Informe de Laboratorio 06

Tema: Django(Admin)

Nota

Estudiante	Escuela	Asignatura
Kevin Andree Llacma Quispe kllacma@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programacion web 2 Semestre: I Código: 20200585

Laboratorio	Tema	Duración
06	Django(Admin)	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	-	-

Programación Web
Laboratorio 06
Tema: Django(Admin)

12 de junio de 2024

1. Marco teorico

1.1. Django

- Django es un framework web Python con el cuál el desarrollo de sitios web son rápidos, seguros y sobre todo fáciles de mantener.
- Django se ocupa de gran parte de las molestias del desarrollo web, por lo que puede concentrarse en escribir su aplicación sin necesidad de reinventar la rueda.
- Es software libre, tiene una comunidad próspera y activa, excelente documentación y muchas opciones de soporte gratuito y de pago.

2. Desarollo del lab

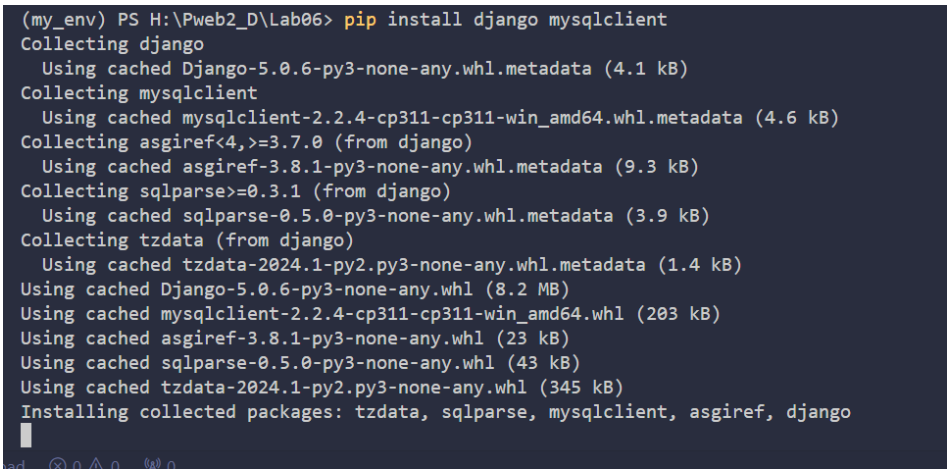
2.1. Descripcion

Proyecto Tienda online

Para empezar este proyecto se creo un entorno virtual

Luego se procedio a instalar django con mysqlclient para base de datos

A continuacion se creo un proyecto llamado "tiendaonlinez se creo la app "shop"



```
(my_env) PS H:\Pweb2_D\Lab06> django-admin startproject tienda_online
(my_env) PS H:\Pweb2_D\Lab06> cd .\tienda_online\
(my_env) PS H:\Pweb2_D\Lab06\tienda_online> python manage.py startapp shop
(my_env) PS H:\Pweb2_D\Lab06\tienda_online>
```

Luego se configuro settings.py para la base de datos, si fuera sqlite no se tendria que identificar

```
77  DATABASES = {
78      'default': {
79          'ENGINE': 'django.db.backends.mysql',
80          'NAME': 'tienda_online',
81          'USER': 'root',
82          'PASSWORD': '123QWERtyuiop789',
83          'HOST': 'localhost',
84          'PORT': '3306',
85      }
86  }
```

Ahora hacemos lo modelos tablas, en este caso que es una tienda se busco que tablas se podria agregar en un caso real (en models.py)
Usuario categoria y producto

- A usuario le pide nombre, email, correro y contraseña
- categoria tiene solo nombre y descripcion
- Producto ademas de eso tiene un precio y un stock

```python
class User(models.Model):
    first_name = models.CharField(max_length=50)
    last_name = models.CharField(max_length=50)
    email = models.EmailField(unique=True)
    password = models.CharField(max_length=100)

    def __str__(self):
        return f'{self.first_name} {self.last_name}'


class Category(models.Model):
    name = models.CharField(max_length=100)
    description = models.TextField(blank=True)

    def __str__(self):
        return self.name


class Product(models.Model):
    name = models.CharField(max_length=100)
    description = models.TextField()
    price = models.DecimalField(max_digits=10, decimal_places=2)
    stock_quantity = models.PositiveIntegerField()

    def __str__(self):
        return self.name
```

Continuamos con los siguietne modelos y explicando un poco en que consiten ProductCategory: Relación entre productos y categorías

- product: Producto relacionado
- category: Categoría relacionada

Order: Representa los pedidos realizados por los usuarios

- user: Usuario que realizó el pedido
- orderdate: Fecha en la que se creó el pedido
- status: Estado del pedido

OrderDetail: Detalles específicos de cada producto en un pedido.

- order: Pedido al que pertenece.
- product: Producto del pedido.
- quantity: Cantidad de producto en el pedido.
- price: Precio del producto en el momento del pedido.

ShoppingCart: Carritos de compras de los usuarios

- user: Usuario al que pertenece el carrito

- createdat: Fecha de creación del carrit

CartDetail: Detalles de los productos en un carrito de compas.

