



INSTITUTO TECNOLÓGICO DE NUEVO LAREDO
INGENIERÍA EN SISTEMAS COMPUTACIONALES



PROGRAMACIÓN MULTIPARADIGMA

UNIDAD 2 **PRACTICAS**

Docente

Ing. Luis Daniel Castillo García

INTEGRANTES:

- JUAN ESTEBAN BALTIERREZ TOBON 18100152
- JOSE ANGEL CABRERA MORALES 18100155
- ANTONIO DE LA CRUZ RIVERA 18100163

30 de octubre del 2022

Practica #1 Realizar una aplicación que se conecte a postgresql y contenga lo siguiente

1. Al menos 3 entidades (ejemplo clase de entidad Persona)
2. Realizar CRUD de las 3 entidades
3. Usar archivo de logs
4. Utilizar pool de conexiones

Comprobación

1. Al menos 3 entidades (ejemplo clase de entidad Persona)

Entidades: Autor, Genero, Libro

Autor

```
class Autor:
    def __init__(self, nombre, apellido, nacionalidad, id=0):
        self._id = id
        self._nombre = nombre
        self._apellido = apellido
        self._nacionalidad = nacionalidad

    @classmethod
    def fromTuple(cls, autor):
        return cls(autor[1], autor[2], autor[3], id=autor[0])

# Getters y Setters
@property
def id(self):
    return self._id

@id.setter
def id(self, value):
    self._id = value
```

Genero

```
class Genero:
    def __init__(self, nombre, id=0):
        self._id = id
        self._nombre = nombre

    @classmethod
    def fromTuple(cls, genero):
        return cls(genero[1], id=genero[0])

# Getters y Setters
@property
def id(self):
    return self._id

@id.setter
def id(self, value):
    self._id = value
```

Libro

```
class Libro:
    def __init__(self, titulo, autor, anio_publicacion, genero, id=0):
        self._id = id
        self._titulo = titulo
        self._autor = autor
        self._anio_publicacion = anio_publicacion
        self._genero = genero

    @classmethod
    def fromTuple(cls, libro):
        return cls(libro[1], libro[2], libro[3], libro[4], id=libro[0])

# Getters y Setters
@property
def id(self):
    return self._id

@id.setter
def id(self, value):
    self._id = value
```

2. Realizar CRUD de las 3 entidades

DTO

Autores

```
class Autores:
    @staticmethod
    def getAll():
        try:
            with Pool() as pool:
                pool.execute("Select * from Autores")
                autores = [Autor.fromTuple(autor) for autor in pool.fetchall()]
                return autores
        except Exception as e:
            return None

    @staticmethod
    def getByPK(id):
        try:
            with Pool() as pool:
                pool.execute("Select * from Autores where id = %s", (id,))
                autor = Autor.fromTuple(pool.fetchone())
                return autor
        except Exception as e:
            return None

    @staticmethod
    def create(autor=Autor):
        try:
            with Pool() as pool:
                pool.execute('Insert into Autores (nombre,apellido, nacionalidad) values (%s,%s,%s)',
                    (autor.nombre, autor.apellido, autor.nacionalidad))
                return pool.rowcount
        except Exception as e:
            return None

    @staticmethod
    def updateByPK(id, autor=Autor):
        try:
            with Pool() as pool:
                pool.execute('Update Autores set nombre = %s, apellido = %s, nacionalidad = %s where id = %s',
                    (autor.nombre, autor.apellido, autor.nacionalidad, id))
                return pool.rowcount
        except Exception as e:
            return None
```

```
    @staticmethod
    def deleteByPK(id):
        try:
            with Pool() as pool:
                pool.execute('Delete from Autores where id = %s', (id,))
                return pool.rowcount
        except Exception as e:
            return None
```

Generos

```
class Generos:
    @staticmethod
    def getAll():
        try:
            with Pool() as pool:
                pool.execute("Select * from Generos")
                generos = [Genero.fromTuple(genero) for genero in pool.fetchall()]
                return generos
        except Exception as e:
            return None

    @staticmethod
    def getByPK(id):
        try:
            with Pool() as pool:
                pool.execute("Select * from Generos where id = %s", (id,))
                genero = Genero.fromTuple(pool.fetchone())
                return genero
        except Exception as e:
            return None

    @staticmethod
    def create(genero=Genero):
        try:
            with Pool() as pool:
                pool.execute('Insert into Generos (nombre) values (%s)', (genero.nombre,))
                return pool.rowcount
        except Exception as e:
            return None

    @staticmethod
    def updateByPK(id, genero=Genero):
        try:
            with Pool() as pool:
                pool.execute('Update Generos set nombre = %s where id = %s', (genero.nombre, id))
                return pool.rowcount
        except Exception as e:
            return None

    @staticmethod
    def deleteByPK(id):
        try:
            with Pool() as pool:
                pool.execute('Delete from Generos where id = %s', (id,))
                return pool.rowcount
        except Exception as e:
            return None
```

Libros

```
class Libros:
    @staticmethod
    def getAll():
        try:
            with Pool() as pool:
                pool.execute("Select id, titulo, autor, anio_publicacion, genero from Libros")
                libros = [Libro.fromTuple(libro) for libro in pool.fetchall()]
                return libros
        except Exception as e:
            return None

    @staticmethod
    def getByPK(id):
        try:
            with Pool() as pool:
                pool.execute("Select id, titulo, autor, anio_publicacion, genero from Libros where id = %s", (id,))
                libro = Libro.fromTuple(pool.fetchone())
                return libro
        except Exception as e:
            return None

    @staticmethod
    def create(libro=Libro):
        try:
            with Pool() as pool:
                pool.execute('Insert into Libros (titulo, autor, anio_publicacion, genero) values (%s, %s, %s, %s)',
                    (libro.titulo, libro.autor, libro.anio_publicacion, libro.genero))
                return pool.rowcount
        except Exception as e:
            return None

    @staticmethod
    def updateByPK(id, libro=Libro):
        try:
            with Pool() as pool:
                pool.execute(
                    'Update Libros set titulo = %s, autor = %s, anio_publicacion = %s, genero = %s where id = %s',
                    (libro.titulo, libro.autor, libro.anio_publicacion, libro.genero, id))
                return pool.rowcount
        except Exception as e:
            return None

    @staticmethod
    def deleteByPK(id):
        try:
            with Pool() as pool:
                pool.execute('Delete from Libros where id = %s', (id,))
                return pool.rowcount
        except Exception as e:
            return None
```

Controladores

Autores

```
def MostrarAutores():
    autores = Autores.getAll()
    if not autores:
        print('No hay autores registrados')
        return
    print('Listado de autores')
    print('ID\tNombre\tApellido\tNacionalidad')
    for autor in autores:
        print(f'{autor.id}\t{autor.nombre}\t{autor.apellido}\t{autor.nacionalidad}')

def MostrarAutor():
    id = input('Ingrese el ID del autor: ')
    autor = Autores.getByPK(id)
    if not autor:
        print('El autor no existe')
        return
    print('Autor')
    print(f'ID: {autor.id}\tNombre: {autor.nombre}\tApellido: {autor.apellido}\tNacionalidad: {autor.nacionalidad}')

def AgregarAutor():
    nombre = input('Ingrese el nombre del autor: ')
    apellido = input('Ingrese el apellido del autor: ')
    nacionalidad = input('Ingrese la nacionalidad del autor: ')
    autor = Autor(nombre, apellido, nacionalidad)
    res = Autores.create(autor)
    if res > 0:
        print('Autor agregado')
    else:
        print('No se agregó el autor')
```

```
def EditarAutor():
    id = input('Ingrese el ID del autor: ')
    autor = Autores.getByPK(id)
    if not autor:
        print('El autor no existe')
        return
    nombre = input(f'Ingrese el nombre del autor ({autor.nombre}): ') or autor.nombre
    apellido = input(f'Ingrese el apellido del autor ({autor.apellido}): ') or autor.apellido
    nacionalidad = input(f'Ingrese la nacionalidad del autor ({autor.nacionalidad}): ') or autor.nacionalidad
    autor = Autor(nombre, apellido, nacionalidad)
    res = Autores.updateByPK(id, autor)
    if res:
        print('Autor editado')
    else:
        print('No se editó el autor')

def EliminarAutor():
    id = input('Ingrese el ID del autor: ')
    autor = Autores.getByPK(id)
    if not autor:
        print('El autor no existe')
        return
    res = Autores.deleteByPK(id)
    if res:
        log.debug(f'Se elimino el autor {autor}')
        print('Autor eliminado')
    else:
        print('No se eliminó el autor')
```

Generos

```
def MostrarGeneros():
    generos = Generos.getAll()
    if not generos:
        print('No hay generos')
        return
    print('Listado de generos')
    print('ID\tNombre')
    for genero in generos:
        print(f'{genero.id}\t{genero.nombre}')

def MostrarGenero():
    id = input('Ingrese el ID del genero: ')
    genero = Generos.getByPK(id)
    if not genero:
        print('El genero no existe')
        return
    print('Genero')
    print('ID\tNombre')
    print(f'{genero.id}\t{genero.nombre}')

def AgregarGenero():
    nombre = input('Ingrese el nombre del genero: ')
    genero = Genero(nombre)
    Generos.create(genero)
    print('Genero agregado')

def EditarGenero():
    id = input('Ingrese el ID del genero: ')
    genero = Generos.getByPK(id)
    if not genero:
        print('El genero no existe')
        return
    nombre = input(f'Ingrese el nombre del genero ({genero.nombre}): ') or genero.nombre
    genero = Genero(nombre)
    Generos.updateByPK(id, genero)
    print('Genero editado')

def EliminarGenero():
    id = input('Ingrese el ID del genero: ')
    genero = Generos.getByPK(id)
    if not genero:
        print('El genero no existe')
        return

    res = Generos.deleteByPK(id)

    if res:
        log.debug(f"Se eliminó el genero {genero}")
        print('Genero eliminado')
    else:
        print('No se eliminó el genero')
```


