FACULTAD 2

Sistema Informático para la Gestión de Políticas de Tecnologías de la Información en las organizaciones cubanas

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

Autores:

Nayeli López Castro Morejón

Osmani Suárez Fernández

Tutor(es):

Dra. C. Mónica Peña Casanova

Ing. Clayret Echenique Quintana

Ing. Manuel Fornés Martínez

La Habana, diciembre de 2023

Año 65 de la Revolución

DECLARACIÓN DE AUTORÍA

Los autores del trabajo de diploma con título ***"Sistema Informático para la Gestión de Políticas de Tecnologías de la Información en las organizaciones cubanas"*** conceden a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la investigación, con carácter exclusivo. De forma similar se declaran como únicos autores de su contenido. Para que así conste firman la presente a los ____ días del mes de ____ del año _____.

Nayeli López Castro Morejón

_____

Firma del Autor

Osmani Suárez Fernández

_____

Firma del Autor

Dra. C. Mónica Peña Casanova

_____

Firma del Tutor

Ing. Clayret Echenique Quintana

_____

Firma del Tutor

Ing. Manuel Fornés Martínez

_____

Firma del Tutor

AGRADECIMIENTOS

De Nayeli:

A mis padres y mi abuela, por todo el amor, por confiar en mí, por preocuparse por mí y por todo el apoyo que me han dado durante todos estos años. A mi hermano por su amor, cariño y por siempre querer ayudarme. Y a toda mi familia por siempre estar ahí. A mi novio Manuel por ayudarme y aconsejarme, por haber estado siempre a mi lado durante estos 5 años y por confiar en mí en todo momento. A mis amistades y compañeros especialmente a Claudia que ha estado ahí desde que inicié en la Universidad apoyándome y ayudándome siempre. A mi amigo y compañero de Tesis Osmani porque sin él no se habría logrado este trabajo. A mis tutores por guiarme a lo largo del desarrollo de este trabajo.

De Osmani:

A mis papás por todo el apoyo que me han dado durante todos estos años de carrera y por siempre creer en mí. A mi hermano que, aunque siempre peleamos, sé que siempre puedo contar con él. A toda mi familia que siempre está presente en todo momento, en especial a Lili, la prima que más me quiere. A mi amiga Liannis de toda la vida, que ha sido y siempre será mi hermana de otra madre. A mis amigas Claudia y Nayeli que siempre han estado a mi lado durante estos 5 años. Y a todos mis profesores y un agradecimiento especial a Clayret, por ser la mejor tutora.

DEDICATORIA

De Nayeli:

Esta tesis está dedicada a:

Mis padres, que han sido la voluntad, la perseverancia y el amor durante toda mi vida. Gracias por su apoyo y confianza.

Mi hermano Abdiel por ser el mejor hermano del mundo.

Mi abuela Ana por siempre cuidarme y apoyarme.

Mi novio por darme su apoyo y amor incondicional.

Esta tesis va dedicada a ustedes por ser las personas más importantes de mi vida.

De Osmani:

Esta tesis está dedicada a mis padres, que han sido mi soporte en todo momento, siempre dándome ánimos para continuar y nunca renunciar, espero que se sientan orgullosos de lo que he podido lograr.

RESUMEN

En el presente documento se aborda el desarrollo de un Sistema Informático para la gestión de las políticas de Tecnologías de la Información (TI) en las organizaciones cubanas. La idea surgió de la necesidad que tienen las organizaciones del país de tener acceso a un sistema que permita la gestión centralizada de las políticas de TI, ya que actualmente no se cuenta con una herramienta informática que lo permita.

El sistema permite crear, modificar, visualizar y eliminar dichas políticas. Para eso se cuenta con funcionalidades para definir los eventos que puedan incidir sobre la red de la organización, así como los archivos (script) donde se almacena la configuración que va a dar respuesta a la aparición de alguno de estos eventos. Una vez que se cuenta con estos dos factores se podrán concatenar formando la política de seguridad que se almacenará. Se empleó Extreme Programming (XP) como metodología de desarrollo de software, como marco de trabajo Django, así como las herramientas PostgreSQL como gestor de base de datos y Git como sistema de control de versiones. Para la validación de esta propuesta de solución se aplicaron pruebas unitarias y funcionales.

Palabras clave: administrador de red, gestión centralizada, organizaciones cubanas, políticas de TI, repositorio, sistema informático, tecnologías de la información.

ABSTRACT

This document addresses the development of a Computer System for the management of Information Technology (IT) policies in Cuban organizations. The idea arose from the need for organizations in the country to have access to a system that allows centralized management of IT policies, since there is currently no computer tool that allows it.

The system allows you to create, modify, view and delete these policies. For this, there are functionalities to define the events that may affect the organization's network, as well as the files (script) where the configuration that will respond to the appearance of any of these events is stored. Once these two factors are available, they can be concatenated to form the security policy that will be stored. Extreme Programming (XP) was used as a software development methodology, framework Django, as well as the PostgreSQL tools as a database manager and Git as a version control system. To validate this proposed solution, Unit tests and functional tests were applied.

Keywords: network administrator, centralized management, Cuban organizations, IT policies, repository, computer system, information technologies.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La tecnología de la información (TI) constituyen el proceso de creación, almacenamiento, transmisión y percepción de la información y por ello se ha convertido en un activo estratégico para las organizaciones. Estas les permiten mejorar procesos, reducir costos y crear productos y servicios innovadores que aumentan la agilidad estratégica y la competitividad (Casanova, 2019).

Sin embargo, el uso de las TI también genera desafíos y riesgos, como la seguridad de la información, la privacidad de los datos y la gestión de la información. Para abordar estos desafíos y riesgos en las organizaciones es esencial establecer políticas TI, que no son más que reglas o principios que guían el acceso y uso de recursos relacionados con las tecnologías de la información. Dichas políticas definen como se deben utilizar los sistemas informáticos, redes, datos y otros activos tecnológicos, y son fundamentales para definir y priorizar objetivos y oportunidades (Casanova, 2019).

Una alternativa para la informatización de las políticas TI es mediante la gestión de red basada en políticas (PBNM por las siglas en inglés de Policy Based Network Management). Mediante el uso de reglas PBMN se permite coordinar y controlar los elementos de la red (Casanova, 2019).

PBNM es un módulo que controla y restringe de forma independiente el comportamiento del sistema de acuerdo con reglas de alto nivel descritas por el administrador. Esto se logra a través de un punto de decisión de políticas que es responsable de ejecutar el proceso de toma de decisiones y múltiples puntos de ejecución de políticas que convierten las operaciones en comandos que pueden ser entendidos por varios dispositivos en la red. Todo ello proporciona un control dinámico y coordinado sobre los elementos de la red gracias a que las decisiones se toman de forma automática. Esta gestión dinámica representa un cambio cualitativo desde una perspectiva empresarial, ya que permite una gestión más eficaz de los recursos y servicios de la red (Chávez & Luis, 2019).

Basado en lo expresado por la Dra. C. Mónica Peña Casanova durante su investigación de doctorado, en el contexto actual de las organizaciones cubanas, se observa una carencia de sistemas informáticos que sean capaces de gestionar y aplicar eficazmente las políticas de Tecnologías de la Información. Una de las principales limitaciones radica en la falta de herramientas informáticas que permitan la informatización y gestión centralizada de estas políticas. Esta carencia representa un desafío significativo, ya que dificulta la implementación efectiva de políticas TI, que son esenciales para el adecuado funcionamiento de las infraestructuras tecnológicas.

Cuando se logran informatizar las políticas TI, se evidencia una problemática adicional. Estas políticas tienden a quedar bajo el dominio exclusivo de los administradores de TI, lo que limita el acceso y comprensión de las mismas por parte de quienes toman decisiones a nivel estratégico en la organización. Este hecho impide que los responsables de la toma de decisiones tengan una visión integral de cómo las políticas TI pueden influir en los objetivos y metas de la organización, lo que resulta en una brecha entre los aspectos técnicos y la estrategia empresarial.

La ejecución de estas políticas, a menudo se lleva a cabo de forma dispersa en múltiples infraestructuras tecnológicas dentro de la organización. Esta dispersión de la ejecución puede generar complejidades adicionales, especialmente en lo que respecta a la detección y corrección de conflictos que puedan surgir entre las diferentes implementaciones de políticas. La falta de una gestión unificada dificulta la coordinación y la resolución de problemas, lo que, en última instancia, puede impactar en la estabilidad y seguridad de la infraestructura de TI.

Un tercer desafío es la ausencia de un mecanismo efectivo y seguro para el intercambio de buenas prácticas, en forma de políticas TI, entre diversas organizaciones. La falta de una plataforma o proceso estandarizado que permita compartir y adoptar políticas exitosas impide que las organizaciones aprovechen lecciones aprendidas y enfoques exitosos de otras entidades. Esto, a su vez, obstaculiza el progreso colectivo en la optimización de la gestión de políticas TI. Abordar estos desafíos es fundamental para que las organizaciones cubanas puedan avanzar hacia una gestión de políticas TI más efectiva y estratégica. La superación de estas barreras podría contribuir significativamente a la alineación de las estrategias de TI con los objetivos de la organización y a una gestión más eficaz de los recursos tecnológicos.

Partiendo de este análisis se fórmula el siguiente **problema de investigación:**

¿Cómo gestionar las políticas de TI en las organizaciones cubanas?

El **objeto de estudio** de la investigación se centra en el proceso de gestión de políticas de TI en las organizaciones, teniendo como **campo de acción**, los sistemas de gestión de políticas de TI en las organizaciones cubanas.

Para solucionar el problema planteado se propone como **objetivo general** desarrollar un sistema informático para la gestión de las políticas de TI en las organizaciones cubanas.

A partir del **objetivo general** definido, se derivan los siguientes **objetivos específicos**:

- Analizar los referentes teóricos y metodológicos relacionados con sistemas informáticos para la gestión de políticas de TI.
- Diseñar un sistema para la gestión de políticas TI en las organizaciones cubanas.
- Implementar un sistema para la gestión de políticas TI en las organizaciones cubanas.
- Realizar pruebas al sistema desarrollado.

Para dar cumplimiento a los **objetivos específicos** se proponen como **tareas de investigación**:

- Revisión de los conceptos y tecnologías requeridas para esclarecer los conocimientos sobre los sistemas de gestión informatizada de políticas de TI.
- Definición del perfil y diseño de la investigación.
- Descripción y análisis del estado actual de las soluciones informáticas para la gestión de las políticas de TI en las organizaciones.
- Estudio de los aspectos de la metodología de desarrollo de software que se aplicará en la investigación.
- Descripción de las herramientas, lenguajes y tecnologías a emplear.
- Investigación de las necesidades y requerimientos del sistema, así como los objetivos que debe cumplir.
- Planificación del proceso de implementación.
- Elaboración de los casos de prueba para la validación del sistema.

Con el fin de dar cumplimiento a los objetivos y las tareas propuestas se emplearon como **métodos de investigación**:

Métodos teóricos:

- Análisis y síntesis: se empleó para la construcción y desarrollo de la teoría, profundización en el tema y la sistematización del conocimiento.
- Histórico-Lógico: se empleó con el objetivo de entender los antecedentes y las tendencias actuales de la gestión de políticas de TI, y su uso por las organizaciones, así como su importancia.
- Modelado: se empleó en la generación de los artefactos que permitan lograr una mejor comprensión entre el equipo de desarrollo y las personas relacionadas.

Métodos empíricos:

- Entrevista: se empleó con el objetivo de recabar datos relacionados con la gestión de políticas de TI, en el que se estableció un proceso de comunicación entre el entrevistador y el entrevistado.

Estructura del documento:

El documento está estructurado en tres capítulos; en el primero se exponen los aspectos generales de la fundamentación teórica del trabajo, iniciando con la conformación del perfil de la investigación; se abordan conceptos relacionados con la gestión de políticas de TI, se incluye un análisis sobre el diseño de políticas de TI, así como de los sistemas existentes para la gestión de políticas, se describen las herramientas, tecnologías, lenguajes de programación y metodología de desarrollo a emplear para la puesta en marcha del proceso de desarrollo. En el segundo capítulo se muestra la propuesta de solución, se realiza el levantamiento de requisitos, las historias de usuarios, el plan de iteraciones, la arquitectura y los patrones de diseño. En el tercero se presenta elementos correspondientes a las pruebas realizadas al sistema.

CAPÍTULO I: FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS SOBRE EL OBJETO DE ESTUDIO

Introducción

En este capítulo se realiza una descripción de los distintos conceptos relacionados con las Políticas de Tecnologías de la Información, su proceso de diseño, así como la Gestión de Redes Basada en Políticas. Además, recoge información esencial de sistemas homólogos. También, aborda temas como las metodologías y las herramientas de desarrollo a emplear.

1.1. Conceptos asociados

1.1.1. Políticas de Tecnologías de la Información

Las políticas de tecnología de la información son un conjunto de reglas que definen el acceso y el uso de los recursos de la infraestructura de TI, divididas en políticas automatizadas y no automatizadas. Las políticas automatizadas son políticas que se pueden implementar y ejecutar utilizando herramientas y sistemas de TI, mientras que las políticas no automatizadas requieren intervención humana para garantizar el cumplimiento. Es importante enfatizar que las políticas de TI deben ser flexibles y adaptarse a los cambios tecnológicos y organizacionales, así como a las necesidades y objetivos de la empresa. Además, la información debe comunicarse a todos los miembros de la organización de forma clara y comprensible para garantizar el cumplimiento y minimizar los riesgos (Peña Casanova & Anías Calderón, 2019).

1.1.2. Acuerdos de nivel de servicio

Acuerdo de Nivel de Servicio (SLA por sus siglas en inglés) es un contrato entre un proveedor de servicios y su cliente que establece el nivel de calidad de ese servicio. Este acuerdo permite consensuar el nivel de calidad del servicio y definir aspectos como tiempo de respuesta, disponibilidad de tiempo, documentación disponible, personal asignado y otros. Su objetivo es identificar y definir las necesidades del cliente y al mismo tiempo establecer expectativas claras sobre la capacidad del proveedor para satisfacerlas. Debe ser un documento flexible y verificable ya que las necesidades y condiciones pueden cambiar con el tiempo. Además, para su efectividad, es importante que ambas partes lo revisen y aprueben antes de prestar el servicio. Las políticas de TI y los SLA se complementan entre sí porque las políticas definen el marco regulatorio y los requisitos generales para la seguridad y el uso adecuado de los recursos tecnológicos, mientras que los SLA definen las obligaciones específicas de calidad y desempeño del proveedor de servicios de TI para con su cliente (García-Peñalvo, 2021).

