**TUTORIAL DJANGO**

**INSTALACIÓN**

Primero deberás de instalar python3 y django.

Para installar python 3 recomiendo instalar anaconda: <https://www.anaconda.com/download/#windows>

Durante la instalación no olvides de seleccionar el casillero donde se dice para instalarlo en el path.

Luego, deberás de instalar django. Para esto, deberás de ir a “command shell” y poner:

pip install django

Esto instalará django.

**CREACIÓN DE UN PROYECTO**

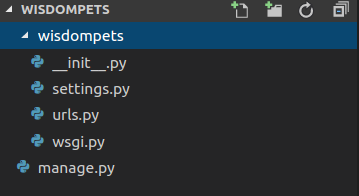
Primero en “command shell” ubicarte en donde quieres crear el proyecto. Cuando crees el proyecto crearás una carpeta. Por ejemplo, si queremos que el nombre del proyecto sea “wisdompets” deberás de poner en el command shell:

django-admin.py startproject wisdompets

Esto creará una nueva carpeta llamada wisdompets en la ubicación donde te encuentras. De no funcionar, prueba escribiendo

django-admin startproject wisdompets

A veces, no debes de poner el “.py” esto depende de multiples factores, el sistema operativo, la versión de windows que uses, la versión de python que uses, entre otros.



Cuando crees el proyecto se creará una carpeta llamada wisdompets, dentro de esta carpeta se crearán distintos archivos “.py”.

Una vez creado el proyecto puedes probar que todo funciones entrando a la carpeta creada en “command shell” con (en este caso, cd wisdompets), dentro de la carpeta del proyecto pon:

python manage.py runserver

Esto abrirá un puerto, normalmente el 8000. Así que anda a un navegador web y pon

localhost:8000

Si todo anda bien podrás entrar a una pantalla de bienvenida de django

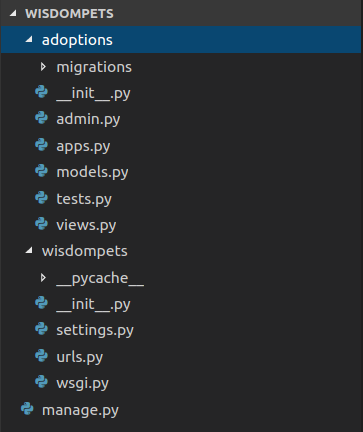
**DESARROLLO DEL PROYECTO**

El primer paso para desarrollar tu proyecto será la creación de un App. Un proyecto puede tener varias Apps. De esta forma se puede organizar el proyecto. Para crear el App, dentro del “command shell” dentro de la ubicación del proyecto pon:

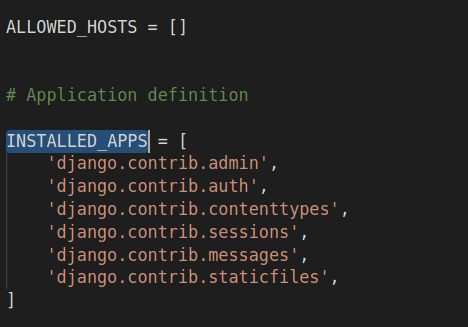
python manage.py startapp adoptions

Este comando creará una nueva carpeta dentro de la carpeta de tu proyecto de nombre “adoptions”. Este es el nombre del App.

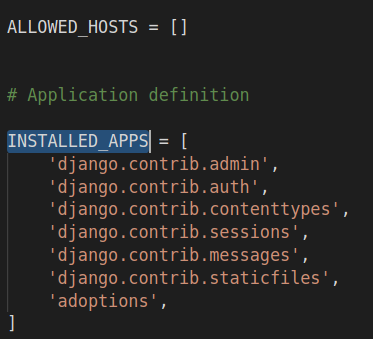
Esta carpeta también tendrá distintos archivos “.py”



Para hacerle saber a “django” que has creado un nuevo App, dentro de la carpeta wisdompets, en el archivo “settings.py” deberás de indicar el nombre del App que has creado. Para esto, dentro de la lista “INSTALLED\_APPS” deberás de incluirlo.



