MODULO DE INVENTARIO

"ALARMAS Y RADIOS LOS SOATÁ"

ELSY ANDREA RODRIGUEZ CUEVAS

Trabajo de grado presentado para optar el titulo de:
Tecnóloga en informática.

CORPORACIÓN UNIVERSITARIA MINUTO DE DIOS

FACULTAD DE INGENIERIA

TECNOLOGÍA EN INFORMÁTICA

CENTRO REGIONAL SOACHA

2010

MODULO DE INVENTARIO

"ALARMAS Y RADIOS LOS SOATÁ"

ELSY ANDREA RODRIGUEZ CUEVAS

Trabajo de grado presentado para optar el titulo de:
Tecnóloga en informática.

Mauricio Bermúdez

Asesor Metodológico

CORPORACIÓN UNIVERSITARIA MINUTO DE DIOS

FACULTAD DE INGENIERIA

TECNOLOGÍA EN INFORMÁTICA

CENTRO REGIONAL SOACHA

2010

TABLA DE CONTENIDO

INTRODUCCIÓN....4

1. TEMA....5

2. ANÁLISIS DEL PROBLEMA....6

3. OBJETIVOS....7

3.1. OBJETIVO GENERAL....7

3.2. OBJETIVOS ESPECIFICOS....7

4. MARCO REFERENCIAL....9

4.1. MARCO TEORICO....9

4.1.1. Manejo de los datos....9

4.1.2. Almacenamiento de datos....12

4.2. MARCO CONCEPTUAL....13

4.3. MARCO ORGANIZACIONAL....14

5. ALCANCE....15

6. SISTEMA ACTUAL....16

6.1. DIAGRAMA CASO DE USO....16

6.2. DIAGRAMA SECUENCIAL....16

7. SISTEMA PROPUESTO....17

7.1. DIAGRAMA DE CASO DE USU....17

7.2. DIAGRAMA SECUENCIAL....17

8. CRONOGRAMA DE ACTIVIDADES....18

9. ARQUITECTURA DE LA SOLUCION DE SOFTWARE....19

9.1. ENTIDAD RELACION BASE DE DATOS....19

10. ANALISI DE LA SOLUCION DE SOFTWARE....20

10.1. CESTRUCTURA DE ALMACENAMIENTO....20

10.2. INTERFAZ GRAFICA DE USUARIO....20

10.3. INTEFAZ DE PROCESAMIENTO DE DATOS....20

11. DISEÑO DE LA SOLUCION DE SOFTWARE....22

11.1. ESTRUCTURA DE ALMACENAMIENTO DE DATOS....22

11.2. INTERFAZ GRAFICA DE USUARIO 29
11.3. INTERFAZ DE PROCESAMIENTO DE DATOS 38
12. DESARROLLO Y PRUEBAS 56
13. CONCLUSIONES 57
13. RECOMENDACIONES 58
- REFERENCIAS 59

INTRODUCCION

El avance de las tecnologías de información hace que el usuario más sencillo y sin conocimientos requiera de ellas, con el fin de realizar de manera metodológica y ordenada la administración de sus negocios; es por esto que a partir de una necesidad básica sacamos el provecho para desarrollar una aplicación de software que cumpla con los requerimientos solicitados por el usuario.

En este caso se mostrara las características básicas para el desarrollo de un modulo de inventario que permitirá llevar la administración y control del mismo para el almacén “ALARMAS Y RADIOS SOATÁ”.

En este proyecto se mostrara la metodología que se llevo a cavo en la realización del modulo de compras sobre el desarrollo y las características del análisis, diseño e implementación del mismo.

1. TEMA

Por medio de los conocimientos adquiridos durante la preparación recibida de la universidad Minuto de Dios se realizo como proyecto la creación de un prototipo de un sistema de información para el manejo oportuno y veraz de la administración del almacén “ALARMAS Y RADIOS SOATÁ” donde inicialmente se entregara funcional el modulo de compras para la administración del inventario.

Se creara este sistema de información para satisfacer las necesidades actuales del almacén y a través del desarrollo total de los módulos del inventario dejar funcional el sistema para la administración oportuna del almacén.

2. ANÁLISIS DEL PROBLEMA

Una sociedad familiar está poniendo en marcha la creación de un almacén que se dedicara a la distribución de productos para autos; ya se cuenta con el capital y el lugar donde se llevara a cabo dicho proyecto; sus propietarios desean implementar una aplicación de software pequeña que les permita administrar los ingresos y egresos del almacén ya que ven la necesidad de manejar de forma organizada los datos referentes a facturación tanto de compras como de ventas, es necesario manejar un inventario eficaz de los productos comercializados en el almacén, se desea también que a partir de los datos almacenados les permita analizar las ganancias para la toma oportuna de decisiones referentes al almacén.

Por ahora la gestión se realiza registrando los datos en un cuaderno contable manualmente por los administradores del almacén, esto hace que la consulta de existencias de los artículos comercializados sea lenta y de igual manera ocurre en el momento de calcular las ganancias del mes.

Es por esto que surge implementar una aplicación que les permita realizar esta gestión de manera sencilla y rápida.

3. OBJETIVOS

3.1 Objetivo general

Crear una solución de software a la medida que permita a los usuarios del almacén “ALARMAS Y RADIOS SOATÁ” la administración organizada de sus facturas, inventario y gestión de ingresos y egresos relacionados con la venta de partes para autos a través de una forma útil, ágil y practica para facilitar la consulta de datos y la toma de decisiones.

3.2 Objetivos específicos

- Identificar las necesidades de los usuarios referentes a la solución de software que desean se realice para ingresar, modificar y consultar de forma ordenada los datos referentes a la administración de ingresos y egresos del almacén.

- Analizar todas las características deseadas y necesarias para el desarrollo de la aplicación de software que permitan satisfacer de manera adecuada los requerimientos de los usuarios para el manejo de datos del almacén y toma de decisiones.

- Planear una estrategia de trabajo que nos permita estimar el tiempo que será necesario para el desarrollo del modulo de administración de ingresos y egresos del almacén “ALARMAS Y RADIOS SOATÁ” y así ejercer un control sobre el avance del mismo.

- Ingeniar un diseño práctico donde se mostrara la estructura, funciones e interacciones de los elementos del sistema referente a la aplicación de software que se desea implementar teniendo en cuenta los requerimientos y recursos existentes a partir de la necesidad de los usuarios del almacén.

- Poner en marcha la construcción y elaboración de los distintos elementos del sistema prototipo del modulo de inicial de compras del almacén teniendo en cuenta los requerimientos de los usuarios, el análisis respectivo de los datos involucrados y el diseño básico del sistema.

- Implantar el diseño del modulo de administración construido para sus respectivas pruebas y de esta forma efectuar su correcta ejecución.

- Dejar funcional el modulo de compras inicialmente, con sus respectivos documentos que harían complemento como guía de uso del mismo.

4. MARCO REFERENCIAL

4.1 Marco Teórico

Modelo de Construcción de Prototipos

El paradigma de construcción de prototipos comienza con la recolección de requisitos. El desarrollador y el cliente encuentran y definen los objetivos globales para el software, identifican los requisitos conocidos y las áreas del esquema en donde es obligatoria más definición. Entonces aparece un «diseño rápido». El diseño rápido se centra en una representación de esos aspectos del software que serán visibles para el usuario/cliente (por ejemplo: enfoques de entrada y formatos de salida). El diseño rápido lleva a la construcción de un prototipo. El prototipo lo evalúa el cliente/usuario y se utiliza para refinar los requisitos del software a desarrollar. La iteración ocurre cuando el prototipo se pone a punto para satisfacer las necesidades del cliente, permitiendo al mismo tiempo que el desarrollador comprenda mejor lo que se necesita hacer. (Pressman, 2002, pág. 23)

4.1.1 Manejo de los datos

Para la realización de este proyecto tendremos en cuenta los conceptos concernientes a la programación orientada a objetos ya que nos permite realizar una mejor encapsulación de los datos para garantizar la seguridad de los mismos ya que a medida que pasa el tiempo se debe implementar mayor seguridad dentro de los sistemas de información.

Cuando se trabaja la programación orientada a objetos se realiza a través de clases donde cada una de ellas lleva las características de los objetos que intervienen en el sistema y a su vez estas características pueden ser heredadas esto con el fin de una manipulación de datos más efectivo y evitar la redundancia de datos.

