Informe de Laboratorio 7

Tema: DJANGO III

Nota

Estudiante	Escuela	Asignatura
Sergio Hancco Mullisaca shanccom@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programacion Web 2 Semestre: II Código:

Laboratorio	Tema	Duración
7	DJANGO III	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 09 Mayo 2024	Al 14 Junio 2024

1. Tarea

- Informe de laboratorio
- Video en Flip
- Ejercicios Propuestos

2. Equipos, materiales y temas utilizados

- VS
- Git 2.39.2.
- Cuenta en GitHub con el correo institucional.

3. URL de Repositorio Github

- URL del video en yt.
- `https://youtu.be/dkZBugjlI2s`
- URL del video en flip.
- `https://flip.com/s/kj5APzDKgEHZ`
- URL del GITHUB.
- `https://github.com/shanccom/Programacion_Web_2.git`

4. Actividades

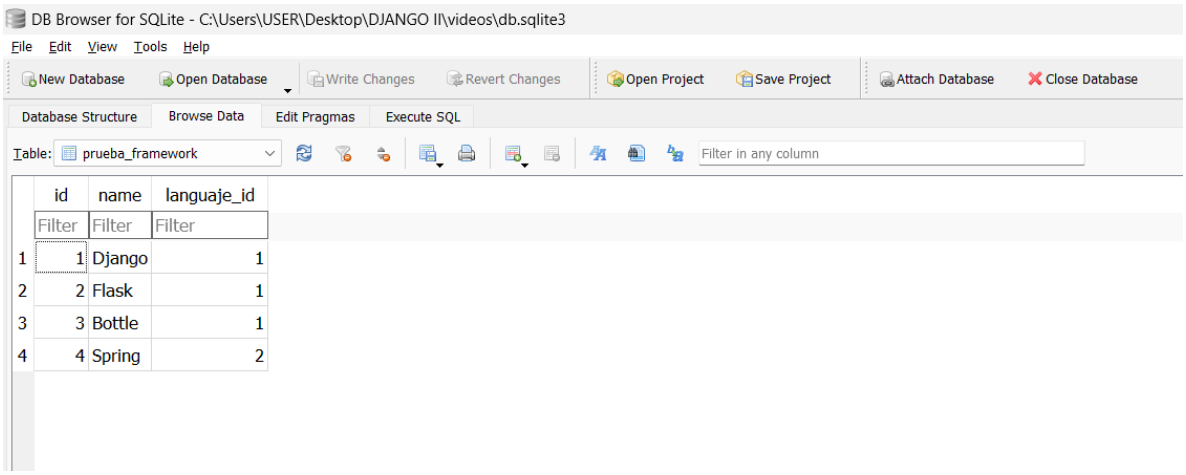
4.1. EJERCICIOS PROPUESTOS

VIDEOS (1-4): CREACION DE CLASES EN MODELS

Listing 1: models.py

```python
from django.db import models

# Create your models here.

class Languaje(models.Model):
    name = models.CharField(max_length = 10)

    def __str__(self):
        return self.name

class FrameWork(models.Model):
    name = models.CharField(max_length = 10)
    languaje = models.ForeignKey(Languaje, on_delete = models.CASCADE)

    def __str__(self):
        return self.name

class Movie(models.Model):
    name = models.CharField(max_length = 10)

    def __str__(self):
        return self.name

class Character(models.Model):
    name = models.CharField(max_length = 10)
    movies = models.ManyToManyField(Movie)

    def __str__(self):
        return self.name
```

Evidencia de la generacion de framework



	id	name	languaje_id
	Filter	Filter	Filter
1	1	Django	1
2	2	Flask	1
3	3	Bottle	1
4	4	Spring	2

Uso de Shell para la creacion

```
>>> from prueba.models import Languaje, FrameWork
>>> python = Languaje(name = 'Python')
>>>
>>> python.save()
>>> django = FrameWork (name = 'Django')
>>> flask = FrameWork (name = 'Flask')
>>> python
<Languaje: Languaje object (1)>
>>> django.languaje = pythom
Traceback (most recent call last):
  File "<console>", line 1, in <module>
NameError: name 'pythom' is not defined. Did you mean: 'python'?
>>> django.languaje = python
>>> flask.languaje = python
>>> bottle = FrameWork(name = 'Bottle', languaje = python)
>>> django.save()
>>> flask.save()
>>> bottle.save()
>>>
>>>
>>> java = Languaje(name = 'Java')
>>> spring = FrameWork (name = 'Spring', languaje = java)
>>> java.save()
>>> spring = FrameWork (name = 'Spring', languaje = java)
>>> spring.save()
>>>
```

