UNIVERSIDAD MAYOR DE SAN ANDRES

FACULTAD DE CIENCIAS PURAS Y NATURALES

INFORMÁTICA

PRACTICA #2

PROGRAMACION WEB 3

Nombre Completo : ADUVIRI ZAPANA ADRIAN JORGE

C.I. : 14878180

Carrera : INFORMATICA

Materia : PROGRAMACION WEB 3

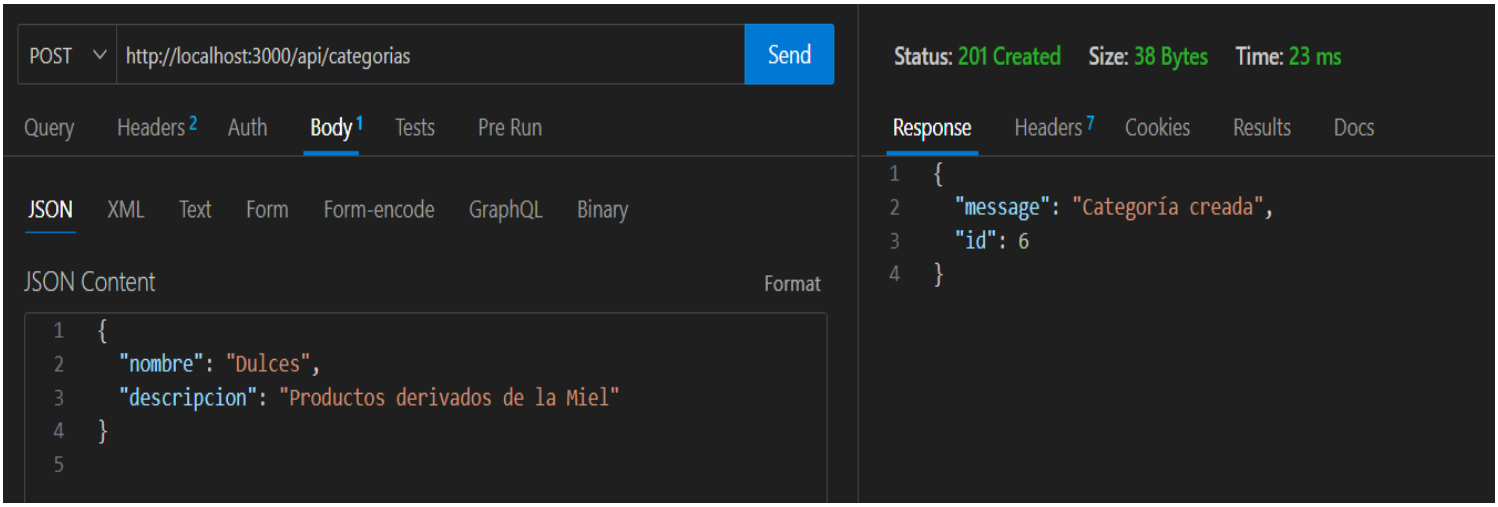
Fecha : 10/11/2025

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

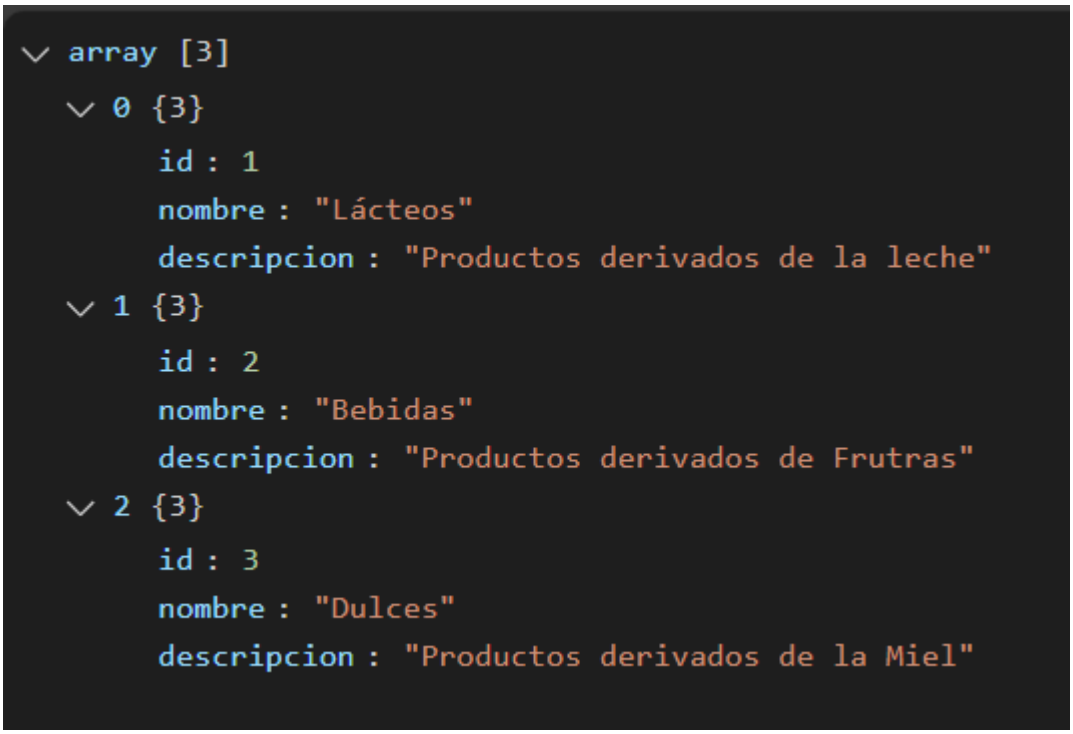
CODIGO

```
// 1. Crear categoría
router.post('/', (req, res) => {
  const { nombre, descripcion } = req.body;
  if (!nombre) return res.status(400).json({ error: 'nombre es requerido' });

  db.query(
    'INSERT INTO categorias (nombre, descripcion) VALUES (?, ?)',
    [nombre, descripcion || null],
    (err, result) => {
      if (err) return res.status(500).json({ error: err.message });
      res.status(201).json({ message: 'Categoría creada', id: result.insertId });
    }
  );
});

