Creación de Apis con Mongo, Python, Html5 y Javascript (jquery)

Autor: Jesús Alfonso Contreras Rdz.

Contenido

Definiciones

Ajax:

Javascript y XML Asincrono, Técnica de desarrollo web que permite establecer comunicación asíncrona y en segundo plano con servicios externos al sitio web, permitiendo realizar cambios en la estructura y diseño de la página sin necesidad de recargar la misma, utilizando Javascript como código para manejo de eventos y Xml como formato de recepción y envío de datos.

API (Application Programming Interface):

Es un conjunto de funciones y procedimientos que cumplen una o muchas funciones con el fin de ser utilizadas por otro software.

REST:

Representational State Transfer, Estilo de arquitectura de software que utiliza el protocolo HTTP para establecer comunicación con componentes web. Permite el envio y recepción de contenido a través del protocolo mencionado anteriormente, sobre procedimientos bien definidos (GET, POST, PUT, DELETE) para establecimiento de comunicación Cliente/Servidor.

MongoDB:

Es una base de datos orientada a documentos. Esto quiere decir que en lugar de guardar los datos en registros, guarda los datos en documentos. Estos documentos son almacenados en BSON, que es una representación binaria de JSON.

Flask:

Es un “micro” Framework escrito en Python y concebido para facilitar el desarrollo de Aplicaciones Web bajo el patrón MVC.

Backend:

Está enfocado en hacer que todo lo que está detrás de un sitio web funcione correctamente. Toma los datos, los procesa y los envía al usuario, además de encargarse de las consultas o peticiones a la Base de Datos, la conexión con el servidor, entre otras tareas que debe realizar en su día a día. Cuenta con una serie de lenguajes y herramientas que le ayudan a cumplir con su trabajo como PHP, Ruby, Python, JavaScript, SQL, MongoDb, MySQL, etc, estos son usados para crear sitios dinámicos.

Frontend:

Se enfoca en el usuario, en todo con lo que podemos interactuar y lo que vemos mientras navegamos. Así como en una primera cita, nuestra web busca causar una buena impresión y agradar al usuario, para lo cual utiliza HTML, CSS y JAVASCRIPT.

Instalación del ambiente

Python:

1.- Instalar el programa de "python-2.7.15.amd64"

2.- Una vez instalado nos vamos a propiedades del sistema/ configuración avanzada del sistema/ Variables de entorno y en variables del sistema/PATH agregamos una nueva que es "C:\Python27"

3.- Abrimos un cmd y nos vamos a "cd/python27/scripts"

4.- Instalamos con el comando "pip install ´1,2,3,4,5,6,7,8´ "

1. Flask
2. Flask-WTF
3. Flask-RESTful
4. pymongo
5. flask_cors
6. requests
7. pyjwt
8. reportlab

5.- Ya después creamos nuestros archivos y desde un cmd nuevo nada más nos dirijimos desde el cmd a la carpeta en donde se encuentre nuestro archivo y ejecutamos el comando python "nombre_del_archivo.py"

MongoDB:

1. Creamos en el disco local C una carpeta llamada “data” y adentro de esta carpeta creamos otra que se llame “db”.

2. Después iniciamos el instalador de mongo.

3. Una vez instalado nos vamos a propiedades del sistema/ configuración avanzada del sistema/ Variables de entorno y en variables del sistema/PATH agregamos una nueva que es " "C:\Program Files\MongoDB\Server\3.6\bin".

4. Una vez agregada la variable nos pasamos a iniciar el mongo.

5. "mongod" para poder iniciar mongo

6. "mongo" para poder ver nuestras bases de datos o insertar modificar, etc...

Programación

Para poder realizar un api primero tenemos que crear nuestros documentos con extensión .py


Directions.py


Functions.py

Directions.py hace el llamado al documento de Functions.py en pocas palabras aquí es donde nosotros crearemos nuestras apis.

directions = ligamiento a las funciones de las apis.

functions = son las funciones que nuestras apis realizaran.

