

Guia de Aprendizaje Pruebas de Software
Módulo Registro de Usuarios

Desarrollado por:
Jeiffer Harley Mosquera Abadia
Yeraldin Olaya Ríos

Septiembre 2025
Instructor: Jonathan
Espitia

SENA

Centro de Gestión de Mercados, Logística y TIC (CGMLTI)
Análisis y Desarrollo de Software (ADSO)

Ficha: 3147246
Bogotá D.C

IV. Planteamiento de Evidencias de Aprendizaje

- **Fase del Proyecto Formativo:** Construcción del software
- **Actividad del Proyecto Formativo:** Aplicación de pruebas funcionales al módulo de gestión del sistema desarrollado

1. **Nombre del Módulo:** Registro de usuarios para Iniciar Sesión en la plataforma WaveSound.

2.1 Desarrollador: Yeraldin Olaya

2.2 Cliente: Jeiffer Mosquera

2. **Enfoque:** Se debe de implementar el Registro de forma que :

3.1 Register:

- En el campo Nombre solo se admiten letras mayúsculas y minúsculas, con espacios y es obligatorio
- En el campo Usuario se admiten letras, números, . y _ (sin espacios) es obligatorio
- En el campo Correo Electrónico Se verifica que tenga el @ para poder ser admitido y es obligatorio .
- En el campo Rol se puede seleccionar solo uno y es obligatorio
- En el campo Contraseña debe tener mínimo 8 a 10 caracteres, se admite todo símbolos, letras, números etc, no se muestra la contraseña solo unos puntos y es obligatorio.

3.3 Requisitos Funcionales:

3.3.1 Registro de Usuario:

RF-01. Campo Nombre

- El sistema debe permitir ingresar únicamente letras mayúsculas y minúsculas, además de espacios.
- El campo es obligatorio.
- Si el usuario ingresa caracteres no válidos (números o símbolos), el sistema debe mostrar un mensaje de error.

RF-02. Campo Usuario (nickname)

- El sistema debe permitir ingresar letras, números, punto (.) y guión bajo (_).
- No se permiten espacios.
- El campo es obligatorio.

- Si se ingresan caracteres diferentes a los permitidos, el sistema debe mostrar un mensaje de error.

RF-03. Campo Correo Electrónico

- El sistema debe validar que el correo tenga el formato correcto (incluyendo @).
- El campo es obligatorio.
- Si no cumple con el formato, se debe mostrar un mensaje de error.

RF-04. Campo Rol

- El sistema debe mostrar una lista de roles disponibles.
- El usuario debe poder seleccionar solo un rol.
- El campo es obligatorio.
- Si no se selecciona un rol, se debe mostrar un mensaje de error.

RF-05. Campo Contraseña

- La contraseña debe tener entre 8 y 10 caracteres.
- Puede contener letras, números y símbolos.
- El campo es obligatorio.
- La contraseña no debe mostrarse en texto plano, sino enmascarada (puntos o asteriscos).
- Si no cumple con los requisitos, el sistema debe mostrar un mensaje de error.

3. Casos de Uso o Historias de Usuario:

4.1HU02-Como productor musical, quiero crear mi cuenta ingresando mi nombre, contraseña y rol (productor, artista, músico, oyente), para tener acceso a las funciones específicas según mi perfil.

4. Descripción:

5.1. Pruebas funcionales/manuales (Pruebas Unitarias): Se realizaron las pruebas para el módulo de Registro, donde se probaron y verificaron que los datos que se insertan en el formulario de registro son correctamente almacenados en la Base de Datos por medio de unas APIS , utilizando pytest para probar.

5.2 Pruebas Pytest para Python:

Se configuro un cliente de prueba (Test Client)

Esto sirve para simular peticiones HTTP (GET, POST, etc.) a la API de Fast API, sin necesidad de tener el servidor corriendo en un navegador.

se usó fixtures

Por ejemplo, mock db → un objeto falso (MagicJack) que simula la base de datos.

Esto evita tocar la base real y permite controlar lo que devuelve las funciones.

se usó monkey patch

Con esto reemplazamos funciones reales por versiones falsas.

Ejemplo: usuarios.registrar usuario fue reemplazado por una función que devuelve un usuario simulado.

Así se controla exactamente el resultado y se puede comprobar la respuesta de la API.

Se probó 2 endpoints principales

/usuarios/register: simula el registro de un usuario y verifica que la API devolviera el nombre correcto.

/usuarios/login: simulas un inicio de sesión, forzando que devuelva un fake token y comprueba que aparezca en la respuesta.

5. **Nombre de las Pruebas:** Pruebas funcionales/manuales(Pruebas Unitarias) y Pruebas de Integración Pytest para Python.

6. **Resultado:** Se realizaron pruebas sobre el módulo de Registro, verificando que los datos ingresados en el formulario se almacenarán correctamente en la Base de Datos a través de las API. Usando pytest, se simuló el envío de los datos al endpoint de registro y se confirmó que el sistema responde correctamente, cumpliendo los criterios de aceptación y asegurando que la creación de usuarios funciona de manera confiable.

7. Capturas de las pruebas:

```
Aprendiz@BOGDFPCMP5XXX MINGW64 ~/Olaya_3147246/WAVESOUND_JY/backend (develop)
$ pytest

===== test session starts =====
platform win32 -- Python 3.13.2, pytest-8.4.2, pluggy-1.6.0
rootdir: C:\Users\Aprendiz\Olaya_3147246\WAVESOUND_JY\backend
plugins: anyio-4.10.0, asyncio-1.2.0
asyncio: mode=Mode.STRICT, debug=False, asyncio_default_fixture_loop_scope=None,
        asyncio_default_test_loop_scope=function
collected 2 items

tests\test_routes.py ..                                         [100%]

===== warnings summary =====
mi_entorno\Lib\site-packages\pydantic\_internal\_config.py:323
  C:\Users\Aprendiz\Olaya_3147246\WAVESOUND_JY\backend\mi_entorno\Lib\site-packa
ges\pydantic\_internal\_config.py:323: PydanticDeprecatedSince20: Support for cl
ass-based 'config' is deprecated, use ConfigDict instead. Deprecated in Pydantic
 V2.0 to be removed in V3.0. See Pydantic V2 Migration Guide at https://errors.p
ydantic.dev/2.11/migration/
    warnings.warn(DEPRECATION_MESSAGE, DeprecationWarning)

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
===== 2 passed, 1 warning in 2.17s =====
(mi_entorno)
```

8. Link del repositorio: https://github.com/olaya19/Pruebas_Software