- shoppingcart: Carrito de compras al que pertenece
- product: Producto en el carrito
- quantity: Cantidad del produto en el carrito

```python
class ProductCategory(models.Model):
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    category = models.ForeignKey(Category, on_delete=models.CASCADE)


class Order(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    order_date = models.DateTimeField(auto_now_add=True)
    status = models.CharField(max_length=50)

    def __str__(self):
        return f'Order {self.id} by {self.user}'


class OrderDetail(models.Model):
    order = models.ForeignKey(Order, on_delete=models.CASCADE)
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    quantity = models.PositiveIntegerField()
    price = models.DecimalField(max_digits=10, decimal_places=2)


class ShoppingCart(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    created_at = models.DateTimeField(auto_now_add=True)


class CartDetail(models.Model):
    shopping_cart = models.ForeignKey(ShoppingCart, on_delete=models.CASCADE)
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    quantity = models.PositiveIntegerField()
```

Se busco informacion para hacer los requisitos de manera real
Direccion y Pago: Address: Direcciones de los usuarios

- user: Usuario al que pertenece la dirección
- addressline: Línea prncipal de la dirección
- city: Ciudad
- state: Estado o provncia
- zipcode: Codigo postal
- country: Pais

Payment: Pagos realizados para los pedidos

- order: Pedido al que se relaciona el pago
- paymentdate: Fecha del pago
- amount: Monto pagad
- paymentmethod: Método de pago

```python
class Address(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    address_line = models.CharField(max_length=255)
    city = models.CharField(max_length=100)
    state = models.CharField(max_length=100)
    zip_code = models.CharField(max_length=20)
    country = models.CharField(max_length=100)


class Payment(models.Model):
    order = models.ForeignKey(Order, on_delete=models.CASCADE)
    payment_date = models.DateTimeField(auto_now_add=True)
    amount = models.DecimalField(max_digits=10, decimal_places=2)
    payment_method = models.CharField(max_length=50)
```

Luego los registramos en Admin.py

```python
from django.contrib import admin

# Register your models here.

from .models import User, Category, Product, ProductCategory, Order, OrderDetail, ShoppingCart, CartDetail, Address, Payment

admin.site.register(User)
admin.site.register(Category)
admin.site.register(Product)
admin.site.register(ProductCategory)
admin.site.register(Order)
admin.site.register(OrderDetail)
admin.site.register(ShoppingCart)
admin.site.register(CartDetail)
admin.site.register(Address)
admin.site.register(Payment)
```

Ahora creamos y aplicamos migraciones

```
(my_env) PS H:\Pweb2_D\Lab06\tienda_online> py .\manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
(my_env) PS H:\Pweb2_D\Lab06\tienda_online>
```

Ahora creando un superusuario

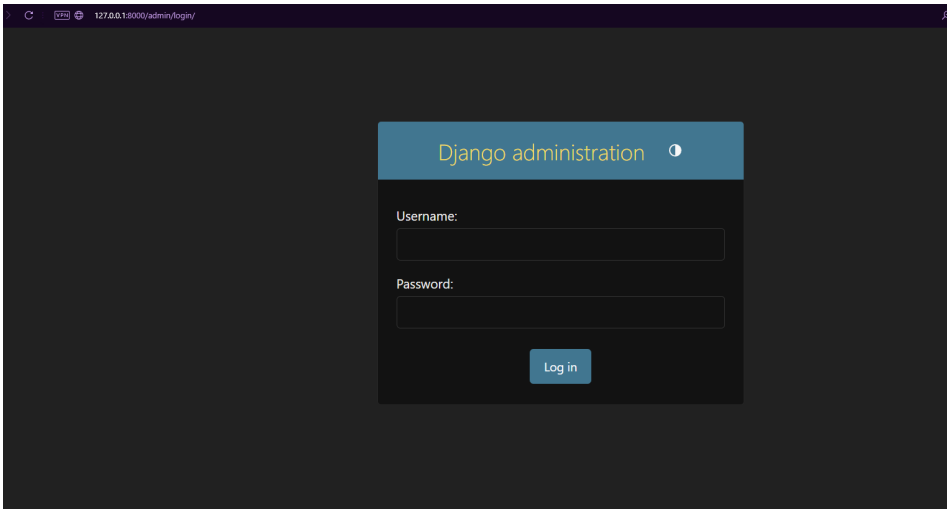
```
(my_env) PS H:\Pweb2_D\Lab06\tienda_online> py .\manage.py createsuperuser
Username (leave blank to use 'usuario'): admin
Email address:
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
This password is entirely numeric.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
(my_env) PS H:\Pweb2_D\Lab06\tienda_online>
```

Iniciamos el server

```
(my_env) PS H:\Pweb2_D\Lab06\tienda_online> py .\manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
June 11, 2024 - 22:35:04
Django version 5.0.6, using settings 'tienda_online.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Observamos como se ve admin y accedemos con superusuario



Accdemos y observamos la tablas que creamos