Softtek®

MIND HUB.

EQUIPO 3

CHALLENGER MINDHUB

CONTENIDO

NUESTRO EQUIPO

GRISELDA QUIPILDOR

LUCRECIA YEDRO

MARINA LENCINAS PERESSI

VERÓNICA COCHRANE

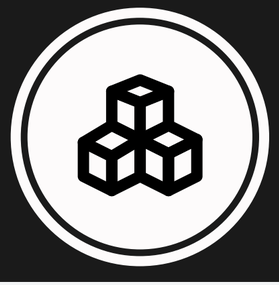
ACTIVIDAD

Durante esta actividad, trabajamos con las herramientas JIRA y Zaphyr para realizar pruebas automatizadas de servicios API. Utilizamos Postman para diseñar y ejecutar colecciones de pruebas, y luego las exportamos para ser ejecutadas desde la línea de comandos mediante Newman, lo cual permitió mayor automatización.
Generamos un reporte de resultados con Newman en formato HTML para visualizar los detalles de las pruebas ejecutadas. A continuación, integramos este flujo en Jenkins, configurando un pipeline que ejecuta Newman automáticamente y genera el reporte como parte del proceso de integración continua.
Finalmente, subimos todo el trabajo (colecciones, scripts, configuraciones de Jenkins y reportes) al repositorio en GitHub, permitiendo el control de versiones y la colaboración del equipo.

API

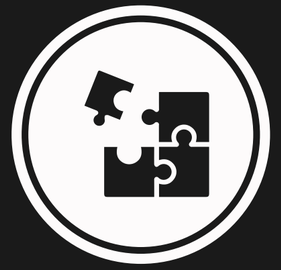
La API de Crossref **(https://api.crossref.org/)**, es una API abierta y gratuita el cual permite acceder a una gran base de datos de publicaciones académicas y científicas. A través de ella se pueden buscar artículos, libros, revistas y otros contenidos usando criterios como título, autor, DOI o año de publicación. Es muy útil para obtener metadatos de publicaciones, como citas, autores, fechas y editores, de forma automatizada y estructurada.

APRENDIMOS



Automatizar pruebas de servicios API

Diseñamos y ejecutamos pruebas automatizadas usando Postman, JIRA, Zaphyr y Newman para asegurar la calidad de los servicios API.



Integrar pruebas automatizadas en el pipeline de Jenkins

Configuramos un pipeline en Jenkins que ejecuta automáticamente las pruebas con Newman y genera reportes en formato HTML. Seguiremos optimizando esta integración para acelerar el ciclo de desarrollo y facilitar la entrega continua.

Gestionar el trabajo con control de versiones colaborativo

Subimos colecciones, scripts, configuraciones y reportes a un repositorio en GitHub, facilitando el trabajo en equipo y el seguimiento de cambios

CRONOLOGÍA

Resumen cronologico del trabajo realizado en el equipo

1

ANALISIS DE API

Analizar la API, descargar la coleccion, verificar endpoints

2

JIRA

Creación de Épicas y las Historias de Usuario

3

ZEPHYR + POSTMAN

Creación de Casos de Prueba con Gherkin.
Creación de request en la colección con script

4

EJECUCION

Se ejecutaron los CP en Zephyr.
Se ejecutó la colección con los Script en Newman y Jenkins

5

METRICAS

Se generaron los reportes en Jira.
Se recolectaron los reportes de Newman y Jenkins

6

GIT

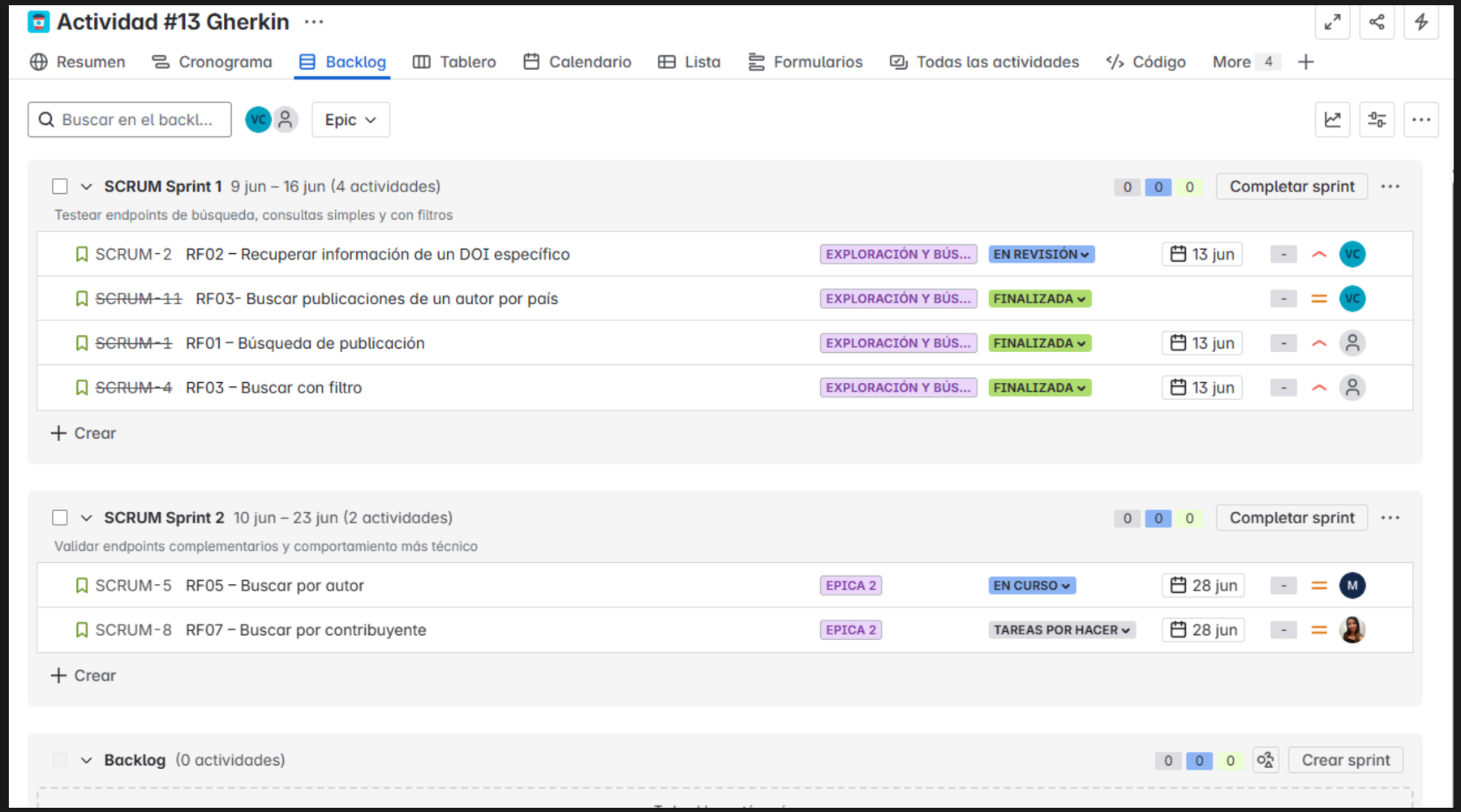
Se sube la presentación a Git en la rama del equipo

JIRA

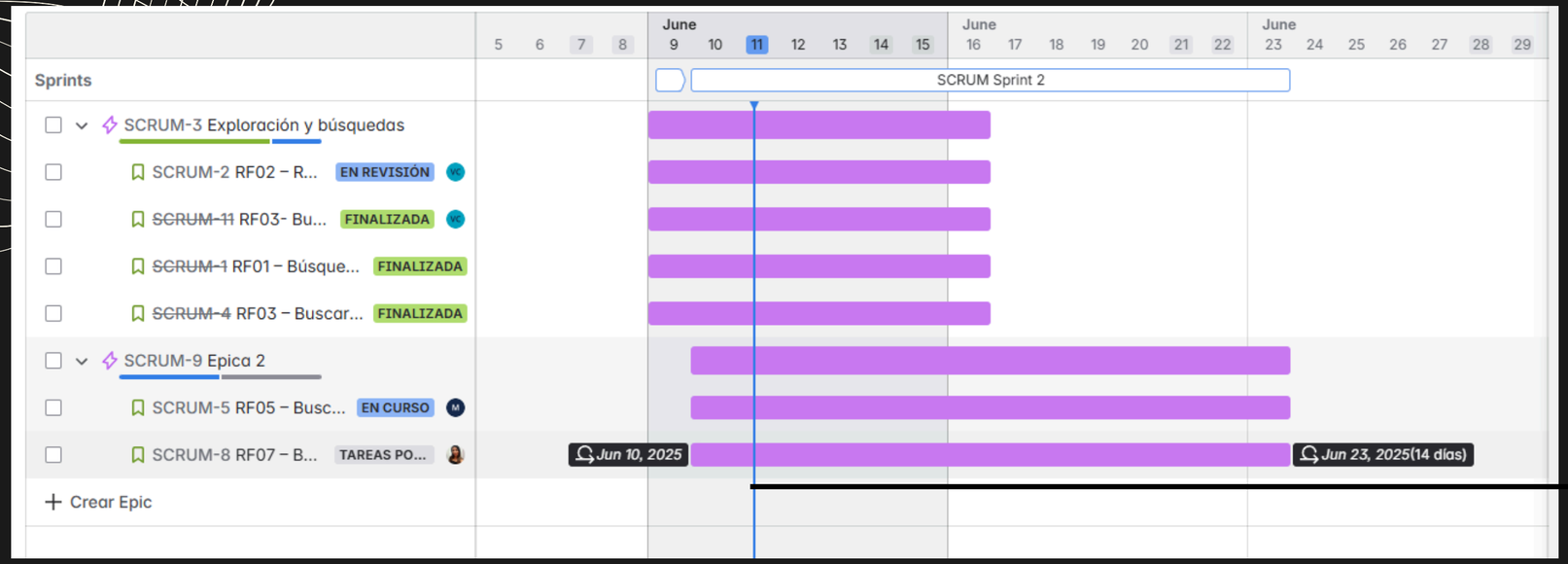
- Seleccionamos los requerimientos funcionales y generamos las Historias de Usuario con sus criterios de aceptación.
- Creamos las épicas y las asignamos a los Sprint correspondiente
- Asignamos las fechas de inicio y fin de cada uno

Link: *Resumen de JIRA*

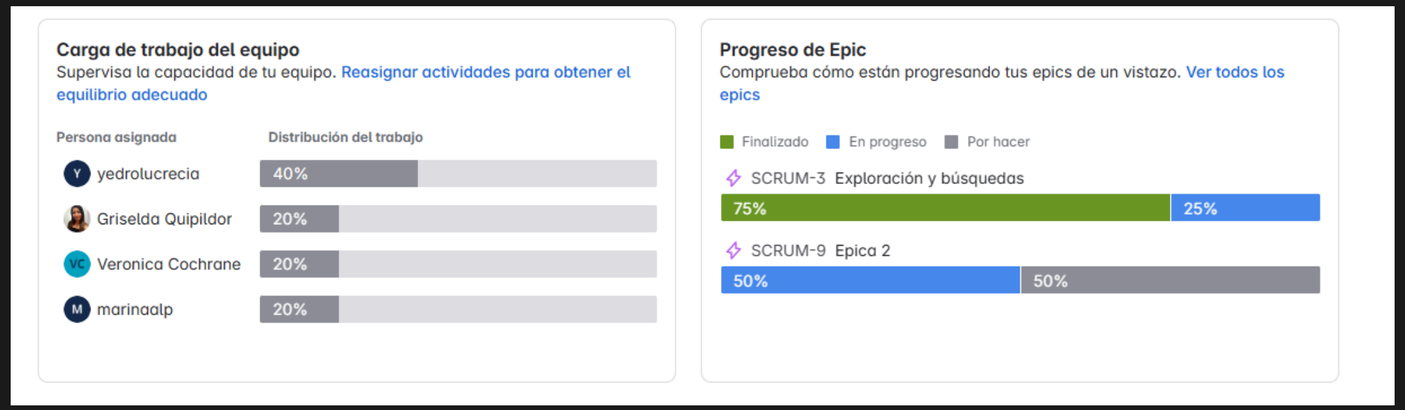
JIRA


Actividad #13 Gherkin
Resumen
Cronograma
Backlog
Tablero
Calendario
Lista
Formularios
Todas las actividades
Código
More 4
Buscar en el backl...
Epic
SCRUM Sprint 1 9 jun – 16 jun (4 actividades)
0 0 0
Completar sprint
Testear endpoints de búsqueda, consultas simples y con filtros
SCRUM-2 RF02 – Recuperar información de un DOI específico
EXPLORACIÓN Y BÚS...
EN REVISIÓN
13 jun
SCRUM-11 RF03- Buscar publicaciones de un autor por país
FINALIZADA
SCRUM-1 RF01 – Búsqueda de publicación
13 jun
SCRUM-4 RF03 – Buscar con filtro
13 jun
Crear
SCRUM Sprint 2 10 jun – 23 jun (2 actividades)
0 0 0
Completar sprint
Validar endpoints complementarios y comportamiento más técnico
SCRUM-5 RF05 – Buscar por autor
EPICA 2
EN CURSO
28 jun
SCRUM-8 RF07 – Buscar por contribuyente
EPICA 2
TAREAS POR HACER
28 jun
Crear
Backlog (0 actividades)
0 0 0
Crear sprint

JIRA



Cronograma y duración de tareas, epicas dentro del sprint

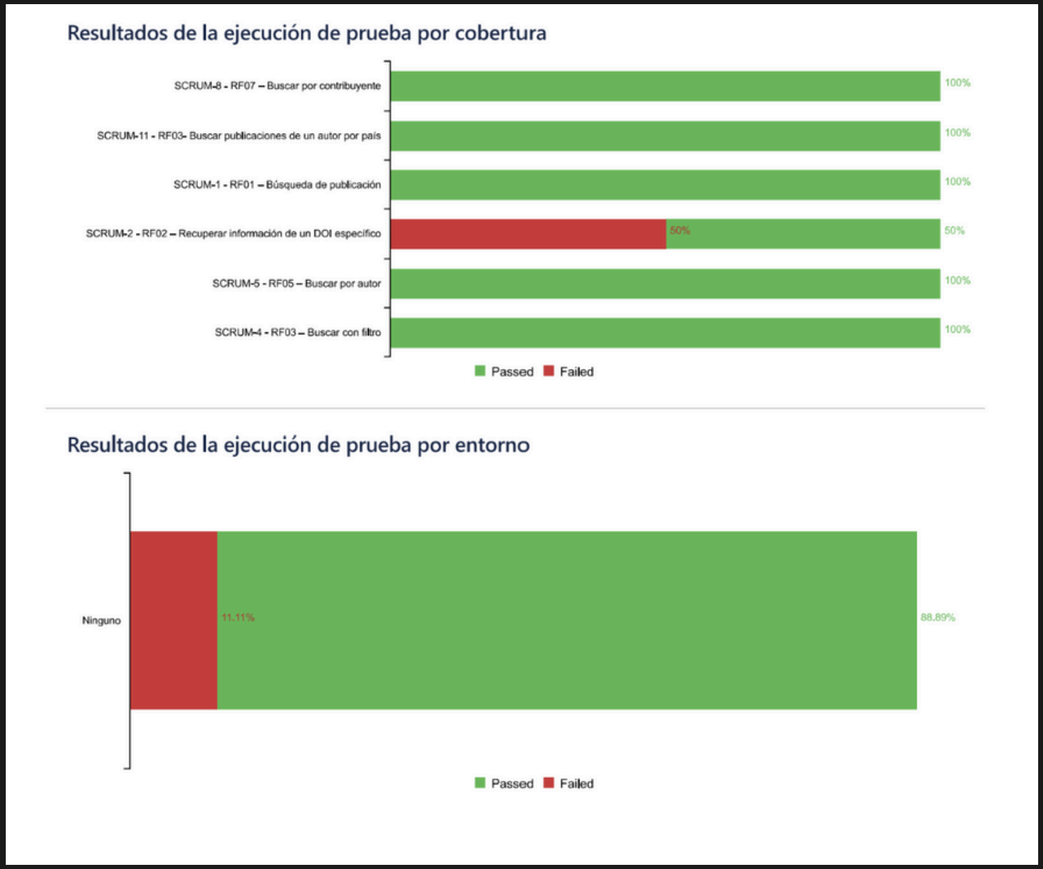


ZEPHYR

- Generamos los casos de prueba correspondientes y escribiéndolos en formato Gherkins para su fácil interpretación.
- Los ejecutamos según los resultados de Postman.
- Generamos los informes correspondientes de las pruebas

Link: Reportes ZEPHY*R*

ZEPHYR


Resultados de la ejecución de prueba por cobertura
SCRUM-8 - RF07 – Buscar por contribuyente
100%
SCRUM-11 - RF03- Buscar publicaciones de un autor por país
100%
SCRUM-1 - RF01 – Búsqueda de publicación
100%
SCRUM-2 - RF02 – Recuperar información de un DOI específico
50%
50%
SCRUM-5 - RF05 – Buscar por autor
100%
SCRUM-4 - RF03 – Buscar con filtro
100%
Passed
Failed
Resultados de la ejecución de prueba por entorno
Ninguno
11.11%
88.89%
Passed
Failed

NEWMAN

En Newman se corrió la colección generada en POSTMAN y luego se generó un reporte en HTML para una mejor visualización de los resultados . En los siguiente links se podrán visualizar dichos reportes

Link: Reporte Consola NEWMAN

Link: Reporte NEWMAN HTML

NEWMAN REPORTE

	executed	failed
iterations	1	0
requests	8	0
test-scripts	7	0
prerequest-scripts	0	0
assertions	14	5
total run duration: 3.7s		
total data received: 36.6kB (approx)		
average response time: 376ms [min: 195ms, max: 988ms, s.d.: 259ms]		

JENKINS

Se logró automatizar las pruebas de Newman a través de Jenkins, donde agiliza fácilmente las pruebas sin pasar por Postman. Recopila todas las pruebas mediante Newman y nos genera un reporte

Link : *Reporte JENKINS*

JENKINS

	executed	failed
iterations	1	0
requests	8	0
test-scripts	7	0
prerequest-scripts	0	0
assertions	14	5
total run duration: 35.5s		
total data received: 36.47kB (approx)		
average response time: 4.3s [min: 186ms, max: 9.9s, s.d.: 3.9s]		

```
io Marina Lencinas

pacio de trabajo C:\Users\marin\.jenkins\workspace\Challenge_Final
cmd /c call C:\Users\marin\AppData\Local\Temp\jenkins11250526986811492411.bat

ins\workspace\Challenge_Final>newman run C:\Users\marin\Downloads\Equipo3_Challenger.postman_collection.json
```

```
os(editores)
ossref.org/members [200 OK, 21.31kB, 2.3s]
200
```

```
└ Listar por autor
  GET https://api.crossref.org/works?query.author={{author}} [200 OK, 8.57kB, 9.9s]
  ✓  Status code is 200
  1. Body incluye 'richard'

└ RF05 -Buscar por contribuyente
  GET https://api.crossref.org/funders?query={{contribuyente}} [200 OK, 679B, 468ms]
  ✓  Status code is 200
  2. Body incluye 'Office of Polar Programs'

□ Sprint 1
└ RF01 - Búsqueda Publicación
  GET https://api.crossref.org/works?query={{searchTerm}} [200 OK, 1.81kB, 4.5s]
  ✓  Status code is 200
  ✓  Body is valid JSON
  3. El título contiene el término de búsqueda
```

POSTMAN

Se utilizó Postman como una herramienta para realizar solicitudes las cuales nos permitiera probar si la Api de Crossref funcionaba correctamente. Luego las ejecutamos de manera automática para verificar sus respuestas y generamos el informe con los resultados.
Se agregaron Asserts y Snnipets para validación de los requerimientos.
Se crearon variables globales para ser utilizadas en los requerimientos.

Link : *Colección*

CONCLUSIONES

Lo aprendido en el diplomado nos permitió organizarnos y trabajar de forma efectiva durante todo el proyecto

CONCLUSIÓN 1

Las herramientas que utilizamos fueron de nuestro agrado en funcionalidades y experiencia de usuario. Nos permitió avanzar en conocimiento

CONCLUSIÓN 2

La API nos pareció poco clara en cuanto a documentación y formato pero Swagger nos permitió comprender y visualizar mejor los requerimientos que se podían realizar

CONCLUSIÓN 3

MUCHAS GRACIAS