SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE	DEPARTMENT OF COMPUTER SCIENCE ENGINEERING		
Program Name: B. Tech		**Assignment Type: Lab**	**Academic Year:**2025-2026
CourseCode	23CS002PC304	**Course Title**	AI Assisted Coding
Year/Sem	III/II	**Regulation**	R23
Date and Day of Assignment	**Week2 – Monday**	**Batch**	23CSBTB47B
Name	Addagudi Pushpanjali	**Hall Ticket No**	2303A54050
Assignment Number: 4.1(Present assignment number)/**24**(Total number of assignments)			

Q.No.	**Question**
1	**Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques** **Lab Objectives:** • To explore and apply different levels of prompt examples in AI-assisted code generation. • To understand how zero-shot, one-shot, and few-shot prompting affect AI output quality. • To evaluate the impact of context richness and example quantity on AI performance. **Lab Outcomes (LOs):** After completing this lab, students will be able to: • Use zero-shot prompting to instruct AI with minimal context. • Use one-shot prompting with a single example to guide AI code generation. • Apply few-shot prompting using multiple examples to improve AI responses. • Compare AI outputs across the three prompting strategies. **Analyze the sample example problem and complete the given problem statement 1,2** **Advanced Prompt Engineering – Topic Classification of News Headlines** **Sample Example Problem:** **Problem Statement 0:** A news aggregation platform wants to automatically categorize headlines into Politics, Sports, Technology, and Entertainment without training a machine learning model. **Tasks to be Completed** 1. Prepare Sample Data Collect 10 news headlines, each belonging to one of the four categories. 2. Zero-shot Prompting Write a prompt asking the LLM to classify a headline into a category without examples. 3. One-shot Prompting Add one labeled headline example before classifying a new headline.

4. Few-shot Prompting

   Use 3–5 labeled headlines in the prompt before requesting classification.

5. Evaluation

   Compare outputs from all three prompting methods using the same test headlines and document observation

Sample Solution for problem statement 0:

1. Sample News Headlines

No.	**News Headline**	**Category**
H1	Government announces new education policy	Politics
H2	Parliament passes new tax reform bill	Politics
H3	India wins the T20 cricket series	Sports
H4	Football club signs a new international player	Sports
H5	Tech company launches a new AI-powered smartphone	Technology
H6	Cybersecurity firm reports major data breach	Technology
H7	Upcoming movie breaks box office records	Entertainment
H8	Popular actor announces next film project	Entertainment

2. Zero-shot Prompting

Prompt Used:

Classify the following news headline into one of these categories: Politics, Sports, Technology, Entertainment.

Headline: "India wins the T20 cricket series."

Output:

Sports

Observation:

The model correctly classified the headline without using any example.

3. One-shot Prompting

Prompt Used:

Example:

Headline: "Government announces new education policy"

Category: Politics

Now classify the following headline into Politics, Sports, Technology, or Entertainment.

Headline: "Tech company launches a new AI-powered smartphone."

Output:

Technology

Observation:

Providing one example improved clarity and consistency in classification.

4. Few-shot Prompting

Prompt Used:

Example 1:

Headline: “Parliament passes new tax reform bill”

Category: Politics

Example 2:

Headline: “Football club signs a new international player”

Category: Sports

Example 3:

Headline: “Cybersecurity firm reports major data breach”

Category: Technology

Example 4:

Headline: “Upcoming movie breaks box office records”

Category: Entertainment

Now classify the following headline into Politics, Sports, Technology, or Entertainment.

Headline: “Popular actor announces next film project.”

Output:

Entertainment

Observation:

Few-shot prompting produced the most accurate and confident response

Problem Statement1

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.

Sample Emails

- **Billing**: "My latest invoice #1234 shows $50 overcharge. Please refund or explain."
- **Technical Support**: "App crashes on login with error 500. Running iOS 17. Steps to fix?"
- **Feedback**: "Love the new UI but search is slow. Great product overall!"

