Memoria del Proyecto

Vesta Risk Manager

T-Code
Agustín Collareda, Cintia Hernandez, Hugo Frey

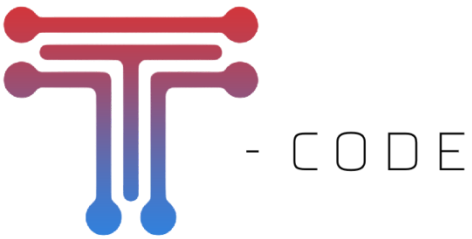

Tabla de contenido

Memoria del Proyecto

Introducción

Contexto del proyecto

El sistema **Vesta Risk Manager** es una sistema web desarrollado como trabajo final de la materia Laboratorio de Desarrollo de Software. Su propósito es servir como una herramienta de apoyo para la gestión de riesgos, además de ser utilizada en asignaturas de las carreras de Analista de Sistemas y Licenciatura en Sistemas, donde los estudiantes llevan a cabo proyectos de software para los cuales se necesita de una planificación rigurosa y estructurada.

Vesta optimiza los procesos de identificación, análisis y monitoreo de riesgos propios de la gestión de riesgos, proporcionando a los usuarios un entorno automatizado para la evaluación de amenazas y su impacto en los proyectos, permitiendo registrar y gestionar información clave a través del uso de formularios.

Además, el sistema incorpora generación de gráficos y resúmenes, que ofrecen una visión clara de los riesgos activos, evaluaciones pendientes, planes de acción, etc. lo que permite al usuario hacer un seguimiento detallado y tomar decisiones para mitigar posibles problemas antes de que afecten significativamente el desarrollo del proyecto.

Alcance del proyecto

Los objetivos del sistema son la reducción de tiempo invertido en la gestión de riesgos de un proyecto junto con los errores y fallas producidos al introducir la interpretación de la información y la trazabilidad de las actividades realizadas para controlar riesgos. **Vesta Risk Manager** fomenta la colaboración con otros estudiantes para sus proyectos de las materias mediante la generación de una solución de Software que permita mejorar la metodología de trabajo de la gestión de los riesgos, como así también generar nuevos vínculos entre profesionales de la misma carrera. Como los estudiantes avanzados de esta carrera conocen el dominio de la gestión de riesgo podría llegar a ser de utilidad para la mayoría de ellos y se podría expandir a diferentes carreras.

Objetivo del Proyecto

Objetivo general

La UNPA UARG no cuenta con un sistema de gestión de riesgos para los diversos proyectos desarrollados por los alumnos de la universidad. Durante el curso, se utilizan las plantillas del PSI, las cuales fueron de gran ayuda pero tuvieron ciertas limitaciones a la hora de gestionar los riesgos y realizar su respectivo seguimiento, un ejemplo de esto es el identificador de los riesgos que se debe hacer de manera manual, la incorporación de las tareas de los planes generados para los riesgos o poder establecer las incidencias para generar mejores análisis retrospectivos en los cierres de iteración. Por ello, T-Code llevará a cabo el desarrollo de un

sistema de gestión de riesgos de uso libre, accesible para cualquier persona interesada en gestionar los riesgos de sus proyectos, ya sean de software o de otra índole. Esto con el fin de optimizar los tiempos asociados en la identificación, análisis y seguimiento de los riesgos junto a la planificación asociada por cada riesgos priorizado.

Además, como parte de la cátedra Laboratorio de Desarrollo de Software, se busca que durante el desarrollo del proyecto los integrantes del equipo adquieran experiencia en la gestión de un proyecto de desarrollo, contemplando la documentación del proyecto, las estimaciones, la gestión de riesgos, la gestión de calidad, entre otros aspectos.

Objetivos específicos

Los objetivos específicos de Vesta Risk Manager son:

- Investigar diferentes técnicas de gestión de riesgos con el fin de generar una solución sencilla y efectiva.
- Implementar una solución que resuelva el problema planteado en la sección de objetivo general.
- Documentar todas las actividades realizadas para garantizar el correcto funcionamiento y poder mejorar su adopción ante terceros.
- Validar el sistema con las presentaciones desarrolladas a lo largo de la cursada para obtener una retroalimentación constante y consistente.

Los objetivos que se tuvieron durante todo el desarrollo de Vesta Risk Manager fueron:

- Experimentar las actividades asociadas a la gestión de proyectos y al proceso de desarrollo de software. Esto se puede apreciar por el modo que se dictó la materia, ya que los docentes de la cátedra hacen mucha énfasis en la importancia de los artefactos generados en la gestión de proyectos.
- Optimizar las actividades referidas a la gestión de riesgos (identificación, análisis y monitoreo) del PSI. En nuestro caso, teníamos una noción básica sobre la gestión de riesgos pero para proponer un software se necesitaría de algunas iteraciones para mejorar el entendimiento del problema
- Trabajar las habilidades blandas como la comunicación efectiva, el liderazgo, el trabajo en equipo, creatividad, etc. Al realizar presentaciones de los documentos realizados, mejoramos mucho nuestra comunicación con personas externas al proyecto. Por otra parte, al tener tiempos muy cortos tuvimos que aprender a gestionar nuestros tiempos para garantizar el cumplimiento de los mismos.

Marco de trabajo

Análisis de metodologías

Metodología PSI

La metodología de desarrollo PSI es un marco de trabajo basado en el Proceso Unificado para el Desarrollo de Software. Se caracteriza por estar dirigido por Casos de Uso, centrado en la arquitectura y por ser iterativo e incremental.

Es una propuesta de mejora a procesos de software pequeños y medianos, siendo principalmente enfocada a alumnos de las carreras Analista de Sistemas y Licenciatura en Sistemas, de igual manera pudiendo ser utilizado en el ámbito profesional donde se apliquen prácticas de ingeniería de software.

Metodología Tongji

La metodología Tongji es un enfoque estructurado para la evaluación y gestión de riesgos en proyectos de software, desarrollada en la Universidad Tongji. Esta metodología combina técnicas cualitativas y cuantitativas para identificar, analizar y responder a los riesgos a lo largo del ciclo de vida del proyecto.

Esta metodología funciona en varias etapas: Primero, se identifican los posibles riesgos que podrían afectar el proyecto, ya sea a nivel técnico, organizativo o de gestión. Luego, cada riesgo se analiza evaluando su **probabilidad de ocurrencia** y su **impacto potencial**. Con esa información, se establece una **prioridad** para enfocar los esfuerzos en los riesgos más críticos. A continuación, se definen estrategias para **prevenir, mitigar o responder** a esos riesgos en caso de que ocurran. Finalmente, se realiza un **seguimiento constante** durante todo el ciclo del proyecto para actualizar la información de riesgos y ajustar las acciones cuando sea necesario.

- La **probabilidad** indica cuán probable es que ocurra el riesgo (1 = muy poco probable, 10 = muy probable).

- El **impacto** representa qué tan grave sería si el riesgo ocurriera (1 = impacto mínimo, 10 = impacto crítico).

Una vez asignados estos valores, se calcula el **nivel de riesgo** multiplicando ambos:

Nivel de riesgo = Probabilidad × Impacto

En base al nivel de riesgo obtenido, se toma una decisión correspondiente: puede ignorarse si el nivel de riesgo es menor a 9; revisarse si el nivel de riesgo está entre 9 y 36; evaluarse para tomar una decisión si el nivel de riesgo está entre 36 y 64; o resolverse de forma inmediata si el nivel de riesgo es mayor a 64.

Enfoque híbrido adaptado

Con el fin de facilitar el uso e implementación del sistema, se optó por adoptar un enfoque híbrido entre la metodología PSI y la metodología Tongji. Este enfoque utiliza las categorías del PSI, mide el impacto y probabilidad, y prioriza los riesgos de acuerdo a la metodología Tongji.

Esto se puede apreciar con mayor claridad en las actividades realizadas. Durante la identificación de riesgos, se propuso una notación específica que facilita la identificación de riesgos, y determinar impacto y probabilidad. La notación sigue la estructura: *"Dada/s <una o más causas>, podría ocurrir <el riesgo>, lo que conduciría a <uno o más efectos>"*. A su vez, cada riesgo tiene asociado un identificador único lo que permite realizar una trazabilidad correcta durante el desarrollo del proyecto.

En el análisis, todos los riesgos pueden ser analizados, a diferencia de la metodología PSI, que realiza una preevaluación. Esto se realiza con la metodología Tongji, la cual resulta más intuitiva al proporcionar una escala del 1 al 10 para medir el impacto y la probabilidad de ocurrencia.

Durante la planificación, los riesgos con mayor factor de riesgo (calculado como el producto entre impacto y probabilidad) se les debe realizar los siguiente tipos de planes: "Minimización", "Mitigación" y "Contingencia". Cada plan tiene asociada una lista de tareas que deben ser puestas en la planificación del proyecto.

Finalmente en el monitoreo, al inicio de cada iteración se deben realizar las actividades mencionadas anteriormente junto con la identificación de incidencias, esta actividad considera los riesgos de la iteración anterior para analizar las consecuencias de los riesgos ocurridos y qué posibles medidas se podrían haberse tomado.

Grupo de desarrollo

Roles y responsabilidades

El equipo de desarrollo de Vesta Risk Manager está compuesto por:

- Agustín Collareda. Este miembro es líder de proyecto, documentador, analista, programador y tester.
- Cintia Hernandez. Este miembro es diseñadora, documentadora, analista, programadora y tester.
- Hugo Frey. Este miembro es el administrador de configuración, documentador, analista y programador.

Esta división de roles fue establecida en la primera iteración del sistema. Sin embargo, en las primeras dos etapas fue muy complicado poder comprender y aplicar correctamente nuestros roles. Esto se debió a la inexperiencia del equipo a la hora de desarrollar un

producto software. Según el análisis retrospectivo de la cátedra, el equipo tuvo un inicio un tanto lento, que fue acelerando a medida que avanzaron las iteraciones.