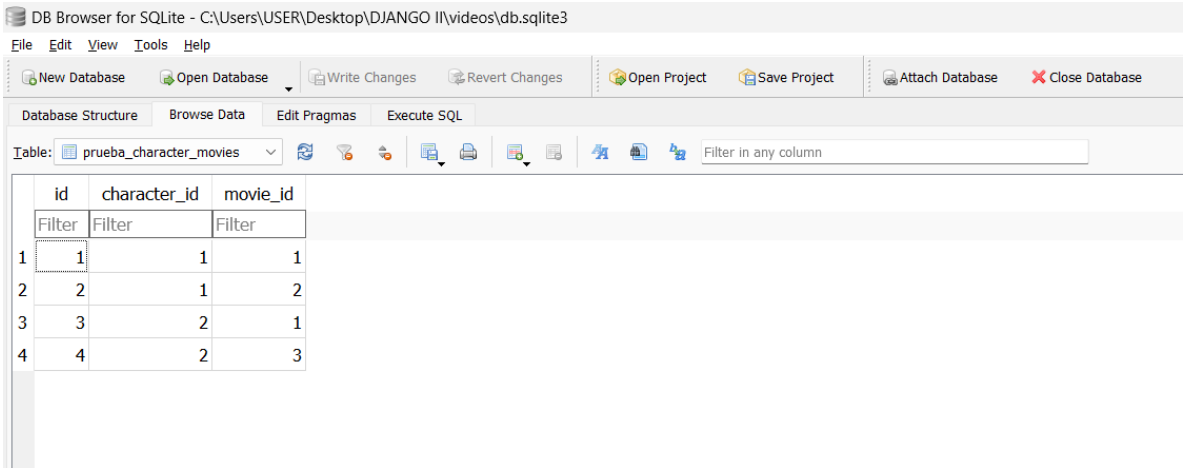
Uso de Shell (COMANDOS)

```
>>> from prueba.models import Languaje, FrameWork
>>> python = Languaje(name = 'Python')
>>>
>>> python.save()
>>> django = FrameWork (name = 'Django')
>>> flask = FrameWork (name = 'Flask')
>>> python
<Languaje: Languaje object (1)>
>>> django.languaje = pythom
Traceback (most recent call last):
  File "<console>", line 1, in <module>
NameError: name 'pythom' is not defined. Did you mean: 'python'?
>>> django.languaje = python
>>> flask.languaje = python
>>> bottle = FrameWork(name = 'Bottle', languaje = python)
>>> django.save()
>>> flask.save()
>>> bottle.save()
>>>
>>>
>>> java = Languaje(name = 'Java')
>>> spring = FrameWork (name = 'Spring', languaje = java)
>>> java.save()
>>> spring = FrameWork (name = 'Spring', languaje = java)
>>> spring.save()
>>>
```

```
Windows PowerShell
Python 3.12.3 (tags/v3.12.3:f6650f9, Apr  9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> exit()
(myvenv) PS C:\Users\USER\Desktop\DJANGO II\videos> python manage.py shell
Python 3.12.3 (tags/v3.12.3:f6650f9, Apr  9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from prueba.models import Languaje, FrameWork
>>> FrameWork.objects.all()
<QuerySet [<FrameWork: Django>, <FrameWork: Flask>, <FrameWork: Bottle>, <FrameWork: Spring>]>
>>> FrameWork.objects.filter(languaje__name = 'Python')
<QuerySet [<FrameWork: Django>, <FrameWork: Flask>, <FrameWork: Bottle>]>
>>> FrameWork.objects.filter(languaje__name = 'Java')
<QuerySet [<FrameWork: Spring>]>
>>> FrameWork.objects.filter(languaje__name__startswith = 'Py')
<QuerySet [<FrameWork: Django>, <FrameWork: Flask>, <FrameWork: Bottle>]>
>>> FrameWork.objects.filter(languaje__name__startswith = 'Pa')
<QuerySet []>
>>> Languaje.objects.filter(framework__name = 'Spring')
<QuerySet [<Languaje: Java>]>
>>> Languaje.objects.filter(framework__name = 'Bottle')
<QuerySet [<Languaje: Python>]>
>>>
```

