

Tabla de contenido

Extensiones Recomendadas en VSCode	2
Framework VS Librería	3
Modelo Vista Controlador	4
1.- Vista General	5
¿Qué es Python y para qué sirve?	5
Django	5
Instalar Django	5
Crear un proyecto Django	6
Crear apps de Django	7
2.- Vistas, Plantillas (Layout y bloques) y Rutas	9
Vistas o Views	9
Plantillas o Templates	9
Rutas o URL`s	11
Estilos (CSS) en plantillas	13
Crear URL Glogables y particulares para cada Apps	22
Filtros en plantillas	23

Extensiones Recomendadas en VSCode

Descargar e instalar las siguientes extenciones en VSCode

- 1. LiveServer**
- 2. LiveShare**
- 3. Material Icon Theme**
- 4. Python**
- 5. Python Debugger**
- 6. Snippet**
- 7. Spanish Language Pack for Visual Studio Code**
- 8. TODO Highlight**
- 9. TODO tree**
- 10. Image Preview**
- 11. HTML CSS Support**
- 12. Better Comments**
- 13. Color Highlight**
- 14. CSS Peek**
- 15. ESLint**
- 16. GitHub Pull Requests**
- 17. GitLens — Git supercharged**
- 18. Lorem ipsum**
- 19. Paste JSON as Code**
- 20. Prettier - Code formatter**
- 21. Quokka.js**
- 22. SVG**
- 23. vscode-icons**
- 24. Django**

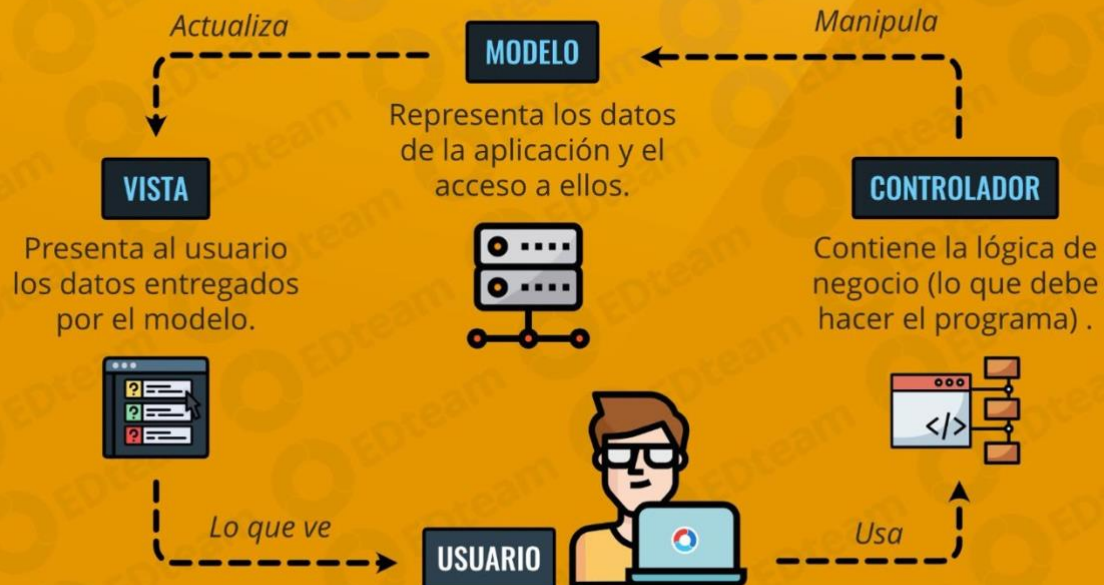
Framework VS Librería




Modelo Vista Controlador

¿QUÉ ES EL MODELO VISTA CONTROLADOR?

(MVC) es una arquitectura de software que nos permite tener una organización adecuada de nuestro código.



Aprende a programar en cualquier lenguaje (primer curso gratis) en:

 ed.team/programacion



1.- Vista General

¿Qué es Python y para qué sirve?

