

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 1</p>

## INFORME DE LABORATORIO

INFORMACIÓN BÁSICA					
ASIGNATURA:	Programación Web 2				
TÍTULO DE LA PRÁCTICA:	Django				
NÚMERO DE PRÁCTICA:	07	AÑO LECTIVO:	2023 A	NRO. SEMESTRE:	III
FECHA DE PRESENTACIÓN	14/07/2023	HORA DE PRESENTACIÓN			
<b>INTEGRANTE (s):</b> Chara Condori Jean Carlo				<b>NOTA:</b>	
<b>DOCENTE(s):</b> Mg. Anibal Sardón Paniagua					

SOLUCIÓN Y RESULTADOS
<p><b>I. SOLUCIÓN DE EJERCICIOS/PROBLEMAS</b></p> <p>REPOSITORIO GITHUB: <a href="https://github.com/JeanChara/pweb2_lab07.git">https://github.com/JeanChara/pweb2_lab07.git</a></p> <p>FlipGrid: <a href="https://flip.com/d08dc816">https://flip.com/d08dc816</a></p> <p>Ejercicio 1: <a href="https://flip.com/s/4EWCQQiu-JQr">https://flip.com/s/4EWCQQiu-JQr</a></p> <p>Ejercicio 2: <a href="https://flip.com/s/Y2L7SGWeYM7L">https://flip.com/s/Y2L7SGWeYM7L</a></p> <p>Ejercicio 3: <a href="https://flip.com/s/RXv3XKSxsSR4">https://flip.com/s/RXv3XKSxsSR4</a></p> <p>Ejercicio 4: <a href="https://flip.com/s/_sL2qXhUmUUw">https://flip.com/s/_sL2qXhUmUUw</a></p> <ul style="list-style-type: none"> <li>Reproducir las actividades de los videos donde trabajamos:             <ol style="list-style-type: none"> <li>1: Relación de uno a muchos</li> <li>2. Relación de muchos a muchos</li> <li>3. Impresión de pdf's</li> <li>4. Envío de emails</li> </ol> </li> </ul>

- Crear su video en FlipGrid

## Resolución:

### Ejercicio 1 - Relación uno a muchos

En este ejercicio se ha creado el proyecto “uno\_a\_muchos” en el cual se referencia a la aplicación “example”. En el cual se trabajó en los modelos para la creación de la relación uno a muchos.

 <b>JeanChara</b> ejecución (modificación en base de datos) <span>1a41bdb · 17 hours ago</span> <a href="#">History</a>		
Name	Last commit message	Last commit date
..		
example	ejecucion (modificación en base de datos)	17 hours ago
uno_a_muchos	correccion en uno a muchos añadido a admin para creacion d...	19 hours ago
db.sqlite3	ejecucion (modificación en base de datos)	17 hours ago
manage.py	creacion proyecto uno a muchos	3 days ago

En example/models se crea las clases siguiendo las pautas del video. Se utiliza principalmente la clase “Language” y “Framework”.



JeanChara añadido str para lenguaje y framework en uno a muchos

Code

Blame 27 lines (21 loc) · 753 Bytes

```
1  from django.db import models
2
3  # Create your models here.
4
5  class Simple(models.Model):
6      text = models.CharField(max_length=10)
7      number = models.IntegerField(null=True)
8      url = models.URLField(default='www.example.com')
9
10     def __str__(self):
11         return self.url
12
13     class DateExample(models.Model):
14         the_date = models.DateField()
15
16     class NullExample(models.Model):
17         col = models.CharField(max_length=10, blank=True, null=True)
18
19     class Language(models.Model):
20         name = models.CharField(max_length=10)
21         def __str__(self):
22             return self.name
23
24     class Framework(models.Model):
25         name = models.CharField(max_length=10)
26         language = models.ForeignKey(Language, on_delete=models.CASCADE)
27         def __str__(self):
28             return self.name
```

La clase framework contiene un language el cual cumple con la relación uno a muchos puesto que el framework solo puede contener un lenguaje pero un mismo lenguaje puede ser contenido en varios frameworks.