Deberás de poner, al final de la lista la nueva app “adoptions”



**CREACIÓN DEL MODELO DE DATOS**

El modelo de datos describe los datos que se usarán en la aplicación, en este caso la aplicación mostrará información sobre mascotas y sobre las vacunas de las mascotas.

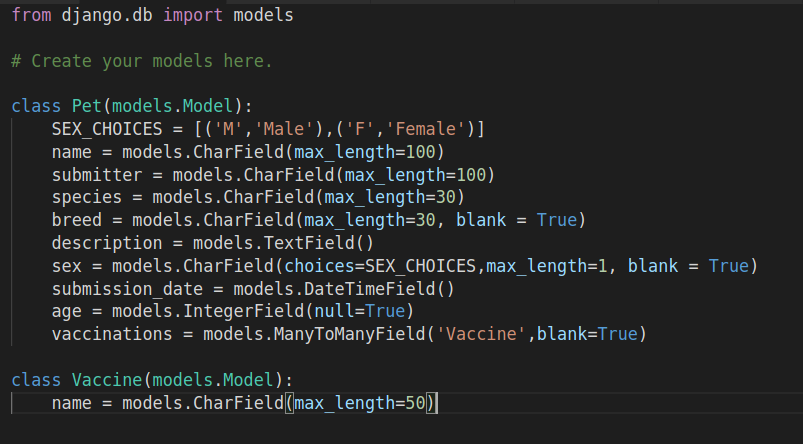
Las variables que tendrás las mascotas serán las siguientes:

|  |  |  |
| --- | --- | --- |
| **Variable** | **Descripción** | **Tipo** |
| name | Nombre de la mascota | CharField |
| submitter | Persona que está dando en adopción la mascota | CharField |
| species | Especie de la mascota (perro, gato) | CharField |
| breed | Raza de la mascota. Puede no tener raza | CharField |
| description | Descripción de la mascota en adopción | TextField |
| sex | Sexo de la mascota (Puede ser nulo | CharField  With choices |
| submission\_date | Fecha que se registro la mascota | DateTimeField |
| age | Edad de la mascota | IntegerField |
| vaccinations | Vacunas que tiene la mascota | ManyToManyField |

El modelo de vacunas, solo tendrá una variable que será el nombre.

|  |  |  |
| --- | --- | --- |
| **Variable** | **Descripción** | **Tipo** |
| Name | Nombre de la vacuna | CharField |

Para crear el modelo de datos dentro de django deberás de ir a la carpeta donde se encuentra en App *adoptions* y deberás modificar el archivo “models.py”



Luego de añadir el código del modelo de datos al archivo “models.py” deberás “decirle” a django que incluya el modelo de datos. Para esto, en “command shell”, en la ubicación del proyecto general deberás de poner:

python manage.py makemigrations

Luego deberás de poner:

python manage.py migrate

Con estos dos comandos, crearás el modelo de datos en la base de datos “db.sqlite3” que se creó cuando creaste el proyecto.

Si quieres ver la base de datos te debes primero de bajar un visualizador de sqlite. Una opción es entrar a: <http://sqlitebrowser.org/> y bajarte el visualizador que se encuentra en la página.

**CARGAR INFORMACIÓN**

Al crear el modelo, estás creando la base de datos con los campos, pero todavía no tienes informacion en esta base de datos.

Para esto del repositorio de github <https://github.com/carlosgl87/desarrollo_soluciones_web/tree/master/django_tutorial/assets>

te deberás de bajar y copiar a tu proyecto la carpeta “management” y el archivo “pet\_data.csv”.

El archivo “pet\_data.csv” contiene un archivo con información de las mascotas. Esta contiene los mismos campos que la base de datos que hemos creado. La carpeta contiene un script que te permitirá copiar la información a la base de datos.

La carpeta pegala dentro de la carpeta adoptions y el archivo con la información crealo en el repositorio inicial.

Luego en “command shell” en la ubicación inicial del proyecto deberás de poner:

python manage.py load\_pet\_data

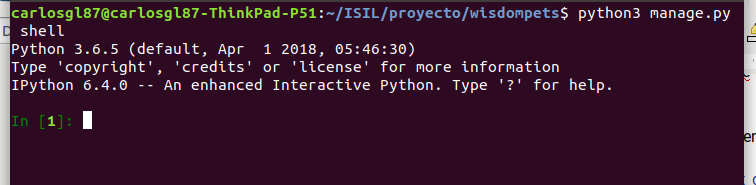
Esto ejecutará el programa de la carpeta “management” y copiará la información del archivo “.csv” a la base de datos en sqllite.