- Es uno de los lenguajes de programación de alto nivel más populares
- Es uno de los lenguajes más solicitados a nivel laboral
- Es interpretado no compilado, el script es interpretado por Python y devuelve un resultado.
- Es un multipropósito (Escritorio/Web/App)
- Es multiparadigma (Estructurada/Imperativa/Orientada a Objetos/Funcional)
- Es multiplataforma (Win/Linux/Mac)
- Curva de aprendizaje sencilla

Django

Es Framework (Marco de Desarrollo o forma de trabajar), se utiliza para facilitar mediante el lenguaje Python la creación de sitios web complejos, trabaja bajo la arquitectura de desarrollo de SW Modelo-Vista-Controlador (MVC)

Instalar Django

<https://www.djangoproject.com/download/>

Linux / macOS:

```
python3 -m pip install Django==5.1.1
```

Windows:

```
py -m pip install Django==5.1.1
```

Verificar la version después de la instalación del paquete o librería

```
python3 -m django --version
```

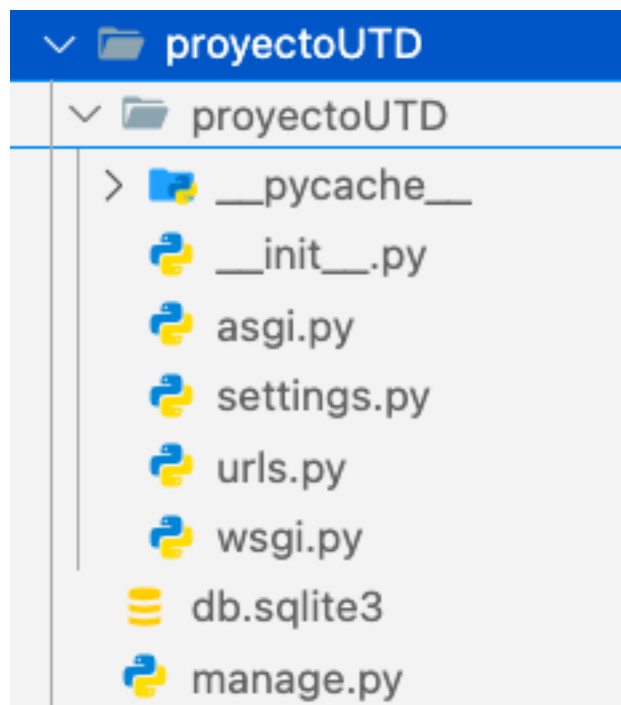
Crear un proyecto Django

django-admin startproject proyectoUTD	Crear un proyecto en Django dentro de un directorio
Cd .. proyectoUTD	Entrar al proyecto creado
python3 manage.py runserver	Arrancar el servidor Local Performing system checks... System check identified no issues (0 silenced). September 18, 2024 - 03:44:18 Django version 5.1.1, using settings 'Proyecto1Django.settings' Starting development server at http://127.0.0.1:8000/ Quit the server with CONTROL-C.
python3 manage.py migrate	Para aplicar los cambios en este caso migraciones
python3 manage.py createsuperuser	Para acceder al panel de administracion que trae Django http://127.0.0.1:8000/admin
pip3 install django-ckeditor	Para instalar un editor en el Panel de Administracion de Django para los TextArea que tenga un editor de texto enriquecido
python3 manage.py makemigrations	Migrar el Modelo de la BD es decir preparar el modelo para su posterior migracion
python3 manage.py sqlmigrate [page 0005]	Convertir la migracion a codigo SQL para que se crear la BD en SQLite o hacer los cambios establecidos
python3 manage.py migrate	Establecer finalmente los cambios del script SQL a la BD de SQLite
python3 -m pip install Pillow	Marco error en el campo "ImageField"
pip3 install mysqlclient	Modulo de MySQL para trabajar con Django
python3 -m pip install PyMySQL	

