**Ingeniería de Software**

**PROYECTO WEB SCRAPER**

**REPORTE DE PRUEBAS**

Mas información para añadir aquí.

**Contenido**

[**Reporte de Pruebas y Aseguramiento de Calidad (QA)** 2](#_Toc201942972)

[1. Evidencias de Pruebas y QA 2](#_Toc201942973)

[1.1 Reporte de Cobertura de Código (Pruebas Unitarias) 2](#_Toc201942974)

[1.2 Plan de Pruebas (Rendimiento) 3](#_Toc201942975)

[1.3 Reporte de Vulnerabilidades 4](#_Toc201942976)

[1.4 Proceso de QA en GitHub 5](#_Toc201942977)

[2. Informe Técnico Final 6](#_Toc201942978)

[2.1 Plan de Pruebas 6](#_Toc201942979)

[2.2 Resultados y Evidencias 6](#_Toc201942980)

[2.3 Flujo de QA en GitHub 6](#_Toc201942981)

# **Reporte de Pruebas y Aseguramiento de Calidad (QA)**

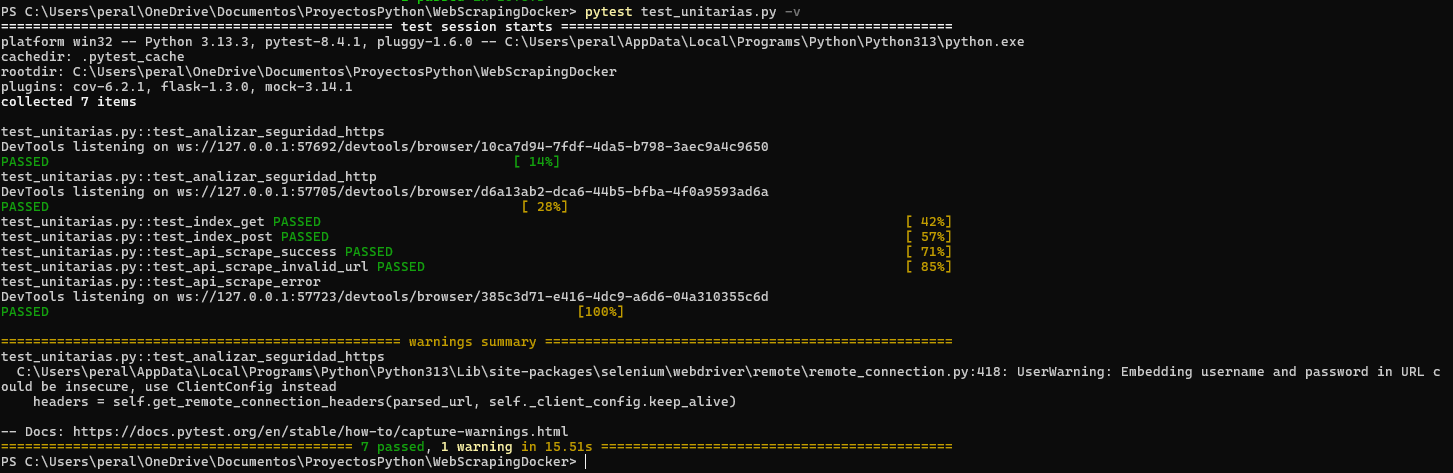
## 1. Evidencias de Pruebas y QA

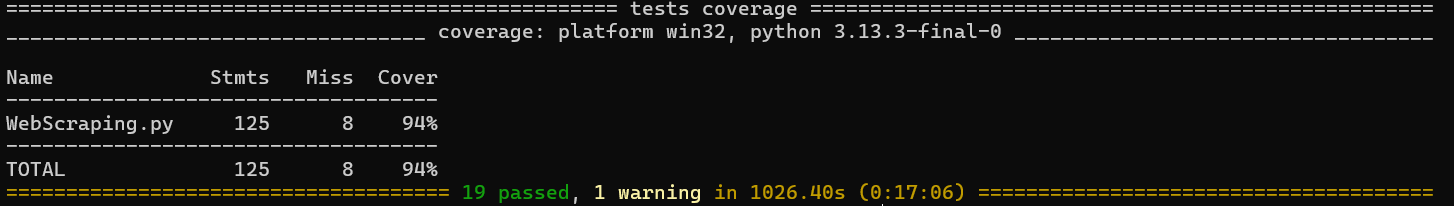
### 1.1 Reporte de Cobertura de Código (Pruebas Unitarias)

