Yeimy Tatiana Bayona Pérez

Servicio Nacional De Aprendizaje

Centro Minero

Análisis y desarrollo de software (ADSO)

Ficha:2993648

Andrés Felipe Sandoval Higuera

Sogamoso, Boyacá

Julio 2025

Tabla de contenido

INTRODUCCIÓN 3

Base de Datos SQLite – Django 4

- **Inserción de Datos** 4
- **Actualización de Datos** 5
  1. **Primer Dato** 5
  2. **Segundo Dato** 6
  3. **Tercer Dato** 7
- **Eliminar Datos** 9
  1. **Primer Dato** 9
  2. **Segundo Dato** 9
  3. **Tercer Dato** 10
- **Agregar dos columnas** 11
  1. **Columnas de teléfono y fecha** 11
- **Ejercicio de agregar 3 columnas mas** 13
- **Agregar los datos a la Base de Datos** 14
  1. **Primer registro** 14
  2. **Segundo registro** 14
  3. **Tercer registro** 15
- **Agregar 3 personas mas con todos los campos** 15

CONCLUSIONES 17

INTRODUCCIÓN

En el presente trabajo se expone el desarrollo de os conceptos fundamentales relacionados con la **gestión de base de datos** utilizando el Framework **Django** y el motor **SQLite**. A lo largo del documento se evidencia, mediante capturas de pantalla y explicaciones detalladas, el manejo de los comandos básicos trabajados en clase y la interacción directa con la base de datos desde la consola de **Python.**

Se abordan procesos esenciales como la inserción, consulta, actualización y eliminación de datos, aplicados dentro del entorno de Django. Además, se presenta un ejercicio integrador en el que se ponen en práctica los conocimientos adquiridos, utilizando múltiples comandos y funcionalidades propias del **ORM** de Django.

Este trabajo permite demostrar el entendimiento adecuado del funcionamiento de las bases de datos en Django, así como el correcto uso de su consola interactiva. A su vez, refleja el avance en la formación tecnológica, fortaleciendo habilidades prácticas en programación, manipulación de datos y desarrollo backend con Python.

Base de Datos SQLite – Django

- **Inserción de Datos**

```
>>> from members.models import Member
>>> Member.objects.all()
<QuerySet []>
>>> member = Member(firstname = 'Tatiana', lastname = 'Bayona')
>>> member.save()
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Tatiana', 'lastname': 'Bayona'}]>
>>> member1 = Member(firstname='Kamila',lastname='Vargas')
>>> member2 = Member(firstname='Andres',lastname='Ardila')
>>> member3 = Member(firstname='Jimy',lastname='Angarita')
>>> member4 = Member(firstname='Mayerly',lastname='Cardenas')
>>> member5 = Member(firstname='Elyan',lastname='Cristancho')
>>> members_list=[member1,member2,member3,member4,member5]
>>> for x in member_list:
    x.save()
```

Se hizo la inserción de datos, por medio de la consola de python en la cual definimos variables, para insertar los datos en la base de datos, se guardan en una lista para poder guardarlos con un ciclo for, en el cual se define una variable x para hacer la inserción de todos los datos.

	id	firstname	lastname
	Fil...	Filtro	Filtro
1	1	Tatiana	Bayona
2	2	Kamila	Vargas
3	3	Andres	Ardila
4	4	Jimy	Angarita
5	5	Mayerly	Cardenas
6	6	Elyan	Cristancho

Se muestran las consultas de la base de datos por medio del programa SQLite, en el cual se ven todos los datos que se insertaron.

- **Actualización de Datos**

1. **Primer Dato**

- Firstname

```
>>> x=Member.objects.all()[4]
>>> x.firstname
'Mayerly'
>>> x.firstname='Nubia'
>>> x.save()
```

En este comando se ve como se actualizan los datos de la base de datos, por medio de consola de Python y ase actualiza el nombre.

	id	firstname	lastname
	Fil...	Filter	Filter
1	1	Tatiana	Bayona
2	2	Kamila	Vargas
3	3	Andres	Ardila
4	4	Jimy	Angarita
5	5	Nubia	Cardenas
6	6	Elyan	Cristancho

