**Texto

Descripción generada automáticamente**

**UNIVERSIDAD SEÑOR DE SIPAN**

ESCUELA PROFESIONAL DE INGENIERIA DE SISTEMAS

**Título:** DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA FRONTEND Y BACKEND PARA EL ÁREA DE LOGÍSTICA EN LA EMPRESA LAIVE

**AUTOR(ES):**

HERNA LOJA, BRYAM ALEXANDER

CHUQUE DÍAZ, ELBIR DILUVID

VIVANCO MORENO, MANUEL SANTIAGO

**DOCENTE:**

ATALAYA URRUTIA, CARLOS

LIMA - PERU

2021

**Contenido**

[**INTRODUCCIÓN** 4](#_Toc84673461)

[**I.** **ANÁLISIS EXPLORATORIO EMPRESARIAL** 5](#_Toc84673462)

[**1.1.** **Diagnóstico de la Empresa** 5](#_Toc84673463)

[1.1.1. Descripción de la empresa 5](#_Toc84673464)

[1.1.2. Estructura organizacional de la empresa 6](#_Toc84673465)

[1.1.3. Direccionamiento Estratégico Empresarial 8](#_Toc84673466)

[**II.** **ALCANCE DEL PRODUCTO / SOFTWARE** 11](#_Toc84673467)

[**III.** **GESTIÓN DE PROCESOS DE NEGOCIO (BPM):** 11](#_Toc84673468)

[**3.1.** **Procesos de negocio identificados** 11](#_Toc84673469)

[3.1.1. Proceso de Ingreso de Materiales al almacén de insumos 11](#_Toc84673470)

[3.1.2. Proceso de Salida por requerimiento de Producción 14](#_Toc84673471)

[Diagrama de contexto del sistema: 16](#_Toc84673472)

[**3.2.** **Modelamiento de procesos de negocio:** 18](#_Toc84673473)

[**3.3.** **Modelado de Requerimientos** 21](#_Toc84673474)

[Diagramas de Caso de Uso 21](#_Toc84673475)

[Diagramas de Actividades 23](#_Toc84673476)

[Descripción de Perfiles 24](#_Toc84673477)

[**IV.** **IMPLEMENTACIÓN DEL DISEÑO FÍSICA DE LA BASE DE DATOS.** 25](#_Toc84673478)

[**V.** **MOCKUPS DEL SISTEMA DE CONTROL DE ALMACÉN.** 25](#_Toc84673479)

**Diagrama**

[Diagrama 1:Flujo general de Compras Locales e Importados de MP/Insumos. 13](file:///D:\02%20-%20UNIVERSIDAD%20SEÑOR%20DE%20SIPAN\007.-%20CLASES%202021-2\1.%20TALLER%20DE%20LENGUAJES%20DE%20PROGRAMACIÓN\PA1\PA1%20-%20DAD%20VF.docx#_Toc84674211)

[Diagrama 2: Flujo de Salida de MP/Insumos para la producción 15](#_Toc84674212)

[Diagrama 3:Diagrama de contexto 16](#_Toc84674213)

[Diagrama 4: Diagrama de contexto 16](#_Toc84674214)

[Diagrama 5:Diagrama de contexto 17](#_Toc84674215)

[Diagrama 6:Diagrama de contexto 17](#_Toc84674216)

[Diagrama 7:Diagrama de contexto 18](#_Toc84674217)

[Diagrama 8:Diagramas de Caso de Uso 22](#_Toc84674218)

[Diagrama 9:Diagramas de Actividades 23](#_Toc84674219)

[Diagrama 10: Base de datos 25](#_Toc84674220)

**Tabla**

[Tabla 1:Proceso general de Ingreso de Materiales al almacén de insumos 12](#_Toc84674221)

[Tabla 2:Proceso general de Salida por requerimiento de Producción 14](#_Toc84674222)

[Tabla 3:Requisitos funcionales 19](#_Toc84674223)

[Tabla 4:Requerimientos no funcionales 21](#_Toc84674224)

[Tabla 5:Descripción de Perfiles 24](#_Toc84674225)

[Tabla 6:Descripción de Perfiles 24](#_Toc84674226)

**Ilustraciones**

[Ilustración 1:Login 28](#_Toc85288782)

[Ilustración 2:Ventana de alerta 28](#_Toc85288783)

[Ilustración 3:Detalle de alerta 29](#_Toc85288784)

[Ilustración 4: Formulario Principal 29](#_Toc85288785)

[Ilustración 5:Ajustes de Ingresos 30](#_Toc85288786)

[Ilustración 6: Ajustes de salida 30](#_Toc85288787)

[Ilustración 7: Traspasos 31](#_Toc85288788)

[Ilustración 8: Ingresos 31](#_Toc85288789)

[Ilustración 9:Salida 32](#_Toc85288790)

[Ilustración 10:Kardex 32](#_Toc85288791)

[Ilustración 11: Resumen 33](#_Toc85288792)

[Ilustración 12: Login Web 33](#_Toc85288793)

[Ilustración 13: Menú Web 34](#_Toc85288794)

[Ilustración 14: Kardex Web 34](#_Toc85288795)

[Ilustración 15: Resumen Web 35](#_Toc85288796)

[Ilustración 16: Capas del proyecto 35](#_Toc85288797)

[Ilustración 17:Formularios realizados en Visual Studio 36](#_Toc85288798)

[Ilustración 18: Formulario Login 36](#_Toc85288799)

[Ilustración 19: Mensaje de Alerta 36](#_Toc85288800)

[Ilustración 20: Formulario de Advertencia de productos por vencer y bajo stock 37](#_Toc85288801)

[Ilustración 21: Formulario principal con el menú de opciones 37](#_Toc85288802)

[Ilustración 22:Formulario de ajuste de ingreso (Administración) 38](#_Toc85288803)

[Ilustración 23: Formulario de ajuste de Salida(Administración) 38](#_Toc85288804)

[Ilustración 24: Formulario traspaso (Administración) 39](#_Toc85288805)

[Ilustración 25: Formulario de Ingreso de productos (Inventario) 39](#_Toc85288806)

[Ilustración 26: Formulario de Salida de productos (Inventario) 40](#_Toc85288807)

[Ilustración 27: Formulario de Kardex (Reportes) 40](#_Toc85288808)

[Ilustración 28: Formulario de Resumen (Reportes) 41](#_Toc85288809)

# **INTRODUCCIÓN**

El presente proyecto busca generar una solución que ayude a agilizar los procesos logísticos de la Empresa “LAIVE”. El objetivo es analizar los procesos de la empresa para definir el sistema de software que satisfaga las necesidades de negocio de clientes y usuarios.

Para analizar las diferentes falencias de la empresa se obtuvo información objetiva con el fin de poder analizar las debilidades de los procesos y así poder enfocarnos en brindar una solución sobre el proceso que genere mayor beneficio para el mismo.

En el desarrollo del sistema se analizará los procesos, los procedimientos y las áreas que comprenden el flujo empresarial y se desarrollará un software para mejorar, controlar y monitorear los procesos de la empresa LAIVE

1. **ANÁLISIS EXPLORATORIO EMPRESARIAL**

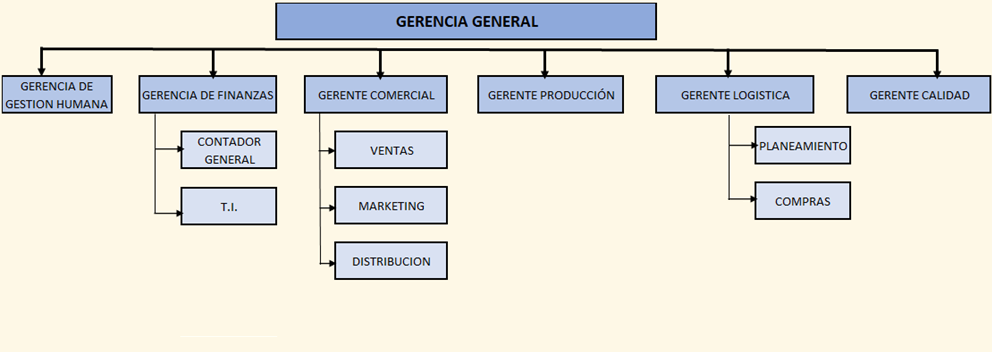
* 1. **Diagnóstico de la Empresa**
     1. Descripción de la empresa

Laive S.A. es una empresa peruana con 116 años desde su fundación. La empresa ha pasado por diversas reestructuraciones productivas: en sus inicios se dedicaba a la crianza de ganado vacuno; en los años de la reforma agraria, hizo un giro hacia la producción de lácteos, una década después incursionó también en el segmento de productos cárnicos. En adelante ha construido diversas plantas de productos ultra-pasteurizados y plantas de evaporación de leche. Sus plantas se encuentran ubicadas en Lima y Arequipa.

Laive también contribuye a lo sociedad peruana, preocupándose por el medioambiente realizando reciclaje de productos, dando alimentación a escolares de zonas alejadas y construyendo viviendas a personas de bajo nivel socioeconómico.

En la actualidad, Laive posee una alta variedad de productos, marcas y presentaciones, con las que puede hacer frente a la demanda de los clientes. Asimismo, Laive apuesta por la investigación y desarrollo, lo cual le permite insertar a los mercados novedosos productos en calidad y presentación, incluso antes que sus competidores. Aprovecha la cobertura que posee a nivel nacional para abastecer la demanda interna (incremento de consumo per cápita a nivel nacional) y externa.

* + 1. **Estructura organizacional de la empresa**
       1. **Estructura Orgánica**

****

* + - 1. **Descripción de las áreas funcionales de la Empresa**

**Gerencia General:**

Esta área se encarga de dirigir y administrar, organizar el funcionamiento de LAIVE S.A. de acuerdo con sus fines y políticas establecidos, y ejercer la representación legal, comercial y administrativa.

**Gerencia de Gestión Humana:**

El área de gestión humana representa tareas de capacitación, administración, evaluación, gestión y análisis de personal, así como también de diversas situaciones para poder medir la eficiencia de los grupos de trabajo, así como reconocer áreas que pueden mejorarse y prevenir escenarios adversos para la organización.

**Gerencia de Finanzas:**

Esta área tiene la función de realizar las cancelaciones de las obligaciones económicas de la empresa. Como así también la de gestionar las partidas de ingresos y gastos que pueda tener la compañía.

Esta área a su vez está a cargo de las siguientes áreas:

* + Área de Contabilidad
  + Área de Soporte y TI

**Gerencia Comercial:**

Más allá de vender y establecer una conexión con el cliente a nivel nacional y algunos clientes extranjeros, una de las principales funciones del departamento comercial es garantizar la satisfacción del consumidor.

Esta área a su vez está a cargo de las siguientes áreas:

* + Área de Ventas
  + Área de Marketing
  + Área de Distribución

**Gerencia de Producción:**

El área de producción es el encargado de la creación y manufacturación de los bienes y servicios finales.

**Gerencia de Logística:**

Es aquella que se encarga de la compra de insumos, así como la planificación de la producción en base al requerimiento del área comercial asegurando la cadena de suministros.

Esta área a su vez está a cargo de las siguientes áreas:

* + Área de planeamiento
  + Área de compras

**Gerencia de Calidad:**

Esta gerencia se encarga de asegurar la inocuidad y calidad de los productos elaborados en la empresa, evitando que se comercialice un producto no apto para el cliente, así como también se encarga de hacer cumplir las normas de seguridad

* + 1. **Direccionamiento Estratégico Empresarial**
       1. **Misión**

Somos una empresa innovadora, eficiente y socialmente responsable, integrada por personas comprometidas que producen y comercializan alimentos de calidad garantizada a través de marcas confiables, contribuyendo a una alimentación saludable de sus consumidores.

* + - 1. **Visión**

Ser una empresa líder en innovación de productos alimenticios saludables

* + - 1. **Valores**

• Respeto: Tratamos a quienes nos rodean como nos gustaría ser tratados y tomamos siempre en cuenta las consecuencias que nuestras acciones puedan generar.

• Excelencia: Tenemos ganas de aprender y mejorar. No nos conformamos y buscamos siempre superarnos.

• Trabajo en Equipo: Cooperamos y trabajamos unidos por un mismo objetivo, interactuando entre áreas para maximizar el resultado.

• Actitud Ganadora: Nos ponemos la valla alta y nos retamos a superarla con valentía y pasión.

* + - 1. **Objetivos:**

Crear productos innovadores para llevar bienestar a las personas, fidelizar a todos los miembros de la familia, ofreciendo un producto diferenciado en calidad a comparación del que ofrecen sus principales consumidores.

* + - 1. **Estrategia:**
* Desarrollar campañas promocionales informativas para conocimiento del producto.
* Capacitar a nuestras promotoras de ventas en los beneficios naturales del producto.
* Publicitar a través de los BTL y ATL el nuevo producto.
* Desarrollar campañas de información a nuestros potenciales compradores y resaltar las diferencias con la competencia.
* Establecer un sistema de geomarketing (zonificación) para actualizar a los nuevos clientes de bodegas, mercados, etc.
* Contratar personal capacitado para la supervisión por zonas para el buen desarrollo del geomarketing (competitiva- estrategia de defensa de lugar).
* Crear un sistema de evaluación de precios en puntos de ventas. (Competitiva - defensa preventiva)
* Planificar promociones agresivas con respecto al precio. (Competitiva - defensa de su lugar).
  + - 1. **Marco Teórico**

La empresa Laive tiene más de 100 años de su creación, fue creada en el año 1910, en las alturas de Junín y Huancavelica tras la unión de tres haciendas que dieron paso a la Sociedad Ganadera del Centro; Laive se fundó en 1910 con el nombre de Sociedad Ganadera del Centro S.A. con la participación de Domingo Olavegoya, Felipe Chávez y Juan Valladares con 50.000 Libras Peruanas, para la compra de las haciendas Acopalca, Chamiseria y Runatullo y posteriormente Huari, Acocra, Punto, Laive e Ingahuasi, ellos comenzaron con la producción de mantequilla que hasta hoy goza de nuestra preferencia por su calidad y sabor original ellos fueron los pioneros en la producción de queso de maduración en 1930, en 1938 inicio su producción de queso fundido, luego en 1972 se establece en Lima con una moderna planta de fabricación de queso, mantequilla y posteriormente yogurt, en 1980 llego a establecerse en nuestra Arequipa. El propósito de la sociedad fue lograr la excelencia en la crianza de ganado lanar y vacuno, aprovechando para ello tierras situadas en las alturas de la Cordillera de los Andes, tierras marginales de agreste puna. Con técnicas avanzadas de manejo de suelos, mejoramiento y selección genética, cuidado del medio ambiente y estímulo al trabajador mediante servicios y condiciones de vida ejemplares, se lograron resultados técnicos que marcaron rumbos en la ganadería nacional y resultados económicos que permitieron crecer y superar los vaivenes de la economía nacional y mundial. Dentro de la Competencia para mantener este liderazgo en el mercado de jugos y néctares, las principales empresas del sector están extendiendo su línea de productos, o bien aumentando su inversión para consolidar su presencia. Hoy la empresa es la segunda más grande en la industria de productos lácteos, pero su oferta se

ha extendido a otras líneas como jugos y embutidos. “Pocas empresas cumplen 100 años”, dice Luis Ferrand, ex gerente general de Laive. “La clave ha sido la adaptación al cambio, ya sea de giro de negocio de productos o de mercado”.

Palacios afirma que Laive se ha mantenido cien años gracias a que la empresa ha llevado un manejo financiero conservador y a que los dueños dieron un paso al costado para profesionalizar la gerencia. Hoy el 43% de la empresa está en manos de la familia Palacios (a través del holding Valor Agro). Un porcentaje similar pertenece a la empresa chilena Santa Carolina, que ingresó a Laive en 1995 a través de un aumento de capital. El resto está en manos de terceros, ya que cotiza en la Bolsa de Valores de Lima.

1. **ALCANCE DEL PRODUCTO / SOFTWARE**

**Propósito**:

Almacenar el registro entrada y salida de productos.

**Beneficios:**

* Mejora en la reducción del tiempo de registro manual de entrada y salida de productos.
* Avisos de cambios en el sistema
* Reducción de costos de economato.
* Rentabilidad a largo plazo.
* Mejora en el control y gestión de inventarios de productos de consumo (Lacteos laive, jugos, embutidos, etc).
* Seguridad de datos de los usuarios y productos.
* Reduce la perdida de datos.
* Reducción del tiempo de despacho de productos.
* Incrementos de clientes nuevos a corto plazo.

