

# **DJANGO VS SYSADMIN**

**PYCONES, SEPTIEMBRE 2017, CÁCERES**

# PRUÉBALO

<https://github.com/klashxx/PyConES2017>

- Estructurada en ramas
- Incremental
- Docker

# AUTOBOMBO

- Sysadmin desde 2004
- Primer programa Amstrad CPC6128

<https://klashxx.github.io/about>



**WEB EXPERIENCE = 0**

# ¿POR QUÉ DJANGO?

- Python
- Fiable
- ORM
- Extensible
- DRF y otras hierbas
- Admin Site

A baby with light brown hair and a sad expression is the central figure. The baby is wearing a white long-sleeved shirt with green trim on the sleeves and collar. They are holding a small, light-colored doll in their right hand. The background is composed of large, overlapping geometric shapes in shades of blue, purple, and teal, creating a modern, abstract setting. The text "WIN OR WIN" is superimposed over the baby's face in a large, white, sans-serif font.

**WIN OR WIN**

**FIRST FAIL**

**CMS**

A man with glasses and a dark shirt is sitting at a desk in a dimly lit room. The background shows a city skyline at night. The text "EMPEZAR POR LOS" is written in large, white, sans-serif capital letters across the upper part of the image. Below it, the word "BASICS" is written in the same style. At the bottom, the word "HACKERMAN" is written in a stylized, metallic, 3D font.

EMPEZAR POR LOS  
BASICS

HACKERMAN



# DESARROLLO

- virtualenv
- Docker
- docker-compose
- Kubernetes - Docker-Swarm



DOCKER ES AMOR!

# DATABASES

- PostgreSQL
- MySQL
- SQLite
- Oracle

**GO FOR POSTGRESQL !**

# settings.py y .env

- Configuración
- Ocultar los secretos
- Variables globales

# DECOUPLE

## .ENV

```
SECRET_KEY=#^p^js0g91jhtarws6bp7s@r&988%%n_ok6#b)q=3s1v8-(si$
```

## SETTINGS.PY

```
from decouple import config  
SECRET_KEY = config('SECRET_KEY')
```

# DJ DATABASE URL

## .ENV

```
DB_URL=postgresql://postgres:postgres@postgres:5432/postgres
```

## SETTINGS.PY

```
DATABASES = {  
'default': dj_database_url.config(default=config('DB_URL')),  
}
```

**SECOND FAIL**

**LA ESTRUCTURA DEL PROYECTO**



# DOCS

```
mysite/  
  manage.py  
  mysite/  
    __init__.py  
    settings.py  
    urls.py  
    wsgi.py  
  polls/  
    __init__.py  
    admin.py  
    migrations/  
      __init__.py  
      0001_initial.py  
  models.py  
  static/
```

# FIRST TRY

```
sysgate/  
  manage.py  
  sysgate/  
    __init__.py  
    settings.py  
    urls.py  
    wsgi.py  
  account/  
    __init__.py  
    ...  
  metrics/  
    __init__.py  
    ...  
  templates/  
    account/
```

# F\*CK YEAH

```
sysgate/  
  manage.py  
  sysgate/  
    __init__.py  
    settings.py  
    urls.py  
    wsgi.py  
    routers.py  
    account/  
      __init__.py  
      ...  
    apps/  
      __init__.py  
      core/  
        templatetags/
```

A man with dark hair, wearing a dark t-shirt, is sitting on a dark blue couch. He is looking down at a laptop screen, which is partially visible in the foreground. His right hand is pressed against his forehead, suggesting stress or frustration. The background is a dimly lit room with a bookshelf filled with books and a doorway leading to another room.

# PRODUCTIVITY BOOST

# urls.py

- Regexp
- Modular

# ROOT

```
urlpatterns = [  
    url(r'^$', include('apps.core.urls', namespace='core'))  
    url(r'^account/', include('account.urls'), name='Auten'  
    url(r'^metrics/', include('apps.metrics.urls'), name='  
    url(r'^manager/', include('apps.manager.urls'), name='  
    url(r'^admin/', admin.site.urls),  
    ]
```

# CORE

```
urlpatterns = [  
    url(r'^$', views.Home.as_view(), name='home'),  
    ]
```

# MODELOS Y VISTAS ...¿DÓNDE ESTÁ EL CONTROLADOR?

- Framework «MTV». «Model», «Template» y «View»