Nota: Cambiar el idioma en español al proyecto, dentro del archivo settings.py modificar lo siguiente a:

LANGUAGE_CODE = 'es-es'

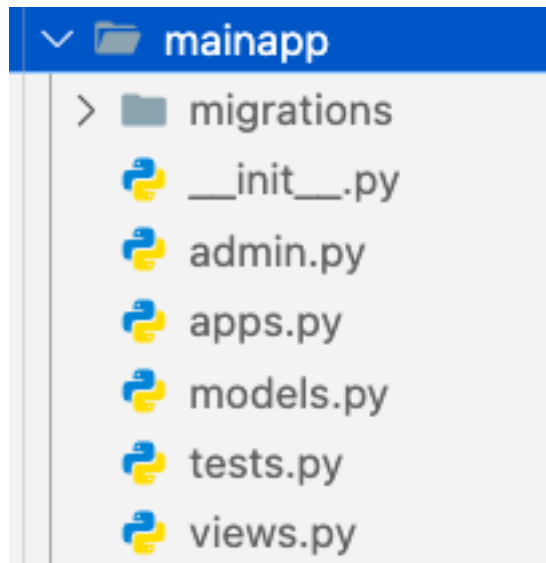
Despues de crear el proyecto queda una estructura de carpetas como las siguientes:



Crear apps de Django

Es un paquete dentro de nuestra aplicación Web Django, cada apps tiene una funcionalidad o aplicación dentro de la aplicación web central... Crear diferentes apps para crear un proyecto en conjunto una apps mas grande. Se recomienda tener una apps para cada Entidad (Tabla) y dentro de esa apps de acuerdo al modelo MVC tener cada uno de los archivos correspondientes. Las apps son reutilizables ...

<code>python3 manage.py startapp mainapp</code>	Crear una app con el nombre de “mainapp” En esta apps se craran las configuraciones principales y generales de todo el proyecto
---	---



Nota: cada apps que sea creada se tiene agregar en el “setting.py”

Application definition

```
INSTALLED_APPS = [
    ... 'django.contrib.admin',
    ... 'django.contrib.auth',
    ... 'django.contrib.contenttypes',
    ... 'django.contrib.sessions',
    ... 'django.contrib.messages',
    ... 'django.contrib.staticfiles',
    ... 'mainapp'
]
```


2.- Vistas, Plantillas (Layout y bloques) y Rutas

Vistas o Views

Crear una primera vista en el fichero de “views.py”, que se encuentra dentro de mi “mainapp”

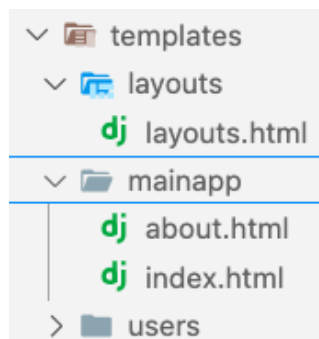
Create your views here.

```
def index(request):  
    return render(request, 'mainapp/index.html', {  
        'title': 'Inicio',  
        'content': ' :: ¡Bienvenido a mi pagina de Inicio! :: '  
    })
```

```
def about(request) :  
    return render(request, 'mainapp/acercade.html', {  
        'title': 'Acerca de Nosotros',  
        'content': 'Somos un equipo destinado al desarrollo de SW'  
    })
```

Plantillas o Templates

Ahora es necesario crear una carpeta llamada: “templates” dentro de “mainapp” dentro de ella agregar las siguientes carpetas y archivos:



