

# Hola

**Ángel Gómez**

[github.com/harvestcore](https://github.com/harvestcore)

[github.com/harvestcore/IPContainer](https://github.com/harvestcore/IPContainer)



The background is a solid red color with abstract geometric patterns of triangles in various shades of red, orange, and yellow. Some triangles are solid, while others are outlined in white. The patterns are scattered across the image, with a denser cluster on the right side.

# IPContainer



## ¿Qué es?

- ▶ IPContainer es un microservicio centrado en el almacenamiento y gestión básica de direcciones IP. Permite tener almacenadas una serie de direcciones IP en un mismo lugar, agrupadas por tipo de red:

- ▶ PAN
- ▶ LAN
- ▶ WAN
- ▶ SAN
- ▶ VLAN
- ▶ WLAN
- ▶ DNS



## ¿Por qué?

- ▶ La idea surge tras necesitar un lugar donde tener una serie de direcciones IP organizadas y almacenadas en un mismo lugar, sin depender de mirar archivos de configuración o uso de otras aplicaciones.



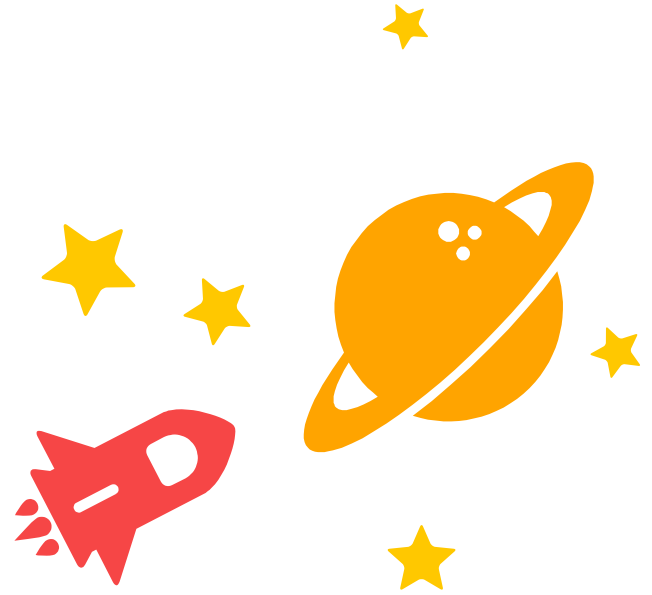
# Herramientas y servicios


- ◀ Python
  - ◀ Flask
  - ◀ SQLAlchemy
- ◀ MySQL (Google Cloud)
- ◀ TravisCI
- ◀ Heroku (PaaS)
- ◀ Docker
- ◀ Google Cloud (IaaS)

# Desarrollo



# Base de datos



- 
- ◀ Utilizo tres:
    - ◀ Tests
    - ◀ Heroku
    - ◀ Google Cloud



# Usuarios

Nombre	Tipo	Tamaño	Otros
id	<b>int</b>	<b>7</b>	<b>CP. AI.</b>
username	<b>varchar</b>	<b>35</b>	<b>Único. Not null.</b>

