

# Evidencias de funcionamiento

Se pensó en un sistema gestor de tareas porque ese fue el proyecto final de Desarrollo Web para una futura escalabilidad si es necesario.

## Añadir una nueva tarea por POST

The screenshot displays the Postman interface with a POST request to `http://localhost:8000/tarea` successfully executed. The request body is a JSON object with the following fields: `titulo`, `descr`, `estado`, and `prioridad`. The response body shows the same JSON object with an added `id` field, indicating a successful creation of the task.

**Request Details:**

- Method: POST
- URL: `http://localhost:8000/tarea`
- Body (JSON):

```
{  "titulo": "Mi nueva tarea",  "descr": "Descripción de la tarea",  "estado": "Pendiente",  "prioridad": "Alta"}
```

**Response Details:**

- Status: 200 OK
- Time: 485 ms
- Size: 273 B
- Body (JSON):

```
{  "id": 1,  "titulo": "Mi nueva tarea",  "descr": "Descripción de la tarea",  "estado": "Pendiente",  "prioridad": "Alta"}
```

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

djangoprueba

- Tables
- Views
- Stored Procedures
- Functions

primer\_proyecto

- Tables
  - tarea
    - Columns
    - Indexes
    - Foreign Keys
    - Triggers
- Views
- Stored Procedures
- Functions

sakila

sys

world

Query 1 x

Limit to 1000 rows

```
1 • select * from tarea;
```

2

Result Grid

	id	titulo	descr	estado	prioridad
▶ 1	1	Mi nueva tarea	Descripción de la tarea	Pendiente	Alta
*	NULL	NULL	NULL	NULL	NULL

tarea 1 x

Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	12:35:31	USE primer_proyecto	0 row(s) affected	0.000 sec
✗ 2	12:35:31	CREATE TABLE tarea ( id INT PRIMARY KEY AUTO_INCREMENT, titulo VARCHAR(25...	Error Code: 1050. Table 'tarea' already exists	0.031 sec
✓ 3	13:06:42	select * from tarea LIMIT 0, 1000	1 row(s) returned	0.015 sec / 0.000 sec

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

- ▼ djangoprueba
  - Tables
  - Views
  - Stored Procedures
  - Functions
- ▼ primer\_proyecto
  - Tables
    - ▼ tarea
      - Columns
      - Indexes
      - Foreign Keys
      - Triggers
    - Views
    - Stored Procedures
    - Functions
  - ▼ sakila
  - ▼ sys
  - ▼ world

Administration Schemas

Information

Schema:  
primer\_proyecto

Query 1 x

```
1 • select * from tarea;
```

Limit to 1000 rows

Result Grid

	id	titulo	descr	estado	prioridad
▶	1	Mi nueva tarea	Descripción de la tarea	Pendiente	Alta
	2	Primer Proyecto Spring Boot	Tengo que crear el repositorio GitHub y mandar ...	Pendiente	Alta
	3	Actualizar Java	Necesito actualizar SDK 15 a uno 17 o superior	Pendiente	Alta
	4	Día Zen	Tiempo para mi y en detox tecnologico no me v...	Pendiente	Media
*	NULL	NULL	NULL	NULL	NULL

tarea 2 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	12:35:31	USE primer_proyecto	0 row(s) affected	0.000 sec
✗ 2	12:35:31	CREATE TABLE tarea ( id INT PRIMARY KEY AUTO_INCREMENT, titulo VARCHAR(25...	Error Code: 1050. Table 'tarea' already exists	0.031 sec
✓ 3	13:06:42	select * from tarea LIMIT 0, 1000	1 row(s) returned	0.015 sec / 0.000 sec
✓ 4	13:12:58	select * from tarea LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec

## Consultar las tareas por medio de un GET

```
[
  {
    "id": 1,
    "titulo": "Mi nueva tarea",
    "descr": "Descripción de la tarea",
    "estado": "Pendiente",
    "prioridad": "Alta"
  },
  {
    "id": 2,
    "titulo": "Primer Proyecto Spring Boot",
    "descr": "Tengo que crear el repositorio GitHub y mandar mi trabajo al profesor",
    "estado": "Pendiente",
    "prioridad": "Alta"
  },
  {
    "id": 3,
    "titulo": "Actualizar Java",
    "descr": "Necesito actualizar SDK 15 a uno 17 o superior",
    "estado": "Pendiente",
    "prioridad": "Alta"
  },
  {
    "id": 4,
    "titulo": "Día Zen",
    "descr": "Tiempo para mi y en detox tecnologico no me vendría mal",
    "estado": "Pendiente",
    "prioridad": "Media"
  }
]
```

HomeWorkspacesExplore

Search Postman

Sign InCreate Account

You are using the Lightweight API Client, sign in or create an account to work with collections, environments and unlock all free features in Postman.

