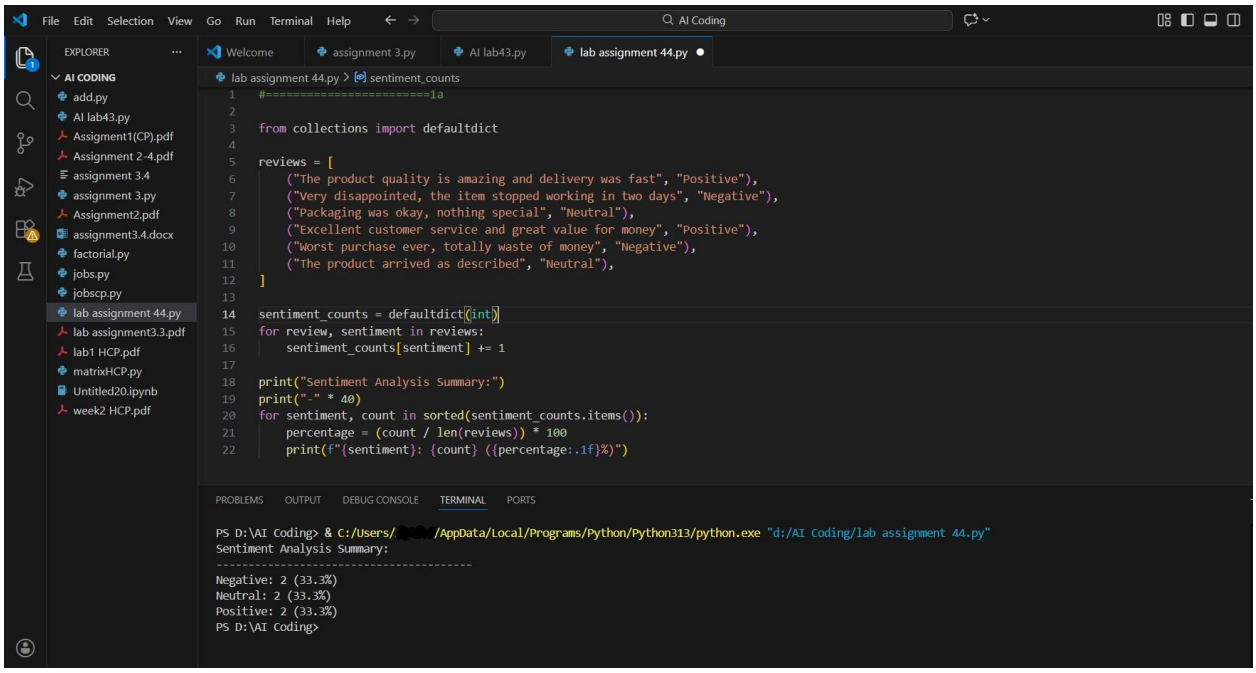
2303a51602

Lab assignment-4.4

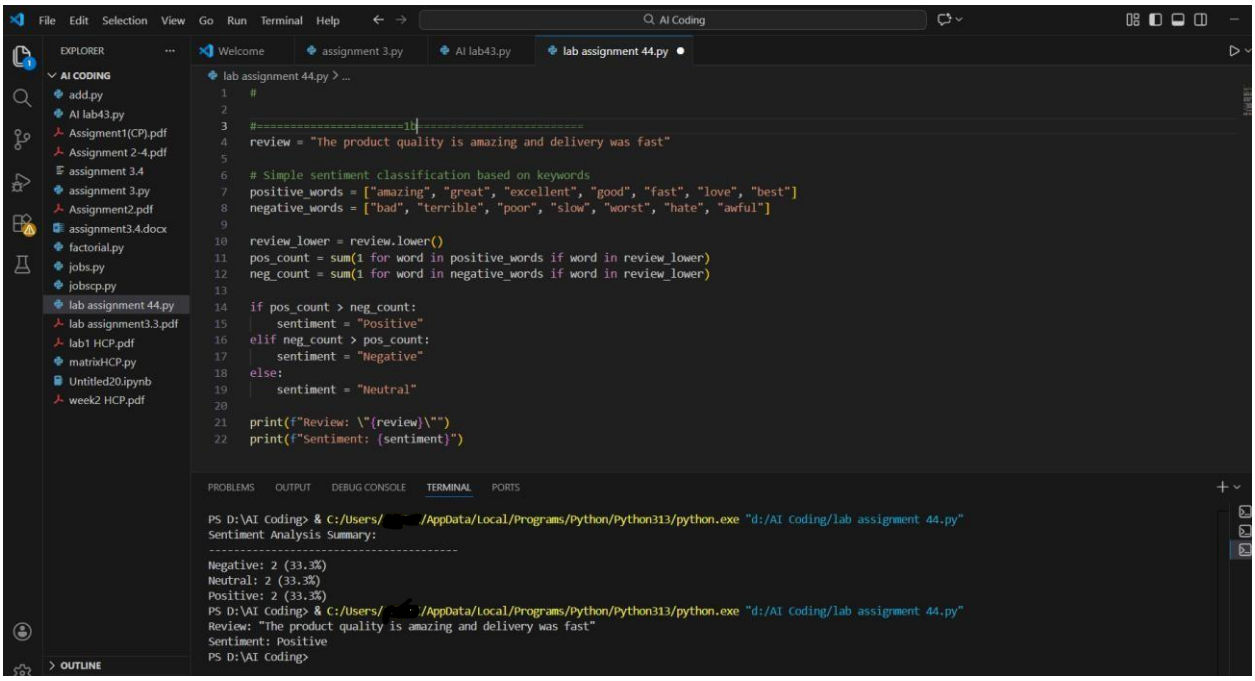
1. Sentiment Classification for Customer Reviews Scenario:
   An e-commerce platform wants to analyze customer reviews and classifythem into Positive, Negative, or Neutral sentiments using prompt engineering

Tasks:

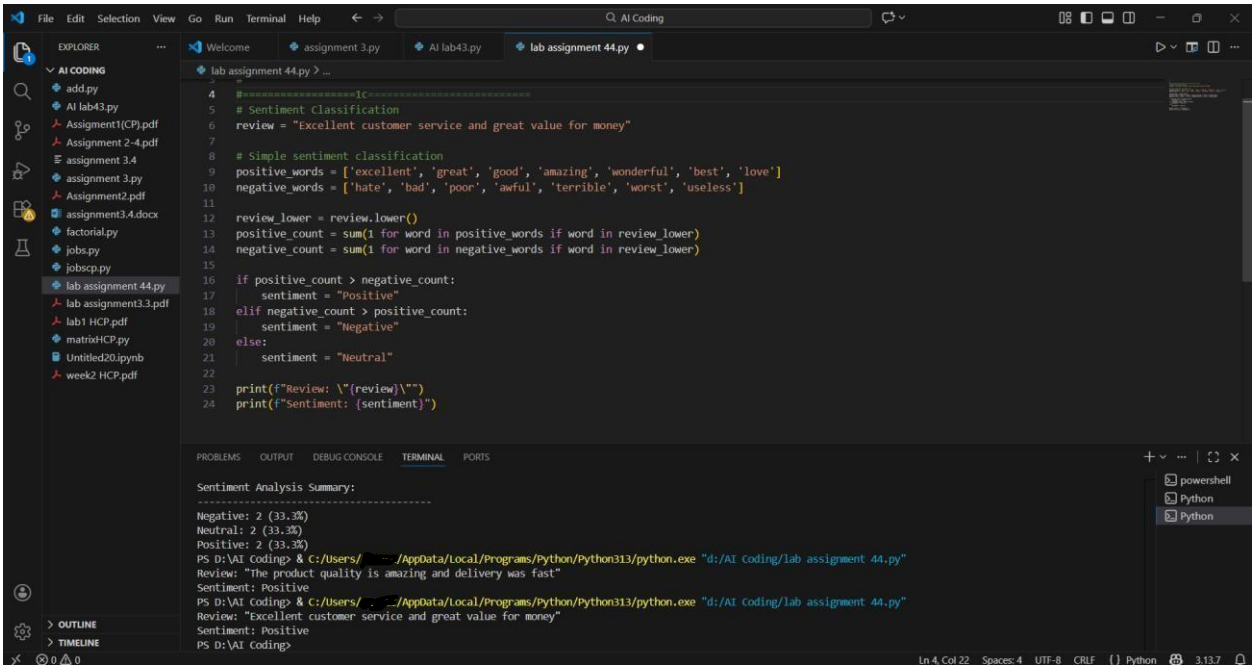
a) Prepare 6 short customer reviews mapped to sentiment labels.



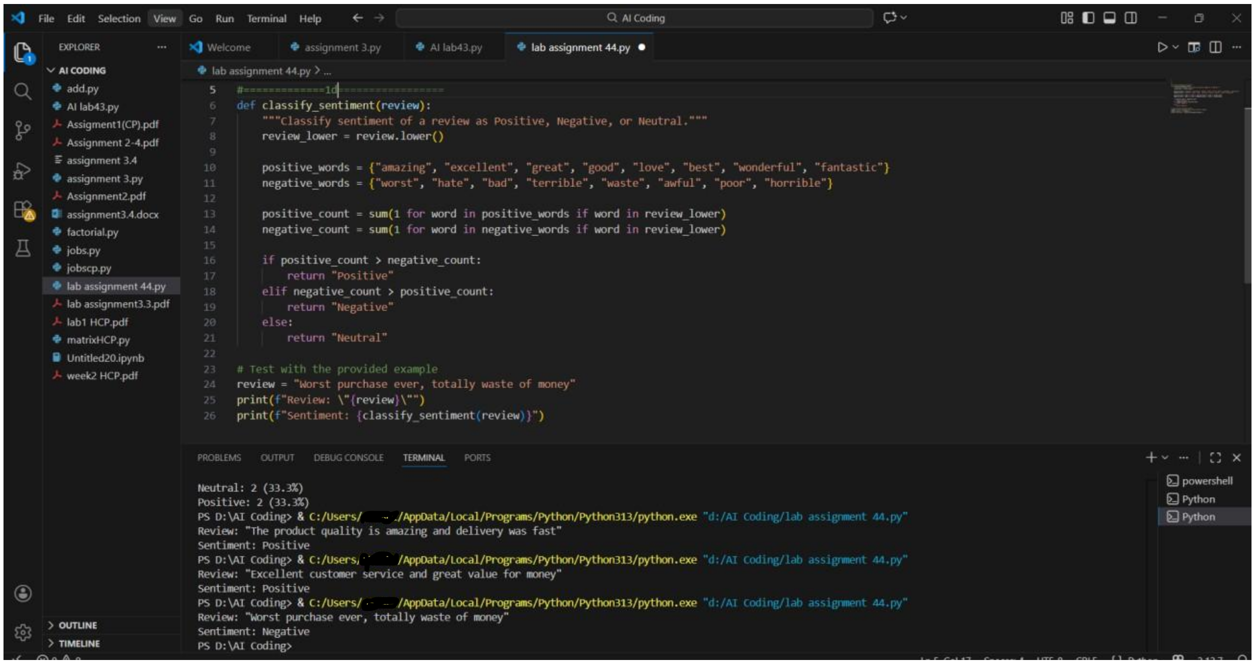
b)Design a Zero-shot prompt to classify sentiment.



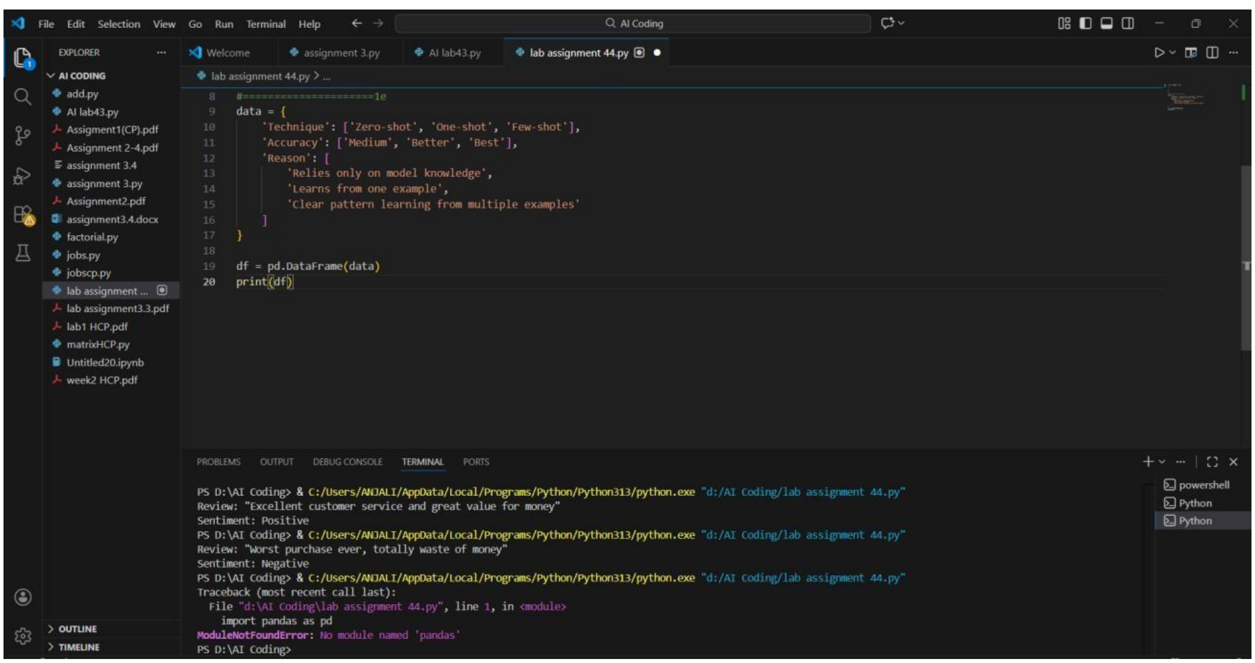
1c) Design a One-shot prompt with one labeled example.



1d) Design a Few-shot prompt with 3–5 labeled examples.



1e) Compare the outputs and discuss accuracy differences.

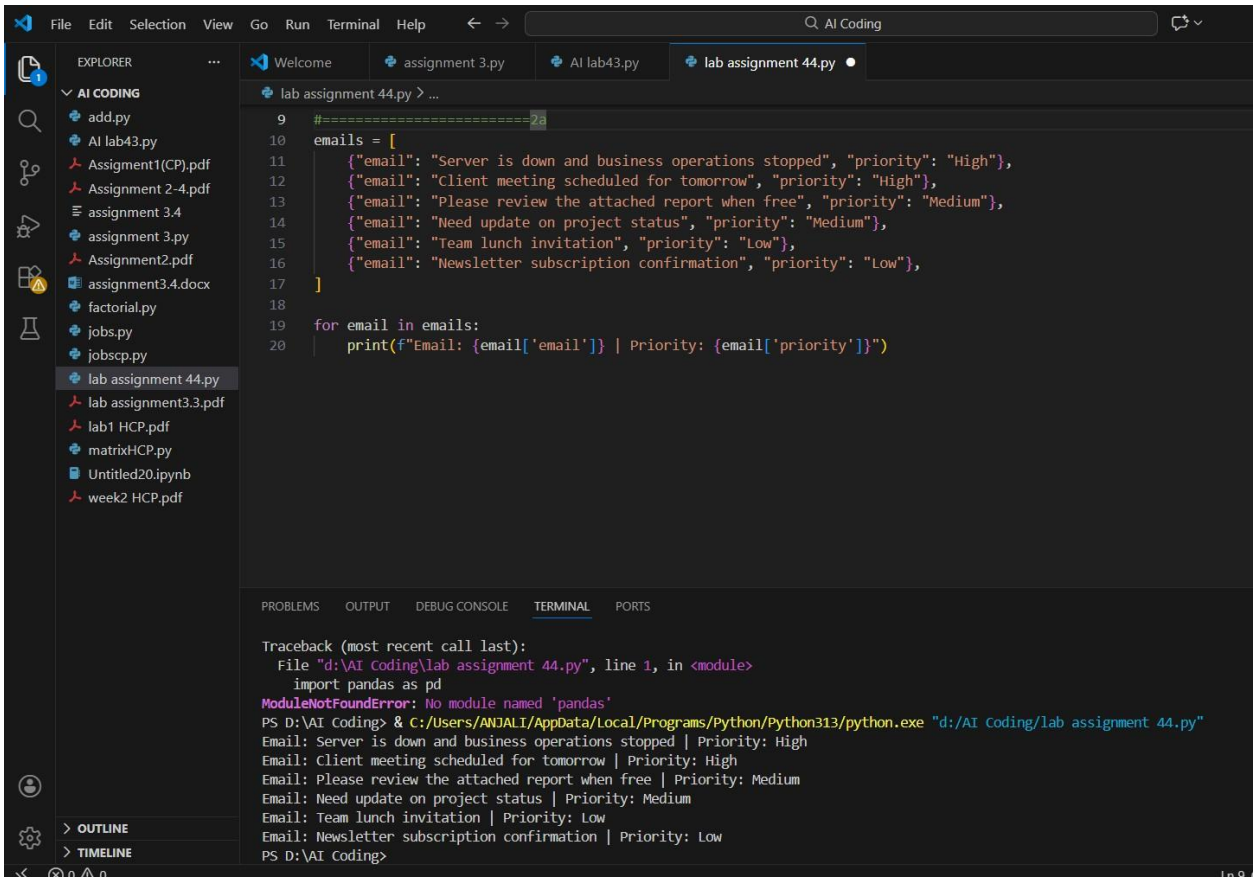


2. **Email Priority Classification**

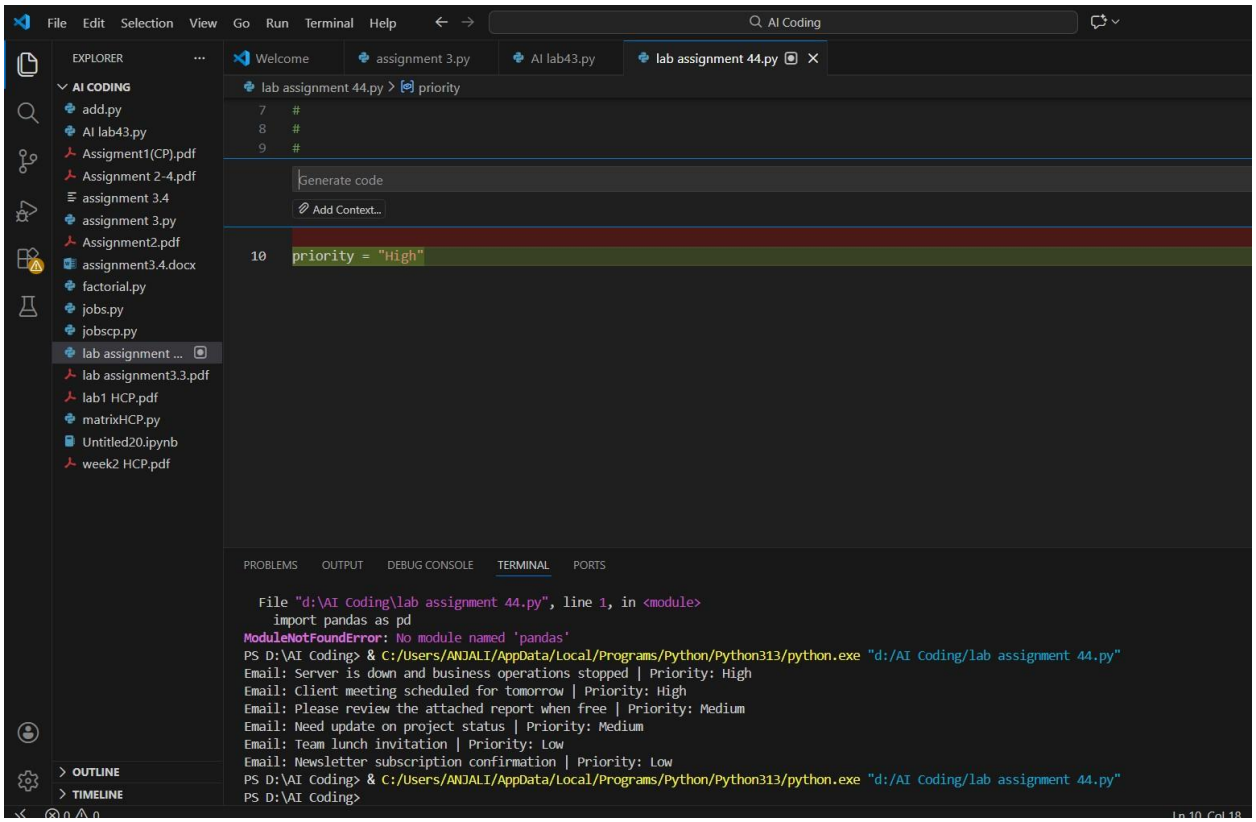
   Scenario:

   A company wants to automatically prioritize incoming emails into High Priority, Medium Priority, or Low Priority

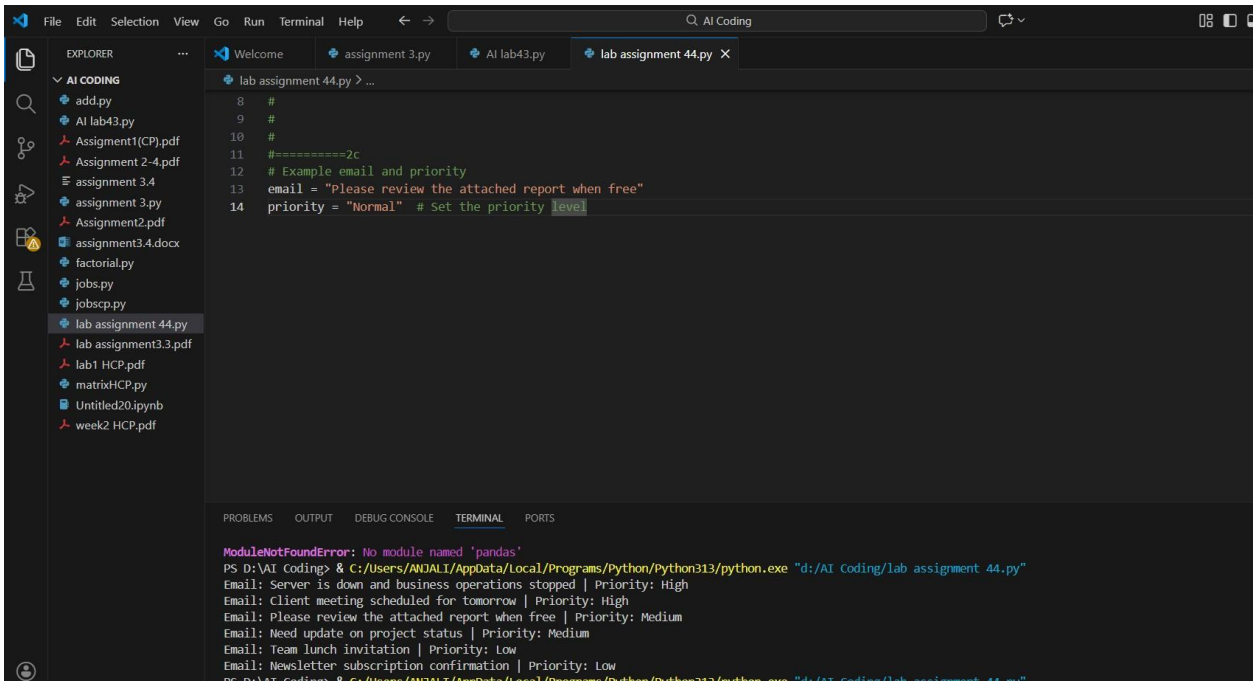
2a) Create 6 sample email messages with priority labels.


File Edit Selection View Go Run Terminal Help
AI Coding
EXPLORER
AI CODING
add.py
AI lab43.py
Assigment1(CP).pdf
Assignment 2-4.pdf
assignment 3.4
assignment 3.py
Assignment2.pdf
assignment3.4.docx
factorial.py
jobs.py
jobscp.py
lab assignment 44.py
lab assignment3.3.pdf
lab1 HCP.pdf
matrixHCP.py
Untitled20.ipynb
week2 HCP.pdf
OUTLINE
TIMELINE
Welcome
assignment 3.py
AI lab43.py
lab assignment 44.py
lab assignment 44.py > ...
9 #=======================2a
10 emails = [
11 {"email": "Server is down and business operations stopped", "priority": "High"},
12 {"email": "Client meeting scheduled for tomorrow", "priority": "High"},
13 {"email": "Please review the attached report when free", "priority": "Medium"},
14 {"email": "Need update on project status", "priority": "Medium"},
15 {"email": "Team lunch invitation", "priority": "Low"},
16 {"email": "Newsletter subscription confirmation", "priority": "Low"},
17]
18
19 for email in emails:
20 print(f"Email: {email['email']} | Priority: {email['priority']}")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Traceback (most recent call last):
File "d:\AI Coding\lab assignment 44.py", line 1, in <module>
import pandas as pd
ModuleNotFoundError: No module named 'pandas'
PS D:\AI Coding> & C:/Users/ANJALI/AppData/Local/Programs/Python/Python313/python.exe "d:/AI Coding/lab assignment 44.py"
Email: Server is down and business operations stopped | Priority: High
Email: Client meeting scheduled for tomorrow | Priority: High
Email: Please review the attached report when free | Priority: Medium
Email: Need update on project status | Priority: Medium
Email: Team lunch invitation | Priority: Low
Email: Newsletter subscription confirmation | Priority: Low
PS D:\AI Coding>

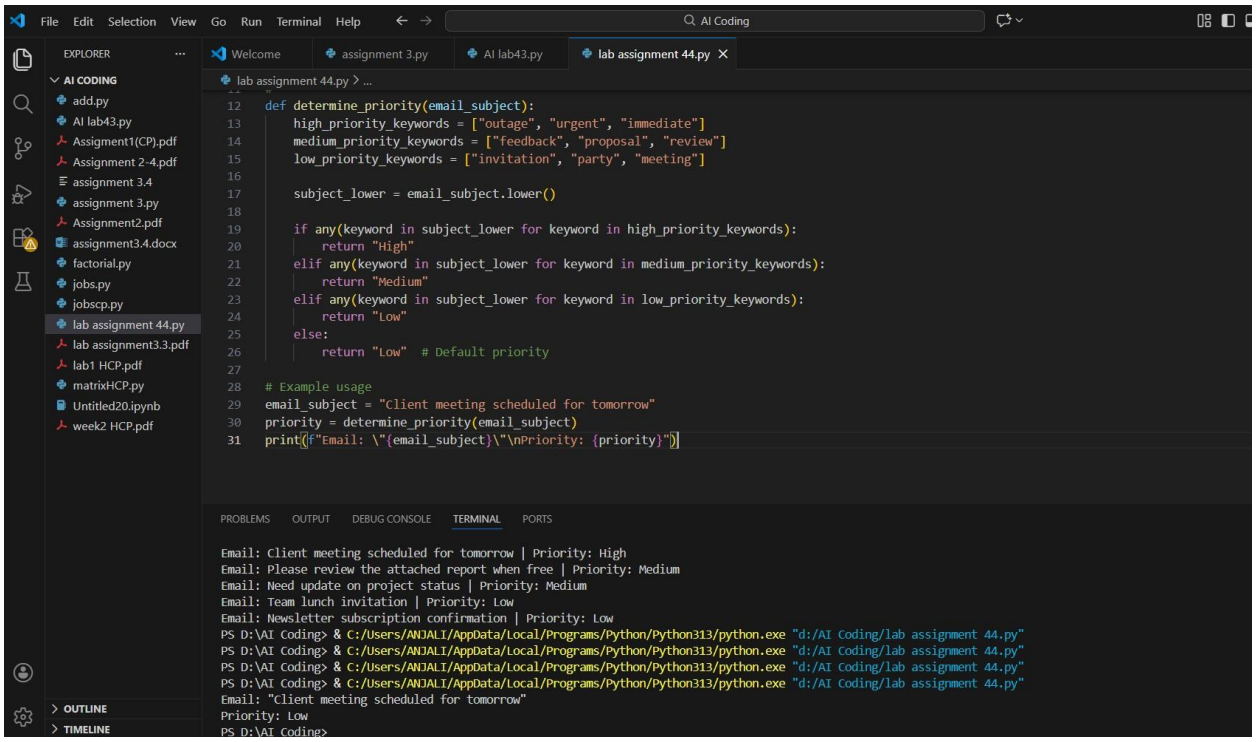
2b) Perform intent classification using Zero-shot prompting