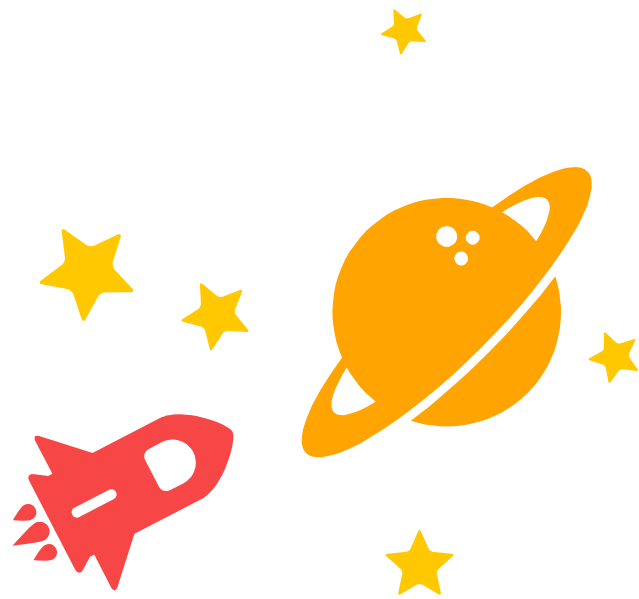
# Redes

Nombre	Tipo	Tamaño	Otros
id	<b>int</b>	<b>7</b>	<b>CP. AI.</b>
username	<b>varchar</b>	<b>35</b>	<b>Not null.</b>
type	<b>varchar</b>	<b>4</b>	<b>Not null.</b>
data	<b>json</b>	<b>-</b>	<b>-</b>

# Usuarios API

Nombre	Tipo	Tamaño	Otros
id	<b>int</b>	<b>7</b>	<b>CP. AI.</b>
public_id	<b>varchar</b>	<b>50</b>	<b>Not null.</b>
name	<b>Varchar</b>	<b>35</b>	<b>Not null.</b>
password	<b>varchar</b>	<b>100</b>	<b>Not null.</b>
admin	<b>tinyint</b>	<b>1</b>	<b>-</b>

# API





## General

- ◀ /
- ◀ /status
- ◀ /status/:user

## Usuarios

- ◀ /User
- ◀ /NOUsers
- ◀ /existsUser
- ◀ /dropUsers

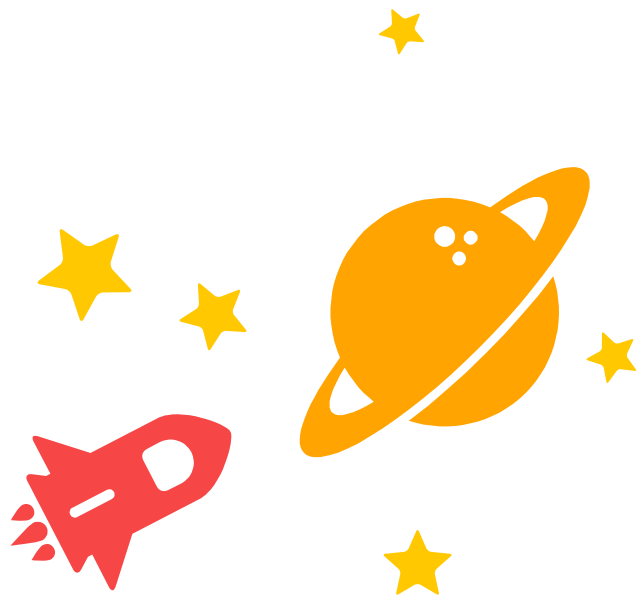
## Redes


- ◀ /Network/:user/:type
- ◀ /IPNetwork/:user/:type
- ◀ /NONetworks
- ◀ /existsNetwork/:user/:type
- ◀ /SzNetwork/:user/:type
- ◀ /Data/:user/:type
- ◀ /dropData

## Usuarios API

- ◀ /APIUser
- ◀ /APIUser/:public\_id
- ◀ /login

# Token Auth



- 
- ▶ Se genera un UUID único para cada usuario.
  - ▶ La contraseña se cifra con SHA256.
  - ▶ El token de acceso se genera a partir del UUID y de una clave secreta.
  - ▶ Expira a los 20 minutos.
  - ▶ El token se añade en las cabeceras del resto de peticiones. (x-access-token)

"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJwdWJsaWNfaWQiOiI4NTc0OTJlYiIjOWI5LTQ5NDMtOTJiZS1lZGEzODEyYjg0MDYiLCJleHAiOiJlNDc0NzI3NDI9.eSLxPbHuRN8XlRekOffKXR2FLUlur\_B9R4Df4k\_0zxo"

# Tests





# TravisCI

## Unittest

Los tests se pasan tras cada push del repo en GitHub.