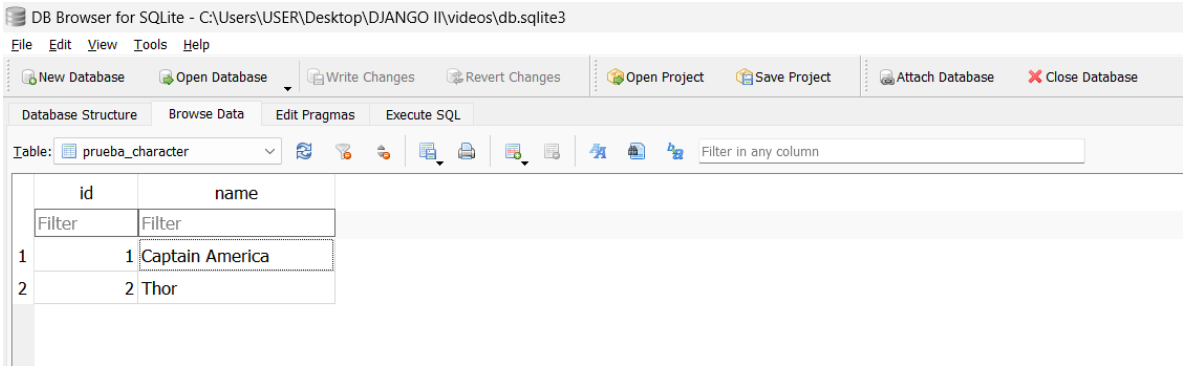
```
(myvenv) PS C:\Users\USER\Desktop\DJANGO II\videos> python manage.py makemigrations
Migrations for 'prueba':
  prueba\migrations\0002_movie_character.py
    - Create model Movie
    - Create model Character
(myvenv) PS C:\Users\USER\Desktop\DJANGO II\videos> python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, prueba, sessions
Running migrations:
  Applying prueba.0002_movie_character... OK
(myvenv) PS C:\Users\USER\Desktop\DJANGO II\videos> python .\manage.py shell
Python 3.12.3 (tags/v3.12.3:f6650f9, Apr  9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from prueba.models import Movie, Character
>>> avengers = Movie(name = 'Avengers')
>>> avengers.save()
>>> captain_america = Character(name = 'Captain America')
>>> captain_america.save()
>>> captain_america.movies.add(avengers)
>>> civil_war = Movie (name = 'Civil War')
>>> thor = Movie (name = 'Thor: Dark World')
>>> thor_character = Character(name = 'Thor')
>>> civil_war.save()
>>> thor.save()
>>> thor_character.save()
>>> captain_america.movies.add(civil_war)
>>> thor_chracter.movies.add(avengers)
Traceback (most recent call last):
  File "<console>", line 1, in <module>
NameError: name 'thor_chracter' is not defined. Did you mean: 'thor_character'?
>>> thor_character.movies.add(avengers)
>>> thor_character.movies.add(thor)
>>> captain_america.movies.create(name = 'Winter Soldier')
<Movie: Winter Soldier>
>>>
```

Muestra de la relacion character movies



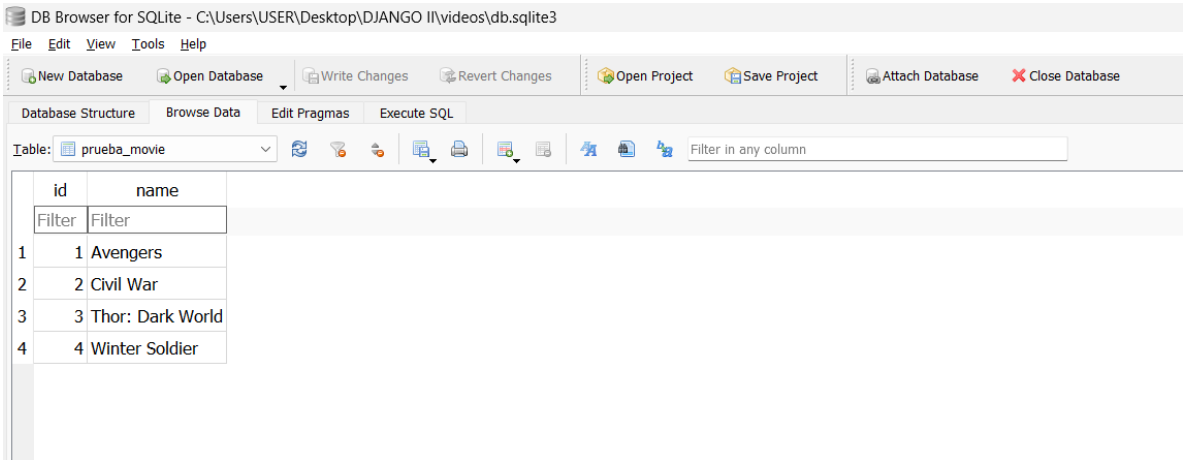
	id	character_id	movie_id
	Filter	Filter	Filter
1	1	1	1
2	2	1	2
3	3	2	1
4	4	2	3

