Universidad ORT Uruguay

Facultad de Ingeniería

Obligatorio – Entrega 1

Ingeniería de Software Ágil 2

Ana Laura Montes de Oca (146669)
Giuliano Rositto (256201)
Sebastián Silvera (242951)

Docentes:
Carina Fontan
Alvaro Ortas

Entregado como Avance de Obligatorio

https://github.com/IngSoft-ISA2-2023-2/obligatorio-montes-de-oca-rositto-silvera

18/Setiembre 2023

DECLARACIÓN DE AUTORÍA

Nosotros, Ana Laura Montes de Oca (146669), Giuliano Rositto (256201) y Sebastián Silvera (242951) declaramos que el trabajo que se presenta en esa obra es de nuestra propia mano. Podemos asegurar que:

• La obra fue producida en su totalidad mientras realizábamos el curso de Ingeniería de Software Ágil 2

• Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad.

• Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra

• En la obra, hemos acusado recibo de las ayudas recibidas

• Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros

• Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

Contenido

Definición del marco de trabajo

Marco general de trabajo KANBAN

El resultado del proyecto es poder mejorar y ofrecerle mantenimiento a una aplicación ya desarrollada por otro equipo con el objetivo de incrementar el valor que se entrega a los usuarios.

Dicho marco de trabajo es desconocido para los integrantes del equipo por lo que va a haber una cierta curva de aprendizaje dentro del desarrollo de la aplicación.

El equipo de desarrollo va a utilizar la forma de trabajo basada en Kanban y además optamos por seguir un ciclo de vida incremental del producto.

Definición y uso del proceso de Ingeniería en proceso Kanban

La Ingeniería de Requerimientos corresponde a un proceso sistemático para el descubrimiento, desarrollo, trazabilidad, análisis, clasificación, comunicación y Gestión de Requerimientos, el cual define un sistema en niveles sucesivos de abstracción. Dicho proceso busca fundamentalmente facilitar las tareas que soporten los demás procesos de negocio al interior de una organización, y como todo proceso, sirve de entrada y salida para muchos otros.

Este proceso, que hemos aprendido a lo largo de varios cursos en la carrera y además tiene sus beneficios y sus desventajas. Muchas y variadas empresas han encontrado una brecha bastante amplia entre lo relevado, tan específico y detallado, que además se realiza en el marco de una metodología en cascada, es un proceso lento, costoso y que no favorece la entrega del producto final.

Las metodologías ágiles han venido para tratar de acortar esta brecha, bajando la intensidad de estas etapas, utilizando artefactos, ceremonias y herramientas visuales que favorecen la calidad, la mejora continua, la flexibilidad, la reducción del desperdicio, entre otros.

En particular Kanban es una metodología de gestión visual que se utiliza comúnmente en entornos de trabajo para mejorar la eficiencia y la productividad. Su objetivo principal es optimizar el flujo de trabajo y la entrega de productos o servicios al eliminar el exceso de trabajo en proceso y minimizar los cuellos de botella.

¿Cómo se mapea esta Ingeniería de requerimientos con nuestra metodología Kanban?

El proceso de Ingeniería de requerimientos se mapea a un proceso muy simple de Planning, dentro del proceso completo Planning-Do-Check, donde las tares de Planificación se corresponden a entrevistas con el cliente donde se describen sus necesidades que luego son transformadas en requerimientos de software.

Es importante en esta etapa identificar las necesidades propias del cliente, así como requerimientos de entorno o potenciales cambios que puedan delimitar nuestro escenario de implantación.

Esta etapa fue lograda en nuestro proyecto a través de la lectura del Obligatorio (la rúbrica), los requerimientos que fueron explicando los profesores y de alguna forma 'desmenusando' estas necesidades en tareas.

Seguidamente este listado de necesidades identificadas es transformado a requerimientos, en general redactados como User Storys, en nuestro caso, aún no contamos con user story si no solo con tareas.
Estas tareas deben ser redactadas de forma clara y precisa evitando ambigüedades. La lista con los requerimientos obtenidos, es priorizada, categorizada teniendo en cuenta cada una de las tareas propuestas, las cuales conforman el producto de trabajo comúnmente conocido como Product Backlog.

Estos requerimientos forman parte del TODO: para hacer, que luego se van moviendo a una o varias columnas que conforman un artefacto visual: El tablero Kanban, donde luego de finalizadas y si cumple los criterios de aceptación se mueven al DONE.

Este proceso de mover las tareas entre las diferentes columnas del Tablero, cumple con el Check, Control y Seguimiento pudiendo visualizar last tareas pendientes, en desarrollo y realizadas. Además permite agregar múltiples atributos que luego explicaremos en detalle.

Cómo adaptamos los 6 principios Kanban a nuestro marco de trabajo:

1) Visualizar el flujo de trabajo:

Dentro de nuestro repositorio creamos un proyecto de nombre Entrega 1 donde iremos gestionando nuestras diferentes entregas. Estas entregas, así como en estas utilizaremos tableros dinámicos, que se podrán ir modificando en la medida que el proyecto así lo requiera. Está visible para todos los integrantes del equipo y cada uno puede gestionarlo a medida que avanza.

2) Limitar el WIP:

Limitamos el trabajo en curso para evitar cuellos de botella y para garantizar que solo entran elementos nuevos cuando hay capacidad de completarlos (no se produce por encima del cuello de botella).
Esto lo logramos evitando que cada uno trabaje en 2 tareas/issues a la vez.

3) Gestionar y medir el flujo:

Medir para ver si se mejora o no y decidir en consecuencia. Podría ser número de tareas completas en el Sprint, o número de Story Points completados, el Lead Time, el Cycle Time, etc..