Necesario *.travis.yml*:

```
language: python
python:
  - "3.6"

# Instalar dependencias
install:
  - pip3 install -r requirements.txt

# Ejecutar script test
script:
  - pytest
```

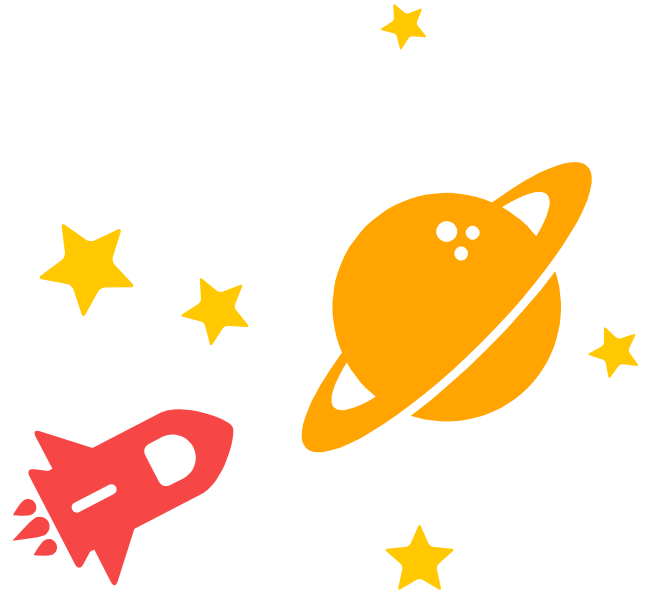
Dos tipos de tests:


- ◀ Clase
- ◀ Despliegues

<https://github.com/harvestcore/IPContainer/blob/master/docs/test.md>

<https://travis-ci.com/harvestcore/IPContainer>

# Heroku





Despliegue tras cada push del repo en GitHub (y se han pasado los tests correctamente).

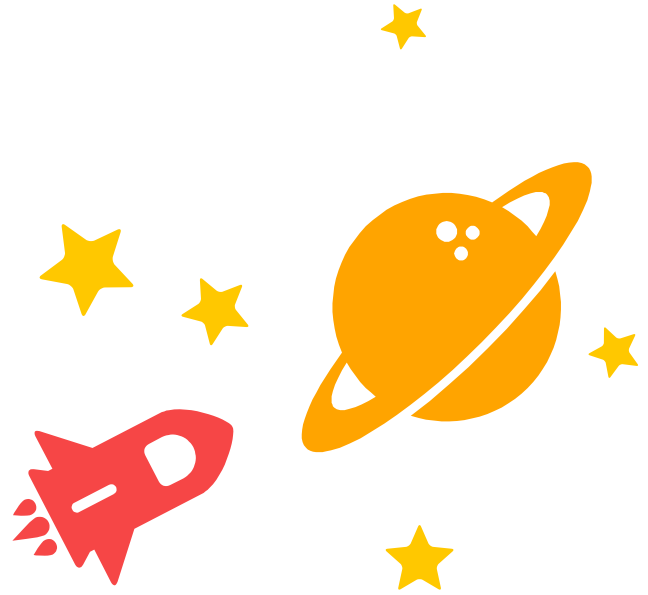
Necesario Procfile:

```
web: gunicorn app:app
```

<https://github.com/harvestcore/IPContainer/blob/master/docs/heroku.md>

<https://ipcontainer.herokuapp.com>

# Docker





La imagen se construye tras cada push del repo en GitHub.

## Necesario **Dockerfile**:

```
FROM python:3.6-slim-stretch

# Copio los archivos necesarios en el directorio /ipc.
COPY . ./ipc

# Actualizo pip e instalo las dependencias necesarias.
RUN pip install --upgrade pip
RUN cd ./ipc && pip3 install -r requirements.txt

# Expongo el puerto 5000
EXPOSE 5000

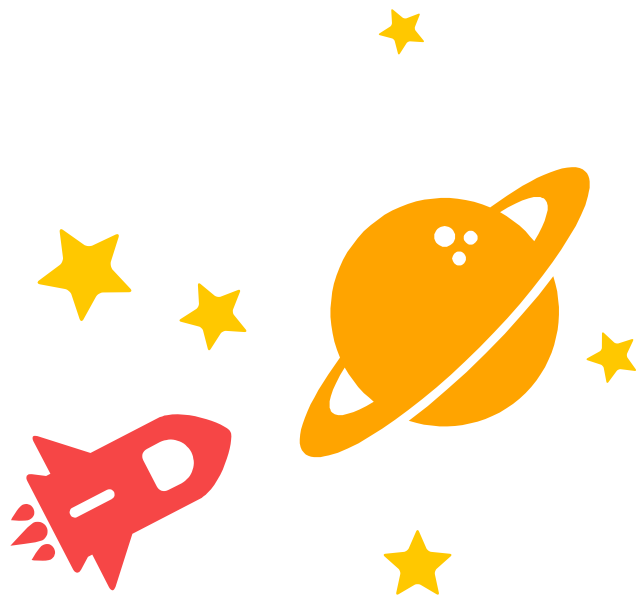
# Ejecuto el microservicio
CMD cd ./ipc && python3 application.py
```


