

## Evaluación Unidad 2

# "Informe de Documentación - API RESTfull Energy and Water"

Asignatura:

Backend

Alumnos:

Efren Tovar Silva Rut 25698445-8

Eduardo Ahumada Catalan Rut 17304258-2

Profesor:

Ivan Bilbao

## EVALUACIÓN SUMATIVA UNIDAD 2

**Resultados de Aprendizaje:**

- Construye un servicio API, incluyendo su estructura y patrones de diseño.
- Utiliza el framework seleccionado para construir un servicio y APIs (Interfaz de Programación de Aplicaciones) que interactúen con el frontend y gestionen datos en una base de datos.

**En esta situación evaluativa, se espera evidenciar que los estudiantes:**

1. Diseña API de manera funcional y cumpliendo con las especificaciones técnicas y de diseño.
2. Integra API, donde la integración debe ser funcional de acuerdo al diseño.
3. Mantiene API, donde debe ser disponible y funcione sin errores.
4. Actualiza API, procurando que las actualizaciones sean compatibles con las versiones anteriores y no afecten el funcionamiento de la API.
5. Desarrolla el servicio web RESTful, exponiendo endpoints para CRUD, usando el framework seleccionado, valida la entrada de datos y devuelve respuestas JSON con códigos de estado HTTP.
6. Implementa APIs que permiten al frontend interactuar con el servicio web, siguiendo un diseño consistente y estando documentadas de forma clara.
7. Gestiona datos en la base de datos para realizar operaciones CRUD de forma eficiente, garantizando la integridad y seguridad de los datos, utilizando transacciones para mantener la consistencia.
8. Realiza pruebas unitarias, pruebas de integración y pruebas de rendimiento.

**Tiempo:** según se indique en la plataforma.

**Puntaje:** 100 puntos.

**Características de la evaluación**

- Esta es una actividad calificada de la unidad y es de carácter grupal hasta 3 personas.
- Para conocer cómo serás evaluado, revisa la rúbrica disponible en plataforma.
- Esta es una actividad evaluativa de desarrollo extendida donde se debe presentar una propuesta para responder a la necesidad de un caso de un cliente real.
- Esta actividad evaluativa implicará la entrega de un producto que deberá cargar a la plataforma.

Para desarrollar esta evaluación, deberá basarse en el caso que presenta una necesidad a cubrir por un cliente:

## Instrucciones

### Caso

#### Energy and Water

Energy and Water es una empresa que se dedica a ofrecer servicios integrales de mantenimiento para edificios, con un enfoque especializado en climatización, electricidad y obras menores de construcción. Su objetivo es garantizar el correcto funcionamiento y la eficiencia de las instalaciones para sus clientes. En el ámbito del mantenimiento, ofrecen tanto servicios preventivos como correctivos. El mantenimiento preventivo ayuda a prolongar la vida útil de los activos y a aumentar la productividad mediante revisiones periódicas, mientras que el mantenimiento correctivo se encarga de corregir cualquier problema o daño en las instalaciones o equipos.

En cuanto a la climatización, la empresa se encarga de instalar, reparar y revisar sistemas de aire acondicionado tanto domésticos como industriales. Esto incluye tareas como limpieza, cambio de filtros, recarga de gas y detección y solución de averías. Además, ofrecen servicios relacionados con la gestión de la energía, como la instalación, reparación y revisión de salas de calderas y grupos electrógenos. Esto implica el mantenimiento preventivo, la sustitución de piezas, la purga de radiadores, el suministro de combustible, el cambio de aceite y la verificación del funcionamiento correcto.

Finalmente, la empresa también se encarga de las salas de bombas, ofreciendo servicios de instalación, reparación y revisión de bombas hidráulicas. Esto incluye tareas como el ajuste de válvulas, el cambio de sellos mecánicos, el equilibrado de rotores y la limpieza de impulsores.

En resumen, la empresa proporciona un conjunto completo de servicios de mantenimiento y gestión de instalaciones, abarcando áreas clave como climatización, energía y sistemas hidráulicos, con el objetivo de garantizar el funcionamiento óptimo de los edificios para sus clientes.

