

# CREACIÓN AUTOMÁTICA DE PULL-REQUESTS A PARTIR DE RESULTADOS DE PYLINT

---

Autor: Raúl Cano Montero  
Tutor: Dr. Gregorio Robles

# Introducción

- Proyectos desarrollados en Python.
- Repositorios alojados en GitHub.
- Análisis mediante Pylint.
- Corrección de errores detectados.
- Realización de Pull-Requests con las modificaciones.
- Aumentar nivel de cumplimiento con PEP8.

Error ID	Error Name	Message
C0303	trailing-whitespace	Trailing whitespace. Used when there is whitespace between the end of a line and the newline.
C0304	missing-final-newline	Final newline missing. Used when the last line in a file is missing a newline.
C0321	multiple-statements	More than one statement on a single line. Used when more than one statement are found on the same line.
C0326	bad-whitespace	%s space %s %s %s. Used when a wrong number of spaces is used around an operator, bracket or block opener.
C0410	multiple-imports	Multiple imports on one line ( %s). Used when import statement importing multiple modules is detected.
C0411	wrong-import-order	%s should be placed before %s. Used when PEP8 import order is not respected (standard imports first, then third-party libraries, then local imports).
C0413	wrong-import-position	Import " %s" should be placed at the top of the module. Used when code and imports are mixed.
W0404	reimported	Reimport %r (imported line %s). Used when a module is reimported multiple times.
W0611	unused-import	Unused %s. Used when an imported module or variable is not used.

# Objetivos

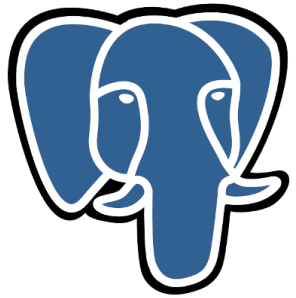
## Objetivo general

Creación de Pull-Requests tras analizar y corregir código que incumple las normas de PEP8.

## Objetivos específicos

- Aplicación web.
- Accesible desde Internet.
- Compatible con el GitLab de la ETSIT.
- Uso de proyectos reales.
- Análisis de aceptación de las Pull-Requests.