Como en esta Etapa tendremos básicamente tareas de diferentes titpo, hemos decidido medir con la unidad de esfuerzo: horas/hombre y no user stories.

Primero porque no tenemos US y luego porque si medimos en días hay tareas que no llevan un día, si no horas. Es una unidad que nos permite medir unidades más pequeñas.

4) Implementar ciclos de feedback:

Son muy importantes y de mucho aprendizaje las retrospectivas y las stand ups. Hemos tenido dificultades para encontrar momentos en común, como lo dejamos reflejado en las retrospectiva, igualmente vamos a intentar mejorar para próximas entregas.
son los standups y retrospectivas. Se debe recibir el feedback en todos los niveles (equipo, cliente, PO, etc.).

5) Explicitar políticas y procedimientos:

Por ejemplo los “Definition of Done”, “Definition of Ready”, valores y principios.
Esto lo vemos en la sección del capítulo que de Políticas, Definition of Donde y Definition of Ready.

6) Mejora continua mediante la colaboración:

Usar modelos y experimentar científicamente: definir una hipótesis, implementar un experimento y medir las consecuencias. Si funciona se sigue y si no comienzas de nuevo el ciclo.

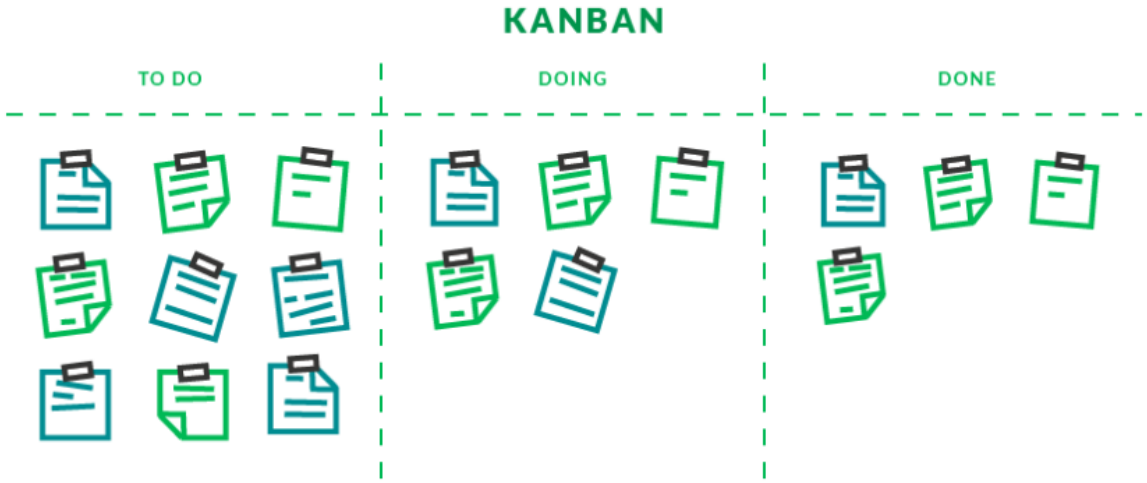
Esta sección la comenzamos a experimentar con los commits de trunk-based. No conocíamos la técnica, la experimentamos y de ahora en más es la que vamos a utilizar o al menos hasta que surja otra que se recomiende en el curso.
Seguro encontraremos otras instancias para este aprendizaje experimental.

Tablero Kanban

https://github.com/orgs/IngSoft-ISA2-2023-2/projects/10

Respecto al tablero:

Para esta iteración se promueve la utilización de un Tablero Kanban Ágile Simple (aunque sabemos que poco a poco se va a ir transformando en Sustentable)



Optamos por esta opción de Tablero y no sustentable, ya que entendemos que las tareas tienen diferentes niveles de desarrollo, algunas pasan por análisis y desarrollo, otras simplemente por ser realizadas como calcular el esfuerzo o las métricas o un informe que requiere lectura previa y reflexiones.

Todas los tipos de tareas se pueden mover a un estado: DOING, donde indicamos que se está desarrollando esa tarea.

Columna 1: Por Hacer (To Do): En esta columna, se enumeran todas las tareas o elementos de trabajo que aún no han comenzado. Representa las tareas pendientes que deben abordarse.
En esta etapa de entrega se encuentran tareas propias de planificación y diseño del Tablero, análisis y preparación de materiales, así como las propias tareas de testing exploratorio propuestas en el proyecto.

Al principio habíamos desglosado en tareas por Roles del proyecto como Admin, dueño, Farmacia, anónimo, nos pareció que quedaban muy abiertas y muy genéricas y no nos iban a permitir calcular buenas métricas, o al menos más detalladas, ya que un Rol puede tener varias funcionalidades.

Luego, fuimos discriminando por Rol - Funcionalidad ya que de esta manera quedaba encapsulada, en lo que luego podría ser una user stroy, para estimar, calcular tiempos de esfuerzo, nos resultó más simple.

Columna 2: En Progreso (In Progress - DOING): Las tareas que están en proceso se mueven a esta columna. Aquí, los miembros del equipo trabajan activamente en estas tareas. El objetivo es limitar la cantidad de tareas en esta columna para evitar la sobrecarga. Y limitar el numero WIP.
Al moverlas se coloca fecha - hora de comienzo y a quien se le asigna.

En nuestro caso en particualr, cada integrante del equipo trabajó en las tareas que fue tomando, agregando fecha y hora de comienzo así como fecha y hora de fin cuando las movía a la columna DONE.

