Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación

Sistema de Consulta de Medicamentos en Farmacia

TESIS

PARA OBTENER EL TITULO DE:
LIC. EN CIENCIAS DE LA COMPUTACIÓN

Presenta:

TOLEDO MATUS VÍCTOR MANUEL

Asesor:
M.C. Pedro Bello López

Puebla, Pue., Agosto de 2007.

AGRADECIMIENTOS.

A mis padres Victor (D.E.P.) y Teresa por su gran y valioso esfuerzo por darme la oportunidad de tener una formación profesional, de igual forma por el estimulo de seguir adelante. Mi infinita gratitud.

A mi hermano Gilberto por su apoyo constante en estos últimos años. Gracias.

A mis demás hermanos y hermanas que de igual forma contribuyeron y confiaron en mí para llegar hasta esta instancia, así como también por su valioso apoyo. A cada uno de ellos mil gracias por todo lo que han hecho por mí.

A mis amigos Alejandro, Daniel, Israel y Santiago por estar conmigo en los momentos agradables y no gratos de mi vida. Gracias.

Al M.C. Pedro Bello López por su valiosa colaboración como asesor de tesis. A usted muchas gracias.

INDICE

CAPÍTULO 3. DISEÑO DEL SISTEMA.

CAPÍTULO 4. IMPLEMENTACIÓN Y PRUEBAS DEL SISTEMA.

INTRODUCCIÓN.

Las farmacias importantes del país cuentan cada una con su propio sitio Web, el cual se centra solo en proporcionar información sobre ellas, y en algunos de los casos información de ciertos medicamentos que se encuentran en promoción de manera muy general.

También las farmacias cuentan con sistemas de manera local, estos se centran en llevar el control de inventarios, ventas y compras; por tanto la información de los medicamentos es de igual forma muy general, para consultar un medicamento lo hacen de forma manual en un diccionario de medicamentos.

En el presente trabajo se presenta el desarrollo de un Sistema Web de consulta de medicamentos en una farmacia, basado en la arquitectura cliente servidor y la creación de una Base de Datos Relacional que tendrá toda la información necesaria de los medicamentos para su posterior consulta.

Para el desarrollo de este trabajo se presenta el contenido temático, en el cual se da una breve descripción de lo que se abarca en cada capítulo.

En el **capítulo 1**, se hace referencia a los temas que son de gran importancia para la elaboración y resolución del problema que se plantea. En primer lugar se hace mención de conceptos de Ingeniería de Software que indican como construir técnicamente el Software.
Se presentan los principales conceptos para la creación y gestión de Base de Datos como lo es el modelo Entidad – Relación y el modelo Relacional.
Por último se da una breve explicación de las herramientas que se utilizaron para el desarrollo del sistema como PHP y MySQL.

En el **capítulo 2**, se abordan los antecedentes que dieron origen al desarrollo del sistema, se realiza la especificación de los requisitos y la descripción de la información mediante diagramas de caso de uso y flujo de eventos.

En el **capítulo 3**, se realiza el diseño de la Base de Datos en sus tres fases: diseño conceptual, lógico y físico que empleará el "Sistema de Consulta de Medicamentos en Farmacia", con el fin de almacenar la información correspondiente.

En el **capítulo 4**, se presenta la implementación del sistema mediante interfaces, se explica la función de cada una de ellas y los elementos que la componen, así como el flujo que debe seguir cada usuario.
Además se aborda en este mismo capítulo las pruebas realizadas a cada interfaz del sistema, se muestran los resultados de aceptación y errores en caso de que los datos no sean correctos o válidos.

De esta manera, se desarrolló el "Sistema de Consulta de Medicamentos en Farmacia", que será de gran utilidad para las farmacias, así como también para los clientes de la misma que estén interesados en adquirir un producto.

CAPITULO 1. MARCO TEÓRICO.

1.1.Ingeniería de Software.

Este término de Ingeniería de Software fue introducido a finales de los 60 a raíz de la crisis del software, a continuación se da una definición de ello.

Según Bauer, la Ingeniería del Software es el establecimiento y uso de principios robustos de la ingeniería, a fin de obtener económicamente Software que sea fiable y que funcione eficientemente sobre máquinas reales [6].

Esta definición proporciona una serie de cuestiones. ¿Cuáles son los principios robustos de la ingeniería aplicables al desarrollo de software?, ¿Cómo construimos el software económicamente para que sea fiable?, ¿Qué se necesita para crear programas que funcionen eficientemente no en una máquina si no en diferentes máquinas reales? Éstas son cuestiones que los ingenieros del software tienen como reto.

1.1.1 Proceso, métodos y herramientas.

Según Pressman, la Ingeniería del Software es una tecnología multicapa, la cual esta constituida por un enfoque de calidad, proceso, métodos y herramientas.

El fundamento de la Ingeniería del Software es la capa de proceso, el cual es la unión que mantiene juntas las capas de tecnología y que permite un desarrollo racional y oportuno de la Ingeniería del Software. Define un marco de trabajo para un conjunto de áreas de proceso que se deben establecer para la entrega efectiva de la Ingeniería del Software. Las áreas claves del proceso forman la base del control de gestión de proyectos del software y establecen el contexto en el que se aplican los métodos técnicos, se obtienen productos del trabajo, se establecen hitos, se asegura la calidad y el cambio se gestiona adecuadamente.

Los métodos de la Ingeniería del Software indican cómo construir técnicamente el software. Los métodos abarcan una gran gama de tareas que incluyen análisis de requisitos, diseño, construcción de programas, pruebas y mantenimiento.

Las herramientas de la Ingeniería del Software proporcionan un enfoque automático o semiautomático para el proceso y para los métodos. Cuando se integran herramientas para que la información creada por una herramienta la pueda utilizar otra, se establece un sistema de soporte para el desarrollo del software llamado Ingeniería del Software asistida por computadora (CASE).

1.1.2 Visión general de la Ingeniería del Software.

La ingeniería es el análisis, diseño, construcción, verificación y gestión de entidades técnicas [6].

Con independencia de la entidad a la que se va a aplicar ingeniería, se deben cuestionar y responder las siguientes preguntas:

- ¿Cuál es el problema a resolver?
- ¿Cuáles son las características de la entidad que se utiliza para resolver el problema?
- ¿Cómo se realiza la entidad?
- ¿Cómo se construirá la entidad?
- ¿Qué enfoque se va a utilizar para no contemplar los errores que se cometieron en el diseño y en la construcción de la entidad?
- ¿Cómo se apoyará la entidad cuando usuarios soliciten correcciones, adaptaciones de la entidad?

El trabajo que se asocia a la Ingeniería del Software se puede dividir en tres fases genéricas, con independencia del área de aplicación, tamaño o complejidad del proyecto. Cada fase se encuentra con una o varias cuestiones de las mencionadas anteriormente.

La fase de definición: se centra sobre el "qué". Es decir, durante la definición, el que desarrolla el software intenta identificar qué información ha de ser procesada, que función y rendimiento se desea, qué comportamiento del sistema, qué interfaces van a ser establecidas, qué restricciones de diseño existen, y que criterios de validación se necesitan para definir un sistema correcto. Por tanto, han de identificarse los requisitos clave del sistema y del software.

La fase de desarrollo: se centra en el "cómo". Es decir, durante el desarrollo un ingeniero del software intenta definir cómo han de diseñarse las estructuras de datos, cómo ha de implementarse la función dentro de una arquitectura de software, como han de implementarse lo detalles procedimentales, cómo han de caracterizarse interfaces, cómo ha de traducirse el diseño en un lenguaje de programación y como ha de realizarse la prueba.

La fase de mantenimiento se centra en el "cambio" que va asociado a la corrección de errores, a las adaptaciones requeridas a medida que evoluciona el entorno del software y a cambios debidos a las mejoras producidas por los requisitos cambiantes del cliente.
Durante la fase de mantenimiento se encuentran cuatro tipos de cambios:

- **Corrección**. El mantenimiento correctivo cambia el software para corregir los defectos.
- **Adaptación.** El mantenimiento adaptativo produce modificación en el software para acomodarlo en los cambios de su entorno externo.

- **Mejora.** El mantenimiento perfectivo lleva al software más allá de sus requisitos funcionales originales.

- **Prevención.** En esencia, este mantenimiento hace cambios en programas de computadora a fin de que se puedan corregir, adaptar y mejorar más fácilmente.

1.1.3 El proceso y los modelos de proceso del software.

Un proceso de software se puede caracterizar estableciendo un marco común del proceso definiendo un pequeño número de actividades del marco de trabajo que son aplicables a todos los proyectos del software, con independencia de su tamaño o complejidad. Un número conjunto de tareas, que permiten que las actividades del marco de trabajo se adapten a las características del proyecto de software y a los requisitos del equipo del proyecto. Finalmente, las actividades de protección abarcan el modelo de procesos. Las actividades de protección son independientes de cualquier actividad del marco de trabajo y aparecen durante todo el proceso.

Para resolver los problemas, un ingeniero debe incorporar una estrategia de desarrollo que acompañe al proceso, métodos y capas de herramientas y las fases genéricas. Esta estrategia a menudo se llama modelo de proceso o paradigma de Ingeniería del Software. Se selecciona un modelo de proceso para la Ingeniería del Software según la naturaleza del proyecto y de la aplicación, lo métodos y las herramientas a utilizarse, y los controles y entregas que se requieren.

En la figura 1.1 se muestra como todo el desarrollo del software se puede caracterizar como bucle de resolución de problemas en el que se encuentran cuatro etapas distintas: “estado actual”, definición de problemas, desarrollo técnico e integración de soluciones. Estado actual, representa el estado actual de sucesos; la definición de problemas identifica el problema específico a resolver; el desarrollo técnico resuelve el problema a través de la aplicación de una tecnología y la integración de soluciones ofrece los resultados a los que solicitan la solución en primer lugar.

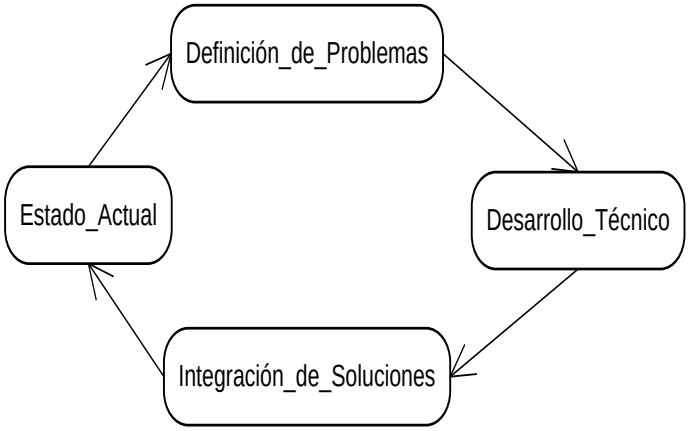


Figura 1.1 Las fases de un bucle de resolución de problemas.

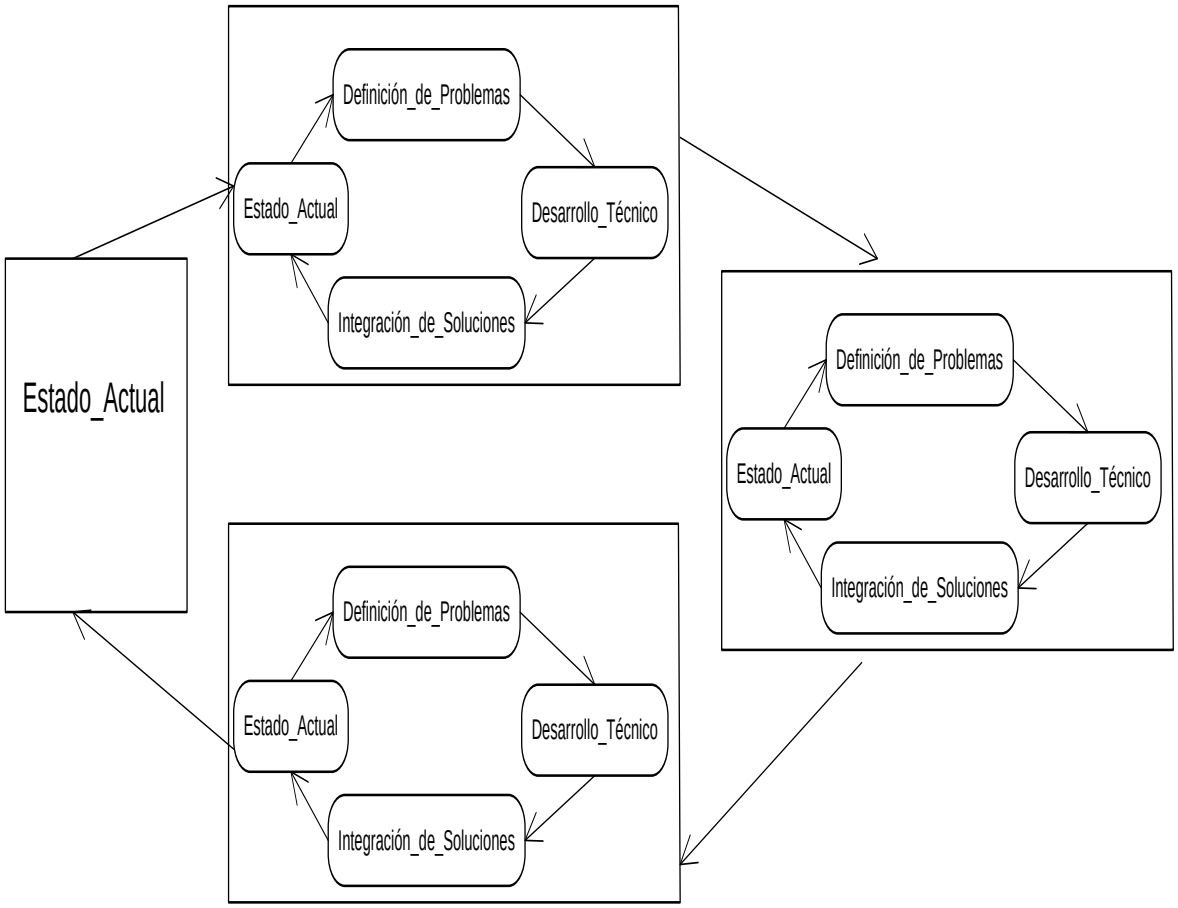


Figura 1.2 Fases dentro de las fases del bucle de resolución de problema.

De igual forma en la figura 1.2 se ve como se puede considerar como una aplicación recursiva del bucle sobre sí mismo, lo cual es llamado comúnmente modelo del caos, que describe el desarrollo del software como una extensión desde el usuario hasta el desarrollador y la tecnología. Conforme progresa el trabajo hacia un sistema completo, las etapas descritas anteriormente se aplican recursivamente a las necesidades del usuario y a la especificación técnica del desarrollador del software.

1.1.4 El modelo lineal secuencial.

Llamado algunas veces ciclo de vida clásico o modelo en cascada, el modelo lineal secuencial (figura 1.3) sugiere un enfoque sistemático, secuencial, para el desarrollo del software que comienza en un nivel de sistemas y progresa con el análisis, diseño, codificación, pruebas y mantenimiento. Este modelo comprende las siguientes actividades [6]:

Ingeniería y modelado de sistemas/Información. El trabajo comienza estableciendo requisitos de todos los elementos del sistema y asignando al software algún subgrupo de

estos requisitos. Esta visión del sistema es esencial cuando el software se debe interconectar con otros elementos como hardware, personas y bases de datos. La ingeniería y el análisis de sistemas comprenden los requisitos que se recogen en el nivel del sistema con una pequeña parte de análisis y de diseño. La ingeniería de información abarca los requisitos que se recogen en el nivel de empresa estratégico y en el nivel del área de negocio.

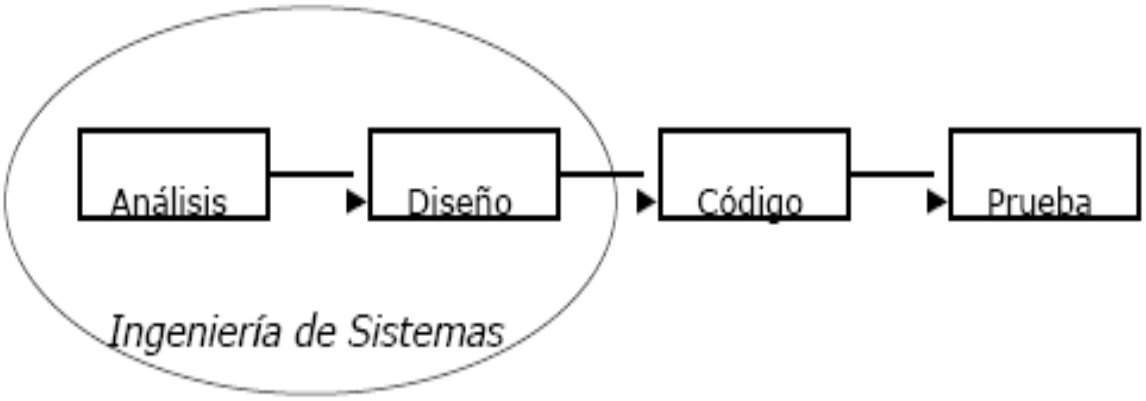


Figura 1.3 El modelo lineal secuencial.

Análisis de los requisitos del software. Es la fase en la cual se reúnen todos los requisitos que debe cumplir el software. Para comprender la naturaleza de los programas a construirse, el ingeniero del software debe comprender el dominio de información del software, así como la función requerida, comportamiento, rendimiento e interconexión.
En esta tapa es fundamental la presencia del cliente que documenta y repasa dichos requisitos.

Diseño. Es una etapa dirigida hacia la estructura de datos, la arquitectura del software, las representaciones de la interfaz y el detalle procedimental (algoritmo). El proceso de diseño traduce requisitos en una representación del software donde se puede evaluar su calidad antes de que comience la codificación.

Generación de código. Es la etapa en la cual se traduce el diseño en forma legible para que sea comprensible por la máquina. Esta etapa va a depender estrechamente de lo detallado del diseño. Es decir si se lleva a cabo el diseño de una forma detallada, la generación de código se realizará mecánicamente.

Pruebas. Una vez que se ha generado el código, comienzan las pruebas del programa, esta etapa se centra en los procesos lógicos internos del software, asegurando que todas las sentencias se han comprobado, y en la detección de errores, asegurando que la entrada definida produce resultados reales de acuerdo con los resultados requeridos.

Mantenimiento. Debido a que el programa puede tener errores, puede no ser del completo agrado del cliente o puede necesitar, eventualmente acoplarse a los cambios en su entorno.

Esto quiere decir que no se rehace el programa, sino que sobre la base de uno ya existente se realizan algunos cambios.

Observaciones:

- Los proyectos reales generalmente no son secuenciales.
- Es difícil que inicialmente se tengan definidos claramente los requisitos.
- Se requiere que el cliente sea paciente.
- El modelo puede provocar estados de bloqueo. [2]

1.2 Base de datos.

Un sistema de gestión de bases de datos (SGBD) consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dicho datos. La colección de datos, normalmente denominada base de datos, contiene información acerca de una empresa particular. El primer objetivo de un SGBD es proporcionar un entorno que sea tanto práctico como eficiente de usar en la recuperación y el almacenamiento de la información de la base de datos [1].

