Métodos/Técnicas de Ingeniería de Software
Evaluación 1 (2025 - 2)

1. Descripción del trabajo

Los alumnos, en forma **personal**, deben desarrollar y desplegar una aplicación web diseñada en base a una arquitectura por capas (monolítica).

2. Lineamientos generales

- La evaluación se realizará en forma "**personal"**.
- Para la evaluación no se debe entregar ningún informe escrito.
- Cada alumno debe presentarse en forma puntual en la fecha/hora programada. En caso contrario se le calificará con la nota mínima 1.0
- A la evaluación solamente deben presentarse aquellos alumnos que fueron planificados para la fecha. No se permitirá el ingreso de otros alumnos.

3. Acerca del proyecto de software

3.1 Contexto del problema

ToolRent, una tienda dedicada al arriendo de herramientas para construcción, reparaciones y proyectos domésticos, enfrenta crecientes dificultades en la gestión de sus operaciones debido al aumento en la demanda y a la diversidad de clientes que atiende. Actualmente, los procesos de solicitud, entrega y devolución de herramientas se realizan de manera manual mediante planillas y registros físicos, lo que genera múltiples inconvenientes: errores en la disponibilidad del inventario, retrasos en la atención de clientes, dificultad para calcular multas por devoluciones tardías y falta de control sobre el estado de las herramientas.

Esta situación ha ocasionado pérdidas económicas por extravío de equipos, tiempos muertos en la atención de clientes y desconfianza respecto a la transparencia del servicio. Asimismo, la ausencia de un sistema centralizado impide generar reportes confiables sobre las herramientas más utilizadas, la frecuencia de daños, los clientes con historial de incumplimientos y la necesidad de reponer o dar de baja equipos.

Para resolver estas limitaciones y optimizar tanto la experiencia de los clientes como la eficiencia operativa, **ToolRent** busca implementar un sistema integral de gestión de préstamos de herramientas. Dicho sistema permitirá automatizar el flujo de solicitudes y devoluciones, aplicar reglas de negocio (límites de préstamos, multas, etc.), controlar el inventario en tiempo real y generar reportes estratégicos que faciliten la toma de decisiones. Con esta solución, la empresa podrá asegurar una operación más eficiente, reducir pérdidas y brindar un servicio confiable y competitivo en el mercado.

3.2 Requisitos funcionales y Reglas de negocio

Épica 1: Gestión inventario de herramientas. Permite administrar el catálogo de herramientas disponibles en la tienda, asegurando un control adecuado de su registro, estado y disponibilidad. Incluye el alta de nuevas herramientas, actualización de stock y control de bajas, garantizando la integridad del inventario y la trazabilidad de cada activo.

Requisitos Funcionales

- **RF1.1** Registrar nuevas herramientas con datos básicos (nombre, categoría, estado inicial, valor de reposición).
- **RF1.2** Dar de baja herramientas dañadas o en desuso.

Reglas de Negocio

- Una herramienta solo puede registrarse si tiene nombre, categoría y valor de reposición.
- Estados válidos: Disponible, Prestada, En reparación, Dada de baja.
- El registro de nuevas herramientas genera un nuevo movimiento en el Kardex.

Épica 2: Gestión de Préstamos y Devoluciones. Es el núcleo del sistema, encargado de controlar todo el ciclo de vida de un préstamo: desde la entrega de herramientas a los clientes, hasta su devolución, cálculo de multas por atrasos y penalizaciones por daños. Incluye validaciones de disponibilidad, restricciones a clientes con deudas, y actualización automática de stock y estados.

Requisitos Funcionales

- **RF2.1** Registrar un préstamo asociando cliente y herramienta, con fecha de entrega y fecha pactada de devolución. Se actualiza el kardex.
- **RF2.2** Validar disponibilidad antes de autorizar el préstamo.
- **RF2.3** Registrar devolución de herramienta, actualizando estado y stock. Se actualiza el kardex.
- **RF2.4** Calcular automáticamente multas por atraso (tarifa diaria).
- **RF2.5** Bloquear nuevos préstamos a clientes con atrasos no regularizados.

Reglas de Negocio

- **Condiciones para realizar un préstamo**
  - El cliente debe estar en estado Activo (no restringido).
  - El cliente no debe tener:
    - Préstamos vencidos.
    - Multas impagas.
    - Deudas por reposición de herramientas.
  - La herramienta debe estar en estado Disponible y tener stock ≥ 1.
  - No se permite prestar más herramientas de las disponibles en stock.
  - El sistema debe verificar que la fecha de devolución no sea anterior a la fecha de entrega.

- **Reglas sobre plazos y devoluciones**
  - Cada préstamo debe tener una fecha pactada de devolución obligatoria.
  - Si la devolución se realiza antes de la fecha pactada, no hay devoluciones de dinero (tarifa mínima siempre es 1 día).
  - Si la devolución se realiza después de la fecha pactada, se debe calcular multa diaria por atraso.
  - Una herramienta devuelta dañada debe cambiar su estado a En reparación hasta que se evalúe el daño.
  - Si el daño es irreparable, la herramienta pasa a estado Dada de baja y se cobra al cliente el valor de reposición.

