

REST API EN HEROKU

Jairo Rastrojo Rodríguez

2ºASIR

Administración de sistemas gestores de bases de datos

Indice

1. Rutas

2. Uso de postman

3. Subida a github

4. Vinculación con mongo atlas

5. Vinculación con heroku

1. Rutas

Para empezar, vamos a empezar explicando, que vamos a trabajar sobre el proyecto del Hospital del primer trimestre (esta vez funciona).

Con este proyecto como base, vamos a crear una serie de rutas para que al acceder a ellas usando diferentes métodos podamos ver diferentes resultados.

Los metodos que vamos a usar son:

-POST → Para crear

-GET → Para consultar

Existen más métodos, pero hemos querido hacerlo simple, para que a la hora del proyecto poder desarroyar los otros métodos que nombraremos a continuación:

PUT → Para modificar

DELETE → Para borrar

NEW → Para que devuelva la modificación que se ha echo

Y usaremos además en el proyecto del segundo trimestre mongo import y mongo export

A continuación vamos a ver las rutas que hemos creado y su utilidad

```
misRutas(){
  this._router.get('/pacientes', this.getPacientes)
  this._router.get('/empleados', this.getEmpleados)
  this._router.post('/newpaciente', this.postpaciente)
  this._router.post('/newempleado', this.postempleado)
}
```

-Primera ruta: Como podemos ver nos lista los pacientes de los que disponemos en la base de datos.

The screenshot shows a web browser at `localhost:3000/pacientes`. The left pane displays a JSON array of patient data. The right pane shows the same data in a table view within the application's interface.

```
[{"_id": 1, "_nombre": "jaime", "_apellido1": "lunar", "_apellido2": "solar", "_edad": 23, "_dni": "93804729r", "_seguro": true, "_telefono": 839662910, "_dolencia": "brecha en la cabeza", "_prueba": "curas", "_test": "", "_preciobase": 89, "_v": 0}, {"_id": 2, "_nombre": "laura", "_apellido1": "lopez", "_apellido2": "luque", "_edad": 34, "_dni": "93487298u", "_seguro": true, "_telefono": 893652954, "_dolencia": "lesion en la pierna", "_prueba": "radiografia", "_test": "", "_preciobase": 94, "_v": 0}]
```

_id	_nombre	_apellido1	_apellido2	_edad	_dni	_seguro	_telefono	_dolencia	_prueba	_test	_preciobase	_v
1	jaime	lunar	solar	23	93804729r	true	839662910	brecha en la cabeza	curas		89	0
2	laura	lopez	luque	34	93487298u	true	893652954	lesion en la pierna	radiografia		94	0

-Segunda ruta: al igual que el anterior lista en este caso los empleados de los que disponemos en la base de datos.

The screenshot shows a web browser at `localhost:3000/empleados`. The left pane displays a JSON array of employee data. The right pane shows the same data in a table view within the application's interface.

```
[{"_id": 1, "_nombrem": "jairo", "_apellido": "rastroy", "_contacto": 628552722, "_pacientes": [], "_especialidad": "oculista", "_segundoIdioma": "", "_v": 0}, {"_id": 2, "_nombrem": "silvia", "_apellido": "espinola", "_contacto": 738626277, "_pacientes": [], "_especialidad": "pediatra", "_segundoIdioma": "", "_v": 0}, {"_id": 3, "_nombrem": "jose", "_apellido": "palacios", "_contacto": 877738845, "_pacientes": [], "_especialidad": "traumatologo", "_segundoIdioma": "", "_v": 0}]
```

_id	_nombrem	_apellido	_contacto	_pacientes	_especialidad	_segundoIdioma	_v
1	jairo	rastroy	628552722	Array	oculista		0
2	silvia	espinola	738626277	Array	pediatra		0
3	jose	palacios	877738845	Array	traumatologo		0

2. Uso de postman

La tercera ruta usa el método post y debemos rellenar los campos que requiere el documento a través del body, podemos ver que no da error y que en mongo local esta el nuevo paciente