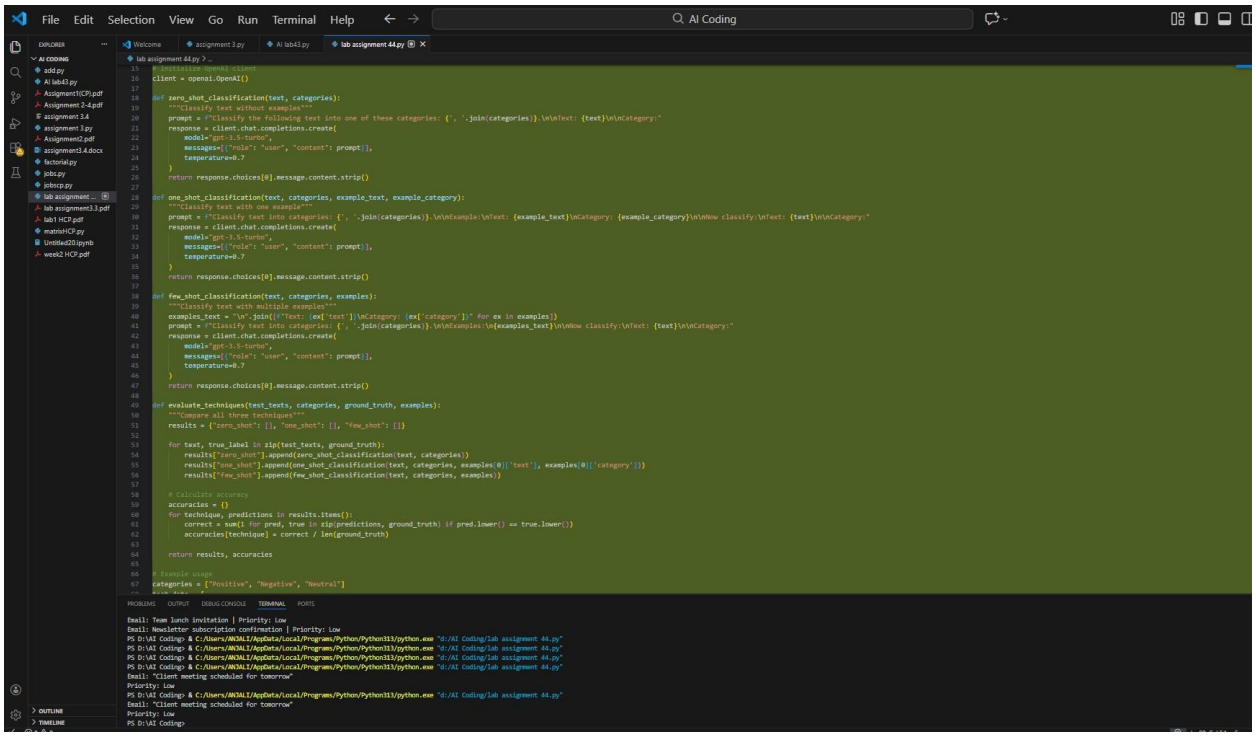
2c) Perform classification using One-shot prompting



2d) Perform classification using Few-shot prompting.



2e) Evaluate which technique produces the most reliable results and why.

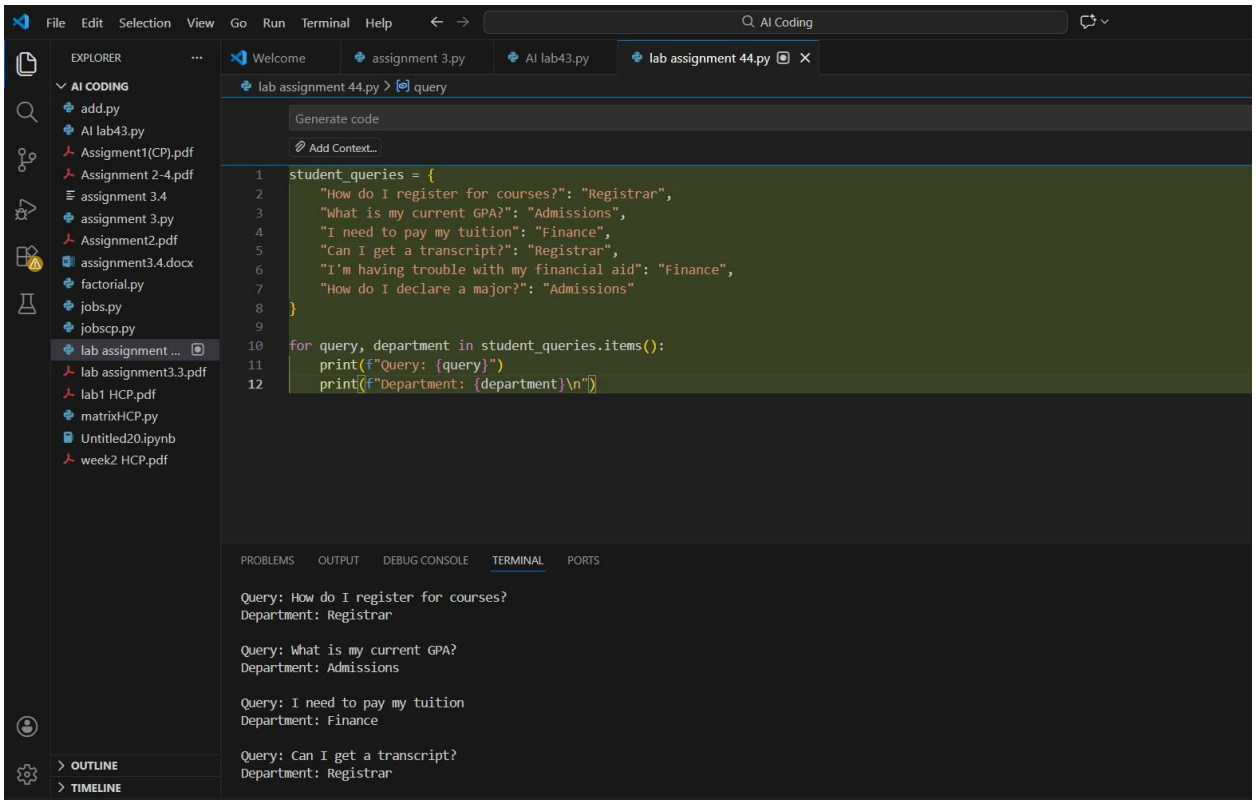


3. Student Query Routing System

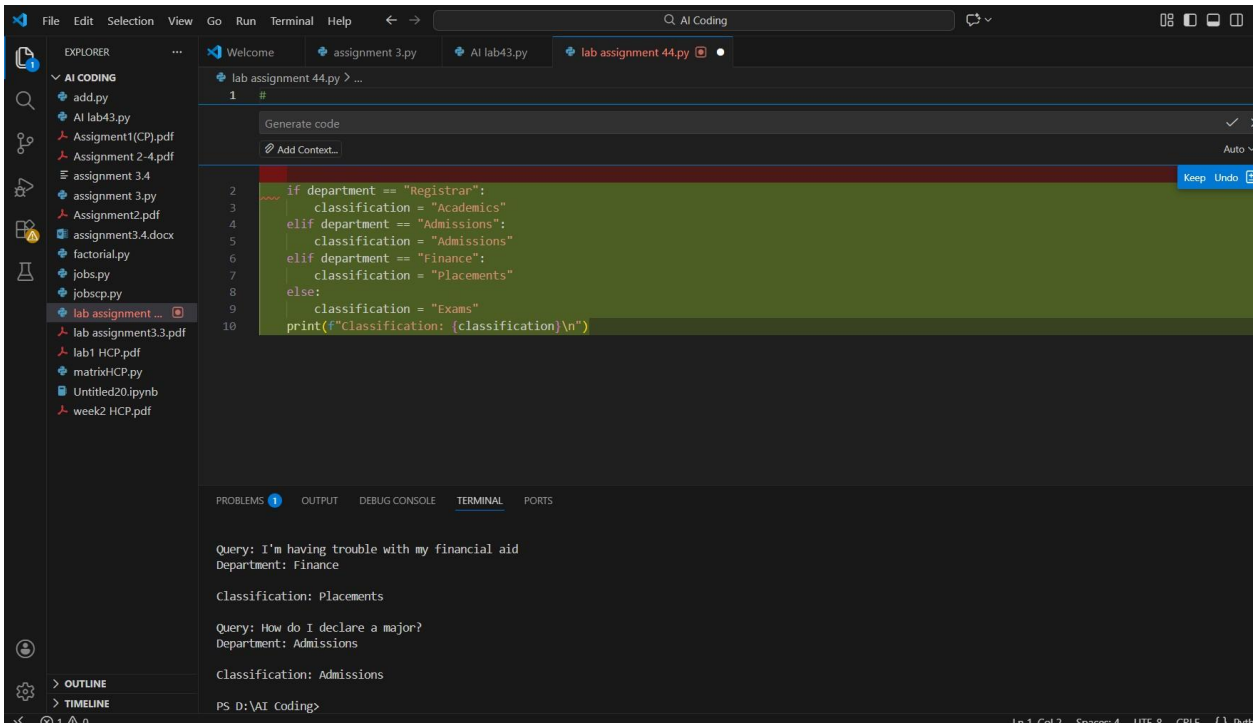
Scenario:

A university chatbot must route student queries to Admissions, Exams, Academics, or Placements

