**# Climate Analysis and Flask API**

This project performs a basic climate analysis and data exploration using the `hawaii.sqlite` climate database, followed by building a Flask API to serve the results.

**## Part 1: Climate Data Analysis**

In this section, we use Python with \*\*SQLAlchemy\*\*, \*\*Pandas\*\*, and \*\*Matplotlib\*\* to explore the climate data.

**### Setup**

- Use `create\_engine()` from SQLAlchemy to connect to the SQLite database.

- Use `automap\_base()` to reflect the database tables into classes.

- Assign the reflected classes to the variables `Station` and `Measurement`.

**### Precipitation Analysis**

- Find the most recent date in the dataset.

- Query the previous 12 months of precipitation data (`date` and `prcp` columns).

- Load the results into a Pandas DataFrame, set the index to the `date` column, and sort by date.

- Plot the precipitation data using Pandas' plot function.

- Print summary statistics for precipitation using Pandas.

**### Station Analysis**

- Query the total number of stations.

- Identify the most active stations by:

- Listing stations and their observation counts in descending order.

- Finding the station ID with the highest number of observations.

- Query the minimum, maximum, and average temperatures for the most active station.

- Retrieve the previous 12 months of temperature observations (`TOBS`) for that station.

- Plot a histogram of the temperature observations with 12 bins.

**## Part 2: Flask API**

Build a Flask API using the analysis results. Create the following routes:

### Available Routes

- `/`

Homepage listing all available API routes.

- `/api/v1.0/precipitation`

Returns the last 12 months of precipitation data as a JSON dictionary with dates as keys and precipitation as values.

- `/api/v1.0/stations`

Returns a JSON list of all station IDs.

- `/api/v1.0/tobs`

Returns a JSON list of temperature observations for the previous year from the most active station.

- `/api/v1.0/<start>`

Returns the minimum, average, and maximum temperatures for all dates greater than or equal to the specified start date.

- `/api/v1.0/<start>/<end>`

Returns the minimum, average, and maximum temperatures for dates between the specified start and end dates, inclusive.

**## Files Provided**

- `Hawaii. SQLite` — The SQLite database.

- `Climate\_starter.ipynb` — Starter notebook for analysis.

- `app.py`—Flask app script (to be created).

**## Requirements**

- Python

- SQLAlchemy

- Pandas

- Matplotlib

- Flask

**## How to Run**

1. Complete the climate data analysis in the notebook.

2. Build and test the Flask API in `app.py`.

3. Run the Flask app:

```bash python app.py