<!--Tutorial-->

Crud en Django

{

<Por="manuel parra
sebastian jaimes"/>

}

configuracion de entorno {

Asegúrate de tener Python instalado en tu sistema. Luego, instala Django usando pip:

```
pip install django
```

Crear una aplicación dentro del proyecto

```
python manage.py startapp myapp
```

Crear un nuevo proyecto Django

```
django-admin startproject myproject
cd myproject
```

}

Definir el modelo en models.py

```
from django.db import models

class Item(models.Model):
    name =
models.CharField(max_length=255)
    description = models.TextField()

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

Aplicar las migraciones

```
python manage.py makemigrations
python manage.py migrate
```

- Edita el archivo models.py dentro de la aplicación (myapp/models.py). Define el modelo que representará tus datos:

- Después de definir el modelo, crea y aplica las migraciones para crear la tabla en la base de datos

}

Crear las vistas en views.py {

```
from django.shortcuts import render, get_object_or_404
from .models import Item
from .forms import ItemForm
from django.shortcuts import redirect

def item_list(request):
    items = Item.objects.all()
    return render(request, 'myapp/item_list.html', {'items': items})

def item_detail(request, pk):
    item = get_object_or_404(Item, pk=pk)
    return render(request, 'myapp/item_detail.html', {'item': item})

def item_new(request):
    if request.method == "POST":
        form = ItemForm(request.POST)
        if form.is_valid():
            item = form.save(commit=False)
            item.save()
            return redirect('item_detail', pk=item.pk)
    else:
        form = ItemForm()
    return render(request, 'myapp/item_edit.html', {'form': form})

def item_edit(request, pk):
    item = get_object_or_404(Item, pk=pk)
    if request.method == "POST":
        form = ItemForm(request.POST, instance=item)
        if form.is_valid():
            item = form.save(commit=False)
            item.save()
            return redirect('item_detail', pk=item.pk)
    else:
        form = ItemForm(instance=item)
    return render(request, 'myapp/item_edit.html', {'form': form})

def item_delete(request, pk):
    item = get_object_or_404(Item, pk=pk)
    item.delete()
    return redirect('item_list')
```

- En myapp/views.py, crea las vistas que manejarán las operaciones CRUD:

}

Crear formularios en forms.py {

```
from django import forms
from .models import Item

class ItemForm(forms.ModelForm):
    class Meta:
        model = Item
        fields = ('name',
'description',)
```

- Crea un archivo forms.py dentro de tu aplicación (myapp/forms.py) y define el formulario para tu modelo:

}

Configurar las URLs en urls.py {

```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.item_list,
name='item_list'),
    path('item/<int:pk>/',
views.item_detail,
name='item_detail'),
    path('item/new/',
views.item_new, name='item_new'),
    path('item/<int:pk>/edit/',
views.item_edit, name='item_edit'),
    path('item/<int:pk>/delete/',
views.item_delete,
name='item_delete'),
]
```

- En el archivo urls.py de tu aplicación (myapp/urls.py), define las URL para tus vistas:

}

Incluir las URLs de la aplicación en el archivo principal urls.py {

```
from django.contrib import admin
from django.urls import include,
path

urlpatterns = [
    path('admin/',
admin.site.urls),
    path('',
include('myapp.urls')),
]
```

- En el archivo urls.py de tu proyecto principal (myproject/urls.py), incluye las URLs de tu aplicación:

}