

## **Curso:**

# Programador Web Inicial-Front End Developer



Módulo 5:

# CMS y despliegue en Heroku

Unidad 4:

CRUD (parte 4) y Despliegue





## Presentación

En esta unidad incluiremos un buscador para las novedades en el administrador para facilitar la edición cuando tengamos varias cargadas. Por último desplegaremos nuestro sitio en la plataforma Heroku y de esta forma podremos compartir nuestra aplicación con el mundo.





## **Objetivos**

### Que los participantes logren...

- Comprender cómo se implementa la función de búsqueda en una aplicación.
- Conocer qué es y para qué sirve Heroku.
- Hacer un despliegue exitoso de una aplicación escrita en Node.js.





# **Bloques temáticos**

- 1. Buscador de novedades
- 2. Despliegue del sitio en Heroku.



## 1. Buscador de novedades

### Paso 1

Modificamos nuestro archivo **views/admin/novedades.hbs** para poder incluir nuestro diseño del buscador. Agregar el icono de la lupa (fa-search)

### Paso 2

En el archivo **models/novedadesModel.js** agregaremos la función **buscarNovedades** que buscará en los campos título, subtítulo y cuerpo lo que pasemos como parámetro.



```
async function buscarNovedades(busqueda) {
   var query = "select * from novedades where titulo like ? OR subtitulo like ?
        OR cuerpo like ? ";
   var rows = await pool.query(query, ['%' + busqueda + '%', '%' + busqueda +
        '%', '%' + busqueda + '%']);
   return rows;
}

module.exports = { getNovedades, insertNovedad, deleteNovedadById, getNovedadById,
   modificarNovedadById, buscarNovedades }
```

### Paso 3

En el archivo **routes/admin/novedades.js** modificamos el código de nuestro controlador para que, en caso de recibir el parámetro **q** por **url** realice la búsqueda en vez de listar todas las novedades disponibles. En cualquiera de los 2 casos pasamos la vista los **parámetros is\_search y q.** 

```
router.get('/', async function (req, res, next) {
    var novedades
    if (req.query.q === undefined) {
        novedades = await novedadesModel.getNovedades();
    } else {
        novedades = await novedadesModel.buscarNovedades(req.query.q);
    }
    res.render('admin/novedades', {
        layout: 'admin/layout',
        usuario: req.session.nombre,
        novedades,
        is_search: req.query.q !== undefined,
        q: req.query.q
    });
});
```



### Paso 4

Modificamos nuestro archivo **views/admin/novedades.hbs** para mostrar un mensaje al usuario en caso de que se haya realizado una búsqueda y no haya habido resultados.

```
{{#if is_search}}
    {{#unless novedades.length}}
    No se encontraron resultados para "<strong>{{q}}</strong>"
    {{/unless}}
{{/if}}
```

Finalmente, en caso de recibir la variable **q** la utilizamos como valor del input para que usuario tenga una referencia de lo que buscó.

```
<input class="form-control mr-sm-2" type="search"
placeholder="Search" aria-label="Search" name="q"
{{#if q}}value="{{q}}"{{/if}}>
```



## 2. Despliegue del sitio en Heroku

Heroku es uno de los PaaS más utilizados en la actualidad en entornos empresariales por su fuerte enfoque en resolver el despliegue de una aplicación. Además te permite manejar los servidores y sus configuraciones, escalamiento y la administración.

Podemos ver su sitio, servicios y costos en <a href="https://www.heroku.com/">https://www.heroku.com/</a>.

# Cuales son los requisitos para poder subir los archivos:

- Tener una cuenta
- Asegurarse de que el proyecto esté en un repositorio git local.
- Descargar la siguiente herramientas: https://devcenter.heroku.com/articles/heroku-cli#download-and-install.

### Pasos para subir la app

#### Paso 1

Escribir heroku login en la consola. En este paso no importa el path o ruta donde escribimos el comando, ya que el proceso de login será válido en toda nuestra computadora.



#### Paso 2

En el directorio de la aplicación escribimos heroku create. Este comando registra nuestra aplicación en el servicio de Heroku y le asigna un nombre al azar. Además agrega como remote de git el servidor que corresponda a nuestra aplicación.