Importante usar *.dockerignore*.

<https://github.com/harvestcore/IPContainer/blob/master/docs/docker.md>

<https://hub.docker.com/r/harvestcore/ipcontainer>

# Heroku + Docker





Despliegue tras cada push del repo en GitHub (y se han pasado los tests correctamente).

Uso del cliente de heroku.

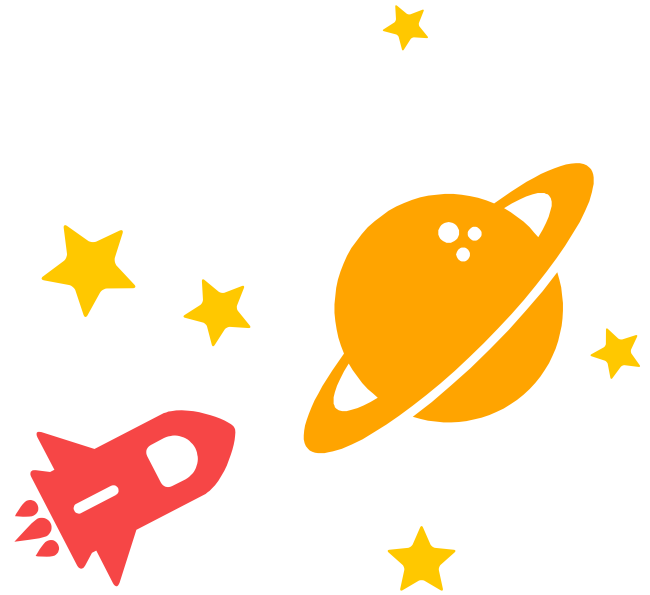
Necesario *heroku.yml*:

```
build:
  docker:
    web: Dockerfile
run:
  web: cd ipc && python3 application.py
```

<https://github.com/harvestcore/IPContainer/blob/master/docs/docker.md>

<https://ipcontainer-docker.herokuapp.com>

# Google Cloud







# Google Cloud

Más fácil de utilizar (a mi parecer) que Microsoft Azure.  
Facturación.

- ▶ No me quedaba saldo en Azure. Lo que me ha servido para conocer GC.

**Sistema operativo:** Ubuntu 16.04 LTS

Uso **Docker**.

- ▶ Había que darle uso al trabajo anterior.
- ▶ Sencillo de desplegar.
- ▶ IPContainer se encuentra más aislado y seguro.

<https://github.com/harvestcore/IPContainer/blob/master/docs/desplieguefinal.md>

<http://35.246.87.7>

# Vagrant

Herramienta de orquestamiento.

Necesario *Vagrantfile*.

