

Ivan de Jesus ramirez lopez

```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| almacen  |
| almacen2 |
| information_schema |
| integrador |
| integradormercadolibre |
| mercadolibre |
| mysql |
| performance_schema |
| phpmyadmin |
| proyecto1 |
| systemtec |
| tecbd |
| test |
+-----+
13 rows in set (0.005 sec)

MariaDB [(none)]> use almacen2;
Database changed
MariaDB [almacen2]> show tables;
+-----+
| Tables_in_almacen2 |
+-----+
| entrada |
| inventario |
| productos |
| salida |
+-----+
4 rows in set (0.005 sec)

MariaDB [almacen2]> |
```

Se crea la base de datos de almacen y se muestra algunas tablas creadas

```
MariaDB [almacen2]> show tables;
+-----+
| Tables_in_almacen2 |
+-----+
| entrada              |
| inventario           |
| productos            |
| salida               |
+-----+
4 rows in set (0.005 sec)
```



```
MariaDB [almacen2]> select * from entrada;
+-----+-----+-----+-----+
| id | producto_id | cantidad | proveedor |
+-----+-----+-----+-----+
| 1  | 1           | 50       | Proveedor 1 |
| 2  | 2           | 10       | Proveedor 2 |
| 3  | 3           | 50       | Proveedor 3 |
| 4  | 4           | 60       | Proveedor 4 |
| 5  | 5           | 20       | Proveedor 5 |
| 6  | 1           | 20       | NULL        |
| 7  | 1           | 20       | NULL        |
+-----+-----+-----+-----+
7 rows in set (0.012 sec)
```

Se muestra en pantalla las tablas creadas y se ejecuta para ver que tiene cada tabla

```
MariaDB [almacen2]> select * from inventario;
```

id	producto	cantidad
1	Camisetas	240
2	Zapatos deportivos	90
3	Pantalones vaqueros	170
4	Gorras	260
5	Relojes	70
6	Nuevo Producto	50

```
6 rows in set (0.001 sec)
```



```
MariaDB [almacen2]> select * from salida;
```

id	producto_id	cantidad
1	1	20
2	2	15
3	3	10
4	4	30
5	5	5

```
5 rows in set (0.000 sec)
```

```
MariaDB [almacen2]> select * from productos;
```

producto_id	nombre_producto	cantidad_en_inventario	cantidad_entrada	proveedor
1	Camisetas	240	50	Proveedor 1
1	Camisetas	240	20	NULL
1	Camisetas	240	20	NULL
2	Zapatos deportivos	90	10	Proveedor 2
3	Pantalones vaqueros	170	50	Proveedor 3
4	Gorras	260	60	Proveedor 4
5	Relojes	70	20	Proveedor 5
6	Nuevo Producto	50	NULL	NULL

```
8 rows in set (0.001 sec)
```

Aquí se muestra la creación de la vista para inventario y entrada que se llama productos

```
MariaDB [almacen2]> SHOW TRIGGERS;
```

Trigger	Timing	Created	Event	Table	Statement	Definer	character_set_client	collation_connection	Database Collation
tr_incrementar_inventario	AFTER	2023-10-16 13:34:44.23	INSERT	entrada	BEGIN UPDATE inventario SET cantidad = cantidad + NEW.cantidad WHERE id = NEW.producto_id; END	root@localhost	cp850	cp850_general_ci	utf8mb4_general_ci

```
1 row in set (0.009 sec)
```

```
MariaDB [almacen2]>
```

Se ve la creación de los triggers

Ivan de jesus ramirez lopez

```
bdjs > ...
1 // db.js
2 const mysql = require('mysql2');
3
4 const connection = mysql.createConnection({
5   host: 'localhost',
6   user: 'root',
7   password: '',
8   database: 'almacen2',
9 });
10
11 connection.connect((err) => {
12   if (err) {
13     console.error('Error de conexión a la base de datos:', err);
14   } else {
15     console.log('Conexión a la base de datos establecida');
16   }
17 });
18
19 module.exports = connection;
20
```

