Introduction

For this exercise, we will ask you to build an API service that allows its users to:

- Create/List/Remove/Delete Locations they are interested in
- Get historical forecasted temperature values for locations they are interested in

CRUD Operations on Locations

Users must be able to perform CRUD operations on **locations**.

When creating a location, users must specify:

- Latitude: a floating-point that specifies the latitude of the location
- Longitude: a floating-point that specifies the longitude of the location
- Slug: a name for the location that's URL-safe. This slug is unique per location

Historical Temperature Forecasting Fetching

Users must be able to query historical temperature forecasted values for **locations** they previously added to the system.

When querying temperatures, the user must specify:

- The location to query. This must be a slug of a previously added location
- The start_date. This is a string of the form YYYY-MM-DD
- The end_date. This is a string of the form YYYY-MM-DD

The API must return a list of the dates contained in the period requested and for each one of them, the minimum and maximum temperatures that were forecasted for that date, in JSON format.

If a user asks for forecasted temperature data between Jan 1 2021 and Jan 3 2021, the API should return :


```
[
  {"date": "2021-01-01", "min-forecasted": 18, "max-forecasted": 26},
  {"date": "2021-01-02", "min-forecasted": 17, "max-forecasted": 24},
  {"date": "2021-01-03", "min-forecasted": 19, "max-forecasted": 25}
]
```

Data Origin

To fetch forecasted temperature values, we ask you to leverage:
http://www.7timer.info/doc.php?lang=en

Use the “Machine-readable API”, which can return values in XML or JSON. This API does not require any key / registration.

The ASTRO product returns temperatures.

In the sample call:
https://www.7timer.info/bin/astro.php?lon=113.2&lat=23.1&ac=0&unit=metric&output=json&tzshift=0

The API returns the forecasted temperature in celsius for every 3 hour window, in the “temp2m” key (Temperature at 2 meters).

Additional Information & Requirements

Challenge Details

Once a location has been added to the list, we need you to allow the users to query historical forecasting starting on the date the location was added, and get accurate data for any day since.

We are aware that the 7timer API does not allow querying on past forecasts, so the system you design must account for that.

Please only leverage that API as a source of data and ensure your system is built to ensure all required data are fetched as needed.

If a user asks for historical forecast with a start date preceding the date the location was added, we expect you to gracefully deal with the error but we know you won’t be able to return any value.

Important / Less important point

- This is purely a backend project, we don’t expect any visual interface.
- We do not require any kind of authentication/security for the API. Consider that this API would run in a secure network with trusted and cooperative clients. Design the API to be as easy and simple to use as possible.
- We would like you to use Ruby on Rails to develop this case. You are free to use any libraries that you want, as long as you don’t leverage any other source of data for the temperature values. If in doubt, don’t hesitate to ask.
- We do care about data correctness and the general resiliency of the architecture of your system.

Delivery

For the delivery of your solution, we ask you to share a repository with us in a Git service such as GitHub, GitLab or BitBucket.

Be mindful that the README documentation may be an important part of your project.