### Ejecución:

Creamos los objetos por el shell:

```
>>> from example.models import Language, Framework
>>> javascript = Language(name = 'JavaScript')
>>> javascript.save()
>>> nodeJS = Framework(name = 'NodeJS')
>>> javascript
<Language: javascript>
>>> nodeJS.language = javascript
>>> nodeJS
<Framework: NodeJS>
>>> nodeJS.save()
>>> python = Language(name = 'Python')
>>> python.save()

>>> dJango = Framework(name = 'DJango', language = python)
>>> dJango.save()
>>> framework_example = Framework(name = 'algun framework', language = javascript)
>>> framework_example.save()
>>>
```

Tabla de lenguajes:

Table: example\_language

	id	name
	Filter	Filter
1	1	JavaScript
2	2	Python

Tabla de frameworks (contiene lenguajes):

Table: example\_framework

	id	name	language_id
	Filter	Filter	Filter
1	1	NodeJS	1
2	2	DJango	2
3	3	Framework X	1

## Ejercicio 2 - Relación muchos a muchos

De manera similar al ejercicio 1, se creo el proyecto “muchos\_a\_muchos” el cual hace referencia a la aplicación example.

<div>  <b>JeanChara</b> <span>ejecucion (modificacion en base de datos)</span> <span>1a41bdb · 17 hours ago</span> <span>History</span> </div>		
Name	Last commit message	Last commit date
..		
example	ejecucion (modificacion en base de datos)	17 hours ago
muchos_a_muchos	correccion muchos a muchos añadido a admin	19 hours ago
db.sqlite3	ejecucion (modificacion en base de datos)	17 hours ago
manage.py	creacion proyecto y app muchos a muchos	3 days ago

En example/models se utiliza la misma base de uno a muchos con la adicion de 2 nuevas clases, Character y Movie.

 **JeanChara** · añadido str para lenguaje y framework en uno a muchos

Code

Blame

41 líneas (29 loc) · 1021 Bytes

```
1  from django.db import models
2
3  class Simple(models.Model):
4      text = models.CharField(max_length=10)
5      number = models.IntegerField(null=True)
6      url = models.URLField(default='www.example.com')
7
8      def __str__(self):
9          return self.url
10
11  class DateExample(models.Model):
12      the_date = models.DateField()
13
14  class NullExample(models.Model):
15      col = models.CharField(max_length=10, blank=True, null=True)
16
17  class Language(models.Model):
18      name = models.CharField(max_length=10)
19
20      def __str__(self):
21          return self.name
22
23  class Framework(models.Model):
24      name = models.CharField(max_length=10)
25      language = models.ForeignKey(Language, on_delete=models.CASCADE)
26
27      def __str__(self):
28          return self.name
29
30  class Movie(models.Model):
31      name = models.CharField(max_length=100)
32
33      def __str__(self):
34          return self.name
35
36  class Character(models.Model):
37      name = models.CharField(max_length=100)
38      movies = models.ManyToManyField(Movie)
39
40      def __str__(self):
41          return self.name
```

La clase movie contiene un nombre de maximo 100 caracteres y una función str la cual llama al nombre del a clase al ser llamada. La clase character además de un nombre, contiene las películas en las cuales ha sido participe. Creando una relación muchos a muchos puesto que un personaje puede estar en varias películas y las películas pueden contener a varios personajes a la vez.

### Ejecución:

Creando Characters y Movies:

```

C:\Windows\system32\cmd.exe - python manage.py shell

4 dirs  7,864,389,632 bytes libres

C:\Users\Jean\Desktop\BACKAP\UNSA universitario\3er CICLO\LAB Programacion web 2\TEMA 7\Nueva carpeta\Nueva carpeta\pw
b2_lab07\muchos_a_muchos\muchos_a_muchos>python manage.py shell
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
(InteractiveConsole)
>> from example.models import Character, Movie
>> iron_man = Character(name = 'Iron man')
>> iron_man.save()
>> capitan_america = Character(name = 'capitan america')
>> capitan_america.save()
>> vengadores = Movie(name = 'Los vengadores')
>> vengadores.save()
>> civil_war = Movie(name = 'Civil War')
>> civil_war.save()
>> iron_man.movies.add(vengadores)
>> iron_man.movies.add(civil_war)
>> capitan_america.movies.add(vengadores)
>> capitan_america.movies.add(civil_war)
>> spiderman = Character(name = 'Spider-man')
>> spiderman.save()
>> thor = Character(name = 'Thor')
>> thor.save()
>> thor.movies.add(vengadores)
>> spiderman.movies.add(civil_war)
>> iron_man.movies.all()
QuerySet [<Movie: Los vengadores>, <Movie: Civil War>]>
>>
  
```

Tabla Characters:



Table:
 

example\_character

▼




	id	name
	Filter	Filter
1	1	Iron man
2	2	capitan america
3	3	Spider-man
4	4	Thor

### Tabla Movies:

Table:  example\_movie 

	id	name
	Filter	Filter
1	1	Los vengadores
2	2	Civil War

### Tabla Characters-Movies (relación muchos a muchos):

Table:  example\_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	2
5	5	4	1
6	6	3	2


## Ejercicio 3 - Impresión de pdf's



En el ejercicio 3 se ha creado el proyecto "impresion\_pdf" el cual tendrá sus propias views y hará uso de un template.



 <b>JeanChara</b> completado, completado views, descarga automatica segun url <span>be0bd8c · 3 days ago</span> <a href="#">History</a>		
Name	Last commit message	Last commit date
..		
impresion_pdf	completado, completado views, descarga automatica segun url	3 days ago
templates/pdf	creado funcion extra para forzar nombre al descargar el pdf	3 days ago
db.sqlite3	creacion de modulo utils.py y completada funcion "render_to_...	3 days ago
manage.py	creacion de modulo utils.py y completada funcion "render_to_...	3 days ago


 <b>JeanChara</b> creado funcion extra para forzar nombre al descargar el pdf <span>6c8e335 · 3 days ago</span> <a href="#">History</a>		
Name	Last commit message	Last commit date
..		
invoice.html	creado funcion extra para forzar nombre al descargar el pdf	3 days ago

 <b>JeanChara</b> completado, completado views, descarga automatica segun url <span>be0bd8c · 3 days ago</span> <a href="#">History</a>		
Name	Last commit message	Last commit date
..		
__pycache__	completado, completado views, descarga automatica segun url	3 days ago
__init__.py	creacion de modulo utils.py y completada funcion "render_to_...	3 days ago
asgi.py	creacion de modulo utils.py y completada funcion "render_to_...	3 days ago
settings.py	creado view Generatepdf y agregado en urls	3 days ago
urls.py	creado view Generatepdf y agregado en urls	3 days ago
utils.py	creacion de modulo utils.py y completada funcion "render_to_...	3 days ago
views.py	completado, completado views, descarga automatica segun url	3 days ago
wsgi.py	creacion de modulo utils.py y completada funcion "render_to_...	3 days ago

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 10</p>

Para la impresión del pdf se utiliza el módulo utils, views y urls

impresión pdf/utils: Se encarga de renderizar el pdf y retornar un html en forma de pdf

 **JeanChara** creación de modulo utils.py y completada funcion "render\_to\_pdf"

**Code** Blame 14 líneas (12 loc) · 480 Bytes

```

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 not pdf.err:
13         return HttpResponse(result.getvalue(), content_type='application/pdf')
14     return None

```

impresion\_pdf/views: Generamos el pdf, brindando un context para la impresion del html el cual se renderizara a pdf, luego verifica si el pdf se reenderizo correctamente, si es asi, se establece el nombre del pdf, el encabezado. Además si se tiene el parametro de download en la solicitud get, se indicará al sistema que el pdf se descargara automáticamente.

 **JeanChara** completado, completado views, descarga automatica segun url**Code**

Blame

27 lines (25 loc) · 1007 Bytes

```
1  from django.http import HttpResponse
2  from django.views.generic import View
3
4  from django.template.loader import get_template
5  from .utils import render_to_pdf
6
7  class GeneratePdf(View):
8      def get(self, request, *args, **kwargs):
9          template = get_template('pdf/invoice.html')
10         context = {
11             'customer_name': 'Mayte Antuanet',
12             'invoice_id': 13,
13             'amount': 39.99,
14             'today': 'Today',
15         }
16         html = template.render(context)
17         pdf = render_to_pdf('pdf/invoice.html', context)
18         if pdf:
19             response = HttpResponse(pdf, content_type='application/pdf')
20             filename = "%s.pdf" %("nombre de descarga")
21             content = "inline; filename=%s" %(filename)
22             download = request.GET.get("download")
23             if download:
24                 content = "attachment; filename=%s" %(filename)
25                 response['Content-Disposition'] = content
26             return response
27         return HttpResponse("Not found")
```

impresion\_pdf/urls: con la url "pdf/" llama al template invoice.html el cual se renderiza a pdf

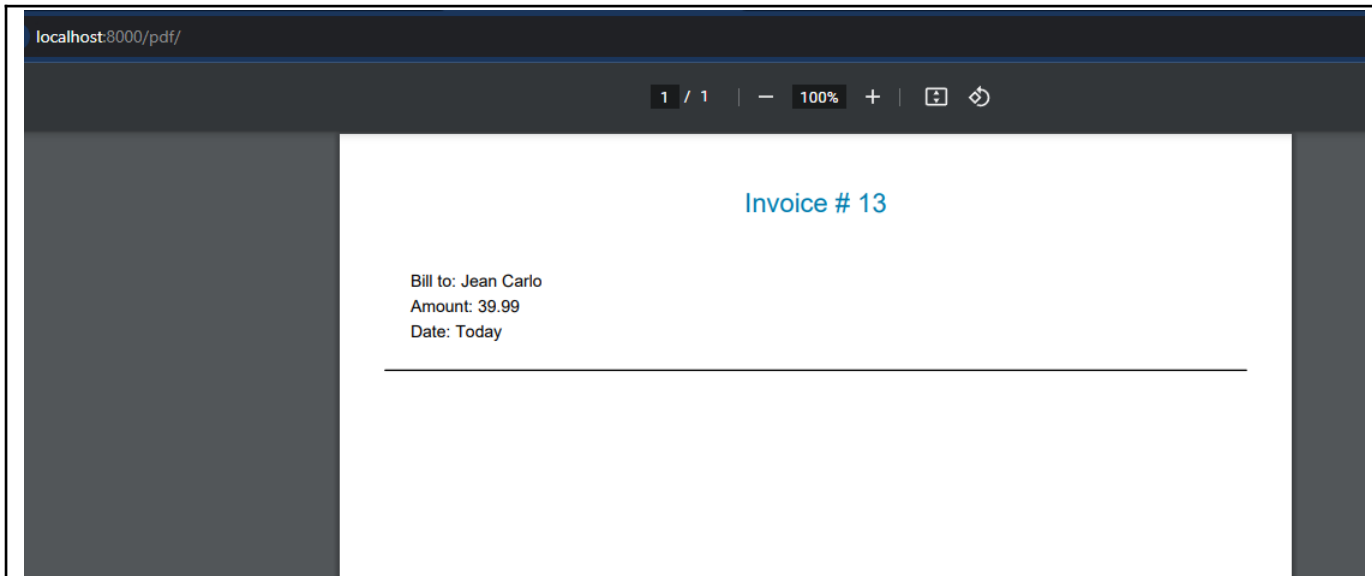
 **JeanChara** creado view Generatepdf y agregado en urls**Code**

Blame

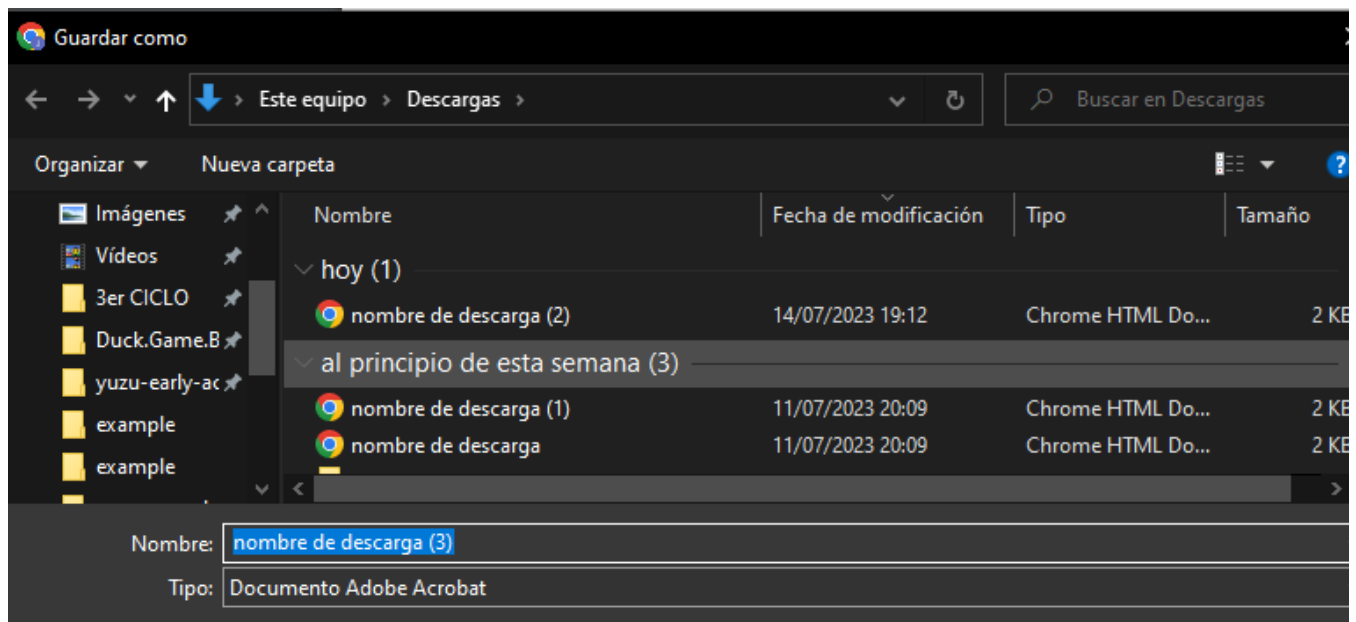
25 lines (22 loc) · 842 Bytes

```
1  """
2  URL configuration for impresion_pdf project.
3
4  The `urlpatterns` list routes URLs to views. For more information please see:
5      https://docs.djangoproject.com/en/4.2/topics/http/urls/
6  Examples:
7  Function views
8      1. Add an import: from my_app import views
9      2. Add a URL to urlpatterns: path('', views.home, name='home')
10 Class-based views
11     1. Add an import: from other_app.views import Home
12     2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
13 Including another URLconf
14     1. Import the include() function: from django.urls import include, path
15     2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
16 """
17 from django.contrib import admin
18 from django.urls import path
19
20 from .views import GeneratePdf
21
22 urlpatterns = [
23     path('admin/', admin.site.urls),
24     path('pdf/', GeneratePdf.as_view()),
25 ]
```

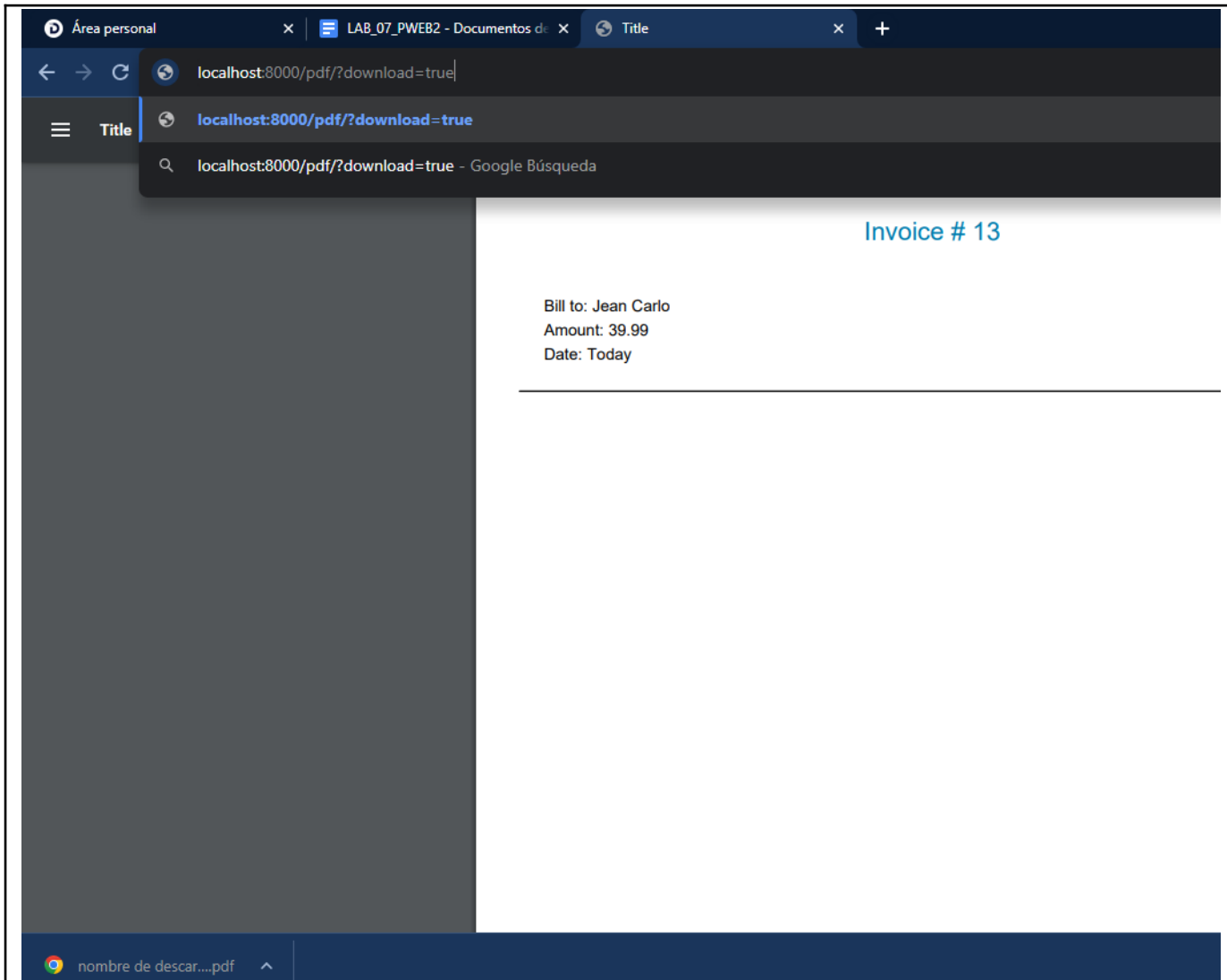
**Ejecución:**



al descargar nos coloca como nombre de archivo “nombre de descarga”



Al establecer el parámetro download=true nos descarga el archivo automáticamente.



## Ejercicio 4 - Envío de emails

envio\_email/settings

colocamos los datos del usuario el cual mandara el email a nuestro correo especificado en views:

```
settings.py X  index.html  urls.py C:\...\send 1  urls.py C:\...\envio_email 3  views.py 2
> 3er CICLO > (LAB) Programacion web 2 > TEMA 7 > Nueva carpeta > Nueva carpeta > pweb2_lab07 > envio_email > envio_email
111 USE_TZ = True
112
113 USE_TZ = True
114
115
116 # Static files (CSS, JavaScript, Images)
117 # https://docs.djangoproject.com/en/4.2/howto/static-files/
118
119 STATIC_URL = 'static/'
120
121 EMAIL_HOST = 'smtp.gmail.com'
122 EMAIL_PORT = 587
123 EMAIL_HOST_USER = 'yinfeik2@gmail.com'
124 EMAIL_HOST_PASSWORD = 'contraseña de aplicacion'
125 EMAIL_USE_TLS = True
126 EMAIL_USE_SSL = False
127
128 # Default primary key field type
129 # https://docs.djangoproject.com/en/4.2/ref/settings/#default-auto-field
130
131 DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
132
```

envio\_email/urls , send/urls

Especificamos que cuando el url se encuentre vacio (default) nos dirija a las urls de la aplicacion send, la cual nos dirige a nuestro view index.

```
settings.py  index.html  urls.py C:\...\send 1  urls.py C:\...\envio_email 3 X  views.p