Muestra character



	id	name
	Filter	Filter
1	1	Captain America
2	2	Thor

Muestra movie



	id	name
	Filter	Filter
1	1	Avengers
2	2	Civil War
3	3	Thor: Dark World
4	4	Winter Soldier

Metodo para que devuelva todas las peliculas

```
>>> captain_america = Character.objects.get(name = 'Captain America')

>>> captain_america
<Character: Captain America>
>>> captain_america.movies.all()
<QuerySet [<Movie: Avengers>, <Movie: Civil War>, <Movie: Winter Soldier>]>
>>> avengers = Movie.objects.get (name = 'Avengers')
>>> avengers
<Movie: Avengers>
>>> avengers.character_set.all()
<QuerySet [<Character: Captain America>, <Character: Thor>]>
>>>
```

VIDEOS 5: HTML A PDF

Metodos n views

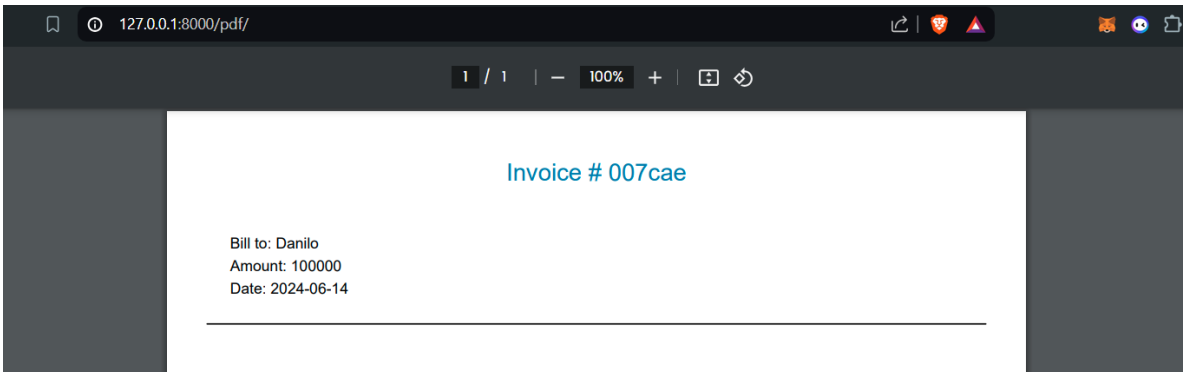
Listing 2: views.py

```python
import locale

from . import renderers


def invoice_view(request):
    context = {
        "bill_to": "Danilo",
        "invoice_number": "007cae",
        "amount": 100_000,
        "date": "2024-06-14",
    }
    return renderers.render_to_pdf("pdfs/invoice.html", context)


def advanced_pdf_view(request):
    locale.setlocale(locale.LC_ALL, "")
    invoice_number = "007cae"
    context = {
        "bill_to": "Sergio",
        "invoice_number": f"{invoice_number}",
        "amount": locale.currency(100_000, grouping=True),
        "date": "2024-23-10",
        "pdf_title": f"Invoice #{invoice_number}",
    }
    response = renderers.render_to_pdf("pdfs/invoice.html", context)
    if response.status_code == 404:
        raise HTTP404("Invoice not found")

    filename = f"Invoice_{invoice_number}.pdf"
    """
    Tell browser to view inline (default)
    """
```

