



UNIVERSIDAD AUTONOMA DE SINALOA



ADMINISTRACIÓN DE PROYECTOS II

GRUPO 4-01 INGENIERIA DE SOFTWARE

PRUEBAS DE SOFTWARE

ALUMNOS:

BECERRA VEGA LUIS ENRIQUE

ESCALANTE SIANUQUI IRVING JARED

FECHA:

25/09/2025

PRUEBAS DE UNIDAD

Utilizado para validar un fragmento de código fuente, aislando unas líneas del lenguaje codificado para saber si el sistema está operando correctamente en una función, proceso o actividad específica.

Pruebas unitarias automatizadas

Pueden ejecutarse múltiples veces mediante una programación previa, ya sea de forma individual o grupal

Pruebas unitarias manuales

Requieren de la intervención y la verificación humana, con mayor susceptibilidad a la omisión de errores, aunque llega a dar poca fiabilidad y velocidad de ejecución

Ventajas

- **Velocidad en su ejecución:** se caracterizan por su velocidad para detectar fallos en los sistemas
- **Reducción de riesgos:** Se aplica en etapas tempranas de desarrollo con el objetivo de prevenir fallos significativos en fases posteriores.
- **Asegura la calidad de tu desarrollo:** De esto depende el buen funcionamiento de los productos informáticos y la satisfacción de los usuarios
- **Integración:** Seccionan las líneas de código en pequeñas partes y contribuyen a que en caso de módulos con mucha complejidad, la integración sea fácil y libre de errores
- **Diseño intuitivo:** Agiliza la labor de diseño de los desarrolladores

```
TS user.spec.ts  back.py  prueba.py  test_prueba.py X
backend > test_prueba.py > test_sum

6  consulta={
7      "nombres": "America Lizbeth",
8      "apellidos": "Vega Fierro",
9      "correo": "Ammeli77@hotmail.com",
10     "contra": "POPOcateptl2012",
11     "confirm_pswd": "POPOcateptl2012"
12 }
13
14
15 op=operaciones()
16 def test_sum():
17     assert op.sum(2,5) == 7
18     assert op.mayorque(2,5) == False
19     assert op.sum(2,5) == 8
20     #for i in range(2000):
21     response=requests.post(url+"/create_user",json=consulta)
22     assert response.status_code == 200

PROBLEMS  DEBUG CONSOLE  OUTPUT  TERMINAL  PORTS
(venv) PS C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend> pytest
===== FAILURES =====
test_sum

def test_sum():
    assert op.sum(2,5) == 7
    assert op.mayorque(2,5) == False
>   assert op.sum(2,5) == 8
E   assert 7 == 8
E   + where 7 = sum(2, 5)
E   +       where sum = <prueba.operaciones object at 0x0000011704387230>.sum

test_prueba.py:19: AssertionError
===== short test summary info =====
FAILED test_prueba.py::test_sum - assert 7 == 8
```

```
TS user.spec.ts  back.py  prueba.py  test_prueba.py X
backend > test_prueba.py > test_sum

6  consulta={
7      "nombres": "America Lizbeth",
8      "apellidos": "Vega Fierro",
9      "correo": "Ammeli77@hotmail.com",
10     "contra": "POPOcateptl2012",
11     "confirm_pswd": "POPOcateptl2012"
12 }
13
14
15 op=operaciones()
16 def test_sum():
17     assert op.sum(2,5) == 7
18     assert op.mayorque(2,5) == False
19     # assert op.sum(2,5) == 8
20     #for i in range(2000):
21     response=requests.post(url+"/create_user",json=consulta)
22     assert response.status_code == 200

PROBLEMS  DEBUG CONSOLE  OUTPUT  TERMINAL  PORTS
(venv) PS C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend> pytest
===== short test summary info =====
1 failed in 0.27s
(venv) PS C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend> pytest
===== test session starts =====
platform win32 -- Python 3.13.3, pytest-8.4.2, pluggy-1.6.0
rootdir: C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend
plugins: anyio-4.10.0
collected 1 item

test_prueba.py . [100%]

===== 1 passed in 0.53s =====
(venv) PS C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend>
```

PRUEBAS DE INTEGRACION

Estas implican probar diferentes módulos de una aplicación de software como grupo, el propósito de este tipo de pruebas es validar la integración de diferentes módulos juntos e identificar los errores y problemas relacionados con ellos

Este tipo se realiza mediante la creación y evaluación de casos de prueba. Siendo su primera fase la identificación los puntos de integración (áreas donde interactúan los módulos), se diseñan casos de prueba en torno a ellos con tal de mostrar cómo funciona cada punto de integración.

