In [1]:
```
# Task 1: Simple FAQ Chatbot for Iron Lady Programs

# Define FAQs
faqs = {
    "program details": "Iron Lady offers leadership programs designed to empower wo
    "duration": "The programs typically run for 8 to 12 weeks depending on the trac
    "mode": "Our programs are conducted online and hybrid with live mentor interact
    "certification": "Yes, participants receive a recognized certificate upon succe
    "mentors": "Our mentors are industry leaders, experienced professionals, and ce
}

def chatbot(question):
    """Simple keyword-based FAQ chatbot"""
    question = question.lower()
    for key in faqs:
        if key in question:
            return faqs[key]
    return "I'm not sure about that. Please visit our website or contact support."

# Test chatbot
print("Chatbot Demo:")
print("Q: What is the program duration?")
print("A:", chatbot("What is the program duration?"))
print()
print("Q: Do you provide certification?")
print("A:", chatbot("Do you provide certification?"))
```

```
Chatbot Demo:
Q: What is the program duration?
A: The programs typically run for 8 to 12 weeks depending on the track.

Q: Do you provide certification?
A: Yes, participants receive a recognized certificate upon successful completion.
```

In [2]:
```
# Bonus: AI Enhanced Chatbot (Requires OpenAI API key)
!pip install openai

import openai

openai.api_key = "sk-proj-OkpWMruFiWQvXE848f3PgbE729vmPufnSxH7KyJrJ6HGA9qpvmf6zwEBz

def ai_chatbot(question):
    """Chatbot powered by OpenAI GPT"""
    prompt = f"Answer FAQs about Iron Lady's leadership programs: {question}"
    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=[{"role": "user", "content": prompt}]
    )
    return response['choices'][0]['message']['content']

# Example
# print(ai_chatbot("Tell me about the mentors in the program"))
```

```
Requirement already satisfied: openai in c:\users\user\anaconda3\lib\site-packages
(1.107.0)
Requirement already satisfied: anyio<5,>=3.5.0 in c:\users\user\anaconda3\lib\site-
packages (from openai) (3.5.0)
Requirement already satisfied: pydantic<3,>=1.9.0 in c:\users\user\anaconda3\lib\si
te-packages (from openai) (2.11.7)
Requirement already satisfied: distro<2,>=1.7.0 in c:\users\user\anaconda3\lib\sit
e-packages (from openai) (1.9.0)
Requirement already satisfied: httpx<1,>=0.23.0 in c:\users\user\anaconda3\lib\sit
e-packages (from openai) (0.28.1)
Requirement already satisfied: sniffio in c:\users\user\anaconda3\lib\site-packages
(from openai) (1.3.1)
Requirement already satisfied: jiter<1,>=0.4.0 in c:\users\user\anaconda3\lib\site-
packages (from openai) (0.10.0)
Requirement already satisfied: tqdm>4 in c:\users\user\anaconda3\lib\site-packages
(from openai) (4.64.1)
Requirement already satisfied: typing-extensions<5,>=4.11 in c:\users\user\anaconda
3\lib\site-packages (from openai) (4.13.2)
Requirement already satisfied: idna>=2.8 in c:\users\user\anaconda3\lib\site-packag
es (from anyio<5,>=3.5.0->openai) (3.3)
Requirement already satisfied: httpcore==1.* in c:\users\user\anaconda3\lib\site-pa
ckages (from httpx<1,>=0.23.0->openai) (1.0.9)
Requirement already satisfied: certifi in c:\users\user\anaconda3\lib\site-packages
(from httpx<1,>=0.23.0->openai) (2025.4.26)
Requirement already satisfied: h11>=0.16 in c:\users\user\anaconda3\lib\site-packag
es (from httpcore==1.*->httpx<1,>=0.23.0->openai) (0.16.0)
Requirement already satisfied: pydantic-core==2.33.2 in c:\users\user\anaconda3\lib
\site-packages (from pydantic<3,>=1.9.0->openai) (2.33.2)
Requirement already satisfied: annotated-types>=0.6.0 in c:\users\user\anaconda3\li
b\site-packages (from pydantic<3,>=1.9.0->openai) (0.7.0)
Requirement already satisfied: typing-inspection>=0.4.0 in c:\users\user\anaconda3
\lib\site-packages (from pydantic<3,>=1.9.0->openai) (0.4.1)
Requirement already satisfied: colorama in c:\users\user\anaconda3\lib\site-package
s (from tqdm>4->openai) (0.4.5)
```

```
WARNING: Ignoring invalid distribution -rotobuf (c:\users\user\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\user\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\user\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\user\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\user\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\user\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\user\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\user\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\user\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\user\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -rotobuf (c:\users\user\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -pencv-python (c:\users\user\anaconda3\lib\s
ite-packages)
```

In [3]:
```python
# Task 2: Basic CRUD Workflow using SQLite (Course Management App)
import sqlite3

DB = "courses.db"

# Initialize DB
def init_db():
    conn = sqlite3.connect(DB)
    cursor = conn.cursor()
    cursor.execute('''CREATE TABLE IF NOT EXISTS courses
                      (id INTEGER PRIMARY KEY AUTOINCREMENT,
                       name TEXT,
                       description TEXT)''')
    conn.commit()
    conn.close()

init_db()

# Add course
def add_course(name, description):
    conn = sqlite3.connect(DB)
    cursor = conn.cursor()
    cursor.execute("INSERT INTO courses (name, description) VALUES (?, ?)", (name,
    conn.commit()
    conn.close()
    print("Course added successfully!")

# View courses
def view_courses():
    conn = sqlite3.connect(DB)
    cursor = conn.cursor()
    cursor.execute("SELECT * FROM courses")
    rows = cursor.fetchall()
    conn.close()
    return rows

# Update course
def update_course(course_id, name, description):
    conn = sqlite3.connect(DB)
    cursor = conn.cursor()
    cursor.execute("UPDATE courses SET name=?, description=? WHERE id=?", (name, de
    conn.commit()
    conn.close()
    print("Course updated successfully!")

# Delete course
def delete_course(course_id):
    conn = sqlite3.connect(DB)
    cursor = conn.cursor()
    cursor.execute("DELETE FROM courses WHERE id=?", (course_id,))
    conn.commit()
    conn.close()
    print("Course deleted successfully!")

# Demo workflow
add_course("Leadership 101", "Introductory program on women leadership.")
add_course("Advanced Leadership", "In-depth leadership training with industry mento

print("All Courses:", view_courses())
```

```
print( All Courses: , view_courses())

update_course(1, "Leadership 101 - Updated", "Updated intro program.")
print("After Update:", view_courses())

delete_course(2)
print("After Deletion:", view_courses())
```

```
Course added successfully!
Course added successfully!
All Courses: [(1, 'Leadership 101 - Updated', 'Updated intro program.'), (3, 'Leade
rship 101', 'Introductory program on women leadership.'), (4, 'Advanced Leadership'
, 'In-depth leadership training with industry mentors.')]
Course updated successfully!
After Update: [(1, 'Leadership 101 - Updated', 'Updated intro program.'), (3, 'Lead
ership 101', 'Introductory program on women leadership.'), (4, 'Advanced Leadership
', 'In-depth leadership training with industry mentors.')]
Course deleted successfully!
After Deletion: [(1, 'Leadership 101 - Updated', 'Updated intro program.'), (3, 'Le
adership 101', 'Introductory program on women leadership.'), (4, 'Advanced Leadersh
ip', 'In-depth leadership training with industry mentors.')]
```

In [ ]: