Django

Desarrollo web en Python

Edwin Caldón

http://vultur.unicauca.edu.co

Jornadas de Software Libre 2007
JSL2007

Agenda

- Arquitectura
- Ejemplo
  - Estructura (proyecto y aplicación)
  - Definir Modelos
  - Vistas
  - Plantillas
- Otros Frameworks
- Links

Django

« Django es un framework de Web de alto nivel en Python que estimula el desarrollo rápido y el diseño limpio y pragmático »

Framework ?

- URL's limpias
- Componentes con bajo acoplamiento
- Diseño amigable de templates
- Poco código como sea posible
-
- Desarrollo realmente rápido

Arquitectura

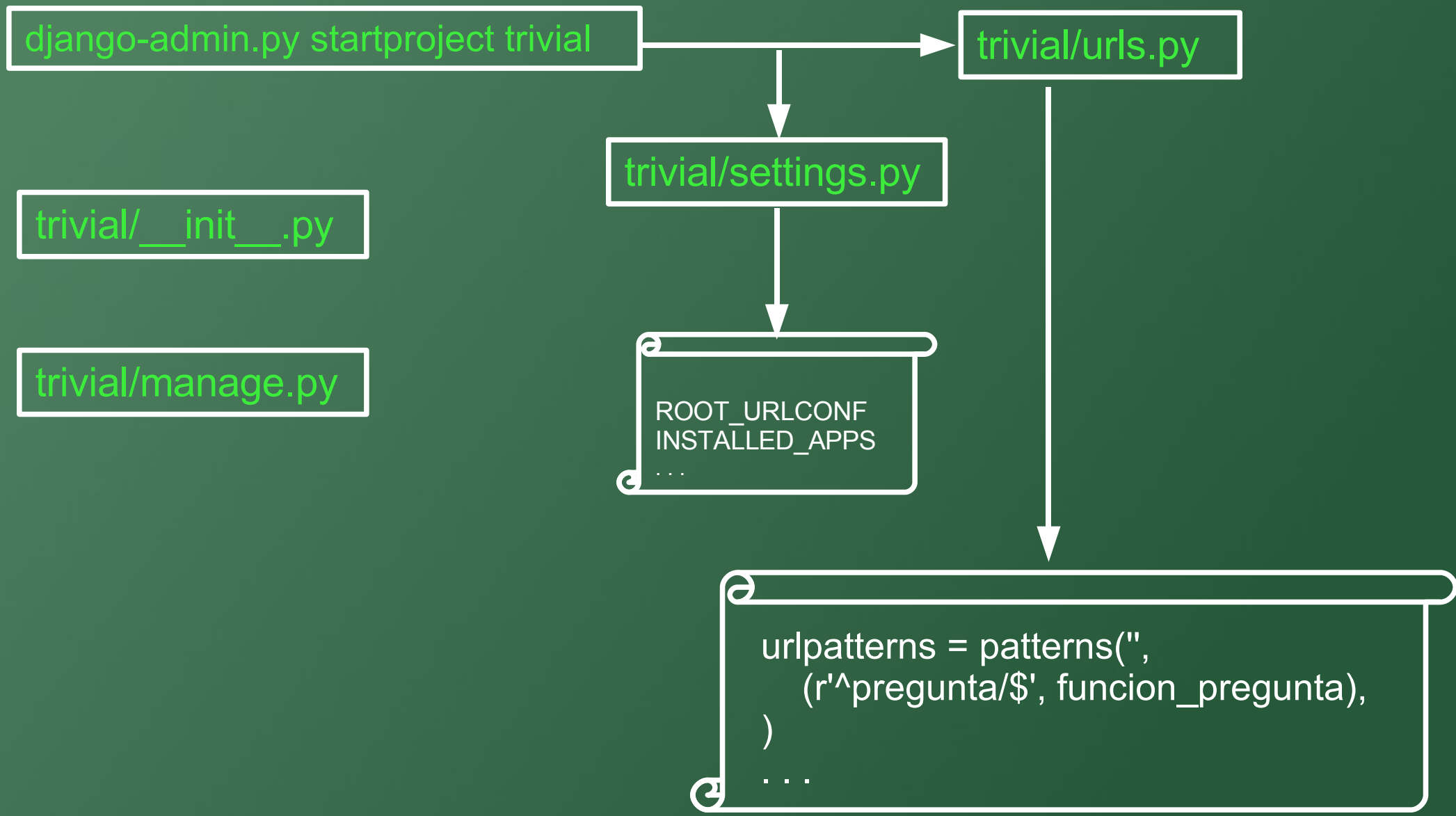
Lógica de Presentación	V	T	Templates
Lógica de Negocio	C	V	Views, URLConf (Models, Templates)
Lógica de Acceso a datos - Persistencia	M	M	Models, Data Access