```

PROCESO



RESPUESTA ESPERADA

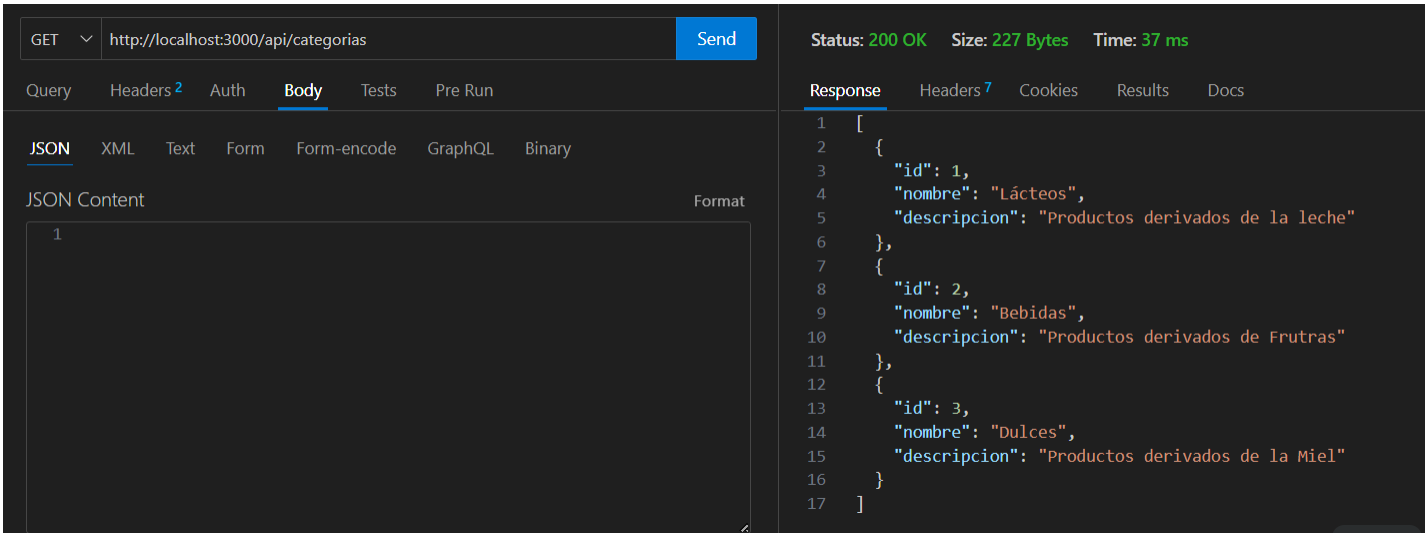


2. Crea un endpoint GET /categorias que devuelva todas las categorías registradas en la base de datos.

CODIGO

```
router.get('/', (req, res) => {
  db.query('SELECT * FROM categorias', (err, results) => {
    if (err) return res.status(500).json({ error: err });
    res.json(results);
  });
});
```

