

ACTIVIDAD CRUD - PLANNING MARIADB - NODEJS

SEMANA 18

Pedro Serna

Tutor/a: Natalia Sotelo
Referente: Kenny Paz



DOCUMENTACIÓN

Conforme a las instrucciones de esta actividad, se realiza una documentación detallada del script para la creación de la Base de datos, también se adjuntan capturas del proceso CRUD.

LINK DE GITHUB :

https://github.com/pedroserna22/CRUD_MariaDB_NodeJs.git

CÓDIGO DE BASE DE DATOS

```
1 CREATE TABLE `planning` (  
2   `id` INT(11) NOT NULL AUTO_INCREMENT,  
3   `name` VARCHAR(50) NOT NULL,  
4   `description` VARCHAR(50) NOT NULL,  
5   `created_at` DATE NOT NULL,  
6   `updated_at` DATE NOT NULL,  
7   `status` VARCHAR(50) NOT NULL,  
8   PRIMARY KEY (`id`) USING BTREE  
9 )  
10 COLLATE='latin1_swedish_ci'  
11 ENGINE=InnoDB  
12 AUTO_INCREMENT=6  
13 ;  
14
```

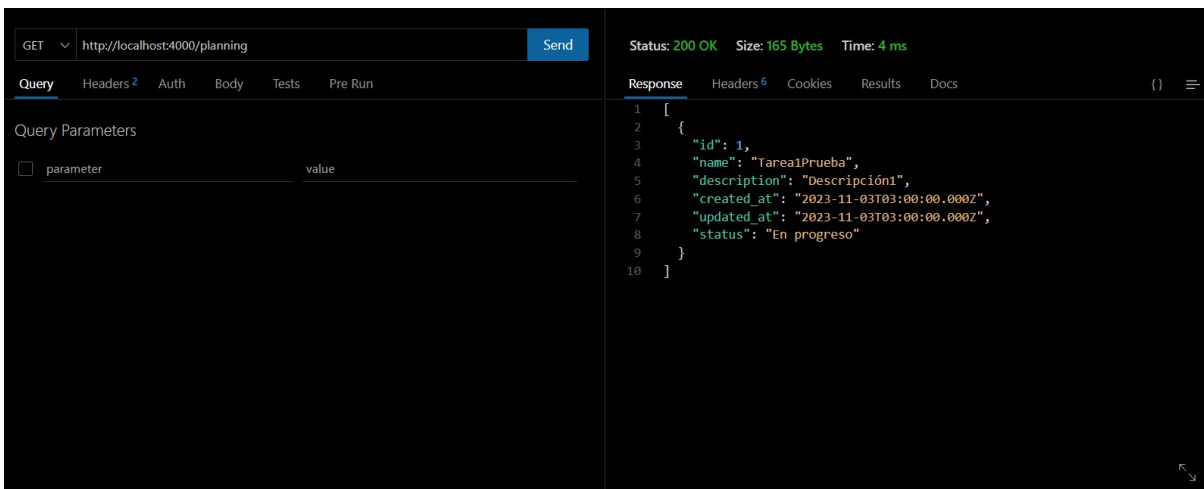
```
CREATE TABLE planning (  
  id INT(11) NOT NULL AUTO_INCREMENT,  
  name VARCHAR(50) NOT NULL,  
  description VARCHAR(50) NOT NULL,  
  created_at DATE NOT NULL,  
  updated_at DATE NOT NULL,  
  status VARCHAR(50) NOT NULL,  
  PRIMARY KEY (id) USING BTREE  
)  
COLLATE='latin1_swedish_ci'  
ENGINE=InnoDB  
AUTO_INCREMENT=6  
;
```

PASO 1 - GET

Se realiza una petición **GET** para obtener los registros creados en la base de datos.

planning.planning: 1 filas en total (aprox) >> Sigüientes <> Mostrar todo | ▼ Ordenación ▼ Columnas (6/6) ▼ Filtro

id	name	description	created_at	updated_at	status
1	Tarea1Prueba	Descripción1	2023-11-03	2023-11-03	En progreso