Con los datos obtenidos de las pruebas, los stakeholders realizaran ajustes necesarios en la base de código.

Tipos:

- **Descendentes:** Centrado en el modulo principal y su funcionamiento antes de evaluar a continuación los submódulos y subrutinas
- **Ascendentes:** Los submódulos y subrutinas se evalúan primero, para luego pasar al módulo principal
- **Mixta:** Combina los métodos de arriba hacia abajo y de abajo hacia arriba, siendo su principal ventaja el como supera la secuencia de procesos forzadas que limita las pruebas descendentes y ascendentes
- **Big Bang:** Todas las unidades, componentes y módulos individuales se integran y prueban juntos a la vez, como si fuera una sola unidad

CASOS DE PRUEBA

ID caso de prueba	ResumenIA-001
Nombre de caso de prueba	Resumen con IA de historial medico
Descripción	La IA realiza un resumen del historial médico de la mascota
Precondición	Es necesario que la mascota esté registrada y que haya sido atendida anteriormente en la veterinaria.
Relaciones Caso de uso	CU-04-Resumen de historial medico
Pasos y condiciones de CdeU	1-Abrir página del historial medico 2-El sistema consulta su historial médico y le pide a la IA que realice un resumen del historial medico
Resultado esperado	El sistema le muestra el resumen con IA
Estado del caso de prueba	Pendiente de ejecución
Resultado obtenido	Al momento de cargar el historial médico se va escribiendo el resumen con IA
Errores Asociados	Ninguno detectado hasta ahora
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Escalante Sianuqui Irving Jared:Tester
Comentarios	

ID caso de prueba	Cartilla-001
Nombre de caso de prueba	Generador de Cartillas mediante datos consultados de la mascota
Descripción	Generación de cartillas medicas de las mascotas registradas anteriormente en formato PDF
Precondición	La mascota debe haber sido registrada con anterioridad
Relaciones Caso de uso	CU-05-Generar Cartilla
Pasos y condiciones de CdeU	1-Se selecciona el botón de ver cartilla medica 2-El sistema genera el documento en formato PDF y se lo muestra al usuario
Resultado esperado	Se muestra al usuario el documento PDF generado de la cartilla
Estado del caso de prueba	Pendiente de ejecución

Resultado obtenido	La cartilla que fue obtenido en PDF
Errores Asociados	Ninguno detectado hasta ahora
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Escalante Sianuqui Irving Jared:Tester
Comentarios	Contendra: nombre, fecha de nacimiento, color, raza, sexo, dirección, propietario, cuando se registró y por quien, sus vacunas

ID caso de prueba	Estadísticas-001
Nombre de caso de prueba	Generación de estadísticas sobre clientes y atenciones médicas
Descripción	El sistema genera estadísticas gráficas relacionadas con clientes atendidos, enfermedades más comunes y procedimientos aplicados.
Precondición	Debe haber tan siquiera alguna mascota registrada y atendida
Relaciones Caso de uso	CU-06-Estadísticas
Pasos y condiciones de CdeU	<p>1- El usuario selecciona la opción “Estadísticas” en el sistema.</p> <p>2- El sistema consulta la base de datos de historiales médicos, para luego procesar los datos y generar estadísticas de:</p> <ul style="list-style-type: none"> Número de clientes atendidos en un periodo. Enfermedades más comunes diagnosticadas. Procedimientos más aplicados a las mascotas. <p>3- El sistema muestra las estadísticas al usuario en forma de gráficas y tablas.</p>
Resultado esperado	El usuario visualiza correctamente las estadísticas con información precisa, clara y organizada.
Estado del caso de prueba	Pendiente de ejecución
Resultado obtenido	Las gráficas fueron generadas en base a los clientes atendidos, enfermedades

	diagnosticadas y procedimientos más aplicados
Errores Asociados	Ninguno detectado hasta ahora
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Escalante Sianuqui Irving Jared:Tester
Comentarios	

