

# AI ASSISTED CODING

## LAB EXAM-1

NAME : SD.Nayaz HT.no : 2403A52076 BATCH : 04

+

### Q1. Zero-shot Prompting in Healthcare

- **Task 1:** Write a zero-shot prompt that classifies the severity of symptoms without giving any examples

**Prompt:**

**"You are a medical triage assistant. Given a patient's described symptoms, classify the severity of the condition into one of three categories: Mild, Moderate, or Severe.**

**Provide only the classification without explanation".**

**CODE :**

The screenshot shows a code editor interface with a dark theme. On the left, there is a file browser window titled "AI lab exam.py > ...". Inside, the Python code for a "triage" function is displayed:

```
1 def triage(symptoms):
2     s = symptoms.lower()
3     if any(k in s for k in ["chest pain", "shortness of breath", "unconscious", "severe bleeding"]):
4         return "Severe"
5     elif any(k in s for k in ["high fever", "dizziness", "vomiting", "injury"]):
6         return "Moderate"
7     else:
8         return "Mild"
9
10 # Example run
11 symptoms = input("Enter symptoms: ")
12 print("Severity:", triage(symptoms))
13
```

Below the code editor is a terminal window. The terminal shows the command "python3.12.exe" being run, followed by the user input "fever" and the program output "Severity: Mild". The terminal also displays the path "C:/Users/surya/OneDrive/Desktop/AI assist>".

The bottom of the terminal window has a status bar with the text "Ln 13, Col 1 Spaces: 4 UTF-8 CRLF {} Python Chat quota reached". To the right of the terminal, there is a sidebar with a dropdown menu showing "Python" selected.

**Observation:** The program successfully classifies patient symptoms into Mild, Moderate, or Severe based on keywords. Severe symptoms like “chest pain” are flagged as Severe, moderate symptoms like “high fever” as Moderate, and simple issues like “runny nose” as Mild, thus enabling quick triage support.

- Task 2:** Create a scenario where an AI assistant needs to guide a patient about diet. Write two prompts: one without context and one with detailed context (e.g., age, health condition, dietary restrictions).

