PRACTICA 2

ESTUDIANTE: Alexander Condori Martinez

CÉDULA DE IDENTIDAD: 8343695

CARRERA: Informatica

MATERIA: Programación Web 3 (INF 133)

DOCENTE: Lic. Jhonny Felipez Andrade

FECHA: 12/11/2025

UNIVERSIDAD MAYOR DE SAN ANDRÉS (UMSA)

La Paz - Bolivia

Noviembre, 2025

1. Crea un endpoint POST /categorias que permita registrar una nueva categoría enviando nombre y descripcion en el body de la petición.

```
export const insertaCategoria = async (nombre, descripcion) => {
    const [resultado] = await db.query(
        'INSERT INTO categorias (nombre, descripcion) VALUES (?, ?)',
        [nombre, descripcion]
    );
    return resultado.insertId;
};
```


```
{
    "id": 4,
    "nombre": "Hogar",
    "message": "Categoría creada exitosamente."
}
```


2. Crea un endpoint GET /categorias que devuelva todas las categorías registradas en la base de datos.

```
export const obtenerTodasCategorias = async () => {
    const [filas] = await db.query('SELECT * FROM categorias');
    return filas;
};
```


```
[
    {
        "id": 1,
        "nombre": "Electrónica",
        "descripcion": "Dispositivos electrónicos y gadgets",
        "fecha_alta": "2025-11-11T18:35:58.000Z",
        "fecha_act": "2025-11-11T18:35:58.000Z"
    },
    {
        "id": 2,
        "nombre": "Oficina",
        "descripcion": "Material y accesorios de\r\noficina",
        "fecha_alta": "2025-11-11T18:35:58.000Z",
        "fecha_act": "2025-11-11T18:35:58.000Z"
    },
    {
        "id": 3,
        "nombre": "Accesorios",
        "descripcion": "Todo tipo de accesorios para tecnología",
        "fecha_alta": "2025-11-11T20:18:03.000Z",
        "fecha_act": "2025-11-11T20:18:03.000Z"
    },
```

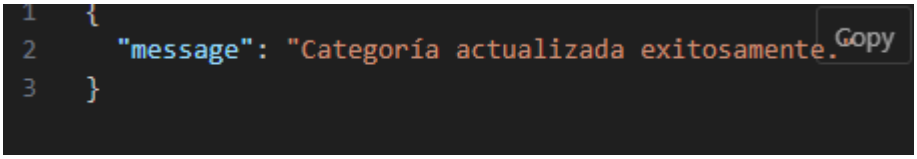

Activar Windows
Ve a Configuración par

3. Crea un endpoint GET /categorias/:id que devuelva la categoría con el ID especificado y además, incluya todos los productos que pertenecen a esa categoría.

```
export const obtenerCategoriaConProductos = async (id) => {

    // a. Obtener la categoría

    const [categoria] = await db.query('SELECT * FROM categorias WHERE id = ?', [id]);



    if (categoria.length === 0) return null;



    // b. Obtener los productos asociados

    const [productos] = await db.query(

        'SELECT * FROM productos WHERE categoria_id = ?',

        [id]

    );



    // c. Combinar y devolver

    return { ...categoria[0], productos };

};
```

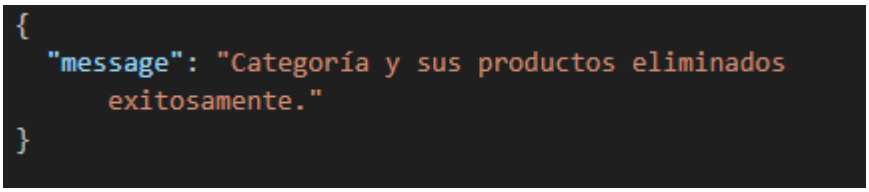
4. Crea un endpoint PUT /categorias/:id que permita actualizar todos los datos de la categoría con el ID especificado.

```
// ejercicio4

export const actualizaCategoria = async (id, nombre, descripcion) => {

    const [resultado] = await db.query(

        'UPDATE categorias SET nombre = ?, descripcion = ?, fecha_act = CURRENT_TIMESTAMP WHERE id = ?',

        [nombre, descripcion, id]

    );

    return resultado.affectedRows;

};
```



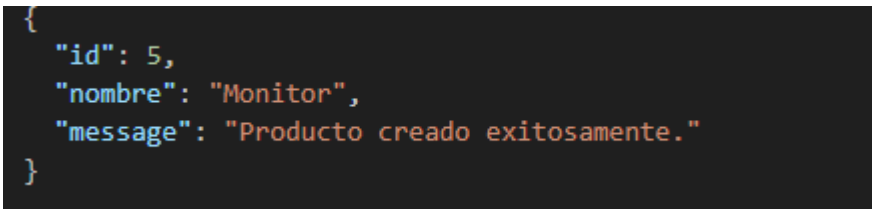
5. Crea un endpoint DELETE /categorias/:id que elimine la categoría indicada y, al mismo tiempo, elimine automáticamente todos los productos que pertenecen a esa categoría

```
// ejercicio5

export const eliminaCategoria = async (id) => {

    // La eliminación de productos es automática (ON DELETE CASCADE)

    const [resultado] = await db.query('DELETE FROM categorias WHERE id = ?', [id]);

    return resultado.affectedRows;

};
```



