Django - parte 2

Programación Web 2

Cambiando los tipo de campos

- Si se desea reiniciar o empezar desde cero con un modelo:
  - Borrar los archivos dentro de “migrations: 00...”, la carpeta “_pycache_” y “db.sqlite3”
  - Con estos pasos se borrarán incluso todos los datos o usuarios que hayamos creado antes.
  - Aunque en esta nueva versión es preferible modificar a empezar de nuevo
- El modelo Persona, con tipos de datos más apropiados para sus campos.

```python
from django.db import models

# Create your models here.
class Persona(models.Model):
  nombre    = models.CharField(max_length = 100)
  apellidos = models.CharField(max_length = 100)
  edad      = models.IntegerField()#(max_digits=3)
```

```
(prueba) apaz src $ python manage.py makemigrations
Migrations for 'personas':
  personas/migrations/0002_auto_20200607_1816.py
    - Alter field apellidos on persona
    - Alter field edad on persona
    - Alter field nombre on persona
```

```
(prueba) apaz src $ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, personas, sessions
Running migrations:
  Applying personas.0002_auto_20200607_1816... OK
```

Viendo los cambios hechos de manera automática


```
(prueba) apaz src $ python manage.py shell
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 03:13:28)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from personas.models import Persona
>>> Persona.objects.create(nombre="Jorge", apellidos="Gonzales", edad=18
<Persona: Persona object (3)>
```

- ¿Qué pasa si añadimos un nuevo campo? los anteriores registros no lo tendrían, entonces ¿Con qué valor se actualizarán?

Cambiando el modelo

```python
from django.db import models

# Create your models here.
class Persona(models.Model):
  nombres   = models.CharField(max_length = 100)
  apellidos = models.CharField(max_length = 100)
  edad      = models.IntegerField()#(max_digits=3)
  donador   = models.BooleanField()
```

Modificar persona

Nombres: Jorge

Apellidos: Gonzales

Edad: 45

Donador

Eliminar

```
(prueba) apaz src $ python manage.py makemigrations
You are trying to add a non-nullable field 'donador' to persona without a defaul
t; we can't do that (the database needs something to populate existing rows).
Please select a fix:
 1) Provide a one-off default now (will be set on all existing rows with a null
value for this column)
 2) Quit, and let me add a default in models.py
Select an option: 1
Please enter the default value now, as valid Python
The datetime and django.utils.timezone modules are available, so you can do e.g.
 timezone.now
Type 'exit' to exit this prompt
>>> True
Migrations for 'personas':
  personas/migrations/0005_persona_donador.py
    - Add field donador to persona
(prueba) apaz src $ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, personas, sessions
Running migrations:
  Applying personas.0005_persona_donador... OK
```

Actualizando los campos antiguos

- Dejar a todos los campos antiguos con el mismo valor, es peligroso!
  - null = True, es legal que el campo se quede sin valor
  - default=True, el valor por defecto será True
  - blank=True significa que se puede dejar en blanco (no es requerido)
- ¿Cómo crearía un campo que sea obligatorio?
  - En el formulario los campos obligatorios aparecen en negrita (blank = False)
- ¿Cuáles de estos elementos afectarían a la base de datos?¿Cuáles no?

https://docs.djangoproject.com/en/3.2/ref/models/fields/

Personalizar la página de inicio

- Esto requiere trabajar con la VISTA (¿Recuerda el patrón MVC?)
- Se requiere crear una clase o función en la vista de una app
- Para practicar crearemos una nueva app llamada inicio
  - python manage.py startapp inicio
  - vim listaContacto/settings.py
    - `INSTALLED_APPS = [... ‘inicio’, ]`
  - vim inicio/views.py

```
from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.
def myHomeView(*args, **kwargs):
  return HttpResponse('<h1>Hola Mundo desde DJango</h1>')
```

Registrando el URL de la nueva página de inicio

- vim listaContactos/urls.py


```
"""primero URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/2.1/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path
from inicio.views import myHomeView

urlpatterns = [
    path('', myHomeView, name='Página de Inicio'),
    path('admin/', admin.site.urls),
]
```

Peticiones y ruteos de URL

- Analizaremos la relación entre patrones de URL y Vistas
  - **View** determina la **lógica**, osea **qué** contenido
  - **URLConfs** determina **dónde** estará el view
  - URLConfs usar una expresiones regulares para mapear una petición a una vista
- vim listaContactos/urls.py


```
from django.contrib import admin
from django.urls import path
from inicio.views import myHomeView

urlpatterns = [
    path('', myHomeView, name='Página de Inicio'),
    path('another/', myHomeView, name='Página de Inicio'),
    path('admin/', admin.site.urls),
]
```

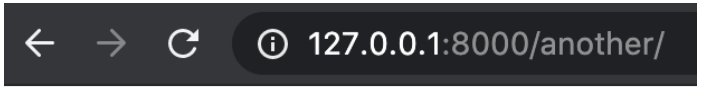
Nuevas vistas: ¡Cree más vistas!

