



**Escuela Politécnica Nacional**  
**Facultad de Ingeniería de Sistemas - Software**  
**Construcción y Evolución de Software**



**Integrantes:** Daniel Carvajal  
Mónica Lucio  
Lisbeth Romo

**Paralelo:** GR2SW

**Fecha:** 14 de septiembre de 2023

## **Proyecto Segundo Bimestre**

### **Contenido**

API Desarrollada .....	2
Documentación de la API de Visualización de Álbumes Musicales .....	3
Manual de Usuario de la API.....	8
Uso de Herramienta DevOps.....	17

## API Desarrollada

### **Tipo de API:** REST API

**Breve Descripción del tipo de API:** Es un conjunto de reglas y protocolos que permite que diferentes aplicaciones o sistemas se comuniquen entre sí a través de la World Wide Web (WWW). Esta arquitectura se basa en los principios de Representational State Transfer (REST), que es un estilo arquitectónico que se utiliza para diseñar servicios web y aplicaciones web que sean simples, escalables y fáciles de mantener.

Las REST APIs utilizan el protocolo HTTP (Hypertext Transfer Protocol) para realizar operaciones en recursos o datos, que pueden ser accedidos y manipulados a través de URLs (Uniform Resource Locators). Cada recurso tiene una URL única y se puede acceder a él utilizando los métodos estándar de HTTP, como GET (para recuperar datos), POST (para crear nuevos recursos), PUT (para actualizar recursos existentes) y DELETE (para eliminar recursos).

Las REST APIs son ampliamente utilizadas en el desarrollo de aplicaciones web y servicios web debido a su simplicidad, facilidad de uso y capacidad de escalabilidad. Son comunes en aplicaciones web, servicios de terceros, aplicaciones móviles y otros casos de uso donde se requiere la comunicación entre sistemas a través de la web.

# Documentación de la API de Visualización de Álbumes Musicales

## Introducción

La API REST de Almacenamiento de Álbumes Musicales es una aplicación desarrollada por Daniel Carvajal, Mónica Lucio y Lisbeth Romo para gestionar información sobre álbumes musicales. Esta API proporciona funcionalidades para agregar, ver, eliminar y buscar álbumes musicales en una memoria temporal.

## Enlace al Repositorio con el Código de la API:

[https://github.com/DanSet2000/creacion\\_api.git](https://github.com/DanSet2000/creacion_api.git)

## URL Base

La URL base para esta API es `http://localhost:80` o la dirección de tu servidor si se encuentra en producción.

## Recursos

### *1. Obtener Información General*

**Endpoint:** '/'

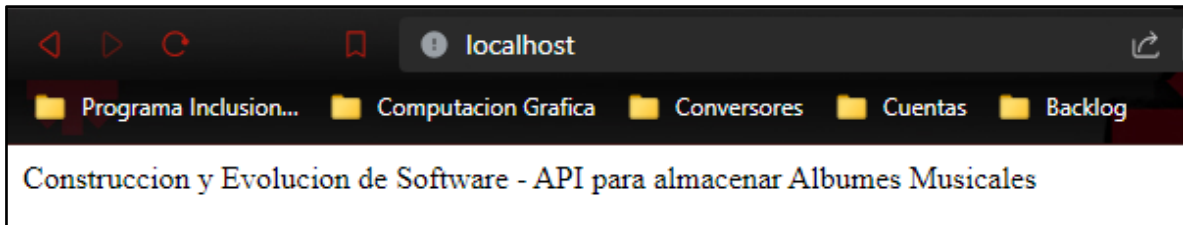
**Método:** GET

**Descripción:** Obtiene información general sobre la API desarrollada.

**Ejemplo de Solicitud:**

```
http
GET /
```

**Respuesta Esperada:**



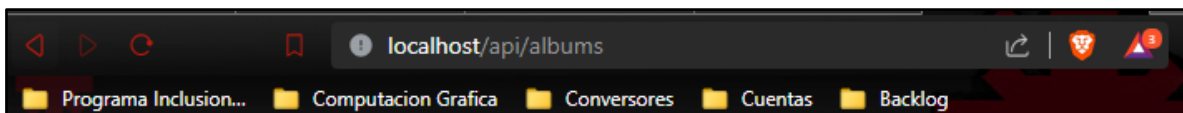
## 2. *Obtener Todos los Álbumes Registrados*

**Endpoint:** '/api/albums'

**Método:** GET

**Descripción:** Obtiene una lista de todos los álbumes musicales registrados en la memoria.

**Ejemplo de Solicitud:**



**Respuesta Esperada:**

```
[
  {
    "id": 1,
    "nombre": "Californication",
    "autor": "Red Hot Chili Peppers",
    "genero": "Funk Rock",
    "anio_lanzamiento": 1999
  },
  {
    "id": 2,
    "nombre": "Invincible",
    "autor": "Michael Jackson",
    "genero": "Pop",
    "anio_lanzamiento": 2001
  },
  // ... otros álbumes ...
]
```

### 3. *Obtener un Álbum por su ID*

**Endpoint:** '/api/albums/{id}'

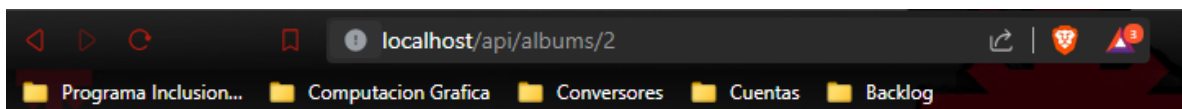
**Método:** GET

**Descripción:** Obtiene detalles de un álbum musical específico según su ID.

**Parámetros:**

**{id}** (entero, obligatorio): El ID único del álbum que se desea obtener.

**Ejemplo de Solicitud:**



**Respuesta Esperada:**

```
{
  "id": 1,
  "nombre": "Californication",
  "autor": "Red Hot Chili Peppers",
  "genero": "Funk Rock",
  "anio_lanzamiento": 1999
}
```

### 4. *Agregar un Nuevo Álbum*

**Endpoint:** '/api/albums'

**Método:** POST

**Descripción:** Agrega un nuevo álbum musical a la memoria.

**Parámetros:**

**nombre** (cadena, obligatorio): **El nombre del álbum.**

**autor** (cadena, obligatorio): **El autor o artista del álbum.**

**genero** (cadena, obligatorio): **El género musical del álbum.**

**anio\_lanzamiento** (entero, obligatorio): **El año de lanzamiento del álbum.**

**Cuerpo de la Solicitud:**

```
{  
  "nombre": "Nuevo Álbum",  
  "autor": "Artista Nuevo",  
  "genero": "Pop",  
  "anio_lanzamiento": 2023  
}
```

