

Informe de Laboratorio 08

Tema: Django - Relaciones

Nota

Estudiante	Escuela	Asignatura
Victor Mamani Anahua vmamanian@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación Web 2 Semestre: III Código: 20230489

Laboratorio	Tema	Duración
08	Django - Relaciones	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 8 Junio 2024	Al 15 Junio 2024

1. Tarea

- Informe de laboratorio
- Video en Flip
- Ejercicios Propuestos

2. Equipos, materiales y temas utilizados

- Sistema Operativo Ubuntu GNU Linux 23 lunar 64 bits Kernell 6.2.v
- Sistema Operativo Windows 11.
- Visual Studio Code
- OpenJDK 64-Bits 19.0.7.
- Git 2.39.2.
- Cuenta en GitHub con el correo institucional.

3. URL'S

- No pude enviar el video por Flip

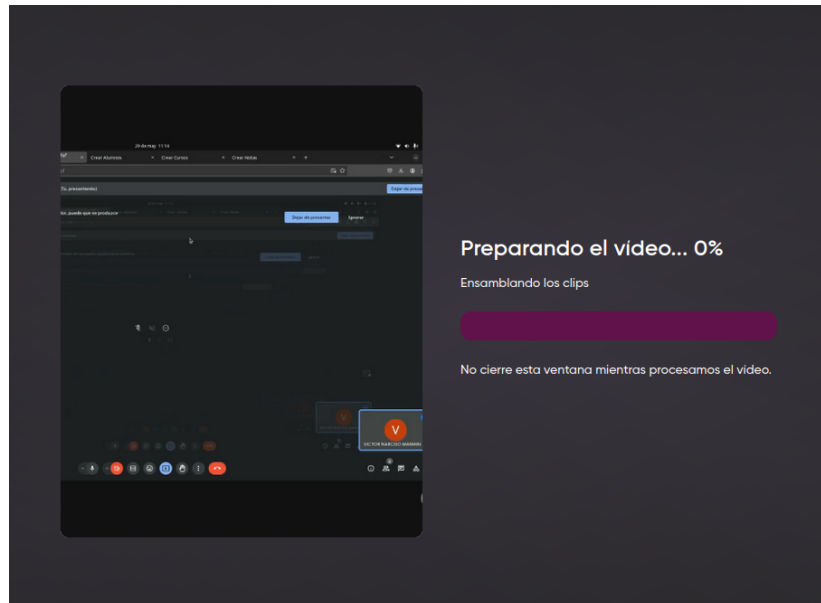


Figura 1: Código y Ejecución

- URL para el video Drive.
-
- URL para el Repositorio GitHub.
- <https://github.com/VictorMA18/Lab08-Django-Relaciones>

4. Actividades del Laboratorio 08

4.1. Hacer migraciones

- Primero activamos nuestro entorno virtual en el cual clonamos el repositorio
- Después hacemos las migraciones respectivas

Listing 1: migraciones

```
python3 manage.py makemigrations
python3 manage.py migrations
```

4.2. Relaciones uno a muchos y de muchos a muchos

- **Language:** Modelo que representa un lenguaje de programación.
 - **name:** Nombre del lenguaje (campo CharField con longitud máxima de 10 caracteres).

- **Framework:** Modelo que representa un framework asociado a un lenguaje.
 - **name:** Nombre del framework (campo `CharField` con longitud máxima de 10 caracteres).
 - **language:** Clave foránea que relaciona el framework con un lenguaje (relación `ForeignKey` hacia el modelo **Language**, con eliminación en cascada).
- **Movie:** Modelo que representa una película.
 - **name:** Nombre de la película (campo `CharField` con longitud máxima de 10 caracteres).
- **Character:** Modelo que representa un personaje de película.
 - **name:** Nombre del personaje (campo `CharField` con longitud máxima de 10 caracteres).
 - **movies:** Relación de muchos a muchos con películas, indicando en qué películas aparece el personaje (relación `ManyToManyField` hacia el modelo **Movie**).

