In [1]:
```
pip install groq python-dotenv
```

```
Collecting groq
  Downloading groq-1.0.0-py3-none-any.whl.metadata (16 kB)
Requirement already satisfied: python-dotenv in /usr/local/lib/python3.12/dist-
packages (1.2.1)
Requirement already satisfied: anyio<5,>=3.5.0 in /usr/local/lib/python3.12/dis
t-packages (from groq) (4.12.1)
Requirement already satisfied: distro<2,>=1.7.0 in /usr/local/lib/python3.12/di
st-packages (from groq) (1.9.0)
Requirement already satisfied: httpx<1,>=0.23.0 in /usr/local/lib/python3.12/di
st-packages (from groq) (0.28.1)
Requirement already satisfied: pydantic<3,>=1.9.0 in /usr/local/lib/python3.12/
dist-packages (from groq) (2.12.3)
Requirement already satisfied: sniffio in /usr/local/lib/python3.12/dist-packag
es (from groq) (1.3.1)
Requirement already satisfied: typing-extensions<5,>=4.10 in /usr/local/lib/pyt
hon3.12/dist-packages (from groq) (4.15.0)
Requirement already satisfied: idna>=2.8 in /usr/local/lib/python3.12/dist-pack
ages (from anyio<5,>=3.5.0->groq) (3.11)
Requirement already satisfied: certifi in /usr/local/lib/python3.12/dist-packag
es (from httpx<1,>=0.23.0->groq) (2026.1.4)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.12/dist-
packages (from httpx<1,>=0.23.0->groq) (1.0.9)
Requirement already satisfied: h11>=0.16 in /usr/local/lib/python3.12/dist-pack
ages (from httpcore==1.*->httpx<1,>=0.23.0->groq) (0.16.0)
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python
3.12/dist-packages (from pydantic<3,>=1.9.0->groq) (0.7.0)
Requirement already satisfied: pydantic-core==2.41.4 in /usr/local/lib/python
3.12/dist-packages (from pydantic<3,>=1.9.0->groq) (2.41.4)
Requirement already satisfied: typing-inspection>=0.4.2 in /usr/local/lib/pytho
n3.12/dist-packages (from pydantic<3,>=1.9.0->groq) (0.4.2)
Downloading groq-1.0.0-py3-none-any.whl (138 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 138.3/138.3 kB 4.1 MB/s eta 0:00:00
Installing collected packages: groq
Successfully installed groq-1.0.0
```

In [13]:
```
import os
from groq import Groq
from getpass import getpass

api_key = getpass("Enter your Groq API Key: ")

try:
    if not api_key.startswith("gsk_"):
        print("Warning: That doesn't look like a standard Groq API key (usuall

    client = Groq(api_key=api_key)
    print("Groq client initialized successfully!")
except Exception as e:
    print(f"Error initializing Groq client: {e}")
```

```
Enter your Groq API Key: ··········
Groq client initialized successfully!
```

In [14]:
```
MODEL_CONFIG = {
    "technical": {
        "system_prompt": "You are a Senior Software Engineer. Provide rigorous
        "model": "llama-3.3-70b-versatile"
    },
    "billing": {
        "system_prompt": "You are a Billing Specialist. Be empathetic and focu
        "model": "llama-3.3-70b-versatile"
    },
    "general": {
        "system_prompt": "You are a helpful and friendly general assistant.",
        "model": "llama-3.3-70b-versatile"
    }
}
```

In [15]:
```
def route_prompt(user_input):
    routing_prompt = f"""
    Classify the following user query into exactly ONE of these categories:
    [technical, billing, general].

    Return ONLY the category name.

    User Query: "{user_input}"
    """

    chat_completion = client.chat.completions.create(
        messages=[{"role": "user", "content": routing_prompt}],
        model="llama-3.3-70b-versatile",
        temperature=0
    )

    category = chat_completion.choices[0].message.content.strip().lower()
    return category
```

