AI & ML-Mostafa Elfaggal-Task6

September 4, 2025

0.1 Full Repo

https://github.com/MostafaBelo/Konecta_Assignments/tree/main

0.2 Imports

[ ]:
```
!pip install -r requirements.txt
```

[1]:
```
import getpass
import os
import sys
from dotenv import load_dotenv

import uuid
import json
import hashlib

from langchain_google_genai import ChatGoogleGenerativeAI

from langchain.schema import Document
from langchain_community.document_loaders import PyPDFLoader
from langchain.text_splitter import RecursiveCharacterTextSplitter

import chromadb
from chromadb.utils import embedding_functions

import sqlite3
from langgraph.checkpoint.sqlite import SqliteSaver

from langchain_core.messages import SystemMessage, HumanMessage, AIMessage,␣
↪RemoveMessage

from IPython.display import Image, display
from langgraph.graph import StateGraph, START, END
from langgraph.graph import MessagesState

from tqdm import tqdm
```

[2]:
```
# Load .env file
load_dotenv()

env_keys = [
    "GOOGLE_API_KEY",
]
for key in env_keys:
    if not os.environ.get(key):
        os.environ[key] = getpass.getpass(f"Enter API key for {key}: ")
```

0.3 Utils

[ ]:
```
# Gemini Model
model = ChatGoogleGenerativeAI(
    model="gemini-2.5-flash",
    temperature=0,
    max_tokens=None, # Max output tokens
    timeout=None,
    max_retries=2,
    google_api_key=os.environ.get("GOOGLE_API_KEY") # Replace with API key from␣
 ↪https://aistudio.google.com/apikey
)
```

[ ]:
```
# Embedings Model
sentence_transformer_ef = embedding_functions.
 ↪SentenceTransformerEmbeddingFunction(
    model_name="sentence-transformers/all-MiniLM-L6-v2"
)
```

0.4 Vector Store - ChromaDB

[5]:
```
chroma_client = chromadb.PersistentClient(path="./db")
```

[6]:
```
collection = chroma_client.get_or_create_collection(
    name="konecta_assignment_6",
    metadata={
        "loaded_documents": "[]",
    },
    embedding_function=sentence_transformer_ef
)
```

[7]:
```
collection.count()
```

[7]:
```
371
```

0.5 Manuals Loading

[8]:
```
pdf_paths = [os.path.join("manuals", pdf_path) for pdf_path  in os.
 ↪listdir("manuals")]
loaded_documents: list[str] = json.loads(collection.
 ↪metadata["loaded_documents"])

docs: list[Document] = []

# Chunker
recursive_splitter = RecursiveCharacterTextSplitter(
    chunk_size=800, chunk_overlap=120, separators=["\n\n", "\n", " ", ""]
)

# metadata extractors
def getId(metadata: str, text: str) -> str:
            return hashlib.sha256(f"{metadata} | {text}".encode("utf-8")).
 ↪hexdigest()
def getMetaData(full_metadata: dict[str, any]):
    res = {
        "document_name": "",
        "page_number": -1,
        "section_heading": "",
    }

    if "source" in full_metadata:
        res["document_name"] = full_metadata["source"]
    if "page" in full_metadata:
        res["page_number"] = full_metadata["page"]

    for key in res.keys():
        if key in full_metadata:
            res[key] = full_metadata[key]

    return res

# Documents loading and storing
for pdf_path in tqdm(pdf_paths, desc="PDF Documents"):
    if pdf_path not in loaded_documents:
        loader = PyPDFLoader(pdf_path)
        docs = loader.load()  # one Document per page

        docs = [c for c in docs if c.page_content != ""]

        recursive_chunks = recursive_splitter.split_documents(docs)

        recursive_chunks = [c for c in recursive_chunks if c.page_content != ""]
```

```
        collection.add(
            ids=[getId(str(c.metadata), c.page_content) for c in␣
 ↪recursive_chunks],
            metadatas=[getMetaData(c.metadata) for c in recursive_chunks],
            documents=[c.page_content for c in recursive_chunks],
        )

        loaded_documents.append(pdf_path)
        collection.modify(metadata={
             "loaded_documents": json.dumps(loaded_documents)
        })
```

```
PDF Documents: 100%|      | 4/4 [00:00<00:00, 45590.26it/s]
```