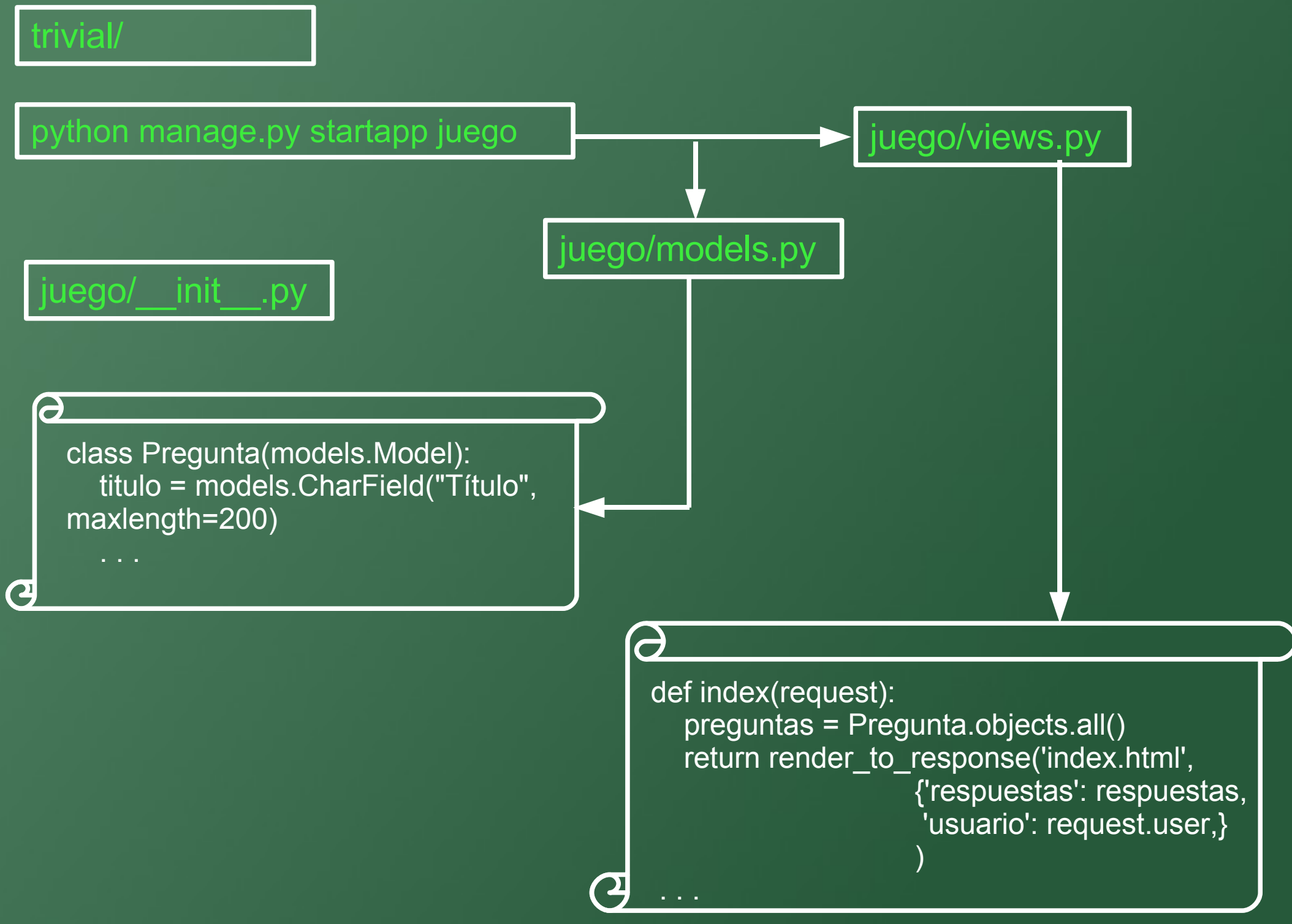
Ejemplo