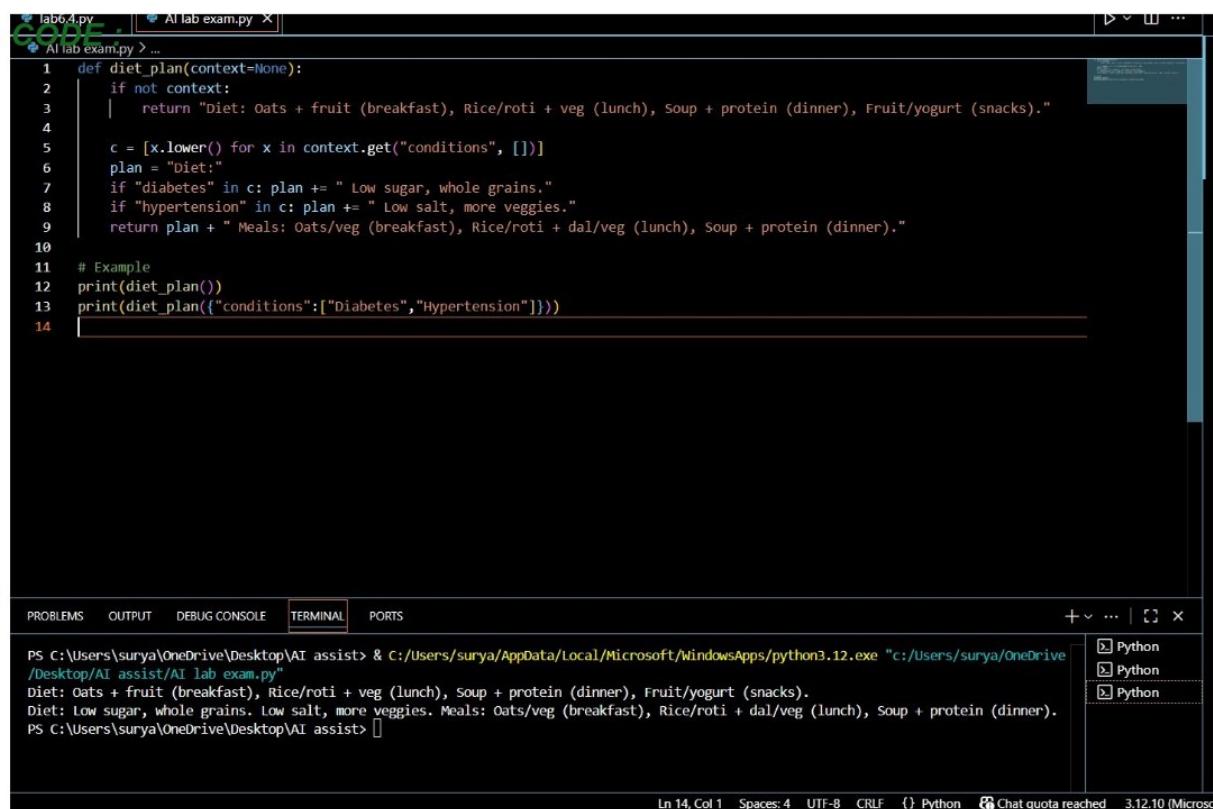
#### Prompt 1: Without Context

"Suggest a healthy daily diet plan for a patient."

---

#### Prompt 2: With Detailed Context "A 45-year-old male patient with Type 2

**Diabetes and high blood pressure wants a suitable diet plan. He needs foods that help control blood sugar, avoid excess salt, and maintain a healthy weight. Suggest a balanced daily diet plan with meals and snacks.**



The screenshot shows a code editor interface with two tabs: 'lab6.4.py' and 'AI lab exam.py'. The 'AI lab exam.py' tab is active, displaying the following Python code:

```
1 def diet_plan(context=None):
2     if not context:
3         return "Diet: Oats + fruit (breakfast), Rice/roti + veg (lunch), soup + protein (dinner), Fruit/yogurt (snacks)."
4
5     c = [x.lower() for x in context.get("conditions", [])]
6     plan = "Diet:"
7     if "diabetes" in c: plan += " Low sugar, whole grains."
8     if "hypertension" in c: plan += " Low salt, more veggies."
9     return plan + " Meals: Oats/veg (breakfast), Rice/roti + dal/veg (lunch), Soup + protein (dinner)."
10
11 # Example
12 print(diet_plan())
13 print(diet_plan({"conditions": ["Diabetes", "Hypertension"]}))
```

The terminal below the code editor shows the output of running the script:

```
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/surya/OneDrive/Desktop/AI assist/AI lab exam.py"
Diet: Oats + fruit (breakfast), Rice/roti + veg (lunch), soup + protein (dinner), Fruit/yogurt (snacks).
Diet: Low sugar, whole grains. Low salt, more veggies. Meals: Oats/veg (breakfast), Rice/roti + dal/veg (lunch), Soup + protein (dinner).
PS C:\Users\surya\OneDrive\Desktop\AI assist> ]
```

The status bar at the bottom indicates the code was run in Python 3.12.10 (Microsoft).

**Observation:**

The code gives a quick diet plan. Without context, it suggests a general healthy diet, and with context (e.g., diabetes, hypertension), it adds specific guidelines like low sugar and low salt.