ID caso de prueba	Historial-001
Nombre de caso de prueba	Historial médico de la mascota
Descripción	El veterinario visualizara el historial médico de la mascota
Precondición	Es necesario que la mascota haya sido registrada con anterioridad
Relaciones Caso de uso	CU-07-Historial medico
Pasos y condiciones de CdeU	1-Abrir página del historial medico 2-El sistema consulta su historial medico
Resultado esperado	El sistema le devuelve el historial médico de la mascota
Estado del caso de prueba	Pendiente de ejecución
Resultado obtenido	Cargue el historial médico de la mascota junto con sus datos
Errores Asociados	Ninguno detectado hasta ahora
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Escalante Sianuqui Irving Jared:Tester
Comentarios	

ID caso de prueba	ConsultarCitas-001
Nombre de caso de prueba	Consulta de citas del veterinario
Descripción	El sistema permite al veterinario visualizar las citas que tiene asignadas, mostrando la lista de las citas pendientes con detalles de mascota, cliente, fecha, hora y motivo de la cita
Precondición	El veterinario debe haber iniciado sesión y deben existir citas pendientes
Relaciones Caso de uso	CU-08-Consulta de citas del veterinario
Pasos y condiciones de CdeU	1-El veterinario selecciona “Mis citas pendientes” 2-El sistema consulta todas las citas pendientes asociadas al veterinario 3-El sistema lista las citas pendientes
Resultado esperado	El veterinario visualiza correctamente todas sus citas pendientes con la información necesaria para atenderlas.
Estado del caso de prueba	Pendiente de ejecución
Resultado obtenido	Se muestra un listado de consultas pendientes por atender
Errores Asociados	Ninguno detectado hasta ahora
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Escalante Sianuqui Irving Jared:Tester
Comentarios	

ID caso de prueba	MarcarAsistencia-001
Nombre de caso de prueba	Marcar asistencia de cita del veterinario
Descripción	Permite al veterinario registrar que asistió a una cita programada, actualizando el estado de la cita a “asistida” o “no asistida”
Precondición	El veterinario debe haber iniciado sesión y la cita debe haberse registrado
Relaciones Caso de uso	CU-09-Marcar asistencia de citas
Pasos y condiciones de CdeU	1-El veterinario selecciona la opción “Mis citas pendientes” 2-Se muestra el listado de citas pendientes asignadas al veterinario 3-El veterinario selecciona la cita correspondiente y selecciona “Marcar asistencia” 4-El sistema actualiza el estado de la cita y le notifica al veterinario
Resultado esperado	La cita seleccionada aparece como asistido y una notificación al veterinario
Estado del caso de prueba	Pendiente de ejecución
Resultado obtenido	Al realizar la operación, actualiza la pagina del listado de citas pendientes
Errores Asociados	Ninguno detectado hasta ahora
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Escalante Sianuqui Irving Jared:Tester
Comentarios	

ID caso de prueba	Vacunas-001
Nombre de caso de prueba	Registrar vacunas de la mascota
Descripción	Permite al veterinario registrar la aplicación de vacunas a una mascota
Precondición	La mascota debe estar registrada en el sistema
Relaciones Caso de uso	CU-10-Registrar vacunas
Pasos y condiciones de CdeU	1-El veterinario accede al historial médico de la mascota 2-Selecciona la opción “Registrar vacuna” 3-El veterinario Ingresa los datos de la vacuna y presiona guardar 4-El sistema registrara la vacuna
Resultado esperado	La vacuna se guardará correctamente en la base de datos
Estado del caso de prueba	Pendiente de ejecución
Resultado obtenido	La vacuna se mostrará como parte del historial médico y se mostrara en la cartilla de la mascota
Errores Asociados	Ninguno detectado hasta ahora
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Escalante Sianuqui Irving Jared:Tester
Comentarios	Se indicará el tipo de vacuna y fecha de aplicación

ID caso de prueba	Desparasitar-001
Nombre de caso de prueba	Registrar desparasitación de la mascota
Descripción	Permite al veterinario registrar la aplicación de un desparasitante a una mascota
Precondición	La mascota debe estar registrada en el sistema y el veterinario debe haber iniciado sesión
Relaciones Caso de uso	CU-11-Registrar desparasitación
Pasos y condiciones de CdeU	1-El veterinario accede al historial medico de la mascota 2-Selecciona “Registrar desparasitación” 3-Ingresa los datos de la aplicación y lo guarda