- **Reglas sobre multas y penalizaciones**
  - Las multas se calculan como:
    - Multa atraso = días de atraso × tarifa diaria de multa.
    - Multa por daño irreparable = valor de reposición de la herramienta.
  - Si la herramienta se devuelve con daños leves, se puede aplicar un cargo de reparación.
  - El sistema debe bloquear préstamos a clientes con multas pendientes hasta que las paguen.

- **Reglas sobre límites de préstamo**
  - Un cliente puede tener un número máximo de 5 préstamos activos simultáneamente.
  - Un cliente no puede tener más de una unidad de la misma herramienta en préstamo al mismo tiempo.
  - El sistema debe impedir que se acumulen préstamos sobre la misma herramienta sin devolución previa.

- **Reglas sobre la gestión del kardex en préstamos**
  - Todo préstamo genera un movimiento en el kardex con tipo Préstamo, reduciendo el stock.
  - Toda devolución genera un movimiento en el kardex con tipo Devolución, aumentando el stock.

- Si se aplica baja por daño, se genera movimiento tipo Baja, reduciendo stock permanentemente.

- **Restricciones adicionales**
  - Un préstamo no puede modificarse una vez registrado; cualquier cambio debe hacerse mediante devolución y nuevo préstamo.
  - Todas las operaciones de préstamo y devolución deben quedar registradas con fecha, hora y usuario responsable.

Épica 3: Gestión de kardex y movimientos. Lleva el registro detallado de todos los movimientos que afectan al inventario, tales como préstamos, devoluciones, reparaciones o bajas. Actúa como un historial de auditoría que asegura la trazabilidad de cada herramienta, permitiendo consultas y reportes de movimientos por fechas o por herramienta específica.

Requisitos Funcionales
- **RF3.1** Registrar automáticamente en el kardex cada movimiento (registro nuevas herramientas, préstamo, devolución, baja, reparación).
- **RF3.2** Consultar historial de movimientos de cada herramienta.
- **RF3.3** Generar listado de movimientos por rango de fechas.

Reglas de Negocio
- Todo cambio en el inventario debe registrarse automáticamente en el kardex.
- Un movimiento debe incluir: tipo (ingreso, préstamo, devolución, baja, reparación), fecha, usuario y cantidad afectada.
- El kardex debe permitir consultas por herramienta y por rango de fechas.

Épica 4: Reportes y consultas. Brinda a los usuarios la capacidad de generar reportes y consultas que permitan visualizar información clave, como préstamos activos, clientes con atrasos y las herramientas más solicitadas. Esta funcionalidad apoya la toma de decisiones y la supervisión del correcto funcionamiento de la tienda.

Requisitos Funcionales
- **RF4.1** Listar préstamos activos y su estado (vigentes, atrasados).
- **RF4.2** Listar clientes con atrasos.
- **RF4.3** Reporte de las herramientas más prestadas (Ranking).

Reglas de Negocio
- Los reportes deben poder filtrarse por rango de fechas.

4. Aspectos del desarrollo del producto

4.1 Respecto del Frontend

- Debe ser desarrollado usando ReactJS.
- Se requiere un único frontend para la aplicación.
- Se sugiere desarrollar usando *Visual Studio Code*.
- El código fuente debe ser escrito en inglés.

4.2 Respecto del Backend

- Debe ser desarrollado usando *Spring Boot* (dependencias a usar: Spring Web, MySQL/PostgreSQL Driver, Spring Data JPA, Lombok).
- Debe ser implementada en el lenguaje de programación Java usando programación orientada a objetos.
- Debe ser una aplicación web monolítica basada en al patrón arquitectural por capas (layers).
- Debe usar una base de datos relacional (Por ejemplo, *MySQL, PostgreSQL, etc.*).
- La aplicación debe ser desarrollada usando un IDE (por ejemplo, *IntelliJ, VS Code, etc.)*.
- El código fuente debe ser escrito en inglés.

4.3 Despliegue del producto en producción

- La aplicación web (frontend y backend) debe quedar desplegada y totalmente funcionando en un servidor de la nube (*AWS*, Azure, GCP, *Digital Ocean*, etc.).
- El despliegue de la aplicación en el servidor se debe realizar manualmente usando *Docker Compose*. Los componentes para desplegar son: Base de Datos (MySQL/PostgreSQL), Backend (3 réplicas), *Nginx* como balanceador de carga para el backend y el frontend. Todos estos componentes deben ser desplegados desde sus imágenes Docker respectivas (almacenadas en Docker Hub).
- La aplicación web debe poder ser accedida desde cualquier navegador web local.