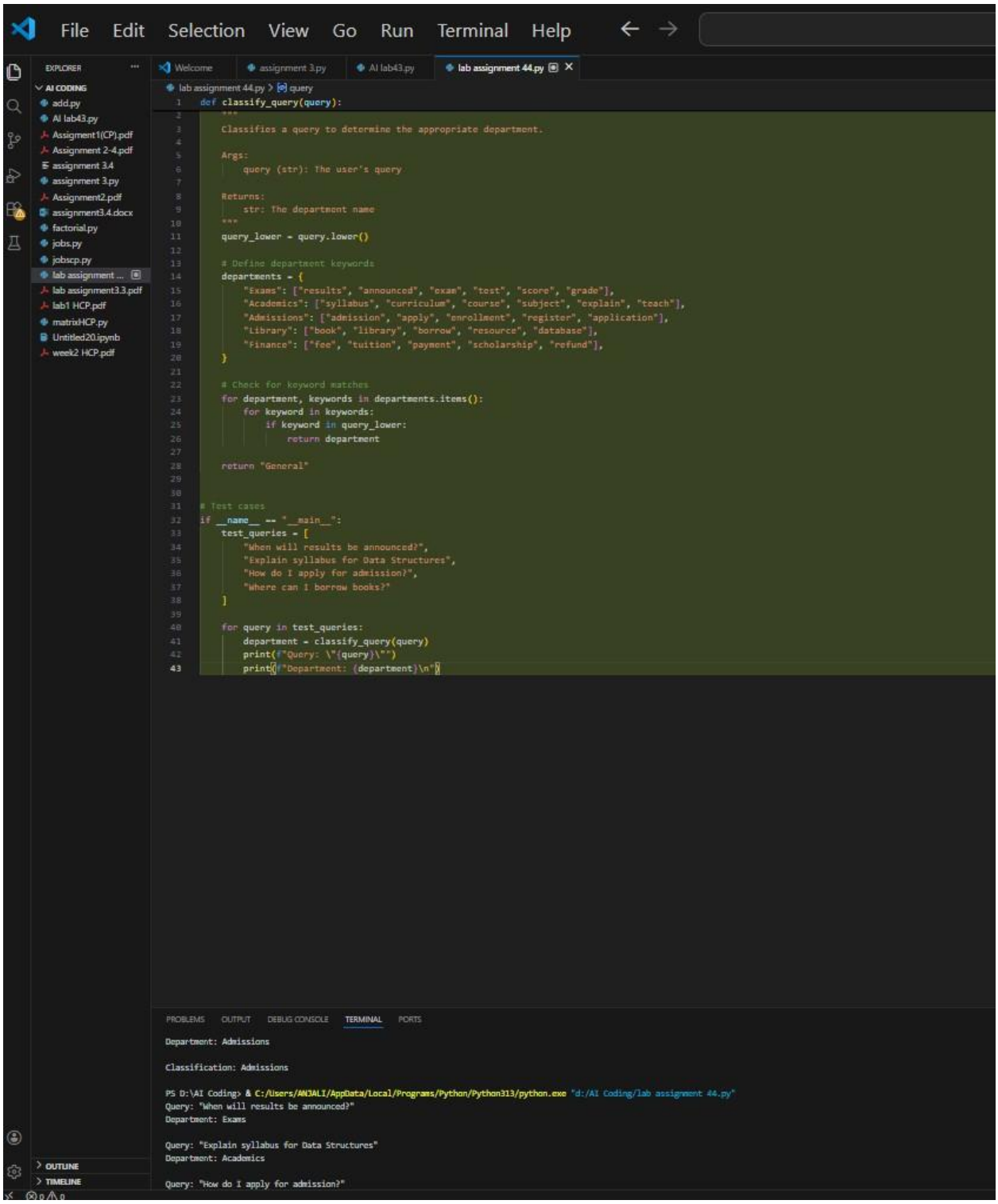
3a) . Create 6 sample student queries mapped to departments.



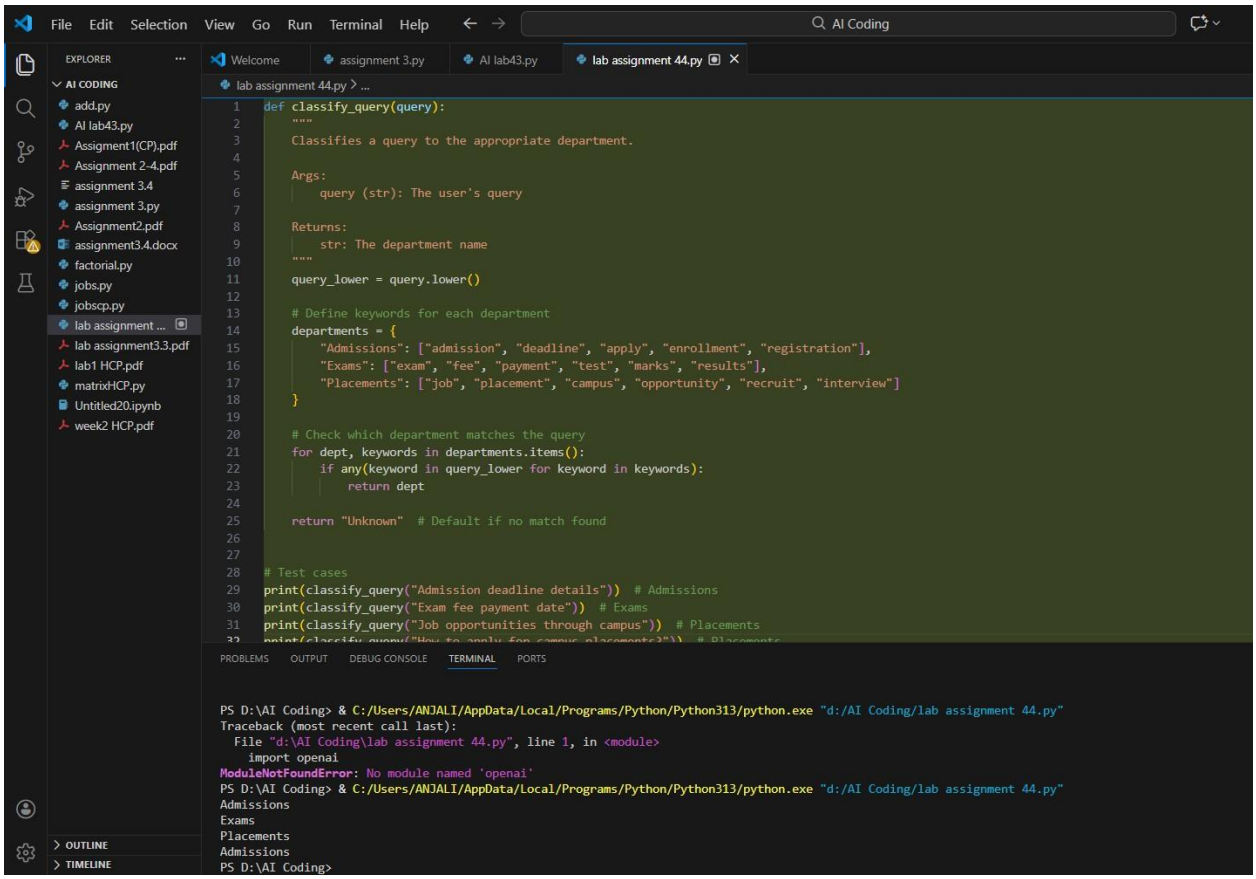
3b) Implement Zero-shot intent classification using an LLM.



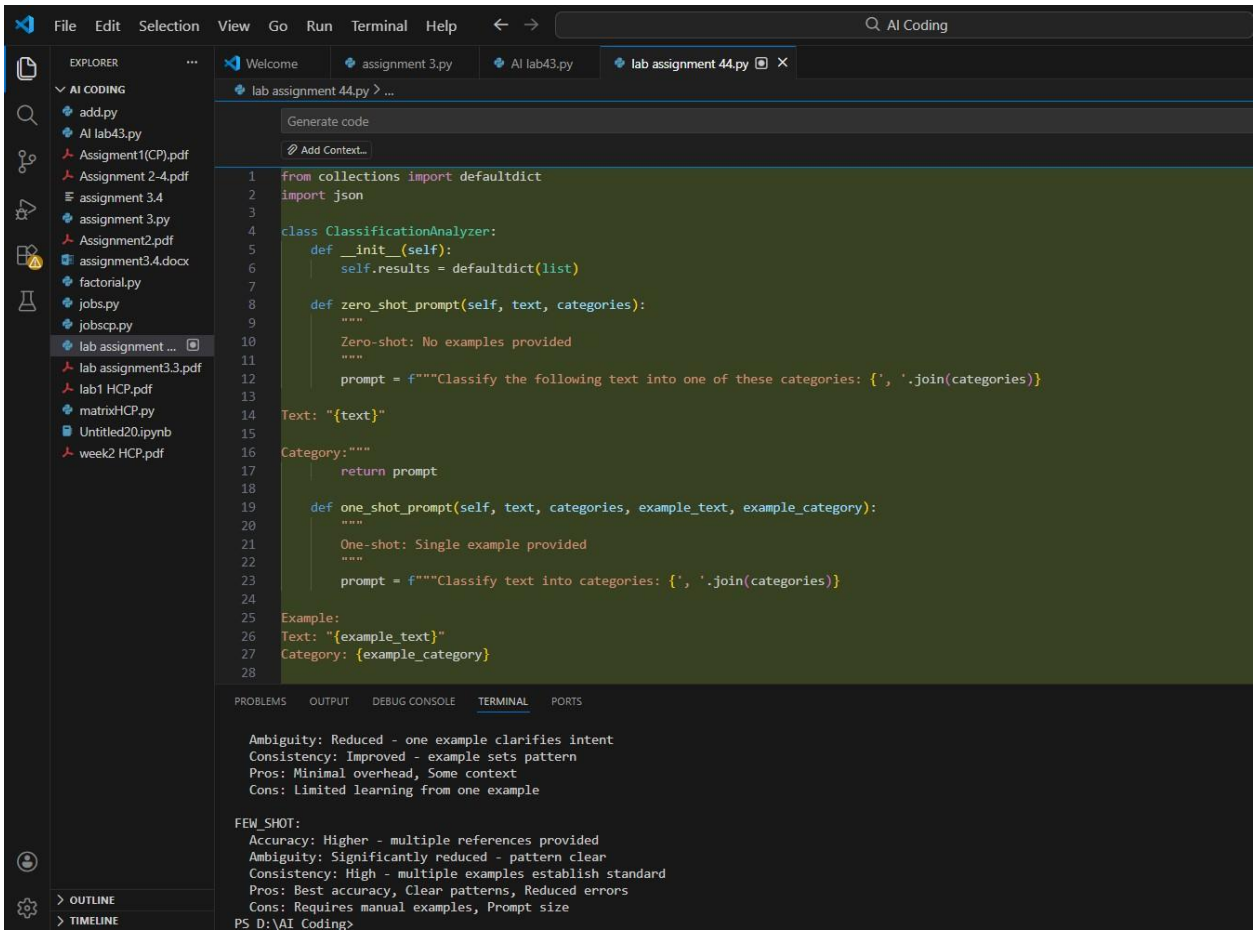
3c) mprove results using One-shot prompting.