Se limita la cantidad de tarjetas que pueden estar en la columna "En Progreso" para evitar la congestión y fomentar la finalización de tareas antes de agregar nuevas. Es decir limitar el WIP.
Cada desarrollador debe realizar de a una tarea a la vez.

Columna 3: Completado (Done): Cuando una tarea se ha finalizado o completado, se traslada a esta columna. Aquí, se muestra claramente qué tareas se han terminado y están listas para su revisión o entrega. Al moverla a DONE se agrega fecha y hora de finalización

Columna 4:
Al ir avanzando en el aprendizaje de la herramienta y metodología vimos que creamos tareas que estaban repetidas, o que habían sido desglosadas en otras más pequeñas o que simplemente las habíamos propuesto pero en realidad no llegamos con los tiempos, para este caso creamos la columna DEPRECADOS.

Conceptualmente se deberían definir límites de WIP: En cada columna se establece un límite de cuántas tarjetas pueden estar en esa etapa al mismo tiempo. Esto evita la congestión y garantiza un flujo de trabajo constante.

En particular, en nuestro equipo limitamos la cantidad de tareas a la vez, que cada miembro puede trabajar.

Roles

En principio y por requerimiento de la letra del obligatorio, todos los integrantes del equipo somos desarrolladores y testers.

Aunque en la metodología KANBAN no tenemos la figura del SCRUM MASTER como obligatoria, vemos imprescindible en esta transición este rol, para ayudarnos a gestionar el tablero, indicar los atributos que tenía que tener cada tarjeta, así como promover las stand up.

Product Owner (PO): en esta etapa de aprendizaje y transición creemos conveniente que los 3 integrantes seamos PO, para llegar a entender con mayor exactitud y claridad los requerimientos y necesidades de los clientes, sobre todo de las prioridades y severidades. Los 3 podemos asignar prioridades y severidades así como opinar y solicitar cambio de alguna tarea que fue asignada a otro integrante del equipo.
Entendemos que esto es importante para fortalecer el equipo y tener una escucha activa y efectiva ante las necesidades.

Políticas de trabajo

Tuvimos muchas dificultades al comienzo del Obligatorio para entrar en 'ritmo', nos costó organizarnos, encontrar espacios en común para las Stand Up y la planificación. Se visualiza en las retrospectiva el reflejo de esta mención, ya que se proponen como acciones correctivas buscar espacios en común síncronos, así como gestionar de forma uniforme el tiempo.

Definición de Ready

En esta entrega del Obligatorio no tenemos historias de usuario definidas como tal, solamente hemos incluido tareas.
Pero es deseable y lo dejamos previsto para las próximas entregas las siguientes pautas:
Antes de que una historia de usuario pueda ser incluida en una iteración, es necesario que cumpla con ciertas condiciones previas.
Estas condiciones incluyen que el equipo de desarrollo debe tener una comprensión compartida de lo que la historia de usuario significa y qué se espera de ella.
También se debe presentar una estimación relativa del trabajo necesario para completar la historia de usuario, y los criterios de aceptación deben ser claros y representados en escenarios para que puedan ser comprobados.
Acá aún debemos definir si vamos a realizar la estimación a través de la técnica de Pocker impartida en ISA 1 u otra técnica.

Además, la historia de usuario debe ser valiosa y no depender de otras historias de usuario para su completitud, sabemos que debe cumplir con los criterios de INVEST
En resumen, estas condiciones previas aseguran que el equipo de desarrollo tenga toda la información necesaria y comprenda completamente lo que se espera de la historia de usuario antes de comenzar a trabajar en ella.

La definición de lista de requisitos previos para una historia de usuario entrar en una iteración incluye que:

1. El título debe ser breve y muy descriptivo.
2. La narrativa debe tener una estructura que explique qué hace el usuario, qué quiere y por qué lo quiere, esquema (Cómo, Quiero, Para...)
3. La estimación de tiempo debe estar en una unidad de tiempo acordada y estar en valores predefinidos.
4. Los criterios de aceptación deben ser claros y presentados como escenarios, explicando el contexto, el evento y los resultados esperados.

Definición de Done

Los criterios para determinar qué historias terminaron con éxito su proceso de implementación serán:

1. Las validaciones correspondiente con el cliente, ajustando su correspondiente feedback. En este caso nuestros clientes serán los docentes y los 'potenciales' clientes de la aplicación que estamos testeando y analizando deuda técnica.

2. El elemento ha sido revisado por un par o por el equipo de desarrollo.
Como los integrantes del equipo somos tres, entendemos suficiente que se validen de a pares. En estos casos la funcionalidad debe ser testeada de forma exhaustiva y/o mejorado la cobertura de código.

3. El elemento ha sido probado y se ha verificado que funciona correctamente en diferentes entornos.

4. Todo el trabajo debe estar integrado a la rama main. En nuestro caso hicimos un merge a la reama main y luego un reléase a Entrega-1

5. La documentación ha sido actualizada y se ha revisado para asegurarse de que esté completa y precisa.

Control de versiones

Se crea el repositorio <https://github.com/IngSoft-ISA2-2023-2/obligatorio-montes-de-oca-rositto-silvera>

Iniciamos nuestro trabajo utilizando la metodología de GitFlow pero será nuestra intención a futuro migrar a una metodología Trunk-Based mientras investigamos características de la misma.

Considerando esta transferencia creamos ramas paralelas desde main y trabajamos sobre ellas hasta considerar nuestro trabajo completo, en donde mergeamos el código a una rama develop. Una vez consideramos nuestro código listo para release mergeamos nuestro Código en develop a main y luego continuamos trabajando en develop.

