Clase 04. El Testing en el Desarrollo del Software

# Funciones de un Tester Manual

### ¿Cuándo probamos?

El alcance de nuestro trabajo recorre todas las etapas del desarrollo de software, ya que estamos desde la creación de la documentación, hasta la implementación del producto.

Hay que comprender que el tester siempre está realizando pruebas, y nuestro alcance es completo al producto.

### Comienzo del desarrollo

En etapas tempranas del desarrollo, nuestro trabajo va a ser el de leer documentación, participando de reuniones que decidan el funcionamiento para dar nuestro punto de vista experto, y también para hacer los pedidos de herramientas necesarias para hacer las pruebas.

### Cierre del desarrollo

Una vez que los desarrolladores nos entregan el producto, y comienzan nuestras pruebas dinámicas, estas se extienden hasta la salida del producto, ya que las pruebas se deben realizar tanto antes de implementar, como post implementación.

Si algo funciona mal, en post implementación, somos nosotros quienes debemos informar.

## Pruebas

### Realizar pruebas

La tarea de realizar pruebas, puede sonar simple, y muchas veces realmente lo es, pero en algunas circunstancias las pruebas pueden ser complejas y llevar mucho tiempo.

Esto es amplio y con la experiencia, vamos encontrando nuevas formas de realizar pruebas. Incluso encontrar distintas formas de realizar las mismas pruebas.

### Prueba Estática

Este es el comienzo de nuestras tareas, que suceden incluso antes del comienzo del desarrollo.

Al momento de realizar pruebas estáticas, debemos estar con seguridad de que toda la documentación que estamos leyendo fue consensuada entre las partes, e incluso podemos chequear que la documentación no esté representada en el producto final y sobre esto elevar un bug.

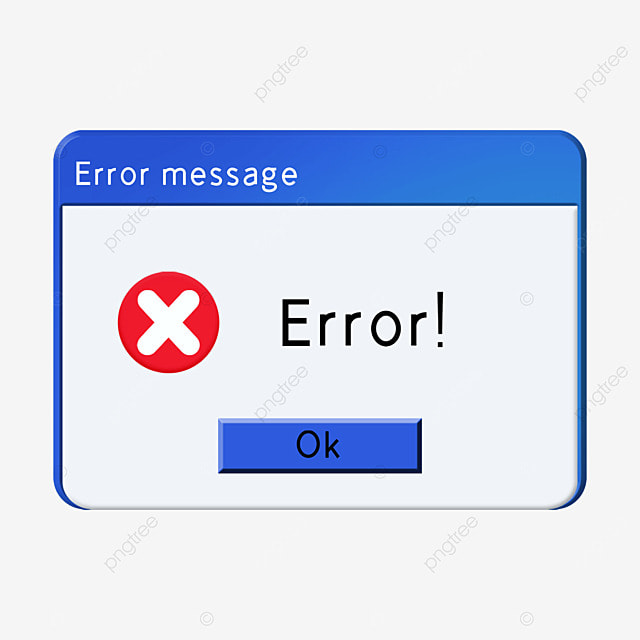
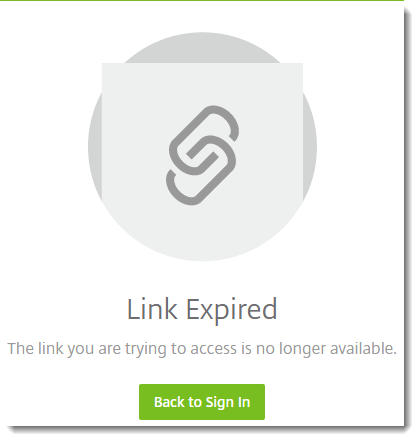
### Prueba Dinámica

Todo tipo de tarea que sea confirmar funcionalidades, se denomina prueba dinámica. Existen tantas formas de realizar pruebas dinámicas, como imaginación tengamos. No existe un límite de trabajo para esto.

Ejemplo de prueba dinámica:

* Que un link funcione.
* Poder loguearse en un sistema.
* Que se envíe un mail.
* Que una página se cargue.
* Que una imagen se vea bien.

Ejemplo de prueba dinámica



### Presentación de Resultados

Una vez finalizada la etapa de pruebas, se realiza una presentación de resultados. Esto incluye, como se indicó previamente, cantidad de bugs encontrados, tiempo que llevo, cantidad de casos completos y otras condiciones más.

Los resultados ayudan a comprender el trabajo que se realizó, y es nuestra herramienta para demostrar el trabajo realizado.

### Tipos y Niveles de Testing

En una primera idea, los tipos y niveles, pueden mezclarse, comprendiendo a qué tipo de actividad estamos realizando.

Cada tipo de actividad de testeo tiene un tipo y nivel, y su función es comprender el tipo de tarea que se está realizando.

