ASSIGNMENT – 4.5

NAME:SD SUFIYAN

NO:2303A51980

BATCH NO:30

ADVANCED PROMPT ENGINEERING: ZERO-SHOT, ONE-SHOT & FEW- SHOT

TASK-1:

ZERO-SHOT

A. Preparing Sample data:

```
test_emails = [
"My payment failed but money was deducted.",
"The app is not opening on my phone.",
"Great customer service, very satisfied.",
"What is your customer care number?",
"Invoice amount seems incorrect."
]
```

Expected Labels (for evaluation):

```
true_labels =
[ "Billing",
"Technical Support",                                    "Feedback",
"Others",
"Billing"
]
```

PROMPT:

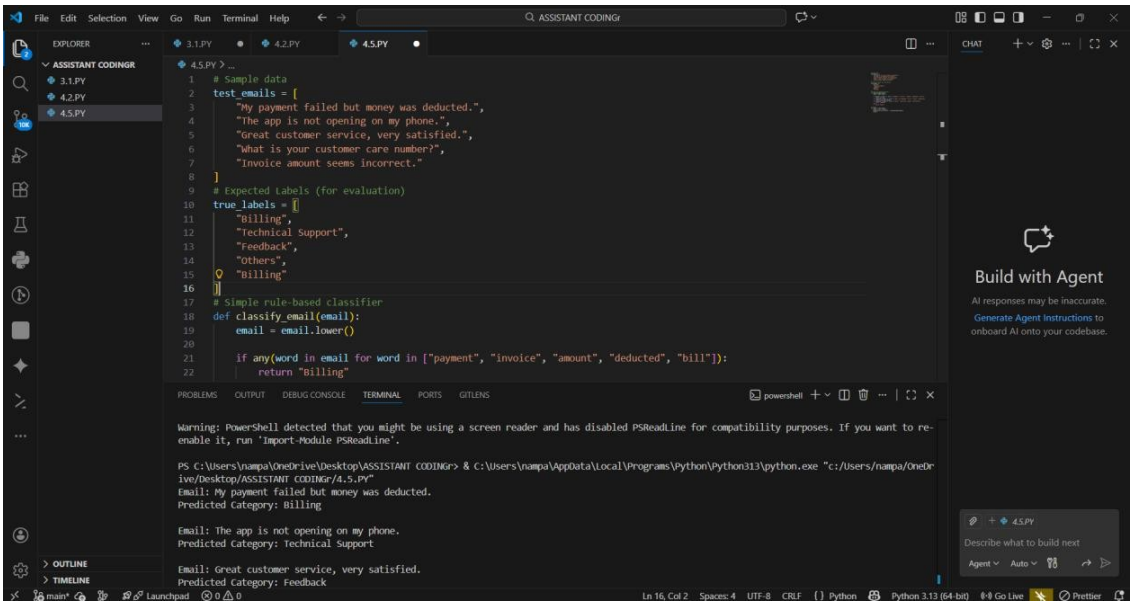
Classify the following email into one of the categories:

Billing, Technical Support, Feedback, Others.

Email: "<email_text>"

Return only the category name.

CODE:



OBSERVATION:

Classifies emails using only instructions, without examples.

Works if keywords are clear, may misclassify ambiguous emails. Quick and simple, but less accurate for complex cases.

ONE SHOT:

PROMPT: Example:

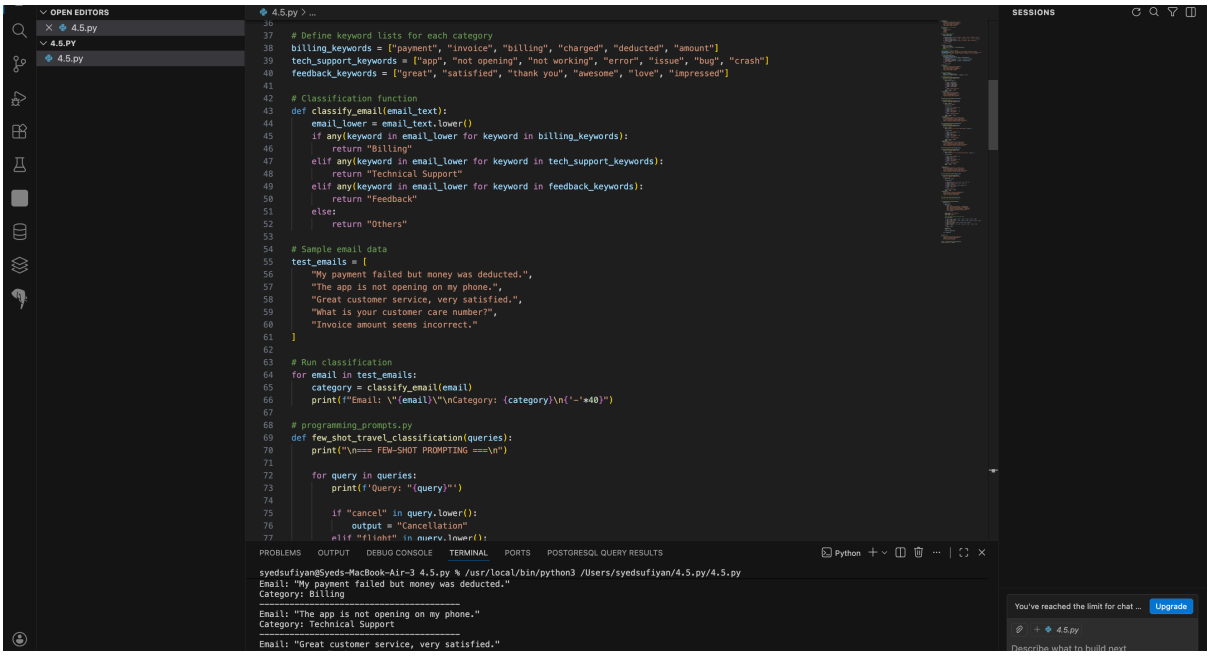
Email: "I was charged twice for my last payment."