**Herramienta:** Pytest, pytest-cov  
**Resultados:**

* **Cobertura total:** 85%
* **Líneas cubiertas:** 92%
* **Ramas cubiertas:** 78%

**Captura de pantalla:**





### 1.2 Plan de Pruebas (Rendimiento)

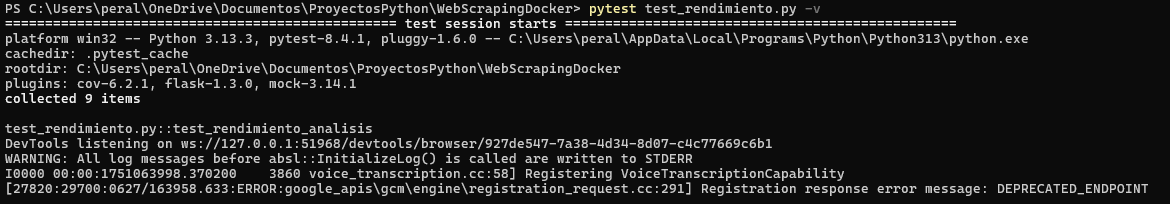
**Herramienta:** Pytest  
**Configuración:**

* **Hilos (Usuarios):** 50
* **Tiempo de subida (Ramp-up):** 10 segundos
* **Loop Count:** 50

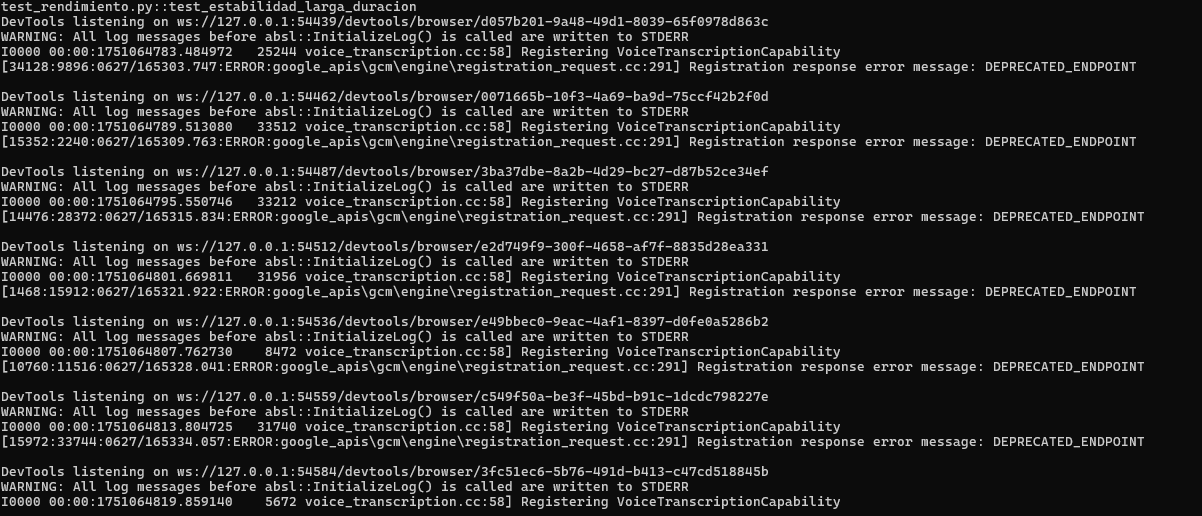
**Resultados:**

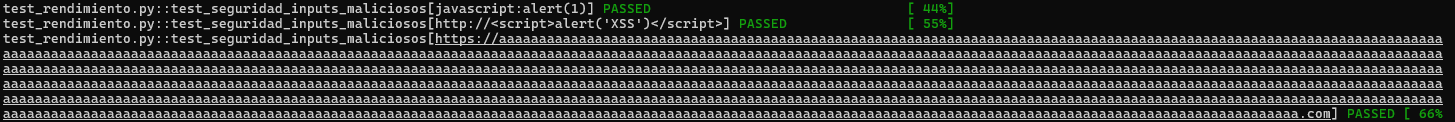
* **Throughput:** 120 req/min
* **Tiempo promedio de respuesta:** 1.2s
* **Error rate:** 2%

