Alumno: Andrés Capone
Legajo: 42821

Arquitectura de Microservicios

Módulo de Puntuación y comentarios de producto

“ MicroReviews “

IMPORTANTE: Se puede ver el APIDOC cuando se ejecuta el código, entrando a http://localhost:3006/

Link de Github:
https://github.com/AndresCapone/MicroReviews---Microservicios-Utn

Profesores:
Andrés Ceccoli
Néstor Marsollier

Alumno:
Andrés Capone
Legajo:
42821

Alumno: Andrés Capone
Legajo: 42821

ÍNDICE:

Alumno: Andrés Capone
Legajo: 42821

Descripción del Microservicio:

MicroReviews es un microservicio programado en Python con el fin de darle a los usuarios la posibilidad de escribir reseñas y poner puntuaciones del 1 al 10 a los productos que han comprado en el ecommerce propuesto por el profesor. De esta forma otros usuarios podrán saber la opinión que se ha dado de los productos que desean comprar y tendrán la información necesaria para saber si es lo que realmente desean o no.

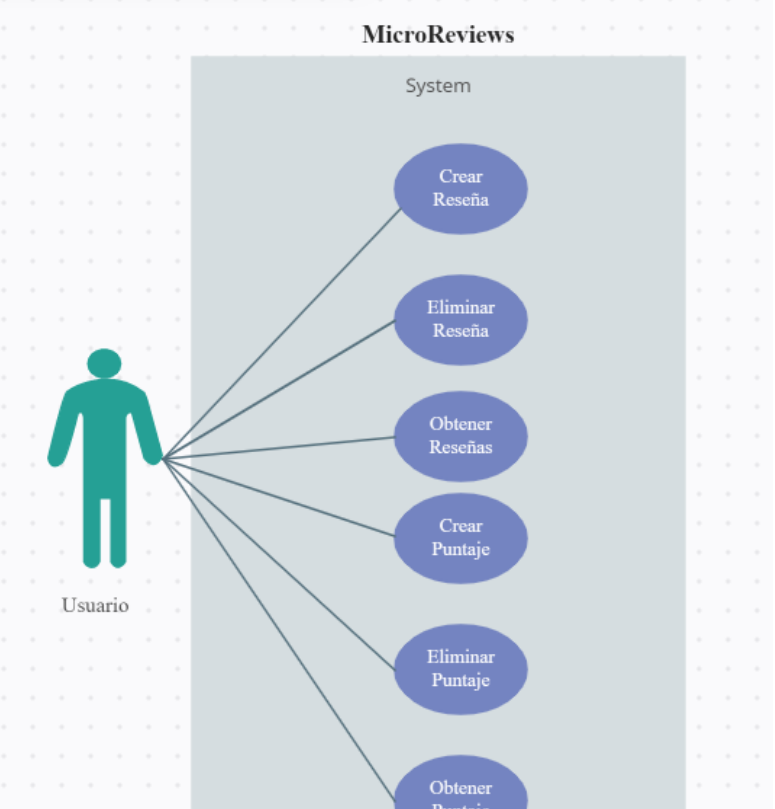
Lista de Casos de Uso:

o Crear Reseña
- **POST:** /v1/reviews/<string: id_article>

o Eliminar Reseña
- **DELETE:** /v1/reviews/<string: id_article>

o Obtener Reseña
- **GET:** /v1/reviews/<string: id_article>

o Crear Puntaje



- **POST:** /v1/scores/<string: id_article>

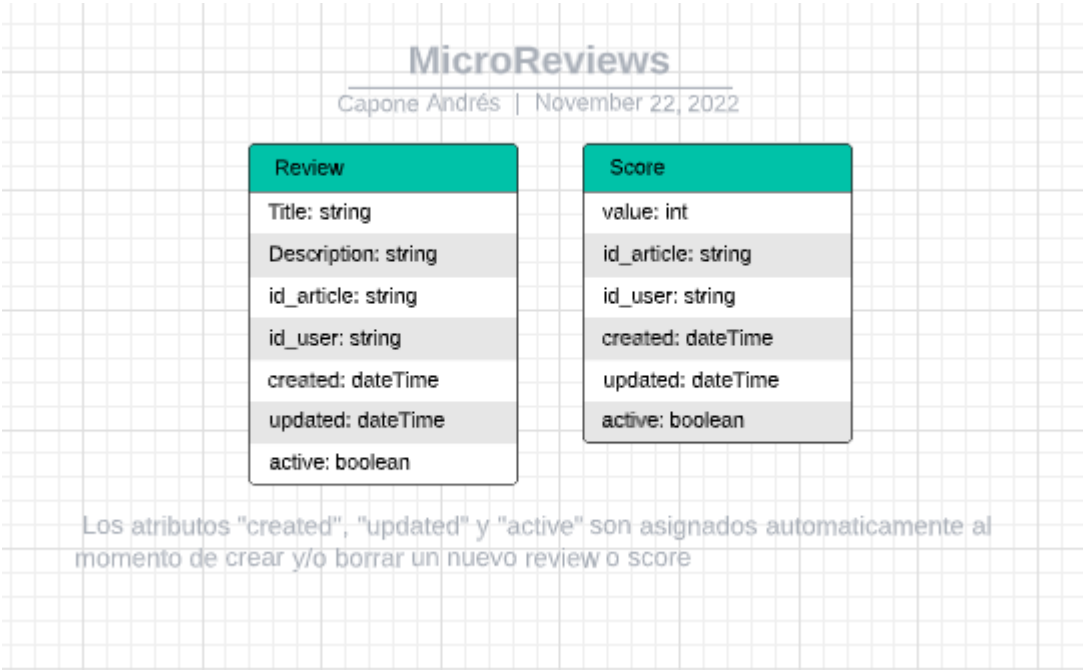
o Eliminar Puntaje

- **DELETE:** /v1/scores/<string: id_article>

o Obtener Puntaje

- **GET:** /v1/scores/<string: id_article>

Modelo Lógico:



La primer tabla se utiliza para guardar la información importante sobre las reviews, así como el título, descripción de la review y a que articulo y usuario se relaciona. y la segunda para guardar la información importante sobre los scores, el valor asignado al producto y a que producto y usuarios está relacionado

Se eligió esta forma de almacenar la información para realizar consultas rápidas sin necesidad de duplicar la información del sistema y de los demás microservicios.

Tecnologías a utilizar:

- **MicroReviews(app):**
  -Python
  -Rabbit MQ
  -MySQL

Alumno: Andrés Capone
Legajo: 42821

- **apiDoc(localhost:3006):**
  -Html
  -Javascript
  -CSS
  -Bootstrap

Funcionamiento:

Cada vez que un usuario compre un producto, éste podrá realizar una reseña o varias reseñas sobre su producto que compró y solo una puntuación del mismo.(Por lo tanto para que un usuario pueda puntuar y reseñar, la orden debe tener el estado de “PAYMENT_DEFINED”, de lo contrario no podra hacerlo)
Cuando el usuario desee hacer esto, el microservicio realizará un request y consultará con los id's correspondientes, en los demás microservicios, los datos necesarios para devolver las consultas con la info necesaria:

- Para los datos de los productos, MicroReviews buscará en el microservicio catalog con el endpoint Buscar Artículo de los microservicios del ecommerce

- Para los datos del, MicroReviews buscará en el microservicio auth con el endpoint Usuario Actual de los microservicios del ecommerce

para los reviews el microservicio solo guardará el título, descripción e id del artículo y usuario y para los scores solo la puntuación junto con los id de usuario y artículo nuevamente ya que es la información relevante para los casos de uso.
Para ambos casos se a su vez guardará automáticamente en la base de datos la información de cuando fueron creados, modificados y si se encuentran activos o no.

APIDOC

Reseñas

Los usuarios pueden realizar todas las reseñas(reviews) que quieran sobre un producto, estas reseñas tendrán los siguientes endpoints:

- **Crear reseña**

Guarda en la base de datos la reseña realizada por un usuario a un producto determinado

POST: /v1/reviews/<string: id_article>

Success Response

- ❖ Respuesta HTTP/1.1 200 OK

```
{
        “_id”: “{id del review}”,
        “id_article”: “{id del artículo}”,
        “id_user”: “{id del artículo}”,
        “title”: “{titulo del review}”,
        “description”: “{comentario del review}”,
        “updated”: {fecha última actualización},
        “created”: {fecha creación}
}
```

Error Response

- 401 Unauthorized: HTTP/1.1 401 Unauthorized

```
{
   "error" : "Unauthorized"
 }
```

- 400 Bad Request: HTTP/1.1 400 Bad Request

```
{
  "messages" : [
        {
        "path" : "{Nombre de la propiedad}",
        "message" : "{Motivo del error}"}
        },
        ...
```

```
        ]
    }
```

- 500 Server Error: HTTP/1.1 500 Internal Server Error

```
{

   "error" : "Not Found"

}
```

- **Eliminar reseña**

Elimina en la base de datos la reseña realizada por un usuario a un producto determinado

DELETE: /v1/reviews/<string: id_article>

Success Response

- Respuesta HTTP/1.1 200 OK

```
{
    “deleted”: true
}
```

Error Response

- 401 Unauthorized: HTTP/1.1 401 Unauthorized

```
{

   "error" : "Unauthorized"

 }
```