Desde nuestro punto de vista, se tuvo un arranque lento por la falta de comprensión y visión del producto y de los artefactos a generar en el desarrollo. La aceleración que se tuvo fue el incremento que brinda el proceso de desarrollo iterativo incremental. Esto permite que la comprensión y visión vayan mejorando.

Comunicación del equipo

La comunicación dentro del equipo de desarrollo se gestionó principalmente a través de **WhatsApp**, herramienta que se utilizó para coordinar tareas rápidas, compartir información y mantener actualizados a todos los integrantes sobre el avance del proyecto de forma ágil y constante. Para las actividades que requerían una mayor colaboración, como el desarrollo conjunto de funcionalidades o la toma de decisiones importantes, se realizaron **llamadas grupales por Discord**, lo que facilitó el trabajo colaborativo en tiempo real.

En cuanto a la comunicación con los **profesores (clientes)**, esta se llevó a cabo principalmente por **correo electrónico**, medio que se utilizó para coordinar reuniones, y **UNPAbimodal** para enviar entregas y realizar consultas puntuales. Además, se mantuvieron instancias de contacto **presencial** durante las clases, donde se aprovechó para realizar correcciones, recibir devoluciones y discutir de manera directa posterior a presentaciones con avances del proyecto.

Esta combinación de herramientas y modalidades nos permitió mantener una comunicación fluida tanto dentro del equipo como con los docentes, asegurando una buena organización y seguimiento continuo del desarrollo del sistema.

Tecnologías utilizadas

Hardware

Todos los integrantes del equipo han trabajado utilizando sus computadoras personales. A continuación, se describen brevemente las características principales de cada equipo utilizado durante el desarrollo del proyecto.

Recurso	Cantidad	Descripción	Integrante
Notebook HP	1	Procesador Intel Core i3 2GHz. Memoria RAM 8GB.	Collareda Agustín
Notebook HP	1	Procesador Intel Core I5, Memoria RAM 8GB	Frey Hugo
PC de escritorio	1	Procesador Intel Core i5 3.30GHz. Memoria RAM 16GB.	Hernandez Cintia

Software

Gestión y diseño:

- **GitHub**: Plataforma de alojamiento de repositorios para el control de versiones y colaboración en equipo.
- **Git**: Sistema de control de versiones distribuido utilizado para el seguimiento de los cambios en el código fuente.
- **Toggl track**: Herramienta de gestión del tiempo utilizada para registrar las horas dedicadas a diferentes tareas y proyectos.
- **Trello:** Aplicación de gestión de proyectos basada en tableros, utilizada inicialmente para organizar tareas y flujos de trabajo. Sin embargo, se optó por dejar de usar esta herramienta ya que generaba demoras en la planificación y no aportaba beneficios concretos al desarrollo del proyecto.
- **Draw.io**: Herramienta para crear diagramas de flujo, arquitecturas de sistema, esquemas y más.
- **Canva**: Plataforma de diseño gráfico utilizada principalmente para crear presentaciones, elementos visuales y recursos gráficos.
- **Figma**: Herramienta de diseño de interfaces y prototipado colaborativo en tiempo real.

Otras:

- **PSI**: Metodología de desarrollo elaborada en la UNPA UARG que integra diferentes propuestas de mejora de procesos de software para proyectos pequeños y medianos.
- **Paquete Office**: Conjunto de herramientas ofimáticas (Word, Excel, PowerPoint) usado para documentación, informes y presentaciones.
- **WhatsApp**: Medio de comunicación informal utilizado para coordinar tareas rápidas entre los integrantes del equipo.
- **Discord**: Plataforma de comunicación en tiempo real usada para reuniones, organización del equipo y soporte continuo.

Frameworks y bibliotecas

- **UARGflow**: Sistema de control de usuarios desarrollado y utilizado en la UNPA UARG.
- **React**: Biblioteca de JavaScript empleada para construir interfaces de usuario dinámicas y reutilizables.

Herramientas de desarrollo

- **PHP 8.2.12**: Lenguaje de programación utilizado en el desarrollo del backend del sistema.
- **JavaScript**: Lenguaje de programación usado en el desarrollo frontend para mejorar la interactividad de la interfaz.

- **XAMPP**: Paquete de software que proporciona un entorno de servidor local con Apache, MariaDB, PHP y Perl, utilizado para ejecutar y probar aplicaciones web de forma local durante el desarrollo.
- **phpMyAdmin**: Herramienta web que permite la administración de bases de datos MySQL de forma gráfica, facilitando la creación, edición y gestión de tablas, consultas y usuarios sin necesidad de utilizar línea de comandos.
- **MySQL**: Sistema de gestión de bases de datos relacional usado para almacenar y gestionar los datos del sistema.
- **VS Code**: Editor de código fuente utilizado como entorno de desarrollo principal.

Proceso de Desarrollo

Origen de Vesta Risk Manager

El desarrollo de **Vesta Risk Manager** comenzó durante la cursada de la materia Laboratorio de Desarrollo de Software, donde se pone en práctica los conocimientos adquiridos a lo largo de la carrera con un proyecto final. Al inicio de la materia los docentes organizaron un sorteo para asignar distintos temas de desarrollo a los equipos de trabajo, el cual dió como resultado a nuestro grupo desarrollar un sistema web para la **gestión de riesgos en proyectos de software**.

El nombre **Vesta Risk Manager** se inspira en la diosa romana **Vesta**, símbolo del hogar, el fuego sagrado y la protección. En la antigua Roma, Vesta aseguraba la estabilidad y seguridad del hogar, previniendo el caos y garantizando su continuidad. Nuestro sistema busca **proteger los proyectos de los usuarios**, gestionando eficientemente los riesgos para evitar problemas y mejorar la toma de decisiones.

Fases del proyecto

Inicio

En esta primera etapa del desarrollo, que fue realizada en una única iteración, se establecieron las bases del proyecto comenzando por la formación del equipo de desarrollo tomando el nombre de **T-Code**. Este nombre surge de la cantidad de integrantes del equipo, la letra “T” representa la cantidad de integrantes, tres, mientras que “Code” hace referencia a nuestro rol como programadores.

Se definió un **estándar de documentación** para que todos los documentos referentes al proyectos sean claros y uniformes durante todo el desarrollo.

Posteriormente, se tuvo la **primera entrevista con los clientes** (los profesores), donde se identificó una restricción clave que es el uso de UARGFlow, esto se debió a que condicionó en gran medida al diseño de la aplicación. A partir de esta reunión, se elaboró un **resumen de la**

entrevista en el cual se encuentra la primera versión de la **lista de requerimientos** del sistema, la cual se tomó como base para ser mejorada en las siguientes fases.

Se presentó una **propuesta de proyecto**, que incluyó un **análisis técnico** y una **propuesta técnica**, donde se detallaron los **casos de uso** junto con la primera **estimación**. También se realizó un **estudio de factibilidad**, en el que se justificó la importancia del proyecto y se realizó una justificación del proyecto, explicando su título, el planteamiento del problema, visión, misión y objetivos, además, se evaluaron la **oferta y demanda**, la **organización del proyecto** y los **beneficios esperados**.

Otro documento importante que fue realizado en esta fase fue el **modelo de negocio** en el cual se detalló la importancia del proyecto, sus objetivos principales y los perfiles involucrados. También fueron descritas las primeras herramientas de apoyo que se utilizarían en el desarrollo.

Con el objetivo de obtener una visión más clara de la estructura del sistema, se elaboraron los primeros **flujogramas**, los cuales permitieron comprender los casos de uso de mejor forma y fueron presentados a los clientes para asegurar una interpretación común del funcionamiento del sistema.

Durante esta etapa **se aprendió a realizar el planteamiento completo de un proyecto de software**, definiendo sus fundamentos técnicos, objetivos y alcance de forma estructurada. Además, se adquirió experiencia en la elaboración de un **presupuesto basado en datos reales**, teniendo en cuenta los recursos necesarios, los tiempos estimados y los costos asociados, lo cual nos permitió comprender mejor la viabilidad económica de un proyecto desde una perspectiva profesional. Por otra lado, se logró comprender una parte del dominio del problema y obtener la primera visión del sistema, lo que sentó bases en las siguientes iteraciones.

Elaboración

Durante la etapa de **Elaboración**, se definieron con mayor precisión los requisitos del sistema, se afinaron los aspectos técnicos fundamentales y comenzó el desarrollo de un prototipo funcional. Esta fase fue clave para transformar las ideas y lineamientos generales definidos en la etapa anterior en una estructura concreta y operativa.

El trabajo se organizó en **dos iteraciones**, cada una con objetivos específicos que permitieron avanzar progresivamente en el diseño, construcción y validación del sistema.

En la **primera iteración** se definieron las bases fundamentales del proyecto, entre los documentos más relevantes generados en esta iteración se encuentra el **Plan de Proyecto**, el cual se elaboró con el objetivo de presentar a los clientes una visión estructurada del rumbo que tomaría el proyecto, incluyendo objetivos, alcance, responsables, mecanismos de control y ajuste, cronograma, etc.

Se desarrolló el primer **Plan de Iteración**, el cual sirvió de modelo para las siguientes iteraciones. En este documento se establecieron las tareas a abordar, sus responsables y las estrategias de seguimiento. También se introdujo las primeras **Estimaciones**.

A partir de la **primera entrevista** con los clientes, se organizaron y se realizó la especificación de los requerimientos. Estos fueron redactados siguiendo la sintaxis formal vista en clase, asegurando claridad, coherencia y trazabilidad.

En esta etapa comenzó a aplicarse la gestión de configuraciones y la gestión de riesgos de manera más formal. Además, se realizó el **Plan de Garantía de Calidad** donde se establecieron las métricas y estándares que guiarán la calidad del desarrollo durante todo el proyecto.

En cuanto a los aprendizajes y experiencias del equipo durante esta iteración, se destaca una mejor comprensión de la estimación por Puntos de Casos de Uso, aunque ya se trabajó con esta técnica anteriormente, en esta iteración se logró comprender mejor su aplicación práctica, lo que permitió realizar estimaciones más precisas y confiables;

Se comenzó a integrar herramientas y conceptos aprendidos en la materia **Gestión de Proyectos de Software**, como la planificación iterativa y el control de cronograma; además se aprendió la importancia de establecer métricas desde el inicio, tanto para la planificación como para el monitoreo del avance y la calidad del proyecto.

