UNIVERSIDAD DE MATANZAS

Facultad de Ciencias Técnicas

Departamento de Informática

Trabajo de diploma para optar el título de Ingeniero Informático

Aplicación web para la gestión de dietas de la Universidad de Matanzas.

Autor: Anailyn Vera Llerena.

Tutor: DrC. Walfredo González Hernández.

Ing. Raúl Suárez Hernández.

Matanzas, Cuba 2018

“La educación es el arma más poderosa que puedes usar para cambiar el mundo”

Nelson Mandela.

A mis padres, todo por ustedes.

Agradecimientos

- *A mis padres por su eterno sacrificio y ejemplo permanente; por su confianza y apoyo incondicional.*
- *A mi esposo por su confianza y paciencia durante todos estos años.*
- *A mi familia por darme fuerza e impulso para seguir adelante.*
- *A todos mis amigos de la universidad por hacer de estos años una experiencia inolvidable.*
- *A mis tutores Walfredo y Raúl, por su sabiduría, paciencia y apoyo total.*
- *A todos mis amigos y compañeros de trabajo que me ayudaron e incitaron a seguir adelante, en especial a Johanna.*
- *A todas aquellas personas que alguna vez pregunto "cómo va la tesis."*

A todos muchas gracias

Declaración de autoría

Yo, Anailyn Vera Llerena, declaro que soy la única autora de este trabajo y autorizo a la Universidad de Matanzas, y en especial, a la Facultad de Ciencias Económicas e Informática, a que hagan el uso que estimen pertinente de él.

Para que así conste, firmo la presente a los _____ días del mes de junio del 2018.

_____	_____
Firma del autor	Firma del tutor
Anailyn Vera Llerena	Dr. C. Walfredo González Hernández

Resumen

Las dietas laborales son las compensaciones económicas que satisfacen las empresas a sus trabajadores cuando éstos se deben desplazar por motivo de trabajo. No forman parte de su salario, puesto que su objetivo es cubrir aquellos costes en los que los trabajadores incurren de forma adicional, y que se entiende que no deben soportar ellos pues están desarrollando su actividad laboral. En la actualidad, que las tecnologías de la información están presentes cada día más en la vida diaria de la sociedad, la Universidad de Matanzas decidió perfeccionar el Sistema de Gestión de Dietas Laborales en dicha entidad, agilizando el procesamiento, gestión y análisis de la documentación de las dietas laborales pertenecientes a sus trabajadores, facilitando la elaboración, modificación y documentación de las mismas, para así llevar a cabo esta actividad de manera eficiente. Para este sistema se usó la metodología SCRUM para el diseño e ingeniería, Apache como servidor web, *MySQL* como gestor de base de datos, PHP *(Hypertext Preprocessor*) como lenguaje de lado del servidor con el *framework Symfony* y HTML 5 como lenguaje del lado cliente.

Summary

The labor allowances are the economic compensations that companies pay their workers when they must move for work. They do not form part of their salary since their objective is to cover those costs in which workers incur additionally, and that is understood that they should not support them since they are developing their work activity. At present, that information technologies are present more and more in the daily life of society, the University of Matanzas decided to improve the Labor Diet Management System in this entity, streamlining the processing, management and analysis of documentation of the labor allowances belonging to their workers, facilitating the elaboration, modification and documentation of the same, in order to carry out this activity efficiently. For this system we used the SCRUM methodology for design and engineering, Apache as a web server, MySQL as a database manager, PHP (Hypertext Preprocessor) as a server-side language with Symfony framework and HTML 5 as a client-side language.

Índice

Introducción 1
Capítulo I: Fundamentación teórica 5
Introducción 5
1.1 Estudio del estado del arte 5
1.1.1 ASSETS 5
1.1.2 Versat Sarasola 6
1.2 Flujo actual de los procesos más importantes involucrados en el campo de acción. 7
1.2.1 Proceso de solicitud de dieta 7
1.2.2 Proceso de liquidación de dieta 9
1.3 Características de ambiente y herramientas 9
1.3.1 Metodología de Desarrollo 10
1.3.2 Lenguaje de programación del lado del cliente. 12
1.3.2.1 Lenguaje de Marcas de Hipertexto 5 (HTML5). 12
1.3.2.2 CSS (Cascading Style Sheets). 13
1.3.2.3 JavaScript. 13
1.3.3 Lenguaje de programación del lado del servidor. 13
1.3.3.1 PHP 13
1.3.4 Patrón de Arquitectura 14
1.3.5 Plataforma y herramientas de desarrollo 14
1.3.5.1 Framework Symfony 14
1.3.5.2 Framework Bootstrap 16
1.3.5.3 MySQL 17
1.3.5.4 PhpStorm 17
1.3.5.5 Apache 18
1.3.5.6 Herramientas CASE 18
Conclusiones parciales del Capítulo I: 19
Capítulo II: Elaboración y diseño del sistema 21
Introducción 21
2.1 Descripción de la propuesta de solución 21
2.2 Planificación inicial 21
2.2.1 Equipo de trabajo y roles 22
2.2.2 Historias de Usuario 23
2.2.3 Historias Técnicas 25

2.2.3.1 Apariencia o interfaz externa: 25
2.2.3.2 Usabilidad: 25
2.2.3.3 Rendimiento 25
2.2.3.4 Seguridad. 25
2.2.3.5 Confiabilidad. 25
2.2.3.6 Portabilidad. 26
2.2.4 Pila del producto 26
2.2.5 La pila del producto en forma de Historias de Usuarios. 28
2.2.6 Planificación del Sprint 31
2.2.6.1 Sprint 1 31
2.2.6.2 Sprint 2 33
2.2.6.3 Sprint 3 34
2.2.6.4 Sprint 4 35
2.2.6.5 Sprint 5 36
2.2.7 Modelo de datos 37
2.2.8 Análisis de factibilidad 37
2.2.9 Beneficios tangibles e intangibles. 38
Conclusiones parciales del Capítulo II 39
Capítulo III: Implementación y validación de la solución propuesta 40
Introducción 40
3.1 Pruebas 40
3.1.1 Pruebas de Aceptación 40
3.2 Prueba Automatizadas 47
3.2.1 Subgraph Vega 47
3.2.1.1 Resultado de una prueba realizada por Subgraph Vega 48
Conclusiones Generales ..... **¡Error! Marcador no definido.**
Bibliografía ..... **¡Error! Marcador no definido.**

Introducción

Los gastos por dietas son aquellos en los que incurren los cuadros, directivos, funcionarios y trabajadores cuando son enviados eventual o temporalmente a realizar labores que implican una variación de su lugar habitual de alimentación y alojamiento. Estos gastos comprenden los de alimentación (desayuno, almuerzo y comida) y alojamiento.

Las dietas se entregan por anticipado o se pueden pagar por las entidades directamente a las instalaciones que prestan los servicios de alojamiento y gastronomía, utilizando los instrumentos de pago vigentes. Estas se calculan tomando como base los días de duración del trabajo a realizar y el importe de la dieta diaria establecida por la resolución vigente. A los trabajadores que tienen autorizado estipendio de alimentación, el importe que reciben por ese concepto se les deduce del monto de la dieta que corresponda a la alimentación o mediante el mecanismo de descuento que se aplique, por cada día laborable solicitado como anticipo (Precio, 2014).

En la actualidad en la universidad de Matanzas se realiza la gestión del pago de las dietas siguiendo lo estipulado en la Resolución No. 267 del Ministerio de Finanzas y Precios del año 2014, planteado anteriormente.

Este proceso comienza con la asignación de las dietas por parte de las personas autorizadas en el centro de costo correspondiente al trabajador al cual se le será asignada. Posteriormente el trabajador debe presentarse en el Departamento Contable para la tramitación de su dieta y a continuación ir a la caja a cobrar el efectivo que la misma representa. Una vez terminada la actividad o evento por el cual se le fue otorgada la dieta, el trabajador tiene el deber de personarse nuevamente en el Departamento Contable para su liquidación o su reembolso en caso de no haberla consumido totalmente.

Del mismo modo, si se diera el caso de presentársele algunos gastos adicionales por motivos imprevistos, de acuerdo a lo establecido, se comprueban y se procede a sufragar los gastos correspondientes, lo cual se realiza a través de los pagos menores.

En estos momentos la gestión de tramitación es realizada con ayuda del sistema informático *Assets*, el cual permite entregar una dieta, liquidarla o reembolsarla según sea el caso, además de gestionar los pagos menores que correspondan con los gastos del trabajador por motivos laborales que no son previsibles anticipadamente.

Pero, aunque este sistema cumple con algunos de los requerimientos del proceso de gestión de dietas, no satisface todas las necesidades que requiere, lo que dificulta en gran

medida la eficiencia del mismo y además introduce errores humanos en algunas actividades que se realizan manualmente.

Por ejemplo, en la actualidad no permite llevar el control del presupuesto de dietas y pagos menores por Áreas, como tampoco el control de anticipos a justificar, la información de indicadores de dietas, ni se permite generar ningún tipo de reporte o informe, por lo que hay que hacerlo de forma manual. Todo ello provoca la aparición de errores e imprecisiones que ralentizan y dificultan las operaciones de cierre de mes en el área contable, lo que además acarrea gastos innecesarios en material de oficina.

Dada la situación problemática anterior, en aras de las nuevas líneas del desarrollo del departamento Contable y puesto que la aplicación informática existente no solventa una gran parte de los requerimientos del proceso de gestión de dietas se platea como **problema científico** la siguiente interrogante: ¿Cómo contribuir al perfeccionamiento del proceso de gestión de dietas de la Universidad de Matanzas?

La investigación tendrá como **objeto de estudio** el proceso de gestión de dietas en las instituciones y universidades cubanas y el **campo de acción** la automatización del proceso de gestión de dieta en la Universidad de Matanzas.

Se trazó como **objetivo general**, desarrollar una aplicación web que contribuya a perfeccionar la gestión de dietas en la Universidad de Matanzas, y como **objetivos específicos** para cumplir el anterior:

1. Investigar los referentes teóricos y determinar herramientas y metodologías más adecuadas para el desarrollo de una Aplicación web que perfeccione el proceso de gestión de dietas en la Universidad de Matanzas.
2. Implementar una Aplicación web que perfeccione y facilite el proceso de gestión de dietas.
3. Probar la Aplicación web para comprobar que cumple con todos los requerimientos establecidos por el cliente.
4. Validar la Aplicación web mediante pruebas de aceptación.

Las tareas que dirigieron la investigación para cumplir los objetivos trazados fueron:

1. Revisión bibliográfica sobre el proceso de gestión de dietas vigente para identificar las actividades y flujos automatizables.
2. Revisión bibliográfica sobre tecnologías y tendencias actuales para la automatización y mejora de este proceso en otras instituciones del país.

3. Levantamiento de historias de usuario de la aplicación
4. Diseño de la aplicación Web.
5. Implementación de la aplicación web.
6. Planificación y ejecución de las pruebas de aceptación del producto.

Todo lo anterior defiende la **hipótesis** de que si se desarrolla una Aplicación web que contribuya a perfeccionar la gestión de dietas se logrará el perfeccionamiento de este proceso en la Universidad de Matanzas.

Variable independiente:
Aplicación web para facilitar la gestión de dietas enla Universidad de Matanzas.

Variable dependiente:
Perfeccionarla gestión de dietas enla Universidad de Matanzas.

Durante la investigación se utilizaron diversos métodos de investigación científica. Entre los **métodos teóricos** estuvieron:

- análisis histórico-lógico.
- analítico-sintético.
- hipotético-deductivo.

Por otra parte, los **métodos empíricos** utilizados fueron:

- observación.
- la entrevista.
- análisis de documentos.

La investigación se estructura de la siguiente manera:

Capítulo 1: "Fundamentación teórica": en este capítulo se recoge el marco teórico referencial del tema y los conceptos principales que constituyen la base teórica de la investigación. Por otra parte, se realiza un análisis sobre los antecedentes del tema. Por último, se muestra en detalles el estudio realizado sobre las diferentes herramientas, tecnologías y metodologías que fueron utilizadas.

Capítulo 2: "Elaboración y diseño del sistema": en este capítulo se documenta el proceso de diseño y aplicación, basado en la metodología SCRUM. Posteriormente se

realiza la planificación preliminar del proyecto y un estudio de factibilidad que incluye el análisis de costos y beneficios.

Capítulo 3: "Implementación y validación de la solución propuesta": en este capítulo se llevará a cabo una serie de pruebas para comprobar la estabilidad del software, sugeridas por la metodología de desarrollo utilizada. Se evaluarán los casos de prueba y el análisis de los resultados obtenidos.

Capítulo I: Fundamentación teórica

Introducción

En el presente capitulo se abordan los principales conceptos asociados al dominio del problema, los sistemas automatizados vinculados al campo de acción, los métodos de investigación empleados, así como un análisis comparativo de otras soluciones existentes con el sistema que se propone. Además, se realiza el proceso de elaboración y diseño del sistema. Esta fase es de suma importancia, ya que tiene como objetivo principal conocer las necesidades de los clientes y conformar la arquitectura del software. También se ofrece una descripción de las herramientas que serán utilizadas como parte del entorno de desarrollo

1.1 Estudio del estado del arte

Para la realizar esta investigación se tuvieron en cuenta investigaciones anteriores que guardan similitud con la idea que se propone desarrollar.

1.1.1 ASSETS

Es un sistema de gestión integral con un módulo de finanzas, concebido con la idea de facilitar al usuario el control de la actividad de los cobros y pagos de su entidad, conocer su estado de cuentas bancarias y de clientes, así como del estado de los anticipos en Cobros y Pagos, suministrando información muy detallada acerca de las cuentas por cobrar y por pagar.

Entre las principales características funcionales de este software se encuentran:

- Permite realizar los cobros a clientes, realizar los pagos a proveedores y suministradores manteniendo actualizados los subsistemas de cuentas por cobrar y por pagar.
- Permite llevar un control de los saldos anticipados de los clientes o en los proveedores, dar tratamiento a las Letras de Cambio, controlando los Efectos por Cobrar y Efectos por Pagar.
- Facilita la obtención de los estados de cuenta de los Clientes y Proveedores, así como un conjunto de informes estadísticos para el análisis de las cuentas por cobrar y por pagar y sus vencimientos.

Además de las anteriores, el módulo de finanzas permite un grupo de operaciones de caja chica y la realización de flujos de caja. La caja son fondos que cualquier institución o empresa emplea para realizar gastos menores, como gastos de combustibles, dietas, gastos de viajes; además, para guardar los documentos generados en los procesos de

cobro, hasta su depósito en banco, almacenar el dinero para el pago de nóminas y otros conceptos.

ASSETS permite el control por Fondos de efectivo, pudiendo existir distintos tipos de fondos de acuerdo a los requerimientos de cada institución. Se procesan los documentos recibidos en los procesos de Cobros y Cobros Directos y los Pagos Menores realizados desde la caja. Además, controla los pagos de anticipos para gastos de viajes y otros conceptos y la liquidación y justificación de los mismos.

Sin embargo, aunque este programa facilita en gran medida la gestión de las dietas y los pagos menores, tiene las siguientes deficiencias:

1. La funcionalidad del software no abarca las actividades del proceso referentes a la disponibilidad de presupuesto por los centros de costo, así como la solicitud de transferencia de presupuesto de un centro de costo a otro.
2. El software no es portable a varias plataformas y debe ser instalado individualmente en las terminales para acceder a sus funcionalidades.
3. No permite obtener ningún tipo de reporte sobre la información almacenada. Por ejemplo, el control del presupuesto de dietas y pagos menores por Áreas, la información de los indicadores de dieta, y los anticipos a justificar.
4. No permite generar informes para el cierre de mes.

1.1.2 Versat Sarasola

Es un sistema de gestión contable-financiero, representa un ejemplo de sustitución de importaciones en materia de aplicaciones informáticas.

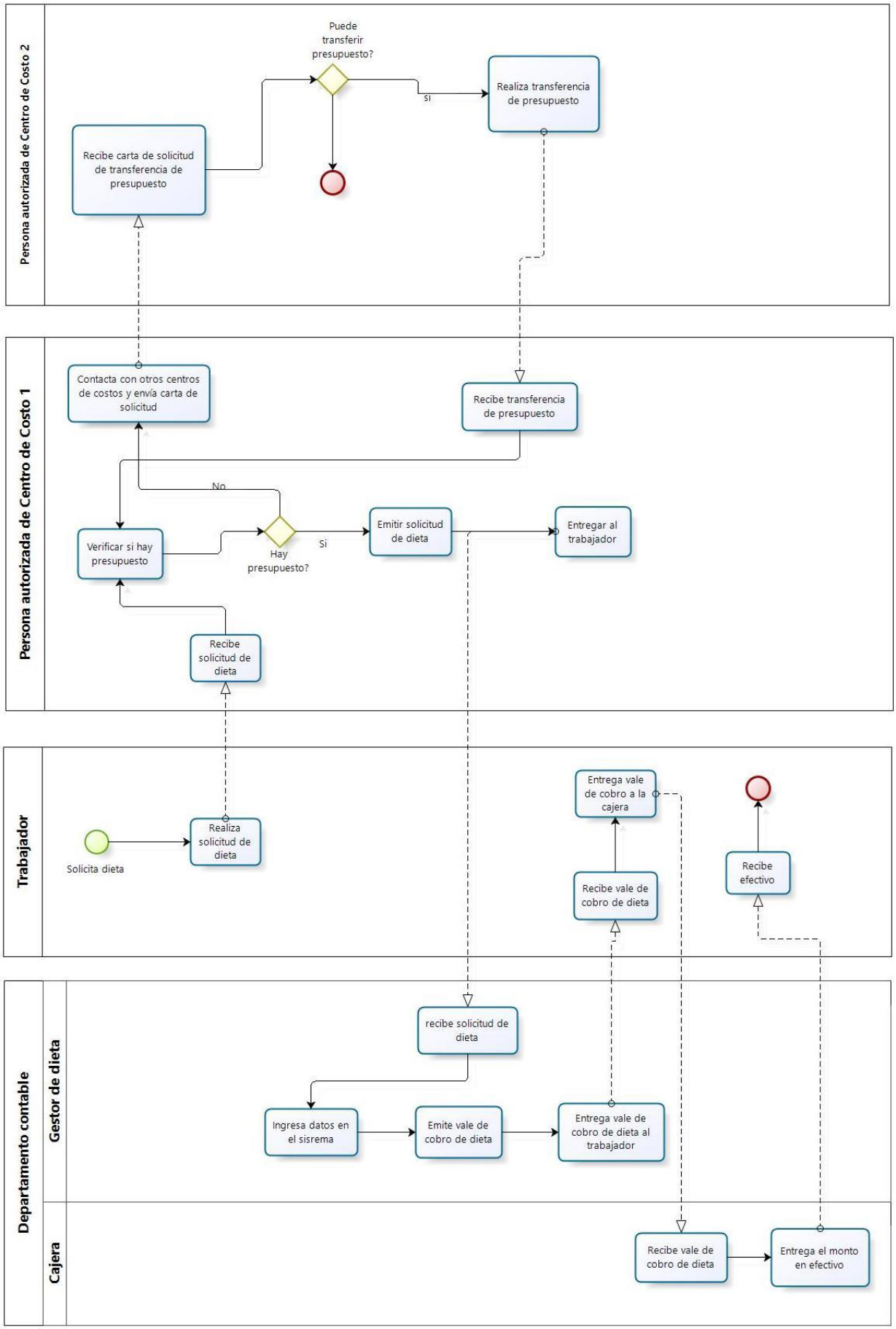
Principales características del VERSAT

- Herramienta para la planificación económica, el control y el análisis de gestión.
- Diseñado para su empleo en cualquier tipo de entidad empresarial o presupuestada.
- Permite llevar el control y registro contable individual de todos los hechos económicos que se originan en las estructuras internas de las entidades, así como exponer el estado financiero y toda la información económica y contable en este universo.

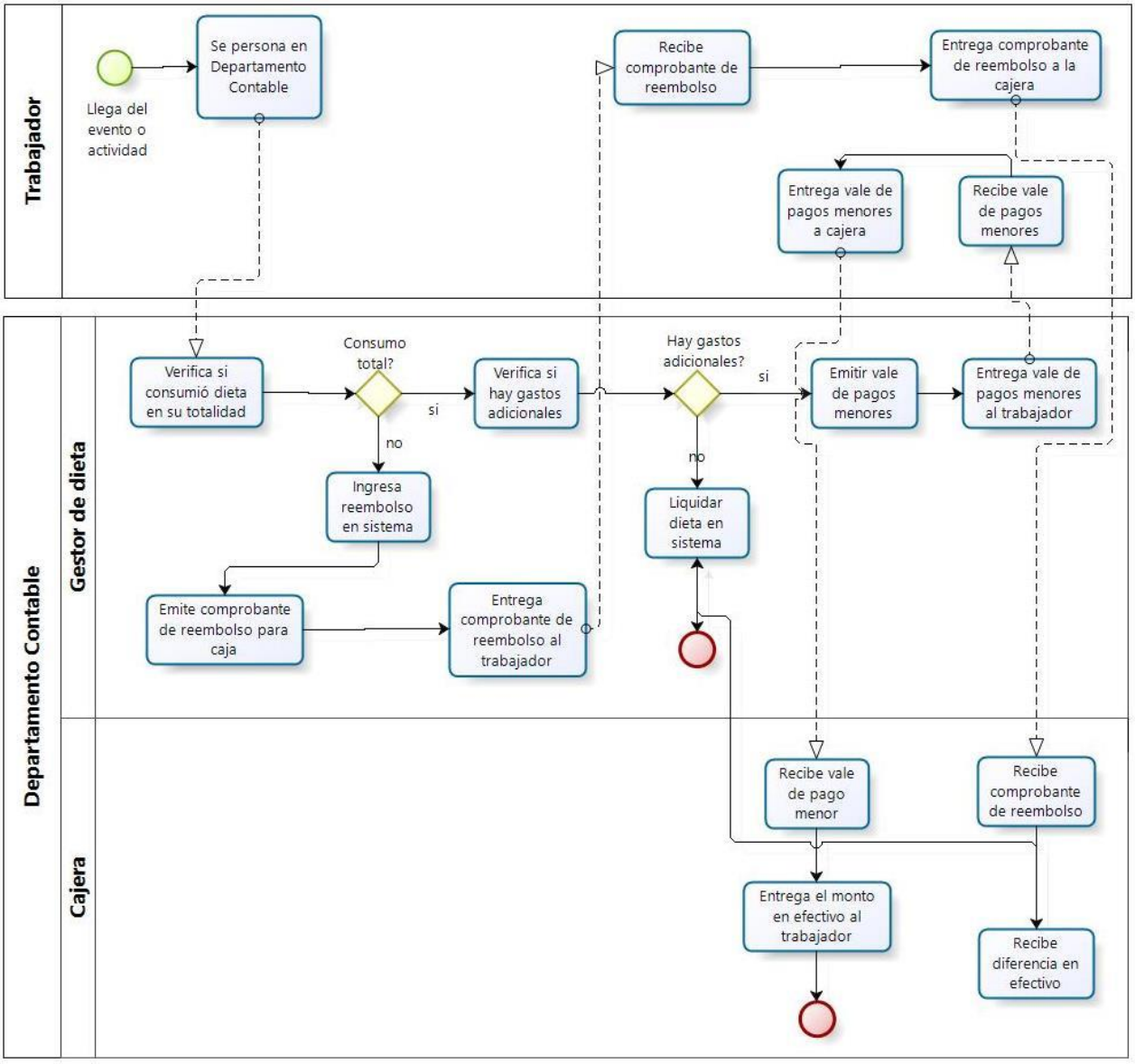
- Se estructura en un grupo de subsistemas en los cuales se procesan y contabilizan los documentos primarios, donde se anotan los movimientos, los recursos materiales, laborales y financieros que se utilizan en una entidad.
- Se logra establecer un proceso de interacción usuario-sistema.
- Rapidez y fiabilidad, a partir de la configuración del proceso de contabilización de los documentos primarios y de las propias posibilidades de trabajo contenidas en cada subsistema.

1.2 Flujo actual de los procesos más importantes involucrados en el campo de acción.

1.2.1 Proceso de solicitud de dieta


Persona autorizada de Centro de Costo 2
Puede transferir presupuesto?
si
Realiza transferencia de presupuesto
Recibe carta de solicitud de transferencia de presupuesto
Persona autorizada de Centro de Costo 1
Contacta con otros centros de costos y envía carta de solicitud
Recibe transferencia de presupuesto
No
Verificar si hay presupuesto
Hay presupuesto?
Si
Emitir solicitud de dieta
Entregar al trabajador
Recibe solicitud de dieta
Trabajador
Solicita dieta
Realiza solicitud de dieta
Entrega vale de cobro a la cajera
Recibe vale de cobro de dieta
Recibe efectivo
Departamento contable
Gestor de dieta
recibe solicitud de dieta
Ingresa datos en el sisrema
Emite vale de cobro de dieta
Entrega vale de cobro de dieta al trabajador
Cajera
Recibe vale de cobro de dieta
Entrega el monto en efectivo

1.2.2 Proceso de liquidación de dieta



1.3 Características de ambiente y herramientas

Se llegó a la conclusión de la necesidad de implementación de un nuevo software para la gestión de dietas en la Universidad de Matanzas teniendo en cuenta los resultados del estudio realizado. A continuación, se describen las metodologías, herramientas y tecnologías que se usaron para la elaboración del mismo.

1.3.1 Metodología de Desarrollo

La **metodología Scrum** para el desarrollo ágil de software es un marco de trabajo diseñado para lograr la colaboración eficaz de equipos en proyectos, que emplea un conjunto de reglas y artefactos y define roles que generan la estructura necesaria para su correcto funcionamiento (Proyectos Agiles, 2018).

Una metodología ágil, es aquella que se basa en la adaptabilidad de cualquier cambio como medio para aumentar las posibilidades de éxito de un proyecto. Empezó a existir a partir de febrero de 2001, cuando se reunieron los representantes de cada una de estas metodologías y terminaron poniendo en común sus ideas en una declaración conjunta (Balaguera, 2013).

Scrum utiliza un enfoque incremental que tiene como fundamento la teoría de control empírico de procesos. Esta teoría se fundamenta en transparencia, inspección y adaptación; la transparencia, que garantiza la visibilidad en el proceso de las cosas que pueden afectar el resultado; la inspección, que ayuda a detectar variaciones indeseables en el proceso; y la adaptación, que realiza los ajustes pertinentes para minimizar el impacto de las mismas (Andrés Navarro Caridad, 2013).

Principios básicos

Existen dos aspectos fundamentales a diferenciar en *Scrum* que son: los actores y las acciones, donde los actores son los que ejecutan las acciones.

Los actores contemplados en esta metodología son los siguientes:

- *Product Owner*, conoce y marca las prioridades del proyecto o producto.
- *Scrum Master*, es la persona que asegura el seguimiento de la metodología guiando las reuniones y ayudando al equipo ante cualquier problema que pueda aparecer. Su responsabilidad es entre otras, la de hacer de paraguas ante las presiones externas.
- *Scrum Team*, son las personas responsables de implementar la funcionalidad o funcionalidades elegidas por el *Product Owner*.
- Usuarios o Cliente, son los beneficiarios finales del producto, y son quienes, viendo los progresos, pueden aportar ideas, sugerencias o necesidades.

Las acciones de *Scrum* forman parte de un ciclo iterativo repetitivo, por lo que el mecanismo y forma de trabajar que a continuación se indica, tiene como objetivo minimizar el esfuerzo y maximizar el rendimiento en el desarrollo:

- *Product backlog*, corresponde con todas las tareas, funcionalidades o requerimientos a realizar que son marcadas por el *Product owner*.
- *Sprint backlog*, corresponde con una o más tareas que provienen de la lista de tareas (Product backlog), donde estas tareas se deben acometer en unas 2 semanas o 4 semanas. Existe una norma fundamental que mientras un Sprint backlog se inicia no debe ser alterado o modificado, hay que esperar a que concluya para hacerlo.
- *Dayli Scrum meeting*, es una tarea iterativa que se realiza todos los días que dure el Sprint backlog con el equipo de desarrollo, con lo cual se busca identificar obstáculos o riesgos que impidan el normal avance, verificar el avance de las tareas y las planificaciones de las mimas para el día.
- *Sprint planning meeting*, son reuniones cuyo objetivo es planificar el *Sprint Backlog* a partir del *Product Backlog*, suelen participar el *Scrum master*, *Scrum team* y el *Product owner.*
- *Sprint review*, una vez finalizado un *Sprint backlog*, se revisa en aproximadamente dos horas si se ha obtenido un producto que pueda ver y tocar el Cliente o usuario, donde un *Scrum team* es quien muestra los avances.
- *Sprint retrospective*, el *Product owner* revisará con el equipo los objetivos marcados inicialmente en el Sprint backlog concluido, se aplicarán los cambios y ajustes si son necesarios, y se marcarán los aspectos positivos (para repetirlos) y los aspectos negativos (para evitar que se repitan) del *Sprint backlog*.

Proceso de desarrollo

El proceso parte de la lista de tareas (*Product backlog*), que no son otra cosa que los requisitos del producto y que actúa como un plan del proyecto. De esta lista el cliente prioriza los requisitos basándose en objetivos, balanceando el valor que le aportan a su coste y quedando repartidos en iteraciones y entregas (*Sprint backlog*),

De manera regular el cliente puede maximizar la utilidad de lo que se desarrolla mediante la replanificación de objetivos que se puede realizar al inicio de cada iteración.

Cada día de una iteración debe realizarse una reunión con los integrantes del equipo con el objetivo de obtener de primera mano los avances de las tareas y los obstáculos que se van presentando a lo largo del desarrollo de la iteración

Una vez finalizado un Sprint backlog, se revisan con el usuario o cliente los productos obtenidos (Sprint review) y si cumplen con las necesidades plasmadas por el usuario al inicio de la iteración. Cada fin de un Sprint Backlog, se debe revisar los aspectos positivos y negativos del mismo con el objetivo de poder utilizar estos para una mejor planificación de la siguiente iteración a realizar. (Schwaber, 2010)

Por la amplia utilización de esta metodología en proyectos informáticos en el mundo (Yampasi, 2012) y por las facilidades que ofrece en el proceso de desarrollo siguiendo un paradigma ágil, se eligió para documentar y planificar la solución de esta investigación.

1.3.2 Lenguaje de programación del lado del cliente.

Un lenguaje del lado cliente es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio.

1.3.2.1 Lenguaje de Marcas de Hipertexto 5 (HTML5).

HTML: *Hypertext Markup Language.* 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). La descripción se basa en especificar en el texto la estructura lógica del contenido (títulos, párrafos de texto normal, enumeraciones, definiciones, citas.), así como los diferentes efectos que se quieren dar (cursiva, negrita, o un gráfico determinado) (Autores, 2014).

Es un lenguaje de marca usado para estructurar y presentar el contenido para la web. Es uno de los lenguajes de marcado más usados en todo el mundo y la razón es bastante obvia: gracias a HTML5 podemos crear la estructura de una página web y texto, imágenes y material multimedia pueden mostrarse correctamente gracias a HTML5.

1.3.2.2 CSS (Cascading Style Sheets).

Las hojas de estilo en cascada son un mecanismo que permite aplicar formato a los documentos escritos en HTML (y en otros lenguajes estructurados, como XML) separando el contenido de las páginas de su apariencia. (Benavidez, 2013) Para el diseñador, esto significa que la información estará contenida en la página HTML, pero este archivo no debe definir cómo será visualizada esa información. Las indicaciones acerca de la composición visual del documento estarán especificadas en el archivo de la CSS. Lo que posibilita crear páginas web de una manera más exacta.

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.

1.3.2.3 JavaScript.

Es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. A pesar de su nombre, legalmente JavaScript es una marca registrada de la empresa *Sun Microsystems*. (Eguiluz, 2016)

1.3.3 Lenguaje de programación del lado del servidor.

La programación del lado del servidor es una tecnología que consiste en el procesamiento de una petición de un usuario mediante la interpretación de un script en el servidor web para generar páginas HTML dinámicamente como respuesta.

1.3.3.1 PHP

Es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. PHP es un acrónimo de "*Hypertext Preprocessor*", es un lenguaje *"Open Source*" interpretado de alto nivel, especialmente pensado para desarrollos web y el cual puede ser embebido en páginas HTML. La mayoría de su sintaxis es similar a C, Java y Perl y es fácil de aprender. La meta de este lenguaje es permitir escribir a los creadores de páginas web, páginas dinámicas de una manera rápida y fácil, aunque se pueda hacer mucho más con PHP. (MALDONADO, 2014)

Por ser este lenguaje de programación usado en más de 240 millones de sitios webs (Villena, 2018), y teniendo en cuenta que el cliente especificó la utilización de herramientas

de código abierto en aras de la migración que está ocurriendo en el país hacia el software libre, se utilizó en este proyecto para la implementación de la lógica del negocio del lado del servidor.

1.3.4 Patrón de Arquitectura

MVC es una propuesta de diseño de software utilizada para implementar sistemas donde se requiere el uso de interfaces de usuario. Surge de la necesidad de crear software más robusto con un ciclo de vida más adecuado, donde se potencie la facilidad de mantenimiento, reutilización del código y la separación de conceptos. Su fundamento es la separación del código en tres capas diferentes, acotadas por su responsabilidad, en lo que se llaman Modelos, Vistas y Controladores (Bascón, 2010).

Modelos: Es la capa donde se trabaja con los datos, por tanto, contendrá mecanismos para acceder a la información y también para actualizar su estado. Los datos estarán habitualmente en una base de datos, por lo que en los modelos estarán todas las funciones que accederán a las tablas y harán los correspondientes seleccionar, actualizar, insertar (Espinosa, 2012).

Vistas: contienen el código de nuestra aplicación que va a producir la visualización de las interfaces de usuario, o sea, el código que nos permitirá generar los estados de nuestra aplicación en HTML.

Controladores: es una capa que sirve de enlace entre las vistas y los modelos, respondiendo a los mecanismos que puedan requerirse para implementar las necesidades de nuestra aplicación. Sin embargo, su responsabilidad no es manipular directamente datos, ni mostrar ningún tipo de salida, sino servir de enlace entre los modelos y las vistas para implementar las diversas necesidades del desarrollo.

Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

1.3.5 Plataforma y herramientas de desarrollo

1.3.5.1 Framework Symfony

Framework: se define en términos generales como un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que puede ser utilizada como referencia para enfrentar y resolver nuevos problemas de índole similar.
Más asociada a la informática en el desarrollo de software es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, en base a la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otras aplicaciones para ayudar a desarrollar y unir los diferentes componentes de un proyecto, es decir, son soluciones completas que llevan incorporado herramientas de apoyo a la construcción (ambiente de trabajo o desarrollo) y motores de ejecución (ambiente de ejecución).
Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio. (PRESSMAN, 2010)
Symfony es un completo *framework* diseñado para optimizar el desarrollo de las aplicaciones web mediante algunas de sus principales características. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web.
Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSql, Oracle y Microsoft SQL Server. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows. (Autores, 2009)

Symfony se diseñó para que se ajustara a los siguientes requisitos:

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares).
- Independiente del sistema gestor de bases de datos. Su capa de abstracción y el uso de Propel, permiten cambiar con facilidad de SGBD en cualquier fase del proyecto.

- Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5.
- Sencillo de usar en la mayoría de casos, está más indicado para grandes aplicaciones Web que para pequeños proyectos.
- Aunque utiliza MVC (Modelo vista controlador), tiene su propia forma de trabajo en este punto, con variantes del MVC clásico como la capa de abstracción de base de datos, el controlador frontal y las acciones.

Basado en la premisa de “convenir en vez de configurar”, en la que el desarrollador solo debe configurar aquello que no es convencional:

- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de *phpDocumentor*y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.
- Una potente línea de comandos que facilitan generación de código ahorrándonos tiempo de trabajo.

1.3.5.2 Framework Bootstrap

Es un *framework* que simplifica el proceso de creación de diseños web combinando CSS y JavaScript. Ha sido desarrollado por *Twitter*. La mayor ventaja es que se puede crear interfaces que se adapten a los distintos navegadores (*responsive design*) apoyando en un *framework* potente con numerosos componentes webs que nos ahorrarán mucho esfuerzo y tiempo. El mismo ofrece una serie de plantillas CSS y ficheros JavaScript que nos permiten integrar el *framework* de forma sencilla y potente en nuestros proyectos webs.
Permite crear interfaces que se adapten a los diferentes navegadores, tanto de escritorio como tabletas y móviles a distintas escalas y resoluciones. Se integra perfectamente con las principales librerías JavaScript, por ejemplo, JQuery. Ofrece un diseño sólido usando LESS y estándares como CSS3/HTML5. Es un *framework* ligero que se integra de forma limpia en nuestro proyecto actual. Funciona con todos los navegadores, incluido Internet

Explorer usando HTML Shim para que reconozca las etiquetas HTML5. Dispone de distintas platillas predefinidas con estructuras fijas a 940 píxeles de distintas columnas o diseños fluidos (Simmons, 2015)

Fue Utilizado dado que este *framework* se adapta a una gran cantidad de navegadores y además posee una gran cantidad de plantillas que ayudo en el diseño de la aplicación.

1.3.5.3 MySQL

Es una base de datos muy rápida en la lectura, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones. Sea cual sea el entorno en el que va a utilizar MySQL, es importante monitorizar de antemano el rendimiento para detectar y corregir errores tanto de SQL como de programación (DuBois, 2012)

Inicialmente, MySQL carecía de elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de ello, atrajo a los desarrolladores de páginas web con contenido dinámico, justamente por su simplicidad.

Poco a poco los elementos de los que carecía MySQL están siendo incorporados tanto por desarrollos internos, como por desarrolladores de software libre. Entre las características disponibles en las últimas versiones se puede destacar:

Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.

Disponibilidad en gran cantidad de plataformas y sistemas.

Posibilidad de selección de mecanismos de almacenamiento que ofrecen diferentes velocidades de operación, soporte físico, capacidad, distribución geográfica, transacciones.

Transacciones y claves foráneas.

Conectividad segura.

Replicación.

Búsqueda e indexación de campos de texto.

1.3.5.4 PhpStorm

Es un potente entorno de desarrollo integrado (IDE), especialmente diseñado a fin de proporcionar a los desarrolladores de HTML, JavaScript y PHP todas las herramientas

necesarias para su trabajo. PhpStorm proporciona un editor de código enriquecido e inteligente para PHP con resaltado de sintaxis, configuración extendida de formateo del código, comprobación de errores sobre la marcha y finalización de código inteligente. PhpStorm ofrece numerosas opciones para depurar el código PHP que te permiten:

- Establecer puntos de interrupción condicionales con paso inteligente, lo que te permite seleccionar un método específico para ejecutar desde la cadena de llamadas.
- Inspeccionar variables locales pertinentes para el contexto y los relojes definidos por ti, incluyendo los arreglos de discos y los objetos complejos y editar los valores sobre la marcha.
- Evaluar una expresión en el tiempo de ejecución.
- Depurar una página en varias sesiones simultáneamente.
- Mantener una sesión de depuración mientras te desplazas entre páginas (Kumar, 2014)

1.3.5.5 Apache

Es usado principalmente para enviar páginas web estáticas y dinámicas en la web, así como PHP y MySQL, también es liberado bajo licencia Open Source y es multiplataforma entre ellas las más populares como Unix, Microsoft Windows y Macintosh. Este se utilizó para desarrollar el sistema porque el servidor HTTP Apache es un servidor web HTTP de código abierto.

Fue utilizado en el sistema dado que este servidor está diseñado para el desarrollo web además fue pedido específicamente por el cliente potencia del software.

1.3.5.6 Herramientas CASE

Se puede definir a las Herramientas CASE como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software (investigación Preliminar, Análisis, diseño, Implementación e Instalación.) (Valle, 2014) Estas herramientas fueron utilizadas en el proceso de desarrollo de este sistema ya que permite mejorar la planificación y productividad del software, así como elevar su calidad y reducir el tiempo y costo de desarrollo del mismo. Además de todo lo antes mencionado facilitan en gran medida el uso de las distintas metodologías propias de la ingeniería del software.

1.3.5.6.1 Visual Paradigm 9.0

Es una herramienta de modelado desarrollada para asistir el proceso de Ingeniería de Software, este se encuentra basado en UML y soporta el ciclo de vida completo del desarrollo de software, además cuenta con funcionalidades más avanzadas que las presentes en el *Rational Rose* (Alonso, 2018), lo que permite agilizar considerablemente el trabajo. Sus principales características son las siguientes: presenta licencia gratuita y comercial, soporta aplicaciones web, disponible en varios idiomas. Es fácil de instalar y actualizar, compatible entre versiones, posee un entorno gráfico amigable para el usuario y está disponible en múltiples plataformas. (Adrián Peñate, 2016)

1.3.5.6.2 *Bizagi Process Modeler* versión 2.7

Reconocido por nuestra comunidad como el más potente entorno de modelamiento BPMN 1 en el mercado, Bizagi permite a los expertos de negocio diseñar, documentar, ejecutar y evolucionar sus modelos de proceso con total confianza. Una intuitiva funcionalidad de arrastrar y soltar, actualizaciones libres de código y las herramientas de generación automática de documentos, permite una excelente experiencia de usuario, incluso sin muchos conocimientos técnicos. (Bizagi, la Plataforma de Negocios digitales, 2018)

Conclusiones parciales del Capítulo I:

- El análisis de los materiales consultados y el estudio de los referentes teóricos ha permitido contar con las herramientas necesarias para dar solución la problemática enfrentada.
- De manera clara queda plasmada la necesidad de elaborar una aplicación web para gestión de dietas de la Universidad de Matanzas.
- Se justifica la utilización de una metodología ágil para el desarrollo de la aplicación, específicamente la metodología Scrum y herramientas flexibles de desarrollo como PHP, Symfony, entre otros.

[1]BPMN (Business ProcessModel and Notation) es un estándar aceptado mundialmente para el modelamiento de procesos.

Capítulo II: Elaboración y diseño del sistema

Introducción

En el presente capítulo se documenta el proceso de diseño y aplicación, basado en la metodología SCRUM, la cual es simple, pero requiere trabajo duro y constante, ya que la gestión no se basa en el seguimiento de un plan, sino en la adaptación continua a medida que evoluciona el proyecto. Posteriormente se realiza la planificación preliminar del proyecto y un estudio de factibilidad que incluye el análisis de costos y beneficios, para considerar si resulta viable o no el desarrollo de la aplicación.

2.1 Descripción de la propuesta de solución.

La aplicación web para gestionar las dietas de los trabajadores de la Universidad de Matanzas tiene como funcionalidad principal tramitar todo el proceso de la asignación de las dietas y los pagos menores a los trabajadores de la universidad. Estas son asignadas por alguna de las tres personas autorizadas en cada centro de costo. El sistema tiene cuatro roles, el rol de **administrador** del sitio web el cual gestiona los usuarios que utilicen el sistema, sus permisos, los centros de costos que están involucrados en el proceso y los cargos de los trabajadores; el rol de **centro de costo** que es el encargado de gestionar sus trabajadores, tramitar todo lo referido a la asignación de la dieta y al presupuesto que le asignaron; el rol de **departamento contable** que se encarga de asignar el presupuesto mensual a cada centro de costo y validar las dietas otorgadas, además de controlar gastos y gestionar todo lo referente a reintegros y pagos menores; y por último, y no menos importante, el rol de **cajera** que es la encargada de manejar el efectivo, ya sea entregarlo o reintegrarlo según corresponda.

2.2 Planificación inicial

La planificación se realiza con el objetivo de lograr una eficiente organización del prototipo inicial del problema y proporcionar así un buen comienzo a una solución eficaz. Con este objetivo y según las ideas del cliente sobre el software se desarrollarán las Historias de Usuarios (HU) para así conformar la Pila del Producto.

2.2.1 Equipo de trabajo y roles.

La metodología SCRUM define tres roles que rigen el proceso de desarrollo donde cada uno tiene sus responsabilidades y rinde cuentas por distintos motivos, los roles definidos para el desarrollo de la solución son:

Cliente o Dueño del Producto (*Product Owner*)

- Gestiona la lista de las funcionalidades de la aplicación y la mantiene actualizada.
- Encargado de darle prioridades a las funcionalidades más importantes para el desarrollo.
- Asegurar que el Equipo de Desarrollo entiende los elementos de la Lista del Producto al nivel necesario.

SCRUM Máster o *SCRUM Manager*

- Se encarga de gestionar y asegurar el proceso Scrum.
- Ayudar a mantener el diálogo entre el propietario del producto y el equipo.
- Se encarga de eliminar impedimentos que puedan afectar a la entrega de producto.

Equipo (*Team*)

- Desarrollan las funcionalidades el producto y se auto-organizan.
- Se encarga de crear un incremento terminado a partir de las tareas de la Pila del Producto seleccionados durante la Planificación del Sprint.
- Son multifuncionales, contando como equipo con todas las habilidades necesarias para crear un Incremento de producto.

A continuación, se definen en la Tabla 1 los roles del equipo de trabajo y los miembros involucrados en el desarrollo del producto, cabe destacar que la solución propuesta ha sido desarrollada por una única persona, la cual asumirá los roles involucrados con el desarrollo del proyecto.

Roles	Asignado a:
Product Owner	Anailyn Vera Llerena
Scrum Manager	Anailyn Vera Llerena
Team	Anailyn Vera Llerena

Cliente	Universidad de Matanzas

Tabla 1: Designación de los roles a las personas involucradas en el desarrollo de la propuesta.

2.2.2 Historias de Usuario

Las Historias de Usuarios son una descripción de una funcionalidad que debe incorporar el sistema de software, y cuya implementación aporta valor al cliente. Las historias de usuario son una forma rápida de administrar los requisitos de los usuarios sin tener que elaborar gran cantidad de documentos formales y sin requerir de mucho tiempo para administrarlos. Las historias de usuario permiten responder rápidamente a los requisitos cambiantes.
En los encuentros realizados, el cliente planteó las siguientes historias para la aplicación:

1. Gestionar usuario.
    1.1. Agregar usuario.
    1.2. Modificar usuario.
    1.3. Eliminar usuario.
    1.4. Listar usuario
    1.5. Buscar usuario.
2. Gestionar Centro de Costo.
    2.1. Agregar Centro de Costo.
    2.2. Modificar Centro de Costo.
    2.3. Eliminar Centro de Costo.
    2.4. Listar Centro de Costo.
    2.5. Buscar Centro de Costo.
3. Gestionar Trabajadores.
    3.1. Agregar Trabajadores.
    3.2. Modificar Trabajadores.
    3.3. Eliminar Trabajadores.
    3.4. Listar Trabajadores.
    3.5. Buscar Trabajadores.
4. Gestionar Concepto de Dieta.
    4.1. Agregar Concepto de Dieta.
    4.2. Modificar Concepto de Dieta.
    4.3. Eliminar Concepto de Dieta.

4.4. Listar Concepto de Dietas

5. Gestionar Clasificación.
   5.1. Agregar Clasificación.
   5.2. Modificar Clasificación.
   5.3. Eliminar Clasificación.
   5.4. Listar Clasificación
6. Insertar nueva dieta (Centro de Costo).
7. Imprimir vale de aprobación de dieta.
8. Eliminar Dieta.
9. Tramitación de dieta (Gestora de Dieta).
10. Listar Dietas
11. Entrega de Efectivo (Cajera).
12. Liquidar Dieta (Gestora de Dieta).
13. Reintegrar Dieta (Gestora de Dieta).
14. Listar Reintegros de Dietas.
15. Imprimir Informe de Reintegros de Dietas.
16. Asignar nuevo presupuesto (Gestora de Dieta).
17. Solicitar presupuesto (Centro de Costo).
18. Listar presupuestos
19. Generar carta de solicitud de Presupuesto.
20. Imprimir Informe de Transferencia de presupuesto (Centro de Costo).
21. Listar Transferencias
22. Nuevo Pago Menor (Gestora de Dieta).
23. Imprimir vale de Pago Menor.
24. Amortizar Pago Menor (Cajera).
25. Imprimir Informe de Pagos Menores.
26. Listar Pagos Menores
27. Gestionar Gastos.
   27.1. Imprimir Informe de Gastos
   27.2. Listar Gastos
28. Sistema de Búsqueda
29. Generar Reportes

2.2.3 Historias Técnicas

Se refiere a cosas que deben hacerse pero que no son un entregable ni están directamente relacionadas con ninguna historia específica, y no son de valor inmediato para el Dueño del Producto.

2.2.3.1 Apariencia o interfaz externa:

El diseño de la interfaz debe ser profesional, agradable, ágil, simple de usar. Se debe mostrar un contenido legible y confiable.

2.2.3.2 Usabilidad:

El sistema podrá ser usado por cualquier persona que posea los conocimientos básicos en el manejo de la computadora y el ambiente web en sentido general, brindándole a cada momento al usuario diversas formas de acceder a las diferentes opciones que cuenta la web.

2.2.3.3 Rendimiento

Dado a que el sistema creado es una aplicación Web, el tiempo de respuesta debe ser lo más cercano posible al tiempo real, se necesita un alto grado de eficiencia y un tiempo de respuesta muy breve, para lograr un incremento de la productividad.

2.2.3.4 Seguridad.

El sitio está creado para la facilitación de la gestión de dietas de la universidad. La zona de administración solo podrá ser utilizada por el personal capacitado para este fin. La información almacenada será revisada a través de validaciones que eliminen la entrada de datos irreales. Las contraseñas para los usuarios se deben encriptar, para almacenarlas en la base de datos, pues de esta manera se asegura que obtenerlas en texto claro, sea más difícil y quizás imposible.

2.2.3.5 Confiabilidad.

El sistema debe garantizar el tratamiento adecuado de la información y la preservación de su integridad.

2.2.3.6 Portabilidad.

El sistema podrá implantarse sobre Windows, Linux o Unix de forma tal que no haya dificultad en cambiar, de una a otra plataforma, sin necesidad de efectuar cambios. Lo anterior se debe a que la aplicación está implementada sobre PHP, que es un lenguaje multiplataforma.

Para la implantación del sistema se requiere de:

- Un servidor Unix, Linux o Windows, Apache HTTP Server como servidor Web.
- Servidor de Base de Datos: MySQL
- PHP 5.5 o superior.
- Del lado del cliente, Navegador de Internet pueden ser Microsoft Internet Explorer (a partir de la versión 4.0), Mozilla Firefox, Google Chrome entre otros.

Para el servidor:

- Procesador Pentium 800MHz o superior, Pentium III, 128Mb de memória RAM.
- Las mencionadas condiciones están en dependencia de la cantidad de usuarios conectados, por lo que las características descritas son las mínimas.

Para el cliente:

- *Display* con resolución 800 x 600.
- Procesador Pentium II o superior con 64 MB de RAM como mínimo.
- Microsoft Windows NT como sistema operativo. o superior
- Las máquinas clientes deben tener acceso al servidor.

2.2.4 Pila del producto

La pila del producto es la lista ordenada de todas las diferentes funcionalidades que debe poseer el producto. Se parte de la visión del resultado que se desea obtener y evoluciona durante el desarrollo. Es el inventario de funcionalidades, mejoras, tecnología y corrección de errores que deben incorporarse al producto a través de los sucesivos *sprints*. Representa todo aquello que esperan el cliente, los usuarios, y en general los interesados.

El Propietario del Producto (*Product Owner*) es el responsable de la Pila del Producto (*Product Backlog*), incluyendo su contenido, disponibilidad y ordenación. (Ken Schwaber, 2016)

Prioridad	Descripción	Estimado dias
1	Diseño y creación de la base de datos.	3
2	Diseño de la interfaz de usuario	2
3	Autenticación de usuario	1
4	Gestionar usuario.	1
5	Gestionar Centro de Costo.	2
6	Gestionar Trabajadores.	2
7	Gestionar Concepto de Dieta.	1/2
8	Gestionar Clasificación.	1/2
9	Insertar nueva dieta (Centro de Costo).	3
10	Imprimir vale de aprobación de dieta.	1
11	Eliminar Dieta.	1
12	Tramitación de dieta (Gestora de Dieta).	2
13	Listar Dietas	1
14	Entrega de Efectivo (Cajera).	1
15	Liquidar Dieta (Gestora de Dieta).	2
16	Reintegrar Dieta (Gestora de Dieta).	3,5
17	Listar Reintegros de Dietas.	1
18	Imprimir Informe de Reintegros de Dietas.	1/2

19	Asignar nuevo presupuesto (Gestora de Dieta).	1
20	Solicitar presupuesto (Centro de Costo).	2
21	Generar carta de solicitud de Presupuesto.	1
22	Listar presupuestos	1
23	Imprimir Informe de Transferencia de presupuesto (Centro de Costo).	1
24	Listar Transferencias	1
25	Nuevo Pago Menor (Gestora de Dieta).	3
26	Imprimir vale de Pago Menor.	1
27	Amortizar Pago Menor (Cajera).	1
28	Imprimir Informe de Pagos Menores.	1
29	Listar Pagos Menores	1
30	Gestionar Gastos.	3
31	Sistema de Búsqueda	2
32	Generar Reportes	2

2.2.5 La pila del producto en forma de Historias de Usuarios.

Se refiere a la descripción de una funcionalidad que debe incorporar el sistema de software, y cuya implementación aporta valor al cliente.

La estructura de una historia de usuario está formada por:

- Nombre breve y descriptivo.
- Descripción de la funcionalidad en forma de diálogo o monólogo del usuario describiendo la funcionalidad que desea realizar.
- Criterio de validación y verificación que determinará para considerar terminado y aceptable por el cliente el desarrollo de la funcionalidad descrita.

Historia de Usuario	
Número: 1	**Usuario:** ninguno
Nombre de la historia: Diseño y creación de la base de datos.	
Prioridad del negocio: 1	**Riesgo en desarrollo:** Alto
Estimado: 3	**sprint:** 1
Programador responsable: Anailyn Vera Llerena	
Descripción: Se inicia cuando el desarrollador realiza el diseño de la base de datos con los datos proporcionados por el cliente y a partir de estos se crean las tablas en dicha base de datos.	
Observaciones:	

Historia de Usuario	
Número: 2	**Usuario:** ninguno
Nombre de la historia: Diseño de la interfaz de usuario	
Prioridad del negocio: 2	**Riesgo en desarrollo:** Alto
Estimado: 2	**sprint:** 1
Programador responsable: Anailyn Vera Llerena	
Descripción: Se inicia cuando el desarrollador diseña y crea la interfaz del sistema teniendo en cuenta las especificaciones del cliente	
Observaciones:	

Historia de Usuario	
Número: 3	**Usuario**: Administrador
Nombre de la historia: Autenticación de usuario	
Prioridad del negocio: 3	**Riesgo en desarrollo**: alto
Estimado:1	**sprint**: 1
Programador responsable: Anailyn Vera Llerena	
Descripción: Se inicia cuando el funcionario o el administrador intentan entrar al sistema para poder realizar las actividades asignadas de acuerdo con los permisos asignados y	

este se autentifique en la página web. Si éste al autentificarse no ingresa al sitio web deberá dirigirse al administrador del sistema para revisar si se encuentra registrado en la base de datos.
Observaciones:

Historia de Usuario	
Número: 4	**Usuario:** Administrador
Nombre de la historia: Gestionar usuario	
Prioridad del negocio: 4	**Riesgo en desarrollo:** Medio
Estimado:1	**sprint:** 1
Programador responsable: Anailyn Vera Llerena	
Descripción: Cuando el administrador decide insertar, modificar o eliminar a un usuario. Esta Historia de Usuario es muy importante en la aplicación web dado que ésta es la que gestiona todo lo referente a los usuarios que interactúan con la página y donde se le asigna los permisos correspondientes.	
Observaciones:	

Historia de Usuario	
Número: 5	**Usuario:** Administrador
Nombre de la historia: Gestionar centro de costo	
Prioridad del negocio: 5	**Riesgo en desarrollo:** bajo
Estimado:2	**sprint:** 1
Programador responsable: Anailyn Vera Llerena	
Descripción: Cuando el administrador decide insertar, modificar o eliminar a un centro de costo. Esta Historia de Usuario es muy importante en la aplicación web dado que ésta es la que gestiona todo lo referente a los centros de costos que intervienen en el proceso de asignación de dietas.	
Observaciones:	

Historia de Usuario

Número: 9	**Usuario: usuarios autorizados en cada centro de costo**
Nombre de la historia: Insertar nueva dieta	
Prioridad del negocio: 9	**Riesgo en desarrollo:** Alto
Estimado:3	**sprint:** 2
Programador responsable: Anailyn Vera Llerena	
Descripción: Cuando el usuario autorizado en cada centro de costo decide aprobar e insertar una nueva dieta. Esta Historia de Usuario es muy importante en la aplicación web dado que ésta es la que inicia el proceso de gestión de dieta.	
Observaciones:	

Historia de Usuario	
Número: 25	**Usuario: usuarios autorizados en el departamento contable**
Nombre de la historia: Nuevo Pago Menor	
Prioridad del negocio: 25	**Riesgo en desarrollo:** alto
Estimado:3	**Iteración asignada:** 3
Programador responsable: Anailyn Vera Llerena	
Descripción: Cuando el usuario autorizado en el departamento contable realiza un nuevo pago menor.	
Observaciones:	

2.2.6 Planificación del Sprint

La Pila del Sprint es la lista de tareas que realiza el equipo durante el sprint para generar el incremento previsto. Los Sprint son de un ciclo corto de 2 a 3 semanas donde se pasa por todas las etapas del ciclo de desarrollo, Planificación, Desarrollo, Pruebas, Revisión y así al finalizar cada uno de los sprint se va a tener como resultado un producto viable y funcional para mostrar al cliente.

2.2.6.1 Sprint 1

Sprint	No Historia, No Tarea	Nombre de Historia, Nombre de Tarea	Fecha Inicial	Fecha Final
1	**Historia 1**	**Diseño y creación de la base de datos.**	15/01/2018	26/01/2018
	1	Diseñar el modelo de la Base de Datos.		
	2	Analizar y definir los tipos de datos.		
	3	Normalizar la base de datos.		
	4	Crear las clases de la base de datos.		
	5	Generar la base de datos.		
	6	Cargar datos de prueba.		
	Historia 2	**Diseño y creación de la interfaz**		
	1	Diseño y creación de la interfaz principal.		
	2	Diseño y creación de las interfaces de usuario en dependencia del rol.		
	3	Diseño y creación de otras interfaces.		
	Historia 3	**Autenticación de usuario**		
	1	Generar formulario de inicio de sesión.		
	2	Crear el procedimiento de encriptación.		
	3	Validar formulario.		
	5	Hacer pruebas de inicio de sesión.		
	Historia 4	**Gestionar usuario.**		
	1	Agregar usuario.		
	2	Modificar usuario.		

	3	Eliminar usuario.		
	4	Listar usuario.		
	5	Buscar usuario.		
	Historia 5	**Gestionar Centro de Costo.**		
	1	Agregar Centro de Costo.		
	2	Modificar Centro de Costo.		
	3	Eliminar Centro de Costo.		
	4	Listar Centro de Costo.		
	5	Buscar Centro de Costo.		

Al final de cada sprint hay una semana de pruebas, esta sería del 29/01/2018 al 02/02/2018.

2.2.6.2 Sprint 2

Sprint	No Historia, No Tarea	Nombre de Historia, Nombre de Tarea	Fecha Inicial	Fecha Final
2	**Historia 6**	**Gestionar Trabajadores.**	05/02/2018	16/02/2018
	1	Agregar Trabajadores.		
	2	Modificar Trabajadores.		
	3	Eliminar Trabajadores.		
	4	Listar Trabajadores.		
	5	Buscar Trabajadores.		
	Historia 7	**Gestionar Concepto de Dieta.**		
	1	Agregar Concepto de Dieta.		
	2	Modificar Concepto de Dieta.		
	3	Eliminar Concepto de Dieta.		

	4	Listar Concepto de Dieta.		
	Historia 8	**Gestionar Clasificación.**		
	1	Agregar Clasificación.		
	2	Modificar Clasificación.		
	3	Eliminar Clasificación.		
	4	Listar Clasificación.		
	Historia 9	**Insertar nueva dieta.**		
	1	Insertar nueva dieta		
	Historia 10	**Imprimir vale de aprobación de dieta.**		
	1	Imprimir vale de aprobación de dieta.		
	Historia 11	**Eliminar dieta.**		
	1	Eliminar dieta.		
	Historia 12	**Tramitación de dieta.**		
	1	Tramitar dieta.		

Al final de cada sprint hay una semana de pruebas, esta sería del 19/02/2018 al 23/02/2018.

2.2.6.3 Sprint 3

Sprint	No Historia, No Tarea	Nombre de Historia, Nombre de Tarea	Fecha Inicial	Fecha Final
3	**Historia 13**	**Listar dieta.**	26/02/2018	09/03/2018
	1	Listar dieta.		
	Historia 14	**Entrega de Efectivo.**		

	1	Entrega de Efectivo.		
	Historia 15	**Liquidar Dieta.**		
	1	Liquidar Dieta.		
	Historia 16	**Reintegrar Dieta.**		
	1	Reintegrar Dieta.		
	Historia 17	**Listar Reintegros de Dietas.**		
	1	Listar Reintegros de Dietas.		
	Historia 18	**Imprimir Informe de Reintegros de Dietas.**		
	1	Imprimir informe de Reintegros de Dietas.		
	Historia 19	**Asignar nuevo presupuesto.**		
	1	Asignar nuevo presupuesto.		

