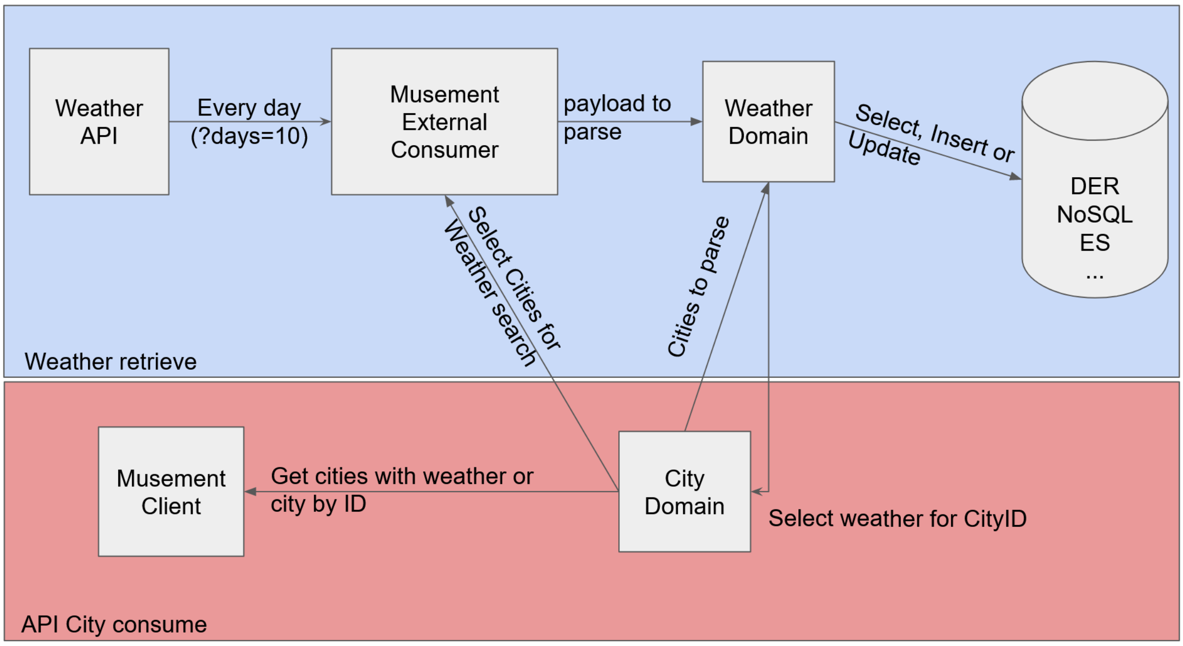
For the architectural issue proposed, I suggest a Consumer (Musement External Consumer) that runs every day, once a day, retrieving data from weather API, and parsing it to our structure model, saving and updating it to an existing data repository (Elastic Search, Mongo, MySQL, etc) creating a cache to futures days
Just remember that the maximum number of days given by Weather API is 10 days, so using this service, we can only deliver 10 days in advance of forecast informantion.



The API to save the forecast information can be POST /forecast and the body like below:

```
[
  {
  “city_id” : int,
  “id”: int,
   “uuid”:”string”,
   "date": "string",
   "day": {
      "maxtemp_c": float,
      "mintemp_c": float,
      "avgtemp_c": float,
      "daily_will_it_rain": boolean,
      "daily_will_it_snow": boolean,
      "condition": {
          "text": "Sunny"
       }
    }
  }
]
```

This body permits saving a single day or a bunk of 10 days at the same time.

The API to retrieve only the weather forecast of a city is GET /forecast/{city_id}/{num_of_day} or, to get the city information with forecast just change the API version to v4 as GET /V4/cities or /V4/cities/{city_id} and change the response to the following:

```
[
 {
      "id": 0,
      "uuid": "string",
      "top": true,
      "name": "string",
      "code": "string",
      "content": "string",
      "content_html": "string",
      "meta_description": "string",
      "meta_title": "string",
      "headline": "string",
      "more": "string",
      "more_html": "string",
      "weight": 0,
      "latitude": 0,
      "longitude": 0,
      "country": {
      "id": 0,
      "name": "string",
      "iso_code": "string",
      "country_prefix": "string",
      "currency_code": "string"
      },
      "cover_image_url": "string",
      "url": "string",
      "activities_count": 0,
      "time_zone": "string",
      "list_count": 0,
      "venue_count": 0,
      "slug": "string",
      "show_in_popular": true
      “forecast”:[
            {
              "date": "string",
              "day": {
                "maxtemp_c": 16.6,
                "mintemp_c": 3.4,
                "avgtemp_c": 8.4,
                "daily_will_it_rain": 0,
                "daily_will_it_snow": 0,
                "condition": {
                  "text": "Sunny"
```

```
                    }
                }
        ]
 }
]
```

Resuming:

- create a crawler to retrieve 10 days of forecast, running every day, once a day;
- create a microservice with an internal endpoint to save the crawled data:
    - POST /forecast
    - Body:

      ```
      [
        {
        “city_id” : int,
        “id”: int,
         “uuid”:”string”,
         "date": "string",
         "day": {
            "maxtemp_c": float,
            "mintemp_c": float,
            "avgtemp_c": float,
            "daily_will_it_rain": boolean,
            "daily_will_it_snow": boolean,
            "condition": {
       "text": "Sunny"
             }
           }
        }
      ]
      ```

- the API to get only the forecast information of a city:
    - GET /forecast/{city_id} - gets the current day
    - GET /forecast/{city_id}/{num_of_day} - get many days in the future from today
    - GET /forecast/{city_id}?day=YYYY-MM-DD - get a specific date in the future (máx 10 days in advance)
- the API to get the City information with a forecast is the same, but we must change the API version due to the fact we change the response body
    - GET /V4/cities
    - GET /V4/cities/{id}
    - We can also add the following querystrings to the address:
        - forecast_day - gets the forecast of a specific date
        - forecast_days - get the forecast of n days in the future, since today