	4-El sistema registrara la desparasitación
Resultado esperado	La desparasitación se guardará correctamente en la base de datos
Estado del caso de prueba	Pendiente de ejecución
Resultado obtenido	La desparasitación se mostrará como parte del historial médico y se mostrara en la cartilla de la mascota
Errores Asociados	Ninguno detectado hasta ahora
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Escalante Sianuqui Irving Jared:Tester
Comentarios	Se guardara la fecha de la aplicación y cuando será la proxima Aplicación

ID caso de prueba	PROGRAMACIONCITA-001
Nombre de caso de prueba	Programar cita veterinaria exitosa
Descripcion	Se valida que el cliente pueda programar una cita con el veterinario seleccionando fecha, hora, mascota, el servicio que desea y veterinario.
Precondicion	<ol style="list-style-type: none"> 1- El cliente debe tener una cuenta registrada y activa en el sistema. 2- La mascota ya está registrada en la cuenta del cliente. 3- Existe disponibilidad de horarios en la agenda del veterinario. 4- El cliente ha iniciado sesión.
Relaciones Caso de uso	CU-12 Programar cita veterinaria
Pasos y condiciones de CdeU	<ol style="list-style-type: none"> 1- El cliente inicia sesión en el sistema. 2- Accede al módulo de “Agendar cita veterinaria”. 3- Selecciona la mascota para la cita. 4- Selecciona la fecha y hora disponible en el calendario. 5- Selecciona el servicio que desea obtener. 6- Selecciona el veterinario que desea que le proporcione el servicio. 7- Confirma la programación de la cita.
Resultado esperado	1- El sistema registra la cita con los datos ingresados.

	2- Muestra mensaje de confirmación: “Su cita ha sido programada exitosamente”. 3- Se genera un folio único de cita. 4- La cita aparece en el historial de citas del cliente.
Estado del caso de prueba	Pendiente de ejecución.
Resultado obtenido	La cita se registró correctamente y se mostró el mensaje de confirmación.
Errores Asociados	Ninguno detectado hasta ahora.
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Irving Jared Escalante Sianuqui: Tester
Comentarios	1- Validar que el calendario no permita seleccionar fechas pasadas. 2- Verificar que solo se muestren los horarios libres del veterinario.
ID caso de prueba	Registromascota-001
Nombre de caso de prueba	Registro exitoso de mascota
Descripción	Validar que el cliente pueda registrar correctamente una mascota en su cuenta proporcionando la información requerida.
Precondición	1- El cliente debe tener una cuenta registrada y activa en el sistema. 2- El cliente ha iniciado sesión.
Relaciones Caso de uso	CU-13 Registrar mascota
Pasos y condiciones de CdeU	1- El cliente accede al módulo “Mis mascotas”. 2- Selecciona la opción “Registrar nueva mascota”. 3- Ingresa los datos obligatorios: nombre, especie, raza, edad, sexo y peso. 4- (Opcional) Adjunta fotografía de la mascota. 5- Confirma el registro.
Resultado esperado	El sistema guarda los datos de la mascota, se muestra el mensaje de confirmación “La mascota se ha registrado correctamente” y la mascota aparece en la lista de mascotas del cliente.
Estado del caso de prueba	Pendiente de ejecución.

Resultado obtenido	La mascota se registra correctamente.
Errores Asociados	Ninguno detectado hasta ahora.
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Irving Jared Escalante Sianuqui: Tester
Comentarios	1- Validar que no se permita registrar mascota sin nombre. 2- Verificar que los campos numéricos (edad, peso) solo acepten valores válidos. 3- Comprobar que no se dupliquen mascotas con el mismo nombre y misma especie en la misma cuenta.

