«Talento Tech»

Testing QA

Clase 15

Buenos Aires Ciudad

Clase N°15 | Diseño y Ejecución de Casos de Prueba en Jira.

Temario:

- Estados de tickets (To Do, In Progress, Done)
- Mover tickets entre columnas
- Asignación y reasignación
- Creación básica de test cases en Jira
- Vinculación de test cases con user stories
- Registro de resultados de pruebas
- Gestión del ciclo básico de un bug
- Diagrama de Flujo Acumulado y Sprint Report

Objetivos de la clase

En esta clase vamos a aprender a gestionar el ciclo completo de pruebas en Jira. Veremos cómo trabajar con los estados de los tickets, moverlos entre columnas, asignarlos y actualizarlos durante un sprint. Aprenderemos también a crear casos de prueba, vincularlos con historias de usuario, registrar resultados y gestionar defectos de forma estructurada. Todo esto aplicado a un entorno de trabajo realista, simulando el ciclo completo de QA dentro de un sprint Scrum.

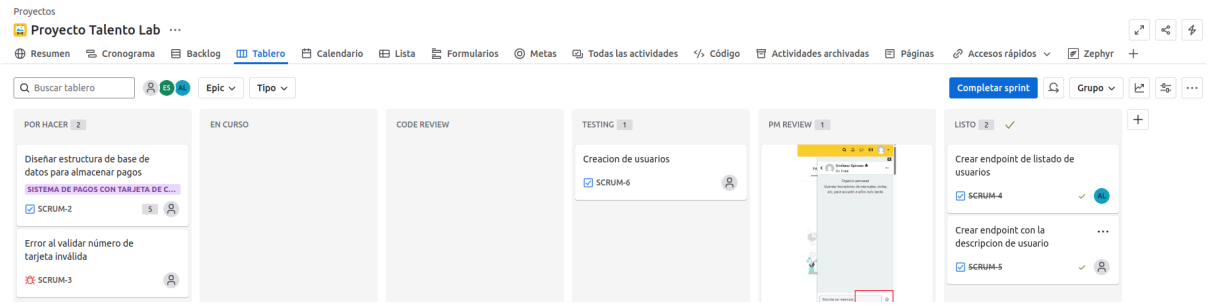
Flujo de trabajo: columnas y su propósito

En Jira, el tablero Scrum se divide en columnas que reflejan los distintos **estados del flujo de trabajo**. A continuación explicamos cada columna común del flujo de QA:

- **TO DO:** La tarea está planificada pero aún no se comenzó. Suele contener todas las historias o bugs que entran al sprint.
- **IN PROGRESS:** Alguien comenzó a trabajar activamente en la tarea. Esto puede incluir desarrollo, documentación, análisis o pruebas.
- **CODE REVIEW:** El desarrollador ya completó su parte y otro miembro del equipo revisará el código para asegurar calidad, estilo y funcionamiento.
- **TESTING:** La funcionalidad ya está implementada y se encuentra en manos del equipo de QA para ser validada mediante casos de prueba. ⚠️ Como QA, esta columna marca el inicio de nuestra responsabilidad. Una vez que una tarea es movida a TESTING, debemos comenzar a ejecutar los test cases previamente diseñados y registrar los resultados.
- **PM REVIEW:** El Product Manager revisa la tarea para confirmar que cumple con la funcionalidad esperada, criterios de aceptación y Definition of Done.
- **LISTO (DONE):** El ticket ha pasado todas las etapas, incluyendo testing y validación, y se considera terminado.

✅ Este flujo ayuda a mantener la trazabilidad de cada tarea y a identificar en qué etapa del ciclo hay cuellos de botella.

En la siguiente imagen puedes ver las columnas con los distintos **estados del flujo de trabajo (Los mismos son configurables)**



Creación de Test Cases utilizando Zephyr en Jira

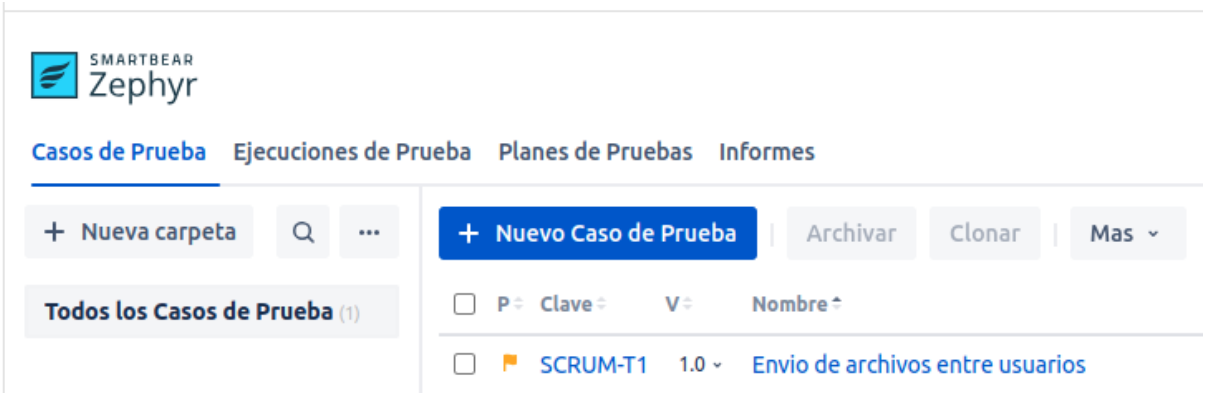
Zephyr es un plugin de Jira que permite gestionar pruebas de forma más avanzada. Permite crear casos de prueba como entidades independientes, organizarlos en ciclos, ejecutarlos, registrar resultados y vincularlos a historias o bugs.

