**Proyecto final inventarios con sistema de notificaciones para productos**

**Tipos de usuarios:**

**Usuario Principal**

El Usuario se encarga de entrar al sistema, verificar las existencias de los productos y ver cuales están disponibles y cuales no, si hay productos faltantes, generara una orden que será enviada al proveedor.

**Proveedor**

el proveedor recibe la orden y en base a lo recibido, deberá checar los productos en existencia, de no haber algún producto, se le notificará de vuelta al usuario antes de que se realice un pedido para confirmar si está seguro.

**Requerimientos funcionales**

**Registro de productos:**

Este permite al usuario registrar los productos que entran al stock de la tienda, ya sea sumando los que ya están disponibles al stock o registrando nuevos productos que han sido agregados.

**Productos disponibles:**

Este requerimiento nos mostrará la cantidad de productos disponibles que tenemos, este permitirá al usuario no solo checar la cantidad de productos, si no también los que hacen falta para realizar un pedido al proveedor, el cual recibirá la lista de los productos faltantes o a punto de agotarse, cuando un producto Esta a punto de agotarse, se le indicará al usuario por medio de un numero en rojo. Los productos se muestran en cuatro secciones, cantidad, Código del producto, nombre y proveedor, esto para facilitar al usuario los pedidos hacia tal proveedor.

**Pedido:**

En este nos mostrará el día y la fecha del pedido a realizar, el cual se le enviará al proveedor por medio del software, esto para facilitar la recepción por parte del proveedor además de que por medio de este el proveedor al momento de recibir, podrá notificar al usuario.

**Notificación de Productos Disponibles:**

Con este módulo el proveedor podrá notificar al usuario de que productos se encuentran en stock y no le llegaran al usuario, por este medio el proveedor puede facilitar las transacciones y ahorrarle tiempo y dinero a ambas empresas.

**Requerimientos No funcionales**

**Registro de productos:**

Este permite al usuario registrar los productos que entran al stock de la tienda, ya sea sumando los que ya están disponibles al stock o registrando nuevos productos que han sido agregados.

**Productos disponibles:**

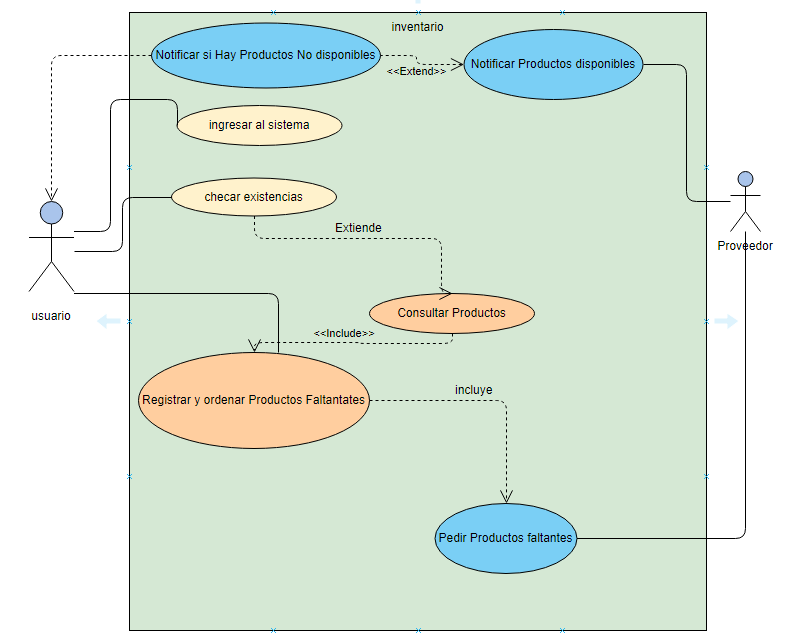
Productos Disponibles muestra los que están en stock en el inventario, sin embargo, deberá mostrar en rojo aquellos productos que estén agotados o a punto de agotarse y deberá haber contador mínimo para determinar cuales están a punto de agotarse, estos deberán ser definidos por el usuario mismo al momento de registrar un producto.

**Pedido:**

En cuanto un producto es enviado, deberá mostrar a que proveedor se ha enviado y que día fue enviado para tener un mejor control además de mostrar el monto de productos a pedir que usualmente es el mínimo para un proveedor.

**Notificación de Productos Disponibles:**

Esta notificación limita a los productos que la empresa tenga en stock que tengan en común con el cliente, por lo tanto, deberá enviársele con tiempo y forma al cliente, de esta forma se espera que sea ahora el cliente quien confirme de vuelta si requiere o no el pedido, para que se realice una transacción o se evite para que el cliente pueda comparar stock con otro proveedor.



**Definición de estándar de codificación de java**

**Nomenclatura**

El idioma por defecto a la hora de dar sentido funcional al nombre de clases, variables, constantes, etc. será una mezcla entre la nomenclatura tradicional en inglés y la nomenclatura funcional adoptada.

Resumiendo, aquella codificación que por estandarización y/o aceptación se pueda escribir en inglés se mantendrá así por convenio, casos como insert, update, delete, create, retrieve, list, set, get, newInstance, Delegate.

Para la parte funcional se utilizara castellano, por lo tanto la nomenclatura de los métodos será: getListEmpresa en sustitución de getListCompany o insertBanco en lugar de insertarBanco.