Dentro de la orientación a objetos se identifican una serie de conceptos o ideas como los atributos y las operaciones, quienes conforman una clase.

Según Pressulan una clase se define como sigue (2002, 346)

> Una clase es un concepto OO que encapsula las abstracciones de datos y procedimientos que se requieren para describir el contenido y el comportamiento de alguna entidad del mundo real.

Una segunda descripción de clase corresponde a una generalización o colección de objetos similares los cuales existen dentro de la misma (IBID). Un objeto al ser considerado un ejemplo del mundo real también recibe el nombre de miembro o instancia (IBID, 345).

Otros de los conceptos de orientación a objetos se refiere al encapsulamiento el cual consiste en el empaquetamiento de la información bajo un nombre y así reutilizarse como un componente del programa (op.cit).

Un concepto más ajustado a clases puede ser (IBID ,346):

Puesto de otra manera, una clase es una descripción generalizada (por ejemplo, una platilla, un patrón o un prototipo) que describe una colección de objetos similares. Por definición, todos los objetos que existen dentro de una clase heredan sus atributos y las operaciones disponibles para la manipulación de los atributos. Una superclase es una colección de clases y una subclase es una instancia de una clase.

El encapsular los datos y las operaciones trabajando en un único paquete proporciona beneficios como:

- Ocultación de datos al mundo exterior, para reducir la propagación de efectos colaterales cuando se realizan cambios.
- La clase al ser una entidad sencilla permite manipular datos y operaciones facilitando la reutilización de componentes.
- El encapsulamiento de datos permite simplificar la interacción y el acoplamiento del sistema para reducir los procesos. (IBID, 348).

Los elementos de un sistema de información inicialmente

Para identificar los elementos de un modelo de objetos se debe realizar un análisis sintáctico gramatical en la narrativa del sistema que se va a construir. (IBID, 350).

Los objetos pueden ser mostrados como:

- Entidades externas: Por ejemplo: otros sistemas, dispositivos o personas.
- Cosas: Por ejemplo: Informes o presentaciones, etc.
- Ocurrencias o sucesos que pueden ocurrir dentro del contexto de una operación en el sistema.
- Unidades organizacionales: divisiones o grupos relevantes en una aplicación.
- Lugares que pueden establecer un contexto del problema y la función general del sistema.
- Estructuras que definen las clases y los objetos y sus relaciones.

(IBID, 351)

Estos objetos se determinan subrayando cada nombre o cláusula nominal e introduciéndola en una tabla simple destacando los sinónimos de sus atributos. (op.cit).

Teniendo en cuenta los objetos son ejemplos de una clase cualquiera, la acción de crear un objeto a partir de una clase se llama instanciar. Para la creación de un objeto se tiene que describir una instrucción especial que puede ser distinta dependiendo al lenguaje de programación, como por ejemplo

Micoche= newCoche() … Con la palabra new especificamos crear una instancia (IBID, desarrolloweb.com)

http://www.desarrolloweb.com/articulos/499.php

4.1.2 Almacenamiento de datos

Para realizar el almacenamiento de los datos se utilizara el gestor de base de datos MYSQL el cual está definido como:

"SQL, Structure Query Language (Lenguaje de Consulta Estructurado) es un lenguaje de programación para trabajar con base de datos relacionales como MySQL, Oracle, etc.

MySQL es un interpretador de SQL, es un servidor de base de datos.

MySQL permite crear base de datos y tablas, insertar datos, modificarlos, eliminarlos, ordenarlos, hacer consultas y realizar muchas operaciones, etc., resumiendo: administrar bases de datos.

Ingresando instrucciones en la línea de comandos o embebidas en un lenguaje como PHP nos comunicamos con el servidor. Cada sentencia debe acabar con punto y coma (;).

La sensibilidad a mayúsculas y minúsculas, es decir, si hace diferencia entre ellas, depende del sistema operativo, Windows no es sensible, pero Linux sí. Por ejemplo Windows interpreta igualmente las siguientes sentencias:

```
create database administracion;
Create DataBase administracion;
```

Pero Linux interpretará como un error la segunda.

Se recomienda usar siempre minúsculas. Es más el sitio mysqlya.com.ar está instalado sobre un servidor Linux por lo que todos los ejercicios deberán respetarse mayúsculas y minúsculas" (MYSQL YA).

El trabajar el almacenamiento de los datos en forma organizada nos permitirá evitar redundancia de datos donde se debe implementar integridad referencial con las cuales se determinan las reglas para relacionar dichos datos.

En MYSQL podemos referenciar unos datos con otro a través de llaves primarias o llaves foráneas según sea el caso.

4.2 Marco Conceptual

Arquitectura de capas: es un método de programación que permite separar el proceso de la siguiente manera:

Almacenamiento: Es el análisis y el diseño de la base de datos.

Presentación: Interfaz grafica del usuario.

Negocio: Capa de procesamiento de datos.

Elemento u objeto: son los datos que intervienen en el sistema.

Operaciones y relaciones: Son los procesos entre los datos que se llevaran a cabo dentro del sistema de información.

Objetos de procesamiento: son aquellos que permiten el tratamiento de datos como pueden ser capturar, modificar, consultar y eliminar de los mismos.

Información: es el resultado de las operaciones realizadas y mostradas al final de un proceso.

Atributos: son las características de las variables.

Comportamientos: propiedades de los métodos.

Métodos: conjunto de instrucciones con un objetivo definido.

Herencia: derivación de las características y propiedades de una clase en otras clases. Permite transferir atributos y métodos.

Instanciar: es obtener un objeto y colocarlo dentro de un contenedor.

4.3. MARCO ORGANIZACIONAL

El almacén fue creado a mediados del mes de febrero de 2010 con la razón social "ALARMAS Y RADIOS SOATÁ", sociedad familiar que se encargara de vender e instalar partes para autos como radios, alarmas eleva vidrios, etc.…

La organización cuenta de 6 socios quienes son los administradores del sistema y quienes interfieren directamente con el funcionamiento del mismo.

Es por esta razón que todos tendrán acceso al sistema de información y serán los directos involucrados con el aplicativo.

5. ALCANCE

El modulo de inventario a desarrollar para la gestión de ingresos y egresos del almacén “ALARMAS Y RADIOS SOATÁ” será una estructura centralizada, desarrollada in-situ en primer instancia, para manejarla posteriormente bajo ambiente cliente-servidor.
Se construirá bajo la plataforma de desarrollo JAVA con la herramienta NetBeans y servidor de aplicaciones Apache TOMCAT, en un paradigma Orientado a Objetos.

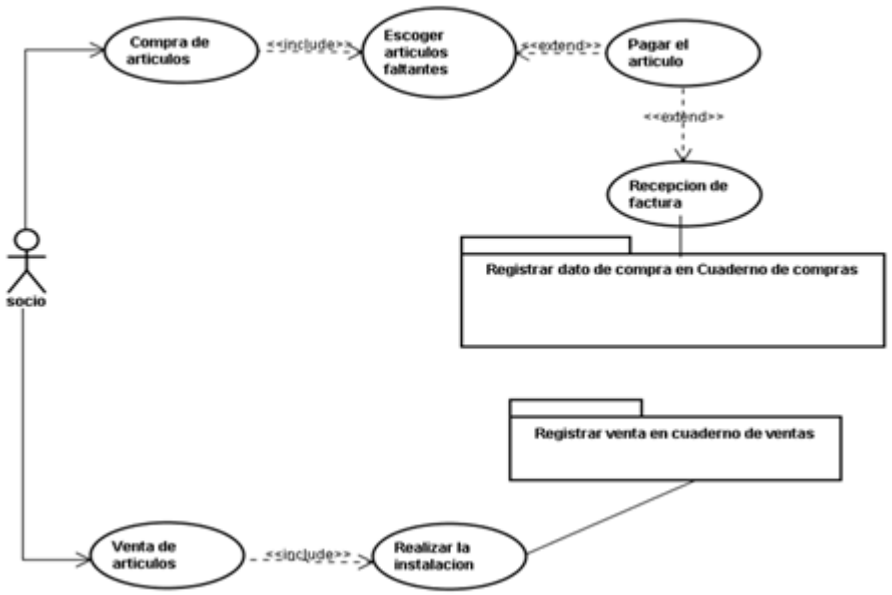
Manejara arquitectura de capas ya que el aplicativo trabajara un entorno WEB que tendrá:

· Almacenamiento (Análisis y diseño de la BD).
· Negocio (Interfaz de procesamiento de datos).
· Presentación (Interfaz grafica del usuario).

