UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN SOFTWARE

INFORME FINAL DEL TRABAJO DE INTEGRACIÓN CURRICULAR, MODALIDAD: PROYECTO DE INVESTIGACIÓN

TEMA:

"ESTUDIO DEL FRAMEWORK JAKARTA EE PARA LA ACTUALIZACIÓN DE MÓDULOS JAVA EE A JAKARTA EE EN EL SISTEMA "SIGEPAA" DE LA EMPRESA PÚBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE EPAA-AA".

Trabajo de integración curricular previo a la obtención del título de Ingeniero en Software presentado ante la noble Universidad Técnica del Norte.

Línea de investigación: Desarrollo, aplicación de software y cyber security (seguridad cibernética).

AUTOR:

Esteban Alexis Narváez Arcos

DIRECTOR:

Msc. Pedro David Granda Gudiño

Ibarra, noviembre 2024

UNIVERSIDAD TÉCNICA DEL NORTE

DIRECCIÓN DE BIBLIOTECA

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CEDULA DE IDENTIDAD	040180795-3		
APELLIDOS Y NOMBRES	NARVÁEZ ARCOS ESTEBAN ALEXIS		
DIRECCIÓN	ATUNTAQUI		
EMAIL	eanarvaeza@utn.edu.ec tebitan789@hotmail.com		
TELÉFONO FIJO		**TELÉFONO MÓVIL**	0987274919

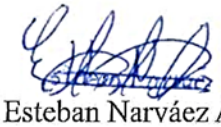
DATOS DE LA OBRA	
TITULO	ESTUDIO DEL FRAMEWORK JAKARTA EE PARA LA ACTUALIZACIÓN DE MÓDULOS JAVA EE A JAKARTA EE EN EL SISTEMA "SIGEPAA" DE LA EMPRESA PÚBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE EPAA-AA
AUTOR(ES)	NARVÁEZ ARCOS ESTEBAN ALEXIS
FECHA: DD/MM/AAAA	15/10/2024
PROGRAMA	PREGRADO
TITULO POR EL QUE OPTA	INGENIERO EN SOFTWARE
ASESOR/DIRECTOR	MSC. MAURICIO REA., MSC. PEDRO GRANDA.

2. CONSTANCIA

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 29 días del mes de noviembre del 2024

EL AUTOR:



Esteban Narváez A.

C.I: 040180795-3

CERTIFICADO DIRECTOR

Ibarra 15 de octubre del 2024

CERTIFICACIÓN DIRECTOR DEL TRABAJO DE TITULACIÓN

Por medio del presente yo MSc. Pedro Granda Gudiño., certifico que el Sr. Esteban Alexis Narvaez Arcos portador de la cedula de ciudadanía número· 040180795-3, ha trabajado en el desarrollo del proyecto de grado **"ESTUDIO DEL FRAMEWORK JAKARTA EE PARA LA ACTUALIZACIÓN DE MÓDULOS JAVA EE A JAKARTA EE EN EL SISTEMA "SIGEPAA" DE LA EMPRESA PÚBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE EPAA-AA.",** previo a la obtención del Título de Ingeniero en Software realizado con interés profesional y responsabilidad que certifico en honor a la verdad.

Es todo en cuanto puedo certificar a la verdad

Atentamente

MSc. Pedro Granda

DIRECTOR DE TRABAJO DE GRADO

CERTIFICADO EMPRESA PÚBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE

EMPRESA PÚBLICA DE AGUA PORTABLE Y ALCANTARILLADO DE ANTONIO ANTE EPAA-AA,

UNIDAD DE ADMINISTRACIÓN DE TALENTO HUMANO

C E R T I F I C A

Que, el señor Esteban Alexis Narváez Arcos portador de la cédula de ciudadanía Nro. 040180795-3 egresado de la Universidad Técnica del Norte, Facultad de Ingeniería en Ciencias Aplicadas FICA, realizó su **TEMA DE TESIS "ESTUDIO DE FRAMEWORK JAKARTA EE PARA LA ACTUALIZACION DE MODULOS JAVA EE A JAKARTA EE EN EL SISTEMA SIGEPA DE LA EMPRESA PUBLICA DE AGUA POTABLE Y ALCANTARILADO DE ANTONIO ANTE DE ATUNTAQUI EPAA-AA"**, proyecto de tesis recibido por en el área de TICS de la **EMPRESA PÚBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE EPAA-AA** a entera satisfacción.

Es todo cuanto puedo certificar, faculto a la parte interesada hacer uso del presente certificado en el momento que crea conveniente.

Atuntaqui, 5 de agosto de 2024

Atentamente:

SANDY ANNABELL BORJA PALMA

Ing. Sandy Borja Palma

ESPECIALISTA DE TALENTO HUMANO EPAA-AA

DEDICATORIA

Dedico este trabajo a mis queridos padres Fernando Narváez y Germania Arcos, a quienes quiero expresar mi más sincero agradecimiento por su apoyo incondicional a lo largo de todos estos años. Siempre han sido la voz que me recuerda que un futuro mejor nos aguarda, la confianza que han depositado en mis proyectos, mis ideas e incluso mis locuras ha sido un faro en mi camino. Cada jalón de orejas que me han dado cuando lo necesitaba ha sido una lección valiosa, y cada noche y madrugada de trabajo arduo y diversión compartida ha sido un tesoro inolvidable.

Su amor, tiempo, paciencia y dedicación inquebrantable son tesoros invaluables que atesoro profundamente. Nunca se doblaron ante los golpes que la vida nos ha propinado, y siempre recordaré los sacrificios que han hecho para mi bienestar. Siempre tuve lo mejor a su lado, y por todo esto y mucho más, les agradezco infinitamente. Los amo profundamente.

A mis queridos hermanos Jhostin Narvaez, Cristhian Arcos y Leonela Moreno, familiares y amigos quiero expresarte mi cariño y admiración por todos los momentos, tanto buenos como desafiantes, que hemos compartido, les agradezco de todo corazón por su apoyo, amor y presencia en mi vida. Son una parte fundamental de mi viaje, y espero continuar compartiendo momentos inolvidables juntos en el futuro.

A la vida misma, en este momento de reflexión, deseo rendir homenaje a la vida que nos ha permitido persistir, a pesar de los fracasos y desafíos. Agradezco al tiempo que aún no se ha agotado y que me ha dado la oportunidad de crecer y evolucionar. Mi constancia, esfuerzo y dedicación son los pilares que me han permitido avanzar. Esteban Alexis Narváez Arcos

AGRADECIMIENTOS

Quiero expresar mi más profundo agradecimiento a mis padres y hermanos por su constante apoyo. Aprecio enormemente los sacrificios que han hecho por mí, su apoyo incondicional, el respaldo económico, y todo el amor y dedicación que me han brindado a lo largo de mi vida.

Mi sincero agradecimiento a la Universidad Técnica del Norte, a sus docentes y tutores, por proporcionarme valiosos conocimientos y experiencias que han sido clave para abrir nuevas oportunidades en mi futuro, especialmente en el campo de la tecnología.

Quiero destacar y agradecer especialmente al MSc. Pedro Granda y al MSc. Mauricio Rea, mi director y asesor de tesis, por su orientación y apoyo incondicional. No solo me han transmitido conocimientos cruciales para mi carrera profesional, sino que también me han compartido principios y consejos que han sido fundamentales para mi desarrollo académico y personal.

Expreso mi más profundo agradecimiento a la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante (EPAA-AA) por su constante apoyo y colaboración, los cuales han sido esenciales para el avance de esta tesis.

Asimismo, quiero reconocer especialmente a la Ing. María Fernanda Aguirre por su invaluable apoyo y guía durante este proyecto. Su experiencia y conocimientos han sido cruciales, brindándome el ánimo y la confianza para enfrentar los desafíos que surgieron. Su dedicación y compromiso con mi desarrollo académico han sido valioso, y su influencia ha dejado una marca significativa en mi formación personal y profesional

Esteban Alexis Narváez Arcos

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

Este proyecto se centra en el estudio y análisis del framework Jakarta EE como parte esencial de un proceso de actualización de módulos Java EE en el sistema “SIGEPAA” de la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante (EPAA-AA). El objetivo principal de este estudio es evaluar la viabilidad y los beneficios de migrar de Java EE a Jakarta EE en un entorno crítico como es el sistema de gestión de la EPAA-AA.

El proyecto de integración curricular aborda la importancia de mantener los sistemas empresariales actualizados y compatibles con las últimas tecnologías para garantizar la eficiencia, la seguridad y la escalabilidad. Jakarta EE se presenta como una opción lógica debido a su enfoque en aplicaciones empresariales basadas en Java, y su compatibilidad con las últimas tendencias y estándares en desarrollo de software.

Se realiza un análisis exhaustivo de los componentes y características clave de Jakarta EE, destacando su capacidad para ofrecer una arquitectura robusta y modular. Se investiga cómo esta transición puede mejorar la flexibilidad y capacidad de respuesta del sistema EPAA-AA “SIGEPAA”, así como su capacidad para integrarse con otros sistemas y adaptarse a futuras evoluciones tecnológicas.

El proceso de migración se aborda desde una perspectiva práctica, evaluando los desafíos potenciales, las estrategias de mitigación de riesgos y las mejores prácticas para llevar a cabo la actualización de los módulos de Java EE a Jakarta EE. Se presta especial atención a asegurar la continuidad operativa durante el proceso de migración.

Este estudio concluye con una evaluación de los beneficios esperados, como la mejora en la eficiencia operativa, la seguridad y la capacidad de expansión del sistema “SIGEPAA”.

ABSTRACT

This project focuses on the study and analysis of the Jakarta EE framework as an essential part of a process of updating Java EE modules in the "SIGEPAA" system of the Public Company of Drinking Water and Sewerage of Antonio Ante (EPAA-AA). The main objective of this study is to evaluate the feasibility and benefits of migrating from Java EE to Jakarta EE in a critical environment such as the EPAA-AA management system.

The curriculum integration project addresses the importance of keeping business systems up to date and compatible with the latest technologies to ensure efficiency, security and scalability. Jakarta EE presents itself as a logical choice due to its focus on Java-based enterprise applications, and its compatibility with the latest trends and standards in software development.

A comprehensive analysis is performed on the key components and features of Jakarta EE, highlighting its ability to deliver a robust and modular architecture. It is investigated how this transition can improve the flexibility and responsiveness of the EPAA-AA "SIGEPAA" system, as well as its ability to integrate with other systems and adapt to future technological evolutions.

The migration process is approached from a practical perspective, evaluating potential challenges, risk mitigation strategies and best practices to carry out the upgrade of Java EE modules to Jakarta EE. Special attention is paid to ensuring operational continuity during the migration process.

This study concludes with an evaluation of the expected benefits, such as improvement in operational efficiency, security and expansion capacity of the "SIGEPAA" system.

INTRODUCCIÓN

En Imbabura, las empresas encargadas de la distribución de agua potable se enfocan en generar mayor beneficio en lo institucional como para la población; es decir, buscan que cada política pública aplicada de un proceso más amplio en la calidad del servicio como del producto. En lo institucional se buscaría desarrollo en su sistema integrado (software), teniendo en cuenta la historia de varios años de trabajo para la sociedad, con un sistema tecnológico tan ambiguo que necesita reforzar su tecnología internamente.

Ahora, a nivel cantonal se busca tener una visión amplia brindando servicios de calidad, donde su principal proyección es la satisfacción del cliente; por ello, la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante, se generaliza en ampliar su visión y reforzar un sistema integrado que brinda mayor seguridad al momento de poner en práctica su nueva meta, que es el servicio agua potable y alcantarillado más eficiente y eficaz.

Situación Actual

La empresa EPAA-AA debido a su constante crecimiento y excelente calidad de servicio muestran que el sistema antiguo ha experimentado problemas de seguridad, rendimiento y escalabilidad ya que tiene una. Este sistema ha sido fundamental para la gestión de sus áreas, como Comercialización, Finanzas, Área Técnica, Gerencia General, Estaciones de Bombeo y Plantas de Tratamiento. Sin embargo, con el tiempo, se han identificado limitaciones y desafíos asociados a su uso.

La falta de actualizaciones tecnológicas ha llevado a vulnerabilidades en la seguridad de los datos y ha limitado la capacidad de la empresa EPAA-AA para adaptarse a nuevas tecnologías y soluciones avanzadas.

PROSPECTIVA

Con el presente trabajo se pretende llevar a cabo una transición tecnológica clave para mejorar la eficiencia, seguridad y capacidad de expansión del sistema SIGEPAA teniendo como objetivo transformar y fortalecer la infraestructura tecnológica de la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante EPAA-AA. para enfrentar los desafíos tecnológicos actuales y futuros, brindando una base sólida para el funcionamiento y el crecimiento continuo de la empresa.

PLANTEAMIENTO DEL PROBLEMA

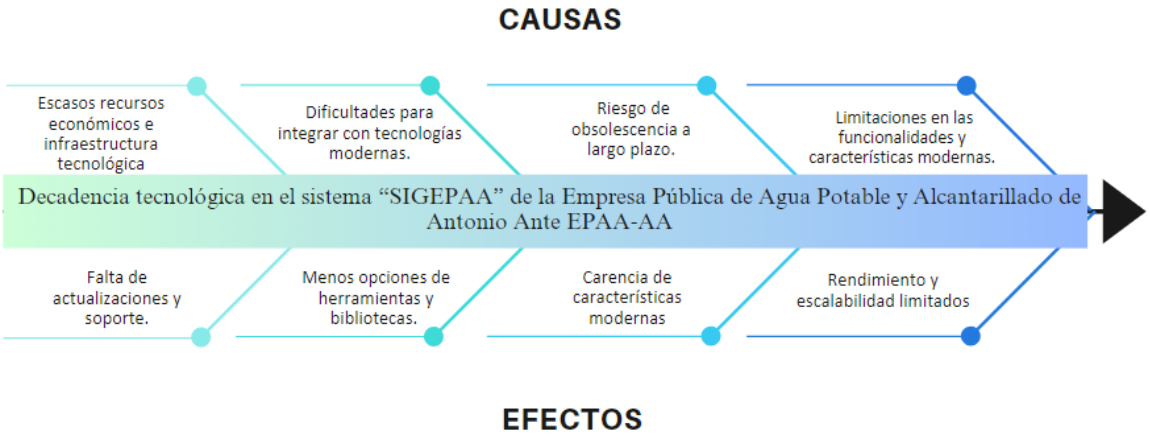
La migración de Java EE 8 a Jakarta EE es un proceso complejo que ha sido destacado en varios estudios (Juneau & Telang, 2022) (Elder, 2020). Estos estudios señalan que los sistemas desarrollados hace nueve años atrás con tecnologías antiguas son actualmente vulnerables, inseguros, incompatibles con nuevas tecnologías y carecen de soporte en mantenimiento.

Jakarta EE, siendo una continuación de Java EE, introduce algunas diferencias significativas que deben tenerse en cuenta durante el proceso de migración. Estas diferencias pueden implicar cambios en la sintaxis, el uso de nuevas bibliotecas y API, así como la eliminación de API obsoletas. Cabe destacar que estos cambios varían según las nuevas características y actualizaciones de las plataformas o frameworks utilizados (Castillo C. F., 2021).

Es importante evaluar cómo estos cambios pueden afectar el sistema existente y tomar medidas adecuadas para abordarlos. La migración a Jakarta EE puede afectar la estructura y arquitectura de la aplicación, lo que a su vez puede tener impacto en su diseño y funcionamiento (Manelli & Zambon, 2020).

Los estudios mencionados resaltan la importancia de migrar de Java EE 8 a Jakarta EE debido a la vulnerabilidad, inseguridad y falta de soporte en los sistemas desarrollados con tecnologías antiguas. Sin embargo, se debe ser consciente de las diferencias significativas entre ambas versiones y tomar las medidas necesarias para abordar los cambios en la sintaxis, el uso de bibliotecas y API, así como la eliminación de API obsoletas, con el fin de garantizar una migración exitosa.

Figura 1 Diagrama de Ishikawa



Nota: Elaborado por: Narváez Esteban, basado en (Diagrama de Ishikawa, 1943)

OBJETIVOS

Objetivo General

Actualizar el módulo de Secretaría del sistema "SIGEPAA" hacia el framework Jakarta EE para fortalecer la arquitectura y características de seguridad en la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante EPAA-AA.

Objetivos Específicos

- Determinar la situación actual del módulo de secretaria del sistema "SIGEPAA" con el fin de establecer los puntos de mejora en su arquitectura.

- Definir un marco teórico con respecto a la arquitectura y características principales del framework Jakarta EE para determinar el impacto de un proceso de actualización.
- Migrar los módulos Java EE 8 del aplicativo “SIGEPAA” para que utilice el nuevo framework de Jakarta EE.
- Evaluar el adecuado funcionamiento del aplicativo “SIGEPAA” bajo la norma ISO/IEC 25010.

ALCANCE

El sistema SIGEPAA cuenta con los siguientes módulos, Módulo Coordinación de Gerencia-sub, módulo Secretaria (Manejo de archivos digitales de todas las áreas de la EPAA-AA) / Módulo Comercialización-sub módulo (Cortes y Reconexión)/Módulo de Área Técnica-sub módulo de Mantenimiento (Formulario de Producción y Mantenimiento) /Módulo Talento Humano (Formulario de Permisos) /Módulo de Tics-Administrador/Módulo DAF sub Módulos-Activos Fijos-Contabilidad-Tesorería-Recaudación/Módulo Asesoría Jurídica.

De los módulos antes mencionados se selecciona el Módulo de Secretaría. En este módulo se maneja toda la información de los archivos digitales que se realiza dentro de la EPAA-AA.

Recopilar la información necesaria para realizar un análisis exhaustivo de la aplicación Java EE existente para comprender su arquitectura, tecnologías utilizadas, dependencias y requisitos funcionales.

Estudiar las diferencias entre Java EE y Jakarta EE para comprender las nuevas características y mejoras de Jakarta EE, así como conocer las implicaciones de migración.

Identificar los cambios necesarios en el código, configuración y dependencias para que la aplicación sea compatible con Jakarta EE. Esto incluye actualizar los paquetes y nombres, resolver incompatibilidades de dependencias y reemplazar API’s obsoletas.

Establecer un plan de proyecto detallado que incluya las tareas, recursos y plazos requeridos para la migración a Jakarta EE realizando un seguimiento regular para asegurar que se cumplan los objetivos establecidos.

Asegurar mediante la norma ISO/IEC 25010 el correcto funcionamiento del aplicativo migrado.

La norma ISO/IEC 25000, también conocida como la serie SQuaRE (System and Software Quality Requirements and Evaluation), incluye la sección de usabilidad como una de las características de calidad del producto de software y sistemas de información. Esta sección se denomina "Características de calidad del producto - Usabilidad" (Product Quality Characteristics - Usability) y se encuentra dentro del estándar ISO/IEC 25010.

La usabilidad es una característica crítica para cualquier producto de software o sistema de información, ya que se enfoca en la facilidad con la que los usuarios pueden aprender, utilizar y comprender el producto, así como su satisfacción general durante el proceso de interacción.

Para la validación de actualización se utilizará la sub característica de Operabilidad del apartado de usabilidad de la norma ISO/IEC 25010 para asegurar el funcionamiento adecuado y la facilidad de uso para el usuario.

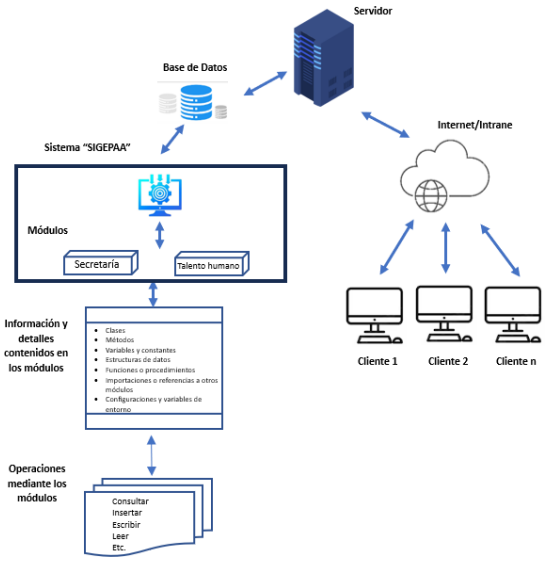
Operabilidad

Capacidad del producto que permite al usuario operarlo y controlarlo con facilidad

(iso25000.com, 2022)

Documentar los cambios realizados, las nuevas configuraciones y cualquier consideración importante relacionada con la migración. Proporcionar capacitación a los miembros del equipo de desarrollo y a los usuarios finales sobre las actualizaciones y cambios introducidos por Jakarta EE.

Figura 2 Diagrama estructural



Fuente: Elaboración propia

METODOLOGÍA

Para el desarrollo de la presente tesis se comprenderá los cambios y diferencias clave entre Java EE 8 y Jakarta EE para actualizar los módulos del aplicativo “SIGEPAA”, con la finalidad de optimizar y mejorar la seguridad, arquitectura, soporte, rendimiento y actualización de API´s (Rubensa, 2019) en la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante EPAA-AA.

Para el cumplimiento del objetivo 1, se recopilará información detallada sobre los módulos del sistema "SIGEPAA" y establecer puntos de mejora en su arquitectura de manera

informada y fundamentada.

Para el cumplimiento del objetivo 2, se realizará una investigación para aprender sobre Jakarta EE y los cambios que presenta en las especificaciones, las estructuras de paquetes en un sistema, comprendiendo las nuevas características y mejoras introducidas en Jakarta EE.

Para el cumplimiento del objetivo 3, se analizará el código fuente del sistema "SIGEPAA", evaluando la compatibilidad de los módulos con el nuevo framework de Jakarta EE para realizar las modificaciones necesarias y que se integre correctamente con el actual framework de Jakarta EE.

Para el cumplimiento del objetivo 4, se realizará pruebas exhaustivas en los módulos ya actualizados para asegurarse de que el sistema funcione correctamente con el nuevo framework de Jakarta EE.

JUSTIFICACIÓN

El presente trabajo de titulación promueve el cumplimiento con uno de los objetivos de Desarrollo Sostenible (ODS) relacionado con el objetivo N°9: "Industria, innovación e infraestructura" (PNUD, 2017) ya que el proceso de migración que se realizará en la institución implica actualizaciones y cambios en la infraestructura tecnológica, además se enfoca en fomentar la aplicación de tecnologías innovadoras.

Java EE ha sido una plataforma de desarrollo empresarial confiable y estable durante muchos años. Sin embargo, debido a problemas relacionados con la evolución y la falta de innovación, se requería una renovación. Jakarta EE representa la continuación y evolución de Java EE, impulsada por la comunidad, con el objetivo de mantener y mejorar la plataforma de

manera sostenible (Scholtz & Tijms, 2022).

La transición de Java EE a Jakarta EE también implica una transición del proceso de desarrollo y gobernanza de Oracle a la Fundación Eclipse. Esto ha permitido una mayor apertura y transparencia en el desarrollo de la plataforma, así como una licencia más flexible y accesible para los usuarios y proveedores de servicios (Eclipse, s.f.).

Java EE ha sido durante mucho tiempo el estándar de facto en el desarrollo de aplicaciones empresariales en Java (Scholtz & Tijms, 2022). Al actualizar a Jakarta EE, las organizaciones pueden asegurarse de que están utilizando una plataforma que está respaldada por una comunidad activa y tiene el potencial de convertirse en el nuevo estándar de la industria.

Jakarta EE introduce mejoras y nuevas características en comparación con Java EE. Estas mejoras incluyen la modularización de la plataforma, permitiendo un desarrollo más ágil y adaptado a las necesidades específicas del proyecto (Späth, 2019). También se han actualizado las especificaciones de las API y se ha adoptado un enfoque basado en microservicios y contenedores, lo que facilita el desarrollo de aplicaciones en arquitecturas modernas.

- **¿Por qué realizar la investigación?** Porque realizar una investigación sobre la actualización de Java EE a Jakarta EE es importante para evaluar los beneficios, la compatibilidad, los costos, los recursos, la comunidad y el ecosistema, así como los requisitos futuros.

  Esta investigación proporcionará una base sólida para tomar decisiones informadas y garantizar el éxito de la actualización.
- **¿Para qué se hace la investigación?** Para que la investigación de actualizar Java EE a Jakarta EE se lleva a cabo comprendiendo la viabilidad, evaluar el impacto,

identificar los beneficios y mejoras, evaluar la compatibilidad y los riesgos, y evaluar el soporte y la comunidad. Estos aspectos ayudan a tomar decisiones informadas y planificar una migración exitosa hacia la plataforma Jakarta EE.

- **Relevancia** La actualización de Java EE a Jakarta EE es relevante debido a la continuidad y evolución que ofrece, el respaldo de una comunidad activa y el apoyo industrial, la adopción de tecnologías modernas, la adhesión a estándares y la portabilidad, y la transparencia y gobernanza que brinda.
- **Importancia** El estudio de actualizar Java EE a Jakarta EE es importante debido a los posibles aportes teóricos y prácticos que puede generar. Esto incluye el avance en el conocimiento técnico, la mejora de la eficiencia y el rendimiento, la adopción de tecnologías modernas, la mejora en la interoperabilidad y la portabilidad, y la contribución a la comunidad y al ecosistema.

Justificación Tecnológica

Actualizar Java EE a Jakarta EE se basa en la evolución y mejora continua de la plataforma, la adopción de tecnologías modernas, la portabilidad, el soporte de una comunidad activa y el cumplimiento de requisitos y regulaciones actuales. Estos aspectos aseguran que las organizaciones puedan desarrollar y mantener aplicaciones empresariales robustas, eficientes y adaptables en un entorno tecnológico en constante evolución.

Justificación Institucional

La actualización y optimización de un sistema mejora la eficiencia institucional al reducir los tiempos de espera y aumentar la productividad, lo que resulta en una mejor utilización de los recursos y una reducción de costos; además, se mejora la calidad del servicio al proporcionar

respuestas rápidas, información precisa y una interacción fluida, lo que incrementa la satisfacción del usuario y fortalece la reputación de la institución. La actualización también brinda mejores medidas de seguridad y cumplimiento normativo, garantizando la protección de los datos y la privacidad. Por último, permite la adopción de nuevas tecnologías e innovaciones, lo cual impulsa la competitividad empresarial.

RIESGOS

La tabla 1 identifica los principales desafíos asociados con la actualización de Java EE a Jakarta EE y describe estrategias de mitigación para cada uno. Esta tabla abarca aspectos como limitaciones presupuestarias, resistencia al cambio, dependencias de terceros y riesgos de compatibilidad, proporcionando un marco para abordar estos riesgos de manera efectiva.

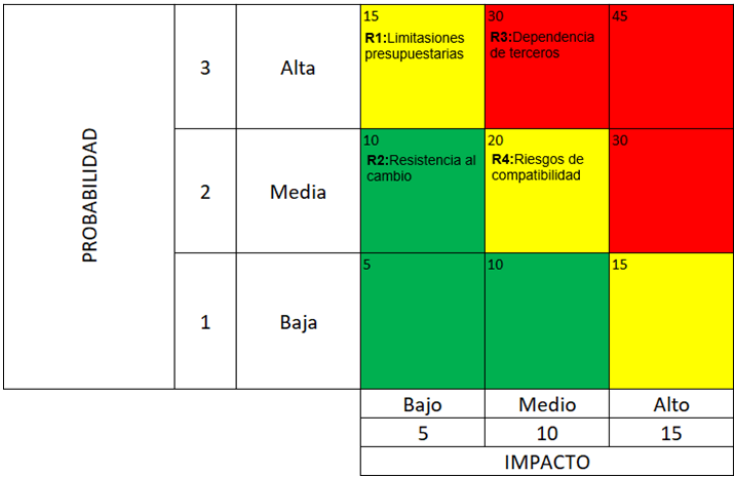
Tabla 1. Riesgos

Nro.	Riesgo	Estrategia de Mitigación
R1	Limitaciones presupuestarias	Priorizar las actividades y los aspectos clave de la actualización en función de los recursos disponibles. Considera la adopción de soluciones de código abierto en lugar de herramientas o servicios comerciales costosos.
R2	Resistencia al cambio	Comunicar de manera clara y transparente los motivos y beneficios de la actualización de Java EE a Jakarta EE. Explicando cómo la transición mejorará la eficiencia, la

		seguridad, el rendimiento u otros aspectos relevantes para la empresa
R3	Dependencias de terceros	Considerar desarrollar internamente componentes o funcionalidades específicas en lugar de depender completamente de dependencias externas. Esto puede brindar mayor control y flexibilidad sobre las tecnologías utilizadas y reducir la dependencia de terceros.
R4	Riesgos de compatibilidad	Revisa todas las bibliotecas y dependencias utilizadas en la aplicación y asegúrate de que sean compatibles con Jakarta EE

Nota: Elaborado por: Narváez Esteban, basado en (Riesgos en la aplicación de Jakarta EnterPrise, 2021)

Tabla 2. Gráfico de representación del análisis de riesgos



Nota: Elaborado por: Narváez Esteban, basado en (Análisis de Riesgos a Jakarta EnterPrise, 2023)

FUNDAMENTACION TEORICA

ANTECEDENTES

La Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante (EPAA-AA) proporciona servicios de agua potable y alcantarillado a la población del cantón, con un enfoque constante en la satisfacción de la comunidad. EPAA-AA por su constante crecimiento y excelente calidad de servicio muestran que el sistema SIGEPAA ha tenido problemas de rendimiento, escalabilidad y seguridad, ya que tiene varios años y ha operado con un sistema tecnológico antiguo en sus operaciones.

Este sistema ha sido fundamental para la gestión de sus áreas, como Comercialización, Finanzas, Área Técnica, Gerencia General, Estaciones de Bombeo y Plantas de Tratamiento. Sin embargo, con el tiempo, se han identificado limitaciones y desafíos asociados a su uso.

La carencia de actualizaciones tecnológicas ha llevado a vulnerabilidades en la seguridad de los datos y ha limitado la capacidad de la empresa para adaptarse a nuevas tecnologías y soluciones avanzadas.

CAPÍTULO I

MARCO TEÓRICO

Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante EPAA-AA

1.1. Introducción

El presente trabajo tiene como finalidad mejorar la infraestructura tecnológica mediante la actualización del framework Java EE a Jakarta EE para aumentar la seguridad y el rendimiento, así como habilitar nuevas soluciones tecnológicas avanzadas. El objetivo final es contribuir al éxito continuo de la empresa y al logro de sus objetivos estratégicos.

1.1.1. Justificación de la Actualización a Jakarta EE

La decisión de migrar a Jakarta EE se fundamenta en la necesidad de mantenerse al día con los estándares tecnológicos y asegurar la viabilidad a largo plazo del sistema.

El autor (Lozano Peralta, 2020) indica que: “La evolución de la tecnología requiere cambios constantes que puedan adaptarse a las demandas de los diferentes dominios en la sociedad (industria, comercio, educación y hogar) (pág. 42). Por ello, Jakarta EE ofrece un conjunto robusto de características, seguridad mejorada y un marco de trabajo evolutivo que respalda la modernización continua del sistema. Por ello, se fija objetivos que generan mejor proceso para el sistema de actualización entre ellos:

- Mejorar la Eficiencia y Escalabilidad al optimizar el rendimiento del módulo de secretaría para garantizar respuestas rápidas y eficientes incluso ante un aumento significativo en la carga de trabajo.

- Adaptar la Arquitectura a Jakarta EE para alinear la arquitectura con los estándares de Jakarta EE, aprovechando sus características avanzadas y asegurando la compatibilidad con futuras actualizaciones.
- Garantizar Continuidad y Compatibilidad asegurando que la actualización no afecte negativamente las funciones existentes y que el sistema sea compatible con versiones anteriores para una transición fluida.

1.2. Descripción del Módulo de Secretaría en el Sistema "SIGEPAA"

El módulo de secretaría en el sistema "SIGEPAA" desempeña un rol crítico y central en la gestión integral de archivos empresariales, abordando aspectos administrativos de manera eficiente y coordinada. Este componente es fundamental para asegurar un flujo de trabajo eficiente, continuo y organizado, en donde una administración efectiva de los recursos, las respuestas ágiles de necesidades empresariales que se generan mediante algunas funciones y características claves.

Funciones clave

a) **Manejo de Documentación Empresarial*:*** El módulo de secretaría facilita la gestión de información crucial en cuanto al manejo de archivos dentro de la EPAA-AA. Permite la carga, clasificación y recuperación eficiente de documentos empresariales, incluyendo informes financieros, políticas institucionales y comunicaciones oficiales.

b) **Seguimiento de Trámites y Procesos*:*** Proporciona una plataforma centralizada para el seguimiento de trámites y procesos administrativos, garantizando una gestión eficaz de solicitudes, autorizaciones y permisos.

c) **Generación de Informes y Estadísticas*:*** Facilita la generación de informes y estadísticas relevantes para evaluar el rendimiento, la eficiencia administrativa y el cumplimiento de objetivos institucionales.

Características Clave

a) **Interfaz Intuitiva:** Está diseñada para ser intuitiva y fácil de usar, facilitando a los usuarios el acceso rápido y eficiente a las funciones clave.
b) **Seguridad y Privacidad:** Implementa medidas de seguridad sólidas para asegurar la confidencialidad y protección de los datos sensibles, cumpliendo con estándares de privacidad institucionales.
c) **Integración con Otros Módulos:** Se integra de manera fluida con otros módulos del sistema "SIGEPAA", como el módulo recursos humanos, para garantizar la coherencia de la información a lo largo de toda la plataforma.
d) **Auditoría de Acceso:** Registra y audita el acceso a información crítica, garantizando la trazabilidad y cumplimiento normativo.
e) **Flexibilidad y Adaptabilidad:** El sistema está diseñado para adaptarse de manera flexible a las transformaciones del entorno empresarial, facilitando actualizaciones continuas y ajustes según las demandas cambiantes del sistema.