Los sistemas de base de datos se diseñan para gestionar grandes cantidades de información. La gestión de los datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información. En suma, los sistemas de base de datos deben proporcionar la fiabilidad de la información almacenada, a pesar de las caídas del sistema o los intentos de acceso sin autorización.

1.2.1 Propósitos de los sistemas de base de datos.

Los objetivos principales de un sistema de base de datos es disminuir los siguientes aspectos:

- **Redundancia e inconsistencia de datos.** Puesto que los archivos que mantienen almacenada la información son creados por diferentes tipos de programas de aplicación existe la posibilidad de que si no se controla detalladamente el almacenamiento, se pueda originar un duplicado de información. Esto aumenta los costos de almacenamiento y acceso a los datos, además de que puede originar la inconsistencia de los datos.

- **Dificultad en el acceso a los datos.** Un sistema de base de datos debe contemplar un entorno de datos que le facilite al usuario el manejo de los mismos.

- **Aislamiento de datos.** Puesto que los datos están repartidos en varios archivos, y estos no pueden tener diferentes formatos, es difícil escribir nuevos programas de aplicación para obtener los datos apropiados.

- **Problemas de integridad.** Los valores de los datos almacenados en la base de datos deben satisfacer cierto tipo de restricciones de consistencia. Estas restricciones se hacen cumplir en el sistema añadiendo códigos apropiados en los diversos programas de aplicación.

- **Problemas de atomicidad.** Un sistema de una computadora está sujeto a fallo. En muchas aplicaciones es crucial asegurar que una vez que un fallo ha ocurrido y se ha detectado, los datos se restauran al estado de consistencia que existía antes del fallo.

- **Anomalías en el acceso concurrente.** Para mejorar el funcionamiento global del sistema y obtener un tiempo de respuesta más rápido, muchos sistemas permiten que múltiples usuarios actualicen los datos simultáneamente. En un entorno así la interacción de actualizaciones concurrentes puede dar por resultado datos inconsistentes. Para prevenir esta posibilidad debe mantenerse alguna forma de supervisión en el sistema.

- **Problemas de seguridad.** Para que un sistema de base de datos sea confiable debe mantener un grado de seguridad que garantice la autentificación y protección de los datos.

1.2.2 Abstracción de datos.

Para que el sistema sea útil, debe recuperar los datos eficientemente. Esta recuperación ha conducido al diseño de estructuras de datos complejas para la representación de los datos en la base de datos. Los desarrolladores esconden la complejidad a los usuarios a través de varios niveles de abstracción para simplificar la interacción de los usuarios con el sistema:

Nivel físico. El nivel más bajo de abstracción describe cómo se almacenan realmente los datos.

Nivel lógico. El siguiente nivel más alto de abstracción describe qué datos se almacenan en la base de datos y qué relaciones existen entre esos datos.

Nivel de vistas. El nivel más alto de la abstracción describe sólo parte de la base de datos completa. El sistema puede proporcionar muchas vistas para la misma base de datos.

La interrelación entre estos tres niveles de abstracción se ilustra en la figura 1.4.

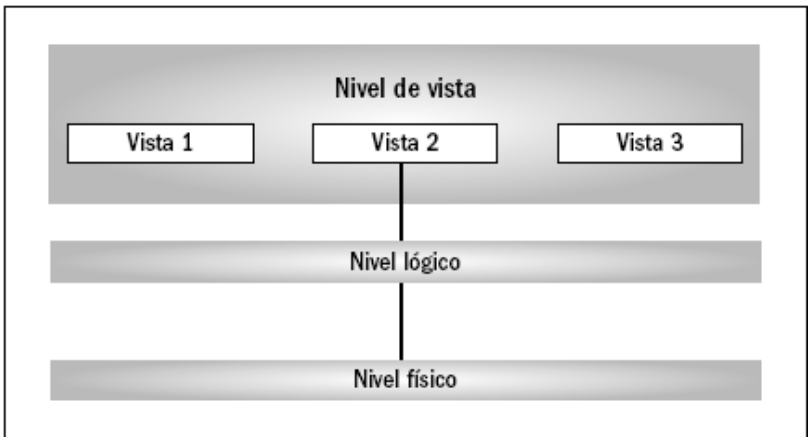


Figura 1.4 Los tres niveles de abstracción de datos.

En el nivel físico, un registro se puede describir como un bloque de posiciones almacenadas consecutivamente (por ejemplo, palabras o bytes). El compilador del lenguaje esconde este nivel de detalle para los programadores.

En el nivel lógico cada registro de este tipo se describe mediante una definición de tipo, y se define la relación entre estos tipos de registros. Los programadores, cuando usan un lenguaje de programación, trabajan en este nivel de abstracción. En forma análoga, los administradores habitualmente trabajan en este nivel de abstracción.

Finalmente, en el nivel de vistas, los usuarios de computadoras ven un conjunto de programas de aplicación que esconden los detalles de los tipos de datos. Análogamente, en el nivel de vistas se definen varias vistas de una base de datos, y los usuarios de la base de datos ven esas vistas. Además de esconder detalles del nivel lógico de la base de datos, las vistas también proporcionan un mecanismo de seguridad para evitar que los usuarios accedan a partes de la base de datos.

1.2.3 Ejemplares y esquemas.

La colección de información almacenada en la base de datos en un momento particular se llama un ejemplar de la base de datos. El diseño completo de la base de datos se llama esquema de la base de datos. Los esquemas son raramente modificados, si es que lo son alguna vez.

Un esquema de base de datos corresponde a una definición de tipo en un lenguaje de programación. Una variable de un tipo dado tiene un valor particular en un instante de tiempo. Así, el valor de una variable en lenguajes de programación corresponde a un ejemplar de un esquema de bases de datos.

Los sistemas de bases de datos tienen varios esquemas divididos, de acuerdo con los niveles de abstracción que se han discutido. En el nivel más bajo, está el esquema físico; en

el nivel intermedio está el esquema lógico, y en el nivel más alto está el subesquema. En general, los sistemas de bases de datos soportan un esquema físico, un esquema lógico y varios subesquemas.

1.2.4 Independencia de datos.

Se refiere a la capacidad para modifica una definición de esquemas en un nivel sin que afecte a una definición de esquema en el siguiente nivel más alto. Existen dos niveles de independencia de datos:

Independencia física de datos: Es la capacidad de modificar el esquema físico sin provocar que se vuelvan a escribir los programas de aplicación.

Independencia lógica de datos: Capacidad de modificar el esquema lógico sin provocar que se vuelvan a escribir los programas de aplicación.

El concepto de independencia de datos es similar en muchos aspectos al concepto de tipos abstractos de datos en los lenguajes de programación modernos. Ambos esconden los detalles de implementación a los usuarios para permitirles concentrarse en la estructura general, más que en los detalles de implementación de nivel más bajo.

1.2.5 Modelos de datos.

La parte esencial de la estructura de base de datos es el modelo de datos, el cual es una colección de herramientas conceptuales para describir los datos, las relaciones de datos, la semántica de datos y las ligaduras de consistencia [4].

Los modelos de datos que se describen se clasifican en tres grupos:

a) Modelos lógicos basados en objetos.
b) Modelos lógicos basados en registros.
c) Modelos físicos de datos.

1.2.5.1 Modelos lógicos basados en objetos.

Se usan para describir datos en los niveles lógico y de vistas. Tienen una capacidad de estructuración bastante flexible y permiten especificar restricciones de datos explícitamente. Existen diferentes modelos de este tipo, pero el más utilizado por su sencillez y eficiencia es el modelo Entidad-Relación [4].

Modelo entidad-relación.

El modelo de datos entidad-relación (E-R) está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados entidades, y de relaciones entre estos objetos llamados atributos [4].

Hay tres nociones básicas que emplea el modelo de datos E-R: conjuntos de entidades, conjuntos de relaciones y atributos.

Conjunto de entidades.

Una entidad es una "cosa" u "objeto" en el mundo real que es distinguible de todos los demás objetos. Una entidad tiene un conjunto de propiedades, y los valores para algún conjunto de propiedades pueden identificar una entidad de forma unívoca. Una entidad puede ser concreta, como una persona o un libro, o puede ser abstracta, como un préstamo, unas vacaciones o un concepto.

Un conjunto de entidades es un conjunto de entidades del mismo tipo que comparten las mismas propiedades, o atributos. Las entidades individuales que constituyen un conjunto se llaman la extensión del conjunto de entidades.

Una entidad se representa mediante un conjunto de atributos. Los atributos describen propiedades que posee cada miembro de un conjunto de entidades. La designación de un atributo para un conjunto de entidades expresa que la base de datos almacena información similar concerniente a cada entidad del conjunto de entidades; sin embargo, cada entidad puede tener su propio valor para cada atributo.

Cada entidad tiene un valor para cada uno de sus atributos. Para cada atributo hay un conjunto de valores permitidos, llamados el dominio, o el conjunto de valores, de ese atributo.

Una base de datos incluye así una colección de conjunto de entidades, cada una de las cuales contiene un número de entidades del mismo tipo.

Formalmente, un atributo de un conjunto de entidades es una función que asigna al conjunto de entidades un dominio. Como un conjunto de entidades puede tener diferentes atributos, cada entidad se puede describir como un conjunto de pares (atributo, valor), un par para cada atributo del conjunto de entidades. Los valores de los atributos que describen una entidad constituirán una porción significante de los datos almacenados en la base de datos.

Un atributo, como se usa en el modelo E-R, se puede caracterizar por los siguientes tipos de atributo:

- **Atributos simples y compuestos.** Los atributos simples no están divididos en subpartes. Los atributos compuestos, en cambio, se pueden dividir en subpartes (es decir, en otros atributos).

- **Atributos monovalorados y multivalorados.** Los atributos monovalorados tienen un solo valor para una entidad concreta. Los atributos multivalorados tienen un conjunto de valores para una entidad específica.

- **Atributos derivados.** El valor para este tipo de atributo se puede derivar de los valores de otros atributos o entidades relacionados. El valor de un atributo derivado no se almacena, sino que se calcula cuando sea necesario.

Un atributo toma un valor nulo cuando una entidad no tiene un valor para un atributo. El valor nulo también puede indicar "no aplicable", es decir, que el valor no existe para la entidad.

Conjuntos de relaciones.

Una relación es una asociación entre diferentes entidades. Un conjunto de relaciones es un conjunto de relaciones del mismo tipo.

La asociación entre conjuntos de entidades se conoce como participación; es decir, los conjuntos E_1, E_2,..., E_n participan en el conjunto de relaciones R. Un ejemplar de relación en un esquema E-R representa que existe una asociación entre las entidades denominadas en la empresa del mundo real que se modela.

La función que se desempeña una entidad en una relación se llama papel de la entidad. Debido a que los conjuntos de entidades que participan en un conjunto de relaciones son generalmente distintos, los papeles están implícitos y no se especifican normalmente. Sin embargo, son útiles cuando el significado de una relación necesita aclaración.

Restricciones.

Un esquema de desarrollo E-R puede definir ciertas restricciones a las que los contenidos de la base de datos se deben adaptar.

Dos de los tipos más importantes de restricciones son:

a) Correspondencia de cardinalidades.
b) Restricciones de participación.

Correspondencia de cardinalidades.

La correspondencia de cardinalidades expresa el número de entidades a las que otra entidad puede estar asociada vía un conjunto de relaciones.

La correspondencia de cardinalidades es la más útil describiendo conjuntos de relaciones binarias, aunque ocasionalmente contribuye a la descripción de conjunto de relaciones que implica más de dos conjuntos de entidades.

Para un conjunto de relaciones binarias R entre los conjuntos de entidades A y B, la correspondencia de cardinalidades debe ser una de las siguientes:

- **Uno a uno.** Una entidad en A se asocia con a lo sumo una entidad en B, y viceversa.

- **Uno a varios.** Una entidad en A se asocia con cualquier número de entidades en B (ninguna o varias). Una entidad en B, sin embargo, se puede asociar con a lo sumo una entidad en A.

- **Varios a uno.** Una entidad en A se asocia con a lo sumo una entidad en B, una entidad en B, sin embargo, se puede asociar con cualquier número de entidades (ninguna o varias) en A.

- **Varios a varios.** Una entidad en A se asocia con cualquier número de entidades (ninguna o varias) en B, y una entidad en B se asocia con cualquier número de entidades (ninguna o varias) en A.

Restricciones de participación.

La participación de un conjunto de entidades E en un conjunto de relaciones R se dice que es total si cada entidad en E participa al menos en una relación en R. Si sólo algunas entidades en E participan en relaciones en R, la participación del conjunto de entidades E en la relación R se llama parcial.

Claves.

Una clave permite identificar un conjunto de atributos suficiente para distinguir las entidades entre sí. Las claves también ayudan a identificar unívocamente a las relaciones y así distinguir las relaciones entre sí.

Una **superclave** es un conjunto de uno o más atributos que, tomados colectivamente, permiten identificar de forma única una entidad en el conjunto de entidades.

A menudo interesan las superclaves que los subconjuntos propios de ellas no son superclave. Tales superclaves mínimas se llaman **claves candidatas.**

Una **clave primaria** denota una clave candidata que es elegida por el diseñador de la base de datos como elemento principal para identificar las entidades dentro de un conjunto de entidades.

Diagrama Entidad – Relación.

La totalidad de estructuras lógicas de una base de datos se pueden expresar gráficamente mediante un diagrama E-R, que consta de los siguientes componentes principales:

- **Rectángulos,** que representan conjuntos de entidades.
- **Elipses,** que representan atributos.
- **Rombos,** que representan relaciones entre conjuntos de entidades.
- **Líneas,** que unen los atributos con los conjuntos de entidades y los conjuntos de entidades con las relaciones.
- **Elipses dobles,** que representan atributos multivalorados.
- **Elipses discontinuas,** que denotan atributos derivados.
- **Líneas dobles,** que indican participación total de una entidad en un conjunto de relaciones.
- **Rectángulos dobles,** que representan conjuntos de entidades débiles.

Cada componente se etiqueta con la entidad o relación que representa.
Además de entidades y relaciones, el modelo E-R representa ciertas restricciones que los contenidos de la base de datos deben cumplir. Una restricción importante es la correspondencia de cardinalidades, que expresa el número de entidades con las que otra entidad se puede asociar a través de un conjunto de relaciones.

1.2.5.2 Modelos lógicos basados en registros.

Se usan para describir datos en los niveles lógico y de vistas. Se usan tanto para especificar la estructura lógica completa de la base de datos como para proporcionar una descripción de alto nivel de la implementación [4].

Este tipo de modelos se llaman así debido a que la base de datos se estructura en registros de formato fijo de diferentes tipos.

Los tres modelos basados en registros más ampliamente aceptados son:

- Modelo Relacional.
- Modelo de Red.
- Modelo Jerárquico.

Modelo relacional.

En este modelo se representan los datos y las relaciones entre estos, a través de una colección de tablas, en las cuales los renglones (tuplas) equivalen a cada uno de los registros que contendrá la base de datos y las columnas corresponden a las características (atributos) de cada registro localizado en la tupla. Para cada atributo hay un conjunto de valores permitidos, llamados dominio de ese atributo [4].

Existen dos formas de representar la relación en este modelo; para ello se define el siguiente concepto.

Llave primaria: Es un atributo el cual definimos como atributo principal, es una forma única de identificar a una entidad.

Ahora bien, las formas de representar las relaciones en este modelo son:

1. Haciendo una tabla que contenga cada una de las llaves primarias de las entidades involucradas en la relación.

2. Incluyendo en alguna de las tablas de las entidades involucradas, la llave de la otra tabla.

Algebra relacional.

El algebra relacional es un lenguaje de consulta procedimental. Consta de un conjunto de operaciones que toman como entrada una o dos relaciones y producen como resultado una nueva relación. Las operaciones son unión, intersección, diferencia, producto, selección, proyección, reunión, división y asignación.

Las cuatro primeras se toman de la teoría de conjunto de las matemáticas; las cuatro siguientes son operaciones propias del álgebra relacional y la última es la operación estándar de dar un valor a un elemento.

- **Unión.** La operación de unión permite combinar datos de varias relaciones.

- **Intersección.** La operación de intersección permite identificar filas que son comunes en dos relaciones.

- **Diferencia.** La operación diferencia permite identificar filas que están en una relación y no en otra.

- **Producto.** La operación producto consiste en la realización de un producto cartesiano entre dos tablas dando como resultado todas las posibles combinaciones entre los registros de la primera y los registros de la segunda.

- **Selección.** La operación selección consiste en recuperar un conjunto de registros de una tabla o de una relación indicando las condiciones que deben cumplir los registros recuperados, de tal forma que los registros devueltos por la selección han de satisfacer todas las condiciones que se hayan establecido. Esta operación es la que normalmente se conoce como consulta.

- **Proyección.** Una proyección es un caso concreto de la operación selección, esta última devuelve todos los campos de aquellos registros que cumplen la condición que he establecido. Una proyección es una selección en la que seleccionamos aquellos campos que deseamos recuperar.

- **Reunión**. La reunión se utiliza para recuperar datos a través de varias tablas conectadas unas con otras mediante cláusulas JOIN, en cualquiera de sus tres variantes INNER, LEFT, RIGHT. La operación reunión se puede combinar con las operaciones selección y proyección.

- **División.** La operación división es la contraria a la operación producto.

- **Asignación.** Esta operación algebraica consiste en asignar un valor a uno o varios campos de una tabla.

Modelo de red.

Este modelo representa los datos mediante colecciones de registros y sus relaciones se representan por medio de ligas o enlaces, los cuales pueden verse como punteros. Los registros en la base de datos se organizan como colecciones de grafos dirigidos.

Modelo jerárquico.

Es similar al modelo de red en cuanto a las relaciones y datos, ya que estos se representan por medio de registros y enlaces. La diferencia radica en que los registros se organizan como colecciones de árboles en lugar de grafos dirigidos.

1.2.5.3 Modelo de datos físico.

Se usan para describir a los datos en el nivel más bajo, aunque existen muy pocos modelos de este tipo, básicamente capturan aspectos de la implementación de los sistemas de base de datos. Existen dos clasificaciones de este tipo que son:

- Modelo de unificación.
- Modelo de memoria por marcos [4].

1.2.6 Lenguajes de base de datos.

Un sistema de base de datos proporciona un lenguaje de definición de datos para especificar el esquema de la base de datos y un lenguaje de manipulación de datos para expresar las consultas y las modificaciones a la base de datos [1].

Lenguaje de definición de datos.

Un esquema de base de datos se especifica mediante un conjunto de definiciones expresadas mediante un lenguaje especial llamado lenguaje de definición de datos (LDD).

El resultado de la compilación de las instrucciones de LDD es un conjunto de tablas que se almacenan en un archivo especial llamado diccionario de datos.

Un diccionario de datos es un archivo que contiene metadatos; es decir, datos acerca de datos. Este archivo se consulta antes de leer o modificar los datos reales del sistema de base de datos.

Lenguaje de manipulación de datos.

La manipulación de datos se refiere a las operaciones de insertar, recuperar, eliminar o modificar la información almacenada en la base de datos; dichas operaciones son realizadas a través del lenguaje de manipulación de datos (LMD), que es quién permite el acceso de los usuarios a los datos.

Existen básicamente dos tipos de lenguajes de manipulación de datos:

- **LMD procedimentales.** Requieren que el usuario especifique qué datos se necesitan y cómo obtenerlos.

- **LMD no procedimentales.** Requieren que el usuario especifique qué datos se necesitan y sin especificar cómo obtenerlos.

1.2.7 Administrador de la base de datos.

Una de las principales razones para usar SGBD es tener un control centralizado tanto de los datos como de los programas que acceden a esos datos. La persona que tiene este control central sobre el sistema se llama administrador de la base de datos (ABD), sus funciones principales son [1]:

- **Definición del esquema.** El ABD crea el esquema original de la base de datos escribiendo un conjunto de definiciones que son traducidas por el compilador de LDD a un conjunto de tablas que son almacenadas permanentemente en el diccionario de datos.

- **Estructura de almacenamiento y Definición del método de acceso**. Los ABD crean las estructuras de almacenamiento y de acceso adecuados escribiendo un conjunto de definiciones, que son traducidas por e compilador del lenguaje de almacenamiento y definición de datos.

- **Esquema y modificación de la organización física**. Los programadores llevan a cabo las relativamente escasas modificaciones sobre el esquema de base de datos o la descripción de la organización de almacenamiento físico escribiendo un conjunto de definiciones que son usadas bien por el compilador LDD o bien por el compilador del lenguaje de definición y almacenamiento de datos para generar la modificaciones en las tablas correspondientes del sistema interno.

- **Concesión de la autorización para el acceso a los datos.** Permite al administrador de la base de datos regular las partes de las bases de datos que van a ser accedidas por varios usuarios.

- **Especificación de las ligaduras de integridad.** Es una serie de restricciones que se encuentran almacenados en una estructura especial del sistema que es consultada por el gestor de base de datos cada vez que se realice una actualización al sistema.

1.2.8 Usuarios de bases de datos.

Podemos definir a los usuarios como toda persona que interactúa con el sistema de base de datos desde que este se diseña, elabora, termina y se usa [1].

Hay cuatro tipos diferentes de usuarios de un sistema de base de datos, diferenciados por la forma en que ellos esperan interactuar con el sistema.

- **Programadores de aplicaciones.** Son los profesionales informáticos que interactúan con el sistema por medio de llamadas del LDM, las cuales están incorporadas en un programa escrito en un lenguaje de programación (Por ejemplo, COBOL, PL/I, Pascal, C, etc.).

- **Usuarios sofisticados.** Interactúan con el sistema sin programas escritos. En cambio escriben sus preguntas en un lenguaje de consultas de base de datos.

- **Usuarios especializados.** Son usuarios sofisticados escriben aplicaciones de base de datos especializadas que no encajan en el marco tradicional de procesamiento de datos.

- **Usuarios normales.** Son usuarios no sofisticados que interactúan con el sistema invocando a uno de los programas de aplicación permanentes que se han escrito anteriormente en el sistema de base de datos, podemos mencionar al usuario normal como el usuario final que utiliza el sistema de base de datos sin saber nada del diseño interno del mismo.

1.2.9 Estructura del sistema completo.

Un sistema de base de datos se encuentra dividido en módulos cada uno de los cuales controla una parte de la responsabilidad total de sistema. En la mayoría de los casos, el sistema operativo proporciona únicamente los servicios más básicos y el sistema de la base de datos debe partir de esa base y controlar además el manejo correcto de los datos. Así el diseño de un sistema de base de datos debe incluir la interfaz entre el sistema de base de datos y el sistema operativo [1].

Los componentes funcionales de un sistema de base de datos se pueden dividir a grandes rasgos en componentes de procesamiento de consultas y componentes de gestión de almacenamiento.

Los componentes de procesamiento de consultan incluyen:

- **Compilador del LMD**, que traduce las instrucciones del LMD en lenguaje de consultas a instrucciones a bajo nivel que entiende el motor de evaluación de consultas.

- **Precompilador del LMD incorporado**, que convierte las instrucciones del LMD incorporadas en un programa de aplicación en llamadas a procedimientos normales en el lenguaje anfitrión.

- **Interprete del LDD,** que interpreta las instrucciones del LDD y las registra en un conjunto de tablas que contiene metadatos.

- **Motor de evaluación de consultas**, que ejecuta las instrucciones a bajo nivel generadas por el compilador del LMD.

El gestor de almacenamiento incluye:

- **Gestor de autorización e integridad,** que comprueba que se satisfagan las ligaduras de integridad y la autorización de los usuarios para acceder a los datos.

- **Gestor de transacciones,** que asegura que la base de datos quede en un estado consistente a pesar de los fallos del sistema, y que las ejecuciones de transacciones concurrentes ocurran sin conflictos.

- **Gestor de archivos,** que gestiona la asignación de espacio en la memoria del disco y de las estructuras de datos usadas para representar información almacenada en disco.

- **Gestor de memoria intermedia,** que es responsable de traer los datos del disco de almacenamiento a memoria principal y decidir qué datos traer en la memoria caché.

Además, se necesitan varias estructuras de datos como parte de la implementación física del sistema**:**

- **Archivos de datos,** que almacenan la base de datos en sí.

- **Diccionario de datos,** que almacena metadatos acerca de la estructura de la base de datos.

- **Índices,** que proporcionan acceso rápido a elementos de datos que tienen valores particulares.

- **Datos estadísticos,** que almacenan información estadística sobre los datos en la base de datos.

1.3 PHP.

PHP (acrónimo de "PHP: Hypertext Preprocessor") es un lenguaje de programación usado principalmente como herramienta de desarrollo de aplicaciones Web [10]. PHP nos permite diseñar paginas dinámicamente de servidor, es decir, generar páginas bajo petición capaces de responder de manera inteligente a las demandas del cliente y que nos permitan la automatización de gran cantidad de tareas.

PHP está muy relacionado con el lenguaje de hipertexto HTML; tanto es así que el código PHP aparece normalmente insertado dentro de un documento HTML. El documento PHP, una vez interpretado correctamente en el servidor, genera una página HTML que será enviada al cliente.

En PHP se combinan muchas características que contribuyen notablemente su utilización masiva; entre otras, está el hecho de ser un software de libre distribución y multiplataforma que sigue la filosofía de Open Source. Una de las características que más ha influido en su popularización es la sencillez de uso que presenta a los programadores principiantes.

PHP proporciona las siguientes funcionalidades:

- **Funciones de correo electrónico:** envío de correos individual o por grupos, parametrizando toda una serie de aspectos tales como el e-mail de procedencia, asunto, destinatario.

- **Gestión de bases de datos:** El lenguaje PHP ofrece interfaces para el acceso a la mayoría de las bases de datos comerciales y por ODBC a todas las bases de datos posibles en sistemas Microsoft.

- **Gestión de archivos:** mediante operaciones de creación, borrado, modificación, además de ofrecer transferencia de archivos.

- **Tratamiento de imágenes:** mediante funciones de automatización de formato, envío de lotes de imágenes y funciones de graficado.

PHP además proporciona las siguientes posibilidades:

- Soporte para múltiples sistemas operativos: Unix, Microsoft Windows, Mac OS X, RISC OS.

- Soporte para múltiples servidores Web: Apache, Microsoft Internet Information Server, Personal Web Server, Netscape e iPlanet, Oreilly Website Pro Server, Caudium, Xitami, OmniHTTPd y muchos otros.

- Soporte para múltiples base de datos: Adabas D, Ingres, Oracle, dBase, InterBase, Ovrimos, Empress, FrontBase, PostgresSQL, mSQL, Solid, Hyperwave, Direct MS-SQL, Sybase, IBM DB2, Informix, Unix dbm y MySQL, entre otras.

- Soporte para ODBC y extensiones DBX.

- Soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM y muchos más.

- Generación de resultados en multiples formatos como XHTML, XML, ficheros de imágenes, ficheros PDF Y películas Flash.

- Funciones de comercio electrónico, como Cybercash, CyberMUT, VeriSign Playflow Pro y CCVS para las pasarelas de pago.

Y un sinfín de posibilidades que van en aumento cada día.

1.4 MySQL.

MySQL es un sistema gestor de base de datos relacionales, que además ofrece compatibilidad con PHP, Perl, C y HTML, y funciones avanzadas de administración y optimización de base de datos para facilitar las tareas habituales [9].

Implementa funcionalidades Web, permitiendo un acceso seguro y sencillo a los datos a través de Internet. Este sistema gestor de base de datos incluye capacidades a análisis integradas, servicios de transformación y duplicación de datos y funciones de programación mejoradas.

Se puede decir que MySQL es un sistema cliente servidor de administración de base de datos relacionales diseñado para el trabajo tanto en los sistemas operativos Windows como en los sistemas Unix/Linux. Además, determinadas sentencias de MySQL pueden ser embebidas en código PHP y HTML para diseñar aplicaciones Web dinámicas que incorporan la información de las tablas de MySQL a páginas Web. Asimismo, MySQL es compatible con el software más potente de diseño Web, como Dreamweaver MX.

MySQL proporciona las siguientes características y ventajas:

- Disponibilidad en gran cantidad de plataformas y sistemas.

- Diferentes opciones de almacenamiento según si se desea velocidad en las operaciones o el mayor número de operaciones disponibles.

- Transacciones y claves foráneas.

- Conectividad segura.

- Replicación.

- Búsqueda e indexación de campos de texto.

- Acceso a las bases de datos de forma simultánea por varios usuarios y/o aplicaciones.

- Seguridad, en forma de permisos y privilegios.

- Portabilidad: SQL es también un lenguaje estandarizado, de modo que las consultas hechas usando SQL son fácilmente portables a otros sistemas y plataformas.

- Escalabilidad: es posible manipular bases de datos enormes, del orden de seis mil tablas y alrededor de cincuenta millones de registros, y hasta 32 índices por tabla.

CAPITULO 2. ANALISIS DEL SISTEMA.

2.1 Especificación de requisitos.

Se debe implementar un sistema de Bases de Datos en Web de consulta de medicamentos existentes en una farmacia con las siguientes especificaciones:

Este servicio se prestará en varias sucursales, cada una mantendrá los datos de cada departamento y producto existentes en esa farmacia. Con esto los usuarios podrán ver la disponibilidad de cada medicamento e información relevante sobre éste, mediante una búsqueda de medicamentos, independiente de la sucursal o departamento, o bien por una búsqueda en una sucursal especifica.

Cada sucursal debe mantener la información de las cuentas de usuarios de consulta y administradores de cada una de ellas. Un usuario de consulta puede ser un cliente de la farmacia o bien un empleado de la misma, quienes solo podrán hacer consulta al sistema; por tanto el administrador se encargara de la manipulación de usuarios, sucursales, departamentos y productos en determinada sucursal.

2.1.1 Requerimientos del sistema.

El sistema de Bases de Datos para su funcionamiento requiere de una computadora con las siguientes características mínimas:

Procesador Intel Celaron a 1.2 GHz., memoria de 256 MB, con sistema operativo Windows XP, un navegador Web como Internet Explorer o cualquier otro.

Para la realización del sistema, se requiere la utilización de las siguientes herramientas de software:

- Wampserver: Es un completo y eficiente paquete que instala Apache, PHP, MySQL y PhpMyAdmin en Windows, realizando la tarea de servidor.

- PHP: Lenguaje de programación que ayuda a crear páginas Web que servirán como interfaz del sistema y también para la introducción de datos por parte del usuario.

- MySQL: Lenguaje que nos permite realizar operaciones con la base de datos, como inserción, eliminación actualización y modificación de los registros en la base de datos.

- Microsoft Visio: Aplicación que nos permite modelar fácilmente diagramas UML.

- Dreamweaver: Software que permite crear páginas Web profesionales, generando código PHP y HTML.

2.2 Definición conceptual del sistema.

El sistema que se desarrolló es una de Bases de Datos en Web donde se muestra la existencia y características de los medicamentos en una Farmacia con Sucursales, además de llevar el control y administración de sucursales, y usuarios dentro del sistema. Teniendo como funciones importantes:

- Control de consultas realizadas.
- Control de sucursales, departamento y productos.
- Información de los usuarios.

La recopilación de la información para el sistema se hizo a través de entrevista con personal de diversas farmacias, así como también de personas que acostumbran adquirir frecuentemente medicamentos en determinada farmacia; en las cuales se les planteo la necesidad de contar con el sistema de consulta, y el beneficio en común que traerá consigo la implementación del mismo.

Las farmacias realizan su proceso de venta y consulta de manera local, el sistema que operan no contiene la información necesaria de un medicamento, se enfocan principalmente a la venta y al manejo de inventario, por consiguiente, no tienen la información necesaria de los medicamentos, en algunos de los casos tienen la necesidad de recurrir a la consulta de un medicamento dentro un diccionario ajeno al sistema.

Teniendo esta información, se obtuvieron los requisitos del sistema, los cuales se abordan en las siguientes secciones.

2.3 Descripción de la información.

Las funciones principales del sistema de Base de Datos son:

- Activar cuentas de usuarios de consulta y administrador.
- Eliminar cuentas de usuarios de consulta.
- Alta de cada sucursal, departamento y productos dentro de la misma.
- Modificar información de cada sucursal, departamento y productos.
- Eliminar productos.
- Mostrar mediante listas desplegables cada una de las sucursales, departamento y productos.
- Búsqueda de medicamentos.

- Generar reporte de los productos consultados.

2.3.1. Casos de uso: Descripción de procesos.

Los casos de uso están diseñados para cumplir los deseos del usuario cuando utiliza el sistema.

De manera precisa, un caso de uso es una descripción de un conjunto de secuencias de acciones, incluyendo variaciones, que un sistema lleva acabo y que conduce a un resultado observable de interés para un actor determinado [3].

A continuación se describen cada uno de los actores y sus casos de uso correspondientes, dentro del sistema.

Identificación de Actores.

Como una primera aproximación se identifica los actores que interactúan con el Sistema:

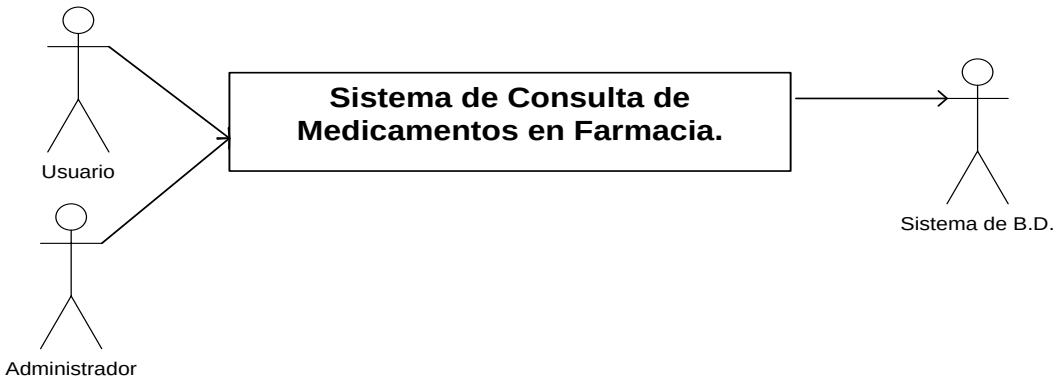


Figura 2.1 Actores que interactúan con el sistema.

Los actores involucrados en el sistema en el contexto de la función y el comportamiento son los siguientes: usuario (cliente o empleado), administrador y el Sistema de Base de Datos. Cuyas actividades de cada uno de los actores dentro del sistema se describen a continuación.

Usuarios:

- Consulta de medicamentos.

Administrador:

- Gestión de usuarios.
- Gestión de sucursal.
- Gestión de reporte de medicamentos consultados.

Sistema de Base de Datos.

- Genera reporte de medicamentos consultados.

Identificación de casos de uso.

Basándose en la secuencia de tareas de los escenarios de uso se crearon los Casos de Uso de manera que se pueda tener una idea clara de que es lo que se quiere funcionalmente del sistema y de la forma en la que se realizan los procesos. A continuación de presentan los casos de uso del sistema:

Caso de uso: Autentificar Usuario.
Actores: Usuario (cliente o empleado) o administrador.
Descripción: El Usuario o administrador proporcionan al sistema datos de acceso para ser validados y así poder acceder a él.

Caso de uso: Consultar Medicamento.
Actores: Usuario (cliente o empleado).
Descripción: El Usuario realiza una consulta de medicamento a través de una búsqueda de manera directa o bien a través de una sucursal.

Caso de uso: Gestionar Sucursal.
Actores: Administrador.
Descripción: El administrador comienza con el registro de una nueva sucursal con su respectivo departamento y productos que ahí se encuentren, así como también los usuarios de la misma, complementa su gestión con baja, modificación y consulta.

Caso de uso: Generar Informe.
Actores: Sistema de Base de Datos.
Descripción: El Sistema de Base de Datos almacena los datos de las consultas realizadas, y despliega esta información cuando es requerida.

Diagrama de caso de uso del sistema.

En el diagrama de la figura 2.2 se muestra como una segunda aproximación el sistema de forma general, en este diagrama se presentan los actores que van a interactuar directamente con el sistema y sus casos de usos sobre los cuales interactúa:

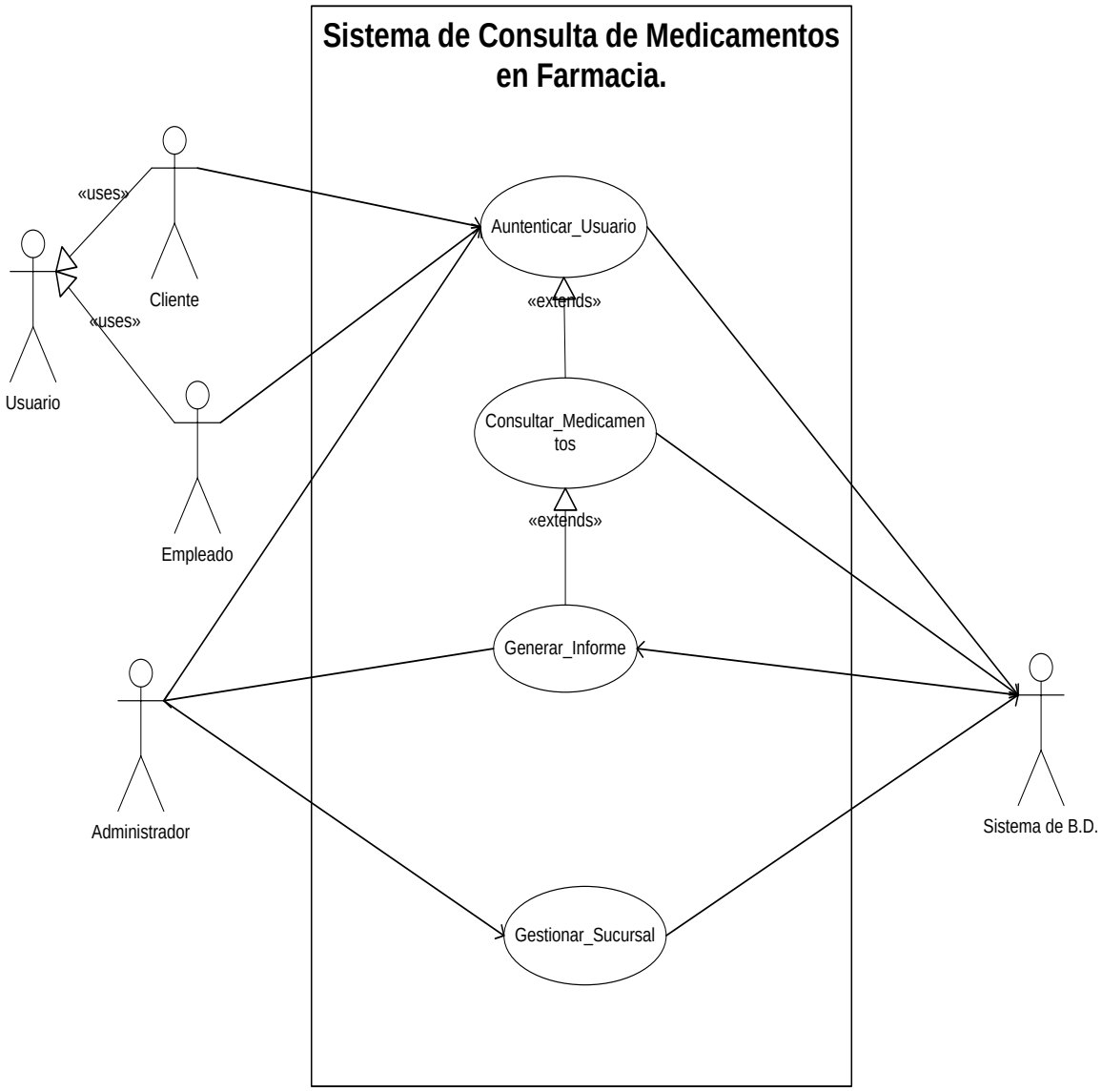


Figura 2.2 Diagrama general de casos de uso del sistema.

En la figura 2.2 aparecen las relaciones “uses” y “extends”. La relación “uses” indican la relación que se da entre un caso abstracto y uno concreto y la relación “extends” permite que un caso de uso se inserte en otro [7].

2.3.2. Flujo de eventos.

El propósito del flujo de eventos es documentar el flujo lógico que seguirán los casos de uso. En el flujo de eventos se describe a detalle lo que realizan los usuarios del sistema y como responde éste a las solicitudes que le son requeridas.

El flujo de eventos incluye:

- **Descripción.** Se refiere a un resumen concreto de lo que realiza el caso de uso.
- **Actores.** Son aquellos que interactúan con el sistema.
- **Precondiciones.** Son los hechos que se han de cumplir para que el flujo de evento se pueda llevar a cabo.
- **Flujo normal.** Describe paso a paso la funcionalidad que ejecuta cada caso de uso, situándose en las condiciones del escenario ideal.
- **Flujo alternativo.** Son los que nos permiten indicar qué es lo que hace el sistema en los casos menos frecuentes e inesperados.
- **Poscondiciones.** Son los hechos que se ha de cumplir si el flujo de eventos normal se ha ejecutado correctamente.

A continuación se presenta el flujo de eventos para cada caso de uso del sistema, representados en la figura 2.2. Se representan primeramente de manera gráfica los actores y casos de uso que intervienen y a continuación el flujo correspondiente.

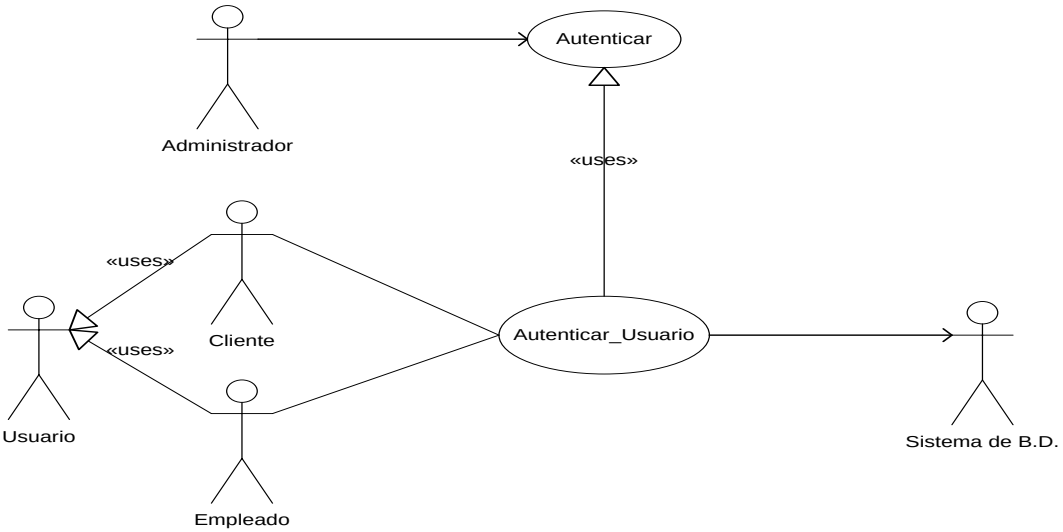


Figura 2.3 Caso de Uso Autentificar Usuario.

Nombre	*Autentificar Usuario.*

Descripción:

Permite validar el acceso a través del Sistema de Base de Datos al usuario (cliente o empleado), en el caso del administrador solo se válida con el sistema; por ello cada uno de los actores goza con distintos privilegios.

Actores:

Usuario previamente registrado en el sistema, administrador del sistema, Sistema de B.D.

Precondiciones:

El usuario tiene que estar registrado en el sistema, para el caso del administrador contar con claves de acceso.

Flujo normal:

1. El sistema solicita el nombre de usuario y clave del actor en cuestión.
2. El actor proporciona su nombre de usuario y clave.
3. El sistema valida con el Sistema de B.D. o con el propio sistema, los datos del usuario o administrador respectivamente.
4. El sistema despliega una nueva página de bienvenida al actor en cuestión.

Flujo Alternativo:

3. El sistema comprueba la validez de los datos, si los datos no son correctos, se avisa al actor de ello permitiéndole proporcionarlos nuevamente.

Poscondiciones:

El actor ha ingresado correctamente al sistema.

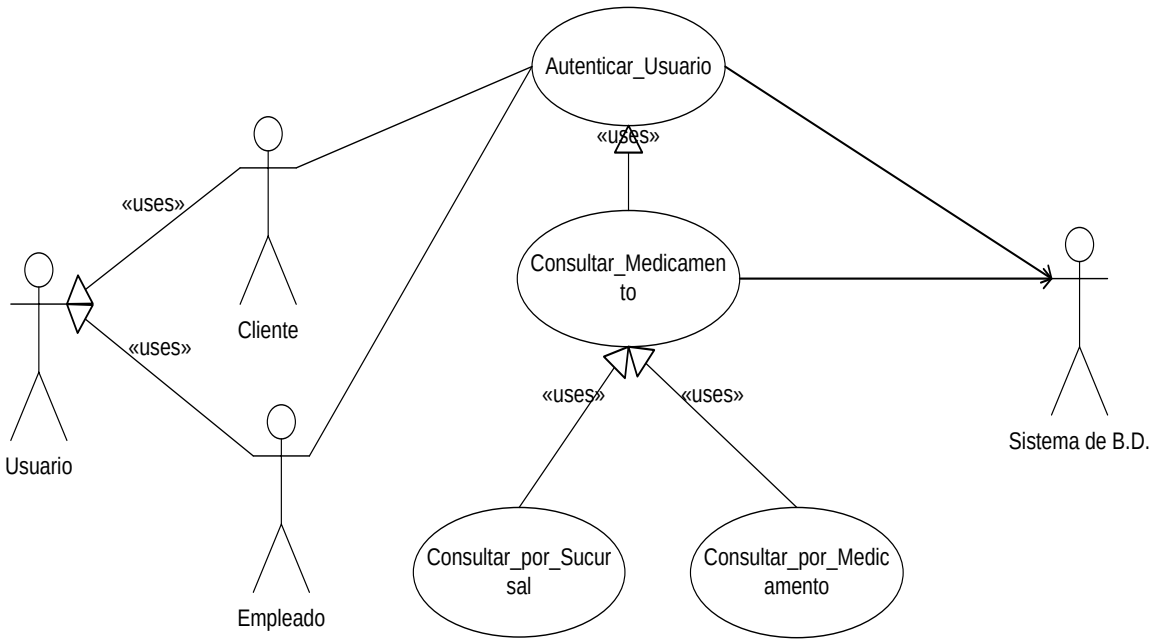


Figura 2.4 Caso de Uso Consultar Medicamento.

Nombre	***Consultar Medicamento.***

Descripción:

Permite al usuario la consulta de un medicamento en farmacia, mediante una búsqueda por medicamento directamente o bien por una sucursal especifica.

Actores:

Usuario (cliente o empleado) del sistema autenticado, Sistema de B.D.

Precondiciones:

El Usuario (cliente o empleado) debe haberse autenticado en el sistema.

Flujo normal:

1. El actor pulsa sobre el botón continuar en la página de bienvenida.
2. El sistema despliega una nueva página con las opciones de buscar un medicamento directamente o bien buscar un medicamento por una sucursal.
3. El actor elige la opción buscar medicamento directamente (por ejemplo).
4. El sistema despliega los medicamentos existentes en farmacia por orden alfabético.
5. El actor elige un medicamento y posteriormente pulsa el botón consultar para obtener la información de tal medicamento.
6. El sistema despliega la información correspondiente al medicamento seleccionado.
7. El sistema almacena los datos de la consulta.
8. El actor regresa al paso 4 del flujo.

Flujo Alternativo:

7. El actor no requiere realizar otra consulta y selecciona la opción salir.

Poscondiciones:

Los datos de la consulta han sido almacenados en el sistema.

Figura 2.5 Caso de Uso Gestionar Sucursal.

Debido a que este caso de uso es extenso, solo se ejemplifica mediante una opción que puede elegir el actor, como lo es la elección de la entidad "sucursal" mediante la operación de dar de "alta", como se puede notar en la figura 2.5 son similares los casos de uso, así que basta ejemplificar solo uno, para ver que son similares las operaciones para los demás.

Nombre	*Gestionar Sucursal.*

Descripción:

Permite al administrador la gestión de sucursales, comenzando con el registro de una nueva sucursal con su respectivo departamento y productos que ahí se encuentren, así como también los usuarios de la misma. Complementa su gestión con baja, modificación y consulta de las entidades dentro de la sucursal.

Actores:

Administrador del sistema autenticado, Sistema de B.D.

Precondiciones:

El Administrador debe haberse autenticado en el sistema.

Flujo normal:

1. El actor pulsa sobre el botón continuar en la página de bienvenida.
2. El sistema despliega una nueva página para la gestión de sucursales, con las opciones de sucursal, departamento, producto y usuario.
3. El actor elige la opción sucursal (por ejemplo).
4. El sistema despliega las opciones de gestión de sucursal, es decir alta, baja, consulta y modificación.
5. El actor elige la opción alta (por ejemplo).
6. El sistema solicita al actor introduzca los datos correspondientes para llevar acabo la alta de una sucursal.
7. El actor introduce los datos requeridos.
8. El sistema valida los datos introducidos y ejecuta un mensaje de que los datos fueron insertados correctamente.
9. El actor confirma el mensaje y regresa al paso 6 del flujo.

Flujo Alternativo:

8. El sistema valida los datos introducidos, si los datos no son correctos o son nulos, el sistema envía un mensaje de error, y solicita nuevamente los datos.

Poscondiciones:

Los cambios hechos por el actor son guardados por el sistema.

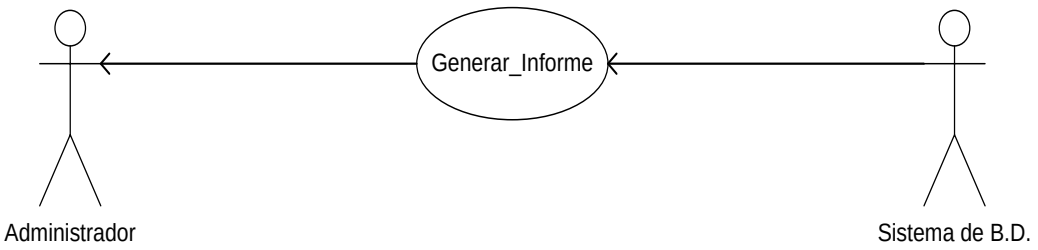


Figura 2.6 Caso de Uso Generar Informe.

Nombre	***Generar Informe.***
Descripción: Permite al Sistema de Base de Datos almacenar los datos de las consultas realizadas, y despliega esta información cuando es requerida por el administrador del sistema.	
Actores: Sistema de B.D., administrador.	
Precondiciones: El administrador debe haberse autenticado en el sistema.	
Flujo normal: 1. El actor selecciona la opción generar informe para ver las consultas. 2. El sistema solicita al actor elija la sucursal para obtener detalles de las consultas hechas en ella. 3. El actor elije una sucursal. 4. El sistema despliega la información de las consultas echas en la sucursal. 5. El actor confirma la información y regresa al paso 2 del flujo.	
Flujo Alternativo: 4. El actor no requiere realizar otro informe y selecciona la opción salir.	
Poscondiciones: Los cambios hechos por el actor son guardados por el sistema.	

CAPITULO 3. DISEÑO DEL SISTEMA.

Esta etapa consta de tres fases: diseño conceptual, diseño lógico y diseño físico de la base de datos [8].

Los objetivos del diseño de la base de datos son:

- Representar los datos que requieren las principales áreas de aplicación y los grupos de usuarios, y representar las relaciones entre dichos datos.
- Proporcionar un modelo de datos que soporte las transacciones que se vayan a realizar sobre los datos.
- Especificar un esquema que alcance las prestaciones requeridas para el sistema.

3.1 Diseño Conceptual.

En esta etapa se debe construir un esquema de la información que se usa en la empresa, independientemente de cualquier consideración física. A este esquema se le denomina esquema conceptual. Al construir este esquema, descubrimos la semántica de los datos de la empresa: se identifican entidades, atributos y relaciones.

Las tareas a realizar en el diseño conceptual son las siguientes:

- Identificar las entidades.
- Identificar las relaciones.
- Identificar los atributos y asociarlos a entidades y relaciones.
- Determinar los dominios de los atributos.
- Determinar los identificadores.
- Determinar las jerarquías de generalización (si las hay).
- Dibujar el diagrama entidad-relación.
- Revisar el esquema conceptual local con el usuario.

3.1.1. Identificación de entidades.

Las entidades identificadas después de haber realizado el análisis de requerimientos del sistema, son las siguientes:

Entidad.	**Descripción.**
Sucursal.	Contiene la información sobre la sucursal de la farmacia.
Departamento.	Contiene la información del departamento dentro de una sucursal.
Producto.	Contiene la información del producto dentro de un departamento.
Consulta.	Contiene los detalles de las consultas realizadas.
Usuario.	Contiene la información de un usuario dentro del sistema.

3.1.2. Identificación de Relaciones.

Una vez definidas las entidades, se procede a identificar las relaciones existentes entre ellas junto con la cardinalidad con la que participa cada entidad en cada una de las relaciones, las cuales se muestran a continuación:

Relaciones.	**Relación.**	**Carnidalidad.**
Sucursal – Departamento.	Tiene.	Uno a Muchos.
Departamento – Producto.	Almacena.	Uno a Muchos.
Sucursal – Consulta.	Genera.	Uno a Muchos.
Usuario – Consulta.	Realiza.	Uno a Muchos.
Producto – Consulta.	Abarca.	Uno a Muchos.

3.1.3. Identificación de atributos y dominios.

De igual forma que con las entidades, se buscan nombres en el análisis de requerimientos. Son atributos los nombres que identifican propiedades, cualidades, identificadores o características de entidades o relaciones. El dominio de un atributo es el conjunto de valores que puede tomar el atributo. A continuación se enlistan los atributos y dominios para cada una de las entidades:

Entidad: Sucursal.				
Atributo	**Descripción**	**Tipo/Longitud**	**Dominio**	**Nulo**
ID_SUCURSAL.	Identificador único de sucursal.	INT (5)	0…9	no
NOMBRE_SUC.	Especifica el nombre de determinada sucursal.	VARCHAR(30)	A…Z	no
DIRECCIÓN.	Muestra la ubicación de la sucursal.	VARCHAR(50)	A…Z y/o 0…9	no
HORARIO.	Especifica el horario de atención de sucursal	VARCHAR(20)	A…Z y/o 0…9	no

Entidad: Consulta.				
Atributo	**Descripción**	**Tipo/ Longitud**	**Dominio**	**Nulo**
ID_CONSULTA.	Identificador único de consulta.	INT (5)	0…9	no
FECHA.	Contiene la fecha en que fue realizada la consulta.	DATETIME ()	0…9	no

Entidad: Usuario.				
Atributo	**Descripción**	**Tipo/Longitud**	**Dominio**	**Nulo**
ID_USUARIO.	Identificador único de usuario.	INT (5)	0…9	no
PASSWORD.	Contiene la clave con la que se autentica el usuario.	VARCHAR (10)	A…Z y/o 0…9	no
NOMBRE.	Contiene el nombre del usuario.	VARCHAR (30)	A…Z	no
APELLIDO_PATERNO.	Contiene el apellido paterno del usuario.	VARCHAR (20)	A…Z	no
APELLIDO_MATERNO.	Contiene el apellido materno del usuario.	VARCHAR (20)	A…Z	no
DIRECCIÓN.	Contiene el domicilio particular del usuario.	VARCHAR (50)	A…Z y/o 0…9	no
TELÉFONO.	Contiene el número telefónico del usuario.	VARCHAR (10)	0…9	no
E-MAIL.	Contiene la dirección de correo electrónico del usuario.	VARCHAR (30)	A…Z y/o 0…9	no

Entidad: Departamento.				
Atributo	**Descripción**	**Tipo/Longitud**	**Dominio**	**Nulo**
NOMBRE_DEPTO.	Especifica el nombre del departamento dentro de una sucursal.	VARCHAR (30)	A…Z	no
TIPO_PRODUCTO.	Contiene el tipo de	VARCHAR	A…Z	no

	productos que se maneja en dicho departamento.	(50)		
CATEGORÍA_PRODUCTO.	Especifica la categoría de productos que se maneja en el departamento.	VARCHAR (50)	A…Z	no
SUBCATEGORÍA.	Especifica la subcategoría de productos que se maneja en el departamento.	VARCHAR (50)	A…Z	no

Entidad: Producto.				
Atributo	**Descripción**	**Tipo/Longitud**	**Dominio**	**Nulo**
ID_PRODUCTO.	Especifica la clave única con la cual se distingue el producto.	INT (5)	0…9	no
NOMBRE.	Identifica el nombre del producto.	VARCHAR (50)	A…Z	no
PRECIO.	Contiene el precio del producto.	FLOAT ()	0…9	no
PRESENTACIÓN.	Muestra la presentación en la que se encuentra disponible el producto.	VARCHAR (50)	A…Z	no
GRAMAJE.	Muestra el gramaje del producto.	VARCHAR (50)	0…9	no

SUSTANCIA_ACTIVA.	Contiene información sobre la sustancia activa que contiene el producto.	VARCHAR (50)	A…Z	no
CONTRAINDICACIONES.	Contiene información sobre la contraindicación del producto.	VARCHAR (100)	A…Z	no
IMAGEN.	Muestra la imagen del producto existente.	VARCHAR (30)	A…Z	no
DISPONIBILIDAD.	Muestra la existencia de tal producto en farmacia	INT (3)	0…9	no
PROVEEDOR.	Muestra la información del proveedor.	VARCHAR (30)	A…Z	no

3.1.4. Determinación de identificadores.

Cada entidad posee al menos un identificador. Se trata de encontrar todos los identificadores de cada una de las entidades, los cuales pueden ser simples o compuestos. De cada entidad hemos escogido uno de los identificadores como clave primaria en la fase del diseño lógico.

Entidad: Sucursal.	
Atributo.	**Identificador.**
ID_SUCURSAL.	PK.

Entidad: Departamento.	
Atributo.	**Identificador.**
NOMBRE_DEPTO.	PK.

Entidad: Producto.	
Atributo.	**Identificador.**
ID_PRODUCTO.	PK.

Entidad: Consulta.	
Atributo.	**Identificador.**
ID_CONSULTA.	PK.

Entidad: Usuario.	
Atributo.	**Identificador.**
ID_USUARIO.	PK.

3.1.5. Diagrama Entidad-Relación.

Una vez identificados todos los conceptos anteriores, se procede a crear el diagrama entidad-relación correspondiente a una de las vistas de los usuarios. El cual fue diseñado en Microsoft Visio, donde las casillas rectangulares representan las entidades, y las líneas una relación entre éstas, con cierta cardinalidad. En la figura 3.1 se muestran las 5 entidades con su respectiva relación y su cardinalidad.

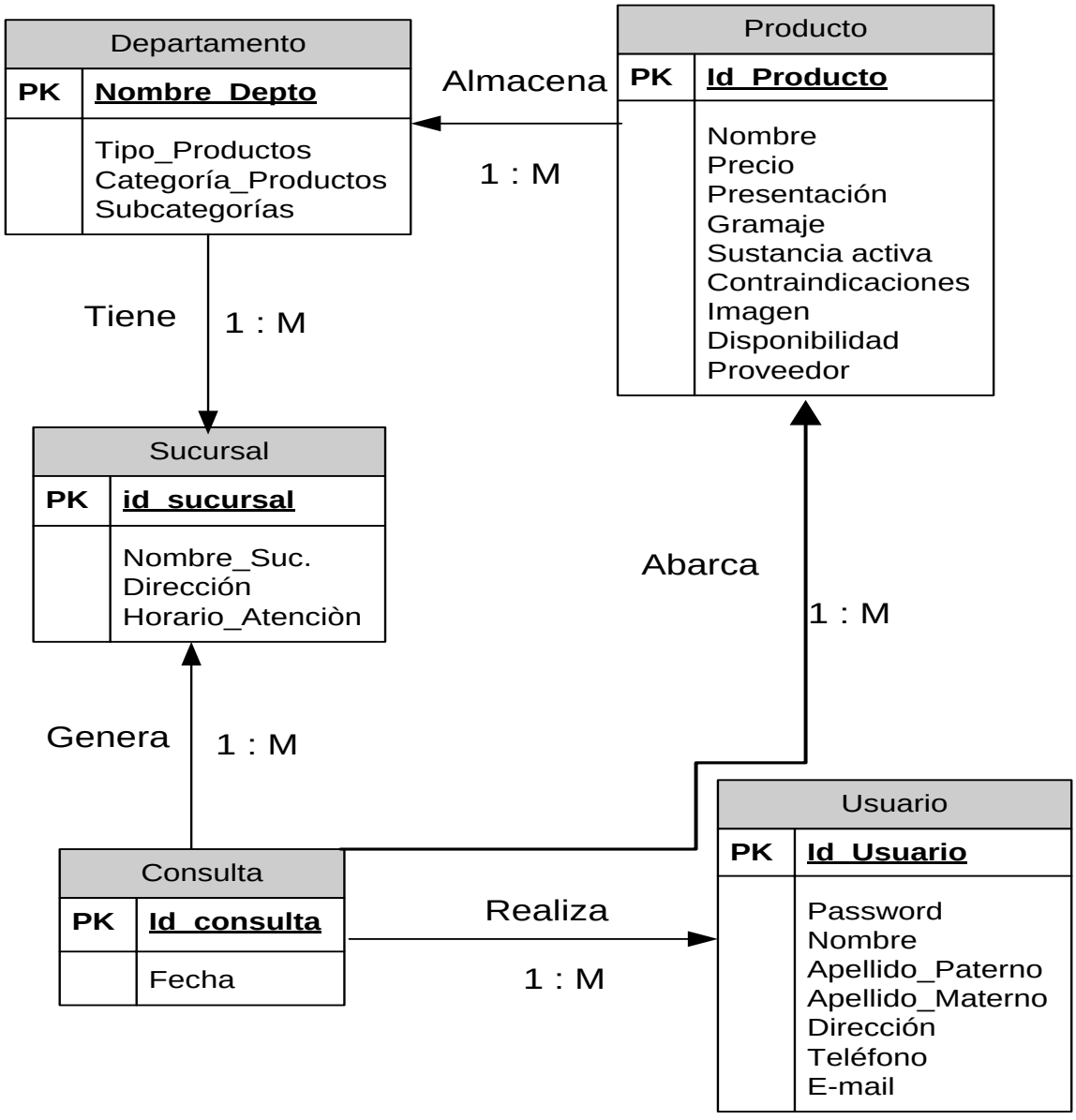


Figura 3.1 Diagrama Entidad-Relación.

3.2.Diseño Lógico.

El diseño lógico es el proceso de construir un esquema de la información que utiliza la empresa, basándose en un modelo de base de datos específico, independiente del SGBD concreto que se vaya a utilizar y de cualquier otra consideración física.

En esta etapa, se transforma el esquema conceptual en un esquema lógico que utilizará las estructuras de datos del modelo de base de datos en el que se basa el SGBD que se vaya a utilizar, en este caso se utiliza el modelo relacional. Conforme se va desarrollando el esquema lógico, éste se va probando y validando con los requisitos de usuario.

3.2.1. Modelo relacional.

Para llevar a cabo la transformación del modelo entidad-relación al modelo relacional se siguen las siguientes reglas:

1. Toda entidad se convierte en una tabla.

2. Toda relación N: M se genera una nueva tabla que contenga cada una de las llaves primarias de las entidades involucradas en la relación.

3. Toda relación 1: M se incluye en la tabla M la llave primaria del la tabla 1 como llave foránea.

De acuerdo a las reglas anteriores el modelo relacional queda de la siguiente manera:

SUCURSAL (ID_SUCURSAL PK, NOMBRE_SUC, DIRECCIÓN, HORARIO).

DEPARTAMENTO (NOMBRE_DEPTO PK, ID_SUCURSAL FK, TIPO_PRODUCTO, CATEGORÍA_PRODUCTO, SUBCATEGORÍA).

PRODUCTO (ID_PRODUCTO PK, NOMBRE_DEPTO FK, NOMBRE, PRECIO, PRESENTACIÓN, GRAMAJE, SUSTANCIA_ACTIVA, CONTRAINDICACIONES, IMAGEN, DISPONIBILIDAD, PROVEEDOR).

CONSULTA (ID_CONSULTA PK, ID_SUCURSAL FK, ID_PRODUCTO FK, ID_USUARIO FK, FECHA).

USUARIO (ID_USUARIO PK, PASSWORD, NOMBRE, APELLIDO_PATERNO, APELLIDO_MATERNO, DIRECCIÓN, TELÉFONO, E-MAIL).

3.2.2. Normalización.

La normalización es una técnica que se utiliza para comprobar la validez de los esquemas lógicos basados en el modelo relacional, ya que asegura que las relaciones (tablas) obtenidas no tienen datos redundantes.

Las ventajas de la normalización son las siguientes:

- Evita anomalías en inserciones, modificaciones y borrados.
- Mejora la independencia de datos.
- No establece restricciones artificiales en la estructura de los datos.

El modelo relacional sólo requiere un conjunto de relaciones en primera forma normal. Las restantes formas normales son opcionales. Sin embargo, para evitar las anomalías de actualización, es recomendable llegar al menos a la tercera forma normal como se muestra a continuación:

Primera forma normal. (1FN):

Una relación está en primera forma normal si, y sólo si, todo los dominios de la misma contienen valores atómicos, es decir, no hay grupos repetidos.

Como cada una de las relaciones maneja valores atómicos, ya se encuentra en primera forma normal, lo cual se ejemplifica con la tabla Sucursal.

id_sucursal	nombre_suc	dirección	horario
10001	San Manuel	Av. 14 sur 5701	9:00 a 21:00 hrs.
10002	CAPU	Blvd. Norte 2502	8:00 a 24:00 hrs.

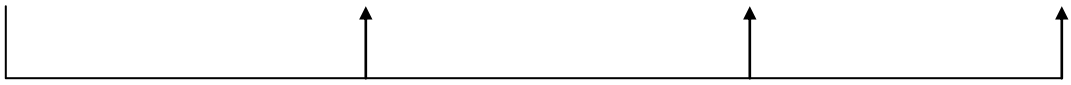
Segunda forma normal. (2FN):

Una relación se encuentra en segunda forma normal, cuando cumple con las reglas de la primera forma normal y todos sus atributos que no son claves (llaves) dependen por completo de la clave.

En todas y cada una de las tablas se cumple esta regla, ya que todos los atributos son totalmente dependientes funcionalmente de la llave primaria como se muestra a continuación:

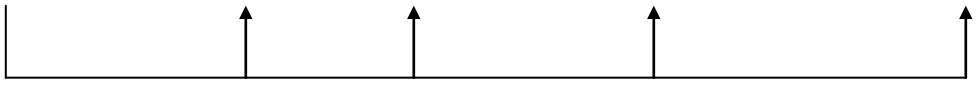
Sucursal:

id_sucursal	nombre_suc	dirección	horario

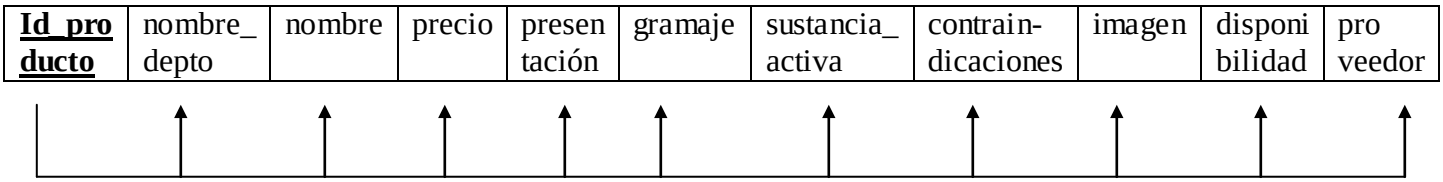


Departamento:

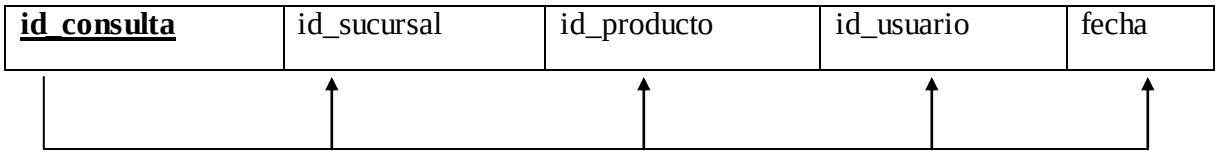
nombre_depto	id_sucursal	tipo_producto	categoría_producto	subcategoría



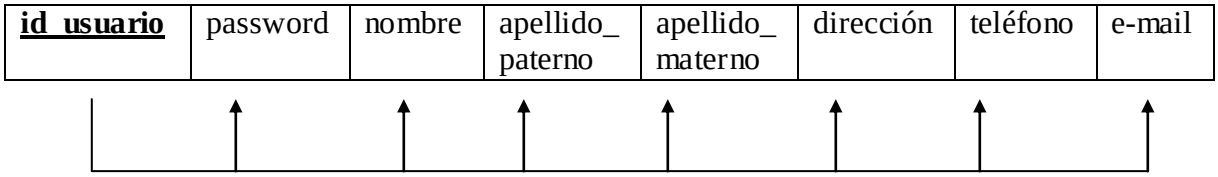
Producto:



Consulta:



Usuario:



Tercera forma normal. (3FN):

Una relación esta en tercera forma normal si, y sólo si, está en segunda forma normal y, además no existen dependencias transitivas entre los atributos, se refiere a dependencias transitivas cuando existe más de una forma de llegar a referenciar a un atributo de una relación.

Como se pudo apreciar anteriormente todos y cada uno de los atributos son dependientes solo de la llave primaria, por tanto ya se encuentra en tercera forma normal.

3.3.Diseño físico.

El diseño físico es el proceso de producir la descripción de la implementación de la base de datos en memoria secundaria: estructuras de almacenamiento y métodos de acceso que garanticen un acceso eficiente a los datos.

Para llevar a cabo esta etapa, se decidió utilizar MySQL como el SGBD, ya que el esquema físico se adapta a él.

3.3.1. Estructuras de las tablas en MySQL:

Base de datos: **`farmacia`**:

Estructura de tabla para la tabla **`consulta`**:

```
CREATE TABLE `consulta` (
  `id_consulta` int(5) NOT NULL auto_increment,
  `id_sucursal` int(5) NOT NULL,
  `id_producto` int(5) NOT NULL,
  `id_usuario` int(5) NOT NULL,
  `fecha` datetime NOT NULL,
  PRIMARY KEY  (`id_consulta`),
  KEY `id_sucursal` (`id_sucursal`,`id_producto`,`id_usuario`),
  KEY `id_producto` (`id_producto`),
  KEY `id_usuario` (`id_usuario`)
)
```

Estructura de tabla para la tabla **`departamento`**:

```
CREATE TABLE `departamento` (
  `nombre_depto` varchar(30) NOT NULL,
  `id_sucursal` int(5) NOT NULL,
  `tipo_producto` varchar(50) NOT NULL,
  `categoria_producto` varchar(50) NOT NULL,
  `subcategoria` varchar(50) NOT NULL,
  PRIMARY KEY  (`nombre_depto`),
  KEY `id_sucursal` (`id_sucursal`)
)
```

Estructura de tabla para la tabla **`producto`**:

```
CREATE TABLE `producto` (
  `id_producto` int(5) NOT NULL auto_increment,
```

```
  `nombre_depto` varchar(30) NOT NULL,
  `nombre` varchar(50) NOT NULL,
  `precio` float NOT NULL,
  `presentacion` varchar(50) NOT NULL,
  `gramaje` varchar(50) NOT NULL,
  `sustancia_activa` varchar(50) NOT NULL,
  `contraindicaciones` varchar(100) NOT NULL,
  `imagen` varchar(30) NOT NULL,
  `disponibilidad` int(3) NOT NULL,
  `proveedor` varchar(30) NOT NULL,
  PRIMARY KEY  (`id_producto`),
  KEY `nombre_depto` (`nombre_depto`)
)
```

Estructura de tabla para la tabla **`sucursal`**:

```
CREATE TABLE `sucursal` (
  `id_sucursal` int(5) NOT NULL auto_increment,
  `nombre_suc` varchar(30) NOT NULL,
  `direccion` varchar(50) NOT NULL,
  `horario` varchar(20) NOT NULL,
  PRIMARY KEY  (`id_sucursal`)
)
```

Estructura de tabla para la tabla **`usuario`**:

```
CREATE TABLE `usuario` (
  `id_usuario` int(5) NOT NULL auto_increment,
  `password` varchar(10) NOT NULL,
  `nombre` varchar(30) NOT NULL,
  `apellido_paterno` varchar(20) NOT NULL,
  `apellido_materno` varchar(20) NOT NULL,
  `direccion` varchar(50) NOT NULL,
  `telefono` varchar(10) NOT NULL,
  `e_mail` varchar(30) NOT NULL,
  PRIMARY KEY  (`id_usuario`)
)
```

CAPITULO 4. IMPLEMENTACIÓN Y PRUEBAS DEL SISTEMA.

Durante esta etapa se crean las definiciones de la base de datos a nivel conceptual, externo e interno, así como los programas de aplicación. [8]

La prueba se emplea de manera experimental para asegurarse que el software no tenga fallas, es decir, que funciona de acuerdo con las especificaciones y en la forma en que los usuarios esperan que lo haga.

A continuación se describe la implementación y las pruebas del "Sistema de Consulta de Medicamentos en Farmacia".

Presentación de las interfaces del sistema y sus funciones correspondientes.

Como se mencionó en el análisis, el sistema cumplirá con los requerimientos iniciales como son: el registro, eliminación, actualización y consulta de sucursales, departamentos, medicamentos y usuarios (cliente o empleado). También incluye la generación de reporte de consulta de los medicamentos.

Como primera interfaz del sistema, se muestra una página de bienvenida para acceder al mismo, en la cual el usuario tiene que autentificarse ingresando su **"Nombre de Usuario"** y **"Password"** de manera correcta como se muestra en la Figura 4.1.

Figura 4.1 Inicio del Sistema.

Si al ingresar el **"Nombre de Usuario"** y **"Password"** estos son incorrectos se negará el acceso y se le mostrará un mensaje de error (Figura 4.2).

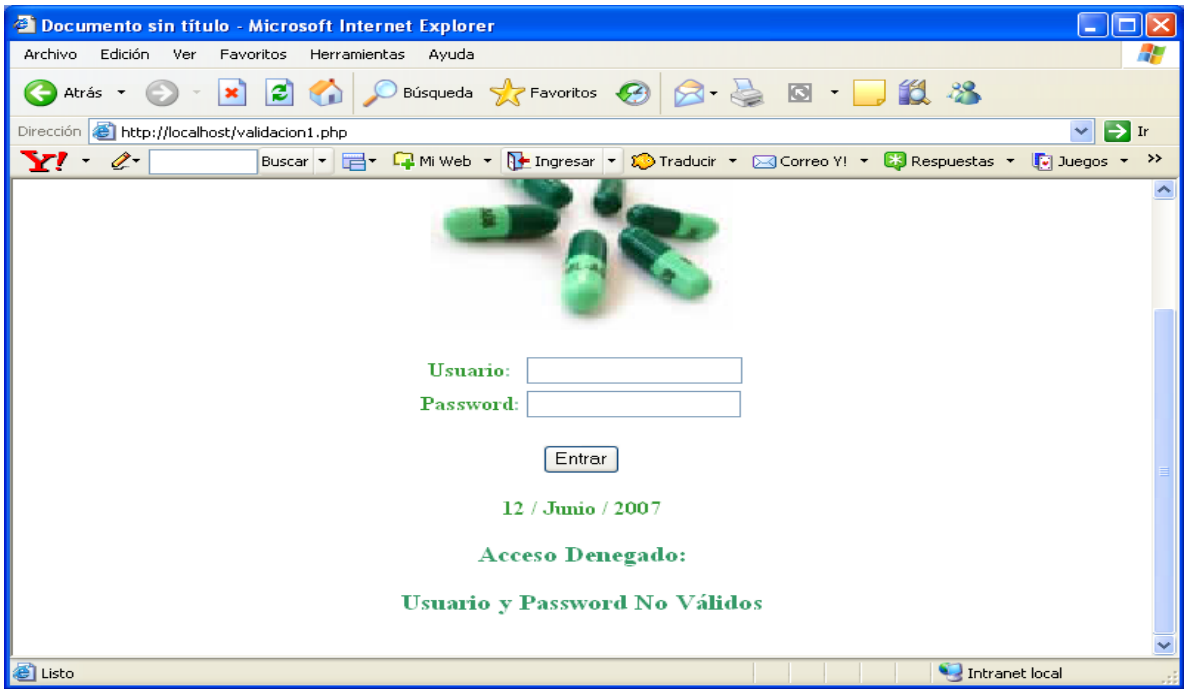


Figura 4.2 Datos de Usuario Incorrectos.

Al ingresar correctamente el **"Nombre de Usuario"** y **"Password"** lo primero que hace el sistema es validar que tipo de usuario es: administrador o usuario (cliente o empleado), una vez que se ha validado que el usuario está registrado en la base de datos se muestra la página acompañada de un menú correspondiente.

4.1 Menú administrador.

Después de haberse identificado el usuario como administrador se muestra una página de bienvenida a éste (Figura 4.3) y que continúa con la presentación de una nueva página que muestra un menú, en éste se muestran la gestión de sucursales, departamentos, medicamentos, usuarios y reportes, lo único que se tiene que hacer es dar un clic en la opción deseada y si requiere salir del sistema debe dar clic sobre el botón **"Salir"**, como se muestra en la Figura 4.4.

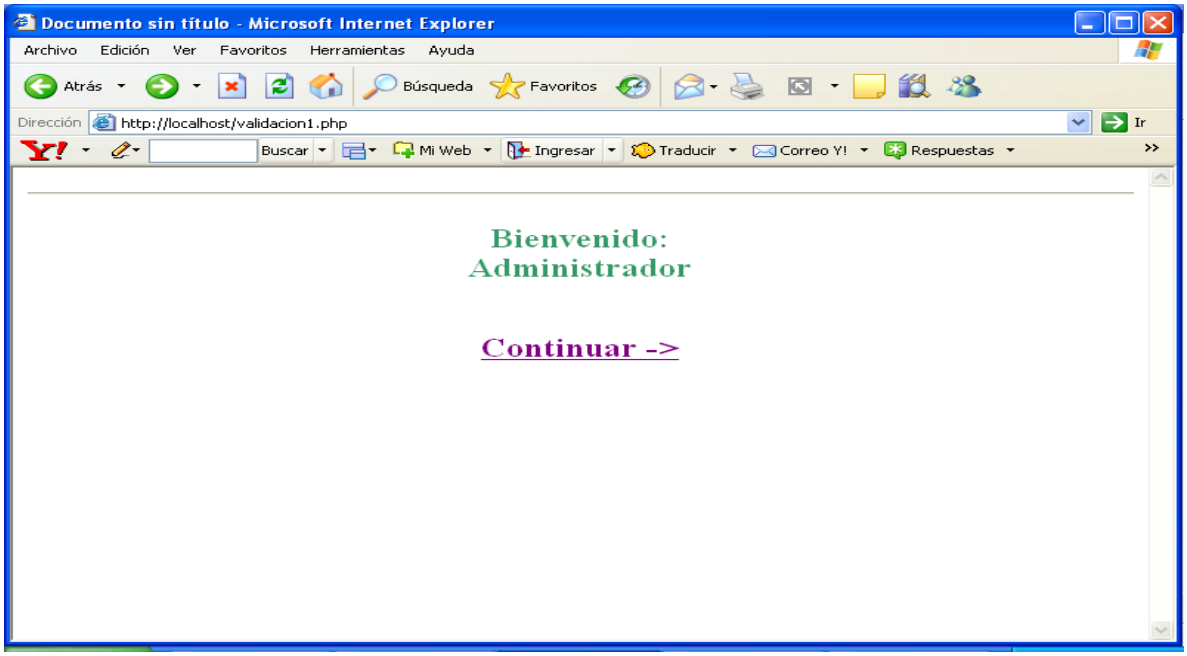


Figura 4.3 Bienvenida al Administrador.

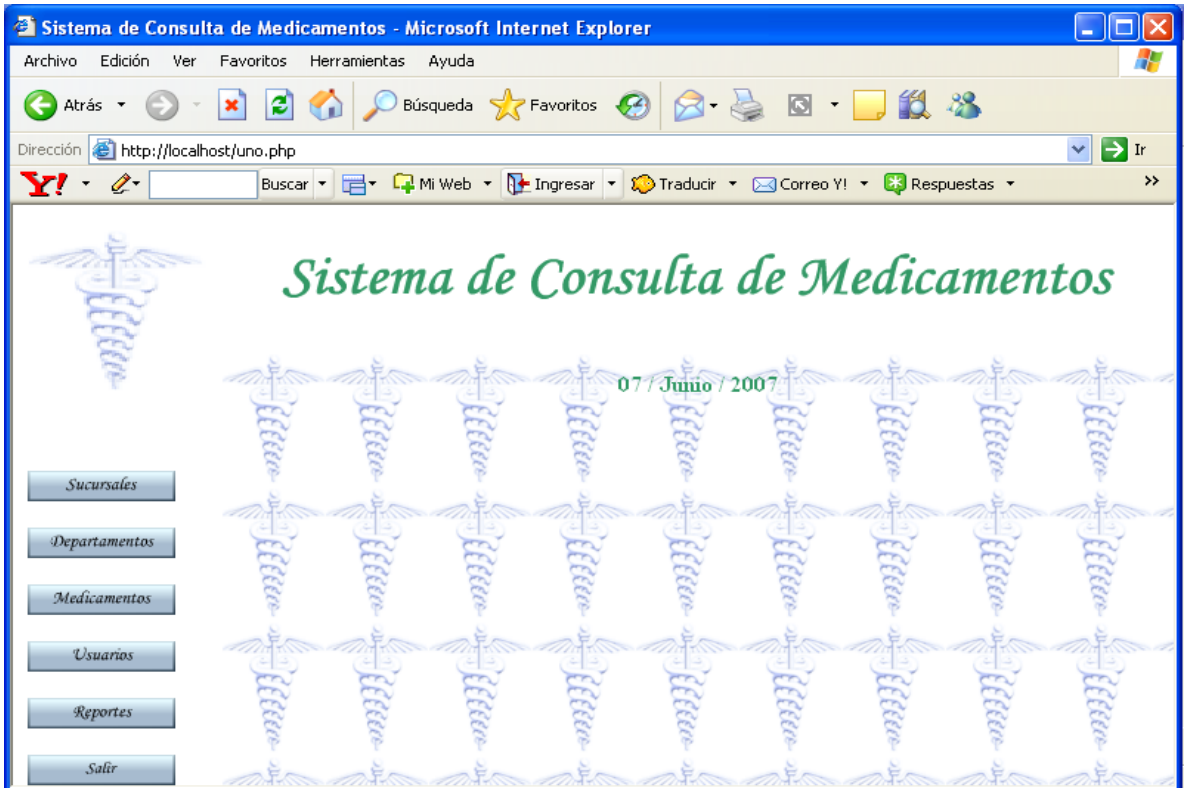
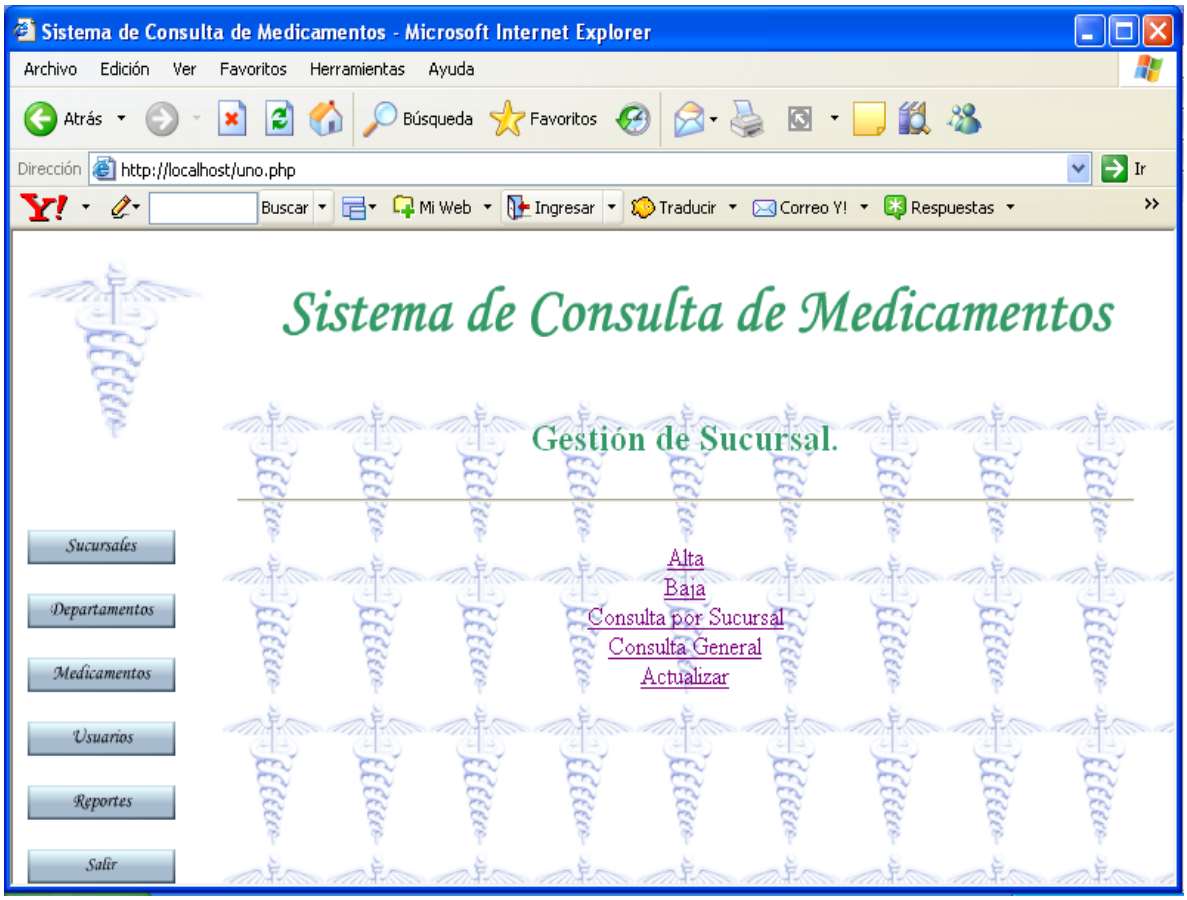


Figura 4.4 Menú Administrador.

4.1.1. Gestión de Sucursal, Departamentos, Medicamentos y Usuarios.

Estas opciones del menú presentan formatos similares en sus interfaces, así como también funciones similares como son: alta, baja, actualización y consulta de cada de una de estas opciones. Para evitar redundar en ellas solo hacemos mención a la opción **"Sucursales"** que se describe a continuación.

Al seleccionar la opción **"Sucursales"** se mostrará un submenú en la página principal, que comprende alta, baja, consulta (por sucursal ó general) y actualización de una sucursal, esto representa la gestión de sucursal. Si se requiere salir de este submenú dar clic en otra opción del menú principal o bien si desea salir del sistema por completo deberá dar clic en el botón **"Salir"**, esto se muestra en la figura 4.5.



4.5 Gestión de Sucursal.

4.1.1.1. Interfaz Alta Sucursal.

Al seleccionar la opción **"Alta"** se muestra en la página principal la solicitud de información para llevar acabo el registro de una nueva sucursal, una vez proporcionado los datos se debe pulsar el botón **"Dar alta"** para que la información sea almacenada en la base

de datos, para regresar a la gestión de sucursal únicamente dar clic en la liga **"Regresar"** (Figura 4.6).

Figura 4.6 Registro de Sucursal.

Una vez que la Sucursal fue dada de alta se notifica mediante un mensaje que el registro fue realizado exitosamente (Figura 4.7).

Figura 4.7 Mensaje de Alta Exitoso.

Si se intenta realizar el registro de una Sucursal de la forma como se muestra en la figura 4.8, esta no será registrada ya que todos los campos son obligatorios y se enviará un mensaje para indicarlo (Figura 4.9).

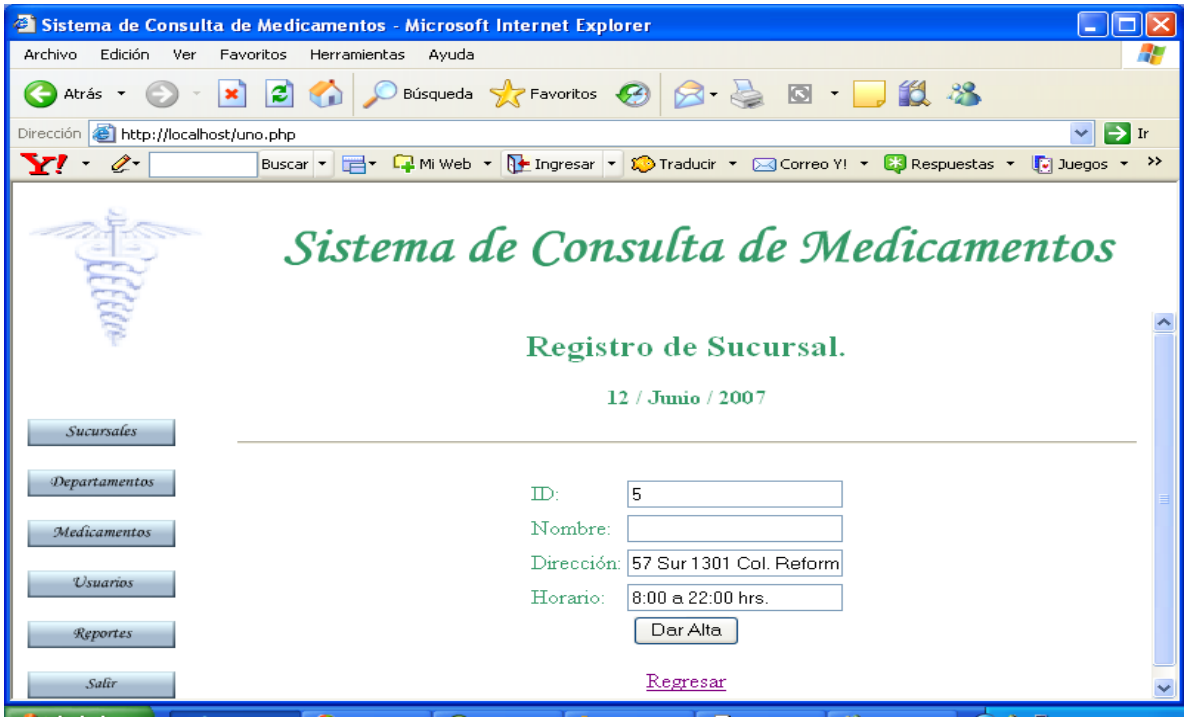


Figura 4.8 Alta de Sucursal Incorrecto.

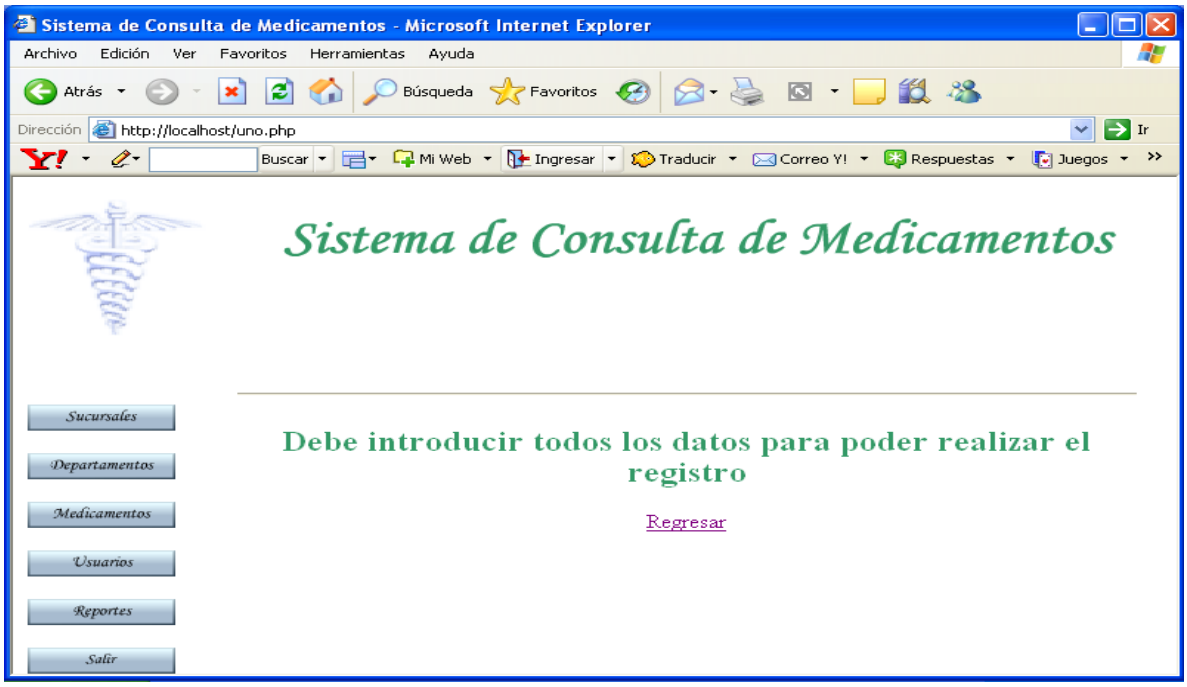


Figura 4.9 Mensaje de Error.

Si se intenta dar de alta una Sucursal que ya existe, esto se identifica con el ID de la Sucursal, se mandará un mensaje para indicar el evento (Figura 4.10). Por ejemplo la Sucursal con ID 3 ya fue registrada anteriormente, si intentamos registrar otra Sucursal con el mismo ID no será permitido.

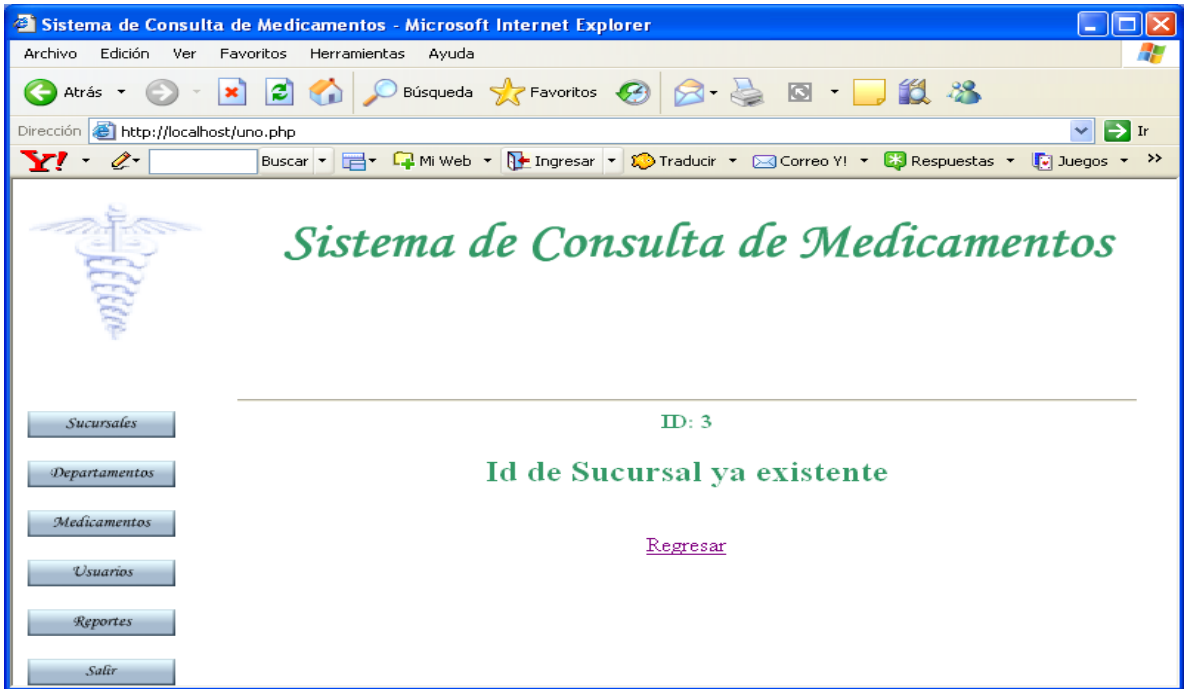


Figura 4.10 Mensaje de ID de Sucursal Repetido.

Para comprobar que las Sucursales fueron dadas de alta se realiza mediante consultas de las mismas, que mas a adelante se explican la forma de llevarlas a cabo.

4.1.1.2. Interfaz Baja Sucursal.

Al seleccionar la opción **"Baja"** se muestra en la página principal la interfaz de la Figura 4.11, la cual muestra las sucursales existentes en la base de datos, se tiene que seleccionar la que se va a eliminar y posteriormente dar clic en el botón **"Eliminar"**, al eliminar una sucursal se eliminan departamentos y medicamentos existentes en ella. De igual manera para regresar a la gestión de sucursal únicamente dar clic en la liga **"Regresar"**.

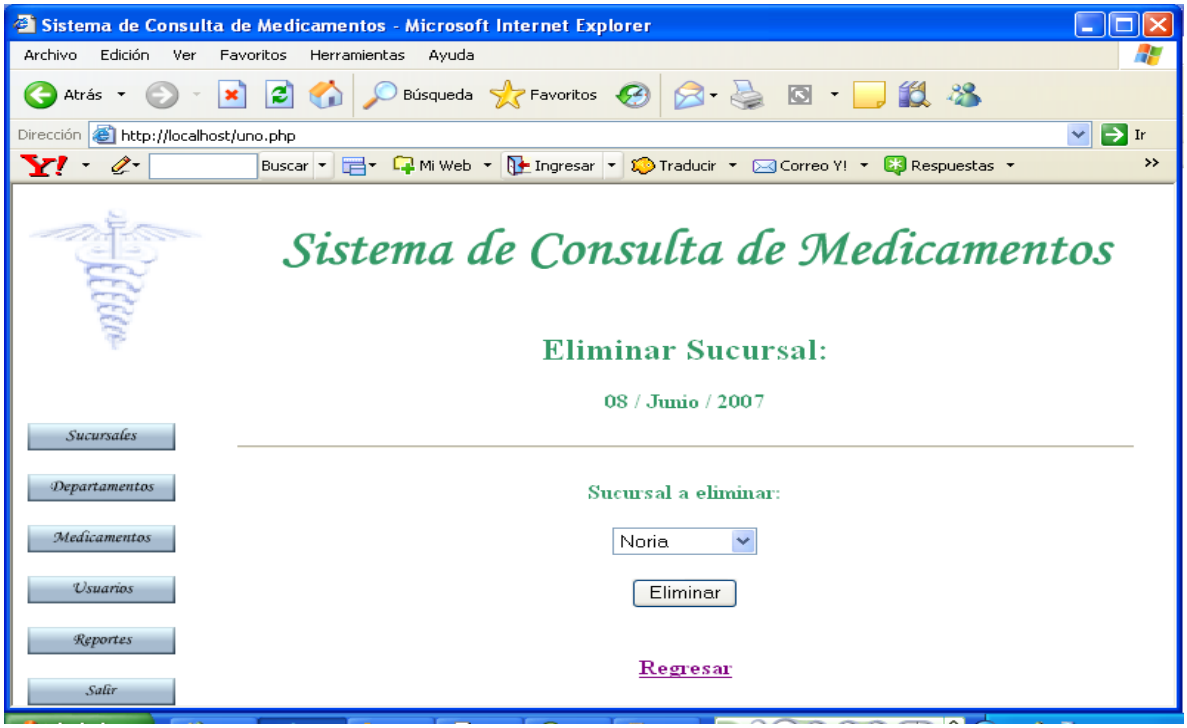


Figura 4.11 Baja de Sucursal.

Una vez que la Sucursal fue eliminada se notifica mediante un mensaje como se muestra en la Figura 4.12.

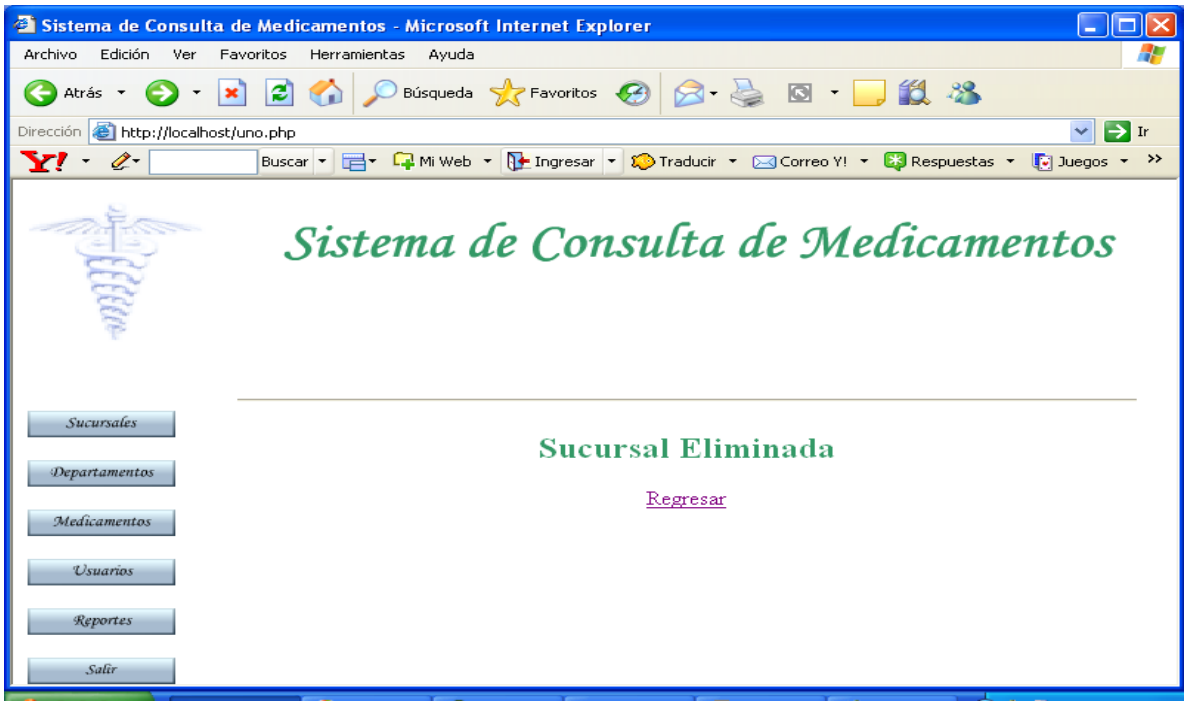


Figura 4.12 Mensaje de Baja.

4.1.1.3. Interfaz Consulta por Sucursal.

Al seleccionar esta opción se despliega una interfaz como se muestra en la Figura 4.13, que despliega el nombre de las sucursales existentes en la base de datos, se debe seleccionar una y posteriormente pulsar sobre el botón **"Consultar"** para ver la información referente a la Sucursal. Para volver solo dar clic en la liga **"Regresar"**.

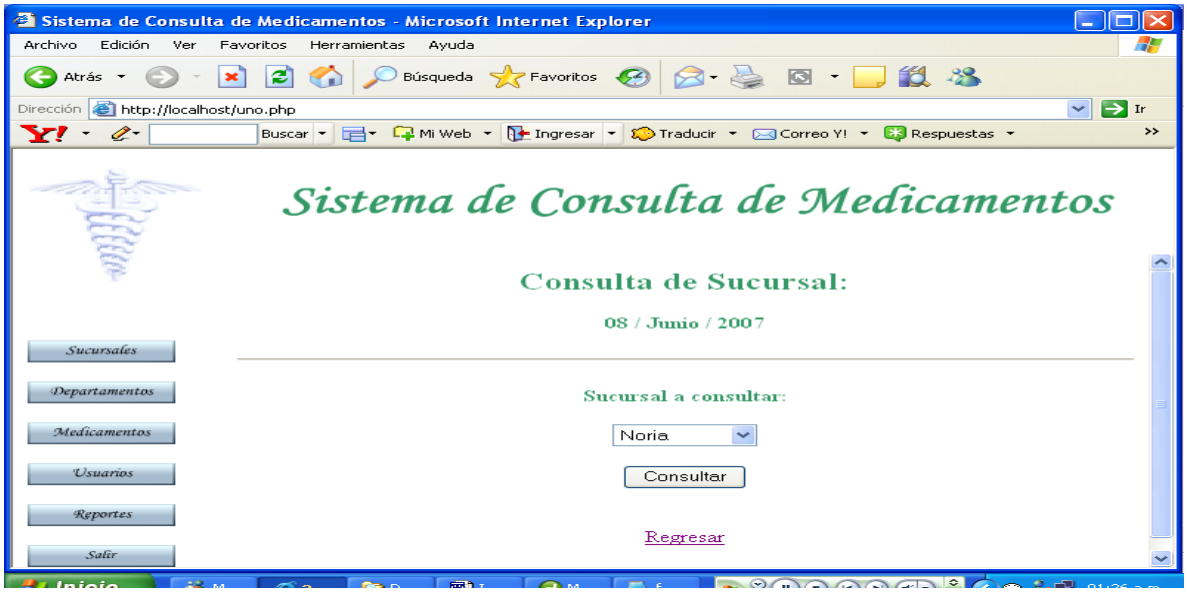


Figura 4.13 Consulta por Sucursal.

Como ejemplo, para mostrar los datos de la Sucursal "Noria" se despliega la interfaz de la figura 4.14.

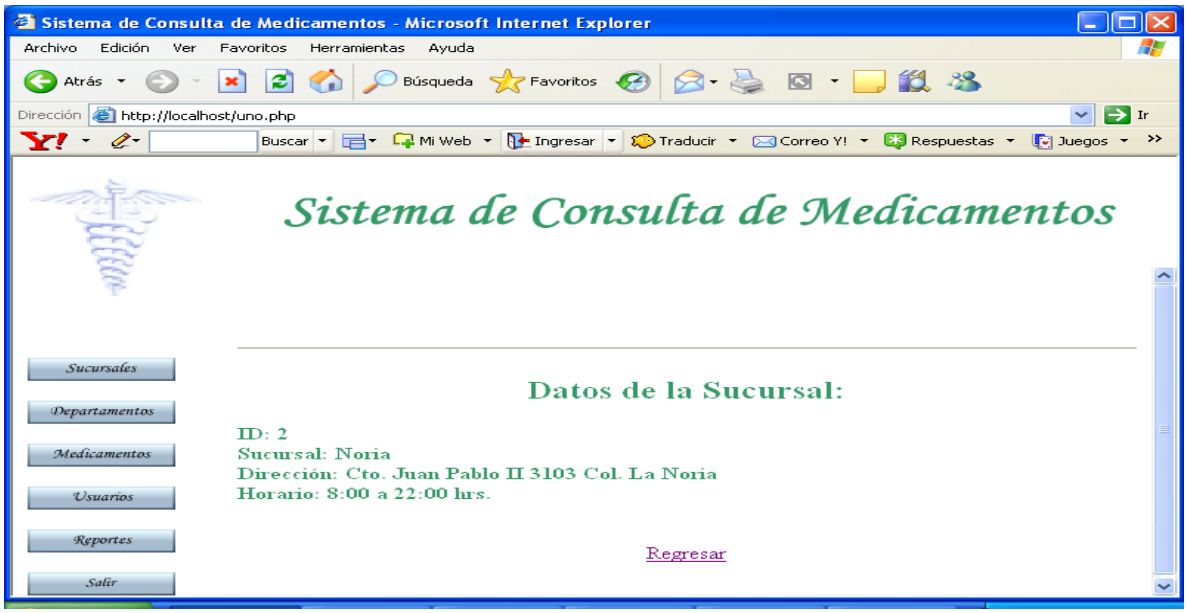


Figura 4.14 Datos de la Sucursal.

4.1.1.4. Interfaz Consulta General.

Al seleccionar esta opción se despliega una interfaz que muestra las sucursales existentes en la base de datos junto con la información de cada una de ellas. Para volver solo dar clic en la liga **"Regresar"**, como se muestra en la Figura 4.15.

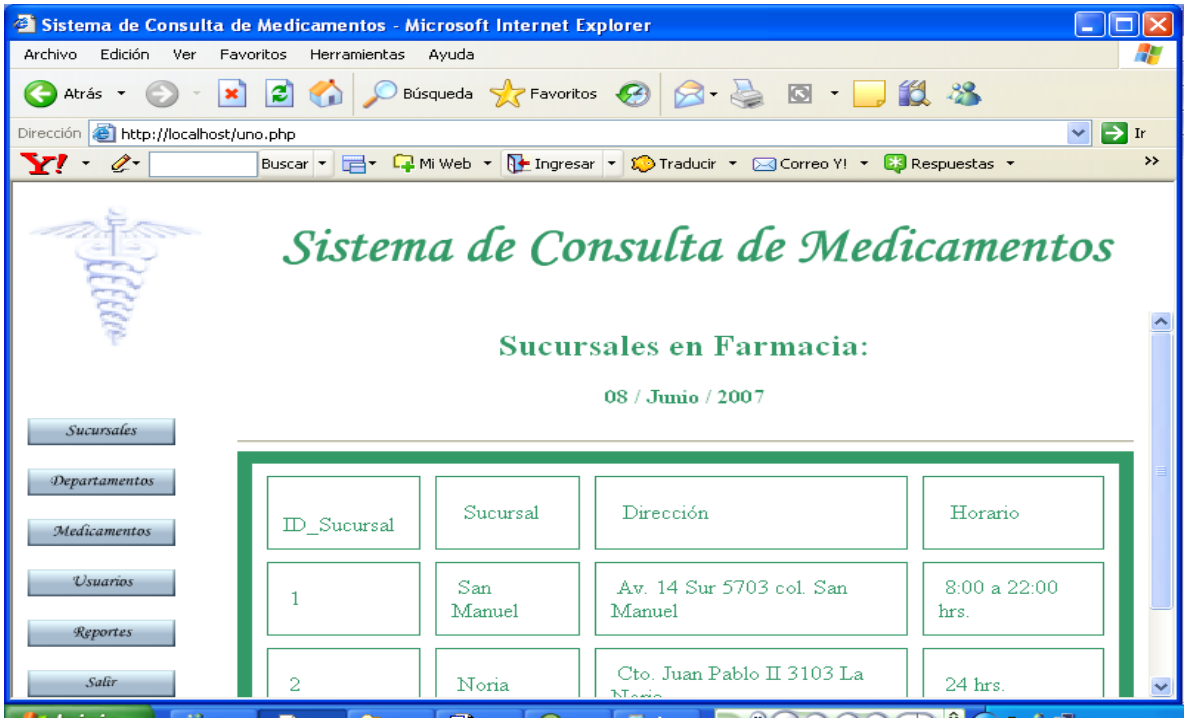


Figura 4.15 Consulta General.

4.1.1.5. Interfaz Actualizar Sucursal.

Al seleccionar esta opción se despliega una interfaz que muestra el nombre de las sucursales existentes en la base de datos, se debe elegir una de ellas para actualizar, posteriormente se deben llenar los campos con la nueva información y por último se debe pulsar el botón **"Actualizar"** para realizar los cambios, es este caso todos los campos requeridos son obligatorios, si alguno no se llena se envía un mensaje para solicitar sea llenado. Para volver solo dar clic en la liga **"Regresar"**. Esta opción es similar a la de **"Alta"** por ello solo se muestra la primera interfaz (Figura 4.16).

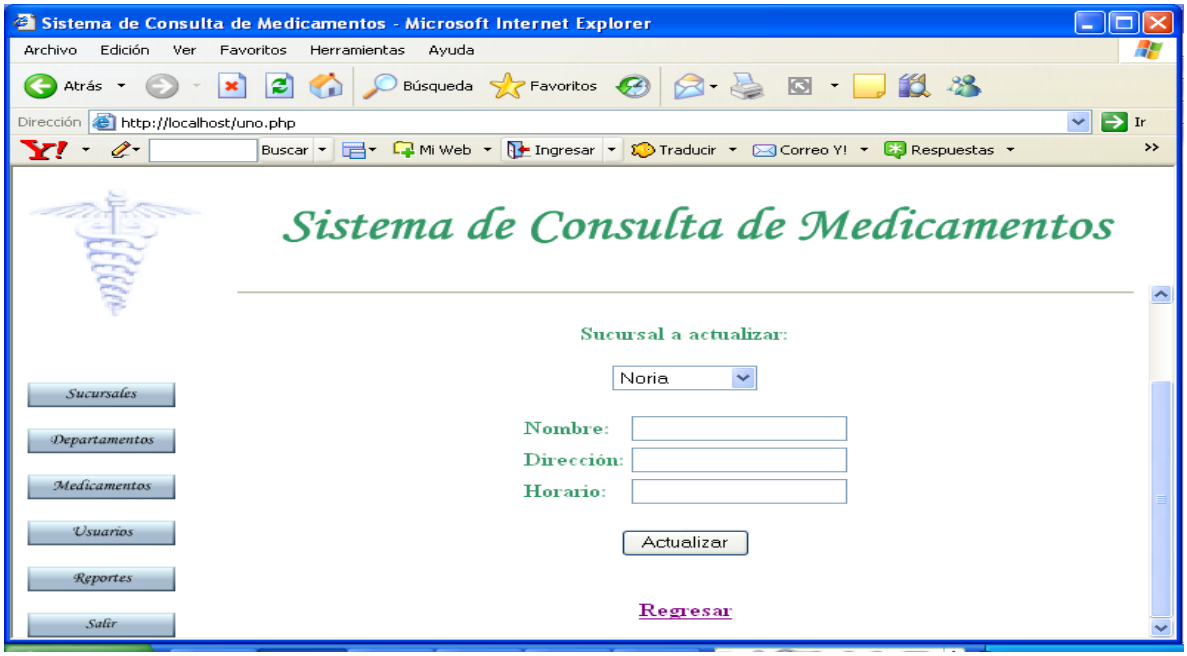


Figura 4.16 Actualizar Sucursal.

4.1.2. Interfaz Reporte.

Al seleccionar esta opción se muestra la interfaz de la figura 4.17, para poder generar el reporte de consultas realizadas por sucursal, únicamente se tiene que seleccionar el número de Sucursal y después pulsar sobre el botón **"Informe"**.

Figura 4.17 Reporte de Consulta.

Una vez seleccionada la Sucursal se despliega una interfaz con los datos de las consultas realizadas en ella como se muestra con el ejemplo de la Sucursal con ID 1 en la Figura 4.18.

Figura 4.18 Contenido del Reporte.

4.2 Menú Usuario (Cliente ó Empleado).

Si el usuario identificado es un cliente ó empleado, se muestra la página de bienvenida al sistema (Figura 4.19) y que continúa con la presentación de una nueva página que despliega un menú, en éste se muestran datos personales, medicamento, sucursales, lo único que se tiene que hacer es dar un clic en la opción deseada y si requiere salir del sistema debe dar clic sobre el botón **"Salir"**, como se muestra en la Figura 4.20.

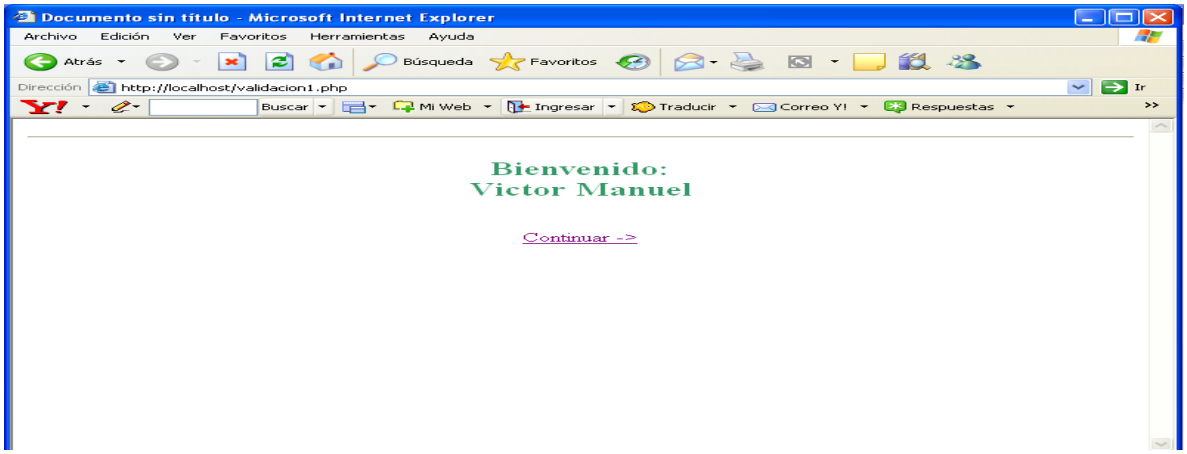


Figura 4.19 Bienvenida al Cliente ó Empleado.

4.20 Menú Cliente ó Empleado.

4.2.1. Interfaz Datos Personales.

Al seleccionar esta opción se muestra en la página principal los datos personales del usuario, así como también la opción de modificar su Password (Figura 4.21), si desea tomar en cuenta esta opción basta con dar clic sobre el botón **"Cambiar Password"** y el sistema cargará una nueva interfaz en la cual solicita introducir el nuevo dato para realizar el cambio (Figura 4.22). Si no desea llevar acabo ninguna modificación puede elegir una nueva opción del menú principal o bien salir del sistema.

Figura 4.21 Datos Personales.

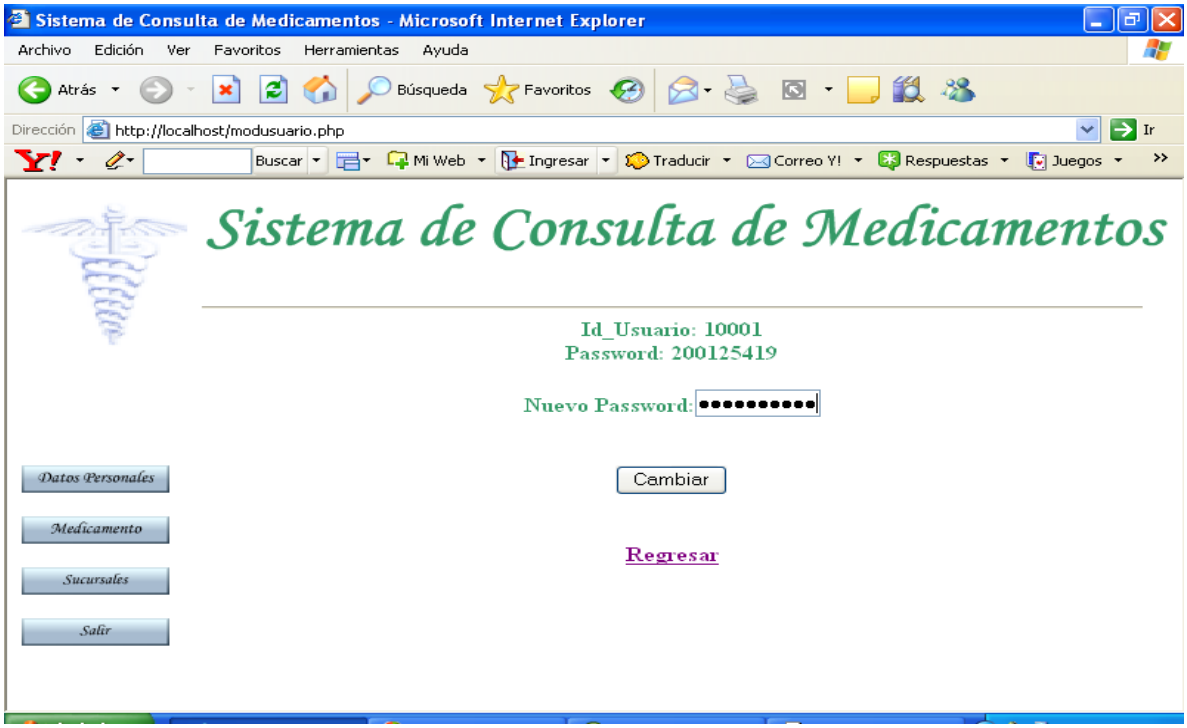


Figura 4.22 Cambio de Password.

Después de haber proporcionado su nuevo password se muestra un mensaje que fue cambiado, además de mostrar el nuevo para su posterior acceso al sistema (Figura 4.23).

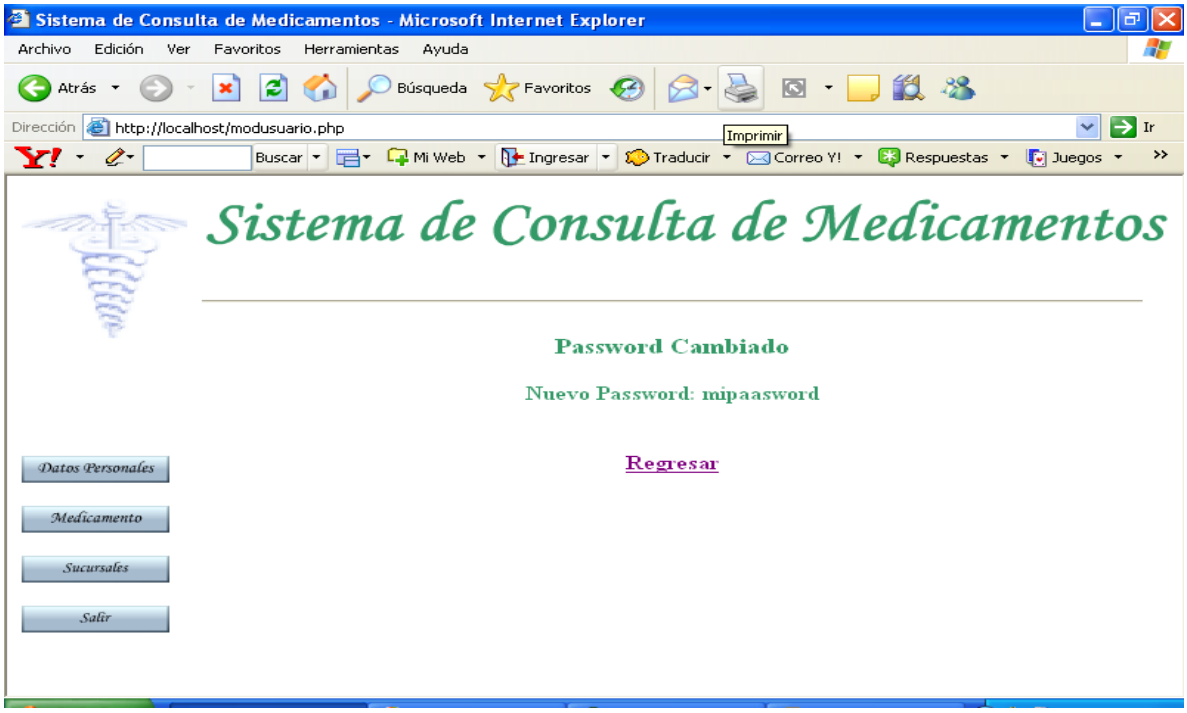


Figura 4.23 Password Cambiado.

4.2.2. Interfaz Medicamento.

Al seleccionar esta opción el sistema despliega la interfaz para realizar una consulta de un Medicamento existente en la base de datos, la consulta se lleva acabo directamente mediante la selección de el nombre del medicamento y posteriormente se da clic sobre el botón **"Consultar"** como se muestra en la Figura 4.24.

Figura 4.24 Consulta por Medicamento.

El despliegue de información de un medicamento seleccionado se realiza como se muestra en la Figura 4.25.

Figura 4.25 Información de Medicamento.

4.2.3. Interfaz Sucursales.

Esta opción es igual de consulta, pero aquí ésta se realiza mediante la búsqueda de un medicamento por sucursal, primero se muestra la interfaz de la Figura 4.26, en la cual se despliegan las sucursales existentes en la base de datos.

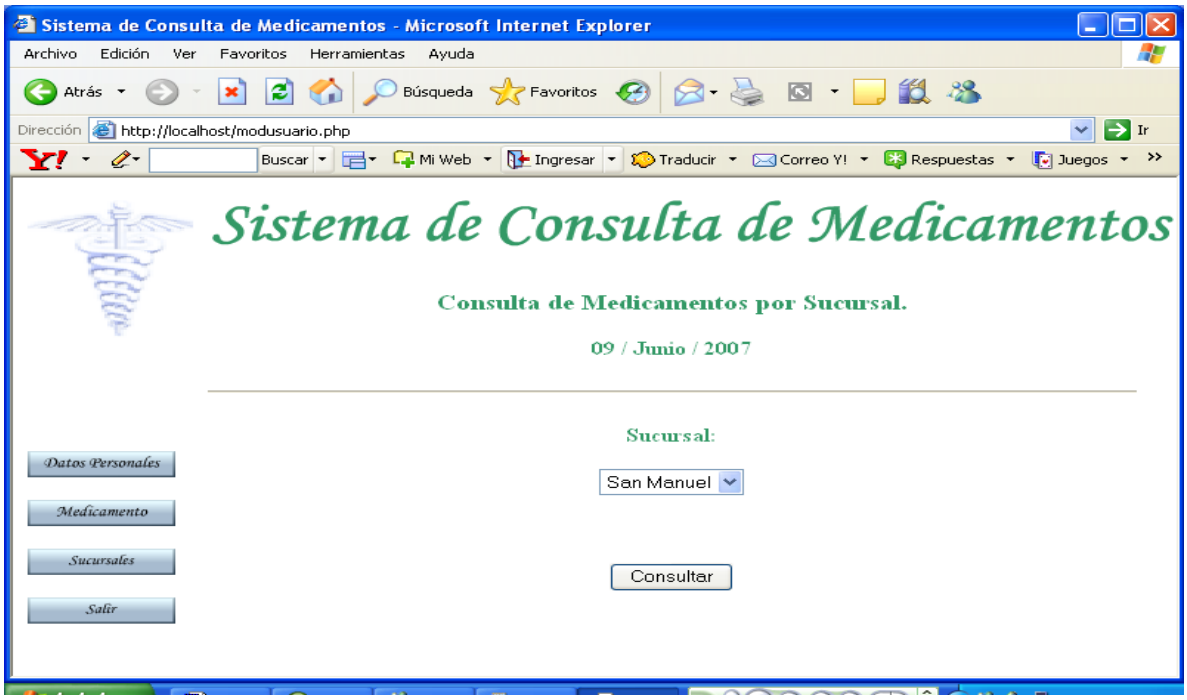


Figura 4.26 Consulta por Sucursal.

Si elige el nombre de una sucursal y posteriormente da clic en el botón **"Consultar"**, el sistema despliega la información de la sucursal en una nueva interfaz junto con un campo de selección que contiene los Departamentos existentes en la Sucursal (Figura 4.27).

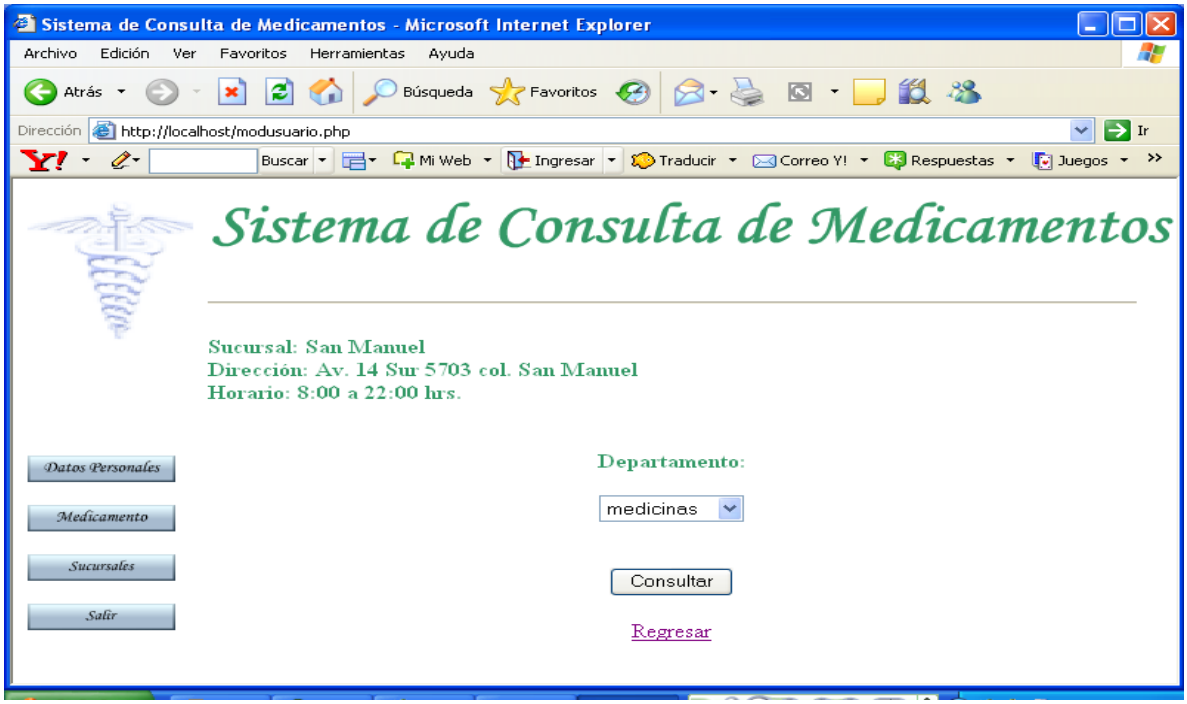


Figura 4.27 Departamentos en Sucursal.

Si selecciona el nombre del departamento y da clic sobre el botón **"Consultar"** el sistema carga la interfaz de la Figura 4.28 en la que despliega los medicamentos existentes en dicho departamento, se debe seleccionar el nombre del medicamento y dar clic sobre el botón **"Consultar".** El despliegue de la información de un medicamento es similar a la interfaz de la figura 4.25.

Figura 4.28 Medicamentos en Departamento.

CONCLUSIONES.

Producto del análisis, diseño e implementación se cumplió con el objetivo principal que se requería en este proyecto y de esta forma cubrir una de las necesidades en las farmacias.

El Sistema desarrollado tiene las capacidades de los sistemas comerciales que realizan tareas de manejo de inventarios y consulta, pero estas funciones han sido adecuadas de acuerdo a las necesidades previstas en este trabajo de tesis. Por otra parte se logró un sistema robusto y confiable de bajo costo.

Para el desarrollo del presente trabajo se pusieron en práctica los conocimientos adquiridos en el transcurso de mi formación en la carrera de Licenciatura en Ciencias de la Computación, principalmente en el área de Ingeniería de Software y Base de Datos.

Se utilizó una Base de Datos Relacional ya que este es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Además de que está basado en la teoría de las matemáticas, que las hacen tan predecibles, seguras y robustas.

La importancia del modelo Cliente/Servidor, radica en la necesidad que tienen las organizaciones en realizar sus operaciones eficientemente, debido a la competencia constante, a la necesidad de que se reduzcan costos, al mismo tiempo que se generen productos y servicios rápidamente y con mejor calidad.

Se requirió el aprendizaje de lenguajes de desarrollo de software, como MySQL para la gestión de datos y PHP para que funcione el Sistema en un entorno Web. Además se utilizaron otras herramientas de programación durante el desarrollo del proyecto como: Macromedia Dreamweaver 8, Microsoft Visio, entre otros.

La razón por la cual se decidió utilizar MySQL, PHP y APACHE, es debido a que este software puede funcionar tanto en Sistemas Operativos Linux y Sistemas Operativos Windows, además de que su principal ventaja es que es software libre, además proporciona confiabilidad a las aplicaciones desarrolladas bajo este ambiente.

PERSPECTIVAS.

El sistema desarrollado tiene la posibilidad de permitir mejoras, se sugieren algunas:

Ya que el sistema se enfoca principalmente a la consulta de medicamentos, podría adaptársele la función de venta de medicamentos en línea, o bien apartado del mismo.

Se podría informar sobre los medicamentos que solo se venden con receta médica o con alguna otra restricción mediante un mensaje de advertencia al cliente.

Otra mejora que se podría realizar en este sistema en cuanto a las consultas y reportes, es la creación de un formato de impresión y contar con dispositivo de impresión para realizarlos.

BIBLIOGRAFÍA:

[1] Abraham Silberschatz, Henry F. Korth, S. Sudarshan., "*Fundamentos de Bases de Datos"*. Mc Graw Hill.

[2] Boone Rojas M. "Ingeniería del Software", Diplomado en Bases de Datos, Facultad de Ciencias de la Computación. BUAP 2006.

[3] I. Jacobson, G. Booch, J. Rumbaugh, "El Proceso Unificado de Desarrollo de Software". Ed. Addison Wesley.

[4] Moyao Martínez Y. "Análisis y Diseño de Bases de Datos", Diplomado en Bases de Datos, Facultad de Ciencias de la Computación. BUAP 2006.

[5] Perdita Stevens, Rob Pooley. "*Utilización de UML en Ingeniería del Software con objetos y componentes"*, Ed. Addison-Wesley.

[6] Pressman, R. S., "*Ingeniería del Software: Un enfoque práctico"*. Cuarta edición. Ed. McGraw-Hill.

[7] Rossainz López M. "Notas de Clase, Ingenieria de Software Avanzada", Facultad de Ciencias de la Computación. BUAP, primavera 2006.

[8] Somodevilla García M. "Desarrollo de Aplicaciones de Bases de Datos Locales con MySQL", Diplomado en Bases de Datos, Facultad de Ciencias de la Computación. BUAP 2006.

[9] Ullman, L., "*MySQL"*. Ed. Prentice Hall.

[10] Valade, J., "*PHP 5 para dummies"*, St Editorial.

Referencias en Internet:

http://macine.epublish.cl/tesis/index-A_1_.html
http://mysql.conclase.net/
http://www.ingenierosoftware.com/analisisydiseno/casosdeuso.php
http://www.programacion.net/html/tutorial/curso/
http://www.webtaller.com/construccion/lenguajes/info/manuales/php/
http://www.desarrolloweb.com/