Se entregara funcional todo el modulo de compras del inventario donde se manejara le ingreso de los artículos y actualizara sus existencias también al realizar el desarrollo total del proyecto incluirá la administración de ingresos y egresos para un manejo eficaz del almacén y la toma de decisiones.

6. SISTEMA ACTUAL COMPRAS

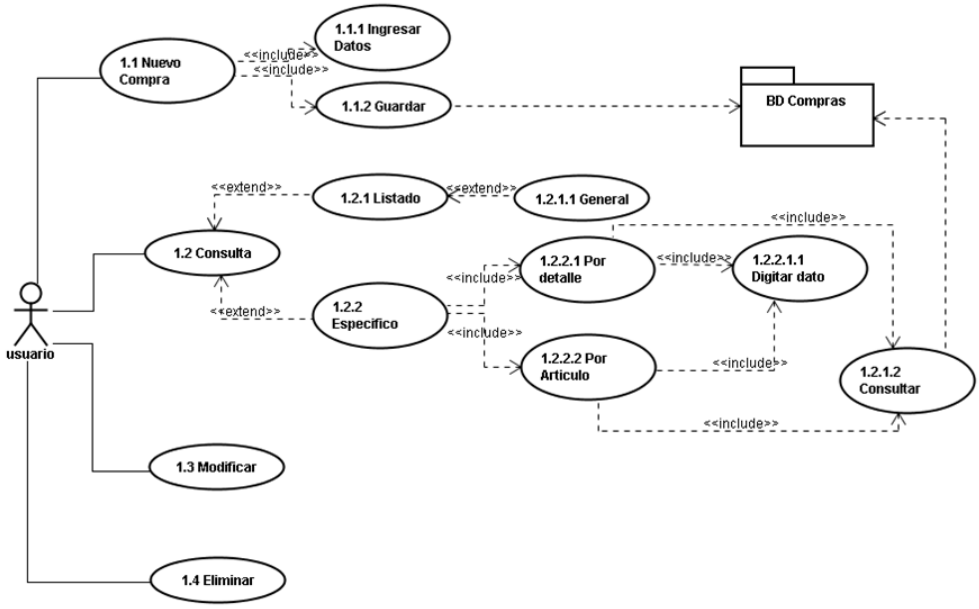
6.1. Diagrama caso de uso



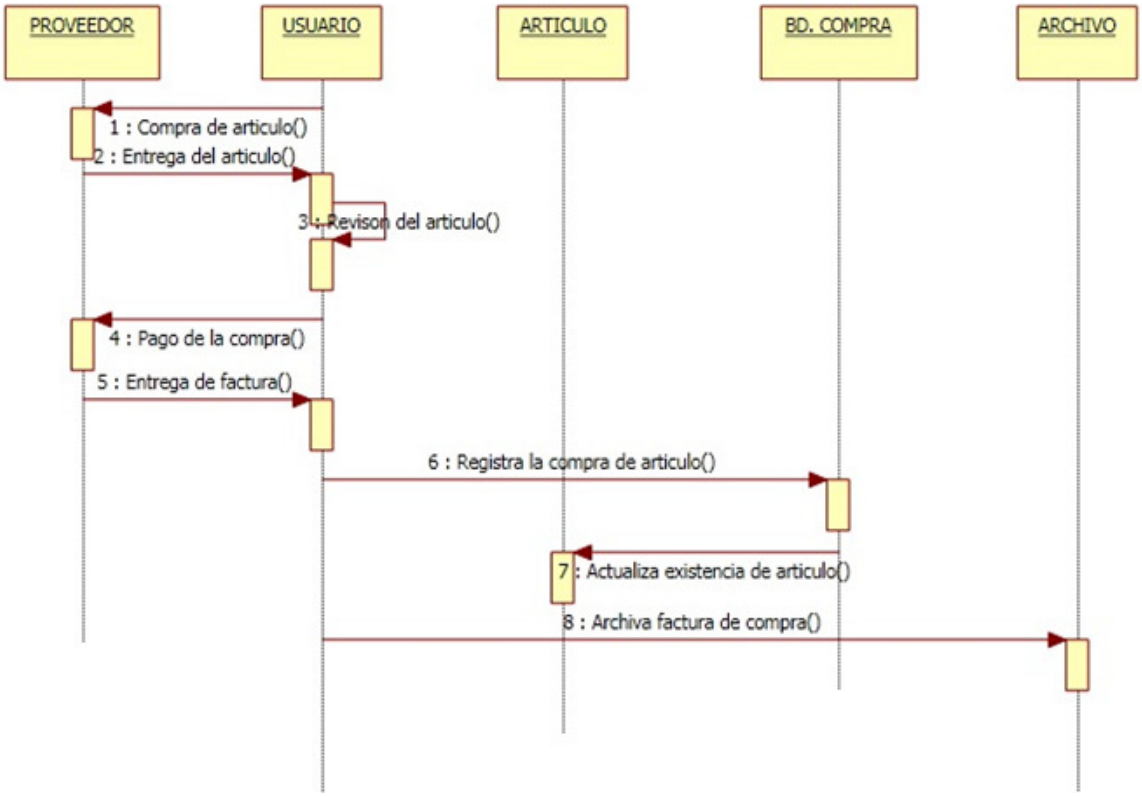
6.2. Diagrama Secuencial

7. SISTEMA PROPUESTO

7.1. Diagrama de caso de uso

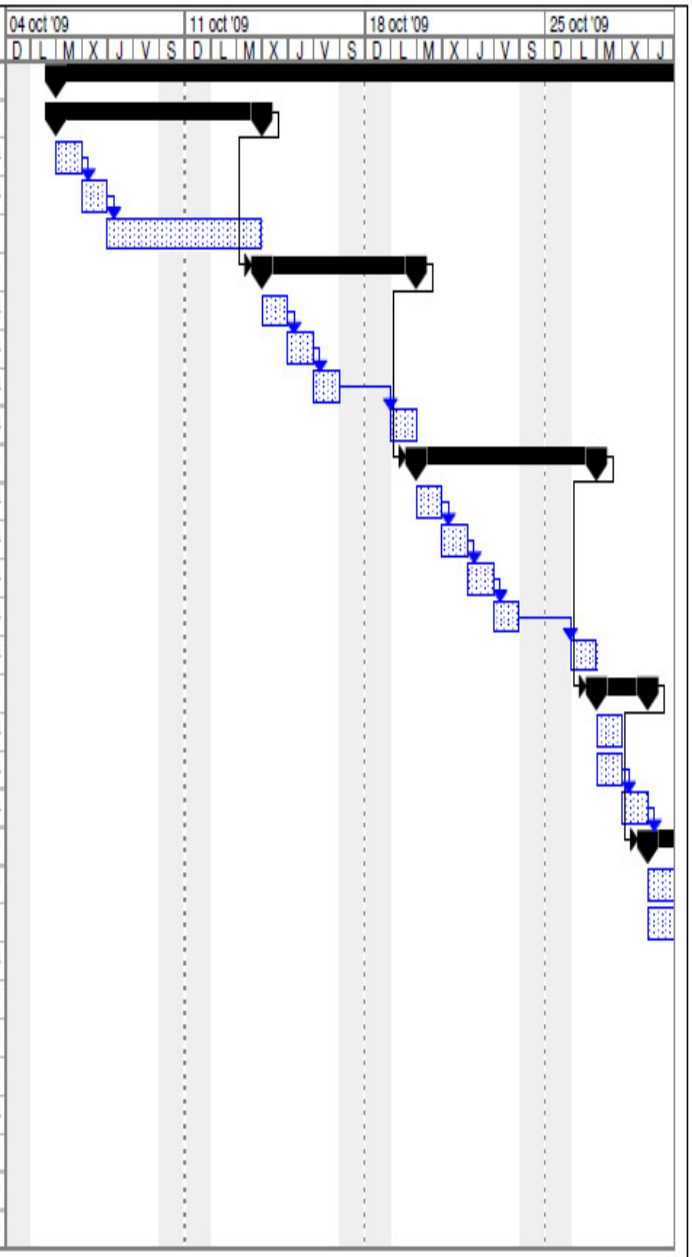


7.2. Diagrama secuencial



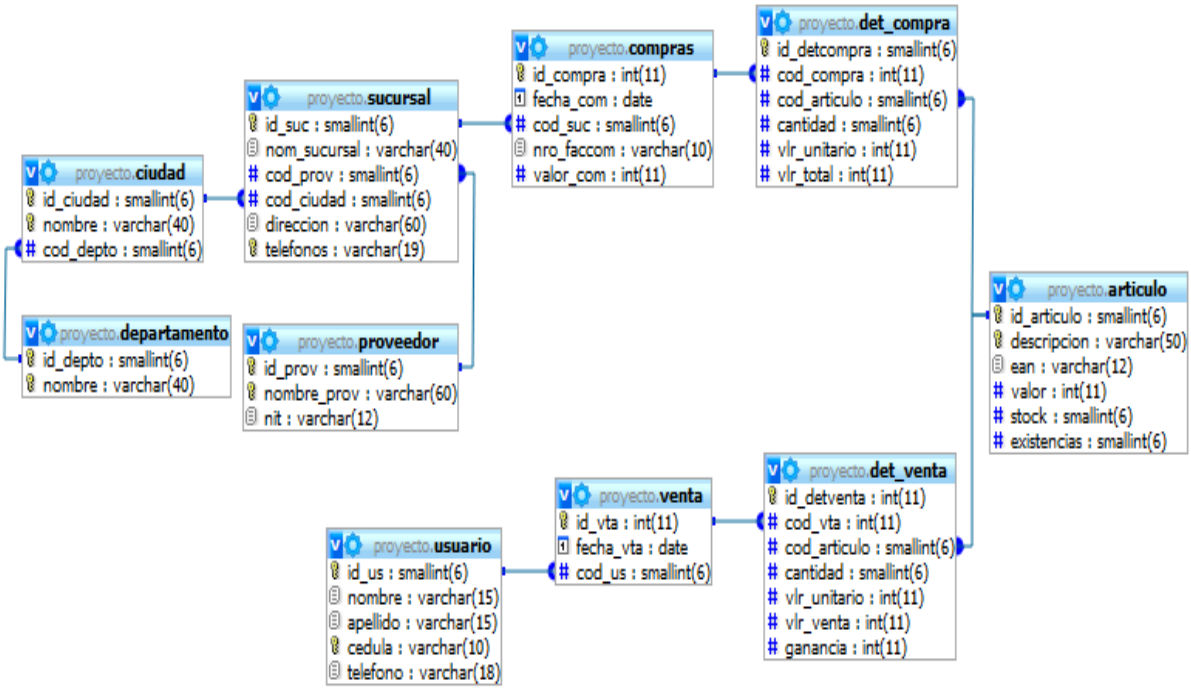
8. CRONOGRAMA DE ACTIVIDADES

Id	Nombre de tarea	Duración
1	**PROYECTO DE SOFTWARE**	**52 días**
2	**Comunicación - Analisis**	**6 días**
3	Analisis del problema	1 día
4	Objetivos	1 día
5	Planeacion	4 días
6	**Actividades**	**4 días**
7	Ambito del Software.	1 día
8	Formulación de estrategias.	1 día
9	Determinación de recursos.	1 día
10	Establecimiento del plan de trabajo.	1 día
11	**Estimacion del proyecto de software**	**5 días**
12	Determinación de Requerimientos	1 día
13	Definición de la Alternativa de Solución	1 día
14	Descripción del Sistema Propuesto	1 día
15	Estudio de Viabilidad	1 día
16	Control	1 día
17	**Modelado**	**2 días**
18	Diagramas UML	1 día
19	Diagramas de uso	1 día
20	Diagramas de actividades	1 día
21	**Marco referencial**	**3 días**
22	Marco teorico	3 días
23	Marco organizacional	3 días
24	Alcance	1 día
25	Sistema propuesto	5 días
26	Arquitectura de la solucion de software	2 días
27	**Diseño de la solucion de software**	**6 días**
28	Estructura de almacenamiento	1 día
29	Interfaz grafica	2 días
30	Interfaz de procesamientos de datos	3 días
31	Desarrollo	15 días



9. ARQUITECTURA DE LA SOLUCION DE SOFTWARE

9.1 Entidad Relación Base de datos



10. ANALISIS DE LA SOLUCION DE SOFTWARE

10.1 ESTRUCTURA DE ALMACENAMIENTO

La funcionalidad del sistema debe ser de almacenamiento de datos en MYSQL que permita a través de métodos ágiles de consulta la administración del almacén y un manejo oportuno y veraz del inventario de los productos comercializados.

10.2 INTERFAZ GRÁFICA DE USUARIO

Los requerimientos deben ser básicos al momento del uso del software ya que tenemos como restricción el conocimiento que tiene el usuario sobre el uso de herramientas informáticas y además el costo del proyecto debe ser bajo ya que a nivel de software y hardware se tiene presupuesto bajo y se requiere algo sencillo sin necesidad de licenciamiento.

Por esta razón la interfaz grafica será en un ambiente WEB ya que por estos tiempos el usuario esta mas familiarizado con este entorno.

10.3 INTERFAZ DE PROCESAMIENTO DE DATOS

