



ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

CARRERA

NRC - ASIGNATURA:	14908
PROFESORA:	Ing. Diego Gamboa
PERÍODO ACADÉMICO:	2024_50

PRUEBA PARCIAL 2

TÍTULO:

PRUEBA PARCIAL 2

ESTUDIANTE

Valdiviezo, Darwin

FECHA DE ENTREGA:

2024 / 07 / 14

CALIFICACIÓN OBTENIDA:

1. Alcance

Desarrollar un sistema de citas médicas, mediante python con Flask y con una base de datos postgres, hacer casos de pruebas correspondientes, y casos de uso.

2. Objetivos

Evaluar el conocimiento y la aplicación de las pruebas de estrés, las pruebas de recuperación, pruebas de integración, análisis de valores, límites.

3. Descripción

CITAS MÉDICAS

CREACIÓN DE BACKEND FLASK

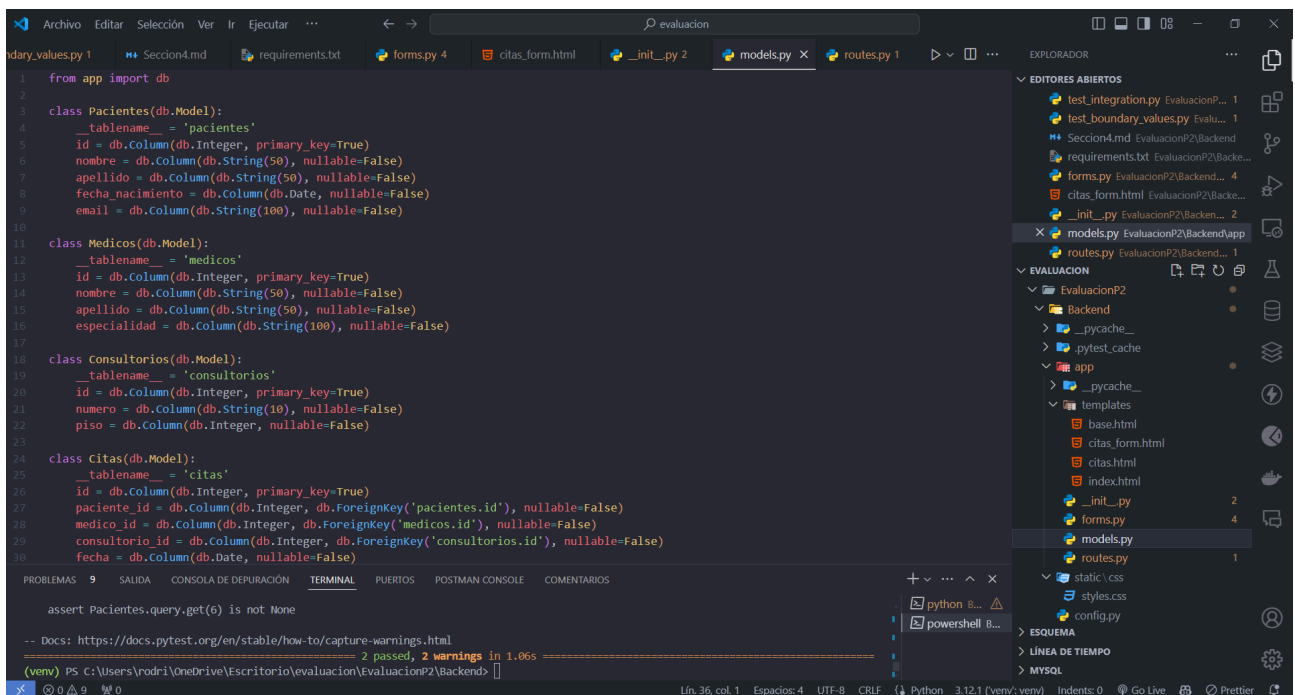


Figura 1

Interfaz Frontend y ejecución:

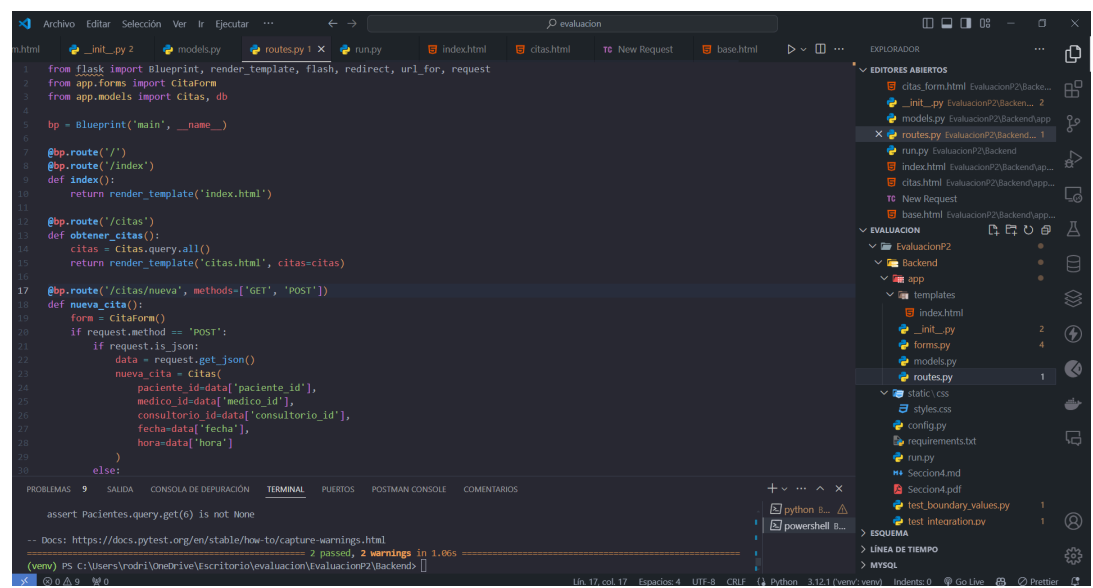


Figura 2

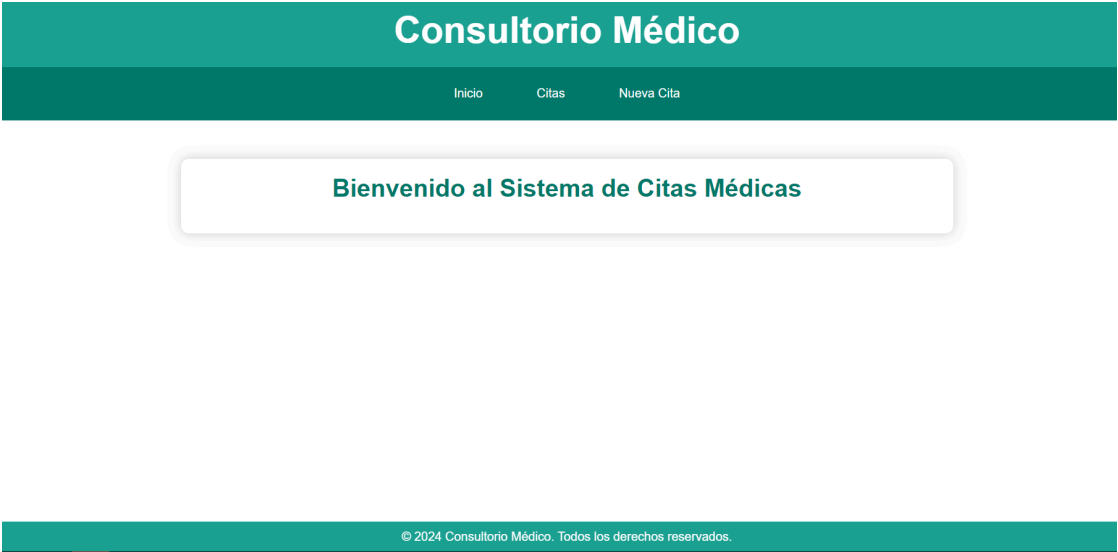


Figura 3



Figura 4



Figura 5

CREACIÓN DE LA BASE DE DATOS:

```
CREATE TABLE IF NOT EXISTS pacientes (  
  id INT PRIMARY KEY,  
  nombre VARCHAR(50) NOT NULL,  
  apellido VARCHAR(50) NOT NULL,  
  fecha_nacimiento DATE NOT NULL CHECK (fecha_nacimiento >= CURRENT_DATE - INTERVAL  
'90 years'),  
  email VARCHAR(100) NOT NULL  
);  
  
CREATE TABLE IF NOT EXISTS medicos (  
  id INT PRIMARY KEY,  
  nombre VARCHAR(50) NOT NULL,
```

```

    apellido VARCHAR(50) NOT NULL,
    especialidad VARCHAR(100) NOT NULL
);

CREATE TABLE IF NOT EXISTS consultorios (
    id INT PRIMARY KEY,
    numero VARCHAR(10) NOT NULL,
    piso INT NOT NULL
);

CREATE TABLE IF NOT EXISTS citas (
    id SERIAL PRIMARY KEY,
    paciente_id INT NOT NULL,
    medico_id INT NOT NULL,
    consultorio_id INT NOT NULL,
    fecha DATE NOT NULL,
    hora TIME NOT NULL,
    FOREIGN KEY (paciente_id) REFERENCES pacientes(id),
    FOREIGN KEY (medico_id) REFERENCES medicos(id),
    FOREIGN KEY (consultorio_id) REFERENCES consultorios(id),
    CONSTRAINT check_fecha CHECK (fecha >= CURRENT_DATE),
    CONSTRAINT check_hora CHECK (hora >= '08:00:00' AND hora <= '17:00:00')
);

INSERT INTO pacientes (id, nombre, apellido, fecha_nacimiento, email) VALUES
(1, 'Darwin', 'Perez', '1990-01-15', 'darwin.perez@example.com'),
(2, 'Andres', 'Gomez', '1985-07-22', 'andres.gomez@example.com'),
(3, 'Paola', 'Ramirez', '1992-11-30', 'paola.ramirez@example.com');

INSERT INTO medicos (id, nombre, apellido, especialidad) VALUES
(1, 'Carlos', 'Fernandez', 'Cardiología'),
(2, 'Maria', 'Lopez', 'Dermatología'),
(3, 'Juan', 'Martinez', 'Pediatría');

INSERT INTO consultorios (id, numero, piso) VALUES
(1, '101', 1),
(2, '202', 2),
(3, '303', 3);

```

Sección 1:

PRUEBAS DE ESTRÉS

- Utilizar Apache JMeter

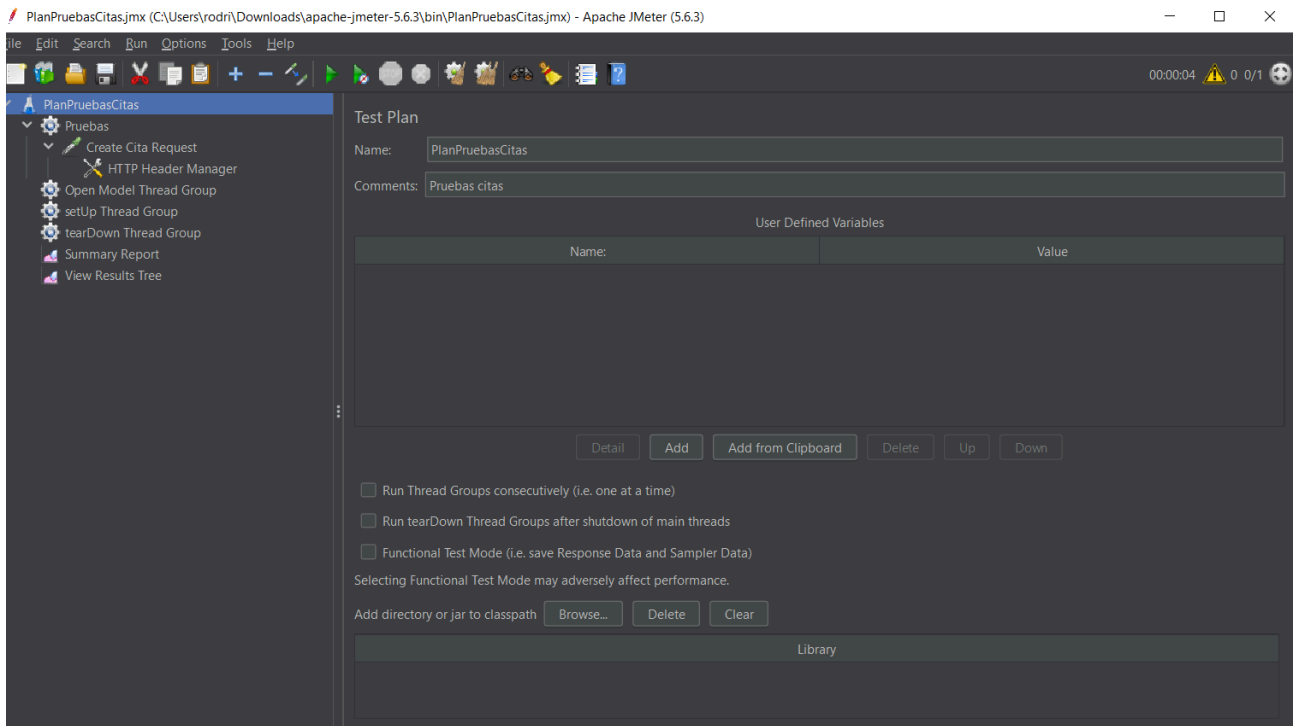


Figura 6

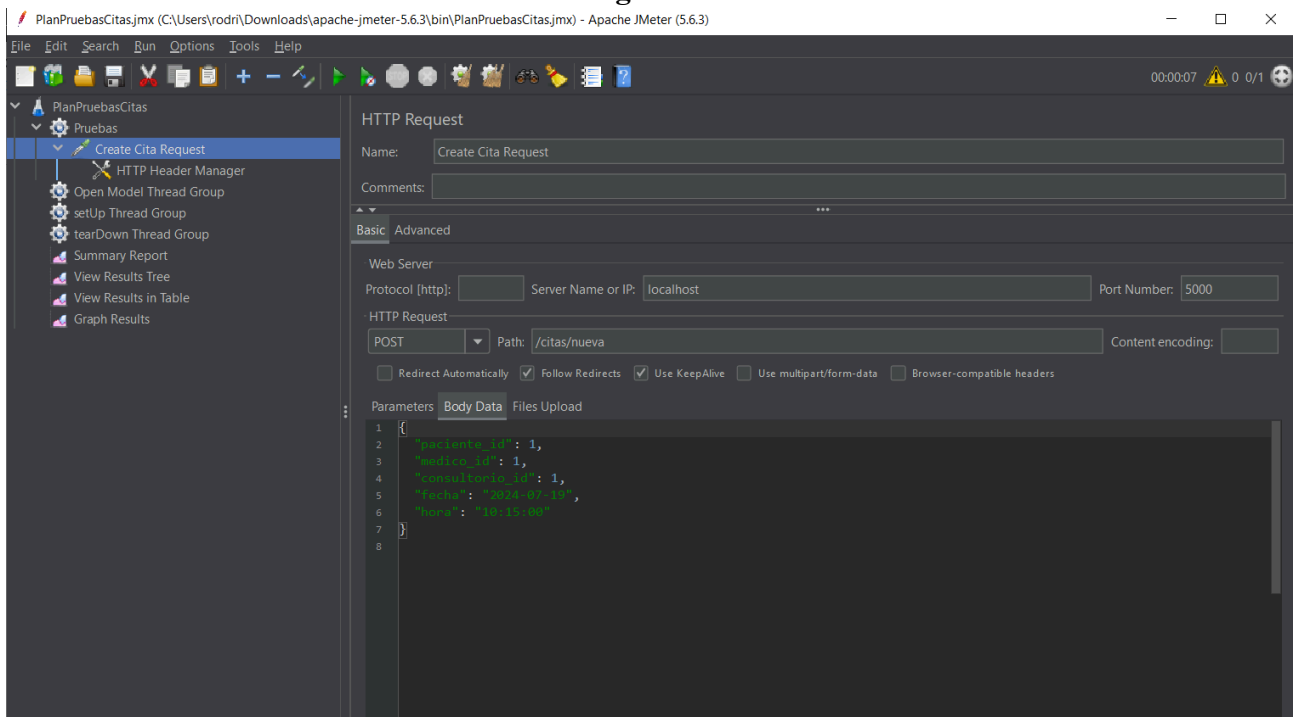


Figura 7

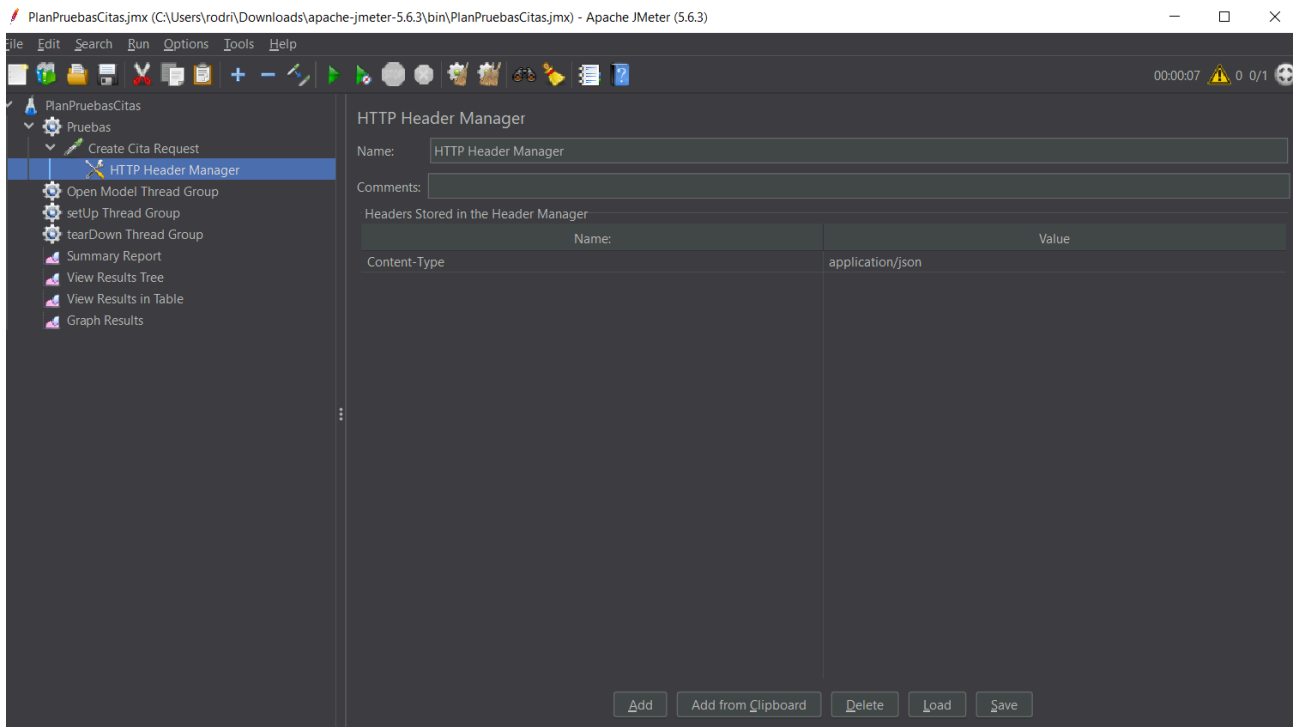


Figura 8

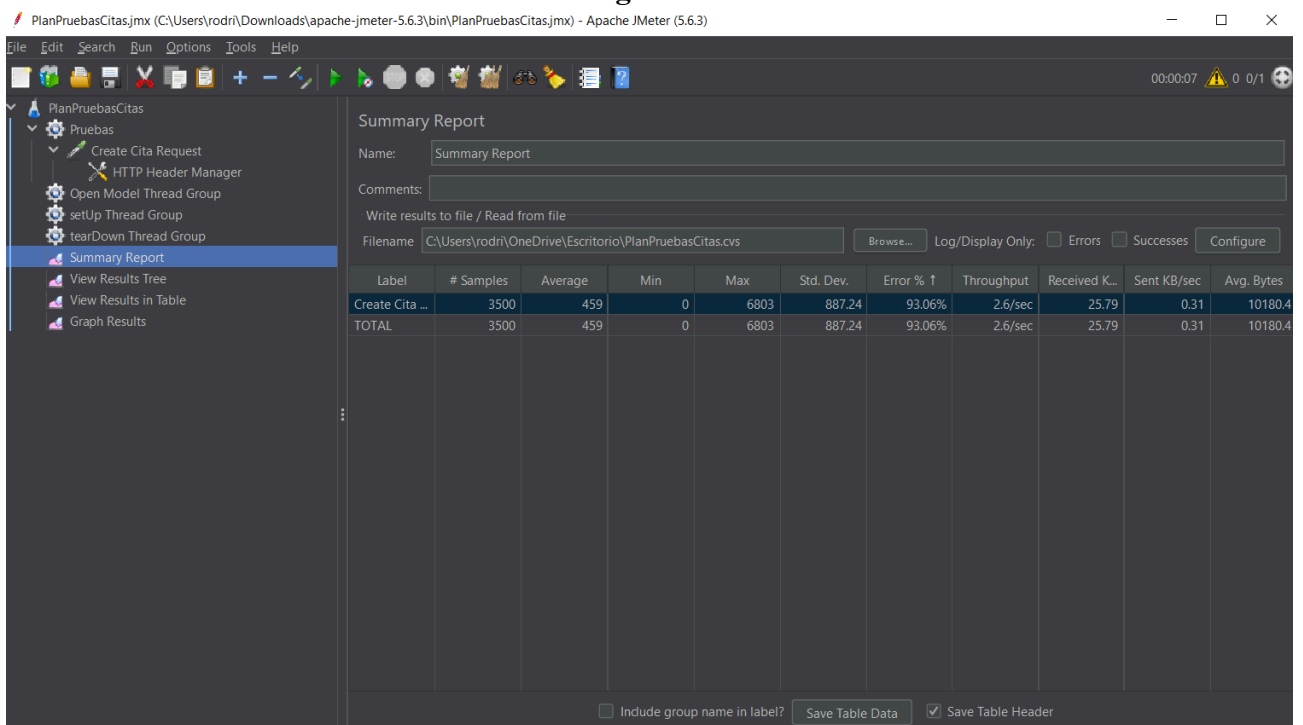


Figura 9

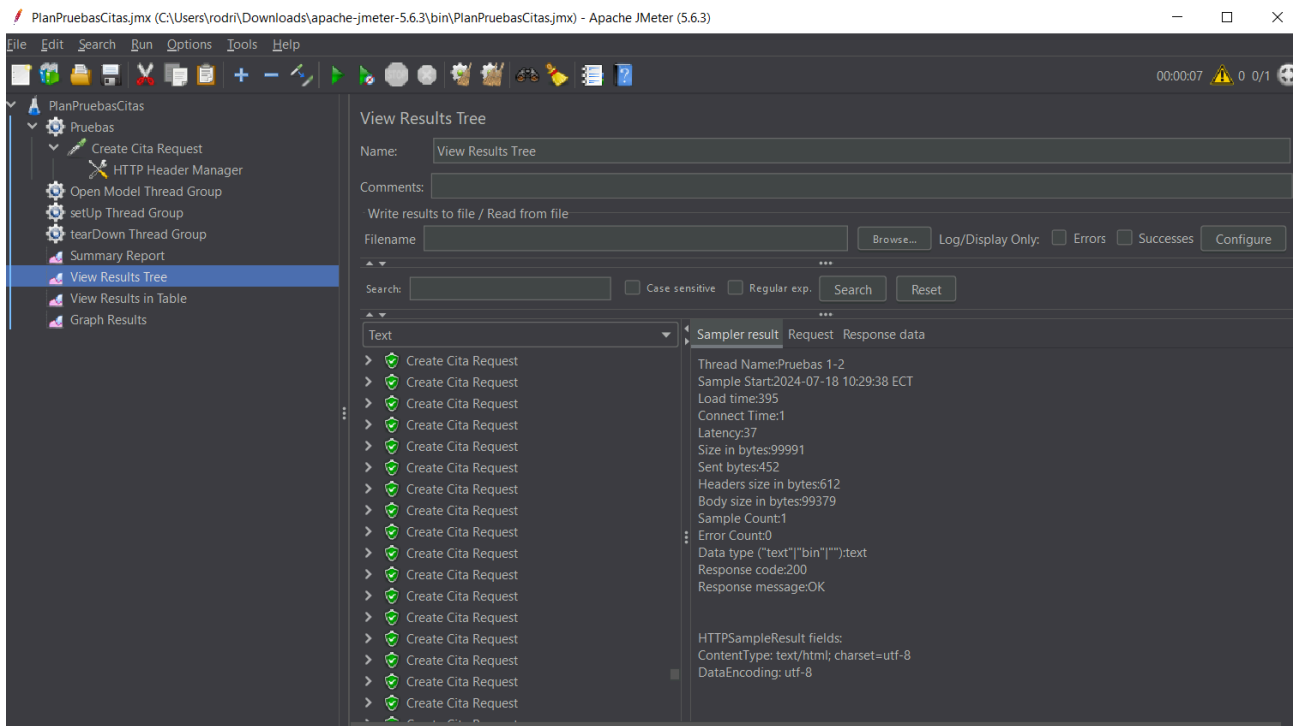


Figura 10

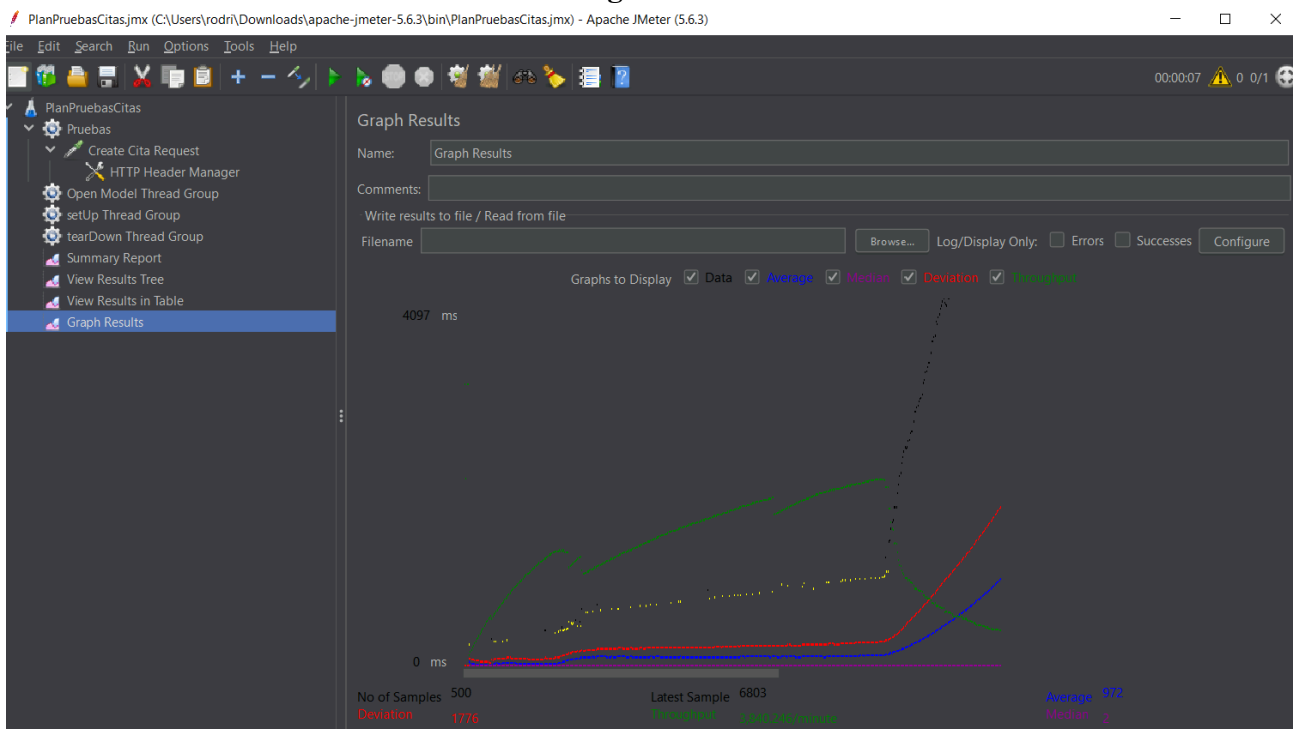


Figura 11

- Usar la API citas médicas “POST”

```
@bp.route('/citas/nueva', methods=['GET', 'POST'])
def nueva_cita():
    form = CitaForm()
    if request.method == 'POST':
        if request.is_json:
            data = request.get_json()
```



```

nueva_cita = Citas(
    paciente_id=data['paciente_id'],
    medico_id=data['medico_id'],
    consultorio_id=data['consultorio_id'],
    fecha=data['fecha'],
    hora=data['hora']
)
else:
    nueva_cita = Citas(
        paciente_id=form.paciente_id.data,
        medico_id=form.medico_id.data,
        consultorio_id=form.consultorio_id.data,
        fecha=form.fecha.data,
        hora=form.hora.data
    )
try:
    db.session.add(nueva_cita)
    db.session.commit()
    flash('Cita creada exitosamente', 'success')
    return redirect(url_for('main.obtener_citas'))
except Exception as e:
    db.session.rollback()
    flash('Error al crear la cita: {}'.format(e), 'danger')
return render_template('citas_form.html', form=form)

```

- **Configurar el plan de pruebas para simular 500 solicitudes concurrentes a la endpoint de creación de citas durante 5 min.**