# models.py

- Clase cuyos objetos instanciados están dotados de persistencia en BD



# TIPS

- managed = False
- Campo único

# views.py

- Extraer datos de nuestros models.py
- NO necesitamos SQL

# TIPS

- Class Based Views

# template.html

- Representa los datos extraídos por la vista

```
<h1>Metrics App</h1>
<p>Bienvenid@ <strong>{{ request.user }}</strong></p>
<p>Disponibles:</p>

{% for metrica in metricas %}

{% endfor %}
<table style="border: 1px solid black;"><tbody><tr>
    <td>{{ metrica.nombre }}</td>
    <td>{{ metrica.get_tipo_display }}</td>
</tr></tbody></table>
```

# AUTENTICACIÓN Y REGISTRO

- AbstractUser
- Auth views

# DRF

## FIRST TRY

```
class Home(View):
    def get(self, request, *args, **kwargs):
        metricas = Metrica.objects.all().order_by('tipo')
        return render(
            request,
            'metrics/home.html',
            context={'metricas': metricas})
```

```
{% for metrica in metricas %}

    {{ metrica.nombre }}
    {{ metrica.get_tipo_display }}

{% endfor %}
```







THE WHITE HOUSE  
WASHINGTON

SEPARA  
EL  
BACKEND  
DEL  
FRONTEND

*Donald Trump*

THE WHITE HOUSE  
WASHINGTON

SEPARA  
EL  
BACKEND  
DEL  
FRONTEND

*Donald Trump*

# SERIALIZANDO ...

```
class SerializerMetricas(serializers.ModelSerializer):  
    class Meta:  
        model = Metrica  
        fields = '__all__'
```

```
class MetricasViewSet(viewsets.ModelViewSet):  
    serializer_class = SerializerMetricas  
    http_method_names = ['get']  
    filter_backends = (OrderingFilter, SearchFilter,)  
    search_fields = ('tipo',)  
    def get_queryset(self):  
        queryset = Metrica.objects.all().order_by('tipo')  
        tipo = self.request.query_params.get('tipo', None)  
        if tipo is not None:  
            queryset = queryset.filter(tipo=tipo)  
        return queryset
```

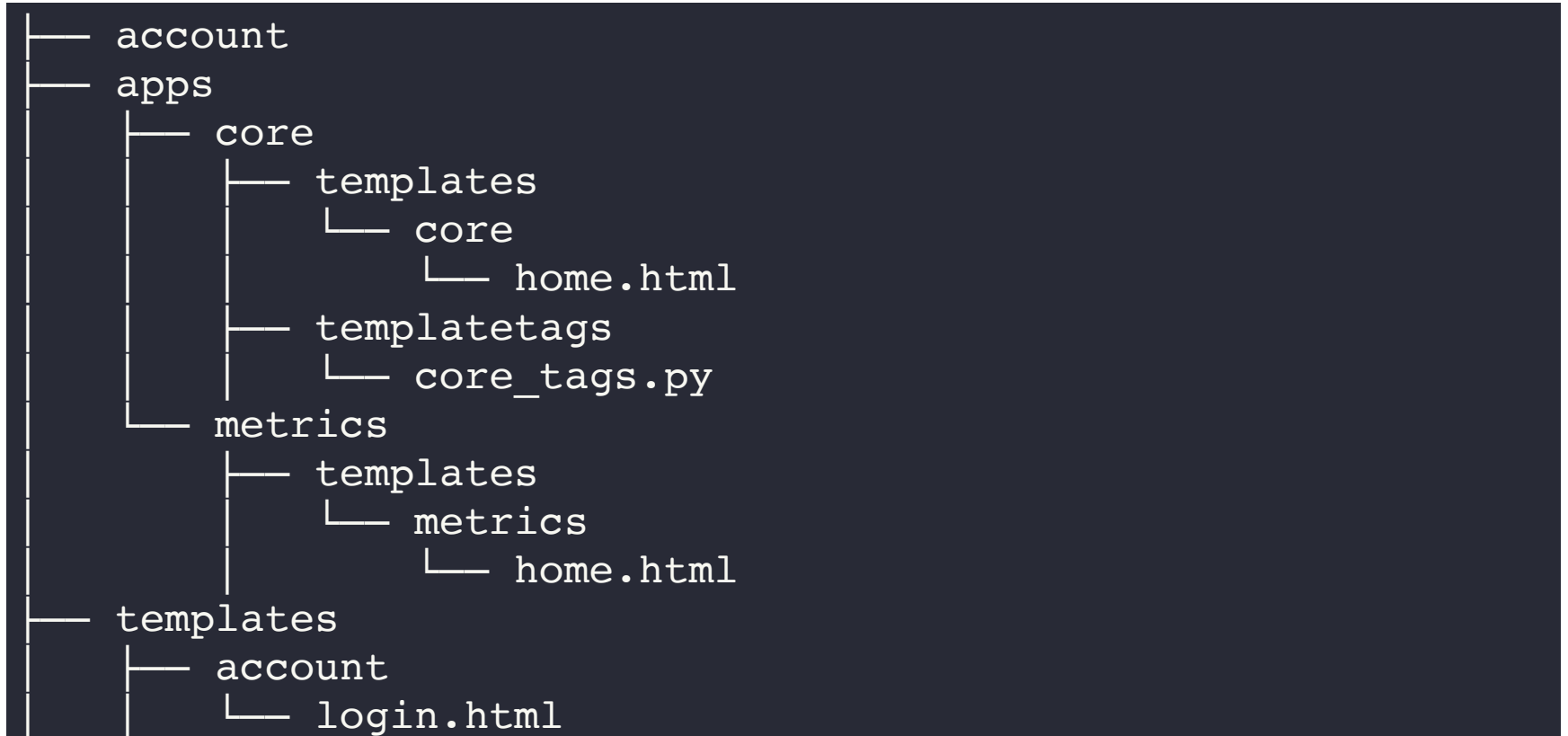
# REST API

```
{
  "nombre": "USERS",
  "long_tipo": "Custom",
  "tipo": "c",
  "alta": "2017-08-13T10:21:19.015000Z",
  "descripcion": "Return users ..."
},
{
  "nombre": "READ",
  "long_tipo": "Disk",
  "tipo": "d",
  "alta": "2017-08-13T10:15:09.516000Z",
  "descripcion": "Number of reads"
}
```