¿Cómo instalar Zephyr?

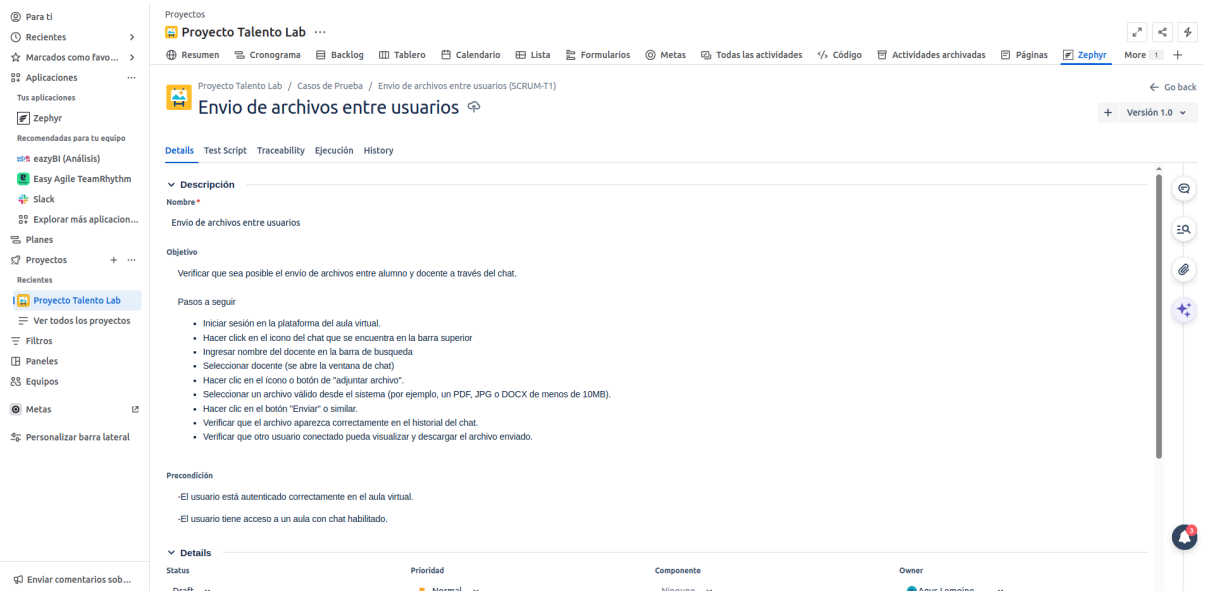
1. Desde Jira, ir a “Apps” > “Find new apps”.
2. Buscar **Zephyr Squad** y hacer clic en “Try it free”.
3. Seguir los pasos para activarlo en tu proyecto.

¿Cómo crear un Test Case con Zephyr?

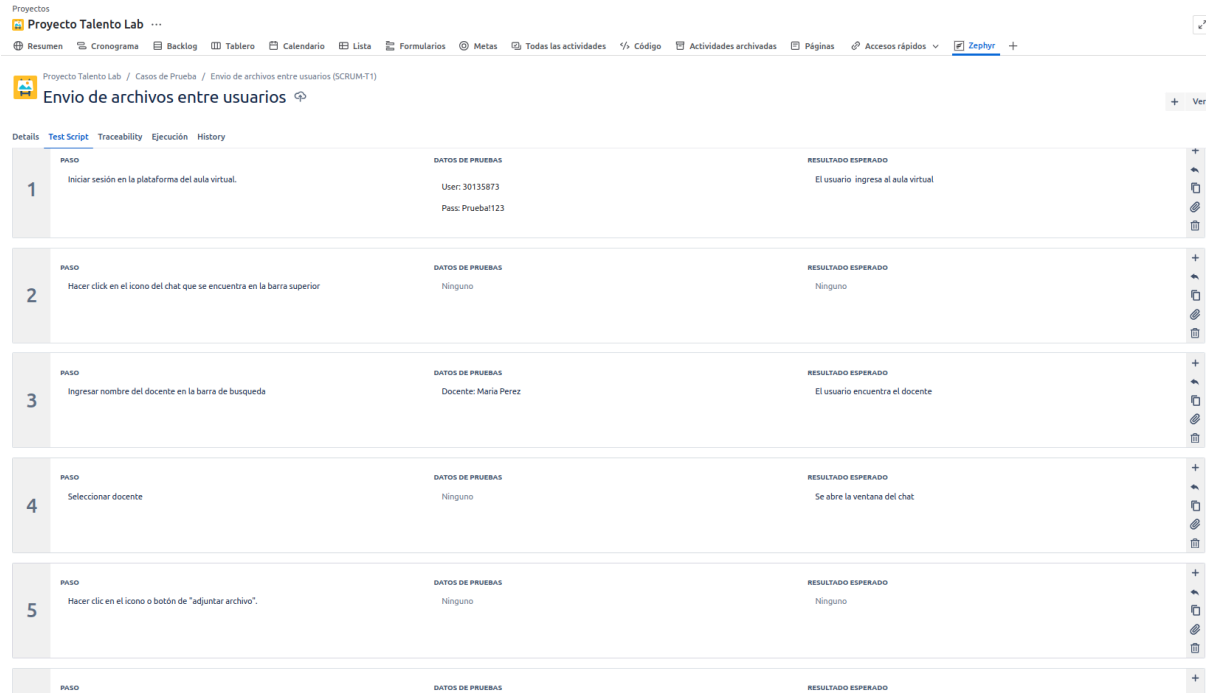
1. Ir a la pestaña **Test** del proyecto Jira.
2. Hacer clic en **“Create a Test” o “Nuevo caso de prueba”**.