Se desea realizar una aplicación que maneje un entorno de fácil acceso para los usuarios del sistema, que permita la administración de los datos de una forma rápida, segura, que el ambiente sea familiar y novedoso para su uso.

Por esta razón se construirá bajo la plataforma de desarrollo JAVA con la herramienta NetBeans y con servidor de aplicaciones Apache TOMCAT, en un paradigama orientado a objetos

Formulación de estrategias

El modelo a implementar en este desarrollo será el de un modulo inicialmente de el de “compras” que será entregado como prototipo ya que a través del desarrollo de esta aplicación permitirá presentar un avance del proyecto donde se mostrara un diseño básico al usuario final quien dará sus opiniones y por medio de estas nos permitiremos realizar arreglos al sistema, además nos permitirá continuar construyendo las otras partes del modulo de administración del almacén “ALARMAS Y RADIOS SOATÁ” con el fin de disminuir las posibles fallas en su ejecución final.

Se creara un sistema de información innovador que permita la interacción fácil entre el usuario administrador del sistema y el computador para agilizar, mejorar el manejo de la información y el procesamiento de datos.

Se analizaran los datos necesarios que se manejan en la administración del almacén para ser guardados y procesados en un una base de datos con un ambiente HTML para una mejor organización y procesamiento de datos con el fin de no dificultar la tarea del usuario final.

11. DISEÑO DE LA SOLUCION DE SOFTWARE

11.1 ESTRUCTURA DE ALMACENAMIENTO

DICCIONARIO DE DATOS

Nombre de la tabla: articulo
Descripción: Tabla que almacena los datos referentes a los artículos comercializados por el almacén.

Campo	Tipo	Longitud	Clave	Unicidad	Obligatoriedad	Indexado	Descripción
id_articulo	smallint	6	SI	SI	SI	SI	Campo que es utilizado como llave primaria para la identificación del artículo el cual es incremental.
Descripción	varchar	50	No	SI	SI	No	Campo que almacena el nombre del artículo,
ean	varchar	12		Si	No		Campo que almacena el código de barras de cada uno de los artículos.
valor	Int	11	No	No	SI	No	Campo que almacena el valor real del articulo
stock	smallint	6	No	No	SI	No	Campo que almacena la cantidad mínima que debe haber por cada artículo.
existencias	smallint	6	No	SI	SI	No	Campo que lleva el registro de las cantidades que hay en el almacén de los artículos.

Nombre de la tabla: Ciudad
Descripción: Tabla que almacena los datos de algunas ciudades de Colombia.

Campo	Tipo	Longitud	Clave	Unicidad	Obligatoriedad	Indexado	Descripción
id ciudad	Smallint	6	SI	SI	SI	SI	Campo que es utilizado como llave primaria para la identificación de las ciudades el cual se autoincrementa.
nom_ciudad	Varchar	40	No	SI	SI	No	Campo que almacena el nombre de la ciudad.
cod_depto	smallint	6	No	No	SI	SI	Campo que hace referencia a los datos del departamento del cual pertenece la ciudad.

