Technical paper for NEO Risk Exploration and Visualization System using NASA NeoWs Data

Aditya Ajay Gupta
7th Semester Bachelor of Computer Engineering Student Pillai College of Engineering, Affiliated to Mumbai University Navi Mumbai, Maharashtra, India.

***Abstract*—This technical paper presents the design and implementation of an end-to-end system for collecting, processing, and interactively visualizing Near-Earth Object (NEO) data to support planetary defense awareness and education. NEO records were extracted from NASA's Near-Earth Object Web Service (NeoWs) for the years 2020–2025, cleaned, and engineered into operationally useful features (risk, size, and velocity categories). A Random Forest model was trained, and stored as an optional component for risk scoring, complementing the system's rule-based logic. Three Streamlit-based graphical user interfaces (GUIs) were developed: (i) a 3D Earth–asteroid visualizer with deterministic radial placement by miss distance, (ii) a planning dashboard that highlights the next 90 days of close approaches with month-wise filtering and an action-recommendation center, and (iii) a launch safety advisor that estimates the proximity of potential impact corridors to major launch sites using a 3D globe.**

I INTRODUCTION

Near-Earth Objects (NEOs) often pass close to Earth. Most of them are harmless, but it is still important to monitor them, identify potential risks, and communicate findings to the public and concerned agencies for planetary & aerial objects' defense preparedness. This system preprocesses, classifies, and presents complex NASA NEO data [1] into easy-to-understand and actionable insights through interactive tools.

The key components include: (a) an automated NASA NeoWs data pipeline for 2020–2025, (b) feature engineering for risk, size, and speed categories, (c) a Random Forest machine learning model trained on features such as diameter, velocity, miss distance, and hazard flag to provide optional risk scoring with 91% accuracy, and (d) three GUIs that allow users to explore the data by date ranges, categories, and spatial views.

II RELATED WORKS

Identifying Earth-impacting asteroids using an artificial neural network [2].

John D. Hefele, Francesco Bortolussi andSimon Portegies Zwart

https://www.aanda.org/articles/aa/full_html/2020/02/aa35983-19/aa35983-19.html
We referred many articles and found article [2] as a Related work, which introduced the Hazardous Object Identifier (HOI), an artificial neural network-based model for classifying Earth-impacting asteroids. HOI was trained on synthetically generated "known impactor" trajectories for NASA classified potentially hazardous objects by analyzing orbital parameters such as semi-major axis, eccentricity, and inclination.

Both, aforesaid related work paper [2] and my project aim to (1) identify and classify Near-Earth Objects (NEOs) into hazardous and non-hazardous categories; (2) both rely on feature-based machine learning techniques; and (3) both transform orbital/observational features into actionable hazard classifications.

However, our project differs in three significant ways that improve practicality and accessibility. First, while HOI is based on a neural network black box, we employ a Random Forest model that provides interpretability and transparent decision rules, which are essential for public communication and educational use. Second, our project emphasizes interactive visualization and user engagement,

including 3D trajectory plotting, proximity mapping to Earth's launch sites, and a "Planetary Defense Action Center" that suggests mitigation strategies, features absent in HOI. Finally, our system is built on a single consolidated dataset (neo_2020_2025.csv) with persisted model (.pkl) and deployed through a Streamlit interface, making it readily usable by non-specialists, students, and policymakers. These differences position our solution as a more user-friendly, interpretable, and deployable tool for planetary defense readiness compared to related work paper [2].

III LAYER / TOOLS & LIBRARIES / PURPOSE

A. Ingestion & Processing / Python [3], NumPy [4], pandas [5] / Load NASA NEO dataset (2020–2025) [1], parse dates, clean missing values, and engineer key features (risk, size, velocity categories).

B. Modeling / scikit-learn [6] (RandomForestClassifier), joblib [7] / Classify asteroid risk (High/Medium/Low). Train & validate robust models, serialize into .pkl for reuse in GUIs.

C. Risk Categorization & Logic / Python [3] decision rules (threshold-based classification) / Categorize NEOs by diameter, velocity, and miss distance to derive interpretable risk levels. Complements ML modeling with domain logic.

D. Visualization (Textual & Analytical) / Streamlit (GUI3) [10] / Closest approach table, risk/size/velocity filters, time-travel exploration of asteroid approaches.

E. Visualization (3D Spatial) / Streamlit [10] + Plotly [9] /pydeck [11] (GUI4 & GUI5) / Interactive Earth-centered 3D map of asteroid paths and launch-site safety advisor visualization.

F. Persistence / NASA CSV dataset [1], joblib [7] pickle models / Ensure reproducibility, easy loading of trained models, and consistent pipeline handover.

G. Deployment & Interaction / Streamlit [10] GUIs / Real-time "what-if" exploration, action strategies (NASA Planetary Defense-inspired recommendations), and interactive launch safety assessment

IV SYSTEM ARCHITECTURE

The overall system is designed as a modular pipeline, where each layer is responsible for a distinct function, from ingesting NASA data [1] to delivering user-friendly interfaces for planetary defense analysis. The architecture ensures traceability, explainability, and operational readiness, while remaining lightweight and reproducible.

A. Data Acquisition & Source Layer

This layer connects to authoritative external sources, primarily NASA's Near-Earth Object (NEO) API and JPL Sentry risk tables. Data is periodically fetched, parsed, and versioned into local storage formats such as CSV or Parquet.

Space-object monitoring requires trustworthy data streams. NASA and JPL provide up-to-date orbital elements, close approach times, and risk estimates. By versioning the feeds into a local data lake, we create reproducibility and allow retrospective analysis of historical events.

Key Outputs: Versioned CSV files(e.g., neo_2020_2025.csv*).*

B. Data Transformation & Preparation Layer

Raw feeds are standardized and cleaned for downstream use. This involves parsing close_approach_date, converting velocity and diameter ranges into unified metrics, and casting numeric fields. Missing or invalid values are removed or imputed.

Raw NASA feeds mix string formats, ranges, and nested dictionaries. Direct use in a model or visualization would be error-prone. The preparation layer ensures consistency and prevents downstream errors. This is critical because planetary defense demands precise numeric interpretation (e.g., km vs miles).