En esta iteración, se tuvieron ciertas complicaciones referidas a la planificación, ya que no se pudo completar la Gestión de riesgos en el plazo establecido. Esto se debió concretamente al no planificarlo con la anticipación necesaria. Esto ayuda a remarcar la importancia de la planificación adecuada lo que nos hizo tomar una métrica particular para orientar las planificaciones siguientes. Por otra parte, una de las mayores complicaciones fueron las métricas establecidas en el plan de garantía de calidad que se debian recopilar del proyecto, en nuestro caso, pusimos muchas de más que no podíamos gestionar, lo que fue remarcado en una de las presentaciones y fueron corregidas para la siguiente.

La **segunda iteración** se centró en profundizar los aspectos técnicos y avanzar en la validación del sistema a través de modelos y pruebas. Se ha realizado el **Modelo de Casos de Uso** donde se especificaron en detalle todos los casos de uso, definiendo los actores involucrados y describiendo el flujo de secuencia de cada uno. Esto permitió comprender con claridad el funcionamiento del sistema y sirvió como guía directa para la etapa de programación, así como el diseño del **Modelo de Datos**, en él se definieron las entidades principales, sus relaciones y atributos, teniendo en cuenta la normalización y la eficiencia del acceso a la información.

Se desarrolló un **Prototipo Funcional** que permitió visualizar el aspecto y la navegación del sistema. A lo largo de dos reuniones con el cliente, se realizaron ajustes y mejoras, asegurando que la interfaz sea adecuada a sus expectativas y necesidades. También se realizó el primer **Plan de Pruebas**, donde se definieron los tipos de prueba a realizar, los casos de prueba específicos, los criterios de aceptación y los responsables de su ejecución.

En cuanto a los aprendizajes y mejoras del equipo durante esta iteración, se destaca una **mejora notable en la organización interna del equipo** en comparación con la iteración anterior, se optimizó la asignación de tareas, los plazos se respetaron de mejor manera y hubo una comunicación más fluida. En cuanto al desarrollo de **Diagramas de Clases**, a pesar de ya contar con experiencia previa en su elaboración, fue solicitado por la cátedra y al no contar con una versión completa, se complicó mucho la elaboración del diseño y el entendimiento de las entidades.

Construcción

La **etapa de construcción** estuvo compuesta por un total de nueva iteraciones, de las cuales tres se desarrollaron durante la cursada y las seis restantes se llevaron a cabo “post-cursada”.

Iteraciones durante la cursada:

Durante la **primera iteración** se elaboró el modelo arquitectónico del sistema, definiendo los objetivos y restricciones que servirían de guía para el desarrollo. Se desarrollaron las principales vistas arquitectónicas, junto con las clases de diseño, el modelo de componentes y la implementación de la base de datos. Además se priorizaron, especificaron e implementaron los casos de uso en una primera tanda, los casos de uso implementados fueron: **CU01 (Autentificarse)**, **CU02 (Administrar acceso al sistema)**, y **CU03 (Administrar proyectos)**, a pesar que el **CU04 (Añadir riesgo a la lista)** también se encontraba planificado para esta iteración, por falta de tiempo tuvo que ser reprogramado a la siguiente iteración.

En la **segunda iteración** se realizaron correcciones a los casos de uso implementados y se priorizó e inició el diseño de la segunda tanda de casos de uso, de los cuales fueron implementados: **CU04 (Añadir riesgo a la lista)** de la iteración anterior y el **CU07 (Realizar evaluación de riesgo)**, el **CU08 (Añadir plan de acción)** fue iniciado en esta iteración pero no finalizado, por lo que fue reprogramado para la siguiente iteración.

Finalmente para la **tercera iteración** se comenzó por finalizar el **CU08 (Añadir plan de acción)** para seguir con la implementación de la tercera tanda de casos de uso: **CU12 (Realizar análisis de riesgo)**, **CU09 (Modificar plan de acción)** y **CU05 (Modificar lista de riesgos)**, los cuales fueron terminados según lo planificado. Además se inició con la primera versión de los manuales de usuario e instalación.

Antes de finalizar la cursada, el equipo de cátedra realizó una semana para que cada equipo presentará las condiciones para rendir el final, lo que se había desarrollado en ese momento. Esto nos dio un pantallazo de como seria el final y que podríamos mejorar para este.

Durante estas tres iteraciones **aprendimos** la importancia de contar con una arquitectura sólida como base para el desarrollo, ya que nos permitió organizar el sistema de manera clara y escalable. También valoramos el rol fundamental del diseño y las pruebas, ya que una buena planificación previa evitó muchos errores durante la implementación.

Iteraciones post-cursada:

En la **cuarta iteración** nos enfocamos principalmente en la revisión y corrección de toda la documentación generada hasta el momento, con el objetivo de mejorar la calidad, coherencia y consistencia de los documentos relacionados al proyecto. Sin embargo, no se lograron alcanzar todos los objetivos propuestos, ya que algunos documentos quedaron pendientes de revisión para etapas posteriores. Esta instancia nos permitió identificar mejoras necesarias y prepararnos mejor para las siguientes iteraciones.

Durante la **quinta iteración** finalizamos la revisión de la documentación pendiente y elaboramos una lista de tareas de programación para organizar el trabajo restante, de manera que todos los integrantes del equipo puedan participar durante la implementación. Sin embargo, debido a la falta de tiempo por la preparación de finales, no se logró comenzar con el desarrollo del **CU10 (Realizar y solicitar informes)** ni con el **CU06 (Administrar categorías de riesgo)**. Aun así, se avanzó en la planificación técnica para abordar estas funcionalidades en las siguientes iteraciones. Cabe destacar que la lista de tareas de programación era una actividad asociada a la mitigación de un riesgo RK13: Dependencia exclusiva de un solo desarrollador para la implementación. En este momento tuvimos que decidir aplazar la siguiente iteración por dos semanas ya que entrábamos en la época de rendir finales y los miembros del equipo los priorizaron.

Durante la **sexta iteración** se realizaron avances importantes en el desarrollo de las vistas del sistema y se dio inicio a la implementación de los casos de uso **CU10 (Realizar y solicitar informes)** y **CU06 (Administrar categorías de riesgo)**. A pesar del progreso logrado, no se logró finalizar el CU10 pero si se finalizó el CU06, y no se pudieron llevar a cabo las pruebas correspondientes para ambas, esto se debe a que ambos casos de uso fueron iniciados de forma paralela, haciendo que no se pueda dedicar los tiempos planificados inicialmente para esta iteración. Esta etapa permitió sentar las bases para completar e integrar estas partes en las próximas iteraciones.

En la **séptima iteración** se logró finalizar las pruebas del **CU06** que se había finalizado en la iteración anterior. Por otro lado, la finalización del **CU10** quedó pendiente. Esto se debió a que, al momento de planificar su implementación se subestimó su complejidad, resultando más desafiante de lo esperado. Cabe destacar que se tuvieron que reveer las plantillas de informes para generar un ejemplo claro para que el desarrollador lo pueda implementar.

Durante la **octava iteración**, aunque se trabajó activamente en el **CU10**, no se logró finalizarlo ni realizar sus pruebas debido a que surgieron ajustes imprevistos en funcionalidades anteriores que requirieron más tiempo del estimado. Además, la carga académica de otras materias también influyó en la disponibilidad del equipo, lo que limitó el avance esperado. Un detalle a tener en cuenta es que se finalizó una parte del CU pero en la planificación estaba contemplado su resolución completa. Además en esta etapa se readaptan las plantillas para que el programador pueda finalizar este CU.

Finalmente en la **novena iteración** se logró finalizar el **CU10**, completando así una de las funcionalidades más esperadas del sistema. Además, se inició y finalizó el **CU11 (Exportar**

archivos), una tarea que en un principio creíamos que sería más compleja, pero resultó ser más sencilla de lo previsto. También se realizaron las pruebas correspondientes para ambos casos de uso. Gracias a una buena organización y al ritmo sostenido de trabajo, la iteración pudo completarse una semana antes de lo planificado. En esta iteración, se decidió separar las funcionalidades del CU10 para finalizarlas lo que fue exitoso.

A lo largo de las etapas de construcción del proyecto, entendimos que muchas veces las tareas no salen como se esperan: algunas resultaron más complejas de lo imaginado y otras, sorprendentemente simples. Supimos adaptarnos a los imprevistos, corregir el rumbo cuando fue necesario y seguir avanzando.

Cierre

Durante la etapa de cierre se trabajó en la elaboración de los documentos fundamentales que acompañan el producto final del proyecto.

Por un lado, se desarrolló el **Manual de instalación**, que detalla paso a paso cómo configurar y poner en funcionamiento el sistema, incluyendo las instrucciones para levantar el entorno y posibles soluciones ante errores comunes.

Por otro lado, se redactó el **Manual de usuario**, pensado para el público final. En él se explican de forma clara y accesible las funcionalidades principales del sistema, cómo navegarlo y cómo utilizar cada una de sus partes. Se priorizó una redacción amigable y acompañada de ejemplos visuales para facilitar su comprensión.

Además, en esta etapa se llevó a cabo una **reunión con los profesores**, donde se presentó el avance del proyecto y se recibieron sugerencias y correcciones importantes. Estas observaciones fueron incorporadas de manera inmediata, ya que esta instancia también cumple una función clave de ajuste y mejora del producto antes de su entrega final.

En esta etapa no solo concluimos el trabajo realizado, sino que también aprendimos nuevas habilidades clave como grupo. La elaboración del manual de instalación y del manual de usuario nos permitió entender la importancia de comunicar claramente el funcionamiento del sistema a personas externas al equipo. Además, gracias a la reunión con los profesores, nos permitió detectar oportunidades de mejora que enriquecieron el sistema, como la incorporación de distintos estados para los proyectos administrados (activo, inactivo, finalizado y abandonado), ampliando su funcionalidad y realismo. Aparte de generar una sección de ayuda a los usuarios finales para garantizar un correcto funcionamiento.

Gestión del proyecto

Estimaciones

Una de las primeras tareas clave en el desarrollo del proyecto fue estimar tanto el tamaño como el esfuerzo y tiempo requerido. Para realizar las estimaciones del tiempo requerido para completar el proyecto, se hizo uso de una plantilla de excel para estimación de la metodología PSI. En ella, se realiza la estimación por el método de puntos de casos de uso, el cual toma en cuenta la complejidad de los usuarios del sistema, de los CU diseñados, factores ambientales, factores técnicos y factores de productividad.

En el **Gráfico 01**, se muestran las estimaciones realizadas en cada iteración con fecha de finalización estimada del proyecto. Se muestran tres escenarios posibles: el peor caso, el caso más probable y el mejor caso, lo que brinda una visión más completa sobre la planificación temporal y la incertidumbre asociada a cada fase.

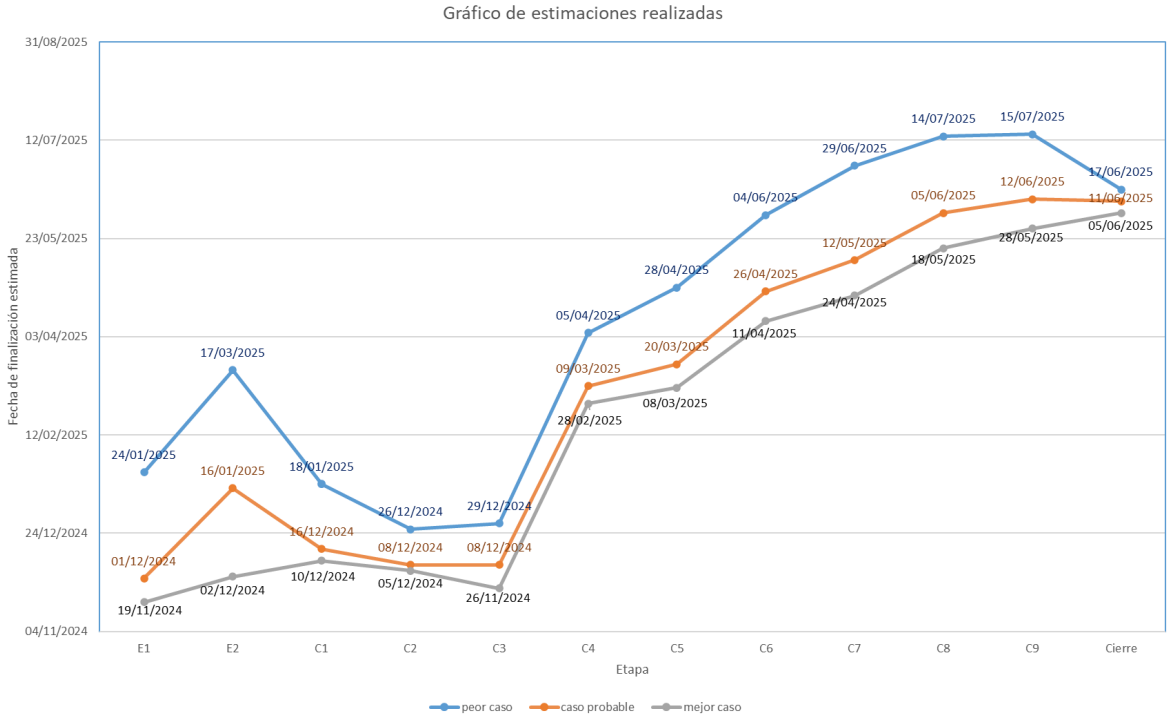

Gráfico 01: Gráfico de estimaciones realizadas con fechas.

En el **Gráfico 02** se presentan las estimaciones realizadas para cada etapa del proyecto, utilizando un formato visual que permite comprender con mayor claridad la evolución de los días restantes hasta la finalización, mostrando también los tres posibles escenarios: peor caso, caso probable y mejor caso.

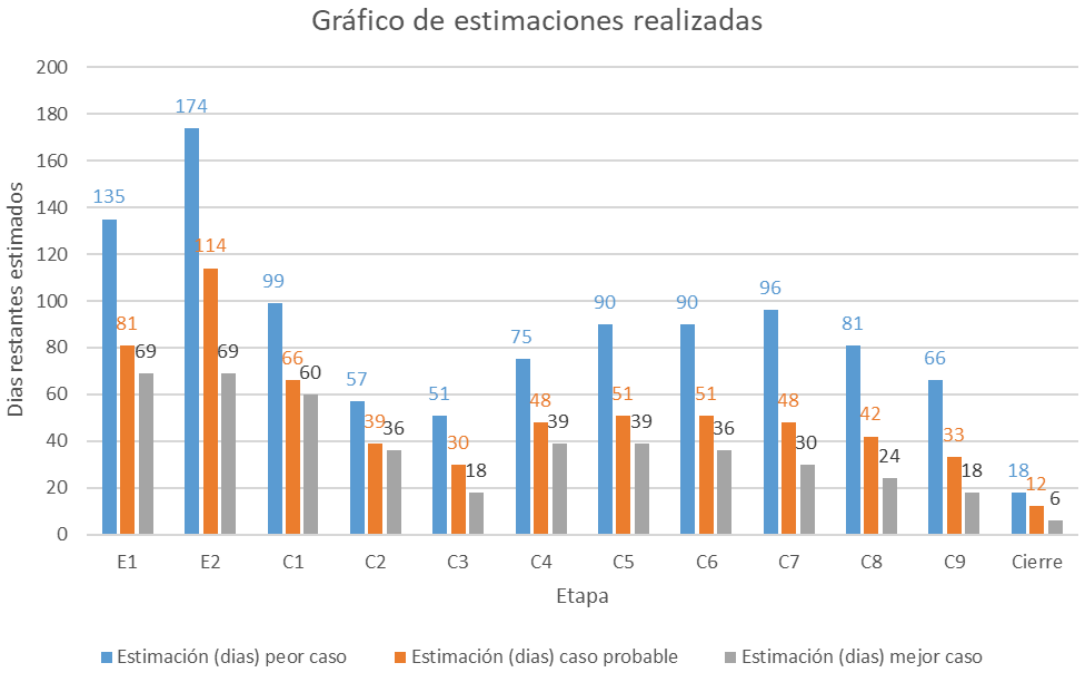

Gráfico 02: Días restantes estimados hasta la finalización del proyecto.

Durante la reunión con los docentes, surgió una observación respecto a las estimaciones realizadas durante la post cursada, **¿por qué las estimaciones seguían aumentando si ya se habían resuelto partes del CU?** Esta pregunta nos llevó a reflexionar y revisar las estimaciones para readaptar los gráficos quedando como se mostraron anteriormente.

Aun así esto fue debido a la limitaciones que poseía las plantillas y a la poca experiencia del grupo para representar correctamente estos cambios en las estimaciones.

Como otra aclaración, las iteraciones C5 y C6 tienen la misma duración, ya que en la C5 no se trabajó con la implementación del sistema. Las variaciones de las demás estimaciones son debido a la diferente disponibilidad y dedicación al proyecto que poseía cada miembro del equipo.

Gestión de riesgos

Categorización de riesgos encontrados

Cronograma:

- RK01: Ausencia de un integrante en ocasiones importantes por enfermedad.
- RK02: Dificultades en estimación de tareas por inexperiencia.
- RK13: Dependencia exclusiva de un solo desarrollador para la implementación.
- RK15: Falta de comunicación por tiempo prolongado.
- RK16: Poca dedicación al proyecto por finales y/o cursada.
- RK18: Gran cantidad de correcciones por reunión con el cliente.

Experiencia y capacidad:

- RK03: Escaso conocimiento del lenguaje de programación (php).
- RK04: Errores humanos en el uso del sistema de control de versiones.
- RK05: Dificultad para encontrar soluciones a problemas técnicos complejos.
- RK06: Diferencias de estilos de codificación entre miembros del equipo.
- RK07: Problemas de compatibilidad entre herramientas y entornos de desarrollo.
- RK08: Falta de experiencia en conducción de un proyecto de software.
- RK09: Dificultades en la implementación de pruebas efectivas y falta de proceso robusto.
- RK10: Problemas con la gestión de permisos y roles en el sistema.
- RK12: Errores no detectados hasta etapas tardías del desarrollo.
- RK17: Desviación del proyecto por suspensión de actividades.

Tecnología:

- RK11: Complejidad de implementación del módulo UARGFlow.
- RK14: Dificultad de implementación de un sistema de notificaciones en tiempo real.

Riesgos por etapa

Elaboración 1:

Durante la etapa de Elaboración 1, estaba previsto iniciar con la planificación de la gestión de riesgos. Sin embargo, debido a la carga de tareas priorizadas en esta iteración (Plan SQA) no se logró destinar el tiempo necesario para abordar correctamente este aspecto. Como resultado, se tomó la decisión consciente de postergar este trabajo para la etapa de Elaboración 2.

Elaboración 2:

En esta etapa se detectaron los primeros 12 riesgos, de los cuales se gestionaron los riesgos RK03, RK11, RK09, RK02 y RK08.

Para minimizar el riesgo **RK03: Escaso conocimiento del lenguaje de programación (php)**, se buscaron cursos de PHP para capacitar a los miembros del equipo y, en caso de tener complicaciones debido a este riesgo, se planificó como contingencia solicitar ayuda a compañeros con más experiencia en el uso de este lenguaje.

Para mitigar el riesgo **RK11: Complejidad de implementación del módulo UARGFlow**, se decidió analizar problemas lógicos y probar la integración con el proyecto, y buscar posibles incompatibilidades de UARGFlow con la versión utilizada de PHP, ya que este framework fue desarrollado con PHP 5 mientras que Vesta Risk Manager utiliza PHP 8. No se descubrieron problemas lógicos ni incompatibilidades.