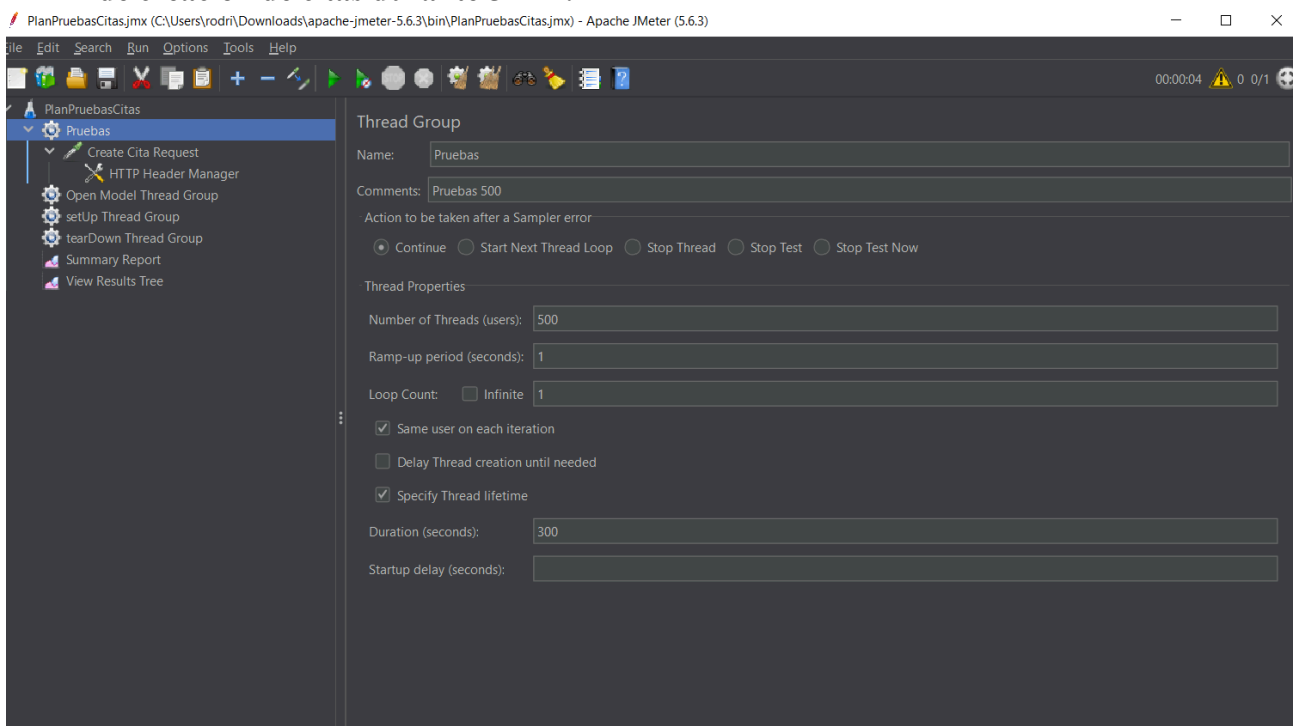
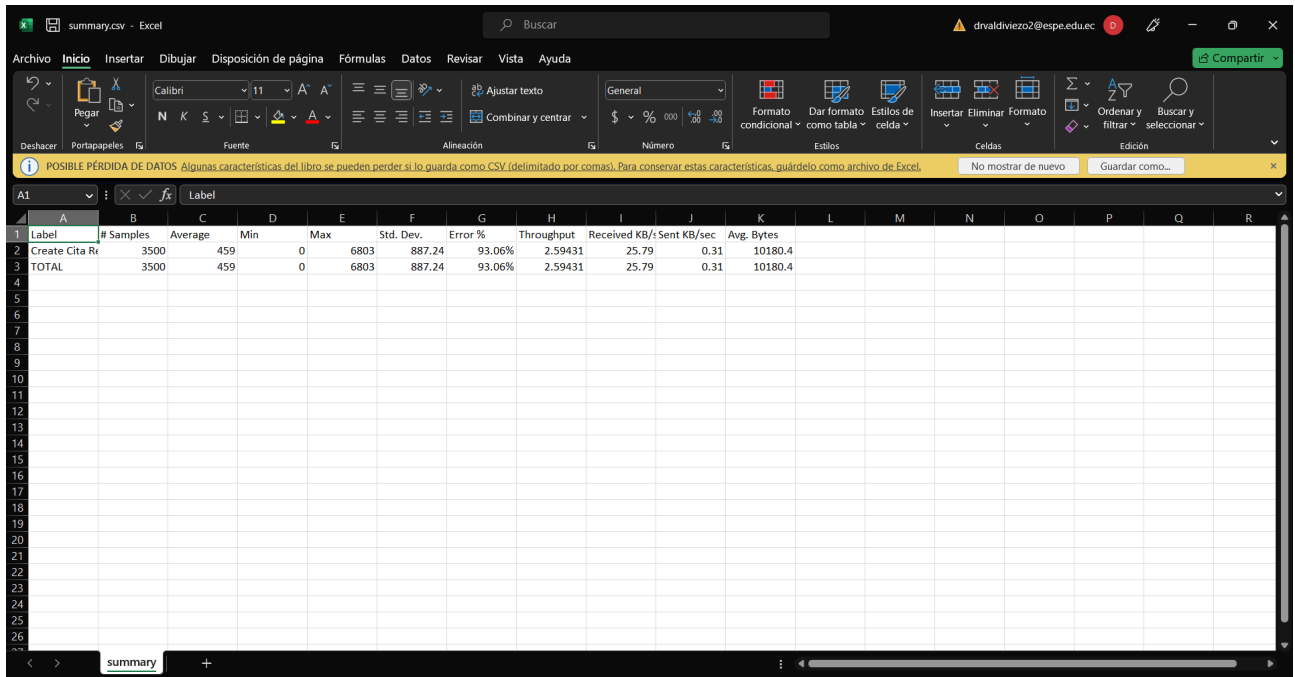


Figura 12

- **Generar un informe de resultados:**
 - **Tiempo de espera promedio**

- Tasa de error



summary.csv - Excel

Archivo Inicio Insertar Dibujar Disposición de página Fórmulas Datos Revisar Vista Ayuda

Calibri 11 Fuente Alineación Número Estilos

POSIBLE PÉRDIDA DE DATOS Algunas características del libro se pueden perder si lo guarda como CSV (delimitado por comas). Para conservar estas características, guárdelo como archivo de Excel.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/s	Sent KB/sec	Avg. Bytes
Create Cita Request	3500	459	0	6803	887.24	93.06%	2.59431	25.79	0.31	10180.4
TOTAL	3500	459	0	6803	887.24	93.06%	2.59431	25.79	0.31	10180.4

Figura 13

- Gráfico de rendimiento

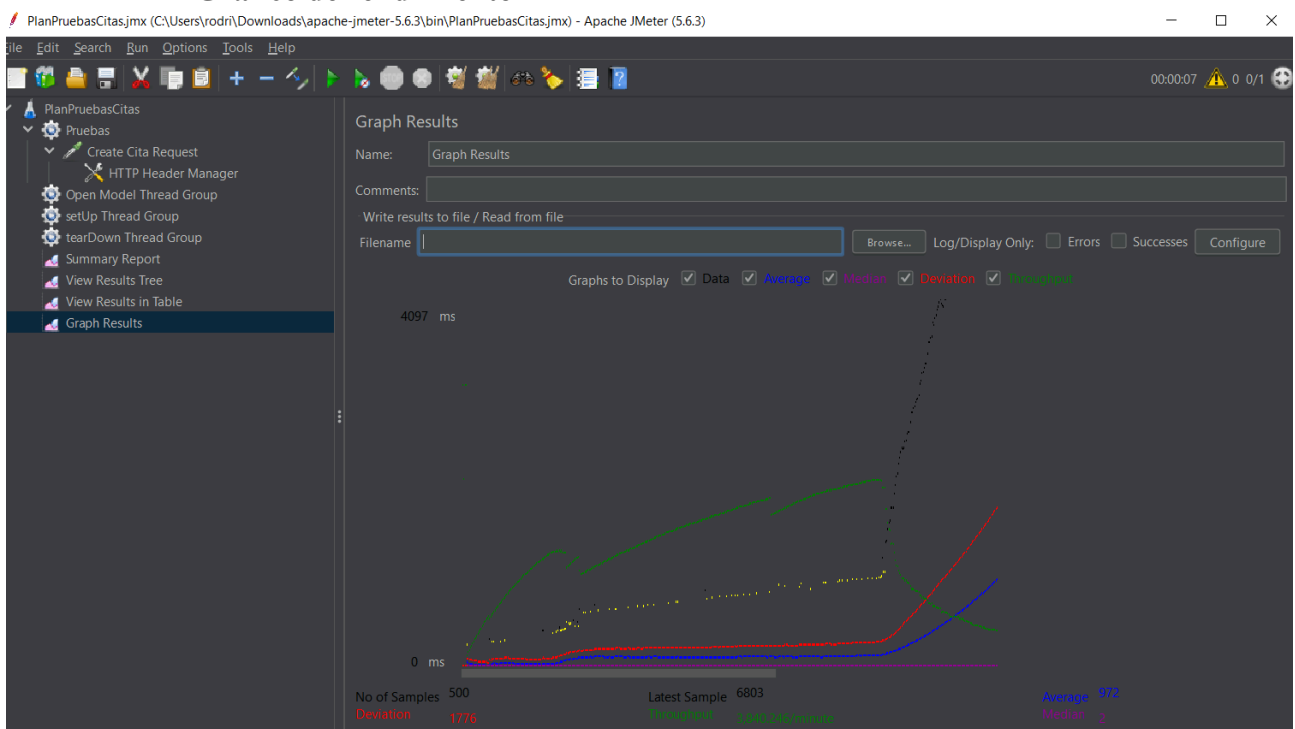


Figura 14

Podemos ver que en total hay un promedio de 459 ms, un mínimo de 0, un máximo de 6803 ms, y una desviación estándar de 887,24.

La tasa de error es del 93,06%.

Y tenemos un rendimiento de 2,594331 solicitudes cada segundo, recibiendo los datos cada 25,79 KB/s, y enviandolos cada 0,31 KB/s.

Lo cual nos indica que el sistema y su base de datos no puede manejar muchos datos de manera correcta, como lo son 500 datos concurrentes de usuarios que hacen citas, por esa razón hay una alta tasa de error, producto que nuestro sistema y su servidor esta con sobrecarga de datos.