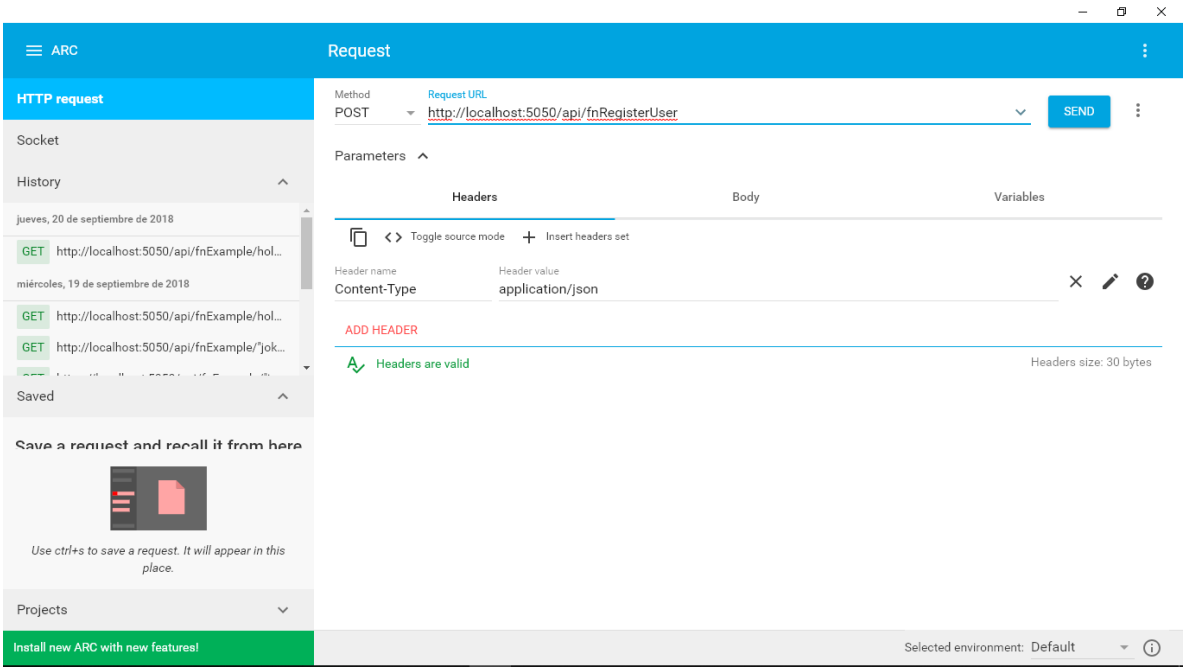
Directions ejemplo:

```
##########################################################################
####
# Api para registrar un nuevo usuario ##############################
##########################################################################
####
@app.route('/api/registerUser', methods=['POST'])
def postRegisterUser():
    try:
        strName = request.json["Name"]
        strEmail = request.json["Email"]
        doubWeight = request.json["Weight"]
        doubHeight = request.json["Height"]
        doubTotal = request.json["Total"]
        strPassword = request.json["strPassword"]

        bolResult =
callMethod.fnRegisterUser(strName,strEmail,doubWeight,doubHeight,doubTotal,strPassword)
        return bolResult
    except Exception as e:
        #print e
        respuesta = {'intResp':'0'}
        return jsonify(respuesta)
```

Functions ejemplo:

```
#########################################################################
######
#      Inserta un nuevo usuario Api tipo POST                  #
#########################################################################
######
def fnRegisterUser(strName,strEmail,doubWeight,doubHeight,doubTotal,strPassword):
   try:

      db = connectDB()
      strEmail = strEmail.lower()
      strSearch = db.clUser.find({'strEmail':strEmail})
      if strSearch.count() != 0:
         i = {
            'intResp':'100',
            'strMessage':'This user was already registered'
         }
         return jsonify(i)

      db.clUser.insert({'strName':strName,'strEmail':strEmail,'strPassword':strPassword,
'doubHeight':float(doubHeight),
'arrIMC':[{'dteIMC':datetime.datetime.now(),'doubWeight':float(doubWeight),
'doubTotal':float(doubTotal)}]})
      i = {
         'intResp':'200',
         'strMessage':'Usuer '+strName+' registered correctly'
      }
      return jsonify(i)

   except excepcion as e:
                     i = {
         'intResp':'500',
         'strMessage':'had an error'
      }

      return i
```

Aquí nosotros registraremos un nuevo usuario en la base de datos para ver si nuestra función en realidad funciona nos descargamos de google una extensión llamada “Advanced REST client”.