PROCESO



RESPUESTA ESPERADA

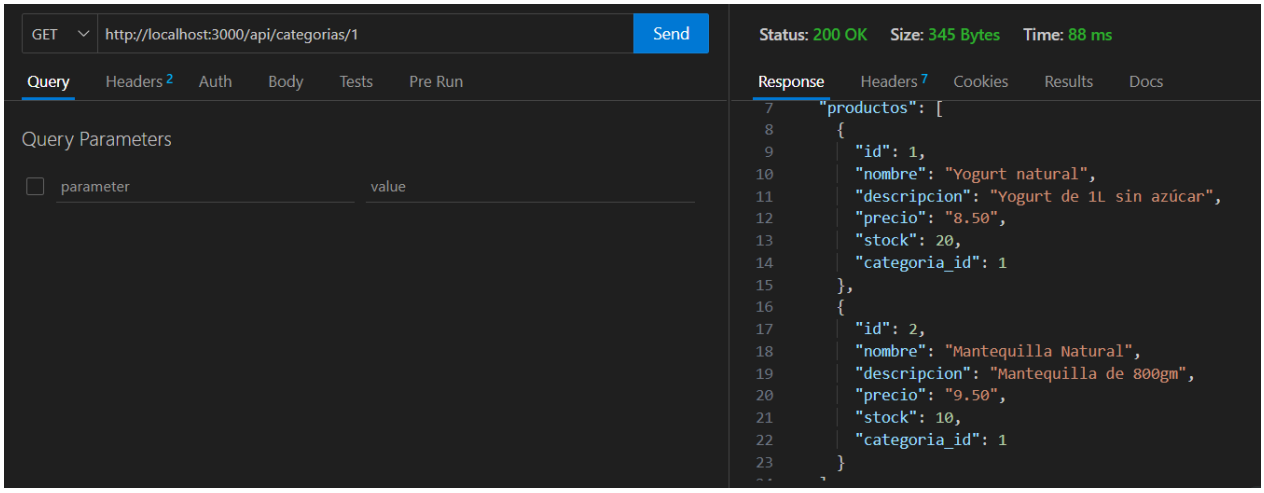
```
array [3]
  0 {3}
      id : 1
      nombre : "Lácteos"
      descripcion : "Productos derivados de la leche"
  1 {3}
      id : 2
      nombre : "Bebidas"
      descripcion : "Productos derivados de Frutras"
  2 {3}
      id : 3
      nombre : "Dulces"
      descripcion : "Productos derivados de la Miel"
```

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.

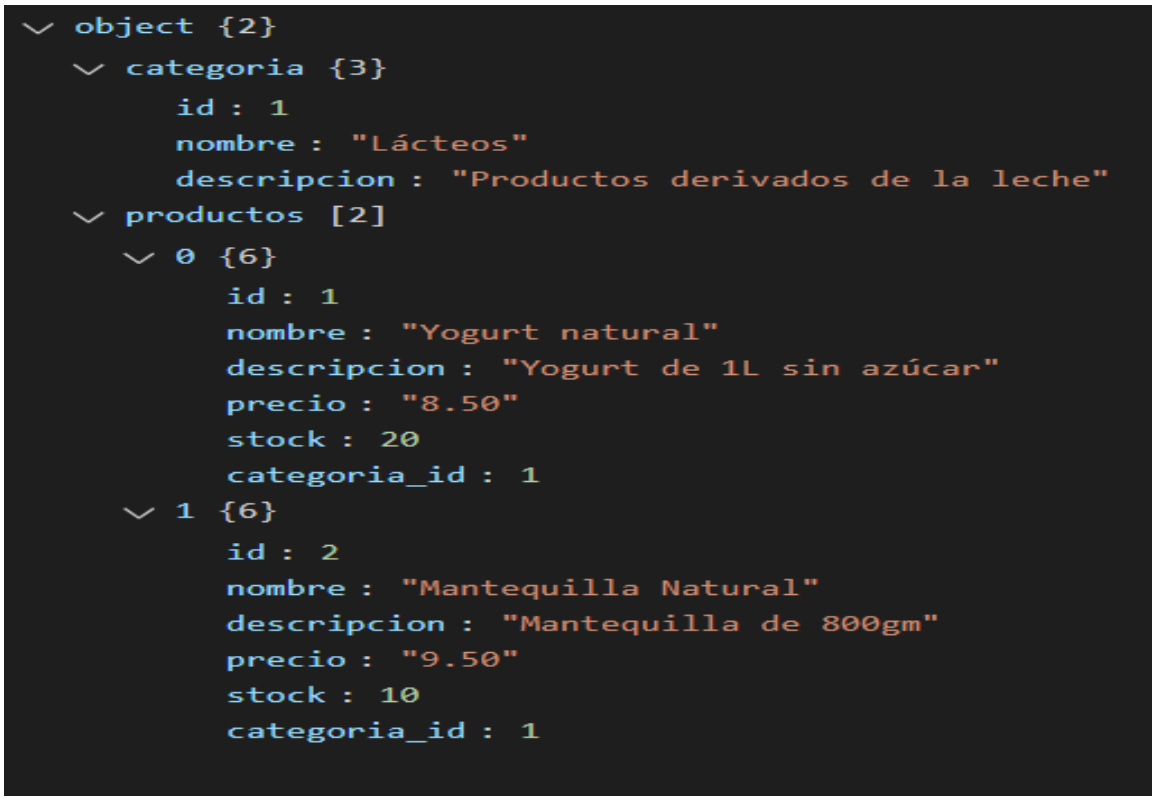
CODIGO

```
// 3. Obtener una categoría por ID (con sus productos)
router.get('/:id', (req, res) => {
  const categoriaId = req.params.id;

  db.query('SELECT * FROM categorias WHERE id = ?', [categoriaId], (err, catRows) => {
    if (err) return res.status(500).json({ error: err.message });
    if (catRows.length === 0) return res.status(404).json({ error: 'Categoría no encontrada' });

    const categoria = catRows[0];

    db.query('SELECT * FROM productos WHERE categoria_id = ?', [categoriaId], (err, prodRows) => {
      if (err) return res.status(500).json({ error: err.message });
      res.json({ categoria, productos: prodRows });
    });
  });
});
```

