## Install Flask

Pip install flask

## First Flask App

Create a file ‘app.py’ as follows:

from flask import Flask

app = Flask(\_\_name\_\_) #this is a special python variable that will give each file a unique name a unique name

then tell our app what requests it is going to serve

@app.route(‘/’) ( #http://www.google.com/ : the home page, /maps : the maps endpoint )

Since decorators always act on methods, so we’ll add a method

def home():

return ‘hello world’

# when we access the / endpoint, we should see ‘hello world’

# then tell the app to run at a specific port:

app.run(port=5000)

Une image contenant texte, capture d’écran, écran, noir

Description générée automatiquement

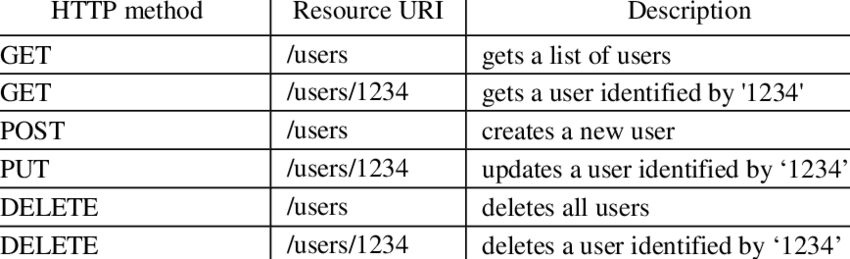
Une image contenant texte

Description générée automatiquement

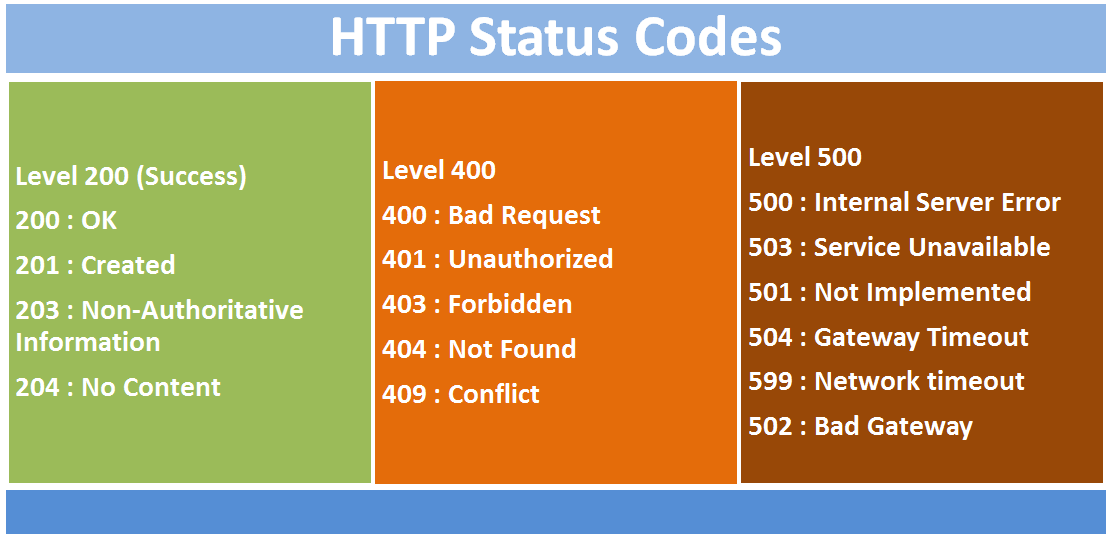
Une image contenant texte

Description générée automatiquement

## HTTP Verbs



## HTTP status codes

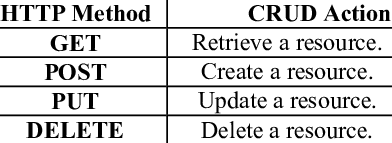


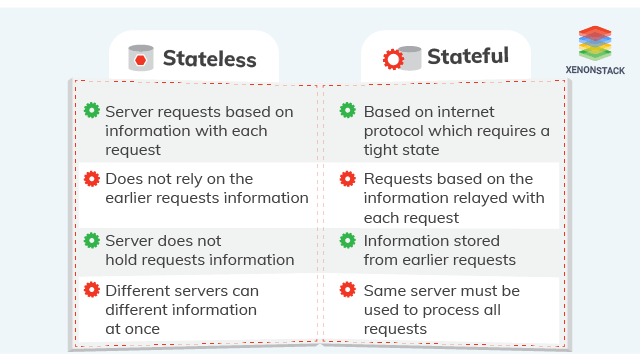
## REST Principals

REST is a way of thinking how a server will respond to a request

It doesn’t respond with just data, it responds with resources.

A resource





## Creating endpoints

Une image contenant texte

Description générée automatiquement

Now that we have our endpoints , we have to implement them

First We create a list of stores

Une image contenant texte, écran, capture d’écran

Description générée automatiquement

## Returning a list of stores

Flask’s jsonify method takes a dictionary and converts it to json

We have to import it

Une image contenant texte

Description générée automatiquement