### Otras Responsabilidades

Existen otras responsabilidades más invisibles que velan por el software y el equipo. El QA tiene que participar de todas las etapas del desarrollo, y también de las reuniones que velan por el futuro del producto.

Un QA también tiene que estar en relación y coordinación con todos los profesionales del equipo, para buscar incluso, la calidad en el equipo.

# Pruebas a lo largo del ciclo de vida del desarrollo de software

## Modelos del ciclo de vida del desarrollo del software

### Introducción

El ciclo de vida del desarrollo del software comprende las etapas por las cuales un proyecto de software avanza, desde el análisis de los requerimientos del cliente, hasta el despliegue de la solución final.

### Definición de Proyecto de Software

Un proyecto es un esfuerzo temporal que se lleva a cabo para crear un producto, servicio o resultado único.

### Características de un Proyecto de Software

* Únicos: todos los proyectos por similares que sean, tienen características que los hacen únicos.
* No ambiguos: deben ser claros, dado que son los responsables de guiar el desarrollo del proyecto.
* Medibles: a fin de poder determinar el avance sobre el proyecto.
* Alcanzables: es decir que los mismos puedan ser realizados por el equipo.

### Ciclos de vida del desarrollo del software

Existen varios modelos o paradigmas de desarrollo, en los cuales nos podemos apoyar para la realización de software.

Los ciclos de vida sirven como guía para llevar adelante un proyecto y nos ayudan a responder las siguientes preguntas:

* ¿Cuáles de las tareas deben ser rastreadas?
* ¿Que tipo de progreso se ha realizado?

## Desarrollo y pruebas de Software

### Introducción

Como tester es importante conocer la metodología aplicada al proyecto, ya que es el marco de trabajo que te indicará qué tareas de pruebas se deben ejecutar y cuales técnicas de prueba utilizar.

### Modelos y Metodologías de desarrollo

Los modelos y metodologías de desarrollo de software son un conjunto de técnicas y métodos organizativos que se aplican para diseñar soluciones de software informático.

Una metodología de desarrollo de software brinda al equipo de trabajo un marco para construir aplicaciones de manera eficiente y rigurosa, garantizando un producto cercano al esperado.

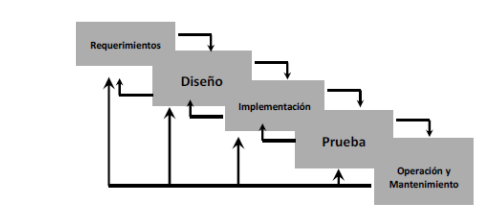
Si no se desarrolla a partir de una metodología, el resultado final será impredecible y no se podrá controlar el avance del proyecto.

## Los modelos de desarrollo se clasifican en:

### Modelo de desarrollo secuencial

Son aquellos donde cualquier fase del proceso de desarrollo comienza una vez que la anterior finalizó. Por ejemplo el modelo en cascada.

Satisfacen la totalidad de los requerimientos del cliente, sin embargo esto puede llevar meses incluso años para su entrega final.

Ejemplo del modelo de desarrollo secuencial

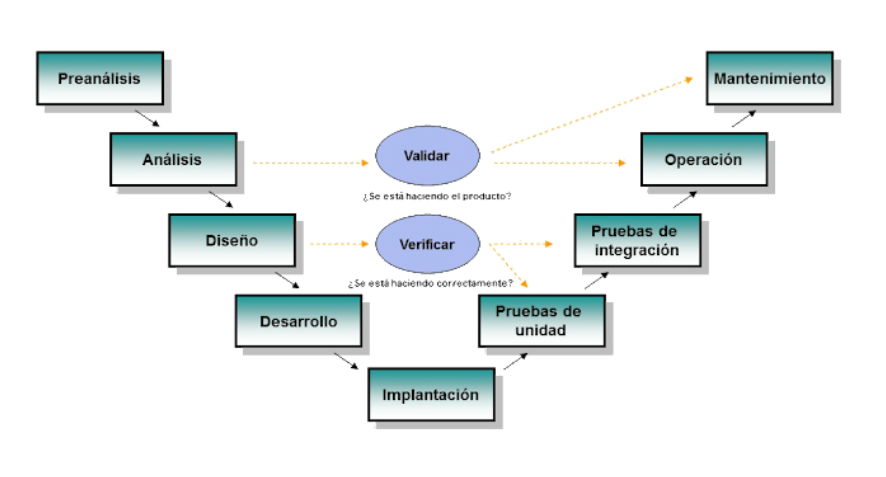
En este modelo, las pruebas comienzan una vez que el desarrollo termina

### Modelo de desarrollo iterativo e incremental

Son aquellos que comienzan desarrollando una especie de versión “ligth” de los requerimientos del sistema, y en cada iteración, se agregan nuevas funcionalidades.