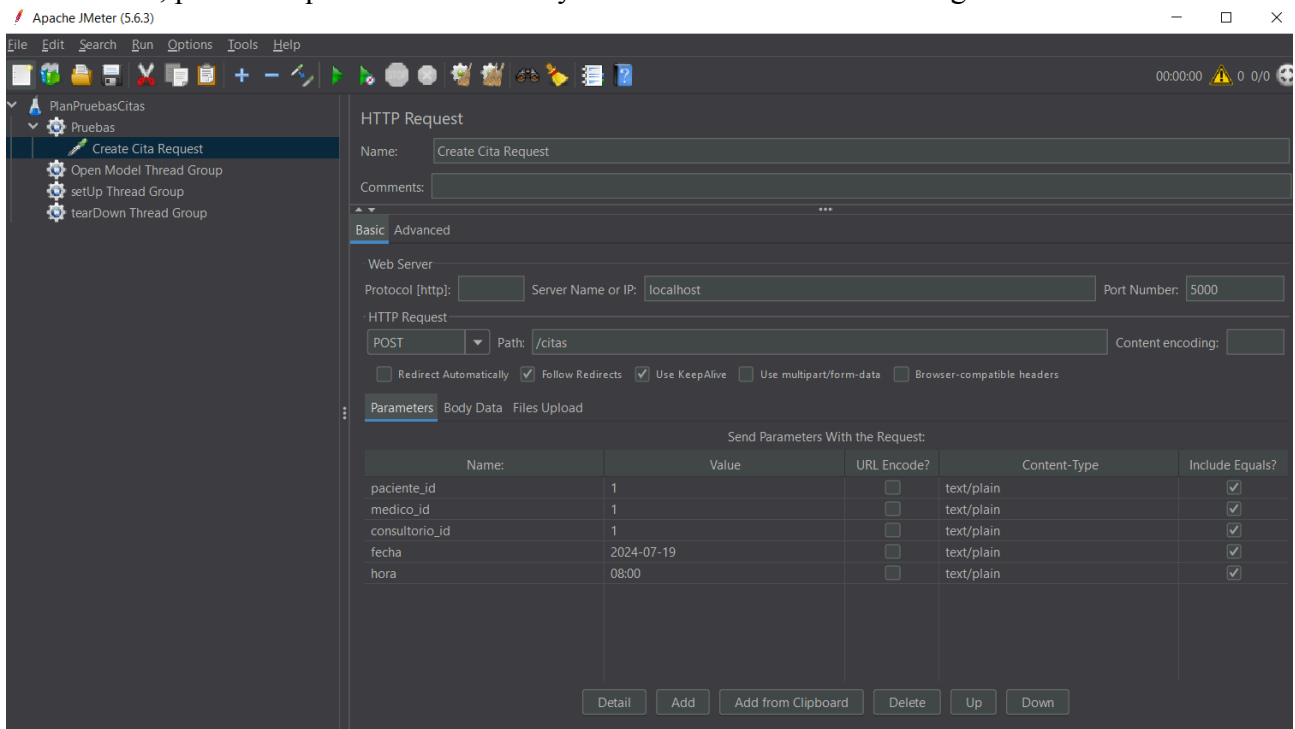


Figura 15

Sección 2:

Caída de la base de datos.

Generación del backup

Generamos el backup haciendo click derecho sobre nuestra base de datos dando clic en crear y backup, y lo guardamos con el nombre que queramos, en nuestro caso BackupPruebas.

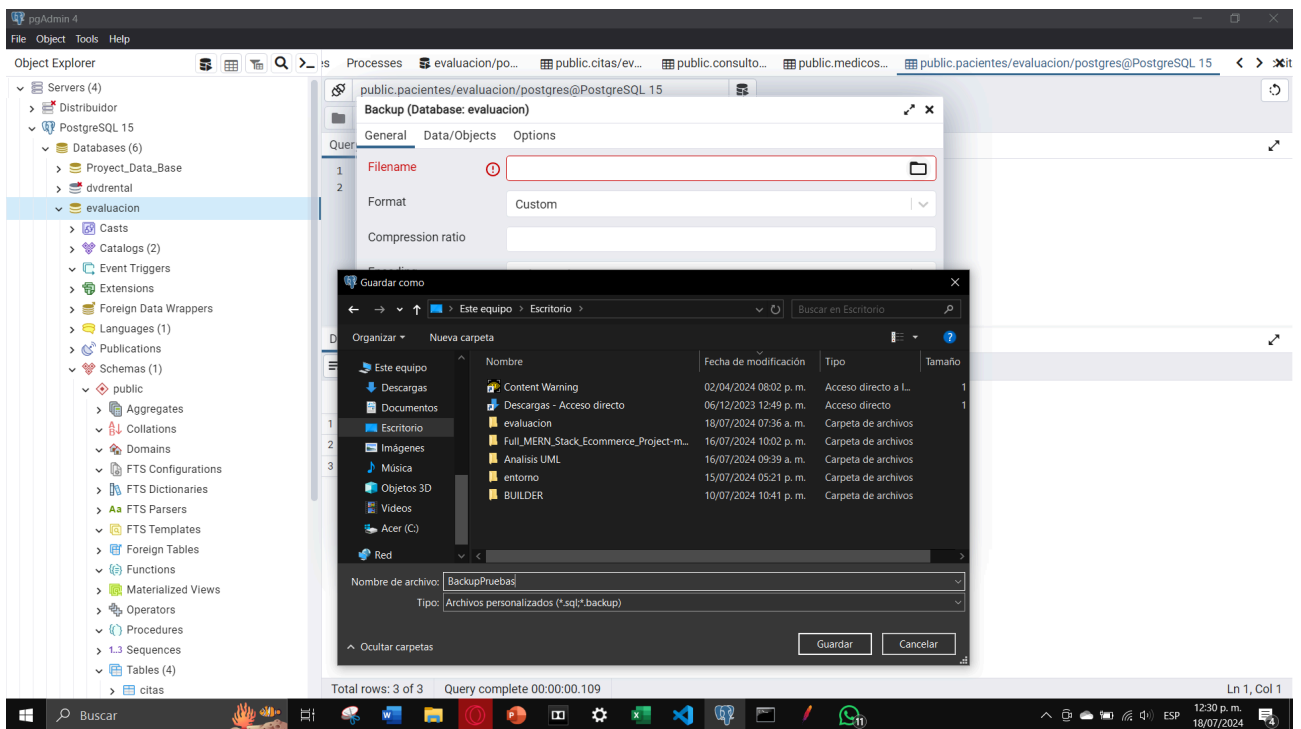


Figura 16

Creación exitosa del backup

Se nos mostrará un mensaje de que el backup fue creado con éxito.

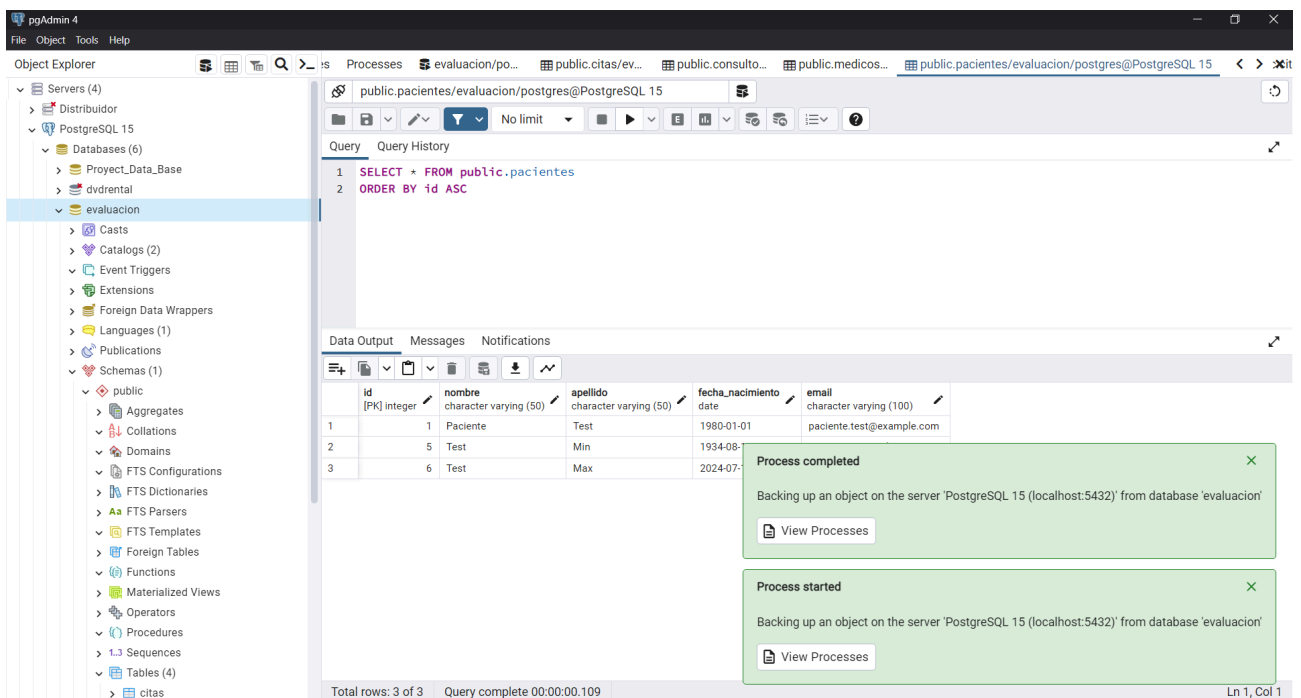


Figura 17

Borramos la tabla citas:

Hacemos uso del comando drop table citas, para borrar la tabla.

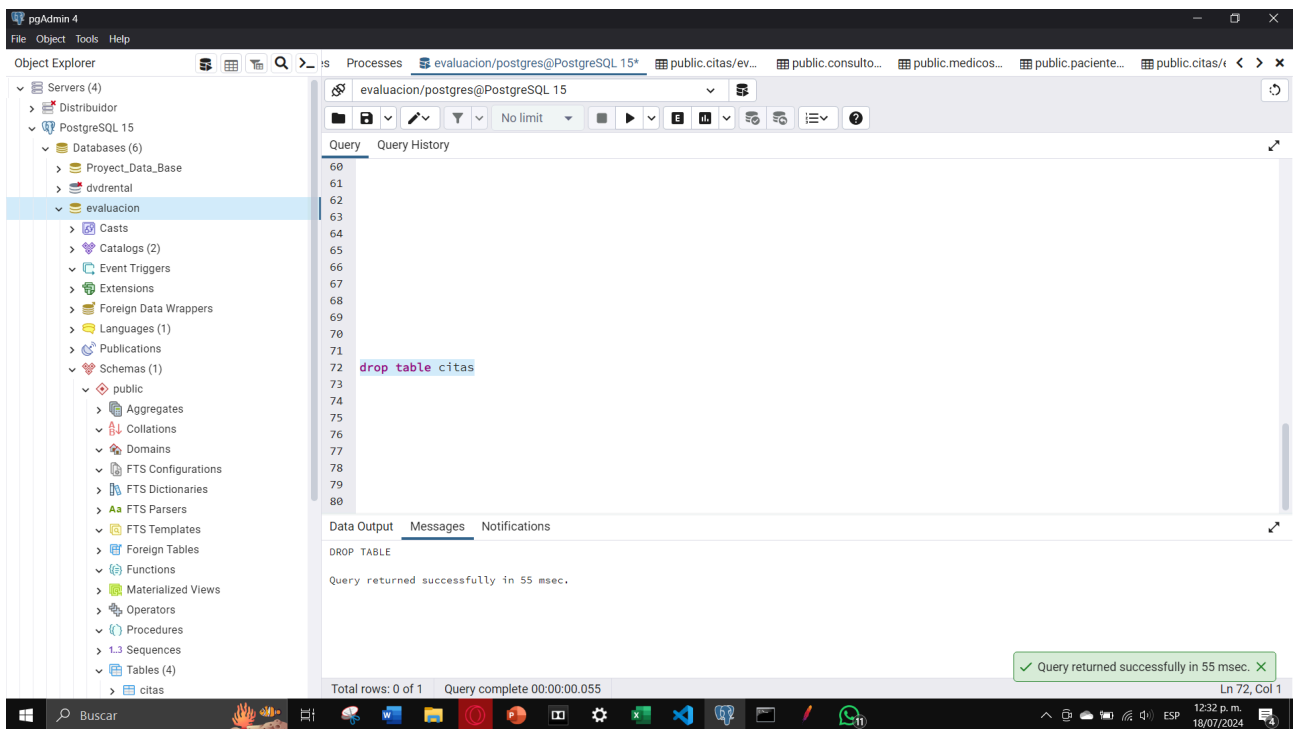


Figura 18

Se visualiza que únicamente quedan 3 tablas y ya no 4 como lo eran antes.

