5.1. Modelo-vista-controlador base de los frameworks

Estimados estudiantes:

Para el estudio de este apartado, se solicita revisar:

- La sección Uso de frameworks del capítulo 6 Introducción a los frameworks del texto básico.

5.1. Modelo-vista-controlador base de los frameworks

Algunas pautas interesante sobre los frameworks de ambiente de web:

- Un framework se abarca temas como:lenguaje de programación, librerías, bibliotecas, herramientas y metodologías de desarrollo.
- El uso del patrón Modelo-Vista-Controlador que permita reutilización de código, facilitar los procesos de desarrollo y mejorar el mantenimiento a futuro de las aplicaciones.

5.2. Clasificacón de frameworks de ambiente web según los lenguajes de programación

Para la comprensión de un determinado framework existen dos puntos claves:

- Manejo de conceptos de: paradigma de programación de orientación a objetos, HTML, CSS y JavaScript.
- Entender el modelo-vista-controlador explicado en el apartado anterior de la guía didáctica.

5.2. Clasificacón de frameworks de ambiente web según los lenguajes de programación

PHP:

- Symfony
- Laveral
- Codeigneter

Ruby:

- Ruby on Rails

JAVA:

- Spring
- Ninja
- Apache Struts

Python:

- Django
- Flask
- Pylons
- TurboGears

5.3. Desarrollo de aplicaciones mediante el framework Django

Para el estudio del presente apartado relacionado con el framework Django se hará uso del recurso abierto denominado La guía definitiva de django – Desarrolla aplicaciones Web de forma rápida y sencilla.

5.3. Desarrollo de aplicaciones mediante el framework Django

5.3.1. Introducción

Algunas características del framework:

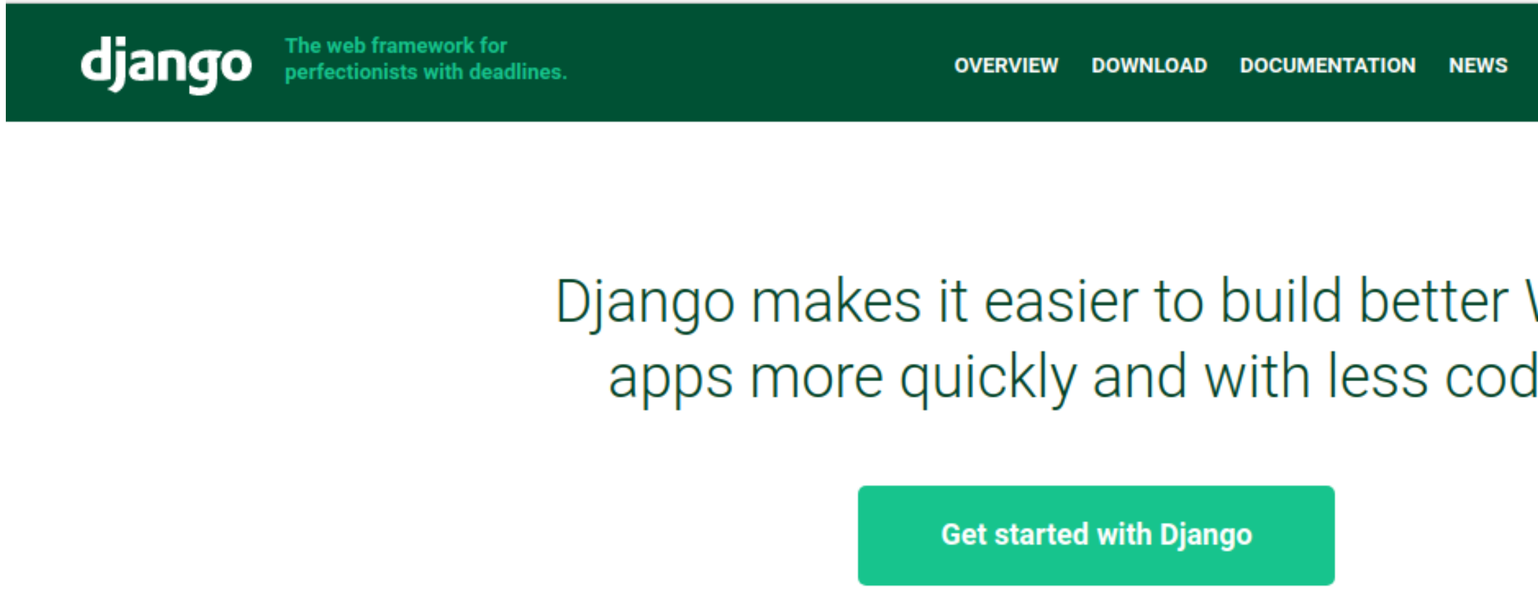
- Ahorra tiempo en el desarrollo de aplicaciones.
- Genera atajos para algunas tareas frecuentes.
- Maneja el patrón Modelo-Vista-Controlador mediante su filosofía Modelo-Vista-Template.
- Existe un acoplamiento débil en el desarrollo de componentes.