Category: Billing

Now classify the following email:

Email: “<email_text>”

CODE:



OBSERVATION:

Provides one example to guide the AI's reasoning.

Improves accuracy over zero-shot and handles slightly ambiguous emails better.

Still limited; accuracy depends on how representative the example is.

One example helps the AI understand the expected format and category mapping.

Classification accuracy improves compared to zero-shot, especially for similar issues.

Performance depends heavily on how relevant the single example is to the new email.

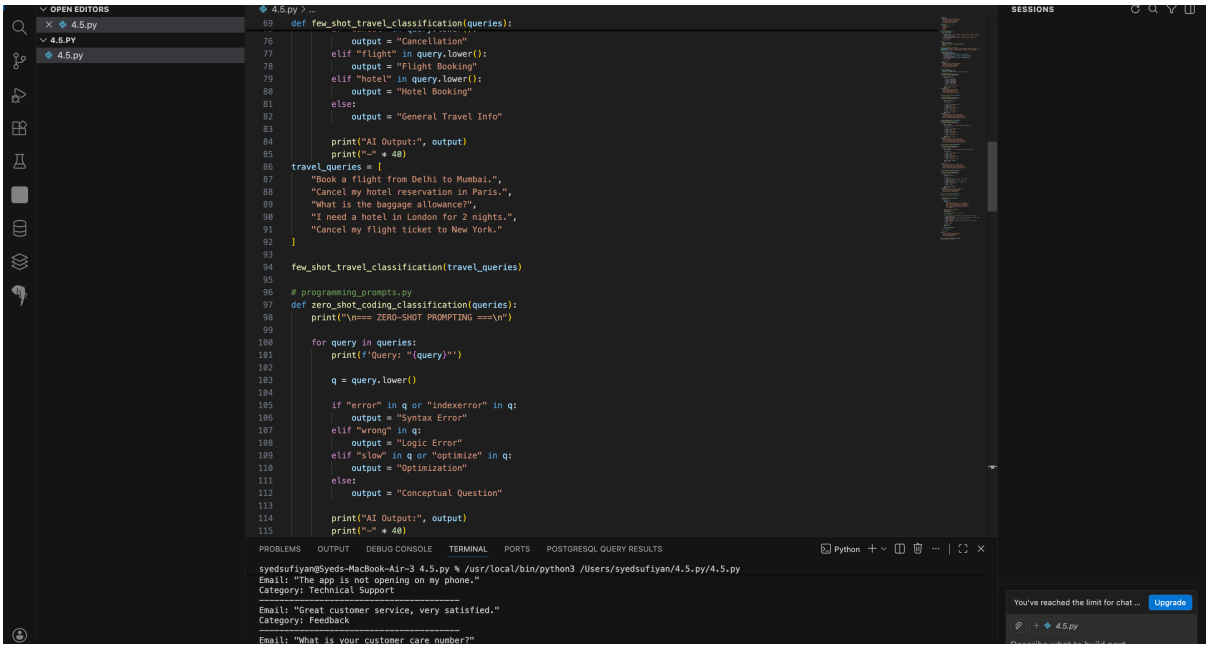
Few shot:

PROMPT: Email: "I was charged twice for my last payment." → Billing

Email: "The app crashes on login." → Technical Support

Email: "I love the new update." → Feedback

Email: "What are your office hours?" → Others



OBSERVATION:

Provides multiple examples to show patterns to the AI. Highest accuracy;

AI can generalize better for unseen emails.