- Lastname

```
>>> x=Member.objects.all()[4]
>>> x.lastname
'Cardenas'
>>> x.lastname = 'Bautista'
>>> x.save()
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Tatiana', 'lastname': 'Bayona'}, {'id': 2, 'firstname': 'Kamila', 'lastname': 'Vargas'}, {'id': 3, 'firstname':
'Andres', 'lastname': 'Ardila'}, {'id': 4, 'firstname': 'Jimy', 'lastname': 'Angarita'}, {'id': 5, 'firstname': 'Nubia', 'lastname': 'Bautista'},
'id': 6, 'firstname': 'Elyan', 'lastname': 'Cristancho'}]>
```

En este comando se ve como se actualizan los datos de la base de datos, por medio de consola de Python y ase actualiza el apellido.

	id	firstname	lastname
	Fil...	Filter	Filter
1	1	Tatiana	Bayona
2	2	Kamila	Vargas
3	3	Andres	Ardila
4	4	Jimy	Angarita
5	5	Nubia	Bautista
6	6	Elyan	Cristancho

2. **Segundo Dato**

```
>>> x=Member.objects.all()[0]
>>> x.firstname
'Tatiana'
>>> x.firstname='Yeimy'
>>> x.save()
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Yeimy', 'lastname': 'Bayona'}, {'id': 2, 'firstname': 'Kamila', 'lastname': 'Vargas'}, {'id': 3, 'firstname': 'A
ndres', 'lastname': 'Ardila'}, {'id': 4, 'firstname': 'Jimy', 'lastname': 'Angarita'}, {'id': 5, 'firstname': 'Nubia', 'lastname': 'Bautista'}, {'i
d': 6, 'firstname': 'Elyan', 'lastname': 'Cristancho'}]>
>>>
```

	id	firstname	lastname
	Fil...	Filter	Filter
1	1	Yeimy	Bayona
2	2	Kamila	Vargas
3	3	Andres	Ardila
4	4	Jimy	Angarita
5	5	Nubia	Bautista
6	6	Elyan	Cristancho

```
>>> x=Member.objects.all()[0]
>>> x.lastname = 'Pérez'
>>> x.save()
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Yeimy', 'lastname': 'Pérez'}, {'id': 2, 'firstname': 'Kamila', 'lastname': 'Fuentes'}, {'id': 3, 'firstname': 'A
ndres', 'lastname': 'Ardila'}, {'id': 4, 'firstname': 'Jimy', 'lastname': 'Angarita'}, {'id': 5, 'firstname': 'Nubia', 'lastname': 'Bautista'}, {'i
d': 6, 'firstname': 'Elyan', 'lastname': 'Cristancho'}]>
>>>
```

	id	firstname	lastname
	Fil...	Filter	Filter
1	1	Yeimy	Pérez
2	2	Kamila	Fuentes
3	3	Andres	Ardila
4	4	Jimy	Angarita
5	5	Nubia	Bautista
6	6	Elyan	Cristancho

3. Tercer Dato

```
>>> x=Member.objects.all()[1]
>>> x.firstname='Maria'
>>> x.save()
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Yeimy', 'lastname': 'Pérez'}, {'id': 2, 'firstname': 'Maria', 'lastname': 'Fuentes'}, {'id': 3, 'firstname': 'An
dres', 'lastname': 'Ardila'}, {'id': 4, 'firstname': 'Jimy', 'lastname': 'Angarita'}, {'id': 5, 'firstname': 'Nubia', 'lastname': 'Bautista'}, {'id
': 6, 'firstname': 'Elyan', 'lastname': 'Cristancho'}]>
```

	id	firstname	lastname
	Fil...	Filter	Filter
1	1	Yeimy	Pérez
2	2	Maria	Fuentes
3	3	Andres	Ardila
4	4	Jimy	Angarita
5	5	Nubia	Bautista
6	6	Elyan	Cristancho

```
>>> x=Member.objects.all()[1]
>>> x.lastname = 'Fuentes'
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Yeimy', 'lastname': 'Bayona'}, {'id': 2, 'firstname': 'Kamila', 'lastname': 'Vargas'}, {'id': 3, 'firstname': 'A
ndres', 'lastname': 'Ardila'}, {'id': 4, 'firstname': 'Jimy', 'lastname': 'Angarita'}, {'id': 5, 'firstname': 'Nubia', 'lastname': 'Bautista'}, {'i
d': 6, 'firstname': 'Elyan', 'lastname': 'Cristancho'}]>
>>> x.save()
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Yeimy', 'lastname': 'Bayona'}, {'id': 2, 'firstname': 'Kamila', 'lastname': 'Fuentes'}, {'id': 3, 'firstname': '
Andres', 'lastname': 'Ardila'}, {'id': 4, 'firstname': 'Jimy', 'lastname': 'Angarita'}, {'id': 5, 'firstname': 'Nubia', 'lastname': 'Bautista'}, {'
```

