Section 1 - Introduction and Data Analysis

• 1.1 Motivation and Questions:

Q1. Which artist is more populuar? (Choose two from our list)

When I talk to people about artists and songs, I usually notice everyone likes different artists and I was thinking, it's a good opportunity that I use this assignment for compare som artists to each others.

Q2. How many times was the artist searched this year? (By month)

I would like to continue my research by finding out their Wikipedia's page's view to see if they are as popular as they may are in Spotify.

Q3. Wanna learn English in a fun way?

English is my third language and I always struggled to learn it with traditional strategy of language learning. So in this question, I can choose an artist and its song, to learn about the definition of a random word from the lyrics.

• 1.2 APIs and Resources:

I didn't use CoPilot at all. I had a problem in my laptop and got json.JSONDecodeError while reading the JSON files from Wikipedia API so I asked ChatGPT what's this error about and how I can handle it. I also watched some YouTube tutorials to learn working with different libraries.

• List of APIs:

- **Spotify API:**

- Artists:

```
url = "https://dit009-spotify-assignment.vercel.app/api/v1/
artists/7dGJo4pcD2V6oG8kP0tJRR/"
```

- Albums:

```
url = "https://dit009-spotify-assignment.vercel.app/api/v1/artists/
7dGJo4pcD2V6oG8kP0tJRR/albums"
```

- Top Tracks:

```
url = "https://dit009-spotify-assignment.vercel.app/api/v1/artists/
7dGJo4pcD2V6oG8kP0tJRR/top-tracks"
```

- **Lyrics API:**

```
url = "https://api.lyrics.ovh/v1/Eminem/Mockingbird"
```

- **Dictionary API:**

```
url = "https://api.dictionaryapi.dev/api/v2/entries/en/happy"
```

- **Wikipedia API:**

```
url = "https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/
de.wikipedia.org/all-access/all-agents/Eminem/monthly/20230101/20231231"
```

- **Libraries:**

- requests
- json
- os
- tabulate
- pandas
- matplotlib
- re

• 1.3 Team Division and Peer Assessment:

All 1000 Euros would be mine because I didn't had any group and I did this project alone.

Section 2 - Results and Analysis

- **2.1 Which artist is more populuar? (Choose two from our list)**

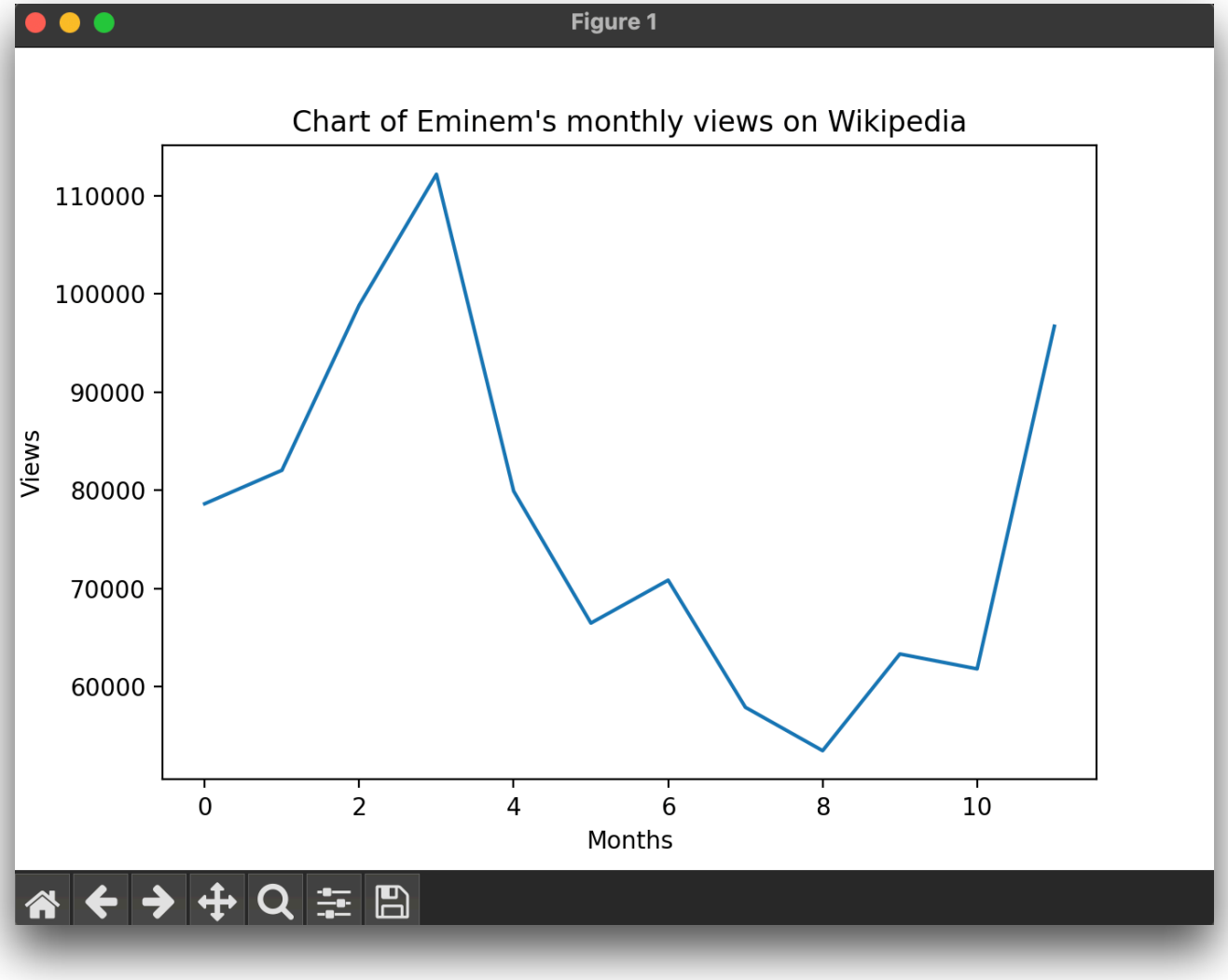
Example of data I gathered was number of albums and singles, number of tracks, number of follower and top 10 tracks from the chosen artists. For finding answers to this question, I saved 10 artists data by help of Spotify API and saved them in thre folders called **artists**, **albums** and **tracks**. Getting data from Spotify API used to take some time and made my program slow. That's why I decided to save those files in advance and use Tham during the runtime. Pandas and Tabulate library was used for creating a table of stats.