3. Completar los campos:
    - **Summary:** Verificar envío de archivos entre alumnos y docentes vía chat.
    - **Description:** Validar que los usuarios puedan enviar y recibir archivos correctamente dentro del sistema de mensajería interna.



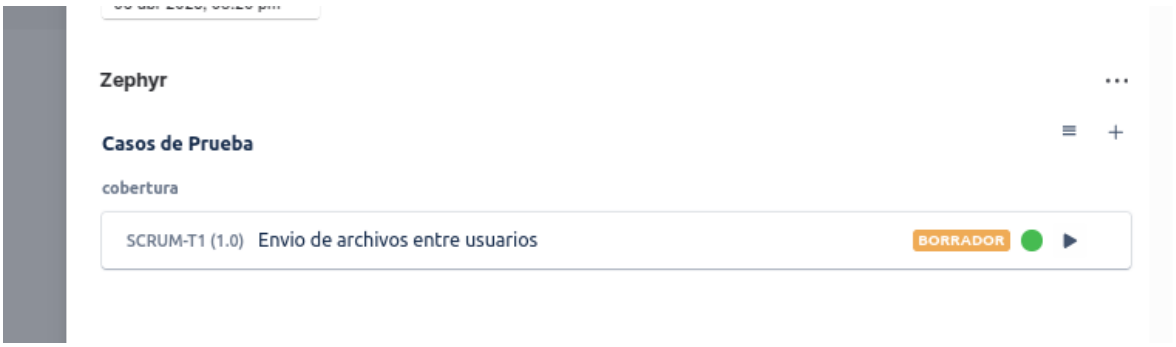
- **Test Steps:**
  - Paso 1: Iniciar sesión en la plataforma del aula virtual.
  - Paso 2: Hacer clic en el icono del chat en la barra superior.
  - Paso 3: Ingresar el nombre del docente en la barra de búsqueda.
  - Paso 4: Seleccionar el docente (se abre la ventana de chat).
  - Paso 5: Hacer clic en el ícono o botón de "adjuntar archivo".
  - Paso 6: Seleccionar un archivo válido desde el sistema (por ejemplo, un PDF, JPG o DOCX de menos de 10MB).
  - Paso 7: Hacer clic en el botón "Enviar".
  - Paso 8: Verificar que el archivo aparezca correctamente en el historial del chat.
  - Paso 9: Verificar que el otro usuario (docente) conectado pueda visualizar y descargar el archivo enviado.
  - **Resultado esperado:** El archivo se visualiza en el chat y es descargable por el receptor.



Guardar el caso.

Vinculación del Test Case a una Task o Bug

1. Abrir el ticket (Story, Task o Bug) al que queremos asociar el test.
2. En la sección “Zephyr Test”, hacer clic en “Link Test”.
3. Buscar el test creado y seleccionarlo.
4. El test ahora estará vinculado y visible desde el ticket.



Ejecutar el Test

1. Desde la pestaña “Test Cycles”, crear un nuevo ciclo de pruebas.
2. Agregar los test cases correspondientes.
3. Ejecutar uno por uno y marcar estado (Passed, Failed, Blocked).
4. Si algún test falla, podés crear un bug directamente desde la pantalla de ejecución y se vinculará automáticamente al paso fallido.

💡 **Ventajas de Zephyr**:

- Mejora la trazabilidad entre historias, pruebas y defectos.
- Permite reportes de cobertura de pruebas.
- Centraliza todo el ciclo de testing dentro de Jira sin herramientas externas.

Vinculación de Test Cases con User Stories

Vincular los test cases con las historias de usuario es una práctica clave en QA porque permite mantener **trazabilidad** entre lo que se desarrolla y lo que se prueba. Esto garantiza que cada funcionalidad tenga al menos una validación asociada.