Libros

```
def MostrarLibros():
    libros = Libros.getAll()
    if not libros:
        print('No hay libros')
        return
    print('Listado de libros')
    print('ID\tTitulo\tAutor\tAño\tGenero')
    for libro in libros:
        print(f'{libro.id}\t{libro.titulo}\t{libro.autor}\t{libro.anio_publicacion}\t{libro.genero}')

def MostrarLibro():
    id = input('Ingrese el ID del libro: ')
    libro = Libros.getByPK(id)
    if not libro:
        print('El libro no existe')
        return
    print('Libro')
    print('ID\tTitulo\tAutor\tAño\tGenero')
    print(f'{libro.id}\t{libro.titulo}\t{libro.autor}\t{libro.anio_publicacion}\t{libro.genero}')

def AgregarLibro():
    titulo = input('Ingrese el titulo del libro: ')
    print("Autores disponibles [0-10]")
    autores = Autores.getAll()[0:10]
    print("ID\tNombre")
    for autor in autores:
        print(f'{autor.id}\t{autor.nombre}')

    autor = input('Ingrese el autor del libro: ')
    año = input('Ingrese el año del libro: ')
    print("Generos disponibles [0-10]")
    generos = Generos.getAll()[0:10]
    print("ID\tNombre")
    for genero in generos:
        print(f'{genero.id}\t{genero.nombre}')

    genero = input('Ingrese el genero del libro: ')
    libro = Libro(titulo, autor, año, genero)
    Libros.create(libro)
    print('Libro agregado')
```

```

def EditarLibro():
    id = input('Ingrese el ID del libro: ')
    libro = Libros.getByPK(id)
    if not libro:
        print('El libro no existe')
        return
    titulo = input(f'Ingrese el titulo del libro ({libro.titulo}): ') or libro.titulo
    print("Autores disponibles [0-10]")
    autores = Autores.getAll()[0:10]
    if not autores:
        print('No hay autores, registre uno primero')
        return
    print("ID\tNombre")
    for autor in autores:
        print(f"{autor.id}\t{autor.nombre}")

    autor = input(f'Ingrese el id del autor del libro ({libro.autor}): ') or libro.autor
    año = input(f'Ingrese el año del libro ({libro.anio_publicacion}): ') or libro.anio_publicacion

    print("Generos disponibles [0-10]")
    generos = Generos.getAll()[0:10]
    if not generos:
        print('No hay generos, registre uno primero')
        return
    print("ID\tNombre")
    for genero in generos:
        print(f"{genero.id}\t{genero.nombre}")

    genero = input(f'Ingrese el id del genero del libro ({libro.genero}): ') or libro.genero

    libro = Libro(titulo, autor, año, genero)
    res = Libros.updateByPK(id, libro)

    if res:
        print('Libro editado')
    else:
        print('No se editó el libro')

```

```

def EliminarLibro():
    id = input('Ingrese el ID del libro: ')
    libro = Libros.getByPK(id)
    if not libro:
        print('El libro no existe')
        return

    res = Libros.deleteByPK(id)

    if res:
        log.debug(f'Se elimino el libro {libro}')
        print('Libro eliminado')
    else:
        print('No se eliminó el libro')

```

3. Usar archivo de logs

Configuracion

```
import logging as log

log.basicConfig(level=log.DEBUG,
                format="%(asctime)s: %(levelname)s [%(filename)s]: %(lineno)s %(message)s",
                datefmt='%I:%M:%S %p',
                handlers=[
                    log.FileHandler('./log.log'),
                    log.StreamHandler()
                ])
|
```

Usos

```
class CursorPool:
    conn = Connection()

    def __init__(self):
        self._conn = None
        self._cursor = None

    def __enter__(self):
        self._conn = self.conn.getConnection()
        self._cursor = self._conn.cursor()
        return self._cursor

    def __exit__(self, exceptionType, exceptionValue, exceptionDetail):
        if exceptionValue:
            log.error(f"Exception: {exceptionValue} {exceptionDetail}")
            self._conn.rollback()
        else:
            self._conn.commit()
            self._cursor.close()
            self.conn.putConnection(self._conn)
```

```
res = generos.delete(pk=id)

if res:
    log.debug(f"Se eliminó el genero {genero}")
    print('Genero eliminado')
else:
    print('No se eliminó el genero')
```

```
if res:
    log.debug(f'Se elimino el libro {libro}')
    print('Libro eliminado')
else:
    print('No se eliminó el libro')
```

Resultado

```
11:40:21 PM:DEBUG [cursor.py]: 19 Exception: 'NoneType' object is not subscriptable <traceback object at 0x000001EFDABCA000>
11:43:48 PM:DEBUG [cursor.py]: 19 Exception: 'NoneType' object is not subscriptable <traceback object at 0x000001F8CE445F80>
11:46:42 PM:DEBUG [cursor.py]: 19 Exception: inserci3n o actualizaci3n en la tabla 4libros4 viola la llave for4nea 4libros_autor_fkey4
DETAIL: La llave (autor)=(2) no est4 presente en la tabla 4autores4.
<traceback object at 0x000001F8CE446040>
11:48:15 PM:DEBUG [cursor.py]: 19 Exception: inserci3n o actualizaci3n en la tabla 4libros4 viola la llave for4nea 4libros_autor_fkey4
DETAIL: La llave (autor)=(2) no est4 presente en la tabla 4autores4.
<traceback object at 0x0000020313501F80>
11:49:12 PM:DEBUG [cursor.py]: 19 Exception: 'NoneType' object is not subscriptable <traceback object at 0x0000020313502C40>
10:36:27 AM:ERROR [cursor.py]: 19 Exception: update o delete en 4generos4 viola la llave for4nea 4libros_genero_fkey4 en la tabla 4libros4
DETAIL: La llave (id)=(1) todav4a es referida desde la tabla 4libros4.
<traceback object at 0x000001ABFB610A00>
10:36:55 AM:ERROR [cursor.py]: 19 Exception: update o delete en 4autores4 viola la llave for4nea 4libros_autor_fkey4 en la tabla 4libros4
DETAIL: La llave (id)=(1) todav4a es referida desde la tabla 4libros4.
<traceback object at 0x000001ABFB610CC0>
10:56:56 AM:DEBUG [app.py]: 10 Se establecio conexion con la base de datos
10:57:21 AM:ERROR [cursor.py]: 19 Exception: 'NoneType' object is not subscriptable <traceback object at 0x0000025FD211600>
11:01:49 AM:DEBUG [libros.py]: 101 Se elimino el libro ID: 3, T4tulo: La Tregua, Autor: 3, A4o: 1974, G4nero: 2
11:02:02 AM:ERROR [cursor.py]: 19 Exception: 'NoneType' object is not subscriptable <traceback object at 0x0000025FD2128C0>
11:02:15 AM:DEBUG [autores.py]: 66 Se elimino el autor ID: 3, Nombre: Mario, Apellido: Benedetti, Nacionalidad: Argentina
10:02:30 AM:DEBUG [generos.py]: 57 Se elimino el genero ID: 2, Nombre: Drama
```

4. Utilizar pool de conexiones

Conexi3n

```
class Connection:
    _host= config['HOST']
    _port= config['PORT']
    _db= config['DB']
    _user= config['USER']
    _password= config['PASS']
    _conn = None
    _pool = None
    _min_conn = 1
    _max_conn = 5

    def __init__(self):
        pass

    @classmethod
    def obtenerPool(cls):
        if cls._pool is None:
            cls._pool = pool.SimpleConnectionPool(cls._min_conn, cls._max_conn, user=cls._user, password=cls._password, host=cls._host, port=cls._port, database=cls._db)
        return cls._pool

    @classmethod
    def getConnection(cls):
        conn = cls.obtenerPool().getconn()
        return conn

    @classmethod
    def putConnection(cls, conexion: connection):
        cls.obtenerPool().putconn(conexion)

    @classmethod
    def closeAll(cls):
        cls.obtenerPool().closeAll()
```

Cursor

```

class CursorPool:
    conn = Connection()

    def __init__(self):
        self._conn = None
        self._cursor = None

    def __enter__(self):
        self._conn = self.conn.getConnection()
        self._cursor = self._conn.cursor()
        return self._cursor

    def __exit__(self, exceptionType, exceptionValue, exceptionDetail):
        if exceptionValue:
            log.error(f"Exception: {exceptionValue} {exceptionDetail}")
            self._conn.rollback()
        else:
            self._conn.commit()
            self._cursor.close()
            self.conn.putConnection(self._conn)

```

```

class CursorPool:
    conn = Connection()

    def __init__(self):
        self._conn = None
        self._cursor = None

    def __enter__(self):
        self._conn = self.conn.getConnection()
        self._cursor = self._conn.cursor()
        return self._cursor

    def __exit__(self, exceptionType, exceptionValue, exceptionDetail):
        if exceptionValue:
            log.error(f"Exception: {exceptionValue} {exceptionDetail}")
            self._conn.rollback()
        else:
            self._conn.commit()
            self._cursor.close()
            self.conn.putConnection(self._conn)

```

Resultados

Menú principal

```
> py app.py
Connection succesful
Menu
[1] Libros
[2] Autores
[3] Generos
[4] Salir
Opcion: 3
```