**Nombres de Interfaces**

Los nombres de interfaces utilizarán el sufijo Interface y estarán compuestos por palabras con la primera letra en mayúscula (CamelCase). Se debe evitar el uso de abreviaciones que dificulten la comprensión del código.

Ejemplo: ConexionInterface, ComponenteTablaInterface

**Nombres específicos de gestiones**

Cuando se trata de gestionar una entidad determinada (Ej. Usuario) se definen los nombres de clases, demás ficheros implicados con la siguientes reglas:

Clase: <<FuncionalidadGenerica>><<Entidad>><<Especificación de Clase>>

Ejemplo: Usuario

UsuarioAction, FindUsuarioAction

**Métodos**

Los métodos deberán ser verbos (en infinitivo), en mayúsculas y minúsculas con la primera letra del nombre en minúsculas, y con la primera letra de cada palabra interna en mayúsculas (lowerCamelCase).

No se permiten caracteres especiales.

El nombre ha de ser lo suficientemente descriptivo, no importando a priori la longitud del mismo.

**Variables**

Los nombres de las variables tanto de instancia como estáticas reciben el mismo tratamiento que para los métodos, con la salvedad de que aquí sí importa más la relación entre la regla mnemónica y la longitud del nombre.

Ejemplo:

Correctos: diaCalculo, fechaIncoporacion

Se evitará en la medida de lo posible la utilización de caracteres especiales, así como nombre sin ningún tipo de significado funcional.

Las excepciones son las variables utilizadas en bucles for, para esos casos se permite utilizar i, j, k, l y siempre en ese orden de anidamiento.

El primer bucle siempre será el que tenga la variable i como iterador. (Esta variable se definirá para el bucle en cuestión).

**Constantes**

Los nombres de constantes de clases deberían escribirse todo en mayúsculas con las palabras separadas por subrayados ("\_"). Todas serán declaradas como public static final

public static final String PROPERTY\_URL\_SERVICIO = "urlServicio";

**Comentarios**

Los comentarios serán utilizados para dar información adicional al desarrollador sobre la implementación del diseño de la clase. Se tiene, por tanto, que evitar referencias al diseño funcional de la misma.

El uso abusivo de los comentarios es desaconsejable, principalmente por el trabajo extra necesario para su correcto mantenimiento. Es preferible rediseñar el código para una mejor compresión del mismo.

**Herramientas a utilizar**

**Eclipse**

Es un proyecto de código abierto y el IDE está disponible para múltiples sistemas operativos. Eclipse es un IDE no solo para Java, sino para muchos otros lenguajes y herramientas de desarrollo. Es considerado por muchos el IDE por excelencia, al incorporar un gran abanico de complementos que facilitan prácticamente todas las tareas relativas al desarrollo de software.

**Google hangout**

Este nos sirve para realizar pruebas de software además de pequeñas reuniones donde checaremos avances en el proyecto además de pequeñas pruebas y determinación de requerimientos faltantes al momento de codificar.

**Github**

Nos permite subir modificaciones al repositorio y versiones, también documentar los cambios hechos al sistema y poder ver quien realizo cambios al momento de subirlo.

**Diseño.**

En esta fase se utiliza toda la información recogida en la fase de especificación y se propone una solución que responda a las necesidades del usuario y se pueda desarrollar. En esta fase se define la arquitectura de la aplicación. Es necesario detallar la estructura y la organización del programa y cómo se relacionan las distintas partes de la aplicación.

El diseño conlleva:

​​- Decidir la estructura de la interfaz

- Qué componentes gráficos se van a utilizar, y cómo se van a relacionar estos componentes

-​​Decidir la disposición (layout) de los componentes

- sirven para organizar los componentes contenidos en los mismos.

​​- Decidir el comportamiento de la interfaz: gestión de eventos

- Algunos componentes son controles: permiten reaccionar ante eventos del usuario. El comportamiento se especifica programando las respuestas a dichos eventos.

**Codificación.**

Esta fase consiste en la programación en Java de las especificaciones de diseño de la fase anterior. Durante esta fase de codificación o implementación se aplican normas de programación para facilitar la legibilidad de los programas Java.

**Prueba.**

En esta fase se compila y se ejecuta la aplicación para verificar que cumple con los requisitos funcionales y técnicos definidos en la fase de especificación. Si el programa no cumple con todos los requisitos, se puede deber a errores de diseño o de programación. En tal caso, es necesario corregir los errores que se hayan detectado y repetir de nuevo el proceso de diseño y codificación. Durante la fase de prueba se verifica que la aplicación cumple con los criterios de calidad establecidos en el proyecto: facilidad de uso, corrección, eficiencia, integridad, fiabilidad, flexibilidad, facilidad de mantenimiento, facilidad de prueba, portabilidad, capacidad de reutilización e interoperabilidad.

Una vez que la aplicación se ha probado y cumple con los requisitos establecidos, entonces se pone en funcionamiento y comienza la fase de operación para que sea utilizada para el fin con el que ha sido desarrollada.

**Mantenimiento.**

Una vez que la aplicación se ha puesto en operación da comienzo la fase de mantenimiento. En esta fase se corrigen errores de funcionamiento de la aplicación, se modifica la funcionalidad o se añaden las nuevas funcionalidades que demandan los usuarios. La fase de mantenimiento es la de mayor duración, pues pueden pasar muchos años desde el inicio de la operación hasta que el producto es retirado.