**INTERACTUAR CON LA BASE DE DATOS**

Una vez que tienes la base de datos con datos de mascotas puedes interactuar con esta base de datos.

Para esto dento de la carpeta inicial deberás de poner:

python manage.py shell



Al hacer esto dentro del shell que estes usando podrás realizar consultas a la base de datos.

Para esto, primero deberás de importar el modelo de mascotas. Para esto primero coloca:

from adoptions.models import Pet

Con esto podrás acceder a la información que se encuentra en la base de datos. Por ejemplo, puedes “pedir” a la base de datos todos los objetos, para esto deberás de poner:

pets = Pet.objects.all()

Pets, en este caso será una lista de objetos. Como toda lista, para acceder al primer elemento de la lista deberás de poner la posición 0 dentro de corchetes

pet = pets[0]

La variable pet será un objeto que tiene la información de la primera mascota de la base de datos, para acceder a su nombre poner:

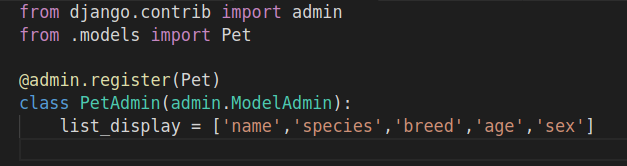
pet.name

Por otro lado, si se quiere sacar información de la mascota con id 1, se deberá de poner:

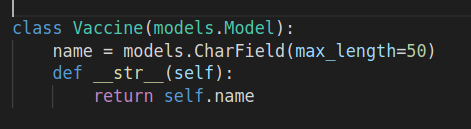
pet = Pet.objects.get(id=1)

**CREAR UNA VISTA DE ADMINISTRADOR**

Para crear una vista de administrador dentro de la carpeta adoptions tienen que modificar el archivo “admin.py”, dentro de este archivo primero llamarán a la modelo de mascotas y luego crearán una clase PetAdmin.



Luego, para poder ver dentro de la vista administrador el nombre de las vacunas deberás de cambiar el modelo de vacunas. Para esto, dentro de la carpeta de adoptions, dentro del archivo “models.py” deberás de hacer el siguiente cambio



Luego deberás de crear el super usuario, es decir el usuario administrador, para esto dentro de “command shell” dentro de la carpeta inicial deberás de poner:

python manage.py createsuperuser

Luego deberás de darle el nombre del usuario y el password.

Finalmente, para poder acceder a la pantalla del administrador deberás de primero entrar a la carpeta inicial en “command shell” y poner

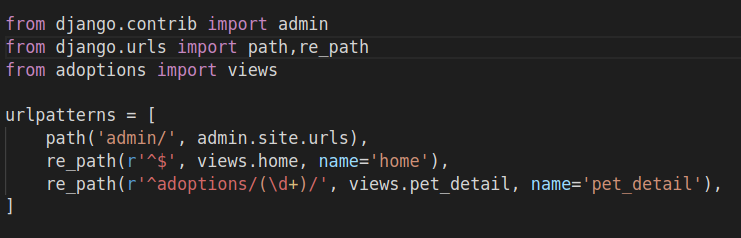
python manage.py runserver

Luego podrás entrar desde un navegador llendo a la dirección localhost:8000/admin

**CREACIÓN DE LAS VISTAS DE LA APLICACIÓN**

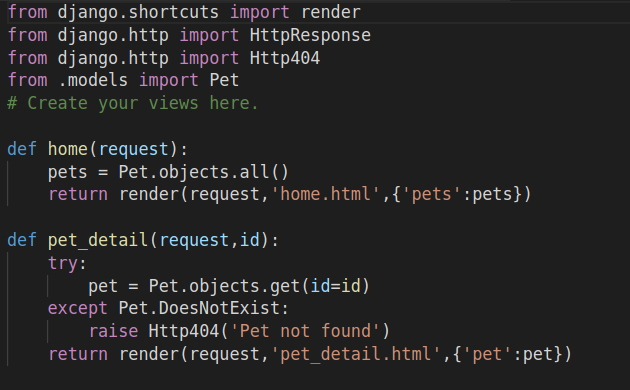
Primero deberás de modificar el archivo “urls.py” que se encuentra dentro de la carpeta wisdompets.