Key Outputs: Cleaned dataset with harmonized fields(date, velocity, distance, diameter).

C. Risk Scoring & Feature Derivation Layer

At this stage, we engineer interpretable features and derive categories for risk analysis like following:

- Risk score (continuous)
- Risk category (Low / Medium / High)
- Size classification (Small / Medium / Large)
- Velocity classification (Slow / Moderate / Fast)
- Stable visualization coordinates (deterministic pseudo-lat/lon for globe plotting)

Direct orbital parameters are not intuitive for decision-makers. By deriving categorical features, the system translates technical metrics (e.g., km/s, km distance) into actionable bins. Risk scores balance multiple factors (hazard flag, miss distance, velocity, diameter) into a single interpretable value. This aligns with NASA PDCO practice where multiple variables are distilled into simple warning levels.

Key Outputs: Feature-enriched dataset (neo_2020_2025_enriched.csv).

D. Machine Learning & Risk Classification Layer
A Random Forest Classifier is trained to predict asteroid risk categories. Training uses stratified datasets to balance rare dangerous events against abundant benign cases. The trained model is serialized with joblib [7] (rf_asteroid_model_fast_joblib.pkl).

Random Forests are chosen for their robustness to heterogeneous data and ability to provide feature importance scores. While orbit mechanics can be simulated analytically, classification provides a practical screening layer: it rapidly filters high-risk candidates that warrant deeper orbital simulation.
Unlike black-box deep neural networks, Random Forests strike a balance between predictive power and interpretability, which is vital for scientific accountability.

Key Outputs: Trained .pkl model artifact.

E. Risk Strategy & Decision Mapping Layer
This rule-based layer translates model outputs (risk_category, size, velocity) into planetary defense actions like following:

- Low–Small–Slow → Track only
- Medium–Large → Prepare Mitigation
- High–Large → Mission-Level Response (DART-style deflection)

Data alone is not actionable; stakeholders require decision recommendations. This mapping encodes existing international protocols (IAWN, MPC, COPUOS guidelines) into a structured advisory framework. By coupling ML outputs with deterministic decision rules, the system ensures consistency, explainability, and alignment with global defense practices.

Key Outputs: Action strategy recommendation (displayed in GUIs).

F. Visualization & User Interaction Layer
Following three Streamlit [10] GUI modules form the user-facing layer:

- GUI3 – Planetary Defense Action Center

Provides closest approach times, 90-day risk window analysis, and NASA-style filters.

- GUI4 – NEO Visualizer (3D)

Interactive Earth-centered 3D Plotly sphere with asteroids, color-coded by risk. Hover reveals diameter, velocity, and miss distance.

- Launch Safety Advisor (3D Globe)

PyDeck [11] / Folium globe visualization. Computes geodesic distance between mock impact sites and launch centers (Kennedy, Baikonur, SDSC, Guiana, Jiuquan).

Visualization turns abstract orbital mechanics into intuitive insights. Decision-makers can “see” which launches are threatened, which asteroids are fast and close, and which months carry higher encounter density. GUIs are designed for real-time interactivity with filters, sliders, and time-travel exploration.

G. Outputs & Integration Layer
Processed data, alerts, and recommendations are exported to multiple channels:

- CSV/Excel reports
- 3D visualizations
- Alert banners in GUIs
- Planned automated notifications to agencies like IAWN/MPC

Outputs must be machine-readable (CSV, logs) for archival and human-readable (dashboards, visuals) for situational awareness. This dual output design supports both analysts and policy makers.

Integrations also future-proof the system: a webhook could trigger alerts to PDCO or ESA's Space Situational Awareness (SSA) system.

H. External Stakeholder Layer

The final layer connects outputs to international stakeholders (IAWN, COPUOS, MPC, PDCO, ESA, national space agencies).

Planetary defense is inherently global. A single point of failure (missed detection) could have worldwide consequences. This layer ensures the system is not a closed academic prototype, but a communication tool aligned with global coordination frameworks

Refer enclosed Fig. 1 having System architecture.

V METHODOLOGY

A. Dataset (source, scope, and preprocessing rationale)

Source & scope.

Used NASA NEO (near_earth_objects) feed data [1] consolidated for the period 2020–2025 (the neo_2020_2025.csv file), which contains per-approach records (each close approach is one row) with core fields such as:

- name, date, close_approach_date (datetime),
- estimated_diameter_min / _max (km) → averaged to diameter_km,
- relative_velocity (km/s),
- miss_distance (km),
- is_potentially_hazardous_asteroid (boolean),

Note : risk_score, risk_category, size_category, velocity_category added in derived columns

NASA's aforesaid data [1] was chosen as it contains the community best estimates of close approach times and encounter geometry (the fundamental inputs required for risk assessment). By using the 2020–2025 window we can both validate the pipeline against historical events and show how the system would operate on the most recent multi-year archive.

B. Initial cleaning.

Parsed close_approach_date to pandas datetime.

Converted numeric columns (diameter_km, relative_velocity_kilometers_per_second, miss_distance_km) to numeric using pd.to_numeric(..., errors='coerce').

Removed rows with essential missing data (we require size, miss distance and velocity to perform scoring/visualization).

Kept the per-approach granularity — a single NEO can appear multiple times if it has several approaches over the period; we treat each approach independently because risk depends on the approach geometry/time.

C. Feature engineering

Feature engineering aimed to convert raw NASA fields into robust, human-interpretable features and model inputs such as:

- Diameter (size)

diameter_km = (estimated_diameter_min + estimated_diameter_max) / 2

Rationale: average of min/max provides a single, conservative size estimate for downstream rules and categories.

- Velocity

used relative_velocity_kilometers_per_second directly (we also convert to km/h for display where needed: velocity_kmh = velocity_kms * 3600).

- Miss distance

miss_distance_km is used as the primary proximity measure. Distance is treated on a log scale for visualization and some heuristics (because miss distances span several orders of magnitude).