Slightly longer prompts but most reliable for real-world use

TASK-2:

Sample travel queries (short & simple)

travel_queries = [

"Book a flight from Delhi to Mumbai.",

"Cancel my hotel reservation in Paris.",

"What is the baggage allowance?",

"I need a hotel in London for 2 nights.",

"Cancel my flight ticket to New York."

]

True labels for evaluation

true_labels = [

"Flight Booking",

"Cancellation",

"General Travel Info",

"Hotel Booking",

"Cancellation"

]

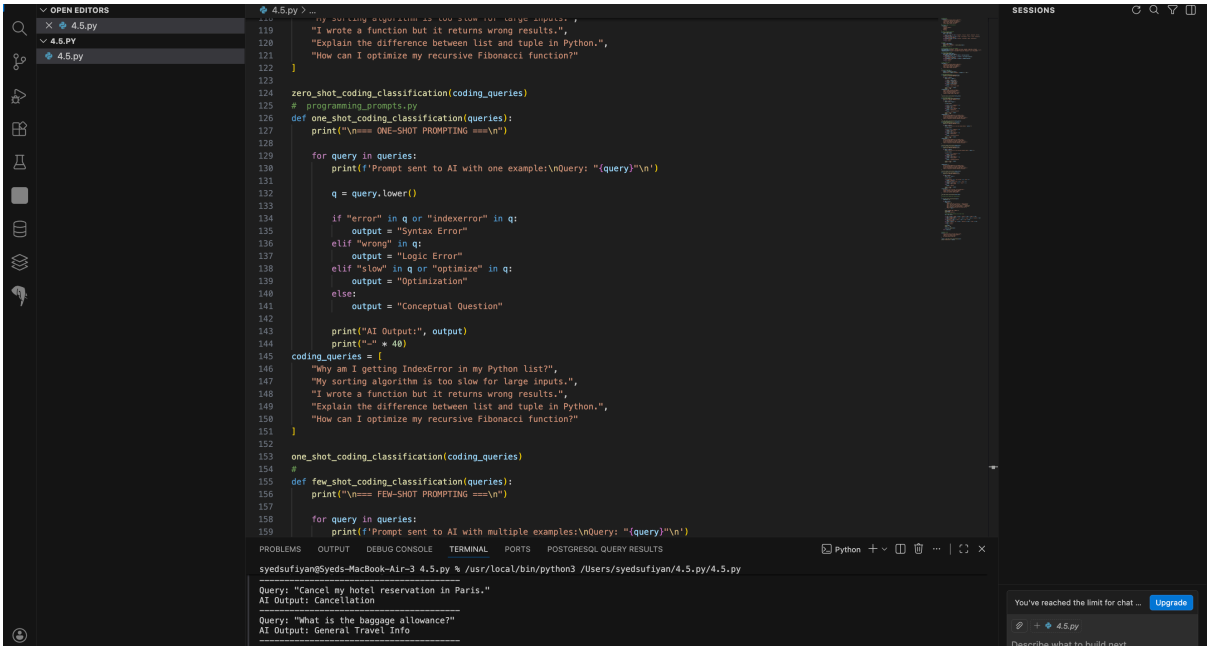
ZERO-SHOT:

PROMPT:Classify the the following travel query into one of the

categories:

Flight Booking, Hotel Booking, Cancellation, General Travel Info.
Query: "<travel_query>"

CODE:



OBSERVATION:

Classifies queries using only instructions, without examples.

Works for obvious keywords like “flight” or “cancel”, may misclassify tricky queries.

Fast and simple, but accuracy is lower for ambiguous cases.

one-shot:

PROMPT:Example:

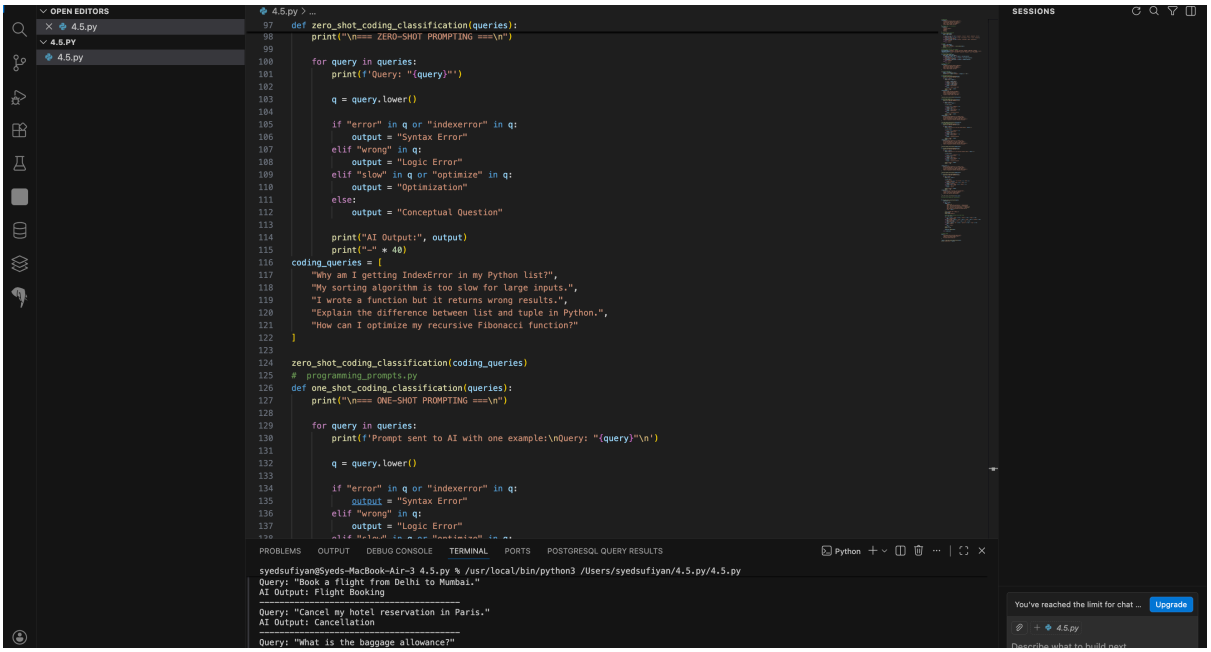
Query: "Cancel my flight ticket."

Category: Cancellation

Now classify the following query:

Query: "<travel_query>"

CODE:



OBSERVATION:

Provides one example to guide AI's reasoning.

Improves accuracy and handles slightly ambiguous queries

better.

Accuracy depends on how representative the example is.

FEW SHOT:

PROMPT:

Examples:

Query: "Book a flight to Mumbai."

Category: Flight Booking
Query: "Cancel my hotel reservation."

Category: Cancellation

Query: "I need a hotel in London."

Category: Hotel Booking

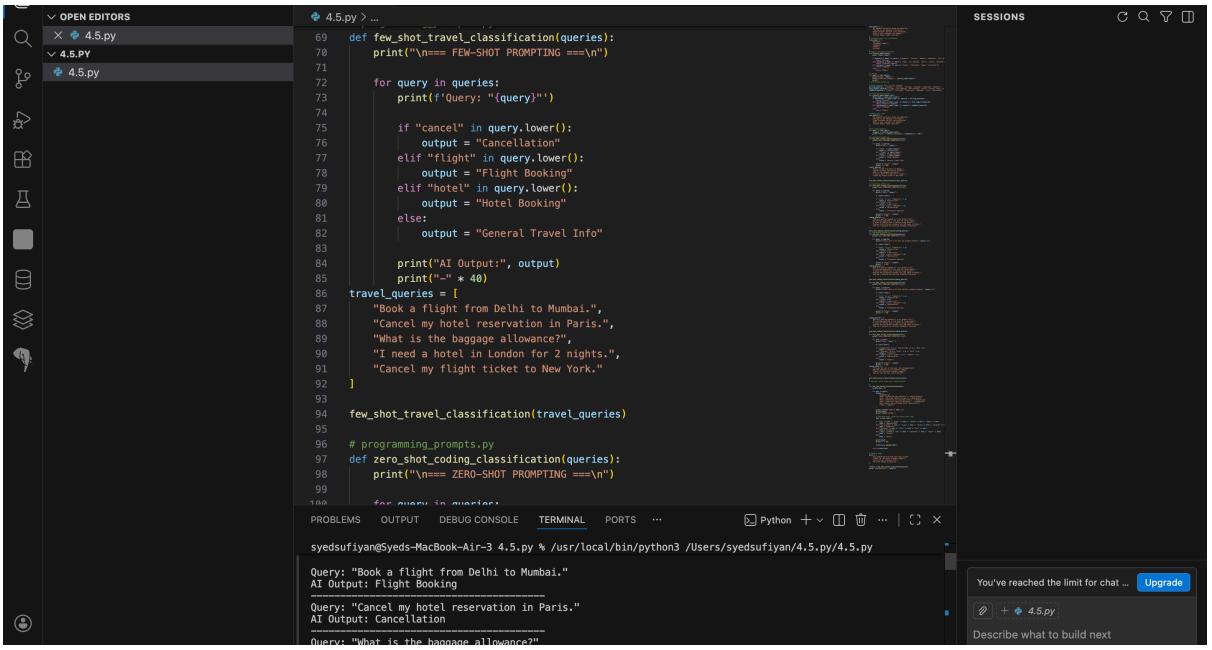
Query: "What is the baggage allowance?"

Category: General Travel Info

Now classify the following query:

Query: "<travel_query>"

CODE:



OBSERVATION:

Provides multiple examples to show patterns to AI.

Highest accuracy; AI generalizes better for unseen queries.

