

Proyecto: Pokeapi - REST Json

Estrategia de Prueba Automatizadas Reto Rest

Historia de revisiones

|  |  |  |  |
| --- | --- | --- | --- |
| Versión | Autor(es) | Descripción | Fecha |
| 1.0 | David Zaraza | Creación del documento | Octubre 2021 |

Tabla de Contenidos

[1.Introducción](#_knb62690yxb3)

[2. Alcance](#_qlpgkz6qysqk)

[3. Roles y Responsabilidades](#_7b58fkv4on7k)

[5. Ambiente y Herramientas de Pruebas](#_d7hpcbc6w3za)

[5.1 Herramientas de Pruebas](#_3dy6vkm)

[5.2 Arquitectura del framework de automatización](#_4d34og8)

[5.3 Ambiente de Pruebas](#_17dp8vu)

[6. Criterios de Entrada y Salida](#_ye1dj1oecscb)

[6.1 Criterios de Entrada](#_26in1rg)

[6.2 Criterios de Salida](#_lnxbz9)

[7. Planificación de ejecución de las pruebas](#_453gv79mohyv)

[7.1 Planificación de las Pruebas de Regresión](#_1ksv4uv)

[8. Reporte de Pruebas](#_44sinio)

# 1.Introducción

En esta Estrategia para la realización de pruebas automatizadas se describe el alcance de las pruebas, el ambiente de pruebas, los recursos necesarios, las herramientas a utilizar, los riesgos, planes de contingencia y el calendario de ejecución de las pruebas del proyecto reto Pokeapi Rest json .

# 2. Alcance

Se realizarán pruebas automatizadas al servicio REST funcionalidad asignada “pokeapi generation” . El servicio REST a ser automatizadas tendrá dos escenarios.

# 3. Roles y Responsabilidades

|  |  |
| --- | --- |
| Roles | Responsabilidades |
| Automatizador de pruebas | Planificación de estrategia de pruebas  Diseño e implementación de las pruebas.  Ejecución de las pruebas automatizadas.  Reporte de resultados de las pruebas. |

4. Riesgos y Planes de Contingencia

# 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Riesgos | Probabilidad de Ocurrencia  (1-5) | Impacto  (1-5) | Severidad  (Prob\*Impacto) | Plan de Contingencia |
| 1 | Caida del webServices | 1 | 5 | 10 | Contar con el recurso, en su defecto automatizar un servicio de terceros. |
| 2 | Daño en la energía del hogar, servicios de internet o equipos de computo donde se desarrolla el proyecto | 1 | 5 | 10 | Se cuenta con un equipo adicional (personal) diferente al suministrado por Sofka. En caso de que el internet se caiga se cuenta con plan de datos para compartir los mismos.  Ante el servicio de energía mantener el equipo cargado en todo momento. |

