

EL programa

```
# -*- coding: utf-8 -*-
```

```
"""hoteles_reservas.py
```

Automatically generated by Colab.

Original file is located at

https://colab.research.google.com/drive/1Zi4b50CV_ZPyMkpWrTyCeSgHOcpHpEWO

Programa que permite realizar varias acciones sobre el archivo de hoteles, clientes y reservas.

Este programa permite trabajar sobre tres archivos JSON :

1. Hoteles.json - Contiene información de los hoteles.
 - Permite crear, consultar, modificar y borrar registros
2. clientes.json - Contiene información de los clients.
 - Permite crear, consultar, modificar y borrar registros
3. reservas.json - Contiene información de las reservas.
 - Permite crear, consultar y borrar registros

El resultado se muestra en pantalla y en cada accion se guarda directamente en cada archivo.

```
"""
```

```
import json
```

```
import os
```

```
from google.colab import drive
```

```
drive.mount('/content/drive', force_remount=True)
```

```
# Definir constantes para las rutas de archivos
```

```
HOTELES_FILE = (
```

```
    "/content/drive/MyDrive/Calidad_software/Semana6/hoteles.json"
```

```
)
```

```
CLIENTES_FILE = (
```

```
    "/content/drive/MyDrive/Calidad_software/Semana6/clientes.json"
```

```
)
```

```
RESERVAS_FILE = (
```

```
    "/content/drive/MyDrive/Calidad_software/Semana6/reservas.json"
```

```
)
```

```
class GestorHoteles:
```

```
    """
```

```
    Clase que permite leer y escribir en el archivo que
```

```
    se requiera
```

```
    """
```

```
    @staticmethod
```

```
    def leer_archivo(ruta):
```

```
        """
```

```
        Leer archivo
```

```
        """
```

```

if not os.path.exists(ruta):
    return []

try:
    with open(ruta, 'r', encoding='utf-8') as file:
        return json.load(file)
except (json.JSONDecodeError, IOError) as e:
    print(f"Error al leer {ruta}: {e}")
    return []

```

@staticmethod

```

def escribir_archivo(ruta, data):
    """
    Escribir archivo
    """
    try:
        with open(ruta, 'w', encoding='utf-8') as file:
            json.dump(data, file, indent=4)
    except IOError as e:
        print(f"Error al escribir {ruta}: {e}")

```

class Hotel:

```

    """
    Se define la clase hotel donde tendra varias
    funciones crear, modificar, eliminar, consultar
    """

    ARCHIVO = HOTELES_FILE

```

```

def __init__(self, hotel_data):
    """
    Inicializa el objeto Hotel a partir de
    un diccionario con los parámetros.
    """
    self.id_hotel = hotel_data.get('id_hotel')
    self.nombre = hotel_data.get('nombre')
    self.ubicacion = hotel_data.get('ubicacion')
    self.habitaciones_disponibles = hotel_data.get(
        'habitaciones_disponibles')
    self.numero_estrellas = hotel_data.get('numero_estrellas')

def to_dict(self):
    """
    Archivo
    """
    return self.__dict__

@classmethod
def crear_hotel(cls, hotel_data):
    """
    Crear registro hotel
    """
    hoteles = GestorHoteles.leer_archivo(cls.ARCHIVO)

```

```

# Verificar si el ID ya existe

if any(hotel["id_hotel"] ==
       hotel_data["id_hotel"] for hotel in hoteles):
    print(f"Error: Ya existe un hotel con ID "
          f"{hotel_data['id_hotel']}".)
    return

# Agregar nuevo hotel
hoteles.append(hotel_data)

GestorHoteles.escribir_archivo(cls.ARCHIVO, hoteles)

print("Hotel agregado exitosamente.")

```

```

@classmethod
def eliminar_hotel(cls, id_hotel):
    """
    eliminar registro hotel
    """

    hoteles = GestorHoteles.leer_archivo(cls.ARCHIVO)
    hoteles = [h for h in hoteles if h['id_hotel'] != id_hotel]
    GestorHoteles.escribir_archivo(cls.ARCHIVO, hoteles)
    print(f"Hotel con ID {id_hotel} eliminado.")

```

```

@classmethod
def modificar_hotel(cls, id_hotel, datos_actualizados):
    """
    modificar registro hotel

```

```

"""

hoteles = GestorHoteles.leer_archivo(cls.ARCHIVO)

for hotel in hoteles:

    if hotel['id_hotel'] == id_hotel:

        for key, value in datos_actualizados.items():

            if value is not None: # Si no es None, actualiza el campo

                hotel[key] = value

        GestorHoteles.escribir_archivo(cls.ARCHIVO, hoteles)

        print("Hotel actualizado exitosamente.")

        return

print(f"Hotel con ID {id_hotel} no encontrado.")