**Respuesta Esperada:**

```
{  
  "id": 6,  
  "nombre": "Nuevo Álbum",  
  "autor": "Artista Nuevo",  
  "genero": "Pop",  
  "anio_lanzamiento": 2023  
}
```

### ***5. Eliminar un Álbum por su ID***

**Endpoint:** `/api/albums/{id}`

**Método:** DELETE

**Descripción:** Elimina un álbum musical de la memoria según su ID.

**Parámetros:**

**{id}** (entero, obligatorio): El ID único del álbum que se desea eliminar.

**Ejemplo de Solicitud:**

```
http
```

```
DELETE /api/albums/6
```

### Respuesta Esperada:

```
{
  "id": 6,
  "nombre": "Nuevo Álbum",
  "autor": "Artista Nuevo",
  "genero": "Pop",
  "anio_lanzamiento": 2023
}
```

### Manejo de Errores

La API utiliza códigos de estado HTTP estándar para indicar el éxito o el fracaso de una solicitud. En caso de un error, se proporcionarán detalles adicionales del error en el cuerpo de la respuesta.

Aquí hay algunos códigos de error comunes:

**400 “Solicitud Incorrecta”** - La solicitud es inválida o falta información obligatoria.

**404 “No Encontrado”** - El recurso solicitado no existe.

**500 “Error Interno del Servidor”** - Ocurrió un error interno en el servidor.

### Conclusiones

La API de Almacenamiento de Álbumes Musicales proporciona una forma sencilla de gestionar información sobre álbumes musicales. Puedes utilizarla para agregar, ver y eliminar álbumes en una memoria temporal. Si tienes alguna pregunta o necesitas asistencia adicional, por favor, ponte en contacto con los desarrolladores.

¡Gracias por utilizar nuestra API!

# Manual de Usuario de la API

¡Bienvenido al Manual de Usuario de la API de Almacenamiento de Álbumes Musicales! Este manual te proporcionará toda la información necesaria para interactuar con la API y realizar operaciones como agregar, ver y eliminar álbumes musicales.

## Introducción

La API de Almacenamiento de Álbumes Musicales es una aplicación diseñada para gestionar información sobre álbumes de música. Permite realizar operaciones como agregar nuevos álbumes, ver detalles de álbumes existentes y eliminar álbumes de la memoria.

## Acceso a la API

La API se encuentra disponible en la siguiente URL base:

- URL Base: **http://localhost:80** (o la dirección de tu servidor si está en producción)

Asegúrate de que el servidor esté en ejecución para acceder a la API. Para esto, dirígete a la terminal de ejecución en el entorno de desarrollo que estes ejecutando y ejecuta el comando

**node** index.js

**Nota:** Se recomienda utilizar Visual Studio Code

## Código del Programa

En caso de encontrarse con un entorno de desarrollador o querer realizar modificaciones a la API por conveniencia personal, se puede encontrar el código fuente en el siguiente repositorio, cabe destacar que la API fue desarrollada en lenguaje JavaScript con ayuda del IDE Visual Studio Code (se requiere tener instalado node.js en el computador para su ejecución).

[https://github.com/DanSet2000/creacion\\_api](https://github.com/DanSet2000/creacion_api)



## Código Utilizado para la API

```
// Importacion de Librerias
const express = require('express');
const app = express();

app.use(express.json());

// Puerto para Trabajar

const port = process.env.port || 80;
app.listen(port, () => console.log(`Escuchando en el puerto ${port}`));

// Elementos de la API con sus atributos
const albums = [
  {id: 1, nombre: 'Californication', autor: 'Red Hot Chili Peppers', genero: 'Funk Rock', anio_lanzamiento: 1999},
  {id: 2, nombre: 'Invincible', autor: 'Michael Jackson', genero: 'Pop', anio_lanzamiento: 2001},
  {id: 3, nombre: 'Nevermind', autor: 'Nirvana', genero: 'Rock', anio_lanzamiento: 1991},
  {id: 4, nombre: 'Starboy', autor: 'The Weeknd', genero: 'R&B', anio_lanzamiento: 2016},
  {id: 5, nombre: 'Facelift', autor: 'Alice in Chains', genero: 'Rock', anio_lanzamiento: 1990}
];
```

```
// Declaracion de Peticiones

app.get('/', (req, res) =>{
  res.send('Construccion y Evolucion de Software -' + ' API para almacenar Albumes Musicales')
});

// Ver todos los Albums Registrados
app.get('/api/albums', (req, res) =>{
  res.send(albums);
});

// Observar Album Por su Id Registrado
app.get('/api/albums/:id', (req, res) =>{
  const album = albums.find(c => c.id === parseInt(req.params.id));

  if (!album) return res.status(404).send("Album No Encontrado en Memoria");
  else res.send(album);
});

// Agregar un Nuevo Album
app.post('/api/albums', (req, res) => {
  const album = {
    id: albums.length + 1,
    nombre: req.body.nombre,
    autor: req.body.autor,
    genero: req.body.genero,
    anio_lanzamiento: parseInt(req.body.anio_lanzamiento)
  };

  albums.push(album);
  res.send(album);
});
```

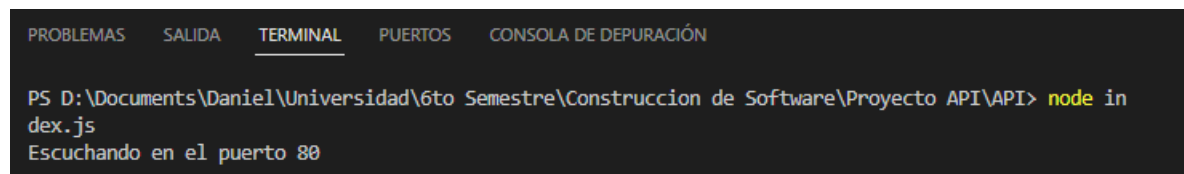
```
// Eliminar un Album de Memoria por su ID
app.delete('/api/albums/:id', (req, res) =>{
  const album = albums.find(c => c.id === parseInt(req.params.id));

  if (!album) return res.status(404).send("Album No Encontrado en Memoria");

  const index = albums.indexOf(album);
  albums.splice(index, 1);
  res.send(album);
});
```

## Funciones de la Aplicación

Una vez comprendido el código y su forma de ejecución, con el servidor levantado, en este caso el puerto 80, se puede empezar con la explicación de cómo funciona la API para la organización y gestión de álbumes musicales.

