AI ASSISTED CODING LAB

TEST-2

ENROLLMENT NO :2503A51L14

BATCH NO: 19

NAME: ROHITH GOPAGANI

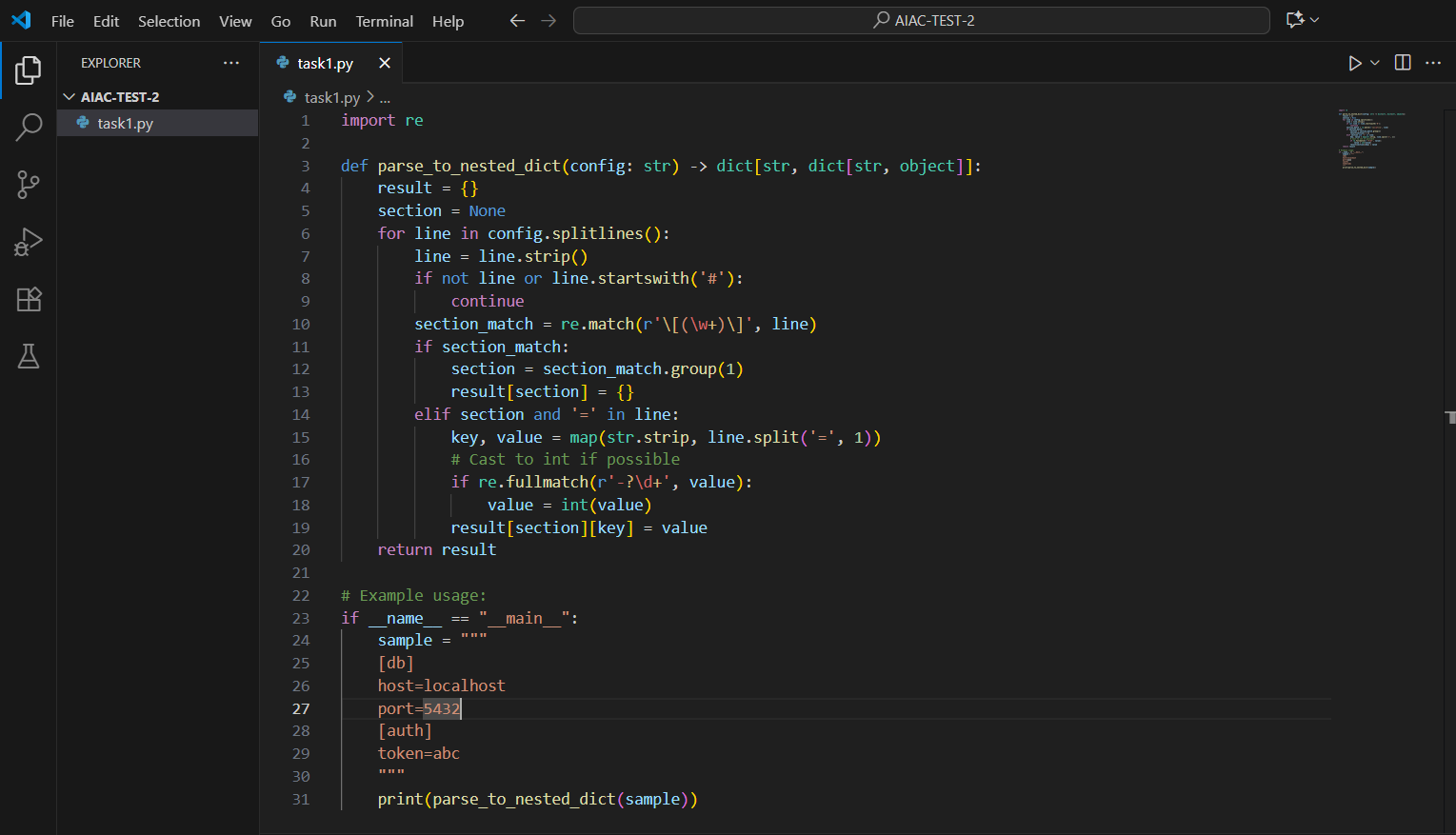
TASK DESCRIPTION 1: Parse to nested dict; cast obvious ints.  
Data & Edge Cases:  
db/auth sections example.  
AI Assistance Expectation:  
AI for robust whitespace handling.  
Constraints & Notes:  
Return dict [str, dict [str, object]].  
Sample Input

