



Basado en Curso Django de Píldoras Informáticas:

<https://www.youtube.com/watch?v=7XO1AzwkPPE&list=PLU8oAIHdN5BmfwxFO7HdPciOCmmYneAB&index=2>



www.sena.edu.co



**Píldoras
Informáticas**

Cursos de informática

Plantillas

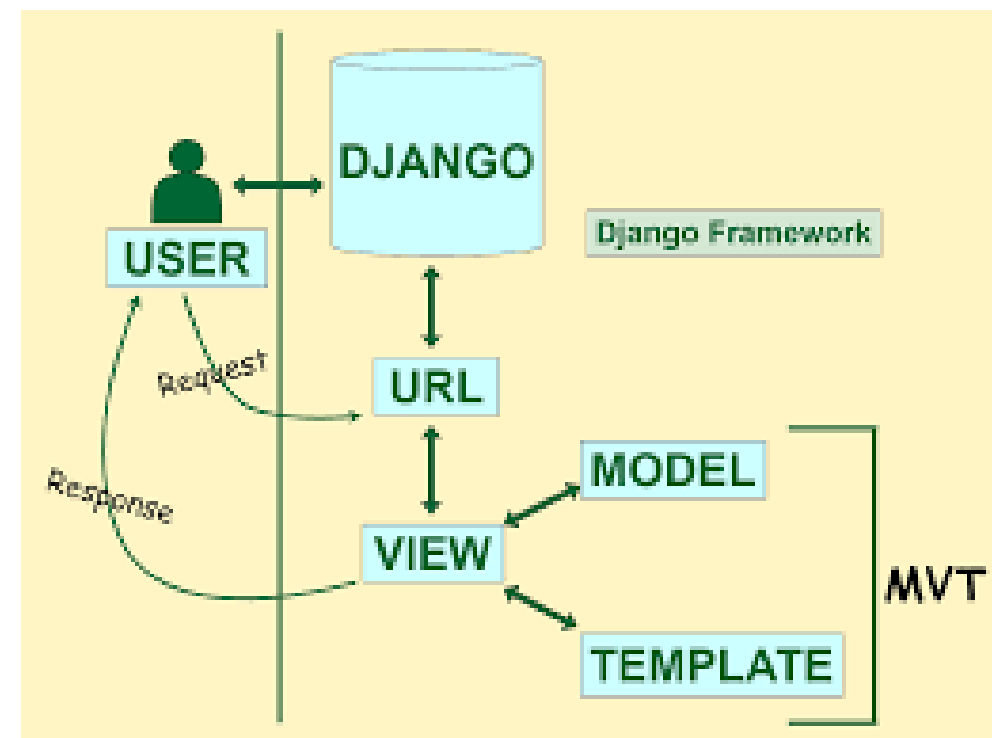


Plantillas

Las plantillas en Django son cadenas de texto que contienen código HTML y se utilizan para separar la lógica del proyecto (Datos) de su aspecto visual, lo que facilita la manipulación del diseño y la lógica de forma independiente.

Utilización de plantillas en Django

La forma más común de utilizar plantillas en Django es guardar la cadena de texto HTML en un archivo independiente y cargarlo en la vista del proyecto.



Ventajas de utilizar plantillas en Django

Separación del código HTML y Python, lo que facilita realizar cambios en el diseño de forma independiente.

Posibilidad de trabajar de forma simultánea en el diseño y la lógica del proyecto, permitiendo la colaboración eficiente entre diseñadores y programadores.

Mejora la organización y mantenimiento del proyecto al evitar la mezcla de lógica y diseño en un mismo archivo.

Pasos básicos para utilizar plantillas en Django

Crear un objeto de tipo template: Se utiliza la clase Template para cargar el documento HTML externo.

```
plt=Template(doc_externo.read())
```

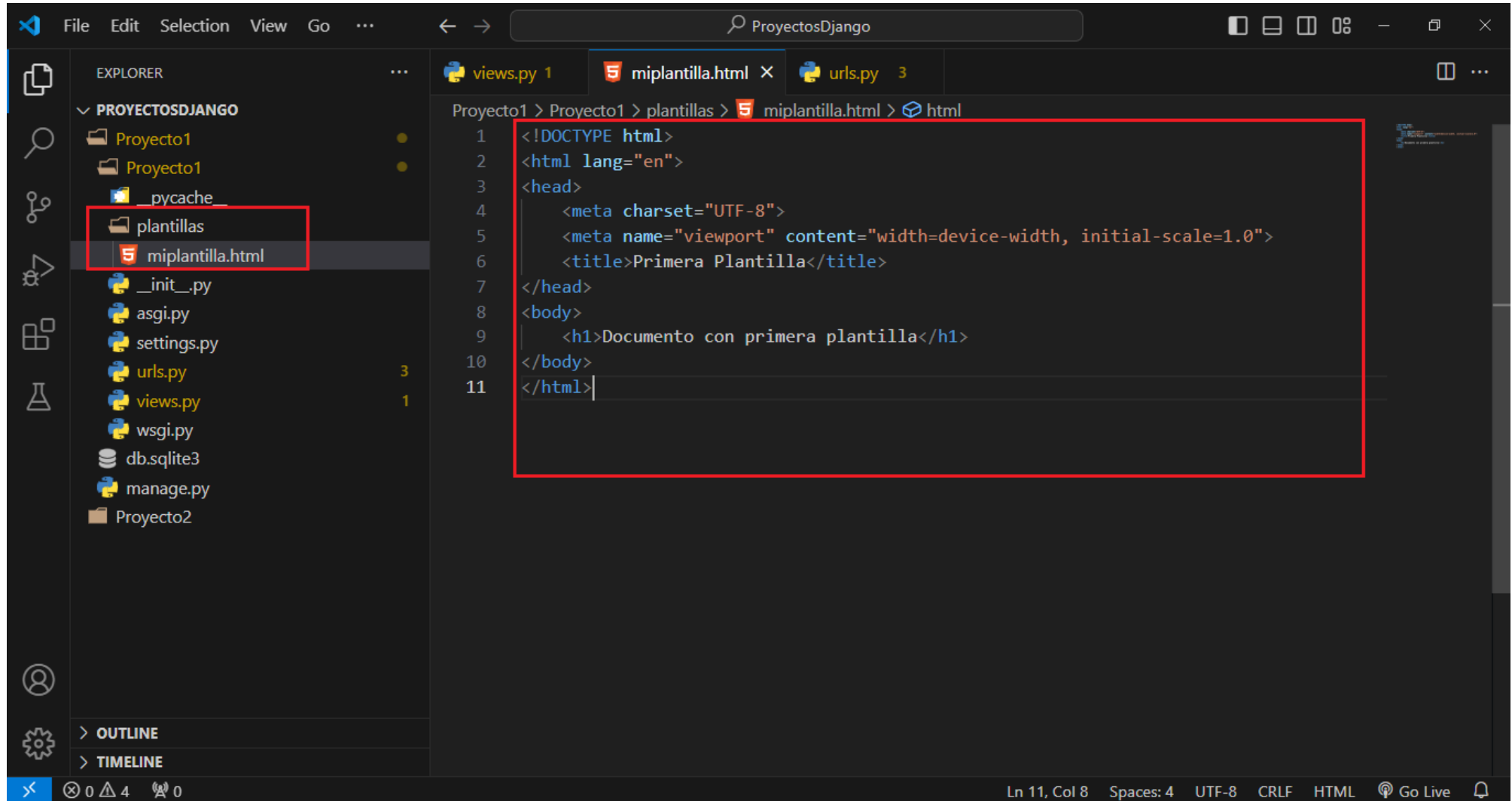
Crear un contexto: El contexto son los datos adicionales que puede utilizar la plantilla, incluso si no son dinámicos.

```
ctx=Context()
```

Renderizar el objeto template: Utilizando el método render, se renderiza el objeto template con el contexto creado.

```
documento=plt.render(ctx)
```

Pasos para crear una plantilla



The screenshot shows the Visual Studio Code interface with a Django project named 'ProyectosDjango'. The Explorer sidebar on the left displays the project structure, with the 'plantillas' folder and the 'miplantilla.html' file highlighted. The main editor window shows the content of 'miplantilla.html', which is a basic HTML template. The code is as follows:

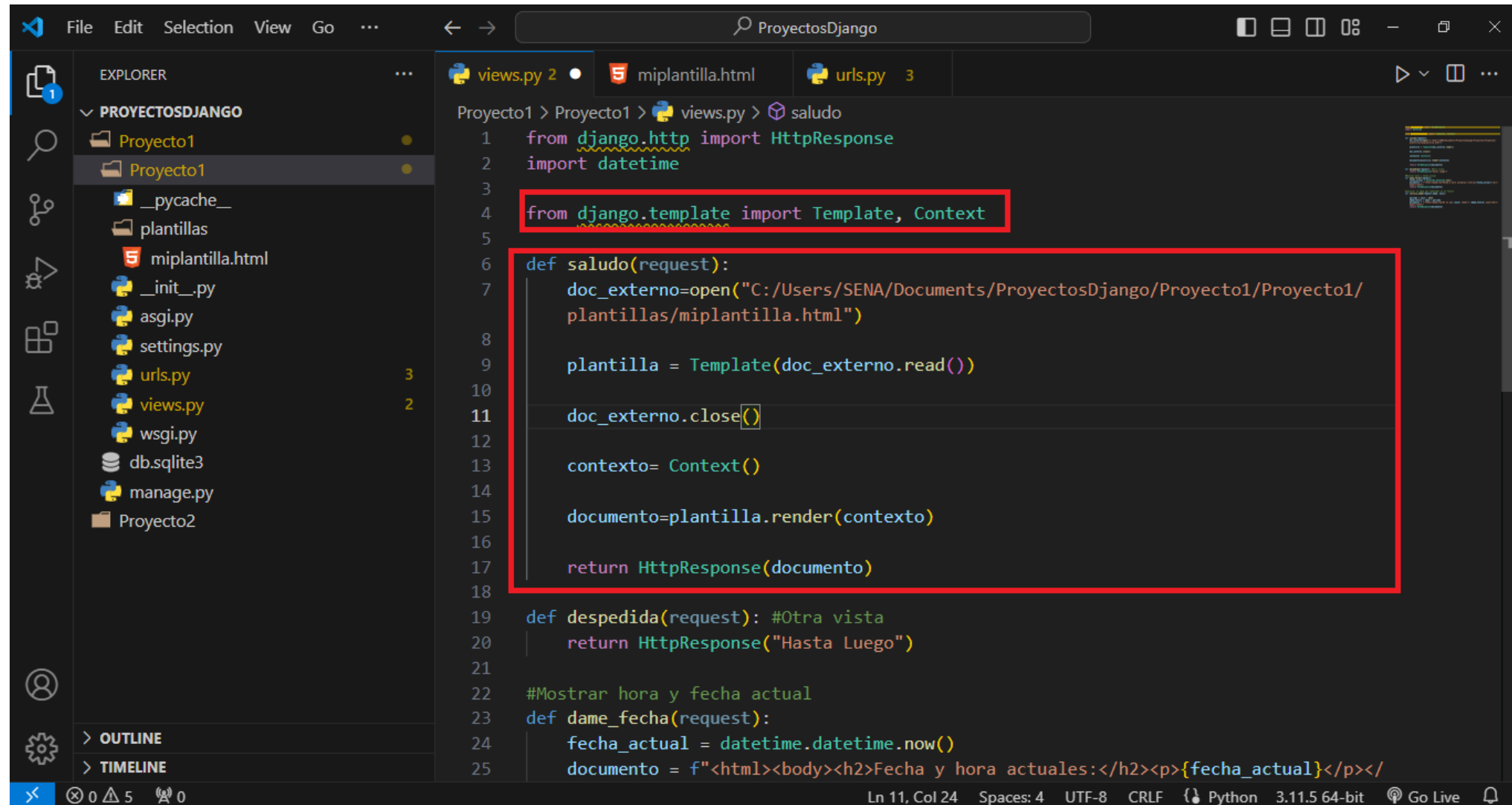
```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Primera Plantilla</title>
7 </head>
8 <body>
9   <h1>Documento con primera plantilla</h1>
10 </body>
11 </html>

```

The status bar at the bottom indicates the current position is Line 11, Column 8, with 4 spaces, UTF-8 encoding, CRLF line endings, and HTML language mode.

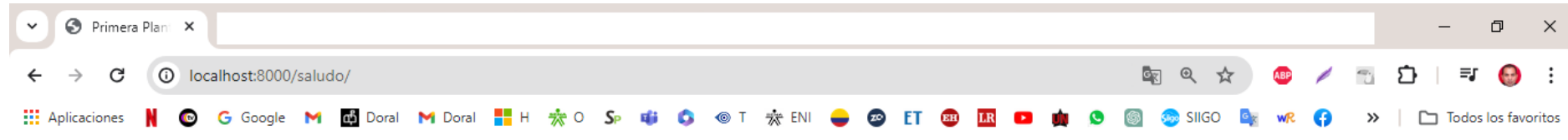
Modificamos la vista Saludo



```

1  from django.http import HttpResponse
2  import datetime
3
4  from django.template import Template, Context
5
6  def saludo(request):
7      doc_externo=open("C:/Users/SENA/Documents/ProyectosDjango/Proyecto1/Proyecto1/
8      plantillas/miplantilla.html")
9
10     plantilla = Template(doc_externo.read())
11
12     doc_externo.close()
13
14     contexto= Context()
15
16     documento=plantilla.render(contexto)
17
18     return HttpResponse(documento)
19
20 def despedida(request): #Otra vista
21     return HttpResponse("Hasta Luego")
22
23 #Mostrar hora y fecha actual
24 def dame_fecha(request):
25     fecha_actual = datetime.datetime.now()
26     documento = f"<html><body><h2>Fecha y hora actuales:</h2><p>{fecha_actual}</p></
  
```

Comprobamos en el Navegador



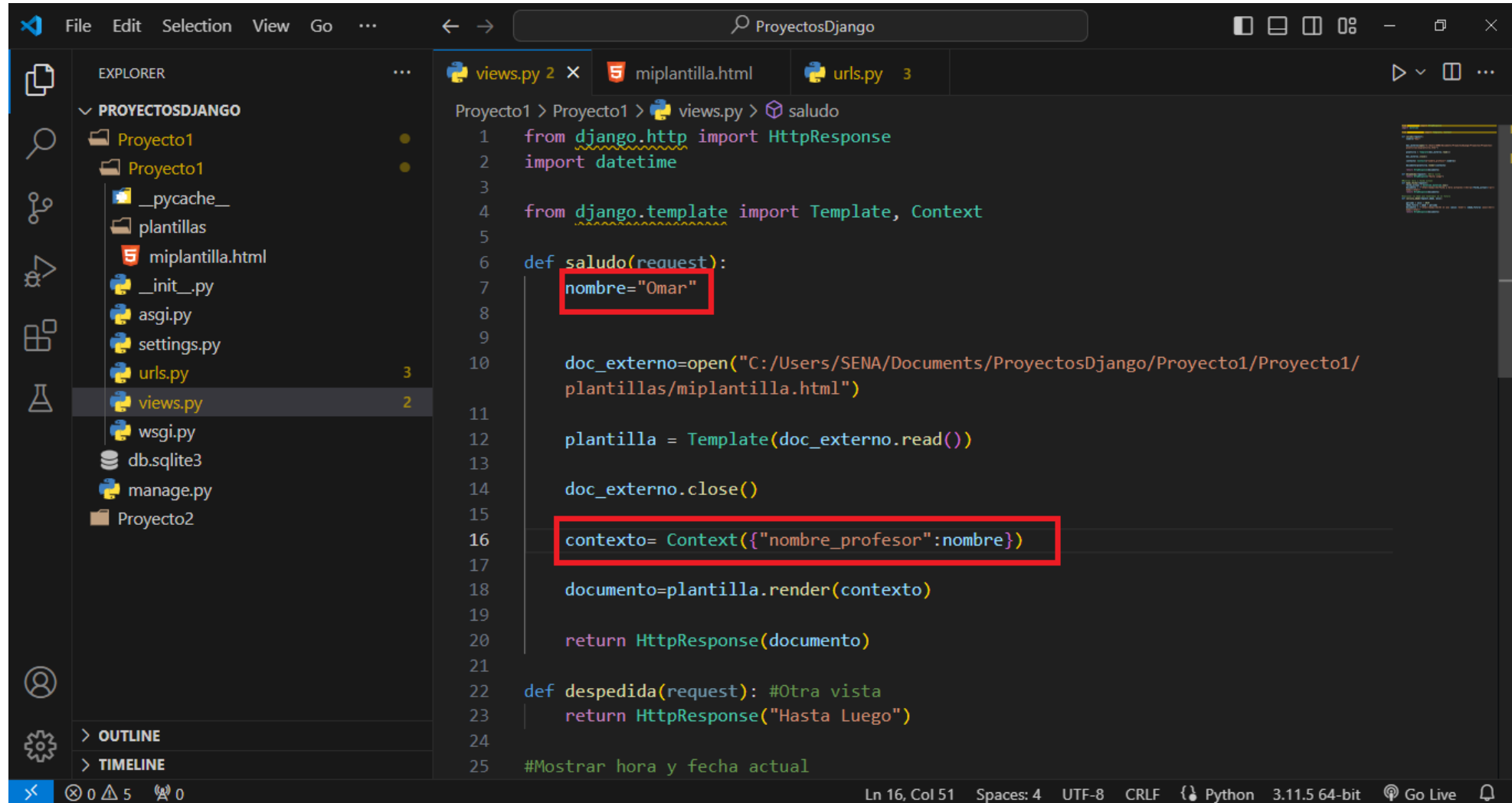
Documento con primera plantilla

Plantillas Variables

- Utilización de variables en plantillas.
- Acceso a valores almacenados en variables desde las plantillas.
 - Uso de diccionarios en contexto

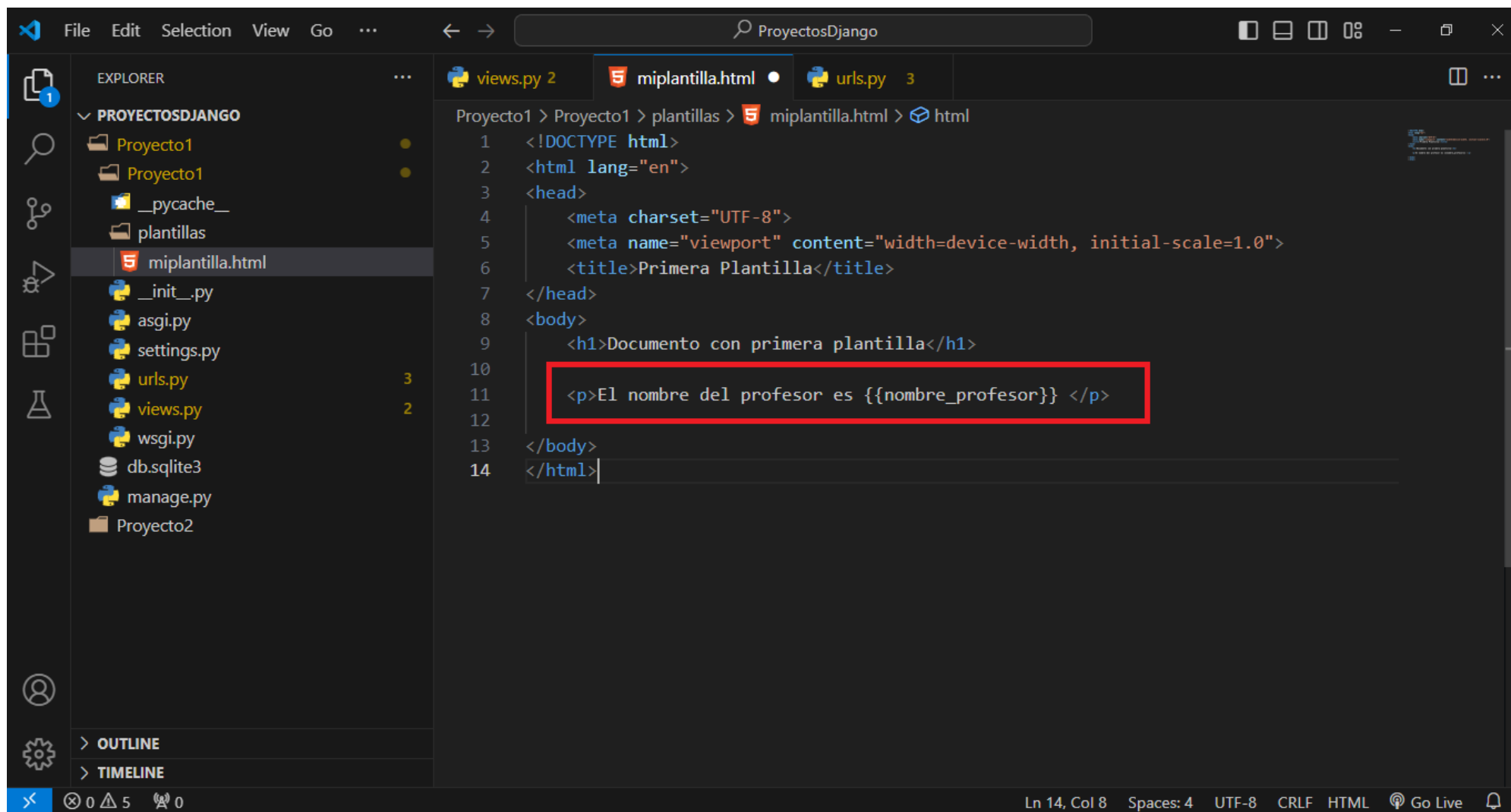
Es posible desde una plantilla hacer referencia a valores almacenados en variables que podamos tener en nuestras vistas en el código python