1.1.3. Diseño de políticas de TI

El proceso de diseño de una Política de Seguridad no debe hacerse a la ligera. Este es un proceso que requiere un enfoque serio para obtener resultados efectivos. Durante este, es necesario realizar un análisis de seguridad de negocios similares y evaluar los costos/beneficios de implementar políticas de seguridad. Se debe realizar un inventario de los recursos de software y hardware, así como de los servicios que brinda la empresa o entidad, para clasificarlos según su importancia y criticidad (Duque Méndez, 2002).

A continuación, deben ser definidos los objetivos relacionados con la seguridad informática, enfocándose en proteger a la organización de amenazas que afecten sus operaciones, la integridad de sus equipos y la información. Para ello es necesario analizar las posibles y reales amenazas a las que se puede enfrentar la organización. Una vez completados estos pasos, se crea una política de seguridad, la cual debe emitirse en un documento claro y sencillo. Además, todos los usuarios de tecnologías de la información (TI) deben ser informados de las obligaciones que tienen con respecto a la seguridad de los recursos de TI y recibir orientación sobre cómo responder a posibles amenazas (Duque Méndez, 2002).

1.1.4. Gestión de redes basada en políticas

Las políticas de gestión de red las definen los administradores de red para gestionar y controlar las redes de forma eficaz. Estos agrupan recursos, ya sean personas o dispositivos, y determinan cómo deben configurarse, controlarse e interactuar entre sí. Este enfoque permite garantizar la calidad del servicio a través de reglas o sistemas automatizados que brindan diferentes clases de servicio en función de las prioridades, ya sea de usuario, servicio o red. El código de declaración de política puede residir en varios elementos de la red, ya sea el sistema de administración de red, enrutadores, conmutadores, servidores, bases de datos, aplicaciones o sistemas de almacenamiento (Reyes & Barba, s. f.).

La gestión de redes basada en políticas, dice que un modelo de política debe de contar con tres elementos: una regla que define el objetivo o el resultado de la política, una condición que son los criterios que se deben de cumplir para poder aplicar la política y por ultimo una acción que representa la política la cual se va a aplicar cuando se cumplan las condiciones (Yepez Rodríguez & Díaz Machuca, 2011).

1.1.4.1. Representación del proceso de gestión de redes basado en políticas

La administración de redes basada en políticas es una herramienta que permite automatizar y auto administrar redes. Esta se basa en tres componentes principales: punto de decisión de políticas (PDP), puntos de implementación de políticas (PEP) y repositorio de políticas. El PDP es responsable de recopilar y almacenar todas las políticas de red. Proporciona una interfaz en la que se definen y configuran las políticas, que posteriormente son traducidas a un lenguaje que los elementos de la red puedan entender. Además, el PDP toma decisiones basadas en políticas ya definidas y almacenadas en el repositorio de políticas. Los PEP son los encargados de hacer cumplir las políticas en toda la red. Antes de aplicar una política, buscan en el PDP instrucciones sobre qué política deben aplicar. También traducen políticas en comandos que los elementos de la red pueden comprender y ejecutar. Finalmente, hay un almacén de políticas que almacena todas las políticas definidas. Este repositorio proporciona acceso rápido y centralizado a todas las políticas, lo que facilita su administración y actualización (Casanova & Calderón, 2020).

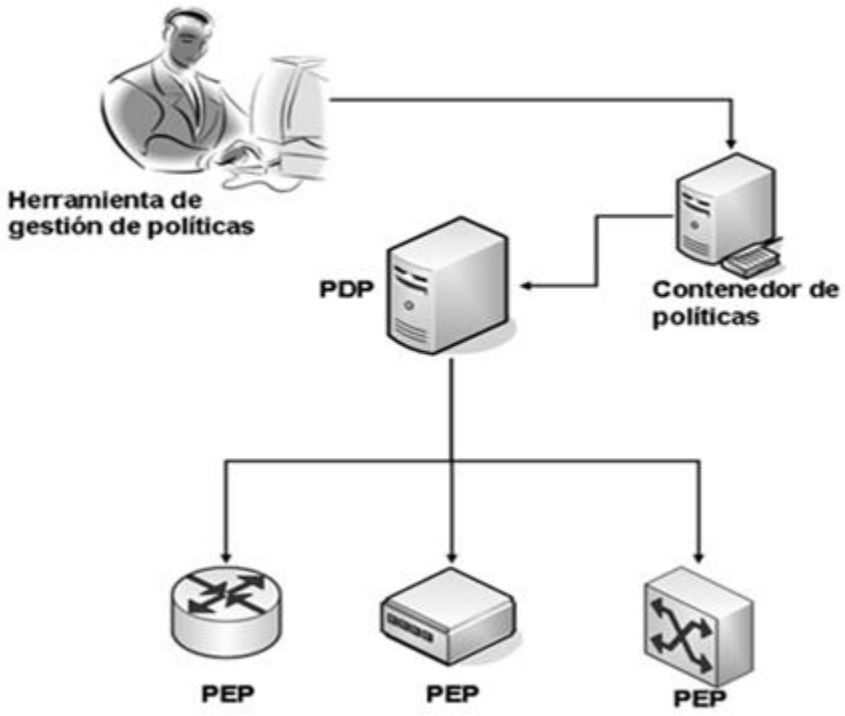


Figura 1: Representación de la arquitectura de PBNM (Casanova & Calderón, 2020).

1.2. Descripción de los sistemas homólogos

1.2.1. ServiceNow

Es una plataforma de gestión de servicios basada en la nube que permite a los usuarios automatizar y gestionar de manera eficiente sus procesos comerciales. Además de integrarse fácilmente con otras aplicaciones y sistemas existentes, facilita la colaboración y la sincronización de datos en toda la empresa. La plataforma ofrece un alto nivel de flexibilidad de personalización para que pueda adaptarse a las necesidades específicas de cada organización. Un módulo destacado de ServiceNow es la Gestión de políticas de TI, que le permite crear y aplicar políticas de seguridad y cumplimiento a su entorno de TI (Aleixandre Sanz, 2022).

Este módulo ayuda a las organizaciones a garantizar el cumplimiento de los estándares de seguridad y la protección de datos confidenciales. ServiceNow está diseñado teniendo en cuenta la facilidad de uso y la experiencia del usuario, con una interfaz intuitiva y fácil de usar que acorta la curva de aprendizaje y aumenta la productividad. Aunque es una plataforma avanzada, la amplia gama de características, funciones y beneficios justifican su costo. Además, también cuenta con un sólido sistema de soporte técnico y una comunidad de usuarios activa para compartir conocimientos y soluciones. La seguridad es una de las principales prioridades en ServiceNow, con medidas mejoradas de protección de datos y auditorías periódicas (Aleixandre Sanz, 2022).

1.2.2. Confluence

Es la herramienta de gestión del conocimiento de Atlassian que permite a los equipos crear, colaborar y organizar el trabajo en un solo lugar. Con una amplia gama de funciones, integración con otras herramientas y opciones de personalización, Confluence es un sistema completo y versátil. Además, su facilidad de uso intuitiva promueve una rápida adopción por parte de los usuarios. En cuanto a costos, Confluence ofrece una variedad de planes que se adaptan a las necesidades de cualquier empresa. Para soporte técnico, Atlassian proporciona documentación confiable y una comunidad de usuarios activa (Fernández Lobán, 2023).

Por otro lado, la seguridad de la información es una prioridad absoluta, ya que Confluence le permite gestionar políticas y procedimientos relacionados con TI para garantizar la confidencialidad y protección de los datos. Confluence también destaca por su capacidad para gestionar eficazmente las políticas de seguridad relacionadas con TI. Con esta herramienta, los equipos de TI pueden crear y mantener una base de conocimientos centralizada para almacenar y compartir de forma segura políticas y procedimientos de seguridad (Fernández Lobán, 2023).

1.2.3. Jira

Es una herramienta de gestión de proyectos y problemas desarrollada por Atlassian que proporciona funcionalidad y flexibilidad para ayudar en la gestión de políticas de TI. Aunque Jira no está diseñado específicamente para esta tarea, puede utilizarse como un complemento eficaz. Los equipos pueden asignar tareas relacionadas con la creación, revisión y aprobación de políticas, y realizar un seguimiento del progreso mediante la configuración de flujos de trabajo personalizados. Jira ofrece una amplia gama de funciones que permiten a los usuarios configurar parámetros relacionados con eventos y gestionarlos de forma eficaz (Marcos & Mobrici, 2020).

Además, se integra con otras herramientas de Atlassian como Confluence para una colaboración perfecta. Jira es personalizable para que puedas adaptar la herramienta a las necesidades específicas de cada equipo. En términos de costo, Jira ofrece una variedad de planes que se adaptan a los requisitos comerciales. Atlassian ofrece un sólido soporte técnico, documentación detallada y una comunidad de usuarios activa para compartir conocimientos y experiencias. En términos de seguridad, Jira ofrece opciones de autenticación, permisos de acceso y cifrado de datos para garantizar la confidencialidad y protección de la información (Marcos & Mobrici, 2020).

1.2.4. Consideraciones del análisis de los sistemas existentes

Tabla 1: Comparativa entre los sistemas homólogos.

	ServiceNow	Confluence	Jira
Gestión de políticas TI	SI	SI	SI
Facilidad de uso	Fácil	Fácil	Difícil para los no desarrolladores
Rendimiento	Rendimiento sólido	Puede ser lento con grandes volúmenes de datos	Rápido
Confidencialidad y Protección de datos	SI	SI	SI

Almacenamiento Centralizado	SI	SI	NO
Plataforma	Privativo	Privativo	Privativo
Capacidad de realizar Búsquedas	SI	NO	SI

Después de analizar cada uno de estos sistemas, prestando especial atención a sus módulos para la gestión de políticas de TI, se puede decir que en su mayoría son fáciles de usar y garantizan la confidencialidad y protección de los datos. Pero, el gran problema que presentan es que ninguno es de código abierto, lo que impide su uso en las organizaciones cubanas.

Por tal motivo, se propone el desarrollo de un sistema informático para la gestión de políticas de TI, aplicando algunas de las características de estos sistemas investigados, como la capacidad de almacenar de forma centralizada las políticas mediante un repositorio, como lo realiza ServiceNow y Confluence. Esto facilita el acceso rápido y garantiza que estén disponibles para todos los usuarios. Además, se buscará incorporar la capacidad de realizar búsquedas, como se aplica en ServiceNow y Jira, para facilitar la búsqueda de elementos dentro del sistema. Con esto, se busca proporcionar una solución accesible para la gestión de políticas de TI en el contexto cubano.

1.3. Metodologías para el desarrollo

La metodología de desarrollo de software es un enfoque sistemático y disciplinado para desarrollar, operar y mantener software. Es un conjunto de pautas y prácticas que ayudan a los equipos de desarrollo a completar proyectos de software de manera eficiente y efectiva, asegurando la calidad y confiabilidad del producto final. La metodología del software se basa en principios y técnicas probados que han evolucionado y mejorado con el tiempo. Estas técnicas incluyen planificación de software, análisis de requisitos, diseño, implementación, pruebas y mantenimiento. Cada una de estas etapas tiene sus propias actividades y tareas específicas que deben completarse en un orden lógico y secuencial (Sommerville, 2011).

1.3.1. Metodología de desarrollo a utilizar

Extreme Programming (XP) es una metodología de desarrollo que sigue un enfoque orientado a objetos. Este paradigma incluye un conjunto de reglas y prácticas divididas en cuatro actividades

estructurales. La primera actividad de XP es la planificación, que comienza con la recopilación de requisitos a través de las historias de usuario. Esto ayuda al equipo a comprender el negocio del software y obtener una visión clara de las funciones principales y requeridas. La siguiente actividad es el diseño, donde el principio es dar preferencia a un diseño simple sobre uno más complejo. XP promueve el uso de mapas de Clase-Responsabilidad-Colaboración (CRC) como una herramienta eficaz para pensar en el software en un contexto orientado a objetos. Durante la fase de codificación, XP recomienda que dos personas trabajen juntas en una estación de trabajo. Esto le permite resolver problemas en tiempo real y garantizar la calidad del código de inmediato. Finalmente, la actividad de prueba es crucial en XP. Se enfatiza el uso de pruebas unitarias al código. Estas pruebas deben implementarse utilizando un marco que permita la automatización y facilite la realización de pruebas de regresión con cada cambio en el código. Dado que XP fomenta el rediseño frecuente, se considera esencial realizar pruebas de aceptación del software. Estas pruebas las especifica el cliente y se derivan de historias de usuario (Pressman, 2010).

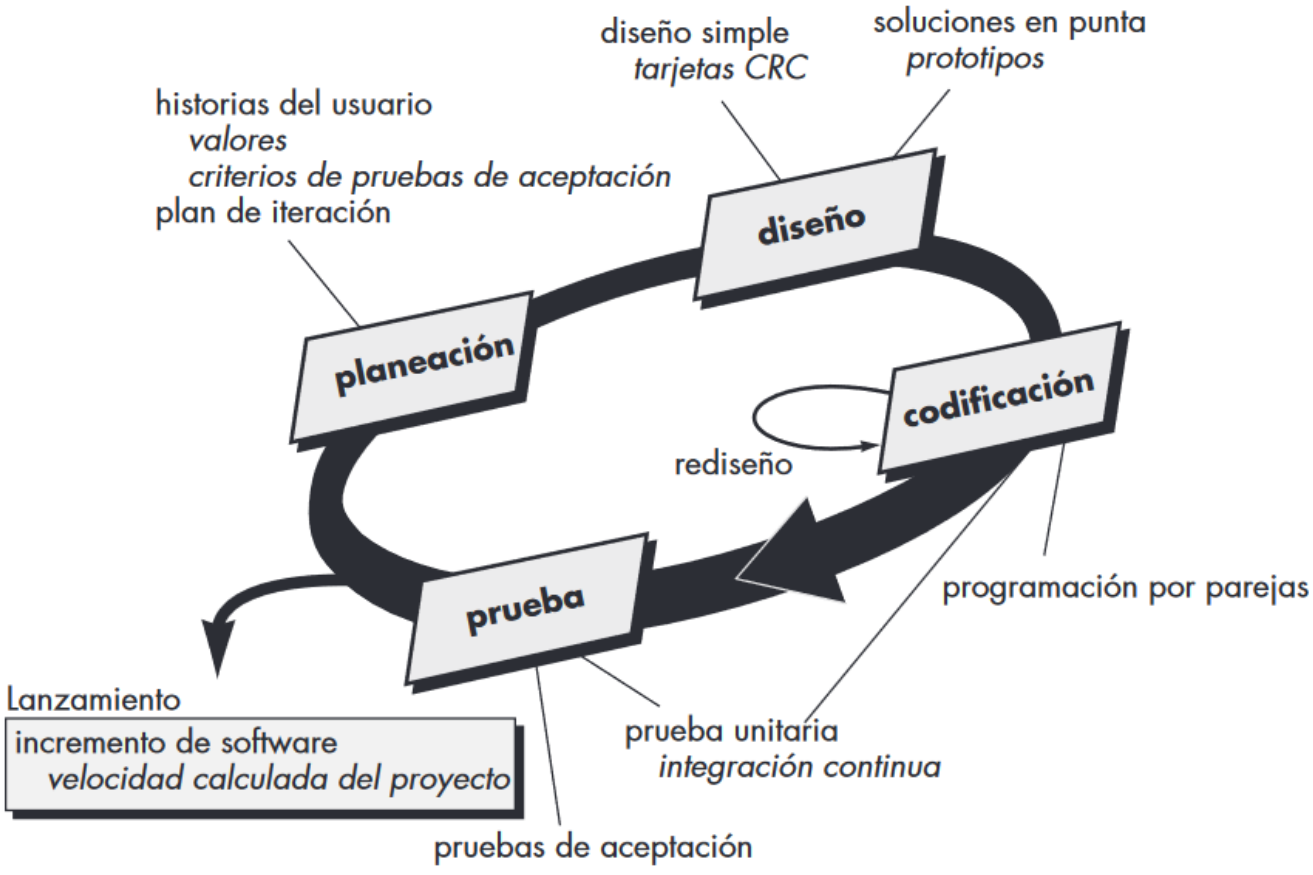


Figura 2: Proceso de desarrollo de software de la metodología XP (Pressman, 2010).

