

Proyecto QA Integral

1. Información General

Nombre: Alejandro Franco Acosta

Rol: QA Tester

Periodo: Sprint 1 – Sprint 2

Herramientas:

- Jira/Xray
- Python
- Selenium
- Pytest
- JUnit
- XML
- MongoDB
- Mongo Compass
- SoapUI
- Git y GitHub
- Docker
- DOM (Document Object Model)

2. Introducción

El presente informe documenta el trabajo realizado durante el Proyecto QA Integral, cuyo objetivo es demostrar mis conocimientos aplicados al aseguramiento de calidad sobre distintas capas de un sistema, combinando pruebas manuales y automatizadas. El proyecto abarcó validaciones a nivel de API, Base de Datos y UI, manteniendo trazabilidad completa mediante Jira/Xray que es nuestra herramienta de gestión clave para mantener el orden y la planeación.

3. Alcance del Proyecto

Las pruebas realizadas cubren las siguientes áreas:

- API: Validación de servicios SOAP mediante WSDL.
- Base de Datos: Validaciones sobre MongoDB (estructura, consistencia y reglas básicas).
- UI: Automatización del flujo de login en aplicación web.
- Gestión QA: Registro, ejecución y trazabilidad de pruebas en Jira/Xray.

Fuera de alcance:

- Pruebas de performance
- Pruebas de seguridad
- Integración CI/CD

4. Pruebas de API (SOAP)

4.1 Descripción

Se validó un servicio SOAP público utilizando SoapUI. A partir del WSDL se generaron operaciones y se crearon Test Suites y Test Cases para validar:

- Disponibilidad del servicio
- Código de respuesta HTTP
- Estructura del mensaje SOAP
- Resultado esperado de la operación

4.2 Evidencias

Request y Response de API en SoapUI (open source)

The screenshot shows the SoapUI interface with the following details:

- File Bar:** File, Project, Suite, Case, Step, Tools, Desktop, Help.
- Toolbar:** Empty, SOAP, REST, Import, Save All, Forum, Trial, Preferences, Proxy, Endpoint Explorer, Search Forum, Online Help.
- Navigator:** Projects section shows QA_SOAP_CALCULATOR with CalculatorSoap, Add, Divide, Multiply, Subtract, and CalculatorSoap12.
- Request View:** Request 1 is selected, showing the XML structure of a division request. The XML includes the envelope, header, body, and a specific `<tem:divide>` element with two integer parameters.
- Response View:** Shows the XML structure of the response, which includes the envelope, body, and a `<DivideResult>` element containing the result of the division.
- Properties View:** Request Properties table shows Name: Request 1, Description: 298, Message Size: 298, Encoding: UTF-8, and Endpoint: http://www.dneonline.com/calculator.asmx.
- Bottom Log:** Shows log entries for SoapUI log, http log, jetty log, error log, wsrm log, and memory log.

Test Suite, Test Case y Assertions(contains)

The screenshot shows the SoapUI interface with the following details:

- File Bar:** File, Project, Suite, Case, Step, Tools, Desktop, Help.
- Toolbar:** Empty, SOAP, REST, Import, Save All, Forum, Trial, Preferences, Proxy, Endpoint Explorer, Search Forum, Online Help.
- Navigator:** Projects section shows QA_SOAP_CALCULATOR with CalculatorSoap, Add, Divide, Multiply, Subtract, Request 1, and CalculatorSoap12.
- Test Case View:** Test Case 003 is selected, showing a TestStep named "SOAP Request 003 - División".
- Request View:** Shows the XML structure of the division request, identical to the one in the previous screenshot.
- Response View:** Shows the XML structure of the response, identical to the one in the previous screenshot.
- Assertions View:** A validation summary indicates "SOAP Response - VALID" and "Valid HTTP Status Codes - VALID".
- Properties View:** TestRequest Properties table shows Name: SOAP Request 003, Description: 279, Message Size: 279, Encoding: UTF-8, Endpoint: http://www.dneonline.com/calculator.asmx, and Timeout: 5000.
- Bottom Log:** Shows log entries for TestCase Log.