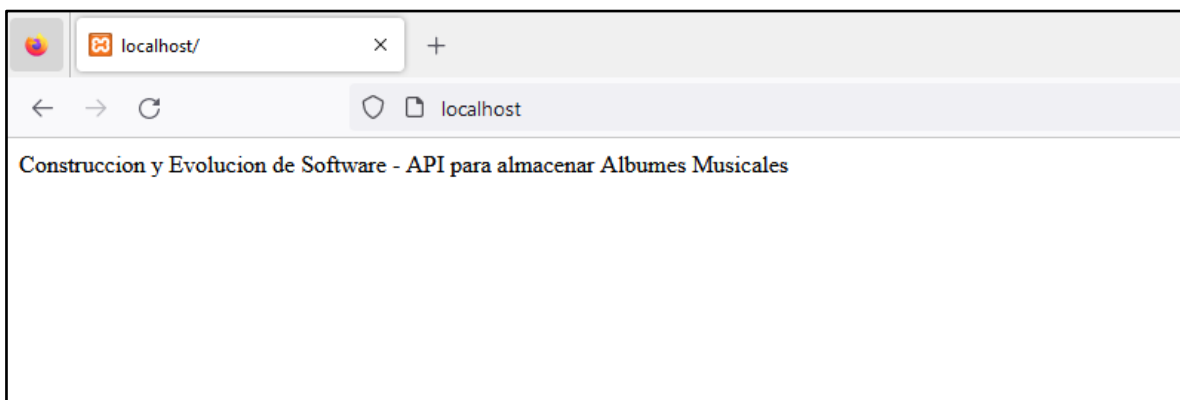


```
PROBLEMAS  SALIDA  TERMINAL  PUERTOS  CONSOLA DE DEPURACIÓN

PS D:\Documents\Daniel\Universidad\6to Semestre\Construccion de Software\Proyecto API\API> node in
dex.js
Escuchando en el puerto 80
```

### *Obtener Información General*

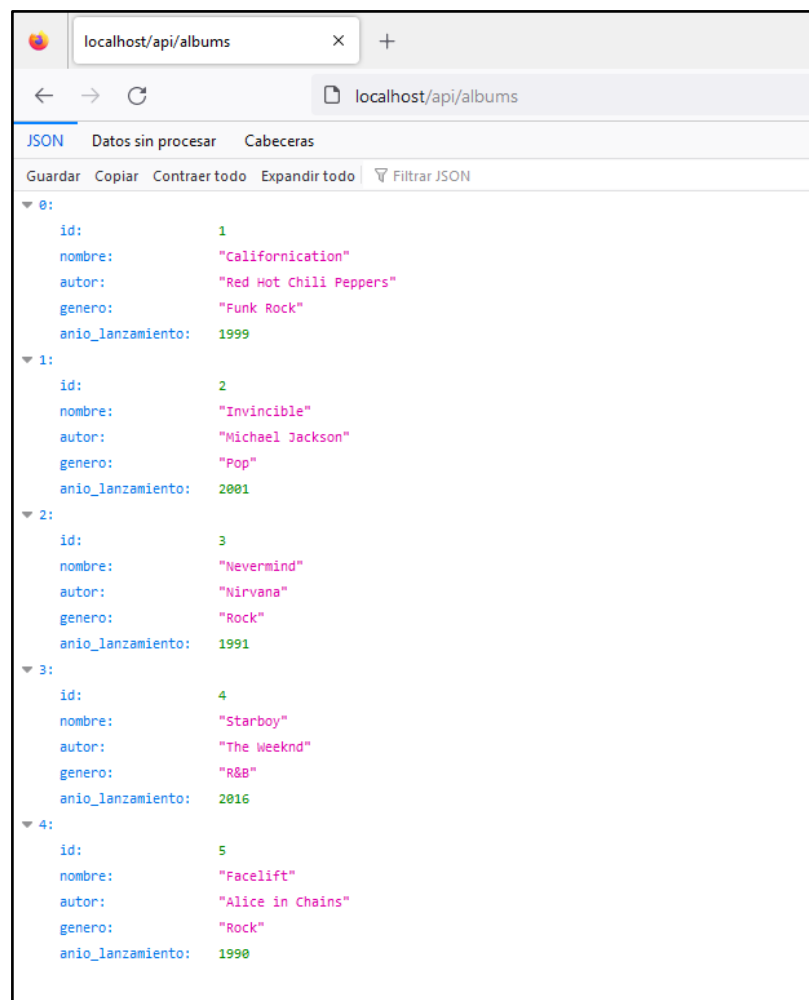
Para ello se necesita ejecutar en el navegador de preferencia por el usuario la URL “localhost”, la cual simplemente mostrará una información general sobre la API, este paso puede servir para comprobar si el servidor se levantó correctamente como se puede observar a continuación.



### *Obtener todos los Álbumes Registrados en Memoria*

Para esto se tiene que ingresar la URL “http://localhost/api/albums” la cual nos permite observar todos los álbumes que se encuentren guardados en la memoria local por medio de un archivo en formato JSON.