Menú Libro

```
[1] Mostrar libros
[2] Mostrar libro
[3] Agregar libro
[4] Editar libro
[5] Eliminar libro
[6] Volver al menu principal
Ingrese una opcion: 1
```

Mostrar Libros

```
[6] Volver al menu principal
Ingrese una opcion: 1
ID      Titulo      Autor  Año    Genero
1       El principito  1      1943   1
```

Agregar Libro

```
[1] Mostrar libros
[2] Mostrar libro
[3] Agregar libro
[4] Editar libro
[5] Eliminar libro
[6] Volver al menu principal
Ingrese una opcion: 3
Ingrese el titulo del libro: La Tregua
Autores disponibles [0-10]
ID      Nombre
1       Antoine
3       Mario
Ingrese el autor del libro: 3
Ingrese el año del libro: 1974
Generos disponibles [0-10]
ID      Nombre
1       Fabula
2       Drama
Ingrese el genero del libro: 1
Libro agregado
```

Editar Libro

```
[1] Mostrar libros
[2] Mostrar libro
[3] Agregar libro
[4] Editar libro
[5] Eliminar libro
[6] Volver al menu principal
Ingrese una opcion: 4
Ingrese el ID del libro: 3
Ingrese el titulo del libro (La Tregua):
Autores disponibles [0-10]
ID      Nombre
1       Antoine
3       Mario
Ingrese el id del autor del libro (3):
Ingrese el año del libro (1974):
Generos disponibles [0-10]
ID      Nombre
1       Fabula
2       Drama
Ingrese el id del genero del libro (1): 2
Libro editado
```

Eliminar libro

```
[1] Mostrar libros
[2] Mostrar libro
[3] Agregar libro
[4] Editar libro
[5] Eliminar libro
[6] Volver al menu principal
Ingrese una opcion: 5
Ingrese el ID del libro: 3
11:01:49 AM:DEBUG [libros.py]: 101 Se elimino el libro ID: 3, Título: La Tregua, Autor: 3, Año: 1974, Género: 2
Libro eliminado
```

Menú Autor

```
[1] Mostrar autores
[2] Mostrar autor
[3] Agregar autor
[4] Editar autor
[5] Eliminar autor
[6] Volver
Ingrese una opcion: 3
```

Agregar autor

```
[1] Mostrar autores
[2] Mostrar autor
[3] Agregar autor
[4] Editar autor
[5] Eliminar autor
[6] Volver
Ingrese una opcion: 3
Ingrese el nombre del autor: Mario
Ingrese el apellido del autor: Benedetti
Ingrese la nacionalidad del autor: Uruguay
Autor agregado
```

Editar Autor

```
[1] Mostrar autores
[2] Mostrar autor
[3] Agregar autor
[4] Editar autor
[5] Eliminar autor
[6] Volver
Ingrese una opcion: 4
Ingrese el ID del autor: 3
Ingrese el nombre del autor (Mario):
Ingrese el apellido del autor (Benedetti):
Ingrese la nacionalidad del autor (Uruguay): Argentina
Autor editado
```

Eliminar autor

```
[1] Mostrar autores
[2] Mostrar autor
[3] Agregar autor
[4] Editar autor
[5] Eliminar autor
[6] Volver
Ingrese una opcion: 5
Ingrese el ID del autor: 3
11:02:15 AM:DEBUG [autores.py]: 66 Se elimino el autor ID: 3, Nombre: Mario, Apellido: Benedetti, Nacionalidad: Argentina
Autor eliminado
```

Menú Genero

```
[1] Mostrar generos
[2] Mostrar genero
[3] Agregar genero
[4] Editar genero
[5] Eliminar genero
[6] Volver al menu principal
Ingrese una opcion: 1
```

Agregar Genero

```
[1] Mostrar generos
[2] Mostrar genero
[3] Agregar genero
[4] Editar genero
[5] Eliminar genero
[6] Volver al menu principal
Ingrese una opcion: 3
Ingrese el nombre del genero: Drana
Genero agregado
```

Editar genero

```
[1] Mostrar generos
[2] Mostrar genero
[3] Agregar genero
[4] Editar genero
[5] Eliminar genero
[6] Volver al menu principal
Ingrese una opcion: 4
Ingrese el ID del genero: 2
Ingrese el nombre del genero (Drana): Drama
Genero editado
```

Eliminar Genero

```
[1] Mostrar generos
[2] Mostrar genero
[3] Agregar genero
[4] Editar genero
[5] Eliminar genero
[6] Volver al menu principal
Ingrese una opcion: 5
Ingrese el ID del genero: 2
11:02:30 AM:DEBUG [generos.py]: 57 Se eliminó el genero ID: 2, Nombre: Drama
Genero eliminado
```


Practica #2

Realizar una aplicación utilizando el Framework DJANGO y que contenga lo siguiente

Instalación del entorno virtual.

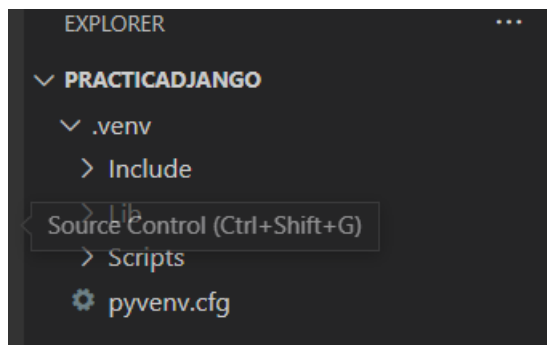
```
Microsoft Windows [Version 10.0.22000.1098]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Aduas\OneDrive\Desktop\PracticaDjango>py -3 -m venv .venv

C:\Users\Aduas\OneDrive\Desktop\PracticaDjango>.venv\scripts\activate

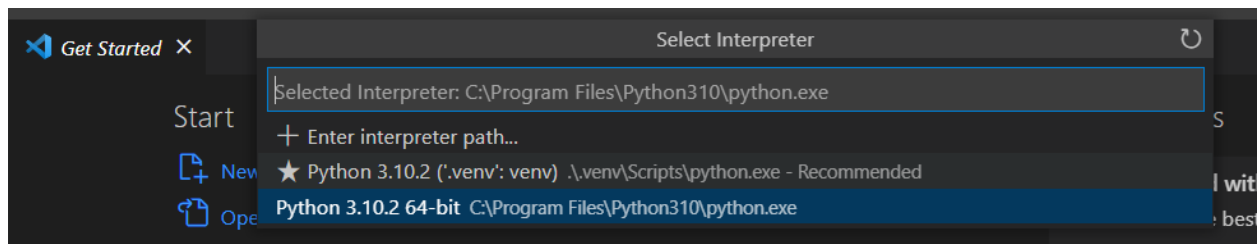
(.venv) C:\Users\Aduas\OneDrive\Desktop\PracticaDjango>
(.venv) C:\Users\Aduas\OneDrive\Desktop\PracticaDjango>
```

Una vez que se instala, se activa el .venv y nos aparecerá una carpeta.



Posterior a ello, vamos a seleccionar el interpretador, el cual será:

Ctrl + shft + p



Una vez que activamos el interpretador, ejecutaremos primero el siguiente comando:

```
python -m pip install --upgrade pip
```

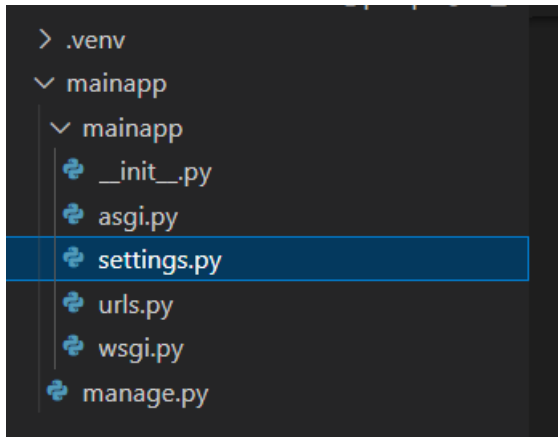
Posterior a ello, se ejecuta el comando para instalar django en nuestro entorno.

```
python -m pip install django
```

Para los siguientes pasos, vamos a crear la carpeta principal mainapp (la cual es

la que se encargara de ejecutar todo entre nuestro entorno virtual y la base de datos).

Una vez que se cree nuestra carpeta de mainapp



1-. Conexión a base de datos postgresql

Vamos a settings a configurar nuestra base de datos.

En postgres creamos la base de datos: transportebd

En la carpeta de settings, pasamos a cambiar el apartado de la conexión a la base de datos.

```
72
73 # Database
74 # https://docs.djangoproject.com/en/4.1/ref/settings/#databases
75
76 DATABASES = {
77     'default': {
78         'ENGINE': 'django.db.backends.postgresql_psycopg2',
79         'NAME': 'transportebd',
80         'USER': 'postgres',
81         'PASSWORD': 'admin',
82         'HOST': 'localhost',
83         'PORT': '5432'
84     }
85 }
86
```

Una vez realizado esto pasaremos a realizar la migración.