Al final de cada sprint hay una semana de pruebas, esta sería del 12/03/2018 al 16/03/2018.

2.2.6.4 Sprint 4

Sprint	No Historia, No Tarea	Nombre de Historia, Nombre de Tarea	Fecha Inicial	Fecha Final
3	**Historia 20**	**Solicitar presupuesto.**	19/03/2018	30/03/2018
	1	Solicitar presupuesto		
	Historia 21	**Listar Presupuestos.**		
	1	Listar presupuestos.		
	Historia 22	**Generar carta de solicitud de Presupuesto.**		

	1	Generar carta de solicitud de presupuesto		
	Historia 23	**Imprimir Informe de Transferencia de presupuesto.**		
	1	Imprimir Informe de Transferencia de presupuesto.		
	Historia 24	**Listar Transferencias.**		
	1	Listar Transferencias.		
	Historia 25	**Nuevo Pago Menor.**		
	1	Nuevo Pago Menor.		
	Historia 26	Imprimir vale de Pago Menor.		
	1	Imprimir vale de Pago Menor.		

Al final de cada sprint hay una semana de pruebas, esta sería del 02/04/2018 al 06/04/2018.

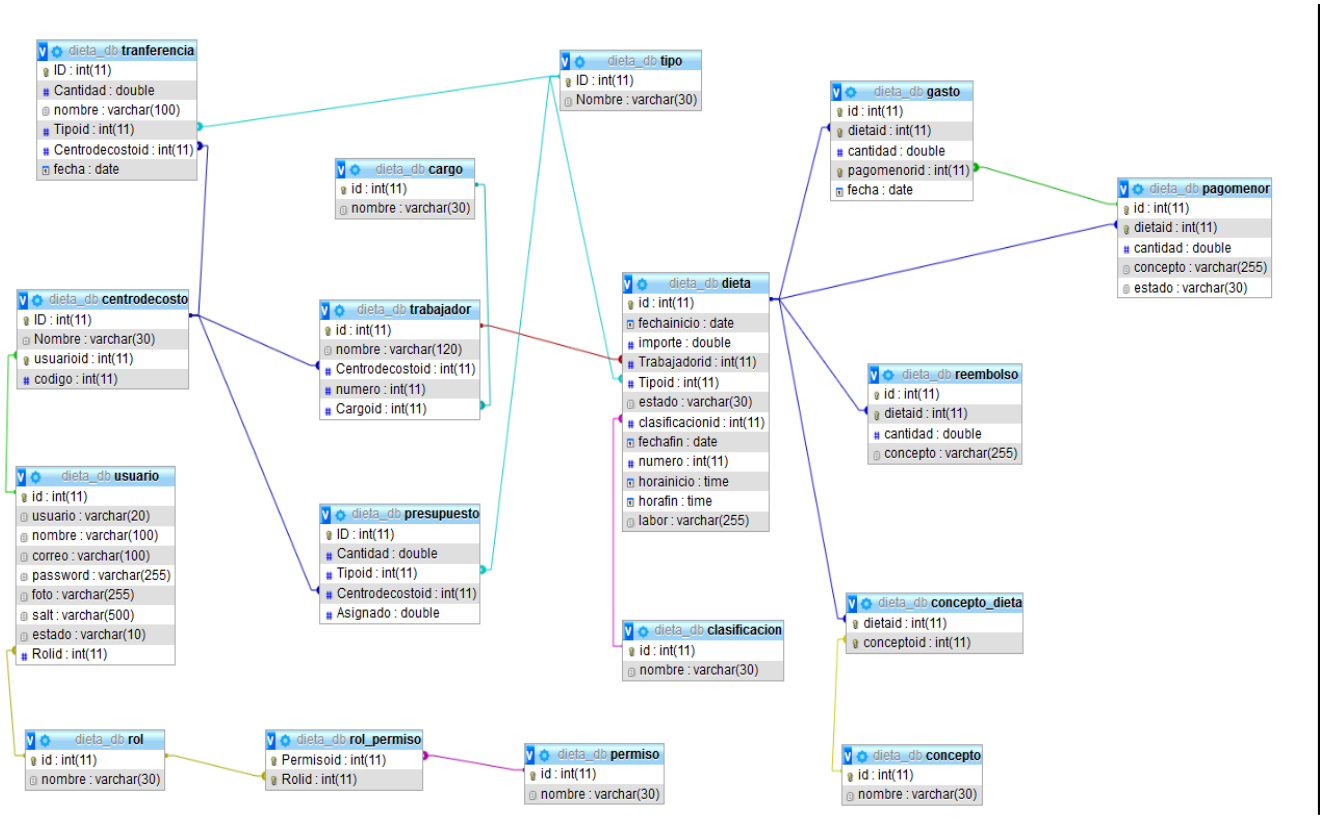
2.2.6.5 Sprint 5

Sprint	No Historia, No Tarea	Nombre de Historia, Nombre de Tarea	Fecha Inicial	Fecha Final
3	**Historia 27**	**Amortizar Pago Menor.**	09/04/2018	20/04/2018
	1	Amortizar Pago Menor		
	Historia 28	**Imprimir Informe de Pagos Menores.**		
	1	Imprimir Informe de Pagos Menores.		
	Historia 29	**Listar Pagos Menores.**		
	1	Listar Pagos Menores		
	Historia 30	**Gestionar Gastos.**		

	1	Imprimir Informe de Gastos.		
	2	Listar Gastos.		
	Historia 31	**Sistema de Búsqueda.**		
	1	Sistema de Búsqueda.		
	Historia 32	**Generar Reportes.**		
	1	Generar Reportes.		

Al final de cada sprint hay una semana de pruebas, esta sería del 23/04/2018 al 27/04/2018.

2.2.7 Modelo de datos.



2.2.8 Análisis de factibilidad

La estimación es el proceso de medición anticipada de la duración, esfuerzos y costes 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)

El software fue elaborado y puesto a disposición del cliente, el cual no financió dicho proyecto. En caso de que lo tuviese que hacer, se calcularía el costo monetario utilizando la siguiente fórmula:

Fórmulas de Bohem:

CT=CH*SM*TD

CT---Costo Total

CH---Cantidad de Hombres

SM---Salario Mensual

TD---Tiempo de Desarrollo

CH = 1 desarrollador

SM= $ 500,00

TD= 3.75 meses (10 semanas de duración de desarrollo + 5 semanas de pruebas= 15 semanas) = (50 días de duración de desarrollo + 25 días de pruebas= 75 días) / 20 días laborables)

CT=CH*SM*TD

CT= 1* $ 500, 00 * 3.75

CT= $ 1875, 00

Considerando un salario mínimo de $500.00 el costo del proyecto es de mil ochocientos setenta y cinco pesos con cero centavos ($ 1875, 00).

2.2.9 Beneficios tangibles e intangibles.

La aplicación desarrollada presenta beneficios muy favorables ya que este software le ofrece una herramienta que facilita el proceso de gestión de dietas en la Universidad de Matanzas y da solución a la problemática planteada. Además, la aplicación permite contar con un sistema de fácil acceso, control y aumento de prestaciones permitiendo de esta forma aumentar la calidad del trabajo y por ende disminuir el esfuerzo del usuario para realizar la actividad, satisfaciéndose así las necesidades requeridas por el cliente.

Conclusiones parciales del Capítulo II

El capítulo II abordó lo referente al proceso de ingeniería de software llevado a cabo por la metodología Scrum, se definió el equipo de trabajo y se le asignaron sus respectivos roles. Se identificaron los requerimientos. Fueron definidas las historias de usuario, la pila del producto y la pila del Sprint. Se plantearon las fases de desarrollo de la aplicación. Se realizó un estudio de factibilidad de manera empírica aplicando la fórmula de Bohem para el análisis de costos y beneficios de la realización del software.

Teniendo en cuenta que dicho análisis arribó a la conclusión de que no se trataba de un proyecto costoso, se puede plantear que resulta provechoso el empleo de tiempo y esfuerzo en la realización de este.

Capítulo III: Implementación y validación de la solución propuesta

Introducción

En el presente y último capítulo se llevará a cabo una serie de pruebas para comprobar la estabilidad del software, sugeridas por la metodología de desarrollo, sobre la solución creada. Se evaluarán los casos de prueba, las estrategias de prueba y el análisis de los resultados obtenidos.

Como resultado del capítulo III se presenta un software que responde a los objetivos planteados en la introducción de este trabajo, y este capítulo estará dedicado esencialmente a la validación dicho software.

3.1 Pruebas

La fase de Prueba es uno de los pilares que sostienen la metodología Scrum. Durante esta etapa serán probados todos los componentes del producto desarrollado, tanto por el cliente como por el equipo de desarrollo. Las pruebas realizadas en algunos casos generarán artefactos que serán guardados por el equipo de desarrollo para futuras mejoras del producto.

3.1.1 Pruebas de Aceptación

Las pruebas de aceptación son las realizadas por el cliente y usuarios finales de la aplicación. En estas serán probadas las funcionalidades exigidas por el cliente y descritas en las historias de usuario, además de los aspectos de seguridad requeridos. Luego de haber superado las pruebas de aceptación podrá considerarse que la aplicación es apta para el uso y despliegue dentro del proyecto.

Como resultado de las pruebas de aceptación se obtendrán artefactos descritos en tablas, estas contarán con los siguientes campos:

1. Número de Caso de prueba**:** servirá como identificador de la prueba realizada, a su vez será sugerente al nombre de la prueba a la que hace referencia.
2. Historia de Usuario: tendrá el nombre de la historia de usuario a la que hace referencia la prueba a realizar.
3. Nombre: nombre que se le da a la prueba a realizar.
4. Descripción: se describe la funcionalidad que se desea probar.

5. Condiciones de Ejecución: mostrará las condiciones que deben cumplirse para poder llevar a cabo el caso de prueba, estas condiciones deben ser satisfechas antes de la ejecución del caso de prueba para que se puedan obtener los resultados esperados.
6. Entradas: se hará la descripción de cada uno de los pasos seguidos durante el desarrollo de la prueba, se tendrá en cuenta cada una de las entradas que hace el usuario con el objetivo de ver si se obtiene el resultado esperado.
7. Resultado esperado: se hará una breve descripción del resultado que se espera obtener con la prueba realizada.
8. Evaluación de la prueba: acorde al resultado de la prueba realizada se emitirá una evaluación sobre la misma. Esta evaluación tendrá uno de los tres resultados que a continuación se describen:

- Satisfactoria: cuando el resultado de la prueba es exactamente el esperado por el usuario.
- Parcialmente satisfactoria: cuando el resultado no es completamente el esperado por el cliente o usuario de la aplicación y muestra resultados erróneos o fuera de contexto.
- No satisfactoria: cuando el resultado de la prueba realizada genera un error de codificación en la aplicación o muestra como resultado elementos no deseados o fuera de contexto y trae como consecuencia que la funcionalidad requerida por el cliente no tenga resultado, lo que invalida también la Historia de Usuario.

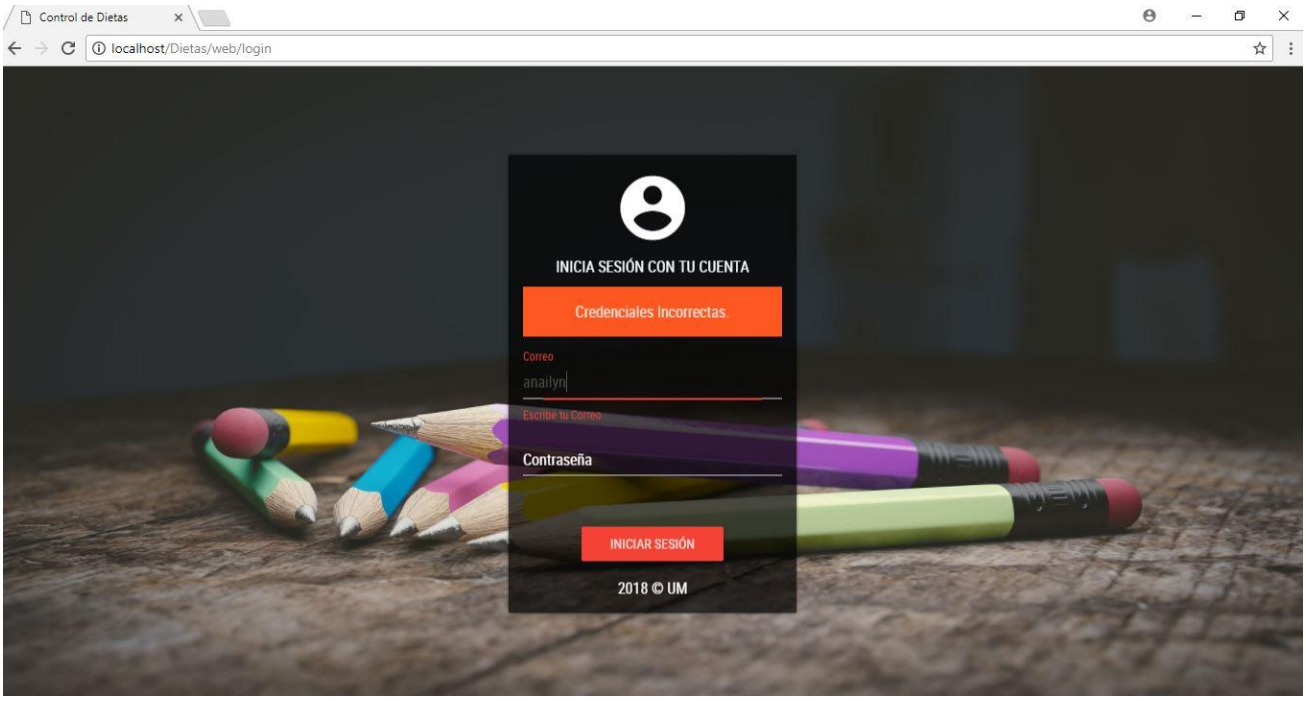
A continuación, se muestran las principales pruebas de aceptación que se realizaron al sistema:

3.1.1.1 Autentificar usuario

Pruebas de Aceptación	
Autentificar usuario	**Número de historia:** 3
Responsables: Anailyn Vera Llerena	

Descripción: Se inserta el nombre de usuario y la contraseña para entrar al sistema. Se insertarán de forma incorrecta introduciendo usuarios y contraseñas que no esté registrada en el sistema, dejando campos en blanco para verificar la validación, se tratará de entrar al sistema poniendo una ruta destino para comprobar que no entre solo si se autentifica correctamente. Luego se insertarán los datos de manera correcta para comprobar esta funcionalidad.
Condiciones de ejecución: El usuario tendrá acceso a las funcionalidades en las que tenga los privilegios correspondientes.
Entrada/Pasos de ejecución: Presionar el botón Entrar con los campos en blanco. Fijar una ruta para acceder a ella sin haberse autentificado. Insertar los datos correctamente y presionar el botón Entrar.
Resultado esperado: El sistema debe alertar al usuario cuando se inserten datos erróneos. Cuando se inserten los datos correctamente, el sistema debe entrar y mostrar las opciones de la entidad a la que pertenece el usuario con los permisos asociados al mismo.
Evaluación de la prueba: Satisfactoria

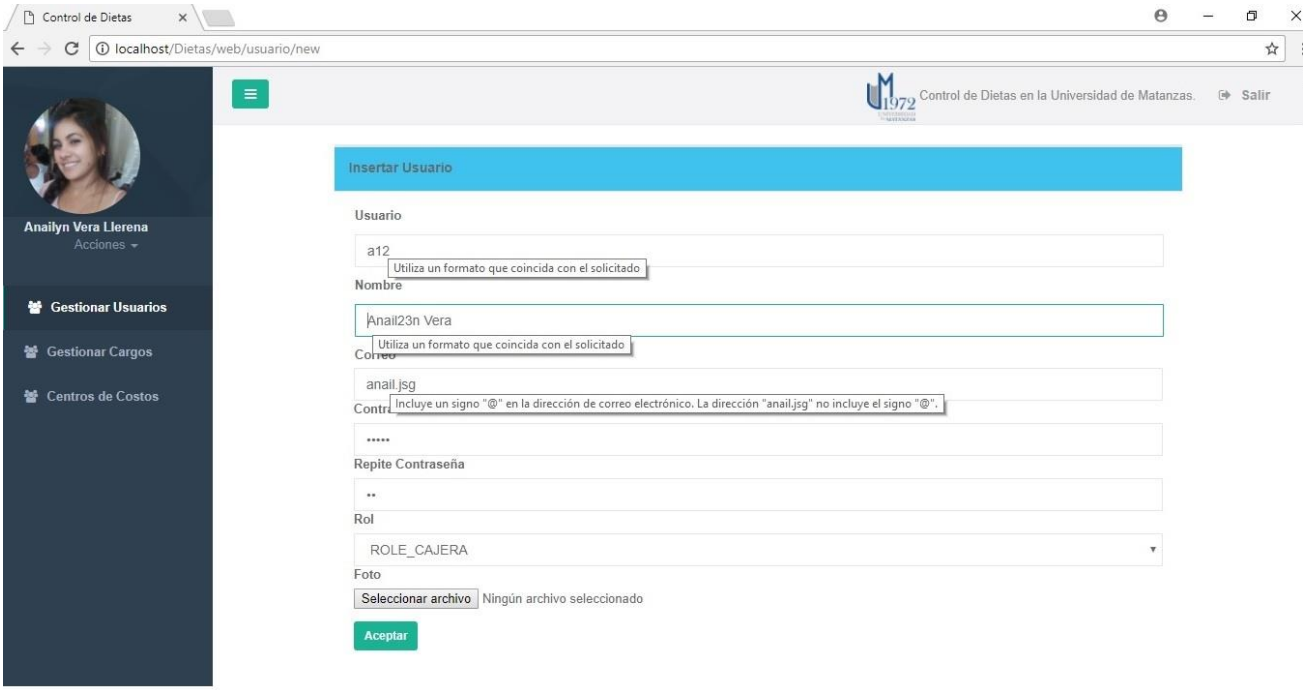
3.1.1.1.1 Resultados de prueba de aceptación validando los campos.



3.1.1.2 Gestionar Usuario

Pruebas de Aceptación	
Gestionar Usuario	**Número de historia:** 4
Responsable: Anailyn Vera Llerena	
Descripción: Se insertan los datos necesarios para crear los usuarios. Se insertarán de forma incorrecta los datos tal es el caso del nombre y apellido, Nombre del usuario, contraseña, confirmar contraseña, dejando campos en blanco, para verificar la validación se tendrá en cuenta que el usuario no esté ya creado. Luego se insertarán de manera correcta para comprobar que los datos sean almacenados. Se modifican los datos de los Usuarios. Se modificarán de forma incorrecta, dejando campos en blanco para verificar la validación, luego se modificarán de manera correcta para comprobar que los datos sean almacenados y cargados. Se eliminarán los usuarios presionando en el botón que indica la acción.	
Condiciones de ejecución: Solo puede agregar un usuario el Administrador, este tiene permiso y acceso sobre esta parte del sistema, siendo también el encargado de eliminar y modificar los usuarios.	
Entrada/Pasos de ejecución: Dejar campos en blanco. Insertar los datos correctamente. Modificar los datos dejando campos en blanco. Modificar los datos de forma correcta. Verificar que se muestren los nuevos datos. Insertar datos de manera incorrecta.	
Resultado esperado: El sistema debe alertar al usuario cuando se inserten datos erróneos. Cuando se inserten los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrarlos. No debe modificar los campos en blanco. Cuando se modifiquen los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrar los nuevos.	
Evaluación de la prueba: Satisfactoria	

3.1.1.2.1 Resultados de prueba de aceptación validando los campos.

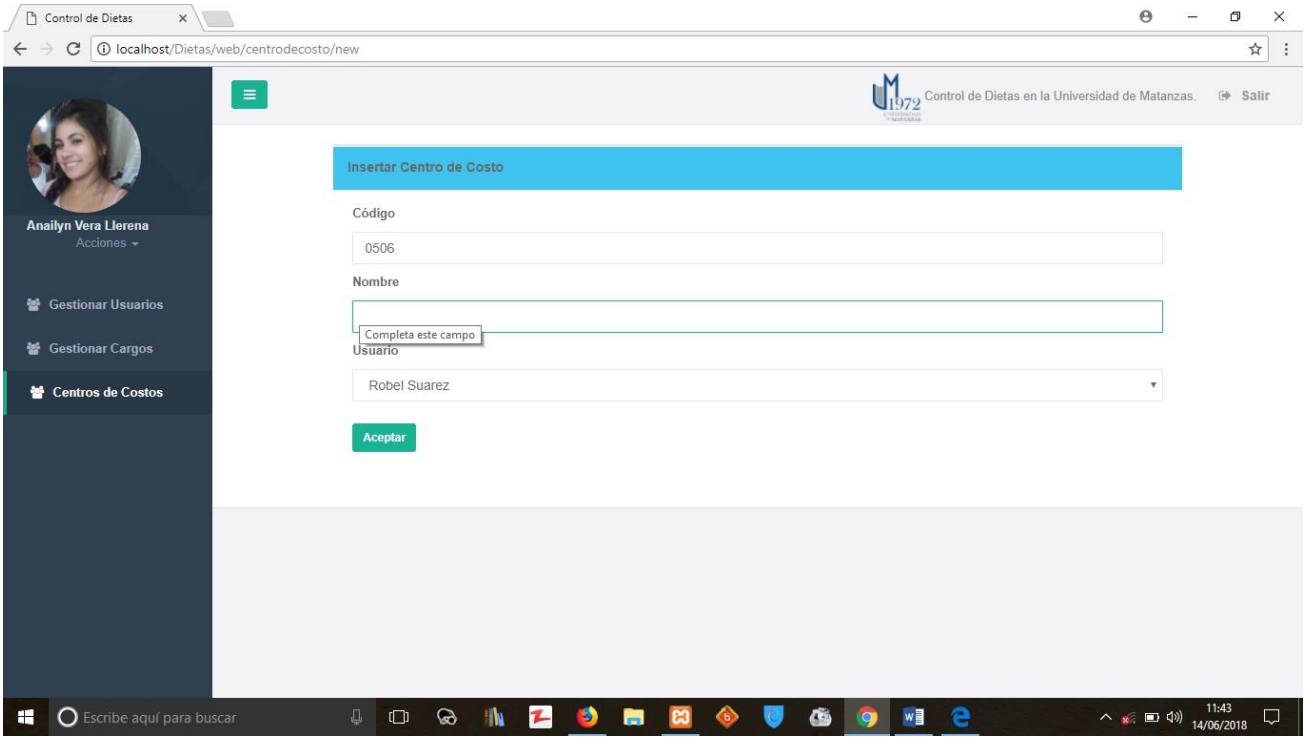


3.1.1.3 Gestionar Centro de Costo

Pruebas de Aceptación	
Gestionar Centro de Costo	**Número de historia:** 5
Responsable: Anailyn Vera Llerena	
Descripción: Se insertan los datos necesarios para crear los centros de costos. Se insertarán de forma incorrecta los datos tal es el caso del nombre del centro de costo, dejando campos en blanco, para verificar la validación se tendrá en cuenta que el componente no esté ya creado. Luego se insertarán de manera correcta para comprobar que los datos sean almacenados. Se modifican los datos de los centros de costos. Se modificarán de forma incorrecta, dejando campos en blanco para verificar la validación, luego se modificarán de manera correcta para comprobar que los datos sean almacenados y cargados. Se eliminará los centros de costos presionando en el botón que indica la acción.	
Condiciones de ejecución: Solo puede agregar un centro de costo el Administrador, este tiene permiso y acceso sobre esta parte del sistema, siendo también el encargado de eliminar y modificar los centros de costo.	

Entrada/Pasos de ejecución: Dejar campos en blanco. Insertar los datos correctamente. Modificar los datos dejando campos en blanco. Modificar los datos de forma correcta. Verificar que se muestren los nuevos datos. Insertar datos de manera incorrecta.
Resultado esperado: El sistema debe alertar al usuario cuando se inserten datos erróneos. Cuando se inserten los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrarlos. No debe modificar los campos en blanco. Cuando se modifiquen los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrar los nuevos.
Evaluación de la prueba: Satisfactoria

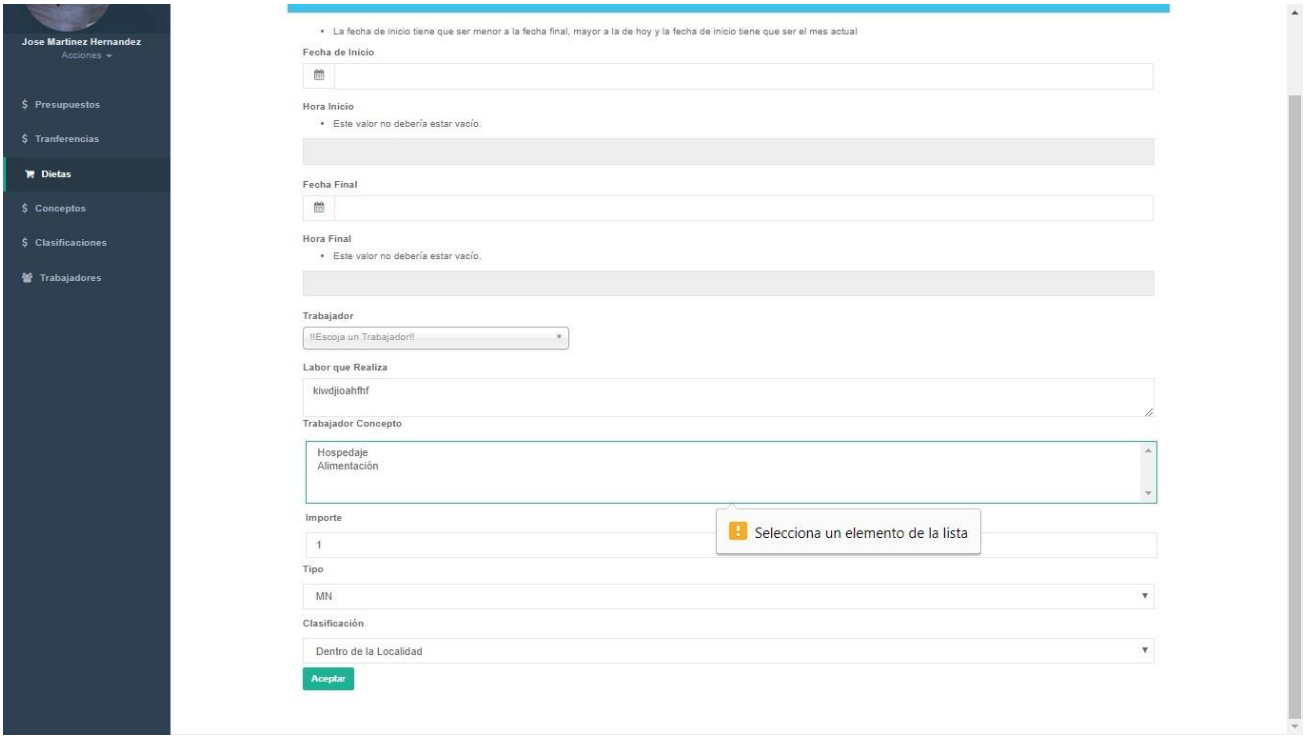
3.1.1.3.1 Resultados de prueba de aceptación validando los campos.



3.1.1.4 Insertar nueva Dieta

Pruebas de Aceptación	
Insertar nueva dieta	**Número de historia:** 9
Responsable: Anailyn Vera Llerena	
Descripción: Se insertan los datos necesarios para crear la nueva dieta. Se insertarán de forma incorrecta los datos tal es el caso de la fecha de inicio y la fecha final, dejando campos en blanco, para verificar la validación. Luego se insertarán de manera correcta para comprobar que los datos sean almacenados. Se eliminará la dieta presionando en el botón que indica la acción.	
Condiciones de ejecución: Solo puede insertar una nueva dieta los usuarios pertenecientes al rol centro de costo, estos tienen los permisos y el acceso sobre esta parte del sistema.	
Entrada/Pasos de ejecución: Dejar campos en blanco. Insertar los datos correctamente. Modificar los datos dejando campos en blanco. Modificar los datos de forma correcta. Verificar que se muestren los nuevos datos. Insertar datos de manera incorrecta.	
Resultado esperado: El sistema debe alertar al usuario cuando se inserten datos erróneos. Cuando se inserten los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrarlos. No debe modificar los campos en blanco. Cuando se modifiquen los datos correctamente, el sistema debe almacenarlos en la base de datos y mostrar los nuevos.	
Evaluación de la prueba: Satisfactoria	

3.1.1.4.1 Resultados de prueba de aceptación validando los campos.



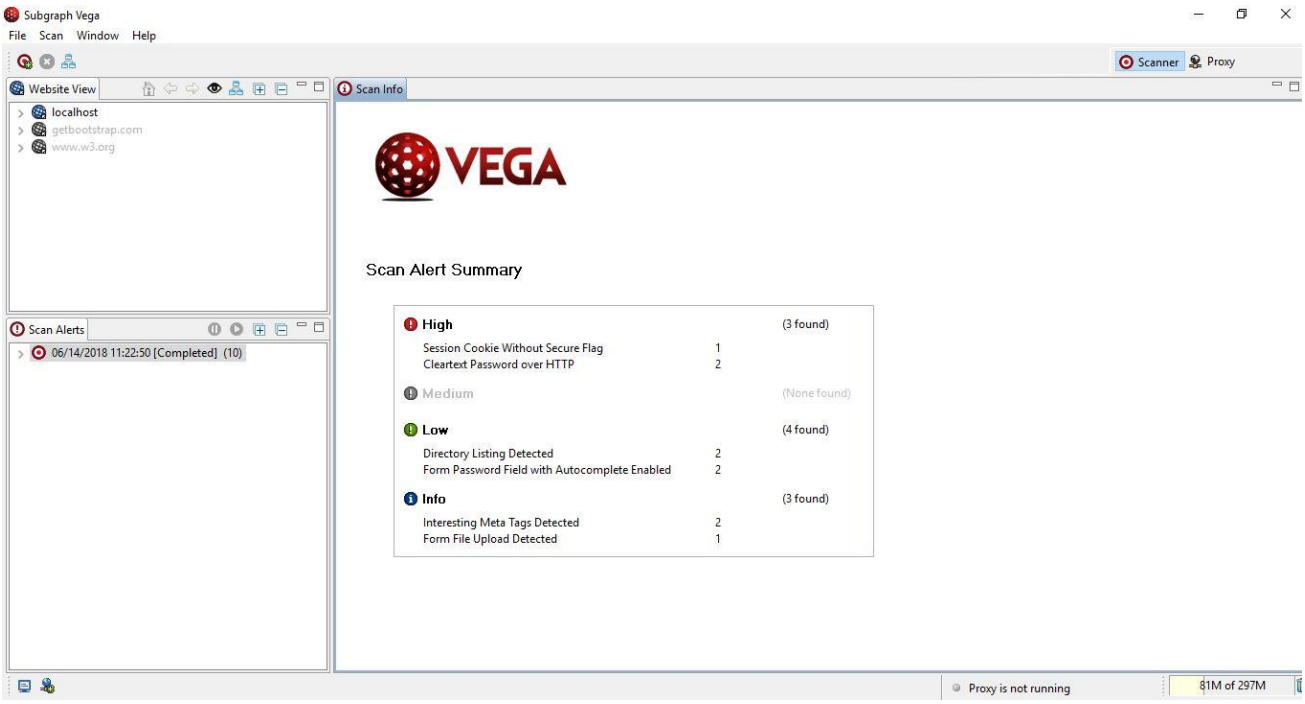
3.2 Prueba Automatizadas

En las pruebas de software, la automatización de pruebas consiste en el uso de software especial (casi siempre separado del software que se prueba) para controlar la ejecución de pruebas y la comparación entre los resultados obtenidos y los resultados esperados. La automatización de pruebas permite incluir pruebas repetitivas y necesarias dentro de un proceso formal de pruebas ya existente o bien adicionar pruebas cuya ejecución manual resultaría difícil.

A continuación, se explica la herramienta que se utiliza para el proceso de pruebas automatizadas del software resultante de este proyecto.

3.2.1 Subgraph Vega**:** es un escáner de Vulnerabilidades de código abierto para probar la seguridad de aplicaciones web, en la cual puede ayudarle a encontrar y validar las Inyecciones SQL, Cross- Site Scripting (XSS), Shell Injection, Local File Inclusion, Integer Overflow y otras vulnerabilidades. Está programada en Java y basado en una interfaz gráfica por la cual se puede ejecutar en Linux, OS X y Windows. Vega incluye un escáner automatizado para pruebas rápidas y también un Proxy para proteger nuestra IP cuando realizamos alguna auditoria web. Vega fue desarrollado por Subgraph en Montreal. Actualmente esta herramienta viene incorporada en el BackTrack, así como en Kali, Bugtraq entre otras.

3.2.1.1 Resultado de una prueba realizada por Subgraph Vega



Conclusiones Generales

Como resultado de esta investigación quedaron cumplidos los objetivos trazados, y la autora arriba a las siguientes conclusiones:

- El estudio realizado sobre los antecedentes, el estado actual de la temática, la bibliografía y documentos relacionados con el objeto de estudio, permitió aportar los elementos necesarios para dar solución a la problemática planteada.
- El uso de la metodología de desarrollo de software Scrum permitió el diseño, implementación y prueba de una aplicación a la medida de las necesidades del cliente y flexible ante los cambios en los requerimientos.
- El empleo del lenguaje PHP y el Framework Symfony para aplicaciones Web resultó factible para la implementación de la propuesta.
- El sistema implementado da solución a la situación problémica que lo originó y su explotación significará una mejora considerable en la calidad y eficiencia de los procesos que automatiza.

Bibliografía

Adrián Peñate, L. G. (2016). Módulo de filtrado y segmentación de imágenes médicas digitales para el proyecto Vismedic. *Revista Cubana de Ciencias Informáticas*, 13-27. Obtenido de http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992016000100002

Alonso, E. M. (2018). *Herramientas CASE para el proceso de desarrollo de Software*. Obtenido de http://www.monografias.com/trabajos73/herramientas-case-proceso-desarrollo-software/herramientas-case-proceso-desarrollo-software2.shtml

Andrés Navarro Caridad, J. D. (julio-diciembre de 2013). Revisión de metodologías ágiles para el desarrollo de software. *PROSPECTIVA, vol. 11*(2), 30-39. Obtenido de http://www.redalyc.org/articulo.oa?id=496250736004

Autores. (2014). *HTML. Hyper Text Markup Language.*

Balaguera, Y. D. (2013). Metodologías ágiles en el desarrollo de aplicaciones para dispositivos móviles. Estado actual. *Revista de Tecnología*, 111-124.

Bascón, E. (2010). *El patrón de diseño modelo-vista-controlador (MVC) y su implementación en Java.*

Benavidez, C. (2013). *Carlos Benavidez, HOJAS DE ESTILO EN CASCADA.* Recuperado el 2018 de mayo de 8, de http://www.carlos-benavidez.net

Bizagi, la Plataforma de Negocios digitales. (2018). Obtenido de http://help.bizagi.com/bpm-suite/es/index.html?overview.htm

Dialnet, F. (s.f.). *Dialnet.* Obtenido de https://dialnet.unirioja.es/descarga/articulo/4752083.pdf

DuBois. (2012). *MySQL.*

Eguiluz, J. (2016). *Libros Web, Introducción a JavaScript*. Recuperado el 28 de mayo de 2018, de http://librosweb.es/libro/javascript/

Espinosa, A. T. (2012). *Automatización de la codificación del patrón modelo-vista-controlador (MVC) en proyectos orientados a la web.*

Ken Schwaber, J. S. (julio de 2016). *La Guía Definitiva de Scrum: Las Reglas del Juego.* Obtenido de http://creativecommons.org/licenses/by-sa/4.0/

Kumar, C. a. (2014). *PhpStorm.*

MALDONADO, A. M. (2014). *Manual de PHP.*

Precio, M. d. (2014). Resolución No.267/2014.

Pressman. (2010).

PRESSMAN, R. S. (2010). *Software Engineere.* España: Mc Graw-Hill.

Proyectos Agiles. (2018). Obtenido de https://proyectosagiles.org/que-es-scrum/

Schwaber, K. (2010). *SCRUM.* Recuperado el 2018 de abril de 5, de http://www.controlchaos.com/

Simmons, D. (2015). *Framework Bootstrap.*

Valle, J. (febrero de 2014). *Herramientas CASE*. Recuperado el 29 de mayo de 2018, de http://www.monografias.com/trabajos24/herramientas-case/herramientas-case.shtml

Villena, J. C. (22 de Febrero de 2018). *Educa Sistema*. Obtenido de http://www.educasistemas.com/2018/02/lenguajes-de-programacion-con-los-que.html

Yampasi, E. H. (2012). Scrum Distribuido Una Metodologia De Desarrollo En La Nube. *Revista de Información, Tecnología y Sociedad*, 77-79.