Mercado Libre Ejercicio de programación

Para coordinar acciones de respuesta ante fraudes, es útil tener disponible información contextual del lugar de origen detectado en el momento de comprar, buscar y pagar.
Para ello, entre otras fuentes, se decide crear una herramienta que dado un IP obtenga información asociada.

Construir una aplicación que dada una dirección IP, encuentre el país al que pertenece, y muestre:

- El nombre y código ISO del país
- Los idiomas oficiales del país
- Hora(s) actual(es) en el país (si el país cubre más de una zona horaria, mostrar todas)
- Distancia estimada entre Buenos Aires y el país, en km.
- Moneda local, y su cotización actual en euros (si está disponible)

Basado en la información anterior, es necesario contar con un mecanismo para poder consultar las siguientes estadísticas de utilización del servicio con los siguientes agregados:

- Distancia más lejana a Buenos Aires desde la cual se haya consultado el servicio
- Distancia más cercana a Buenos Aires desde la cual se haya consultado el servicio
- Distancia promedio de todas las ejecuciones que se hayan hecho del servicio.

Ver ejemplo

País	Distancia	Invocaciones
Brasil	2862 km	10
España	10040 km	5

En este caso la cuenta a realizar debería ser: (2862 km * 10 + 10040 km* 5) / 15 = 5254 km

Para resolver la información, pueden utilizarse las siguientes APIs públicas:

- Geolocalización de IPs: https://ip2country.info/
- Información de paises: http://restcountries.eu/
- Información sobre monedas: http://fixer.io/

Desafíos

Desarrollar una aplicación que sea api REST que respete las siguientes firmas:

- POST → */trace/*

  {

  “ip”:”123.123.123.132”

  }

  - Debe devolver la información relacionada a la ip en formato JSON
- GET → */stats/*
  - No recibe parámetros, devuelve las 3 estadísticas en formato JSON

Condiciones:

- La aplicación deberá estar desarrollada en Java / Goland / Node (Javascript)
- La aplicación deberá hacer un uso racional de las APIs, evitando hacer llamadas innecesarias.
- Tener en cuenta que ambos endpoints pueden recibir fluctuaciones agresivas de tráfico (Entre 100 y 2 millón de peticiones por minuto). Ejemplo se recibe una notificación push en el celular que envía tráfico a un frontend que utiliza este servicio por ende va a recibir muchos requests de dichos usuarios.
- La aplicación puede tener estado persistente entre invocaciones.
- Además de funcionamiento, prestar atención al estilo y calidad del código fuente.
- El código fuente debe estar alojado en algún repositorio público (Github, Gitbucket, etc).
- Además del código fuente, entregar cualquier pieza adicional que consideres aporta a la calidad del software construido.

BONUS

- Hostear la aplicación en algún cloud (Heroku, Google Cloud Platform, AWS, etc)
- La aplicación deberá poder correr ser construida y ejecutada dentro de un contenedor Docker (incluir un Dockerfile e instrucciones para ejecutarlo).

Ejemplo del /trace (el formato es tentativo y no tiene porque ser exactamente así):

```
> curl -X POST -d "{'ip':'83.44.196.93'}"
"meli-trace.cloud.com/trace"
```

Respuesta tentativa:

```
{
     "ip":"83.44.196.93",
     "date":"21/11/2020 15:12:03",
     "country":"España (spain)",
     "iso_code":"es",
     "languages": ["Español (es)"],
     "currency": "EUR (1 EUR = 1.0631 U$S)",
     "times": ["20:01:23 (UTC)", "19:01:23 (UTC-01)"],
     "estimated_distance": "10270 kms"
}
```

Los formatos de los parámetros pueden variar pero deben contener toda la información