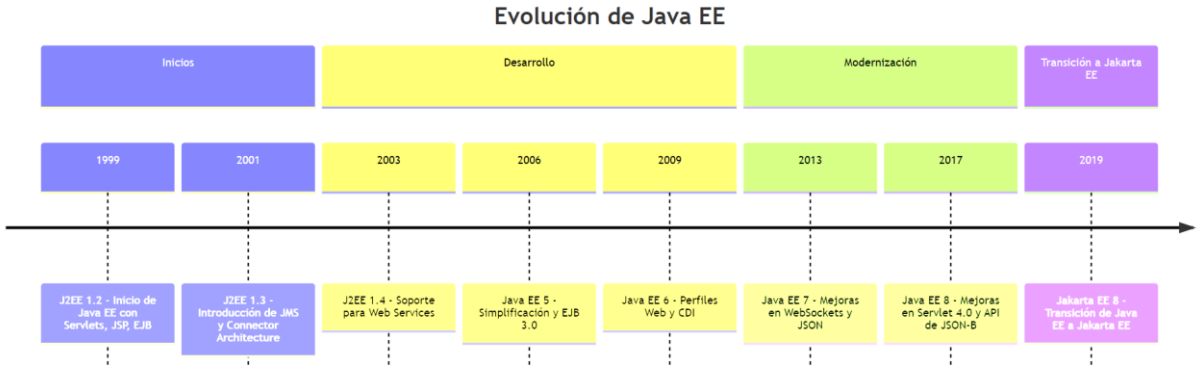
El módulo de secretaría en el sistema "SIGEPAA" constituye un pilar fundamental para la gestión eficiente de archivos empresariales y la administración de manera cohesiva para impulsar el éxito organizacional.

1.3. Introducción a Jakarta EE y Java EE

El lenguaje de programación Java fue desarrollado por James Gosling y el equipo de Sun Microsystems en 1991, siendo lanzado al mercado en 1995. Su principal objetivo era permitir la creación de pequeñas aplicaciones interactivas, conocidas como applets, diseñadas para ejecutarse en navegadores web. Aunque la visión inicial del aplicativo Java se convirtiese en la nueva forma de programar aplicaciones no se cumplió, por eso Java ha evolucionado con gran éxito gracias a su carácter multiplataforma y su adopción en diversas áreas.

En estos procesos, siempre es necesario tener en cuenta que las fases de evolución histórica cambian con el pasar de los tiempos; es decir, cada sistema debería estar compuesto por un sin número de herramientas y especificación que garanticen un buen desarrollo de los aplicativos, (Castillo C. F., 2022) menciona que: “Jakarta EE es una plataforma derivada de Java Enterprise (JEE) que se compone de un amplio conjunto de especificaciones, las cuales oermiten cubrir la mayoría de las necesidades del desarrollo de una aplicación empresarial” . (pág. 19)

Figura 3 Evolución de Jave EE



Nota. Por Esteban Narváez, 2024.

Se puede identificar dos componentes que refieren a un sistema operativo institucional, en donde lo principal es buscar que contienen cada uno de ellos, en este caso nos referimos a:

1.3.1. Java EE

El autor (Veracruz, 2022) menciona que Jakarta EE es la nueva plataforma de Java Enterprise Edition que ha sido creada para ser de código abierto, ya que Java EE era controlada por Oracle. Jakarta EE incluye especificaciones clave como Jakarta Enterprise JavaBeans (EJB), Jakarta Persistence (JPA) y Jakarta RESTful Web Services (JAX-RS). (pág. 42)

Este proceso reconoce que Java ha sido exitoso al no limitarse a los navegadores y abarcar una amplia variedad de plataformas, desde teléfonos móviles hasta aplicaciones de escritorio. Además, ha destacado en el desarrollo de APIs multiplataforma y en la programación en entornos de Internet, como en Java EE (Enterprise Edition), que se enfoca en aplicaciones empresariales multiusuario que requieren conectividad y seguridad.

Los autores (Ordax Cassá & Ocaña Díaz, 2009) lo definen como la plataforma Java Enterprise Edition (Java EE) define las APIs y requerimientos necesarios para poder ejecutar aplicaciones Java servidoras, con todo lo que ello supone: cliente-servidor, multiusuario, transaccionalidad, escalabilidad, etc., definitiva, características que no eran importantes o imprescindibles en aplicaciones de escritorio. (pág. 4)

Por ello, el mundo de las aplicaciones empresariales se lo simplificaría en la dominación de sistemas propios donde la convergencia hacia estándares abiertos en la industria informática que posiblemente permitía un mayor aprovechamiento y reutilización de recursos. En este contexto, Java al estar enlazado con estándares abiertos y el ofrecimiento de una amplia colección de servicios interconectables y transportables convierte una opción ideal para las aplicaciones empresariales y corporativas.

1.3.2. Jakarta EE

Los desarrolladores actuales reconocen la necesidad de aplicaciones empresariales portátiles, transaccionales y distribuidas que aprovechen la tecnología del lado del servidor de manera rápida, segura y confiable, el autor (Castillo C. F., 2021) afirma que: “Jakarta EE es una plataforma derivada de Java Enterprise Edition (JEE) que se compone de un amplio conjunto de especificaciones, las cuales permiten cubrir la mayoría de las necesidades del desarrollo de una aplicación empresarial” (pág. 14). De esta forma Jakarta EE ofrece un conjunto poderoso de API para facilitar el desarrollo de aplicaciones empresariales Java de manera más sencilla y rápida.

Se desarrolla a través del proceso de especificación de Jakarta EE, donde grupos de expertos crean las Especificaciones de Jakarta para definir las tecnologías de EE. La plataforma utiliza un modelo de programación simplificado con anotaciones en archivos fuente de Java, lo que permite una configuración más sencilla y flexible. La inyección de dependencia en Jakarta EE permite que el contenedor inserte automáticamente referencias a otros componentes o recursos necesarios mediante anotaciones.

1.3.3. Jakarta EE y su relación con Java EE.

Primero que todo, se puede decir que, Jakarta EE es un conjunto de tecnologías diseñadas para desarrollar y ejecutar aplicaciones empresariales en Java, cuya referencia ambigua es el conocerla como Java EE, en este caso Jakarta EE es una evolución de esta plataforma y se desarrolla de forma comunitaria y abierta, siendo liderada por la Fundación Eclipse. El autor (Castillo C. F.) indica que: “Las aplicaciones empresariales deben disponer de algunos mecanismos eficientes para poder almacenarlos y recuperarlos en una base de datos” (pág. 24).

La relación entre Jakarta EE y Java EE radica en que Jakarta EE es una continuación y una extensión de Java EE. A principios de 2018, Oracle transfirió todas las especificaciones, tecnologías y las marcas comerciales relacionadas con Java EE a la Fundación Eclipse, y como resultado, Java EE pasó a llamarse Jakarta EE. Desde entonces, Jakarta EE ha estado desarrollándose para seguir siendo una plataforma de aplicación empresarial robusta y escalable, manteniendo siempre la compatibilidad con Java EE y evolucionando de acuerdo con las necesidades cambiantes del desarrollo de software empresarial.

Sin embargo, el autor (Shannon, 2020) destaca que, aunque Jakarta EE es una evolución de Java EE, existen algunas diferencias importantes entre ambas plataformas, como los nombres de paquetes y las marcas comerciales. Además, Jakarta EE tiene como objetivo principal moverse hacia una estructura de gobierno más abierta y comunitaria, en contraposición a la estructura de gobierno controlado por Oracle en el caso de Java EE (pág. 12).

Por ello se puede decir que, Jakarta EE es una continuación y una extensión de Java EE, ofreciendo un conjunto de tecnologías de software para el desarrollo de aplicaciones empresariales en Java. Aunque tienen una relación estrecha, existen algunas diferencias entre ambas plataformas y Jakarta EE se desarrolla de forma abierta y comunitaria bajo el liderazgo de la Fundación Eclipse. Jakarta EE es una plataforma para el desarrollo de aplicaciones empresariales utilizando Java. Su objetivo es brindar especificaciones y estándares que faciliten la creación de aplicaciones escalables, seguras y robustas.

1.3.4. Características y objetivos de Jakarta EE.

En su inicio, Jakarta Enterprise busca brindar a los desarrolladores las herramientas y tecnologías necesarias para construir aplicaciones empresariales robustas, escalables, seguras y

portátiles, al tiempo que mejora la productividad y la interoperabilidad. “Las aplicaciones empresariales deben disponer de algunos mecanismos eficientes para poder almacenarlos y recuperarlos de una base de datos” (Castillo C. F., 2022). Es necesario por ello generar un visón más amplia de algunas características y objetivos, como lo observamos a continuación:

Las características principales de Jakarta Enterprise como lo afirma (Berenguel Gómez , 2024) son como siguen:

a) **Arquitectura basada en componentes:** Jakarta Enterprise utiliza una arquitectura basada en componentes para facilitar el desarrollo de aplicaciones empresariales. Esto permite dividir el sistema en componentes reutilizables, lo que mejora el modularidad y la mantenibilidad del código.
b) **Plataforma neutra:** Jakarta Enterprise es una plataforma neutra y compatible con múltiples sistemas operativos y servidores de aplicaciones. Esto permite ejecutar las aplicaciones en diferentes entornos sin tener que realizar modificaciones importantes.
c) **Soporte para estándares:** Jakarta Enterprise sigue estándares industriales como Java EE, lo que garantiza la interoperabilidad con otras plataformas y facilita la integración con diferentes tecnologías.
d) **Escalabilidad y rendimiento:** Jakarta Enterprise está diseñado para aplicaciones empresariales de alto rendimiento y escalabilidad. Proporciona mecanismos para manejar la concurrencia, la transaccionalidad y la distribución de carga, lo que permite escalar la aplicación según las necesidades del negocio.
e) **Seguridad:** Jakarta Enterprise proporciona mecanismos para gestionar la seguridad de las aplicaciones, como autenticación, autorización y gestión de roles. También el soporte para la protección de datos y cifrado.

f) **Integración con sistemas existentes**: Jakarta Enterprise facilita la integración de las aplicaciones con sistemas existentes, como bases de datos, servicios web, colas de mensajes y sistemas de archivos.

g) **Desarrollo ágil:** Jakarta Enterprise es compatible con enfoques de desarrollo ágil, como DevOps, lo que permite una implementación rápida y continúa de las aplicaciones en un entorno empresarial. Estas características hacen de Jakarta Enterprise una plataforma robusta y completa para el desarrollo de aplicaciones empresariales. (pág. 17)

Dentro de cada aspecto o creación de aplicativos se demuestra características que resuelven dudas al momento de aplicarlo; es decir, es necesario la creación de aplicaciones con bases fundamentales que sobrelleven la eficiencia de una empresa, pero como hacerlo, sobrecargando objetivos que demuestren la complementariedad de dicho factor.

El autor (Castillo C. F., 2022) indica los siguientes objetivos principales:

a) Proporcionar una plataforma estándar y portable para el desarrollo de aplicaciones empresariales en Java.

b) Permitir el uso de componentes reutilizables y basados en estándares para facilitar la interoperabilidad entre diferentes sistemas y tecnologías.

c) Mejorar la productividad y eficiencia del desarrollo de aplicaciones empresariales mediante el uso de APIs y herramientas estándar.

d) Ofrecer un entorno seguro y confiable para la ejecución de aplicaciones empresariales, asegurando la coherencia de la información y la continuidad operativa de los sistemas.

e) Facilitar la creación de aplicaciones escalables y de alto rendimiento, capaces de manejar grandes volúmenes de datos y de usuarios simultáneos.

f) Fomentar la compatibilidad y la reusabilidad de aplicaciones, permitiendo la migración y el despliegue en diferentes servidores de aplicaciones.

g) Promover la colaboración y el intercambio de conocimiento entre desarrolladores y empresas, a través de una comunidad de código abierto que impulsa la evolución y el avance de Jakarta EE. (pág. 72)

1.3.5. Diferencias clave entre Java EE y Jakarta EE.

Es importante, tener en cuenta que una de las principales ejecuciones y sobre todo lo que diferencia de otros sistemas no es más que las mejorías en dichos aplicativos, con los que cada operador se sienta a gusto con el manejo de estos, por ello, el autor (Arrambide, 2010), indica que:

> proyecto Jakarta EE, que no es más que Java EE, en el cual Oracle dona a la Fundación Eclipse todo el trabajo y especificaciones de la última versión de Java EE. Esto conlleva una nueva estructura organizativa y un nuevo logo. (pág. 1)

En este caso, se puede entender que, estos aplicativos son esencialmente diferentes es decir la operabilidad es avanzar mas no estancar al sistema dentro de una empresa o institución, Java EE y Jakarta EE están denominadas como dos plataformas de desarrollo empresarial para aplicaciones Java, y aunque comparten muchas similitudes, también tienen algunas diferencias clave:

a) **Nombre y marca:** La transición de Java EE a Jakarta EE se llevó a cabo para resolver problemas legales relacionados con la marca "Java". Como resultado, Jakarta EE es la marca registrada y el nuevo nombre para esta plataforma.

b) **Origen:** Java EE es la versión anterior de la plataforma, desarrollada originalmente por Sun Microsystems y luego adquirida por Oracle. Jakarta EE es la versión posterior, que surgió después de que Oracle transfiriera las responsabilidades comerciales y la propiedad de Java EE a la Fundación Eclipse.

c) **Modelo de gobernanza:** Java EE estaba gobernado por Oracle y otras compañías miembros del Comité Ejecutivo de Java Community Process (JCP). En cambio, Jakarta EE es una iniciativa de código abierto bajo la Fundación Eclipse, que cuenta con una comunidad diversa de desarrolladores y empresas.

d) **Proceso de desarrollo:** En Java EE, Oracle tenía un control central sobre el proceso de desarrollo y lanzamiento, y todas las especificaciones tenían que pasar por el JCP. En cambio, Jakarta EE sigue un modelo de desarrollo más abierto y colaborativo, con comités técnicos y grupos de trabajo de la Fundación Eclipse.

e) **Licencia:** Java EE fue desarrollado y distribuido bajo la licencia de código binario comunitario (CDDL) y la Licencia Pública de Java (GPL). Jakarta EE, por otro lado, usa y se adhiere a la Licencia Pública de Eclipse (EPL), una licencia de software de código abierto.

f) **Componentes y tecnologías:** Desde el punto de vista funcional, Java EE y Jakarta EE ofrecen características similares y se basan en las mismas tecnologías básicas, como Servlets, JSP, JPA, EJB, etc. Sin embargo, Jakarta EE puede incluir

componentes y tecnologías adicionales desarrolladas por proyectos de la comunidad Eclipse.

Las diferencias clave entre Java EE y Jakarta EE radican en su origen, organización de gobernanza, proceso de desarrollo, licencia y marca. Aunque su finalidad es la misma (el desarrollo de aplicaciones empresariales en Java), Jakarta EE se destaca por ser más abierto y colaborativo, y está respaldado por una comunidad más amplia y diversa.

1.3.6. Estado de la tecnología Java EE y la necesidad de migración

El estado de la tecnología Java EE ha experimentado algunos cambios en los últimos años. Anteriormente, Java EE era mantenida y desarrollada por Oracle Corporation, pero en 2017, Oracle transfirió la responsabilidad de Java EE a la Eclipse Foundation, una organización de desarrollo de software de código abierto, es decir el estado de la tecnología Java EE ha evolucionado hacia Jakarta EE, con un enfoque más abierto y colaborativo. El autor (Thierry , 2020) menciona que el uso de datos en una aplicación web es inevitable y las bases de datos relacionales se utilizan de manera muy habitual. Dado que existen varios SGBDR (sistemas de gestión de bases de datos relacionales), Oracle (originalmente Sun) ha establecido, mediante una especificación, las pautas sobre cómo debe interactuar una aplicación con una base de datos relacional.

Es factible entender por ello que la migración a Jakarta EE puede ofrecer beneficios en términos de flexibilidad y compatibilidad, pero puede requerir tiempo y recursos para llevar a cabo una migración exitosa.

1.4. Problemas y limitaciones identificados en Java EE.

Es importante tener en cuenta que Java EE sigue siendo una opción viable para muchas aplicaciones empresariales, especialmente aquellas que requieren una gran escalabilidad y seguridad. Sin embargo, es fundamental evaluar las necesidades específicas del proyecto es decir los posibles problemas y limitaciones que se identifican en Java EE. Por ello el autor (Castillo C. F., 2021) se afirma que, la complejidad, la rigidez, el tiempo de desarrollo, la sobrecarga de recursos y memoria, el tamaño de la comunidad, su proceso de innovación lenta y la flexibilidad, son factores que limitan la garantía de un buen desarrollo, teniendo en cuenta que sus beneficios son mayores para no limitar el uso de tecnologías como la aplicación de Jakarta EE. (pág. 21)

Sin embargo, es importante destacar que, algunos de estos problemas y limitaciones están siendo abordados en las últimas versiones de Java EE, como Jakarta EE, que sigue evolucionando para ajustarse a las demandas cambiantes de desarrollo de aplicaciones empresariales, sobre todo cuando existen ventajas potenciales que miran la necesidad de migrar a jakarta EE.

1.5. Ventajas potenciales de migrar a Jakarta EE.

Migrar a Jakarta EE puede ofrecer mayores potenciales sobre todo al momento de elegir dichos movimientos migratorios, por ello se requiere destacar estas ventajas, el autor (Castillo C. F.) indica las siguientes:

1.5.1. Mayor compatibilidad y portabilidad

Jakarta EE es una plataforma de desarrollo empresarial de código abierto, lo que implica que las aplicaciones creadas con ella pueden trasladarse sin dificultades a diversos servidores de

aplicaciones que sean compatibles con Jakarta EE. Esto asegura una mayor flexibilidad y compatibilidad con una amplia gama de entornos de implementación.

1.5.2. Mayor productividad y eficiencia

Jakarta EE proporciona una amplia biblioteca de componentes y APIs estándar que pueden ser utilizados para desarrollar aplicaciones empresariales de manera eficiente. Estos componentes predefinidos simplifican el desarrollo y reducen el tiempo necesario para crear aplicaciones empresariales complejas.

1.5.3. Comunidad activa y soporte

Jakarta EE cuenta con una comunidad activa de desarrolladores y usuarios que brindan soporte continuo y recursos útiles para ayudar a los desarrolladores a resolver problemas. Esto asegura que siempre haya una fuente confiable de conocimiento y soluciones disponibles para cualquier problema que pueda surgir durante la migración o el desarrollo de aplicaciones en Jakarta EE.

1.5.4. Soluciones empresariales robustas

Jakarta EE ofrece una amplia gama de especificaciones y tecnologías para abordar los desafíos más comunes en la construcción de aplicaciones empresariales. Esto incluye capacidades como manejo de transacciones, seguridad, gestión de errores y manejo de concurrencia, que son críticos para muchas aplicaciones empresariales.

1.5.5. Actualizaciones y mejoras continuas

Jakarta EE es una plataforma en evolución continua, con actualizaciones y mejoras frecuentes. Esto significa que los desarrolladores pueden beneficiarse de nuevas características y

funcionalidades, así como de correcciones de errores y actualizaciones de seguridad, manteniendo sus aplicaciones al día y constantemente mejoradas. Migrar a Jakarta EE proporciona una plataforma de desarrollo empresarial robusto y escalable con una gran comunidad de soporte, lo que facilita el desarrollo de aplicaciones eficientes y compatibles en entornos empresariales (Castillo C. F., 2021).

1.6.Fundamentos de Jakarta EE

En resumen, los fundamentos de Jakarta EE se centran en la construcción de aplicaciones empresariales modulares, reutilizables y escalables, que pueden desplegarse en diferentes entornos y adaptarse a los cambios de manera flexible, lo cual se puede describir de la siguiente manera:

a) **Modelo de programación basado en contenedores. -** Jakarta EE se basa en un modelo de programación basado en contenedores, en el cual las aplicaciones se ejecutan dentro de un entorno de ejecución llamado contenedor. El contenedor se encarga de gestionar el ciclo de vida de los componentes de la aplicación y proporcionar servicios como seguridad, transacciones y acceso a bases de datos.
b) **Despliegue en múltiples ambientes. -** Jakarta EE está diseñado para desplegarse en diferentes entornos, como servidores de aplicaciones Java EE, nubes públicas y privadas. Esto permite a las aplicaciones Jakarta EE ser portables y ejecutarse en diferentes plataformas sin modificaciones.

Es por ello que el autor (Parra, 2021) indica que: Tolo ello se trata de la principal alternativa al Spring Framework y, aunque en el desarrollo no lo dirigen desde Oracle, sino que

lo lleva la fundación Eclipse, continúa siendo la primera opción para hacer aplicaciones de lado servidor en el lenguaje de programación Java, sobre todo con el auge de MicroProfile. (pág. 32)

1.7.Patrones de diseño y buenas prácticas en Jakarta EE.

Los patrones de diseño o más tomados como las buenas prácticas del sistema en Jakarta EE., es distinguir los beneficios que traen en el rendimiento empresarial

El autor (Brocheto, 2017) menciona que: “Los estándares de codificación y las prácticas recomendadas son directrices que le ayudan a escribir código limpio, coherente y legible, así como a mejorar la calidad, la seguridad y la capacidad de mantenimiento del código” (pág. 12). Los patrones de diseño y las buenas prácticas en Jakarta EE (anteriormente conocido como Java EE) son técnicas y enfoques recomendados para el desarrollo de aplicaciones empresariales utilizando las tecnologías y especificaciones proporcionadas por Jakarta EE.

1.7.1. Algunos patrones de diseño comunes en Jakarta EE incluyen:

a) **Modelo-Vista-Controlador (MVC):** Este patrón divide la lógica de presentación de una aplicación en tres componentes clave: el modelo, la vista y el controlador.

b) **Inyección de dependencias:** este patrón permite la creación y administración de objetos a través de un contenedor de inversión de control, lo que simplifica la gestión de dependencias y la configuración de los componentes.

c) **Data Access Object (DAO):** este patrón proporciona una capa de abstracción entre las operaciones de acceso a datos y las capas superiores de la aplicación, permitiendo una mayor flexibilidad y reutilización del código.

d) **Servicios web RESTful:** este patrón se utiliza para crear servicios web que se adhieren al principio de la arquitectura orientada a servicios (SOA) y utilizan el estilo de arquitectura REST.

e) **Bean de sesión:** este patrón se utiliza para mantener el estado de una sesión de usuario en una aplicación web, lo que permite la persistencia de datos entre las solicitudes del cliente.

1.8.Comparativa de frameworks

La tabla 3 proporciona una comparativa entre tres populares frameworks de desarrollo: Jakarta EE, Django y Angular. Evalúa varios criterios clave, como facilidad de uso, rendimiento, escalabilidad, seguridad e integración con otras tecnologías. Además, analiza aspectos como la compatibilidad con Java EE, la comunidad y soporte, el costo de implementación y la calidad de la documentación y recursos disponibles. Esta comparación ayuda a identificar las fortalezas y debilidades de cada framework, facilitando una decisión informada para proyectos de desarrollo.

Tabla 3. Comparativa entre Jakarta EE, Django y Angular

Criterio	**Jakarta EE**	**Django (Python)**	**Angular (TypeScript)**
Facilidad de uso	Media	Alta	Media
Rendimiento	Alto	Alto	Alto
Compatibilidad con java EE	Completa	No aplica	No aplica
Soporte y comunidad.	Amplia y activa	Amplia y activa	Amplia y activa
Escalabilidad	Excelente	Excelente	Excelente

Seguridad	Robusta	Fuerte	Fuerte
Innovación y actualizaciones	Constantes	Regulares	Constantes
Integración con otras tecnologías	Fácil	Moderada	Moderada
Costo de implementación	Medio	Bajo	Medio
Documentación y recursos	Exhaustiva	Amplia	Amplia

Nota. Elaborado por: Narváez Esteban, basado en: (Fundamentos Generales de Jakarta Enter Prise, 2018).

Según la tabla 3 la comparativa entre Jakarta EE, Django y Angular destaca a Jakarta EE como la mejor opción en varios aspectos clave. Jakarta EE ofrece un rendimiento alto y una escalabilidad excelente, comparable a los otros frameworks, pero se distingue por su completa compatibilidad con Java EE y su robusta seguridad. Además, Jakarta EE proporciona una integración fácil con otras tecnologías y se beneficia de actualizaciones constantes que mantienen la plataforma innovadora. Aunque su facilidad de uso es media y el costo de implementación es mayor en comparación con Django, la exhaustiva documentación y la amplia y activa comunidad de soporte hacen de Jakarta EE una elección superior para proyectos de desarrollo empresariales que requieren fiabilidad y robustez.

En este caso, el autor (Garcia, 2024) afirma que: “Jakarta representa el futuro del desarrollo de aplicaciones empresariales en Java, con un enfoque renovado y más dinámico, dejando atrás a otros elementos que beneficiaban al desarrollo y que marcaron la base de muchas tecnologías empresariales actuales” (pág. 12).

1.9.Frameworks y las bases de datos

En el ámbito del desarrollo de software, los frameworks y las bases de datos desempeñan roles críticos en la estructuración y gestión eficiente de las aplicaciones. Los frameworks brindan una base estructurada para la construcción de aplicaciones, mientras que las bases de datos se encargan del almacenamiento, gestión y recuperación de datos.

El autor (Aguirre, 2021) indica que: “A la hora de tabajar con algún frameworks, es importante respetar las convenciones establecidas, ya que, aunque no se trabaje en equipo es posible que a futuro sea vulnerable para algunos sujetos” (pág. 82).

Por ello, se determina que existe un vínculo que logra difundir sistemas proporcionales que permiten acercar a la realidad, teniendo como fin garantizar una buena estructura dando beneficio al desempeño de una buena base de datos, garantizando una mayor fluidez en el momento de usarla.

Bases de Datos

Según el autor (Gomez, 2012) afirma que: “es un conjunto de datos que pertenecen al mismo contexto, almacenados sistemáticamente para su posterior uso, es una colección de datos estructurados según un modelo que refleje las relaciones y restricciones existentes en el mundo real” (pág. 12). Las bases de datos son elementos fundamentales en los sistemas informáticos, responsables del almacenamiento, gestión y recuperación de datos; además, dicho autor menciona que, existen diversos tipos, como las bases de datos relacionales y no relacionales; donde la bases de datos relacionales utilizan estructuras tabulares para organizar los datos, permitiendo establecer relaciones claras entre diferentes conjuntos de datos.

Dentro de las bases de datos relacionales más utilizadas se encuentran:

1. **MySQL**

    - Conocida por su velocidad y fiabilidad.
    - Frecuentemente usada en aplicaciones web y servicios de backend.

2. **Oracle**

    - Reconocida por su robustez y capacidad de escalamiento.
    - Ofrece características avanzadas como clustering y replicación.

3. **SQL Server**

    - Desarrollada por Microsoft, conocida por su integración con otros productos de Microsoft.
    - Amplio uso en entornos corporativos.

1.9.1. PostgreSQL

Dentro de las bases de datos relacionales, PostgreSQL se destaca por ser una de las más avanzadas y robustas. Reconocido por su capacidad de expansión y cumplimiento con los estándares SQL, PostgreSQL proporciona una variedad de características avanzadas que lo hacen una opción preferida para aplicaciones que demandan alta fiabilidad y rendimiento:

- **Extensibilidad**: Facilita la creación de tipos de datos personalizados y funciones definidas por el usuario, índices personalizados y procedimientos almacenados en múltiples lenguajes.
- **Soporte para JSON**: Proporciona capacidades robustas para el almacenamiento y la manipulación de datos JSON, lo que lo hace adecuado para aplicaciones que necesitan manejar datos semi-estructurados.

- **Transacciones ACID**: Asegura la atomicidad, consistencia, aislamiento y durabilidad de las transacciones (ACID), fundamentales para aplicaciones que requieren alta fiabilidad y recuperación ante fallos.
- **Rendimiento y Escalabilidad**: Soporta operaciones de lectura y escritura concurrentes con alta eficiencia, además de herramientas avanzadas para la optimización de consultas y manejo de grandes volúmenes de datos.

1.10. Metodologías y enfoques para la migración tecnológica.

Al existir un proceso más elevado al momento de usar técnicas que mejoren la migración tecnológica es necesario tener en cuenta que los enfoques de este aplicativo deben contener metodologías importantes para sobresalir en la migración y que sea más precisa al momento de ejecutarla. Los estándares de codificación y las prácticas recomendadas son directrices que le ayudan a escribir código limpio, coherente y legible, así como a mejorar la calidad, la seguridad y la capacidad de mantenimiento del código. Cuando se trabaja con servlets, es importante utilizar nombres significativos y descriptivos para clases, métodos, variables y parámetros. Además, se debe usar la sangría, el espaciado y los comentarios adecuados para estructurar el código. (García, 2022, pág. 42)

Es importante señalar las diversas metodologías y enfoques que se pueden utilizar para una migración tecnológica. Algunas de las más comunes incluyen:

a) **Enfoque de parcheo**: esta metodología implica realizar cambios incrementales al sistema existente, sin interrumpir las operaciones comerciales. Esto puede incluir la actualización de hardware o software, o la implementación de nuevas funcionalidades de forma gradual.

b) **Enfoque de reemplazo completo**: en este enfoque, se reemplaza por completo el sistema existente con una nueva tecnología. Esto puede implicar migrar datos y funcionalidades a una nueva plataforma o realizar una reconstrucción completa del sistema.
c) **Enfoque de modelo híbrido:** este enfoque combina la metodología de parche y de reemplazo completo. Se pueden implementar cambios incrementales mientras se trabaja en la migración completa del sistema.
d) **Enfoque de migración por fases:** esta metodología implica dividir el proceso de migración en fases o etapas. Cada fase se ejecuta de forma secuencial, y se realiza una evaluación y ajuste antes de pasar a la siguiente fase.
e) **Enfoque de migración en paralelo:** el sistema antiguo y el nuevo se ejecutan simultáneamente durante un tiempo determinado, permitiendo que los usuarios se familiaricen con la nueva tecnología antes de eliminarlo por completo

Cada enfoque presenta sus propias ventajas y desafíos, y la selección de la metodología más adecuada dependerá de factores como el alcance de la migración, los recursos disponibles y las necesidades particulares de la organización. (Baena Chavarriaga, 2023, pág. 32)

1.10.1. Evaluación de la arquitectura existente y preparativa para la migración.

La tabla 4 presenta una lista de desafíos comunes en diversos ámbitos y las estrategias de mitigación correspondientes. Aborda problemas como la falta de comunicación efectiva, resistencia al cambio, falta de recursos adecuados, resistencia cultural o política, falta de capacidad o habilidades necesarias, y falta de motivación o compromiso. Cada desafío se acompaña de estrategias prácticas para abordarlos, proporcionando un marco útil para superar estos obstáculos y asegurar el éxito de proyectos y organizaciones.

Algunos desafíos comunes en diferentes ámbitos y las estrategias de mitigación correspondientes son:

Tabla 4. Desafíos y Estrategias de mitigación

Desafío: Falta de comunicación efectiva
Estrategia de Mitigación: Establecer canales de comunicación efectivos y fomentar la comunicación abierta y honesta entre todas las partes interesadas. Fomentar reuniones regulares, emplear herramientas de colaboración en línea y asignar responsabilidades claras para asegurar una comunicación fluida y eficiente.
Desafío: Resistencia al cambio
Estrategia de Mitigación: Proporcionar información clara y educación sobre los beneficios del cambio. Involucrar a todas las partes interesadas en cada una de las etapas iniciales del proceso de cambio y brindar oportunidades para que expresen sus preocupaciones y sugerencias. Establecer un plan de implementación gradual y adaptarse a medida que surgen nuevos desafíos.
Desafío: Falta de recursos adecuados
Estrategia de Mitigación: Realizar una evaluación de necesidades para identificar los recursos necesarios. Buscar oportunidades de colaboración y buscar financiamiento adicional a través de subvenciones, donaciones o alianzas estratégicas. Priorizar y optimizar el uso de los recursos existentes.
Desafío: Resistencia cultural o política
Estrategia de Mitigación: Comprender y respetar las diferencias culturales y políticas y buscar formas de integrarlas en el proceso. Promover la participación y la colaboración de todas las partes interesadas para abordar las preocupaciones y encontrar soluciones que sean aceptables para todos.

Desafío: Falta de capacidad o habilidades necesarias
Estrategia de Mitigación: Identificar las brechas de capacidad y ofrecer capacitación y desarrollo profesional para cerrar esas brechas. Establecer asociaciones con organizaciones o expertos con experiencia en el área para recibir asistencia técnica o compartir conocimientos especializados.
Desafío: Falta de motivación o compromiso
Estrategia de Mitigación: Crear un sentido de propósito y visión compartida, destacando los beneficios y recompensas tanto a nivel individual como organizativo. Fomentar la participación, reconocer y recompensar los logros y contribuciones individuales y de equipo.

Nota. Elaborado por: Narváez Esteban, basado en: (Desafíos comunes a la mitigación, 2021).

Estas son solo algunas estrategias de mitigación para desafíos comunes. Cada situación puede requerir enfoques y soluciones específicas, pero estas estrategias pueden servir como punto de partida para abordar los desafíos y minimizar su impacto.

1. **Herramientas y recursos para la migración**

Durante la migración del sistema SIGEPAA a Jakarta EE, se pueden utilizar diversas herramientas y recursos para facilitar y agilizar el proceso. A continuación, se mencionan algunas de las herramientas y recursos más importantes. **Eclipse IDE con el complemento Jakarta EE**., Siendo un entorno de desarrollo integrado ampliamente utilizado y el complemento Jakarta EE proporciona todas las herramientas necesarias para trabajar con las tecnologías de Jakarta EE. Es fundamental para desarrollar y migrar aplicaciones en esta plataforma.

2. **Servidores de Aplicaciones Compatibles con Jakarta EE**

Utilizar un servidor de aplicaciones que sea totalmente compatible con Jakarta EE, como WildFly, Payara o TomEE, es esencial para implementar y probar la aplicación migrada. Proporcionan un entorno de tiempo de ejecución compatible con Jakarta EE.

3. **Documentación y Tutoriales de Jakarta EE**

La documentación oficial y los tutoriales de Jakarta EE son recursos fundamentales para los desarrolladores que migran la aplicación. Proporcionan guía paso a paso y ejemplos prácticos para comprender y utilizar las tecnologías de Jakarta EE de manera efectiva.

4. **Herramientas automatizadas de migración de código.**

Estos procesos son necesarios, permiten tener un mejor funcionamiento de transformación automatizada al momento de requerir un proyecto, el autor (Brocheto, 2017) menciona que, es un enfoque que elimina algunas de las barreras existentes cuando es necesario unificar código desarrollado en diferentes lenguajes. Así mismo, cuando es necesario migrar código de un lenguaje a otro, se facilita el uso continuo de software legado, evitando así los inconvenientes asociados con su obsolescencia. (pág. 32). Por ello se explican a continuación dos herramientas que son fundamentales para el proceso de aceleración en la migración.

a) **Apache NetBeans Java EE Migration Tool:** Esta herramienta es esencial ya que proporciona una forma automatizada de convertir proyectos Java EE a proyectos Jakarta EE. Facilita la actualización de la estructura del proyecto y las dependencias, lo cual es fundamental para la migración exitosa del sistema.
b) **JBoss EAP Migration Toolkit:** Este kit de herramientas proporcionado por Red Hat es otra herramienta importante, ya que puede ayudar en la migración de aplicaciones Java EE a Jakarta EE, ofreciendo una guía paso a paso y soporte para

la conversión de código. Al ser proporcionado por uno de los líderes en el desarrollo de servidores Jakarta EE, el toolkit es una valiosa fuente de información y asistencia en el proceso de migración.

Estas dos herramientas automatizadas de migración de código son fundamentales para acelerar el proceso de migración, identificar posibles incompatibilidades y facilitar la transición de la aplicación del sistema SIGEPAA a la plataforma Jakarta EE.

1.10.2. Frameworks y librerías compatibles con Jakarta EE.

1. **Spring Framework:** Spring es uno de los frameworks más populares en el mundo Java y es compatible con Jakarta EE. Ofrece una amplia variedad de funcionalidades diseñadas específicamente para el desarrollo de aplicaciones empresariales, como la gestión de dependencias, la inyección de dependencias, el control transaccional y la seguridad.
2. **Hibernate:** es una herramienta de mapeo objeto-relacional (ORM) que facilita la conversión de objetos Java a tablas de bases de datos relacionales. Es una opción ampliamente utilizada para el acceso a bases de datos en aplicaciones empresariales y es completamente compatible con Jakarta EE.
3. **EclipseLink:** EclipseLink es otro ORM que es parte integral de Jakarta EE. Proporciona capacidades avanzadas de mapeo objeto-relacional y es compatible con diversas bases de datos y proveedores de persistencia.
4. **PrimeFaces:** PrimeFaces es una biblioteca de componentes de interfaz de usuario (UI) que facilita la creación de interfaces de usuario ricas y atractivas en

aplicaciones web. Es compatible con Jakarta EE y se integra bien con otros frameworks como JSF (JavaServer Faces).

5. **Apache MyFaces:** MyFaces es una implementación de la especificación JSF y es compatible con Jakarta EE. Permite el desarrollo de aplicaciones web basadas en componentes JavaServer Faces.
6. **Apache CXF:** CXF es un framework para el desarrollo de servicios web basados en Java. Es compatible con Jakarta EE y facilita la creación de servicios web RESTful y SOAP.
7. **Apache Shiro:** Shiro es un framework de seguridad que proporciona una solución fácil de usar para autenticación, autorización, criptografía y manejo de sesiones. Es compatible con Jakarta EE y es ampliamente utilizado para agregar capas de seguridad a aplicaciones empresariales.
8. **Apache Commons:** Apache Commons es una colección de librerías que proporcionan funcionalidades reutilizables y comunes en el desarrollo de aplicaciones Java. Muchas de estas librerías son compatibles con Jakarta EE y facilitan tareas como manipulación de archivos, operaciones matemáticas, manejo de colecciones, entre otras.

1.11. Consideraciones de rendimiento y seguridad en Jakarta EE

1.11.1. Optimización del rendimiento en aplicaciones Jakarta EE.

- **Optimización de Consultas:** Revisar y optimizar las consultas a la base de datos utilizando índices adecuados y técnicas de optimización para mejorar la velocidad de respuesta.

- **Gestión de Transacciones:** Optimizar el manejo de transacciones para reducir el tiempo de bloqueo y mejorar la eficiencia de las operaciones de lectura y escritura.
- **Uso Eficiente de Recursos:** Monitorear y optimizar el uso de recursos como CPU, memoria y conexiones a la base de datos para evitar cuellos de botella y sobrecargas.
- **Pruebas de Rendimiento:** Realizar pruebas de rendimiento periódicas para medir y comparar el rendimiento del sistema en diferentes escenarios y cargas de trabajo.

1.11.2. Consideraciones de seguridad específicas de Jakarta EE.

- **Gestión de Autenticación y Autorización:** Establecer un mecanismo robusto de autenticación y autorización que asegure que solo los usuarios con los permisos adecuados puedan acceder al sistema y utilizar las funcionalidades correspondientes.
- **Prevención de Ataques de Inyección:** Validar y sanitizar rigurosamente todas las entradas de usuario para proteger el sistema contra ataques de inyección, como SQL injection y Cross-Site Scripting (XSS), que representan amenazas significativas para la seguridad.
- **Configuración Segura:** Revisar y configurar de manera segura el servidor Jakarta EE y la aplicación para evitar vulnerabilidades y asegurar que la información sensible esté protegida.
- **Cifrado de Datos:** Emplear técnicas de cifrado avanzadas para proteger los datos confidenciales tanto en reposo como en tránsito, asegurando que la información sensible permanezca segura y accesible solo para usuarios autorizados.

1.12. Plan de implementación para la migración en SIGEPAA

1.12.1. Etapas y cronograma de la migración.

La tabla 5 detalla el plan de implementación para la migración en SIGEPAA a Jakarta EE, dividiéndolo en etapas con sus respectivas actividades y cronogramas. Este plan asegura una transición estructurada y minimiza riesgos, abordando cada aspecto de la migración desde la preparación hasta el monitoreo y soporte post-implementación.

Tabla 5. Plan de implementación para migración

Etapa 1: Preparación y Planificación
• Definir los objetivos y alcance de la migración. • Formar el equipo de migración y asignar roles y responsabilidades. • Realizar un análisis detallado del sistema actual y sus componentes. • Identificar los requisitos de hardware, software y recursos necesarios para la migración. • Semana 1-2.
Etapa 2: Evaluación de la Infraestructura
• Verificar que la infraestructura actual cumple con los requisitos de Jakarta EE. • Realizar pruebas de compatibilidad de las aplicaciones actuales con la nueva plataforma. • Identificar posibles ajustes y mejoras necesarios en la infraestructura. • Semana 3-4.
Etapa 3: Preparación del Ambiente de Desarrollo y Pruebas
• Instalar y configurar el ambiente de desarrollo con Jakarta EE. • Realizar pruebas de concepto y prototipos para familiarizar al equipo con la nueva plataforma.

- Desarrollar y ejecutar casos de prueba en el ambiente de desarrollo.
- Semana 5-6.

Etapa 4: Desarrollo e Implementación

- Comenzar el desarrollo del sistema actualizado en Jakarta EE.
- Realizar pruebas unitarias y de integración de los módulos desarrollados.
- Implementar los cambios en un ambiente de prueba o preproducción para realizar pruebas más exhaustivas.
- Semana 7-10.

Etapa 5: Pruebas y Ajustes

- Realizar pruebas exhaustivas en el ambiente de prueba o preproducción.
- Identificar y corregir posibles errores y problemas.
- Obtener retroalimentación del equipo de usuarios y realizar ajustes según sus comentarios.
- Semana 11-14.

Etapa 6: Capacitación

- Capacitar al personal interno en el uso de la nueva plataforma Jakarta EE.
- Proporcionar documentación y recursos de capacitación.
- Semana 15-16.

Etapa 7: Implementación en Producción

- Realizar la migración final del sistema a Jakarta EE en el ambiente de producción.
- Realizar pruebas post-implementación y verificar la funcionalidad completa del sistema.
- Semana 17-18.

Etapa 8: Monitoreo y Soporte

- Realizar un monitoreo continuo del rendimiento y la estabilidad del sistema en el entorno de producción, identificando proactivamente posibles problemas y tomando medidas correctivas para garantizar la disponibilidad y eficiencia del sistema.
- Proporcionar soporte técnico para abordar cualquier problema que surja.
- Semana 19 en adelante.

Nota: Elaborado por: Narváez Esteban, basado en: (Garantías de Jakarta Enter Prise, 2021).

1.13. Metodologías de desarrollo de software

Las metodologías de desarrollo de software son enfoques organizados y planificados que guían la creación y gestión de proyectos de software. Estas metodologías establecen un marco estructurado que define procesos, roles, actividades y herramientas, con el fin de garantizar que los proyectos se ejecuten de forma eficiente, cumpliendo con los requisitos establecidos y optimizando los recursos disponibles.

Mirándolo del punto de vista el autor (García, 2022) indica que: “En informática se define la metodología como «un proceso para producir software de forma organizada, empleando una colección de técnicas y convenciones de notación predefinidas.»” (pág. 32).

Por ello, las metodologías ayudan a organizar el trabajo, facilitan la comunicación entre los miembros del equipo y proporcionan un medio para controlar el progreso del proyecto. Algunas metodologías comunes incluyen la metodología en cascada, el desarrollo ágil, Scrum, Kanban, Lean, entre otras. Cada metodología de desarrollo posee características distintivas, junto con ventajas y desventajas particulares. La clave radica en elegir la más adecuada según las necesidades particulares del proyecto, así como las capacidades y dinámicas del equipo de desarrollo.

1.13.1. Tipos de metodología

En cada proceso para implantar un nuevo modelo de aplicativo es esencial reconocer que cada metodología integrada en un sistema operativo es esencial para su funcionamiento, considerando que no son solo varios procesos metodológicos para crear dicho aplicativo.

El auto (Velez, 2022) menciona que:

> Existen distintas metodologías de desarrollo de software. Para desarrolla este TFG nos hemos basado en una metodología clásica basada en el método incremental, ya que se partía de una base de conocimiento del proceso de negocio previa pero no así de la tecnología a usar. (pág. 32)

En este caso dentro de la presente investigaciones se reconocer algunas metodologías que garantiza la viabilidad de los aplicativos; además que, Las metodologías tradicionales de desarrollo de software se fundamentan en procesos estructurados y predefinidos, donde cada fase se planifica y se ejecuta de manera secuencial para gestionar proyectos, entre ella están:

a) **Metodologías Tradicionales:** En estas metodologías, el proyecto se organiza en etapas claramente definidas, como análisis, diseño, implementación, pruebas y mantenimiento, y cada una de estas fases se completa de manera secuencial antes de avanzar a la siguiente. El modelo en cascada es un ejemplo de una metodología tradicional, donde el progreso fluye en una dirección lineal hacia abajo, similar a una cascada.
b) **Metodologías Ágile:** Las metodologías ágiles son enfoques flexibles y adaptables para el desarrollo de software, que se centran en la entrega continua e incremental. En lugar de seguir una estructura rígida, estas metodologías promueven la

colaboración constante entre los miembros del equipo, el intercambio fluido de información con los clientes y la capacidad de ajustarse rápidamente a los cambios. A través de ciclos cortos de desarrollo llamados sprints, los métodos ágiles permiten entregar software funcional de forma frecuente y eficiente, donde se van desarrollando y refinando las características del producto de manera continua. Scrum, Kanban y Extreme Programming (XP) son ejemplos populares de metodologías ágiles.

1.13.2. Cuadro comparativo para la selección de Metodología de desarrollo

Cada metodología de desarrollo de software posee características particulares que la hacen más adecuada para ciertos tipos de proyectos y contextos. Es importante evaluar las necesidades y características específicas del proyecto antes de elegir la metodología más adecuada.

La Tabla 6 presenta una comparativa entre Jakarta EE, Django y Angular en varios aspectos clave relacionados con sus metodologías.

Tabla 6. Comparativa entre Jakarta EE, Django y Angular

Aspectos	**Metodología en cascada**	**Metodología Ágil**
Enfoque	Secuencial	Interactivo e incremental
Flexibilidad	Baja	Alta
Adaptabilidad	Baja	Alta
Requisitos	Definidos al principio	Pueden cambiar
Entrega de valor	Al final del proyecto	en cada iteración
Riesgos	Identificados al principio	Abordados continuamente

Comunicación	Formal	Informal
Feedback	Al final del proyecto	Constante
Ciclos de desarrollo	Lineales	Interactivos

Nota: Elaborado por: Narváez Esteban, basado en: (Fundamentos Generales de Jakarta Enter Prise, 2018).

A continuación, se presenta un análisis detallado de los aspectos propuestos en la tabla 6.

Enfoque: La Metodología en Cascada sigue un enfoque secuencial y lineal, donde cada fase del desarrollo se completa antes de pasar a la siguiente. Agile, en cambio, adopta un enfoque iterativo e incremental, dividiendo el trabajo en ciclos o iteraciones.

Flexibilidad y Adaptabilidad: La Metodología en Cascada tiene poca flexibilidad y adaptabilidad, mientras que Agile es más flexible y adaptable, permitiendo ajustes a medida que avanza el proyecto.

Requisitos: En la Metodología en Cascada, los requisitos se definen al principio del proyecto y se mantienen relativamente estables. En Agile, los requisitos pueden evolucionar a medida que avanza el proyecto.

Entrega de valor: En la Metodología en Cascada, el valor se entrega al final del proyecto, mientras que en Agiles, el valor se entrega en cada iteración.

Gestión de riesgos: La Metodología en Cascada aborda los riesgos al principio del proyecto, mientras que Agile integra la gestión de riesgos de manera continua.

Comunicación y Feedback: La Metodología en Cascada suele tener una comunicación más formal y feedback menos frecuente, mientras que Agile fomenta una comunicación más informal y feedback constante.

Ciclos de desarrollo: Mientras que la Metodología en Cascada sigue ciclos de desarrollo lineales, Agile utiliza ciclos iterativos.

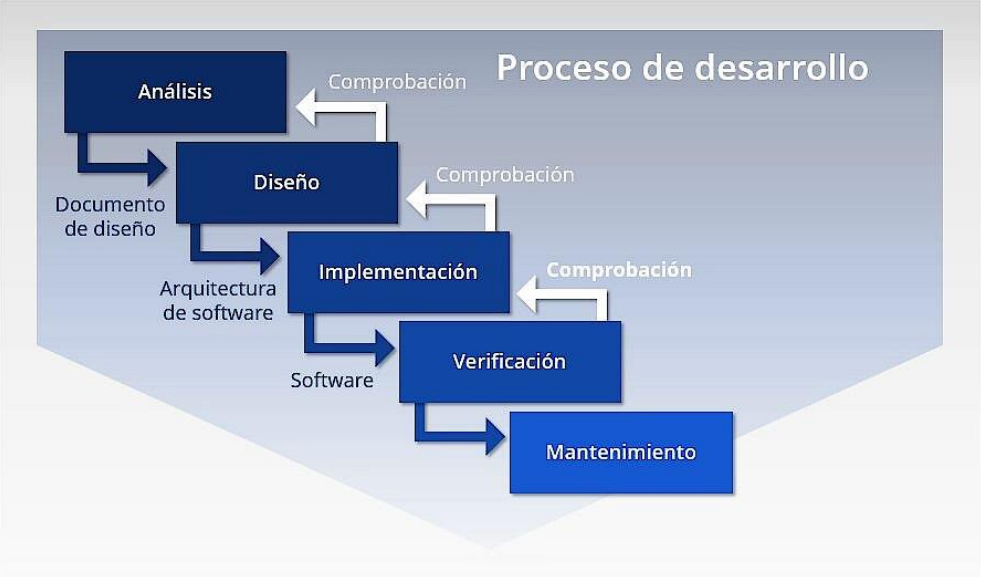
En criterio se menciona que tanto la Metodología en Cascada como Agile tienen sus propias características y se adaptan mejor a diferentes tipos de proyectos y entornos organizacionales. La selección entre estas metodologías depende de las necesidades específicas del proyecto, así como de las preferencias y capacidades del equipo de desarrollo.

1.14. Descripción de la Metodología en cascada una Visión Tradicional del Desarrollo de Software

La Metodología en Cascada, también conocida como el modelo de desarrollo secuencial, representa una de las formas más tradicionales de gestionar proyectos de software. En este enfoque, el proceso de desarrollo sigue un enfoque lineal y secuencial, en el que cada fase debe concluir antes de avanzar a la siguiente. Este modelo se caracteriza por una planificación detallada al principio del proyecto y una implementación incremental, lo que significa que los cambios en etapas posteriores pueden ser costosos y difíciles de manejar.

El modelo en cascada de cinco niveles, desarrollado por Winston W. Royce, desglosa el proceso de desarrollo en las siguientes fases: análisis, diseño, implementación, verificación y mantenimiento. La figura 4 muestra una de las expansiones del modelo propuesto por Royce, que sugiere la verificación de los resultados de cada fase, asegurando que cumplan con los requisitos y especificaciones establecidas en la fase previa.

Figura 4 Proceso de desarrollo



Nota. Por Winston W. Royce, 1970.

1.14.1. Fases de la metodología en cascada

Fase 1: Análisis

En esta etapa inicial, se lleva a cabo un análisis exhaustivo de los requisitos del proyecto. Se recopilan e identifican los detalles y las necesidades del sistema. Es crucial establecer una comunicación clara con el cliente para comprender completamente sus expectativas y objetivos. Además, se planifica el alcance del proyecto, se define el cronograma y se asignan recursos adecuados. La documentación de los requisitos es una parte fundamental de esta fase, esto proporciona una base sólida que facilita el desarrollo continuo del proyecto, asegurando que cada fase cumpla con los objetivos establecidos antes de avanzar.

Fase 2: Diseño

En la fase de diseño, se elabora una arquitectura detallada del sistema, fundamentada en los requisitos obtenidos en la fase anterior. Se definen las estructuras de datos, los algoritmos y

las interfaces de usuario necesarias para la implementación del software. Los diseñadores colaboran estrechamente con los clientes y desarrolladores, asegurando que el diseño sea claro, completo y fácilmente comprensible. Además, se crea un plan exhaustivo de pruebas para garantizar la funcionalidad del sistema una vez que haya sido implementado.

Fase 3: Implementación

En esta etapa, se traduce el diseño del sistema en código ejecutable. Los programadores desarrollan el software utilizando lenguajes de programación específicos y herramientas de desarrollo. Es importante seguir las pautas y estándares establecidos durante la fase de diseño para garantizar la coherencia y la calidad del código. Además, se realizan pruebas unitarias a medida que se desarrolla el software para identificar y corregir errores de manera oportuna.

Fase 4: Verificación

Una vez completa la implementación, se hacen pruebas exhaustivas para validar la funcionalidad y la calidad del software. Se realizan pruebas de unidad, integración y sistema para identificar cualquier defecto o problema en el sistema. Los probadores trabajan en estrecha colaboración con los desarrolladores para resolver cualquier problema encontrado y garantizar que el software cumpla con los requisitos del cliente.

Fase 5: Mantenimiento

La fase de mantenimiento comienza una vez que el software está en funcionamiento. Se realizan actualizaciones, correcciones de errores y mejoras de manera continua para garantizar que el sistema siga siendo efectivo y cumpla con las necesidades del cliente a lo largo del tiempo. El equipo de desarrollo trabaja en estrecha colaboración con el cliente para abordar cualquier problema que surja y garantizar la satisfacción continua del usuario.

1.14.2. Beneficios que brinda la metodóloga de desarrollo en cascada

- **Estructura clara y secuencial**

La metodología en cascada sigue un enfoque paso a paso, lo que facilita la comprensión y planificación del proyecto. Cada fase se completa antes de pasar a la siguiente, lo que brinda una estructura clara y ordenada para el desarrollo del software.

- **Planificación temprana**

La fase inicial de análisis y planificación permite establecer requisitos y objetivos claros del proyecto desde el principio. Esto ayuda a reducir los riesgos y los cambios de alcance durante las etapas posteriores del desarrollo.

- **Seguimiento del progreso**

Como cada fase tiene sus entregables específicos, la metodología en cascada facilita el seguimiento del progreso del proyecto. Esto permite a los equipos de desarrollo identificar y resolver problemas de manera rápida y eficiente, minimizando posibles retrasos en el proceso.

- **Enfoque orientado a la calidad**

Cada fase incluye actividades de revisión y prueba, lo que garantiza la calidad del software en cada etapa del desarrollo. Esto ayuda a detectar y corregir errores temprano, lo que reduce los costos y el tiempo asociados con la corrección de problemas en etapas avanzadas del proyecto.

- **Documentación exhaustiva**

La metodología en cascada promueve la elaboración de documentación exhaustiva en cada fase del proyecto, lo que genera un registro integral de los requisitos, el diseño, la implementación y las pruebas del software. Esto facilita el mantenimiento y la escalabilidad a largo plazo.

1.15. Norma ISO/IEC 25000-25010

La Norma ISO/IEC 25000, conocida como la familia de normas SQuaRE (Software Quality Requirements and Evaluation), es un conjunto de estándares internacionales y guías que se centran en la calidad del software. Esta familia de normas ofrece un marco y herramientas para evaluar y mejorar la calidad de los productos de software a lo largo de todo su ciclo de vida. El autor (Lozano Peralta, 2020) indica que:

> Dentro de la familia ISO/IEC 25000, la Norma ISO/IEC 25010 juega un papel crucial. Específicamente, la ISO/IEC 25010 se centra en los "Modelos de Calidad de Sistemas y Software". Esta norma sustituyó a la anterior ISO/IEC 9126, ampliando y refinando los conceptos de calidad del software. (pág. 48)

El modelo de calidad es fundamental para evaluar la calidad de un producto software, definiendo las características esenciales que se considerarán en su evaluación. Este modelo no solo identifica, sino que también categoriza la calidad que se desglosa en características y subcaracterísticas específicas, tales como funcionalidad, rendimiento, seguridad y mantenibilidad. El modelo de calidad del producto, según lo definido por la ISO/IEC 25010, se compone de ocho características clave, que se detallan en la siguiente figura:

Figura 5 Calidad de producto software

a) **Funcionalidad:** Capacidad del software para proporcionar funciones que satisfagan tanto necesidades explícitas como implícitas.

b) **Eficiencia de desempeño:** Proporción entre el rendimiento del sistema y la cantidad de recursos utilizados.

c) **Compatibilidad:** La capacidad del software de coexistir y colaborar con otros productos.

d) **Usabilidad:** Facilidad de uso y aprendizaje del software.

e) **Fiabilidad:** Capacidad para mantener un nivel de rendimiento bajo condiciones específicas y durante un período definido.

f) **Seguridad:** Capacidad para proteger la información y los datos, permitiendo el acceso solo a personas autorizadas.

g) **Mantenibilidad:** Facilidad con la que se pueden realizar modificaciones y correcciones en el software.

h) **Portabilidad:** La facilidad con la que el software puede ser trasladado de un entorno a otro.

CAPÍTULO II

2. Importancia de las Migraciones en el Mantenimiento y Evolución de Sistemas de Software

Las migraciones son cruciales en el ámbito técnico para mantener la viabilidad y seguridad de los sistemas de software. Estas actualizaciones permiten adaptarse a nuevas tecnologías, aprovechar mejoras de rendimiento y parches de seguridad, y garantizar la compatibilidad con los sistemas operativos, navegadores web y otros componentes de infraestructura. Además, las migraciones posibilitan la adopción de nuevas funcionalidades, optimizando así los recursos de hardware y software y mejorando la eficiencia operativa. En conjunto, estas acciones aseguran que los sistemas estén actualizados, seguros y preparados para enfrentar los desafíos tecnológicos en un entorno digital en constante cambio.

Beneficios que brinda la metodología de desarrollo en cascada

De la teoría presentada en el capítulo anterior, hemos llevado a cabo una exhaustiva investigación para seleccionar la metodología que mejor se adapte al desarrollo de proyecto para la migración de Java EE a Jakarta EE del módulo de secretaria en el sistema SIGEPAA, se tomó en cuenta la investigación realizada sobre las metodologías Tradicionales y Agiles además la sugerencia del experto PhD. Irving Reascos. Docente de la Universidad Técnica del Norte para optar por la selección de la metodología Tradicional dado que el proyecto implica una migración técnica de una plataforma a otra sin cambios significativos en los requisitos funcionales, una metodología más tradicional como la metodología en cascada es adecuada.

La metodología en cascada se centra en fases secuenciales y bien definidas, lo que puede ser beneficioso cuando el alcance y los requisitos del proyecto están claramente definidos desde el inicio, y no se anticipan cambios significativos.

Esta metodología facilita una planificación detallada y una implementación gradual, lo cual puede resultar eficiente para un proyecto de migración técnica. La metodología en cascada puede ser menos flexible en cuanto a cambios durante el proceso, por lo que es crucial asegurarse de que los requisitos estén completamente comprendidos y definidos antes de comenzar la migración.

Sin duda alguna, al utilizar un segmento cognitivo que beneficie un proceso de migración tomado como estrategia para la actualización de una arquitectura en un sistema aplicable, donde se aprovechen las ventajas que dicho este puede proporcionar; es decir, el proceso de migración del módulo de secretaria el cual se encuentra usando el framework Java EE 8 del aplicativo "SIGEPAA" al framework de Jakarta EE, garantiza las proporciones de Jakarta EE en términos de modularidad, eficiencia y compatibilidad con estándares modernos.

Inicialmente se cuenta con una planificación de la Migración, en donde el autor (Baena Chavarriaga, 2023) menciona que:

El estilo arquitectónico influye en que tan fácil o difícil es aplicar las evoluciones o correcciones, en donde el uso de buenas prácticas de programación, uso de patrones de diseño, etc., dado que es posible encontrar desarrollos altamente acoplados o con baja cohesión que hagan difícil o imposible aplicar estos cambios mencionados, cuando se tienen un sistema bastante avanzado en su desarrollo. (pág. 9).

Por ello, teniendo en referencia que un proceso de planificación de la migración genera buenos hábitos en donde su principal función es encontrar desarrollos que se acoplan a las necesidades de la empresa, como aquellas que mejorarían al sistema de esta; por ello, no se deja

atrás las evoluciones diarias de un proceso que facilitara tener mayor eficiencia para la sociedad y esto al momento de aplicar un sistema reforzado en una empresa.

Para mejorar un sistema, se tiene claro que es necesaria la planificación; así mismo, se dimensiona que un vínculo con la asignación de recursos como de los roles en los procesos de actualización es necesario.

Diseño del Proceso de Migración según las Fases de la metodología en cascada

2.1. Fase 1: Análisis y Planificación

2.1.1 Evaluación de la Arquitectura del Sistema SIGEPAA en Java EE

La *tabla 7* presenta una evaluación de la arquitectura del sistema SIGEPAA en Java EE, indicando que el sistema actualmente utiliza la versión 8 de Java. Este análisis proporciona una base sólida para planificar la migración a Jakarta EE, permitiendo identificar áreas clave que requieren atención durante el proceso de actualización.

Tabla 7. Arquitectura del Sistema SIGEPAA en Java EE

Punto	Versión en Java EE 8	Descripción
Enterprise Archive (EAR)	**8.0**	La versión del módulo de aplicación empaquetado (EAR) utilizada en el sistema SIGEPAA es la versión 8. El archivo EAR es un estándar de empaquetado utilizado en el desarrollo de aplicaciones empresariales Java EE. Contiene uno o más archivos WAR (Web ARchive), que contienen componentes web, así como archivos JAR (Java ARchive) que contienen componentes empresariales como EJBs (Enterprise JavaBeans) y bibliotecas de clases. La elección de la versión 8 del formato EAR indica que el sistema SIGEPAA sigue las especificaciones y estándares establecidos para el empaquetado de aplicaciones empresariales en el entorno Java EE.
JDBC (Java Database Connectivity)	**4.3**	La aplicación SIGEPAA utiliza JDBC (Java Database Connectivity) 4.3 para establecer conexiones con la base de datos y ejecutar consultas SQL. Esta funcionalidad permite que la aplicación acceda y manipule datos almacenados en la base de datos de manera eficiente. Por ejemplo, se utilizan conexiones JDBC para recuperar información de los usuarios, gestionar registros de transacciones y actualizar datos en diferentes módulos del sistema.

JPA (Java Persistence API)	**2.1**	La aplicación SIGEPAA hace uso de JPA (Java Persistence API) 2.1 permitiendo mapear objetos Java a tablas de bases de datos y viceversa, facilitando la interacción con la base de datos y viceversa. Esta tecnología facilita la interacción con la base de datos y simplifica el acceso y la manipulación de datos dentro de la aplicación. Por ejemplo, las entidades JPA se utilizan para representar entidades de negocio como usuarios, productos y pedidos, y se gestionan las operaciones de creación, lectura, actualización y eliminación de estos datos mediante consultas JPA.
Servlets	**3.1**	Los Servlets 3.1 son utilizados en la aplicación SIGEPAA para gestionar solicitudes HTTP y generar respuestas dinámicas. Por ejemplo, los Servlets procesan formularios de registro de usuarios, autenticar credenciales de inicio de sesión y mostrar datos dinámicos en las páginas web. Además, los Servlets interactúan con otros componentes del sistema para ejecutar lógica de negocio específica y gestionar la navegación del usuario dentro de la aplicación.
CDI (Contexts and Dependency Injection)	**1.2**	La aplicación SIGEPAA utiliza CDI (Contexts and Dependency Injection) 1.2 para administrar la creación y la inyección de dependencias entre los diferentes componentes de la aplicación. Por ejemplo, CDI se utiliza para inyectar instancias de beans de servicio en los controladores de servlets y las clases de lógica de negocio, facilitando la comunicación y colaboración entre los diferentes componentes del sistema.

JAX-RS (Java API for RESTful Web Services)	**2.0**	JAX-RS 2.0 se emplea en SIGEPAA para desarrollar servicios web RESTful que permiten la interacción con recursos específicos de la aplicación a través de la web. Por ejemplo, se utilizan servicios JAX-RS para consultar y manipular datos de usuarios, productos y pedidos mediante operaciones HTTP como GET, POST, PUT y DELETE. Estos servicios web proporcionan una interfaz programática para acceder y gestionar datos de forma eficiente y segura.
JSF (JavaServer Faces)	**2.2**	JSF 2.2 se utiliza en SIGEPAA para construir la interfaz de usuario basada en componentes. Por ejemplo, las páginas web de la aplicación se crean mediante la definición de componentes JSF como formularios, tablas y botones, que se vinculan a la lógica de negocio y a los datos subyacentes. Además, JSF ofrece funcionalidades como manejo de eventos, validación de datos y administración de estados para optimizar la experiencia del usuario y mejorar la interactividad de la aplicación.
EJB Module (Enterprise JavaBeans)	**3.2**	El sistema SIGEPAA se basa en el módulo EJB (Enterprise JavaBeans) versión 3.2. Los EJB son componentes de servidor utilizados en el entorno Java EE para encapsular la lógica empresarial de una aplicación.

JDK(Kit de Desarrollo de Java)	**1.8.0_241**	El sistema SIGEPAA utiliza JDK 1.8.0_241 como entorno de desarrollo para la implementación de sus componentes y funcionalidades. El JDK 1.8.0_241, o Kit de Desarrollo de Java versión 8 update 241, ofrece las herramientas necesarias para compilar, depurar y ejecutar aplicaciones Java. Esta versión del JDK incluye el compilador de Java (javac), las bibliotecas de clases de Java, el depurador de Java (jdb) y otras utilidades esenciales para el desarrollo de software en Java. La elección del JDK 1.8.0_241 como plataforma de desarrollo garantiza la compatibilidad con las especificaciones y tecnologías utilizadas en el sistema SIGEPAA, permitiendo una implementación coherente y eficiente de sus características.
Frameworks y Bibliotecas Externas	**Primefaces 7**	SIGEPAA hace uso de varios frameworks y bibliotecas externas para mejorar su funcionalidad y productividad. Por ejemplo, se emplea PrimeFaces para enriquecer la interfaz de usuario con componentes visuales avanzados como gráficos, calendarios y menús desplegables. También se utiliza Hibernate como implementación de JPA para simplificar el acceso a la base de datos y manejar la persistencia de datos de manera eficiente y transparente.
WildFly	**18**	El sistema SIGEPAA utiliza WildFly 18 como su servidor de aplicaciones principal. WildFly es una plataforma de servidor de código abierto desarrollada por Red Hat y se basa en el proyecto JBoss Application Server.

Nota: Adaptado de: Desarrollo de aplicaciones web con Jakarta EE (Cesar Castillo, 2022)

2.1.2 Patrón de Diseño Modelo-Vista-Controlador (MVC)

El sistema SIGEPAA sigue el patrón de diseño Modelo-Vista-Controlador (MVC), un enfoque arquitectónico comúnmente utilizado para organizar y estructurar aplicaciones web. Este patrón se compone de tres componentes principales:

- **Modelo (Model)**: Representa tanto los datos como la lógica de negocio de la aplicación. En el caso del SIGEPAA, el modelo abarca la gestión de datos y la lógica subyacente que define el comportamiento del sistema, incluyendo la interacción con la base de datos y el procesamiento de la información.
- **Vista (View)**: Es la interfaz de usuario que presenta la información al usuario final. En el contexto del SIGEPAA, la vista incluye las páginas web, formularios y otros elementos de la interfaz de usuario que permiten a los usuarios interactuar con el sistema y visualizar los datos de manera significativa.
- **Controlador (Controller)**: Actúa como un intermediario entre el modelo y la vista, gestionando las solicitudes del usuario y coordinando la interacción entre los componentes del sistema. En el SIGEPAA, el controlador se encarga de procesar las solicitudes del usuario, actualizar el modelo según sea necesario y seleccionar la vista apropiada para mostrar los resultados al usuario.

Al adoptar el patrón MVC, el sistema SIGEPAA se beneficia de una estructura clara y modular que facilita el mantenimiento, la escalabilidad y la reutilización del código. Además, permite una separación efectiva de las preocupaciones, lo que mejora la claridad y la coherencia del diseño de la aplicación. En ello se aplica la identificación de Componentes Críticos del Sistema**,** centrándose en identificar los componentes críticos del sistema SIGEPAA y comprender su importancia y dependencias.

En la tabla 8 se enumera los componentes críticos del sistema SIGEPAA, proporcionando una visión clara de los elementos esenciales que deben ser considerados y gestionados cuidadosamente durante el proceso de migración a Jakarta EE. Estos componentes son fundamentales para el correcto funcionamiento y rendimiento del sistema.

Tabla 8. Componentes críticos del sistema

Punto Critico	Versión en Java EE 8	Versión en Jakarta EE 10	Descripción Java EE 8	Descripción Jakarta EE 10	Crítico al Actualizar	Razón de Criticalidad
Enterprise Archive(EAR)	8.0	10	La versión del módulo de aplicación empaquetado (EAR) utilizada en el sistema SIGEPAA es la versión 8.	La elección de la versión 10 del formato EAR indica que el sistema SIGEPAA sigue las especificaciones y estándares establecidos para el empaquetado de aplicaciones empresariales en el entorno Jakarta EE.	Sí	La actualización del formato EAR es crítica para garantizar la compatibilidad y el cumplimiento de las especificaciones y estándares actuales en el entorno Jakarta EE 10.

JDBC (Java Database Connectivity)	4.3	5.0	La aplicación SIGEPAA utiliza JDBC (Java Database Connectivity) 4.3 para establecer conexiones con la base de datos y ejecutar consultas SQL.	En la versión 5.0 de JDBC se introducen mejoras significativas en el rendimiento y la seguridad, así como nuevas funcionalidades para trabajar con bases de datos modernas y alinearse con los cambios en los estándares de la industria.	Sí	La actualización de JDBC es crítica para aprovechar las mejoras en el rendimiento y la seguridad, así como para garantizar la compatibilidad con las bases de datos modernas en el entorno Jakarta EE 10.
JPA (Java Persistence API)	2.1	3.0	La aplicación SIGEPAA utiliza JPA (Java Persistence API) 2.1 para mapear objetos Java a tablas de bases de datos y viceversa.	La versión 3.0 de JPA introduce mejoras en la administración de transacciones, la gestión de caché y el soporte para bases de datos NoSQL, lo que proporciona una mayor	Sí	La actualización de JPA es crítica para aprovechar las mejoras en la administración de transacciones y la gestión de caché, así como para

				flexibilidad y rendimiento en la persistencia de datos.		garantizar la compatibilidad con bases de datos NoSQL en el entorno Jakarta EE 10.
Servlets	3.1	5.0	Los Servlets 3.1 son utilizados en la aplicación SIGEPAA para manejar solicitudes HTTP y generar respuestas dinámicas.	Con la versión 5.0 de Servlets se mejoran la escalabilidad, la seguridad y el manejo de solicitudes asíncronas, lo que permite una mejor gestión del tráfico web y una experiencia de usuario más fluida.	Sí	La actualización de Servlets es crítica para mejorar la escalabilidad y la seguridad, así como para aprovechar las capacidades de manejo de solicitudes asíncronas en el entorno Jakarta EE 10.

CDI (Contexts and Dependency Injection)	1.2	3.0	La aplicación SIGEPAA utiliza CDI (Contexts and Dependency Injection) 1.2 para administrar la creación y la inyección de dependencias entre los diferentes componentes de la aplicación.	La versión 3.0 de CDI introduce características avanzadas como la inyección de dependencias condicional y la integración con la especificación de microperfil de Jakarta EE, lo que facilita la creación de aplicaciones modulares y de alto rendimiento.	Sí	La actualización de CDI es crítica para aprovechar las características avanzadas y mejorar el rendimiento de las aplicaciones en el entorno Jakarta EE 10.
JAX-RS (Java API for RESTful Web Services)	2.0	3.0	JAX-RS 2.0 se emplea en SIGEPAA para desarrollar servicios web RESTful que permiten la	Con la versión 3.0 de JAX-RS se mejoran la compatibilidad con HTTP/2, la gestión de errores y la integración con otros componentes	Sí	La actualización de JAX-RS es crítica para garantizar la compatibilidad con HTTP/2 y mejorar la gestión de errores

			interacción con recursos específicos de la aplicación a través de la web.	de Jakarta EE, lo que proporciona una mayor interoperabilidad y eficiencia en el desarrollo de servicios web.		en el entorno Jakarta EE 10.
JSF (JavaServer Faces)	2.2	3.0	JSF 2.2 se utiliza en SIGEPAA para construir la interfaz de usuario basada en componentes.	La versión 3.0 de JSF trae consigo mejoras en el rendimiento, la seguridad y la facilidad de uso, así como nuevas características para la creación de aplicaciones web modernas y altamente interactivas.	Sí	La actualización de JSF es crítica para mejorar el rendimiento y la seguridad, así como para proporcionar una experiencia de usuario mejorada en el entorno Jakarta EE 10.

EJB Module(Enterprise JavaBeans)	3.2	4.0	El sistema SIGEPAA se basa en el módulo EJB (Enterprise JavaBeans) versión 3.2.	La versión 4.0 de EJB introduce mejoras en la administración de transacciones, la seguridad y la interoperabilidad, lo que facilita el desarrollo de componentes empresariales escalables y fiables en el entorno Jakarta EE.	Sí	La actualización de EJBs es crítica para mejorar la escalabilidad y la seguridad, así como para facilitar el desarrollo de componentes empresariales en el entorno Jakarta EE 10.
JDK(Kit de Desarrollo de Java)	1.8.0_241	21	El sistema SIGEPAA utiliza JDK 1.8.0_241 como entorno de desarrollo para la implementación de sus componentes y funcionalidades.	El JDK 11 proporciona herramientas avanzadas de desarrollo, soporte mejorado para lenguajes alternativos, y características de seguridad y rendimiento mejoradas	Sí	La actualización del JDK es crítica para aprovechar las mejoras en el rendimiento y la seguridad, así como para garantizar la compatibilidad con las últimas

				para la construcción de aplicaciones empresariales en Jakarta EE 10.		especificaciones y tecnologías en el entorno Jakarta EE 10.
Frameworks y Bibliotecas Externas	Primefaces 7	Primefaces 10 Jakarta	SIGEPAA hace uso de varios frameworks y bibliotecas externas para mejorar su funcionalidad y productividad.	La versión 10 de Primefaces para Jakarta EE introduce nuevas características, componentes y mejoras de rendimiento para enriquecer la interfaz de usuario y acelerar el desarrollo de aplicaciones web modernas en el entorno Jakarta EE.	Sí	La actualización de PrimeFaces es crítica para aprovechar las nuevas características y mejoras en el rendimiento y la usabilidad de los componentes visuales en el entorno Jakarta EE 10.
WildFly	18	30	El sistema SIGEPAA utiliza WildFly 18 como su	WildFly 30 proporciona un entorno de ejecución	Sí	La actualización de WildFly es crítica para mejorar la

	servidor de aplicaciones principal.	robusto y altamente escalable para aplicaciones empresariales en Jakarta EE 10, con características avanzadas de administración, seguridad y despliegue para garantizar un rendimiento óptimo y una alta disponibilidad del sistema.	escalabilidad, la seguridad y la administración del servidor en el entorno Jakarta EE 10.

Nota: Adaptado de: Desarrollo de aplicaciones web con Jakarta EE (Cesar Castillo, 2022)

La tabla 9, muestra un análisis de la escalabilidad y modularidad del sistema, proporcionando una visión detallada de cómo el sistema SIGEPAA puede expandirse y adaptarse para cumplir con las demandas futuras. Este análisis es crucial para determinar cómo Jakarta EE puede mejorar y optimizar estas capacidades clave del sistema.

Tabla 9. Análisis de Escalabilidad y Modularidad del Sistema

Punto	Descripción	Impacto
Capacidad de añadir nuevas funcionalidades	Evaluación de la capacidad del sistema SIGEPAA para integrar nuevas funcionalidades sin perturbar las existentes.	Sí, para mantener la competitividad y responder a las necesidades del usuario.
Escalabilidad horizontal y vertical	Análisis de la capacidad del sistema para aumentar su capacidad tanto horizontal como verticalmente para manejar cargas adicionales. La escalabilidad horizontal se refiere a la capacidad de un sistema para gestionar un aumento en la carga distribuyendo el trabajo entre múltiples instancias del sistema, como agregar más servidores a un conjunto web. La escalabilidad vertical es la capacidad de un sistema para manejar un aumento en la carga aumentando los recursos en una sola instancia del sistema, como aumentar la capacidad de memoria o CPU en un servidor existente.	Sí, para soportar el crecimiento del negocio y picos de demanda.

Facilidad de mantenimiento y actualización	Evaluación de la modularidad y claridad del diseño del sistema para determinar la facilidad de realizar mantenimientos y actualizaciones.	Sí, para garantizar la continuidad operativa y la adaptabilidad a cambios futuros.

Nota: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

2.1.3 Revisión de Especificaciones de Jakarta EE 10

Esta sección se enfoca en comparar detalladamente las especificaciones y estándares de Java EE 8 y Jakarta EE 10 para identificar los cambios y las nuevas características introducidas en la plataforma. El objetivo es analizar la evolución de la tecnología y su impacto en la actualización del sistema.

La tabla 10, indica una revisión de las especificaciones de Jakarta EE 10, comparándolas detalladamente con los estándares de Java EE 8. Este análisis se centra en identificar los cambios y nuevas características introducidas en la plataforma, con el objetivo de entender la evolución tecnológica y su impacto en la actualización del sistema SIGEPAA.

Tabla 10. Comparación de Especificaciones y Estándares

Aspecto	Java EE	Jakarta EE	Mejoras y Razón
Especificaciones Actualizadas	Servlet 3.1	Servlet 4.0	Soporte para características avanzadas de servlets. Esto incluye mejoras en la manipulación de peticiones HTTP, manejo de streams y mejora en la seguridad.
	JPA 2.1	JPA 2.2	Introducción de nuevas funcionalidades de mapeo objeto-relacional, mejoras en consultas y en la gestión de transacciones, lo que resulta en un mejor rendimiento y facilidad de uso.
	JSF 2.2	JSF 2.3	Incorporación de nuevas características para el desarrollo de interfaces de usuario, incluyendo mejoras en el rendimiento, soporte para

			HTML5 y una mayor flexibilidad en la configuración.
	JAX-RS 2.0	JAX-RS 2.1	Introducción de nuevas funcionalidades para el desarrollo de servicios web RESTful, como mejoras en la seguridad, manejo de errores y soporte para protocolos avanzados.
	CDI 1.2	CDI 2.0	Mejora en la inyección de dependencias y la gestión de contextos, facilitando el desarrollo de aplicaciones modulares y altamente escalables.
		JSON-B 1.0	Introducción de una API estándar para la serialización y deserialización de objetos Java a JSON, simplificando el intercambio de datos entre aplicaciones.

		JSON-P 1.1	Incorporación de una API estándar para el procesamiento de documentos JSON, facilitando la manipulación de datos en formato JSON.
		Bean Validation 2.0	Mejora en la validación de datos en aplicaciones empresariales, con soporte para nuevas anotaciones y reglas de validación.
Nuevas Características Introducidas	- -	Introducción de MicroProfile	Adición de un conjunto de especificaciones y APIs diseñadas para facilitar el desarrollo de microservicios Java empresariales en entornos de nube.
	- -	Inclusión de nuevos APIs	Incorporación de nuevas APIs y funcionalidades para abordar desafíos modernos en el desarrollo de aplicaciones,

			como la gestión de datos no estructurados y la mejora en la seguridad.
	- -	Mejoras en la seguridad y el rendimiento	Implementación de medidas adicionales para fortalecer la seguridad y mejorar el rendimiento de las aplicaciones empresariales, garantizando su funcionalidad y confiabilidad.
	- -	Integración con Jakarta NoSQL (opcional)	
		Soporte para HTTP/2	
		API de Reactivo	
		Mejoras en la compatibilidad de Microservices	
		Actualizaciones de los servidores de aplicaciones	Actualización de los servidores de aplicaciones compatibles con Jakarta EE 10 para ofrecer un mejor rendimiento, mayor

Cambios en los Estándares de Implementación	- Soporte para servidores como Glassfish,WildFly, TomEE, etc	compatibles como WildFly, Payara, TomEE, etc.	seguridad y soporte para las últimas tecnologías.
		Cambios en las bibliotecas de terceros	Incorporación de nuevas versiones de bibliotecas externas para mejorar la compatibilidad y el rendimiento de las aplicaciones, aprovechando al máximo las funcionalidades más recientes disponibles.
		Mayor soporte para contenedores Docker y Kubernetes	
		Mejoras en la integración con Eclipse MicroProfile	
		Aumento de la modularidad y extensibilidad del framework	

Mejoras de Rendimiento y Escalabilidad	- -	Optimizaciones en el manejo de memoria	Estas optimizaciones son críticas porque pueden ayudar a mejorar el rendimiento general de la aplicación, reducir el riesgo de errores relacionados con la memoria y permitir que la aplicación se ejecute de manera más eficiente en entornos con recursos limitados.
	- -	Mejoras en el rendimiento de las consultas JPA	Optimización de la gestión de memoria para reducir el uso de recursos y mejorar la eficiencia del sistema. Optimización de la ejecución de consultas JPA para mejorar la interacción con la base de datos y reducir los tiempos de respuesta.

	- -	Optimización de la serialización JSON	Mejora de la serialización JSON para agilizar la transmisión de datos entre el cliente y el servidor.
	- -	Implementación de caché distribuida	Introducción de una caché distribuida para mejorar el rendimiento y la escalabilidad del sistema, reduciendo la necesidad de acceder a recursos externos para recuperar datos.
	- -	Soporte para escalabilidad horizontal y vertical	Mejora de la capacidad del sistema para gestionar cargas de trabajo variables y el crecimiento futuro, mediante escalabilidad horizontal y vertical, asegurando un rendimiento óptimo.
	- -	Optimizaciones en la gestión de hilos	Optimización de la gestión de hilos para incrementar la eficiencia y la capacidad de respuesta del sistema, reduciendo los tiempos

			de procesamiento y mejorando la escalabilidad.
	- -	Mejoras en la gestión de recursos	Mejora en la utilización eficiente de recursos como memoria, CPU y conexiones de red, optimizando el rendimiento y la estabilidad del sistema.
	- -	Reducción del tiempo de arranque y despliegue	Reducción del tiempo necesario para iniciar y desplegar la aplicación, lo que permite una puesta en marcha más rápida y una menor interrupción del servicio.
	- -	Mayor tolerancia a fallos	Implementación de medidas para aumentar la estabilidad del sistema y reducir el impacto de posibles fallos, garantizando una mayor disponibilidad y confiabilidad.

Nota: Adaptado de: Desarrollo de aplicaciones web con Jakarta EE (Cesar Castillo, 2022)

2.1.4 Análisis de Compatibilidad con Jakarta EE 10

En la *tabla 11* se realiza un estudio exhaustivo de la compatibilidad del sistema con Jakarta EE 10, evaluando las APIs y tecnologías específicas utilizadas. Este análisis considera diversos aspectos clave para asegurar una migración sin problemas y efectiva hacia la nueva plataforma en la cual se consideran los siguientes aspectos:

Tabla 11. Compatibilidad con Jakarta EE 10

Punto	Descripción	Compatibilidad con Jakarta EE 10
Compatibilidad de APIs y Bibliotecas	- Se analiza la compatibilidad de las APIs y bibliotecas utilizadas en el sistema, como JPA, JSF, JAX-RS, CDI, etc., con las versiones correspondientes en Jakarta EE 10.	Sí, muchas de las APIs y bibliotecas de Java EE 8 son compatibles con Jakarta EE 10, ya que Jakarta EE es una evolución de Java EE y se centra en la continuidad y la compatibilidad con versiones anteriores.
Evaluación de Dependencias	- Se examinan las dependencias del sistema con respecto a las APIs y tecnologías específicas para identificar posibles conflictos o problemas de compatibilidad al migrar a Jakarta EE 10.	Algunas dependencias pueden requerir actualizaciones o ajustes menores para garantizar la compatibilidad con Jakarta EE 10. Esto puede incluir la actualización de versiones o la modificación de configuraciones.
Pruebas de Integración	- Se realizan pruebas de integración para verificar la compatibilidad del sistema	Las pruebas de integración pueden confirmar la compatibilidad del sistema con

	con Jakarta EE 10 y asegurar que todas las funcionalidades se mantengan intactas después de la actualización.	Jakarta EE 10 al demostrar que todas las funcionalidades se ejecutan correctamente en el nuevo entorno sin errores ni fallos.

Nota: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

2.1.5 Análisis de la Integración con WildFly 30, Eclipse 2024 y PrimeFaces 10 Jakarta

La *tabla 12* proporciona una visión clara de la comparativa entre las dos versiones de WildFly, ayudando a evaluar las mejoras y cambios que ofrecen, especialmente en términos de soporte, rendimiento, seguridad y compatibilidad con Jakarta EE.

Tabla 12. Puntos de mejora en WildFly

Característica	**WildFly 18**	**WildFly 30**
Actualizaciones	Menos frecuentes	Más frecuentes
Soporte	Limitado	Mejorado
Mejoras de rendimiento	Menos optimizaciones	Más optimizaciones
Seguridad	Básica	Mejoras significativas
Características nuevas	Menos	Más (Mejoras en la escalabilidad y el rendimiento)
Estabilidad	Algunos problemas	Mayor estabilidad
Compatibilidad	Compatible	Mejor compatibilidad

Documentación	Adecuada	Mejorada
Soporte Jakarta EE	No	Si

Nota: Elaborado por: Narváez Esteban, basado en (Jakarta EnterPrise, 2020)

A continuación, se realiza un análisis detallado de los aspectos de mejora propuestos en la tabla 12.

- **Actualizaciones:** En WildFly 18, las actualizaciones eran menos frecuentes, lo que limitaba la capacidad de los usuarios para acceder a las últimas mejoras y correcciones de errores. Con WildFly 30, las actualizaciones se han vuelto más frecuentes, lo que permite una mayor rapidez en la implementación de nuevas características y correcciones.
- **Soporte:** WildFly 18 ofrecía un soporte limitado, lo que significaba que los usuarios podían enfrentar dificultades para obtener ayuda o soluciones a problemas técnicos. Con WildFly 30, el nivel de soporte ha mejorado significativamente, lo que se traduce en una mayor disponibilidad de recursos de asistencia técnica y una mejor atención al cliente.
- **Mejoras de rendimiento:** En WildFly 18, las optimizaciones de rendimiento eran menos frecuentes y efectivas, lo que podía afectar el rendimiento general del servidor en cargas de trabajo exigentes. Con WildFly 30, se han implementado más optimizaciones de rendimiento, lo que se traduce en una mejor respuesta y eficiencia del servidor.
- **Seguridad:** En WildFly 18, las funcionalidades de seguridad eran básicas, lo que podía dejar al servidor vulnerable a ataques y violaciones de seguridad. Con

WildFly 30, se han introducido mejoras significativas en la seguridad, como nuevas características de protección y correcciones de vulnerabilidades conocidas.

- **Características nuevas:** WildFly 18 tenía menos características nuevas en comparación con WildFly 30, lo que limitaba la funcionalidad y la flexibilidad del servidor. Con WildFly 30, se han introducido más características nuevas, como compatibilidad con tecnologías emergentes y mejoras en la administración y el monitoreo del servidor.
- **Estabilidad:** En WildFly 18, algunos problemas de estabilidad podían afectar la experiencia del usuario y la disponibilidad del servidor. Con WildFly 30, se ha mejorado significativamente la estabilidad del servidor, lo que ha resultado en una menor frecuencia de fallos y caídas del sistema.
- **Compatibilidad:** WildFly 18 era compatible con versiones anteriores, pero podía presentar problemas de interoperabilidad con algunas tecnologías más recientes. Con WildFly 30, se ha mejorado la compatibilidad con nuevas tecnologías y estándares, lo que facilita la integración con otros sistemas y aplicaciones.
- **Documentación:** La documentación en WildFly 18 era adecuada, pero podía carecer de detalles o ejemplos prácticos en ciertos casos. Con WildFly 30, se ha mejorado la documentación, proporcionando información más completa y detallada sobre la configuración, la administración y el uso del servidor.

*2.1.6. **Actualización del entorno de desarrollo integrado (IDE) Eclipse.***

La tabla 13 destaca las ventajas clave que los usuarios pueden obtener al actualizar el IDE Eclipse a versiones más recientes, proporcionando mejoras significativas en rendimiento, funcionalidad y soporte.

Tabla 13. Comparativa de mejoras Eclipse

Aspectos	Eclipse 2020	Eclipse 2024
Rendimiento	Rendimiento aceptable en proyectos medianos	Mejora significativa en la velocidad de carga y respuesta en proyectos de gran escala.
Funcionalidad	Ofrece las funcionalidades básicas necesarias para el desarrollo.	Introduce nuevas características y mejoras en herramientas existentes, como refactorización de código, análisis estático, etc.
Soporte	Soporte estable proporcionado por la comunidad y el equipo de Eclipse.	Soporte continuo y actualizado con correcciones de errores y parches de seguridad. Posibilidad de recibir soporte comercial.
Seguridad	Nivel de seguridad aceptable, pero algunas vulnerabilidades pueden existir.	Mejoras en la seguridad con actualizaciones regulares para abordar nuevas amenazas de seguridad.
Integración de Plugins	Compatibilidad con una amplia gama de plugins, pero algunos pueden no ser totalmente compatibles.	Mayor compatibilidad y estabilidad con plugins de terceros. Actualizaciones frecuentes de los plugins populares.

Corrección de Errores	Corrección de errores periódica, pero algunas características pueden estar sujetas a fallos.	Mayor eficiencia en la detección y corrección de errores, con un proceso de prueba más riguroso.
Interfaz de Usuario	Interfaz de usuario familiar, pero puede carecer de modernidad en comparación con otros IDEs.	Mejoras en la interfaz de usuario para una experiencia más moderna y personalizable. Mejora en la usabilidad y accesibilidad.
Desarrollo Colaborativo	Herramientas de colaboración básicas, como la capacidad de compartir proyectos a través de repositorios compartidos.	Mejora en las herramientas de colaboración, incluyendo la edición en tiempo real, comentarios contextuales, y la integración con herramientas de gestión de proyectos.
Soporte de Lenguajes	Soporte para una amplia gama de lenguajes de programación, pero puede carecer de soporte completo para las últimas versiones o características de los lenguajes.	Ampliación del soporte para nuevos lenguajes y versiones, con herramientas de desarrollo específicas y análisis de código mejorado.
Desarrollo en la Nube	Capacidades limitadas para el desarrollo en la nube, con algunas integraciones disponibles, pero no completamente optimizadas.	Mejora en las capacidades de desarrollo en la nube, con integración más profunda con plataformas de nube populares y herramientas de despliegue automatizado.

Nota: Elaborado por: Narváez Esteban, basado en (Fundamentos a Jakarta E.E., 2019)

La tabla 14 resalta las ventajas significativas de actualizar las librerías de PrimeFaces, lo que contribuye a mejorar la funcionalidad, seguridad y experiencia del usuario en aplicaciones web basadas en esta tecnología.

Tabla 14. Comparativa de mejoras PrimeFaces

Aspecto	PrimeFaces 7	PrimeFaces 10 Jakarta
Nuevas Funcionalidades y Mejoras	Ofrece un conjunto sólido de componentes de interfaz de usuario.	Introduce nuevas funcionalidades y mejoras para mejorar la experiencia del usuario y agregar capacidades adicionales.
Compatibilidad y Mantenibilidad	Compatible con versiones anteriores, pero puede carecer de características y optimizaciones más recientes.	Alineado con las últimas tecnologías y estándares, facilitando la compatibilidad y asegurando la mantenibilidad a largo plazo.
Corrección de Errores y Parches de Seguridad	Puede contener errores conocidos o vulnerabilidades de seguridad que se han corregido en versiones posteriores.	Incluye correcciones de errores y parches de seguridad para mejorar la estabilidad y proteger contra amenazas de seguridad.
Soporte de la Comunidad	Ha recibido soporte y atención de la comunidad de desarrolladores.	Recibe un mayor soporte y atención debido a su reciente lanzamiento y actualizaciones adicionales.

Nota: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

2.1.5 Planificación del Alcance y Recursos

La actualización del sistema SIGEPAA a Jakarta EE 10 representa una iniciativa estratégica para modernizar la infraestructura tecnológica, mejorar el rendimiento y garantizar la compatibilidad con las últimas especificaciones y estándares. Este proceso de migración incluye

la identificación y actualización de funcionalidades clave, como servicios web, componentes de la interfaz de usuario y lógica empresarial. Además, se añadirá el módulo de Secretaría o archivo, mejorando la funcionalidad del sistema y facilitando nuevas características operativas.

2.1.7 Identificación de Funcionalidades para Migrar

Es Importantes destacar que, para tener una buena funcionalidad en este tipo de procesos, se procede a realizar una buena identificación y los posibles cambios que lograrían tener una eficiencia en la migración del sistema.

- **Identificación:** Se realizará un inventario exhaustivo de los servicios web actualmente implementados en el sistema SIGEPAA.
- **Cambios Necesarios:** Se evaluarán y aplicarán las modificaciones necesarias en la implementación y configuración para garantizar la compatibilidad con Jakarta EE 10. Esto incluye la actualización de anotaciones y posibles ajustes en las clases de recursos y controladores.

2.1.8. Migración de Páginas JSF y Componentes PrimeFaces:

- **Identificación:** Se identificarán todas las páginas JSF (XHTML) y los componentes PrimeFaces utilizados en la interfaz de usuario.
- **Cambios Necesarios:** Se evaluarán los cambios necesarios en la estructura y comportamiento de estos componentes. La migración a PrimeFaces 10 Jakarta incluirá la actualización de bibliotecas, ajuste de componentes y pruebas de funcionalidad.

2.2. Migración de Componentes de Lógica Empresarial

- **Identificación**: Se identificarán los EJBs (Enterprise JavaBeans) y CDI beans (Contexts and Dependency Injection) que encapsulan la lógica empresarial del sistema.
- **Cambios Necesarios**: Se revisarán las diferencias en la implementación y configuración de estos componentes en Jakarta EE 10, asegurando que se mantenga la funcionalidad y se optimice el rendimiento

Asignación de recursos y roles en el proceso de actualización

Cuando se considera una actualización, la gestión de recursos se convierte en un aspecto clave. La asignación de recursos implica identificar y distribuir los recursos disponibles de manera estratégica para una iniciativa específica. Una asignación efectiva de recursos permite aprovechar al máximo las capacidades del equipo, optimizando los resultados y apoyando los objetivos del proyecto. Para desarrollar un plan de asignación de recursos, es fundamental identificar los recursos adecuados, como personal, herramientas y presupuesto, necesarios para cumplir con los objetivos del proyecto.

Al abordar una actualización, es crucial prestar atención especial a los recursos tecnológicos y financieros. Estos recursos no solo respaldan el proyecto en curso, sino que también garantizan su viabilidad a largo plazo. En este sentido, disponemos de:

Recursos Tecnológicos.

Se puede afirmar que los recursos tecnológicos para la implementación de Jakarta EE son los elementos esenciales para desarrollar, desplegar y ejecutar aplicaciones basadas en esta tecnología, tales como contenedores de aplicaciones, servidores de bases de datos, herramientas de desarrollo, frameworks y APIs específicas de Jakarta EE. Entre estos recursos se incluyen:

- **Hardware y Software:** Garantizar la disponibilidad del hardware y software necesarios para la implementación de Jakarta EE y el sistema actualizado.
- **Infraestructura de Red:** Garantizar que la infraestructura de red esté disponible y tenga el rendimiento adecuado para soportar el tráfico generado por el sistema actualizado.

Por otro lado, la necesidad de más recursos que complemente a la migración en el sistema de Jakarta EE se referiría aquellos fondos económicos necesarios para llevar a cabo el desarrollo, mantenimiento y promoción de la plataforma Java para aplicaciones empresariales, es decir recursos financieros.

Recursos Financieros

Los recursos financieros son aquellos indispensables para asegurar la continuidad y mejora constante de Jakarta EE, permitiendo su evolución y adaptación a las necesidades cambiantes de la industria de desarrollo de software empresarial, entre ellos esta:

- **Presupuesto:** Asignar un presupuesto adecuado para cubrir los costos relacionados con la migración, incluyendo la adquisición de herramientas y recursos adicionales si es necesario.
- **Respaldos y recuperación de datos**. Realizar copias de seguridad completas de todos los datos y configuraciones críticas antes de iniciar la migración. Además, contar con un plan claro de recuperación de datos en caso de pérdida de información durante el proceso de migración.

2.1.9. Establecimiento de Límites del Proyecto

En la *tabla 10* se identifica y establece los límites del proyecto de actualización del sistema SIGEPAA a Jakarta EE 10, definiendo claramente los módulos y componentes que no serán modificados durante este proceso. El objetivo es delimitar el alcance del proyecto y evitar cambios innecesarios en partes del sistema que funcionan correctamente en la versión actual.

Tabla 15. Análisis en la modificación de componentes

Aspecto	Descripción	Justificación
Módulos y Componentes no Modificados	Identificación de Módulos y Componentes	Se identificarán los módulos y componentes del sistema SIGEPAA que no serán modificados durante la actualización a Jakarta EE 10. Esto incluye funcionalidades heredadas, módulos obsoletos y componentes que no son relevantes para la migración. **Ejemplos**: - Módulo de Reportes Históricos: Este módulo, utilizado para generar informes basados en datos antiguos, no requiere actualización ya que sigue cumpliendo con los requisitos funcionales actuales. - Componentes de Autenticación y Autorización: Estos componentes han demostrado ser robustos y seguros en su versión actual, y no se prevé que necesiten ajustes con la migración.
Razones para no Modificar	Sin modificación	Se proporcionarán justificaciones claras para no modificar estos módulos y componentes durante la actualización. Esto incluye consideraciones de tiempo, recursos, impacto en otras áreas del sistema y la estabilidad de las funcionalidades actuales. **Consideraciones de Tiempo**: - Actualizar estos módulos podría extender significativamente el cronograma del proyecto sin un beneficio claro para la

	funcionalidad o el rendimiento del sistema. **Recursos Limitados**: - El equipo de desarrollo y los recursos tecnológicos disponibles están optimizados para centrarse en las áreas críticas que requieren actualización. **Impacto en Otras Áreas**: - Modificar estos componentes podría introducir riesgos adicionales y complicaciones en otras áreas del sistema que actualmente funcionan de manera estable. **Estabilidad y Funcionalidad Actuales**: - Algunos módulos y componentes han demostrado ser altamente fiables y cumplen con los requisitos actuales, por lo que no se justifica su modificación en esta etapa.

Nota: Elaborado por: Narváez Esteban, basado en (Fundamentación Teórica a Jakarta EnterPrise, 2023)

Los límites claros para el proyecto de actualización del sistema SIGEPAA a Jakarta EE 10 es crucial para asegurar que el proyecto se mantenga enfocado y eficiente. Al identificar y justificar los módulos y componentes que no serán modificados, se pueden optimizar los recursos y el tiempo, minimizando riesgos y asegurando que las partes críticas del sistema se actualicen de manera efectiva**.**

2.1.10. Estimación de Tiempos por Fase del Proyecto

En esta sección se estimarán los tiempos requeridos para las diferentes fases del proyecto de actualización del sistema SIGEPAA a Jakarta EE 10. Este proceso incluye un análisis exhaustivo, diseño, implementación, pruebas y despliegue.

Tabla 16. Tiempos de estimación fases del proyecto

Fase del Proyecto	Descripción	Tiempo Estimado	Razón de la Estimación
Análisis	Se calculará el tiempo necesario para realizar un análisis exhaustivo del sistema actual y definir los requisitos de migración a Jakarta EE 10.	3 semanas	Involucra un estudio detallado del sistema actual, identificación de dependencias y evaluación de compatibilidad con Jakarta EE 10.
Diseño	Se estimará el tiempo requerido para diseñar la arquitectura actualizada del sistema en Jakarta EE 10, incluyendo la planificación de la migración de componentes y funcionalidades.	2 semanas	Requiere la creación de un plan detallado de migración, diagramas de arquitectura y diseño de nuevas funcionalidades.
Implementación	Se calculará el tiempo necesario para llevar a cabo la migración efectiva del código y la configuración del sistema a Jakarta EE 10, así como para integrar nuevas funcionalidades y realizar pruebas unitarias.	6 semanas	Incluye la actualización del código, integración de nuevas APIs, ajustes de configuración y desarrollo de nuevas funcionalidades.
Pruebas	Se estimará el tiempo necesario para realizar pruebas exhaustivas del sistema actualizado, que incluirán pruebas unitarias, pruebas de integración, de rendimiento y aceptación por parte del usuario.	4 semanas	Comprende la realización de pruebas para asegurar la funcionalidad, estabilidad, rendimiento y aceptación del usuario final.

Despliegue	Se calculará el tiempo necesario para preparar el entorno de producción, desplegar el sistema actualizado y realizar cualquier configuración adicional necesaria para su funcionamiento óptimo.	2 semanas	Involucra la preparación del entorno de producción, despliegue del sistema y configuraciones finales para asegurar el rendimiento óptimo.
Mantenimiento	Se planificará el tiempo para realizar el mantenimiento post-despliegue y capacitar al personal de EPAAA para que se encargue de ello.	1 semanas	Incluye el monitoreo del sistema, corrección de errores, optimizaciones adicionales y capacitación del personal para asegurar un mantenimiento continuo y efectivo.

3. ***Nota***: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

3.1.9. Identificación, Evaluación y Gestión de Riesgos

La *tabla 17* muestra una visión clara y detallada de los riesgos potenciales, sus causas subyacentes, el impacto potencial y las estrategias de mitigación, permitiendo una gestión proactiva y efectiva de los riesgos durante la actualización del sistema SIGEPAA a Jakarta EE 10.

Tabla 17. Identificación de Riesgos Potenciales en la migración

Riesgo Potencial	Descripción	Causa	Impacto Potencial	Estrategia de Mitigación
Problemas de Compatibilidad	Riesgos relacionados con la incompatibilidad entre	- Falta de soporte para ciertas APIs en Jakarta EE 10.	Alta: Puede causar fallos en el funcionamiento del	- Realizar una auditoría exhaustiva de todas las

	las tecnologías utilizadas en la versión actual del sistema y las especificaciones de Jakarta EE 10.	- Dependencias de bibliotecas externas no compatibles. - Diferencias en la implementación de servicios web y componentes de UI.	sistema, bloqueos y errores críticos que impidan la operación normal del sistema.	dependencias y bibliotecas utilizadas. - Probar cada componente en un entorno de desarrollo antes de la migración completa. - Mantener una lista de soluciones alternativas para problemas de compatibilidad conocidos.
Pérdida de Datos	Riesgos asociados con la pérdida accidental o no autorizada de datos durante el proceso de migración o debido a errores en la implementación.	- Errores en scripts de migración de base de datos. - Falta de copias de seguridad adecuadas. - Fallos en la integración de nuevas funcionalidades con datos existentes.	Muy Alta: La pérdida de datos puede afectar gravemente la integridad y confiabilidad del sistema, causando pérdida de información crucial y afectando la confianza de los usuarios.	- Realizar copias de seguridad completa y verificada antes de iniciar la migración. - Desarrollar y probar minuciosamente scripts de migración de datos. - Implementar medidas de recuperación de datos y planes de contingencia.

Interrupción del Servicio	Riesgos relacionados con la interrupción del servicio del sistema durante la actualización, lo que podría afectar la disponibilidad y el rendimiento del sistema para los usuarios finales.	- Tiempo de inactividad durante el despliegue de la actualización. - Problemas de rendimiento debido a configuraciones inadecuadas. - Fallos en la coordinación de despliegue.	Media: La interrupción del servicio puede causar inconvenientes a los usuarios finales, afectando la productividad y la percepción del sistema.	- Planificar ventanas de mantenimiento con anticipación y comunicar claramente a los usuarios. - Realizar pruebas de carga y estrés para asegurar el rendimiento post-migración. - Tener un plan de reversión listo para situaciones de emergencia.

4. ***Nota***: Elaborado por: Narváez Esteban, basado en (Características Fundamentales Jakarta EnterPrise, 2021)

4.1.9. Impacto de actualización Java EE a Jakarta EE

En la *tabla 18* se determina el impacto que tiene actualizar java EE a Jakarta EE

Tabla 18. Impacto de actualización

Aspecto	Descripción	Impacto Actualización
Modernización y Soporte Continuo	Jakarta EE 10 es la evolución de Java EE 8, ofreciendo soporte para tecnologías modernas.	La actualización asegura que el sistema se mantenga actualizado con las últimas tecnologías y estándares, facilitando el mantenimiento y mejorando la seguridad.

Rendimiento y Escalabilidad	Introduce mejoras en rendimiento y escalabilidad, como optimizaciones en el manejo de memoria y gestión de hilos.	Permite que el sistema maneje mayores volúmenes de tráfico y datos de manera más eficiente, mejorando la experiencia del usuario y la capacidad de respuesta.
Nuevas Funcionalidades y APIs	Incorpora nuevas APIs y funcionalidades, como MicroProfile y mejoras en la serialización JSON.	Permite desarrollar aplicaciones más ricas y eficientes, con capacidades avanzadas que no estaban disponibles en Java EE 8, mejorando la competitividad del sistema.
Mejoras en la Seguridad	Incluye mejoras en la seguridad, abordando vulnerabilidades y añadiendo nuevas características de seguridad.	Mejora la protección contra amenazas de seguridad, asegurando que el sistema sea más robusto y resistente a ataques, manteniendo la confianza de los usuarios.
Compatibilidad con Ecosistema Moderno	Compatible con las versiones más recientes de servidores de aplicaciones y herramientas de desarrollo como WildFly 30, Eclipse 2024 y PrimeFaces 10 Jakarta.	Facilita la integración con otras tecnologías modernas, mejorando la productividad del desarrollo y la interoperabilidad del sistema con otras aplicaciones y servicios.
Optimización de Desarrollo	Mejora en herramientas y bibliotecas de desarrollo.	Mejora la eficiencia del equipo de desarrollo, reduciendo el tiempo de implementación y optimizando la calidad del código.
Soporte para Microservicios	Mejor soporte para arquitectura de microservicios a través de MicroProfile.	Facilita la creación de aplicaciones más modulares y escalables, permitiendo una mejor gestión y despliegue de servicios independientes.

Ecosistema de Innovación	Mayor participación de la comunidad y contribuciones al proyecto Jakarta EE.	Acceso a una comunidad activa y en constante innovación, asegurando mejoras continuas y soporte a largo plazo.
Mejor Gestión de Recursos	Mejoras en la gestión de recursos y optimización de la utilización de hardware.	Disminuye los costos operativos al optimizar el uso de los recursos del servidor y mejorar el rendimiento.
Reducción del Tiempo de Arranque y Despliegue	Optimizaciones en el tiempo de arranque y despliegue de aplicaciones.	Permite despliegues más rápidos y frecuentes, mejorando la agilidad en el desarrollo y la respuesta a cambios en el mercado.

Nota: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

2.2. **Fase 2**. Diseño (Diseño de la Actualización a Jakarta EE 10)

Definición de la Nueva Arquitectura

En esta subsección se propondrá una nueva arquitectura actualizada para el sistema SIGEPAA, aprovechando las características y mejoras de Jakarta EE 10. Se describirán los cambios en la estructura de capas, la distribución de componentes y la integración de nuevas tecnologías.

1. **Clientes**

Los clientes del sistema incluyen tanto los navegadores web como los dispositivos móviles. Estos clientes son responsables de interactuar con el servidor de aplicaciones para solicitar y enviar información. La interfaz de usuario se implementa utilizando tecnologías web estándar como HTML, CSS y JavaScript para navegadores, y frameworks específicos de Jakarta EE.

2. **Servidor de Aplicaciones**

El servidor de aplicaciones actúa como el núcleo del sistema donde se despliegan y ejecutan los módulos de la aplicación. Utiliza un servidor de aplicaciones compatible con Jakarta EE como WildFly 30.0.0. La modularización del sistema facilita la actualización y mantenimiento de los componentes, permitiendo que cada módulo se actualice de manera independiente.

3. **Capa de Presentación (View Layer)**

La capa de presentación utiliza tecnologías como JSF (Jakarta Server Faces) o JSP (Jakarta Server Pages) para generar las interfaces de usuario. Esta capa es responsable de manejar la interacción con el usuario final, renderizando las vistas y procesando las entradas del usuario.

4. **Capa de Negocio (Business Layer)**

La lógica de negocio del sistema se encuentra en esta capa. Se utiliza EJB (Enterprise JavaBeans) o CDI (Contexts and Dependency Injection) para gestionar las transacciones y la lógica de negocio de manera eficiente. Esta capa garantiza que las reglas de negocio se apliquen correctamente en todas las operaciones.

5. **Capa de Persistencia (Persistence Layer)**

La capa de persistencia se encarga de la interacción con la base de datos, utilizando JPA (Jakarta Persistence API) para gestionar las entidades y llevar a cabo operaciones CRUD (Crear, Leer, Actualizar, Eliminar). Esta capa asegura que los datos se almacenen y se recuperen de manera eficiente y segura.

6. **Base de Datos**

La base de datos del sistema es un componente fundamental que almacena toda la información clave. Se emplea un modelo relacional para organizar los datos en tablas y definir las relaciones entre ellas.

7. ***Integración de Tecnologías de Jakarta EE 10***

Se detallará cómo se integrarán las nuevas tecnologías y APIs de Jakarta EE 10 en el diseño de la arquitectura actualizada del sistema SIGEPAA. Se destacarán las ventajas y beneficios de estas integraciones.

8. ***APIs Nuevas y Actualizadas en Jakarta EE***

La *tabla 19* presenta una descripción detallada de varias tecnologías de Jakarta EE, incluyendo sus funcionalidades clave y las novedades introducidas en las versiones más recientes. Estas tecnologías abarcan desde la creación de servicios web y la gestión de datos hasta la inyección de dependencias y la comunicación asincrónica, proporcionando un conjunto robusto de herramientas para el desarrollo de aplicaciones empresariales.

Tabla 19. Tecnologías actualizadas en Jakarta EE

Tecnología	Descripción	Novedades
Jakarta RESTful Web Services (JAX-RS 3.1)	Facilita la creación de servicios web RESTful con soporte para patrones de diseño modernos y mejoras en la integración.	Mejoras en la configuración, nuevas anotaciones y soporte mejorado para tipos de retorno asíncronos.
Jakarta Persistence (JPA 3.1)	Proporciona una API para gestionar datos relacionales en aplicaciones Java.	Mejoras en el rendimiento de las consultas, soporte para tipos JSON y otras optimizaciones.
Jakarta Faces (JSF 4.0)	Utilizado para construir interfaces de usuario web basadas en componentes.	Nuevas características y componentes, optimizaciones de rendimiento y mejor integración con otras tecnologías.

Jakarta Server Pages (JSP 3.1)	Permite la creación de páginas web dinámicas basadas en Java.	Mejoras en el rendimiento y nuevas funcionalidades de scripting.
Jakarta Server Faces (JSF 4.0)	Framework para la construcción de interfaces de usuario web basadas en componentes.	Soporte para nuevas tecnologías web y mejoras en la experiencia del desarrollador.
Jakarta Dependency Injection (CDI 4.0)	Gestiona la inyección de dependencias en aplicaciones Java.	Nuevas características de inyección, mejoras en la configuración y rendimiento optimizado.
Jakarta Batch (Batch Processing 2.1	API para gestionar y ejecutar procesos batch en aplicaciones empresariales.	Soporte para nuevos patrones de procesamiento y mejor integración con otras APIs de Jakarta EE.
Jakarta Messaging (JMS 3.1)	Proporciona una API para la comunicación asincrónica basada en mensajes.	Nuevas características para mejorar la eficiencia y la capacidad de manejo de mensajes.
Jakarta Concurrency (Concurrency Utilities 3.0)	Facilita la gestión de tareas concurrentes y multi-hilo en aplicaciones Java.	Nuevas utilidades y mejoras en la gestión de tareas concurrentes.
Jakarta Security (Security 3.0)	Proporciona una API para gestionar la seguridad en aplicaciones empresariales.	Mejoras en la gestión de autenticación y autorización, y nuevas características de seguridad.
Jakarta JSON Processing (JSON-P 2.1)	Proporciona APIs para procesar y generar JSON.	Soporte para nuevas funcionalidades JSON y

		optimizaciones en el rendimiento de procesamiento.
Jakarta JSON Binding (JSON-B 2.1)	Proporciona una API para enlazar objetos Java a JSON y viceversa.	Mejoras en la flexibilidad de mapeo y optimización del rendimiento.
Jakarta WebSocket (WebSocket 2.1)	Proporciona una API para la comunicación bidireccional entre cliente y servidor.	Mejoras en la eficiencia de la comunicación y nuevas características de configuración.
Jakarta MVC (MVC 2.0)	Ofrece un marco de trabajo para el desarrollo de aplicaciones web siguiendo el patrón Modelo-Vista-Controlador.	Mejor integración con otras tecnologías Jakarta EE y nuevas características para mejorar la productividad del desarrollador.
Jakarta NoSQL (NoSQL 1.0)	Proporciona una API para interactuar con bases de datos NoSQL.	Introducción de soporte estándar para bases de datos NoSQL, facilitando su uso en aplicaciones Jakarta EE.
Jakarta Transactions (JTA 2.0)	Facilita la gestión de transacciones distribuidas en aplicaciones empresariales.	Mejoras en la confiabilidad y eficiencia de la gestión de transacciones.

Nota: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

Diseño de Componentes y Módulos

Tabla 20. Definición de Funcionalidades de los Componentes

Componente	Funcionalidades Clave	Interfaces y Métodos
Capa de Presentación	- Mostrar la interfaz de usuario con los elementos necesarios para la interacción. - Recibir y procesar la entrada del usuario.	- Interfaz gráfica que incluye métodos para la manipulación de elementos de la interfaz, eventos del usuario y navegación entre pantallas.
Capa de Lógica de Negocio	- Ejecutar la lógica de negocio requerida para procesar las solicitudes del usuario. - Validar la entrada del usuario y aplicar reglas de negocio.	- Interfaces que definen los servicios ofrecidos por la capa de negocio. - Métodos que implementan la lógica de negocio, incluyendo la validación de datos y la ejecución de procesos.
Capa de Acceso a Datos	- Acceder a la base de datos para recuperar y almacenar información. - Realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar).	- Interfaces que especifican métodos para realizar consultas a la base de datos, así como para insertar, actualizar y eliminar registros.
Capa de Seguridad	- Garantizar la autenticación y autorización de usuarios.	- Interfaces que definen métodos para autenticar usuarios, autorizar acciones y cifrar datos.

	- Proteger los datos sensibles mediante encriptación.	
Gestión de Sesiones	- Mantener y gestionar la información de sesión de usuario. - Controlar el flujo de la sesión y la gestión de cookies.	- Interfaz para crear, actualizar y eliminar sesiones de usuario. - Métodos para gestionar el ciclo de vida de la sesión y almacenar información asociada a la sesión.

Nota: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

Tabla 21. Diseño de Interfaces y Métodos

Componente	**Funcionalidades Clave**	**Interfaces y Métodos**
Gestión de Transacciones	- Coordinación de transacciones distribuidas.	- Interfaces para iniciar, confirmar o revertir transacciones distribuidas.
Gestión de Configuraciones	- Administración de configuraciones del sistema. - Almacenamiento y recuperación de configuraciones.	- Interfaces para leer y escribir configuraciones del sistema. - Métodos para gestionar la configuración del sistema.
Gestión de Logs	- Registro de eventos y actividades del sistema. - Diagnóstico de problemas y errores.	- Interfaces para escribir registros de eventos en archivos o bases de datos.

		- Métodos para analizar y consultar registros de eventos.
Gestión de Caché	- Almacenamiento en caché de datos para mejorar el rendimiento. - Gestión de la invalidación y actualización de la caché.	- Interfaces para acceder y manipular datos en la caché. - Métodos para configurar y gestionar la caché.

Nota: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

División en Módulos Funcionales

En la *tabla 22* se puede apreciar el contexto del sistema SIGEPAA y la agrupación de funciones relacionadas en módulos específicos, esto es fundamental para organizar y optimizar las operaciones clave asegurando que las actividades relacionadas con la autenticación, configuración del sistema, manipulación de datos y experiencia del usuario estén integradas de manera eficiente, facilitando así una gestión efectiva y una experiencia fluida para los usuarios del sistema.

Tabla 22. Agrupación de Funciones Relacionadas

Módulo Funcional	Descripción
Gestión de Usuarios	Agrupa las funcionalidades relacionadas con la autenticación, autorización y gestión de perfiles de usuario.
Administración de Configuraciones	Incluye las funcionalidades para la configuración del sistema, como definición de parámetros de aplicación, ajustes de seguridad y preferencias de usuario.

Gestión de Datos	Incluye todas las operaciones relacionadas con la manipulación de datos, como la administración de bases de datos y las operaciones CRUD (Crear, Leer, Actualizar, Eliminar).
Interfaz de Usuario	Contiene las funcionalidades para la interfaz de usuario, incluyendo presentación de información y gestión de eventos de usuario.

Nota: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

En la tabla 23 se detalla cómo la adaptación y personalización pueden ser aplicadas en el contexto del sistema SIGEPAA, enfocándose en mejorar el rendimiento, eficiencia y cumplimiento de requisitos específicos.

Tabla 23. Personalización y Configuración de Jakarta EE

Aspecto	Descripción
Adaptación de Tecnologías y Frameworks	Se realizan configuraciones específicas para optimizar el rendimiento y la eficiencia del sistema.
	Se ajustan parámetros de configuración de tecnologías como el servidor de aplicaciones y las bases de datos.
	Se realizan ajustes en los frameworks utilizados, como PrimeFaces y Hibernate, para mejorar su funcionamiento.
	Se implementan técnicas de caché y optimización de consultas para reducir los tiempos de respuesta.
Personalización de Funcionalidades	Se modifican las funcionalidades y comportamientos de tecnologías y frameworks para satisfacer los requisitos.

	Se desarrollan componentes personalizados para abordar necesidades específicas del sistema.
	Se implementan reglas de negocio personalizadas utilizando EJBs y CDI beans para adaptarse a los requerimientos.
	Se optimizan los algoritmos y procesos de negocio para mejorar el rendimiento y la escalabilidad del sistema.

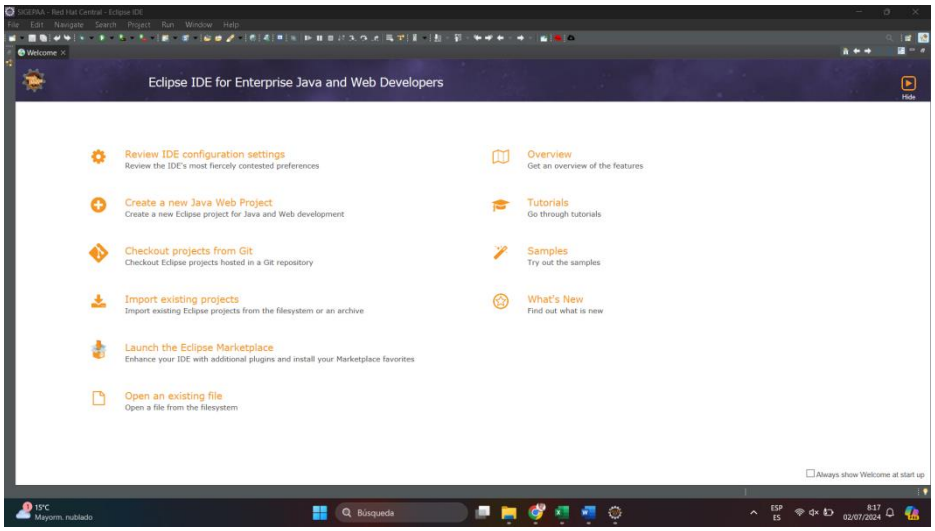
Nota: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

2.3. Fase 3 Implementación

El proceso de migración, detallado en la fase uno, comenzó con la actualización de varias herramientas clave. Se realizó la transición de JDK 1.8.0_241 a JDK 21, proporcionando mejoras significativas en rendimiento y seguridad al sistema web SIGEPAA. Asimismo, se actualizó Eclipse IDE de la versión 2020-03 a la versión 2024-06, optimizando el entorno de desarrollo con nuevas características y mejor soporte para Jakarta EE. Finalmente, el servidor WildFly se actualizó de la versión 18.0.0.Final a la versión 30.0.0.Final, lo que permitió aprovechar las mejoras en la gestión y ejecución de aplicaciones empresariales. Estas actualizaciones fueron esenciales para garantizar la compatibilidad y optimización del sistema SIGEPAA con Jakarta EE 10.

Para iniciar un nuevo proyecto con Jakarta EE en Eclipse IDE 2024, es fundamental configurar correctamente el entorno de desarrollo. Como se muestra en la Figura 6 Primero, asegúrese de que Eclipse IDE 2024 esté instalado y actualizado a la versión más reciente.

Figura 6 Entorno de desarrollo



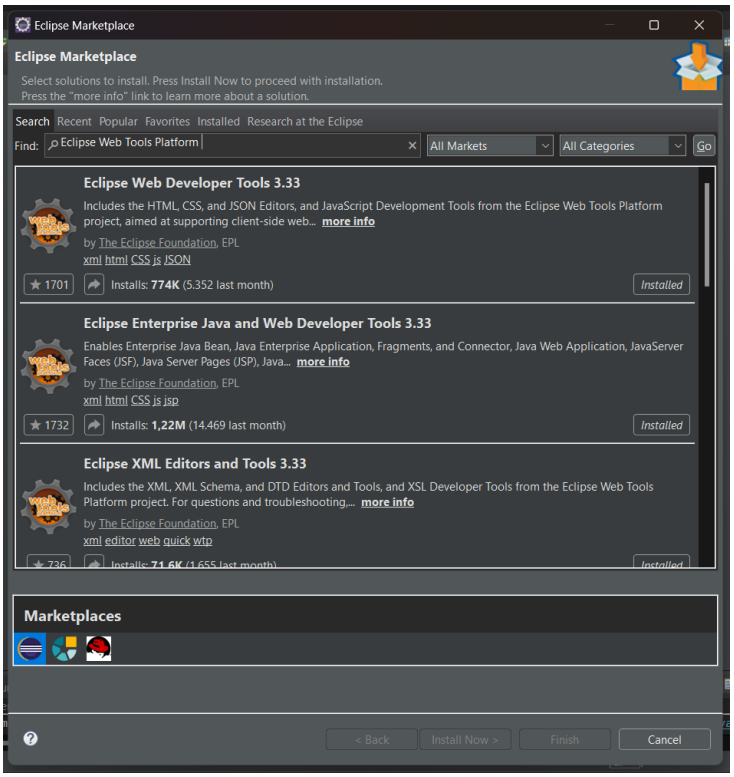
Luego, procedemos a instalar los plugins necesarios para Jakarta EE. Estos plugins se pueden encontrar en el marketplace de Eclipse, lo cual facilita la integración y el soporte de las especificaciones de Jakarta EE dentro del entorno de desarrollo.

- **Eclipse IDE for Enterprise Java Developers:** La instalación como se muestra en la Figura 7 de este paquete incluye la mayoría de las herramientas necesarias para el desarrollo de aplicaciones Java EE/Jakarta EE, incluyendo herramientas para JSP, JPA, EJB, y Servlets .

Figura 7 Herramientas para el desarrollo de aplicaciones

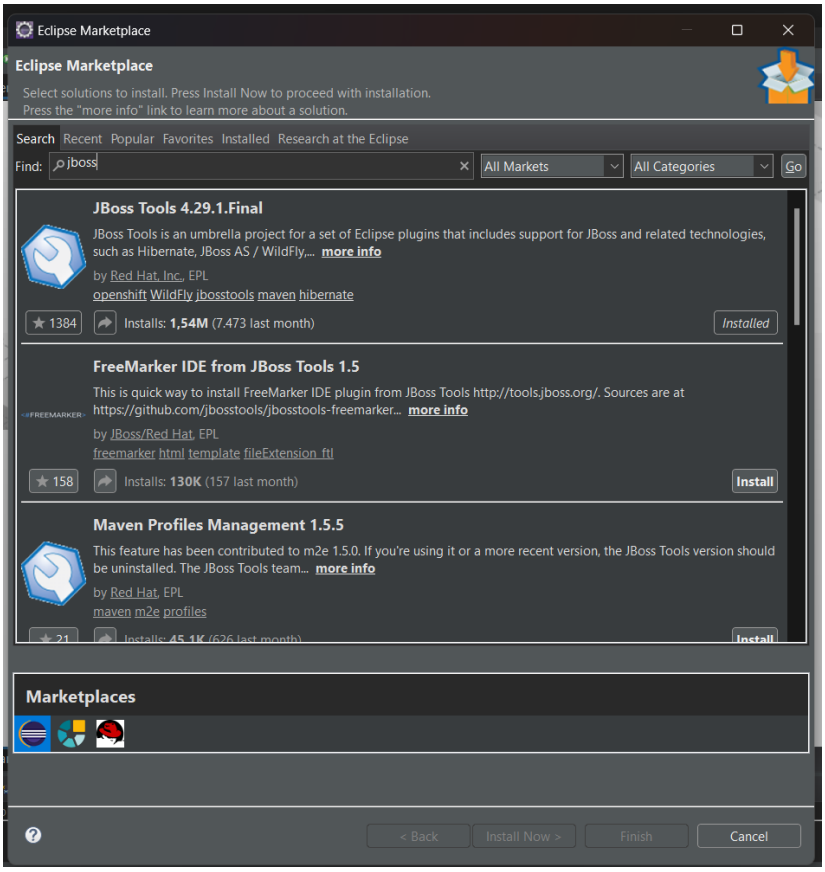
- **Eclipse Web Tools Platform (WTP)**: Como se muestra en la Figura 8, la herramienta Proporciona un entorno para el desarrollo de aplicaciones web y servicios web, incluyendo soporte para HTML, CSS, JavaScript, y otros lenguajes web.

Figura 8 Eclipse Web Tools Platform



- **Jakarta EE Tools**: Herramientas específicas para trabajar con las especificaciones de Jakarta EE, incluyendo JPA, EJB, y CDI.
- **JBoss Tools**: Como se muestra la herramienta en la Figura 9 JBoss Tools proporciona un conjunto de herramientas para desarrolladores que utilizan servidores de aplicaciones JBoss/WildFly. Incluye soporte para desarrollo y despliegue de aplicaciones Jakarta EE.

Figura 9 Herramientas de desarrollo



La creación de un proyecto utilizando Enterprise Application Project en Eclipse IDE para desarrollar aplicaciones Jakarta EE involucra varios pasos para configurar correctamente el

entorno de desarrollo y preparar las estructuras del proyecto necesarias como se establece a continuación:

Creación de un Proyecto mediante Enterprise Application Project en Eclipse IDE 2024

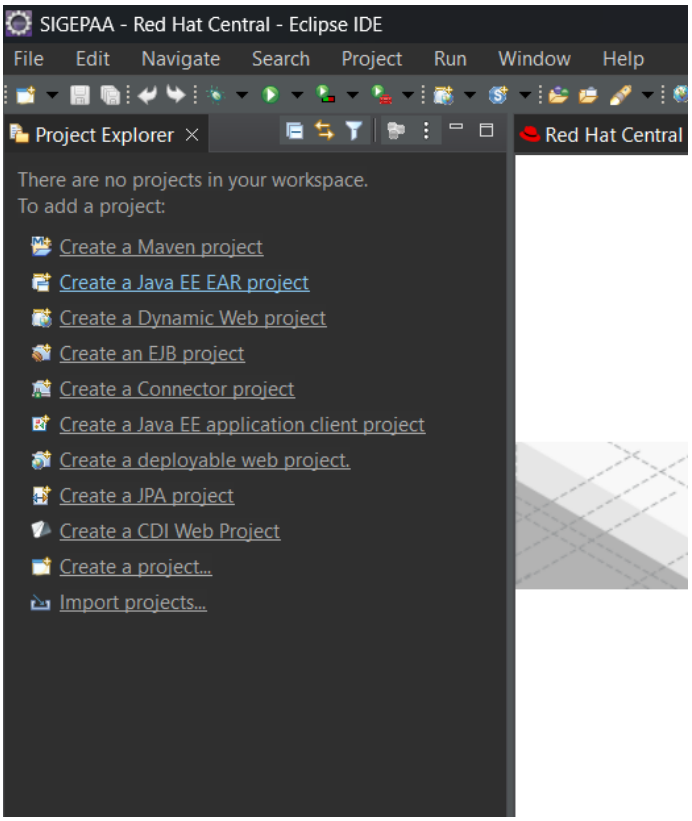
Paso 1: Abrir Eclipse IDE

1. Inicia Eclipse IDE.
2. Selecciona un espacio de trabajo donde almacenarás tu proyecto.

Paso 2: Crear un Nuevo Proyecto

1. Ve a File > New > Other....
2. En el cuadro de diálogo, expande Java EE (o Jakarta EE) y selecciona Enterprise Application Project. Haz clic en Next.

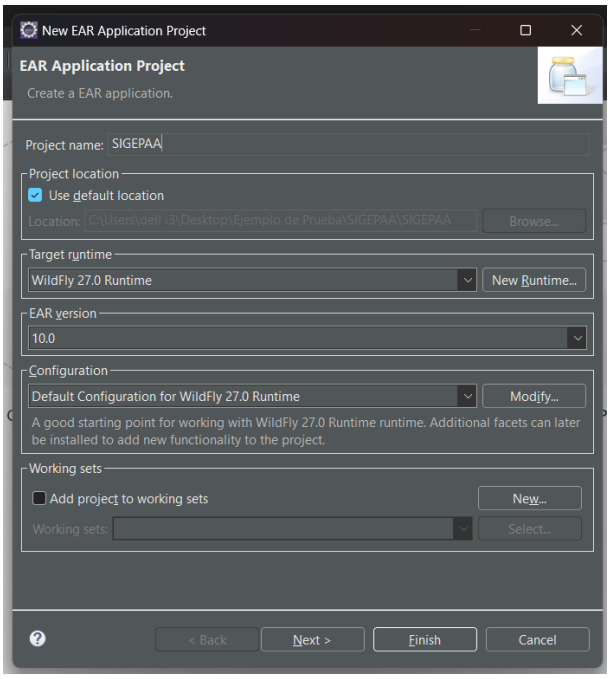
Figura 10 Cuadro de dialogo (Creación de proyecto)



Paso 3: Configurar el Proyecto

1. **Nombre del Proyecto:** Introduce un nombre para tu proyecto.
2. **Target Runtime:** Selecciona el servidor de aplicaciones que deseas utilizar (por ejemplo, WildFly). Si no has configurado un servidor aún, puedes hacerlo en este paso.
3. **Configuration:** Como se observa en la Figura 11 se puede Seleccionar Default Configuration para [tu servidor].

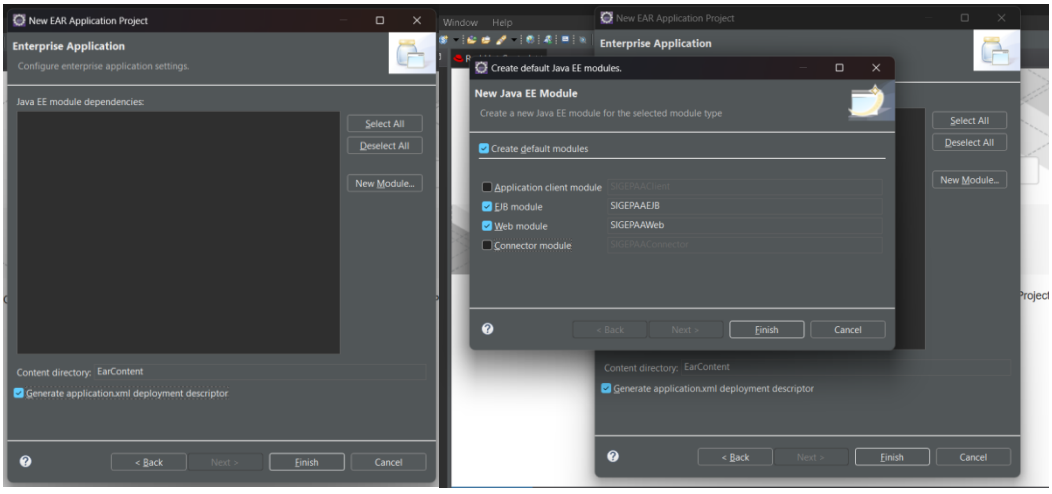
Figura 11 Sección de configuraciones



Paso 4: Módulos del Proyecto

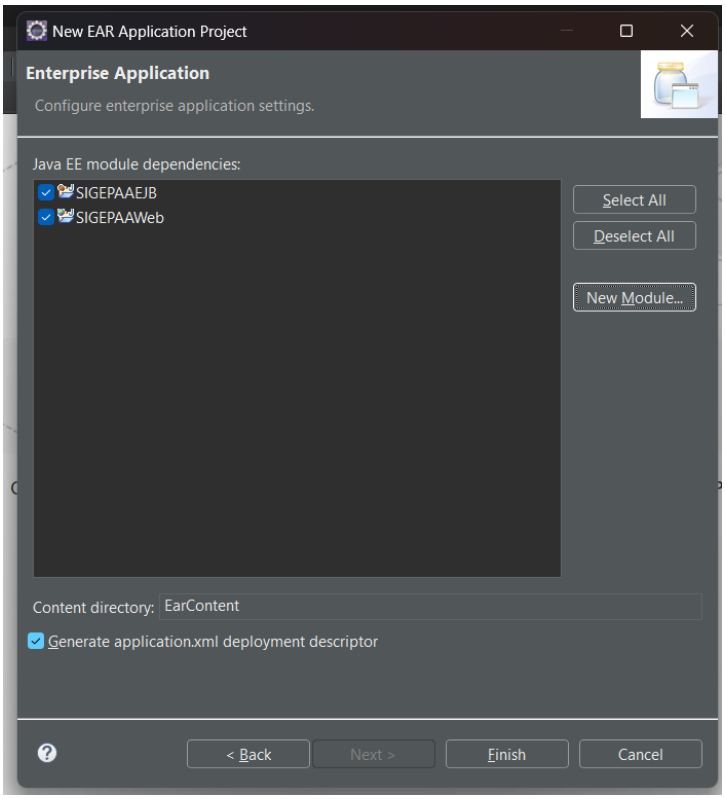
1. **Project Facets:** Asegúrate de que las facetas de EJB Module y Web Module están seleccionadas.

Figura 12 Selección de facetas



2. **EAR Project:** Como se muestra en la Figura 13. El asistente creará un archivo EAR que empaquetará los módulos EJB y Web.

Figura 13 Creación de archivo EAR (asistente)



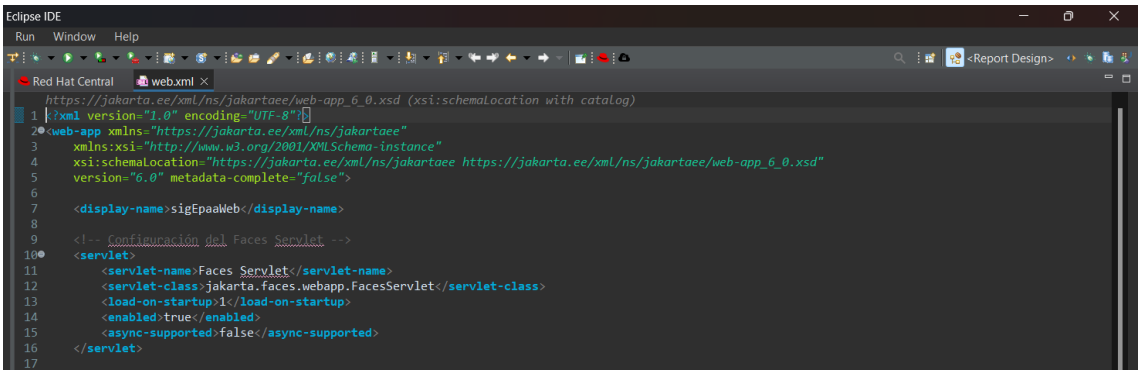
3. Haz clic en Finish.

Paso 5: Configurar los Módulos

1. **Configuración del módulo Web:**

   - **Dynamic Web Module Version:** Selecciona la versión que soporta Jakarta EE 10 (generalmente versión 5.0 o en este caso la versión 6.0).
   - **Context Root:** Define el contexto raíz para la aplicación web.
   - **Content Directory:** Deja la configuración por defecto o cámbiala según tus necesidades.

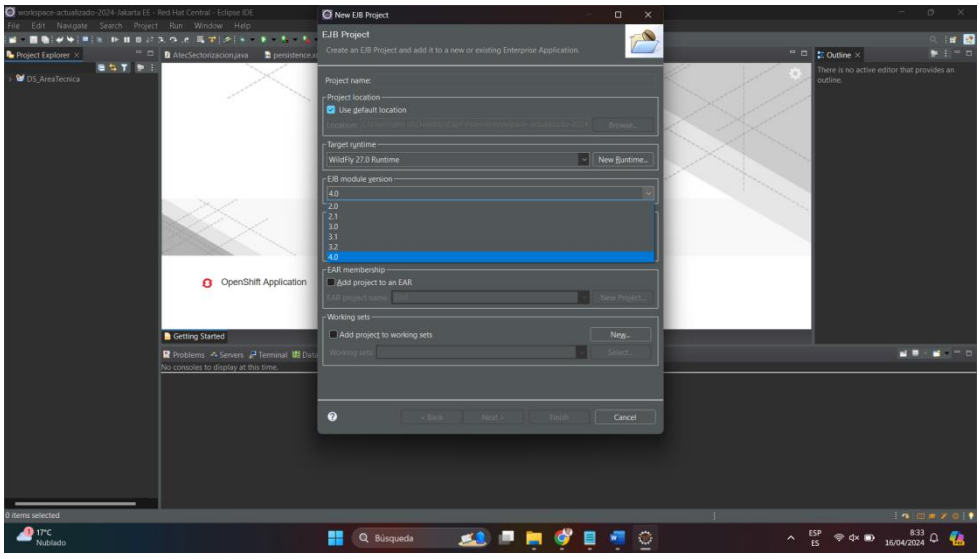
Figura 14 Programador de módulos



2. **Configuración del módulo EJB:**

- EJB Module Version: Asegúrate de seleccionar una versión compatible con Jakarta EE 10.

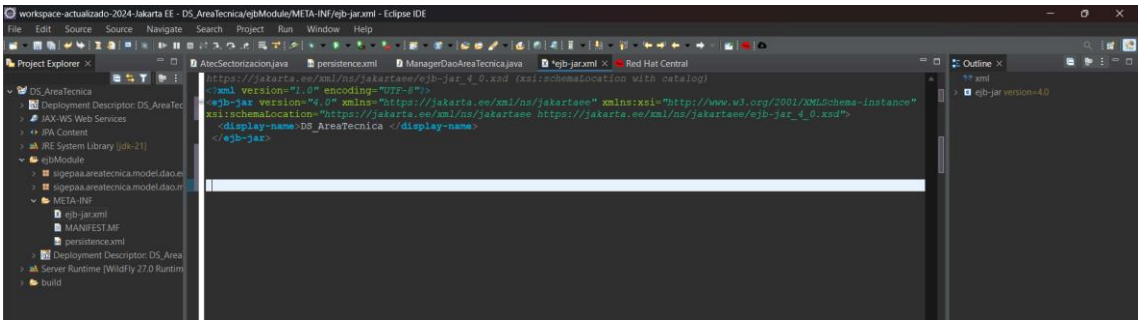
Figura 15 Configuración de módulos



3. **Configuración del módulo de Aplicación:**

- Este módulo contiene la configuración global y el descriptor de despliegue para el proyecto.

Figura 16 Identificador Jakarta EE



Paso 6: Finalizar la Creación del Proyecto

1. Revisa la configuración del proyecto en la pantalla de resumen.
2. Haz clic en Finish para crear el proyecto.

Paso 7: Configurar el Servidor de Aplicaciones

1. **Configurar Servidor:** Si no lo has hecho anteriormente, agrega y configura el servidor (WildFly 30) en Eclipse.
    - Selección del Servidor en este caso la actualización de WildFly versión 30.0.0

Figura 17 Configuración de servidores para selección

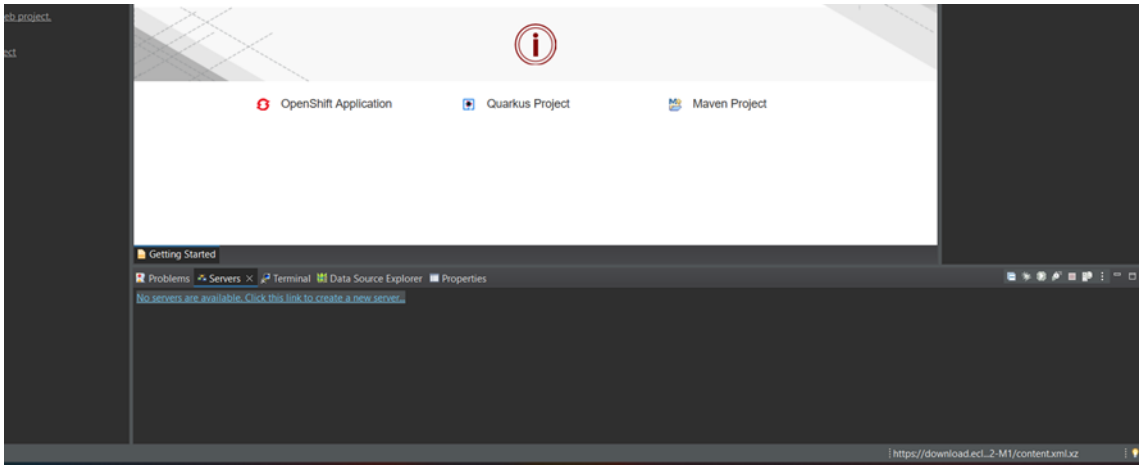


Figura 18 Selección de servidor

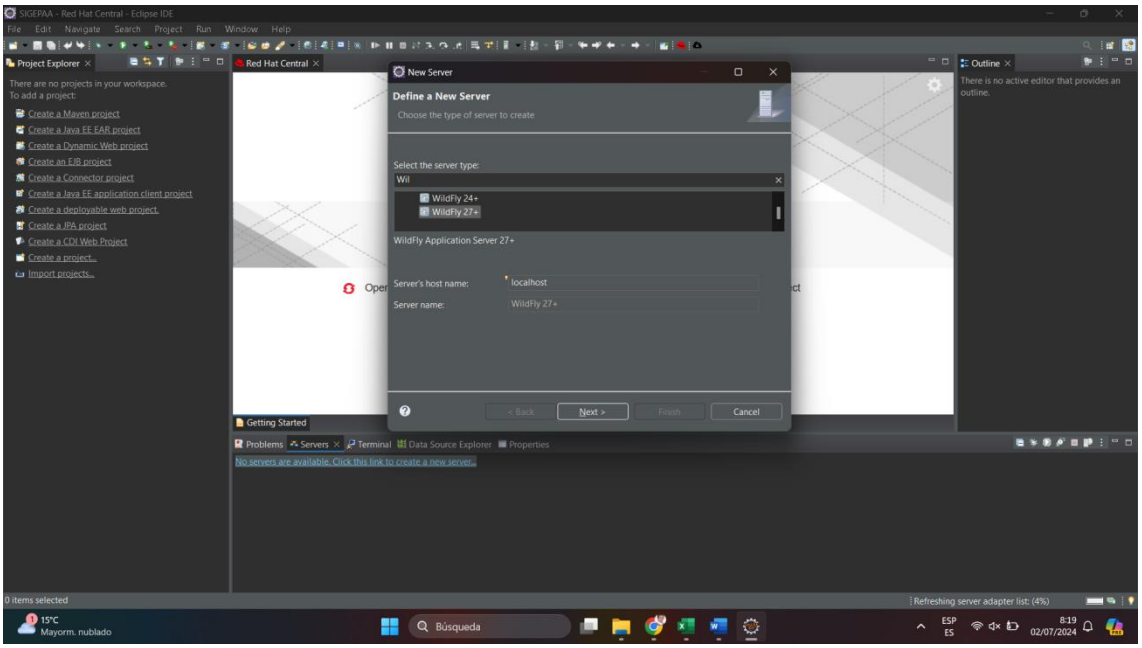


Figura 19 Proceso 2 de selección de servidor

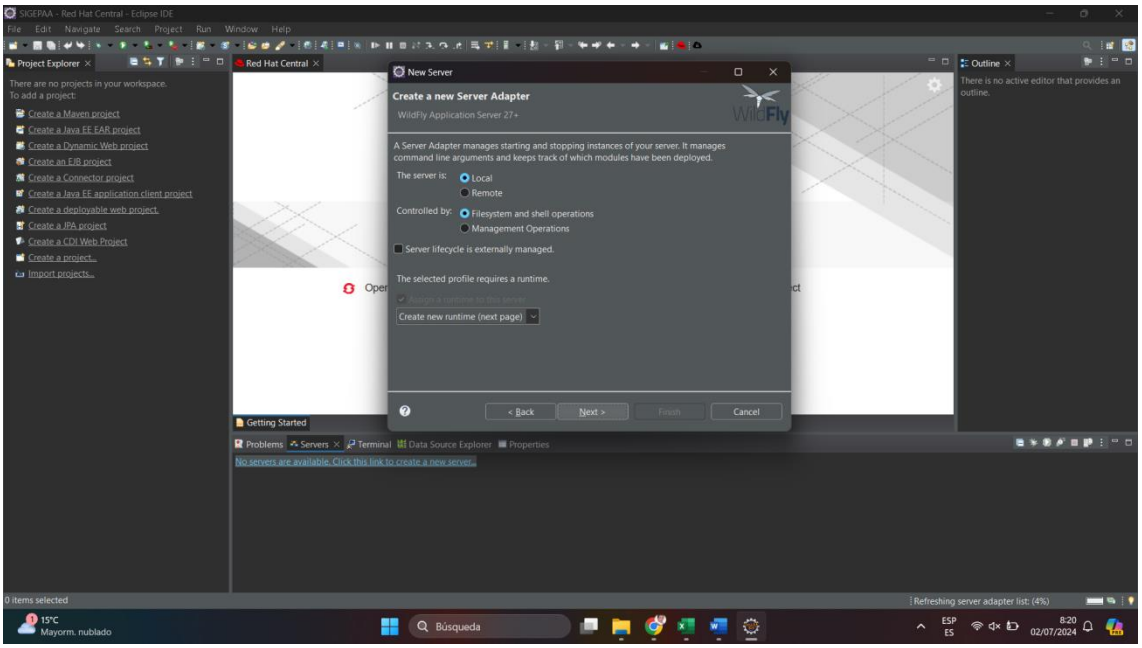
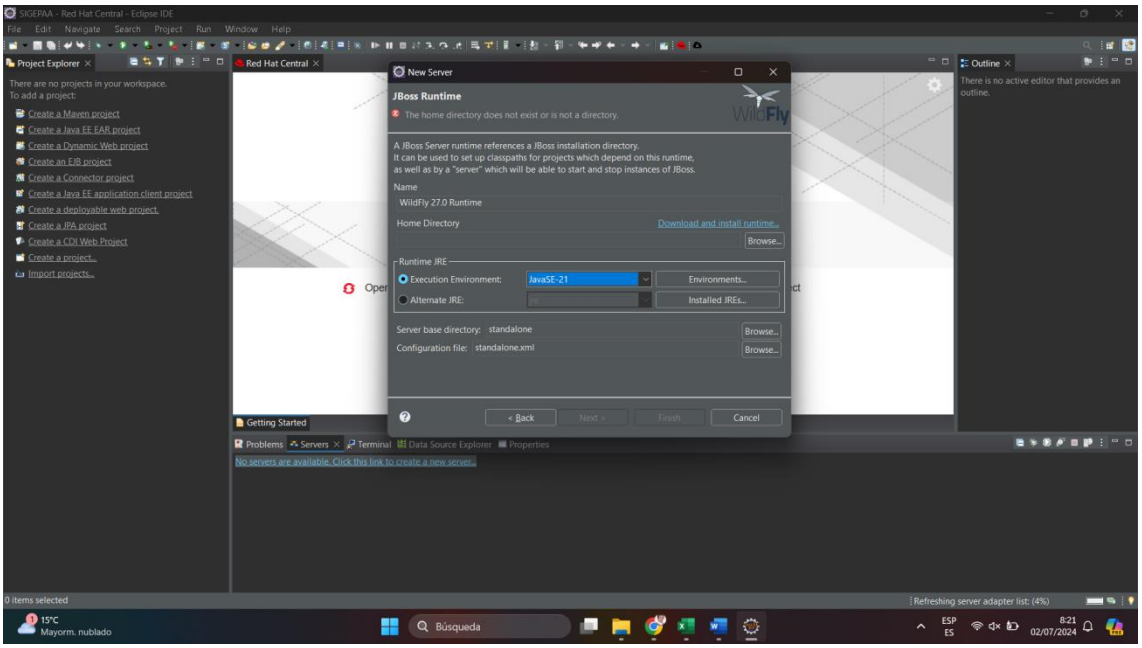


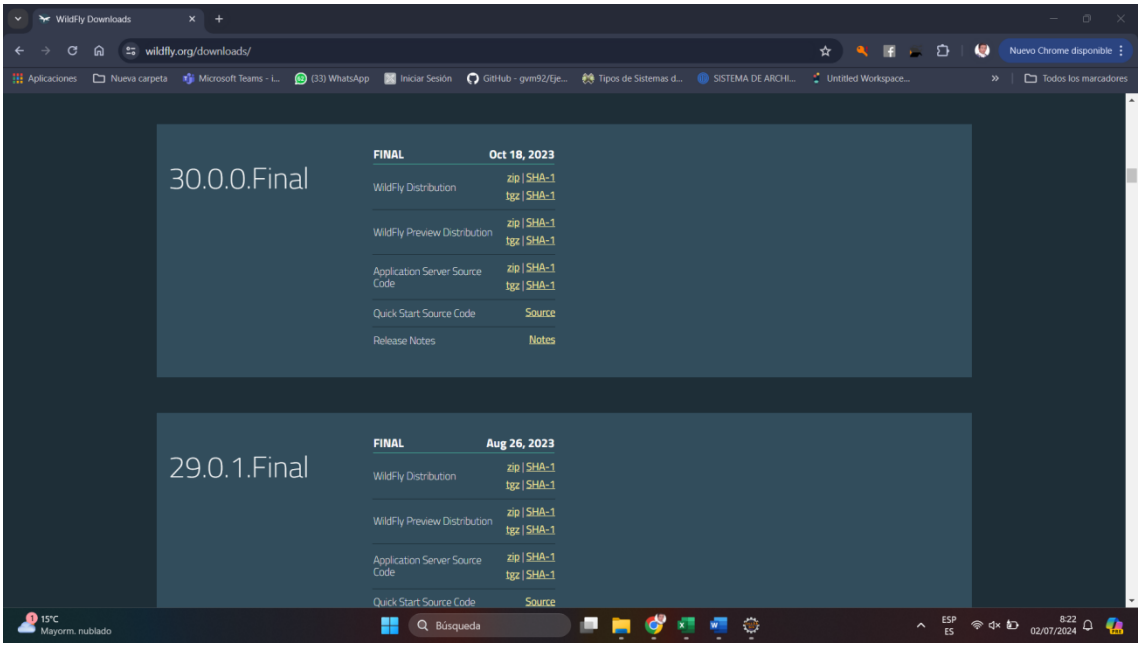
Figura 20 Proceso 3 de selección de servidor



Descarga de WildFly 30.0.0

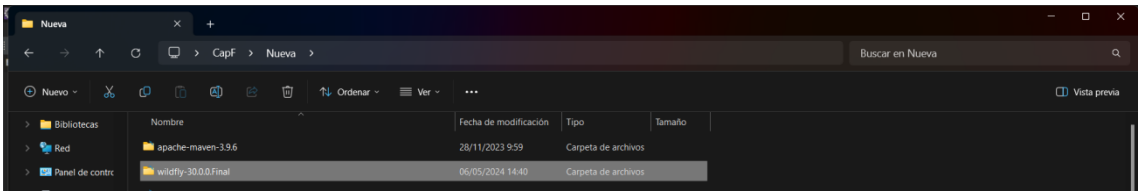
(https://www.wildfly.org/downloads/)

Figura 21 Descarga de WildFly 30.0.0



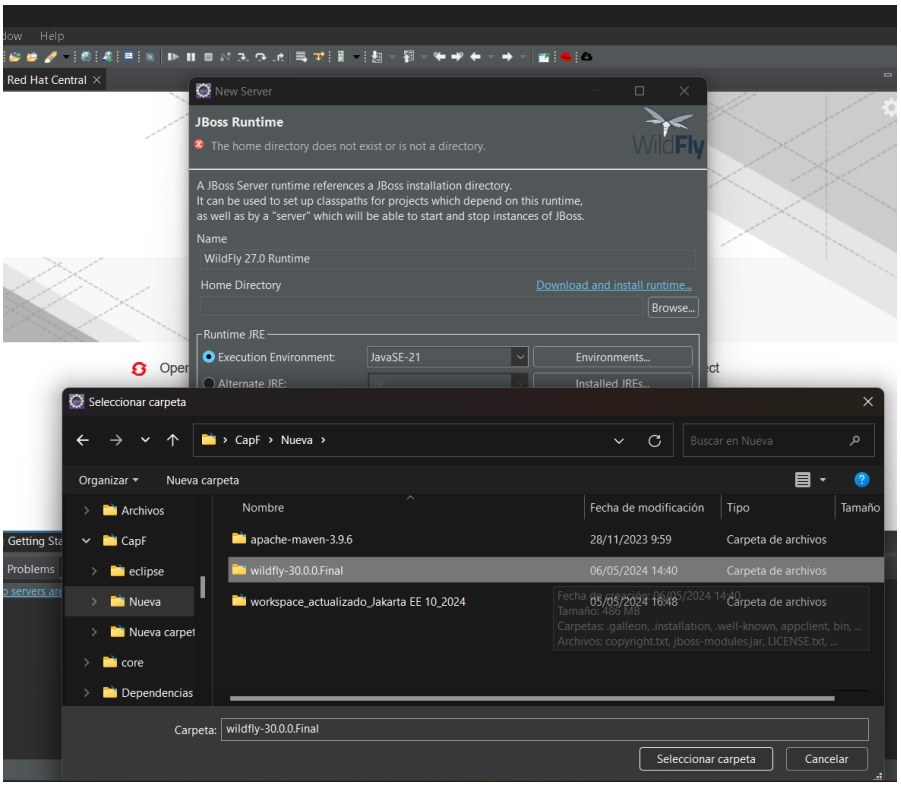
Identificación de la carpeta de descarga para luego proceder a cargar wildFly

Figura 22 Identificador de la carpeta descargada



Cargar WildFly el cual descarga desde el repositorio oficial de WildFly.

Figura 23 Carga WildFly desde el repositorio



WildFly 30.0.0 Cargado para el servidor

Figura 24 Comprobación de Servidor

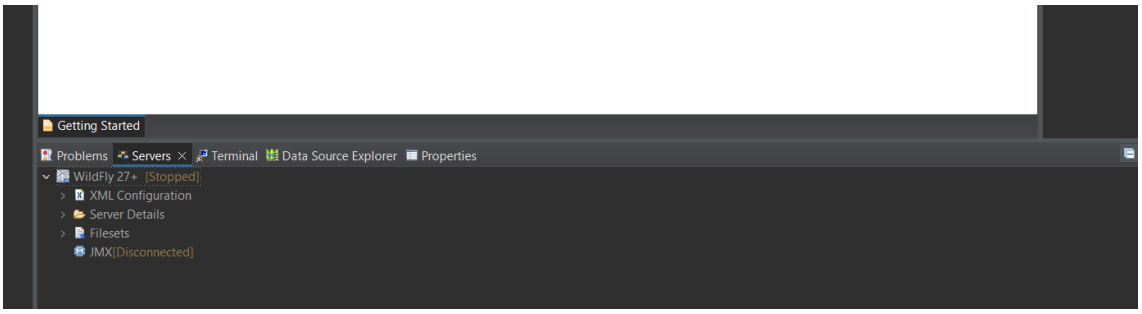
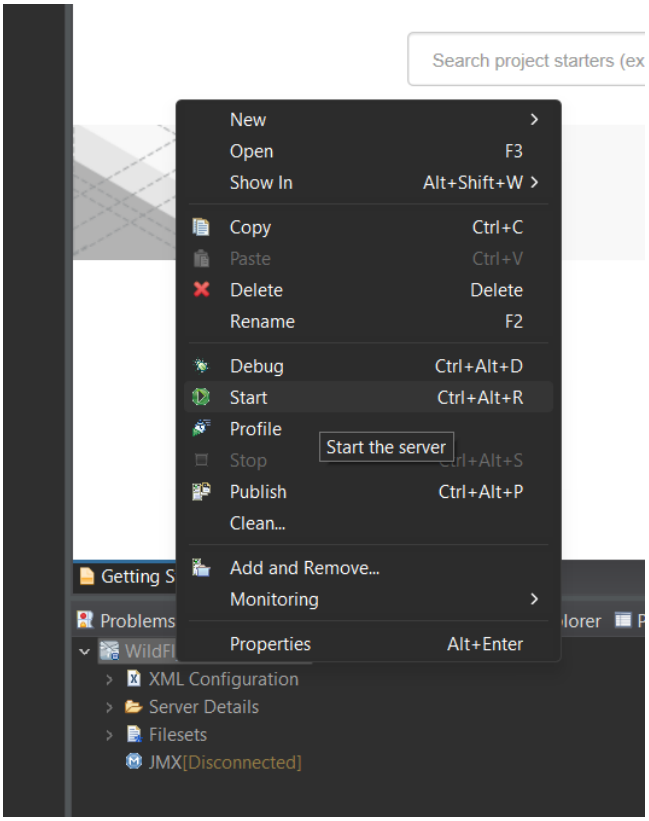
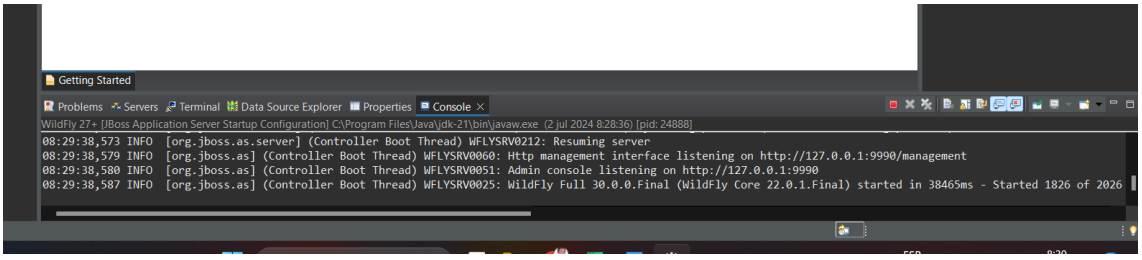


Figura 25 Ejecución de WildFly



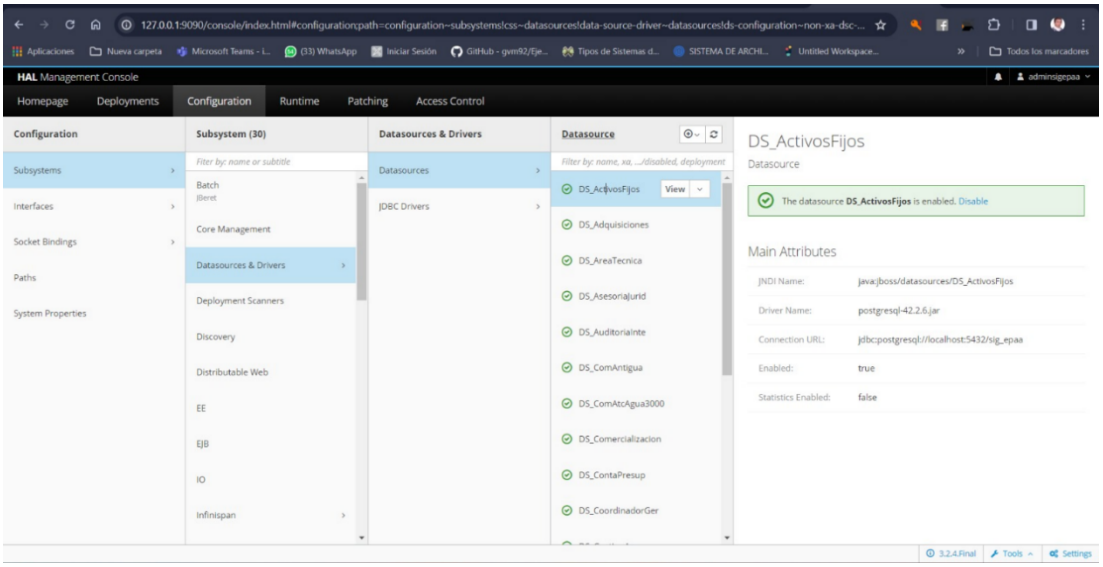


Dominio Local (Localhost http://127.0.0.1:9990)

-Agregar las respectivas conexiones a la Base de Datos BDD

Durante la actualización de WildFly, se establecieron y configuraron las conexiones necesarias a la base de datos para asegurar un rendimiento óptimo del sistema. Esto implicó ajustar parámetros clave, validar conexiones y optimizar el rendimiento para garantizar un acceso eficiente y seguro a los datos.

Figura 26 Base de datos para rendimiento óptimo del sistema



2. **Despliegue**: Asegúrate de que tu servidor está configurado correctamente para desplegar y ejecutar tu aplicación.

Paso 8: Desarrollo del Proyecto

1. **Estructura del Proyecto**:

- El proyecto se estructurará en módulos como EAR, Web y EJB.
- El módulo EAR contendrá los archivos.ear y la configuración de despliegue.
- El módulo Web contendrá la estructura estándar de una aplicación web (WEB-INF, web.xml, etc.).
- El módulo EJB contendrá los beans de negocio y su configuración.

2. **Desarrollo y Configuración**:

   - Agrega tus clases, servlets, beans y otros componentes necesarios en los módulos correspondientes.
   - Configura los archivos de despliegue (web.xml, ejb-jar.xml, application.xml, etc.) según sea necesario.

3. **Prueba y Despliegue**:

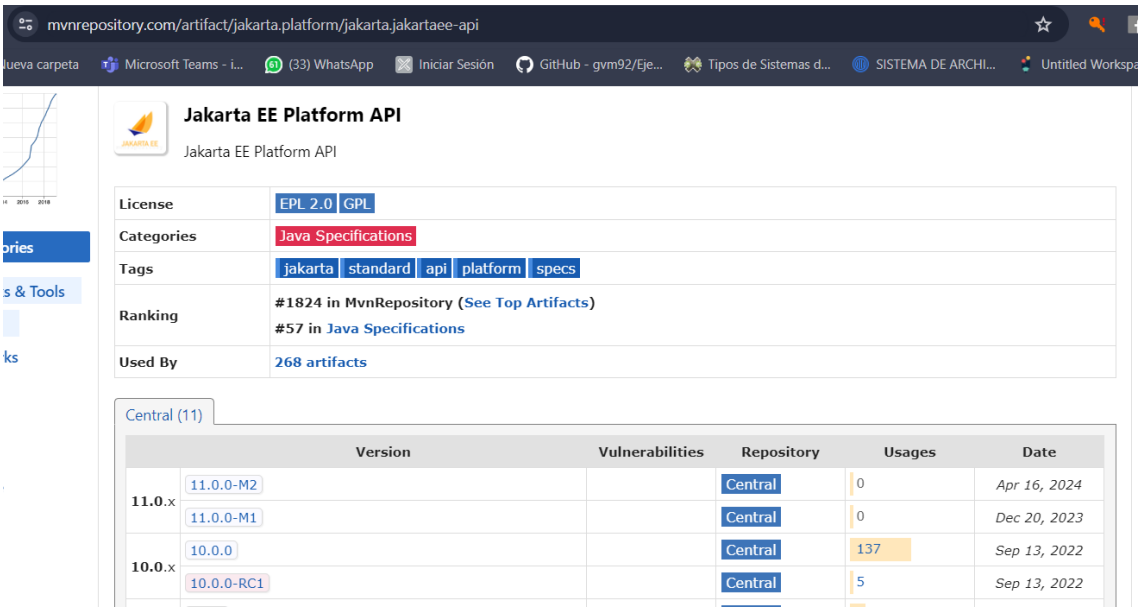
   - Despliega tu aplicación en el servidor configurado.
   - Prueba las funcionalidades desarrolladas para asegurarte de que todo está funcionando correctamente.

Actualización de Dependencias

Se agregan las respectivas dependencias de acorde al proyecto a realizar las cuales se descargan desde un repositorio seguro en este caso el repositorio Maven (https://mvnrepository.com/).

La selección de versiones se aplica según las más usadas actualmente como se muestra en la imagen 27

Figura 27 Verificación de la versión en la aplicación



Nota. Por MVN Repository, 2006.

En la *tabla 24* se destaca los beneficios clave que se obtienen al mantener actualizadas las dependencias del proyecto, mejorando tanto la seguridad como la funcionalidad del sistema.

Tabla 24. Veneficios de Jakarta EE con base a las respectivas APIs

Jakarta EE API	Dependencia	Versión	Descripción
Jakarta EE API	jakarta.jakartaee-api	10.0.0	Proporciona todas las APIs centrales de Jakarta EE 10 en un solo JAR.
Activación y Anotaciones	jakarta.activation-api	2.1.2	Proporciona soporte para la activación de servicios y anotaciones en Jakarta EE.
	jakarta.annotation-api	3.0.0	Define las anotaciones comunes utilizadas en Jakarta EE.

Batch y CDI	jakarta.batch-api	2.1.1	API para el procesamiento por lotes en aplicaciones Jakarta EE.
EJB y EL	jakarta.ejb-api	4.0.1	Proporciona la API para Enterprise JavaBeans en Jakarta EE.
	jakarta.el-api	6.0.0	Proporciona la API de Expression Language utilizada en Jakarta EE.
JSF y JSON	jakarta.faces-api	4.0.1	Proporciona la API para Jakarta Server Faces.
	jakarta.json-api	2.1.1	Define la API para el procesamiento de JSON en Jakarta EE.
	jakarta.json.bind-api	3.0.0	Proporciona la API para la vinculación de datos JSON en Jakarta EE.
JMS y Mail	jakarta.jms-api	3.1.0	Proporciona la API para el servicio de mensajería Java en Jakarta EE.
	jakarta.mail-api	2.1.0	Define la API para el manejo de correos electrónicos en Jakarta EE.
Persistencia y Resource	jakarta.persistence-api	3.1.0	Proporciona la API para la persistencia de datos con JPA en Jakarta EE.

	jakarta.resource-api	2.1.0	Define la API para el manejo de recursos en Jakarta EE.
Seguridad y Servlet	jakarta.security.enterprise-api	3.0.0	Proporciona la API para la seguridad empresarial en Jakarta EE.
	jakarta.servlet-api	6.0.1	Define la API para la gestión de servlets en Jakarta EE.
Transacción y Validación	jakarta.transaction-api	2.1.0	Proporciona la API para la gestión de transacciones en Jakarta EE.
	jakarta.validation-api	3.1.0	Define la API para la validación de datos en Jakarta EE.
WS-RS y XML Binding/SOAP	jakarta.ws.rs-api	3.1.1	Proporciona la API para los servicios web RESTful en Jakarta EE.
	jakarta.xml.bind-api	4.1.0	Define la API para la vinculación de datos XML en Jakarta EE.
	jakarta.xml.soap-api	2.1.0	Proporciona la API para los servicios web SOAP en Jakarta EE.
	jakarta.xml.ws-api	4.1.0	Define la API para los servicios web XML en Jakarta EE.

Implementaciones y Utilidades	hibernate-core	6.5.0.Final	Proporciona el núcleo del framework Hibernate para JPA en Jakarta EE.
	weld-core	5.1.2. Final	Proporciona la implementación del framework Weld para CDI en Jakarta EE.
Jackson	jackson-core	2.17.0	Proporciona la biblioteca central de Jackson para el procesamiento de JSON.
	jackson-databind	2.17.0	Define la biblioteca de vinculación de datos de Jackson para JSON.
	jackson-annotations	2.17.0	Proporciona las anotaciones de Jackson para el procesamiento de JSON.
Logging	log4j-api	2.23.1	Define la API de Log4j para el registro y monitoreo de aplicaciones en Jakarta EE.
	log4j-core	2.23.1	Proporciona la implementación central de Log4j para el registro y monitoreo de aplicaciones.

Nota: Elaborado por: Narváez Esteban, basado en (Características Generales de Jakarta EnterPrise, 2018)

Migración de Código:

Modificar el código existente para hacer uso de las nuevas características y APIs introducidas en Jakarta EE 10

1. **Identificar las partes del código que requieren actualización**:

   - **Clases de Servlets**: Revisar todas las clases que implementan javax.servlet y actualizarlas a jakarta.servlet como se muestra en la imagen #.
   - **EJBs (Enterprise JavaBeans)**: Identificar y actualizar todas las clases que usan javax.ejb a jakarta.ejb como se muestra en la imagen #.
   - **Managed Beans**: Actualizar las clases anotadas con javax.faces.bean a jakarta.faces.bean como se muestra en la imagen #.

Figura 28 Verificación de actualización con JAVA EE

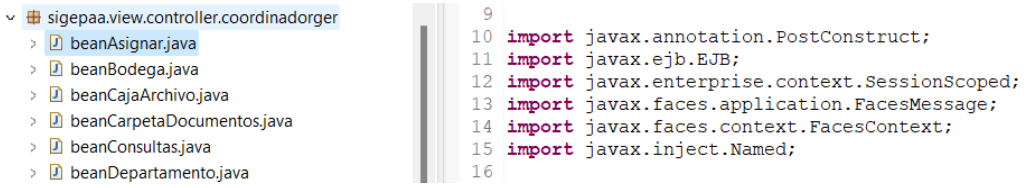


Figura 29 Verificación de actualización con JAKARTA EE

- **Entidades JPA (Java Persistence API)**: Cambiar las importaciones de javax.persistence a jakarta.persistence como se muestra en la imagen.

Figura 30 Importaciones con JAVA EE

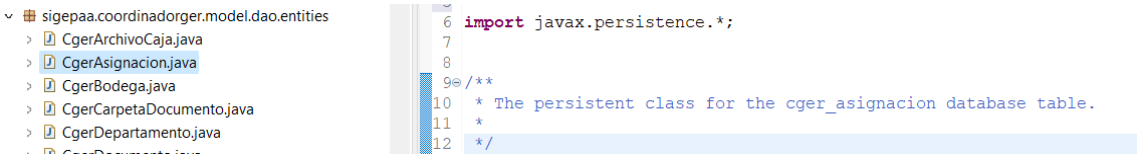
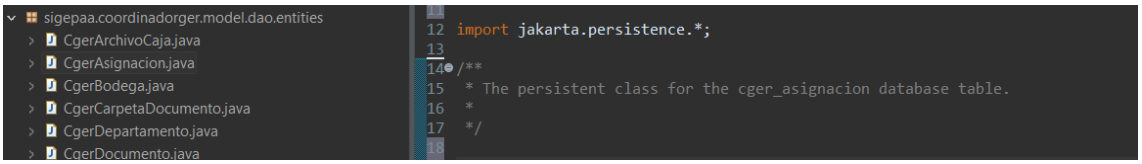


Figura 31 Importaciones con Jakarta EE



Revisión de la documentación de Jakarta EE 10 para comprender las API y características nuevas o modificadas:

- **Documentación oficial**: Se consultará la documentación oficial de Jakarta EE 10 para obtener detalles sobre las nuevas características y cambios en las API. Esto asegurará una implementación precisa y conforme a las especificaciones.
- **Notas de versión**: Las notas de versión de Jakarta EE 10 serán revisadas para entender las mejoras y correcciones de errores introducidas, lo cual es esencial para un funcionamiento óptimo del sistema SIGEPAA.

Actualización del código existente según las mejores prácticas y estándares de Jakarta EE 10:

- **Refactorización de clases**: Se llevará a cabo una revisión exhaustiva de todas las clases y paquetes en SIGEPAA para garantizar que cumplan con las convenciones de nomenclatura y las mejores prácticas recomendadas por Jakarta EE 10. Esto incluirá la reorganización de paquetes y la mejora de la estructura del código.
- **Optimización del código**: El código será revisado y optimizado para mejorar tanto el rendimiento como la eficiencia, aprovechando las mejoras introducidas en el compilador y la ejecución de Jakarta EE 10. Esto abarca la optimización de consultas a bases de datos y la eliminación de código obsoleto.

Ejecutar pruebas de regresión para garantizar que las modificaciones realizadas no introduzcan errores en el sistema.

- **Ejecución de pruebas de integración**: Se realizarán pruebas de integración para verificar que los distintos módulos del sistema funcionen de manera correcta y coherente entre sí, garantizando la integridad del sistema en su totalidad.
- **Pruebas de sistema**: Se ejecutarán pruebas completas del sistema para confirmar que todas las funcionalidades de SIGEPAA operen según lo esperado. Estas pruebas incluirán la validación de flujos de trabajo completos y la interacción de los usuarios finales.

Refactorización:

Realizar cambios en el código para mejorar su estructura y eficiencia, si es necesario.

Analizar el código existente para identificar áreas que puedan beneficiarse de la refactorización, como la reorganización de paquetes o la optimización de las consultas a bases de datos para mejorar el rendimiento y la eficiencia del sistema.

Aplicación de Técnicas de Refactorización

Mejorar la legibilidad:

1. **Renombrar variables y métodos**:

   Utilizar nombres descriptivos y consistentes para mejorar la comprensión del código.

2. **Documentar el código**:

Añadir comentarios y documentación para explicar la lógica compleja y el propósito de las clases y métodos.

Mejorar la mantenibilidad:

1. **Dividir clases grandes**:

   Separar clases monolíticas en varias clases más pequeñas con responsabilidades específicas.

2. **Eliminar dependencias innecesarias**:

   Reducir las dependencias entre clases y paquetes para mejorar la modularidad.

Mejorar el rendimiento:

1. **Optimización de bucles y condiciones**:

   Revisar y optimizar los bucles y las condiciones para minimizar el tiempo de ejecución.

2. **Uso eficiente de recursos**:

   Implementar patrones de diseño que optimicen el uso de memoria y CPU, como Singleton y Factory.

Integración Continua

La implementación de la integración continua en SIGEPAA no solo mejorará la calidad y estabilidad del sistema, sino que también aumentará la eficiencia del equipo de desarrollo, permitiendo una entrega más rápida y segura de nuevas funcionalidades. Esto garantizará que

SIGEPAA continúe siendo una herramienta confiable y eficiente para la gestión de procesos en la empresa EPAAA.

Utilizar herramientas de integración continua como GitLab.

Configurar scripts de compilación y pruebas automatizadas para ejecutarse cada vez que se realicen cambios en el repositorio de código.

Implementar flujos de trabajo que incluyan la ejecución de pruebas unitarias, pruebas de integración y análisis estático de código.

Desplegar de manera automática el código actualizado en entornos de prueba una vez que haya superado exitosamente las pruebas automatizadas.

Despliegue en Entornos de Prueba:

Desplegar el sistema actualizado en entornos de prueba para realizar pruebas de integración y validar su funcionamiento en un ambiente controlado.

Configurar entornos de prueba que reproduzcan fielmente el entorno de producción.

Desplegar la aplicación actualizada en los entornos de prueba utilizando herramientas de despliegue automatizado.

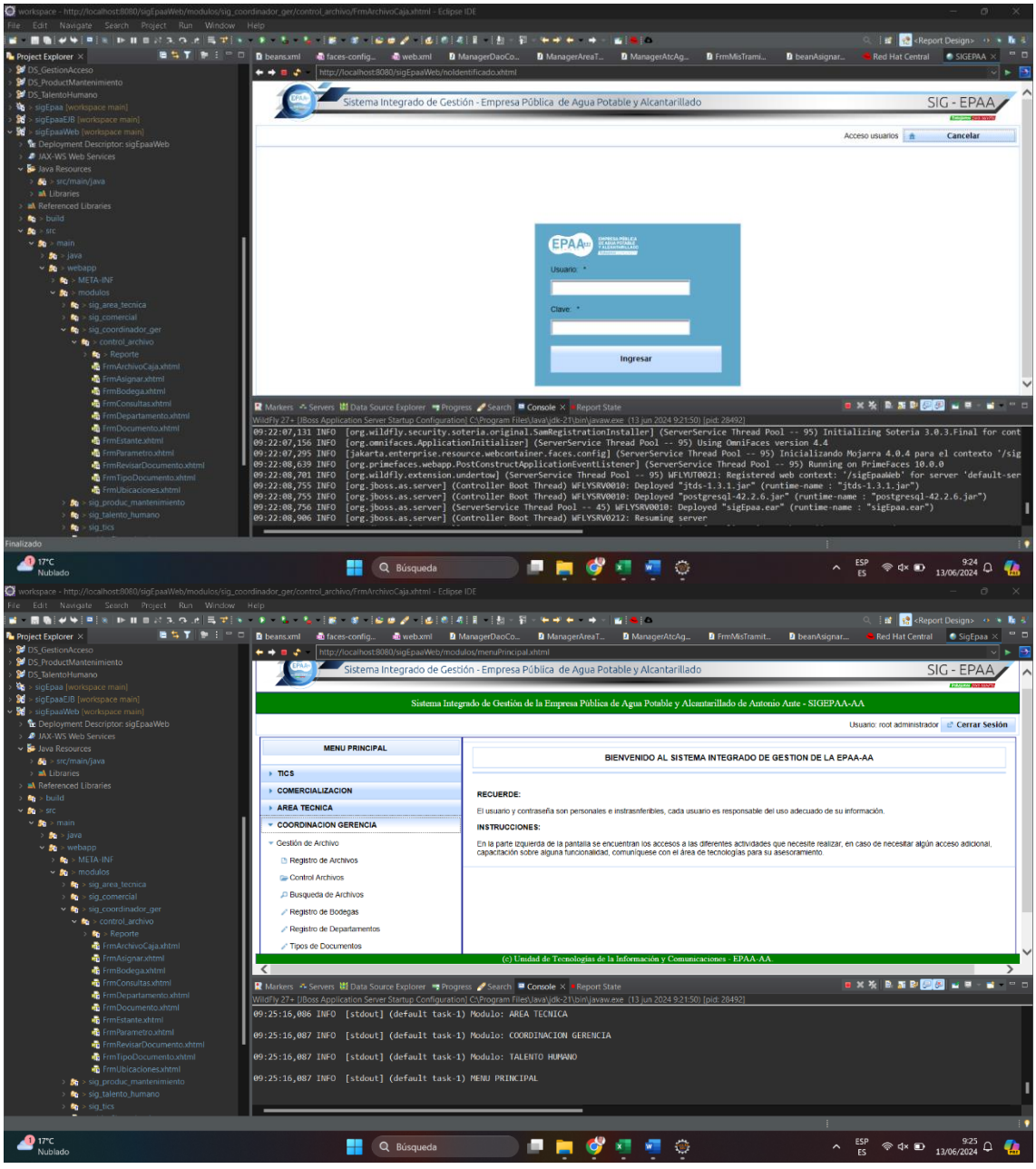
Llevar a cabo pruebas de integración para comprobar la interoperabilidad entre los distintos componentes del sistema y su integración con otros sistemas.

Ejecución localhost del sistema SIGEPAA actualizado a Jakarta EE

La actualización de SIGEPAA a Jakarta EE 10 marca un hito importante en la evolución del sistema, garantizando que EPAAA dispone de una herramienta tecnológica avanzada, segura

y eficiente para la gestión de sus operaciones. Con esta actualización, SIGEPAA no solo se mantiene al día con las últimas tendencias tecnológicas, sino que también se posiciona como un sistema preparado para afrontar los desafíos futuros, mejorando continuamente el servicio ofrecido a sus usuarios.

Figura 32 Ejecución localhost del sistema



2.4. Fase 4 Verificación

Durante la fase de pruebas de integración, se llevan a cabo pruebas exhaustivas para verificar tanto la funcionalidad como la estabilidad de la aplicación migrada en el entorno de Jakarta EE. Esto incluye la ejecución de pruebas unitarias y de integración, así como pruebas de rendimiento para evaluar el rendimiento y la escalabilidad de la aplicación bajo diferentes condiciones de carga.

Actividades Clave

2.4.1. Pruebas Unitarias y de Integración

Se desarrollaron y realizaron pruebas unitarias e integrales para verificar la funcionalidad de los componentes individuales y asegurar su correcta integración en la aplicación. Estas pruebas mejoraron y aceleraron el proceso de verificación, garantizando que la migración de Java EE a Jakarta EE mantuviera el rendimiento y la funcionalidad del sistema ya existente.

2.4.2. Pruebas de Rendimiento

Se llevan a cabo pruebas de rendimiento para evaluar el tiempo de respuesta del servidor y la escalabilidad de la aplicación bajo diversas condiciones de carga. Se identifican y corrigen posibles cuellos de botella, asegurando así un rendimiento óptimo

Es importante realizar pruebas de integración en la aplicación actual con las versiones más recientes de Java EE. Esto implica revisar las APIs y funcionalidades que se utilizan en la aplicación y verificar si hay cambios o deprecaciones en las versiones más recientes.

Para cada fase, se establecieron métricas específicas, incluyendo el tiempo de respuesta del servidor y la tasa de errores, para evaluar el éxito de la migración (Martínez, 2024).

2.4.3. Monitoreo constante.

Establecer un sistema de monitoreo continuo del rendimiento y disponibilidad del sistema durante y después de la migración. Esto permitirá detectar y abordar cualquier problema rápidamente para minimizar el impacto en el servicio.

2.5. Fase 5 Mantenimiento

2.5.1. Plan de retroceso (rollback).

Desarrollar un plan de reversión para restaurar el estado anterior del sistema en caso de problemas graves que no puedan resolverse de inmediato. Esto permitirá revertir la migración de forma segura y reducirá al mínimo el tiempo de inactividad.

2.5.2. Soporte técnico.

Establecer un equipo de soporte técnico especializado para responder de forma ágil a cualquier problema que surja tras la migración. Establecer una línea de atención directa y definir tiempos específicos de respuesta para la resolución de incidentes.

2.5.3. Evaluación posterior a la migración.

Llevar a cabo una evaluación exhaustiva post-migración para identificar lecciones aprendidas y áreas de mejora. Aprovechar estos hallazgos para optimizar futuras migraciones y procesos de actualización.

En definitiva, se diría que un plan de contingencia para Java EE es fundamental ya que garantizar la disponibilidad, confiabilidad y seguridad de una aplicación Java Enterprise Edition, proporcionando respuestas claras y rápidas ante posibles problemas o situaciones adversas.

CAPÍTULO III

3.1. Evaluación de Usabilidad del Sistema "SIGEPAA" Migrado a Jakarta EE según la Norma ISO/IEC 25010.

La evaluación de la usabilidad del sistema SIGEPAA, tras su actualización de Java EE a Jakarta EE, se realiza conforme a los criterios establecidos en la norma ISO/IEC 25010. Esta norma proporciona un marco de referencia para evaluar diversas características de calidad del software, incluida la usabilidad, que son cruciales para garantizar una experiencia de usuario efectiva y satisfactoria (© iso25000, 2022).

3.1.1. Escala de Liker

La escala de Likert es una herramienta que facilita la evaluación de actitudes y permite identificar el grado de acuerdo de los encuestados con diferentes afirmaciones. Este método se emplea en contextos donde es importante captar las opiniones de los usuarios.

Las opciones de respuesta en una escala de Likert reflejan la intensidad de los sentimientos del participante hacia cada afirmación, midiendo aspectos como el nivel de conformidad, la frecuencia de una acción, la relevancia de ciertos factores, la probabilidad de uso continuado y la percepción general de un producto, servicio o empresa.

Tabla 25. Puntuación de la escala de Likert

Respuesta	Valor
Totalmente en desacuerdo	1
En desacuerdo	2
Ni de acuerdo, ni en desacuerdo	3

De acuerdo	4
Totalmente de acuerdo	5

Nota. (Bautista-Ortega & Santillán Fernández, 2020)

3.1.2. Cuestionario SUS

SUS, que significa Scale Usability Systems, es el cuestionario más comúnmente empleado para evaluar la percepción de usabilidad de cualquier sistema o aplicación. Esta escala destaca por su simplicidad, y múltiples estudios han demostrado que sus resultados suelen ser muy confiables y precisos. Debido a esto, SUS se ha consolidado como uno de los métodos más populares para medir la usabilidad en la experiencia del usuario.

3.1.3. Análisis e interpretación de resultados

Para validar los resultados, se aplicó la norma ISO/IEC 25010, con especial énfasis en el aspecto de usabilidad. Se diseñó una encuesta fundamentada en la escala de usabilidad (SUS) para evaluar la experiencia del usuario, ya que esta herramienta es reconocida por su practicidad y confiabilidad en la medición de la percepción de facilidad de uso. La escala SUS incluye diez preguntas, divididas de manera equitativa entre afirmaciones positivas y negativas en su formato original.

El cuestionario está compuesto por las siguientes preguntas:

1. ¿Creo que me gustaría utilizar este sistema con frecuencia??
2. ¿Encontró el sistema más complejo de lo que debería ser?
3. ¿Pienso que el sistema es fácil de usar?
4. ¿Cree que necesitaría el apoyo de un técnico en software para poder utilizar este sistema??

5. ¿Encontró Ud. que las diversas funciones de este sistema están bien integradas?
6. ¿Pensé que había demasiada inconsistencia en este sistema?
7. ¿Me imagino que la mayoría de la gente aprendería a utilizar este sistema muy rápidamente?
8. ¿Encontró el sistema complicado de usar?
9. ¿Me sentí muy seguro usando el sistema?
10. ¿Necesitaba aprender muchas cosas antes de empezar con este sistema?

La tabla 26 presenta los resultados obtenidos de las encuestas realizadas, mostrando los resultados por cada pregunta de manera detallada.

Tabla 26. Resultados de la encuesta por pregunta

Respuestas	Preguntas									
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**
Totalmente en desacuerdo	0	13	0	9	0	12	0	12	0	12
En desacuerdo	0	5	0	6	0	4	0	8	0	4
Ni de acuerdo ni en desacuerdo	0	2	0	5	1	4	0	0	5	4
De acuerdo	8	0	8	0	7	0	7	0	6	0
Totalmente de acuerdo	12	0	12	0	12	0	13	0	9	0
Total	20	20	20	20	20	20	20	20	20	20

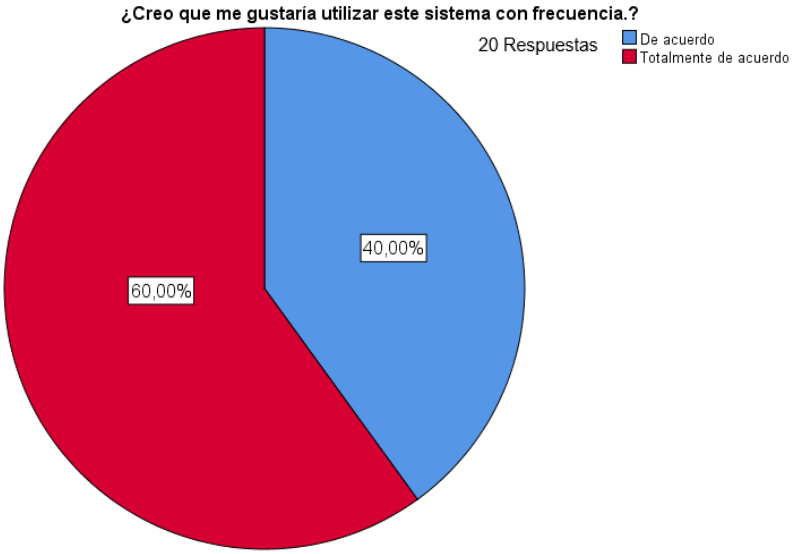
Nota: Elaborado por: Narváez Esteban

RESULTADOS DE LOS DATOS OPTENIDOS

Resultados de las encuestas por cada pregunta

A continuación, se muestran los resultados obtenidos de cada pregunta del cuestionario SUS aplicado a 20 de los 42 funcionarios de planta que conforman el personal de la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante EPAA-AA.

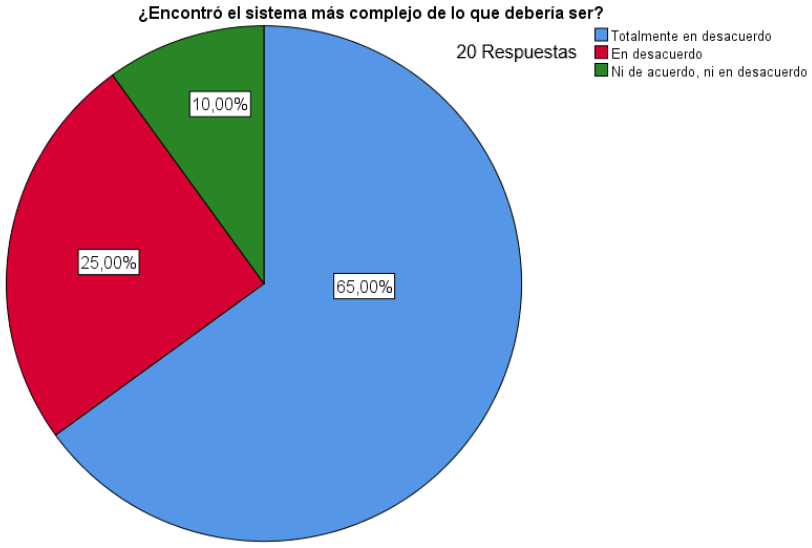
Figura 33 Resultados pregunta 1



Análisis

Según los resultados mostrados en la Figura 33, el 60% de los encuestados está "Totalmente de acuerdo" en utilizar con frecuencia el sistema SIGEPAA actualizado con Jakarta EE, mientras que el 40% restante está "De acuerdo". Esto sugiere que el sistema está plasmado a ser utilizado con frecuencia por los funcionarios de la EPAA-AA.

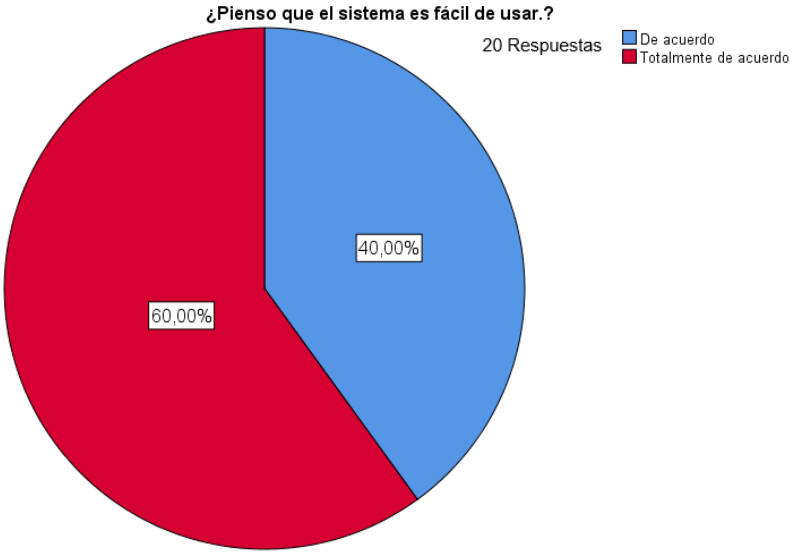
Figura 34 Resultados pregunta 2



Análisis

Según los resultados mostrados en la Figura 34, se puede observar que el 65% de los encuestados dice estar “Totalmente en desacuerdo” en que el sistema web es complejo, el 25% señala estar "En desacuerdo" y el 10% restante da a conocer que no está “Ni de acuerdo, ni en desacuerdo”. Esto sugiere que la aplicación web es muy fácil de comprender por parte de los funcionarios de la EPAA-AA.

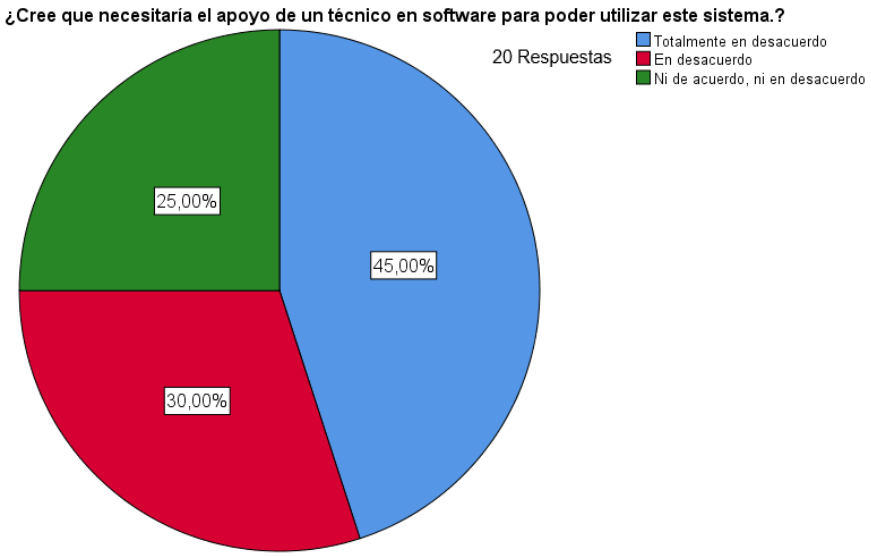
Figura 35 Resultados pregunta 3



Análisis

Según los resultados mostrados en la Figura 35, se puede observar que el 60% de encuestados dice estar “Totalmente de acuerdo” con la factibilidad de uso del sistema web, mientras que el 40% restante está "De acuerdo". Esto indica que la aplicación web es percibida como muy dinámica e intuitiva de manejar por los funcionarios de la empresa EPAA-AA.

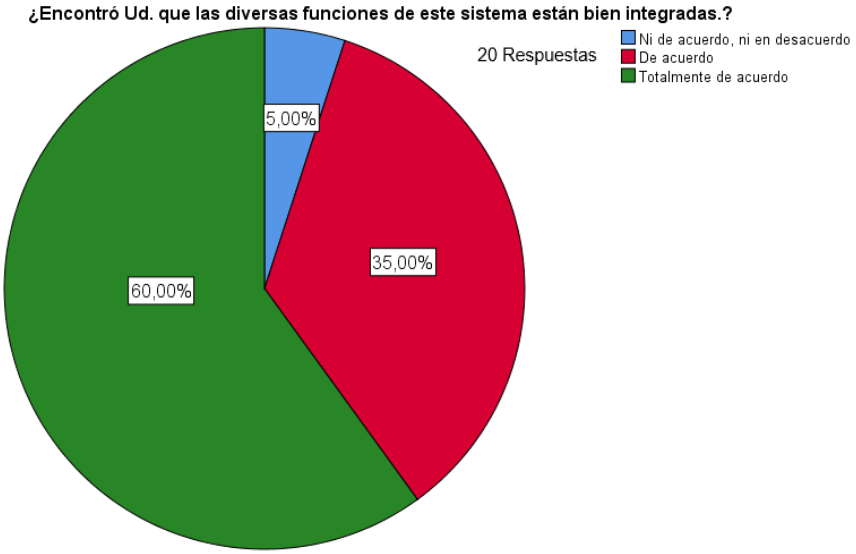
Figura 36 Resultados pregunta 4



Análisis

Según los resultados mostrados en la Figura 36, se puede observar que el 45% de encuestados está “Totalmente en desacuerdo” en el apoyo de personal técnico en software, para utilizar la aplicación, mientras que el 30% restante da a conocer que esta “En desacuerdo” y el 25% restante menciona que está “Ni de acuerdo, ni en desacuerdo”. De este modo, con los resultados obtenidos se puede concluir que los funcionarios de la EPAA-AA requieren poca o ninguna asistencia para interactuar con la aplicación SIGEPAA actualizada. Esto incluye a aquellos empleados que no están familiarizados con el uso de tecnologías o aplicaciones web.

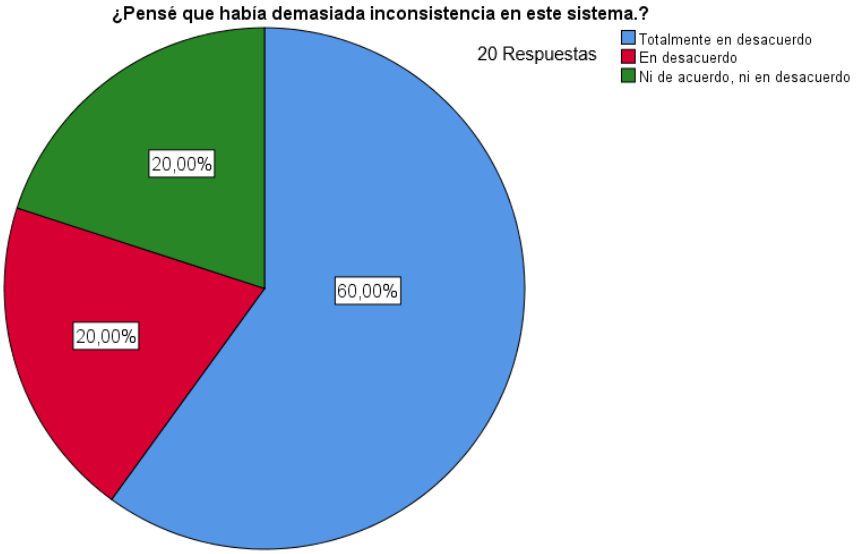
Figura 37 Resultados pregunta 5



Análisis

Según los resultados mostrados en la Figura 37, se puede observar que el 60% de encuestados está “Totalmente de acuerdo” en el proceso que con lleva la aplicación web, el 35% restante está “De acuerdo” mientras que el 5% de encuestados restantes plantea estar “Ni de acuerdo, ni en desacuerdo”. Así, con los resultados obtenidos, se puede establecer que el proceso llevado a cabo por la empresa EPAA-AA está perfectamente integrado con la aplicación web, cumpliendo eficientemente su función en cada uno de los procesos incorporados.

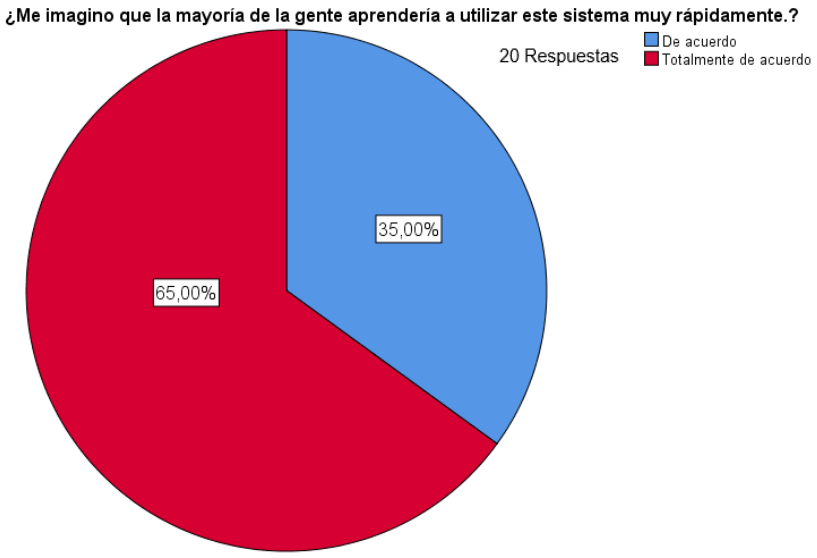
Figura 38 Resultados pregunta 6



AnálisiS

Según los resultados mostrados en la Figura 38, se puede observar que el 60% de encuestados está “Totalmente en desacuerdo” en que se presentan inconsistencias dentro del aplicativo web SIGEPAA, el 20% restante está “En desacuerdo” mientras que el otro 20% de encuestados plantea estar “Ni de acuerdo, ni en desacuerdo”. Por lo tanto, se puede manifestar que el sistema es completamente funcional y cumple con las funcionalidades requeridas para los funcionarios de la EPAA-AA.

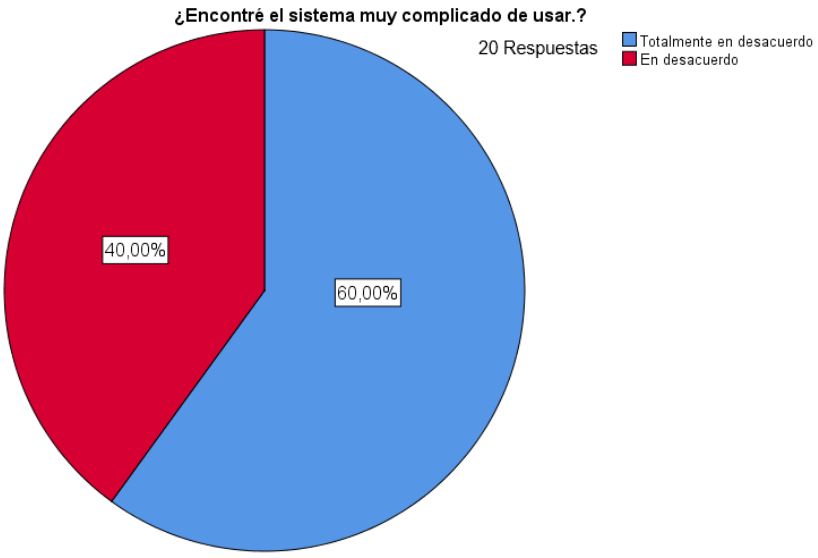
Figura 39 Resultados pregunta 7



Análisis

Según los resultados mostrados en la Figura 39, se puede observar que el 65% de encuestados está “Totalmente de acuerdo” dando a entender que el sistema SIGEPAA es fácil de aprender y el 35% restante de los encuestados establece estar “De acuerdo”. Por lo tanto, se pude concluir que el sistema es intuitivo y fácil de aprender, incluso para aquellos que no tienen experiencia previa con tecnología.

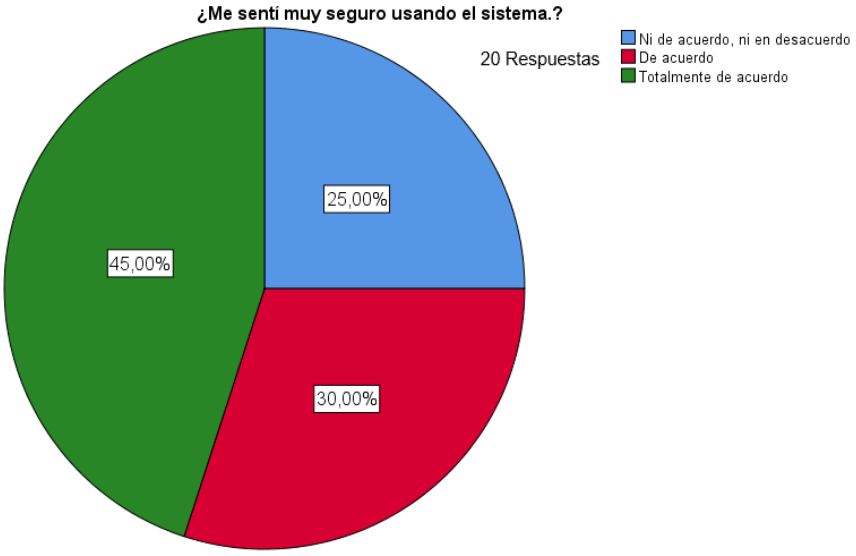
Figura 40 Resultados pregunta 8



Análisis

Según los resultados mostrados en la Figura 40, se puede observar que el 60% de encuestados dice estar “Totalmente en desacuerdo” en que el sistema web es difícil de usar, el 40% establece estar “En desacuerdo”. Por lo tanto, se pude entender que la aplicación web actualizada incorpora adecuadamente las funcionalidades solicitadas por el usuario, permitiendo el uso dinámico del sistema.

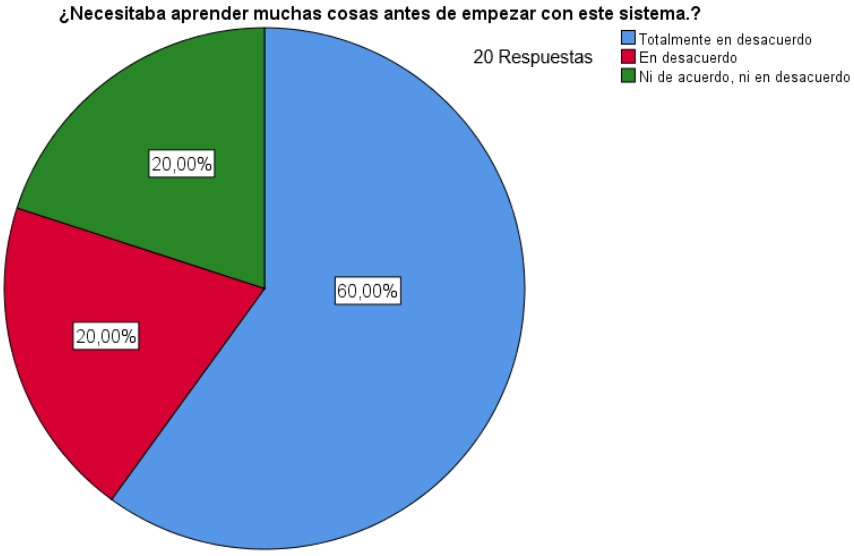
Figura 41 Resultados pregunta 9



Análisis

Según los resultados mostrados en la Figura 41, se puede observar que el 45% de encuestados dice estar “Totalmente de acuerdo” al sentirse seguros al usar el sistema Web, el 30% restante está “De acuerdo” mientras que el otro 20% de encuestados plantea estar “Ni de acuerdo, ni en desacuerdo”. Por lo tanto, se pude entender que la aplicación web es confiable y segura ante los funcionarios de la EPAA-AA.

Figura 42 Resultados pregunta 10



Análisis

Según los resultados mostrados en la Figura 42, se puede observar que el 60% de encuestados está “Totalmente en desacuerdo” con que es necesario aprender alguna tecnología de software para entender la capacidad y usabilidad del sistema web, el 20% está “En desacuerdo” mientras que el otro 20% de encuestados plantea estar “Ni de acuerdo, ni en desacuerdo”. Por lo tanto, estos resultados proporcionan que el manejo de del sistema web no requiere aprender varias cosas de tecnología previa para su usabilidad, operabilidad y compresión del personal.

Figura 43 Resultados pregunta adicional al cuestionario SUS

Análisis

Según los resultados mostrados en la Figura 43, el 100% de encuestados indica estar "Totalmente de acuerdo" en que el sistema SIGEPAA de la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante EPAA-AA debe estar actualizado con Jakarta EE para estar a la vanguardia de las innovaciones tecnológicas. Este resultado proporciona cual importante es para los funcionarios de la empresa que el Sistema Web SIGEPAA este actualizado con Jakarta EE para que EPAA-AA este a la vanguardia tecnológica.

Análisis de los resultados obtenidos.

Para determinar el puntaje SUS de la aplicación web, se recopilaron individualmente cada una de las encuestas y se asignó a cada respuesta su valor numérico correspondiente. Estos valores se utilizaron para realizar una evaluación, que se presenta de manera detallada en la Tabla 27.

Tabla 27. Puntación de los resultados por encuesta

Respuestas	Preguntas									
	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
1	5	1	5	1	5	3	5	1	5	1
2	4	2	4	2	4	3	4	2	4	2
3	4	2	4	3	4	2	5	2	3	3
4	5	2	5	3	5	1	5	1	5	1
5	5	1	4	1	5	1	5	2	5	1
6	4	2	4	2	4	2	4	2	4	2
7	5	1	5	1	5	1	5	2	4	2
8	5	1	5	2	5	2	4	1	4	2
9	5	1	5	1	5	1	5	1	5	1
10	5	1	5	3	5	1	5	1	5	1
11	4	1	5	2	4	2	5	1	5	1
12	5	1	5	1	5	1	5	1	5	1
13	4	3	4	1	4	1	4	1	3	3
14	4	1	4	3	4	1	4	1	4	1
15	5	1	5	1	5	1	5	2	5	1
16	4	3	4	2	3	3	4	2	3	3
17	5	2	4	3	4	3	5	2	3	3
18	4	1	5	2	5	1	5	1	3	1
19	5	1	5	1	5	1	4	1	4	1
20	5	1	5	1	5	1	5	1	5	1

Nota: Elaborado por: Narváez Esteban

Al asignar los valores correspondientes a cada respuesta, se procede a calcular la suma de las preguntas impares por grupo, como se ilustra en la Tabla 28 y la Tabla 29. Para realizar esta suma, se utiliza la variable **"*X*"** para representar las preguntas impares, mientras que la variable **"*Y*"** se asigna a las preguntas pares. Posteriormente, se aplican las siguientes fórmulas: ***X* = P1 +**

P3 + P5 + P7 + P9 y ***Y* = P2 + P4 + P6 + P8 + P10**. Es importante no alterar el orden de las preguntas, conforme a lo indicado por la metodología SUS.

Preguntas Impares

La tabla 28. Indica la sumatoria de las preguntas impares de cada encuesta para poder obtener los respectivos valores de la variable *X*.

Tabla 28. Sumatoria de los resultados impares de cada encuesta.

Respuestas	Preguntas					x
	P1	P3	P5	P7	P9	
1	5	5	5	5	5	25
2	4	4	4	4	4	20
3	4	4	4	5	3	20
4	5	5	5	5	5	25
5	5	4	5	5	5	24
6	4	4	4	4	4	20
7	5	5	5	5	4	24
8	5	5	5	4	4	23
9	5	5	5	5	5	25
10	5	5	5	5	5	25
11	4	5	4	5	5	23
12	5	5	5	5	5	25
13	4	4	4	4	3	19
14	4	4	4	4	4	20
15	5	5	5	5	5	25
16	4	4	3	4	3	18
17	5	4	4	5	3	21
18	4	5	5	5	3	22

19	5	5	5	4	4	23
20	5	5	5	5	5	25

Nota: Elaborado por: Narváez Esteban

Preguntas Pares

La tabla 29. Indica la sumatoria de las preguntas pares de cada encuesta para poder obtener los respectivos valores de la variable Y.

Tabla 29. Sumatoria de los resultados pares de cada encuesta.

Respuestas	Preguntas					y
	P2	P4	P6	P8	P10	
1	1	1	3	1	1	7
2	2	2	3	2	2	11
3	2	3	2	2	3	12
4	2	3	1	1	1	8
5	1	1	1	2	1	6
6	2	2	2	2	2	10
7	1	1	1	2	2	7
8	1	2	2	1	2	8
9	1	1	1	1	1	5
10	1	3	1	1	1	7
11	1	2	2	1	1	7
12	1	1	1	1	1	5
13	3	1	1	1	3	9
14	1	3	1	1	1	7
15	1	1	1	2	1	6
16	3	2	3	2	3	13
17	2	3	3	2	3	13
18	1	2	1	1	1	6

19	1	1	1	1	1	5
20	1	1	1	1	1	5

Nota: Elaborado por: Narváez Esteban

Como se muestra en la Tabla 28 y la Tabla 29, tras sumar los resultados de las preguntas pares e impares, se aplicaron las siguientes fórmulas: $\boldsymbol{x_0 = X - 5}$ y $\boldsymbol{y_0 = 25 - Y}$, Estas fórmulas se utilizan para determinar el puntaje SUS mediante la fórmula $\boldsymbol{SUS = (x_0 + y_0) * 2.5}$. En la Tabla 30 se presenta el puntaje SUS obtenido en cada encuesta, así como el promedio general (SUS), que evalúa el nivel de usabilidad o la experiencia del usuario en la aplicación.

Tabla 30. Resultados de x_0, y_0 y promedio SUS

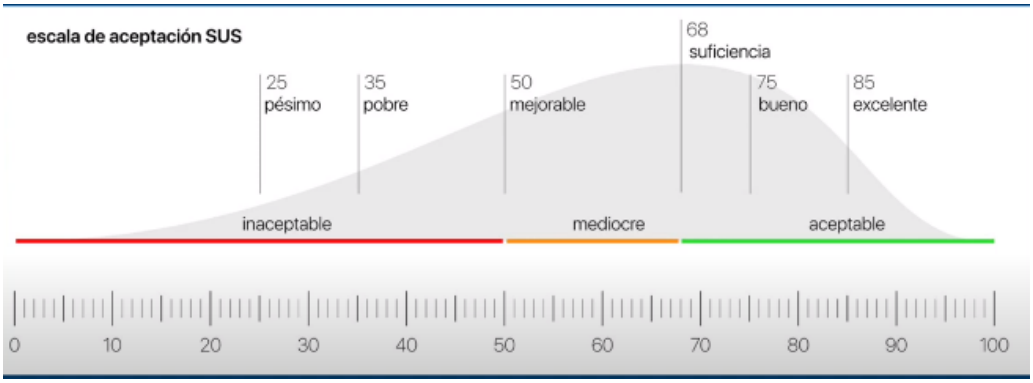
Encuesta	x	y	x_0	y_0	SUS
1	25	7	20	18	95
2	20	11	15	14	72,5
3	20	12	15	13	70
4	25	8	20	17	92,5
5	24	6	19	19	95
6	20	10	15	15	75
7	24	7	19	18	92,5
8	23	8	18	17	87,5
9	25	5	20	20	100
10	25	7	20	18	95
11	23	7	18	18	90
12	25	5	20	20	100
13	19	9	14	16	75
14	20	7	15	18	82,5
15	25	6	20	19	97,5
16	18	13	13	12	62,5

17	21	13	16	12	70
18	22	6	17	19	90
19	23	5	18	20	95
20	25	5	20	20	100
				Promedio	86,9

Nota: Elaborado por: Narváez Esteban

Al calcular el promedio SUS (System Usability Scale) a partir de las encuestas realizadas a los diferentes funcionarios de la empresa EPAA-AA, se obtuvo un puntaje promedio de 86.9. Según se indica en la Figura 44 la Escala de Usabilidad del Sistema, este resultado se clasifica en el Grado A, lo que indica una calificación excelente y un nivel de aceptación categorizado como "Aceptable".

Figura 44 Escala de Usabilidad del Sistema (SUS)



Nota. (Brooke, 2013)

La adopción de Jakarta EE ha permitido mejorar la experiencia de los usuarios, ya que el framework proporciona mayor estabilidad, escalabilidad y eficiencia en el desarrollo de los módulos del sistema. Esto ha contribuido a una mayor satisfacción, al optimizar los tiempos de respuesta y la interacción con el sistema.

CONCLUSIONES

- La revisión bibliográfica permitió evaluar detalladamente la situación actual del módulo de secretaria del sistema SIGEPAA, identificando puntos clave para su mejora arquitectónica. Esta revisión fue fundamental para proponer actualizaciones alineadas con el framework Jakarta EE, optimizando así su rendimiento y funcionalidad.
- Durante la implementación de la actualización del sistema SIGEPAA en la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante EPAA-AA, se confirmó que Jakarta EE es completamente compatible con los módulos previamente desarrollados en Java EE. Esta compatibilidad garantizó una transición fluida y sin interrupciones para los usuarios y operadores del sistema.
- Para la implementación de Jakarta EE fue crucial utilizar herramientas como Eclipse IDE 2024-06 y el servidor de aplicaciones WildFly 30.0.0, dado que estas versiones ofrecen una mayor adaptabilidad y compatibilidad con el framework. Esta actualización permitió optimizar la integración del sistema, asegurando un entorno de desarrollo más eficiente y preparado para aprovechar las capacidades avanzadas de Jakarta EE.
- La actualización del sistema SIGEPAA de Java EE a Jakarta EE trajo importantes mejoras en rendimiento, seguridad y escalabilidad, alineando los módulos con estándares actuales. El sistema ahora es más eficiente y fácil de mantener, con una mayor capacidad para integrarse con tecnologías emergentes. Además, esta actualización dejó al sistema preparado para futuras expansiones y actualizaciones, garantizando su sostenibilidad a largo plazo y reconociendo su crucial importancia para posicionar a la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante EPAA-AA a la vanguardia de la tecnología.

RECOMENDACIONES

- Se recomienda continuar explorando nuevas funcionalidades y mejoras que puedan incorporarse en futuras actualizaciones del sistema SIGEPAA de la Empresa Pública de Agua Potable y Alcantarillado.
- Planificar y ejecutar actualizaciones regulares del framework Jakarta EE y sus componentes. Esto garantizará que el sistema permanezca protegido frente a vulnerabilidades de seguridad y se mantenga optimizado para aprovechar las tecnologías más recientes.
- Adoptar un enfoque proactivo para seguir las innovaciones tecnológicas diarias, permitiendo incorporar beneficios estratégicos que impulsen la mejora continua de los sistemas. Esto no solo asegura mantenerse a la vanguardia, sino también un impacto positivo en los objetivos empresariales.
- Utilizar las versiones más recientes de herramientas compatibles con Jakarta EE, como Eclipse IDE, que ofrecen adaptaciones más eficientes al framework, mejorando la experiencia de desarrollo y optimizando los procesos del sistema.

Bibliografía

Abdellatif, M. S.-s. (s.f.). *sciencedirect*. Obtenido de sciencedirect: https://doi.org/10.1016/j.jss.2020.110868

Arrambide, J. (2010). *El futuro de Java EE, ahora es Jakarta EE.* Bogota: Jarrambide. Obtenido de https://jarrambide.com/java/porque-cambia-de-nombre-java-ee-a-jakarta-ee/

Baena Chavarriaga, M. (2023). *Migración de módulos en un ERP empresarial basado en Java a frameworks modernos.* Medellin: Sistema de Bibliotecas. Obtenido de https://bibliotecadigital.udea.edu.co/bitstream/10495/33808/5/BaenaMateo_2023_MigracionModulosEmpresarial.pdf

Bolaño, O. A. (2017). *Comparativo de Frameworks web MVC open source en java, para la migración del administrador vortal en el CIADTI.* Bogota: unipamplona. Obtenido de http://repositoriodspace.unipamplona.edu.com

Brocheto, W. (2017). *Cuáles son las mejores prácticas para diseñar y desarrollar servlets en Jakarta EE.* España: LinkedIn.

BRUSELAS, B. (22 de 09 de 2022). *ECLIPSE FOUNDATION*. Obtenido de ECLIPSE FOUNDATION: https://newsroom.eclipse.org/news/announcements/enterprise-java-and-jakarta-ee-continue-grow-results-2022-jakarta-ee-developer

Castillo, C. F. (2021). Desarrollo de aplicaciones web con Jakarta EE. En C. F. Castillo, *Desarrollo de aplicaciones web con Jakarta EE* (pág. 500). ALPHAEDITORIAL.

Castillo, C. F. (2021). *Desarrollo de aplicaciones web con Jakarta EE.* Londres: Marcombo. Obtenido de https://www.perlego.com/es/book/2423713/desarrollo-de-aplicaciones-web-con-jakarta-ee-pdf

Castillo, C. F. (2022). *Desarrollo de aplicaciones web con Jakarta EE* (Primera ed.). Bogota: Alpha Editorial S.A. Obtenido de https://books.google.es

Castillo, C. F. (2022). *Desarrollo de aplicaciones WEB con jakarta EE.* Bogota: Alpha Editorial S.A. Obtenido de https://books.google.es

COLIMBA HUERTAS, I. P. (21 de Julio de 2016). *Repositorio UTN*. Obtenido de Repositorio UTN: http://repositorio.utn.edu.ec/bitstream/123456789/5698/1/04%20ISC%20427%20TRABAJO%20DE%20GRADO.pdf

Eclipse. (s.f.). *Entorno de desarrollo integrado*. Obtenido de Entorno de desarrollo integrado: https://www.ecured.cu/Eclipse,_entorno_de_desarrollo_integrado

Elder, M. (2020). JAKARTA EE Cookbook. En M. Elder, *JAKARTA EE Cookbook* (pág. 752). Packt Publishing; 2nd edición (29 Mayo 2020).

Foundation, E. (s.f.). *JAKARTA EE SPECIFICATIONS*. Obtenido de JAKARTA EE SPECIFICATIONS: https://jakarta.ee/specifications/

García , C. (2022). *Software de gestión de contratación con arquitectura Java EE multicapa para empresas proveedoras de servicios* (Vol. primero). España: Euro-Inf. Obtenido de https://ruc.udc.es/

Gudiño Quinteros, A. R. (15 de Enero de 2019). *Repositorio UTN*. Obtenido de Repositorio UTN: http://repositorio.utn.edu.ec/handle/123456789/8806

IDEA, I. (19 de Diciembre de 2022). *Jakarta Persistence (JPA)*. Obtenido de Jakarta Persistence (JPA): https://www.jetbrains.com/help/idea/jakarta-persistence-jpa.html

iso25000.com. (2022). *ISO 25000 Calidad de Sofware y datos* . Obtenido de ISO 25000 Calidad de Sofware y datos : https://iso25000.com/index.php/normas-iso-25000/iso-25010

Jetbrains. (09 de diciembre de 2020). *Be Productive in Jakarta EE*. Obtenido de Be Productive in Jakarta EE: https://lp.jetbrains.com/intellij-idea-jakartaee/?source=google&medium=cpc&campaign=19555376879&term=java%20ee&content=644625780124&gad=1&gclid=Cj0KCQjw_O2lBhCFARIsAB0E8B9XUamFd_gGNXc1-m5_-Cp-QmrnPJRv4PPgsmZ0FJukR8gLsGOn8TgaAlgaEALw_wcB

Jetbrains. (21 de Abril de 2023). *IntelliJ IDEA*. Obtenido de IntelliJ IDEA: https://www.jetbrains.com/help/idea/creating-and-running-your-first-java-ee-application.html

Jetbrains. (21 de Abril de 2023). *IntelliJ IDEA*. Obtenido de Enterprise application support: https://www.jetbrains.com/help/idea/enabling-java-ee-application-support.html

Juneau, J., & Telang, T. (2022). Java EE to Jakarta EE 10 Recipes: A Problem-Solution Approach for Enterprise Java. En J. J. Telang, *Java EE to Jakarta EE 10 Recipes: A Problem-Solution Approach for Enterprise Java* (pág. 1299). Apress; N.º 3 edición (18 julio 2022).

López, M., & Suco, C. (Marzo de 2021). *Universidad Politécnica Salesiana(UPS)*. Obtenido de Universidad Politécnica Salesiana(UPS): https://dspace.ups.edu.ec/bitstream/123456789/20276/5/UPS-CT009135.pdf

Lozano Peralta, C. (2020). *ARQUITECTURA DE SOFTWARE BASADA EN MICROSERVICIOS PARA EL USO EN DISPOSITIVOS DE INTERNET DE LAS COSAS*. La Salle . Lima: ulasalle. Obtenido de https://repositorio.ulasalle.edu.pe

Manelli, L., & Zambon, G. (2020). Beginning Jakarta EE Web Development. En L. Manelli, & G. Zambon, *Beginning Jakarta EE Web Development* (pág. 426). Apress; 3er edición.

Martins, J. (2022). *Qué es un plan de contingencia y cómo crear uno en 8 pasos para evitar riesgos.* San Francisco: asana. Obtenido de https://asana.com/es/resources/contingency-plan

Milinkovich, M. (15 de 03 de 2023). *ECLIPSE FOUNDATION*. Obtenido de ECLIPSE FOUNDATION: https://blogs.eclipse.org/post/mike-milinkovich/take-2023-jakarta-ee-developer-survey

Obradovic, T., & Grimstad, I. (22 de 09 de 2022). *JAKARTA EE*. Obtenido de JAKARTA EE: https://jakarta.ee/news/jakarta-ee-10-released/

Ordax Cassá, J., & Ocaña Díaz, P. (2009). *Programación Web en JAVA.* España: Aula Mentor. Obtenido de https://sede.educacion.gob.es/publiventa/PdfServlet?pdf=VP15838.pdf&area=E

Parra, A. (2021). *Fundamentos de Jakarta EE (Java EE).* Chile : Makigas S.A.

Peñaranda Huerta, J. E. (2018). *Cientifica*. Obtenido de cientifica: https://repositorio.cientifica.edu.pe/bitstream/handle/20.500.12805/635/TB-Pe%C3%B1aranda_Huerta.pdf?sequence=1

PNUD. (19 de 06 de 2017). *Programa de las Naciones Unidas para el Desarrollo*. Obtenido de Programa de las Naciones Unidas para el Desarrollo: https://www.undp.org/es/sustainable-development-goals/industria-innovacion-infraestructura

Rubensa. (15 de Enero de 2019). *El Blog de rubensa*. Obtenido de De Java EE a Jakarta EE: https://rubensa.wordpress.com/2019/01/15/de-java-ee-a-jakarta-ee/

SALINAS FLORES, J. E. (s.f.). *ESPE*. Obtenido de ESPE: https://repositorio.espe.edu.ec/bitstream/21000/5859/1/T-ESPE-034357.pdf

Scholtz, B., & Tijms, A. (2022). The Definitive Guide to Jakarta Faces in Jakarta EE 10: Building Java-Based Enterprise. En B. Scholtz, & A. Tijms, *The Definitive Guide to Jakarta Faces in Jakarta EE 10: Building Java-Based Enterprise* (pág. 713). Apress; 2nd edición.

Shannon, B. (2020). *Jakarta EE Platform.* New York: Eclipse Foundation. Obtenido de https://jakarta.ee/specifications/platform/9/jakarta-platform-spec-9.pdf

Späth, P. (2019). Beginning Jakarta EE. En P. Späth, *Beginning Jakarta EE* (pág. 539). Apress; 1st ed. edición (31 Agosto 2019).

Thierry , R. (2020). *Desarrolle aplicaciones web en Java.* España: Ediciones ENI. Obtenido de https://www.ediciones-eni.com

Velez, M. (2022). *Software de gestión de contratación con arquitectura Java EE multicapa para empresas proveedoras de servicios.* España: Edisiones UDS. Obtenido de https://ruc.udc.es

Veracruz, J. (2022). *Qué es Jakarta EE.* Mexico: Arquitectura Java. Obtenido de https://es.scribd.com/document/572188185/Que-es-Jakarta-EE

Villarreal Bravo, A., Mendoza González, G., & Gonçalves de Santana, O. (2020). *Building Modern Web Applications With Jakarta EE, NoSQL Databases and Microservices.* BPB Publications.

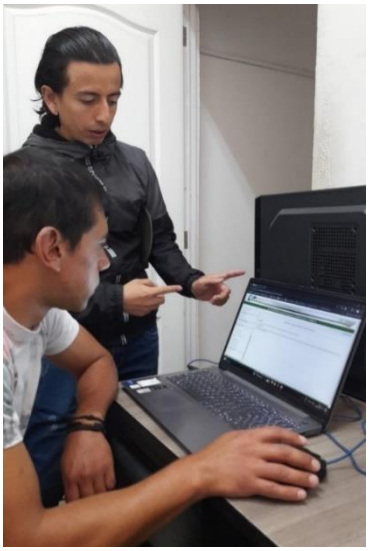
ANEXOS

Anexo A:

Figura 45 Interfaz inicial de Ejecución del Sistema SIGEPAA con Jakarta EE

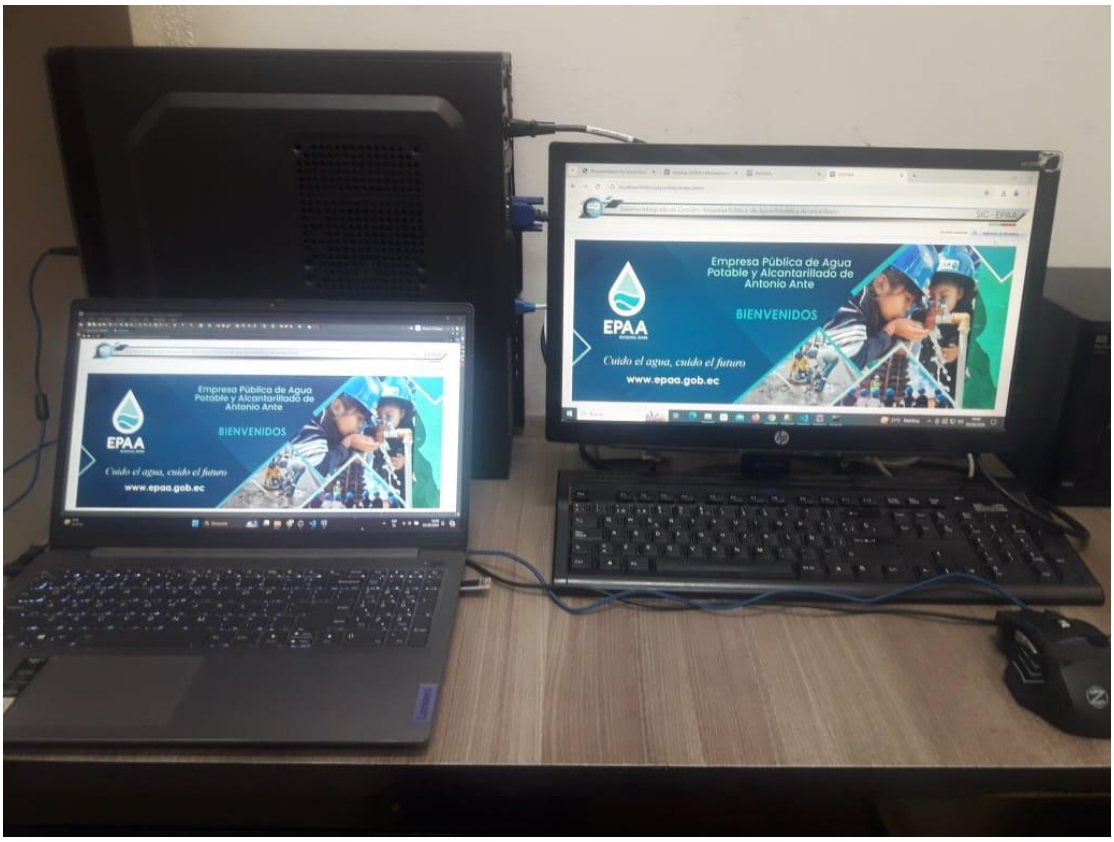
Figura 46 Revisión y explicación del módulo de Archivo del Sistema SIGEPAA actualizado de Java EE a Jakarta EE.

Anexo B:



Anexo C:

Figura 47 Ejecución del ambiente de desarrollo en equipo de cómputo de la EPAA-AA



Anexo D:

Figura 48 Manual de Instalación y configuración del ambiente de trabajo del Sistema SIGEPAA a Jakarta EE en computador EPAA-AA-F de la empresa pública de agua potable y alcantarillado de Antonio ante EPAA-AA área de TIC`s

Anexo E:

Figura 49 Oficio de Entrega del Sistema SIGEPAA actualizado a Jakarta EE en EPAA-AA.

OFICIO

Atuntaqui, 14 de agosto del 2024

Asunto: Entrega de documentación e instalación y configuración del ambiente de trabajo del Sistema SIGEPAA a Jakarta EE en computador EPAA-AA-F de la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante EPAA-AA área de TICs.

Señor Ing. Milton Franco.

Gerente General de la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante EPAA-AA.

De mi consideración:

Me dirijo a usted para expresar mi más sincero agradecimiento por el apoyo que me ha brindado para realizar el desarrollo de mi tema de tesis con **Oficio Nro. EPAA-AA-GG-2023-0364-OF** titulado "ESTUDIO DEL FRAMEWORK JAKARTA EE PARA LA ACTUALIZACIÓN DE MÓDULOS JAVA EE A JAKARTA EE EN EL SISTEMA "SIGEPAA" DE LA EMPRESA PÚBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE EPAA-AA".

Con fecha 12 de agosto del 2024, se ha completado exitosamente la instalación del ambiente de desarrollo a Jakarta EE del sistema SIGEPAA en el computador EPAA-AA-F designado por el Ing. Marco Sarmiento de las siguientes características:

Características Equipo EPAA-AA-F	
Nombre del Equipo:	DESKTOP-2SF5HFT
Edición de Windows:	Windows 10 Pro
Procesador:	AMD Ryzen 3 3200G with Radeon Vega Graphics 3.60GHz
Memoria Instalada (RAM):	8,00 GB (5.93 GB Utilizable)
Tipo de Sistema:	Sistema Operativo de 64 bits, procesador x64

Por medio del presente me permito entregar el documento de instalación del ambiente de Jakarta EE con el módulo de archivo funcionando al 100% incluyendo una copia en disco extraíble de TICs, PDF del manual de instalación y configuración en digital en el escritorio EPAA016 de la empresa EPAA-AA.

Cumpliendo con todos los requisitos y especificaciones previamente acordados en memorandos y acta de acuerdo firmada con el área de TICS de la EPAA-AA.

Adjunto al presente oficio, se incluye el documento correspondiente que detalla la instalación y configuración realizada en el computador especificado.

EPAA RECIBIDO
EMPRESA PUBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE
14 AGO 2024
HORA: 2:20
FIRMA:

Anexo F:

Figura 50 Acta de Recepción del Sistema SIGEPAA Actualizado a Jakarta EE

EPAA Antonio Ante — Empresa Pública de Agua Potable y Alcantarillado

INFORME DE RECEPCIÓN A SATISFACCIÓN DEL PRODUCTO DEL TEMA DE TESIS "ESTUDIO DE FRAMEWORK JAKARTA EE PARA LA ACTUALIZACION DE MODULOS JAVA EE A JAKARTA EE EN EL SISTEMA SIGEPA DE LA EMPRESA PUBLICA DE AGUA POTABLE Y ALCANTARILADO DE ANTONIO ANTE DE ATUNTAQUI EPAA-AA.

PRIMERO. - OBJETIVO. –

En la ciudad de Atuntaqui, a los 01 días del mes de Agosto de 2024, en representación de la Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante, el Ing. Marco Sarmiento Rosero en calidad de Especialista en Tecnologías y el Sr Esteban Narváez Estudiante UTN desarrollador del producto del Tema de Tesis **"ESTUDIO DE FRAMEWORK JAKARTA EE PARA LA ACTUALIZACION DE MODULOS JAVA EE A JAKARTA EE EN EL SISTEMA SIGEPA DE LA EMPRESA PUBLICA DE AGUA POTABLE Y ALCANTARILADO DE ANTONIO ANTE DE ATUNTAQUI EPAA-AA.SEGUNDO. –**

SEGUNDO. –ANTECEDENTES. –
Con fecha 19 de diciembre del 2023 mediante oficio Oficio Nro. EPAA-AA-GG-2023-0364-OF, La Empresa Pública de Agua Potable y Alcantarillado de Antonio Ante, Autorización para el desarrollo de la tesis del Sr. Narváez Arcos Esteban Alexis de la Universidad Técnica del Norte UTN en la EPAA-AA.

TERCERA. – ENTREGA DEL PRODUCTO. –

El Sr. Sr. Narváez Arcos Esteban Alexis, con fecha 30 de julio del 2024, comunica que los productos objeto del tema de Tesis son entregados y recibidos por la EPAA-AA, por lo cual solicita la Certificación de entrega del sistema SIGEPAA actualizado de Java EE a Jakarta EE.

CUARTA. – RECEPCIÓN DEL PRODUCTO. –

De conformidad a las especificaciones técnicas estipuladas en el Tema de Tesis **"ESTUDIO DE FRAMEWORK JAKARTA EE PARA LA ACTUALIZACION DE MODULOS JAVA EE A JAKARTA EE EN EL SISTEMA SIGEPA DE LA EMPRESA PUBLICA DE AGUA POTABLE Y ALCANTARILADO DE ANTONIO ANTE DE ATUNTAQUI EPAA-AA.**

QUINTA. – INFORME DE RECEPCIÓN A SATISFACCIÓN. –

Como Especialista en Tecnologías de la empresa Pública de Agua Potable y Alcantarillado de Antonio Ante y una vez que se ha cumplido y entregado todos los productos del tema de Tesis, SE DA POR RECIBIDO A SATISFACCIÓN los productos objeto de la presente TESIS **"ESTUDIO DE FRAMEWORK JAKARTA EE PARA LA ACTUALIZACION DE MODULOS JAVA EE A JAKARTA EE EN EL SISTEMA SIGEPA DE LA EMPRESA PUBLICA DE AGUA POTABLE Y ALCANTARILADO DE ANTONIO ANTE DE ATUNTAQUI EPAA-AA.** Se recibe a satisfacción la actualización del sistema SIGEPAA de java EE a Jakarta EE se adjunta la documentación de respaldo (Manual de Instalación, Manual de Usuario).

MARCO ANTONIO SARMIENTO ROSERO

Ing. Marco Sarmiento

Especialista en Tecnologías.

Anexo G:

Figura 51 Acuerdo de Confidencialidad

Empresa Pública de Agua Potable y Alcantarillado Antonio Ante

EMPRESA PÚBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE EPAA-AA

ACUERDO DE CONFIDENCIALIDAD

Este **Acuerdo de Confidencialidad de la Información** se celebra y se establece en la ciudad de Atuntaqui, Ecuador, el día 20 de diciembre **de 2023** por y entre **Ing. Fernanda Aguirre – Técnico de Sistemas y Tecnología** y **Sr. Narváez Arcos Esteban Alexis – ESTUDIANTE CSOFT UTN** con Cédula de Identidad **040180795-3**, en relación con la divulgación de **Información Confidencial** de la **Empresa Pública de Agua Potable y Alcantarillado EPAA-AA** hacia el **Sr. Narváez Arcos Esteban Alexis – ESTUDIANTE CSOFT UTN.**

1. OBJETO DEL ACUERDO

La **Ing. Fernanda Aguirre – Técnico de Sistemas y Tecnología** tienen la intención de revelar y entregar **Información Confidencial**, con la previa autorización de la **Máxima Autoridad o Jefes Inmediatos**, al **Sr. Narváez Arcos Esteban Alexis – ESTUDIANTE CSOFT UTN** con el propósito: **en referencia al Oficio Nro.EPAA-AA-GG-2023-0364-OF, de fecha 19 de Diciembre de 2023, en el que se procede AUTORIZAR "...REALICE EL DESARROLLO DE SU PROYECTO DE TITULACIÓN CUYO TEMA ES: ESTUDIO DEL FRAMEWORK JAKARTA EE PARA LA ACTUALIZACIÓN DE MODULOS JAVA EE A JAKARTA EE EN EL SISTEMA "SIGEPAA" DE LA EMPRESA PUBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE EPAA-AA"**

El objeto del acuerdo de confidencialidad es fijar las condiciones bajo las cuales las partes comunicarán y mantendrán la confidencialidad de la Información compartida de forma escrita y digital.

2. INFORMACIÓN CONFIDENCIAL

La **Información Confidencial** a ser que incluye el acuerdo es la siguiente:

En base al PROYECTO DE TITULACIÓN CUYO TEMA ES: ESTUDIO DEL FRAMEWORK JAKARTA EE PARA LA ACTUALIZACIÓN DE MODULOS JAVA EE A JAKARTA EE EN EL SISTEMA "SIGEPAA" DE LA EMPRESA PUBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIA ANTE EPAA-AA, TICS con respecto a la base de datos, se le va a brindar

9. MODIFICACIONES DEL ACUERDO

a) El presente **Acuerdo de Confidencialidad de la Información** establece el acuerdo completo entre las partes en lo que respecta a la divulgación de información confidencial. Cualquier adición o modificación a este Acuerdo debe hacerse por escrito y ser firmada por ambas partes.

10. RESPONSABLE DE ENTREGA Y RECEPCIÓN DE LA INFORMACIÓN

La **Ing. Fernanda Aguirre – Técnico de Sistemas y Tecnología,** dará seguimiento a la información confidencial brindada al **Sr. Narváez Arcos Esteban Alexis – ESTUDIANTE CSOFT UTN** para uso exclusivamente de su **PROYECTO DE TITULACIÓN CUYO TEMA ES: ESTUDIO DEL FRAMEWORK JAKARTA EE PARA LA ACTUALIZACIÓN DE MODULOS JAVA EE A JAKARTA EE EN EL SISTEMA "SIGEPAA" DE LA EMPRESA PUBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE EPAA-AA**. Al concluir el proyecto, la encargada se asegurará que el producto final de la tesis este en estado de producción y posterior se realizará la devolución y la eliminación de la información del computador personal o cualquier dispositivo así evitando se retenga cualquier copia de manera permanente, el ente intelectual de esta información será **LA EMPRESA PÚBLICA DE AGUA POTABLE Y ALCANTARILLADO DE ANTONIO ANTE EPAA-AA**. Este seguimiento tiene como objetivo principal asegurar la protección efectiva de la información confidencial compartida.

11. CONSIDERACIONES FINALES

Si alguna de las disposiciones del presente **Acuerdo de Confidencialidad de la Información** se considera inaplicable, el resto se aplicará lo más plenamente posible y la(s) disposición(es) inaplicable(s) se considerará(n) modificada(s) en la medida necesaria para permitir la aplicación del Acuerdo en su conjunto. Y en prueba de conformidad y aceptación de lo establecido, ambas partes firman este acuerdo y aceptan voluntariamente los derechos y obligaciones que en él se establecen, en el lugar y fecha previamente indicados.

.....	.....
Ing. Miltón Franco	Sr. Esteban Alexis Narváez Arcos
Gerente General EPAA-AA	**ESTUDIANTE CSOFT UTN**