Astronomy APIs

1) NASA Open API (https://api.nasa.gov/)
   NASA Open API contains several APIs, which provide multiple kinds of data. One prominent one is the Exoplanet Archive API, which contains data on exoplanets, including host star, orbital period, and discovery method. Some more are the Mars Trek WMTS API and the Insight API, which show geographical and meteorological data respectively for Mars. They could be combined to create a simulation of the conditions on Mars for a service similar to Google Earth. I chose this API because of the vast variety of options given for data.
2) IP Geolocation Astronomy API (https://ipgeolocation.io/documentation/astronomy-api.html)
   IP Geolocation Astronomy API mainly provides data on sunrise, sunset, and moon and sun locations for a given latitude and longitude. The API can also be called using an IP address, so a possible application could be simulating the position of the sun and moon for the user's current location. Simulation can also be accomplished for different times, allowing for the position of the moon and sun to be tracked across the sky. This can also be combined with the IP Geolocation API in order to detect a user's location and generate content in different languages as a result. I chose this API because it could be used to simulate sunrise and sunset due to the given geometries and to create a virtual sunrise/sunset.
3) Where the ISS at? API (https://wheretheiss.at/w/developer)
   Where the ISS at? API provides data on satellites the ISS is connected to and the location of the ISS. I chose this API because it could be used to simulate ISS transit and make an interactive ISS simulation on Liquid Galaxy.
4) Mooncalc/Suncalc API (https://www.torsten-hoffmann.de/apis/suncalcmooncalc/link_en.html)
   Mooncalc/Suncalc API can be used to monitor the conditions on the moon.sun. For example, the phase and elapsed time in the lunar cycle can be retrieved. Moonset, sunset, moonrise, and sunrise times can also be retrieved. This can also be used to simulate the path of the moon in Liquid Galaxy. The times of lunar and solar eclipses can also be called, so it would be helpful to eclipse chasers or for creating a calendar of astronomical events. I chose this API because of its usefulness in creating a app that functions like a calendar of astronomical events and for simulation the passage of the moon and sun across the sky.

Airspace APIs

1) OpenSky Network API (https://opensky-network.org/apidoc/)
   OpenSky Network API contains a massive database that tracks hundreds of individual flights across the globe. It can be used to create flight maps, analyze aviation traffic, and analyze the effectiveness of possible new routes, which is useful in the airline industry. It also contains data on several airports, including airlines operating there, cities, and timezones. Tracking on individual flights is very extensive, including location, speed, and estimated time of arrival when in the air, and status, location, and take-off information

when at the airport. I chose this API because of its potential applications in creating a more user-friendly airport experience and because of its availability.

2) Aeronautical Data and Products API (https://www.faa.gov/got_data/aero_data/)
The Aeronautical Data and Products API contains data more closely aligned with aeronautics, and stores information on optimal routes for several locations in the United States. This can help to optimize flight time, as well as allow airports to plan out routes to accommodate the maximum number of people. Other aviation-related non-commercial information is included, such as the schedules of helicopters along the coast and data on airplane manufacturers and models. I chose this API because of its scientifically rigorous data and its usefulness in simulating and teaching about aerodynamics.
3) Airport API (https://www.developer.aero/Airport-API/API-Overview)
Airport API focuses more on individual airports rather than flights. It can retrieve a vast amount of information about major airports, including its full name, country, city, location, and time zone. This simple functionality is further extended to find the shortest path between airports, which helps to optimize flight paths, find a list of airports matching a specified code, or find airports nearest a location, which is useful for potential passengers to choose the nearest airport. I chose this API because of its simple design and its relevance in calculating optimal flight paths.