3d) Further refine results using Few-shot prompting.


File Edit Selection View Go Run Terminal Help
AI Coding
EXPLORER
AI CODING
add.py
AI lab43.py
Assigment1(CP).pdf
Assignment 2-4.pdf
assignment 3.4
assignment 3.py
Assignment2.pdf
assignment3.4.docx
factorial.py
jobs.py
jobscp.py
lab assignment ...
lab assignment3.3.pdf
lab1 HCP.pdf
matrixHCP.py
Untitled20.ipynb
week2 HCP.pdf
OUTLINE
TIMELINE
Welcome
assignment 3.py
AI lab43.py
lab assignment 44.py
lab assignment 44.py > ...
def classify_query(query):
"""
Classifies a query to the appropriate department.
Args:
query (str): The user's query
Returns:
str: The department name
"""
query_lower = query.lower()
# Define keywords for each department
departments = {
"Admissions": ["admission", "deadline", "apply", "enrollment", "registration"],
"Exams": ["exam", "fee", "payment", "test", "marks", "results"],
"Placements": ["job", "placement", "campus", "opportunity", "recruit", "interview"]
}
# Check which department matches the query
for dept, keywords in departments.items():
if any(keyword in query_lower for keyword in keywords):
return dept
return "Unknown" # Default if no match found
Test cases
print(classify_query("Admission deadline details")) # Admissions
print(classify_query("Exam fee payment date")) # Exams
print(classify_query("Job opportunities through campus")) # Placements
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\AI Coding> & C:/Users/ANJALI/AppData/Local/Programs/Python/Python313/python.exe "d:/AI Coding/lab assignment 44.py"
Traceback (most recent call last):
File "d:\AI Coding\lab assignment 44.py", line 1, in <module>
import openai
ModuleNotFoundError: No module named 'openai'
PS D:\AI Coding> & C:/Users/ANJALI/AppData/Local/Programs/Python/Python313/python.exe "d:/AI Coding/lab assignment 44.py"
Admissions
Exams
Placements
Admissions
PS D:\AI Coding>

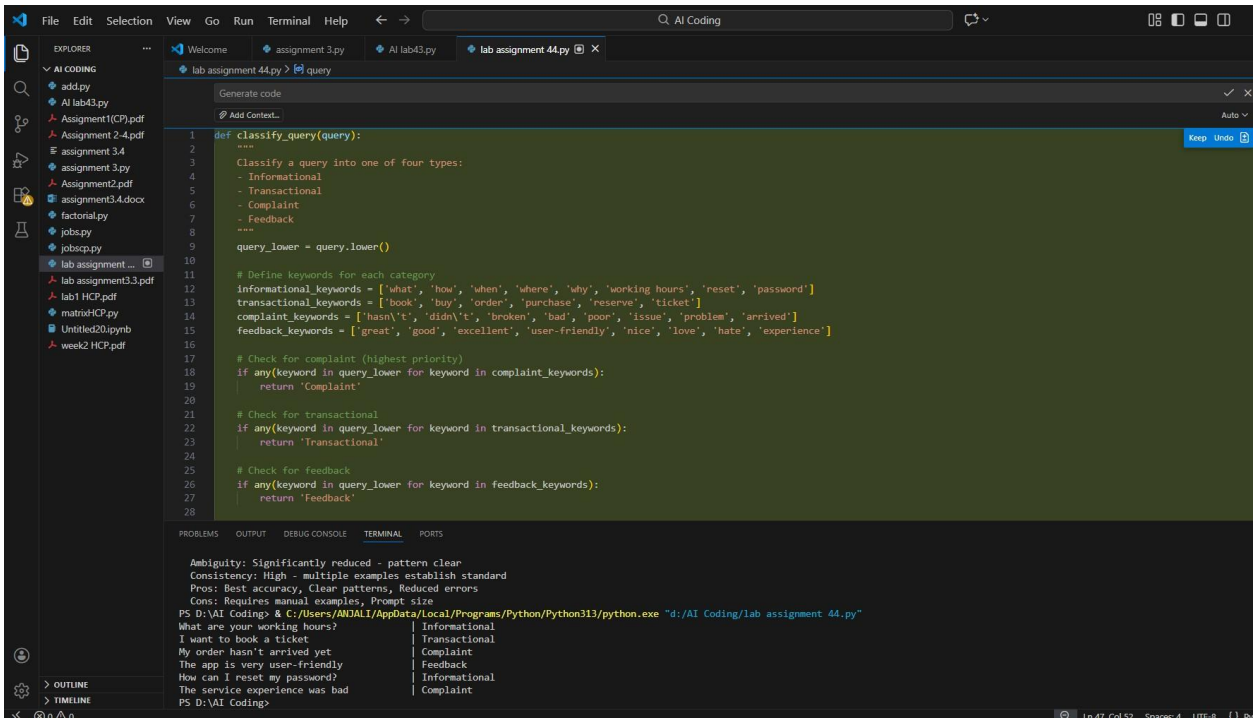
3e) Analyze how contextual examples affect classification accuracy.



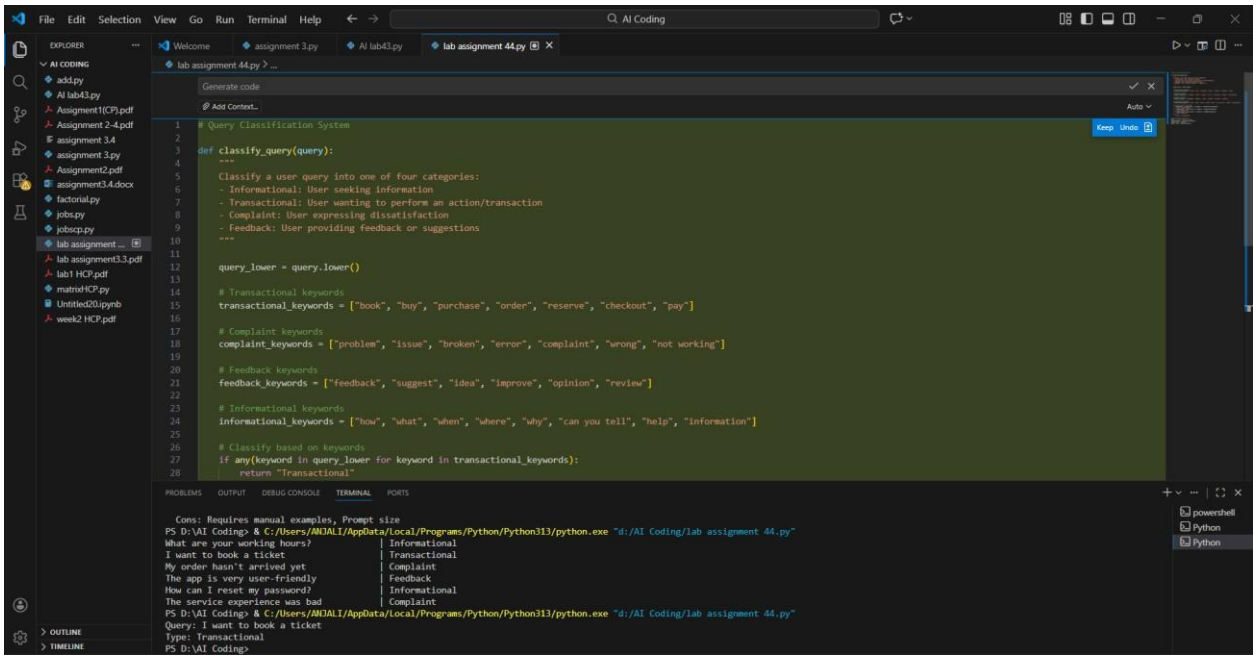
4)Chatbot Question Type Detection Scenario:

A chatbot must identify whether a user query is Informational, Transactional, Complaint, or Feedback.