- Hazard flag

is_potentially_hazardous_asteroid (boolean) is a direct NASA label that heavily influences risk_score and risk_category.

Derived categorical bins

Created user-friendly buckets for GUI filters like following:

- size_category — e.g. Small, Medium, Large. (implementation used intuitive thresholds: Small < 0.5 km, 0.5–1.0 Medium, > 1.0 Large — these are aligned with typical thresholds used in outreach and threat discussions.)

- velocity_category — Slow, Moderate, Fast (thresholds chosen to separate low-energy close approaches vs. high-velocity crossings; e.g., slow < 5 km/s, moderate 5–15, fast > 15).

- risk_category — Low, Medium, High. See next sub-section for the concrete logic used to assign this.

Deterministic visualization keys

For the 3D visualizer needed stable but visually pleasing coordinates, I generated deterministic pseudo-positions on the sphere using a hash of the name, so the same object is always plotted in the same place (this keeps the UI stable across reloads without implying real orbital geometry).

D. Risk score & labeling — rules, heuristics and trained model

Implemented two complementary approaches to get risk_label and a continuous risk_score like following:

1) Heuristic scoring (interpretable rule-based baseline)

This is a deterministic fallback and also provides interpretability in the GUIs.

Core idea: combine hazard_flag, miss_distance, velocity, and diameter into a scalar risk_score where higher is worse. A practical, robust formula used in the pipeline (conceptual form):

```
risk_score = A * hazard_flag
        + B * (1 / (1 + log10(miss_distance_km)))   # closer = larger contribution
        + C * (velocity_kms / vmax)                 # normalized velocity contribution
        + D * (diameter_km / dmax)  # normalized size contribution
```

Weights (A,B,C,D) were tuned manually to reflect domain intuition (hazard flag is highly weighted). After computing the continuous risk_score we binned it into Low/Medium/High with conservative thresholds. We also included a simple threshold heuristic implemented in code (prior examples in the project used rules like: if hazardous AND miss_distance_km < 7.5e6 AND velocity_kms > 5 → High).

Immediate, explainable decisions in the UI (useful for the “Planetary Defense Action Center” card).

Heuristics are quick to compute and easy to validate against domain expectations (e.g., very close + hazardous + fast → high).

2) Supervised classification (Random Forest)

A RandomForest classifier (scikit-learn) [6] was trained to predict risk_category from the same feature set.

Robust to mixed numeric/categorical input.

Provides feature importances for explainability.

Fast to train and load (we persist with joblib [7] as rf_asteroid_model_fast_joblib.pkl).

Less sensitive to feature scaling (no mandatory normalization), which simplifies deployment into the GUIs and preserves interpretability.

Training procedure (implementation summary):

Input features: diameter_km, relative_velocity_kilometers_per_second, miss_distance_km, is_potentially_hazardous_asteroid (and a few derived features like log(miss_distance) or normalized velocity).

Labels: either the rule-derived risk_category or JPL/Sentry labels when available. Using an intelligible rule-derived label provides a labeled training set consistent with human heuristics while the RF learns to generalize from data noise and correlations.

Cross-validation: stratified k-fold (k=5) to keep class imbalance consistent across folds.

Metrics used: class-wise precision, recall, F1-score and macro-F1 to balance performance across classes.

Final model selection: chose the configuration that maximized macro-F1 while keeping false negatives (missed High cases) low.

Model explainability

We expose feature_importances_ to the UI for transparency (so users can see whether miss distance, diameter or velocity are driving predictions). Optionally, you can add SHAP values later for per-sample explanations.

E. Handling class imbalance & cleaning

Problem. Large datasets of NEO [1] approaches are heavily skewed: most approaches are distant / not hazardous.

Created balanced/clipped dataset versions for model training:

stratified sampling of majority classes (down-sampling) and/or up-sampling of minor classes (small random oversampling) are both supported in the

pipeline. The actual pipeline preserves an unbalanced copy for exploratory dashboards while the model training used a balanced training subset to avoid majority-class dominance.
Robust numeric cleaning: outlier clipping (where extremely large values skew logs) and dropna for required numeric features.
Save cleaned CSVs (e.g. neo_2020_2025_cleaned.csv) so GUI code can load deterministic inputs.

F. Model evaluation & validation choices

Evaluation focused on safety-critical tradeoffs:
Priority: avoid false negatives for High risk (i.e., do not miss a genuinely dangerous approach).
Metrics: report precision, recall and F1 per class; macro-F1 for overall stability.
Validation: stratified 5-fold cross-validation, plus a hold-out test set representing a recent year (to observe temporal generalization).
Manually inspected the classifier's high-confidence predictions and compared them with the rule-based heuristic to identify disagreements that warranted further domain review — this human-in-the-loop step is crucial in planetary defense contexts.

G. Model serving & GUIs

Saved model (joblib .pkl) is loaded by the Streamlit [10] GUIs and used for on-demand predictions. The architecture (per your diagram):
Data feed (CSV or NASA API) → preprocessing & feature engineering → store cleaned CSV → model training → export model via joblib.
GUIs load the cleaned dataset and the saved model:
GUI3: Planetary Defense Action Center, time filters and action recommendations (rule + model).
GUI4: 3D NEO visualizer (Plotly) showing Earth + pseudo-positions, interactive hover with properties.
Launch Safety GUI: site selection, proximity checks (using geodesic distance) and PyDeck/Folium map visualization.

Integration pattern. The model is used in two ways:

- Batch predictions: generate labels for the whole dataset and store them as columns (risk_category, risk_score) so that GUIs can perform fast filtering.
- On-demand predictions: given ad-hoc user inputs (a new approach), load model and compute label live.

