

# Mobilender\_technical\_test

---

Hola, soy Jorge Armando Blanquicett Matos, aspirante al puesto de desarrollador senior backend Python

El proyecto se encuentra en mi repositorio personal

[https://github.com/jblanquicett92/mobilender\\_technical\\_test](https://github.com/jblanquicett92/mobilender_technical_test),

el README.me cuenta con las especificaciones necesarias para arrancar el proyecto

--  
**"La única forma de  
hacer un gran trabajo,  
es amar lo que haces"**

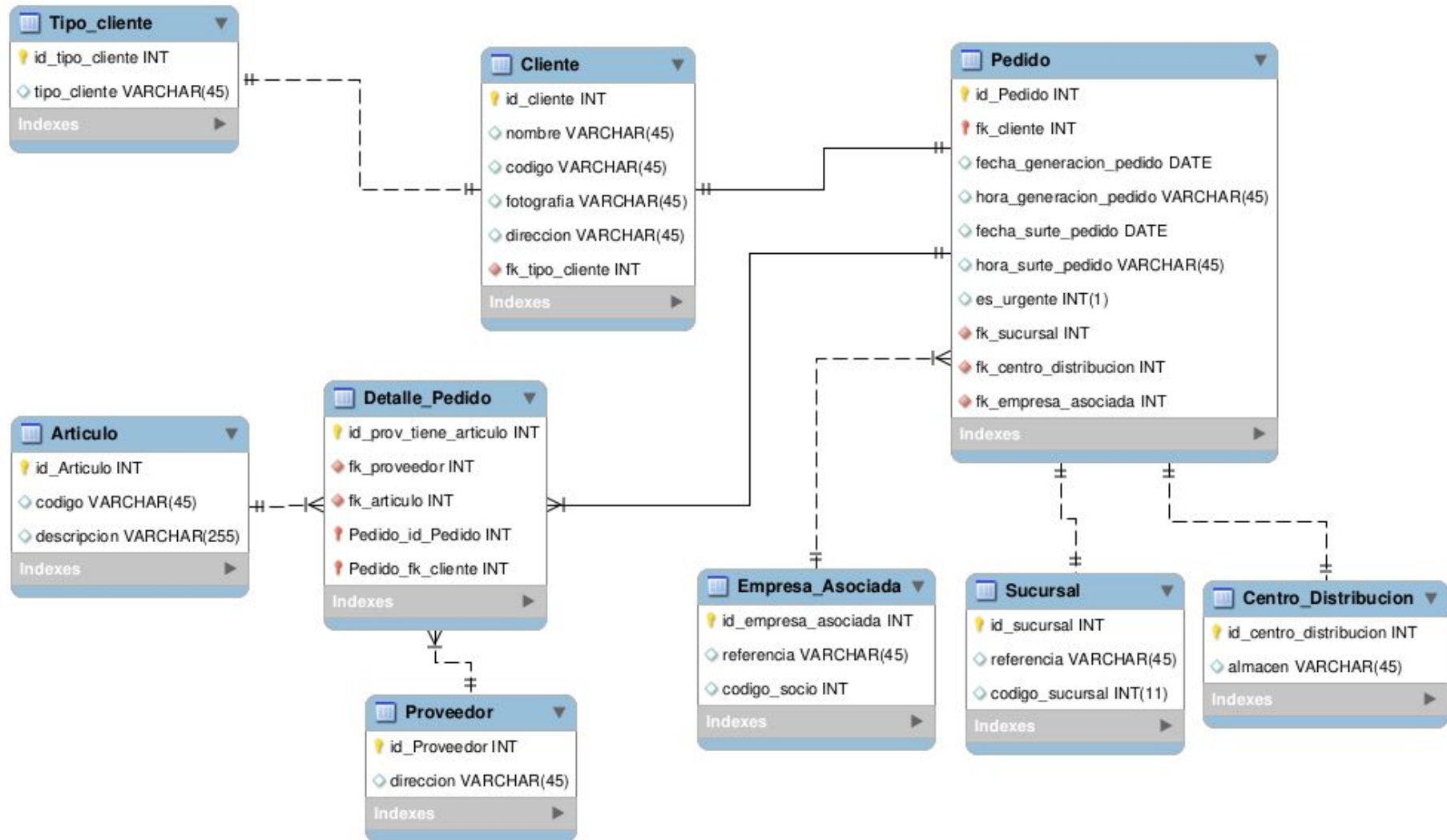
**—Steve Jobs**



# Mobilender\_technical\_test

---

1. Proponer un modelo de base de datos relacional que refleje adecuadamente las necesidades del sistema.



# Mobilender\_technical\_test

---

1. Diseñar las clases Django que representen las entidades del modelo propuesto en el punto anterior.

teniendo en cuenta el principio de composición indicó las clases no débiles primero.