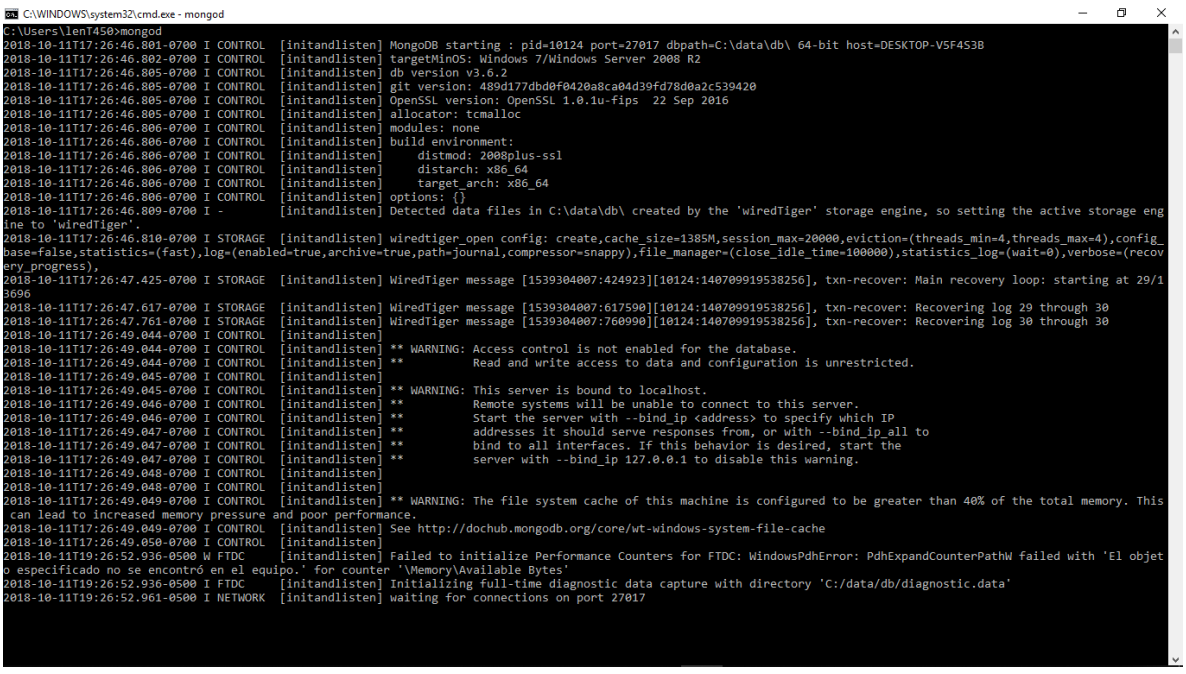


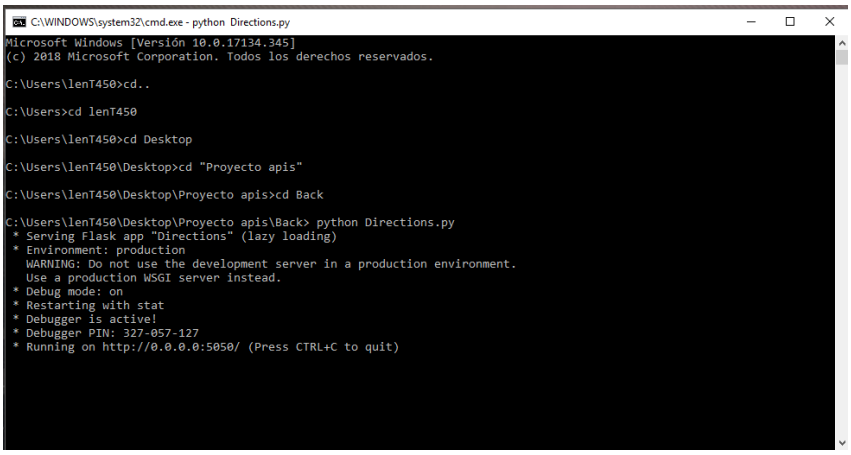
Si nuestra función fuera get nosotros en el method lo cambiamos a GET y los parámetros van en la de Request URL

Ejemplo get:

http://localhost:5050/api/fnRegisterUser/parametro1/parametro2

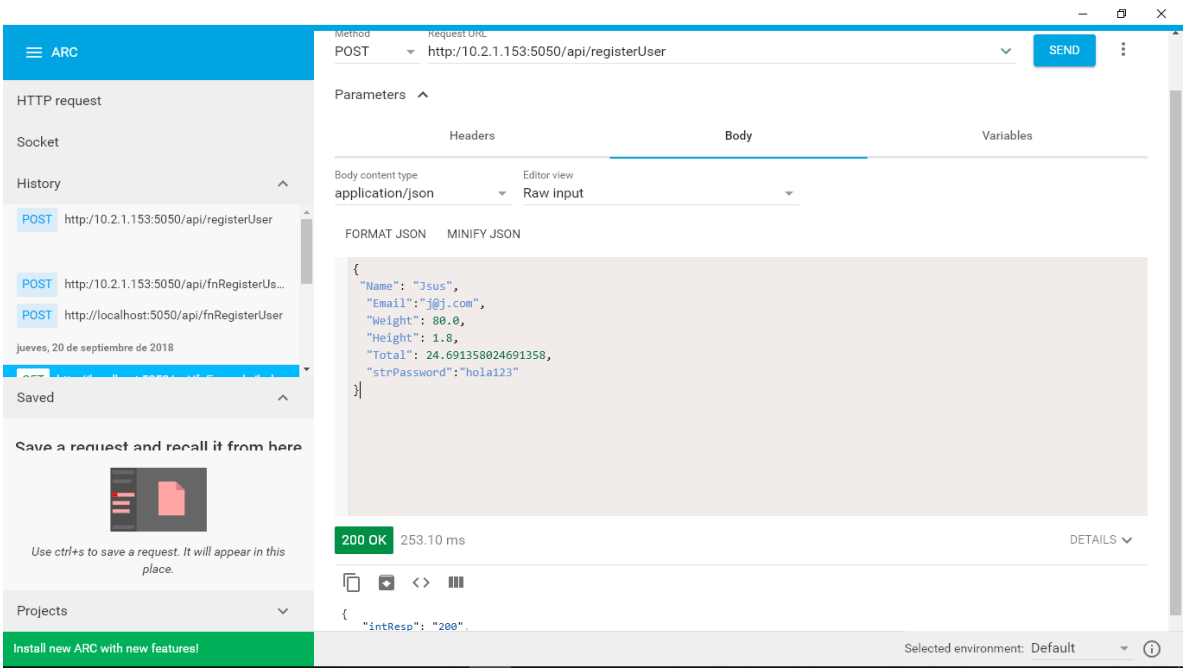
Antes de hacer lo de la prueba del api tenemos que habilitar nuestra base de datos y correr nuestro documento .py.



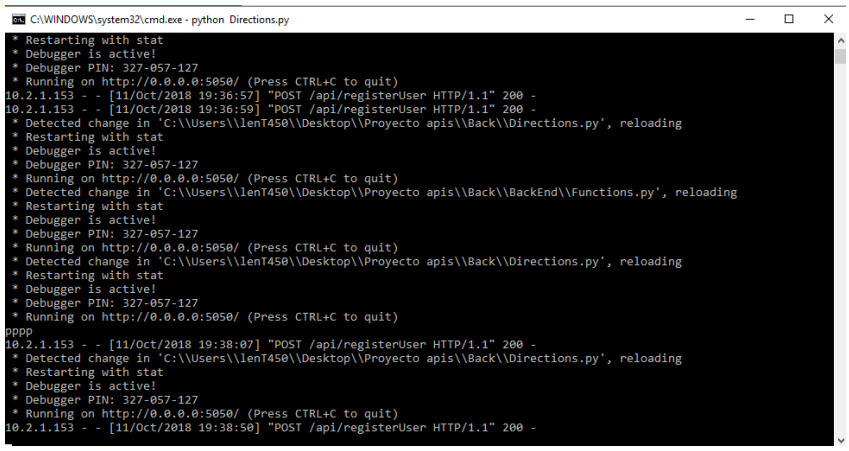


Si nuestra función está correctamente nos tendrá que devolver algo como esto.

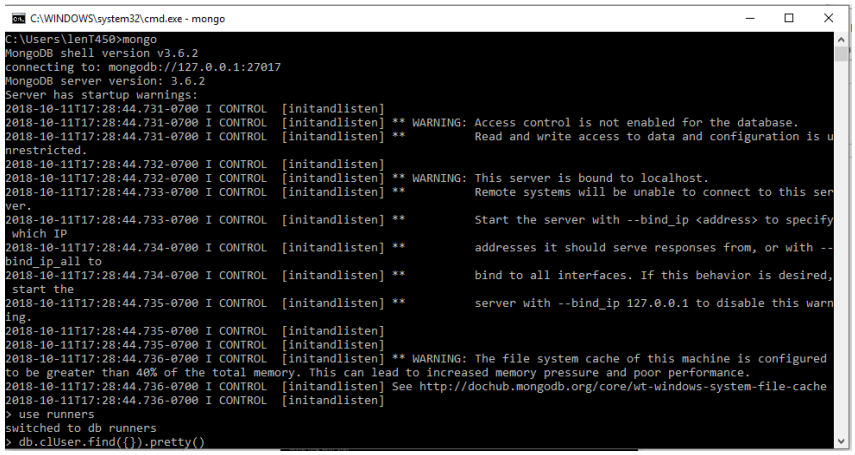
{
"intResp": "200",
"strMessage": "Usuer Jsus registered correctly"
}



Para ver la función nos vamos al archivo .py que iniciamos que en este caso es directions desde el cmd.