En este archivo se deberá de crear la lógica de como acceder a las vistas, es decir, como accederas a las vistas mediante un url. Para esto se usará expresiones regulares.



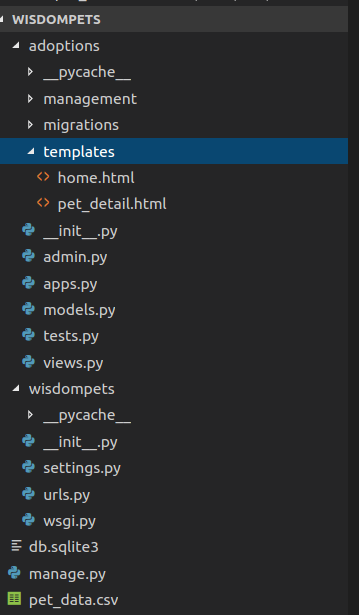
Con estos paths, si no se pone nada en el url, en este caso si se pone ***localhost:8000/*** se irá a la vista home. Por otro lado, al poner el url ***localhost:8000/adoptions/1/***, se irá a la vista del detalle de mascotas (pet\_detail) y se verá el detalle de la mascota 1.

Luego deberás de crear las vistas, para esto dentro de la carpeta adoptions, se tendrá que modificar el archivo “views.py”. En este archivo se crearán las dos vistas: home y pet\_detail.

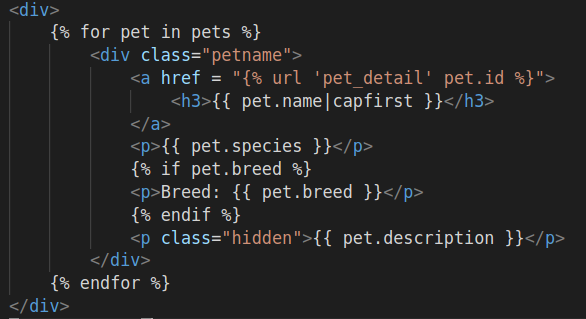


En estas vistas se llaman a dos archivos html: home.html y pet\_detail.html. A cada uno de estos archivos se le pasará información.

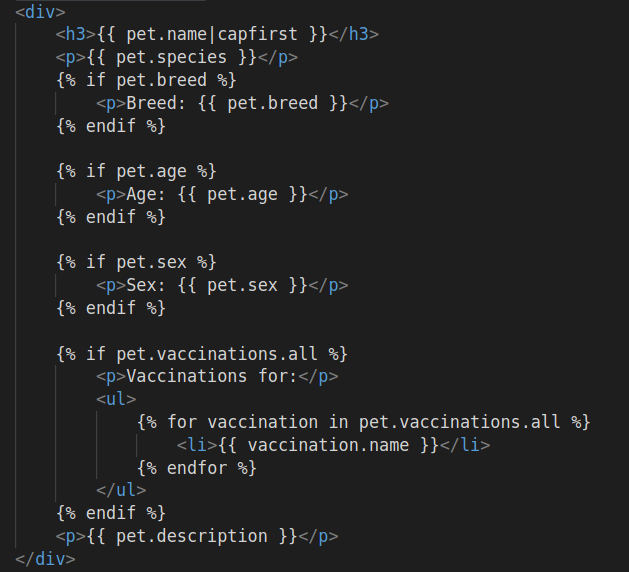
Por lo que el siguiente paso es crear los archivos html, para esto dentro de la carpeta adoptions, crea una carpeta templates, en esta carpeta crea dos archivos: home.html y pet\_detail.html.



