Ricardo Alfonso Casanova Lozano.

Cristian Camilo Morales Tapias.

Configuración y Descarga de Python3

Descargamos Python 3.6 usando los siguientes comandos: *sudo apt-get update* y *sudo apt-get install python3.6.*

Después instalamos el pip: *sudo apt-get install python3-pip python3-dev*, ya con pip instalado hemos instalado el virtualenv con*: sudo pip3 install virtualenv* .

Preparamos el entorno virtual, primero creamos una carpeta iroom con: *mkdir iroom*, luego accedemos a la carpeta con: *cd iroom* y finalmente procedemos a preparar el entorno virtual con: *virtualenv flask.*

Para activar el entorno virtual lo hacemos con los siguientes comandos: *cd iroom* y *. flask/bin/actívate.* Comprobamos que nos encontramos en el entorno porque el sistema nos respondió de la siguiente forma:

```
(flask) ricardo@ubuntu:~/iroom/iroom$
```

Una vez en el entorno virtual, procedimos a instalar flask y los otros módulos necesarios: *pip3 install flask, pip3 install urllib3* y *pip3 install mysql-connector.*

PARTE 1 DESARROLLO DEL INTERFAZ ENTRE IROOM Y LA BASE DE DATOS.

Hemos realizado el ejercicio de la siguiente forma:

```
def updateSensor(code):
        db = mysql.connector.connect(host = "localhost", user = "ricardo", passwd = "rcasanova9", db = "BaseDeDatos")
        cursor = db.cursor()
        try:
                """ PARTE 1:COMPLETAR AQUÍ EL CÓDIGO PARA LEER EL VALOR DE UN SENSOR CON API REST"""
                response = http.request('GET', server + code)
                data = json.loads(response.data)
                value = data[code]
        except ValueError:
                print ('Error de leer dato del sensor')
        if value != last_value[code]:
                try:
                        type_sensor[code] = value
                        """ PARTE 1: COMPLETAR AQUÍ EL CÓDIGO PARA ESCRIBIR EN LA BASE DE DATOS EL VALOR DEL SENSOR"""
                        cursor.execute ("INSERT INTO sensors(nombre, valor) values(%s, %s)", (code, value))
                        db.commit()
                except ValueError:
                        print ('Error al insertar en base de datos')
        db.close()
```

De esta forma se lee el valor del sensor y se escribe en la base de datos.

```python
def controlLightColor():
        db = mysql.connector.connect(host = "localhost", user = "ricardo", passwd = "rcasanova9", db = "BaseDeDatos")
        cursor = db.cursor()
        for color in RGB_values:
                try:
                        cursor.execute ("SELECT valor FROM sensors WHERE nombre='"+color+"' ORDER BY time DESC LIMIT 1")
                        cursor_value = int(cursor.fetchone()[0])
                        if cursor_value != RGB_values[color]:
                                RGB_values[color] = cursor_value
                                print (color.capitalize()+": " + str(cursor_value))
                                response = http.request('PUT', server + color + '/' + str(cursor_value))
                except ValueError:
                        print ('Error al consultar de base de datos o conectar con iroom')
        db.close()
```

Después de completar el código, procedemos a cambiar un valor en la base de datos para comprobar que funciona:

```
ricardo@ubuntu:~/iroom$ python inserdb.py red 58
('red', '58')
```

Comprobamos que en la Base de Datos que se guardó el valor:

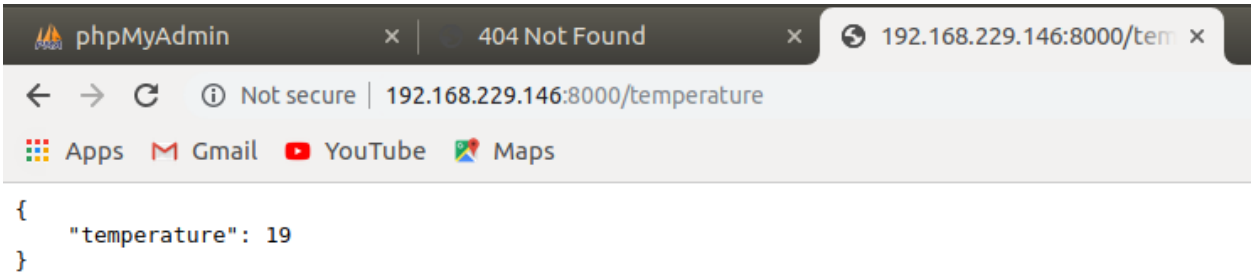
time	nombre	valor
2019-12-18 13:44:06	Temperatura	100
2019-12-18 13:44:23	Temperatura	34
2019-12-18 13:44:52	Temperatura	22
2019-12-18 13:45:13	red	43
2019-12-18 13:45:13	green	43
2019-12-18 13:45:13	blue	90
2019-12-18 13:49:30	Temperatura	200
2019-12-18 13:49:41	Temperatura	45
2019-12-19 01:05:22	red	149
2019-12-19 01:05:22	green	255
2019-12-19 01:05:22	blue	0
2019-12-19 01:07:01	red	0
2019-12-19 01:07:01	green	0
2019-12-19 01:07:01	blue	255
2019-12-19 01:07:03	red	0
2019-12-19 01:07:03	green	0
2019-12-19 01:07:03	blue	255
2019-12-19 04:47:26	red	58