Antes de esto, tenemos que instalar psycopg2.

```
(.venv) C:\Users\Aduas\OneDrive\Desktop\PracticaDjango\mainapp>pip install psycopg2
Collecting psycopg2
  Using cached psycopg2-2.9.4-cp310-cp310-win_amd64.whl (1.2 MB)
Installing collected packages: psycopg2
Successfully installed psycopg2-2.9.4
```

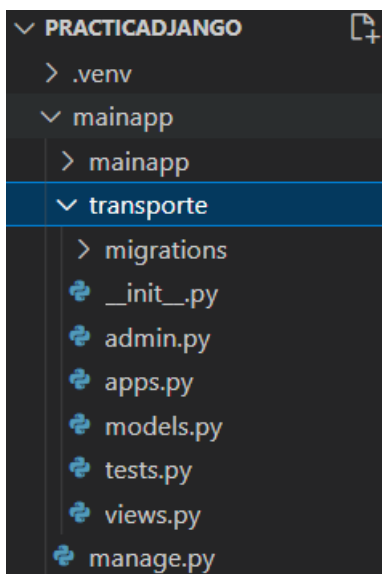
2. Utilizar al menos 4 entidades

Viajes	Rutas	Chofer	Ciudad
idViaje	idRuta	idChofer	idCiudad
idRuta	nombreRuta	nombreChofer	nombreciudad
idChofer	distanciaRuta	domicilioChofer	
fechaViaje		Edad	
observaciones		idCiudadNacimiento	

Una vez que tenemos diseñadas las tablas a utilizar, procedemos a crear dentro de mainapp, nuestra app para esta bd.

```
Applying sessions.0001_initial... OK
(.venv) C:\Users\Aduas\OneDrive\Desktop\PracticaDjango\mainapp>python manage.py startapp transporte
```

Y nos crea una carpeta, en la cual tenemos actualmente las condiciones necesarias para poder realizar los modelos de nuestras entidades.



Nos vamos al apartado de models, y crearemos los modelos.

En este caso se creará de manera inversa, debido a que Python es un lenguaje que va de manera secuencial.

3. Una de las entidades debe relacionarse con otra

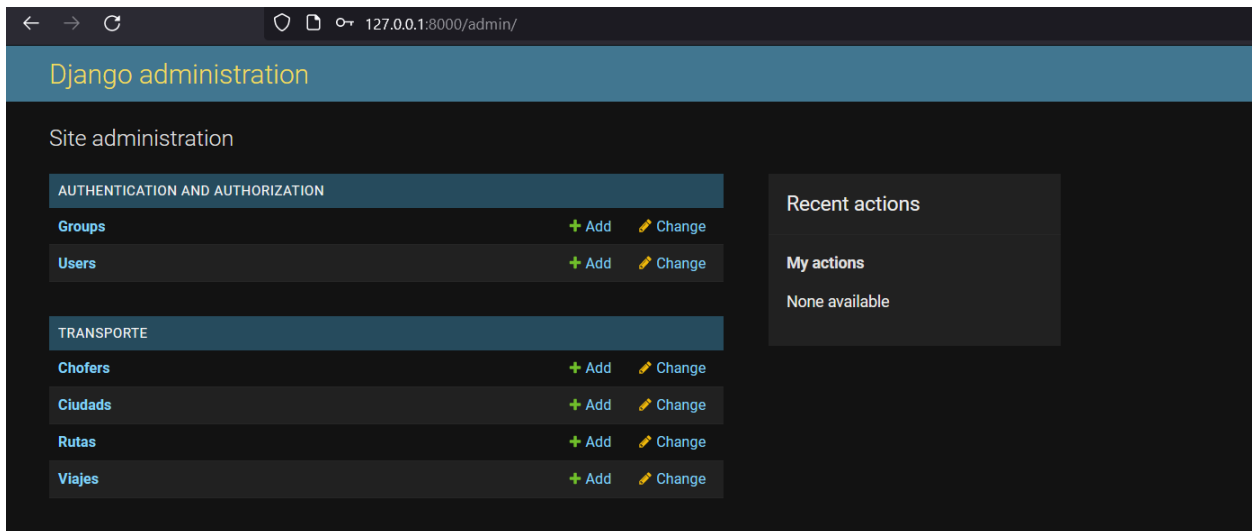
```
settings.py  models.py  admin.py

mainapp > transporte > models.py > ...
1  from django.db import models
2
3  # Create your models here.
4  class Ciudad(models.Model):
5      nombreCiudad = models.CharField(max_length=255)
6
7      def __str__(self) -> str:
8          return f'Ciudad {self.id}: {self.nombreCiudad}'
9
10 class Chofer(models.Model):
11     nombreChofer = models.CharField(max_length=255)
12     domicilioChofer = models.CharField(max_length=255)
13     edadChofer = models.IntegerField()
14     ciudadNacimiento = models.ForeignKey(Ciudad,on_delete=models.SET_NULL,null=True)
15     def __str__(self) -> str:
16         return f'Chofer {self.id}: {self.nombreChofer}'
17
18 class Ruta(models.Model):
19     nombreRuta = models.CharField(max_length=255)
20     distanciaRuta = models.DecimalField(max_digits=20, decimal_places=2)
21     def __str__(self) -> str:
22         return f'Ruta {self.id}: {self.nombreRuta}'
23
24 class Viaje(models.Model):
25     ruta = models.ForeignKey(Ruta,on_delete=models.SET_NULL,null=True)
26     chofer = models.ForeignKey(Chofer,on_delete=models.SET_NULL,null=True)
27     fechaViaje = models.DateField()
28     observaciones = models.CharField(max_length=255)
29     def __str__(self) -> str:
30         return f'Viaje {self.id}: {self.ruta}, chofer {self.chofer}, fechaViaje {self.fechaViaje}'
31
```

4. Realizar migraciones

Se realiza la migración de las tablas de los modelos anteriores.

```
(.venv) C:\Users\Aduas\OneDrive\Desktop\PracticaDjango\mainapp>python manage.py makemigrations
Migrations for 'transporte':
  transporte\migrations\0001_initial.py
    - Create model Chofer
    - Create model Ciudad
    - Create model Ruta
    - Create model Viaje
    - Add field ciudadNacimiento to chofer
```



5. CRUD de las 4 entidades utilizando templates

Viaje.

Id Viaje	Ruta	chofer	fechaViaje	observaciones	Opciones		
1	Ruta 2: Laredo - Monterrey	Chofer 3: Francisco Luis	Oct. 18, 2022	Viaje de Nuevo Laredo a Monterrey.	Editar	Eliminar	Consulta
2	Ruta 5: Silao - Laredo Tx	Chofer 3: Francisco Luis	Dec. 12, 1999	test	Editar	Eliminar	Consulta
4	Ruta 1: Laredo - San Antonio	Chofer 2: Juan Alberto	Jan. 11, 2022	Viaje de juan	Editar	Eliminar	Consulta

[Agregar](#) [Menu Principal](#)

Viaje Agregar

Agregar Viaje

Ruta:
Chofer:
FechaViaje:
Observaciones:

[Menu Principal](#)

Viaje Editar

Editar Viaje

Ruta:
Chofer:
FechaViaje:
Observaciones:

[Menu Principal](#)

Viaje Consulta.

1 Ruta 2: Laredo - Monterrey Chofer 3: Francisco Luis Oct. 18, 2022 Viaje de Nuevo Laredo a Monterrey.

[Menu Principal](#)

Ruta

Ruta Agregar

Agregar Ruta

NombreRuta:

DistanciaRuta:

[Menu Principal](#)

Ruta Editar

Editar Ruta

NombreRuta:

DistanciaRuta:

[Menu Principal](#)

Ruta Consulta

1 Laredo - San Antonio 200.50

[Menu Principal](#)

Chofer

Chofer Agregar

Agregar Chofer

NombreChofer:

DomicilioChofer:

EdadChofer:

CiudadNacimiento:

[Menu Principal](#)

Ciudad 2: Monterrey
Ciudad 3: Saltillo
Ciudad 4: Reynosa1
Ciudad 5: Tijuana

Chofer Editar

Editar chofer

NombreChofer:

DomicilioChofer:

EdadChofer:

CiudadNacimiento:

[Menu Principal](#)

Chofer consulta

2 Juan Alberto

[Menu Principal](#)

Ciudad

Ciudad Agregar

Agregar Ciudad

NombreCiudad:

[Menu Principal](#)

Ciudad Editar

Editar Ciudad

NombreCiudad:

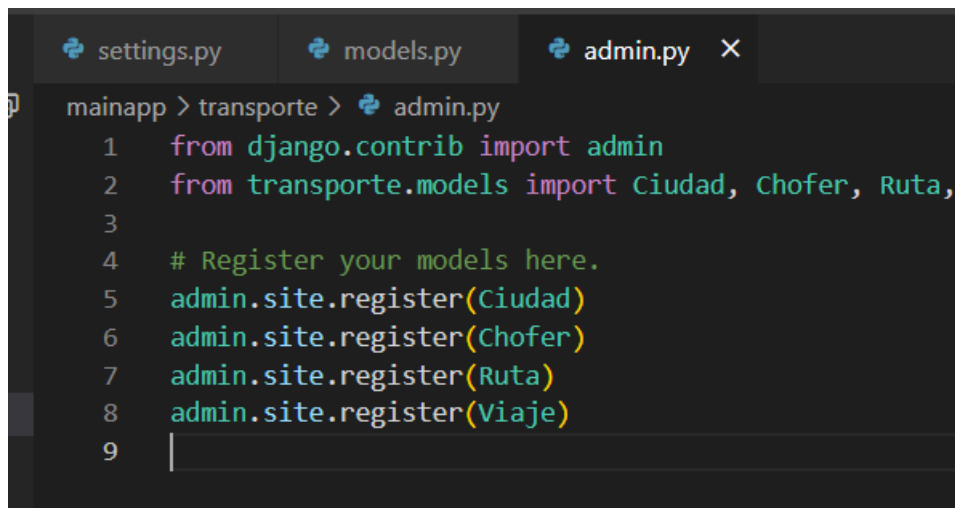
[Menu Principal](#)

2 Monterrey

[Menu Principal](#)

6. Agregar las 4 entidades a la página de administración

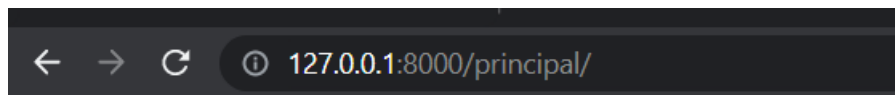
Se agregan las páginas a admin.



```
mainapp > transporte > admin.py
1  from django.contrib import admin
2  from transporte.models import Ciudad, Chofer, Ruta,
3
4  # Register your models here.
5  admin.site.register(Ciudad)
6  admin.site.register(Chofer)
7  admin.site.register(Ruta)
8  admin.site.register(Viaje)
9  |
```

Resultados.