— — —

```
class Tipo_cliente(models.Model):
    id_tipo_cliente = models.AutoField(primary_key=True)
    tipo_cliente = models.CharField(max_length=45)

    def __str__(self):
        return f'id: {self.id_tipo_cliente} tipo cliente: {self.tipo_cliente}'
```

```
class Cliente(models.Model):
    id_cliente = models.AutoField(primary_key=True)
    nombre = models.CharField(max_length=45)
    codigo = models.CharField(max_length=45)
    fotografia = models.CharField(max_length=45)
    direccion = models.CharField(max_length=45)
    fk_tipo_cliente = models
    .ForeignKey(Tipo_cliente, on_delete=models.SET_NULL, null=
    True)

    def __str__(self):
        return f'id: {self.id_cliente} nombre: {self
    .nombre} codigo: {self.codigo} fk_tipo_cliente: {self
    .fk_tipo_cliente}'
```

```
class Artículo(models.Model):
    id_articulo = models.AutoField(primary_key=True)
    codigo = models.CharField(max_length=45)
    descripcion = models.CharField(max_length=255)
    precio = models.FloatField()

    def __str__(self):
        return f'id: {self.id_articulo} tipo codigo: {self.codigo}'

class Proveedor(models.Model):
    id_proveedor = models.AutoField(primary_key=True)
    nombre = models.CharField(max_length=45)
    direccion = models.CharField(max_length=45)

    def __str__(self):
        return f'id: {self.id_proveedor} nombre: {self.nombre}'

class Empresa_asociada(models.Model):
    id_empresa_asociada = models.AutoField(primary_key=True)
    referencia = models.CharField(max_length=45)
    codigo_socio = models.IntegerField()

    def __str__(self):
        return f'id: {self.id_empresa_asociada} referencia: {self.referencia}'
```




```
class Sucursal(models.Model):
    id_sucursal = models.AutoField(primary_key=True)
    referencia = models.CharField(max_length=45)
    codigo_sucursal = models.IntegerField()

    def __str__(self):
        return f'id: {self.id_sucursal} referencia: {self.referencia}'

class Centro_distribucion(models.Model):
    id_centro_distribucion = models.AutoField(primary_key=True)
    almacen = models.CharField(max_length=45)

    def __str__(self):
        return f'id: {self.id_centro_distribucion} referencia: {self.referencia}'
```





```
class Pedido(models.Model):

    id_pedido = models.AutoField(primary_key=True)
    fk_cliente = models.ForeignKey(Cliente, on_delete=
models.SET_NULL, null=True)
    fecha_gen_pedido = models.DateField()
    hora_gen_pedido = models.CharField(max_length=45)
    fecha_surte_pedido = models.DateField(null=True)
    hora_surte_pedido = models.CharField(max_length=45
, null=True)
    es_urgente = models.BooleanField()
    fk_sucursal = models.ForeignKey(Sucursal, on_delete=
models.SET_NULL, null=True)
    fk_centro_distribucion = models
.ForeignKey(Centro_distribucion, on_delete=models.SET_NULL
, null=True)
    fk_empresa_asociada = models
.ManyToManyField(Empresa_asociada, blank=True)
```



```
class Detalle_pedido(models.Model):  
    id_detalle_pedido = models.AutoField(primary_key=True)  
    fk_proveedor = models.ManyToManyField(Proveedor, blank=True)  
    fk_articulo = models.ManyToManyField(Articulo, blank=True)  
    fk_cliente = models.ForeignKey(Cliente, on_delete=models.SET_NULL, null=True)  
    fk_pedido = models.ForeignKey(Pedido, on_delete=models.SET_NULL, null=True)
```

# Mobilender\_technical\_test

---

3. Escribir un "administrador de pedidos" que tenga dos servicios:

1. Crear un pedido (considerar tener la información extra necesaria en fixtures).
2. Obtener la relación de pedidos urgentes a Centros de distribución que pertenezcan a cliente PLATINO y que aún no se han surtido.