Instalamos la librería flask-restful con: *pip3 install flask-restful*

```
(flask) ricardo@ubuntu:~/iroom/iroom$ sudo pip3 install flask-restful
[sudo] password for ricardo:
The directory '/home/ricardo/.cache/pip/http' or its parent directory is not ow
ned by the current user and the cache has been disabled. Please check the perm
ssions and owner of that directory. If executing pip with sudo, you may want su
do's -H flag.
The directory '/home/ricardo/.cache/pip' or its parent directory is not owned b
y the current user and caching wheels has been disabled. check the permissions
and owner of that directory. If executing pip with sudo, you may want sudo's -H
 flag.
Requirement already satisfied: flask-restful in /home/ricardo/.local/lib/python
3.6/site-packages
Requirement already satisfied: pytz in /home/ricardo/.local/lib/python3.6/site-
packages (from flask-restful)
Requirement already satisfied: six>=1.3.0 in /home/ricardo/.local/lib/python3.6
/site-packages (from flask-restful)
Requirement already satisfied: aniso8601>=0.82 in /home/ricardo/.local/lib/pyth
on3.6/site-packages (from flask-restful)
Requirement already satisfied: Flask>=0.8 in /home/ricardo/.local/lib/python3.6
/site-packages (from flask-restful)
Requirement already satisfied: click>=5.1 in /home/ricardo/.local/lib/python3.6
/site-packages (from Flask>=0.8->flask-restful)
Requirement already satisfied: itsdangerous>=0.24 in /home/ricardo/.local/lib/p
ython3.6/site-packages (from Flask>=0.8->flask-restful)
Requirement already satisfied: Jinja2>=2.10.1 in /home/ricardo/.local/lib/pytho
n3.6/site-packages (from Flask>=0.8->flask-restful)
Requirement already satisfied: Werkzeug>=0.15 in /home/ricardo/.local/lib/pytho
n3.6/site-packages (from Flask>=0.8->flask-restful)
Requirement already satisfied: MarkupSafe>=0.23 in /home/ricardo/.local/lib/pyt
hon3.6/site-packages (from Jinja2>=2.10.1->Flask>=0.8->flask-restful)
(flask) ricardo@ubuntu:~/iroom/iroom$
```

Se arranca el emulador con: *python3 emuiroom.py*

```
(flask) ricardo@ubuntu:~/iroom$ python3 emuiroom.py
 * Serving Flask app "emuiroom" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://0.0.0.0:8000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 154-514-423
192.168.229.146 - - [19/Dec/2019 05:01:48] "GET /temperatura HTTP/1.1" 404 -
192.168.229.146 - - [19/Dec/2019 05:01:48] "GET /favicon.ico HTTP/1.1" 404 -
192.168.229.146 - - [19/Dec/2019 05:01:51] "GET /temperature HTTP/1.1" 200 -
192.168.229.146 - - [19/Dec/2019 05:01:54] "GET /temperature HTTP/1.1" 200 -
192.168.229.146 - - [19/Dec/2019 05:01:55] "GET /temperature HTTP/1.1" 200 -
192.168.229.146 - - [19/Dec/2019 05:01:55] "GET /temperature HTTP/1.1" 200 -
```



PARTE 2: DESARROLLO DE LA APLICACIÓN WEB I: SENSORES.

Modificamos el archivo iroom.cfg

```
MYSQL_DATABASE_HOST = 'localhost'
MYSQL_DATABASE_PORT = 3306
MYSQL_DATABASE_USER = 'ricardo'
MYSQL_DATABASE_PASSWORD = 'rcasanova9'
MYSQL_DATABASE_DB = 'BaseDeDatos'
USERNAME = 'administrador'
PASSWORD = 'admin1234'
SECRET_KEY = 'admin'
```

Después configuramos la variable de entorno IROOM_SETTING, añadiéndolo al final del fichero .bashrc

$> cd nano .bashrc

Export IROOM_SETTINGS=/home/ricardo/iroom/iroom/config/iroom.cfg

Comprobamos que se ha exportado.

```
ricardo@ubuntu:~$ printenv IROOM_SETTINGS
/home/iroom/iroom/config/iroom.cfg
```