Primero desarrollaremos el archivo home.html. Esta vista lo que debe de generar es una lista de todas las mascotas y mostrar algunos detalles. Para esto dentro de este archivo html deberá de decir lo siguiente



Por otro lado, en el archivo pet\_detail.html, se deberá mostrar información detallada de cada mascota, solo de una mascota. Para esto en este archivo deberá de decir lo siguiente:



Con estos dos archivos ya podremos ver la aplicación al iniciar el servidor. Para esto vamos a la carpeta inicial en “command shell” y ponemos

python manage.py runserver

Y luego accedemos a la aplicación a través de un navegador web en localhost:8000

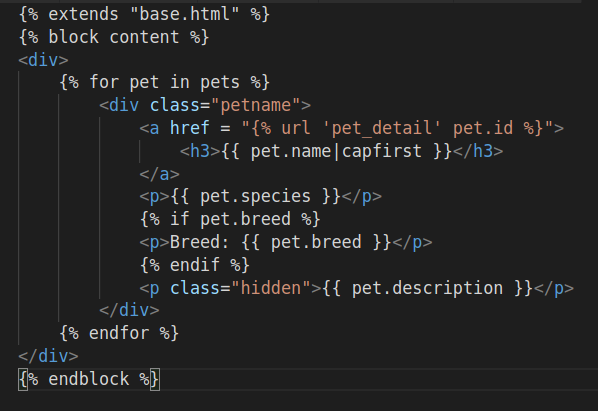
**FORMATO / ESTILO**

La aplicación ya esta lista y las lógicas terminadas. Lo que toca ahora es darle un poco de estilo. Para esto primero vamos a dentro de la carpeta templates un archivo llamado “base.html” este archivo nos ayudará a darle estilo a nuestra aplicación.

El código que irá dentro de este archivo se encuentra en <https://github.com/carlosgl87/desarrollo_soluciones_web/tree/master/django_tutorial/assets>

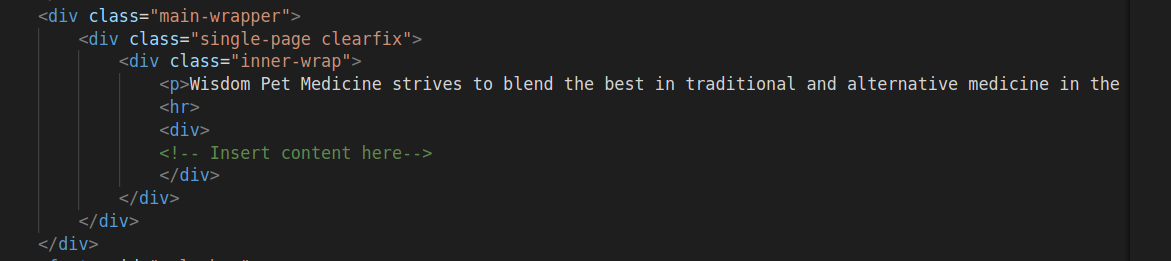
El archivo llamado “starter\_template.html”. El contenido de este archivo copienlo dentro del archivo base.html

Luego para que cada una de las vistas llame a este archivo, se tendrán que modificar los archivos home.html y pet\_detail.html. En los dos se deberán de poner dos lineas antes de código y una al final como se muestra acontinuación

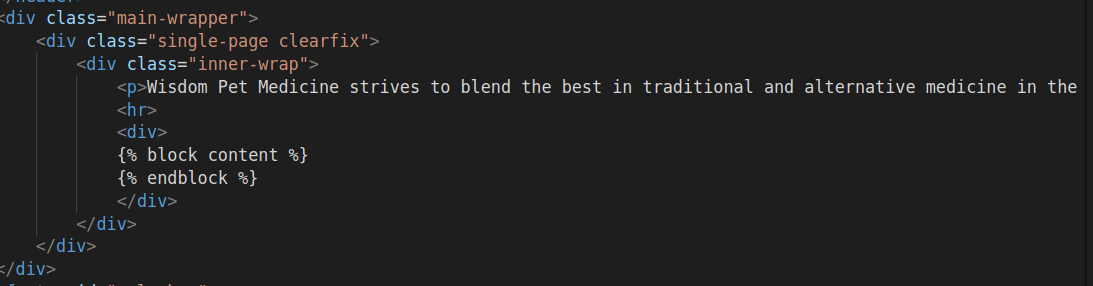


Luego el archivo base.html deberás de modificarlo.

Llegando al final del archivo base.html dentro de la clase “main-wrapper” se encuentra lo siguiente:



Deberás de reemplazar “Insert content here” por:



Luego inicia el servidor y mira los resultados.