[9]:
```
collection.count()
```

[9]:
```
371
```

0.6 Chat Bot with Memory

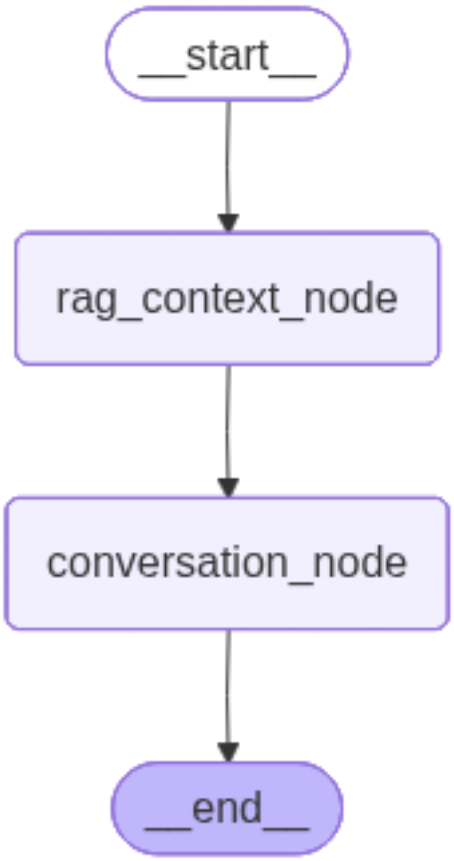
[10]:
```
# In memory
conn = sqlite3.connect(":memory:", check_same_thread = False)

os.makedirs("state_db", exist_ok=True)
db_path = "state_db/example.db"
conn = sqlite3.connect(db_path, check_same_thread=False)

memory = SqliteSaver(conn)
```

[11]:
```
system_message = """
You are a helpful assistant that has access to a few manuals.
Answer questions based on previous chat history and the content within the␣
 ↪manual snippets provided to you in the prompts.
If the question of the user concerns content not covered by previous chat␣
 ↪history, not by the manuals and not in the snippets of context retrerived,␣
 ↪respond accordingly to the user stating that you don't have the answer.
Make sure to cite all your responses from the manuals' metadata. The metadata␣
 ↪will include the manual's file name and the page from which the snippet is␣
 ↪extracted.
"""

rag_system_template = """
Use the following manual snippets, any previously provided ones, and previous␣
 ↪chat history, as context to respond to the following user query. Not all␣
 ↪snippets may be relevant.
```

```
Answer questions based on previous chat history and the content within the␣
 ↪manual snippets provided to you in the prompts.
If the question of the user concerns content not covered by previous chat␣
 ↪history, not by the manuals and not in the snippets of context retrerived,␣
 ↪respond accordingly to the user stating that you don't have the answer.
Make sure to cite all your responses from the manuals' metadata. The metadata␣
 ↪will include the manual's file name and the page from which the snippet is␣
 ↪extracted.
Also keep in mind the previous chat history in case of follow up questions.

{context}
"""

def rag_node(state: MessagesState):
    global rag_system_template

    if len(state["messages"]) < 1:
        # raise Exception("No Prompt to RAG")
        return {}

    last_message = state["messages"][-1]
    if not isinstance(last_message, HumanMessage):
        # raise Exception("No Prompt to RAG")
        return {}

    document_names: list[str] = json.loads(collection.
 ↪metadata["loaded_documents"])

    combined_results = {
        "documents": [],
        "metadatas": []
    }

    for document_name in document_names:
        results = collection.query(
            query_texts=[last_message.content],
            n_results=5,
            where={
                "document_name": document_name
            }
        )

        combined_results["documents"] += results["documents"][0]
        combined_results["metadatas"] += results["metadatas"][0]

    context_content = "\n\n".join([f"Metadata: {meta}\nContent: {doc}" for doc,␣
 ↪meta in zip(combined_results["documents"], combined_results["metadatas"])])
```

```
    context_message = rag_system_template.format(context=context_content)

    return {
        "messages": [
            RemoveMessage(id=last_message.id),
            SystemMessage(content=context_message),
            HumanMessage(content=last_message.content),
        ]
    }

def call_model_node(state: MessagesState):
    messages = [SystemMessage(content=system_message)] + state["messages"]

    response = model.invoke(messages)
    return {"messages": response}
```