In [16]:
```
def process_request(user_input):
    category = route_prompt(user_input)
    print(f"--- [Router detected: {category.upper()}] ---")

    config = MODEL_CONFIG.get(category, MODEL_CONFIG["general"])

    response = client.chat.completions.create(
        messages=[
            {"role": "system", "content": config["system_prompt"]},
            {"role": "user", "content": user_input}
        ],
        model=config["model"],
        temperature=0.7
    )

    return response.choices[0].message.content
```

In [17]:
```
print("User: My python script is throwing an IndexError on line 5.")
print(f"Assistant: {process_request('My python script is throwing an IndexErro
```

```
print("\n" + "="*50 + "\n")

print("User: I was charged twice for my subscription this month.")
print(f"Assistant: {process_request('I was charged twice for my subscription t
print("\n" + "="*50 + "\n")

print("User: Hello! How are you doing today?")
print(f"Assistant: {process_request('Hello! How are you doing today?')}")
```

User: My python script is throwing an IndexError on line 5.
--- [Router detected: TECHNICAL] ---
Assistant: # Step-by-step analysis of the problem:
1. **IndexError in Python** typically occurs when you're trying to access an el
ement in a list or other sequence type using an index that is out of range.
2. **Reviewing the code** is essential to identify the issue. However, since yo
u haven't provided the code, I'll provide a general approach to solving this pr
oblem.
3. **Common causes** include:
   - Accessing an index that is greater than or equal to the length of the lis
t.
   - Using a negative index that is less than the negative length of the list.
   - Attempting to access an index in an empty list.

# Fixed solution:
Without your specific code, I'll provide an example of how to fix an `IndexErro
r` in a simple list access scenario:
```python
# Example of a list with 5 elements
my_list = [1, 2, 3, 4, 5]

# Incorrect way to access the list (this will throw an IndexError)
# print(my_list[5])  # IndexError: list index out of range

# Correct way to access the list
print(my_list[4])  # Outputs: 5

# Alternatively, you can use a try-except block to handle the error
try:
    print(my_list[5])
except IndexError:
    print("Error: Index out of range")
```

# Explanation of changes:
*   **Checking the index**: Before accessing an element in a list, ensure that
the index is within the valid range (0 to length - 1).
*   **Using a try-except block**: This can help catch and handle `IndexError` e
xceptions, providing a way to recover from the error or provide a meaningful er
ror message.

# Tests and example uses:
To avoid `IndexError`, you can use the following techniques:
*   **Check the length of the list**: Use `len(my_list)` to get the number of e
lements in the list.
*   **Use a loop to iterate over the list**: This can help avoid accessing indi
ces directly.
*   **Use list methods**: Methods like `append()`, `insert()`, and `index()` ca
n help modify and access list elements safely.

```python
# Example of using a loop to iterate over the list
for i, element in enumerate(my_list):
    print(f"Index: {i}, Element: {element}")
```

```
# Example of using list methods
my_list.append(6)  # Add an element to the end of the list
print(my_list)  # Outputs: [1, 2, 3, 4, 5, 6]
```

==========

User: I was charged twice for my subscription this month.
--- [Router detected: BILLING] ---
Assistant: I'm so sorry to hear that you were charged twice for your subscripti
on this month. I can imagine how frustrating that must be for you. I'm here to
help you resolve this issue as quickly as possible.

Can you please provide me with more details about the duplicate charge? For exa
mple, what is the date of the charges, and what is the amount that was deducted
from your account? Also, have you received any email confirmations or receipts
for these transactions?

Our company's financial policy is to promptly refund any duplicate or incorrect
charges. I'll do my best to investigate this matter and ensure that the correct
amount is refunded to you. If you have any questions or concerns, please don't
hesitate to ask. Your satisfaction is our top priority.

==========

User: Hello! How are you doing today?
--- [Router detected: GENERAL] ---
Assistant: Hello. I'm doing great, thanks for asking. It's wonderful to connect
with you. Is there anything I can help you with or would you like to chat for a
bit? I'm all ears and here to assist you in any way I can.