PROCESO



RESPUESTA ESPERADA

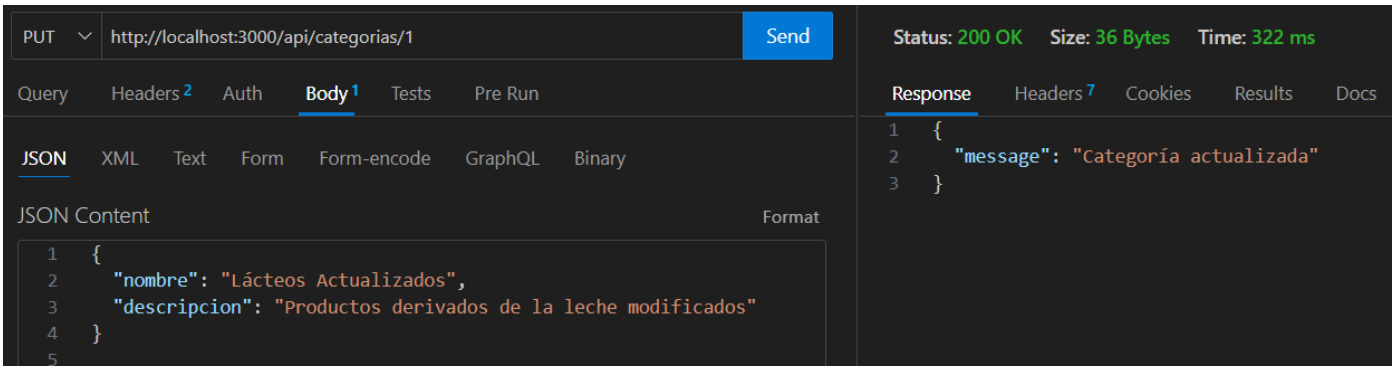


4. Crea un endpoint PUT /categorias/:id que permita actualizar todos los datos de la categoría con el ID especificado.

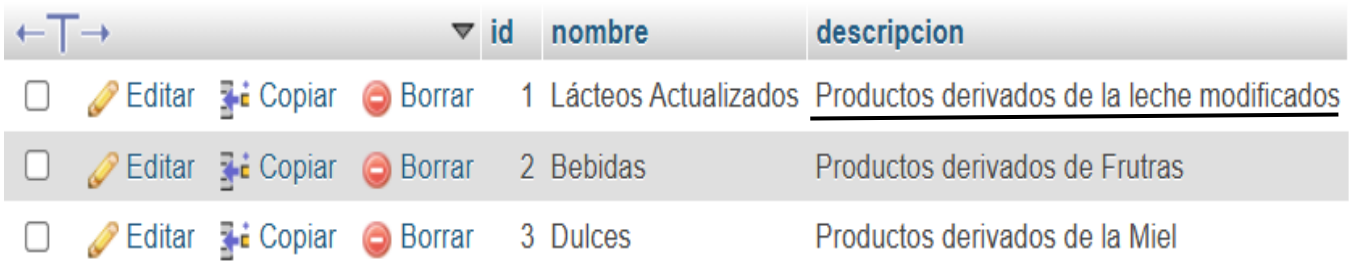
CODIGO

```
// 4. Actualizar categoría
router.put('/:id', (req, res) => {
  const id = req.params.id;
  const { nombre, descripcion } = req.body;
  if (!nombre) return res.status(400).json({ error: 'nombre es requerido' });

  db.query(
    'UPDATE categorias SET nombre = ?, descripcion = ? WHERE id = ?',
    [nombre, descripcion || null, id],
    (err, result) => {
      if (err) return res.status(500).json({ error: err.message });
      if (result.affectedRows === 0) return res.status(404).json({ error: 'Categoría no encontrada' });
      res.json({ message: 'Categoría actualizada' });
    }
  );
});
```

PROCESO



RESPUESTA ESPERADA

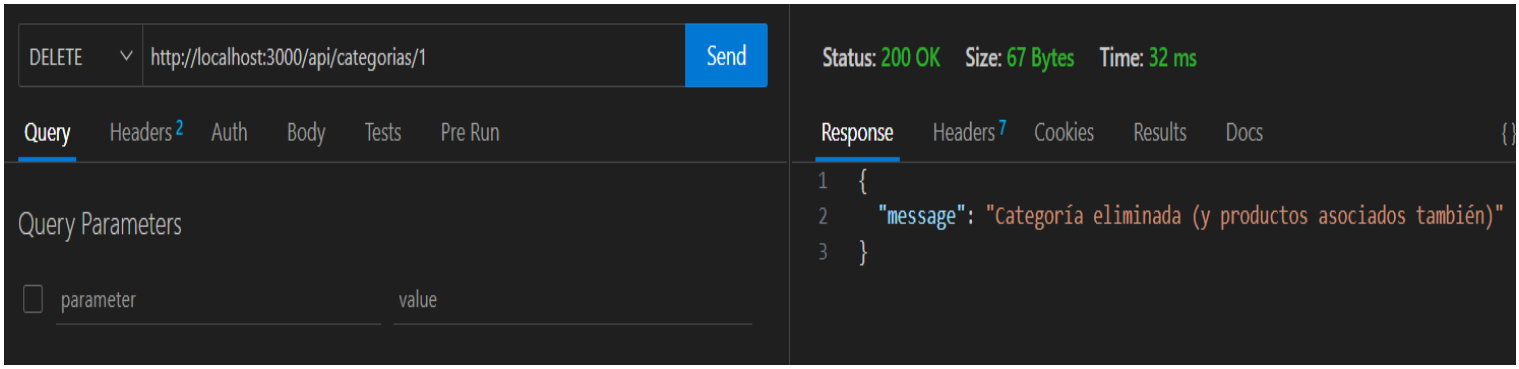


5. Crea un endpoint DELETE /categorias/:id que elimine la categoría indicada y, al mismo tiempo, elimine automáticamente todos los productos que pertenecen a esa categoría.