Dentro del archivo “layouts.html” agregar el siguiente código:

```

{% load static %}
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>
        {% block title %}{% endblock %}Django Proyecto UTD
    </title>
    <link rel="stylesheet" href="{% static 'css/estilos.css' %}" type="text/css">
</head>
<body>
    <header>
        <div id="logotipo">
            
            <h1>Django Proyecto Web</h1>
        </div>
    </header>
    <nav>
        <ul>
            <li><a href="{% url 'inicio' %}">Inicio</a></li>
            <li><a href="{% url 'acercade' %}">Acerca de</a></li>
        </ul>
    </nav>
    <section id="content">
        <div class="box">
            {% block content %}{% endblock content %}
        </div>
    </section>
    <footer>
        <p>Curso de Django con Dagonline &copy; 2024 | Visitado el: {% now "Y-m-d H:i" %}</p>
    </footer>
</body>
</html>

```

Dentro del archivo "index.html" agregar el siguiente código:

```
{% extends "layouts/layouts.html" %}
```

```
{% block title %}
```

```
    {{title}}
```

```
{% endblock title %}
```

```
{% block content %}
```

```
    <h1>{{title}}</h1>
```

```
    <hr>
```

```
    <p>{{content}}</p>
```

```
{% endblock content %}
```

Dentro del archivo “acercade.html” agregar el siguiente código:

```
{% extends "layouts/layouts.html" %}
```

```
{% block title %}
```

```
    {{title}}
```

```
{% endblock title %}
```

```
{% block content %}
```

```
    <h1>{{title}}</h1>
```

```
    <hr>
```

```
    <p>{{content}}</p>
```

```
{% endblock content %}
```

Rutas o URL's

Es importante definir las rutas Globales y Particulares dentro de cada “App”, para esto se realiza lo siguiente:

1. Crear un nuevo archivo dentro de la app de “mainapp” que se llame: “urls.py”, y agregar el siguiente código:

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

```
from . import views
```

```
urlpatterns = [  
    path("", views.index, name='inicio' ),  
    path('inicio/', views.index, name='inicio'),  
    path('acercade/', views.about, name='acercade')  
]
```

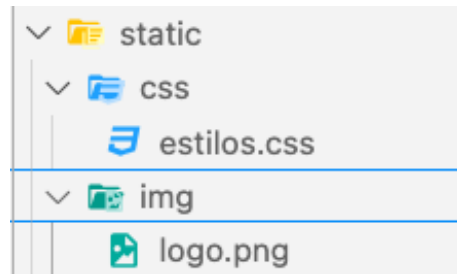
2. Enseguida agregar las rutas en el archivo global de las rutas que se encuentra en "proyectoUTD", el código es:

```
from django.contrib import admin  
from django.urls import path, include
```

```
urlpatterns = [  
    path('admin/', admin.site.urls),  
    path("", include('mainapp.urls')),  
]
```

Estilos (CSS) en plantillas

Es momento de agregar los CSS e IMG dentro del proyecto para esto es necesario crear las siguientes carpetas:



Nota: la carpeta de “static” colocar dentro de “mainapp” al mismo nivel que “templates”