Modificamos la vista Saludo



```
File Edit Selection View Go ... ProyectoDjango
EXPLORER
PROYECTOSDJANGO
  Proyecto1
    Proyecto1
      __pycache__
      plantillas
      miplantilla.html
      __init__.py
      asgi.py
      settings.py
      urls.py
      views.py
      wsgi.py
      db.sqlite3
      manage.py
    Proyecto2
  > OUTLINE
  > TIMELINE
views.py 2 x miplantilla.html urls.py 3
Proyecto1 > Proyecto1 > views.py > saludo
1 from django.http import HttpResponse
2 import datetime
3
4 from django.template import Template, Context
5
6 def saludo(request):
7     nombre="Omar"
8
9
10     doc_externo=open("C:/Users/SENA/Documents/ProyectosDjango/Proyecto1/Proyecto1/
11     plantillas/miplantilla.html")
12
13     plantilla = Template(doc_externo.read())
14
15     doc_externo.close()
16     contexto= Context({"nombre_profesor": nombre})
17
18     documento=plantilla.render(contexto)
19
20     return HttpResponse(documento)
21
22 def despedida(request): #Otra vista
23     return HttpResponse("Hasta Luego")
24
25 #Mostrar hora y fecha actual
Ln 16, Col 51 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit Go Live
```

Modificamos Plantilla



The screenshot shows the Visual Studio Code editor interface. On the left, the Explorer sidebar displays the project structure for 'PROYECTOSDJANGO'. The file 'miplantilla.html' is selected. The main editor area shows the content of 'miplantilla.html' with the following code:

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Primera Plantilla</title>
7 </head>
8 <body>
9     <h1>Documento con primera plantilla</h1>
10     <p>El nombre del profesor es {{nombre_profesor}} </p>
11
12
13 </body>
14 </html>

```

The line containing the Django template tag `<p>El nombre del profesor es {{nombre_profesor}} </p>` is highlighted with a red rectangular box. The status bar at the bottom indicates the current position is 'Ln 14, Col 8'.

Comprobamos en el Navegador

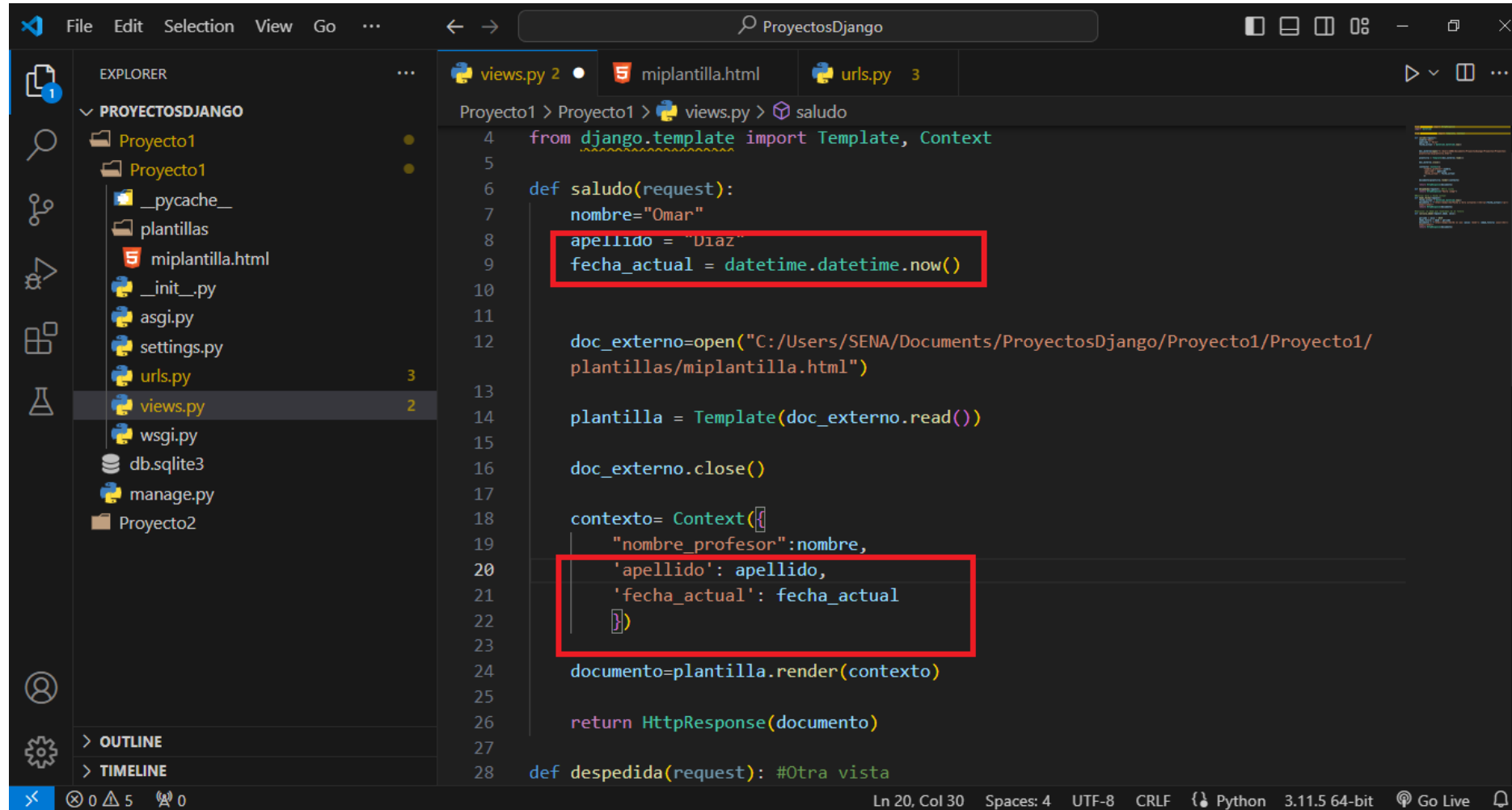


Documento con primera plantilla

El nombre del profesor es Omar



Modificamos la vista Saludo

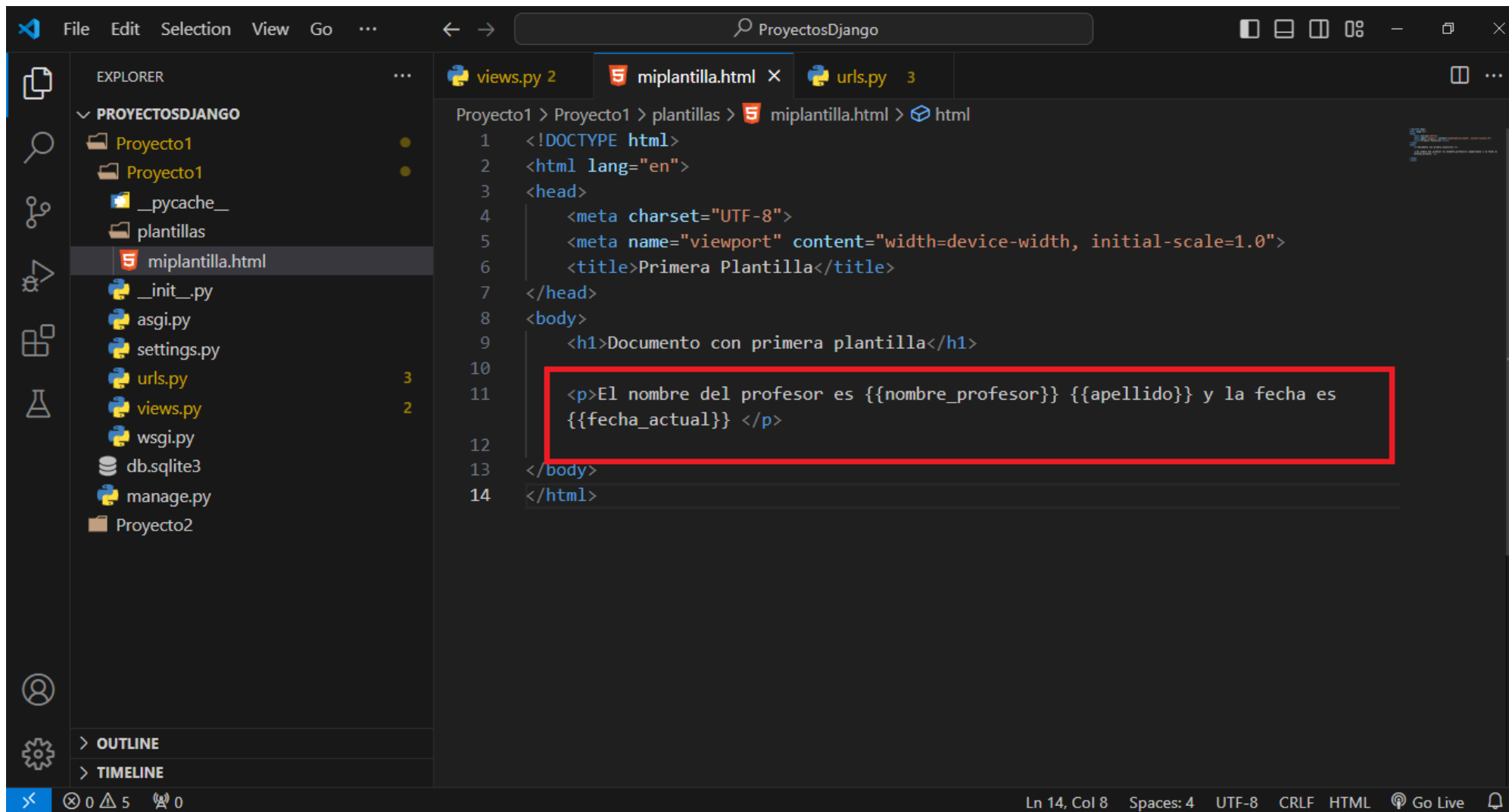


```

4  from django.template import Template, Context
5
6  def saludo(request):
7      nombre="Omar"
8      apellido = "Diaz"
9      fecha_actual = datetime.datetime.now()
10
11
12     doc_externo=open("C:/Users/SENA/Documents/ProyectosDjango/Proyecto1/Proyecto1/
13     plantillas/miplantilla.html")
14
15     plantilla = Template(doc_externo.read())
16
17     doc_externo.close()
18
19     contexto= Context({
20         "nombre_profesor":nombre,
21         'apellido': apellido,
22         'fecha_actual': fecha_actual
23     })
24
25     documento=plantilla.render(contexto)
26
27     return HttpResponse(documento)
28
29 def despedida(request): #Otra vista

```

Modificamos Plantilla

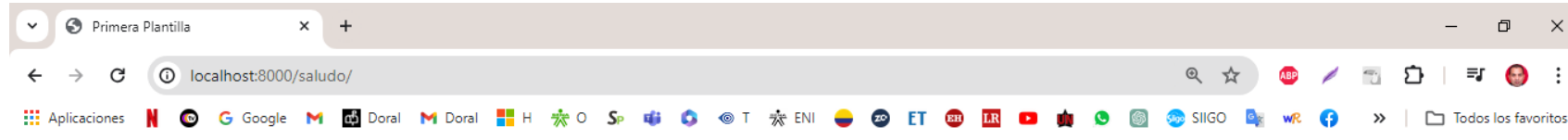


The screenshot shows the Visual Studio Code editor interface. The Explorer sidebar on the left displays the project structure for 'PROYECTOSDJANGO', including folders 'Proyecto1' and 'plantillas', and files like 'miplantilla.html'. The main editor window shows the 'miplantilla.html' file with the following HTML code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Primera Plantilla</title>
7 </head>
8 <body>
9   <h1>Documento con primera plantilla</h1>
10
11   <p>El nombre del profesor es {{nombre_profesor}} {{apellido}} y la fecha es
12     {{fecha_actual}} </p>
13 </body>
14 </html>
```

A red rectangular box highlights the paragraph template on line 11: `<p>El nombre del profesor es {{nombre_profesor}} {{apellido}} y la fecha es {{fecha_actual}} </p>`. The status bar at the bottom indicates the cursor is at line 14, column 8.

Comprobamos en el Navegador



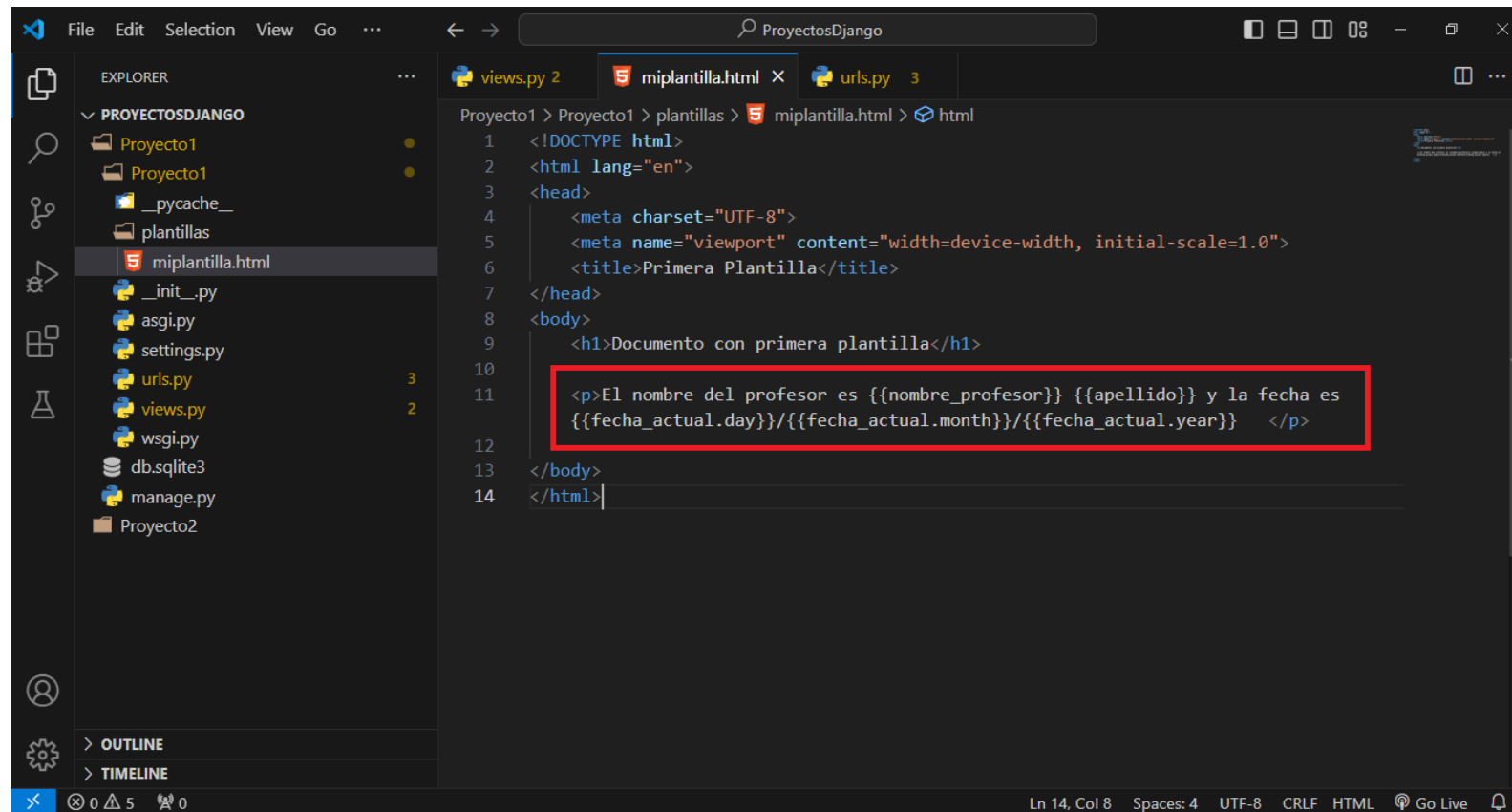
Documento con primera plantilla

El nombre del profesor es Omar Díaz y la fecha es April 17, 2024, 11:50 a.m.



Plantillas Variables

- Puedo acceder a las propiedades de los objetos con la nomenclatura del punto.

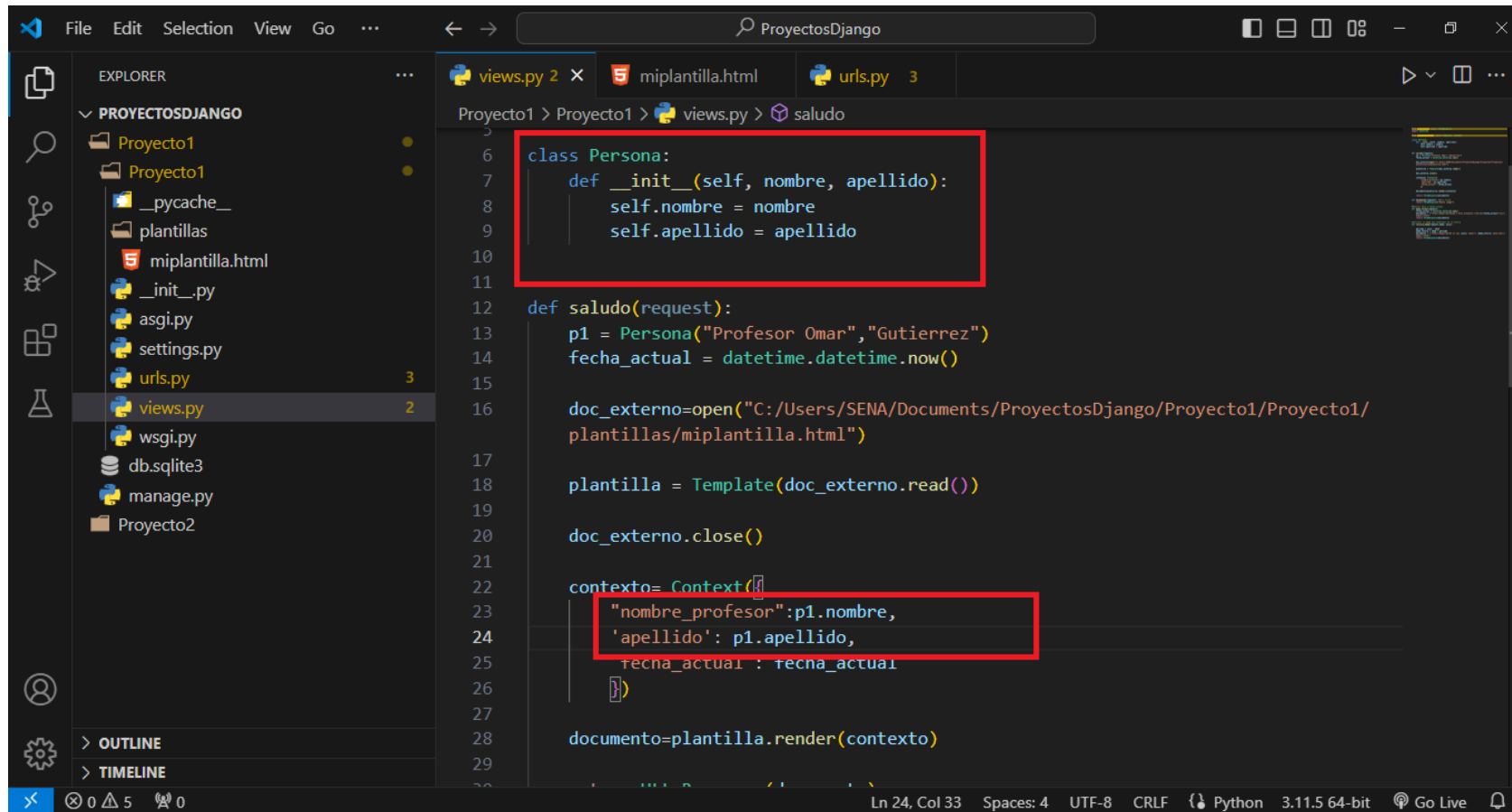


```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Primera Plantilla</title>
7 </head>
8 <body>
9   <h1>Documento con primera plantilla</h1>
10
11   <p>El nombre del profesor es {{nombre_profesor}} {{apellido}} y la fecha es
12     {{fecha_actual.day}}/{{fecha_actual.month}}/{{fecha_actual.year}} </p>
13 </body>
14 </html>
  
```


Clases en las vistas

- Podemos crear Clases en las vistas.



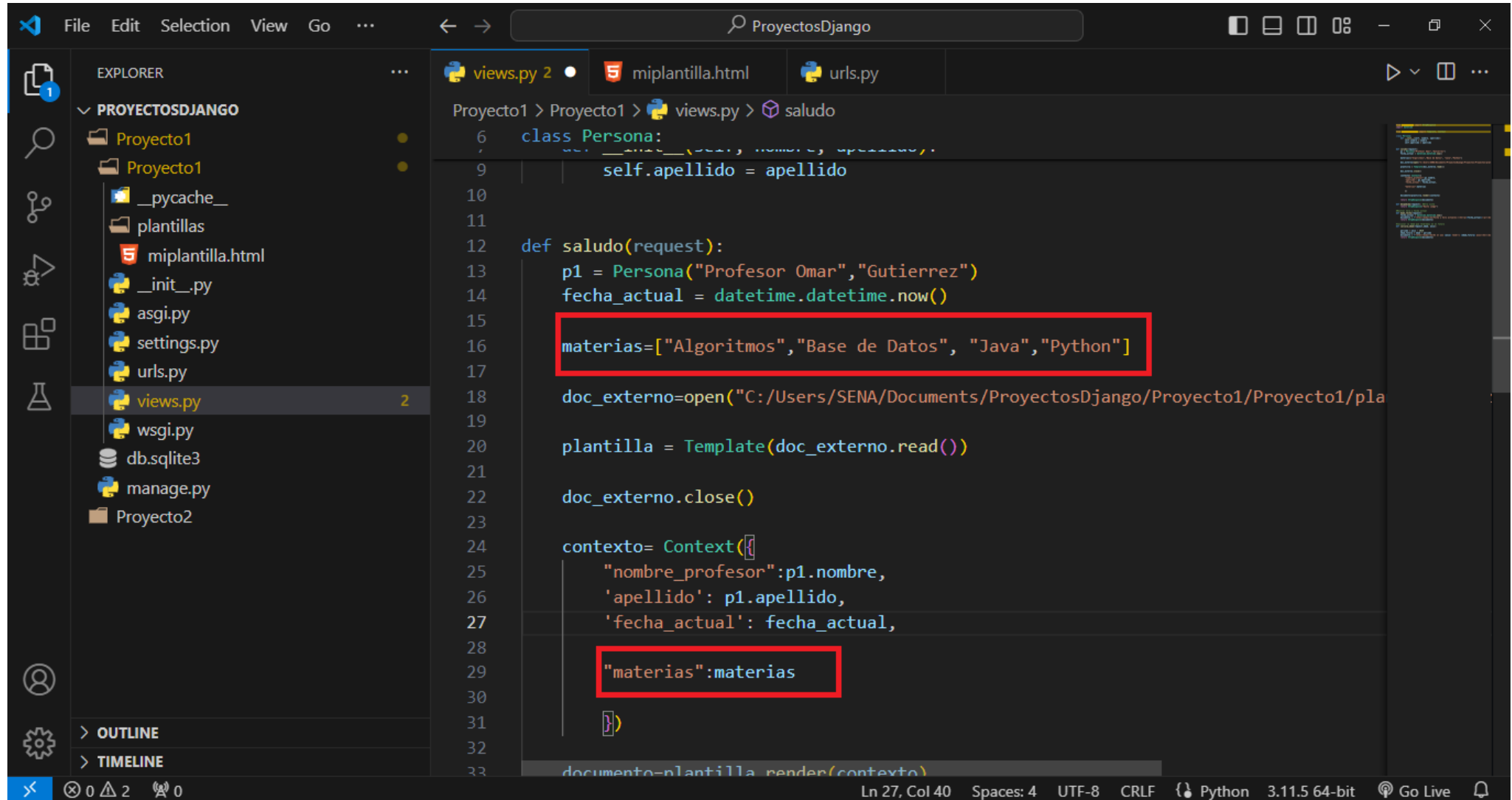
```

File Edit Selection View Go ...
ProjectosDjango
EXPLORER
PROJECTOSDJANGO
  Proyecto1
    Proyecto1
      __pycache__
      plantillas
      miplantilla.html
      __init__.py
      asgi.py
      settings.py
      urls.py 3
      views.py 2
      wsgi.py
      db.sqlite3
      manage.py
    Proyecto2
  > OUTLINE
  > TIMELINE
views.py 2 x miplantilla.html urls.py 3
Projecto1 > Proyecto1 > views.py > saludo
6 class Persona:
7     def __init__(self, nombre, apellido):
8         self.nombre = nombre
9         self.apellido = apellido
10
11
12 def saludo(request):
13     p1 = Persona("Profesor Omar", "Gutierrez")
14     fecha_actual = datetime.datetime.now()
15
16     doc_externo=open("C:/Users/SENA/Documents/ProyectosDjango/Proyecto1/Proyecto1/
17     plantillas/miplantilla.html")
18
19     plantilla = Template(doc_externo.read())
20
21     doc_externo.close()
22
23     contexto= Context({
24         "nombre_profesor":p1.nombre,
25         'apellido': p1.apellido,
26         fecha_actual : fecha_actual
27     })
28
29     documento=plantilla.render(contexto)
  
```

Plantillas Bucles y condicionales

- Llamar a métodos desde plantillas y jerarquía u orden que existe en llamadas a plantillas
- Usar listas en contexto y en las plantillas
- Estructuras de control de flujo en plantillas : bucles y condicionales

Modificamos la vista Saludo para incluir una lista



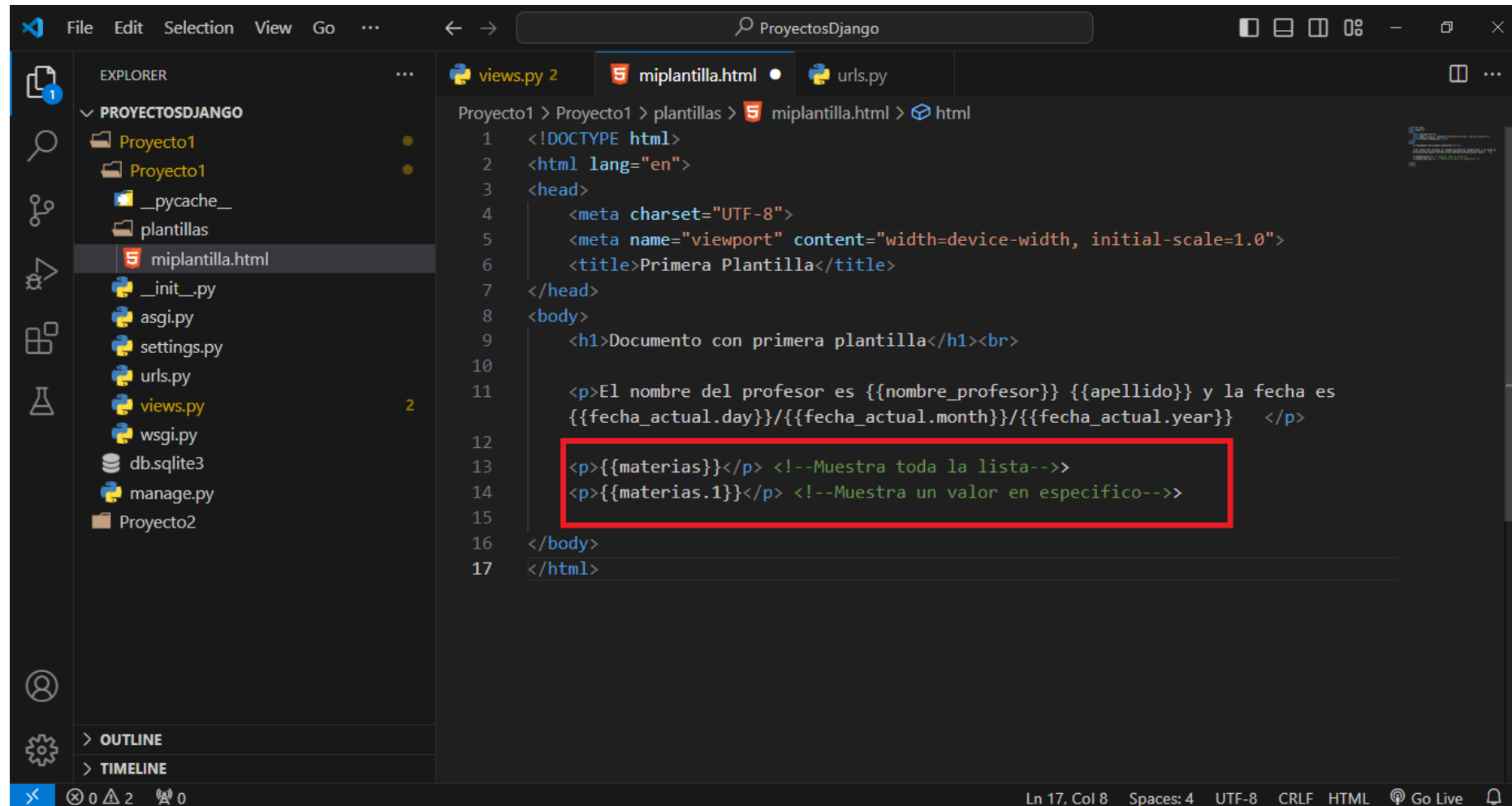
The screenshot shows the Visual Studio Code editor with the Django project 'ProyectosDjango' open. The file explorer on the left shows the project structure, including 'Projecto1' and 'Projecto2'. The main editor window displays the 'views.py' file, specifically the 'saludo' view function. The code is as follows:

```

6 class Persona:
7     def __init__(self, nombre, apellido):
8         self.nombre = nombre
9         self.apellido = apellido
10
11
12 def saludo(request):
13     p1 = Persona("Profesor Omar", "Gutierrez")
14     fecha_actual = datetime.datetime.now()
15
16     materias=["Algoritmos", "Base de Datos", "Java", "Python"]
17
18     doc_externo=open("C:/Users/SENA/Documents/ProyectosDjango/Projecto1/Projecto1/pla
19
20     plantilla = Template(doc_externo.read())
21
22     doc_externo.close()
23
24     contexto= Context({
25         "nombre_profesor":p1.nombre,
26         'apellido': p1.apellido,
27         'fecha_actual': fecha_actual,
28
29         "materias":materias
30     })
31
32     documento=plantilla.render(contexto)
33 
```

Two red boxes highlight the changes made to the view: the first box highlights the new list of subjects (`materias=["Algoritmos", "Base de Datos", "Java", "Python"]`) and the second box highlights the addition of the `"materias":materias` entry to the context dictionary.

Modificamos la plantilla para utilizar la lista



```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Primera Plantilla</title>
7 </head>
8 <body>
9     <h1>Documento con primera plantilla</h1><br>
10
11     <p>El nombre del profesor es {{nombre_profesor}} {{apellido}} y la fecha es
12     {{fecha_actual.day}}/{{fecha_actual.month}}/{{fecha_actual.year}} </p>
13     <p>{{materias}}</p> <!--Muestra toda la lista-->
14     <p>{{materias.1}}</p> <!--Muestra un valor en especifico-->
15
16 </body>
17 </html>
    
```

Jerarquía de llamadas en el punto (.) en las plantillas

La jerarquía de llamadas en plantillas de Django sigue este orden:

- Busca si la llamada corresponde a un elemento de un **diccionario**.
- Si no corresponde a un diccionario, verifica si es un **atributo de una clase**.
- Si tampoco es un atributo, comprueba si es un **método de un objeto**.
- En último lugar, busca si la llamada está accediendo a un **índice de una lista**.

Bucle for en plantillas Django

Los bucles for permiten recorrer listas y otros iterables en las plantillas de Django.

{% for elemento in Iterable %}

<!-- Haz algo con cada elemento -->

{{ elemento }}

{% endfor %}

En esta sintaxis:

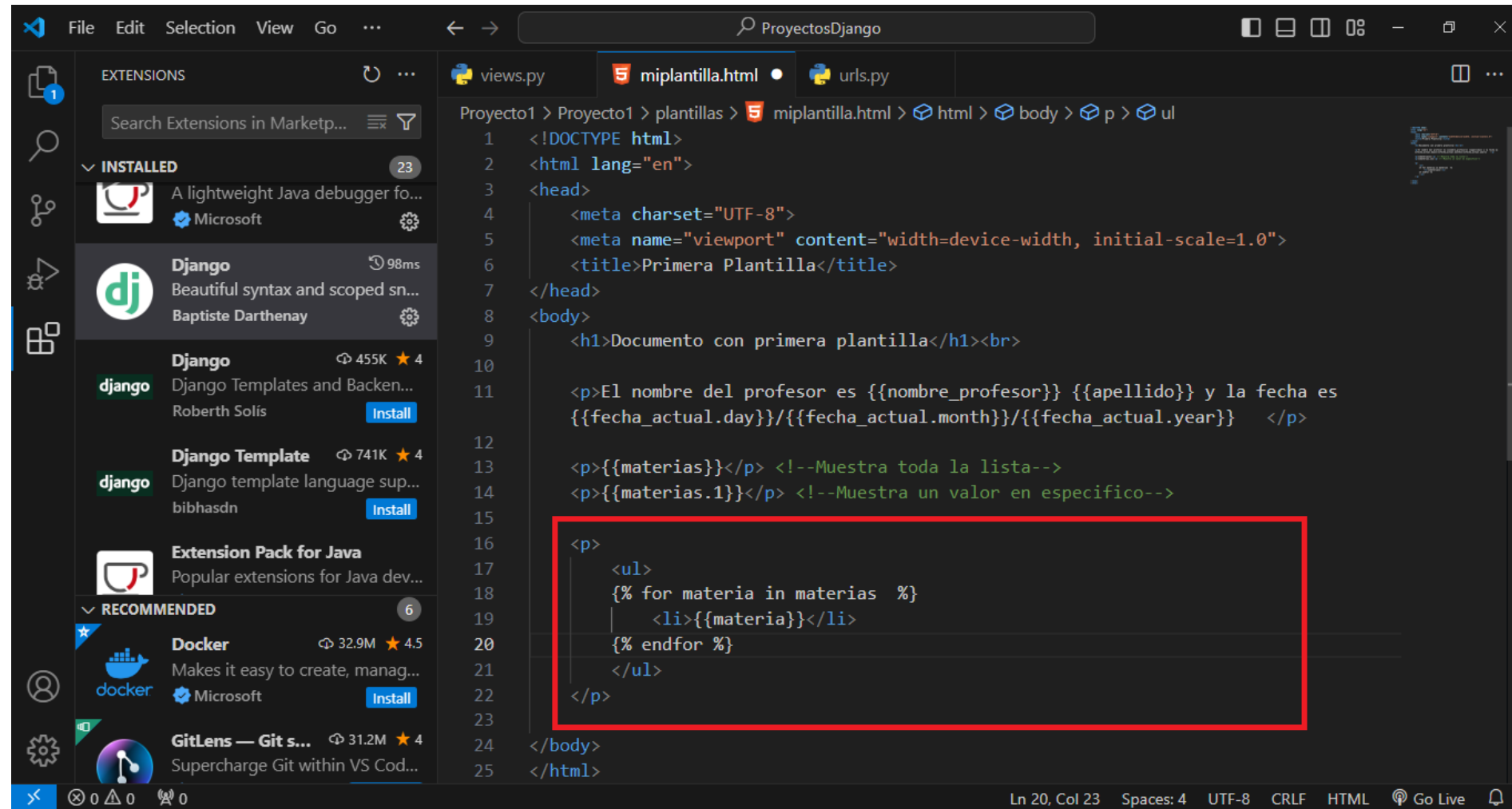
{% for elemento in iterable %}: Inicia el bucle, donde elemento es una variable que representa cada elemento del iterable.

<!-- Haz algo con cada elemento -->: Aquí colocas el código HTML o las etiquetas de Django que deseas repetir para cada elemento.

{{ elemento }}: Muestra el valor del elemento en el HTML. Esto puede ser cualquier dato, como texto, números o propiedades de objetos.

{% endfor %}: Finaliza el bucle.

Modificamos la plantilla para utilizar el for



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Primera Plantilla</title>
7 </head>
8 <body>
9   <h1>Documento con primera plantilla</h1><br>
10
11   <p>El nombre del profesor es {{nombre_profesor}} {{apellido}} y la fecha es
12     {{fecha_actual.day}}/{{fecha_actual.month}}/{{fecha_actual.year}} </p>
13
14   <p>{{materias}}</p> <!--Muestra toda la lista-->
15   <p>{{materias.1}}</p> <!--Muestra un valor en especifico-->
16
17   <p>
18     <ul>
19       {% for materia in materias %}
20       <li>{{materia}}</li>
21       {% endfor %}
22     </ul>
23   </p>
24 </body>
25 </html>
```

Bucle if en plantillas Django

En Django, puedes utilizar la estructura condicional `{% if %}` en tus plantillas HTML para mostrar contenido basado en ciertas condiciones. Sintaxis

```
{% if condicion %}
```

```
<!-- Haz algo si la condición es verdadera -->
```

```
{% elif otra_condicion %}
```

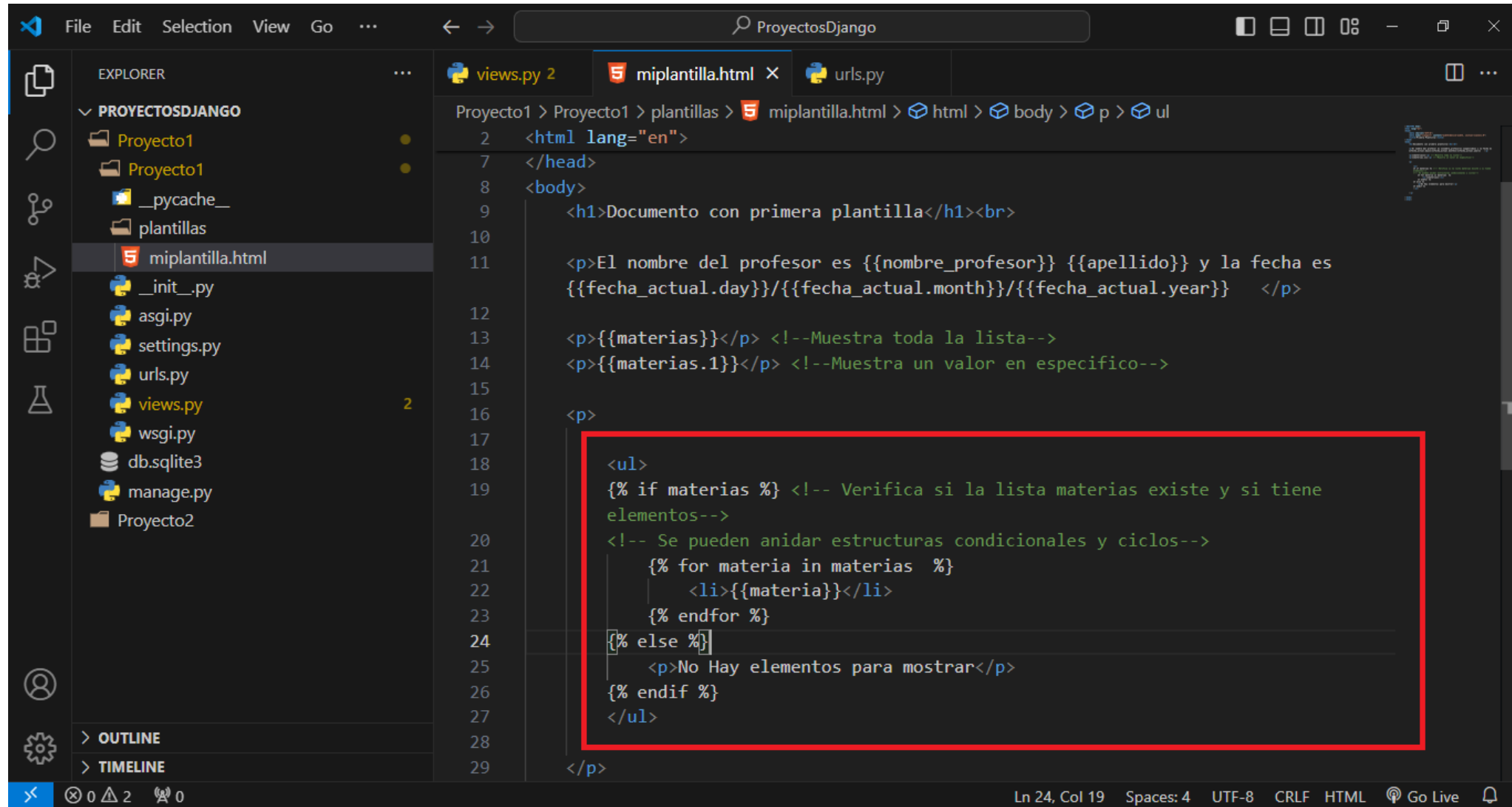
```
<!-- Haz algo si la otra condición es verdadera -->
```

```
{% else %}
```

```
<!-- Haz algo si ninguna de las condiciones anteriores es verdadera -->
```

```
{% endif %}
```


Modificamos la plantilla para utilizar el if

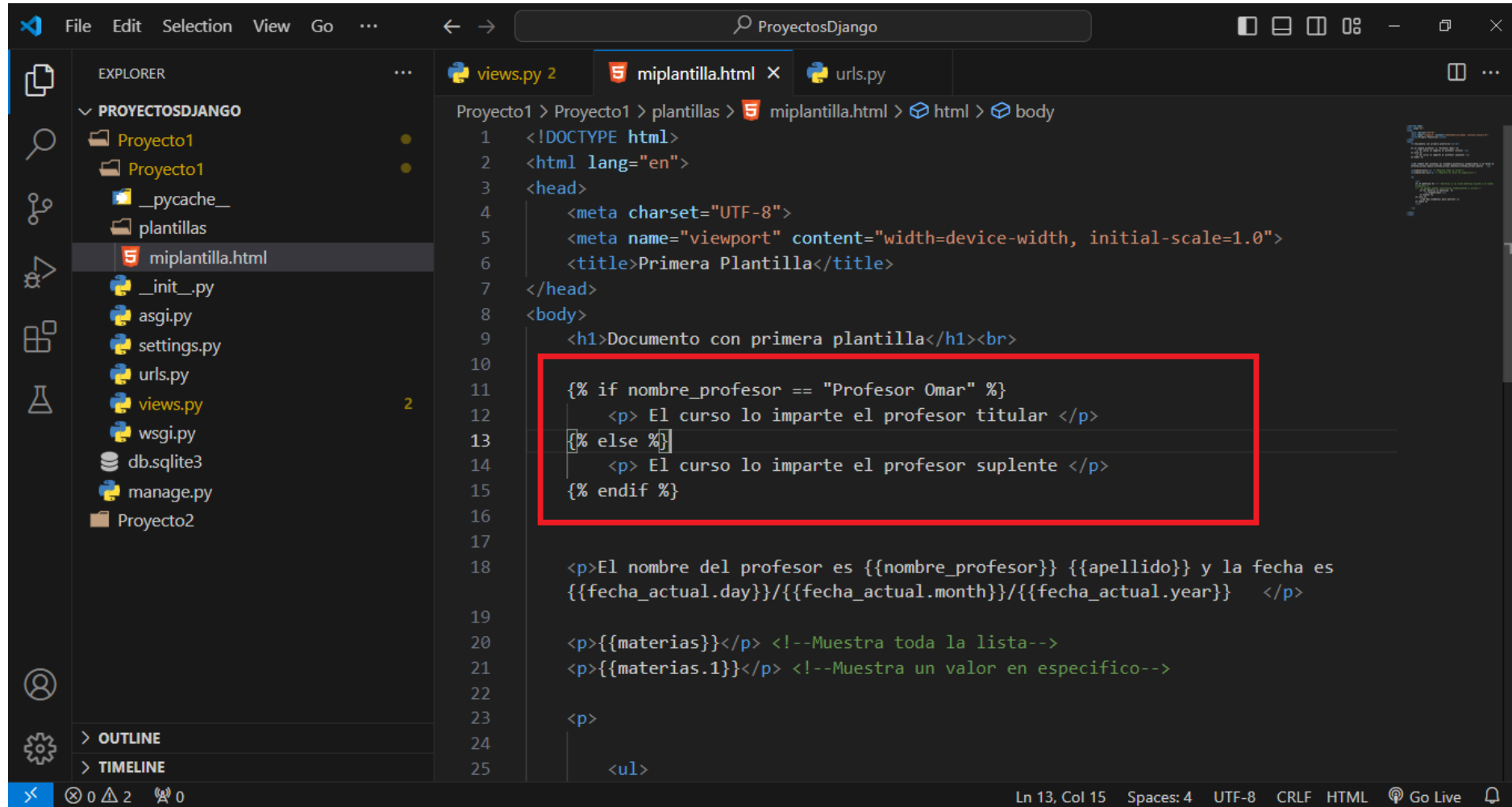


```

2  <html lang="en">
7  </head>
8  <body>
9      <h1>Documento con primera plantilla</h1><br>
10
11     <p>El nombre del profesor es {{nombre_profesor}} {{apellido}} y la fecha es
12         {{fecha_actual.day}}/{{fecha_actual.month}}/{{fecha_actual.year}}    </p>
13
14     <p>{{materias}}</p> <!--Muestra toda la lista-->
15     <p>{{materias.1}}</p> <!--Muestra un valor en especifico-->
16
17     <p>
18         <ul>
19             {% if materias %} <!-- Verifica si la lista materias existe y si tiene
20                 elementos-->
21                 <!-- Se pueden anidar estructuras condicionales y ciclos-->
22                 {% for materia in materias %}
23                     <li>{{materia}}</li>
24                 {% endfor %}
25             {% else %}
26                 <p>No Hay elementos para mostrar</p>
27             {% endif %}
28         </ul>
29     </p>

```

Modificamos la plantilla para utilizar el if con operadores de comparación



The screenshot shows the Visual Studio Code editor with the Django project 'ProyectosDjango' open. The file explorer on the left shows the project structure, including 'Proyecto1' and 'plantillas'. The main editor window displays the 'miplantilla.html' file. The code in the file is as follows:

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Primera Plantilla</title>
7 </head>
8 <body>
9     <h1>Documento con primera plantilla</h1><br>
10
11     {% if nombre_profesor == "Profesor Omar" %}
12         <p> El curso lo imparte el profesor titular </p>
13     {% else %}
14         <p> El curso lo imparte el profesor suplente </p>
15     {% endif %}
16
17
18     <p>El nombre del profesor es {{nombre_profesor}} {{apellido}} y la fecha es
19     {{fecha_actual.day}}/{{fecha_actual.month}}/{{fecha_actual.year}} </p>
20
21     <p>{{materias}}</p> <!--Muestra toda la lista-->
22     <p>{{materias.1}}</p> <!--Muestra un valor en especifico-->
23
24     <p>
25         <ul>

```

The if statement block (lines 11-15) is highlighted with a red box, indicating the modification being made to use comparison operators.

Comprobamos en el Navegador

Documento con primera plantilla

El curso lo imparte el profesor titular

El nombre del profesor es Profesor Omar Gutierrez y la fecha es 19/4/2024

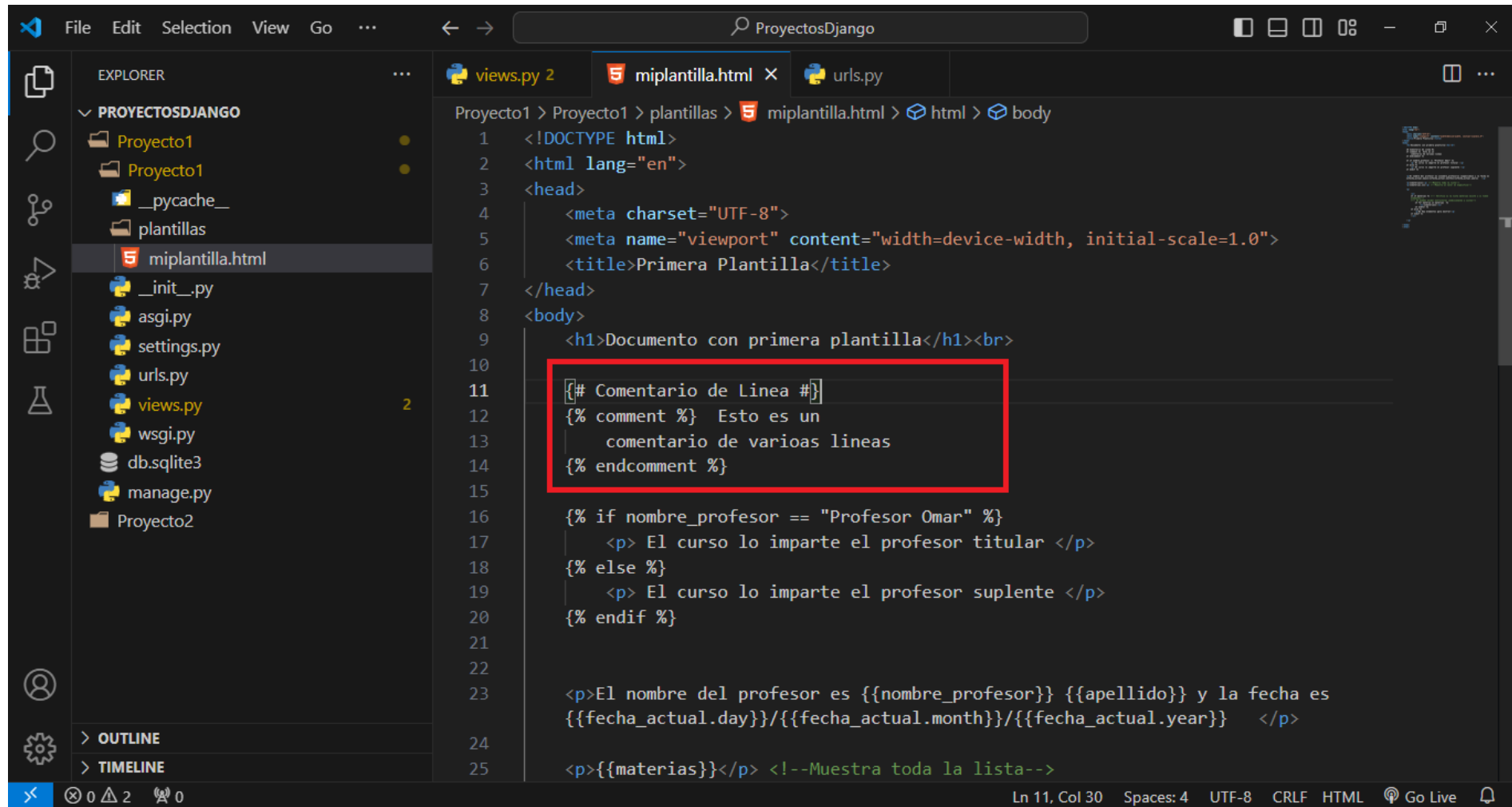
[]

No Hay elementos para mostrar

Plantillas Comentarios, Filtros y Cargadores

- Comentarios en plantillas
- Filtros en las plantillas y cómo utilizarlos
- Cargadores de plantillas

Modificamos la plantilla para introducir Comentarios



The screenshot shows the Visual Studio Code editor with the Django project 'ProyectosDjango' open. The file explorer on the left shows the project structure, with 'miplantilla.html' selected under the 'plantillas' directory. The main editor window displays the content of 'miplantilla.html', which is an HTML template. A red box highlights the addition of a multi-line comment block between lines 10 and 14. The comment block is enclosed in curly braces and uses Django's comment syntax: `{% comment %}` for the start, `{% endcomment %}` for the end, and `{% comment %}` for the content. The content of the comment is 'Esto es un comentario de varias lineas'. The rest of the template code remains unchanged, including the DOCTYPE, head, body, and various Django template tags for displaying professor information and a list of subjects.

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Primera Plantilla</title>
7 </head>
8 <body>
9     <h1>Documento con primera plantilla</h1><br>
10
11     {% comment %}
12     {% comment %} Esto es un
13     comentario de varias lineas
14     {% endcomment %}
15
16     {% if nombre_profesor == "Profesor Omar" %}
17         <p> El curso lo imparte el profesor titular </p>
18     {% else %}
19         <p> El curso lo imparte el profesor suplente </p>
20     {% endif %}
21
22
23     <p>El nombre del profesor es {{nombre_profesor}} {{apellido}} y la fecha es
24     {{fecha_actual.day}}/{{fecha_actual.month}}/{{fecha_actual.year}} </p>
25
26     <p>{{materias}}</p> <!--Muestra toda la lista-->
  
```

Filtros en Plantillas

En las plantillas de Django, los filtros son funciones que permiten modificar el contenido de las variables antes de mostrarlo en la página web. Ayudan a formatear datos, realizar operaciones simples y mostrar información de manera más legible. Algunos filtros comunes en Django son:

1. **{{ variable | filtro }}**: En una plantilla de Django, puedes aplicar un filtro a una variable utilizando el símbolo "|" (pipe) seguido del nombre del filtro.
2. **Filtros de texto:**
 1. **{{ texto | lower }}**: Convierte el texto a minúsculas.
 2. **{{ texto | upper }}**: Convierte el texto a mayúsculas.
 3. **{{ texto | title }}**: Convierte el primer carácter de cada palabra a mayúsculas.
3. **Filtros numéricos:**
 1. **{{ numero | intcomma }}**: Agrega comas como separadores de miles en números enteros.
 2. **{{ numero | floatformat }}**: Limita el número de decimales en números de punto flotante.

Filtros en Plantillas

4. Filtros de fecha y hora:

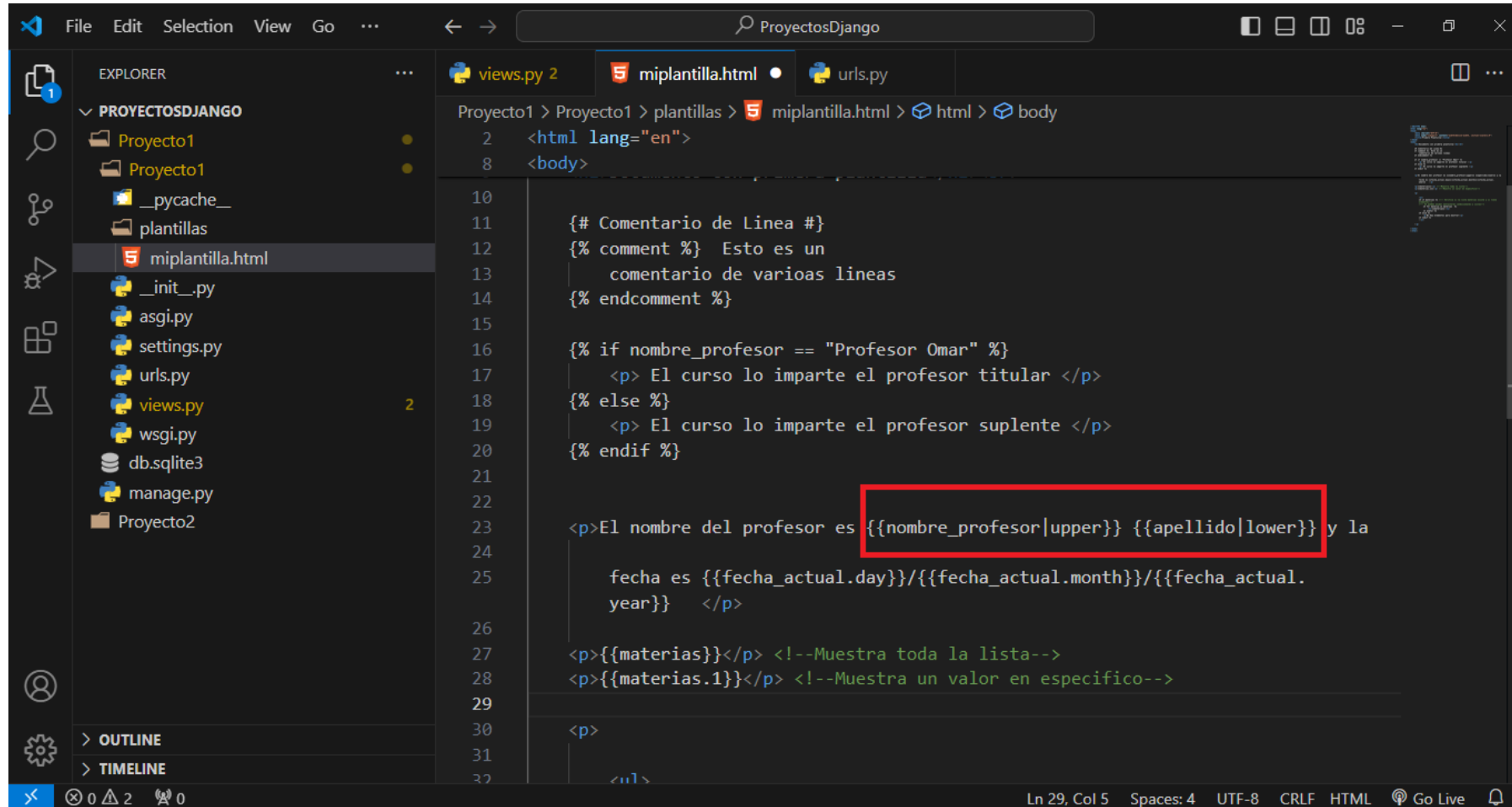
1. `{{ fecha | date:"D d M Y" }}`: Formatea una fecha según el formato especificado.
2. `{{ hora | time:"h:i a" }}`: Formatea una hora según el formato especificado.

5. Filtros de lista y diccionario:

1. `{{ lista | join:", " }}`: Une los elementos de una lista con el separador especificado.
2. `{{ diccionario | dictsort }}`: Ordena un diccionario por las claves.

6. Filtros personalizados: También puedes crear tus propios filtros personalizados en Django para realizar operaciones específicas según tus necesidades.

Modificamos la plantilla para introducir filtros