History

NewImport

GET http://localhost:8000/tarea

+

...

HTTP

http://localhost:8000/tarea

Save

</>

GET

http://localhost:8000/tarea

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Body

Cookies

Headers (5)

Test Results

Status: 200 OK

Time: 212 ms

Size: 713 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1  [
2    {
3      "id": 1,
4      "titulo": "Mi nueva tarea",
5      "descr": "Descripción de la tarea",
6      "estado": "Pendiente",
7      "prioridad": "Alta"
8    },
9    {
10     "id": 2,
11     "titulo": "Primer Proyecto Spring Boot",
12     "descr": "Tengo que crear el repositorio GitHub y mandar mi trabajo al profesor",
13     "estado": "Pendiente",
14     "prioridad": "Alta"
15   },
16   {
17     "id": 3,
18     "titulo": "Actualizar Java",
19     "descr": "Necesito actualizar SDK 15 a uno 17 o superior",
```

# Buscar una tarea por id específico usando GET

The screenshot shows the Postman interface. At the top, there's a navigation bar with 'Home', 'Workspaces', and 'Explore'. A search bar is labeled 'Search Postman'. On the right, there are links for 'Sign In' and 'Create Account'. A blue banner below the navigation bar states: 'You are using the Lightweight API Client, sign in or create an account to work with collections, environments and unlock all free features in Postman.'

The main interface is divided into three panels. The left panel, titled 'History', shows a list of recent requests under the 'Today' tab. The middle panel shows the details of a selected GET request to 'http://localhost:8000/tarea/1'. The right panel shows the response body in JSON format.

**History Panel:**

- GET http://localhost:8000/tarea/1
- GET http://localhost:8000/tarea
- POST http://localhost:8000/tarea
- POST http://localhost:8000/tarea
- POST http://localhost:8000/tarea
- POST http://localhost:8000/tarea
- POST http://localhost:8000/tarea
- POST http://localhost:8000/tarea
- GET http://localhost:8000/tarea
- GET http://localhost:8000/tarea

**Request Panel:**

Method: GET, URL: http://localhost:8000/tarea/1

Params, Authorization, Headers (8), Body, Pre-request Script, Tests, Settings, Cookies

**Response Panel:**

Status: 200 OK, Time: 262 ms, Size: 273 B

Body (JSON):

```
{
  "id": 1,
  "titulo": "Mi nueva tarea",
  "descr": "Descripción de la tarea",
  "estado": "Pendiente",
  "prioridad": "Alta"
}
```

At the bottom, there's a status bar showing 'Console' and 'Not connected to a Postman account'.

# Marcar una tarea como completada por medio de PUT [Mejor usar PATCH]

The screenshot displays the Postman API client interface. At the top, there's a navigation bar with 'Home', 'Workspaces', and 'Explore' links, along with a search bar and buttons for 'Sign In' and 'Create Account'. A blue banner below the navigation bar states: 'You are using the Lightweight API Client, sign in or create an account to work with collections, environments and unlock all free features in Postman.'

The main interface is divided into three sections:

- History:** A list of recent requests on the left side, including PUT and GET requests to 'http://localhost:8000/tarea/1/completar' and 'http://localhost:8000/tarea/1'.
- Request Editor:** The central area where a PUT request is configured. The URL is 'http://localhost:8000/tarea/1/completar'. The method is 'PUT'. The body is set to 'JSON'.
- Response:** The bottom section showing the response of the request. The status is '200 OK', the time is '341 ms', and the size is '274 B'. The response body is displayed in 'Pretty' format as a JSON object.

The JSON response body is as follows:

```
1 {
2   "id": 1,
3   "titulo": "Mi nueva tarea",
4   "desc": "Descripción de la tarea",
5   "estado": "completada",
6   "prioridad": "Alta"
7 }
```