Sistema para la gestión de la prenómina del Departamento de Tecnología de la Facultad 4

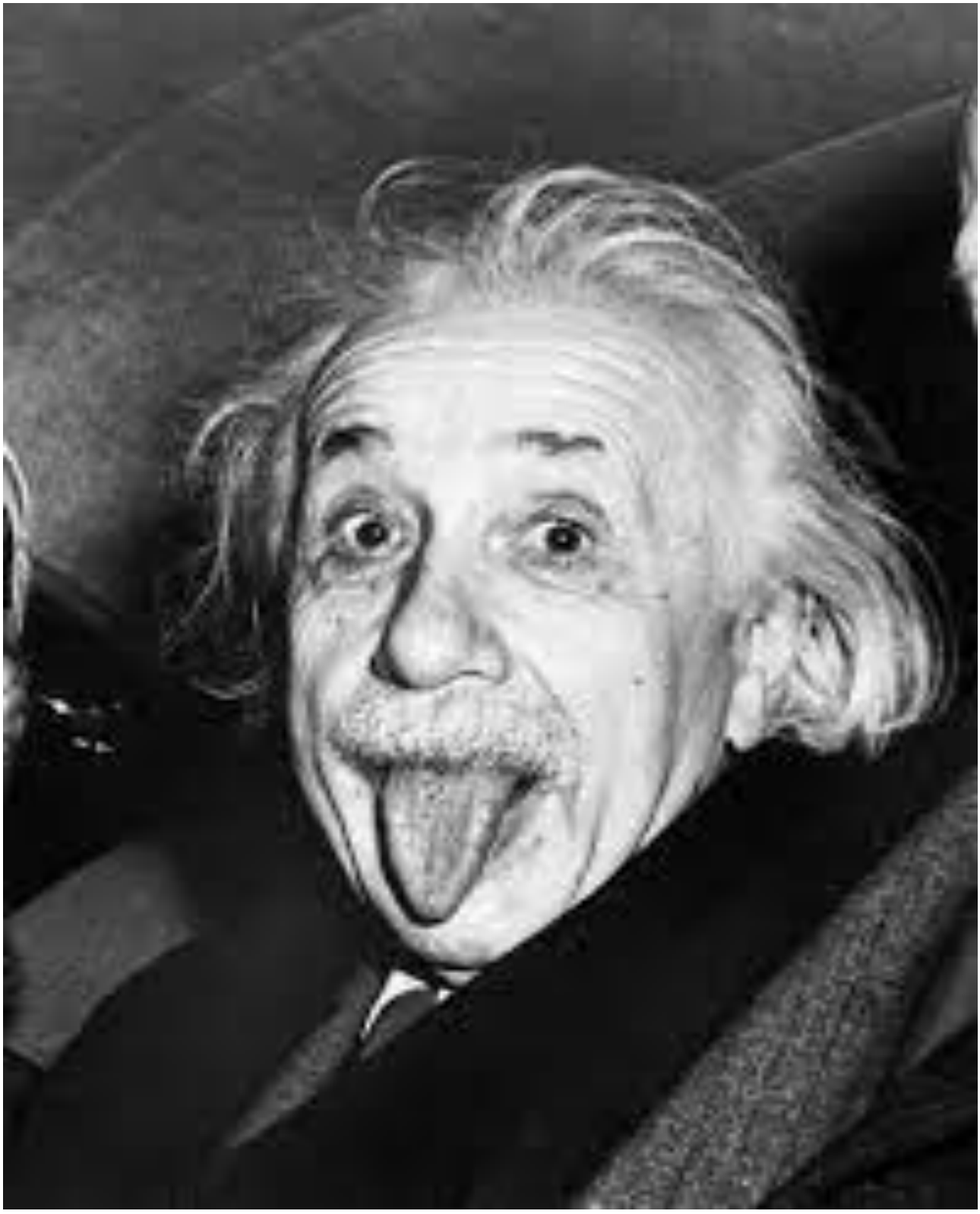
Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Alejandro David Monagas Torrecilla

Tutor: M.Sc. Yordankis Matos López

La Habana, diciembre 2022

"Año 64 de la Revolución"



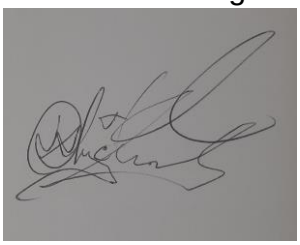
"Todo debe hacerse lo más simple posible, pero no más sencillo"

Albert Einstein

Declaración de Autoría

Declaro ser autor de la presente tesis que tiene por título “Sistema para la gestión de la prenómina del Departamento de Tecnología de la Facultad 4” y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los días 8 del mes de diciembre del año 2022

Alejandro David Monagas Torrecilla



M.Sc. Yordankis Matos López

Datos de Contacto

Nombre y apellidos del tutor: M.Sc. Yordankis Matos López

Institución: Universidad de las Ciencias Informática

Título: Ingeniero en Ciencias Informática

Correo electrónico: luguen@uci.cu

Agradecimientos

A Samira, gracias samy por hacerme fácil la UCI.

A mis padres sobre todas las cosas por todo lo que han hecho por mi.

A mi novia por siempre estar apoyandome y dandome ánimo.

A mis amigos y familiares por su preocupación.

A mi tutor por ser compresivo.

Dedicatoria

A mis padres, a uno de ellos por ser un hierro y estar presente siempre que lo he necesitado y al otro por ser la mejor y más cariñosa ladilla que se pueda tener.

Resumen

En la Facultad 4 se realiza manualmente toda la documentación referente a la gestión de la prenómina del Departamento de Tecnología, lo cual produce dificultades y lentitud de este proceso. La elaboración de la prenómina es una de las tareas que tiene el Departamento de Tecnología donde lleva por cada trabajador las ausencias, impuntualidades, licencias, vacaciones y otros factores que incidan en el tiempo a pagar. Este proceso para una mayor comodidad e informatización de la facultad es objetivo del presente trabajo de diploma desarrollar una aplicación web que permita su gestión administrativa. Para la implementación se tiene en cuenta el proceso de desarrollo de software guiado por la metodología XP, el servidor web Apache, el IDE NetBeans, el gestor de base de datos PostgreSQL, el administrador de base de datos pgAdmin y como lenguaje de programación Java. Entre las funciones del software se puede destacar la posibilidad de calcular de forma rápida y sencilla las horas de guardia nocturna y la selección de las incidencias del tiempo no laboral trabajado. Además, permitirá llevar un registro de todas las guardias que han efectuado los trabajadores. También permitirá un escaneo del solapin del trabajador para su asistencia o puntualidad de su turno laboral, donde se registrará en la tabla Hoja de Firma con el horario y la fecha en que fue escaneado. El sistema fue validado por pruebas unitarias y de aceptación, dando ocho no conformidades del tipo funcional y de interfaz, siete significativas y una no significativa, obteniéndose resultados excelentes.

Palabras Claves: *aplicación web; gestión; prenómina; proceso.*

Abstract

At Faculty 4, all the documentation related to the management of the Technology Department's payroll is done manually, which causes difficulties and slowness in this process. The preparation of the payroll is one of the tasks of the Technology Department, where it keeps for each worker the absences, absenteeism, leaves, vacations and other factors that affect the time to be paid. This process for greater convenience and computerization of the faculty is the objective of this diploma work to develop a web application that allows its administrative management. The implementation takes into account the software development process guided by the XP methodology, the Apache web server, the NetBeans IDE, the PostgreSQL database manager, the pgAdmin database administrator and the Java programming language. Among the software's functions, it is possible to quickly and easily calculate the hours of night duty and to select the incidents of non-working time worked. In addition, it will allow a record to be kept of all on-call hours worked by employees. It will also allow a scan of the worker's solapin for attendance or punctuality of their work shift, where it will be recorded in the Signature Sheet table with the time and date it was scanned. The system was validated by unit and acceptance tests, giving eight functional and interface nonconformities, seven significant and one non-significant, obtaining excellent results.

Keywords: *web application; management; pre-payroll; process.*

Índice de Contenido

Índice de Figuras

Índice tablas

Introducción

El sector de las Tecnologías de la Información y Comunicación (TIC) es actualmente uno de los más importantes, pues estas son consideradas como unas herramientas que permiten el fácil acceso a la información y una comunicación eficiente, rápida y clara. En este sentido, las TIC garantizan un aprendizaje didáctico con plena adquisición de conocimientos para el mejor desenvolvimiento del ser humano en su entorno, favoreciendo sectores vitales como la educación, la salud y las finanzas.(Rasobares 2018)

Cuba está consciente de que una sociedad para ser más eficaz, eficiente y competitiva debe aplicar la informatización en todas sus esferas y procesos. Ha identificado desde muy temprano la conveniencia y necesidad de dominar e introducir en la práctica social las TIC y lograr una cultura digital como una de las características imprescindibles del hombre nuevo, lo que facilitaría a nuestra sociedad acercarse más hacia el objetivo de un desarrollo justo, equitativo, sostenible y alcanzable. Cuba además de estar inmersa en una verdadera revolución educacional se ha dado la tarea de automatizar al máximo la sociedad cubana actual, desarrollando aplicaciones informáticas para contribuir al proceso de informatización de este sector, en empresas tanto nacionales como del exterior. Una de las instituciones que ha dado grandes pasos en este sentido es la Universidad de las Ciencias Informáticas (UCI).

La UCI tiene entre sus objetivos la informatización de sus procesos, con el propósito de mejorar el funcionamiento que tiene lugar en el centro. El proceso de la prenómina del Departamento de Tecnología actual de la universidad posee muchas fases manuales, por lo que representa una pérdida de recursos y es engorroso al no disponer de una herramienta específica para la publicación y consulta de estas prenóminas. Además, le resulta complicado a los principales actores consultar los datos laborales por el cúmulo de información almacenada en formato duro. Todo esto puede provocar problemas o atraso en la prenómina.

El área del Departamento de Tecnología de la Facultad 4 tiene establecido un registro primario por métodos manuales para anotar diariamente los datos relacionados con la asistencia de los trabajadores. La gestión administrativa del Departamento de Tecnología de esta facultad, se ha visto afectada por varias limitaciones donde se puede mencionar que todas las tareas correspondientes del centro están parcialmente informatizadas, debido a que solo se utilizan las herramientas de la suite de productos de Microsoft Office.

Las actividades de confección, reporte de incidencias por las diferentes claves, entrega y archivo de las evidencias que respalden cada una de estas actividades, el registro y control por separado de cada incidencia se realiza de forma manual. Estas actividades provocan que no se tomen elementos en cuenta por el corto lapso de tiempo para la conciliación de la prenómina; trayendo como consecuencia que en ocasiones se tuvieran que realizar actualizaciones de la prenómina de pago, incluso cuando varias de estas ya habían sido firmadas por la decana. Además, se le suma a esto que las actualizaciones antes mencionadas consumen tiempo y que en muchas ocasiones se entrega al tope de fecha o pasadas de estas.

La Facultad 4 y la UCI carece de aplicaciones informáticas capaces de realizar de forma automática la tarea anteriormente expuesta, lo que provoca la siguiente **situación problemática**:

- El proceso de remitir las incidencias de los trabajadores se dificulta y se hace muy complejo para el personal encargado de consultar las planillas archivadas y confeccionar manualmente el informe solicitado.
- La revisión manual de las hojas de firmas perteneciente a cada trabajador es engorrosa y consume mucho tiempo.
- Las prenóminas son confeccionadas a partir de las hojas de firmas. La persona encargada de crearlas es propensa a cometer errores al ser un trabajo engorroso y aumenta el esfuerzo con el número alto de trabajadores.

Por lo anterior se define el siguiente **problema de investigación**:

¿Cómo contribuir al proceso de gestión de la prenómina del Departamento de Tecnología de la Facultad 4 para mejorar su planificación y control?

Para solucionar el problema planteado se define como **objeto de estudio** la gestión de la planificación de la prenómina. Enmarcado en el **campo de acción** de los sistemas para la gestión de la planificación de la prenómina del Departamento de Tecnología de la Facultad 4.

Se propone como **objetivo general** del presente trabajo de diploma: desarrollar un sistema web para la gestión de la planificación de la prenómina del Departamento de Tecnología de la Facultad 4.

Para lograr el cumplimiento del objetivo general se definen las siguientes **tareas**

investigativas:

1. Elaboración del marco teórico a partir del estado del arte existente sobre el tema a investigar.
2. Realizar el análisis de sistemas homólogos a nivel nacional y global.
3. Selección de una metodología de desarrollo que sirva de guía para documentar el proceso de desarrollo de la aplicación web propuesta.
4. Análisis de las tecnologías, lenguajes y herramientas a utilizar para el desarrollo de la aplicación web.
5. Documentación de las funcionalidades de la aplicación web para su posterior implementación.
6. Definición de una arquitectura con el fin de obtener una visión general de la aplicación web que se propone.
7. Desarrollo de un sistema que contribuya a la gestión de prenóminas.
8. Análisis de los métodos de pruebas existentes para la validación de la aplicación web a partir de pruebas unitarias y de aceptación.

Resultado a obtener

Aplicación web para la gestión de la planificación y control de la prenómina del Departamento de Tecnología de la Facultad 4. Para la realización de este trabajo se contó con el apoyo de los siguientes **métodos científicos:**

Métodos teóricos:

- **Histórico-lógico:** permite mediante el análisis de la evolución de la gestión de la prenómina, determinar las principales características que debía poseer el sistema que se describe en la presente investigación, así como las herramientas y tecnologías a utilizar.
- **Analítico-sintético:** permite realizar el estudio teórico de la investigación facilitando el análisis de documentos y la extracción de los elementos más importantes relacionados con el proceso de desarrollo de sistemas de gestión de prenómina que hicieron posible la elaboración de conclusiones relacionadas con el objeto de estudio.

- **Inductivo-deductivo:** se aplica para la determinación de las generalidades y se parte del análisis de casos particulares encontrados, para arribar a razonamientos que permitieron la fundamentación teórica y elaboración del sistema presentado.

Método empírico:

- **Observación:** dentro de este método se utiliza las entrevistas que juegan un papel fundamental en la validación del problema a resolver. Permitiendo a un grupo de expertos examinar las facilidades que brinda la aplicación web para la gestión de la planificación y control de la prenómina del Departamento de Tecnología de la Facultad 4.

Estructura en capítulos

Capítulo 1. Fundamentación teórica: se analizan los conceptos fundamentales acerca de la gestión y planificación de la prenómina del Departamento de Tecnología de la Facultad 4 y aspectos elementales relacionados con las tecnologías a emplear en el desarrollo de la aplicación. Se realiza un análisis detallado de la metodología que guiará el proceso de desarrollo de software y las soluciones existentes actualmente en Cuba y en el mundo. Se definen los conceptos fundamentales, técnicas, tendencias, lenguajes, metodologías y herramientas, que se utilizaron en la construcción de la solución.

Capítulo 2. Descripción de la propuesta de solución: en este capítulo se describe una propuesta detallada de cómo deberá funcionar la aplicación web propuesta. Se presentarán los requisitos funcionales y no funcionales con los que debe cumplir la aplicación propuesta. Se presentan los resultados de las fases de exploración y planificación de la propuesta de solución. Se explicará la utilización de un conjunto de patrones dentro del diseño de la aplicación. En consecuencia, se obtendrá un conjunto de artefactos establecidos por la metodología XP.

Capítulo 3. Implementacion y prueba: en este capítulo se definen los estándares de codificación utilizados durante el desarrollo y se presenta la validación de los requisitos para fundamentar el funcionamiento correcto de la aplicación. Se validará el diseño de la solución a través de pruebas unitarias y de aceptación, con el propósito de validar la aplicación, la evaluación de su ejecución y se describen los resultados obtenidos.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción:

En el presente capítulo se realiza un análisis del proceso de elaboración de la prenómina del Departamento de Tecnología de la Facultad 4, se describe el marco teórico de las principales aplicaciones existentes a escala nacional y global. Además, se describen las tecnologías usadas actualmente para el análisis, diseño e implementación del sistema sobre las cuales se apoya la propuesta de solución a desarrollar.

1.1 Conceptos generales relacionados con la investigación

Para lograr una correcta comprensión es necesario definir conceptos claves que serán usados en la misma:

Control de asistencia: permite registrar de forma sencilla y efectiva los tiempos de llegada y salida de los empleados de la empresa, al igual que vela porque los tiempos de permanencia de los empleados en la misma se ajusten a las normas establecidas.(Lahoucine 2021)

Prenómina: Es el proceso que se realiza para conocer los días trabajados, las incidencias del tipo no laboral y sueldo bruto a pagar de cada trabajador, así como el cálculo de primas vacacionales y dominicales. [1]

Nocturnidad: Se interpreta como aumento salarial por tiempo de trabajo nocturno. Se puede clasificar según el tiempo en que se trabaja: hasta las 12:00am (nocturnidad1- 0.6) y (nocturnidad2- 1.15). Se expresa en horas para los trabajadores que tienen cargo Técnico en Ciencias Informáticas con un máximo en el mes de 32 horas para la nocturnidad 1 y de 64 horas para la nocturnidad 2. [2]

Sistema de información: es un conjunto de componentes que interactúan entre sí con un fin común. En informática, los sistemas de información ayudan a administrar, recolectar, recuperar, procesar, almacenar y distribuir información relevante para los procesos fundamentales y las particularidades de cada organización.(Herredo, Agius, y Romero 2019)

[1] https://www.collinsdictionary.com/es/diccionario/ingles/prenomina
[2] https://dpej.rae.es/lema/nocturnidad

1.2 Estado del arte asociado a los sistemas homólogos

Se describen los principales programas o páginas web que realizan el proceso de prenómina o gran parte de este, a escala nacional y global. Son de gran relevancia para el desarrollo de esta investigación, donde se hace un resumen de los datos de los sistemas basadas en las particularidades del negocio. Además se hace un compendio de las características de los sistemas basadas en las particularidades del negocio como se muestra en la tabla 1. Por último, los aportes vistos de los sistemas estudiados.

A continuación se describen los sistemas internacionales estudiados.

Infinite Consulting.SA: es una empresa especializada en la instalación de dispositivos biométricos. Cuenta con el software Mantra para la gestión de asistencia que le permite crear los reportes característicos de un control de asistencia. Sus principales beneficios son: permite crear reportes de asistencia dentro de los parámetros de tiempo que el cliente elija. Evitará posibles fraudes de usurpación de identidad muy comunes por el uso de tarjetas. Permite saber en tiempo real quien está o no, así como hacer seguimientos globales o personales para saber el desempeño asistencial de su personal.[3]

GrupoTress: Concentra toda la información relacionada con los empleados de la empresa para agilizar la administración, consulta, modificación y organización de la misma. GrupoTress entre otras funcionalidades brinda al cliente la posibilidad de, mediante una tarjeta magnética de uso personal llevar el control de la asistencia, retardos, horas extras de cada trabajador. Además, es capaz de generar de forma automática la pre-nómina. Brinda informes de horas trabajadas durante el periodo de nómina. Manejo de horas de comida y descanso, entre otros. Además, proporcionan una gran cantidad de reportes con el objetivo de optimizar el control empresarial.[4]

A continuación se describen los sistemas nacionales estudiados.

Fastos (Sistema Integral para la gestión de Recursos Humanos): El sistema de Recursos Humanos, está formado por los módulos Configuración, Personal, Capacitación, Cuadros, y Evaluación de Desempeño [5] . Permite controlar las informaciones fundamentales de los empleados de una entidad, también realizar varios procesos y

[3] http://www.infiniteconsulting.com
[4] https://tress.com.mx/

[5] http://www.rodasxxi.cu

operaciones que son inherentes al área de recursos humanos, tales como:

- **Registro de los empleados:** se guardan los datos de los empleados, así como informaciones referentes a los reportes de vacaciones, certificados médicos, licencias, resolución.
- **Control de la plantilla:** Permite establecer la estructura organizativa de las plazas de la entidad.
- **Control de asistencia:** lo cual incorpora el control de claves de asistencias, turnos de trabajos, horarios, tarjeta de asistencia e incidencias de cada empleado.
- Permite acoplar relojes (RTA 600) para actualizar la información de la tarjeta de asistencia de forma automática.
- **Informes y modelos:** Permite obtener un total de 56 informes, por ejemplo, cierre del periodo, análisis de fondo de tiempo, estadísticos, entre otros.

Rodas XXI: Sistema Integral Económico Administrativo desarrollado por la empresa cubana CITMATEL. El sistema es capaz de guardar por trabajador los pagos y retenciones fijas que se realizan a cada uno de ellos por lo que para la confección de las nóminas cada mes sólo es necesario actualizar las incidencias que correspondan y todo el trabajo posterior de cálculo es realizado de forma automática. Este módulo permite el control, planificación y gestión de la actividad de recursos humanos aplicable en todas las entidades[6]. Incluye Administración de personal y Cuadros. Cuenta con ocho módulos:

- **Contabilidad:** Incluye la importación de comprobantes generados por las operaciones en el resto de los módulos permitiendo su revisión antes de ser traspasados al Mayor General, es posible incluso realizar ajustes a los comprobantes importados antes de traspasarlos. Este módulo le brinda, mediante una opción, la posibilidad de revertir comprobantes de forma automática facilitando la realización de ajustes.
- **Finanzas:** Este subsistema permite tener el registro de cheques tanto emitidos, como recibidos, así como llevar el registro de otros instrumentos de pago y efectuar las operaciones de cobros y de pagos. Además, le permite tener un control de dietas, reembolso, vales para pagos menores y el registro de ingresos. Usted podrá llevar los submayores bancarios, realizar la conciliación bancaria y tener todo el

[6] https://www.desoft.cu/es/productos/

control de caja con la posibilidad de realizar el arqueo correspondiente.

- **Activos fijos:** le permite tener un control detallado de los activos fijos de su entidad, realizando en el mismo momento que se registra un movimiento, su contabilización. Se pueden realizar todo tipo de operaciones de activos fijos con facilidad en el momento que se desee, generando el documento asociado al movimiento de que se trate de forma automática.
- **Nóminas:** brinda grandes facilidades a las entidades para el cálculo y emisión de sus nóminas. Se pueden calcular y emitir todos los tipos de nóminas que se utilizan en nuestro país.
- **Inventario:** le permite tener un control detallado de los inventarios de su entidad, realizando en el mismo momento que se registra un movimiento, su contabilización.
- **Facturación:** Permite Pre facturar, Facturar y Re facturar.
- **Recursos humanos:** Este módulo permite el control, planificación y gestión de la actividad de recursos humanos aplicable en todas las entidades. Incluye Administración de personal y Cuadros.
- **Tele Cobranzas:** permite organizar estratégicamente la función de cobros y efectuar un seguimiento más estrecho de las gestiones con el cliente.

A continuación, se muestra la tabla 1 donde se puede apreciar un resumen de las características distintivas para el proceso de prenómina en relación con los sistemas antes estudiados.

Tabla 1. Sistemas similares. Fuente:(Elaboración propia)

Tabla comparativa				
Sistemas	**Licencia**	**Código abierto**	**Dominio de aplicación (web)**	**Realiza Prenómina**
GrupoTress	Privativo	No	No	Si
Infinite Consulting.SA	Privativo	No	No	No

Fastos	Privativo	No	No	No
Rodas XXI	Privativo	No	No	Si

Después de revisar los principales sistemas que informatizan el proceso de prenómina, se ha llegado a los siguientes resultados pues se percibe que todos son privativos por lo que no es posible obtener sus códigos, 2 no permiten generar la prenómina y ninguno es de dominio web.

Por todo lo anterior planteado se hace necesario el desarrollo de una herramienta informática nueva. Se identificaron funcionalidades que se pueden utilizar para el desarrollo de la misma, como son: la generación de la prenómina en formato pdf, el registro de la información asociada a la prenómina y la implementación de una aplicación donde se encuentre centralizada toda la información.

1.3 Herramientas y tecnologías

Las herramientas de desarrollo de software son productos informáticos que dan soporte a una tarea concreta dentro de las actividades de desarrollo de software para facilitar y asegurar la entrega de un sistema con calidad. Estos programas y aplicaciones pueden ser utilizados por muchas personas y se caracterizan por ser de fácil uso y ofrecer intercambio de información y conocimientos (Gómez 2021). A continuación, se describen y fundamentan la selección de las herramientas y tecnologías a utilizar en el desarrollo de la propuesta solución.

1.3.1 Lenguajes de programación

JAVA: es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de Lenguaje de Programación C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. Permite gestionar datos de fecha y hora de forma mucho mas natural y fácil de comprender (López 2019).

Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código máquina nativo también es posible. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución,

aunque la ejecución directa por hardware del bytecode por un procesador Java también es posible. La implementación original y de referencia del compilador, la máquina virtual y las bibliotecas de clases de Java fueron desarrolladas por Sun Microsystems en 1995. Desde entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a través del Java Community Process, otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre (López 2019).

Python: es un lenguaje de programación interpretado creado por Guido van Rossum en el año 1991. Se compara habitualmente con TCL, Perl, Scheme, Java y Ruby. En la actualidad Python se desarrolla como un proyecto de código abierto, administrado por la Python Software Foundation. La última versión estable del lenguaje es la 3.0. Python es considerado como la "oposición leal" a Perl, lenguaje con el cual mantiene una rivalidad amistosa. Los usuarios de Python consideran a éste mucho más limpio y elegante para programar. (Quintero 2022)

Permite dividir el programa en módulos reutilizables desde otros programas Python. Viene con una gran colección de módulos estándar que se pueden utilizar como base de los programas (o como ejemplos para empezar a aprender Python). También hay módulos incluidos que proporcionan E/S de ficheros, llamadas al sistema, sockets y hasta interfaces a GUI (interfaz gráfica con el usuario). Por ser un lenguaje de programación interpretado ahorra un tiempo considerable en el desarrollo del programa, pues no es necesario compilar ni enlazar. El intérprete se puede utilizar de modo interactivo, lo que facilita experimentar con características del lenguaje, escribir programas desechables o probar funciones durante el desarrollo del programa. (Quintero 2022)

JavaScript: es un lenguaje de programación que se puede utilizar para construir sitios web y para hacerlos más interactivos. Aunque comparte muchas de las características y de las estructuras del lenguaje Java, fue desarrollado independientemente. El lenguaje JavaScript puede interactuar con el código HTML, permitiendo a los programadores web utilizar contenido dinámico. Por ejemplo, hace fácil responder a los acontecimientos iniciados por usuarios (como introducción de datos en formularios) sin tener que utilizar interfaz de entrada común (CGI). El lenguaje JavaScript es de código abierto, por lo que cualquier persona puede utilizarlo sin comprar una licencia. (Gómez 2021) (Marijn 2018)

PHP: es un lenguaje de programación interpretado, diseñado originalmente para la

creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor, pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica. (Aguirre 2022) (Nixon 2019)

Ventajas

- Es un lenguaje multiplataforma.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad.
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite las técnicas de Programación Orientada a Objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.

Lenguaje seleccionado

Luego del análisis de los lenguajes anteriores se decide utilizar para el desarrollo del sistema el lenguaje **JAVA 8**. Por tener una programación orientada a objeto, ser de fácil uso y ofrecer intercambio de información, tiene un modelo de objetos más simple y elimina herramientas de bajo nivel y gestiona datos de fecha y hora de forma bastante natural y facil de comprender. Además es uno de los lenguajes más usados en la universidad, del que se posee una amplia documentación y el que domina más el desarrollador.

1.3.2 Otros lenguajes utilizados

CCS3: las hojas de estilo en cascada (Cascading Style Sheets, CSS) son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). Permite separar la estructura de un documento de su presentación. La información de estilo puede ser adjuntada tanto como un documento separado o en el mismo documento HTML. En este último podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo

"style". (Gauchat 2018)
Las Ventajas de utilizar CSS son:

- Control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.
- Los Navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad.
- Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario.
- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.

HTML 5: HTML (Hyper Text Markup Language) en su versión 5, no sólo trata de incorporar nuevas etiquetas o eliminar otras, sino que supone mejoras en áreas que hasta ahora quedaban fuera del lenguaje y para las que se necesitaba utilizar otras tecnologías. Los cambios comienzan añadiendo semántica y accesibilidad implícitas, especificando cada detalle y borrando cualquier ambigüedad. Se tiene en cuenta el dinamismo de muchos sitios webs, donde su aspecto y funcionalidad son más semejantes a aplicaciones webs que a documentos. También cuenta con nuevas APIs para interfaz de usuario: temas tan utilizados como el "drag&drop" (arrastrar y soltar) en las interfaces de usuario de los programas convencionales, serán incorporadas al HTML 5 por medio de un API(Gauchat 2018).

Se usaron HTMl5 y CSS3 porque además de las ventajas anteriormente planteadas permite al diseñador crear un sitio web compatible con todos los navegadores, son lenguajes de los que se tiene una amplia documentación y se pueden usar sin conexión a Internet.

1.3.3 Frameworks de desarrollo

Spring Boot: busca que el desarrollador solo se centre en el desarrollo de la solución, olvidándose por completo de la compleja configuración que actualmente tiene Spring Core para poder funcionar. En Spring Boot es muy fácil desarrollar aplicaciones basadas en Spring usando Java o Groovy y reduce mucho tiempo de desarrollo y aumenta la productividad. Además, evita escribir mucho código repetitivo, comentarios y configuración

XML y proporciona un servidor HTTP incorporado, como Tomcat, Jetty para desarrollar y probar aplicaciones web con mucha facilidad. Proporciona herramientas de CLI (interfaz de línea de comandos) desde el símbolo del sistema para desarrollar y probar aplicaciones Spring Boot (Java o Groovy) de manera muy fácil y rápida. Como micro marco, Spring Boot aún está lejos de la realización de microservicios. Spring Boot es solo para mejorar la eficiencia y la productividad del desarrollo. No se proporcionan las funciones de soporte de registro y descubrimiento de servicio correspondientes, y la función de supervisión proporcionada por su propio actuador también necesita estar conectada con la supervisión existente. No existe una solución de gestión y control de seguridad de soporte. (Hinkula 2022)

Vue.js: es un framework progresivo para construir interfaces de usuario. A diferencia de otros frameworks monolíticos, Vue.js está diseñado desde cero para ser utilizado incrementalmente. La librería central está enfocada solo en la capa de visualización, y es fácil de utilizar e integrar con otras librerías o proyectos existentes. Por otro lado, Vue.js también es perfectamente capaz de impulsar sofisticadas Single-Page-Aplicaciones cuando se utiliza en combinación con herramientas modernas y librerías de apoyo. Algunas de las ventajas de Vue.js es que se tarda menos tiempo en aprender los básico, para poder desenvolverte con el framework que con otros. El código de los componentes queda mucho más ordenado y limpio en Vuejs. A diferencia de React con sus archivos jsx, en los archivos vue puedes escribir html y css sin tener que hacerlo dentro de javascript. Está todo perfectamente separado. Comunidad en crecimiento. Aunque Vue no tiene una comunidad tan grande como React y Angular, su comunidad ha tenido un crecimiento espectacular en los últimos tiempos y se según las encuestas va a ir a más.

Una de las características más importantes de Vue.js es el trabajo con componentes. Un componente Vue.js, en términos simples, es un elemento el cual se encapsula código reutilizable. Dentro de un componente podremos encontrar etiquetas HTML, estilos de CSS y código JavaScript. Los componentes nos permiten desarrollar proyectos modularizados y fáciles de escalar, se puede reemplazar un componente por otro de una forma muy sencilla, como si de piezas de lego se tratasen.[7]

JavaServer Faces: es una tecnología y framework para aplicaciones Java basadas en

[7] https://vuejs.org/guide/introduction.html#what-is-vue

web que simplifica el desarrollo de interfaces de usuarios en aplicaciones Java EE. **JSF** usa JavaServer Pages (JPS como la tecnología que permite hacer el despliegue de las páginas, pero también se puede acomodar a otras tecnologías como XUL (acrónimo de XML-based User-interface Language - lenguaje basado en XML para la interfaz de usuario) JSF incluye:

- Un conjunto de APIs para representar componentes de una interfaz de usuario y administrar su estado, manejar eventos, validar entrada, definir un esquema de navegación de las páginas y dar soporte para internacionalización y accesibilidad.
- Un conjunto por defecto de componentes para la interfaz de usuario.
- Dos bibliotecas de etiquetas personalizadas para JavaServer Pages que permiten expresar una interfaz JavaServer Faces dentro de una página JSP.
- Un modelo de eventos en el lado del servidor.
- Administración de estados.
- Beans administrados.

Existen diferentes versiones de Java Server Faces entre las que se encuentra PrimeFaces. Esta es una biblioteca de componentes para JSF de código abierto que cuenta con un conjunto de componentes enriquecidos que facilitan la creación de las aplicaciones web. PrimeFaces está bajo la licencia de Apache License V2. (Galez 2018)

Vaadin: es un framework para el desarrollo de la capa de presentación de aplicaciones web en Java, que ofrece una interfaz de escritorio tradicional, sin necesidad de usar JavaScript ni JSON ni XML ni siquiera HTML.Está orientado a componentes y eventos.Se programa en el servidor. La definición del interfaz (páginas) se realiza en el servidor utilizando Java (se elimina JavaScript). Todo se compila y se puede depurar. Trae una serie de componentes de interfaz definidos (parte servidor), y permite crear nuevos por composición de los que ya hay incluyendo llamadas al negocio. Además, de permitir extender un widget para hacer uno propio (parte cliente), así como crear uno nuevo, usando GWT (compilador de Google para elementos de la capa de presentación). Este framework puede integrarse con cualquier otro que sea de Java, como JEE (CDI), Spring, Guice entre otros. Puede exportar los componentes propios e importar componentes

desarrollados por terceros mediante un sistema de Add-Ons. Ofrece un plugin para Eclipse para generar proyectos, compilar widgetset (GWT).[8]

Hibernate: es una herramienta de Mapeo Objeto-Relacional (ORM) para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) o anotaciones en los beans de las entidades que permiten establecer estas relaciones (Martínez 2021). Esta herramienta presenta las siguientes características:

- Hibernate es de software libre.
- Permite al desarrollador detallar cómo es su modelo de datos, qué relaciones existen y qué forma tienen.
- Manipula los datos de la base operando sobre objetos, con todas las características de la POO.
- Convierte los datos entre los tipos utilizados por Java y los definidos por SQL.
- Genera las sentencias SQL y libera al desarrollador del manejo manual de los datos que resultan de la ejecución de dichas sentencias, manteniendo la portabilidad entre todos los motores de bases de datos con un ligero incremento en el tiempo de ejecución.

Struts: framework de aplicación web open source desarrollado por Apache. Struts está basado en el patrón MVC. Se utiliza para construir aplicaciones Web basadas en servlets y JSP que pueden ejecutarse en cualquier contenedor de servlets incluyendo los servidores de aplicaciones J2EE. Mediante Struts se hace uso de patrones de diseño J2EE que son útiles en la construcción de aplicaciones J2EE. Y puesto que Struts sigue los patrones incluidos en el MVC, este framework es relativamente simple de usar y aprender. Struts proporciona al desarrollador un conjunto de etiquetas JSP personalizadas que facilitan la integración del framework con las páginas JSP.
Las principales características de Struts son las siguientes:

- Modelo Correcto. Permite modelar las aplicaciones basadas en JSP y servlets mediante el uso del patrón de diseño MVC.

[8] https://vaadin.com/blog

- Objetos “listos para usar”. Struts incorpora el concepto de objetos “listos para usar” a través de ficheros de configuración en XML.
- Patrones de Diseño pre-construidos. Struts incluye patrones de diseño internos en el framework.
- Por lo que no se ha de preocupar de crear los propios patrones de diseño.
- Características Extras. Incorpora características extras como validación, internacionalización.

Dentro de las ventajas de Struts se encuentran:

- El transporte automático de los datos introducidos en el cliente (JSP) hasta el controlador (Action) mediante formularios (ActionForm).
- Transporte automático de los datos enviados por el controlador (Action) a la parte de presentación (JSP) mediante formularios (ActionForm).
- Implementa la parte común a todas las aplicaciones en la parte de Controlador (ActionServlet); la parte particular de cada aplicación es fácilmente configurable (struts-config.xml).
- La separación de los componentes en capas (MVC) simplifica notablemente el desarrollo y su mantenimiento.

A pesar de los múltiples beneficios de usar Struts, este presenta una serie de desventajas como el hecho de no abarcar todas las capas de la aplicación web (deja fuera la capa de negocio y la capa de persistencia). Esto hace que la interfaz entre Struts y estas capas no esté tan automatizado, convirtiendo los accesos a los datos (DAO) en monótonos de desarrollar. (García 2018)

Frameworks seleccionados

Luego del análisis de los frameworks anteriores atendiendo a su comodidad, integración y a la amplia documentación existente se decide utilizar para el desarrollo del sistema los siguientes:

- Spring Boot 3.0.0
- Vue.js 3

1.3.4 Sistema Gestor de Base de Datos

Un sistema gestor de base de datos se define como el conjunto de programas que

administran y gestionan la información contenida en una base de datos. Ayuda a realizar las siguientes acciones: Definición de los datos, Mantenimiento de la integridad de los datos dentro de la base de datos, Control de la seguridad y privacidad de los datos, Manipulación de los datos.

PostgreSQL 9.4

Es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia Distribución De Software De Berkeley o Berkeley Software Distribution (BSD) y con su código fuente disponible libremente. Es un sistema de gestión de bases de datos de código abierto potente en el mercado y en sus últimas versiones tiene todas las funcionalidades que otras bases de datos comerciales puedan tener. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Sus características técnicas lo hacen uno de los gestores de bases de datos más potentes y robustos del mercado. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.(ByKibet 2022)

1.3.5 Administrador de Base Datos

pgAdmin4 v6.1: es la herramienta oficial para administrar las bases de datos en PostgreSQL. Nos permite desde hacer búsquedas SQL hasta desarrollar toda nuestra base de datos de forma muy fácil e intuitiva; directamente desde la interfaz gráfica. Con pgAdmin crear una nueva base de datos y definir sus propiedades es muy sencillo, por lo que esto permite que tanto principiantes como expertos se sientan cómodos con el sistema, también se pueden crear respaldos, restaurar la base de datos o ejecutar tareas de mantenimiento de forma muy sencilla, los usuarios podrán seguir accediendo a los datos durante el proceso. PostgreSQL te permite desarrollar bases de datos relacionales robustas y eficientes. (Perez 2021)

1.3.6 Entorno de Desarrollo Integrados (IDE)

NetBeans 8.2: es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE es un producto libre y gratuito sin restricciones de uso. NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios,

una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo.

Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software. (Bai 2022)

1.3.7 Servidor Web

Apache 2.0: Apache es un servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos (HTTP 1.1). Su flexibilidad, potencia, seguridad y bajo costo lo han convertido en el número uno de manera indiscutida (Kabir 2020). Otras de sus características son:

- Es multiplataforma.
- Es un servidor de web conforme al protocolo HTTP/1.1.
- Modular: puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, y con las API de programación de módulos, para el desarrollo de módulos específicos.
- Basado en hebras.
- Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos.
- Se desarrolla de forma abierta.
- Extensible: gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP, un lenguaje de programación del lado del servidor.

1.4 Metodologías de desarrollo de software

Metodología es un proceso de software detallado y completo. Las metodologías de

desarrollo de software se han clasificado en: ágiles y tradicionales. Las metodologías ágiles enfatizan la comunicación con el cliente mientras que suelen ser señaladas por la falta de documentación técnica. Las tradicionales o pesadas son aquellas con mayor énfasis en la planificación y control del proyecto (Pressman 2020). Entre las metodologías ágiles, las más empleadas son XP (eXtreme Programming), OpenUP (Open Unified Process), Scrum y Crystal. Así mismo se puede mencionar RUP (Rational Unified Process) y MSF (Microsoft Solution Framework) como las más usadas dentro de las metodologías pesadas.

1.4.1 Metodología XP (*Extreme Programming*, Programación Extrema)

Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre. A continuación, se presentan las características esenciales de XP organizadas en los tres apartados siguientes: Características, Valores y Reglas.(Beck 2019)

Características: se diferencia de las metodologías tradicionales principalmente porque está diseñada para adaptarse a los procesos de desarrollo en constante cambio. Se aplica de manera dinámica durante todo el ciclo de vida del software. Propone que exista una interacción constante entre el cliente y el equipo de desarrollo. Esta colaboración entre ambos será la que marque la marcha del proyecto y asegure su éxito.

Valores: XP brinda 4 valores que se describen a continuación:

- **Simplicidad:** es la base de la programación extrema. Se simplifica el diseño para agilizar el desarrollo y facilitar el mantenimiento.

- **Comunicación:** se realiza de diferentes formas. Para los programadores el código comunica mejor cuanto más simple sea. La comunicación con el cliente es fluida ya que el cliente forma parte del equipo de desarrollo y es quien decide qué características tienen prioridad y siempre debe estar disponible para solucionar dudas.

- **Retroalimentación:** al estar el cliente integrado en el proyecto, su opinión sobre el estado del proyecto se conoce en tiempo real.

- **Coraje o valentía:** muchas de las prácticas implican valentía. Una de ellas es siempre diseñar y programar para hoy y no para mañana. Esto es un esfuerzo para impedir el curso en el diseño y requerir demasiado tiempo y trabajo para implementar todo lo demás del proyecto.

Reglas de XP:

- **Desarrollo iterativo e incremental:** Pequeñas mejoras, unas tras otras.
- **Pruebas unitarias continuas:** Son frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.
- **Programación en parejas:** Se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que la mayor calidad del código escrito de esta manera -el código es revisado y discutido mientras se escribe- es más importante que la posible pérdida de productividad inmediata.
- **Frecuente integración del equipo de programación con el cliente o usuario:** Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- **Corrección de todos los errores antes de añadir nueva funcionalidad:** Hacer entregas frecuentes.
- **Refactorización del código:** Es decir, reescribir ciertas partes del código para aumentar su legibilidad y Mantenibilidad, pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- **Propiedad del código compartido:** en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.

- **Simplicidad del código:** es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

El ciclo de vida ideal de XP concibe 4 fases:

- Fase I Planeación
- Fase II Diseño
- Fase III Codificación
- Fase IV Prueba.

En resumen, se puede concluir que XP es una metodología cuyo objetivo es crear sistemas de alta calidad, basados en una estrecha interacción con los clientes, pruebas constantes y ciclos de desarrollo cortos.

1.4.2 Rational Unified Process (RUP)

RUP (del Inglés RationalUnifiedProcess), Proceso Unificado de Desarrollo, que es un proceso de desarrollo de software que junto con el Lenguaje Unificado de Modelado UML, forman la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. El Proceso Unificado es más que un simple proceso, provee un marco genérico de trabajo que puede especializarse para una gran variedad de sistemas de software. RUP en su modelación define como sus principales elementos: Trabajadores: Son los que realizan las actividades y son propietarios de elementos. Actividades: Tareas que tienen un propósito claro. Artefactos: Productos tangibles de un proyecto como: modelos, código fuente y ejecutables (Ivar, Booch, y Rumbaugh 2020). Un proyecto realizado siguiendo RUP se divide en cuatro fases:

- **Inicio:** Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.
- **Elaboración:** Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen.

- **Construcción:** Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario.

- **Transición:** El release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores. Las características fundamentales que se definen en el Proceso Unificado son: dirigido por casos de uso, iterativo e incremental y centrado en la arquitectura. Además de estas Está basado en componentes. Estos componentes a su vez están conectados entre sí a través de interfaces y utiliza el UML como notación básica.

El modelado de negocio de la metodología RUP ha permitido que las empresas puedan adquirir toda la información necesaria para un análisis del negocio actual y por ende identificar qué áreas se pueden mejorar.

1.4.3 SCRUM

Desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle. Scrum es una metodología para la gestión de proyectos. Define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos 10 años. Está especialmente indicada para proyectos con un rápido cambio de requisitos (Clark 2020). Sus principales características se pueden resumir en:

- El desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente.

- Reuniones a lo largo del proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración. Scrum, presenta solo tres roles principales, en los cuales se dividen todas las responsabilidades de manejo de un proyecto: ProductOwner (Hombre de Negocios o Dueño del Producto): Es quien representa a los interesados (stakeholders) y es parte de la compañía que solicita el producto.

- TheTeam (El Equipo): Es un equipo auto-gestionado, auto-organizado y multi-funcional, que incluye a los desarrolladores. Tiene la responsabilidad de entregar el producto con calidad.

- Scrum Master (Maestro Scrum o Facilitador): Se encarga de mantener los procesos y tareas de manera similar a un Jefe de Proyecto.

En resumen, la metodología Scrum es un proceso para llevar a cabo un conjunto de tareas de forma regular con el objetivo principal de trabajar de manera colaborativa, con este método de trabajo se pretende alcanzar el mejor resultado de un proyecto determinado.

1.4.4 Fundamentación del uso de la metodología XP

La metodología a utilizar en este trabajo será XP, después de haber realizado el estudio y análisis de algunas metodologías de desarrollo, la misma es adaptable al software a desarrollar, así como a las condiciones de trabajo, de forma general. Plantea la comunicación y satisfacción del cliente como requisito principal. Lo más importante en XP es definir los requerimientos y las pruebas de calidad.

Esta metodología es flexible cuando los requerimientos sufren cambios algo que sucede a menudo y permite administrarlos de forma óptima. Uno de sus objetivos principales es potenciar al máximo el trabajo en grupo, donde los jefes de proyecto, los clientes y desarrolladores son parte del equipo y están involucrados en el desarrollo del software.

Conclusiones parciales

Después de realizar un análisis del marco teórico de la investigación, se puede arribar a las siguientes conclusiones:

- El análisis de los sistemas informáticos permitió conocer funcionalidades a desarrollar y sentó la base a la necesidad de crear uno nuevo.
- La definición de los principales conceptos asociados al dominio de la presente investigación, permitió sentar las bases teóricas para el desarrollo de un sistema de gestión de la prenómina.
- El análisis de varias herramientas de trabajo arroja como resultado la selección de un entorno de desarrollo altamente integrado y compatible entre sí, facilitando el desarrollo del sistema.
- El análisis de las diferentes metodologías permitió seleccionar la metodología XP para guiar el proceso de desarrollo de software.

CAPÍTULO 2. DESCRIPCIÓN DE LA PROPUESTA DE SOLUCIÓN

Introducción

En este capítulo se analizan y se describen las características principales del sistema que se desea implementar, definidas a partir del de la metodología seleccionada para guiar el desarrollo de la propuesta de solución. Se muestran también los requisitos funcionales y no funcionales con los que deberá cumplir la aplicación web, así como algunas particularidades del diseño e implementación de dicha aplicación web

2.1. Descripción del proceso de negocio Prenómina

El sistema informático que se presenta, permitirá establecer un control constante de la asistencia de los trabajadores. Facilitará la toma de decisiones de los directivos a la hora de llevar a cabo la evaluación de desempeño y la gestión de las sanciones administrativas. Además, el sistema será capaz de ir confeccionando las hojas de firmas según se vaya registrando la asistencia o incidencia del trabajador. La propuesta será una valiosa herramienta de control que permitirán facilitar la confección de la Prenómina salarial para su posterior exportación.

2.2. Descripción de la propuesta de solución

La aplicación web a desarrollar permite la gestión de la prenómina del Departamento de Tecnología de la Facultad 4 y está diseñada de la siguiente manera: en la parte superior izquierda se muestra el nombre de la aplicación “GESPRE”, debajo de esta estructura se encuentran en la parte izquierda los vínculos a las diferentes interfaces que se mostrarán. En la aplicación sólo pueden entrar el usuario “ADMIN” que es el único en tener permiso para loguearse.

Los vínculos son: inicio, Gestionar Trabajadores, Gestionar Hojas de Firmas, Gestionar Incidencia, Gestionar Prenómina, Gestionar Clave y Firmar. Una vez que el usuario” ADMIN” se autentica se muestra la interfaz de inicio en la cual se entra a la página de inicio: En la tabla Gestionar Trabajadores se mostrará el listado de todos los trabajadores con sus respectivos nombres, apellidos, cargo, id expediente y un botón para eliminar trabajador. También en la parte superior de la tabla se encuentra el botón de Generar Trabajadores, que es el encargado de tomar todas las personas de los servicios UCI que pertenezcan al departamento de Tecnología de la Facultad 4.

En la tabla Gestionar Incidencia se mostrará el listado de las incidencias con las respectivas claves, descripciones, nombre y apellido del incidente, fecha, cantidad de días,

con dos botones para modificar los datos de una existente y eliminarla del listado. En la interfaz Gestionar Hoja de Firma se mostrará el listado de las hojas de firmas con su nombre y apellidos del trabajador, organismo, mes, número de expediente, año y tres botones para añadir exportar, agregar incidencia y eliminarla. También en la parte superior de la tabla se encuentra el botón de Generar Hojas Firmas, que es el encargado de crearle a todos los trabajadores sus respectivas hojas de firmas.

En la interfaz Gestionar Prenómina se observará el modelo correspondiente con todas las prenóminas donde se muestran su Organismo, Entidad, Área y fecha. También posee tres botones para exportar, modificar, eliminar y en la parte superior de la tabla el botón de agregar. Para agregar una Prenómina es necesario completar los campos Organismo, Entidad, Área, Fecha, Confeccionado por, Revisado por y Aprobado por. Una vez creada la prenómina, el sistema permite dar clip al botón de Generar prenómina donde este creará la prenómina con cada trabajador con respectiva hoja de firma y cada una de sus incidencias. Firmar, es la interfaz donde el trabajador ingresará su código de barra para acceder a su hoja de firma correspondiente y firmar el día laboral.

2.3. Fase de planeación

Levantamiento de Requisitos

Teniendo en cuenta los aspectos teóricos que identifican requisitos y que fueron abordados en el capítulo 1, se definieron los siguientes requisitos funcionales (RF) y requisitos no funcionales (RnF), haciéndose necesaria la utilización de algunas técnicas que sirvieron para verificar la veracidad de los objetivos propuestos para el desarrollo de esta aplicación, permitiendo que dieran satisfacción a las necesidades del cliente. Dentro de las técnicas utilizadas se encuentran las entrevistas, aplicadas a las personas encargadas de realizar la prenómina en el Departamento de Tecnología de la Facultad 4, que fueron consideradas proveedores válidos para las cuales va a ser desarrollada la aplicación. Estas entrevistas posibilitaron los debates, donde los clientes explicaron de forma clara la necesidad de la informatización del proceso de prenómina. Para la aplicación se determinaron los siguientes RF y RnF.

Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir. Constituyen una base para poder identificar lo que realmente debe hacer el sistema. Los mismos deben ser comprendidos tanto por los desarrolladores como por los clientes

(Pressman 2020). A continuación, se muestran los que la aplicación a desarrollar debe cumplir:

R1- Autenticar Usuario.

R2-Generar Trabajador.

R3-Eliminar Trabajador.

R4-Generar Hoja Firma.

R5-Eliminar Hoja Firma.

R6-Exportar Hoja Firma.

R7-Guardar Hoja Firma Predefinida

R8-Agregar Prenómina.

R9-Modificar Prenómina.

R10-Eliminar Prenómina.

R11-Guardar Prenómina Predefinida

R12-Exportar Prenómina

R13- Agregar Incidencia.

R14-Modificar Incidencia.

R15-Eliminar Incidencia.

R16-Agregar Clave

R17-Modificar Clave

R18-Eliminar Clave

R19-Buscar por código barra

Requisitos No Funcionales

Los requisitos no funcionales son las propiedades o cualidades que un sistema debe tener (Pressman 2020). Para obtener un óptimo funcionamiento del sistema se requiere la existencia de los siguientes requisitos en el servidor y las máquinas clientes que harán uso de la aplicación.

- **Requisitos de Software:**

  Cliente

  - Red activa.
  - Cualquier navegador web.

Servidor

-Servidor de Base de Datos PostgreSQL 9.1.

-Servidor Web apache-maven-3.6.3.

-Lenguaje JAVA 8.

- **Requisitos de Seguridad:**

  Se deberán establecer los permisos para el acceso a las funcionalidades de administración del sistema y garantizar que las mismas se visualicen de acuerdo al nivel de usuario que está activo. El producto deberá tener protección contra acciones no autorizadas o que puedan afectar la integridad de los datos. La información manejada por el sistema podrá ser accedida por las personas que tienen los permisos para ello, por lo que estará protegida de acceso no autorizado y divulgación.

- **Requisitos de Portabilidad:**

  El sistema deberá ser multiplataforma y permitir que se acceda a él desde cualquier navegador. La interfaz debe adaptarse a cualquier resolución de pantalla.

- **Requisitos de Interfaz:**

  El producto deberá ser legible y acatar las normas y estándares de la universidad. La interfaz del sistema se realizará mediante una página web y su diseño deberá estar orientado a una navegación sencilla y fácil de usar.

- **Usabilidad:**

  La interfaz no puede tener más de 15 pestañas. Tamaño de letra mayor de 10 puntos. El sistema deberá presentar una interfaz fácil de entender y usar. Agrupar vínculos y botones por grupos funcionales, además una uniformidad entre cuadros de texto y botones. Para interactuar con el sistema se requiere una preparación previa, permitiendo la fácil comprensión del mismo.

- **Rendimiento:**

  La aplicación debe contar con una velocidad óptima de carga y actualización de los datos de la página, no sobrepasando los 10 segundos. Debe tener tolerancia a fallos o exepciones. Debe soportar un mínimo de 5 usuarios.

2.3.1 Historias de Usuario

Las historias de usuarios(HU), es uno de los artefactos generados por la metodología XP

y se crean a partir de los requisitos funcionales definidos. A continuación, se muestran algunas de las historias de usuario con prioridad alta concebidas a partir de los requisitos funcionales del sistema. El resto de las historias de usuario se encuentran en el Anexo 1.

Tabla 2:Generar Trabajadores. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 02	**Nombre:** Generar Trabajadores
Usuario: Administrador	
Prioridad del Negocio: Alta.	**Nivel de Complejidad:** Media.
Punto de Estimación: 3	**Iteración Asignada:** 1
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar trabajadores se muestra un botón "Generar Trabajadores" que al hacerle clic se crean de forma automática.	
Observaciones: Si los servicios de la UCI están caídos este botón no funciona.	

Tabla 3:Agregar Prenómina. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 08	**Nombre:** Agregar prenómina.
Usuario: Administrador	
Prioridad del Negocio: Alta.	**Iteración Asignada:** 2
Punto de Estimación: 4	**Riesgo en el desarrollo:** Alto.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: Se presiona el botón "Agregar Prenómina", el sistema muestra la interfaz con una tabla prenómina con los campos a llenar (fecha, revisado por, aprobado por), también los campos predefinidos (organismo, entidad, área, confec. por) y el botón salvar, cancelar y predefinido guardar.	

Observaciones: Si presiona el botón salvar y ya existe una prenómina del mismo mes y año el sistema muestra el mensaje “Ya existe una prenómina de este mes” o de lo contrario se guarda y se muestra el mensaje “Guardado”, si presiona cancelar el sistema te lleva para la página “Gestionar Prenómina.

Matriz de trazabilidad

La matriz de trazabilidad es una tabla que relaciona cada uno de los requerimientos con el entregable que se haya solicitado. Este cuadro es de doble sentido. Te permite identificar qué resultado se alcanza a través de cada requisito y, a la vez, qué requisitos son los que permiten obtener un determinado entregable. En la tabla 4 se muestra en las columnas los requisitos funcionales y en las filas las historias de usuario, asociando cada requisito a una HU.

Tabla 4.Matriz de trazabilidad. Fuente:(Elaboración propia)

REQUISITO	Autenticar Usuario	G. Usuario	Eliminar Usuario	G.Hoja Firma	E.Hoja Firma	Exportar Hoja Firma	G. Hoja Firma Predefinida	A.Prenómina	M.Prenómina	E.Prenómina	Predefinida G.Prenómina	Exportar Prenómina	A.Incidencia	M.Incidencia	E.Incidencia	A.Clave	M.Clave	E.clave	B. código barra
HU																			
1	X																		
2		X																	
3			X																
4				X															
5					X														
6						X													
7							X												
8								X											
9									X										
10										X									
11											X								
12												X							
13													X						
14														X					
15															X				
16																X			
17																	X		
18																		X	
19																			X

La matriz de trazabilidad fue la base para controlar qué requisitos están validados, cuáles están pendientes o cuáles han sido rechazados. Permitió además identificar la línea base de requisitos correspondiente a desarrollar.

2.3.2 Estimación del esfuerzo por Historias de Usuario y Plan de

Iteraciones

En la metodología XP, la creación del sistema se divide en etapas para facilitar su realización. Por lo general, los proyectos de desarrollo constan de más de tres etapas, las cuales toman el nombre de iteraciones; de allí se obtiene el concepto de metodología iterativa. La duración ideal de una iteración es de una a tres semanas.

Para cada iteración se define un módulo o conjuntos de historias que se van a implementar. Al final de cada iteración se obtiene como resultado la entrega del módulo correspondiente, el cual debe de haber superado las pruebas de aceptación que establece el cliente para verificar el cumplimiento de los requerimientos (Cano 2022). La siguiente tabla muestra los resultados de estimación obtenidos por cada Historia de Usuario identificada, así como la prioridad, número y fin de iteración.

Tabla 5:Plan de Iteraciones. Fuente:(Elaboración propia)

Núm. de Iteración	Historia de Usuario	Prioridad	Puntos de Estimación		Fin de Iteración
1	Autenticar Usuario	Alta	4	14	2da y 3era Semana de junio.
1	Generar Trabajador	Alta	3		
1	Generar Hoja Firma	Alta	3		
1	Eliminar Trabajador	Media	2		
1	Eliminar Hoja Firma	Media	2		
2	Agregar Prenómina.	Alta	4	12	4ta y 1era Semana. de junio
2	Exportar Hoja Firma	Alta	2		
2	Exportar Prenómina	Alta	2		
2	Eliminar Prenómina	Alta	2		
2	Modificar Prenómina.	Media	2		
3	Agregar Incidencia	Alta	3	10	2da y 3era Semana. de Julio
3	Modificar Incidencia	Alta	2		

3	Eliminar Incidencia	Alta	2		
3	Agregar Clave	Alta	3		
4	Modificar Clave	Media	2	12	4rta y 5ta Semana. de Julio
4	Eliminar Clave	Media	2		
4	Guardar Hoja Firma Predefinida		2		
4	Guardar Prenómina Predefinida		2		
4	Buscar por código barra	Alta	4		

2.4 Fase de diseño

A diferencia de las metodologías pesadas, el diseño se realiza durante todo el tiempo de vida del proyecto, siendo constantemente revisado y muy probablemente modificado debido a cambios presentados. XP establece prácticas especializadas, que inciden directamente en la realización y elaboración del diseño de un software, sin embargo, no requiere que la representación del sistema sea mediante diagramas de clases basados en UML, sino que pueden emplearse indistintamente sencillos esquemas descritos en pizarras u otras técnicas como las tarjetas Clase-Responsabilidad-Colaboración (CRC).

2.4.1 Descripción de las tarjetas CRC

Las Tarjetas CRC son artefactos de diseño que se generan como parte del proceso XP. Permiten visualizar el sistema en términos de objetos que componen la aplicación, identificando clases candidatas, responsabilidades y la forma que colaboran entre ellas. Para la aplicación fueron especificadas, donde cada tarjeta debe corresponder a una o más Historias de Usuario (Tarapués, Ingrid, y Jorge 2022). Seguidamente se muestran algunas tarjetas CRC el resto de ellas están ubicadas en el Anexo 2.

Tabla 6:Prenómina.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: Prenomina.java	
Responsabilidades:	**Colaboraciones**

Contiene todas las variables correspondientes a las prenóminas, los metodos set y get y los datos correspondientes a los trabajadores y sus hojas de firmas.	Trabajador.java HojaFirma.java

Tabla 7:HojaFirmaController.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: HojaFirmaController.java	
Responsabilidades:	**Colaboraciones**
ObtenrHojasFirmas() ObtenrHojaFirmaByUser() SalvarHojaFirma() GenerarHojasFirmas() ActualizarHojaFirma() EliminarHojaFirma()	IHojaFirmaService.java HojaFirmaDTO.java IAsistenciaService.java Asistencia.java

Tabla 8:ServiceImplTrabajador.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: ServiceImplTrabajador.java	
Responsabilidades:	**Colaboraciones**
mapearDTO() mapearEntidad() GenerarTrabajadoresServicio() Eliminar()	ModelMapper.java ITrabajadorRepo.java ClienteAssetsUCIWSDL.java TrabajadorDTO.java Trabajador.java

2.4.2 Patrones arquitectónicos y de diseño

Un estilo arquitectónico es un patrón de diseño o práctica recurrente, expresa esquemas de organización estructural fundamentales para los sistemas de software. Proporcionan un conjunto de subsistemas predefinidos, especifican sus responsabilidades e incluyen guías y lineamientos para organizar las relaciones entre ellos. A continuación, se describen los que se han utilizado para la realización del presente trabajo de diploma

Patrón Arquitectónico

Los patrones arquitectónicos, o patrones de arquitectura, son patrones de diseño de software que ofrecen soluciones a problemas de arquitectura de software en ingeniería de software. Dan una descripción de los elementos y el tipo de relación que tienen junto con

un conjunto de restricciones sobre cómo pueden ser usados. Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de software, que consta de subsistemas, sus responsabilidades e interrelaciones (Fernández 2018). En comparación con los patrones de diseño, los patrones arquitectónicos tienen un nivel de abstracción mayor.

El patrón Modelo-Vista-Controlador (MVC) es una guía para el diseño de arquitecturas de aplicaciones que ofrezcan una fuerte interactividad con trabajadores. Este patrón organiza la aplicación en tres modelos separados, el primero es un modelo que representa los datos de la aplicación y sus reglas de negocio, el segundo es un conjunto de vistas que representa los formularios de entrada y salida de información, el tercero es un conjunto de controladores que procesa las peticiones de los trabajadores y controla el flujo de ejecución del sistema.(Fernández 2018)

Este patrón arquitectónico presenta varias ventajas:

- Hay una clara separación entre los componentes de un programa; lo cual permite implementarlos por separado.
- La conexión entre el Modelo y sus Vistas es dinámica; se produce en tiempo de ejecución, no en tiempo de compilación.

Se decidió utilizar este patrón para el desarrollo del sistema en cuestión debido a que con él la aplicación se puede desarrollar rápidamente, de forma modular y mantenible. Separar las funciones de la aplicación en modelos, vistas y controladores hace que la aplicación sea muy ligera. El diseño modular permite a los diseñadores y a los desarrolladores trabajar conjuntamente, así como realizar rápidamente el prototipado. Esta separación también permite hacer cambios en una parte de la aplicación sin que las demás se vean afectadas.

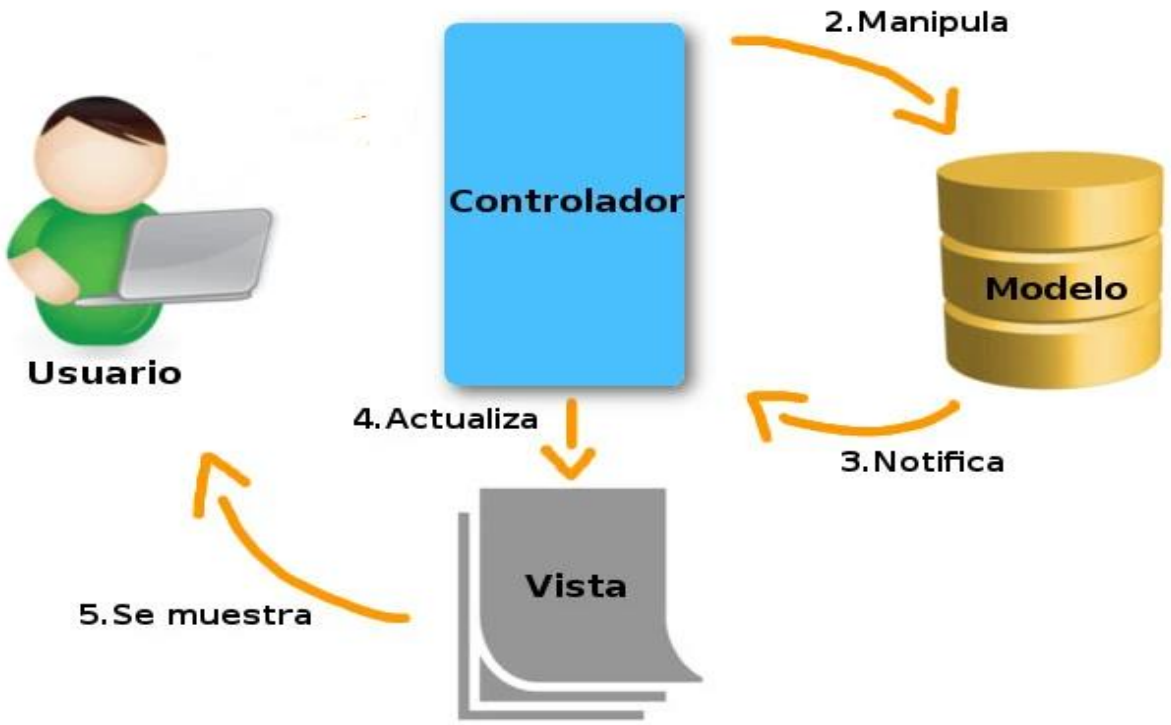


Figura 1:Modelo Vista Controlador. Fuente:(Fernández, 2018)

El modelo es quien administra el comportamiento y contiene los datos que manejara el sistema, los cuales han sido representados en formas de paquetes. El paquete modelo contiene las clases Trabajador, Prenomina, HojaFirma, Incidencia, Clave y Asistencia. El servicio usa los datos del modelo para devolvérselos al controlador, donde este tiene una dependencia del repositorio el cual es el puente para acceder a la base de datos. La vista es la encargada de mostrar la información a través de las interfaces como GestionTrabajadorsPage y GestionPrenominaPage. Mientras que el controlador es el responsable de implementar toda la lógica de negocio y la interacción entre la vista y el modelo como son las clases PrenominaController y HojaFirmaController.

Patrones de Diseño

Un patrón de diseño provee un esquema para refinar los subsistemas o componentes de un sistema de software, o las relaciones entre ellos. Describe la estructura comúnmente recurrente de los componentes en comunicación, que resuelve un problema general de diseño en un contexto particular. Son menores en escala que los patrones arquitectónicos, y tienden a ser independientes de los lenguajes y paradigmas de programación. Su aplicación no tiene efectos en la estructura fundamental del sistema, pero sí sobre la de un subsistema, debido a que especifica a un mayor nivel de detalle, sin llegar a la

implementación, el comportamiento de los componentes del subsistema.(Debrauwer 2018)

Patrones GRASP

Los patrones de diseño GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones. GRASP es un acrónimo que significa General Responsibility Assignment Software Patterns (Shvets 2021). El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos. Se pasa enumerarlos junto con su definición formal:

- **Experto**: Se encarga de asignar una responsabilidad al experto en información, o sea, aquella clase que cuenta con la información necesaria para cumplir la responsabilidad. Está presente en las clases entidades como Trabajador, HojaFirma que contiene todos sus métodos set y get para tener poca dependencia del resto de las clases.

- **Creador:** Permite creas una nueva instancia por la clase que usa directamente las instancias creadas del objeto, almacena o maneja varias instancias de la clase. Este patrón se observa en las clases TrabajadorController ya que ella construye la página cliente como Listar_Trabajador.

- **Bajo Acoplamiento:** Diseñar con el objetivo de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases este patrón se envidencia en las clases Trabajador, Prenomina, HojaFirma las cuales tienen una débil dependencia.

- **Alta Cohesión:** Asigna una responsabilidad de forma tal que la información que maneja una clase sea coherente y esté relacionada con la clase. Al evidenciarse el patrón “Bajo acoplamiento” también se evidencia este patrón en las clases TrabajadorController, HojaFirmaController, para no sobrecargar el sistema y no colapse además de evidenciarse en sus propias entidades HojaFirma y Trabajador.

- **Controlador:** Este patrón se evidencia en las clases controladoras, responsables de la interacción entre el Modelo y la Vista. Encargado de seleccionar la Vista mediante la acción del Usuario, proporcionándole toda la información necesaria

para construir la interfaz, ejemplo la clase TrabajadorController.

Patrones GOF

Creación: Corresponden a patrones de diseño software que solucionan problemas de creación de instancias. Nos ayudan a encapsular y abstraer dicha creación:

- **Builder** (constructor virtual): abstrae el proceso de creación de un objeto complejo, centralizando dicho proceso en un único punto.
    - Ejemplo: Clase entidad HojaFirma

Estructurales: Son los patrones de diseño software que solucionan problemas de composición (agregación) de clases y objetos:

- **Decorator** (Decorador): Añade funcionalidad a una clase dinámicamente.

```
/**
 *
 * @author Ale
 */
@RestController
@CrossOrigin("*")
@RequestMapping("/gespre/usuario")
public class UsuarioController {
```

Figura 2:Evidencia del Patrón. Fuente:(Elaboración propia)

- **Singleton:**

  El patrón singleton en Spring completa la segunda mitad de la oración, que proporciona un punto de acceso global BeanFactory. Pero no hay un control singleton desde el nivel del constructor, porque Spring administra objetos Java arbitrarios.

```
@Autowired
IHojaFirmaService firmaService;
@Autowired
IAsistenciaService asistService;
```

Figura 3:Patrón Singleton. Fuente:(Elaboración propia)

Comportamiento: Se definen como patrones de diseño software que ofrecen soluciones

respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan:

- **Observer** (Observador): Define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él.
- **Template Method** (Método plantilla): Define en una operación el esqueleto de un algoritmo, delegando en las subclases algunos de sus pasos, esto permite que las subclases redefinan ciertos pasos de un algoritmo sin cambiar su estructura.

2.5 Descripción de la base de datos

Un diagrama o modelo entidad-relación (DER) es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información, así como sus interrelaciones y propiedades (Pressman 2020). La figura muestra las tablas donde se almacena la información utilizada en la propuesta solución. La base de datos se encuentra normalizada, cumpliendo con las normas de diseño de base de datos establecida.

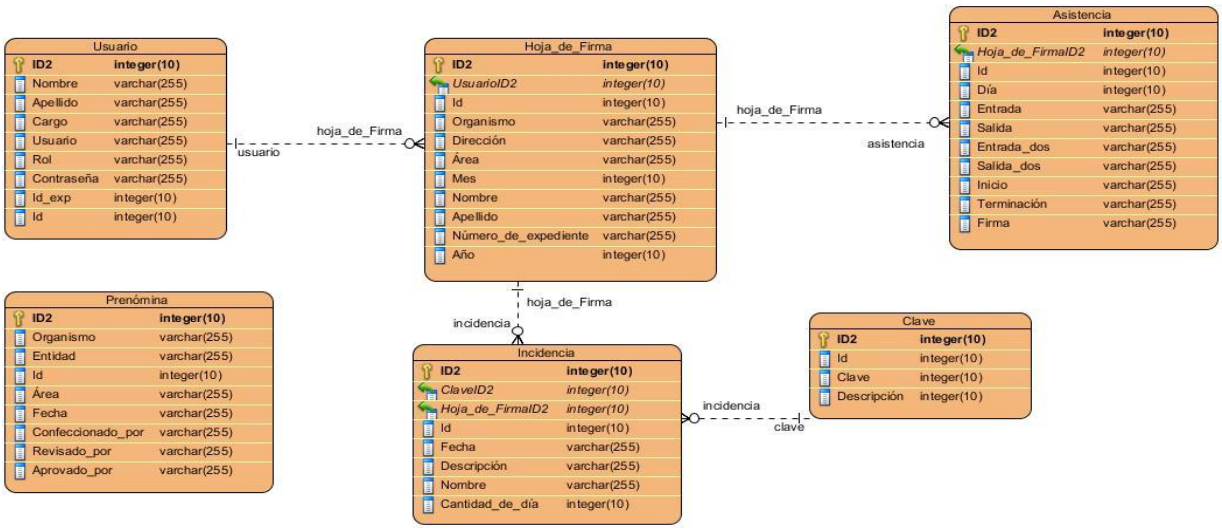


Figura 4:Base de Datos. Fuente:(Elaboración propia)

2. 6 Estándar de codificación

La codificación debe hacerse ateniendo a estándares de codificación ya creados. Programar bajo estándares mantiene el código consistente y facilita su comprensión y escalabilidad. (Torres 2019)

A continuación, se muestran las principales reglas y características del estándar de codificación utilizado en la implementación del sistema:

- Añadir un solo espacio antes y después de los operadores (==, and, or y otros).
- Los comentarios multilíneas en el código JAVA se escriben al iniciar con los caracteres ¨/*¨ y al terminar con ¨*/¨, los comentarios de una sola línea comienzan con los caracteres ¨//¨.
- El código se encuentra tabulado a través del formato que aplica la combinación de teclas ALT+SHIFT+F del NetBeans IDE.
- Los nombres de las clases del modelo, las clases controladoras y los métodos utilizan en sus nombres el estándar Upper Camel Case.
- Todos los atributos de las clases del modelo están encapsulados. Son privados y el acceso a ellos se hace por medio de los métodos getAtributo y/o setAtributo.
- Las variables deben ser explícitas, aunque se pueden usar abreviaturas siempre y cuando no violen este principio.
- Las clases comenzarán con mayúscula y en caso que sean más de una palabra serán escritas continuas y cada letra inicial en mayúscula. Ejemplo: public class ServiceImplTrabajador.java.
- Las variables se escribirán siempre con la primera palabra en minúscula y en caso de que sean necesarias dos o más palabras serán escritas continuas e iniciarán con mayúscula. Ejemplo: int cantDias.
- Los métodos al igual que las clases siempre se escribirán con mayúscula y en caso de usar más de una palabra, estas deben estar unidas y cada inicio de palabra con mayúscula. Ejemplo: void EliminarIncidencia().

Conclusiones parciales

Después de realizado el análisis y diseño de la propuesta de solución y haber generado los diferentes artefactos que dispone la metodología XP, se puede concluir lo siguiente:

- El trabajo en conjunto, del cliente y el desarrollador posibilitaron la identificación de diecinueve requisitos funcionales, descritos en diecinueve historias de usuario.
- Se trazaron requisitos no funcionales de software, hardware, seguridad, portabilidad, interfaz, usabilidad y rendimiento restringiendo el sistema para que cumpla con propiedades o cualidades que debe tener.
- La metodología XP permitió realizar un trabajo organizado y estructurado.
- La generación de las tarjetas CRC permitieron identificar las clases del sistema a desarrollar y la relación entre ellas.
- Un mejor entendimiento del código fue posible gracias a los estándares de codificación.

CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBA

Introducción

La metodología XP plantea que la implementación de un producto software debe realizarse de forma iterativa e incremental, lo que va a permitir que se obtenga un producto funcional. Este entregable debe ser previamente probado para aumentar la visión de los desarrolladores y clientes ante potenciales cambios. En el presente capítulo se define la arquitectura de la propuesta de solución y se analizan los estándares de codificación. Por último, se realizan las pruebas para detectar y corregir posibles errores.

3.1 Fase de Codificación

Tareas de ingeniería

"Se desarrolla en cada iteración el conjunto de historias de trabajador que se han seleccionado para la misma. Cada una de ellas se clasifica en tareas de ingeniería que vendrían a considerarse como las entradas de trabajo para cada equipo de programadores. A nivel documental, como sucede con las historias de usuario, las tareas de ingeniería vendrían a ser como una ficha que contendría el número identificador de la tarea, el identificador de la historia de usuario con la que está relacionada, el nombre de la tarea, el tipo (nuevo desarrollo, corrección, mejora, etc…), la fecha de inicio, su fecha de fin, el equipo responsable y la descripción." (Beck 2019)

La metodología XP plantea dividir las HU en Tareas de Ingeniería con el propósito de facilitar la implementación del sistema a desarrollar. Estas tareas son descritas por los desarrolladores, se realizan en un lenguaje técnico y fácil de comprender para los que la crean. A continuación, se muestra cada iteración donde fueron implementadas las HU con sus respectivas tareas de ingeniería.

Primera iteración

En esta iteración se desarrollan las HU 1,2,3,4 y 5 relacionadas con el documento de requisitos y la redacción de requisitos. Para ello se realizaron las siguientes tareas y el resto se encuentran el Anexo 3:

Tabla 9: Diseño autenticar usuario. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 1	**Número de HU:** 1
Nombre de la tarea: diseño de autenticar usuario	

Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñaron dos input, uno de tipo text y password con los nombres “Usuario” y “Contraseña”, un botón de tipo submit “Aceptar”.	

Tabla 10: Autenticar usuario . Fuente:(Elaboración propia)

Tarea	
Número de tarea: 2	**Número de HU:** 1
Nombre de la tarea: Autenticar usuario	
Tipo de tarea: Desarrollo	**Estimación:** 3 días
Descripción: Se implementó el método authenticateUser () y las clases SecurityConfig, JwtTokenProvider, JwtAuthenticationFilter, JwtAuthenticationEntryPoint, JWTAuthResonseDTO y CustomUserDetailsService para garantizar la autenticación de los usuarios al sistema.	

Tabla 11: Generar trabajadores. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 3	**Número de HU:** 2
Nombre de la tarea: Generar Trabajadores	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Descripción: Se implementó el método GenerarTrabajadoresServicio() y las clases ClienteAssetsUCIWSDL, ClienteUCIWSDLConfig para obtener los trabajadores de los servicios de la UCI.	

Segunda iteración

Tabla 12:Diseño de exportar hoja firma. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 4	**Número de HU:** 6
Nombre de la tarea: Diseño de Exportar Hoja Firma	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un botón con el nombre “Exportar” y una tabla que muestra la Hoja de Firma.	

Tabla 13:Exportar hoja firma. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 5	**Número de HU:** 6
Nombre de la tarea: Exportar Hoja Firma	
Tipo de tarea: Desarrollo	**Estimación:** 1 días

Descripción: Se exporta la tabla a través de la dependencia “jspdf-html2canvas”

Tercera iteración

Tabla 14:Diseño Agregar Incidencia. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 6	**Número de HU:** 13
Nombre de la tarea: Diseño de Agregar Incidencia	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñaron dos botones con el nombre “Salvar” y “Cancelar”, un input de tipo date y uno de tipo textarea, también un select con la lista de todas las claves .	

Tabla 15:Agregar Incidencia. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 7	**Número de HU:** 13
Nombre de la tarea: Agregar Incidencia	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Descripción: Se implementó el método salvarIcidencia () con la función de guardar la incidencia a su hoja de firma correspondiente.	

Cuarta iteración

Tabla 16: Diseño de buscar por código barra. Fuente:(Elaboración propia*)*

Tarea	
Número de tarea: 8	**Número de HU:** 19
Nombre de la tarea: Diseño de Buscar por código barra	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un StreamBarcodeReader de la dependencia "vue-barcode-reader”.	

Tabla 17:Exportar Prenómina. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 9	**Número de HU:** 19
Nombre de la tarea: Exportar Prenómina	
Tipo de tarea: Desarrollo	**Estimación:** 3 días

Descripción: Se implementó los métodos GetHojaFirmaByBarCode () y obtenerPersonaDadoCodigoBarra () y la clase ClienteDatosUCIWSDL.

3.2 Fase de Prueba

La metodología XP divide las pruebas en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñada por los programadores, y pruebas de aceptación o pruebas cliente destinadas a evaluar si se consiguió la funcionalidad requerida diseñadas por el cliente final (Pressman 2020). A continuación, se describen estas pruebas.

3.2.1 Pruebas unitarias

Las pruebas unitarias son una de las piedras angulares de XP. Todos los módulos deben de pasar estas pruebas antes de ser liberados o publicados. Estas pruebas son pruebas de caja blanca, en la que los casos de prueba se basan en la estructura interna. El probador elige entradas para explorar rutas particulares y determina la salida apropiada. El propósito de las pruebas unitarias es examinar los componentes individuales o piezas de métodos / clases para verificar la funcionalidad, asegurando que el comportamiento sea el esperado. El alcance exacto de una “unidad” a menudo se deja a la interpretación, pero una buena regla general es que una unidad contenga la menor cantidad de código que realice una tarea independiente (por ejemplo, un solo método o clase). (Frédéric 2018) (Serafín 2018) (Cíceri 2018)

La herramienta que se encarga de automatizar las pruebas anteriormente descrita y que se utilizó para realizar las misma es conocida como Junit, framework de pruebas para Java, utilizado para escribir y ejecutar pruebas unitarias y de regresión. Es una instancia de la arquitectura JUnit para frameworks de pruebas unitarias. Es de código abierto, inicialmente utilizado para programación extrema. Esta herramienta ejecuta pruebas de manera automática y evita el problema de tener que ejecutar pruebas una por una, de manera manual con posibilidad de errores (Koushik 2018). A continuación, se muestra las pruebas unitarias realizadas a los casos de pruebas listar, actualizar y eliminar prenómina.

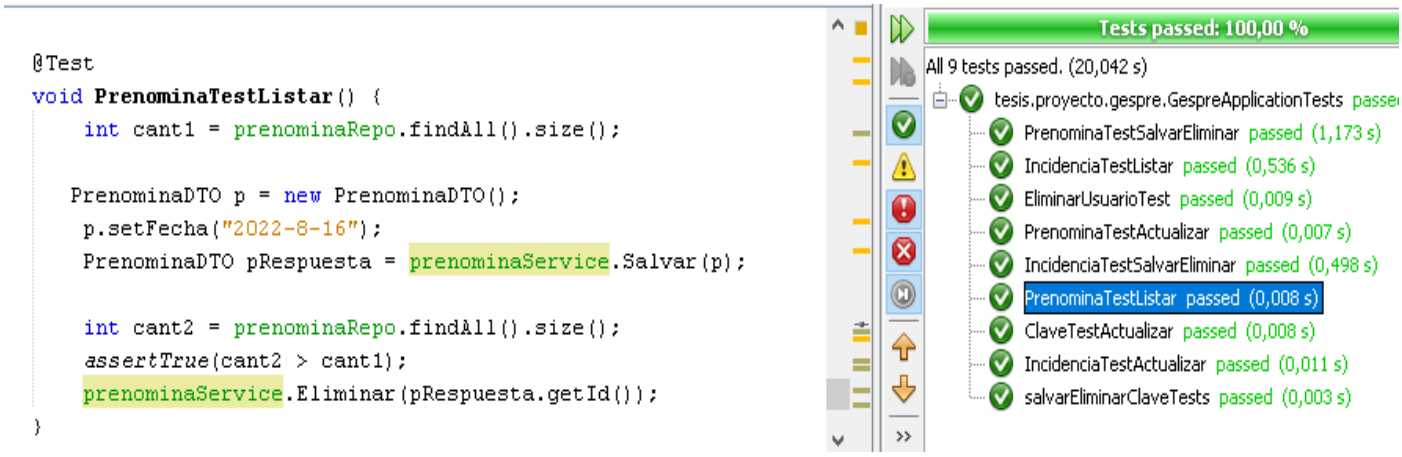


Figura 5:Listar prenómina. Fuente:(Fernández 2018)

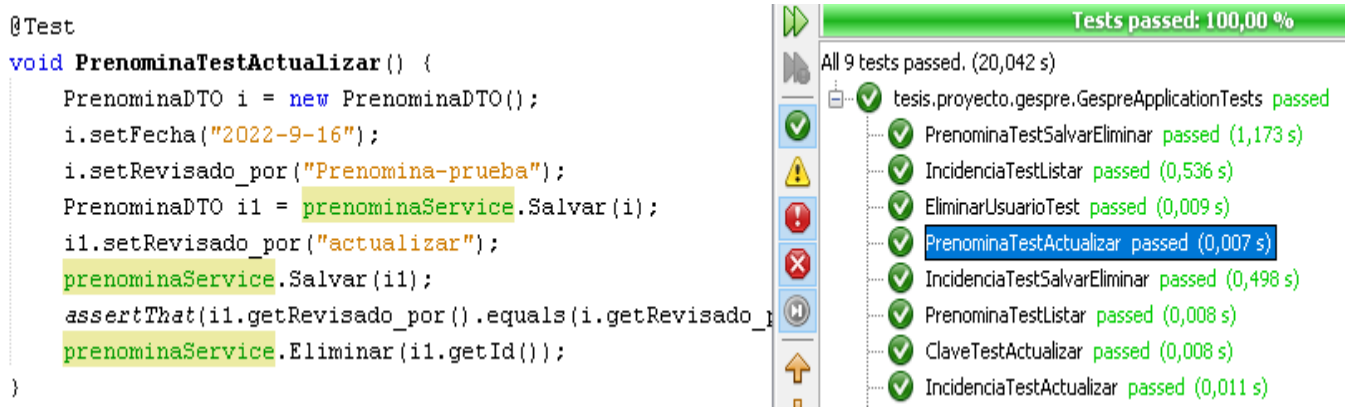


Figura 6:Actualizar prenómina. Fuente:(Fernández 2018)

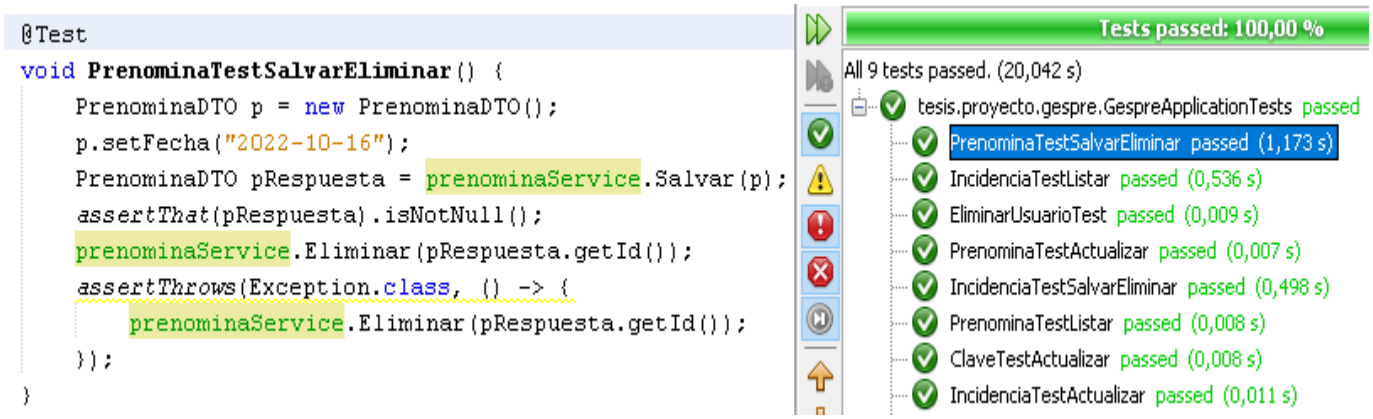


Figura 7:SalvarEliminar prenómina. Fuente:(Elaboración propia)

3.2.2 Pruebas de aceptación

Las pruebas de aceptación o pruebas clientes, se centran en las características y funcionalidades del sistema que son visibles y revisables por parte del cliente. Estas pruebas parten de las historias de usuario y los clientes son los encargados de escribir las pruebas para cada historia de usuario que deba validarse (Castro y Morán 2018) (Ríos y Souto 2019). A continuación, se muestra un caso de prueba de aceptación para la HU 2.

Generar Trabajadores. El resto de los casos de pruebas se encuentran disponibles en el Anexo 4.

Tabla 18:Generar trabajador. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P2HU **Número de HU: 2**
Nombre: Generar Trabajadores
Descripción: Se comprueba que se creen los trabajadores del departamento de tecnología de la Facultad 4.
Condiciones de ejecución: Se tiene que tener conexión a los servicios UCI.
Entrada/Pasos de ejecución: 1. Se da clic en el botón Generar Trabajador.
Resultado esperado: Se crean todos los trabajadores del departamento de tecnología como trabajador en la base de datos de la aplicación y se listan.
Evaluación de la prueba: Satisfactorio

Para realizar estas pruebas de aceptación se realizó la técnica de generación de casos de prueba, como parte del proceso de validación de los requisitos funcionales de la herramienta, se diseñaron un conjunto de CP a partir de cada HU definida. Se realizaron revisiones formales a cada uno de los requisitos por parte del cliente y del equipo de desarrollo, obteniéndose un total de 4 no conformidades de tipo técnicas y de ortografía, las que fueron corregidas satisfactoriamente a tiempo. Con el cliente se llevaron a cabo dos revisiones de los requisitos. En el primer encuentro, 4 de los requisitos fueron modificados con el objetivo de mejorar la interpretación y la ortografía de los mismos. En un segundo encuentro, el cliente quedó satisfecho con el trabajo efectuado por el equipo de desarrollo, generándose de este encuentro un Acta de Aceptación por parte del cliente (ver Anexo 5).

3.2.2 Pruebas funcionales

Con el método de caja negra se examina si la herramienta funciona y cumple con su objetivo para satisfacer al cliente. Como parte de este método se realizó las pruebas funcionales por parte de la cliente para dar solución a las no conformidades encontradas en cada iteración. Partición de equivalencia fue la técnica empleada para realizar esta prueba con el objetivo de detectar errores en las funcionalidades y validar los requisitos de la propuesta de solución

A continuación, se expone el caso de prueba realizado al requisito: Generar

Trabajadores.El resto de los casos de prueba se pueden consultar en el Anexo 6.

Tabla 19:Caso de Prueba. Generar Trabajadores. Fuente (Elaboración propia)

Escenario	Usuario	Contraseña	Respuesta del sistema	Resultado de la prueba
EC2.1 Generar trabajadores exitosamente	[V,I,N/A]	[V,I,N/A]	Muestra la lista de trabajadores creados y el mensaje con la cantidad que se crearon junto con "Trabajadores generados con exito"	Se generaron satisfactoriamente
	N/A	N/A		
EC2.2 Servicios UCI caído	N/A	N/A	El sistema muestra el mensaje "Este servicio UCI está caído o no tienes conexión "	Satisfactoria el sistema alerto al usuario del problema

3.2.2 Resultado de las pruebas

A través de las pruebas realizadas se pude demostrar el correcto funcionamiento del software y las opciones que este facilita, pues las pruebas son una forma de observar si las funciones cumplen con el objetivo por el cual se creó. Además de su codificación eficiente, garantizando de ese modo la calidad de la aplicación. Como se puede observar en la tabla 20 las no conformidades más significativas son las de funcionalidad, las cuales fueron solucionadas para brindarle al cliente un software lo más semejante posible a lo solicitado.

Tabla 20:No conformidades. Fuente:(Elaboración propia)

Elemento	No	No conformidad	Significativa	No Significativa	Tipo no conformidad	Estado NC

Modificar prenómina	1	Al modificar una prenómina se crea una nueva	X		Funcional	Resuelta
Eliminar Hoja de firma	2	Se eliminaba el trabajador correspondiente a la hoja de firma si esta se eliminaba	X		Funcional	Resuelta
Exportar prenómina	3	El sistema utiliza todas las hojas de firma y no las correspondiente al mes	X		Funcional	Resuelta
Eliminar Trabajador	4	El sistema permite eliminar el trabajador, pero no muestra mensaje de confirmación	X		Funcional	Resuelta
Escanear código barra	5	El sistema no permitía escanear el código barra.	X		Funcional	Resuelta
Autenticar usuario	6	El sistema permitía a cualquier usuario la entrada al sitio web	X		Funcional	Resuelta
Registrar clave	7	Al agregar una clave no aparecía la descripción		X	interfaz	Resuelta
Interfaz sugerente	8	Los botones de las paginas no eran sugerentes	X		Funcional	Resuelta

Con el objetivo de comprobar que las funcionalidades del sistema, se realizaron correctamente y responden a las necesidades del cliente, se probó la funcionalidad de este en 4 iteraciones. Obteniendo un total de 8 no conformidades, al finalizar las

iteraciones todas quedaron solucionadas como se muestra en la figura 8.

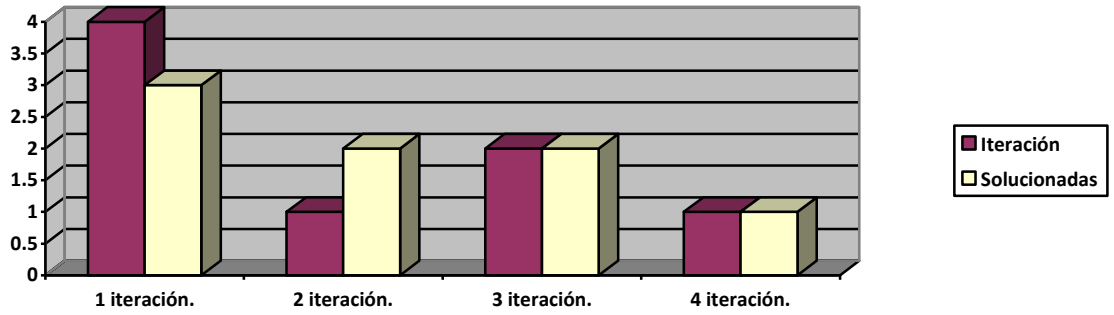


Figura 8:No conformidades. Fuente:(Elaboración propia)

Conclusiones parciales

Tras representar las etapas de implementación y pruebas del software en este capítulo, se plantean las siguientes conclusiones:

- Las pruebas al software permitieron comprobar la calidad del sistema propuesto.
- Las tareas de ingeniería ayudaron al equipo de proyecto a lograr identificar, controlar, rastrear los requisitos y los cambios en cualquier etapa mientras se desarrollaba el proyecto.
- Las pruebas de unidad verificaron el correcto funcionamiento de cada método y arrojaron resultados satisfactorios.
- Las realizaciones de las pruebas de aceptación aprobaron el correcto funcionamiento y el cumplimiento de los requisitos de software definidos para la aplicación.

CONCLUSIONES GENERALES

Al terminar la presente investigación se arribó a las siguientes conclusiones generales:

- El análisis realizado del estado del arte, fundamentó la necesidad de llevar a cabo el desarrollo de un sistema web que contribuye al proceso de gestión de prenóminas del Departamento de Tecnología de la Facultad 4.
- El uso de la metodología XP como guía del proceso de desarrollo de software, así como las herramientas y tecnologías tales como el framework Vue.js en su versión 3, el lenguaje de programación Java 8 y el servidor web Apache 2.0, facilitó el correcto desarrollo de la propuesta solución.
- Los requisitos del sistema, la arquitectura modelo vista controlador y el uso de los patrones de diseño, permitieron obtener una aplicación web optima que cumple con las expectativas del cliente.
- La realización de las pruebas unitarias, funcionales y de aceptación demostró que la implementación satisface los requisitos propuestos por el cliente, lo cual da cumplimiento al objetivo trazado en la presente investigación.

RECOMENDACIONES

Luego de concluir con la presente investigación se arribó a las siguientes recomendaciones:

- Realizar el despliegue del sistema a todas las áreas de la Facultad 4 y posteriormente a otras áreas de la UCI.
- Implementar las asistencias de los trabajadores mediante el uso del dispositivo electrónico lector de código de barras o escáner.

REFERENCIAS BIBLIOGRÁFICAS

1. Aguirre, Santiago. 2022. «PHP Avanzado».
2. Bai, Ying. 2022. *SQL Server Database Programming with Java: Concepts, Designs and Implementations*.
3. Beck, Addison Wesley. 2019. *Extreme Programming Explained: Embrace Change*.
4. ByKibet, John. 2022. «Best Books To Learn PostgreSQL Database».
5. Cano, Isabel María. 2022. «Fundamentos de ingeniería de los requisitos».
6. Castro, Martha Irene Romero, y Grace Liliana Figueroa Morán. 2018. «introducción a la seguridad informática y el análisis de vulnerabilidades».
7. Cíceri, Marcelo. 2018. «Introducción a Laravel: Aplicaciones robustas y a gran escala».
8. Clark, Wesley. 2020. «Una Guía definitiva para principiantes para el dominio de la metodología de gestión de proyectos Scrum».
9. Debrauwer, Laurent. 2018. *Patrones de diseño en Java. Los 23 modelos de diseñ*.
10. Fernández, Yenisleidy Romero. 2018. «Patrón Modelo-Vista-Controlador». http://revistatelematica.cujae.edu.cu/index.php/tele.
11. Frédéric, Déléchamp. 2018. «Java y Eclipse: desarrolle una aplicación con Java y Eclipse».
12. Galez, Ramon. 2018. «JavaServer Faces».
13. García, Ramón Egido. 2018. «Curso Avanzado de Programación en Java EE: Struts, JSF, Ajax, EJB, JPA», 2018.
14. Gauchat, Juan Diego. 2018. «El gran libro de HTML5, CSS3 y JAVASCRIPT».
15. Gómez, Mario Rubiales. 2021. «Curso de desarrollo Web. HTML, CSS y JavaScript».
16. Herredo, Carmen de Pablos, José Joaquín López Hermoso Agius, y Santiago Martín-Romo Romero. 2019. *Organización y transformación de los sistemas de información*.
17. Hinkula, Juha. 2022. «Full Stack Development with Spring Boot».
18. Ivar, Jacobson, Grady Booch, y James Rumbaugh. 2020. «El Proceso Unificado de Desarrollo de Software». ISBN 0-201-57169-2.
19. Kabir, Mohammed J. 2020. «Servidor Apache 2».
20. Koushik, Das. 2018. *Create an Enterprise-Level Test Automation Framework with*

Appium.

21. Lahoucine, Souabi. 2021. *Registro De Asistencia Del Empleado: Control de Asistencia de Personal*.
22. Lopez, Raul. 2019. «Java Application developer's guide».
23. Martínez, Eugenia Pérez. 2021. «Hibernate. Persistencia de objetos en JEE».
24. Nixon. 2019. «Aprender PHP, MySQL y JavaScript».
25. Perez, Jesus R. 2021. «diseño y administración de bases de datos en MySQL».
26. Pressman, Roger S. 2020. «Ingeniería del software : un enfoque práctico. McGrawHill.»
27. Quintero, Sergio Delgado. 2022. «Aprende Python».
28. RASOBARES, Alejandro HERNANDEZ T. 2018. «Los sistemas de información: evolución y desarrollo».
29. Ríos, Jimmy Rolando Molina, y María de las Nieves Pedreira Souto. 2019. «"SWIRL", metodología para el diseño y desarrollo de aplicaciones web».
30. Serafín, Miguel Muñoz. 2018. «Programador Jr. de Aplicaciones ASP. NET MVC: Manual de Estudiante».
31. Shvets, Alexander. 2021. «Sumérgete en los patrones de diseño».
32. Tarapués, Escobar, Paola Ingrid, y Humberto Jorge. 2022. *Herramientas para la gestión de las relaciones con los clientes*.
33. Torres, Alejandro Daniel. 2019. *Manual de estandares de codificación y buenas prácticas en c#, javascript, transact sql y arquitectura*.

ANEXOS:

Anexo 1: Historia de Usuario

Tabla 21:Autenticar Usuario. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 01	**Nombre:** Autenticar Usuario
Usuario: Administrador	
Prioridad del Negocio: Alta.	**Iteración Asignada:** 1
Punto de Estimación: 4	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: Para entrar al sistema deberá autenticarse con usuario y contraseña que será los dos campos a llenar en la interfaz del login y presionar el botón aceptar.	
Observaciones: Solo tiene permiso el ADMIN, si presiona el botón aceptar y el usuario o contraseña son incorrectos el sistema muestra el mensaje “Autorizando a:”	

Tabla 22:Eliminar trabajador. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 03	**Nombre:** Eliminar trabajador.
Usuario: Administrador	
Prioridad del Negocio: Media.	**Iteración Asignada:** 1
Punto de Estimación: 2	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar trabajador se muestra un botón eliminar que al hacerle clic aparece una notificación preguntando si desea eliminarlo o no, en caso afirmativo se elimina del listado y en caso negativo se regresa a la interfaz gestionar trabajadors	

Observaciones: Si eliminas un trabajador se elimina con él todo lo referente a sus datos.

Tabla 23:Generar hoja de firma. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 04	**Nombre:** Generar Hoja Firma
Usuario: Administrador	
Prioridad del Negocio: Alta.	**Iteración Asignada:** 1
Punto de Estimación: 3	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar hojas de firmas se muestra un botón "Generar Hoja Firmas" que al presionarlo se va a otra interfaz donde aparece la tabla de hoja de firma a crear de forma automática con los campos predefinido (organismo, área), los campos automáticos (nombre y apellidos, no. Expediente, mes, año) y tres botones generar, cancelar, predefinido guardar.	
Observaciones: Si presiona el botón generar y el trabajador ya tiene una hoja de firma de ese mes y año, no lo sobrescribe. Si presiona cancelar regresa a la interfaz gestionar hojas de firmas.	

Tabla 24:Eliminar hoja de firma. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 05	**Nombre:** Eliminar Hoja Firma.
Usuario: Administrador	
Prioridad del Negocio: Media.	**Iteración Asignada:** 1
Punto de Estimación: 2	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar hoja de firmas se muestra un botón eliminar que al hacerle clic aparece una notificación preguntando si desea eliminarlo o no, en caso afirmativo se elimina del listado y en caso negativo se regresa a la interfaz gestionar hoja de firmas.	

Observaciones: Si eliminas una hoja de firma se elimina todo lo referente a sus datos.

Tabla 25:Exportar hoja de firma. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 06	**Nombre:** Exportar Hoja Firma.
Usuario: Administrador	
Prioridad del Negocio: Alta.	**Iteración Asignada:** 2
Punto de Estimación: 2	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: Se presiona el botón ”Exportar” el sistema muestra la interfaz de la tabla Hoja Firma con todas sus asistencias. También dos botones, exportar y cancelar.	
Observaciones: Si hace clic en el botón exportar, una vez hecho esto se guarda el archivo hojaFirma.pdf y si presiona cancelar se redirecciona hacia la interfaz de ”Gestionar Hoja de Firmas”.	

Tabla 26.Guardar Hoja Firma Predefinida. Fuente: (Elaboración propia)

HISTORIA DE USUARIO	
No: 07	**Nombre:** Guardar Hoja Firma Predefinida.
Usuario: Administrador	
Prioridad del Negocio: Baja.	**Iteración Asignada:** 4
Punto de Estimación: 2	**Riesgo en el desarrollo:** Baja.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar hojas de firmas se muestra un botón “Generar Hoja Firmas” que al presionarlo se va a otra interfaz donde aparece la tabla de hoja de firma a crear de forma automática con los campos predefinido (organismo, área), los campos automáticos (nombre y apellidos, no. Expediente, mes, año) y tres botones generar, cancelar, predefinido guardar.	
Observaciones: Si presiona el botón Predefinido Guardar se guardan los datos de organismo y área, para que aparezcan de forma automática.	

Tabla 27:Eliminar prenómina. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 10	**Nombre:** Eliminar Prenómina.
Usuario: Administrador	
Prioridad del Negocio: Alta.	**Iteración Asignada:** 2
Punto de Estimación: 2	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar prenómina se muestra un botón eliminar que al hacerle clic aparece una notificación preguntando si desea eliminarlo o no, en caso afirmativo se elimina del listado y en caso negativo se regresa a la interfaz gestionar prenómina.	
Observaciones:	

HISTORIA DE USUARIO	
No: 09	**Nombre:** Modificar prenómina.
Usuario: Administrador	
Prioridad del Negocio: Media.	**Iteración Asignada:** 2
Punto de Estimación: 2	**Riesgo en el desarrollo:** Alto.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: Se presiona el botón modificar, el sistema muestra la intefaz con una tabla prenómina con los campos a modificar (fecha, revisado por, aprobado por, organismo, entidad, área, confec. por) y el botón salvar y cancelar.	
Observaciones: Si presiona el botón salvar y ya existe una prenómina del mismo mes y año el sistema muestra el mensaje “Ya existe una prenómina de este mes” o de lo contrario se guarda y se muestra el mensaje “Guardado”, si presiona cancelar el sistema te lleva para la página ”Gestionar Prenómina”, si presiona predefinido guardar, salva los datos de los campos predefinidos.	

Tabla 28:Modificar prenómina. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 11	**Nombre:** Guardar Prenómina Predefinida
Usuario: Administrador	
Prioridad del Negocio: bajo.	**Iteración Asignada:** 4
Punto de Estimación: 2	**Riesgo en el desarrollo:** Baja.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: Se presiona el botón “Agregar Prenómina”, el sistema muestra la interfaz con una tabla prenómina con los campos a llenar (fecha, revisado por, aprobado por), también los campos predefinidos (organismo, entidad, área, confec. por) y el botón salvar, cancelar y predefinido guardar.	
Observaciones: Si presiona el botón Predefinido Guardar se guardan los datos de organismo, entidad, área y confec. Por, para que aparezcan de forma automática.	

HISTORIA DE USUARIO	
No: 12	**Nombre:** Exportar prenómina.
Usuario: Administrador	
Prioridad del Negocio: Alta.	**Iteración Asignada:** 2
Punto de Estimación: 2	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: Se presiona el botón “Exportar” el sistema muestra la interfaz de la tabla prenómina confeccionada con todas las hojas de firmas correspondientes a la fecha, asi como las incidencias de cada trabajador con su clave. También dos botones, exportar y cancelar.	
Observaciones: Si hace clic en el botón exportar, una vez hecho esto se guarda el archivo prenomina.pdf y si presiona cancelar se redirecciona hacia la interfaz de “Gestionar Prenómina”.	

Tabla 29:Exporatar prenómina. Fuente:(Elaboración propia)

HISTORIA DE USUARIO

No: 13	**Nombre:** Agregar Incidencia.
Usuario: Administrador	
Prioridad del Negocio: Alta.	**Iteración Asignada:** 3
Punto de Estimación: 3	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar hoja de firmas se muestra un botón “Incidencia” que al hacerle clic se pasa a otra interfaz donde se introducen los datos (fecha de la incidencia, descripción, cantidad de días, clave) y dos botones, salvar y cancelar	
Observaciones: Si presiona el botón salvar se guarda la incidencia a su prenómina correspondiente y si presiona cancelar se re direcciona hacia la interfaz “Gestionar Hoja Firma”.	

Tabla 30:Agregar incidencia. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 14	**Nombre:** Modificar Incidencia.
Usuario: Administrador	
Prioridad del Negocio: Alta.	**Iteración Asignada:** 3
Punto de Estimación: 2	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar Incidencia se muestra un botón modificar que al hacerle clic se pasa a otra interfaz donde se introducen los datos a modificar (fecha de la incidencia, descripción, cantidad de días, clave) y dos botones, salvar y cancelar	
Observaciones: Si presiona el botón salvar se actualiza la incidencia y si presiona cancelar se redirecciona hacia la interfaz gestionar Incidencia.	

Tabla 31:Modificar incidencia. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 15	**Nombre:** Eliminar Incidencia.
Usuario: Administrador	

Prioridad del Negocio: Alta.	**Iteración Asignada:** 3
Punto de Estimación: 2	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar Incidencia se muestra un botón eliminar que al hacerle clic aparece una notificación preguntando si desea eliminarlo o no, en caso afirmativo se elimina del listado y en caso negativo se regresa a la interfaz gestionar Incidencia.	
Observaciones:	

Tabla 32:Eliminar incidencia. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 16	**Nombre:** Agregar Clave.
Usuario: Administrador	
Prioridad del Negocio: Alta.	**Iteración Asignada:** 3
Punto de Estimación: 3	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar clave se muestra un botón “Agregar Clave” que al hacerle clic se pasa a otra interfaz donde se introducen los datos (Clave, descripción) y dos botones, salvar y cancelar	
Observaciones:	

Tabla 33:Agregar Clave. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 17	**Nombre:** Modificar Clave.
Usuario: Administrador	
Prioridad del Negocio: Media.	**Iteración Asignada:** 4
Punto de Estimación: 2	**Riesgo en el desarrollo:** Medio.

Programador responsable: Alejandro David Monagas Torrecilla
Descripción: En la interfaz gestionar clave se muestra un botón modificar que al hacerle clic se pasa a otra interfaz con los campos a modificar (Clave, descripción) y dos botones, salvar y cancelar
Observaciones:

Tabla 34:Modificar clave. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 18	**Nombre:** Eliminar Clave.
Usuario: Administrador	
Prioridad del Negocio: Media.	**Iteración Asignada:** 4
Punto de Estimación: 2	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	
Descripción: En la interfaz gestionar clave se muestra un botón eliminar que al hacerle clic aparece una notificación preguntando si desea eliminarlo o no, en caso afirmativo se elimina del listado y en caso negativo se regresa a la interfaz gestionar clave.	
Observaciones:	

Tabla 35:Eliminar clave. Fuente:(Elaboración propia)

HISTORIA DE USUARIO	
No: 19	**Nombre:** Buscar por código barra.
Usuario: Trabajador	
Prioridad del Negocio: Alta.	**Iteración Asignada:** 4
Punto de Estimación: 4	**Riesgo en el desarrollo:** Medio.
Programador responsable: Alejandro David Monagas Torrecilla	

Descripción: En la interfaz firmar se muestra un escaner donde el trabajador que ponga su código barra del solapín le saldra la asistencias de su hoja de firma correspondiente a firmar con los datos (mañana entrada, mañada salida, tarde entrada, tarde salida, terminación, inicio, firma) .
Observaciones: Si no tiene una hoja de firma o no esta actualizada le saldrá este mensaje “Este trabajador no posee una hoja de firma actualizada”

Tabla 36:Buscar por código barra. Fuente:(Elaboración propia)

Anexo 2: Tarjetas CRC

TARJETA CRC	
Clase: Incidencia.java	
Responsabilidades:	**Colaboraciones**
Contiene todas las variables correspondientes a a las icidencias y los metodos set y get	HojaFirma.java Clave.java

Tabla 37:Incidencia.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: Clave.java	
Responsabilidades:	**Colaboraciones**
Contiene todas las variables correspondientes a a las claves y los métodos set y get	

Tabla 38:Clave.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: Trabajador.java	
Responsabilidades:	**Colaboraciones**
Contiene todas las variables correspondientes a los trabajadores y los métodos set y get.	HojaFirma.java

Tabla 39:Trabajador.java. Fuente:(Elaboración propia)

TARJETA CRC

Clase: Asistencia.java	
Responsabilidades:	**Colaboraciones**
Contiene todas las variables correspondientes a las asistencias y los métodos set y get.	HojaFirma.java

Tabla 40Asitencia.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: HojaFirma.java	
Responsabilidades:	**Colaboraciones**
Contiene todas las variables correspondientes a las hojas de firmas y los métodos set y get	Trabajador.java Asistencia.java Incidencia.java

Tabla 41:HojaFirma.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: TrabajadorController.java	
Responsabilidades:	**Colaboraciones**
GenerarTrabajadorsServicio() ObtenrTrabajadors() EliminarTrabajadorPorId()	ITrabajadorService.java TrabajadorDTO.java

Tabla 42:TrabajadorController.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: PreominaController.java	
Responsabilidades:	**Colaboraciones**
obtenrPrenomina() EliminarPrenomina() ActualizarPrenomina()	IPrenominaService.java PrenominaDTO.java

Tabla 43:PrenominaController.java. Fuente:(Elaboración propia)

TARJETA CRC

Clase: IncidenciaController.java	
Responsabilidades:	**Colaboraciones**
ObtenrIncidencia() ObtenerIncidenciaPorId() SalvarIcidencia() ActualizarIncidencia() EliminarIncidenciaPorId()	IIncidenciaService.java IncidenciaDTO.java

Tabla 44:IncidenciaController.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: ClaveController.java	
Responsabilidades:	**Colaboraciones**
ObtenrClave() ObtenerClavePorId() SalvarClave() ActualizarClave() EliminarClavePorId()	IClaveService.java ClaveDTO.java

Tabla 45:ClaveController.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: ServiceImplClave.java	
Responsabilidades:	**Colaboraciones**
mapearDTO() mapearEntidad() Salvar() FindAll() ObtenerPorId() Actualizar() Eliminar()	ModelMapper.java IClaveRepo.java IIncidenciaRepo.java IncidenciaDTO.java Incidencia.java ClaveDTO.java Clave.java

Tabla 46:ServiceImplClave.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: ServiceImplIncidencia.java	
Responsabilidades:	**Colaboraciones**

mapearDTO() mapearEntidad() Salvar() FindAll() ObtenerPorId() Actualizar() Eliminar()	ModelMapper.java IHojaFirmaRepo.java IIncidenciaRepo.java IClaveRepo.java IncidenciaDTO.java Incidencia.java ClaveDTO.java Clave.java HojaFirmaDTO.java HojaFirma.java

Tabla 47:ServiceImplIncidencia.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: ServiceImplPrenomina.java	
Responsabilidades:	**Colaboraciones**
mapearDTO() mapearEntidad() Salvar() FindAll() ObtenerPorId() Actualizar() Eliminar()	ModelMapper.java IHojaFirmaRepo.java IPrenominaRepo.java HojaFirmaDTO.java HojaFirma.java PrenominaDTO.java Prenomina.java

Tabla 48:ServiceImplPrenomina.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: ServiceImplHojaFirma.java	
Responsabilidades:	**Colaboraciones**
mapearDTO() mapearEntidad() Salvar() FindAll() ListarPorTrabajador() ObtenerPorId() actualizarHojaFirmaPorIDTrabajador() GetHojaFirmaByUser() getHojaFirmaByBarCode() Eliminar()	ModelMapper.java IHojaFirmaRepo.java IPrenominaRepo.java ITrabajadorRepo.java IAsistenciaRepo.java ClienteDatosUCIWSDL.java HojaFirmaDTO.java HojaFirma.java PrenominaDTO.java Prenomina.java TrabajadorDTO.java Trabajador.java AsistenciaDTO.java Asistencia.java

Tabla 49:ServiceImplHojaFirma.java. Fuente:(Elaboración propia)

TARJETA CRC	
Clase: ServiceImplAsistencia.java	
Responsabilidades:	**Colaboraciones**
mapearDTO() mapearEntidad() SalvarPorId() findByHojaFirma() findByHojaFirmaYasistencia()	ModelMapper.java IHojaFirmaRepo.java ITrabajadorRepo.java IAsistenciaRepo.java ClienteDatosUCIWSDL.java HojaFirmaDTO.java HojaFirma.java AsistenciaDTO.java Asistencia.java TrabajadorDTO.java Trabajador.java

Tabla 50:ServiceImplAsistencia.java. Fuente:(Elaboración propia)

Anexo 3: Tareas de Ingeniería

Tarea	
Número de tarea: 10	**Número de HU:** 2
Nombre de la tarea: Diseño de Generar Trabajador	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un botón con el nombre “Generar Trabajadores”	

Tabla 51.:Diseño de generar trabajadores. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 11	**Número de HU:** 4
Nombre de la tarea: Diseño de Generar Hoja Firma	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un botón con el nombre “Generar Hoja Firmas”	

Tabla 52:Diseño de generar hoja firma. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 12	**Número de HU:** 4
Nombre de la tarea: Generar Hoja Firma	
Tipo de tarea: Desarrollo	**Estimación:** 2 días

Descripción: Se implementó el método GenerarHojasFirmas () encargado de crearle a todos los trabajadores del sistema su hoja de firma correspondiente del mes.

Tabla 53:Generar hoja firma. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 13	**Número de HU:** 3
Nombre de la tarea: Diseño de Eliminar Trabajador	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un botón con un icono de un cesto en rojo.	

Tabla 54:Diseño de eliminar trabajador. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 14	**Número de HU:** 3
Nombre de la tarea: Eliminar Trabajador	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Descripción: Se implementó el método EliminarTrabajadorPorId () encargado de eliminar un trabajador por su id.	

Tabla 55:Eliminar trabajador. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 15	**Número de HU:** 5
Nombre de la tarea: Diseño de Eliminar Hoja Firma	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un botón con un icono de un cesto en rojo.	

Tabla 56:Diseño eliminar hoja firma. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 16	**Número de HU:** 5
Nombre de la tarea: Eliminar Hoja Firma	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Descripción: Se implementó el método EliminaHojaFirmaPorId () encargado de eliminar una hoja de firma por su id.	

Tabla 57:Eliminar hoja firma. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 17	**Número de HU:** 7
1. **Nombre de la tarea:** Diseño de Guardar Hoja Firma Predefinida	

Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un botón con el nombre “Predefinido Guardar”	

Tabla 58.Diseño de Guardar Hoja Firma Predefinida. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 18	**Número de HU:** 7
Nombre de la tarea: Guardar Hoja Firma Predefinida	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Descripción: Se implementó el método PredefindoGuardarHojasFirmas() encargado guardar una Hoja de Firma.	

Tabla 59.Guardar Hoja Firma Predefinida. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 19	**Número de HU:** 8
Nombre de la tarea: Diseño de Agregar Prenómina	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó una tabla de Prenómina con siete input de tipo text y uno de tipo date, tres botones con nombres “Salvar”, “ Cancelar”, ”Predefinido Guardar ”	

Tabla 60:Diseño agregar prenómina. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 20	**Número de HU:** 8
Nombre de la tarea: Agregar Prenómina	
Tipo de tarea: Desarrollo	**Estimación:** 3 días
Descripción: Se implementó el método salvarPrenomina () encargado de guardar una Prenómina.	

Tabla 61:Agregar prenómina. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 21	**Número de HU:** 9
Nombre de la tarea: Diseño de Modificar Prenómina	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se obtiene los datos de la prenómina a través de método ObtenerPrenominaPorId() y se generan los datos necesarios para su modificación.	

Tabla 62.Diseño de Modificar Prenómina. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 22	**Número de HU:** 9

Nombre de la tarea: Modificar Prenómina	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Descripción: Se implementó el método ActualizarPrenomina() con la función de actualizar los datos de la prenómina correspondiente a su id.	

Tabla 63.Modificar Prenómina. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 23	**Número de HU:** 10
Nombre de la tarea: Diseño de Eliminar Prenómina	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un botón con un icono de un cesto en rojo.	

Tabla 64.Diseño de Eliminar Prenómina. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 24	**Número de HU:** 10
Nombre de la tarea: Eliminar Prenómina	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Descripción: Se implementó el método EliminarPrenominaPorId () encargado de eliminar una prenómina por su id.	

Tabla 65.Eliminar Prenómina. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 25	**Número de HU:** 11
2. **Nombre de la tarea:** Diseño de Guardar Prenómina Predefinida	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó una tabla de Prenómina con siete input de tipo text y uno de tipo date, tres botones con nombres “Salvar”, “ Cancelar”, ”Predefinido Guardar ”	

Tabla 66.Diseño de Guardar Prenómina Predefinida. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 26	**Número de HU:** 11
Nombre de la tarea: Guardar Prenómina Predefinida	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Descripción: Se implementó el método salvarPrenominaPredefinida() encargado de guardar una Prenómina.	

Tabla 67.Guardar Prenómina Predefinida. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 27	**Número de HU:** 12
Nombre de la tarea: Diseño de Exportar Prenómina	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un botón con el nombre “Exportar” y una tabla que muestra la Hoja de Firma.	

Tabla 68:Diseño exportar prenómina. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 28	**Número de HU:** 12
Nombre de la tarea: Exportar Prenómina	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Descripción: Se exporta la tabla a través de la dependencia “jspdf-html2canvas”	

Tabla 69:Exportar prenómina. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 29	**Número de HU:** 14
Nombre de la tarea: Diseño de Modificar Incidencia	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se obtiene los datos de la incidencia a través de método ObtenerIncidenciaPorId() y se generan los datos necesarios para su modificación.	

Tabla 70:Diseño modificar incidencia. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 30	**Número de HU:** 14
Nombre de la tarea: Modificar Incidencia	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Descripción: Se implementó el método ActualizarIncidencia() con la función de actualizar los datos de la incidencia correspondiente a su id.	

Tabla 71:Agregar incidencia. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 31	**Número de HU:** 15
Nombre de la tarea: Diseño de Eliminar Incidencia	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un botón con un icono de un cesto en rojo.	

Tabla 72.Diseño de Eliminar Incidencia. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 32	**Número de HU:** 15
Nombre de la tarea: Eliminar Incidencia	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Descripción: Se implementó el método EliminarIncidenciaPorId () encargado de eliminar una incidencia por su id.	

Tabla 73.Eliminar Incidencia. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 33	**Número de HU:** 16
Nombre de la tarea: Diseño de Agregar Clave	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñaron dos botones con el nombre “Salvar” y “Cancelar”, un input del tipo number y otro del tipo text.	

Tabla 74.Diseño de Agregar Clave. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 34	**Número de HU:** 16
Nombre de la tarea: Agregar Clave	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Descripción: Se implementó el método salvarClave() con la función de guardar una clave.	

Tabla 75.Agregar Clave. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 35	**Número de HU:** 17
Nombre de la tarea: Diseño de Modificar Clave	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se obtiene los datos de la clave a través de método ObtenerClavePorId() y se generan los datos necesarios para su modificación.	

Tabla 76:Diseño modificar clave. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 36	**Número de HU:** 17
Nombre de la tarea: Modificar Clave	
Tipo de tarea: Desarrollo	**Estimación:** 1 días

Descripción: Se implementó el método ActualizarClave() con la función de actualizar los datos de la incidencia correspondiente a su id.

Tabla 77.Modificar Clave. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 37	**Número de HU:** 18
Nombre de la tarea: Diseño de Eliminar Clave	
Tipo de tarea: Diseño	**Estimación:** 1 días
Descripción: Se diseñó un botón con un icono de un cesto en rojo.	

Tabla 78.Diseño de Eliminar Clave. Fuente:(Elaboración propia)

Tarea	
Número de tarea: 38	**Número de HU:** 18
Nombre de la tarea: Eliminar Clave	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Descripción: Se implementó el método EliminarClavePorId () encargado de eliminar una Clave por su id.	

Tabla 79.Eliminar Clave. Fuente:(Elaboración propia)

Anexo 4: Pruebas de Aceptación

Tabla 80:Autenticar usuario. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P1HU **Número de HU: 1**
Nombre: Autenticar Usuario
Descripción: Validación de entrada de los datos de los usuarios
Condiciones de ejecución: Se debe introducir un nombre de usuario y una contraseña.
Entrada/Pasos de ejecución: 1. Se escribe nombre de usuario y contraseña y luego da clic en el botón Aceptar.
Resultado esperado: Si el usuario tiene acceso para entrar a la aplicación e inserta sus datos correctamente entrará sin problemas al Sistema. Se emite un mensaje de error en caso de que: Se inserte los datos de un usuario no válido para el Sistema o incorrectos.

Se dé clic en el botón Entrar sin insertar nada en los campos de texto.
Evaluación de la prueba: Satisfactorio

Caso de Prueba de Aceptación
Código: P3HU **Número de HU: 3**
Nombre: Eliminar trabajador.
Descripción: Se verifica si se puede eliminar un trabajador.
Condiciones de ejecución: Se tiene que tener un trabajador creado para poder eliminarlo.
Entrada/Pasos de ejecución: 1. Se da clic en el botón del icono eliminar. 2. Se da clic en el botón OK.
Resultado esperado: Se elimina un trabajador .
Evaluación de la prueba: Satisfactorio

Tabla 81:Eliminar trabajador. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P4HU **Número de HU: 4**
Nombre: Generar Hoja Firma.
Descripción: Se verifica que se creen todas las hojas de firmas del mes actual a todos los trabajadores del sistema.
Condiciones de ejecución: Se tiene que tener todos los trabajadores para crear su hoja de firma.
Entrada/Pasos de ejecución: 1. Se da clic en el botón Generar Hoja Firma. 2. Se completan los campos requeridos. 3. Se da clic en el botón Generar
Resultado esperado: Se generaron todas las hojas de firmas por cada trabajador.
Evaluación de la prueba: Satisfactorio

Tabla 82:Generar hoja firma. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P5HU **Número de HU: 5**
Nombre: Eliminar Hoja Firma.
Descripción: Se verifica si se puede eliminar una hoja de firma.
Condiciones de ejecución: Se tiene que tener una hoja de firma creado para poder

eliminarla.
Entrada/Pasos de ejecución: 1. Se da clic en el botón del icono eliminar. 2. Se da clic en el botón OK.
Resultado esperado: Se elimina la hoja de firma .
Evaluación de la prueba: Satisfactorio

Tabla 83:Eliminar hoja firma. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P6HU **Número de HU: 6**
Nombre: Exportar Hoja Firma.
Descripción: Se verifica si se puede exportar una hoja de firma.
Condiciones de ejecución: Se tiene que tener una hoja de firma creado para poder exportarla.
Entrada/Pasos de ejecución: 1. Se da clic en el botón de Exportar de la tabla. 2. Se da clic en el botón Exportar.
Resultado esperado: Se exporta la hoja de firma.
Evaluación de la prueba: Satisfactorio

Tabla 84:Exportar hoja firma. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P7HU **Número de HU: 7**
Nombre: Guardar Hoja Firma Predefinida.
Descripción: Se verifica que se cree una hoja de firma con los datos predefinidos.
Condiciones de ejecución: Se deben llenar los campos organismo y área en caso de no tenerlos
Entrada/Pasos de ejecución: 1. Se da clic en el botón Generar Hoja Firma. 2. Se completan los campos requeridos. 3. Se da clic en el botón Predefinido Guardar
Resultado esperado: Se guardan los datos y se quedan como predefindos.
Evaluación de la prueba: Satisfactorio

Tabla 85.Guardar Hoja Firma Predefinida. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P10HU **Número de HU: 10**
Nombre: Eliminar Prenómina.

Descripción: Se verifica si se puede eliminar una Prenómina.
Condiciones de ejecución: Se tiene que tener una Prenómina creada para poder eliminarla.
Entrada/Pasos de ejecución: 1. Se da clic en el botón del icono eliminar. 2. Se da clic en el botón OK.
Resultado esperado: Se elimina la Prenómina.
Evaluación de la prueba: Satisfactorio

Tabla 86:Eliminar prenómina. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P8HU **Número de HU: 8**
Nombre: Agregar prenómina.
Descripción: Se verifica que se cree una prenómina.
Condiciones de ejecución: Se debe llenar los campos revisado por, aprobado por y los predefinidos en caso de no tenerlo.
Entrada/Pasos de ejecución: 1. Se completan los campos y se da clic en el botón Salvar.
Resultado esperado: Se crea la prenómina. Se eminete un mensaje de error en caso de que exista una prenómina con el mismo mes y año.
Evaluación de la prueba: Satisfactorio

Tabla 87:Agregar prenómina. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P9HU **Número de HU: 9**
Nombre: Modificar prenómina.
Descripción: Se verifica que se modifique una prenómina.
Condiciones de ejecución: Se debe tener una prenómina creada y modificar por lo menos un campo.
Entrada/Pasos de ejecución: 1. Se modifican los campos y se da clic en el botón Salvar.
Resultado esperado: Se modifica la prenómina.
Evaluación de la prueba: Satisfactorio

Tabla 88:Modificar prenómina. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P11HU **Número de HU: 11**
Nombre: Guardar Prenomina Predefinida.

Descripción: Se verifica que se cree una prenómina con los datos predefinidos.
Condiciones de ejecución: Se deben llenar los campos organismo, entidad, área, confec. por en caso de no tenerlos
Entrada/Pasos de ejecución: 1. Se da clic en el botón Generar Hoja Firma. 2. Se completan los campos requeridos. 3. Se da clic en el botón Predefinido Guardar
Resultado esperado: Se guardan los datos y se quedan como predefindos.
Evaluación de la prueba: Satisfactorio

Tabla 89.Guardar Prenomina Predefinida. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P12HU **Número de HU: 12**
Nombre: Exportar Prenómina.
Descripción: Se verifica si se puede exportar una prenómina.
Condiciones de ejecución: Se tiene que tener una prenómina creada para poder exportarla.
Entrada/Pasos de ejecución: 1. Se da clic en el botón de Exportar de la tabla. 2. Se da clic en el botón Exportar.
Resultado esperado: Se exporta la prenómina.
Evaluación de la prueba: Satisfactorio

Tabla 90:Exportar prenómina. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P13HU **Número de HU: 13**
Nombre: Agregar Incidencia.
Descripción: Se verifica que se cree una incidencia.
Condiciones de ejecución: Se debe seleccionar una fecha, una clave, agregar una descripción y cantidad de días.
Entrada/Pasos de ejecución: 1. Se completan los campos y se da clic en el botón Salvar.
Resultado esperado: Se crea la incidencia.
Evaluación de la prueba: Satisfactorio

Tabla 91:Agregar incidencia. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P14HU **Número de HU: 14**

Nombre: Modificar incidencia.
Descripción: Se verifica que se modifique una incidencia.
Condiciones de ejecución: Se debe tener una incidencia creada y modificar por lo menos un campo.
Entrada/Pasos de ejecución: 1. Se modifican los campos y se da clic en el botón Salvar.
Resultado esperado: Se modifica la incidencia.
Evaluación de la prueba: Satisfactorio

Tabla 92:Modificar incidencia. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P15HU **Número de HU: 15**
Nombre: Eliminar incidencia.
Descripción: Se verifica si se puede eliminar una incidencia.
Condiciones de ejecución: Se tiene que tener una incidencia creada para poder eliminarla.
Entrada/Pasos de ejecución: 1. Se da clic en el botón del icono eliminar. 2. Se da clic en el botón OK.
Resultado esperado: Se elimina la incidencia.
Evaluación de la prueba: Satisfactorio

Tabla 93:Eliminar incidencia. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P16HU **Número de HU: 16**
Nombre: Agregar Clave.
Descripción: Se verifica que se cree una clave.
Condiciones de ejecución: Se debe agregar un número de clave y la descripción.
Entrada/Pasos de ejecución: 1. Se completan los campos y se da clic en el botón Salvar.
Resultado esperado: Se crea la clave.
Evaluación de la prueba: Satisfactorio

Tabla 94:Agregar clave. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P117HU **Número de HU: 117**
Nombre: Modificar clave.

Descripción: Se verifica que se modifique una clave.
Condiciones de ejecución: Se debe tener una clave creada y modificar por lo menos un campo.
Entrada/Pasos de ejecución: 1. Se modifican los campos y se da clic en el botón Salvar.
Resultado esperado: Se modifica la clave.
Evaluación de la prueba: Satisfactorio

Tabla 95:Modificar clave. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P18HU **Número de HU: 18**
Nombre: Eliminar clave.
Descripción: Se verifica si se puede eliminar una clave.
Condiciones de ejecución: Se tiene que tener una clave creada para poder eliminarla.
Entrada/Pasos de ejecución: 1. Se da clic en el botón del icono eliminar. 2. Se da clic en el botón OK.
Resultado esperado: Se elimina la clave.
Evaluación de la prueba: Satisfactorio

Tabla 96:Eliminar clave. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P19HU **Número de HU: 19**
Nombre: Escanear código barra.
Descripción: Se verifica que se decodifique el código barra .
Condiciones de ejecución: El trabajador tiene que estar generado en el sistema y tener una hoja de firma actualizada.
Entrada/Pasos de ejecución: 1. Escanear el código barra. 2. Llenar los campos correspondientes y dar clic en el botón salvar
Resultado esperado: Se actualiza su asistencia en su hoja de firma correspondiente. Se emite un mensaje de error en caso de que: Los servicios UCI esten cahídos. El código barra de un trabajador no sea válido para el Sistema.

No este creada la hoja de firma actualizada del trabajador.
Evaluación de la prueba: Satisfactorio

Tabla 97:Buscar por código barra. Fuente:(Elaboración propia)

Caso de Prueba de Aceptación
Código: P20HU **Número de HU: 19**
Nombre: Escribir código.
Descripción: Se escribe el código barra descodificado. .
Condiciones de ejecución: El trabajador tiene que estar generado en el sistema y tener una hoja de firma actualizada.
Entrada/Pasos de ejecución: 1. Escribir el código. 2. Llenar los campos correspondientes y dar clic en el botón salvar
Resultado esperado: Se actualiza su asistencia en su hoja de firma correspondiente. Se emite un mensaje de error en caso de que: Los servicios UCI esten cahídos. El código barra de un trabajador no sea válido para el Sistema. No este creada la hoja de firma actualizada del trabajador.
Evaluación de la prueba: Satisfactorio

Tabla 98.Escribir código. Fuente:(Elaboración propia)

Anexo 5: *Acta de* Aceptación

Anexo 6: Pruebas funcionales

Tabla 99.Caso de Prueba. Autenticar Usuario. Fuente (Elaboración propia)

Escenario	Usuario	Contraseña	Respuesta del sistema	Resultado de la prueba
EC 1.1 Autenticar Usuario correctamente	[V,I,N/A]	[V,I,N/A]	Muestra la página principal y un mensaje"Autorización exitosa"	Se autenticó satisfactoriamente
	V	V		
	alejandrodmt	123456		
EC1.2 Campos vacíos	V	I	El sistema muestra el mensaje "Este campo no puede estar vacío"	Satisfactoria el sistema alerto al usuario del problema
	alejandrodmt			
EC1.3 Datos incorrectos	I	V	El sistema muestra el mensaje "Autorización denegada, usuario o contraseña incorrecta"	Satisfactoria el sistema alerto al usuario del problema
	edielmt	123456		

Escenario	Organismo	Área	Respuesta del sistema	Resultado de la prueba
EC3.1 Generar hojas de firmas exitosamente	[V,I,N/A]	[V,I,N/A]	Muestra la lista de las hojas de firmas creadas y el mensaje con la cantidad que se crearon	Se generaron satisfactoriamente
	V	V		
	UCI	Ciencia y		

		Tecnología	junto con “Hojas de firmas”	
EC3.2 Campos vacíos	V	I	El sistema muestra el mensaje “Rellene este campo”	Satisfactoria el sistema alerto al usuario del problema
	UCI			

Tabla 100.Caso de Prueba. Generar Trabajadores. Fuente (Elaboración propia)

Escenario	Organismo	Área	Respuesta del sistema	Resultado de la prueba
EC4.1 Confirmar eliminar	[V,I,N/A]	[V,I,N/A]	Muestra el mensaje: ”¿Está seguro de eliminar?	Se mostró de forma satisfactoria un mensaje para confirmar.
	N/A	N/A		
EC4.2 Eliminar OK	N/A	N/A	Muestra el mensaje: ″Eliminado″.	Se eliminó de forma satisfactoria.
EC4.3 Eliminar Cancel	N/A	N/A	No ocurre nada	Se cancelo de forma satisfactoria la acción de eliminar

Tabla 101.Caso de Prueba. Eliminar. Fuente (Elaboración propia)

Escenario	Aprobado.po	Revisado.	Con fec.	Fecha	Entidad	Organismo	Área	Respuesta	del	Resultado	de	la

	r	por	por					sistema	prueba
EC5.1 Crear prenómina exitosamente	[V,I, N/A]	[V,I, N/A]	[V,I, N/A]	[V,I, N/A]	[V,I, N/A]	[V,I, N/A]	[V,I, N/A]	Muestra la lista de las prenóminas creadas y el mensaje “Guardado”	Se creo satisfactoria mente la prenó mina
	V	V	V	V	V	V	V		
	Yordankis	Yordankis	Yordankis	10/12/2022	Facultad 4	UCI	Ciencia y Tecnología		
EC5.2 Campos vacíos	V	V	V	I	V	V	V	El sistema muestra el mensaje “Rellene este campo”	Se alerto de forma satisfactoria el problema de campos vacíos
	Yordankis	Yordankis	Yordankis		Facultad 4	UCI	Ciencia y Tecnología		
EC5.3 Fechas iguales	V	V	V	V	V	V	V	El sistema muestra el mensaje “Ya existe una prenómina de este mes”	Se alerto de forma satisfactoria el problema de prenóminas con fechas iguales
	Yordankis	Yordankis	Yordankis	10/12/2022	Facultad 4	UCI	Ciencia y Tecnología		

Tabla 102.Caso de Prueba. Crear prenómina. Fuente (Elaboración propia)