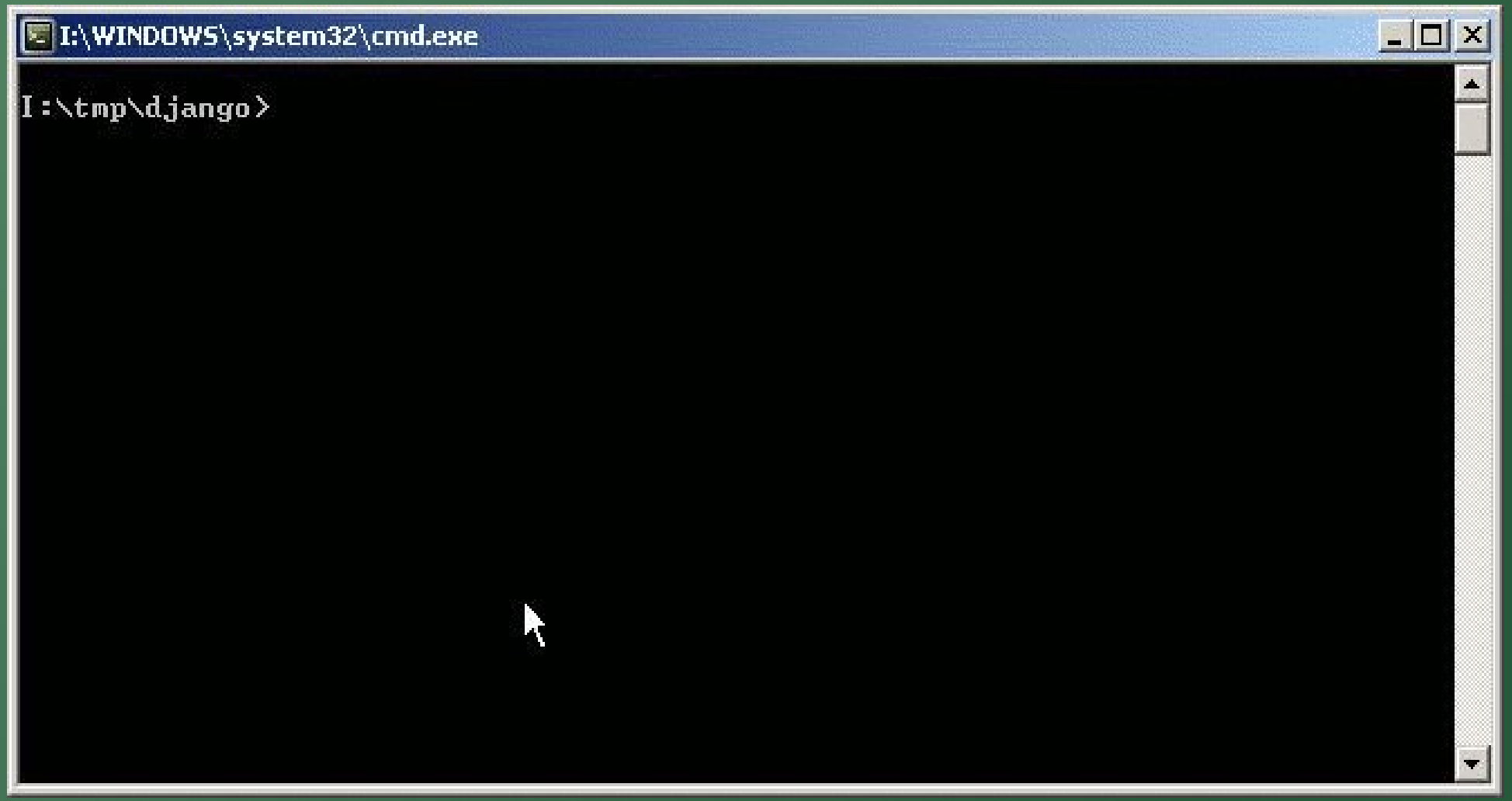
Juego de preguntas (Trivial)

