Facultad 1

Aplicación web para la gestión de procesos de Recursos Humanos de la empresa LABIOFAM.

Trabajo de diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autor(es): Luis Ernesto Rodríguez Larred

Tutor(es): Ing. Adrián Castillo Chávez

MsC. Grettel Pérez Rey.

Asesor: David Domingo Larred Rodríguez

La Habana, junio de 2021

"Año 63 de la Revolución"

DECLARACIÓN **DE AUTORÍA**

El autor del trabajo de diploma con título "**Aplicación web para la gestión de procesos de Recursos Humanos de la empresa Labiofam"**, concede a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la investigación, con carácter exclusivo. De forma similar se declara como único autor de su contenido. Para que así conste firma la presente a los <día> días del mes de <mes> del año <año>.

Luis Ernesto Rodríguez Larred

_____
Firma del Autor

Adrián Castillo Chávez

_____

_____
Firma del Tutor

Grettel Pérez Rey

_____

_____
Firma del Tutor

DATOS DE CONTACTO

<Curriculum e información de contacto del tutor: nombre y apellidos, títulos académicos, formación de postgrado recibida, lugar de trabajo, responsabilidades laborales asumidas, experiencia profesional, líneas de trabajo y/o investigación, correo electrónico, perfiles en redes profesionales>

<Curriculum e información de contacto del asesor: nombre y apellidos, títulos académicos, formación de postgrado recibida, lugar de trabajo, responsabilidades laborales asumidas, experiencia profesional, líneas de trabajo y/o investigación, correo electrónico, perfiles en redes profesionales>

<Curriculum e información de contacto del consultante: nombre y apellidos, títulos académicos, formación de postgrado recibida, lugar de trabajo, responsabilidades laborales asumidas, experiencia profesional, líneas de trabajo y/o investigación, correo electrónico, perfiles en redes profesionales>

AGRADECIMIENTOS

Quisiera agradecer en primer lugar a mis padres que me han apoyado en todo momento y a lo largo de mi trayectoria estudiantil. En segundo lugar, quisiera agradecer a mi profesor Luis que gracias a su tiempo y esfuerzo logre mis objetivos. Agradecer a mis abuelos que siempre han estado presente en los momentos decisivos y también a mi tío David el cual fue uno de los pilares fundamentales en el desarrollo de esta investigación. A mis tutores y profesores de la Universidad de las Ciencias Informáticas que me aclararon las dudas a medida que estas fueron surgiendo. En fin, muchas gracias a todos, este logro también es de ustedes.

DEDICATORIA

<Insertar dedicatoria a personas naturales o jurídicas a las que se desee dedicar especialmente el trabajo, bien por vínculos afectivos, familiares, o de membresía. No deben confundirse con la sección “Agradecimientos” que tiene otros objetivos. Debe ser breve y de argumentarse la razón de inclusión, debe mantenerse un lenguaje respetuoso y científico. Esta sección es totalmente opcional y de no utilizarse se suprime del documento. Puede utilizarse un formato de letra distinto al que oficialmente se establece para el resto del documento, aunque pudiera esta selección producir un contraste no favorable para la lectura y legibilidad de la obra. No pueden exceder una cuartilla en su extensión>

RESUMEN

Con el desarrollo de las nuevas tecnologías de la informática y las comunicaciones, diversos procesos que se llevan a cabo en las organizaciones se han favorecido, convirtiéndose en un factor determinante en la organización del trabajo. Específicamente en el área de la gestión de recursos humanos, donde se trabaja con numerosos y diversos datos de cada trabajador, los sistemas informáticos de gestión de información resultan muy útiles en la toma de decisiones empresariales. En el Departamento de Capital Humano de la Empresa Laboratorio de Hemoderivados, Sueros, Bioterios y de Producción Agropecuaria, perteneciente al Grupo Empresarial LABIOFAM, se han venido presentando dificultades en la gestión de información de la fuerza de trabajo calificada (FTC) de la empresa. Se realizó la investigación que lleva como título “Aplicación web para la gestión de procesos de Recursos Humanos de la empresa LABIOFAM.” con el objetivo de desarrollar una aplicación web que permita la gestión de la información de los trabajadores de la empresa. Se realizó una amplia revisión bibliográfica, que permitió elaborar y exponer la fundamentación teórica de las aplicaciones web, así como las herramientas y tecnologías utilizadas. El proceso de desarrollo fue guiado por la metodología de desarrollo XP, utilizando JavaScript como lenguaje de programación. La construcción del sistema propuesto tiene un carácter iterativo, permitiendo la inserción de nuevas funcionalidades y la corrección de errores.

PALABRAS CLAVE
Gestión, Proceso, Aplicación web.

ABSTRACT

With the development of new information and communication technologies, various processes carried out in organizations have been favored, becoming a determining factor in the organization of work. Specifically, in the area of human resource management, where we work with numerous and diverse data from each worker, information management computer

systems are very useful in making business decisions. In the Human Capital Department of the Laboratory of Hemoderivatives, Serums, Bioteriums and Agricultural Production, belonging to the LABIOFAM Business Group, there have been difficulties in the management of information from the qualified workforce (FTC) of the company. The investigation was carried out entitled "Web application for the management of Human Resources processes of the company LABIOFAM." with the aim of developing a web application that allows the management of the information of the company's workers. An extensive bibliographic review was carried out, which allowed to elaborate and expose the theoretical foundations of web applications, as well as the tools and technologies used. The development process was guided by the XP development methodology, using JavaScript as the programming language. The construction of the proposed system has an iterative nature, allowing the insertion of new functionalities and the correction of errors.

KEYWORDS

Management, Process, Web Application.

TABLA DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

OPINIÓN DEL(OS) TUTOR(ES)

<Contenido de la opinión de los tutores>

AVAL DEL CLIENTE

<Contenido del aval del cliente sobre la solución desarrollada>

INTRODUCCIÓN

La gestión de los recursos humanos es primordial para las organizaciones empresariales; en las últimas décadas se ha convertido en uno de los factores fundamentales de las políticas empresariales, su impacto tiene alcance no sólo al interior de la organización, es un fenómeno que trasciende al ámbito social. El personal de una empresa, ya no es solo percibido como recurso o capital humano; además, es visto como talento humano susceptible de ser potenciado en beneficio de la gestión empresarial competitiva. Toda organización que busca posicionarse y mantenerse sólidamente en el mercado requiere satisfacer las exigencias del cliente, de manera que lo distinga de sus competidores por la calidad y eficiencia de los productos y servicios que oferta, para lo que precisa del talento humano (Armijos Mayon et al. 2019).

Al respecto (Agudelo et al. 2016) expresan: *"Hoy el desafío al que se deben enfrentar los dirigentes de las organizaciones está fundamentado, entre otros aspectos, en la dirección de su recurso humano hacia una labor orientada a alcanzar la eficacia y la eficiencia, con el fin de lograr altos estándares de rendimiento fundamentado en valor agregado y en una notoria ventaja competitiva. Es importante tener presente que cuando la organización hace las cosas bien, se tienen grandes ganancias a diferencia de las que no lo hacen".* (p.2)

Los empresarios han comprendido que la gestión de los recursos humanos juega un rol fundamental en la consecución de este propósito, es por ello, que cada día más se interesan por conocer qué motiva a sus empleados, qué aspiraciones tienen en la vida laboral, cuáles son sus potencialidades laborales y qué requerimientos de capacitación y competencia tienen que cubrir (Hernando & Van 2007).

Por otra parte, los avances científico-técnicos en las ramas de la informática y las comunicaciones han revolucionado al mundo en todos los aspectos, la humanidad los ha venido utilizando en su beneficio para un mejor desempeño dentro de la sociedad. Dentro de estos avances el gran auge de las Tecnologías de la Información y las Comunicaciones (TIC) ha generado un cambio en la forma de gestionar y acceder a la información. Las TIC son cada vez más

usadas para el apoyo e informatización de todas las actividades de las empresas. Gracias a ellas las organizaciones han conseguido obtener importantes beneficios, entre los que cabe destacar la mejora de sus operaciones, llegada a una mayor cantidad de clientes, optimización de sus recursos, apertura a nuevos mercados y un conocimiento más profundo acerca de las necesidades de los empleados y clientes para así brindar un servicio de mejor calidad y una comunicación más fluida.

En el año 1960, producto del desabastecimiento de vacunas y medicamentos con el fin de lograr el necesario desarrollo de la ganadería como estrategia del país para garantizar la alimentación del pueblo surge la industria cubana de vacunas y medicamentos veterinarios que con su evolución se ha convertido en el Grupo Empresarial LABIOFAM. Con el transcurso del tiempo y ante los cambios de la economía mundial y los altibajos del comercio internacional, se diseña una estrategia de diversificación. Esta estrategia permite el desarrollo de las bases para otras producciones como plaguicidas biológicos para el control de epidemias y enfermedades, productos para la higiene, suplementos dietéticos de origen natural y alimentos. La gestión actual del Grupo está dirigida a vacunas y medicamentos de uso veterinario y cuenta con una amplia gama de productos biológicos y farmacéuticos. LABIOFAM se integra como todos los centros de producción–investigación–desarrollo de la actividad biotecnológica a otras entidades de la misma esfera, con cuya integración se potencializa su capacidad de investigación (Ecured 2018).

Con el desarrollo de las nuevas tecnologías de la informática y las comunicaciones, diversos procesos que se llevan a cabo en las organizaciones se han favorecido, convirtiéndose en un factor determinante en la organización del trabajo. Específicamente en el área de la gestión de recursos humanos, donde se trabaja con numerosos y diversos datos de cada trabajador, los sistemas informáticos de gestión de información resultan muy útiles en la toma de decisiones empresariales.

En el Departamento de Capital Humano de la Empresa Laboratorio de Hemoderivados, Sueros, Bioterios y de Producción Agropecuaria, perteneciente al Grupo Empresarial LABIOFAM, se han venido presentando dificultades en la gestión de información de la fuerza de trabajo calificada (FTC) de la empresa. El Sistema Unificado Geforza en su versión 4.0,

software que distribuye el Ministerio de Trabajo y Seguridad Social (MTSS) para la gestión de información de la FTC, no funciona correctamente en la empresa, y presenta deficiencias en la mayoría de sus funcionalidades, principalmente porque no permite adicionar nuevos trabajadores, y por consiguiente no se puede actualizar la base de datos. Por estas razones no se puede acceder a la proyección de demanda anual de graduados que se debe entregar a los ministerios de educación ni a estadísticas importantes para la toma de decisiones en cuanto al desarrollo profesional de los trabajadores. El sistema, además, tiene una interfaz poco intuitiva lo que dificulta su uso por parte de los trabajadores. Como promedio mensual en la empresa laboran más de 500 trabajadores, pertenecientes a ocho áreas de trabajo diferentes, dispersas además en las provincias La Habana, Artemisa y Mayabeque, por lo que la cantidad de información que se gestiona sobre los mismos complejiza el proceso. Por tanto, el **problema de investigación** queda definido de la siguiente forma:

¿Cómo contribuir a la gestión de procesos de Recursos Humanos de la empresa Laboratorios de Hemoderivados Sueros Bioterios y de Producción Agropecuaria?

Siendo identificado como **objeto de estudio** el proceso de desarrollo de aplicaciones web. Delimitándose el siguiente **campo de acción:** Aplicaciones web para la gestión de procesos.

Para darle solución al problema anteriormente planteado, se define como **objetivo general** de la investigación: Desarrollar una aplicación web que permita la gestión de procesos de Recursos Humanos de la empresa Laboratorios de Hemoderivados Sueros Bioterios y de Producción Agropecuaria.

Para darle cumplimiento al **objetivo general** de la investigación se trazan los siguientes **objetivos específicos**:

1. Construir el marco teórico referencial de la investigación, relacionado con la gestión de procesos de Recursos Humanos.
2. Desarrollar una aplicación web que permita gestionar los procesos de Recursos Humanos de la empresa Laboratorios de Hemoderivados Sueros Bioterios y de Producción Agropecuaria.
3. Validar la aplicación web desarrollada a partir de los métodos científicos definidos.

Hipótesis:

El desarrollo de la aplicación web en el Departamento de Recursos Humanos de la Empresa Laboratorio de Hemoderivados, Sueros, Bioterios y de Producción Agropecuaria, contribuirá a la gestión de procesos relacionados con los trabajadores de la empresa.

Se proponen las siguientes **tareas de la investigación** para cumplir con los objetivos:

1. Realización de un estudio sobre las tendencias actuales en el desarrollo de sistemas homólogos.
2. Ejecución de una entrevista que permita obtener los requisitos funcionales y no funcionales de la aplicación.
3. Elaboración de un estudio para la selección de la metodología de desarrollo de software, las tecnologías y estándares que se necesitan para implementar la propuesta de solución.
4. Selección de librerías de componentes que apoyen a la implementación, desarrollo de la aplicación web y sus funcionalidades.
5. Ejecución de las pruebas funcionales, de carga y estrés y de seguridad para validar el correcto funcionamiento de la aplicación web.

Para el desarrollo de la investigación se utilizaron los siguientes métodos científicos:

Métodos teóricos

Histórico-Lógico: se utilizó en el análisis de los sistemas homólogos que gestionan procesos relacionados con los recursos humanos de una empresa de manera que permita buscar elementos que los caractericen y aspectos para fundamentar la propuesta de solución a la problemática planteada.

Analítico-Sintético: este método permitió recuperar toda la información necesaria para el desarrollo del presente trabajo, además del análisis de las diferentes tecnologías a utilizar en el desarrollo de la aplicación.

Modelación: este método se utilizó con el objetivo de materializar la representación abstracta de determinadas características del sistema mediante la construcción de diagramas y modelos a lo largo del desarrollo de la investigación.

Métodos empíricos

Entrevista: se le realizaron entrevistas al cliente al usuario obtener información acerca de las funcionalidades que debía cumplir la aplicación web.

Observación: permitió conocer cuáles son las dificultades de la empresa que se pueden resolver con el nuevo sistema.

El trabajo de diploma está estructurado en los siguientes capítulos:

Capítulo I. Aspectos teóricos acerca de aplicaciones web, tecnologías para su desarrollo.

En este capítulo se exponen los principales conceptos asociados al dominio del problema, se realiza un estudio de homólogos, se caracteriza el estado actual de las tecnologías escogidas para la realización de la investigación. Se tratan los elementos teóricos que sustentan y fundamentan los objetivos del trabajo.

Capítulo II. Planificación y diseño de la solución.

En este capítulo se define la propuesta de solución atendiendo al problema de investigación. Se identifican las funcionalidades, partiendo de las descripciones de las necesidades planteadas por el cliente mediante las historias de usuarios. Además, de diseñar la arquitectura que mejor se adapta al proyecto y el modelo de datos.

Capítulo III. Implementación y pruebas al sistema.

En este capítulo se detallan los temas referentes a la implementación de la solución y se realizan las pruebas a la aplicación con el objetivo de evaluar su correcto funcionamiento.

CAPÍTULO I: ASPECTOS TEORICOS ACERCA DE LAS APLICACIONES WEB, HERRAMIENTAS Y TECNOLOGIAS PARA SU DESARROLLO

En este capítulo se hace referencia al marco conceptual, citando las principales definiciones de interés asociadas al dominio del problema. Se realiza un estudio del estado del arte de las aplicaciones web que gestionan procesos relacionados con los recursos humanos de una empresa a nivel internacional. Se muestra además un análisis detallado de las tecnologías a emplear en la construcción de la solución.

I.1 Definiciones de interés

Gestión

La **gestión** es un conjunto de procedimientos y acciones que se llevan a cabo para lograr un determinado objetivo. La palabra gestión suele relacionarse principalmente con el ambiente corporativo, sin embargo, no solo se gestionan compañías o proyectos, sino cualquier tipo de recurso (Westreicher 2020).

Para (Robbins y Coulter, 2005), **gestión** o **administración** se refiere a la coordinación de actividades de trabajo, de modo que se realicen de manera eficiente y eficaz con otras personas y a través de ellas, lo cual se convierte en el objetivo principal de toda **gestión**.

Taylor, considerado padre de la administración, y con base en sus principios de la organización científica del trabajo desarrollados en 1911, “la **gestión** es el arte de saber lo **que** se quiere hacer y a continuación, hacerlo de la mejor manera y por el camino más eficiente” (Taylor, 2005).

Para la presente investigación se adopta la definición de Taylor (2005) ya que el cliente sabe que es lo que necesita exactamente y cuál es el siguiente requisito que se debe implementar, además el equipo trabaja para desarrollar la aplicación de la mejor manera y lo más eficiente posible.

Procesos

Según, (Krajewski, Ritzman y Malhotra, 2008), es un **proceso** es cualquier actividad o grupo de actividades en las que se transforman uno o más insumos para obtener uno o más productos para los clientes, sin embargo, el concepto puede ser aún mucho más amplio; un proceso

puede tener su propio conjunto de objetivos, abarcar un flujo de trabajo que traspase las fronteras departamentales y requerir recursos de varios departamentos.

Por su parte (Aquilano, Jacobs y Chase, 2004)un **proceso** es cualquier parte de una organización que recibe insumos y los transforma en productos o servicios, mismos que se esperan que sean de mayor valor para la organización que los insumos originales. Se considera que la comprensión del funcionamiento del proceso es esencial para asegurar la competitividad de una compañía; un proceso que no se ajusta a las necesidades de la empresa castigara a la misma cada minuto que opere.

Continuando con este margen de ideas, (Harrington, 1996), concuerda con lo anteriormente expuesto al definir un **proceso** como cualquier actividad o grupo de actividades que emplee un insumo, sin embargo, el mismo enfatiza la necesidad de agregarle valor a este y suministrar un producto a un cliente interno o externo.

Para el proyecto se adopta como definición de proceso la que brinda Harrington (1996) ya que la definición, planeación e implementación del proyecto emplea un insumo con el objetivo de agregarle valor y suministrarle a un cliente externo un producto de calidad.

Gestión de Procesos

Gestión por Procesos puede ser conceptualizada como la forma de gestionar toda la organización basándose en los Procesos, siendo definidos estos como una secuencia de actividades orientadas a generar un valor añadido sobre una entrada para conseguir un resultado, y una salida que a su vez satisfaga los requerimientos del cliente. (Negrín, 2003)

En esta se precisa que es una forma de organización de los procesos que se realizan en la empresa y que la misma se guía por las necesidades del cliente. Ello se evidencia también en la siguiente definición.

La gestión por Procesos es una forma de organización diferente de la clásica organización funcional, y en el que prima la visión del cliente sobre las actividades de la organización. (Sescam ,2002)

Para Gómez, 2009, la gestión por procesos es una forma de conducir o administrar una organización, concentrándose en el valor agregado para el cliente y las partes interesadas, (Alfaro

Gómez, 2009).Este autor introduce un nuevo elemento que es la consideración de los intereses de la empresa, luego no absolutiza la visión del cliente.

Por otra parte, Bergholz, 2011, considera que la gestión por procesos puede definirse como una forma de enfocar el trabajo, donde se persigue el mejoramiento continuo de las actividades de una organización mediante la identificación, selección, descripción, documentación y mejora continua de los procesos,(Pepper Bergholz, 2011). Esta autora aporta elementos de cómo concretar esta forma de organización.

Para la investigación se adopta como definición de Gestión de procesos la ofrecida por (Negrín, 2003) ya que se puede ver la gestión de procesos como una vía de gestionar la aplicación web mediante procesos siendo estos definidos como actividades secuenciales que se realizan para obtener un valor añadido y escalable y una vez que salga satisfaga las necesidades del cliente.

Recursos Humanos

En la administración de empresas, se denomina recursos humanos (**RR. HH.**) al conjunto de los empleados o colaboradores de una organización o sector económico. También se utiliza para referirse al sistema o proceso de gestión que se ocupa de seleccionar, contratar, formar, emplear y retener al personal que la organización necesita para lograr sus objetivos. El objetivo de las empresas es vincular el área o profesionales de (RR. HH.) con la estrategia de la organización (Rendón 2015). Es de vital importancia conocer que no se administran personas ni recursos humanos, sino que se administra con las personas, siendo estos agentes activos con un papel de gran importancia en las empresas.

Generalmente la función de los recursos humanos está compuesta por áreas tales como reclutamiento y selección, contratación, capacitación, administración o gestión del personal durante la permanencia en la empresa (Puchol, 2007).

I.1.1 Aplicaciones web

Una aplicación *web* es un programa de aplicación que se almacena en un servidor remoto y se entrega a través de la interfaz de navegador (Martínez, 2021).

Funcionamiento de las aplicaciones *web*:

Las aplicaciones web son ejecutadas por medio de un navegador web en una red lo que significa que los datos o los archivos en los que se trabaja son procesados y almacenados dentro de una red a través de un navegador. Por este motivo, este tipo de aplicaciones por lo general, no necesitan ser instaladas en el ordenador o el móvil. Las páginas web pueden contener elementos que permiten una comunicación activa entre el usuario y la información, haciendo que éste acceda a los datos de forma interactiva, ya que el sitio web se encargará de responder a cada una de las acciones que éste ejecute por ejemplo acceder a gestores de bases de datos de todo tipo, publicar e interactuar con los contenidos, rellenar y enviar formularios (Strapp Inc, 2019).

Ventajas del uso de las aplicaciones web:

Las ventajas más importantes que tiene el desarrollo de una aplicación web son las siguientes (Strapp Inc, 2019):

- No necesita instalación ya que accedes a través de un navegador.
- Una aplicación web es multiplataforma y multidispositivo.
- Nuestro ordenador o dispositivo no se afecta en su memoria por el peso de la aplicación, ya que esta se soporta en el servidor donde está alojada.
- La aplicación puede estar en la nube, accesible para cualquier ordenador o dispositivo que tenga acceso a Internet. También podría ser una aplicación local en una intranet.
- Es muy adaptable y muy fácil de actualizar.

  Tipos de aplicaciones *web* (Maluenda de Vega, 2020)**:**

  - Las **aplicaciones *web* estáticas** o también llamadas **sitios *web*** se suelen desarrollar en HTML y CSS y puede utilizarse algo de *JavaScript*. Pueden presentar contenido digital con movimiento, como vídeos, audio o *banners*. No dispone de muchas funcionalidades y el usuario no puede modificarla por sí mismo. Ofrecen poca o ninguna interactividad. Su actualización es compleja, es un proceso lento, tedioso y manual. Estas se emplean para ofrecer información concisa y permanente.

- Las **aplicaciones *web* dinámicas** presentan mayor complejidad técnica. Utilizan bases de datos para cargar la información y los contenidos se actualizan cada vez que el usuario accede a la aplicación. La actualización de los contenidos es sencilla, la mayoría se administra mediante un CMS, no se requiere acudir al servidor. Para su desarrollo existen numerosos lenguajes. Permiten implementar numerosas funcionalidades, como foros o bases de datos. Admite muchas posibilidades de diseño y presentación.
- Los **sistemas de gestión de contenidos o CMS** (*Content Management System*) permiten a los usuarios administradores crear y gestionar el contenido de la aplicación web de forma sencilla. Es la opción más recomendable cuando el contenido de la aplicación deba ser actualizado continuamente, como en los ejemplos de aplicaciones mencionados en el apartado anterior. Por ello, muchas aplicaciones web dinámicas suelen disponer de un gestor de contenidos.
- Un ***e-commerce*** es el tipo de aplicación web utilizada para el comercio electrónico. Su desarrollo es más complejo que el de las anteriores, ya que debe permitir la realización de transacciones *online* a través de los distintos métodos de pago.
- El **portal *web app*** es un tipo de aplicación *web* que permite acceder a los diversos apartados, categorías o secciones a través de un *home*. En él se puede disponer de un perfil del usuario y acceder a foros, chats, correo electrónico, buscadores, contenido reciente, etc. En definitiva, permite acceder a toda la información que se quiere proporcionar al usuario desde un solo punto.

Single-Page Applications

Una SPA es un sitio donde existe un único punto de entrada, generalmente el archivo *index.html.* En la aplicación no hay ningún otro archivo HTML al que se pueda acceder de manera separada y que nos muestre un contenido o parte de la aplicación, toda la acción se produce dentro del mismo *index.html* (Álvarez 2016).

I.1.2 Navegador web

Un navegador *web* es un programa que permite ver la información que contiene una página *web*. El navegador interpreta el código, HTML generalmente, en el que está escrita la página *web* y lo presenta en pantalla permitiendo al usuario interactuar con su contenido.

I.2 Estudios de sistemas homólogos

Las aplicaciones *web* destinadas a la gestión de los Recursos Humanos son de gran utilidad ya que brindan servicios a las empresas que necesitan reforzar los procesos relacionados con sus trabajadores.

Se consultaron varias aplicaciones *web* que dedican su funcionamiento a la gestión de los Recursos Humanos, además de brindar servicios como captación de personal, evaluación del desempeño y gestión del tiempo dentro de las empresas.

I.2.1 Ámbito internacional

Bizneo HR

Bizneo HR es actualmente uno de los líderes indiscutibles del sector. Su aplicación web de recursos humanos cubre todas las necesidades de gestión del talento en cualquier tipo de empresa. Su sistema se encuentra en la nube y es utilizado a diario por usuarios de más de 14 países del mundo. Esta cuenta con módulos que optimizan todos los procesos: desde la captación de personal, las métricas para evaluar el desempeño, hasta la gestión del tiempo dentro de la compañía. Es una solución muy flexible que gestiona por completo el ciclo laboral del empleado. Bizneo HR ofrece herramientas con las que logra diferenciarse de la competencia y las agrupa en dos tipos de tarifas: planes para la gestión de talento y planes para reclutamiento y selección (SoftwarePara.net 2021). Una de las principales desventajas que tiene el uso de este servicio radica en que es un servicio de pago lo cual afecta el presupuesto de la empresa y no contempla las funcionalidades que se necesitan implementar.

Factorial

Factorial es una empresa que nació en Barcelona y cuyo *software* de (RR.HH.) propone la reducción de errores y una gestión de datos impecable. Con su programa es posible centralizar

toda la información en un solo sitio, tales como contratos, nóminas, *e-mails* y demás documentos relevantes. Resulta una solución intuitiva y sencilla. Factorial ofrece varias herramientas para optimizar distintos procesos: gestión de ausencias, registro de jornada con distintos métodos, evaluación del desempeño y seguimiento de objetivos (SoftwarePara.net 2021)

Ámbito nacional

En el ámbito nacional se estudia la aplicación de escritorio Geforza en su versión 4.0 esta es desarrollada por el Ministerio del Trabajo y Seguridad Social para la gestión de la fuerza de trabajo calificada y brinda los siguientes servicios.

1. Disponibilidad de Graduados
2. Demanda y Proyección de graduados
3. Existencia de Trabajadores
4. Eficiencias
5. Fluctuación de las Bajas

I.2.1 Resultados del estudio de sistemas homólogos

Se estudiaron dos de los sistemas en el ámbito internacional y en el ámbito nacional se estudió una, se opta como guía para la solución al sistema Geforza, este cuenta con todas las funcionalidades especificadas por el cliente. Se precisa desarrollar una aplicación *web* con un mayor nivel de usabilidad y sea más entendible para los usuarios. En un futuro se podrían tomar algunas de las funcionalidades de los sistemas Bizneo y Factorial para enriquecer la aplicación *web* y de esta manera brindar más facilidades al cliente.

I.3 Metodologías de desarrollo de software

Una metodología de desarrollo de *software* es un marco de trabajo que se usa para estructurar, planificar y controlar el proceso de desarrollo de sistemas de información. Una gran variedad de estos marcos de trabajo ha evolucionado durante los años, cada uno con sus propias fortalezas y debilidades. Una metodología de desarrollo de sistemas no tiene que ser necesariamente adecuada para usarla en todos los proyectos. Cada una de las metodologías disponibles es más adecuada para tipos específicos de proyectos, basados en consideraciones técnicas, organizacionales, de proyecto y de equipo (Maida, Esteban Grabiel Pacienzia y Julian 2015).

Existe una gran variedad de metodologías para la creación del *software*, las que se clasifican en dos grandes grupos:

Las **metodologías tradicionales** son denominadas, a veces, de forma despectiva, como metodologías pesadas. Centran su atención en llevar una documentación exhaustiva de todo el proyecto, la planificación y control del mismo, en especificaciones precisas de requisitos y modelado y en cumplir con un plan de trabajo, definido todo esto, en la fase inicial del desarrollo del proyecto. Estas metodologías tradicionales imponen una disciplina rigurosa de trabajo sobre el proceso de desarrollo del *software*, con el fin de conseguir un *software* más eficiente. Para ello, se hace énfasis en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto *software*. Se centran especialmente en el control del proceso, mediante una rigurosa definición de roles, actividades, artefactos, herramientas y notaciones para el modelado y documentación detallada. Además, las metodologías tradicionales no se adaptan adecuadamente a los cambios, por lo que no son métodos adecuados cuando se trabaja en un entorno, donde los requisitos no pueden predecirse o bien pueden variar. Otra de las características importantes dentro de este enfoque, son los altos costes al implementar un cambio y la falta de flexibilidad en proyectos donde el entorno es volátil (Maida, Esteban Grabiel Pacienzia y Julian 2015).

Algunas de las metodologías pesadas existentes:

- RUP (*Rational Unified Process*)
- MSF (*Microsoft Solution Framework*)

Una **metodología de desarrollo ágil**, generalmente es un proceso Incremental (entregas frecuentes con ciclos rápidos), también Cooperativo (clientes y desarrolladores trabajan constantemente con una comunicación muy fina y constante), Sencillo (el método es fácil de aprender y modificar para el equipo) y finalmente Adaptativo (capaz de permitir cambios de último momento). Las metodologías ágiles proporcionan una serie de pautas y principios junto a técnicas pragmáticas que hacen que la entrega del proyecto sea menos complicada y más satisfactoria tanto para los clientes como para los equipos de trabajo, evitando de esta manera los caminos burocráticos de las metodologías tradicionales, generando poca documentación y no haciendo

uso de métodos formales. Estas metodologías ponen de relevancia que la capacidad de respuesta a un cambio es más importante que el seguimiento estricto de un plan (Maida, Esteban Grabiel Pacienzia y Julian 2015).

Algunas de las metodologías ágiles existentes.

- Scrum.
- XP (*Extreme Programming*).

I.3.1 Metodología Scrum

Scrum es un proceso de desarrollo de *software* iterativo y creciente utilizado comúnmente en entornos basados en el desarrollo ágil de *software*. Aunque Scrum estaba enfocado a la gestión de procesos de desarrollo de software, puede ser utilizado en equipos de mantenimiento de *software*, o en una aproximación de gestión de programas (Moreta Fernández, 2013).

Características:

- Adaptabilidad a los cambios entre iteraciones.
- Sencillo de entender.
- Auto organización del equipo.
- Rápido sin necesidad de planificaciones iniciales.
- Blindaje de cada iteración con respecto al cambio: Las tareas que fueron implementadas y el cliente ha mostrado su conformidad ya no se vuelven a tocar en ningún momento.
- Cada miembro del equipo trabaja de forma individual.

I.3.2 *Extreme Programming*

Extreme Programming (XP)es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de *software* preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre

todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. Para el desarrollo de la propuesta de solución se escoge XP como metodología que guiará dicho proceso, teniendo en cuenta los aspectos que se listan a continuación:

- El proyecto es de corta duración y todo el trabajo de codificación es llevado a cabo por un programador
- No es necesaria la generación excesiva de artefactos y roles ya que el proyecto es pequeño y está diseñado para ser realizado en el menor tiempo posible.
- Los requisitos suelen cambiar con frecuencia en la medida en que avanza el desarrollo del proyecto, así el cliente puede ir añadiendo Historias de Usuario, dividirlas para agilizar el trabajo o eliminarlas simplemente. XP permite al equipo de trabajo modificar los planes de desarrollo conforme a lo anterior.
- Tanto cliente como desarrolladores forman parte del equipo de trabajo de forma tal que se logra una interacción constante entre ellos. Esto posibilita la retroalimentación, corrección de errores y finalmente la realización de un producto que cumpla las expectativas de las partes involucradas en el proceso.

I.3.3 Fases de la Metodología *Extreme Programming*

1. Fase de planeación: la planeación inicia con las historias de usuario que describen las características y funcionalidades del software. El cliente asigna un valor o prioridad a la historia, los desarrolladores evalúan cada historia y le asignan un costo el cual se mide en semanas de desarrollo.

2. Fase de diseño: el proceso de diseño debe procurar diseños simples y sencillos para facilitar el desarrollo. Se recomienda elaborar un glosario de términos y la correcta especificación de métodos y clases para facilitar posteriores modificaciones, ampliaciones o reutilización del código.

3. Fase de codificación: en esta fase los desarrolladores deben diseñar las pruebas de unidad que ejercitarán cada historia de usuario. Después de tener las pruebas, los desarrolladores

trabajarán en parejas para concentrarse en lo que debe implementarse para pasar la prueba de unidad.

4. Fase de pruebas: las pruebas de unidad deben implementarse con un marco de trabajo que permita automatizarlas, con la finalidad de realizar pruebas de integración y validación diarias, esto proporcionará al equipo un indicador del progreso y revelarán a tiempo si existe alguna falla en el sistema.

I.3.4 Fundamentos de la metodología seleccionada

El proceso de desarrollo del sistema propuesto se caracterizará por un constante intercambio con el cliente y estará conformado por varias iteraciones donde existe la posibilidad de que los requisitos sean cambiados o surjan nuevas funcionalidades, sin importar la fase o etapa. Además, el equipo encargado del desarrollo es pequeño, por lo que es de vital importancia la comunicación entre los miembros, por tanto:

Scrum se base en el principio de “lo que se termina, funciona y está bien, se aparta y ya no se toca”; por esta razón su selección no es la más adecuada ya que en el proceso de desarrollo del sistema existe la posibilidad de que los requisitos puedan ser cambiados en cualquiera de las fases en las que se encuentre el proyecto.

Por otro lado, XP, se utiliza principalmente en proyectos y equipos pequeños, tiene un desarrollo iterativo e incremental, los requisitos pueden ser cambiados e interactúa con el usuario final, convirtiéndolo en un miembro del equipo. Además, el equipo de desarrollo se encuentra familiarizado con la misma.

De acuerdo con las características que presenta el proceso de desarrollo del sistema propuesto y tras el estudio realizado a las diferentes metodologías, se determina como metodología de desarrollo a utilizar XP.

I.4 Tecnologías y lenguajes definidos para la solución

Partiendo de los requerimientos definidos para la implementación del sistema y las características del entorno donde se aplicará la solución propuesta, se realizó un estudio de las tendencias y tecnologías existentes en la actualidad para el desarrollo de aplicaciones *web* enfocadas en la gestión de información.

I.4.1 *Frameworks* para el desarrollo de aplicaciones web

Son aquellos *frameworks* que se utilizan específicamente para la creación de proyectos *online*. Desde el diseño web de una página hasta los servicios web más específicos con la que estas cuenten. Dentro de estos *frameworks* existen otros tipos, dependiendo del lenguaje de programación utilizado. Sin embargo, nada impide que se pueda utilizar un *framework* originalmente pensado en un lenguaje de programación, en otro diferente (Muente, 2020).

1. React.
2. Angular.

React es una librería de *JavaScript* para el desarrollo UI (Interfaz de usuario) por lo que las aplicaciones escritas con React para que puedan ser utilizadas necesitan librerías adicionales. Por ejemplo, Redux, React Router o Helmet optimizan los procesos de gestión de estado, enrutamiento e interacción con la API. Funciones como las de enlazado de datos, el enrutamiento basado en componentes, la generación de proyectos, la validación de formularios, o la inyección de dependencias requieren la instalación de módulos o librerías adicionales. React es minimalista: sin inyección de dependencias, sin plantillas clásicas, sin funciones demasiado complicadas. La librería es bastante sencilla de entender si se conoce bien *JavaScript*. Sin embargo, se necesita bastante tiempo para aprender a configurar un proyecto porque no existe una estructura de proyecto predefinida. También es necesario aprender a manejar la librería *Redux*, que se usa en más de la mitad de las aplicaciones de *React* para la gestión de estado. Las actualizaciones constantes de la librería también requieren de un esfuerzo adicional por parte del desarrollador. Además, existen muchas buenas prácticas en *React*, que son necesarias para poder hacer las cosas bien (Delgado, 2020).

Angular 11.2.12

Angular es un *framework* MVC de *JavaScript* para el desarrollo *web frontend* que permite crear aplicaciones SPA (*Single-Page Applications*). Es un proyecto de código abierto, realizado en *JavaScript* que contiene un conjunto de librerías útiles para el desarrollo de aplicaciones *web* y propone una serie de patrones de diseño para llevarlas a cabo. En pocas palabras, es lo que se conoce como un *framework* para el desarrollo, en este caso sobre el lenguaje *JavaScript* con programación del lado del cliente (Basalo, 2014). Angular promueve y usa patrones de diseño de *software* con el cual se implementa la arquitectura MVC, estos patrones nos marcan la separación del código de los programas. Eso permite repartir la lógica de la aplicación por capas, lo que resulta muy adecuado para aplicaciones de negocio y para las aplicaciones SPA (*Single-Page Applications*). Arquitectura de Angular Modelo Vista Controlador (MVC) es un estilo de arquitectura de *software* que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos (Universidad de Alicante, 2015).

Vistas: Es la representación visual de los datos, será el HTML y todo lo que represente datos o información.

Controladores: Se encargarán de la lógica de la aplicación, recibe las órdenes del usuario y se encarga de solicitar los datos al modelo y de comunicárselos a la vista.

Modelo: Se encarga de los datos, consultando la base de datos. Actualizaciones, consultas, búsquedas

I.4.2 Fundamentación sobre el *framework* seleccionado

Para la presente investigación podría seleccionarse *React* ya que es una librería para desarrollar IU (*User-Interface*), pero se selecciona como *framework* de trabajo Angular en su versión 11.2.12 debido a que es un *framework* completo para el desarrollo de *software*, el cual generalmente no requiere de librerías adicionales. Todas las funciones como los enlaces de datos, el enrutamiento basado en componentes, la generación de proyectos, la validación de formularios, y la inyección de dependencias, se pueden implementar con los medios proporcionados

por el paquete de Angular, además de que se cuenta con previo conocimiento en el manejo de esta tecnología y comenzar con *React* supondría destinar tiempo en aprender las buenas prácticas de estructuración de proyectos.

***TypeScript* 4.2.4**

TypeScript es un *superset* de *JavaScript*. Una tecnología es un *superset* de un lenguaje de programación, cuando puede ejecutar programas de la tecnología, *TypeScript* en este caso, y del lenguaje del que es el *superset*, *JavaScript* en este mismo ejemplo. En resumen, esto significa que los programas de *JavaScript* son programas válidos de *TypeScript*, a pesar de que *TypeScript* sea otro lenguaje de programación (Hernández, 2018).

***PrimeNG* 12.0.0**

PrimeNG es una colección de componentes de interfaz de usuario para Angular. Todos los *widgets* son de código abierto y de uso gratuito bajo la licencia MIT. *PrimeNG* es desarrollado por *PrimeTek Informatics*, un proveedor con años de experiencia en el desarrollo de soluciones UI de código abierto (admin 2019).

***Node* 14.16.1**

Node es un entorno *JavaScript* del lado del servidor, basado en eventos. *Node* ejecuta *JavaScript* utilizando el motor V8, desarrollado por Google para uso de su navegador Chrome. Aprovechando el motor V8 permite a *Node* proporcionar un entorno de ejecución del lado del servidor que compila y ejecuta *JavaScript* a velocidades increíbles. El aumento de velocidad es importante debido a que V8 compila *Javascript* en código de máquina nativo, en lugar de interpretarlo o ejecutarlo como *bytecode*. *Node* es de código abierto, y se ejecuta en Mac OS X, *Windows* y *Linux* (NetConsulting, 2015). La principal característica de *Node* es la conexión asíncrona, es decir que no tiene un intervalo de tiempo constante entre cada evento esto hace que *Node* sea muy rápido. El asincronismo es una característica de cualquier sistema de comunicación en el que el transmisor puede enviar datos sin previo aviso. El receptor debe estar preparado para aceptar datos en cualquier momento.

I.4.3 Sistema de gestión de base de datos

Un sistema de gestión de bases de datos es un *software* o conjunto de programas que permite crear y mantener una base de datos. El SGBD actúa como interfaz entre los programas de aplicación (Usuarios) y el sistema operativo. El objetivo de un SGBD es proporcionar un entorno eficiente a la hora de almacenar y recuperar la información de una base de datos (Cobo 2007).

I.4.3.1 Tipos de bases de datos

Hay muchos tipos diferentes de bases de datos en esta investigación se estará hablando sobre las Bases de datos Relacionales y las Bases de Datos NoSQL también conocidas por Bases de datos no relacionales.

Bases de datos SQL o Relacionales

Es una colección de elementos de datos organizados en un conjunto de tablas formalmente descritas, desde donde se puede acceder a los datos o volver a montarlos de muchas maneras diferentes sin tener que reorganizar las tablas de la base. La interfaz estándar de programa de usuario y aplicación a una base de datos relacional, es el Lenguaje de Consultas Estructuradas (SQL). Los comandos SQL se utilizan tanto para consultas interactivas como para obtener información de una base de datos relacional y la recopilación de datos para informes.

Las bases de datos relacionales se basan en la organización de la información en partes pequeñas que se integran mediante identificadores; a diferencia de las bases de datos no relacionales que, como su nombre lo indica, no tienen un identificador que sirva para relacionar dos o más conjuntos de datos. Además, son más robustas, es decir, tienen mayor capacidad de almacenamiento, y son menos vulnerables ante fallas (Rendón, 2019).

Algunos gestores de base de datos relacionales existentes:

1. PostgreSQL.
2. MariaDB.

PostgreSQL o como también es llamado ***Postgres***, es un sistema de código abierto de administración de bases de datos del tipo relacional, aunque también es posible ejecutar consultas que sean no relaciones. En este sistema, las consultas relacionales se basan en *SQL*, mientras que las no relacionales hacen uso de *JSON*. Se trata de un sistema de código abierto y además gratuito, y su desarrollo es llevado adelante por una gran comunidad de colaboradores de todo el mundo que día a día ponen su granito de arena para hacer de este sistema una de las opciones más sólidas a nivel de bases de datos. PostgreSQL posee data *types* (tipos de datos) avanzados y permite ejecutar optimizaciones de rendimiento avanzadas, que son características que por lo general solo se ven en sistemas de bases de datos comerciales, como por ejemplo *SQL Server* de *Microsoft* u *Oracle* de la compañía homónima (Borges, 2019).

MariaDB es un sistema de gestión de bases de datos que está muy relacionado con MySQL, ya que fue desarrollado por uno de los desarrolladores. El objetivo de su desarrollo fue el de mantener el *software* de gestión de base de datos en un modelo de *software* libre. El sistema de gestión de bases de datos MariaDB incorpora las distintas funciones características de MySQL añadiendo algunas mejoras, como la posibilidad de ejecutar consultas complejas y almacenarlas directamente en caché, la nueva gestión de conexiones a BD, la posibilidad de acceder a *cluster* de datos (interesante para el trabajo en la nube) o el soportar la utilización de jerarquías de *graphs* y estructuras más complejas. En cuanto a seguridad y rendimiento, MariaDB incorpora mejoras, estando siempre en constante evolución gracias a la aportación de una gran comunidad que se encuentra tras de ella (Felipe, 2020).

Bases de datos NoSQL o no Relacionales

Una NoSQL, o una base de datos no relacional, permite que los datos no estructurados y semiestructurados se almacenen y manipulen, a diferencia de una base de datos relacional, que define cómo deben componerse todos los datos insertados en la base de datos. Las bases de datos NoSQL se han hecho populares a medida que las aplicaciones *web* se han hecho más comunes y complejas.

Algunos gestores de base de datos no relacionales existentes:

1. Apache Cassandra.
2. MongoDB.

Apache Cassandra se trata de un *software* NoSQL distribuido y basado en un modelo de almacenamiento de clave-valor, de código abierto que está escrita en Java. Permite grandes volúmenes de datos en forma distribuida. Este gestor de bases de datos es utilizado por Twitter para su plataforma.

1. Es una base de datos distribuida, es decir, lo que significa que los servidores van a estar distribuidos.
2. Escala linealmente, lo que significa que, si tenemos dos nodos, se podrán realizar 100000 operaciones por segundo. Si tuviéramos cuatro nodos podremos realizar el doble de operaciones, y así sucesivamente, cada vez que dupliquemos el número de nodos, duplicaremos el número de operaciones por segundo.
3. No sigue un patrón maestro-esclavo, sino que es *peer-to-peer* o P2P. Esto lo que conlleva es que, si se cae un nodo, el servicio puede seguir funcionando, no como en el patrón maestro-esclavo, en el que, de forma resumida, si se cae el maestro el sistema cae también.
4. Permite la escalabilidad horizontal, que es diferente a la escalabilidad vertical. En la segunda lo que se aumenta es la máquina, como por ejemplo tener una máquina con 16 gigas de RAM y la aumentamos a 32 gigas de RAM. Y en la primera tenemos una máquina con 16 gigas de RAM y lo que hacemos es poner otra máquina también con 16 gigas de RAM trabajando en paralelo con la otra.
5. Es tolerante a fallos, gracias a que posee la replicación de datos, es decir, los datos cuando son escritos en un nodo se replican en otros nodos, por lo que, si uno de estos nodos cae, no pasa nada porque el dato está replicado en otros dos.
6. Permite definir el nivel de consistencia.
7. Usa el lenguaje CQL, que es un lenguaje muy similar a SQL.
8. Permite la replicación en varios *data center*, siendo cada *data center* un anillo de máquinas Cassandra, ya que permite que el anillo 1 replique sus datos en el anillo 2.

9. Es de código abierto.
10. Tiene una consistencia ajustable.
11. Tiene una fácil escalabilidad (Requena Mesa, 2019).

MongoDB 4.0.8

Es un sistema de base de datos NoSQL orientado a documentos de código abierto y escrito en C++, que en lugar de guardar los datos en tablas lo hace en estructuras de datos *BSON* (similar a *JSON*) con un esquema dinámico. Al ser un proyecto de código abierto, sus binarios están disponibles para los sistemas operativos *Windows*, *GNU/Linux*, OS X. La razón de esto es que MongoDB, al estar escrito en C++, cuenta con una más que notoria capacidad para aprovechar los recursos de la máquina y, al estar licenciado bajo una licencia GNU AGPL 3.0, es posible adaptarlo a nuestras necesidades. La principal característica a destacar de MongoDB, sin duda es la velocidad, que alcanza un balance perfecto entre rendimiento y funcionalidad gracias a su sistema de consulta de contenidos. Pero sus características principales no se limitan solo a esto, MongoDB cuenta, además, con otras que lo posicionan como el preferido de muchos desarrolladores (Robledano, 2019).

Características principales

- **Consultas ad hoc**. *MongoDB* permite realizar todo tipo de consultas tales como búsqueda por campos, consultas de rangos y expresiones regulares. Además, estas consultas pueden devolver un campo específico del documento, pero también puede ser una función JavaScript definida por el usuario.
- **Indexación**. El concepto de índices en *MongoDB* es similar al empleado en bases de datos relacionales, con la diferencia de que cualquier campo documentado puede ser indexado y añadir múltiples índices secundarios.
- **Replicación**. Del mismo modo, la replicación es un proceso básico en la gestión de bases de datos. *MongoDB* soporta el tipo de replicación primario-secundario. De este modo, mientras permite realizar consultas con el primario, el secundario actúa como réplica de datos en solo lectura a modo copia de seguridad con la particularidad de que los nodos secundarios tienen la habilidad de poder elegir un nuevo primario en caso de que el primario actual deje de responder.

- **Balanceo de carga**. *MongoDB* puede escalar la carga de trabajo. *MongoDB* tiene la capacidad de ejecutarse de manera simultánea en múltiples servidores, ofreciendo un balanceo de carga o servicio de replicación de datos, de modo que podemos mantener el sistema funcionando en caso de un fallo del *hardware*.
- **Almacenamiento de archivos**. Aprovechando la capacidad de *MongoDB* para el balanceo de carga y la replicación de datos, Mongo puede ser utilizado también como un sistema de archivos. Esta funcionalidad, llamada *GridFS* e incluida en la distribución oficial, permite manipular archivos y contenido.
- **Ejecución de JavaScript del lado del servidor.** *MongoDB* tiene la capacidad de realizar consultas utilizando *JavaScript*, haciendo que estas sean enviadas directamente a la base de datos para ser ejecutadas.

  MongoDB ofrece una serie de **ventajas**, ya que es ideal para entornos con pocos recursos de computación, esta se puede montar en cualquier servidor u ordenador personal. Esta herramienta tiene un coste bajo. Posee una documentación muy buena, amplia y detallada en comparación con otras bases de datos NoSQL. Es un complemento perfecto para *JavaScript* lo que permite utilizar toda la potencia de sus funciones y sus operadores en *MongoDB*. El uso de *MongoDB* también trae **desventajas** ya que esta no es una base de datos adecuada para transacciones complejas. Es una tecnología joven a pesar de ser ampliamente usada en la actualidad (Martin Gómez 2020).

Cuando utilizar SQL o NoSQL:

Se utiliza **SQL** cuando:

1. El volumen de datos no crece o crece muy poco.
2. Cuando las necesidades de proceso se pueden asumir en un solo servidor
3. Cuando no se tiene picos de uso del sistema por parte de los usuarios más allá de los previstos.

Se utiliza **NoSQL** cuando:

1. El volumen de datos crece muy rápidamente en momentos puntuales.

2. Cuando las necesidades de proceso no se pueden prever.
3. Cuando se tiene picos de uso del sistema por parte de los usuarios en múltiples ocasiones.

I.4.3 Fundamentación del Sistema gestor de base de datos seleccionado

Para la presente investigación se elige *MongoDB* como gestor se base de datos, una de las razones principales por la cual se elige es que el servidor donde se va a realizar el despliegue es de pocos recursos. Para esto *MongoDB* se adapta perfectamente ya que permite su despliegue en ordenadores con pocos recursos, además cuenta con mucha velocidad de lectura ya que se evita los *JOINS* que se realizan entre las tablas en bases de datos relacionales. Es un complemento perfecto para *JavaScript* que es el lenguaje que se va a utilizar. Además, tiene muy buena documentación ya que es muy usada por la comunidad.

I.4.4 *Docker* 3.0.0

Simplificando la definición, *Docker* es una especie de máquina virtual, pero muy ligera. *Docker* es **una plataforma de *software* para contenedores**. De manera similar a como una máquina virtual 'virtualiza' (elimina la necesidad de administrar directamente) el *hardware* del servidor, los contenedores 'virtualizan' el sistema operativo de un servidor. *Docker* se instala en cada servidor y proporciona comandos sencillos que puede utilizar para crear, iniciar o detener contenedores (Bernal, 2020).

Fundamentación sobre la selección de Docker:

Para la ejecución de la presente investigación se elige *Docker* ya que esta nos permite desplegar una aplicación en cualquier entorno en el que se encuentre instalado. Una aplicación incluye no solo la aplicación como tal, sino que se tiene un motor, un conjunto de librerías y el software. *Docker* permite agrupar todo esto en un contenedor y almacenarlo en nuestro disco duro. De forma que copia todo lo que pertenece a un mismo sistema operativo y lo almacena en una zona del disco duro para luego ejecutarla independientemente del sistema operativo. Esto permite que se pueda transportar y ejecutar la aplicación donde se necesite, esto brinda una gran movilidad y sencillez a la hora de desplegar la aplicación *web* ya que no hay que preocuparse por el *software* o si se tienen todos los componentes de la aplicación para iniciarla.

Además, esta da la posibilidad de tener múltiples contenedores que comparten un mismo núcleo y permite restringir el uso de la CPU que pueden usar, gracias a esto se puede trabajar con diferentes aplicaciones.

I.4.5 Entorno de desarrollo integrado (IDE)

Un Entorno de Desarrollo Integrado IDE, es una aplicación que facilita el desarrollo de aplicaciones. Estos Entornos de Desarrollo Integrados consisten en un editor de código, un compilador, un depurador y una interfaz gráfica, que ayudan a un desarrollador de *software* a crear aplicaciones, de forma independiente o que sean parte de aplicaciones ya existentes. (UNIR, 2021).

Visual Studio Code 1.54.1

Visual Studio Code es un editor de código fuente que permite trabajar con diversos lenguajes de programación, admite gestionar tus propios atajos de teclado y refactorizar el código. Es gratuito, de código abierto y nos proporciona una utilidad para descargar y gestionar extensiones con las que podemos personalizar y potenciar esta herramienta (Aitana Soluciones ERP Y CRM, 2018)

Fundamentación de la selección del IDE de desarrollo:

Para el desarrollo de la solución propuesta se seleccionó *Visual Studio Code* ya que este posee resaltado de sintaxis, permite autocompletar lo que es importante para ahorrar tiempo al programador, posee atajos para hacer que la codificación sea eficiente. Este IDE cuenta con un mercado de extensiones para admitir muchas funcionalidades adicionales (por ejemplo, extensiones de *Docker*). *Visual Studio Code* cuenta con una gran potencia, es ligero y multilenguaje, es decir, que soporta diversos lenguajes (*JavaScript, TypeScript*, etc.), cuenta con una mejor interfaz de usuario, complementos fáciles y buena integración lo que la hace más usable. La principal característica por la cual se elige es debido a que es uno de los mejores para el desarrollo de *JavaScript*.

I.4.6 Lenguaje Unificado de Modelado

El lenguaje Unificado de modelado (por sus siglas en inglés UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de *software*. Captura las decisiones y los conocimientos sobre los sistemas que se deben construir. Se usa para entender, diseñar, configurar, mantener y controlar la información sobre tales sistemas. UML incluye conceptos semánticos, notación, y principios generales (Rumbaugh, Booch y Jacobson, 2000).

Los objetivos de UML son muchos, pero se pueden sintetizar sus funciones:

- Visualizar: UML permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.
- Especificar: UML permite especificar cuáles son las características de un sistema antes de su construcción.
- Construir: A partir de los modelos especificados se pueden construir los sistemas diseñados. Documentar: Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión (Orallo 2002).

UML es además un método formal de modelado, lo que aporta las siguientes ventajas:

- Mayor rigor en la especificación.
- Permite realizar una verificación y validación del modelo realizado.
- Se pueden automatizar determinados procesos y permite generar códigos a partir de los modelos y a la inversa (a partir del código fuente generar los modelos). Esto permite que el modelo y el código estén actualizados, con lo que siempre se puede mantener la visión en el diseño, de más alto nivel, de la estructura de un proyecto (Orallo 2002).

Herramientas *Case*

Las herramientas de Ingeniería de *Software* Asistida por Computadora, (por sus siglas en inglés CASE), son las aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de *software*, reduciendo el coste de las mismas en términos de tiempo y de dinero.

Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del *software* en tareas como: el proceso de realizar un diseño del proyecto, cálculo de costos, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras (guestf131a9, 2008).

Visual Paradigm 8.0

Para el modelado con UML se utilizará *Visual Paradigm* 8.0 por ser una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite modelar todos los tipos de diagramas de clases, generar código desde diagramas y generar documentación. La herramienta agiliza la construcción de aplicaciones con calidad y a un menor coste. Posibilita la generación de bases de datos, transformación de diagramas de Entidad-Relación en tablas de base de datos, así como obtener ingeniería inversa de bases de datos (Rondón et al., 2011).

Conclusiones del capítulo

Como parte del desarrollo de este capítulo se determinan las siguientes conclusiones parciales:

1. El estudio de los conceptos asociados a la problemática planteada permitió lograr un mejor entendimiento de la investigación que se realiza.

2. Con el estudio de las aplicaciones web usadas a nivel internacional se pudo conocer funcionalidades comunes en las aplicaciones destinadas a la gestión de Recursos Humanos.

3. El estudio y comparativa entre las metodologías tradicionales y agiles ayudo en el momento de optar por una que se ajuste a los objetivos y requerimientos de la empresa.

4. El estudio de las herramientas y tecnologías seleccionadas tributa a un mayor dominio del tema lo que permite un desarrollo más fácil de la aplicación *web* partiendo de los requerimientos definidos en el levantamiento de información.

CAPÍTULO II: PLANIFICACION EN EL DISEÑO DE LA SOLUCION

El diseño es la parte del proceso de desarrollo de *software* cuyo propósito es decidir cómo el sistema se llevará a cabo. Durante el diseño, se toman decisiones estratégicas y tácticas para cumplir los requerimientos funcionales y de calidad de un sistema.

En este capítulo se describen las actividades desarrolladas durante todo el proceso de análisis y diseño de la solución. Se definen las historias de usuarios, las diferentes iteraciones que se deben tener en cuenta para la elaboración de la aplicación, así como la forma en que la aplicación esta modelada.

Propuesta de solución.

Utilizando la información recopilada en el capítulo precedente y dando como solución al problema planteado en el inicio de este trabajo, se propone el desarrollo de una aplicación *web* como un único punto de autenticación a todos los servicios que esta brinda, permitirá administrar y brindar información al departamento de recursos humanos de LABIOFAM para que sea consultada por sus miembros.

La aplicación *web* tiene como intención brindar a los usuarios la posibilidad de tener acceso a la misma de forma rápida y sencilla, en cualquier momento. La misma permitirá realizar consultas referidas a información interna de la empresa, tal como los datos de cada uno de los trabajadores que la componen.

II.1 Modelo de dominio

Un modelo de dominio es una representación de las clases conceptuales del mundo real, no de componentes *software*. No se trata de un conjunto de diagramas que describen clases *software*, u objetos *software* con responsabilidades (Larman, 2002).

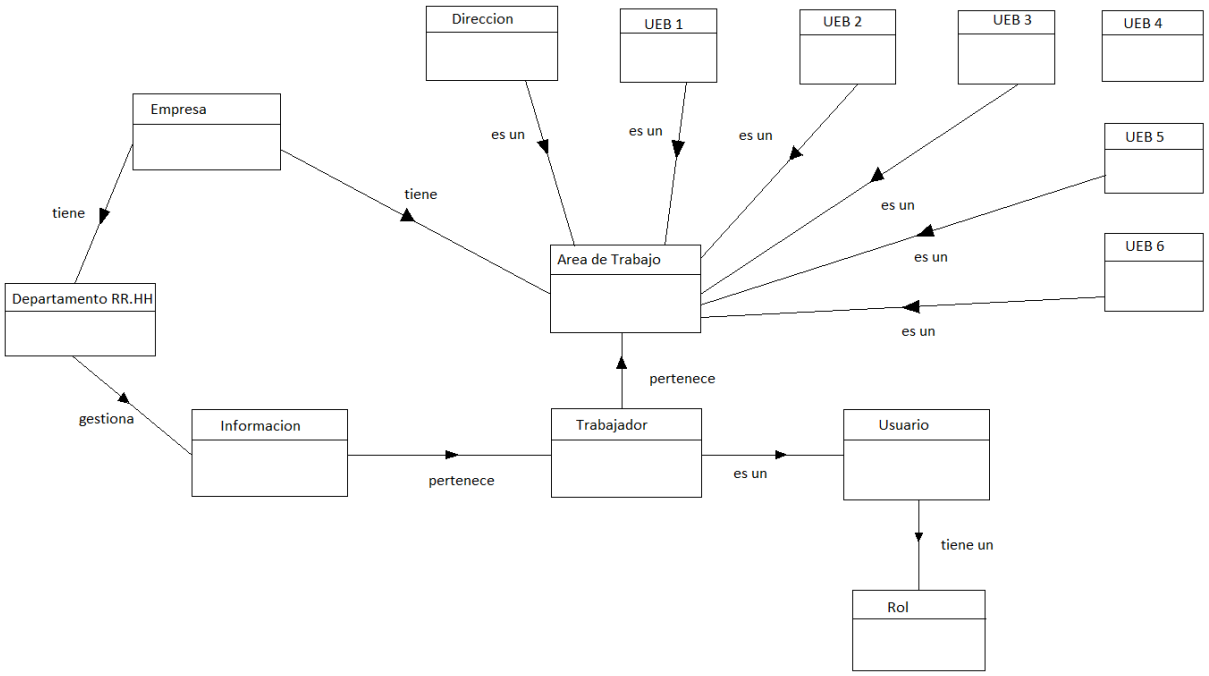

Ilustración 1: Modelo de dominio (Diseño propio)

II.2 Requisitos

Requisitos Funcionales

Son declaraciones de los servicios que proveerá el sistema, de manera en que éste reaccionará en situaciones particulares(Sommerville, 2002). A continuación, se listan los requisitos funcionales que va a contener la aplicación *web*:

- **RF 1** Crear usuario: El sistema debe permitir insertar un nuevo usuario en la base de datos. De cada usuario se registra su nombre de usuario y contraseña.
- **RF 2** Modificar usuario: El sistema debe permitir modificar la información de los usuarios que están registrados en el sistema.
- **RF 3** Listar usuario: El sistema debe permitir listar todos los usuarios guardados en la base de datos.
- **RF 4** Eliminar usuario: El sistema debe permitir eliminar un usuario seleccionado.
- **RF 5** Modificar Administrador: El sistema debe permitir modificar la información del administrador del sistema. Del administrador se registra su nombre de usuario y contraseña.

- **RF 6** Crear Trabajador: El sistema debe permitir insertar la información de nuevos trabajadores a la base de datos de la empresa. De cada trabajador se registra su nombre, apellidos, carnet de identidad, sexo, área de trabajo (referido exactamente a que UEB o dirección), puesto de trabajo, fecha de alta de la entidad, nivel educacional (6to grado, 9no grado, 12 grado, técnico medio o nivel superior), categoría ocupacional (cuadro, técnico, administrativo, operario y servicios), estudios actuales (estudiando en la universidad o en cursos de posgrados), categoría docente, categoría de investigador , categoría científica y especialidad.
- **RF 7** Modificar trabajador: El sistema debe permitir modificar la información de un trabajador determinado y guardar los cambios.
- **RF 8** Listar trabajador: El sistema muestra una lista de los trabajadores existentes en la base de datos.
- **RF 9** Dar baja a un trabajador: El sistema debe permitir eliminar los datos del trabajador de la tabla trabajadores y en la tabla (fluctuación) agregar el nombre del trabajador, el motivo de la baja y la fecha de baja ya que en esa tabla de la base de datos se consulta la fluctuación.
- **RF 10** Hacer reportes exportables: El sistema debe permitir al usuario imprimir la información obtenida en formato pdf o word.
- **RF 11** Filtrar por sexo: El sistema debe ser capaz de filtrar por el sexo de cada trabajador y arrojar una lista con los resultados de la búsqueda.
- **RF 12** Filtrar por rango de edades: El sistema debe permitir filtrar por un rango de edades establecido.
- **RF 13** Filtrar por nivel educacional: El sistema debe permitir filtrar por el nivel educacional de cada trabajador guardado en la base de datos.
- **RF 14** Filtrar por categoría ocupacional: El sistema debe permitir filtrar por la categoría ocupacional que posee cada trabajador.
- **RF 15** Filtrar por estudios: El sistema debe filtrar la lista de trabajadores y mostrar cuales están cursando estudios y que están estudiando.
- **RF 16** Filtrar trabajadores por categoría científica: El sistema debe permitir filtrar a los trabajadores por la categoría científica seleccionada.

- **RF 17** Filtrar trabajadores por categoría docente: El sistema debe permitir filtrar a los trabajadores por la categoría docente seleccionada.
- **RF 18** Demanda de fuerza calificada: Muestra la demanda a 10 años de fuerza de trabajo calificada y en qué carrera o especialidad.
- **RF 19** Autenticar usuario: El sistema debe permitir que los usuarios accedan mediante un sistema de autenticación como única vía, estos deberán introducir su nombre de usuario y contraseña para acceder a la aplicación y en caso de los administradores realizar cambios.
- **RF 20** Realizar copia de seguridad: El sistema debe brindar la opción de realizar copias de seguridad de la base de datos para no perder la información en caso de fallos de sistema.

Requisitos no funcionales

Son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo, estándares, etc.(Sommerville, 2002).

Requisitos de Seguridad

- RNF 1: Validación de credenciales en el lado del cliente para evitar posibles ataques al sistema y la entrada de personal sin autorización a la aplicación.

Requisitos de Usabilidad

- RNF 2: El sistema debe ser usable por cualquier usuario sin importar la experiencia que tenga.
- RNF 3: La interfaz debe ser adaptable a todas las resoluciones de pantalla.
- RNF 4: El sistema presentará buena navegabilidad, proporcionando varias opciones para acceder a cada servicio.

Requisitos de Software

- RNF 5 Cliente: Cualquier sistema operativo y Navegador *Web*.

- RNF 6 Servidor: Sistema Operativo *Linux* o *Windows*, Navegador *Web*, servidor *web Nginx*, lenguaje de programación *TypeScript* 4.2.4, gestor de base de datos *MongoDB* 4.0.8 y para desplegar la aplicación *Docker* 3.0.0.

Requisitos de Hardware

- RNF 6 Servidor: Procesador Intel Core i3 a 3.40GHz, 256 Gb de disco duro, 4gb de memoria RAM o más.
- RNF 7 Cliente: Procesador Intel o AMD, 2Gb de memoria RAM o más.

Requerimientos de Desarrollo

- RNF 8: El sistema debe ser escalable, regularmente debe recibir mantenimiento y ser actualizado.

II.2.1 Arquitectura de la información

- **Roles del sistema.**
- En la siguiente tabla se muestran los roles del sistema y sus responsabilidades.

Roles	Descripción
Usuario(U)	Solo puede realizar consultas en la aplicación web ,este no puede realizar ninguna modificación en la información guardada en la base de datos.
Administrador(ADMIN)	Existe un único administrador de la aplicación web, este accede al sistema a través de un sistema de autenticación , es el único que

	puede realizar cambios en la información que esta almacenada en la base de datos.

II.2.2 Primera fase: Planeación del Proyecto

Fase de planeación: la planeación inicia con las historias de usuario que describen las características y funcionalidades del *software*. El cliente asigna un valor o prioridad a la historia, los desarrolladores evalúan cada historia y le asignan un costo el cual se mide en semanas de desarrollo.

II.2.3 Historia de usuario

Las historias de usuarios (HU) son utilizadas como herramientas para dar a conocer los requerimientos del sistema al equipo de desarrollo. Son pequeños textos para describir una actividad que realizará el *software*. Se puede considerar que estas juegan un papel similar a los casos de uso en otras metodologías, pero en realidad son muy diferentes porque solo muestran la silueta de una tarea a realizarse. Las HU también son utilizadas para estimar el tiempo que el equipo de desarrollo tomará para realizar las entregas. En una entrega se puede desarrollar una o más HU, esto depende solo del tiempo que demore la implementación de cada una de las mismas (González, 2004).

Las HU fueron elaboradas partiendo de las necesidades y funcionalidades establecidas por el cliente, donde el mismo definió un tiempo de entrega y una prioridad.

En las siguientes tablas de muestran las historias de usuarios en tablas (A.1- A.3)

Historias de Usuario	
Numero: UH1	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Adicionar Usuario	

Prioridad de Negocio: Alta	**Iteración:** 1
Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación *web* debe estar presente la función “Agregar Usuario” ,esta consiste en rellenar un formulario para poder crear un usuario.	

Tabla A.1 Historias de usuario # 1

Historias de Usuario	
Numero: UH2	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Modificar Usuario	
Prioridad de Negocio: Alta	**Iteración:** 1
Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación *web* debe estar presente la función “Modificar Usuario” ,esta consiste en seleccionar el ID del usuario que se quiera modificar, luego se re-llena el formulario con los nuevos datos y se modifica satisfactoriamente.	

Tabla A.2 Historias de usuario # 2

Historias de Usuario	
Numero: UH3	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Listar Usuario	
Prioridad de Negocio: Alta	**Iteración:** 1
Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación *web* debe estar presente la función “Listar Usuario” ,esta consiste en listar todos los usuarios guardados en la aplicación *web*.	

Tabla A.3 Historias de usuario # 3

II.2.4 Definición de audiencia

La aplicación *web* constará de una audiencia conformada por un grupo de usuarios, integrado por el Departamento de Recursos Humanos y los trabajadores, los cuales fueron definidos por el cliente.

En la siguiente tabla se muestra la definición de la audiencia.

Grupos de Usuario		**Descripción**
Usuarios	Departamento de Recursos Humanos(RR.HH)	Los miembros de Recursos Humanos son el rol más importante del sistema, pues es a él

		a quien va dirigido la aplicación y está constituido por todos los miembros del departamento. Este rol puede acceder a los diferentes servicios de consultas disponibles en el la aplicación.

II.2.5 Estimación de esfuerzo

Es necesario establecer para cada historia de usuario la prioridad y estimación del esfuerzo necesario para implementar cada una de ellas. Esto se expresa utilizando como unidad de medida el punto. Un punto equivale a una semana ideal de programación. Esta medida generalmente toma valores de 1 a 3.

En la siguiente tabla se muestra la estimación del esfuerzo para cada HU a desarrollar.

No.	UH a desarrollar	Punto de estimación
1.	Adicionar usuario	1
2.	Modificar usuario	1
3.	Listar Usuario	1
4.	Eliminar usuario	0.5
5.	Modificar administrador	1

6.	Adicionar trabajador	1
7.	Modificar información del trabajador	1
8.	Listar trabajadores	1
9.	Dar baja al trabajador	1
10.	Hacer reportes exportables	1
11.	Listar trabajadores por sexo	0.5
12.	Listar trabajadores por rango de edades	1
13.	Listar trabajadores por nivel educacional	1
14.	Listar trabajadores por categoría ocupacional	1
15.	Listar trabajadores que cursan estudios	0.5
16.	Listar trabajadores por categoría científica	1
17.	Listar trabajadores por categoría docente	1
18.	Demanda de fuerza de trabajo calificada	1
19.	Autenticar usuario	1

20.	Realizar copia de seguridad	1

II.2.6 Plan de iteraciones

Se definieron tres iteraciones para la realización del sistema:

Iteración 1: Las UH que el cliente definió que presentan mayor prioridad serán implementadas en esta primera iteración, dando al sistema las primeras funcionalidades, en la primera iteración se pretende desarrollar un CRUD para la información de los trabajadores de la empresa, así como uno para los usuarios y una funcionalidad para modificar al administrador de la aplicación que será almacenada en la base de datos, así como guardar los cambios realizados.

Iteración 2: En la segunda iteración se implementará la funcionalidad “Autenticar usuario”, esta permitirá validar que el usuario es administrador de la aplicación, además habilitar la funcionalidad para realizar reportes exportables.

Iteración 3: En la tercera iteración se implementarán las UH relacionadas con filtrar información a través de parámetros seleccionados por el usuario y listar información.

En la siguiente tabla se muestra el plan de duración de las iteraciones.

Iteraciones	**Historias de Usuario**	**Duración Total(semanas)**
1.	Adicionar usuario	8.5 semanas
	Modificar usuario	

	Listar usuarios	
	Eliminar usuario	
	Modificar administrador	
	Adicionar trabajador	
	Modificar información del trabajador	
	Listar trabajadores	
	Dar baja al trabajador	
	Realizar copia de seguridad	
2.	Hacer reportes exportables	2 semanas
	Autenticar usuario	
3.	Listar trabajadores por sexo	7 semanas
	Listar trabajadores por rango de edades	
	Listar trabajadores por nivel educacional	

	Listar trabajadores por categoría ocupacional	
	Listar trabajadores que cursan estudios	
	Listar trabajadores por categoría científica	
	Listar trabajadores por categoría docente	
	Demanda de fuerza de trabajo calificada	
Total		17.5 semanas

Plan de entrega

En el momento en que culmina el proceso de elaboración de las UH, se inicia el proceso de creación de un plan de entrega. El cual tiene como objetivo fundamental la obtención por parte de los programadores de una estimación detallada del período de tiempo que deben tener en cuenta para la implementación.

En la tabla se muestra el plan de entrega teniendo en cuenta las historias de usuario.

Historias de Usuario	**Final 1ra Iteración**	**Final 2da Iteración**	**Final 3ra Iteración**
Adicionar usuario	X		
Modificar usuario	X		

Listar usuarios	X		
Eliminar usuario	X		
Modificar administrador	X		
Adicionar trabajador	X		
Modificar información del trabajador	X		
Listar trabajadores	X		
Dar baja al trabajador	X		
Realizar copia de seguridad	X		
Hacer reportes exportables		X	
Autenticar usuario		X	
Listar trabajadores por sexo			X
Listar trabajadores por rango de edades			X
Listar trabajadores por nivel educacional			X
Listar trabajadores por categoría ocupacional			X
Listar trabajadores que cursan estudios			X
Listar trabajadores por categoría científica			X
Listar trabajadores por categoría docente			X
Demanda de fuerza de trabajo calificada			X

II.2.7 Segunda fase: Diseño

Fase de diseño: el proceso de diseño debe procurar diseños simples y sencillos para facilitar el desarrollo. Se recomienda elaborar un glosario de términos y la correcta especificación de métodos y clases para facilitar posteriores modificaciones, ampliaciones o reutilización del código.

Diagrama de clases.

En esta fase se definen las clases del sistema, su tipo y la relación entre ellas, con todas sus funciones (atributos y métodos).

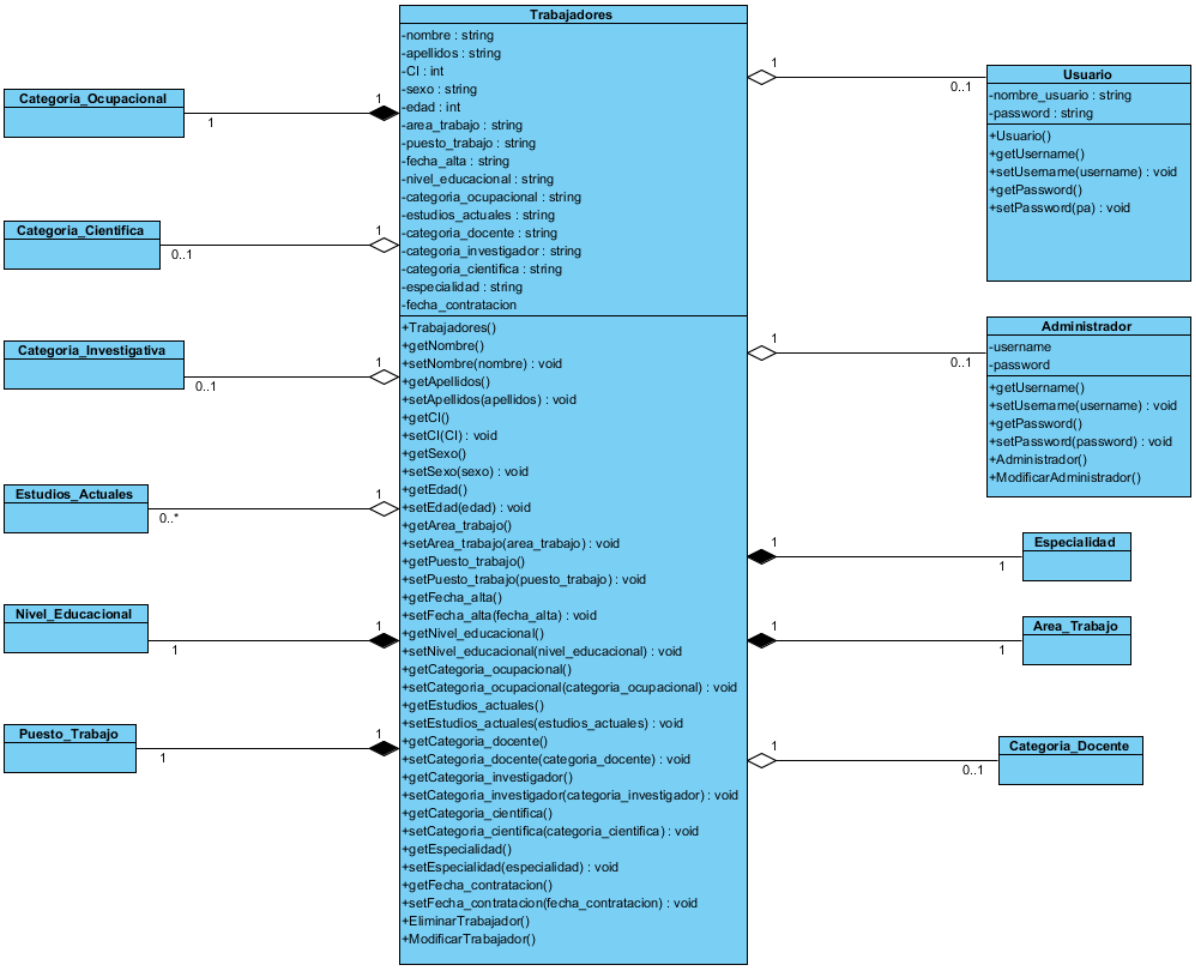

Ilustración 2: Diagrama de clases (Diseño propio)

II.3 Arquitectura de *Software*

La arquitectura del *software* define la estructura que debe de tener un *software*, las piezas que debemos construir y el modo en el que se deben de juntar y trabajar entre ellas. Se define a alto nivel mediante una serie de patrones y abstracciones que seguir para el desarrollo del *software* y para la interacción entre sus diversas piezas (Zúñiga, 2020).

En una arquitectura de *software* confluyen tres elementos fundamentales:

- Los modelos que definen la estructura, topología y dinámica del sistema.
- La trazabilidad o correspondencia de dichos modelos con los requisitos o necesidades establecidas en el contexto que va a operar la solución.

- Las reglas, los principios y justificaciones que rigen la arquitectura y que sustentan las decisiones que se tomaron.

Al diseñar una arquitectura de *software* se crean y representan componentes que interactúan entre sí, con responsabilidades específicas y se organizan de forma tal que se logren los requerimientos establecidos. Se puede partir con patrones de soluciones probados que se conocen con el nombre de estilos arquitectónicos, patrones arquitectónicos y patrones de diseño.

Un estilo arquitectónico es un conjunto coordinado de restricciones arquitectónicas que regula las funciones/características de los elementos arquitectónicos y las relaciones permitidas entre estos dentro de cualquier arquitectura que se adapte a ese estilo (Fielding, 2000).

Un patrón arquitectónico, en cambio, expresa una organización estructural para los sistemas de *software*, proporciona un conjunto de subsistemas predefinidos, especifica sus responsabilidades e incluye reglas y directrices para organizar la relación entre ellos. Los patrones arquitectónicos son plantillas para arquitecturas de *software* concretas. La selección de un patrón arquitectónico es, por lo tanto, una decisión de diseño fundamental en el desarrollo de un sistema de *software* (Buschmann, 1999).

Los patrones de diseño trabajan a una escala intermedia y son independientes del lenguaje de programación. Definen un esquema de refinamiento de los subsistemas o componentes dentro de un sistema, o las relaciones entre estos. Describe una estructura común y recurrente de componentes interrelacionados, que resuelve un problema general de diseño dentro de un contexto particular (Gamma et al,1995).

Arquitectura basada en componentes:

Un componente es una unidad de composición de aplicaciones, que posee un conjunto de interfaces especificadas contractualmente y dependencias del contexto explícitas, puede ser desplegado de forma independiente y está sujeto a la composición por terceras partes (Szyperski, 2002). El estilo de arquitectura basada en componentes, describe un acercamiento

al diseño de sistemas como un conjunto de componentes que exponen interfaces bien definidas y que colaboran entre sí para resolver el problema. El uso de este estilo facilita el despliegue, pues permite sustituir un componente por su nueva versión sin afectar a otros componentes o al sistema y favorece la reusabilidad de los componentes independientes del contexto, permitiendo que se empleen en otras aplicaciones y sistemas.

Arquitectura en capas:

Este estilo define una organización jerárquica tal que cada capa proporciona servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. Entre las principales características de este estilo se encuentran la descomposición de los servicios de forma que la mayoría de las interacciones ocurren solo entre capas vecinas. Los componentes de cada capa se comunican con los componentes de otras capas a través de interfaces conocidas y cada nivel agrega las responsabilidades y abstracciones del nivel inferior (De la Torre et al, 2010).

Tuberías y filtros:

Una tubería (pipeline) es un conjunto de nodos de procesamiento de datos conectados en serie, la salida de un elemento es la entrada del siguiente. Cada etapa del procesamiento se encapsula en un filtro, los datos se transmiten a través de tubos entre filtros adyacentes, (De la Torre et al, 2010) . El uso de este estilo fomenta la reutilización, el bajo acoplamiento y la concurrencia.

II.3.1 Patrón Arquitectónico utilizado

El patrón Modelo-Vista-Controlador (MVC) surge con el objetivo de reducir el esfuerzo de programación, necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos, a partir de estandarizar el diseño de las aplicaciones (Fernández Romero y Díaz González, 2012).

Fue diseñado para reducir el esfuerzo de programación necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos. Sus características principales están dadas por el hecho de que, el Modelo, las Vistas y los Controladores se tratan como entidades separadas; esto hace que cualquier cambio producido en el Modelo se refleje automáticamente en cada una de las Vistas. Este modelo de arquitectura se puede emplear en sistemas de representación gráfica de datos, donde se presentan partes del diseño con diferente escala de aumento, en ventanas separadas. Este modelo de arquitectura presenta varias ventajas (Catalani, 2007) :

• Separación clara entre los componentes de un programa; lo cual permite su implementación por separado.

• Interfaz de Programación de Aplicaciones API (*Aplication Programming Interface*) muy bien definida; cualquiera que use el API, podrá reemplazar el Modelo, la Vista o el Controlador, sin aparente dificultad.

• Conexión entre el Modelo y sus Vistas dinámica; se produce en tiempo de ejecución, no en tiempo de compilación.

Al incorporar el modelo de arquitectura MVC a un diseño, las piezas de un programa se pueden construir por separado y luego unirlas en tiempo de ejecución. Si uno de los componentes, posteriormente, se observa que funciona mal, puede reemplazarse sin que las otras piezas se vean afectadas. Este escenario contrasta con la aproximación monolítica típica de muchos programas de pequeña y mediana complejidad. Todos tienen un *Frame* que contiene todos los elementos, un controlador de eventos, un montón de cálculos y la presentación del resultado.

En la ilustración 1 se muestra de manera gráfica el funcionamiento del patrón utilizado, en este caso MVC:

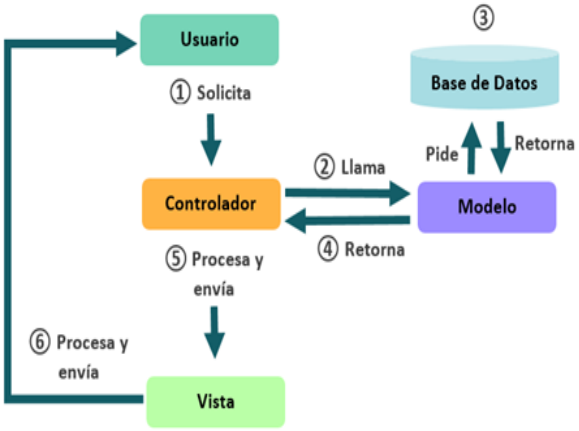

Ilustración 3: Funcionamiento del patrón MVC.

Definición de las partes.

- El Modelo es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El Modelo no tiene conocimiento específico de los Controladores o de las Vistas, ni siquiera contiene referencias a ellos. Es el propio sistema el que tiene encomendada la responsabilidad de mantener enlaces entre el Modelo y sus Vistas, y notificar a las Vistas cuando cambia el Modelo (Catalani, 2007).
- La Vista es el objeto que maneja la presentación visual de los datos representados por el Modelo. Genera una representación visual del Modelo y muestra los datos al usuario. Interactúa preferentemente con el Controlador, pero es posible que trate directamente con el Modelo a través de una referencia al propio Modelo.

El Controlador es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el Modelo, centra toda la interacción entre la Vista y el Modelo. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del Modelo o por alteraciones de la Vista. Interactúa con el Modelo a través de una referencia al propio Modelo.

II.3.2 Diseño de la base de datos

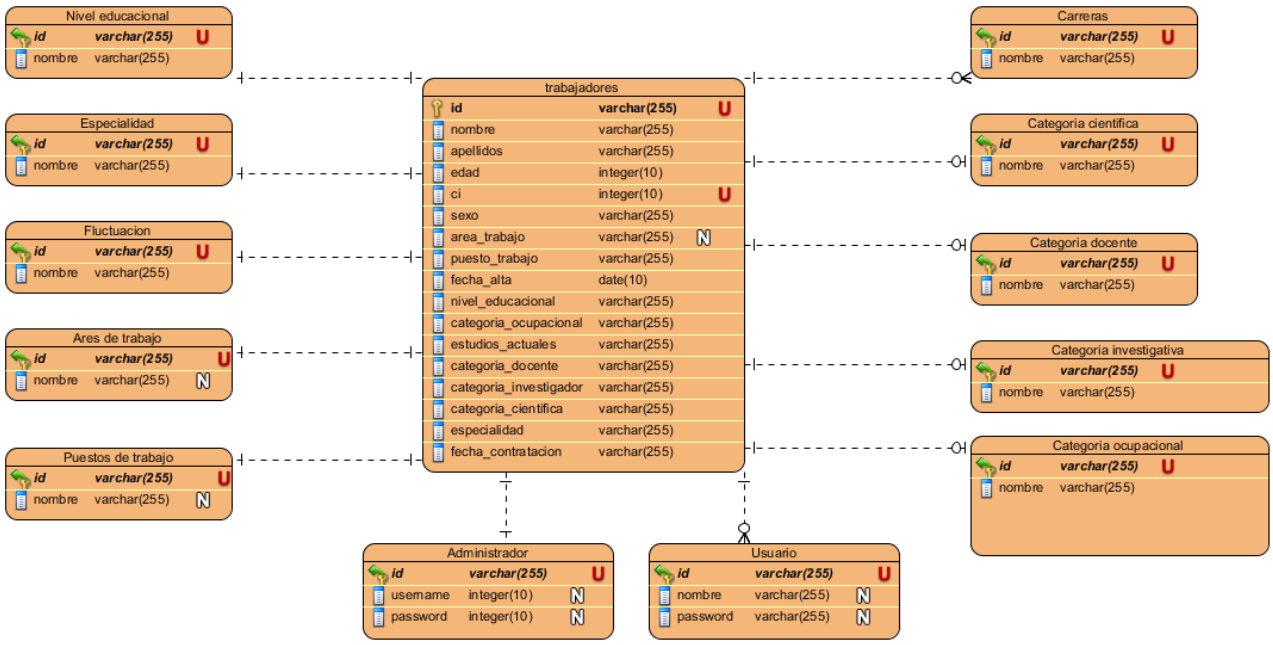

Ilustración 4: Modelo Entidad Relación (Diseño propio)

El diseño de la base de datos es una colección de pasos que ayudan a crear, implementar y mantener los sistemas de administración de datos de una empresa. El propósito principal del diseño de una base de datos es producir modelos físicos y lógicos de diseños para el sistema de base de datos propuesto (Naeem, 2019).

Los patrones GRASP (*General Responsibility Assignment Software Patterns*) describen los principios fundamentales de la asignación de responsabilidades a objetos. El nombre se eligió para sugerir la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos.

Experto es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. Da origen a diseños donde el objeto de software realiza las operaciones que normalmente se aplican a la cosa real que representa, por lo que ofrece una analogía con el mundo real. Con la utilización de este patrón se conserva el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide. El comportamiento se distribuye entre las clases que cuentan

con la información requerida, alentando con ello definiciones de clases sencillas y más cohesivas que son más fáciles de comprender y mantener.

El patrón **Creador** guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Brinda un soporte a un bajo acoplamiento lo que supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización.

El **bajo acoplamiento** es un principio que se debe tener siempre en cuenta durante las decisiones de diseño. Es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño. Este patrón estimula asignar una responsabilidad de modo que su colocación no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento. Soporta el diseño de clases más independientes, que reducen el impacto de los cambios, y también más reutilizables, que acrecienten la oportunidad de una mayor productividad. No puede considerarse en forma independiente de otros patrones como Experto o Alta cohesión, sino que más bien ha de incluirse como uno de los principios del diseño que influyen en la decisión de asignar responsabilidades. Con el uso de este patrón los componentes no se afectan por cambios de otros componentes, son fáciles de entender por separado y fáciles de reutilizar.

El patrón **Alta Cohesión** es la meta principal que ha de tenerse en cuenta en cada momento en todas las decisiones de diseño. Es un patrón evaluativo que el desarrollador aplica al valorar sus decisiones de diseño. Una clase de alta cohesión posee un número relativamente pequeño, con una importante funcionalidad relacionada y poco trabajo que hacer. Colabora con otros objetos para compartir el esfuerzo si la tarea es grande. Con el uso de este patrón mejoran la claridad y la facilidad con que se entiende el diseño. Se simplifican el mantenimiento y las mejoras en funcionalidad. A menudo se genera un bajo acoplamiento. La ventaja de una gran funcionalidad soporta una mayor capacidad de reutilización, porque una clase muy cohesiva puede destinarse a un propósito muy específico.

Patrón Controlador

La mayor parte de los Sistemas reciben eventos de entrada externa, los cuales generalmente incluyen una interfaz gráfica para el usuario operado por una persona. Otros medios de entrada son los mensajes externos o las señales procedentes de sensores como sucede en los sistemas de control de procesos. En todos los casos, si se recurre a un diseño orientado a objetos, hay que elegir los controladores que manejen esos eventos de entrada. Este patrón ofrece una guía para tomar decisiones apropiadas que generalmente se aceptan. La misma clase controlador debería utilizarse con todos los eventos sistémicos de un caso de uso, de modo que se pueda conservar la información referente al estado del caso. Garantiza que la empresa o los procesos de dominio sean manejados por la capa de los objetos del dominio y no por la de la interfaz. Al delegar a un controlador la responsabilidad de la operación de un sistema entre las clases del dominio favorece la reutilización de la lógica para manejar los procesos afines del negocio en aplicaciones futuras.

Patrones GoF

Estos patrones se dividen en las siguientes categorías: Creacionales, de Comportamiento y Estructurales.

- Son soluciones concretas. Proponen soluciones a problemas concretos, no son teorías genéricas.
- Son soluciones técnicas. Indican resoluciones técnicas basadas en Programación Orientada a Objetos (POO). En ocasiones tienen más utilidad con algunos lenguajes de programación y en otras son aplicables a cualquier lenguaje.
- Se utilizan en situaciones frecuentes. Debido a que se basan en la experiencia acumulada al resolver problemas reiterativos.
- Favorecen la reutilización de código. Ayudan a construir software basado en la reutilización, a construir clases reutilizables. Los propios patrones se reutilizan cada vez que se vuelven a aplicar.
- El uso de un patrón no se refleja en el código. Al aplicar un patrón, el código resultante no tiene por qué delatar el patrón o patrones que lo inspiró. No obstante,

últimamente hay múltiples esfuerzos enfocados a la construcción de herramientas de desarrollo basados en los patrones y frecuentemente se incluye en los nombres de las clases el nombre del patrón en que se basan facilitando así la comunicación entre desarrolladores.

- Es difícil reutilizar la implementación de un patrón. Al aplicar un patrón aparecen clases concretas que solucionan un problema concreto y que no será aplicable a otros problemas que requieran el mismo patrón.

Conclusiones del capítulo

En este capítulo se plantea como propuesta de solución el desarrollo de una aplicación web con un único punto de autenticación a todos los servicios, brindando información para que sea consultada por los miembros de la organización. Con el levantamiento de los requerimientos del sistema se logró determinar las funcionalidades básicas a desarrollar durante el proceso; definiendo 20 HU para implementarse en 3 iteraciones. Se realizó una estimación del tiempo de entrega, lo que arrojó un resultado de 17.5 semanas para su desarrollo. El patrón arquitectónico seleccionado es MVC, el cual permite una arquitectura reutilizable y fácilmente mantenible. Como artefactos necesarios para la posterior implementación de la aplicación web, se realizó el diagrama de clases y el modelo entidad relación.

CAPÍTULO III: IMPLEMENTACION Y PRUEBAS AL SISTEMA

Durante el desarrollo de este capítulo se abordarán los aspectos fundamentales del proceso de desarrollo y pruebas. Se desarrollan las tareas de implementación generadas por cada historia de usuario, en correspondencia a las iteraciones anteriormente definidas, obteniendo en cada una de ellas una versión del producto presentable, además se realizarán pruebas de caja negra, de aceptación.

III.1 Tercera Fase: Codificación

Disponibilidad del Cliente, Uno de los requerimientos de XP es tener al cliente disponible durante todo el proyecto. No solamente como apoyo a los desarrolladores, sino formando parte del grupo. El Involucramiento del cliente es fundamental para que pueda desarrollarse un proyecto con la metodología XP. Al comienzo del proyecto, el este debe proporcionar las historias de usuarios. Pero, dado que estas historias son expresamente cortas y de “alto nivel”, no contienen los detalles necesarios para realizar el desarrollo del código. Estos detalles deben ser proporcionados por el cliente, y discutidos con los desarrolladores, durante la etapa de desarrollo (Meléndez Valladarez, Gaitan y Pérez Reyes, 2016).

Propiedad Colectiva del Código, En un proyecto XP, todo el equipo puede contribuir con nuevas ideas que apliquen a cualquier parte del proyecto. Asimismo, una pareja de programadores puede cambiar el código que sea necesario para corregir problemas, agregar funciones o re codificar.

Ritmo Sostenido, La Metodología XP indica que debe llevarse un ritmo sostenido de trabajo. El concepto que se desea establecer con esta práctica es planificar el trabajo de forma a mantener un ritmo constante y razonable, sin sobrecargar al equipo.

III.1.1 Plan de tareas de implementación

Una vez que se han desarrollado las historias de usuarios y el usuario establece las prioridades de estas para su implementación, el equipo de desarrollo evalúa cada escenario y lo divide en tareas de implementación.

Tareas de las historias de usuario implementadas en la **primera iteración**.

1. Adicionar usuario.

2. Modificar usuario.
3. Listar usuarios.
4. Eliminar usuario.
5. Modificar administrador.
6. Adicionar trabajador.
7. Modificar información del trabajador.
8. Listar trabajadores.
9. Dar baja al trabajador.
10. Realizar copia de seguridad

Tareas de Implementación	
Numero de tarea: 1	Numero de historia de usuario: 1
Nombre de la tarea: Adicionar usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 1 de junio del 2021	Fecha de fin: 8 de junio del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita crear nuevos usuarios de la aplicación.	

Tabla T.1 Tarea de implementación # 1

Tareas de Implementación	
Numero de tarea: 2	Numero de historia de usuario: 2
Nombre de la tarea: Modificar Usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 9 de junio del 2021	Fecha de fin: 16 de junio del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	

Descripción: Se implementa exitosamente la funcionalidad que permita modificar usuarios de la aplicación.

Tabla T.2 Tarea de implementación # 2

Tareas de las historias de usuario implementadas en la **segunda iteración**.

1. Hacer reportes exportables
2. Autenticar usuario

Tareas de Implementación	
Numero de tarea: 11	Numero de historia de usuario: 10
Nombre de la tarea: Hacer reportes exportables	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 6 de agosto del 2021	Fecha de fin: 13 de agosto del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad de hacer reportes exportables.	

Tabla T.11 Tarea de implementación # 11

Tareas de las historias de usuario implementadas en la **tercera iteración**.

1. Listar trabajadores por sexo.
2. Listar trabajadores por rango de edades.
3. Listar trabajadores por rango de edades.
4. Listar trabajadores por nivel educacional.
5. Listar trabajadores por categoría ocupacional.
6. Listar trabajadores que cursan estudios.
7. Listar trabajadores por categoría científica.
8. Listar trabajadores por categoría docente.
9. Demanda de fuerza de trabajo calificada.

Tareas de Implementación	
Numero de tarea: 13	Numero de historia de usuario: 12
Nombre de la tarea: Listar trabajadores por sexo	
Tipo de tarea: Desarrollo	Puntos estimados: 3
Fecha de inicio: 22 de agosto del 2021	Fecha de fin: 25 de agosto del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: : Se implementa exitosamente la funcionalidad que permita filtrar a un trabajador almacenado en la base de datos por su sexo.	

Tabla T.13 Tarea de implementación # 13

III.2 Seguridad en las aplicaciones *web*.

Actualmente, no hay día que no se reporten páginas de internet, servidores *web* y dispositivos móviles que son blanco de la ciberdelincuencia. Lamentablemente, los sitios web son muy propensos a ser atacados por los llamados 'piratas informáticos'. Y los servidores de internet también tienden a recibir ataques, sin olvidar los riesgos creados por los propios empleados que desconocen los temas de ciberseguridad o usan inadecuadamente los recursos.

¿Qué es seguridad en la *web* y por qué es tan importante?

En pocas palabras, seguridad *web* son las medidas aplicadas para proteger una página *web* y garantizar que los datos no están expuestos ante los cibercriminales. En este sentido, la seguridad *web* es un proceso continuo y una parte esencial de administrar un sitio *web* (Lara Galicia, 2020).

La ciberseguridad es importante porque los sitios web desprotegidos están expuestos a sufrir situaciones como:

- Robo de información guardada en el servidor *web*.

- Explotación de datos personales –desde direcciones de correo electrónico hasta información de pagos– de los visitantes para utilizarlos inadecuadamente (robo de identidad, extorsiones, abuso de confianza, estafas, etc.).
- Redireccionamiento a páginas *web* maliciosas.
- Mostrar anuncios no deseados.
- Realizar descargas de *software* malicioso.

La aplicación *web* desarrollada implementa un conjunto de controles para mitigar los riesgos de seguridad a los cuales podría estar expuesto, los mismos son:

1. Gestión de políticas de contraseñas: Las políticas de contraseñas definen las medidas de seguridad corporativas como políticas de bloqueo de usuario y caducidad de la contraseña.

Entre las políticas de contraseñas del sitio se encuentra:

- Verificación sintáctica, las contraseñas serán verificadas para evitar tamaños demasiados cortos
- Habilitar bloqueo, se bloqueará la cuenta de un usuario si supera el límite de intentos de entradas fallidas.

2. Gestión de políticas de roles y usuarios: Las políticas de roles y usuarios definen los permisos de acceso a la información y aplicaciones que ofrece el portal.

- No todas las aplicaciones que se brindan a través del portal serán de carácter público, por tal motivo se debe proteger del acceso no autorizado a estas. El portal cuenta con secciones reservadas para el personal autorizado a acceder a dichas aplicaciones.

3. Se define además un control para autorizar a los usuarios con diferentes privilegios de acceso a la información, el cual se basa en los grupos y roles definidos para estos en el sistema.

- Protocolo de Transferencia de Hipertexto Seguro (HTTPS) se utiliza para garantizar la privacidad y la seguridad en la transmisión de la información del usuario hacia el servidor de aplicaciones. Este es un protocolo ubicado en la Capa de Aplicación del Modelo

OSI basado en HTTP. Es la versión segura de HTTP, que es el método más común de intercambio de información en Internet a través de páginas *web*.

III.3 Cuarta Fase: Fase de pruebas.

Una de las características de la metodología XP es el cambio constante, por eso cuando el código de una función está listo se somete a una serie de pruebas unitarias continuas, con el objetivo de corregir fallas periódicamente**.** XP trabaja con tiempos relativamente cortos, por lo que el control automatizado y constante es muy importante. Muchas veces, es el propio cliente quien cumple las funciones de *Tester* cuando tiene conocimientos de programación (se recomienda que sea así para que sus apreciaciones sean realmente válidas para el equipo).

III.3.1 Pruebas Unitarias

Las pruebas unitarias consisten en aislar una parte del código y comprobar que funciona a la perfección. Son pequeños *tests* que validan el comportamiento de un objeto y la lógica.

- Las pruebas unitarias demuestran que la lógica del código está en buen estado y que funcionará en todos los casos.
- Aumentan la legibilidad del código y ayudan a los desarrolladores a entender el código base, lo que facilita hacer cambios más rápidamente.
- Los test unitarios bien realizados sirven como documentación del proyecto.
- Se realizan en poco tiempo por lo que se podrán realizar varias de ellas en muy poco tiempo.

III.3.2 Pruebas de caja negra

Las pruebas de caja negra, es una técnica de pruebas de *software* en la cual la funcionalidad se verifica sin tomar en cuenta la estructura interna de código, detalles de implementación o escenarios de ejecución internos en el *software*. Las pruebas de caja negra, se enfocan sola-

mente en las entradas y salidas del sistema, sin preocuparse en tener conocimiento de la estructura interna del programa de *software*. Para obtener el detalle de cuáles deben ser esas entradas y salidas, se basa únicamente en los requerimientos de *software* y especificaciones funcionales.

De la tabla 1-19 se muestran las pruebas de caja negra realizadas a las funcionalidades críticas de la aplicación *web*: A continuación, se muestran de la 1-6 las restantes pueden ser consultadas en los anexos.

Id del escenario	Escenario	Nombre de usuario	Contraseña	Respuesta del sistema	Resultado de la prueba
EC 1	"Usuario creado correctamente"	"lerodriguez"	"asd123bd"	El usuario se ha creado satisfactoriamente	Respuesta Satisfactoria
EC 2	"Error al crear el usuario, faltan datos obligatorios"	"lerodriguez"	""	El usuario no se creó satisfactoriamente porque faltan datos obligatorios	Respuesta Satisfactoria

Prueba de caja negra, Tabla 1: Crear usuario, escenario Crear usuario.

Id del escenario	Escenario	Nuevo nombre de usuario	Nueva contraseña	Respuesta del sistema	Resultado de la prueba
EC 1	"Usuario modificado correctamente"	"lerodriguez"	"65soyadmin"	El usuario se ha modificado satisfactoriamente	Respuesta Satisfactoria

EC 2	“Error al modificar el usuario”	“ ”	“”	El usuario no se modificó satisfactoriamente porque faltan datos obligatorios	Respuesta Satisfactoria

Prueba de caja negra, Tabla 2: Modificar usuario, escenario Modificar usuario.

Id del escenario	Escenario	Nombre de usuario	Respuesta del sistema	Resultado de la prueba
EC 1	“Usuarios listados correctamente”	“ismael”, “gonzalo”	Usuarios listados satisfactoriamente	Respuesta Satisfactoria
EC 2	“La lista de usuarios esta vacía”	“ ”	No hay usuarios registrados en la base de datos.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 3: Listar usuario, escenario Listar usuario.

Id del escenario	Escenario	ID de usuario	Respuesta del sistema	Resultado de la prueba
EC 1	“Usuario eliminado correctamente”	“6133d4a08f0cbc3c0ca02ec7”	El usuario se ha eliminado satisfactoriamente	Respuesta Satisfactoria
EC 2	“Error al eliminar el usuario”	“jdbkb374736583746537465”	El usuario no se eliminó satisfactoriamente porque el ID no existe en la base de datos	Respuesta Satisfactoria

Prueba de caja negra, Tabla 4: Eliminar usuario, escenario Eliminar Usuario.

Id del escenario	Escenario	Nuevo nombre de usuario	Nueva contraseña	Respuesta del sistema	Resultado de la prueba

EC 1	“Usuario administrador modificado correctamente”	“larredo”	“9702asd”	El usuario administrador se ha modificado correctamente	Respuesta Satisfactoria
EC 2	“Error al modificar el administrador”	“”	“”	El usuario administrador no se modificó correctamente porque los campos están vacíos.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 5: Modificar administrador, escenario Modificar administrador.

Id del escenario	Escenario	Nombre	Apellidos	Respuesta del sistema	Resultado de la prueba
EC 1	“Trabajador creado correctamente”	“Pedro”	“Hernandez”	El trabajador se ha creado satisfactoriamente	Respuesta Satisfactoria
EC 2	“Error al crear el trabajador, faltan datos obligatorios”	“Pedro”	“”	El trabajador no se creó satisfactoriamente porque faltan datos obligatorios	Respuesta Satisfactoria

Prueba de caja negra, Tabla 6: Crear trabajador, escenario Crear trabajador.

III.3.3 Pruebas de confirmación

Las pruebas de confirmación se realizan para asegurarse que el error ha sido eliminado con éxito. El caso de prueba que detectó el error originalmente se ejecuta de nuevo y esta vez debería pasar sin problemas.

III.3.4 Pruebas de integración

Cada vez que se agrega un nuevo módulo como parte de las pruebas de integración, el *software* cambia. Se establecen nuevas rutas de flujo de datos, ocurren nuevas operaciones de entrada/salida y se invoca nueva lógica de control. Dichos cambios pueden causar problemas con las funciones que anteriormente trabajaban sin fallas. En el contexto de una estrategia de prueba de integración, la prueba de regresión es la nueva ejecución de algún subconjunto de pruebas que ya se realizaron a fin de asegurar que los cambios no propagaron efectos colaterales no deseados (Pressman, 2010).

Conclusiones del capítulo del capítulo

En este capítulo de la investigación se evaluó y dividió cada escenario de las HU, obteniendo 21 tareas de implementación. Las mismas fueron implementadas, logrando el desarrollo de la solución propuesta en el capítulo anterior. Se implementaron como controles de seguridad políticas de contraseñas y roles de usuario, permitiendo proteger la aplicación *web* de vulnerabilidades y amenazas. Como mecanismo para asegurar la correcta ejecución de las funcionalidades del sistema se realizaron pruebas de caja negra, alcanzando resultados satisfactorios.

CONCLUSIONES FINALES

Una vez desarrollada la presente investigación y cumplidos los objetivos propuestos para realizar la misma, se plantea como conclusión que:

- El estado del arte analizado como parte de un estudio previo de conceptos y aplicaciones *web* dedicadas a la gestión en el mundo, sirvió de base para adquirir conocimientos acerca de sus características principales.
- Las tecnologías y herramientas utilizadas, permitieron la construcción de un sistema adaptable a diferentes entornos de despliegue.
- Las definiciones de las necesidades del cliente a través de las historias de usuario, junto al diseño e implementación, propició el funcionamiento adecuado de la aplicación *web* para la empresa LABIOFAM.
- La implementación de las funcionalidades definidas para la aplicación *web*, posibilitó el acceso e intercambio de información entre los miembros a nivel de organización.

Glosario de términos

1. HTML (*HyperText Markup Language*) Lenguaje de Marcas de Hipertexto
2. CSS (*Cascading Style Sheets*) Hojas de Estilo en Cascada
3. CMS (*Content Management System*) Sistema de Gestión de Contenido
4. SPA (*Single-Page Application*) Aplicación de Pagina Única
5. MVC Modelo-Vista-Controlador
6. SGBD Sistema Gestor de Bases de Datos
7. UML Lenguaje Unificado de Modelado

RECOMENDACIONES

Se recomienda realizar pruebas de aceptacion para cada una de las historias de usuario desarrolladas con el objetivo de validar la satisfaccion del cliente.

Se recomienda agregar algunas funcionalidades de las estudiadas en los sistemas homologos con el objetivo de enriquecer la aplicación web y aumentar el número de facilidades que esta brinda a los trabajadores del departamento de Recursos Humanos.

REFERENCIAS BIBLIOGRÁFICAS

1. ADMIN, 2019. Manticore Labs ec - Desarrollo Web y Móvil en el Ecuador Componentes de Primeng en Angular Angular. *Manticore Labs ec - Desarrollo Web y Móvil en el Ecuador* [en línea]. [Consulta: 15 julio 2021]. Disponible en: https://manticore-labs.com/2019/02/25/componentes-de-primeng-en-angular/.

2. AGUDELO, M., ALVEIRO, C., SAAVEDRA, B. y RAMIRO, M., 2016. EL RECURSO HUMANO COMO ELEMENTO FUNDAMENTAL PARA LA GESTIÓN DE CALIDAD Y LA COMPETITIVIDAD ORGANIZACIONAL. , pp. 21.

3. AITANA SOLUCIONES ERP Y CRM, 2018. Visual Studio Code: Funcionalidades y extensiones. *El Blog de Aitana – Partner Microsoft y Sage en España* [en línea]. [Consulta: 7 julio 2021]. Disponible en: https://blog.aitana.es/2018/10/16/visual-studio-code/.

4. ALFARO GOMEZ, S., 2009. *Gestion Por Procesos, Business Process Management*. S.l.: s.n.

5. ALVAREZ, M.A., 2016. Qué es una SPA. [en línea]. [Consulta: 6 julio 2021]. Disponible en: https://desarrolloweb.com/articulos/que-es-una-spa.html.

6. AQUILANO, JACOBS y CHASE, 2004. *Administración de la producción y operaciones para una ventaja competitiva*. 10. S.l.: s.n.

7. ARMIJOS MAYON, F.B., BERMÚDEZ BURGOS, A.I., MORA SÁNCHEZ, N.V., ARMIJOS MAYON, F.B., BERMÚDEZ BURGOS, A.I. y MORA SÁNCHEZ, N.V., 2019. Gestión de administración de los Recursos Humanos. *Revista Universidad y Sociedad* [en línea], vol. 11, no. 4, pp. 163-170. [Consulta: 15 julio 2021]. ISSN 2218-3620. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_abstract&pid=S2218-36202019000400163&lng=es&nrm=iso&tlng=es.

8. BASALO, A., 2014. Qué es AngularJS. [en línea]. [Consulta: 6 julio 2021]. Disponible en: https://desarrolloweb.com/articulos/que-es-angularjs-descripcion-framework-javascript-conceptos.html.

9. BERNAL, J., 2020. Docker: concepto e instalación. *SDOS* [en línea]. [Consulta: 6 julio 2021]. Disponible en: https://sdos.es/blog/docker-concepto-e-instalacion.

10. BORGES, S., 2019. ¿Qué es PostgreSQL? - Para qué sirve, Características e Instalación. *Infranetworking* [en línea]. [Consulta: 9 noviembre 2021]. Disponible en: https://blog.infranetworking.com/servidor-postgresql/.

11. BUSCHMANN, F., 1999. Pattern oriented software architecture: a system of patters. ,

12. CATALANI, E., 2007. ARQUITECTURA Modelo/Vista/Controlador | Exequiel Catalani. [en línea]. [Consulta: 11 octubre 2021]. Disponible en: https://exequielc.wordpress.com/2007/08/20/arquitectura-modelovistacontrolador/.

13. COBO, A., 2007. *Diseño y programación de bases de datos*. S.l.: Vision Libros. ISBN 978-84-9983-147-3.

14. DE LA TORRE, C., ZORRILLA, U., CALVARRO, N.J., RAMOS, M.A., MANTEIGA, C., CORTÉS, F. y GARCÍA, I., 2010. *Guía de Arquitectura N-Capas orientada al Dominio con .NET 4.0.* Krasis Press. S.l.: s.n.

15. DELGADO, L., 2020. Angular vs React: Cuál Elegir Para tu Aplicación. *freeCodeCamp.org* [en línea]. [Consulta: 11 noviembre 2021]. Disponible en: https://www.freecodecamp.org/espanol/news/angular-vs-react-cual-elegir-para-su-aplicacion/.

16. ECURED, 2018. LABIOFAM - EcuRed. [en línea]. [Consulta: 15 julio 2021]. Disponible en: https://www.ecured.cu/LABIOFAM.

17. FELIPE, 2020. Qué es MariaDB y cuáles son sus características | Blog | Hosting Plus Perú. *Hosting Plus* [en línea]. [Consulta: 9 noviembre 2021]. Disponible en: https://www.hostingplus.pe/blog/que-es-mariadb-y-cuales-son-sus-caracteristicas/.

18. FERNÁNDEZ ROMERO, Y. y DÍAZ GONZÁLEZ, Y., 2012. Patrón Modelo-Vista-Controlador. ,

19. GAMMA, E., HELM, R., JOHNSON, R. y VLISSIDES, J., 1995. *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley Professional. S.l.: s.n.

20. GONZÁLEZ, D.A.H., 2004. Un Método para el Diseño de la Base de Datos a partir del Modelo Orientado a Objetos. , pp. 15.

21. GUESTF131A9, 2008. Herramientas Case. [en línea]. S.l. [Consulta: 6 julio 2021]. Disponible en: https://es.slideshare.net/guestf131a9/herramientas-case.

22. HARRINGTON, 1996. *The Complete Benchmarking Implementation*. S.l.: s.n.

23. HERNANDEZ, U., 2018. Qué es TypeScript. *CódigoFacilito* [en línea]. [Consulta: 7 julio 2021]. Disponible en: https://codigofacilito.com/articulos/typescript.

24. KRAJEWSKI, RITZMAN y MALHOTRA, 2008. *Administracion de operaciones. Procesos y cadenas de valor*. Mexico: Prentice Hall.

25. LARA GALICIA, F.P., 2020. ¿Qué es seguridad en la web? Manual básico. *Garage* [en línea]. [Consulta: 11 noviembre 2021]. Disponible en: https://pe.godaddy.com/blog/que-es-seguridad-en-la-web-manual-basico/.

26. LARMAN, C., 2002. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process*. 2da Ed. S.l.: Prentice Hall.

27. MAIDA, ESTEBAN GRABIEL PACIENZIA y JULIAN, 2015. Metodologías de desarrollo de software. , pp. 117.

28. MALUENDA DE VEGA, R., 2020. Tipos de desarrollo de aplicaciones web: ejemplos y características. *Profile Software Services* [en línea]. [Consulta: 15 julio 2021]. Disponible en: https://profile.es/blog/desarrollo-aplicaciones-web/.

29. MARTIN GOMEZ, P., 2020. Ventajas y desventajas de MongoDB. *OpenWebinars.net* [en línea]. [Consulta: 15 julio 2021]. Disponible en: https://openwebinars.net/blog/ventajas-y-desventajas-de-mongodb/.

30. MARTINEZ, L., 2021. ¿Qué es una aplicación web? [2021]. *https://www.crehana.com* [en línea]. [Consulta: 15 julio 2021]. Disponible en: https://www.crehana.com/blog/desarrollo-web/aplicacion-web-que-es/.

31. MELÉNDEZ VALLADAREZ, S.M., GAITAN, M.E. y PÉREZ REYES, N.N., 2016. Metodología Ágil Programación Extrema XP. , pp. 146.

32. MORETA FERNANDEZ, M., 2013. *Informáticas, Desarrollo de una aplicación web que permita incrementar la celeridad en el procesamiento de la información generada en las secciones sindicales de la Universidad de las Ciencias.* 2013. S.l.: s.n.

33. MUENTE, G., 2020. Framework: ¿qué es y cuál es su función en Internet? *Rock Content - ES* [en línea]. [Consulta: 15 julio 2021]. Disponible en: https://rockcontent.com/es/blog/framework/.

34. NAEEM, T., 2019. Todo lo que necesitas saber sobre el diseño de bases de datos. *Astera* [en línea]. [Consulta: 13 octubre 2021]. Disponible en: https://www.astera.com/es/tipo/blog/todo-lo-que-necesitas-saber-sobre-el-dise%C3%B1o-de-bases-de-datos/.

35. NEGRÍN, 2003. NEGRÍN, E (2003) METODOLOGÍA PARA EL PERFECCIONAMIENTO DE LOS PROCESOS EN EMPRESAS HOTELERAS [internet], Diponible en. [en línea]. [Consulta: 12 noviembre 2021]. Disponible en: https://1library.co/article/negr%C3%ADn-metodolog%C3%ADa-perfeccionamiento-procesos-empresas-hoteleras-internet-diponible.y9637lry.

36. NETCONSULTING, 2015. Node.js ▹ ¿Qué es y para que sirve NodeJS? ? ¿Qué no es NodeJS? *Apasionados: Agencia de Marketing y Consultoría Estratégica Online* [en línea]. [Consulta: 6 julio 2021]. Disponible en: https://apasionados.es/blog/nodejs-4430/.

37. ORALLO, E.H., 2002. El Lenguaje Unificado de Modelado (UML). , pp. 6.

38. PEPPER BERGHOLZ, S., 2011. *Definition of process management.* S.l.: s.n.

39. PRESSMAN, 2010. *Software Engineering: A Practitioner's Approach*. 7th Edition. S.l.: s.n.

40. PUCHOL, L., 2007. *Dirección y gestión de recursos humanos*. 7a. ed. act. edición. Madrid: Díaz de Santos: s.n. ISBN ISBN 978-84-7978-831-5.

41. RENDÓN, W., 2015. MÉTODO DE GESTIÓN EMPRESARIAL 333 – Productividad Colombia. [en línea]. [Consulta: 5 julio 2021]. Disponible en: https://cyecompetitividad.wordpress.com/2016/12/13/el-trabajador-no-es-una-extension-de-la-maquina/.

42. RENDÓN, Y.A., 2019. Bases de datos relacionales vs. no relacionales. [en línea]. [Consulta: 9 noviembre 2021]. Disponible en: https://www.pragma.com.co/academia/lecciones/bases-de-datos-relacionales-vs.-no-relacionales.

43. REQUENA MESA, A., 2019. Qué es Apache Cassandra | OpenWebinars. [en línea]. [Consulta: 9 noviembre 2021]. Disponible en: https://openwebinars.net/blog/que-es-apache-cassandra/.

44. ROBBINS, S.P. y COULTER, M., 2005. *Administraci??n $c.* M??xico: Pearson Educaci??n. ISBN 978-970-26-0555-3.

45. ROBLEDANO, 2019. Qué es MongoDB y características. *OpenWebinars.net* [en línea]. [Consulta: 6 julio 2021]. Disponible en: https://openwebinars.net/blog/que-es-mongodb/.

46. R.T. BUSCHMANN, 2000. *Architectural styles and the design of network-based software architectures. Ph.D. dissertation.* 2000. S.l.: University of California, California.

47. RUMBAUGH, J., BOOCH, G. y JACOBSON, I., 2000. *El lenguaje unificado de modelado: manual de referencia* [en línea]. S.l.: s.n. [Consulta: 6 julio 2021]. ISBN 978-84-7829-037-6. Disponible en: https://dialnet.unirioja.es/servlet/libro?codigo=156687.

48. SESCAM, 2002. *Servicio de Calidad de la Atención Sanitaria.* [en línea]. 2002. S.l.: s.n. Disponible en: https://www.chospab.es/calidad/archivos/Documentos/Gestiondeprocesos.pdf.

49. SOFTWAREPARA.NET, 2021. TOP 10 Software de Recursos Humanos | RRHH | 2021. [en línea]. [Consulta: 6 julio 2021]. Disponible en: https://softwarepara.net/sistemas_de_recursos_humanos/.

50. SOMMERVILLE, I., 2002. Requerimientos del software. , pp. 11.

51. STRAPP INC, 2019. ¿Qué es una Aplicación Web? – Desarrollo de Software y Consultoría Tecnológica. [en línea]. [Consulta: 15 julio 2021]. Disponible en: https://www.strappinc.com/blog/strapp-datos/que-es-una-aplicacion-web.

52. SZYPERSKI, A., 2002. *Component software: Beyond Object-Oriented programming.* Addison-Wesley. S.l.: s.n.

53. TAYLOR, 2005. *Principios de la administración científica*. Buenos Aires: El Ateneo.

54. UNIR, 2021. ¿Qué es un IDE en programación? *UNIR* [en línea]. [Consulta: 15 julio 2021]. Disponible en: https://www.unir.net/ingenieria/revista/ide-programacion/.

55. UNIVERSIDAD DE ALICANTE, 2015. Modelo vista controlador (MVC). [en línea]. [Consulta: 6 julio 2021]. Disponible en: https://si.ua.es/es/documentacion/asp-net-mvc-3/1-dia/modelo-vista-controlador-mvc.html.

56. WESTREICHER, G., 2020. Gestión | Economipedia. [en línea]. [Consulta: 5 julio 2021]. Disponible en: https://economipedia.com/definiciones/gestion.html.

57. ZÚÑIGA, F.G. de, 2020. ¿Qué es la arquitectura del software? *Blog de arsys.es* [en línea]. [Consulta: 7 octubre 2021]. Disponible en: https://www.arsys.es/blog/arquitectura-software/.

ANEXOS

Historias de Usuario

Historias de Usuario	
Numero: UH4	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Eliminar Usuario	
Prioridad de Negocio: Alta	**Iteración:** 1
Riesgo de desarrollo: Medio	**Puntos Estimados:** 0.5
Descripción: En la aplicación web debe estar presente la función “Eliminar Usuario” ,esta consiste en eliminar un usuario por su ID.	

Tabla A.4 Historia de usuario # 4

Historias de Usuario	
Numero: UH5	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Modificar Administrador	

Prioridad de Negocio: Alta	**Iteración:** 1
Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación web debe estar presente la función “Modificar Administrador” ,esta consiste en una vez autenticado el usuario administrador este pueda cambiar su contraseña.	

Tabla A.5 Historia de usuario # 5

Historias de Usuario	
Numero: UH6	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Adicionar trabajador	
Prioridad de Negocio: Alta	**Iteración:** 1
Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación web debe estar presente la función “Agregar” ,esta consiste en rellenar un formulario con toda la información de cada trabajador de la empresa para posteriormente agregarlo a la base de datos para futuras consultas.	

Tabla A.6 Historia de usuario # 6

Historias de Usuario	
Numero: UH7	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Modificar información del trabajador	
Prioridad de Negocio: Alta	**Iteración:** 1
Riesgo de desarrollo: Medio	**Puntos Estimados:** 1
Descripción: En la aplicación web debe estar presente la función “Modificar” ,esta consiste en modificar la información de los trabajadores de la empresa que esta almacenada en la base de datos .	

Tabla A.7 Historia de usuario # 7

Historias de Usuario	
Numero: UH8	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Listar trabajadores	
Prioridad de Negocio: Alta	**Iteración:** 1
Riesgo de desarrollo: Medio	**Puntos Estimados:**1

Descripción: En la aplicación web debe estar presente la función “Listar trabajadores” ,esta tiene como objetivo mostrar una lista de cada trabajador de la empresa y toda la información referente a cada uno .

Tabla A.8 Historia de usuario # 8

Historias de Usuario	
Numero: UH9	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Dar baja al trabajador	
Prioridad de Negocio: Alta	**Iteración:** 1
Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación web debe estar presente la función “Dar baja” ,esta tiene como objetivo dar de baja a un trabajador ,agregando el motivo de la baja y la fecha en la que se realizó ,estos datos deben quedar registrados para consultar la fluctuación .	

Tabla A.9 Historia de usuario # 9

Historias de Usuario	
Numero: UH10	**Usuario:** Desarrollador

Nombre de Historia de Usuario: Hacer reportes exportables	
Prioridad de Negocio: Alta	**Iteración:** 2
Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación web debe estar presente la función “Crear Reportes” permite a los usuarios obtener un reporte con toda la información requerida en forma de documento(Word ,Pdf ,etc.) .	

Tabla A.10 Historia de usuario # 10

Historias de Usuario	
Numero: UH11	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Listar trabajadores por sexo	
Prioridad de Negocio: Alta	**Iteración:** 3
Riesgo de desarrollo: Medio	**Puntos Estimados:** 0.5
Descripción: En la aplicación web debe estar presente la función “Listar por sexo” que permite a los usuarios filtrar a los trabajadores por el sexo deseado y obtener una lista .	

Tabla A.11 Historia de usuario # 11

Historias de Usuario	
Numero: UH12	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Listar trabajadores por rango de edades	
Prioridad de Negocio: Alta	**Iteración:** 3
Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación web debe estar presente la función “Listar por rango de edades” esta funcionalidad permite a los usuarios filtrar trabajadores por rangos de edad establecidos por el usuario y de esta forma obtener una lista .	

Tabla A.12 Historia de usuario # 12

Historias de Usuario	
Numero: UH13	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Listar trabajadores por nivel educacional	
Prioridad de Negocio: Alta	**Iteración:** 3

Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación web debe estar presente la función “Listar por nivel educacional” esta funcionalidad permite a los usuarios filtrar a los trabajadores por su nivel educacional y de esta forma obtener una lista .	

Tabla A.13 Historia de usuario # 13

Historias de Usuario	
Numero: UH14	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Listar trabajadores por categoría ocupacional	
Prioridad de Negocio: Alta	**Iteración:** 3
Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación web debe estar presente la función “Listar por nivel educacional” esta funcionalidad permite a los usuarios filtrar a los trabajadores por su categoría ocupacional y de esta forma obtener una lista .	

Tabla A.14 Historia de usuario # 14

Historias de Usuario

Numero: UH15	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Listar trabajadores que cursan estudios	
Prioridad de Negocio: Alta	**Iteración:** 3
Riesgo de desarrollo: Medio	**Puntos Estimados:** 0.5
Descripción: En la aplicación web debe estar presente la función “Listar por estudios” esta funcionalidad permite a los usuarios filtrar a los trabajadores que están cursando estudios y en qué carrera y de esta forma obtener una lista .	

Tabla A.15 Historia de usuario # 15

Historias de Usuario	
Numero: UH16	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Listar trabajadores por categoría científica	
Prioridad de Negocio: Alta	**Iteración:** 3
Riesgo de desarrollo: Medio	**Puntos Estimados:**1

Descripción: En la aplicación web debe estar presente la función “Listar por categoría científica” esta funcionalidad permite a los usuarios filtrar a los trabajadores por su categoría científica y de esta forma obtener una lista .

Tabla A.16 Historia de usuario # 16

Historias de Usuario	
Numero: UH17	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Listar trabajadores por categoría docente	
Prioridad de Negocio: Alta	**Iteración:** 3
Riesgo de desarrollo: Medio	**Puntos Estimados:**1
Descripción: En la aplicación web debe estar presente la función “Listar por categoría docente” esta funcionalidad permite a los usuarios filtrar a los trabajadores por su categoría docente y de esta forma obtener una lista .	

Tabla A.17 Historia de usuario # 17

Historias de Usuario

Numero: UH18	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Demanda de fuerza de trabajo calificada	
Prioridad de Negocio: Alta	**Iteración:** 3
Riesgo de desarrollo: Alto	**Puntos Estimados:**1
Descripción: En la aplicación web debe estar presente la función “Demanda” esta funcionalidad permite a los usuarios ver la demanda de fuerza de trabajo calificada en los últimos 10 años ,esta demanda debe estar desglosada por años y debe mostrar en qué carrera o especialidad .	

Tabla A.18 Historia de usuario # 18

Historias de Usuario	
Numero: UH19	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Autenticar usuario	
Prioridad de Negocio: Alta	**Iteración:** 2
Riesgo de desarrollo: Alta	**Puntos Estimados:** 1
Descripción: En la aplicación web debe estar presente la función “Autenticar usuario” esta funcionalidad por medio de la autenticación permite acceder al sistema al usuario	

administrador para que este pueda efectuar los cambios necesarios en la base de datos .

Tabla A.19 Historia de usuario # 19

Historias de Usuario	
Numero: UH20	**Usuario:** Desarrollador
Nombre de Historia de Usuario: Realizar copia de seguridad	
Prioridad de Negocio: Alta	**Iteración:** 3
Riesgo de desarrollo: Media	**Puntos Estimados:**1
Descripción: En la aplicación web debe estar presente la función “Guardar cambios” esta funcionalidad permite al usuario administrador guardar los cambios realizados a la información en la base de datos .	

Tabla A.20 Historia de usuario # 20

Tareas de implementación

Tareas de Implementación	
Numero de tarea: 3	Numero de historia de usuario: 3
Nombre de la tarea: Listar Usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 7

Fecha de inicio: 17 de junio del 2021	Fecha de fin: 24 de junio del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita listar los usuarios de la aplicación.	

Tabla T.3 Tarea de implementación # 3

Tareas de Implementación	
Numero de tarea: 4	Numero de historia de usuario: 4
Nombre de la tarea: Eliminar Usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 3
Fecha de inicio: 25 de junio del 2021	Fecha de fin: 28 de junio del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita eliminar usuarios de la aplicación.	

Tabla T.4 Tarea de implementación # 4

Tareas de Implementación	
Numero de tarea: 5	Numero de historia de usuario: 5
Nombre de la tarea: Modificar administrador	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 29 de julio del 2021	Fecha de fin: 4 de julio del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita modificar los datos de los administradores de la aplicación.	

Tabla T.5 Tarea de implementación # 5

Tareas de Implementación	
Numero de tarea: 6	Numero de historia de usuario: 6
Nombre de la tarea: Adicionar trabajador	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 5 de julio del 2021	Fecha de fin: 12 de julio del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita crear y guardar en la base de datos a nuevos trabajadores.	

Tabla T.6 Tarea de implementación # 6

Tareas de Implementación	
Numero de tarea: 7	Numero de historia de usuario: 7
Nombre de la tarea: Modificar información del trabajador	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 13 de julio del 2021	Fecha de fin: 20 de julio del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita modificar nuevos la información de los trabajadores almacenados en la aplicación.	

Tabla T.7 Tarea de implementación # 7

Tareas de Implementación	
Numero de tarea: 8	Numero de historia de usuario: 8

Nombre de la tarea: Listar trabajador	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 21 de julio del 2021	Fecha de fin: 28 de julio 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita listar a los trabajadores almacenados en la base de datos.	

Tabla T.8 Tarea de implementación # 8

Tareas de Implementación	
Numero de tarea: 9	Numero de historia de usuario: 9
Nombre de la tarea: Dar baja al trabajador	
Tipo de tarea: Desarrollo	Puntos estimados: 3
Fecha de inicio: 29 de julio del 2021	Fecha de fin: 1 de agosto del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se extrae la información del trabajador del modelo “Trabajadores” permanentemente.	

Tabla T.9 Tarea de implementación # 9

Tareas de Implementación	
Numero de tarea: 10	Numero de historia de usuario: 9
Nombre de la tarea: Dar baja al trabajador	
Tipo de tarea: Desarrollo	Puntos estimados: 3
Fecha de inicio: 2 de agosto del 2021	Fecha de fin: 5 de agosto del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	

Descripción: Se almacena la información del trabajador en el modelo “Fluctuación” con la fecha introducida por la persona que otorga la baja para su posterior consulta.

Tabla T.10 Tarea de implementación # 10

Tareas de Implementación	
Numero de tarea: 11	Numero de historia de usuario: 10
Nombre de la tarea: Hacer reportes exportables	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 6 de agosto del 2021	Fecha de fin: 13 de agosto del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad de hacer reportes exportables.	

Tabla T.11 Tarea de implementación # 11

Tareas de Implementación	
Numero de tarea: 12	Numero de historia de usuario: 11
Nombre de la tarea: Autenticar usuario	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 14 de agosto del 2021	Fecha de fin: 21 de agosto del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa la interfaz y funcionalidad que permita autenticarse en el sistema tanto a usuarios como administradores.	

Tabla T.12 Tarea de implementación # 12

Tareas de Implementación

Numero de tarea: 13	Numero de historia de usuario: 12
Nombre de la tarea: Listar trabajadores por sexo	
Tipo de tarea: Desarrollo	Puntos estimados: 3
Fecha de inicio: 22 de agosto del 2021	Fecha de fin: 25 de agosto del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: : Se implementa exitosamente la funcionalidad que permita filtrar a un trabajador almacenado en la base de datos por su sexo.	

Tabla T.13 Tarea de implementación # 13

Tareas de Implementación	
Numero de tarea: 14	Numero de historia de usuario: 13
Nombre de la tarea: Listar trabajadores por rango de edades	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 26 de agosto del 2021	Fecha de fin: 2 de septiembre del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita filtrar a uno o varios trabajadores almacenados en la base de datos por un rango de edades establecido .	

Tabla T.14 Tarea de implementación # 14

Tareas de Implementación	
Numero de tarea: 15	Numero de historia de usuario: 14
Nombre de la tarea: Listar trabajadores por nivel educacional	
Tipo de tarea: Desarrollo	Puntos estimados: 7

Fecha de inicio: 3 de septiembre del 2021	Fecha de fin: 10 de septiembre del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita filtrar a un trabajador almacenado en la base de datos por su nivel educacional.	

Tabla T.15 Tarea de implementación # 15

Tareas de Implementación	
Numero de tarea: 16	Numero de historia de usuario: 15
Nombre de la tarea: Listar trabajadores por categoría ocupacional	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 11 de septiembre del 2021	Fecha de fin: 18 de septiembre del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita filtrar a un trabajador almacenado en la base de datos por su categoría ocupacional.	

Tabla T.16 Tarea de implementación # 16

Tareas de Implementación	
Numero de tarea: 17	Numero de historia de usuario: 16
Nombre de la tarea: Listar trabajadores que cursan estudios	
Tipo de tarea: Desarrollo	Puntos estimados: 4
Fecha de inicio: 13 de septiembre del 2021	Fecha de fin: 17 de septiembre del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita filtrar a un trabajador almacenado en la base de datos si está cursando estudios.	

Tabla T.17 Tarea de implementación # 17

Tareas de Implementación	
Numero de tarea: 18	Numero de historia de usuario: 17
Nombre de la tarea: Listar trabajadores por categoría científica	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 18 de septiembre del 2021	Fecha de fin: 25 de septiembre del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita filtrar a un trabajador almacenado en la base de datos por la categoría científica.	

Tabla T.18 Tarea de implementación # 18

Tareas de Implementación	
Numero de tarea: 19	Numero de historia de usuario: 18
Nombre de la tarea: Listar trabajadores por categoría docente	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 26 de septiembre del 2021	Fecha de fin: 3 de octubre del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita filtrar a un trabajador almacenado en la base de datos por la categoría docente.	

Tabla T.19 Tarea de implementación # 19

Tareas de Implementación	
Numero de tarea: 20	Numero de historia de usuario: 19

Nombre de la tarea: Demanda de fuerza calificada	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 4 de octubre del 2021	Fecha de fin: 11 de octubre del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que muestre la demanda de fuerza calificada en los últimos 10 años desglosada por años.	

Tabla T.20 Tarea de implementación # 20

Tareas de Implementación	
Numero de tarea: 21	Numero de historia de usuario: 20
Nombre de la tarea: Realizar copia de seguridad	
Tipo de tarea: Desarrollo	Puntos estimados: 7
Fecha de inicio: 12 de octubre del 2021	Fecha de fin: 19 de octubre del 2021
Programador responsable: Luis Ernesto Rodríguez Larred	
Descripción: Se implementa exitosamente la funcionalidad que permita realizar copias de seguridad en caso de fallos del sistema.	

Tabla T.21 Tarea de implementación # 21

Pruebas de Caja negra

Id del escenario	Escenario	Nombre	CI	Respuesta del sistema	Resultado de la prueba
EC 1	"Trabajador modificado	"Luis Ernesto"	"97020804615"	El trabajador se ha modificado correctamente	Respuesta Satisfactoria

	correc-tamente”				
EC 2	“Error al modi-ficar el trabaja-dor”	“Alberto”	“76940856”	El usuario no se modificó satisfac-toriamente por-que el CI es inco-rrecto	Respuesta Satisfacto-ria

Prueba de caja negra, Tabla 7: Modificar trabajador, escenario Modificar trabajador

Id del escenario	Escenario	Nombre	Ci	Respuesta del sistema	Resultado de la prueba
EC 1	“Trabajadores listados correc-tamente”	“ismael”, “gonzalo”	“97020808567”, “97020808542”	Trabajadores lis-tados satisfacto-riamente	Respuesta Satisfactoria
EC 2	“La lista de tra-bajadores está vacía”	“ ”	“”	No hay trabajado-res registrados en la base de datos.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 8: Listar trabajador.

Id del escenario	Escenario	Ci	Fecha de alta	Respuesta del sistema	Resultado de la prueba
EC 1	“Trabajador dado de alta”	“97020808456”	“7/11/2015”	Trabajador dado de alta satisfacto-riamente.	Respuesta Satisfactoria
EC 2	“Error al dar de alta”	“9702080845 ”	“9/06/2005”	Error al dar de alta , Ci incorrecto.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 9: Dar baja al trabajador

Id del escenario	Escenario	Tabla	Formato	Respuesta del sistema	Resultado de la prueba

EC 1	"Documento exportable realizado con éxito"	"trabajadores"	"pdf"	El usuario administrador se ha modificado satisfactoriamente	Respuesta Satisfactoria
EC 2	"Error al realizar documento exportable"	"trabajadores "	""	Formato necesario para realizar el reporte.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 10: Hacer reportes exportables

Id del escenario	Escenario	Filtro	Nombres	Respuesta del sistema	Resultado de la prueba
EC 1	"Trabajadores listados por sexo correctamente"	"femenino"	"Maria", "Rosario", "Miriam"	Trabajadores filtrados correctamente.	Respuesta Satisfactoria
EC 2	"Error al listar por sexo"	" "	""	"" No es un filtro valido para el sexo	Respuesta Satisfactoria

Prueba de caja negra, Tabla 11: Listar trabajadores por sexo

Id del escenario	Escenario	Rango de edades	Nombres	Respuesta del sistema	Resultado de la prueba
EC 1	"Trabajadores listados por rango de edades correctamente"	"25-67"	"Pedro", "Joana", "Marissa"	Trabajadores filtrados correctamente.	Respuesta Satisfactoria
EC 2	"Error al listar por edades"	" "	""	Se necesita un rango de edades válido.	Respuesta Satisfactoria

EC 3	"Error al listar por edades"	"-4-68 "	""	Se necesita un rango de edades válido.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 12: Listar trabajadores por rango de edades.

Id del escenario	Escenario	Nivel educacional	Nombres	Respuesta del sistema	Resultado de la prueba
EC 1	"Trabajadores listados por nivel educacional correctamente"	"12 grado"	"Julian", "Aurora", "Tania"	Trabajadores filtrados satisfactoriamente	Respuesta Satisfactoria
EC 2	"Error al listar por nivel educacional"	" "	""	El filtro establecido no es válido.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 13: Listar trabajadores por nivel educacional.

Id del escenario	Escenario	Categoría ocupacional	Nombres	Respuesta del sistema	Resultado de la prueba
EC 1	"Trabajadores listados por categoría ocupacional correctamente"	"oficinista"	"Julian", "Aurora", "Tania"	Trabajadores filtrados satisfactoriamente.	Respuesta Satisfactoria
EC 2	"Error al listar por la categoría ocupacional"	" "	""	El filtro establecido no es válido.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 14: Listar trabajadores por categoría ocupacional.

Id del escenario	Escenario	Cursando estudios	Nombres	Respuesta del sistema	Resultado de la prueba

EC 1	“Trabajadores listados por estudios correctamente”	“Ingeniería mecánica”	“Julian”, “Aurora”, “Tania”	Trabajadores filtrados satisfactoriamente.	Respuesta Satisfactoria
EC 2	“Error al listar por estudios”	“ ”	“”	El filtro establecido no es válido.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 15: Listar trabajadores que cursan estudios.

Id del escenario	Escenario	Categoría científica	Nombres	Respuesta del sistema	Resultado de la prueba
EC 1	“Trabajadores listados por categoría científica correctamente”	“Licenciado”	“Julian”, “Aurora”, “Tania”	Trabajadores filtrados satisfactoriamente.	Respuesta Satisfactoria
EC 2	“Error al listar por categoría científica”	“ ”	“”	El filtro establecido no es válido.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 16: Listar trabajadores por categoría científica.

Id del escenario	Escenario	Categoría docente	Nombres	Respuesta del sistema	Resultado de la prueba
EC 1	“Trabajadores listados por categoría docente correctamente”	“Titular”	“Julian”, “Aurora”, “Tania”	Trabajadores filtrados satisfactoriamente.	Respuesta Satisfactoria
EC 2	“Error al listar por categoría docente”	“ ”	“”	El filtro establecido no es válido.	Respuesta Satisfactoria

Prueba de caja negra, Tabla 17: Listar trabajadores por categoría docente.

Id del escenario	Escenario	Demanda	Respuesta del sistema	Resultado de la prueba
EC 1	"Demanda de graduados existente"	"Ing Mecánico", "Ing Industrial"	Demanda de graduados	Respuesta Satisfactoria
EC 2	"No hay demanda de graduados"	" "	La tabla de demanda de graduados esta vacía	Respuesta Satisfactoria

Prueba de caja negra, Tabla 18: Demanda de fuerza de trabajo calificada.

Id del escenario	Escenario	Nombre de usuario	Contraseña	Respuesta del sistema	Resultado de la prueba
EC 1	"Usuario autenticado"	"lerodriguez"	"65soyadmin"	Usuario autenticado	Respuesta Satisfactoria
EC 2	"Error al autenticar"	"lerodrigues "	"65soyadmin"	Usuario o contraseña incorrecta .	Respuesta Satisfactoria

Prueba de caja negra, Tabla 19: Autenticar usuario.