Ante nuestra transición entre gitflow y trunkbased iremos introduciendo metodologías de trunk lentamente hasta alcanzar la modalidad, por ejemplo haciendo pruebas sobre esta en algunas de nuestras ramas paralelas.

La rama issues-merge es una rama sobre la cual probaremos utilizar la metodología trunk-based.
Para esta entrega ya experimentamos esta metodología, notando que es más ágil, rápida y con mejores resultados ya que los merge resultan más simples

Nombre de ramas, commits y PRs

1.Para nombrar las ramas:

- Los nombres deben ser descriptivos y concisos.
- Para trabajos similares se añade el nombre del desarrollador para distinguir las ramas en la que trabajo cada uno
- Utilizar minúsculas y separa las palabras con guiones.
- Agregar prefijos como "feature/" para nuevas características, "bugfix/" para correcciones de errores, "hotfix/" para soluciones urgentes, entre otros.

2.Para nombrar los commits:

- Los nombres deben ser claros y descriptivos.
- Utilizar verbos en tiempo presente para indicar la acción realizada, seguidos de una breve descripción del cambio.
- Limitar la longitud a unos 50 caracteres.
- Evitar agregar información innecesaria o detalles irrelevantes.

3.Para nombrar los Pull Requests:

- Comenzar con un prefijo que indique el propósito, como "Feature:", "Fix:", "Docs:", etc.
- Continuar con una breve descripción del cambio o problema que aborda.
- Utilizar un estilo conciso y claro.

ISSUES - ANÁLISIS DE DEUDA TÉCNICA

Issue 1

a) Login de usuario no existente en la base de datos

Descripción:
Un usuario no registrado aún en la base de datos, intenta loguearse, el sistema responde lanzando una excepción.

Impacto:
Los usuarios se crean por invitación del Admin. o son usuarios 'anónimos'. En este caso al intentar loguearse un usuario no registrado lanza una excepción no controlada:

Excepción

throw new InvalidResourceException("Invalid Password");
Se trata de una exception no contolada, no se devuelve el código de error: 404, como menciona en la documentación
Error 404 - Cuando no se encuentra en el sistema un usuario con el nombre de usuario enviado. ("The user does not exist")
El usuario que está utilizando no puede continuar. Se debe reinciar

Solución ideal:
La solución ideal sería revisar en qué porción del código se da esta excepción y controlarla. Luego devolver y controlar este error en el Front

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta deuda técnica, como:

- Identificar la excepción de error que lanza esta excepción.
- Analizar cada caso para determinar qué tipo de excepción específica debería usarse.
- Analizar si se están utilizando Filtros o bloques try-catch desde el back. Solucionarlo desde esta perspectiva.
- Se inspecciona el código, hay una clase Filtros que no está funcionando correctamente.
- Actualizar la documentación y las pruebas correspondientes.

Clasificación:
- Prioridad: Baja
- Severidad: Menor

Issue 2

Descripción
La funcionalidad del Admin: Alta de Farmacia

Con el Rol Admin, al ingresar a dar de Alta una farmacia se lanza un excepción no controlada, cuando el nombre de la farmacia tiene un largo > 50, que obliga a reiniciar el servidor del back-end.

Impacto:
Estando logueado como Admin y al intentar crear una farmacia con nombre de la misma con largo > 50 se lanza una excepción en el sistema que obliga a reiniciar el server.
Esto genera a nivel de usuario, incomidad y pobre usabilidad ya que no permite continuar. No aparece ningún mensaje de aclaración de que se debe controlar el largo del nombre.
A nivel del usuario Admin que genera las farmacias el impacto es que no se puede generar ningún movimiento a nivel de ésta: Medicamentos, Stock, Solicitudes de Medicamentos, etc.

Excepción
Excepción: PharmaGo.Exceptions.InvalidResourceException: 'The Pharmacy is not correctly created.'

Solución ideal:
Una solución posible es controlar esta excepción y desde el Front emitir el mensaje correspondiente, discriminando en caso de error, sin necesidad que haya una excepción, permitiendo continuar el usuario con el resto de las funcionalidades

Plan de acción:
- Revisar a nivel de Back en qué módulo, método o clase se genera esta excepción.
- Corregir la excepción ya sea con bloques try-catch o a través de filtros.
- Se hizo análisis exploratorio a nivel de código y se utiliza una clase Filtros, revisar y si es necesario corregirla o capturar la excepción a nivel de bloques try-catch
- Realizar pruebas unitarias y de integración.
- Revisar, corregir, mantener y/o superar la cobertura de código.
- Corregir desde el Front la visualización de este mensaje al usuario.

- Actualizar la documentación correspondiente

Clasificación:
- Prioridad: Alta
- Severidad: Crítica

Issue 3

Funcionalidad del Admin: Alta de Farmacia - Con Nombre Duplicado

Descripción:
La funcionalidad del Admin: Alta de Farmacia cuando el nombre está duplicado, lanza una excepción que obliga a reiniciar el servidor del back.
throw new InvalidResourceException("The Pharmacy is not correctly created.");

Impacto:
Sería bastante frecuente en la carga inicial del Admin, al cargar todas las farmacias y hacer pausas entre las mismas que no retenga que farmacias ha ingresado o que quiera ingresar una ya existente.
Ante esta repetición por nombre, el Admin recibe una excepción que obliga a reiniciar el servidor del back.