¿Por qué es importante?

- Permite saber qué funcionalidades están cubiertas por pruebas.
- Facilita el análisis de cobertura.
- Mejora la comunicación entre QA y desarrollo.
- Ayuda a justificar el cierre de historias durante la revisión del sprint.

¿Cómo vincular test cases a user stories en Jira con Zephyr?

1. Crear la **User Story** (por ejemplo: “Como usuario, quiero enviar archivos por chat”).
2. Crear uno o más **test cases** asociados a esa historia.
3. Abrir la historia desde el tablero o backlog.
4. Buscar la sección **“Zephyr: Tests”** (si está activado el plugin).
5. Hacer clic en **“Link Test”**.
6. Buscar el test case por nombre o código y vincularlo.

💡 También se puede hacer desde el test case: ir al panel de vínculos y agregar un “relates to” o “tests” apuntando a la historia.

Registro de Resultados de Pruebas

Una vez creados y ejecutados los test cases, es fundamental **registrar los resultados** de cada uno. Esto permite evaluar la calidad del producto y tomar decisiones informadas.

Tipos de resultados posibles:

- **Passed:** El comportamiento fue el esperado.
- **Failed:** El sistema no se comportó como se esperaba.
- **Blocked:** El test no pudo ejecutarse por un impedimento externo (por ejemplo, caída del entorno).
- **Not Executed:** El test aún no se ejecutó.

¿Dónde se registran?

1. Ir a la sección **Test Cycles** en Zephyr.
2. Seleccionar el ciclo de pruebas activo (ej. “Sprint 3 - QA”).
3. Elegir el test case a ejecutar.
4. Completar:
    - Resultado (Passed/Failed/Blocked).
    - Evidencia (captura de pantalla, logs, comentarios).
    - Tiempo de ejecución (opcional).
5. Guardar.

💡 Si un test falla, Zephyr permite **crear un bug directamente desde el paso fallido** para mantener el vínculo automático.

Gestión del Ciclo Básico de un Bug

Cuando un test falla, se detecta un defecto (bug) en el sistema. Saber gestionar correctamente el ciclo de vida del bug es esencial para el trabajo de QA.

Fases del ciclo de vida de un bug:

1. **Detección:**
    - QA detecta una falla mientras ejecuta un test.

- Se recopila toda la información necesaria para reportarlo.

2. **Reporte:**

    - Se crea un ticket en Jira con tipo “Bug”.
    - Se completa la siguiente información:
        - Título descriptivo.
        - Entorno (navegador, sistema operativo, etc.).
        - Pasos para reproducir.
        - Resultado esperado vs. obtenido.
        - Capturas o video.
        - Prioridad y severidad.

3. **Asignación:**

    - El bug se asigna a un desarrollador responsable.
    - Se puede etiquetar con el módulo al que pertenece.

4. **Resolución:**

    - El desarrollador arregla el problema.
    - Cambia el estado a “Ready for QA” o similar.

5. **Validación:**

    - QA reejecuta el test relacionado.
    - Si el problema está resuelto, el bug se cierra.
    - Si persiste, se reabre.

6. **Cierre:**

    - Una vez validado, se cambia el estado a “Done” o “Closed”.
    - Se adjunta evidencia de la validación.

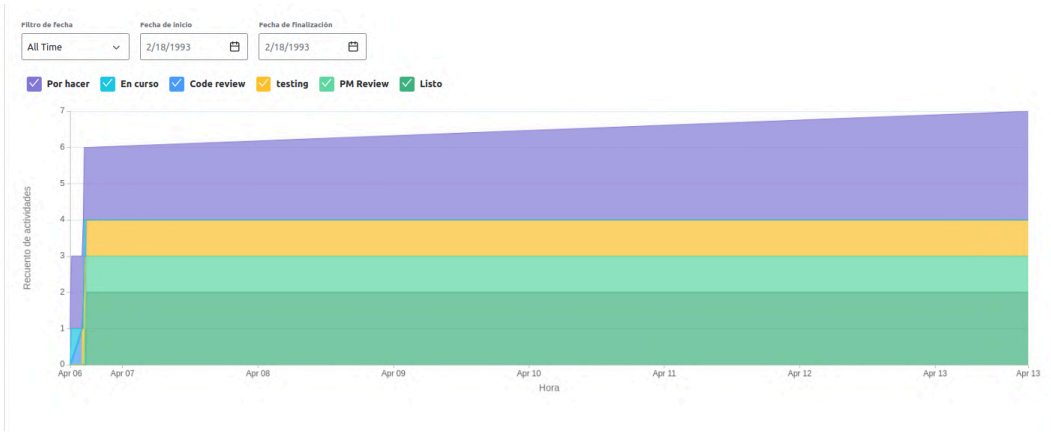
Diagrama de Flujo Acumulado y Sprint Report

¿Qué son y para qué sirven?