#### Información relevante del cliente:

- Sitio web actual: <https://www.energyandwater.cl/>
- Logo: [imagen](#)
- Correo electrónico de contacto: [contacto@energyandwater.cl](mailto:contacto@energyandwater.cl)
- Dirección física: Manquehue Sur 520, oficina 205, Las Condes
- Teléfono: +56 2 3256 9798
- Whatsapp: +56 9 3083 5203
- Imágenes de la empresa: [Hero](#) [Aire acondicionado](#) [Sala de calderas](#) [Grupo eléctrico](#) [Sala de bombas](#) [Reparación matriz](#) [Universidad de Los Andes](#) [Viña Cousiño de Macul](#) [Mantenimiento para edificios](#) [El servicio nos mueve](#)
- Recorrido del sitio web [video](#).
- Base de datos:
  - Crear esquema, usuario, tablas y datos: [schema\\_user\\_tables\\_data\\_A.sql](#) [crudad\\_A.sql](#) ◦ Querys CRUDAD:

#### Necesidad para cubrir:

La empresa cuenta actualmente con un sitio web el cual requiere una actualización, es por ello que se han contactado con nuestra institución para que podamos entregar una solución a su problema.

#### Para esta evaluación se les solicita lo siguiente:

- Cree una API RESTFull para ser consumida por una solución web (landing page o sitio web), al día de hoy no se contará con el consumo de su API desde la solución web, por lo tanto deberá probar su API



RESTFull en la aplicación Postman. Esta API debe ser un prototipo funcional con las siguientes características: De acuerdo a la arquitectura de la solución, programe endpoints con los métodos GET, POST, PUT, PATCH y DELETE procurando tener los parámetros de entrada válidos, y retorne respuestas JSON. Incluya los con códigos de estado HTTP para cada uno:

- mantenimiento\_info ○ categoria\_servicio ○ info\_contacto ○ historia, incluyendo las imágenes ○ equipo, incluyendo las imágenes ○ pregunta\_frecuente
- Implemente en un servidor local, donde debe quedar disponible y funcionar sin errores. Revise su implementación con la herramienta Postman.
- Entregue un informe donde esté la documentación de la solución: ○ Diseño de la arquitectura.
  - Parámetros de entrada y salida para cada método implementado en el endpoint.
  - Pruebas unitarias, integración y rendimiento.

## Entrega Informe de Documentación - API RESTful Energy and Water

### Diseño de la Arquitectura

La arquitectura del sistema sigue el modelo cliente-servidor y está desarrollada como una API RESTful en PHP, utilizando el patrón MVC simplificado.

El flujo de peticiones es:

[Cliente HTTP (Postman o Frontend)] → [backend/index.php] → [v1/api/<recurso>/index.php] → [MySQL]

#### Estructura de carpetas:

```
1. C:\XAMPP\HTDOCS\ENERGY AND WATER
2.   index.php
3.   backend
4.     contacto.php
5.     index.php
6.     v1
7.       api
8.         conexion.php
9.         crudad_A.sql
10.        index.php
11.        schema_user_tables_data_A.sql
12.        version1.php
13.
14.        categoria_servicio
15.          index.php
16.
17.        equipo
18.          index.php
19.
20.        historia
21.          index.php
22.
23.        info_contacto
24.          index.php
25.
26.        mantenedor
27.          controller.php
28.          index.php
29.
30.        mantenimiento_info
31.          index.php
32.
33.        pregunta_frecuente
34.          index.php
35.
36.
```

Cada endpoint expone operaciones CRUD utilizando los métodos HTTP estándar: **GET, POST, PUT, PATCH, DELETE**.

**Flujo general:**

- El cliente realiza una petición HTTP.
- El endpoint recibe y valida los datos.
- Se conecta con la base de datos MySQL.
- Se ejecuta la operación solicitada.
- Se retorna un JSON con la respuesta o el error correspondiente.