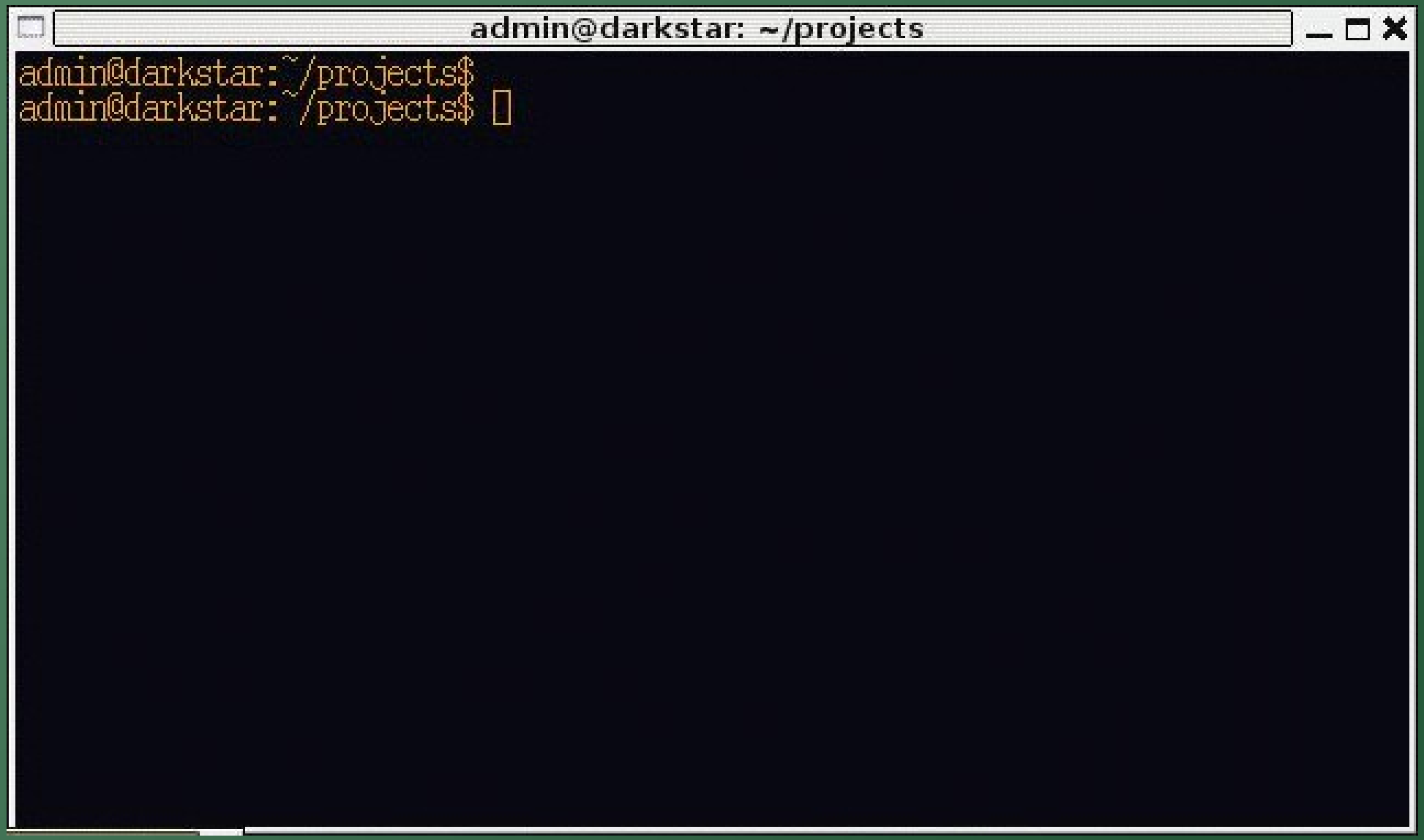
- Las preguntas y los usuarios los crea un administrador.
- Existen diferentes categorías de preguntas.
- Cada usuario tiene su propio juego (esto es, responde a sus preguntas).
- Es obligatorio estar validado en el sistema para jugar.

Estructura


django-admin.py startproject trivial
trivial/urls.py
trivial/settings.py
trivial/__init__.py
trivial/manage.py
ROOT_URLCONF
INSTALLED_APPS
. . .
urlpatterns = patterns('',
(r'^pregunta/$', funcion_pregunta),
)
. . .


trivial/
python manage.py startapp juego
juego/views.py
juego/models.py
juego/__init__.py
class Pregunta(models.Model):
titulo = models.CharField("Título",
maxlength=200)
. . .
def index(request):
preguntas = Pregunta.objects.all()
return render_to_response('index.html',
{'respuestas': respuestas,
'usuario': request.user,}
)
. . .