# Tecnologías



PostgreSQL

# django

 heroku



# GitHub



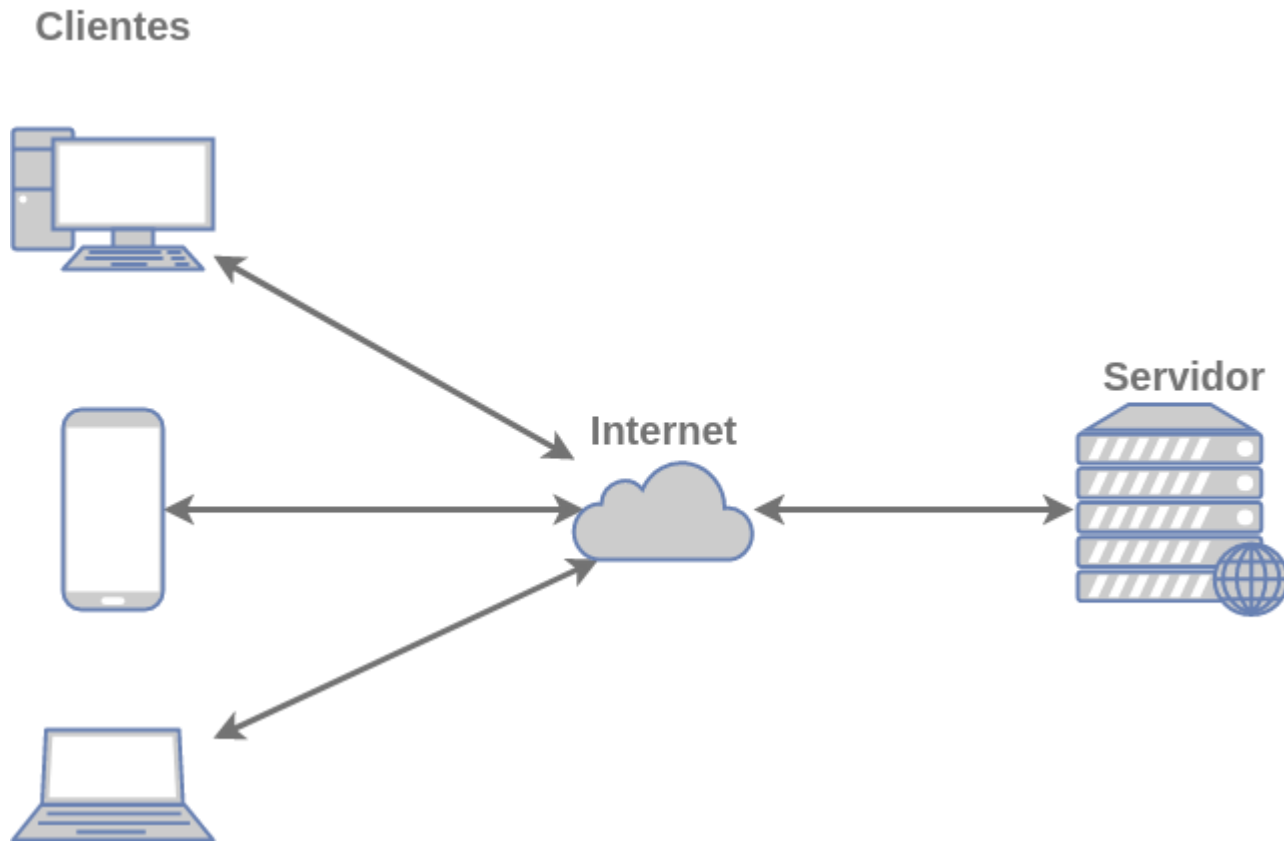
GitLab



# PyLint

Star your Python code!

# Diseño e implementación



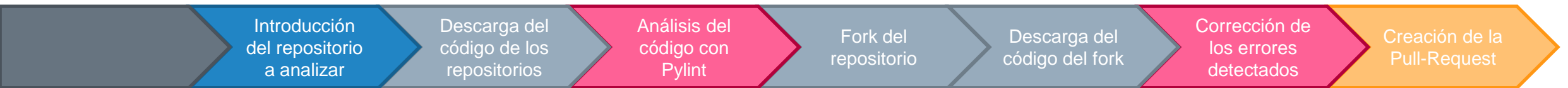
## Cliente

Navegador web.

