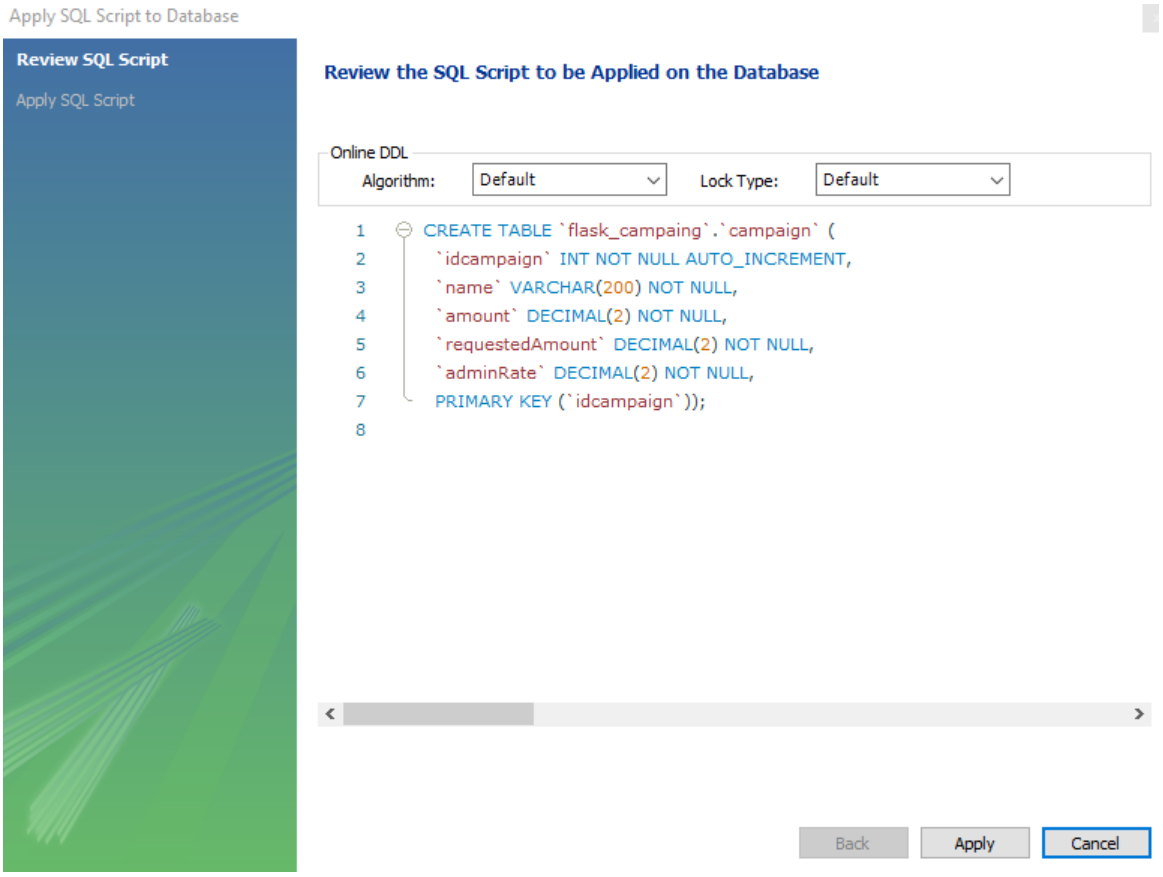
Autor: Jeisson Poveda.

Explicación y documentación.

Microservicio. A2censo.

El servicio debe retornar las campañas de forma ordenada por la cantidad de recursos o recursos solicitados.

1. Se crea la Base de datos usando el motor mysql.




```
INSERT INTO `flask_campaing`.`campaign` (`idcampaign`, `name`, `amount`, `requestedAmount`)
VALUES ('3', 'RobinFood2.0', '200000000', '250000000');

INSERT INTO `flask_campaing`.`campaign` (`idcampaign`, `name`, `amount`, `requestedAmount`)
VALUES ('1', 'T4 Tea For U', '1000000000', '1200000000');

INSERT INTO `flask_campaing`.`campaign` (`idcampaign`, `name`, `amount`, `requestedAmount`)
VALUES ('2', 'Smoking Burguers', '200000000', '200000000');
```