Se registra evidencias en Jira

The screenshot shows a Jira story card for '1.2 Pruebas de Servicios SOAP'. The story has a status of 'Done' and is assigned to 'alejandro franco'. It was created by 'alejandro franco' and has a due date of 'Dec 16, 2025'. The story is part of a sprint starting on 'Dec 15, 2025'. There are five subtasks listed under 'Subtasks':

| Work | Pri... | Stor... | As... | Status |
|-----------------------------------------------------------|--------|---------|-------|--------|
| QA2-9 1.2.1 Importar un WSDL en SoapUI | = M | AF | DONE | |
| QA2-10 1.2.2 Crear test suite + test case | = M | AF | DONE | |
| QA2-11 1.2.3 Crear assertions(status, XML output, reglas) | = M | AF | DONE | |
| QA2-12 1.2.4 Exportar evidencias (XML + capturas) | = M | AF | DONE | |
| QA2-13 1.2.5 Registrar caso de prueba manual en Xray | = M | AF | DONE | |

4.3 Conocimientos aplicados

- Lectura e interpretación de WSDL
- Estructura de mensajes SOAP
- Uso de assertions funcionales
- Organización de pruebas en suites y casos

4.4 Recomendaciones

- Versionar los proyectos SoapUI en el repositorio
- Separar datos de prueba cuando el servicio lo permita
- Documentar claramente los servicios probados

5. Pruebas de Base de Datos (MongoDB)

5.1 Descripción

Se crea una base de datos de prueba en MongoDB para validar reglas básicas de integridad y consistencia de datos. Las validaciones se realizaron tanto de forma manual (MongoDB Compass) como automatizada mediante scripts en Python usando PyMongo.

Las validaciones incluyeron:

- Existencia de registros
- Validación de usuarios activos
- Validación de formato de emails
- Verificación de campos obligatorios

5.2 Evidencias

Creación de base de datos local con MongoDB que aprovecha el modelado flexible de documentos NoSQL. Visualización en Mongo Compass

The screenshot shows the MongoDB Compass application interface. The left sidebar lists connections and databases, with 'qa_testing' selected. The main area shows the 'test_usuarios' collection within the 'qa_testing' database. The interface includes a search bar, document list, and various management buttons like 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'. Three documents are visible in the list:

- `_id: ObjectId('6034ed0011d09fe054e89398')`
nombre: "Luis Rojas"
email: "luis@testing.com"
rol: "User"
activo: false
- `_id: ObjectId('6034ed0011d09fe054e89397')`
edad: 35
profesion: "Alfarero"
- `_id: ObjectId('6044c3abada7585672ce9579')`
nombre: "Alejandro Franco"
email: "alejandro@gmail.com"
rol: "Admin"
activo: true
telefono: "+5997654321"
permisos: [0, 1]
habilidades: [Array (4)]
proyectos_asignados: [Array (2)]
certificaciones: [Array (2)]
ultimo_acceso: "2020-11-23T10:30:00Z"

Ejecución de script para confirmar la existencia de usuarios activos en DDBB

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows the project structure under "PROYECTO-QA-INTEGRAL". The "validar_usuarios.py" file is selected.
- Terminal:** Displays the command "python .\db\mongo\validar_usuarios.py" being run, followed by the output:

```
(venv) PS C:\QA Tester\proyecto-qa-integral> python .\db\mongo\validar_usuarios.py
● Iniciando validaciones MongoDB...
✓ Existen usuarios activos: 3 encontrado(s)
✓ Validación MongoDB finalizada
```

5.3 Conocimientos aplicados

- Uso de MongoDB Compass
- Modelado flexible de documentos.
- Conexión a MongoDB con PyMongo
- Validaciones lógicas mediante scripts

5.4 Recomendaciones

- Utilizar MongoDB en Docker para mayor portabilidad del proyecto
- Definir validaciones de esquema cuando el proyecto crezca
- Separar claramente datos de prueba y datos productivos

