

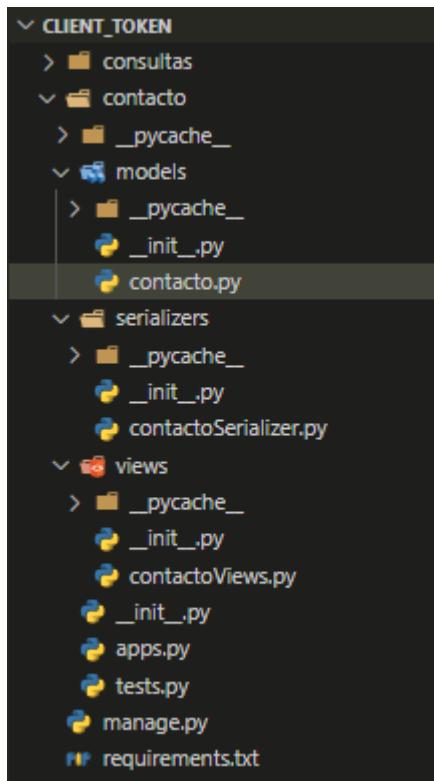
## Segundo Microservicio:

**Link del Repositorio:** <https://github.com/DavidCastro88/4a-ms2>

**Link de Despliegue:** <https://misiontic-g02.herokuapp.com>

**Framework:** Django **Base de Datos:** MongoDB

**Atención al Cliente:** Por medio de un formulario donde se le pregunta al cliente la razón de su inquietud junto con sus datos personales.



## Modelo:

```
from django import models

class Contacto(models.Model):
    inquietudes = (
        (0, "Producto"),
        (1, "Servicio de entrega"),
        (2, "Otro"),
    )
    _id = models.ObjectIdField(primary_key=True)
    nombre = models.CharField(max_length=50, null=True)
    telefono = models.IntegerField(null=True)
    email = models.EmailField(max_length=50, null=True)
    inquietudes = models.CharField(max_length=2, choices=inquietudes, null=True)
    comentario = models.TextField(max_length=2000, null=True)
    sol_activa = models.BooleanField(default=True, null=True)
```

## Serializer:

```
from django.db.models import fields
from rest_framework import serializers
from contacto.models import Contacto

class contactoSerializer(serializers.ModelSerializer):
    class Meta:
        model = Contacto
        fields = '__all__'

    def to_representation(self, instance):
        return {
            "_id": str(instance._id),
            "nombre": instance.nombre,
            "email": instance.email,
            "telefono": instance.telefono,
            "inquietudes": instance.inquietudes,
            "comentario": instance.comentario,
            "sol_activa": instance.sol_activa,
        }
```

## Views:

```
from contacto.models import Contacto
from contacto.serializers import contactoSerializer
from rest_framework import generics, status
from django.shortcuts import get_object_or_404
from rest_framework.response import Response
from bson.objectid import ObjectId
from django.core import serializers
import json

class contactoCreate(generics.ListCreateAPIView):
    serializer_class = contactoSerializer
    queryset = Contacto.objects.all()

class contactoDetails(generics.RetrieveAPIView):
    serializer_class = contactoSerializer
    def get(self, request, *args, **kwargs):
        """
        se crea un objeto de tipo objectid para filtrar por la llave en mongo atlas,
        luego se trae el objeto y se serializa para enviar luego los campos necesarios
        """
        contacto = get_object_or_404(Contacto, _id=ObjectId(kwargs["_id"]))
        serialized_obj = serializers.serialize('json', (contacto,))
        struct = json.loads(serialized_obj)
        return Response(struct[0]["fields"], status=status.HTTP_200_OK)

class filterByEmail(generics.ListAPIView):
    serializer_class = contactoSerializer
    def get_queryset(self):
        queryset = Contacto.objects.filter(email=self.kwargs["email"])
        return queryset
```

## URLS:

```
from django.urls import path
from contacto import views

urlpatterns = [
    path('contacto/', views.contactoCreate.as_view()), # Crear una sol de contacto o listar todas
    path('contacto/', views.contactoDetails.as_view()), # Obtener una solicitud por id, modificar o borrar
    path('contacto/email/<str:email>', views.filterByEmail.as_view()), # Obtener solicitudes por email
]
```

## Pruebas en los endpoints:

### Post:

The screenshot shows a Postman request to `https://misiontic-g02.herokuapp.com/contacto/`. The response status is `201 Created`, size is `174 Bytes`, and time is `727 ms`. The JSON response body is:

```
1 ↴ [
2   "_id": "61a3cf849208a3a884c7e5e5",
3   "nombre": "alexandra",
4   "email": "alexandra@gmail.com",
5   "telefono": 202020202,
6   "inquietudes": 0,
7   "comentario": "despliegue exitoso",
8   "sol_activa": true
9 ]
```