```

@classmethod

```

def mostrar_info(cls, id_hotel=None):

    """

    consultar registro hotel

    """

    hoteles = GestorHoteles.leer_archivo(cls.ARCHIVO)

    if id_hotel:

        try:

            id_hotel = int(id_hotel) # Convertir a entero

        except ValueError:

            print("ID inválido. Debe ser un número.")

            return

    for hotel in hoteles:

```

```
        if hotel['id_hotel'] == id_hotel:

            print(json.dumps(hotel, indent=4))

            return

        print(f"Hotel con ID {id_hotel} no encontrado")

    else:

        print(json.dumps(hoteles, indent=4))
```

```
class Cliente:
```

```
    """
```

```
    Se define la clase cliente donde tendra varias
    funciones crear, modificar, eliminar, consultar
```

```
    """
```

```
    ARCHIVO = CLIENTES_FILE
```

```
    def __init__(self, id_cliente, nombre, email):
```

```
        self.id_cliente = id_cliente
```

```
        self.nombre = nombre
```

```
        self.email = email
```

```
    def to_dict(self):
```

```
        """
```

```
        archivo
```

```
        """
```

```
        return self.__dict__
```

```
    @classmethod
```

```

def crear_cliente(cls, id_cliente, nombre, email):
    """
    Crear registro cliente
    """

    clientes = GestorHoteles.leer_archivo(cls.ARCHIVO)

    # Verificar si el ID ya existe
    if any(cliente["id_cliente"] == id_cliente for cliente in clientes):
        print(f"Error: Ya existe un cliente con ID {id_cliente}.")
        return

    # Agregar nuevo cliente
    clientes.append(Cliente(id_cliente, nombre, email).to_dict())
    GestorHoteles.escribir_archivo(cls.ARCHIVO, clientes)
    print("Cliente agregado exitosamente.")

    @classmethod
    def eliminar_cliente(cls, id_cliente):
        """
        eliminar registro cliente
        """

        clientes = GestorHoteles.leer_archivo(cls.ARCHIVO)
        clientes = [c for c in clientes if c["id_cliente"] != id_cliente]
        GestorHoteles.escribir_archivo(cls.ARCHIVO, clientes)

    @classmethod

```



```
def modificar_cliente(cls, id_cliente, nombre=None, email=None):
```

```
    """
```

```
    modificar registro cliente
```

```
    """
```

```
    clientes = GestorHoteles.leer_archivo(cls.ARCHIVO)
```

```
    for cliente in clientes:
```

```
        if cliente['id_cliente'] == id_cliente:
```

```
            if nombre is not None:
```

```
                cliente['nombre'] = nombre
```

```
            if email is not None:
```

```
                cliente['email'] = email
```

```
            GestorHoteles.escribir_archivo(cls.ARCHIVO, clientes)
```

```
            print("Cliente actualizado")
```

```
            return
```

```
    print("Cliente no encontrado")
```

```
    return
```

```
@classmethod
```

```
def mostrar_info(cls, id_cliente=None):
```

```
    """
```

```
    consultar registro cliente
```

```
    """
```

```
    clientes = GestorHoteles.leer_archivo(cls.ARCHIVO)
```

```
    if id_cliente:
```

```
        try:
```

```
            id_cliente = int(id_cliente) # Convertir a entero
```

```

except ValueError:

    print("ID inválido. Debe ser un número.")

    return

for cliente in clientes:

    if cliente['id_cliente'] == id_cliente:

        print(json.dumps(cliente, indent=4))

        return

    print(f"Cliente con ID {id_cliente} no encontrado")

else:

    print(json.dumps(clientes, indent=4))

```

```

class Reserva:

```

```

    """

```

```

    Definir clase reserva

```

```

    """

```

```

    ARCHIVO = RESERVAS_FILE

```

```

    def __init__(self, id_reserva, id_cliente, id_hotel):

```

```

        self.id_reserva = id_reserva

```

```

        self.id_cliente = id_cliente

```

```

        self.id_hotel = id_hotel

```

```

    def to_dict(self):

```

```

        """

```

```

        Archivo

```

```

        """

