Challenge: MeliDiscount

Contexto:

Con el objetivo de incrementar las ventas de los vendedores, Mercado Libre está creando un nuevo tipo de promoción (de ahora en más llamada "meli discount"), la cual consiste en que para cada vendedor en un determinado momento, el mismo puede indicar a qué publicación (de ahora en más llamada “ítem”) con estado “activo” quiere que tenga este meli discount.

Premisas:

- Dado un vendedor, el mismo solamente puede tener 1 ítem en estado activo con el meli discount.
- Un ítem está en estado “activo” entre las fechas de **date_created** y **last_updated,** y pertenecen únicamente a un vendedor determinado por el campo **seller_id** (estos campos se encuentran en la API de ítems de Mercado Libre).
- Una vez que un ítem tiene el meli discount, él mismo es válido desde date_created y last_updated y no se puede modificar.
- date_created y last_updated están en formato ISO 8601 y **date_created < last_updated**.
- Un ítem con meli discount no se puede solapar con otro item con meli discount, es decir que el campo last_updated del item1 **debe ser menor** al campo date_created del item2.

Desafío

Nivel 0:

Implementar una estrategia de Mocking para los **Recursos** de Mercado Libre a consumir, sin incluir la capa de seguridad.

Nivel 1:

Crear una API REST, con el servicio “**/meli_discount”** en donde se pueda enviar una lista de item ids. El endpoint debe devolver de manera ordenada la mayor cantidad de item ids que cumplan con las premisas previamente mencionadas.

Ejemplo:
Dada la siguiente lista de items:

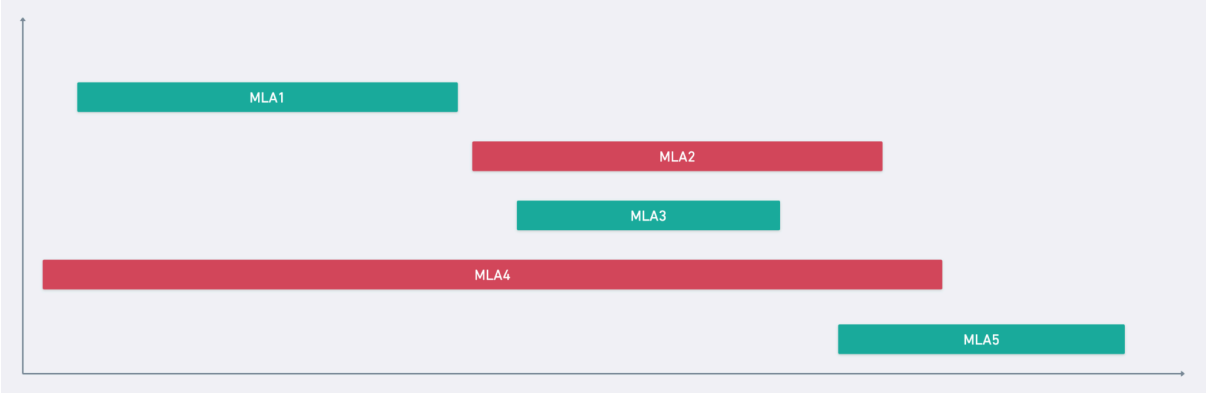
Item Id	date_created	last_updated

MLA1	2024-04-05T16:21:40.000Z	2024-04-10T19:40:06.000Z
MLA2	2024-04-11T08:26:30.000Z	2024-06-05T23:23:19.000Z
MLA3	2024-04-13T14:53:20.000Z	2024-04-25T18:11:43.000Z
MLA4	2023-04-05T13:41:32.000Z	2024-06-20T16:21:40.000Z
MLA5	2024-06-04T09:17:36.000Z	2024-07-05T20:51:32.000Z

La respuesta sería:

```
{
      "item_ids": ["MLA1", "MLA3", "MLA5"]
}
```

Explicación:



Si ponemos en un eje X los distintos ítems, podemos observar que la mayor cantidad de ítems que no se solapan para obtener el MeliDiscount son 3 (MLA1, MLA3 y MLA5).
A la combinación de MLA1 y MLA2 no se le puede agregar los otros ítems ya que se solapan, por ende como son solo 2, la combinación anterior tiene mayor cantidad de ítems.
Lo mismo pasa con MLA4, que si bien arranca en 2023-04-05T13:41:32.000Z, se solapa con todos los demás items.