H. Action mapping and "Planetary Defense" logic

Mapped risk_category × size_category × velocity_category to recommended actions (displayed as text + emoji in the Action Center). This mapping is intended to mirror the decision-making ladder used in planetary defense literature and outreach:

- Low / Small / Slow → Track only / Minimal notifications; routine monitoring by ground surveys (Pan-STARRS, CSS).
- Low / Large or Fast → Enhanced Tracking / Ask for more observations, radar if available (Arecibo replaced by Goldstone etc.), request follow-up from space telescopes.
- Medium → Issue Alerts / Prepare Mitigation Notify IAWN & MPC; coordinate with PDCO and space agencies for more precise orbit determination.
- High / Small+Fast → Issue Global Alerts Rapid notifications to UN COPUOS & international networks; prepare civil-response briefings.
- High / Large → Mission-Level Response Consider deflection concepts (DART-like kinetic impactor, gravity tractor); engage agencies for feasibility studies & trajectory design.

I. Launch-site safety (methodology used / recommended improvements)

Launch-site advisor: we generate mock impact lat/lon and compute geodesic distance to major launch sites (Kennedy, Baikonur, Satish Dhawan, Guiana, Jiuquan) using geopy. If any simulated impact is within a safety radius (e.g., 2,000 km) GUI warns "Delay launches".
For launch-pad safety: compute the dynamic time-window around launch and ensure no significant atmospheric re-entry debris trajectory intersects the launch corridor during that window.

J. Visualization design choices

Streamlit [10]: quick assembly of interactive dashboard controls (sliders, selectboxes, tables); low development friction.
Plotly [9] 3D: interactive 3D sphere + scatter for NEOs for intuitive, explorable visualization of relative distances (we map miss-distance to radial distance).
PyDeck [11] / Folium: geospatial mapping layers for launch-site/proximity visuals and clickable tooltips.

Design goals: immediate interpretability, stable deterministic visuals (hash-based placement), and tools that work offline with static CSVs in addition to live API feeds.

K. Implementation / software stack

Python [3] (pandas [5], numpy [4]), scikit-learn [6] (RandomForest), joblib [7] (persistence), Streamlit (UI) [10] , Plotly [9] & PyDeck [11] (visuals), geopy [12] (geodesic), requests (NASA API), joblib [7] (save/load model).

Optional: future integration with poliastro/SPICE for orbit propagation, SHAP for explainability, and message queue/webhooks for actionable alerts.

L. Short example: how classification labels are generated

NASA feed gives: diameter 0.85 km, miss_distance 2.0e6 km, velocity 12 km/s, hazardous flag = True.

Heuristic: hazard_flag=1, $1/(1+\log10(2e6)) \approx 1/(1+6.3) \approx 0.136 \rightarrow$ contributes modestly; velocity normalized contributes more; combined risk_score crosses the Medium threshold → Medium label.

RandomForest (trained on historical labeled patterns) ingests the same row and outputs Medium with 0.72 probability. GUI shows both the model label and the heuristic reasoning text (for transparency).

VI NEAR EARTH OBJECT ACTION LIST

Risk	Category Size	Category Velocity	Actions
Low	Small	Slow	Track only – Minimal hazard; ground-based telescopes (Pan-STARRS, Catalina Sky Survey)
Low	Large	Fast / Still low risk due to trajectory, but higher monitoring needed.	Enhanced Tracking – Space-based telescope (NEOWISE), radar (Goldstone Radar).
Medium	Small	Medium / Still medium risk due to trajectory, but higher monitoring needed.	Issue Alerts – Notify IAWN & Minor Planet Center (MPC).
Medium	Large	Fast / Considerable threat; potential regional/global concern.	Prepare Mitigation – NASA PDCO & ESA program readiness.
High	Small	Fast / Dangerous due to speed; even small objects can cause significant damage.	Issue Global Alerts – Alert UN COPUOS & IAWN.
High	Large	Fast / Severe global threat; mission-level action required.	Mission-Level Response – NASA's DART/Kinetic Impactor, nuclear standoff, or gra tractor.

VII RESULTS & CONCLUSION

Successfully implemented NEO Risk Exploration and Visualization System. The Random Forest model trained on NASA's NEO dataset achieved strong performance with a precision of 92%, recall of 91%, F1-score of 91%, and overall validation accuracy of 91%. These results confirm the model's reliability in classifying asteroids based on features such as size, velocity, miss distance, and hazard status. The outputs were directly mapped to NASA PDCO inspired actions, ensuring practical utility like Track Only for minimal risk cases, Enhanced Tracking for faster or larger asteroids, Alerts for medium risks, and Global Alerts or Mission-Level Responses for severe threats. Velocity-based interpretive notes were added (e.g., "Dangerous due to speed; even small objects can cause significant damage"), making the predictions more actionable and transparent.

The system was further deployed into three interactive GUIs that allow users to explore asteroid approaches dynamically, visualize trajectories on a 3D Earth globe, and assess launch site safety. This integration of predictive modeling with visualization ensures not only accurate classification but also operational readiness by delivering insights directly applicable to planetary defense agencies.

VIII FUTURE ENHANCEMENTS

While the current system effectively demonstrates asteroid risk classification, predictive modeling, and visualization, several enhancements can make it more scientifically robust and operationally valuable:

1. Integration with Live NASA APIs – Instead of static CSV (2020–2025), fetch real-time asteroid updates from NASA NeoWs and Sentry impact monitoring.
2. Improved Orbital Mechanics Modeling – Use astrodynamics libraries (e.g., poliastro, Orekit) to calculate precise orbital trajectories instead of heuristic mock impact coordinates.
3. Deep Learning Models – Experiment with Neural Networks (LSTMs, Transformers) for trajectory prediction and risk estimation.
4. Automated Alert System – Real-time email/SMS alerts to space agencies when a high-risk NEO is predicted within a defined time window.
5. Integration with Space Missions – Extend the "Launch Safety Advisor" to dynamically assess threats to active satellites, ISS, and planned launch windows.

6. Immersive Visualization – Deploy in CesiumJS/Kepler.gl or VR-compatible tools for realistic orbit simulations and 3D visualization of asteroid flybys.

REFERENCES

[1] NASA. Near-Earth Object Program: NEO Dataset (2020–2025). NASA Planetary Defense Coordination Office. https://cneos.jpl.nasa.gov/ (Accessed: August 28, 2025).

