**CONFIGURACION BOOTSTRAP DJANGO**

Lo que vamos a realizar es la utilización de Bootstrap dentro de Django.

Interfaz de usuario gráfica, Aplicación, Sitio web

El contenido generado por IA puede ser incorrecto.

Comenzamos creando un proyecto nuevo llamado **proyectochulo**

Montamos el proyecto con un elemento llamado **televisión** e iniciamos en una página llamada **index.html**

Debemos ir a la dirección web de Bootstrap

<https://getbootstrap.com>

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Copiamos el LINK dentro de **<head**> de nuestra página **index.html**

Gráfico

El contenido generado por IA puede ser incorrecto.

Copiamos el SCRIPT antes del cierre del **</body>**

Texto

El contenido generado por IA puede ser incorrecto.

Y ya tenemos instalado Bootstrap para jugar.

En cualquier página, debemos copiar el Link y el Script para mantener la integridad

**INDEX.HTML**

**{% load static %}**

**<!DOCTYPE html>**

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

**<head>**

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

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

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

**<link href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"* rel*="stylesheet"* integrity*="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH"* crossorigin*="anonymous"*>**

**</head>**

**<body>**

**<h1>Proyecto chulo televisión</h1>**

**<img class*="rounded-circle"* src*="{% static 'images/serieslogo.png' %}"*/>**

**<script src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"* integrity*="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"* crossorigin*="anonymous"*></script>**

**</body>**

**</html>**

A continuación, agregamos una página **menu.html**

Buscamos en Google navbar Bootstrap 5

**<nav class*="navbar navbar-expand-lg bg-body-tertiary"*>**

**<div class*="container-fluid"*>**

**<a class*="navbar-brand"* href*="#"*>Navbar</a>**

**<button class*="navbar-toggler"* type*="button"* data-bs-toggle*="collapse"* data-bs-target*="#navbarSupportedContent"* aria-controls*="navbarSupportedContent"* aria-expanded*="false"* aria-label*="Toggle navigation"*>**

**<span class*="navbar-toggler-icon"*></span>**

**</button>**

**<div class*="collapse navbar-collapse"* id*="navbarSupportedContent"*>**

**<ul class*="navbar-nav me-auto mb-2 mb-lg-0"*>**

**<li class*="nav-item"*>**

**<a class*="nav-link active"* aria-current*="page"* href*="/"*>Home</a>**

**</li>**

**<li class*="nav-item"*>**

**<a class*="nav-link"* href*="#"*>Link</a>**

**</li>**

**<li class*="nav-item dropdown"*>**

**<a class*="nav-link dropdown-toggle"* href*="#"* role*="button"* data-bs-toggle*="dropdown"* aria-expanded*="false"*>**

**Dropdown**

**</a>**

**<ul class*="dropdown-menu"*>**

**<li><a class*="dropdown-item"* href*="#"*>Action</a></li>**

**<li><a class*="dropdown-item"* href*="#"*>Another action</a></li>**

**<li><hr class*="dropdown-divider"*></li>**

**<li><a class*="dropdown-item"* href*="#"*>Something else here</a></li>**

**</ul>**

**</li>**

**<li class*="nav-item"*>**

**<a class*="nav-link disabled"* aria-disabled*="true"*>Disabled</a>**

**</li>**

**</ul>**

**<form class*="d-flex"* role*="search"*>**

**<input class*="form-control me-2"* type*="search"* placeholder*="Search"* aria-label*="Search"*>**

**<button class*="btn btn-outline-success"* type*="submit"*>Search</button>**

**</form>**

**</div>**

**</div>**

**</nav>**

Vamos a representar las series en nuestro nuevo proyecto.

Sobre **models.py** creamos un nuevo modelo para Serie y un servicio llamado **ServiceSeries**

**MODELS.PY**

**from django.db import models**

**import oracledb**

**# Create your models here.**

***class* Serie:**

**idSerie = 0**

**titulo = ""**

**imagen = ""**

**year = 0**

***class* ServiceSeries:**

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

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

**, *password*='oracle'**

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

***def* getSeries(*self*):**

**sql = "select \* from SERIES"**

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

**cursor.execute(sql)**

**lista = []**

**for row in cursor:**

**serie = Serie()**

**serie.*idSerie* = row[0]**

**serie.*titulo* = row[1]**

**serie.*imagen* = row[2]**

**serie.*year* = row[3]**

**lista.append(serie)**

**cursor.close()**

**return lista**

Dentro de **views.py** creamos un nuevo método para devolver todas las series.