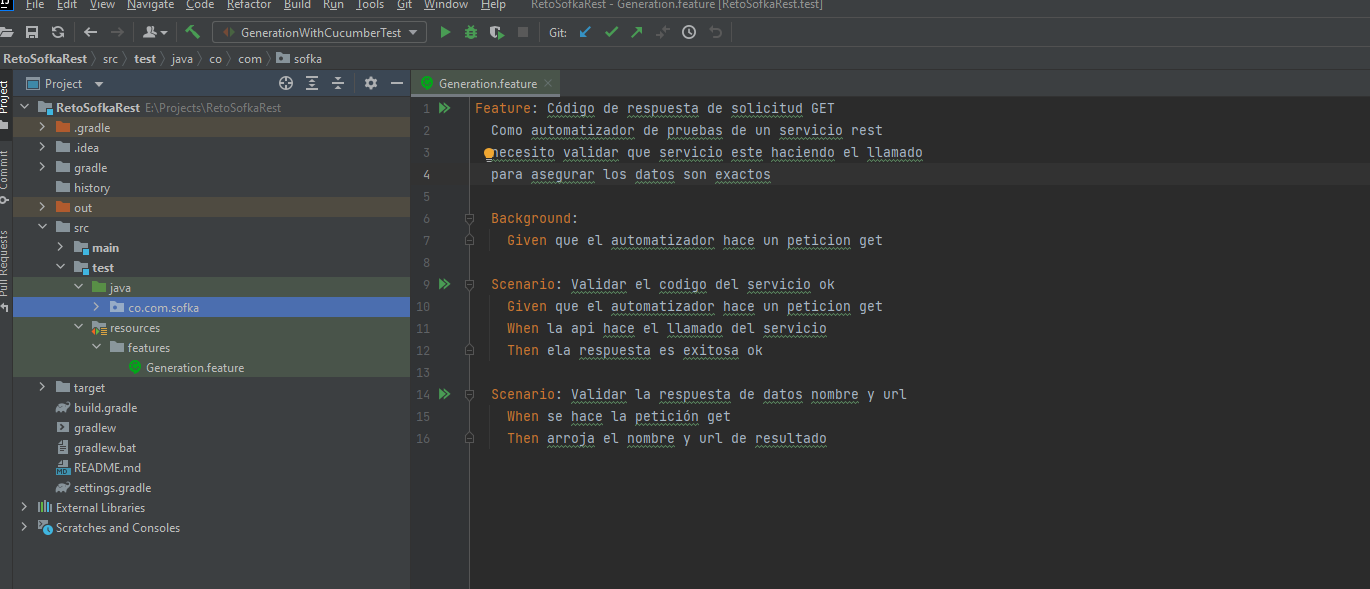
# 5. Ambiente y Herramientas de Pruebas

## 5.1 Herramientas de Pruebas

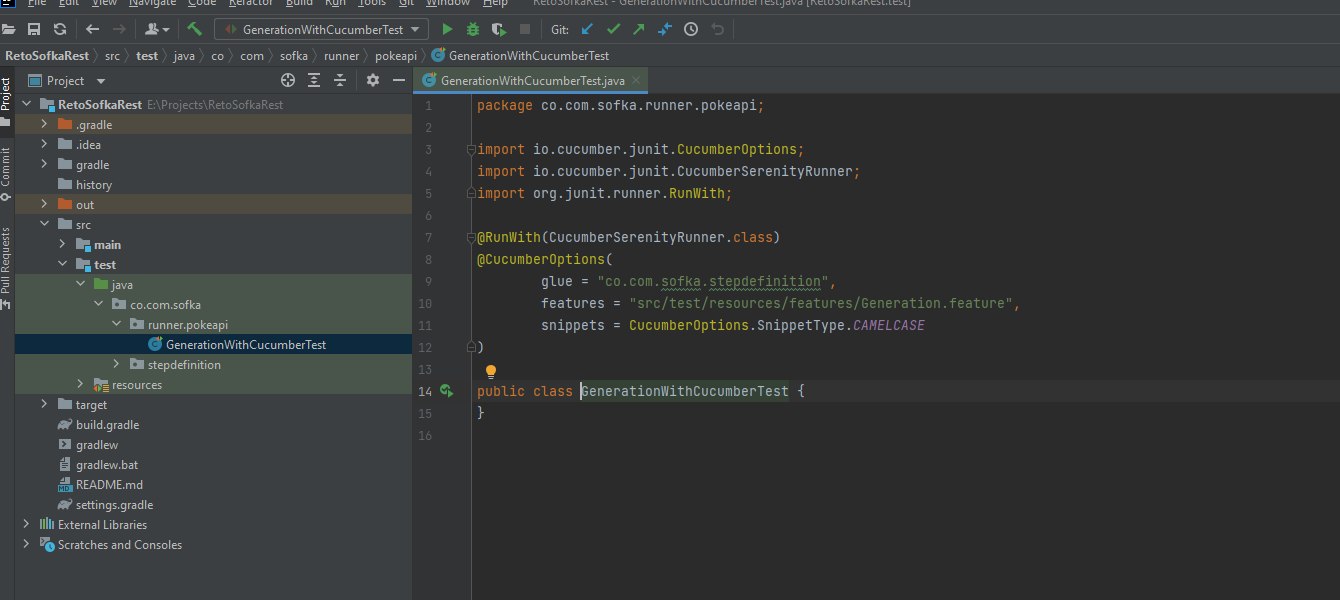
|  |  |
| --- | --- |
| Herramienta | Función |
| Serenity | Framework para automatizar sistemas Web |
| Gradle | Creación de la estructura de proyectos y uso e importación de librerías |
| IntelliJ | IDE para la codificación y compilación del proyecto |
| Java JDK y JSE | V 1.8 como lenguaje para trabajar el framework Serenity y el DSL Cucumber con Gherkin |
| Cucumber y Gherkin | Herramienta para crear los escenarios con orientación al BDD |
| POSTMAN | Verificación del servicio a automatizar |

## 5.2 Arquitectura del framework de automatización

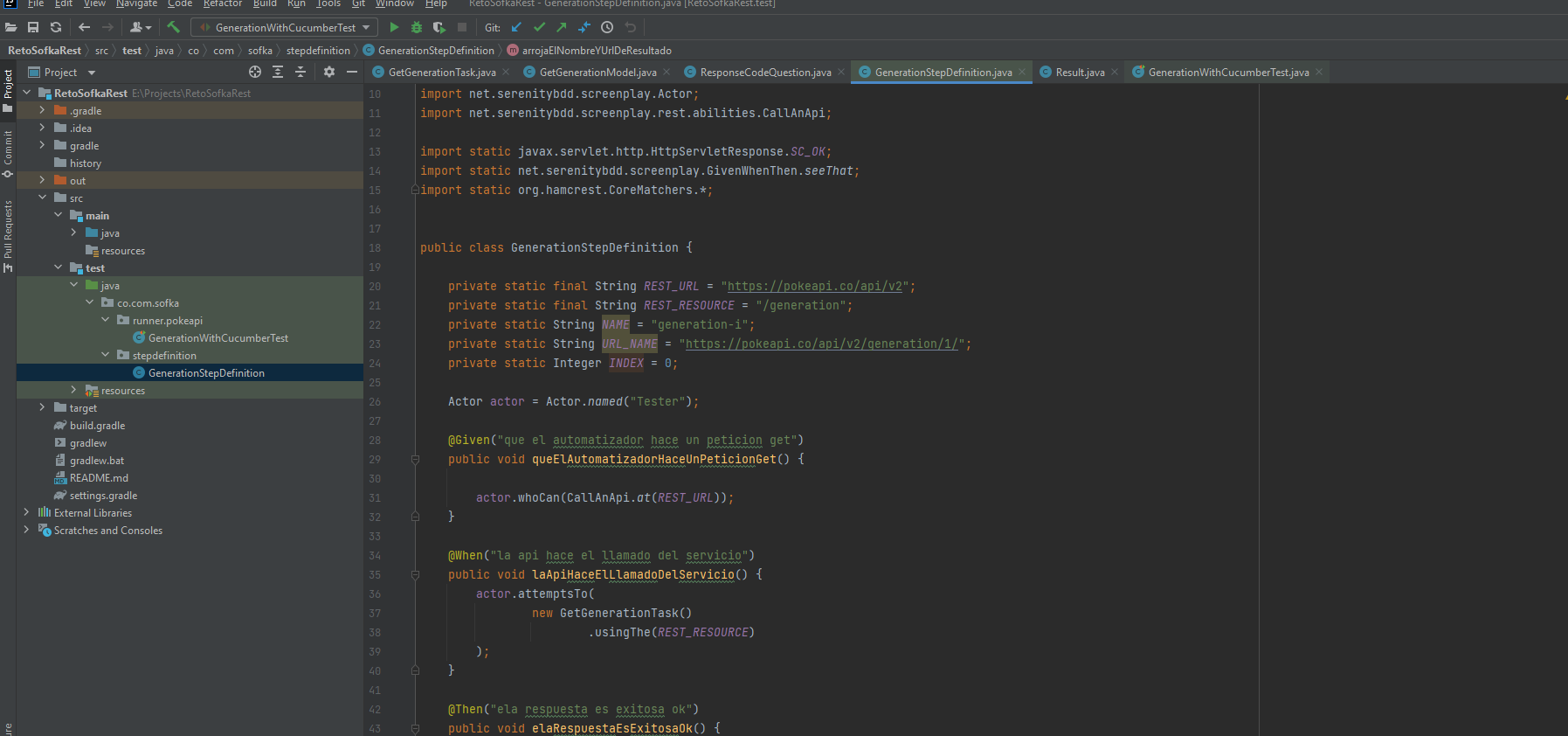
El proyecto estará desarrollado bajo el patrón Screenplay. En el archivo .feature se propondran los escenario a analizar mendiante la estructura Cucumber, con el feature, Scenario, Given, When y Then.