	id	firstname	lastname
	Fil...	Filter	Filter
1	1	Yeimy	Bayona
2	2	Kamila	Fuentes
3	3	Andres	Ardila
4	4	Jimy	Angarita
5	5	Nubia	Bautista
6	6	Elyan	Cristancho

➢ Eliminar Datos

1. Primer Dato

```
>>> x=Member.objects.all()[5]
>>> x.firstname
'Elyan'
>>> x.delete()
(1, {'members.Member': 1})
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Yeimy', 'lastname': 'Pérez'}, {'id': 2, 'firstname': 'Maria', 'lastname': 'Fuentes'}, {'id': 3, 'firstname': 'An
dres', 'lastname': 'Ardila'}, {'id': 4, 'firstname': 'Jimy', 'lastname': 'Angarita'}, {'id': 5, 'firstname': 'Nubia', 'lastname': 'Bautista'}]>
>>>
```

	id	firstname	lastname
	Fil...	Filter	Filter
1	1	Yeimy	Pérez
2	2	Maria	Fuentes
3	3	Andres	Ardila
4	4	Jimy	Angarita
5	5	Nubia	Bautista

2. Segundo Dato

```
>>> x=Member.objects.all()[3]
>>> x.firstname
'Jimy'
>>> x.delete()
(1, {'members.Member': 1})
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Yeimy', 'lastname': 'Pérez'}, {'id': 2, 'firstname': 'Maria', 'lastname': 'Fuentes'}, {'id': 3, 'firstname': 'An
dres', 'lastname': 'Ardila'}, {'id': 5, 'firstname': 'Nubia', 'lastname': 'Bautista'}]>
```

id	firstname	lastname
Fil...	Filter	Filter
1	Yeimy	Pérez
2	Maria	Fuentes
3	Andres	Ardila
5	Nubia	Bautista

3. **Tercer Dato**

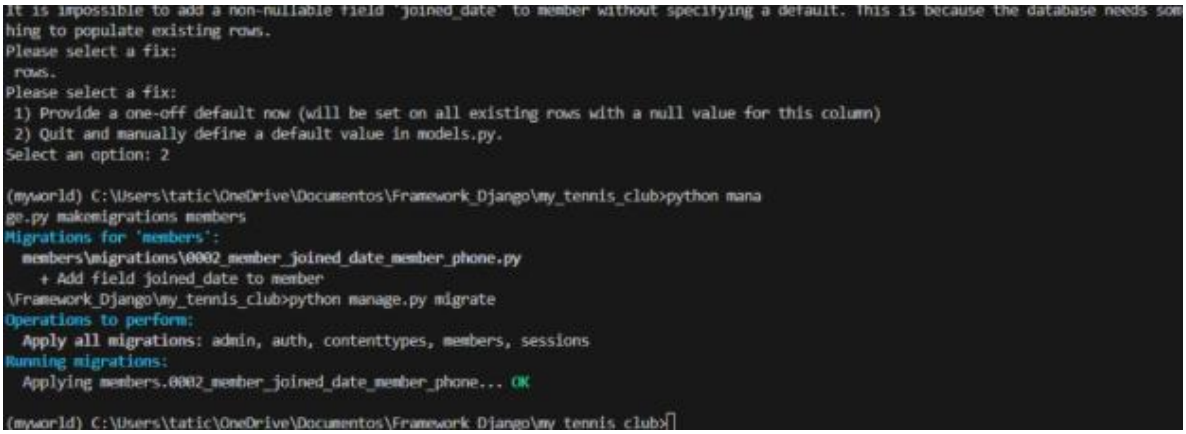
```
>>> x=Member.objects.all()[3]
>>> x.firstname
'Nubia'
>>> x.delete()
(1, {'members.Member': 1})
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Yeimy', 'lastname': 'Pérez'}, {'id': 2, 'firstname': 'Maria', 'lastname': 'Fuentes'}, {'id': 3, 'firstname': 'An
dres', 'lastname': 'Ardila'}]>
```

	id	firstname	lastname
	Fil...	Filter	Filter
1	1	Yeimy	Pérez
2	2	Maria	Fuentes
3	3	Andres	Ardila