La metodología XP se presenta como una alternativa altamente ventajosa para desarrollar este proyecto por sus especificaciones. XP facilita la adaptación a los requisitos del proyecto. Esta característica, en una situación como la actual de este proyecto, donde los requisitos no están claramente definidos desde el principio, la convierte en una metodología especialmente valiosa. Utilizando un enfoque iterativo, XP permite al equipo de desarrollo comenzar con los requisitos existentes y luego adaptar el producto a medida que aprenden más sobre los requisitos. Otro punto a considerar son las pruebas unitarias, que verifican la funcionalidad de cada componente del software individualmente, lo que promueve la detección temprana de errores y la integración continua que garantiza que cada componente del software funcione como un todo, lo que reduce la posibilidad de problemas de compatibilidad y mejora la calidad del producto final.

1.4. Herramientas, lenguajes y tecnologías

El desarrollo continuo de las TICs propicia el perfeccionamiento de las herramientas informáticas, por lo que se hace necesario considerar las posibles herramientas, lenguajes y tecnologías a utilizar para el desarrollo del sistema. Seguidamente se describen las empleadas en la presente investigación, realizando un análisis de las principales características de cada una de ellas.

1.4.1. Herramienta de modelado

Visual Paradigm (versión **17.0.20**) es una herramienta CASE: Ingeniería de Software Asistida por Computación. Se caracteriza por tener disponibilidad en múltiples plataformas (Windows, Linux), poseer un diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad, el uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación. La misma está diseñada para una amplia gama de usuarios, incluyendo ingenieros de software, analistas de sistema, analistas de negocio, arquitectos de sistemas y quienes estén interesados en la construcción de sistemas de software de forma fiable a través de la utilización de un enfoque Orientado a Objetos (De Arma-Hernández & Sablón-Fernández, 2019).

1.4.2. Lenguajes de programación

Python (versión **3.8.10**) es un lenguaje de programación de propósito general que se usa ampliamente en la actualidad. Una de las características más destacadas de Python es su enfoque en la legibilidad del código y en la reducción del número de líneas necesarias para expresar determinada funcionalidad. Esto facilita a los programadores escribir y comprender programas. Python es un lenguaje de programación de tipo dinámico, lo que significa que no es necesario declarar explícitamente el tipo de datos de una variable. Esta característica acelera el desarrollo y reduce la cantidad de código necesario

para realizar tareas comunes. Además, Python cuenta con una variedad de implementaciones e intérpretes disponibles para diferentes sistemas operativos, asegurando su portabilidad y facilidad de uso en diferentes entornos (Sharma et al., 2020).

JavaScript (versión **ES6**) es un lenguaje de programación para el desarrollo de sitios web, tanto del lado del cliente como del lado del servidor utilizando principalmente C[1]. Es un lenguaje potente y a la vez sencillo que, al integrarse en un motor web, ejecuta rápidamente su sintaxis. Es versátil tanto para el desarrollo web dinámico como el desarrollo de aplicaciones móviles. Es multiplataforma y soportado por todos los dispositivos móviles actuales. Elimina el peso adicional que puede generar un framework, reduciendo así el consumo de ancho de banda. Se vincula con HTML y CSS de una manera muy fluida y transparente y además es la base de los marcos más populares (Luna, 2019).

1.4.3. Lenguaje de marcado

HTML5 (Lenguaje de Marcas de Hipertexto, del inglés *HyperText Markup Language*) es el componente más básico de una web y define su significado y la estructura de su contenido. Junto a este se utilizan generalmente otras tecnologías para describir la apariencia de una página web (CSS) o la funcionalidad (JavaScript). En su corazón, HTML es un lenguaje muy sencillo compuesto de elementos, que se pueden aplicar a piezas de texto para darles un significado diferente en un documento, estructura un documento en secciones lógicas, y añade imágenes y vídeos en una página (Tabarés, 2021).

1.4.4. Hoja de estilo

CSS3, que significa Cascading Style Sheets, es un lenguaje de estilo que define cómo se muestran las páginas web. Como uno de los lenguajes centrales de la web, CSS se utiliza hoy en día en casi todos los sitios web para mejorar la experiencia web. En esta nueva versión introduce nuevas propiedades para trabajar con colores y sombras e incluyen la capacidad de utilizar fuentes personalizadas. Otra característica importante es que permite la creación de efectos visuales dinámicos y atractivos sin necesidad de utilizar JavaScript o Flash. Ha sido una revolución en la World Wide Web y es un lenguaje que todo aquel que trabaja en diseño web debe aprender (Olsson, 2019).

[1] *Lenguaje de programación* (Chávez, 2019).

1.4.5. Framework de desarrollo

Django (versión **4.2.6**) es un marco de desarrollo web basado en Python. Permite definir proyectos y aplicaciones web, genera automáticamente archivos de configuración y proporciona una capa de mapeo relacional de objetos para la interacción con bases de datos relacionales. También incluye herramientas para visualización de datos mediante plantillas HTML y para operaciones CRUD (del inglés create, read, update y delete) mediante plantillas de administración. Su licencia gratuita y su sencillez en el desarrollo hace que sea muy utilizado y existan multitud de extensiones y servidores de acceso gratuito (Vidal-Silva et al., 2021).

Bootstrap 5 es una biblioteca de código abierto desarrollada por Twitter para crear sitios web responsivos utilizando HTML, CSS y JavaScript. Con el paso de los años ha ido evolucionando hasta la versión 4. Esta última sufrió dos cambios importantes. En primer lugar, se realizó una migración del pre-procesamiento de CSS de LESS (Leaner Style Sheets) a SASS (Syntactically Awesome Stylesheets), lo que proporciona más flexibilidad y personalización en la configuración de estilo y, en segundo lugar, el cambio a CSS-Flexbox permite una mejor manipulación de elementos y diseños flexibles. Bootstrap también ofrece una amplia gama de componentes listos para usar que hacen que el desarrollo front-end sea más rápido, al proporcionar una base sólida y reutilizable. Además de ser compatible con los navegadores web más populares, garantiza una experiencia consistente en diferentes plataformas (Aryal, 2019).

1.4.6. Sistema de control de versiones

Git (versión **2.25.1**) es un sistema de control de versiones distribuido, libre y de código abierto, diseñado para manejar proyectos pequeños o muy grandes con rapidez. Este es un sistema centrado en la eficiencia, la confiabilidad y compatibilidad del mantenimiento de versiones en proyectos con cualquier tamaño de manera eficiente y rápida. Su objetivo es mantener el registro de cambios en los archivos digitales, incluida la coordinación del trabajo realizado por varias personas en un archivo conjunto en el repositorio de código. Actualmente, este es el sistema de control más utilizado y tiene soporte de grandes empresas como Microsoft, Google y Twitter (Bustos Tejo, 2022).

La característica Git que realmente hace que se aparte de casi todos los otros gestores de código fuente es su modelo de ramificación. Permite tener múltiples ramas locales que pueden ser totalmente independientes entre sí. La creación, la fusión y la supresión de esas líneas de desarrollo toma segundos.

Esto significa que se pueden ejecutar acciones como las indicadas en (Torvalds y Hamano 2010):

- ✓ Conmutación de contexto sin fricción. Crear una rama para probar una idea, comience varias veces, cambie de nuevo a la rama de donde se ramificó, aplique un parche, cambie de nuevo a la rama donde está experimentando y fíjelo.
- ✓ Reglas Basadas en el Rol. Tener una rama que siempre contiene solo lo que va a la producción, otra que se fusiona en el trabajo para la prueba, y varias más pequeñas para el trabajo diario.
- ✓ Flujo de trabajo basado en funciones. Crear nuevas ramas para cada nueva función en la que esté trabajando, de modo que pueda cambiar sin problemas entre ellas y, a continuación, suprimir cada una de las ramas cuando se fusione en su línea principal.
- ✓ Experimentación desechable. Crear una rama para experimentar, darse cuenta de que no va a funcionar, y solo eliminarlo, abandonar el trabajo.

1.4.7. Gestor de Base de Datos

PostgreSQL (versión **15.2**) es un potente sistema de base de datos relacional de objetos de código abierto que aprovecha y amplía el lenguaje SQL, combinado con muchas funciones que almacenan y escalan de forma segura las cargas de trabajo de datos más complicadas. Este administrador de bases de datos se ha ganado una buena reputación debido a su arquitectura, confiabilidad, integridad de datos y extensibilidad. Se ejecuta en todos los principales sistemas operativos y tiene complementos potentes, lo que la convierte en la base de datos de código abierto preferida por muchas personas y organizaciones (Chávez, 2020).

1.4.8. Entorno de desarrollo integrado

Visual Studio Code (versión **1.82.2**) como Entorno de Desarrollo Integrado (IDE por sus siglas en inglés) por ser un editor de código liviano y multiplataforma para escribir aplicaciones web y en la nube. Este es altamente personalizable, lo que significa que se podrá adaptar a las necesidades y preferencias de los desarrolladores. Presenta una interfaz intuitiva y muy fácil de usar facilitando la navegación y el acceso a sus distintas funcionalidades. Ofrece a los desarrolladores soporte integrado para múltiples idiomas. Además de su integración con herramientas de control de versiones como Git permitiendo la colaboración entre los desarrolladores y el seguimiento de los cambios en el código (Code, 2019).

1.4.9. Servidor Web

Apache (versión **2.4**) se caracteriza por ser estable, multiplataforma, modular y altamente configurable, es decir, que puede adaptarse a diferentes necesidades. Además, es altamente personalizable y se puede ampliar con módulos adicionales para satisfacer las necesidades específicas del usuario. Otra

característica importante de Apache es su compatibilidad con varios sistemas operativos, incluidos Linux, Unix, Windows y otros. Esto permite a los usuarios implementar Apache en el sistema operativo de su elección, haciéndolo flexible y adaptable a diferentes entornos informáticos. También es de código abierto y gratuito esto hace que sea una opción popular para servidores web en una amplia variedad de entornos, desde pequeñas empresas hasta grandes corporaciones (Montoya et al., 2013).

Conclusiones del capítulo

En este capítulo fueron analizados algunos conceptos relacionados con la investigación, lo que permitió ampliar los conocimientos relacionados con la gestión de las políticas de TI. Se realizó un estudio de los sistemas homólogos existentes con el objetivo de obtener una mayor comprensión y claridad sobre los diversos aspectos que deben considerarse al implementar el sistema. Se identificaron las herramientas, tecnologías y lenguajes de programación relevantes para este proyecto, seleccionando aquellas que cumplan con las características necesarias para la puesta en práctica de la propuesta de solución y se seleccionó la metodología de desarrollo ágil (XP) para facilitar el trabajo y ofrecer al cliente el mejor resultado posible.

CAPÍTULO II: PLANIFICACIÓN, DISEÑO Y CODIFICACIÓN DE LA SOLUCIÓN PROPUESTA AL PROBLEMA CIENTÍFICO

Introducción

En este capítulo se plantea la propuesta de solución, el análisis de la propuesta, las funcionalidades y restricciones del sistema, las historias de usuario y aspectos de diseño (el estilo y patrón arquitectónico, los patrones de diseño). Se plantea el modelo de datos que contiene las relaciones entre las entidades que conforman la base de datos y se propone la arquitectura de red y los protocolos que le darán soporte al sistema.

2.1. Descripción de la propuesta de solución

Luego de realizar un estudio del arte, de sistemas relacionados con la gestión de políticas de TI y seleccionado cada una de las herramientas, tecnologías y lenguajes de programación para el desarrollo del sistema se está en condiciones de ejecutar un proceso de diseño e implementación de un Sistema para la Gestión de Políticas de TI (SGPTI). Se propone el desarrollo de un sistema basado en la web, en el cual, el usuario con el rol de administrador, una vez registrado en el sistema, será capaz de gestionar cada una de las políticas. Contará con un módulo desde el cual puedan ser cargados, por el propio administrador, los archivos que darán respuesta a cada política, estos serán almacenados en un repositorio de archivos. También, permitirá la gestión de eventos que no son más que las condiciones bajo las cuales se va a crear una política. Además, el usuario que no posea el rol de administrador, podrá acceder al sistema, ver las políticas que hay registradas y descargar el archivo relacionado a cada política; para así poder utilizarlo para su beneficio y el d la empresa. Con la puesta en práctica de esta solución se planea poder gestionar las políticas de TI de una determinada organización.

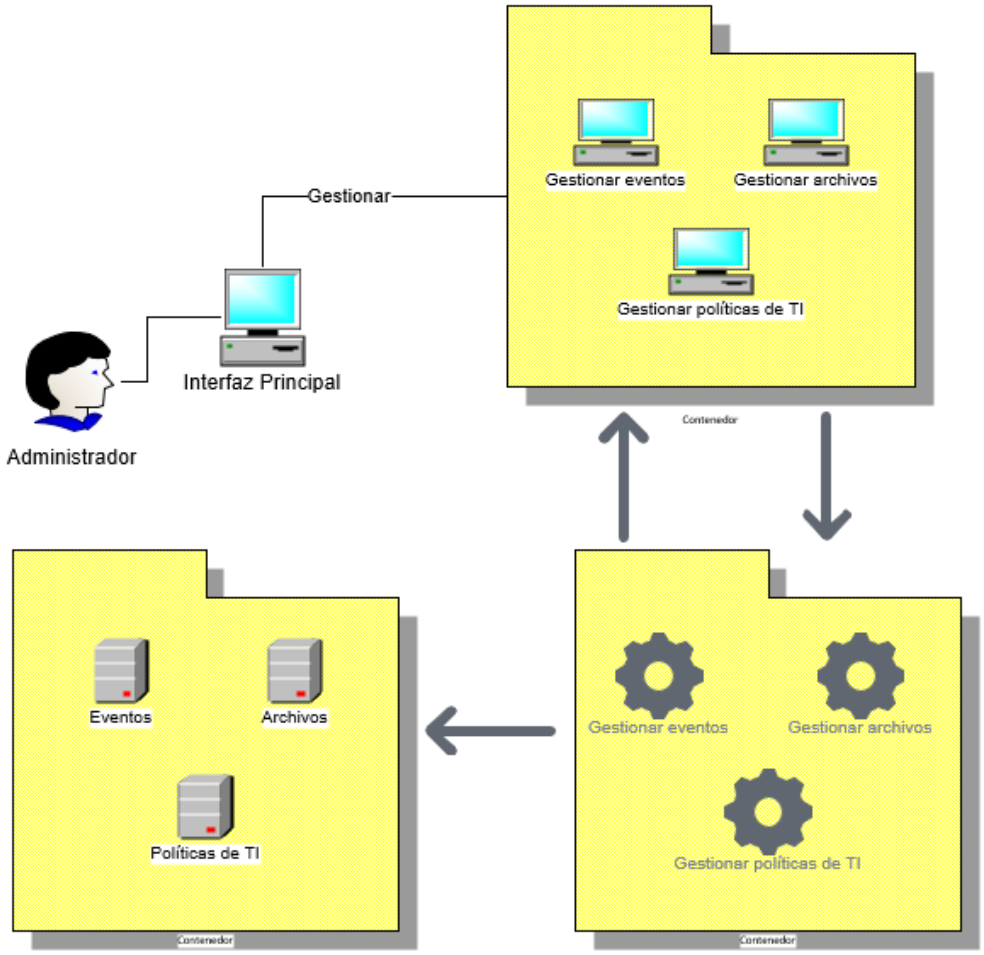


Figura 3: Representación gráfica de la propuesta de solución.

2.2. Restricciones del sistema

Las restricciones del sistema tienen que ver con características que de alguna forma puedan limitar el sistema. Son condiciones que se deben considerar para lograr la entrega efectiva del sistema.

A continuación, se muestran las restricciones del sistema:

Tabla 2: Restricciones.

No R	Tipo.	Descripción.
R 1 R 2	Rendimiento	• Debe de ser capaz de realizar las operaciones de forma eficaz. • Responder a las solicitudes del usuario en un tiempo de respuesta aceptable.

R 3	Escalabilidad	• Debe de ser capaz de manejar un creciente aumento en el número de datos y usuarios.
R 4	Seguridad	• Es obligatorio que los usuarios se autentiquen para interactuar con el sistema.
R 5 R 6 R 7	Usabilidad	• Debe ser de fácil entendimiento, con interfaces agradables y fácil de usar. • Los elementos de la interfaz como botones, iconos y terminologías deben ser consistentes en todo el sistema. • En caso de algún error el sistema debe ser capaz de proporcionar un mensaje de error.
R 8	Portabilidad	• Debe ser compatible con Firefox 118.0.2 (32-bit), Chrome 109 y Safari 16.5.

2.3. Funcionalidades:

Las funcionalidades son las capacidades o características específicas que un sistema debe tener para cubrir las necesidades de los usuarios.

A continuación, se enumeran las funcionalidades definidas para el sistema:

Tabla 3: Funcionalidades.

Funcionalidades
1. Autenticar Usuario
2. Gestionar eventos
3. Gestionar archivos
4. Gestionar políticas de TI
5. Buscar por criterios

2.4. Historias de Usuario

Entre los artefactos que define la metodología seleccionada se encuentran las historias de usuario que son utilizadas para especificar las funcionalidades que brindará el sistema. Las historias de usuarios

se utilizan como herramienta de comunicación entre clientes y equipos de desarrollo. Cada historia de usuario es una representación de un requerimiento de software escrito en una o dos frases utilizando el lenguaje común del usuario. Son herramientas que simplifican el proceso de gestión de requisitos, reduciendo la cantidad de documentación y tiempo requerido. El tratamiento de las historias de usuario es muy dinámico y flexible, en cualquier momento se pueden modificar, reemplazar por otras más específicas o generales o añadirse nuevas (Menzinsky et al., 2018).

Como resultado del análisis realizado se identificaron un total de 16 historias de usuario, a continuación, se muestran algunas de ellas:

Tabla 4: Historia de Usuario: Añadir evento al sistema.

Historia de Usuario	
Número: 2	Nombre: Añadir evento al sistema.
Usuario: Administrador.	
Prioridad en negocio: Alta (Alta/Media/Baja)	Riesgo en desarrollo: Alta (Alto/Medio/Bajo)
Puntos estimados: 2 puntos estimados.	Iteración asignada: 1ra iteración.
Programador responsable: Nayeli López-Castro Morejón.	
Descripción: 1- Objetivo: El módulo debe ser capaz de permitir añadir un evento en el sistema. 2- Flujo de la acción a realizar: - Cuando el usuario selecciona el botón “Crear Evento” se muestra una ventana para que introduzca los datos necesarios. - Si se selecciona el botón “Añadir” y todos los datos son correctos, se crea el evento y se cierra la ventana. - Si se selecciona el botón “Cancelar” se cierra la ventana	

Prototipo de Interfaz:

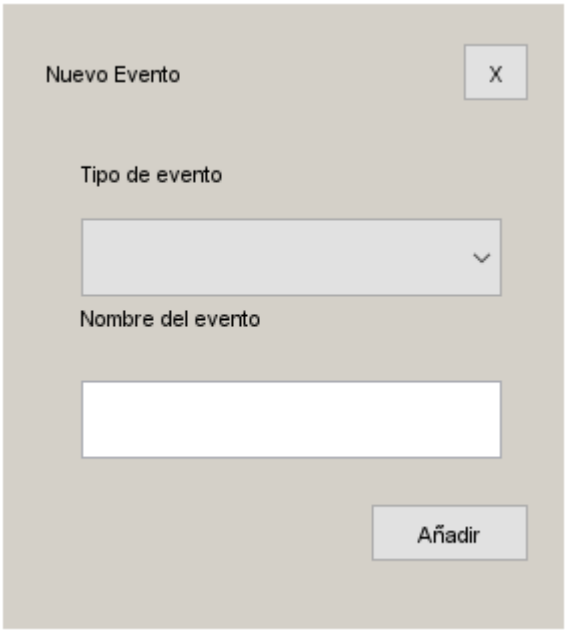


Tabla 5: Historia de usuario: Mostrar eventos existentes en el sistema.

Historia de Usuario	
Número: 3	Nombre: Mostrar eventos existentes en el sistema.
Usuario: Administrador.	
Prioridad en negocio: Alta (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 2 puntos estimados.	Iteración asignada: 1ra iteración.
Programador responsable: Nayeli López-Castro Morejón.	
Descripción: 1- Objetivo: El módulo debe ser capaz listar los eventos existentes en el sistema.	

2- Flujo de la acción a realizar:

- Cuando se cree un nuevo evento este se lista automáticamente, con los datos otorgados, en la tabla que contiene el listado de eventos
- Si existen más de 10 eventos se muestra la paginación y se listan los eventos en otra página.
- Si se selecciona el botón siguiente se muestran los eventos de la página que sigue.
- Si se selecciona el botón anterior se muestran los eventos de la página anterior.

Prototipo de Interfaz:

Tabla 6: Historia de usuario: Modificar eventos existentes en el sistema.

Historia de Usuario	
Número: 4	Nombre: Modificar eventos existentes en el sistema.
Usuario: Administrador.	
Prioridad en negocio: Alta (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 2 puntos estimados.	Iteración asignada: 1ra iteración.

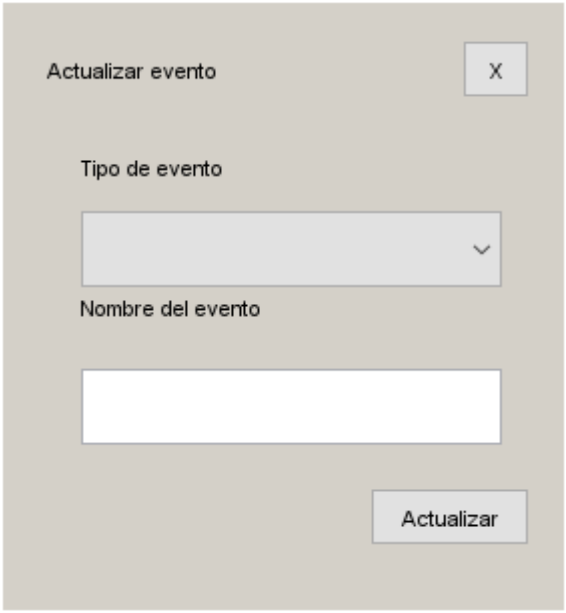
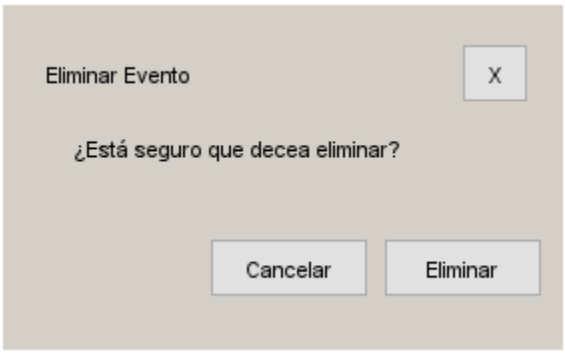
Programador responsable: Nayeli López-Castro Morejón.
Descripción: 1- Objetivo: El módulo debe ser capaz de modificar un evento seleccionado del sistema. 2- Flujo de la acción a realizar: - Cuando el usuario selecciona el botón de editar se muestra una ventana con los datos del evento seleccionado y pueden ser modificados. - Si se selecciona el botón de “Actualizar” y todos los datos son correctos, se modifican los datos y se cierra la ventana. - Si se selecciona el botón “Cancelar” se cierra la ventana.
Prototipo de Interfaz: 

Tabla 7: Historia de usuario: Eliminar eventos del sistema.

Historia de Usuario	
Número: 5	Nombre: Eliminar eventos del sistema.

Usuario: Administrador.	
Prioridad en negocio: Alta (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 2 puntos estimados.	Iteración asignada: 1ra iteración.
Programador responsable: Nayeli López-Castro Morejón.	
Descripción: 1- Objetivo: El módulo debe ser capaz de eliminar un evento seleccionado del sistema. 2- Flujo de la acción a realizar: - Cuando el usuario selecciona la opción de eliminar se muestra un mensaje para confirmar que desee eliminar. - Si se selecciona el botón “Eliminar” se elimina el evento. - Si se selecciona el botón “Cancelar” se cierra la ventana y no se elimina el evento.	
Prototipo de Interfaz: 	

2.4.1. Estimación del esfuerzo por historia de usuario

La estimación de esfuerzo es fundamental en la información que se encuentra en las historias de usuario. Esta permite que el equipo de desarrollo tenga una idea clara del tiempo y esfuerzo que requiere determinada funcionalidad. Una correcta estimación facilita la planificación y seguimiento del

proyecto, estableciendo plazos realistas y asignando los recursos adecuados para cumplir con los objetivos del proyecto.

Para la puesta en marcha de la propuesta de solución, se ejecutó un plan de estimación a cada una de las historias de usuario definidas. Para esto se estableció como patrón de estimación que cada punto corresponde a un día laboral, definido por 8 horas de trabajo al día y 5 días a la semana. A continuación, se muestra una tabla donde se relacionan los resultados obtenidos:

Tabla 8: Estimación de esfuerzo por historia de usuario.

Historia de Usuario	Puntos de estimación
Autenticar un usuario en el sistema.	1 punto
Añadir evento al sistema.	2 puntos
Mostrar eventos existentes en el sistema.	2 puntos
Modificar eventos existentes en el sistema.	2 puntos
Eliminar eventos del sistema.	2 puntos
Añadir archivo.	2 puntos
Modificar archivo del sistema.	2 puntos
Mostrar archivo del sistema.	2 puntos
Eliminar archivo del sistema.	2 puntos
Mostrar políticas de TI del sistema	2 puntos
Añadir políticas de TI al sistema.	2 puntos
Modificar políticas de TI del sistema.	2 puntos
Eliminar políticas de TI del sistema.	2 puntos
Buscar evento.	1 punto

Buscar archivos.	1 punto
Buscar políticas de TI.	1 punto

2.5. Plan de Iteraciones

A continuación, se muestra el plan de iteración, que es una herramienta utilizada en la metodología de desarrollo de software ágil, para organizar y gestionar el trabajo. En él se definirán las historias de usuario a cubrir en cada iteración. Este plan ayuda a mantener un enfoque claro y realista en el desarrollo del proyecto y permite que el equipo se centre en tareas específicas y evite una carga de trabajo excesiva. Al final de cada iteración, se entregarán mejoras funcionales del software.

Tabla 9: Plan de Iteraciones.

Iteraciones	Historias de Usuario	Tiempos por iteración
1	- Añadir evento al sistema. - Mostrar eventos existentes en el sistema. - Modificar eventos existentes en el sistema. - Eliminar eventos del sistema. - Autenticar usuario.	9 puntos estimados que representan una semana y cuatro días
2	- Añadir archivo. - Modificar archivo del sistema. - Mostrar archivo del sistema. - Eliminar archivo del sistema.	8 puntos estimados que representan una semana y tres días
3	- Añadir políticas de TI al sistema. - Modificar políticas de TI del sistema. - Mostrar políticas de TI del sistema. - Eliminar políticas de TI del sistema. - Buscar evento.	11 puntos estimados que representan dos semanas y un día

	- Buscar archivos. - Buscar políticas de TI.	

2.5.1. Plan de entregas

El plan de entregas se presenta a continuación. Esta estrategia tiene como objetivo establecer el orden y las fechas en las que se implementarán las diversas características o componentes del proyecto. Este plan permite organizar y priorizar las distintas tareas y funcionalidades a desarrollar, estableciendo plazos para cada una. Esto facilita la gestión del proyecto y permite controlar el progreso y el cumplimiento de los plazos. Así, se asegura la calidad y el correcto funcionamiento de cada componente o fase del proyecto.

Tabla 10: Plan de entregas.

Fecha de Inicio	Entrega 1ra Iteración	Entrega 2da Iteración	Entrega 3ra Iteración
05/10/2023	20/10/2023	03/11/2023	22/11/2023

2.6. Tarjetas CRC

Las tarjetas CRC (Clase-Responsabilidad-Colaboración) son una herramienta muy útil para identificar y organizar las clases relevantes en el desarrollo de software. Cada tarjeta pertenece a una clase, esta menciona su nombre, responsabilidades (que representan las operaciones que debe realizar) y colaboradores (estas son las otras clases con las que interactúa y de las que depende para realizar sus responsabilidades). El uso de CRC facilita la comprensión de un sistema y ayuda a tomar decisiones sobre su diseño y arquitectura (Lara & Figueroa, 2020).

Tabla 11: Tarjeta CRC de la clase políticas.

Clase: políticas	
Responsabilidad	**Colaboración**
• Gestionar las políticas de TI en el sistema	• eventos • archivos

Tabla 12: Tarjeta CRC de la clase archivos.

Clase: archivos	
Responsabilidad	**Colaboración**
• Gestionar los archivos que se almacenan en el sistema.	• Política

Tabla 13: Tarjeta CRC de la clase eventos.

Clase: eventos	
Responsabilidad	**Colaboración**
• Gestionar los eventos del sistema.	• Tipo Evento • Política

2.7. Arquitectura de software

La arquitectura del software se refiere a la estructura y el diseño global de un sistema, incluidos componentes, relaciones y principios que los regulan. Este es el nivel de diseño más alto y proporciona un marco claro para la implementación del sistema. La arquitectura es muy importante para un desarrollo de software efectivo, ya que permite comprender cómo se regulan y comunican los diferentes componentes del sistema (Blancarte, 2020).

2.7.1. Patrón arquitectónico

Los patrones arquitectónicos son soluciones probadas que se desarrollaron con el tiempo y se consideran buenas prácticas en el desarrollo del software. Permiten a los desarrolladores resolver de

manera efectiva los problemas generales, evitando la necesidad de inventar la rueda cada vez. Al aprender sobre la implementación de estos patrones, los desarrolladores reciben conocimiento acerca de cómo estructurar y organizar su código de manera efectiva. Esto les permite crear un sistema más flexible, escalable y fácil de mantener. Además, contribuyen a la reutilización del código y fomentan la modularidad, lo que facilita la cooperación entre los desarrolladores y reduce el tiempo de desarrollo (Alvarez et al., 2022).

2.7.1.1. Patrón arquitectónico Modelo-Vista-Plantilla

El Modelo-Vista-Plantilla (MVT según sus siglas en inglés) es un patrón de arquitectura de software utilizado por el framework del lenguaje de programación Python, Django, es una variante del Modelo-Vista-Controlador (MVC) pero con sus propias peculiaridades. Es un patrón de diseño de software que se utiliza para separar la lógica de presentación de un sistema de su lógica de negocio y sus datos. En este patrón, el "modelo" representa la capa de acceso a la base de datos, esta capa contiene toda la información sobre los datos: cómo acceder a estos, cómo validarlos, cuál es el comportamiento que tiene, y las relaciones entre los datos. La "vista" es la capa de la lógica de negocios, esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiada: se puede pensar en esto como un puente entre el modelo y las plantillas. Y la "plantilla" es la capa de presentación, esta capa es la que se emplea para mostrar los datos a los usuarios, generalmente la página HTML. El objetivo principal de MVT es facilitar la modularidad y el mantenimiento del código, separando claramente las responsabilidades y permitiendo la reutilización de componentes. Sin embargo, su incorrecta implementación puede agregar complejidad adicional al proyecto y requerir mayores esfuerzos de desarrollo y tiempo (Alvarez et al., 2022).

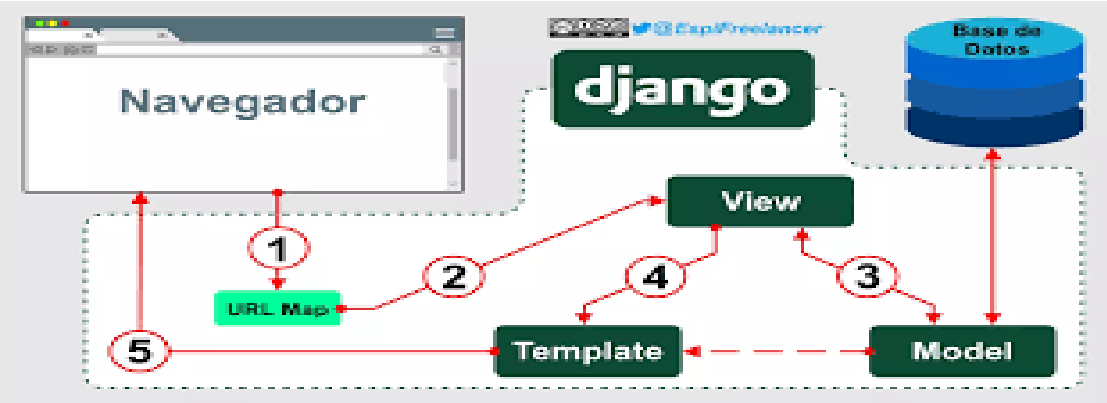


Figura 4: Representación del patrón Modelo-Vista-Plantilla (imagen tomada de https://es.quora.com/Qu%C3%A9-es-un-modelo-en-Django).

2.8. Patrones de diseño

Los patrones de diseño son un conjunto de artefactos que encapsulan el conocimiento sobre los problemas de diseño que surgen en un determinado contexto. Representan una solución reciclable para estos problemas en el diseño de software, permitiendo crear software de fácil mantenimiento. (Castaño et al., 2021).

2.8.1. Patrones Generales de Software para la Asignación de Responsabilidades

Los patrones de diseño de GRASP (por sus siglas en inglés), en el proceso de desarrollo de software, son una herramienta documental que rompe el paradigma de los flujos de información. En lugar de centrarse en la interacción entre funciones, se centra en funciones específicas, lo que facilita la gestión de conceptos fundamentales de análisis, diseño y programación en el marco de la programación orientada a objetos. El uso de estos patrones de diseño permite visualizar y comprender mejor las funciones específicas del sistema, lo que, a su vez, ayuda a identificar problemas e implementar soluciones (Ortega, 2021).

Los patrones GRASP que se utilizan en el sistema son:

- **Experto**: mediante su uso, se asignan responsabilidades a la clase que cuenta con la información necesaria (Larman & Valle, 2003). Este patrón se evidencia en el modelo FileScripts y en Event.
- **Creador:** permite crear objetos de una clase determinada (Larman & Valle, 2003). Este patrón se evidencia en la vista add_policy.
- **Alta cohesión**: este patrón caracteriza a las clases que posean responsabilidades estrechamente relacionadas, es decir, que no realicen un trabajo enorme (Larman & Valle, 2003). Se evidencia en las vistas delete_event, search_policy, all_events y evento_by_id.
- **Bajo acoplamiento**: posibilita que una clase no dependa mucho de otras clases (Larman & Valle, 2003). Este patrón se evidencia en las vistas delete_file y file_by_id.

2.8.2. Gang of Four

Los patrones de diseño GoF (por sus siglas en inglés) se consideran soluciones probadas y eficientes para problemas comunes en el diseño de software. Estos son herramientas poderosas que ayudan a mejorar la flexibilidad y la extensibilidad del código al proporcionar soluciones para problemas comunes en el diseño de software (Castaño et al., 2021).

Los patrones GoF que se utilizan en el sistema son:

- **Decorator:** es un patrón de diseño estructural que te permite añadir dinámicamente nuevas responsabilidades a un objeto, proporcionando una alternativa flexible a la extensión de una clase para aumentar la funcionalidad (Larman & Valle, 2003).

En este caso, el decorador @check_group asegura que las vistas solo sean accesibles para usuarios, pertenecientes a grupos, que posean permisos para acceder a ellas, redirigiendo automáticamente a los usuarios que no poseen permisos a una página de error. A continuación, se muestra su implementación:

```python
from functools import wraps
from django.http import Http404

nayeli lopez-castro morejon
def check_group(*group_names):
    nayeli lopez-castro morejon
    def _check_group(view_func):
        nayeli lopez-castro morejon
        @wraps(view_func)
        def wrapper(request, *args, **kwargs):
            user_groups = request.user.groups.values_list('name', flat=True)
            if not any(group_name in user_groups for group_name in group_names):
                raise Http404
            return view_func(request, *args, **kwargs)
        return wrapper
    return _check_group
```

Figura 5 Patrón Decorador.

2.9. Modelo de Datos

Un modelo de datos es un esquema que especifica las reglas y definiciones esenciales de los datos, esto proporciona una estructura para organizar y representar la información de una manera coherente y significativa. Este modelo se representa a través de un diagrama entidad-relación, que muestra las entidades (objetos o conceptos sobre los que se recopila información), los atributos (características o propiedades de las entidades) y las relaciones entre ellas. Con este diagrama se puede visualizar cómo se relacionan los diferentes elementos dentro del sistema de información. El modelo de datos es fundamental para diseñar sistemas de información eficientes y efectivos ya que proporciona una estructura lógica para organizar y representar la información, lo que facilita la gestión y manipulación

de los datos. Además, ofrece una forma práctica de implementar el modelo de datos en bases de datos, lo que permite almacenar, recuperar y manipular la información (Domingo et al., 2008).

Se muestra a continuación una representación de la estructura que presenta la base de datos del sistema, mediante el uso de este modelo de datos. Este se compone por 7 entidades, tres de estas (auth_user, auth_permission y auth_group) constituyen entidades propias generadas por el framework Django y se encuentran relacionadas entre ellas. A su vez la entidad (auth_user) se relaciona con la entidad (sgp_filescripts) quien se encuentra relacionada de uno a uno con (sgp_policy). Esta última tiene una relación de mucho a mucho con (sgp_event) quien presenta una relación de uno a muchos con la entidad (sgp_enenttype).

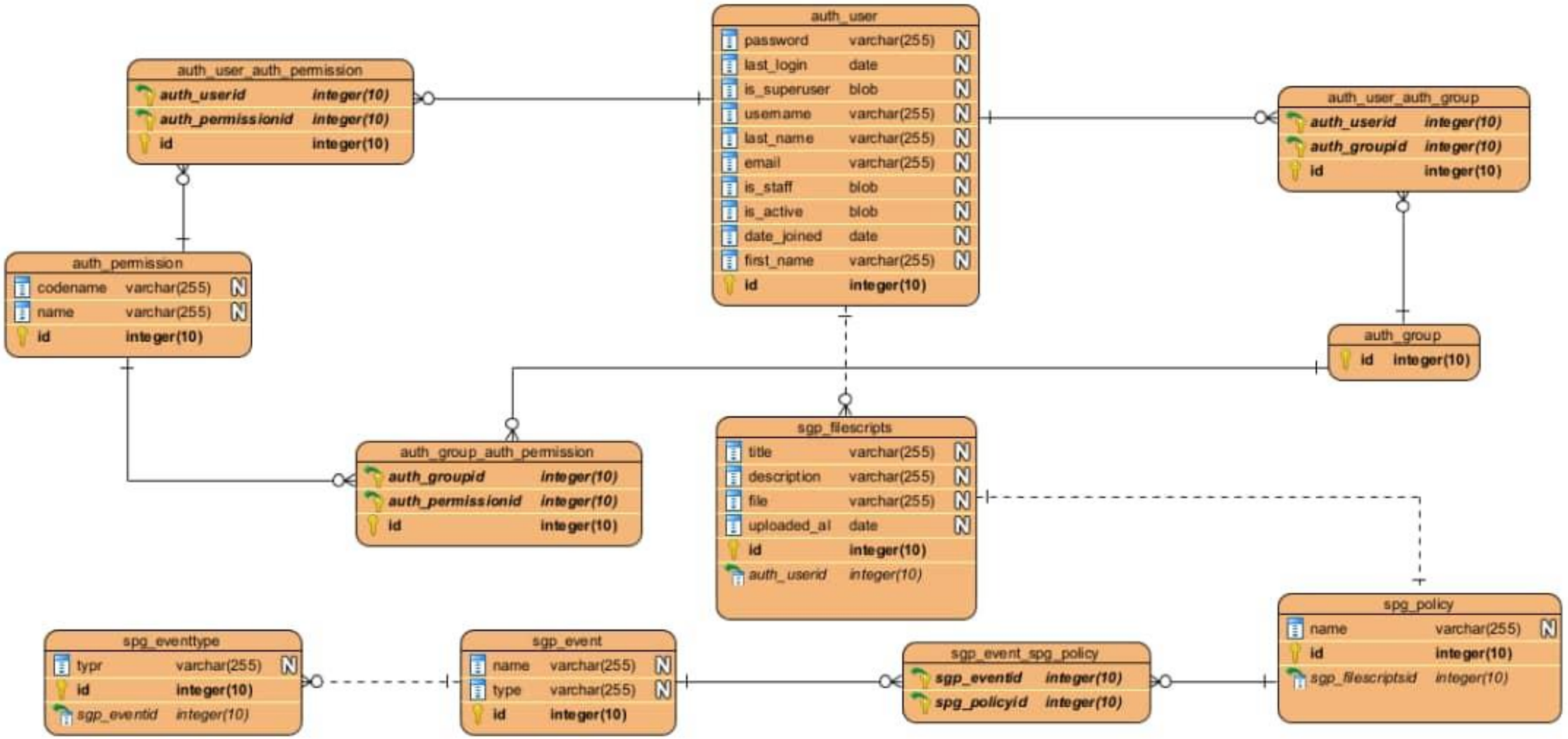


Figura 6: Modelo de Datos.

2.10. Propuesta de despliegue de la solución

Un diagrama de despliegue es una herramienta útil para representar la configuración en tiempo de ejecución de los elementos del proceso y los componentes de software, artefactos y procesos que contienen. Estos se componen de dos elementos principales: nodos y rutas de comunicación. Los nodos representan el entorno físico o virtual en el que operan los componentes, artefactos y procesos de software. Estos nodos pueden ser una variedad de objetos, como servidores físicos, máquinas virtuales, dispositivos de red o incluso dispositivos móviles. Las rutas de comunicación son importantes para comprender cómo fluye la información entre los diferentes componentes de un sistema distribuido. Estas rutas pueden representar conexiones físicas, como cables de red o conexiones inalámbricas, o conexiones lógicas, como protocolos de red o servicios web. El objetivo principal de los diagramas de despliegue es permitir a los desarrolladores, arquitectos de software y otros miembros del equipo comprender mejor la estructura física del sistema y cómo están interconectados los distintos componentes (López, 2012).

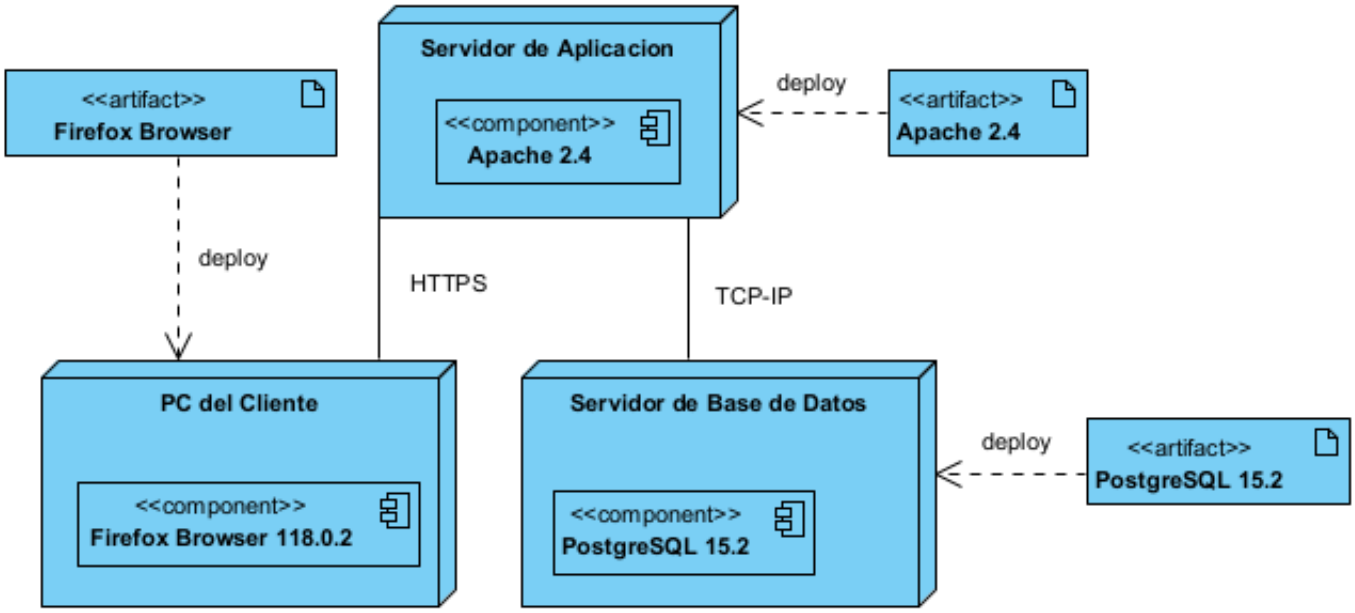


Figura 7: Diagrama de Despliegue.

Conclusiones del capítulo

En el presente capítulo quedó descrito la propuesta de solución, para la cual se realizaron los artefactos definidos en la fase de diseño de la metodología XP, los cuales permiten una caracterización de la propuesta con el fin de garantizar una correcta documentación para la posterior referencia y continuidad

de desarrollo, así como sentar las bases metodológicas para la siguiente fase de desarrollo (implementación). Fueron definidos cada uno de los requerimientos demandados por el usuario tanto estética como funcionalmente. Se redactaron las historias de usuario correspondiente a las funcionalidades, en base a estas se estimaron los tiempos de duración para la implementación de cada una de ellas. Una vez terminada la estimación de tiempo fue confeccionado el plan de iteraciones y se fijaron las fechas de entrega luego de cada iteración. Además, se identificaron los patrones de diseño y arquitectónico y se plasmaron tanto el modelo de datos como el diagrama de despliegue.

CAPÍTULO III: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

Introducción

En el presente capítulo se abordan cada uno los componentes que integran la etapa de implementación y prueba del sistema a desarrollar. Se exponen los diseños de los casos de pruebas, así como los resultados obtenidos del proceso de verificación de la calidad.

3.1. Pruebas de software

El proceso de pruebas de software son un conjunto de actividades planificadas y sistemáticas que tiene los siguientes objetivos (Sommerville, 2011):

- ✓ Demostrar al desarrollador y al cliente que el software satisface sus requisitos. Esto significa que debería haber al menos una prueba para cada requerimiento o característica que se incorporará a la entrega del producto.
- ✓ Descubrir defectos en el software en el que el comportamiento de este es incorrecto, no deseable o no cumple su especificación. La prueba de defectos está relacionada con la eliminación de todos los tipos de comportamientos del sistema no deseables, tales como caídas del sistema, interacciones no permitidas con otros sistemas, cálculos incorrectos y corrupción de dato.

3.1.1. Pruebas funcionales

Las pruebas funcionales se centran en identificar inconsistencias, asegurar funcionalidades y garantizar que los sistemas cumplan sus funciones específicas. Estas pruebas, buscan validar entradas y salidas sin adentrarse en cómo se generan las respuestas del sistema. Su objetivo es reducir costos por no conformidades, evitar reprocesos, mejorar la productividad y aumentar la satisfacción del cliente. Estas pruebas suelen ser la etapa final antes del paso a producción. Las pruebas funcionales se consideran una subcategoría de las pruebas de caja negra, ya que ambas se centran en la funcionalidad del software sin requerir conocimiento interno de su estructura (Sanchez Peño, 2015).

3.1.1.1. Diseño de Casos de Pruebas

Los Casos de Pruebas han sido realizados sobre la base de las funcionalidades del sistema y tienen como objetivo fundamental encontrar la mayor cantidad posible de deficiencias existentes en las funcionalidades implementadas.

Tabla 14: Caso de prueba. Gestionar eventos.

Caso de Prueba		
Código: CP_02	**Nombre de la funcionalidad :** Gestionar eventos	
Descripción: Permitir la crear, modificar, mostrar y eliminar eventos en el sistema.		
Escenario	**Descripción**	**Respuesta**
EC 1. Opción de Crear Evento.	Selecciona la opción de crear evento la cual mostrará una ventana, con los campos nombre y tipo, para crear un evento.	Brinda la posibilidad de crear un evento con los datos, nombre y tipo.
EC 2. Se introduce campos vacíos.	El sistema deberá alertar sobre la obligación de los campos.	El sistema muestra un borde rojo sobre el campo omitido y una alerta con el mensaje: "*Este campo es obligatorio.*"
EC 3. Se introducen letras y números en el campo "nombre".	El sistema deberá permitir la creación del evento con estos datos.	El sistema crea el evento correctamente.
EC 4. Cancelar la operación "Crear Evento".	El sistema debe cerrar la interfaz de crear eventos.	El sistema cierra la ventana de crear eventos correctamente.
EC 5. Se introducen caracteres extraños (#*/+) en el campo nombre "nombre".	El sistema debe lanzar un mensaje alertando sobre la presencia caracteres inválidos.	El sistema muestra en rojo una alerta con el mensaje: "*Caracteres no válidos*", y no permite crear el evento.
EC 6. Opción de Modificar Evento.	Selecciona la opción de modificar evento la cual	El sistema muestra una ventana que carga los datos del evento

	mostrará una ventana con los datos del evento a modificar.	seleccionado y brinda la posibilidad de modificarlos.
EC 7. Modificar evento dejando campos vacíos.	No se debe modificar un evento dejando campos vacíos.	El sistema no modifica el evento si los campos no están llenos.
EC 8. Modificar evento insertando caracteres extraños (#*/+).	No se debe modificar un evento si los campos tienen caracteres inválidos.	El sistema no modifica el evento si los datos introducidos poseen caracteres extraños.
EC 9. Mostrar eventos en la tabla.	El sistema debe mostrará los eventos en una tabla, automáticamente, después de ser creados.	Cuando se crea un nuevo evento este se muestra automáticamente en la tabla.
EC 10. Mostrar opciones de eliminar y editar por cada evento en la tabla.	Mostrará en la tabla de eventos las opciones de editar y eliminar al lado de cada evento.	Cuando se muestran los eventos tienen al lado las opciones de editar y eliminar.
EC 11. Eliminar evento.	Selecciona la opción eliminar y muestra un mensaje para confirmar la eliminación del evento.	Cuando se selecciona la opción de eliminar se muestra un mensaje "*Esta seguro que desea eliminar*" donde en caso de confirmar, se elimina el evento seleccionado.
EC 12. Cancelar opción de eliminar.	Selecciona la opción eliminar y muestra un mensaje con la posibilidad de eliminar o cancelar la operación.	Cuando se selecciona la opción de eliminar se muestra un mensaje "*Esta seguro que desea eliminar*" donde en caso de cancelar, se cierra la ventana y no se elimina el evento.

3.1.2. Pruebas Unitarias

Las pruebas unitarias se centran en examinar la lógica interna del código, lo que la sitúa dentro del enfoque de caja blanca al requerir conocimiento detallado de la estructura interna del software (Sanchez Peño, 2015). Para la realización de las pruebas unitarias se utilizó TestCase que es una subclase de Unittest, paquete nativo de Python para realizar Pruebas Unitarias, permite crear clases y dentro de ellas funciones las cuales representarán las pruebas. TestCase implementa un método llamado setUp () el cual permite definir las instancias iniciales o estructuras iniciales que se utilizarán a lo largo de todas las pruebas. A continuación, se muestran imágenes de las pruebas unitarias realizadas durante el proceso de implementación:

```
class EventTestCase(TestCase):
    nayeli lopez-castro morejon
    def setUp(self):
        self.client = Client()
        self.event = Event.objects.create(type='SomeType', name='SomeName')

    nayeli lopez-castro morejon
    def test_get_request(self):
        response = self.client.get(reverse('events_create'), follow=True)
        self.assertEqual(response.status_code, 200)

    nayeli lopez-castro morejon
    def test_ajax_post_valid_form(self):
        data = {'type': 'SomeType', 'name': 'SomeName'}
        response = self.client.post(reverse('events_create'), data, HTTP_X_REQUESTED_WITH='XMLHttpRequest', follow=True)
        self.assertEqual(response.status_code, 200)
        self.assertTrue(Event.objects.filter(type='SomeType', name='SomeName').exists())
```

Figura 8: Pruebas Unitarias realizadas al sistema I.

```python
    def test_invalid_form(self):
        invalid_data = {
            'type': '',
            'name': 'ValidName'
        }
        form = EventForm(data=invalid_data)
        self.assertFalse(form.is_valid())

    nayeli lopez-castro morejon
    def test_non_ajax_request_returns_template(self):
        user = User.objects.create_user(username='testuser', password='password123')
        self.client.force_login(user)

        response = self.client.post(reverse('events_create'))
        self.assertEqual(response.status_code, 200)

        self.assertTemplateUsed(response, 'events.html')
```

Figura 9: Pruebas Unitarias realizadas al sistema II.

3.2. Resultados de las Pruebas

Con la realización de las pruebas unitarias se evidencia la aceptación de todas las verificaciones:

```
Creating test database for alias 'default'...
Found 24 test(s).
System check identified no issues (0 silenced).
........................
----------------------------------------------------------------------
Ran 24 tests in 0.408s

OK
Destroying test database for alias 'default'...
```

Figura 10 Aceptación de las pruebas Unitarias.

Luego de aplicado cada uno de los casos de prueba fueron detectadas las siguientes no conformidades:

Tabla 15: No conformidades detectadas.

No.	Funcionalidad	Descripción	Complejidad	Estado
1	F 2	Se guardan los eventos creados con campos vacíos.	Baja.	Resuelta.
2	F 2	Se guardan los eventos creados con caracteres extraños.	Baja.	Resuelta.
3	F 2	No se muestran los eventos en la tabla.	Alta.	Resuelta.
4	F 2	Elimina inmediatamente sin mostrar el mensaje para confirmar o cancelar la eliminación de un evento.	Media.	Resuelta.
5	F 2	No muestra ningún tipo de evento para ser seleccionado.	Media.	Resuelta.
6	F 3	No se muestra el mensaje de campos obligatorios cuando se intenta agregar un archivo con el nombre vacío.	Baja.	Resuelta.
7	F 3	Al seleccionar la opción de eliminar archivos el sistema no responde a la petición.	Alta.	Resuelta.

8	F 3	Al seleccionar la opción de modificar archivo no se muestra la ruta del archivo.	Alta.	Resuelta.
9	F 3	Al seleccionar la opción de modificar, si se dejan vacíos campos obligatorios, permite su modificación.	Baja.	Resuelta.
10	F 4	Al seleccionar la opción de editar políticas no muestra los datos de la política seleccionada.	Alta.	Resuelta.
11	F 4	Cuando se intenta seleccionar un archivo para crear una política no muestra la tabla con los archivos existentes.	Alta.	Resuelta.
12	F 4	No se puede descargar el archivo de una política.	Alta.	Resuelta.
13	F 4	Permite seleccionar más de 1 archivos para crear una política.	Alta.	Resuelta.
14	F 4	Al listar una política no se muestran en la tabla las opciones de eliminar, editar y descargar la política.	Media.	Resuelta.

15	F 5	Si se deja la barra de búsqueda de políticas en blanco no se muestra ningún mensaje que pida insertar datos.	Baja.	Resuelta.
16	F 5	Al buscar un evento no se muestran las opciones de editar y eliminar.	Baja.	Resuelta.
17	F 5	No permite buscar eventos por tipo.	Baja.	Resuelta.
18	F 5	Al intentar buscar una política no se muestra ningún elemento.	Alta.	Resuelta.

En la siguiente gráfica se muestran los datos correspondientes a cada iteración de prueba por las que transitó el sistema:

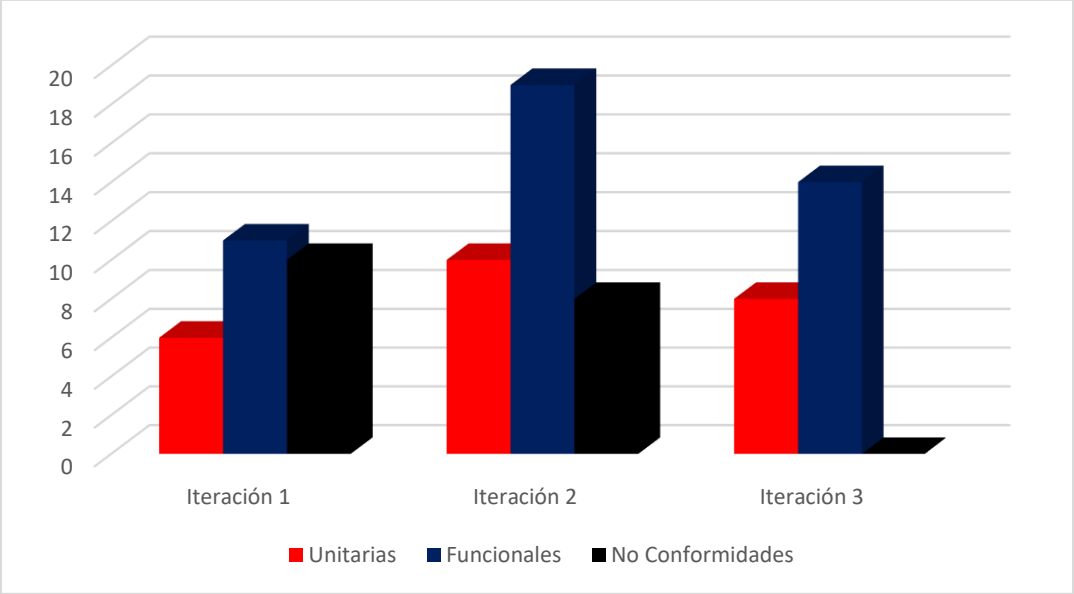


Figura 11: Grafico de no conformidades por iteración.

Conclusiones del capítulo

En el presente capítulo se llevaron a cabo las pruebas para validar la propuesta de solución. Se Aplicaron pruebas de Funcionalidad a cada uno de los requerimientos propuestos por el cliente para comprobar su correcto funcionamiento. Una vez realizados todos los casos de prueba, los errores fueron plasmados en una tabla de no conformidades y se les fue dando solución a cada uno. También se realizaron pruebas Unitarias al código las cuales arrojaron resultados satisfactorios durante su ejecución. Una vez aplicadas todas las pruebas y visto que no se producían errores, se comprobó el correcto funcionamiento del sistema.

CONCLUSIONES FINALES

Como resultado del proceso de Investigación-Desarrollo realizado se arribó a las siguientes conclusiones generales:

- El análisis de cada uno de los referentes teóricos y metodológicos relacionados con sistemas informáticos para la gestión de las políticas de TI, permitió alcanzar un mayor entendimiento del funcionamiento de estos sistemas en el ámbito internacional. Además, favoreció a la elección adecuada tanto de la metodología de desarrollo de software como de los lenguajes, herramientas y tecnologías a emplear durante el proyecto.
- El análisis y diseño de la propuesta de solución, garantizaron que se desarrollara acorde a los requerimientos identificados.
- Los artefactos generados por la metodología seleccionada, la correcta planificación para la entrega de los resultados y el uso de los patrones de diseño adecuados garantizaron que se realizara un proceso de codificación eficaz, permitiendo el producto final cuente con la calidad adecuada.
- La eficacia de las funcionalidades implementadas, está avalada por los resultados de las pruebas realizadas, que demostraron el correcto funcionamiento de cada una de ellas.

Una vez planteado esto se puede concluir que fueron cumplidos de forma satisfactoria cada uno de los objetivos propuestos en la investigación.

RECOMENDACIONES

A partir del estudio realizado y luego de haber analizado los resultados obtenidos se recomiendan los siguientes elementos a tener en cuenta para futuros trabajos:

- Incorporar un servicio REST para la integración de otras aplicaciones.
- La integración de Inteligencia Artificial para automatizar la generación de políticas.

REFERENCIAS BIBLIOGRÁFICAS

Aleixandre Sanz, J. (2022). *La transformación digital en la empresa española: El caso de SilverStorm.* Univerdidad de Valladolid.

Alvarez, O. D. G., Larrea, N. P. L., & Valencia, M. V. R. (2022). Análisis comparativo de Patrones de Diseño de Software. *Polo del Conocimiento: Revista científico-profesional*, *7*(7), 2146-2165.

Aryal, S. (2019). *Bootstrap: A front-end framework for responsive web design*. http://www.theseus.fi/handle/10024/161309

Blancarte, O. (2020). Introducción a la Arquitectura de Software: Un enfoque práctico. *Retrieved*, *11*(29), 2020.

Bustos Tejo, A. (2022). *Desarrollo de una aplicación web mapping para la difusión de los monumentos arqueológicos de la Provincia de Choapa.* [Universidad de Concepción Facultad de Arquitectura, Urbanismo y Geografía Departamento de Geografía]. http://repositorio.udec.cl/handle/11594/10292

Casanova, M. P., & Calderón, C. A. (2020). Modelo para la gestión de infraestructuras de tecnologías de la información. *TecnoLógicas*, *23*(48), 32-54.

Castaño, L. E. G., Soto, S. V. M., González, G. V. M., & Betancur, R. A. M. (2021). Coffee Challenge: Un juego para la enseñanza de patrones de diseño de software. *Revista Politécnica*, *17*(33), 36-46.

Chavez, A., & Luis, J. (2019). *Diseño de arquitectura para herramientas de gestión de clientes basada en políticas*. Universidad de las Ciencias Informáticas. Facultad 2.

Chávez, J. D. (2019). Fundamentos de Programación en Lenguaje C. *IEASS, Editores*.

Chávez, J. D. (2020). *Cliente PSQL de PostGresql*. EASS, Editores.

Code, V. S. (2019). Visual studio code. *línea]. Available: https://code. visualstudio. com*. https://code. visualstudio. com

De Arma-Hernández, A., & Sablón-Fernández, L. E. (2019). Aplicación web para la gestión de la información especializada en Geociencia. *Ciencia & Futuro*, *9*(2), 106-127.

Domingo, C., Ramírez, V., Besembel, I., Espinoza, M. A., & Espinoza, M. V. (2008). Modelo de datos del sistema de relaciones inter-empresariales: RIE. *Ciencia e Ingeniería*, *29*(1), 41-46.

Duque Méndez, N. D. (2002). Diseño e implementación de una política de seguridad. *Universidad Nacional de Colombia - Sede Manizales. Departamento de Informática y Computación*.

Fernández Lobán, M. (2023). *Gesticon* [Universitat Oberta de Catalunya (UOC)]. http://hdl.handle.net/10609/147263

Lara, C., & Figueroa, L. M. (2020). Metodología ágil para el desarrollo de aplicaciones móviles educativas. *XV Congreso Nacional de Tecnología en Educación y Educación en Tecnología (TE&ET 2020)(Neuquén, 6 y 7 de julio de 2020)*. http://sedici.unlp.edu.ar/handle/10915/103770

Larman, C., & Valle, B. M. (2003). *UML y patrones: Una introducción al análisis y diseño orientado a objetos y al proceso unificado* (Segunda Edición). Pearson Educación. https://books.google.com.cu/books?id=3NEaPwAACAAJ

López, F. M. S. (2012). Diseño de un módulo de carga de pagos en entidades públicas mediante mensajería con spring framework. *Industrial data*, *15*(2), 73-79.

Luna, F. (2019). *JavaScript-Aprende a programar en el lenguaje de la web* (Primera Edición). Six Ediciones.

Marcos, G. P., & Mobrici, F. L. (2020). *Integración del entorno Bonitasoft con la herramienta de Gestión de Incidencias Jira*. Universidad Nacional de La Plata.

Menzinsky, A., López, G., Palacio, J., Sobrino, M., Álvarez, R., & Rivas, V. (2018). Historias de usuario. *Ingeniería de requisitos ágil*.

Montoya, C. E. G., Uribe, C. A. C., & Rodríguez, L. E. S. (2013). Seguridad en la configuración del servidor web Apache. *Inge Cuc*, *9*(2), 31-38.

Olsson, M. (2019). *CSS3 Quick Syntax Reference* (2.ª ed.). Apress Berkeley, CA.

Ortega, G. A. V. (2021). Lineamientos para el diseño de aplicaciones web soportados en patrones GRASP. *Ciencia e Ingenieria*, *8*(2), e5716304-e5716304.

Peña Casanova, M., & Anías Calderón, C. (2019). Sistema para ejecutar políticas sobre infraestructuras de Tecnologías de la Información. *Ingeniare. Revista chilena de ingeniería*, *27*(3), 479-494.

Pressman, R. S. (2010). *Ingeniería del Software.“Un Enfoque Práctico”* (Séptima Edición). McGraw-Hill.

Reyes, A., & Barba, A. (s. f.). *Lenguajes de libre distribución para la gestión de redes basada en políticas*. Recuperado 8 de octubre de 2023, de https://www.academia.edu/download/45361880/Lenguajes_de_libre_distribucin_para_la_g20160504-77637-1wp54ce.pdf

Sanchez Peño, J. M. (2015). *Pruebas de software. Fundamentos y técnicas* [UNIVERSIDAD POLITÉCNICA DE MADRID]. https://oa.upm.es/id/eprint/40012

Sharma, A., Khan, F., Sharma, D., Gupta, S., & Student, F. Y. (2020). Python: The programming language of future. *Int. J. Innovative Res. Technol*, *6*(2), 115-118.

Sommerville, I. (2011). *Ingeniería de software* (Novena Edición). Pearson Education, Inc.

Tabarés, R. (2021). HTML5 and the evolution of HTML; tracing the origins of digital platforms. *Technology in Society*, *65*, 101529.

Vidal-Silva, C. L., Sánchez-Ortiz, A., Serrano, J., & Rubio, J. M. (2021). Experiencia académica en desarrollo rápido de sistemas de información web con Python y Django. *Formación universitaria*, *14*(5), 85-94.

Yepez Rodríguez, L. V., & Díaz Machuca, J. E. (2011). *Diseño e implementación de políticas en gestión de redes para la Universidad Tecnológica de Bolívar*.

ANEXOS

Anexos #1: Historias de usuario

Tabla 16: Historia de usuario: Autenticar usuario.

Historia de Usuario	
Número: 1	Nombre: Autenticar usuario.
Usuario: Administrador.	
Prioridad en negocio: Alta (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 1 puntos estimados.	Iteración asignada: 1ra iteración.
Programador responsable: Nayeli López-Castro Morejón.	
Descripción: 1- Objetivo: El módulo debe ser capaz de permitir autenticar al usuario en el sistema. 2- Flujo de la acción a realizar: - Cuando el usuario introduzca correctamente sus datos de inicio de sesión y seleccione el botón de “Acceder” podrá entrar al sistema.	
Prototipo de Interfaz:	

Tabla 17: Historia de usuario: Añadir archivo.

Historia de Usuario	
Número: 6	Nombre: Añadir archivo.
Usuario: Administrador.	
Prioridad en negocio: Alta (Alta/Media/Baja)	Riesgo en desarrollo: Alta (Alto/Medio/Bajo)
Puntos estimados: 3 puntos estimados.	Iteración asignada: 2da iteración.
Programador responsable: Osmani Suárez Fernández.	
Descripción: 1- Objetivo: El módulo debe ser capaz de añadir un nuevo archivo al sistema. 2- Flujo de la acción a realizar: - Cuando el usuario selecciona el botón “Subir Archivo” se muestra una ventana para que introduzca los datos necesarios. - Si se selecciona el botón “Añadir” y todos los datos son correctos se guarda el archivo con los datos introducidos y se cierra la ventana. - Si se selecciona el botón “Cancelar” se cierra la ventana	
Prototipo de Interfaz:	

Tabla 18: Historia de usuario: Modificar archivo del sistema.

Historia de Usuario	
Número: 7	Nombre: Modificar archivo del sistema.
Usuario: Administrador.	
Prioridad en negocio: Media (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 2 puntos estimados.	Iteración asignada: 2da iteración.
Programador responsable: Osmani Suárez Fernández.	
Descripción: 1- Objetivo: El módulo debe ser capaz de permitir modificar un archivo seleccionado del sistema. 2- Flujo de la acción a realizar: - Cuando el usuario selecciona el botón de editar se muestra una ventana con los datos del archivo seleccionado y pueden ser modificados. - Si se selecciona el botón de “Actualizar” y todos los datos son correctos se modifican los datos y se cierra la ventana. - Si se selecciona el botón “Cancelar” se cierra la ventana.	
Prototipo de Interfaz:	

Tabla 19: Historia de usuario: Mostrar archivo del sistema.

Historia de Usuario	
Número: 8	Nombre: Mostrar archivo del sistema.
Usuario: Administrador.	
Prioridad en negocio: Media (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 1 punto estimado.	Iteración asignada: 2da iteración.
Programador responsable: Osmani Suárez Fernández.	
Descripción: 1- Objetivo: El módulo debe ser capaz de listar los archivos existentes en el sistema mostrando nombre y descripción. 2- Flujo de la acción a realizar: - Cuando el usuario agrega un nuevo elemento en el repositorio de archivos este se lista, automáticamente, en la tabla que contiene el listado de archivo. - Si existen más de 10 elementos se muestra la paginación y se listan en otra página. - Si se selecciona el botón siguiente se muestran los eventos de la página que sigue. - Si se selecciona el botón anterior se muestran los eventos de la página anterior.	
Prototipo de Interfaz:	

Tabla 20: Historia de usuario: Eliminar archivo del sistema.

Historia de Usuario	
Número: 9	Nombre: Eliminar archivo del sistema.
Usuario: Administrador.	
Prioridad en negocio: Media (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 1 punto estimado.	Iteración asignada: 2da iteración.
Programador responsable: Osmani Suárez Fernández.	
Descripción: 1- Objetivo: El módulo debe ser capaz de eliminar un archivo seleccionado del sistema. 2- Flujo de la acción a realizar: - Cuando el usuario selecciona la opción de eliminar se muestra un mensaje para confirmar que desee eliminar. - Si se selecciona el botón “Eliminar” se elimina el archivo tanto en la lista como en la base de datos. - Si se selecciona el botón “Cancelar” se cierra la ventana y no se elimina el archivo.	
Prototipo de Interfaz:	

Tabla 21: Historia de usuario: Añadir políticas de TI al sistema.

Historia de Usuario	
Número: 10	Nombre: Añadir políticas de TI al sistema.
Usuario: Administrador.	
Prioridad en negocio: Alta (Alta/Media/Baja)	Riesgo en desarrollo: Alta (Alto/Medio/Bajo)
Puntos estimados: 3 puntos estimados.	Iteración asignada: 3ra iteración.
Programador responsable: Nayeli López-Castro Morejón.	
Descripción: 1- Objetivo: El módulo debe ser capaz de añadir políticas de TI al sistema. 2- Flujo de la acción a realizar: - Cuando el usuario selecciona el botón “Crear Política” se muestra una ventana donde puede seleccionar los eventos y el archivo que va a asociar a la política. - Si se selecciona el botón “Adicionar Evento” Se muestran todos los eventos del sistema para seleccionar los necesarios. - Si se selecciona el botón “Archivos” Se muestran todos los archivos del sistema para seleccionar el necesarios. - Si se selecciona el botón “Crear” y todos los datos son correctos se crea la política y se cierra la ventana.	
Prototipo de Interfaz:	

Tabla 22: Historia de usuario: Modificar políticas de TI del sistema.

Historia de Usuario	
Número: 11	Nombre: Modificar políticas de TI del sistema.
Usuario: Administrador.	
Prioridad en negocio: Media (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 3 puntos estimados.	Iteración asignada: 3ra iteración.
Programador responsable: Nayeli López-Castro Morejón.	
Descripción: 1- Objetivo: El módulo debe ser capaz de modificar la política de TI seleccionada en el sistema. 2- Flujo de la acción a realizar: - Cuando el usuario selecciona el botón de editar se muestra una ventana con los datos de la política seleccionada y pueden ser modificados. - Si se selecciona el botón “Adicionar Evento” Se muestran todos los eventos del sistema para adicionar los necesarios. - Si se selecciona el botón “Archivos” Se muestran todos los archivos del sistema para cambiar el que ya tenía. - Si se selecciona el botón de “Actualizar” y todos los datos son correctos se modifican los datos y se cierra la ventana.	
Prototipo de Interfaz:	

Tabla 23: Historia de usuario: Mostrar políticas de TI del sistema.

Historia de Usuario	
Número: 12	Nombre: Mostrar políticas de TI del sistema.
Usuario: Administrador.	
Prioridad en negocio: Media (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 1 punto estimado.	Iteración asignada: 3ra iteración.
Programador responsable: Nayeli López-Castro Morejón.	
Descripción: 1- Objetivo: El módulo debe ser capaz de mostrar las políticas de TI existentes en el sistema. 2- Flujo de la acción a realizar: - Cuando el usuario agrega una nueva política esta se lista, automáticamente, en la tabla que contiene el listado de políticas. - Si existen más de 10 elementos se muestra la paginación y se listan en otra página. - Si se selecciona el botón siguiente se muestran los eventos de la página que sigue. - Si se selecciona el botón anterior se muestran los eventos de la página anterior.	
Prototipo de Interfaz:	

Tabla 24: Historia de usuario: Eliminar políticas de TI del sistema.

Historia de Usuario	
Número: 13	Nombre: Eliminar políticas de TI del sistema.
Usuario: Administrador.	
Prioridad en negocio: Media (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 1 punto estimado.	Iteración asignada: 3ra iteración.
Programador responsable: Nayeli López-Castro Morejón.	
Descripción: 1- Objetivo: El módulo debe ser capaz de eliminar una política de TI seleccionada del sistema. 2- Flujo de la acción a realizar: - Cuando el usuario selecciona la opción de eliminar se muestra un mensaje para confirmar que desee eliminar. - Si se selecciona el botón “Eliminar” se elimina la política tanto en la lista como en la base de datos. - Si se selecciona el botón “Cancelar” se cierra la ventana y no se elimina la política.	
Prototipo de Interfaz:	

Tabla 25: Historia de usuario: Buscar evento.

Historia de Usuario	
Número: 14	Nombre: Buscar evento.
Usuario: Administrador.	
Prioridad en negocio: Media (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 1 punto estimado.	Iteración asignada: 3ra iteración.
Programador responsable: Osmani Suárez Fernández.	
Descripción: 1- Objetivo: Debe ser capaz de buscar un evento en el módulo Evento. 2- Flujo de la acción a realizar: - Cuando se introduce el nombre o tipo de evento y se selecciona el botón buscar se muestran los eventos que coinciden.	
Prototipo de Interfaz:	

Tabla 26: Historia de usuario: Buscar archivos.

Historia de Usuario	
Número: 15	Nombre: Buscar archivos.
Usuario: Administrador.	
Prioridad en negocio: Media (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 1 punto estimado.	Iteración asignada: 3ra iteración.
Programador responsable: Osmani Suárez Fernández.	
Descripción: 1- Objetivo: Debe ser capaz de buscar un archivo en el módulo Repositorio. 2- Flujo de la acción a realizar: - Cuando se introduce el nombre o descripción de un archivo y se selecciona el botón buscar se muestran los archivos que coincide.	
Prototipo de Interfaz:	

Tabla 27: Historia de usuario: Buscar políticas de TI.

Historia de Usuario	
Número: 16	Nombre: Buscar políticas de TI.
Usuario: Administrador.	
Prioridad en negocio: Media (Alta/Media/Baja)	Riesgo en desarrollo: Media (Alto/Medio/Bajo)
Puntos estimados: 1 punto estimado.	Iteración asignada: 3ra iteración.
Programador responsable: Osmani Suárez Fernández.	
Descripción: 1- Objetivo: Debe ser capaz de buscar una política de TI En el módulo Repositorio y en Inicio. 2- Flujo de la acción a realizar: - Cuando se introduce el nombre de una política y se selecciona el botón buscar se muestran las políticas que coincide.	
Prototipo de Interfaz:	

Anexos #2: Casos de Prueba:

Tabla 28 Caso de Prueba Autenticar Usuario.

Caso de Prueba		
Código: CP_01	**Nombre de la funcionalidad:** Autenticar Usuario	
Descripción: Permitir a los usuarios autenticarse en el sistema.		
Escenario	**Descripción**	**Respuesta**
EC 1. Introducir datos correctos en los campos	Selecciona la opción de iniciar sesión y permite acceder al sistema.	Al iniciar sesión se puede acceder al sistema.
EC 2. Introducir nombre de usuario mal.	Selecciona la opción de iniciar sesión y no permite acceder al sistema.	Cuando se intenta iniciar sesión muestra en rojo un mensaje "*usuario o contraseña incorrecta*" y no se puede acceder al sistema.
EC 3. Introducir contraseña mal.	Selecciona la opción de iniciar sesión y no permite acceder al sistema.	Cuando se intenta iniciar sesión muestra en rojo un mensaje "*usuario o contraseña incorrecta*" y no se puede acceder al sistema.
EC 4. Dejar campos en blanco	Selecciona la opción de iniciar sesión y no permite acceder al sistema.	Cuando se intenta iniciar sesión muestra en rojo un mensaje "*campo obligatorio*" y no se puede acceder al sistema.

Tabla 29 Caso de Prueba Gestionar Archivos

Caso de Prueba		
Código: CP_03	**Nombre de la funcionalidad:** Gestionar archivos	
Descripción: Permitir la crear, modificar, mostrar y eliminar archivos en el sistema.		
Escenario	**Descripción**	**Respuesta**
EC 1. Opción de Crear Archivo.	Selecciona la opción de crear archivo la cual mostrará una ventana, con los campos necesarios para guardar un archivo con un nombre una descripción.	Brinda la posibilidad de guardar un archivo con un nombre y una descripción.
EC 2. Se introduce campos vacíos.	El sistema deberá alertar sobre la obligación de los campos.	El sistema muestra una alerta en rojo con el mensaje: "*Este campo es obligatorio.*"
EC 3 Se introducen letras y números en los campos.	El sistema deberá permitir guardar el archivo con estos datos.	El sistema guarda el archivo correctamente.
EC 4. Se introducen caracteres extraños (#*/+) en el campo nombre "nombre"	El sistema debe lanzar un mensaje alertando sobre la presencia caracteres inválidos.	El sistema muestra en rojo una alerta con el mensaje: "*Caracteres no válidos*", y no permite guardar el archivo.
EC 5. Se introducen caracteres extraños (#*/+) en el campo nombre "descripción"	El sistema deberá permitir guardar el archivo con estos datos.	El sistema guarda el archivo correctamente.

EC 6. Modificar archivo dejando campos vacíos.	No se debe modificar los datos del archivo si se dejan campos vacíos.	El sistema no modifica si los campos no están llenos.
EC 7. Modificar un archivo insertando caracteres extraños en campos que no los permiten (#*/+).	No se debe modificar si los campos tienen caracteres inválidos.	El sistema no modifica si los datos introducidos poseen caracteres extraños en los campos que no los permiten.
EC 8. Opción de Modificar archivo.	Selecciona la opción de modificar la cual mostrará una ventana con los datos del archivo (nombre, descripción y ruta del archivo) a modificar.	El sistema muestra una ventana que carga el nombre, descripción y ruta del archivo seleccionado y brinda la posibilidad de modificarlos.
EC 9. Mostrar opciones de descargar, eliminar y editar por cada archivo en la tabla.	Mostrará en la tabla de archivos las opciones de descargar, editar y eliminar al lado de cada archivo.	Cuando se muestran el nombre y descripción de los archivos tienen al lado las opciones de descargar editar y eliminar.
EC 10. Eliminar archivo.	Selecciona la opción eliminar y muestra un mensaje para confirmar la eliminación del archivo.	Cuando se selecciona la opción de eliminar se muestra un mensaje "*Esta seguro que desea eliminar*" donde en caso de confirmar, se elimina el archivo seleccionado.

Tabla 30 Caso de Prueba Gestionar Políticas de TI.

Caso de Prueba	
Código: CP_04	**Nombre de la funcionalidad:** Gestionar políticas de TI.
Descripción: Permitir la crear, modificar, mostrar y eliminar políticas en el sistema.	

Escenario	Descripción	Respuesta
EC 1. Opción de Crear Política.	Selecciona la opción de crear política la cual mostrará una ventana para seleccionar los eventos y el archivo relacionados con la política.	Brinda la posibilidad de crear una política con los eventos y archivos almacenados en el sistema.
EC 2. Seleccionar Eventos para la política.	Para crear políticas es necesario seleccionar los eventos del sistema con los que va a contar la política.	Cuando se accede a la opción de “Crear política” y se selecciona el botón “Añadir Eventos” se muestran todos los eventos existentes en el sistema.
EC 3. Seleccionar Archivo para la política.	Para crear políticas es necesario seleccionar un único archivo que va a dar solución a la política.	Cuando se accede a la opción de “Crear política” y se selecciona el botón “Archivos” se muestran todos los archivos existentes en el sistema, para seleccionar solo 1.
EC 4. Opción de Modificar Política.	Selecciona la opción de modificar la cual mostrará una ventana con los datos de la política a modificar.	El sistema muestra una ventana que tiene los datos de la política seleccionada y brinda la posibilidad de modificarlos.
EC 5. Descargar archivo de la política.	Selecciona la opción de descargar y permite descargar el archivo relacionado con la política.	Cuando se selecciona la opción de descargar se descarga el archivo.
EC 6. Mostrar políticas en la tabla.	El sistema debe mostrará las políticas en una tabla,	Cuando se crea una nueva política se muestra

	automáticamente, después de ser creadas.	automáticamente en la tabla de políticas.
EC 7. Mostrar opciones de descargar, eliminar y editar por cada política en la tabla.	Mostrará en la tabla de políticas las opciones de descargar, editar y eliminar al lado de cada política.	Cuando se muestran las políticas tienen al lado las opciones de descargar, editar y eliminar.
EC 8. Eliminar políticas.	Selecciona la opción eliminar y muestra un mensaje con la posibilidad de eliminar o cancelar la operación.	Cuando se selecciona la opción de eliminar se muestra un mensaje "*Esta seguro que desea eliminar*" donde en caso de cancelar, se cierra la ventana y no se elimina la política, y en caso de confirmar se elimina y se cierra la ventana.

Tabla 31 Caso de Prueba Buscar por criterios

Caso de Prueba		
Código: CP_05	**Nombre de la funcionalidad:** Buscar por criterios.	
Descripción: Permitir buscar evento, archivos y políticas según sus atributos.		
Escenario	**Descripción**	**Respuesta**
EC 1. Buscar Evento.	En el módulo de eventos se insertan datos en la barra de búsqueda y se selecciona la opción de buscar.	Brinda la posibilidad de buscar eventos almacenados en el sistema por su nombre o tipo.

EC 2. Barra de buscar eventos vacía.	El sistema debe mostrar un mensaje alertando sobre el campo vacío.	Se muestra un mensaje pidiendo que introduzca nombre o tipo para buscar.
EC 3. Buscar evento que no existe.	Mostrará una alerta de no encontrado.	Se muestra la tabla vacía y un mensaje" El *evento no ha sido encontrado*".
EC 4. Buscar evento por tipo.	Se inserta un tipo de evento en la barra y mostrara todos los eventos que coinciden con ese tipo.	Se muestran todos los eventos con tipo igual al buscado.
EC 5. Buscar evento por nombre.	Se inserta un nombre o fragmento de uno, en la barra de búsqueda, y mostrara todos los eventos que contienen ese nombre.	Se muestran todos los eventos que contienen en el nombre el insertado en la barra de búsqueda.
EC 6. Mostrar Opciones de eliminar y editar en los eventos buscados.	Muestra las opciones de eliminar y editar al lado de los eventos encontrados.	Cuando se busca algún evento se muestran las opciones de eliminar y editar al lado de cada resultado de la búsqueda.
EC 7. Barra de buscar políticas vacías.	El sistema debe mostrar un mensaje alertando sobre el campo vacío.	Se muestra un mensaje pidiendo que introduzca datos para la búsqueda.
EC 8. Mostrar Opciones de descargar, eliminar y editar en los archivos buscados.	Muestra las opciones de descargar, eliminar y editar al lado de los archivos encontrados.	Cuando se busca algún archivo, ya sea por nombre o descripción, se muestran las opciones de eliminar, editar y descargar al lado de cada resultado de la búsqueda.

EC 9. Barra de buscar archivos vacía.	El sistema debe mostrar un mensaje alertando sobre el campo vacío.	Se muestra un mensaje pidiendo que introduzca datos para la búsqueda.
EC 10. Buscar política.	Se inserta un nombre o fragmento de uno, en la barra de búsqueda, y mostrara todas las políticas que contienen ese nombre.	Se muestran Todas las políticas que coinciden con el nombre insertado en la barra de búsqueda.