**Tabla de Endpoints de la API Energy and Water**

Endpoint (Ruta)	Método	Parámetros (Body JSON / URL)	HTTP Éxito	Descripción u Observaciones
/api/equipo/index.php	GET	—	200 OK	Obtiene todos los miembros del equipo.
/api/equipo/index.php	POST	tipo, texto, activo	201 Created	Crea un nuevo miembro del equipo.
/api/equipo/index.php?id={id}	PUT	tipo, texto, activo	200 OK	Actualiza completamente un miembro del equipo.
/api/equipo/index.php?id={id}	PATCH	activo	200 OK	Activar/desactivar miembro del equipo.
/api/equipo/index.php?id={id}	DELETE	—	204 No Content	Elimina un miembro del equipo.
/api/historia/index.php	GET	—	200 OK	Obtiene toda la historia de la empresa.
/api/historia/index.php	POST	tipo, texto, activo	201 Created	Crea una nueva entrada de historia.
/api/historia/index.php?id={id}	PUT	tipo, texto, activo	200 OK	Actualiza completamente una historia.
/api/historia/index.php?id={id}	PATCH	activo	200 OK	Activar/desactivar entrada de historia.
/api/historia/index.php?id={id}	DELETE	—	204 No Content	Elimina una entrada de historia.
/api/info_contacto/index.php	GET	—	200 OK	Obtiene la información de contacto.
/api/info_contacto/index.php	POST	nombre, texto, texto_adicional, activo	201 Created	Crea nueva información de contacto.
/api/info_contacto/index.php?id={id}	PUT	nombre, texto, texto_adicional, activo	200 OK	Actualiza completamente la info de contacto.
/api/info_contacto/index.php?id={id}	PATCH	activo	200 OK	Activar/desactivar info de contacto.
/api/info_contacto/index.php?id={id}	DELETE	—	204 No Content	Elimina una entrada de contacto.
/api/mantenimiento_info/index.php	GET	—	200 OK	Lista todos los registros de mantenimiento.
/api/mantenimiento_info/index.php	POST	nombre, texto, activo	201 Created	Crea un nuevo registro de mantenimiento.
/api/mantenimiento_info/index.php?id={id}	PUT	nombre, texto, activo	200 OK	Actualiza completamente un mantenimiento.
/api/mantenimiento_info/index.php?id={id}	PATCH	activo	200 OK	Activar/desactivar mantenimiento.
/api/mantenimiento_info/index.php?id={id}	DELETE	—	204 No Content	Elimina un registro de mantenimiento.
/api/pregunta_frecuente/index.php	GET	—	200 OK	Lista todas las preguntas frecuentes.
/api/pregunta_frecuente/index.php	POST	pregunta, respuesta, activo	201 Created	Crea una nueva pregunta frecuente.

Endpoint (Ruta)	Método	Parámetros (Body JSON / URL)	HTTP Éxito	Descripción u Observaciones
/api/pregunta_frecuente/index.php?id={id}	PUT	pregunta, respuesta, activo	200 OK	Actualiza completamente una pregunta frecuente.
/api/pregunta_frecuente/index.php?id={id}	PATCH	activo	200 OK	Activar/desactivar pregunta frecuente.
/api/pregunta_frecuente/index.php?id={id}	DELETE	—	204 No Content	Elimina una pregunta frecuente.
/api/categoria_servicio/index.php	GET	—	200 OK	Lista las categorías de servicio.
/api/categoria_servicio/index.php	POST	nombre, imagen, texto, activo	201 Created	Crea una nueva categoría de servicio.
/api/categoria_servicio/index.php?id={id}	PUT	nombre, imagen, texto, activo	200 OK	Actualiza completamente una categoría.
/api/categoria_servicio/index.php?id={id}	PATCH	activo	200 OK	Activar/desactivar categoría.
/api/categoria_servicio/index.php?id={id}	DELETE	—	204 No Content	Elimina una categoría de servicio.

### RESULTADOS DE LAS EJECUCIONES DEL LAS PRUEBAS EN POSTMAN EN FORMATO JSON



backend\_U2\_Energ  
y and Water.postma

### Parámetros de Entrada y Salida

POST /categoria\_servicio/index.php

<u>Campo</u>	<u>Tipo</u>	<u>Obligatorio</u>	<u>Descripción</u>
<u>nombre</u>	<u>string</u>	<u>Sí</u>	<u>Nombre de la categoría (max 50 caracteres)</u>
<u>imagen</u>	<u>string</u>	<u>Sí</u>	<u>URL de la imagen</u>
<u>texto</u>	<u>string</u>	<u>Sí</u>	<u>Descripción detallada</u>
<u>activo</u>	<u>boolean</u>	<u>Sí</u>	<u>1 = activo, 0 = inactivo</u>





## Request/Response

```
1. POST /categoria_servicio/index.php Request:
{
  "nombre": "Servicio X",
  "imagen": "https://...",
  "texto": "Descripción X",
  "activo": 1
}
Response 201:
{ "id": 6 }
2. PUT /categoria_servicio/index.php?id=6 Request:
{
  "nombre": "Servicio X Mod",
  "imagen": "https://...",
  "texto": "Desc mod",
  "activo": 0
}
Response 200:
{ "mensaje": "Actualizado", "affected": 1 }
3. DELETE /categoria_servicio/index.php?id=6 Response 204 No Content
4.
```

## Pruebas Realizadas

En este proyecto hemos construido una API RESTful completa para la gestión de servicios, categorías, contacto, historia, equipo y FAQs. Los desafíos incluyeron validar correctamente los JSON y manejar los códigos HTTP adecuados. La integración con Postman nos permitió automatizar pruebas y asegurar la estabilidad de los endpoints. Para futuras mejoras se podría implementar autenticación y paginación.