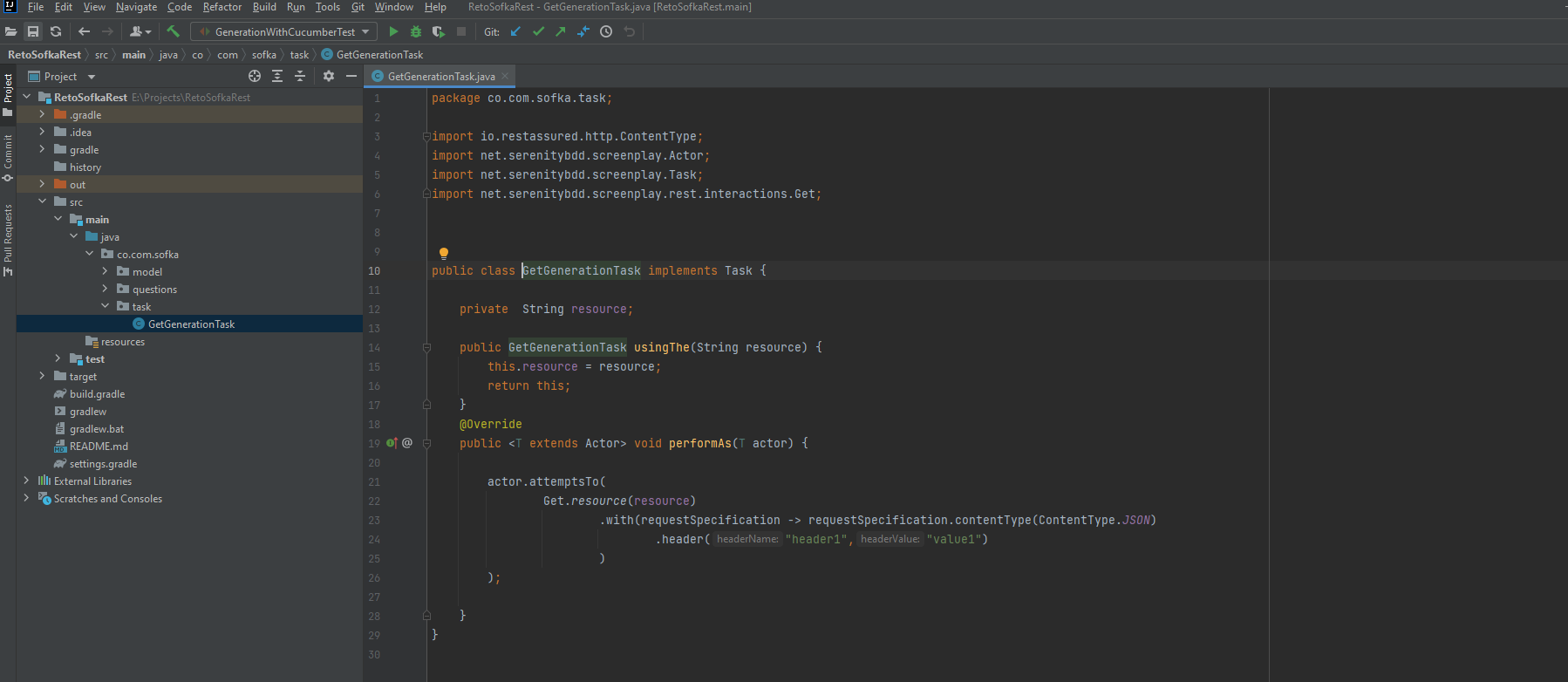
En segundo lugar, el proyecto tendra estructura por capas: en la capa de runner será la ejecución del proyecto.