Para el **RK09: Dificultades en la implementación de pruebas efectivas y falta de proceso robusto**, se decidió realizar un análisis de los casos de prueba diseñados y su efectividad en cada una de las etapas del proyecto con el fin de encontrar posibles optimizaciones, y se elaboró la estructura de un informe de verificación, en el que se registraron todos los defectos detectados durante las pruebas para garantizar que sean corregidos.

Para mitigar el **RK02: Dificultades en estimación de tareas por inexperiencia**, se decidió realizar una retrospectiva de las tareas planificadas y las realizadas, con el fin de obtener la información necesaria para realizar una mejor planificación en iteraciones posteriores.

Para la mitigación del **RK08: Falta de experiencia en conducción de un proyecto de software**, se decidió realizar en cada iteración una revisión de la calidad del producto desarrollado, y se planificó como contingencia una reunión para asignar tareas y realizar revisiones y correcciones de calidad.

Figura 01: Resumen de riesgos de Elaboración 2.

<u>Construcción 1</u>:

Se descubrió que PHP es muy similar a otros lenguajes con los que el equipo de desarrollo estaba más familiarizado, y que se había sobreestimado la dificultad que podía presentar su aprendizaje. Por lo tanto, no se siguió gestionando el riesgo **RK03: Escaso conocimiento del lenguaje de programación (php)**.

Figura 02: Resumen de riesgos de Construcción 1.

Construcción 2:

Se detectó el riesgo **RK13: Dependencia exclusiva de un solo desarrollador para la implementación**. Para este riesgo, se decidió realizar la planificación de las iteraciones de forma tal que los esfuerzos del programador principal se orientaran mayormente en la implementación del sistema, y se delegaron tareas de implementación menos complejas al resto de los miembros del equipo.

Figura 03: Resumen de riesgos de Construcción 2.

Construcción 3:

Gracias a la gestión del riesgo **RK13: Dependencia exclusiva de un solo desarrollador para la implementación**, se logró avanzar satisfactoriamente con la implementación del sistema, ya que los miembros del equipo se encargaban de tareas menos complejas, como la generación de vistas, mientras el programador principal se encargaba de las tareas de programación más difíciles.

Figura 04: Resumen de riesgos de Construcción 3.

Construcción 4:

Para esta etapa se detectaron los riesgos **RK15: Falta de comunicación por tiempo prolongado**, **RK16: Poca dedicación al proyecto por finales y/o cursada** y **RK17: Desviación del proyecto por suspensión de actividades**, todos relacionados al final de la cursada y la necesidad del equipo de desarrollo de continuar con el desarrollo del proyecto de forma independiente, luego de su suspensión durante un tiempo prolongado a causa de finales.

Para el riesgo **RK15**, se planificó claramente, previo al final de la cursada, la fecha de inicio de la siguiente iteración y las tareas a realizar, para el **RK16**, se realizó la planificación de cada iteración teniendo en cuenta las fechas de finales y los horarios reducidos de los miembros del equipo, y para el **RK17**, se revisó y corrigió la documentación para garantizar que esta funcione como guía para los desarrolladores, aun después de la suspensión del proyecto.

Figura 05: Resumen de riesgos de Construcción 4.

Construcción 5:

A causa del **RK16: Poca dedicación al proyecto por finales y/o cursada**, no se pudieron realizar avances significativos durante esta iteración. Sin embargo, algo positivo a destacar es que el **RK17: Desviación del proyecto por suspensión de actividades** no resultó ser un problema gracias a las revisiones realizadas con el objetivo de mitigar este riesgo.

Figura 06: Resumen de riesgos de Construcción 5.

Construcción 6:

El programador principal delegó la implementación de vistas del sistema a los demás miembros del equipo, como parte del plan de mitigación del riesgo **RK13: Dependencia exclusiva de un solo desarrollador para la implementación**, lo que permitió un mayor avance en la implementación del sistema. Sin embargo, no se realizaron las pruebas necesarias a causa del poco tiempo de los miembros del equipo para destinar al proyecto.

Figura 07: Resumen de riesgos de Construcción 6.

Construcción 7:

Se completaron las pruebas que no se realizaron en la iteración anterior, sin embargo, no se realizaron los avances esperados debido al poco tiempo disponible de los miembros del equipo de desarrollo.

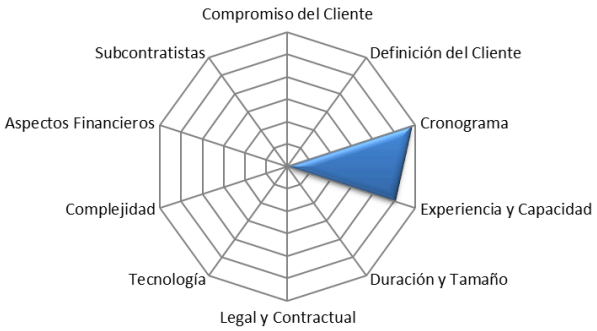

Figura 08: Resumen de riesgos de Construcción 7.

Construcción 8:

El riesgo **RK01: Ausencia de un integrante en ocasiones importantes por enfermedad** tuvo un ligero aumento, ya que efectivamente se presentó una situación de este tipo en una instancia clave del desarrollo.

Por otro lado, el **RK17: Desviación del proyecto por suspensión de actividades** se cerró, debido a que no se presentaron problemas de comprensión e implementación del proyecto, en parte gracias a la revisión y corrección de la documentación realizada en iteraciones previas como medida de prevención contra este riesgo.

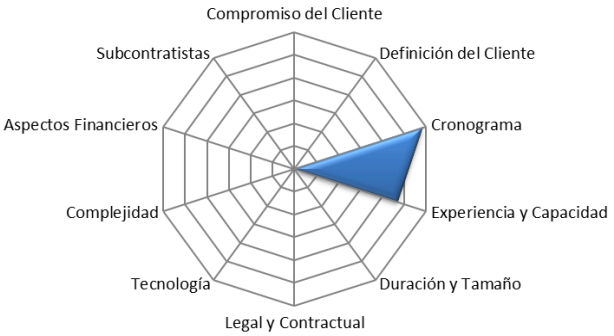

Figura 09: Resumen de riesgos de Construcción 8.

Construcción 9:

Se logró dedicar un tiempo considerable al desarrollo del proyecto, completando todas las tareas planificadas de la forma esperada y terminando la implementación del proyecto. Por esta razón, la mayoría de los riesgos aún no cerrados se redujeron drásticamente. Se detectó además el **RK18: Gran cantidad de correcciones por reunión con el cliente**, debido a que en esta iteración se planificó la reunión con los clientes de la siguiente etapa, en la cual existía el riesgo de recibir una amplia cantidad de correcciones que retrasarían la finalización del proyecto. Para mitigar esto, se analizó y planificó con antelación las fechas para realizar las correcciones necesarias.

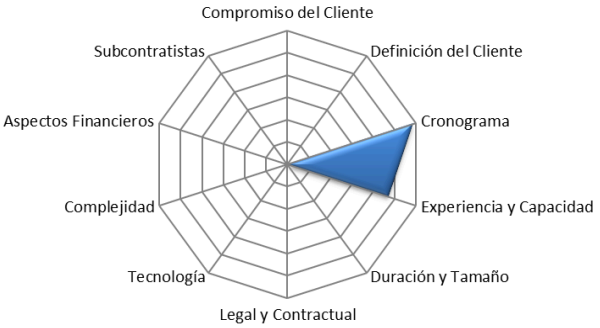

Figura 10: Resumen de riesgos de Construcción 9.

Cierre:

El **RK18: Gran cantidad de correcciones por reunión con el cliente**, detectado en la iteración anterior, no resultó ser un problema ya que, si bien se recibieron correcciones como resultado de la reunión, estas no requirieron una gran cantidad de tiempo, y la retroalimentación recibida fue mayormente positiva.

Figura 11: Resumen de riesgos de la etapa de Cierre.

El **Gráfico 04** permite visualizar de forma clara y resumida cómo evolucionaron los distintos riesgos identificados a lo largo de las etapas del proyecto. A través de este seguimiento se puede observar en qué momentos ciertos riesgos se mantuvieron estables, se mitigaron o dejaron de ser relevantes, así como también aquellos que surgieron con mayor fuerza en etapas específicas.

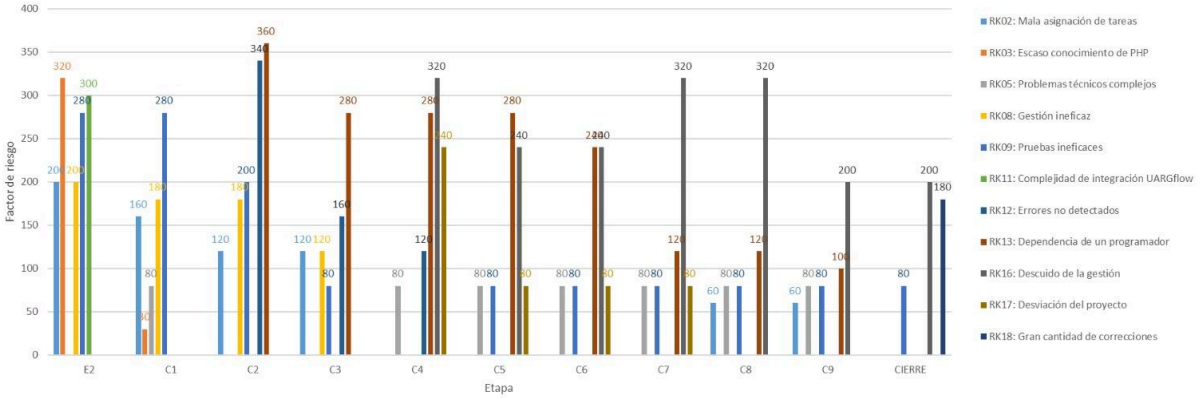

Gráfico 03: Resumen de riesgos por etapa.

Cabe destacar que este gráfico fue realizado con el resumen de los riesgos por el método PSI. En nuestro caso también se gestionaron con el método tongji lo que ayudó en gran medida a elaborar el método híbrido (usando lo mejor de cada uno) para la solución planteada.

El proceso de gestión de riesgos demostró ser fundamental para el avance del proyecto. A pesar de desafíos significativos (como la carga académica, dependencia de miembros clave y brechas técnicas), esta gestión permitió:

- Riesgos inicialmente críticos (RK03/PHP, RK13/dependencia) se neutralizaron mediante capacitación, delegación estratégica y revisiones iterativas.
- La revisión exhaustiva de documentación (para RK17) evitó desviaciones tras la suspensión por finales, facilitando la reactivación eficiente.
- El riesgo RK18 (correcciones masivas del cliente) no materializó su impacto potencial gracias a planificación anticipada y alineación temprana.

Validación y verificación

La verificación del sistema se realizó mediante casos de prueba de valores límite, funcionalidad, integración, ortografía y usabilidad, documentados en una plantilla de excel. Cada prueba contaba con: la característica a probar, los pasos a seguir, el resultado esperado y el resultado obtenido. Cada caso de uso fue diseñado y ejecutado por un integrante del equipo de desarrollo que no participó de la implementación de la característica a probar, y los defectos encontrados fueron documentados en el informe de verificación, el cual contiene para cada caso de prueba: una descripción del problema encontrado, un estado (pendiente, corregido y resuelto), y el identificador del caso de prueba que detectó el defecto. Una vez que un defecto era marcado como corregido (se realizaron modificaciones para corregir el error), se realizaban pruebas de regresión para verificar la corrección efectiva del defecto, y la ausencia de efectos colaterales. Una vez comprobadas ambas, se modifica el estado por resuelto.

La validación del sistema, durante la cursada de la materia Laboratorio de Desarrollo de Software, se realizó mediante presentaciones del trabajo realizado en todas las clases de la materia, en las cuales se recibía retroalimentación de los profesores. Para validar lo realizado en post-cursada, se coordinó una reunión extracurricular con los profesores de la cátedra, antes de la inscripción a la mesa de examen por parte de los integrantes del grupo de desarrollo.

A su vez, los miembros del grupo realizaron verificaciones de los documentos antes de ser entregados, tal como se contempló la planificación de la iteración, cada uno de los miembros revisó los artefactos generados y señaló correcciones de ser necesario.

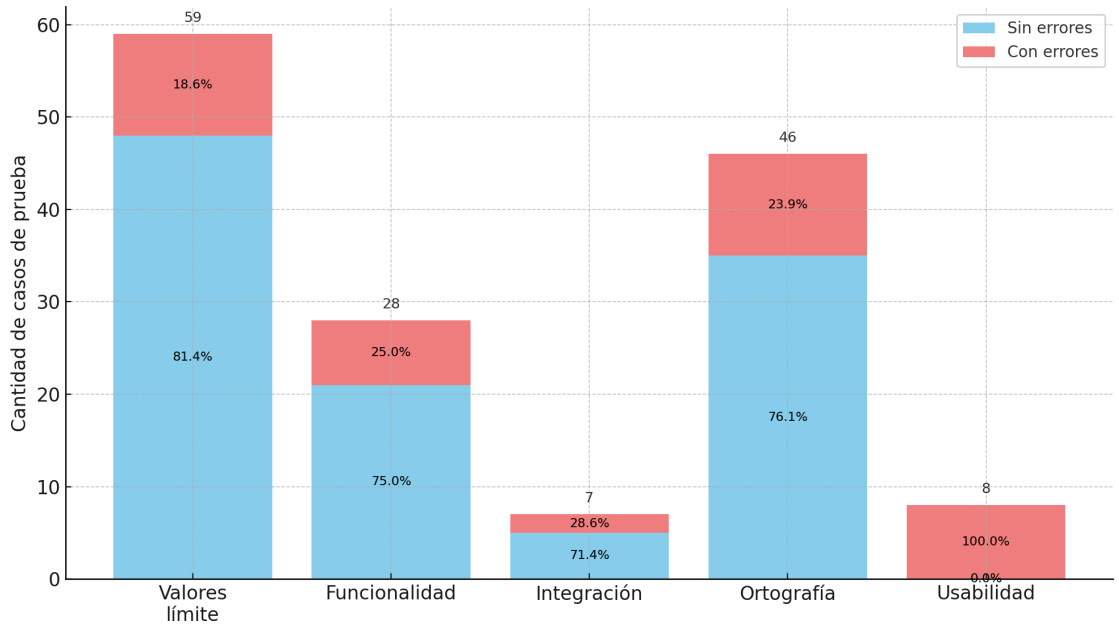

Gráfico 04: Resumen de pruebas por categoría.

Resultados

Hitos destacados

Durante la etapa de **inicio**, se investigó sobre las soluciones de software orientadas a la gestión de riesgos, se realizó la propuesta de desarrollo, la primera entrevista con los clientes, y otras actividades orientadas a generar los documentos básicos en los que se plantearon las características principales y los objetivos del proyecto.

Durante la etapa de **elaboración**, se generaron la especificación de requerimientos de software, y los planes de calidad, estimación, de proyecto y de pruebas. También se realizaron los modelos de casos de uso, de datos y de diseño, necesarios para iniciar la implementación.

Durante la etapa de **construcción**, entre las iteraciones 1 y 3, se diseñó la arquitectura del sistema y se implementaron las funcionalidades principales, siendo estas la administración de usuarios, las actividades de gestión de riesgo: identificación, evaluación y planificación, entre otras. Entre las iteraciones 4 y 9, se revisó y corrigió documentación existente, y se implementaron las funcionalidades faltantes, como la administración de categorías de riesgos y la generación, y exportación de informes y gráficos.

Finalmente, durante la etapa de **cierre**, se actualizaron los manuales de usuario e instalación, se realizó una reunión con los clientes, requerida antes de la presentación al final de la materia, y se realizaron las correcciones solicitadas en dicha reunión.

Lecciones aprendidas

Problemas encontrados

1. Al inicio del desarrollo, la planificación se organizaba de modo que cada miembro del equipo era responsable de completar una sección específica de cada documento. Sin embargo, este enfoque presentaba varios problemas: era necesario coordinar cuidadosamente los *commits* para evitar sobrescribir el trabajo de otros; todos los miembros debían estar al tanto del progreso de los demás para no duplicar esfuerzos sobre los mismos documentos; y, en muchos casos, una sección dependía de otra asignada a un compañero, por lo que era necesario esperar a que esta estuviera terminada y revisarla antes de continuar con el propio trabajo. Esto generó retrasos y pérdida de información por sobreescrituras con información no actualizada.
2. Al momento de comunicar los problemas encontrados de las plantillas del PSI, el equipo, al no contar con la experiencia sobre esta temática, tuvo diferentes errores para expresarlo y no generar un conflicto con los docentes de la cátedra. Esto pudo llevar a que los docentes se sintieran cuestionados y no estuvieran dispuestos a ayudar con el desarrollo del software.

3. Durante la cursada, se desarrolló un diagrama de clases que no era eficiente debido a referencias innecesarias. Poder solucionarlo llevó cierto tiempo, ya que se buscaba garantizar un diseño eficiente. Esto se debió a que nunca se había elaborado un diagrama de clases correspondiente a un proyecto tan complejo como el elaborado. Por otra parte, al comparar el modelo de datos con el diagrama de clases, género ciertas desviaciones, debido a la redundancia que se generaba. Esto derivó en problemas en la implementación de la base de datos lo que generó ciertos retrasos en el desarrollo del sistema.
4. En distintas etapas se solicitó realizar documentos con los que el equipo nunca había experimentado, como la arquitectura de software, el modelo de diseño, el plan de garantía de calidad, entre otros. Además, el tiempo acotado para desarrollarlos, y sin una totalidad de ejemplos representativos para elaborarlos dificultó su correcta elaboración.
5. Uno de los principales problemas que surgieron después de la cursada fueron las estimaciones, esto se debió a que no se comprendieron correctamente los parámetros que podían modificarse ni cómo se debía actuar cuando un caso de uso era más complicado de lo que parecía. Como consecuencia, estas estimaciones representaban que no se había desarrollado nada, lo que es entendible si siempre está aumentando. Otro problema relacionado fue representar adecuadamente los avances en la documentación del proyecto y de los casos de uso. Las restricciones de la plantilla no permitía reflejar de forma precisa los avances que se tenían en cada iteración.

Soluciones aplicadas

1. Se reorganizó la planificación para que cada miembro del equipo fuera responsable de un documento específico. Se hicieron excepciones en el caso de documentos muy extensos o en iteraciones donde esta asignación dejara a algún miembro con pocas tareas. Los demás integrantes del equipo revisaban el contenido y realizaban sugerencias al responsable, pero no modificaban directamente el documento, lo que ayudó a evitar conflictos, mejorar la coordinación y el cumplimiento de las fechas límite planificadas.
2. En las próximas reuniones, cuando se mencionaban las problemáticas que se buscaban resolver con el proyecto Vesta Risk Manager, se optó por destacar la utilidad de las plantillas PSI y presentar sus limitaciones como oportunidades de mejora. De esta manera, se evitó que los docentes se sintieran cuestionados, promoviendo un diálogo constructivo y facilitando su disposición a colaborar en el desarrollo del proyecto.
3. Se planificaron tareas específicas para corregir el modelo de clases basándose en las sugerencias recibidas, junto con la revisión de otros diagramas y documentos relacionados, como el modelo de datos.

4. En las reuniones con los docentes, un integrante del equipo se encargó de registrar todas las observaciones y correcciones sugeridas. Estas correcciones se implementaban lo antes posible. En las últimas iteraciones, una vez adquirida mayor experiencia en la elaboración de documentación, se dedicó una iteración completa a la revisión y mejora de todos los documentos producidos hasta ese momento.
5. Se corrigieron errores en las plantillas de estimación, siendo el más significativo la falta de actualización de las horas dedicadas al proyecto, que habían disminuido a menos de la mitad en las iteraciones posteriores a la cursada. También se ajustó la complejidad de los casos de uso que, aunque no estaban finalizados, ya presentaban un grado considerable de avance. Esto permitió reflejar con mayor precisión el progreso real del proyecto.

Funcionalidades implementadas

El diagrama de casos de uso representa las funcionalidades principales del sistema y cómo interactúan los distintos tipos de usuarios con estas. Se identificaron tres actores: **Administrador**, **Líder del proyecto** y **Desarrollador**.

El **Administrador** tiene acceso a las funcionalidades relacionadas con la configuración general del sistema:

- Administración de usuarios (**CU02**).
- Administración de proyectos (**CU03**).
- Administración de categorías de riesgos (**CU06**).

El **Líder del proyecto** es quien tiene mayor interacción con el sistema, puede:

- Exportar archivos (**CU11**).
- Solicitar y realizar informes (**CU10**).
- Modificar la lista de riesgos (**CU05**).
- Añadir y modificar planes de acción (**CU08** y **CU09**).
- Añadir riesgos a la lista (**CU04**).
- Y realizar análisis y evaluaciones de riesgos (**CU12** y **CU07**).

Por otro lado, el **Desarrollador** participa en funcionalidades un poco más limitadas, que son compartidas con el **Líder del proyecto**, como:

- Añadir riesgos a la lista (**CU04**).
- Añadir plan de acción (**CU08**).
- Y realizar análisis y evaluaciones de riesgos (**CU12** y **CU07**).

Todos los casos de uso están vinculados al **CU01 (Autentificarse)**, ya que fue requisito indispensable que los usuarios inician sesión para acceder al sistema.

El diagrama sirvió de base para comprender la funcionalidad del sistema, facilitando la organización del trabajo.

Diagrama 01: Diagrama de casos de uso de Vesta Risk Manager.

Métricas del proyecto

Para registrar el tiempo real dedicado de la producción del proyecto, tanto durante la cursada como en las etapas “post-cursada”, se optó por utilizar la herramienta "Toggl Track". Cada integrante del equipo activaba un temporizador al momento de realizar cualquier actividad relacionada con el proyecto, ya fuera programación, documentación o preparación de presentaciones. Si bien los tiempos se registraban de forma individual, luego se sumaron para obtener el total semanal de dedicación grupal.

En el **Gráfico 05** se visualiza el tiempo registrado de **forma grupal** a lo largo de un total de 42 semanas. Es importante señalar que entre las semanas 14 y 21 no se registraron actividades debido a la coincidencia con períodos de exámenes finales y fechas festivas. Las semanas 26 y 27 tampoco contaron con actividad por el mismo motivo.

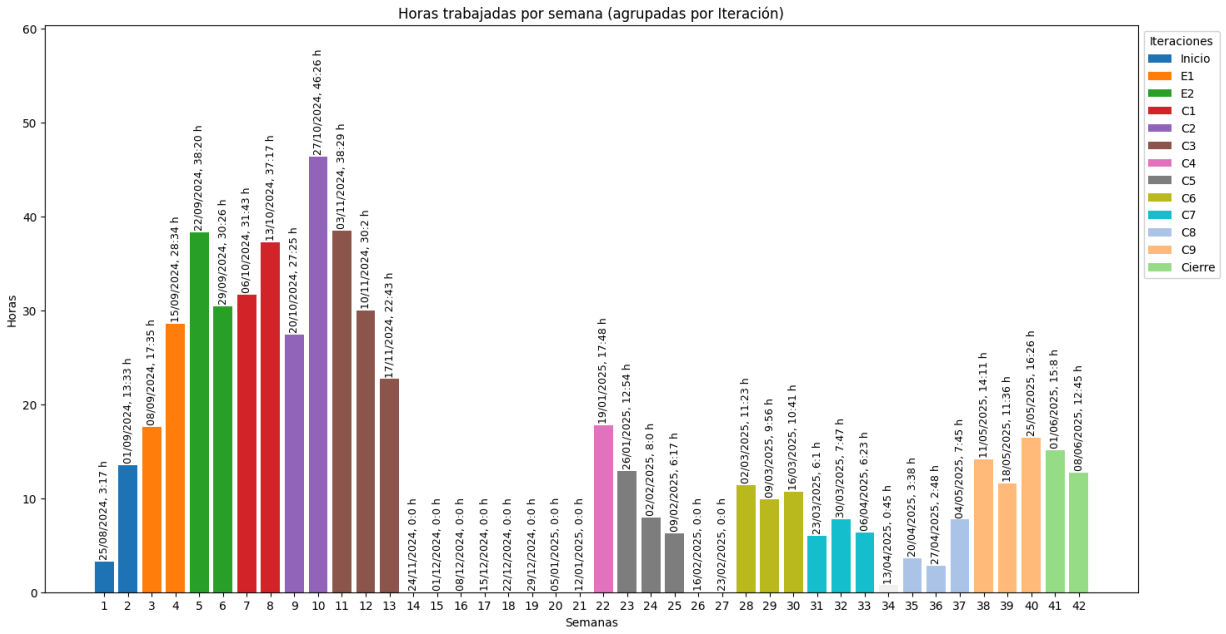

Gráfico 05: Resumen grupal de tiempo real dedicado.

En el **Gráfico 06** muestra la división del esfuerzo del proyecto en tres actividades principales. La mayor parte del tiempo se destinó a la **documentación** con un 53% (282:42:21), seguida por la **codificación** con un 37% (198:18:19) y, en menor medida, a las **presentaciones** con un 10% (56:04:46). Cabe destacar que en la estimación, se tomaba un 40% la codificación y un 60% otras actividades, con los valores obtenidos podemos reemplazarlos en la plantilla y obtener una estimación un poco más eficiente.

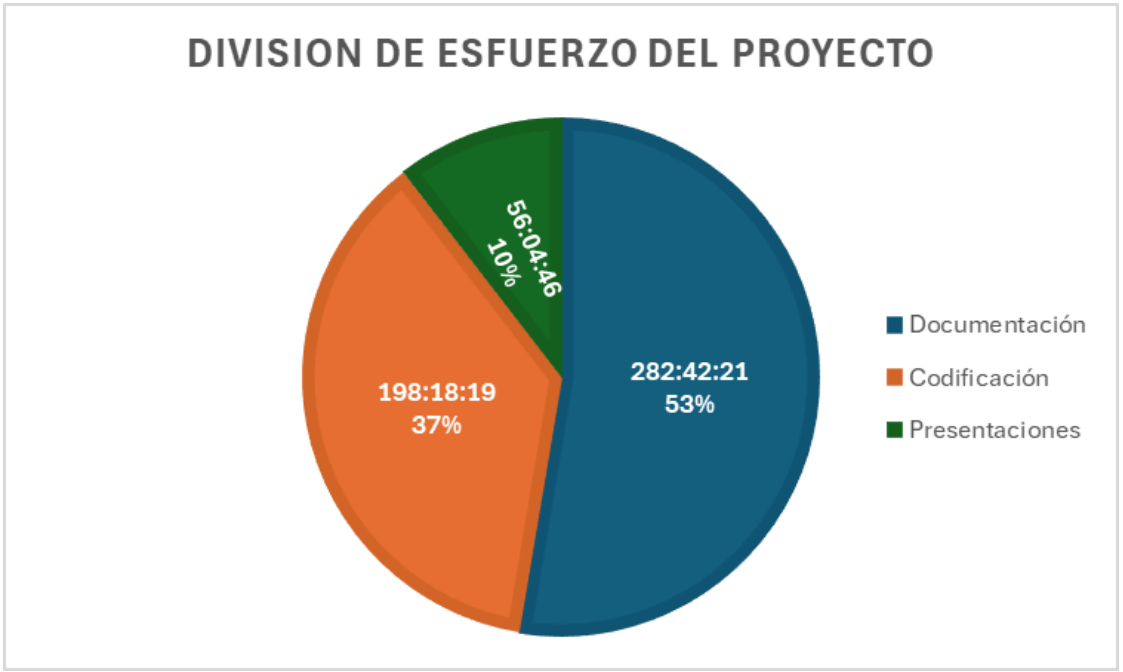

Gráfico 06: Horas y porcentajes de esfuerzo dedicado al proyecto.

Teniendo en cuenta los anteriores gráficos se pueden sacar una métrica valiosa que es la cantidad de días que nos hubiera llevado el proyecto con cierta dedicación.

- Suponiendo que hubiéramos trabajado 2 horas por día nos hubiera llevado 269 días lo que equivale a aproximadamente 9 meses.
- Si hubiéramos trabajado 4 horas por día, nos hubiera llevado 135 días lo que equivale a aproximadamente 5 meses.
- Si se hubiera trabajado 6 horas por día, nos hubiera llevado 90 días lo que equivale aproximadamente 3 meses.
- Si se hubiera trabajado 8 horas por día, nos hubiera llevado 68 días lo que equivale aproximadamente a 2 meses.

Nuestro proyecto inició el 27 de agosto de 2024 y finalizó el 12 de junio de 2025, lo que equivale a 289 días. Sin embargo, se deben considerar los tiempos de trabajo menos estrictos y los tiempos de inactividad del proyecto.

En el **Gráfico 07** se visualiza la cantidad de tareas planificadas y las que fueron cumplidas a lo largo de las diferentes iteraciones del proyecto. Los datos representados fueron obtenidos a partir de los planes de iteración generados durante todas las etapas del desarrollo del sistema. Esta visualización permite evaluar el nivel de cumplimiento en cada etapa.

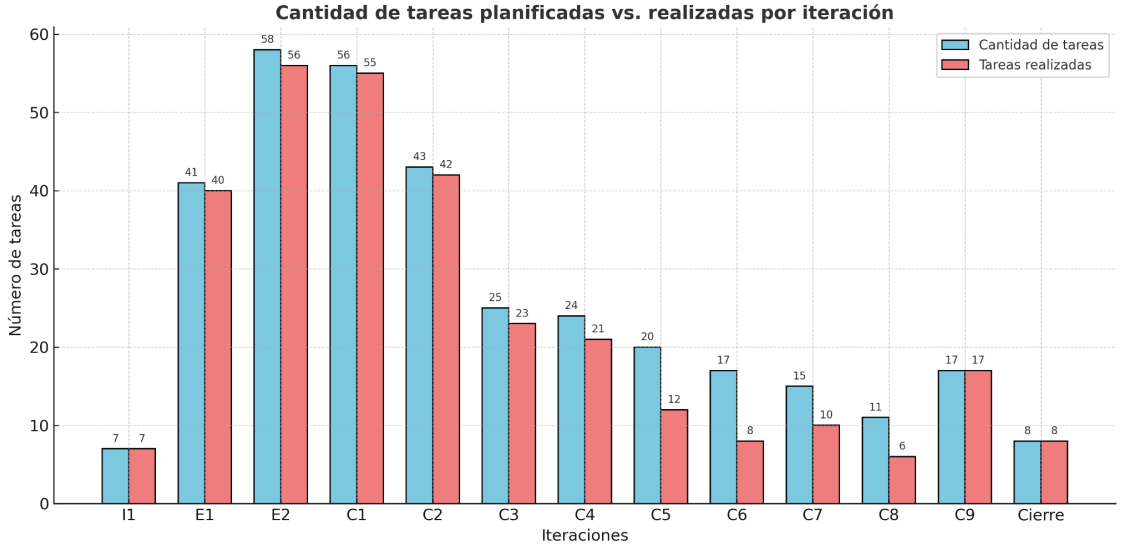

Gráfico 07: Resumen tareas realizadas.

A partir de este gráfico se observó que el equipo logró cumplir con la mayoría de las tareas planificadas en cada iteración, destacando las etapa de Inicio, Construcción 9 y Cierre, las cuales fueron realizadas por completo.

Sin embargo, entre las iteraciones Construcción 5 y Construcción 8, se visualiza una disminución de las tareas realizadas, los motivos de esta fueron detallados previamente en la sección correspondiente a cada iteración.

Esta visualización permitió identificar con claridad la variación del cumplimiento en cada iteración y demostrar el trabajo realizado en cada una de las iteraciones.

Conclusión

Experiencia Personal

Agustín Collareda

La realización de este proyecto final en el marco de la cátedra de Laboratorio de Desarrollo de Software representó una oportunidad valiosa para aplicar los conceptos que anteriormente habíamos aprendido de forma mayoritariamente teórica en otras materias, y en forma práctica en actividades mucho más simples de lo que representan en un contexto real. Antes de comenzar el proyecto, si bien conocía los fundamentos de la gestión de proyectos, calidad, riesgos, pruebas de software, entre otras, sentía dudas sobre mi capacidad para llevar adelante un proyecto de desarrollo de software debido a la falta de experiencia práctica.

Durante el desarrollo del proyecto, hubo desafíos que nos permitieron desarrollar habilidades concretas en cada una de las áreas mencionadas. Desde la planificación hasta las reuniones con los profesores en su rol de clientes, la ejecución de pruebas y la coordinación del equipo, fueron actividades que nos permitieron aprender y mejorar nuestras capacidades en el desarrollo de proyectos de software.

Finalizado el proyecto, me siento mucho más preparado para afrontar desafíos similares en el mundo profesional. Aunque soy consciente de que aún tengo mucho por aprender, esta experiencia me dio mayor seguridad y confianza en mi formación, y demostró la importancia de la práctica para afianzar los conocimientos adquiridos.

Cintia Hernandez

Trabajar en el proyecto *Vesta Risk Manager* fue una experiencia muy valiosa e importante. Al comienzo fue complejo organizarnos como grupo y definir claramente por dónde empezar, pero con el tiempo fuimos mejorando en cuanto a tiempos, comunicación, distribución de tareas y toma de decisiones. Esta experiencia me sirvió muchísimo para aplicar los conocimientos adquiridos a lo largo de la carrera, y me ayudó a comprender cómo es realmente participar en un proyecto colaborativo, con tiempos establecidos, decisiones compartidas y una comunicación constante con los clientes. Fue un proceso que nos exigió compromiso, responsabilidad y adaptación a los retos que aparecieron a lo largo del desarrollo.

Además, trabajar con un cliente real nos obligó a ser más profesionales, a escuchar activamente sus necesidades y a traducirlas en soluciones funcionales. También aprendimos a documentar cada etapa del proceso, algo que muchas veces se subestima pero que fue fundamental para organizar el trabajo, tomar decisiones y no olvidar los objetivos del proyecto.

Lo más lindo fue ver cómo una idea inicial fue tomando forma hasta convertirse en un sistema funcional y concreto. Me voy con la sensación de haber aprendido mucho y de que todo el esfuerzo valió la pena.

Hugo Frey

Partiendo de mi punto de vista, participar en el proyecto y desarrollarlo fue una actividad costosa y gratificante. Esto se debió principalmente a tres causas, la primera hace referencia a la gestión de proyectos. Mucho de los contenidos teóricos referido a esta temática no los había podido aplicar correctamente o no lo entendía. A medida que se fueron desarrollando las iteraciones y las fases pude terminar de cerrar la idea sobre los conceptos de la planificación, estimación y gestión de riesgos. Esto fue debido a que se entendió el significado de porque se hacía lo que se hacía.

La segunda causa hace referencia a los artefactos generados. doy pie a este tema, debido a que fue uno de los más complicados para mi, ya que hubieron muchos documentos que en primer momento no tenían sentido, otros que me preguntaba porque no estaban antes y unos que nunca en mi vida hice. Me gusto mucho la actividad de establecer el objetivo del sistema pero la termine de entender en una de las últimas iteraciones, un dia me consultaron qué hacía nuestro software y al responder con ese mismo objetivo lo pude relacionar con todo el desarrollo que hicimos y que terminó de cerrar la idea que habíamos propuesto en esa actividad.

La tercera y última causa es la presión constante, al tener plazos de entregas muy cortos generaron muchas situaciones de estrés en mi vida tanto estudiantil como personal, aun así, la experiencia de trabajar con esta presión la valoro mucho, ya que a día de hoy, planificar mis actividades con plazos muy cortos de tiempo han permitido organizarme y aliviarme bastante.

A modo de cierre, trabajar con el equipo que tengo lo valoro mucho, siento que muchas veces si no hubiera tenido a un grupo tan dedicado como el nuestro hubiera sido 3 o 4 veces más difícil. Por otro lado, la habilidad para comunicar efectivamente es otra que aprecio haberla trabajado y me gustaría poder seguir perfeccionando el software a medida que voy aprendiendo diferentes herramientas, técnicas, usos. En la última iteración y en la elaboración de esta memoria he cerrado muchas de las incógnitas que he tenido desde que inicié la carrera y me siento capaz de seguir adelante.

Experiencia Grupal

Como equipo de desarrollo, enfrentamos algunas dificultades a lo largo del proyecto, especialmente relacionadas con la comunicación, diferencias en la forma de abordar la implementación y la elaboración de la documentación, así como problemas de organización y coordinación. Estas complicaciones, aunque no fueron demasiado graves, sí generaron algunos desafíos en las primeras etapas del trabajo. Sin embargo, fueron disminuyendo progresivamente a medida que avanzábamos en el desarrollo del proyecto y mejorábamos nuestra dinámica de trabajo en equipo.

A lo largo del proyecto, además de aprender a resolver conflictos internos y a coordinarnos mejor, comprendimos la importancia del trabajo en equipo. En muchas ocasiones, alguno de

los integrantes tenía mayor conocimiento o una mejor comprensión de ciertos problemas, lo que facilitaba su resolución. En otras, ninguno estaba completamente seguro de cómo proceder, pero al intercambiar ideas y trabajar en conjunto, lográbamos encontrar una solución. Esto nos permitió apoyarnos mutuamente, confiar en nuestras capacidades como grupo y ser más pacientes. Trabajar juntos nos permitió apoyarnos mutuamente, distribuir las responsabilidades de manera más equilibrada y enfrentar los obstáculos que, de forma individual, hubieran sido más difíciles de resolver.

También aprendimos a valorar la importancia de planificar con claridad, distribuir tareas de forma realista y ser flexibles ante los imprevistos. Al principio, era común subestimar el tiempo que requería una actividad o asumir que todos entendíamos lo mismo respecto a una tarea. Esto generó algunas confusiones, pero con el tiempo fuimos afinando la comunicación y aprendimos a ser más específicos y detallados al planificar. Esta mejora nos permitió avanzar con mayor orden y reducir retrabajos.

Además, experimentamos de primera mano lo que implica adaptarse a un ritmo de trabajo sostenido en el tiempo, con entregas periódicas y la necesidad de mantener una documentación actualizada. Aprendimos que un equipo no solo se construye con conocimientos técnicos, sino también con responsabilidad, compromiso y respeto por el trabajo del otro. Esto se reflejó en las etapas finales, donde a pesar del cansancio acumulado, logramos mantener el foco y cumplir con los objetivos propuestos.

Retroalimentación de la cátedra

Redundancia en las plantillas de documentación: Observamos que varias secciones se repiten en diferentes plantillas, como es el caso de los requerimientos y los diagramas de casos de uso, que aparecen tanto en la plantilla de *Especificación de Requerimientos* como en la del *Modelo de Casos de Uso*. Consideramos que, para los fines pedagógicos de la cátedra, estas repeticiones podrían reducirse o integrarse de manera más eficiente, facilitando así el trabajo y evitando duplicaciones innecesarias que luego se vuelven difíciles de gestionar.

Falta de claridad en ciertas actividades prácticas: Al inicio de la cátedra, nos encontramos con dificultades para abordar temas como la gestión de configuraciones, el control de calidad y la administración del repositorio Git, ya que no contábamos con experiencia previa en estas áreas. Sería útil que los docentes brindaran una guía más clara o recursos específicos sobre qué se espera en estos aspectos, especialmente en las primeras etapas del proyecto. Por otra parte, se podría realizar un análisis o recapitulación al final de cada una de las iteraciones respondiendo a la pregunta porque se hizo tal documento y para qué serviría en un caso real, también se podría realizar una presentación como se dio en la gestión de riesgos.

Anexo

Repositorio GitHub

- https://github.com/fxex/Vesta_Risk_Manager.git