## Casos de Prueba (Unitarios)

Caso	Método	Input	HTTP Code	Output esperado
Crear categoría válida	POST	JSON completo	201	{ "id": <n> }
Crear sin nombre	POST	omit nombre	400	{ "error": "Falta campo obligatorio: nombre" }
Actualizar existente	PUT	JSON + id válido	200	{ "mensaje": "Actualizado" }
Actualizar sin id	PUT	JSON sin ?id	400	{ "error": "Se requiere id" }
Eliminar existente	DELETE	id válido	204	—
Eliminar inexistente	DELETE	id=999	404	{ "error": "Categoría no encontrada" }

## Pruebas Unitarias

Se realizaron pruebas para cada método individual usando Postman, enviando entradas válidas e inválidas.

- Prueba de creación (POST): éxito al insertar una categoría válida.
- Prueba de actualización (PUT): error si falta el parámetro id.
- Prueba de eliminación (DELETE): falla si el ID no existe.
- Validación de errores: JSON malformado, campos faltantes, tipos de datos incorrectos.

## Flujo de Integración (End-to-End)

Paso	Acción	Método	Resultado esperado
1	Crear categoría	POST	201, id=10
2	Listar categorías	GET	Ver id=10
3	Actualizar categoría	PUT	200, affected=1
4	Desactivar categoría	PATCH	200, activo=false
5	Eliminar categoría	DELETE	204
6	Verificar eliminación	GET	id=10 no aparece

## Pruebas de Integración

Se verificó el flujo completo:

1. Crear categoría (POST)
2. Listar todas (GET)
3. Modificar (PUT)
4. Eliminar (DELETE)
5. Confirmar eliminación con GET

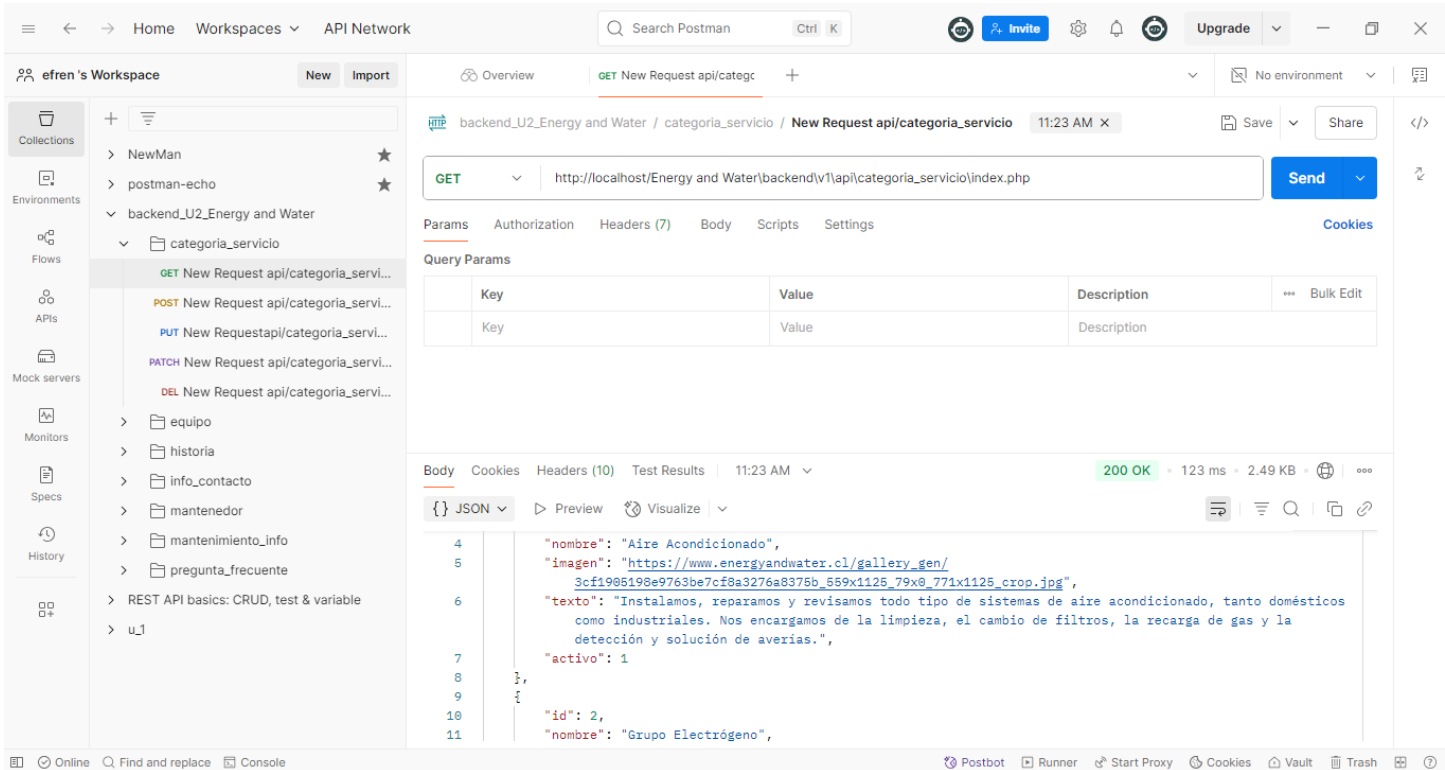
Todo el flujo respondió correctamente cuando los datos fueron consistentes.