```

```
return self.__dict__
```

```
@classmethod
```

```
def crear_reserva(cls, id_reserva, id_cliente, id_hotel):
```

```
    """
```

```
    Crear registro reserva
```

```
    """
```

```
    reservas = GestorHoteles.leer_archivo(cls.ARCHIVO)
```

```
    # Verificar si el ID ya existe
```

```
    if any(reserva["id_reserva"] == id_reserva for reserva in reservas):
```

```
        print(f"Error: Ya existe una reserva con ID {id_reserva}.")
```

```
    return
```

```
    # Agregar nueva reserva
```

```
    reservas.append(Reserva(id_reserva, id_cliente, id_hotel).to_dict())
```

```
    GestorHoteles.escribir_archivo(cls.ARCHIVO, reservas)
```

```
    print("Reserva agregada exitosamente.")
```

```
@classmethod
```

```
def cancelar_reserva(cls, id_reserva):
```

```
    """
```

```
    Cancelar registro reserva
```

```
    """
```

```
    reservas = GestorHoteles.leer_archivo(cls.ARCHIVO)
```

```
    reservas = [r for r in reservas if r['id_reserva'] != id_reserva]
```

```
GestorHoteles.escribir_archivo(cls.ARCHIVO, reservas)
```

```
@classmethod
```

```
def mostrar_info(cls, id_reserva=None):
```

```
    """
```

```
    Mostrar registro reserva
```

```
    """
```

```
    reservas = GestorHoteles.leer_archivo(cls.ARCHIVO)
```

```
    if id_reserva:
```

```
        try:
```

```
            id_reserva = int(id_reserva) # Convertir a entero
```

```
        except ValueError:
```

```
            print("ID inválido. Debe ser un número.")
```

```
        return
```

```
    for reserva in reservas:
```

```
        if reserva['id_reserva'] == id_reserva:
```

```
            print(json.dumps(reserva, indent=4))
```

```
        return
```

```
    print(f"Reserva con ID {id_reserva} no encontrada")
```

```
    else:
```

```
        print(json.dumps(reservas, indent=4))
```

```
def prueba1():
```

```
    """
```

```
    Se define funciones para en este caso
```

consultar la informacion de cada uno de los archivos

"""

while True:

print("\nSeleccione una opción:")

print("1. Mostrar hoteles")

print("2. Mostrar clientes")

print("3. Mostrar reservas")

print("4. Salir")

opcion = input("Ingrese el número de la opción: ")

if opcion == "1":

id_hotel = input(

"Ingrese el ID del hotel (o presione Enter para ver todos): "

).strip()

Hotel.mostrar_info(id_hotel if id_hotel else None)

elif opcion == "2":

id_cliente = input(

"Ingrese el ID del cliente (o presione Enter para ver todos): "

).strip()

Cliente.mostrar_info(id_cliente if id_cliente else None)

elif opcion == "3":

id_reserva = input(

"Ingrese el ID de la reserva(o presione Enter ver todos): "

```

        ).strip()

Reserva.mostrar_info(id_reserva if id_reserva else None)

elif opcion == "4":
    print("Saliendo...")
    break
else:
    print("Opción no válida, intente de nuevo.")

if __name__ == "__main__":
    prueba1()

# Ejemplo de uso
def prueba2():
    """
    Se dejecutan las funciones para crear registros
    en cada uno de los archivos
    """
    # Hotel.crear_hotel(3, "Hotel Sol", "Santa Martha", 100, 5)
    # Hotel.crear_hotel(4, "Sol caribe", "Barranquilla", 85, 3)
    # Hotel.crear_hotel(10, "Hotel Amazonas", "Amazonas", 30, 4)
    # Hotel.crear_hotel(11, "Hotel Riviera", "Mexico", 150, 5)
    hotel_data = {
        "id_hotel": 3,
        "nombre": "Hotel Sol",
        "ubicacion": "Santa Martha",

```

```

    "habitaciones_disponibles": 100,
    "numero_estrellas": 5
}

Hotel.crear_hotel(hotel_data)

Cliente.crear_cliente(3, "Maritza Guerrero", "malige1@hotmail.com")

Cliente.crear_cliente(5, "Oscar Rodriguez", "OscarR@gmail.com")

Cliente.crear_cliente(4, "Elim Gutierrez", "ElimG@gmail.com")

Cliente.crear_cliente(6, "Alfonso Ospina", "AlfonsoO@hotmail.com")

Reserva.crear_reserva(3, 1, 1)

Reserva.crear_reserva(3, 3, 3)

Reserva.crear_reserva(4, 4, 3)

Reserva.crear_reserva(5, 10, 6)


if __name__ == "__main__":
    prueba2()


# Ejemplo de uso modificacion

def prueba3():
    """
    Se ejecutan las funciones para modificar registros
    en cada uno de los archivos
    """

    # Hotel.modificar_hotel(2, ubicacion='Cartagena')

    # Hotel.modificar_hotel(12, ubicacion='Colombia')

    # Hotel.modificar_hotel(4, nombre='Sol Caribe y Son')

    datos_actualizados = {

```

```
"nombre": "Nuevo Nombre de Hotel",  
"ubicacion": "Nueva Ubicacion",  
"habitaciones_disponibles": 60  
}  
Hotel.modificar_hotel(1, datos_actualizados)  
Cliente.modificar_cliente(1, nombre="Juan Carlos Fernandez")  
Cliente.modificar_cliente(7, nombre="Omar Geles")  
Cliente.modificar_cliente(4, email='ElimGM@gmail.com')  
  
if __name__ == "__main__":  
    prueba3()
```

Elabore primero el código para que cumpliera con los primeros puntos requeridos, es decir, crear, modificar, consultar, eliminar tanto para el archivo de hoteles, como para el de clientes y reservas.