1

---

## Untitled Request

POST  [http://127.0.0.1:8000/api/admin\\_pedidos/crear\\_pedido/](http://127.0.0.1:8000/api/admin_pedidos/crear_pedido/)

Send Save

Params   Authorization   Headers (8)   **Body**   Pre-request Script   Tests   Settings

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL JSON ▼

```
1 {
2   "fk_cliente": 2,
3   "fecha_gen_pedido": "2021-07-11",
4   "hora_gen_pedido": "00:02:06",
5   "es_urgente": true,
6   "fk_sucursal": 0,
7   "fk_centro_distribucion": 1,
8   "fk_empresa_asociada": 0,
9   "fk_proveedor": 2,
10  "fk_articulo": [
11    {
12      "fk_articulo": 3
13    },
14    {
15      "fk_articulo": 2
16    }
17  ]
18 }
```

Body Cookies Headers Test Results

Pretty Raw Preview Visualize Text ▼

admin\_pr x

Authentication

admin\_pedidos v

POST admin\_pedidos\_crear\_pedido...

GET admin\_pedidos\_listar\_urgent...

articulo >

articulo\_en\_proveedor >

centro\_distribucion >

cliente >

detalle\_pedido >

empresa\_asociada >

pedido >

proveedor >

proveedor\_tiene\_articulo >

sucursal >

tipo\_cliente >

Documentation Powered by ReDoc

## admin\_pedidos\_crear\_pedido\_create

AUTHORIZATIONS: Basic

QUERY PARAMETERS

fk_cliente	integer <int64> [id_cliente] por ejemplo: 1
fecha_gen_pedido	string <date> [Fecha DD-MM-YYYY] por ejemplo: 2021-07-12
hora_gen_pedido	string [Alfanumérico] por ejemplo: 10:15
es_urgente required	boolean [Booleano] por ejemplo: False
fk_sucursal required	integer <int64> [id_sucursal] por ejemplo: 1
fk_centro_distribucion required	integer <int64> [id_centro_distribucion] por ejemplo: 1
fk_empresa_asociada required	integer <int64> [id_empresa_asociada] por ejemplo: 1
fk_proveedor required	integer <int64> [id_proveedor] por ejemplo: 1
fk_articulo required	Array of integers [id_articulo] por ejemplo: 1

REQUEST BODY SCHEMA: application/json

POST /admin\_pedidos/crear\_pedido/ v

### Request samples

Payload

Content type  
application/json

Copy Expand all Collapse all

{ }

### Response samples

200 400

Content type  
application/json

Copy Expand all Collapse all

```
{
  "id_detalle_pedido": 0,
  - "fk_cliente": {
    "id_cliente": 0,
    "nombre": "string",
    "codigo": "string",
    "fotografia": "string",
    "direccion": "string",
    "fk_tipo_cliente": 0
  },
  - "fk_pedido": {
    "id_pedido": 0,
    "fecha_gen_pedido": "2019-08-24",
```

2

— — —

+ New

Import

Runner

My Workspace

Invite

Upgrade

adm

×

History

Collections

APIs

☐ Save Responses

Clear all

▼ Today

GET http://127.0.0.1:8000/api/admin\_pedidos/listar\_urgente/

▼ Yesterday

POST http://127.0.0.1:8000/api/admin\_pedidos/crear\_pedido/

POST http://127.0.0.1:8000/api/admin\_pedidos/crear\_pedido/

POST http://127.0.0.1:8000/api/admin\_pedidos/crear\_pedido/

POST http://127.0.0.1:8000/api/admin\_pedidos/listar\_urgente/

PUT http://127.0.0.1:8000/api/admin\_pedidos/listar\_urgente/

GET http://127.0.0.1:8000/api/admin\_pedidos/listar\_urgente/

POST http://127.0.0.1:8000/api/admin\_pedidos/listar\_urgente/

GET http://127.0.0.1:8000/api/admin\_pedidos/listar\_urgente/

GET http://127.0.0.1:8000/api/admin\_pedidos/listar\_urgente/

+

...