Solución ideal:
La solución ideal sería que el usuario Admin puede visualizar la lista de Farmacias ya creadas o que ante un nombre duplicado emita un mensaje amigable: 'Este nombre ya existe dentro de las Farmacias del Sistema'

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta deuda técnica, como:

- Identificar el módulo, método o clase donde se registra la excepción.
- Darle el tratamiento correspondiente ya sea a nivel de try catch o Filtros como en los casos anteriores.
- En la invocación del Front al servicio controlar este mensaje de error.
- Realizar pruebas de integración exhaustivas para garantizar que las funcionalidad está correctamente corregida.
- Actualizar el porcentaje de cobertura de código, en lo posible mejorarlo.

Clasificación:
- Prioridad: Inmediata
- Severidad: Crítica

Issue 4

Funcionalidad del Admin: Alta de Farmacia - Sin Dirección

Descripción:
La funcionalidad del Admin: Alta de Farmacia cuando no tiene dirección.
Mensaje genérico, poco explicativo.

Impacto:
Con Rol Admin, al querer ingresar una Farmacia sin dirección proporciona una mensaje bastante genérico sin explicación detallada de cuál es el error.
Esto ocaciona que el Admin no sepa donde está el error, aunque el Admin podría saberlo de antemano si se le dio un instructivo al respecto.

Solución ideal:
La solución ideal sería que el usuario Admin puede visualizar un mensaje más detallado al crear la Farmacia sin dirección.
Esto se debería controlar a nivel de manejo de control de nulos. Si el back está devolviendo este error personalizado, que se pueda controlar en el Font para emitir un mensaje personalizado.

Plan de acción:

El plan de acción podría incluir pasos específicos para abordar esta deuda técnica, como:

- Identificar si el método del back está devolviendo un mensaje personalizado o no.
- Si el mensaje es correcto, entonces analizar en el Front donde se está devolviendo este mensaje para cambiarlo por un mensaje personalizado.
- Realizar pruebas de integración exhaustivas para garantizar que las funcionalidad está correctamente corregida.
- Actualizar en la documentación del Sistema e instructivo si los hubiera

Clasificación:
- Prioridad: Media
- Severidad: Leve

Issue 5

Excepción al crear una invitación para un usuario con el rol de Administrador:

Descripción:
En la funcionalidad de administrador, al intentar crear una invitación para un usuario con el rol de Administrador y habiendo seleccionado una farmacia, se lanza una excepción no controlada con el mensaje "A pharmacy is not required." ("No se requiere una farmacia"). Esta excepción es inapropiada ya que la farmacia realmente fue seleccionada.

Impacto:
La excepción inapropiada al crear una invitación para un usuario con el rol de Administrador puede llevar a confusión y problemas en la administración de usuarios en el sistema. Puede resultar en la incapacidad de asignar el rol correcto y afectar la funcionalidad general de la aplicación.

Solución ideal:
La solución ideal sería corregir la lógica que causa la excepción inapropiada y asegurarse de que se pueda crear una invitación correctamente para un usuario con el rol de Administrador y una farmacia seleccionada.

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta preocupación, como:

- Identificar la parte del código responsable de lanzar la excepción incorrecta.
- Ajustar la lógica para que permita crear invitaciones con el rol de Administrador y una farmacia seleccionada.
- Realizar pruebas exhaustivas para verificar que la corrección funcione de manera adecuada.
- Actualizar la documentación para reflejar los cambios realizados.

Clasificación:
- Prioridad: Media
- Severidad: Moderada

Issue 6

Excepción al iniciar sesión como invitado con las credenciales generadas:

Descripción:
Después de que el administrador crea una invitación, se genera un usuario y un código de invitación, que se listan correctamente en la lista de invitados. Sin embargo, cuando el invitado intenta iniciar sesión con estas credenciales, se lanza una excepción no controlada con el mensaje "The user does not exist" ("El usuario no existe").

Impacto:
La excepción al intentar iniciar sesión como invitado con las credenciales generadas incorrectamente afecta la experiencia del usuario y puede causar frustración. Los usuarios no pueden acceder a la aplicación como invitados, lo que interrumpe el flujo esperado de uso.

Solución ideal:
La solución ideal sería identificar y corregir la lógica que causa la excepción al iniciar sesión como invitado con las credenciales generadas. Los usuarios deberían poder iniciar sesión exitosamente como invitados después de recibir una invitación.

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta preocupación, como:

- Identificar la parte del código responsable de lanzar la excepción incorrecta al iniciar sesión como invitado.
- Ajustar la lógica para permitir que los invitados inicien sesión correctamente con las credenciales generadas.
- Realizar pruebas exhaustivas para verificar que la corrección funcione de manera adecuada.
- Actualizar la documentación para reflejar los cambios realizados.

Clasificación:
- Prioridad: Media
- Severidad: Moderada

Issue 7

Excepción al dar de alta una farmacia sin dirección en la funcionalidad del Administrador:

Descripción:
En la funcionalidad de administrador, cuando se intenta dar de alta una farmacia sin proporcionar una dirección, se lanza una excepción no controlada en el servidor del back-end. Esta excepción no se maneja adecuadamente y requiere reiniciar el servidor para resolverla.

Impacto:
La excepción no controlada al dar de alta una farmacia sin dirección tiene un impacto negativo en la disponibilidad y la estabilidad del sistema. Requiere intervención manual

(reinicio del servidor) para recuperarse, lo que puede causar interrupciones en el servicio y afectar la experiencia del usuario.

Solución ideal:
La solución ideal sería manejar adecuadamente la excepción generada cuando se intenta dar de alta una farmacia sin dirección. Esto debería incluir el manejo de errores en el servidor y proporcionar un mensaje de error claro al usuario.