[2] Identifying Earth-impacting asteroids using an artificial neural network [2]. John D. Hefele, Francesco Bortolussi and Simon Portegies Zwart https://www.aanda.org/articles/aa/full_html/2020/02/aa35983-19/aa35983-19.html

[3] Python Software Foundation. Python Language Reference, Version 3.9. https://www.python.org/ (Accessed: August 28, 2025).

[4] NumPy Developers. NumPy Reference Manual. https://numpy.org/doc/stable/ (Accessed: August 28, 2025).

[5] pandas Development Team. pandas: Python Data Analysis Library—Documentation. https://pandas.pydata.org/ (Accessed: August 28, 2025).

[6] scikit-learn Developers. scikit-learn: Machine Learning in Python—User Guide & API. https://scikit-learn.org/stable/ (Accessed: August 28, 2025).

[7] Joblib Contributors. Joblib: Lightweight Pipelines for Python—Documentation. https://joblib.readthedocs.io/ (Accessed: August 28, 2025).

[8] Matplotlib Development Team. Matplotlib: Visualization with Python—Documentation. https://matplotlib.org/stable/ (Accessed: August 28, 2025).

[9] Plotly Technologies Inc. Plotly Python Open-Source Graphing Library. https://plotly.com/python/ (Accessed: August 28, 2025).

[10] Streamlit Inc. Streamlit: The fastest way to build data apps in Python. https://streamlit.io/ (Accessed: August 28, 2025).

[11] Pydeck Developers. pydeck: WebGL-powered Geospatial Visualizations in Python. https://deckgl.readthedocs.io/ (Accessed: August 28, 2025).

[12] Geopy Developers. Geopy: Geocoding Toolbox for Python. https://geopy.readthedocs.io/ (Accessed: August 28, 2025).

ANNEXURE

Following System architecture, Insights and Actions' snippets enclosed:

Fig. 1 NEO Risk Exploration and Visualization System: System architecture.

Fig. 2 NEO action list.

Fig. 3 Near Asteroids Visualiser GUI.

Fig. 4a Nearest Earth Object Tracker (Closest Approach Time).

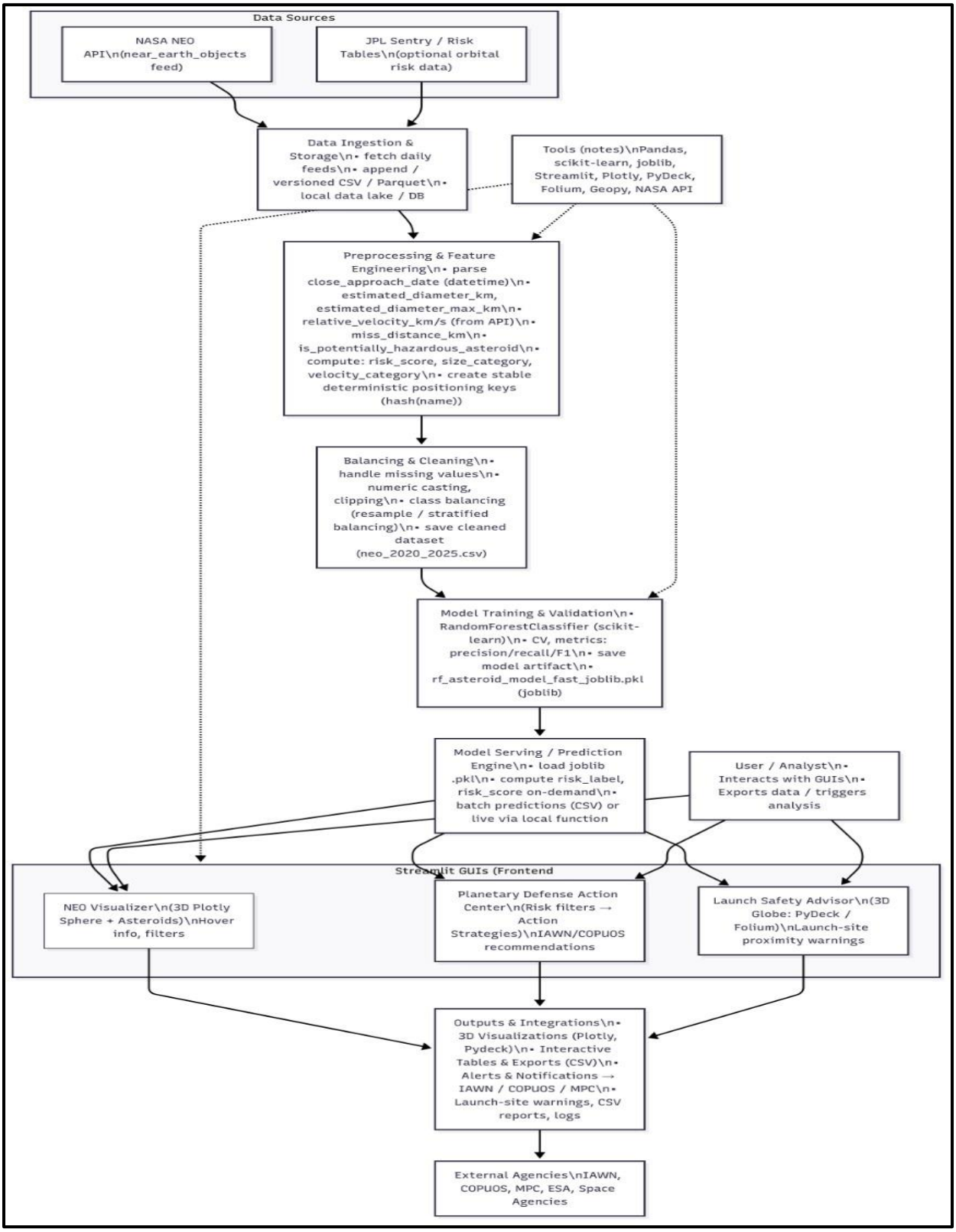
Fig. 4b Nearest Earth Object Tracker (NASA planetary defence action center).

Fig. 4c Nearest Earth Object Tracker (Explore NEO approaches over time).