Name	Eminem	50 Cent
Albums	20	7
Singles	0	13
Tracks	469	125
Followers	95849831	16625782
Top Tracks	Without Me Mockingbird Houdini The Real Slim Shady Lose Yourself Godzilla (feat. Juice WRLD) Love The Way You Lie Till I Collapse Superman Stan	In Da Club Candy Shop 21 Questions P.I.M.P. Many Men (Wish Death) Baby By Me Just A Lil Bit If I Can't Disco Inferno Window Shopper

Answer: Eminem has introduced 20 albums while 50 Cent has introduced 7 albums only. But 50 Cent has 13 singles while Eminem has none. For me, it was also interesting that Eminem has almost 4 times more tracks and 6 times more followers than 50 Cent on Spotify.

- **2.2 How many times was the artist searched this year? (By month)**

For this question, my API was online and I could find the information during the runtime. I used Wikipedia API for this question and during runtime, my program requested and got data from Wikipedia and showed the answer as a chart. For creating a chart, I used Matplotlib library.



Answer: The chart shows that some months, Eminem's Wikipedia page's viewers passed 80.000 which made me wondering why would so many people, search his name on Wikipedia each month.

- **2.3 Wanna learn English in a fun way?**

For answering this question, I used Lyrics and Dictionary APIs. I also used very simple regex (re library) and random library to randomly find and match a word from the chosen lyrics and find its definition.

```
Enter the name of the artist from our list above: eminem
Eminem is chosen.

Enter the song from our list above: lose yourself
Lose Yourself is chosen.

1: Play the game:
2: Go Back
3: Exit the program.

Choose one of the following options from the Menu (1-3): 1
Today's word is - MOMENT -

Defination of MOMENT is: A brief, unspecified amount of time.
```

Answer: There is no fixed answer to this question because every time I run the program, the word changes which means the definition of the word I get changes. But it was actually fun and I ran this part of my program many times.

Section 3 - Learning Goals and Code Explanations

- **3.1 Code Quality**

I avoided using '**while True**' in my project because we discussed that this is a bad practice and instead, I used flags to break the loop. I also tried to have as much '**try — except**' as I could for avoiding my program interrupts during the runtime. I also used a Python built-in function '**.title( )**' to make sure that no matter if the user types lowercase or uppercase, it will match.

```
275  def choose_two_artists():
276      chosen_artists_name = []
277      chosen_artists_id = []
278      index = 0
279      try:
280          while index < 2:
281              artist = input("Which artist is more populuar (Choose two)? ").title()
282
283              if artist.title() in database.keys():
284
285                  if artist.title() in chosen_artists_name:
286                      print(f"{artist} is already chosen!")
287
288                  else:
289                      chosen_artists_name.append(artist)
290                      chosen_artists_id.append(database[artist.title()])
291                      index += 1
292                      print(f"{artist} added for comparing.")
293
294              else:
295                  print(f"{artist} is not in our list.")
296
297          return chosen_artists_name, chosen_artists_id
298
299      except Exception as e:
300          print(f"Something went wrong: {e}")
301
```

- **3.2 Reflections**

- **Regex:** We talked about and learned regex 'regular expressions' in the lectures. For question 3, I used the pattern **r'\b\w+\b**' to make sure that it will match to 1 or more word character **[A-Za-z0-9_]** followed by boundaries before and after the word.
- **Functions:** I have created as many functions as possible and tried to follow the **S** from the **SOLID** principle which means that every method (in our case, functions) has a single responsibility.
- **Parsing JSON:** Reading and parsing JSON files was scary from the beginning but as more as I worked with it, it became more easier and also fun. Every API used stored data in JSON files and I learned a lot working with them in this project.