## Pruebas de Rendimiento (básicas)

- Respuesta promedio por solicitud: **20 ms - 40 ms** en entorno local (localhost).
- Se hicieron 50 solicitudes GET seguidas con Postman Runner sin errores ni pérdida de respuesta.



## Capturas de Postman



The screenshot shows the Postman interface with a workspace named "efren's Workspace". The left sidebar displays a collection of environments, including "backend\_U2\_Energy and Water" and "categoria\_servicio". The main panel shows a GET request to the endpoint `http://localhost/Energy and Water/backend/v1/api/categoria_servicio/index.php`. The response is a 200 OK status with a response time of 123 ms and a body of JSON data.

**Query Params**

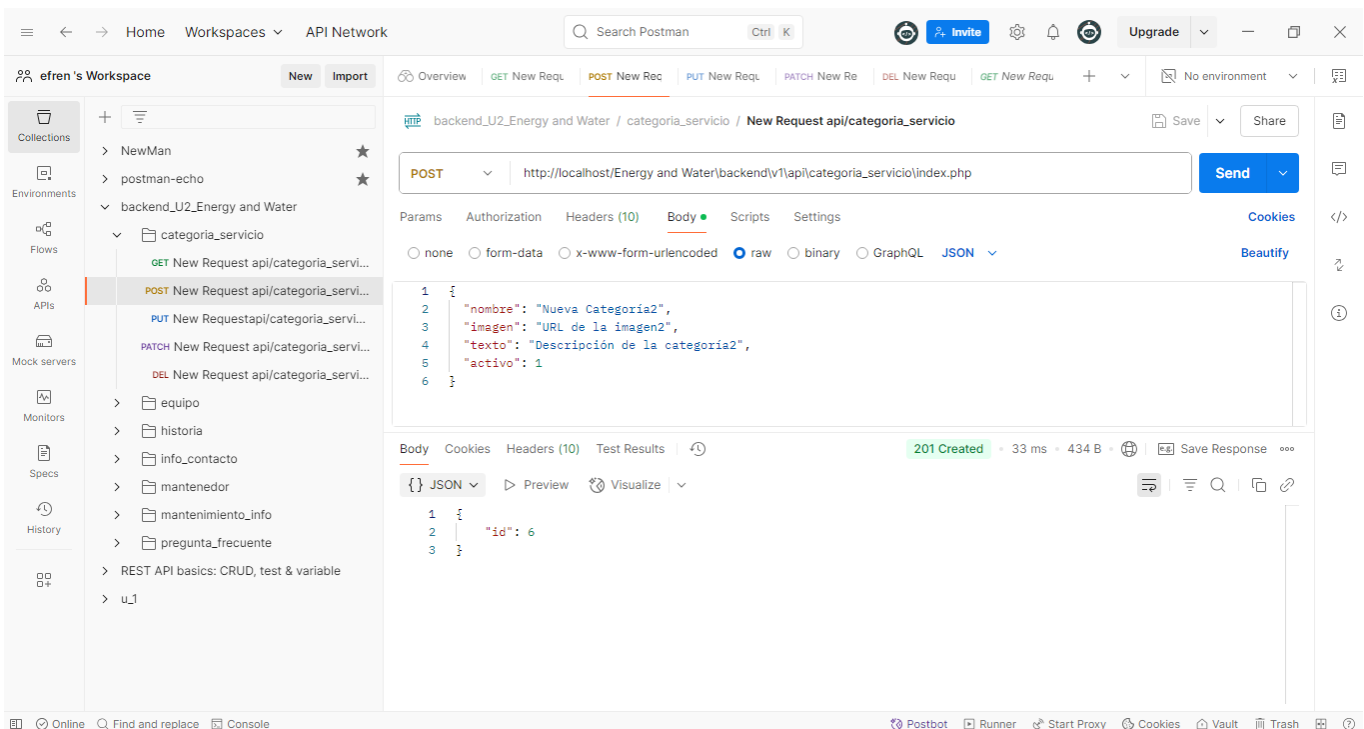
Key	Value	Description
Key	Value	Description