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta preocupación, como:

- Identificar el código o la lógica que causa la excepción al intentar dar de alta una farmacia sin dirección.
- Agregar un manejo adecuado de la excepción en el servidor para evitar interrupciones y proporcionar un mensaje de error al usuario.
- Realizar pruebas exhaustivas para garantizar que la corrección no introduzca nuevos problemas.
- Actualizar la documentación para reflejar los cambios realizados.

Clasificación:
- Prioridad: Alta
- Severidad: Crítica

Issue 8

Excepción al crear una invitación para un usuario con el rol de Administrador:

Descripción:
En la funcionalidad de administrador, al intentar crear una invitación para un usuario con el rol de Administrador y habiendo seleccionado una farmacia, se lanza una excepción no controlada con el mensaje "A pharmacy is not required." ("No se requiere una farmacia"). Esta excepción es inapropiada ya que la farmacia realmente fue seleccionada.

Impacto:
La excepción inapropiada al crear una invitación para un usuario con el rol de Administrador puede llevar a confusión y problemas en la administración de usuarios en el sistema. Puede resultar en la incapacidad de asignar el rol correcto y afectar la funcionalidad general de la aplicación.

Solución ideal:
La solución ideal sería corregir la lógica que causa la excepción inapropiada y asegurarse de que se pueda crear una invitación correctamente para un usuario con el rol de Administrador y una farmacia seleccionada.

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta preocupación, como:

- Identificar la parte del código responsable de lanzar la excepción incorrecta.

- Ajustar la lógica para que permita crear invitaciones con el rol de Administrador y una farmacia seleccionada.
- Realizar pruebas exhaustivas para verificar que la corrección funcione de manera adecuada.
- Actualizar la documentación para reflejar los cambios realizados.

Clasificación:
- Prioridad: Media
- Severidad: Moderada

Issue 9

Excepción al iniciar sesión como invitado con las credenciales generadas:

Descripción:
Después de que el administrador crea una invitación, se genera un usuario y un código de invitación, que se listan correctamente en la lista de invitados. Sin embargo, cuando el invitado intenta iniciar sesión con estas credenciales, se lanza una excepción no controlada con el mensaje "The user does not exist" ("El usuario no existe").

Impacto:
La excepción al intentar iniciar sesión como invitado con las credenciales generadas incorrectamente afecta la experiencia del usuario y puede causar frustración. Los usuarios no pueden acceder a la aplicación como invitados, lo que interrumpe el flujo esperado de uso.

Solución ideal:
La solución ideal sería identificar y corregir la lógica que causa la excepción al iniciar sesión como invitado con las credenciales generadas. Los usuarios deberían poder iniciar sesión exitosamente como invitados después de recibir una invitación.

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta preocupación, como:

- Identificar la parte del código responsable de lanzar la excepción incorrecta al iniciar sesión como invitado.
- Ajustar la lógica para permitir que los invitados inicien sesión correctamente con las credenciales generadas.
- Realizar pruebas exhaustivas para verificar que la corrección funcione de manera adecuada.
- Actualizar la documentación para reflejar los cambios realizados.

Clasificación:
- Prioridad: Media
- Severidad: Moderada

Issue 10

Excepción Testing Exploratorio Funciones Dueño - Excepción Detalle de Compras Realizadas

Descripción:

Al seleccionar la opción de Listado Detallado de Compras se lanza una excepción que obliga a reiniciar el server.
Excepción lanzada:

Microsoft.Data.SqlClient.SqlException: 'Invalid column name 'PharmacyId'.
Invalid column name 'PharmacyId'.
Invalid column name 'Status'.
Invalid column name 'TrackingCode'.'

Impacto:
Con el Rol como dueño, si se quiere acceder al listado de compras realizadas se lanza una excepción que obliga a reiniciar el server. El impacto es a nivel del usuario, se refleja que el software no está cumpliendo con la funcionalidad esperada, pero sobre todo la experiencia del usuario no es amigable, ya que no puede seguir utilizando el resto de las funcionalidades de su rol.

Solución ideal:
Observando la excepción se muestra que no se está guardando o leyendo bien el ID de la Farmacia.
Se puede revisar a nivel de Front, sería menos costoso solucionar a nivel de form o de cookies si fuera el caso. En caso de no encontrar el error buscaríamos a nivel de Back.

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta excepción, como:

- Identificar la parte del código del Front, si existe, responsable de lanzar la excepción incorrecta al iniciar sesión como dueño y querer acceder al listado.
- Si no encontramos posible error, entonces buscamos en el Back, siguiendo los mismos pasos. Aunque en el back está lanzando la excepción, no se está controlando de forma adecuada.
- Completar pruebas unitarias del back del módulo que está experimentando la excepción
- Mantener o superar la cobertura de código.
- Actualizar la documentación para reflejar los cambios realizados.

Clasificación:
- Prioridad: Media
- Severidad: Crítica

Issue 11
búsqueda de remedios filtra por título exacto

Descripción:
Se tiene que escribir el nombre preciso del medicamento, espacios y números incluidos para encontrar el remedio elegido.

No ocurren casos donde aparecen todos los remedios que tengan la palabra "parace" como paracetamol 200 o paracetamol 500.
Se ignoran las mayúsculas por lo que el error no se extiende más de esto.

Impacto:
Aquellos usuarios que sean poco familiares con el sistema, o no conozcan con exactitud el nombre completo del remedio pueden encontrar dificultades al momento de encontrar su objetivo, posiblemente causando abandono

Solución ideal:
Se implementa un filtro que devuelve resultados que coincidan parcialmente con la búsqueda.