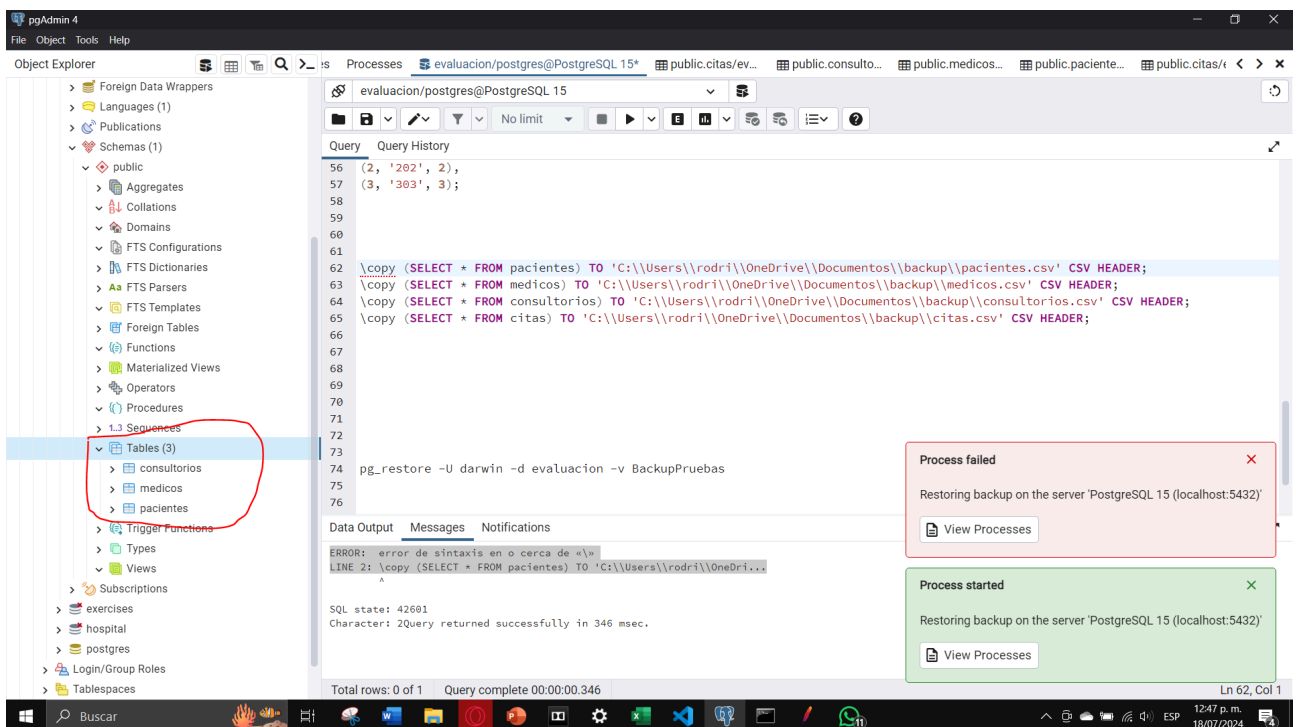


Figura 19

Error al generar y al querer visualizar las citas

Al volver a ejecutar el programa podemos ver que nos da un error significativo al querer obtener los datos de nuestra tabla cita para mostrar en pantalla los datos.

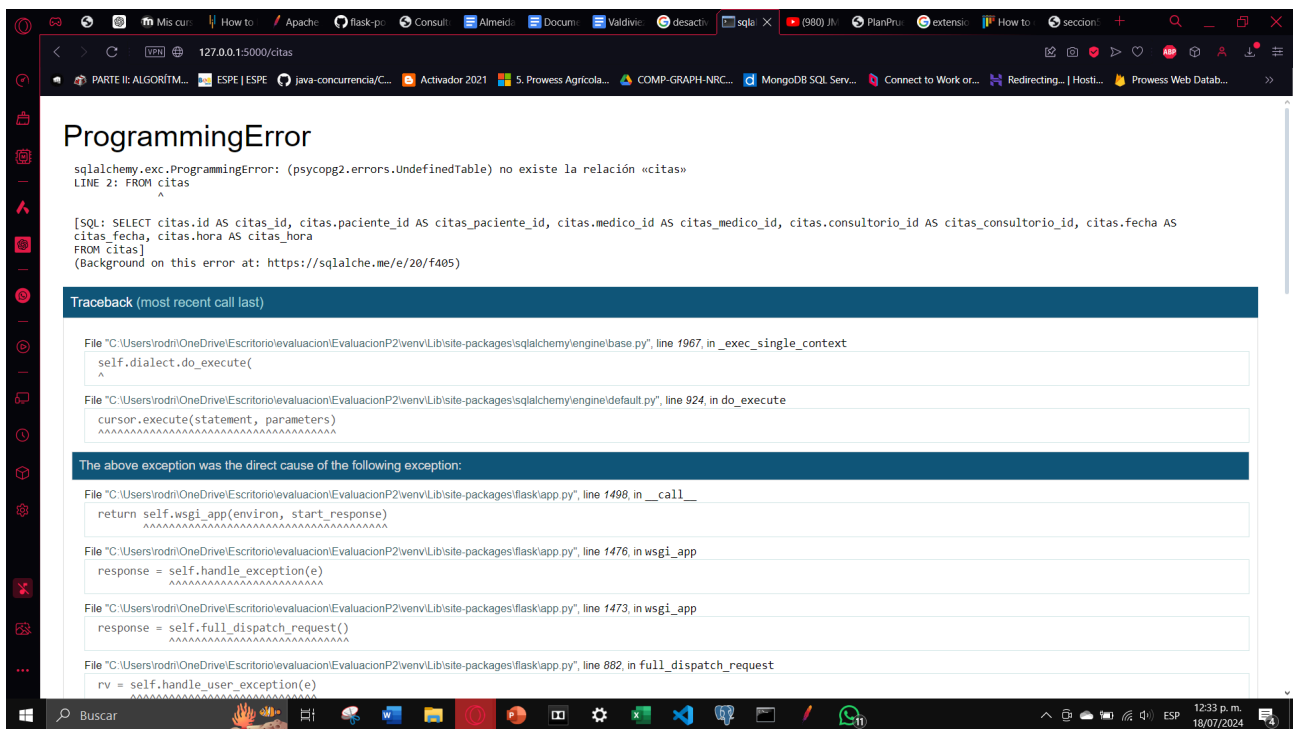


Figura 20

Restauración

Comenzamos con la generación del backup y vemos que al principio nos da error pero las tablas se volverán a generar, únicamente la que borramos.