**VIEWS.PY**

**from django.shortcuts import render**

**from television.models import ServiceSeries**

**# Create your views here.**

***def* index(*request*):**

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

***def* metodoSeries(*request*):**

**servicio = ServiceSeries()**

**series = servicio.getSeries()**

**context = {**

**"series": series**

**}**

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

Sobre **urls.py** creamos un nuevo **path** para las series

**URLS.PY**

**from django.urls import path**

**from television import views**

**urlpatterns = [**

**path('', views.*index*, *name*='index'),**

**path('series/', views.*metodoSeries*, *name*='series'),**

**]**

Sobre **templates/pages** creamos una nueva página llamada **series.html**

**SERIES.HTML**

**{% load static %}**

**<!DOCTYPE html>**

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

**<head>**

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

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

**<title>Series</title>**

**<link href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"* rel*="stylesheet"* integrity*="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH"* crossorigin*="anonymous"*>**

**</head>**

**<body>**

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

**<h1>Series</h1>**

**<div style*="width: 50%;margin: 0* auto*;"*>**

**{% for s in series %}**

**<div class*="card"* style*="width: 18rem;"*>**

**<img src*="{{*s*.*imagen*}}"* class*="card-img-top"* alt*="..."*>**

**<div class*="card-body"*>**

**<h5 class*="card-title"*>{{s.titulo}}</h5>**

**<p class*="card-text"*>Año: {{s.year}}</p>**

**<a href*="#"* class*="btn btn-primary"*>Go somewhere</a>**

**</div>**

**</div>**

**{% endfor %}**

**</div>**

**<script src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"* integrity*="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"* crossorigin*="anonymous"*></script>**

**</body>**

**</html>**

Por último, en nuestro menú, incluimos un Link para llegar a series.html

Imagen que contiene Escala de tiempo

El contenido generado por IA puede ser incorrecto.

A continuación vamos a mostrar los personajes:

**MODELS.PY**

***class* Personaje:**

**idPersonaje = 0**

**nombre = ""**

**imagen = ""**

**idSerie = 0**

***def* getPersonajes(*self*):**

**sql = "select \* from PERSONAJES"**

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

**cursor.execute(sql)**

**lista = []**

**for row in cursor:**

**person = Personaje()**

**person.*idPersonaje* = row[0]**

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

**person.*imagen* = row[2]**

**person.*idSerie* = row[3]**

**lista.append(person)**

**cursor.close()**

**return lista**

El siguiente paso es crear un nuevo método dentro de views.py

**VIEWS.PY**

***def* metodoPersonajes(*request*):**

**servicio = ServiceSeries()**

**personajes = servicio.getPersonajes()**

**context = {**

**"personajes": personajes**

**}**

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

Sobre urls.py creamos un nuevo path

**URLS.PY**

Diagrama

El contenido generado por IA puede ser incorrecto.

El siguiente ejemplo mostrará los personajes por una serie.

Debemos crear un método nuevo dentro de Models para buscar los personajes por un ID de Serie.

**MODELS.PY**

***def* getPersonajesSerie(*self*, *idserie*):**

**sql = "select \* from PERSONAJES where IDSERIE=:p1"**

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

**cursor.execute(sql, (idserie, ))**

**lista = []**

**for row in cursor:**

**person = Personaje()**

**person.*idPersonaje* = row[0]**

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

**person.*imagen* = row[2]**

**person.*idSerie* = row[3]**

**lista.append(person)**

**cursor.close()**

**return lista**

Creamos un nuevo método dentro de views.py que recibirá un parámetro que llamaremos **idserie** y que será recuperado mediante GET porque lo vamos a enviar con un Link (<a href>)

**VIEWS.PY**

***def* personajesSeries(*request*):**

**#PREGUNTAMOS SI HEMOS RECIBIDO EL DATO DE idserie EN GET**

**if ('idserie' in request.GET):**

**servicio = ServiceSeries()**

**idserie = request.GET['idserie']**

**personajes = servicio.getPersonajesSerie(idserie)**

**context = {**

**"personajes": personajes**

**}**

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

**else:**

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

Sobre urls.py creamos un nuevo path

**URLS.PY**

Imagen que contiene Diagrama

El contenido generado por IA puede ser incorrecto.

**PERSONAJESSERIE.HTML**

**{% load static %}**

**<!DOCTYPE html>**

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

**<head>**

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

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

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

**<link href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"* rel*="stylesheet"* integrity*="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH"* crossorigin*="anonymous"*>**

**</head>**

**<body>**

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

**<h1 style*="color:*red*"*>Personajes Serie</h1>**

**<div>**

**{% for p in personajes %}**

**<div class*="card"* style*="width: 18rem;"*>**

**<img src*="{{*p*.*imagen*}}"* class*="card-img-top"* alt*="..."*>**

**<div class*="card-body"*>**

**<h5 class*="card-title"*>{{p.nombre}}</h5>**

**<p class*="card-text"*>Serie: {{p.idSerie}}</p>**

**<a href*="#"* class*="btn btn-warning"*>Go somewhere</a>**

**</div>**

**</div>**

**{% endfor %}**

**</div>**

**<script src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"* integrity*="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"* crossorigin*="anonymous"*></script>**

**</body>**

**</html>**

Por último, necesitamos enviar el parámetro IDSERIE hacia nuestro método en **views.py**

Para ello, en el Link de series, vamos a enviar el parámetro del idserie.