But our stores is a list, and json is a dictionary so we turn it into a dictionary

Une image contenant texte

Description générée automatiquement

Then go to the browser and make sure to add/stores

SO a rest API will return some json after doing some processing

Notice that json always uses double quotes and never single quotes

## Implementing other endpoints

To create a new store

We import request



Une image contenant texte

Description générée automatiquement

To retrieve a specific store

Une image contenant texte

Description générée automatiquement

To retrieve all stores

Une image contenant texte

Description générée automatiquement

To create a new item

Une image contenant texte, téléphone, moniteur, téléphone mobile

Description générée automatiquement

To retrieve items in a store

Une image contenant texte

Description générée automatiquement

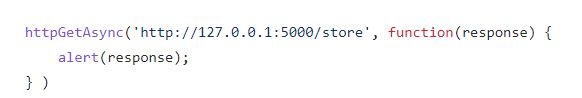
## Calling the API from javascript



Une image contenant texte

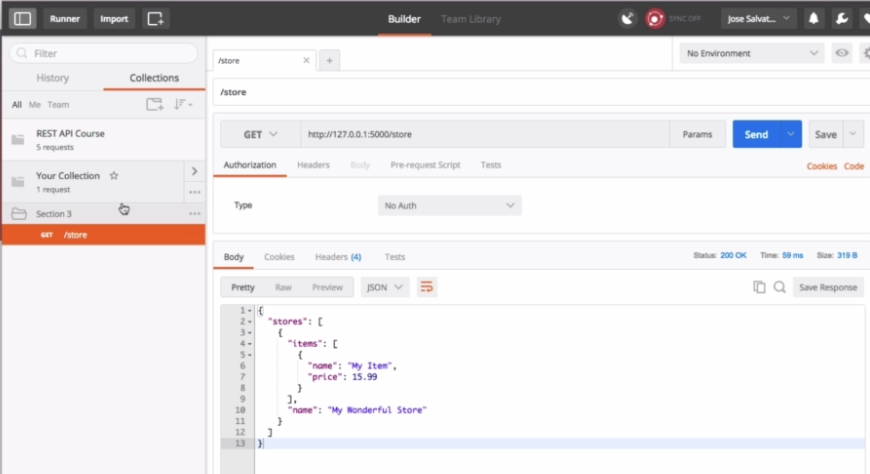
Description générée automatiquement

Then add this to the http template to run the javascript function



This just shows the API works, real tests will be done with Postman

## Testing the API with Postman



We test our three get endpoints , which is easy, the more interesting is to test the post endpoints

Set content type to json and body to raw

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Post new item

Une image contenant texte

Description générée automatiquement

Errors can be checked in the terminal

And when we correct them we restart the server, which returns it to the initial state, that’s why we need a database to persist the data