Luego comencé a probar las ejecuciones, me salieron varios errores, no me salía la información, con lo cual poco a poco fui ajustando el desarrollo.

Luego ejecute el desarrollo validando que no me fuera crear registros con los mismos ID de cada archivo, me los creo, con lo cual ajuste el código para asegurar que no me cree registros con llaves ya creadas previamente


Colocando los controles, ya se emite el mensaje:


```

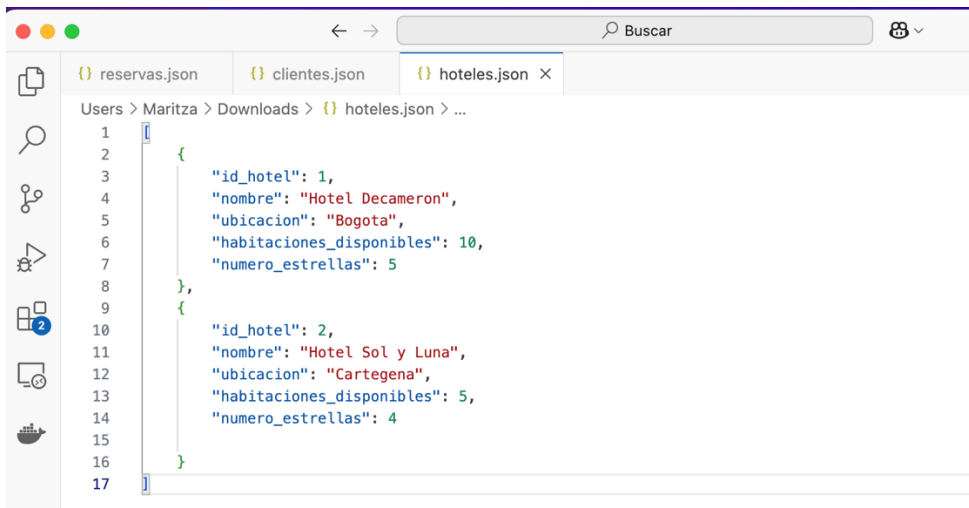
# Ejemplo de uso
def main():
    Hotel.crear_hotel(1, "Hotel Sol", "Mexico", 200, 5)
    Cliente.crear_cliente(1, "Maritza Guerrero", "malige1@hotmail.com")
    Reserva.crear_reserva(1, 1, 1)

if __name__ == "__main__":
    main()

```

 Error: Ya existe un hotel con ID 1.
 Error: Ya existe un cliente con ID 1.
 Error: Ya existe una reserva con ID 1.

Antes de comenzar a ejecutar las sentencias, los archivos tienen:

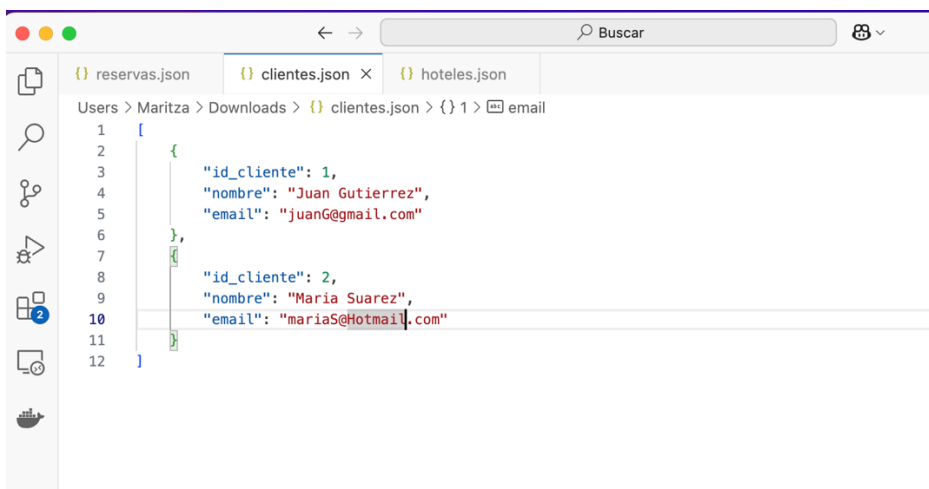


The screenshot shows a code editor with three tabs: reservas.json, clientes.json, and hoteles.json. The hoteles.json tab is active, showing a JSON array with two hotel objects. The first object has id_hotel: 1, nombre: "Hotel Decameron", ubicacion: "Bogota", habitaciones_disponibles: 10, and numero_estrellas: 5. The second object has id_hotel: 2, nombre: "Hotel Sol y Luna", ubicacion: "Cartegena", habitaciones_disponibles: 5, and numero_estrellas: 4.

```

1 [
2   {
3     "id_hotel": 1,
4     "nombre": "Hotel Decameron",
5     "ubicacion": "Bogota",
6     "habitaciones_disponibles": 10,
7     "numero_estrellas": 5
8   },
9   {
10    "id_hotel": 2,
11    "nombre": "Hotel Sol y Luna",
12    "ubicacion": "Cartegena",
13    "habitaciones_disponibles": 5,
14    "numero_estrellas": 4
15  }
16 ]
17