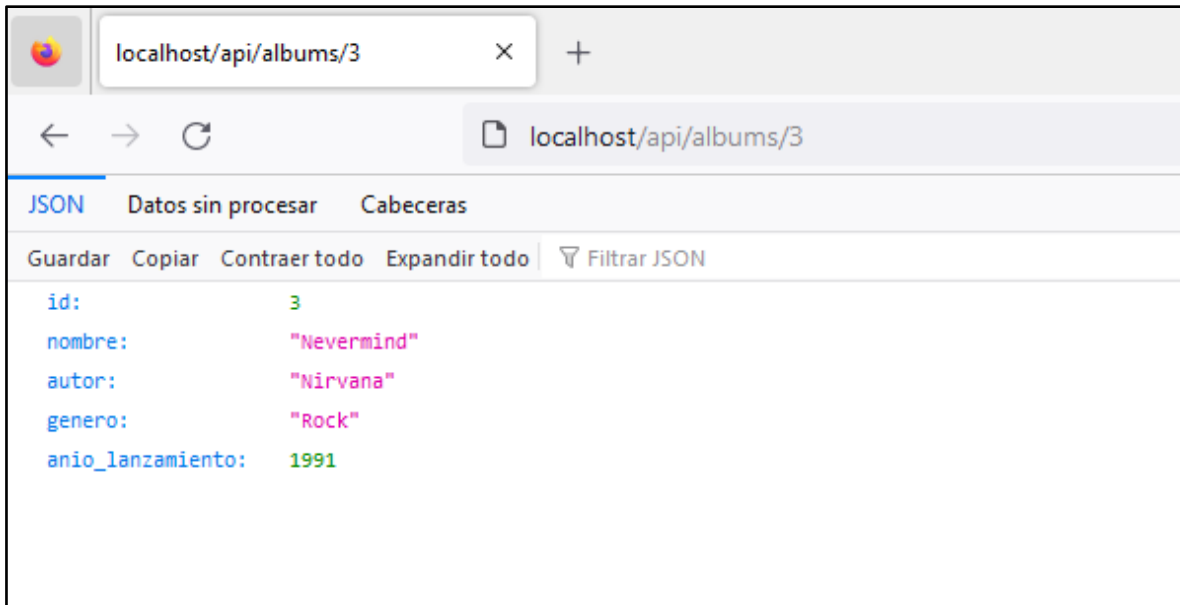
```
// Ver todos los Albums Registrados
app.get('/api/albums', (req, res) =>{
  res.send(albums);
});
```



### *Obtener un álbum por su ID*

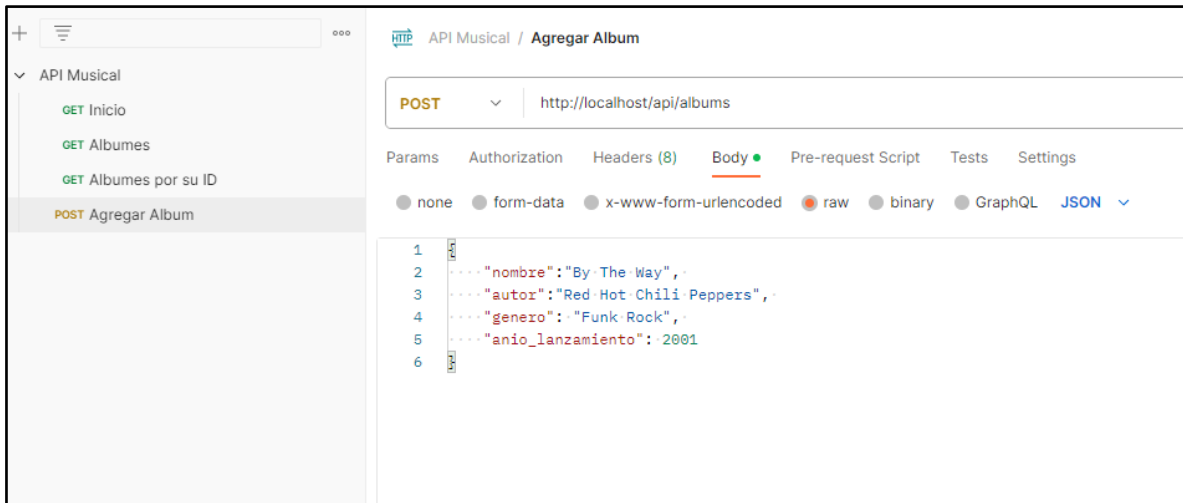
Para poder observar cada uno de los álbumes disponibles en la memoria local del servidor se debe ingresar la siguiente URL “http://localhost/api/albums/{id}” en donde el id es el número

del álbum almacenado en la memoria. Por ejemplo, si queremos solamente ver el álbum denominado “Nevermind” cuyo id es el número 3 lo debemos hacer de la siguiente manera.