## Filtrar por correo electrónico:

The screenshot shows a Postman request to `https://misiontic-g02.herokuapp.com/contacto/email/alexandra@`. The response status is `200 OK`, size is `178 Bytes`, and time is `687 ms`. The JSON response body is:

```
1 ↴ []
2 ↴ [
3   {
4     "_id": "61a3cf849208a3a884c7e5e5",
5     "nombre": "alexandra",
6     "email": "alexandra@gmail.com",
7     "telefono": 202020202,
8     "inquietudes": 0,
9     "comentario": "despliegue exitoso",
10    "sol_activa": true
11  }
11 ]
```

## Filtrar por ID:

```
GET https://misiontic-g02.herokuapp.com/contacto/61a3cb51305ea3a8afa0a541
```

Status: 200 OK Size: 133 Bytes Time: 1.03 s

Response Headers Cookies Test Results { }  
1 ↴ [ {  
2 "nombre": "carlos",  
3 "telefono": "2344444",  
4 "email": "carlos@gmail.com",  
5 "inquietudes": "1",  
6 "comentario": "contacto exitoso",  
7 "sol\_activa": true  
8 } ]

## Get de todos los contactos:

```
GET https://misiontic-g02.herokuapp.com/contacto/
```

Status: 200 OK Size: 833 Bytes Time: 572 ms

Response Headers Cookies Test Results { }  
1 ↴ [ [ {  
2 "\_id": "61a3c9eb1cf9d0cce04d4117",  
3 "nombre": "martin",  
4 "email": "yyyyy@gmail.com",  
5 "telefono": "41723283",  
6 "inquietudes": "0",  
7 "comentario": "su clenor",  
8 "sol\_activa": true  
9 },  
10 {  
11 "\_id": "61a3cac3305ea3a8afa0a53f",  
12 "nombre": "hugo",  
13 "email": "lolo@gmail.com",  
14 "telefono": "41723283",  
15 "inquietudes": "0",  
16 "comentario": "su clenor",  
17 "sol\_activa": true  
18 },  
19 {  
20 "\_id": "61a3cb03305ea3a8afa0a540",  
21 "nombre": "oscar",  
22 "email": "oscar@gmail.com",  
23 "telefono": "31489393",  
24 "inquietudes": "2",  
25 "comentario": "5 horas dandole a esto".  
26 } ]

## Conexión con la base de datos de MONGO-Atlas:

```
85 DATABASES = {  
86     # 'default': {  
87         #     'ENGINE': 'django.db.backends.sqlite3',  
88         #     'NAME': BASE_DIR / 'db.sqlite3',  
89         # }  
90     'default': {  
91         'ENGINE': 'djongo',  
92         'NAME': 'Contacto',  
93         'ENFORCE_SCHEMA': False,  
94         'CLIENT': {  
95             'host': 'mongodb+srv://DavidC:123U5678@cluster0.ndlt2.mongodb.net/myFirstDatabase?retryWrites=true&w=majority',  
96             'authMechanism': 'SCRAM-SHA-1',  
97         }  
98     }  
99 }  
100 }
```

## Atlas con las datos:

The screenshot shows the MongoDB Atlas interface with the 'Collections' tab selected for the 'Contacto.contacto\_contacto' collection. The left sidebar shows 'Project 0' and various service links like Data Lake, Data Services, Security, and Data API (Preview). The main panel displays the collection details: Collection Size: 732B, Total Documents: 5, Indexes Total Size: 36KB. It includes tabs for Find, Indexes, Schema Anti-Patterns, Aggregation, and Search Indexes. A query results table shows 1-5 of 5 documents, each with fields: \_id, nombre, telefono, email, inquietudes, comentario, and sol\_activa. Below the table, three specific document examples are shown in code format.

```
_id: ObjectId("61a3cb03305ea3a8afa0a540")
nombre: "oscar"
telefono: 31489393
email: "oscar@gmail.com"
inquietudes: "2"
comentario: "5 horas dandole a esto"
sol_activa: true
```

```
_id: ObjectId("61a3cb51305ea3a8afa0a541")
nombre: "carlos"
telefono: 2344444
email: "carlos@gmail.com"
inquietudes: "1"
comentario: "contacto exitoso"
sol_activa: true
```

```
_id: ObjectId("61a3cf849208a3a884c7e5e5")
nombre: "alexandra"
telefono: 20202020
email: "alexandra@gmail.com"
inquietudes: "0"
comentario: "despliegue exitoso"
sol_activa: true
```

## EVIDENCIA DE TRABAJO EN EL JIRA:

Sprint #3 21 nov. – 28 nov. (4 incidencias)		0 0 0	Completar sprint	...
<span style="color: green;">■</span>	G2-10 Escoger un Framework para el segundo microservicio	<span style="color: green;">FINALIZADA</span>		
<span style="color: green;">■</span>	G2-11 Desplegar y conectar una base de datos No relacional	<span style="color: green;">FINALIZADA</span>		
<span style="color: green;">■</span>	G2-12 Crear un repositorio llamado 4a-ms2	<span style="color: green;">FINALIZADA</span>		
<span style="color: green;">■</span>	G2-13 Implementar un CRUD en este microservicio, en el caso de spring crear al menos un modelo, un repositorio y 4 métodos	<span style="color: green;">FINALIZADA</span>		
<a href="#">+ Crear incidencia</a>				