[12]:
```
# Define a new graph
workflow = StateGraph(MessagesState)
workflow.add_node("rag_context_node", rag_node)
workflow.add_node("conversation_node", call_model_node)

# Set the entrypoint as conversation
workflow.add_edge(START, "rag_context_node")
workflow.add_edge("rag_context_node", "conversation_node")
workflow.add_edge("conversation_node", END)

# Compile
graph = workflow.compile(checkpointer=memory)
display(Image(graph.get_graph().draw_mermaid_png()))
```



[13]:
```python
class ChatThread:
    def __init__(self, thread_id: str | None = None):
        if thread_id is None:
            self.thread_id = str(uuid.uuid4())
        else:
            self.thread_id = thread_id
        self.config = {"configurable": {"thread_id": self.thread_id}}

        self.state: MessagesState = {
            "messages": []
        }

    def invoke(self, prompts: str | list[str]):
        if not isinstance(prompts, list):
            prompts = [prompts]

        for prompt in prompts:
            input_message = HumanMessage(content=prompt)
            output = graph.invoke({"messages": [input_message]}, self.config)
```

```
        self.state = output

    def print_history(self, show_system=False):
        for m in self.state["messages"]:
            if isinstance(m, SystemMessage) and not(show_system):
                continue
            m.pretty_print()
```

[14]:
```
chat = ChatThread()

prompts = [
    "what are the different laptop serires covered by the manuals?",
    "what ram does each laptop series in any of the manuals have?",
]
chat.invoke(prompts)

chat.print_history()
```

```
================================ Human Message

================================

what are the different laptop serires covered by the manuals?
================================== Ai Message

==================================

The manuals cover the following laptop series:

*   **GIGABYTE G6** (manuals/g6_mf.pdf, page 0)
*   **Lenovo IdeaPad L340 Series** (manuals/ideapad_l340_series.pdf, page 4)
*   **Lenovo LOQ Series** (specifically models 15AHP9, 15ARP9, 15IAX9, 15IAX9I,
and 15IRX9) (manuals/loq_15ahp9.pdf, page 0)
*   **ASUS TUF Gaming F17 (FX707Z) series** (manuals/tuf_gaming_f17_series.pdf,
page 3, 5)
================================ Human Message

================================

what ram does each laptop series in any of the manuals have?
================================== Ai Message

==================================

Based on the provided manuals, here is the RAM information for each laptop
series:

*   **GIGABYTE G6:** The specific type or capacity of RAM is not detailed in the
```

provided snippets for this series (manuals/g6_mf.pdf).
*   **Lenovo IdeaPad L340 Series:** This series uses Double Data Rate 4 (DDR4) Small Outline Dual In-line Memory Modules (SODIMM) (manuals/ideapad_l340_series.pdf, page 13).
*   **Lenovo LOQ Series (15AHP9, 15ARP9, 15IAX9, 15IAX9I, and 15IRX9):** These models use Double Data Rate 5 (DDR5) Small Outline Dual In-line Memory Modules (SO-DIMM) and have 2 physical slots (manuals/loq_15ahp9.pdf, page 15).
*   **ASUS TUF Gaming F17 (FX707Z) series:** The manual mentions "DDR" as a component and provides instructions for installing "RAM" or "Memory Module," but it does not specify the exact type (e.g., DDR4, DDR5) (manuals/tuf_gaming_f17_series.pdf, page 3, 6, 15).