POST localhost:3000/newpaciente

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

● none ● form-data ● x-www-form-urlencoded ● **raw** ● binary ● GraphQL **JSON**

```
1 {
2   ... "_id": 5,
3   ... "_nombre": "jorge",
4   ... "_apellido1": "romano",
5   ... "_apellido2": "vasco",
6   ... "_edad": 12,
7   ... "_dni": "83748965i",
8   ... "_seguro": false,
9   ... "_telefono": "736859490",
10  ... "_dolencia": "contacto covid",
11  ... "_tipo": "covid",
12  ... "_prueba": "",
13  ... "_test": "pcr",
14  ... "_preciobase": 88
15 }
```

```
> {
  _id: 5
  _nombre: "jorge"
  _apellido1: "romano"
  _apellido2: "vasco"
  _edad: 12
  _dni: "83748965i"
  _seguro: false
  _telefono: 736859490
  _dolencia: "contacto covid"
  _tipo: "covid"
  _prueba: ""
  _test: "pcr"
  _preciobase: 88
  __v: 0
}
```

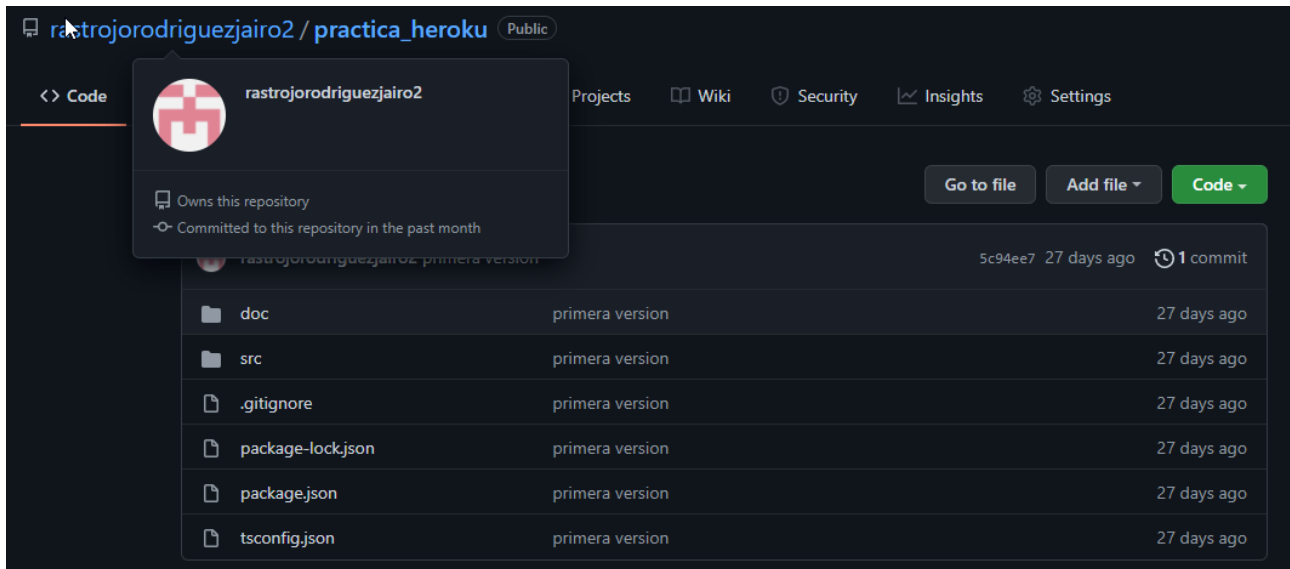
Body Cookies Headers (11) Test Results

Pretty Raw Preview Visualize HTML

```
1 El documento se ha introducido correctamente en la base de datos {
2   _id: 5,
3   _nombre: 'jorge',
4   _apellido1: 'romano',
5   _apellido2: 'vasco',
6   _edad: 12,
```

3. Subida a github




Lo hemos subido al github: https://github.com/rastrojorodriguezjairo2/practica_heroku



4. Vinculación con mongo atlas

Primero, debemos poner la ruta de mongo atlas en nuestro archivo database.ts con un usuario y su contraseña.

```
database.ts x TS index.ts TS server.ts {} package.json 1 TS pacientes.ts TS empleados.ts
src > database > TS database.ts > DataBase > _cadenaConexion2
1 import mongoose from 'mongoose';
2
3 class DataBase {
4
5   public _cadenaConexion2: string = 'mongodb://localhost/Hospital'
6   public _cadenaConexion: string = 'mongodb+srv://jairo:1234@cluster0.dynye.mongodb.net/myFirstDatabase?retryWrites=true&w=majority'
7   constructor(){
```

-
- **Connect with the MongoDB Shell**
Interact with your cluster using MongoDB's interactive Javascript interface >
-
- **Connect your application**
Connect your application to your cluster using MongoDB's native drivers >
-
- **Connect using MongoDB Compass**
Explore, modify, and visualize your data with MongoDB's GUI >
-

