Sample Problem

Build an API (1) that allows a user to enter a date time range (2) and get back the price at which they would be charged to park for that time span (3)

4. The application publishes an API that computes a price for an input datetime range
5. The user specifies input date/times as ISO-8601 with timezones
6. Rates are specified by a JSON file

   a. A rate is comprised of a price, time range the rate is valid, and days of the week the rate applies to

   b. See here for an example
7. User input can span more than one day, but the API shouldn't return a valid price - it should return `unavailable`
8. User input can span multiple rates, but the API shouldn't return a valid price - it should return `unavailable`
9. Rates will not span multiple days
10. The application publishes a second API endpoint where rate information can be updated by submitting a modified rates JSON and can be stored in memory

Requirements

11. Preferred languages are Python, Kotlin, Java, or Go to complete this
    - If you are planning on using a different language, please let the recruiting team know your language of choice
12. It should support JSON over HTTP
13. API endpoints should be documented
14. Tests need to be in place

Extra Credit

15. Include a Swagger Spec
16. Include a Dockerfile
17. Metrics for endpoint(s) captured and available to be queried via an endpoint (e.g. average response time). Add the metrics you feel would be appropriate to identify the health and performance of your application

Submitting

- Zip up your submission and submit it via GreenHouse
  - You can achieve this with `git archive --format zip --output /full/path/to/zipfile.zip master` if using git
- Include any instructions on how to build, run, and test your application

Sample JSON for testing

```
{
    "rates": [
        {
            "days": "mon,tues,thurs",
            "times": "0900-2100",
            "tz": "America/Chicago",
            "price": 1500
        },
        {
            "days": "fri,sat,sun",
            "times": "0900-2100",
            "tz": "America/Chicago",
            "price": 2000
        },
        {
            "days": "wed",
            "times": "0600-1800",
            "tz": "America/Chicago",
            "price": 1750
        },
        {
            "days": "mon,wed,sat",
            "times": "0100-0500",
            "tz": "America/Chicago",
            "price": 1000
        },
        {
            "days": "sun,tues",
            "times": "0100-0700",
            "tz": "America/Chicago",
            "price": 925
        }
    ]
}
```

The timezones specified in the JSON file adhere to the 2017c version of the tz database. Assume that there could be other (non America/Chicago) timezones specified. For more information: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones

Assume that rates in this file will never overlap

Sample result

Datetime ranges should be specified in ISO-8601 format. A rate must completely encapsulate a datetime range for it to be available.

- `2015-07-01T07:00:00-05:00` to `2015-07-01T12:00:00-05:00` should yield `1750`
- `2015-07-04T15:00:00+00:00` to `2015-07-04T20:00:00+00:00` should yield `2000`
- `2015-07-04T07:00:00+05:00` to `2015-07-04T20:00:00+05:00` should yield `unavailable`