### Paso 3

Para subir nuestra aplicación, debemos asegurarnos de tener todo commiteado localmente y luego hacer el push al repositorio que Heroku nos agregó. Para eso usamos el comando git push heroku master.

Al finalizar el proceso, en caso de ser exitoso, nos mostrará un link a nuestra aplicación

### Pasos para trabajar con la BD

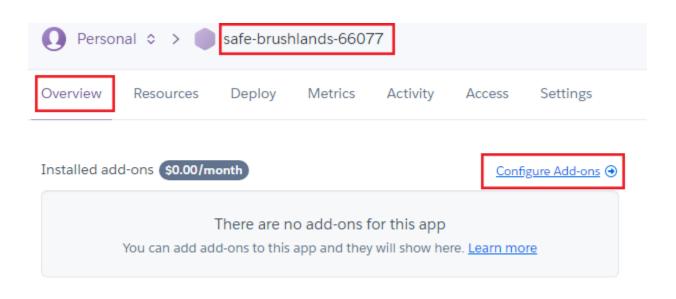
Para implementar bases de datos **MySQL** en Heroku vamos a utilizar uno de los tantos plugins que ofrece. Para agregar cualquiera de ellos a nuestro servicio será necesario tener validada una tarjeta de crédito dentro de los datos de perfil.

No debemos preocuparnos, ya que mientras nos mantengamos dentro de los límites que el plan gratis nos define no recibiremos cargo alguno en esta tarjeta.

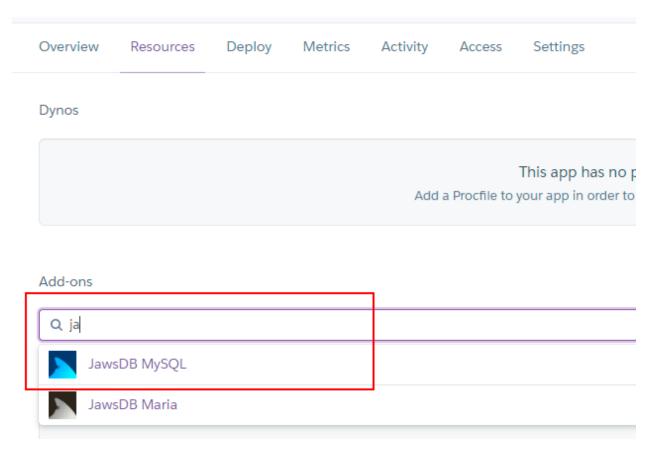
### Paso 1

Dentro del panel de administración de Heroku buscamos nuestra aplicación creada anteriormente, en este caso (safe-brushlands-66077), y hacemos click en la opción **configure Add-ons**.

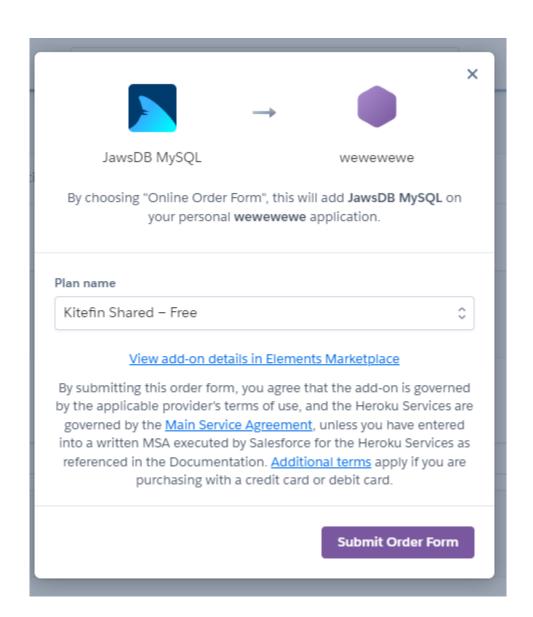




En el buscador escribimos mysql y seleccionamos el servicio de JawsBDMySQL.







### Paso 2

Vamos a configurar las variables de entorno de nuestra aplicación para que se conecte a la base de datos que acabamos de dar de alta. Para esto vamos a la opción **Settings** y clickeando en Reveal Config Vars veremos la ruta a la base de



datos, la cual deberemos descomponer en valores que igualen a los que usabamos en nuestro archivo env local.