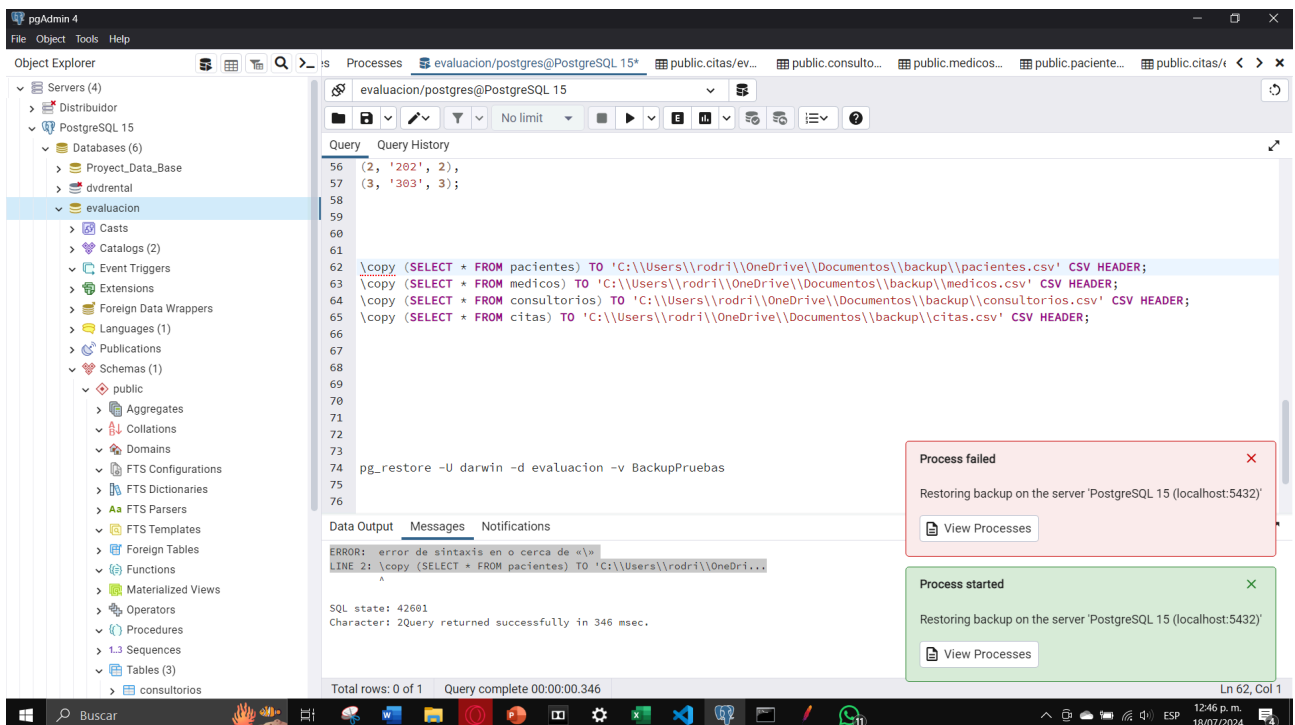


Figura 21

Va a salir error pero si se hará el backup completo

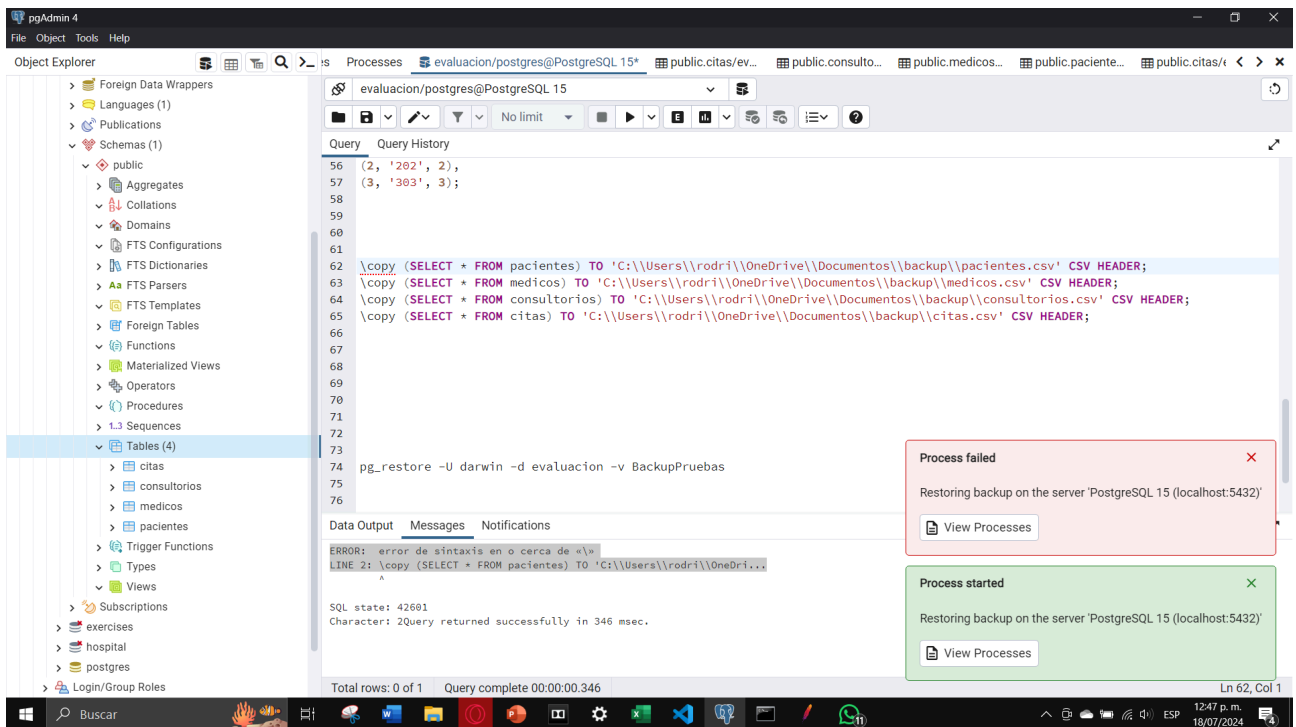


Figura 22

Todo se realiza por la misma interfaz de pgAdmin, el único comando que usamos es el de drop database citas, de esa manera borramos la cita, aunque se puede eliminar de igual manera manualmente por la interfaz gráfica, y como vemos el backup funcionó correctamente.

Sección 3: Se utilizó Pytest para Python

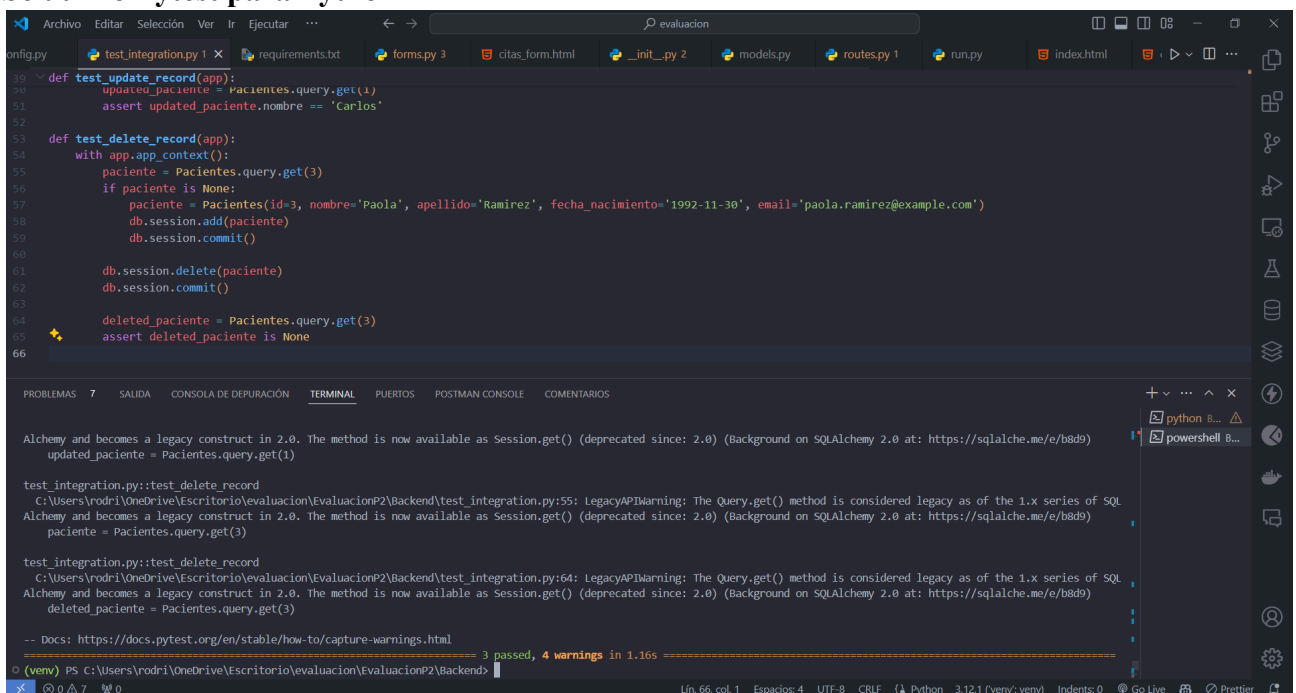


Figura 23

Test_Insert_Records

Se ingresa un paciente, un medio y una cita en la base de datos, los cuales son:

Paciente: id=4, nombre='Laura', apellido='Gonzalez', fecha_nacimiento='1989-12-10', email='laura.gonzalez@example.com'

Médico: id=4, nombre='Ana', apellido='Gomez', especialidad='Neurología'

Cita: paciente_id=4, medico_id=4, consultorio_id=4, fecha='2024-07-20', hora='10:00:00'

Consultorio: id=4, numero='404', piso=4

Los registros se deben ingresar y ver en las tablas que corresponden.

Test_Update_Record

Vamos a actualizar el nombre de un paciente en la BDD, cambio a:

id=1, nombre='Juan', apellido='Perez', fecha_nacimiento='1980-01-01', email='juan.perez@example.com'

Se debe ver el nombre del paciente actualizado en la base.

Test_Delete_Record

Se debe eliminar un paciente de la BDD, el cual será:

id=3, nombre='Paola', apellido='Ramirez', fecha_nacimiento='1992-11-30', email='paola.ramirez@example.com'

El paciente se elimina correctamente.

Resultados de las pruebas:

Test_Insert_Records - **Pasó Exitosamente.**

Test_Update_Record - **Pasó Exitosamente.**

Test_Delete_Record - **Pasó Exitosamente.**

Una observación que se vio fue:

Se generaron warnings debido al uso de Query.get(), que ahora se hereda en SQLAlchemy.

Comparación:

Pacientes:

Antes:

	id [PK] integer	nombre character varying (50)	apellido character varying (50)	fecha_nacimiento date	email character varying (100)
1	1	Darwin	Perez	1990-01-15	darwin.perez@example.com
2	2	Andres	Gomez	1985-07-22	andres.gomez@example.com
3	3	Paola	Ramirez	1992-11-30	paola.ramirez@example.com

Figura 20

Después:

	id [PK] integer	nombre character varying (50)	apellido character varying (50)	fecha_nacimiento date	email character varying (100)
1	1	Carlos	Perez	1990-01-15	darwin.perez@example.com
2	2	Andres	Gomez	1985-07-22	andres.gomez@example.com
3	4	Laura	Gonzalez	1989-12-10	laura.gonzalez@example.com

Figura 24

Médicos:

Antes:

	id [PK] integer	nombre character varying (50)	apellido character varying (50)	especialidad character varying (100)
1	1	Carlos	Fernandez	Cardiología
2	2	Maria	Lopez	Dermatología
3	3	Juan	Martinez	Pediatría

Figura 25

Después:

	id [PK] integer	nombre character varying (50)	apellido character varying (50)	especialidad character varying (100)
1	1	Carlos	Fernandez	Cardiología
2	2	Maria	Lopez	Dermatología
3	3	Juan	Martinez	Pediatría
4	4	Ana	Gomez	Neurología

Figura 26

Consultorios:

Antes:

	id [PK] integer	numero character varying (10)	piso integer
1	1	101	1
2	2	202	2
3	3	303	3

Figura 27

Después:

	id [PK] integer	numero character varying (10)	piso integer
1	1	101	1
2	2	202	2
3	3	303	3
4	4	404	4

Figura 28

Citas:

Antes:

	id [PK] integer	paciente_id integer	medico_id integer	consultorio_id integer	fecha date	hora time without time zone
--	--------------------	------------------------	----------------------	---------------------------	---------------	--------------------------------

Figura 29

Después:

	id [PK] integer	paciente_id integer	medico_id integer	consultorio_id integer	fecha date	hora time without time zone
1	1	4	4	4	2024-07-20	10:00:00

Figura 30

Sección 4:

Análisis de valores límites

```
(venv) PS C:\Users\rodri\OneDrive\Escritorio\evaluacion\EvaluacionP2\Backend> pytest test_boundary_values.py
>>
===== test session starts =====
platform win32 -- Python 3.12.1, pytest-8.2.2, pluggy-1.5.0
rootdir: C:\Users\rodri\OneDrive\Escritorio\evaluacion\EvaluacionP2\Backend
collected 2 items

test_boundary_values.py .. [100%]

===== 2 passed in 0.95s =====
(venv) PS C:\Users\rodri\OneDrive\Escritorio\evaluacion\EvaluacionP2\Backend>
```