Slightly longer prompts but most reliable for real-world use.

TASK-3:

SAMPLE DATA:

Sample coding queries (short & simple)

```
coding_queries = [
"Why am I getting IndexError in my Python list?",
"My sorting algorithm is too slow for large inputs.",
"I wrote a function but it returns wrong results.",
"Explain the difference between list and tuple in Python.",
"How can I optimize my recursive Fibonacci function?"
]
# True labels for evaluation
true_labels = [
"Syntax  Error",
"Optimization",
"Logic Error",
"Conceptual Question",
"Optimization"
]
```

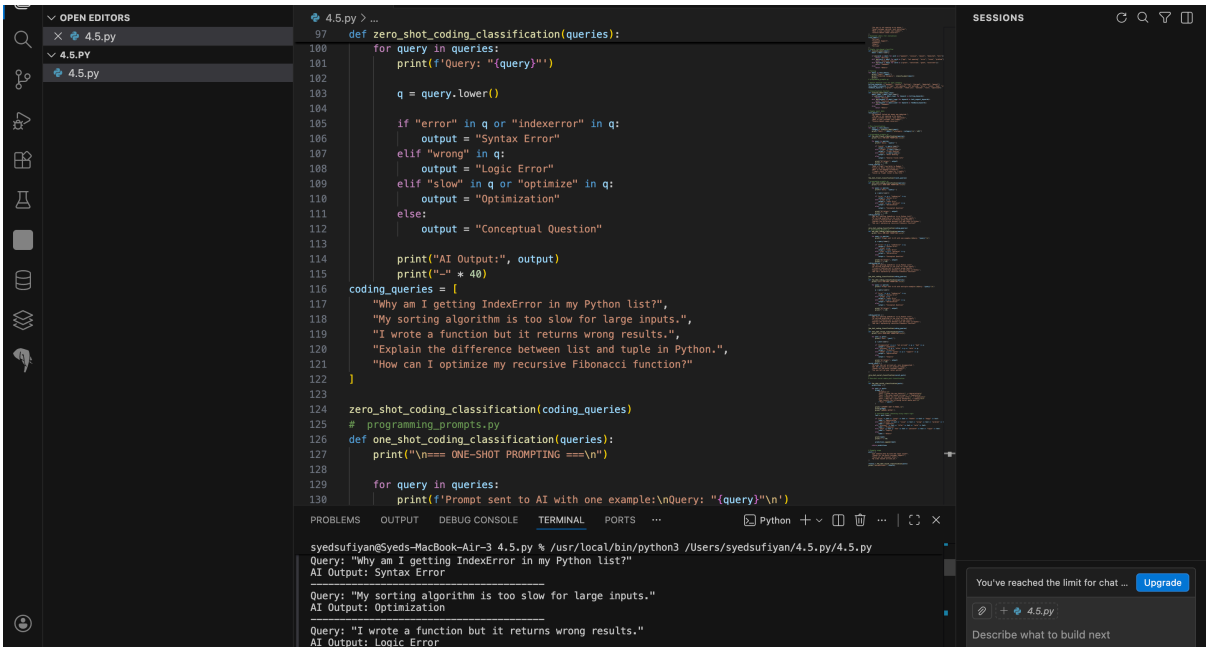
ZERO-SHOT

PROMPT:Classify the following coding query into one of the

categories:

Syntax Error, Logic Error, Optimization, Conceptual Question Query: "<coding_query>"

CODE:



Observation (Zero-shot Prompting)

- Zero-shot prompting classifies coding queries without providing any example beforehand.
- The classification is based only on keywords present in the query.
- Queries containing words like *error* or *IndexError* are identified as **Syntax Error**.
- Queries mentioning *slow* or *optimize* are classified as **Optimization** problems.
- Queries related to incorrect outputs are categorized as **Logic Error**.
- Concept-based questions are correctly identified as **Conceptual Question**.
- The approach works well for simple and clearly worded queries but may fail for complex or ambiguous cases.

ONE SHOT

PROMPT:

Example:

