Stella Steinfeld (229398)
Eugenia Matto (227056)
Santiago Vairo(244023)

Diseño de aplicaciones 2

Evidencia de Diseño y Especificación de la API

https://github.com/IngSoft-DA2/Steinfeld-Vairo-Matto

Discusión sobre los criterios seguidos para asegurar que la API cumple con los principios REST

La API implementada sigue correctamente los principios de la arquitectura REST. Uno de los puntos más destacados es el uso adecuado de los verbos HTTP, como GET, POST, PATCH y DELETE, para realizar las distintas operaciones sobre los recursos. Por ejemplo, GET se usa para obtener información de los recursos, como en los endpoints /api/accounts o /api/companies, mientras que POST se emplea para crear o agregar nuevos recursos, como en /api/homes o /api/permissions. Esto es clave para cumplir con el principio REST de usar los métodos HTTP estándar de manera apropiada.

Además, los recursos son tratados de forma uniforme a través de rutas bien estructuradas. La API sigue el estándar de nombrar los endpoints en plural y de manera descriptiva, lo que hace que sea intuitivo comprender qué tipo de datos se manejan en cada ruta. La representación de los recursos es clara, utilizando URLs que indican los recursos que se están manipulando, como homes, devices, members, entre otros.

Otro punto a favor es que la API es stateless, es decir, cada solicitud es independiente y contiene toda la información necesaria para ser procesada sin depender de estados previos en el servidor. Esto se logra con el uso de tokens en los headers de las solicitudes, lo que permite que el servidor autentique cada request de forma aislada.

Por último, los códigos de respuesta están bien implementados. El uso de 200 OK para indicar éxito y 401 Unauthorized para problemas de autenticación es consistente, lo que ayuda a los clientes a entender el estado de la solicitud de manera clara.

Descripción del mecanismo de autenticación de requests

El mecanismo de autenticación de esta API utiliza tokens basados en GUID (Global Unique Identifier). Estos tokens se envían en el header de cada request bajo el campo Authorization. Al ser GUID, garantizan unicidad a nivel global, lo que asegura que cada token sea único y no tenga colisiones. Este tipo de token es más ligero y fácil de validar que otros formatos como JWT, lo que permite una autenticación rápida y eficiente. El servidor verifica el GUID en cada solicitud y, si es válido, permite el acceso; de lo contrario, responde con un 401 Unauthorized.

Este enfoque garantiza un mecanismo de autenticación seguro y eficiente, permitiendo que la API mantenga su arquitectura sin estado y sea fácil de escalar.

Recurso	Verbo	Ruta	Headers	Body	Response	Errores	Descripción

Account	GET	/api/accounts	Authorization: {{Token}}	-	200 OK
					{
					Item[
					{
					“name”:
					“Surname”:
					“Fullname”:
					“Role”:
					“CreatedAt”:
					
					}],
					“Totalcount”:
					“Pagenumber”:
					“Pagesize”:
					
					}
						401 si el token no es correcto	Obtiene la lista de cuentas.

Administrator	DELETE	/api/administrators?accountId={{id}}	Authorization: {{Token}}	-	200 OK
					{
					“message”:
					Deleted “Admin account”: “GUID”
					}
						401 si el token no es correcto
						
						500 internal error si se intenta borrar una cuenta que no es administrador o un id invalido ”there was an error when processing the request:
							Elimina un administrador.

						Administrator not found”	

Administrator	POST	/api/administrators	Authorization: {{Token}}	{{ "Name"}}: "{{name}}", "Surname": "{{surname}}", "Email": "{{email}}", "Password": "{{password}}" }	200 OK
					{
					“token”: token
					}
						401 si el token no es correcto
						
						403 Forbidden: message: “Missing permission createadminaccount-administrator”
						
						500: internal server error: si se intenta registrar un administrador con email existente: “there was an error when processing the request: user with email xx@xx already exists.”
						400 Bad request: si no se pasan datos en el body: “invalid request”
							Registra un administrador.

Company	GET	/api/companies	Authorization: {{Token}}	-	200 OK	401 si el token no es correcto	Obtiene la lista de compañías.

Company	POST	/api/companies	Authorization: {{Token}}	{ "name": "{{name}}", "logoUrl": "{{url}}", "rut": "{{rut}}" }	200 OK
					{
					“id”:
					}
						401 si el token no es correcto
						400: “message: el request es invalido”
						500: “there was an error when processing the request. already the owner of a company.”
							Registra una nueva compañía.

CompanyOwner	POST	/api/company-owners	Authorization: {{Token}}	{ "Name": "{{name}}", "Surname": "{{surname}}", "Email": "{{email}}", "Password": "{{pass}}" }	200 OK
					{
					“token”: token
					}
						401 si el token no es correcto	Registra un dueño de compañía.