Configuramos la actualización de los sensores:

```
type_sensor = {"Temperatura": 0, "Humedad": 0, "Luz": 0, "Sonido": 0, "Movimiento": 0}
sensores = ["temperatura", "humedad", "luz", "sonido", "movimiento"]
```

Usamos una biblioteca para poder comparar los valores de sensores que se encuentran en la base de datos y un array para poder configurar los tipos de sensores

```python
def event_sensor():
        while True:
                conn = mysql.connect()
                cursor = conn.cursor()
                i=0
                for sensor in type_sensor:
                        cursor.execute ("SELECT valor FROM sensors where nombre='" + sensor + "' ORDER BY time DESC LIMIT 1")
                        ret_value = int(cursor.fetchone()[0])
                        if ret_value != type_sensor[sensor]:
                                j_data = {"tipo": sensores[i], "valor": ret_value}
                                data_json = json.dumps(j_data)
                                print (j_data)
                                yield 'data: %s\n\n' % str(data_json)
                                type_sensor[sensor] = ret_value
                                #flash("Actualizado sensor de " + sensor)
                        i=i+1

                conn.close()
```

Configuramos el acceso a sensores.html

```python
@app.route('/sensors')
def sensors():
        return render_template('sensors.html')
```

Problema de bloqueo.

```
(flask) ricardo@ubuntu:~/iroom/iroom$ pip3 install gevent
Collecting gevent
  Downloading https://files.pythonhosted.org/packages/f2/ca/5b5962361ed832847b6
b2f9a2d0452c8c2f29a93baef850bb8ad067c7bf9/gevent-1.4.0-cp36-cp36m-manylinux1_x8
6_64.whl (5.5MB)
    100% |[redacted]| 5.5MB 140kB/s
Collecting greenlet>=0.4.14; platform_python_implementation == "CPython" (from
gevent)
  Downloading https://files.pythonhosted.org/packages/bf/45/142141aa47e01a5779f
0fa5a53b81f8379ce8f2b1cd13df7d2f1d751ae42/greenlet-0.4.15-cp36-cp36m-manylinux1
_x86_64.whl (41kB)
    100% |[redacted]| 51kB 1.2MB/s
Installing collected packages: greenlet, gevent
Successfully installed gevent-1.4.0 greenlet-0.4.15
(flask) ricardo@ubuntu:~/iroom/iroom$
```

```
(flask) ricardo@ubuntu:~/iroom/iroom$ pip3 install gunicorn
Collecting gunicorn
  Downloading https://files.pythonhosted.org/packages/69/ca/926f7cd3a2014b16870
086b2d0fdc84a9e49473c68a8dff8b57f7c156f43/gunicorn-20.0.4-py2.py3-none-any.whl
(77kB)
    100% |[redacted]| 81kB 1.4MB/s
Collecting setuptools>=3.0 (from gunicorn)
  Cache entry deserialization failed, entry ignored
  Cache entry deserialization failed, entry ignored
  Downloading https://files.pythonhosted.org/packages/54/28/c45d8b54c1339f9644b
87663945e54a8503cfef59cf0f65b3ff5dd17cf64/setuptools-42.0.2-py2.py3-none-any.wh
l (583kB)
    100% |[redacted]| 583kB 1.0MB/s
Installing collected packages: setuptools, gunicorn
Successfully installed gunicorn-20.0.4 setuptools-42.0.2
(flask) ricardo@ubuntu:~/iroom/iroom$
```

```
(flask) ricardo@ubuntu:~/iroom/iroom$ gunicorn -k gevent -w 4 -b 0.0.0.0:8000 ir
oom:app
[2019-12-19 06:43:46 -0800] [3863] [INFO] Starting gunicorn 20.0.4
[2019-12-19 06:43:46 -0800] [3863] [INFO] Listening at: http://0.0.0.0:8000 (386
3)
[2019-12-19 06:43:46 -0800] [3863] [INFO] Using worker: gevent
[2019-12-19 06:43:46 -0800] [3866] [INFO] Booting worker with pid: 3866
[2019-12-19 06:43:46 -0800] [3867] [INFO] Booting worker with pid: 3867
[2019-12-19 06:43:46 -0800] [3868] [INFO] Booting worker with pid: 3868
[2019-12-19 06:43:46 -0800] [3869] [INFO] Booting worker with pid: 3869
```