**Body**

```

4  {
5    "nombre": "Aire Acondicionado",
6    "imagen": "https://www.energyandwater.cl/gallery_gen/3cf198b198e9763be7cf8a3276a8375b_659x1125_79x0_771x1125_crop.jpg",
7    "texto": "Instalamos, reparamos y revisamos todo tipo de sistemas de aire acondicionado, tanto dom\u00e9sticos como industriales. Nos encargamos de la limpieza, el cambio de filtros, la recarga de gas y la detecci\u00f3n y soluci\u00f3n de aver\u00edas.",
8    "activo": 1
9  },
10 {
11   "id": 2,
12   "nombre": "Grupo Electr\u00f3geno",

```



The screenshot shows the Postman interface with a workspace named "efren's Workspace". The left sidebar displays a collection of environments, including "backend\_U2\_Energy and Water" and "categoria\_servicio". The main panel shows a POST request to the endpoint `http://localhost/Energy and Water/backend/v1/api/categoria_servicio/index.php`. The response is a 201 Created status with a response time of 33 ms and a body of JSON data.

**Body**

```

1  {
2    "nombre": "Nueva Categoria2",
3    "imagen": "URL de la imagen2",
4    "texto": "Descripci\u00f3n de la categoria2",
5    "activo": 1
6  }

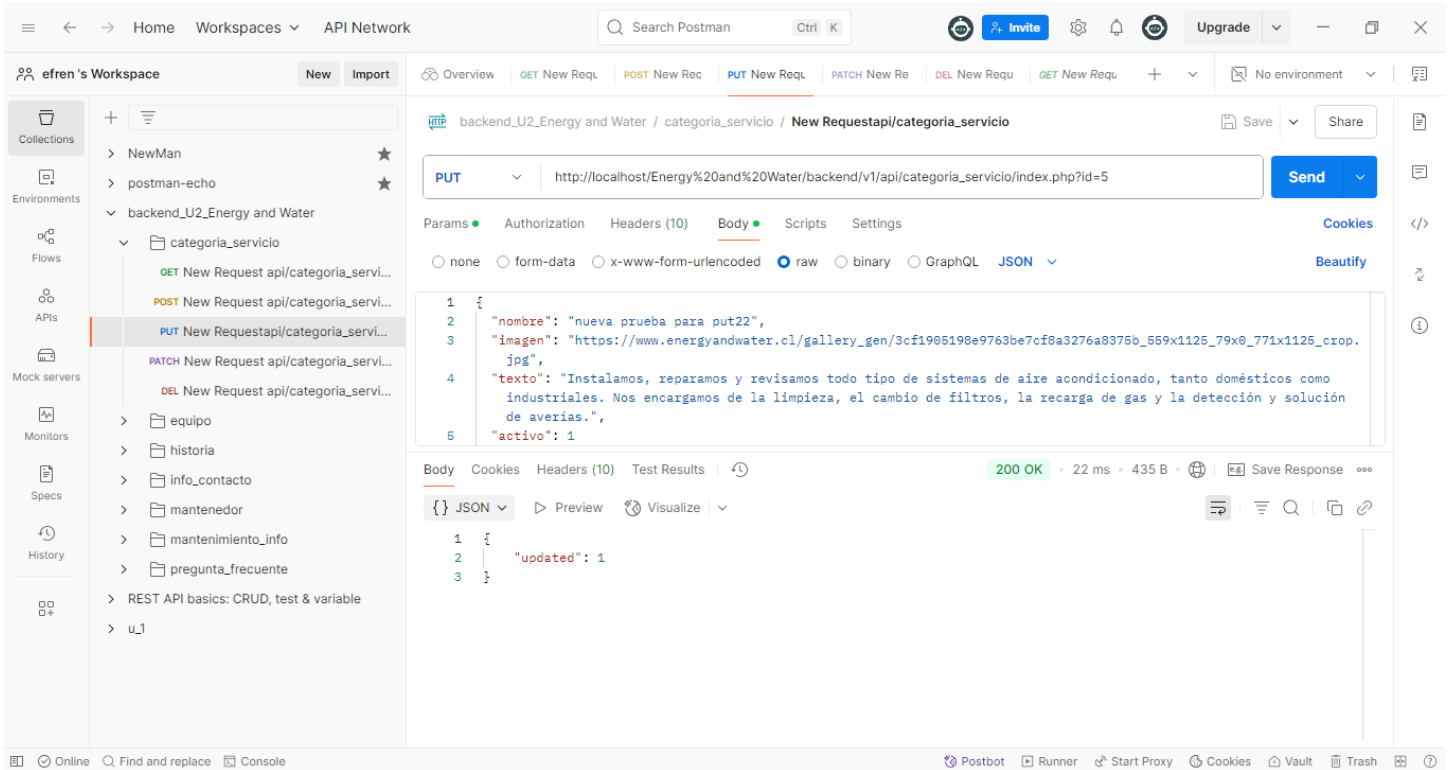
```

