

Week 8 — Track 3: Multimodal App Scaffold (Speech + Visualization + Routing)

Route user requests to speech, viz, or project QA tools.

```
def route(query: str) -> str:
    q = query.lower()
    if any(k in q for k in ['plot', 'chart', 'visualize']):
        return 'viz'
    if any(k in q for k in ['audio', 'speak', 'voice']):
        return 'speech'
    return 'qa'
print(route('please plot metric A by year'))
```

viz

Step 1: Setup — install any missing packages

```
# Run only if you don't have them already
!pip install matplotlib gTTS IPython
```

```
Requirement already satisfied: matplotlib in /usr/local/lib/python3.12/dist-packages (3.10.0)
Collecting gTTS
  Downloading gTTS-2.5.4-py3-none-any.whl.metadata (4.1 kB)
Requirement already satisfied: IPython in /usr/local/lib/python3.12/dist-packages (7.34.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.3.3)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (4.60.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (1.4.9)
Requirement already satisfied: numpy>=1.23 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (25.0)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (11.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (3.2.5)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.12/dist-packages (from matplotlib) (2.9.0)
Requirement already satisfied: requests<3,>=2.27 in /usr/local/lib/python3.12/dist-packages (from gTTS) (2.32.4)
Collecting click<8.2,>=7.1 (from gTTS)
  Downloading click-8.1.8-py3-none-any.whl.metadata (2.3 kB)
Requirement already satisfied: setuptools>=18.5 in /usr/local/lib/python3.12/dist-packages (from IPython) (75.2.0)
Collecting jedi>=0.16 (from IPython)
  Downloading jedi-0.19.2-py2.py3-none-any.whl.metadata (22 kB)
Requirement already satisfied: decorator in /usr/local/lib/python3.12/dist-packages (from IPython) (4.4.2)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.12/dist-packages (from IPython) (0.7.5)
Requirement already satisfied: traitlets>=4.2 in /usr/local/lib/python3.12/dist-packages (from IPython) (5.7.1)
Requirement already satisfied: prompt-toolkit!=3.0.0,!<3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.12/dist-packages (from IPython) (3.0.47)
Requirement already satisfied: pygments in /usr/local/lib/python3.12/dist-packages (from IPython) (2.19.2)
Requirement already satisfied: backcall in /usr/local/lib/python3.12/dist-packages (from IPython) (0.2.0)
Requirement already satisfied: matplotlib-inline in /usr/local/lib/python3.12/dist-packages (from IPython) (0.1.7)
Requirement already satisfied: pexpect>4.3 in /usr/local/lib/python3.12/dist-packages (from IPython) (4.9.0)
Requirement already satisfied: parso<0.9.0,>=0.8.4 in /usr/local/lib/python3.12/dist-packages (from jedi>=0.16->IPython) (0.8.4)
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.12/dist-packages (from pexpect>4.3->IPython) (0.7.0)
Requirement already satisfied: wcwidth in /usr/local/lib/python3.12/dist-packages (from prompt-toolkit!=3.0.0,!<3.0.1,<3.1.0,>=2.0.0->IPython) (0.2.13)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.7->matplotlib) (1.17.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests<3,>=2.27->gTTS) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests<3,>=2.27->gTTS) (3.10.1)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests<3,>=2.27->gTTS) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests<3,>=2.27->gTTS) (2025.10.14)
Downloading gTTS-2.5.4-py3-none-any.whl (29 kB)
Downloading click-8.1.8-py3-none-any.whl (98 kB)
98.2/98.2 kB 8.8 MB/s eta 0:00:00
Downloading jedi-0.19.2-py2.py3-none-any.whl (1.6 MB)
1.6/1.6 MB 52.8 MB/s eta 0:00:00
Installing collected packages: jedi, click, gTTS
  Attempting uninstall: click