6. Pruebas UI Automatizadas (Selenium + Pytest)

6.1 Descripción

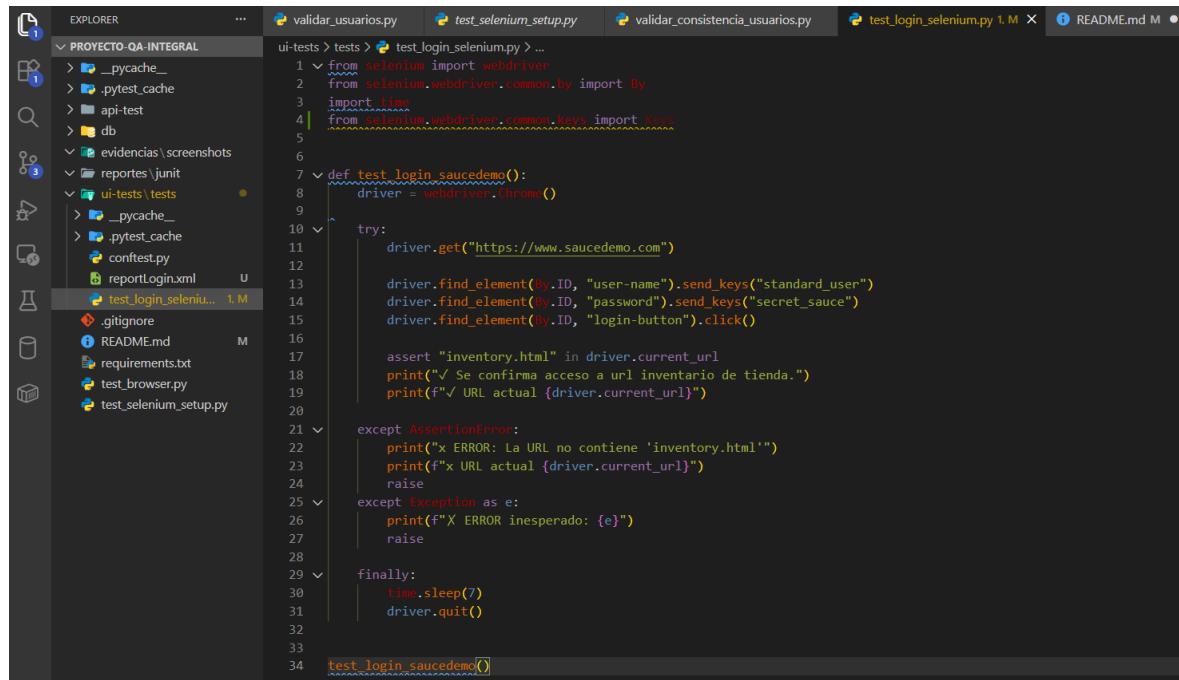
Se automatizó el flujo de login de una aplicación web utilizando Selenium WebDriver con Python y Pytest. El objetivo fue validar el acceso exitoso y comprobar la correcta redirección posterior al login.

El script incluye:

- Inicialización del navegador
- Interacción con elementos UI - DOM
- Aserciones funcionales
- Manejo básico de errores
- Captura de evidencias

6.2 Evidencias

Script de automatización para login



The screenshot shows a code editor with a dark theme. On the left is the Explorer sidebar showing project files like `PROYECTO-QA-INTEGRAL`, `ui-tests`, and `test_login_selenium.py`. The main pane displays the content of `test_login_selenium.py`:

```
1  from selenium import webdriver
2  from selenium.webdriver.common.by import By
3  import time
4  from selenium.webdriver.common.keys import Keys
5
6
7  def test_login_saucedemo():
8      driver = webdriver.Chrome()
9
10     try:
11         driver.get("https://www.saucedemo.com")
12
13         driver.find_element(By.ID, "user-name").send_keys("standard_user")
14         driver.find_element(By.ID, "password").send_keys("secret_sauce")
15         driver.find_element(By.ID, "login-button").click()
16
17         assert "inventory.html" in driver.current_url
18         print("✓ Se confirma acceso a url inventario de tienda.")
19         print(f"✓ URL actual {driver.current_url}")
20
21     except AssertionError:
22         print("✗ ERROR: La URL no contiene 'inventory.html'")
23         print(f"✗ URL actual {driver.current_url}")
24         raise
25     except Exception as e:
26         print(f"✗ ERROR inesperado: {e}")
27         raise
28
29
30     finally:
31         time.sleep(7)
32         driver.quit()
33
34 test_login_saucedemo()
```

Captura de acceso a página con login ejecutado con script

The screenshot shows a web browser window with the title "Swag Labs". The address bar indicates the URL is "saucedemo.com/inventory.html". A message at the top of the page reads "Un software automatizado de pruebas está controlando Chrome.". The main content area is titled "Products" and features three items:

- Sauce Labs Backpack**: \$29.99. Description: "carry.allTheThings() with the sleek, streamlined Sly Pack that melds uncompromising style with unequaled laptop and tablet protection." An "Add to cart" button is present.
- Sauce Labs Bike Light**: \$9.99. Description: "A red light isn't the desired state in testing but it sure helps when riding your bike at night. Water-resistant with 3 lighting modes, 1 AAA battery included." An "Add to cart" button is present.
- Sauce Labs Bolt T-Shirt**: \$15.99. Description: "Get your testing superhero on with the Sauce Labs bolt T-shirt. From American Apparel, 100% ringspun combed cotton, heather gray with red bolt." An "Add to cart" button is present.

Resultado usando Pytest en línea de comando

```
● (venv) PS C:\QA Tester\proyecto-qa-integral> pytest .\ui-tests\tests\test_login_selenium.py -v
=====
platform win32 -- Python 3.13.3, pytest-9.0.2, pluggy-1.6.0 -- C:\venv_qa\venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\QA Tester\proyecto-qa-integral
collected 1 item

ui-tests\tests\test_login_selenium.py::test_login_saucedemo PASSED [100%]

=====
● (venv) PS C:\QA Tester\proyecto-qa-integral> []
```

Documento Junit

```
ui-tests > tests > 📄 reporteLogin.xml > ⚡ xml
1   <?xml version="1.0" encoding="utf-8"?>
2   <testsuites name="pytest tests">
3       <testsuite name="pytest" errors="0" failures="0" skipped="0" tests="1" time="23.440"
4           | timestamp="2025-12-22T11:54:29.118419-03:00" hostname="SCL076PCAD005">
5               <testcase classname="test_login_selenium" name="test_login_saucedemo" time="11.555" />
6           | </testsuite>
7       </testsuites>
```

6.3 Conocimientos aplicados

- Automatización UI con Selenium
- Uso de selectores - DOM
- Aserciones con Pytest
- Estructuración de pruebas automatizadas

6.4 Recomendaciones

- Implementar Page Object Model para mayor mantenibilidad
- Externalizar credenciales y datos sensibles
- Integrar ejecución en pipelines CI/CD

7. Gestión de Pruebas con Jira y Xray

7.1 Descripción

Se utilizó Jira con el complemento Xray para:

- Registrar casos de prueba manuales
- Registrar pruebas automatizadas
- Crear Test Plans y Test Executions
- Importar resultados automatizados vía XML (formato JUnit)
- Mantener trazabilidad entre tareas, tests y ejecuciones

7.2 Evidencias

Creación de Test con pasos detallados

The screenshot shows a test case titled "Souce Demo - Login exitoso - Usuario Válido". The description states: "Probar login exitoso de saucedemo.com con credenciales válidas." Below the description is a screenshot of a browser window showing a login form. The browser title is "Swag Labs". The login page has fields for "Username" and "Password", and a green "Login" button. Below the form, a message says "Accepted usernames are:" followed by a list of user names: "standard_user", "locked_out_user", "problem_user", "performance_glitch_user", "customer_service", and "visual_user". To the right of the browser window, there is a sidebar with "To Do" and "Improve Test" buttons. The main sidebar is titled "Details" and contains the following fields:

| | |
|-------------|--------------------------------|
| Assignee | alejandro franco |
| Reporter | alejandro franco |
| Developer | Create branch Create commit |
| Labels | None |
| Due date | None |
| Start date | None |
| Priority | Medium |
| Test Status | Open Test Status |

At the bottom of the sidebar, there is a "More fields" link.

Estructurar pasos de forma detallada y ordenada

The screenshot shows a "Test details" interface with tabs for "Test details", "Preconditions", "Test Sets", "Test Plans", and "Test Runs". The "Test details" tab is selected. On the left, there is a "Test Type" dropdown set to "Manual". On the right, there are buttons for "Test Repository" and "Dataset". Below these are buttons for "Edit in Dialog", search, and "Add Step".

The main area displays a sequence of five numbered steps:

- 1 Abrir navegador
- 2 Ingresar username
- 3 Ingresar password
- 4 Click en login
- 5 Validar pantalla principal

Each step has a "Click to expand" button next to it. At the bottom right, there are "New Step" and "Call Test" buttons.