Dentro del archivo “estilos.css”, agregar el siguiente código:

```
/* Aplicar estilos */
```

```
{
    margin: 0px;
    padding: 0px;
    font-family: sans-serif;
    text-decoration: none;
}
```

```
p,h1,h2{
    margin-top: 20px;
    margin-bottom: 20px;
}
```

```
ul,ol{
    margin-left: 20px;
}
```

```
body{
```

```
background-color: #555151;  
}
```

```
/* Estilos cabecera */
```

```
header{  
width: 1212px;  
height: 140px;  
background-color: #2ba977;  
border: 1px solid #333;  
margin: 0px auto;  
}
```

```
#logotipo{  
width: 45%;  
height: 130px;  
margin: 0 auto;  
padding-top: 10px;  
}
```

```
#logotipo img{  
display: block;  
width: 60px;  
float: left;  
margin-top: 20px;  
}
```

```
#logotipo h1{  
display: block;  
float: left;  
letter-spacing: 2px;  
margin-top: 40px;  
margin-left: 20px;  
font-weight: bolder;  
color: white;  
}
```

```
/* Estilos Barra de navegacion */
```

```
nav{  
  width: 1250px;  
  height: 40px;  
  background-color: #1b1e1f;  
  border: 1px solid #333333;  
  margin:0 auto;  
  box-shadow: 0px 22px 22px gray;  
  font-size: 15px;  
}
```

```
nav > ul{  
  text-decoration: none;  
  list-style: none;  
}
```

```
nav > ul > li{  
  line-height: 40px;  
  float: left;  
  
}
```

```
nav > ul > li > a{  
  color:#d1d4d6;  
  display: block;  
  padding-left: 15px;  
  padding-right: 15px;  
}
```

```
nav > ul > li > a:hover{  
  background-color:#2ba977 ;  
  box-shadow: 0px 0px 5px #444 inset;  
  color: white;  
  transition: all 300ms;  
}
```

```
nav > ul > li > ul{  
    display: none;  
    position: absolute;  
    color: white;  
    text-align: left;  
    width: 160px;  
    box-shadow: 0px 2px 2px gray;  
}
```

```
nav > ul > li:hover > ul{  
    display: block;  
    color: white;  
    margin: 0px;  
    list-style: none;  
}
```

```
nav > ul > li:hover > ul li{  
    background-color: #f2f1f0;  
    border-bottom: 1px solid #666;  
    padding: 8px;  
    line-height: 25px;  
    font-size: 13px;  
    transition: all 300ms;  
}
```

```
nav > ul > li:hover > ul li a{  
    color: #666;  
    transition: all 300ms;  
}
```

```
nav > ul > li:hover > ul li:hover{  
    background-color: #2ba977;  
    font-weight: bold;  
}
```

```
nav > ul > li:hover > ul li:hover a{
```



```
display: block;
color:white;
font-size: 14px;
transition: all 300ms;
}
```

```
/* Estilos contenido central */
```

```
.title{
color:#444;
letter-spacing: 1px;
font-size: 30px;
margin-bottom: 10px;
margin-top: 5px;
}
```

```
.date{
display: block;
margin-top: 20px;
margin-bottom: 20px;
color: gray;;
text-align: right;
}
```

```
#content{
width: 1211px;
min-height:930px ;
margin: 0 auto;
margin-top: 30px;
margin-bottom: 30px;
text-align: justify;
line-height: 20px;
}
```

```
.box{
```

```
background-color: white;
width: 95%;
min-height: 930px;
padding: 20px;
border: 1px solid #ddd;
border-radius: 20px;
margin: 0 auto;
}
```

```
.box .image{
width: 45%;
float: left;
margin-right: 40px;
}
```

```
.box .image img{
width: 100%
}
```

```
.box a{
color: #444;
transition: all 300ms;
}
```

```
.box a:hover{
color: #2ba977;
}
```

```
.box form{
width: 80%;
}
```

```
.box form input,
.box form label{
```

```
display: block;
padding: 5px;
padding-left: 0;
margin-bottom: 5px;
font-size: 1rem;
font-style: bold;
text-decoration: underline;
color: black;
}
```

```
.box form input[type="text"],
.box form input[type="email"],
.box form input[type="password"],
.box form textarea,
.box form select
{
width: 100%;
margin-bottom: 10px;
}
```

```
.box form select
{
width: 100px;
}
```

```
.message,
.alert-success{
padding: 20px;
background-color: #2ba977;
color: white;
text-align: center;
margin-bottom: 10px;
font-size: 2rem;
}
```

```
.alert-warning{
padding: 20px;
```

```
background-color: red;
color: white;
text-align: center;
margin-bottom: 10px;
font-size: 2rem;
}
```

```
.title{
color: #444;
letter-spacing: 1px;
font-size: 30px;
padding-bottom: 50px;
margin-bottom: 10px;
margin-top: 5px;
}
```

```
/* Estilos al Footer */
```

```
footer{
width: 1250px;
background-color: #2ba977;
border: 1px solid #333;
color: white;
text-align: center;
padding-top: 20px;
padding-bottom: 20px;
box-shadow: 0px 0px 20px gray;
margin: 0 auto;
}
```