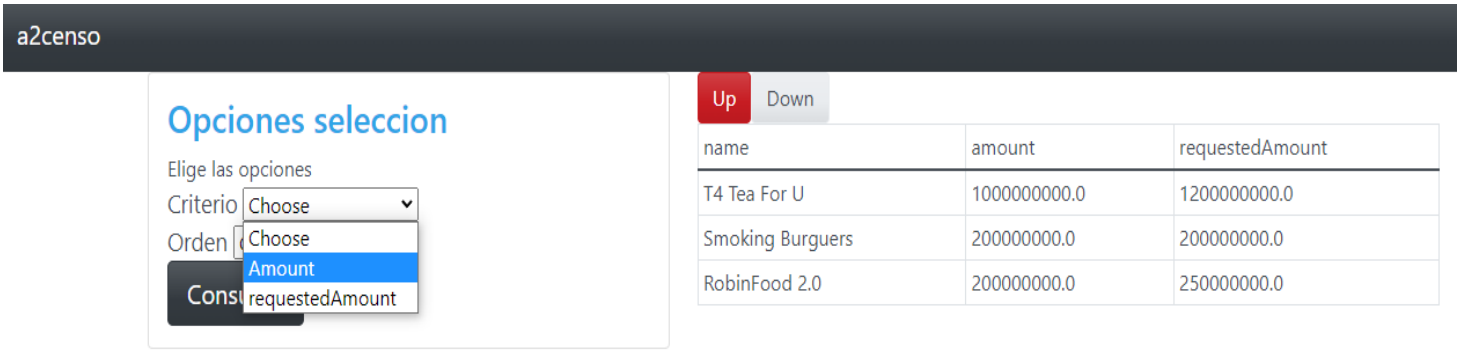
La creacion de la base de datos tambien es viable mediante sqlite

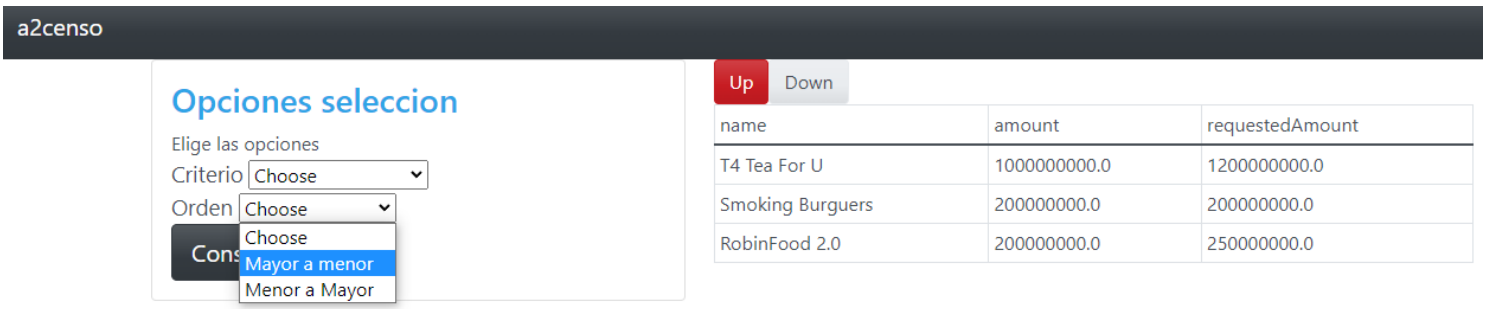
2. Mediante el framework FLASK y Python, se realiza la codificación del microservicio.
   Se tiene 3 archivos:
   - App.py: el cual contiene el Backend, es decir, la lógica que permite el funcionamiento de la página web.
   - Index.html: Se decribe el front de la aplicación web.
   - Layout.html: se le realiza el diseño de estilos usando bootstrap.

En el Front de la aplicación web, se le da al usuario la opción de elegir por:

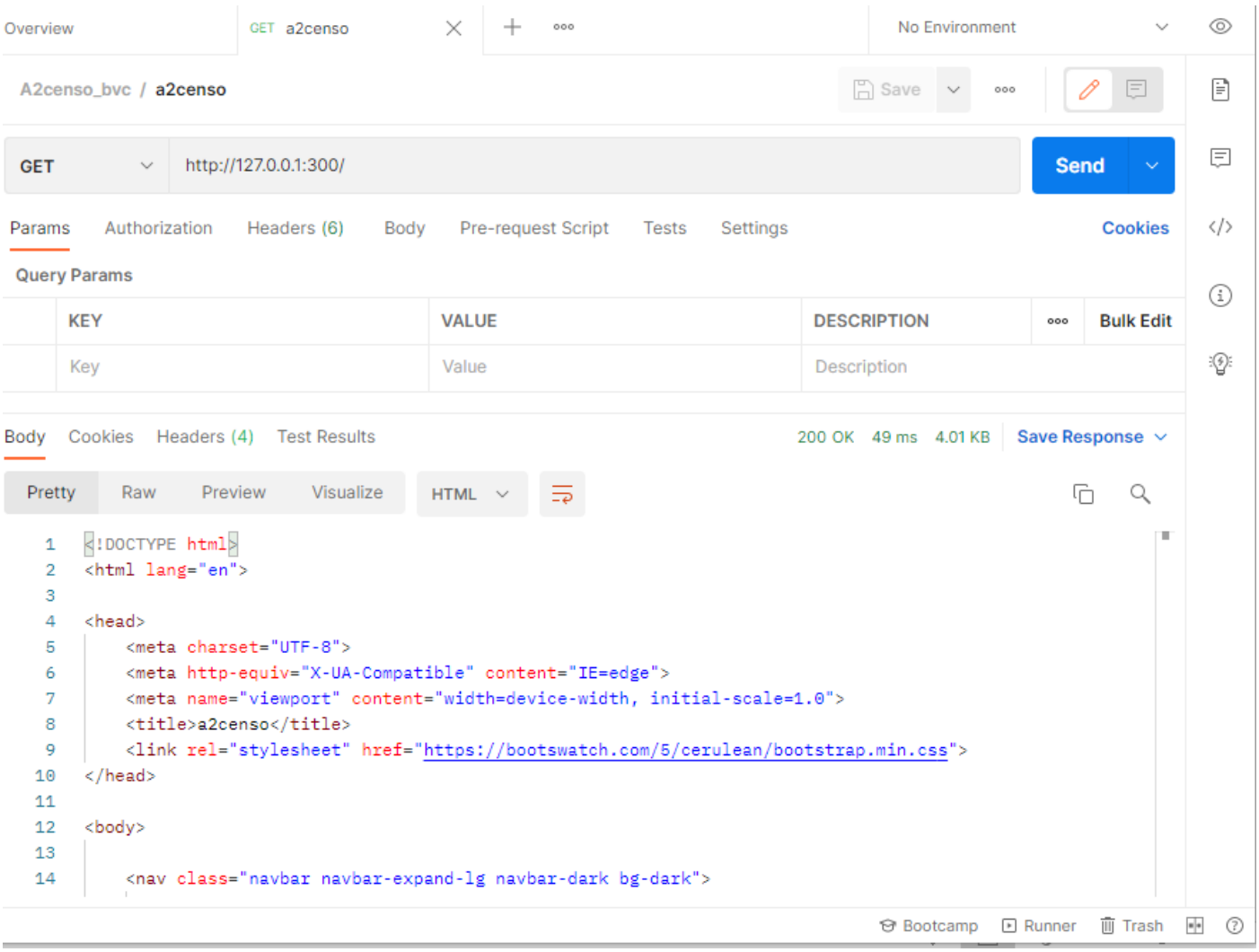
- Criterio: Amount o requestedAmount.