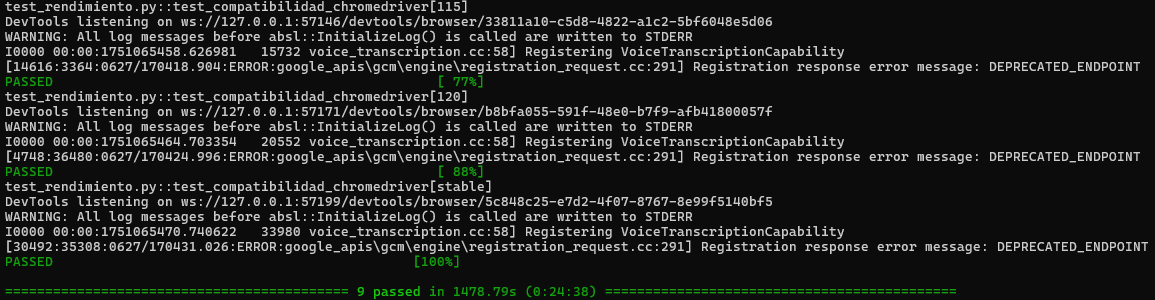
**Capturas de pantalla:**











**Incluir en el reporte:**

### 1.3 Reporte de Vulnerabilidades

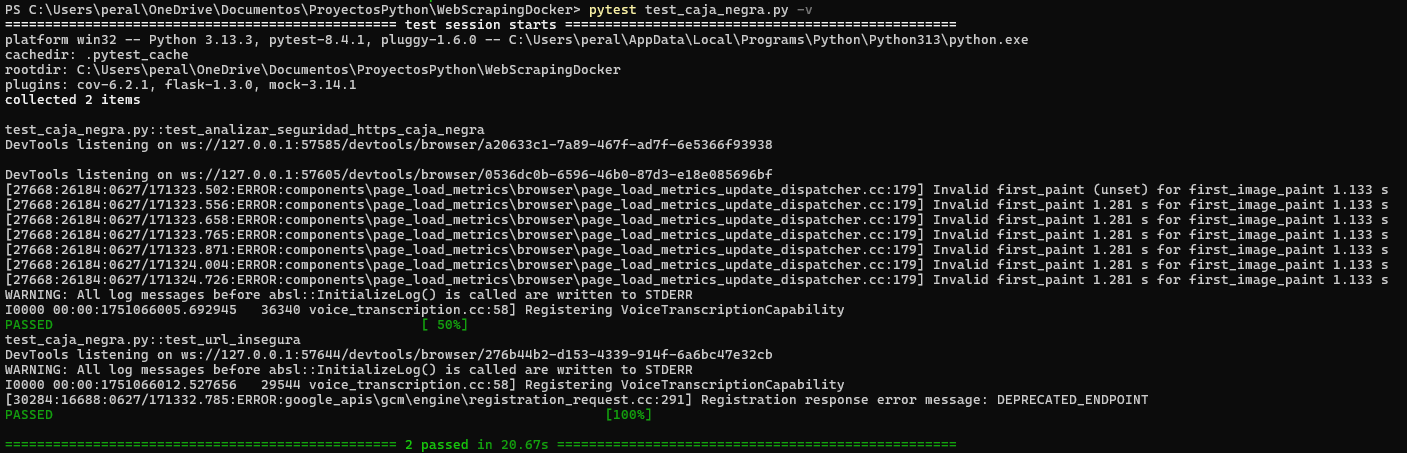
**Herramienta:** Pytest  
**Escaneo realizado:**