[15]:
```
chat = ChatThread()

prompts = [
    "How do phones work?",
]
chat.invoke(prompts)

chat.print_history()
```

```
================================ Human Message

================================

How do phones work?
================================ Ai Message

================================

I apologize, but the provided manuals do not contain information on how phones work. The manuals focus on the features and components of GIGABYTE, Lenovo IdeaPad, Lenovo LOQ, and ASUS TUF Gaming laptops.
```

[ ]:
```
chat = ChatThread()

prompts = [
    "what are the hotkeys and their functions of Gigabyte G6?",
    "Which of them would I use to put the laptop to sleep?",
]
chat.invoke(prompts)

chat.print_history()
```

```
================================ Human Message

================================

what are the hotkeys and their functions of Gigabyte G6?
```

========== **Ai Message**

==========

The Gigabyte G6 computer employs hotkeys, which are key combinations used to access various controls. To activate these hotkeys, you need to press and hold the `<Fn>` key before pressing the other key in the combination.

Here are the hotkeys and their functions:

*   **Fn + ~**: Play / Pause (in Audio/Video Programs)
*   **Fn + F1**: Touchpad Toggle
*   **Fn + F2**: Turn LCD Backlight OFF (Press a key or use Touchpad to turn on)
*   **Fn + F3**: Mute Toggle
*   **Fn + F5/F6**: Volume Decrease/Increase
*   **Fn + F7**: Display Toggle
*   **Fn + F8/F9**: Brightness Decrease/Increase
*   **Fn + F10**: PC Camera Power on and Off
*   **Fn + F11**: Airplane Mode Toggle
*   **Fn + F12**: Sleep Toggle
*   **NumLK**: Number Lock Toggle
*   **Fn + ScrLK**: Scroll Lock Toggle
*   **Caps Lock**: Caps Lock Toggle
*   **Fn + Esc**: Control Center Toggle
*   **Fn + !**: Fan Automatic Control / Full Power
*   **Fn + Backspace**: Disable/Enable Flexikey®
*   **Fn + #**: Power Modes Toggle
*   **Fn + (Up Arrow)**: Launch the LED Keyboard Application
*   **Fn + (Right Arrow)**: Toggle the Keyboard LED On/Off
*   **Fn + (Left Arrow)**: Keyboard LED Illumination Decrease
*   **Fn + (Down Arrow)**: Keyboard LED Illumination Increase

(Source: manuals/g6_mf.pdf, page 3)
========== **Human Message**

==========

Which of them would I use to put the laptop to sleep?
========== **Ai Message**

==========

To put the Gigabyte G6 laptop to sleep, you would use the hotkey combination **Fn + F12**. This functions as the Sleep Toggle.

(Source: manuals/g6_mf.pdf, page 3)

[ ]:
```
chat = ChatThread()

prompts = [
    "Which of the laptops has the best help and support post-purchase?",
    "Which of the laptops is objectively the most capable specs wise?",
    "In your opinion, which of the laptops would be the besst balance between␣
 ↪customer support and specs?",
]
chat.invoke(prompts)

chat.print_history()
```

```
================================ Human Message

================================

Which of the laptops has the best help and support post-purchase?
================================ Ai Message

==================================

Based on the provided manual snippets, the **Lenovo LOQ series** appears to
offer the most comprehensive help and support post-purchase, as it explicitly
states that additional services can be purchased both during and after the
warranty period (manuals/loq_15ahp9.pdf, page 40).

Here's a breakdown of the support mentioned for each:

*   **Lenovo LOQ 15AHP9, 15ARP9, 15IAX9, 15IAX9I, and 15IRX9:**
    *   Offers services during the warranty period including problem
determination, Lenovo hardware repair, and engineering change management
(manuals/loq_15ahp9.pdf, page 39).
    *   Crucially, you can **purchase additional services during and after the
warranty period** (manuals/loq_15ahp9.pdf, page 40).

*   **Lenovo IdeaPad L340 Series:**
    *   Provides "Lenovo Vantage," a customized one-stop solution for automated
updates, fixes, hardware settings, and personalized support, including system
updates for firmware and drivers (manuals/ideapad_l340_series.pdf, page 19).
    *   Offers services during the warranty period such as problem
determination, Lenovo hardware repair, and engineering change management
(manuals/ideapad_l340_series.pdf, page 39).

*   **GIGABYTE G6 MF:**
    *   Provides links to a GIGABYTE service website for warranty and service
information, as well as a link for FAQs (manuals/g6_mf.pdf, page 7).

*   **ASUS TUF Gaming F17 (FX707Z) series:**
```