Se muestra en pantalla la conexión de la base de datos

```
app.js > app.delete('/entrada/id') callback
1 // app.js
2 const express = require('express');
3 const bodyParser = require('body-parser');
4 const db = require('./bd');
5
6 const app = express();
7 app.use(bodyParser.json());
8
9 // rutas para obtener los valores
10
11 app.get('/entrada', (req, res) => {
12   // obtener todos los registros de entrada
13   db.query('SELECT * FROM entrada', (err, results) => {
14     if (err) {
15       console.error('Error al obtener las entradas:', err);
16       res.status(500).json({ error: 'Error al obtener las entradas' });
17     } else {
18       res.status(200).json(results);
19     }
20   });
21 });
22
23 app.post('/entrada', (req, res) => {
24   // crear un registro de entrada
25   const { producto_id, cantidad, proveedor } = req.body;
26   const query = 'INSERT INTO entrada (producto_id, cantidad, proveedor) VALUES (?, ?, ?)';
27   const values = [producto_id, cantidad, proveedor];
28
29   db.query(query, values, (err, result) => {
30     if (err) {
31       console.error('Error al crear una entrada:', err);
32       res.status(500).json({ error: 'Error al crear una entrada' });
33     }
34   });
35 });
```

Se inicia con la creación de la API para consultar con el CRUD

```
app.put('/entrada/:id', (req, res) => {
  // actualizar un registro de entrada
  const entradaId = req.params.id;
  const { producto_id, cantidad, proveedor } = req.body;

  const query = 'UPDATE entrada SET producto_id = ?, cantidad = ?, proveedor = ? WHERE id = ?';
  const values = [producto_id, cantidad, proveedor, entradaId];

  db.query(query, values, (err) => {
    if (err) {
      console.error('Error al actualizar la entrada:', err);
      res.status(500).json({ error: 'Error al actualizar la entrada' });
    } else {
      res.status(200).json({ message: 'Entrada actualizada con éxito' });
    }
  });
});

app.delete('/entrada/:id', (req, res) => {
  // eliminar un registro de entrada
  const entradaId = req.params.id;

  db.query('DELETE FROM entrada WHERE id = ?', [entradaId], (err) => {
    if (err) {
      console.error('Error al eliminar la entrada:', err);
      res.status(500).json({ error: 'Error al eliminar la entrada' });
    } else {
      res.status(200).json({ message: 'Entrada eliminada con éxito' });
    }
  });
});
```

Lector de pantalla optimizado

Lín. 68, col. 6

Espacios: 2

UTF-8

CRLF

JavaScript

Go Live

```
// "salida"
app.get('/salida', (req, res) => {
  // obtener todos los registros de salida
  db.query('SELECT * FROM salida', (err, results) => {
    if (err) {
      console.error('Error al obtener las salidas:', err);
      res.status(500).json({ error: 'Error al obtener las salidas' });
    } else {
      res.status(200).json(results);
    }
  });
});

app.post('/salida', (req, res) => {
  // crear un registro de salida
  const { producto_id, cantidad } = req.body;
  const query = 'INSERT INTO salida (producto_id, cantidad) VALUES (?, ?)';
  const values = [producto_id, cantidad];

  db.query(query, values, (err, result) => {
    if (err) {
      console.error('Error al crear una salida:', err);
      res.status(500).json({ error: 'Error al crear una salida' });
    } else {
      res.status(201).json({ message: 'Salida creada con éxito', id: result.insertId });
    }
  });
});
```

```
bdjs  appjs  X
appjs > app.delete('/entrada/id') callback
100 app.put('/salida/:id', (req, res) => {
101   // actualizar un registro de salida
102   const salidaId = req.params.id;
103   const { producto_id, cantidad } = req.body;
104
105   const query = 'UPDATE salida SET producto_id = ?, cantidad = ? WHERE id = ?';
106   const values = [producto_id, cantidad, salidaId];
107
108   db.query(query, values, (err) => {
109     if (err) {
110       console.error('Error al actualizar la salida:', err);
111       res.status(500).json({ error: 'Error al actualizar la salida' });
112     } else {
113       res.status(200).json({ message: 'Salida actualizada con éxito' });
114     }
115   });
116 });
117
118 app.delete('/salida/:id', (req, res) => {
119   // eliminar un registro de salida
120   const salidaId = req.params.id;
121
122   db.query('DELETE FROM salida WHERE id = ?', [salidaId], (err) => {
123     if (err) {
124       console.error('Error al eliminar la salida:', err);
125       res.status(500).json({ error: 'Error al eliminar la salida' });
126     } else {
127       res.status(200).json({ message: 'Salida eliminada con éxito' });
128     }
129   });
130 });
---
```