Listing 2: Código de views.py

```
1 from django.db import models
2
3 # Create your models here.
4 class Language(models.Model):
5     name = models.CharField(max_length=10)
6
7     def __str__(self):
8         return self.name
9
10 class Framework(models.Model):
11     name = models.CharField(max_length=10)
12     language = models.ForeignKey(Language, on_delete=models.CASCADE)
13
14     def __str__(self):
15         return self.name
16
17 class Movie(models.Model):
18     name = models.CharField(max_length=10)
19
20     def __str__(self):
21         return self.name
22
23 class Character(models.Model):
24     name = models.CharField(max_length=10)
25     movies = models.ManyToManyField(Movie)
26
27     def __str__(self):
28         return self.name
```

- **Ahora veremos la base de datos:** Base de datos

	id	name
	Filter	Filter
1	3	Python
2	4	Java

Figura 2: Base de Datos

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

Figura 3: Base de Datos

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

Figura 4: Base de Datos

	id	name	
	F...	Filter	
1	1	Captain America	
2	2	Thor	

Figura 5: Base de Datos

	id	character_id	movie_id
	Filter	Filter	Filter
1	1	1	1
2	2	1	2
3	3	2	1
4	4	2	3
5	5	1	4

Figura 6: Base de Datos

4.3. Impresión de pdfs y envío de emails

- Primero descargamos la librería de python xhtml2pdf que nos ayuda a renderizar nuestro html a pdf

Listing 3: migraciones

```
pip install xhtml2pdf
```

- Archivo **utils.py**:
- Función **render_to_pdf**:
 - Descripción: Renderiza un template HTML con un contexto y genera un archivo PDF utilizando xhtml2pdf.
 - Parámetros:
 - **template_src**: Ruta al template HTML que se va a renderizar.
 - **context_dict**: Diccionario con el contexto que se va a pasar al template (opcional, por defecto es un diccionario vacío).
 - Acciones:
 - Carga el template especificado utilizando **get_template**.
 - Renderiza el template con el contexto utilizando **template.render**.
 - Convierte el HTML renderizado a un documento PDF utilizando **pisa.pisaDocument**.
 - Retorna un objeto **HttpResponse** con el contenido del PDF si la conversión fue exitosa.
 - Si hay errores durante la conversión a PDF, retorna una respuesta de error con código 400 y un mensaje de "Invalid PDF".

Listing 4: Código de views.py

```
1 from io import BytesIO
2 from django.http import HttpResponse
3 from django.template.loader import get_template
4
5 from xhtml2pdf import pisa
6
7 def render_to_pdf(template_src, context_dict={}):
8     template = get_template(template_src)
9     html = template.render(context_dict)
10    result = BytesIO()
11    pdf = pisa.pisaDocument(BytesIO(html.encode("ISO-8859-1")), result)
12    if pdf.err:
13        return HttpResponse("Invalid PDF", status_code=400, content_type='text/plain')
14    return HttpResponse(result.getvalue(), content_type='application/pdf')
```

■ Archivo **views.py**:

■ Función **get_pdf**:

- Descripción: Genera un documento PDF de una factura utilizando un template HTML y lo devuelve como respuesta HTTP.
- Acciones:
 - Carga el template 'invoice.html' usando **get_template**.
 - Define un contexto con datos de la factura como número de factura, nombre del cliente, monto y fecha.
 - Renderiza el template a HTML utilizando **template.render**.
 - Genera un PDF llamando a **render_to_pdf**.
 - Si se genera correctamente el PDF:
 - ◊ Crea una respuesta HTTP con el PDF y establece el tipo de contenido como 'application/pdf'.
 - ◊ Configura el nombre del archivo y si debe descargarse o mostrarse en línea según el parámetro **download** en la solicitud.
 - ◊ Retorna la respuesta HTTP con el PDF.
 - Si no se puede generar el PDF, retorna una respuesta HTTP indicando "Not found".

■ Función **get_pdf_advanced**:

- Descripción: Genera un documento PDF más avanzado de una factura con datos formateados y lo devuelve como respuesta HTTP.
- Acciones:
 - Configura el entorno local para formateo de moneda usando **locale.setlocale**.
 - Define datos de la factura como nombre del cliente, número de factura, monto formateado como moneda, fecha y título del PDF.
 - Llama a **utils.render_to_pdf** para generar el PDF utilizando el template 'invoice.html' y el contexto definido.
 - Maneja la respuesta del PDF:
 - ◊ Si la generación tiene éxito, configura la respuesta con el nombre del archivo y el tipo de contenido.
 - ◊ Decide si el archivo debe descargarse o mostrarse en línea según el parámetro **download** en la solicitud.

- ◊ Retorna la respuesta HTTP con el PDF generado.
- ◊ Si no se encuentra la factura (código 404), lanza una excepción HTTP404.