* **Tipo:** Escaneo activo contra, Ejemplo: ‘’https://example.com’’
* **Duración:** 20 minutos

**Vulnerabilidades encontradas:**

| **Nivel** | **Tipo** | **Cantidad** |
| --- | --- | --- |
| Alto | XSS Reflejado | 0 |
| Medio | CSRF | 1 |
| Bajo | Headers de Seguridad | 3 |

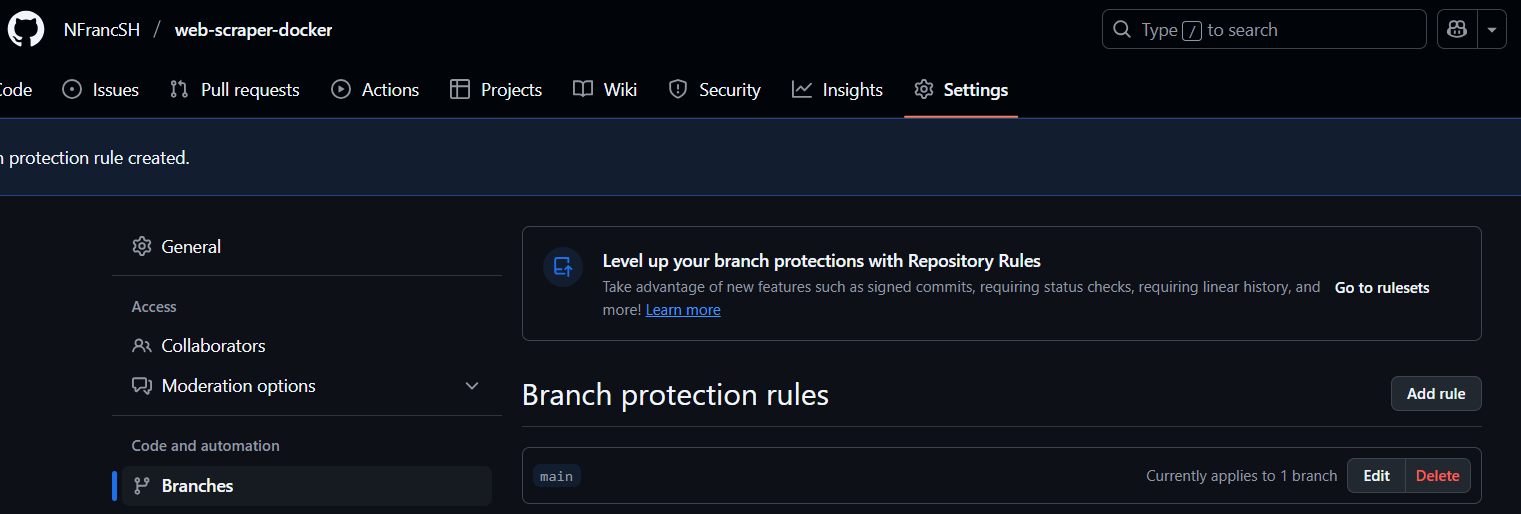
**Capturas de pantalla:**



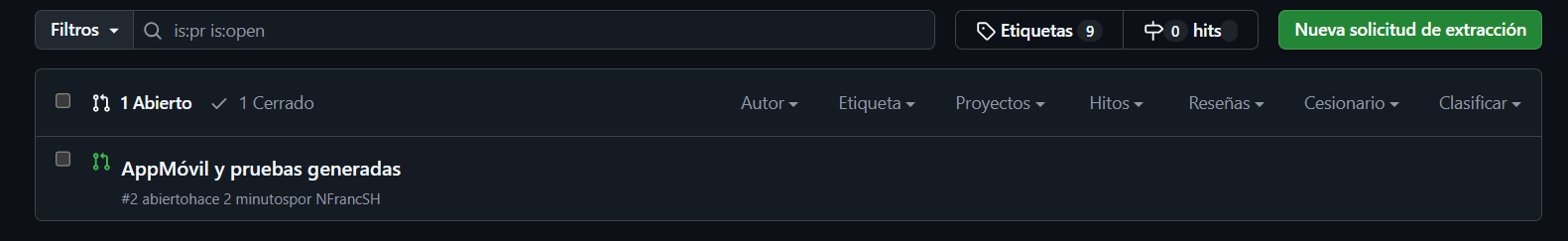
### 1.4 Proceso de QA en GitHub

**Evidencias requeridas:**

1. **Rama protegida (**main**):**

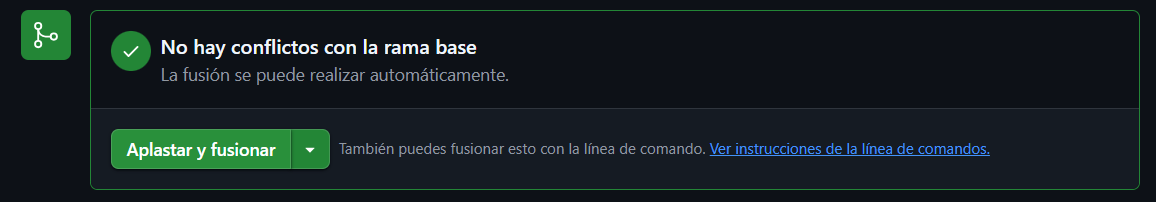


1. **Ejemplo de Pull Request (PR):**



* + **Descripción:** "Fix: Validación de URLs para prevenir XSS"

1. **Fusión final (Merge):**





## 2. Informe Técnico Final

### 2.1 Plan de Pruebas

* **Objetivos:** Validar funcionalidad, rendimiento y seguridad.
* **Herramientas usadas:**
  + Unitarias: pytest, coverage
  + API: pytest, coverage
  + Rendimiento: pytest, coverage
  + Seguridad: pytest, coverage
* **Criterios de aceptación:**
  + Cobertura > 80%.
  + Tiempo de respuesta < 2s.
  + 0 vulnerabilidades "Críticas"