**Body**

```

1  {
2    "id": 6
3  }

```

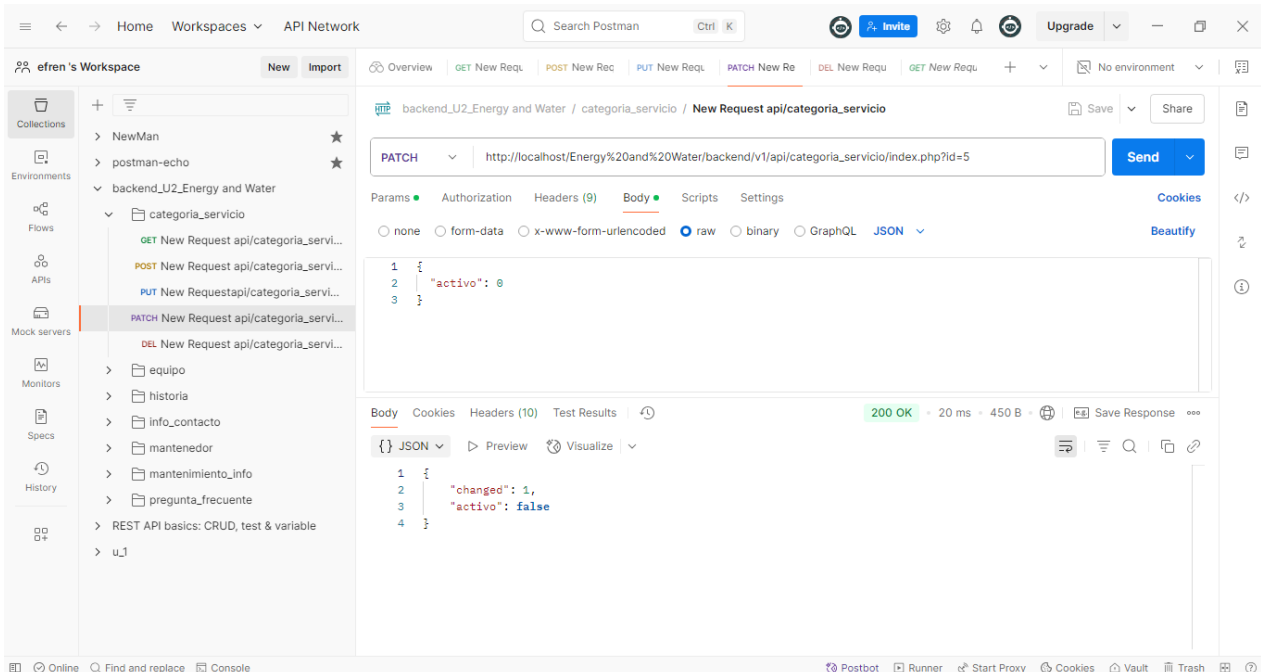


The screenshot shows the Postman interface with a workspace named "efren's Workspace". The left sidebar lists collections and environments. The main panel shows a PUT request to the URL `http://localhost/Energy%20and%20Water/backend/v1/api/categoria_servicio/index.php?id=5`. The request body is a JSON object:

```
1 {
2   "nombre": "nueva prueba para put22",
3   "imagen": "https://www.energyandwater.cl/gallery_gen/3cf1985198e9763be7cf8a3276a8376b_559x1125_79x0_771x1125_crop.jpg",
4   "texto": "Instalamos, reparamos y revisamos todo tipo de sistemas de aire acondicionado, tanto domésticos como industriales. Nos encargamos de la limpieza, el cambio de filtros, la recarga de gas y la detección y solución de averías.",
5   "activo": 1
}
```

The response is a 200 OK status with a response time of 22 ms and a body size of 435 B. The response body is a JSON object:

```
1 {
2   "updated": 1
3 }
```