1. **GESTIÓN DE PROCESOS DE NEGOCIO (BPM):** 
   1. **Procesos de negocio identificados**
      1. **Proceso de Ingreso de Materiales al almacén de insumos**

|  |  |
| --- | --- |
| **Nombre de Proceso** | Proceso de Ingreso de Materiales al almacén de insumos. |
| **Descripción** | Este procedimiento aplica a los productos comprados como insumos, envases, embalajes e insumos químicos, desde la recepción física y el ingreso al sistema SAP hasta su despacho interno al área usuaria. |
| **Propietario** | Superintendente de Almacén |
| **Tareas** | 1. El jefe de planificación elabora el plan de entregas logísticas (compras) y el plan de capacidades. 2. Subgerente del área de compras revisa el plan de compras 3. Genera órdenes de compra para las solicitudes a los proveedores. 4. Realiza el seguimiento de las órdenes de compra. 5. El supervisor de Almacén realiza la recepción e ingreso de los nuevos insumos 6. El jede de calidad analiza la muestra del insumo ingresado. 7. El jefe de calidad libera el producto de estar conforme si no lo rechaza de estar no conforme. 8. El subgerente realiza el reclamo al proveedor. 9. El supervisor de almacén realiza la salida del insumo del almacén para su despacho o baja |
| **Áreas involucradas** | Planificación, Compras, Almacén, Calidad |
| **Recursos** | Excel, Papel Continuo, Power BI, e-mail |
| **Documentos necesarios** | Plan de entregas logísticas, plan de capacidades, Orden de compra, Orden de producción, Factura |

Tabla 1:Proceso general de Ingreso de Materiales al almacén de insumos

**Imagen que contiene interior, computadora, computer, tabla

Descripción generada automáticamenteFlujo de Compras Locales e Importados de MP/Insumos.**

Diagrama :Flujo general de Compras Locales e Importados de MP/Insumos.

* + 2. **Proceso de Salida por requerimiento de Producción**

|  |  |
| --- | --- |
| **Nombre de Proceso** | Proceso de Salida por requerimiento de Producción |
| **Descripción** | Este procedimiento aplica a los productos comprados como insumos, envases, embalajes e insumos químicos, desde solicitud por parte de la planta de producción hasta la salida del almacén y entrega a planta. |
| **Propietario** | Superintendente de Almacén |
| **Tareas** | 1. El jefe de planta genera una reserva de productos en el sistema. 2. El supervisor de almacén revisa el documento de reserva generada. 3. Valida si se cuenta con stock disponible para la reserva. 4. Si no se cuenta con stock parcial o total, informa al subgerente de compras para que se genere la necesidad y con ella la orden de compra. 5. Si se cuenta con stock se procede a enviar a preparar lo solicitado. 6. El encargado de almacén prepara los materiales solicitados. 7. Revisa lo preparado para el envió a planta. 8. El supervisor de almacén realiza la salida en el Kardex. 9. Se entrega los materiales a planta para su consumo. |
| **Áreas involucradas** | Planta de producción, Almacén, Compras |
| **Recursos** | Papel Bond, E-mail |
| **Documentos necesarios** | Orden de reserva, Hoja de picking. |

Tabla 2:Proceso general de Salida por requerimiento de Producción

**Flujo de Salida de MP/Insumos para la producción**

Diagrama

Descripción generada automáticamente

Diagrama 2: Flujo de Salida de MP/Insumos para la producción

* + 1. **Proceso de Toma de Inventario Físico**

|  |  |
| --- | --- |
| **Nombre de Proceso** | Proceso de Toma de Inventario Físico |
| **Descripción** | Este procedimiento consiste en el conteo físico de cada existencia en el almacén en base al material el cual permite corregir las diferencias existentes sea positivas o negativas. |
| **Propietario** | Superintendente de Almacén |
| **Tareas** | 1. El Superintendente de Almacén solicita la paralización del requerimiento de ingreso y salida para inicial el inventario. 2. El superintendente de Almacén genera el reporte de Kardex. 3. Este reporte es entregado a Contabilidad para el acompañamiento en el conteo. 4. El supervisor de almacén inicia el conteo de las existencias en orden por código de material. 5. Estas se validan en el reporte de Kardex, de haber diferencias se apuntan en el documento. 6. Contabilidad Confirma valida las diferencias encontradas. 7. Contabilidad indica el motivo del ajuste 8. Se realiza las modificaciones en el sistema realizando los procesos de ajustes o de traspasos. |
| **Áreas involucradas** | Contabilidad, Almacén. |
| **Recursos** | Papel Bond, E-mail |
| **Documentos necesarios** | Kardex. |

Tabla 3: Proceso de Toma de Inventario Físico

**Flujo de Proceso de Toma de Inventario Físico**

Diagrama

Descripción generada automáticamente

### **Diagrama de contexto del sistema:**

* 1. **diagrama de contexto 0**

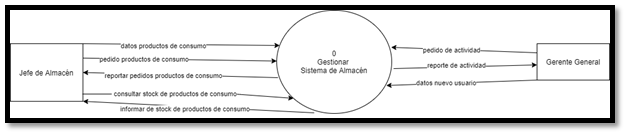


Diagrama 3:Diagrama de contexto

* 1. **diagrama de contexto 1**

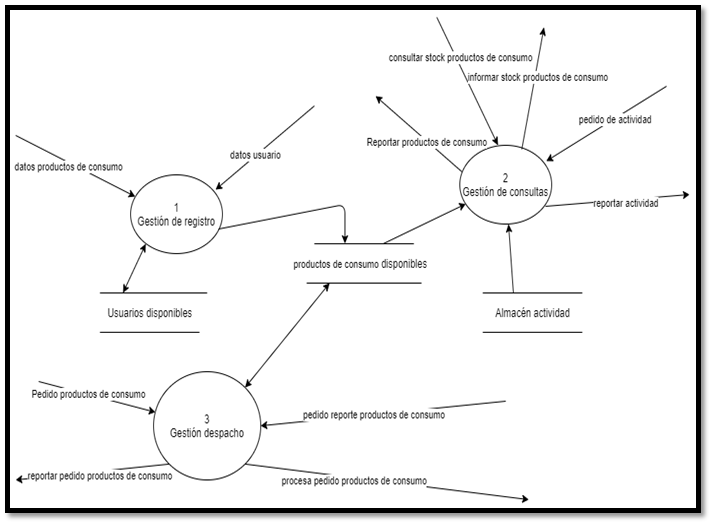


Diagrama 4: Diagrama de contexto

* 1. **diagrama de contexto 1.1**

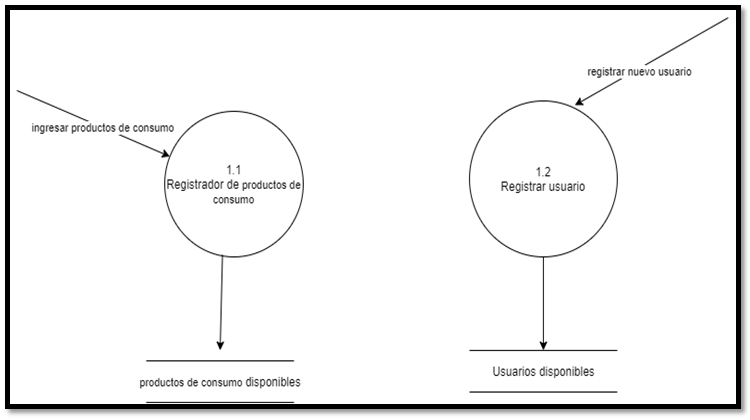


Diagrama 5:Diagrama de contexto

* 1. **Diagrama de contexto 2**

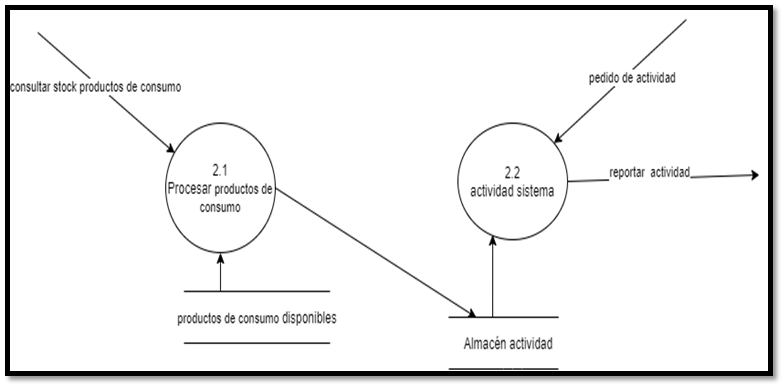


Diagrama 6:Diagrama de contexto

* 1. **Diagrama de contexto 3**

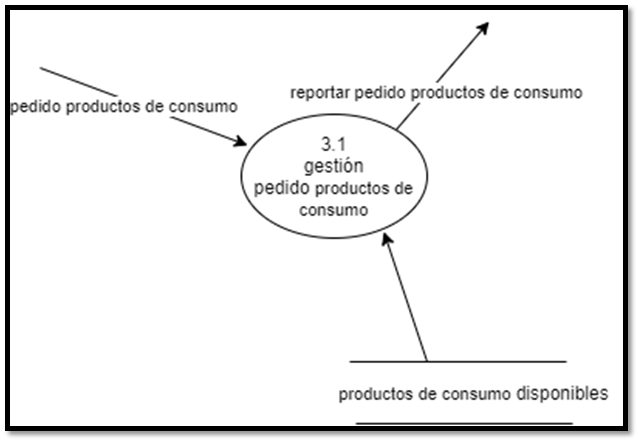


Diagrama 7:Diagrama de contexto

* 1. **Modelamiento de procesos de negocio:**

En esta parte se describe del alcance del software que se está especificando, incluyendo:

* 1. **Requisitos funcionales**

|  |  |  |  |
| --- | --- | --- | --- |
| **N°** | **Requisito funcional** | **Detalle** | **Regla de negocio** |
| **01** | Registrar los productos que salen. | Se debe registrar:   * Cantidad * Nombres lacteos laive, jugos, embutidos, etc. | Registro de salidas (Lacteos laive, jugos, embutidos, etc) |
| **02** | Registrar lacteos, jugos, embutidos, etc ingresados. | Se debe registrar:   * Cantidad * Nombres de lacteos, jugos, embutidos, etc * Precio * Nombre de proveedor * Fecha de entrada | Se deben registran los productos ingresados y el proveedor. |
| **03** | Enviar una notificación electrónica al área de gestión financiera; si existiese una desconformidad con el pedido. | Se incluye:   * Motivo de desconformidad * Nombre de producto * Firma electrónica de remitente | El jefe de almacén debe reportar los productos que incumplen las condiciones de calidad. |
| **04** | Clasificar los lacteos, jugos, embutidos, etc. | Lista ordenada alfabéticamente y agrupada por categoría de los lacteos, jugos, embutidos, etc | El almacén debe tener los productos ordenados de A hasta Z y agrupados por categorías. |
| **05** | Crear un kardex (a partir de las entradas y salidas de productos). | Tabla que indica:   * Código * Cantidad de productos ingresados. * Cantidad de productos que salen. * Detalle de entrada. * Detalle de salida. * Fecha de entrada. * Fecha de salida. * Costo unitario del producto | Se debe llevar un registro de las entradas y salidas de lacteos, jugos, embutidos, etc. |
| **06** | Actualizar automáticamente el stock. | Tabla que tiene:   * Número de lacteos, jugos, embutidos, etc, disponibles * Nombre de los lacteos, jugos, embutidos, etc que están disponibles * Cantidad de lacteos, jugos, embutidos, etc disponibles | Se necesita tener una tabla con los lacteos, jugos, embutidos, etc a disposición para facilitar el control de ventas. |
| **07** | Generar reportes generales de la entrada y salida de productos | * El reporte incluye: * Código * Fecha * Adjuntado Kardex semanales * Adjuntado desconformidades en el producto. * Breve descripción * Comprobantes de pago. | Es necesario llevar una documentación detallada del flujo de movimientos de productos. |
| **08** | Enviar un email con los reportes adjuntados al Gerente o dueño | En el email se incluirá:   * Reportes generales. * Firma electrónica de remitente. | Toda actividad desarrollada por el personal debe ser consultada y verificada por el gerente. |

Tabla 3:Requisitos funcionales

* 1. **Requerimientos no funcionales**

|  |  |  |  |
| --- | --- | --- | --- |
| **N°** | **Requisito no funcional** | **Detalle** | **Impacto (1-4)** |
| **1** | **Eficiencia** |  |  |
| **1.1** | Diseño | El sistema de debe tener un diseño fácil de manejar e intuitivo al usuario. | 4 |
| **1.2** | Tiempo de respuesta | El sistema debe tener un tiempo de respuesta máximo de 5 segundos. | 4 |
| **1.3** | Ancho de banda | El sistema debe tener un ancho de banda de 10 a 30 Mbps. | 4 |
| **1.4** | Servidor | El sistema estará alojado en un servidor compartido. | 4 |
| **2** | **Seguridad lógica y de datos** |  |  |
| **2.1** | Codificación | El nuevo sistema se desarrolló en su mayoría con patrones de diseño | 4 |
| **2.2** | Restricciones | La restricción de usuarios del sistema estaría asignada según jerarquías. | 3 |
| **2.3** | Seguridad | Asegurar que la información enviada al destinatario se la misma que se transmitida. | 4 |
| **2.4** |  | Se debe disponer de servidores alternativos de transmisión de información entre diferentes puntos. | 3 |
| **2.5** | Copias de seguridad | Asegurar de tener un backup del sistema de almacén | 3 |
| **2.6** | Prevención | Medidas de protección del sistema de almacén frente ataques cibernéticos internos o externos. | 4 |
| **3** | **Seguridad industrial** |  |  |
| **3.1** | Funcionamiento | El sistema se apagará automáticamente cuando la temperatura de ambiente no se adecuada para el funcionamiento del sistema | 4 |
| **3.2** | Riesgos | Monitorear identificar y controlar riesgos posibles. | 4 |
| **3.3** | Seguridad hardware | Uso de una cámara de vigilancia donde se encuentren los dispositivos. | 4 |
| **3.4** | Áreas restringidas | Controlar la entrada a las áreas donde se encuentren los equipos | 4 |
| **4** | **Usabilidad** |  |  |
| **4.1** | Manejo del sistema | Facilidad de interacción y de uso del sistema para fácil manejo del usuario | 4 |
| **4.2** | Frecuencia y gravedad de errores | Plantea la ayuda que se le entrega a los usuarios para apoyarlos cuando deban enfrentar los errores que cometen al usar el sistema | 4 |
| **4.3** | Reportes | El sistema debe reportar mensajes de alerta por mala manipulación orientados los usuarios. | 4 |
| **4.4** | Satisfacción | Los usuarios pueden cumplir con las funciones (actividades) de forma satisfactoria. | 4 |

Tabla 4:Requerimientos no funcionales

* 1. **Modelado de Requerimientos**

### **Diagramas de Caso de Uso**

Administrar la gestión del almacén se desde consultar, registrar y pedir productos de consumo.

-El caso uso inicia cuando el jefe de almacén consulta el stock para verificar los productos faltantes.

-El caso uso finaliza cual el jefe del almacén hace el requerimiento de los productos al gerente o dueño.