No Environment

Untitled Request

BUILD

GET

http://127.0.0.1:8000/api/admin\_pedidos/listar\_urgente/

Send

Save

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Code

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body

Cookies

Headers (9)

Test Results

Status: 200 OK

Time: 2.56 s

Size: 706 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   "id_pedido": 11,
3   "fecha_gen_pedido": "2021-07-11",
4   "hora_gen_pedido": "00:02:06",
5   "fecha_surte_pedido": null,
6   "hora_surte_pedido": null,
7   "es_urgente": true,
8   "fk_cliente": {
9     "id_cliente": 2,
10    "nombre": "CECILIA MARTINEZ",
11    "codigo": "CECI9109",
12    "fotografia": "---",
13    "direccion": "SALTILLO - AVANCE",
14    "fk_tipo_cliente": 4
15  },
16 }
```

Find and Replace

Console

Bootcamp

Build

Browse



# Mobilender\_technical\_test

---

4. Exponer cada servicio del punto anterior como un Web Service tipo REST utilizando las herramientas ofrecidas por Django.

Los datos para la solicitud y respuesta deben ser tipo JSON

Selecciona el tipo de método que consideres adecuado (GET, POST, PUT, PATCH, etc.) - Proponer las urls bajo las que estarían publicados los servicios.

views.py urls.py settings.py models.py Extension: Snapcode test\_commands.py test\_models.py

app > core > urls.py > ...