[db]  
host=localhost  
port=5432  
[auth]  
token=abc  
Sample Output  
{'db':{'host':'localhost','port':5432},'auth':{'token':'abc'}}  
Acceptance Criteria: Casts ports; preserves strings

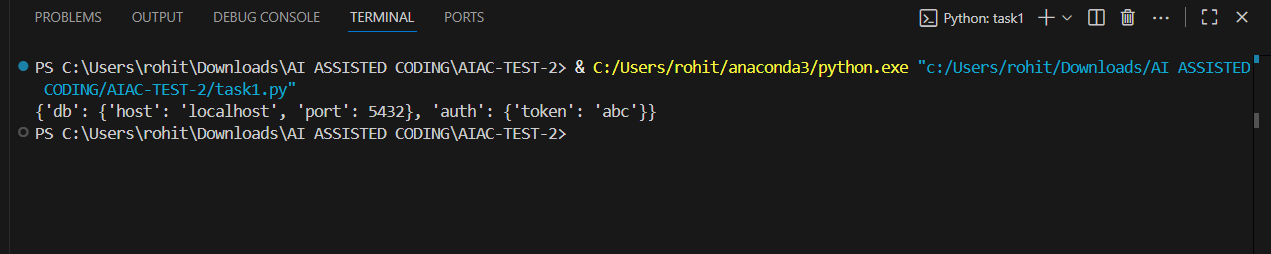
PROMPT 1:

Convert an INI-style string into a nested Python dictionary. Python dict [str, dict [str, object]] (no extra explanation). Section headers are [section]. Each section becomes a top-level key. Inside a section, lines are key=value. Trim surrounding whitespace.Ignore empty lines and lines outside any section.If a value is an integer cast it to int; otherwise keep it as str.Preserve original string values (do not split or strip inner content other than surrounding whitespace).

CODE GENERATED :



OUTPUT :



OBSERVATION :

The problem is about parsing a simple INI-style configuration into a nested Python dictionary. Each section enclosed in square brackets ([section]) becomes a dictionary key, and its entries (key=value) form key–value pairs inside it. The parser must handle whitespace robustly, skip irrelevant lines, and cast obvious integers (like 5432) to int while preserving strings as-is. The goal is to produce a dict[str, dict[str, object]] structure.

TASK DESCRIPTION 2 :

Average ticket durations in minutes from opened/closed ISO timestamps.  
Data & Edge Cases:  
List of dicts with timestamps.  
AI Assistance Expectation:  
AI for datetime parsing.  
Constraints & Notes:  
Naive timestamps fine.