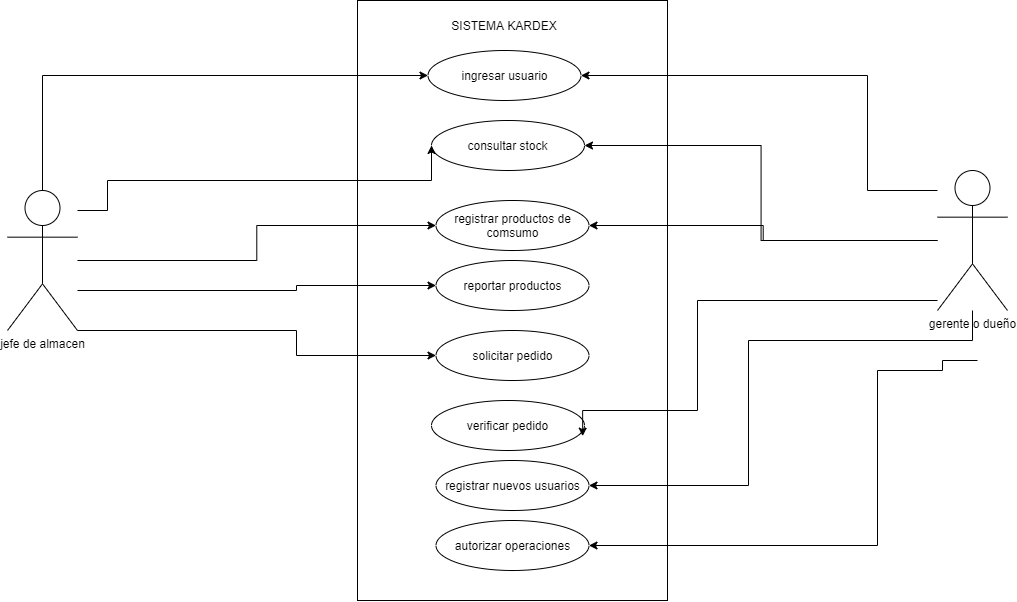


Diagrama 8:Diagramas de Caso de Uso

### **Diagramas de Actividades**

Diagrama de actividad mostrara las acciones que realiza el jefe del almacén y el gerente o dueño tales como:

**Jefe de Almacén**

* Consultar Stock
* solicitar pedido en caso el producto (productos de consumo) este agotado.
* Decepcionar los productos.

**Gerente o Dueño**

* Realizar la Compra
* Registrar Productos

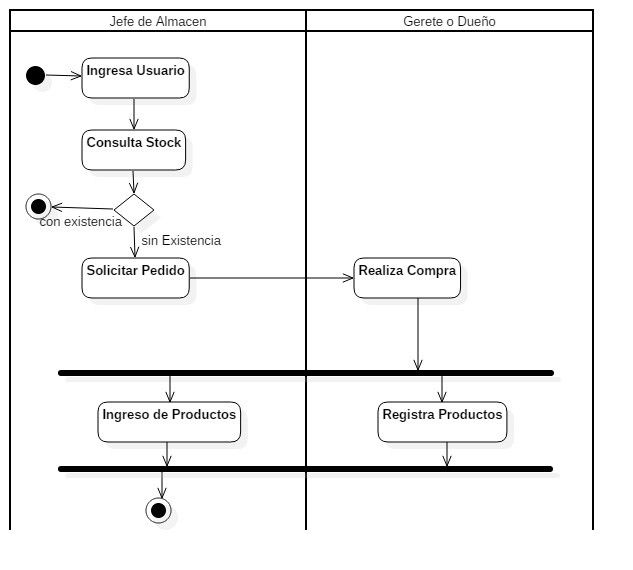


Diagrama 9:Diagramas de Actividades

### **Descripción de Perfiles**

En esta parte describimos los perfiles de los usuarios del sistema y la interactividad y las funcionalidades que se muestra en ambos perfiles

|  |  |  |
| --- | --- | --- |
| **Perfil** | **Usuario jefe de almacén** | |
| Opciones que tiene el perfil | Tipo de acceso | |
| * Registro de productos (ingreso y salida de lacteos, jugos, embutidos, etc) * Consultar stock * Notificar disconformidad * Visualizar de reportes | Lectura  Lectura  Lectura | Escritura |

Tabla 5:Descripción de Perfiles

|  |  |  |
| --- | --- | --- |
| **Perfil** | **Usuario gerente o dueño** | |
| Opciones que tiene el perfil | Tipo de acceso | |
| * Registro de productos (ingreso y salida de lacteos, jugos, embutidos, etc) * Consultar stock * Registrar de usuarios | Escritura  Escritura | Lectura |