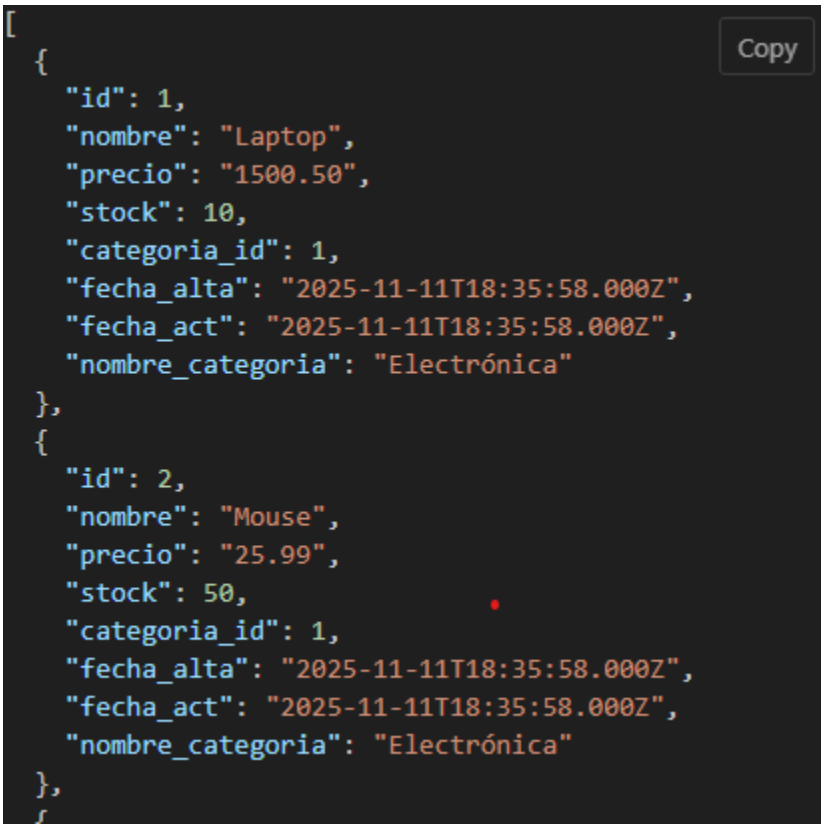
6. Crea un endpoint POST /productos que permita registrar un nuevo producto enviando nombre, precio, stock y categoria_id para asociarlo a una categoría existente.

```
// ejercicio6. POST /productos

export const insertaProducto = async (nombre, precio, stock, categoria_id) => {

   const [resultado] = await db.query(

      'INSERT INTO productos (nombre, precio, stock, categoria_id) VALUES (?, ?, ?, ?)',

      [nombre, precio, stock, categoria_id]

   );

   return resultado.insertId;

};
```



7. Crea un endpoint GET /productos que devuelva todos los productos y muestre el nombre de la categoría a la que pertenece cada uno.

```
// ejercicio7. GET /productos
export const obtenerTodosProductos = async () => {
  const [filas] = await db.query(`
    SELECT
      p.*,
      c.nombre AS nombre_categoria
    FROM
      productos p
    JOIN
      categorias c ON p.categoria_id = c.id
  `);
  return filas;
};
```



8. Crea un endpoint GET /productos/:id que devuelva la información de un producto por su ID, incluyendo el nombre de la categoría asociada.

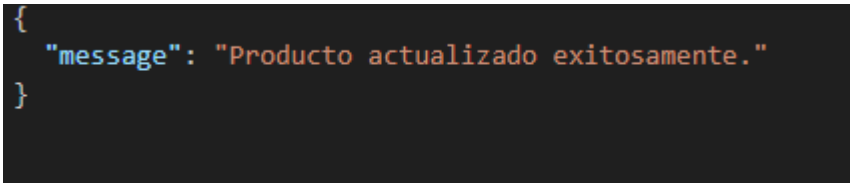
```
// ejercicio8. GET /productos/:id

export const obtenerProductoPorId = async (id) => {
  const [filas] = await db.query(`
    SELECT
      p.*,
      c.nombre AS nombre_categoria
    FROM
      productos p
    JOIN
      categorias c ON p.categoria_id = c.id
    WHERE p.id = ?
  `, [id]);
  return filas[0];
};
```


```
{
  "id": 2,
  "nombre": "Mouse",
  "precio": "25.99",
  "stock": 50,
  "categoria_id": 1,
  "fecha_alta": "2025-11-11T18:35:58.000Z",
  "fecha_act": "2025-11-11T18:35:58.000Z",
  "nombre_categoria": "Electrónica"
}
```


9. Crea un endpoint PUT /productos/:id que permita actualizar todos los datos de un producto, incluyendo la opción de cambiar la categoría a la que pertenece mediante categoria_id.

```
// ejercicio9. PUT /productos/:id

export const actualizaProducto = async (id, nombre, precio, stock, categoria_id) => {
  const [resultado] = await db.query(
    `UPDATE productos SET nombre = ?, precio = ?, stock = ?, categoria_id = ?, fecha_act = CURRENT_TIMESTAMP
WHERE id = ?`,
    [nombre, precio, stock, categoria_id, id]
  );
```

```
    return resultado.affectedRows;
};
```



10. Crea un endpoint PATCH /productos/:id/stock que permita incrementar o decrementar el stock de un producto enviando al body la cantidad que se desea sumar o restar.

```
// ejercicio10. PATCH /productos/:id/stock
export const actualizaStock = async (id, cantidad) => {
    const [resultado] = await db.query(
        `UPDATE productos SET stock = stock + ?, fecha_act = CURRENT_TIMESTAMP WHERE id = ?`,
        [cantidad, id] // 'cantidad' puede ser positiva (incrementar) o negativa (decrementar)
    );
    return resultado.affectedRows;
};
```