- ▶ Manera sencilla de crear máquinas virtuales + provisionamiento.
- ▶ Posible gracias al plugin de Google Cloud para Vagrant.
- ▶ **Tipo máquina:** *g1-small*
- ▶ **Zona:** *europe-west2-a*

<https://github.com/harvestcore/IPContainer/blob/master/docs/vagrant.md>

```
Vagrant.configure("2") do | config |

  # Defino máquina "ipcontainer"
  config.vm.define "ipcontainer" do | ipcontainer |

    # Máquina base.
    config.vm.box = "google/gce"

    # Versión de la máquina.
    config.vm.box_version = "0.1.0"

    # Evito que Vagrant busque actualizaciones de la máquina base.
    config.vm.box_check_update = false

    # Configuración de la máquina.
    ipcontainer.vm.provider "google" do | gcloud, override |

      # Configuración de las credenciales de Google Cloud.
      gcloud.google_project_id = ENV['PROJECT_ID']
      gcloud.google_client_email = ENV['CLIENT_EMAIL']
      gcloud.google_json_key_location = ENV['JSON_KEY_LOCATION']

      # Configuración básica de la VM
      gcloud.image_family = 'ubuntu-1604-lts'
      gcloud.zone = 'europe-west2-a'
      gcloud.name = 'ipcontainer'
      gcloud.machine_type = 'g1-small'

      # Configuración usuario y clave privada SSH.
      override.ssh.username = "aagomezies"
      override.ssh.private_key_path = '~/ssh/id_rsa'
    end

    # Provisionamiento con Ansible.
    config.vm.provision "ansible" do | ans |
      ans.playbook = "provision/playbook.yml"
    end
  end
end
```

# Ansible

Herramienta de provisionamiento.

Manera sencilla de crear provisionar VMs.

▶ Necesario *Playbook*.

▶ Necesario *hosts*.

```
[staging]
192.168.56.105
```

```
[ipcontainer]
35.246.87.7
```

<https://github.com/harvestcore/IPContainer/blob/master/docs/provision.md>

```
---
- hosts: ipcontainer
  remote_user: aagomezies
  tasks:
    - name: Agregar repo python 3.6
      become: true
      apt_repository: repo=ppa:deadsnakes/ppa state=present

    - name: Update apt
      become: true
      apt:
        upgrade: yes
        update_cache: yes

    - name: Instalar docker
      become: true
      apt: pkg=docker.io state=present

    - name: Cambio permisos docker
      become: true
      file: path=/var/run/docker.sock owner=aagomezies group=docker

    - name: Copio env.list
      copy: src=./env.list dest=/home/aagomezies owner=aagomezies group=aagomezies

    - name: Instalar Python 3.6
      become: true
      apt: pkg=python3.6 state=present

    - name: Instalar pip3
      become: true
      apt: pkg=python3-pip state=latest

    - name: Instalar pip
      become: true
      apt: pkg=python-pip state=latest

    - name: Instalar docker-py
      become: true
      pip: name=docker-py state=latest

    - name: Instalar setuptools
      become: true
      pip: name=setuptools state=latest

    - name: Ejecutar servicio Docker
      become: true
      service: name=docker state=started

    - name: Descargar docker IPC
      docker_image: name=harvestcore/ipcontainer state=present
```

# Fabric

Herramienta para el despliegue.

Manera sencilla de crear provisionar VMs.

- ◀ Necesario *Fabfile.py*.
- ◀ Hosts definidos en la configuración de SSH.

```
ServerAliveInterval 30

# Máquina de staging
Host ubuntu
    HostName 192.168.56.105
    User ubuntu
    IdentityFile ~/.ssh/id_rsa

# Máquina de producción
Host ipcontainer
    HostName 35.246.87.7
    User aagomezies
    IdentityFile ~/.ssh/id_rsa
```

```
from fabric.api import *

env.use_ssh_config = True

def staging():
    env.hosts = ['ubuntu']

def production():
    env.hosts = ['ipcontainer']

def app_up():
    run('docker run -d -p 80:5000 --env-file ~/env.list --name ipc harvestcore/ipcontainer')

def app_down():
    run('docker stop ipc')

def dockersock():
    run('sudo chown aagomezies:docker /var/run/docker.sock')

def dockerprune():
    run('docker system prune -f')

def dockerimages():
    run('docker images')

def dockerps():
    run('docker ps')

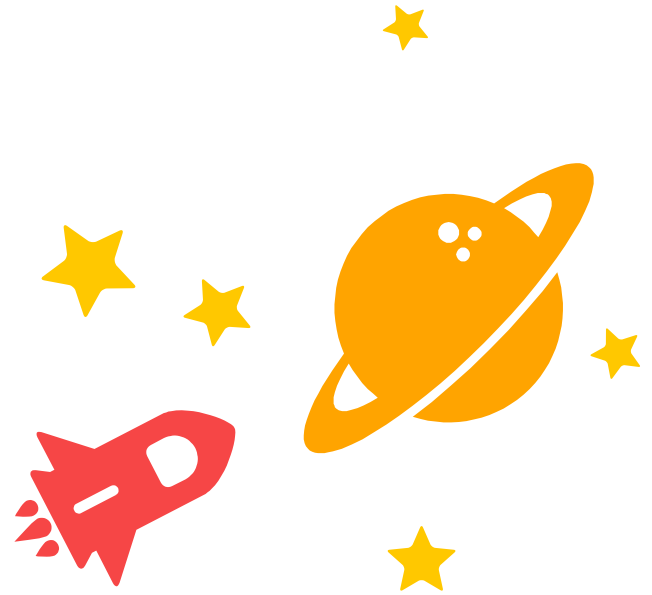
def update_app():
    run('docker pull harvestcore/ipcontainer')

def ipc_up():
    execute(dockersock)
    execute(dockerprune)
    execute(app_up)

def ipc_down():
    execute(app_down)
```