universitario > 3er CICLO > (LAB) Programacion web 2 > TEMA 7 > Nueva carpeta > Nueva carpeta > pweb2_lab07 > en
1  """
2  URL configuration for envio_email project.
3
4  The `urlpatterns` list routes URLs to views. For more information please see:
5  |   https://docs.djangoproject.com/en/4.2/topics/http/urls/
6  Examples:
7  Function views
8  |   1. Add an import:  from my_app import views
9  |   2. Add a URL to urlpatterns:  path('', views.home, name='home')
10 Class-based views
11 |   1. Add an import:  from other_app.views import Home
12 |   2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
13 Including another URLconf
14 |   1. Import the include() function: from django.urls import include, path
15 |   2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
16 """
17 from django.contrib import admin
18 from django.urls import path,include
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('', include('send.urls')),
23 ]
24
```

```
settings.py  index.html  urls.py C:\...\send 1 X
universitario > 3er CICLO > (LAB) Programacion web 2 > TEMA 7 > Nueva carp
1  from django.urls import path
2  from . import views
3
4  urlpatterns = [
5     path('', views.index),
6 ]
7
```





## send/views

Utilizamos la función `send_mail` la cual se encargara de mandar el email a nuestro correo institucional, especificando el correo remitente y el correo que recibe el email. Seguidamente, renderizamos nuestro html indicando que el email se envio correctamente.