- **Diagrama de Flujo Acumulado (Cumulative Flow Diagram, CFD)**
  Un gráfico que muestra la evolución del número de issues (historias, tareas, bugs) en cada estado del flujo (To Do, In Progress, Testing, Done) a lo largo del tiempo.
  Objetivo:

    - Detectar cuellos de botella (columnas que crecen sin avanzar).
    - Verificar estabilidad del proceso (líneas de cada estado paralelas y suaves).
    - Medir ritmo de entrega y tasa de acumulación.
    - Eje horizontal: días del sprint.
    - Eje vertical: cantidad de tickets.
    - Áreas de color: cada estado (cuelga el color con la columna del tablero).

        - Si el área “In Progress” crece demasiado, indica bloqueo o sobrecarga.
        - Si las líneas de “To Do” y “Done” se mantienen paralelas y estables, el flujo es saludable.



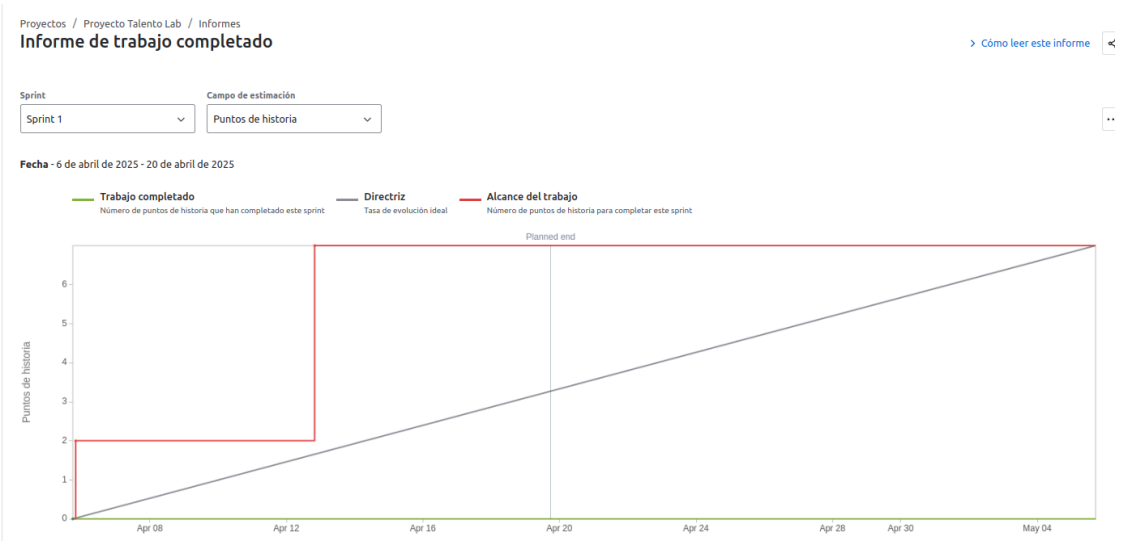
- **Sprint Report - Informe del trabajo completado**
  Un informe generado al cerrar el sprint que resume:

    - Historias planificadas vs. completadas
    - Puntos de historia alcanzados
    - Issues no terminados que se pasan al siguiente sprint
    - Gráfica Burn-Down/Burn-Up y métricas clave (velocidad, alcance).

Objetivo:

- Evaluar cumplimiento de la Definition of Done.
- Planificar mejoras en estimación y alcance para siguientes sprints.
- Facilitar la retrospectiva mostrando dónde se acumuló trabajo pendiente.
- Línea ideal (progresión constante) vs. línea real (cómo avanzó el equipo).

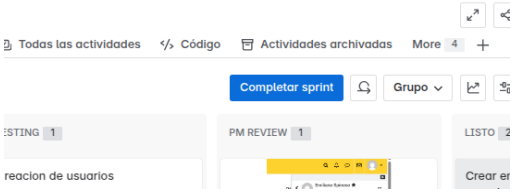
- Si la línea real va por encima, hay retraso; si va muy por debajo, puede haber sobrecapacidad o historias desglosadas insuficientemente.



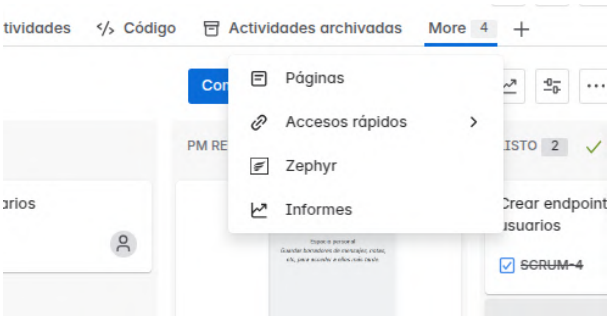
¿Cuándo y cómo se generan?

1. **Al completar el sprint**

   - Dentro del tablero Scrum de Jira, hacer clic en **“Complete Sprint”**.
   - Jira solicita confirmar el paso de issues pendientes al backlog o al siguiente sprint.