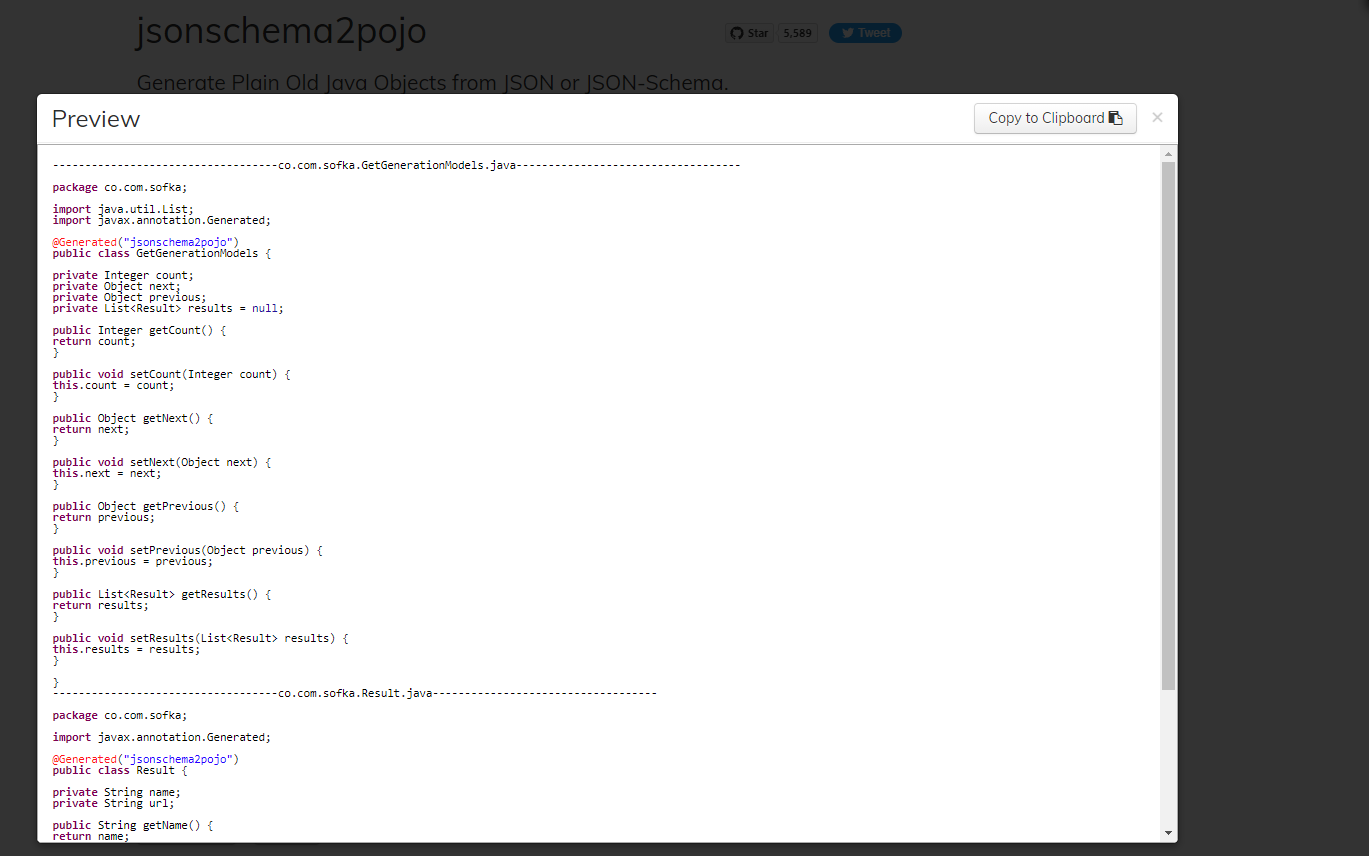
Contará conla capa de los StepDefinitions, el cual tendrá en código el paso a paso de la pruebas automatizada,para los dos escenarios.