I:\WINDOWS\system32\cmd.exe
I:\tmp\django>


admin@darkstar: ~/projects
admin@darkstar:~/projects$
admin@darkstar:~/projects$

Definir Modelos - Models

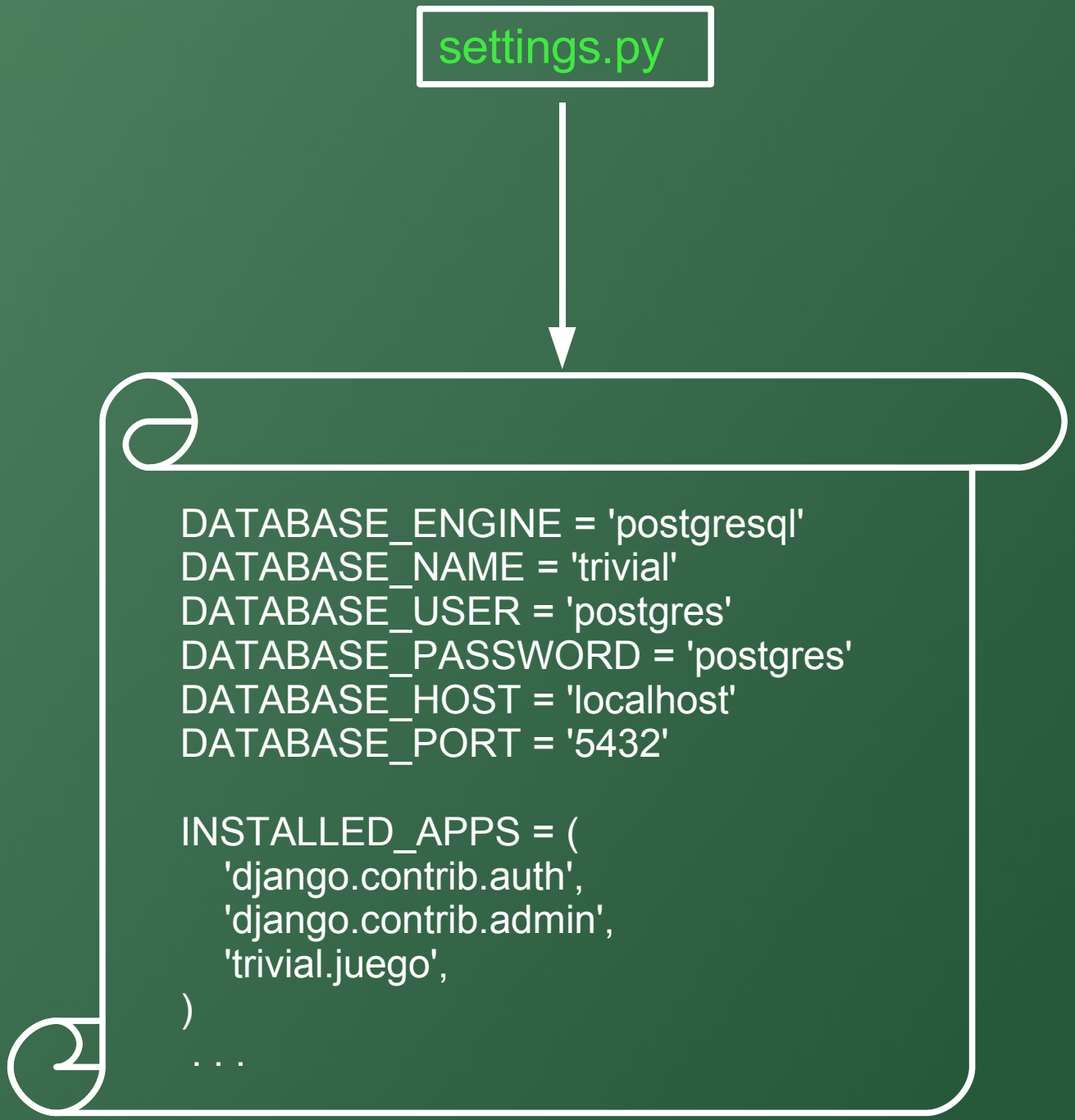
```
from django.db import models
from django.contrib.auth.models import User

class Pregunta(models.Model):
    titulo = models.CharField("Título", maxlength=200)
    texto = models.TextField("Texto de la pregunta")
    respuesta_1 = models.CharField(maxlength=200)
    respuesta_2 = models.CharField(maxlength=200)
    respuesta_3 = models.CharField(maxlength=200)
    respuesta_4 = models.CharField(maxlength=200)
    respuesta_correcta = models.CharField(maxlength=200)
    foto = models.CharField(maxlength=200)

    def __str__(self):
        return self.titulo

. . .
```