Y tenemos las colecciones en atlas

Cluster0

Overview

Real Time

DATABASES: 1 COLLECTIONS: 2

+ Create Database

Q NAMESPACES

▼ myFirstDatabase

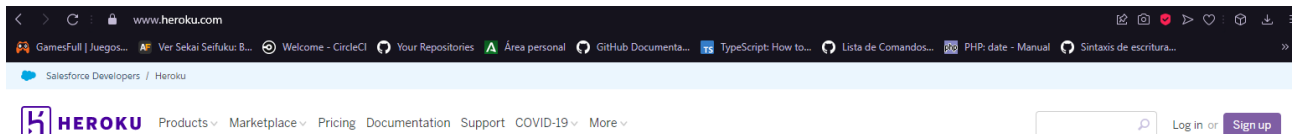
empleados

pacientes

5. Vinculación con heroku

Por último vamos a vincular heroku con github y vamos a subirlo.

Primero accedemos a heroku



Creamos una cuenta y a continuación creamos una nueva app

A screenshot showing the Heroku account creation and app creation process. On the left, there's a sidebar with 'Free account', 'Your app platform', and 'Deploy now' sections. The main form on the right includes fields for 'First name' (jairo), 'Last name' (rastroy), 'Email address' (jrasrod934@educand.es), 'Company name', 'Role' (Student), 'Country' (Spain), and 'Primary development language' (Node.js). A 'CREATE FREE ACCOUNT' button is at the bottom. A modal window is open over the form, showing a 'New' dropdown menu with options 'Create new app' and 'Create new pipeline'. A hand cursor is pointing at 'Create new app'.

En este caso hospita-56 será el nombre que hemos elegido para esta practica

A screenshot of the Heroku app creation form. The 'App name' field contains 'hospital-56' and has a green checkmark icon to its right. Below it, a message states 'hospital-56 is available'. The 'Choose a region' dropdown menu is set to 'Europe'. The 'Add this app to a pipeline' section includes a link to 'Learn more' and a 'Choose a pipeline' dropdown menu. At the bottom, there is a purple 'Create app' button.

Seleccionamos github para conectarnos



Heroku Git
Use Heroku CLI



GitHub
Connect to GitHub



Container Registry
Use Heroku CLI

View your code diffs on GitHub

Connect your app to a GitHub repository to see commit diffs in the activity log.

Deploy changes with GitHub

Connecting to a repository will allow you to deploy a branch to your app.

Automatic deploys from GitHub

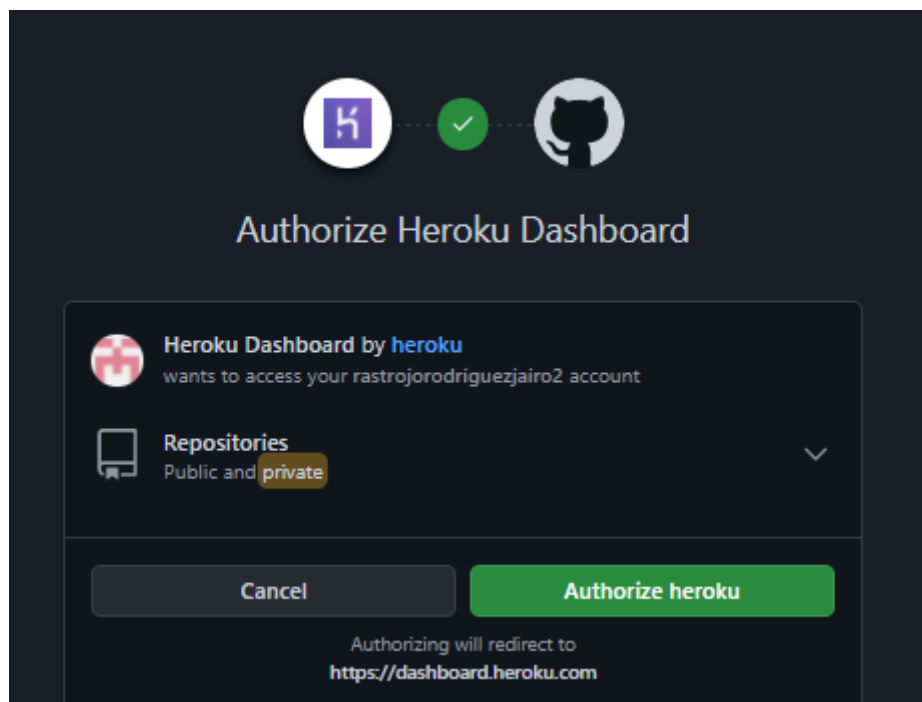
Select a branch to deploy automatically whenever it is pushed to.