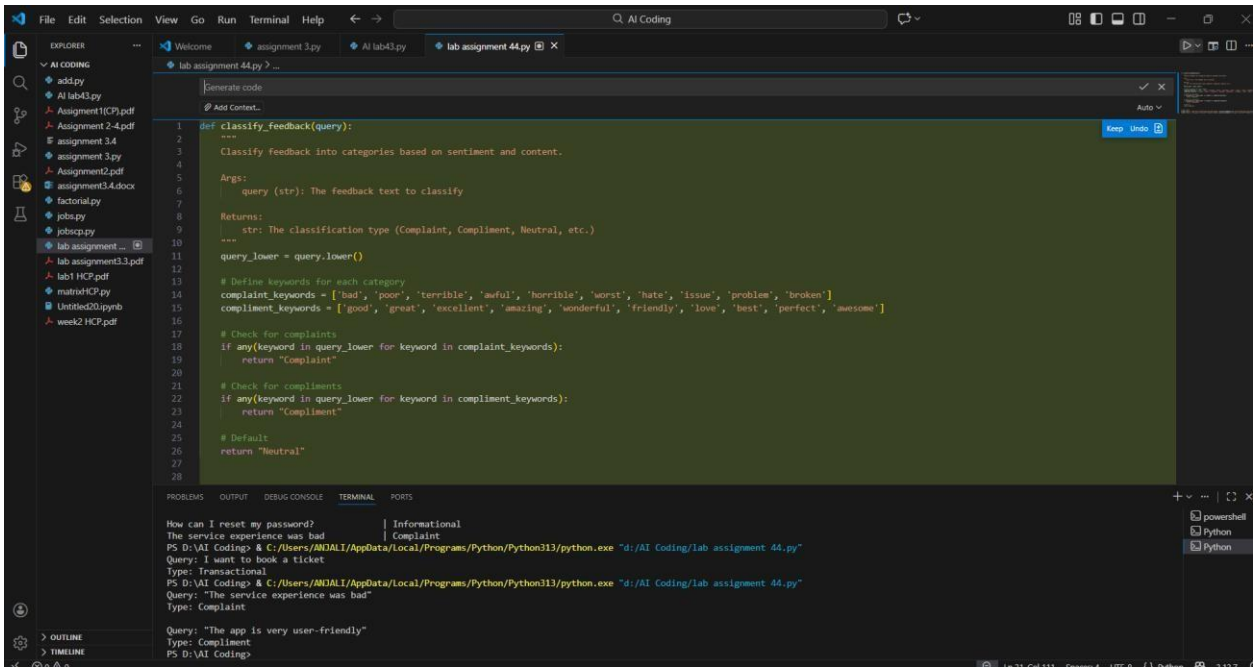
4a) Prepare 6 chatbot queries mapped to question types.



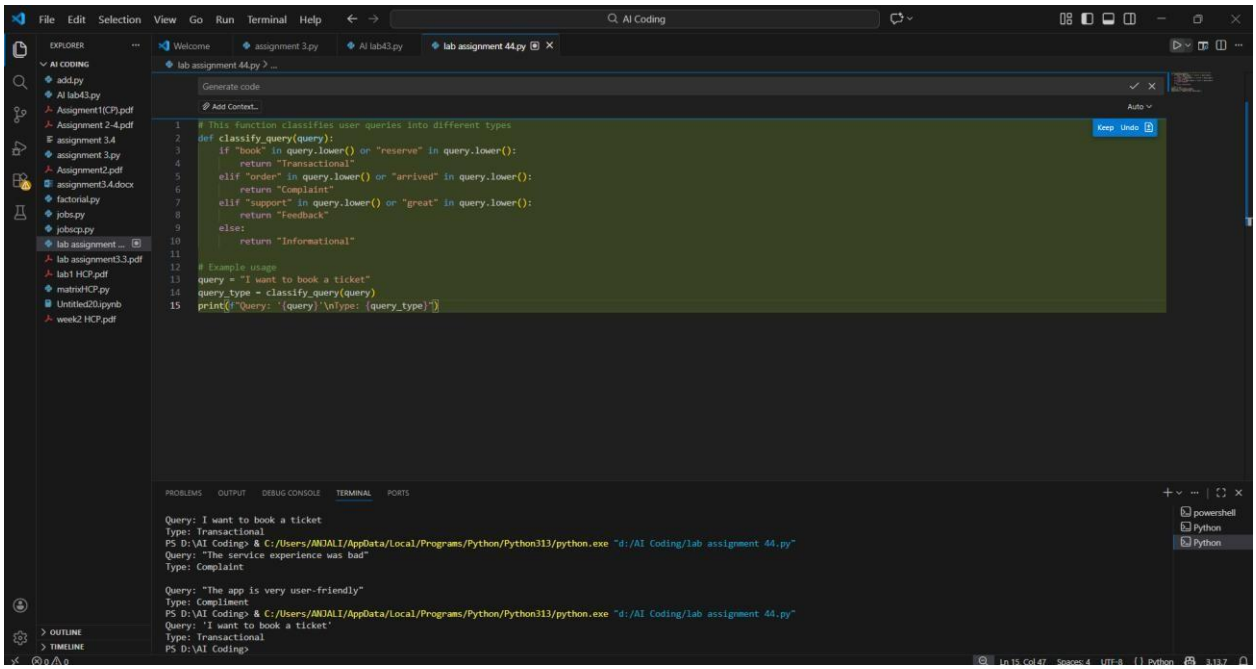
4b) Design prompts for Zero-shot, One-shot, and Few-shot learning.



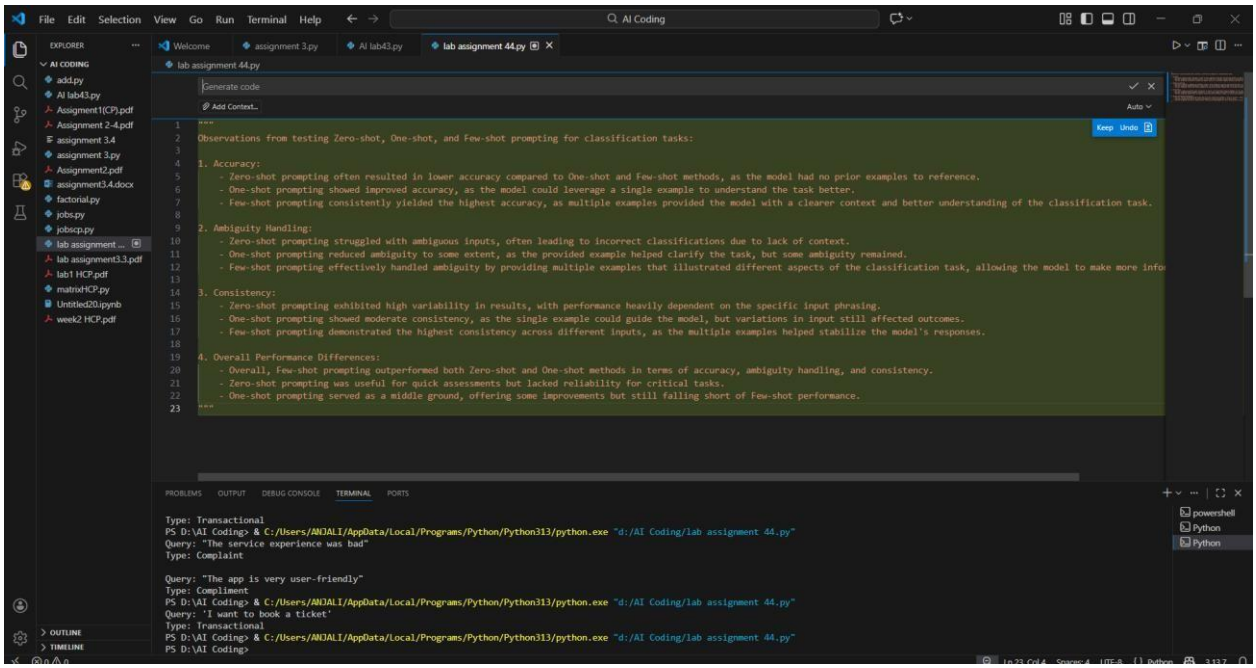
4c) Test all prompts on the same unseen queries.



4d) Compare response correctness and ambiguity handling.



4e) Document observations.