juego/models.py


settings.py
DATABASE_ENGINE = 'postgresql'
DATABASE_NAME = 'trivial'
DATABASE_USER = 'postgres'
DATABASE_PASSWORD = 'postgres'
DATABASE_HOST = 'localhost'
DATABASE_PORT = '5432'
INSTALLED_APPS = (
'django.contrib.auth',
'django.contrib.admin',
'trivial.juego',
)
. . .

$ **python manage.py sqlall juego**

```
CREATE TABLE "juego_pregunta" (
    “id” serial NOT NULL PRIMARY KEY,
    “titulo” varchar(200) NOT NULL,
    “texto” varchar(200) NOT NULL,
    “respuesta_1” varchar(200) NOT NULL,
    . . .
);
```

$ **python manage.py syncdb**

Creating table juego_pregunta

. . .

```
>>> from juego.models import Pregunta
>>> p = Pregunta(titulo='', texto='',
...     respuesta_1='', respuesta_2='', . . .)
>>> p.save()
```

Database API

- Operaciones CRUD
- Clase Modelo → Tabla
- Instancia de la clase → Registro

Interfaz de administración

Views - URLConf

URLConf (urls.py)

Despachador de URL's

http://www.example.com/pregunta/3/

```
from django.contrib.auth.views import login, logout
from django.conf.urls.defaults import *
from settings import MEDIA_ROOT

urlpatterns = patterns('',
 (r'^/?$', 'trivial.juego.views.index'),
 (r'^pregunta/(\d+)/$', 'trivial.juego.views.pregunta'),
 (r'^admin/', include('django.contrib.admin.urls')),
. . .
)
```

views (juego/views.py)

```
from django.shortcuts import render_to_response
from django.contrib.auth.decorators import login_required
from trivial.juego.models import Pregunta, Respuesta
@login_required

def pregunta(request, id):
    pregunta = Pregunta.objects.get(id=id)
    try:
        respuesta = Respuesta.objects.get(pregunta=id,
usuario=request.user)
    except ObjectDoesNotExist:
        respuesta = None
    return render_to_response('pregunta.html',
                              {'pregunta': pregunta,
                               'respuesta': respuesta,
                               'tiempo': str(int(time.time())),
                              }
                             )
```

Plantillas - Templates

Molde (guía) que contiene **variables**, las cuales se reemplazan cuando se evalúa la plantilla, y **tags** (for, extend, filter, load, include ...), las cuales controlan la lógica del template.

base.html

```
<html>
    <head>
        <title>
            {% block title %}{% endblock %}
        </title>
    </head>
    <body>
        {% block cuerpo %}{% endblock %}
        <div id=”footer”>
            {% block footer %}(c) 2007{% endblock %}
        <div>
    </body>
</html>
```

Herencia de Templates

Permite construir un “esqueleto” que contiene todos los elementos comunes del sitio y define **bloques** que los hijos pueden sobreescribir

```
{% extends "base.html" %}

{% block cuerpo %}

    <h2>Categoría: {{ pregunta.categoria }}</h2>
    <h3>{{ pregunta.titulo }}</h3>

    {% if texto_error %}
        <p class="error">{{ texto_error }}</p>
    {% endif %}

    <img class="foto" src="/site_media/{{ pregunta.foto }}">

    <p>{{ pregunta.texto }}</p>

    {% if respuesta %}
        <p>Ya has respondido antes a la pregunta.</p>
        <p>Tiempo empleado: {{ respuesta.tiempo }} segundos.</p>
        <p>El resultado fue
            {% if respuesta.resultado %}    CORRECTO
            {% else %}    INCORRECTO
            {% endif %}
        </p>
    {% else %}
        <form method="post" action="/responder/">
            <input type="hidden" name="pregunta" value="{{ pregunta.id }}">
            <input type="hidden" name="tiempo" value="{{ tiempo }}">
            <input type="radio" value="{{ pregunta.respuesta_1 }}"
                    name="respuesta">{{ pregunta.respuesta_1 }}<br>
         . . .
            <br>
            <input type="submit" value="Responder">
        </form>
    {% endif %}

{% endblock %}
```

