

# Manual: Funcionamiento del Patrón MVC en Spring Boot (CRUD de Artículos)

## ¿Qué es MVC?

MVC es un patrón de arquitectura de software que separa una aplicación en tres componentes principales: Modelo (Model), Vista (View) y Controlador (Controller). En aplicaciones Spring Boot que exponen APIs REST, el foco está en el flujo entre el Cliente, el Controlador, el Servicio, el Repositorio y la Base de Datos.

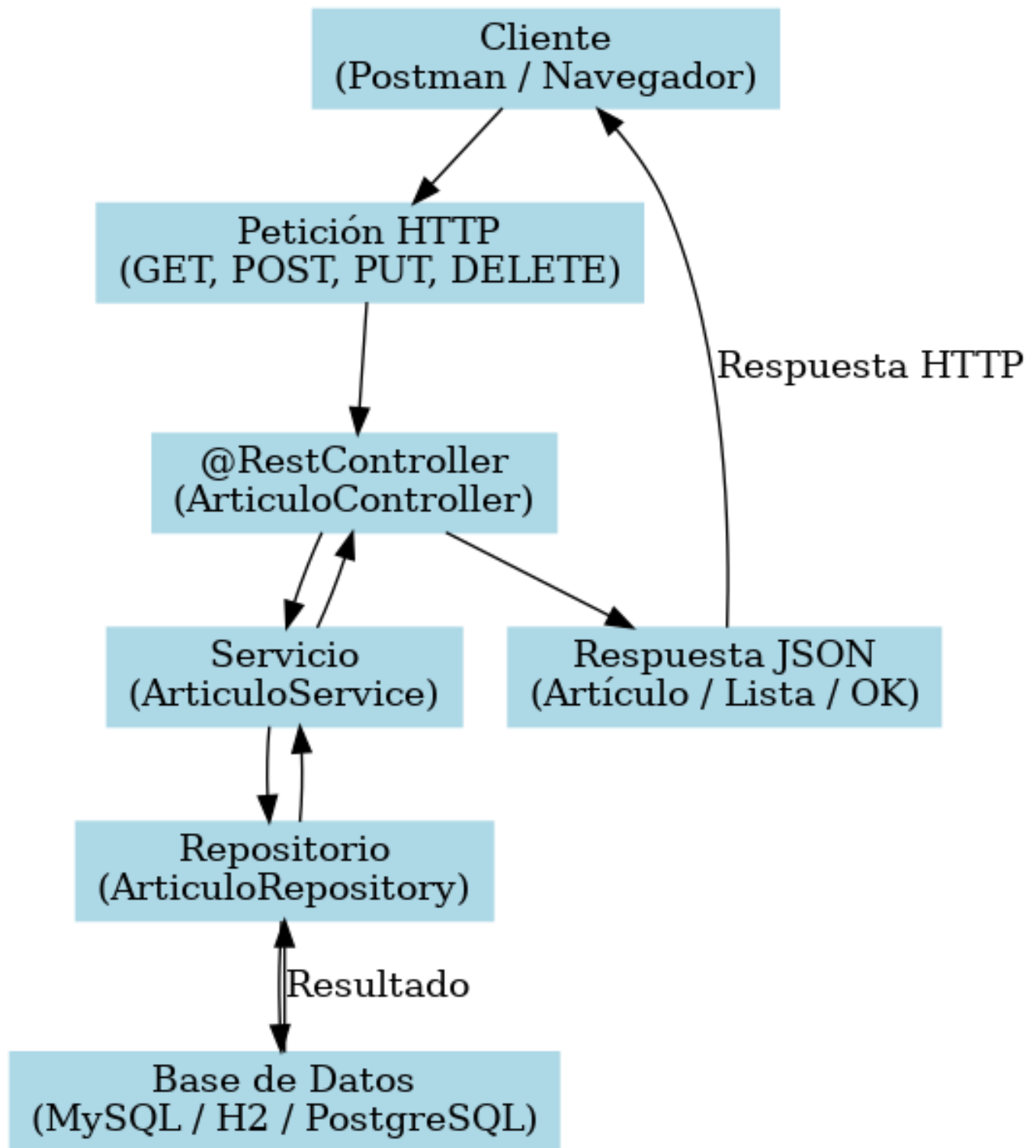
## Flujo del CRUD de Artículos en Spring Boot

1. El Cliente (por ejemplo, Postman o un navegador) realiza una solicitud HTTP (GET, POST, PUT o DELETE).
2. El Controlador (@RestController) recibe la solicitud y llama al Servicio.
3. El Servicio (ArticuloService) contiene la lógica del negocio y delega al Repositorio.
4. El Repositorio (ArticuloRepository) accede a la base de datos para realizar operaciones CRUD.
5. La Base de Datos guarda o recupera la información.
6. La respuesta (un artículo, lista, o estado OK) vuelve desde el Repositorio al Servicio, luego al Controlador.
7. Finalmente, el Controlador responde al Cliente con datos en formato JSON.

## Diagrama de Flujo

A continuación se presenta el diagrama de flujo del funcionamiento del patrón MVC aplicado al CRUD de artículos en Spring Boot:

## Manual: Funcionamiento del Patrón MVC en Spring Boot (CRUD de Artículos)



### Roles de cada componente

- Controlador: maneja las solicitudes HTTP entrantes.
- Servicio: contiene la lógica de negocio.
- Repositorio: interactúa con la base de datos utilizando Spring Data JPA.
- Modelo (Entidad): representa la estructura de los datos, por ejemplo, la clase Articulo.

## Manual: Funcionamiento del Patrón MVC en Spring Boot (CRUD de Artículos)

### Ejemplo aplicado (POST /articulos)

1. El cliente envía un JSON con los datos del artículo.
2. El controlador recibe el JSON y lo pasa al servicio.
3. El servicio llama al repositorio para guardar el artículo.
4. El repositorio persiste el artículo en la base de datos.
5. La base de datos responde, y se propaga la respuesta al cliente como JSON.