



School of Computer Science and Artificial Intelligence

Lab Assignment-4.1

Course Title : **AI Assistant Coding**
Name of Student : **Shaik Naved Ahmed**
Enrollment No. : **2303A54053**
Batch No. : **47-B**

Problem Statement 1: Customer Email Classification

A company receives a large number of customer emails every day and wants to automatically classify them into the following categories:

- Billing
- Technical Support
- Feedback
- Others

Instead of training a new machine learning model, the company decides to use prompt engineering techniques with an existing large language model.

Tasks

1. Prepare five short sample emails, each belonging to one of the above categories.
2. Write a zero-shot prompt to classify a given email into one of the categories without providing any examples.
3. Write a one-shot prompt by including one labeled email example and ask the model to classify a new email.
4. Write a few-shot prompt by including two or three labeled email examples and ask the model to classify a new email.
5. Compare the outputs obtained using zero-shot, one-shot, and few-shot prompting techniques and briefly comment on their effectiveness

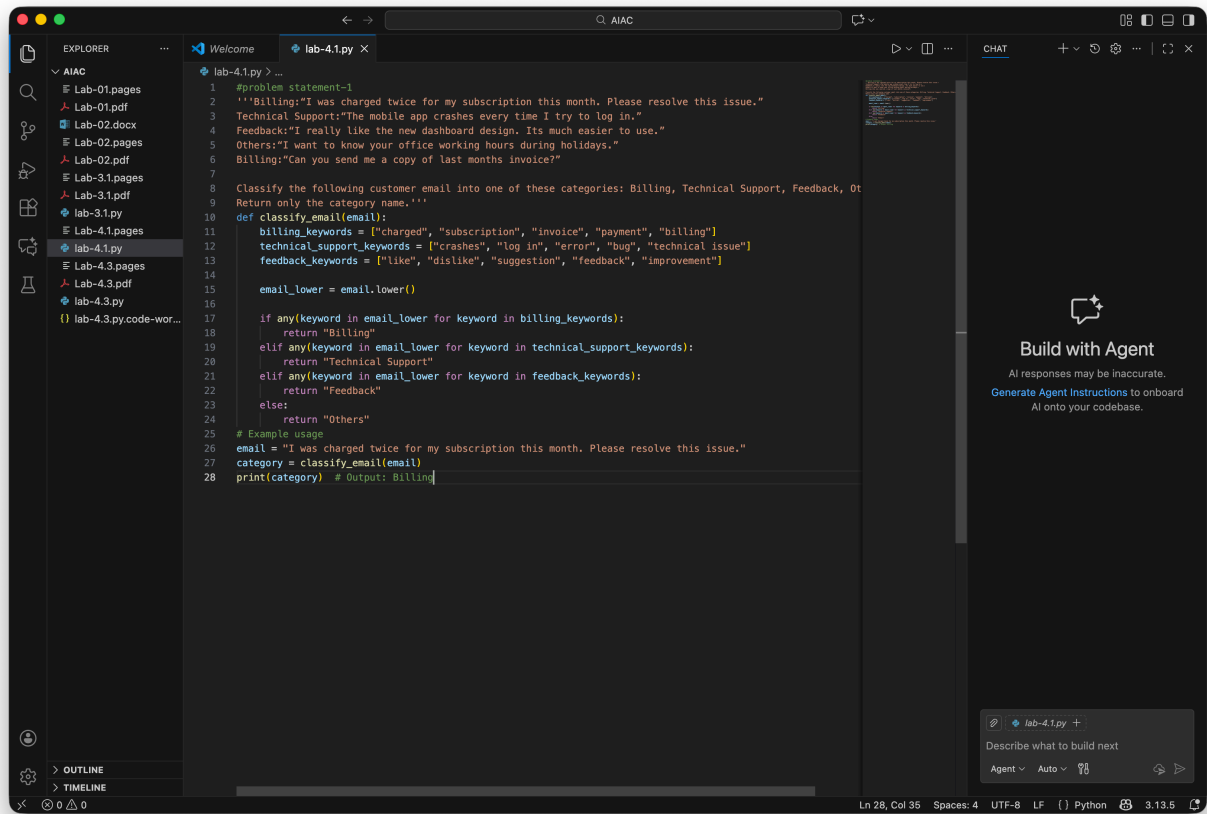
1. Sample Customer Emails (5)