Overview	Resources	Deploy	Metrics	Activity	Access	Settings	
App Information			App Name safe-brushlands-66077				
			Parier		10		
			Region	United		e Stack	
			Framework	No framew	ork detect	ed	
			Slug size	No slug de	tected		
			Heroku git URL	https://g	it.heroku.	com/safe-brush	lands-66077.git
Config Vars	5		Reveal Cor	nfig Vars			

El valor que nos mostrará tendrá el siguiente formato y deberemos ir extrayendo cada uno de los valores que necesitemos y asignarlo a una de las variables de entorno que nuestra aplicación espera.

mysql://fkc9354g7wqfu015:bfpdk0shbqlnz62t@lyn7gfxo996yjjco.cbe txkdyhwsb.us-east-1.rds.amazonaws.com:3306/tey46qz1q787wrgl

mysql://user:password@host/database



Config Vars				
JAWSDB_URL	mysql://v3p5qh0zfzezjvl8:l1psk2ywp1l484	gv	×	
MYSQL_DB_NAME	sexjce6xnk3byj3e	,	×	
MYSQL_HOST	l6glqt8gsx37y4hs.cbetxkdyhwsb.us-east-1	.r	×	
MYSQL_PASSWORD	l1psk2ywp1l484gv	,	×	
MYSQL_USER	v3p5qh0zfzezjvl8	/	×	

Recordar sacar el puerto 3306 tanto en la variable JAWSDB\_URL Y MYSQL\_HOST

### Paso 3

@echo off

Con el siguiente código vamos a poder migrar los datos de nuestra base de datos local a la base de datos que acabamos de agregar a Heroku.

```
heroku config | findstr JAWSDB_URL > config.txt
set /p url=<config.txt
```

```
set "string=%url:?=" & set "x=%"
set "x=%string:/=" & set "dbname=%"
echo DB name: %dbname%
echo DB name: %dbname% >> config.txt
set "x=%string:@=" & set "substring=%"
set "host=%substring:/=" & set "x=%"
```



echo Host: %host%

echo Host: %host% >> config.txt

set "x=%string::=" & set "substring=%"

set "password=%substring:@=" & set "x=%"

echo Password: %password%

echo Password: %password% >> config.txt

set "x=%string:://=" & set "substring=%"

set "user=%substring::=" & set "x=%"

echo User: %user%

echo User: %user% >> config.txt

mysql -u %user% -p%password% -h %host% -D %dbname%
--ssl-mode=DISABLED < %1</pre>

Copiamos el código y lo guardamos en un archivo con el nombre: **import.bat** al mismo nivel donde está el archivo app.js. Cabe destacar que esta opción sirve solo para Windows.

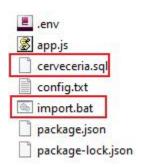


```
@echo off
heroku config | findstr JAWSDB_URL > config.txt
set /p url=<config.txt</pre>
set "string=%url:?=" & set "x=%"
set "x=%string:/=" & set "dbname=%"
echo DB name:
                %dbname%
                %dbname% >> config.txt
echo DB name:
set "x=%string:@=" & set "substring=%"
set "host=%substring:/=" & set "x=%"
echo Host:
                %host%
echo Host:
                %host% >> config.txt
set "x=%string::=" & set "substring=%"
set "password=%substring:@=" & set "x=%"
echo Password: %password%
echo Password: %password% >> config.txt
set "x=%string:://=" & set "substring=%"
set "user=%substring::=" & set "x=%"
echo User:
                %user%
                %user% >> config.txt
echo User:
mysql -u %user% -p%password% -h %host% -D %dbname% --ssl-mode=DISABLED < %1
```

### Paso 4

Exportamos la base de datos desde phpMyAdmin, y la guardamos como **cerveceria.sql** junto con el archivo **import.bat**. Finalmente en la consola escribimos el siguiente comando: **import cerveceria.sql**. Una vez terminado, todos los datos que teníamos en nuestra base de datos local habrán sido copiados al servidor MySQL remoto.









# Bibliografía utilizada y sugerida

### Artículos de revista en formato electrónico:

Heroku. Disponible desde la URL: <a href="https://www.heroku.com/">https://www.heroku.com/</a>

**npmjs.** Disponible desde la URL: <a href="https://www.npmjs.com/">https://www.npmjs.com/</a>