Nombre de la tabla: Departamento.
Descripción: Tabla que almacena los datos de algunos departamentos de Colombia.

Campo	Tipo	Longitud	Clave	Unicidad	Obligatoriedad	Indexado	Descripción
id_depto	Smallint	6	SI	SI	SI	SI	Campo que es utilizado como llave primaria para la identificación de los departamentos la cual se autoincrementa.
nom_depto	Varchar	4o	No	SI	SI	No	Campo que almacena el nombre que identifica a cada departamento.

Nombre de la tabla: Proveedor.
Descripción: Tabla que almacena los datos básicos de cada uno de los proveedores que suministran los artículos para la venta en el almacén.

Campo	Tipo	Longitud	Clave	Unicidad	Obligatoriedad	Indexado	Descripción
id_prov	smallint	6	SI	SI	SI	SI	Campo que es utilizado como llave primaria para la identificación de los proveedores del almacén y se autoincrementa.
nom_prov	varchar	60	No	SI	SI	No	Campo que almacena la razón social de los proveedores.
nit	varchar	12	No	SI	No	No	Campo que almacena la identificación de cada uno de los proveedores.

Nombre de la tabla: Sucursal
Descripción: Tabla que almacena los datos de las direcciones y teléfonos de cada una de las sedes de los proveedores.

Campo	Tipo	Longitud	Clave	Unicidad	Obligatoriedad	Indexado	Descripción
id_suc	smallint	6	SI	SI	SI	SI	Campo que es utilizado como llave primaria para la identificación de cada una de las sedes donde están ubicados los proveedores del almacén.
nom_sucursal	varchar	40	No	No	Si	No	Campo que identifica el nombre de las sucursales correspondientes a cada uno de los proveedores.
cod_prov	smallint	6	No	No	Si	Si	Campo que hace referencia a la identificación del proveedor y sus datos básicos.
cod_ciudad	smallint	6	No	No	SI	Si	Campo que hace referencia a la ciudad que pertenece el proveedor.
direccion	varchar	60	No	SI	Si	No	Campo que almacena la dirección de los proveedores.
telefono	varchar	19	No	SI	SI	No	Campo que almacena los números telefónicos de los proveedores.

Nombre de la tabla: Compras.
Descripción: Tabla que almacena los datos básicos de cada una de las compras realizadas de los artículos comercializados por el almacén.

Campo	Tipo	Longitud	Clave	Unicidad	Obligatoriedad	Indexado	Descripción
id_compra	int	11	SI	SI	SI	SI	Campo que es utilizado como llave primaria que autoincrementa para la identificación de cada una de las compras realizadas por el almacén.
fecha_comp	date	-	No	No	SI	No	Campo que almacena la fecha de la compra realizada por el almacén.
cod_suc	smallint	6	No	No	SI	SI	Campo que hace referencia a la sede del proveedor al cual se le realizo la compra.
nro_faccom	varchar	10	No	Si	SI	No	Campo que almacena el número de factura con el cual se realizo la compra.
valor_com	int	11	No	No	SI	No	Campo que almacena el valor total de la compra realizada.

Nombre de la tabla: Ventas.
Descripción: Tabla que almacena los datos de las ventas realizadas por el almacén.

Campo	Tipo	Longitud	Clave	Unicidad	Obligatoriedad	Indexado	Descripción
id_vta	int	11	SI	SI	SI	SI	Campo que es utilizado como llave primaria que autoincrementa para la identificación de cada una de las ventas realizadas por el almacén.
fecha	date	-	No	No	SI	No	Campo almacena la fecha de cada una de las ventas.
cod_us	smallint	6	No	No	SI	SI	Campo que hace referencia al código del usuario que realiza la venta.

Nombre de la tabla: det_compra.
Descripción: Tabla que almacena los detalles de cada una de las compras realizadas por el almacén.

Campo	Tipo	Longitud	Clave	Unicidad	Obligatoriedad	Indexado	Descripción
id_detcompra	int	11	SI	SI	SI	SI	Campo que es utilizado como llave primaria que autoincrementa para la identificación de cada uno de los detalles de las compras realizadas por el almacén.
cod_compra	smallint	6	No	No	SI	SI	Campo que hace referencia al código de compra realizada por el almacén.
cod_articulo	smallint	6	No	No	SI	SI	Campo que hace referencia al los datos del articulo comprado.
cantidad	int	11	No	No	SI	No	Campo que almacena la cantidad de artículos comprados por cada detalle.
vlr_unitario	int	11	No	No	SI	No	Campo que almacena el valor de compra de cada uno de los artículos.
vlr_total	int	11	No	No	SI	No	Campo que efectúa la operación del valor total de la compra del artículo según la cantidad.

Nombre de la tabla: det_venta.
Descripción: tabla que almacena los detalles de cada una de las ventas realizadas por el almacén.

Campo	Tipo	Longitud	Clave	Unicidad	Obligatoriedad	Indexado	Descripción
id_detventa	int	11	SI	SI	SI	SI	Campo que es utilizado como llave primaria que autoincrementa para la identificación de cada uno de los detalles de las ventas realizadas por el almacén.
cod_vta	int	11	No	No	SI	SI	Campo que hace referencia al código de venta realizado por el almacén.
cod_articulo	smallint	6	No	No	SI	SI	Campo que hace referencia a la identificación del artículo vendido.
cantidad	smallint	6	No	No	SI	No	Campo que almacena el número de los artículos vendidos.
vlr_unitario	int	11	No	No	SI	No	Campo que almacena el valor real de venta del artículo.
vlr_venta	int	11	No	No	SI	No	Campo que efectúa la operación del valor total de la venta del artículo según la cantidad.
ganancia	int	11	No	No	SI	No	Campo que almacena la ganancia de la venta.

Nombre de la tabla: usuario.
Descripción: Tabla que almacena los datos básicos de los usuarios que interactuaran con el sistema y además quienes realizaran ventas en el almacén.

Campo	Tipo	Longitud	Clave	Unicidad	Obligatoriedad	Indexado	Descripción
id_us	smallint	6	SI	SI	SI	SI	Campo que es utilizado como llave primaria que autoincrementa para la identificación de cada uno de los usuarios del sistema.
nombre	varchar	15	No	No	SI	No	Campo que almacena el nombre de los usuarios que interfieren en la administración del almacén.
Apellido	varchar	15	No	No	Si	No	Campo que almacena el Apellido de los usuarios que interfieren en la administración del almacén.
cc	varchar	10	No	SI	SI	No	Campo que almacena la identificación de cada uno de los usuarios.
dir_us	varchar	22	No	SI	No	No	Campo que almacena la dirección de la casa de cada uno de los usuarios.
tel_us	varchar	18	No	Si	SI	No	Campo que almacena los números telefónicos de los usuarios.