5.3. Desarrollo de aplicaciones mediante el framework Django

5.3.1. Introducción

Importante:

- La documentación oficial de Django siempre será un recurso importante para profundizar en algunas temáticas.
  - https://www.djangoproject.com/

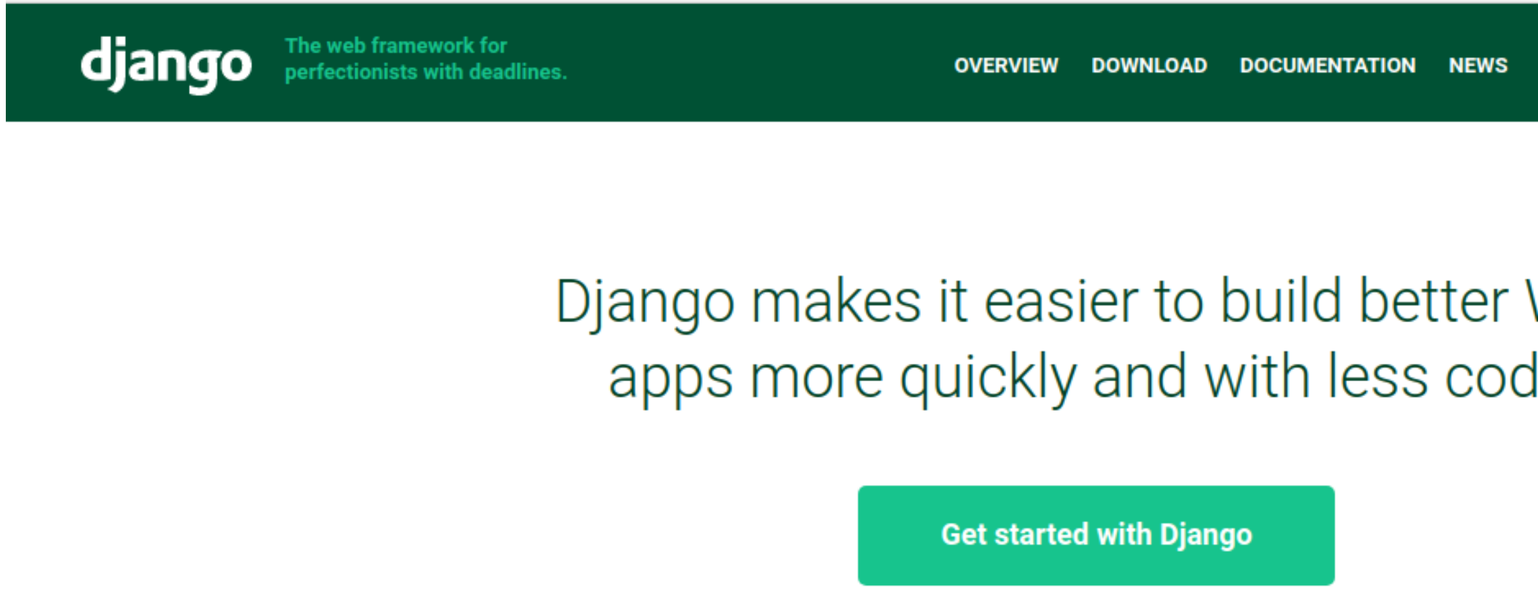


5.3. Desarrollo de aplicaciones mediante el framework Django

5.3.1. Introducción

Importante:

- La documentación oficial de Django siempre será un recurso importante para profundizar en algunas temáticas.
  - https://www.djangoproject.com/



5.3. Desarrollo de aplicaciones mediante el framework Django

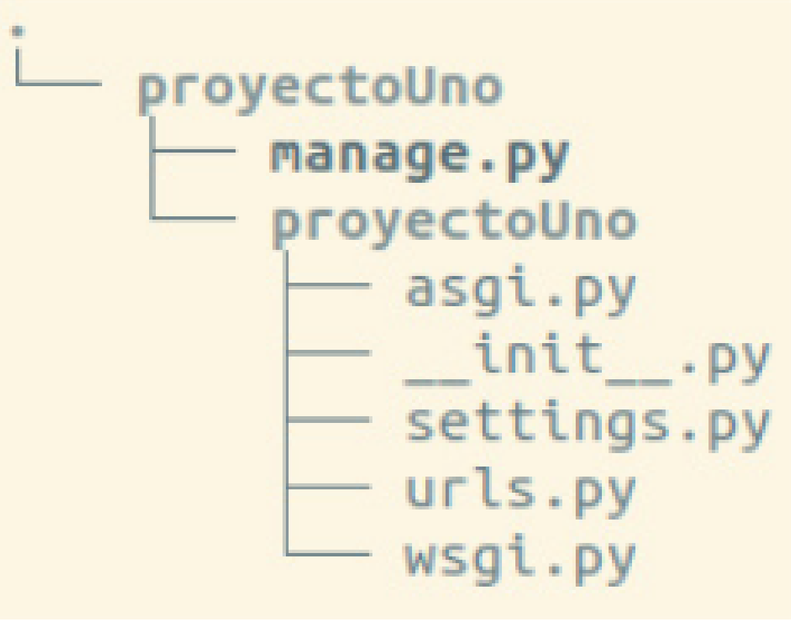
Instalación de Django

- Usar el gestor de paquetes de Python: pip
- Para instalar el framework se puede usar la forma:
  - pip install Django

5.3. Desarrollo de aplicaciones mediante el framework Django

Creación de un proyecto en Django

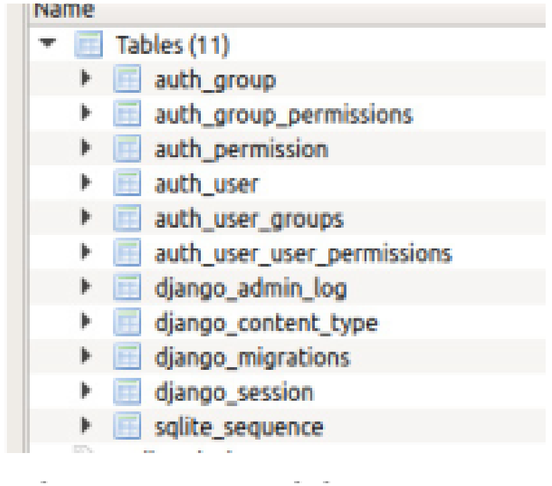
- Para la creación de un proyecto se puede usar el comando:
  - django-admin startproject [nombre del proyecto]



5.3. Desarrollo de aplicaciones mediante el framework Django

Para crear las tablas por defecto del proyecto se usar el siguiente comando:

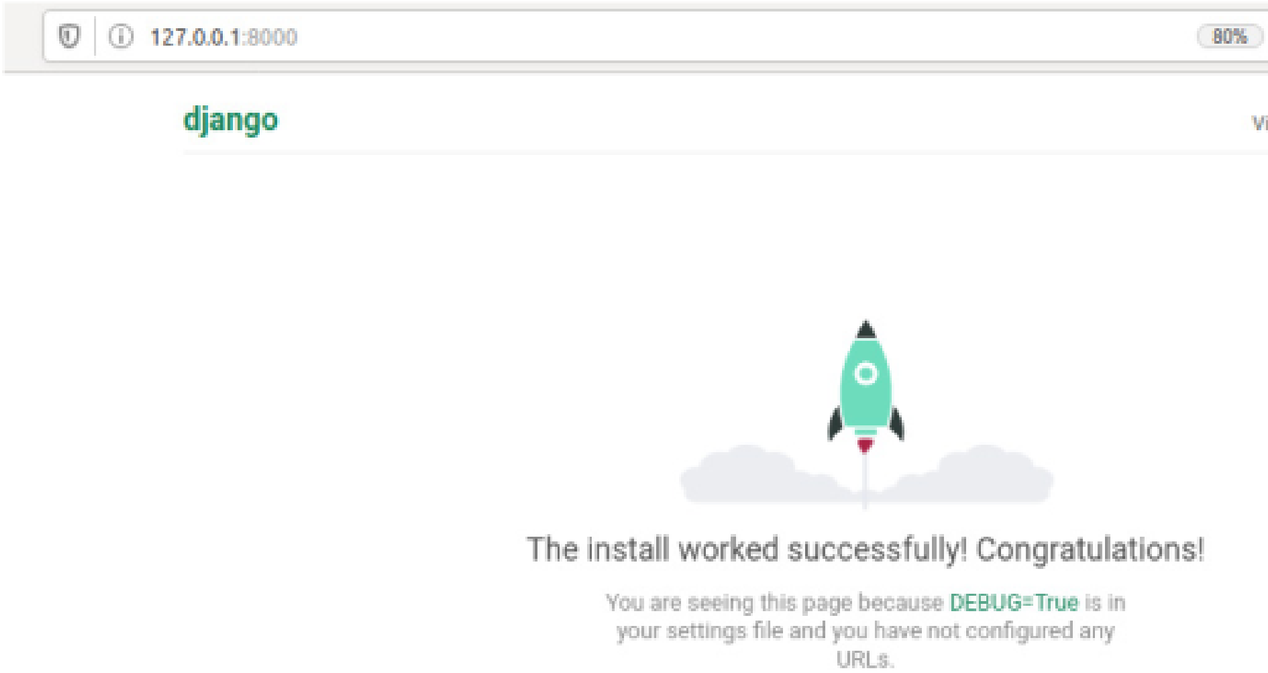
- python manage.py migrate



5.3. Desarrollo de aplicaciones mediante el framework Django

Para ver en acción el proyecto se usa el siguiente comando:

- python manage.py runserver



5.3. Desarrollo de aplicaciones mediante el framework Django

En un proyecto se puede agregar muchas aplicaciones. El comando para crear una aplicación es:

- python manage.py startapp [nombre de aplicacion]

```
.
├── administrativo
│   ├── admin.py
│   ├── apps.py
│   ├── __init__.py
│   ├── migrations
│   │   └── __init__.py
│   ├── models.py
│   ├── tests.py
│   └── views.py
├── manage.py
└── proyectoUno
    ├── asgi.py
    ├── __init__.py
    ├── __pycache__
    │   ├── __init__.cpython-36.pyc
    │   └── settings.cpython-36.pyc
    ├── settings.py
    ├── urls.py
    └── wsgi.py
```

5.3. Desarrollo de aplicaciones mediante el framework Django

En un proyecto se puede agregar muchas aplicaciones. El comando para crear una aplicación es:

- python manage.py startapp [nombre de aplicacion]

A continuación se describe los principales archivos que se crean en cada aplicación:

- admin.py permite agregar las clases o entidades que serán puestas en el proceso de administración del proyecto
- __init__.py, es el archivo que permite indicar a Python que el directorio sea considerado com paquete.
- models.py, se ubican las clases en Python que luego se convierten tablas de la base de datos.
- views.py se gestiona principalmente a través de funciones.