```
1 from django.urls import path
2 from . import views
3
4 urlpatterns = []
5     path('tipo_cliente/', views.tipo_clienteView.as_view()),
6     path('tipo_cliente/<int:id>', views.tipo_clienteView.as_view()),
7     path('cliente/', views.clienteView.as_view()),
8     path('cliente/<int:id>', views.clienteView.as_view()),
9     path('articulo/', views.ArticuloView.as_view()),
10    path('articulo/<int:id>', views.ArticuloView.as_view()),
11    path('proveedor/', views.ProveedorView.as_view()),
12    path('proveedor/<int:id>', views.ProveedorView.as_view()),
13    path('empresa_asociada/', views.Empresa_asociadaView.as_view()),
14    path('empresa_asociada/<int:id>', views.Empresa_asociadaView.as_view()),
15    path('sucursal/', views.SucursalView.as_view()),
16    path('sucursal/<int:id>', views.SucursalView.as_view()),
17    path('centro_distribucion/', views.Centro_distribucionView.as_view()),
18    path('centro_distribucion/<int:id>', views.Centro_distribucionView.as_view()),
19    path('pedido/', views.PedidoView.as_view()),
20    path('pedido/<int:id>', views.PedidoView.as_view()),
21    path('detalle_pedido/', views.Detalle_PedidoView.as_view()),
22    path('detalle_pedido/<int:id>', views.Detalle_PedidoView.as_view()),
23    path('proveedor_tiene_articulo/<int:id>', views.Proveedor_tiene_articuloView.as_view()),
24    path('proveedor_tiene_articulo/', views.Proveedor_tiene_articuloView.as_view()),
25    path('articulo_en_proveedor/', views.Articulo_en_proveedorView.as_view()),
26    path('articulo_en_proveedor/<int:id>', views.Articulo_en_proveedorView.as_view()),
27    path('admin_pedidos/crear_pedido/', views.Crear_Nuevo_PedidoViewSet.as_view({'post': 'create'})),
28    path('admin_pedidos/listar_urgente/', views.Listar_Pedido_UrgenteViewSet.as_view()),
29
30
```

< main Python 3.8.10 64-bit ('env': venv) 0 0 21 python | urls.py Ln 29, Col 2 Spaces: 4 UTF-8 LF Py

NewImportRunner

My Workspace

Invite

RefreshShareSettingsNotificationsHeartFire

Upgrade

cliente

HistoryCollectionsAPIs

Save ResponsesClear all

POST http://127.0.0.1:8000/api/tipo\_cliente

GET http://127.0.0.1:8000/api/tipo\_cliente

POST http://127.0.0.1:8000/api/tipo\_cliente

GET http://127.0.0.1:8000/api/tipo\_cliente

GET http://127.0.0.1:8000/tipo\_cliente

POST http://127.0.0.1:8000/api/cliente/

July 8

GET http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

PUT http://127.0.0.1:8000/api/cliente/

GET http://127.0.0.1:8000/api/cliente/

POST http://127.0.0.1:8000/api/cliente/

+

...

No Environment

BUILD

Send

Save

ParamsAuthorizationHeaders (8)BodyPre-request ScriptTestsSettingsCookiesCode

noneform-datax-www-form-urlencodedrawbinaryGraphQLJSON

1

2

3

4

5

6

7

"nombre": "JORGE ANGEL BLANQUICETT",

"codigo": "BEBE\_01",

"fotografia": "---",

"direccion": "Calle las manzanas No. 123",

"fk\_tipo\_cliente": 2

BodyCookiesHeaders (9)Test Results

PrettyRawPreviewVisualizeJSON

1

2

3

4

5

6

7

8

9

10

11

"id\_cliente": 3,

"nombre": "JORGE ANGEL BLANQUICETT",

"codigo": "BEBE\_01",

"fotografia": "---",

"direccion": "Calle las manzanas No. 123",

"fk\_tipo\_cliente": {

"id\_tipo\_cliente": 2,

"tipo\_cliente": "PLATA"

}

Find and ReplaceConsole

BootcampBuildBrowse

Help

+ New

Import

Runner

My Workspace

Invite

Upgrade

articulo

x

History

Collections

APIs

Save Responses

Clear all

POST http://127.0.0.1:8000/api/proveedor\_tiene\_articulo

POST http://127.0.0.1:8000/api/proveedor\_tiene\_articulo

POST http://127.0.0.1:8000/api/articulo\_en\_proveedor/

POST http://127.0.0.1:8000/api/articulo\_en\_proveedor/

POST http://127.0.0.1:8000/api/articulo\_en\_proveedor/

POST http://127.0.0.1:8000/api/articulo\_en\_proveedor/

POST http://127.0.0.1:8000/api/articulo\_en\_proveedor/

POST http://127.0.0.1:8000/api/articulo\_en\_proveedor/

POST http://127.0.0.1:8000/api/articulo\_en\_proveedor/

POST http://127.0.0.1:8000/api/articulo\_en\_proveedor/

POST http://127.0.0.1:8000/api/articulo\_en\_proveedor/

POST http://127.0.0.1:8000/api/proveedor\_tiene\_articulo

PUT http://127.0.0.1:8000/api/proveedor\_tiene\_articulo

GET http://127.0.0.1:8000/api/articulo/

GET http://127.0.0.1:8000/api/articulo/

GET http://127.0.0.1:8000/api/articulo/

+

...

No Environment

Untitled Request

BUILD

GET

http://127.0.0.1:8000/api/articulo/

Send

Save

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Code

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

Beautify

1

2

3

4

5

"codigo": "QBE.12221",

"descripcion": "LAPTOP HP 15 CORE I3",

"precio": 6625.1

Body

Cookies

Headers (9)

Test Results

Status: 200 OK

Time: 65 ms

Size: 990 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

3

4

5

6

7

8

9

10

11

12

13

14

15

16

"id\_articulo": 1,

"codigo": "wll\_k1123",

"descripcion": "Dell latitude 2021",

"precio": 19250.51

},

{

"id\_articulo": 2,

"codigo": "13212AA",

"descripcion": "TV ROKU",

"precio": 33250.9

},

{

"id\_articulo": 3,

"codigo": "TTT12323",

Find and Replace

Console

Bootcamp


Build

Browse

# Mobilender\_technical\_test

---

5. Realizar las pruebas unitarias que consideres necesarias para asegurar el correcto funcionamiento.



```
from unittest.mock import patch
```

```
from django.core.management import call_command
from django.db.utils import OperationalError
from django.test import TestCase
```

```
class CommandsTestCase(TestCase):
```

```
    def test_wait_for_db_ready(self):
        """Test waiting for db when db is available"""
```

```
        with patch('django.db.utils.ConnectionHandler.__getitem__') as gi:
            gi.return_value = True
            call_command('wait_for_db')
            self.assertEqual(gi.call_count, 1)
```