■ Función emails:

- Descripción: Maneja el envío de correos electrónicos desde un formulario POST y muestra un mensaje de confirmación.
- Acciones:
 - Si la solicitud es POST, obtiene el asunto, mensaje y destinatario del formulario.
 - Utiliza `send_mail` para enviar el correo electrónico utilizando los datos obtenidos.
 - Retorna una respuesta HTTP con un mensaje de confirmación y un enlace para volver al menú principal.
 - Si la solicitud no es POST, renderiza el template 'emails.html'.

■ Función index:

- Descripción: Renderiza la página principal ('index.html').
- Acciones:
 - Retorna la respuesta renderizada del template 'index.html'.

■ Correo del Remitente debe ser cambiado por el correo personal

Listing 5: Código de settings.py del proyecto

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

Listing 6: Código de views.py

```
1 from django.http import HttpResponse
2 from django.shortcuts import render
3 from django.views.generic import View
4 from django.template.loader import get_template
5 from django.urls import reverse
6 from .utils import render_to_pdf
7 from django.core.mail import send_mail
8 from django.conf import settings
9 import locale
10 from . import utils
11 # Create your views here.
12
13 def get_pdf(request, *args, **kwargs):
14     template = get_template('invoice.html')
15     context = {
16         "invoice_number": 123,
17         "customer_name": "Victor Mamani",
18         "amount": 1399.9,
19         "date": "Today",
20     }
```



```
21     html = template.render(context)
22     pdf = render_to_pdf('invoice.html', context)
23     if pdf:
24         response = HttpResponse(pdf, content_type='application/pdf')
25         filename = "Invoice_%s" %("12345")
26         content = "inline; filename='%s'" %(filename)
27         download = request.GET.get("download")
28         if download:
29             content = "attachment; filename='%s'" %(filename)
30         response['Content-Disposition'] = content
31         return response
32     return HttpResponse("Not found")
33
34 def get_pdf_advanced(request):
35     locale.setlocale(locale.LC_ALL, "")
36     invoice_number = "007cae"
37     context = {
38         "customer_name": "Victor",
39         "invoice_number": f"{invoice_number}",
40         "amount": locale.currency(100_000, grouping=True),
41         "date": "2024-15-06",
42         "pdf_title": f"Invoice #{invoice_number}",
43     }
44     response = utils.render_to_pdf("invoice.html", context)
45     if response.status_code == 404:
46         raise HTTP404("Invoice not found")
47     filename = f"Invoice_{invoice_number}.pdf"
48     content = f"inline; filename={filename}"
49     download = request.GET.get("download")
50     if download:
51         content = f"attachment; filename={filename}"
52     response["Content-Disposition"] = content
53     return response
54
55 def emails(request):
56     if request.method == 'POST':
57         asunto = request.POST.get('subject', '')
58         mensaje = request.POST.get('message', '')
59         destinatario = request.POST.get('recipient', '')
60         send_mail(asunto, mensaje, settings.EMAIL_HOST_USER, [destinatario],
61                 fail_silently=False)
62         emails_url = reverse('emails')
63         return HttpResponse(f'Correo enviado correctamente. <br> <p><a  

64             href="{emails_url}">Volver al men principal</a></p>')
65     return render(request, 'emails.html')
66
67 def index(request):
68     return render(request, 'index.html')
```

- Archivos HTML emails.html , index.html , invoice.html:
- Archivo emails.html:

Listing 7: Código de views.py

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Enviar Correo</title>
5   <style>
6     body {
7       font-family: Arial, sans-serif;
8       background-color: #b0e0e6;
9       margin: 0;
10      padding: 20px;
11      background-size: cover;
12      background-position: center;
13    }
14    .container {
15      max-width: 500px;
16      margin: 0 auto;
17      background-color: rgba(255, 255, 255, 0.9);
18      padding: 30px;
19      border-radius: 10px;
20      box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
21    }
22    h2 {
23      color: #2c3e50;
24      text-align: center;
25      margin-bottom: 30px;
26      font-size: 28px;
27    }
28    label {
29      display: block;
30      margin-bottom: 10px;
31      color: #34495e;
32      font-weight: bold;
33    }
34    input[type="text"],
35    input[type="email"],
36    textarea {
37      width: 100%;
38      padding: 12px;
39      border: 1px solid #ccc;
40      border-radius: 4px;
41      box-sizing: border-box;
42      font-size: 16px;
43      margin-bottom: 20px;
44      background-color: #ecf0f1;
45    }
46    textarea {
47      resize: vertical;
48    }
49    input[type="submit"] {
50      background-color: #3498db;
51      color: #fff;
52      border: none;
53      padding: 12px 24px;
54      text-align: center;
55      text-decoration: none;
56      display: inline-block;
```

```

57     font-size: 18px;
58     border-radius: 4px;
59     cursor: pointer;
60     transition: background-color 0.3s ease;
61 }
62 input[type="submit"]:hover {
63     background-color: #2980b9;
64 }
65 </style>
66 </head>
67 <body>
68     <h2>Enviar Correo</h2>
69     <form method="post">
70         {% csrf_token %}
71         <label for="id_subject">Asunto:</label><br>
72         <input type="text" id="id_subject" name="subject" required><br><br>
73
74         <label for="id_message">Descripcin:</label><br>
75         <textarea id="id_message" name="message" rows="4" cols="50"
76             required></textarea><br><br>
77
78         <label for="id_recipient">Destinatario:</label><br>
79         <input type="email" id="id_recipient" name="recipient" required><br><br>
80
81         <input type="submit" value="Enviar">
82     </form>
83 </body>
</html>

```

■ Archivo index.html:

Listing 8: Código de views.py

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>Index</title>
5     <style>
6         body {
7             font-family: 'Arial', sans-serif;
8             background-color: #f4f4f4;
9             margin: 0;
10            padding: 20px;
11            text-align: center;
12        }
13
14        h2 {
15            color: #333;
16        }
17
18        .button-container {
19            margin-top: 20px;
20        }
21
22        .button {
23            display: inline-block;

```

```

24         background-color: #007bff;
25         color: #fff;
26         padding: 10px 20px;
27         text-decoration: none;
28         border-radius: 4px;
29         margin: 0 10px;
30         transition: background-color 0.3s ease;
31     }
32
33     .button:hover {
34         background-color: #0056b3;
35     }
36 </style>
37 </head>
38 <body>
39     <h2>Seleccione una opción:</h2>
40     <div>
41         <a href="{% url 'emails' %}" class="button">Enviar Correo</a>
42         <a href="{% url 'generar_pdf_basic' %}" class="button">Generar PDF Básico</a>
43         <a href="{% url 'generar_pdf_advanced' %}" class="button">Generar PDF
44             Avanzado</a>
45     </div>
46 </body>
47 </html>

```

■ Archivo invoice.html:

Listing 9: Código de views.py

```

1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
2   "http://www.w3.org/TR/html4/loose.dtd">
3 <html>
4     <head>
5         <title>Title</title>
6         <style type="text/css">
7             body {
8                 font-weight: 200;
9                 font-size: 14px;
10            }
11            .header {
12                font-size: 20px;
13                font-weight: 100;
14                text-align: center;
15                color: #007cae;
16            }
17            .title {
18                font-size: 22px;
19                font-weight: 100;
20                /* text-align: right;*/
21                padding: 10px 20px 0px 20px;
22            }
23            .title span {
24                color: #007cae;
25            }
26            .details {
27                padding: 10px 20px 0px 20px;

```

```

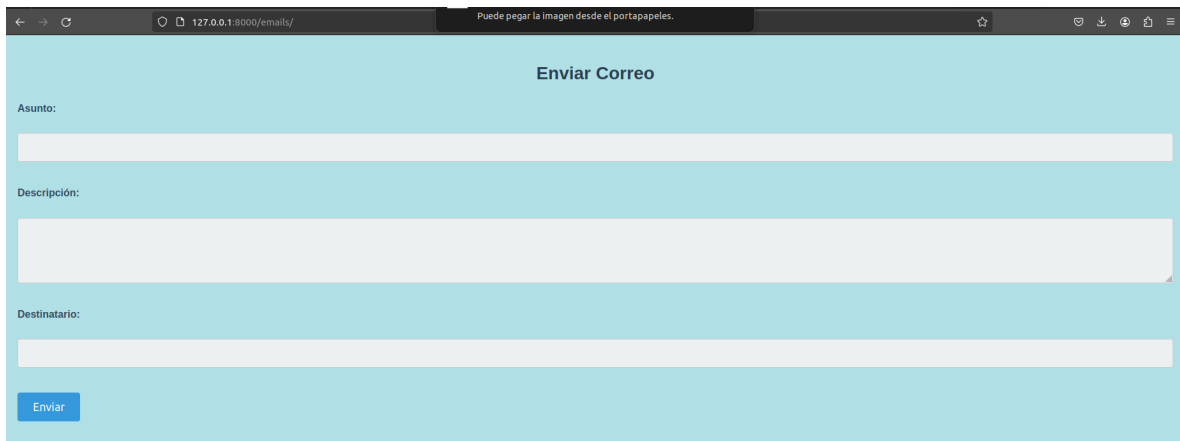
27         text-align: left !important;
28         /*margin-left: 40%;*/
29     }
30     .hrItem {
31         border: none;
32         height: 1px;
33         /* Set the hr color */
34         color: #333; /* old IE */
35         background-color: #fff; /* Modern Browsers */
36     }
37     </style>
38 </head>
39 <body>
40     <div class='wrapper'>
41         <div class='header'>
42             <p class='title'>Invoice #{ invoice_number}</p>
43         </div>
44         <div>
45             <div class='details'>
46                 Bill to: {{ customer_name }}<br/>
47                 Amount: {{ amount }} <br/>
48                 Date: {{ date }}
49                 <hr class='hrItem' />
50             </div>
51         </div>
52     </body>
53 </html>

