ID	Método	Endpoint	Descripción	Request Body	Headers	Expected Status	Expected Response	Actual Status	Actual Response	Resultado
TC01	GET	/users/1	Obtener los datos del usuario con ID 1			200 OK	Debe retornar JSON con nombre, email y demás datos del usuario con ID 1	200 OK	id: 1, name: Leanne Graham, email: Sincere@april.biz	OK
TC02	GET	/users/2	Obtener los datos del usuario con ID 2			200 OK	Debe retornar JSON con nombre, email y demás datos del usuario con ID 2	200 OK	id: 2, name: Ervin Howell, email: Shanna@melissa.tv	OK
TC03	POST	/users	Crear usuario nuevo en el servidor	{ "name": "Luis Tester",		201	Debe retornar un JSON con los datos enviados (name, username, email) y un ID generado por el servidor.	201	"name": "Luis Tester", "username": "luisqa", "email": "luis@example.com",	OK
TC04	GET	/Users/9999	Consultar usuarios inexistentes			404 Not Found	El codigo de estado debe ser 404 Not Found	404 Not Found	404 Not Found	OK
TC05	POST	/users	Crear un usuario con campos vacios			400 bad request	Debe devolver un JSON indicando: {"error": "Missing required fields"}	201 Created	{ "id": 11 }	Falla, la API permite crear usuario con campo vacio.
TC06	DELETE	/users/2	Eliminar al usuario con ID 2			204 No Content	El codigo de estado deberia ser 204 No Content	200 OK	"{}"	Incongruencia - la API devuelve 200 OK en lugar de 204 No Content
TC07	PUT	posts/1	Actualizar un Post con Id 1	{ "id": 1, "title": "Nuevo título", "body": "Contenido actualizado", "userId": 1 }	Content-Type: application/json	200 OK	{ "id": 1, "title": "Nuevo título", "body": "Contenido actualizado", "userId": 1 }	200 OK		OK

Ejecutadas con Postman | Analizadas manualmente | Probada con una API publica