**ACCESO DATOS ORACLE DJANGO**

* Vamos a visualizar cómo podemos acceder a datos dentro del entorno de django.
* Lo primero que debemos hacer es ejecutar, en nuestro entorno virtual, la instrucción: **pip install oracledb**



* En django no tenemos ningún lugar dónde incluir el concepto que tenemos de Servicios, es decir, tener un servicio desde el que podamos recuperar datos.
* Para poder utilizar **Servicios**, es decir, clases de acceso a datos, es necesario hacerlo desde una única clase llamada **models.py**
* En dicha clase, iremos incluyendo las diferentes clases que vayamos teniendo de acceso a datos.

Tabla

El contenido generado por IA puede ser incorrecto.

Ejemplo de **MODELS.PY**

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Necesitamos también, dentro de **models.py**, las clases que vayamos a devolver

**Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.**

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.

**MODELS.PY**

**from django.db import models**

**import oracledb**

**# Create your models here.**

***class* Departamento:**

**numero = 0**

**nombre = ""**

**localidad = ""**

***class* ServiceDepartamentos:**

***def* \_\_init\_\_(*self*):**

***self*.*connection* = oracledb.connect(*user*='SYSTEM', *password*='oracle', *dsn*='localhost/xe')**

***def* getDepartamentos(*self*):**

**sql = "select \* from DEPT"**

**cursor = *self*.*connection*.cursor()**

**cursor.execute(sql)**

**departamentos = []**

**for row in cursor:**

**dept = Departamento()**

**dept.*numero* = row[0]**

**dept.*nombre* = row[1]**

**dept.*localidad* = row[2]**

**departamentos.append(dept)**

**cursor.close()**

**return departamentos**

Posteriormente, debemos utilizar el Servicio creado en **models.py** dentro de **views.py**

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

**VIEWS.PY**

***def* datosDepartamentos(*request*):**

**servicio = ServiceDepartamentos()**

**departamentos = servicio.getDepartamentos()**

**context = {**

**"departamentos": departamentos**

**}**

**return render(request, 'informacion/datos.html', context)**

Y ya podremos realizar el dibujo que deseemos, por ejemplo, una tabla

**DEPARTAMENTOS.HTML**

**<!DOCTYPE html>**

**<html lang*="en"*>**

**<head>**

**<meta charset*="UTF-8"*>**

**<meta name*="viewport"* content*="width=device-width, initial-scale=1.0"*>**

**<title>Document</title>**

**</head>**

**<body>**

**{% include 'informacion/menu.html' %}**

**<h1>Hola mundo</h1>**

**<table border*="1"*>**

**<tr>**

**<th>Id departamento</th>**

**<th>Nombre</th>**

**<th>Localidad</th>**

**</tr>**

**{% for d in departamentos%}**

**<tr>**

**<td>{{d.numero}}</td>**

**<td>{{d.nombre}}</td>**

**<td>{{d.localidad}}</td>**

**</tr>**

**{% endfor %}**

**</table>**

**</body>**

**</html>**

Necesitamos la misma práctica, pero con Hospitales

Tabla

El contenido generado por IA puede ser incorrecto.

Sobre **models.py** creamos una clase llamada **Hospital** y otra llamada **ServiceHospitales** con un método llamado **getHospitales()**

**MODELS.PY**

***class* Hospital:**

**idHospital = 0**

**nombre = ""**

**direccion = ""**

**telefono = ""**

**camas = 0**

***class* ServiceHospital:**

***def* \_\_init\_\_(*self*):**

***self*.*connection* = oracledb.connect(*user*='SYSTEM'**

**, *password*='oracle', *dsn*='localhost/xe')**

***def* getHospitales(*self*):**

**sql = "select \* from HOSPITAL"**

**cursor = *self*.*connection*.cursor()**

**cursor.execute(sql)**

**lista = []**

**for row in cursor:**

**hosp = Hospital()**

**hosp.*idHospital* = row[0]**

**hosp.*nombre* = row[1]**

**hosp.*direccion* = row[2]**

**hosp.*telefono* = row[3]**

**hosp.*camas* = row[4]**

**lista.append(hosp)**

**cursor.close()**

**return lista**

Posteriormente, creamos una página llamada **hospitales.html** dentro de **templates/pages**

Creamos un método llamado **hospitalesBBDD()** dentro de **views.py** y devolvemos la página

Creamos una nueva ruta dentro de **urls.py** y ponemos la página en nuestro **menú.html**

Por último, dentro de **views.py** recuperamos los datos de hospitales del Servicio y los dibujamos en la página **hospitales.html**

**VIEWS.PY**



***def* hospitalesBBDD(*request*):**

**servicio = ServiceHospital()**

**hospitales = servicio.getHospitales()**

**context = {**

**"hospitales": hospitales**

**}**

**return render(request, 'pages/hospitales.html', context)**

Dibujamos una tabla en la página HTML

**HOSPITALES.HTML**

**<!DOCTYPE html>**

**<html lang*="en"*>**

**<head>**

**<meta charset*="UTF-8"*>**

**<meta name*="viewport"* content*="width=device-width, initial-scale=1.0"*>**

**<title>Document</title>**

**</head>**

**<body>**

**{% include 'pages/menu.html' %}**

**<h1>Hospitales</h1>**

**<table border*="1"*>**

**<tr>**

**<th>Id</th>**

**<th>Nombre</th>**

**<th>Dirección</th>**

**<th>Teléfono</th>**

**<th>Camas</th>**

**</tr>**

**{% for h in hospitales %}**

**<tr>**

**<td>{{h.idHospital}}</td>**

**<td>{{h.nombre}}</td>**

**<td>{{h.direccion}}</td>**

**<td>{{h.telefono}}</td>**

**<td>{{h.camas}}</td>**

**</tr>**

**{% endfor %}**

**</table>**

**</body>**

**</html>**