Se crea Test Execution y agrega Test Manual con sus pasos para su ejecución

The screenshot shows a Jira QA board interface. At the top, there are navigation links for Spaces, Quality Assurance 2, QA2 board, Summary, Timeline, Backlog, Active sprints, Calendar, Reports, List, Forms, All work, Components, Development, Code, Releases, Archived work items, Pages, Testing Board, and More. The Testing Board tab is selected.

The main area displays a test execution titled "Sauce Demo - Login exitoso - Usuario Válido". The execution status is "PASSED". Key details include:

- Execution Status:** PASSED
- Started On:** 26/Dec/2025 12:38 PM
- Assignee:** Unassigned
- Versions:** v1
- Test Version:** v1
- Timer:** 00:00:12
- Finished On:** 26/Dec/2025 12:38 PM
- Executed By:** alejandro franco
- Revision:** -
- Test Environments:** -
- No time logged**

Below the summary, there are tabs for Findings and Evidence (1). The Evidence tab is selected, showing one entry: "Probar login exitoso de saucedemo.com con credenciales válidas."

En la ejecución podemos registrar las evidencias según realicemos el paso a paso.

The screenshot shows the "Steps" section of a test execution. It lists five steps for a login process:

- Action:** Abrir navegador. **Data:** https://www.saucedemo.com/. **Expected Result:** Página login visible. **Step State:** PASSED.
- Action:** Ingresar username. **Data:** Usuarios Registrados: standard_user. **Expected Result:** Username ingresado. **Step State:** PASSED.
- Action:** Ingresar password. **Data:** Contraseña: secret_sauce. **Expected Result:** Password ingresado. **Step State:** PASSED.
- Action:** Click en login. **Data:** Dato de referencia: id="login-button". **Expected Result:** Usuario autenticado. **Step State:** PASSED.
- Action:** Validar pantalla principal. **Data:** - **Expected Result:**
 - Dashboard visible
 - URL contiene saucedemo.com/inventory.html
 - Se visualiza listado de productos**Step State:** PASSED.

Each step includes sections for Actual Result, Comment, Defects, and Evidence.

Se crea Test Execution y agrega XML de Test automatizado con Python

The screenshot shows the Jira Test Management interface for a test execution named "EXE - Automatizado - Login Exitoso".

Overall Execution Status:

- 1 PASSED
- TOTAL TESTS: 1

Test Details:

| Rank | Key | Summary | Test Type | Dataset | #Defects | Status | Action |
|------|--------|----------------------|-----------|---------|----------|--------|--------|
| 1 | QA2-45 | test_login_saucedemo | Generic | grid | 0 | PASSED | View |

Details Panel:

- Assignee: alejandro franco
- Reporter: alejandro franco
- Developer: Create branch, Create commit
- Labels: None
- Due date: None
- Start date: None
- Priority: Medium
- Test Plans: Open Test Plans
- Test Environments: Open Test Environments

More fields:

- Original estimate: 0m
- Time tracking: No time logged
- Components: Add components
- Sprint: None
- Team: None

7.3 Conocimientos aplicados

- Gestión de pruebas en Jira
- Uso de Xray para trazabilidad
- Importación de resultados automatizados (Documento estándar JUnit)
- Lectura e interpretación de reportes

7.4 Recomendaciones

- Definir convenciones de nombres para tests
- Centralizar evidencias en Test Executions
- Usar Test Plans para ciclos de prueba formales

8. Conclusiones

El Proyecto QA Integral permitió aplicar un enfoque completo de aseguramiento de calidad, abarcando múltiples capas del sistema y combinando pruebas manuales y automatizadas. Se adquirieron conocimientos prácticos en herramientas ampliamente utilizadas en la industria, así como en la gestión y trazabilidad de pruebas.

El proyecto deja una base sólida para evolucionar hacia prácticas más avanzadas como CI/CD, pruebas de performance y automatización a mayor escala.

Filosofía de Calidad

"El código no es evidencia. La evidencia es el resultado observable de la prueba: screenshots, logs y reportes. El script vive en Git, pero la evidencia de ejecución vive en Jira/Xray".