Por lo tanto, las etapas de Requerimientos, Análisis, Diseño, Implementación y Prueba se realizan en pequeñas partes agrupando funcionalidades, y cada vez que terminemos una iteración, se entrega una versión funcional del sistema.

Ejemplo del modelo iterativo e incremental - Modelo en V



#### Modelo en V

Es una evolución del modelo en cascada.

Al final de cada iteración, se obtiene un producto funcional.

En este modelo, se integra el proceso de pruebas al proceso de desarrollo, es decir, para cada etapa de desarrollo tenemos una etapa de pruebas asociada.

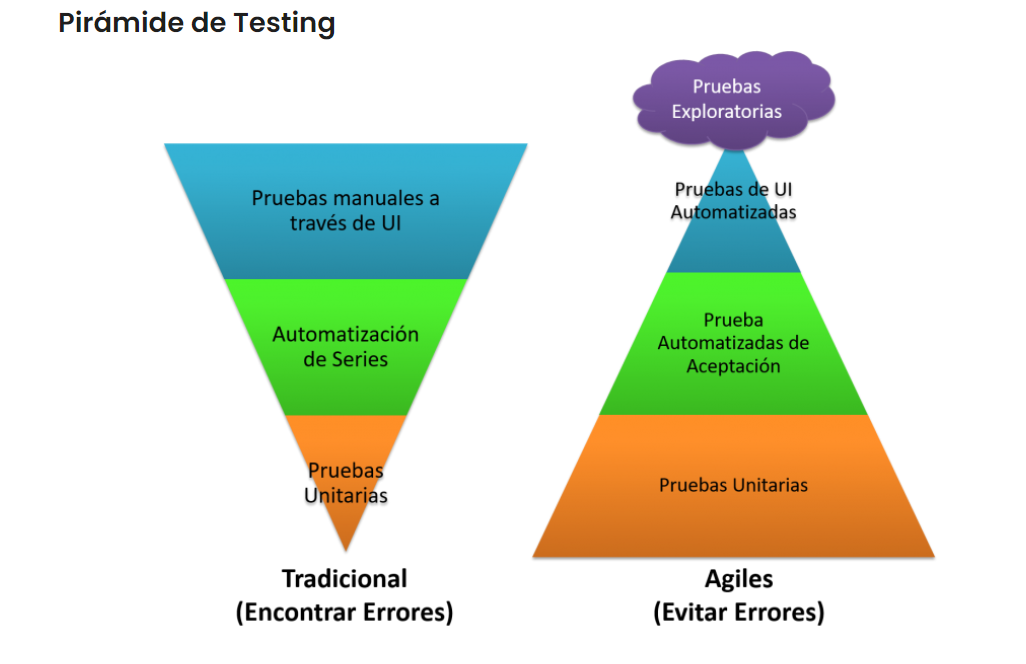
Por ejemplo, en la etapa de análisis de requerimientos del proyecto, la Validación de los requerimientos documentados permite garantizar que se va a construir el software idóneo para satisfacer la necesidad del cliente y a su vez diseñar las pruebas de aceptación.

Y del mismo modo, cada fase de desarrollo tiene su respectiva fase de pruebas asociadas, a fin de encontrar defectos y evitar que se propaguen a etapas posteriores donde es más costoso repararlos.

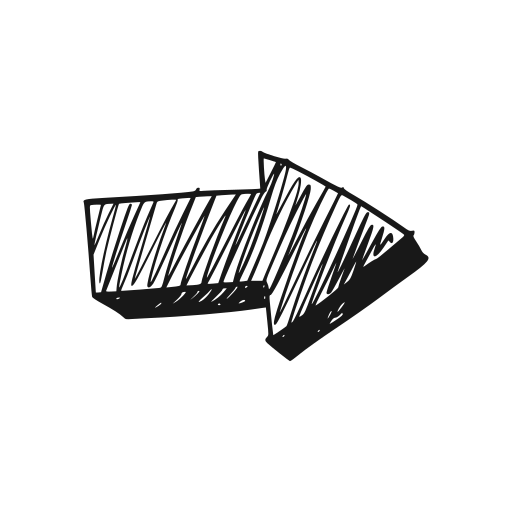
#### Scrum

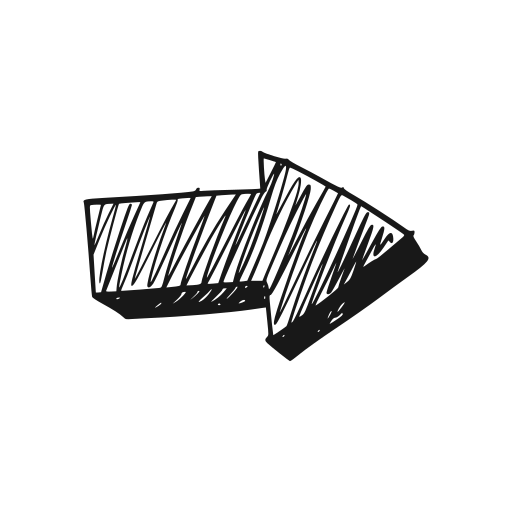
Es otro modelo Iterativo e incremental, tiene como objetivo hacer entregas de funcionalidades terminadas en periodos cortos de tiempo.