```

■ Visualizacion:

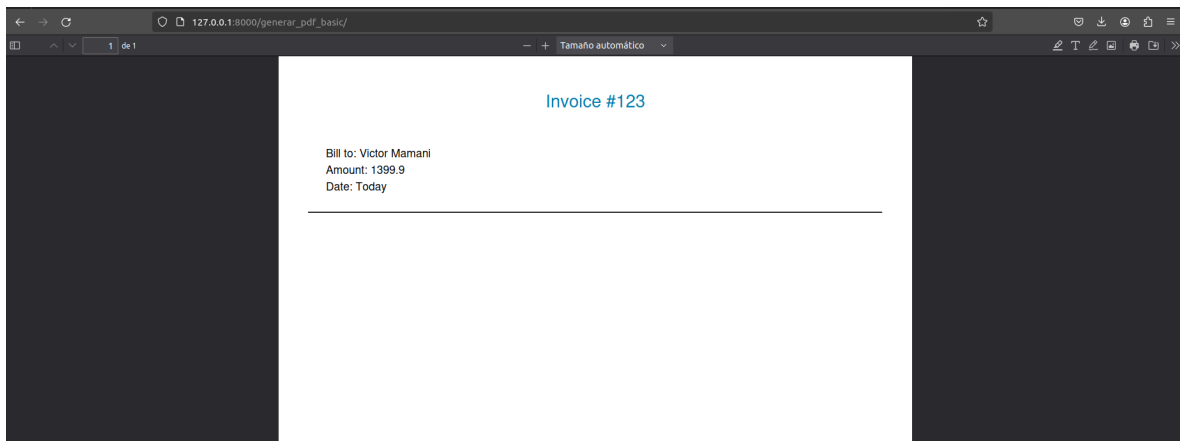


Figura 7: INDEX.HTML



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:8000/emails/'. The page has a light blue background and is titled 'Enviar Correo'. It contains three input fields: 'Asunto:' (Subject), 'Descripción:' (Description), and 'Destinatario:' (Recipient). Below these fields is a blue button labeled 'Enviar' (Send).

Figura 8: EMAILS.HTML



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:8000/generar_pdf_basic/'. The page displays an invoice titled 'Invoice #123'. The invoice details are: 'Bill to: Victor Mamani', 'Amount: 1399.9', and 'Date: Today'. The page is framed by dark sidebars.

Figura 9: INVOICE.HTML



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:8000/generar_pdf_advanced/'. The page displays an invoice titled 'Invoice #007cae'. The invoice details are: 'Bill to: Victor', 'Amount: S/ 100,000.00', and 'Date: 2024-15-06'. The page is framed by dark sidebars.

Figura 10: INVOICE.HTML

4.4. Links de las paginas

- INDEX:
- <http://127.0.0.1:8000/>
- CORREOS:
- <http://127.0.0.1:8000/emails/>
- PDF BASICO:
- http://127.0.0.1:8000/generar_pdf_basic/
- PDF AVANZADO:
- http://127.0.0.1:8000/generar_pdf_advanced/

5. Referencias

- <https://docs.djangoproject.com/es/3.2/>
- <https://docs.djangoproject.com/es/3.2/ref/models/fields/#field-types>
- <https://www.w3schools.com/django/>