11.2 Interfaz Gráfica De Usuario

INGRESO A LA APLICACIÓN

Pantalla principal

Esta grafica la pantalla principal del programa…

Descripción Del Modulo De Compras

Página de inicio

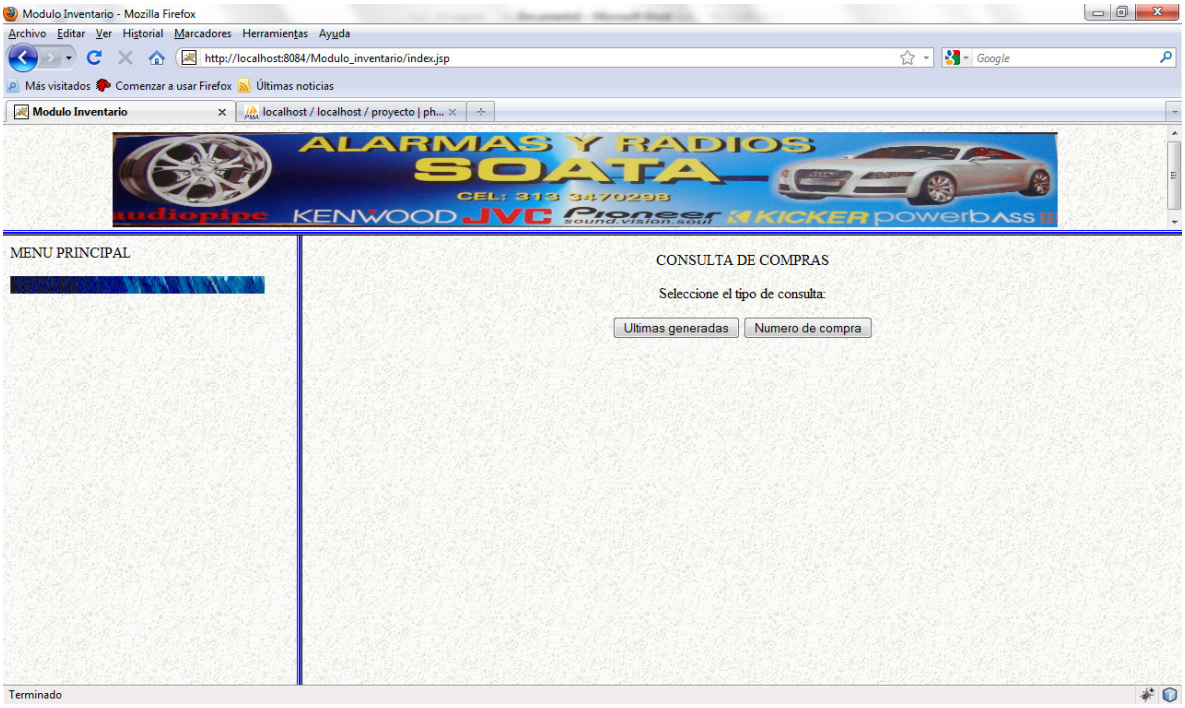
La solución de software tiene como funcional el modulo de compras donde se puede ingresar a través del menú inicial.

Ingresar compras

En la opción de ingreso compras encontrara en el formulario listas desplegables donde se seleccionaran datos como Proveedor, sucursal y articulo…

También se debe ingresar como la fecha de compra y numero de factura de compra y la cantidad de los artículos comprados… El resto de campos se diligenciaran automáticamente como el valor de los artículos y el valor total de la compra. Todos los datos son requeridos para ser almacenados correctamente al sistema.

Consultar Compras



En esta imagen se muestra que se puede realizar dos tipos de consulta como son:

Ultimas generadas o por número de compra

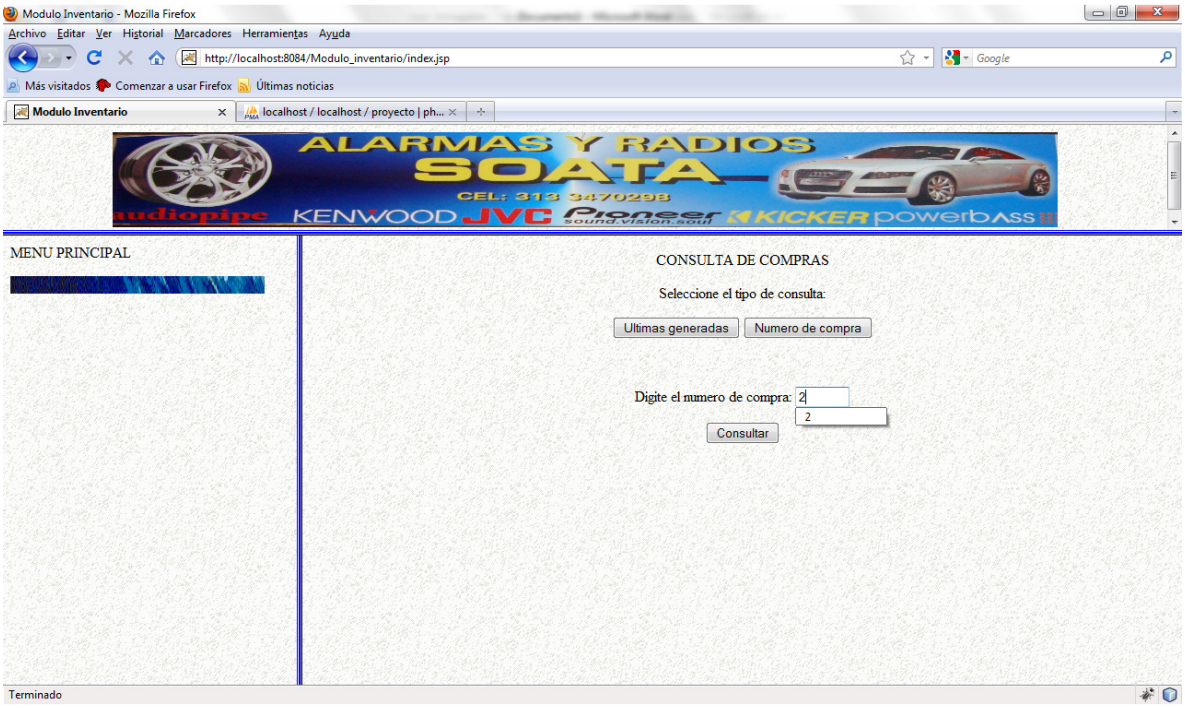
Ultimas generadas

Muestra el listado de las últimas compras registradas en el sistema y de igual manera permite consultar cada uno de los detalles de la compra seleccionada.

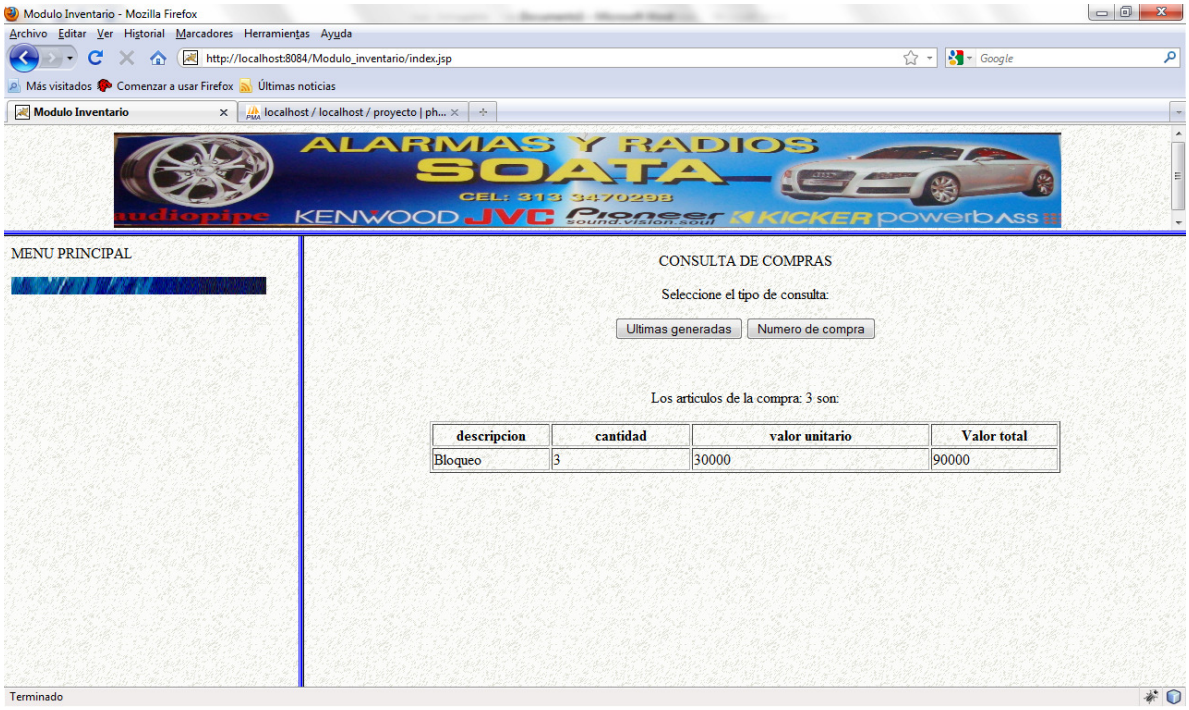
Consultar detalle

Se debe seleccionar la Compra para ver los detalles que serán mostrados de la siguiente manera