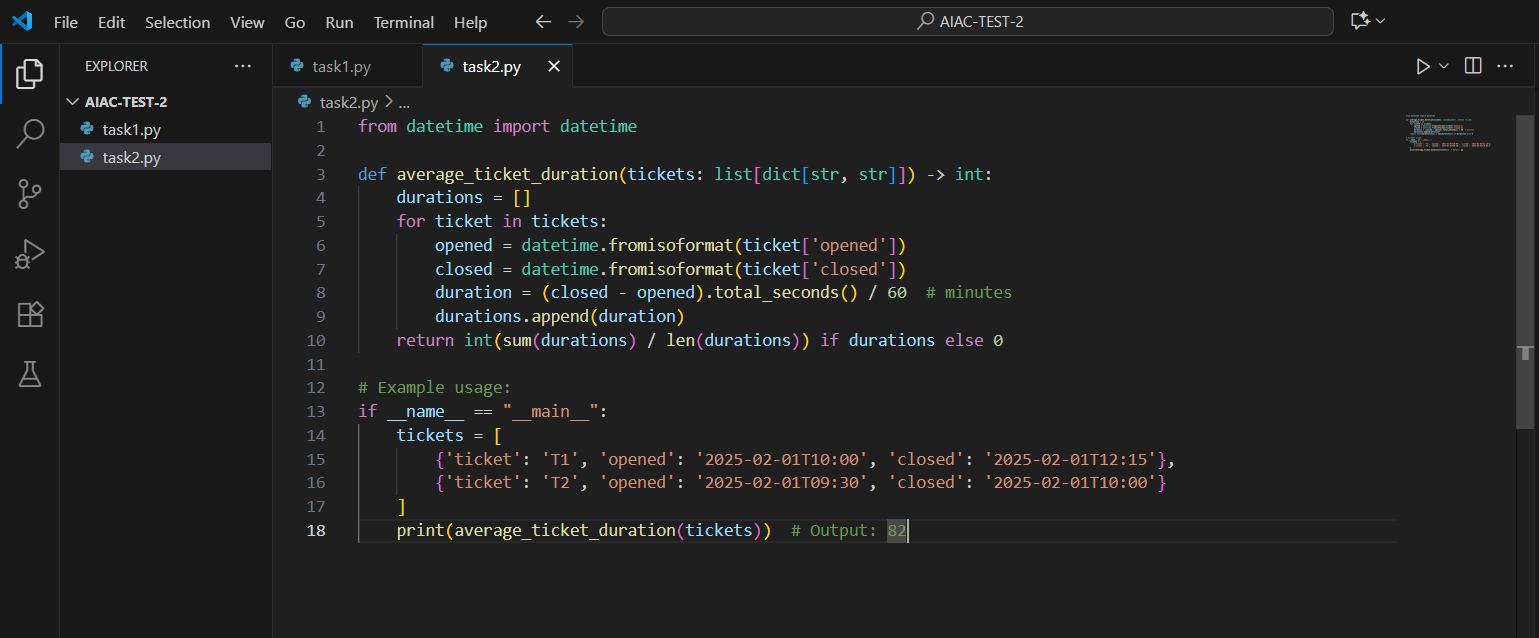
PROMPT 1 :

Write a Python function that calculates the average ticket duration in minutes.Input: A list of dictionaries, each with "ticket", "opened", and "closed" keys. Timestamps are ISO format strings (YYYY-MM-DDTHH:MM).Task: Parse timestamps, compute the duration (in minutes) for each ticket, and return the integer average of all durations.

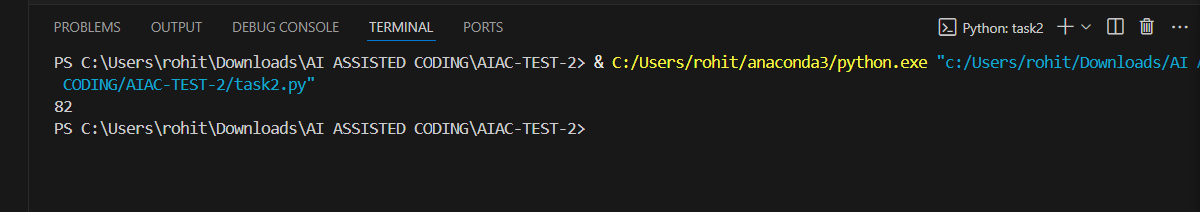
Constraints:

* + Naive datetime parsing is fine.
  + Return a single integer (floor/truncate if needed).

CODE GENERATED :



OUTPUT :



OBSERVATION :

The task is to compute the average duration of tickets using their opened and closed ISO-format timestamps. Each ticket is represented as a dictionary with "opened" and "closed" fields. The solution must parse the timestamps into datetimes, calculate the difference in minutes, and then average the values. Naive timestamp parsing is acceptable since no timezone handling is required. The output should be a single integer representing the average duration across all tickets.