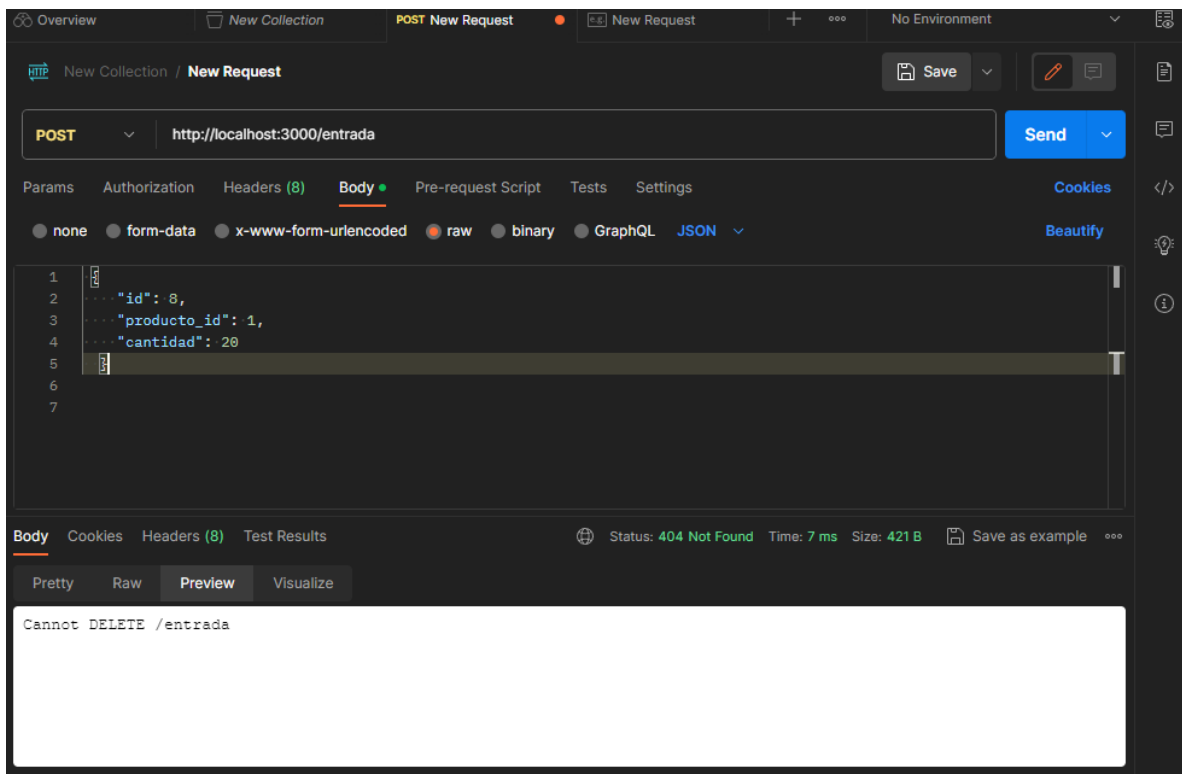
```
// Iniciar el servidor Express
const port = 3000;
app.listen(port, () => {
  console.log(`Servidor API CRUD escuchando en el puerto ${port}`);
});
```