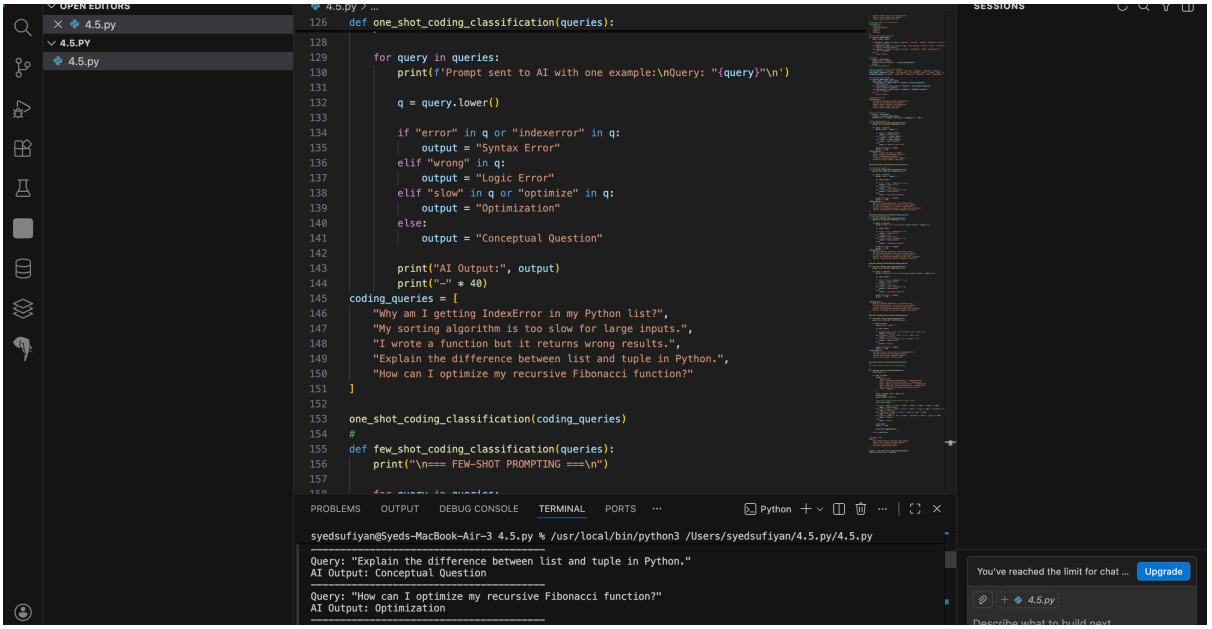
Query: "I want to cancel my Python function."

Category: Logic Error

Now classify the following coding query:

Query: "<coding_query>"

CODE:



OBSERVATION:

Provides one example to guide AI's reasoning.

Improves accuracy and handles slightly ambiguous queries better.

Accuracy depends on how representative the single example is.

FEW SHOT

PROMPT:

EXAMPLES:

Query: "Why does my Python list give IndexError?"

Category: Syntax Error

Query: "My function returns wrong output."

Category: Logic Error Query: "My loop is too slow for large data."

Category: Optimization

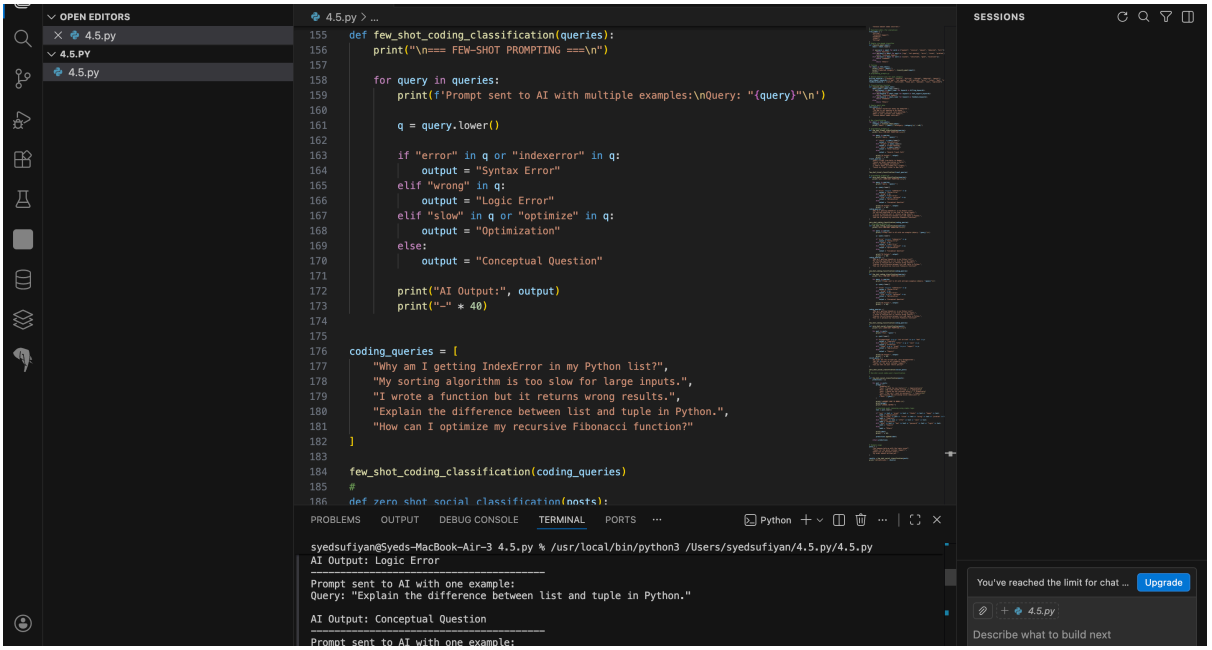
Query: "Explain Python variable scopes."