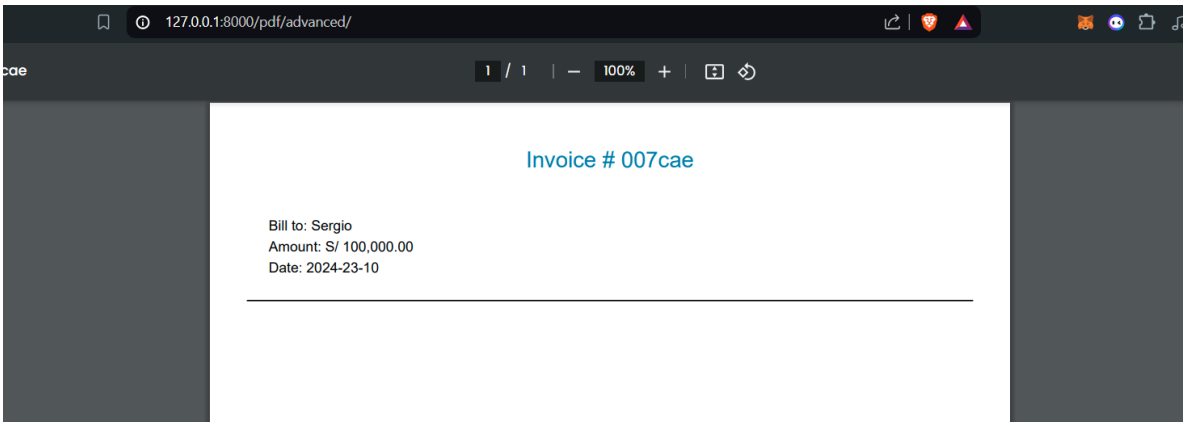
```
    content = f"inline; filename={filename}"
    download = request.GET.get("download")
    if download:
        """
        Tells browser to initiate download
        """
        content = f"attachment; filename={filename}"
    response["Content-Disposition"] = content
    return response
    return response
```

CAPTURAS DE EVIDENCIA

USO BASICO



USO AVANZADO



VIDEOS 6: ENVIO DE GMAIL

Listing 3: views.py

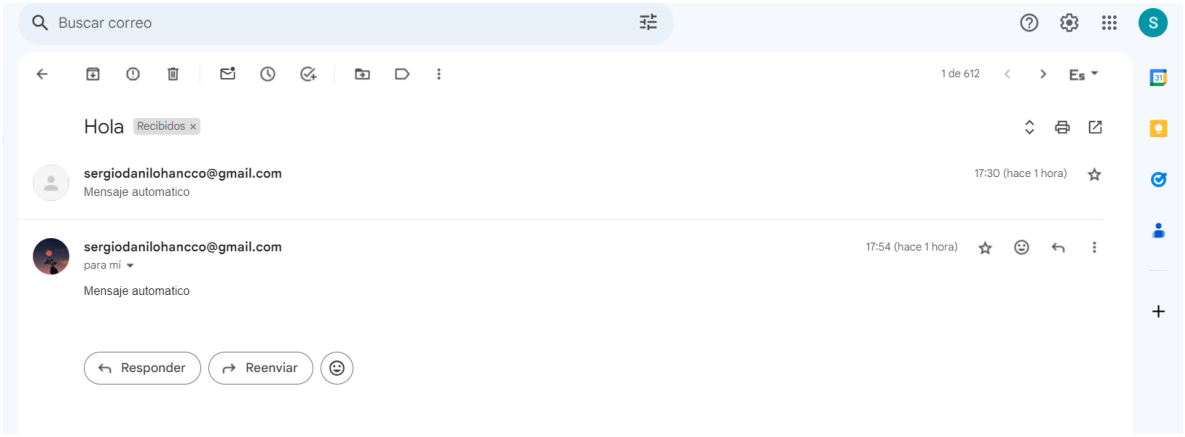
```
from django.shortcuts import render
from django.core.mail import send_mail

def index(request):

    send_mail('Hola',
    'Mensaje automatico PRUEBA 1',
    'correo@gmail.com',
    ['correo@gmail.com'],
    fail_silently = False)

    return render(request, 'send/index.html')
```

Listing 4: settings.py

```
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_PORT = 587
EMAIL_HOST_USER = 'correo@gmail.com'
EMAIL_USE_TLS = True
```

EVIDENCIA DEL ENVIO

Enviar un Email!



5. Rúbricas

5.1. Entregable Informe

Tabla 1: Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

5.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplio con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos lo items.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 2: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 3: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	4	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	1	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	4	
Total		20		19	