ID caso de prueba	Edicionmascota-001
Nombre de caso de prueba	Edición exitosa de datos de mascota
Descripcion	Validar que el cliente pueda editar correctamente los datos de una mascota previamente registrada en su cuenta.
Precondicion	1- El cliente debe tener una cuenta registrada y activa en el sistema. 2- La mascota ya esta registrada en el sistema. 3- El cliente ha iniciado sesión.
Relaciones Caso de uso	CU-14 Editar mascota
Pasos y condiciones de CdeU	1- El cliente accede al módulo “Mis mascotas”. 2- Selecciona la mascota que desea editar. 3- Presiona la opción “Editar”. 4- Modifica los campos deseados (peso, edad, foto o nombre, raza, sexo, especie). 5- Confirma los cambios realizados.
Resultado esperado	El sistema actualiza los datos de la mascota y se muestra mensaje de confirmación: “Los datos de la mascota han sido actualizados exitosamente”. Los cambios se reflejan en la vista de detalles de la mascota.
Estado del caso de prueba	Pendiente de ejecución.
Resultado obtenido	El sistema actualizo correctamente los datos.
Errores Asociados	Ninguno detectado hasta ahora.
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Irving Jared Escalante Sianuqui: Tester
Comentarios	1- Validar que no se dejen los campos obligatorios en blanco durante la edición. 2- Verificar que las modificaciones se reflejen en consultas posteriores (ej. al programar una cita).

ID caso de prueba	Historialcitas-001
Nombre de caso de prueba	Consulta exitosa del historial de citas realizadas.
Descripción	Verificar la visualización y detalle del historial de citas realizadas del cliente, mostrando información detallada de la misma.
Precondición	<ol style="list-style-type: none"> 1- El cliente tiene que tener una cuenta registrada y activa en el sistema. 2- El cliente ha iniciado sesión. 3- El cliente tiene al menos una cita realizada previamente.
Relaciones Caso de uso	CU-15 Consultar historial de citas
Pasos y condiciones de CdeU	<ol style="list-style-type: none"> 1- El cliente accede al módulo “Historial de citas”. 2- El sistema consulta la base de datos de citas del cliente. 3- El sistema muestra una lista de citas realizadas, ordenadas por fecha (más recientes primero). 4- El cliente selecciona una cita específica para ver detalles (mascota, fecha, hora, servicio, diagnóstico, veterinario asignado, etc).
Resultado esperado	El historial de citas se muestra con todas las citas realizadas por el cliente.
Estado del caso de prueba	Pendiente de ejecución.
Resultado obtenido	El historial muestra correctamente las citas realizadas por el cliente y permite ver detalles.
Errores Asociados	Ninguno detectado hasta ahora.
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Irving Jared Escalante Sianuqui: Tester
Comentarios	<ol style="list-style-type: none"> 1- Validar que, si el cliente no tiene citas, el sistema muestre el mensaje: “No se encontraron citas en el historial”. 2- Revisar que el historial solo muestre citas del cliente autenticado. 3- Verificar que la información de citas canceladas aparezca con un estatus adecuado (ej. cancelada).

ID caso de prueba	Recomendación cuidados-001
Nombre de caso de prueba	Generación exitosa de recomendaciones de cuidados según estado médico de la mascota
Descripción	Validar que el sistema muestre recomendaciones de cuidados apropiadas según el diagnóstico o estado médico registrado de la mascota.
Precondición	<ol style="list-style-type: none"> 1- El cliente tiene una cuenta activa en el sistema. 2- La mascota está registrada en la cuenta del cliente. 3- La mascota tiene al menos un diagnóstico médico registrado por el veterinario (ejemplo: sobrepeso, alergia, vacunación pendiente, enfermedad específica).
Relaciones Caso de uso	CU-16 Recomendación de cuidados
Pasos y condiciones de CdeU	<ol style="list-style-type: none"> 1- El cliente accede al perfil de la mascota. 2- Selecciona la opción "Recomendaciones de cuidados". 3- El sistema consulta el estado médico más reciente de la mascota. 4- El sistema genera y muestra las recomendaciones correspondientes (ejemplo: dieta especial, ejercicio moderado, medicamentos, próximas vacunas).
Resultado esperado	<ol style="list-style-type: none"> 1- El sistema muestra un listado de recomendaciones personalizadas con base en el estado médico de la mascota. 2- El cliente puede visualizar consejos prácticos y recordatorios (ej. próximas vacunas o revisiones) y la información es clara, completa y está relacionada con el diagnóstico médico.
Estado del caso de prueba	Pendiente de ejecución

Resultado obtenido	El sistema mostró recomendaciones correctas en base al último diagnóstico de la mascota.
Errores Asociados	Ninguno detectado hasta ahora.
Responsable de diseño	Luis Enrique Becerra Vega: Diseñador
Responsable de ejecución	Irving Jared Escalante Sianuqui: Tester
Comentarios	<ol style="list-style-type: none"> 1. Validar que, si no existe estado médico registrado, el sistema muestre el mensaje: <i>“No hay recomendaciones disponibles, acuda a consulta veterinaria”</i>. 2. Verificar que las recomendaciones sean coherentes con el diagnóstico más reciente. 3. Revisar que se actualicen automáticamente cuando el veterinario registre un nuevo estado médico.