    *   The provided snippets focus on a service guide, disassembly steps, and repair overview, but do not detail user-facing help, support, or warranty services for the end-user (manuals/tuf_gaming_f17_series.pdf, page 0, 2, 3, 8).

While both Lenovo models offer strong support during the warranty, the explicit mention of purchasing additional services *after* the warranty period for the Lenovo LOQ series makes it stand out for long-term post-purchase support.
================================ **Human Message**

================================

Which of the laptops is objectively the most capable specs wise?
================================ **Ai Message**

================================

Based on the provided manual snippets, it is challenging to definitively declare one laptop as "objectively the most capable specs-wise" because crucial specifications like specific CPU models, GPU models, and RAM capacities are not consistently provided for all models.

However, we can highlight the strengths of each based on the available information:

*   **Lenovo LOQ 15AHP9, 15ARP9, 15IAX9, 15IAX9I, and 15IRX9:**
    *   **Display:** Offers the highest specified display resolution and refresh rate among the listed laptops, with options for **2560 × 1440 pixels (WQHD models)** and a maximum refresh rate of **165 Hz or 144 Hz** (manuals/loq_15ahp9.pdf, page 16).
    *   **Storage:** Specifies Solid-state drive (SSD) with **M.2 2242 or M.2 2280 form factors** and a **PCI Express bus**, which generally indicates faster performance than SATA (manuals/loq_15ahp9.pdf, page 16).
    *   **Connectivity:** Features **Wi-Fi 6** and **Bluetooth 5.2**, along with SuperSpeed USB 5 Gbps and an HDMI connector supporting up to **3840 × 2160 @ 60 Hz** (manuals/loq_15ahp9.pdf, page 17).

*   **ASUS TUF Gaming F17 (FX707Z) series:**
    *   **Connectivity:** Stands out with a **Thunderbolt 4 port**, which offers high-speed data transfer, video output, and power delivery capabilities (manuals/tuf_gaming_f17_series.pdf, page 5). It also includes a USB 3.2 Gen 2 Type-C®/DisplayPort/Power Delivery combo port and a LAN port (manuals/tuf_gaming_f17_series.pdf, page 5).
    *   **Battery:** Mentions a **90W battery** (manuals/tuf_gaming_f17_series.pdf, page 6).

*   **Lenovo IdeaPad L340 Series:**
    *   **Storage:** Offers a variety of storage options including **2.5 Inch SATA SSD, M.2 PCIe/SATA SSD*, 2.5 Inch SATA HDD, and Intel® Optane memory**

(manuals/ideapad_l340_series.pdf, page 13).
    *   **Display:** Available in 15.6-inch and 17.3-inch models, with resolutions up to **1920 x 1080 pixels FHD** (manuals/ideapad_l340_series.pdf, page 13).

*   **GIGABYTE G6 MF:**
    *   **Connectivity:** Includes USB 3.2 (Type-A and Type-C), USB 2.0, Ethernet, HDMI, and a **Mini DisplayPort 1.4** (manuals/g6_mf.pdf, page 2).