Section 4 - Multiple Data Sources

- **4.1 Integrating API Data**

I used 4 APIs in my project. Spotify was the most difficult one to learn how to use it. Lyrics and Dictionary APIs were very easy to use but I found a serious issue in Wikipedia API that I couldn't manage to fix without ChatGPT. I got the error **json.JSONDecodeError** and it was very hard to handle it.

For more readability, I tried to be organised and created many different folders for different types of data. I gathered data from Spotify API about the artists, their albums and their top tracks. Additionally, I used Lyrics API for save songs' lyrics as JSON files, Wikipedia API to find out the artist's Wikipedia page's views monthly and Dictionary API to find and save definitions of random words from the saved lyrics. So for every data, I had different folders such as artists, albums, tracks, dictionary, etc. I also had another folder called charts to save a png of every chart created during the runtime. I created some functions that I reused later and it increased the readability of my program. One of them was the function read_json(data) and I reused it in only one line of code, anywhere I needed to read a json file.

- **4.2 Lessons learned**

Some benefits of integrating JSON files are that the readability of my program increased and I didn't need to worry about that I will see those data everywhere on my screen. So I had access to a bunch of data and could use them whenever I needed them. Using other APIs helped me understand how powerful my program can become by having different functionalities and features. Using other APIs after working with Spotify API became easier but it had also some challenges. Issues I faced were some lyrics didn't exist in Lyrics API's database. Same with Dictionary that some word's definitions were missing. It was also challenging to mix up two different APIs together to build up a new functionality in my program. But as more I worked on the project, as more I felt comforted by using APIs and JSON files.

Section 5 - Improving your Application

- **5.1 Reflections on Error Handling:**

User interaction is a part of my program and that means I will never know the actions of the users. So I had to think further and take care of error handling so that my program doesn't interrupt if unexpected behavior occurs. My program requires to match some artists' names and their songs to users' input and without error handling, it could be very disturbing to start the program from the beginning again and again after every errors.

```
def submenu_option_one():
    print("\n1: Compare the Artists: ")
    print("2: Go Back")
    print("3: Exit the program.")

    invalid_input = False
    while not invalid_input:
        try:
            submenu_option = int(input("\nChoose one of the following options from the Menu (1-3): "))

            if 0 < submenu_option < 4:
                invalid_input = True
                return submenu_option
            else:
                print("The number shall be between 1 to 3.")

        except ValueError:
            print("Invalid input! Enter only numbers!")

        except Exception as e:
            print(f"Something went wrong: {e}")
```

This is an example of one of the error handling I did. First of all, the input I want from the user is a number between 1-3. So based on that I used a while loop to ensure that the user types the answer repeatedly until it meets my requirements. Then I have a try-except block to make sure that if any unexpected behavior happens, It will raise an exception and let the user try again. I have also used if-statement here as the condition to make sure that the user enters a number between 1-3 before my program continues.

```
def user_interaction():
    type_of_json_list = ['artists', 'albums', 'top-tracks']
    invalid_input = False

    try:
        while not invalid_input:
            type_of_json = input("Choose one of the options: Artists -- Albums -- Top-Tracks: ").lower()

            if type_of_json in type_of_json_list:
                if type_of_json == "artists":
                    type_of_json = ""

                invalid_input = True
                return type_of_json
            else:
                print("Invalid input! Please choose from Artists, Albums, or Top-Tracks.")

    except Exception as e:
            print(f"Something went wrong! {e}")
```

- **5.2 Using Plotting Libraries**

Creating a plot, more specific, a chart of my result was very easy to understand and use. Because I have already programmed the logic of my functionality and had the result in my hands. The only thing that was left was to call the result inside the plot.

```
def wikipedia_stats(data, artist):
    data = read_json(f'MusicData/resources/wikipedia/{artist}_wiki.json')

    views = []
    for view in data['items']:
        views.append(view['views'])

    total_views = 0
    for view in views:
        total_views += view

    plt.title(f"Chart of {artist}'s monthly views on Wikipedia")
    plt.ylabel("Views")
    plt.xlabel("Months")
    plt.plot(views)

    save_chart(artist)

    plt.show()


def save_chart(artist):
    path = f'MusicData/resources/charts/{artist}_chart.png'
    plt.savefig(path, format='png', dpi=300)
```

Using the libraries, Pandas together with Tabulate was difficult to understand. I watched many tutorials and read documentation before I managed to use it my program.