```

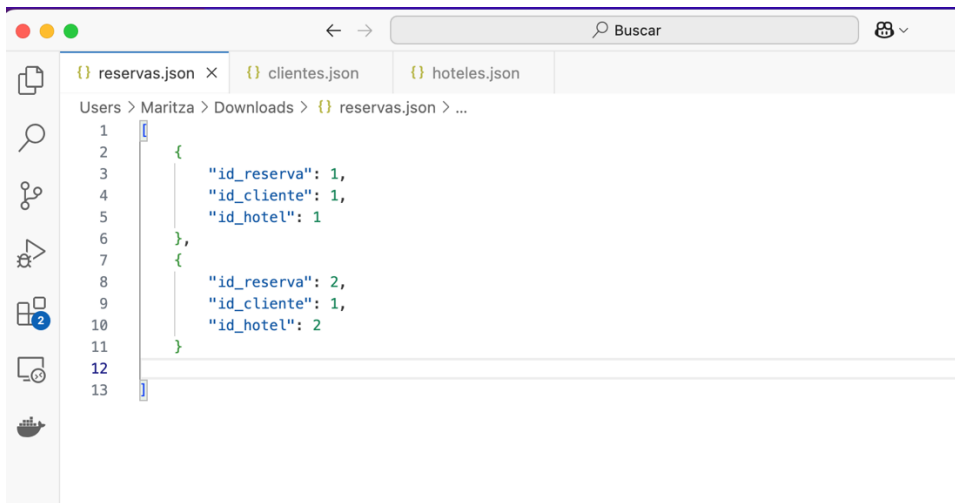


The screenshot shows a code editor with three tabs: reservas.json, clientes.json, and hoteles.json. The clientes.json tab is active, showing a JSON array with two client objects. The first object has id_cliente: 1, nombre: "Juan Gutierrez", and email: "juanG@gmail.com". The second object has id_cliente: 2, nombre: "Maria Suarez", and email: "mariaS@hotmail.com".

```

1 [
2   {
3     "id_cliente": 1,
4     "nombre": "Juan Gutierrez",
5     "email": "juanG@gmail.com"
6   },
7   {
8     "id_cliente": 2,
9     "nombre": "Maria Suarez",
10    "email": "mariaS@hotmail.com"
11  }
12 ]

```



Se ejecuto sentencia de creación en cada uno de los archivos

```
# Ejemplo de uso
def main():
    Hotel.crear_hotel(3, "Hotel Sol", "Santa Martha", 100, 5)
    Hotel.crear_hotel(4, "Sol caribe", "Barranquilla", 85, 3)
    Hotel.crear_hotel(10, "Hotel Amazonas", "Amazonas", 30, 4)
    Hotel.crear_hotel(11, "Hotel Riviera", "Mexico", 150, 5)
    Cliente.crear_cliente(3, "Maritza Guerrero", "malige1@hotmail.com")
    Cliente.crear_cliente(5, "Oscar Rodriguez", "OscarR@gmail.com")
    Cliente.crear_cliente(4, "Elim Gutierrez", "ElimG@gmail.com")
    Cliente.crear_cliente(6, "Alfonso Ospina", "Alfonso0@hotmail.com")
    Reserva.crear_reserva(3, 1, 1)
    Reserva.crear_reserva(3, 3, 3)
    Reserva.crear_reserva(4, 4, 3)
    Reserva.crear_reserva(5, 10, 6)

if __name__ == "__main__":
    main()
```

El resultado fue:

```
Hotel agregado exitosamente.
Hotel agregado exitosamente.
Hotel agregado exitosamente.
Hotel agregado exitosamente.
Cliente agregado exitosamente.
Cliente agregado exitosamente.
Cliente agregado exitosamente.
Cliente agregado exitosamente.
Reserva agregada exitosamente.
Error: Ya existe una reserva con ID 3.
Reserva agregada exitosamente.
Reserva agregada exitosamente.
```

Consultando el archivo se evidencia el ingreso de nuevos registros

```
"ubicacion": "Cartegena",
"habitaciones_disponibles": 5,
"numero_estrellas": 4
},
{
  "id_hotel": 3,
  "nombre": "Hotel Sol",
  "ubicacion": "Santa Martha",
  "habitaciones_disponibles": 100,
  "numero_estrellas": 5
},
{
  "id_hotel": 4,
  "nombre": "Sol caribe",
  "ubicacion": "Barranquilla",
  "habitaciones_disponibles": 85,
  "numero_estrellas": 3
},
{
  "id_hotel": 10,
  "nombre": "Hotel Amazonas",
  "ubicacion": "Amazonas",
  "habitaciones_disponibles": 30,
  "numero_estrellas": 4
},
{
  "id_hotel": 11,
  "nombre": "Hotel Riviera",
  "ubicacion": "Mexico",
  "habitaciones_disponibles": 150,
  "numero_estrellas": 5
}
```

Captura de Pantalla

Revisando el archivo de clientes se evidencia los datos nuevos incluidos:

```

{
  "id_cliente": 2,
  "nombre": "Maria Suarez",
  "email": "mariaS@Hotmail.com"
},
{
  "id_cliente": 3,
  "nombre": "Maritza Guerrero",
  "email": "maligel@hotmail.com"
},
{
  "id_cliente": 5,
  "nombre": "Oscar Rodriguez",
  "email": "OscarR@gmail.com"
},
{
  "id_cliente": 4,
  "nombre": "Elim Gutierrez",
  "email": "ElimG@gmail.com"
},
{
  "id_cliente": 6,
  "nombre": "Alfonso Ospina",
  "email": "AlfonsoO@hotmail.com"
}

```

Captura de Pantalla

Revisando el archivo de Reservas, se evidencia los nuevos registros incluidos:

```

[
  {
    "id_reserva": 1,
    "id_cliente": 1,
    "id_hotel": 1
  },
  {
    "id_reserva": 2,
    "id_cliente": 1,
    "id_hotel": 2
  },
  {
    "id_reserva": 3,
    "id_cliente": 1,
    "id_hotel": 1
  },
  {
    "id_reserva": 4,
    "id_cliente": 4,
    "id_hotel": 3
  },
  {
    "id_reserva": 5,
    "id_cliente": 10,
    "id_hotel": 6
  }
]

```

Se ejecuto la sentencia para modificar datos de los hoteles:

```

▶ # Ejemplo de uso modificacion
def main():

    Hotel.modificar_hotel(2, ubicacion='Cartagena')
    Hotel.modificar_hotel(12, ubicacion='Colombia')
    Hotel.modificar_hotel(4, nombre='Sol Caribe y Son')
    Cliente.modificar_cliente(1, nombre="Juan Carlos Fernandez")
    Cliente.modificar_cliente(7, nombre="Omar Geles")
    Cliente.modificar_cliente(4, email="ElimGM@gmail.com")

if __name__ == "__main__":
    main()

```

Modificaciones en hoteles

```

{
  "id_hotel": 2,
  "nombre": "Hotel Sol y Luna",
  "ubicacion": "Cartagena",
  "habitaciones_disponibles": 5,
  "numero_estrellas": 4
},
{
  "id_hotel": 3,
  "nombre": "Hotel Sol",
  "ubicacion": "Santa Martha",
  "habitaciones_disponibles": 100,
  "numero_estrellas": 5
},
{
  "id_hotel": 4,
  "nombre": "Sol Caribe y Son",
  "ubicacion": "Barranquilla",
  "habitaciones_disponibles": 85,
  "numero_estrellas": 3
},
{
  "id_hotel": 10,
  "nombre": "Hotel Amazonas",
  "ubicacion": "Amazonas",
  "habitaciones_disponibles": 30,
  "numero_estrellas": 4
}

```

Modificaciones en Clientes

```
[
    {
        "id_cliente": 1,
        "nombre": "Juan Carlos Fernandez",
        "email": "juanG@gmail.com"
    },
    {
        "id_cliente": 2,
        "nombre": "Maria Suarez",
        "email": "mariaS@Hotmail.com"
    },
    {
        "id_cliente": 3,
        "nombre": "Maritza Guerrero",
        "email": "maligel@hotmail.com"
    },
    {
        "id_cliente": 5,
        "nombre": "Oscar Rodriguez",
        "email": "OscarR@gmail.com"
    },
    {
        "id_cliente": 4,
        "nombre": "Elim Gutierrez",
        "email": "ElimGM@gmail.com"
    },
    {

```

Resultado de ejecutar la sentencia

```

Hotel actualizado
Hotel no encontrado.
Hotel actualizado
Cliente actualizado
Cliente no encontrado
Cliente actualizado

```

Después de asegurar que se está ejecutando correctamente el programa, lo ajusto según recomendaciones PEP8 , es decir lo documento , ajusto indentación, longitud de línea, espacios nombres de variables entre otras.