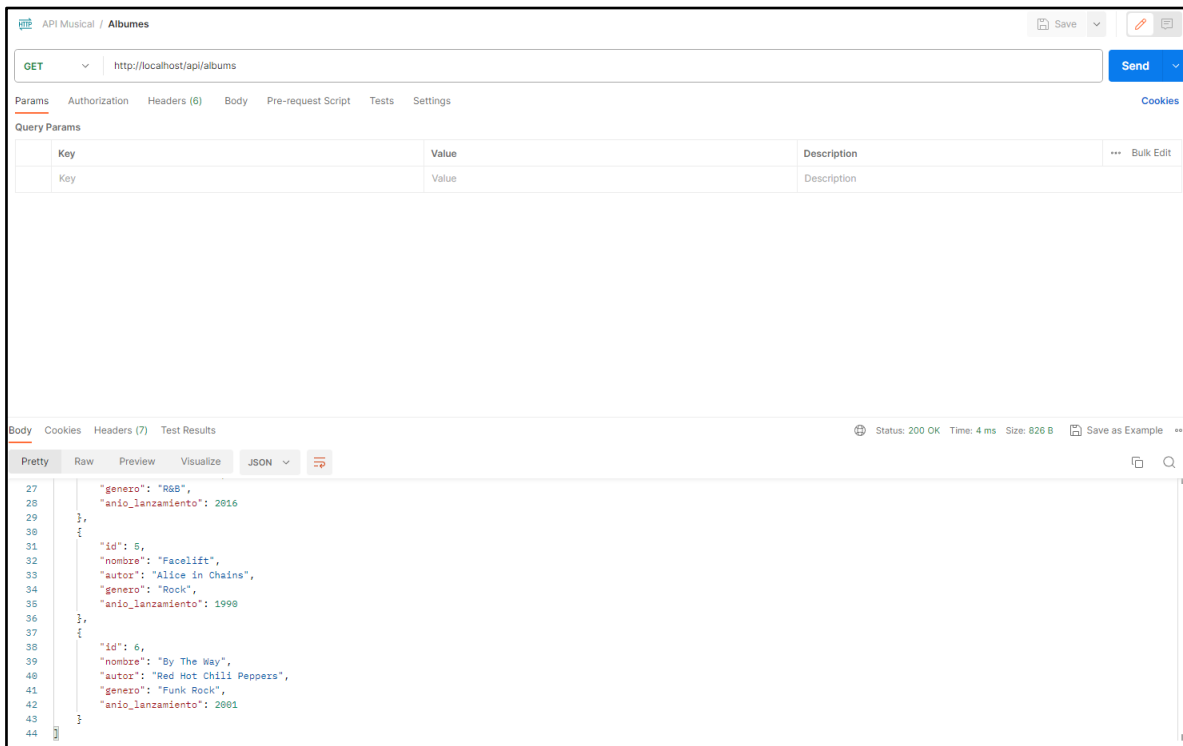


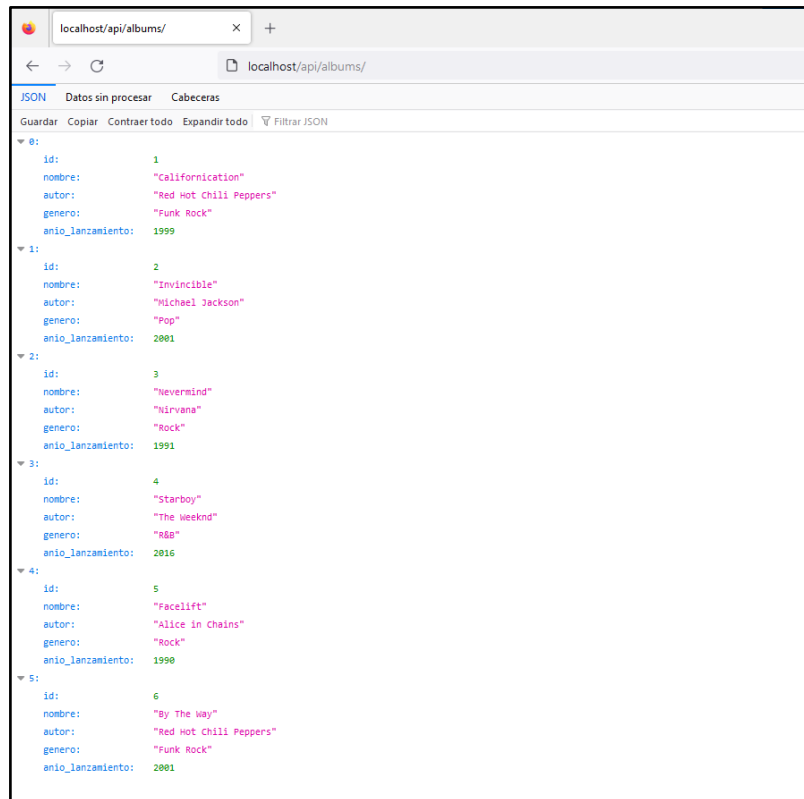
### *Agregar un Nuevo Álbum*

Para este y el siguiente método nos ayudamos del programa PostMan, el cual puede ser descargado e instalado fácilmente de manera gratuita. Para su ejecución simplemente debemos abrir el programa, crear una nueva petición de tipo **POST**, la cual denominaremos “Agregar Álbum” y en su cuerpo agregamos cada uno de los parámetros predefinidos previamente, es decir el nombre, autor, género y año de lanzamiento. (El id se genera auto incrementalmente de manera automática). Estos datos deben ser ingresados por medio del formato JSON en el cuerpo y finalmente se envía la petición al URL esperando la respuesta que debería ser el nuevo elemento ya almacenado en la memoria.



Para comprobar que se ha enviado correctamente, podemos volver a observar todos los álbumes en memoria, para esto ahora lo vamos a comprobar de ambas formas tanto en PostMan como por medio de la URL en el navegador.

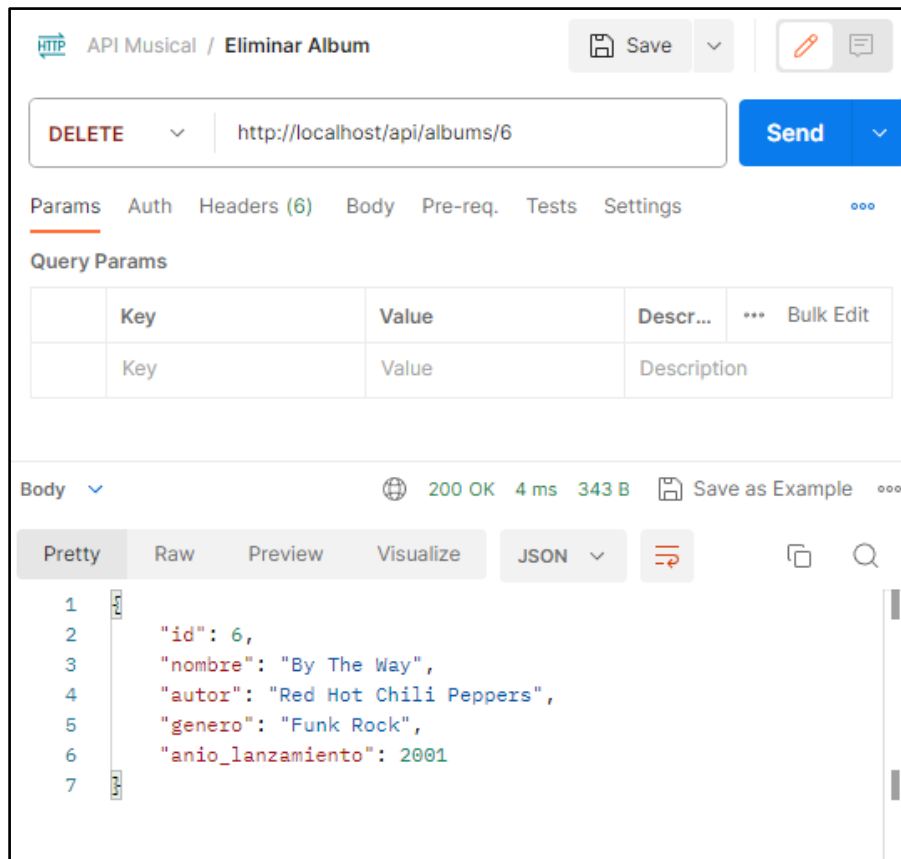




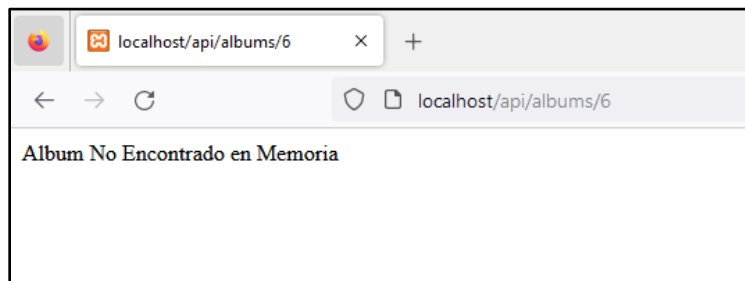
Se puede apreciar que en ambas maneras se visualiza correctamente el método POST para agregar nuevos álbumes a la memoria.