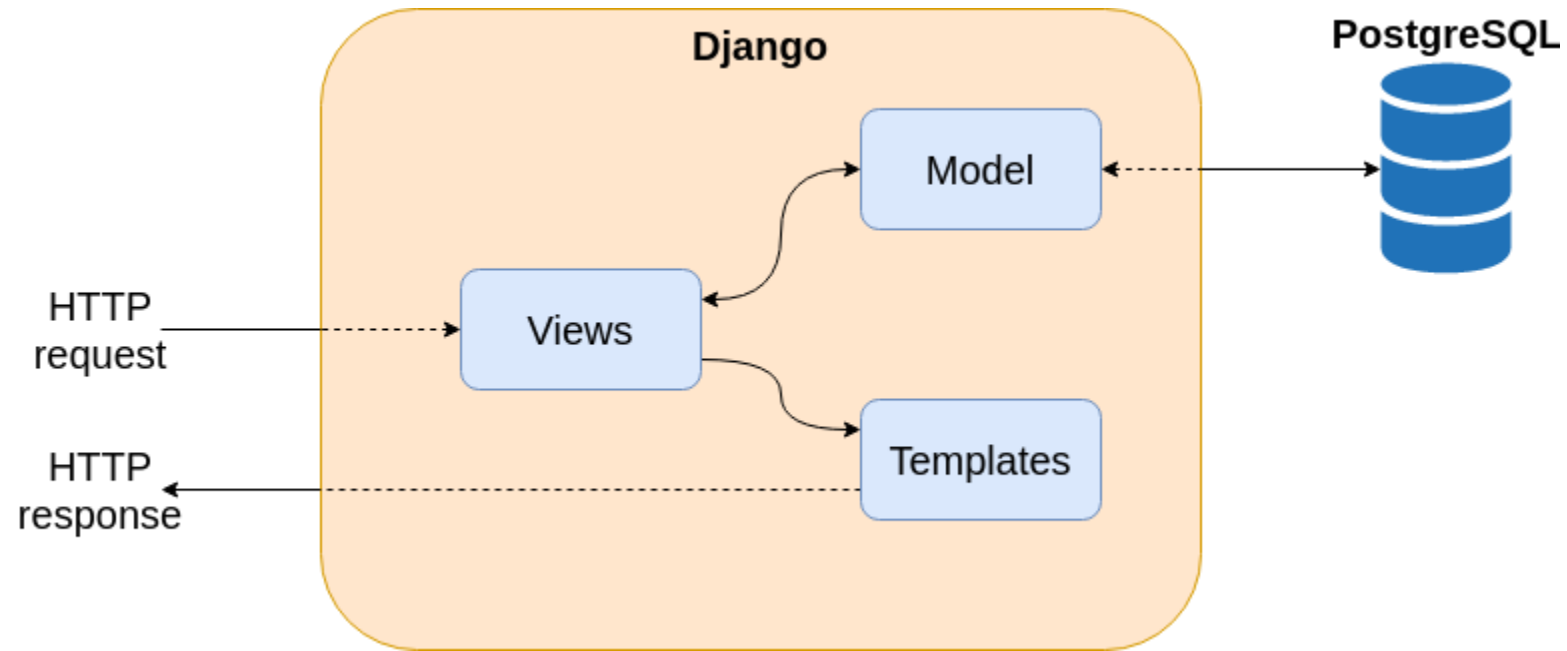
## Objetivo general

- Aplicación Django.
- Base de datos PostgreSQL.
- Despliegue en Heroku.