GET http://localhost:4000/planning Send

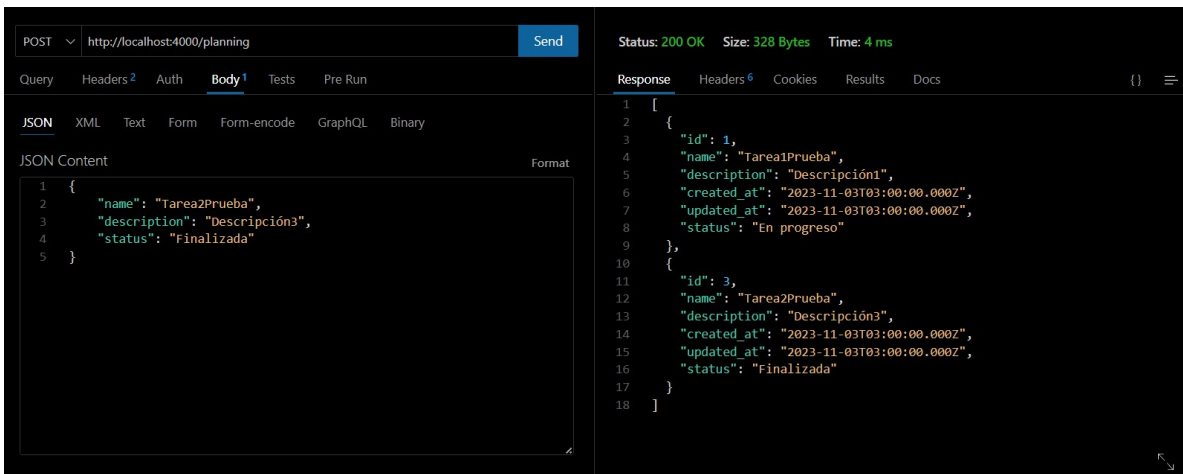
Status: 200 OK Size: 165 Bytes Time: 4 ms

Response

```
[
  {
    "id": 1,
    "name": "Tarea1Prueba",
    "description": "Descripción1",
    "created_at": "2023-11-03T03:00:00.000Z",
    "updated_at": "2023-11-03T03:00:00.000Z",
    "status": "En progreso"
  }
]
```

PASO 2 - POST

Se utiliza **POST** para crear un nuevo registro.



POST http://localhost:4000/planning Send

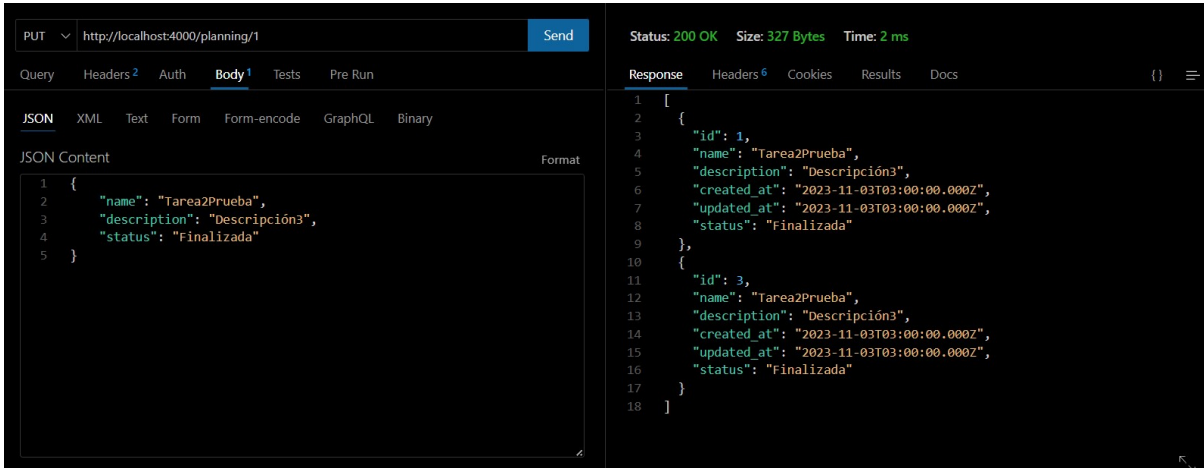
Status: 200 OK Size: 328 Bytes Time: 4 ms

Response

```
[
  {
    "id": 1,
    "name": "Tarea1Prueba",
    "description": "Descripción1",
    "created_at": "2023-11-03T03:00:00.000Z",
    "updated_at": "2023-11-03T03:00:00.000Z",
    "status": "En progreso"
  },
  {
    "id": 3,
    "name": "Tarea2Prueba",
    "description": "Descripción3",
    "created_at": "2023-11-03T03:00:00.000Z",
    "updated_at": "2023-11-03T03:00:00.000Z",
    "status": "Finalizada"
  }
]
```

PASO 3 - PUT

Se utiliza **PUT** para modificar el registro con el ID 1.



The screenshot shows a REST client interface with a PUT request to `http://localhost:4000/planning/1`. The request body is a JSON object:

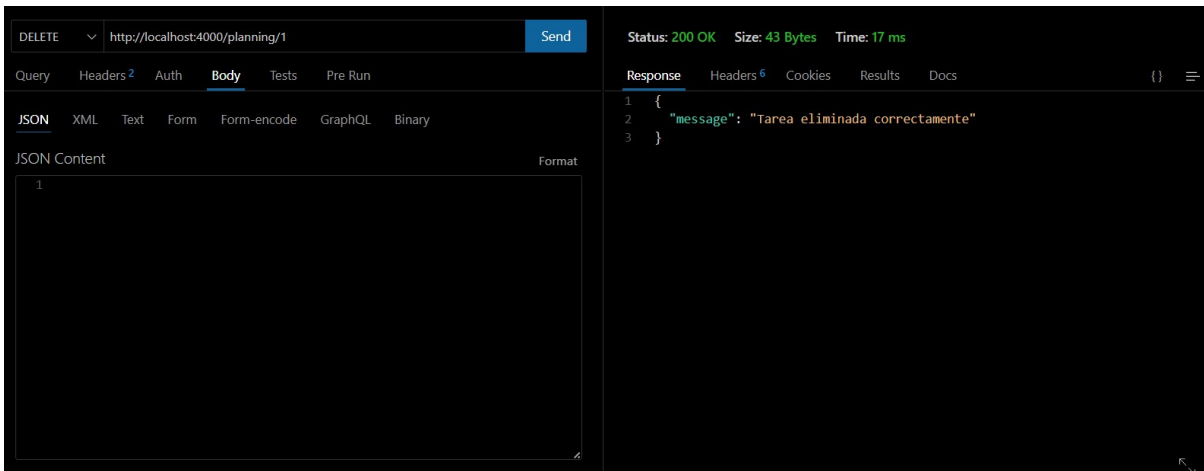
```
1 {
2   "name": "Tarea2Prueba",
3   "description": "Descripción3",
4   "status": "Finalizada"
5 }
```

The response is a 200 OK status with a size of 327 Bytes and a time of 2 ms. The response body is a JSON array:

```
1 [
2   {
3     "id": 1,
4     "name": "Tarea2Prueba",
5     "description": "Descripción3",
6     "created_at": "2023-11-03T03:00:00.000Z",
7     "updated_at": "2023-11-03T03:00:00.000Z",
8     "status": "Finalizada"
9   },
10  {
11    "id": 3,
12    "name": "Tarea2Prueba",
13    "description": "Descripción3",
14    "created_at": "2023-11-03T03:00:00.000Z",
15    "updated_at": "2023-11-03T03:00:00.000Z",
16    "status": "Finalizada"
17  }
18 ]
```

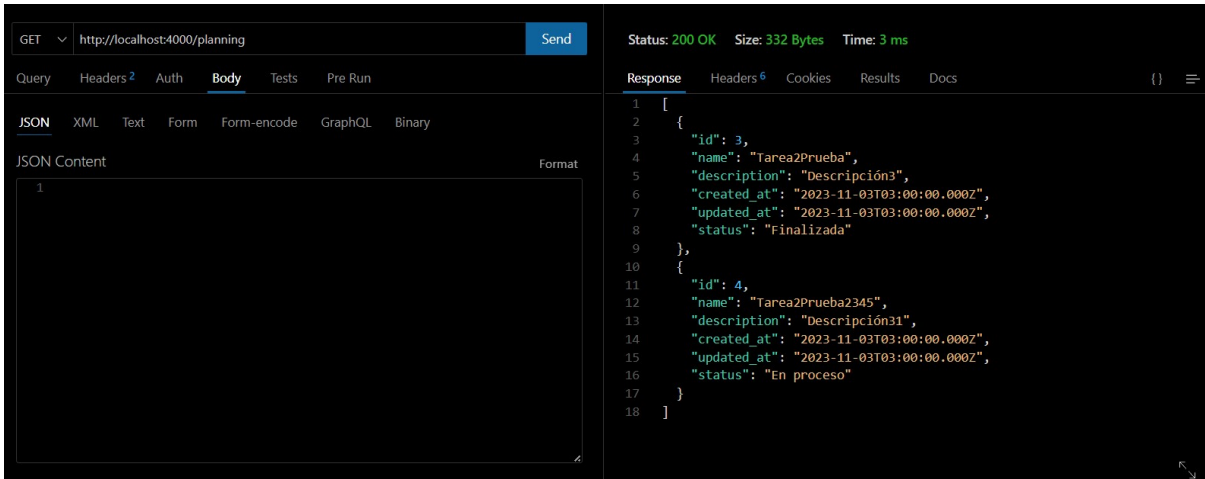
PASO 4 - DELETE

Lo siguiente para completar el CRUD es utilizar DELETE, por lo que se elimina el registro con el ID 1 y se hace un GET para verificar si se eliminó.



The screenshot shows a REST client interface with a DELETE request to `http://localhost:4000/planning/1`. The response is a 200 OK status with a size of 43 Bytes and a time of 17 ms. The response body is a JSON object:

```
1 {
2   "message": "Tarea eliminada correctamente"
3 }
```



GET

Query Headers² Auth **Body** Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

```

Status: 200 OK Size: 332 Bytes Time: 3 ms

Response Headers⁶ Cookies Results Docs {} ≡

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

```

PASO 5 - RESULTADO

El resultado final en la Base de Datos es el siguiente:

planning.planning: 2 filas en total (aprc >> Siguintes

id	name	description	created_at	updated_at	status
3	Tarea2Prueba	Descripción3	2023-11-03	2023-11-03	Finalizada
4	Tarea2Prueba2345	Descripción31	2023-11-03	2023-11-03	En proceso