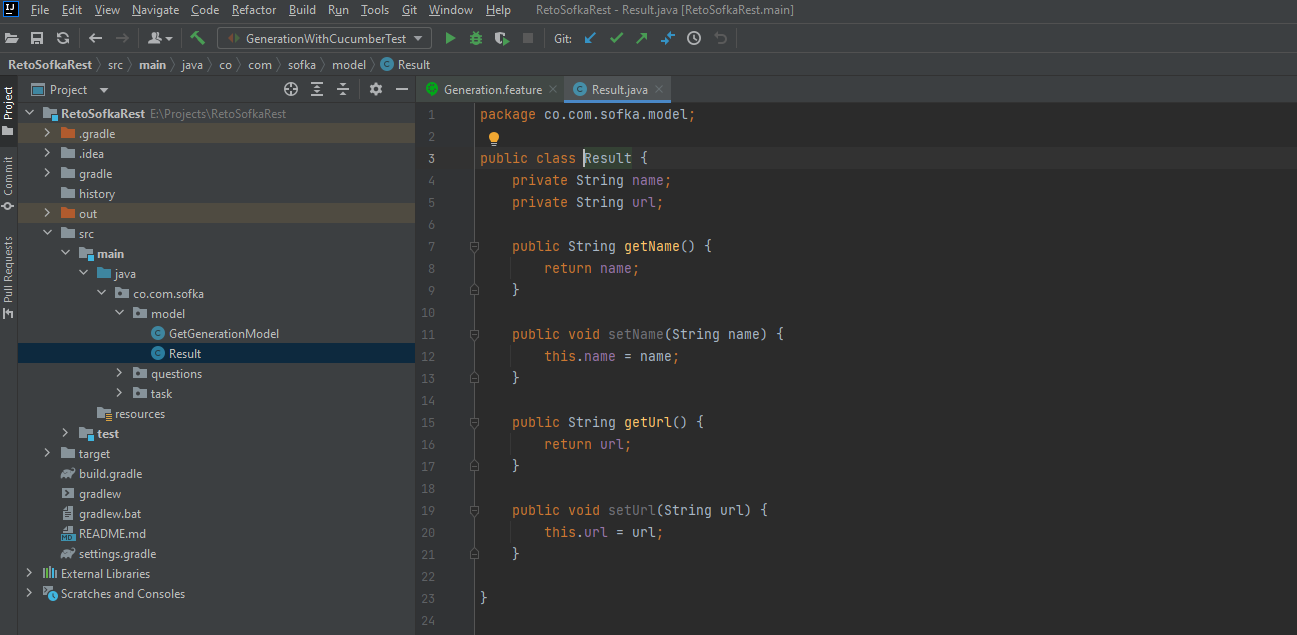


En la carpeta Main, se trabajará bajo patron Screenplay como se ha mencionado, y se crearan paquetes y clases de soporte para el stepdefinition, tanto en la questions como con Task.



Por último una capa para el modelo de datos, con el apoyo en del mapeo de: jsonschema2pojo





## 5.3 Ambiente de Pruebas

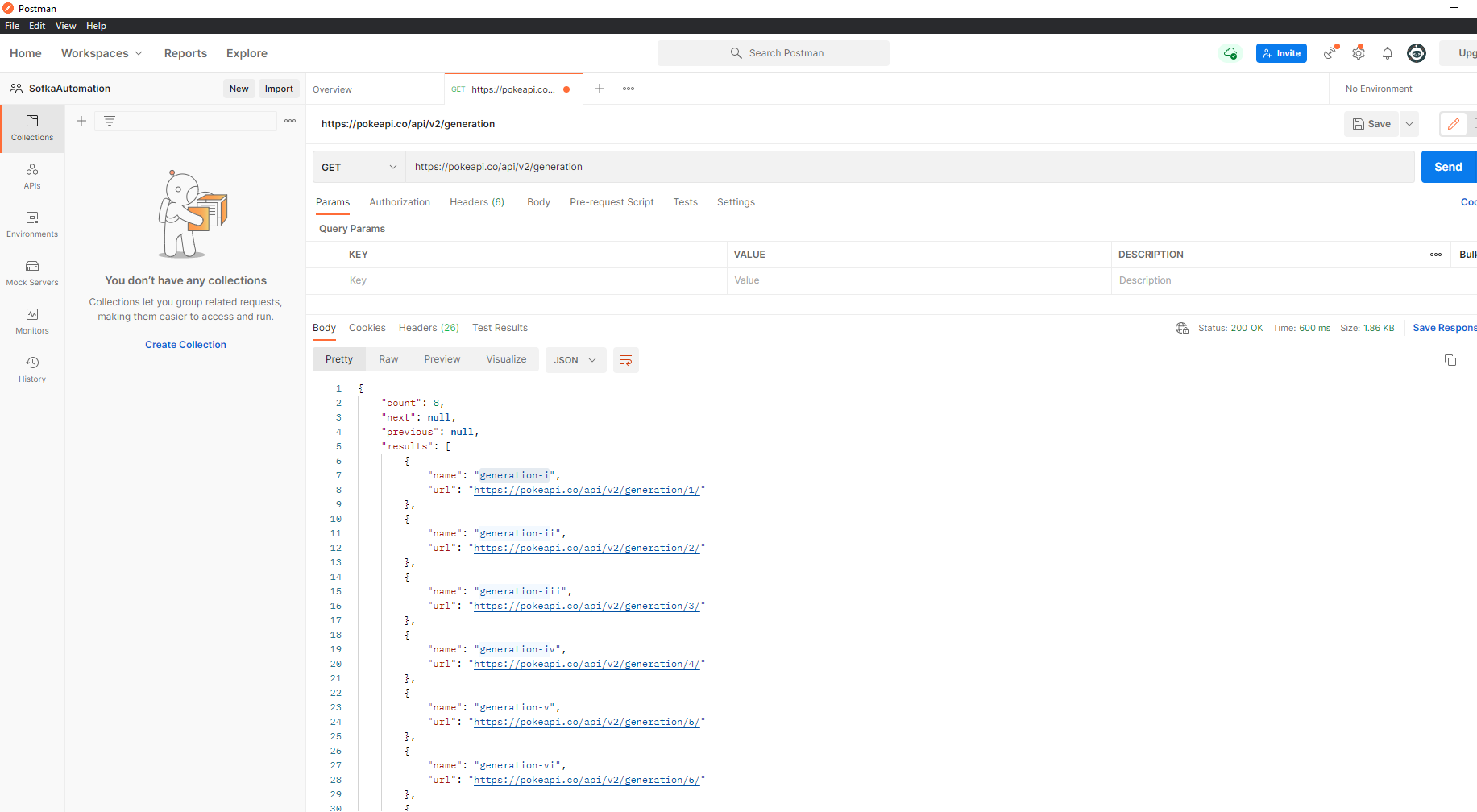
|  |  |
| --- | --- |
| Navegadores | Chrome |
| Sistemas Operativos | Windows 10 de 64 bits |

# 6. Criterios de Entrada y Salida

## 6.1 Criterios de Entrada

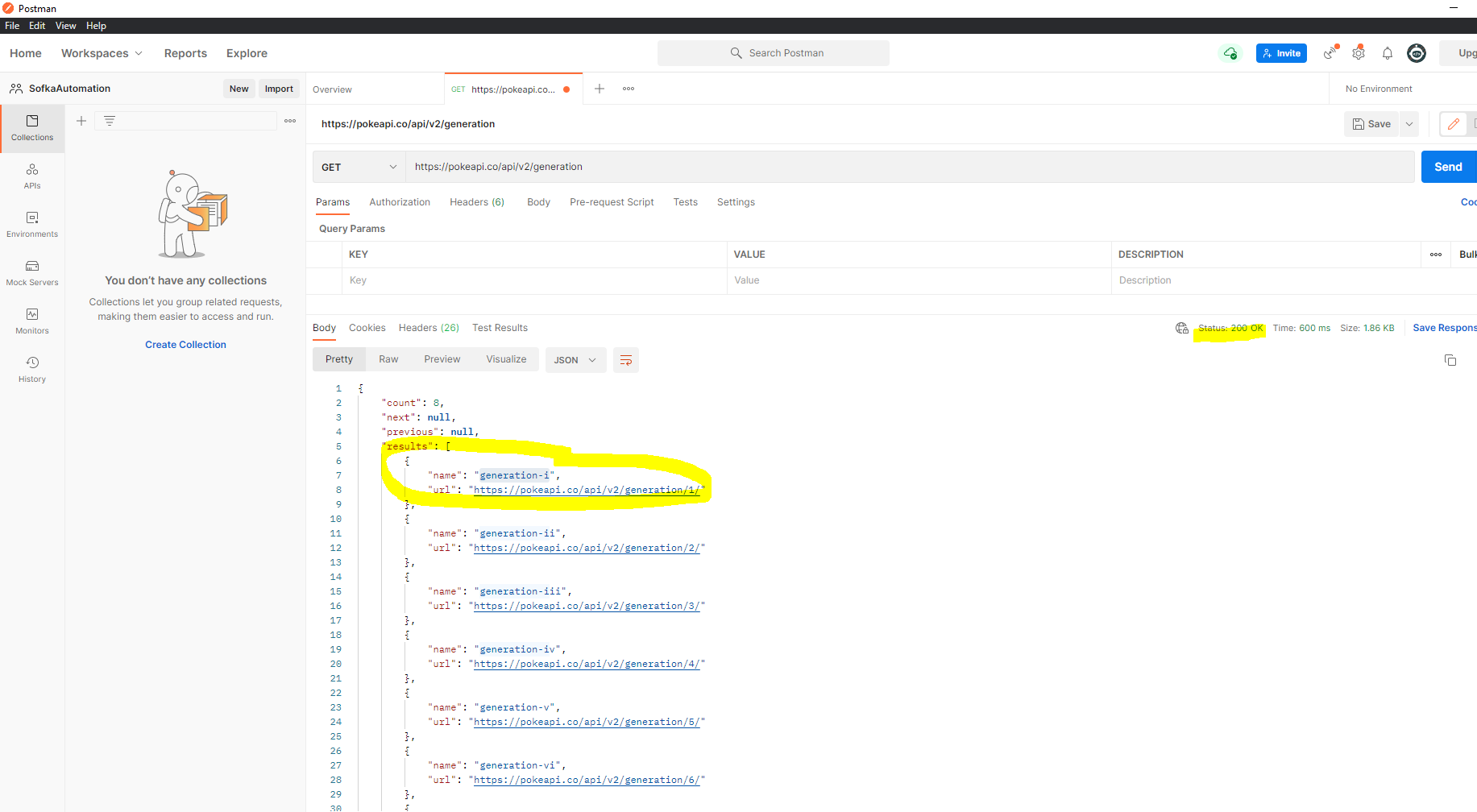
La URL base para trabajar: https://pokeapi.co/api/v2/generation

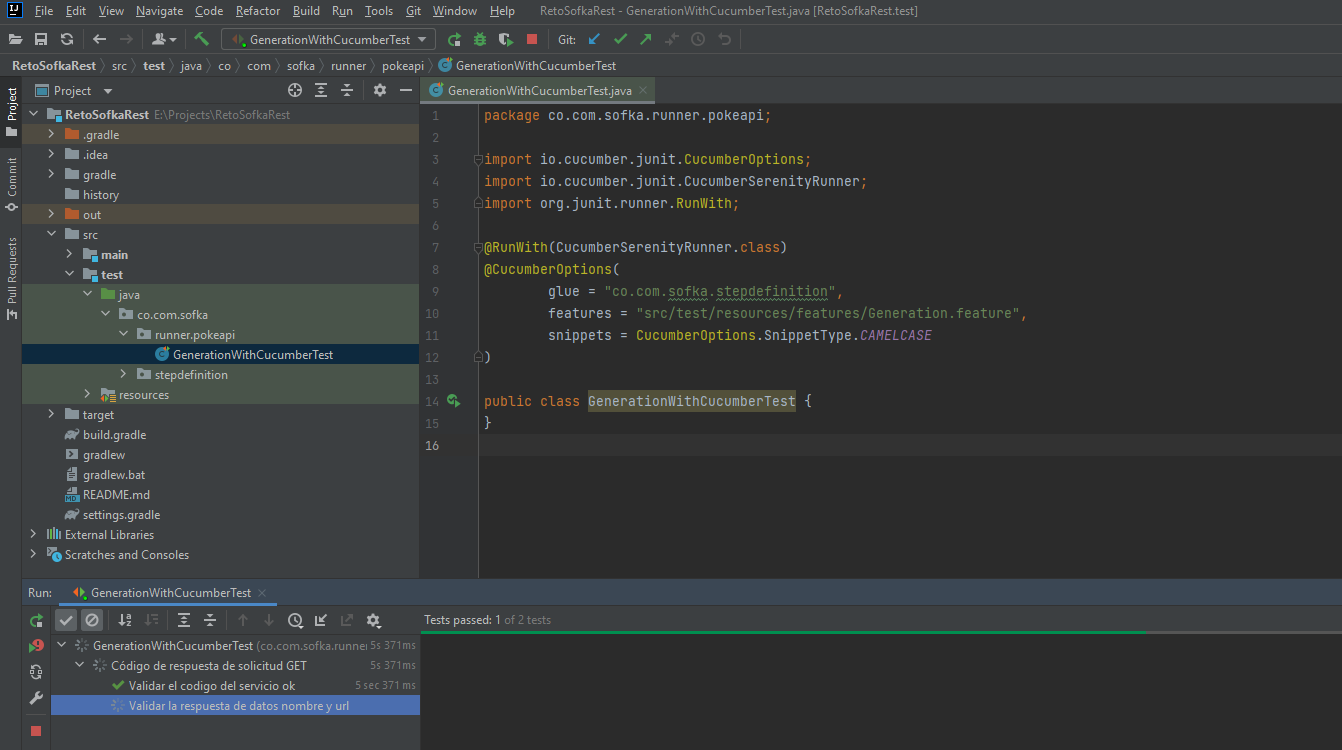
Servicio: generation

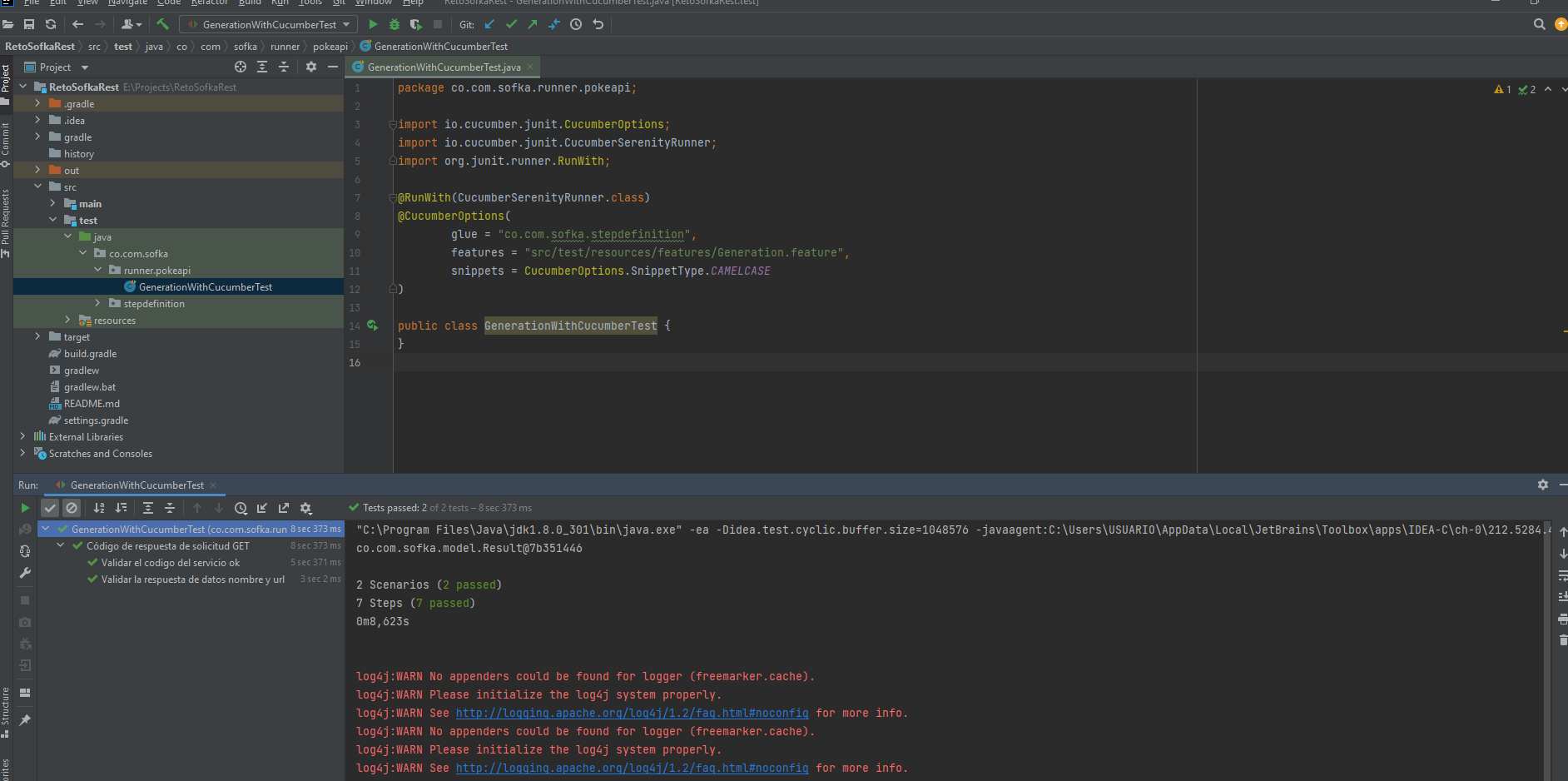


## 6.2 Criterios de Salida

Al ejecutar la prueba el resultado debe ser exitoso y fallido.