Device	GET	/api/devices?name={{DeviceName}}&model={{DeviceModel}}&pageNumber={{pageNumber}}&pageSize={{pageSize}}	Authorization: {{Token}}	-	200 OK	401 si el token no es correcto	Obtiene dispositivos según nombre y modelo.
Device	GET	/api/homes/{{HomeId}}/devices	Authorization: {{Token}}	-	200 OK	401 si el token no es correcto	Obtiene los dispositivos

							asociados a un hogar.
Device	POST	/api/homes/{{HomeId}}/devices	Authorization: {{Token}}	{ "DeviceId":"f64f4cbc-a8b0-4e71-c75c-08dce4e1000f" }	200 OK	401 si el token no es correcto	Asocia un dispositivo al hogar.
DeviceTypes	GET	/api/device-types	Authorization: {{Token}}	-	200 OK	401 si el token no es correcto	Obtiene tipos de dispositivos.
Home	POST	/api/homes	Authorization: {{Token}}	{ "Street": "casa mia", "StreetNumber": "232", "Ubication": { "Lat": 66, "Lon": -69 }, "MaxMembers": 2 }	200 OK	401 si el token no es correcto	Permite registrar un hogar.
HomeOwner	POST	/api/home-owners	Authorization: {{Token}}	{ "Name": "{{name}}", "Surname": "{{surname}}", "Email": "{{email}}", "Password": "{{pass}}", "ProfilePhoto": "{{ProfilePhoto}}" }	200 OK	401 si el token no es correcto	Registra un dueño de hogar.
Member	GET	/api/members?homeId={{HomeId}}	Authorization: {{Token}}	-	200 OK	401 si el token no es correcto	Obtiene los miembros

							asociados a un hogar.
Member	PATCH	/api/members/{{HomeId}}/notification	Authorization: {{Token}}	[{{member}}]	200 OK	401 si el token no es correcto	Actualiza las notificaciones de los miembros.
Member	POST	/api/members/{{HomeId}}	Authorization: {{Token}}	{ "memberEmail": "member@gmail.com", "homeId": "{{homeID}}" }	200 OK	401 si el token no es correcto	Añade un miembro a un hogar.
Notification	GET	/api/notifications	Authorization: {{Token}}	-	200 OK	401 si el token no es correcto	Obtiene notificaciones.
Permission	DELETE	/api/permissions/users?email={{Email}}	Authorization: {{Token}}	{ "Name":"newhome-home" }	200 OK	401 si el token no es correcto	Revoca permisos de un usuario.
Permission	POST	/api/permissions?email={{Email}}	Authorization: {{Token}}	{ "Name":"getmembers-members" }	200 OK	401 si el token no es correcto	Otorga permisos a un usuario.

Permission	POST	/api/permissions/users?email={{Email}}	Authorization: {{Token}}	{ "Name":"getdevicetypes-devicetype" }	200 OK	401 si el token no es correcto	Asigna permisos a un usuario para diversos módulos.
WindowSensor	POST	/api/sensors	Authorization: {{Token}}	{ "Name": "Window Sensor mega", "ModelNumber": "KS-133", "Description": "A rapid window sensor for home automation.", "Photos": ["https://megaK.com/photo1.jpg", "https://megaK.com/photo2.jpg"] }	200 OK	401 si el token no es correcto	Agrega un sensor de ventana
WindowSensor	POST	/api/sensors/open?hardware_id={{id}}	Authorization: {{Token}}	-	200 OK	401 si el token no es correcto	Indica que se abrio la ventana

WindowSensor	POST	/api/sensors/close?hardware_id={{id}}	Authorization: {{Token}}	-	200 OK	401 si el token no es correcto	Indica que se cerró la ventana

SecurityCam	POST	/api/cameras	Authorization: {{Token}}	{
				"Name": "",
				"ModelNumber": "",
				"Description": "",
				"Photos": ["", ""],
				"UseTypes": [
				"",
				"",
				],
				"SupportedActions": [
				"",
				"",
				]
				}
					200 OK	401 si el token no es correcto	Agrega una cámara

SecurityCam	POST	/api/cameras/motion-detection?hardware_id={{id}}	Authorization: {{Token}}	-	200 OK	401 si el token no es correcto	detecta movimiento motion
SecurityCam	POST	/api/cameras/person-detection?hardware_id={{id}}	Authorization: {{Token}}	-	200 OK	401 si el token no es correcto	detecta movimiento de persona

Login	POST	/api/user/login		{
				"Email":"",
				"Password":""
				}
					200 OK
					“token” = “”
						400 Bad request	hace login de un usuario