```
    @patch('time.sleep', return_value=None)
```

```
    def test_wait_for_db(self, ts):
        """Test waiting for db"""
```

```
        with patch('django.db.utils.ConnectionHandler.__getitem__') as gi:
            gi.side_effect = [OperationalError] * 5 + [True]
            call_command('wait_for_db')
            self.assertEqual(gi.call_count, 6)
```

# Mobilender\_technical\_test

---

6. proponer e incluir un webservice que exponga la información para un tablero informativo (Dashboard).

R: No entendi muy bien la pregunta, pero a mi entender se puede exponer la informacion en Heroku, EC2 (AWS)

# Mobilender\_technical\_test

---

7 . proponer Frontend para mostrar la información en el tablero informativo.

-proponer diseño

R: Propongo un dashboard de bootstrap 5



Dashboard

Orders

Products

Customers

Reports

Integrations

SAVED REPORTS

Current month

Last quarter

Social engagement

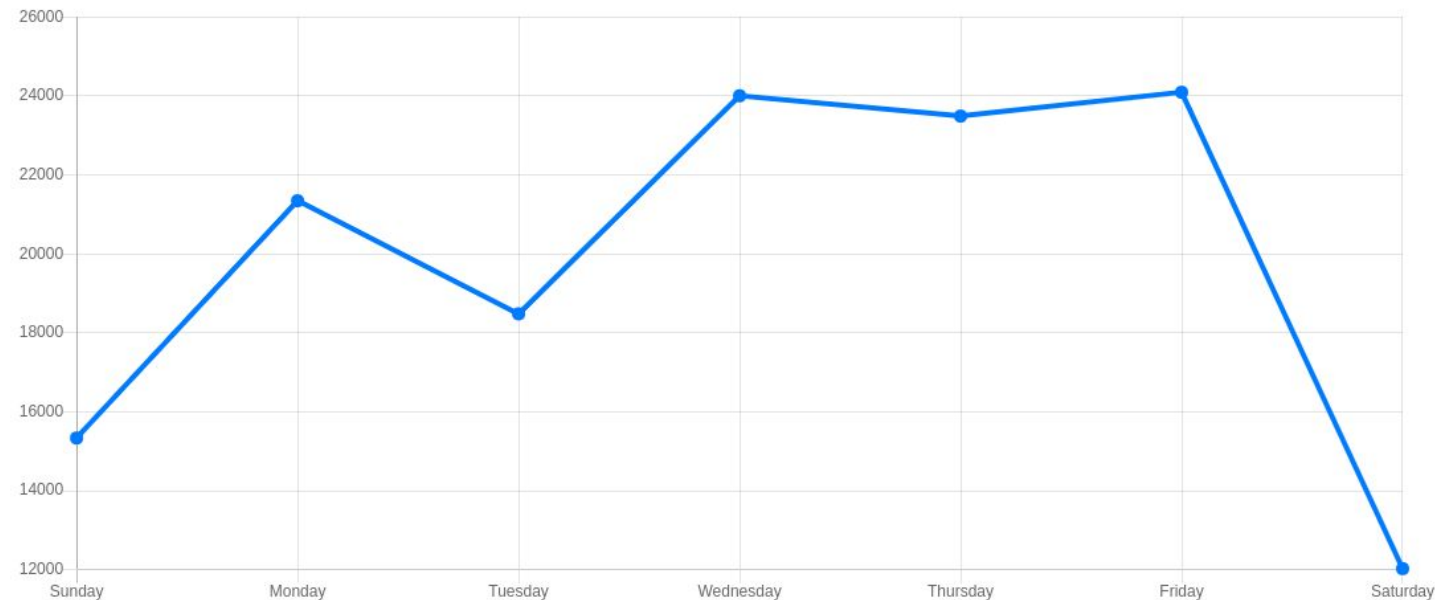
Year-end sale

# Dashboard

Share

Export

This week



## Section title

#	Header	Header	Header	Header
1,001	random	data	placeholder	text
1,002	placeholder	irrelevant	visual	layout

# Mobilender\_technical\_test

---

8. incluir documentación swagger para cada servicio.

# Mobilender Resfull API v0.1

[ Base URL: 127.0.0.1:8000/api ]

<http://127.0.0.1:8000/swagger/?format=openapi>

Documentacion publica mobilender

[Terms of service](#)

[Contact the developer](#)

[BSD License](#)

Schemes

HTTP

Django

admin

Django Logout

Authorize



Filter by tag

## admin\_pedidos



POST

/admin\_pedidos/crear\_pedido/

admin\_pedidos\_crear\_pedido\_create



GET

/admin\_pedidos/listar\_urgente/

admin\_pedidos\_listar\_urgente\_list



## articulo



# Mobilender\_technical\_test

---

9. Crear archivo README con las especificaciones del sistema y con las instrucciones para poder realizar la ejecución.

# MOBILENDER PEDIDOS

\_PRUEBA DE PROGRAMACIÓN

## Inicio 🚀

*Backend de sistema de pedidos recibidos desde clientes.*

Mira **Deployment** para conocer como desplegar el proyecto.

## Pre-requisitos 📖

\_Pyhton, Django, Postgres, docker, git