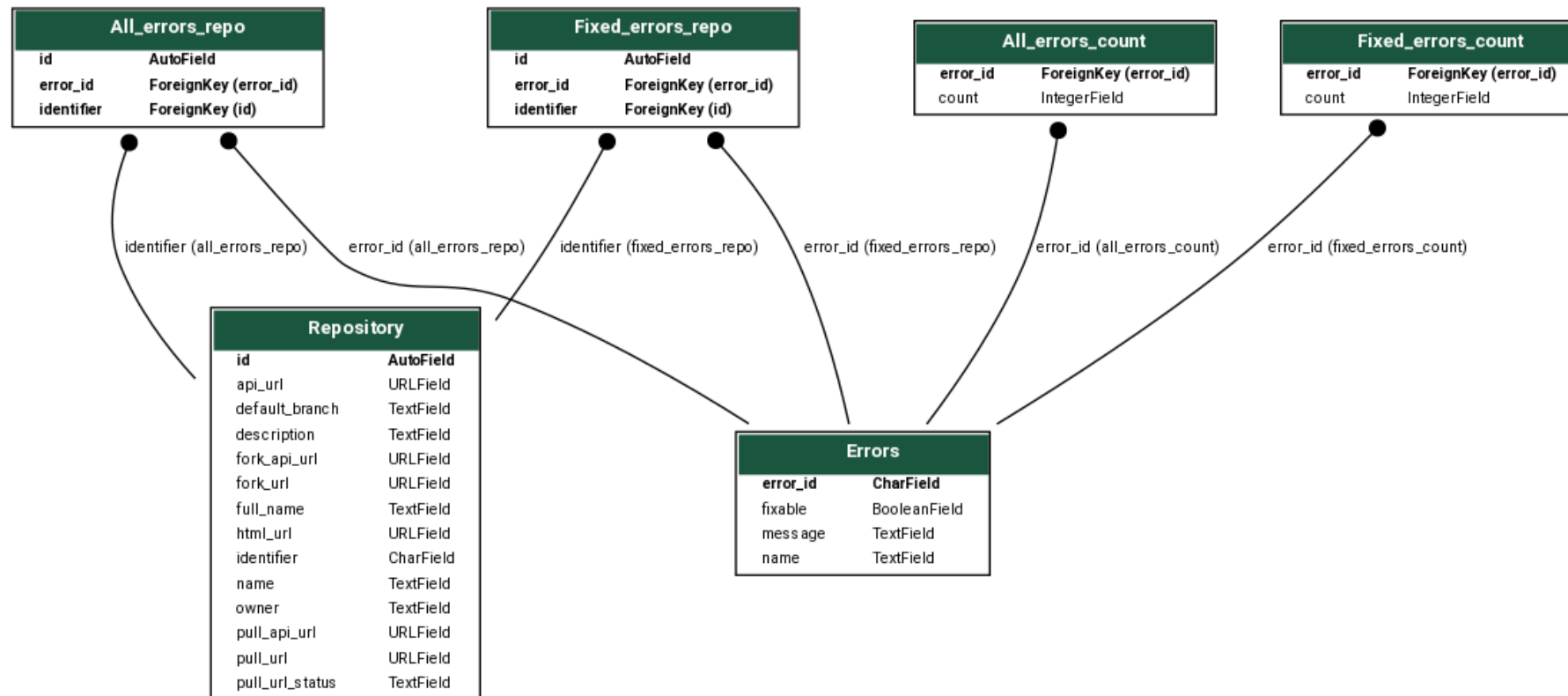
# Diseño e implementación del servidor



# Diseño e implementación del servidor: Django



# Diseño e implementación del servidor: BBDD





# Diseño e implementación del servidor: Errores de Pylint

```
from django.contrib import admin
from museos.models import Museo
admin.site.register(Museo)
from museos.models import Comentario
admin.site.register(Comentario)
from museos.models import ConfigUsuario
admin.site.register(ConfigUsuario)
from museos.models import Seleccionado
admin.site.register(Seleccionado)
```



```
from museos.models import Seleccionado
from museos.models import ConfigUsuario
from museos.models import Comentario
from django.contrib import admin
from museos.models import Museo
admin.site.register(Museo)
admin.site.register(Comentario)
admin.site.register(ConfigUsuario)
admin.site.register(Seleccionado)
```

```
for key,value in self.attributes:
    node.attributes[key] = value
return node
```



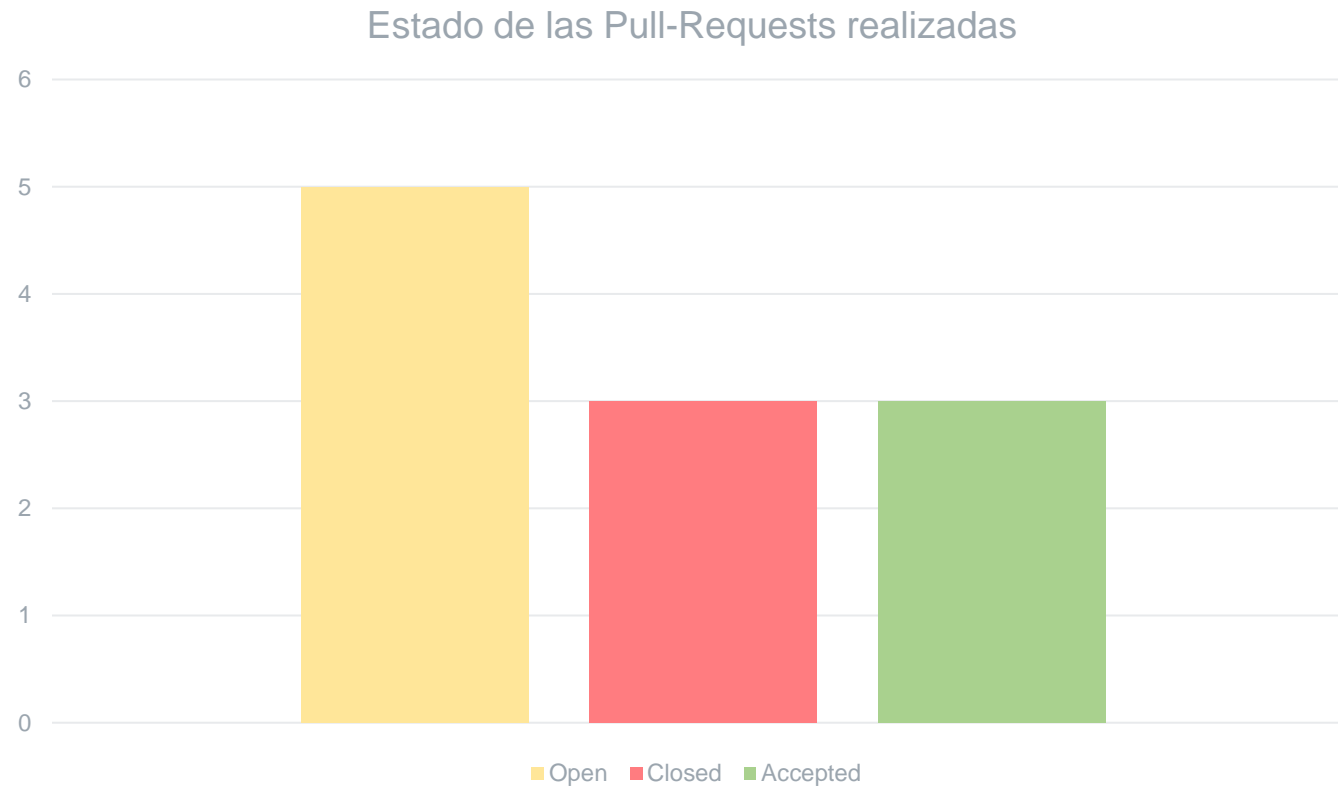
```
for key,value in self.attributes:
    node.attributes[key] = value
return node
```

```
if(num_museos==0):
    if request.method == "GET":
        name = 'Cargar'
        value = "Cargar"
```