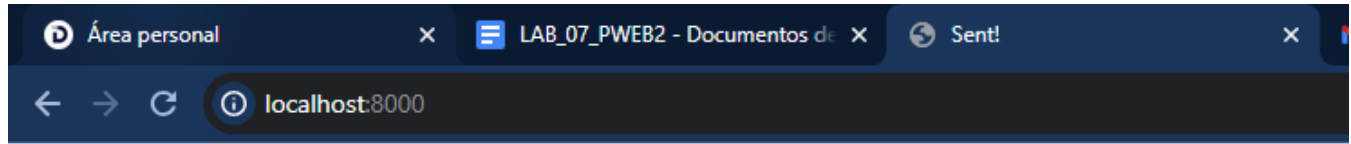
```
rio > 3er CICLO > (LAB) Programacion web 2 > TEMA 7 > Nueva carpeta > Nueva carpeta
1  from django.shortcuts import render
2  from django.core.mail import send_mail
3
4  def index(request):
5
6      send_mail('Hello from Django',
7              'Hello there. This is an automated message.',
8              'yinfeik2@gmail.com',
9              ['jcharaco@unsa.edu.pe'],
10             fail_silently=False)
11
12     return render(request, 'send/index.html')
13
```

## send/template/index.html

```
io > 3er CICLO > (LAB) Programacion web 2 > TEMA 7 > Nueva carpeta > Nueva carpeta > pweb2_lab07 > envio_email
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4
5      <meta charset="UTF-8">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7
8      <title>Sent!</title>
9
10 </head>
11 <body>
12
13     <h1>Sent an email!</h1>
14
15 </body>
16 </html>
```

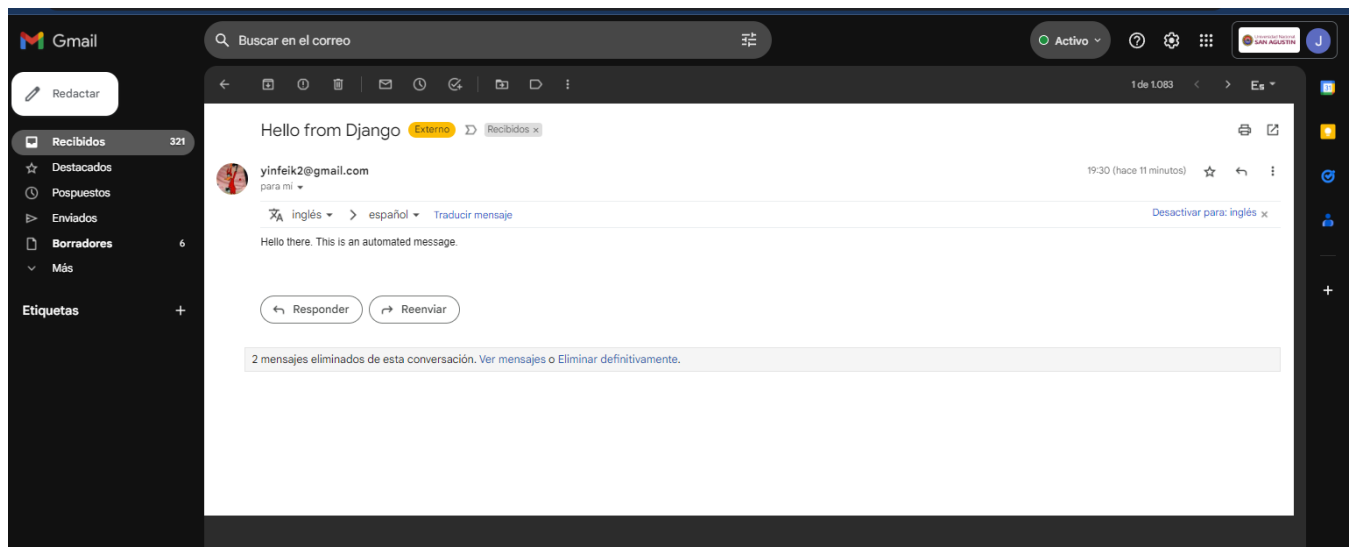
	<p style="text-align: center;"><b>UNIVERSIDAD NACIONAL DE SAN AGUSTIN</b>  <b>FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS</b>  <b>ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</b></p>	
<p style="text-align: center;"><b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p><b>Aprobación:</b> 2022/03/01</p>	<p><b>Código:</b> GUIA-PRLD-001</p>	<p><b>Página:</b> 18</p>

## Ejecución:






















































**Sent an email!**

Revisando el correo institucional:



## Commits del trabajo:

Commits on Jul 13, 2023		
 JeanChara committed yesterday	<b>ejecucion (modificacion en base de datos)</b>	 1a41bdb <>
 JeanChara committed yesterday	<b>añadido str para lenguaje y framework en uno a muchos</b>	 5f99d48 <>
 JeanChara committed yesterday	<b>correccion muchos a muchos añadido a admin</b>	 981e1d8 <>
 JeanChara committed yesterday	<b>correccion en uno a muchos añadido a admin para creacion de objetos y...</b>	 7f58aa4 <>
Commits on Jul 11, 2023		
 JeanChara committed 3 days ago	<b>correccion zona horaria</b>	 adb27c9 <>
 JeanChara committed 3 days ago	<b>correccion en settings y url</b>	 4815d88 <>
 JeanChara committed 3 days ago	<b>añadido variables para envío de email en settings.py</b>	 86b429f <>
 JeanChara committed 3 days ago	<b>avance views, redireccion en urls y creacion de index en templates</b>	 ee4657b <>
 JeanChara committed 3 days ago	<b>creacion proyecto envio_email</b>	 945dad7 <>
 JeanChara committed 3 days ago	<b>completado, completado views, descarga automatica segun url</b>	 be0bd8c <>
 JeanChara committed 3 days ago	<b>creado funcion extra para forzar nombre al descargar el pdf</b>	 6c8e335 <>
 JeanChara committed 3 days ago	<b>correccion invoice html tags</b>	 8594669 <>
 JeanChara committed 3 days ago	<b>creado view Generatepdf y agregado en urls</b>	 3e26aa8 <>
 JeanChara committed 3 days ago	<b>creacion de invoice.html en templates/pdf</b>	 8b11182 <>
 JeanChara committed 3 days ago	<b>creacion de modulo utils.py y completada funcion "render_to_pdf"</b>	 e3be7aa <>
 JeanChara committed 3 days ago	<b>creacion y completado clase Character</b>	 35829bf <>
 JeanChara committed 3 days ago	<b>creacion clase Movie</b>	 b3ecc95 <>
 JeanChara committed 3 days ago	<b>importando modelos de uno_a_muchos</b>	 b7b26c2 <>
 JeanChara committed 3 days ago	<b>creacion proyecto y app muchos a muchos</b>	 77bf085 <>
 JeanChara committed 3 days ago	<b>creado y completado clase language y clase framework en example/models</b>	 9871a24 <>
 JeanChara committed 3 days ago	<b>creacion date y null examples</b>	 13c2981 <>
 JeanChara committed 3 days ago	<b>avance creacion models</b>	 0e01aff <>
 JeanChara committed 3 days ago	<b>creacion proyecto uno a muchos</b>	 94d1de4 <>
 JeanChara committed 3 days ago	<b>Initial commit</b>	  ed3a461 <>
<div>Newer Older</div>		

	<p style="text-align: center;"><b>UNIVERSIDAD NACIONAL DE SAN AGUSTIN</b>  <b>FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS</b>  <b>ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</b></p>	
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 20

## II. SOLUCIÓN DEL CUESTIONARIO

## III. CONCLUSIONES

- Django nos permite manejar relaciones complejas entre modelos como lo es el uno a muchos y muchos a muchos apoyándose de campos como ForeignKey y ManyToManyField. Asimismo nos permite realizar acciones como lo es la generación de archivos PDF mediante diversas Bibliotecas.
- El envío de correos electrónicos es una funcionalidad desarrollada de manera sencilla a través de Django, pero puede verse afectada por factores externos que impiden la realización plena del proceso (medida de seguridad de Google la cual califica a este acceso poco seguro).
- La relación uno a muchos y muchos a muchos es fundamental en el diseño de base de datos y modelado de aplicaciones.

## REFERENCIAS Y BIBLIOGRAFÍA

**FlipGrid:** <https://flip.com/d08dc816>

Ejercicio 1: <https://flip.com/s/4EWCQQiu-JQr>

Ejercicio 2: <https://flip.com/s/Y2L7SGWeYM7L>

Ejercicio 3: <https://flip.com/s/RXv3XKSxsSR4>

Ejercicio 4: [https://flip.com/s/\\_sL2qXhUmUUw](https://flip.com/s/_sL2qXhUmUUw)

<https://docs.djangoproject.com/en/4.2/topics/email/>

<https://xhtml2pdf.readthedocs.io/en/latest/usage.html>