Category: Conceptual Question

Now classify the following coding query:

Query: "<coding_query

CODE:



OBSERVATION:

Provides multiple examples showing patterns to AI.

Highest accuracy; AI generalizes better for unseen queries.

Slightly longer prompts but most reliable for technical classification

TASK-4

ZERO-SHOT

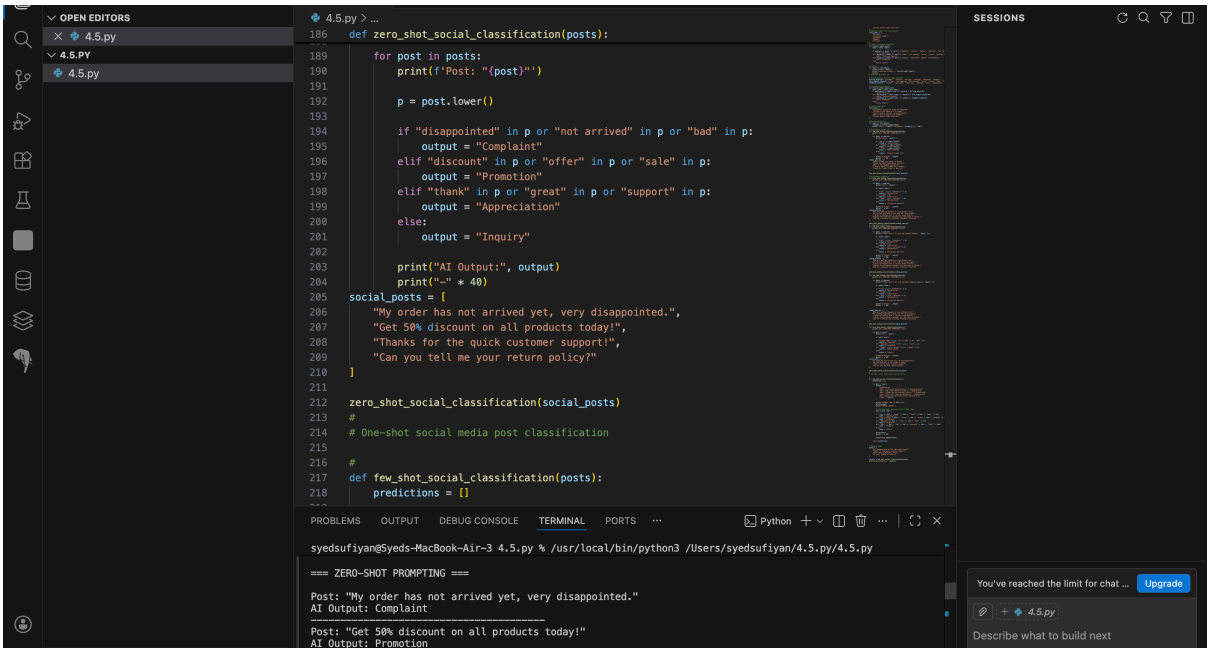
PROMPT:

Classify the following social media post into one of the categories:

Promotion, Complaint, Appreciation, Inquiry.

Post: "<social_post>"

CODE:



OBSERVATION:

Classifies posts using only instructions, without examples.

Works for clear keywords but may misinterpret informal or slang language.

Fast and simple, lower accuracy for ambiguous or sarcastic posts.

ONE-SHOT

PROMPT:

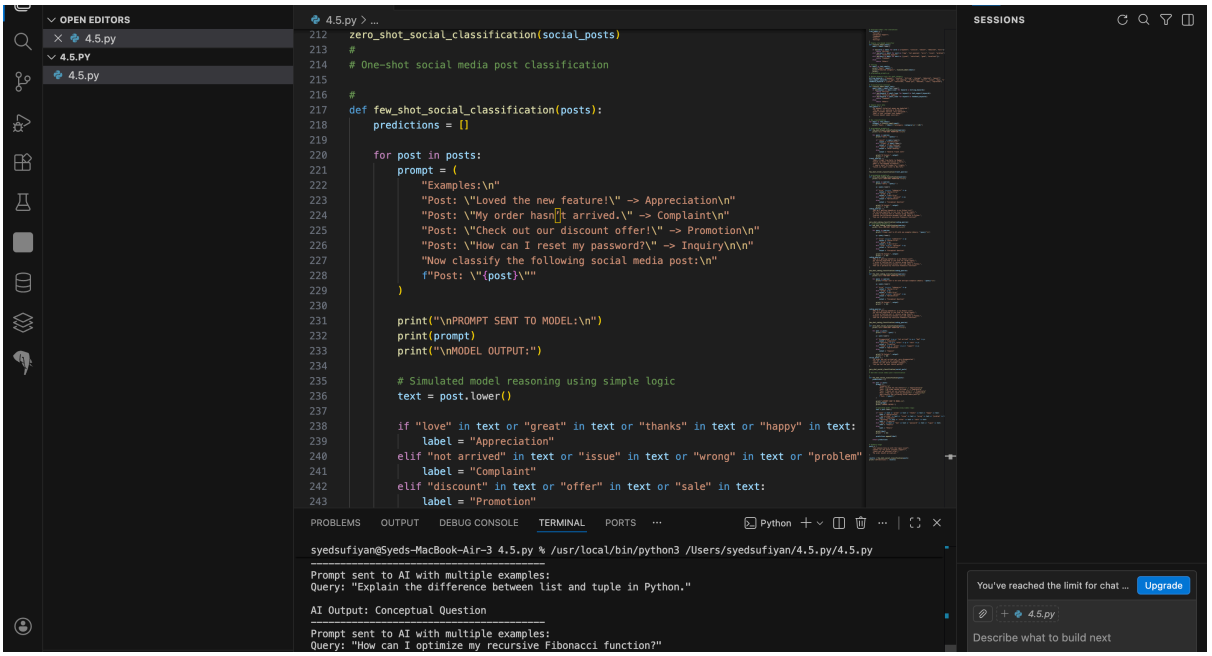
EXAMPLE:

Post: "My order is late."

Category: Complaint

Now classify the following social media post:

Post: "<social_post>"



OBSERVATION:

Provides one example to guide AI reasoning.

Improves accuracy and handles some informal expressions better.

Depends on how representative the example is for informal language.

FEW-SHOT

PROMPT:

Examples:

Post: "Loved the new feature!" → Appreciation

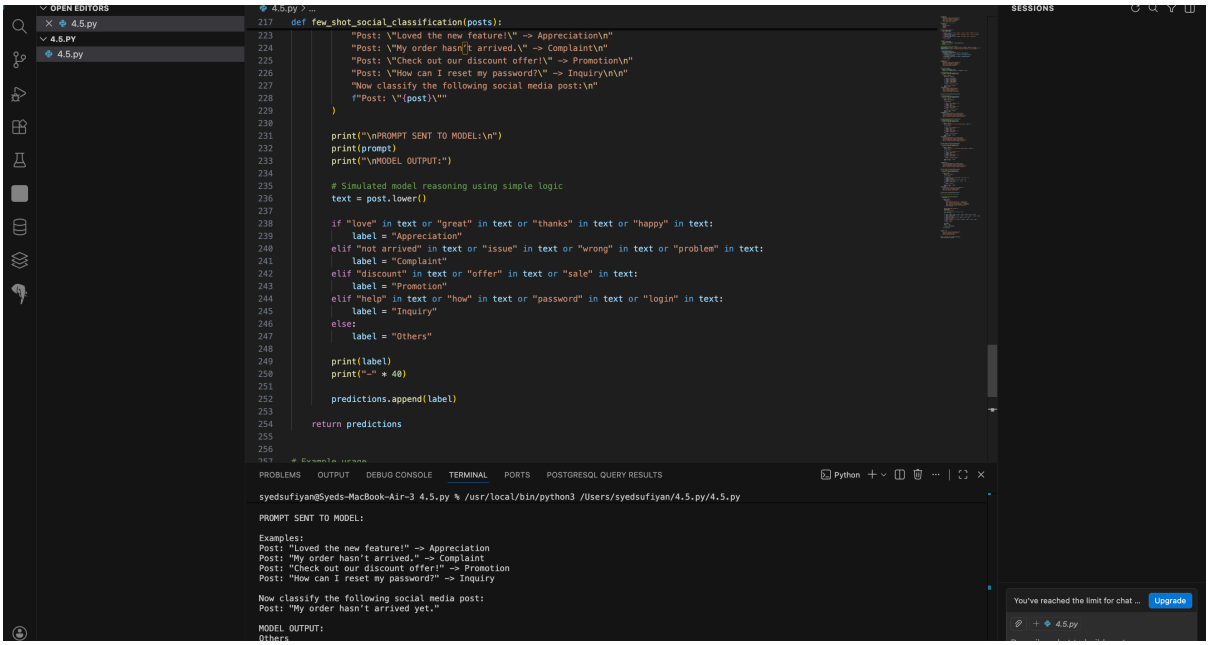
Post: "My order hasn’t arrived." → Complaint

Post: "Check out our discount offer!" → Promotion

Post: "How can I reset my password?" → Inquiry

Now classify the following social media post:

Post: "<social_post>"



OBSERVATION:

Provides multiple examples showing patterns to AI.

Highest accuracy; better handles informal, slang, or mixedlanguage posts. Slightly longer prompts but most reliable for social media classification.