Se crea menú principal.



Menu principal

[Viajes](#) [Rutas](#) [Chofer](#) [Ciudad](#)

Cada uno de ellos nos envía a las entidades correspondientes. En cada uno se puede agregar, eliminar, editar y consultar.

Si nos dirigimos al de viaje:

127.0.0.1:8000/viaje/					
Id Viaje	Ruta	chofer	fechaViaje	observaciones	Opciones
1	Ruta 2: Laredo - Monterrey	Chofer 3: Francisco Luis	Oct. 18, 2022	Viaje de Nuevo Laredo a Monterrey.	Editar Eliminar Consulta
2	Ruta 4: Mty - Leon	Chofer 3: Francisco Luis	Dec. 12, 1999	test	Editar Eliminar Consulta

[Agregar](#) [Menu Principal](#)

Se ve los viajes agregados, y la información que contiene.

Si nosotros damos clic en Agregar, nos mandara a su formulario:

127.0.0.1:8000/viaje/agregar_viaje

Agregar Viaje

Ruta:

Chofer:

FechaViaje:

Observaciones:

[Menu Principal](#)

Este form contiene las relaciones con Ruta, y Chofer, por ende, las otras entidades actualmente tienen la funcionalidad de Agregar, Editar, Eliminar y consultar.

Agregar Viaje

Ruta:

Chofer:

FechaViaje:

Observaciones:

[Menu Principal](#)

Al momento darle agregar, nos mandara a la pantalla principal de viajes.

Id Viaje	Ruta	chofer	fechaViaje	observaciones	Opciones		
1	Ruta 2: Laredo - Monterrey	Chofer 3: Francisco Luis	Oct. 18, 2022	Viaje de Nuevo Laredo a Monterrey.	Editar	Eliminar	Consulta
2	Ruta 5: Silao - Laredo Tx	Chofer 3: Francisco Luis	Dec. 12, 1999	test	Editar	Eliminar	Consulta
4	Ruta 1: Laredo - San Antonio	Chofer 2: Juan Alberto	Jan. 11, 2022	Viaje de juan	Editar	Eliminar	Consulta
5	Ruta 1: Laredo - San Antonio	Chofer 2: Juan Alberto	Feb. 11, 2022	Viaje de prueba	Editar	Eliminar	Consulta

[Agregar](#) [Menu Principal](#)

Y se actualiza al instante.

Si nosotros damos clic en el botón de editar.

Editar Viaje

Ruta:

Chofer:

FechaViaje:

Observaciones:

[Menu Principal](#)

De igual manera, al momento de darle en editar, se edita al instante.

5	Ruta 1: Laredo - San Antonio	Chofer 2: Juan Alberto	Feb. 11, 2022	Viaje de pruebas juan alberto	Editar	Eliminar	Consulta
---	------------------------------	------------------------	---------------	-------------------------------	------------------------	--------------------------	--------------------------

[Agregar](#) [Menu Principal](#)

Consulta:

←
→
↺
🔒 127.0.0.1:8000/viaje/detalle_viaje/5
📄
🔗

5 Ruta 1: Laredo - San Antonio Chofer 2: Juan Alberto Feb. 11, 2022 Viaje de pruebas juan alberto

[Menu Principal](#)

Y si le damos en eliminar, solo nos aparecerán los viajes actuales.

Id Viaje	Ruta	chofer	fechaViaje	observaciones	Opciones		
1	Ruta 2: Laredo - Monterrey	Chofer 3: Francisco Luis	Oct. 18, 2022	Viaje de Nuevo Laredo a Monterrey.	Editar	Eliminar	Consulta
2	Ruta 5: Silao - Laredo Tx	Chofer 3: Francisco Luis	Dec. 12, 1999	test	Editar	Eliminar	Consulta
4	Ruta 1: Laredo - San Antonio	Chofer 2: Juan Alberto	Jan. 11, 2022	Viaje de juan	Editar	Eliminar	Consulta

[Agregar](#) [Menu Principal](#)

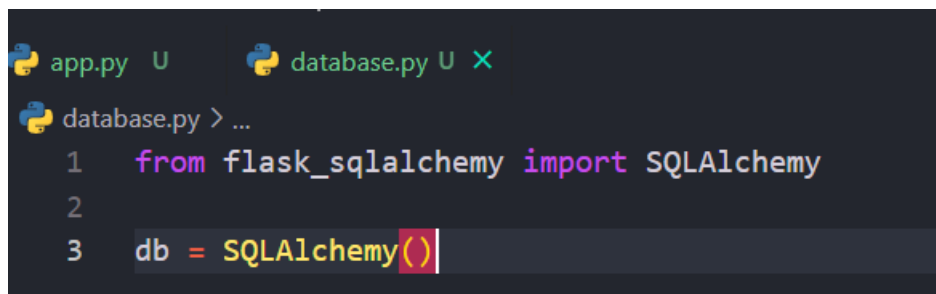
Practica #3 Realizar una aplicación utilizando Framework Flask

5. Conexión a base de datos postgresql con SQLAlchemy
6. Al menos un formulario con estilo css
7. Utilizar app logging
8. Utilizar al menos 3 entidades
9. CRUD con pantallas de 2 entidades
10. CRUD de al menos una entidad utilizando peticiones HTTP
11. Utilizar migraciones

Comprobación

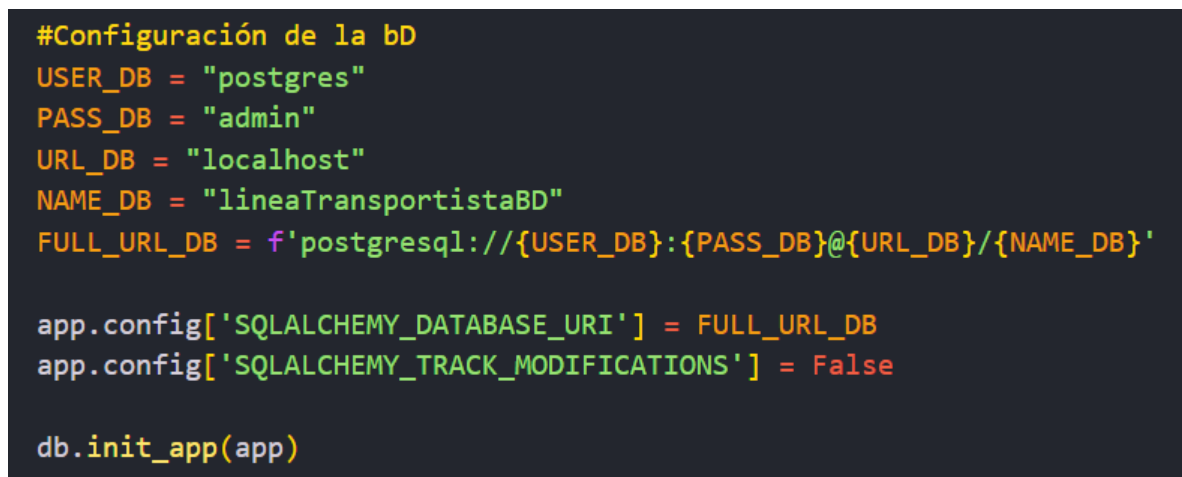
1. Conexión a BD postgresql con SQLAlchemy

Conexión a la BD

A screenshot of a code editor with a dark background. At the top, there are two tabs: 'app.py' and 'database.py'. The 'database.py' tab is active. Below the tabs, the code in 'database.py' is shown. Line 1: 'from flask_sqlalchemy import SQLAlchemy'. Line 2: an empty line. Line 3: 'db = SQLAlchemy()' with the cursor at the end of the line.

```
app.py U database.py U X
database.py > ...
1 from flask_sqlalchemy import SQLAlchemy
2
3 db = SQLAlchemy()
```

Configuración para la conexión a la BD

A screenshot of a code editor with a dark background showing the configuration for a PostgreSQL database connection. The code is as follows:

```
#Configuración de la bD
USER_DB = "postgres"
PASS_DB = "admin"
URL_DB = "localhost"
NAME_DB = "lineaTransportistaBD"
FULL_URL_DB = f'postgresql://{USER_DB}:{PASS_DB}@{URL_DB}/{NAME_DB}'

app.config['SQLALCHEMY_DATABASE_URI'] = FULL_URL_DB
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

db.init_app(app)
```

2. Formulario con CSS

Se creo un archivo css llamado “formulario” en el cual se definieron los estilos para todos los formularios que se utilizaron para la práctica.