Eliminamos los datos con el comando **x.delete()**, teniendo en cuenta que el procedimiento es igual a actualizar solo que se cambia el comando para poder eliminar el dato, y al final nos dará la información de que se elimino el dato de **members.Member** es decir la base de datos.

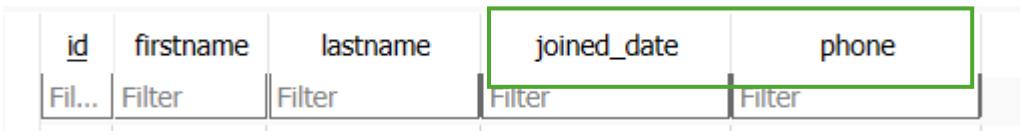
- **Agregar dos columnas**

1. **Columnas de teléfono y fecha**



Con el comando **Python manage.py makemigrations members** nos mandara un menú de opciones, ya que, al hacer la migración, el sistema observa que las columnas no existen y no sabe que información va dentro de ellas, al seleccionar la opción 2 nos dejara añadir las dos columnas vacías, en las cuales podemos insertar luego los datos, aquí mismo volvemos a ejecutar el mismo comando y ya se añaden las migraciones correctamente en members.

Con el comando **Python manage.py migrate** se aplican todas las migraciones correctamente en la base de datos.



Y ya podemos observar que quedan añadidas las dos columnas

1. **Agregar datos**

```
>>> x = Member.objects.all()[0]
>>> x.phone=321223
>>> x.save()
>>> x.joined_date='2025-11-13'
>>> x.save()
```

Con el comando **x=Member.objects.all()[0]** estamos llamando en la posición que queremos agregar el nuevo dato de las dos columnas que hacen falta

	id	firstname	lastname	joined_date	phone
	Fil...	Filter	Filter	Filter	Filter
1	1	Yeimy	Pérez	2025-11-13	321223
2	2	Maria	Fuentes	NULL	NULL
3	3	Andres	Ardila	NULL	NULL

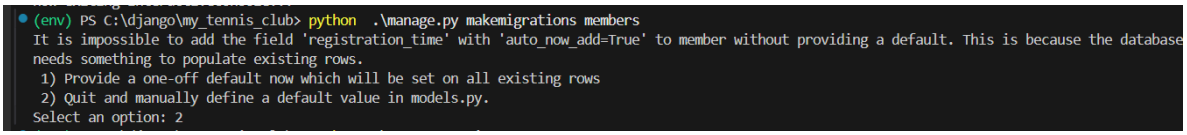
Y podemos observar que quedan los dos datos que hacían falta.