Cada iteración tiende a ser relativamente corta, estamos hablando de horas, días, o un par de semanas .



Fuente: [enlace](https://www.cgtscorp.com/language/es/agile-testing/)

En la forma tradicional (pirámide de la izquierda), la gran mayoría de las pruebas son manuales y funcionales, pudiendo existir algún pequeño grado de automatización y de pruebas unitarias por parte de los desarrolladores.

La pirámide de la derecha representa cómo se ejecuta el Software Testing en Agile, donde la gran mayoría de las pruebas son unitarias automatizadas, de aceptación automatizadas y de interfaz gráfica automatizadas, buscando reducir al mínimo las pruebas funcionales manuales.

### Metodologías de desarrollo

| **Metodologías Ágiles** | **Metodologías tradicionales** |
| --- | --- |
| Desarrollo iterativo | Análisis, diseño, implementación y pruebas |
| Ágil, dinámico y muy flexible. Preparados para cambios durante el proyecto | Estricto, rígido y con cierta resistencia al cambio |
| El cliente es parte del equipo de desarrollo | El cliente interactúa con el equipo de desarrollo mediante reuniones |
| Grupos pequeños(menos de 10 integrantes) | Grupos grandes y posiblemente distribuidos |
| geom\_quantile() | Regresión de cuantiles |
| Pocos roles | Más roles |
| Pocos artefactos | Más artefactos |

### Buenas prácticas en la ejecución de las pruebas

El modelo escogido debe corresponder a los objetivos del proyecto, pero hay ciertas características que debemos tener en cuenta al momento de elegirlo, ya que definen las buenas prácticas en la ejecución de las pruebas.

**Ciclos con pruebas de Validacion y Verificacion:** cada ciclo de desarrollo debe contar con pruebas de validacion y verificacion

**Actividades de desarrollo con actividades de pruebas:** cada actividad de desarrollo tiene una actividad de pruebas relacionada. No importa que modelo se elija para el proyecto, las etapas de pruebas deben comenzar en las etapas tempranas del ciclo de vida, respetando el principio de pruebas tempranas

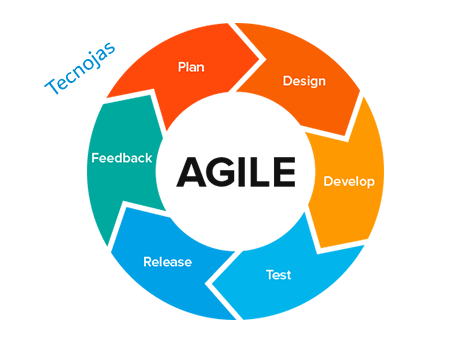
**Cada nivel de pruebas tiene objetivos específicos:** por ejemplo, las pruebas de componentes tienen como objetivo identificar posibles defectos, mientras que las pruebas de aceptación buscan crear confianza en el usuario sobre el uso del software.

**Análisis y diseño de niveles de pruebas sincronizado a las actividades de desarrollo:** el análisis y diseño de cada nivel de prueba debe comenzar durante la actividad de desarrollo que corresponda. Esto quiere decir que las actividades relacionadas con las pruebas de componentes deben iniciarse durante la etapa de desarrollo de componentes, y así cada prueba cuando corresponda.

**Participar en la definición de requerimientos y diseño:** los testers deben participar en la definición y refinamiento de requerimientos y diseño

## Círculo de trabajo ágil

1. **Planificación:** es la etapa inicial de planificación en donde todas las partes involucradas en la metodología ágil, deben trabajar en conjunto. Una vez realizado el análisis de los requisitos, el siguiente paso es definir y documentar claramente los requisitos del producto y obtener la aprobación del cliente o de los analistas.
2. Diseño: otra de las fases más importantes que servirán para definir la arquitectura y tecnologías que se utilizarán como también el diseño de interfaz de usuario del software a construir.
3. Desarrollo: En esta etapa comienza el desarrollo real y se construye el producto. Si el diseño se realiza de forma detallada y organizada, la generación de código puede llevarse a cabo sin demasiados problemas.
4. Testing: esta etapa se refiere a la fase de prueba del producto, en la que se informa de los defectos del producto, se rastrean, se arreglan y se vuelven a probar, hasta que el producto alcanza los estándares de calidad definidos.
5. Release (Lanzamiento): En esta etapa del Agile Testing, el equipo va a revisar las características de todo aquello que se haya desarrollado e implementado hasta el momento. Y se evaluará si ya se encuentra listo para el cliente o no.
6. Feedback: Finalmente, la metodología ágil se encargará de reunir las aportaciones de las partes interesadas y los usuarios, y de esta forma se obtendrá una retroalimentación general para el próximo ciclo de implementación.