**SERIES.HTML**

Texto

El contenido generado por IA puede ser incorrecto.

El siguiente paso es crear una funcionalidad para modificar un Personaje.

Al seleccionar un personaje, podremos ir a modificar sus propiedades.

Tendremos la posibilidad de cambiar su nombre, su imagen y su serie.

La serie la seleccionaremos desde un desplegable.

Creamos un nuevo método para update Personaje dentro de Model

**MODELS.PY**

***def* findPersonaje(*self*, *idPersonaje*):**

**sql = "select \* from PERSONAJES where IDPERSONAJE=:p1"**

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

**cursor.execute(sql, (idPersonaje, ))**

**row = cursor.fetchone()**

**person = Personaje()**

**person.*idPersonaje* = row[0]**

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

**person.*imagen* = row[2]**

**person.*idSerie* = row[3]**

**cursor.close()**

**return person**

***def* updatePersonaje(*self*, *idPersonaje*, *nombre*, *imagen*, *idSerie*):**

**sql = """**

**update PERSONAJES set PERSONAJE=:p1**

**, IMAGEN=:p2, IDSERIE=:p3**

**where IDPERSONAJE=:p4**

**"""**

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

**cursor.execute(sql, (nombre, imagen, idSerie, idPersonaje))**

***self*.*connection*.commit()**

**cursor.close()**

Sobre views.py, creamos el método para el Update

**VIEWS.PY**

***def* modificarPersonaje(*request*):**

**servicio = ServiceSeries()**

**if ('cajaimagen' in request.POST):**

**idpersonaje = request.POST['cajaidpersonaje']**

**nombre = request.POST['cajanombre']**

**imagen = request.POST['cajaimagen']**

**idserie = request.POST['cajaserie']**

**servicio.updatePersonaje(idpersonaje, nombre, imagen, idserie)**

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

**elif ('idpersonaje' in request.GET):**

**idpersonaje = request.GET['idpersonaje']**

**personaje = servicio.findPersonaje(idpersonaje)**

**context = {**

**"personaje": personaje**

**}**

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

**else:**

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

Agregamos la ruta sobre **URLS.PY**

Imagen que contiene Diagrama

El contenido generado por IA puede ser incorrecto.

Dentro de las páginas de personajes, modificamos el código para poder entrar a Modificar personaje

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.

En la versión 2, en lugar de mostrar un Id para la serie, mostramos un option para seleccionar la serie

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

**VIEWS.PY**

***def* modificarPersonaje(*request*):**

**servicio = ServiceSeries()**

**if ('cajaimagen' in request.POST):**

**idpersonaje = request.POST['cajaidpersonaje']**

**nombre = request.POST['cajanombre']**

**imagen = request.POST['cajaimagen']**

**idserie = request.POST['cajaserie']**

**servicio.updatePersonaje(idpersonaje, nombre, imagen, idserie)**

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

**elif ('idpersonaje' in request.GET):**

**idpersonaje = request.GET['idpersonaje']**

**personaje = servicio.findPersonaje(idpersonaje)**

**series = servicio.getSeries()**

**context = {**

**"personaje": personaje,**

**"series": series**

**}**

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

**else:**

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

Modificamos la página para dibujar las Series en un control <select>

**MODIFICARPERSONAJE.HTML**

**{% load static %}**

**<!DOCTYPE html>**

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

**<head>**

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

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

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

**<link href*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"* rel*="stylesheet"* integrity*="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH"* crossorigin*="anonymous"*>**

**</head>**

**<body>**

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

**<h1>Update personaje</h1>**

**<div>**

**<form method*="post"*>**

**{% csrf\_token %}**

**<input type*="hidden"* name*="cajaidpersonaje"***

**value*="{{*personaje*.*idPersonaje*}}"*/>**

**<label>Nombre: </label>**

**<input type*="text"* name*="cajanombre"***

**value*="{{*personaje*.*nombre*}}"***

**class*="form-control"*/>**

**<label>Imagen: </label>**

**<input type*="text"* name*="cajaimagen"***

**value*="{{*personaje*.*imagen*}}"***

**class*="form-control"*/>**

**<label>Serie</label>**

**<select name*="cajaserie"* class*="form-control"*>**

**{% for s in series %}**

**<option value*="{{*s*.*idSerie*}}"*>{{s.titulo}}</option>**

**{% endfor %}**

**</select>**

**<button class*="btn btn-info"*>Update</button>**

**</form>**

**</div>**

**<script src*="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"* integrity*="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"* crossorigin*="anonymous"*></script>**

**</body>**

**</html>**

Necesito una nueva página para modificar un Personaje a una nueva serie.

No lo haremos pulsando sobre el Personaje, la página es independiente

En la nueva página llamada **updatepersonserie.html** tendremos un <select> para todos los personajes y un <select> para todas las series.

Seleccionaremos un Personaje, seleccionaremos una Serie y haremos el cambio en base de datos

Interfaz de usuario gráfica, Aplicación, Teams

El contenido generado por IA puede ser incorrecto.