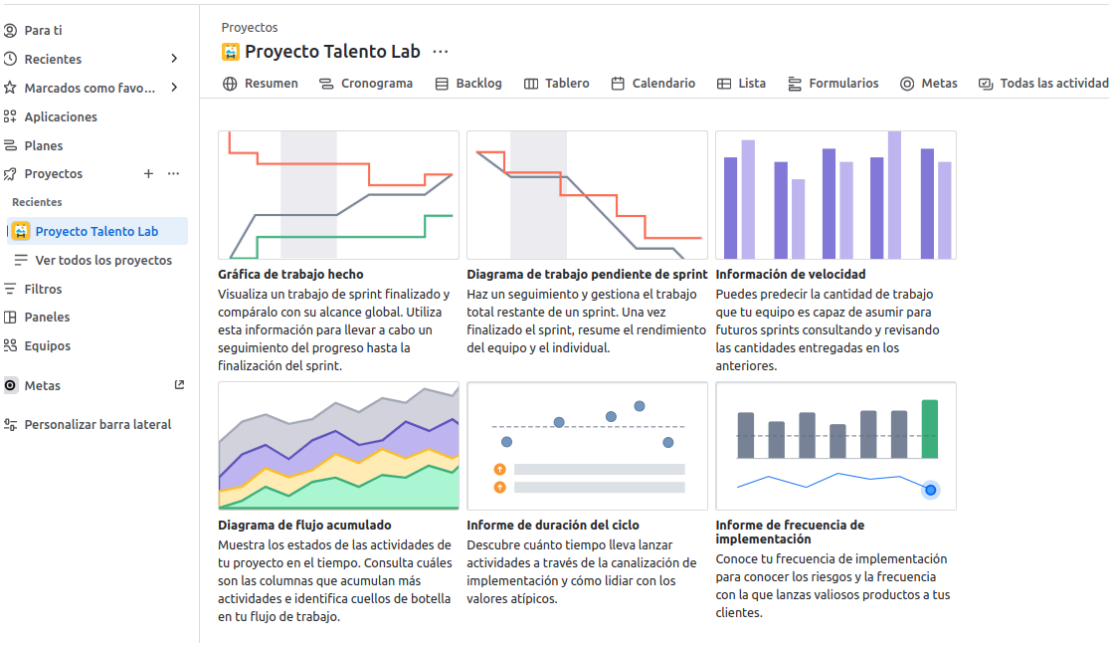


2. **Acceso a los reportes**



   - Tras cerrar el sprint, en la sección **“Reports”** del proyecto, seleccionar:
     - **Cumulative Flow Diagram**

- **Sprint Report**
  - Configurar el rango de fechas del sprint que acaba de finalizar.



Próximos pasos

Al terminar esta clase, ya contás con los conocimientos fundamentales para integrarte en un equipo ágil de QA, diseñar casos de prueba efectivos, y gestionar el ciclo completo de testing utilizando Jira y Zephyr. Para seguir desarrollando tus habilidades, te recomendamos:

Practicar en proyectos reales o simulados
Continuá usando tu cuenta de Jira para practicar la creación de tickets, organización de tableros, ejecución de casos de prueba y seguimiento de bugs. Podés inventar funcionalidades de una app o replicar flujos reales de plataformas conocidas.

Explorar dashboards y reportes en Jira
Comenzá a usar las herramientas de métricas integradas (como burn down chart, velocity report o control chart) para obtener insights del progreso y calidad del proyecto.

Participar de daily meetings simuladas
Simulá una reunión de equipo donde compartas avances, bloqueos y próximos pasos. Es una buena práctica para integrar QA con el resto del equipo Scrum.

Aprender más sobre herramientas complementarias
Explorá otras herramientas que suelen integrarse con Jira, como:

- **Confluence:** documentación colaborativa.
- **Slack o Microsoft Teams:** comunicación entre equipos.

- **TestRail o Xray:** alternativas a Zephyr para gestión avanzada de pruebas.

Empezar a vincularte con el mundo Automation
Si te sentís cómodo con la lógica de trabajo de QA, podés dar el siguiente paso y comenzar a automatizar esos mismos casos de prueba usando herramientas como Pytest, Selenium o Postman + Newman.

Material complementario

Aquí te dejamos algunos recursos útiles para seguir profundizando:

- **Guía oficial de Zephyr Squad**
- **Guía sobre el ciclo de vida de los bugs en Jira**
- **Curso básico de Jira en español**
- **Buenas prácticas para escribir test cases**

¡Trabajando en Talento Lab!

Silvia y Matías te esperan en la sala de reuniones con un nuevo desafío. El equipo de Talento Lab está en plena ejecución del Sprint 6 y se aproxima una **entrega clave** del módulo de mensajería entre usuarios.

Silvia te comenta:

"En este sprint queremos validar todas las funcionalidades que impactan directamente en la experiencia del usuario: el envío de archivos por chat, las notificaciones en tiempo real y el historial de mensajes. Como equipo de QA, su tarea es diseñar las pruebas, ejecutarlas y dejar todo documentado."