### ***Eliminar un Álbum por su ID***

Como en el caso anterior nos ayudamos de PostMan pero ahora usamos el método HTTP DELETE, ingresando en el cuerpo de la petición simplemente el id del álbum el cual deseamos eliminar de la memoria, para este ejemplo eliminaremos el último álbum ingresado en el sistema cuyo id es el “6” y su nombre es “By the Way”.



Para comprobar que se ha eliminado correctamente simplemente volvemos a listar todos los álbumes o podemos buscar el álbum eliminado por su ID y la API nos dirá que no existe un álbum con ese id en su memoria.



## Manejo de Errores

La API maneja los siguientes errores:

**404 (No Encontrado)** - El recurso solicitado no existe.

**500 (Error Interno del Servidor)** - Ocurrió un error interno en el servidor.

## Conclusión

La API de Almacenamiento de Álbumes Musicales proporciona una forma sencilla de gestionar información sobre álbumes musicales. Puedes utilizarla para agregar, ver y eliminar álbumes en una memoria temporal. Si tienes alguna pregunta o necesitas asistencia adicional, por favor, ponte en contacto con los desarrolladores.

¡Gracias por utilizar nuestra API!



## **Uso de Herramienta DevOps**

### **¿Qué es DevOps?**

DevOps es una práctica y cultura de colaboración entre los equipos de desarrollo (Dev) y operaciones (Ops) que busca automatizar y agilizar la entrega de software. El objetivo principal de DevOps es acortar el ciclo de vida del desarrollo de software, desde la planificación y codificación hasta la implementación y operación, para brindar valor de manera más rápida y confiable a los usuarios finales. Esto se logra a través de la automatización de procesos, la integración continua, la entrega continua, la monitorización y la colaboración eficaz entre equipos.

### **¿Para qué se usan?**

DevOps se usa para agilizar el desarrollo, la implementación y la operación de software. Al combinar los equipos de desarrollo y operaciones, DevOps ayuda a las organizaciones a crear y lanzar software más rápido, con mayor calidad y confiabilidad.

Los beneficios específicos de DevOps incluyen:

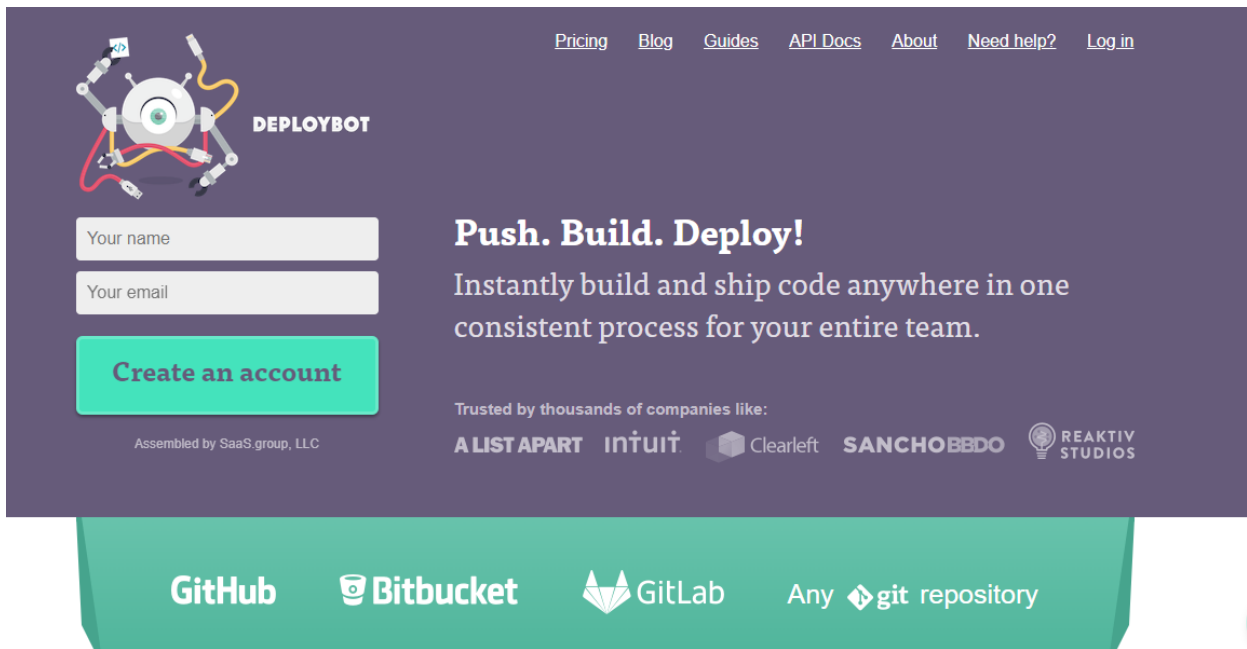
- Tiempo de comercialización más rápido: DevOps puede ayudar a las organizaciones a reducir el tiempo de comercialización de nuevas funciones y productos. Esto es posible gracias a la automatización de procesos, la colaboración entre equipos y la adopción de una cultura de experimentación.
- Mejor calidad del software: DevOps puede ayudar a las organizaciones a mejorar la calidad del software mediante la implementación de pruebas e integración continuas (CI/CD). Esto ayuda a identificar y corregir errores antes de que lleguen a los clientes.
- Reducción de costos: DevOps puede ayudar a las organizaciones a reducir los costos al automatizar procesos y eliminar la necesidad de tareas manuales.
- Mejor satisfacción del cliente: DevOps puede ayudar a las organizaciones a mejorar la satisfacción del cliente al proporcionar nuevas funciones y productos con mayor rapidez.

## DeployBot

DeployBot es una herramienta de implementación de código que permite a los equipos de desarrollo enviar y construir código en cualquier lugar en un proceso consistente. Puede trabajar con su repositorio Git existente para implementar rápidamente nuevo código sin tiempo de inactividad. DeployBot también proporciona una variedad de herramientas para múltiples entornos, lo que le permite enviar código desde diferentes ramas a uno o muchos servidores simultáneamente. Además, puede ejecutar o compilar cualquier código en los servidores de DeployBot durante la implementación, utilizando contenedores Docker predefinidos o completamente personalizados. También puede ejecutar cualquier script de shell en su propio servidor antes, después o durante la implementación.

