**✅ API Requirements**

* **Endpoint**: POST https://your-api.com/api/
* **Input**: Accepts a file (like question.txt) via multipart/form-data

bash

CopyEdit

curl https://your-api.com/api/ -F "@question.txt"

* **Processing**:
  + Read and parse the task
  + Scrape, load, or query data
  + Analyze (e.g., correlation, regression)
  + Visualize (e.g., return plot as base64 image URI, under 100 kB)

**✅ Response Format**

Return a **JSON array of exactly 4 elements**:

json

CopyEdit

[

1, // e.g., number answer

"Titanic", // e.g., string or label

0.485782, // e.g., regression/correlation result

"data:image/png;base64,..." // base64 plot (under 100kB)

]

**✅ Regression Example (Python logic inside your backend)**

python

CopyEdit

import numpy as np

from sklearn.linear\_model import LinearRegression

import matplotlib.pyplot as plt

import base64

import io

X = df[['Rank']].values

y = df['Peak'].values

model = LinearRegression().fit(X, y)

slope = model.coef\_[0]

intercept = model.intercept\_

r\_squared = model.score(X, y)

# Plot with red dotted regression line

plt.scatter(X, y)

plt.plot(X, model.predict(X), 'r--')

plt.xlabel("Rank")

plt.ylabel("Peak")

buf = io.BytesIO()

plt.savefig(buf, format='png', dpi=150)

plt.close()

img\_str = base64.b64encode(buf.getvalue()).decode()

**✅ Tips to Maximize Evaluation Score**

* Ensure all 4 answers are correct and present
* JSON must be exactly an array of 4 items
* Regression plots must:
  + Be red dotted line
  + Show axes and labels
  + Be under 100kB
* If correlation/regression needed, match required precision (e.g. ±0.001)

When you’re ready to test or need help deploying (e.g., via Render, Railway, FastAPI, or Flask), let me know — I’ll help you get your API live and score-ready!