Plan de acción:
-Analizar la ubicación en código del sistema de búsqueda
-Implementar una solicitud a base de datos menos estricta

Clasificación:
- Prioridad : Baja
- Severidad : Menor

Issue 12

No se informa al Usuario cuando una farmacia no posee ningún medicamento que coincida con la búsqueda

Descripción:
La lista de medicamentos queda vacía y no indica una falta de resultados que puede confundirse con un estado congelado de la pagina

Impacto:
Usuarios pueden malinterpretar el estado en blanco del resultado como un error de performance de la página o malinterpretarlo como un error

Solución ideal:
Se implementa un Mensaje que indica la ausencia de medicamentos que cumplan con los filtros indicados

Plan de acción:
Añadir un if module que detecte cuando la lista resultado esta vacía y cargar un mensaje que indique la ausencia de resultados que coincidan con el filtro

Clasificación:
- Prioridad : Baja
- Severidad : Leve

Issue 13

No se informa al Usuario cuando una farmacia no posee ningún medicamento que coincida con la búsqueda

Descripción:
Los medicamentos poseen un display de imágenes a pesar de que el sistema no implementa dicha funcionalidad

Impacto:
El hecho de que no se muestren imágenes de los medicamentos evita que los usuarios pueden verificar que es el medicamento que buscan

Solución ideal:
Se descarta el sistema de imágenes / Se implementa la funcionalidad y se realiza una solicitud de carga de imágenes para los medicamentos ya ingresados

Plan de acción:
Se Cambia el componente de medicamento por uno que no posea imágenes
/
- Se realiza la implementación del sistema de imágenes
- Se explora la base de datos en búsqueda de medicamentos sin imagen
- Se implementa un sistema de edición de imágenes
- Se les indica a los usuarios empleados sobre la falta de imágenes en los medicamentos que brinda su farmacia y se les solicita su actualización

Clasificación:
- Prioridad : Baja
- Severidad : Leve

Issue 14

Hora de Transaccion desfasada

Descripción:
Cuando se realiza una compra en todo el sistema la hora se encuentra adelantada por 3 horas, incluso bajo la situación en donde el cliente y el server se encuentran en el mismo dispositivo

Impacto:
El desfasaje puede causar confusión o errores administrativos

Solución ideal:
Se define una hora estándar en función de los clientes esperados, esta hora será la hora de nuestro servidor y debera ser usada al momento de indicar la hora de la realización en las transacciones

Plan de acción:
- Encontrar la ubicación del error que causa el desfasaje

- Garantizar que la hora coincide con el servidor al momento de la transacción
- Ajustar la hora del servidor como se considere adecuado

Clasificación:
- Prioridad : Baja
- Severidad : Leve

Issue 15

Cuando el cliente llama a un servidor offline no se le informa de esto

Descripción:
El llamado a un servidor offline es respondido con un undefined que no informa de la actual inactividad del servidor.

Impacto:
Un usuario puede continuar intentando utilizar los servicios a pesar de que el servidor se encuentra inactivo sin comprender la causa de los errores

Solución ideal:
Se implementa un mensaje dedicado a los errores causados por inactividad del servidor, informando adecuadamente que este inactivo

Plan de acción:
Implementar un sistema capaz de identificar la ausencia de un servidor
Utilizar este mecanismo para alterar el mensaje de error para indicar dicha ausencia

Clasificación:
- Prioridad : Baja
- Severidad : Leve

Issues 16

a) Creación de medicamento con errores se genera una excepción general:

Descripción:
El sistema actual utiliza excepciones genéricas o errores generales para manejar problemas en lugar de excepciones específicas.

Impacto:
Esto dificulta la identificación y corrección de errores, lo que puede llevar a una mayor carga de trabajo en el mantenimiento y una menor calidad del software, y dificulta la usabilidad para el propio usuario del sistema.

Solución ideal:
La solución ideal sería revisar y refactorizar el código para reemplazar las excepciones genéricas con excepciones específicas que proporcionen información detallada sobre el error.

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta deuda técnica, como:

- Identificar todas las instancias de excepciones genéricas en el código.
- Analizar cada caso para determinar qué tipo de excepción específica debería usarse.
- Reemplazar las excepciones genéricas con excepciones específicas.
- Actualizar la documentación y las pruebas correspondientes.

Reproducción del error:
Logueado como empleado --> create drug
se genera la siguiente droga
- Code: Xa11
- Name: a
- Symptom: aa
- Quantity: 1
- Price: 2
- Prescription: mg
- Capsule

Clasificación:
- Prioridad: Media
- Severidad: Leve

Issue 17

a) Creación de request de stock con valores negativos:

Descripción:
El sistema actual permite la creación de solicitudes de stock con valores negativos, lo que puede llevar a problemas de seguimiento y control de inventario, así como a la generación de informes incorrectos.

Impacto:
La creación de solicitudes de stock con valores negativos puede resultar en un desequilibrio en el inventario y errores en el cálculo de existencias. Esto podría llevar a problemas de disponibilidad de productos y pérdida de ventas.

Solución ideal:
La solución ideal sería implementar validaciones en el sistema que impidan la creación de solicitudes de stock con valores negativos y proporcionen mensajes de error claros a los usuarios.

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta deuda técnica, como:

- Identificar las áreas del sistema donde se permite la creación de solicitudes de stock con valores negativos.
- Agregar validaciones para verificar que los valores no sean negativos antes de crear una solicitud de stock.

- Actualizar la interfaz de usuario para proporcionar retroalimentación visual y mensajes de error informativos.
- Realizar pruebas exhaustivas para garantizar que las validaciones funcionen correctamente.