DeployBot ofrece una variedad de características para ayudar a las organizaciones a automatizar sus despliegues, incluyendo:

- Automatización de tareas: DeployBot puede automatizar una variedad de tareas de despliegue, incluyendo la compilación de código, la implementación de código y la configuración de servidores.
- Integración con sistemas existentes: DeployBot puede integrarse con una variedad de sistemas existentes, incluyendo sistemas de control de versiones, servidores y aplicaciones.
- Control de versiones: DeployBot puede rastrear los cambios en los despliegues para facilitar la identificación y el seguimiento de problemas.
- Notificaciones: DeployBot puede enviar notificaciones sobre los despliegues para mantener a los equipos actualizados.



## Primeros pasos con DeployBot

En esta guía cubriremos los pasos para la implementación:

- Conexión al repositorio
- Creando un ambiente
- Agregar un servidor
- Desplegar

### Conexión al repositorio

Hay dos formas de agregar un repositorio a su cuenta DeployBot:

- Autorice su cuenta de GitHub o Bitbucket y elija el repositorio para conectarse.
- Conecte un repositorio Git autohospedado con una URL HTTPS o SSH. Esto funciona para cualquier repositorio alojado, como Beanstalk o GitLab.

Al agregar un repositorio, agregue un nombre para el repositorio en DeployBot y aplique una etiqueta de color. Cuando esté listo para agregar el repositorio, haga clic en "Conectar". Si

tiene usuarios en DeployBot, el siguiente paso es otorgarles permiso para ver el repositorio y ejecutar implementaciones.

The screenshot shows the 'Connect a repository' interface. At the top, there's a header 'Connect a repository' with two tabs: '1 General Settings' and '2 Users & Permissions'. Below the tabs, there are three buttons: 'GitHub', 'Bitbucket', and 'Self-hosted'. The 'GitHub' button is selected. Below these buttons, there's a section for the user 'briankerr' with a 'Connect new account' button. A note states: 'You need to have administrator permissions in the repository to connect to it. Why? You will be able to deauthorize your GitHub account later on the [Repositories](#) page after you connect to the desired repository.' Below this, there's a 'Title:' input field and a 'Color label' section with a dropdown menu showing 'No color label' and a color picker. At the bottom, there's a 'Connect' button and a 'Go back' link.

## Creando un ambiente

Después de agregar su repositorio, el siguiente paso es agregar un entorno. Este es un lugar para agregar servidores y administrar implementaciones.

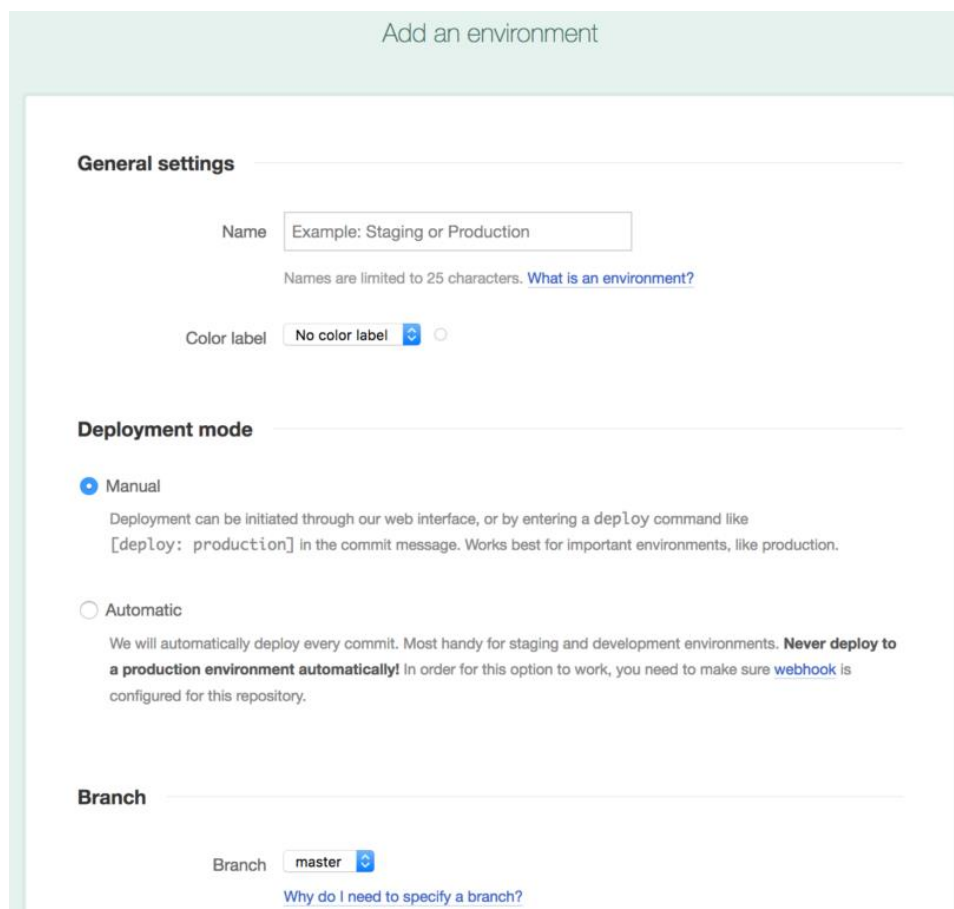
Le recomendamos que siempre tenga un entorno de prueba o de ensayo, que combine con su entorno de producción. Esto crea un flujo de trabajo de implementación que es estable y repetible sin implementar directamente en Producción. También creemos que las implementaciones de producción siempre deben ser manuales. Tener un humano involucrado

garantiza que las implementaciones de producción no se ejecuten en momentos inoportunos. Si algo sale mal, la persona que maneja la implementación puede deshacer los cambios.

Al agregar cualquier entorno, hay algunas configuraciones necesarias:

- Nombre (i.e, alto secreto, puesta en escena, producción, etc.)
- Modo de implementación: Manual o Automático
- Elija la rama desde la que el entorno implementa las confirmaciones
- Opcional: agregue una etiqueta de color
- Opcional: disparadores

¿Cambiaste de opinión sobre algunas configuraciones? Cada entorno tiene una página de configuración para actualizar la configuración más adelante.



The screenshot shows a web form titled "Add an environment". It is divided into three sections: "General settings", "Deployment mode", and "Branch".