Luego ejecute el comando pylint para realizar validaciones estáticas y me salieron los siguientes errores:

```

!pylint "/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py"
***** Module Hoteles_reservas
drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:0: C0304: Final newline missing (missing-final-newline)
drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:0: C0301: Line too long (19630/100) (line-too-long)
drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:0: C0114: Missing module docstring (missing-module-docstring)
drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:0: C0103: Module name "Hoteles_reservas" doesn't conform to snake_case (invalid-module-name)
drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:0: W0104: Statement seems to have no effect (pointless-statement)

Your code has been rated at 0.00/10

```

Voy a ejecutarlo en Visual Studio

```
PROBLEMAS 3 SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS bash

hoteles_reservas_py.py:221:0: W0311: Bad indentation. Found 15 spaces, expected 16 (bad-indentation)
hoteles_reservas_py.py:223:0: W0311: Bad indentation. Found 15 spaces, expected 16 (bad-indentation)
hoteles_reservas_py.py:224:0: W0311: Bad indentation. Found 15 spaces, expected 16 (bad-indentation)
hoteles_reservas_py.py:229:0: W0311: Bad indentation. Found 19 spaces, expected 20 (bad-indentation)
hoteles_reservas_py.py:230:0: W0311: Bad indentation. Found 19 spaces, expected 20 (bad-indentation)
hoteles_reservas_py.py:364:0: C0304: Final newline missing (missing-final-newline)
hoteles_reservas_py.py:27:0: E0401: Unable to import 'google.colab' (import-error)
hoteles_reservas_py.py:47:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:58:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:66:0: R0913: Too many arguments (6/5) (too-many-arguments)
hoteles_reservas_py.py:66:0: R0917: Too many positional arguments (6/5) (too-many-positional-arguments)
hoteles_reservas_py.py:71:14: E0602: Undefined variable 'GestorArchivos' (undefined-variable)
hoteles_reservas_py.py:71:42: E1101: Module 'hoteles_reservas_py' has no 'ARCHIVO' member (no-member)
hoteles_reservas_py.py:81:4: E0602: Undefined variable 'GestorArchivos' (undefined-variable)
hoteles_reservas_py.py:81:36: E1101: Module 'hoteles_reservas_py' has no 'ARCHIVO' member (no-member)
hoteles_reservas_py.py:84:0: C0115: Missing class docstring (missing-class-docstring)
hoteles_reservas_py.py:91:4: R0913: Too many arguments (6/5) (too-many-arguments)
hoteles_reservas_py.py:91:4: R0917: Too many positional arguments (6/5) (too-many-positional-arguments)
hoteles_reservas_py.py:99:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:103:4: C0116: Missing function or method docstring (missing-function-docstring)
```

```
PROBLEMAS 3 SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS bash

hoteles_reservas_py.py:127:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:127:4: R0913: Too many arguments (6/5) (too-many-arguments)
hoteles_reservas_py.py:127:4: R0917: Too many positional arguments (6/5) (too-many-positional-arguments)
hoteles_reservas_py.py:147:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:164:0: C0115: Missing class docstring (missing-class-docstring)
hoteles_reservas_py.py:176:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:180:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:194:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:201:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:217:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:235:0: C0115: Missing class docstring (missing-class-docstring)
hoteles_reservas_py.py:243:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:247:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:261:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:267:4: C0116: Missing function or method docstring (missing-function-docstring)
hoteles_reservas_py.py:326:0: E0102: function already defined line 284 (function-redefined)
hoteles_reservas_py.py:348:0: E0102: function already defined line 284 (function-redefined)

-----
Your code has been rated at 6.82/10
```

Me dedico a ajustar lo que salió y me costo mucho, ya que se tenían muchos parámetros en algunas clases, espacios, no estaban documentados todos las clases

```
Your code has been rated at 9.64/10 (previous run: 9.46/10, +0.18)

MacBook-de-Maritza:Downloads Maritza$ pylint hoteles_reservas_py.py
***** Module hoteles_reservas_py
hoteles_reservas_py.py:27:0: E0401: Unable to import 'google.colab' (import-error)

-----
Your code has been rated at 9.74/10 (previous run: 9.64/10, +0.10)

MacBook-de-Maritza:Downloads Maritza$
```

Cerrando el anterior

Me dispongo en ejecutar el Flake8

```
!flake8 "/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py"
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18466: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18473: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18482: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18517: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18533: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18538: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18553: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18612: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18648: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18663: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18668: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18724: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18780: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18840: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18911: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:18971: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:19034: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:19039: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:19044: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:19077: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:19092: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:19098: E231 missing whitespace after ':'
/content/drive/MyDrive/Calidad_software/Semana6/Hoteles_reservas.py:1:19098: E231 missing whitespace after ':'
```

