Continuous Assessment Report

Jose Alberto Cruz Sanchez (23191236)

Programming for Artificial Intelligence - H9PAI

MSc in Artificial Intelligence - MSCAI1A

National College of Ireland

1 API Key Setup:

An API key is requested directly on the NASA website to access NASA API servicer. In this way, the NASA URL is configured from where we will be obtaining the data.

```
# API key for accesing NASA's API services.
API_KEY = "PYateWtTaDdIvDTDEy82JoVBqKaX0FBCFgmOpOiE"
# Base URL for NASA's Astronomy Picrture of the Day (APOD) API
API_URL = "https://api.nasa.gov/planetary/apod"
```

✓ 0.0s

Generate our code to save the received data into a JSON file. Create an error condition in case the document cannot be saved, it shows us the error.

```
def save_data_to_json(data: list, filename: str) -> None:
    try:
        # Open the file in write mode
        with open(filename, 'w') as f:
            # Use json.dump to serialize the data and write it to the file
            json.dump(data, f)
        # Log a success message if the file is saved successfully
        logger.info(f"JSON file saved successfully: {filename}")
    except Exception as e:
        # Log an error message if any exception occurs during file saving
        logger.error(f"Error saving JSON file: {e}")
```

✓ 0.0s

Setup the parameters to select the pictures of the days we want to receive in our JSON file.

```python
# Parameters
start_date = "2020-01-01"  # Start date for fetching APOD data (YYYY-MM-DD)
end_date = "2020-10-27"  # End date for fetching APOD data (YYYY-MM-DD)
filename = "apod_data.json"  # Name of the JSON file to save the data
try:
    # Fetch APOD data for the spicified date range and save to JSON
    data = fetch_multiple_apod_data(start_date, end_date)
    # Check if data is not empty
    if data:
    # Save data Json file
        save_data_to_json(data, filename)
except Exception as e:
    logger.error(f"Task failed: {e}")
# Log a completion message
logger.info("Task complete")
```

✓ 29m 45.6s

```
ERROR:__main__:HTTP error for date 2020-06-10: 404 Client Error: Not Found for url: https://api.nas
ERROR:__main__:Error retrieving data for date: 2020-06-10
INFO:__main__:JSON file saved successfully: apod_data.json
INFO:__main__:Task complete
```

2 Challenges faced and important decision made during implementation:

- One of the main challenges I encountered while generating the code was with Fetch, as I am not familiar with its implementation. It took me a while to understand the logic behind it. I watched some coding tutorials and reviewed codes I found on GitHub to help me overcome the hurdle.
- Another challenge I faced was managing large amounts of data and creating JSON file to receive all the pictures data request from NASA's website. Converting the received data into a directory to keep it organized was also a bit difficult to handle and process. I think this part of the coding process stressed me out the most, as the code took a long time to compile.

3 Problem 4:

How many data points are there in this data set?

We have a total of 150 data points.

```python
# Number of data points
num_data_points = df.shape[0]
print(f"\nNumber of data points: {num_data_points}")
```

What are the data types of the columns?

We have two different data types, object and float.

```
# Data types of the column
column_data_types = df.dtypes
print(f"\nNames of the columns: {column_data_types}")
```

What are the column names?

We have 7 columns with the following names, “sepal_length”, “sepal_width”, “petal_length”, “petal_width”, “species”, “petal_relation” and “sepal_relation”.

```
# Names of the columns
column_names = df.columns.tolist()
print(f"\nNames of the columns: {column_names}")
```

How many species of flower are included in the data?

We have three different types of species, “setosa”, “versicolor”, and “virginica”.

```
# Number of unique flower species
num_flower_species = df['species'].nunique()
print(f"\nNumber of unique flower species: {num_flower_species}")
```

```
Number of data points: 150

Data types of the column: sepal_length      float64
sepal_width       float64
petal_length      float64
petal_width       float64
species            object
petal_relation    float64
sepal_relation    float64
dtype: object

Names of the columns: ['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species', 'petal_relati

Number of unique flower species: 3

Species counts: species
setosa        50
versicolor    50
virginica     50
Name: count, dtype: int64
```

What can you infer from this?

It is a dataset to see the relationship of between sepas, as well as their variations in each species.

4 Conclusion:

Upon completing this project, I acquired valuable skills and experience in handling large datasets, implementing fetch, and organizing information in JSON files. I learned to overcome technical obstacles and optimize the compilation process. This experience allowed me to develop my ability to analyze complex problems, seek effective solutions, and improve my

code for more efficient results. I am ready to apply this knowledge in tuture projects and continue enhancing my coding skills.