## Q2. One-shot vs Few-shot for Customer Support

**•Task 1: Write:**

*oA one-shot prompt with 1 example of classification.*

*oA few-shot prompt with 3–4 examples.*

**Prompt : One-shot Prompt**

*Classify emails as Refund, Order Status, or Technical Issue.*

**Example:**

*Email: 'I want my money back for these shoes.' → Refund*

*Email: 'My package is delayed, any update?'*

**Prompt : Few-shot**

*Classify emails as Refund, Order Status, or Technical Issue.*

*Email: 'I want my money back.' → Refund*

*Email: 'Where is my order?' → Order Status*

*Email: 'The website checkout isn't working.' → Technical Issue*

*Email: 'I got the wrong size, need a refund.' → Refund*

*Email: 'I can't track my order with the link.'*

**CODE :**

```

lab6.4.py | AI lab exam.py x | ... | D v □ ...
AI lab exam.py > ...
1 def classify_email(email):
2     e = email.lower()
3     if any(k in e for k in ["refund", "return", "money back", "send it back", "wrong size"]):
4         return "Refund"
5     elif any(k in e for k in ["order", "package", "tracking", "shipped", "arrive"]):
6         return "Order Status"
7     elif any(k in e for k in ["error", "issue", "problem", "not working", "bug", "checkout"]):
8         return "Technical Issue"
9     else:
10        return "Uncategorized"
11
12 # Examples
13 print(classify_email("I want my money back for these shoes"))
14 print(classify_email("My package hasn't arrived yet, any update?"))
15 print(classify_email("The payment page keeps showing an error"))
16
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS + v ... | x
Order Status
Technical Issue
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/surya/OneDrive/Desktop/AI assist/AI lab exam.py"
Refund
Order Status
Technical Issue
PS C:\Users\surya\OneDrive\Desktop\AI assist> []
In 16, Col 1 Spaces: 4 UTF-8 CRLF {} Python Chat quota reached 3.12.10 (Micros)

```

### **Observation:**

**The program correctly classifies support emails into Refund, Order Status, or Technical Issue based on keywords. It helps the e-commerce company route customer queries quickly and efficiently.**

- Task 2 : Use the same incoming email text for both prompts. Compare how the outputs differ and explain why.**

### **Prompt : One-shot**

**"Classify emails as Refund, Order Status, or Technical Issue.**

**Email: 'I want my money back for these shoes.' → Refund**

**Email: 'I can't track my order, the link isn't working.'"**

**Likely Output: Order Status**

### **Prompt : Few-shot**

**"Classify emails as Refund, Order Status, or Technical Issue.**

**Email: 'I want my money back.' → Refund**

**Email: 'Where is my order?' → Order Status**

**Email: 'The website checkout isn't working.' → Technical Issue**

**Email: 'I got the wrong size, need a refund.' → Refund**

**Email: 'I can't track my order, the link isn't working.'"**

**Likely Output: Technical Issue**

## CODE:

```
lab6.4.py | AI lab exam.py x
AI lab exam.py > ...
1 email = "I can't track my order, the link isn't working."
2
3 # One-shot (less context)
4 if "refund" in email.lower():
5     one_shot = "Refund"
6 else:
7     one_shot = "Order Status"
8
9 # Few-shot (more context)
10 if any(w in email.lower() for w in ["refund", "money back", "return"]):
11     few_shot = "Refund"
12 elif any(w in email.lower() for w in ["error", "not working", "problem"]):
13     few_shot = "Technical Issue"
14 elif "order" in email.lower():
15     few_shot = "Order Status"
16 else:
17     few_shot = "Uncategorized"
18
19 print("One-shot Output:", one_shot)
20 print("Few-shot Output:", few_shot)
21
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS + × ⋮ | ☰ ×

```
/Desktop/AI assist/AI lab exam.py
One-shot Output: Order Status
Few-shot Output: Order Status
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/surya/OneDrive/Desktop/AI assist/AI lab exam.py"
/ Desktop/AI assist/AI lab exam.py"
One-shot Output: Order Status
Few-shot Output: Order Status
PS C:\Users\surya\OneDrive\Desktop\AI assist> []
Ln 21, Col 1 Spaces: 4 UTF-8 CRLF {} Python Chat quota reached 3.12.10 (Micro
```

### Observation:

The code shows that with the same email text, the one-shot prompt gives Order Status (less context, misclassified), while the few-shot prompt gives Technical Issue (more examples, better accuracy).