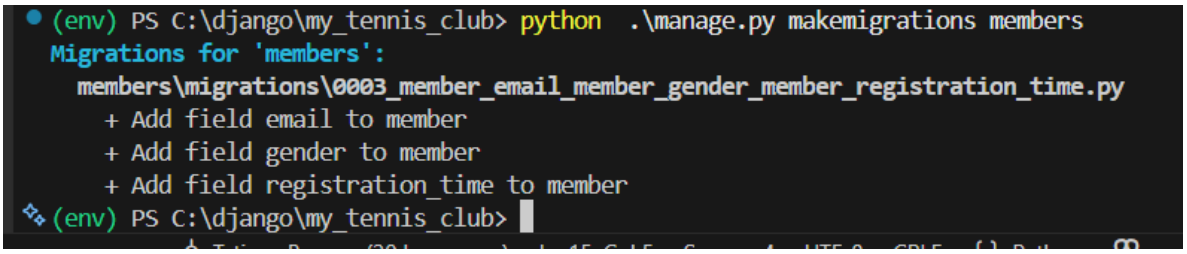
- **Se añaden los datos restantes**

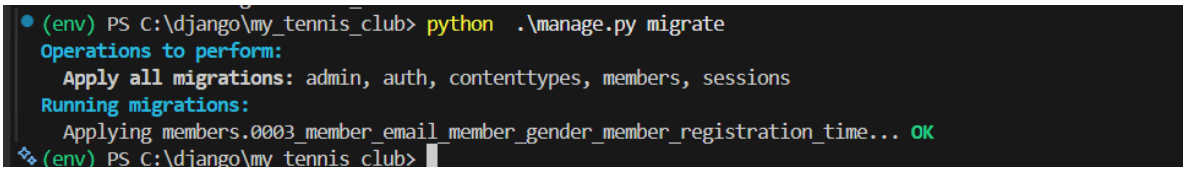
```
>>> x = Member.objects.all()[1]
>>> x.phone=321212
>>> x.joined_date='2025-11-13'
>>> x.save()
>>> x = Member.objects.all()[2]
>>> x.phone=3212343
>>> x.joined_date='2025-11-13'
>>> x.save()
>>>
```

	id	firstname	lastname	joined_date	phone
	Fil...	Filter	Filter	Filter	Filter
1	1	Yeimy	Pérez	2025-11-13	321223
2	2	Maria	Fuentes	2025-11-13	321212
3	3	Andres	Ardila	2025-11-13	3212343

- **Ejercicio de agregar 3 columnas mas**







Aquí se agregan otras 3 columnas que son género, email y la hora de registro, con el comando **Python manage.py makemigations members** se hacen las migraciones necesarias y con **Python manage.py migrate** se aplican todas las migraciones a la base de datos de members.

	id	firstname	lastname	joined_date	phone		email	gender	registration_time
	Fil...	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	Yeimy	Pérez	2025-11-13	321223		NULL	NULL	0
2	2	Maria	Fuentes	2025-11-13	321212		NULL	NULL	0
3	3	Andres	Ardila	2025-11-13	3212343		NULL	NULL	

Y podemos observar como se agregaron las columnas correctamente a la base de datos.

- **Agregar los datos a la Base de Datos**

1. **Primer registro**

```
>>> from members.models import Member
>>> x=Member.objects.all().[0]
```

```
>>> x.email='yeimy@gmail.com'
>>> x.save()
>>> x=Member.objects.all()[0]
>>> x.gender ='femenino'
>>> x.save()
```

```
>>> x.registration_time ='2025-11-14 8:17:40'
>>> x.save()
```

	id	firstname	lastname	joined_date	phone	email	gender	registration_time
	Fil...	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	Yeimy	Pérez	2025-11-13	321223	yeimy@gmail.com	femenino	2025-11-14 08:17:40

Se agregaron datos correctamente de a primera persona en la base de datos, gracias a los comandos vistos anteriormente, y teniendo en cuenta las comillas, para las fechas para evitar errores, gracias al from y la variable se pueden trabajar los datos.

2. **Segundo registro**

```
>>> x=Member.objects.all()[1]
>>> x.email='maria@gmail.com'
>>> x.gender='femenino'
>>> x.registration_time='2025-11-14 8:22:50'
>>> x.save()
```

Table: members_member

Filter in any column

	id	firstname	lastname	joined_date	phone	email	gender	registration_time
	Fil...	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	Yeimy	Pérez	2025-11-13	321223	yeimy@gmail.com	femenino	2025-11-14 08:17:40
2	2	Maria	Fuentes	2025-11-13	321212	maria@gmail.com	femenino	2025-11-14 08:22:50

Se agregaron los datos de la segunda persona correctamente, teniendo en cuenta los comandos anteriormente usados, y cuidando que toda la digitación sea correcta en la consola de Python.

3. Tercer registro

```
>>> x=Member.objects.all()[2]
>>> x.email='andres@gmail.com'
>>> x.gender='masculino'
>>> x.registration_time='2025-11-14 9:22:50'
>>> x.save()
```

Table: members_member

Filter in any column

	id	firstname	lastname	joined_date	phone	email	gender	registration_time
	Fil...	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	Yeimy	Pérez	2025-11-13	321223	yeimy@gmail.com	femenino	2025-11-14 08:17:40
2	2	Maria	Fuentes	2025-11-13	321212	maria@gmail.com	femenino	2025-11-14 08:22:50
3	3	Andres	Ardila	2025-11-13	3212343	andres@gmail.com	masculino	2025-11-14 09:22:50

Se agregaron los datos de la tercera persona correctamente, teniendo en cuenta los comandos anteriormente usados, y cuidando que toda la digitación sea correcta en la consola de Python.