Chofer

Nombre

Apellido

Telefono

Licencia

Enviar

```
templates > choferes > agregarChofer.html > html > body > div.container > form.formulario > p.field.field_v3
4 <meta charset="UTF-8" />
5 <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6 <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7 <link
8   rel="stylesheet"
9   href="{url_for('static', filename='styles/formulario.css')}"
10 />
11 <title>Agregar Chofer</title>
12 </head>
13 <body>
14 <div class="container">
15 <h3 align="center">Chofer</h3>
16 <form action="/choferes/agregar" method="POST" class="formulario">
17   {{forma.csrf_token}}
18   <p class="field field_v3">
19     {{forma.nombre.label(class="has-screen-reader")}}<br />
20     {{forma.nombre(size=30,class="field_input")}}
21     <span class="field_label-wrap" aria-hidden="true">
22       {{forma.nombre.label(class="field_label")}}<br />
23     </span>
24   </p>
25   <p class="field field_v3">
26     {{forma.apellido.label(class="has-screen-reader")}}<br />
27     {{forma.apellido(size=30,class="field_input")}}
28     <span class="field_label-wrap" aria-hidden="true">
29       {{forma.apellido.label(class="field_label")}}<br />
30     </span>
31   </p>
32   <p class="field field_v3">
33     {{forma.telefono.label(class="has-screen-reader")}}<br />
34     {{forma.telefono(size=30,class="field_input")}}
35     <span class="field_label-wrap" aria-hidden="true">
```

```
1 container {
2   display: flex;
3   flex-direction: column;
4   width: fit-content;
5   margin-top: 2rem;
6   margin-left: auto;
7   margin-right: auto;
8   padding: 1rem;
9   box-shadow: 0px 100px 100px 0px #7d2fed;
10  border-radius: 1rem;
11 }
12 .formulario {
13   display: flex;
14   flex-direction: column;
15 }
16 input[type="submit"] {
17   appearance: none;
18   backface-visibility: hidden;
19   background-color: #7d2fed;
20   border-radius: 10px;
21   border-style: none;
22   box-shadow: none;
23   box-sizing: border-box;
24   color: #fff;
25   cursor: pointer;
26   display: inline-block;
27   font-family: Inter, -apple-system, system-ui, "Segoe UI", Helvetica, Arial, sans-serif;
28   font-size: 15px;
29   font-weight: 500;
30   height: 50px;
31   letter-spacing: normal;
```

3. App Logging

```
app.py > ...
1 from flask import Flask,request,url_for,render_template,redirect
2 from database import db
3 from flask_migrate import Migrate
4 from forms import ChoferForm, RemolqueForm, TractorForm
5 from models import Chofer, Remolque, Tractor
6 import logging
7
8 app = Flask(__name__)
9
10 logging.basicConfig(filename='error.log',level=logging.DEBUG)
```

Importación y configuración del logger, se le dio el nombre de “error” al archivo de log.

```
#Chofer
@app.route("/choferes")
def choferes():
    try:
        choferes = Chofer.query.all()
        app.logger.info("Se obtuvieron los choferes")
        return render_template("/choferes/choferes.html",choferes=choferes)
    except Exception as e:
        app.logger.error(e)
        return render_template("error.html")
```

Se agregaron en cada endpoints, para registrar las acciones en la aplicación y los errores que pudiesen producirse.

```
510 INFO:app:Se obtuvieron los choferes
511 INFO:werkzeug:127.0.0.1 - - [29/Oct/2022 22:03:40] "GET /choferes HTTP/1.1" 200 -
512 INFO:werkzeug:127.0.0.1 - - [29/Oct/2022 22:03:41] "[36mGET /static/styles/listado.css HTTP/1.1[0m" 304 -
513 INFO:werkzeug:127.0.0.1 - - [29/Oct/2022 22:03:43] "GET /choferes/agregar HTTP/1.1" 200 -
514 INFO:werkzeug:127.0.0.1 - - [29/Oct/2022 22:03:43] "[36mGET /static/styles/formulario.css HTTP/1.1[0m" 304 -
515
```

4. Entidades

Se utilizaron 3 entidades, Chofer, Tractor y Remolques, para cada una se creó el Modelo correspondiente.

```
app.py U  models.py U X  database.py U
models.py > Chofer
3 class Chofer(db.Model):
4     id = db.Column(db.Integer, primary_key = True)
5     nombre = db.Column(db.String(250))
6     apellido = db.Column(db.String(250))
7     telefono = db.Column(db.String(250))
8     licencia = db.Column(db.String(250))
9
10    def __str__(self) -> str:
11        return (f'ID: {self.id}, '
12                f'Nombre: {self.nombre}, '
13                f'Apellido: {self.apellido}, '
14                f'telefono: {self.telefono}, '
15                f'licencia: {self.licencia}')
16
17 class Tractor(db.Model):
18     id=db.Column(db.Integer,primary_key=True)
19     numero=db.Column(db.String(250))
20     placas=db.Column(db.String(250))
21     marca=db.Column(db.String(250))
22     modelo=db.Column(db.String(250))
23
24    def __str__(self) -> str:
25        return (f'ID: {self.id}, '
26                f'Numero: {self.numero}, '
27                f'Placas: {self.placas}, '
28                f'Marca: {self.marca}, '
29                f'Modelo: {self.modelo}')
30
```

```
class Remolque(db.Model):
    id=db.Column(db.Integer,primary_key=True)
    numero=db.Column(db.String(250))
    placas=db.Column(db.String(250))
    tipo=db.Column(db.String(250))

    def __str__(self) -> str:
        return (f'ID: {self.id}, '
                f'Numero: {self.numero}, '
                f'Placas: {self.placas}, '
                f'Tipo: {self.tipo}')
```

5. CRUD de entidades

Para las 3 entidades se creo una pantalla de listado, además de las pantallas para agregar y editar registros de cada una de las entidades.

Pantallas de la entidad Chofer:

Choferes

Index		Agregar		
Nombre	Apellido	Telefono	Licencia	-
Antonio	Rivera	8672811605	12381283281382	<div>EditarEliminar</div>

Chofer

Nombre

Apellido

Telefono

Licencia

Enviar

Se utilizo el mismo diseño para las pantallas para agregar y editar un chofer

Pantallas de la entidad Tractor

Tractores				
Index		Agregar		
Número	Placas	Marca	Modelo	-
32	33	4	5	<div> <div>Editar</div> <div>Eliminar</div> </div>

Agregar Tractor

Numero

Placas

Marca

Modelo

Enviar

Se utilizo el mismo diseño para la pantalla de agregar y editar un tractor.

Pantallas de la entidad Remolque

Remolques			
Index		Agregar	
Número	Placas	Tipo	
12	123	123	Editar Eliminar

Agregar Remolque

Numero

Placas

Tipo

Enviar

Se utilizo el mismo diseño para las pantallas para agregar y editar un remolque.

6. CRUD utilizando peticiones HTTP

Para todas las entidades se utilizaron peticiones HTTP para obtener el listado de cada una de las entidades, además de agregar, editar y eliminar registros de cada entidad

Ejemplo de las peticiones HTTP definidas para la entidad Chofer.

```
#Chofer
@app.route("/choferes")
def choferes():
    try:
        choferes = Chofer.query.all()
        app.logger.info("Se obtuvieron los choferes")
        return render_template("/choferes/choferes.html", choferes=choferes)
    except Exception as e:
        app.logger.error(e)
        return render_template("error.html")

@app.route("/choferes/agregar", methods=['GET', 'POST'])
def agregarChofer():
    try:
        chofer = Chofer()
        choferForm = ChoferForm(obj=chofer)
        if request.method == 'POST':
            if choferForm.validate_on_submit():
                choferForm.populate_obj(chofer)
                db.session.add(chofer)
                db.session.commit()
                logging.info("Se agregó un chofer")
                return redirect(url_for('choferes'))
            return render_template("/choferes/agregarChofer.html", form=choferForm)
        except Exception as e:
            logging.error(e)
            return render_template("error.html")

@app.route("/choferes/editar/<int:id>", methods=['GET', 'POST'])
def editarChofer(id):
    try:
        chofer = Chofer.query.get_or_404(id)

        choferForm = ChoferForm(obj=chofer)
        if request.method == 'POST':
            if choferForm.validate_on_submit():
                choferForm.populate_obj(chofer)
                #update
                db.session.commit()
                logging.info("Se editó un chofer")
                return redirect(url_for('choferes'))
            return render_template("/choferes/editarChofer.html", form=choferForm)
        except Exception as e:
            logging.error(e)
            return render_template("error.html")

@app.route("/choferes/eliminar/<int:id>")
def eliminarChofer(id):
    try:
        chofer = Chofer.query.get_or_404(id)
        db.session.delete(chofer)
        db.session.commit()
        return redirect(url_for('choferes'))
    except Exception as e:
        logging.error(e)
        return render_template("error.html")
```