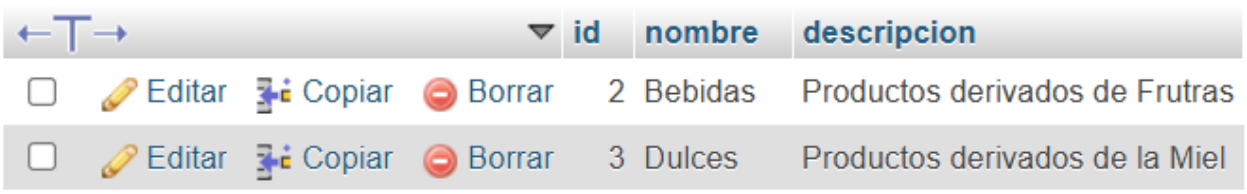
CODIGO

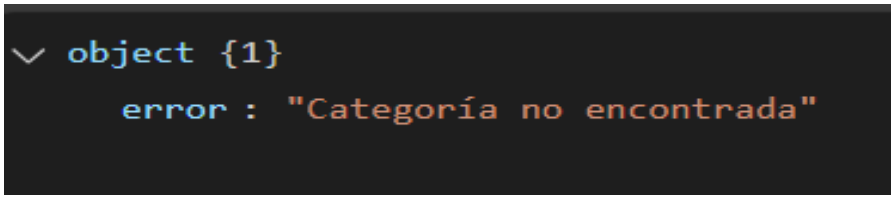
```
// 5. Eliminar categoría
router.delete('/:id', (req, res) => {
  const id = req.params.id;
  db.query('DELETE FROM categorias WHERE id = ?', [id], (err, result) => {
    if (err) return res.status(500).json({ error: err.message });
    if (result.affectedRows === 0) return res.status(404).json({ error: 'Categoría no encontrada' });
    res.json({ message: 'Categoría eliminada (y productos asociados también)' });
  });
});

module.exports = router;
```

PROCESO



RESPUESTA ESPERADA





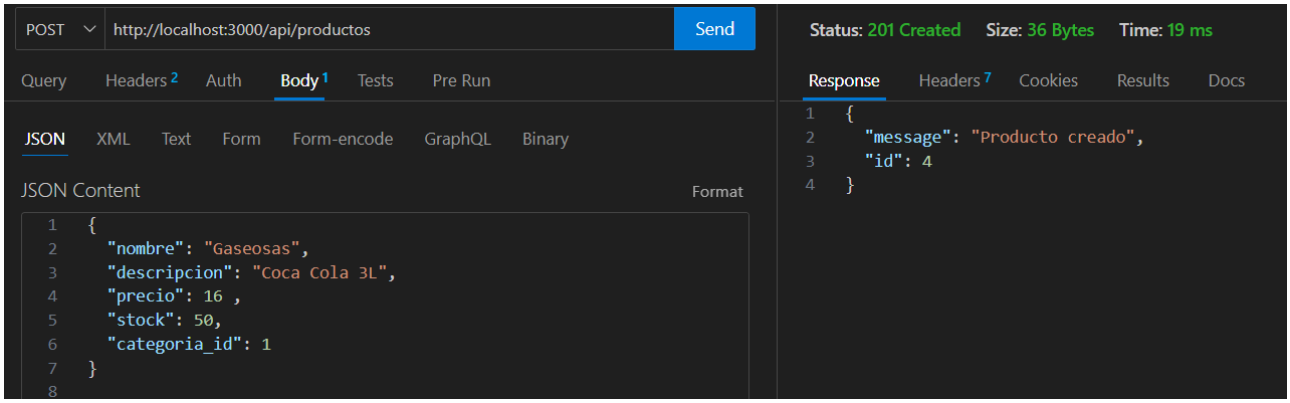
(Categoria con id 1 ELIMINADA)

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.