Tabla 6:Descripción de Perfiles

2. **IMPLEMENTACIÓN DEL DISEÑO FÍSICA DE LA BASE DE DATOS.**

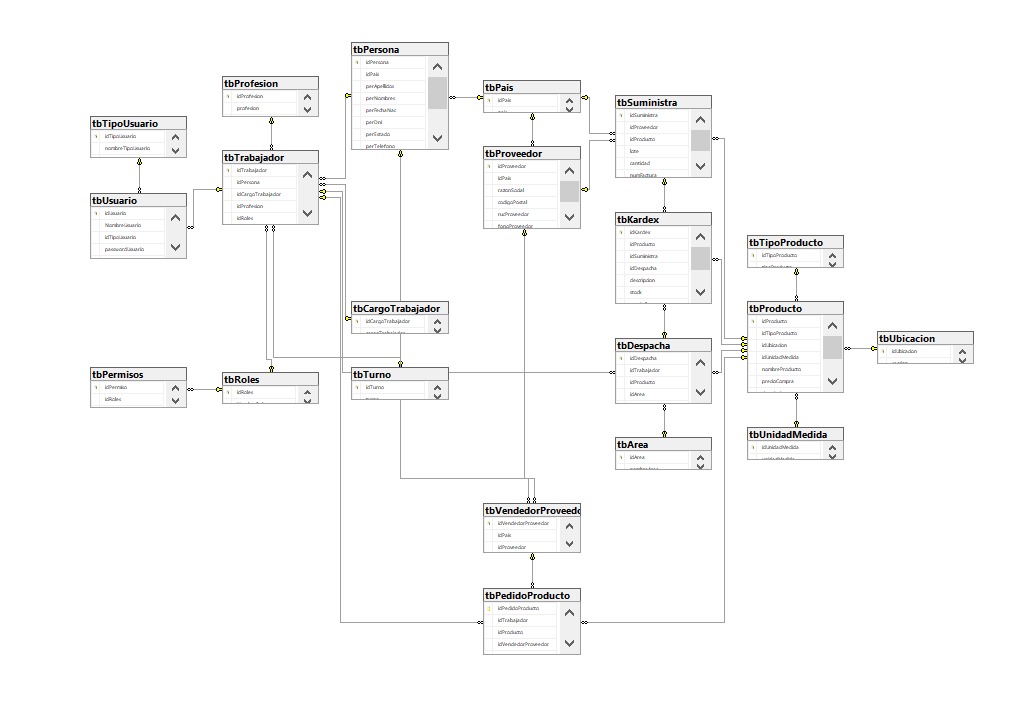


Diagrama 10: Base de datos

1. **IMPLEMENTACIÓN DEL DISEÑO CLASES PARA ACCESO A DATOS.**

Diagrama, Esquemático

Descripción generada automáticamente

Diagrama 1: Clase de datos

1. **MOCKUPS DEL SISTEMA DE CONTROL DE ALMACÉN.**

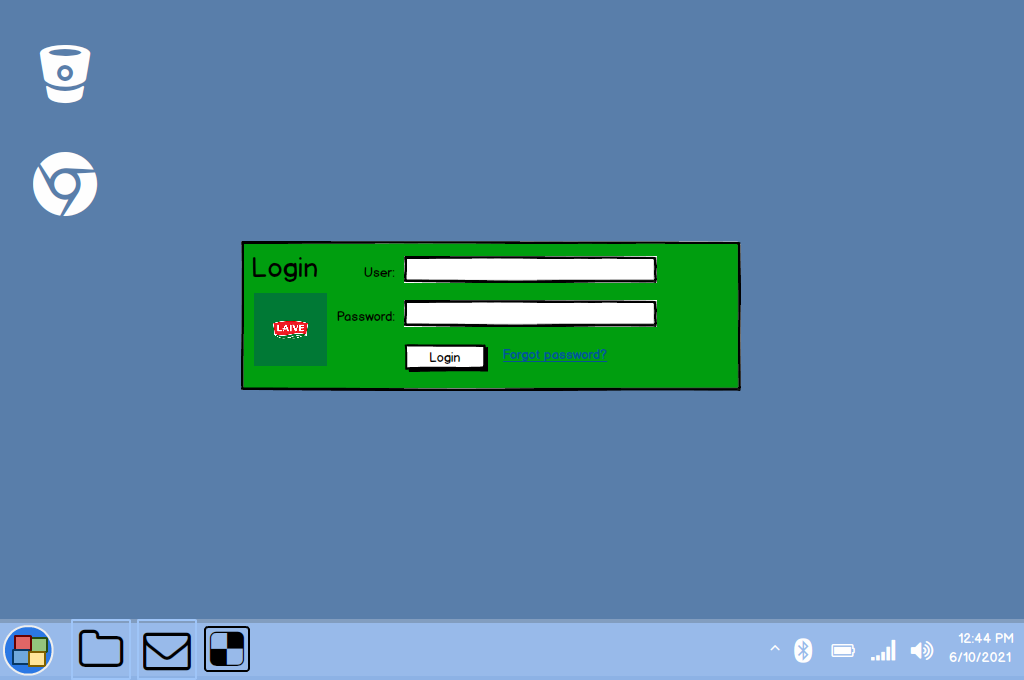


Ilustración 1:Login

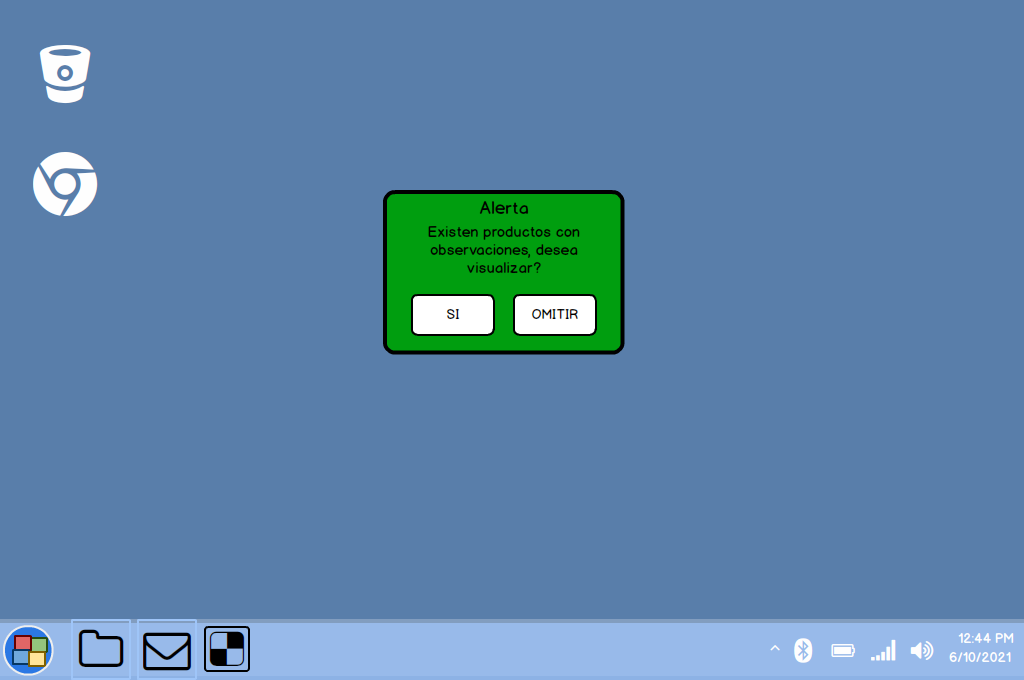


Ilustración 2:Ventana de alerta

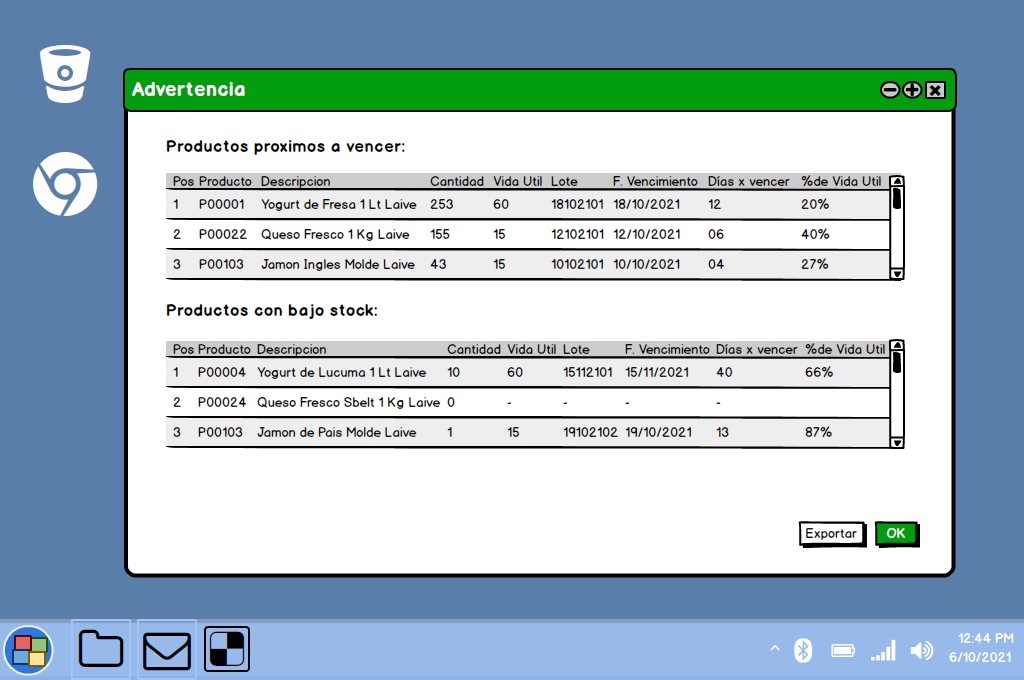


Ilustración 3:Detalle de alerta

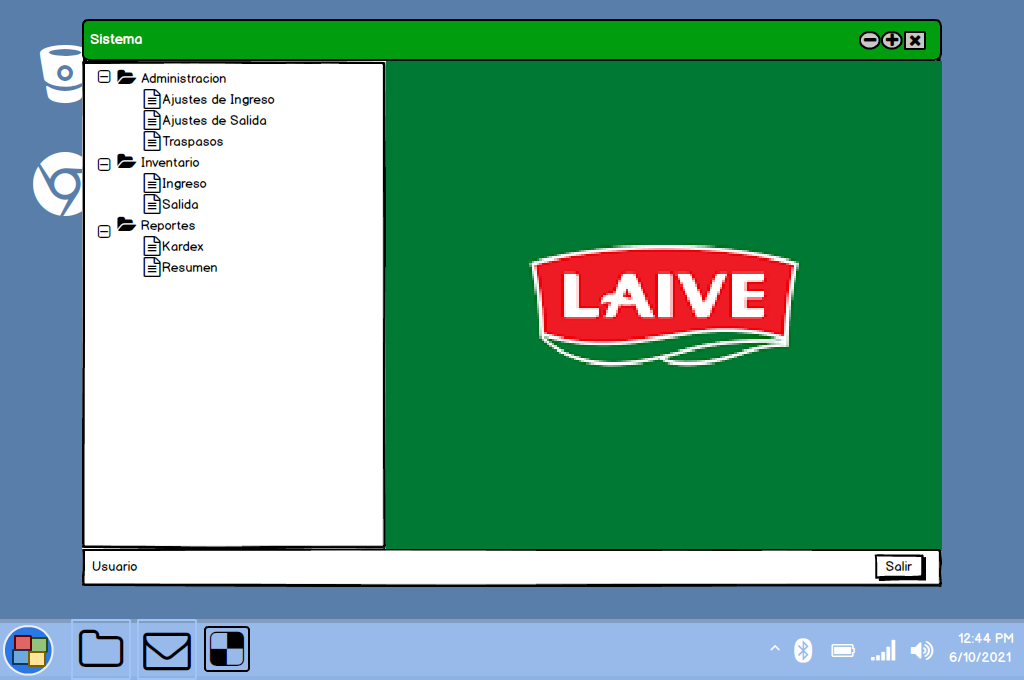


Ilustración 4: Formulario Principal

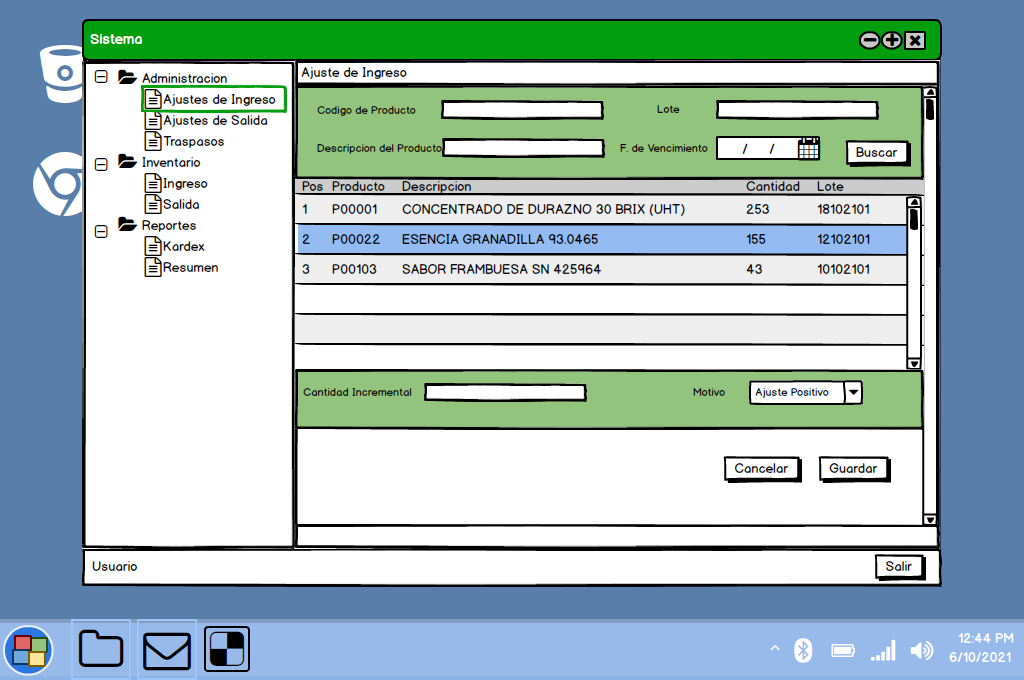


Ilustración 5:Ajustes de Ingresos

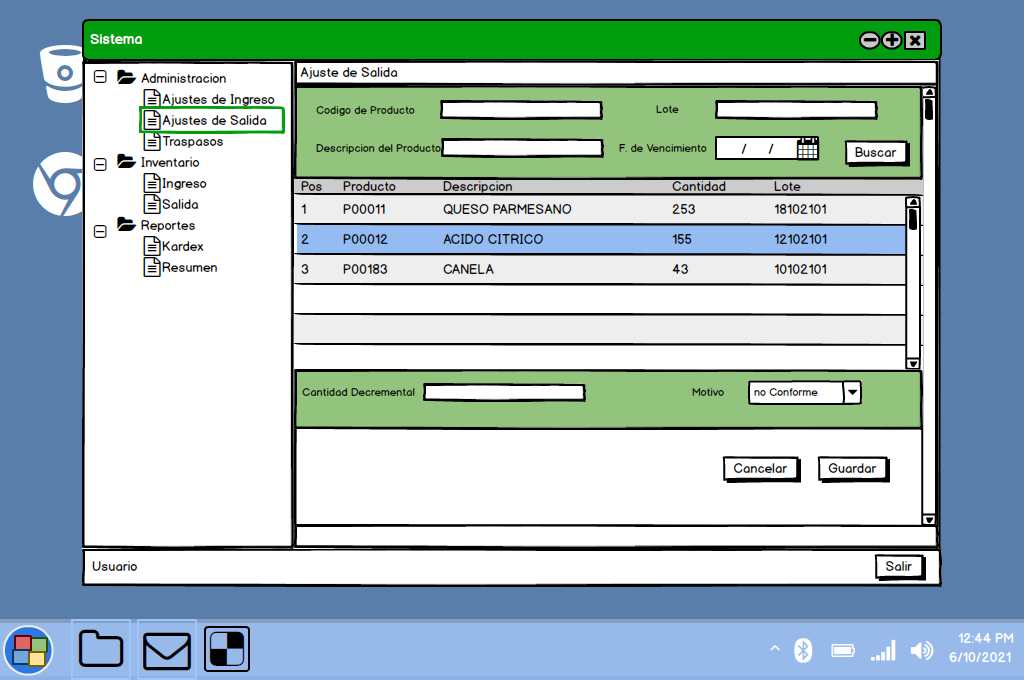


Ilustración 6: Ajustes de salida

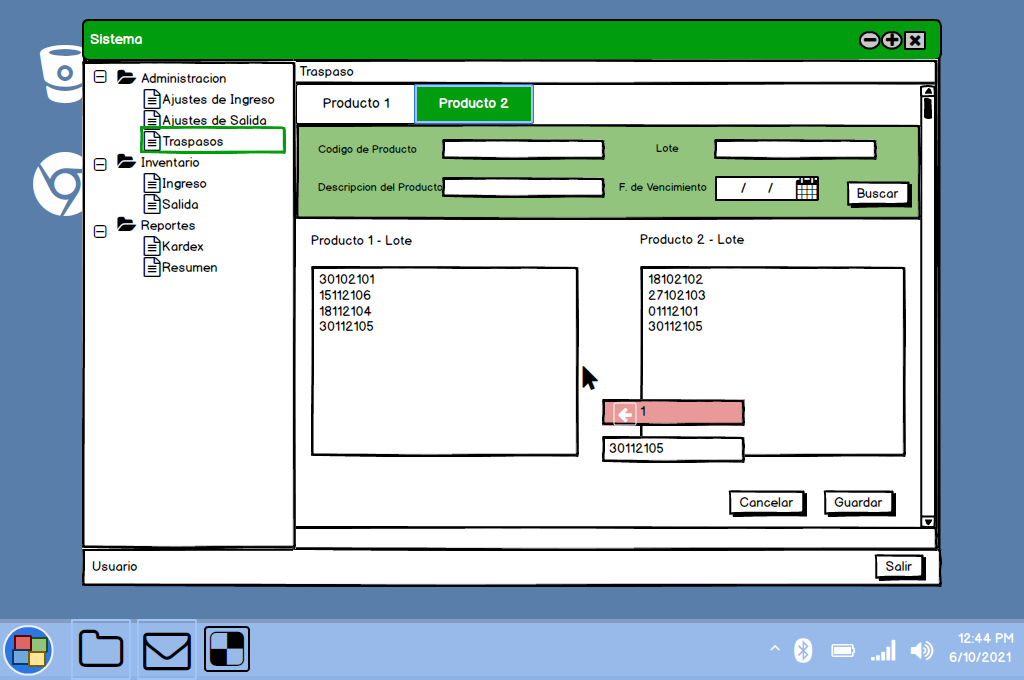


Ilustración 7: Traspasos

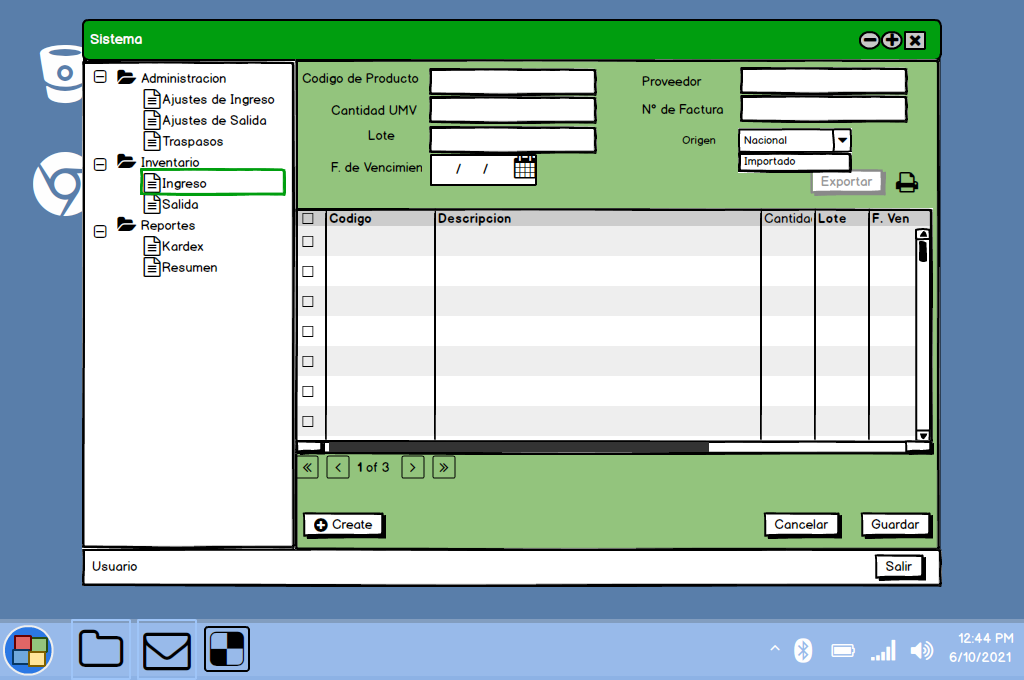


Ilustración 8: Ingresos

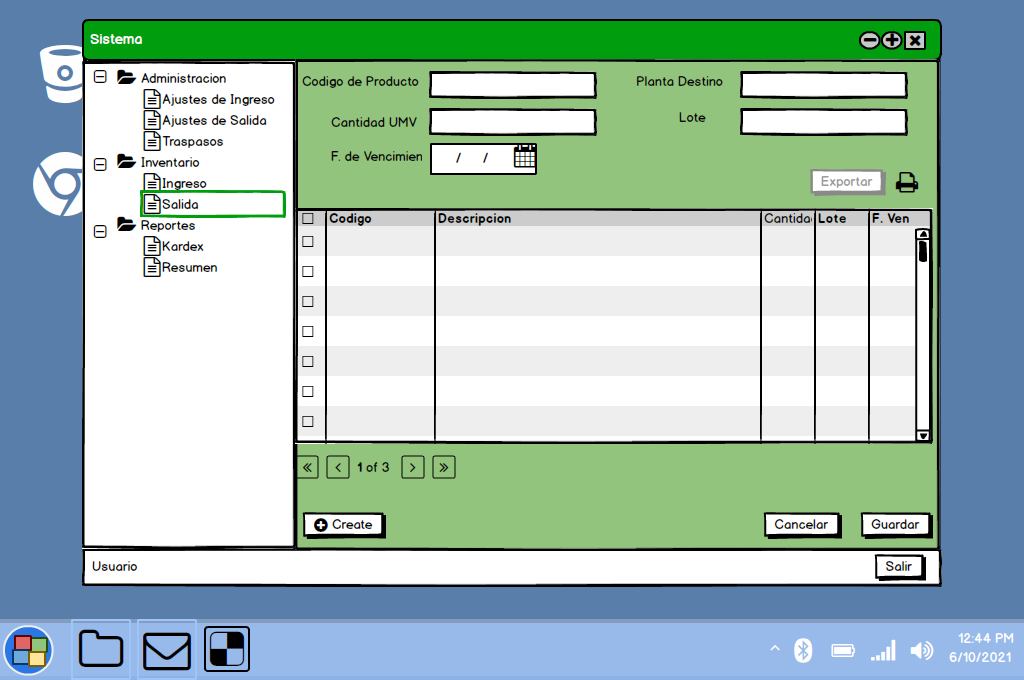


Ilustración 9:Salida

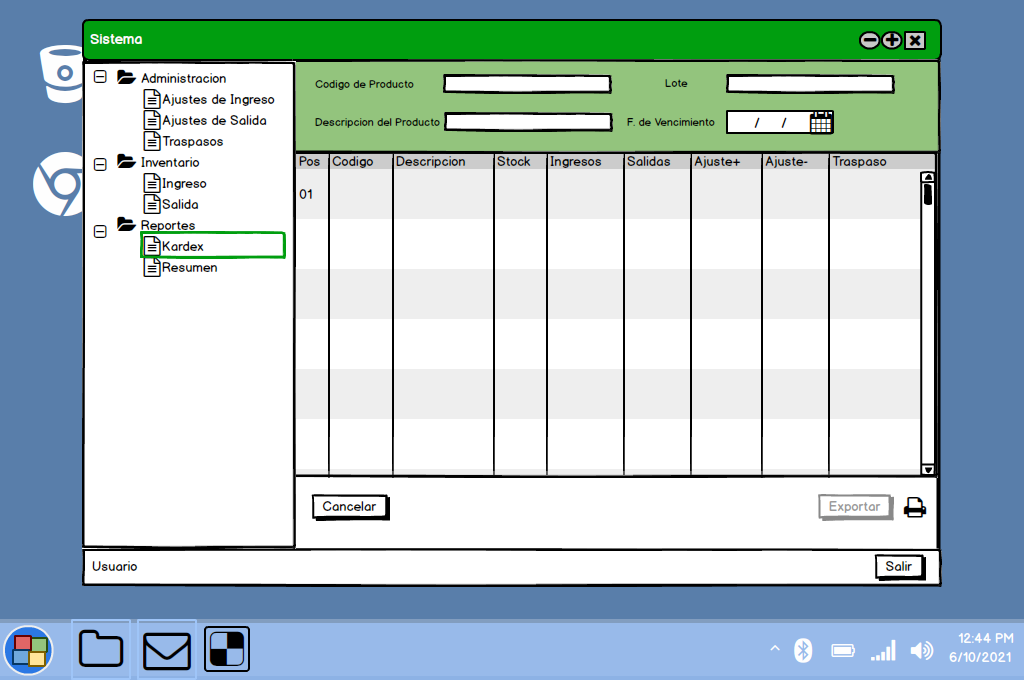
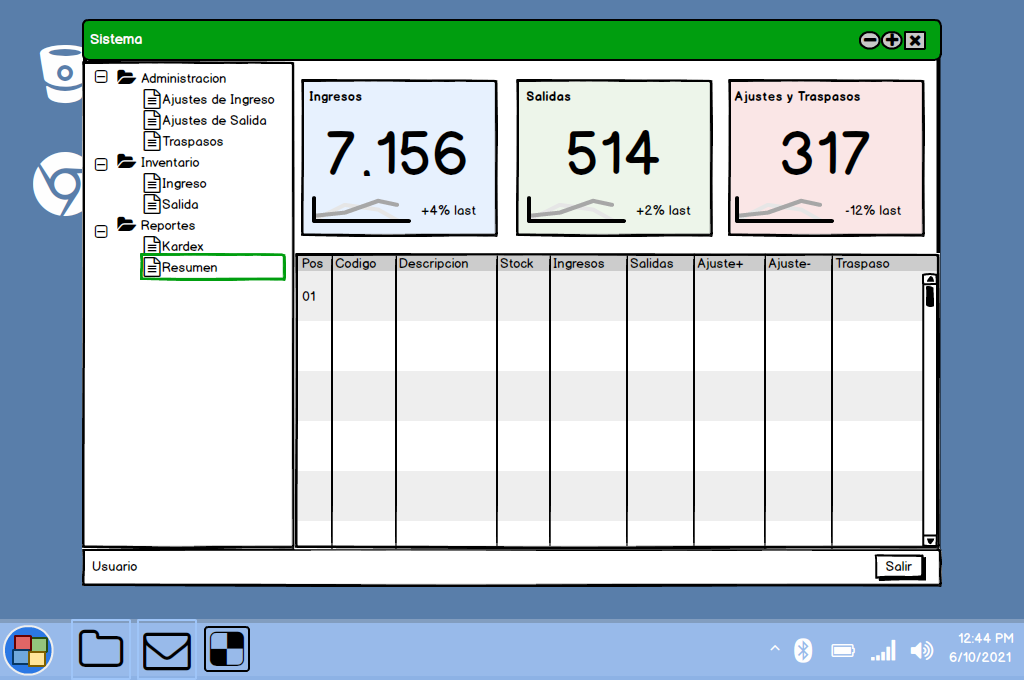