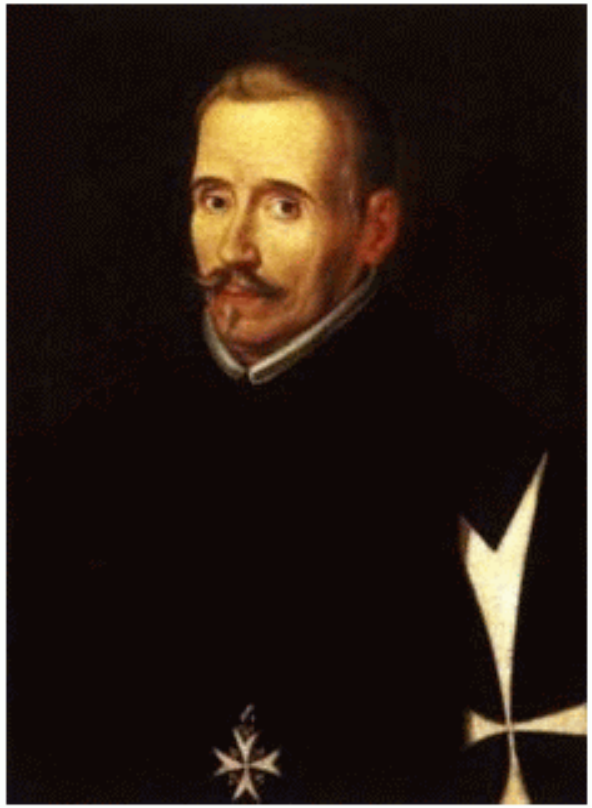
pregunta.html

http://localhost:8000/pregunta/1/

Juego Trivial | Tutorial de Django

Categoría: Literatura

Lope de Vega



¿Cuál de éstas obras NO está escrita por Lope de Vega?

Ya has respondido antes a la pregunta.

Tiempo empleado: 3 segundos.