CODIGO

```js
// 6. Crear producto
router.post('/', (req, res) => {
  const { nombre, descripcion, precio, stock, categoria_id } = req.body;

  if (!nombre || !precio || !categoria_id) {
    return res.status(400).json({ error: 'nombre, precio y categoria_id son requeridos' });
  }

  const sql = 'INSERT INTO productos (nombre, descripcion, precio, stock, categoria_id) VALUES (?, ?, ?, ?, ?)';
  db.query(sql, [nombre, descripcion || null, precio, stock || 0, categoria_id], (err, result) => {
    if (err) return res.status(500).json({ error: err.message });
    res.status(201).json({ message: 'Producto creado', id: result.insertId });
  });
});

module.exports = router;
```

PROCESO



RESPUESTA ESPERADA

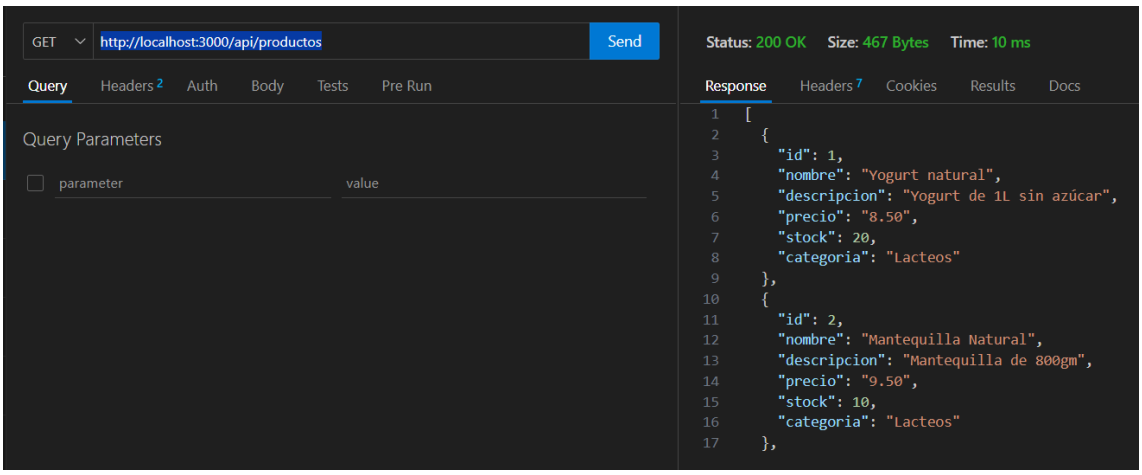
	id	nombre	descripcion	precio	stock	categoria_id
Editar Copiar Borrar	1	Yogurt natural	Yogurt de 1L sin azúcar	8.50	20	1
Editar Copiar Borrar	2	Mantequilla Natural	Mantequilla de 800gm	9.50	10	1
Editar Copiar Borrar	4	Gaseosas	Coca Cola 3L	16.00	50	1

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

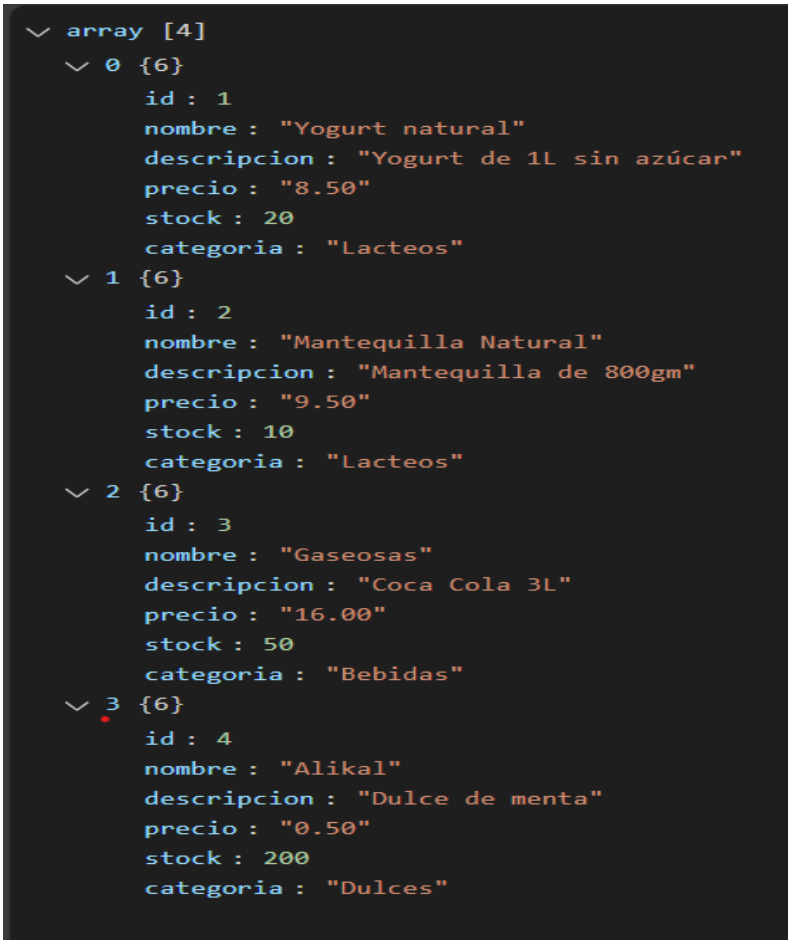
CODIGO

```js
// 7. Obtener todos los productos
router.get('/', (req, res) => {
  const sql = `
    SELECT p.id, p.nombre, p.descripcion, p.precio, p.stock, c.nombre AS categoria
    FROM productos p
    JOIN categorias c ON p.categoria_id = c.id
  `;

  db.query(sql, (err, results) => {
    if (err) return res.status(500).json({ error: err.message });
    res.json(results);
  });
});


module.exports = router;
```