Matías agrega:

"Necesitamos que simulen el ciclo completo de QA: desde que la historia se mueve a Testing, hasta que se genera el reporte final. Esto incluye test cases funcionales, validaciones no funcionales, y por supuesto, la gestión de cualquier bug que encuentren en el camino."

Además, ambos insisten en que todas las tareas deben cumplir con la Definition of Done (DoD) establecida por el equipo, que incluye cobertura de pruebas, validación de criterios de aceptación y que no queden bugs abiertos al cierre del sprint.

Tu trabajo será clave para garantizar que todo lo que llegue a producción esté verificado, trazado y correctamente documentado.

Ejercicio práctico:

Gestionar el ciclo de QA en Jira para el proyecto Talento Lab

Objetivo: Tomar el backlog y los artefactos de las Clases 1–8 (User Stories, Test Cases y Bugs) referidos a el sitio: https://talentolab-test.netlify.app/ y ejecutar un Sprint completo de pruebas en Jira + Zephyr.

1. **Configura tu Sprint**
   - Crea un nuevo Sprint de 1 semana en tu proyecto **Talento Lab Web – QA**.
   - Arrastra al Sprint **4 User Stories** ya definidas (por ejemplo, las de mensajería: envío de archivos, notificaciones, historial, reporte de actividad).

2. **Mueve las Stories al flujo Scrum**
   - Deja cada Story en **TO DO** al inicio.
   - Simula que comienza el desarrollo, muévelas a **IN PROGRESS**.
   - Cuando estén implementadas, colócalas en **TESTING** para tu turno de QA.

3. **Crea y vincula Test Cases (Zephyr)**
   - Para cada Story, ve a la pestaña **Test** y haz **Create a Test**:
     1. Resume: *"Verificar [funcionalidad] en mensajería"*
     2. Steps y Resultado esperado (uno funcional + uno no funcional).
   - En la Story, haz **Link Test** para asociar esos casos.

4. **Ejecuta los Test Cases**
   - Abre **Test Cycles**, crea un ciclo "Sprint X – QA".
   - Agrega todos los Test Cases y ejecútalos uno por uno, marcando **Passed/Failed/Blocked**.
   - Adjunta evidencia (capturas o comentarios) en cada ejecución.

5. **Gestiona los Bugs**
   - Si un Test falla, desde Zephyr genera un **Bug** enlazado automáticamente al paso fallido.
   - Completa Title, Description, Steps, Expected vs. Actual, Severity/Priority, Assignee.
   - Mueve ese Bug por los estados: **TO DO → IN PROGRESS → READY FOR QA → DONE**.

6. **Cierre del Sprint**
   - Una vez todos los test cases estén en **Passed** y todos los Bugs en **Closed**, mueve las Stories a **DONE**.
   - Comprueba que no queden tickets en **TESTING** ni Bugs abiertos.

7. **Demo del ciclo QA**
   - Prepara un breve pantallazo del tablero mostrando columnas y tickets en cada estado.
   - Resume en 3 puntos:
     1. Nº de Test Cases ejecutados vs. planificados
     2. Bugs encontrados y resueltos
     3. Stories cerradas según la Definition of Done

Entrega Final de Proyecto.

Objetivo general.

Consolidar y aplicar todas las prácticas de testing vistas en el curso para garantizar la calidad de la plataforma Talento Lab (web y mobile) disponible en https://talentolab-test.netlify.app dentro de un entorno ágil gestionado en Jira y Zephyr.

Descripción general.

1. Integrá todos los artefactos generados desde la **Clase 1** hasta la **Clase 15** en un único proyecto QA sobre https://talentolab-test.netlify.app.
2. Desde el storytelling y el desglose ágil de requerimientos, pasando por la planificación y ejecución de pruebas (manuales y mobile), hasta la gestión de defectos y métricas de cobertura.
3. Organizá el backlog y simulá un sprint completo de **1 semana** en Jira + Zephyr, documentando cada paso con evidencias alojadas en la plataforma.

Descripción detallada.

1) Storytelling (Clase 8)

- Redactar la narrativa completa de un usuario real navegando en el sitio (e.g.: registrarse → carga de CV).
- Incluir al menos un párrafo que describa cómo el usuario accede a “Carga tu CV” y recibe confirmación.

2) Requerimientos Ágiles (Clases 2–3)

- Definir **2 Épicas**, **4 Features** y **8 User Stories** con sus criterios de aceptación:
  - Funcionales
  - No funcionales
  - Manejo de errores
- Todas las historias deben referirse a funcionalidades presentes en talentolab-test.netlify.app

3) **Plan de Pruebas** (Clase 4)
    - Documentar alcance, objetivos, estrategia (manual y mobile), recursos y cronograma en un documento formal.

4) **Casos de Prueba** (Clase 5)
    - Crear **mínimo 10 Test Cases** (funcionales, no funcionales y exploratorios) en hoja de cálculo, indicando:
        - ID
        - User Story asociada
        - Pasos
        - Datos de prueba
        - Resultado esperado

5) **Ejecución de Pruebas**
    - **Manual** (Clases 4–5): Incluir screenshots de talentolab-test.netlify.app para cada Test Case ejecutado.
    - **Mobile/Desktop** (Clase 13): Completar una tabla de responsividad para **4 módulos** críticos (navegación, carga de CV, servicios, clientes) con capturas en Desktop/Tablet/Móvil.