Conclusion:

Based on the available details:

*   The **Lenovo LOQ series** appears to have the most capable **display specifications** (WQHD resolution and high refresh rates).
*   The **ASUS TUF Gaming F17 series** offers the most advanced **connectivity** with its Thunderbolt 4 port.

Without specific CPU, GPU, and RAM capacities for all models, a complete objective comparison of overall capability is not possible from the provided snippets.
================================ **Human Message**

================================

In your opinion, which of the laptops would be the besst balance between customer support and specs?
================================ **Ai Message**

================================

Based on the provided information, the **Lenovo LOQ series (15AHP9, 15ARP9, 15IAX9, 15IAX9I, and 15IRX9)** appears to offer the best balance between customer support and specifications.

Here's why:

*   **Customer Support:** The Lenovo LOQ series explicitly states that you can **purchase additional services during and after the warranty period** (manuals/loq_15ahp9.pdf, page 40). This indicates a robust and flexible long-term support system beyond the initial warranty, which is a significant advantage for post-purchase care. It also offers standard warranty services like problem determination, hardware repair, and engineering change management (manuals/loq_15ahp9.pdf, page 39).
*   **Specifications:** The Lenovo LOQ series boasts the highest specified display resolution and refresh rate among the listed laptops, with options for **2560 × 1440 pixels (WQHD models)** and a maximum refresh rate of **165 Hz or 144 Hz** (manuals/loq_15ahp9.pdf, page 16). It also specifies SSD storage with a

PCI Express bus, generally indicating faster performance (manuals/loq_15ahp9.pdf, page 16).

While the ASUS TUF Gaming F17 series offers a notable Thunderbolt 4 port (manuals/tuf_gaming_f17_series.pdf, page 5), the provided snippets lack detailed information regarding its customer support services, making it difficult to assess its balance. The Lenovo IdeaPad L340 series offers good support through Lenovo Vantage and standard warranty services (manuals/ideapad_l340_series.pdf, page 19, 39), but its display specifications (up to 1920 x 1080 FHD) are not as high as the LOQ series (manuals/ideapad_l340_series.pdf, page 13). The GIGABYTE G6 MF provides links to support resources but doesn't detail the extent of its services (manuals/g6_mf.pdf, page 7).

Therefore, the **Lenovo LOQ series** strikes a strong balance by offering excellent, long-term customer support options alongside impressive display and storage specifications.

[19]:
```
chat = ChatThread()

prompts = [
    "How is each laptop's battery-life?",
]
chat.invoke(prompts)

chat.print_history()
```

================================ **Human Message**

================================

How is each laptop's battery-life?
================================ **Ai Message**

==================================

I apologize, but the provided manual snippets do not contain information regarding the specific battery life (how long the battery lasts on a single charge) for any of the laptops.

However, the manuals do provide details on battery charging times and management:

* **Lenovo IdeaPad L340 Series:** The battery is fully charged in approximately four to eight hours. If the power adapter supports rapid charge, the battery can reach 80% charge in about 1 hour when the computer is turned off. The recommended temperature range for charging is between 10°C (50°F) and 35°C (95°F). The computer will not start recharging if the remaining power is greater than 95% to maximize battery life (manuals/ideapad_l340_series.pdf, page 26).

*   **Lenovo LOQ 15ahp9:** In normal charging mode, it typically takes 2 to 4 hours for the battery to charge from 0% to 100%. The recommended temperature range for charging is between 10°C (50°F) and 35°C (95°F). To maximize battery life, once fully charged, the battery must discharge to 94% or lower before it will recharge again (manuals/loq_15ahp9.pdf, page 30). There is also a conservation mode that keeps the battery charge between 75%-80% for long-term health (manuals/loq_15ahp9.pdf, page 31).
*   **GIGABYTE G6:** The manual advises against removing the built-in lithium battery and notes that storing or using the Notebook PC in direct sunlight or temperatures exceeding 112°F (45°C) can lead to Lithium-ion battery expansion and aging (manuals/g6_mf.pdf, page 2).