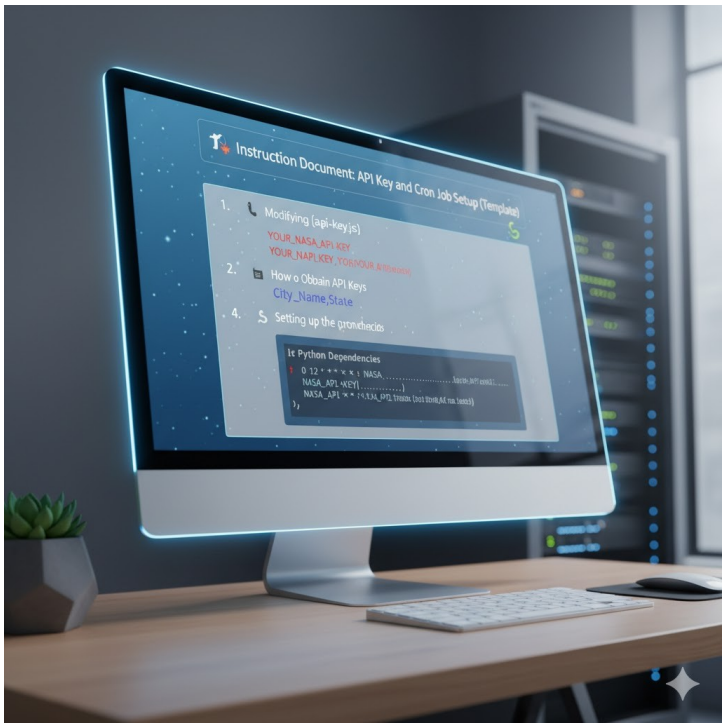
🛠️ Instruction Document: API Key and Cron Job Setup (Template) 🐍

This document provides instructions for modifying the `js/api-key.js` file, setting up the required Python environment, and configuring a cron job to schedule the `fetch_nasa_data.py` script. **All current values are placeholders and MUST be replaced with your actual service keys and location data.**



---

1. 🔑 Modifying `js/api-key.js` File

You must replace the placeholder values for all API keys, locations, and coordinates with your actual, live data.

Required Changes in `js/api-key.js`

Constant	Description	Current Placeholder Value	Action: **Replace with Your Real Value**
`GLOBAL_API_KEY`	Visual Crossing Weather API Key	`YOUR_VISUAL_CROSSING_API_KEY`	Your **Visual Crossing** API Key.
`GLOBAL_LOCATION`	Location for Visual Crossing	`City_Name,State`	Your target **Location** (e.g., "Dallas,Texas").
`OWM_API_KEY`	OpenWeatherMap Current Weather API Key	`YOUR_OWM_CURRENT_API_KEY`	Your **OpenWeatherMap** Current API Key.
`OWM_LAT`	OpenWeatherMap Latitude	`0.0000`	The **Latitude** for your location.
`OWM_LON`	OpenWeatherMap Longitude	`0.0000`	The **Longitude** for your location.
`FCAST_API_KEY`	OpenWeatherMap Forecast API Key	`YOUR_OWM_FORECAST_API_KEY`	Your **OpenWeatherMap** Forecast API Key.
`FCAST_LAT`	Forecast Latitude	`0.0000`	The **Latitude** for

Constant	Description	Current Placeholder Value	Action: **Replace with Your Real Value**
			your location.
`FCAST_LON`	Forecast Longitude	`0.0000`	The **Longitude** for your location.

---

2. 📝 How to Obtain API Keys

The following keys are required for the application to function:

A. Visual Crossing API Key

- **Purpose:** Used for the `GLOBAL_API_KEY` and to provide weather data for the primary application logic.
- **Action:** You need to register for an account and generate an API key on the **Visual Crossing** website.

B. OpenWeatherMap (OWM) API Keys

- **Purpose:** Two keys (`OWM_API_KEY` and `FCAST_API_KEY`) are being used for current weather and forecast data.
- **Action:** You need to register on the **OpenWeatherMap** website. Upon registration, a default API key is often generated. You may need to generate a second key within your account dashboard.

C. NASA API Key

- **Purpose:** Used by the `fetch_nasa_data.py` script to pull data from the DONKI API (Coronal Mass Ejection, Solar Flare, and Geomagnetic Storm data).
- **Action:** Obtain your API key by registering for an account with the specific **NASA** service the script is connecting to.

---

3. 🐍 Python Dependencies and Environment Setup

The `fetch_nasa_data.py` script requires a Python 3 environment and one external library to function correctly.

A. Prerequisites

1. **Python 3:** Ensure Python 3 (specifically the `python3` executable) is installed on your server.
2. **`pip`:** The Python package installer (`pip` or `pip3`) is needed to install external libraries.

B. Installing Dependencies

The script uses the `requests` library for making HTTP calls to the NASA DONKI API.

1. **Install `requests`:** Execute the following command on your server:

   Bash

   ```
   pip3 install requests
   ```

   Note: If you encounter a permission error, you may need to use `sudo` or install to a virtual environment.

C. Execution Environment

The cron job relies on:

- The correct path to the Python 3 interpreter (e.g., `/usr/bin/python3`).
- The `NASA_API_KEY` being passed as an **environment variable** in the cron job command, as the script is configured to read it from `os.environ.get("NASA_API_KEY")`.

---

4. 🗓️ Setting up the Cron Job

This cron job schedules the `fetch_nasa_data.py` Python script to run daily at a specific time.

A. Cron Job Entry (Template)

The entry below is configured to run the script **daily at 12:00 PM (noon)** and passes the NASA API Key as an environment variable.

Bash

```
# Minute (0) Hour (12) Day of Month (*) Month (*) Day of Week (*)
0 12 * * * NASA_API_KEY="YOUR_NASA_API_KEY" /usr/bin/python3 /var/www/html/dash-email/js/fetch_nasa_data.py
```

- **Schedule (`0 12 * * *`)**: Executes at **12:00 PM (noon)** every day.
- **Environment Variable**: Sets the **NASA_API_KEY** for this specific command execution.
  - **Action**: **Replace** `"YOUR_NASA_API_KEY"` with your live NASA API key.
- **Command**: Executes the Python script using the explicit path to the Python 3 interpreter.

B. File Location and Permissions (CRITICAL)

The script is configured to save its output files (`cme_latest.json`, `solar_flares_latest.json`, and `geomagnetic_storms_latest.json`) directly into the directory where it is executed.

- **Location:** It is **strongly recommended** to keep the script (`fetch_nasa_data.py`) and its output files within the web-accessible clone repository directory: `/var/www/html/dash-email/js/`.
- **Permissions:** You must ensure that the user running the cron job (often the web server user like **`www-data`** or the user who installed the cron job) has **full read and write permissions** to the entire `/var/www/html/dash-email/js/` directory to allow the creation and updating of all required JSON files.

C. Installation Steps (Linux/Unix)

1. **Open the Crontab:** Use the `crontab -e` command to edit the cron job file for the user who will be running this task.

   Bash

   ```
   crontab -e
   ```

2. **Add the Job:** Copy and paste the fully modified cron job entry from Section 4.A to the end of the crontab file.
3. **Save and Exit:** Save and close the file. The cron daemon will automatically install and activate the new scheduled job.

Footnote [^1]: It is highly recommended to create a separate instruction document detailing specific logging/output behavior and error handling (such as the script's reliance on writing output to JSON files in the same directory as the script) in addition to this setup guide. This ensures that the script's environment is correctly configured and monitored.