Impresión con formato estilístico ☒

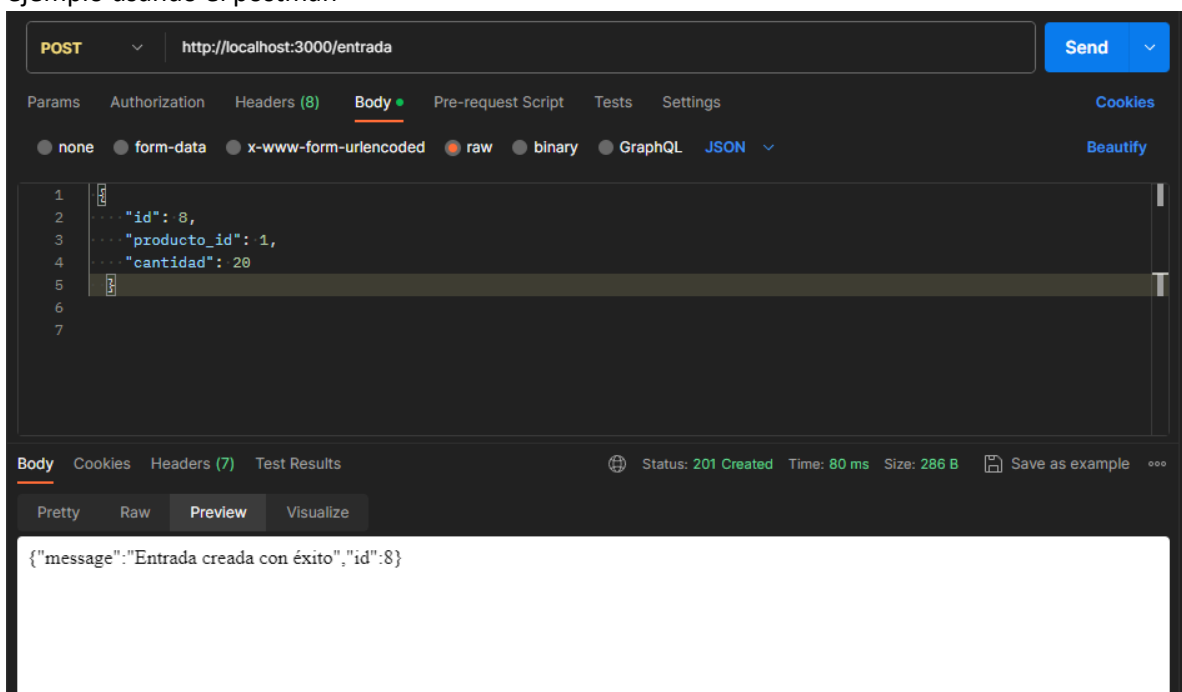
```
[
  {
    "id": 1,
    "producto_id": 1,
    "cantidad": 50,
    "proveedor": "Proveedor 1"
  },
  {
    "id": 2,
    "producto_id": 2,
    "cantidad": 10,
    "proveedor": "Proveedor 2"
  },
  {
    "id": 3,
    "producto_id": 3,
    "cantidad": 50,
    "proveedor": "Proveedor 3"
  },
  {
    "id": 4,
    "producto_id": 4,
    "cantidad": 60,
    "proveedor": "Proveedor 4"
  },
  {
    "id": 5,
    "producto_id": 5,
    "cantidad": 20,
    "proveedor": "Proveedor 5"
  },
  {
    "id": 6,
    "producto_id": 1,
    "cantidad": 20,
    "proveedor": null
  },
  {
    "id": 7,
    "producto_id": 1,
    "cantidad": 20,
    "proveedor": null
  }
]
```

Ejemplo de que si se conectaron

Ivan de jesus ramirez lopez



ejemplo usando el postman



se dio con \u00e9xito

```
{  "id": 8,  "producto_id": 1,  "cantidad": 20,  "proveedor": null}
```