Ilustración 10:Kardex



Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteIlustración 11: Resumen

Ilustración 12: Login Web

Interfaz de usuario gráfica

Descripción generada automáticamente

Ilustración 13: Menú Web

Interfaz de usuario gráfica

Descripción generada automáticamente

Ilustración 14: Kardex Web

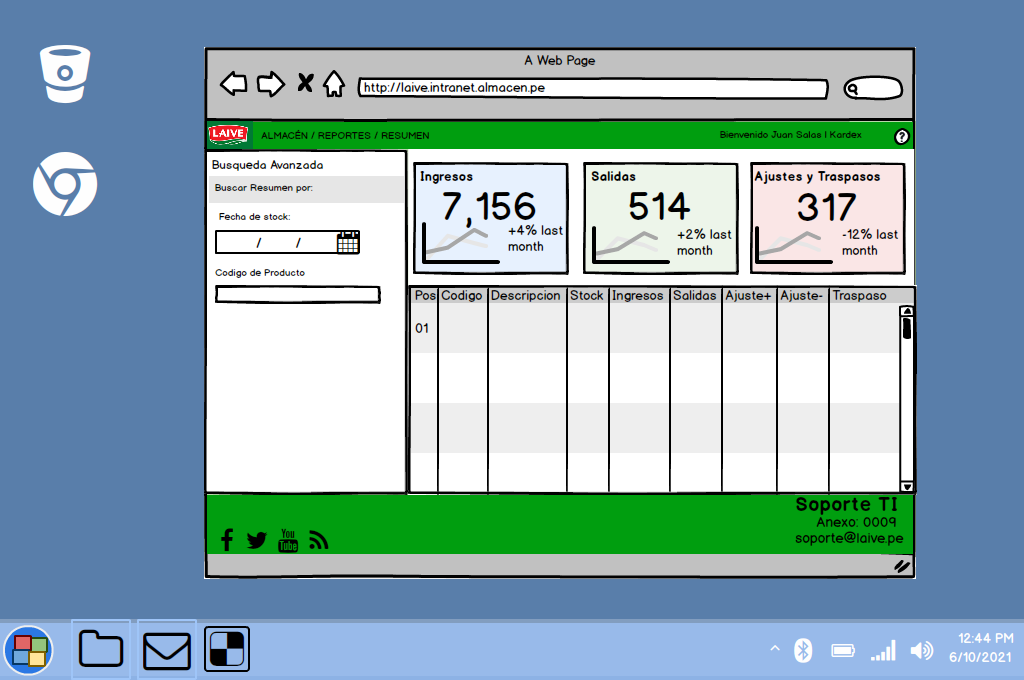


Ilustración 15: Resumen Web

1. **DISEÑO DE CLASES EN VISUAL STUDIO**

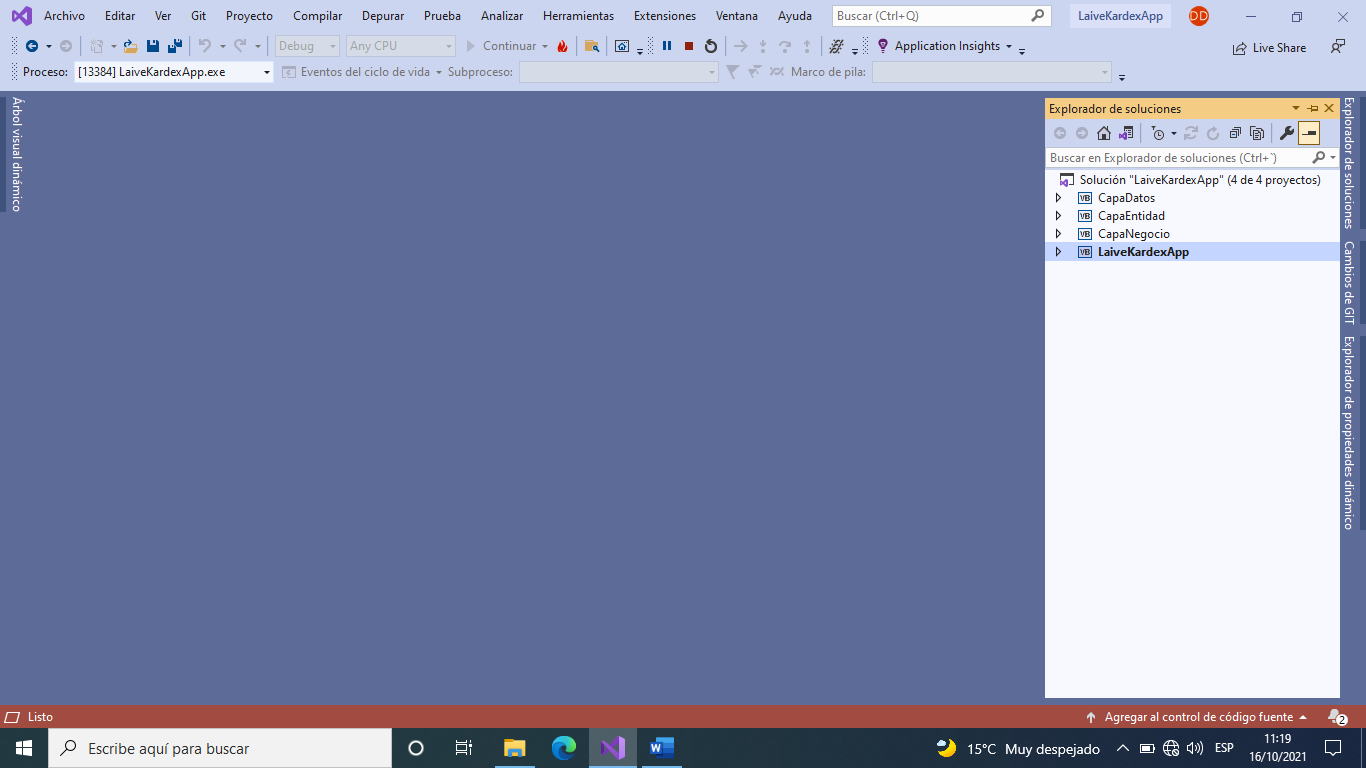


Ilustración 16: Capas del proyecto

1. **DISEÑO DE FORMULARIOS EN VISUAL STUDIO**

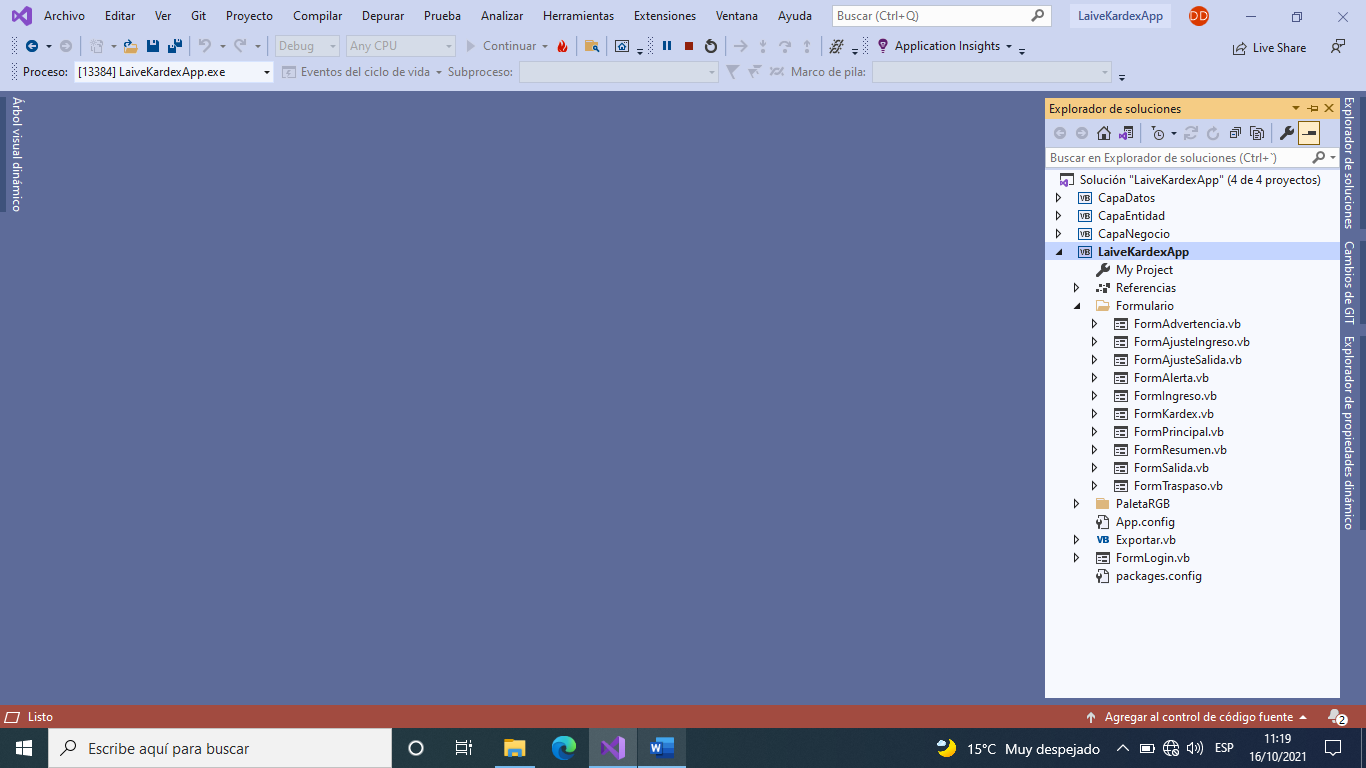


Ilustración 17:Formularios realizados en Visual Studio

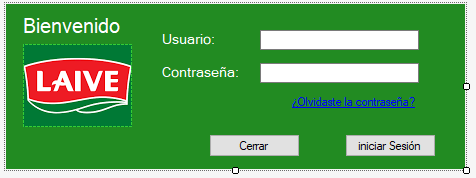


Ilustración 18: Formulario Login

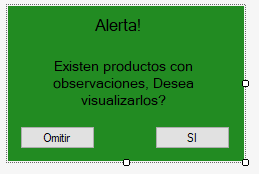


Ilustración 19: Mensaje de Alerta

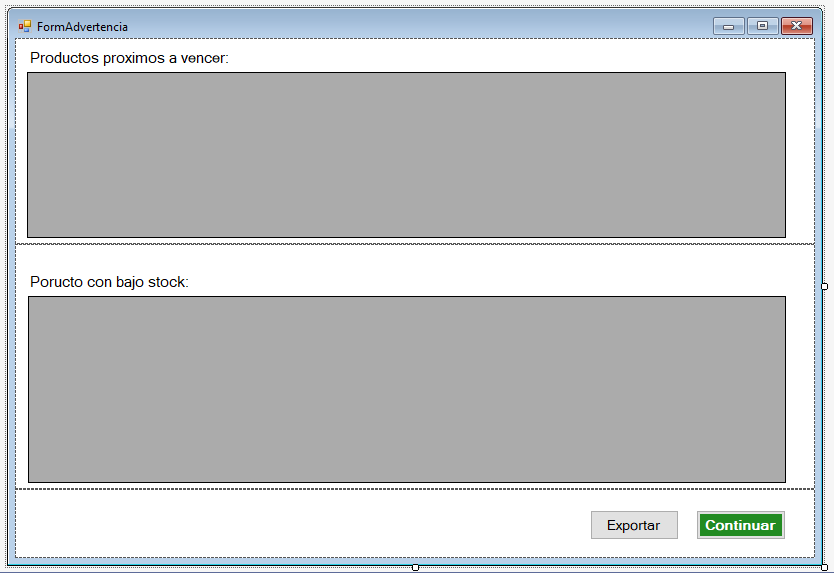


Ilustración 20: Formulario de Advertencia de productos por vencer y bajo stock

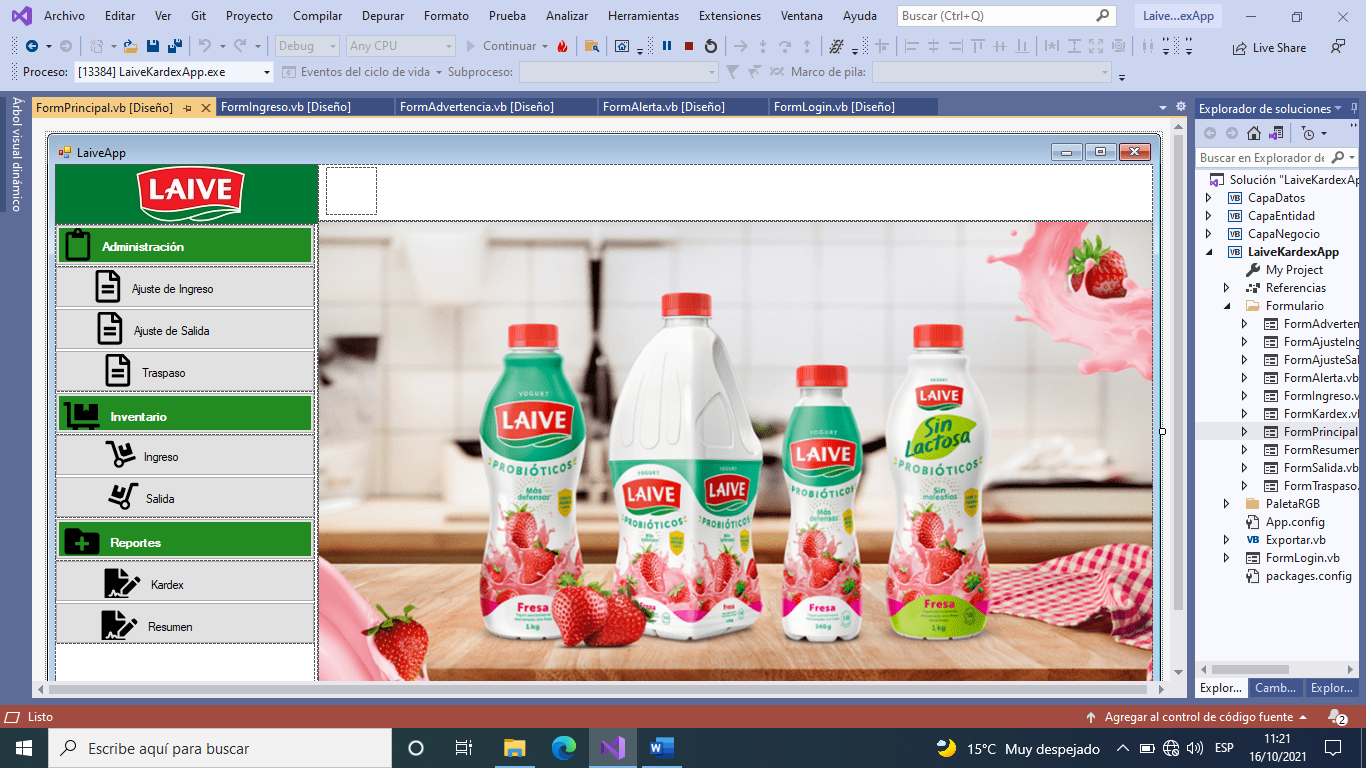


Ilustración 21: Formulario principal con el menú de opciones

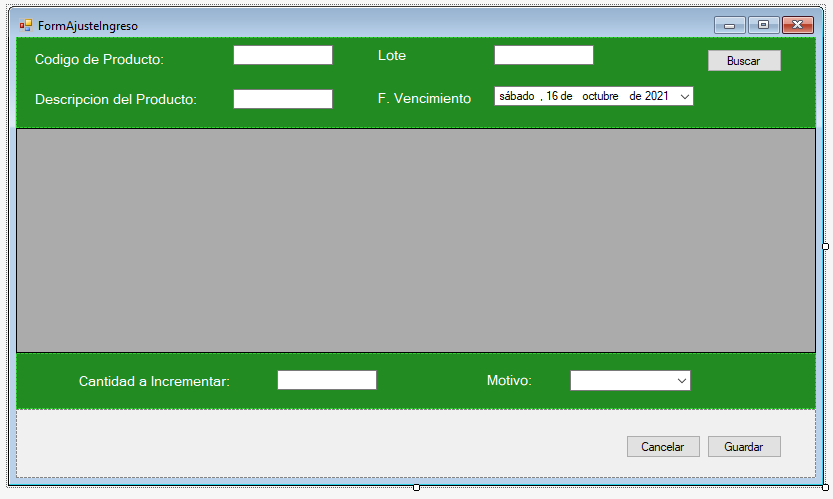


Ilustración 22:Formulario de ajuste de ingreso (Administración)

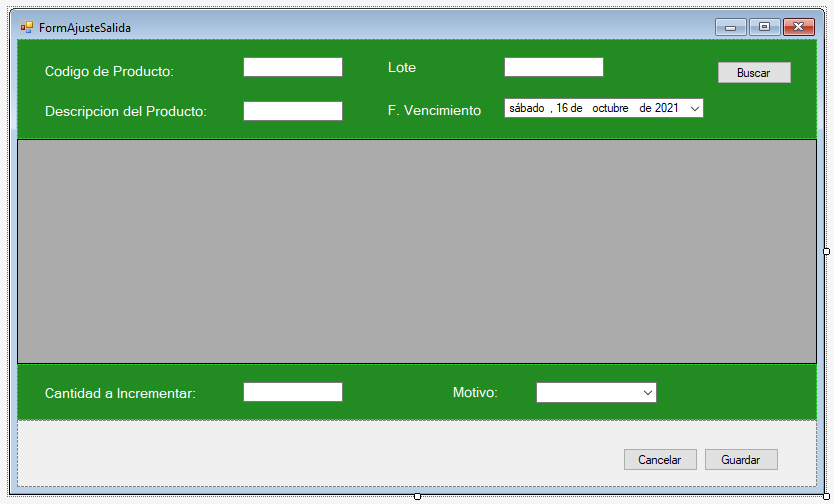


Ilustración 23: Formulario de ajuste de Salida(Administración)