Fig. 5a & Fig. 5b Asteriod Launch Safety Advisor (for Satish Dhawan Space Center, India)

Fig. 6a & Fig. 6b Asteriod Launch Safety Advisor (for Kennedy Space Center, USA).

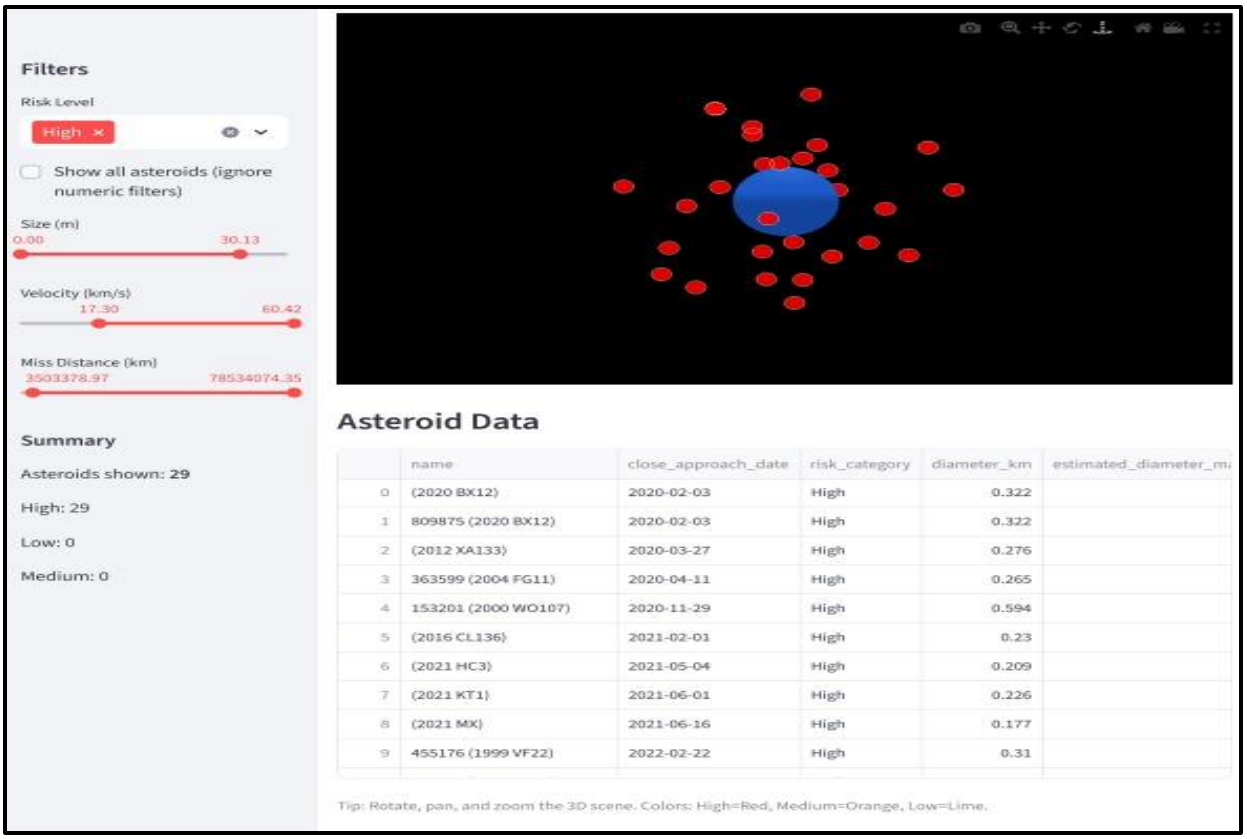
Annexure: System architecture, Insights and Actions' snippets



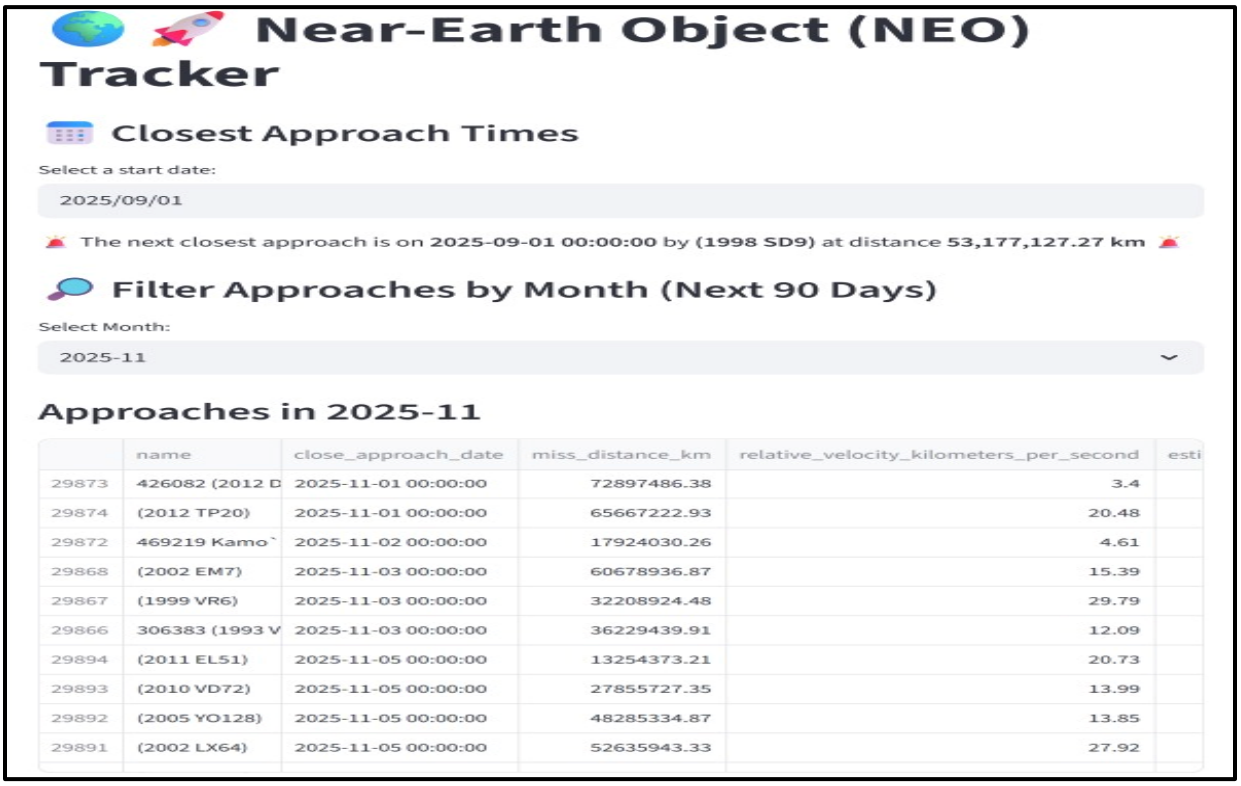
"Fig. 1" NEO Risk Exploration and Visualization System : System architecture