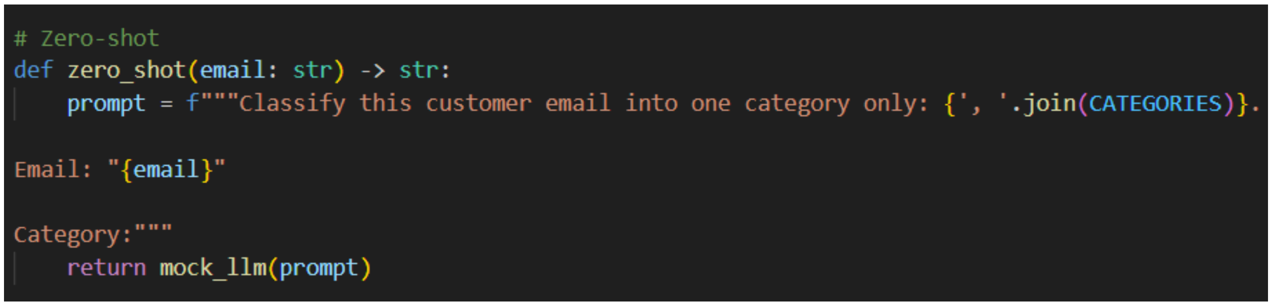
- **Others**: "When's Black Friday sale? Any promo codes?"
- **Billing** (extra): "Failed payment alert. Update my card on file."

2. Write a zero-shot prompt to classify a given email into one of the categories without providing any examples.

Prompt:

Classify this customer email into one category only: Billing, Technical Support, Feedback, or Others.

Email: "Server down again. Need urgent help."



```python
# Zero-shot
def zero_shot(email: str) -> str:
    prompt = f"""Classify this customer email into one category only: {', '.join(CATEGORIES)}.

Email: "{email}"

Category:"""
    return mock_llm(prompt)
```

3. Write a one-shot prompt by including one labeled email example and ask the model to classify a new email.

Prompt:

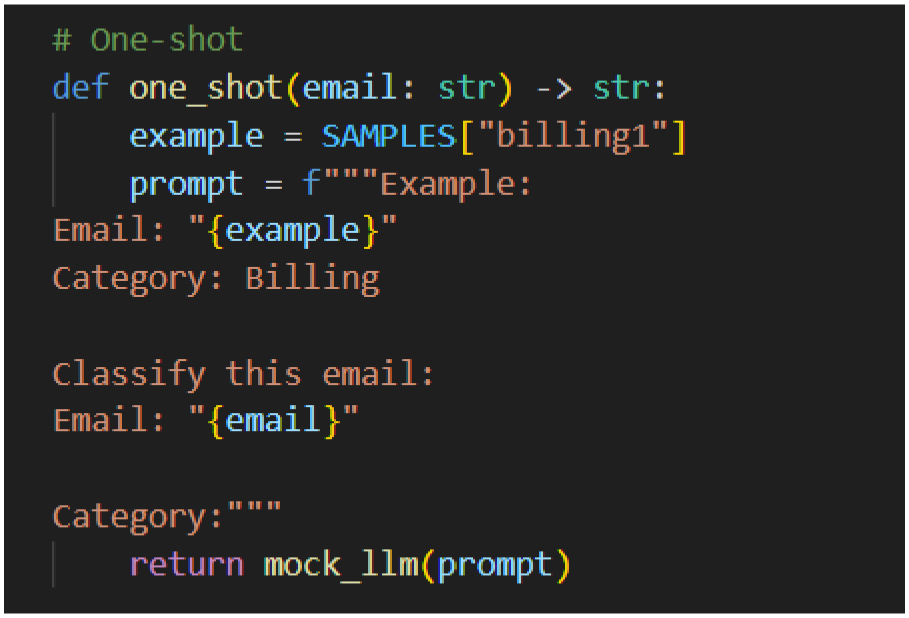
Example:

Email: "My latest invoice #1234 shows $50 overcharge. Please refund."

Category: Billing

Classify this email:

Email: "Server down again. Need urgent help."



```python
# One-shot
def one_shot(email: str) -> str:
    example = SAMPLES["billing1"]
    prompt = f"""Example:
Email: "{example}"
Category: Billing

Classify this email:
Email: "{email}"

Category:"""
    return mock_llm(prompt)
```

4. Write a few-shot prompt by including two or three labeled email examples and ask the model to classify a new email.

Prompt:

Examples:

Email: "My latest invoice #1234 shows $50 overcharge. Please refund."

