UNIVERSIDAD DE MATANZAS

FACULTAD DE CIENCIAS TÉCNICAS

DEPARTAMENTO DE INFORMÁTICA

Aplicación Web para la gestión del proceso de contratación en la Estación Experimental de Pastos y Forrajes Indio Hatuey

Trabajo de Diploma en opción al título de Ingeniero Informático

Autor: Daniel Navarro Machín

Tutor: Dr.C Emilio Soler Cárdenas

Matanzas, 2022

Dedicatoria

A mi familia y amigos por estar siempre ahí, en especial a mis padres por inspirarme y permitirme ser quien soy hoy en día. Este trabajo, el reflejo de años de sacrificio y superación personal, se los dedico a ustedes y que sepan que siempre, estemos donde estemos, les deberé todo en mi vida.

Agradecimientos

Quiero agradecer a todos los que de una forma u otra influyeron en mí durante esta tan dura etapa de estudios y superación.

A mis padres, por su apoyo incondicional en todo momento y por hacer de mi el hombre que soy hoy; por ser mi faro, guía y modelo a seguir; los quiero, admiro y estaré en deuda por siempre con ustedes.

A mis amigos, responsables de mis momentos de liberación, por su apoyo moral y material. Kenidel, Asbell, Yordanis, Ley Yoan; esta va por ustedes, un último trago amargo para después celebrarlo juntos, porque de una forma u otra este título es de ustedes también.

A mis compañeros de carrera, en especial a los que han estado conmigo desde el inicio, aguantándome en la beca.

A mi tutor Emilio por su guía certera y precisa por este último tramo de mi camino a cumplir una de mis metas.

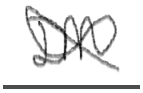
A la Estación Experimental de Pastos y Forrajes Indio Hatuey, por servir de cuartel general, por confiar en mí y permitirme el desarrollo de mis sueños.

A la Universidad de Matanzas, por permitirme todos estos momentos en sus instalaciones y por brindarme los medios para convertirme en el profesional que soy hoy.

En fin, a todos los que estuvieron ahí cuando necesité que lo estuvieran, gracias y esta va por ustedes.

Declaración de autoría

Yo, Daniel Navarro Machín, declaro ser autor de la presente tesis y reconozco a la Universidad de Matanzas los derechos patrimoniales de la misma. Para que así conste firmo la presente a los ____ días del mes de _____ del 2022.



Firma del Autor

Daniel Navarro Machín

Firma del Tutor

Dr.C Emilio Soler

Opinión del usuario del Trabajo de Diploma

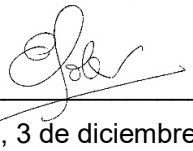
En nuestra entidad, la actividad económica en lo fundamental se desarrolla a traves de contratos. Este acto jurídico se ejecuta en un plazo prefijado entre la empresa y los actores económicos, permitiendo la ejecución del presupuesto y todos los ingresos financieros. El Gestor de Contratos realizado por el estudiante de Ingeniería en Informática Daniel Navarro Machín, es de gran interés, muy necesario y útil para nuestra empresa. La automatización perfecciona el proceso contractual, ahorra tiempo, humaniza el trabajo y previene ante la comisión de ilegalidades.

Opinión del Tutor del Trabajo de Diploma

El desarrollo de este trabajo de diplomas viene a dar una solución a la problemática relacionada con la gestión de contratos en la empresa Estación Experimental de Pastos y Forrajes Indio Hatuey de Matanzas. La aplicación Web fue desarrollada con software libre, es escalable, lo que facilitará el mantenimiento y su introducción en similares Estaciones de Pastos y Forrajes del país.

El estudiante Daniel Navarro Machín cumplió en detalles las exigencias del cliente respecto a la automatización del proceso de contratos, ajustándose a la infraestructura informática existente en la entidad y las políticas de seguridad establecidas. No tengo dudas de la dedicación y los esfuerzos realizados por Daniel, mostró mucha independencia y creatividad. También fue muy receptivo y respetuoso con las orientaciones recibidas, tanto desde el cliente como del tutor. Puedo afirmar que siento un gran honor por haber orientado este trabajo de diplomas, el tiempo será testigo sobre la formidable formación adquirida por Daniel durante su carrera.

Por último, recomiendo que el estudiante Daniel Navarro Machín sea evaluado con la máxima calificación, en dependencia de la calidad de la presentación y defensa de este trabajo de diploma.



Matanzas, 3 de diciembre de 2022
Año 64 de la Revolución
Tutor Dr.C Emilio Soler

Resumen

En la Estación Experimental de Pastos y Forrajes Indio Hatuey se realiza el proceso de gestión de los contratos de manera manual, entonces el proceso se hace muy ineficiente a la hora de consultar datos de un contrato en específico, o de consultar los que estén próximos a vencer. Se desea realizar una aplicación Web que permita realizar todo este proceso de gestión de manera informatizada, que permita hacer este tipo de consultas y que notifique en ciertos casos a los administradores del sistema.

Abstract

At the Indio Hatuey Pasture and Forage Experimental Station, the contract management process is done manually, so the process becomes very inefficient when consulting data of a specific contract, or consulting those that are about to expire. We would like to develop a Web application that allows to carry out this management process in a computerized way, that allows to make this type of queries and that notifies in certain cases to the system administrators.

Índice

Introducción

Desde los inicios del proceso revolucionario la Administración Central del Estado cubano encargó al naciente Ministerio de la Agricultura (MINAG) proponer e implantar la política sobre el uso, tenencia y explotación sostenible de la superficie agrícola de Cuba; la producción agropecuaria y forestal para la satisfacción de las necesidades alimentarias de la población, la industria y la exportación. En la ejecución de esta política participan 1 400 000 personas, de las cuales el 20 % son mujeres, 24 000 graduados de nivel superior, 75 000 de nivel medio. En la esfera científico-técnica hay 2 626 especialistas y técnicos, de ellos 625 investigadores (Ecured, Ministerio de la Agricultura, s.f.).

El desarrollo de la ganadería constituye una rama importante del MINAG, a partir de 1969, por indicación expresa del Comandante en Jefe se creó la primera Microestación de Pastos, con el objetivo fomentar la investigación científica y el desarrollo de pastos y forrajes para la alimentración del ganado. A la par, se fueron creando estaciones a lo largo del país para la introducción y evaluación de pastos y forrajes en distintas regiones del país. De esta manera se crea la Estación Experimental de Pastos y Forrajes Indio Hatuey en el municipio de Perico de la provincia de Matanzas, la cual inicialmente estuvo adscripta a la Universidad de Matanzas “Camilo Cienfuegos”.

Estas estaciones de pastos y forrajes están dedicadas desarrollar investigaciones con pastos y forrajes para contribuir a la sostenibilidad de la producción ganadera del país. La transferencia de las tecnologías desarrolladas en todas las estaciones del país llevan a cabo mediante contratos con las empresas ganaderas del territorio, política esta que se alinea con el Título II, Artículo 21 de la actual Constitución de la República de Cuba, donde refiere: “El Estado promueve el avance de la ciencia, la tecnología y la innovación como elementos imprescindibles para el desarrollo económico y social. Igualmente implementa formas de organización, financiamiento y gestión de la actividad científica; propicia la introducción sistemática y acelerada de sus resultados en los procesos productivos y de servicios, mediante el marco institucional y regulatorio correspondiente”. (Constitución de la República de Cuba [Const.], 2019). (cubano, 2022)

Tal y como acontece durante las últimas décadas, el desarrollo tecnológico ha tenido una alta incidencia en la modernización y eficiencia de todos los sectores de la sociedad. La dirección del país en la Resolución Económica del V congreso del Partido Comunista de Cuba (PCC), celebrado en 1997, establece: “El país debe encaminarse resueltamente a la modernización informática mediante un programa integral que involucre a las organizaciones que deben proveer los recursos materiales, financieros e intelectuales y a las entidades económicas, políticas y sociales que deben traducirlos en más y mejores productos y servicios (…)”. En julio de 2017 el Ministerio de las Comunicaciones de la

República de Cuba publicó Política Integral para el Perfeccionamiento de la Informatización de la Sociedad en Cuba, dando cumplimiento a varios lineamientos de la política económica y social aprobados en el 8vo Congreso del PCC.

Es vital para la transferencia tecnológica de las estaciones de Pastos y Forrajes del país la concepción y ejecución de contratos con empresas y productores dedicados a la ganadería, en especial en la Estación Experimental Indio Hatuey constatamos que este proceso se realiza de manera manual, provocando pérdidas de tiempo en todo el proceso de comercialización. Las búsquedas realizadas en Internet mostraron la existencia de varios softwares que permiten automatizar este tipo de procesos, pero la mayoría eran de pago o no se adecuaban al flujo deseado para este tipo de empresa. También mediante entrevistas e intercambios de trabajo con el comercial y otros directivos de la unidad de Pastos y Forrajes de Indio Hatuey pudimos verficar que no existe aplicación alguna en el resto de las entidades del país este tipo. Por ello constituye una necesidad desarrollar una plataforma Web que facilite la gestión del proceso de contratación que se lleva a cabo en la estación Experimental de Pastos y Forrajes Indio Hatuey de Matanzas.

Por lo anteriormente planteado surge como **problema de investigación**: ¿Cómo mejorar desde la informática la eficiencia del proceso de contratación que desarrolla la estación Experimental de Pastos y Forrajes Indio Hatuey de Matanzas?

Atendiendo al problema enunciado se asume como **objeto de estudio** el proceso de contratación y como **campo de acción** la informatización del proceso de contratación que desarrolla la Estación Experimental de Pastos y Forrajes Indio Hatuey de Matanzas.

El **objetivo general** de la presente investigación consiste en desarrollar una aplicación Web para la gestión de contratos de la Estación Experimental de Pastos y Forrajes Indio Hatuey de Matanzas que garantice la eficiencia del proceso.

Idea a defender: El desarrollo de una aplicación Web para gestionar de forma rápida y eficiente la ejecución de contratos permitrá ahorrar tiempo, humanizar el trabajo, establecer alertas sobre la caducidad de contratos y reducir la posibilidad de errores en la codificación de los contratos.

Se han determinado las siguientes Tareas:

- Caracterización de los procesos de gestión de información de la codificación de los contratos.
- Diagnóstico de la situación actual en relación al uso de las tecnologías de la informática dentro de la Estación Experimental de Pastos y Forrajes Indio Hatuey de Matanzas.
- Seleccionar las técnicas y herramientas de programación a emplear, así como la metodología a seguir.

- Diseño de un sistema informático que posibilite la gestión de la contratación entre empresas.
- Valoración de especialistas sobre la viabilidad y posibles impactos del sistema propuesto.

Métodos Teóricos:

- Método histórico: Se investigó el comportamiento de la gestión de contratos durante los años de existencia de la Estación de Pastos y Forrajes.
- Histórico lógico: Fueron utilizados para el estudio crítico de los trabajos anteriores en esta temática.

Métodos Empíricos:

- Entrevista y Encuesta: Se entrevistaron a los máximos encargados de la gestión de contratos, a la jurídica y algunos representantes de empresas contratadas, para obtener información de interés.
- Análisis de Documentos: Fueron analizados a profundidad todas las normativas y resoluciones vigentes sobre la temática a investigar.

Para exponer los resultados de la investigación, este trabajo de diploma está dividido en una introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos.

Después de esta introducción aparece el Capítulo 1, que contiene una explicación detallada de los referentes teóricos que argumentan la propuesta. Se presenta sobre las tendencias y tecnologías actuales que serán usadas para el desarrollo de la plataforma Web.

El Capítulo 2 describe la propuesta de solución empleando la metodología XP, comenzamos por las histroias de usuarios. No aparecen los requisitos funcionales y no funcionales de la aplicación.

El capítulo 3 muestra el proceso de pruebas, para comprobar que la plataforma cumple con los requisitos del cliente.

CAPÍTULO 1 MARCO TEÓRICO-REFERENCIAL

1.1 Introducción del capítulo

La hostilidad creciente del gobierno de los EEUU contra Cuba impone la necesidad de emplear tecnologías que no estén sujetas a restricciones por parte de dicho gobierno, pues la aplicación que se propone tiene como beneficiaro a una entidad gubernamental, una Estación Experimental de Pastos y Forrajes perteneciente al Ministerio de la Agricultura de Cuba. En este capítulo hacemos un análisis de la selección de las tecnologías empleadas en el trabajo, softwares, frameworks, Sistema Gestor de Base de Datos (SGBD) y otras herramientas a emplear cuya licencia permita su uso desde nuestro país.

1.2 Descripción del proceso

El proceso a informatizar en la actualidad se desarrolla de la siguiente forma. El cliente interesado en contatar algún servicio de la entidad se pone en contacto con el representante de esa rama en la institución, este explica los términos de contratación y una vez llegan a un acuerdo, proceden a la redacción del contrato. Una vez redactado el documento, este es remitido a la jurídica de la empresa, esta revisa el contenido legal del mismo y en caso de no encontrar incongruencias, da su autorización para la firma. Una vez puesto en vigor el contrato, este pasa a ser archivado en la entidad. Para consultar cualquier información referente al contrato es necesario buscar manualmente en el archivo y revisar la información deseada. (Ver Gráfico 1)

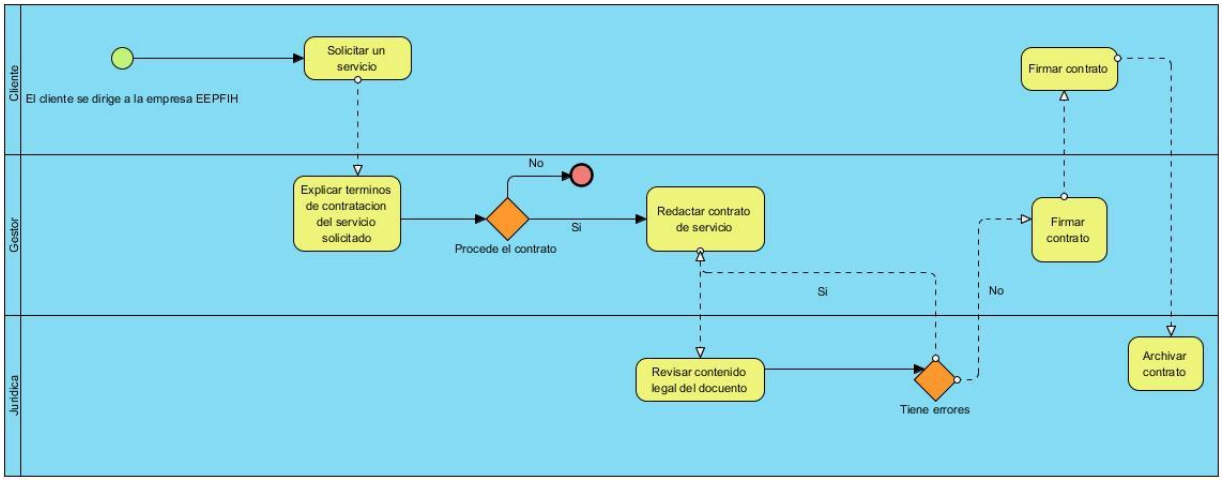


Gráfico 1 - Flujo actual del negocio.

1.2.1 Antecedentes del trabajo

La gestión de contratos es el proceso que coordina todo el ciclo de vida del contrato desde las solicitudes iniciales, cronogramas y fechas de pago hasta su vencimiento y su renovación (Christino, 2020).

La búsqueda en internet permitió constatar la existencia de algunos softwares que informatizan este proceso (asesorias.com, s.f.):

- **Panda Doc**: Panda Doc es un programa de gestión de documentos en general y contratos en particular mediante firma electrónica. La versión más básica cuesta 9 dólares al mes y permite un máximo de 2 usuarios y 60 documentos por año, por lo que está bastante limitada.
- **Icertis**: Es una herramienta de gestión de contratos en la nube. Se trata de una de las plataformas más conocidas en este ámbito, no en vano cuenta con más de 2 millones de usuarios en 90 países que gestionan con Icertis alrededor de 7 millones de contratos.
- **Symfact**: Es otra app que permite una integración sencilla de un software de gestión empresarial. Su módulo de contratos está diseñado para ayudar a la empresa durante todo el proceso de vida del contrato y en la planificación de recursos.

De los softwares revisados se concluye que es una mejor opción desarrollar uno que satisfaga las necesidades de la empresa, debido a que la mayoría o era de pago o no se adecuaba a las necesidades de la entidad. Las entrevistas a los responsables del proceso y algunos beneficiados de este, tambieen iinfluyeron en la decisión.

1.3 Metodologías de desarrollo

En sus inicios apenas se sabía como desarrollar un software, la experiencia acumulada establece que no es tarea fácil. Fue necesario establecer un marco de trabajo para estructurar, planificar y controlar el proceso de desarrollo de sistemas de información. Esto es lo que llamamos metodología de desarrollo del software. Durante los últimos tiempos han aparecido variadas propuestas académicas y de la industria del software para desarrollar softwares. Existe un amplio debate entre dos grandes corrientes: las denominadas metodologías tradicionales, centradas en el control del proceso, con un riguroso seguimiento de las actividades involucradas en ellas.Por otro lado, están las metodologías ágiles, centradas en el factor humano, en la colaboración y participación del cliente en el proceso de desarrollo con iteraciones muy scortas que facilitan la entrega de software. (Oscar, Pedro, & Julio, 2010)

Dado que existen variadas metodologías, tanto de tipo ágil y tradicional, se decidió analizar dos de las principales:

1.3.1 Metodología ágil

Entre las metodologías ágiles más conocidas se encuentran:

- Crystal Clear.
- Dynamic Systems Developmemt Method (DSDM).
- Scrum.
- Extreme Programming (XP).

Extreme Programming (eXtreme Programming , XP) es una metodología que hace el seguimiento constante y la realización de diversas pruebas y pequeños ajustes en proyectos que necesitan agilidad o que están en constante cambio. (SYDLE, 2022)

Se trata de una metodología cuyo objetivo es crear sistemas de alta calidad, basados en una estrecha interacción con los clientes, pruebas constantes y ciclos de desarrollo cortos. (SYDLE, 2022)

Sobre XP:

- Nace en busca de simplificar el desarrollo del software y que se lograra reducir el costo del proyecto.
- No produce demasiada sobrecarga sobre las actividades de desarrollo, y no impide el avance de nuestros proyectos.
- El desarrollo de software es riesgoso y difícil de controlar.
- Se rediseñará todo el tiempo (refactoring), dejando el código siempre en el estado más simple posible.
- XP tiene una debilidad cuando se utiliza en dominios de aplicaciones complejas o situaciones difíciles en la organización: el rol del cliente no refleja los diferentes intereses, habilidades y fuerzas a las que enfrentan los programadores durante el desarrollo de proyectos.

1.3.2 Metodología tradicional

Entre las metodologías tradicionales más conocidas se encuentran:

- Waterfall (cascada).
- Prototipado.
- Espiral.
- Diseño rápido de aplicaciones (RAD).

- Rational Unified Process (UP-Rational).

El Rational Unified Process o Proceso Unificado de Racional. Es un proceso de ingeniería de software que suministra un enfoque para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta y de mayor calidad para satisfacer las necesidades de los usuarios que tienen un cumplimiento al final dentro de un límite de tiempo y presupuesto previsible. Es una metodología de desarrollo iterativo que es enfocada hacia " diagramas de los casos de uso, y manejo de los riesgos y el manejo de la arquitectura" como tal. (Metodología RUP y Ciclo de Vida, 2012)

Sobre RUP:

- Forma disciplinada de asignar tareas y responsabilidades en una empresa de desarrollo (quién hace qué, cuándo y cómo).
- Es un marco del proyecto que describe una clase de los procesos que son iterativos e incrementales.
- Define un manojo entero de las actividades y de los artefactos que usted necesita elegir para construir los propios, proceso individual.
- Es el proceso de desarrollo más general de los existentes actualmente.
- Los procesos de RUP estiman tareas y horario del plan midiendo la velocidad de iteraciones concerniente a sus estimaciones originales. Las iteraciones tempranas de proyectos conducidos RUP se enfocan fuertemente sobre arquitectura del software; la puesta en práctica rápida de características se retrasa hasta que se ha identificado y se ha probado una arquitectura firme.
- Hace especial énfasis en los requisitos y el diseño.

1.3.3 Comparación entre metodologías tradicionales y ágiles.

Aspecto	Tradicional	Ágil
Requisitos	Requieren los requisitos detallados desde el inicio del proyecto. Los requisitos no pueden cambiar	Los requisitos son muy cambiantes. La verdad es que en software los requisitos cambian continuamente, y se requiere de un feedback sobre un

		resultado obtenido para determinar si es lo requerido o no.
Requisitos (funcionalidades innecesarias)	Debido a la recolección inicial de requisitos es frecuente que se soliciten funcionalidades innecesarias.	El enfoque continuo en el valor para el negocio no permite que se incluyan funcionalidades innecesarias
Cambios	Hacer un cambio al alcance requiere de un proceso formal de control de cambios	El cambio es bienvenido en cualquier momento del proyecto
Tiempo	Existe un compromiso respecto al tiempo de entrega del proyecto (no siempre se cumple esta meta)	Existe incertidumbre respecto al tiempo de entrega de todo el producto. Lo cierto es que máximo cada 2 meses (máximo un mes en Scrum) hay entrega de producto de valor para el cliente
Costo	El costo del proyecto es definido para el proyecto	Existe incertidumbre respecto al costo del proyecto. Se invierte en las funcionalidades que más valor le dan al cliente y cíclicamente se avanza hasta que se logre, ya sea: • el producto deseado • se acabe el presupuesto

Documentación	Atención exhaustiva a la documentación.	Solo se genera la documentación que genera valor al cliente y al proyecto
El cliente	El cliente apoya el desarrollo del producto mediante la participación en reuniones.	Involucración directa del cliente en el desarrollo del producto El cliente es parte de equipo.
Iteraciones	Pocas iteraciones que generan gran volumen de información y software para construcción del producto.	Utilización de múltiples iteraciones de desarrollo para aprender y evolucionar el producto
Riesgos	Los riesgos son asumidos por el proveedor	Voluntad del cliente para compartir la responsabilidad en las decisiones y riesgos
Se valora más	El proceso	El individuo y las interacciones de los mismos
La planeación	Requieren un plan detallado desde el inicio del proyecto	Se va planeando a medida que se avanza en el proyecto. Planeación gradual y constante.
El éxito del proyecto	Es dado por el seguimiento del plan	Es dado por la entrega continua de valor y funcionalidad al cliente
Elaboración de entregables	Se generan entregables que requieren mucho tiempo de elaboración.	Se centran en hacer entregables en tiempos cortos con alta calidad inmersa
La retroalimentación del cliente	Es conocida al final, pudiendo generar insatisfacción.	Es constante a lo largo del proyecto

Participación del equipo	Empodera al Gerente de proyecto para el éxito del mismo, este decide si participa de este poder o no al equipo o no.	Empodera al equipo para trabajar de forma creativa e innovadora.
Proceso (Plantillas)	Innumerables plantillas y artefactos para cumplir con el proceso	Pocas plantillas y artefactos (solo los estrictamente necesarios para construir el producto)
Roles	Muchos roles para ejecutar el proyecto	Pocos roles
Arquitectura	Es un ejercicio que se realiza al inicio o en una etapa del proyecto.	Es un ejercicio constante durante el proyecto

Tabla 1 - Comparación entre Metodologías Tradicionales y Ágiles (Londoño, 2014)

1.3.4 Definir Metodología a emplear

En la actualidad se hace más necesario implantar metodologías de desarrollo de software que garanticen simplicidad y velocidad y permitan entregar un producto de calidad en tiempo y con el menor costo posible. Seleccionamos XP para desarrollar nuestra propuesta dada su eficiencia, bajo riesgo, y flexibilidad. Se centra fundamentalmente en potenciar las relaciones interpersonales, promoviendo el trabajo en equipo y el aprendizaje de los desarrolladores. Es adecuada para proyectos con requisitos imprecisos y muy cambiantes, donde existe un alto riesgo técnico. Simplifica el proceso de desarrollo y reduce el costo. Se mantiene en cambio constante, realizando pruebas en cada iteración, comprobando el cumplimiento de los requirimietos.

Algunas de las características principales de esta metodología que la distingue de otras son:

- Pruebas unitarias continuas y frecuentemente repetidas.
- Programación en parejas.
- Interacción frecuente del equipo de programación con el cliente o usuario.
- Corrección de todos los errores antes de añadir una nueva funcionalidad.

- Hacer entregas frecuentes.
- Refactorización del código.
- Propiedad del código compartido: promueve que todo el personal pueda corregir y extender cualquier parte del proyecto.
- Simplicidad en el código.

1.4 Tendencias tecnológicas actuales: herramientas y tecnologías a utilizar

1.4.1 Arquitectura en capas

La arquitectura en capas es una de las más utilizadas, no solo por su simplicidad, sino porque también es utilizada por defecto cuando no estamos seguros que arquitectura debemos de utilizar para nuestra aplicación.

La arquitectura en capas consta en dividir la aplicación en capas, con la intención de que cada capa tenga un rol muy definido, como podría ser, una capa de presentación (User Interface o UI), una capa de reglas de negocio (servicios) y una capa de acceso a datos (Data Access Object o DAO), sin embargo, este estilo arquitectónico no define cuantas capas debe de tener la aplicación, sino más bien, se centra en la separación de la aplicación en capas (Aplica el principio Separación de preocupaciones (Separation of Concerns o SoC)).

En la práctica, la mayoría de las veces este estilo arquitectónico es implementado en 4 capas, presentación, negocio, persistencia y base de datos, sin embargo, es habitual ver que la capa de negocio y persistencia se combinan en una solo capa, sobre todo cuando la lógica de persistencia está incrustada dentro de la capa de negocio.

En una arquitectura en capas, todas las capas se colocan de forma horizontal, de tal forma que cada capa solo puede comunicarse con la capa que está inmediatamente por debajo, por lo que, si una capa quiere comunicarse con otras que están mucho más abajo, tendrán que hacerlo mediante la capa que está inmediatamente por debajo. Por ejemplo, si la capa de presentación requiere consultar la base de datos, tendrá que solicitar la información a la capa de negocio, la cual, a su vez, la solicitará a la capa de persistencia, la que a su vez, la consultará a la base de datos, finalmente, la respuesta retornará en el sentido inverso hasta llegar la capa de presentación.

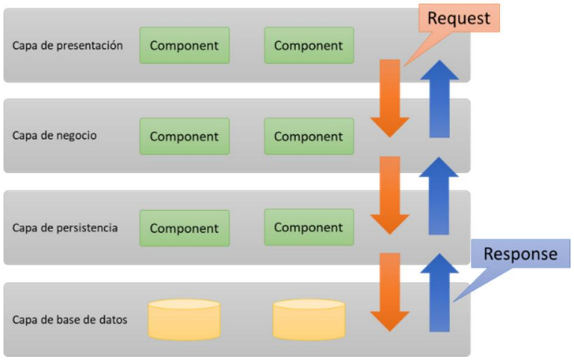


Gráfico 2- Esquema de la arquitectura en capas.

La separación de la aplicación en capas busca cumplir con el principio de separación de preocupaciones, de tal forma que cada capa se encargue una tarea muy definida, por ejemplo, la capa de presentación solo se preocupa por presentar la información de forma agradable al usuario, pero no le interesa de donde viene la información ni la lógica de negocio que hay detrás, en su lugar, solo sabe que existe una capa de negocio que le proporcionará la información. Por otra parte, la capa de negocio solo se encarga de aplicar todas las reglas de negocio y validaciones, pero no le interesa como recuperar los datos, guardarlos o borrarlos, ya que para eso tiene una capa de persistencia que se encarga de eso. Por otro lado, la capa de persistencia es la encargada de comunicarse a la base de datos, crear las instrucciones SQL para consultar, insertar, actualizar o borrar registros y retornarlos en un formato independiente a la base de datos. De esta forma, cada capa se preocupa por una cosa y no le interesa como le haga la capa de abajo para servirle los datos que requiere. (Blancarte, 2020)

1.4.2 Servidor de Bases de Datos

Una base de datos (BD) es una colección o conjunto de datos interrelacionados de carácter persistente en la computadora que pueden ser consultados o modificados por uno o más usuarios o aplicaciones de manera concurrente e independiente. Una base de datos permite almacenar gran número de información de una forma organizada para su futura consulta, realización de búsquedas, nuevo ingreso de datos, etc. Todo esto lo permite realizar de una forma rápida y simple desde un ordenador. El lugar donde se guardan los datos, donde reside la BD junto con los dispositivos asociados se llama hardware.

Entre la base de datos y los usuarios del sistema se encuentran una o dos capas de software, una de estas capas que consiste en una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones se llama Sistema de

Gestión de Base de Datos (SGBD). Un SGBD se compone de un lenguaje de Definición de Datos, de un Lenguaje de Manipulación de Datos y de un Lenguaje de Consulta.

La historia recoge tres generaciones de SGBD: la etapa Pre-Relacional (1950-1972), la etapa Relacional (1972-1972), cuando Edgar Frank Codd introdujo el Sistema Relacional, como su nombre lo indica, basado en la teoría de conjuntos y las relaciones matemáticas entre éstos. Le sigue la etapa actual (2005-actualidad), en esta etapa surgue las bases de datos Not Only Structure Query (NoSQL), las bases de datos NewSQL y también el Big Data. Como afirma el grupo Gartner, aún en nuestros días el mercado de las bases de datos es dominado por los sistemas relacionales (http://ww.gartner.com/), entre los que se encuentra MySQL, que pasa a ser propietario, pero mantiene la versión MariaDB de código abierto y libre. La Gráfico 3, muestra las tres etapas de desarrollo de las bases de datos:

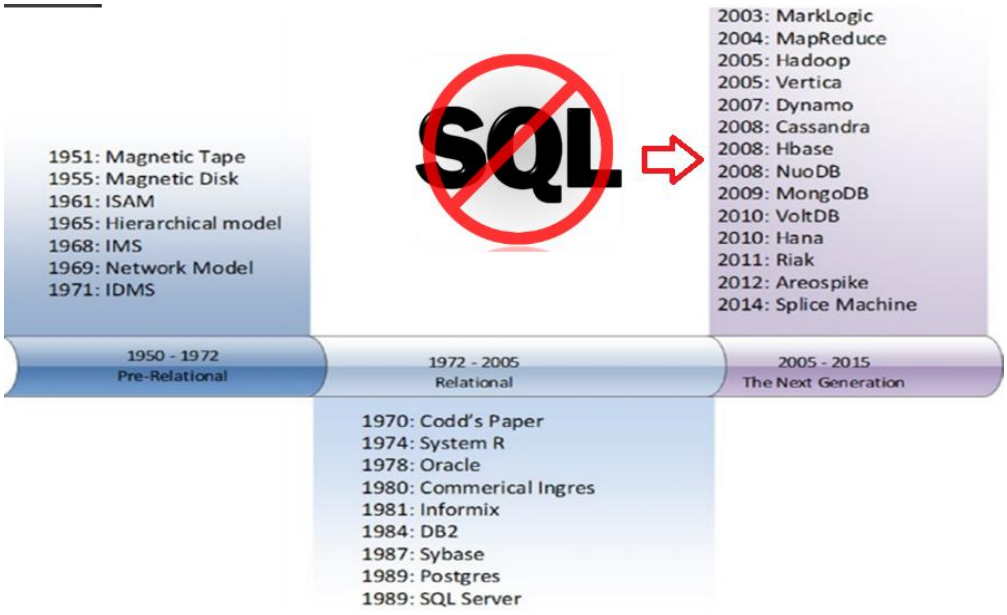


Gráfico 3 - Etapas de desarrollo de las bases de datos.

En la Gráfica 4 se muestran los SGBDs más usados en 2020 (Statista, 2022), se destaca MySQL en el segundo lugar, que constituye nuestra elección en su versión libre, MariaDB.

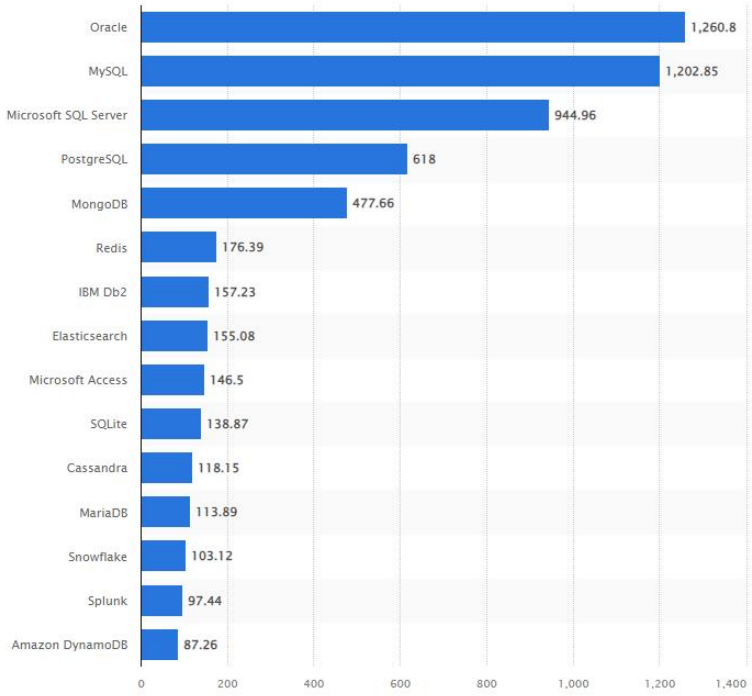


Gráfico 4 - SGBDs más usados en 2020 (Statista, 2022).

MariaDB es un sistema de gestión de bases de datos derivado de MySQL con licencia GPL (General Public License). Es desarrollado por Michael (Monty) Widenius —fundador de MySQL—, la fundación MariaDB y la comunidad de desarrolladores de software libre. Introduce dos motores de almacenamiento nuevos, uno llamado Aria —que reemplaza a MyISAM— y otro llamado XtraDB —en sustitución de InnoDB—. Tiene una alta compatibilidad con MySQL ya que posee las mismas órdenes, interfaces, API y bibliotecas, siendo su objetivo poder cambiar un servidor por otro directamente.

Este SGBD surge a raíz de la compra de Sun Microsystems —compañía que había comprado previamente MySQL AB por parte de Oracle. MariaDB es una bifurcación directa de MySQL que asegura la existencia de una versión de este producto con licencia GPL. Widenius decidió crear esta variante porque estaba convencido de que el único interés de Oracle en MySQL era reducir la competencia que MySQL suponía para Oracle. (MariaDB, s.f.)

Gráfico 2 – Logo de MariaDB.

Ventajas de usar MariaDB (Ortiz, 2020):

- Se incluyen características de seguridad adicionales, lo que hace que la plataforma sea más segura.
- El software MariaDB es más rápido y, como resultado, puede mejorar las velocidades de carga de su sitio.
- Obtendrá acceso a una asistencia al usuario mejor y más disponible.
- Posee licencia libre.

1.4.3 Servidor Web

Un servidor Web es un software que forma parte del servidor y tiene como misión principal devolver información (páginas) cuando recibe peticiones por parte de los usuarios.

En otras palabras, es el software que permite que los usuarios que quieren ver una página Web en su navegador puedan hacerlo.

De acuerdo con las estadísticas de w3tech.com, los servidores Web más utilizados ahora mismo son: Ngynx, Apache y CloudFlare, pero esas no son las únicas opciones, también existen:

- HTTP Apache
- Nginx
- LiteSpeed
- Microsoft Internet Information Services (IIS)
- Sun Java System Web Server
- CloudFlare

Apache es un servidor Web HTTP de código abierto. Está desarrollado y mantenido por una comunidad de usuarios en torno a la Apache Software Foundation. Por su disponibilidad y amplia comunidad de colaboradores fue nuestra elección.

La funcionalidad principal de este servicio Web es servir a los usuarios todos los ficheros necesarios para visualizar la Web. Las solicitudes de los usuarios se hacen normalmente mediante un navegador (Chrome, Firefox, Safari, etc.).

Por ejemplo, cuando un usuario escribe en su navegador dinahosting.com, esa petición llegará a nuestro servidor Apache que mediante el protocolo HTTP este se encargará de facilitarle los textos, imágenes, estilos, etc. que conforman la portada de nuestra Web de forma segura. (dinahosting, s.f.)

Las principales ventajas de usar este el servicio Web son las siguientes:

- De código abierto y gratuito, con una gran comunidad de usuarios.
- Parches de seguridad regulares y actualizados con frecuencia.
- Estructura basada en módulos.
- Multiplataforma. Está disponible en servidores Windows y Linux.
- Personalización mediante .htaccess independiente en cada hosting.
- Compatible con los principales Sistemas de gestión de contenido (Content Management System o CMS) y tiendas online y plataformas e-learning.

1.4.4 Lenguajes de programación

En la actualidad existen muchos lenguajes de programación que nos permiten desarrollar, por medio de un código que cumplen la función de intermediarios entre el desarrollador y el hardware. (Casale, 2016).

Una clasificación general de los lenguajes de programación se refiere a cuánto de cerca o de lejos está del hardware o del programador. Los lenguajes muy próximos a la arquitectura del hardware reciben el nombre de lenguajes de bajo nivel (rígidos y difíciles de aprender). Hay otro grupo de lenguajes de programación que están más cercanos a los programadores y usuarios, denominados lenguajes de alto nivel (son más comprensibles para el lenguaje humano). El Gráfico 6 muestra una representación clara de este concepto (Casale, 2016)

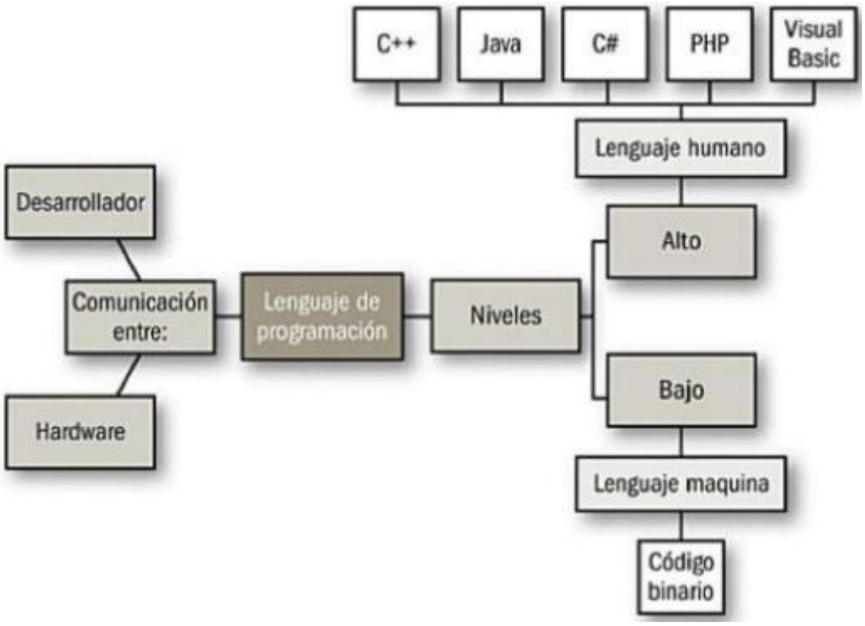


Gráfico 3 – Diagrama de los niveles de los lenfuajes de programación.

En el Gráfico 7 podemos apreciar el TOP 10 de los lenguajes de programación más usados hasta noviembre del 2022 (TIOBE, 2022). Dentro de ese ranking se encuentran PHP, SQL y Javascript; que pertenecen a la selecion de lenguajes para el desarrollo de la aplicación. A continuacion una breve reseña de cada uno de los lenguajes utilizados:

Nov 2022	Nov 2021	Change	Programming Language	Ratings	Change
1	1		Python	17.18%	+5.41%
2	2		C	15.08%	+4.35%
3	3		Java	11.98%	+1.26%
4	4		C++	10.75%	+2.46%
5	5		C#	4.25%	-1.81%
6	6		Visual Basic	4.11%	-1.61%
7	7		JavaScript	2.74%	+0.08%
8	8		Assembly language	2.18%	-0.34%
9	9		SQL	1.82%	-0.30%
10	10		PHP	1.69%	-0.12%

Gráfico 4 – Índice TIOBE, noviembre del 2022.

1.4.4.1 HTML

Lenguaje de Marcas de Hipertexto, del inglés HyperText Markup Language (**HTML**) es el componente más básico de la Web. Define el significado y la estructura del contenido Web. Además de HTML, generalmente se utilizan otras tecnologías para describir la apariencia/presentación de una página Web (CSS) o la funcionalidad/comportamiento (JavaScript). (contributors, HTML: Lenguaje de etiquetas de hipertexto, 2022)

"Hipertexto" hace referencia a los enlaces que conectan páginas Web entre sí, ya sea dentro de un único sitio Web o entre sitios Web. Los enlaces son un aspecto fundamental de la Web. Al subir contenido a Internet y vincularlo a las páginas creadas por otras personas, te conviertes en un participante activo en la «World Wide Web» (Red Informática Mundial). (contributors, HTML: Lenguaje de etiquetas de hipertexto, 2022)

A pesar de no ser reconocido en el índice TIOBE como un lenguaje de programación lo tenemos en cuenta, como muchos autores.

1.4.4.2 CSS

CSS son las siglas en inglés de Cascading Style Sheets, que significa «hojas de esilo en cascada». Es un lenguaje que se usa para estilizar elementos escritos en un lenguaje de marcado como HTML. (B., 2022)

CSS fue desarrollado por W3C (World Wide Web Consortium) en 1996 por una razón muy sencilla. HTML no fue diseñado para tener etiquetas que ayuden a formatear la página. Está hecho solo para escribir el marcado para el sitio.

CSS no es técnicamente una necesidad, pero no querrás tener un sitio que solo tenga **HTML**, ya que se vería completamente desnudo.

1.4.4.3 JavaScript

JavaScript (**JS**) es un lenguaje de programación o de secuencias de comandos que te permite implementar funciones complejas en páginas Web, cada vez que una página Web hace algo más que sentarse allí y mostrar información estática para que la veas, muestra oportunas actualizaciones de contenido, mapas interactivos, animación de Gráficos 2D/3D, desplazamiento de máquinas reproductoras de vídeo, etc., puedes apostar que probablemente JavaScript está involucrado. (contributors, ¿Qué es JavaScript?, 2022)

1.4.4.4 PHP

Preprocesador de hipertexto, Hypertext Preprocessor en inglés (**PHP**) es uno de los lenguajes más populares utilizados para el desarrollo de aplicaciones Web. El lenguaje ha evolucionado para permitir al programador rápidamente desarrollar programas bien formados y libres de errores utilizando procedimientos y técnicas de programacion orientada a objetos. Proporciona la capacidad de utilizar muchas bibliotecas preexistentes de código que vienen con la instalación básica o se pueden instalar dentro del ambiente de PHP. Esto brinda múltiples formas de completar una tarea en particular. Proporciona más flexibilidad que muchos otros idiomas. La facilidad con la que bibliotecas adicionales de el código se puede agregar al entorno es una de las muchas fuerzas impulsoras de su popularidad. (Prettyman, 2020)

1.4.5 Herramientas utilizadas para el desarrollo

1.4.5.1 Visual Studio Code

Visual Studio Code (**VS Code**) es un editor de código fuente desarrollado por Microsoft. Es software libre y multiplataforma, está disponible para Windows, GNU/Linux y macOS. VS

Code tiene una buena integración con Git, cuenta con soporte para depuración de código, y dispone de un sinnúmero de extensiones, que básicamente te da la posibilidad de escribir y ejecutar código en cualquier lenguaje de programación. (Flores, 2022)

Para tener una idea de la popularidad de Visual Studio Code y la aceptación que ha tenido en el mundo de desarrollo, podemos consultar datos. Según una encuesta realizada por Stack Overflow a más de 80,000 desarrolladores en mayo del 2021, Visual Studio Code es el entorno de desarrollo más usado y con mucha diferencia, un 71.06%. En el Gráfico 8, puedes ver el top 10. (Flores, 2022)

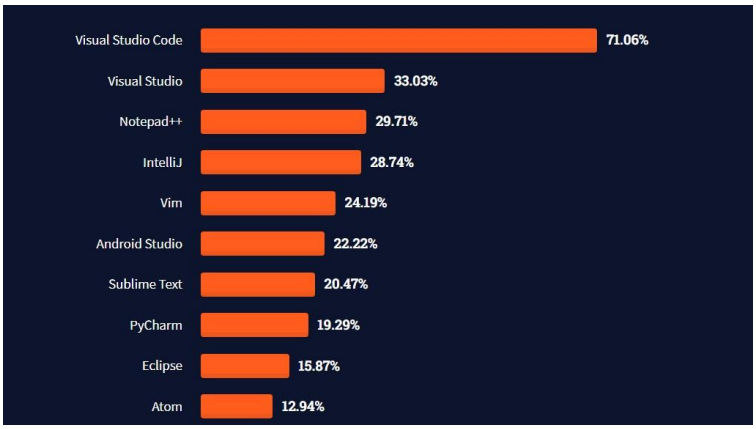


Gráfico 5 – Top 10 entornos de desarrollo (overflow, 2021)

1.4.5.2 MySQL Workbench

MySQL Workbench es un entorno gráfico de diseño de bases de datos, servidores, administración y mantenimiento para el sistema MySQL. Además, esta herramienta gráfica fue desarrollada y distribuida por la compañía de desarrollo de nube y locales Oracle Corporation. (KeepCoding, ¿Qué es MySQL Workbench?, 2022)

La herramienta MySQL Workbench se encuentra disponible para su uso comercial, teniendo total compatibilidad con las versiones del servidor MySQL 5.6 en adelante. MySQL Workbench permite a los desarrolladores, arquitectos de datos y demás clientes diseñar, modelar, gestionar y generar bases de datos de manera visual o gráfica, incluyendo todos los elementos necesarios para realizar modelos con un alto nivel de complejidad. (KeepCoding, ¿Qué es MySQL Workbench?, 2022)

1.4.5.3 Bootstrap 5

Bootstrap es un framework **CSS** desarrollado por Twitter en 2010 utilizado en aplicaciones front-end — es decir, en la pantalla de interfaz con el usuario— para desarrollar aplicaciones que se adaptan a cualquier dispositivo. El framework combina **CSS** y JavaScript para estilizar los elementos de una página **HTML**. Permite mucho más que, simplemente, cambiar el color de los botones y los enlaces. (Author, 2020)

Esta es una herramienta que proporciona interactividad en la página, por lo que ofrece una serie de componentes que facilitan la comunicación con el usuario, como menús de navegación, controles de página, barras de progreso y más. Además de todas las características que ofrece el framework, su principal objetivo es permitir la construcción de sitios Web responsive para dispositivos móviles. (Author, 2020)

1.4.5.4 jQuery

jQuery es un software libre y de código abierto (posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2). Cuenta con un diseño que facilita la navegación por un documento y seleccionar elementos DOM proporcionando a los desarrolladores de aplicaciones Web complementos que agilizan el desarrollo de proyectos. Esto permite a los desarrolladores centrarse en lo importante y crear abstracciones para interacción y animación de bajo nivel, efectos avanzados y widgets temáticos de alto nivel sin invertir tiempo en desarrollar complejos algoritmos y métodos que los controlen desde cero y generando menos código que las aplicaciones hechas con JS puro. Por ese motivo jQuery es muy popular y podemos verlo en muchas páginas Web. (Parada, 2019)

1.4.5.5 DataTables

DataTables es un plugin para **jQuery** que permite agregar en las tablas de tu página Web un aspecto liviano y las funciones para buscar, ordenar y paginar los resultados de forma rápida. Es una herramienta altamente flexible, construida sobre los fundamentos de la mejora progresiva, que permite agregar fácilmente todas estas funciones avanzadas a cualquier tabla **HTML**. (Aguilar, s.f.)

1.4.5.6 SweetAlert

SweetAlert es un plugin de **jQuery** y con el cual podremos dar un aspecto profesional a los mensajes que lancemos a los usuarios acorde a las tendencias actuales. Además, tenemos la posibilidad de configurar el plugin de muchas formas diferentes.

Algo muy buenos es que **SweetAlert** se centra automáticamente en la página Web y se ve muy bien si estás usando una computadora de escritorio, un dispositivo móvil o una tableta. (Estrada, s.f.)

1.4.5.7 Chart

Chart.js es una librería JavaSript open-source ideal para la visualización de datos en gráficos, como puede ser de barras, circular, líneas… donde destaca su sencillez. (Ramos, 2022)

1.5 Conclusiones del capítulo

El concluir el capítulo se constata que no existe software que resuelva la problemática. Por las características del problema a tratar se alinea mejor a una metodología ágil, en este caso XP.

Se seleccionaron las herramientas y lenguajes a utilizar, así como los servidores de base de datos y Web.

CAPÍTULO 2 SOLUCIÓN TEÓRICA DEL PROBLEMA CIENTÍFICO

2.1 Introducción del capítulo

Extreme Programming (XP) es una de las metodologías ágiles más usadas en el desarrollo de software. Debido a su alto grado de interacción con el cliente, su efectividad y simpleza en el proceso. Aspira a producir softwares eficientes y que cumplan con las demandas de los clientes, haciendo a este coprotagonista en todo momento del proceso. Para esto se divide en varias fases: Planificación, Diseño, Desarrollo o implementación y Pruebas.

2.2 Captura de requisitos

Requisitos del proyecto describen las condiciones que deben cumplir o las capacidades que deben tener los productos entregables del proyecto para satisfacer un contrato, norma, especificación o cualquier otro documento formalmente impuesto. El análisis de los interesados que incluyen la totalidad de sus necesidades, deseos y expectativas se traducen en requisitos priorizados.

2.2.1 Requisitos funcionales

Los requisitos funcionales obtenidos de las entrevistas con los clientes son los siguientes:

RF1 – Iniciar sesión con roles de “Usuario” y “Administrador”, cada uno con privilegios diferentes.

RF2 – Agregar contratos.

RF3 – Listar contratos organizados por estados.

RF4 – Cambiar de estado los contratos.

RF5 – Revisar automáticamente la vigencia de los contratos.

RF6 – Cambiar automáticamente los contratos a “Próximos a vencer” y “Vencidos” si lo requieren.

RF7 – Enviar correo electrónico automáticamente a los gestores notificando del cambio de estado de los contratos.

2.2.2 Requisitos no funcionales

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones sobre los tiempos de respuesta, el proceso de desarrollo y estándares. Los requisitos no funcionales se aplican al sistema en su totalidad.

Los requisitos no funcionales se dividen en requisitos de software, de hardware, de soporte, usabilidad, confiabilidad, rendimiento y seguridad.

2.2.2.1 Requisitos de software

- Sistema operativo: Windows XP, 7, 8, 10 o cualquier versión posterior.
- Sistema Gestor de Bases de Datos: MariaDB 10.1.38 o cualquier versión anterior.
- Servidor Web: Apache

2.2.2.2 Requisitos de hardware

- El sistema debe funcionar con máquinas con un mínimo de: 1GB RAM, Pentium, 1GHz y 500 MB libres en disco duro.

2.2.2.3 Requisitos de soporte

- Historias de usuario.
- Casos de prueba de aceptación.
- Arquitectura del software

2.2.2.4 Rendimiento

- El sistema debe ser rápido y preciso, con un tiempo de respuesta aproximado de 2 segundos.

2.2.2.5 Seguridad

- El sistema no debe ser vulnerable a ataques cibernéticos.
- Los roles solo deben tener acceso a la información que tiene permiso a ver.
- Las consultas a la base de datos deben ser seguras, previendo problemas en el servidor Web.

2.3 Fase de planificación

En la metodología XP, la fase que da inicio al proceso de desarrollo del software es la planificación. En esta fase se crean las Historias de Usuario (HU) y comienza el proceso de familiarización del equipo de desarrollo con las tecnologías y herramientas que se utilizarán durante la construcción del proyecto.

Estas historias de usuarios se redactan por parte del cliente sin usar un lenguaje técnico, para mostrar una breve descripción del sistema. Pueden ser tanto requisitos funcionales, como no funcionales. Se crea una por cada característica principal y se emplean para hacer estimaciones de tiempo y son posteriormente un elemento fundamental en las pruebas de aceptación.

Las HU redactadas en esta investigación son:

Historia de Usuario	
Número: HU_1	Nombre Historia de Usuario: Diseñar y crear la base de datos
Usuario: Administrador	Iteración asignada: 1
Prioridad en negocio: Alta	Riesgo de desarrollo: Alto
Descripción: Se debe diseñar y crear una base de datos que respalde la estructura de la aplicación y soporte los tipos de datos necesarios.	

Gráfico 6 – HU_1

Historia de Usuario	
Número: HU_2	Nombre Historia de Usuario: Iniciar sesión
Usuario: Administrador	Iteración asignada: 2
Prioridad en negocio: Alta	Riesgo de desarrollo: Alto
Descripción: Se debe ingresar al sitio mediante uno de los roles definidos: Usuario o Administrador; definiendo así los privilegios que tendrá en la aplicación.	

Gráfico 7 – HU_2

Historia de Usuario	
Número: HU_3	Nombre Historia de Usuario: Agregar contrato
Usuario: Administrador	Iteración asignada: 2
Prioridad en negocio: Alta	Riesgo de desarrollo: Alto
Descripción: Agregar un contato al sistema, recogiendo todos los datos necesarios para esta tarea: Número de contrato, Número de cuenta, Gestor, Tipo de contrato y Tipo de cliente; dependiendo de este útimo se deben agregar algunos campos: - Personas naturales: Nombre y Apellidos - Trabajador por cuenta propia: Nombre, Apellidos y Fecha de aprobación - Cooperativa no agropecuaria, MiPyMe, Entidades: Nombre	

Gráfico 8 – HU_3

Historia de Usuario	
Número: HU_4	Nombre Historia de Usuario: Listar contratos
Usuario: Administrador	Iteración asignada: 4
Prioridad en negocio: Media	Riesgo de desarrollo: Bajo
Descripción: Se listarán todos los contratos separados en distintas páginas y agrupados en estas atendiendo al Tipo de cliente en tres tablas.	

Gráfico 9 – HU_4

Historia de Usuario	
Número: HU_5	Nombre Historia de Usuario: Enviar a "Activos" contratos "En gestión"
Usuario: Administrador	Iteración asignada: 5
Prioridad en negocio: Media	Riesgo de desarrollo: Medio
Descripción: En las tablas de los contratos "En gestión", agegar un botón en cada fila para enviar el contrato en cuestión a "Activos", solicitando la Fecha de inicio y la de culminación.	

Gráfico 10 – HU_5

Historia de Usuario	
Número: HU_6	Nombre Historia de Usuario: Enviar a "Pendientes" contratos "En gestión"
Usuario: Administrador	Iteración asignada: 5
Prioridad en negocio: Media	Riesgo de desarrollo: Medio
Descripción: En las tablas de los contratos "En gestión", agegar un botón en cada fila para enviar el contrato en cuestión a "Pendientes".	

Gráfico 11 – HU_6

Historia de Usuario	
Número: HU_7	Nombre Historia de Usuario: Enviar a "Activos" contratos "Pendientes"
Usuario: Administrador	Iteración asignada: 5
Prioridad en negocio: Media	Riesgo de desarrollo: Medio
Descripción: En las tablas de los contratos "En gestión", agegar un botón en cada fila para enviar el contrato en cuestión a "Activos", solicitando la Fecha de inicio y la de culminación.	

Gráfico 12 – HU_7

Historia de Usuario	
Número: HU_8	Nombre Historia de Usuario: Enviar a "Archivados" contratos "Pendientes"
Usuario: Administrador	Iteración asignada: 6
Prioridad en negocio: Media	Riesgo de desarrollo: Medio
Descripción: En las tablas de los contratos "Pendientes", agegar un botón en cada fila para enviar el contrato en cuestión a "Archivados", solicitando las observaciones con respecto al cumplimiento del contrato.	

Gráfico 13 – HU_8

Historia de Usuario	
Número: HU_9	Nombre Historia de Usuario: Enviar a "Archivados" contratos "Vencidos"
Usuario: Administrador	Iteración asignada: 6
Prioridad en negocio: Media	Riesgo de desarrollo: Medio
Descripción: En las tablas de los contratos "Vencidos", agegar un botón en cada fila para enviar el contrato en cuestión a "Archivados", solicitando las observaciones con respecto al cumplimiento del contrato.	

Gráfico 14 – HU_9

Historia de Usuario	
Número: HU_10	Nombre Historia de Usuario: Revisar contratos "Activos"
Usuario: Administrador	Iteración asignada: 3
Prioridad en negocio: Alta	Riesgo de desarrollo: Alto
Descripción: El sistema a las 12:00 PM revisará la tabla de "Activos" y atendiendo a la Fecha de culminación de los contratos, cambiará su estado a "Próximos a vencer" cuando reste un mes para esa fecha y a "Vencidos" cuando alcance esa fecha.	

Gráfico 15 – HU_10

Historia de Usuario	
Número: HU_11	Nombre Historia de Usuario: Enviar correo electrónico a los Gestores
Usuario: Administrador	Iteración asignada: 3
Prioridad en negocio: Alta	Riesgo de desarrollo: Alto
Descripción: Cuando el sistema realiza la revisión descrita en la HU_9, notifica a través de un correo electrónico a todos los Gestores del cambio de estado de un contrato.	

Gráfico 16 – HU_11

Historia de Usuario	
Número: HU_12	Nombre Historia de Usuario: Gestionar notificaciones
Usuario: Administrador	Iteración asignada: 7
Prioridad en negocio: Baja	Riesgo de desarrollo: Bajo
Descripción: Cuando el sistema realiza la revisión descrita en la HU_9, permite observar en una pestaña de notificaciones en el sistema, los cambios realizados en los estados de los contratos. Permitirá elimiar las notificacioes individual y manualmente una vez que el Administrador decida no verla más.	

Gráfico 17 – HU_12

Historia de Usuario	
Número: HU_13	Nombre Historia de Usuario: Mostrar estadísticas
Usuario: Administrador	Iteración asignada: 7
Prioridad en negocio: Baja	Riesgo de desarrollo: Bajo
Descripción: El sistema en la página de inicio mostrará estadísticas relacionadas con la cantidad de contratos por estados. Estas notificaciones se actualizarán automáticamente cada minuto, pero mediante un botón dispuesto en la interfaz se podrán actualizar manualmente.	

Gráfico 18 – HU_13

Cada Historia de usuario representa las distintas funcionalidades que tendrá el sistema. Cada una está compuesta por el número de historia, el nombre, el usuario que las elaboró, la iteración a la que fue asignada su implementación, la prioridad, el riesgo y una breve descripción de la misma. Se definieron 13 historias, 5 de prioridad alta, 6 de prioridad media y 2 de prioridad baja.

Mediante el plan de entregas confeccionado una vez el terminadas por parte del cliente las historias de usuario, se realiza una estimación del tiempo que puede tardar la implementación de cada HU a partir de la elaboración del plan de entregas.

	1ra Iteración 10 días	2da Iteración 15 días	3ra Iteración 15 días	4ta Iteración 10 días	5ta Iteración 10 días	6ta Iteración 10 días	7ma Iteración 10 días
HU_1	X						
HU_2		X					
HU_3		X					
HU_4				X			
HU_5					X		
HU_6					X		
HU_7					X		
HU_8						X	
HU_9						X	
HU_10			X				
HU_11			X				
HU_12							X
HU_13							X

Tabla 2 – Plan de entregas

Se definieron 7 iteraciones, cada una entre los 10 y 15 días. Según la planificación elaborada, debe estar concluida la primera versión de la aplicación en 80 días (aproximadamente 2 meses y medio).

Para determinar las entregas intermedias y la final, se define el plan de iteraciones. Este proceso incorpora cada una de las HU en las iteraciones del proyecto, teniendo en cuenta la prioridad establecida en el plan de entregas.

Iteración	Descripción de la iteración	HU	Duración total
1ra	Se diseñará y se creará la base de datos, teniendo en cuenta todos los datos que se necesitan almacenar.	HU_1	10 días

2da	En esta iteración se implementará el inicio de sesión mediante los dos roles definidos: Usuario y Administrador; definiendo la información y funcionalidades que va a tener en el sitio. El Usuario solamente podrá ver las estadísticas de los contratos, así como la información de los contratos organizados por estados; el Administrador podrá hacer lo anterior descrito, pero además podrá agregar contratos, cambiar el estado de estos y recibir notificaciones. En esta iteración, además se implementará todo lo relacionado con la funcionalidad de agregar con tratos.	HU_2, HU_3	15 días
3ra	En esta iteración se implementará el script que se ejecutará automáticamente en el servidor para cambiar a “Próximos a vencer” y “Vencidos” los contratos que lo requieran. Junto con este cambio de estado, se implementará la función que notificará a los gestores de estos cambios de estados.	HU_10, HU_11	15 días
4ta	En esta iteración se implementarán todas las vistas de los estados, en cada una se listarán los contratos que se encuentren en el estado agrupados en tres tablas atendiendo el Tipo de cliente.	HU_4	10 días
5ta	Se implementarán las funcionalidades de enviar a “Activos” y a “Pendientes” los contratos que se encuentren “En gestión”, así como la de enviar a “Activos” los contratos que estén “En gestión”.	HU_5, HU_6, HU_7	10 días
6ta	Se implementarán las funcionalidades para enviar a “Archivados” los contratos que se encuentren “Pendientes” y “Próximos a vencer”.	HU_8, HU_9	10 días

7ma	Se implementarán las funcionalidades para recibir y eliminar las notificaciones obtenidas en la revisión descrita en la HU_9. Así como las de las estadísticas que se mostrarán en la página de inicio.	HU_12, HU_13	10 días

Tabla 3 – Plan de iteraciones

2.4 Fase de diseño

La metodología XP plantea la utilización de un diseño simple, fácil de implementar, respetando siempre el orden establecido en las iteraciones. Se buscará que sea un código sencillo, con el flujo indispensable para hacer funcionar la historia del usuario y considerando siempre su experiencia.

Esta metodología menciona una práctica llamada recodificación, que consiste en reescribir el código de una funcionalidad las veces que sean necesarias, con la finalidad de optimizarlo.

La metodología plantea que los nombres a usar en el diseño deben ser fáciles de comprender por todo el equipo de desarrollo, para generar un mayor entendimiento entre todos y una mayor fluidez a la hora de trabajar.

2.4.1 Definición de términos

A continuación, definimos algunos términos que se utilizarán en el proyecto:

Estado: es el estado de gestión en que se encuentran los contratos.

Existen 6 estados por los que puede transitar un contrato:

- **En gestión:** es por el primer estado que transitan los contratos, hace referencia al momento en que comienzan las negociaciones entre ambas partes.
- **Pendiente:** es un estado opcional por el que transitan los contratos de ser necesario; si en el proceso de gestión de algún contrato es encontrado algún error por alguna de las partes, este pasa a este estado. Posteriormente se decide si una vez arreglados los errores se activará o se pasará directamente al archivo (estado que más adelante se tratará).

- **Activo:** representa la entrada en vigor de un contrato, se le asigna una fecha de inicio y otra de culminación que definirá el período de permanencia del contrato en este estado.
- **Próximo a vencer:** los contratos pasarán automáticamente a este estado reste un mes para su fecha de culminación.
- **Vencido:** es un estado temporal por el que pasan los contratos una vez alcanzada la fecha de culminación, a la espera de que se introduzcan observaciones con respecto al cumplimiento del mismo.
- **Archivado:** es el estado final y definitivo de cada contrato introducido al sistema.

Rol: define los privilegios que tendrá el usuario en dependencia de las credenciales que utilice para acceder al sistema.

Existen dos roles en el sistema:

- **Usuario:** solo tendrá posibilidad de observar información de los contratos.
- **Administrador:** tendrá control total sobre el flujo del sistema, podrá insertar y cambiar los estados de los contratos.

2.5 Fase de desarrollo o implementación

Comienza la fase de programación. Se trabaja en base a obtener un código de propiedad colectiva, de modo que sea fácil de comprender para todos los desarrolladores y para futuros programadores responsables de mantenimiento y propagación.

2.5.1 Tareas de Ingeniería

Las Historias de Usuario definidas se implementan en la iteración asignada. Las HU se descomponen en tareas de desarrollo o tareas de ingeniería. Las tareas de ingeniería se componen del nombre y el número de la tarea, además del número de la Historia de Usuario a la pertenece y el usuario que la elaboró. Otro dato que se recoge es la iteración a la que fue asignada para su implementación, así como el tipo de tarea a la que se refiere, los programadores responsables de su implementación y una breve descripción de la tarea. A continuación, se detallan algunas de las tareas de ingeniería desarrolladas para implementar la aplicación.

Tarea de ingeniería	
Número tarea: 1	**Número Historia de Usuario:** HU_1
Nombre tarea: Diseñar la base de datos	
Usuario: Administrador	**Iteración asignada:** 1
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se diseña la base de datos, con todas las tablas necesarias para almacenar los datos de la plataforma en el formato correspondiente.	

Gráfico 19 – TI_1

Tarea de ingeniería	
Número tarea: 2	Número Historia de Usuario: HU_1
Nombre tarea: Crear la base de datos	
Usuario: Administrador	Iteración asignada: 1
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se crea la base de datos.	

Gráfico 20 – TI_2

Tarea de ingeniería	
Número tarea: 3	**Número Historia de Usuario:** HU_2
Nombre tarea: Validar credenciales	
Usuario: Administrador	**Iteración asignada:** 2
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se validan las credencales introducidas por el usuario para inicar sesión.	

Gráfico 21 – TI_3

Tarea de ingeniería	
Número tarea: 4	**Número Historia de Usuario:** HU_2
Nombre tarea: Iniciar sesión	
Usuario: Administrador	**Iteración asignada:** 2
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Una vez validadas las credenciales, se inicia la sesión, garantizando que se muestre la información y las funcionalidades otorgadas a cada rol del sistema.	

Gráfico 22 – TI_4

Tarea de ingeniería	
Número tarea: 5	Número Historia de Usuario: HU_3
Nombre tarea: Validar datos del contrato	
Usuario: Administrador	Iteración asignada: 2
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se validan los datos que ingresa el Administrador para agregar un contrato. Las validaciones se realizan con JavaScript y se van haciendo a medida que el Administrador va introduciendo los datos, para una mayor interacción.	

Gráfico 23 – TI_5

El resto de Tareas de ingeniería se pueden encontrar en la sección de los Anexos (Anexo 2 - 18)

2.6 Estimación del costo

Para calcular el costo del proyecto tendremos en cuenta el costo de personal, el costo de hardware y de software.

2.6.1 Costo de personal

Para calcular el costo de personal tendremos en cuenta el total de horas trabajadas por cada integrante del equipo, así como el papel que desempeña en el proceso de desarrollo.

Fase	Categoría	Horas	Coste/hora	Coste
Documentación	Analista	30	200 CUP	6000 CUP
Análisis	Analista	60	200 CUP	12000 CUP
Diseño	Diseñador	200	150 CUP	30000 CUP
Implementación	Programador	300	100 CUP	30000 CUP
Pruebas	Programador	50	100 CUP	5000 CUP
Total		640		83000 CUP

Tabla 4 – Coste de personal

2.6.2 Coste de hardware

Equipo	Coste	Coste de amortizado
CPU	14400 CUP	1440 CUP
Monitor	1920 CUP	192 CUP
Teclado	240 CUP	24 CUP
Mouse	144 CUP	14 CUP
Computadora personal	7200 CUP	720 CUP
Total	23904 CUP	2390 CUP

Tabla 5 – Coste de hardware

2.6.3 Coste de software

Para el desarrollo de la aplicación no se usan softwares de pago, todo el software que se usa es libre. Por lo que no se reportan gastos en el software.

2.6.4 Coste total

Para obtener el coste total del proyecto se suma el coste de personal, el de hardware y el de software.

Tipo de coste	Total
Coste de personal	83000 CUP
Coste de hardware	2390 CUP
Coste de software	0
Total	85390 CUP

Tabla 6 – Coste total

2.7 Fase de pruebas

La metodología XP gira en torno al proceso de pruebas, para garantizar el aumento de la calidad de los sistemas y reducir el número de errores en la etapa de lanzamiento.

Durante todo el desarrollo de la aplicación se realizan pruebas unitarias y de aceptación. Realizando pruebas al finalizar cada funcionalidad para comprobar su correcto funcionamiento y compatibilidad con otros elementos.

Las pruebas unitarias fueron diseñadas por los programadores involucrados en el proyecto en conjunto con el cliente, estas están diseñadas a evaluar si al final de cada iteración se consigue la funcionalidad descrita en las historias de usuarios correspondientes. Estas se realizan al final de cada iteración y se realizan cada vez que un cambio ocurre para garantizar que al finalizar la iteración se cumpla con la funcionalidad deseada.

Se tuvieron en cuenta tres tipos pruebas de no conformidad a la hora de detectar errores: Significativas (S), No significativas (NS) y Recomendaciones (R). Dentro de las S se encuentran las de aplicación (Funcionalidad, validación, excepciones y opciones que no funcionan) y las de documentación (Ortografía, redacción, correspondencia con otra documentación, formato, error técnico.

Las no conformidades detectadas fueron Significativas de tipo validación, ortografía y de funcionalidad. Se detectaron además algunas No significativas y se tomaron algunas Recomendaciones en cuanto a la interfaz.

Todas las pruebas fueron fructíferas pues no se detectaron errores graves que comprometieran el ritmo del proyecto, pero si algunos detalles que influían en la calidad del producto como se presentan a continuación:

- Problemas en la ubicación del logo del sistema dependiendo del tamaño del dispositivo desde donde se accede.
- Incongruencias en los colores.
- Errores en los mensajes de validación en el formulario de Agregar contrato.

2.8 Conclusiones del capítulo

En este capítulo se explican cómo se desarrollaron las fases de planificación, diseño desarrollo o implementación y pruebas.

En la fase de planificación fueron claves las 13 Historias de Usuarios redactadas por el cliente, pues están constituyen el punto de partida del proyecto. En esta etapa del proyecto se definieron también la cantidad de iteraciones (7) y el tiempo que debían tardar cada una (rondando entre 10 y 15 días).

En la fase de diseño se definieron términos que se utilizarán en el proyecto y que su comprensión es de gran importancia por parte de todo el personal.

En la fase de implementación se definieron las tareas de ingeniería y se asignaron a la persona responsable de su desarrollo, también se hizo una breve descripción sobre cada tarea.

Una vez concluida esta etapa se procede a efectuar las pruebas para demostrar que el producto cumple con los requisitos establecidos en un inicio.

CAPÍTULO 3 PROPUESTA DE SOLUCIÓN PRÁCTICA AL PROBLEMA CIENTÍFICO

3.1 Introducción del capítulo

En este capítulo se muestran los resultados obtenidos a partir de la aplicación de las metodologías explicadas en el Capítulo 2. Se muestran, además, las pruebas que se realizaron a la plataforma, que nos permiten conocer el grado de calidad del producto, y comprobar de esta forma si el sistema es capaz de realizar todas las funcionalidades requeridas por el cliente. Se prueban las características más importantes del sistema con el fin de verificar la fiabilidad y seguridad de la plataforma como un todo.

3.2 Interfaz de usuario

La interfaz de la plataforma presenta un diseño minimalista e intuitivo para facilitar y mejorar la experiencia del usuario. La barra de actividades, visible desde cualquier vista de la aplicación, exceptuando el inicio de sesión, muestra y garantiza el acceso a todas las opciones de la plataforma desde cualquier lugar en que esté el usuario, procurando que tenga acceso solo a las opciones a las cuales tiene permiso a acceder, dependiendo del rol por el cual inició sesión en la plataforma.

Predominan colores claros, contrastando con los colores oscuros de la barra de actividades. En la barra superior encontramos el logo de la plataforma y el botón para contraer la barra de actividades, en la parte derecha encontramos un ícono diferenciando el rol por el cual inició sesión en la plataforma, seguido del botón para cerrar la sesión. Esta barra superior es de color verde, respetando el Manual de Identidad del órgano para el cual se desarrolla la plataforma.

3.3 Pruebas al software

En el ciclo de desarrollo de un software la fase de pruebas es una parte integral, ya que permiten identificar defectos, fallas y errores en el mismo. Son una garantía de la calidad del software y representan una revisión final de las especificaciones, del diseño y de la codificación. Es el proceso de ejecución de un programa con la intención de descubrir errores. Los softwares deben probarse desde dos perspectivas: la lógica interna del programa y el cumplimiento de los requisitos del usuario. Para lograr este objetivo se llevan a cabo técnicas de diseño de casos de prueba de “caja blanca” y “caja negra”.

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

Integrando las Historias de Usuario al proceso de pruebas se logra un software más eficaz. Las pruebas son un conjunto de actividades planeadas con anticipación y realizadas de manera sistemática. Una estrategia de pruebas debe incluir tanto pruebas de alto como de bajo nivel. Son parte de la Verificación y Validación incluidas en el aseguramiento de la calidad del software.

Verificación: Comprobar que el software está de acuerdo con su especificación, donde se debe comprobar que satisface las Historias de Usuario.

Validación: El objetivo es asegurar que el software satisface las expectativas del cliente.

El plan de pruebas de software se elabora Para especificar qué elementos o componentes se van a probar, para que el grupo de trabajo pueda realizar el proceso de Validación y Verificación de las Historias de Usuario. Al desarrollar el plan de pruebas se puede obtener información sobre los errores, defectos y fallas que tiene el prototipo, así se realizan las correcciones pertinentes, según el caso y se asegura la calidad del producto que se va a entregar.

En el presente epígrafe, se presentan los procedimientos empleados para la realización de pruebas al sistema una vez concluido su ciclo de desarrollo.

3.4 Plan de pruebas

El plan de pruebas del software se realiza con el objetivo de definir qué elementos se van a probar, para que el grupo de trabajo pueda realizar el proceso de Validación y Verificación de las Historias de Usuario. Además, a través del plan de pruebas se puede continuar con la trazabilidad de los requerimientos de la plataforma, sirviendo de referencia para analizar el porcentaje de avance que se ha logrado hasta cierto momento.

No.	Historia de usuario	Tareas asignadas
1	HU_1	Test de base de datos

2	HU_2	Test de credenciales vacías Test de credenciales incorrectas
3	HU_3	Test de campos del formulario vacíos Test de campos del formulario en formato incorrecto
4	HU_10	Test del script para revisar contratos “Activos”
5	HU_11	Test de correos enviados al cambiar de estado los contratos
6	HU_4	Test de datos correctos en tablas correspondientes
7	HU_5	Test de activar contratos “En gestión”
8	HU_6	Test de enviar a “Pendientes” contratos “En gestión”
9	HU_7	Test de activar contratos “Pendientes”
10	HU_8	Test de archivar contratos “Pendientes”
11	HU_9	Test de archivar contratos “Vencidos”
12	HU_12	Test de listar notificaciones Test de eliminar notificaciones
13	HU_13	Test de estadísticas en tempo real

Tabla 7 – Plan de pruebas

3.5 Pruebas unitarias

Las pruebas unitarias o unit testing son una forma de comprobar que un fragmento de código funciona correctamente. Es un procedimiento más de los que se llevan a cabo dentro de una metodología ágil de trabajo. Las pruebas unitarias consisten en aislar una parte del código y comprobar que funciona a la perfección. Son pequeños test que validan el comportamiento de un objeto y la lógica. Con ellas se detectan antes errores que, sin las pruebas unitarias, no se podrían detectar hasta fases más avanzadas como las pruebas de sistema, de integración e incluso en la beta. Realizar pruebas unitarias con regularidad supone, al final, un ahorro de tiempo y dinero. (yeeply, 2021)

3.6 Pruebas de integración

En las pruebas de integración se examinan las interfaces entre grupos de componentes o subsistemas para asegurar que son llamados cuando es necesario y que los datos o mensajes que se transmiten son los requeridos. (Cillero, 2020)

El objetivo de las pruebas de integración es verificar el correcto ensamblaje entre los distintos componentes una vez que han sido probados unitariamente con el fin de comprobar que interactúan correctamente a través de sus interfaces, tanto internas como externas, cubren la funcionalidad establecida y se ajustan a los requisitos especificados en las verificaciones correspondientes. (Cillero, 2020)

3.7 Pruebas de seguridad

Las pruebas de seguridad son un tipo de prueba de software que revela vulnerabilidades, amenazas, riesgos en una aplicación de software y previene ataques maliciosos de intrusos. El objetivo principal es identificar las amenazas en el sistema y medir las vulnerabilidades potenciales, para que las amenazas puedan ser detectadas y el sistema no deje de funcionar o no pueda ser explotado. También ayuda a detectar todos los posibles riesgos de seguridad en el sistema y ayuda a los desarrolladores a solucionar los problemas mediante la codificación. (Online, s.f.)

3.8 Pruebas de carga y estrés

Las pruebas de carga de definición generalmente se refieren a las pruebas como un subconjunto del proceso de pruebas de rendimiento del software, que normalmente también incluye varios otros tipos de pruebas, como pruebas de esfuerzo, pruebas de remojo, pruebas de picos, pruebas de resistencia, pruebas de volumen y pruebas de escalabilidad, entre otros tipos de pruebas. (LoadView, s.f.)

Las pruebas de carga simulan escenarios reales en sus sitios, aplicaciones y sistemas. A través de la información recopilada durante y después de las pruebas de carga, los desarrolladores pueden medir los límites y obtener información sobre las métricas que pueden ayudar a responder preguntas como las siguientes (LoadView, s.f.):

- ¿Cómo afectará el número de usuarios al rendimiento?
- ¿Cuántos usuarios simultáneos puede manejar mi sitio Web, aplicación o sistema?
- ¿Dónde están los cuellos de botella?
- ¿Cuántas transacciones podemos manejar durante un período específico?
- ¿Cuál es el punto de quiebre? ¿Cuándo me quedo sin recursos?

Las pruebas de estrés (stress test) son uno de los diferentes tipos de pruebas de carga. Mientras que las pruebas de carga se ocupan principalmente de evaluar el rendimiento de los sistemas, el propósito de las pruebas de estrés es evaluar la disponibilidad y la estabilidad del sistema bajo una carga pesada.

3.9 Pruebas funcionales o caja negra

Las pruebas de caja negra, conocidas también como black box testing, pueden definirse como una técnica donde se busca la verificación de las funcionalidades del software o aplicación analizada, sin tomar como referente la estructura del código interno, las rutas de tipo internas ni la información referente a la implementación. Esto quiere decir que la prueba se lleva a cabo con desconocimiento del funcionamiento del sistema interno, debido a que se enfoca en la entrada y salida de un software, tomando como base sus especificaciones y requisitos. De manera que se puede asegurar que el objetivo de las pruebas de caja negra está relacionado con la validación de los recursos funcionales del software o aplicación que se busca examinar. (KeepCoding, ¿Qué son las pruebas de caja negra?, 2022)

Estas se enfocan en buscar:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

Caso de Uso	Iniciar sesión
Caso de Prueba	Iniciar sesión con campos vacíos.
Resultado Esperado	Se envía un error al usuario de que el campo no puede estar vacío.
Resultado Obtenido	Se produce un error y se emite una advertencia.
% Cumplimiento	100 %

Tabla 8 – Prueba funcional 1

Caso de Uso	Iniciar sesión
Caso de Prueba	Iniciar sesión con credenciales incorrectas.
Resultado Esperado	Se envía un error al usuario de que las credenciales son incorrectas.
Resultado Obtenido	Se produce un error y se recibe una notificación de que las credenciales son incorrectas.
% Cumplimiento	100 %

Tabla 9 – Prueba funcional 2

Caso de Uso	Agregar contrato
Caso de Prueba	Agregar contrato con campos vacíos.
Resultado Esperado	Se envía un error al usuario de que los campos no pueden estar vacíos.
Resultado Obtenido	Se produce un error y se emite una advertencia.
% Cumplimiento	100 %

Tabla 10 – Prueba funcional 3

Caso de Uso	Agregar contrato
Caso de Prueba	Agregar contrato con Número de contrato que ya exista.
Resultado Esperado	Se envía un error al usuario de que existe un contrato con ese Número.
Resultado Obtenido	Se produce un error y se emite una advertencia notificando que existe un contrato con ese Número.
% Cumplimiento	100 %

Tabla 11 – Prueba funcional 4

Caso de Uso	Listar contratos
Caso de Prueba	Listar los contratos que se encuentren “En gestión”
Resultado Esperado	Se muestra al usuario en la vista, una tabla con todos los datos de los contratos “En gestión”
Resultado Obtenido	Se muestra una tabla con todos los datos de los contratos “En gestión” existentes.
% Cumplimiento	100 %

Tabla 12 – Prueba funcional 5

3.10 Pruebas de aceptación

Las pruebas de aceptación son las últimas pruebas realizadas donde el cliente prueba el software y verifica que cumpla con sus expectativas. Estas pruebas generalmente son funcionales y se basan en los requisitos definidos por el cliente y deben hacerse antes de la

salida a producción. Las pruebas de aceptación son fundamentales por lo cual deben incluirse obligatoriamente en el plan de pruebas de software. Estas pruebas se realizan una vez que ya se ha probado que cada módulo funciona bien por separado, que el software realice las funciones esperadas y que todos los módulos se integran correctamente. (Training, 2020)

El test de aceptación termina de definir el nivel de calidad de la aplicación y le permite conocer al equipo qué tan bien supo interpretar correctamente los requerimientos del usuario. Es el cliente quien tendrá la decisión final de aprobar o no el producto, como también de solicitar modificaciones. (Training, 2020)

Se consideró con el cliente las pruebas de aceptación a realizar, dependiendo de la prioridad para el negocio y se obtuvieron () pruebas, a continuación, se describe la estructura de las mismas:

Campos descritos:

- **Número del caso de prueba:** Sirve para identificar la prueba realizada y al mismo tiempo sugiere su nombre.
- **Número de Historia de Usuario:** Nombre de la Historia de Usuario a la que hace referencia
- **Nombre del Caso de Prueba:** Nombre de la prueba
- **Descripción:** Describe la funcionalidad que se desea probar
- **Condiciones de Ejecución:** Condiciones a cumplirse para poder llevar a cabo el caso de prueba
- **Entradas / Pasos de ejecución:** Descripción de cada uno de los pasos durante el desarrollo de la prueba
- **Resultado esperado:** Descripción del resultado que se espera obtener con la prueba realizada
- **Evaluación de la prueba:** Evaluación emitida acorde al resultado de la prueba realizada

Las evaluaciones pueden ser:

- Satisfactorio: El resultado es completamente el esperado por el usuario
- Parcialmente Bien: El resultado no es completamente el esperado por el cliente o usuario de la aplicación
- Mal: El resultado de la prueba genera un error de codificación en la aplicación o muestra como resultado elementos no deseados o fuera de contextos.

Pasos a seguir para realizar pruebas de aceptación:

- Redacción de los casos de prueba teniendo en cuenta el orden de las HU y los niveles de prioridad dados anteriormente a las funcionalidades.
- Planificación con el cliente de cuales pruebas se llevarán a cabo y en qué momento.
- Completar cada uno de los campos de las tablas de las pruebas de aceptación con el resultado de la prueba.

Las pruebas de aceptación obtenidas son:

Prueba de Aceptación	
Número del caso de prueba: 1	**Número de Historia de Usuario: 2**
Nombre del Caso de Prueba: Iniciar Sesión	
Descripción: Comprobar que el sistema permite el acceso al sitio Web solo a los usuarios que ingresen las credenciales correctamente. El sistema debe verificar que los datos obligatorios para entrar al sistema sean correctos y no estén vacíos. En estos casos se muestra un mensaje de error impidiendo el acceso al sitio hasta que los datos cumplan con sus requerimientos.	
Condiciones de ejecución: Que el sistema esté conectado y tenga acceso a las bases de datos, comprobando además el rol que tenga asignado este usuario	
Entrada / Pasos de ejecución: El usuario introduce su nombre de usuario y su contraseña y pulsa el botón “Entrar”.	
Resultado esperado: El sistema debe aceptar y entrar al sistema en el caso de que el nombre de usuario y la contraseña introducidos coincida con las credenciales almacenadas en la base de datos, de intentar introducir un valor erróneo o nulo deberá mostrar una alerta.	
Evaluación de la prueba: Satisfactorio	

Tabla 13 – Prueba de Aceptación 1

Prueba de Aceptación	
Número del caso de prueba: 2	**Número de Historia de Usuario: 3**
Nombre del Caso de Prueba: Agregar campos del contrato en formato incorrecto.	
Descripción: Comprobar que el sistema no permite agregar ningún contrato con campos en formato incorrecto.	
Condiciones de ejecución: Que sea un usuario con rol "Administrador" el que tenga la sesión iniciada.	
Entrada / Pasos de ejecución: El usuario rellena con datos en formato incorrecto el formulario para agregar un contrato.	
Resultado esperado: El sistema debe notificar del error en el formato de los datos introducidos.	
Evaluación de la prueba: Satisfactorio	

Tabla 14 – Prueba de Aceptación 2

Prueba de Aceptación	
Número del caso de prueba: 3	**Número de Historia de Usuario: 3**
Nombre del Caso de Prueba: Agregar contrato con campos vacíos.	
Descripción: Comprobar que el sistema no permite agregar ningún contrato con campos vacíos.	
Condiciones de ejecución: Que sea un usuario con rol "Administrador" el que tenga la sesión iniciada.	
Entrada / Pasos de ejecución: El usuario deja campos del formulario para agregar un contrato vacíos.	
Resultado esperado: El sistema debe notificar los campos obligatorios a rellenar.	
Evaluación de la prueba: Satisfactorio	

Tabla 15 – Prueba de Aceptación 3

Prueba de Aceptación	
Número del caso de prueba: 4	**Número de Historia de Usuario: 10**
Nombre del Caso de Prueba: Revisar contratos “Activos”	
Descripción: Comprobar que el script que permite cambiar automáticamente a “Próximos a vencer” y “Vencidos” los contratos está funcionando correctamente.	
Condiciones de ejecución: Tener acceso al servidor Web donde se aloja la plataforma.	
Entrada / Pasos de ejecución: Ejecutar el archivo cron.bat que se encuentra en \app\tareas.	
Resultado esperado: Los contratos que cumplan las condiciones establecidas se deben haber movido a “Próximos a vencer” y a “Vencidos”.	
Evaluación de la prueba: Satisfactorio	

Tabla 16 – Prueba de Aceptación 4

Prueba de Aceptación	
Número del caso de prueba: 5	**Número de Historia de Usuario: 11**
Nombre del Caso de Prueba: Enviar correos a los gestores	
Descripción: Comprobar que al ejecutar el script descrito en la HU_9 se envíen los respectivos contratos a los gestores.	
Condiciones de ejecución: Tener acceso al servidor Web donde se aloja la plataforma.	
Entrada / Pasos de ejecución: Ejecutar el archivo cron.bat que se encuentra en \app\tareas.	
Resultado esperado: Si se cambió el estado de algún contrato la lista de gestores debió haber recibido un correo de notificación.	
Evaluación de la prueba: Satisfactorio	

Tabla 17 – Prueba de Aceptación 5

Prueba de Aceptación	
Número del caso de prueba: 6	**Número de Historia de Usuario: 4**
Nombre del Caso de Prueba: Listar contratos	
Descripción: Comprobar que las tablas donde se listan los contratos funcionen correctamente	
Condiciones de ejecución: Tener la sesión iniciada en cualquiera de los roles disponibles.	
Entrada / Pasos de ejecución: Entrar a todas las vistas donde se listan los contratos. Los vínculos para acceder a estas vistas se encuentran en la barra de actividades a la izquierda del todo. Las vistas están organizadas por los estados de los contratos.	
Resultado esperado: Por cada estado por el que transitan los contratos se debe mostrar en su vista correspondiente tres tablas, una por cada tipo de cliente, y en ellas los contratos deben aparecer listados con todos los datos disponibles.	
Evaluación de la prueba: Satisfactorio	

Tabla 18 – Prueba de Aceptación 6

3.11 Conclusiones del capítulo

En este capítulo se comprueba que la planificación presentada en el Capítulo 2 fue acertada, porque permitió el desarrollo de la aplicación en el tiempo estimado, cumpliendo con las exigencias del cliente. Confirmando que las pruebas realizadas mediante el uso de las técnicas descritas anteriormente fueron de gran importancia para demostrar el correcto funcionamiento del software y el cumplimiento de los requisitos del cliente.

Como resultado final se obtiene una aplicación Web con una interfaz intuitiva y fácil de usar, cumpliendo con las funcionalidades requeridas y cumpliendo las expectativas del cliente.

Conclusiones

- Se comprendió a fondo el proceso de contratación, sus etapas y datos necesarios para su realización.
- En la Estación Experimental de pastos y Forrajes Indio Hatuey existe un bajo nivel de informatización en sus procesos, derrochando una enorme cantidad de tiempos y recursos.
- Se definieron las técnicas, herramientas y lenguajes de programación a utilizar en el proceso de desarrollo.
- Se diseñó una aplicación Web capaz de responder al proceso de contratación y cumpliendo todos los requisitos impuestos por el cliente.
- Por la posibilidad de ser escalable, el proceso podrá significar un avance en muchas entidades con similar estructura y necesidades.

Recomendaciones

Se recomienda a la entidad:

- Desplegar esta aplicación y pensar en expandir su uso a otras Estaciones de Pastos y Forrajes.
- Continuar trabajando en la interfaz de la aplicación.
- Agregar funcionalidades como: exportar tablas a PDF o Excel.

Bibliografía

Aguilar, J. (s.f.). *DataTables jQuery plugin*. Recuperado el 2022, de Jose Aguilar: https://www.jose-aguilar.com/blog/datatables-jquery-plugin/

asesorias.com. (s.f.). Recuperado el 1 de 7 de 2022, de Software de gestión de contratos. Te ayudamos a elegir el mejor: https://asesorias.com/empresas/programas-gratis/software-gestion-contratos/

Author, G. (12 de 4 de 2020). *Bootstrap: guía para principiantes de qué es, por qué y cómo usarlo*. Recuperado el 2022, de Rock content: https://rockcontent.com/es/blog/bootstrap/

B., G. (12 de 7 de 2022). *¿Qué es CSS?* Recuperado el 11 de 2022, de Hostinguer Tutoriales: https://www.hostinger.es/tutoriales/que-es-css

Blancarte, O. (2020). *Arquitectura en Capas*. Recuperado el 2022, de Reactive programming: https://reactiveprogramming.io/blog/es/estilos-arquitectonicos/capas

Casale, J. C. (2 de 7 de 2016). *Google Libros*. Recuperado el 2022, de Google Libros: https://books.google.com.py/books?id=MKacDAAAQBAJ&printsec=frontcover&hl=es#v=onepage&q&f=false

Christino, C. (7 de diciembre de 2020). *Gestión de Contratos: Cómo hacer que su proceso sea más eficiente*. Recuperado el 1 de julio de 2022, de Excellence Blog: https://blog.softexpert.com/es/gestion-contratos-eficiente/

Cillero, M. (2020). *Pruebas de Integración*. Recuperado el 2022, de manuel.cillero.es: https://manuel.cillero.es/doc/metodologia/metrica-3/tecnicas/pruebas/integracion/

contributors, M. (11 de 11 de 2022). *¿Qué es JavaScript?* Recuperado el 11 de 2022, de MDN web docs: https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript

contributors, M. (7 de 11 de 2022). *HTML: Lenguaje de etiquetas de hipertexto*. Recuperado el 11 de 2022, de MDN web docs: https://developer.mozilla.org/es/docs/Web/HTML

cubano, P. (6 de 2022). *Contitución de la República.* Recuperado el 11 de 2022, de Parlamento cubano: https://www.parlamentocubano.gob.cu/sites/default/files/documento/2022-06/Constituci%C3%B3n-de-la-Rep%C3%BAblica-de-Cuba-1976.pdf

dinahosting. (s.f.). *¿Qué es Apache y para qué sirve?* Recuperado el 2022, de dinahosting: https://dinahosting.com/ayuda/que-es-apache-y-para-que-sirve/

Ecured. (2018). *Metodología Ágil de Desarrollo SXP*. Recuperado el 2022, de Ecured: https://www.ecured.cu/index.php?title=Metodologia_Agil_de_Desarrollo_SXP&oldid=138 9913

Ecured. (s.f.). *Ministerio de la Agricultura*. Recuperado el 2022, de Ecured: https://www.ecured.cu/Ministerio_de_la_Agricultura

Estrada. (s.f.). *Mensajes de notificación profesionales al usuario con jQuery y SweetAlert*. Recuperado el 2022, de Estrada web group: https://estradawebgroup.com/Post/Mensajes-de-notificacion-profesionales-al-usuario-con-jQuery-y-SweetAlert/4252

Flores, F. (22 de 7 de 2022). *Open Webinars*. Recuperado el 11 de 2022, de Qué es Visual Studio Code y qué ventajas ofrece: https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/

KeepCoding, R. (11 de 4 de 2022). *¿Qué es MySQL Workbench?* Recuperado el 2022, de Keep coding: https://keepcoding.io/blog/que-es-mysql-workbench/

KeepCoding, R. (3 de 6 de 2022). *¿Qué son las pruebas de caja negra?* Recuperado el 11 de 2022, de KeepCoding: https://keepcoding.io/blog/que-son-las-pruebas-de-caja-negra/#Que_son_las_pruebas_de_caja_negra

LoadView. (s.f.). *Pruebas de carga*. Recuperado el 2022, de LoadView: https://www.loadview-testing.com/es/pruebas-de-carga/

Londoño, J. H. (14 de 7 de 2014). *Lecciones Aprendidas en Desarrollo de Software*. Recuperado el 2022, de Lecciones Aprendidas: http://www.lecciones-aprendidas.info/2014/07/tabla-comparativa-entre-metodologias.html

MariaDB. (s.f.). Recuperado el 2022, de MariaDB: https://mariadb.com/kb/en

Metodología RUP y Ciclo de Vida. (3 de 7 de 2012). Recuperado el 2022, de Metodología RUP: http://rupmetodologia.blogspot.com/

Online, E. (s.f.). *¿Qué es una prueba de seguridad? Tipos con ejemplo*. Recuperado el 2022, de Ebooks Online: https://ebooksonline.es/que-es-una-prueba-de-seguridad-tipos-con-ejemplo/#%C2%BFQue_es_una_prueba_de_seguridad

Ortiz, A. E. (16 de 6 de 2020). *3 ventajas de usar MariaDB sobre MySQL*. Recuperado el 2022, de Host Dime Blog: https://www.hostdime.com.ar/blog/3-ventajas-de-usar-mariadb-sobre-mysql/

Oscar, T. G., Pedro, R., & Julio, S. (31 de 12 de 2010). *Criterios de selección de metodologías de desarrollo de software*. Recuperado el 2022, de Revistas de investigación UNMSM: https://revistasinvestigacion.unmsm.edu.pe/index.php/idata/article/view/6191/5386

overflow, I. S. (5 de 2021). *Developer Survey*. Recuperado el 2022, de Stack overflow: https://insights.stackoverflow.com/survey/2021#most-popular-technologies-new-collab-tools

Parada, M. (31 de 10 de 2019). *Qué es jQuery*. Recuperado el 2022, de Open Webinars: https://openwebinars.net/blog/que-es-jquery/

Prettyman, S. (2020). *Learn PHP 8 Using MySQL, JavaScript, CSS3, and HTML5* (Second Edition ed.). Recuperado el 2022

Ramos, L. C. (1 de 7 de 2022). *Cómo añadir gráficos en tu web con Chart.js*. Recuperado el 11 de 2022, de Adictos al trabajo: https://www.adictosaltrabajo.com/2022/07/01/como-anadir-graficos-en-tu-web-con-chart-js/

Statista. (8 de 2022). *Ranking of the most popular database management systems worldwide, as of August 2022*. Recuperado el 9 de 2022, de statista: https://www.statista.com/statistics/809750/worldwide-popularity-ranking-database-management-systems/

SYDLE. (15 de 05 de 2022). *Extreme Programming: ¿qué es y cómo funciona?* Recuperado el 11 de 2022, de SYDLE: https://www.sydle.com/es/blog/extreme-programming-602ee205da4d096809438c9c/

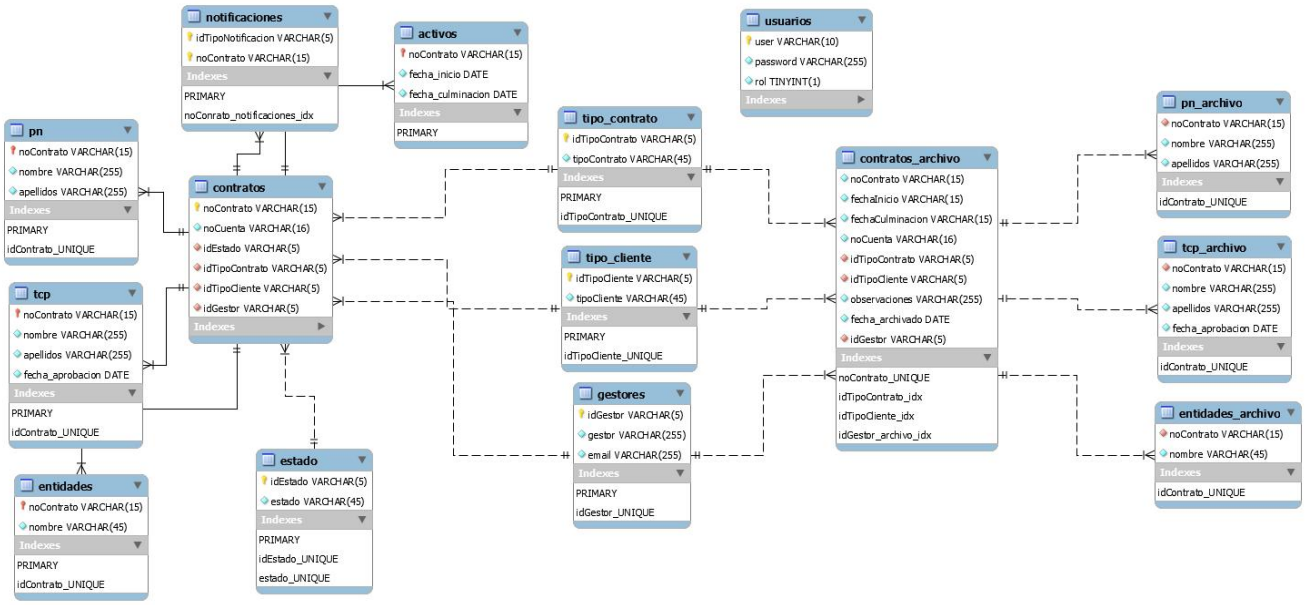
TIOBE. (11 de 2022). *TIOBE Index for November 2022*. Recuperado el 11 de 2022, de TIOBE: https://www.tiobe.com/tiobe-index/

Training, L. A. (11 de 4 de 2020). *Que son las pruebas de aceptación?* Recuperado el 2022, de Linkedin: https://es.linkedin.com/pulse/que-son-las-pruebas-de-aceptaci%C3%B3n-los-andes-training

yeeply. (2021). *¿Qué son las pruebas unitarias y cómo llevar una a cabo?* Recuperado el 2022, de yeeply: https://www.yeeply.com/blog/que-son-pruebas-unitarias/#que

Anexos

Anexo 1- Modelo lógico de la base de datos



Anexo 2 – TI_6

Tarea de ingeniería	
Número tarea: 6	**Número Historia de Usuario:** HU_3
Nombre tarea: Agregar contrato	
Usuario: Administrador	**Iteración asignada:** 2
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Una vez validados los datos del contrato introducidos por el Administrador, se procede a agregar el contrato mediante una peticion Ajax a la capa de Persistencia, esta a su vez envía la solicitud a la capa de Datos la cual interactúa directamente con la base de datos, insertando el contrato en ella.	

Anexo 3 – TI_7

Tarea de ingeniería	
Número tarea: 7	Número Historia de Usuario: HU_10
Nombre tarea: Revisar contratos "Activos"	
Usuario: Administrador	Iteración asignada: 3
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se realiza el script PHP que se ejecutará automáticamente en el servidor a las 12:00 PM todos los días. Este script revisará en la base de datos los contratos "Activos" y pasará a "Próximos a vencer" los contratos los cuales reste un mes para llegar a su Fecha de culminación y a "Vencidos" los que vayan alcanzando esa fecha.	

Anexo 4 – TI_8

Tarea de ingeniería	
Número tarea: 8	Número Historia de Usuario: HU_11
Nombre tarea: Enviar correo	
Usuario: Administrador	Iteración asignada: 3
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se realiza el script PHP que se ejecutará automáticamente en el servidor a las 12:00 PM todos los días. Este script revisará en la base de datos los contratos "Activos" y pasará a "Próximos a vencer" los contratos los cuales reste un mes para llegar a su Fecha de culminación y a "Vencidos" los que vayan alcanzando esa fecha.	

Anexo 5 – TI_9

Tarea de ingeniería	
Número tarea: 9	Número Historia de Usuario: HU_4
Nombre tarea: Listar contratos "En gestión"	
Usuario: Administrador	Iteración asignada: 4
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se implementa la página donde se listan (utilizando el plugin DataTables de jQuery) los contratos "En gestión", agrupados en tres tablas (Personas naturales, Trabajador por cuenta propia y Entidades), atendiendo al Tipo de cliente.	

Anexo 6 – TI_10

Tarea de ingeniería	
Número tarea: 10	Número Historia de Usuario: HU_4
Nombre tarea: Listar contratos "Pendientes"	
Usuario: Administrador	Iteración asignada: 4
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se implementa la página donde se listan (utilizando el plugin DataTables de jQuery) los contratos "Pendientes", agrupados en tres tablas (Personas naturales, Trabajador por cuenta propia y Entidades), atendiendo al Tipo de cliente.	

Anexo 7 – TI_11

Tarea de ingeniería	
Número tarea: 11	Número Historia de Usuario: HU_3
Nombre tarea: Listar contratos "Activos"	
Usuario: Administrador	Iteración asignada: 4
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se implementa la página donde se listan (utilizando el plugin DataTables de jQuery) los contratos "Activos", agrupados en tres tablas (Personas naturales, Trabajador por cuenta propia y Entidades), atendiendo al Tipo de cliente.	

Anexo 8 – TI_12

Tarea de ingeniería	
Número tarea: 12	Número Historia de Usuario: HU_4
Nombre tarea: Listar contratos "Próximos a vencer"	
Usuario: Administrador	Iteración asignada: 4
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se implementa la página donde se listan (utilizando el plugin DataTables de jQuery) los contratos "Próximos a vencer", agrupados en tres tablas (Personas naturales, Trabajador por cuenta propia y Entidades), atendiendo al Tipo de cliente.	

Anexo 9 – TI_13

Tarea de ingeniería	
Número tarea: 13	Número Historia de Usuario: HU_4
Nombre tarea: Listar contratos "Vencidos"	
Usuario: Administrador	Iteración asignada: 4
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se implementa la página donde se listan (utilizando el plugin DataTables de jQuery) los contratos "Vencidos", agrupados en tres tablas (Personas naturales, Trabajador por cuenta propia y Entidades), atendiendo al Tipo de cliente.	

Anexo 10 – TI_14

Tarea de ingeniería	
Número tarea: 14	Número Historia de Usuario: HU_4
Nombre tarea: Listar contratos "Archivados"	
Usuario: Administrador	Iteración asignada: 4
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se implementa la página donde se listan (utilizando el plugin DataTables de jQuery) los contratos "Archivados", agrupados en tres tablas (Personas naturales, Trabajador por cuenta propia y Entidades), atendiendo al Tipo de cliente.	

Anexo 11 – Tl_15

Tarea de ingeniería	
Número tarea: 15	Número Historia de Usuario: HU_5
Nombre tarea: Activar contratos "En gestión"	
Usuario: Administrador	Iteración asignada: 5
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Esta funcionalidad solo estará disponible para el Administrador. Se agrega un botón a cada fila de las tablas para cambiar el estado del contrato de "En gestión" a "Activos". Se implementa la función que permite dicha acción.	

Anexo 12 – Tl_16

Tarea de ingeniería	
Número tarea: 16	Número Historia de Usuario: HU_6
Nombre tarea: Enviar a "Pendientes" contratos "En gestión"	
Usuario: Administrador	Iteración asignada: 5
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Esta funcionalidad solo estará disponible para el Administrador. Se agrega un botón a cada fila de las tablas para cambiar el estado del contrato de "En gestión" a "Pendientes". Se implementa la función que permite dicha acción.	

Anexo 13 – TI_17

Tarea de ingeniería	
Número tarea: 17	Número Historia de Usuario: HU_7
Nombre tarea: Activar contratos "Pendientes"	
Usuario: Administrador	Iteración asignada: 5
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Esta funcionalidad solo estará disponible para el Administrador. Se agrega un botón a cada fila de las tablas para cambiar el estado del contrato de "Pendientes" a "Activos". Se implementa la función que permite dicha acción.	

Anexo 14 – TI_18

Tarea de ingeniería	
Número tarea: 18	Número Historia de Usuario: HU_8
Nombre tarea: Enviar a "Archivados" contratos "Pendientes"	
Usuario: Administrador	Iteración asignada: 6
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Esta funcionalidad solo estará disponible para el Administrador. Se agrega un botón a cada fila de las tablas para cambiar el estado del contrato de "Pendientes" a "Archivados". Se implementa la función que permite dicha acción.	

Anexo 15 – TI_19

Tarea de ingeniería	
Número tarea: 19	Número Historia de Usuario: HU_9
Nombre tarea: Enviar a "Archivados" contratos "Vencidos"	
Usuario: Administrador	Iteración asignada: 6
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Esta funcionalidad solo estará disponible para el Administrador. Se agrega un botón a cada fila de las tablas para cambiar el estado del contrato de "Vencidos" a "Archivados". Se implementa la función que permite dicha acción.	

Anexo 16 – TI_20

Tarea de ingeniería	
Número tarea: 20	Número Historia de Usuario: HU_12
Nombre tarea: Listar notificaciones	
Usuario: Administrador	Iteración asignada: 7
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se implementa la página donde se listan las notificaciones obtenidas de la revisión descrita en la HU_9, cada tipo de notificación con un estilo propio para su fácil identificación.	

Anexo 17 – TI_21

Tarea de ingeniería	
Número tarea: 21	Número Historia de Usuario: HU_12
Nombre tarea: Eliminar notificaciones	
Usuario: Administrador	Iteración asignada: 7
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se implementa la función que permita eliminar las notificaciones una vez que el Administrador considere no sean necesarias.	

Anexo 18 – TI_22

Tarea de ingeniería	
Número tarea: 22	Número Historia de Usuario: HU_13
Nombre tarea: Mostrar estadísticas	
Usuario: Administrador	Iteración asignada: 7
Tipo tarea: Desarrollo	
Programador responsable: Daniel Navarro Machin	
Descripción: Se implementa la función que permita mostrar las notificaciones en la página de inicio.	