Agregar una imagen de “Django” y renombrarla con el nombre de: “logo.png”

Asegurar de tener las siguientes lineas de codigo en el “layouts.html”:

```
</title>
<link rel="stylesheet" href="{% static 'css/estilos.css'%}" type="text/css">
```

```
</head>
```

```
<body>
```

```
<header>
```

```
<div id="logotipo">
```

```

```

Crear URL Glogables y particulares para cada Apps

Como la cantidad de url's van a incrementar de manera considerable dentro de cada aplicación y eso dependerá de la cantidad de páginas que se realicen es necesario que se realice un pequeño ajuste en donde por cada paquete (apps) se creen solo las url's correspondientes y posterior esas url's se importen en el archivo general de url's del proyecto.

1. Hay que crear un archivo llamado "urls.py" dentro del paquete "mainapp", posteriormente agregar las rutas generadas en ese paquete. Agregar el siguiente código:

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

```
urlpatterns = [
    path("", views.index, name='inicio' ),
    path('inicio/', views.index, name='inicio'),
    path('acercade/', views.about, name='acercade')
]
```

2. Enseguida hay realizar la actualizacion en el archivo global de url's del proyecto dejando unicamente el siguiente código:

```
from django.contrib import admin
from django.urls import path, include
```

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path("", include('mainapp.urls')),
]
```

Nota: de aquí en adelante este proceso se repite y se recomienda que por cada app se agregue un archivo local de rutas de cada una de las páginas que genere esa aplicación y posteriormente se incluya el archivo de rutas en las url's del proyecto.

Filtros en plantillas

Con esto es posible realizar varias cosas de manera más fácil sobre str, list, maps, objetos, etc. En pocas palabras te simplifica varias cosas, algunos ejemplos son:

Nota: agregar el siguiente código en “index.html”

```
<hr>
```

```
{% comment %} incluir un a plantilla dentro de otra {% endcomment %}
```

```
{% include "fecha-actual.html" %}
```

```
<hr>
```

```
{% comment Aplicar filtros en templates %} Corta u o algun tipo de caracteres {% endcomment %}
```

```
<p>
```

```
  {{ "Creado por Dagonline"|cut:"Cr" }}
```

```
</p>
```

```
{% comment %} Cambia un texto vacio por una cadena {% endcomment %}
```

```
<p>
```

```
  {{ texto|default:"Aqui no hay nada" }}
```

```
</p>
```

```
{% comment %} Separar por una coma o cualquier otro elementos de una lista {% endcomment %}
```

```
<p>
```

```
  {{ lenguajes|join:"@ " }}
```

```
</p>
```

```
{% comment %} Mostrar aleatoriamente algun elemento de la lista {% endcomment %}
```

```
<p>
```

```
  {{ lenguajes|random }}
```

```
</p>
```

```
{% comment %} Contar el numero de caracteres {% endcomment %}
```

```
<p>
    {{"dagfiscal@gmail.com"|length}}
</p>
```

```
{% comment %} Texto en Mayusculas {% endcomment %}

<p>
    {{"dagfiscal@gmail.com"|upper}}
</p>
```

```
{% comment %} Texto en Minusculas {% endcomment %}

<p>
    {{"Dagfiscal@Gmail.com"|lower}}
</p>
```

```
{% comment %} Quitar todo el HTML {% endcomment %}

<p>
    {{"<h2>Dag</h2>fiscal<h4>@Gmail.com</h4>"|striptags}}
</p>
```

```
{% comment %}
    Para ver mas información de filtros checar la documentacion
    https://docs.djangoproject.com/en/5.1/ref/templates/builtins/
{% endcomment %}
```