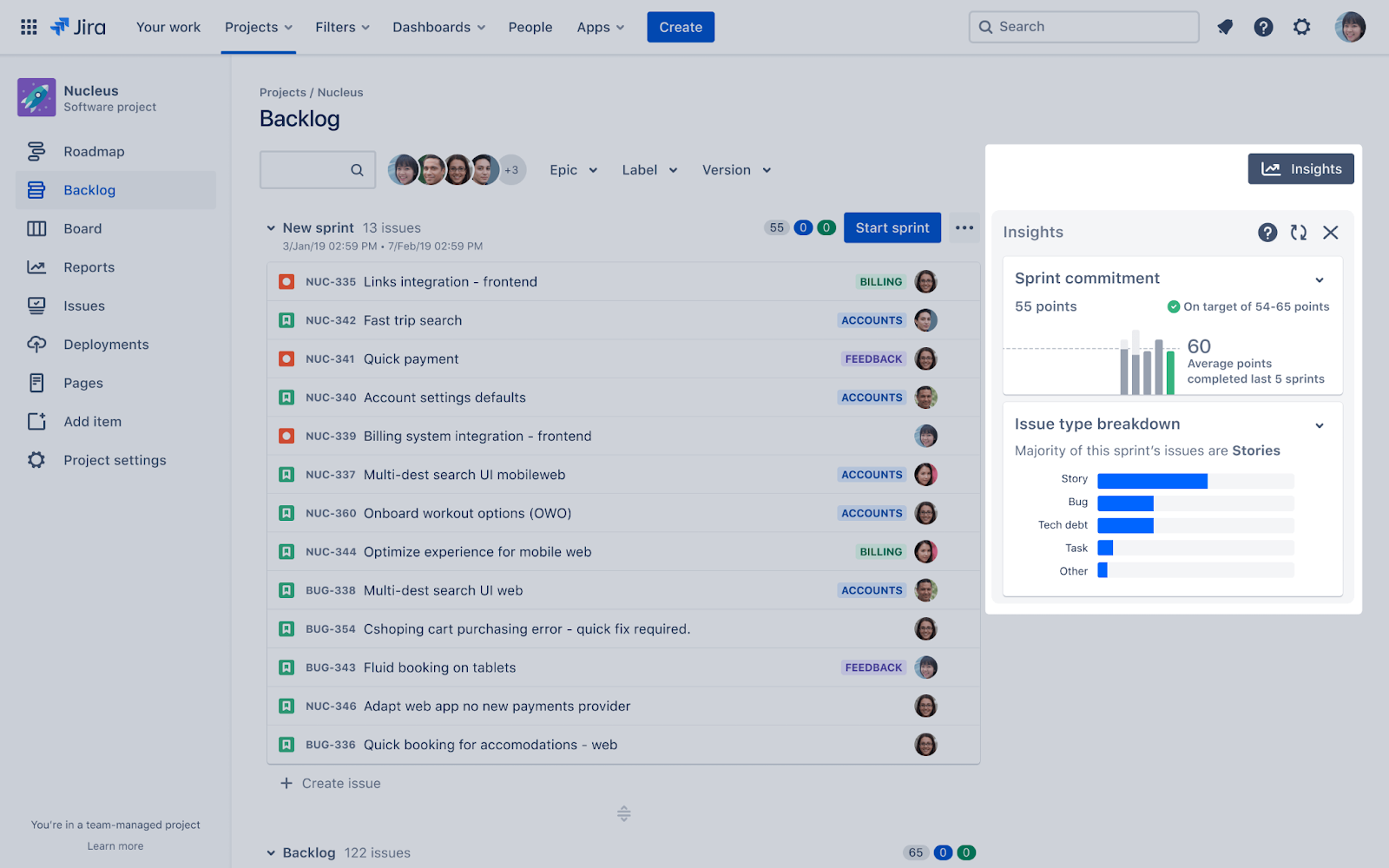
## Herramientas

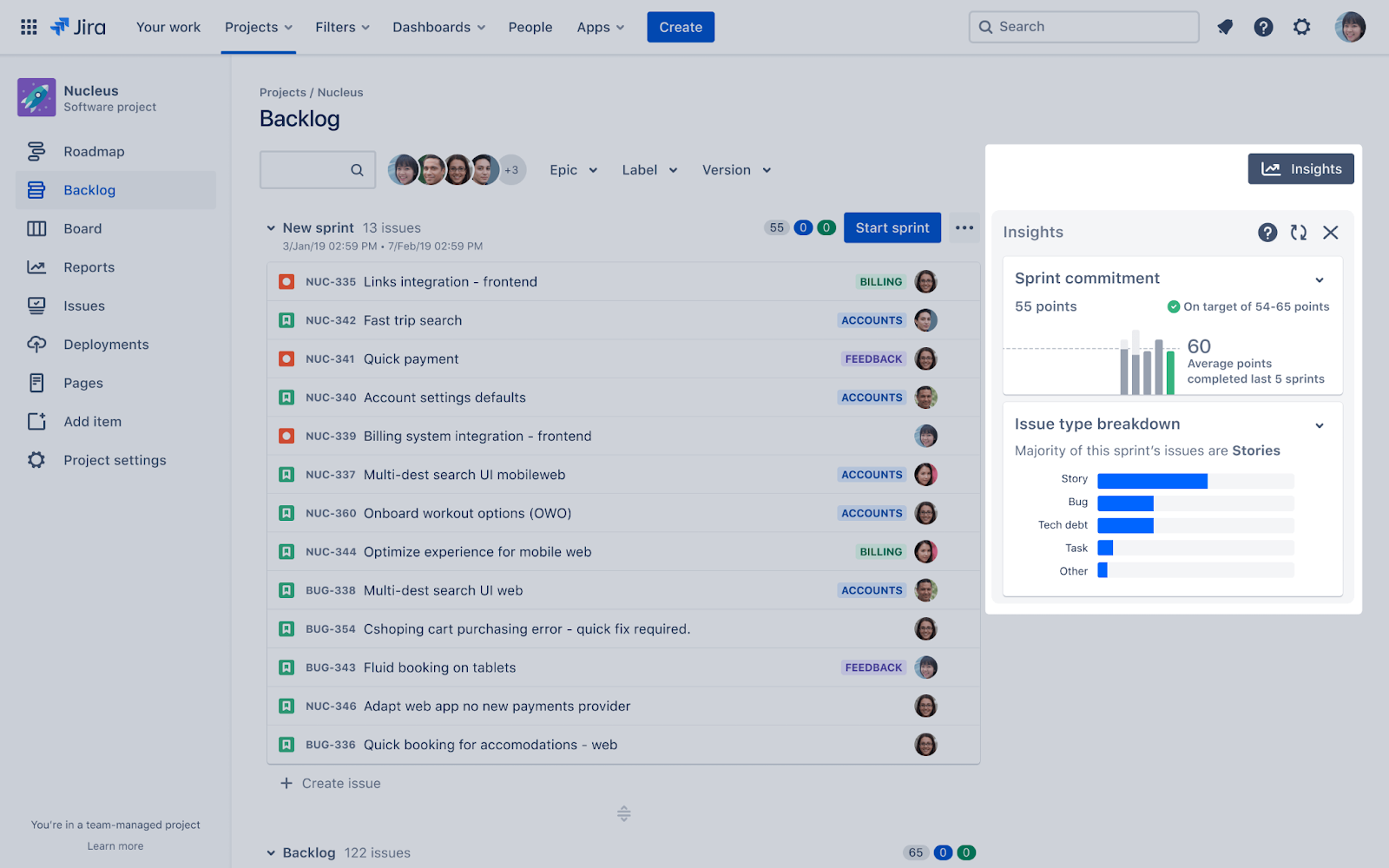
### JIRA

Es un software que ayuda al equipo a gestionar sus tareas y actividades.

Es una herramienta muy completa que permite la gestión y organización de proyectos, a fin de optimizar muchas de las tareas que la construcción de un producto implica. Su mayor atractivo radica en la adaptación que proporciona a proyectos ágiles, pero este potente software ofrece mucho más.

Puede adaptarse a empresas que no tienen nada que ver con el mundo del desarrollo de software, ya que su flujo de procesos también permite adecuarse a diferentes industrias y necesidades.