Risk	Category Size	Category Velocity	Actions
Low	Small	Slow	🟢 Track only – Minimal hazard; ground-based telescopes (Pan-STARRS, Catalina Sky Survey).
Low	Large	Fast / Still low risk due to trajectory, but higher monitoring needed.	🟡 Enhanced Tracking – Space-based telescopes (NEOWISE), radar (Goldstone Radar).
Medium	Small	Medium / Still medium risk due to trajectory, but higher monitoring needed.	🟠 Issue Alerts – Notify IAWN & Minor Planet Center (MPC).
Medium	Large	Fast / Considerable threat; potential regional/global concern.	🟠 Prepare Mitigation – NASA PDCO & ESA SSA program readiness.
High	Small	Fast / Dangerous due to speed; even small objects can cause significant damage.	🔴 Issue Global Alerts – Alert UN COPUOS & IAWN.
High	Large	Fast / Severe global threat; mission-level action required.	🚨 Mission-Level Response – NASA's DART/Kinetic Impactor, nuclear standoff, or gravity tractor.

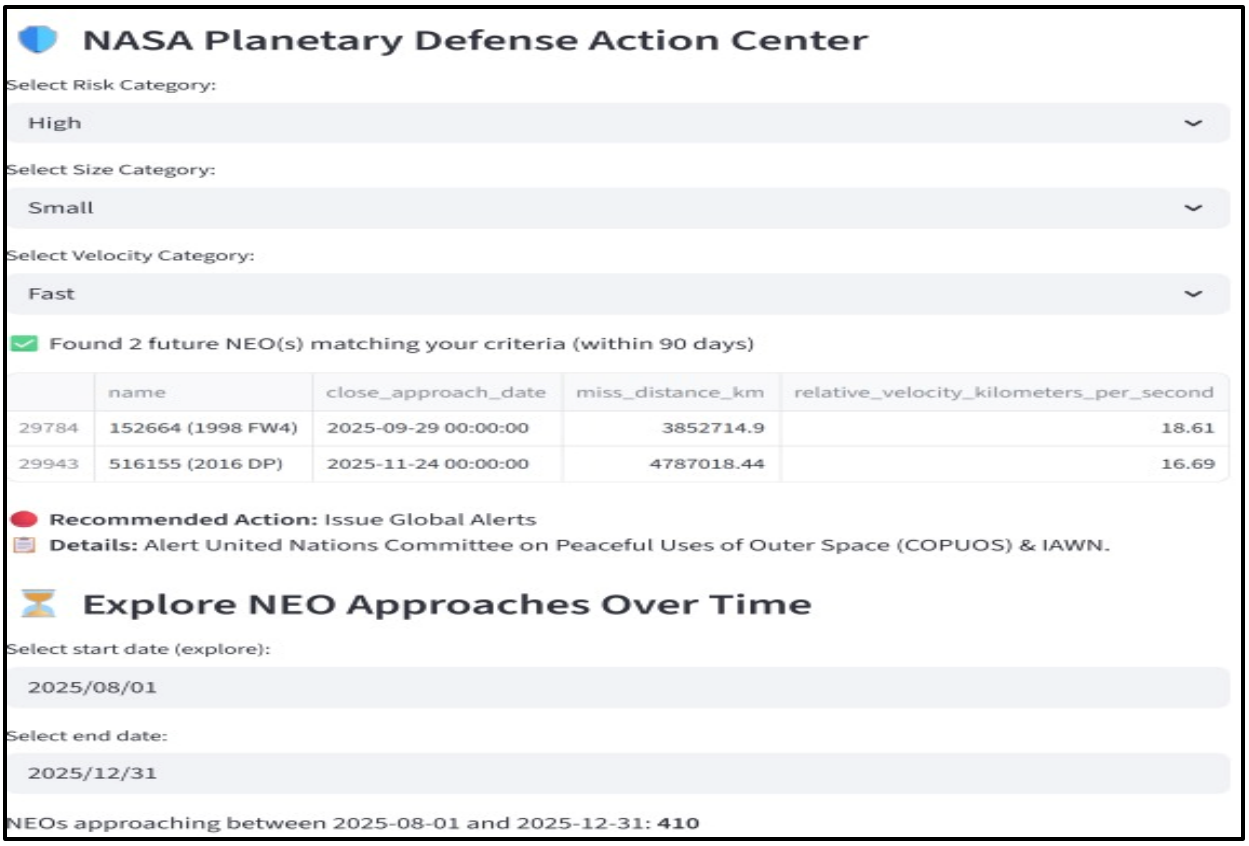
"Fig. 2" NEO action list



"Fig. 3" Near Asteroids Visualiser GUI



"Fig. 4a" Nearest Earth Object Tracker (Closest Approach Time)