Figura 31

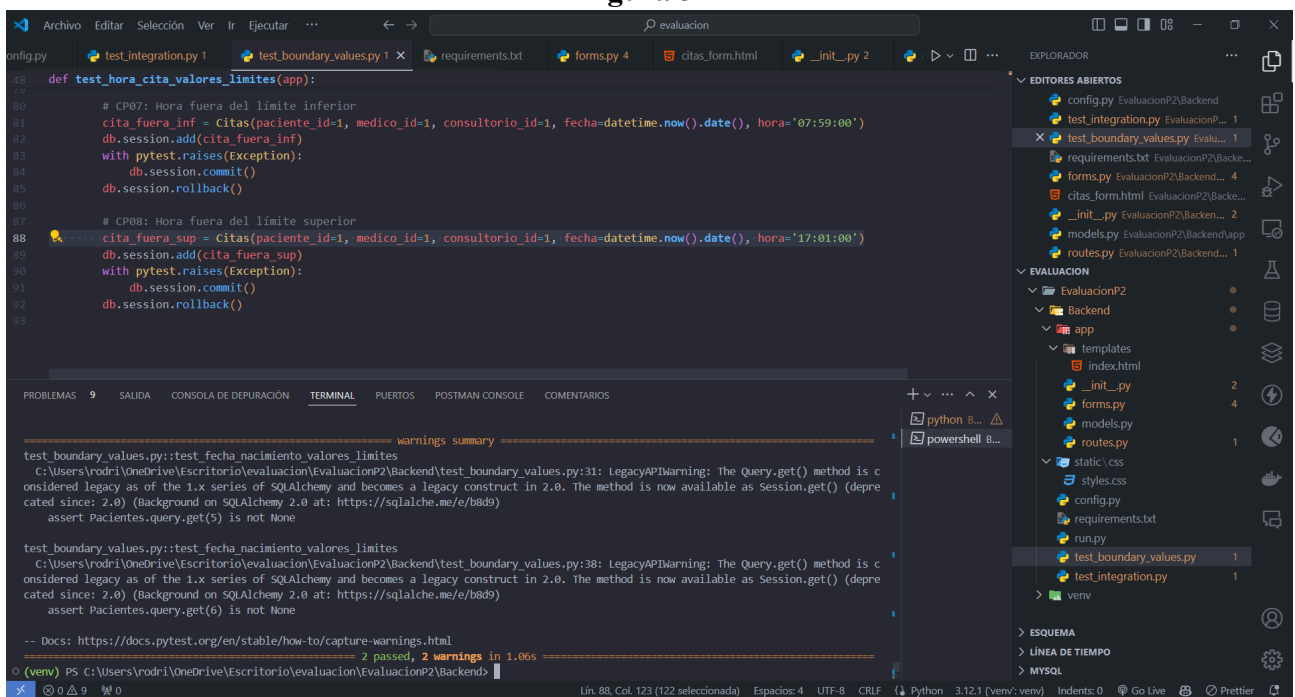


Figura 32

- Identificación de los valores límites

Fecha de nacimiento:

Solo va a ingresar personas que aún no cumplan más de 90 años

Se tomará como guía la fecha de hoy

No pueden ingresar mayores de 90

Hora

Valor mínimo de la hora es a las 8 am que abre el consultorio

El máximo es la hora de cierre a las 5 pm

Está fuera del límite inferior las 7:59 am

Y está fuera del límite superior las 5:01 pm

Diseño de casos de prueba

Casos de prueba para fecha_nacimiento:

CP01: Fecha mínima válida

Datos de prueba	fecha_nacimiento = fecha actual - 90 años
Resultado esperado	Registro insertado correctamente.

Tabla 1

CP02: Fecha máxima válida

Datos de prueba	fecha_nacimiento = fecha actual
Resultado esperado	Registro insertado correctamente.

Tabla 2

CP03: Fecha fuera del límite inferior

Datos de prueba	fecha_nacimiento = fecha actual - 91 años
Resultado esperado	Error de inserción.

Tabla 3

CP04: Fecha fuera del límite superior

Datos de prueba	fecha_nacimiento = fecha actual + 1 día
Resultado esperado	Error de inserción.

Tabla 4

Casos de prueba para hora:

CP05: Hora mínima válida

Datos de prueba	hora = 08:00:00
Resultado esperado	Registro insertado correctamente.

Tabla 5

CP06: Hora máxima válida

Datos de prueba	hora = 17:00:00
Resultado esperado	Registro insertado correctamente.

Tabla 6

CP07: Hora fuera del límite inferior

Datos de prueba	hora = 07:59:00
Resultado esperado	Error de inserción.

Tabla 7

CP08: Hora fuera del límite superior

Datos de prueba	hora = 17:01:00
Resultado esperado	Error de inserción.

Tabla 8

Documentación de los casos de uno se anexara el pdf:

Casos de Prueba para `fecha_nacimiento`

CP01: Fecha mínima válida

- **Datos de prueba:**
 - o `id = 5`
 - o `nombre = 'Test'`
 - o `apellido = 'Min'`
 - o `fecha_nacimiento = fecha actual - 90 años`
 - o `email = 'test.min@example.com'`
- **Resultado esperado:** Registro insertado correctamente.
- **Resultado obtenido:** Registro insertado correctamente. [PASSED]

CP02: Fecha máxima válida

- **Datos de prueba:**
 - o `id = 6`
 - o `nombre = 'Test'`
 - o `apellido = 'Max'`
 - o `fecha_nacimiento = fecha actual`
 - o `email = 'test.max@example.com'`
- **Resultado esperado:** Registro insertado correctamente.
- **Resultado obtenido:** Registro insertado correctamente. [PASSED]

CP03: Fecha fuera del límite inferior

- **Datos de prueba:**
 - o `id = 7`
 - o `nombre = 'Test'`
 - o `apellido = 'FueraInf'`
 - o `fecha_nacimiento = fecha actual - 91 años`
 - o `email = 'test.fuerainf@example.com'`
- **Resultado esperado:** Error de inserción.
- **Resultado obtenido:** Error de inserción. [PASSED]

CP04: Fecha fuera del límite superior

- **Datos de prueba:**

Figura 33

Sección 5:

- **Generación de casos de pruebas generar citas:**

CREACIÓN

Caso de Prueba	Datos de Prueba	Resultado Esperado
CP01: Creación exitosa de una cita	paciente_id = 1, medico_id = 1, consultorio_id = 1, fecha = 2024-07-20, hora = 10:00:00	La cita se crea exitosamente.
CP02: Creación con fecha inválida (pasada)	paciente_id = 1, medico_id = 1, consultorio_id = 1, fecha = 2023-07-20, hora = 10:00:00	Error.
CP03: Creación con hora inválida (fuera de horario)	paciente_id = 1, medico_id = 1, consultorio_id = 1, fecha = 2024-07-20, hora = 07:00:00	Error.

Tabla 9

MODIFICACIÓN

Caso de Prueba	Datos de Prueba	Resultado Esperado
CP04: Modificación exitosa de una cita	cita_id = 1, nuevo_paciente_id = 2, nuevo_medico_id = 2, nuevo_consultorio_id = 2, nueva_fecha = 2024-07-21, nueva_hora = 11:00:00	La cita se modifica exitosamente.
CP05: Modificación con fecha inválida (pasada)	cita_id = 1, nuevo_paciente_id = 2, nuevo_medico_id = 2, nuevo_consultorio_id = 2, nueva_fecha = 2023-07-21, nueva_hora = 11:00:00	Error.
CP06: Modificación con hora inválida (fuera de horario)	cita_id = 1, nuevo_paciente_id = 2, nuevo_medico_id = 2, nuevo_consultorio_id = 2, nueva_fecha = 2024-07-21, nueva_hora = 18:00:00	Error.

Tabla 10

ELIMINAR

Caso de Prueba	Datos de Prueba	Resultado Esperado
CP07: Eliminación exitosa de una cita	cita_id = 1	La cita se elimina exitosamente.
CP08: Eliminación de una cita inexistente	cita_id = 99	Error.

Tabla 11

- Documentar los casos de pruebas y los resultados esperados.

Se documentó con una herramienta, se anexara la sección 5 en pdf:

Casos de Prueba para la Funcionalidad de Agendar Citas Médicas

1. Creación de una cita

CP01: Creación exitosa de una cita

- **Datos de Prueba:**
 - o `paciente_id = 1`
 - o `medico_id = 1`
 - o `consultorio_id = 1`
 - o `fecha = 2024-07-20`
 - o `hora = 10:00:00`
- **Resultado Esperado:** La cita se crea exitosamente.

CP02: Creación con fecha inválida (pasada)

- **Datos de Prueba:**
 - o `paciente_id = 1`
 - o `medico_id = 1`
 - o `consultorio_id = 1`
 - o `fecha = 2023-07-20`
 - o `hora = 10:00:00`
- **Resultado Esperado:** Error.

CP03: Creación con hora inválida (fuera de horario)

- **Datos de Prueba:**
 - o `paciente_id = 1`
 - o `medico_id = 1`
 - o `consultorio_id = 1`
 - o `fecha = 2024-07-20`
 - o `hora = 07:00:00`
- **Resultado Esperado:** Error.

2. Modificación de una cita

CP04: Modificación exitosa de una cita

- **Datos de Prueba:**
 - o `cita_id = 1`
 - o `nuevo_paciente_id = 2`

Figura 34

4. Conclusiones

Se puede observar que en el caso de las pruebas de estrés, al poner 500 generaciones de citas, algunas de las pruebas se pierden y otras salen exitosas debido a la cantidad de datos que las pruebas tienen que generar, haciendo de nuestro sistema en el caso de generación de citas a gran escala, un caso fallido porque se perderán muchas fallas, por lo que hay que optimizar el sistema.

En el caso de la generación de pruebas es una parte fundamental para saber como actúa con los usuarios, por esa razón en la delimitación de límites, edades, horas, se puede ver que en ese caso se observa como funcionaria si un usuario ingresara mal sus datos.

5. Recomendaciones

Se recomienda siempre realizar los casos de prueba durante el ciclo de desarrollo del software, para garantizar el buen funcionamiento del sistema, esto nos ayudará a que se pueda garantizar un buen producto para los clientes, las pruebas de estrés nos ayudan a ver cómo se comporta en el mercado y creo que deberían aplicarse muchas más pruebas.

6. Bibliografía (normas APA)