PROCESO



RESPUESTA ESPERADA

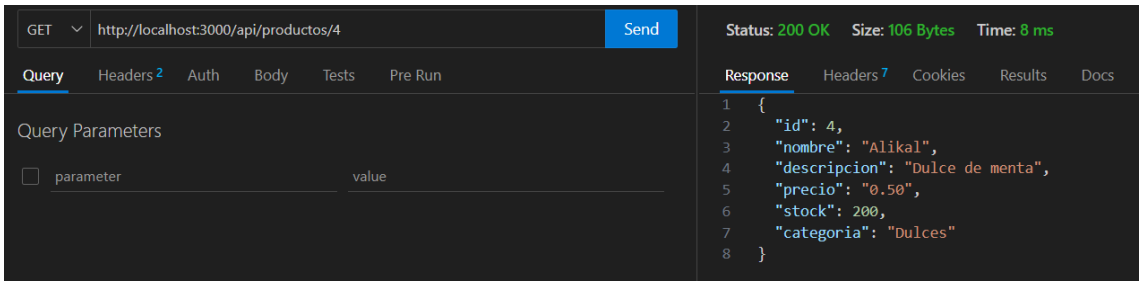


8. Crea un endpoint GET /productos/:id que devuelva la información de un producto por su ID, incluyendo el nombre de la categoría asociada.

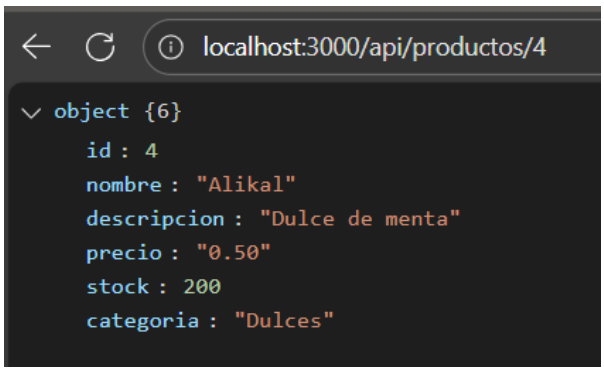
CODIGO

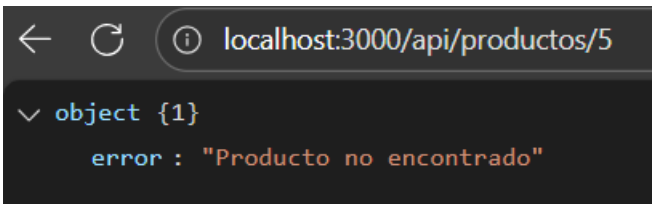
```
// 8. Obtener un producto por su ID
router.get('/:id', (req, res) => {
  const id = req.params.id;

  const sql = `
    SELECT p.id, p.nombre, p.descripcion, p.precio, p.stock, c.nombre AS categoria
    FROM productos p
    LEFT JOIN categorias c ON p.categoria_id = c.id
    WHERE p.id = ?
  `;

  db.query(sql, [id], (err, results) => {
    if (err) return res.status(500).json({ error: err.message });
    if (results.length === 0) return res.status(404).json({ error: 'Producto no encontrado' });
    res.json(results[0]);
  });
});
```

PROCESO



RESPUESTA ESPERADA



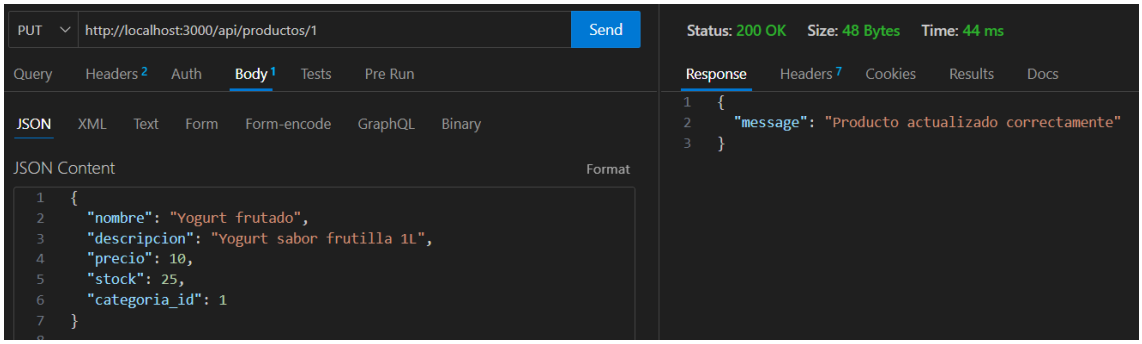


9. Crea un endpoint PUT /productos/:id que permita actualizar todos los datos de un producto, incluyendo la opción de cambiar la categoría a la que pertenece mediante categoria_id.