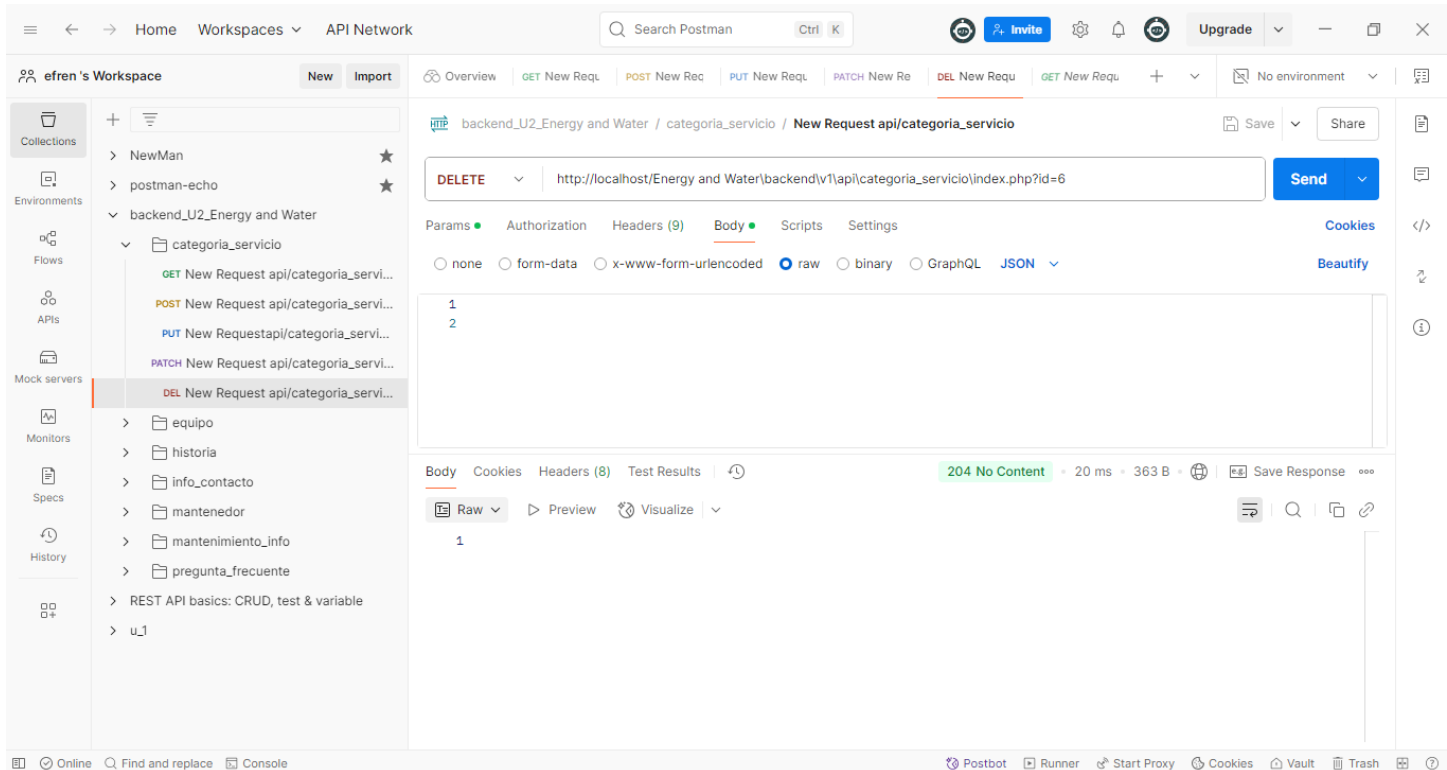


The screenshot shows the Postman interface with a workspace named "efren's Workspace". The left sidebar lists collections and environments. The main panel shows a PATCH request to the URL `http://localhost/Energy%20and%20Water/backend/v1/api/categoria_servicio/index.php?id=5`. The request body is a JSON object:

```
1 {
2   "activo": 0
3 }
```

The response is a 200 OK status with a response time of 20 ms and a body size of 450 B. The response body is a JSON object:

```
1 {
2   "changed": 1,
3   "activo": false
4 }
```



## Conclusiones

En este proyecto hemos construido una API RESTful completa para la gestión de servicios, categorías, contacto, historia, equipo y FAQs. Los desafíos incluyeron validar correctamente los JSON y manejar los códigos HTTP adecuados. La integración con Postman nos permitió automatizar pruebas y asegurar la estabilidad de los endpoints. Para futuras mejoras se podría implementar autenticación y paginación.

Cabe destacar que, debido a la falta de recursos específicos por parte de la institución para orientar sobre cómo estructurar correctamente este informe, se recurrió al uso de herramientas de inteligencia artificial para apoyar su elaboración. La IA fue utilizada como asistente técnico y de redacción, permitiendo organizar el contenido, documentar correctamente la API y garantizar una presentación clara y profesional.