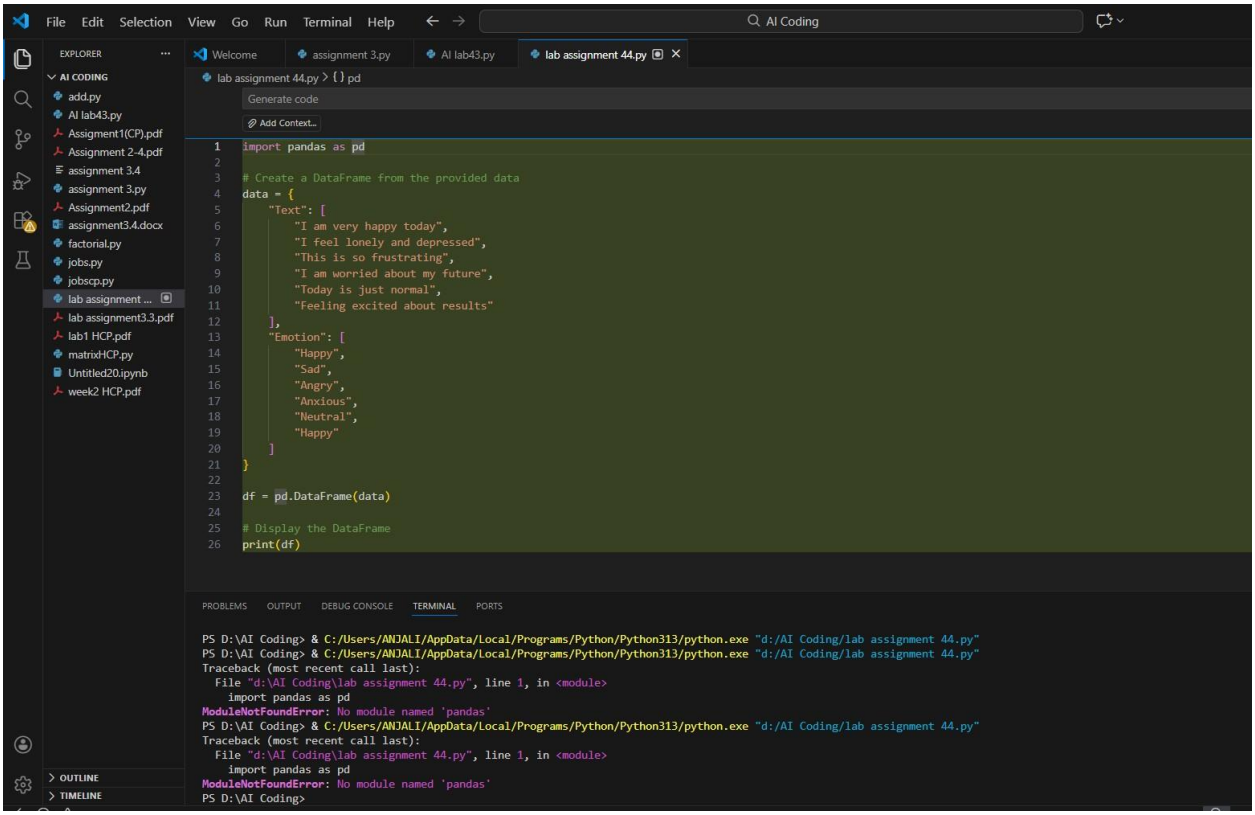


5)Emotion Detection in Text

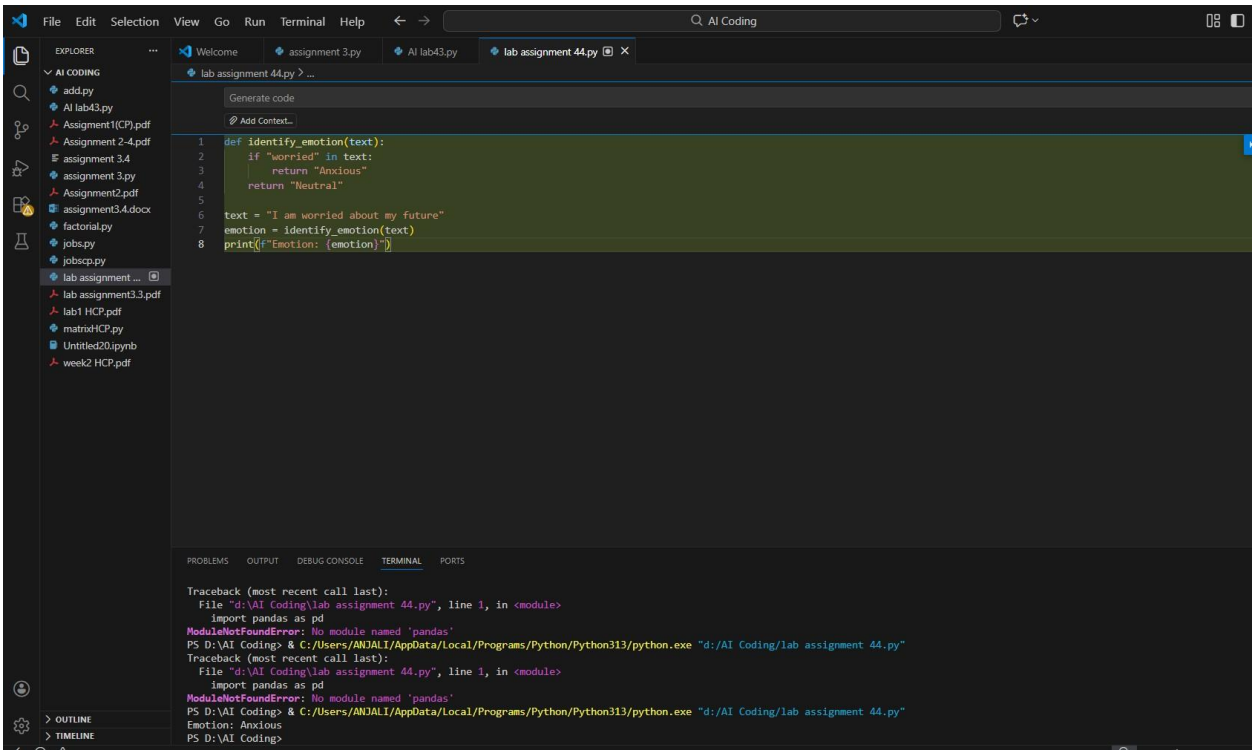
Scenario:

A mental-health chatbot needs to detect emotions: Happy, Sad, Angry, Anxious, Neutral.

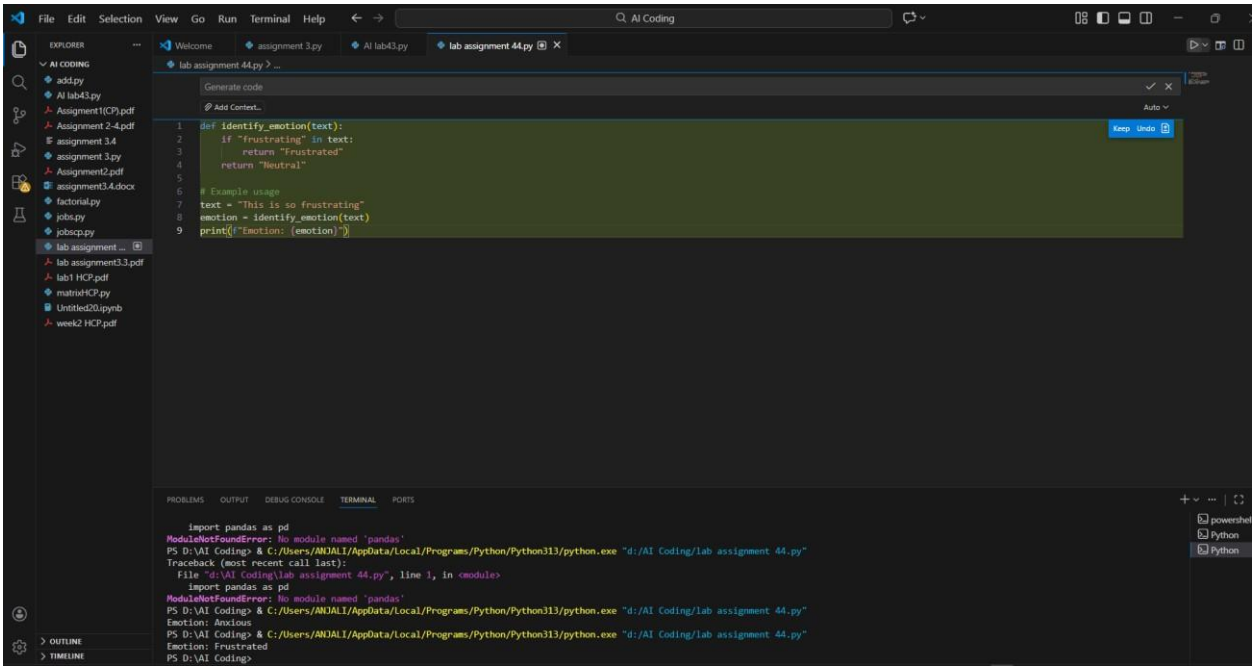
5a)Create labeled emotion samples.



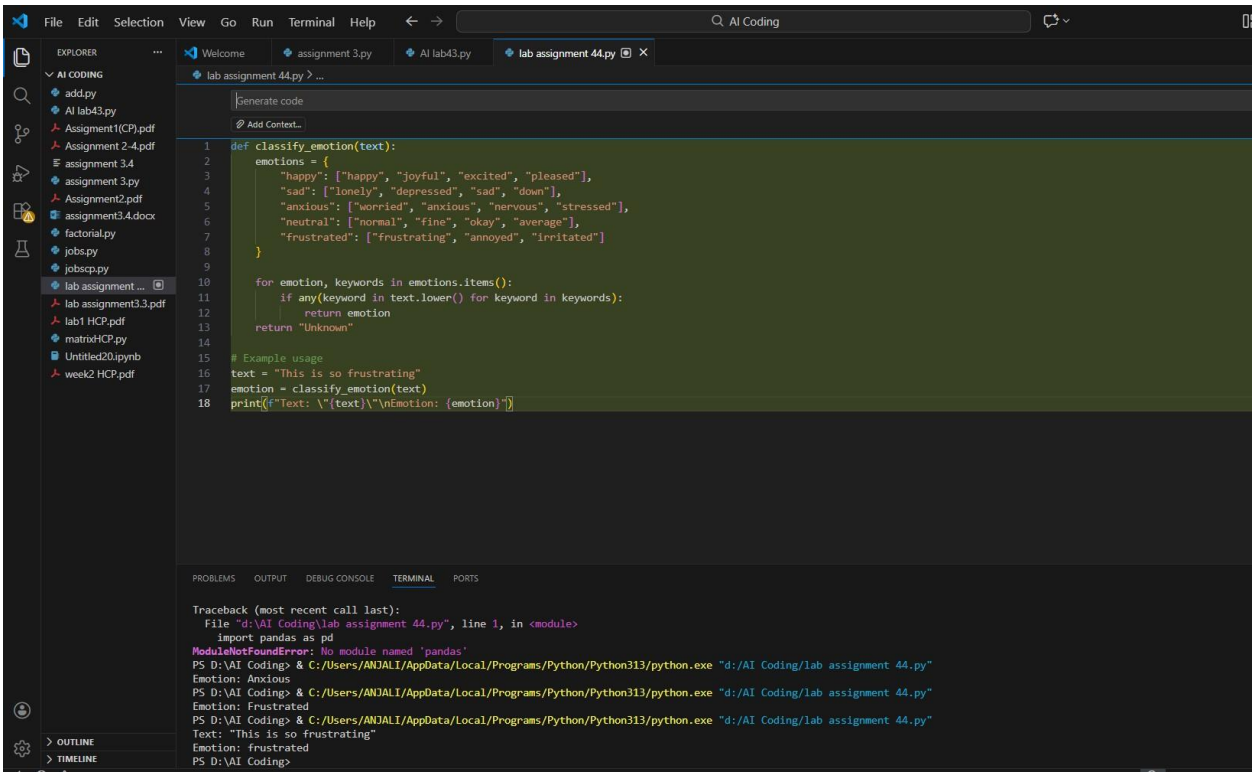
5b) Use Zero-shot prompting to identify emotions.



5c) Use One-shot prompting with an example.



5d) Use Few-shot prompting with multiple emotions.



5e) Discuss ambiguity handling across techniques.