5.3. Desarrollo de aplicaciones mediante el framework Django

Es momento de usar el shell de Django para interactuar con la base de datos a través del uso del ORM de Django mediante las clases del modelo. Para acceder se usa el comando:

- python manage.py shell

```
rom django.db import models

 Create your models here.

lass Estudiante(models.Model):
   nombre = models.CharField(max_length=30)
   apellido = models.CharField(max_length=30)
   cedula = models.CharField(max_length=30, unique=True)

   def __str__(self):
       return "%s %s %s" % (self.nombre,
                self.apellido,
                self.cedula)

lass NumeroTelefonico(models.Model):
   telefono = models.CharField(max_length=100)
   tipo = models.CharField(max_length=100)
   estudiante = models.ForeignKey(Estudiante, on_delete=models.CASCADE

   def __str__(self):
       return "%s %s" % (self.telefono, self.tipo)
```

```
# La sentencias mostradas a continuación se usando el shell de Django
# Primero, se importa las clases del modelo
from administrativo.models import Estudiante, NumeroTelefonico
# Inserción de datos
# se crea un objeto de tipo Estudiante
# se hace referencia a los atributos de la clase
e1 = Estudiante(nombre="Luisa", apellido="Tene", cedula="1100991133")
e2 = Estudiante(nombre="Rolando", apellido="Sarango", cedula="1100991122")
# se guarda el objeto en la base de datos, mediante el método save()
e1.save()
e2.save()
# se crea un objeto de tipo NumeroTelefonico
# para el atributo estudiante del objeto t, se usa el objeto de tipo Estudiante e
t = NumeroTelefonico(telefono="09988776655", tipo="particular", estudiante=e1)
```

5.3. Desarrollo de aplicaciones mediante el framework Django

Es momento de usar el shell de Django para interactuar con la base de datos a través del uso del ORM de Django mediante las clases del modelo. Para acceder se usa el comando:

- python manage.py shell

```python
from django.db import models

# Create your models here.

class Estudiante(models.Model):
    nombre = models.CharField(max_length=30)
    apellido = models.CharField(max_length=30)
    cedula = models.CharField(max_length=30, unique=True)

    def __str__(self):
        return "%s %s %s" % (self.nombre,
                self.apellido,
                self.cedula)

class NumeroTelefonico(models.Model):
    telefono = models.CharField(max_length=100)
    tipo = models.CharField(max_length=100)
    estudiante = models.ForeignKey(Estudiante, on_delete=models.CASCADE)

    def __str__(self):
        return "%s %s" % (self.telefono, self.tipo)

```

```
# Consulta de datos
# Seleccionar todos los datos del modelo Estudiante
estudiantes = Estudiante.objects.all()
for e in estudiantes:
  print(e)
#salida:
####
# Rolando Sarango 1100991122
# Luisa Tene 1100991133
####
# Filtrar todos los estudiantes que tiene como valor en nombre: “Rolando"
estudiantes = Estudiante.objects.filter(nombre="Rolando").all()
# Filtrar todos los estudiantes que tengan un número de teléfono con la secuencia “8888"
# numerotelefonico es la clase asociada, se recuerda la relación
Estudiante.objects.filter(numerotelefonico__telefono__contains="8888")
# Filtrar todos los números telefónicos que pertenecen a estudiantes que tengan en su apellido la secuencia “Te”
telefonos = NumeroTelefonico.objects.filter(estudiante__apellido__contains="Te")
for t in telefonos:
  print(“%s - %s" % (t, t.estudiante))
#salida
##
# 09988776655 particular - Luisa Tene 1100991133
```

Gracias

rrelizalde@utpl.edu.ec
@reroes