PERFORMANCE TESTING

Son un tipo de testeo de software no funcional, enfocado en el mejoramiento de la experiencia de los usuarios y el incremento de la calidad de los productos informáticos.

Consiste en evaluar aspectos del rendimiento de los sistemas como su velocidad, tiempo de respuesta y la estabilidad bajo distintas cargas de trabajo. Por lo general se aplican después de las pruebas funcionales cuando se comprobó que las funciones se comportan como lo esperado.

Tipos:

- **Pruebas de Carga:** Verifican que el sistema funcione óptimamente bajo cargas de trabajo normales, de esta forma se comprueba si se cuenta con la capacidad necesaria para cumplir con las funciones básicas
- **Prueba de estrés:** Este somete al sistema a cargas máximas para medir su capacidad de respuesta frente a situaciones críticas
- **Pruebas de resistencia:** Se encargan de evaluar la capacidad de un software al encontrarse en condiciones críticas bajo distintos escenarios
- **Pruebas de escalabilidad:** Verifican el comportamiento de los sistemas al someterse a cargas superiores a lo esperado, con el objetivo de evitar su colapso ante un incremento de uso durante situaciones reales

```
TS user.spects  prueba.py  back.py  test_prueba.py X
backend > test_prueba.py > ...
1  import requests
2  from prueba import operaciones
3
4  url="http://127.0.0.1:8000"
5
6  consulta={
7      "nombres": "America Lizbeth",
8      "apellidos": "Vega Fierrro",
9      "correo": "Ammeli77@hotmail.com",
10     "contra": "POPOcateptl2012",
11     "confirm_pswd": "POPOcateptl2012"
12 }
13
14 for i in range(8000):
15     response=requests.post(url+"/create_user",json=consulta)
16     assert response.status_code == 200
17 #op=operaciones()
18 #def test_sum():

PROBLEMS  DEBUG CONSOLE  OUTPUT  TERMINAL  PORTS

(venv) PS C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend> pytest
ERROR test_prueba.py - requests.exceptions.ConnectionError: ('Connection aborted.', ConnectionResetError(10054, 'Se ha forzado la interrupci n de una conexi n ...
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Interrupted: 1 error during collection !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
===== 1 error in 2227.86s (0:37:07) =====
(venv) PS C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend> pytest
===== test session starts =====
platform win32 -- Python 3.13.3, pytest-8.4.2, pluggy-1.6.0
rootdir: C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend
plugins: anyio-4.10.0
collected 0 items

===== no tests ran in 3579.73s (0:59:39) =====
(venv) PS C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend> 
```

```
S user.spects  prueba.py  back.py  test_prueba.py X
backend > test_prueba.py > test_sum
10     "contra": "POPOcateptl2012",
11     "confirm_pswd": "POPOcateptl2012"
12 }
13
14
15 op=operaciones()
16 def test_sum():
17     assert op.sum(2,5) == 7
18     for i in range(2000):
19         response=requests.post(url+"/create_user",json=consulta)
20         assert response.status_code == 200
21 #     assert op.sum(2,5) == 7
22 #     assert op.sum(3,5) == 8
23 #     assert op.sum(9,10) == 14

PROBLEMS  DEBUG CONSOLE  OUTPUT  TERMINAL  PORTS

(venv) PS C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend> pytest
(to show a full traceback on KeyboardInterrupt use --full-trace)
===== no tests ran in 7.17s =====
(venv) PS C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend> pytest
===== test session starts =====
platform win32 -- Python 3.13.3, pytest-8.4.2, pluggy-1.6.0
rootdir: C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend
plugins: anyio-4.10.0
collected 1 item

test_prueba.py . [100%]

1 passed in 786.51s (0:13:06)
(venv) PS C:\Users\lbece\OneDrive\Documentos\GitHub\Parroquia\backend> 
```