Universidad de Matanzas

Facultad de Ciencias Técnicas

Especialidad Ingeniería Informática

Trabajo para optar por el Título de Ingeniero en Informática.

Software de gestión de la información del restaurant: San Severino

Autor: Malena Amaya Rodríguez

Tutor: Dayana Olivia Hernández Revilla

Matanzas, 2020

"El arte desafía a la tecnología y la tecnología inspira al arte".

John Lasseter

Dedicatoria

A mi mamá sin ella esto hoy no sería posible

Agradecimientos:

- A mi mamá que ha sido mi motor en este camino para impulsarme y convertirme en una gran profesional.

- A Ernesto Lantigua Montes de Oca por su apoyo incondicional al que admiro y quiero como a un padre.

- A mi abuela Ana Perdomo de la cual recibí las mejores ayudas en todos estos años.

- A mi amiga en todas: Lorena Varela García que aun cuando no continuamos esta carrera juntas siempre tuvo tiempo para brindarme todo su apoyo, ella sabe que yo la adoro y que muchos de los créditos son suyos.

- A mi tutora: Dayana Olivia Hernández Revilla por sus enseñanzas, por sus consejos pero sobre todo por su paciencia.

- A los profesores de la Universidad y del Departamento de Informática que durante esta etapa me formaron como profesional.

A todos,

Un Millón de Gracias

Opinión del Tutor del Trabajo de Diploma.

Declaración de autoría

Yo, Malena Amaya Rodríguez, declaro que soy el único autor de este trabajo y autorizo a la Universidad de Matanzas “Camilo Cienfuegos”, especialmente a la Facultad de Ciencias Técnicas a que hagan el uso que estimen pertinente de él.

Y para que así conste, firmo la presente a los _____días del mes de _____ del 2020

_____ _____

Firma del Autor Firma del Tutor

Resumen

En el presente trabajo se describe la concepción e implementación de un software para la gestión de la información del Restaurante:" San Severino "ubicado en el centro de la ciudad de Matanzas, con el fin de mejorar el proceso económico de dicha entidad que se desarrolla por cuenta propia. El sistema automatizado tiene el propósito de proveer una herramienta que permita registrar, procesar y controlar de forma integrada toda la información correspondiente, lo que propiciará un mayor control, ahorro de tiempo y recursos.

La investigación realizada posee un aporte práctico en la recopilación, almacenamiento, procesamiento y gestión, de forma eficiente y segura de todas las informaciones referentes al Restaurante: "San Severino"; lo que permite garantizar la planificación del flujo de trabajo y realizar análisis profundos y necesarios para la toma de decisiones de forma tal que se eviten errores que se comenten en la persistencia, manipulación y extracción de la información.

Abstract

In the present work describes the conception and implementation. The automated system has the purpose of providing the entity with an efficient tool that allows to register, process and control in an integrated manner all the corresponding information, which will lead to greater control, saving time and resources.

The research carried out has a practical contribution in the collection, storage, processing and management, in an efficient and safe way of all the information related to the Career Collective in the University of Matanzas; what allows to guarantee an efficient planning of the flow of work and to carry out deep and necessary analyzes for the decision making in such a way that errors are avoided that are commented on the persistence, manipulation and extraction of the information.

Contenido

Índice de Tablas

Índice de ilustraciones

.Introducción

La Informatización de la Sociedad ha sido una de las estrategias prioritarias del Estado cubano desde hace algunos años. El país ha identificado desde muy temprano la conveniencia y necesidad de dominar e introducir en la práctica social las Tecnologías de la Información y las Comunicaciones y lograr una cultura digital como una de las características imprescindibles del hombre nuevo, lo que facilitaría a nuestra sociedad acercarse más hacia el objetivo de un desarrollo sostenible. El impacto social de las Tecnologías de la Información y las Comunicaciones (TIC) toca muy de cerca a las entidades que se desarrollan por cuenta propia y que hoy en día contamos con un sin número de ellas, especialmente restaurantes, propiciando modificaciones en las formas tradicionales de llevar a cabo y de manera más factible el manejo de su información.

Montar un restaurante es mucho más que alquilar un local, contratar personal y organizar un menú. La administración de restaurantes de forma adecuada es realmente complicada y requiere realizar ciertos equilibrios. Actualmente la gestión de la información del Restaurante San Severino no se realizan con medios y soportes que garanticen la seguridad de los datos, se hace engorroso mantener un control exacto ya que la misma se almacena en formato duro que se archivan por períodos muy largos y su manipulación por parte de los administradores y cajeros trae consigo errores recurrentes, poca planificación y posible pérdida de información, siendo así la **situación problémica** existente en el Restaurante San Severino.

Por todo lo anterior se constata la existencia de un problema real a resolver que se convierte en el **problema científico** de la siguiente manera: ¿Cómo facilitar la gestión de la información del Restaurante San Severino?

Como **objeto de estudio** de esta investigación se tiene la gestión de la información del Restaurante San Severino

El **campo de acción** es la informatización del proceso de gestión de la información del Restaurante San Severino

Del problema anterior se deduce la siguiente **hipótesis**: Si se implementa un software entonces se incrementará la gestión de la información del Restaurante San Severino

Objetivo General: desarrollar un software para la gestión de información del Restaurante San Severino

Los **objetivos específicos** de este trabajo son:

1. Determinar los referentes teóricos – metodológicos sobre la gestión de información del Restaurante San Severino
2. Determinar las tendencias tecnológicas para la realización de un software para la gestión de información del Restaurante San Severino
3. Implementar una propuesta de un software para la gestión de la Información del Restaurante San Severino
4. Validar el software para la gestión de información del Restaurante San Severino

Métodos empíricos

Entrevista: Este método fue esencial para el desarrollo de la investigación ya que permitió conocer las necesidades reales del cliente y los beneficios que podría aportar la aplicación.

Estudio documental: El análisis de documentos incluye la revisión y estudio de toda la información archivada para comprender las funcionalidades de los dueños, administradores y cajeros ya que son posibles usuarios del software.

Observación: La observación fue necesaria desde el comienzo para comprender como se realizaban los procesos y detectar sus insuficiencias para posteriormente darles solución.

Métodos teóricos:

Histórico - Lógico: Permitió estudiar la trayectoria, evolución y desarrollo de los sistemas de gestión de la información en los restaurantes, directamente dirigido al Restaurante San Severino

Análisis - Síntesis: Este método se evidenció en el análisis de los elementos de la situación problemática, posibilitando descomponer mentalmente un todo complejo en sus partes y cualidades, para luego sobre la base de los resultados alcanzados previamente por el análisis establecer mentalmente la unión entre las partes, permitiendo descubrir relaciones y características generales entre los elementos de la realidad.

Inductivo - Deductivo: Su uso fue necesario tanto en la revisión bibliográfica, como en el análisis de los resultados, permitiendo arribar a conclusiones que se infirieron a partir de propiedades y relaciones existentes entre los elementos que conforman el fenómeno objeto de estudio.

La tesis está estructurada en una introducción, tres capítulos, las conclusiones, las recomendaciones, la bibliografía y anexos

En el Capítulo 1 se encuentra descrito el **Marco Teórico-Referencial**, se hace una descripción de cada una de las herramientas que se utilizan para la confección del sistema

En el capítulo 2 se realiza el **Análisis, Diseño de la Solución Propuesta y Construcción** donde se argumenta la solución que se propone al problema de investigación, presentando una planificación inicial del proyecto, con el empleo de la metodología ágil de desarrollo de software Programación Extrema. Se construye la solución propuesta, presentando una planificación por iteraciones.

En el capítulo 3 se presenta la **Validación de la Solución Propuesta** se realizan pruebas funcionales y se hace un análisis de los resultados obtenidos, basándose en el criterio del clientes y los propios de la metodología de software. Se realiza además un estudio de los beneficios tangibles e intangibles como resultado de la realización del proyecto de software. Las pruebas son una actividad en la cual un sistema es ejecutado bajo condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema. Se realiza la validación ala solución propuesta, con el objetivo de comprobar la eficiencia de las clases u operaciones utilizadas para dar respuesta a los distintos requisitos planteados por el cliente.

Finalmente, se presentan las **Conclusiones y Recomendaciones** de la investigación para dejar el camino abierto a futuros estudios relacionados con la temática abordada.

Asimismo, quedan recogidas las **Bibliografías** empleadas y **Anexos** que fueron necesarios para el desarrollo de todo el trabajo y un mejor entendimiento del mismo.

Capítulo 1. Marco Teórico Referencial

1.1 Introducción

El Restaurante San Severino tiene el desafío de lograr la calidad del negocio que desarrolla mediante la creatividad y flexibilidad para manejar la información. Este capítulo contiene las principales definiciones asociadas al dominio del problema a resolver, los antecedentes del trabajo, el objeto de estudio; se abordan también los aspectos relacionados con las tendencias tecnológicas, tales como las tecnologías que influyeron en el desarrollo del software y la metodología de ingeniería del software a utilizar, el servidor de base de datos y los lenguajes utilizados.

1.2 Descripción de los procesos que serán automatizados

Será objeto de automatización el proceso de la gestión de la información en el restaurante San Severino, dicho proceso desemboca en lo relacionado con las ventas diarias, gastos, ganancias, existencia de productos en almacén, creación de menú, salario que se determinara según la plaza que ocupe cada trabajador de este último se archivara todo la información que corresponda, las comandas serán automatizadas y mediante ellas se obtendrán todos los datos necesarios para el cierre del día a través de un arqueo de caja que efectuara la cajera en turno. El presente software pretende facilitar todo este proceso ya que lo genera automáticamente.

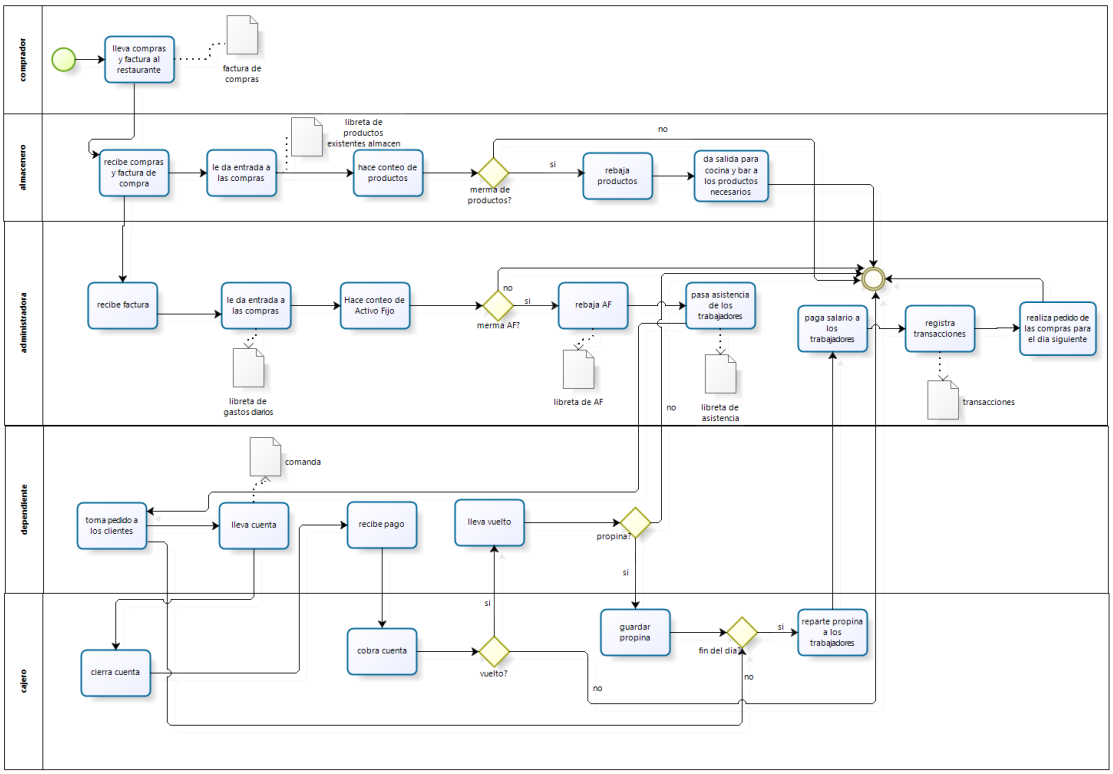

Ilustración 1Diagrama de flujo del negocio. Elaboración propia

1.3 Antecedentes de la investigación

Software	Características	Resultados
Epos Now	Te permite manejar tu inventario fácilmente, mejora el rendimiento de tu personal, y realiza reportes por medio de un panel personalizado.	Este software permitirá acceder a la información en tiempo real de productos y ventas , también brinda la seguridad de tener siempre provisiones de los productos que más se venden; pero no gestiona la información referente a recursos humanos: nominas, contratos, ausencias e información de empleados, no permite

		calcular ganancias, no gestiona menu, mesas, comandas, propina.
Bacon	Ofrece visualización conjunta del restaurante, hace reportes semanales de los gastos principales. Hace reportes avanzados de inventario.	Este software permite hacer un seguimiento de los gastos en comida, bebida y mano de obra, así como a compararlos y crear un nuevo presupuesto que se ajuste a las necesidades de los dueños; pero no gestiona la información referente a recursos humanos, no calcula ganancias, no automatiza tareas como la creación de informes, no gestiona menu, mesas, comandas, propina.
MarketMan	Costo de alimentos, gestión de inventario, gestión de cocina, gestión de menu.	Este software facilita el proceso de obtención de todos los productos e inventario, se obtiene el costo preciso de los alimentos, de esta manera sabes a donde va tu dinero en tiempo real. Te mantiene informado constantemente: puedes configurar alertas cuando los artículos del menu sean menos rentables y recibir notificaciones cuando

		fluctúen los precios del los proveedores; pero no gestiona la información referente a recursos humanos, comandas, mesas, propinas y además su licencia es de pago.
Aplicación web para la gestión económica y promoción del restaurante Don Qko.	Garantiza la promoción del restaurante, gestiona unidad de medidas, mercancías, útiles, personal, consumibles, mesas, cuentas gastos y crea reportes.	Esta aplicación es útil para el restaurante en función de su información pero no gestiona el pago a los trabajadores, propinas, ni menu.

1.4 Fundamentación de la metodología a utilizar

Para la elaboración de la aplicación que se propone como solución al problema planteado es necesario un conjunto de técnicas y herramientas que permitan agilizar el proceso, así como una metodología de desarrollo de software que sirva de guía en el transcurso del mismo.

Las metodologías de desarrollo de software tradicionales aparecen como pesadas y de mucha documentación cuando se trata de proyectos pequeños como el que ocupa esta tesis. Como respuesta a esto, se ha visto en los últimos tiempos el surgimiento de “Metodologías Ágiles”. Estos nuevos métodos buscan un punto medio entre la ausencia de procesos y el abuso de los mismos, proponiendo un proceso cuyo esfuerzo valga la pena. Los métodos ágiles cambian significativamente algunos de los énfasis de las metodologías “clásicas”(RocketTheme, 2009):

- Los métodos ágiles son adaptables en lugar de predictivos. Los métodos “clásicos” tienden a intentar planear una gran parte del proceso del software en gran detalle para un plazo largo de tiempo. Para los métodos ágiles, no obstante, el cambio es bienvenido. Intentan ser procesos que se adaptan y crecen en el cambio.
- Los métodos ágiles son orientados a la gente y no orientados al proceso. El objetivo de los métodos “clásicos” es definir un proceso que funcionará bien independientemente de quien lo utilice. Los métodos ágiles afirman que ningún proceso podrá nunca maquillar las habilidades del equipo de desarrollo, de modo que el papel del proceso es apoyar al equipo de desarrollo, los métodos ágiles son adaptables en lugar de predictivos. Los métodos “clásicos” tienden a intentar planear una gran parte del proceso del software en gran detalle para un plazo largo de tiempo. Para los métodos ágiles, no obstante, el cambio es bienvenido. Intentan ser procesos que se adaptan y crecen en el cambio.
- Los métodos ágiles son orientados a la gente y no orientados al proceso. El objetivo de los métodos “clásicos” es definir un proceso que funcionará bien independientemente de quien lo utilice. Los métodos ágiles afirman que ningún proceso podrá nunca maquillar las habilidades del equipo de

desarrollo, de modo que el papel del proceso es apoyar al equipo de desarrollo de trabajo.

1.4.1 Metodología de desarrollo de software: XP

"Extreme Programming" o "Programación Extrema" es una de las llamadas metodologías ágiles de desarrollo de software más exitosas de los tiempos recientes. La metodología propuesta en XP está diseñada para entregar el software que los clientes necesitan en el momento en que lo necesitan. XP alienta a los desarrolladores a responder a los requerimientos cambiantes de los clientes, aún en fases tardías del ciclo de vida del desarrollo. La metodología también enfatiza el trabajo en equipo. Tanto gerentes como clientes y desarrolladores son partes del mismo equipo dedicado a entregar software de calidad.(J. Fuentes, 2015)

Extreme Programming (XP) surge como una nueva manera de encarar proyectos de software, proponiendo una metodología basada esencialmente en la simplicidad y agilidad. El ciclo de vida de un proyecto XP incluye, entender lo que el cliente necesita, estimar el esfuerzo, crear la solución y entregar el producto final al cliente. XP propone un ciclo de vida dinámico, donde se admite expresamente que en muchos casos los clientes no son capaces de especificar sus requerimientos al comienzo de un proyecto.

Por esto, se trata de realizar ciclos de desarrollo cortos (llamados iteraciones), con entregables funcionales al finalizar cada ciclo. En cada iteración se realiza un ciclo completo de análisis, diseño, desarrollo y pruebas. XP propone que se desarrolle en pares de programadores, ambos trabajando juntos en un mismo ordenador. Si bien parece que ésta práctica duplica el tiempo asignado al proyecto (y por ende, los costos en recursos humanos), al trabajar en pares se minimizan los errores y se logran mejores diseños, compensando la inversión en horas. El producto obtenido es por lo general de mejor calidad que cuando el desarrollo se realiza por programadores individuales.(I. J. Joskowicz, 2012)

1.6 Herramientas y tecnologías utilizadas

1.6.1Tendencias tecnológicas a considerar

Arquitectura Cliente Servidor:

Es una arquitectura de procesamiento cooperativo de información donde uno de los componentes pide servicios a otro. Es una relación entre procesos corriendo en máquinas separadas. El cliente es un consumidor de servicios y el servidor es un proveedor de servicios e interactúan por un mecanismo de pasaje de datos: pedido de servicios – respuesta.

Cliente: Una estación de trabajo o microcomputador (PC: Computadora Personal) conectado a una red que le permite acceder y gestionar una serie de recursos. En este caso se refiere a un microcomputador conectado al sistema de información y en el que se realiza una parte mayoritaria de los procesos.

Servidor o Back-end: Está dedicado a responder a los requerimientos del cliente. Los servidores pueden estar conectados a los clientes a través de redes LANs o WANs y suministran servicios tales como acceso a base de datos, archivos, comunicaciones, impresión, procesamiento de imágenes, correo entre otros. Este mecanismo favorece flexibilidad y dinamismo en las organizaciones.(M. Á. Álvarez, 2007)

Patrón de diseño Modelo Vista Controlador

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos, email, entre otras). El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación(Tedeschi, 2013):

▪ El modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.

▪ La vista transforma el modelo en una página web que permite al usuario interactuar con ella.

▪ El controlador se encarga de procesar las interacciones del usuario, manejo de las peticiones, de la seguridad, cargar la configuración de la aplicación y realiza los cambios apropiados en el modelo o en la vista**.**

Sistema Gestor de Base de Datos (SGBD)

Un Sistema Gestor de Base de Datos es un conjunto de programas no visibles que administran y gestionan la información que contiene una base de datos. A través de él se maneja todo acceso a la base de datos con el objetivo de servir de interfaz entre ésta, el usuario y las aplicaciones.

SQLite

A diferencia de los sistemas de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que las comunicaciones entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un solo fichero estándar en la maquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción.

PostgreSQL

Es un servidor de base de datos objeto relacional libre, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipo de datos, funciones, restricciones, disparadores, reglas e integridad transaccional, liberado bajo la licencia BSD. Actualmente estos son llamados objetos. (21 ene. 2008)

Ventajas de PostgreSQL

Instalación ilimitada y gratuita: Podemos instalarlo en todos los equipos que queramos.

Gran escalabilidad: Nos permite configurar PostgreSQL en cada equipo según el hardware.

Estabilidad y confiabilidad: Tiene más de 20 años de desarrollo activo y en constante mejora.

1.6.2Lenguajes de programación:

Entre los distintos lenguajes de programación para la Web que existen en la actualidad, se destacan dos grupos, que se diferencian entre sí por el lugar que ocupan en la arquitectura Cliente/Servidor, característica de los sistemas Web.

- El primer grupo lo integran los lenguajes que se ejecutan en el cliente, ejemplos de ellos son: HTML, CSS, JavaScript, estos se encargan de aportar dinamismo a la aplicación en los navegadores.
- El segundo grupo está formado por los lenguajes que se ejecutan en el servidor, dentro de los que se encuentra Python, estos lenguajes se caracterizan por desarrollar la lógica de negocio dentro del Servidor, además de ser los encargados del acceso a bases de datos y el tratamiento de la información.

Un lenguaje del lado cliente es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio. El navegador es una especie de aplicación capaz de interpretar las órdenes recibidas en forma de código HTML fundamentalmente y convertirlas en las páginas que son el resultado de dicha orden.

Del lado del cliente:

HTML5 (HyperTextMarkupLanguage)

Es un lenguaje muy sencillo que permite describir hipertexto, es decir, texto presentado de forma estructurada y agradable, con enlaces (hyperlinks) que conducen a otros documentos o fuentes de información relacionadas, y con inserciones multimedia (gráficos, sonido). Está compuesto por una serie de etiquetas que el navegador interpreta y da forma en la pantalla. La utilización de HTML como uno de los lenguajes para el desarrollo, responde a las necesidades de manipulación y maquetación de los elementos visuales de la aplicación. La descripción se basa en especificar en el texto la estructura lógica del contenido (títulos, párrafos de texto normal, enumeraciones, definiciones, citas, entre otras), así como los diferentes efectos que se quieren dar (cursiva, negrita, o un gráfico determinado) y dejar que luego la presentación final de dicho hipertexto se realice por un programa especializado(R. Álvarez, 2012).

CSS 3 (Cascade StyleSheets)

Hojas de estilo en cascada (o CSS, siglas en inglés de CascadingStylesheets) no es como tal un lenguaje de programación, sino un lenguaje de estilos, con el cual se define la apariencia de una página web definiendo como debe mostrarse cada uno de sus elementos HTML. CSS está diseñado principalmente para marcar la separación del contenido del documento y la forma de presentación de este, características tales como las capas, los colores y las fuentes. Esta separación busca mejorar la accesibilidad del documento, proveer más flexibilidad, permitir que varios documentos HTML compartan un mismo estilo usando una sola hoja de estilos separada en un archivo .css, y reducir la complejidad y la repetición de código en la estructura del documento.(Cristian, 2008)

JavaScript

Es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario.

Del lado del servidor:

Python

Es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional.

Ventajas

Simplificado y rápido. Este lenguaje simplifica mucho la programación “hace que te adaptes a un modo de lenguaje de programación, Python te propone un patrón”. Es un gran lenguaje para scripting, si usted requiere algo rápido (en el sentido de la ejecución del lenguaje), con unas cuantas líneas ya está resuelto.

Elegante y flexible. El lenguaje le da muchas herramientas, si usted requiere listas de varios tipo de datos es un lenguaje tan flexible que usted no se preocupa tanto por los detalles.

Ordenado y limpio. El orden que mantiene Python, es de lo que más le gusta a sus usuarios, es muy legible, cualquier otro programador lo puede leer y trabajar sobre el programa escrito en Python. Los módulos están bien organizados, a diferencia de otros lenguajes.

Portable. Es un lenguaje muy portable (ya sea en Mac, Linux o Windows) en comparación con otros lenguajes. La filosofía de baterías incluidas, son las librerías que mas usted necesita al día a día de programación, ya están dentro del intérprete, no tiene la necesidad de instalarlas adicionalmente como en otros lenguajes.

Comunidad. Algo muy importante para el desarrollo de un lenguaje es la comunidad, la misma comunidad de Python cuida el lenguaje y casi todas las actualizaciones se hacen de manera democrática.

1.6.3Framework de desarrollo web

Un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un framework proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener.

Por último, un framework facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

Django

Es un framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como MVC.

Ventajas

Es universal y escalable, lo que lo convierte en el framework de primera elección para la mayoría de los desarrolladores web. Se adapta a cualquier proyecto, tanto web como móvil, incluida aplicaciones grandes que tienen que lidiar con enormes cantidades de datos.

Bootstrap 4.0.0

Es un framework web de código abierto desarrollado por Twitter. Contiene plantillas para casi todo, como plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS y JavaScript. La utilización de este framework, ahorra la parte de tener que escribir tu propio código CSS u hoja de estilos, ya que ya viene todo montado y tan solo tienes que añadir ciertas clases a tus elementos HTML para que la página cobre vida como por arte de magia. Sus posibilidades son ilimitadas(Solanas & Sierra).

1.7 Conclusiones del capítulo

En el desarrollo del primer capítulo se dio a conocer las bases teóricas sobre las cuales se sustenta la propuesta de trabajo. Se analiza cómo se efectúa el proceso y se corroboró la necesidad de diseñar este software para solucionar los problemas existentes. Se realiza un estudio detallado de las tendencias existentes para el desarrollo de software y se definen las tecnologías y técnicas que se aplicarán al diseño e implementación del software informático. Se justifica la utilización de una metodología ágil para el desarrollo de la aplicación, específicamente la metodología de programación extrema XP. Se realiza un análisis detallado de las tecnologías a utilizar optando por utilizar el

lenguaje de programación Python y usando los gestores de bases de datos SQLite y PostgreSQL

. Capítulo 2: Descripción de la solución propuesta.

2.1 Introducción al capítulo

En el capítulo se exponen los elementos necesarios para la descripción de la solución propuesta. A través de las Historias de Usuarios (HU) que acumulan la necesidad existente definida por el cliente, es llevado a cabo el análisis de los requerimientos. Se aplica la Metodología XP, Extreme Programming (Programación Extrema), metodología ágil de desarrollo, con el objetivo de garantizar el diseño de un programa lo más ajustado posible y se logra como ventaja la incorporación del cliente como un miembro del equipo de desarrollo.

2.2 Descripción de la solución

Se propone desarrollar un software que gestione la información del Restaurante “San Severino” de forma tal que los datos existentes puedan ser almacenados sin riesgo de que puedan perderse o deteriorarse y sea fácil consultarlos. Les permitirá a los dueños del negocio llevar a cabo el proceso económico con mayor facilidad estará automatizado y los cajeros podrán realizar el cierre de comandas con mayor agilidad, lo que implicaba antes mas tiempo. Con el desarrollo de este software la información va a estar más organizada y segura

2.3 Etapa de planificación

La metodología XP plantea la planificación como un diálogo continuo entre las partes involucradas en el proyecto, incluyendo al cliente, a los programadores y a los coordinadores o gerentes. El proyecto comienza recopilando “Historias de usuarios”, las que sustituyen a los tradicionales “casos de uso”. Una vez obtenidas las “Historias de Usuarios”, los programadores evalúan rápidamente el tiempo de desarrollo de cada una. Si alguna de ellas tiene “riesgos” que no permiten establecer con certeza la complejidad del desarrollo, se realizan pequeños programas de prueba, para reducir estos riesgos. Una vez realizadas estas estimaciones, se organiza una reunión de planificación, con los diversos actores del proyecto (cliente, desarrolladores, gerentes), a los efectos de establecer un plan o cronograma de entregas en los que todos estén de

acuerdo. Una vez acordado este cronograma, comienza una fase de iteraciones, en dónde en cada una de ellas se desarrolla, prueba e instala unas pocas “historias de usuarios”.(I. Joskowicz, 2012)

2.3.1 Equipo de trabajo y roles

La metodología XP define roles de trabajo asociando a cada uno con diversas actividades. A continuación, se definen los roles, quedando designado el programador que sería el encargado de producir el código del sistema, el jefe de proyecto y el cliente que no es más que el que escribe las historias de usuario, les asigna la prioridad y diseña las pruebas funcionales para validar su implementación. A continuación, se muestra la asignación de estos roles a las personas responsables(I. J. Joskowicz, 2012).

Miembros	Roles
Malena Amaya Rodríguez	Programador
Dayana Olivia Hernández Revilla	Jefe del Proyecto, Tester,
Dunia Betancourt	Cliente

Tabla 1 Equipo de trabajo y roles

2.3.2 Historias de Usuario Iniciales

Las “Historias de usuarios” sustituyen a los documentos de especificación funcional, y a los “casos de uso”. Estas “historias” son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. La diferencia más importante entre estas historias y los tradicionales documentos de especificación funcional se encuentra en el nivel de detalle requerido. Las historias de usuario deben tener el detalle mínimo como para que los programadores puedan realizar una estimación poco riesgosa del tiempo que llevará su desarrollo. Cuando llegue el momento de la implementación, los desarrolladores dialogarán directamente con el cliente para obtener todos los detalles necesarios. Son utilizadas para estimar tiempos de desarrollo de la parte del software que describen. También se utilizan en la fase de pruebas, para verificar si el software cumple con lo que especifica la historia de usuario(J. R. L. Fuentes, 2015).

Escalas equivalentes a la prioridad en el negocio:

Alta: Asignada a las Historias de Usuario que corresponden a funcionalidades esenciales en el desarrollo del proyecto, a las que el cliente define como primordiales.

Media: Dada a las Historias de Usuario que resultan para el cliente como funcionalidades a tener en cuenta, sin que estas tengan una afectación directa sobre el proyecto que se esté desarrollando.

Baja: Se le otorga a las Historias de Usuario que constituyen funcionalidades que sirven de ayuda al control de elementos asociados al equipo de desarrollo, a la estructura y no tienen nada que ver con el proyecto en desarrollo

Escala Nominal de Riesgo en Desarrollo:

Alta: Cuando para la implementación de la Historia de Usuario se considera la posible existencia de errores que lleven a inoperatividad del código.

Media: Cuando pueden aparecer errores en la implementación de la Historia de Usuario que puedan retrasar la entrega de la versión.

Baja: Cuando pueden aparecer errores que serán tratados con relativa facilidad sin que traigan perjuicios para el desarrollo del proyecto.

A continuación se colocan los requisitos funcionales detectados:

No	Nombre	Prioridad	Riesgo	Esfuerzo	Iteracion	Entrega
1	Diseño y Creación de la base de datos	Alta	Alto	2	1	1
2	Diseño de la interfaz de usuario	Media	Medio	1		
3	Autentificación del usuario	Alta	Medio	1		

4	Gestionar usuario	Alta	Alto	1		
5	Gestionar almacenes	Alta	Alto	1		
6	Gestionar compras	Alta	Medio	1		
7	Gestionar mermas	Medio	Medio	1		
8	Gestionar familia	Alta	Medio	1		
9	Gestionar unidad de medida	Medio	Medio	1		
10	Gestionar productos	Alta	Medio	1	2	2
11	Gestionar activo fijo	Medio	Medio	1		
12	Gestionar compra de ACTIVO FIJO	Medio	Medio	1		
13	Gestionar merma de ACTIVO FIJO	Medio	Medio	1		
14	Gestionar trabajadores	Alta	Medio	1		
15	Gestionar cargo	Medio	Medio	1		
16	Gestionar asistencia	Medio	Medio	1	3	3

17	Gestionar salarios	Medio	Medio	1		
18	Gestionar cuentas	Alta	Alto	1	4	4
19	Gestionar ventas	Alta	Alto	1		
20	Gestionar transacciones económicas	Alta	Alto	1		
21	Gestionar mesas	Alta	Alto	1		
22	Gestionar pedidos	Alta	Alto	1		
23	Gestionar menu	Alta	Alto	1		
24	Gestionar cierre	Alta	Alto	1		
25	Gestionar reportes	Medio	Medio	2		
26	Gestionar contadores	Medio	Medio	1		
27	Gestionar lectura de contadores	Medio	Medio	1		
28	Calcular consumo eléctrico	Medio	Medio	1		

Tabla 2 Planificación de las Historias de Usuario

Una vez detectadas los requisitos pues es importante elaborar el plan de entrega y de iteraciones.

2.3.3 Plan de Iteraciones

Las historias de usuarios seleccionadas para cada entrega son desarrolladas y probadas en un ciclo de iteración, de acuerdo al orden preestablecido. Al comienzo de cada ciclo, se realiza una reunión de planificación de la iteración. Cada historia de usuario se traduce en tareas específicas de programación.

El proyecto fue dividido en cuatro iteraciones, por lo que se obtuvo un total de cuatro entregas para las cuales se desarrollaron partes del software completamente funcionales. Para la determinación de cada una de las iteraciones se tuvo en cuenta la opinión del cliente a través de las entrevistas que se le realizaron continuamente antes de comenzar a desarrollar cada iteración donde se tomaron todos los acuerdos necesarios. Una vez concluida la iteración que va a estar conformada por un conjunto de historias de usuarios, teniendo en cuenta los requisitos exigidos por el cliente, el software poseerá mayor número de funcionalidades.

En la ilustración se muestran las cuatro iteraciones y las semanas en las que se desarrollan.

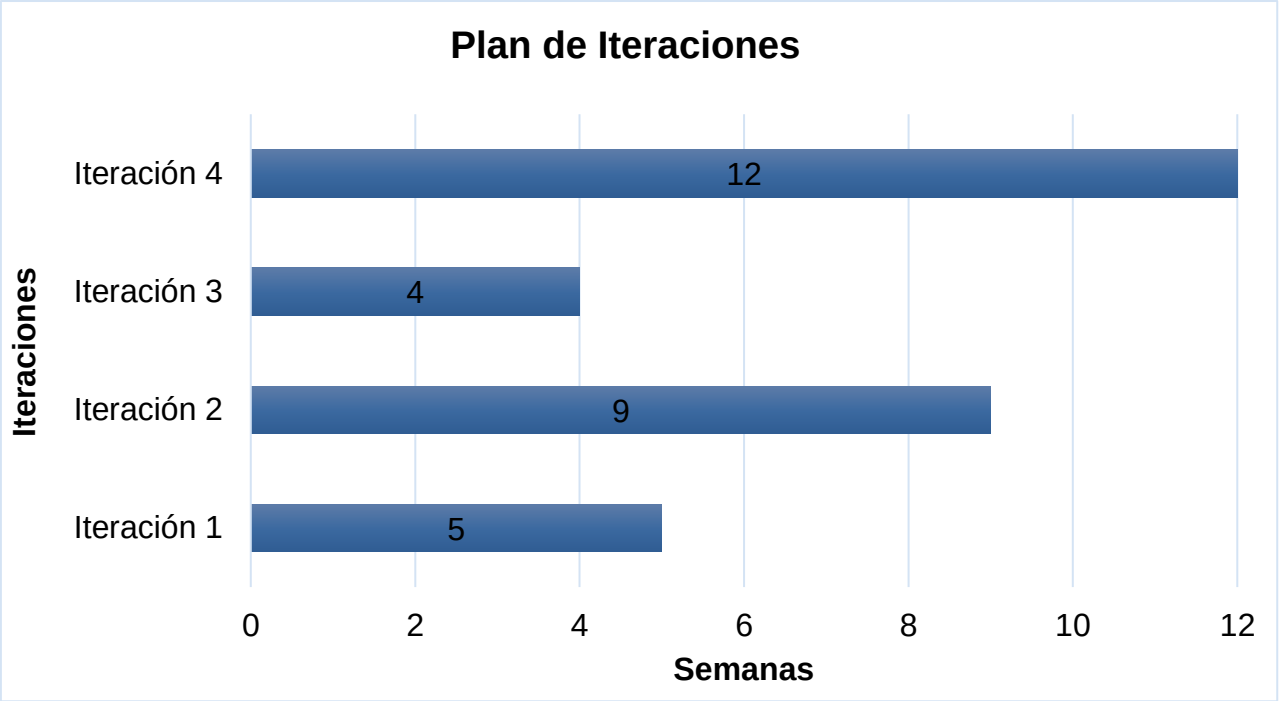

Ilustración 2 Plan de Iteraciones. Elaboración propia

2.3.4 Plan de Entregas

El cronograma de entregas establece qué historias de usuario serán agrupadas para conformar una entrega, y el orden de las mismas. Este cronograma será el resultado de una reunión entre todos los actores del proyecto. Típicamente el cliente ordenará y agrupará según sus prioridades las historias de usuario. El cronograma de entregas se realiza en base a las estimaciones de tiempos de desarrollo realizadas por los desarrolladores. Luego de algunas iteraciones es recomendable realizar nuevamente una reunión con los actores del proyecto, para evaluar nuevamente el plan de entregas y ajustarlo si es necesario.

Entrega	Descripción
1	Esta entrega comprende - Diseño y creación de la base de datos - Diseño de la interfaz de usuario - Autenticación - Gestión de usuario
	Esta entrega comprende - Gestionar almacenes

2	- Gestionar compras - Gestionar mermas - Gestionar familia - Gestionar unidad de medida - Gestionar productos - Gestionar activo fijo - Gestionar compra de activo fijo - Gestionar merma de activo fijo
3	Esta entrega comprende - Gestionar trabajadores - Gestionar cargo - Gestionar asistencia - Gestionar salario
4	Esta entrega comprende - Gestionar cuentas - Gestionar ventas - Gestionar transacciones económicas - Gestionar mesas - Gestionar pedidos - Gestionar menu - Gestionar cierre - Gestionar reportes - Gestionar contadores - Gestionar lectura de contadores - Calcular consumo eléctrico

Tabla 3 Plan de entrega

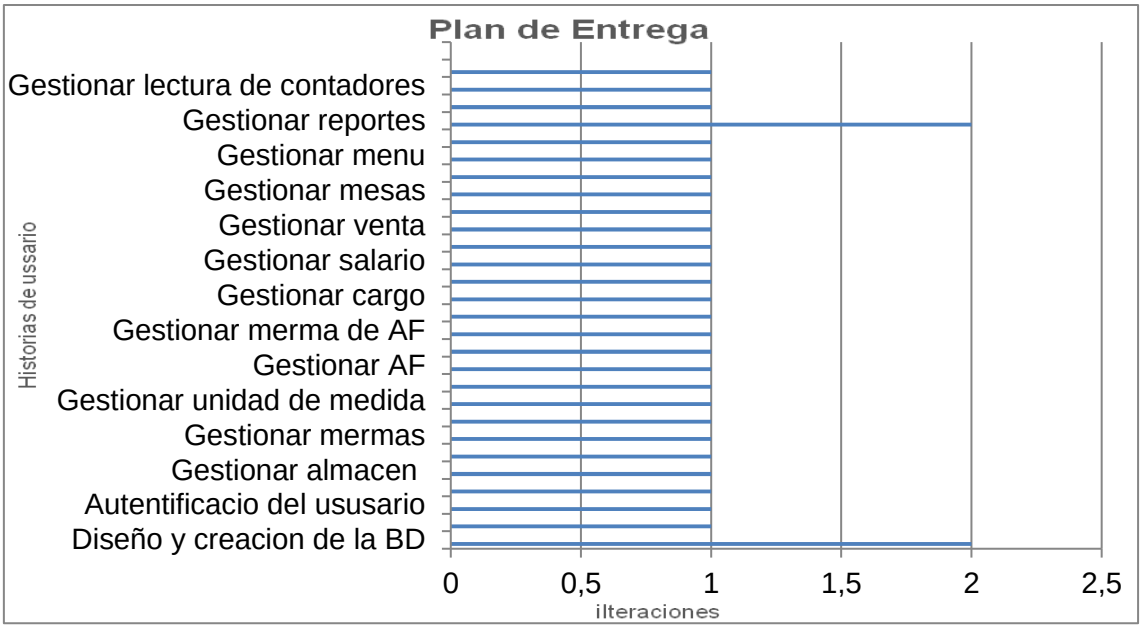

Ilustración 3Plan de entrega. Elaboración propia

2.4 Etapa de diseño

La metodología XP sugiere que hay que conseguir diseños simples y sencillos. Hay que procurar hacerlo todo lo menos complicado posible para conseguir un diseño fácilmente entendible que a la larga costará menos tiempo y esfuerzo desarrollar. Usar glosarios de términos y una correcta especificación de los nombres de métodos y clases ayudará a comprender el diseño y facilitará sus posteriores ampliaciones y reutilización del código. Si surgen problemas potenciales durante el diseño, XP sugiere utilizar una pareja de desarrolladores para que investiguen y reduzcan al máximo el riesgo que supone ese problema(I. J. Joskowicz, 2012).

A continuación se muestran las historias de usuarios que se consideran más importantes en detalle para que se pueda comprender el proceso:

Historia de Usuario	
Número: 1	**Usuario:** Super Admin
Nombre: Diseño y creación de la Base de datos	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Alto
Puntos estimados: 2	**Iteración Asignada**: 1

Programador responsable: Malena Amaya Rodríguez
Descripción: Se diseña e implementa la base de datos
Observaciones: Verificar si a la base de datos se le estableció una contraseña.

Tabla 4 Diseño y creación de la Base de datos

Historia de Usuario	
Número: 3	**Usuario:**Super admin, Administrador, cajero, almacenero
Nombre: Autentificación del Usuario	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada**: 1
Programador responsable: Malena Amaya Rodríguez	
Descripción: El usuario escribe su nombre y contraseña para acceder al sistema, de ser correctos se le mostrará un entorno de trabajo acorde con su nivel de acceso y de no serlo, el sistema denegara el acceso.	
Observaciones: Se debe verificar si son correctos los datos para acceder al sistema y si el usuario está registrado en las bases de datos	

Tabla 5 Autentificación del Usuario

Historia de Usuario	
Número: 4	**Usuario**:Super Administrador
Nombre: Gestionar Usuario	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada**: 1
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el Super admin accede a esta funcionalidad para gestionar los usuarios. Podrá insertar, editar y eliminar un usuario del sistema además de asignarle un rol y o permiso.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 6 Gestionar Usuarios

Historia de Usuario	
Número: 5	**Usuario**: Super admin; almacenero.
Nombre: Gestionar Almacenes	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Alto
Puntos estimados: 1	**Iteración Asignada:** 2
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un almacén. El usuario podrá insertar, editar y eliminar un almacén del sistema.	
Observaciones: Se debe verificar que los datos introducidos sean correctos.	

Tabla 7 Gestionar Almacenes

Historia de Usuario	
Número: 6	**Usuario**: Super admin, Almacenero
Nombre: Gestionar compra	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 2
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una compra. El usuario podrá insertar, editar y eliminar una compra del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 8 Gestionar Compra

Historia de Usuario	
Número: 7	**Usuario**:Super admin, Almacenero
Nombre: Gestionar merma	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 2

Programador responsable: Malena Amaya Rodríguez
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una merma. El usuario podrá insertar, editar y eliminar una merma del sistema.
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.

Tabla 9 Gestionar Merma

Historia de Usuario	
Número: 8	**Usuario**:Super admin, Almacenero
Nombre: Gestionar familia	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 2
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una familia. El usuario podrá insertar, editar y eliminar una familia del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 10 Gestionar familia

Historia de Usuario	
Número: 9	**Usuario**: Super admin, Almacenero
Nombre: Gestionar unidad de medida	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 2
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una um. El usuario podrá insertar, editar y eliminar una um del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 11 Gestionar unidad medida

Historia de Usuario	
Número: 10	**Usuario**:Super admin, Almacenero
Nombre: Gestionar productos	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 2
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un producto. El usuario podrá insertar, editar y eliminar un producto del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 12 Gestionar productos

Historia de Usuario	
Número: 11	**Usuario**: Super admin,Almacenero
Nombre: Gestionar Activo Fijo	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 2
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un ACTIVO FIJO. El usuario podrá insertar, editar y elimina un ACTIVO FIJO del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 13 Gestionar Activo fijo

Historia de Usuario	
Número: 12	**Usuario**:Super admin Almacenero
Nombre: Gestionar compra de ACTIVO FIJO	

Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 2
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una compra de ACTIVO FIJO. El usuario podrá insertar y editar una compra de ACTIVO FIJO del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 14 Gestionar compra de activo fijo

Historia de Usuario	
Número: 13	**Usuario**: Super admin,Almacenero
Nombre: Gestionar merma de ACTIVO FIJO	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 2
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una merma de ACTIVO FIJO. El usuario podrá insertar, editar y eliminar una merma de ACTIVO FIJO del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 15 Gestionar merma de activo fijo

Historia de Usuario	
Número: 14	**Usuario**: Super admin ,Administrador
Nombre: Gestionar trabajadores	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 3
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un trabajador. El usuario podrá insertar, editar y eliminar un trabajador del sistema.	

Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.

Tabla 16 Gestionar trabajadores

Historia de Usuario	
Número: 15	**Usuario**:Super admin, Administrador
Nombre: Gestionar Cargo	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 3
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un cargo. El usuario podrá insertar, editar y eliminar un cargo del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 17 Gestionar cargo

Historia de Usuario	
Número: 16	**Usuario**: Super admin, Administrador
Nombre: Gestionar Asistencia	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 3
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una asistencia. El usuario podrá insertar, editar y eliminar una asistencia del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 18 Gestionar asistencia

Historia de Usuario	
Número: 17	**Usuario**: Super admin,

	Administrador
Nombre: Gestionar Salario	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 3
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un salario. El usuario podrá insertar, editar y eliminar un salario del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 19 Gestionar salario

Historia de Usuario	
Número: 18	**Usuario**: Super admin
Nombre: Gestionar Cuenta	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una cuenta. El usuario podrá insertar, editar y eliminar una cuenta del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 20 Gestionar cuenta

Historia de Usuario	
Número: 19	**Usuario**: Cajero
Nombre: Gestionar Venta	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Alto
Puntos estimados: 1	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una venta. El usuario podrá pagar propina y cerrar la venta del día	

Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.

Tabla 21 Gestionar venta

Historia de Usuario	
Número: 20	**Usuario**: Super Admin, Administrador
Nombre: Gestionar Transacción Económica	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Alto
Puntos estimados: 1	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un Transacción Económica. El usuario podrá insertar, editar y eliminar una Transacción Económica del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 22 Gestionar Transacción económica

Historia de Usuario	
Número: 21	**Usuario**:Super admin, Administrador
Nombre: Gestionar Mesas	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Alto
Puntos estimados: 1	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una mesa. El usuario podrá insertar, editar y eliminar una mesa del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 23 Gestionar Mesas

Historia de Usuario	
Número: 22	**Usuario**: Cajero

Nombre: Gestionar Pedidos	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Alto
Puntos estimados: 1	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un pedido. El usuario podrá insertar, editar y eliminar un pedido del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 24 Gestionar pedidos

Historia de Usuario	
Número: 23	**Usuario**: Super admin, Administrador
Nombre: Gestionar Menu	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Alto
Puntos estimados: 1	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un menu. El usuario podrá insertar, editar y eliminar un menu del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 25 Gestionar Mesas

Historia de Usuario	
Número: 24	**Usuario**:Super admin Administrador, Cajero
Nombre: Gestionar Cierre	
Prioridad en negocio: Alta	**Riesgo en desarrollo**: Alto
Puntos estimados: 1	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un cierre. El usuario podrá insertar un cierre al sistema.	

Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.

Tabla 26 Gestionar Cierre

Historia de Usuario	
Número: 25	**Usuario**: Super admin, Administrador
Nombre: Gestionar Reportes	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 2	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un reporte. El usuario podrá insertar, editar y eliminar un reporte del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 27 Gestionar Reportes

Historia de Usuario	
Número: 26	**Usuario**: Super admin, Administrador
Nombre: Gestionar Contadores	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar un contador. El usuario podrá insertar, editar y eliminar un contador del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 28Gestionar Contadores

Historia de Usuario	
Número: 27	**Usuario**: Super admin,

	Administrador
Nombre: Gestionar Lectura de Contadores	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Inicia cuando el usuario accede a esta funcionalidad para gestionar una lectura. El usuario podrá insertar, editar y eliminar una lectura del sistema.	
Observaciones: Se debe verificar que los datos introducidos en cada uno de los campos sean los correctos.	

Tabla 29 Gestionar Lectura de contadores

Historia de Usuario	
Número: 28	**Usuario**:Super admin Administrador
Nombre: Calcular Consumo Eléctrico	
Prioridad en negocio: Media	**Riesgo en desarrollo**: Medio
Puntos estimados: 1	**Iteración Asignada:** 4
Programador responsable: Malena Amaya Rodríguez	
Descripción: Calcula el consumo entre una lectura inicial y una actual	
Observaciones: Comprobar consumo contra valor establecido por la empresa eléctrica.	

Tabla 30 Gestionar Consumo eléctrico

Es importante descomponer cada una de las historias de usuario en tareas de ingeniería. Cada una de ellas se muestra a continuación:

No	Nombre HU	No	Tarea de Ingeniería
1	Diseño y creación de la base de datos.	1	Diseño de la base de datos
		2	Creación de la base de datos
2	Diseño de la interfaz de usuario	3	Interfaz de usuario
3	Autenticación del usuario	4	Autenticación del usuario
		5	Insertar usuario
		6	Modificar usuario

4	Gestionar usuario	7	Eliminar usuario
		8	Listar usuario
5	Gestionar almacén	9	Insertar almacén
		10	Modificar almacén
		11	Eliminar almacén
		12	Listar almacén
6	Gestionar compras	13	Insertar compras
		14	Modificar compras
		15	Eliminar compras
		16	Listar compras
7	Gestionar mermas	17	Insertar mermas
		18	Modificar mermas
		19	Eliminar mermas
		20	Listar mermas
8	Gestionar familia	21	Insertar familia
		22	Modificar familia
		23	Eliminar familia
		24	Insertar familia
		25	Listar familia
9	Gestionar unidad de medida	26	Insertar unidad de medida
		27	Modificar unidad de medida
		28	Eliminar unidad de medida
		29	Listar unidad de medida
10	Gestionar producto	30	Insertar producto
		31	Modificar producto
		32	Eliminar producto
		33	Listar producto
11	Gestionar Activo fijo	34	Insertar ACTIVO FIJO
		35	Modificar ACTIVO FIJO
		36	Eliminar ACTIVO FIJO
		37	Listar ACTIVO FIJO

12	Gestionar compra de ACTIVO FIJO	38	Insertar compra de ACTIVO FIJO
		39	Modificar compra de ACTIVO FIJO
		40	Listar compra de ACTIVO FIJO
13	Gestionar merma de ACTIVO FIJO	41	Insertar merma de ACTIVO FIJO
		42	Modificar merma de ACTIVO FIJO
		44	Listar merma de ACTIVO FIJO
14	Gestionar trabajadores	45	Insertar trabajadores
		46	Modificar trabajadores
		47	Eliminar trabajadores
		48	Listar trabajadores
15	Gestionar cargo	49	Insertar cargo
		50	Modificar cargo
		51	Eliminar cargo
		52	Listar cargo
16	Gestionar asistencia	53	Insertar asistencia
		54	Modificar asistencia
		55	Eliminar asistencia
		56	Listar asistencia
17	Gestionar salario	57	Insertar salario
		58	Modificar salario
		59	Eliminar salario
		60	Listar salario
18	Gestionar cuenta	61	Insertar cuenta
		62	Modificar cuenta
		63	Eliminar cuenta
		64	Listar cuenta
19	Gestionar venta	65	Cierre diario
		66	Pago de propina
20	Gestionar transacciones económicas	67	Insertar TE
		68	Modificar TE
		69	Eliminar TE

		70	Listar TE
21	Gestionar mesas	71	Insertar mesa
		72	Modificar mesa
		73	Eliminar mesa
		74	Listar mesa
22	Gestionar pedido	75	Insertar pedido
		76	Modificar pedido
		77	Eliminar pedido
		78	Listar pedido
23	Gestionar menú	79	Insertar menú
		80	Modificar menu
		81	Eliminar menú
		82	Listar menú
24	Gestionar cierre del día	83	Insertar cierre
		84	Eliminar cierre
		85	Modificar cierre
		86	Listar cierre
25	Gestionar reportes	87	Insertar reporte
		88	Modificar reporte
		89	Eliminar reporte
		90	Listar reporte
26	Gestionar contadores	91	Inserta contador
		92	Modificar contador
		93	Eliminar contador
		94	Listar contador
27	Gestionar lectura de contadores	95	Insertar Lectura de contadores
		96	Modificar Lectura de contadores
		97	Eliminar Lectura de contadores
		98	Listar Lectura de contadores
28	Calcular consumo eléctrico	99	Calcular consumo eléctrico

Tabla 31 Resumen de las TI

En las tablas que se muestran se relacionan algunas tareas de ingeniería que tenían mayor peso en el desarrollo de esta investigación

Tarea de Ingeniería	
Número de Tarea: 1	**Número de Historia de Usuario**: 1
Nombre: Diseño de la base de datos	
Tipo de tarea: Diseño	**Puntos estimados**: 1
Programador Responsable: Malena Amaya Rodríguez	
Descripción: Analizar profundamente el negocio para crear la base de datos que permita almacenar dicha información.	

Tabla 32 TI Diseño de la base de datos

Tarea de Ingeniería	
Número de Tarea: 2	**Número de Historia de Usuario**: 1
Nombre: Creación de la base de datos	
Tipo de tarea: Desarrollo	**Puntos estimados**: 1
Programador Responsable: Malena Amaya Rodríguez	
Descripción: Crear la base de datos con sus respectivas relaciones y la integridad correspondiente entre las tablas.	

Tabla 33 TI Creación de la base de datos

Tarea de Ingeniería	
Número de Tarea: 5	**Número de Historia de Usuario**: 4
Nombre: Insertar usuario	
Tipo de tarea: Desarrollo	**Puntos estimados**: 0.25
Programador Responsable: Malena Amaya Rodríguez	
Descripción:Se mostrará un formulario donde el usuario con los permisos necesarios podrá insertar un usuario y podrá asignarle el rol correspondiente y o permisos	

Tabla 34 TI Insertar usurario

Tarea de Ingeniería	
Número de Tarea: 6	**Número de Historia de Usuario**: 4

Nombre: Modificar usuario	
Tipo de tarea: Desarrollo	**Puntos estimados**: 0.25
Programador Responsable: Malena Amaya Rodríguez	
Descripción: Por cada usuario que se muestre en el listado de usuarios, habrá la opción de modificar, esta mostrará un formulario con los datos del usuario seleccionado donde se podrán modificar, todos los campos deben estar llenos.	

Tabla 35 TI Modificar usuario

Tarea de Ingeniería	
Número de Tarea: 7	**Número de Historia de Usuario**: 4
Nombre: Eliminar usuario	
Tipo de tarea: Desarrollo	**Puntos estimados**: 0.25
Programador Responsable: Malena Amaya Rodríguez	
Descripción: Por cada usuario que se muestre en el listado de usuarios, habrá la opción de eliminar, se eliminará de la base de datos los datos relacionados con ese usuario.	

Tabla 36 TI Eliminar usuario

Tarea de Ingeniería	
Número de Tarea: 22	**Número de Historia de Usuario:** 8
Nombre: Modificar familia	
Tipo de tarea: Desarrollo	**Puntos estimados**: 0.25
Programador Responsable: Malena Amaya Rodríguez	
Descripción: Se mostrará un formulario donde el usuario con los permisos necesarios podrá modificar los datos de una familia	

Tabla 37 TI Modificar familia

Tarea de Ingeniería	
Número de Tarea: 10	**Número de Historia de Usuario:** 5
Nombre: Modificar almacén	
Tipo de tarea: Desarrollo	**Puntos estimados**: 0.25
Programador Responsable: Malena Amaya Rodríguez	
Descripción: Se mostrará un formulario donde el usuario con los	

permisos necesarios podrá modificar un almacén

Tabla 38 TI modificar almacén

Tarea de Ingeniería	
Número de Tarea: 100	**Número de Historia de Usuario:** 28
Nombre: Calcular consumo eléctrico	
Tipo de tarea: Desarrollo	**Puntos estimados**: 0.4
Programador Responsable: Malena Amaya Rodríguez	
Descripción: Se calcula el consumo eléctrico mediante una formula	

Tabla 39 TI Calcular consumo eléctrico

2.4.1 Modelo físico de la Base de datos

En la ilustración se observa el modelo físico de la base de datos del software:

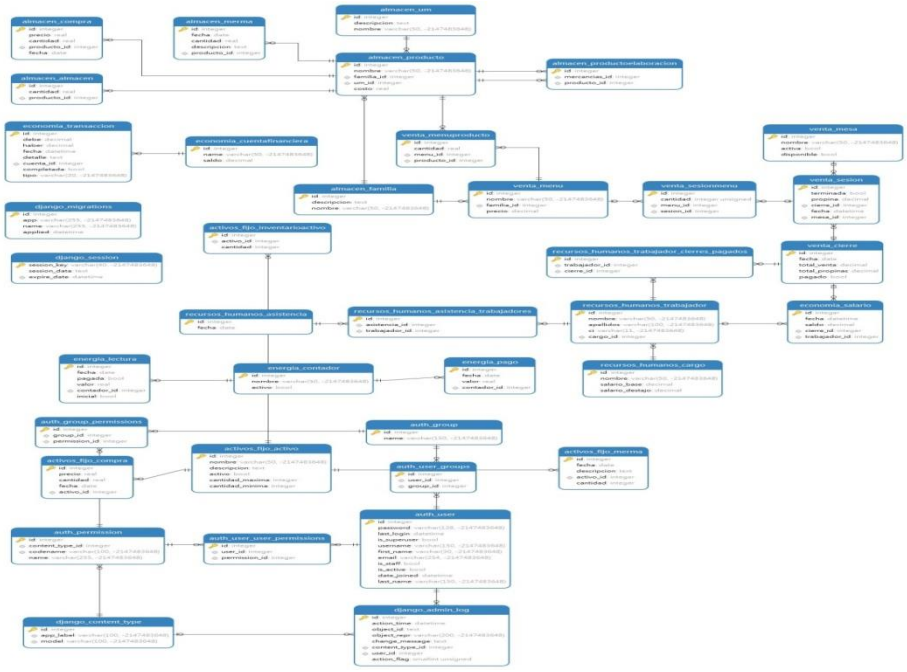

Ilustración 4 Modelo físico de la base de datos

2.4.2 Tarjetas de Clase, Responsabilidad y Colaboración

Se muestras algunas tarjetas de Clase, Responsabilidad y Colaboración (CRC) creadas

Usuario
Superclase:

Subclase:	
Descripción: En esta clase se guardan los elementos referidos a los usuarios.	
Atributos	
id usuario	Int
Usuario	String
Nombre	String
Apellido	String
Correo	String
Rol	String

Tabla 40 Tarjeta CRC usuarios

2.4.3Análisis del costo

La estimación es el proceso de medición anticipada de la duración, esfuerzos y costos necesarios para realizar todas las actividades y obtener todos los productos asociados a un proyecto. Es necesario tener en cuenta numerosos aspectos que afectan a la estimación como la complejidad del proyecto, su estructuración, el tamaño, los recursos involucrados y los riesgos asociados(Pressman, 2010)

La estimación del costo de un software es el proceso de predecir la cantidad de esfuerzo requerido para el desarrollo del sistema y el tiempo para ello. Existen diversos modelos para realizar la estimación del costo de un software como por ejemplo: COCOMO I, COCOMO II, Puntos de Función, Botton – Up, Top – Down, entre otros.

Método Puntos de Función

Es una métrica que permite traducir en un **número** el tamaño de la funcionalidad que brinda un producto de software desde el punto de vista del usuario, a través de una suma ponderada de las características del producto.

Componentes:

EI: Procesos en los que se introducen datos y que suponen la actualización de cualquier archivo interno.

EO: Procesos en los que se envía datos al exterior de la aplicación.

EQ: Procesos consistentes en la combinación de una entrada y una salida, en el que la entrada no produce ningún cambio en ningún archivo y la salida no contiene información derivada.
ILF: Grupos de datos relacionados entre sí internos al sistema.
EIF: Grupos de datos que se mantienen externamente.

Una vez obtenidos los diferentes elementos del sistema se utilizan las tablas reflejadas en el **Anexo 1**para asignar pesos en función del número de atributos que tengan y el número de archivos a los que afecte.

Componentes	Cantidad de Componentes por su Peso	Total
EI (Entradas)	23 * 3	69
EO (Salidas)	25*4	100
EQ (Consultas)	25 *4	100
FLI	26 *5	130
FLE	1*5	5

Cálculo de los Puntos de Función sin Ajustar (PFSA)

Los PFSA se calculan como la suma de los productos de cada componente por su peso determinado en la tabla correspondiente.

PFSA= 69+100+100+130+5 = 404

Cálculo de los Puntos de Función Ajustados (PFA)

PFA = PFSA * [0.65 + (0.01 * ACT)]

Ajuste de Complejidad Técnica (ACT)

Para calcular el ACT se le va dando un valor entre 0 y 5 a cada Factor de Ajuste como se muestra en el **Anexo 5.** Cuando cada Factor tenga un valor, se suman todos y así obtenemos el ACT. A continuación se muestra dicho procedimiento.

Cálculo del factor de ajuste.

PFA=PFSA* [0.65 + [0.01*ACT]]

PFA= 404*[0.65 + (0.01*29)]

PFA=404 *[0.65 + 0.29]

PFA= 404 *0.94

PFA= 379.7

Cálculo del Esfuerzo

Líneas de Código (LC)=PFA * Líneas por PF

LC= 379.7 * 20

Líneas de Código= 7594

Esfuerzo en horas / persona (E)

E = PFA / [1/8 persona- hora]

E = 379.7/ (1/8)

E = 379.7 *8

E = 3037 / 192 ctd de horas al mes.

E = 15 meses

Cálculo del Presupuesto del Proyecto

Costo Total del proyecto = sueldo de 1 participante * cantidad de participantes* Tiempo de desarrollo
Costo Total del proyecto = 400 * 1 * 15
Costo Total del proyecto = 6000

2.4.4 Análisis de los beneficios

Realizando un análisis del costo respecto a los beneficios descritos a lo largo de este trabajo se puede concluir que los aspectos positivos son superiores a los costos. Además, debe señalarse que para esta institución este software resultó gratuito puesto que es el resultado del trabajo de diploma del autor.

2.5 Conclusiones parciales del Capítulo

En este capítulo se plantean las etapas necesarias para desarrollar el software según la metodología XP, con la excepción de las pruebas. Se escriben las historias de usuario tareas iniciales que se agrupan en iteraciones y entregan. Se pudo concretar final de las iteraciones la entrega del sistema informático completamente en funcionamiento.

Capítulo 3: Validación de la solución propuesta

3.1 Introducción

En este capítulo se realizan las pruebas al software que permiten comprobar la calidad de este producto, lo que constituye uno de los pasos más importantes en el desarrollo de un sistema. No debe existir ninguna característica en el programa que no haya sido probada con la intención de mostrar un error no descubierto hasta entonces y con el fin de verificar la fiabilidad y calidad de la aplicación como un todo.

3.2 Pruebas al software

El proceso de pruebas es el instrumento más adecuado para determinar el status de la calidad de un producto. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos o si es el software que se quería desarrollar. En las pruebas se usan casos de prueba, especificados de forma estructurada mediante Técnicas de Prueba.

Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces. Los niveles de trabajo en los cuales se pueden realizar las pruebas son:

- Prueba de Unidad.
- Prueba de Integración.
- Prueba de Sistema.
- Prueba de Aceptación.
- Prueba de Seguridad.

3.2.1 Plan de pruebas

Según, el plan de pruebas de software se elabora con el fin de especificar qué elementos o componentes se van a probar para que el grupo de trabajo pueda realizar el proceso de Validación y Verificación de los requerimientos funcionales y no funcionales. Además, a través del plan de pruebas se puede continuar con la trazabilidad de los requerimientos, con lo cual el grupo de trabajo, identifica el porcentaje de avance que se ha logrado hasta cierto momento.(V. Loaiza, & Zorro, L. , 2010)

Al desarrollar el plan de pruebas, se puede obtener información sobre los errores, defectos o fallas que tiene el prototipo, así se realizan las correcciones pertinentes, según el caso y se asegura la calidad del producto que se está entregando al cliente.

A continuación se muestra en la **Tabla 40** el plan de pruebas

No	Nombre HU	Pruebas a realizar
1	Diseño y creación de la base de datos.	Test Diseño de la base de datos
		Test Creación de la base de datos
2	Diseño de la interfaz de usuario	Test Interfaz de usuario
3	Autenticación del usuario	Test Autenticación del usuario
4	Gestionar usuario	Test Insertar usuario
		Test Modificar usuario
		Test Eliminar usuario
		Test Listar usuario
5	Gestionar almacén	Test Insertar almacén
		Test Modificar almacén
		Test Eliminar almacén
		Test Listar almacén
6	Gestionar compras	Test Insertar compras
		Test Modificar compras
		Test Eliminar compras
		Test Listar compras
7	Gestionar mermas	Test Insertar mermas
		Test Modificar mermas
		Test Eliminar mermas
		Test Listar mermas
8	Gestionar familia	Test Insertar familia
		Test Modificar familia
		Test Eliminar familia
		Test Insertar familia
		Test Listar familia
		Test Insertar unidad de medida

9	Gestionar unidad de medida	Test Modificar unidad de medida
		Test Eliminar unidad de medida
		Test Listar unidad de medida
10	Gestionar producto	Test Insertar producto
		Test Modificar producto
		Test Eliminar producto
		Test Listar producto
11	Gestionar Activo fijo	Test Insertar ACTIVO FIJO
		Test Modificar ACTIVO FIJO
		Test Eliminar ACTIVO FIJO
		Test Listar ACTIVO FIJO
12	Gestionar compra de ACTIVO FIJO	Test Insertar compra de ACTIVO FIJO
		Test Modificar compra de ACTIVO FIJO
		Test Listar compra de ACTIVO FIJO
13	Gestionar merma de ACTIVO FIJO	Test Insertar merma de ACTIVO FIJO
		Test Modificar merma de ACTIVO FIJO
		Test Listar merma de ACTIVO FIJO
14	Gestionar trabajadores	Test Insertar trabajadores
		Test Modificar trabajadores
		Test Eliminar trabajadores
		Test Listar trabajadores
15	Gestionar cargo	Test Insertar cargo
		Test Modificar cargo
		Test Eliminar cargo

		Test Listar cargo
16	Gestionar asistencia	Test Insertar asistencia
		Test Modificar asistencia
		Test Eliminar asistencia
		Test Listar asistencia
17	Gestionar salario	Test Insertar salario
		Test Modificar salario
		Test Eliminar salario
		Test Listar salario
18	Gestionar cuenta	Test Insertar cuenta
		Test Modificar cuenta
		Test Eliminar cuenta
		Test Listar cuenta
19	Gestionar venta	Test Cierre diario
		Test Pago de propina
20	Gestionar transacciones económicas	Test Insertar TE
		Test Modificar TE
		Test Eliminar TE
		Test Listar TE
21	Gestionar mesas	Test Insertar mesa
		Test Modificar mesa
		Test Eliminar mesa
		Test Listar mesa
22	Gestionar pedido	Test Insertar pedido
		Test Modificar pedido
		Test Eliminar pedido
		Test Listar pedido
23	Gestionar menu	Test Insertar menu
		Test Modificar menu
		Test Eliminar menu
		Test Listar menu
24	Gestionar cierre del día	Test Insertar

25	Gestionar reportes	Test Insertar reporte
		Test Modificar reporte
		Test Eliminar reporte
		Test Listar reporte
26	Gestionar contadores	Test Inserta contador
		Test Modificar contador
		Test Eliminar contador
		Test Listar contador
27	Gestionar lectura de contadores	Test Insertar Lectura de contadores
		Test Modificar Lectura de contadores
		Test Eliminar Lectura de contadores
		Test Listar Lectura de contadores
28	Calcular consumo eléctrico	Test Calcular consumo eléctrico

Tabla 41 Plan de pruebas

3.2.2 Pruebas de aceptación

Uno de los pilares de la metodología XP es el uso de pruebas para comprobar el funcionamiento de los códigos que se vayan implementando. Las pruebas de aceptación son creadas en base a las historias de usuarios, en cada ciclo de la iteración del desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente implementada. Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. Asimismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una historia de usuario no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación. Dado que la responsabilidad es grupal, es recomendable publicar los resultados de las pruebas de aceptación, de manera que todo el equipo esté al tanto de esta información. (V. Loaiza & Zorro, 2010)

Caso de Prueba	
Número: 1	**Historia de usuario**: 1

Nombre de caso de prueba: Test Base de Datos
Descripción: Verifica el funcionamiento de la base de datos.
Condición de ejecución: Estar conectado a la base de datos.
Entradas: Valores para leer o escribir en la base de datos por ejemplo: usuario, contraseña.
Resultado esperado: Se muestran o guardan los datos correctamente.
Evaluación: Prueba Satisfactoria.

Tabla 42 PA Test base de datos

Caso de Prueba	
Número: 3	**Historia de usuario**: 3
Nombre de caso de prueba: Test Autenticación del usuario	
Descripción: Se inserta el nombre de usuario y la contraseña para entrar al sistema. Se insertarán de forma incorrecta, dejando campos en blanco para verificar la validación. Luego se insertarán los datos de manera correcta para comprobar esta funcionalidad.	
Condición de ejecución: Que el software este en ejecución	
Entradas: Insertar datos:nombre de usuario y contraseña	
Resultado esperado: El sistema debe alertar al usuario cuando se inserten datos erróneos o cuando se dejen campos en blanco. Cuando se inserten los datos correctamente, según el rol que tenga el usuario, se le otorgan los permisos correspondientes.	
Evaluación: Prueba Satisfactoria.	

Tabla 43 PA Autenticación del usurario

Caso de Prueba	
Número: 14	**Historia de usuario**: 14
Nombre de caso de prueba: Test Gestionar trabajadores	
Descripción: Se inserta el nombre del trabajador para crear un nuevo trabajador, se inserta de forma incorrecta dejando el campo en blanco, luego se inserta de forma correcta para verificar que los datos sean almacenados. Se modifican los datos del trabajador, se modifican incorrectamente dejando campos en blanco para verificar la validación, se modifican correctamente para verificar que los datos sean	

almacenados. Se elimina el trabajador aceptando el mensaje de confirmación.
Condición de ejecución: Estar conectado a la base de datos.
Entradas: Datos del trabajador
Resultado esperado: El sistema debe alertar al usuario cuando se insertan datos erróneos. Cuando se insertan los datos correctamente el sistema debe almacenarlos en la base de datos y mostrarlos.
Evaluación: Prueba Satisfactoria.

Tabla 44 PA Gestionar trabajadores

En la **Tabla 44** se muestran las clases de equivalencia de las pruebas de aceptación al sistema en la HU: Gestionar Trabajadores

Condición de entrada	Clases válidas	Representante	Clases inválidas	Representante
Nombre	1. Cualquier cadena de caracteres alfabéticos que comiencen con mayúscula. Permite espacio.	Juan	2. Cadena que contenga caracteres numéricos	Ju6n
			3.Cadena que no comience con mayúscula	Juan
			4-Cadena vacía.	NULL
Apellidos	5 Cualquier cadena de caracteres alfabéticos que comiencen con mayúscula. Permite espacio.		6. Cadena que contenga caracteres numéricos	Padr0on

		Padrón	7. Cadena que no comience con mayúscula	Padron
			8. Cadena vacía.	NULL
Carnet Identidad	9. Cadena numérica de 11 dígitos que no inicie con símbolos, que no contenga caracteres alfabéticos, donde los dígitos correspondientes al mes sean menor o igual que 12 y los correspondientes al año sean menor o igual que 31.	95072431452	10. Cadena que contenga caracteres alfabéticos	950724**g**3145
			11. Cadena numérica mayor de 11 dígitos	950724314529
			12.Cadena numérica menor de 11 dígitos	95072431
			13. Cadena que inicie con símbolos	-95072431452
			14. Cadena donde los dígitos correspondientes al mes sean mayores que 12	95**14**2431452
			15. Cadena donde los dígitos correspondientes al dia sean mayores que 31	9507**32**31452
			16. Cadena vacía	NULL

Cargo	17 .seleccionar algunas de las opciones	administrador	18. no seleccionar ninguna de las opciones	NULL

Tabla 45Clases de equivalencia gestionar trabajadores

En la **Tabla 45** se muestra un resumen de algunos casos de pruebas necesarios para realizar las pruebas de aceptación.

Resumen de Casos de prueba

No	Clases equiv.	Nombre	Apellidos	CI	Cargo	Resultado
1	1,5,9,17	Juan	Padrón	95072431452	Administrador	Trabajador Insertado
2	1,5,9,18	Juan	Padrón	95072431452	NULL	Inserte un cargo
3	1,5,10,17	Juan	Padrón	9507d431452	Administrador	Escriba solo números
4	1,5,11,17	Juan	Padrón	950724314524	Administrador	La cadena tiene que ser igual a 11 dígitos
5	1,5,12,17	Juan	Padrón	9507243145	Administrador	La cadena tiene que ser igual a 11 dígitos
6	1,5,13,17	Juan	Padrón	-95072431452	Administrador	La cadena tiene q empezar con dígitos
7	1,5,14,17	Juan	Padrón	95132431452	Administrador	Los dígitos correspondientes al mes tienen que ser menor o

						igual que 12
8	1,5,15,17	Juan	Padrón	95073231452	Administrador	Los dígitos correspondientes al día tienen que ser menor o igual a31
9	1,5,16,17	Juan	Padrón	NULL	Administrador	Inserte CI
10	1,5,10,18	Juan	Padrón	9507d231452l	NULL	Escriba solo números e inserte un cargo
11	1,5,11,18	Juan	Padrón	950724314523	NULL	La cadena tiene que ser igual a 11 dígitos e inserte un cargo
12	1,5,12,18	Juan	Padrón	9507243145	NULL	La cadena tiene que ser igual a 11 dígitos e inserte un cargo
13	1,5,13,18	Juan	Padrón	-95072431452	NULL	La cadena tiene que empezar con dígitos e inserte un cargo
14	1,5,14,18	Juan	Padrón	95132431452	NULL	Los digito correspondientes al mes

						tienen que ser menor o igual que 12 e inserte un cargo
15	1,5,15,18	Juan	Padrón	95073231452	NULL	Los dígitos correspondientes al día tienen que ser igual o menor que 31 e inserte un cargo
16	1,5,16,18	Juan	Padrón	NULL	NULL	Inserte CI, Inserte un cargo

Tabla 46 resumen de casos de prueba

A continuación se muestran algunas Tablas de Prueba por Caso de Prueba

Test insertar trabajador

Tabla de Prueba	
No. Caso de Prueba	1
Requerimiento	Estar conectado a la base de datos.
Objetivo	Probar la acción de insertar un equipo en el sistema (Para cubrir las clases válidas 1,5,9,17)
Tipo de Prueba	Funcional.
Hardware	Sistema de cómputo con un Procesador Intel Pentium - Disco Duro de 1T - Memoria RAM de 4GB.
Software	Sistema Operativo Windows 7 o versiones posteriores – Base de Datos SQLite – Navegador de internet Mozilla Firefox 3.5.

Personal	Encargados de Pruebas	
Caso de Prueba		
Datos de Entrada	Nombre: Juan, Apellido: Padrón, CI: 95072431452, Cargo Administrador.	
Resultados Esperados	Trabajador insertado correctamente.	
Resultados Obtenidos	SI(X) NO()	
Casos de Excepción		
Aprobado por: Dayana Olivia Hernández Revilla	**Cago:** Jefe del Proyecto, Tester	**Líder**: Malena Amaya

Tabla 47Test Insertar trabajador. Caso de preba1

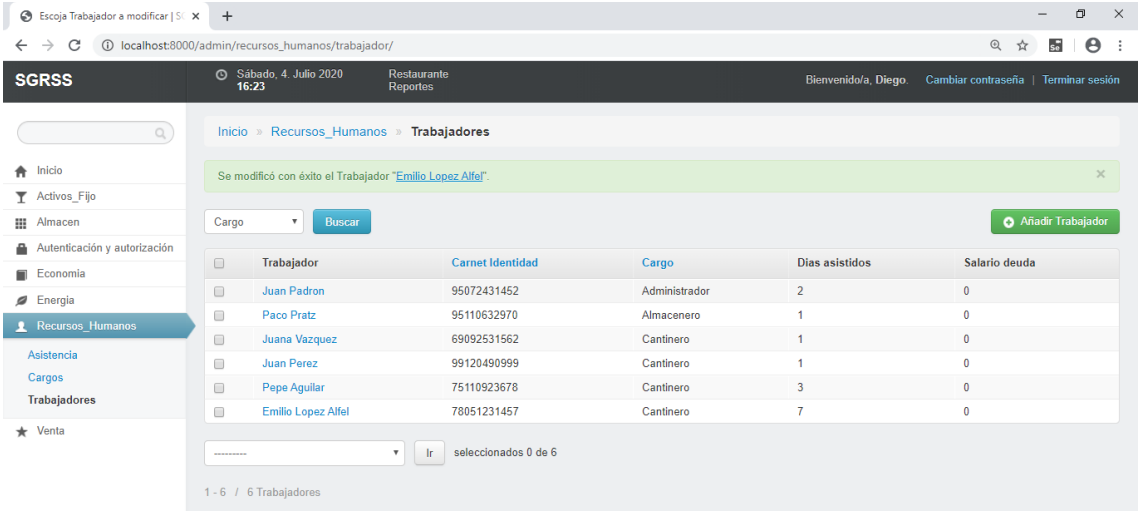

Ilustración 5 Interfaz trabajadores

Tabla de Prueba	
No. Caso de Prueba	2
Requerimiento	Estar conectado a la base de datos.
Objetivo	Probar la acción de insertar un equipo en el sistema (Para cubrir las clases1,5,9,18)
Tipo de Prueba	Funcional.
Hardware	Sistema de cómputo con un Procesador Intel Pentium - Disco Duro de 1T - Memoria RAM de 4GB.
Software	Sistema Operativo Windows 7 o versiones

	posteriores – Base de Datos SQlite – Navegador de internet Mozilla Firefox 3.5.	
Personal	Encargados de Pruebas	
Caso de Prueba		
Datos de Entrada	Nombre: Juan, Apellido: Padrón, CI:950731452 Cargo: Null	
Resultados Esperados	Inserte cargo	
Resultados Obtenidos	SI(X) NO()	
Casos de Excepción		
Aprobado por: Dayana Olivia Hernández Revilla	**Cago:** Jefe del Proyecto, Tester	**Líder**: Malena Amaya

Tabla 48 Test Insertar trabajador. Caso de preba2

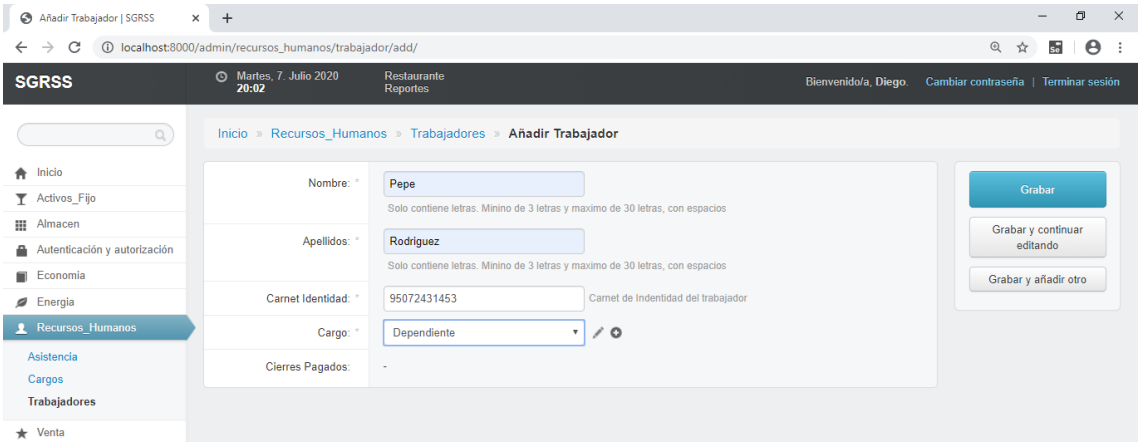

Ilustración 6 Insertar trabajador

3.2.3Pruebas de unidad

```python
class ProductoTestCase(TestCase):

    def test_insert_producto_test(self):
        Familia.objects.create(nombre='Familia de Prueba 1')
        UM.objects.create(nombre='UM de Prueba 1')

        familia = Familia.objects.get(nombre='Familia de Prueba 1')
        um = UM.objects.get(nombre='UM de Prueba 1')
        Producto.objects.create(nombre="producto de prueba",
                                um=um,
                                familia=familia)
        producto = Producto.objects.get(nombre="producto de prueba")
        self.assertIsNotNone(producto, 'No se inserto el producto de prueba')

    def test_modifed_producto_test(self):
        Familia.objects.create(nombre='Familia de Prueba 1')
        UM.objects.create(nombre='UM de Prueba 1')

        familia = Familia.objects.get(nombre='Familia de Prueba 1')
        um = UM.objects.get(nombre='UM de Prueba 1')
        Producto.objects.create(nombre="producto de prueba",
                                um=um,
                                familia=familia)
        producto = Producto.objects.get(nombre="producto de prueba")
        producto.nombre = "producto modificado"
        producto.save()
        modificado = Producto.objects.get(nombre='producto modificado')
        self.assertIsNotNone(modificado, 'No se modifico el producto de prueba')
```

Ilustración 7 PU de Producto

```python
from django.test import TestCase
from almacen.models import  UM


class UMTestCase(TestCase):

    def test_insert_um_test(self):
        UM.objects.create(nombre='g')
        UM.objects.create(nombre='kg')

        prueba1 = UM.objects.get(nombre='g')
        prueba2 = UM.objects.get(nombre='kg')
        self.assertIsNotNone(prueba1, "No existe la UM de prueba 1")
        self.assertIsNotNone(prueba2, "No existe la UM de prueba 1")

    def test_update_um_test(self):
        UM.objects.create(nombre='UM de Prueba 1')
        prueba1 = UM.objects.get(nombre='UM de Prueba 1')
        prueba1.nombre = "Prueba Modificada"
        prueba1.save()
        prueba_modificada = UM.objects.get(nombre="Prueba Modificada")
        self.assertIsNotNone(prueba_modificada, "Prueba Modificada")

    def test_delete_um_test(self):
        UM.objects.create(nombre='UM de Prueba 1')
        prueba1 = UM.objects.get(nombre='UM de Prueba 1')
        prueba1.delete()
        prueba_eliminada = UM.objects.filter(nombre="UM de Prueba 1").first()
        self.assertIsNone(prueba_eliminada, "No se elimino la prueba")
```

Ilustración 8 PU de unidad de medida

```python
from django.test import TestCase
from almacen.models import Familia


class FamiliaTestCase(TestCase):

    def test_insert_families_test(self):
        Familia.objects.create(nombre='Familia de Prueba 1')
        Familia.objects.create(nombre='Familia de Prueba 2')

        prueba1 = Familia.objects.get(nombre='Familia de Prueba 1')
        prueba2 = Familia.objects.get(nombre='Familia de Prueba 2')
        self.assertIsNotNone(prueba1,"No existe la familia de prueba 1")
        self.assertIsNotNone(prueba2,"No existe la familia de prueba 2")

    def test_update_families_test(self):
        Familia.objects.create(nombre='Familia de Prueba 1')
        prueba1 = Familia.objects.get(nombre='Familia de Prueba 1')
        prueba1.nombre = "Prueba Modificada"
        prueba1.save()
        prueba_modificada = Familia.objects.get(nombre="Prueba Modificada")
        self.assertIsNotNone(prueba_modificada, "Prueba Modificada")

    def test_delete_families_test(self):
        Familia.objects.create(nombre='Familia de Prueba 1')
        prueba1 = Familia.objects.get(nombre='Familia de Prueba 1')
        prueba1.delete()
        prueba_eliminada = Familia.objects.filter(nombre="Familia de Prueba 1").first()
        self.assertIsNone(prueba_eliminada, "No se elimino la prueba")
```

Ilustración 9 PU de Familia

```
Creating test database for alias 'default'...
System check identified no issues (0 silenced).
........
----------------------------------------------------------------------
Ran 8 tests in 0.018s

OK
Destroying test database for alias 'default'...
```

Ilustración 10 Resultados de las Pruebas de Unidad

3.3 Análisis de los resultados obtenidos

Después de desarrollar todo un proceso de pruebas con un nivel medio de sencillez se lograron resultados satisfactorios, pues tras la detección de diferentes errores obtenidos fundamentalmente con las pruebas realizadas, se solucionaron varios problemas que impedían el cumplimiento de los requisitos fundamentales del sistema en cuestión.

Las primeras pruebas fueron planeadas y ejecutadas en módulos individuales del programa y a medida que fueron avanzando se desplazaron a módulos integrados, hasta que finalmente llegaron al sistema completo y se logró

obtener un software cuyas funciones se encuentra en correspondencia con las especificaciones acordadas y que además cumple con los requerimientos de rendimiento. El desarrollo del sistema cumple las expectativas trazadas al inicio del proyecto y satisface al cliente en su totalidad.

3.4 Conclusiones parciales

Las pruebas realizadas permitieron validar el funcionamiento del software y los resultados satisfactorios de dichas pruebas. Una vez realizadas las pruebas se logró brindarle al cliente una versión del software que facilitara la gestión de la información en el Restaurante San Severino.

Conclusiones Generales

1. Se elaboró un mapa de procesos asociado al campo de investigación a partir del análisis de los trabajos más importantes relacionados con la gestión de la información del Restaurante San Severino.
2. Se utilizó la metodología XP para el desarrollo del software que se presenta, así como tecnologías de avanzada para el diseño del mismo.
3. Se implementa la solución utilizando el lenguaje de programación Python que permite agilizar los procesos.

Mediante la aplicación de pruebas al software resultó posible obtener resultados favorables, con el consiguiente análisis de errores detectados que fueron subsanados como parte del desarrollo de este software

Recomendaciones

Desde el punto de vista del alcance del presente trabajo y teniendo en cuenta el momento de desarrollo del mismo, se proponen las siguientes recomendaciones:

- Implantar esta aplicación en el resto de restaurantes de la ciudad de Matanzas y luego en el país

-Agregarle módulos que enriquezcan la información de la aplicación.

- Hacerle una vista de cara a internet

para poder reservar online

Bibliografía

Álvarez, M. Á. (2007). DesarrolloWeb. from http://desarrolloweb.com

Álvarez, R. (Producer). (2012). Introducción al HTML. Retrieved from

http://www.desarrolloweb.com/articulos/arquitectura-cliente-servidor.html.
Cristian. (2008). Ventajas de usar CSS. from

http://www.stardustxs.com/2008/03/05/ventajas-de-usar-css/

Fuentes, J. (2015). Desarrollo de Softwre Ágil. Extremme Programming y Scrum. .

Fuentes, J. R. L. (2015). Desarrollo de Softwre Ágil. Extremme Programming y Scrum.
Joskowicz, I. (2012). Reglas y Prácticas en Exreme Programming.

Joskowicz, I. J. (2012). Reglas y Prácticas en Exreme Programming.

Loaiza, V., & Zorro, L. . (2010). Plan de Pruebas de Software.

. Loaiza, V., & Zorro, L. (2010). Plan de Pruebas de Software.
Pressman. (2010). Ingeniería del software.Un enfoque práctico.

RocketTheme. (2009). Metodologías de desarrollo de software.

Solanas, A., & Sierra, V. Bootstrap: fundamentos e introducción a sus aplicaciones.
Tedeschi, N. (2013). ¿Qué es un Patrón de Diseño?

Anexos

Anexo 1 de ponderaciones para EI, EQ, EO, ILF, EIF

CLASIFICACION DE ENTRADAS Y CONSULTAS	1-4 Atributos	5-15 Atributos	Más de 15 Atributos
0 o 1 ficheros accedidos	BAJA 3	BAJA 3	MEDIA 4
2 ficheros accedidos	BAJA 3	MEDIA 4	ALTA 6
Más de 2 ficheros accedidos	MEDIA 4	ALTA 6	ALTA 6

CLASIFICACION DE SALIDAS	1-5 Atributos	6-19 Atributos	Más de 19 Atributos
0 o 1 ficheros accedidos	BAJA 4	BAJA 4	MEDIA 5
2 o 3 ficheros accedidos	BAJA 4	MEDIA 5	ALTA 7
Más de 3 ficheros accedidos	MEDIA 5	ALTA 7	ALTA 7

FICHEROS LÓGICOS INTERNOS	1-19 Atributos	20-50 Atributos	Más de 50 Atributos
1 Entidad o registro lógico	BAJA 7	BAJA 7	MEDIA 10
2 - 5 Entidades o registros lógicos	BAJA 7	MEDIA 10	ALTA 15
Más de 5 Entidades o registros lógicos	MEDIA 10	ALTA 15	ALTA 15

FICHEROS LÓGICOS EXTERNOS	1-19 Atributos	20-50 Atributos	Más de 50 Atributos
1 Entidad o registro lógico	BAJA 5	BAJA 5	MEDIA 7
2 - 5 Entidades o registros lógicos	BAJA 5	MEDIA 7	ALTA 10
Más de 5 Entidades o registros lógicos	MEDIA 7	ALTA 10	ALTA 10

Anexo 2 Cálculo del Factor de Ajuste

№	Nombre del Factor	Valor
1	Comunicación de datos	2
2	Procesamiento distribuido	2
3	Rendimiento	3
4	Configuración de Explotación Compartida	2
5	Tasa de transacciones	1
6	Entrada de Datos en Línea	2
7	Eficiencia con el Usuario Final	4
8	Actualizaciones en Línea	0
9	Procesamiento complejo	2
10	Reusabilidad del Código	1
11	Facilidad de implementación	2
12	Facilidad de Operación	4
13	Instalaciones Múltiples	2
14	Facilidad de Cambios	2
	Ajuste de Complejidad Técnica(ACT)	29

Anexo 3 Tabla para obtener las Líneas por Puntos de Función

Entorno y Lenguaje	Líneas de Código por PF	Horas por PF	Entorno y Lenguaje
Lenguajes 2GL: Ensamblador, C,…	300	20 a 30	**Lenguajes 2GL: Ensamblador, C,…**
Lenguajes 3GL: Cobol	100	10 a 20	**Lenguajes 3GL: Cobol**
Lenguajes 4GL: VisualXX	20	5 a 10	**Lenguajes 4GL: VisualXX**