lb

1