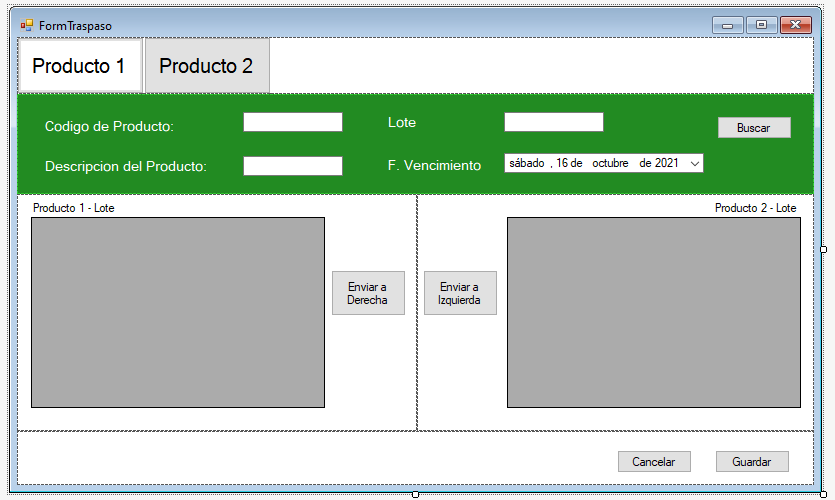


Ilustración 24: Formulario traspaso (Administración)

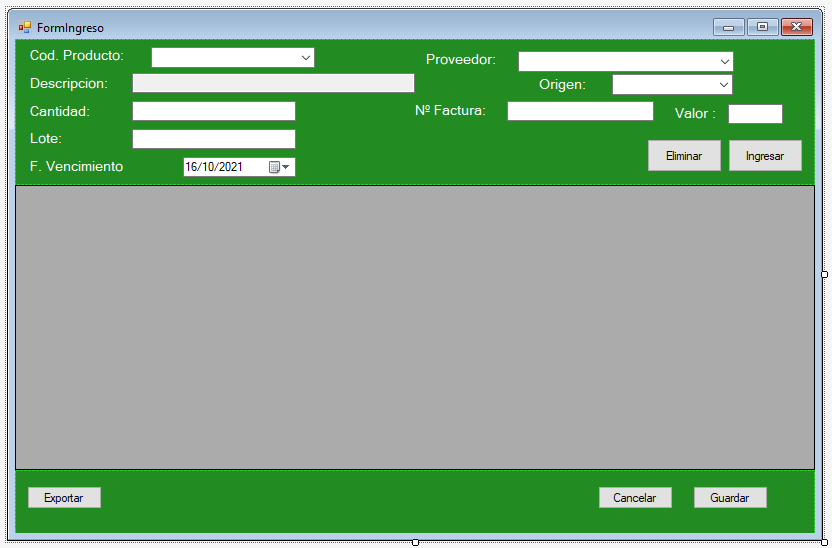


Ilustración 25: Formulario de Ingreso de productos (Inventario)

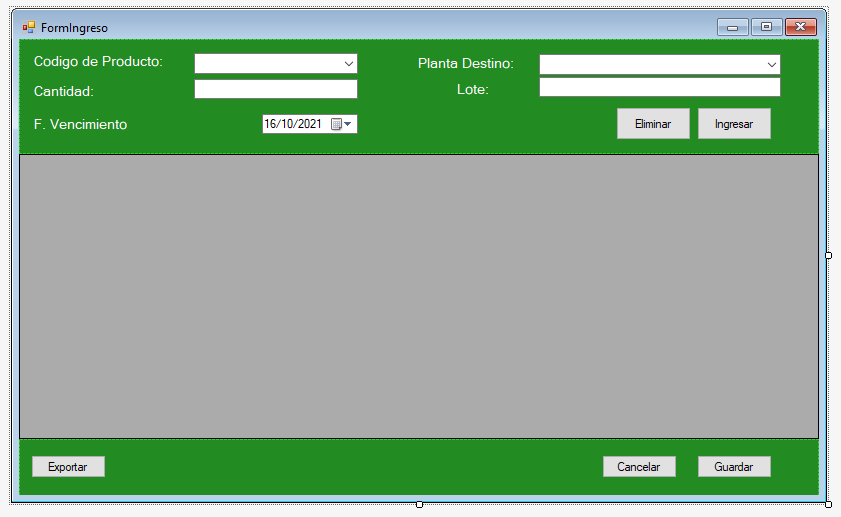


Ilustración 26: Formulario de Salida de productos (Inventario)

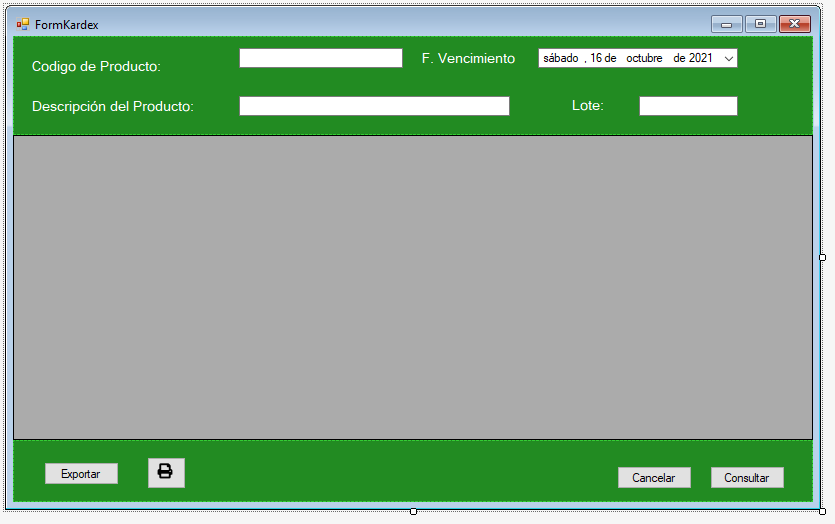


Ilustración 27: Formulario de Kardex (Reportes)

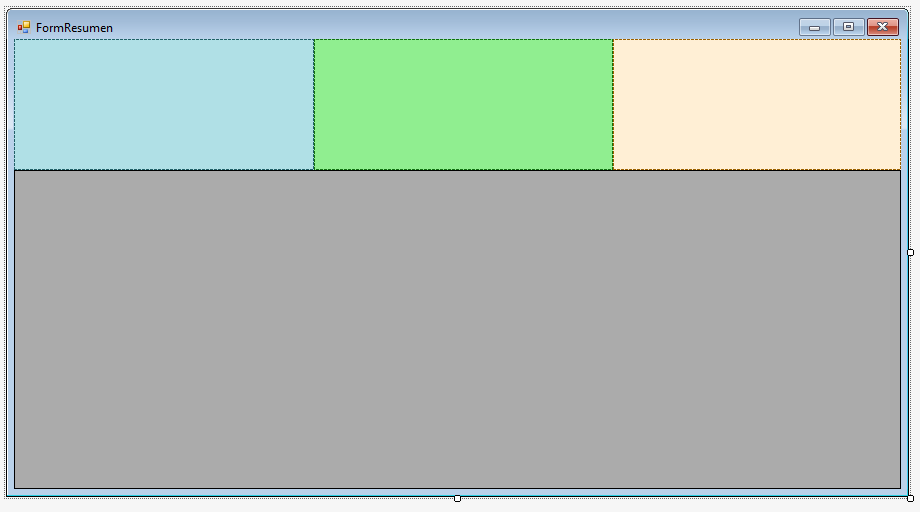


Ilustración 28: Formulario de Resumen (Reportes)

1. **BASE DE DATOS EN SQL**

IF DB\_ID('BD\_EmpreLaive') IS NOT NULL

BEGIN

USE master

DROP DATABASE BD\_EmpreLaive

END

CREATE DATABASE BD\_EmpreLaive

USE BD\_EmpreLaive

--TABLA KARDEX

IF OBJECT\_ID('tbKardex') IS NOT NULL

BEGIN

DROP TABLE tbKardex

END

ELSE

BEGIN

CREATE TABLE tbKardex(

idKardex varchar(10) PRIMARY KEY,

idProducto varchar(10) NOT NULL,

idSuministra varchar(10) NOT NULL,

idDespacha varchar(10) NOT NULL,

descripcion varchar(50) NOT NULL,

stock int NOT NULL,

precioCompra float NOT NULL,

precioVenta float NOT NULL,

cantidad int NOT NULL

)

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--TABLA TURNO

IF OBJECT\_ID('tbTurno') IS NOT NULL

BEGIN

DROP TABLE tbTurno

--PRINT 'La tabla [tbTurno] fue eliminada'

END

ELSE

BEGIN

--En esta tabla se registran los turnos de los trabajadores Mañana/Tarde/Noche

CREATE TABLE tbTurno(idTurno varchar(10) PRIMARY KEY,

turno varchar(15) NOT NULL)

--PRINT 'La tabla [tbTurno] fue creada con exito...!'

END

GO

--TABLA TRABAJADOR

IF OBJECT\_ID('tbTrabajador') IS NOT NULL

BEGIN

DROP TABLE tbTrabajador

END

ELSE

BEGIN

CREATE TABLE tbTrabajador(

idTrabajador varchar(10) NOT NULL PRIMARY KEY, --sera su dni su id

idPersona varchar(10) NOT NULL,

idCargoTrabajador varchar(10) NOT NULL,

idProfesion varchar(10) NOT NULL,

idRoles varchar(10) NOT NULL,

idTurno varchar(10) NOT NULL,

fechaInicio date NOT NULL)

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--TABLA SUMINISTRA

IF OBJECT\_ID('tbSuministra') IS NOT NULL

BEGIN

DROP TABLE tbSuministra

END

ELSE

BEGIN

CREATE TABLE tbSuministra

(

idSuministra varchar(10) NOT NULL PRIMARY KEY,

idProveedor varchar(10) NOT NULL,

idProducto varchar(10) NOT NULL,

lote varchar(10) NOT NULL,

cantidad int NOT NULL,

numFactura varchar(10) NOT NULL,

paisOrigen varchar(10) NOT NULL,

precio\_compra float NOT NULL

)

END

GO

--TABLA DESPACHA

IF OBJECT\_ID('tbDespacha') IS NOT NULL

BEGIN

DROP TABLE tbDespacha

END

ELSE

BEGIN

CREATE TABLE tbDespacha

(

idDespacha varchar(10) NOT NULL PRIMARY KEY,

idTrabajador varchar(10) NOT NULL,

idProducto varchar(10) NOT NULL,

idArea varchar(10) NOT NULL,

cantidad int NOT NULL,

lote varchar(10) NOT NULL

)

END

GO

IF OBJECT\_ID('tbPais') IS NOT NULL

BEGIN

DROP TABLE tbPais

--PRINT 'La tabla [tbPais] fue eliminada'

END

ELSE

BEGIN

--Esta tabla registra los paises

CREATE TABLE tbPais(idPais varchar(10) NOT NULL PRIMARY KEY,

pais varchar(25) NOT NULL)

--PRINT 'La tabla [tbPais] fue creada con exito...!'

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--TABLA PROFESION

IF OBJECT\_ID('tbProfesion') IS NOT NULL

BEGIN

DROP TABLE tbProfesion

--PRINT 'La tabla [tbProfesion] fue eliminada'

END

ELSE

BEGIN

--Esta tabla registra el grado academico

CREATE TABLE tbProfesion(idProfesion varchar(10) NOT NULL PRIMARY KEY,

profesion varchar(25) NOT NULL,

estado bit NOT NULL) --0(En curso) 1(Terminado)

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--TABLA CARGO TRABAJADOR

IF OBJECT\_ID('tbCargoTrabajador') IS NOT NULL

BEGIN

DROP TABLE tbCargoTrabajador

END

ELSE

BEGIN

CREATE TABLE tbCargoTrabajador(idCargoTrabajador varchar(10) NOT NULL PRIMARY KEY,

cargoTrabajador varchar(25) NOT NULL)

END

GO

--TABLA AREA

IF OBJECT\_ID('tbArea') IS NOT NULL

BEGIN

DROP TABLE tbArea

END

ELSE

BEGIN

CREATE TABLE tbArea(idArea varchar(10) NOT NULL PRIMARY KEY,

nombreArea varchar(25) NOT NULL

)

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--TABLA ROLES

IF OBJECT\_ID('tbRoles') IS NOT NULL

BEGIN

DROP TABLE tbRoles

END

ELSE

BEGIN

CREATE TABLE tbRoles (

idRoles varchar(10) NOT NULL PRIMARY KEY,

NombreRol varchar(30) NOT NULL

)

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--TABLA PERMISOS

IF OBJECT\_ID('tbPermisos') IS NOT NULL

BEGIN

DROP TABLE tbPermisos

END

ELSE

BEGIN

CREATE TABLE tbPermisos(

idPermiso varchar(10) NOT NULL PRIMARY KEY,

idRoles varchar(10) NOT NULL,

NombrePermiso varchar(30) NOT NULL

)

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

IF OBJECT\_ID('tbUsuario') IS NOT NULL

BEGIN

DROP TABLE tbUsuario

END

ELSE

BEGIN

CREATE TABLE tbUsuario(

idUsuario varchar(10) NOT NULL PRIMARY KEY,

NombreUsuario varchar(40) NOT NULL,

idTipoUsuario varchar(10) NOT NULL,

--USUARIO IMAGEN

passwordUsuario varchar(40) NOT NULL,

idTrabajador varchar(10) NOT NULL,

estadoUsuario bit NOT NULL,

)

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

IF OBJECT\_ID('tbTipoUsuario') IS NOT NULL

BEGIN

DROP TABLE tbTipoUsuario

END

ELSE

BEGIN

CREATE TABLE tbTipoUsuario(

idTipoUsuario varchar(10) NOT NULL PRIMARY KEY,

nombreTipoUsuario varchar(40) NOT NULL,

descripcionTipoUsuario varchar(50) NOT NULL

)

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--TABLA PERSONA

IF OBJECT\_ID('tbPersona') IS NOT NULL

BEGIN

DROP TABLE tbPersona

END

ELSE

BEGIN

CREATE TABLE tbPersona(

idPersona varchar(10) PRIMARY KEY,

idPais varchar(10) NOT NULL,

perApellidos varchar(50) NOT NULL,

perNombres varchar(50) NOT NULL,

perFechaNac date NOT NULL,

--perFoto

perDni varchar(8) NOT NULL,

perEstado bit NOT NULL,

perTelefono varchar(40) NOT NULL,

perCorreo varchar(40) NULL,

perDireccion varchar(40) NULL,

perSexo bit NOT NULL)

END

GO

--TABLA TIPO PRODUCTO

IF OBJECT\_ID('tbTipoProducto') IS NOT NULL

BEGIN

DROP TABLE tbTipoProducto

--PRINT 'La tabla [tbTipoProducto] fue eliminada'

END

ELSE

BEGIN

--en esta tabla se registran los diversos tipos de producto

CREATE TABLE tbTipoProducto(idTipoProducto varchar(10) PRIMARY KEY,

tipoProducto varchar(25) NOT NULL)

END

GO

--TABLA UNIDAD DE MEDIDA

IF OBJECT\_ID('tbUnidadMedida') IS NOT NULL

BEGIN

DROP TABLE tbUnidadMedida

--PRINT 'La tabla [tbUnidadMedida] fue eliminada'

END

ELSE

BEGIN

CREATE TABLE tbUnidadMedida(idUnidadMedida varchar(10) PRIMARY KEY,

unidadMedida varchar(15) NOT NULL) -- Kilos, Litros, Metros, Centimetros,

--PRINT 'La tabla [tbUnidadMedida] fue creada con exito...!'