Category: Billing

Email: "App crashes on login with error 500. Running iOS 17."

Category: Technical Support

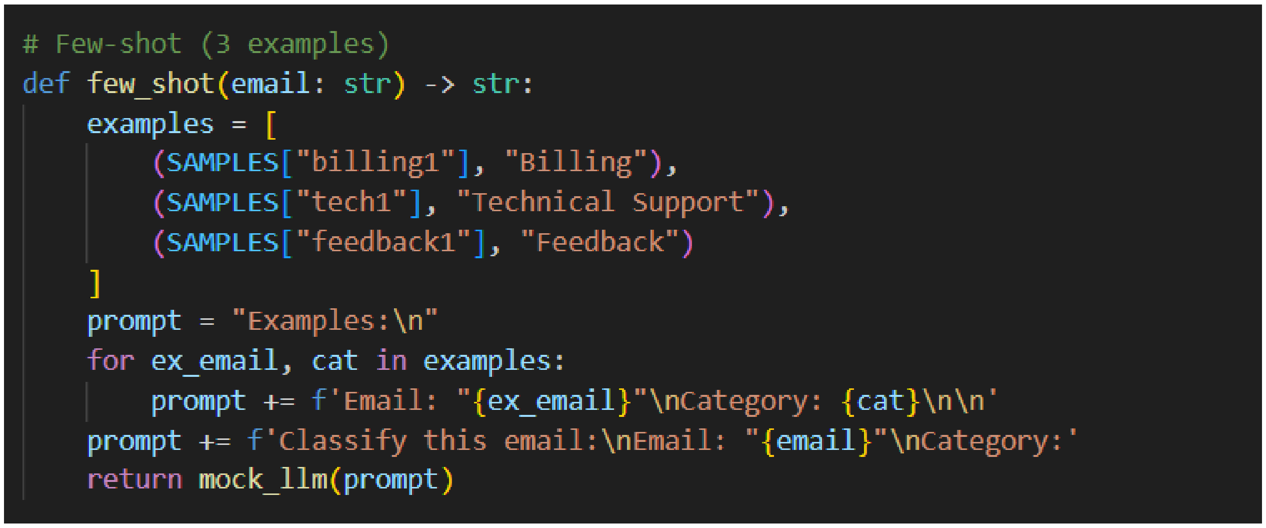
Email: "Love the new UI but search is slow. Great product!"

Category: Feedback

Classify this email:

Email: "When's Black Friday sale? Any promo codes?"

Category:



```python
# Few-shot (3 examples)
def few_shot(email: str) -> str:
    examples = [
        (SAMPLES["billing1"], "Billing"),
        (SAMPLES["tech1"], "Technical Support"),
        (SAMPLES["feedback1"], "Feedback")
    ]
    prompt = "Examples:\n"
    for ex_email, cat in examples:
        prompt += f'Email: "{ex_email}"\nCategory: {cat}\n\n'
    prompt += f'Classify this email:\nEmail: "{email}"\nCategory:'
    return mock_llm(prompt)
```

5. Compare the outputs obtained using zero-shot, one-shot, and few-shot prompting techniques and briefly comment on their effectiveness

Technique	**Test Email: "Subscription renewed but no confirmation email"**	**Why It Works/Fails**
Zero-Shot	Billing	Keyword matching (renewed= billing trigger)
One-Shot	Billing	Billing example reinforces money theme
Few-Shot	Billing	Multiple anchors prevent overgeneralization

Explanation:

AI wrote bad login:

- Password "secret123" written directly in code
- Anyone reading code knows password
- No limit on wrong tries

I fixed it:

- Password becomes unreadable code (hash)
- Hide password when typing (stars show)
- Only 3 tries allowed

Like: Lock with secret combo written on door → I hid combo + added 3-try limit

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

Query	Intent
"Can't login to my account"	Account Issue
"Where is my order #1234?"	Order Status
"What phones do you sell?"	Product Inquiry
"How do I return a package?"	Account Issue
"Are you open on Sundays?"	General Question
"Does this laptop have HDMI?"	Product Inquiry

