UNIVERSIDAD TÉCNICA DE BABAHOYO

FACULTAD DE ADMINISTRACIÓN, FINANZAS E INFORMÁTICA

PROCESO DE TITULACIÓN

OCTUBRE 2018–MARZO 2019

PROPUESTA TECNOLÓGICA DE GRADO O DE FIN DE CARRERA

PRUEBA PRÁCTICA

INGENIERÍA EN SISTEMAS

PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS

TEMA:

Sistema Informático de Facturación de citas y detalle de tratamientos médicos en el Hospital del Día "Gabriel Andrade" del Cantón Ventanas.

EGRESADO:

Mejía Guanoluisa Kevin Jim

TUTOR:

Ing. Miguel Ángel Zúñiga Sánchez

AÑO 2019

ÍNDICE

ÍNDICE I

ÍNDICE DE GRÁFICOS II

ÍNDICE DE ILUSTRACIONES III

ÍNDICE DE TABLAS III

ÍNDICE DE FOTOS IV

INTRODUCCIÓN 1

CAPÍTULO I 1

1. Diagnóstico de necesidades y requerimientos 1

1.1. Ámbito de Aplicación: descripción del contexto y hechos de interés 1

1.1.1. Planteamiento del problema 2

1.1.2. Localización del problema objeto de estudio 2

1.1.3. Problema general 2

1.2. Establecimiento de requerimientos 2

1.2.1. Metodología 3

1.2.2. Arquitectura 3

1.2.3. Patrón de desarrollo 4

1.2.4. Requerimientos mínimos de funcionalidad 4

1.2.4.1. Requisitos de hardware 4

1.2.4.2. Requisitos de software 4

1.3. Justificación del requerimiento a satisfacer 4

CAPITULO II 6

2. Desarrollo del prototipo tecnológico 6

2.1. Definición del prototipo 6

2.2. Fundamentación de teorías del prototipo. 6

2.2.1. Metodología.....6

2.2.1.1. UML.....7

2.2.2. Arquitectura.....8

2.2.3. Tecnologías 9

2.2.3.1. Tecnologías usadas para el desarrollo.....9

Modelo físico.....28

CAPÍTULO III 49

3. Evaluación del prototipo.....49

3.1. Plan de evaluación. 49

3.2. Resultados de la evaluación. 50

3.2.1. Prueba de caja blanca.....50

BIBLIOGRAFÍA.....53

CONCLUSIONES.....54

ANEXOS 56

ÍNDICE DE GRÁFICOS

Gráfico 1 .- Funcionalidad de la Aplicación: Administrador. 14

Gráfico 2.- Funcionalidad de la Aplicación: Otros usuarios. 14

Gráfico 3.- Diagrama de Clases 15

Gráfico 4.- Diagrama de actividades.....16

Gráfico 5.- Diagrama caso de uso ingreso al sistema 17

Gráfico 6.- Diagrama caso de uso registro de nuevo servicio. 17

Gráfico 7.- Diagrama caso de uso registro de nuevo paciente. 18

Gráfico 8.- Diagrama caso de uso actualización de paciente. 18

Gráfico 9.- Diagrama caso de uso registro de servicios 19

Gráfico 10.- Diagrama caso de uso registro de médico 19

Gráfico 11.- Diagrama caso de uso consulta de historial 20

ÍNDICE DE ILUSTRACIONES

Ilustración 1 Pantalla de Inicio de Sesión 42

Ilustración 2 Pantalla de inicio del administrador 43

Ilustración 3 Pantalla de inicio de usuario con funciones específicas. 43

Ilustración 4 Pantalla gestión de usuarios y permisos. 44

Ilustración 5 Pantalla gestión de servicios y facturación. 44

Ilustración 6 Pantalla gestión de pacientes. 45

Ilustración 7 Pantalla gestión de servicios. 45

Ilustración 8 Pantalla gestión de doctores. 46

Ilustración 9 Pantalla consulta de historial. 46

Ilustración 10 Pantalla de reportes. 47

Ilustración 11 Crear clases de pruebas 51

Ilustración 12 Selección de complemento de testeo 51

Ilustración 13 Clases generadas 52

Ilustración 14 Resultado del Testeo 52

ÍNDICE DE TABLAS

Tabla 1.- **tbbotones**.....24

Tabla 2.- **tbdoctores**.....24

Tabla 3.- **tbespecialidades**.....25

Tabla 4.- **tbfacturacion**.....25

Tabla 5.- **tbhimedico**.....26

Tabla 6.- **tbpacientes**.....26

Tabla 7.- **tbservicios**.....27

Tabla 7.- **tbsubmodulos**.....27

Tabla 8.- **tbusuarios**.....27

ÍNDICE DE FOTOS

Foto 1.- Hospital del Día.....58

Foto 2.- Recepción.....58

Foto 3.- Recepcionistas.....59

Foto 4.- Director del Hospital.....59

Foto 5.- El director con una de sus pacientes.....60

Foto 6.- Entrevista al director.....60

INTRODUCCIÓN

La tecnología está avanzando en la actualidad muy rápido, y es muy común que la mayoría de las empresas deseen automatizar sus procesos, para reducir el tiempo de atención a sus clientes y mejorar la calidad de los servicios que estas ofrecen. Es por eso que la tecnología es muy importante hoy en día.

El Hospital del Día "Gabriel Andrade" ubicado del Cantón Ventanas, provincia de Los Ríos, no se quiere quedar atrás, ya que se desea desarrollar un sistema Informático de Facturación de citas y detalle de tratamientos médicos, para que a los pacientes les notifique por correo electrónico y mensajes SMS, todas las incidencias que realizan en centro salud mencionado.

El presente proyecto se ha realizado con el fin de desarrollar un sistema, el cual ayudará de una manera sistematizada a una entidad de salud privada. La idea de este proyecto viene de generar facturación de los tratamientos médicos que se realizan en el Hospital del Día "Gabriel Andrade", donde se trata de llevar un control veraz y preciso de economía de la clínica, dando confianza, rapidez y seguridad y así poder mejorar la atención del servicio del hospital.

El software va a facilitar el trabajo administrativo, así poder brindarle mejor calidad de servicio, y a la vez que tanto los funcionarios y beneficiarios les generen conformidad. Con este proyecto se pretende dar la solución a los problemas comunes al momento de dar una factura. Este prototipo permitirá el acceso a la información de todos los médicos autorizados de manera rápida, fácil y oportuna de la mencionada clínica.

La línea de investigación del presente proyecto es desarrollo de software relacionado con la sublínea desarrollo de aplicaciones informáticas.

CAPÍTULO I

1. Diagnóstico de necesidades y requerimientos

1. Ámbito de Aplicación: descripción del contexto y hechos de interés

La salud es un derecho fundamental que tienen todos los ciudadanos ecuatorianos, que debe cumplir una serie de requisitos para garantizar el bienestar de las personas. Este servicio debe ser de calidad y debe estar acorde a las necesidades de la sociedad actual. Es por ello que la mayoría de los centros de salud se está implementado la tecnología para automatizar sus procesos con el objetivo de brindar un servicio de calidad y en un menor tiempo posible.

El problema que presenta Hospital del Día "Gabriel Andrade" del Cantón Ventanas es la falta de un sistema informático que permita realizar la facturación electrónica de las citas médicas y el notificar el detalle de los tratamientos que se realizan en la institución, lo que provoca que el tiempo de atención sea más de lo esperado.

La falta de innovación provoca que las empresas se queden en una brecha tecnológica, que no les permite adaptarse a las tendencias actuales. Las dificultades que provoca la ausencia de tecnología en los negocios hacen que se queden estancadas en procesos manuales tediosos, tanto para los administrativos como para la clientela en general. (BALLESTEROS JARAIZ, y otros, 2018)

En la actualidad, hay muchos instrumentos tecnológicos que permiten automatizar casi todo tipo de procesos para resolver estos problemas, y sobretodo que las organizaciones como las de la salad, puedan brindar servicios de calidad a sus clientes para que estos queden satisfechos.

La presente propuesta proyecta desarrollar un Sistema Informático de Facturación de citas y detalle de tratamientos médicos en el Hospital del Día "Gabriel Andrade" del Cantón Ventanas, el cual automatizará los procesos de facturación y detalle de tratamientos los cuales serán notificados al correo electrónico del paciente.

1. Planteamiento del problema

Realizar procesos de manera manual los procesos de facturación de las citas médicas en el Hospital del Día "Gabriel Andrade" del Cantón Ventanas provoca que largas espera por parte de los pacientes y que el tiempo de atención largo y tedioso, teniendo como consecuencia el descontento tanto del personal que atiende como el de los pacientes.

Por ello, el Hospital del Día "Gabriel Andrade" del Cantón Ventanas está buscando soluciones para optimizar estos procesos que aún se realizan de manera manual, se puedan automatizar, para que sus pacientes se sientan cómodo con la atención que se brinda en este centro de salud particular.

A raíz que auge tecnológico crece, las instituciones dedicadas a la salud quieren migrar todos sus procesos que se realizan de manera analógica, implementando equipos informáticos y adquiriendo software acorde a sus necesidades. El Hospital del Día "Gabriel Andrade" del Cantón Ventanas, pretende que sus pacientes estén informados de los valores a pagar de su factura por medio de correo electrónico.

2. Localización del problema objeto de estudio

La investigación se localiza geográficamente en la provincia de Los Ríos; Cantón Ventanas; en las calles 9 de octubre entre 10 de noviembre y Héctor Cabrera en el Hospital del Día "Gabriel Andrade".

3. Problema general

¿Cómo mejorará el tiempo de atención de los pacientes, con el desarrollo de un Sistema Informático de Facturación de citas y detalle de tratamientos médicos en el Hospital del Día "Gabriel Andrade" del Cantón Ventanas?

2. Establecimiento de requerimientos

Una vez obtenido los inconvenientes que están presente en Hospital del Día "Gabriel Andrade" del Cantón Ventanas, se desarrollará un Sistema Informático de Facturación de citas y detalle de tratamientos médicos, el cual tendrá las siguientes funcionalidades:

Administrador del sistema (Gerente del Hospital del Día)

1. Administración y gestión de usuarios (Doctores, Secretaria).

2. Crear nuevos usuarios (Doctores, Secretaria).

3. Modificar información de los usuarios (Doctores, Secretaria).

4. Cambiar el estado de los usuarios (activo, inactivo).

5. Cambiar perfil de los usuarios (Doctores, Secretaria).

6. Buscar usuarios (Doctores, Secretaria).

Secretaria.

1. Ingresar datos de los pacientes.

2. Registrar especialidades de los Doctores.

3. Ingresar datos para la respectiva facturación.

1. Metodología

La metodología de investigación del presente prototipo tecnológico, es de tipo cualitativa, porque se procedió a la visita al hospital, donde se utilizó como herramienta de investigación la entrevista. Con esta entrevista hecha directamente al gerente del hospital, obtuvo los requerimientos de software frente a las necesidades que presenta la institución.

Además de utilizar esta metodología de desarrollo de software, se utilizó como metodología de investigación a la observación (Ver Anexo I), porque se procedió a hacer una visita al hospital y se pudo notar que esta cuenta con equipamiento tecnológico, pero no cuenta con un sistema con el cual se pueda gestionar los procesos administrativos de la institución.

La metodología de desarrollo será orientada a objetos, ya que se utilizará como instrumento de diagramación del sistema a UML, y como lenguaje de programación JAVA, los cuales permiten aplicar esta metodología. Gracias a estas herramientas tecnológicas ayudan a modular al presente prototipo en relación al mundo real lograr la solución más óptima y abstracta.

2. Arquitectura

La arquitectura del presente prototipo será cliente/servidor, ya que la aplicación funcionará con una base de datos que estará instalada en otro equipo. La aplicación de escritorio cliente, hará uso de la base datos, con el objetivo de mantener la información persistente y descentralizada.

3. Patrón de desarrollo

El patrón de desarrollo será modelo, vista, controlador (M.V.C.), el cual se adapta muy bien con el lenguaje a utilizar en el presente prototipo como lo es JAVA. Este patrón de desarrollo separa la lógica de negocio (modelo), de la interfaz gráfica (vista), donde un intermediario (controlador) se encarga de ambos se comuniquen.

4. Requerimientos mínimos de funcionalidad

1. Requisitos de hardware

1. Memoria RAM: 2 GB.
2. Espacio en Disco Duro: 250 GB.
3. Procesador X86 o X64 de 1.8 GHz

1. Requisitos de software

4. Sistema operativo de escritorio Windows (7, 8.1, 10).
5. Ambiente de ejecución de JAVA (JRE 8).

1. Justificación del requerimiento a satisfacer

En el Hospital del Día "Gabriel Andrade" del Cantón Ventanas, los procesos de facturación y la notificación de los detalles de tratamientos médicos aún se realizan de forma manual. Con mucha frecuencia esto provoca larga colas de espera y a su vez esto provoca insatisfacción por parte de los pacientes debido a la demora del servicio brindado.

Durante la entrevista, con el director del hospital, se pudo identificar estos problemas, los cuales necesitan solucionarse a tiempo, porque de no ser así puede que los pacientes busquen otras alternativas, y el hospital pierda demanda, teniendo como consecuencia perdidas económicas.

Con el desarrollo de este prototipo se pretende automatizar estos procesos, para que tanto la administración del hospital como los pacientes se sientan conformes con el servicio que se brinda, donde se automatizará el procese de facturación, y los detalles de la factura como los de tratamientos médicos serán notificados por correo electrónico.

CAPITULO II

6. Desarrollo del prototipo tecnológico

1. Definición del prototipo

Sistema Informático de Facturación de citas y detalle de tratamientos médicos en el Hospital del Día "Gabriel Andrade" del Cantón Ventanas, será una aplicación de escritorio, que cumple las necesidades de los procesos de la institución, los cuales fueron encontrados mediante una entrevista al director del hospital.

El presente prototipo será una aplicación de Escritorio, la cual será instalada en cada uno los departamentos del hospital a requerir la aplicación. El sistema funcionará a través de la red, donde se usará una base de datos MySQL de forma descentralizada para el almacenamiento de los datos, es decir, con una arquitectura cliente servidor, donde los clientes gestionan los datos desde sus equipos.

El lenguaje de programación que con el cual se desarrolla este prototipo tecnológico es JAVA, usando las herramientas de diseño de interfaces gráficas JFrame, la cual tiene la ventaja de ejecutarse en diferentes sistemas operativos como lo pueden ser Windows, Linux o MacOS. Java, es un potente lenguaje de programación con grandes beneficios, ya que provee un gran número de herramientas y librerías que le brinda al desarrollador una manera ágil para crear programas informáticos de propósitos específicos.

Para la notificación a través de mensajería instantánea, se usará la plataforma Arduino, conjuntamente con un módulo denominado SIM900, con el cual se gestionará los mensajes de las citas y los valores a pagar de las mismas a los respectivos pacientes del hospital.

2. Fundamentación de teorías del prototipo.

El presente prototipo ofrecerá soluciones a los problemas de los procesos encontrados en el Hospital del Día "Gabriel Andrade" del Cantón Ventanas, para mejorar la calidad de servicio que ofrece la institución a la sociedad en general, y además impulsar el uso de la tecnología para que el hospital se encuentre acorde a las tendencias actuales

La interfaz del sistema, contará con un diseño atractivo, con la cual el usuario podrá utilizar el sistema de manera fácil, según los perfiles asignados al sistema, ellos podrán realizar las tareas específicas a partir de los procesos que se realicen en el hospital.

El sistema funcionará en red, es decir cualquier usuario que esté conectado a la red, podrá usar el sistema, ya que por este medio es que se registran la información administrada en el hospital hacia la base de datos. Para garantizar la seguridad de la información cada usuario debe usar credenciales de acceso para ingresar a las funciones que ofrece el sistema.

1. Metodología

La metodología utilizada en el presente proyecto de desarrollo de software es la metodología orientada a objetos, la cual es un enfoque de desarrollo de sistemas que fomenta y facilita la reutilización de componentes de software. Con esta metodología, un sistema informático puede desarrollarse sobre una base de componentes que permita la reutilización efectiva de los componentes existentes y facilite la compartición de sus componentes con otros sistemas. Mediante la adopción de esta, se puede lograr una mayor productividad, menores costos de mantenimiento y mejor calidad.

Esta metodología emplea el lenguaje de modelado unificado (UML) estándar internacional del Object Management Group (OMG). UML es un estándar de modelado para análisis y diseño de OO que ha sido ampliamente adoptado en la industria de TI. (ogcio, 2018)

1. UML

UML, abreviatura de Unified Modeling Language, es un lenguaje de modelado estandarizado que consiste en un conjunto integrado de diagramas, desarrollado para ayudar a los desarrolladores de sistemas y software a especificar, visualizar, construir y documentar los artefactos de los sistemas de software, así como para el modelado de negocios y Otros sistemas que no sean software.

El UML representa una colección de las mejores prácticas de ingeniería que han demostrado ser exitosas en el modelado de sistemas grandes y complejos. El UML es una parte muy importante del desarrollo de software orientado a objetos y del proceso de desarrollo de software. El UML utiliza principalmente notaciones gráficas para expresar el diseño de proyectos de software. El uso de UML ayuda a los equipos de proyectos a comunicarse, explorar posibles diseños y validar el diseño arquitectónico del software.

El objetivo de UML es proporcionar una notación estándar que pueda ser utilizada por todos los métodos orientados a objetos y seleccionar e integrar los mejores elementos de las notaciones precursoras. UML ha sido diseñado para una amplia gama de aplicaciones. Por lo tanto, proporciona construcciones para una amplia gama de sistemas y actividades (por ejemplo, sistemas distribuidos, análisis, diseño y despliegue de sistemas).

UML es una notación que resultó de la unificación de OMT de:

Técnica de modelado de objetos OMT [James Rumbaugh 1991]: fue la mejor para el análisis y los sistemas de información con uso intensivo de datos.

Booch [Grady Booch 1994] - fue excelente para el diseño y la implementación. Grady Booch había trabajado mucho con el lenguaje Ada y había sido un jugador importante en el desarrollo de técnicas orientadas a objetos para el idioma. Aunque el método Booch fue fuerte, la notación fue menos bien recibida (muchas de las formas de las nubes dominaron sus modelos, no muy ordenadas)

OOSE (Ingeniería de software orientada a objetos [Ivar Jacobson 1992]) - presentó un modelo conocido como casos de uso. Los casos de uso son una técnica poderosa para comprender el comportamiento de un sistema completo (un área donde OO tradicionalmente ha sido débil).

En 1994, Jim Rumbaugh, el creador de OMT, sorprendió al mundo del software cuando abandonó General Electric y se unió a Grady Booch en Rational Corp. El objetivo de la asociación era fusionar sus ideas en un método único y unificado (el título de trabajo para el método era de hecho el "Método Unificado").

En 1995, el creador de OOSE, Ivar Jacobson, también se había unido a Rational, y sus ideas (particularmente el concepto de "Casos de uso") se incorporaron al nuevo Método Unificado, ahora denominado Lenguaje de Modelado Unificado1. El equipo de Rumbaugh, Booch y Jacobson son conocidos cariñosamente como los "Tres Amigos". (Visual-Paradigm, 2017)

2. Arquitectura

Arquitectura cliente-servidor, arquitectura de una red informática en la que muchos clientes (procesadores remotos) solicitan y reciben servicio desde un servidor centralizado (computadora central). Las computadoras cliente proporcionan una interfaz para permitir que un usuario de computadora solicite servicios del servidor y muestre los resultados que el servidor devuelve.

Los servidores esperan que las solicitudes lleguen de los clientes y luego les responden. Idealmente, un servidor proporciona una interfaz transparente estandarizada a los clientes para que los clientes no tengan que conocer las características específicas del sistema (es decir, el hardware y el software) que proporciona el servicio.

Los clientes suelen estar ubicados en estaciones de trabajo o en computadoras personales, mientras que los servidores están ubicados en otros lugares de la red, generalmente en máquinas más potentes. Este modelo informático es especialmente efectivo cuando los clientes y el servidor tienen tareas distintas que realizan de manera rutinaria. En el procesamiento de datos hospitalarios, por ejemplo, una computadora cliente puede ejecutar un programa de aplicación para ingresar información del paciente mientras la computadora del servidor ejecuta otro programa que administra la base de datos en la que la información se almacena permanentemente.

Muchos clientes pueden acceder a la información del servidor simultáneamente y, al mismo tiempo, una computadora cliente puede realizar otras tareas, como el envío de correo electrónico. Debido a que tanto las computadoras cliente como las de servidor se consideran dispositivos inteligentes, el modelo cliente-servidor es completamente diferente del antiguo modelo de "mainframe", en el cual una computadora central centralizada realizó todas las tareas para sus terminales "tontas" asociadas. (Britannica IT, 2019)

3. Tecnologías

1. Tecnologías usadas para el desarrollo

1. NetBeans

NetBeans IDE es el IDE oficial para Java 8. Con sus editores, analizadores de código y convertidores, puede actualizar sus aplicaciones de manera rápida y sin problemas para usar nuevas construcciones de lenguaje Java 8, como lambdas, operaciones funcionales y referencias de métodos.

Se proporcionan analizadores de lotes y convertidores para buscar a través de múltiples aplicaciones al mismo tiempo, combinando patrones para la conversión a nuevas construcciones de lenguaje Java 8.

Con su Editor de Java en constante mejora, muchas funciones avanzadas y una amplia gama de herramientas, plantillas y muestras, NetBeans IDE establece el estándar para el desarrollo con tecnologías de vanguardia listas para usar.

NetBeans IDE ofrece diferentes vistas de sus datos, desde múltiples ventanas de proyectos hasta herramientas útiles para configurar sus aplicaciones y administrarlas de manera eficiente, lo que le permite profundizar en sus datos de manera rápida y sencilla, mientras le brinda herramientas de control de versiones a través de Subversion, Mercurial y la integración de Git fuera de la caja. Cuando nuevos desarrolladores se unen a su proyecto, pueden entender la estructura de su aplicación porque su código está bien organizado. (Netbeans, 2018)

2. HeidiSQL

HeidiSQL es un software gratuito, y tiene el objetivo de ser fácil de aprender. "Heidi" le permite ver y editar datos y estructuras desde computadoras que ejecutan uno de los sistemas de base de datos MariaDB, MySQL, Microsoft SQL o PostgreSQL. Inventado en 2002 por Ansgar, con un pico de desarrollo entre 2009 y 2013, HeidiSQL pertenece a las herramientas más populares para MariaDB y MySQL en todo el mundo. (HeidiSQL, 2018)

3. StarUML

StarUML es una herramienta de modelado de software de código abierto que admite UML (Unified Modeling Language). Se basa en la versión 1.4 de UML, proporciona once tipos diferentes de diagramas y acepta la notación UML 2.0. Apoya activamente el enfoque MDA (Model Driven Architecture) al admitir el concepto de perfil UML y permite generar código para múltiples idiomas.

Cuando inicia un nuevo proyecto, StarUML propone qué enfoque desea utilizar: 4 + 1 (Krutchen), Rational, componentes UML (del libro de Cheesman y Daniels), predeterminado o vacío. Dependiendo del enfoque, los perfiles y / o marcos pueden ser incluidos y cargados. Si no sigue un enfoque específico, se podría usar la opción "vacía". Aunque un proyecto se puede administrar como un archivo, puede ser conveniente dividirlo en muchas unidades y administrarlos por separado si muchos desarrolladores están trabajando juntos en él. StarUML hace una clara distinción conceptual entre modelos, vistas y diagramas.

Un modelo es un elemento que contiene información para un modelo de software. Una vista es una expresión visual de la información contenida en un modelo, y un diagrama es una colección de elementos de vista que representan los pensamientos de diseño específicos del usuario. (Martinig, 2019)

2. Tecnologías de ejecución

1. MySQL

MySQL es un sistema de gestión de bases de datos relacionales de código abierto respaldado por Oracle (RDBMS) basado en lenguaje de consulta estructurado (SQL). MySQL se ejecuta en prácticamente todas las plataformas, incluidos Linux, UNIX y Windows. Aunque se puede usar en una amplia gama de aplicaciones, la mayoría de las veces, MySQL se asocia con aplicaciones web y publicaciones en línea.

Originalmente concebido por la compañía sueca MySQL AB, MySQL fue adquirido por Sun Microsystems en 2008 y luego por Oracle cuando compró Sun en 2010. Los desarrolladores pueden usar MySQL bajo la Licencia Pública General de GNU (GPL), pero las empresas deben obtener una licencia comercial de Oracle.

Hoy en día, MySQL es el RDBMS detrás de muchos de los sitios web más importantes del mundo e innumerables aplicaciones web orientadas al consumidor y corporativas, como Facebook, Twitter y YouTube.