```

ProjectosDjango
EXPLORER
  PROJECTOSDJANGO
    Proyecto1
    Proyecto1
      __pycache__
      plantillas
        miplantilla.html
      __init__.py
      asgi.py
      settings.py
      urls.py
      views.py
      wsgi.py
      db.sqlite3
      manage.py
    Proyecto2
  OUTLINE
  TIMELINE
  views.py 2
  miplantilla.html
  urls.py
Projecto1 > Proyecto1 > plantillas > miplantilla.html > html > body
2  <html lang="en">
8  <body>
10
11  {# Comentario de Linea #}
12  {% comment %} Esto es un
13     comentario de varias lineas
14  {% endcomment %}
15
16  {% if nombre_profesor == "Profesor Omar" %}
17      <p> El curso lo imparte el profesor titular </p>
18  {% else %}
19      <p> El curso lo imparte el profesor suplente </p>
20  {% endif %}
21
22
23  <p>El nombre del profesor es {{nombre_profesor|upper}} {{apellido|lower}} y la
24
25      fecha es {{fecha_actual.day}}/{{fecha_actual.month}}/{{fecha_actual.
26          year}} </p>
27
28  <p>{{materias}}</p> <!--Muestra toda la lista-->
29  <p>{{materias.1}}</p> <!--Muestra un valor en especifico-->
30
31  <p>
32

```


Comprobamos en el Navegador

Documento con primera plantilla

El curso lo imparte el profesor titular

El nombre del profesor es PROFESOR OMAR gutierrez y la fecha es 19/4/2024

[]

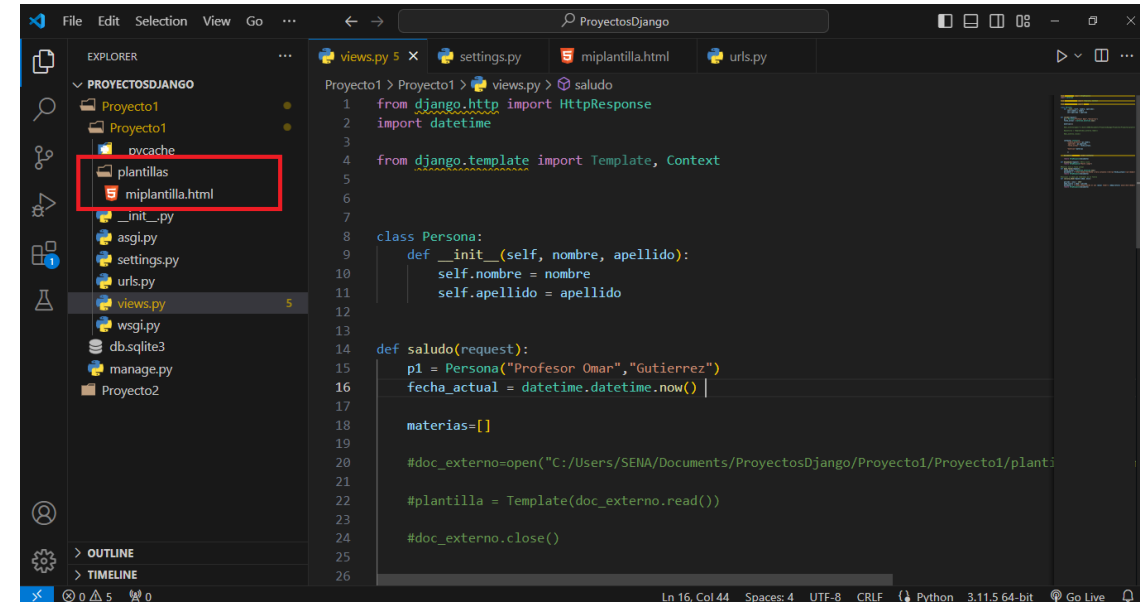
No Hay elementos para mostrar

Cargadores Django

En el contexto de Django, los "cargadores" (loaders en inglés) son componentes que se utilizan para cargar recursos externos o datos en una aplicación Django. Un ejemplo común es el uso de cargadores de plantillas para cargar plantillas desde ubicaciones especificadas en el archivo **settings.py** en la lista **DIRS.bb**

Pasos

Paso 1: Crea tus plantillas HTML dentro del directorio seleccionado. Por ejemplo, puedes crear un archivo llamado `miplantilla.html` dentro del directorio `templates` o `plantillas`.

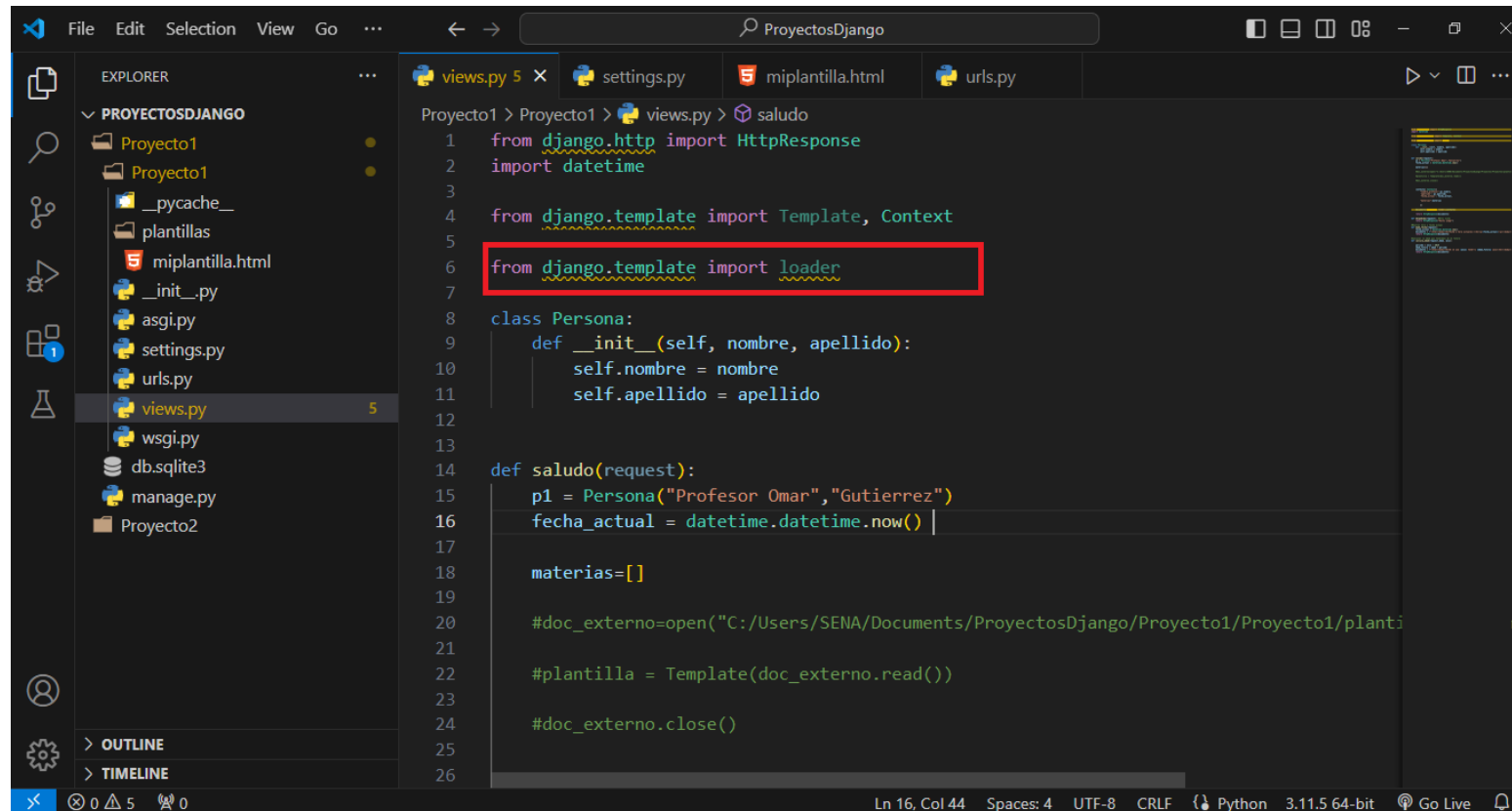


The screenshot shows a code editor with a Django project structure on the left and a view function in `views.py` on the right. The project structure includes a `PROJECTOSDJANGO` directory with subdirectories `Projecto1` and `Projecto2`. Inside `Projecto1`, there is a `plantillas` directory (highlighted with a red box) containing `miplantilla.html`. The `views.py` file shows a `saludo` function that creates a `Persona` object, sets the current date, and reads a template from a file.

```
1 from django.http import HttpResponse
2 import datetime
3
4 from django.template import Template, Context
5
6
7
8 class Persona:
9     def __init__(self, nombre, apellido):
10         self.nombre = nombre
11         self.apellido = apellido
12
13
14 def saludo(request):
15     p1 = Persona("Profesor Omar", "Gutierrez")
16     fecha_actual = datetime.datetime.now()
17
18     materias=[]
19
20     #doc_externo=open("C:/Users/SENA/Documents/ProyectosDjango/Projecto1/Projecto1/plantillas/miplantilla.html")
21
22     #plantilla = Template(doc_externo.read())
23
24     #doc_externo.close()
```

Cargadores Django

Paso 2: Define una vista en tu aplicación Django que renderice una plantilla. Por ejemplo, en el archivo `views.py` de tu aplicación, debe ser importado : **`from django.template import loader`**

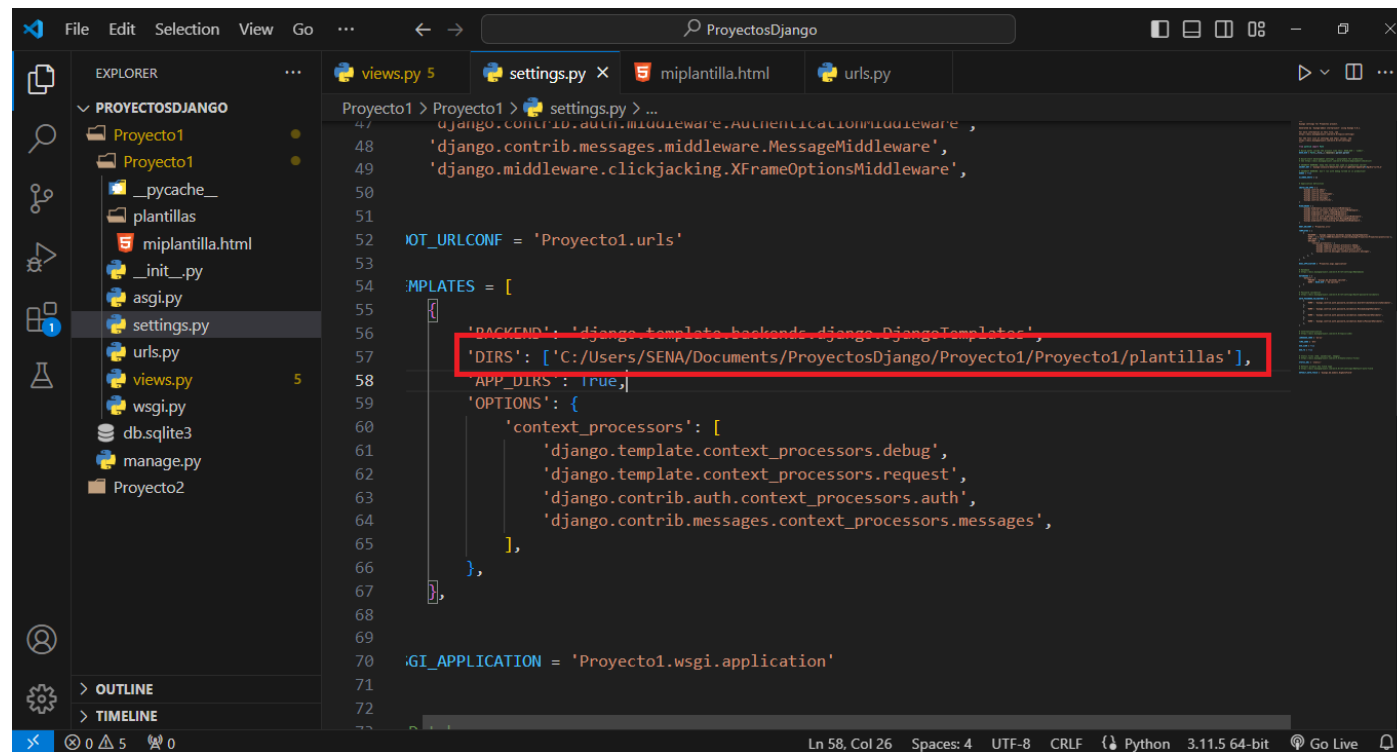


```

ProjectosDjango
views.py x settings.py miplantilla.html urls.py
Projecto1 > Proyecto1 > views.py > saludo
1 from django.http import HttpResponse
2 import datetime
3
4 from django.template import Template, Context
5
6 from django.template import loader
7
8 class Persona:
9     def __init__(self, nombre, apellido):
10         self.nombre = nombre
11         self.apellido = apellido
12
13
14 def saludo(request):
15     p1 = Persona("Profesor Omar", "Gutierrez")
16     fecha_actual = datetime.datetime.now()
17
18     materias=[]
19
20     #doc_externo=open("C:/Users/SENA/Documents/ProyectosDjango/Proyecto1/Proyecto1/planti
21
22     #plantilla = Template(doc_externo.read())
23
24     #doc_externo.close()
25
26
  
```

Cargadores Django

Paso 3: Configurar los directorios de plantillas en la configuración del proyecto Django. Abre el archivo settings.py y asegúrate de que la lista DIRS en la configuración de TEMPLATES incluya la ruta al directorio donde se encuentran tus plantillas.



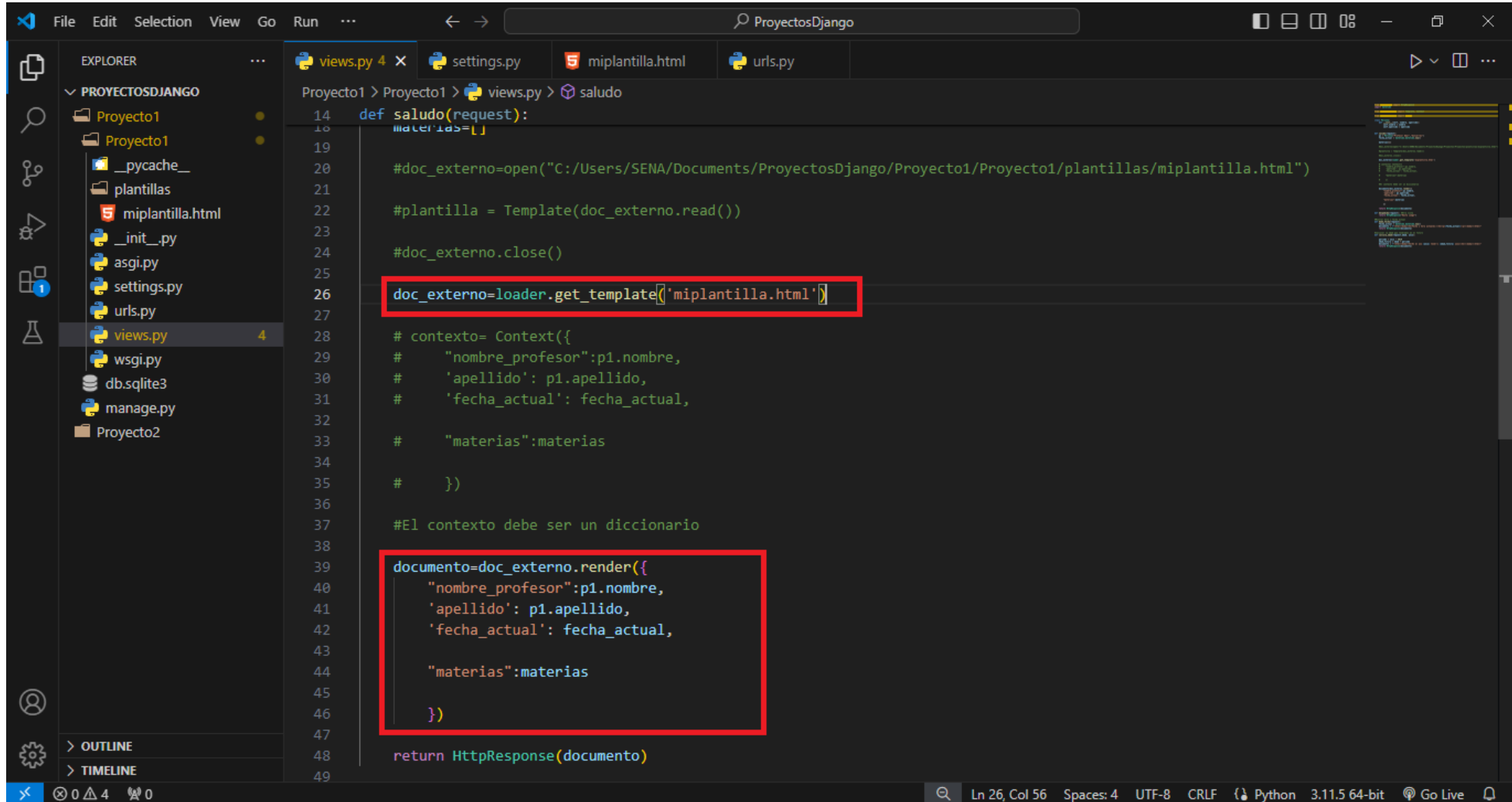
```

47 django.contrib.auth.middleware.AuthenticationMiddleware,
48 django.contrib.messages.middleware.MessageMiddleware',
49 'django.middleware.clickjacking.XFrameOptionsMiddleware',
50
51
52 IOT_URLCONF = 'Proyecto1.urls'
53
54 TEMPLATES = [
55     {
56         'BACKEND': 'django.template.backends.django.DjangoTemplates',
57         'DIRS': ['C:/Users/SENA/Documents/ProyectosDjango/Proyecto1/Proyecto1/plantillas'],
58         'APP_DIRS': True,
59         'OPTIONS': {
60             'context_processors': [
61                 'django.template.context_processors.debug',
62                 'django.template.context_processors.request',
63                 'django.contrib.auth.context_processors.auth',
64                 'django.contrib.messages.context_processors.messages',
65             ],
66         },
67     },
68 ]
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925

```

Cargadores Django

Paso 4: Modifica el archivo views.py para ejecutar el loader



The screenshot shows a code editor with the following files open: views.py, settings.py, miplantilla.html, and urls.py. The Explorer panel on the left shows the project structure for 'PROYECTOSDJANGO', including 'Proyecto1' and 'Proyecto2'. The 'views.py' file is selected, and the code is as follows:

```

14 def saludo(request):
15     materias=[]
16
17
18
19
20     #doc_externo=open("C:/Users/SENA/Documents/ProyectosDjango/Proyecto1/Proyecto1/plantillas/miplantilla.html")
21
22     #plantilla = Template(doc_externo.read())
23
24     #doc_externo.close()
25
26     doc_externo=loader.get_template('miplantilla.html')
27
28     # contexto= Context({
29     #     "nombre_profesor":p1.nombre,
30     #     'apellido': p1.apellido,
31     #     'fecha_actual': fecha_actual,
32
33     #     "materias":materias
34
35     # })
36
37     #El contexto debe ser un diccionario
38
39     documento=doc_externo.render({
40         "nombre_profesor":p1.nombre,
41         'apellido': p1.apellido,
42         'fecha_actual': fecha_actual,
43
44         "materias":materias
45     })
46
47
48     return HttpResponse(documento)
49

```

Two red boxes highlight the changes made in the code: the first box highlights the line `doc_externo=loader.get_template('miplantilla.html')` at line 26, and the second box highlights the `documento=doc_externo.render(...)` block from line 39 to line 45.

Plantillas Módulo de Django Shortcuts

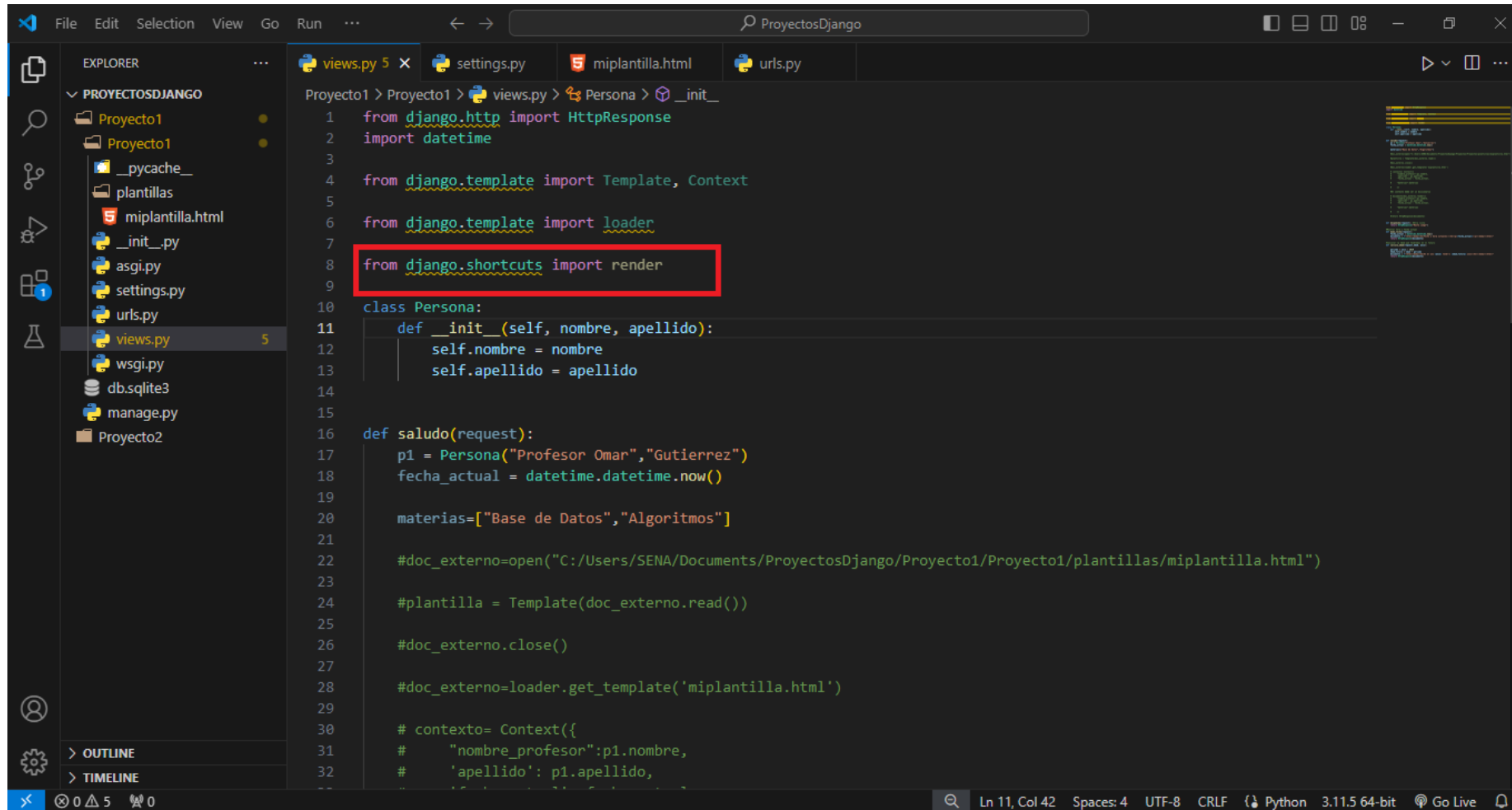
- En Django, los "Shortcuts" son funciones útiles que simplifican tareas comunes en el desarrollo web, como redireccionar a una página, renderizar una plantilla HTML o enviar una respuesta de JSON. El módulo `django.shortcuts` proporciona estas funciones para facilitar el desarrollo y mejorar la legibilidad del código.
- Algunos ejemplos de los shortcuts más comunes que proporciona este módulo:
 - `render()`: Esta función se utiliza para renderizar una plantilla HTML con un contexto y devolver la respuesta HTTP resultante. Es una forma abreviada de combinar la carga de una plantilla y la creación de una respuesta HTTP.

Plantillas Módulo de Django Shortcuts

- En Django, los "Shortcuts" son funciones útiles que simplifican tareas comunes en el desarrollo web, como redireccionar a una página, renderizar una plantilla HTML o enviar una respuesta de JSON. El módulo `django.shortcuts` proporciona estas funciones para facilitar el desarrollo y mejorar la legibilidad del código.
- Uno de los ejemplos de los shortcuts más comunes que proporciona este módulo es:
 - ✓ `render()`: Esta función se utiliza para renderizar una plantilla HTML con un contexto y devolver la respuesta HTTP resultante. Es una forma abreviada de combinar la carga de una plantilla y la creación de una respuesta HTTP. Para utilizarlo debe ser importado en el modulo `views.py` así:

```
from django.shortcuts import render
```

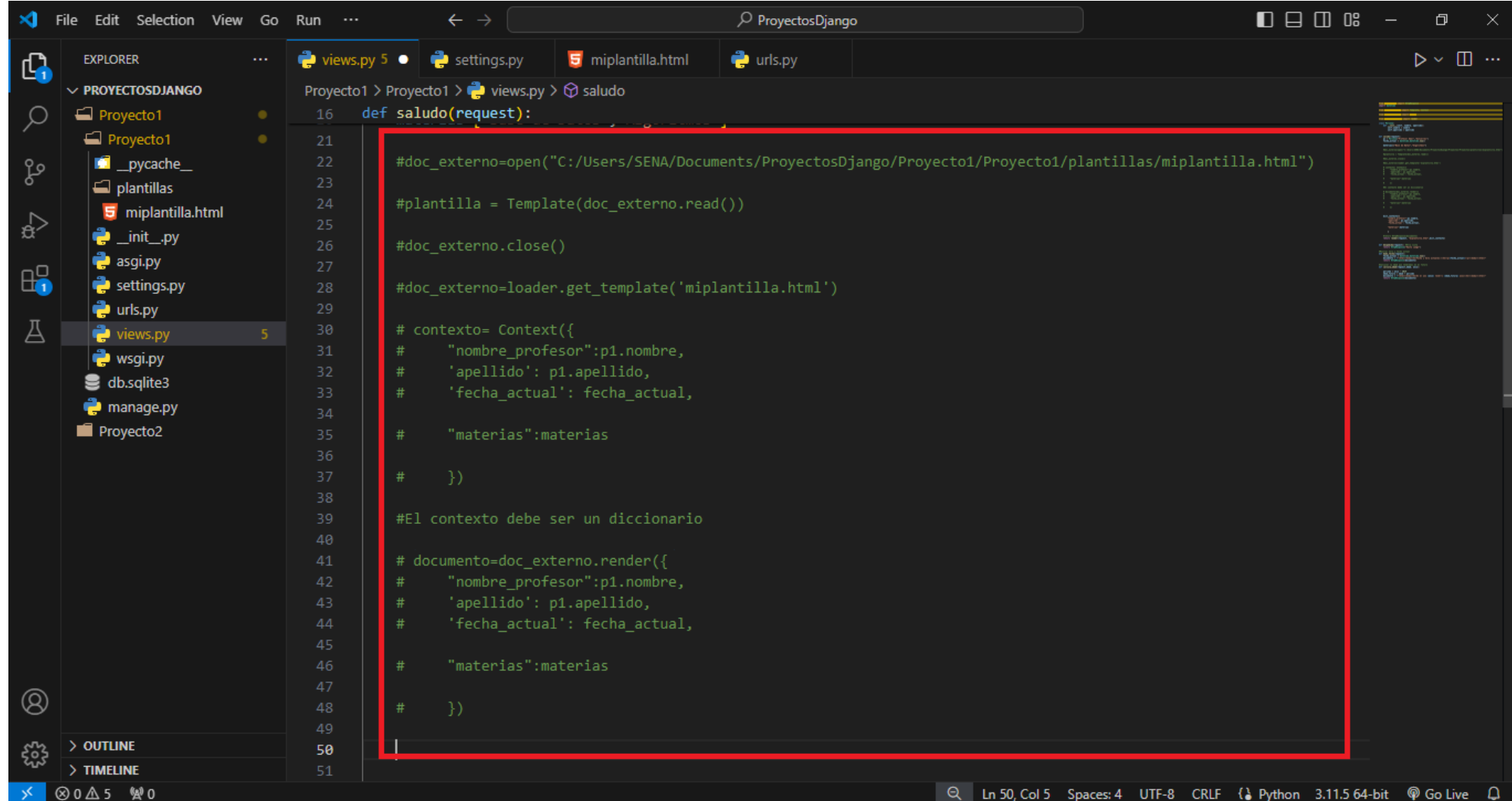
Plantillas Módulo de Django Shortcuts



```
1 from django.http import HttpResponse
2 import datetime
3
4 from django.template import Template, Context
5
6 from django.template import loader
7
8 from django.shortcuts import render
9
10 class Persona:
11     def __init__(self, nombre, apellido):
12         self.nombre = nombre
13         self.apellido = apellido
14
15
16 def saludo(request):
17     p1 = Persona("Profesor Omar", "Gutierrez")
18     fecha_actual = datetime.datetime.now()
19
20     materias=["Base de Datos", "Algoritmos"]
21
22     #doc_externo=open("C:/Users/SENA/Documents/ProyectosDjango/Proyecto1/Proyecto1/plantillas/miplantilla.html")
23
24     #plantilla = Template(doc_externo.read())
25
26     #doc_externo.close()
27
28     #doc_externo=loader.get_template('miplantilla.html')
29
30     # contexto= Context({
31     #     "nombre_profesor":p1.nombre,
32     #     'apellido': p1.apellido,
```


Plantillas Módulo de Django Shortcuts

- Comentamos todo el código anterior

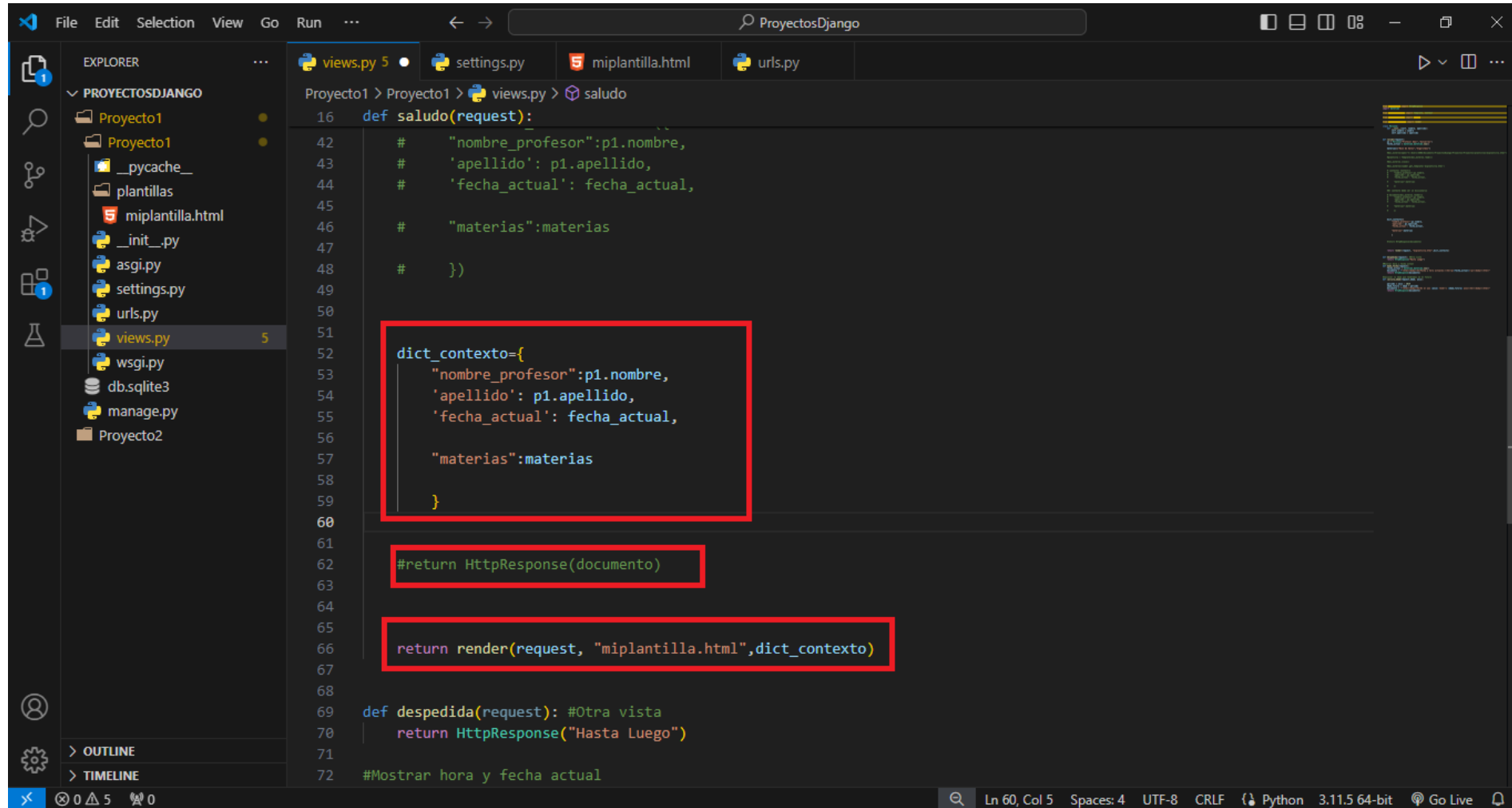


The screenshot shows a code editor with a dark theme. The Explorer panel on the left shows a project structure with folders 'Proyecto1' and 'Proyecto2', and files like 'miplantilla.html', 'views.py', 'urls.py', 'settings.py', 'asgi.py', 'wsgi.py', 'db.sqlite3', and 'manage.py'. The main editor window shows the 'views.py' file with a function 'saludo(request)' starting at line 16. The code is commented out, indicated by a red box around the function body. The comments are in Spanish and describe the process of loading a template and rendering it with context data.

```
16 def saludo(request):
21
22     #doc_externo=open("C:/Users/SENA/Documents/ProyectosDjango/Proyecto1/Proyecto1/plantillas/miplantilla.html")
23
24     #plantilla = Template(doc_externo.read())
25
26     #doc_externo.close()
27
28     #doc_externo=loader.get_template('miplantilla.html')
29
30     # contexto= Context({
31     #     "nombre_profesor":p1.nombre,
32     #     'apellido': p1.apellido,
33     #     'fecha_actual': fecha_actual,
34
35     #     "materias":materias
36
37     # })
38
39     #El contexto debe ser un diccionario
40
41     # documento=doc_externo.render({
42     #     "nombre_profesor":p1.nombre,
43     #     'apellido': p1.apellido,
44     #     'fecha_actual': fecha_actual,
45
46     #     "materias":materias
47
48     # })
49
50
51
```

Plantillas Módulo de Django Shortcuts

- Agregamos el siguiente código



```
16 def saludo(request):
42     # "nombre_profesor": p1.nombre,
43     # 'apellido': p1.apellido,
44     # 'fecha_actual': fecha_actual,
45     # "materias": materias
46     # ))
51
52     dict_contexto={
53         "nombre_profesor": p1.nombre,
54         'apellido': p1.apellido,
55         'fecha_actual': fecha_actual,
56         "materias": materias
57     }
58
59
60
61
62     #return HttpResponseRedirect(documento)
63
64
65
66     return render(request, "miplantilla.html", dict_contexto)
67
68
69 def despedida(request): #Otra vista
70     return HttpResponseRedirect("Hasta Luego")
71
72 #Mostrar hora y fecha actual
```

Plantillas Módulo de Django Shortcuts

Documento con primera plantilla

El curso lo imparte el profesor titular

El nombre del profesor es PROFESOR OMAR gutierrez y la fecha es 3/5/2024

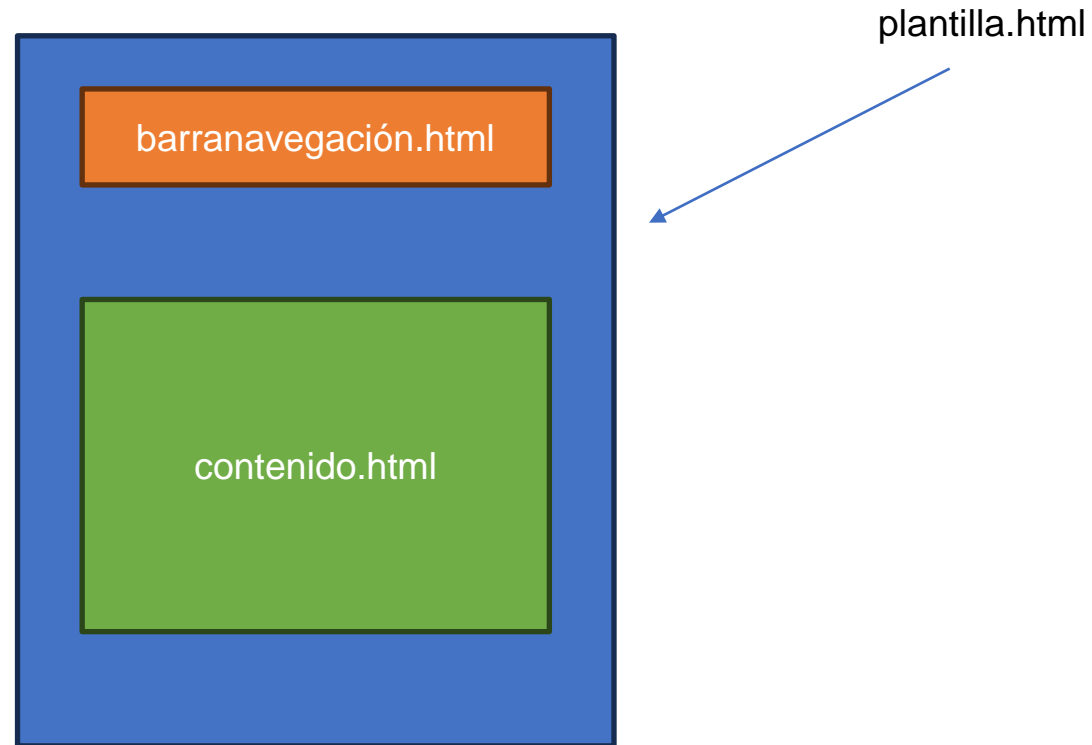
['Base de Datos', 'Algoritmos']

Algoritmos

- Base de Datos
- Algoritmos

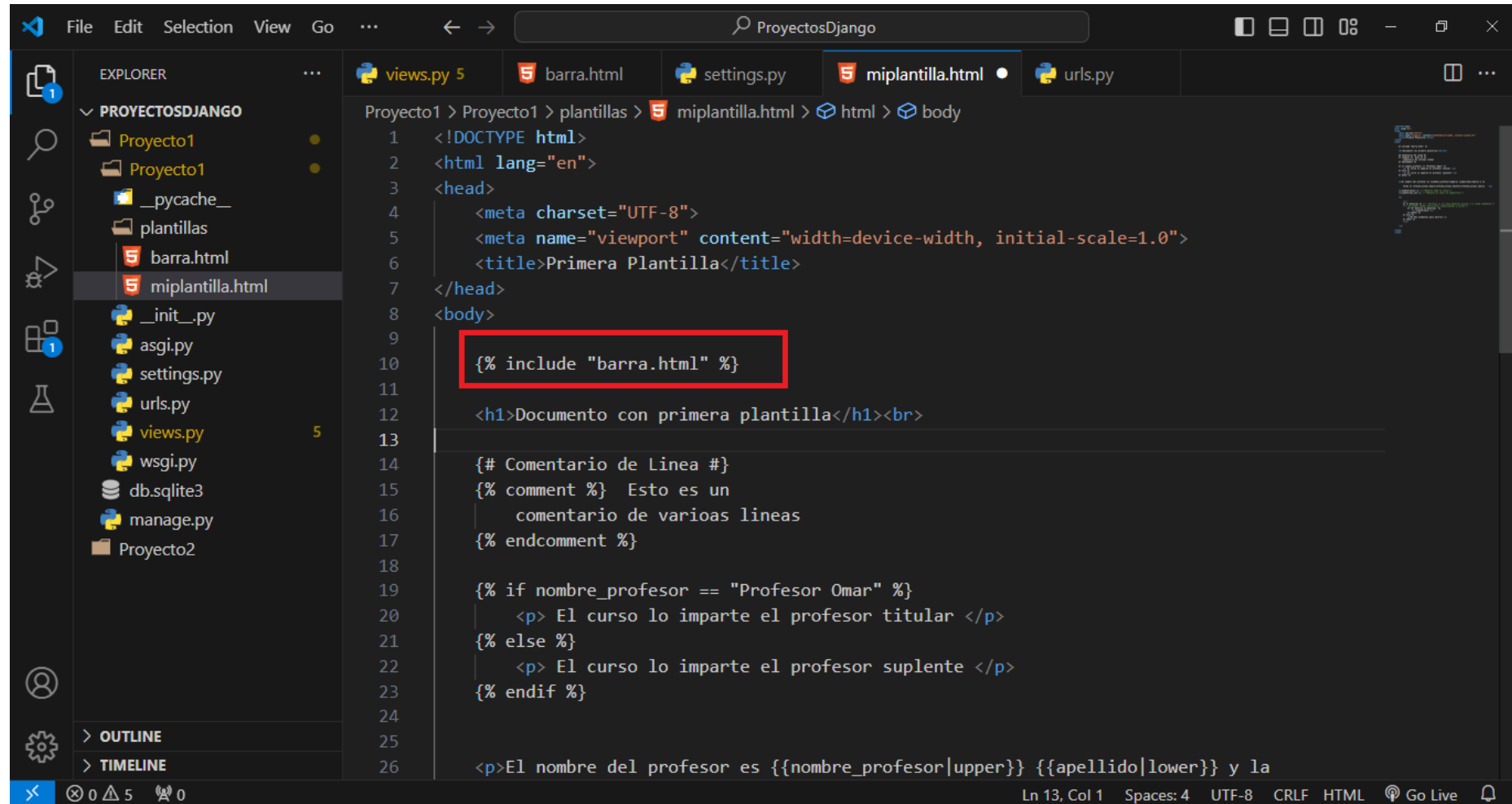
Plantillas Incrustadas

- Como su nombre lo indica son plantillas dentro de otras plantillas. Sintaxis:
`{% include "nombreplantilla.html" %}`



Plantillas Incrustadas

- En el archivo miplantilla.html



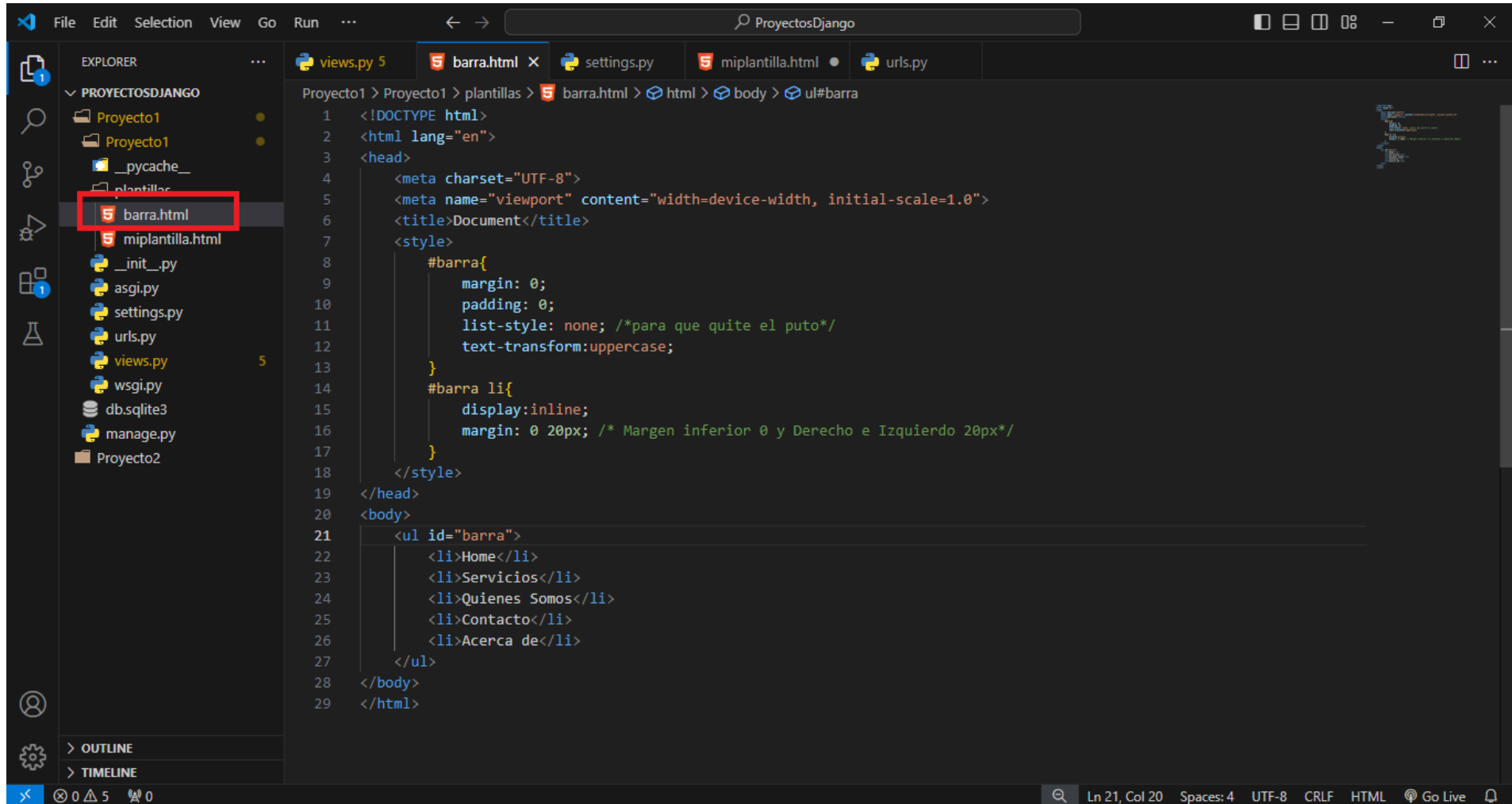
The screenshot shows the Visual Studio Code editor interface. On the left, the Explorer sidebar displays the project structure for 'ProyectosDjango'. It includes a 'Plantillas' folder containing 'barra.html' and 'miplantilla.html'. The 'miplantilla.html' file is selected and open in the main editor. The editor shows the following HTML code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Primera Plantilla</title>
7 </head>
8 <body>
9   {% include "barra.html" %}
10
11   <h1>Documento con primera plantilla</h1><br>
12
13   {# Comentario de Linea #}
14   {% comment %} Esto es un
15     comentario de varias lineas
16   {% endcomment %}
17
18   {% if nombre_profesor == "Profesor Omar" %}
19     <p> El curso lo imparte el profesor titular </p>
20   {% else %}
21     <p> El curso lo imparte el profesor suplente </p>
22   {% endif %}
23
24
25   <p>El nombre del profesor es {{nombre_profesor|upper}} {{apellido|lower}} y la
```