El resultado fue INCORRECTO

Inicio

http://localhost:8000/pregunta/3/

Google

Juego Trivial

Tutorial de Django

Categoría: Ciencia

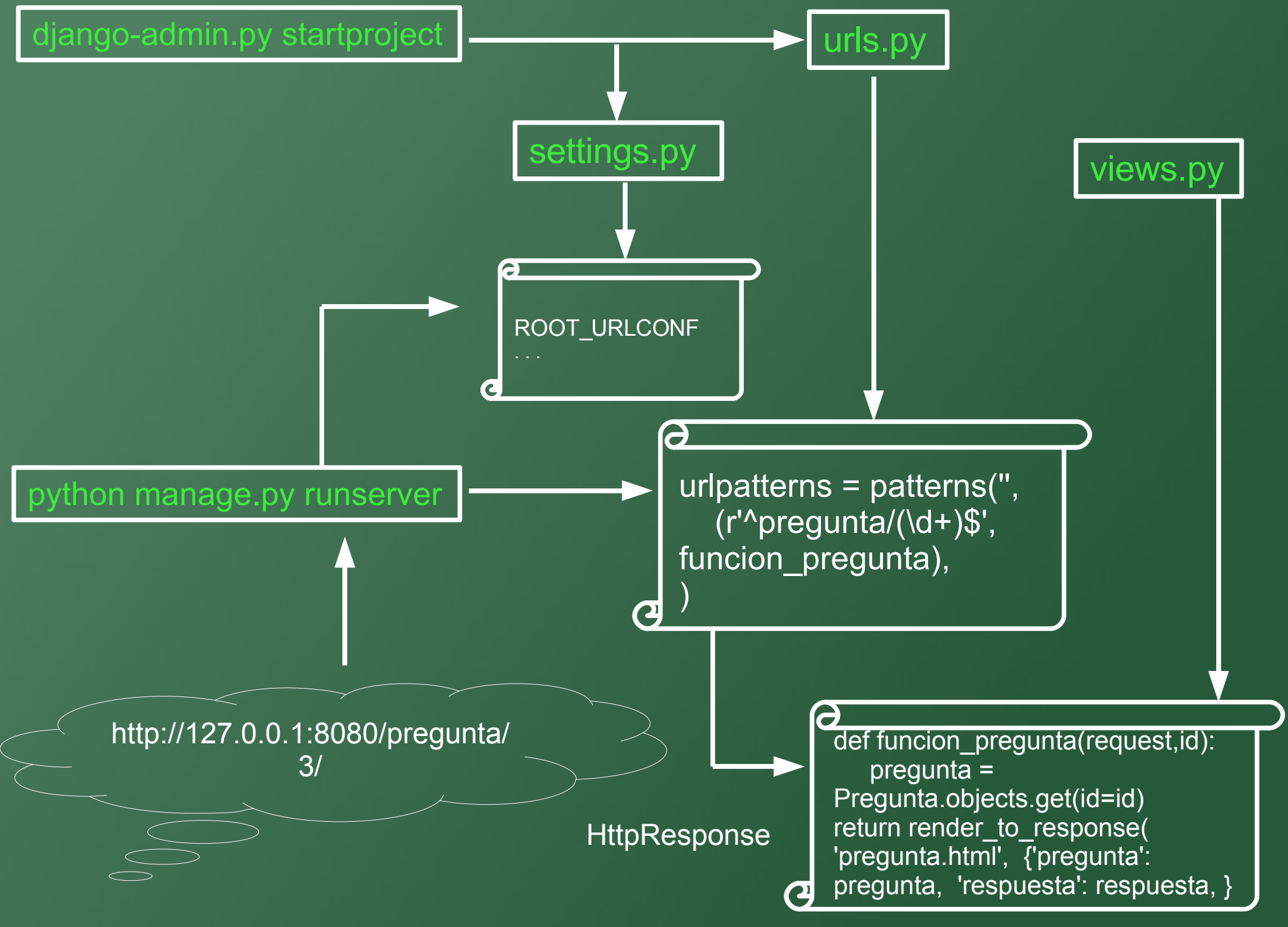
Vinos

Los organismos que permiten la fermentación del mosto para producir alcohol en el proceso de producción de vino son ...

- Hongos
- Bacterias
- Levaduras
- Virus

Responder

Inicio


django-admin.py startproject
urls.py
settings.py
views.py
ROOT_URLCONF
. . .
python manage.py runserver
urlpatterns = patterns('',
(r'^pregunta/(\d+)$',
funcion_pregunta),
)
http://127.0.0.1:8080/pregunta/
3/
HttpResponse
def funcion_pregunta(request,id):
pregunta =
Pregunta.objects.get(id=id)
return render_to_response(
'pregunta.html', {'pregunta':
pregunta, 'respuesta': respuesta, }

Otros Frameworks

Proyecto	*Lenguaje*	*I18N*	*ORM*	*Migrar DB*	*Templates*	*Seguridad*
Struts	Java	X	X		Jakarta Tiles	
Sympony	PHP5	X	Propel, Doctrine	Plugin	X	Plugin
Django	Python	X	Django ORM, SQLAlquemy	Desarrollo	X	ACL
Pylons	Python	X	SQLObject, SQLAlquemy		X	
TurboGears	Python	X	SQLObject, SQLAlquemy		Kid	
Ruby on Rails	Ruby	X, L10N	ActiveRecord	X	ERB, RJS	Plugin

http://en.wikipedia.org/wiki/Comparison_of_web_application_frameworks

Cosas por ver

- Filters
- Testing
- Django-LDAP
- Formularios
- OpenID
- Autenticación
- Internacionalización
- Ajax

Links

http://djangobook.com

http://djangoproject.com/documentation/

http://www.django.es [google groups]

http://davidasorey.net/static/django-tutorial/

Ben Askins and Alan Green.(2007). **A Rails / Django Comparison**. The Python Papers, Volume 2, Issue 2. Disponible en pythonpapers.org.

Gracias

Por su atención

Preguntas