6) **Gestión de Defectos (Clases 6 y 15)**

Durante la ejecución de tus pruebas sobre Talento Lab, identifica y reporta de 3 a 5 defectos reales. Para cada uno, debes:

- **Describir el proceso de reporte y ciclo de vida en Jira/Zephyr**
    - Desde la **detección** en un Test Case hasta el **cierre** definitivo.
    - Incluir los estados: *Open* → *In Progress* → *Code Review* → *Testing* → *PM Review* → *Closed*.
- **Completar la "Tabla de Bug Reports"** con los siguientes campos:
    - Bug ID
    - User Story asociada
    - Resumen
    - Pasos para reproducir
    - Resultado esperado

        - Resultado obtenido
        - Severidad
        - Prioridad
        - Estado actual
        - Test Case vinculado
        - Evidencia (captura o log)
    - **Ilustrar el ciclo de vida de un bug** (p. ej. BUG-002) describiendo brevemente cada transición de estado.

7. Métricas y Cobertura (Clases 7 & 12)

- Calcular:
    - % de ejecución de test cases
    - Tasa de defectos
    - Distribución por severidad
    - Desviación del tiempo estimado
    - Cobertura de pruebas sobre al menos 3 funcionalidades
- Comparar con estándares de la industria (< 20 % defectos, > 90 % ejecución, ≤ $ 80 costo/defecto) y proponer mejoras.

8. Jira & Zephyr (Clases 14 & 15)

    - Incluir capturas del backlog, Épicas, User Stories, Tasks, Sub-tasks.
    - Configurar y ejecutar un sprint de 1 semana con 4 User Stories. (Incluir capturas)
    - Crear y vincular Test Cases en Zephyr; ejecutar el ciclo de pruebas y adjuntar evidencia.
    - Al finalizar el sprint, clicar **Complete Sprint** y generar:
        - **Diagrama de Flujo Acumulado**
        - **Sprint Report**

Entregables.

Adjuntar en el campus virtual un link a drive público, que contenga:

1. **Documento PDF** que contenga:
    - Storytelling narrativo
    - Tabla de requerimientos ágiles
    - Plan de Pruebas completo
    - Métricas y análisis de cobertura con cálculos comparativos
    - Tabla de responsividad (Desktop/Tablet/Móvil) con capturas y observaciones
    - Capturas del Diagrama de Flujo Acumulado y Sprint Report
    - **Referencias cruzadas a las hojas del archivo XLSX** con los Test Cases, Bug Reports y reportes de Jira.

2. **Archivo XLSX**:

    Debe contener **6 hojas**, cada una con la siguiente información:

Hoja	Contenido	Detalles
1	Requerimientos Ágiles (Clases 2–3)	2 Épicas, 4 Features, 8 User Stories (funcionales, no funcionales, manejo de errores), todos vinculados a funcionalidades reales de talentolab-test.netlify.app
2	Plan de Pruebas (Clase 4)	Alcance, objetivos, estrategia (manual + mobile), recursos y cronograma
3	Casos de Prueba (Clase 5)	Mínimo 10 test cases: ID, historia asociada, pasos, datos de prueba, resultado esperado
4	Ejecución de Pruebas (Clases 4, 5 y 13)	Screenshots por Test Case + tabla de responsividad con capturas por dispositivo
5	Gestión de Defectos (Clases 6 y 15)	Bug Reports estructurados (3 a 5 defectos) con ciclo de vida completo en Jira/Zephyr
6	Métricas y Cobertura (Clases 7 & 12)	Porcentajes, distribución por severidad, comparación con benchmarks, y propuesta de mejora

3. **Capturas de Jira**:
    - Backlog con Épicas/Stories (Clase 14)
    - Tablero Scrum antes y después del sprint (Clase 15)
    - Zephyr Test Cycle con resultados y vínculos a Bugs (Clase 15)

Criterios de evaluación

- **Integración y coherencia**
  Todos los artefactos deben estar vinculados y referir al mismo proyecto en talentolab-test.netlify.app.
- **Cobertura y calidad de pruebas**
  Incluye pruebas funcionales, no funcionales y mobile con ≥ 80 % cobertura en funcionalidades críticas.
- **Gestión de defectos**
  Bugs correctamente documentados y con ciclo de vida completo en Jira/Zephyr.
- **Uso de herramientas**
  Configuración y exportación adecuadas de Jira y Zephyr .
- **Evidencias y métricas**
  Capturas claras; métricas calculadas y comparadas con estándares; propuestas de mejora bien fundamentadas.
- **Presentación y claridad**
  Documentación profesional, estructurada y justificada en cada sección.

Buenos Aires aprende

Agencia de Habilidades para el Futuro

Buenos Aires Ciudad