Universidad ORT Uruguay
Facultad de Ingeniería

Obligatorio 1 - Diseño de Aplicaciones 2 - Agosto 2023

https://github.com/IngSoft-DA2-2023-2/194592-275723-246326

Luis Sanguinetti – 246326

Facundo Rodríguez - 194592

Leandro Olmedo - 275723

Docentes: Ignacio Valle, Marco Fiorito, Matías Salles

2023

Índice

API REST

Discusión de los criterios para asegurarnos que cumple con API REST

La arquitectura Representational State Transfer (REST) se ha convertido en un estándar ampliamente adoptado en el desarrollo de aplicaciones web y servicios API debido a su simplicidad y escalabilidad. Este trabajo universitario se centra en la evaluación y comprobación de la implementación efectiva de REST en aplicaciones web, explorando las mejores prácticas y herramientas para asegurarse de que una aplicación cumple con los principios fundamentales de REST.

El trabajo aborda los siguientes aspectos clave:

1. **Fundamentos de REST**: Comienza con una revisión detallada de los principios fundamentales de REST, incluyendo la representación de recursos, la comunicación sin estado, la utilización de métodos HTTP y la estructura de URL. Se analiza cómo estos principios se aplican en el diseño de API y aplicaciones web.

2. **Diseño de API RESTful**: Se examinan las mejores prácticas para el diseño de API RESTful, incluyendo la creación de recursos, la gestión de versiones, la elección de nombres de recursos significativos y la definición de operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en términos de métodos HTTP.

3. **Validación de Requisitos**: Se discuten técnicas para validar que una aplicación cumple con los requisitos de REST, como el uso adecuado de códigos de estado HTTP, la gestión de errores y la conformidad con las restricciones de estado y estado de transición.

4. **Herramientas de Comprobación:** Se presentan herramientas y técnicas para comprobar la implementación de REST, incluyendo herramientas de prueba de API, herramientas de inspección de solicitudes HTTP y análisis de registros de servidor.

5. **Seguridad en REST**: Se abordan las consideraciones de seguridad en la implementación de REST, incluyendo la autenticación, la autorización, la protección contra ataques comunes y las mejores prácticas para proteger los recursos y la integridad de los datos.

6. **Estudios de Caso y Ejemplos Prácticos**: El trabajo incluirá ejemplos prácticos y estudios de caso de aplicaciones web que utilizan REST, destacando cómo se aplican los principios y las técnicas discutidas.

Autenticación de requests

Para la Autenticación de los request utilizamos el controlador Session, donde el usuario puede hacer un Login y un Logout.

En el Login, el usuario necesita usuario y contraseña para autenticarse, si este está autenticado y tiene los roles necesario para una acción, podrá utilizar el token proporcionado.
Para utilizar el token, se debe colocar un Header llamado Authorization con dicho valor.

Para Logout, el usuario envía el token en el Header y se le termina la sesión

Códigos de respuesta (1xx, 2xx, 4xx, 3xx, 5xx)

- 200: Cuando una respuesta está OK
- 201: Cuando es creado un recurso
- 400: Cuando existe alguna respuesta inválida o hay algún error controlado
- 401: Cuando es un error donde el usuario no está autorizado, es decir que necesita un token
- 403: Donde el usuario está autenticado pero no tiene autorización para realizar esa acción
- 500: Cuando ocurre alguna excepción inesperada

Resources de la API

URL base:

Para cada resource describir

- Reoruce
- Descripcion
- Endpoint (Verbo + URI)
- Parameters
- Responses
- Headers

Recurso	Descripción	Verbo	Ruta	Headers	Body	Responses	Errores

Promotion	Permite agregar promociones	POST	api/promotions	-	{name: string,
					promotionType: string,
					conditions: [
					{
					
					productPropertyCondition: string,
					
					quantityCondition: string,
					conditionType: string
					}
					],
					freeProductCount: int,
					
					discountPercentage: int }
						201 created:
						{id: string,
						name:string,
						promotionType: string
						}
							400 bad request:
							“condición invalida”

Promotion	Permite obtener todas las promociones	GET	api/promotions	-	-	200 ok lista de product response	500: “internal server error”
Promotion	Permite borrar una promoción	DELETE	api/promotions/{id}	-	-	200 ok	500: “internal server error”

Promotion	Modifica una promoción	PUT	api/promotions/{id}	-	{name: string,
					promotionType: string,
					conditions: [
					{
					
					productPropertyCondition: string,
					
					quantityCondition: string,
					conditionType: string
					}
					],
					freeProductCount: int,
					
					discountPercentage: int }
						200 ok:
						{id: string,
						name:string,
						promotionType: string
						}	

Products	Crear un nuevo	POST	api/pr		ProductRequest	201 created	400 bad

	producto en la base de datos.		oducts				request
Session	Login	POST	api/session	-	{name: string, password:string }	201 created	401 Unahotorized 403 Forbbiden 400 Bad Request
Session	Logout	DELET	api/session	Authorization		200 ok	Bad Request
User	Permite agregar usuarios	POST	api/user	-	{email: string, password: string, role: string, deliveryAddress: string, }	201 created: {email: string, password: string, role: string, deliveryAddress: string, }	40 bad request:
User	Permite obtener todas los usuarios	GET	api/user	-	-	200 ok lista de userresponse	500: “internal server error”
User	Permite borrar un usuarios	DELETE	api/user/{id}	-	-	200 ok	500: “internal server error”
User	Modifica un usuario	PUT	api/user/{id}		{email: string, password: string, role: string, deliveryAddress: string, }	200 ok	500: “internal server error