7. Migraciones

Se hicieron las migraciones y esto se demuestra ya que se generó la carpeta migrations con los modelos correspondientes que antes habían sido definidos.

```
EXPLORER
... app.py U 379f7ef1f830_py U X database.py U
OPEN EDITORS migrations > versions > 379f7ef1f830_py > downgrade
app.py U 18
379f7ef1f830_py... U 19
database.py U 20
3-FLASK
  __pycache__ 21
  migrations 22
  versions 23
  __pycache__ 24
  379f7ef1f830_py U 25
  alembic.ini U 26
  env.py U 27
  README U 28
  script.py.mako U 29
  static/styles 30
  templates 31
  venv 32
  app.py U 33
  database.py U 34
  error.log U 35
  forms.py U 36
  models.py U 37
def upgrade():
    38
    39
    40
    41
    42
    43
    44
    45
    46
    47
    48
    49
    50
    51
    52
    53
    54
    55
    56
    57
    58
    59
    60
    61
    62
    63
    64
    65
    66
    67
    68
    69
    70
    71
    72
    73
    74
    75
    76
    77
    78
    79
    80
    81
    82
    83
    84
    85
    86
    87
    88
    89
    90
    91
    92
    93
    94
    95
    96
    97
    98
    99
    100
    101
    102
    103
    104
    105
    106
    107
    108
    109
    110
    111
    112
    113
    114
    115
    116
    117
    118
    119
    120
    121
    122
    123
    124
    125
    126
    127
    128
    129
    130
    131
    132
    133
    134
    135
    136
    137
    138
    139
    140
    141
    142
    143
    144
    145
    146
    147
    148
    149
    150
    151
    152
    153
    154
    155
    156
    157
    158
    159
    160
    161
    162
    163
    164
    165
    166
    167
    168
    169
    170
    171
    172
    173
    174
    175
    176
    177
    178
    179
    180
    181
    182
    183
    184
    185
    186
    187
    188
    189
    190
    191
    192
    193
    194
    195
    196
    197
    198
    199
    200
    201
    202
    203
    204
    205
    206
    207
    208
    209
    210
    211
    212
    213
    214
    215
    216
    217
    218
    219
    220
    221
    222
    223
    224
    225
    226
    227
    228
    229
    230
    231
    232
    233
    234
    235
    236
    237
    238
    239
    240
    241
    242
    243
    244
    245
    246
    247
    248
    249
    250
    251
    252
    253
    254
    255
    256
    257
    258
    259
    260
    261
    262
    263
    264
    265
    266
    267
    268
    269
    270
    271
    272
    273
    274
    275
    276
    277
    278
    279
    280
    281
    282
    283
    284
    285
    286
    287
    288
    289
    290
    291
    292
    293
    294
    295
    296
    297
    298
    299
    300
    301
    302
    303
    304
    305
    306
    307
    308
    309
    310
    311
    312
    313
    314
    315
    316
    317
    318
    319
    320
    321
    322
    323
    324
    325
    326
    327
    328
    329
    330
    331
    332
    333
    334
    335
    336
    337
    338
    339
    340
    341
    342
    343
    344
    345
    346
    347
    348
    349
    350
    351
    352
    353
    354
    355
    356
    357
    358
    359
    360
    361
    362
    363
    364
    365
    366
    367
    368
    369
    370
    371
    372
    373
    374
    375
    376
    377
    378
    379
    380
    381
    382
    383
    384
    385
    386
    387
    388
    389
    390
    391
    392
    393
    394
    395
    396
    397
    398
    399
    400
    401
    402
    403
    404
    405
    406
    407
    408
    409
    410
    411
    412
    413
    414
    415
    416
    417
    418
    419
    420
    421
    422
    423
    424
    425
    426
    427
    428
    429
    430
    431
    432
    433
    434
    435
    436
    437
    438
    439
    440
    441
    442
    443
    444
    445
    446
    447
    448
    449
    450
    451
    452
    453
    454
    455
    456
    457
    458
    459
    460
    461
    462
    463
    464
    465
    466
    467
    468
    469
    470
    471
    472
    473
    474
    475
    476
    477
    478
    479
    480
    481
    482
    483
    484
    485
    486
    487
    488
    489
    490
    491
    492
    493
    494
    495
    496
    497
    498
    499
    500
    501
    502
    503
    504
    505
    506
    507
    508
    509
    510
    511
    512
    513
    514
    515
    516
    517
    518
    519
    520
    521
    522
    523
    524
    525
    526
    527
    528
    529
    530
    531
    532
    533
    534
    535
    536
    537
    538
    539
    540
    541
    542
    543
    544
    545
    546
    547
    548
    549
    550
    551
    552
    553
    554
    555
    556
    557
    558
    559
    560
    561
    562
    563
    564
    565
    566
    567
    568
    569
    570
    571
    572
    573
    574
    575
    576
    577
    578
    579
    580
    581
    582
    583
    584
    585
    586
    587
    588
    589
    590
    591
    592
    593
    594
    595
    596
    597
    598
    599
    600
    601
    602
    603
    604
    605
    606
    607
    608
    609
    610
    611
    612
    613
    614
    615
    616
    617
    618
    619
    620
    621
    622
    623
    624
    625
    626
    627
    628
    629
    630
    631
    632
    633
    634
    635
    636
    637
    638
    639
    640
    641
    642
    643
    644
    645
    646
    647
    648
    649
    650
    651
    652
    653
    654
    655
    656
    657
    658
    659
    660
    661
    662
    663
    664
    665
    666
    667
    668
    669
    670
    671
    672
    673
    674
    675
    676
    677
    678
    679
    680
    681
    682
    683
    684
    685
    686
    687
    688
    689
    690
    691
    692
    693
    694
    695
    696
    697
    698
    699
    700
    701
    702
    703
    704
    705
    706
    707
    708
    709
    710
    711
    712
    713
    714
    715
    716
    717
    718
    719
    720
    721
    722
    723
    724
    725
    726
    727
    728
    729
    730
    731
    732
    733
    734
    735
    736
    737
    738
    739
    740
    741
    742
    743
    744
    745
    746
    747
    748
    749
    750
    751
    752
    753
    754
    755
    756
    757
    758
    759
    760
    761
    762
    763
    764
    765
    766
    767
    768
    769
    770
    771
    772
    773
    774
    775
    776
    777
    778
    779
    780
    781
    782
    783
    784
    785
    786
    787
    788
    789
    790
    791
    792
    793
    794
    795
    796
    797
    798
    799
    800
    801
    802
    803
    804
    805
    806
    807
    808
    809
    810
    811
    812
    813
    814
    815
    816
    817
    818
    819
    820
    821
    822
    823
    824
    825
    826
    827
    828
    829
    830
    831
    832
    833
    834
    835
    836
    837
    838
    839
    840
    841
    842
    843
    844
    845
    846
    847
    848
    849
    850
    851
    852
    853
    854
    855
    856
    857
    858
    859
    860
    861
    862
    863
    864
    865
    866
    867
    868
    869
    870
    871
    872
    873
    874
    875
    876
    877
    878
    879
    880
    881
    882
    883
    884
    885
    886
    887
    888
    889
    890
    891
    892
    893
    894
    895
    896
    897
    898
    899
    900
    901
    902
    903
    904
    905
    906
    907
    908
    909
    910
    911
    912
    913
    914
    915
    916
    917
    918
    919
    920
    921
    922
    923
    924
    925
    926
    927
    928
    929
    930
    931
    932
    933
    934
    935
    936
    937
    938
    939
    940
    941
    942
    943
    944
    945
    946
    947
    948
    949
    950
    951
    952
    953
    954
    955
    956
    957
    958
    959
    960
    961
    962
    963
    964
    965
    966
    967
    968
    969
    970
    971
    972
    973
    974
    975
    976
    977
    978
    979
    980
    981
    982
    983
    984
    985
    986
    987
    988
    989
    990
    991
    992
    993
    994
    995
    996
    997
    998
    999
    1000
    1001
    1002
    1003
    1004
    1005
    1006
    1007
    1008
    1009
    1010
    1011
    1012
    1013
    1014
    1015
    1016
    1017
    1018
    1019
    1020
    1021
    1022
    1023
    1024
    1025
    1026
    1027
    1028
    1029
    1030
    1031
    1032
    1033
    1034
    1035
    1036
    1037
    1038
    1039
    1040
    1041
    1042
    1043
    1044
    1045
    1046
    1047
    1048
    1049
    1050
    1051
    1052
    1053
    1054
    1055
    1056
    1057
    1058
    1059
    1060
    1061
    1062
    1063
    1064
    1065
    1066
    1067
    1068
    1069
    1070
    1071
    1072
    1073
    1074
    1075
    1076
    1077
    1078
    1079
    1080
    1081
    1082
    1083
    1084
    1085
    1086
    1087
    1088
    1089
    1090
    1091
    1092
    1093
    1094
    1095
    1096
    1097
    1098
    1099
    1100
    1101
    1102
    1103
    1104
    1105
    1106
    1107
    1108
    1109
    1110
    1111
    1112
    1113
    1114
    1115
    1116
    1117
    1118
    1119
    1120
    1121
    1122
    1123
    1124
    1125
    1126
    1127
    1128
    1129
    1130
    1131
    1132
    1133
    1134
    1135
    1136
    1137
    1138
    1139
    1140
    1141
    1142
    1143
    1144
    1145
    1146
    1147
    1148
    1149
    1150
    1151
    1152
    1153
    1154
    1155
    1156
    1157
    1158
    1159
    1160
    1161
    1162
    1163
    1164
    1165
    1166
    1167
    1168
    1169
    1170
    1171
    1172
    1173
    1174
    1175
    1176
    1177
    1178
    1179
    1180
    1181
    1182
    1183
    1184
    1185
    1186
    1187
    1188
    1189
    1190
    1191
    1192
    1193
    1194
    1195
    1196
    1197
    1198
    1199
    1200
    1201
    1202
    1203
    1204
    1205
    1206
    1207
    1208
    1209
    1210
    1211
    1212
    1213
    1214
    1215
    1216
    1217
    1218
    1219
    1220
    1221
    1222
    1223
    1224
    1225
    1226
    1227
    1228
    1229
    1230
    1231
    1232
    1233
    1234
    1235
    1236
    1237
    1238
    1239
    1240
    1241
    1242
    1243
    1244
    1245
    1246
    1247
    1248
    1249
    1250
    1251
    1252
    1253
    1254
    1255
    1256
    1257
    1258
    1259
    1260
    1261
    1262
    1263
    1264
    1265
    1266
    1267
    1268
    1269
    1270
    1271
    1272
    1273
    1274
    1275
    1276
    1277
    1278
    1279
    1280
    1281
    1282
    1283
    1284
    1285
    1286
    1287
    1288
    1289
    1290
    1291
    1292
    1293
    1294
    1295
    1296
    1297
    1298
    1299
    1300
    1301
    1302
    1303
    1304
    1305
    1306
    1307
    1308
    1309
    1310
    1311
    1312
    1313
    1314
    1315
    1316
    1317
    1318
    1319
    1320
    1321
    1322
    1323
    1324
    1325
    1326
    1327
    1328
    1329
    1330
    1331
    1332
    1333
    1334
    1335
    1336
    1337
    1338
    1339
    1340
    1341
    1342
    1343
    1344
    1345
    1346
    1347
    1348
    1349
    1350
    1351
    1352
    1353
    1354
    1355
    1356
    1357
    1358
    1359
    1360
    1361
    1362
    1363
    1364
    1365
    1366
    1367
    1368
    1369
    1370
    1371
    1372
    1373
    1374
    1375
    1376
    1377
    1378
    1379
    1380
    1381
    1382
    1383
    1384
    1385
    1386
    1387
    1388
    1389
    1390
    1391
    1392
    1393
    1394
    1395
    1396
    1397
    1398
    1399
    1400
    1401
    1402
    1403
    1404
    1405
    1406
    1407
    1408
    1409
    1410
    1411
    1412
    1413
    1414
    1415
    1416
    1417
    1418
    1419
    1420
    1421
    1422
    1423
    1424
    1425
    1426
    1427
    1428
    1429
    1430
    1431
    1432
    1433
    1434
    1435
    1436
    1437
    1438
    1439
    1440
    1441
    1442
    1443
    1444
    1445
    1446
    1447
    1448
    1449
    1450
    1451
    1452
    1453
    1454
    1455
    1456
    1457
    1458
    1459
    1460
    1461
    1462
    1463
    1464
    1465
    1466
    1467
    1468
    1469
    1470
    1471
    1472
    1473
    1474
    1475
    1476
    1477
    1478
    1479
    1480
    1481
    1482
    1483
    1484
    1485
    1486
    1487
    1488
    1489
    1490
    1491
    1492
    1493
    1494
    1495
    1496
    1497
    1498
    1499
    1500
    1501
    1502
    1503
    1504
    1505
    1506
    1507
    1508
    1509
    1510
    1511
    1512
    1513
    1514
    1515
    1516
    1517
    1518
    1519
    1520
    1521
    1522
    1523
    1524
    1525
    1526
    1527
    1528
    1529
    1530
    1531
    1532
    1533
    1534
    1535
    1536
    1537
    1538
    1539
    1540
    1541
    1542
    1543
    1544
    1545
    1546
    1547
    1548
    1549
    1550
    1551
    1552
    1553
    1554
    1555
    1556
    1557
    1558
    1559
    1560
    1561
    1562
    1563
    1564
    1565
    1566
    1567
    1568
    1569
    1570
    1571
    1572
    1573
    1574
    1575
    1576
    1577
    1578
    1579
    1580
    1581
    1582
    1583
    1584
    1585
    1586
    1587
    1588
    1589
    1590
    1591
    1592
    1593
    1594
    1595
    1596
    1597
    1598
    1599
    1600
    1601
    1602
    1603
    1604
    1605
    1606
    1607
    1608
    1609
    1610
    1611
    1612
    1613
    1614
    1615
    1616
    1617
    1618
    1619
    1620
    1621
    1622
    1623
    1624
    1625
    1626
    1627
    1628
    1629
    1630
    1631
    1632
    1633
    1634
    1635
    1636
    1637
    1638
    1639
    1640
    1641
    1642
    1643
    1644
    1645
    1646
    1647
    1648
    1649
    1650
    1651
    1652
    1653
    1654
    1655
    1656
    1657
    1658
    1659
    1660
    1661
    1662
    1663
    1664
    1665
    1666
    1667
    1668
    1669
    1670
    1671
    1672
    1673
    1674
    1675
    1676
    1677
    1678
    1679
    1680
    1681
    1682
    1683
    1684
    1685
    1686
    1687
    1688
    1689
    1690
    1691
    1692
    1693
    1694
    1695
    1696
    1697
    1698
    1699
    1700
    1701
    1702
    1703
    1704
    1705
    1706
    1707
    1708
    1709
    1710
    1711
    1712
    1713
    1714
    1715
    1716
    1717
    1718
    1719
    1720
    1721
    1722
    1723
    1724
    1725
    1726
    1727
    1728
    1729
    1730
    1731
    1732
    1733
    1734
    1735
    1736
    1737
    1738
    1739
    1740
    1741
    1742
    1743
    1744
    1745
    1746
    1747
    1748
    1749
    1750
    1751
    1752
    1753
    1754
    1755
    1756
    1757
    1758
    1759
    1760
    1761
    1762
    1763
    1764
    1765
    1766
    1767
    1768
    1769
    1770
    1771
    1772
    1773
    1774
    1775
    1776
    1777
    1778
    1779
    1780
    1781
    1782
    1783
    1784
    1785
    1786
    1787
    1788
    1789
    1790
    1791
    1792
    1793
    1794
    1795
    1796
    1797
    1798
    17
```