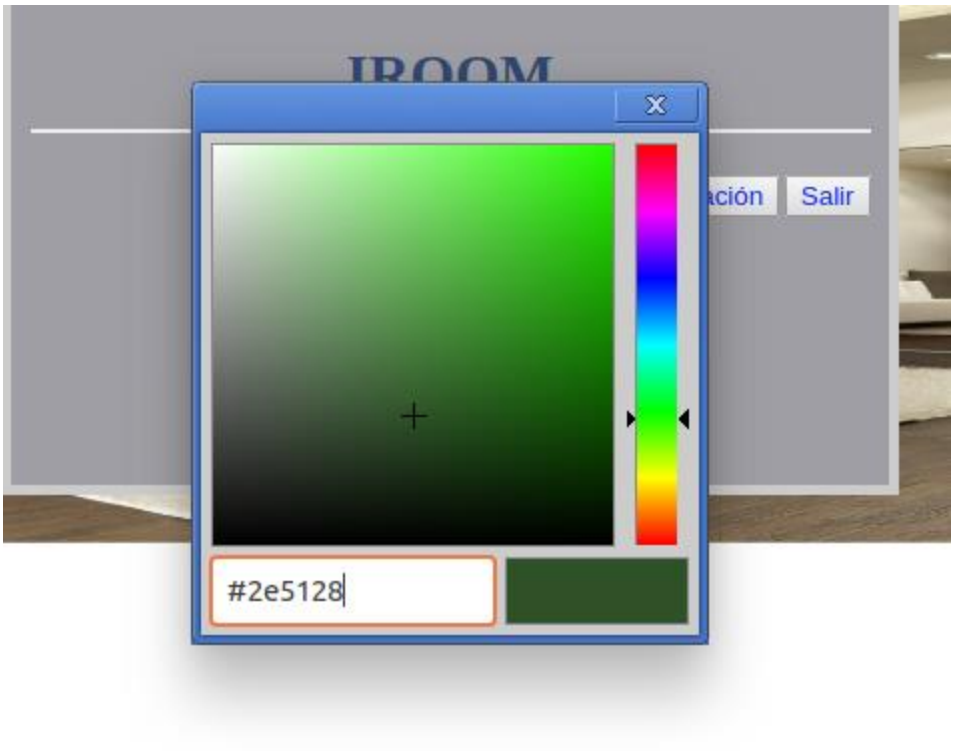
Comprobamos que funciona:

PARTE 3: DESARROLLO DE LA APLICACIÓN WEB II: ACTUADORES.

Configuramos la actualización de los colores.

```
@app.route('/setcolor', methods=['GET'])
def setcolor():
        conn = mysql.connect()
        cursor = conn.cursor()
        RGB = ["red", "green", "blue"]
        indice = 1
        for color in RGB:
                cursor.execute("INSERT INTO sensors(nombre, valor) VALUES(%s, %s)", (color, int("0x" +request.args.get("color")[indice:indice
+ 2],16)))
                indice += 2
        conn.commit()
        conn.close()
        return {'color': request.args.get("color")}
```

Enviamos el siguiente color

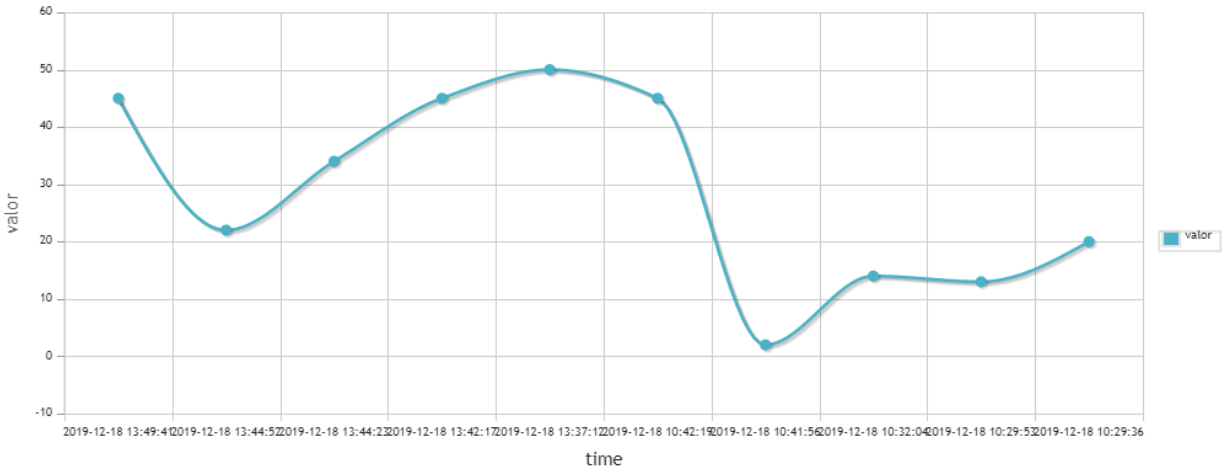


Comprobamos que se ha guardado en la Base de Datos en decimal.

2019-12-19 06:49:10	red	46
2019-12-19 06:49:10	green	81
2019-12-19 06:49:10	blue	40

<u>Mejoras:</u>

<u>Grafica de evolución histórica de la Temperatura:</u>



Mejorar la página web dada con css:

Alerta si se desea introducir un valor mayor: