

# Django FormView



**MG. Cristian Camilo Ordoñez**  
**Ingeniería De Sistemas**  
**Fundación Universitaria De Popayán**

# Temas

## 1. Modelos

1.1 Formulario y modelo

1.2 Restricciones

1.3 Prueba

1.4 widgets

*(Se anexa archivos.py con todo lo visto en esta presentación)*

# FormView

FormView se refiere a una vista (lógica) para mostrar y verificar un formulario Django. Por ejemplo, un formulario para registrar usuarios en registros. Las vistas basadas en clases proporcionan una forma alternativa de implementar vistas como objetos de Python en lugar de funciones.

No reemplazan las vistas basadas en funciones, pero tienen ciertas diferencias y ventajas en comparación con las vistas basadas en funciones:

- La organización del código relacionado con métodos HTTP específicos (GET, POST, etc.) se puede abordar mediante métodos separados en lugar de la ramificación condicional.
- Se pueden utilizar técnicas orientadas a objetos como mixins (herencia múltiple) para factorizar el código en componentes reutilizables.

# FormView

Juan Mellado

## Ingreso de registro

Datos de Instructor

|                         |  |                        |  |
|-------------------------|--|------------------------|--|
| <b>RUN</b>              | <input type="text"/>                                   | <b>Nombre Completo</b> | <input type="text"/>                                   |
| <b>Fecha Nacimiento</b> | <input type="text" value="dd - mm - aaaa"/>            | <b>Email</b>           | <input type="text" value="@"/>                         |
| <b>Teléfono</b>         | <input type="text" value="12"/>                        | <b>Dirección</b>       | <input type="text"/>                                   |
| <b>Ciudad</b>           | <input type="text" value="Seleccione una opción ..."/> | <b>Comuna</b>          | <input type="text" value="Seleccione una opción ..."/> |
| <b>Auto</b>             | <input type="text" value="XXXX-15"/>                   |                        |  |

**Añadir** **Cancelar**

|   |           |                            |                           |        |
|---|-----------|----------------------------|---------------------------|--------|
| 6 | 7493351-3 | Jorge Hernán Castro Toledo | jorgecastro36@hotmail.com | ACTIVO |
|---|-----------|----------------------------|---------------------------|--------|

# FormView

Crear el archivo forms.py

```
from django import forms

from .models import Prueba

"""MF"""
class PruebaForm(forms.ModelForm):
    """Form definition for Prueba."""

    class Meta:
        """Meta definition for Pruebaform."""

        model = Prueba
        fields = ('__all__')
```

En el archivo views.py importo el nuevo archivo

```
from .forms import PruebaForm
```

# FormView

En el archivo views.py cabio la configuración de los fields por form\_class de la siguiente manera

```
class PruebaCreateView(CreateView):  
    template_name = "home/add.html"  
    model = Prueba  
    form_class = PruebaForm // cambio fields por la vista generada por form  
    success_url = '/'
```

Compilo el proyecto e identifico que todo es normal

# FormView

The screenshot displays a web browser window with a form titled "FormView". The form contains three input fields: "Titulo:", "Subtitulo:", and "Cantidad:". The browser's developer tools are open, showing the HTML structure of the form. The HTML code is as follows:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
  </head>
  <body>
    <p>
      <label for="id_titulo">Titulo:</label>
      <input type="text" name="titulo" maxlength="100" required id="id_titulo">
    </p>
    <p>
      <label for="id_subtitulo">Subtitulo:</label>
      <input type="text" name="subtitulo" maxlength="50" required id="id_subtitulo">
    </p>
    <p>
      <label for="id_cantidad">Cantidad:</label>
      <input type="number" name="cantidad" required id="id_cantidad">
    </p>
  </body>
</html>
```

The right-hand pane of the developer tools shows the "Styles" panel, which is currently empty. Below the styles panel, a diagram illustrates the box model for the selected element, showing the margin, border, padding, and the content area (360 x 127).

Inspecciono el proyecto e identifico que el formulario este utilizando los campos como están definidos en el modelos

# Ventajas de validaciones en formularios

Añado la restricción al formulario se puede utilizar a el texto a un numero

```
class PruebaForm(forms.ModelForm):
    """Form definition for Prueba."""

    class Meta:
        """Meta definition for Pruebaform."""

        model = Prueba
        fields = ('__all__')

    ##validación
    def clean_cantidad(self):
        cantidad = self.cleaned_data['cantidad']
        if cantidad < 10:
            raise forms.ValidationError('ingrese un valor mayor a 10')
        return cantidad
```

Configuro la url

```
path('add/', views.PruebaCreateView.as_view(), name = 'prueba_add'),
```



# Identificar la restricción

← → ↻ ⓘ 127.0.0.1:8000/add/

## Aquí esta mi primer formulario

Título:

Subtítulo:

- ingrese un valor mayor a 10

Cantidad:

**Identifico que la restricción  
este funcionando**

# Creación de form

|  |                          |
|--|--------------------------|
| Tipo de documento (*):                     | Número de documento (*): |
| <input type="text" value="Seleccione..."/> | <input type="text"/>     |
| Nombres (*):                               | Segundo nombre:          |
| <input type="text"/>                       | <input type="text"/>     |
| Apellidos (*):                             | Segundo apellido:        |
| <input type="text"/>                       | <input type="text"/>     |

**Realice el siguiente formulario**

- 1. Crear modelo**
- 2. Crear form**
- 3. Crear template**
- 4. Crear vista**
- 5. Validar los campos cedula debe ser igual 10**
- 6. Validar nombres deben ser igual a 15 caracteres**

# Preguntas?