- General settings:** Includes a "Name" input field with the placeholder text "Example: Staging or Production". Below it, a note states "Names are limited to 25 characters. [What is an environment?](#)". There is also a "Color label" section with a dropdown menu currently showing "No color label" and a small circular color selection icon.
- Deployment mode:** Features two radio button options. The "Manual" option is selected. Below it, text explains: "Deployment can be initiated through our web interface, or by entering a deploy command like `[deploy: production]` in the commit message. Works best for important environments, like production." The "Automatic" option is unselected, with text below stating: "We will automatically deploy every commit. Most handy for staging and development environments. **Never deploy to a production environment automatically!** In order for this option to work, you need to make sure [webhook](#) is configured for this repository."
- Branch:** Includes a dropdown menu currently set to "master". Below it, a link reads: "[Why do I need to specify a branch?](#)".

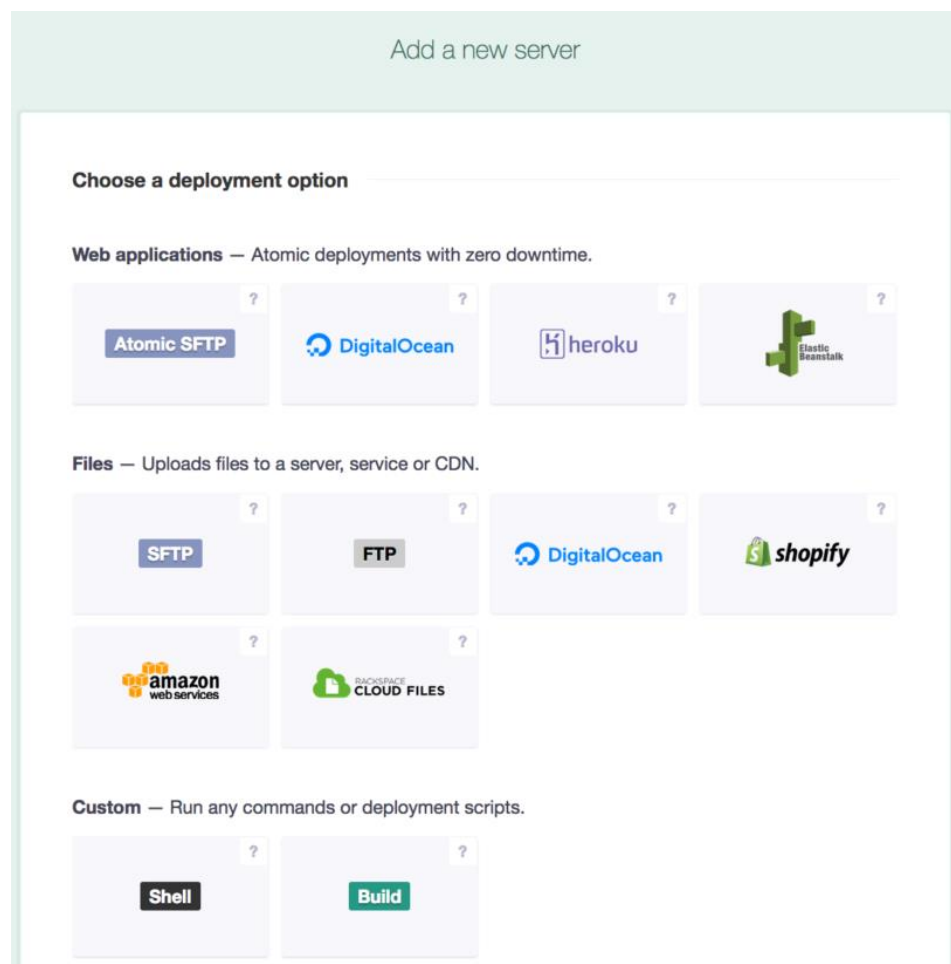
## Agregar un servidor

Después de agregar un entorno, el siguiente paso es agregar un servidor.

DeployBot admite implementaciones en DigitalOcean, Shopify, Heroku, AWS Elastic Beanstalk, AWS S3 y Rackspace Cloud Files. Implemente en cualquier otro lugar con servidores SFTP, FTP, Shell y Build.

Al configurar su servidor, dos configuraciones a tener en cuenta son:

- Ruta de origen: el directorio en la rama de su repositorio desde donde se implementa DeployBot.
- Ruta remota/destino/aplicación: la ubicación de implementación raíz en el servidor.



## Implementación

Con las implementaciones automáticas, la implementación es tan simple como enviar su código a su repositorio remoto. Si una rama que DeployBot está observando recibe una nueva confirmación, se implementa automáticamente.

Si su implementación es manual, hay dos lugares para activarla:

- En el Panel hay una sección de confirmaciones no implementadas.
- Vaya a la página Entorno y seleccione el botón "Implementar".

Luego estará en la nueva página de implementación donde podrá revisar los cambios que se implementarán y podrá editar diferentes opciones de implementación. Por ejemplo, personalice el mensaje de implementación (el valor predeterminado es el mensaje de confirmación). Cuando esté listo para implementar, haga clic en "Iniciar implementación".

Durante y después de la implementación, DeployBot proporcionará un registro de implementación. El registro le permite ver el orden de ejecución de los comandos y ser una base para solucionar problemas si hay algún error de shell.

New deployment to **Live**

Updated text fields on about us page  
0d56917c by Brian Kerr on December 06, 2016 at 6:04PM - [Change commit & options](#)

**Write a deployment note**

Keep your team informed by including a clear and detailed deployment message.

Updated text fields on about us page

**Review & Deployment**

You're about to deploy to the **Live** environment. Depending on the size of the changes, deployment could take some time.

Server	From	To	Commits
<b>FTP</b> Production Site	98 0c97e9fe ...	0d56917c	1

Preview the files to be deployed

Start deployment or [Go back](#)

## Bibliography

Khalid, T. (2022, mayo 17). *9 plataformas de automatización de implementación para aplicaciones modernas*. Geekflare.

<https://geekflare.com/es/deployment-automation-platforms/>

*¿Qué es DevOps y para qué sirve?* (2019). Netapp.com; NetApp.

<https://www.netapp.com/es/devops-solutions/what-is-devops/>

(S/f). Deploybot.com. Recuperado el 13 de septiembre de 2023, de

<https://deploybot.com/guides-code-deployment-tools/getting-started-with-deploybot..>