Alumno: Andrés Capone
Legajo: 42821

- 400 Bad Request: HTTP/1.1 400 Bad Request

```
{
                    "messages" : [
                            {
                            "path" : "{Nombre de la propiedad}",
                            "message" : "{Motivo del error}"}
                            },
                            ...
                    ]
}
```

- 500 Server Error: HTTP/1.1 500 Internal Server Error

```
                {
                   "error" : "Not Found"
                }
```

- **Obtener reseña**

Alumno: Andrés Capone
Legajo: 42821

Busca en la base de datos todas las reseñas realizadas a un producto determinado

GET: /v1/reviews/<string: id_article>

Success Response

- Respuesta HTTP/1.1 200 OK

```
[{
“_id”: “{id del review}”,
“id_article”: “{id del artículo}”,
“id_user”: “{id del artículo}”,
“title”: “{titulo del review}”,
“description”: “{comentario del review}”,
“updated”: {fecha última actualización},
“created”: {fecha creación}
}, ...]
```

Error Response

- 401 Unauthorized: HTTP/1.1 401 Unauthorized

```
{
   "error" : "Unauthorized"
}
```

- 400 Bad Request: HTTP/1.1 400 Bad Request

```
{
  "messages" : [
```

```
                {

                "path" : "{Nombre de la propiedad}",
                "message" : "{Motivo del error}"}
                },

                ...

            ]
        }
```

- ❖ 500 Server Error: HTTP/1.1 500 Internal Server Error

```
        {

            "error" : "Not Found"

        }
```

PUNTAJE (SCORES)

Los usuarios pueden realizar SOLO UN puntaje a un producto, de esta forma no habra una puntuación desmedida sobre un producto, estos puntajes tendrán los siguientes endpoints:

- **Crear Puntaje**

Guarda en la base de datos la puntuación realizada por un usuario a un producto determinado

POST: /v1/reviews/<string: id_article>

Success Response

- ❖ Respuesta HTTP/1.1 200 OK

```
        {
            "_id": "{id del score}",
            "id_article": "{id del artículo}",
            "id_user": "{id del artículo}",
            "value": int del 1 al 10,
            "updated": {fecha última actualización},
```

```
                “created”: {fecha creación}
}
```

Error Response

- 401 Unauthorized: HTTP/1.1 401 Unauthorized

```
{

   "error" : "Unauthorized"

}
```

- 400 Bad Request: HTTP/1.1 400 Bad Request

```
{

  "messages" : [

        {

        "path" : "{Nombre de la propiedad}",
        "message" : "{Motivo del error}"}
        },

        ...

 ]
}
```

- 500 Server Error: HTTP/1.1 500 Internal Server Error

```
{
```

```
                "error" : "Not Found"

            }
```

- **Eliminar Puntaje**

Elimina en la base de datos la puntuación realizada por un usuario a un producto determinado

DELETE: /v1/scores/<string: id_article>

Success Response

- Respuesta HTTP/1.1 200 OK

```
{
        “deleted”: true
}
```

Error Response

- 401 Unauthorized: HTTP/1.1 401 Unauthorized

```
{

   "error" : "Unauthorized"

 }
```

- 400 Bad Request: HTTP/1.1 400 Bad Request

```
{

            "messages" : [
```

```
                    {

                    "path" : "{Nombre de la propiedad}",
                    "message" : "{Motivo del error}"}
                    },

                    ...

                ]
}
```

❖ 500 Server Error: HTTP/1.1 500 Internal Server Error

```
                {

                  "error" : "Not Found"

                }
```

- **Obtener puntaje**

busca en la base de datos todas las puntuaciones realizadas por un a un producto determinado, con esas puntuaciones saca un promedio y las muestra

GET: /v1/scores/<string: id_article>

Success Response

❖ Respuesta HTTP/1.1 200 OK

```
                [{
```

```
                "value": promedio de puntuaciones del artículo,
                "id_article": "{id del artículo puntuado}",

        }, ...]
```

Error Response

- 401 Unauthorized: HTTP/1.1 401 Unauthorized

```
{

   "error" : "Unauthorized"

 }
```

- 400 Bad Request: HTTP/1.1 400 Bad Request

```
{

  "messages" : [

        {

        "path" : "{Nombre de la propiedad}",
        "message" : "{Motivo del error}"}
        },

        ...

  ]
}
```

- 500 Server Error: HTTP/1.1 500 Internal Server Error

Alumno: Andrés Capone
Legajo: 42821

```
{

   "error" : "Not Found"

}
```