# 7. Planificación de ejecución de las pruebas

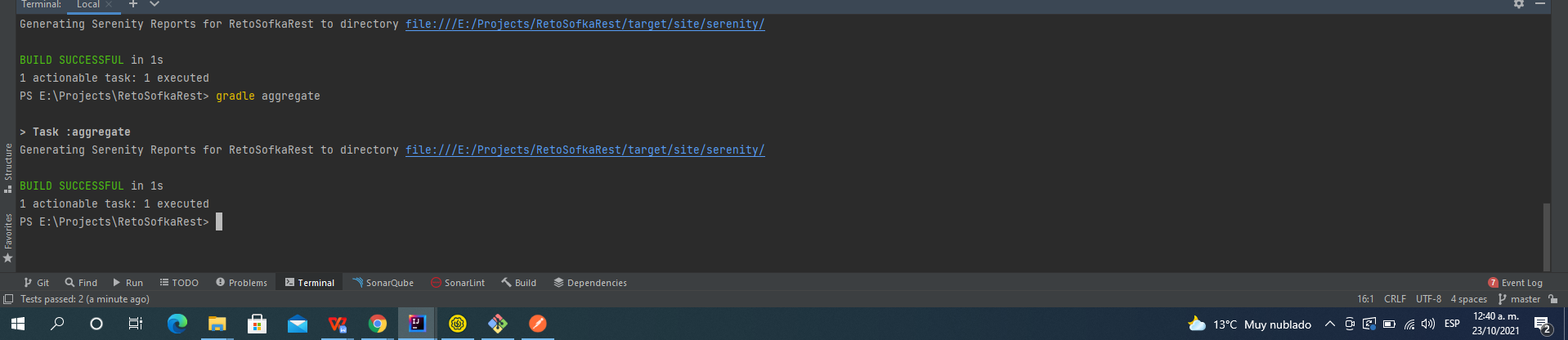
Lista de funcionalidades a ser automatizadas por el reto

|  |  |  |
| --- | --- | --- |
| Reto | Funcionalidad | Comentarios |
| 1 | Language Name | Contiene 2 escenarios   * Buscar el nombre de idioma con el código iso. * Validar mensaje de código ISO no encontrado. |

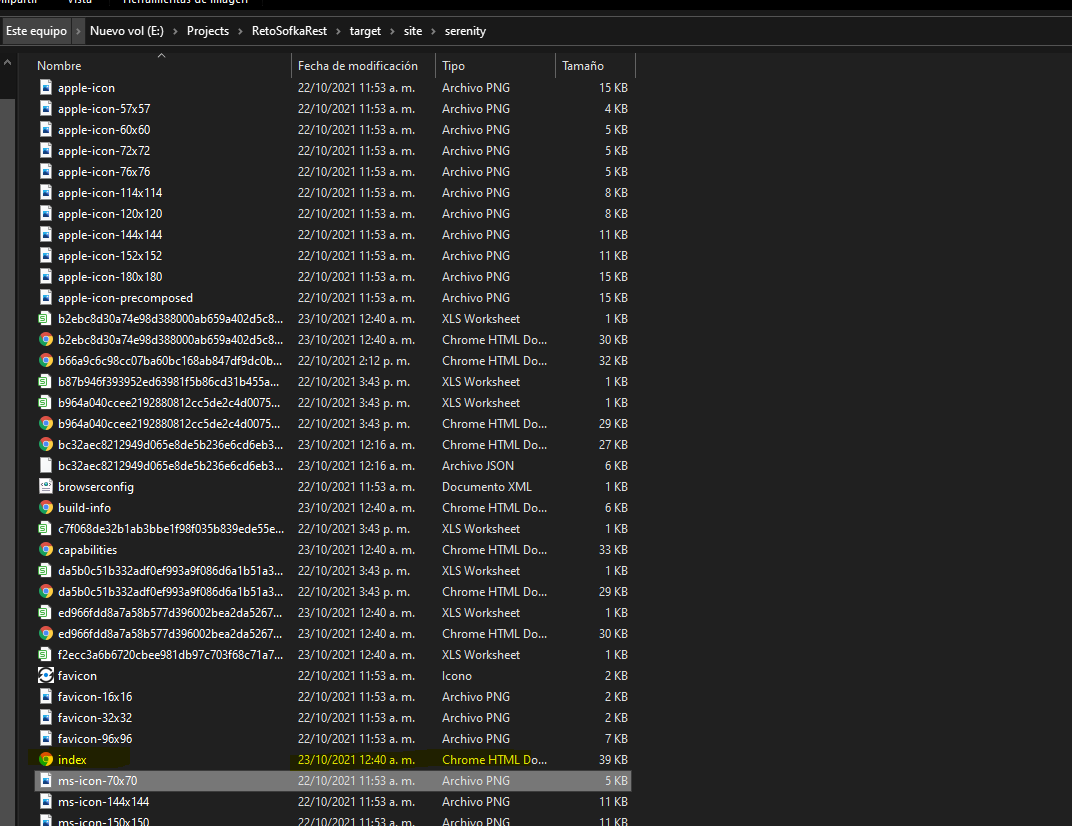
# 8. Reporte de Pruebas

El Reporte de pruebas se obtendra mediante el plugin de Serenity Aggregate, ejecutando en consola el comando:

> gradle aggregate



Y posteriormente al finalizar la tarea ejecutada, se busca el Index.html en la carpeta Target.



Se abre el reporte en navegador web