Create review apps in pipelines


Pipelines connected to GitHub can enable **review apps**, and create apps for new pull requests. [Learn more](#).


Connect to GitHub


Nos logeamos con github para autorizar la vinculación



Elegimos el repositorio de practica-heroku para vincularlo con nuestro heroku

 Heroku Git
Use Heroku CLI

 GitHub
Connect to GitHub


 Container Registry
Use Heroku CLI


 rastrojorodriguezjairo2


repo-name

Search

Se confirma la vinculación,

 Heroku Git
Use Heroku CLI

 GitHub
Connected

 Container Registry
Use Heroku CLI

Connected to  [rastrojorodriguezjairo2/practica_heroku](#) by  [rastrojorodriguezjairo2](#)

Disconnect...

 Releases in the [activity feed](#) link to GitHub to view commit diffs

Confirmamos pulsando en “Deploy Branch”

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more.](#)

Choose a branch to deploy

 main 

Deploy Branch

Receive code from GitHub



Build main 5c94ee7e



Release phase



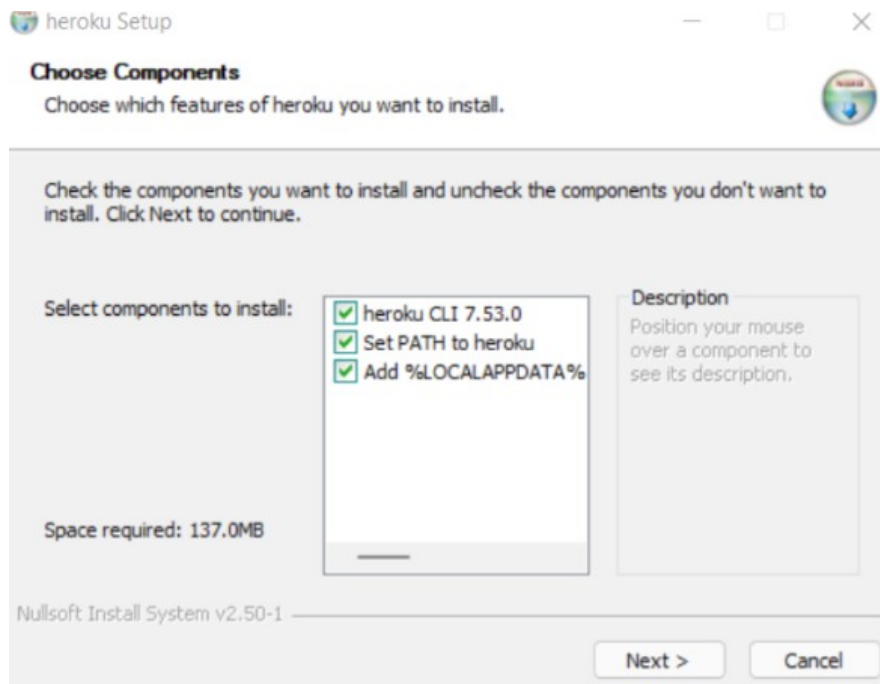
Deploy to Heroku



Your app was successfully deployed.

 View

Por último vamos a instalar heroku cli, es decir el cliente local de heroku, para poder trabajar de forma local



Para comprobar que se ha instalado vamos a consultar la ayuda del comando heroku

```
C:\WINDOWS\system32>heroku --help
» Warning: Our terms of service have changed: https://dashboard.heroku.com/terms-of-service
CLI to interact with Heroku

VERSION
heroku/7.53.0 win32-x64 node-v12.21.0

USAGE
$ heroku [COMMAND]

COMMANDS
access          manage user access to apps
addons          tools and services for developing, extending, and operating your app
apps            manage apps on Heroku
auth            check 2fa status
authorizations  OAuth authorizations
autocomplete    display autocomplete installation instructions
buildpacks      scripts used to compile apps
certs           a topic for the ssl plugin
ci              run an application test suite on Heroku
clients         OAuth clients on the platform
config          environment variables of apps
container       Use containers to build and deploy Heroku apps
domains         custom domains for apps
drains          forward logs to syslog or HTTPS
features        add/remove app features
git             manage local git repository for app
help            display help for heroku
keys            add/remove account ssh keys
labs            add/remove experimental features
add/remove experimental features
```