<https://github.com/harvestcore/IPContainer/blob/master/docs/despliegue.md>

# Otros despliegues





## Zeit

- ◀ Necesario *now.json*
- ◀ Se construye la imagen tras cada push al repo.

## Microsoft Azure

- ◀ Se despliega tras cada push al repo.
- ◀ Similar a Heroku (PaaS).
- ◀ RIP crédito.

# Ejemplos



GET ▼ http://35.246.87.7/status

Params Authorization Headers Body Pre-reqs

KEY

Key

Body Cookies Headers (4) Test Results

Pretty Raw Preview JSON ≡

```
1 {
2   "main": {
3     "networks": {
4       "dns": 0,
5       "lan": 0,
6       "pan": 0,
7       "san": 0,
8       "vlan": 0,
9       "wan": 0,
10      "wlan": 0
11    },
12    "noofnetworks": 0,
13    "noofusers": 0
14  },
15  "routes": {
16    "status": "OK"
17  }
18 }
```





GET ▼ http://35.246.87.7/NOUsers

Params Authorization Headers Body Pre-reqs

	KEY
	Key

Body Cookies Headers (4) Test Results

Pretty Raw Preview JSON ↺

```
1 {  
2   "message": "Token is missing."  
3 }
```





GET

http://35.246.87.7/NOUsers

Send

Save

Params

Authorization

Headers (1)

Body

Pre-request Script

Tests

Cookies

Code

Comments (0)

	KEY	VALUE	DESCRIPTION	...	Bulk Edit	Presets
<input checked="" type="checkbox"/>	x-access-token	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJwdWJsa...				
	Key	Value	Description			

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 122 ms

Size: 158 B

Download

Pretty

Raw

Preview

JSON

1

2

3

{  
 "users": 0  
}



http://35.246.87.7/User/angel

POST

http://35.246.87.7/User/angel

Send

Save

Params

Authorization

Headers (1)

Body

Pre-request Script

Tests

Cookies Code Comments (0)

KEY

VALUE

DESCRIPTION

...

Bulk Edit

Presets



x-access-token

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJwdWJsa...

Key

Value

Description

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 167 ms

Size: 163 B

Download

Pretty

Raw

Preview

JSON



```
1 {  
2   "success": true  
3 }
```

POST

http://35.246.87.7/Network/angel/wlan

Send

Save

Params

Authorization

Headers (1)

Body

Pre-request Script

Tests

Cookies

Code

Comments (0)

	KEY	VALUE	DESCRIPTION	...	Bulk Edit	Presets
<input checked="" type="checkbox"/>	x-access-token	eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJwdWJsa...				
	Key	Value	Description			

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 281 ms

Size: 163 B

Download

Pretty

Raw

Preview

JSON

```
1 {
2   "success": true
3 }
```



POST ▼ http://35.246.87.7/IPNetwork/angel/wlan Send Save ▼

Params Authorization Headers (2) **Body** Pre-request Script Tests

Cookies Code Comments (0)

