**Project Proposal**

**Were Vincent**

**Course: Computer Science**

**Name of Project:** Climate Data Integration and Analysis

**Type of Project:** Option B1: Substantial Data Collection (General)

**Introduction**

Climate change and its associated impacts are some of the most pressing challenges facing humanity today. Understanding and analyzing climate data is essential for making informed decisions that mitigate risks and enhance resilience. However, the diverse and scattered nature of climate data presents a significant challenge, requiring sophisticated methods to integrate and analyze information from multiple sources.

The **Climate Data Integration and Analysis** project addresses this challenge by creating a robust pipeline for acquiring, cleaning, and integrating data from APIs, web scraping, and SQL databases. By focusing on three distinct data sources OpenWeatherMap API for real-time weather data, NOAA’s historical climate data through web scraping, and supplementary climate-related data from a SQL database this project showcases the practical application of advanced data engineering and analysis techniques.

The project emphasizes technical proficiency in building a Python package that simplifies the data workflow. This package will enable users to collect, clean, and analyze climate data efficiently, providing actionable insights for researchers, policymakers, and environmentalists. By tackling common hurdles like API rate limits, web scraping complexities, and data integration issues, this project aims to highlight problem-solving abilities and technical skills in data science. Ultimately, this project demonstrates how diverse data sources can be unified into a cohesive, analysis-ready format to support meaningful climate-related studies and decision-making processes.

**Purpose**

The purpose of this project is to demonstrate technical proficiency in acquiring, cleaning, transforming, organizing, and presenting data. This involves leveraging multiple data acquisition techniques such as APIs, web scraping, and SQL queries to collect data from at least two sources. The project aims to wrangle these datasets into a cohesive and analysis-ready format while thoroughly documenting the entire process within a Python package. By completing this project, I intend to showcase my ability to handle complex data workflows and effectively present results in a user-friendly manner.

**Links to Data Sources**

* **API Data Source -** [OpenWeatherMap API](https://openweathermap.org/api) will be used to collect weather data, including temperature, humidity, and precipitation, for specific locations and time periods.
* **Web Scraped Data Source -**  [National Oceanic and Atmospheric Administration (NOAA)](https://www.noaa.gov/) will provide historical climate data through web scraping techniques.
* **SQL Database -** A publicly available database such as [SQLite Sample Database](https://www.sqlitetutorial.net/sqlite-sample-database/) will offer supplementary data, potentially related to climate patterns or significant weather events.

**Outline of Technical Steps/Challenges**

1. ***Data Acquisition -***The data acquisition phase involves collecting data from three distinct sources:

* The OpenWeatherMap API will be accessed using a Python script to retrieve weather data for specific geographical locations and periods. This process must account for potential API rate limits and data restrictions.
* A custom Python web scraping script will extract historical climate data from NOAA’s website. This step requires handling pagination, dynamically loaded content, and potential CAPTCHA challenges.
* An SQL database will be queried to fetch supplementary climate-related data. This may include leveraging SQL queries to filter, sort, and retrieve relevant information efficiently.

1. ***Data Cleaning and Transformation -* Once the data is collected, the cleaning and transformation process begins:**

* The datasets from all sources will be merged into a single cohesive dataset, ensuring compatibility between different formats.
* Missing values and inconsistencies, such as varying date formats and units of measurement, will be addressed to ensure data integrity.
* Numerical and categorical variables will be normalized and standardized to facilitate analysis.
* Regular expressions will be employed to clean and transform string data, such as standardizing location names.

1. **Data Organization and Documentation -** To make the data analysis-ready, the following steps will be undertaken:

* The cleaned dataset will be organized into a structured format, such as a pandas DataFrame, for seamless manipulation and analysis.
* Reusable functions will be created for each step of data acquisition and cleaning, promoting efficiency and reusability.
* Comprehensive documentation, including docstrings and markdown explanations, will accompany each function and the overall data cleaning pipeline to ensure clarity and usability.

1. ***Package Creation -***The final dataset and associated functions will be packaged into a Python library:

* All functions and datasets will be integrated into a Python package, making them accessible to other users.
* A README.md file will provide detailed information about the package’s purpose, installation steps, and usage instructions.
* A license will be included, and the package will be published on PyPi (or TestPyPi) for public availability.
* Automated tests for key functions will be written using pytest or similar frameworks to ensure reliability and robustness.

1. ***Presentation -***To showcase the outcomes, a Jupyter Notebook will be developed:

* The notebook will serve as a vignette demonstrating how to use the Python package, complete with code examples and outputs.
* Basic visualizations, such as histograms and scatter plots, will provide insights into the cleaned dataset.
* Challenges faced during data acquisition and cleaning will be documented, along with the strategies used to overcome them.

**Significant Hurdles**

1. ***API Rate Limits -***Potential restrictions on the number of API calls per minute or day may limit data collection. Strategies such as caching responses or using alternative APIs will be explored.
2. ***Web Scraping Complexity -*** Dynamic content and CAPTCHA-protected pages on NOAA’s website could pose challenges. Workarounds, such as automated browser tools, will be considered.
3. ***Data Integration -***Merging datasets with differing formats and handling missing data will require careful planning to ensure the final dataset’s usability.
4. ***Testing -***Writing effective tests for functions that rely on external data sources can be challenging due to the dynamic nature of these sources.

**Impact of Unresolved Hurdles**

1. If API rate limits are not addressed, the scope of data collection may be restricted. Cached responses or alternative APIs can serve as contingency measures.
2. Web scraping issues may reduce the availability of historical data, but manual downloads or alternative sources can fill gaps.
3. Challenges in data integration might lead to partial analyses, which, while not ideal, could still provide valuable insights.

**Conclusion**

This project aims to emphasize the practical application of data acquisition, cleaning, and packaging skills acquired during the course. By focusing on substantial data collection from diverse sources, it offers a comprehensive demonstration of technical proficiency and problem-solving abilities in data science. The outcome will be a robust Python package that facilitates data analysis and supports future projects in climate-related studies.