A man with dark hair, wearing a blue button-down shirt, is shown from the chest up. He has his arms raised in a celebratory gesture, with his hands near his head. He is smiling and looking towards the camera. The background is a home office with a bookshelf filled with books and a desk with a computer monitor. The lighting is warm and indoor.

**PRODUCTIVITY  
BOOST INCREASED!**

# FRONTEND (TEMPLATING)



# CONCEPTOS

- Herencia
- Lenguaje
- Custom Tags y Filtros

```
from django import template
register = template.Library()

@register.filter('en_grupo')
def en_grupo(user, group_name):
    groups = user.groups.all().values_list('name', flat=True)
    return True if group_name in groups else False
```

```
{% extends 'base.html' %}
{% load core_tags %}
{% block content %}

{% if request.user|en_grupo:"pycones" or user.is_superuser %}
    <h1>Metrics</h1>
    <a class="btn" href="{% url 'metrics:home' %}">Acceder
{% endif %}

{% if not user.is_authenticated %}
    <button type="button" class="close"></button>Longin
{% endif %}
{% endblock %}
```







CUTE!

**Y ESTO ...**

**¿CÓMO SE VE EN EL MÓVIL?**



¿OLA K ASE?

# MEJORANDO EL FRONTEND

- Bootstrap
- Javascript

```
var v = new Vue({
  delimiters: ['[[', ']]'],
  el: '#container',
  data: {
    metricas: []
  },
  mounted: function(){
    $.get('/metrics/api/v1/metricas/', function(data){
      v.metricas = data;
    })
  })
})
```

```
<table>
  <tbody>
    <tr v-for="metrica in metricas">
      <td>[[ metrica.nombre ]]</td>
      <td>[[ metrica.long_tipo ]]</td>
    </tr>
  </tbody>
</table>
```



# PERMISOS

- Templates
- Views
- Serializers

```
@method_decorator(login_required, name='dispatch')
class Home(UserPassesTestMixin, View):
    def test_func(self):
        return super_o_esta_en_grupo(self.request.user)

    def get(self, request, *args, **kwargs):
        return render(request, 'metrics/home.html')
```

```
class EsSuperOTienePermisosPorGrupo(BasePermission):
    def has_permission(self, request, view):
        gr_auth_map = getattr(view, 'grupos_autorizados', {})
        gr_auth = gr_auth_map.get(request.method, [])
        return any([super_o_esta_en_grupo(request.user, grupo)
                    for grupo in gr_auth])
```



# DEPLOY

- NGINX
- Gunicorn
- Docker



**¡MUCHAS GRACIAS!**