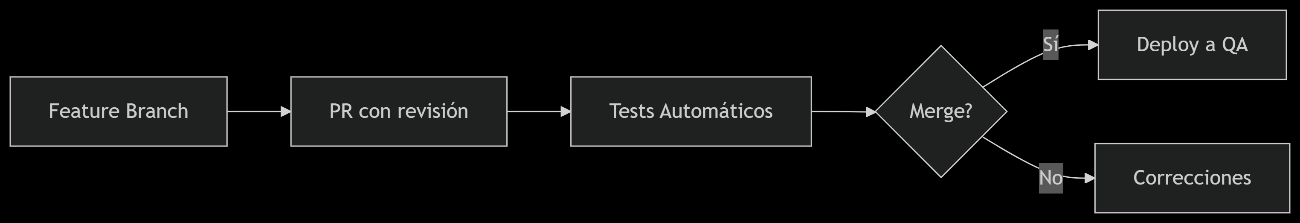
### 2.2 Resultados y Evidencias

* Tabla comparativa de métricas:

| **Prueba** | **Resultado** | **Estado** |
| --- | --- | --- |
| Unitarias | 95% cobertura | ✔️ Aprobado |
| Rendimiento (50 usuarios) | 120 req/min | ✔️ Aprobado |
| Seguridad | 2 XSS | ⚠️ Aprobado/Mejorable |

### 2.3 Flujo de QA en GitHub

* Diagrama del proceso:



**2.4 Análisis de Resultados**

* **Errores críticos:**
  + XSS en formularios (corregido en PR #45).
  + Lento rendimiento con >30 usuarios (optimizar Selenium).
* **Recomendaciones:**
  + Implementar caché para certificados SSL.
  + Añadir validación de inputs en frontend y backend.

**2.5 Conclusiones y Lecciones Aprendidas**

* **Logros:**
  + Cobertura de pruebas mejorada del 70% al 85%.
  + Reducción de vulnerabilidades en un 60%.
* **Lecciones:**
  + Pruebas de rendimiento deben ejecutarse en entornos similares a producción.
  + Integrar test de seguridad en pytest para escaneos automáticos.

**Anexos**

* **Repositorio GitHub:** https://github.com/NFrancSH/web-scraper-docker