Consultar por número de compra

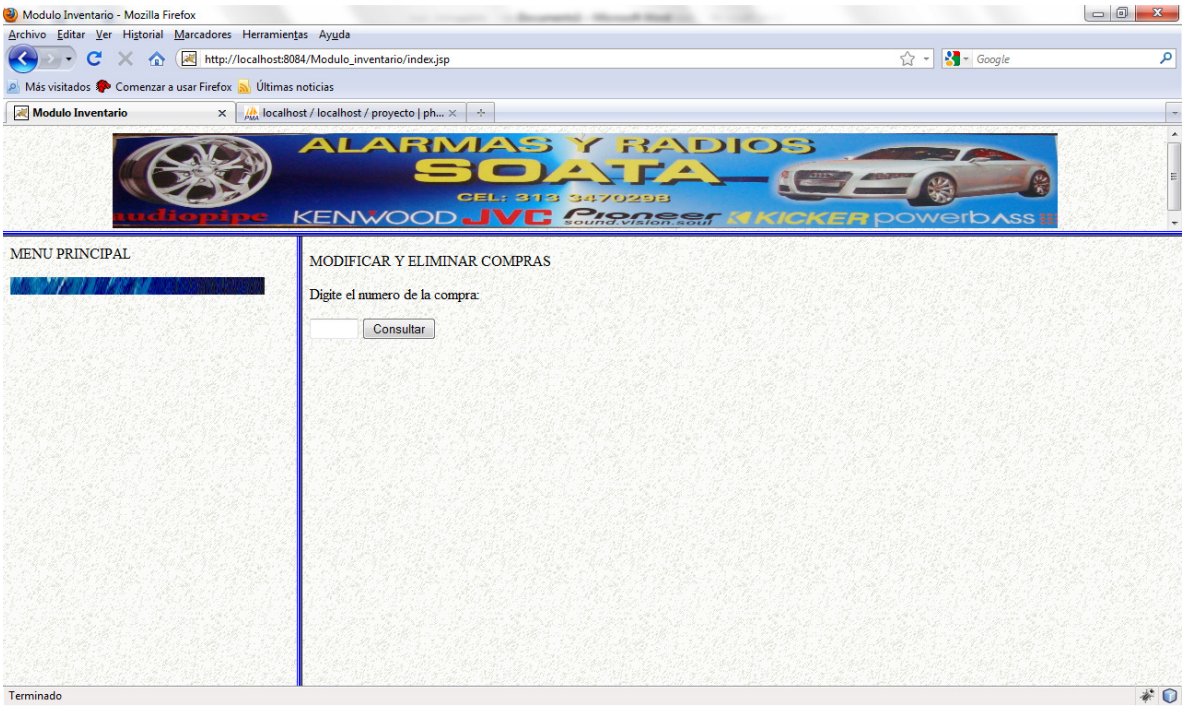


De igual manera Mostrara los detalles relacionados con la esa compra.



Modificar y eliminar compras

Se debe digitar el numero de compra que se desea modificar asi:



Pantalla de modificación y eliminación.

Donde podemos modificar:

El numero de factura de compra

Cantidad de artículos comprados dentro del detalle

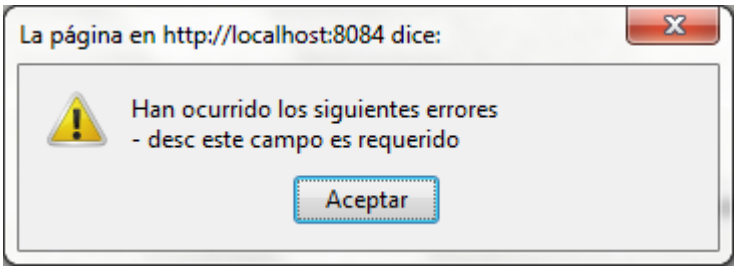
Para eliminar podemos:

Eliminar un detalle de una compra.

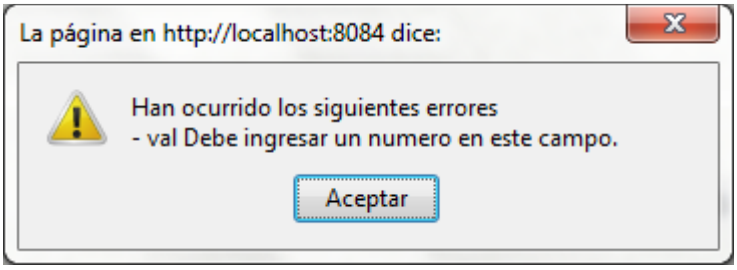
Eliminar una compra siempre y cuando no existan relacionados detalles a la compra.

Mensajes De Error

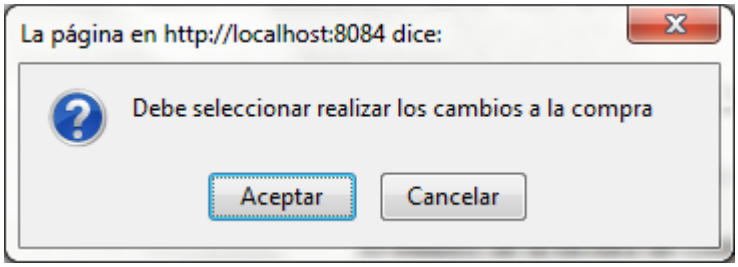
Cuando un campo es requerido aparecerá:



Cuando un campo debe ser numérico aparecerá:



Cuando un campo debe ser seleccionar un detalle aparecerá:



11.3 INTERFAZ DE PROCESAMIENTO DE DATOS

Clases del sistema

Article

```
public class articulo {

    protected int id_articulo;

    protected String descripcion;

    protected String ean;

    protected int valor;

    protected int stock;

    protected int existencias;


    public articulo()

    {

        id_articulo = 0;

        descripcion = "";

        ean = "";

        valor = 0;

        stock = 0;

        existencias = 0;

    }//Cierre del metodo constructor
```

```
//Metodos de asignacion

    public void set_id_articulo(int id_articulo)
    {       this.id_articulo = id_articulo;    }

    public void set_descripcion(String descripcion)
    {       this.descripcion = descripcion;    }

    public void set_ean(String ean)
    {       this.ean = ean;    }

    public void set_valor(int valor)
    {       this.valor = valor;    }

    public void set_stock(int stock)
    {       this.stock = stock;    }

    public void set_existencias(int existencias)
    {       this.existencias = existencias;    }
```

```
//Metodos de obtencion
    public int get_id_articulo()
    {       return id_articulo;    }

    public String get_descripcion()
    {       return descripcion;    }

    public String get_ean()
    {       return ean;    }

    public int get_valor()
    {       return valor;    }

    public int get_stock()
    {       return stock;    }

    public int get_existencias()
    {       return existencias;    }
}//Cierre de la clase
```

Ciudad

```
public class ciudad {
    protected int id_ciudad;
    protected String nombre;
    protected int cod_depto;

    public ciudad()
    {
        id_ciudad = 0;
        nombre = "";
        cod_depto = 0;
    }//Cierre del metodo constructor

    //Declaracion de metodos de obtencion
      public void set_id_ciudad(int id_ciudad)
    {       this.id_ciudad = id_ciudad;    }

    public void set_nombre(String nombre)
    {       this.nombre = nombre;    }
```

```
    public void set_cod_depto(int cod_depto)
    {       this.cod_depto = cod_depto;    }

    //Declaracion de metodos de obtención
    public int get_id_ciudad()
    {       return id_ciudad;    }

    public String get_nombre()
    {       return nombre;    }

       public int get_cod_depto()
    {       return cod_depto;    }

}//Cierre de la clase
```

Compras

```
public class compras {
    protected int id_compra;
    protected String fecha_com;
    protected int cod_suc;
    protected String nro_faccom;
    protected int valor_com;

    public compras()
    {
        id_compra = 0;
        fecha_com = "";
        cod_suc = 0;
        nro_faccom = "";
        valor_com = 0;
    }//Cierre  del metodo constructor
    //Declaración de métodos de asignación
    public void set_id_compra(int id_compra)
    {       this.id_compra = id_compra;    }
```

```
    public void set_fecha_com(String fecha_com)
    {       this.fecha_com = fecha_com;    }
    public void set_cod_suc(int cod_suc)
    {       this.cod_suc = cod_suc;    }
    public void set_nro_faccom(String nro_faccom)
    {       this.nro_faccom = nro_faccom;    }
    public void set_valor_com(int valor_com)
    {       this.valor_com = valor_com;    }
    //Declaracion de metodos de obtención
    public int get_id_compra()
    {       return id_compra;    }
       public String get_fecha_com()
    {       return fecha_com;    }
    public int get_cod_suc()
    {       return cod_suc;    }
    public String get_nro_faccom()
    {       return nro_faccom;    }
    public int get_valor_com()
    {       return valor_com;    }
}// Cierre de la clase compras
```