En Visual Studio

```
PROBLEMAS 1 SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
[notice] To update, run: pip install --upgrade pip
MacBook-de-Maritza:Downloads Maritza$ flake8 hoteles_reservas_py.py
hoteles_reservas_py.py:41:1: E302 expected 2 blank lines, found 1
hoteles_reservas_py.py:71:1: E302 expected 2 blank lines, found 1
hoteles_reservas_py.py:77:5: E301 expected 1 blank line, found 0
hoteles_reservas_py.py:79:80: E501 line too long (81 > 79 characters)
hoteles_reservas_py.py:84:80: E501 line too long (82 > 79 characters)
hoteles_reservas_py.py:101:80: E501 line too long (81 > 79 characters)
hoteles_reservas_py.py:102:80: E501 line too long (80 > 79 characters)
hoteles_reservas_py.py:111:5: E303 too many blank lines (2)
hoteles_reservas_py.py:138:5: E303 too many blank lines (2)
hoteles_reservas_py.py:159:1: E302 expected 2 blank lines, found 1
hoteles_reservas_py.py:204:5: E303 too many blank lines (2)
hoteles_reservas_py.py:236:13: E303 too many blank lines (2)
hoteles_reservas_py.py:244:1: E302 expected 2 blank lines, found 1
hoteles_reservas_py.py:308:1: E302 expected 2 blank lines, found 1
hoteles_reservas_py.py:336:80: E501 line too long (80 > 79 characters)
hoteles_reservas_py.py:346:1: E305 expected 2 blank lines after class or function definition, found 1
hoteles_reservas_py.py:350:1: E302 expected 2 blank lines, found 1
hoteles_reservas_py.py:355:5: E265 block comment should start with '#'
hoteles_reservas_py.py:356:5: E265 block comment should start with '#'
hoteles_reservas_py.py:357:5: E265 block comment should start with '#'
hoteles_reservas_py.py:358:5: E265 block comment should start with '#'
hoteles_reservas_py.py:360:5: E122 continuation line missing indentation or outdented
hoteles_reservas_py.py:361:5: E122 continuation line missing indentation or outdented
hoteles_reservas_py.py:362:5: E122 continuation line missing indentation or outdented
hoteles_reservas_py.py:363:5: E122 continuation line missing indentation or outdented
hoteles_reservas_py.py:364:5: E122 continuation line missing indentation or outdented
hoteles_reservas_py.py:373:31: E231 missing whitespace after ','
hoteles_reservas_py.py:373:33: E202 whitespace before ')'
hoteles_reservas_py.py:376:1: E305 expected 2 blank lines after class or function definition, found 1
```

Después de eliminar demasiados espacios, no dejar espacio entre # y el siguiente comando, aun líneas demasiadas largas, se pudo concluir satisfactoriamente las validaciones, dando como resultado:


```
hoteles_reservas_py.py:107:80: E501 line too long (80 > 79 characters)
hoteles_reservas_py.py:162:1: E302 expected 2 blank lines, found 1
❌ MacBook-de-Maritza:Downloads Maritza$ flake8 hoteles_reservas_py.py
hoteles_reservas_py.py:107:20: E999 SyntaxError: unterminated f-string literal (detected at line 107)
❌ MacBook-de-Maritza:Downloads Maritza$ flake8 hoteles_reservas_py.py
hoteles_reservas_py.py:108:20: E999 SyntaxError: invalid syntax
❌ MacBook-de-Maritza:Downloads Maritza$ flake8 hoteles_reservas_py.py
hoteles_reservas_py.py:79:5: E301 expected 1 blank line, found 0
hoteles_reservas_py.py:79:36: W291 trailing whitespace
hoteles_reservas_py.py:81:47: W291 trailing whitespace
hoteles_reservas_py.py:105:36: W291 trailing whitespace
hoteles_reservas_py.py:107:80: E501 line too long (80 > 79 characters)
❌ MacBook-de-Maritza:Downloads Maritza$ hoteles_reservas_py.py:105:36: W291 trailing whitespace
bash: hoteles_reservas_py.py:105:36:: command not found
❌ MacBook-de-Maritza:Downloads Maritza$ flake8 hoteles_reservas_py.py
hoteles_reservas_py.py:79:5: E301 expected 1 blank line, found 0
hoteles_reservas_py.py:107:80: E501 line too long (80 > 79 characters)
❌ MacBook-de-Maritza:Downloads Maritza$ flake8 hoteles_reservas_py.py
hoteles_reservas_py.py:79:1: W293 blank line contains whitespace
hoteles_reservas_py.py:108:80: E501 line too long (80 > 79 characters)
❌ MacBook-de-Maritza:Downloads Maritza$ flake8 hoteles_reservas_py.py
hoteles_reservas_py.py:108:80: E501 line too long (80 > 79 characters)
● MacBook-de-Maritza:Downloads Maritza$ flake8 hoteles_reservas_py.py
○ MacBook-de-Maritza:Downloads Maritza$
```

Captura de Pantalla