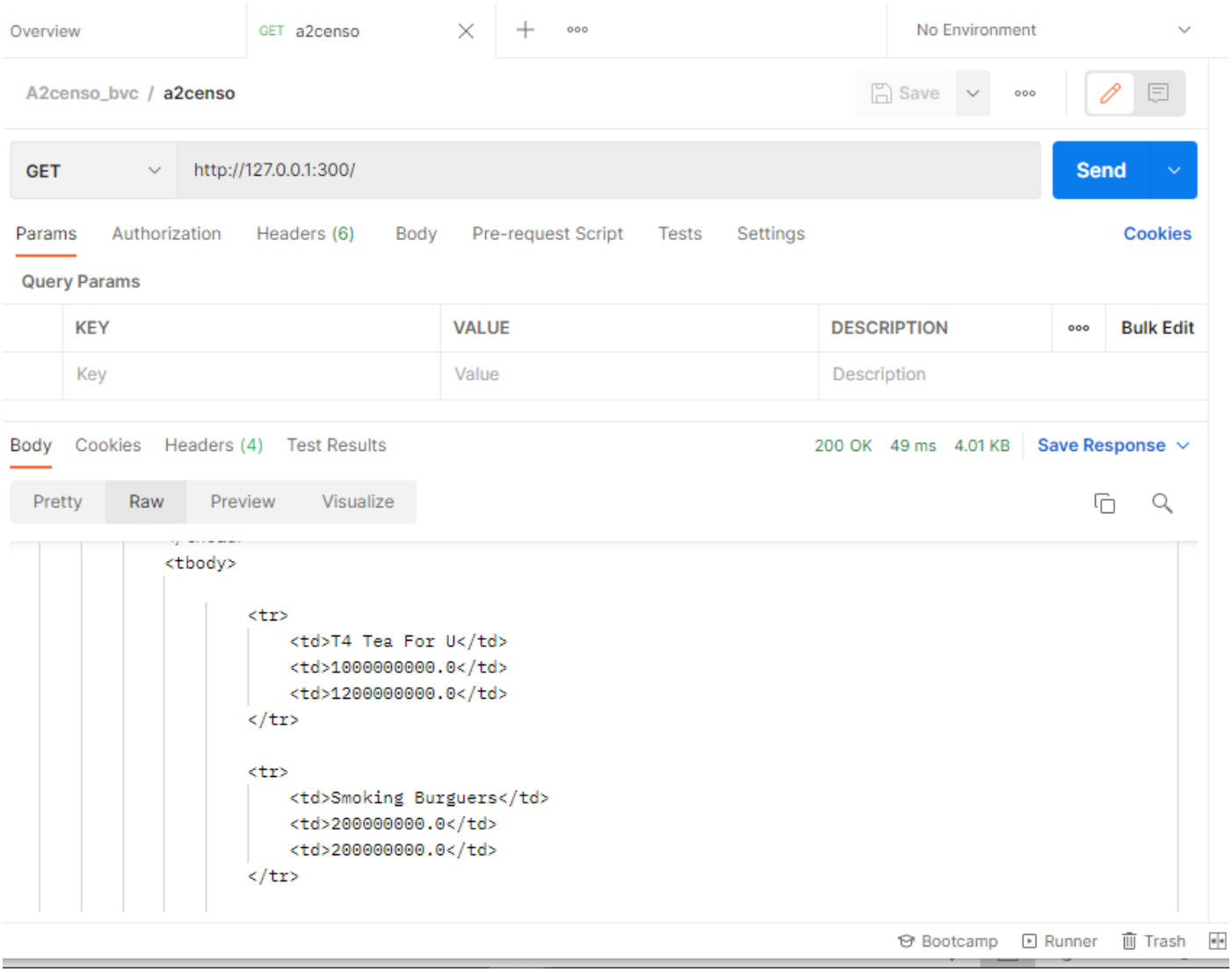
- Orden: Mayor a menor, menor a mayor.





En postman se realiza la peticiones al microservicio, comprobando su correcto funcionamiento.





Bloque de código que permite traer todos los datos de la Base de datos creada en MySQL.

```
#Guardar datos seetings
app.secret_key='mysecretkey'
@app.route('/')  #
def Index():
    cur=mysql.connection.cursor()
    cur.execute('SELECT *FROM campaign')
    data=cur.fetchall() #para obtener todos esos datos
    print(data)

    return render_template('index.html',contacts=data)  # cada vez que el usuario visite la ruta
```

Bloque de código que realiza la lógica de ordenamiento a partir de la selección por parte del usuario.

```python
@app.route('/data', methods=['POST'])
def data():
    if request.method=='POST':
        criterio= request.form.get("sim")
        orden=request.form.get("config")
        print("Este es el criterio: " +criterio)
        print("Orden: "+ orden)
        print(type(orden))
        datos=()
        print(datos)

        if orden=="ASC":
            cur=mysql.connection.cursor()
            cur.execute('SELECT * FROM campaign ORDER BY (%s) ASC',(criterio,))
            mysql.connection.commit()
            datos=cur.fetchall()
            print(datos)
            print("pasa ASC")
        elif orden=='DESC':
            cur=mysql.connection.cursor()
            cur.execute('SELECT * FROM campaign ORDER BY (%s) DESC',(criterio,))
            mysql.connection.commit()
            datos=cur.fetchall()
            print("pasa DESC")
        print(datos)
    return render_template('index.html',contacts=datos)

```

Testing.

Con selenium es una buena opción para realizar las pruebas del microservicio, y en una metodología screenplay con el uso de cucumber y serenity se tendría una prueba que da una historia de usuario.