Podemos ver el backlog (debajo) y el sprint en curso (el que se llama “New Sprint” suele llamarse por el número de sprint o la fecha)

A la izquierda se elige si se quiere ver el BACKLOG (seleccionado) o el “BOARD” (para ver el tablero del Sprint)

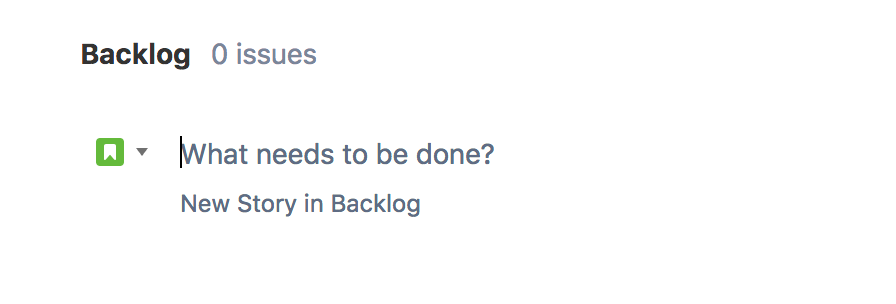
#### Elementos en JIRA

Podemos ver BUGS (el cuadro rojo con círculo blanco)

Se pueden definir user stories, tareas específicas -suelen ser un check azul- (como por ejemplo un reporte de cantidad de usuarios), epicas - violetas con un rayo blanco-

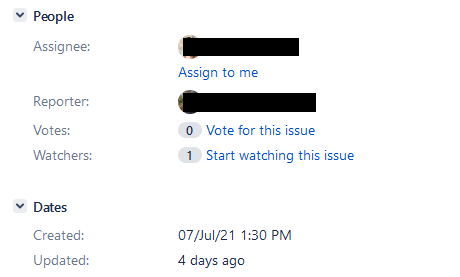
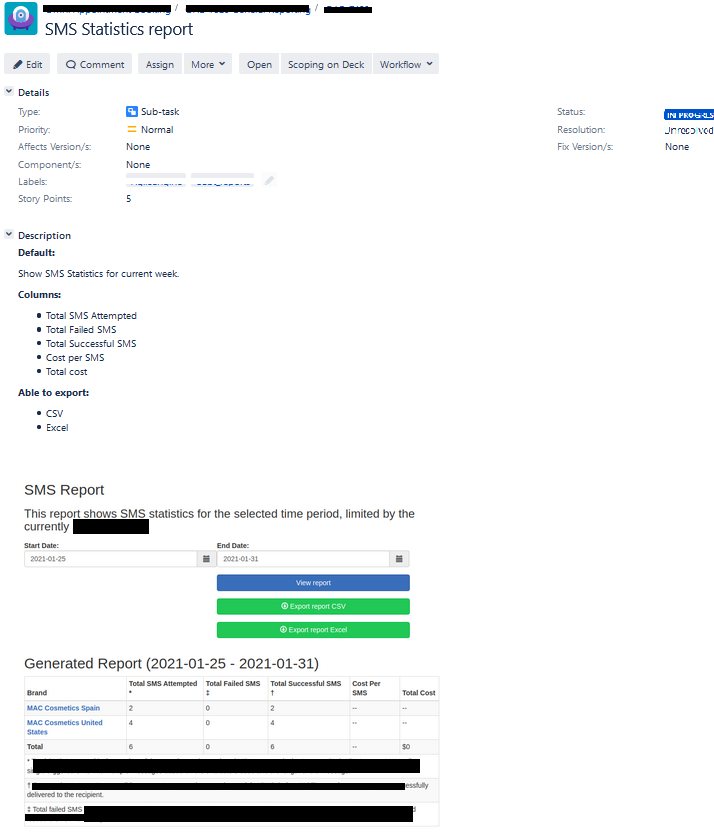
##### Ejemplo: tarea

Paso 1: Creación de User Story

Crear historias de usuario o tareas en el backlog

Paso 2: Llenar la descripción de la tarea.