Reproducción del error:
- Iniciar sesión como empleado.
- Acceder a la opción de "Crear solicitud de stock".
- Ingresar un valor negativo en el campo de cantidad via teclado.
- Continuar con la creación de la solicitud.

Clasificación:
- Prioridad: Baja
- Severidad: Menor

Issue 18

Falta de botón claro para volver atrás en la aplicación:

Descripción:
Actualmente, al ingresar al menú de la aplicación, no existe un botón claro o una opción intuitiva que permita a los usuarios volver atrás. Esto genera confusión y dificulta la navegación fluida dentro de la aplicación.

Impacto:
La falta de un botón claro para volver atrás puede llevar a una experiencia de usuario frustrante y aumentar la curva de aprendizaje para nuevos usuarios. También puede generar confusión y requerir que los usuarios realicen acciones adicionales, como hacer clic en el logo de la aplicación, para regresar a una pantalla anterior.

Solución ideal:
La solución ideal sería implementar un botón o una opción clara y fácil de encontrar que permita a los usuarios volver atrás en la aplicación de manera intuitiva.

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta preocupación, como:

- Realizar un análisis de usabilidad para identificar la ubicación óptima del botón de retroceso.
- Diseñar e implementar un botón de retroceso en una ubicación visible y coherente en todas las pantallas de la aplicación.
- Actualizar la interfaz de usuario y proporcionar indicaciones visuales para destacar la existencia y la funcionalidad del botón de retroceso.
- Realizar pruebas de usuario para garantizar que la nueva funcionalidad sea fácilmente comprensible y utilizable.

Clasificación:
- Prioridad: Media
- Severidad: Moderada

Issue 19

Discrepancia en el número de artículos solicitados durante la compra como usuario anónimo:

Descripción:
Cuando un usuario anónimo realiza una compra de varios artículos en la aplicación, se ha observado que a veces los artículos no se incluyen correctamente en la lista de compras. Esto resulta en una discrepancia en el número de artículos solicitados y los que finalmente se incluyen en la compra.

Impacto:
La discrepancia en el número de artículos solicitados puede llevar a una experiencia insatisfactoria para los usuarios, ya que no reciben todos los productos que esperaban. Esto podría generar confusión y frustración, así como pérdida de ventas y la posibilidad de recibir devoluciones.

Solución ideal:
La solución ideal sería identificar y corregir el problema que causa la discrepancia en la compra de artículos como usuario anónimo. Esto podría requerir una revisión detallada del flujo de compra y la identificación de cualquier error en el proceso.

Plan de acción:
El plan de acción podría incluir pasos específicos para abordar esta preocupación, como:

- Realizar pruebas exhaustivas para replicar y comprender el problema de la discrepancia en la compra.
- Identificar las áreas del flujo de compra donde podría producirse el error.
- Corregir cualquier error en el proceso de compra que esté causando la discrepancia.
- Realizar pruebas de usuario para verificar que el problema se ha resuelto satisfactoriamente.

Reproducción del error:
- Crear una orden anónima con varios ítems incluidos en el carrito
- Confirmar el pedido
- Ingresar como empleado
- Ingresar a View Stock Request
- Localizar la compra realizada

Clasificación:
- Prioridad: Alta
- Severidad: Critico

Retrospectiva:

Como fue explicado anteriormente a lo largo de este informe, la falta de coordinación por superposición de horarios ya sea laborales o por asistir a otras asignaturas es un denominador común en el equipo.

Debido a esto hubo 2 instancias de retrospectivas:
- Una de Ana Laura con Sebastián y otra de Ana Laura con Giuliano.
Ambas están documentadas y en el espacio de trabajo de Metro Retro, que es la herramienta que hemos utilizado en otros semestres.

Se incluyen en este repositorio el link al espacio de trabajo:

https://metroretro.io/BO5EQ6TVNWUT

Así como links a las carpetas de print de pantallas del trabajo durante la retrospectiva.
Las mismas se encuentran en la carpeta: /Entrega1/Retrospectiva

Surgen como acciones correctivas:

1) Definir días y horarios fijos para las reuniones, no son efectivas las reuniones de Stand Up asíncronas o con la herramienta de chat instantáneo, ya que se lee a destiempo y muchas veces no se entiende la necesidad del momento.
2) Debemos elegir un único Scrum Master que haga el seguimiento de todas las instancias en tiempo y forma
3) Nivelar el tiempo dedicado de cada integrante del equipo para no sobrecargar al resto. Lograr un compromiso uniforme a lo largo de toda la entrega.

Link de Video de Retrospectiva

<https://fi365-my.sharepoint.com/:v:/g/personal/am146669_fi365_ort_edu_uy/EQD5n4--4_JJuEhbFKLN7AUBQT1Y_6_xJjzaSagEkkj4OA>

https://fi365-my.sharepoint.com/:v:/g/personal/am146669_fi365_ort_edu_uy/EQD5n4--4_JJuEhbFKLN7AUBQT1Y_6_xJjzaSagEkkj4OA

Estimaciones de esfuerzo y Registro de Tareas

Se entregan en la carpeta: /Entrega1/Esfuerzo de Tareas

Allí podemos encontrar archivos excel y pdf que nos muestran el resumen de tareas desglosado por integrante y por equipo.

Los resúmenes fueron emitidos por la herramienta Clockify que fuimos utilizando.
La dedicación completa del equipo es de 37,92.

Completando un total de 19 issues que fueron informadas y listadas en el ReadMe
Los archivos de tiempo detallado también nos muestran las métricas: Cicle Time y Lead Time.