END

GO

-------------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------------

--TABLA UBICACION

IF OBJECT\_ID('tbUbicacion') IS NOT NULL

BEGIN

DROP TABLE tbUbicacion

--PRINT 'La tabla [tbUbicacion] fue eliminada'

END

ELSE

BEGIN

CREATE TABLE tbUbicacion(idUbicacion varchar(10) PRIMARY KEY, --Lugar fisico asignado segun producto

seccion varchar(25) NOT NULL)

--PRINT 'La tabla [tbUbicacion] fue creada con exito...!'

END

GO

-------------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------------

--TABLA PRODUCTO

IF OBJECT\_ID('tbProducto') IS NOT NULL

BEGIN

DROP TABLE tbProducto

END

ELSE

BEGIN

CREATE TABLE tbProducto(

idProducto varchar(10) PRIMARY KEY,

idTipoProducto varchar(10) NOT NULL,

idUbicacion varchar(10) NOT NULL,

idUnidadMedida varchar(10) NOT NULL,

nombreProducto varchar(100) NOT NULL,--

precioCompra float NOT NULL,

descripcion varchar(350),

stock int NOT NULL,

caducidad date NOT NULL

)

END

-------------------------------------------------------------------------------------------

-------------------------------------------------------------------------------------------

--TABLA PEDIDO PRODUCTO

IF OBJECT\_ID('tbPedidoProducto') IS NOT NULL

BEGIN

DROP TABLE tbPedidoProducto

--PRINT 'La tabla [tbPedidoProducto] fue eliminada'

END

ELSE

BEGIN

CREATE TABLE tbPedidoProducto(idPedidoProducto varchar(10) PRIMARY KEY,

idTrabajador varchar(10) NOT NULL,

idProducto varchar(10) NOT NULL,--OK

idVendedorProveedor varchar(10) NOT NULL, --OK

cantidadPedido int NOT NULL,

estadoPedido bit NOT NULL) -- 0 (En camino), 1 (Recibido)

--PRINT 'La tabla [tbPedidoProducto] fue creada con exito...!'

END

GO

--------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------------

--TABLA VENDEDOR PROVEEDOR

IF OBJECT\_ID('tbVendedorProveedor') IS NOT NULL

BEGIN

DROP TABLE tbVendedorProveedor

--PRINT 'La tabla [tbVendedorProveedor] fue eliminada'

END

ELSE

BEGIN

CREATE TABLE tbVendedorProveedor(idVendedorProveedor varchar(10) PRIMARY KEY, --Numero de documento

idPais varchar(10) NOT NULL,

idProveedor varchar(10) NOT NULL,

idPersona varchar(10) NOT NULL )

--PRINT 'La tabla [tbVendedorProveedor] fue creada con exito...!'

END

GO

--------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------------

--TABLA PROVEEDOR

--Tablas referentes a proceso de compra

IF OBJECT\_ID('tbProveedor') IS NOT NULL

BEGIN

DROP TABLE tbProveedor

--PRINT 'La tabla [tbProveedor] fue eliminada'

END

ELSE

BEGIN

CREATE TABLE tbProveedor(idProveedor varchar(10) PRIMARY KEY,

idPais varchar(10) NOT NULL,

razonSocial varchar(50) NOT NULL,

codigoPostal varchar(20) NOT NULL,

rucProveedor varchar(13) NOT NULL,

fonoProveedor varchar(12) NOT NULL,--

emailProveedor varchar(50) NOT NULL,

direccion varchar(50) NOT NULL)

--PRINT 'La tabla [tbProveedor] fue creada con exito...!'

END

GO

select \* from tbProveedor

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--RELACIONES DE TRABAJADOR

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--RELACION TABLA TRABAJADOR CON LA TABLA CARDO DEL TRABAJADOR

IF OBJECT\_ID('FK\_tbTrabajador\_tbCargoTrabajador') IS NOT NULL

BEGIN

ALTER TABLE tbTrabajador

DROP CONSTRAINT FK\_tbTrabajador\_tbCargoTrabajador

PRINT 'la relacion FK\_tbTrabajador\_tbCargoTrabajador fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbTrabajador

ADD CONSTRAINT FK\_tbTrabajador\_tbCargoTrabajador

FOREIGN KEY (idCargoTrabajador) REFERENCES tbCargoTrabajador(idCargoTrabajador)

END

GO

IF OBJECT\_ID('FK\_tbTrabajador\_tbPersona') IS NOT NULL

BEGIN

ALTER TABLE tbTrabajador

DROP CONSTRAINT FK\_tbTrabajador\_tbPersona

PRINT 'la relacion FK\_tbTrabajador\_tbPersona fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbTrabajador

ADD CONSTRAINT FK\_tbTrabajador\_tbPersona

FOREIGN KEY (idPersona) REFERENCES tbPersona(idPersona)

END

GO

--RELACION TABLA TRABAJADOR CON LA TABLA PROFESION

IF OBJECT\_ID('FK\_tbTrabajador\_tbProfesion') IS NOT NULL

BEGIN

ALTER TABLE tbTrabajador

DROP CONSTRAINT FK\_tbTrabajador\_tbProfesion

PRINT 'la relacion FK\_tbTrabajador\_tbProfesion fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbTrabajador

ADD CONSTRAINT FK\_tbTrabajador\_tbProfesion

FOREIGN KEY (idProfesion) REFERENCES tbProfesion(idProfesion)

END

GO

--RELACION TABLA TRABAJADOR CON LA TABLA ROLES

IF OBJECT\_ID('FK\_tbTrabajador\_tbRoles') IS NOT NULL

BEGIN

ALTER TABLE tbTrabajador

DROP CONSTRAINT FK\_tbTrabajador\_tbRoles

PRINT 'la relacion FK\_tbTrabajador\_tbRoles fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbTrabajador

ADD CONSTRAINT FK\_tbTrabajador\_tbRoles

FOREIGN KEY (idRoles) REFERENCES tbRoles(idRoles)

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--PERMISOS TABLA PERMISOS

--RELACION TABLA PERMISOS CON LA TABLA ROLES

IF OBJECT\_ID('FK\_tbPermisos\_tbRoles') IS NOT NULL

BEGIN

ALTER TABLE tbPermisos

DROP CONSTRAINT FK\_tbPermisos\_tbRoles

PRINT 'la relacion FK\_tbPermisos\_tbRoles fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbPermisos

ADD CONSTRAINT FK\_tbPermisos\_tbRoles

FOREIGN KEY (idRoles) REFERENCES tbRoles (idRoles)

END

GO

--RELACION TABLA USUARIO CON LA TABLA TRABAJADOR

IF OBJECT\_ID('FK\_tbUsuario\_tbTrabajador') IS NOT NULL

BEGIN

ALTER TABLE tbUsuario

DROP CONSTRAINT FK\_tbUsuario\_tbTrabajador

PRINT 'la relacion FK\_tbUsuario\_tbTrabajador fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbUsuario

ADD CONSTRAINT FK\_tbUsuario\_tbTrabajador

FOREIGN KEY (idTrabajador) REFERENCES tbTrabajador (idTrabajador)

END

GO

--RELACION TABLA USUARIO CON LA TABLA TIPO USUARIO

IF OBJECT\_ID('FK\_tbUsuario\_tbTipoTrabajador') IS NOT NULL

BEGIN

ALTER TABLE tbUsuario

DROP CONSTRAINT FK\_tbUsuario\_tbTipoTrabajador

PRINT 'la relacion FK\_tbUsuario\_tbTipoTrabajador fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbUsuario

ADD CONSTRAINT FK\_tbUsuario\_tbTipoTrabajador

FOREIGN KEY (idTipoUsuario) REFERENCES tbTipoUsuario (idTipoUsuario)

END

GO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--RELACIONES PROVEEDOR

IF OBJECT\_ID('FK\_tbProveedor\_FK\_Pais') IS NOT NULL

BEGIN

ALTER TABLE tbProveedor

DROP CONSTRAINT FK\_tbProveedor\_FK\_Pais

PRINT 'La relacion [FK\_tbProveedor\_FK\_Pais] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbProveedor

ADD CONSTRAINT FK\_tbProveedor\_FK\_Pais

FOREIGN KEY(idPais) REFERENCES tbPais(idPais)

END

GO

--RELACIONES PRODUCTO

------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------

--tbUnidadMedida(PK) tbProducto(FK)

IF OBJECT\_ID('FK\_Producto\_tbUnidadMedida') IS NOT NULL

BEGIN

ALTER TABLE tbProducto

DROP CONSTRAINT FK\_Producto\_tbUnidadMedida

PRINT 'La relacion [FK\_Producto\_tbUnidadMedida] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbProducto

ADD CONSTRAINT FK\_Producto\_tbUnidadMedida

FOREIGN KEY (idUnidadMedida) REFERENCES tbUnidadMedida(idUnidadMedida)

END

GO

--tbTipoProducto(PK) tbProducto(FK)

IF OBJECT\_ID('FK\_Producto\_tbTipoProducto') IS NOT NULL

BEGIN

ALTER TABLE tbProducto

DROP CONSTRAINT FK\_Producto\_tbTipoProducto

PRINT 'La relacion [FK\_Producto\_tbTipoProducto] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbProducto

ADD CONSTRAINT FK\_Producto\_tbTipoProducto

FOREIGN KEY (idTipoProducto) REFERENCES tbTipoProducto(idTipoProducto)

END

GO

--tbUbicacion(PK) tbProducto(FK)

IF OBJECT\_ID('FK\_Producto\_tbUbicacion') IS NOT NULL

BEGIN

ALTER TABLE tbProducto

DROP CONSTRAINT FK\_Producto\_tbUbicacion

PRINT 'La relacion [FK\_Producto\_tbUbicacion] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbProducto

ADD CONSTRAINT FK\_Producto\_tbUbicacion

FOREIGN KEY (idUbicacion) REFERENCES tbUbicacion(idUbicacion)

END

GO

--tbProducto(PK) tbPedidoProducto(FK)

IF OBJECT\_ID('FK\_PedidoProducto\_tbProducto') IS NOT NULL

BEGIN

ALTER TABLE tbPedidoProducto

DROP CONSTRAINT FK\_PedidoProducto\_tbProducto

PRINT 'La relacion [FK\_PedidoProducto\_tbTrabajador] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbPedidoProducto

ADD CONSTRAINT FK\_PedidoProducto\_tbProducto

FOREIGN KEY (idProducto) REFERENCES tbProducto(idProducto)

END

GO

--tbVendedorProveedor(PK) tbPedidoProducto(FK)

IF OBJECT\_ID('FK\_PedidoProducto\_tbVendedorProveedor') IS NOT NULL

BEGIN

ALTER TABLE tbPedidoProducto

DROP CONSTRAINT FK\_PedidoProducto\_tbVendedorProveedor

PRINT 'La relacion [FK\_PedidoProducto\_tbVendedorProveedor] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbPedidoProducto

ADD CONSTRAINT FK\_PedidoProducto\_tbVendedorProveedor

FOREIGN KEY (idVendedorProveedor) REFERENCES tbVendedorProveedor(idVendedorProveedor)

END

GO

--idTrabajador

IF OBJECT\_ID('FK\_PedidoProducto\_tbTrabajador') IS NOT NULL

BEGIN

ALTER TABLE tbPedidoProducto

DROP CONSTRAINT FK\_PedidoProducto\_tbTrabajador

PRINT 'La relacion [FK\_PedidoProducto\_tbTrabajador] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbPedidoProducto

ADD CONSTRAINT FK\_PedidoProducto\_tbTrabajador

FOREIGN KEY (idTrabajador) REFERENCES tbTrabajador(idTrabajador)

END

GO

--tbProveedor(PK) tbVendedorProveedor(FK)

IF OBJECT\_ID('FK\_tbVendedorProveedor\_tbProveedor') IS NOT NULL

BEGIN

ALTER TABLE tbVendedorProveedor

DROP CONSTRAINT FK\_tbVendedorProveedor\_tbProveedor

PRINT 'La relacion [FK\_tbVendedorProveedor\_tbProveedor] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbVendedorProveedor

ADD CONSTRAINT FK\_tbVendedorProveedor\_tbProveedor

FOREIGN KEY (idProveedor) REFERENCES tbProveedor(idProveedor)

END

GO

IF OBJECT\_ID('FK\_tbSuministra\_FK\_Pais') IS NOT NULL

BEGIN

ALTER TABLE tbSuministra

DROP CONSTRAINT FK\_tbSuministra\_FK\_Pais

PRINT 'La relacion [FK\_tbSuministra\_FK\_Pais] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbSuministra

ADD CONSTRAINT FK\_tbSuministra\_FK\_Pais

FOREIGN KEY(paisOrigen) REFERENCES tbPais(idPais)

END

GO

IF OBJECT\_ID('FK\_tbPersona\_FK\_Pais') IS NOT NULL

BEGIN

ALTER TABLE tbPersona

DROP CONSTRAINT FK\_tbPersona\_FK\_Pais

PRINT 'La relacion [FK\_tbPersona\_FK\_Pais] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbPersona

ADD CONSTRAINT FK\_tbPersona\_FK\_Pais

FOREIGN KEY(idPais) REFERENCES tbPais(idPais)

END

GO

IF OBJECT\_ID('FK\_tbSuministra\_FK\_Producto') IS NOT NULL

BEGIN

ALTER TABLE tbSuministra

DROP CONSTRAINT FK\_tbSuministra\_FK\_Producto

PRINT 'La relacion [FK\_tbSuministra\_FK\_Producto] fue eliminada'

END

BEGIN

ALTER TABLE tbSuministra

ADD CONSTRAINT FK\_tbSuministra\_FK\_Producto

FOREIGN KEY(idProducto) REFERENCES tbProducto(idProducto)

END

GO

IF OBJECT\_ID('FK\_tbSuministra\_FK\_Proveedor') IS NOT NULL

BEGIN

ALTER TABLE tbSuministra

DROP CONSTRAINT FK\_tbSuministra\_FK\_Proveedor

PRINT 'La relacion [FK\_tbSuministra\_FK\_Proveedor] fue eliminada'

END

BEGIN

ALTER TABLE tbSuministra

ADD CONSTRAINT FK\_tbSuministra\_FK\_Proveedor

FOREIGN KEY(idProveedor) REFERENCES tbProveedor(idProveedor)

END

GO

IF OBJECT\_ID('FK\_tbDespacha\_FK\_tbArea') IS NOT NULL

BEGIN

ALTER TABLE tbDespacha

DROP CONSTRAINT FK\_tbDespacha\_FK\_tbArea

PRINT 'La relacion [FK\_tbDespacha\_FK\_tbArea] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbDespacha

ADD CONSTRAINT FK\_tbDespacha\_FK\_tbArea

FOREIGN KEY(idArea) REFERENCES tbArea(idArea)

END

GO

IF OBJECT\_ID('FK\_tbDespacha\_FK\_tbProducto') IS NOT NULL

BEGIN

ALTER TABLE tbDespacha

DROP CONSTRAINT FK\_tbDespacha\_FK\_tbProducto

PRINT 'La relacion [FK\_tbDespacha\_FK\_tbProducto] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbDespacha

ADD CONSTRAINT FK\_tbDespacha\_FK\_tbProducto

FOREIGN KEY(idProducto) REFERENCES tbProducto(idProducto)

END

GO

IF OBJECT\_ID('FK\_tbDespacha\_FK\_tbTrabajador') IS NOT NULL

BEGIN

ALTER TABLE tbDespacha

DROP CONSTRAINT FK\_tbDespacha\_FK\_tbTrabajador

PRINT 'La relacion [FK\_tbDespacha\_FK\_tbTrabajador] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbDespacha

ADD CONSTRAINT FK\_tbDespacha\_FK\_tbTrabajador

FOREIGN KEY(idTrabajador) REFERENCES tbTrabajador(idTrabajador)

END

GO

IF OBJECT\_ID('FK\_tbKardex\_tbProducto') IS NOT NULL

BEGIN

ALTER TABLE tbKardex

DROP CONSTRAINT FK\_tbKardex\_tbProducto

END

ELSE

BEGIN

ALTER TABLE tbKardex

ADD CONSTRAINT FK\_tbKardex\_tbProducto

FOREIGN KEY(idProducto) REFERENCES tbProducto(idProducto)

END

GO

IF OBJECT\_ID('FK\_tbKardex\_tbSuministra') IS NOT NULL

BEGIN

ALTER TABLE tbKardex

DROP CONSTRAINT FK\_tbKardex\_tbSuministra

END

ELSE

BEGIN

ALTER TABLE tbKardex

ADD CONSTRAINT FK\_tbKardex\_tbSuministra

FOREIGN KEY(idSuministra) REFERENCES tbSuministra(idSuministra)