MySQL se basa en un modelo cliente-servidor. El núcleo de MySQL es el servidor MySQL, que maneja todas las instrucciones (o comandos) de la base de datos. El servidor MySQL está disponible como un programa separado para su uso en un entorno de red cliente-servidor y como una biblioteca que se puede incrustar (o vincular) en aplicaciones separadas.

MySQL funciona junto con varios programas de utilidad que admiten la administración de bases de datos MySQL. Los comandos se envían a MySQLServer a través del cliente MySQL, que se instala en una computadora. (Rouse, 2013)

2. Java

Java es un lenguaje de programación informática. Permite a los programadores escribir instrucciones de computadora utilizando comandos basados en inglés en lugar de tener

que escribir en códigos numéricos. Es conocido como un lenguaje de alto nivel porque los humanos pueden leerlo y escribirlo fácilmente.

Al igual que el inglés, Java tiene un conjunto de reglas que determinan cómo se escriben las instrucciones. Estas reglas se conocen como su sintaxis. Una vez que se ha escrito un programa, las instrucciones de alto nivel se traducen en códigos numéricos que las computadoras pueden entender y ejecutar.

A principios de los años 90, Java, que originalmente se llamaba Oak y luego Green, fue creado por un equipo dirigido por James Gosling para Sun Microsystems, una compañía que ahora es propiedad de Oracle.

Java fue diseñado originalmente para su uso en dispositivos móviles digitales, como teléfonos celulares. Sin embargo, cuando Java 1.0 se lanzó al público en 1996, su enfoque principal cambió a su uso en Internet, brindando interactividad a los usuarios al brindarles a los desarrolladores una forma de producir páginas web animadas.

Sin embargo, ha habido muchas actualizaciones desde la versión 1.0, como J2SE 1.3 en 2000, J2SE 5.0 en 2004, Java SE 8 en 2014 y Java SE 10 en 2018. A lo largo de los años, Java ha evolucionado como un lenguaje exitoso para su uso tanto dentro como fuera de Internet.

Java fue diseñado con algunos principios clave en mente:

Facilidad de uso: los fundamentos de Java provienen de un lenguaje de programación llamado C ++. Aunque C ++ es un lenguaje poderoso, tiene una sintaxis compleja e inadecuada para algunos de los requisitos de Java. Java desarrolló y mejoró las ideas de C ++ para proporcionar un lenguaje de programación que fuera potente y fácil de usar.

Confiabilidad: Java necesitaba reducir la probabilidad de errores fatales de los errores del programador. Con esto en mente, se introdujo la programación orientada a objetos. Cuando los datos y su manipulación se empaquetaron en un solo lugar, Java fue robusto.

Seguridad: como Java originalmente estaba dirigido a dispositivos móviles que intercambiarían datos a través de redes, se creó para incluir un alto nivel de seguridad. Java es probablemente el lenguaje de programación más seguro hasta la fecha.

Independencia de la plataforma: los programas deben funcionar independientemente de las máquinas en las que se ejecutan. Java fue escrito para ser un lenguaje portátil y multiplataforma que no se preocupa por el sistema operativo, el hardware o los dispositivos en los que se está ejecutando.

El equipo de Sun Microsystems logró combinar estos principios clave, y la popularidad de Java puede atribuirse a que es un lenguaje de programación robusto, seguro, fácil de usar y portátil. (Leahy, 2019)

3. Java Swing

Java Swing es un kit de herramientas de widget de interfaz gráfica de usuario (GUI) de peso ligero que incluye un amplio conjunto de widgets. Forma parte de Java Foundation

Classes (JFC) e incluye varios paquetes para desarrollar aplicaciones de escritorio enriquecidas en Java. Swing incluye controles incorporados como árboles, botones de imagen, paneles con pestañas, controles deslizantes, barras de herramientas, selectores de color, tablas y áreas de texto para mostrar el formato HTTP o texto enriquecido (RTF). Los componentes de Swing están escritos completamente en Java y, por lo tanto, son independientes de la plataforma.

Swing ofrece la personalización de la apariencia de cada componente en una aplicación sin realizar cambios significativos en el código de la aplicación. También incluye una característica de apariencia y tacto conectable, que le permite emular la apariencia de los componentes nativos sin dejar de tener la ventaja de la independencia de la plataforma. Esta característica particular facilita la escritura de aplicaciones en Swing y la distingue de otros programas nativos.

Swing se distribuyó como una biblioteca descargable y se ha incluido como parte de la edición estándar de Java 1.2. Originalmente, la biblioteca de gráficos para Java, desarrollada por Netscape Communication Corporation, se llamaba Internet Foundation Classes (IFC). La primera publicación de IFC fue el 16 de diciembre de 1996. La evolución de JFC se remonta a 1997, cuando Sun Microsystems y Netscape Communication Corporation tuvieron la idea de fusionar IFC con otras tecnologías. (Techopedia, 2018)

4. JFrame

Siempre que cree una interfaz gráfica de usuario con la funcionalidad Java Swing, necesitará un contenedor para su aplicación. En el caso de Swing, este contenedor se llama JFrame. Todas las aplicaciones GUI requieren un JFrame. De hecho, algunos applets incluso utilizan un JFrame.

No se puede construir una casa sin cimientos. Lo mismo ocurre en Java: sin un contenedor en el que colocar todos los demás elementos, no tendrá una aplicación GUI. En otras palabras, el JFrame se requiere como base o contenedor base para todos los demás componentes gráficos.

Las aplicaciones Java Swing se pueden ejecutar en cualquier sistema que admita Java. Estas aplicaciones son ligeras. Esto significa que no ocupan mucho espacio ni utilizan muchos recursos del sistema. (Gibbs, 2019)

5. Arduino

Arduino es una plataforma electrónica de código abierto basada en hardware y software fáciles de usar. Las placas Arduino pueden leer entradas (luz en un sensor, un dedo en un botón o un mensaje de Twitter) y convertirla en una salida: activar un motor, encender un LED y publicar algo en línea. Puede decirle a su tarjeta qué debe hacer

enviando un conjunto de instrucciones al microcontrolador en la tarjeta. Para hacerlo, utiliza el lenguaje de programación Arduino (basado en Wiring) y el software Arduino (IDE), basado en el procesamiento.

A lo largo de los años, Arduino ha sido el cerebro de miles de proyectos, desde objetos cotidianos hasta instrumentos científicos complejos. Una comunidad mundial de creadores (estudiantes, aficionados, artistas, programadores y profesionales) se ha reunido en torno a esta plataforma de código abierto, sus contribuciones han sumado una cantidad increíble de conocimiento accesible que puede ser de gran ayuda para los principiantes y expertos por igual. (Arduino, 2019)

6. Módulo SIM900

El SIM900 es una solución GSM / GPRS cuatribanda completa en un módulo SMT que puede integrarse en las aplicaciones del cliente. Con una interfaz estándar de la industria, el SIM900 ofrece un rendimiento GSM / GPRS 850/900/1800 / 1900MHz para voz, SMS, datos y fax en un formato pequeño y con bajo consumo de energía. (SimCom, 2018)

3. Objetivos del prototipo tecnológico

1. Objetivo General

Mejorar la calidad de servicio brindado a los clientes del Hospital del Día "Gabriel Andrade" del Cantón Ventanas, con el desarrollo de un sistema Informático de Facturación de citas y detalle de tratamientos médicos, y notificaciones a través de mensajería móvil SMS.

2. Objetivos Específicos

7. Diseñar una interfaz gráfica de usuario comprensible y entendible para el manejo de los servicios del hospital.

8. Automatizar los procesos de los servicios de facturación que se realizan en el hospital.

9. Enviar notificaciones a los pacientes del hospital a través de mensajería móvil (SMS).

1. Diseño del prototipo tecnológico

Según el rol de cada usuario, este será el encargado de notificar directamente a cada paciente, los valores a pagar de los servicios que se ofrecen el hospital. Para ello se procedió a automatizar los procesos de tratamiento y recepción de citas que se realizan en el hospital para luego de ello proceder a enviar las notificaciones por medio de mensajes de texto a los pacientes.

Se utilizará la tecnología adecuada para que el uso de la aplicación sea fácil y amigable para que supla las necesidades que tiene cada usuario del sistema. Toda la información que manipule el sistema será almacenada en la base de datos, manteniendo como datos históricos, a la cual se podría acceder desde cualquier momento que se desee.

1. Fase de análisis

Se procedió a realizar una entrevista el director del hospital, quien es el encargado de gestionar los procesos administrativos de la institución. Durante la entrevista se pudo determinar los inconvenientes que están presentes en la institución médica.

Durante este proceso se logó obtener los requerimientos necesarios para el desarrollo del sistema, basados en las necesidades funcionales que presenta la institución médica, orientadas a la mejora de la calidad de servicios que se ofrecen en el hospital.

2. Fase de diseño

El sistema desarrollado ha automatizado los procesos administrativos como gestión de pacientes, doctores, citas médicas y tratamientos, así como la generación de reportes. El sistema también consta con un módulo para la administración de usuarios, el cual permite además de la creación de usuarios, también permite activar o desactivar funciones del sistema a determinados usuarios. El sistema integra un módulo de envío de notificaciones a través de mensajería de texto (SMS).

Funcionalidad del Sistema		
Ingreso al sistema	Ingresos de datos (crear , actualizar, activar, inactivar usuarios, doctores, pacientes)	Enviar notificaciones Generar reportes

Gráfico 1 .- Funcionalidad de la Aplicación: Administrador.

Funcionalidad del Sistema		
Ingreso al sistema	Ingresos de datos (Realizará procesos según las funciones asignadas por el administrador)	Según las funciones asignadas del administrador.

Gráfico 2.- Funcionalidad de la Aplicación: Otros usuarios.

Diagrama de clases

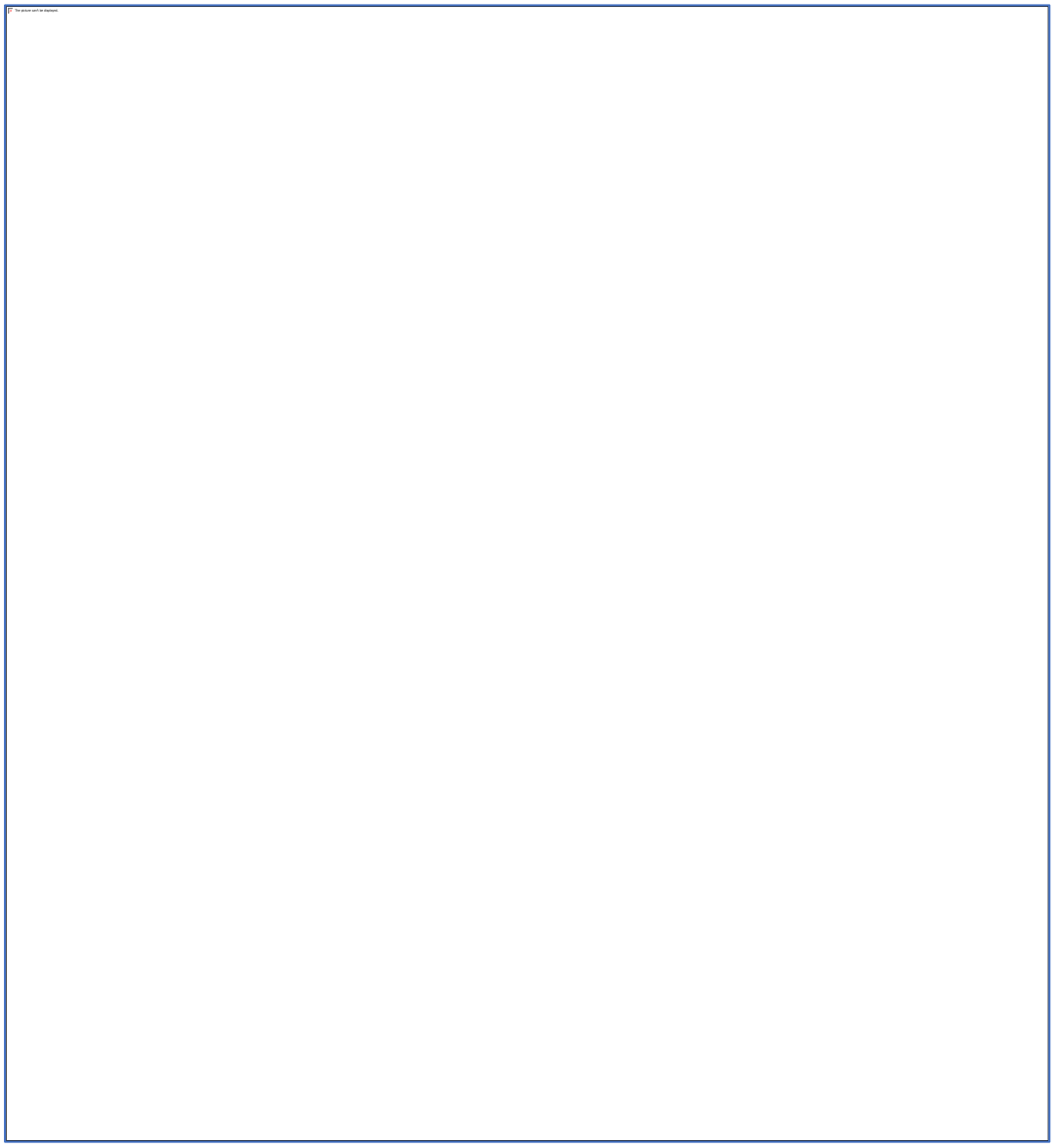

Gráfico 3.- Diagrama de Clases

Diagrama de actividades

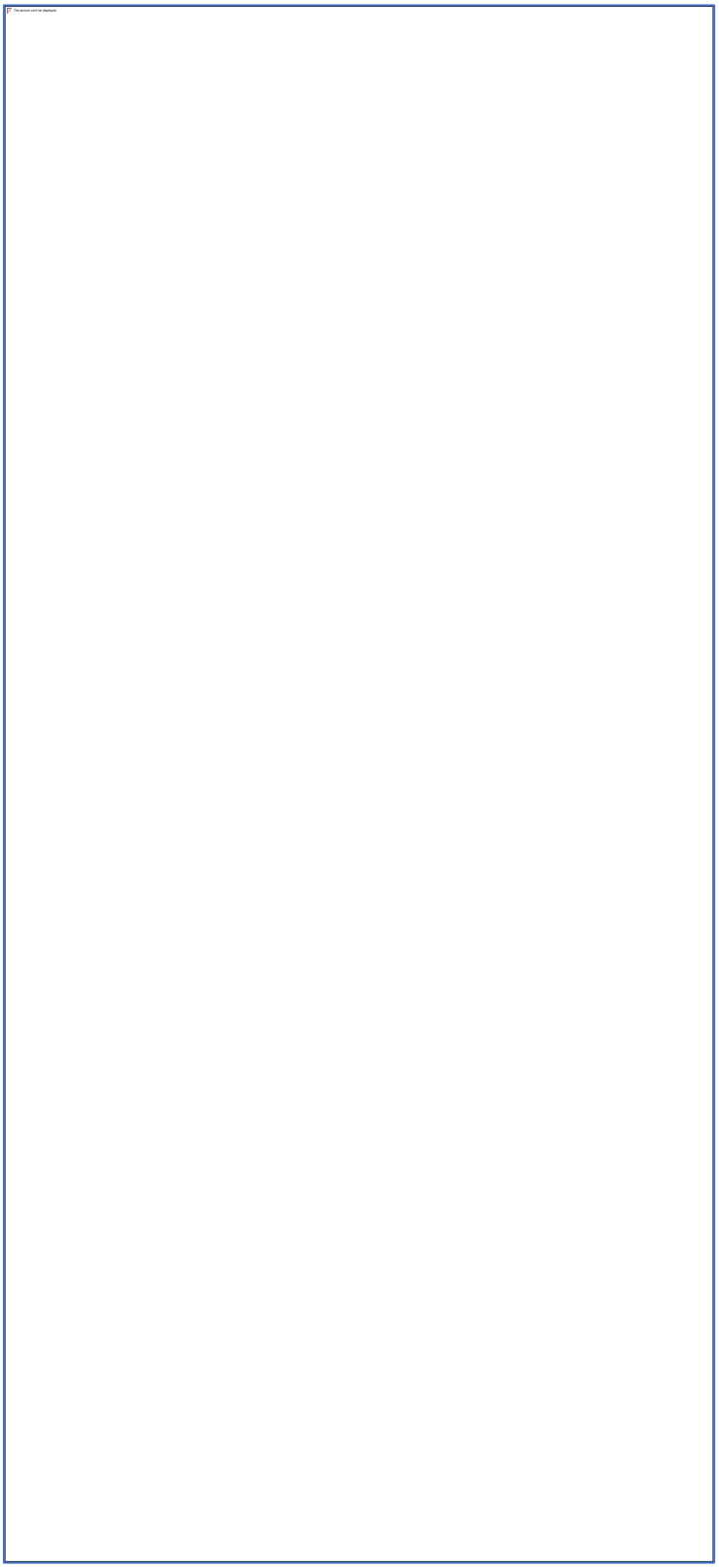

Gráfico 4.- Diagrama de actividades

Casos de uso

Gráfico 5.- Diagrama caso de uso ingreso al sistema

Gráfico 6.- Diagrama caso de uso registro de nuevo servicio.

Gráfico 7.- Diagrama caso de uso registro de nuevo paciente.

Gráfico 8.- Diagrama caso de uso actualización de paciente.

Gráfico 9.- Diagrama caso de uso registro de servicios

Gráfico 10.- Diagrama caso de uso registro de médico

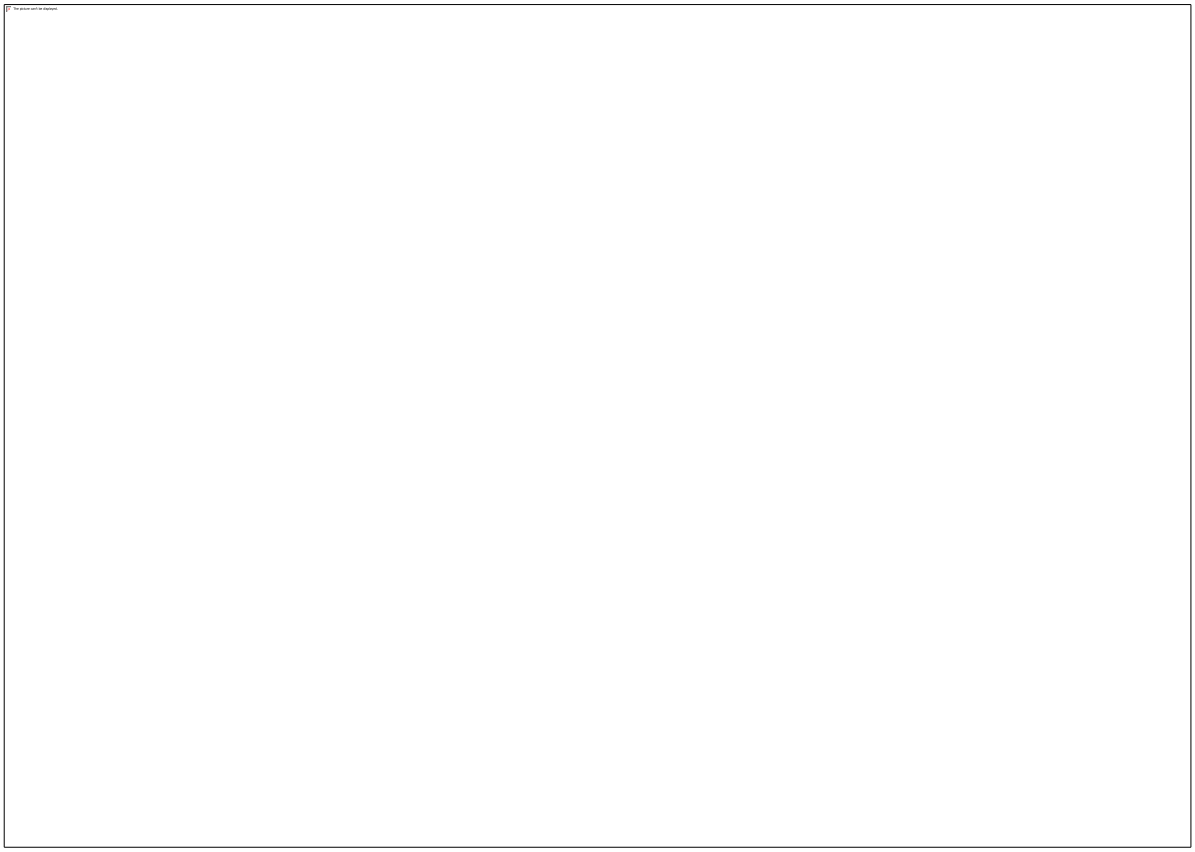

Gráfico 11.- Diagrama caso de uso consulta de historial

Modelo conceptual

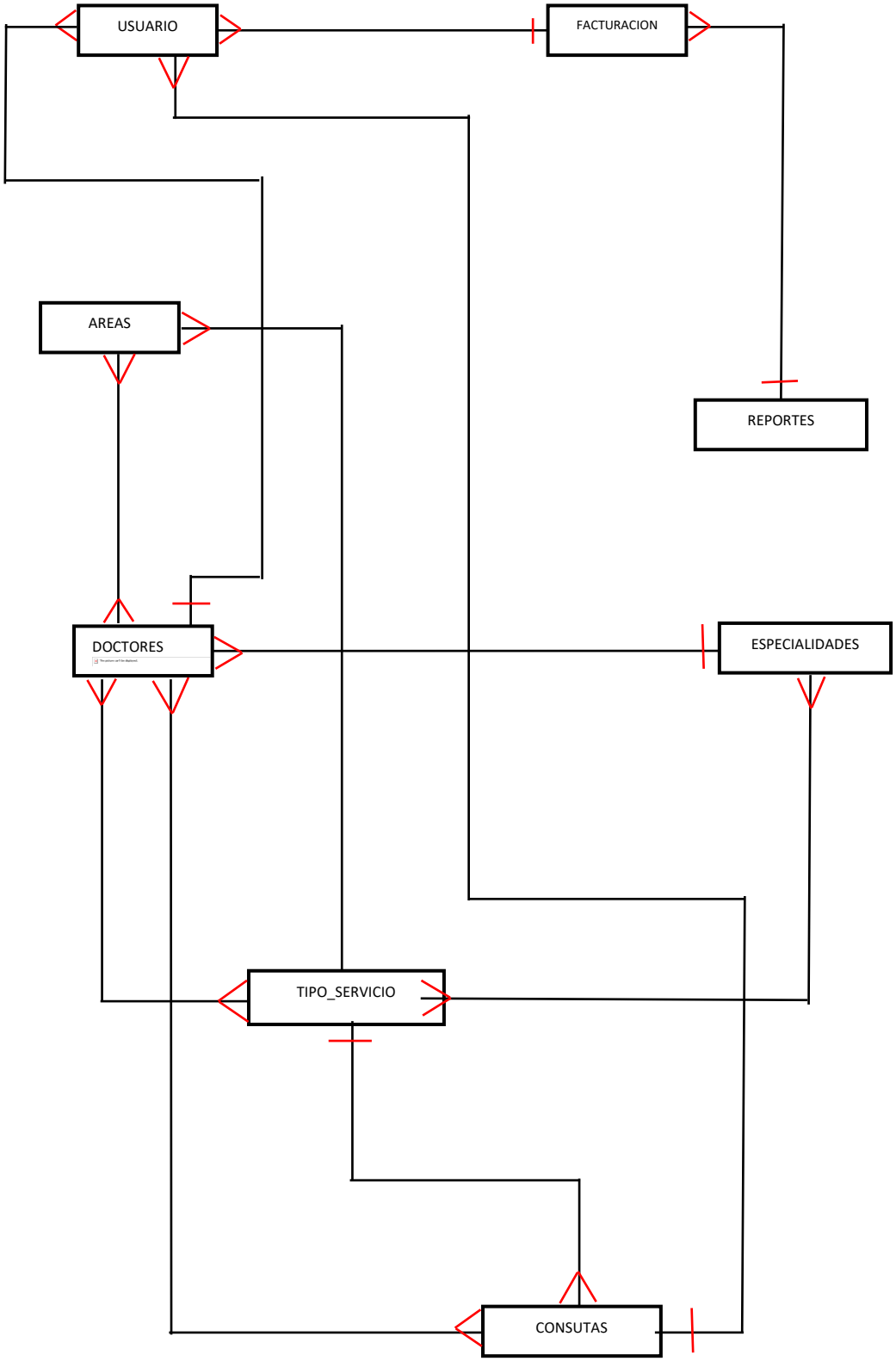

Grafico 1.- Diagrama Entidad Relación

Modelo Lógico

Grafico 2.- Diagrama Entidad Relación

Diccionario de datos

Columna	Tipo	Nulo
BT_ACT_DOC	int(11)	No
BT_ELI_DOC	int(11)	No
BT_ACT_ESP	int(11)	No
BT_ELI_ESP	int(11)	No
BT_ACT_PAC	int(11)	No
BT_ELI_PAC	int(11)	No
BT_ACT_TPSERV	int(11)	No
BT_ELI_TPSERV	int(11)	No
iduser	int(11)	No

Tabla 1.- **tbbotones**

Columna	Tipo	Nulo
cedd *(Primaria)*	char(10)	No
apellidos	varchar(30)	No
nombres	varchar(30)	No
telef	char(10)	Sí
cel	char(10)	No
idesp	int(11)	No
nclgdo	int(11)	No
freg	char(10)	No
iduser	int(11)	No

Tabla 2.- **tbdoctores**

Columna	Tipo	Nulo
idesp *(Primaria)*	int(11)	No

nesp	varchar(100)	No

Tabla 3.- **tbespecialidades**

Columna	**Tipo**	**Nulo**
idfact *(Primaria)*	char(10)	No
ced	char(10)	No
idesp	int(11)	No
idserv	int(11)	No
detallefact	varchar(100)	No
punit	double(7,2)	No
subtot	double(7,2)	No
dscto	double(7,2)	No
iva	double(7,2)	No
neto	double(7,2)	No
tpago	char(10)	No
fechaa	char(10)	No
iduser	int(11)	No

Tabla 4.- **tbfacturacion**

Columna	**Tipo**	**Nulo**
idh *(Primaria)*	int(11)	No
ced	char(10)	No
detallepadece	char(250)	Sí
idserv	int(11)	Sí
nmedicatrata	char(250)	Sí
dosis	char(50)	Sí

vadministra	varchar(50)	Sí
frecuencia	char(50)	Sí
duracion	char(50)	Sí
tipmedica	varchar(30)	Sí
medi_medi	varchar(20)	Sí
tfabri	varchar(20)	Sí
freg	char(10)	Sí
iduser	int(11)	No

Tabla 5.- **tbhimedico**

Columna	**Tipo**	**Nulo**
ced *(Primaria)*	char(10)	No
nom	varchar(50)	No
edad	int(11)	No
sexo	char(1)	No
telf	char(10)	No
freg	char(10)	No
iduser	int(11)	No

Tabla 6.- **tbpacientes**

Columna	**Tipo**	**Nulo**
idserv *(Primaria)*	int(11)	No
nserv	varchar(200)	Sí
pserv	double(4,2)	Sí
giva	char(2)	Sí
idesp	int(11)	Sí

fregis	char(10)	Sí

Tabla 7.- **tbservicios**

Columna	**Tipo**	**Nulo**
SUB_MOD_AREAS	int(11)	No
SUB_MOD_USUARIOS	int(11)	No
iduser	int(11)	No

Tabla 8.- **tbsubmodulos**

Columna	**Tipo**	**Nulo**
iduser *(Primaria)*	int(11)	No
nuser	varchar(20)	No
pass	char(50)	No
estdo	int(11)	No
MOD_FACTURACION	int(11)	No
MOD_TPSERVICIOS	int(11)	No
MOD_DOCTORES	int(11)	No
MOD_AREAS	int(11)	No
MOD_PACIENTES	int(11)	No
MOD_SEGURIDAD	int(11)	No
freg	char(10)	No

Tabla 9.- **tbusuarios**

Modelo físico

Script de la base de datos

```
  DROP DATABASE citas

CREATE DATABASE citas

USE citas

DROP TABLE IF EXISTS tbusuarios;

CREATE TABLE tbusuarios(

iduser INT NOT NULL,

nuser VARCHAR(20) NOT NULL,

pass CHAR(50) NOT NULL,

estdo INT DEFAULT 0,

facturacion INT DEFAULT 0,

servicios INT DEFAULT 0,

doctores INT DEFAULT 0,

areas    INT DEFAULT 0,

pacientes INT DEFAULT 0,

seguridad INT DEFAULT 0,

freg CHAR(10) DEFAULT 0,

CONSTRAINT pkusr PRIMARY KEY (iduser)

)ENGINE=INNODB DEFAULT CHARSET=utf8;
```

```
CREATE VIEW mostrar_usuarios

AS

SELECT estdo,nuser

FROM tbusuarios

SELECT * FROM tbpermisos

DROP TABLE tbpacientes

CREATE TABLE tbpacientes(

ced CHAR(10) NOT NULL,

nom VARCHAR(50) NOT NULL,

edad INT NOT NULL,

sexo CHAR(1)NOT NULL,

telf CHAR(10) NOT NULL,

freg CHAR(10) NOT NULL,

iduser INT NOT NULL,

CONSTRAINT PK_CED PRIMARY KEY (ced)

)ENGINE=INNODB DEFAULT CHARSET=utf8;

CREATE VIEW vw_MostrarPacientes

AS

SELECT * FROM tbpacientes
```

```
DROP TABLE tbtitulos

DROP TABLE citas

DROP TABLE tbareas

DROP TABLE tbespecialidades

CREATE TABLE tbespecialidades(

idesp INT NOT NULL,

nesp VARCHAR (100) NOT NULL,

CONSTRAINT PK_IDESP PRIMARY KEY (idesp)

)ENGINE=INNODB DEFAULT CHARSET=utf8;

CREATE VIEW vw_MostrarEspecialidades

AS

SELECT * FROM tbespecialidades

DROP TABLE tbdoctores

CREATE TABLE tbdoctores(

ci CHAR(10) NOT NULL,

apellidos VARCHAR(30) NOT NULL,

nombres VARCHAR(30) NOT NULL,

telef CHAR(10) ,

cel CHAR(10) NOT NULL,
```

```
idesp INT NOT NULL,

nclgdo INT NOT NULL,

freg CHAR(10) NOT NULL,

iduser INT NOT NULL,

CONSTRAINT PK_CEDD PRIMARY KEY (ci)

)ENGINE=INNODB DEFAULT CHARSET=utf8;

CREATE VIEW vw_MostrarDoctores

AS

SELECT ci,apellidos,nombres,tbespecialidades.`nesp`,nclgdo

FROM tbdoctores

INNER JOIN tbespecialidades ON tbdoctores.idesp=tbespecialidades.`idesp`;

CREATE VIEW vw_MostrarDoctores

AS

SELECT tbdoctores.`ci`,tbdoctores.`apellidos`,tbdoctores.`nombres`,

tbespecialidades.`nesp`,tbdoctores.`nclgdo`

FROM tbdoctores

INNER JOIN tbespecialidades ON tbdoctores.`idesp`=tbespecialidades.`idesp`;

ALTER TABLE tbdoctores

ADD CONSTRAINT FK_IDESPECILIDAD_DOCTOR
```

```
FOREIGN KEY (idesp)

REFERENCES tbespecialidades(idesp)ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE tbdoctores

ADD CONSTRAINT FK_IDUSER_TBUSUARIOS

FOREIGN KEY (iduser)

REFERENCES tbusuarios(iduser)ON DELETE CASCADE ON UPDATE CASCADE;

create table tbafilia_labor(

idtbf int auto_increment primary key,

idesp INT NOT NULL,

cedd CHAR(10));

DROP TABLE tbfacturacion

CREATE TABLE tbfacturacion(

idfact CHAR (20) NOT NULL,

ced CHAR(10) NOT NULL,

idesp INT NOT NULL,

idserv INT NOT NULL,

detallefact VARCHAR(100) NOT NULL,

punit DOUBLE (7,2)NOT NULL,

subtot DOUBLE (7,2)NOT NULL,
```

```
dscto DOUBLE (7,2)NOT NULL,

iva DOUBLE (7,2)NOT NULL,

neto DOUBLE (7,2)NOT NULL,

tpago CHAR(10) NOT NULL,

fechaa CHAR(10) NOT NULL,

iduser INT NOT NULL,

CONSTRAINT PK_IDFACT PRIMARY KEY (idfact)

)ENGINE=INNODB DEFAULT CHARSET=utf8;

alter table tbfacturacion add tipodoc char(20);

alter table tbfacturacion add  medida char(20);

alter table tbfacturacion add  marca char(20);

alter table tbfacturacion add dosis char(100);

alter table tbfacturacion add frecuencia char(100);

alter table tbfacturacion add duracion char(100);

alter table tbfacturacion add va char(20);

alter table tbfacturacion add  tipomed char(20);

alter table tbfacturacion add observacion char(200);

SELECT * FROM tbfacturacion

SELECT MAX(idfact) AS idfact FROM tbfacturacion
```

```
DROP TABLE tbhimedico

CREATE TABLE tbhimedico(

ced CHAR(10) NOT NULL,

detallepadece CHAR(250) ,

idserv INT ,

nmedicatrata CHAR(250) ,

dosis CHAR(50),

frecuencia CHAR(50) ,

duracion CHAR(50) ,

vadministra VARCHAR(50),

tipmedica VARCHAR (30),

medi_medi VARCHAR(20),

tfabri VARCHAR(20),

freg CHAR(10),

iduser INT NOT NULL

)ENGINE=INNODB DEFAULT CHARSET=utf8;

ALTER TABLE tbhimedico

ADD CONSTRAINT FK_IDPACIENTE_HISTORIAL

FOREIGN KEY (ced)
```

```
REFERENCES tbpacientes (ced)ON DELETE CASCADE ON UPDATE CASCADE;

SELECT
tbhimedico.`ced`,tbpacientes.`nom`,tbhimedico.`nmedicatrata`,tbhimedico.`dosis`,tbhimedi
co.`vadministra`,
tbhimedico.`frecuencia`,tbhimedico.`duracion`,tbhimedico.`tipmedica`,
tbhimedico.`medi_medi`,tbhimedico.`tfabri`,tbhimedico.`detallepadece`
FROM tbhimedico
INNER JOIN tbpacientes ON tbhimedico.`ced`=tbpacientes.`ced` WHERE
tbpacientes.`ced`='1203482361'
 DROP TABLE tbservicios
CREATE TABLE tbservicios(
idserv INT NOT NULL,
nserv VARCHAR(200) NOT NULL,
pserv DOUBLE (7,2) NOT NULL,
giva CHAR(2) NOT NULL,
idesp INT NOT NULL,
fregis CHAR(10) NOT NULL,
CONSTRAINT PK_IDSERV PRIMARY KEY (idserv)
)ENGINE=INNODB DEFAULT CHARSET=utf8;
```

```
CREATE VIEW vw_MostrarTPServicios

AS

SELECT

tbservicios.`idserv`,tbservicios.`nserv`,tbservicios.`pserv`,tbservicios.`giva`,tbespecialidades.`

nesp`,tbservicios.`idesp`

FROM tbservicios

INNER JOIN tbespecialidades ON tbservicios.`idesp`=tbespecialidades.`idesp`;

SELECT * FROM tbservicios

SELECT b.BT_ACT_DOC,u.iduser

FROM tbusuarios u INNER JOIN tbbotones b ON u.iduser=b.iduser

SELECT tbpacientes.ced,tbpacientes.nom,tbservicios.`nserv`,tbpacientes.`freg`

FROM

tbpacientes tbpacientes

INNER JOIN tbhimedico  ON tbhimedico.ced=tbpacientes.ced

INNER JOIN tbservicios  ON tbhimedico.idserv=tbservicios.idserv WHERE

tbpacientes.freg='31/01/2017'

SELECT

tbpac.ced AS tbpacientes_ced,

tbpac.nom AS tbpacientes_nom,
```

```
tbserv.nserv AS tbservicios_nserv,

tbhi.freg AS tbhi_freg

FROM tbpacientes tbpac

INNER JOIN tbhimedico tbhi ON tbhi.ced=tbpac.ced

INNER JOIN tbservicios tbserv ON tbhi.idserv=tbserv.idserv WHERE tbhi.freg='01/02/2017'

SELECT

f.idfact AS tbfacturacion_idfact,

p.ced AS tbpacientes_ced,

p.nom AS tbpacientes_nom,

p.telf AS tbpacientes_telf,

s.nserv AS tbservicios_nserv,

f.detallefact AS tbfacturacion_detallefact,

f.punit AS tbfacturacion_punit,

f.subtot AS tbfacturacion_subtot,

f.dscto AS tbfacturacion_dscto,

f.iva AS tbfacturacion_iva,

f.neto AS tbfacturacion_neto

FROM

tbfacturacion f
```

```
INNER JOIN tbpacientes p ON f.ced=p.ced

INNER JOIN tbservicios s ON f.idserv=s.idserv WHERE p.ced='1202654146'

SELECT p.ced,p.edad,p.sexo,p.telf,p.freg,p.nom,s.nserv,h.nmedicatrata

FROM tbpacientes p

INNER JOIN tbhimedico h ON h.ced=p.ced

INNER JOIN tbservicios s ON h.idserv=s.idserv WHERE nom=nomb

SELECT tbpacientes.ced,tbpacientes.nom ,tbpacientes.edad ,tbpacientes.sexo
,tbhimedico.detallepadece ,tbservicios.nserv ,

tbhimedico.nmedicatrata ,tbhimedico.dosis ,tbhimedico.vadministra ,tbhimedico.frecuencia
,tbhimedico.duracion ,tbhimedico.tipmedica ,

tbhimedico.medi_medi ,tbhimedico.tfabri,tbhimedico.freg ,tbdoctores.apellidos
,tbdoctores.nombres ,tbdoctores.nclgdo,tbespecialidades.nesp

FROM tbpacientes

INNER JOIN tbhimedico  ON tbhimedico.ced=tbpacientes.ced

INNER JOIN tbservicios  ON tbhimedico.idserv=tbservicios.idserv

INNER JOIN tbespecialidades  ON tbservicios.idesp=tbespecialidades.idesp

INNER JOIN tbdoctores  ON tbdoctores.idesp=tbespecialidades.idesp WHERE
tbpacientes.ced=$P{ced}
```

```
SELECT
tbhimedico.ced,tbhimedico.detallepadece,tbservicios.nserv,tbhimedico.`nmedicatrata`,tbhi
medico.`dosis`,
tbhimedico.`frecuencia`,tbhimedico.`duracion`,tbhimedico.`vadministra`,tbhimedico.`tipme
dica`,tbhimedico.`medi_medi`,tbhimedico.`tfabri`
FROM tbhimedico
INNER JOIN tbservicios ON tbhimedico.`idserv`=tbservicios.`idserv`
tbpacientes.edad AS tbpacientes_edad,
tbpacientes.sexo AS tbpacientes_sexo,
medi.detallepadece AS tbhimedico_detallepadece,
svr.nserv AS tbservicio_nserv,
medi.nmedicatrata AS tbhimedico_nmedicatrata,
medi.dosis AS medi_dosis,
medi.vadministra AS medi_vadministra,
medi.frecuencia AS medi_frecuencia,
medi.duracion AS medi_duracion,
medi.tipmedica AS medi_tipmedica,
medi.medi_medi AS medi_medi_medi,
medi.tfabri AS medi_rfabri,
```

```
medi.freg AS medi_freg

FROM tbpacientes tbpacientes

INNER JOIN tbhimedico medi ON medi.ced=tbpacientes.ced

INNER JOIN tbservicios svr ON medi.idserv=svr.idserv

SELECT

tbpacientes.ced AS tbpacientes_ced,

tbpacientes.nom AS tbpacientes_nom,

tbpacientes.edad AS tbpacientes_edad,

tbpacientes.sexo AS tbpacientes_sexo,

medi.detallepadece AS tbhimedico_detallepadece,

es.nesp AS tbespecialidades_nesp,

svr.nserv AS tbservicio_nserv,

medi.nmedicatrata AS tbhimedico_nmedicatrata,

medi.dosis AS medi_dosis,

medi.vadministra AS medi_vadministra,

medi.frecuencia AS medi_frecuencia,

medi.duracion AS medi_duracion,

medi.tipmedica AS medi_tipmedica,

medi.medi_medi AS medi_medi_medi,
```

```
medi.tfabri AS medi_rfabri,

medi.freg AS medi_freg

FROM tbpacientes tbpacientes

INNER JOIN tbhimedico medi ON medi.ced=tbpacientes.ced

INNER JOIN tbespecialidades es ON medi.idesp=es.idesp

INNER JOIN tbservicios svr ON medi.idserv=svr.idserv

WHERE tbpacientes.ced=$P{ced}
```

3. Fase de desarrollo

Las herramientas de desarrollo que se utilizan que se utilizan durante esta fase son las siguientes:

Netbeans IDE (Entorno de Desarrollo Integrado) un poderoso gestor de código open source, en cual se puede desarrollar aplicaciones escritas en JAVA. También soporta números lenguajes de programación haciéndolo una herramienta muy utilizada por los desarrolladores.

Para su funcionamiento se requiere el JDK (Kit de Desarrollo de JAVA), pieza fundamental para el funcionamiento del presente prototipo en el lado del servidor. Este Kit provee al desarrollador una variedad de herramientas para crear aplicaciones tanto web como de escritorio.

MySQL se utilizó la gestión de la base de datos del prototipo. Esta herramienta provee administración visual para el Sistema Manejador de Bases de Datos PostgreSQL.

Arduino IDE es el entorno para programar las placas de código abierto de Arduino. La placa de Arduino para realizar las correspondientes pruebas de envío de mensajes de texto a teléfonos móviles fue Arduino UNO junto con el módulo GPRS/SIM900.

2. Ejecución del prototipo tecnológico

Al ingresar al sistema los usuarios visualizarán el formulario de inicio de sesión del sistema.

Ilustración 1 Pantalla de Inicio de Sesión

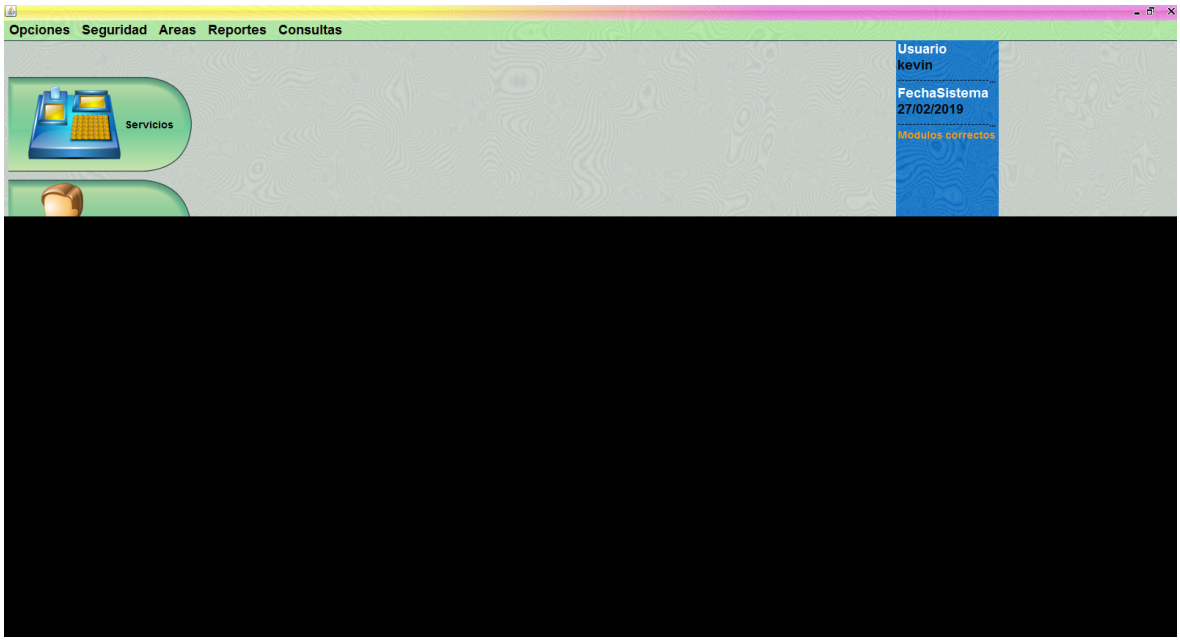

Ilustración 2 Pantalla de inicio del administrador

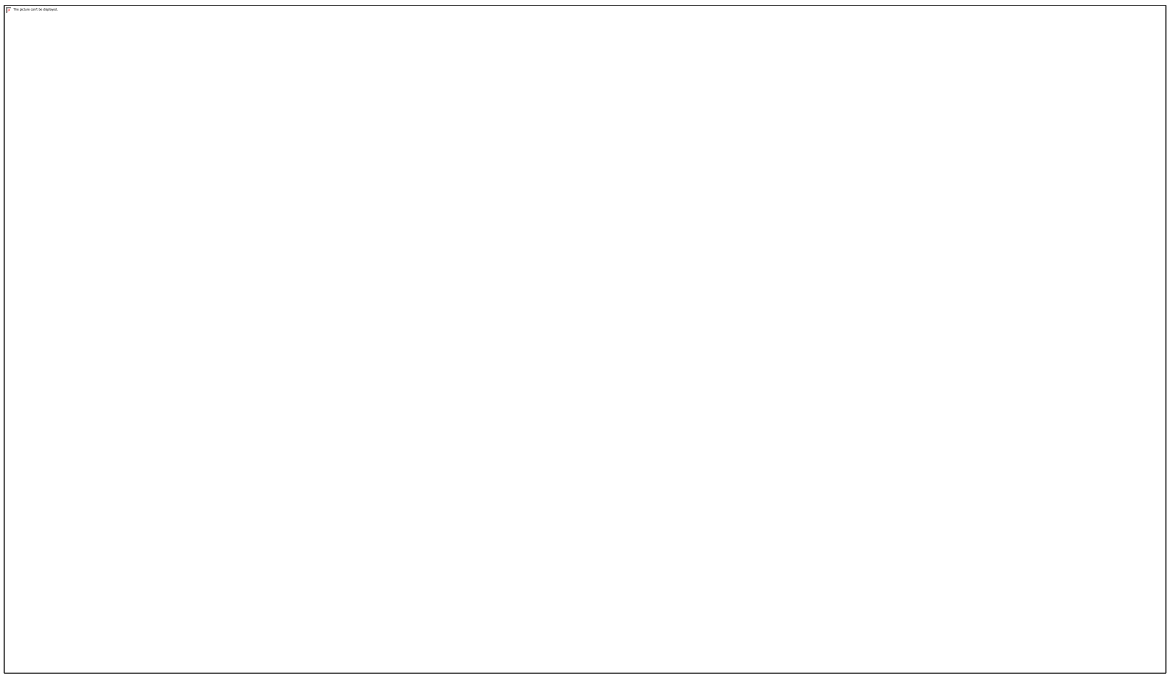

Ilustración 3 Pantalla de inicio de usuario con funciones específicas.

Ilustración 4 Pantalla gestión de usuarios y permisos.

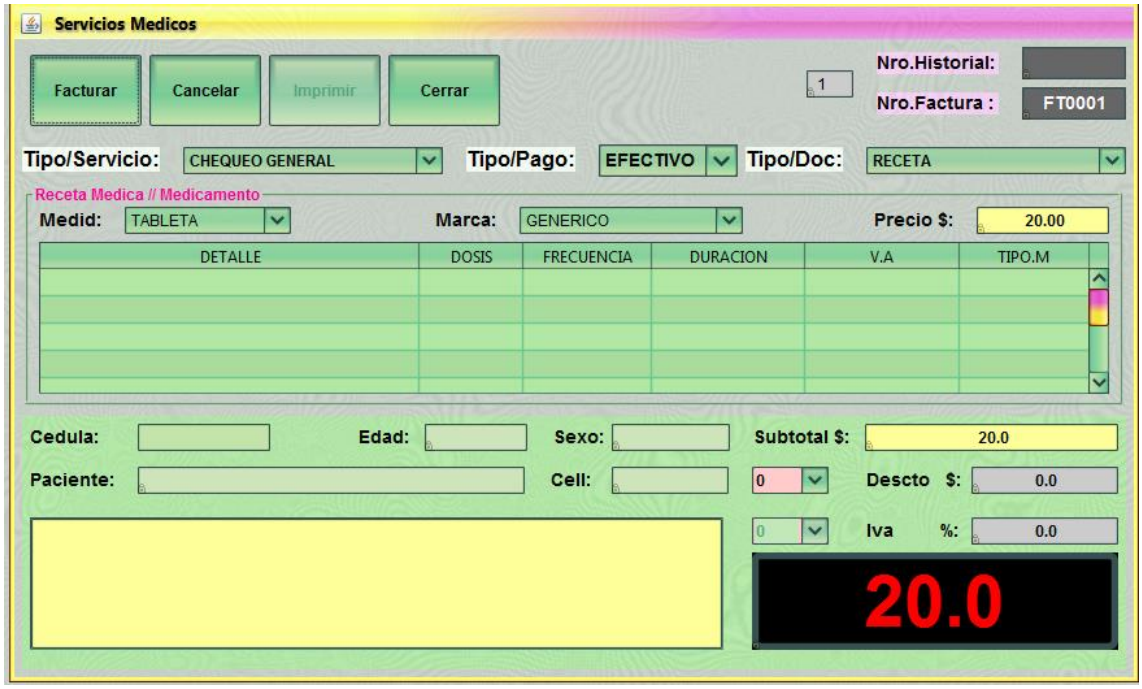

Ilustración 5 Pantalla gestión de servicios y facturación.

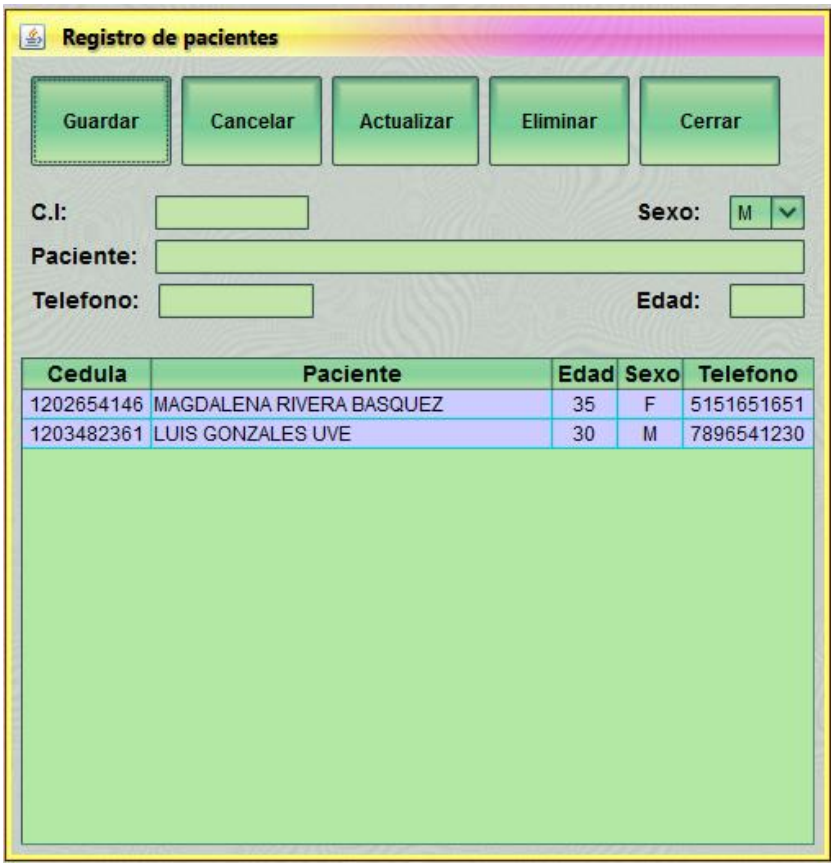

Ilustración 6 Pantalla gestión de pacientes.

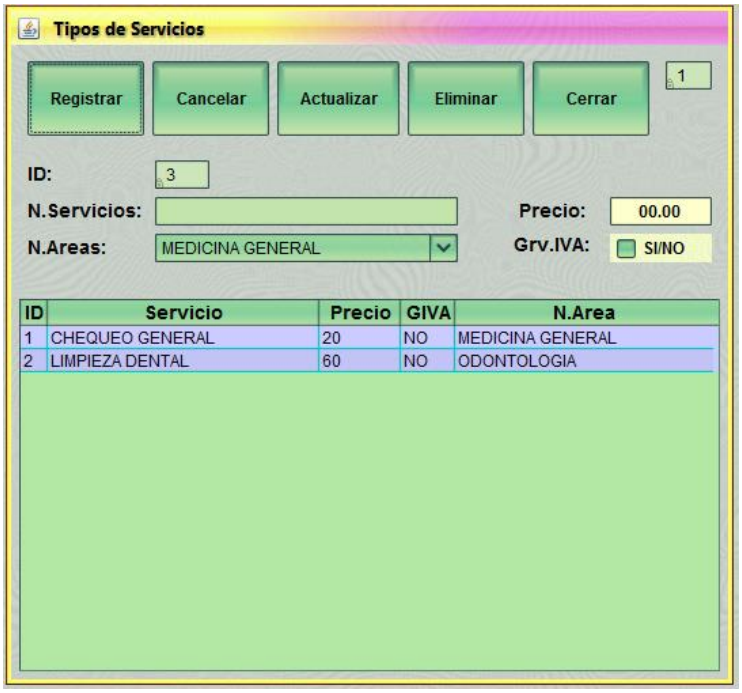

Ilustración 7 Pantalla gestión de servicios.

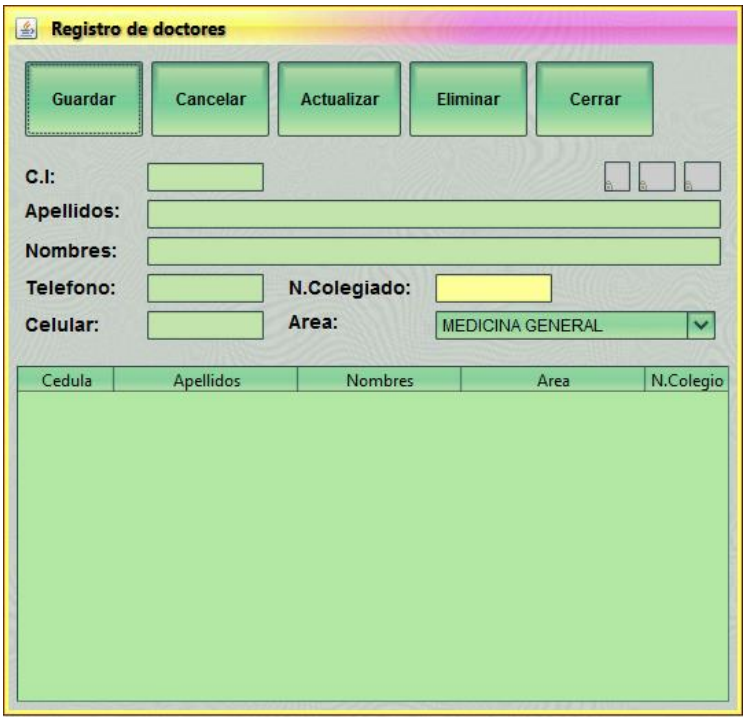

Ilustración 8 Pantalla gestión de doctores.

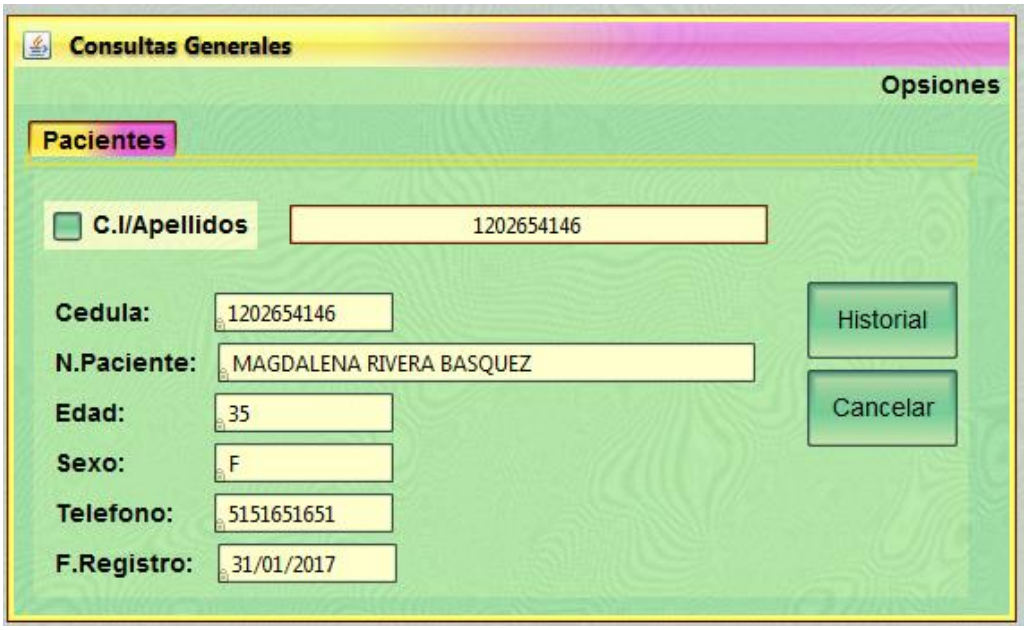

Ilustración 9 Pantalla consulta de historial.

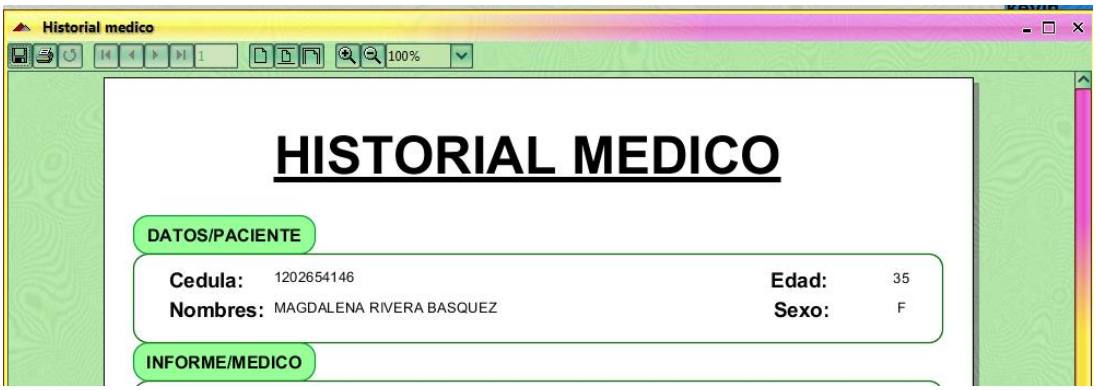

Ilustración 10 Pantalla de reportes.

CAPÍTULO III

10. Evaluación del prototipo

1. Plan de evaluación.

Cronograma de Actividades del desarrollo del Prototipo	
Fecha	**Actividad**
17/10/2018	Entrevista con el Director del hospital.
18/10/2018	Recolección de los requerimientos necesarios para el desarrollo del sistema.
20/10/2018	Instalación y configuración de las herramientas tecnológicas necesarias para el desarrollo del prototipo.
21/10/2018	Elaboración de la Base de Datos, utilizando los requerimientos recolectados en el hospital.
26/10/2018	Análisis de requerimientos funcionales y no funcionales mediante la construcción y diseños de diagramas UML.
28/10/2018	Implementación del modelo lógico al físico de la base de datos en el gestor de PostgreSQL.
30/10/2018	Creación de las relaciones entre tablas. Inserción de datos a tablas estáticas.
03/11/2018	Creación de procedimientos almacenados y funciones en PostgreSQL
05/11/2018	Comprobar que la Base de Datos no tenga problema realizando pruebas básicas.
15/11/2018	Desarrollo del Prototipo. Prueba de conexión entre JAVA y la Base de Datos con el Sistema.
16/11/2018	Desarrollo del Prototipo. Elaboración formulario de Inicio de Sesión.
20/11/2018	Desarrollo del Prototipo. Elaboración del Menú del Sistema.

25/11/2018	Desarrollo del Prototipo. Diseño de las páginas para ver los formularios de registro, ingresar, editar, y buscar datos de los estudiantes, padres de familia, aulas y docentes.
30/11/2018	Desarrollo del Prototipo. Conexión entre java y Arduino.
01/12/2018	Desarrollo del Prototipo. Elaboración de validaciones y verificación de datos
15/12/2018	Pruebas y testeo.
18/12/2018	Prueba de Caja blanca.

1. Resultados de la evaluación.

El sistema informático desarrollado y testeado en el Hospital del Día Gabriel Andrade; del Cantón Ventanas. Se realizaron pruebas en tiempo real y tanto los usuarios que manipulan el sistema con su respectivo rol y los clientes se mostraron muy satisfechos durante la realización de las pruebas. Se comprobó que el sistema cumple con los factores de eficacia, validez se reconoció que se cumplieron y se superaron las expectativas de los requerimientos planteados sobre él; la construcción del prototipo que se esperaba.

Se enviaron mensajes de texto del estado de facturación y valores a pagar, cumpliendo a cabalidad los el objetivo de mejorar calidad de servicio brindada por parte de la institución y sus pacientes.

1. Prueba de caja blanca

La prueba de caja blanca es un método de prueba de software que prueba las estructuras internas o el funcionamiento de una aplicación, a diferencia de su funcionalidad En las pruebas de caja blanca se utiliza una perspectiva interna del sistema, así como las habilidades de programación, para diseñar casos de prueba.

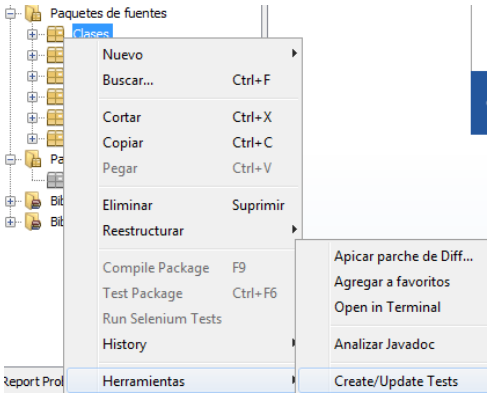

Ilustración 11 Crear clases de pruebas

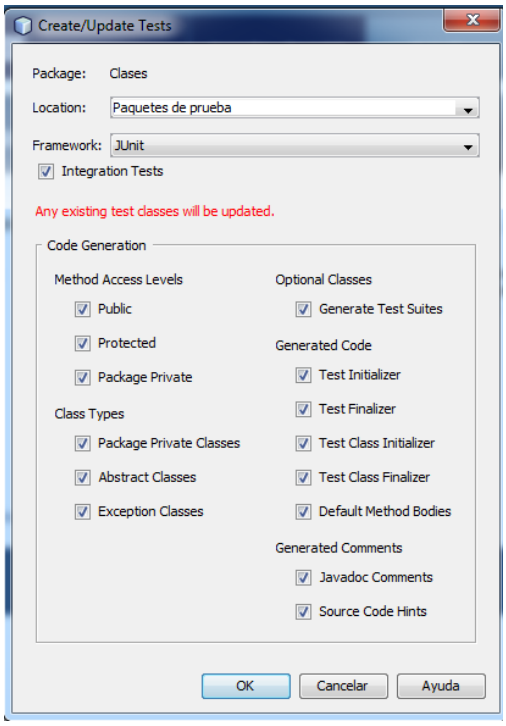

Ilustración 12 Selección de complemento de testeo

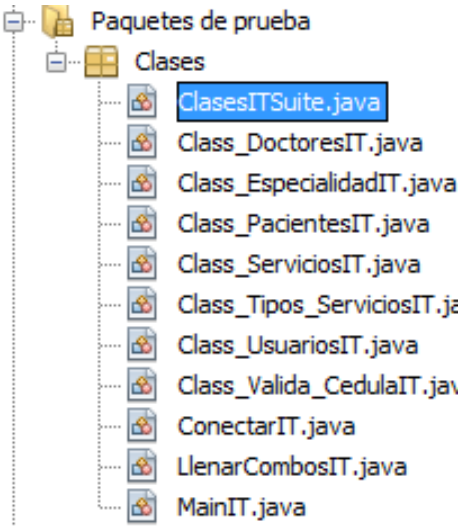

Ilustración 13 Clases generadas

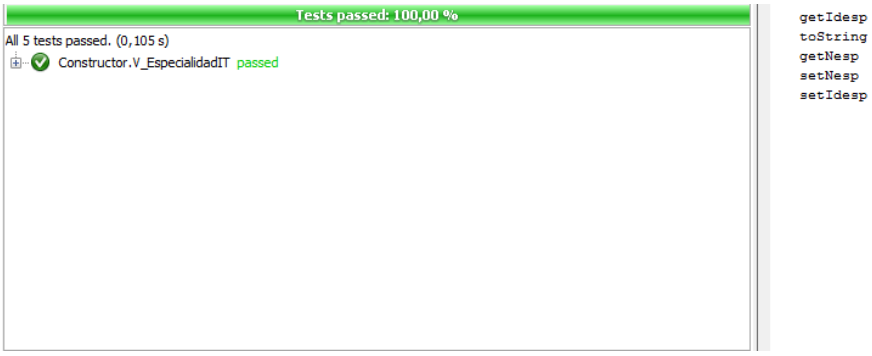

Ilustración 14 Resultado del Testeo

BIBLIOGRAFÍA

Arduino. (2019). *What is Arduino?* Obtenido de https://www.arduino.cc/en/Guide/Introduction

BALLESTEROS JARAIZ, A., Alfredo, B. F., Daniel, D. F., Víctor, F. P., María, G. P., Marcos, R. G., . . . María, S. C. (2018). *SOCIEDAD DIGITAL, TECNOLOGÍA Y EDUCACIÓN.* Editorial UNED.

Britannica IT. (2019). Obtenido de Client-server architecture: https://www.britannica.com/technology/client-server-architecture

Gibbs, M. (2019). *What is a JFrame Class in Java? - Uses & Examples*. Obtenido de https://study.com/academy/lesson/what-is-a-jframe-class-in-java-uses-examples.html

HeidiSQL. (2018). *What is HeidiSQL?* Obtenido de https://www.heidisql.com/

Leahy, P. (2019). *What Is Java?* . Obtenido de https://www.thoughtco.com/what-is-java-2034117

Martinig, F. (2019). *StarUML - Open Source UML Tool*. Obtenido de http://www.methodsandtools.com/tools/staruml.php

Netbeans. (2018). *NetBeans IDE Features*. Obtenido de https://netbeans.org/features/

ogcio. (2018). *IT Infrastructure & Standards*. Obtenido de https://www.ogcio.gov.hk/en/our_work/infrastructure/methodology/system_development/past_documents/oom/

Rouse, M. (20 de Septiembre de 2013). *An admin's guide to AWS data management*. Obtenido de MySQL: https://searchoracle.techtarget.com/definition/MySQL

SimCom. (2018). *SIM900* . Obtenido de http://www.propox.com/download/docs/SIM900.pdf

Techopedia. (2018). *Java Swing* . Obtenido de https://www.techopedia.com/definition/26102/java-swing

Visual-Paradigm. (2017). Obtenido de What is Unified Modeling Language (UML)?: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/

CONCLUSIONES

El sistema informático obtuvo resultados eficientes por Hospital del Día Gabriel Andrade; del Cantón Ventanas, al cumplir todos los objetivos planteados. El sistema informático para la gestión de citas notificación de los valores a pagar de las misma a través de mensajes de texto, ayudará de forma eficiente al hospital en brindar un servicio de calidad y eficiente.

El sistema informático está diseñado en base a la metodología orientada a objetos metodología de gran notoriedad utilizada en la elaboración de software y el patrón de diseño MVC (Modelo Vista Controlador), muy conveniente para futuras actualizaciones y fácil mantenimiento. Se utilizó el lenguaje UML (Lenguaje de Modelado Unificado) para visualizar, especificar, construir y evidenciar el funcionamiento del sistema.

Para desarrollar el sistema informático se utilizó la JAVA, que va muy acorde con la metodología de desarrollo escogida. La programación orientada a objetos encapsula la información procesada brindando mayor seguridad a los datos.

Aunque se ha utilizado en su mayor parte software open source para la construcción de este prototipo, se debe pagar el servicio de mensajería SMS a una determinada operadora; para la realización de las pruebas se utilizó una tarjeta SIM de la red móvil de CLARO.

RECOMENDACIONES

La gran mayoría de empresas ecuatorianas mantiene importantes niveles de informalidad en sus procesos que afectan al rendimiento de las mismas, en la calidad de servicios de las mismas, y en general al desarrollo económico y social del país.

El prototipo descrito en el presente trabajo representa un proyecto que puede aportar significativamente a la incorporación de la tecnología en el área de salud con un amplio conjunto de agentes que permiten mejorar la calidad de servicio brindado, es por ello que este debe ser implementado y puesto en producción.

Los datos que se gestionen en el sistema informático deben ser administrados con mucha confidencialidad, ya que estos datos son personales para el paciente y deben manejarse con mucha prudencia por parte de los usuarios que manipulen el sistema puesto que este es uno de los objetivos de la seguridad informática.

Las credenciales de acceso al sistema, son confidenciales para cada usuario y estas no deben ser compartidas a otras personas, mucho menos a personas ajenas al sistema y a la institución, ya que cualquier intruso podría causar daños al sistemas, pérdidas o robo de información valiosa de la institución.

ANEXOS

1. Ficha de observación.

Ficha de Observación	**Proyecto:** Sistema Informático de Facturación de citas y detalle de tratamientos médicos en el Hospital del Día "Gabriel Andrade" del Cantón Ventanas.
	Responsable: Kevin Mejía
Fecha: 22 de enero 2018	**OBSERVACIÓN**
Hora: 09:30	Hospital del Día "Gabriel Andrade" del Cantón Ventanas brinda servicios médicos a la ciudadanía en general.
Lugar: Hospital del Día "Gabriel Andrade" del Cantón Ventanas	Se observó una red LAN y cableado estructurado, 3 máquinas clientes en los diferentes departamentos.
	El cable es UTP categoría 6. Se usa una topología en Árbol.
	Los equipos de escritorio cuentan con el sistema operativo Windows 7.
	Se notó que las tareas administrativas y facturación se hacen de forma manual y usando software de ofimática.

2. Visitas al hospital

Foto 1.- Hospital del Día

Foto 2.- Recepción

Foto 3.- Recepcionistas

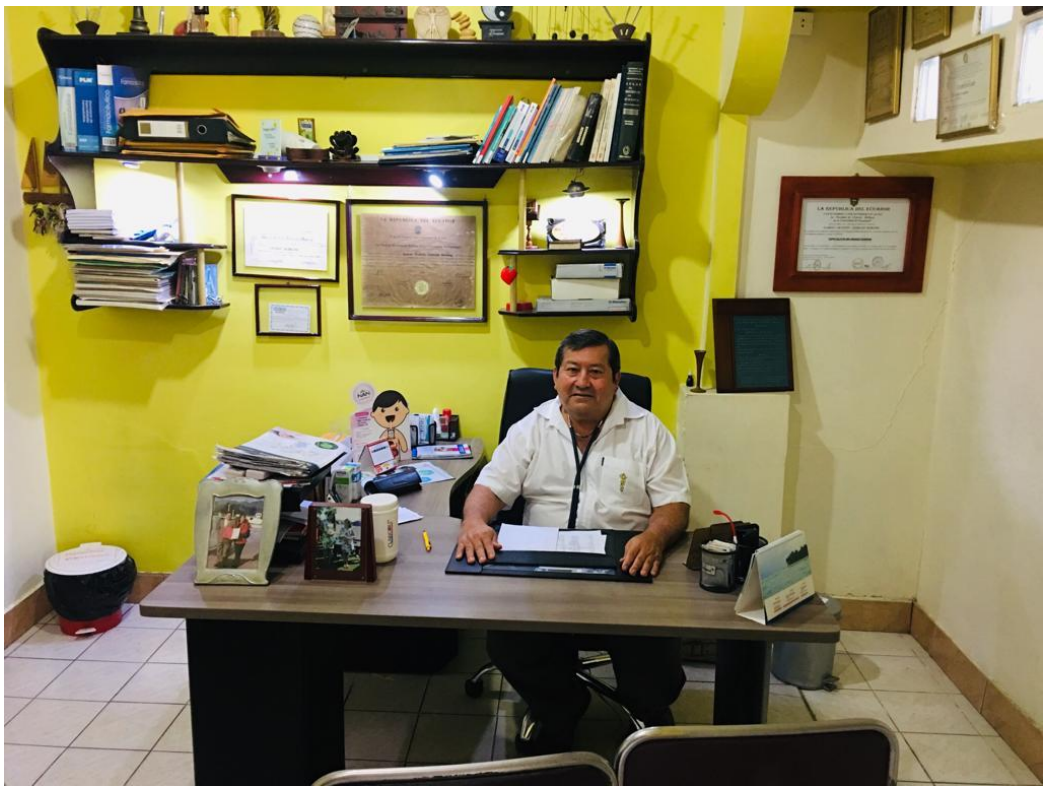

Foto 4.- Director del Hospital

Foto 5.- El director con una de sus pacientes

Foto 6.- Entrevista al director