Software Development Process

Programming Project:

Utilization of Open Data

NASA NEO (Near Earth Objects) Data Analysis Project Report

Mohammed Shameem

Karuvara Kunnath

1. Task Description and Objectives

The goal of this project is to create a functional Python program that uses **open data** to analyze real asteroid information from NASA's *Near Earth Objects (NEO)* API.
This API provides publicly accessible data about asteroids approaching Earth, including their size, speed, distance, and hazard status.

Objectives

- Fetch data from an open government API
- Process and analyze the data using Python
- Extract meaningful insights (largest, closest, hazardous asteroids)
- Visualize the results with charts
- Export structured data to CSV
- Demonstrate modular, PEP8-friendly programming practices

Chosen Open Data Source

NASA NEO Web Service (Open Data)
API Link: https://api.nasa.gov/neo/rest/v1/feed

Dataset Description:
NASA's NEO API provides information about asteroids that come close to Earth.
The API returns:

- Estimated diameter
- Velocity
- Miss distance
- Whether the asteroid is potentially hazardous
- Close approach date
- Orbiting body (usually Earth)

Problem to Be Solved

NASA provides Near-Earth Object (asteroid) data as raw JSON through an open API. This information is difficult to read and understand in its original form. Users cannot easily see which asteroids are hazardous, how big they are, or how close they come to Earth.

Project Goal

The goal of this project is to take NASA's open asteroid data, process it with Python, and turn it into clear, easy-to-understand information.

Expected Outcomes

- A clean table (CSV) of asteroid data
- Identification of hazardous, largest, and closest asteroids
- Simple charts showing size, danger level, and speed comparisons
- A small Python tool anyone can run to explore real NASA data

2. Project Architecture and Structure

Programming Language Justification (Python)

Python was chosen for this project because:

- It is excellent for data analysis (pandas, matplotlib)
- Very easy to work with APIs using requests
- Great support for visualization
- Clean, readable syntax that fits academic requirements
- Ideal for modular programming and OOP structure

Python makes data processing simple and efficient, which is perfect for working with an API dataset like NASA NEO.

Program Structure

The program is divided into four separate modules to follow clean, maintainable design:

1. data_fetcher.py

Responsible for:

- Sending requests to the NASA API
- Fetching raw JSON data
- Handling failed requests

2. data_processor.py

Processes and analyzes the data:

- Converts NASA JSON into a pandas DataFrame
- Extracts diameter, velocity, distance, hazard status
- Identifies:
    - **Top 10 largest asteroids**
    - **Closest asteroid to Earth**

3. visualization.py

Generates charts:

- Diameter distribution histogram
- Pie chart of hazardous vs non-hazardous
- Scatter plot comparing diameter vs velocity
- Saves all charts as PNG files

4. main.py

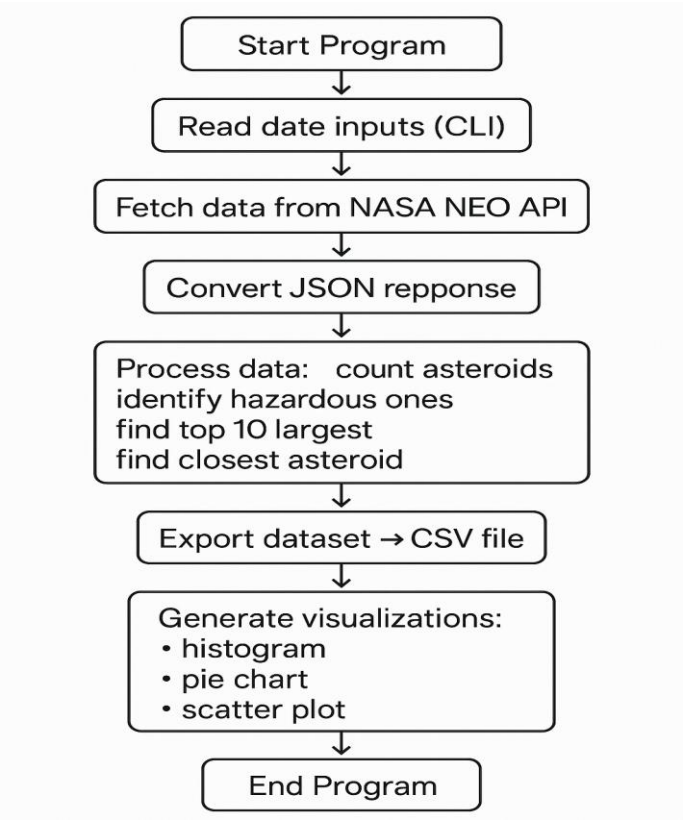
The central controller:

- Handles command-line arguments (--start and --end dates)
- Calls all modules in sequence
- Exports CSV file
- Prints summary results
- Generates graphs

List of Main Files

```
students/
└── Mohammed-Shameem/
    ├── Report/
    │   └── NasaDataReport.pdf
    ├── Results/
    │   ├── asteroid_data.csv
    │   ├── diameter_distribution.png
    │   ├── hazardous_pie.png
    │   └── velocity_vs_diameter.png
    ├── README.md
    ├── data_fetcher.py
    ├── data_processor.py
    ├── main.py
    ├── requirements.txt
    └── visualization.py
```

Algorithm / Flowchart



3. Result Examples

After running the program with:

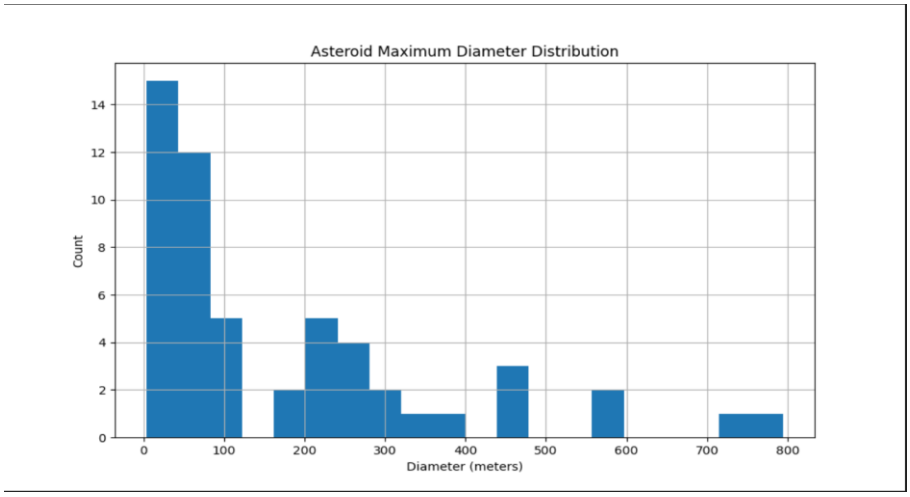
- python main.py --start 2024-01-01 --end 2024-01-03

```
PS C:\Users\kksh3\Desktop\Nasa> python main.py --start 2024-01-01 --end 2024-01-03
Fetching data from NASA...
Processing data...

Summary:
                name  is_hazardous  diameter_min_m  ...  miss_distance_km  close_approach_date  orbiting_b
ody
0  415949 (2001 XY10)         False      355.267088  ...      5.045241e+07           2024-01-02          Ea
rth
1         (2003 SR84)         False       16.770846  ...      1.979817e+07           2024-01-02          Ea
rth
2         (2005 YQ96)          True      199.781365  ...      2.498473e+07           2024-01-02          Ea
rth
3         (2009 HC21)         False      101.054342  ...      7.360980e+07           2024-01-02          Ea
rth
4         (2010 XA11)         False       16.016034  ...      3.527551e+07           2024-01-02          Ea
rth

[5 rows x 8 columns]

Number of asteroids: 54
Hazardous asteroids: 4

Exporting dataset to asteroid_data.csv...
CSV export complete.

Top 10 Largest Asteroids:
```

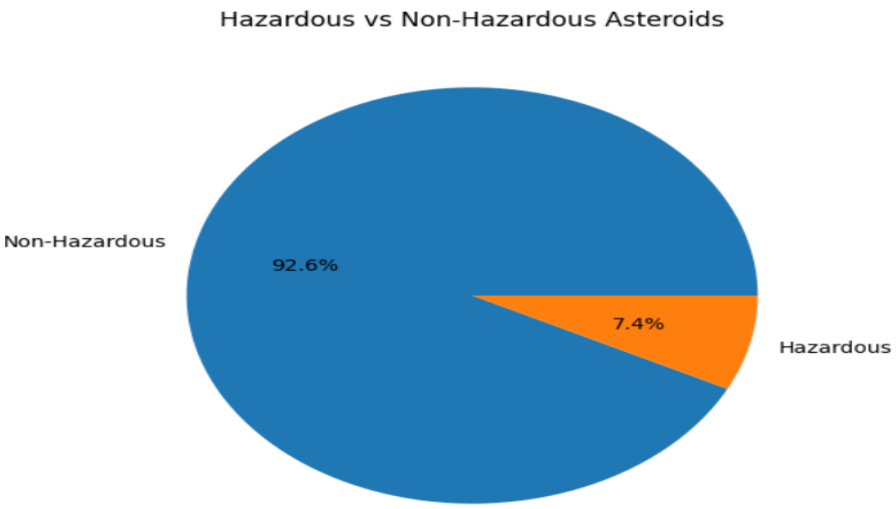
```
Top 10 Largest Asteroids:
                 name  is_hazardous  diameter_min_m  ...  miss_distance_km  close_approach_date  orbiting_
body
0   415949 (2001 XY10)         False      355.267088  ...      5.045241e+07           2024-01-02          E
arth
36         (2013 NT11)         False      336.166921  ...      5.802478e+07           2024-01-03          E
arth
6          (2012 SD22)          True      259.749653  ...      2.459731e+07           2024-01-02          E
arth
19  669051 (2012 SD22)          True      256.185754  ...      2.459731e+07           2024-01-02          E
arth
16          (2024 AA2)         False      207.278843  ...      2.879416e+07           2024-01-02          E
arth
2          (2005 YQ96)          True      199.781365  ...      2.498473e+07           2024-01-02          E
arth
9   613286 (2005 YQ96)          True      198.863453  ...      2.498480e+07           2024-01-02          E
arth
37        (2016 GZ215)         False      161.642283  ...      4.741643e+07           2024-01-03          E
arth
22         (2015 OD22)         False      152.249185  ...      6.541738e+07           2024-01-01          E
arth
52          (2024 BB8)         False      141.434375  ...      6.950695e+07           2024-01-03          E
arth

[10 rows x 8 columns]
```

```
[10 rows x 8 columns]

Closest Asteroid to Earth:
name                     (2024 AA)
is_hazardous                 False
diameter_min_m            8.761026
diameter_max_m           19.59025
velocity_km_s            21.432109
miss_distance_km       610571.843294
close_approach_date       2024-01-02
orbiting_body                 Earth
Name: 13, dtype: object

Generating visualizations (and saving PNG files)...
Charts saved as PNG files.
PS C:\Users\kksh3\Desktop\Nasa>
PS C:\Users\kksh3\Desktop\Nasa>
```

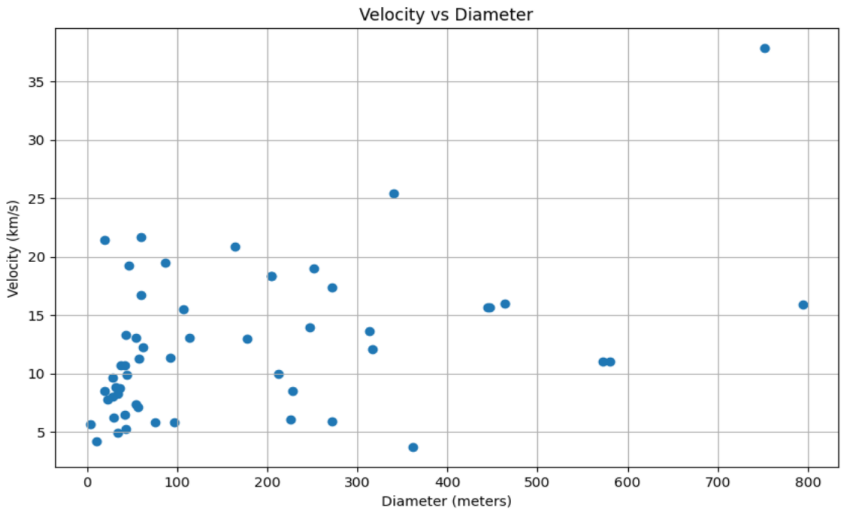
diameter_distribution.png



hazardous_pie.png



velocity_vs_diameter.png



Explanation of Results

- The dataset contained **54 asteroids** over the chosen date range.
- Only **4** were marked as potentially hazardous.
- The largest asteroid had a diameter of over **300 meters**.
- Visualizations help clearly see size distribution, hazard proportion, and speed/size relationship.
- Data export allows reuse in Excel or other analysis tools.

These results show how open data can be collected, structured, analyzed, and visualized to extract real scientific insights.

Program Code

The complete source code is available in the GitHub

GitHub Repository:
2025-Open-Data/students/Mohammed-Shameem at sh3meem-patch-1 · VUSA-Data-Lab/2025-Open-Data

Conclusion

In this project, I used open data from NASA to explore real information about asteroids that pass near Earth. With Python, I was able to download the data, clean it, analyze it, and create visual charts that make the results easy to understand. The program shows which asteroids are hazardous, which ones come closest, and how their sizes and speeds compare.

Overall, the project helped me practice working with APIs, processing data, and following good programming structure. Python was a very suitable choice because it made the data analysis simple and efficient. The final results show that even complex scientific data can be turned into clear information with the right tools.