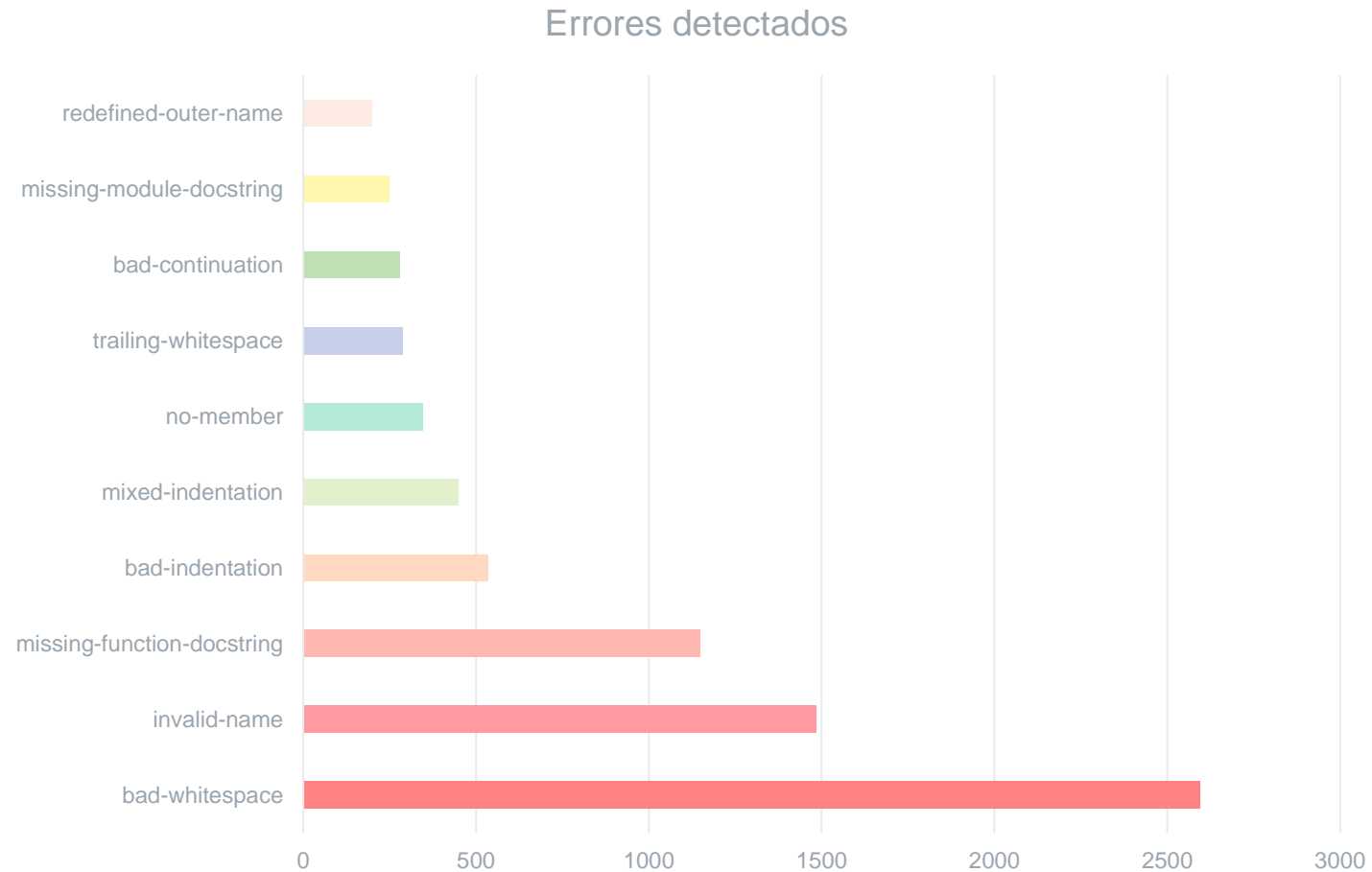


```
if(num_museos == 0):
    if request.method == "GET":
        name = 'Cargar'
        value = "Cargar"
```

# Resultados: Pull-Requests



# Resultados: Errores detectados



# Resultados: Errores corregidos



# Conclusiones

## Objetivos cumplidos

- ✓ Creación de Pull-Requests tras analizar y corregir código que incumple las normas de PEP8.
- ✓ Aplicación web.
- ✓ Compatible con el GitLab de la ETSIT.
- ✓ Uso de proyectos reales.
- ✓ Análisis de aceptación de las Pull-Requests.

## Objetivos cumplidos parcialmente

- Accesible desde internet.
  - Entrada en reposo.
  - Límite de líneas en la base de datos.