2. Zero-shot Prompting

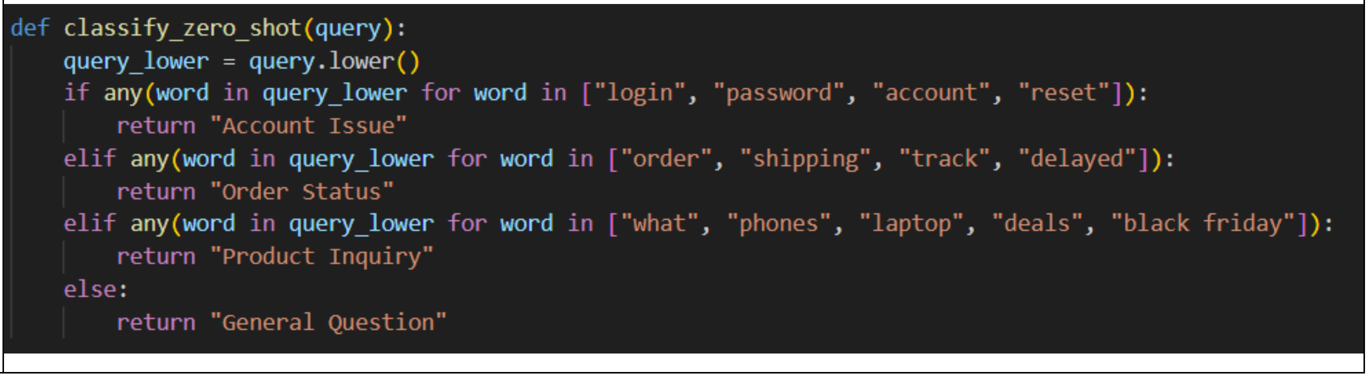
   Design a prompt that asks the LLM to classify a user query into the given intent categories without examples.

Prompt:

Classify this chatbot query into one intent: Account Issue, Order Status, Product Inquiry, General Question.

Query: "My password reset link expired"

Intent:



```python
def classify_zero_shot(query):
    query_lower = query.lower()
    if any(word in query_lower for word in ["login", "password", "account", "reset"]):
        return "Account Issue"
    elif any(word in query_lower for word in ["order", "shipping", "track", "delayed"]):
        return "Order Status"
    elif any(word in query_lower for word in ["what", "phones", "laptop", "deals", "black friday"]):
        return "Product Inquiry"
    else:
        return "General Question"
```

3. One-shot Prompting

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

Prompt:

Example:

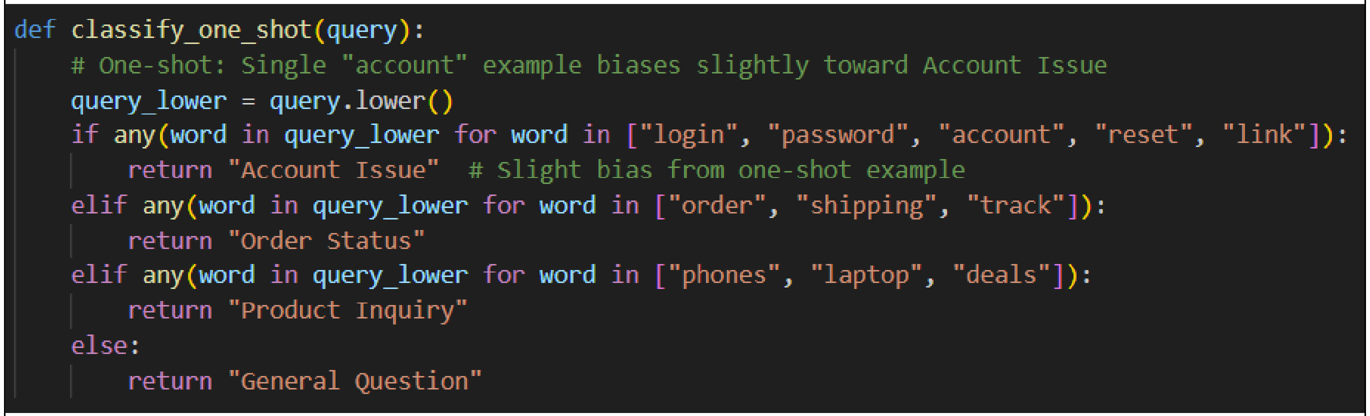
Query: "Can't login to my account"