1. **Billing:**
“I was charged twice for my subscription this month. Please resolve this issue.”
2. **Technical Support:**
“The mobile app crashes every time I try to log in.”
3. **Feedback:**
“I really like the new dashboard design. It’s much easier to use.”
4. **Others:**
“I want to know your office working hours during holidays.”
5. **Billing:**
“Can you send me a copy of last month’s invoice?”

2. Zero-Shot Prompt

Classify the following customer email into one of these categories: Billing, Technical Support, Feedback, Others.
Return only the category name.

SCREENSHOT OF GENERATED CODE:



INPUT & OUTPUT:

Input Email	Output	Reason
"I was charged twice for my subscription this month."	Billing	Contains keywords charged and subscription.

EXPLANATION:

The function classifies an email by checking for specific keywords related to billing, technical support, and feedback. It converts the email to lowercase to avoid case-sensitivity issues and then matches keywords using conditional checks. If no keywords are found, the email is categorized as Others.

3. One-Shot Prompt

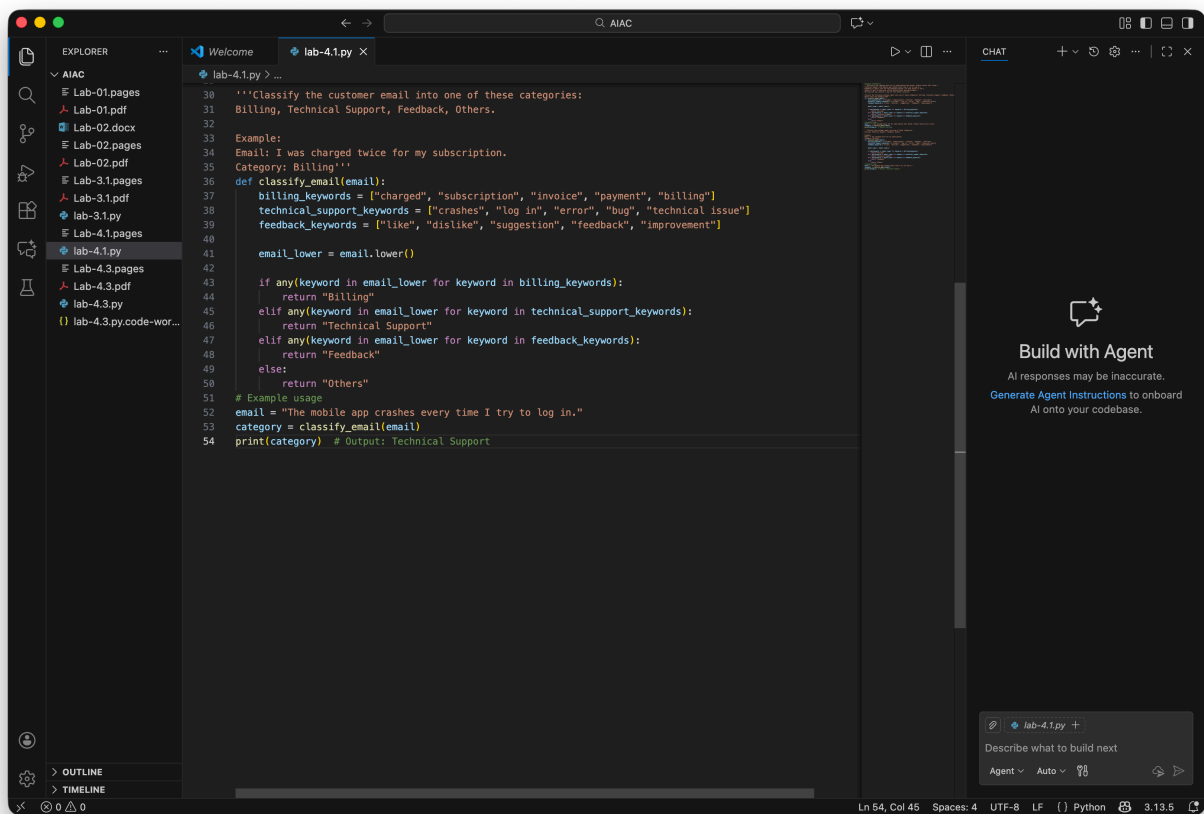
Classify the customer email into one of these categories:
Billing, Technical Support, Feedback, Others.