CODIGO

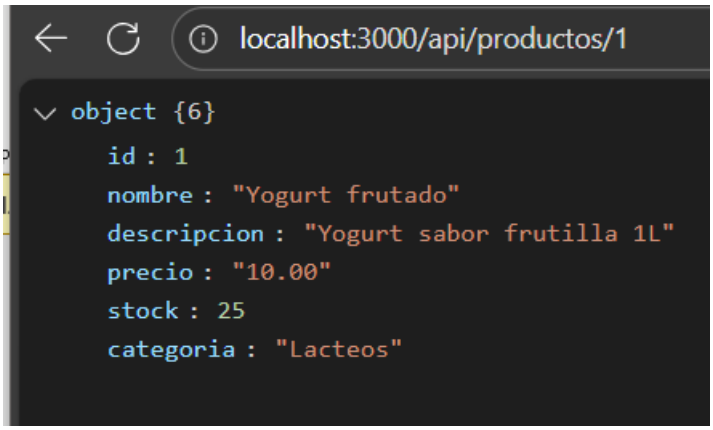
```
// 9. Actualizar un producto
router.put('/:id', (req, res) => {
  const id = req.params.id;
  const { nombre, descripcion, precio, stock, categoria_id } = req.body;

  if (!nombre || !precio || !categoria_id) {
    return res.status(400).json({ error: 'nombre, precio y categoria_id son requeridos' });
  }

  const sql = `
    UPDATE productos
    SET nombre = ?, descripcion = ?, precio = ?, stock = ?, categoria_id = ?
    WHERE id = ?
  `;

  db.query(sql, [nombre, descripcion || null, precio, stock || 0, categoria_id, id], (err, result) => {
    if (err) return res.status(500).json({ error: err.message });
    if (result.affectedRows === 0) return res.status(404).json({ error: 'Producto no encontrado' });
    res.json({ message: 'Producto actualizado correctamente' });
  });
});

```

PROCESO



RESPUESTA ESPERADA

	id	nombre	descripcion	precio	stock	categoria_id
Editar Copiar Borrar	1	Yogurt frutado	Yogurt sabor frutilla 1L	10.00	25	1
Editar Copiar Borrar	2	Mantequilla Natural	Mantequilla de 800gm	9.50	10	1
Editar Copiar Borrar	3	Gaseosas	Coca Cola 3L	16.00	50	2
Editar Copiar Borrar	4	Alikal	Dulce de menta	0.50	200	3

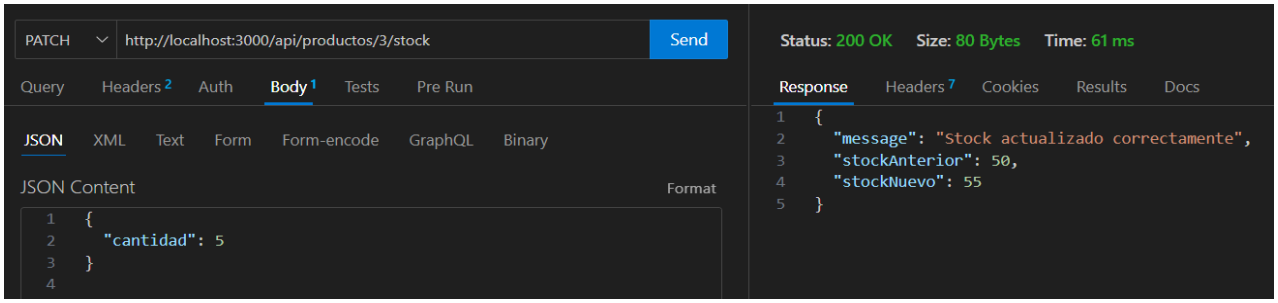


10. Crea un endpoint PATCH /productos/:id/stock que permita incrementar o decrementar el stock de un producto enviando al body la cantidad que se desea sumar o restar.

CODIGO

```js
// 10. Actualizar el stock de un producto
router.patch('/:id/stock', (req, res) => {
  const id = req.params.id;
  const { cantidad } = req.body;

  if (typeof cantidad !== 'number') {
    return res.status(400).json({ error: 'Debes enviar un número en "cantidad"' });
  }

  // Primero obtenemos el stock actual
  db.query('SELECT stock FROM productos WHERE id = ?', [id], (err, results) => {
    if (err) return res.status(500).json({ error: err.message });
    if (results.length === 0) return res.status(404).json({ error: 'Producto no encontrado' });

    const stockActual = results[0].stock;
    const nuevoStock = stockActual + cantidad;

    if (nuevoStock < 0) {
      return res.status(400).json({ error: 'El stock no puede ser negativo' });
    }

    // Actualizamos el stock
    db.query('UPDATE productos SET stock = ? WHERE id = ?', [nuevoStock, id], (err2) => {
      if (err2) return res.status(500).json({ error: err2.message });
      res.json({ message: 'Stock actualizado correctamente', stockAnterior: stockActual, stockNuevo: nuevoStock });
    });
  });
});
```

PROCESO



RESPUESTA ESPERADA

	id	nombre	descripcion	precio	stock	categoria_id
Editar Copiar Borrar	1	Yogurt frutado	Yogurt sabor frutilla 1L	10.00	25	1
Editar Copiar Borrar	2	Mantequilla Natural	Mantequilla de 800gm	9.50	10	1
Editar Copiar Borrar	3	Gaseosas	Coca Cola 3L	16.00	55	2
Editar Copiar Borrar	4	Alikal	Dulce de menta	0.50	200	3

```json
{
  "message": "Stock actualizado correctamente",
  "stockAnterior": 50,
  "stockNuevo": 55
}
```