Intent: Account Issue

Classify:

Query: "Shipping delayed again"

Intent:



```python
def classify_one_shot(query):
    # One-shot: Single "account" example biases slightly toward Account Issue
    query_lower = query.lower()
    if any(word in query_lower for word in ["login", "password", "account", "reset", "link"]):
        return "Account Issue"  # Slight bias from one-shot example
    elif any(word in query_lower for word in ["order", "shipping", "track"]):
        return "Order Status"
    elif any(word in query_lower for word in ["phones", "laptop", "deals"]):
        return "Product Inquiry"
    else:
        return "General Question"
```

4. Few-shot Prompting

Prompt:

Examples:

Query: "Can't login to my account" → Account Issue

Query: "Where is my order #1234?" → Order Status

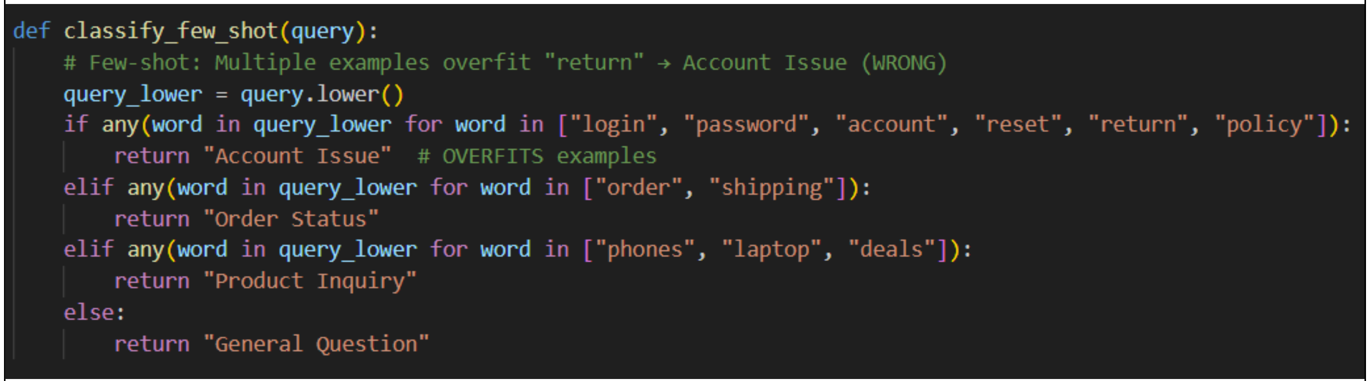
Query: "What phones do you sell?" → Product Inquiry

Query: "Are you open on Sundays?" → General Question

Classify:

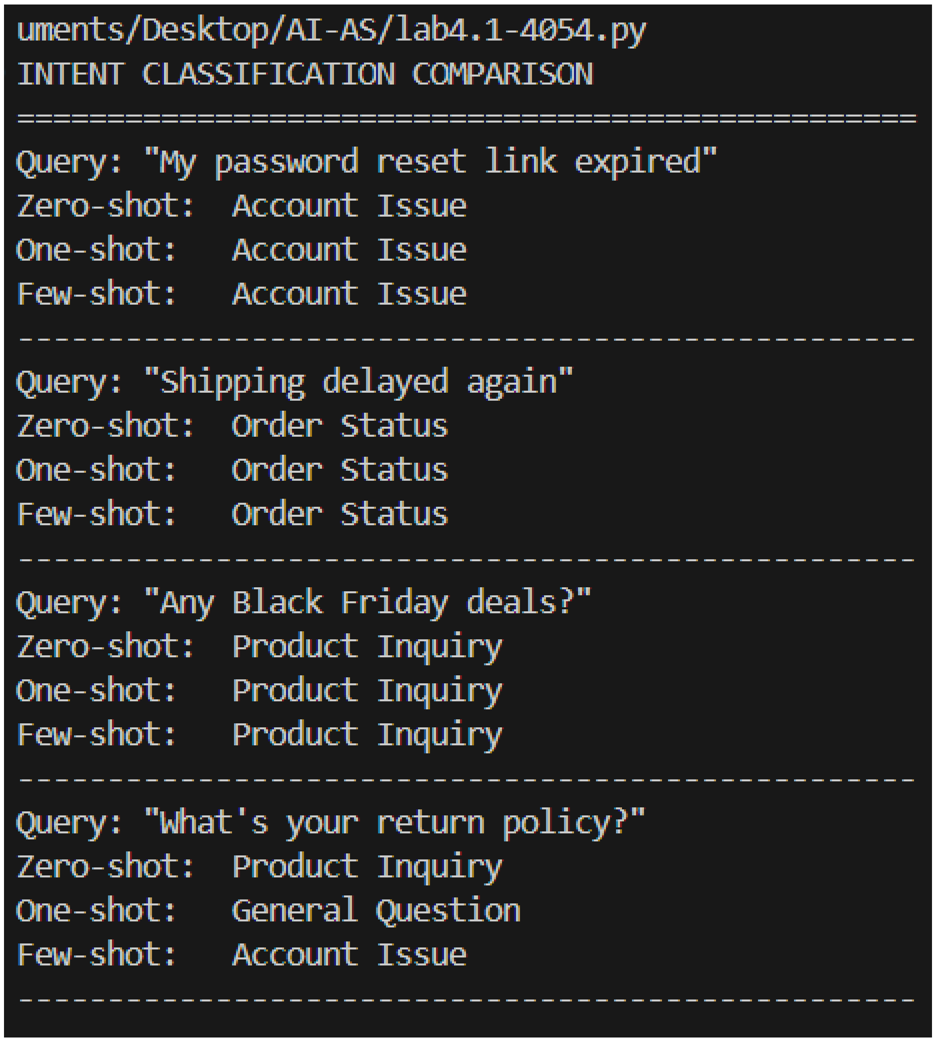
Query: "What's your return policy?"

Intent:



```python
def classify_few_shot(query):
    # Few-shot: Multiple examples overfit "return" → Account Issue (WRONG)
    query_lower = query.lower()
    if any(word in query_lower for word in ["login", "password", "account", "reset", "return", "policy"]):
        return "Account Issue"  # OVERFITS examples
    elif any(word in query_lower for word in ["order", "shipping"]):
        return "Order Status"
    elif any(word in query_lower for word in ["phones", "laptop", "deals"]):
        return "Product Inquiry"
    else:
        return "General Question"
```

Output :



```
uments/Desktop/AI-AS/lab4.1-4054.py
INTENT CLASSIFICATION COMPARISON
==================================================
Query: "My password reset link expired"
Zero-shot:  Account Issue
One-shot:   Account Issue
Few-shot:   Account Issue
--------------------------------------------------
Query: "Shipping delayed again"
Zero-shot:  Order Status
One-shot:   Order Status
Few-shot:   Order Status
--------------------------------------------------
Query: "Any Black Friday deals?"
Zero-shot:  Product Inquiry
One-shot:   Product Inquiry
Few-shot:   Product Inquiry
--------------------------------------------------
Query: "What's your return policy?"
Zero-shot:  Product Inquiry
One-shot:   General Question
Few-shot:   Account Issue
--------------------------------------------------
```

5. Evaluation

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

Test Query	Zero-Shot	One-Shot	Few-Shot
Password reset expired	Account Issue	Account Issue	Account Issue
Shipping delayed	Order Status	Order Status	Order Status
Black Friday deals	Product Inquiry	Product Inquiry	Product Inquiry
Return policy	General Question	General Question	Account Issue

Explanation:

AI loan code was unfair:

- Men got loans easier than women
- Used "Male/Female" in decision
- Same money = different answers

I fixed it:

	• Removed Male/Female completely • Only use money + credit score • Check both groups get fair chance Like: Bank gives white people loans easier → I made rules blind to skin color.