    Found existing installation: click 8.3.0
    Uninstalling click-8.3.0:
      Successfully uninstalled click-8.3.0
  Successfully installed click-8.1.8 gTTS-2.5.4 jedi-0.19.2
```

```
import matplotlib.pyplot as plt
import numpy as np
from gTTS import gTTS
import IPython.display as ipd
```

```
def guardrails(query):
    """Block unsafe or irrelevant requests."""
    blocked = ["password", "api key", "ssn", "private", "secret"]
    return not any(term in query.lower() for term in blocked)

def multimodal_router(query):
    """Detect intent: Speech, Visualization, or QA."""
    q = query.lower()
    if any(x in q for x in ["speak", "say", "voice", "listen", "audio"]):
        return "speech"
    elif any(x in q for x in ["plot", "chart", "graph", "visualize", "trend", "show"]):
        return "visualization"
    elif any(x in q for x in ["who", "what", "when", "how", "why", "explain", "summarize"]):
        return "qa"
    else:
        return "unknown"
```

```
def speech_tool(text="Hello! This is the speech tool in action."):
    """Convert text to speech and play it."""
    tts = gTTS(text)
    tts.save("response.mp3")
    return ipd.Audio("response.mp3")

def visualization_tool():
    """Generate a simple random plot."""
    data = np.random.randn(20).cumsum()
    plt.figure(figsize=(6,4))
    plt.plot(data, marker="o")
    plt.title("📊 Example Visualization Output")
    plt.xlabel("Step")
    plt.ylabel("Value")
    plt.grid(True)
    plt.show()

def qa_tool(query):
    """Stub for QA – replace with your Week 6/7 backend call."""
    answer = f"🗨️ [QA] Responding to: '{query}' – (connect to project backend)"
    return answer
```

```
def run_router(query):
    """Route to the correct tool based on query intent."""
    if not guardrails(query):
        return "🚫 Unsafe or irrelevant query detected."



    route = multimodal_router(query)
    print(f"🔍 Detected route: {route.upper()}")

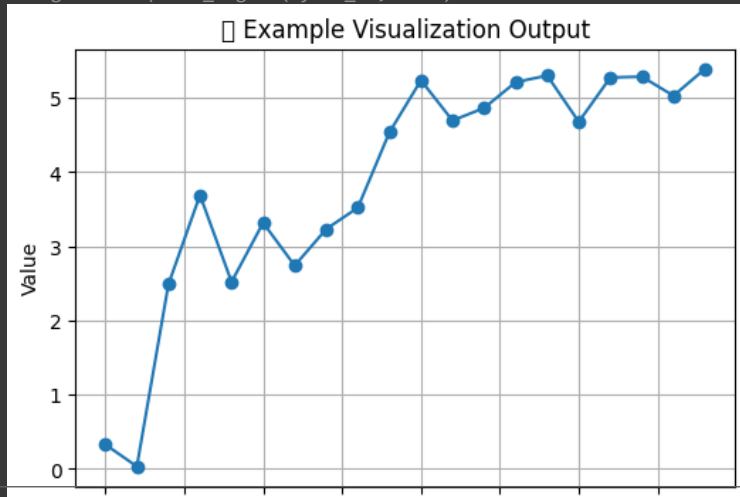
    if route == "speech":
        return speech_tool("This is an example of speech output from the router.")
    elif route == "visualization":
        visualization_tool()
        return "✅ Visualization generated successfully."
    elif route == "qa":
        return qa_tool(query)
    else:
        return "⚠️ Could not determine route. Please rephrase your query."
```

```
# Speech example
run_router("please speak the latest summary")

# Visualization example
run_router("plot accuracy trend over epochs")

# QA example
run_router("what is the diagnosis summary?")
```

 Detected route: SPEECH
 Detected route: VISUALIZATION
 /usr/local/lib/python3.12/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 128202 (\N{BAR CHART}) missing from font.
 fig.canvas.print_figure(bytes_io, **kw)



```

def tool_project_qa(query: str):
    return {'answer': f'(stub QA) {query} → answer with citations'}

def tool_viz(query: str):
    return {'chart': 'chart_displayed_here', 'note': 'Generated with matplotlib'}

def tool_speech(query: str):
    return {'speech': 'handled (stub)'}
  
```

```

BLOCK = ['malware', 'ssn', 'phi']

def guardrails(q: str) -> bool:
    return not any(b in q.lower() for b in BLOCK)

def run(query: str):
    if not guardrails(query):
        return {'error': 'Refused by guardrails.'}
    r = route(query)
    if r=='viz':
        return tool_viz(query)
    if r=='speech':
        return tool_speech(query)
    return tool_project_qa(query)

print(run('plot accuracy by year'))
print(run('answer this question about our dataset'))
  
```

```

{'chart': 'chart_displayed_here', 'note': 'Generated with matplotlib'}
{'answer': '(stub QA) answer this question about our dataset → answer with citations'}
  
```