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary JSON (application/json) ▼

Beautify

```
1 {  
2   "data": {  
3     "ip": "192.168.1.150",  
4     "mac": "00:A0:C9:14:C8:29",  
5     "gateway": "192.168.1.255",  
6     "network": "192.168.1.0",  
7     "netmask": "255.255.255.0",  
8     "broadcast": "192.168.1.1",  
9     "tipo": "PC",  
10    "nombre": "ANGELPC",  
11    "dispositivo": "PC de Angel"  
12  }  
13 }
```

Body Cookies Headers (4) Test Results

Status: 200 OK Time: 225 ms Size: 163 B

Download

Pretty Raw Preview

JSON ▼



```
1 {  
2   "success": true  
3 }
```

POST ▼ http://35.246.87.7/IPNetwork/angel/wlan Send ▼ Save ▼


Params Authorization Headers (2) **Body** ● Pre-request Script Tests Cookies Code Comments (0)

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary JSON (application/json) ▼ Beautify

```
1 {  
2   "data": {  
3     "ip": "192.168.1.200",  
4     "mac": "00:A0:C9:14:C8:29",  
5     "gateway": "192.168.1.255",  
6     "network": "192.168.1.0",  
7     "netmask": "255.255.255.0",  
8     "broadcast": "192.168.1.1",  
9     "tipo": "PC",  
10    "nombre": "ANGELPC",  
11    "dispositivo": "PC de Angel"  
12  }  
13 }
```

**Body** Cookies Headers (4) Test Results

Status: 200 OK Time: 205 ms Size: 163 B Download

Pretty Raw Preview JSON ▼ 

```
1 {  
2   "success": true  
3 }
```



GET

http://35.246.87.7/Data/angel/wlan

Send

Save

Params

Authorization

Headers (2)

Body

Pre-request Script

Tests

Cookies

Code

Comments (0)

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 185 ms

Size: 619 B

Download

Pretty

Raw

Preview

JSON

```
1 {
2   "data": {
3     "data": [
4       {
5         "broadcast": "192.168.1.1",
6         "dispositivo": "PC de Angel",
7         "gateway": "192.168.1.255",
8         "ip": "192.168.1.150",
9         "mac": "00:A0:C9:14:C8:29",
10        "netmask": "255.255.255.0",
11        "network": "192.168.1.0",
12        "nombre": "ANGELPC",
13        "tipo": "PC"
14      },
15      {
16        "broadcast": "192.168.1.1",
17        "dispositivo": "PC de Angel",
18        "gateway": "192.168.1.255",
19        "ip": "192.168.1.200",
20        "mac": "00:A0:C9:14:C8:29",
21        "netmask": "255.255.255.0",
22        "network": "192.168.1.0",
23        "nombre": "ANGELPC",
24        "tipo": "PC"
25      }
26    ],
27    "network": "wlan",
28    "user": "angel"
29  }
30 }
```





GET

http://35.246.87.7/APIUser

Send

Save

Params

Authorization

Headers (2)

Body

Pre-request Script

Tests

Cookies

Code

Comments (0)

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 119 ms

Size: 507 B

Download

Pretty

Raw

Preview

JSON

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

```
{
  "apiusers": [
    {
      "password": "sha256$imEr41xM$b4801b7d1613e9e9b5598152044686b2d91099e2f4c3a2657e5622c156b62ff8",
      "public_id": "8099bfa6-870d-4a38-ab38-36dc9ca01e61",
      "username": "admin"
    },
    {
      "password": "sha256$XJJ6DVGh$e5dd15336f9090a97a5a79e36d5f892b2f644a4943680455438bdf09d6ed3658",
      "public_id": "857492eb-c9b9-4943-92be-eda3812b8406",
      "username": "test"
    }
  ],
  "noofusers": 2
}
```









# ¡Gracias!

¿ Alguna pregunta?

- [github.com/harvestcore](https://github.com/harvestcore)
- @harvestcore