```
sudo apt-get update
sudo apt install python3
sudo apt install python3-django
sudo apt install postgresql postgresql-contrib
sudo apt install pgadmin4
sudo apt install docker-ce docker-ce-cli containerd.io
sudo apt install git
```

Django Web Framework `django==3.2.5`

Django Cors Headers - Used to enable CORS headers in API responses, and allow requests to be made to your API server from other origins. `django-cors-headers==3.5.0`

Django Rest Framework - Api Logic `django-rest-framework==3.12.4`

## Languages

● Python 99.1% ● Dockerfile 0.9%

# Mobilender\_technical\_test

---

10. Crear archivo docker-compose con las imágenes, servicios y lo que se considere necesario para su ejecución.

```
version: "3"

services:
  app:
    build:
      context: .
    ports:
      - "8000:8000"
    volumes:
      - ./app:/app
    command: >
      sh -c "python manage.py wait_for_
db &&
python manage.py migrate &
python manage.py runserver
0.0.0.0:8000"
    environment:
      - DB_HOST=db
      - DB_NAME=app
      - DB_USER=postgres
      - DB_PASS=supersecretpassword
    depends_on:
      - db





  db:
    image: postgres:10-alpine
    environment:
      - POSTGRES_DB=app
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=supersecretpa
ssword

  pgadmin:
    image: dpage/pgadmin4
    depends_on:
      - db
    ports:
      - "8889:80"

    environment:
      PGADMIN_DEFAULT_EMAIL
: admin@example.com
      PGADMIN_DEFAULT_PASSWORD: root
```

- FTS Templates
- Foreign Tables
- Functions
- Materialized Views
- 1.3 Sequences
- Tables (22)
  - auth\_group
  - auth\_group\_permissions
  - auth\_permission
  - auth\_user
  - auth\_user\_groups
  - auth\_user\_user\_permissions
  - core\_articulo
  - core\_centro\_distribucion
  - core\_cliente
  - core\_detalle\_pedido
  - core\_detalle\_pedido\_fk\_articulo
  - core\_detalle\_pedido\_fk\_proveedor
  - core\_empresa\_asociada
  - core\_pedido
  - core\_pedido\_fk\_empresa\_asociada
  - core\_proveedor
  - core\_sucursal
  - core\_tipo\_cliente
  - django\_admin\_log
  - django\_content\_type
  - django\_migrations
  - django\_session
- Trigger Functions
- Types
- Views
- Subscriptions

```
1 SELECT id_articulo, codigo, descripcion, precio
2 FROM public.core_articulo;
```

	 Id_articulo [PK] integer	 codigo character varying (45)	 descripcion character varying (255)	 precio double precision
1	1	wl_k1123	Dell latitude 2021	19250.51
2	2	13212AA	TV ROKU	33250.9
3	3	TTT12323	LAPTOP	12950.31
4	4	Xiaomi-9987	Xiaomi-mi10T	9899.31
5	5	Xiaomi-3361	Xiaomi-redmi2	9899.31
6	6	QBE.12221	LAPTOP HP 15 CORE I3	6625.1
7	7	QBE.12221	LAPTOP HP 15 CORE I3	6625.1
8	8	QBE.12221	LAPTOP HP 15 CORE I3	6625.1



# Mobilender\_technical\_test

---

Muchas gracias por permitirme desarrollar esta prueba, fue retadora, interesante y sobre todo muy enriquecedora.

Contacto:

[jorgeabm1992@gmail.com](mailto:jorgeabm1992@gmail.com)

+52 5580158148