Asteroid Dashboard Project - Documentation

# 1. Project Overview

This project is a data dashboard built using Streamlit that visualizes and analyzes Near-Earth Objects (asteroids) using data from NASA's NEO API. The dashboard allows users to filter, explore, and query asteroid information interactively. The project was developed as part of a GUVI assignment using NASA open data.

# 2. Key Features

- 📡 NASA API integration for live asteroid data.  
- 📊 Streamlit-based dashboard with wide layout.  
- 🔍 Filter asteroids by velocity, magnitude, size, and date.  
- ✅ 25+ predefined SQL queries.  
- 📈 Visualizations using Plotly charts.  
- 💾 Download filtered data as CSV.

# 3. Tech Stack Used

- Python 3  
- Streamlit  
- SQLite  
- Pandas  
- Plotly  
- NASA NEO API

# 4. Step-by-Step Explanation

\*\*Step 1: Fetching Asteroid Data\*\*  
- Used the NASA NEO API to fetch close-approach data for asteroids between specific dates.  
- Data collected includes velocity, distance, size, and hazard status.  
  
\*\*Step 2: Data Cleaning and Transformation\*\*  
- Parsed nested JSON safely and handled missing values.  
- Extracted key attributes like `relative\_velocity\_kmph`, `miss\_distance\_km`, `estimated\_diameter`, etc.  
  
\*\*Step 3: Storing in SQLite\*\*  
- Created two tables: `asteroids` and `close\_approach`.  
- Inserted cleaned records using parameterized SQL to avoid duplication.  
  
\*\*Step 4: Streamlit Dashboard\*\*  
- Built a 4-page layout: Home, Filter Criteria, Queries, and About.  
- Home: Displays overview and image.  
- Filter: Multiple sliders and date inputs to filter asteroid data.  
- Queries: 25 predefined queries from GUVI + personal queries.  
- About: Info about the app, data source, and credits.  
  
\*\*Step 5: Visualizations\*\*  
- Used Plotly to create histograms and bar charts.  
- User can select which column to visualize from query results.  
  
\*\*Step 6: Final Enhancements\*\*  
- Wide page layout using `st.set\_page\_config(layout="wide")`.  
- Image added to Home using `st.image()`.  
- Download CSV option for filtered results.

# 5. Sample SQL Query

-- Top 10 fastest asteroids  
SELECT neo\_reference\_id, AVG(relative\_velocity\_kmph) AS avg\_velocity  
FROM close\_approach  
GROUP BY neo\_reference\_id  
ORDER BY avg\_velocity DESC  
LIMIT 10;

# 6. Conclusion

This project demonstrates practical use of real-time API data processing, SQL analytics, and interactive UI creation. It highlights how open space data can be transformed into actionable insights using modern data tools.