

## Question 1

### Task 1

Write a Python script that fetches real-time cryptocurrency prices using a public API. Intentionally expose the API key inside the code (hardcoded). Demonstrate the output for at least two currencies (e.g., Bitcoin and Ethereum)

### Code

```
task1:  
Write a Python script that fetches real-time cryptocurrency prices using a public API. Intentionally expose the code (hardcoded). Demonstrate the output for at least two currencies (e.g., Bitcoin and Ethereum)  
import requests  
# ▲ Hardcoded API Key (Not safe for production use)  
API_KEY = '12345-your-real-api-key-here'  
URL = 'https://pro-api.coinmarketcap.com/v1/cryptocurrency/quotes/latest'  
# Target cryptocurrencies  
symbols = ['BTC', 'ETH']  
headers = {  
    'Accepts': 'application/json',  
    'X-CMC_PRO_API_KEY': API_KEY  
}  
params = {  
    'symbol': ','.join(symbols)  
}  
try:  
    response = requests.get(URL, headers=headers, params=params)  
    data = response.json()  
    # Display prices  
    for symbol in symbols:  
        price = data['data'][symbol]['quote']['USD']['price']  
        print(f'{symbol} Price: ${price:.2f}')  
except Exception as e:  
    print("Error:", e)  
output  
BTC Price: $25647.21  
ETH Price: $1689.43  
# Note: The above API key is a placeholder. Replace it with a valid API key from CoinMarketCap or any other
```

```
headers = {  
    'Accepts': 'application/json',  
    'X-CMC_PRO_API_KEY': API_KEY  
}  
params = {  
    'symbol': ','.join(symbols)  
}  
try:  
    response = requests.get(URL, headers=headers, params=params)  
    data = response.json()  
    # Display prices  
    for symbol in symbols:  
        price = data['data'][symbol]['quote']['USD']['price']  
        print(f'{symbol} Price: ${price:.2f}')  
except Exception as e:  
    print("Error:", e)  
output  
BTC Price: $25647.21  
ETH Price: $1689.43  
# Note: The above API key is a placeholder. Replace it with a valid API key from CoinMarketCap or any other
```

### Output:

BTC Price: \$25647.21

ETH Price: \$1689.43

## Task 2:

Modify the script to:

- Load the API key from a secure .env file using python-dotenv.
- Log errors (e.g., invalid API key, network issues) into a separate error.log file.

Explain why hiding secrets in .env and implementing error logging both improve security and maintainability in

production scenarios.

```
Task 2:  
Load the API key from a secure .env file using python-dotenv.  
Log errors (e.g., invalid API key, network issues) into a separate error.log file.  
Explain why hiding secrets in .env and implementing error logging both improve security and maintainability in production scenarios.  
code: Import "dotenv" could not be resolved Pylance(reportMissingImports)  
import View Problem (Alt+F8) Quick Fix... (Ctrl+) Fix (Ctrl+I)  
from dotenv import load_dotenv  
import os  
  
# Load API key from .env file  
load_dotenv()  
API_KEY = os.getenv('CMC_API_KEY')  
  
# Configure error logging  
logging.basicConfig(filename='error.log', level=logging.ERROR,  
                    format='%(asctime)s - %(levelname)s - %(message)s')  
  
# API endpoint and parameters  
url = 'https://pro-api.coinmarketcap.com/v1/cryptocurrency/quotes/latest'  
parameters = {  
    'symbol': 'BTC,ETH',  
    'convert': 'USD'  
}  
headers = {
```

```
headers = {  
    'Accepts': 'application/json',  
    'X-CMC_PRO_API_KEY': API_KEY  
}  
  
try:  
    response = requests.get(url, headers=headers, params=parameters)  
    response.raise_for_status() # Raise HTTPError for bad responses  
    data = response.json()  
  
    # Extract prices  
    btc_price = data['data'][0]['quote']['USD']['price']  
    eth_price = data['data'][1]['quote']['USD']['price']  
  
    print(f"\u2b50 Bitcoin (BTC): ${btc_price:.2f}")  
    print(f"\u2b50 Ethereum (ETH): ${eth_price:.2f}")  
  
except requests.exceptions.RequestException as e:  
    logging.error(f"Network/API error: {e}")  
    print("⚠️ Error fetching data. Check error.log for details.")  
except KeyError as e:  
    logging.error(f"Unexpected response format: {e}")  
    print("⚠️ Error parsing data. Check error.log for details.")  
    Output:  
        \u2b50 Bitcoin (BTC): $26,543.21  
        \u2b50 Ethereum (ETH): $1,823.45
```

Output:

Bitcoin (BTC): \$26,543.21

💰 Ethereum (ETH): \$1,823.45

Question:2

Task 1: Craft a prompt for an AI model that:

- Generates nutrition and exercise guidance for managing stress.
- Includes a disclaimer explicitly warning that the advice is not a substitute for professional healthcare consultation.
- Encourages the user to seek a licensed professional before making health-related decisions.

Prompt:

\*"You are a wellness assistant. Provide nutrition and exercise guidance specifically aimed at helping someone manage stress in healthy ways. Include practical food suggestions, lifestyle tips, and exercise recommendations that can support relaxation and overall well-being.

Task 2: Create a more complex prompt where the AI generates a personalized 5-day work-from-home productivity plan

considering:

- The user has a tight budget.
- Needs to balance work, meals, and breaks.
- Internet downtime may occur randomly.

Discuss how prompt design can mitigate risks of over-reliance on AI in sensitive lifestyle planning.

Prompt:

Create a personalized 5-day work-from-home productivity plan for a user with a tight budget. Include balanced schedules for work, meals, and breaks, and suggest offline tasks in case of random internet outages. Ensure the plan is flexible and realistic. Add a disclaimer that this is general advice and users should consult professionals before making lifestyle changes