Resultados

En la practica se hizo uso de tres entidades, para cada una de ellas se generó el modelo correspondiente, así como las pantallas, rutas y formulario correspondiente para así mostrar un listado de cada entidad, así como agregar, editar y eliminar registros. También se hizo la configuración del logger para posteriormente registrar las acciones que se daban en la aplicación y también para registrar los posibles errores. Ademas de esto se hizo la configuración de la conexión a la BD haciendo uso del ORM SQLAlchemy, posteriormente se hizo la migración de los modelos a la BD para así hacer las operaciones haciendo uso de la BD.

Ejemplo de funcionamiento de la aplicación con la entidad Chofer.

- Pantalla principal de la aplicación

Linea Transportista

Choferes

Tractores

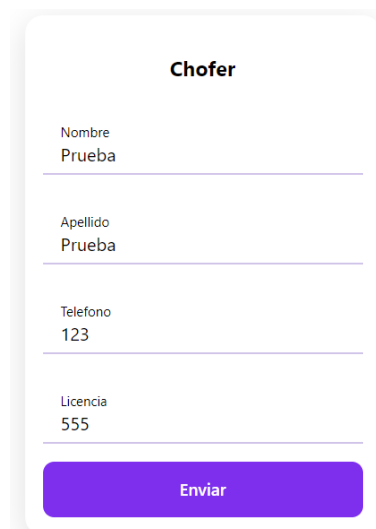
Remolques

- Listado de Choferes



Nombre	Apellido	Telefono	Licencia	
Antonio	Rivera	8672811605	12381283281382	<button>Editar</button> <button>Eliminar</button>

- Registro de un nuevo Chofer



Chofer

Nombre
Prueba

Apellido
Prueba

Telefono
123

Licencia
555

Enviar

Choferes

Index		Agregar		
Nombre	Apellido	Telefono	Licencia	-
Antonio	Rivera	8672811605	12381283281382	<button>Editar</button> <button>Eliminar</button>
Prueba	Prueba	123	555	<button>Editar</button> <button>Eliminar</button>

- Edición de un Chofer

Editar Chofer

Nombre
Prueba Editada

Apellido
Prueba

Telefono
123

Licencia
555

Enviar

Choferes

Index		Agregar		
Nombre	Apellido	Telefono	Licencia	-
Antonio	Rivera	8672811605	12381283281382	<button>Editar</button> <button>Eliminar</button>
Prueba Editada	Prueba	123	555	<button>Editar</button> <button>Eliminar</button>

- Eliminación de un Chofer

Choferes

Index		Agregar		
Nombre	Apellido	Telefono	Licencia	-
Prueba Editada	Prueba	123	555	<button>Editar</button> <button>Eliminar</button>

Repositorio

<https://github.com/an-delacruz/Pyrhon-U2>

Conclusiones y Resultados

Es importante conocer los distintos entornos en los que un lenguaje como Python es capaz de permitirnos desplegar aplicaciones, desde lo más clásico como un programa en consola hasta web que hoy en día está en su auge. Este tipo de herramientas que nos ofrecen los frameworks nos dan las herramientas suficientes para poder seguir creando aplicaciones de gran nivel y funcionamiento además de que estar actualizados con este tipo de tecnologías añade puntos positivos a nuestra experiencia de cara a un mercado profesional.

Juan Esteban Baltierrez Tobon

Durante esta unidad, se conocieron diferentes framework que para futuros desarrollos se pueden implementar, y sobre todo mejorar en un entorno web. Python a sido un lenguaje de programación que es practico y fácil de entender cada una de las cosas, y se pueden mejorar en cuestiones de lógica de programación, entendiendo los diferentes framework que se han utilizado. Algo a resaltar con las librerías que durante esta unidad han sido de gran ayuda, y que han sido fáciles de implementar.

José Ángel Cabrera Morales.

Esta unidad fue interesante y de mucha ayuda para conocer las diferentes formas con las que se puede trabajar con el lenguaje Python en un entorno web, ya sea sin el uso de frameworks o bien haciendo uso de alguno de ellos, los frameworks que se vieron (Django y flask) son bastante utilizados en el mundo laboral por lo que fue de gran ayuda el ver una introducción a estos frameworks para así tener una noción del funcionamiento y los posibles caso de uso que pueden tener, además de la utilización de librerías que pueden ser de mucha ayuda, especialmente la de psycpg2. Es por lo antes mencionado que esta unidad apporto mucho conocimiento nuevo que puede ser de ayuda en un futuro.

Antonio De la Cruz Rivera