Nota: Para buscar ítems reales de un mismo vendedor, puedes entrar en la página de Mercado Libre y usar cualquiera de los que aparezca en el listado. Luego de seleccionar uno, scrollea más abajo hasta que aparezca la opción de **“Ver más productos del vendedor”,** esta información puede ser usada para la implementación de la estrategia de monking

GET → /meli_discount?item_ids=MLA1,MLA2,MLA3,MLA4,MLA5

Response:

```
{
      "item_ids": ["MLA1", "MLA3", "MLA5"]
}
```

Nivel 2:

Ahora se quiere ser un poco más granular con el meli discount y Mercado Libre quiere segmentarlo por categorías. Esto significa que un vendedor puede tener varios ítems activos con el meli discount en un mismo tiempo pero todos deben ser de diferentes categorías padres.

Premisas:

- Un ítem siempre tiene una única categoría hoja asociada, indicada con el campo **category_id**
- Una categoría hoja es una categoría **que no tiene hijos** (jerarquía descendente)
- Una categoría padre es una categoría que tiene **al menos 1 categoría hija**
- Una categoría puede tener N categorías en su jerarquía ascendente, indicadas en el campo **path_from_root** (Si tiene solo 1, se trata de ella misma)
- Una categoría puede tener N categorías hijas, indicadas en el campo **children_categories** (N puede ser 0).
- Las categorías suelen ser estáticas.

Ejemplo:
Dadas estas 2 categorías de ejemplo:

```
{
    "id": "MLA1000",
```

```
    "name": "Categoria de ejemplo 1",
    "path_from_root": [
        {
            "id": "MLA1000"
        }
    ],
    "children_categories": [
        {
            "id": "MLA1001"
        },
        {
            "id": "MLA1002"
        },
        {
            "id": "MLA1003"
        }
    ]
}
```

```
{
  "id": "MLA4000",
  "name": "Categoria de ejemplo 2",
  "path_from_root": [
    {
      "id": "MLA2000"
    },
    {
      "id": "MLA3000"
    },
    {
      "id": "MLA4000"
    }
  ],
  "children_categories": [
    {
      "id": "MLA4001"
    },
    {
      "id": "MLA4002"
    },
    {
      "id": "MLA4003"
    }
  ]
}
```

Y los siguientes items con sus categorías (los campos date_created y last_updated son los mismos que en el ejemplo inicial):

Item Id	category_id
MLA1	MLA1001
MLA2	MLA1002
MLA3	MLA4001
MLA4	MLA1003
MLA5	MLA4002

GET → /meli_discount/categories?item_ids=MLA1,MLA2,MLA3,MLA4,MLA5

La respuesta sería:

```
[
  {
    "root_category_id": "MLA1000",
    "item_ids": ["MLA1","MLA2"]
  },
  [
    {
      "root_category_id": "MLA2000",
      "item_ids": ["MLA3","MLA5"]
    }
  ]
]
```

Nivel 3:

Hostear las APIs en un cloud computing libre (Google App Engine, Amazon AWS, etc)

Recursos:

Para obtener los campos **date_created**, **last_updated**, **category_id** y **seller_id** de un ítem se puede consultar a la API de ítems:

Llamado al api de ítems

```
curl --location 'https://api.mercadolibre.com/items/$ITEM_ID'

curl --location 'https://api.mercadolibre.com/items/MLA1473083829'
```

Response resumido del api de items

```
{
    "id": "MLA1473083829",
    "seller_id": "1064597768",
    "category_id": "MLA30835",
    ..........
    "date_created": "2025-01-23T18:50:14.899Z",
    "last_updated": "2025-01-23T18:50:17.542Z",
    ..........
}
```

Para obtener los campos **path_from_root** y **children_categories** de una categoría se puede consultar a la API de categorías:

Llamada al api de Categorias

```
curl --location 'https://api.mercadolibre.com/categories/$CATEGORY_ID'

curl --location 'https://api.mercadolibre.com/categories/MLA1051'
```

Response resumido del api de items

```
{
    "id": "MLA1051",
    ..............
    "path_from_root": [
        {
            "id": "MLA1051",
            "name": "Celulares y Teléfonos"
        }
    ],
    "children_categories": [
        {
            "id": "MLA3502",
            "name": "Accesorios para Celulares"
        },
        {
            "id": "MLA1055",
            "name": "Celulares y Smartphones"
        },
        ..............
    ],
    ............
}
```

Puedes encontrar más información sobre los recursos públicos de mercado libre acá.

Consideraciones:

- Un vendedor puede tener **miles** de ítems.
- Cada ítem tiene un id único e irrepetible.
- Esta API tendría que escalar para soportar tráfico de hasta 100K RPM (Requests Per Minute).

Entregables:

- Codebase (Repositorio de Github privado). Dar permiso de lectura al usuario **SupplySCReview**.
- Lenguaje: Preferentemente realizar el ejercicio en Java o en su defecto Go.
- Instrucciones para levantar la app.
- URL de las API (Nivel 3).
- Cualquier información/documentación que te parezca relevante.