Universidad Nacional
Facultad de Ciencias Exactas y Naturales
Escuela De Informática

EXAMEN FINAL

CURSO:	Programación IV
PROFESOR:	Darío Ríos Navarro
GRUPOS:	**Máximo dos personas**.
FECHA DE ENTREGA:	**Miércoles 24 de noviembre de 2021**

La empresa **Todo Mascotas CR** lo ha contratado para el desarrollo de una API que le permita mantener la lista de productos de su sitio web de mascotas. El sitio web cuenta con la siguiente información que se debe ver reflejado en su base de datos:

Hay dos tablas en la base de datos. La primera es una tabla de productos que tiene las siguientes columnas: id del producto que es un entero auto incremental, nombre del producto que es un varchar de 255 caracteres, descripción del producto que corresponde a un varchar de 500 caracteres. Además, tiene un campo url que corresponde a la imagen, de 500 caracteres. Finalmente, cuenta con un precio de tipo double y una cantidad de inventario de tipo entero.

La otra tabla es de categorías. Esta tabla tiene una relación foránea con productos. Un producto tiene muchas categorías. La tabla categorías cuenta con los campos de id que es un entero auto incremental, y descripción que es un campo varchar de 255 caracteres.

I PARTE. API de productos

La API de productos debe tener los siguientes puntos de acceso, considerando su verbo http:

GET : api/productos – retorna una lista de objetos JSON con todos los datos de la tabla productos

Si no hay datos en la base de datos, devuelva una lista vacía.

GET : api/productos/{id} – retorna un objeto JSON correspondiente con los datos del id dado.

Si no encontró el registro indicado con ese id, devuelva un error 404, con el mensaje: Error el producto no fue encontrado.

POST: api/productos – permite la inserción de un producto, con toda la información descrita anteriormente.

Debe validar que el cuerpo de la petición contenga todos los valores descritos. Debe controlar errores de casteo. Utilice un patrón MVC con una capa de servicio que permita validar situaciones descritas antes de hacer la inserción en la BD. Si hay problemas con los datos, debe devolver un error 422, indicando cuál fue el campo que falló y la razón.

PUT api/productos/{id} – permite la actualización de un producto, con toda la información descrita anteriormente según el id indicado.

Debe validar que el cuerpo de la petición contenga todos los valores descritos. Debe controlar errores de casteo. Utilice un patrón MVC con una capa de servicio que permita validar situaciones descritas antes de hacer la inserción en la BD. Si hay problemas con los datos, debe devolver un error 422, indicando cuál fue el campo que falló y la razón. Si no encontró el id correspondiente, debe devolver un error 404.

PATCH api/productos/{id} – permite la modificación del inventario. Debe solicitar una operación que puede ser un string indicando que debe hacerse. Por ejemplo, si le envía una operación "agregar" debe sumarle a lo que tiene el inventario, el dato que envíe el usuario. Si es una operación "modificar" se actualiza la información con el nuevo dato que el usuario envíe. Si es una operación "restar" se le disminuye al inventario, lo que usuario envíe. El cuerpo de la petición tendrá la siguiente estructura:

```
{
        operation: "agregar",
        cantidad: 5
}
```

Por ejemplo, la operación agregar, si el inventario tiene 10 productos, le sumará 5. Quedando con unos 15 productos en inventario.

Si la operación es modificar, y el inventario tiene 10 productos, modificará la cantidad a 5, quedando el inventario con 5 productos.

Si la operación es restar, y el inventario tiene 8 productos, restará la cantidad de 5, quedando el inventario con 3 productos

La terminología relacionada a PATCH y PUT puede ser confusa. Lo que tiene que saber es que PATCH normalmente, no es idempotente y PUT si lo es. Esto quiere decir que PATCH en su lógica, solo permite la modificación de un registro específico. En el caso anterior, el campo de inventario solamente debe modificarse, y considerando las operaciones descritas. Este link les puede brindar más información de las diferencias entre ambos: https://www.it-swarm-es.com/es/json/rest-api-put-vs-patch-con-ejemplos-de-la-vida-real/1051132050/

Debe validar que el cuerpo de la petición contenga los valores descritos. Debe controlar errores de casteo. Utilice un patrón MVC con una capa de servicio que permita validar situaciones descritas antes de hacer la inserción en la BD. Si hay problemas con los datos, debe devolver un error 422, indicando cuál fue el campo que falló y la razón. Por ejemplo, en los casos que la operación no exista, debe devolver un 422. Si no encontró el id correspondiente, debe devolver un error 404.