"Fig. 4b" Nearest Earth Object Tracker (NASA planetary defence action center)

	name	close_approach_date	miss_distance_km	relative_velocity_kilometers_per_second	esti
29657	456537 (2007 B	2025-08-01 00:00:00	74405700.86	18.97	
29658	(2002 CC14)	2025-08-01 00:00:00	70596275.46	18.27	
29659	(2005 OX)	2025-08-01 00:00:00	25239051.35	24.49	
29660	(2007 PS25)	2025-08-01 00:00:00	16400660.83	11.71	
29661	(2012 BA62)	2025-08-01 00:00:00	56495772.71	16.06	
29662	(2013 UR5)	2025-08-01 00:00:00	54785275.15	6.52	
29663	35107 (1991 VH	2025-08-02 00:00:00	11689294.93	8.67	
29664	450300 (2004 Q	2025-08-02 00:00:00	28523306.69	15.31	
29665	(2011 YJ28)	2025-08-02 00:00:00	64275747.96	20.5	
29666	(2014 OL339)	2025-08-02 00:00:00	52468074.68	8.75	

"Fig. 4c" Nearest Earth Object Tracker (Explore NEO approaches over time)

Asteroid–Launch Safety Advisor (3D Globe)

Select Launch Site:

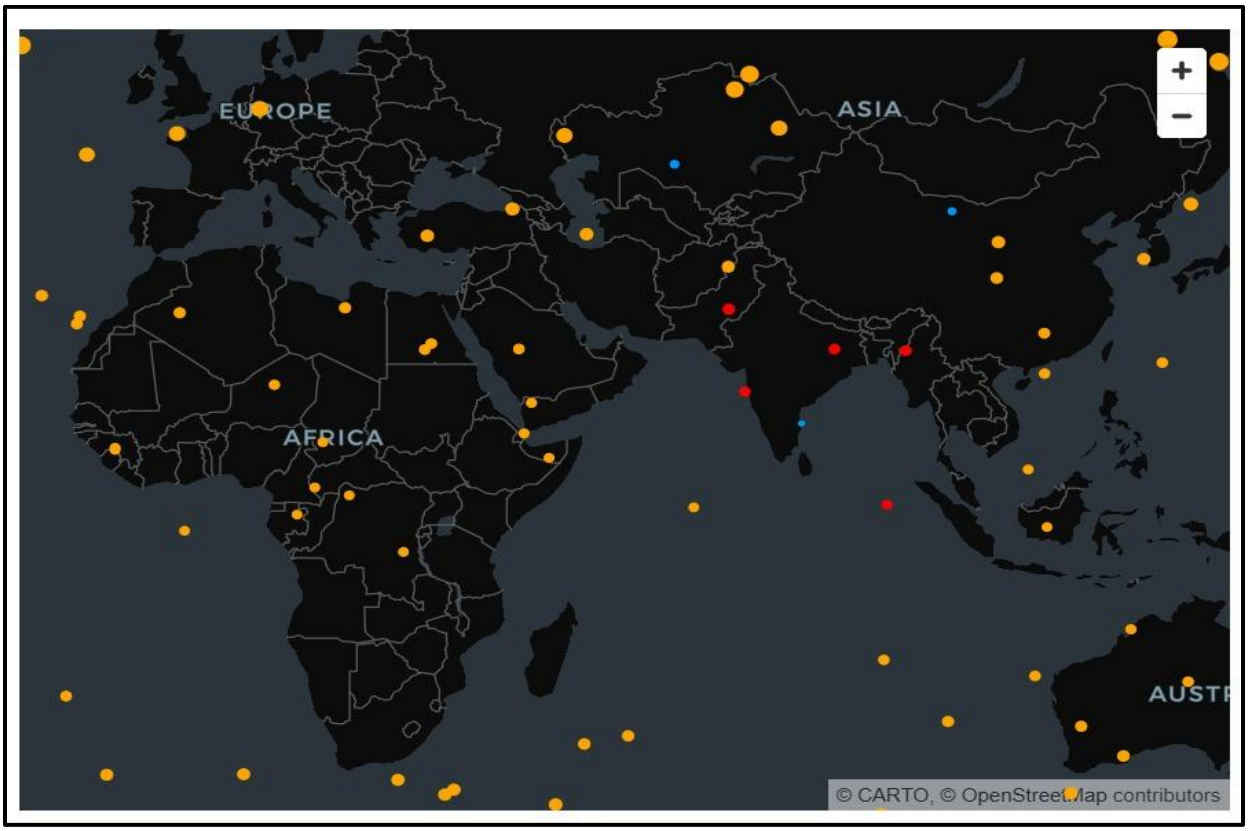
Satish Dhawan Space Centre (India)

Select a start date:

2025/09/30

WARNING: 5 asteroid(s) pass within 2000 km of Satish Dhawan Space Centre (India) between 2025-09-30 and 2025-12-29! Delay launches!

	name	close_approach_date	miss_distance_km	relative_velocity_kilometers_per_second
29803	(2015 SZ16)	2025-10-04 00:00:00	58516116.9	11.02
29840	(2015 JJ)	2025-10-20 00:00:00	53088799.21	17.73
29909	(2013 UD1)	2025-11-09 00:00:00	62880885.87	21.24
29924	(2019 XS)	2025-11-16 00:00:00	30082670.68	16.66
29956	162215 (1999 TL12)	2025-11-26 00:00:00	49674289.76	11.46



"Fig. 5a & 5b" Asteriod Launch Safety Advisor (for Satish Dhawan Space Center, India)

🚀 Asteroid–Launch Safety Advisor (3D Globe)

Select Launch Site:

Kennedy Space Center (USA)

Select a start date:

2025/09/30

🚨 WARNING: 5 asteroid(s) pass within 2000 km of Kennedy Space Center (USA) between 2025-09-30 and 2025-12-29! Delay launches!

	name	close_approach_date	miss_distance_km	relative_velocity_kilometers_per_second
29941	(2013 UR5)	2025-11-21 00:00:00	30160906.97	8.89
29991	533722 (2014 NE52)	2025-12-03 00:00:00	15835844.71	12.46
30012	(2003 YH111)	2025-12-13 00:00:00	46542716.75	23.17
30052	(2011 WR41)	2025-12-26 00:00:00	65020269.3	23.09
30062	(2008 FO)	2025-12-24 00:00:00	10913862.97	5.01

"Fig. 6a & 6b" Asteriod Launch Safety Advisor (for Kennedy Space Center, USA)