Example:

Email: *I was charged twice for my subscription.*

Category: Billing

SCREENSHOT OF GENERATED CODE:



The screenshot shows a VS Code editor window with a file explorer on the left, a code editor in the center, and a chat panel on the right. The code editor displays a Python script named `lab-4.1.py` that defines a `classify_email` function. The function takes an email string as input and returns a category based on predefined keywords. The keywords are grouped into three lists: `billing_keywords`, `technical_support_keywords`, and `feedback_keywords`. The function converts the email to lowercase and checks for keywords in each list in order. If a keyword is found, it returns the corresponding category; otherwise, it returns "Others". An example usage is provided at the bottom of the script.

```
30 '''Classify the customer email into one of these categories:
31 Billing, Technical Support, Feedback, Others.
32
33 Examples:
34 Email: I was charged twice for my subscription.
35 Category: Billing'''
36 def classify_email(email):
37     billing_keywords = ["charged", "subscription", "invoice", "payment", "billing"]
38     technical_support_keywords = ["crashes", "log in", "error", "bug", "technical issue"]
39     feedback_keywords = ["like", "dislike", "suggestion", "feedback", "improvement"]
40
41     email_lower = email.lower()
42
43     if any(keyword in email_lower for keyword in billing_keywords):
44         return "Billing"
45     elif any(keyword in email_lower for keyword in technical_support_keywords):
46         return "Technical Support"
47     elif any(keyword in email_lower for keyword in feedback_keywords):
48         return "Feedback"
49     else:
50         return "Others"
51
52 # Example usage
53 email = "The mobile app crashes every time I try to log in."
54 category = classify_email(email)
55 print(category) # Output: Technical Support
```

INPUT & OUTPUT:

Input Email	Output
"The mobile app crashes every time I try to log in."	Technical Support

EXPLANATION:

The function classifies an email by searching for predefined keywords related to Billing, Technical Support, and Feedback. It converts the email text to lowercase to ensure accurate matching and checks each keyword list in order. If a matching keyword is found, the corresponding category is returned; otherwise, the email is classified as Others.

4. Few-Shot Prompt

Classify the customer email into one of these categories:
Billing, Technical Support, Feedback, Others.

Examples:

Email: *My card was charged extra this month.*

Category: Billing

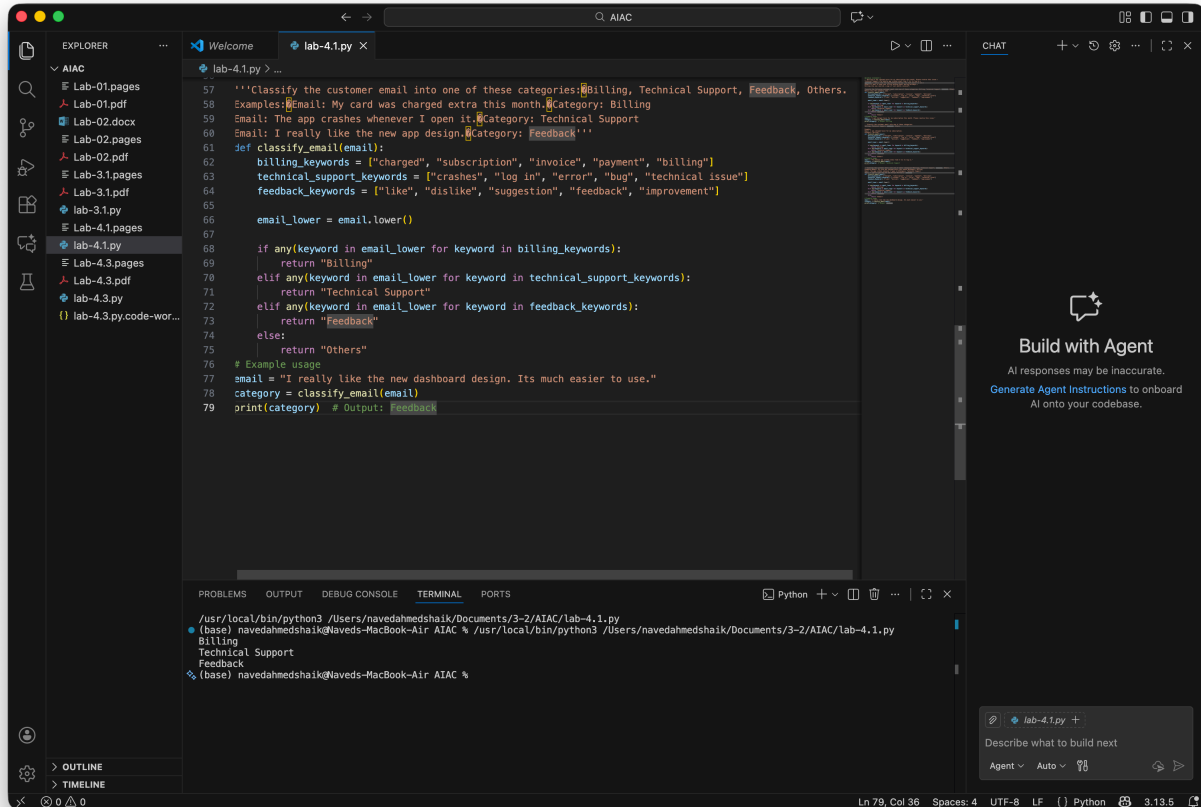
Email: *The app crashes whenever I open it.*

Category: Technical Support

Email: *I really like the new app design.*

Category: Feedback

SCREENSHOT OF GENERATED CODE:



```
57 '''Classify the customer email into one of these categories: Billing, Technical Support, Feedback, Others.
58 Examples: Email: My card was charged extra this month. Category: Billing
59 Email: The app crashes whenever I open it. Category: Technical Support
60 Email: I really like the new app design. Category: Feedback'''
61 def classify_email(email):
62     billing_keywords = ["charged", "subscription", "invoice", "payment", "billing"]
63     technical_support_keywords = ["crashes", "log in", "error", "bug", "technical issue"]
64     feedback_keywords = ["like", "dislike", "suggestion", "feedback", "improvement"]
65
66     email_lower = email.lower()
67
68     if any(keyword in email_lower for keyword in billing_keywords):
69         return "Billing"
70     elif any(keyword in email_lower for keyword in technical_support_keywords):
71         return "Technical Support"
72     elif any(keyword in email_lower for keyword in feedback_keywords):
73         return "Feedback"
74     else:
75         return "Others"
76
77 # Example usage
78 email = "I really like the new dashboard design. Its much easier to use."
79 category = classify_email(email)
80 print(category) # Output: Feedback
```

Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard
AI onto your codebase.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
/usr/local/bin/python3 /Users/navedahmedshaik/Documents/3-2/AIAC/lab-4.1.py
(base) navedahmedshaik@Naveds-MacBook-Air AIAC % /usr/local/bin/python3 /Users/navedahmedshaik/Documents/3-2/AIAC/lab-4.1.py
Billing
Technical Support
Feedback
(base) navedahmedshaik@Naveds-MacBook-Air AIAC %
```

Ln 79, Col 36 Spaces: 4 UTF-8 LF Python 3.13.5

INPUT & OUTPUT:

Input Email	Output
"I really like the new dashboard design. Its much easier to use."	Feedback

EXPLANATION:

The function converts the email text to lowercase and checks for predefined keywords related to billing, technical support, and feedback. Since the email contains the word "like", which is a feedback keyword, it is classified as Feedback.

5. Comparison & Effectiveness

Prompting Technique	Accuracy	Clarity	Reliability	Remarks
Zero-Shot Prompting	Medium	Depends on email clarity	Low	May misclassify vague or mixed emails
One-Shot Prompting	High	Clear	Medium	Single example improves understanding
Few-Shot Prompting	Very High	Very clear	High	Multiple examples give best classification

Problem Statement 2: Intent Classification for Chatbot Queries

A company wants to deploy a chatbot to handle customer queries. Each query must be classified into one of the following intents:

Account Issue, Order Status, Product Inquiry, or General Question using prompt engineering techniques.

Tasks to be Completed

1. Prepare Sample Data Create 6 short chatbot user queries, each mapped to one of the four intents.

2. Zero-shot Prompting

Design a prompt that asks the LLM to classify a user query into the given intent categories without examples.

3. One-shot Prompting

Provide one labeled query in the prompt before classifying a new query.

4. Few-shot Prompting

Include 3–5 labeled intent examples to guide the LLM before classifying a new query.

5. Evaluation

Apply all three techniques to the same set of test queries and document differences in performance.

1. Sample Chatbot User Queries

Query No.	User Query	Intent
1	“I can’t log into my account even after resetting my password.”	Account Issue
2	“Where is my order? It was supposed to arrive yesterday.”	Order Status
3	“Does this phone support 5G and wireless charging?”	Product Inquiry
4	“What are your customer support working hours?”	General Question
5	“My account was locked after multiple failed login attempts.”	Account Issue
6	“Can I track my order using my mobile number?”	Order Status

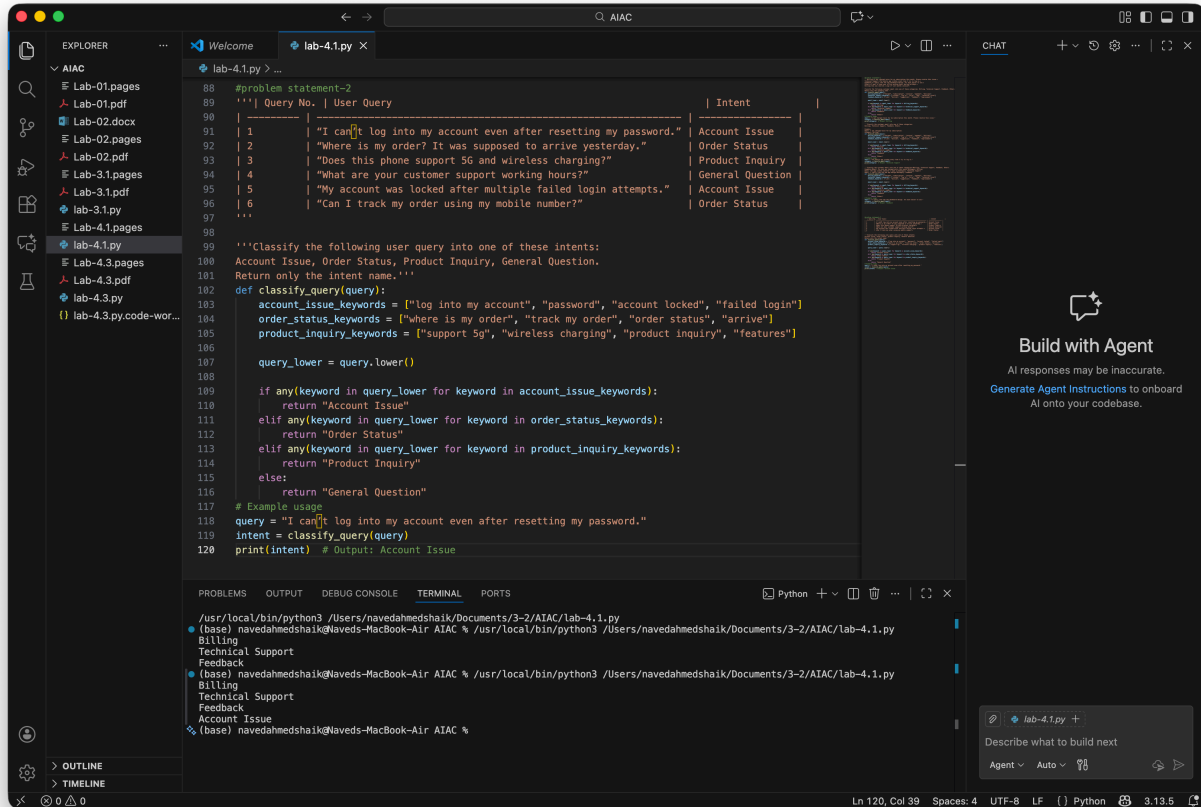
2. Zero-Shot Prompt

Classify the following user query into one of these intents: Account Issue, Order Status, Product Inquiry, General Question. Return only the intent name.

INPUT & OUTPUT:

Input Query	Output	Reason
I can’t log into my account even after resetting my password.	Account Issue	Contains keywords “log into my account” and “password”.

SCREENSHOT OF GENERATED CODE:



```
88 #problem statement-2
89 '''| Query No. | User Query | Intent |
90 |-----|-----|-----|
91 | 1 | "I can't log into my account even after resetting my password." | Account Issue |
92 | 2 | "Where is my order? It was supposed to arrive yesterday." | Order Status |
93 | 3 | "Does this phone support 5G and wireless charging?" | Product Inquiry |
94 | 4 | "What are your customer support working hours?" | General Question |
95 | 5 | "My account was locked after multiple failed login attempts." | Account Issue |
96 | 6 | "Can I track my order using my mobile number?" | Order Status |
97 ...
98
99 '''Classify the following user query into one of these intents:
100 Account Issue, Order Status, Product Inquiry, General Question.
101 Return only the intent name.'''
102
103 def classify_query(query):
104     account_issue_keywords = ["log into my account", "password", "account locked", "failed login"]
105     order_status_keywords = ["where is my order", "track my order", "order status", "arrive"]
106     product_inquiry_keywords = ["support 5g", "wireless charging", "product inquiry", "features"]
107
108     query_lower = query.lower()
109
110     if any(keyword in query_lower for keyword in account_issue_keywords):
111         return "Account Issue"
112     elif any(keyword in query_lower for keyword in order_status_keywords):
113         return "Order Status"
114     elif any(keyword in query_lower for keyword in product_inquiry_keywords):
115         return "Product Inquiry"
116     else:
117         return "General Question"
118
119 # Example usage
120 query = "I can't log into my account even after resetting my password."
121 intent = classify_query(query)
122 print(intent) # Output: Account Issue
```

EXPLANATION

The function converts the query to lowercase and checks it against predefined keyword lists for different categories. If any keyword related to account issues, order status, or product inquiries is found, it returns the corresponding category. If no keywords match, it classifies the query as a General Question

3. One-Shot Prompt

Classify the following user query into one of these intents:
Account Issue, Order Status, Product Inquiry, General Question.

Example:

User Query: *I forgot my password and cannot access my account.*

Intent: Account Issue

INPUT & OUTPUT:

Input Query	Output	Reason
Where is my order? It was supposed to arrive yesterday.	Order Status	Contains keywords “where is my order” and “arrive”.

SCREENSHOT OF GENERATED CODE:

```
'''Classify the following user query into one of these intents: Account Issue, Order Status, Product Inquiry
Example: User Query: I forgot my password and cannot access my account. Intent: Account Issue'''
def classify_query(query):
    account_issue_keywords = ["log into my account", "password", "account locked", "failed login"]
    order_status_keywords = ["where is my order", "track my order", "order status", "arrive"]
    product_inquiry_keywords = ["support 5g", "wireless charging", "product inquiry", "features"]

    query_lower = query.lower()

    if any(keyword in query_lower for keyword in account_issue_keywords):
        return "Account Issue"
    elif any(keyword in query_lower for keyword in order_status_keywords):
        return "Order Status"
    elif any(keyword in query_lower for keyword in product_inquiry_keywords):
        return "Product Inquiry"
    else:
        return "General Question"

# Example usage
query = "Where is my order? It was supposed to arrive yesterday."
intent = classify_query(query)
print(intent) # Output: Order Status
```

Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard
AI onto your codebase.

Python + Python 3.13.5

EXPLANATION:

The function converts the query to lowercase and checks it against predefined keyword lists. Since the query contains order-related keywords, it is classified as Order Status.

4. Few-Shot Prompt

Classify the following user query into one of these intents:
Account Issue, Order Status, Product Inquiry, General Question.

Examples:

User Query: *My account got locked after too many login attempts.*

Intent: Account Issue

User Query: *When will my order be delivered?*

Intent: Order Status

User Query: *Does this laptop come with a warranty?*

Intent: Product Inquiry

User Query: *What payment methods do you accept?*

Intent: General Question

SCREENSHOT OF GENERATED CODE:

```
150 Account Issue, Order Status, Product Inquiry, General Question.
151
152 Examples:
153 User Query: My account got locked after too many login attempts.
154 Intent: Account Issue
155
156 User Query: When will my order be delivered?
157 Intent: Order Status
158
159 User Query: Does this laptop come with a warranty?
160 Intent: Product Inquiry
161
162 User Query: What payment methods do you accept?
163 Intent: General Question'''
164 def classify_query(query):
165     account_issue_keywords = ["log into my account", "password", "account locked", "failed login"]
166     order_status_keywords = ["where is my order", "track my order", "order status", "arrive"]
167     product_inquiry_keywords = ["support 5g", "wireless charging", "product inquiry", "features"]
168
169     query_lower = query.lower()
170
171     if any(keyword in query_lower for keyword in account_issue_keywords):
172         return "Account Issue"
173     elif any(keyword in query_lower for keyword in order_status_keywords):
174         return "Order Status"
175     elif any(keyword in query_lower for keyword in product_inquiry_keywords):
176         return "Product Inquiry"
177     else:
178         return "General Question"
179
180 # Example usage
181 query = "Does this phone support 5G and wireless charging?"
182 intent = classify_query(query)
183 print(intent) # Output: Product Inquiry
```

```
(base) navedahmedshaik@Naveds-MacBook-Air AIAC % /usr/local/bin/python3 /Users/naledahmedshaik/Documents/3-2/AIAC/lab-4.1.py
Billing
Technical Support
Feedback
Account Issue
Order Status
(base) navedahmedshaik@Naveds-MacBook-Air AIAC % /usr/local/bin/python3 /Users/naledahmedshaik/Documents/3-2/AIAC/lab-4.1.py
Billing
Technical Support
Feedback
Account Issue
Order Status
Product Inquiry
(base) navedahmedshaik@Naveds-MacBook-Air AIAC %
```

INPUT & OUTPUT:

Input Query	Output	Reason
Where is my order? It was supposed to arrive yesterday.	Order Status	Contains keywords “where is my order” and “arrive”.

EXPLANATION:

The function converts the query to lowercase and checks it against predefined keyword lists. Since the query contains order-related keywords, it is classified as Order Status.

5. Evaluation of Prompting Techniques

Technique	Accuracy	Intent Understanding	Consistency	Remarks
Zero-Shot	Medium	Basic	Low	Works only for very clear queries
One-Shot	High	Better	Medium	Example improves intent recognition
Few-Shot	Very High	Strong	High	Best performance for ambiguous queries