END

GO

IF OBJECT\_ID('FK\_tbKardex\_tbDespacha') IS NOT NULL

BEGIN

ALTER TABLE tbKardex

DROP CONSTRAINT FK\_tbKardex\_tbDespacha

END

ELSE

BEGIN

ALTER TABLE tbKardex

ADD CONSTRAINT FK\_tbKardex\_tbDespacha

FOREIGN KEY(idDespacha) REFERENCES tbDespacha(idDespacha)

END

GO

--tbTurno(PK) tbTrabajador(FK)

IF OBJECT\_ID('FK\_tbTrabajador\_tbTurno') IS NOT NULL

BEGIN

ALTER TABLE tbTrabajador

DROP CONSTRAINT FK\_tbTrabajador\_tbTurno

PRINT 'La relacion [FK\_tbTrabajador\_tbTurno] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbTrabajador

ADD CONSTRAINT FK\_tbTrabajador\_tbTurno

FOREIGN KEY (idTurno) REFERENCES tbTurno(idTurno)

END

GO

--tbProveedor(PK) tbVendedorProveedor(FK)

IF OBJECT\_ID('FK\_tbVendedorProveedor\_tbPersona') IS NOT NULL

BEGIN

ALTER TABLE tbVendedorProveedor

DROP CONSTRAINT FK\_tbVendedorProveedor\_tbPersona

PRINT 'La relacion [FK\_tbVendedorProveedor\_tbPersona] fue eliminada'

END

ELSE

BEGIN

ALTER TABLE tbVendedorProveedor

ADD CONSTRAINT FK\_tbVendedorProveedor\_tbPersona

FOREIGN KEY (idPersona) REFERENCES tbPersona(idPersona)

END

GO

-------------------------------------------------------------------

-------------------------------------------------------------------

-------------------------------------------------------------------

-- PROCEDIMIENTOS

--PAIS

CREATE PROCEDURE SP\_RegistraPais

@CodPais varchar(10),

@pais varchar(25)

AS

DECLARE @totalReg int

DECLARE @enlazado varchar(10)

SELECT @totalReg = COUNT(\*) FROM tbPais

SET @enlazado = @CodPais +CAST(@totalReg AS varchar(6))

IF EXISTS(SELECT \*FROM tbPais WHERE tbPais.idPais= @enlazado)

BEGIN

PRINT 'Id ya registrado por favor haga modificacion en letras...';

END

ELSE

BEGIN

INSERT INTO tbPais VALUES(@enlazado,@pais)

PRINT 'Los datos fueron registrados en la tabla [PAIS] con exito...!'

END

GO

EXECUTE SP\_RegistraPais 'RP','PERÚ'

EXECUTE SP\_RegistraPais 'RP','CHILE'

EXECUTE SP\_RegistraPais 'RP','BOLIVIA'

EXECUTE SP\_RegistraPais 'RP','VENEZUELA'

EXECUTE SP\_RegistraPais 'RP','ARGENTINA'

-------------------------------------------------------------------

-------------------------------------------------------------------

--PROCEDIMIENTO REGISTRAR PROVEEDORES

-------------------------------------------------------------------

-------------------------------------------------------------------

CREATE PROCEDURE SP\_RegistraProveedor

@codProveedor varchar(10),

@codPais varchar(10),

@razonSocial varchar(50),

@codigoPostal varchar(20),

@rucProveedor varchar(13),

@fonoProveedor varchar(12),

@emailProveedor varchar(50),

@direccion varchar(50)

AS

DECLARE @totalReg int

DECLARE @enlazado varchar(8)

SELECT @totalReg = COUNT(\*) FROM tbProveedor

SET @enlazado = @codProveedor +CAST(@totalReg AS varchar(6))

IF EXISTS(SELECT \*FROM tbProveedor WHERE tbProveedor.idProveedor= @enlazado)

BEGIN

PRINT 'Id ya registrado por favor haga modificacion en letras...';

END

ELSE

BEGIN

INSERT INTO tbProveedor VALUES(@enlazado,@codPais,@razonSocial,@codigoPostal,@rucProveedor,

@fonoProveedor,@emailProveedor,@direccion)

PRINT 'Los datos fueron registrados en la tabla [PROVEEDOR] con exito...!'

END

GO

select \* from tbProveedor

select \* from tbPais

EXECUTE SP\_RegistraProveedor 'PRO','RP0','JCR INGENIERIA INDUSTRIAL','91600','2062154856', '598-25115121', 'INFO@JCR.COM.UY', 'MATEO CORTE 4941 - CP 11400 MONTEVIDEO'

EXECUTE SP\_RegistraProveedor 'PRO','RP1','TROX DO BRASIL DIFUSAO DE','69900-000','2012452545', '55-413316-8418', 'OI\_CRISTINA@TROXBRASIL.COM.BR', 'RUA CYRO COREIA PEREIRA, 300-CIC'

EXECUTE SP\_RegistraProveedor 'PRO','RP2','UNION CARGO INTERNACIONAL','110911','2032565482', '(57) 1 4215832', 'GCOMERCIAL@UNIONCARGO.NET', 'CR 106 23 D-50 P-3 DISTRITO CAPITAL'

EXECUTE SP\_RegistraProveedor 'PRO','RP3','UNION SOUTH INTERNATIONAL','33126','524526548', '305 539-1057', 'INFO@JCR.COM.UY', '1948 NW 82ND AVE, DORAL, FL'

EXECUTE SP\_RegistraProveedor 'PRO','RP4','COSBERT SA','91600','2545254858', '59825182965', 'LUCIA.GOMEZ@LATAM.INTERFOOD.COM', 'RUTA 8, KM 17.500 EDIFICIO BIOTEC PLAZAX'

EXECUTE SP\_RegistraProveedor 'PRO','RP0','VERBEAI INGREDIENTS & PAI','56130','0033299912485', '598-25115121', 'PHILIPPE.LERAV@VINPAI.FR', '6 PARC DACTIVITÉS DE LA FOUÉE'

EXECUTE SP\_RegistraProveedor 'PRO','RP1','VISCOFAN CZ S.R.O.','370 03','10266589455', '34 948 198 444', 'LUIS@QUIMSAPERU.COM', 'XFCP+F2 ?ESKÉ BUD?JOVICE, CHEQUIA'

EXECUTE SP\_RegistraProveedor 'PRO','RP2','VISCOFAN DO BRASIL SOC. C','69900-000','202565455552', '(11) 5180-7400', 'LUIS@QUIMSAPERU.COM', 'AV. ROQUE PETRONI JUNIERO, - 1 ANDAR'

-------------------------------------------------------------------

-------------------------------------------------------------------

--PROCEDIMIENTO MOSTRAR AREAS EMPRESA

CREATE PROCEDURE sp\_ListarAreaCod

@codArea varchar(10)

AS

SELECT \* FROM tbArea WHERE tbArea.idArea = @codArea

GO

-------------------------------------------------------------------

-------------------------------------------------------------------

--PROCEDIMIENTO MOSTRAR PRODUCTOS

CREATE PROCEDURE sp\_ListarProductoCod

@codProducto varchar(10)

AS

SELECT \* FROM tbProducto WHERE tbProducto.idProducto = @codProducto

GO

--PROCEDIMIENTO REGISTRAR DESPACHA

CREATE PROCEDURE sp\_RegistrarDespacha

@codDespacha varchar(10),

@idTrabajador varchar(10),

@idProducto varchar(10),

@idArea varchar(10),

@cantidad int,

@lote varchar(10)

AS

DECLARE @totalReg int

DECLARE @enlazado varchar(8)

SELECT @totalReg = COUNT(\*) FROM tbDespacha

SET @enlazado = @codDespacha +CAST(@totalReg AS varchar(6))

IF EXISTS(SELECT \*FROM tbDespacha WHERE tbDespacha.idDespacha = @enlazado)

BEGIN

PRINT 'Id ya registrado por favor haga modificacion en letras...';

END

ELSE

BEGIN

INSERT INTO tbDespacha VALUES(@enlazado,@idTrabajador,@idProducto,@idArea,

@cantidad,@lote)

UPDATE tbProducto set tbProducto.stock = tbProducto.stock - @cantidad

WHERE tbProducto.idProducto=@idProducto

PRINT 'Los datos fueron registrados en la tabla [DESPACHO] con exito...!'

END

GO

--PROCEDIMIENTO ACTUALIZAR LA TABLA SUMINISTRO

CREATE PROCEDURE sp\_ActualizarSuministra

@codSuministra varchar(10),

@codProveedor varchar(10),

@codProducto varchar(10),

@lote varchar(10),

@cantidad int,

@numFactura varchar(10),

@paisOrigen varchar(10),

@precio\_compra float

AS

BEGIN

UPDATE tbSuministra

set tbSuministra.cantidad=@cantidad, tbSuministra.idProducto=@codProducto,

tbSuministra.lote=@lote, tbSuministra.idProveedor=@codProveedor, tbSuministra.numFactura= @numFactura,

tbSuministra.paisOrigen = @paisOrigen, tbSuministra.precio\_compra = @precio\_compra

WHERE tbSuministra.idSuministra = @codSuministra

UPDATE tbProducto set tbProducto.stock = tbProducto.stock + @cantidad WHERE tbProducto.idProducto=@codProducto

PRINT 'Los datos fueron ACTUALIZADOS en la tabla [SUMINISTRA] con exito...!'

END

GO

----------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

--PROCEDIMIENTO DE REGISTRO DE TIPO DE PRODUCTO

----------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

--EMBUTIDOS

--LACTEOS

CREATE PROCEDURE sp\_RegistrarTipoProducto

@codTipoProducto varchar(10),

@nomTipoProducto varchar(25)

AS

DECLARE @totalReg int

DECLARE @enlazado varchar(8)

SELECT @totalReg = COUNT(\*) FROM tbTipoProducto

SET @enlazado = @codTipoProducto +CAST(@totalReg AS varchar(6))

IF EXISTS(SELECT \*FROM tbTipoProducto WHERE tbTipoProducto.idTipoProducto = @enlazado)

BEGIN

PRINT 'Id ya registrado por favor haga modificacion en letras...';

END

ELSE

BEGIN

INSERT INTO tbTipoProducto VALUES(@enlazado,@nomTipoProducto)

PRINT 'Los datos fueron registrados en la tabla [TIPO PRODUCTO] con exito...!'

END

GO

EXECUTE sp\_RegistrarTipoProducto 'RTP','EMBUTIDOS'

EXECUTE sp\_RegistrarTipoProducto 'RTP','LACTEOS'

--------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

--PROCEDIMIENTO DE REGISTRO DE UBICACION DE PRODCUCTO

----------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

CREATE PROCEDURE SP\_RegistraUbicacionProducto

@Cod varchar(2),

@seccion varchar(25)

AS

DECLARE @totalReg int

DECLARE @enlazado varchar(8)

SELECT @totalReg = COUNT(\*) FROM tbUbicacion

SET @enlazado = @Cod +CAST(@totalReg AS varchar(6))

IF EXISTS(SELECT \*FROM tbUbicacion WHERE idUbicacion = @enlazado)

BEGIN

PRINT 'Id ya registrado por favor haga modificacion en letras...';

END

ELSE

BEGIN

INSERT INTO tbUbicacion VALUES(@enlazado,@seccion)--pañales,etc

PRINT 'Los datos fueron registrados con exito...!'

END

GO

EXECUTE SP\_RegistraUbicacionProducto 'RUP', 'UBI. LACTEOS'

--------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

--PROCEDIMIENTO DE REGISTRO DE UNIDAD DE MEDIDA

----------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

CREATE PROCEDURE SP\_RegistraUnidadMedida

@Cod varchar(2),

@UnidadMdedida varchar (15)

AS

DECLARE @totalReg int

DECLARE @enlazado varchar(8)

SELECT @totalReg = COUNT(\*) FROM tbUnidadMedida

SET @enlazado = @Cod +CAST(@totalReg AS varchar(6))

IF EXISTS(SELECT \*FROM tbUnidadMedida WHERE idUnidadMedida = @enlazado)

BEGIN

PRINT 'Id ya registrado por favor haga modificacion en letras...';

END

ELSE

BEGIN

INSERT INTO tbUnidadMedida VALUES(@enlazado,@UnidadMdedida)

PRINT 'Los datos fueron registrados con exito...!'

END

GO

EXECUTE SP\_RegistraUnidadMedida 'RUM', 'KILOS'

--------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------

CREATE PROCEDURE SP\_RegistraProducto

@codProducto varchar(10),

@idTipoProducto varchar(10),

@idUbicacion varchar(10),

@idUnidadMedida varchar(10),

@nombreProducto varchar(100),

@precioCompra float,

@descripcion varchar(350),

@stock int,

@caducidad date

AS

DECLARE @totalReg int

DECLARE @enlazado varchar(8)

SELECT @totalReg = COUNT(\*) FROM tbProducto

SET @enlazado = @codProducto +CAST(@totalReg AS varchar(6))

IF EXISTS(SELECT \*FROM tbProducto WHERE tbProducto.idProducto = @enlazado)

BEGIN

PRINT 'Id ya registrado por favor haga modificacion en letras...';

END

ELSE

BEGIN

INSERT INTO tbProducto VALUES(@codProducto,@idTipoProducto, @idUbicacion,@idUnidadMedida,@nombreProducto,@precioCompra,@descripcion,

@stock, @caducidad)

PRINT 'Los datos fueron registrados la tabla PRODUCTO con exito...!'

END

GO

EXECUTE SP\_RegistraProducto 'RP', 'RTP1', 'RU0', 'RU0','YOGURT GLORIA', 20.9,'BEBIDA NUTRITIVA CON FRESAS NATURALES',10, '07/04/2020'

------------------------------------------------------------------------------------------------------

------------------------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

--PROCEDIMIENTO DE REGISTRO SUMINISTRO

----------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

CREATE PROCEDURE SP\_RegistraSuministra

@codSuministra varchar(10),

@codProveedor varchar(10),

@codProducto varchar(10),

@lote varchar(10),

@cantidad int,

@numFactura varchar(10),

@paisOrigen varchar(10),

@precio\_compra float

AS

DECLARE @totalReg int

DECLARE @enlazado varchar(8)

SELECT @totalReg = COUNT(\*) FROM tbSuministra

SET @enlazado = @codSuministra +CAST(@totalReg AS varchar(6))

IF EXISTS(SELECT \*FROM tbSuministra WHERE tbSuministra.idSuministra = @enlazado)

BEGIN

PRINT 'Id ya registrado por favor haga modificacion en letras...';

END

ELSE

BEGIN

INSERT INTO tbSuministra VALUES(@enlazado,@codProveedor, @codProducto, @lote, @cantidad, @numFactura, @paisOrigen, @precio\_compra);

UPDATE tbProducto

set tbProducto.stock = tbProducto.stock + @cantidad

WHERE tbProducto.idProducto = @codProducto

PRINT 'Los datos fueron registrados en la tabla [SUMINISTRA] con exito...!'

END

GO

EXECUTE SP\_RegistraSuministra 'RS', 'PRO0010','RP', '635121', '4', '123456', 'RP4', 25.7

----------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

--PROCEDIMIENTO REGISTRO DE PROFESION

----------------------------------------------------------------------------------------

----------------------------------------------------------------------------------------

CREATE PROCEDURE SP\_RegistraProfesion

@Cod varchar(2),

@Profesion varchar(25),

@estadoProfesion bit

AS

DECLARE @totalReg int

DECLARE @enlazado varchar(8)

SELECT @totalReg = COUNT(\*) FROM tbProfesion

SET @enlazado = @Cod +CAST(@totalReg AS varchar(6))

IF EXISTS(SELECT \*FROM tbProfesion WHERE idProfesion = @enlazado)

BEGIN

PRINT 'Id ya registrado por favor haga modificacion en letras...';

END

ELSE

BEGIN

INSERT INTO tbProfesion VALUES(@enlazado,@Profesion,@estadoProfesion)-- MEDICO,ENFERMERO

PRINT 'Los datos fueron registrados con exito...!'

END

GO

EXECUTE SP\_RegistraProfesion 'RP','farmaceutico','1'

1. **REFERENCIAS**

* Elmasri R., Navathe B. (2007). Fundamentos de sistemas de bases de datos. Madrid
* Aguilar Joyanes L. Zahonero Martinez (2014)Programacion en C, C++, java y UML. Mexico