Para cualquier otro error no identificado por la API, debe devolver un error 500.

II PARTE. API de categorías

GET : api/categorias – retorna una lista de objetos JSON con todos los datos de la tabla categorías

Si no hay datos en la base de datos, devuelva una lista vacía.

GET : api/categorías/{id} – retorna un objeto JSON correspondiente con los datos del id dado.

Si no encontró el registro indicado con ese id, devuelva un error 404, con el mensaje: Error la categoría no fue encontrada.

El API de GET no tendrá las opciones de mantenimiento. No se puede insertar ni modificar una categoría.

Para cualquier otro error no identificado por la API, debe devolver un error 500.

III. PARTE. Entrega de colección de pruebas y script de base de datos.

Haciendo uso del programa Postman, como lo vimos en clase, construya una colección que demuestre el funcionamiento de cada una de las API descritas. Debe entregar una colección de Postman con los requests indicados anteriormente. Este set de petciones debe contener lo que usted considere necesario para efectuar las pruebas. Como mínimo, se espera que contenga todos los puntos de enlace antes descritos en las dos secciones anteriores.

Debe además brindar un script de la base de datos, con datos de prueba que permitan verificar el funcionamiento de la API. **Nota Importante: Si el examen no cuenta con un script de base de datos, el mismo no funciona o tiene errores, o no contiene datos de prueba, su examen no será revisado por completo.**

IV PARTE. Seguridad - Puntaje extra.

La siguiente parte es para puntaje extra y es opcional. Implemente que los API de post, put y patch de productos, tengan seguridad. Lo idea es aplicar un endpoint de que devuelva un JWT validado por un usuario y contraseña en base de datos. Para eso debe tener una tabla de usuarios extra y debe autenticar al usuario para devolver un token.

Aspectos técnicos: Estos son APIs REST. Deben usar JAX-RS como lo vimos en clase. Si quieren usar un framework que les facilite la creación de los endpoints como spring, lo pueden hacer, pero deben asegurarse que el proyecto funcione con todas sus dependencias. Si el proyecto no funciona por dependencias, no será revisado.

Rúbrica de evaluación	
I. API de productos	**60pts**
1. Tiene una API get que devuelve una lista de productos.	5 pts

2. Tiene una API get que devuelve un único producto por su id	5 pts
3. Tiene una API post que permite la inserción de un producto	10 pts
4. Tiene una API put que permite la actualización de un producto	10 pts
5. Tiene una API patch que permite la actualización del inventario de un producto bajo las operaciones indicadas	15 pts
6. Se han realizado validaciones solicitadas en los get, put y patch que devuelven un 404 cuando algún producto no fue encontrado.	5 pts
7. Se controlan los errores con 422 para los post, put y patch de la aplicación.	5pts
8. Para cualquier otro error no controlado, se devuelve un error 500	5pts
II. **Parte. API de categorías**	**12 pts**
9. Tiene una API get que devuelve una lista de categorías.	5 pts
10. Tiene una API get que devuelve una única categoría por su id	5 pts
11. Se ha realizado validación en get que devuelven un 404 cuando alguna categoría no fue encontrada.	2 pts
III. Colección de pruebas y script	**28 pts**
12. Entrega una colección de pruebas que tiene todos los endpoints correspondientes para probar sus APIS	10 pts
13. Entrega un script de base de datos con datos de prueba	18 pts
IV Parte. Seguridad. Puntaje extra	**1.5%**
14. Realizar un API para validar un usuario y contraseña en BD y que genera un token	1%
15. Los endpoints de post, put y patch funcionan con la seguridad del JWT	0.5%
Total del examen	**100 pts**
Puntaje extra (sobre la nota del curso)	**1.5%**

Fin del examen final
Muchos éxitos