- Agreguemos una nueva vista para la URL another
- vim inicio/views.py

```python
from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.
def myHomeView(*args, **kwargs):
  return HttpResponse('<h1>Hola Mundo desde DJango</h1>')

def anotherView(request):
  return HttpResponse('<h1>Sólo otra página</h1>')
```

- vim listaContactos/urls.py

```python
from django.contrib import admin
from django.urls import path
from inicio.views import myHomeView
from inicio.views import anotherView

urlpatterns = [
    path('', myHomeView, name='Pagina de Inicio'),
    path('another/', anotherView),
    path('admin/', admin.site.urls),
]
```



El argumento de la funciones de las vistas

- Las funciones de las vistas reciben un argumento, examinemos su contenido
- https://docs.djangoproject.com/en/3.2/ref/request-response/
- vim inicio/views.py

```python
def myHomeView(request, *args, **kwargs):
  print(args, kwargs)
  print(request.user)
  return HttpResponse('<h1>Hola Mundo desde DJango</h1>')
```

- En la consola del servidor

```
() {}
carlocorrales
[29/May/2021 17:01:39] "GET /another/ HTTP/1.1" 200 24

() {}
AnonymousUser
[29/May/2021 17:02:27] "GET /another/ HTTP/1.1" 200 24
```

Plantillas (Templates) de Django

- Los templates son la forma que tiene Django para generar código HTML
  - URL es la ruta
  - Views es la lógica
  - Template es el HTML
- mkdir templates
- vim templates/home.html
- vim listaContactos/settings.py
- vim inicio/views.py

```
<!DOCTYPE html>
<html>
  <head>
    <title>Template Base</title>
  </head>

  <body>
    <h1>Hola Mundo desde DJango</h1>
    <h2>Con Templates</h2>
  </body>
</html>
```

```
import os
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.D
        'DIRS': [os.path.join(BASE_DIR,"templates")],
        'APP_DIRS': True,
        'OPTIONS': {
```

```
def myHomeView(request, *args, **kwargs):
  print(args, kwargs)
  print(request.user)
  return render(request,"home.html",{})
```

¡CREE SUS PROPIAS PLANTILLAS!

Django template language (DTL): Variables

- Una variable muestra un valor según el contexto. Las variables se encuentran encerradas entre llaves dobles {{variable}}
- En el contexto de nuestro ejemplo podemos usar al objeto “request” como variable:
  - {{ request.user }} muestra el usuario en una pag web Django.
  - {{ request.user.is_authenticated }} nos indica si está logeado.
- vim templates/home.html


```
<!DOCTYPE html>
<html>
  <head>
    <title>Template Base</title>
  </head>

  <body>
    <h1>Hola Mundo desde DJango</h1>
    <h2>Con Templates</h2>
    {{ request.user }}
    <br>
    {{ request.user.is_authenticated }}
  </body>
</html>
```

Revise la documentación sobre el objeto request y pruebe otras de sus propiedades (atributos o métodos)

Django template language (DTL): Tags

- Para DTL, un tag :
  - Puede servir como una estructura de control ("if" o un ciclo "for")
  - Puede servir como estructura de bloque (las llaves en lenguajes como java)
  - Puede servir para sacar contenido de una base de datos o incluso dar acceso a otros tags.
  - Los tags están encerrados entre llaves con signos de porcentaje `{% tag %}`
- Usando el tag block
  - Editar una plantilla base
  - vim templates/base.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Codigo para Estudiantes desde Django</title>
  </head>

  <body>
    <h1> Este es un texto de Base</h1>
    {% block content %}
      Reemplazame
    {% endblock %}
  </body>
</html>
```

https://docs.djangoproject.com/en/3.2/ref/templates/language/#template-inheritance

El tag extends

Contando con la plantilla base, se pueden definir nuevos y antiguos templates reusando algo de código, en home.html:

```
{% extends 'base.html' %}
{% block content %}
<h2>Hola Mundo desde DJango</h2>
<h3>Con Templates</h3>
{{ request.user }}
<br>
{{ request.user.is_authenticated }}
{% endblock %}
```

Realice sus propias plantillas usando la Base

Cree más bloques

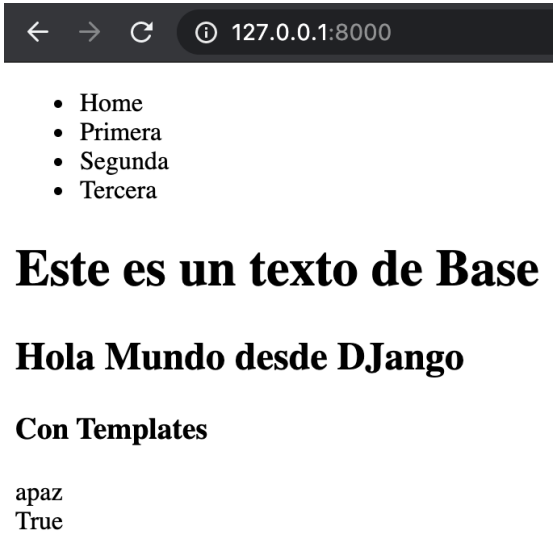
https://docs.djangoproject.com/en/3.2/ref/templates/builtins/#extends

El tag include

- Es la manera de incrustar código de otros templates
- vim templates/nav.html
- vim templates/base.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Codigo para Estudiantes desde Django</title>
  </head>

  <body>
    {% include 'nav.html' %}
    <h1> Este es un texto de Base</h1>
    {% block content %}
      Reemplazame
    {% endblock %}
  </body>
</html>
```

```
<nav>
  <ul>
    <li>Home</li>
    <li>Primera</li>
    <li>Segunda</li>
    <li>Tercera</li>
  </ul>
</nav>
```



Verifique las opciones de navegación que aparecen en el resto de sus vistas

https://docs.djangoproject.com/en/3.2/ref/templates/builtins/#include