

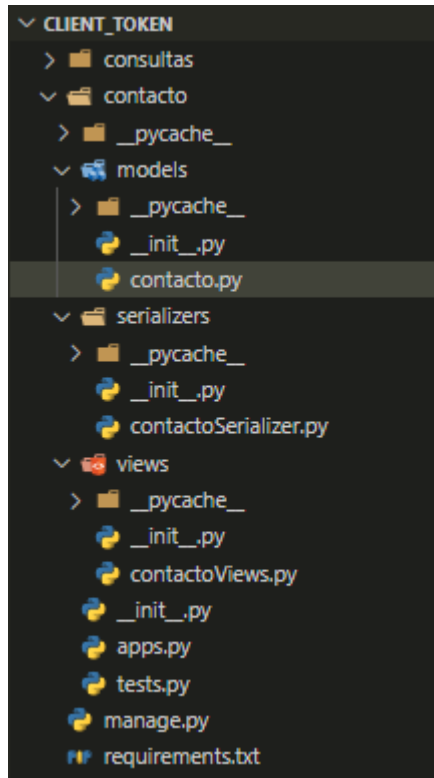
## Segundo Microservicio:

**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/<str:id>', 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 REST client interface with a POST request to `https://misiontic-g02.herokuapp.com/contacto/`. The request body is a JSON object: `{ "nombre": "alexandra", "email": "alexandra@gmail.com", "inquietudes": "0", "telefono": "202020202", "comentario": "despliegue exitoso" }`. The response status is `201 Created` with a size of `174 Bytes` and a time of `727 ms`. The response body is a JSON object: `{ "_id": "61a3cf849208a3a884c7e5e5", "nombre": "alexandra", "email": "alexandra@gmail.com", "telefono": "202020202", "inquietudes": "0", "comentario": "despliegue exitoso", "sol_activa": true }`.

### Filtrar por correo electrónico:

The screenshot shows a REST client interface with a GET request to `https://misiontic-g02.herokuapp.com/contacto/email/alexandra@gmail.com`. The response status is `200 OK` with a size of `178 Bytes` and a time of `687 ms`. The response body is a JSON array containing one object: `[ { "_id": "61a3cf849208a3a884c7e5e5", "nombre": "alexandra", "email": "alexandra@gmail.com", "telefono": "202020202", "inquietudes": "0", "comentario": "despliegue exitoso", "sol_activa": true } ]`.

## Filtrar por ID:

GET <https://misionotic-g02.herokuapp.com/contacto/61a3cb51305ea3a8afa0a541> Send

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

Query Headers<sup>2</sup> Auth Body Tests Response Headers<sup>11</sup> Cookies Test Results {}

Json Xml Text Form Form-encode GraphQL Binary

Json Content

```
1 |
```

```
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://misionotic-g02.herokuapp.com/contacto/> Send

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

Query Headers<sup>2</sup> Auth Body Tests Response Headers<sup>11</sup> Cookies Test Results {}

Json Xml Text Form Form-encode GraphQL Binary

Json Content

```
1 |
```

```
1 [
2   {
3     "_id": "61a3c9eb1cf9d0cce04d4117",
4     "nombre": "martin",
5     "email": "yyyyyy@gmail.com",
6     "telefono": 41723283,
7     "inquietudes": "0",
8     "comentario": "su claor",
9     "sol_activa": true
10  },
11  {
12    "_id": "61a3cac3305ea3a8afa0a53f",
13    "nombre": "hugo",
14    "email": "lolo@gmail.com",
15    "telefono": 41723283,
16    "inquietudes": "0",
17    "comentario": "su claor",
18    "sol_activa": true
19  },
20  {
21    "_id": "61a3cb03305ea3a8afa0a540",
22    "nombre": "oscar",
23    "email": "oscar@gmail.com",
24    "telefono": 31489393,
25    "inquietudes": "2",
26    "comentario": "5 horas dandole a esto".
27  }
28 ]
```

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

```
85
86 DATABASES = {
87   # 'default': {
88     # 'ENGINE': 'django.db.backends.sqlite3',
89     # 'NAME': BASE_DIR / 'db.sqlite3',
90   # }
91   'default': {
92     'ENGINE': 'djongo',
93     'NAME': 'Contacto',
94     'ENFORCE_SCHEMA': False,
95     'CLIENT': {
96       'host': 'mongodb+srv://DavidC:123U5678@cluster0.ndlt2.mongodb.net/myFirstDatabase?retryWrites=true&w=majority',
97       'authMechanism': 'SCRAM-SHA-1',
98     }
99   }
100 }
101
```

Atlas con las datos:

Mision Tic

Access Manager

Billing

All Clusters

Get Help

David

Project 0

Atlas

Realm

Charts

DEPLOYMENT

DATABASES

DATA SERVICES

SECURITY

Overview

Real Time

Metrics

Collections

Search

Profiler

Performance Advisor

Online Archive

Command Lir

DATABASES: 1 COLLECTIONS: 13

+ Create Database

NAMESPACES

Namespaces

Microservicio

\_\_schema\_\_

auth\_group

auth\_group\_permissions

auth\_permission

auth\_user

auth\_user\_groups

auth\_user\_user\_permi...

contacto\_contacto

django\_admin\_log

django\_content\_type

Contacto.contacto\_contacto

COLLECTION SIZE: 732B TOTAL DOCUMENTS: 5 INDEXES TOTAL SIZE: 36KB

Find

Indexes

Schema Anti-Patterns

Aggregation

Search Indexes

INSERT DOCUMENT

FILTER { field: 'value' }

OPTIONS

Apply

Reset

QUERY RESULTS 1-5 OF 5

\_id: ObjectId("61a3cb03305ea3a8afa0a540")

nombre: "martin"

telefono: 41723283

email: "yyyyyy@gmail.com"

inquietudes: "0"

comentario: "su clao"

sol\_activa: true

```
_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: 202020202
email: "alexandra@gmail.com"
inquietudes: "0"
comentario: "despliegue exitoso"
sol_activa: true
```

EVIDENCIA DE TRABAJO EN EL JIRA: