

Informe de Laboratorio 06

Tema: Python - Django

Nota

Estudiante	Escuela	Asignatura
Jose Alonzo Gordillo Mendoza jgordillome@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación Web 2 Semestre: III Código: 20220577

Laboratorio	Tema	Duración
06	Python - Django	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - A	Del 14 Junio 2023	Al 28 Junio 2023

1. Tarea

- Crearemos una pagina de viajes, usando un video como base: <https://www.youtube.com/watch?v=OTmQQjsl0eg>



2. URL de Repositorio Github

- URL para el laboratorio 06 en el Repositorio GitHub.
- <https://github.com/JoseGordilloMendoza/LAB06-PWEB2/tree/main>

3. Ejercicios

3.1. Estructura de laboratorio 05

- La distribucion de archivos sera la siguiente (teniendo en cuenta solo el entorno virtual y los archivos mas importantes, por ejemplo lo descargado en el entorno es demasiado extense como para incluirlo):

```
lab06-PW2/
|----lab/
|----Lib
|----mysite
|----destinos
|----migrations
|----__pycache__
|----__init__.py
|----admin.py
|----apps.py
|----models.py
|----tests.py
|----urls.py
|----views.py
|----cuenta
|----migrations
|----__pycache__
|----__init__.py
|----admin.py
|----apps.py
|----models.py
|----tests.py
|----urls.py
|----views.py
|----templates
|----destinos_turisticos.html
|----detalle_destino.html
|----login.html
|----register.html
|----static
|----style.css
|----media
|----imagenes
|----mysite
|----__pycache__
|----__init__.py
|----settings.py
|----urls.py
|----wsgi.py
|----db.sqlite3
|----manage.py
|----Scripts
|----pyenv.cfg
```

3.2. Análisis de archivos

En primer lugar veamos lo mas resaltante de mysite/settings.py:

Listing 1: Analizando mysite/settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
```

```
'django.contrib.contenttypes',  
'django.contrib.sessions',  
'django.contrib.messages',  
'django.contrib.staticfiles',  
'destinos.apps.DestinosConfig',  
]
```

Estas son las configuraciones de las aplicaciones instaladas, la ultima aplicacion "destinos.es" la que creamos para nuestra pagina de viajes.

Listing 2: urls.py del proyecto

```
from django.contrib import admin  
from django.urls import path, include  
from django.conf import settings  
from django.conf.urls.static import static  
  
urlpatterns = [  
    path('', include('destinos.urls')),  
    path('admin/', admin.site.urls),  
    path('cuenta/', include('cuenta.urls'))  
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Veamos que nuestras direcciones URL ahora cuentan con las url de la aplicacion destinos y cuenta que veremos despues

- Aplicacion /destinos

Listing 3: models.py

```
class DestinosTuristicos(models.Model):  
    nombreCiudad = models.CharField(max_length=100)  
    descripcionCiudad = models.TextField()  
    imagenCiudad = models.ImageField(upload_to='imagenes/')  
    precioTour = models.DecimalField(max_digits=8, decimal_places=2)  
    ofertaTour = models.BooleanField(default=False)  
  
    def __str__(self):  
        return self.nombreCiudad
```

Este es nuestro modelo DestinosTuristicos que sera nuestra clase para los diferentes destinos turisticos que necesitamos, como vemos este modelo tiene como atributos el nombre de la ciudad, una descripcion, su imagen, el precio y la oferta, cada uno con su respectivo tipo de dato, por ejemplo en la oferta sera de tipo booleano (verdadero y falso) dependiendo a eso se mostrara un aviso.

Listing 4: views.py

```
def destinos_turisticos(request):  
    destinos = DestinosTuristicos.objects.all()  
    return render(request, 'destinos_turisticos.html', {'destinos': destinos})  
  
def detalle_destino(request, destino_id):  
    destino = get_object_or_404(DestinosTuristicos, pk=destino_id)  
    return render(request, 'detalle_destino.html', {'destino': destino})
```

La funcion destinos_turisticos Obtiene todos los objetos de la clase DestinosTuristicos usand DestinosTuristicos.objects.all(). Luego, utiliza la función render para renderizar (generar) una respuesta

HTML utilizando una plantilla llamada 'destinos_turisticos.html'. También pasa un diccionario con la variable 'destinos' que contiene la lista de todos los destinos turísticos obtenidos en el paso anterior. La segunda función, detalle_destino utiliza la función `get_object_or_404` para obtener un objeto de la clase `DestinosTuristicos` que coincida con el id proporcionado. Si no se encuentra ningún objeto con ese identificador, se mostrará una página de error 404. Luego, se genera una respuesta HTML utilizando la plantilla 'detalle_destino.html'. También pasa un diccionario con la variable 'destino' que contiene el destino turístico específico obtenido en el paso anterior.

Listing 5: urls.py

```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.destinos_turisticos, name='destinos_turisticos'),
    path('destino/<int:destino_id>/', views.detalle_destino,
         name='detalle_destino'),
]
```

Estas son las rutas que usaremos en nuestra página, la primera siendo la ruta raíz o la página principal, la segunda ruta se utiliza para mostrar detalles de un destino específico, que usa el id del destino para acceder al mismo.

- Aplicación /cuenta

Listing 6: Primera función de views.py

```
def login(request):
    if request.method == 'POST':
        username = request.POST['username']
        password = request.POST['password']

        user = auth.authenticate(username=username, password=password)

        if user is not None:
            auth.login(request, user)
            return redirect("/")
        else:
            messages.info(request, 'invalid credentials')
            return redirect('login')

    else:
        return render(request, 'login.html')
```

Esta función maneja el proceso de inicio de sesión verificando las credenciales ingresadas por el usuario y autenticando al usuario si las credenciales son válidas. Luego, redirige al usuario a la página principal si la autenticación es exitosa, o muestra un mensaje de error y redirige al usuario nuevamente a la página de inicio de sesión si la autenticación falla. Si el usuario accede a la página de inicio de sesión por primera vez, se muestra la página de inicio de sesión para que el usuario pueda ingresar sus credenciales.

Listing 7: Segunda función de views.py

```
def register(request):
    if request.method == 'POST':
        first_name = request.POST['first_name']
        last_name = request.POST['last_name']
```

```
username = request.POST['username']
password1 = request.POST['password1']
password2 = request.POST['password2']
email = request.POST['email']
if password1==password2:
    if User.objects.filter(username=username).exists():
        messages.info(request,'Username Taken')
        return redirect('register')
    elif User.objects.filter(email=email).exists():
        messages.info(request,'Email Taken')
        return redirect('register')
    else:
        user = User.objects.create_user(username=username, password=password1,
            email=email,first_name=first_name,last_name=last_name)
        user.save();
        print('user created')
        return redirect('login')
else:
    messages.info(request,'password not matching..')
    return redirect('register')
return redirect('/')
else:
    return render(request,'register.html')
```

Esta segunda función maneja el proceso de registro de nuevos usuarios verificando la coincidencia de contraseñas, la disponibilidad de nombre de usuario y correo electrónico, y crea un nuevo usuario si todos los datos son válidos. Luego, redirige al usuario a la página de inicio de sesión si el registro es exitoso, o muestra mensajes de error y redirige al usuario nuevamente a la página de registro si se encuentran problemas con los datos proporcionados.

Listing 8: Tercera función de views.py

```
def logout(request):
    auth.logout(request)
    return redirect('/')
```

Esta función cierra la sesión del usuario y redirige al usuario a la página principal después de cerrar sesión en la aplicación.

Listing 9: urls.py

```
urlpatterns = [
    path("register", views.register, name="register"),
    path("login",views.login, name="login"),
    path("logout",views.logout,name="logout"),
]
```

Estos patrones de URL definen las rutas en las que se espera que los usuarios accedan a la aplicación para realizar acciones como registro, inicio de sesión y cierre de sesión. Cada patrón de URL está asociado a una función en el archivo views.py que se encarga de procesar la solicitud y devolver una respuesta adecuada. El nombre asignado a cada patrón de URL se utiliza para referirse a esa URL específica en el código, como en las redirecciones o generación de enlaces.

- TEMPLATES HTML (haremos uso de variables con llaves, include, if, for, entre otros)

_turisticos.html

```
{% load static %}
{% static "images" as baseUrl %}

<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Lugares Tursticos</title>
    <link rel="stylesheet" href="{% static 'style.css' %}">
</head>
<body>
    <header>
        <div class="logo">
            
        </div>
        <nav>
            <ul>
                <li><a href="http://127.0.0.1:8000/" class="sub">Pagina
                    principal</a></li>

                {% if user.is_authenticated %}
                <li class="sub welcome-message">Bienvenido,
                    <span>{{user.first_name}}</span></li>
                <li><a href="cuenta/logout" class="sub">Salir</a></li>
                {% else %}
                <li><a href="http://127.0.0.1:8000/cuenta/register"
                    class="sub">REGISTER</a></li>
                <li><a href="http://127.0.0.1:8000/cuenta/login"
                    class="sub">LOGIN</a></li>
                {% endif %}

                <li><a href="#" class="carrito" data-counter="0">&#128722;</a></li>
            </ul>
        </nav>
    </header>
```

Este fragmento de código HTML crea la estructura básica de una página web y muestra un encabezado con un logotipo, elementos de navegación y un icono de carrito de compras. La visibilidad de algunos elementos de navegación depende del estado de autenticación del usuario. También se cargan archivos estáticos, como una hoja de estilo CSS, utilizando la etiqueta static..

Listing 10: Segunda parte de destinos_turisticos.html

```
{% block content %}
<div class="turismo-peru1">
    <h2>TURISMO EN EL PERU</h2>
</div>

<div class="turismo-peru">
    <div class="content">
        <div class="image">
            
```

```

    </div>
    <div class="text">
        <p>El Per es uno de los pases ms variados del mundo. Un pas
            multicultural, lleno de tradiciones, una laureada </p>
        <p>gastronoma y vastas reservas naturales. Posee 12 patrimonios mundiales
            reconocidos por Unesco y es dueo de 84 de las 117 zonas de vida que
            existen en el mundo.</p>
        <p>En su vasto territorio, de ms de 1.2 millones de km, abarca tres
            regiones: Costa, Sierra y Selva. Su poblacin actual supera los 31.5
            millones de habitantes.</p>
        <p>El espaol es el idioma oficial del Per. Sin embargo, en el pas se
            hablan 47 lenguas nativas, incluyendo el quechua y el aymara. </p>
        <p>Gracias al legado de civilizaciones milenarias, Per alberga ms de 5000
            lugares arqueolgicos. Muchos de ellos se encuentran an rodeados de
            misterio, pero igual son capaces de transportar al visitante a la poca
            donde florecieron estas culturas.</p>
    </div>
</div>

<div class="turismo-peru1">
    <h2>LUGARES TURISTICOS</h2>
</div>

<main>
    <div class="destinos-grid">
        {% for destino in destinos %}
            <div class="destino">
                <h2>{{ destino.nombreCiudad }}</h2>
                
                <p>Precio: S/ {{ destino.precioTour }}</p>
                {% if destino.ofertaTour %}
                    <p class="oferta">Oferta disponible</p>
                {% endif %}
                <a href="{{ url 'detalle_destino' destino.id }}"
                    class="ver-btn">VER</a>
            </div>
        {% endfor %}
    </div>
</main>
{% endblock %}

<footer>
    <p> 2023 Lugares Tursticos Peruanos. Todos los derechos reservados. Hecho
        por Jose Gordillo Mendoza</p>
</footer>
</body>
</html>

```

Este utiliza la sintaxis de Django para generar contenido dinámico. El bloque `block content` se utiliza en el contexto de Django como una etiqueta de inicio para definir una sección del contenido que se puede sobrescribir en plantillas secundarias. el bloque `block content` indica una sección en la plantilla que puede ser reemplazada en las plantillas secundarias. Es parte del mecanismo de herencia de plantillas en Django para generar contenido dinámico y personalizable.

Listing 11: detalle.destino.html

```
{% extends 'destinos_turisticos.html' %}
{% load static %}

{% block content %}

<div class="centered-container">
  <div class="detalle-destino">
    <div class="imagen">
      
    </div>
    <div class="descripcion-precio">
      <div class="descripcion">
        <h2>{{ destino.nombreCiudad }}</h2>
        <p>{{ destino.descripcionCiudad }}</p>
      </div>
      <div class="precio-oferta">
        <div class="precio">
          <p>Precio: S/ {{ destino.precioTour }}</p>
        </div>
        <div>
          {% if destino.ofertaTour %}
            <p class="oferta">Oferta disponible</p>
          {% endif %}
        </div>
        <div class="comprar-btn">
          <a href="http://127.0.0.1:8000/cuenta/login"
            class="comprar-btn">COMPRAR</a>
        </div>
      </div>
    </div>
  </div>
{% endblock %}
```

Con respecto a este fragmento de código se define la estructura y contenido de la página de detalle de un destino turístico, utilizando variables y lógica de plantillas de Django para mostrar información dinámica.

Listing 12: login.html

```
{% load static %}
{% static "images" as baseUrl %}

<!DOCTYPE html>
<html>
<head>
  <meta charset='utf-8'>
  <meta http-equiv='X-UA-Compatible' content='IE=edge'>
  <title>Ingresar...</title>
  <meta name='viewport' content='width=device-width, initial-scale=1'>
  <link rel="stylesheet" href="{% static 'style.css' %}">
</head>

<header>
```



```
<div class="logo">

</div>
<nav>
  <ul >
    <li><a href="http://127.0.0.1:8000/" class="sub">Pagina
      Principal</a></li>
    <li><a href="http://127.0.0.1:8000/cuenta/register"
      class="sub">Registrarse</a></li>
    <li><a href="#" class="carrito" data-counter="0">&#128722;</a></li>
  </ul>
</nav>
</header>

<body>

  <h1 class="titLoginRegister" align="center">INGRESAR</h1>

  <form action="login" method="post">
    {% csrf_token %}
    <input type="text" name="username" placeholder="Usuario"><br>
    <input type="password" name="password" placeholder="Contrasea"><br>
    <input type="Submit" value="Sign in">
  </form>

  <div>
    {% for message in messages %}
    <h3> {{message}} </h3>
    {% endfor %}
  </div>
</body>
</html>
```

Este código es una plantilla HTML con código de Django que genera una página de inicio de sesión básica, carga archivos estáticos como CSS y muestra mensajes de Django en caso de que haya errores o información adicional después del inicio de sesión.

Listing 13: register.html

```
{% load static %}
{% static "images" as baseUrl %}

<!DOCTYPE html>
<html>
<head>
  <meta charset='utf-8'>
  <meta http-equiv='X-UA-Compatible' content='IE=edge'>
  <title>Registrarme..</title>
  <meta name='viewport' content='width=device-width, initial-scale=1'>
  <link rel="stylesheet" href="{% static 'style.css' %}">
</head>
<header>
  <div class="logo" >

  </div>
  <nav>
    <ul >
```

```

        <li><a href="http://127.0.0.1:8000/" class="sub">Pagina
            principal</a></li>
        <li><a href="http://127.0.0.1:8000/cuenta/login"
            class="sub">Login</a></li>
        <li><a href="#" class="carrito" data-counter="0">&#128722;</a></li>
    </ul>
</nav>
</header>

<body>

    <h1 class="titLoginRegister" align="center">REGISTRAR</h1>

    <form action="register" method="post">
        {% csrf_token %}

        <input type="text" name="first_name" placeholder="Ingresa tus nombres"><br>
        <input type="text" name="last_name" placeholder="Ingresa Apellidos"><br>
        <input type="text" name="username" placeholder="Tu usuario: "><br>
        <input type="email" name="email" placeholder="Correo: "><br>
        <input type="password" name="password1" placeholder="Contrasea"><br>
        <input type="password" name="password2" placeholder="Vuelve a escribir la
            contrasea"><br>
        <input type="Submit" value="Registrarme">

    </form>

    <div>
        {% for message in messages %}
        <h3> {{message}} </h3>
        {% endfor %}
    </div>

</body>
</html>

```

Este código es una plantilla HTML con código de Django que genera una página de registro básica, carga archivos estáticos como CSS y muestra mensajes de Django en caso de que haya errores o información adicional después del registro.

- Accediendo a los estilos /static

Listing 14: styles.css

```

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    font-family: Arial, Helvetica, sans-serif;
}

body {
    font-family: Impact;
    background-color: #f2f2f2;
    color: #333;
}

```

```
}

.container {
  max-width: 1200px;
  margin: 0 auto;
  padding: 20px;
}

header {
  background-color: #260303;
  padding: 1px;
  display: flex;
  align-items: center;
  justify-content: space-between;
}

.logo img {
  max-width: 110px;
  height: auto;
  margin-right: 30px;
  margin-left: 30px;
}

nav ul {
  display: flex;
  list-style-type: none;
  margin: 20px;
  padding: 10px;
}

nav ul li {
  margin-left: 20px;
}

.sub {
  font-weight: bold;
  font-family: Arial Black;
  display: inline-block;
  padding: 10px;
  background-color: #e9e9e9;
  border-radius: 5px;
  text-decoration: none;
  color: #000000;
  transition: 0.5s;
}

.sub:hover {
  background-color: #8f8f8f49;
  color: #f4e8c5;
  transform: scale(1.15);
}

.welcome-message span {
  font-weight: bold;
  text-transform: uppercase;
}
```

```
        color: #000000;
        transition: 0.3s;
    }.welcome-message span:hover {
        font-weight: bold;
        color: #ffffff;
    }

    .carrito {
        position: relative;
        display: inline-block;
        padding: 10px;
        background-color: #e9e9e9;
        border-radius: 50%;
        text-decoration: none;
        color: #333;
        transition: background-color 0.3s ease;
    }

    .carrito:hover {
        background-color: #ccc;
    }

    .carrito::after {
        content: attr(data-counter);
        position: absolute;
        top: -5px;
        right: -5px;
        display: flex;
        align-items: center;
        justify-content: center;
        width: 20px;
        height: 20px;
        background-color: #ff0000;
        color: #fff;
        font-size: 12px;
        font-weight: bold;
        border-radius: 50%;
    }

    .turismo-peru {
        background-color: #fff;
        padding: 20px;
        border-radius: 5px;
        box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
        text-align: center;
        display: flex;
        margin: 20px;
    }

    .turismo-peru1 {
        font-family: Impact;
        color: #ffffff;
        background-color: #160505;
        padding: 20px;
        border-radius: 5px;
```

```
        box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
        text-align: center;
    margin: 20px;
}

.turismo-peru3 {
    align-items: center;
    color: #fff;
    background-color: #ffffff;
    padding: 20px;
    border-radius: 5px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    margin: 20px;
}

.turismo-peru h2 {
    font-size: 24px;
    text-align: center;
    margin-bottom: 20px;
}

.turismo-peru .content {
    display: flex;
    align-items: center;
    justify-content: space-between;
    margin: 5px;
}

.turismo-peru .image img {
    max-width: 600px;
    height: auto;
}

.turismo-peru .text p {
    text-align: justify;
    font-size: 18px;
    margin: 1px 30px;
}

main {
    padding: 20px;
}

h1 {
    font-size: 32px;
    margin-bottom: 20px;
}

.destinos-grid {
    display: grid;
    grid-template-columns: repeat(3, 1fr);
    gap: 20px;
}
```

```
.destino {
  background-color: #fff;
  padding: 20px;
  border-radius: 5px;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
  text-align: center;
  transition: 0.6s;
}
.destino:hover{
  transform: scale(1.2);
}

.destino h2 {
  font-family: Arial Black;
  font-size: 24px;
  margin-bottom: 10px;
}

.destino img {
  width: 100%;
  max-height: 200px;
  object-fit: cover;
  margin-bottom: 10px;
}

.destino p {
  margin-bottom: 5px;
}

.oferta {
  font-family: 'Arial Narrow Bold', sans-serif;
  font-weight: bold;
  color: rgba(92, 26, 26, 0.433);
}

.ver-btn {
  font-family: 'Arial Black';
  display: inline-block;
  padding: 10px 20px;
  background-color: #333;
  color: #ebe9dfd1;
  text-decoration: none;
  border-radius: 5px;
  transition: 0.5s ease-in-out;
}
.ver-btn:hover {
  transform: scale(1.15);
  background-color: #00000058;
  color: #252525;
}

footer {
  background-color: #333;
  color: #fff;
  padding: 20px;
  text-align: center;
```

```
}

/* -----Estilos de loginnn y registerr----- */

form {
    background-color: #fff;
    width: 400px;
    margin: 0 auto;
    padding: 20px;
    border-radius: 5px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
}

input[type="text"],
input[type="email"],
input[type="password"] {
    align-content: center;
    width: 100%;
    padding: 10px;
    margin-bottom: 10px;
    border: 1px solid #ccc;
    border-radius: 5px;
}

input[type="submit"] {
    background-color: #333;
    color: #fff;
    padding: 10px 20px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    transition: 0.5s;
}

input[type="submit"]:hover {
    background-color: #333;
    transform: scale(1.2);
}

div.message {
    margin-top: 10px;
    text-align: center;
}

h3 {
    color: #333;
    font-size: 16px;
    margin: 5px 0;
}

nav ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    display: flex;
```

```
}

nav ul li {
    margin-right: 20px;
}

nav ul li a {
    text-decoration: none;
    color: #000000;
    font-weight: bold;
}

.navbar a.carrito {
    font-size: 24px;
}

.titLoginRegister {
    text-align: center;
    color: #333;
    font-size: 28px;
    margin-top: 30px;
    margin-bottom: 20px;
}

/* -----Estilos de detalles de lso destinos----- */
body {
    background-color: #f4f4f4;
}

.container {
    max-width: 960px;
    margin: 0 auto;
    padding: 20px;
}

.detalle-destino {
    background-color: #ffffff;
    border: 1px solid #dddddd;
    border-radius: 5px;
    padding: 20px;
    margin: 0 auto;
    max-width: 800px;
}

.imagen {
    flex: 1;
    margin-right: 20px;
    display: flex;
    justify-content: center;
    align-items: center;
}

.imagen img {
    max-width: 100%;
    max-height: 100%;
}
```



```
.descripcion-precio {
  flex: 2;
  display: flex;
  flex-direction: column;
}

.descripcion {
  margin-bottom: 20px;
}

.precio-oferta {
  font-family: Arial Black;
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 20px;
}

.precio {
  font-family: Arial Black;
  background-color: #ffffff;
  border: 1px solid #dddddd;
  border-radius: 5px;
  padding: 10px;
}

.oferta {
  font-family: Arial Black;
  padding: 10px;
  background-color: #651b1b;
  color: #ffffff;
  text-align: center;
  font-weight: bold;
  border-radius: 5px;
}

.comprar-btn {
  font-family: Arial Black;
  padding: 10px 20px;
  background-color: #111611;
  color: #ffffff;
  text-decoration: none;
  border-radius: 5px;
  transition: 0.4s;
}


.comprar-btn:hover{
  transform: scale(1.2);
}
```

Estos son los estilos que se usaron en los html, es importante mencionar a efectos como hover (cuando se pasa el puntero sobre un objeto), que dinamizan la pagina web

3.3. Demostracion de pagina


Ahoa veamos a la pagina como tal:

Primero tenemos que correr nuestro server con el comando "python manage.py run-server", una vez dentro veamos la pantalla principal:



Pagina principal
Bienvenido, JOSE
Salir

TURISMO EN EL PERU




El Perú es uno de los países más variados del mundo. Un país multicultural, lleno de tradiciones, una laureada gastronomía y vastas reservas naturales. Posee 12 patrimonios mundiales reconocidos por Unesco y es dueño de 84 de las 117 zonas de vida que existen en el mundo.

En su vasto territorio, de más de 1.2 millones de km², abarca tres regiones: Costa, Sierra y Selva. Su población actual supera los 31.5 millones de habitantes.

El español es el idioma oficial del Perú. Sin embargo, en el país se hablan 47 lenguas nativas, incluyendo el quechua y el aymara. Gracias al legado de civilizaciones milenarias, Perú alberga más de 5000 lugares arqueológicos. Muchos de ellos se encuentran aún rodeados de misterio, pero igual son capaces de transportar al visitante a la época donde florecieron estas culturas.


LUGARES TURISTICOS

Cuzco - MachuPicchu


Precio: S/ 100,00


Oferta disponible

VER

Puno - Lago Titicaca


Precio: S/ 100,00


VER

Arequipa - Salinas y Aguada Blanca


Precio: S/ 50,00

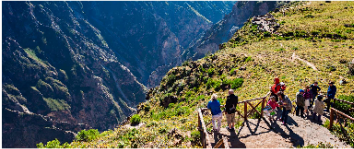
Oferta disponible

VER

Trujillo - Ruinas de Chan Chan


Precio: S/ 75,00


VER

Arequipa - Valle del Colca


Precio: S/ 150,00

Oferta disponible

VER

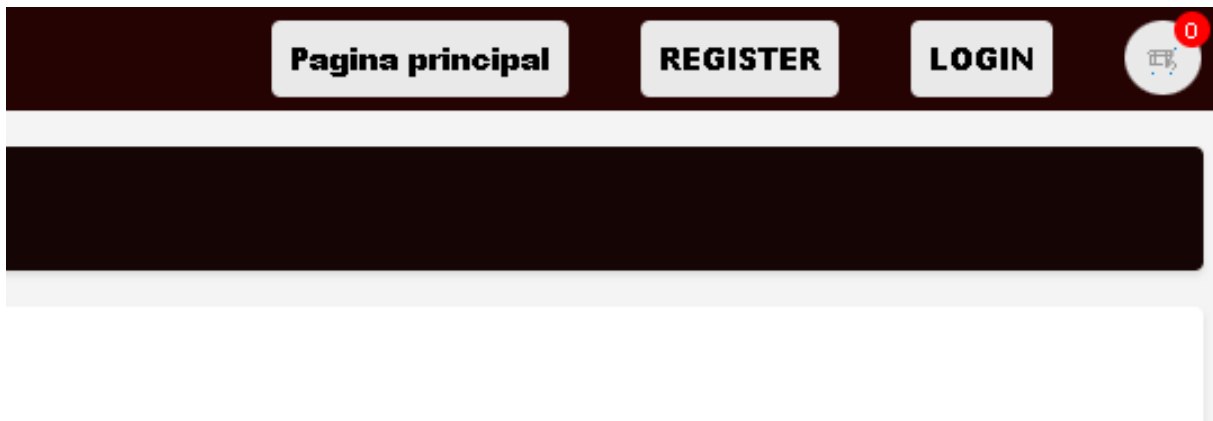
Piura - Mancora


Precio: S/ 250,00

Oferta disponible

VER

Esa fue la pantalla, pero que tenia a un usuario ya logeado, por lo que al hacer click en salir se nos dara la opcion de registrarnos o logearnos de nuevo:



Ahora veamos lo que nos aparece haciendo click en "VER" de un destino es específico:



Como vemos se nos muestra el destino con sus respectivos atributos, siguiendo el modelo creado

Ahora veamos el apartado de registro, en donde se nos pedira llenar un forms simple:

[Pagina principal](#)
[Login](#)

REGISTRAR

Por otra parte esta el apartado de logeo, donde accederemos a un usuario ya existente

[Pagina Principal](#)
[Registrarse](#)

INGRESAR

Por ultimo revisemos usuarios y los destinos guardados en la pagina de administracion:

Administración de Django
BIENVENIDOS, JOSE. [VER EL SITIO](#) / [CAMBIAR CONTRASEÑA](#) / [CERRAR SESIÓN](#)

Inicio > Autenticación y autorización > Usuarios

Empezar a escribir para filtrar...

AUTENTICACIÓN Y AUTORIZACIÓN
Grupos [+ Añadir](#)
Usuarios [+ Añadir](#)
DESTINOS
Destinos turísticos [+ Añadir](#)

Seleccione usuario a modificar

Acción: Ir seleccionados 0 de 3

<input type="checkbox"/>	NOMBRE DE USUARIO	DIRECCIÓN DE CORREO ELECTRÓNICO	NOMBRE	APELLIDOS	ES STAFF
<input type="checkbox"/>	Alonzo	jgordillome@unsa.edu.pe	Jose	Gordillo	✓
<input type="checkbox"/>	Jose	sjbalonzo201c@gmail.com	Jose	Gordillo	✓
<input type="checkbox"/>	asd	Cooliemx19@hotmail.com	PROBANDO	asdasd	✗

3 usuarios

FILTRO
☐ Por es staff
☐ Por estado de superusuario
☐ Por activo

Administración de Django BIENVENIDOS, JOSE. VER EL SITIO / CAMBIAR CONTRASEÑA / CERRAR SESIÓN

Inicio > Destinos > Destinos turísticos > Piura - Mancora

Empiece a escribir para filtrar...

AUTENTICACIÓN Y AUTORIZACIÓN

Grupos + Añadir

Usuarios + Añadir

DESTINOS

Destinos turísticos + Añadir

Modificar destinos turísticos HISTÓRICO

Piura - Mancora

NombreCiudad:

DescripcionCiudad:

Máncora se ubica en la región Piura, al noroeste de Perú. Es un sitio que debes visitar, si deseas disfrutar de escenarios naturales únicos y si al mismo tiempo, estás en la búsqueda de socializar y tomarte las vacaciones de tu vida.

Máncora está a sólo 15 minutos de la playa Los Órganos, un lindo balneario con playas de arena blanca y aguas claras, así como a 10 minutos de Vichaylto, una playa al sur de Máncora, mucho más tranquila y con menos concurrencia que en Máncora. Casas de Vacaciones Perú, cuenta dos casas aquí.

ImagenCiudad: Actualmente: [imagenes/Mancora.jpg](#)
Modificar: Ninguno archivo selec.

PrecioTour:

☒ OfertaTour

Link donde esta alojado el video (FlipGrid):

GRUPO: <https://flip.com/groups/14644257/topics/36969916/responses>

VIDEO ESPECIFICO: <https://flip.com/s/rYjXvXLSCPu>

4. Cuestionario

- Sin cuestionario -

5. Conclusiones

- El desarrollo de aplicaciones utilizando Django ofrece una combinación de productividad, escalabilidad, seguridad y flexibilidad. Su enfoque basado en convenciones, su sólida arquitectura y su amplia comunidad de desarrolladores hacen de Django una opción poderosa para crear aplicaciones web de calidad..
- El uso de tags en Django mejora la flexibilidad, la legibilidad y la modularidad de las plantillas HTML, permitiendo la generación de contenido dinámico y la separación clara entre la lógica del backend y la presentación visual. Esto contribuye a un desarrollo más eficiente y estructurado de aplicaciones web utilizando el framework Django.
- El uso de plantillas y archivos estáticos en Django ofrece flexibilidad, reutilización de código, separación de la lógica y la presentación, y un rendimiento eficiente en la entrega de recursos estáticos. Estas características hacen que el desarrollo y mantenimiento de aplicaciones web sea más organizado, escalable y fácil de mantener.
- Las vistas en Django son componentes clave para controlar la lógica del backend en una aplicación web. Permiten la separación de responsabilidades, interactúan con modelos y bases de datos, generan respuestas personalizadas y pueden ser probadas para garantizar un correcto funcionamiento. Su uso adecuado contribuye a un desarrollo eficiente, escalable y mantenible de aplicaciones web utilizando el framework Django.

6. Referencias

- https://www.w3schools.com/python/python_reference.asp
- <https://docs.python.org/3/tutorial/>
- <https://docs.djangoproject.com/en/4.0/>
- <https://www.youtube.com/watch?v=M4NIs4BM1dk>
- https://pip.pypa.io/en/latest/user_guide/
- <https://www.youtube.com/watch?v=OTmQ0jsl0eg>