Abrimos otro cmd y ahí ponemos “mongo” una ves hecho esto usaremos la base de datos en este caso es “runners” asi que colocamos el comando “use runners”



Y con el comando db.clUser.find({}).pretty() nos mostrara el usuario guardado en la base de datos.

Ahora para que nuestra api funcione en el html tenemos que usar un javascript.

Javascript ejemplo:

```
var strIpAddress = "10.2.1.153" //Variable para hacer las conexiones a las
apis
                                //Aqui se coloca la direccion ipv4 para
poder usar nuestro sitio en otro dispositivo.
```

En la variable strIpAddres nosotros vamos a cambiarla por nuestra ip.

```
/*#########################################################
    # Inserta un usuario a la base de datos              #
    # Date:20/09/2018                                    #
    #########################################################*/
    function fnPutInsertNewUser(){
        var strName = $('#strName').val();
        var strEmail = $('#strEmail').val();
        var doubWeight = $('#doubWeight').val();
        var doubHeight = $('#doubHeight').val();
        var doubTotal = $('#doubTotal').val();
        var strPassword = $('#strPassword').val();
        var strConfirmPassword = $('#strConfirmPassword').val();
        if(strName != '' && strEmail != '' && doubWeight != '' && doubHeight
!= '' && doubTotal != ''
        && strPassword != '' && strConfirmPassword != '' && strPassword ==
strConfirmPassword){
            $.ajax({
                type: 'POST',
```

```
                url: 'http://' + strIpAddress + ':5050/api/registerUser',
                contentType: 'application/json; charset=utf-8',
                dataType: 'json',
                data: JSON.stringify({
                    'Name': strName,
                    'Email':strEmail,
                    'Weight':doubWeight,
                    'Height':doubHeight,
                    'Total':doubTotal,
                    'strPassword':strPassword
                }),
                success: function (result) {
                    if(result.intResp == '200'){
                        $('#mdlMessages').modal('show');
                        $('#mdlMessagesTitle').text('Successfull');
                        $('#mdlMessagesBody').text(result.strMessage);
                        $('#mdlMessagesBtnAction').html('<button
type="button" onclick="window.location.replace(\'index.html\')" class="btn
btn-default" data-dismiss="modal">Close</button>');
                    }else{
                        $('#mdlMessages').modal('show');
                        $('#mdlMessagesTitle').text('Error');
                        $('#mdlMessagesBody').text(result.strMessage);
                    }
                },
                error: function (result) {
                alert('had a error');
                }
            });
        }
    }
```

Html5 ejemplo:

```
<!--Inicio Form---------------------------------------------->
    <div class="container">
      <h2>Basic Form</h2>
      <form method="POST"> <!--action="insertar.php" method="POST"-->

        <div class="form-group">
          <label for="text">Name:</label>
          <input type="text" class="form-control" id="strName"
placeholder="Enter Name" name="Name" required>
        </div>

        <div class="form-group">
          <label for="email">Email:</label>
          <input type="email" class="form-control" id="strEmail"
placeholder="Enter email" name="Email" required>
        </div>

        <div class="form-group">
          <label for="password">Password:</label>
          <input type="password" class="form-control" id="strPassword"
placeholder="Password" name="pwd">
        </div>

          <div class="form-group">
             <label for="confirmPassword">Confirm Password:</label>
             <input type="password" class="form-control"
id="strConfirmPassword" placeholder="Confirm Password" min="0"
name="confirmPwd">
          </div>

        <div class="form-group">
          <label for="weight">Weight:</label>
          <input type="number" class="form-control" step="any"
id="doubWeight" onclick="fnImc()" onkeyup="fnImc()" placeholder="Weight"
min="0" name="Sal">
        </div>

        <div class="form-group">
            <label for="height">Height:</label>
            <input type="number" class="form-control" step="any"
id="doubHeight" onclick="fnImc()" onkeyup="fnImc()" placeholder="Height"
min="0" name="Incomes">
        </div>
```

```
        <div class="form-group">
            <label for="total">Total:</label>
            <input type="number" class="form-control" id="doubTotal"
placeholder="Total" name="Total" disabled>
        </div>

        <!-- <button type="submit" onclick="fnGetMessage()" class="btn btn-
success">Accept</button>-->
        <input type="button" onclick="fnPutInsertNewUser()" class="btn btn-
success" value="Accept">
        <button type="reset" class="btn btn-danger">Delete</button>

      </form>
    </div>
    <!--Fin Form---------------------------------------------->
```