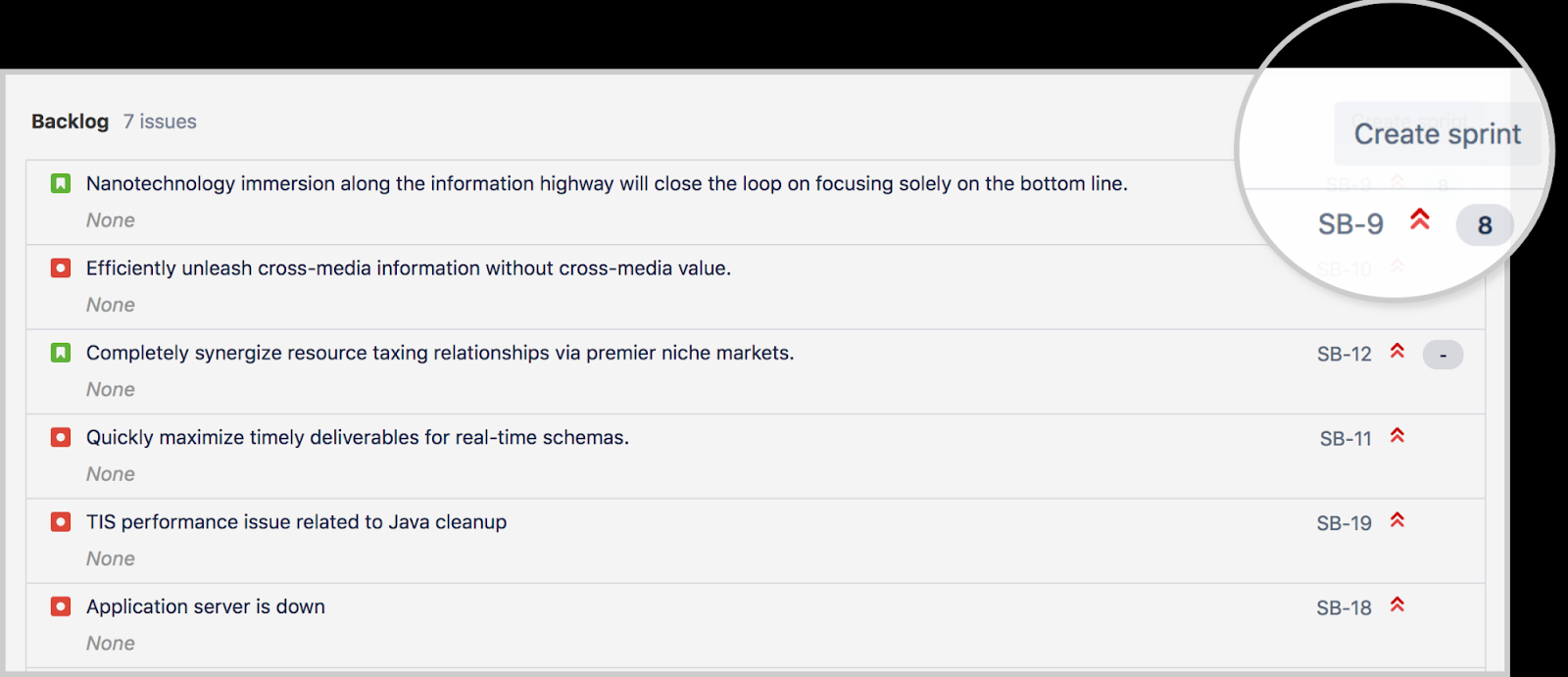
(Ejemplo Jira)



Esto se mencionara en la Planning y es clave para poder estimar! Si no se entiende algo, no se puede estimar.

Paso 3: Cuando ya están todas las tareas creadas, estimadas y asignadas.

Al finalizar la Planning vamos a comenzar el Sprint!



### Trello

Es una herramienta para gestión de tareas online fácil de usar, visual e intuitivo.

Permite organizar las diferentes tareas a realizar en un único espacio, un tablero.

Permite dejar de lado los post-its acumulados por tu mesa de trabajo, las listas de tareas pendientes y los recordatorios en mil lugares diferentes.

#### ¿Qué puedo hacer con Trello?

* Crear diferentes proyectos (tableros), independientes entre ellos.
* Crear proyectos públicos, privados o cooperativos.
* Crear columnas para cada grupo de tareas que debas realizar.
* Incluir archivos adjuntos en las tareas que así lo requieran.
* Realizar comentarios en algunas tareas, sin que eso influya en la visualización de tu tablero.
* Temporalizar tus tareas y visualizarlas en un calendario.
* Etiquetar las tareas por categorías.
* Trabajar en equipo en tiempo real.

## Glosario

**Prueba estática**: consiste en la revisión de productos de trabajo como documentos de requerimientos, casos de prueba, planes de prueba, código, guías de usuario etc.

**Prueba dinámica:** es la acción de probar

**Ciclo de vida:** es una secuencia estructurada y bien definida de las etapas en ingeniería de software para desarrollar el software deseado.

**Backend:** El backend es la parte del desarrollo web que se encarga de que toda la lógica de una página web funcione. Se trata del conjunto de acciones que pasan en una web pero que no vemos como, por ejemplo, la comunicación con el servidor

**Interfaz de usuario:** La interfaz de usuario (UI) es el conjunto de los controles y canales sensoriales mediante los cuales un usuario puede comunicarse con una máquina

**Jira:** es una herramienta de gestión de proyectos que sirve para gestionar tareas

#### ¿QUIERES SABER MÁS? TE DEJAMOS MATERIAL AMPLIADO DE LA CLASE

* [Tutorial de Jira para testers parte 1](https://www.youtube.com/watch?v=LeRxpTy-eNU&t=146s)
* [Tutorial de Jira para testers parte 2](https://www.youtube.com/watch?v=jfKiywGUxtA)
* [Tutorial de Trello 2021](https://www.youtube.com/watch?v=gGcLLDRVYcc&t=6s)

#DEMOCRATIZANDOLAEDUCACIÓN