Backend Problem Statement

Task: Real-time Weather Forecast API

Scenario:

You're tasked with creating a RESTful API that provides real-time weather forecasts based on geographical locations. The API should fetch data from an external weather service and expose endpoints to retrieve weather information for specific locations.

Requirements:

Location Management:

Users can add, retrieve, update, and delete locations.
Each location should have a name, latitude, and longitude.

Weather Forecast:

Users can request weather forecasts for a specific location.
The API should fetch real-time weather data from a weather service (e.g., OpenWeatherMap, WeatherAPI).
Provide forecasts for parameters like temperature, humidity, wind speed, etc.

Endpoints:

Design the following endpoints:

/locations (GET, POST): Get all locations or add a new location
/locations/<location_id> (GET, PUT, DELETE): Get, update, or delete a specific location by ID
/weather/<location_id> (GET): Get the weather forecast for a specific location
/history (last 7 days, last 15 days, last 30 days) (GET): Get the historical data and show the summary.

Technical Guidelines:

Integrate with an external weather service API to fetch real-time weather data based on location coordinates.
Implement caching mechanisms to reduce the number of external API calls.
Ensure proper error handling for cases where the external service is unavailable or returns errors.
Use proper validation for location data and handle edge cases gracefully.

Additional Considerations:

Implement rate limiting to prevent abuse of the API.
Implement logging for API requests, especially those interacting with the external service.
Use environment variables for sensitive information like API keys.

Evaluation Criteria:

Correctness and functionality of the API endpoints, including proper integration with the external weather service.
Proper handling of location management and real-time data retrieval.
Consistency with RESTful API design principles.
Error handling for scenarios like unavailable services or invalid location data.
Efficiency in terms of API response time and caching strategies.