

**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERIA**

**Escuela Profesional de Ingeniería de Sistemas**

**Trabajo Final-Unidad II-Proyecto**

**“Aplicación del NOSQL en el Proyecto Aplicativo Web y Sistemas de Ventas Electrodomésticos”**

Curso: Base de Datos II

Docente: Ing. Patrick José Cuadros

INTEGRANTES:

* Linares Chávez, Cesar (2019063854)
* Lira Álvarez , Rodrigo (2019063331)
* Perez Vizcarra, Juan Jose (2019063636)

**Tacna – Perú**

**2022**

**INDICE GENERAL**

[RESUMEN 3](#_Toc101765838)

[ABSTRACT 3](#_Toc101765839)

[1. ANTECEDENTES O INTRODUCCIÓN 4](#_Toc101765840)

[2. TITULO 4](#_Toc101765841)

[3. AUTORES 4](#_Toc101765842)

[4. PLANTEAMIENTO DEL PROBLEMA 4](#_Toc101765843)

[5. OBJETIVOS 7](#_Toc101765847)

[6. REFERENTES TEORICOS 7](#_Toc101765849)

[7. DESARROLLO DE LA PROPUESTA 10](#_Toc101765850)

[8. BIBLIOGRAFIA 33](#_Toc101765853)

[9. ANEXOS 33](#_Toc101765854)

# **RESUMEN**

La Tienda Nutrifit, ubicada en la ciudad de Tacna, se dedica a la venta de artículos electrodomésticos, opera desde la mañana hasta la tarde tiene un pequeño almacén es su segundo piso y actualmente están buscando una renovación de su sistema.

La empresa necesita una recolección de datos de forma rápida y simple, que desea adquirir a través de la obtención de datos que puedan ser utilizados en las labores de los empleados cuando ellos lo necesiten.

Según sus necesidades se plantean que en el nuevo sistema tenga como prioridad el registro de ventas, el registro de los empleados, el registro de clientes y la impresión de las boletas.

También se hará la integración del aplicativo web que contendrá los artículos que estén en disponibilidad para que los clientes puedan realizar sus compras con facilidad.

# **ABSTRACT**

The Nutrifit Store, located in the city of Tacna, is dedicated to the sale of household appliances, operates from morning to afternoon, has a small warehouse on the second floor and is currently looking for a renovation of its system.

The company needs a quick and simple data collection, which they want to acquire through the collection of data that can be used in the work of employees when they need it.

According to their needs, the new system will have as a priority the registration of sales, the registration of employees, the registration of customers and the printing of receipts.

The integration of the web application that will contain the items that are in availability so that customers can make their purchases easily will also be done.

# **1. Antecedentes o Introducción**

La tienda Nutrifit, es una empresa pequeña de electrodomésticos que se encuentra en el centro de la ciudad de Tacna, ofrece variedad de productos electrónicos especialmente lideres en venta de electrodomésticos, ofreciendo diferentes marcas tales como Sony, LG, Samsung, AIWA y otros.

Agregando también, que poseen muchas categorías de productos de los cuales tienen para ofrecer, en estos momentos, al aumentar las categorías como también productos nuevos y la acumulación de información anteriores y registros, surge la necesidad de realizar nuevas actualizaciones, consultas rápidas y completas para ver la información detallada de los productos, como también funciones para las consultas e inserciones de los empleados, llegando a utilizar diversos modelos para que las consultas de datos se desarrollen con éxito.

# **2. TITULO**

Aplicación del NOSQL en el Proyecto de Aplicativo Web Tienda de Electrodomésticos (ARTEC) y Sistemas de Ventas Electrodomésticos.

# **3. AUTORES**

|  |  |  |
| --- | --- | --- |
| N | Nombre | Cargo |
| 1 | Linares Chávez, Cesar | Jefe de Proyecto |
| 2 | Linares Chávez, Cesar | Programador |
| 3 | Perez Vizcarra, Juan Jose | Analista |
| 4 | Perez Vizcarra, Juan Jose | DBA |
| 5 | Lira Álvarez , Rodrigo | Programador |

# **4. PLANTEAMIENTO DEL PROBLEMA**

# **4.1 PROBLEMA**

En la actualidad, La Tienda Nutrifit de momento no cuenta con un sistema que tenga los mantenimientos de los empleados, productos y ventas, que sea rápida y eficaz en las consultas de la base de datos, por tal motivo la empresa requiere una nueva mejora en su sistema.

Para realizar tal mejora e implementado el software requerido por la empresa, se ha desempeñado en la utilización de una metodología bastante útil para el desarrollo de base de datos.

Se trata del OR-M o ORM (Objet Relational Mapping) es un modelo de programación que permitirá convertir los datos de los objetos en formato correcto para guardar la información en una base de dato y quedan vinculados a la base de datos (persistencia) pasando por la creación de una base de datos virtual. Es complicado transformar la información que reciba en los objetos de la aplicación, es decir, hacer tablas en ello.

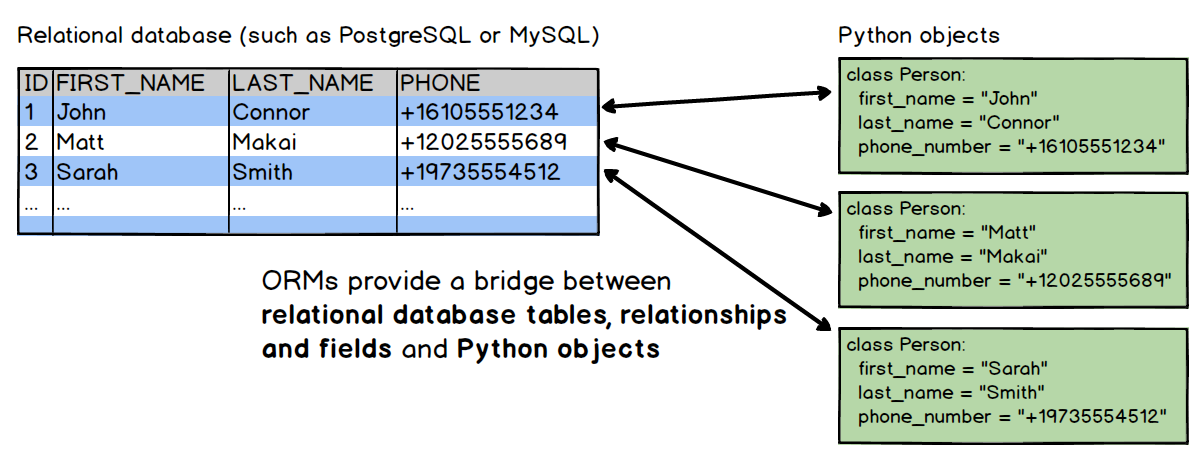
Siendo que este modelo, se basa en las estructuras de la base de datos relacional que son vinculadas con las entidades lógicas o base de datos virtual definida en el ORM, de modo que las acciones CRUD (Create, Read, Update, Delete) se ejecutan sobre la base de datos física y realizan de forma indirecta por medio del ORM.

En resumen, con el modelo de programación mencionado, debamos utilizar las clases entidades y presentaciones, dejando al lado las clases de negocios para poder ahorrar código y consultar los datos de una forma sencilla.

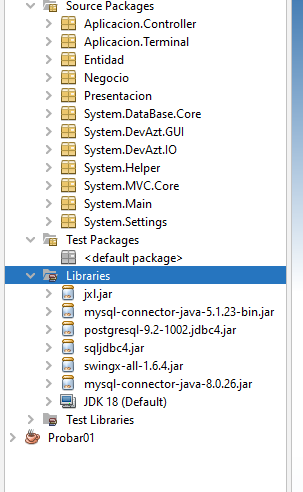
El uso de un ORM en la base de datos permite adaptarse a los nuevos tiempos y darnos una mayor versatilidad a la hora de manejar los datos.

De sus funcionalidades se puede mencionar:

* El ORM funciona como una capa intermedia completamente separada de las bases de datos
* Hace que una migración sea más sencilla.
* Es un código más legible y con menos líneas.
* Otra ventaja importante es la seguridad. Al ser una capa independiente a los datos, nos permite protegerlos de ciberataques al no estar en el mismo nivel.
* Un ORM nos guarda y carga toda la información de una base de datos relacional automáticamente.
* Nos evita escribir a mano consultas de SQL necesarias**.**



El modelo OR-M, puede aclarar el objetivo principal de todo el problema de solucionar y ahorrar el tiempo de las consultas y la utilización del CRUD sin tener que utilizar el modelo de negocio o Clases de negocios en el proyecto siguiendo el proceso de objeto relacional mapeado



# **4.2 JUSTIFICACION**

La finalidad de realizar este proyecto buscar facilitar las consultas de los datos, mantenimiento de los productos y empleados de la tienda para que así la empresa tenga un manejo óptimo de sus recursos permitiéndoles de manera notable incrementar su confianza al realizar sus ventas.

También el proyecto tiene que realizarse en el plazo de tiempo solicitado por la propia empresa, permitiéndoles obtener nuevos aspectos tecnológicos para la automatización de procesos por medio del sistema planteado, acortando distancias, disminuyendo costos y ofreciendo todo tipo de productos electrónicos para toda la comunidad virtual.

# **4.3 ALCANCE**

El sistema tendrá una disponibilidad y alcance de los mantenimientos de los empleados, productos y ventas en lo que pueda ser utilizado con plenitud, también tendrá consultas incluidas dentro del propio sistema para que se pueda obtener información de los datos correspondientes. El sistema es únicamente para los empleados de la tienda y para los clientes, solo tendrá que acceder a las compras de los productos que ofrece la tienda y que después de ello se guardará los datos de la venta, permitiendo optimizar el proceso de mantenimiento y de todos los procesos relacionados con el mismo,

# **5. OBJETIVOS**

# **5.1. GENERAL**

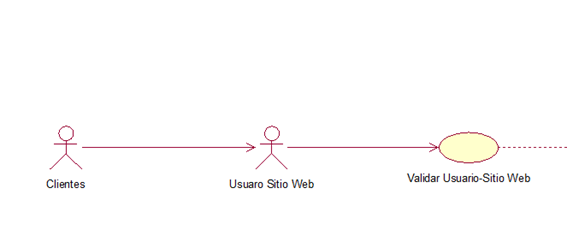
* Establecer los métodos y tipos de mantenimiento a aplicar, de forma adecuada para procurar maximizar la disponibilidad y confiabilidad de la administración de datos y el mantenimiento de los datos de forma eficiente a través de un software para la Empresa Nutrifit.

**5.2. ESPECIFICOS**

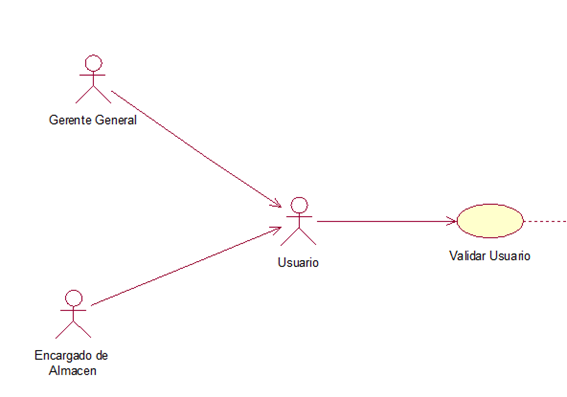
* Reducir el tiempo de respuesta de las consultas de los datos guardados en la base de datos.
* Reducir los costos de mantenimiento.
* Mantener un control del inventario usando el sistema y aplicativo siendo la principal, para los clientes y el otro, la modificación de productos y mantenimientos para los empleados.
* Aumentar la confiabilidad de administración de datos.
* Aumentar la disponibilidad de administración de datos.

# **6. REFERENTES TEORICOS**

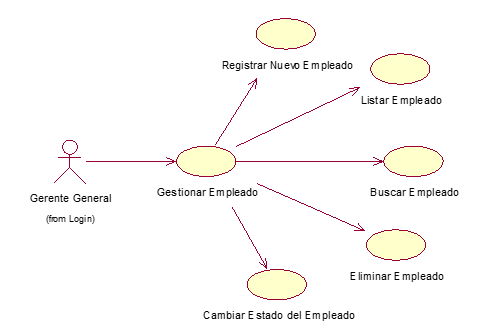
* Diagramas De Casos De Uso
* Paquete – Login – Sitio Web

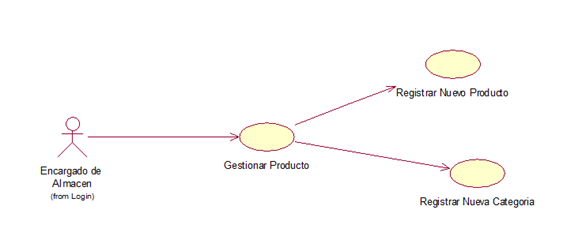
Validar Usuario- Sitio Web

* Paquete-Login

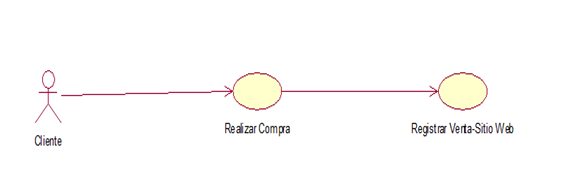
Validar Usuario

* Paquete-Empleados: Gestionar Empleado

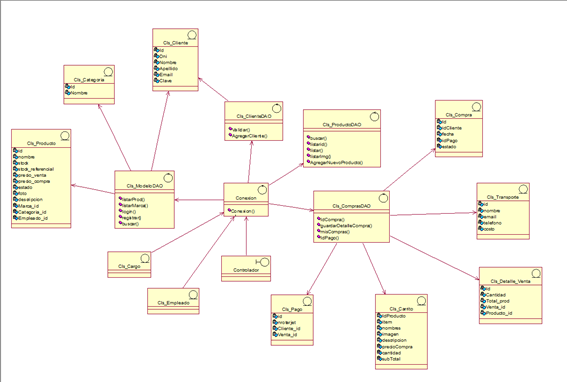


* Paquete-Productos: Gestionar Productos
* Paquete-Ventas-Sitio Web

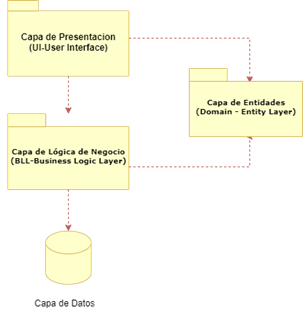
Registrar Venta



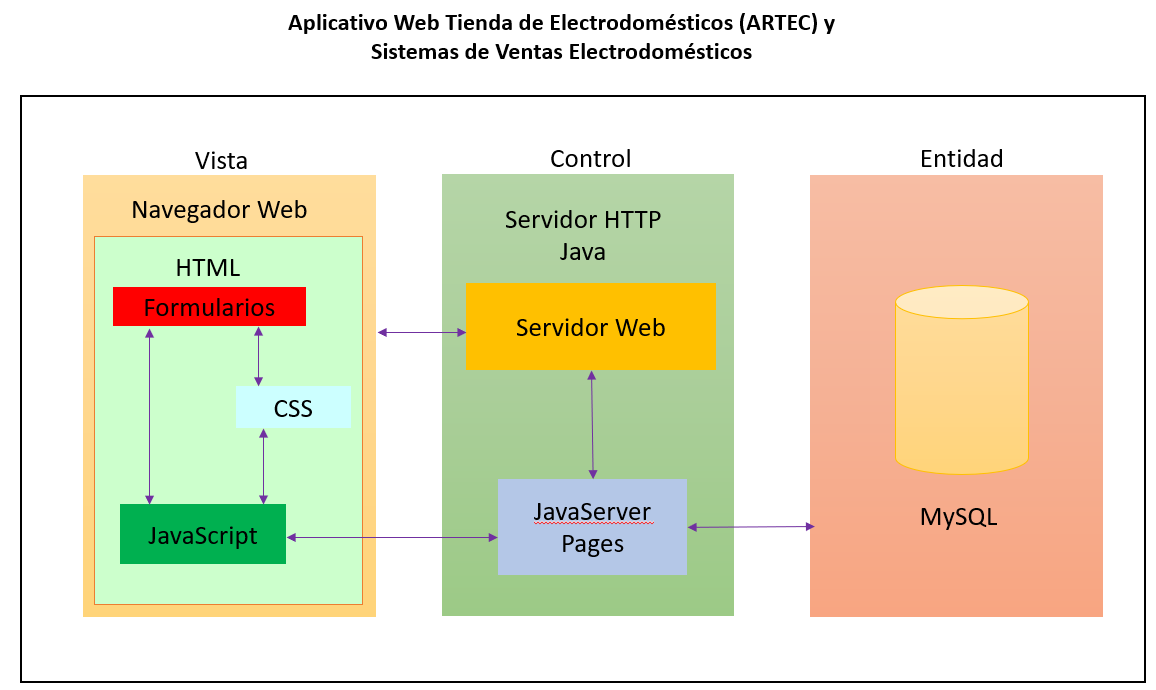
* Diagramas de clases



* Diagramas de componentes

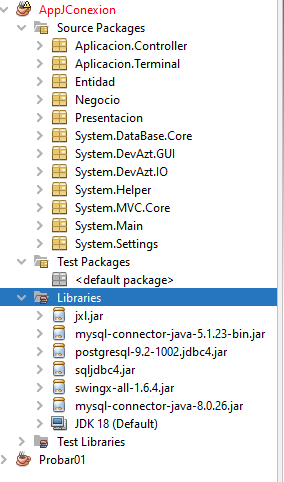


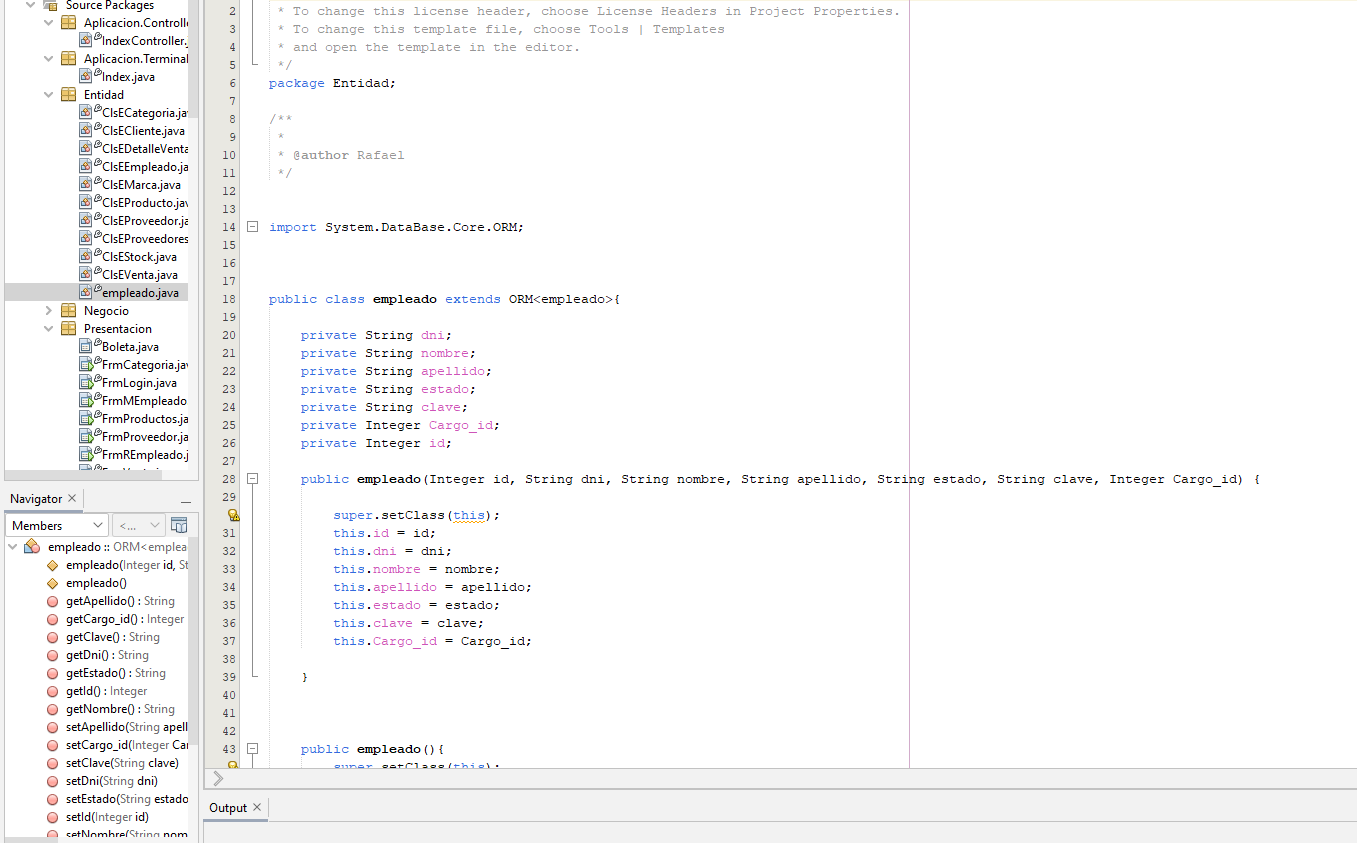
* **Diagrama de Arquitectura de la aplicación**



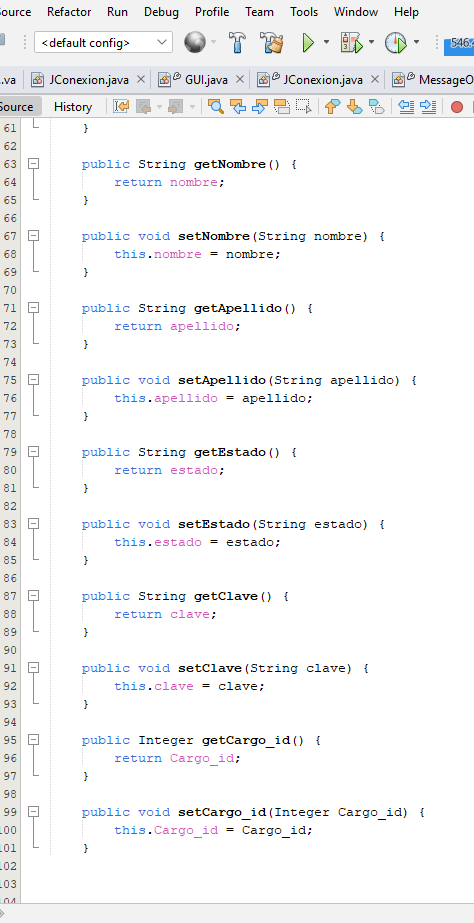
# **7. DESARROLLO DE LA PROPUESTA**

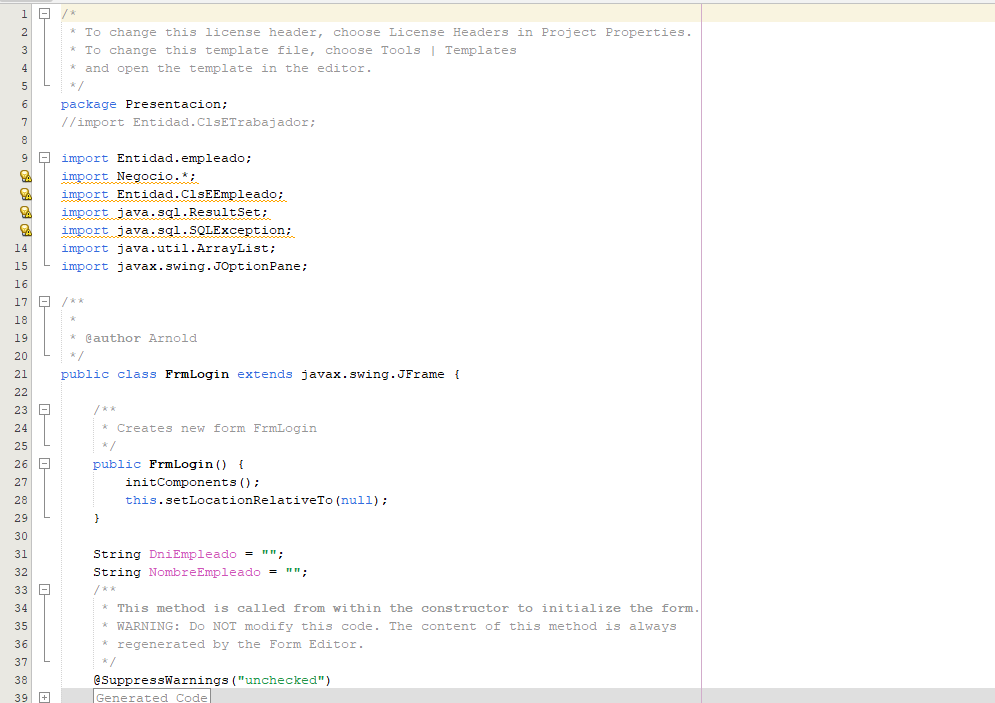
# **7.1 CAPTURAS DE CODIGO DE APLICACION DEL OR/M**

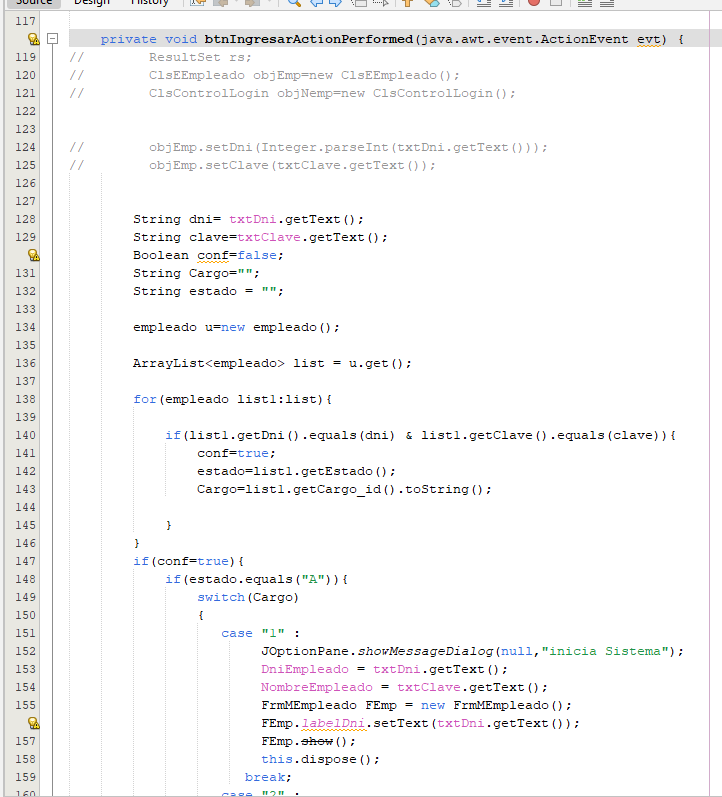








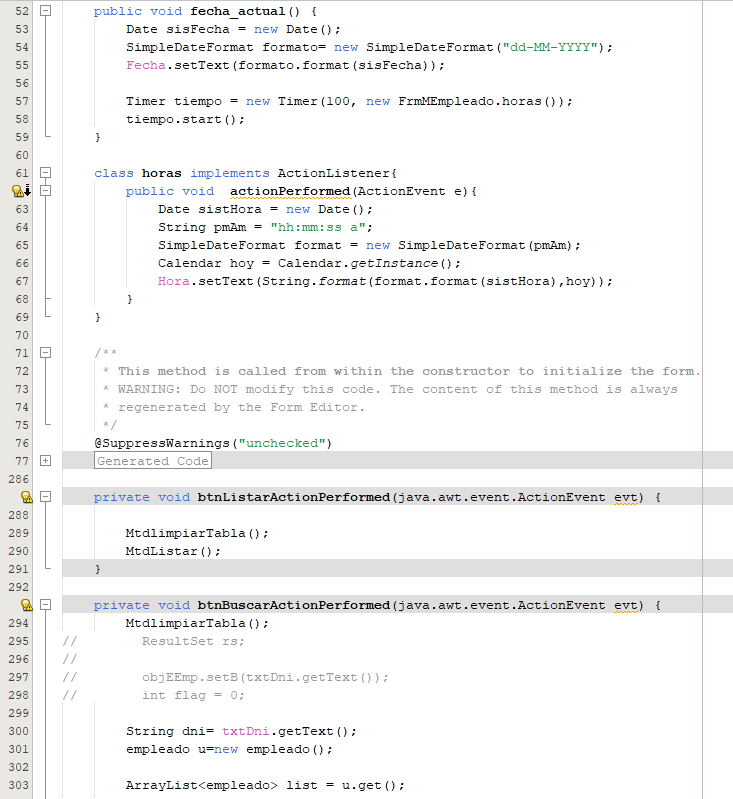








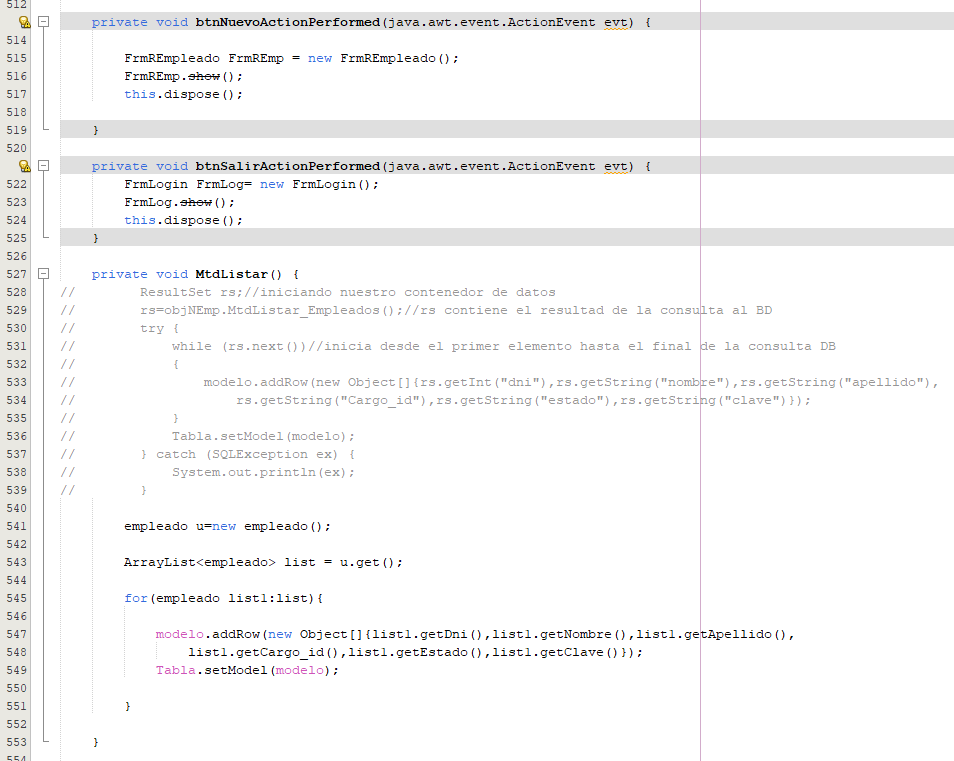


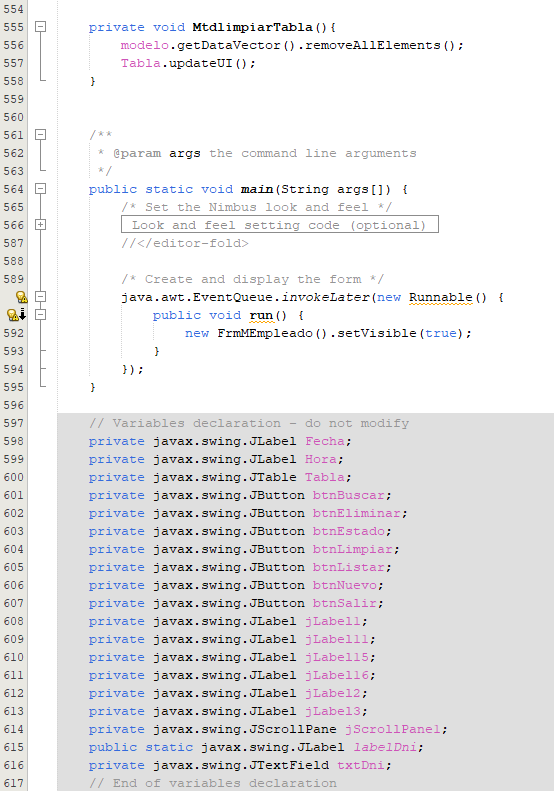






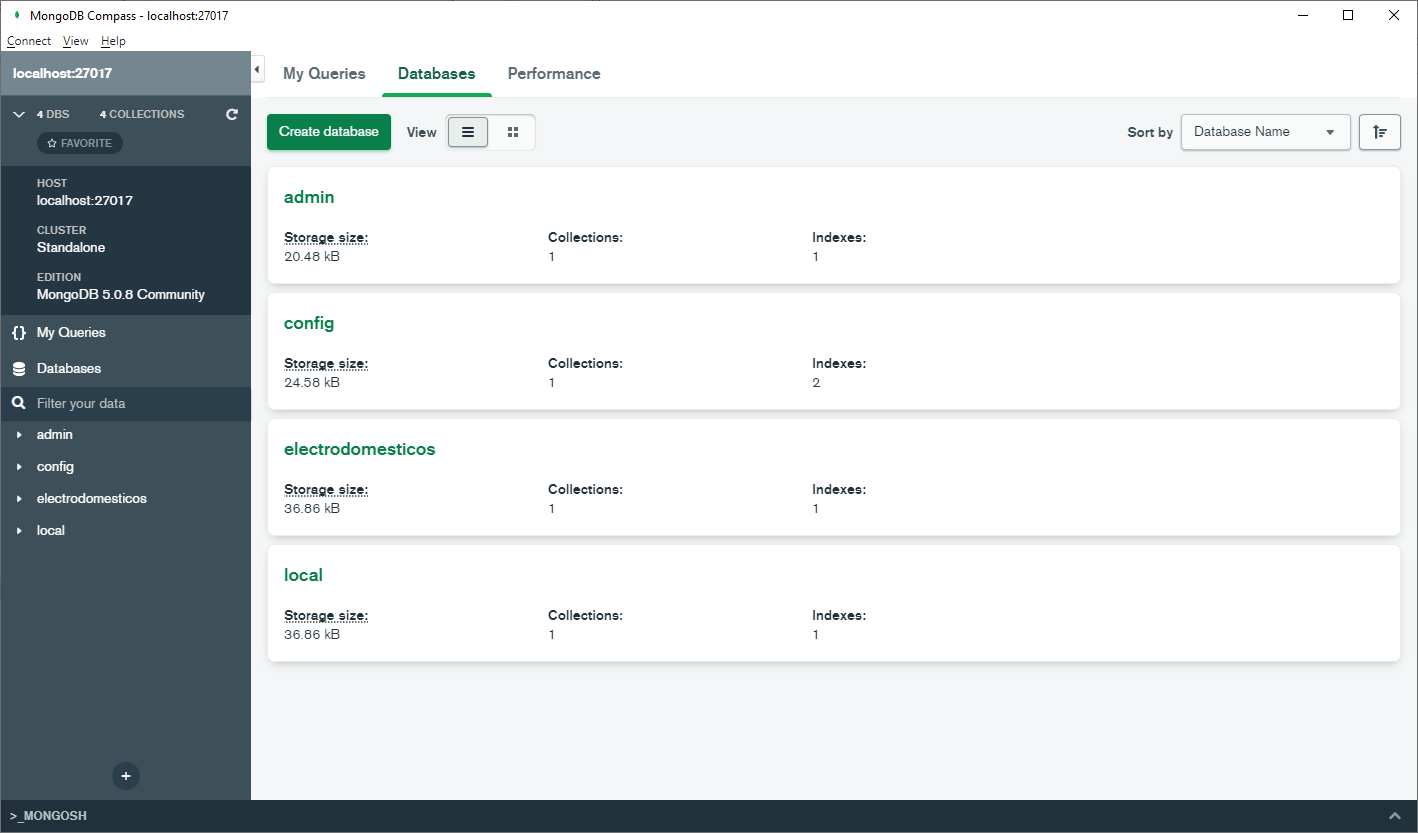




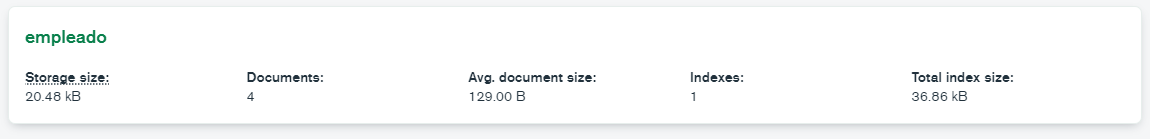


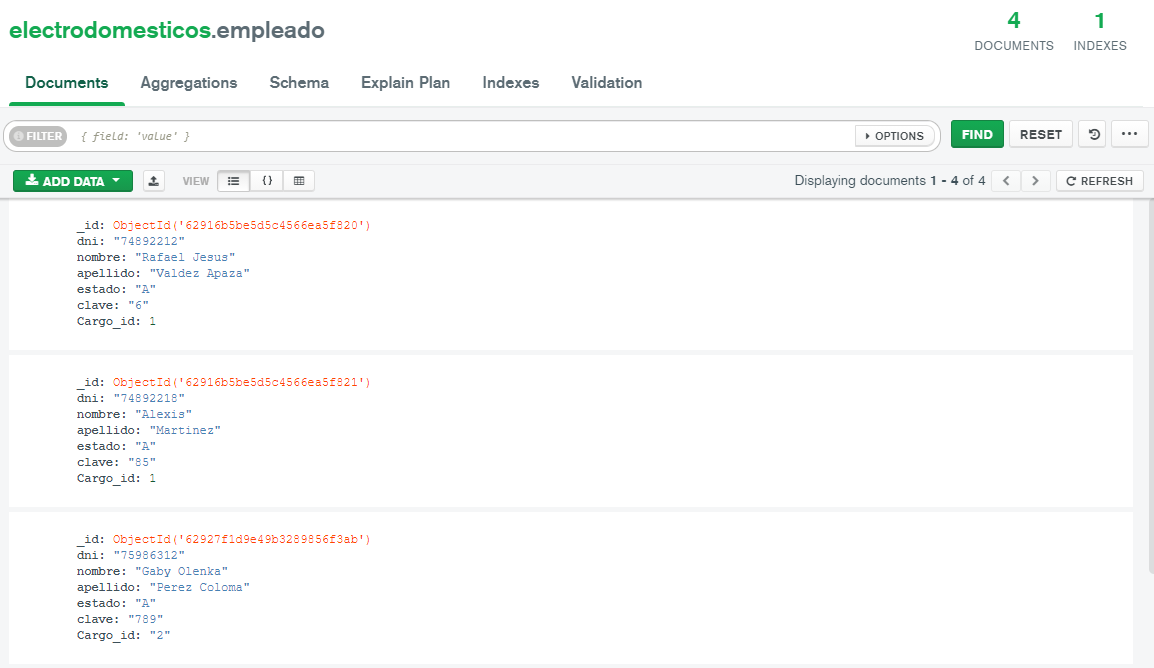
**7.2 Implementar una parte de su proyecto utilizando una base de datos NoSQL, puede ser un requerimiento funcional o no funcional**

Primero se creó una base de datos llamada electrodomésticos, mismo nombre de la bd SQL.

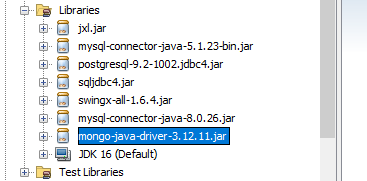


Después se creó una colección llamada empleado, la cual almacenara documentos con los datos de cada empleado.

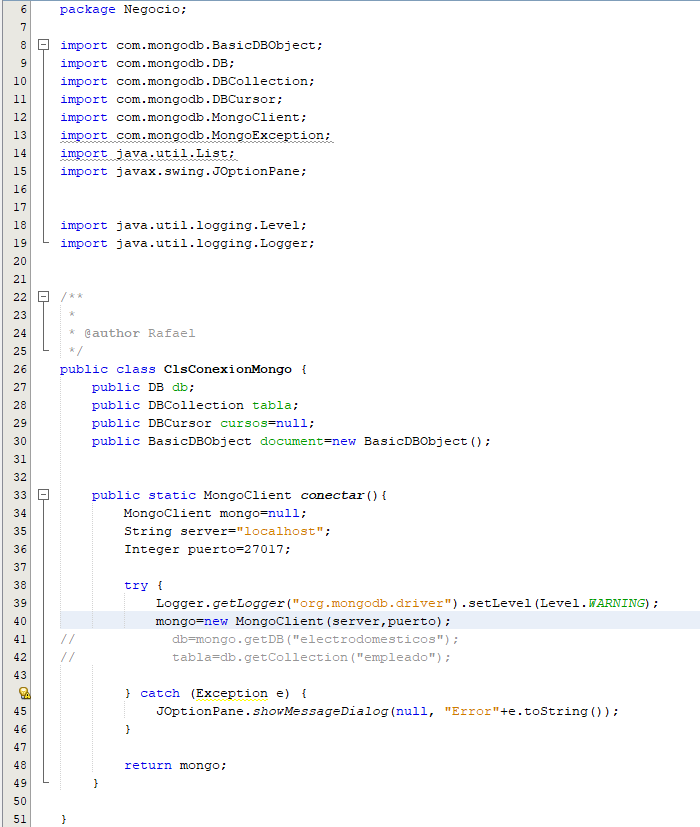




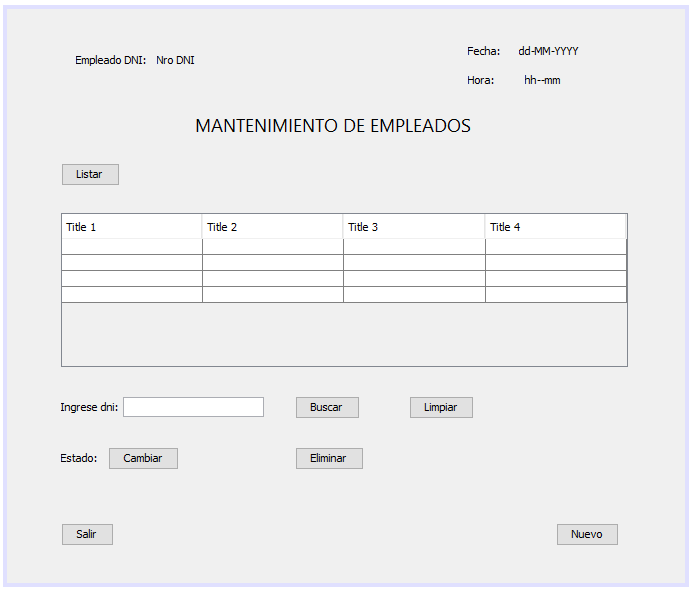
Luego en NetBeans se importó el controlador para usar mongodb dentro de nuestro sistema.



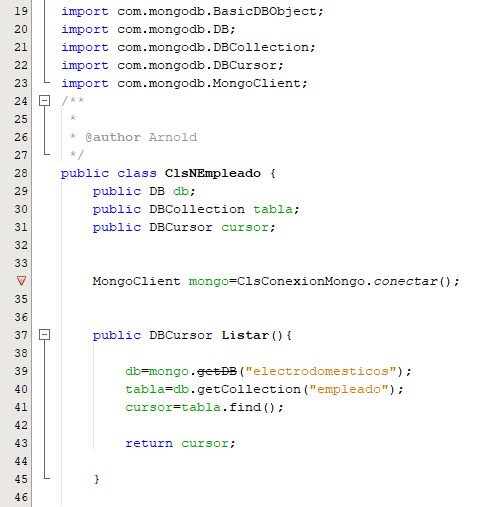
Creamos una clase de conexión donde realizamos la conexión con nuestra base de datos “Electrodomésticos”



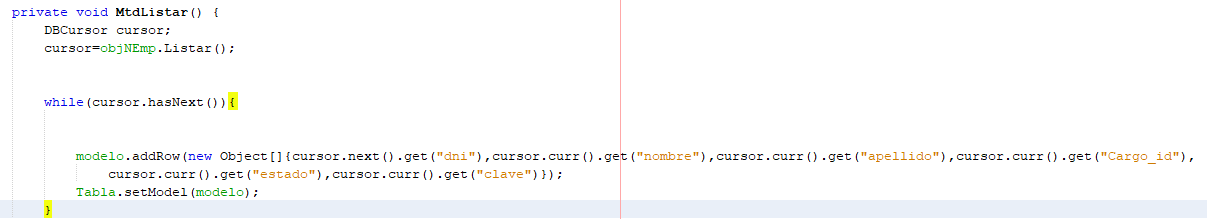
El requerimiento que seleccionamos para utilizar una base de datos NoSQL fue la de gestionar empleado.



Función Listar



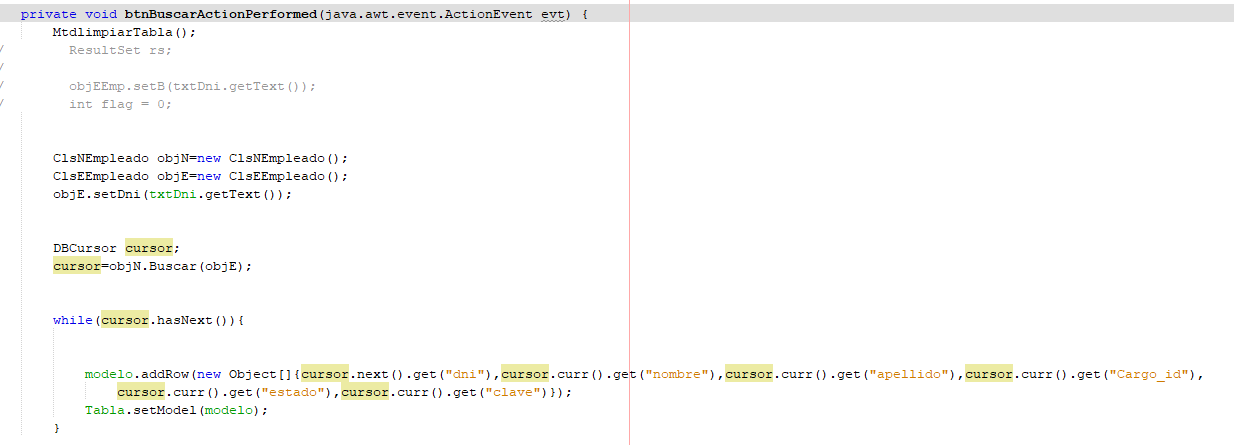
Procedimiento que invoca a la función Listar



Función Buscar



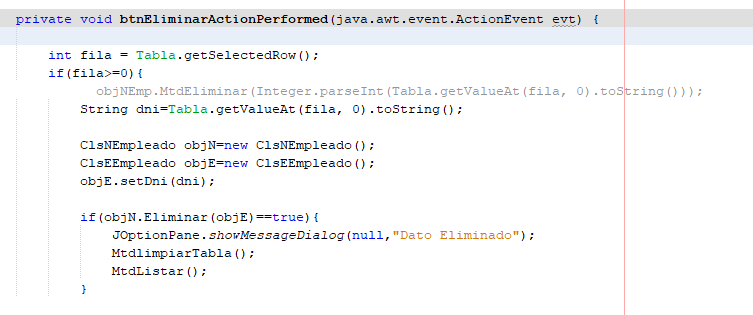
Procedimiento que invoca a la función Buscar



Función Eliminar



Procedimiento que invoca a la función Eliminar

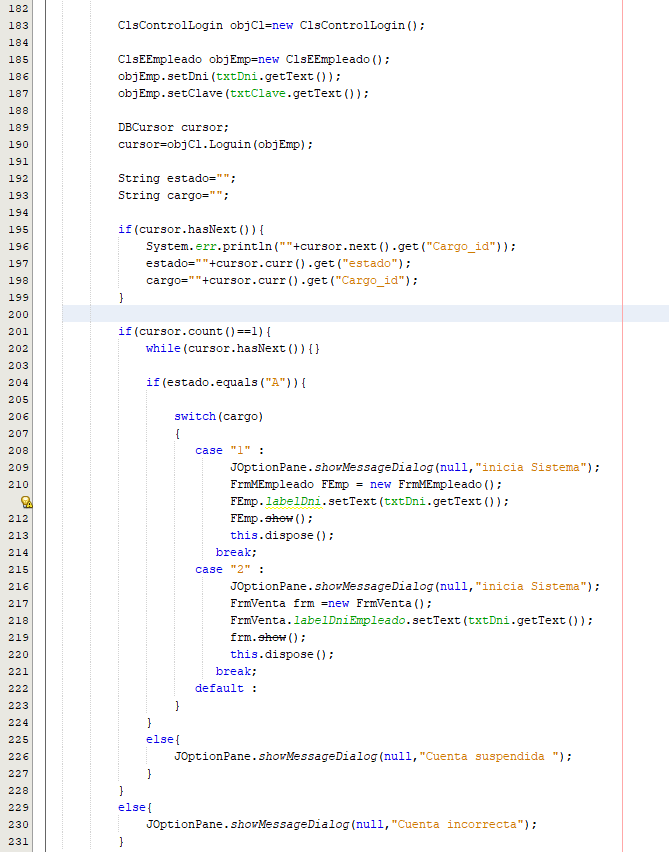


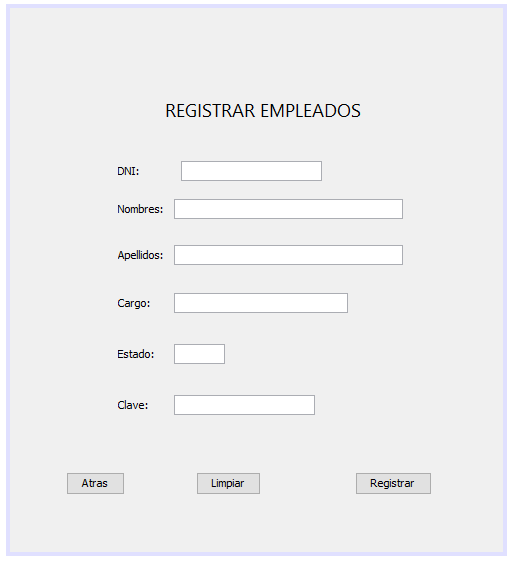


Función Loguin



Procedimiento de Logueo

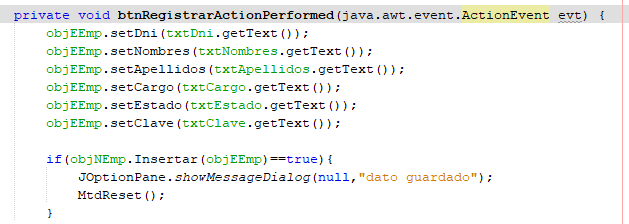




Función Insertar



Botón de invocación



# **7.3 DIAGRAMA DE BASE DE DATOS**

Diagrama SQL:

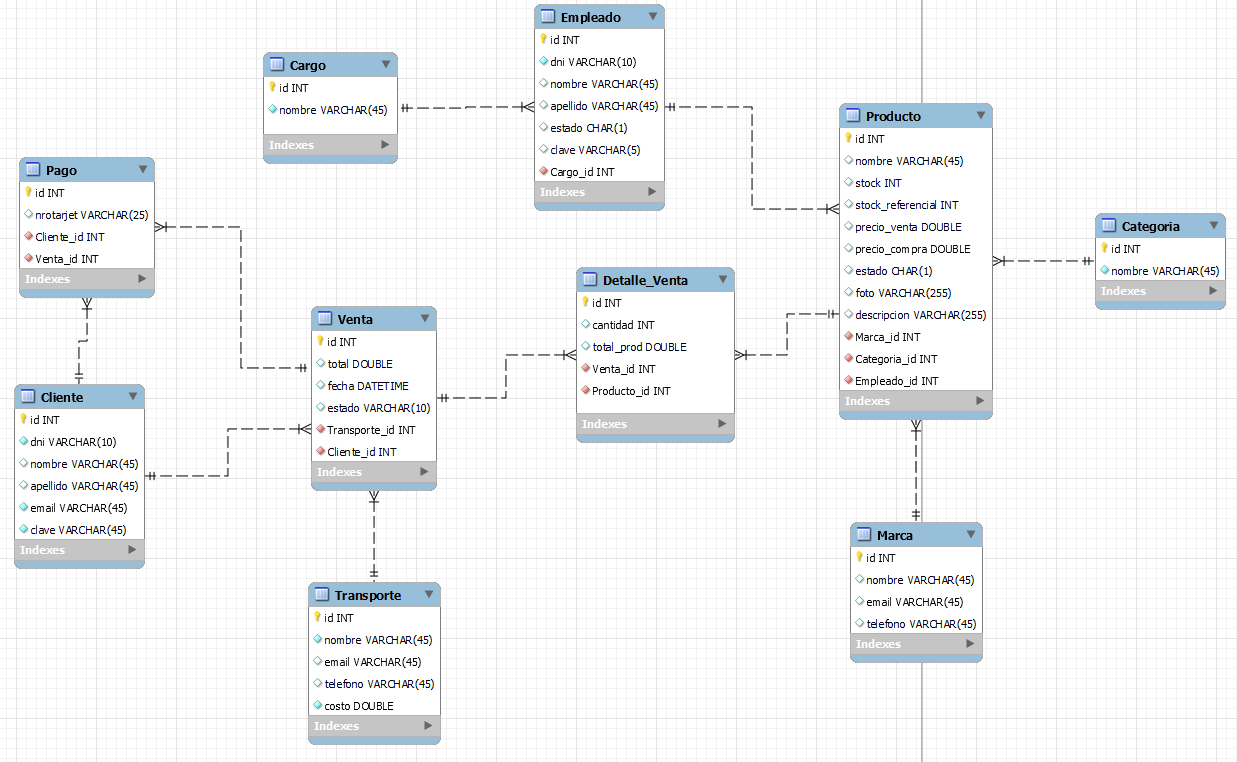
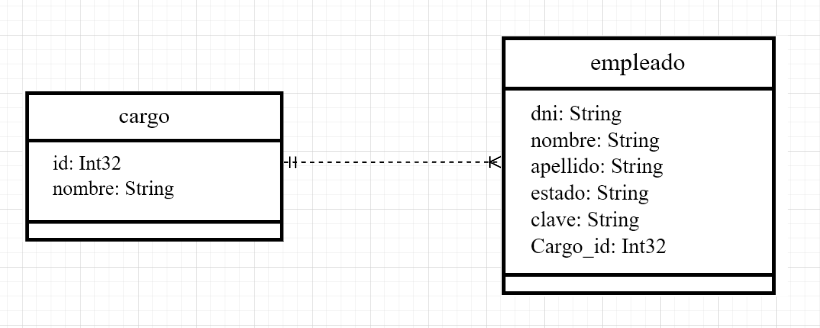


Diagrama NoSQL:

Requerimiento Gestionar Empleado:

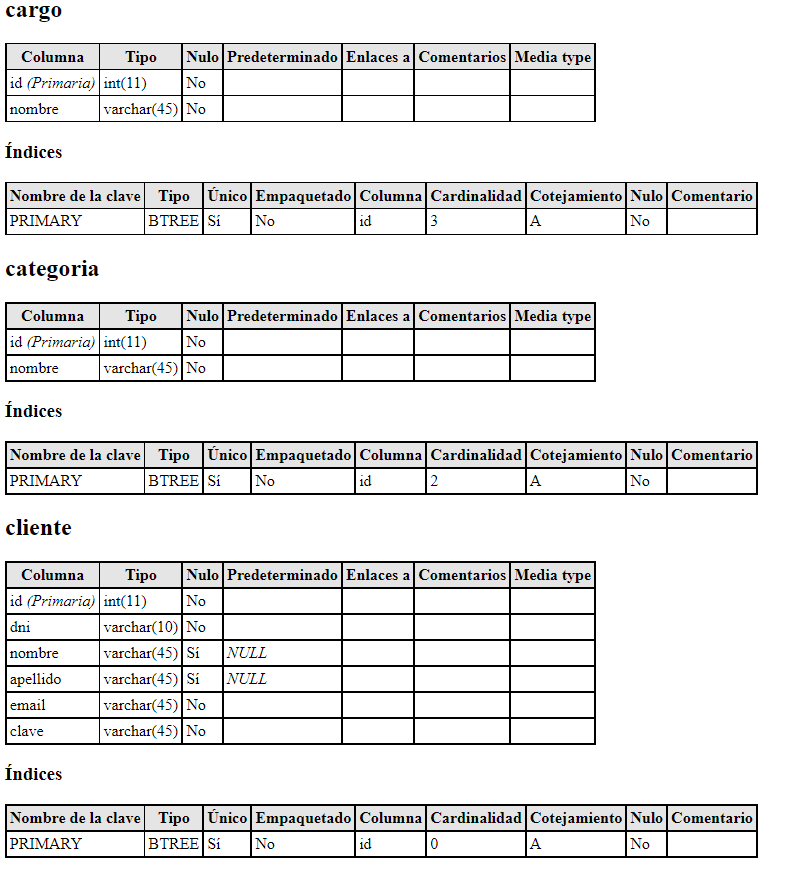


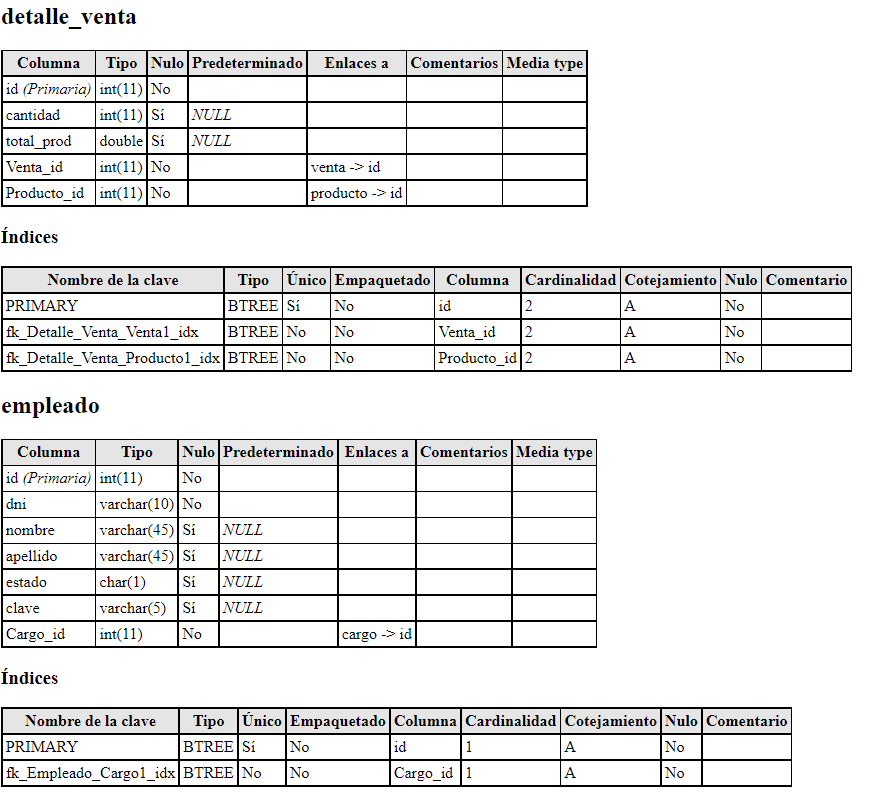
# **8. BIBLIOGRAFIA**

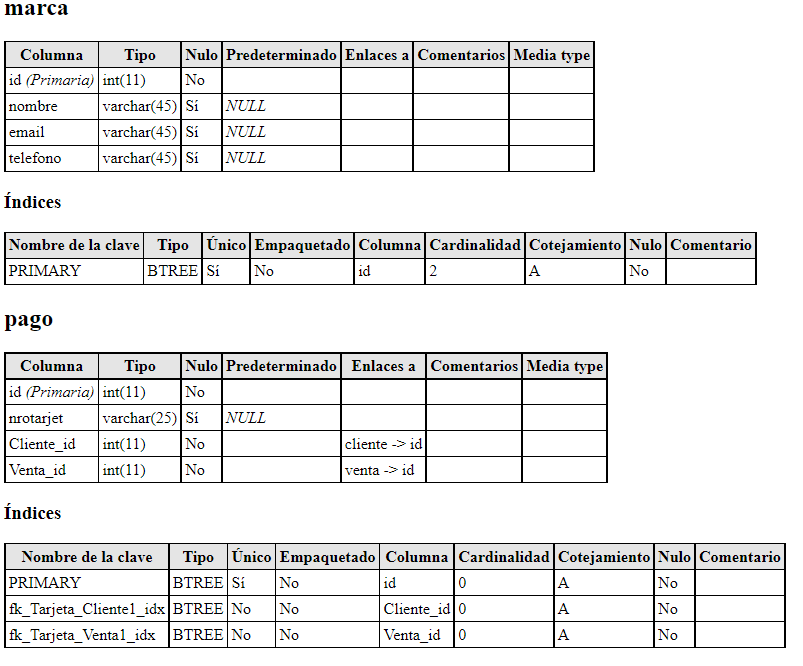
* ESIC Business & Marketing School. (2018). El ORM como herramienta eficiente de trabajo. Esic.edu; ESIC. Recuperado de <https://www.esic.edu/rethink/tecnologia/el-orm-como-herramienta-eficiente-de-trabajo>
* ¿Qué es un ORM? Código Facilito. Recuperado de <https://codigofacilito.com/articulos/orm-explicacion>
* ¿Qué es un ORM? Programarfacil.com. Recuperado de <https://programarfacil.com/blog/que-es-un-orm/>
* ¿Qué es un ORM? (2018). Deloitte Spain. Recuperado de <https://www2.deloitte.com/es/es/pages/technology/articles/que-es-orm.html>
* ¿Cuándo usar un ORM? (2018). Deloitte Spain. Recuperado de <https://www2.deloitte.com/es/es/pages/technology/articles/cuando-usar-orm.html>

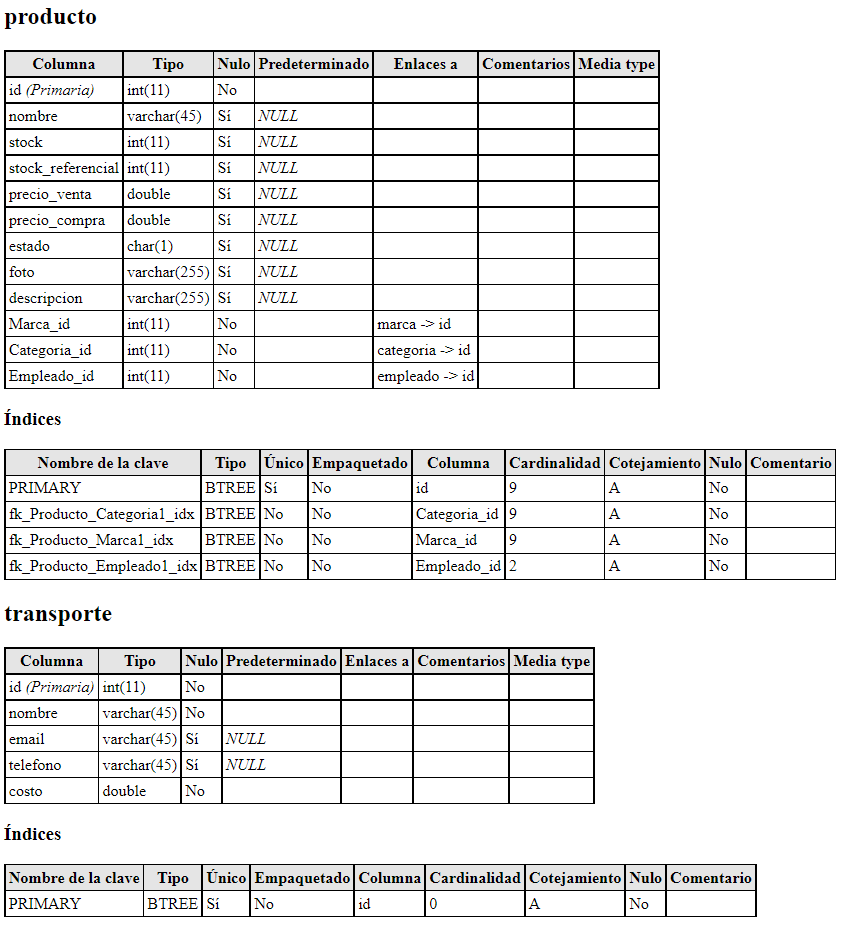
# **9. ANEXOS**

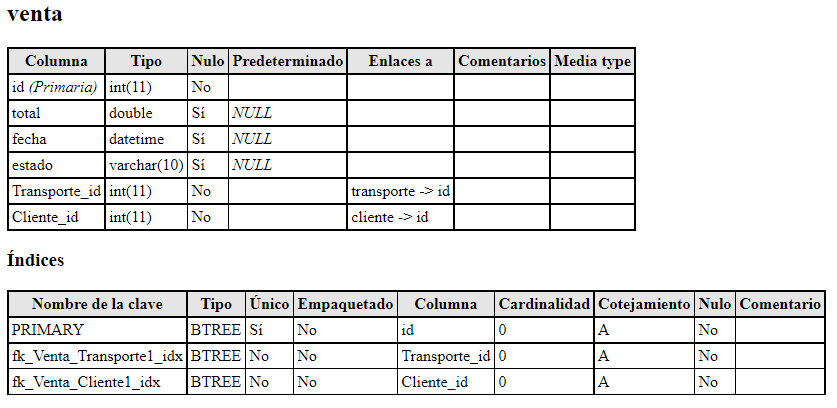
# **9.1. DICCIONARIO DE DATOS DE SU BASE DE DATOS RELACIONAL**











# **9.1.1. NOMBRE DE BASE DE DATOS, DESCRIPCION, TIPO RELACIONAL**

|  |  |  |
| --- | --- | --- |
| **NOMBRE DE BASE DE DATOS** | **DESCRIPCION** | **TIPO RELACIONAL** |
| **electrodomésticos** | Base de Datos transaccional. Se encuentra almacenado todos los datos de las tablas del Sistema y Aplicación. | MySQL |

# **9.1.2. TABLAS CON LAS SIGUIENTES CAMPOS: NOMBRE DE COLUMNA, TIPO DE DATO, PERMITE NULOS, DESCRIPCION, CONSTRAINT, DEFAULT**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| cargo | | | | | |
| NOMBRE DE COLUMNA | TIPO DE DATO | PERMITE  NULOS | DESCRIPCION | CONSTRAINT | DEFAULT |
| id | Int(11) | NO | Numero de Id del cargo | PRIMARY | NINGUNA |
| nombre | Varchar(45) | NO | Nombre del cargo |  | NINGUNA |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| categoría | | | | | |
| NOMBRE DE COLUMNA | TIPO DE DATO | PERMITE  NULOS | DESCRIPCION | CONSTRAINT | DEFAULT |
| id | Int(11) | NO | Numero de id de categoría | PRIMARY | NINGUNA |
| nombre | Varchar(45) | NO | nombre de la categoría |  | NINGUNA |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| cliente | | | | | |
| NOMBRE DE COLUMNA | TIPO DE DATO | PERMITE  NULOS | DESCRIPCION | CONSTRAINT | DEFAULT |
| id | Int(11) | NO | Número del id del cliente | PRIMARY | NINGUNA |
| dni | Varchar(10) | NO | DNI del cliente |  | NINGUNA |
| nombre | Varchar(10) | SI | Nombre del cliente |  | NULL |
| apellido | Varchar(10) | SI | Apellido del Cliente |  | NULL |
| email | Varchar(10) | NO | Correo Electrónico del Cliente |  | NINGUNA |
| contraseña | Varchar(10) | NO | Contraseña del Cliente |  | NINGUNA |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| detalle\_venta | | | | | |
| NOMBRE DE COLUMNA | TIPO DE DATO | PERMITE  NULOS | DESCRIPCION | CONSTRAINT | DEFAULT |
| id | Int(11) | NO | Id de detalles de venta | PRIMARY | NINGUNA |
| cantidad | Int(11) | SI | Cantidad de detalle de ventas |  | NULL |
| Total\_prod | double | SI | Total, de los productos de los detalles de ventas |  | NULL |
| Venta\_id | Int(11) | NO | Número del Id de Venta | Fk\_Detalle\_Venta\_Venta | NINGUNA |
| Producto\_id | Int(11) | NO | Número del id del Producto | Fk\_Detalle\_Venta\_Producto | NINGUNA |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| empleado | | | | | |
| NOMBRE DE COLUMNA | TIPO DE DATO | PERMITE  NULOS | DESCRIPCION | CONSTRAINT | DEFAULT |
| id | Int(11) | NO | Numero de id del empleado | PRIMARY | NINGUNA |
| dni | Varchar(10) | NO | DNI del empleado |  | NINGUNA |
| nombres | Varchar(45) | SI | Nombres del Empleado |  | NULL |
| apellidos | Varchar(45) | SI | Apellidos del Empleado |  | NULL |
| estado | Varchar(1) | SI | Estado del Empleado |  | NULL |
| clave | Varchar(5) | SI | Contraseña del Empleado |  | NULL |
| Cargo\_id | Int(11) | NO | Numero de Id de Cargo | FK\_Empleado\_Cargo | NINGUNA |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| proveedor | | | | | |
| NOMBRE DE COLUMNA | TIPO DE DATO | PERMITE  NULOS | DESCRIPCION | CONSTRAINT | DEFAULT |
| id | Int(11) | NO | Numero de Id de proveedor | PRIMARY | NINGUNA |
| nombre | Varchar(45) | SI | Nombre del Proveedor |  | NULL |
| email | Varchar(45) | SI | Correo Electrónico del Proveedor |  | NULL |
| telefono | Varchar(45) | SI | Teléfono del Proveedor |  | NULL |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| metodo\_pago | | | | | |
| NOMBRE DE COLUMNA | TIPO DE DATO | PERMITE  NULOS | DESCRIPCION | CONSTRAINT | DEFAULT |
| id | Int(11) | NO | Numero de Id del Método de PAGO | PRIMARY | NINGUNA |
| nrotarjet | Varchar(25) | SI | Número de Tarjeta |  | NULL |
| fechaven | date | SI | Fecha |  | NULL |
| cvv | Varchar(4) | SI | Código |  | NULL |
| Cliente\_id | Int(11) | NO | Número del Id Del Cliente | FK\_Tarjeta\_Cliente | NINGUNA |
| Venta\_id | Int(11) | NO | Numero de Id de Venta | FK\_Tarjeta\_Venta | NINGUNA |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| producto | | | | | |
| NOMBRE DE COLUMNA | TIPO DE DATO | PERMITE  NULOS | DESCRIPCION | CONSTRAINT | DEFAULT |
| id | Int(11) | NO | Id del producto | PRIMARY | Ninguna |
| nombre |  | SI | Nombre del producto |  | NULL |
| stock | Int(11) | SI | Stock del producto |  | NULL |
| Stock\_referencial | Int(11) | SI | Numero referencial de la cantidad del producto |  | NULL |
| Precio\_venta | double | SI | Venta del producto |  | NULL |
| Precio\_compra | double | SI | Compra del producto |  | NULL |
| estado | Char(1) | SI | estado |  | NULL |
| foto | longblob | SI | foto |  | NULL |
| descripcion | Varchar(255) | SI | Descripción del producto |  | NULL |
| Proveedor\_id | Int(11) | NO | Numero de id del Proveedor | FK\_Producto\_Proveedor | Ninguna |
| Categoria\_id | Int(11) | NO | Numero id de Categoría | FK\_Producto\_Categoria | Ninguna |
| Empleado\_id | Int(11) | NO | Numero id del Empleado | FK\_Producto\_Empleado | Ninguna |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| transporte | | | | | |
| NOMBRE DE COLUMNA | TIPO DE DATO | PERMITE  NULOS | DESCRIPCION | CONSTRAINT | DEFAULT |
| id | Int(11) | NO | Numero de Id de Transporte | PRIMARY | NINGUNA |
| nombre | Varchar(25) | NO | Nombre del Transporte |  | NINGUNA |
| email | Varchar(25) | SI | Correo Electrónico de Transporte |  | NULL |
| teléfono | Varchar(25) | SI | Teléfono de Transporte |  | NULL |
| costo | double | NO | Costo del Transporte |  | NINGUNA |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| venta | | | | | |
| NOMBRE DE COLUMNA | TIPO DE DATO | PERMITE  NULOS | DESCRIPCION | CONSTRAINT | DEFAULT |
| id | Int(11) | No | Numero de Id de Venta | PRIMARY | NINGUNA |
| total | double | Si | Número Total |  | NULL |
| fecha | datetime | Si | Fecha |  | NULL |
| estado | Varchar(10) | Si | Estado |  | NULL |
| Transporte\_id | Int(11) | No | Numero Id de Transporte | FK\_Venta\_Transporte | NINGUNA |
| Cliente\_id | Int(11) | No | Numero id del Cliente | FK\_Venta\_Cliente | NINGUNA |

# **9.2. DICCIONARIO DE DATOS DE SU BASE NOSQL**

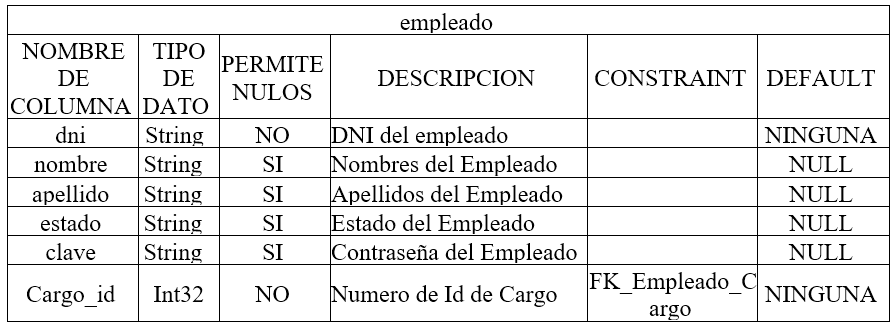
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nombre de la Colección:** | | | empleado | | | |  |
| **Descripción de la Colección:** | | | Esta colección se encarga de almacenar documentos con los datos de los empleados registrados. | | | |  |
| **Objetivo:** | | | Almacenar datos de los empleados. | | | |  |
| **Relaciones con otras Tablas:** | | | Posee una clave foránea de la colección cargo. | | | |  |
| **Descripción de los campos** | | | | | | |  |
| **Nro.** | **Nombre del campo** | **Tipo dato longitud** | **Permite nulos** | **Clave primaria** | **Clave foránea** | **Descripción del campo** |  |
|  |
| 1 | dni | String | NO |  |  | Dni del empleado. |  |
| 2 | nombre | String | NO |  |  | Nombre del empleado. |  |
| 3 | apellido | String | NO |  |  | Apellido del empleado. |  |
| 4 | estado | String | NO |  |  | Estado del empleado (Inactivo o Activo). |  |
| 5 | clave | String | NO |  |  | Contraseña del empleado para ingresar al sistema. |  |
| 6 | Cargo\_id | Int32 | NO |  | FK | Clave foránea de la colección cargo. |  |

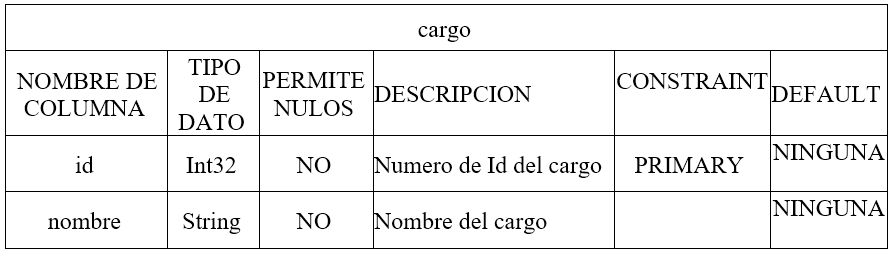
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nombre de la Colección:** | | | cargo | | | |  |
| **Descripción de la Colección:** | | | Esta colección se encarga de almacenar documentos con los datos de los cargos disponibles para los empleados. | | | |  |
| **Objetivo:** | | | Almacenar los cargos disponibles. | | | |  |
| **Relaciones con otras Tablas:** | | | Su id es clave foránea en la colección empleado. | | | |  |
| **Descripción de los campos** | | | | | | |  |
| **Nro.** | **Nombre del campo** | **Tipo dato longitud** | **Permite nulos** | **Clave primaria** | **Clave foránea** | **Descripción del campo** |  |
|  |
| 1 | id | Int32 | NO | PK |  | Id del documento. |  |
| 2 | nombre | String | NO |  |  | Nombre del cargo. |  |

# **9.2.1. NOMBRE DE BASE DE DATOS, DESCRIPCION, TIPO RELACIONAL**

|  |  |  |
| --- | --- | --- |
| **NOMBRE DE BASE DE DATOS** | **DESCRIPCION** | **TIPO RELACIONAL** |
| **electrodomesticos** | Base de Datos NoSQL dentro de mongodb, la cual almacena la colección empleado y cargo. | NoSQL |

**9.2.2. TABLAS CON LAS SIGUIENTES CAMPOS: NOMBRE DE COLUMNA, TIPO DE DATO, PERMITE NULOS, DESCRIPCION, CONSTRAINT, DEFAULT**





# **9.3. ESTANDAR DE PROGRAMACION**

**electrodomésticos**

# **1.** **Declaración de variables**

Se propone que la declaración de las variables, se ajusten al motivo para la que se requieran. El mnemotécnico definido se establece tomando en consideración principalmente lo siguiente:

* La longitud debe ser lo más recomendable posible. No debe ser tan grande de tal forma que el programador tenga la facilidad de manejo sobre la variable y ni tan corta que no pueda describirse claramente. Para el caso establecemos una longitud máxima de variable de 20 caracteres.
* Alcance de la variable

No tenemos un prefijo definido para cada variable en específico por lo que nuestra consideración la hacer una variable es que además de la relación que debe tener con la clase esta se describa a primera vista de que trata esa variable.

* El tipo de dato al que pertenece la variable.

Por lo tanto, la estructura de la variable es como sigue:

|  |  |
| --- | --- |
| **Estructura** | **Descripción de la Variable** |
| Longitud máx. | 🡨 1 🡪🡨 100 🡪 |
| Formato | *Dependiendo de los productos que desee comprar el cliente, o el número de clientes como tal.* |
| Ejemplo | Registro\_De\_Venta[] MT = new Registro\_De\_Venta[100];    Producto[] DA = new Producto[100]; |
| **Estructura** | **Descripción de la Variable** |
| Longitud máx. | 🡨 1 🡪🡨 100 🡪 |
| Formato | *Dependiendo de los productos que desee comprar el cliente, o el número de clientes como tal.* |
| Ejemplo | Registro\_De\_Empleado[] MT = new Registro\_De\_Empleado[100];    Empleado[] DA = new Empleado[100]; |

## **1.1** **Descripción de la Variable.**

Nombre que se le asignará a la variable para que se le identifique y deberá de estar asociada al motivo para la cual se le declara.

* nombre; private String, Registrar Nombre de Producto
* marca; private String, Registrar Marca de Producto
* etiqueta; private double, Registrar Etiqueta Producto
* fecha; private String, Registrar Fecha de Producto
* precio; private double, Registrar Precio de Producto
* cantidad; private double, Registrar Cantidad de Productos
* nombre\_cliente; private String, Registrar Nombre Cliente
* apellido\_cliente; private String, Registrar Apellido Cliente
* dni; protected String, Registrar DNI
* ID\_ListadoProducto; private String, Registrar ID Producto
* nproductos = 0; private Int , Registrar Productos
* Total\_De\_Productos = 0; protected double, Contar productos
* nombres; private String, Registrar Nombre de Empleado
* apellidos; private String, Registrar Apellidos de Empleado
* cargo; private String, Registrar Cargo de Empleado
* estado; private String, Registrar Estado de Empleado
* clave; private String, Registrar Clave de Empleado
* vrt; String,
* f=false; bool, Confirmación
* op,op1; Int , Variable de opciones múltiples
* n=0; Int, contador
* m=0; Int, contador

## **1.2** **Variables de Tipo Arreglo**

En el caso de las definiciones de arreglos de elementos se declarará la variable con el prefijo de “lista”, el cual nos dará entender que se trata de una variable del tipo arreglo la cual contendrá de cero a más datos, según el tamaño declarado.

* listaMT; public; Lista para registrar clientes
* listaDA; public; Lista para registrar ventas de los productos
* listamod; public; Lista para modificaciones de los registros de ventas
* listasecciones; private; Lista de los productos para calcularlos en base a la modificación, aumento o eliminación del registro del producto

* listaME; public; Lista para registrar empleados
* listamode; public; Lista para modificaciones de los empleados

# **2.** **Definición de Controles**

Para poder determinar el nombre de un control dentro de cualquier aplicación de tipo visual, se procede a identificar el tipo al cual pertenece y la función que cumple dentro de la aplicación.

## **2.1** **Tipo de datos**

|  |  |  |
| --- | --- | --- |
| **Tipo de variable** | **Mnemónico** | **Descripción** |
| Byte | by | Entero de 8 bits sin signo. |
| Integer | in | Entero de 32 bits con signo. |
| Char | ch | Un carácter UNICODE de 16 bits |
| String | st | Cadena de caracteres |
| Date | dt | Formato de fecha/hora |
| Boolean | bl | Valor lógico: verdadero y falso |
| Float | fl | Comas flotantes, 11-12 dígitos significativos. |
| Double | db | Coma flotante, 64 bits (15-16 dígitos significativos) |
| Object | ob | Objeto genérico |

## **2.2** **Declaración de variables, atributos y objetos**

1. Se debe declarar una variable por línea.

|  |  |
| --- | --- |
| **Nombre** | **Descripción** |
| **Sintaxis** | [String] [Nombre] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres.  El nombre de la variable puede incluir solo 2 nombres de la persona. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. |
| **Ejemplo** | Public String nombre  Indica una variable o atributo que guardará un nombre. |

|  |  |
| --- | --- |
| **Marca** | **Descripción** |
| **Sintaxis** | [String] [marca] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 20 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:  1. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. |
| **Ejemplo** | Public String marca  Indica una variable o atributo que guardará una marca para un producto. |

|  |  |
| --- | --- |
| **Etiqueta** | **Descripción** |
| **Sintaxis** | [double] [etiqueta] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, o, ú. |
| **Ejemplo** | Public double etiqueta  Indica una variable o atributo que guardará una etiqueta de un producto. |

|  |  |
| --- | --- |
| **Fecha** | **Descripción** |
| **Sintaxis** | [String] [fecha] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public String fecha  Indica una variable o atributo que guardará una fecha del producto. |

|  |  |
| --- | --- |
| **Precio** | **Descripción** |
| **Sintaxis** | [double] [precio] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public Double precio  Indica una variable o atributo que guardará un valor del precio del Producto. |

|  |  |
| --- | --- |
| **Cantidad** | **Descripción** |
| **Sintaxis** | [double] [cantidad] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public Double cantidad  Indica una variable o atributo que guardará la cantidad de Productos. |

|  |  |
| --- | --- |
| **Nombre del Cliente** | **Descripción** |
| **Sintaxis** | [String] [nombre\_cliente] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. |
| **Ejemplo** | Public String nombre\_cliente  Indica una variable o atributo que guardará un nombre de Cliente de los Productos. |

|  |  |
| --- | --- |
| **Apellido del Cliente** | **Descripción** |
| **Sintaxis** | [String] [apellido\_cliente] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. |
| **Ejemplo** | Public String apellido\_cliente  Indica una variable o atributo que guardará un apellido de Cliente de los Productos. |

|  |  |
| --- | --- |
| **DNI** | **Descripción** |
| **Sintaxis** | [String] [dni] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public String dni  Indica una variable o atributo que guardará un dni del Cliente de los Productos. |

|  |  |
| --- | --- |
| **ID del Listado del Producto** | **Descripción** |
| **Sintaxis** | [String] [ ID\_ListadoProducto] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public String ID\_ListadoProducto  Indica una variable o atributo que guardará un ID del listado de los Productos pertenecientes al Cliente. |

|  |  |
| --- | --- |
| **Cantidad de Productos** | **Descripción** |
| **Sintaxis** | [Int] [nproductos] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public Int nproductos  Indica una variable o atributo que guardará un numero de Productos registrados por el sistema. |

|  |  |
| --- | --- |
| **Totalidad de los Productos** | **Descripción** |
| **Sintaxis** | [double] [Total\_De\_Productos] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public double Total\_De\_Productos  Indica una variable o atributo que guardará la suma total de los Productos de la Lista del Cliente. |

|  |  |
| --- | --- |
| **Totalidad de los Productos** | **Descripción** |
| **Sintaxis** | [String] [nombres] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public String nombres  Indica una variable o atributo que guardará los nombres del empleado. |

|  |  |
| --- | --- |
| **Totalidad de los Productos** | **Descripción** |
| **Sintaxis** | [String] [apellidos] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public String apellidos  Indica una variable o atributo que guardará los apellidos del empleado. |

|  |  |
| --- | --- |
| **Totalidad de los Productos** | **Descripción** |
| **Sintaxis** | [String] [cargo] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public String cargo  Indica una variable o atributo que guardará el cargo del empleado. |

|  |  |
| --- | --- |
| **Totalidad de los Productos** | **Descripción** |
| **Sintaxis** | [String] [estado] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public String estado  Indica una variable o atributo que guardará el estado del empleado. |

|  |  |
| --- | --- |
| **Totalidad de los Productos** | **Descripción** |
| **Sintaxis** | [String] [clave] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Public String clave  Indica una variable o atributo que guardará la clave del empleado. |

|  |  |
| --- | --- |
| **Verificación del ID del Cliente** | **Descripción** |
| **Sintaxis** | [String] [vrt] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | String vrt  Indica la confirmación del ID del cliente. |

|  |  |
| --- | --- |
| **Comprobación del Id Cliente y De Productos** | **Descripción** |
| **Sintaxis** | [bool] [f] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | bool f=false  Verifica y Confirma el registro de clientes e Productos del Cliente. |

|  |  |
| --- | --- |
| **Opciones de Selección** | **Descripción** |
| **Sintaxis** | [Int] [op,op1] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Int op,op1  Indica la elección de las opciones que tienen en los menús. |

|  |  |
| --- | --- |
| **Contador Principal** | **Descripción** |
| **Sintaxis** | [Int] [n=0] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Int n=0  Indica el conteo de cada registro de cliente |

|  |  |
| --- | --- |
| **Contador Secundario** | **Descripción** |
| **Sintaxis** | [Int] [m=0] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Int m=0  Indica el conteo de cada registro de producto perteneciente al cliente. |

|  |  |
| --- | --- |
| **Listado de Registro de Clientes** | **Descripción** |
| **Sintaxis** | [public] [MT] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Registro\_De\_Venta[] MT = new Registro\_De\_Venta[100]  Almacena los datos de los clientes |

|  |  |
| --- | --- |
| **Listado de Registro de los Productos** | **Descripción** |
| **Sintaxis** | [public] [DA] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Producto[] DA = new Producto[100]  Almacena cantidad datos de los productos de cada lista de cada cliente |

|  |  |
| --- | --- |
| **Listado de las Modificaciones para los Productos** | **Descripción** |
| **Sintaxis** | [public] [mod] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Producto mod = new Producto()  Permite modificar los productos de una lista del cliente |

|  |  |
| --- | --- |
| **Listado de Productos Registrados para el Calculo** | **Descripción** |
| **Sintaxis** | [private] [secciones] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | private Producto[] secciones = new Producto[100]  Almacena datos de los cálculos de los productos |

## **2.3** **Declaración de clases**

|  |  |
| --- | --- |
| **Registro de Cliente** | **Descripción** |
| **Sintaxis** | [] Class [Registro\_De\_Venta] |
| **Descripción** | El nombre de las clases tendrá una longitud máxima de 30 caracteres y las primeras letras de todas las palabras estarán en mayúsculas. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Class Registro\_De\_Venta  Indica una clase Registro de Venta que hace que se registre la venta de un cliente a manos del usuario. |

|  |  |
| --- | --- |
| **Producto** | **Descripción** |
| **Sintaxis** | [] Class [Producto] |
| **Descripción** | El nombre de las clases tendrá una longitud máxima de 30 caracteres y las primeras letras de todas las palabras estarán en mayúsculas. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Class Producto  Indica una clase Producto para ser registrado por el usuario |

|  |  |
| --- | --- |
| **Producto** | **Descripción** |
| **Sintaxis** | [] Class [Registro\_De\_Empleado] |
| **Descripción** | El nombre de las clases tendrá una longitud máxima de 30 caracteres y las primeras letras de todas las palabras estarán en mayúsculas. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Class Registro\_De\_Empleado  Indica una clase empleado para ser registrado por el administrador |

|  |  |
| --- | --- |
| **Producto** | **Descripción** |
| **Sintaxis** | [] Class [Empleado] |
| **Descripción** | El nombre de las clases tendrá una longitud máxima de 30 caracteres y las primeras letras de todas las palabras estarán en mayúsculas. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Class Registro\_De\_Empleado  Indica una clase empleado para ser registrado por el administrador |

## **2.4** **Declaración de métodos**

|  |  |
| --- | --- |
| **Registro del Producto** | **Descripción** |
| **Sintaxis** | Agregar\_Empleado [(Vacio)] |
| **Descripción** | El nombre del método constará hasta de 25 caracteres.  La primera letra de la primera palabra del nombre será escrita en minúscula y las siguientes palabras empezarán con letra mayúscula. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void AgregarEmpleado()  Indica un método AgregarEmpleado, que recibe varias variables por valor de Tipo Int y String para el registro del producto. |

AgregarEmpleado()

Eliminar\_Empleado ()

Cambiar\_Estado ()

Listar\_Empleado ()

Buscar\_Empleado ()

|  |  |
| --- | --- |
| **Eliminación del Producto** | **Descripción** |
| **Sintaxis** | Eliminar\_Empleado [(int Id, String)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Eliminar\_Empleado (int Id, String)  Se incluye el método Eliminar para borrar el empleado. |

|  |  |
| --- | --- |
| **Cambiar Estado** | **Descripción** |
| **Sintaxis** | Cambiar\_Estado[(String)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Cambiar\_Estado (String)  Se incluye el método cambiar para modificar el estado del empleado. |

|  |  |
| --- | --- |
| **Listar Empleado** | **Descripción** |
| **Sintaxis** | Listar\_Empleado[(String)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Listar\_Empleado (Id, String)  Se incluye el método listar para mostrar los empleados registrados. |

|  |  |
| --- | --- |
| **Buscar Empleado** | **Descripción** |
| **Sintaxis** | Buscar\_Empleado[(Id, String)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Buscar\_Empleado (Id, String)  Se incluye el método buscar para encontrar el registro de un empleado. |

|  |  |
| --- | --- |
| **Registro del Producto** | **Descripción** |
| **Sintaxis** | AgregarProducto [(Vacio)] |
| **Descripción** | El nombre del método constará hasta de 25 caracteres.  La primera letra de la primera palabra del nombre será escrita en minúscula y las siguientes palabras empezarán con letra mayúscula. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void AgregarProducto()  Indica un método AgregarProducto, que recibe varias variables por valor de Tipo String, Int y double para el registro del producto. |

|  |  |
| --- | --- |
| **Cálculo del Precio** | **Descripción** |
| **Sintaxis** | CalcularPrecio [(Vacio)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void CalcularPrecio()  Indica un método CalcularPrecio que recibe una variable por valor de tipo double al cálculo del precio del producto |

|  |  |
| --- | --- |
| **Visualización del Producto** | **Descripción** |
| **Sintaxis** | mostrar\_Producto [(Vacio)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void mostrar\_Producto()  Indica un método mostrar\_Producto que recibe varias variables para visualizarlas en la pantalla dentro de un listado |

|  |  |
| --- | --- |
| **Totalidad del Precio** | **Descripción** |
| **Sintaxis** | Precio\_total[(Vacio)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public double Precio\_total()  Indica un método Precio\_total que recibe una variable por valor de tipo double al precio total de los productos |

|  |  |
| --- | --- |
| **Registro del Cliente** | **Descripción** |
| **Sintaxis** | Agregar\_Registro0 [(Vacio)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Agregar\_Registro0()  Indica un método Agregar\_Registro0 que recibe varias variables de Tipo String para el registro del cliente |

|  |  |
| --- | --- |
| **Visualización del Registro del Cliente** | **Descripción** |
| **Sintaxis** | Mostrar\_Registro0 [(Vacio)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Mostrar\_Registro0()  Indica un método Mostrar\_Registro0 que recibe las variables ingresadas del Agregar\_Registro0 para poder ser visualizadas en la pantalla. |

|  |  |
| --- | --- |
| **Identificación de lista de Productos del Cliente** | **Descripción** |
| **Sintaxis** | Agregar\_Registro [(Vacio)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Agregar\_Registro()  Indica un método Agregar\_Registro que recibe una variable por valor de tipo String para la confirmación del ID de listado de productos del Cliente. |

|  |  |
| --- | --- |
| **Visualización del Listado de los Productos y El Total de la Venta** | **Descripción** |
| **Sintaxis** | Mostrar\_Registro [(Vacio)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Mostrar\_Registro()  Indica un método Mostrar\_Registro que recibe una variable ingresada de tipo String junto con la variable double del precio total de los productos para ser visualizadas |

|  |  |
| --- | --- |
| **Visualización de los Productos del Listado** | **Descripción** |
| **Sintaxis** | mostrarProductos [(Vacio)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void mostrarProductos()  Indica un método mostrarProductos que recibe varias variables de tipo String y tipo double para ser mostradas en fila por cada producto registrado |

|  |  |
| --- | --- |
| **Cálculo de la Totalidad de los Productos** | **Descripción** |
| **Sintaxis** | Calcular\_Total [(double c)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Calcular\_Total (double c)  Indica un método Calcular\_Total que recibe una variable por valor de tipo double para sumar la cantidad del precio por cada producto registrado que este dentro de la lista |

|  |  |
| --- | --- |
| **Visualización del ID del Listado** | **Descripción** |
| **Sintaxis** | ID\_Mostrar [(Vacio)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public String ID\_Mostrar()  Indica el ID\_Mostrar, que devuelve el valor de tipo String para mostrar la identificación del listado de los productos del cliente |

|  |  |
| --- | --- |
| **Aumento de la cantidad de los Productos** | **Descripción** |
| **Sintaxis** | AdherirProducto [(Producto b)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void AdherirProducto(Producto b)  Indica un método AdherirProducto que recibe la variable arreglo para agregar más productos para ser almacenados |

|  |  |
| --- | --- |
| **Modificación de los Productos** | **Descripción** |
| **Sintaxis** | Modificar [(Producto s, int Idm, double f1)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Modificar (Producto s, int Idm, double f1)  Indica el método Modificar que recibe varias variables de tipo String y tipo double para modificar el producto halla dentro de la lista. |

|  |  |
| --- | --- |
| **Eliminación del Producto** | **Descripción** |
| **Sintaxis** | Eliminar [(int Idy, double f)] |
| **Descripción** | Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. |
| **Observaciones** | En la declaración de métodos no se deberá utilizar caracteres como:   1. Letra Ñ o ñ. 2. Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. 3. Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | public void Eliminar (int Idy, double f)  Se incluye el método Eliminar que se borrar(reemplazar) el producto y dejar vacío el producto eliminando y agregando otro producto más. |

## **2.5** **Control de versiones de código fuente**

Cada modificación realizada será guardada de la forma:

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Formato** | [Registro De Ventas][ \_ ][20210107][ \_ ][14:45:52]. |
| **Descripción** | Se generará archivo con la siguientes extension:.cs.  Proyecto\_Final.cs |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Formato** | [Registro De Empleados][ \_ ][20210107][ \_ ][14:45:52]. |
| **Descripción** | Se generará archivo con la siguientes extension:.cs.  Proyecto\_Final.cs |

# **3.** **Clases**.

El nombre de las clases no tiene prefijos para ser nombrados por que se guían de una lógica que tiene más concordancia con el tema así que no estamos usando los prefijos, ya que buscamos dar el nombre de la clase de manera más directa.

* Registro\_De\_Venta
* Producto
* Registro\_De\_Empleado
* Empleado

# **4.** **Métodos, procedimientos y funciones definidos por el Usuario.**

* AgregarProducto()
* CalcularPrecio()
* Listar\_Producto()
* Precio\_total()

* AgregarEmpleado()
* Eliminar\_Empleado ()
* Cambiar\_Estado ()
* Listar\_Empleado ()
* Buscar\_Empleado ()

* Agregar\_Registro0()
* Mostrar\_Registro0()
* Mostrar\_Registro\_Cliente()
* Agregar\_Registro()
* Mostrar\_Registro()
* ListarProductos()
* Buscar\_Productos ()
* Calcular\_Total(double c)
* ID\_Mostrar()
* AdherirProducto(Producto b)
* Modificar (Producto s, int Idm, double f1)
* Eliminar (int Idy, double f)

# **5.** **Beneficios**

1. La documentación siempre es importancia para la mayor compresión del Usuario, Programador y los que se ocupan del Mantenimiento.
2. Si se tiene una documentación bien elaborada, no habrá pierde en caso se necesite mantenimiento al sistema.
3. Ayuda a ver ciertos errores que podría presentar el código, dado a que al analizar todo al detalle, nuestro campo de visión aumenta significativamente.

# **6.** **Conclusiones**

1. Documentar un programa, es sumamente importante, dado a que ayuda a detectar ciertas fallas que podría presentar el programa que se está desarrollando.
2. El manejo de las variables se debe de trabajarse con mucha cautela, dado a que son eje principal de todo el sistema.