

Asignación - Ejercicio (IA-Copilot) con Django

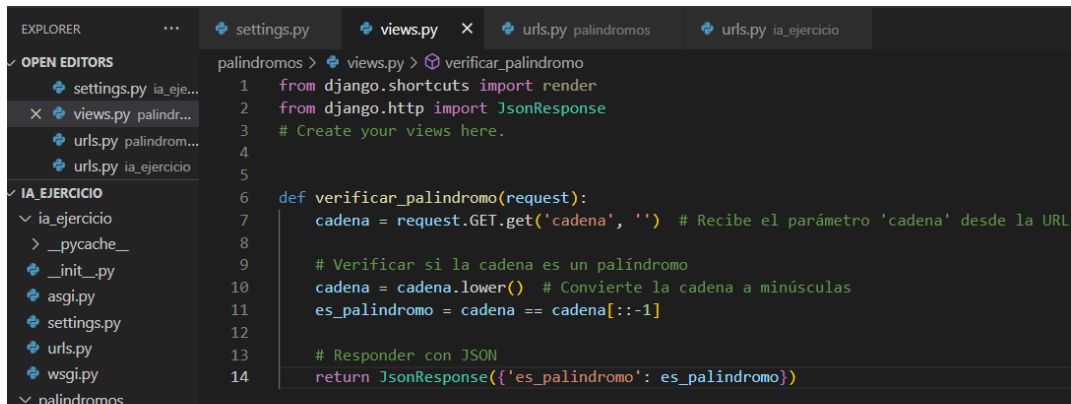
Ejercicio 1:

Realizar un código que verifique si una cadena es un palíndromo o no.

1. Incluir la aplicación para palíndromos:

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
  
    'palindromos',  
]
```

2. Función para verificar palíndromos:



The screenshot shows a code editor with the following content:

```
palindromos > views.py > verificar_palindromo  
1 from django.shortcuts import render  
2 from django.http import JsonResponse  
3 # Create your views here.  
4  
5  
6 def verificar_palindromo(request):  
7     cadena = request.GET.get('cadena', '') # Recibe el parámetro 'cadena' desde la URL  
8  
9     # Verificar si la cadena es un palíndromo  
10    cadena = cadena.lower() # Convierte la cadena a minúsculas  
11    es_palindromo = cadena == cadena[::-1]  
12  
13    # Responder con JSON  
14    return JsonResponse({'es_palindromo': es_palindromo})
```

3. Incluir las urls de la aplicación.

```
1 from django.urls import path  
2 from . import views  
3  
4 urlpatterns = [  
5     path('verificar_palindromo/', views.verificar_palindromo, name='verificar_palindromo'),  
6 ]  
7
```

```
20 urlpatterns = [  
21     path('admin/', admin.site.urls),  
22     path('palindromos/', include('palindromos.urls')),  
23 ]  
24
```

Pruebas:

```
127.0.0.1:8000/palindromos/verificar_palindromo/?cadena=radar
```

Dar formato al texto ☐

```
{"es_palindromo": true}
```

```
127.0.0.1:8000/palindromos/verificar_palindromo/?cadena=copiloto
```

Dar formato al texto ☐

```
{"es_palindromo": false}
```

```
127.0.0.1:8000/palindromos/verificar_palindromo/?cadena=Dracula%20y%20alucard
```

Dar formato al texto ☐

```
{"es_palindromo":true}
```

Ejercicio 2:

Realizar un código que implemente la búsqueda binaria.

1. Incluir la aplicación de búsqueda binaria.
2. Función de búsqueda binaria:

```
from django.http import JsonResponse

# Create your views here.

def busqueda_binaria(request):
    try:
        # Recibir la lista y el valor a buscar desde la URL
        lista = list(map(int, request.GET.get('lista', '').split(',')))
        valor = int(request.GET.get('valor', '')) # Convertir el valor

        # Realizar búsqueda binaria
        inicio = 0
        fin = len(lista) - 1
        encontrado = False
        indice = -1

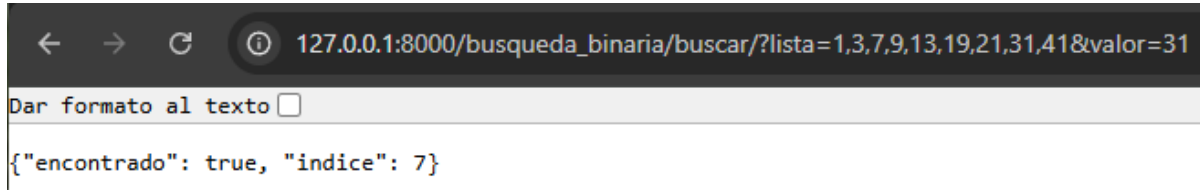
        while inicio <= fin:
            medio = (inicio + fin) // 2
            if lista[medio] == valor:
                encontrado = True
                indice = medio
                break
            elif lista[medio] < valor:
                inicio = medio + 1
            else:
                fin = medio - 1

        if encontrado:
            return JsonResponse({'encontrado': True, 'indice': indice})
        else:
            return JsonResponse({'encontrado': False})

    except Exception as e:
        return JsonResponse({'error': str(e)})
```

3. Incluir las urls de la aplicación.

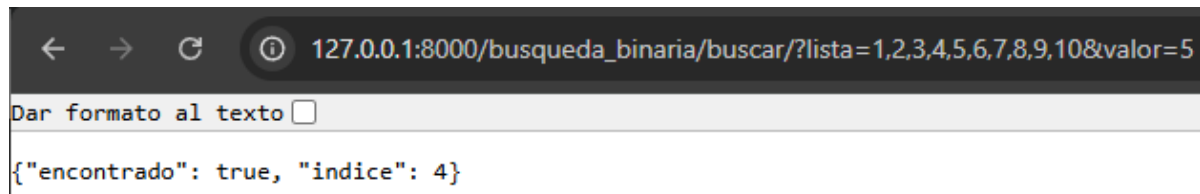
Pruebas:



127.0.0.1:8000/busqueda_binaria/buscar/?lista=1,3,7,9,13,19,21,31,41&valor=31

Dar formato al texto ☐

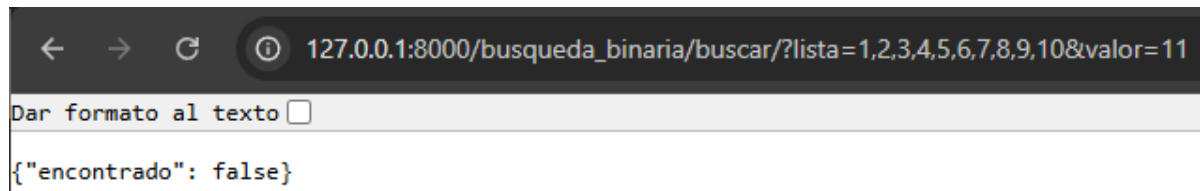
```
{"encontrado": true, "indice": 7}
```



127.0.0.1:8000/busqueda_binaria/buscar/?lista=1,2,3,4,5,6,7,8,9,10&valor=5

Dar formato al texto ☐

```
{"encontrado": true, "indice": 4}
```



127.0.0.1:8000/busqueda_binaria/buscar/?lista=1,2,3,4,5,6,7,8,9,10&valor=11

Dar formato al texto ☐

```
{"encontrado": false}
```