Departamento

```
public class departamento {
    //Declaracion de atributos
    protected int id_depto;
    protected String nombre;
    public departamento()
    {
        id_depto = 0;
        nombre = "";
    }//Cierre  del metodo constructor
    //Declaración de métodos de asignación
    public void set_id_depto(int id_depto)
    {       this.id_depto = id_depto;    }
    public void set_nombre(String nombre)
    {       this.nombre = nombre;    }
    public int get_id_depto()
    {       return id_depto;    }
    public String get_nombre()
    {       return nombre;    }
}//Cierre de la clase departamento
```

Det_compra

```
public class det_compra {

     //Declaracion de atributos

    protected int id_detcompra;

    protected int cod_compra;

    protected int cod_articulo;

    protected int cantidad;

    protected int vlr_unitario;

    protected int vlr_total;



    public det_compra()

    {

        id_detcompra = 0;

        cod_compra = 0;

        cod_articulo = 0;

        cantidad = 0;

        vlr_unitario= 0;

        vlr_total = 0;

    }//Cierre del metodo constructor
```

```
//Declaración de métodos de asignación
public void set_id_detcompra(int id_detcompra)
{       this.id_detcompra = id_detcompra;    }

public void set_cod_compra(int cod_compra)
{       this.cod_compra = cod_compra;    }

public void set_cod_articulo(int cod_articulo)
{       this.cod_articulo = cod_articulo;    }

public void set_cantidad(int cantidad)
{       this.cantidad = cantidad;    }

public void set_vlr_unitario(int vlr_unitario)
{       this.vlr_unitario = vlr_unitario;    }

public void set_vlr_total(int vlr_total)
{       this.vlr_total = vlr_total;    }
```

```
    //Declaracion de metodos de obtención

    public int get_id_detcompra()
    {       return id_detcompra;    }

    public int get_cod_compra()
    {       return cod_compra;    }

    public int get_cod_articulo()
    {       return cod_articulo;    }

    public int get_cantidad()
    {       return cantidad;    }

    public int get_vlr_unitario()
    {       return vlr_unitario;    }

    public int get_vlr_total()
    {       return vlr_total;    }
}//Cierre de la clasedet_compra
```

Proveedor

```
public class proveedor {
    protected int id_prov;
    protected String nombre_prov;
    protected String nit;
        public proveedor()
    {
        id_prov = 0;
        nombre_prov = "";
        nit = "";
    }//Cierre  del metodo constructor

    //Declaración de métodos de asignación
    public void set_id_prov(int id_prov)
    {       this.id_prov = id_prov;    }
    public void set_nombre_prov(String nombre_prov)
    {       this.nombre_prov = nombre_prov;    }
    public void set_nit(String nit)
    {       this.nit = nit;    }
```

```
    //Declaracion de metodos de obtencion


    public int get_id_prov()
    {       return id_prov;    }
    public String get_nombre_prov()
    {       return nombre_prov;    }
    public String get_nit()
    {       return nit;    }
}//Cierre de la clase proveedor
```

Sucursal

```
public class sucursal {
    //Declaracion de atributos
    protected int id_suc;
    protected String nom_sucursal;
    protected int cod_prov;
    protected int cod_ciudad;
    protected String direccion;
    protected String telefonos;

    //Declaracion del metodo constructor
    public sucursal()
    {
        id_suc = 0;
        nom_sucursal = "";
        cod_prov = 0;
        cod_ciudad = 0;
        direccion = "";
        telefonos = "";
    }//Cierre  del metodo constructor
```

```
//Declaración del método de asignación del id_suc
public void set_id_suc(int id_suc)
{
    this.id_suc = id_suc;
}//Cierre del método de asignación del id_suc

//Declaración del método de asignación del nom_sucursal
public void set_nom_sucursal(String nom_sucursal)
{
    this.nom_sucursal = nom_sucursal;
}//Cierre del método de asignación del nom_sucursal

//Declaración del método de asignación del cod_prov
public void set_cod_prov (int cod_prov)
{
    this.cod_prov = cod_prov;
}//Cierre del método de asignación del cod_prov

//Declaración del método de asignación del cod_ciudad
```

```
public void set_cod_ciudad (int cod_ciudad)
{
    this.cod_ciudad = cod_ciudad;
}//Cierre del método de asignación del cod_ciudad

//Declaración del método de asignación del direccion
public void set_direccion(String direccion)
{
    this.direccion = direccion;
}//Cierre del método de asignación del direccion

//Declaración del método de asignación del telefonos
public void set_telefonos(String telefonos)
{
    this.telefonos = telefonos;
}//Cierre del método de asignación del telefonos

//Declaracion del metodo de obtencion del id_suc
public int get_id_suc()
{
```

```
    return id_suc;
}//Cierre del metodo de obtencion del id_suc

//Declaracion del metodo de obtencion del nom_sucursal
public String get_nom_sucursal()
{
    return nom_sucursal;
}//Cierre del metodo de obtencion del nom_sucursal

//Declaracion del metodo de obtencion del cod_prov
public int get_cod_prov()
{
    return cod_prov;
}//Cierre del metodo de obtencion del cod_prov

//Declaracion del metodo de obtencion del cod_ciudad
public int get_cod_ciudad()
{
    return cod_ciudad;
}//Cierre del metodo de obtencion del cod_ciudad
```

```
    //Declaracion del metodo de obtencion del direccion
    public String get_direccion()
    {
        return direccion;
    }//Cierre del metodo de obtencion del direccion

    //Declaracion del metodo de obtencion del telefonos
    public String get_telefonos()
    {
        return telefonos;
    }//Cierre del metodo de obtencion del telefonos

}//Cierre de la clase departamento
```

12. DESARROLLO Y PRUEBAS

Se desarrolla el modulo de compras con el tratamiento de datos funcionales de almacenamiento, consulta, modificación y eliminación con el fin de cumplir los requisitos necesarios del almacén “ALARMAS Y RADIOS SOATÁ”.

Este modulo es la parte inicial para continuar con la realización del modulo completo de administración del almacén.

Se realizan las pruebas de cada una de las funciones del sistema realizando las correcciones necesarias para cumplir los requerimientos del usuario, dejando totalmente funcional el modulo de compras que permite adicionalmente actualizar los datos de ingreso de los artículos del inventario.

13. CONCLUSIONES

Esta totalmente comprobado que los sistemas de información son aplicables a muchos sectores empresariales, se sabe que son necesarios para llevar de forma organizada la administración de una empresa y aplicable al manejo oportuno y veraz de un inventario.

El desarrollo de una aplicación de software permite poner en práctica los conocimientos adquiridos en la universidad durante los semestres cursados en la tecnología informática.

El manejo de java orientado a objetos nos permite garantizar aun más la seguridad de los datos involucrados en un sistema de información y además nos permite manejar un lenguaje de programación puro.

La realización de este proyecto me permite adquirir experiencia en el desarrollo de software y además afianzar mis conocimientos sobre programación.

14. RECOMENDACIONES

Leer totalmente los manuales técnicos y de usuario para realizar una mejor interpretación del aplicativo.

Validar que el antivirus, java y el sistema operativo permanezcan actualizados para el correcto funcionamiento del sistema.

Todos los usuarios del sistema deben asistir a capacitación sobre el manejo del aplicativo.

REFERENCIAS

Pressman, Rogers. *INGENIERÍA DEL SOFTWARE: un enfoque Práctico.* (2002) McGraw-Hill .Aravaca(Madrid). (5 ed.).

Desarrollo WEB. http://www.desarrolloweb.com/articulos/499.php. [Consultado Marzo 10 de 2010].

Mysqlya. http://www.mysqlya.com.ar/temarios/descripcion.php?cod=2&punto=1 .[Consultado abril de 2010].