The line containing the Django template include tag, `{% include "barra.html" %}`, is highlighted with a red rectangle. The status bar at the bottom indicates the current position is Line 13, Column 1, with 4 spaces, in UTF-8 encoding, using CRLF line endings, and the file is an HTML document.

Plantillas Incrustadas

- Creamos el archivo barra.html



The screenshot shows a code editor with the following details:

- Explorer Panel:** Shows a project structure with folders 'Proyecto1' and 'Proyecto2'. Inside 'Proyecto1', there is a 'plantillas' folder containing 'barra.html' (highlighted with a red box), 'miplantilla.html', and other files like '__init__.py', 'asgi.py', 'settings.py', 'urls.py', 'views.py', 'wsgi.py', 'db.sqlite3', and 'manage.py'.
- Editor Panel:** Displays the content of 'barra.html'. The file path is 'Proyecto1 > Proyecto1 > plantillas > barra.html'. The code is as follows:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <style>
8     #barra{
9       margin: 0;
10      padding: 0;
11      list-style: none; /*para que quite el puto*/
12      text-transform:uppercase;
13    }
14    #barra li{
15      display:inline;
16      margin: 0 20px; /* Margen inferior 0 y Derecho e Izquierdo 20px*/
17    }
18  </style>
19 </head>
20 <body>
21   <ul id="barra">
22     <li>Home</li>
23     <li>Servicios</li>
24     <li>Quienes Somos</li>
25     <li>Contacto</li>
26     <li>Acerca de</li>
27   </ul>
28 </body>
29 </html>
```
- Status Bar:** Shows 'Ln 21, Col 20', 'Spaces: 4', 'UTF-8', 'CRLF', 'HTML', and 'Go Live'.

Plantillas Incrustadas



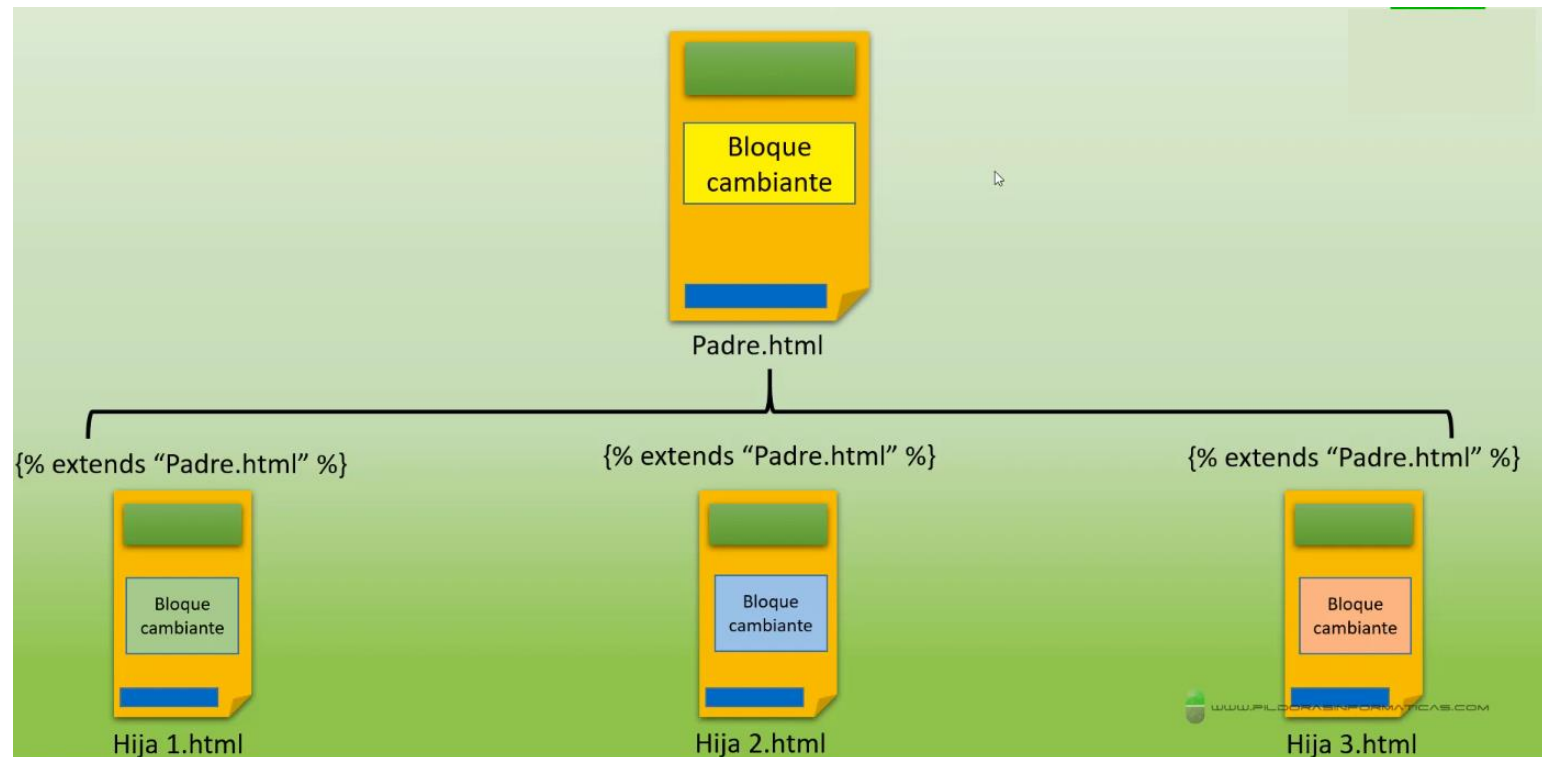
- Se pueden crear subcarpetas para organizar las plantillas

A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows a project structure with a folder named 'plantillas' containing a subfolder 'superior' and a file 'barra.html'. The 'superior' folder is highlighted with a red rectangle. The main editor window shows the 'miplantilla.html' file, which contains HTML code with Django template tags. A red rectangle highlights the line '{% include "superior/barra.html" %}' on line 10. The code also includes a doctype, meta tags, a title, and a body section with a heading, comments, and conditional logic for displaying professor information.

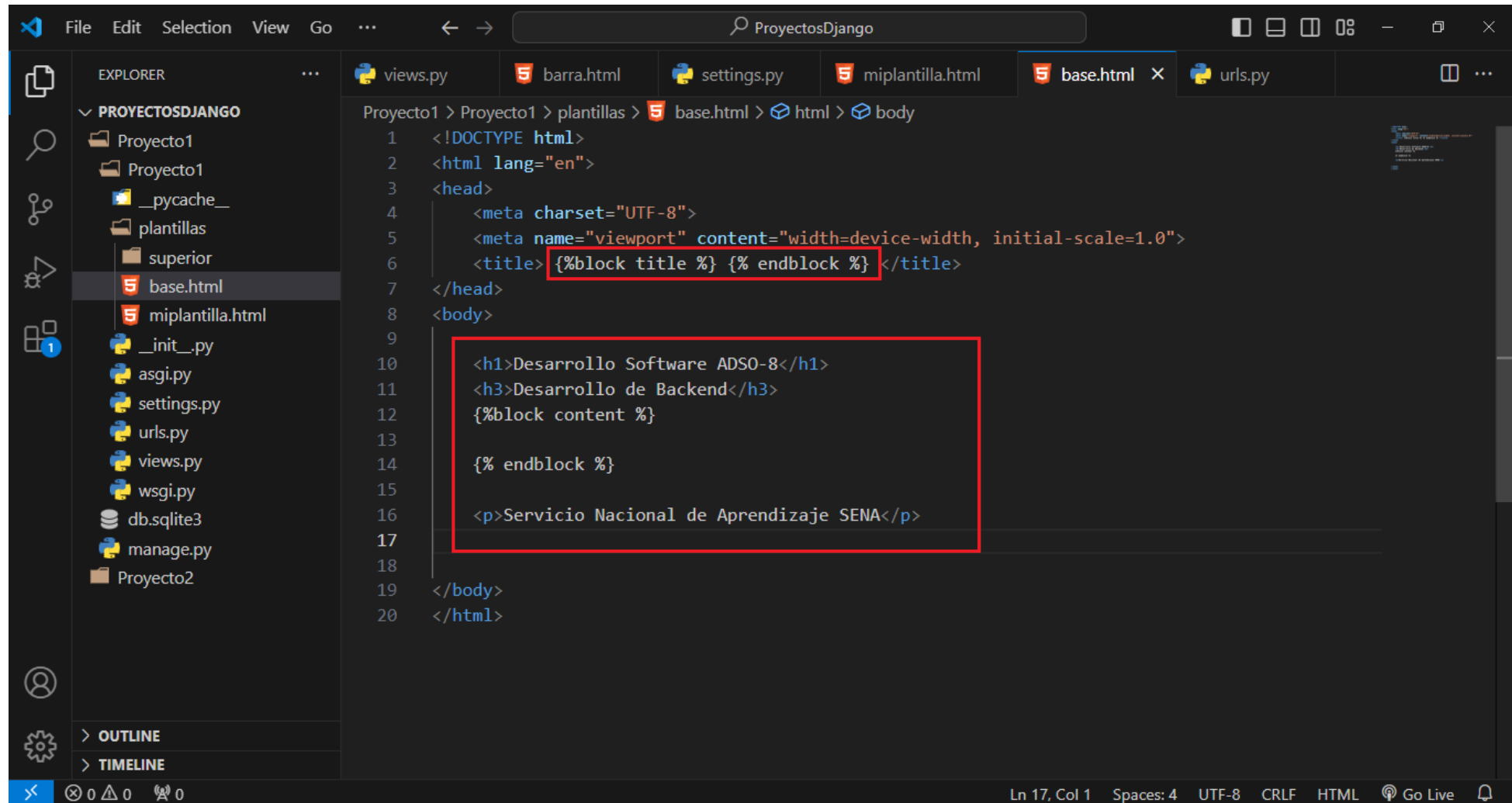
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Primera Plantilla</title>
7 </head>
8 <body>
9
10   {% include "superior/barra.html" %}
11
12   <h1>Documento con primera plantilla</h1><br>
13
14   {# Comentario de Linea #}
15   {% comment %} Esto es un
16     comentario de varias lineas
17   {% endcomment %}
18
19   {% if nombre_profesor == "Profesor Omar" %}
20     <p> El curso lo imparte el profesor titular </p>
21   {% else %}
22     <p> El curso lo imparte el profesor suplente </p>
23   {% endif %}
24
25   <p>El nombre del profesor es {{nombre_profesor|upper}} {{apellido|lower}} y la
```

Plantillas Herencia

- La herencia de plantillas en Django es una técnica que permite crear una estructura de plantilla base y luego extenderla en plantillas secundarias. Esto facilita la reutilización de código HTML común en múltiples páginas de un sitio web y ayuda a mantener una estructura coherente en todo el proyecto.



- Creamos la plantilla padre llamada **base.html**



The screenshot shows the Visual Studio Code editor interface. The Explorer panel on the left displays the project structure for 'PROYECTOSDJANGO', including folders 'Proyecto1' and 'plantillas', and files like 'base.html' and 'miplantilla.html'. The main editor window shows the content of 'base.html' with the following code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>{%block title %} {% endblock %}</title>
7 </head>
8 <body>
9
10  <h1>Desarrollo Software ADSO-8</h1>
11  <h3>Desarrollo de Backend</h3>
12  {%block content %}
13
14  {% endblock %}
15
16  <p>Servicio Nacional de Aprendizaje SENA</p>
17
18
19 </body>
20 </html>
```

The code is highlighted with red boxes around the title block and the body content block. The status bar at the bottom indicates 'Ln 17, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', 'HTML', and 'Go Live'.

Plantillas Herencia



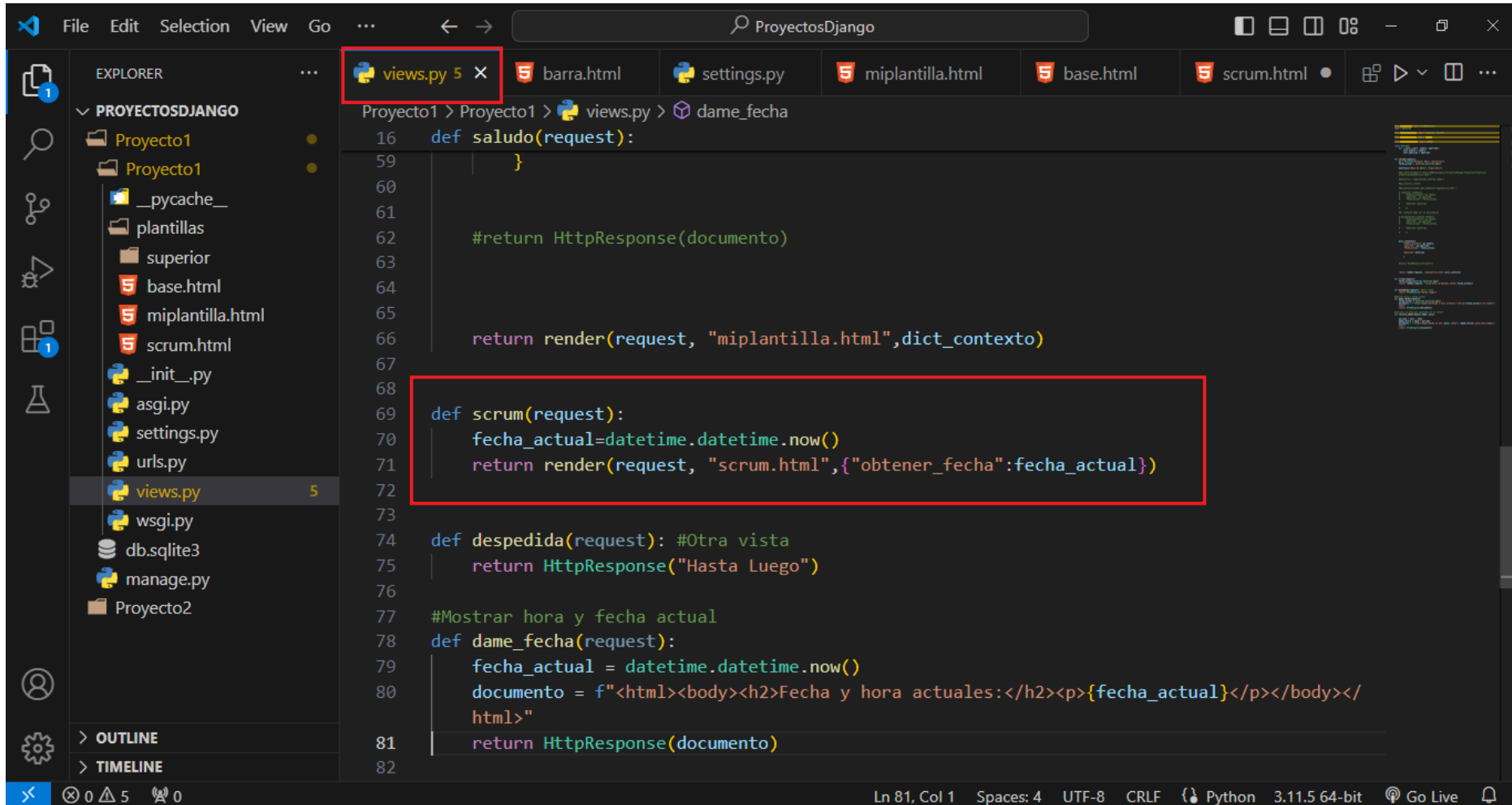
- Creamos la plantilla hija llamada **scrum.html**

A screenshot of the Visual Studio Code editor interface. The Explorer panel on the left shows a project structure with a folder named 'plantillas' containing 'base.html', 'miplantilla.html', and 'scrum.html'. The 'scrum.html' file is selected. The main editor area shows the content of 'scrum.html', which is a Django template that extends 'base.html'. The template includes a title block, a content block, and a YouTube video player. The code is as follows:

```
1 {% extends "base.html" %}
2
3
4 {% block title %}Taller Scrum{% endblock title %}
5
6 {% block content %}
7
8 <p>Estamos en la fecha: {{obtener_fecha}}</p>
9
10 <iframe width="560" height="315" src="https://www.youtube.com/embed/HhC75IonpOU?
11 si=5o5xL_E0X6MLWkBE" title="YouTube video player" frameborder="0" allow="accelerometer;
12 autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share"
referrerpolicy="strict-origin-when-cross-origin" allowfullscreen></iframe>
13
14 {% endblock content %}
```

The code is highlighted with a red border. The status bar at the bottom indicates the current position is Line 12, Column 23, with 4 spaces, UTF-8 encoding, CRLF line endings, and HTML language mode.

- Creamos la vista en el archivo **views.py** para que renderice la plantilla hija **scrum.html**

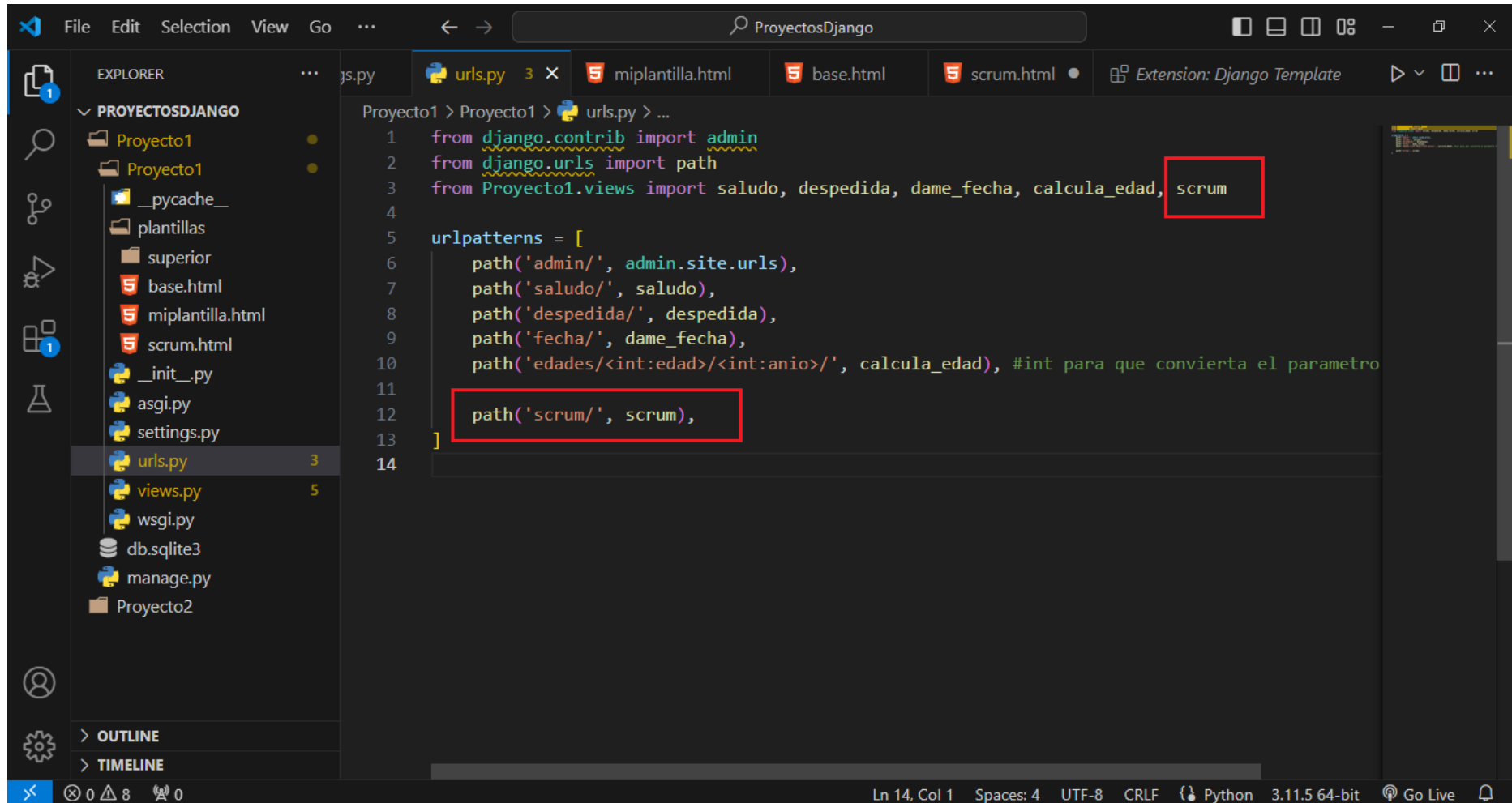


```
File Edit Selection View Go ... < > ProyectosDjango
EXPLORER
PROYECTOSDJANGO
  Proyecto1
    Proyecto1
      __pycache__
      plantillas
        superior
      base.html
      miplantilla.html
      scrum.html
      __init__.py
      asgi.py
      settings.py
      urls.py
      views.py 5
      wsgi.py
      db.sqlite3
      manage.py
      Proyecto2
  > OUTLINE
  > TIMELINE

Projecto1 > Proyecto1 > views.py > dame_fecha
16 def saludo(request):
59     }
60
61
62     #return HttpResponse(documento)
63
64
65
66     return render(request, "miplantilla.html", dict_contexto)
67
68
69 def scrum(request):
70     fecha_actual=datetime.datetime.now()
71     return render(request, "scrum.html", {"obtener_fecha":fecha_actual})
72
73
74 def despedida(request): #Otra vista
75     return HttpResponse("Hasta Luego")
76
77 #Mostrar hora y fecha actual
78 def dame_fecha(request):
79     fecha_actual = datetime.datetime.now()
80     documento = f"<html><body><h2>Fecha y hora actuales:</h2><p>{fecha_actual}</p></body></html>"
81     return HttpResponse(documento)
82
```

Ln 81, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit Go Live

- Asociamos una **url** a la vista **scrum** definida en el archivo **views.py**, para ello utilizamos el archivo **urls.py**



```
ProjectosDjango
Projecto1 > Proyecto1 > urls.py > ...
1 from django.contrib import admin
2 from django.urls import path
3 from Proyecto1.views import saludo, despedida, dame_fecha, calcula_edad, scrum
4
5 urlpatterns = [
6     path('admin/', admin.site.urls),
7     path('saludo/', saludo),
8     path('despedida/', despedida),
9     path('fecha/', dame_fecha),
10    path('edades/<int:edad>/<int:anio>/', calcula_edad), #int para que convierta el parametro
11
12    path('scrum/', scrum),
13 ]
14
```

Plantillas Herencia



- Podemos cambiar elementos en el elemento base para que se reflejen en todos los hijos

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title> {%block title %} {% endblock %} </title>
7 </head>
8 <body>
9
10     <h1 style="background-color: darkgreen; color: white; text-align: center;">Desarrollo
    Software ADSO-8</h1>
11
12     <h3>Desarrollo de Backend</h3>
13     {%block content %}
14
15     {% endblock %}
16
17     <p style="background-color: darkgreen; color: white; text-align:right;">Servicio
    Nacional de Aprendizaje SENA</p>
18
19
20 </body>
21 </html>
```

- Quedando Así

Desarrollo Software ADSO-8

Desarrollo de Backend

Estamos en la fecha: May 6, 2024, 8:37 p.m.



Plantillas Herencia



- Vamos a crear otra plantilla hija llamada **algoritmos.html**

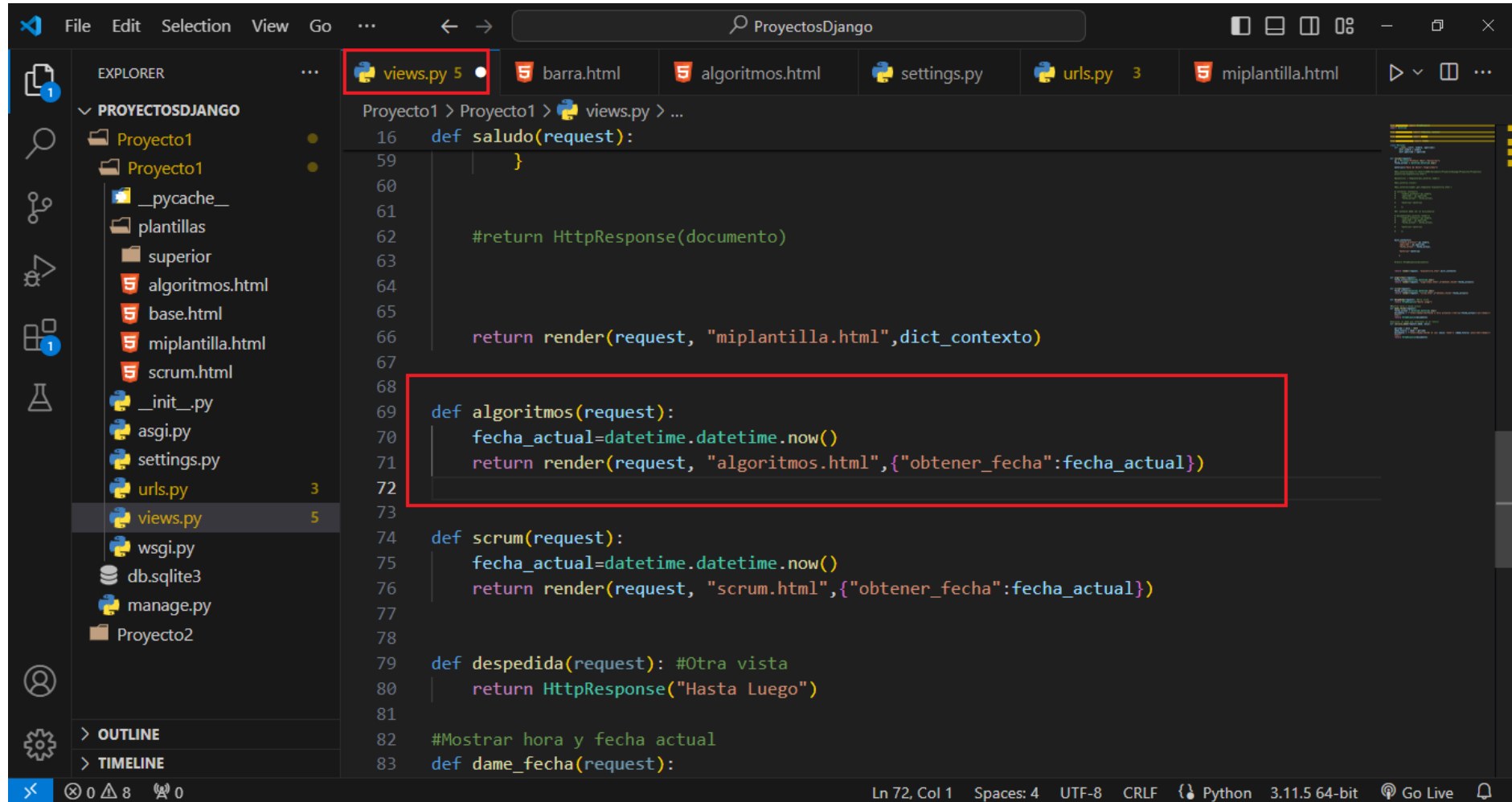
A screenshot of the Visual Studio Code editor interface. The Explorer panel on the left shows a project structure with a 'plantillas' directory containing 'algoritmos.html', 'base.html', 'miplantilla.html', and 'scrum.html'. The 'algoritmos.html' file is selected. The main editor area shows the content of 'algoritmos.html', which is a Django template that extends 'base.html'. The template includes a title block, a content block, and a paragraph of text. It also contains an iframe for a YouTube video. The code is as follows:

```
1 {% extends "base.html" %}
2
3
4 {% block title %}Algoritmos{% endblock title %}
5
6 {% block content %}
7
8 <p>Estamos en la fecha: {{obtener_fecha}}</p>
9
10 <iframe width="560" height="315" src="https://www.youtube.com/embed/gAsQEvSXsKc?
    si=UUb2aFgpw2cHVJ81" title="YouTube video player" frameborder="0" allow="accelerometer;
    autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share"
    referrerpolicy="strict-origin-when-cross-origin" allowfullscreen></iframe>
11
12 {% endblock content %}
```

Plantillas Herencia



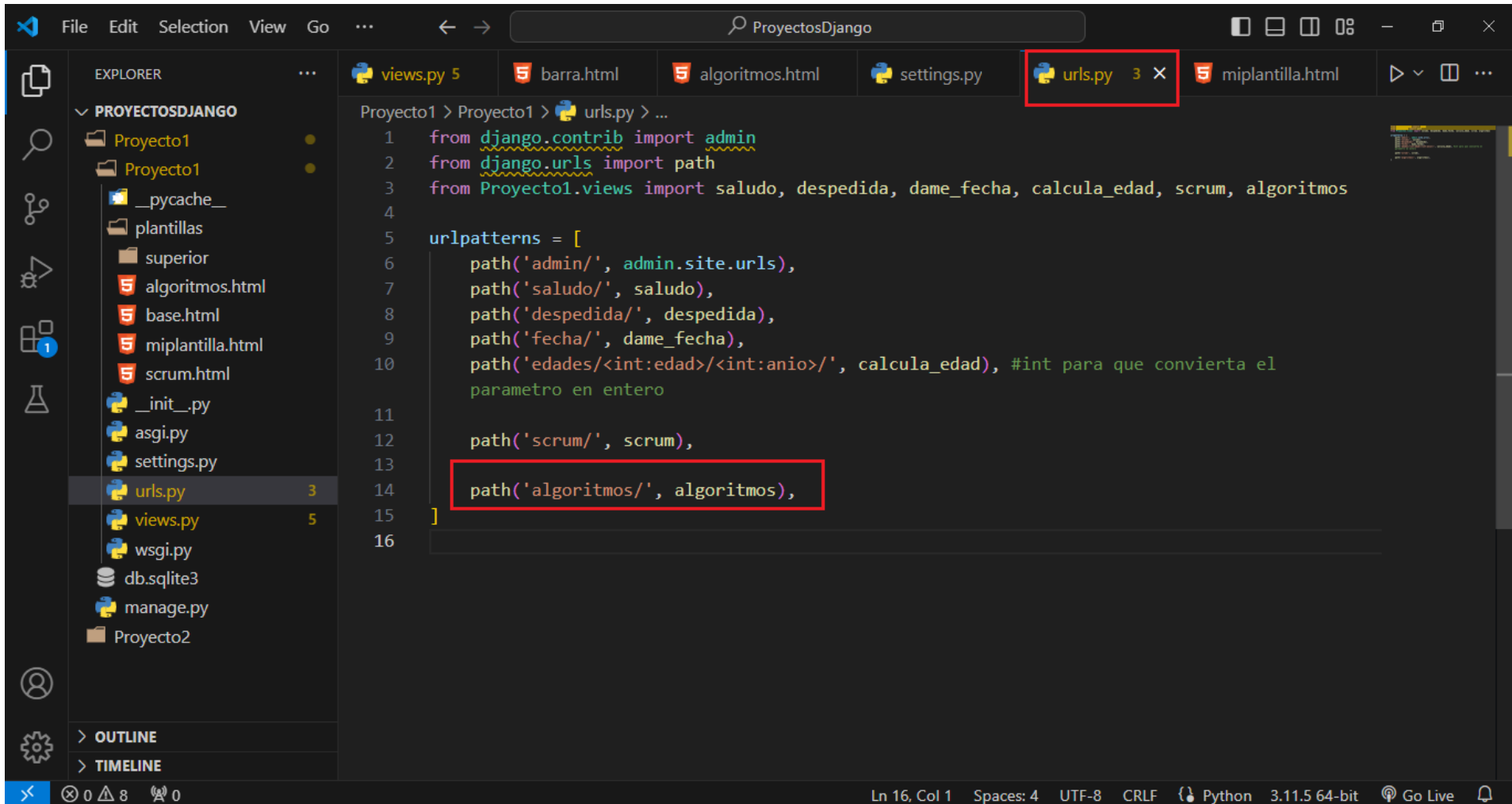
- Creamos la vista

A screenshot of a code editor (VS Code) showing a Django project named 'ProyectosDjango'. The Explorer panel on the left shows the project structure, including a 'plantillas' folder and various HTML templates. The main editor window displays the 'views.py' file, which contains several view functions. The 'saludo' function is highlighted with a red box, and the 'algoritmos' function is also highlighted with a red box. The status bar at the bottom indicates the current line and column (Ln 72, Col 1) and the Python version (3.11.5 64-bit).

```
16 def saludo(request):
59     }
60
61
62     #return HttpResponse(documento)
63
64
65
66     return render(request, "miplantilla.html", dict_contexto)
67
68
69 def algoritmos(request):
70     fecha_actual=datetime.datetime.now()
71     return render(request, "algoritmos.html", {"obtener_fecha": fecha_actual})
72
73
74 def scrum(request):
75     fecha_actual=datetime.datetime.now()
76     return render(request, "scrum.html", {"obtener_fecha": fecha_actual})
77
78
79 def despedida(request): #Otra vista
80     return HttpResponse("Hasta Luego")
81
82 #Mostrar hora y fecha actual
83 def dame_fecha(request):
```


Plantillas Herencia

- Registramos la url



The screenshot shows a code editor with a dark theme. The Explorer panel on the left shows a project named 'PROYECTOSDJANGO' with a sub-project 'Proyecto1'. Inside 'Proyecto1', there is a 'plantillas' folder and several files: 'superior', 'algoritmos.html', 'base.html', 'miplantilla.html', 'scrum.html', '__init__.py', 'asgi.py', 'settings.py', 'urls.py' (highlighted with a count of 3), 'views.py' (count of 5), and 'wsgi.py'. The main editor area shows the content of 'urls.py'. The code defines 'urlpatterns' as a list of path() functions. The paths are: 'admin/' (using 'admin.site.urls'), 'saludo/' (using 'saludo'), 'despedida/' (using 'despedida'), 'fecha/' (using 'dame_fecha'), 'edades/<int:edad>/<int:anio>/' (using 'calcula_edad' with a comment '#int para que convierta el parametro en entero'), 'scrum/' (using 'scrum'), and 'algoritmos/' (using 'algoritmos'). The 'algoritmos/' path is highlighted with a red box. The status bar at the bottom indicates 'Ln 16, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', 'Python 3.11.5 64-bit', and 'Go Live'.

```
1 from django.contrib import admin
2 from django.urls import path
3 from Proyecto1.views import saludo, despedida, dame_fecha, calcula_edad, scrum, algoritmos
4
5 urlpatterns = [
6     path('admin/', admin.site.urls),
7     path('saludo/', saludo),
8     path('despedida/', despedida),
9     path('fecha/', dame_fecha),
10    path('edades/<int:edad>/<int:anio>', calcula_edad), #int para que convierta el
    parametro en entero
11
12    path('scrum/', scrum),
13
14    path('algoritmos/', algoritmos),
15 ]
16
```

Plantillas Herencia

- Incluiremos la barra de navegación creada anteriormente

The screenshot shows the Visual Studio Code editor interface. The Explorer sidebar on the left displays the project structure, with the 'base.html' file selected. The main editor window shows the content of 'base.html', which is a Django template. The code is as follows:

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title> {%block title %} {% endblock %}</title>
7 </head>
8 <body>
9
10     <h1 style="background-color: darkgreen; color: white; text-align: center;">Desarrollo
11     Software ADSO-8</h1>
12     {% include "superior/barra.html" %}
13
14     <h3 style="text-align: right;">Desarrollo de Backend</h3>
15     {%block content %}
16
17     {% endblock %}
18
19     <p style="background-color: darkgreen; color: white; text-align:right;">Servicio
20     Nacional de Aprendizaje SENA</p>
21
22 </body>
23 </html>

```

The status bar at the bottom indicates the current position is Line 22, Column 8, with 4 spaces, UTF-8 encoding, CRLF line endings, and HTML language mode.

Ejercicios para desarrollar



G R A C I A S

Línea de atención al ciudadano: 01 8000 910270
Línea de atención al empresario: 01 8000 910682



www.sena.edu.co