- **Agregar 3 personas mas con todos los campos**

```
>>> from members.models import Member
>>> Member.objects.all().values()
<QuerySet [{'id': 1, 'firstname': 'Yeimy', 'lastname': 'Pérez', 'phone': 321223, 'joined_date': datetime.date(2025, 11, 13), 'email': 'yeimy@gmail.
com', 'gender': 'femenino', 'registration_time': datetime.datetime(2025, 11, 14, 8, 17, 40, tzinfo=datetime.timezone.utc)}, {'id': 2, 'firstname':
'Maria', 'lastname': 'Fuentes', 'phone': 321212, 'joined_date': datetime.date(2025, 11, 13), 'email': 'maria@gmail.com', 'gender': 'femenino', 'reg
istration_time': datetime.datetime(2025, 11, 14, 8, 22, 50, tzinfo=datetime.timezone.utc)}, {'id': 3, 'firstname': 'Andres', 'lastname': 'Ardila',
'phone': 3212343, 'joined_date': datetime.date(2025, 11, 13), 'email': 'andres@gmail.com', 'gender': 'masculino', 'registration_time': datetime.dat
etime(2025, 11, 14, 9, 22, 50, tzinfo=datetime.timezone.utc)}]>
```

Con el comando **Member.objects.all().values()** llamamos a todos los campos para poder agregarlos nuevos registros

```
>>> member1=Member(firstname= 'Isabella',lastname= 'Torres',phone= 3212155, joined_date= '2025-10-12', email='isabella@gmail.com',gender='femenino'
)
>>> member2=Member(firstname= 'Camilo',lastname= 'Fonseca',phone= 3252155, joined_date= '2025-09-02', email='camilo@gmail.com',gender='masculino',r
egistration_time='2025-11-14 12:12:01')
>>> member3=Member(firstname= 'Felipe',lastname= 'Perez',phone= 32998155, joined_date= '2025-01-12', email='felipe@gmail.com',gender='masculino',re
gistration_time='2025-11-14 9:12:01')
>>> memebers_list=[member1,member2,member3]
>>> for x in memebers_list:
...     x.save()
```

Definimos las variables que se van agregar con todos sus datos, se guardan en una lista para poder guardarlos con un ciclo for.

Table: members_member

Filter in any column

	id	firstname	lastname	joined_date	phone	email	gender	registration_time
	Fil...	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	Yeimy	Pérez	2025-11-13	321223	yeimy@gmail.com	femenino	2025-11-14 08:17:40
2	2	Maria	Fuentes	2025-11-13	321212	maria@gmail.com	femenino	2025-11-14 08:22:50
3	3	Andres	Ardila	2025-11-13	3212343	andres@gmail.com	masculino	2025-11-14 09:22:50
4	4	Isabella	Torres	2025-10-12	3212155	isabella@gmail.c…	femenino	2025-11-14 10:13:03
5	5	Camilo	Fonseca	2025-09-02	3252155	camilo@gmail.com	masculino	2025-11-14 12:12:01
6	6	Felipe	Perez	2025-01-12	32998155	felipe@gmail.com	masculino	2025-11-14 09:12:01

Y podemos observar que se agregan correctamente los tres datos a la base datos con todos sus campos.

CONCLUSIONES

- Se evidenció un sólido manejo de las operaciones fundamentales de bases de datos **(CRUD)** en Django, utilizando SQLite para la gestión de la información del proyecto.

- El desarrollo de este trabajo permitió comprender y aplicar de manera práctica el funcionamiento del **ORM** de **Django** y su integración con el motor de base de datos **SQLite.**

- Las actividades realizadas contribuyeron al fortalecimiento de competencias técnicas necesarias para el desarrollo backend, permitiendo acercarme a una experiencia real de interacción con bases de datos mediante código.