Universidad ORT Uruguay

Obligatorio

Ingeniería de Software Ágil 2

Agustina Disiot 221025

Beate Lidstrom 292178

Iván Monjardin 239850

Francisco Rossi 219401

ISA2/Obligatorio (github.com)

Rama: entrega-2

2022

Índice:

Introducción

En este informe se va a encontrar la explicación del trabajo realizado para la segunda entrega del obligatorio de Ingeniería de Ágil 2.

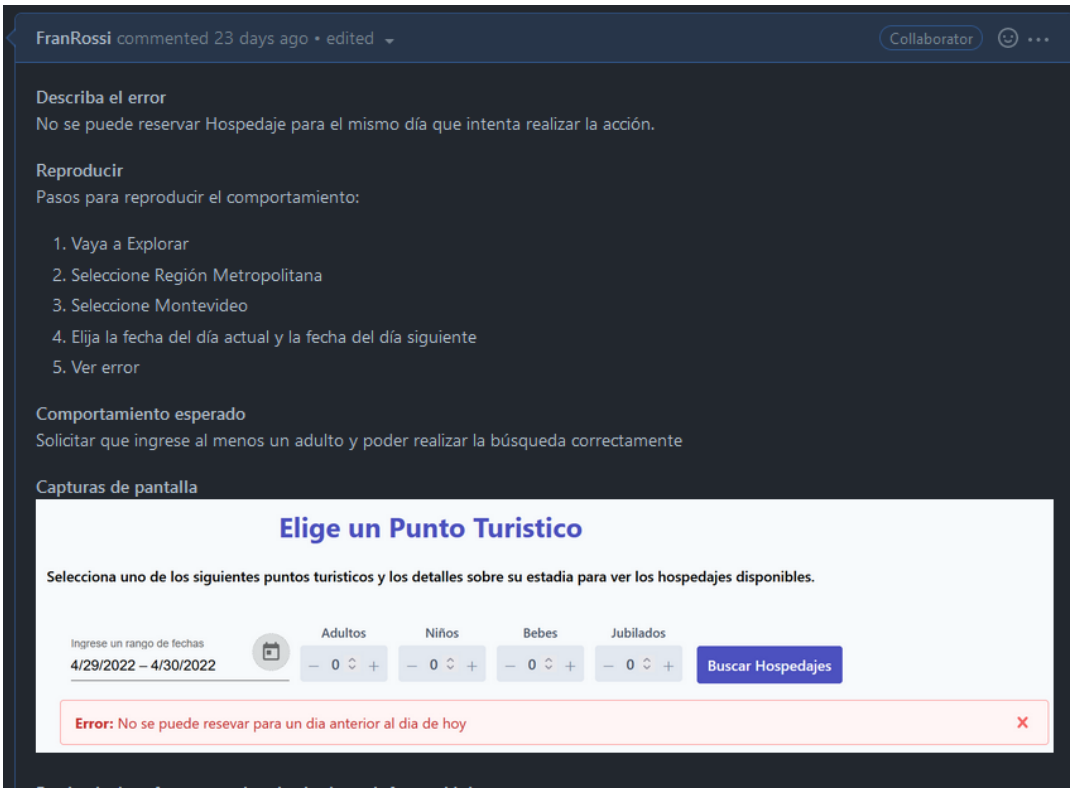
Se planificó realizar lo pedido en la letra para la segunda entrega lo cual cumplimos en hacer. En las distintas secciones se incluyen comentarios detallando los resultados, justificaciones y explicación de los procesos.

Los subtítulos con números al inicio corresponden a los puntos pedidos en la Definition of Done correspondiente.

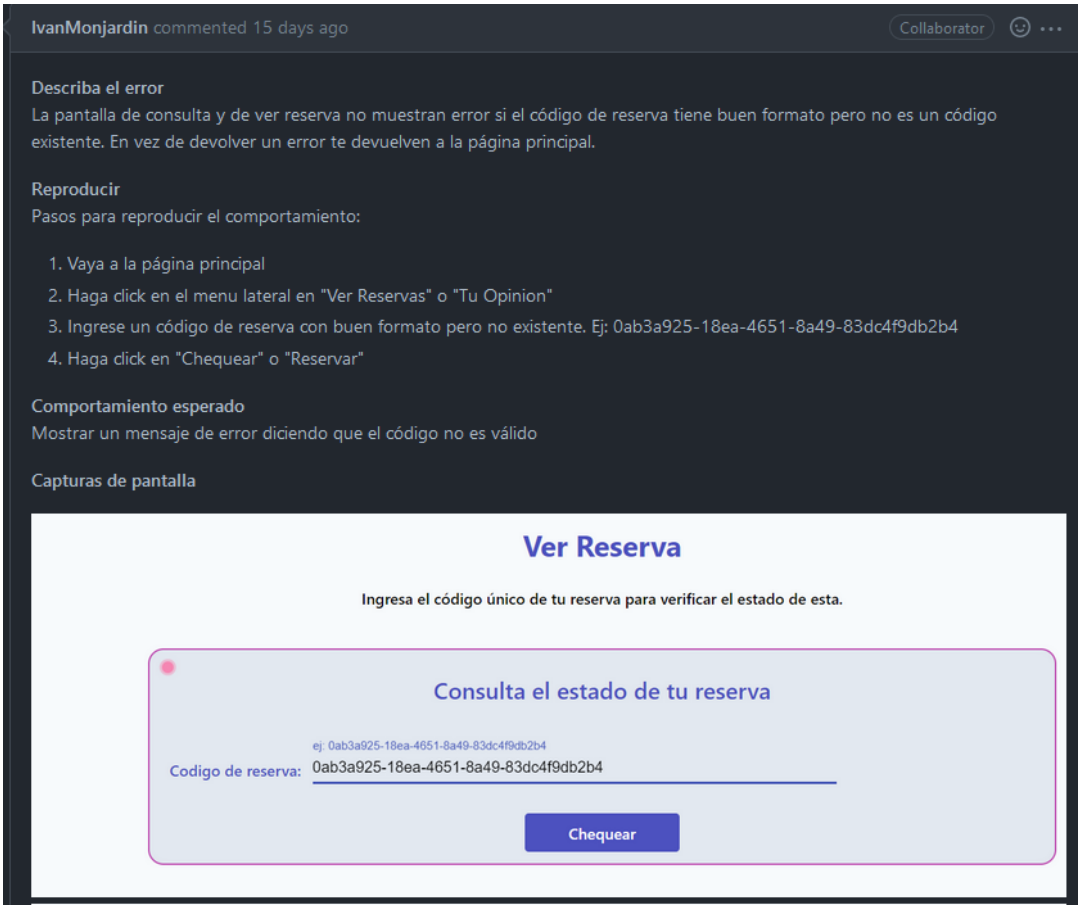
1. Sobre la reparación de bugs:

Siguiendo nuestra definición de severidad que definimos en la primera entrega, obtuvimos 2 bugs con máxima severidad (S1) que decidimos evaluar para arreglar.
En este análisis nos dimos cuenta que unos de los bugs que habíamos reportado con severidad S1, realmente no era un bug. Esto se dió por un mal entendido de que en cierta parte de la letra se indicaba cierta funcionalidad con la posibilidad de cambiar la capacidad de un hospedaje, pero en otra sección de la letra se indicaba que esto sería manejado externamente. Debido a este error ya no contábamos con 2 bugs con severidad S1, sino con 1. Para poder cumplir con la Definition of Done, evaluamos los bugs de severidad S2 y cambiamos el de mayor severidad a S1.

El primer bug se trataba del siguiente:
https://github.com/agustinadisiot/isa2-m7a-06/issues/10,
en el cuál la aplicación no permitía poder realizar una reserva para el mismo día que se utilizaba la aplicación. Esto nos pareció de suma importancia arreglarlo, porque pensando en una situación real, un usuario debería poder reservar un hospedaje para el mismo día y no por un bug impedir esta funcionalidad.



El segundo bug que se decidió arreglar fue el siguiente:
https://github.com/agustinadisiot/isa2-m7a-06/issues/29



Donde se encontró un error al chequear código de reserva y decimos que era importante repararlo, ya que puede llevar a dejar una mala impresión al usuario cuando intenté utilizar esta funcionalidad. Pero también como un punto importante de seguridad, ya que no se hacía un chequeo apropiado de lo que se ingresaba en el campo fuera totalmente correcto y existiera antes de hacer alguna acción.

2. Sobre las nuevas funcionalidades:

Se crearon las dos funcionalidades solicitadas en la Definition of Done, alta y baja de los puntos de carga.
Las restricciones de las propiedades como la cantidad de dígitos o de carácteres para el nombre, descripción, Id, dirección se tomaron como máximos. Lo que implica esto es que decidimos seguir al pie de la letra estas restricciones por lo que nuestra solución podrá almacenar como máximo unos 9999 puntos de carga.

Lecciones aprendidas y Errores

Se comenzó siguiendo el pipeline del GitHub previamente definido y realizando los criterios de aceptación en GitHub únicamente. Para la alta de puntos de carga, se utilizó TDD ya que en la primera semana no se había mencionado de que no era necesario e inicialmente no se utilizó SpecFlow ya que tampoco se había mencionado.

El error más severo que cometimos fue el no poder implementar los casos de prueba de BDD con una base de datos en memoria, por lo que al activar la acción en GitHub actions, los test de SpecFlow fallan, al intentar probarlo con la base de datos.
Esto llevó a que tampoco podamos autenticar al usuario que crea el punto de carga como administrador, ya que no supimos como hacer para realizar el login antes que la creación desde los puntos de carga. Al no definir explícitamente en la letra, decidimos interpretar que no era un requerimiento y sacamos este control de autorización para las nuevas funcionalidades.

Si bien hubo errores, durante esta entrega lo más importante que se aprendió fue la implementación con código de test de aceptación, automatización de procesos usando GitHub Actions y pusimos en práctica conceptos como BDD, TDD, principios de DevOps como las tres vías (las cuales se ven reflejadas en el tablero, los tests, la rétrospective y otros).

3. Sobre la revisión (review con el product owner)

Se realizaron 4 reviews con el product owner siendo algunos de los integrantes que no participó en algunas de las User Stories.

Proceso

Se trató de simular lo que sería una demo, mostrando la funcionalidad de la nueva feature corriendo o los bugs arreglados.
Para esta entrega se realizó una única instancia de Review la cual fue grabada y se puede ver en el siguiente link:
https://youtu.be/dATKHah60pI

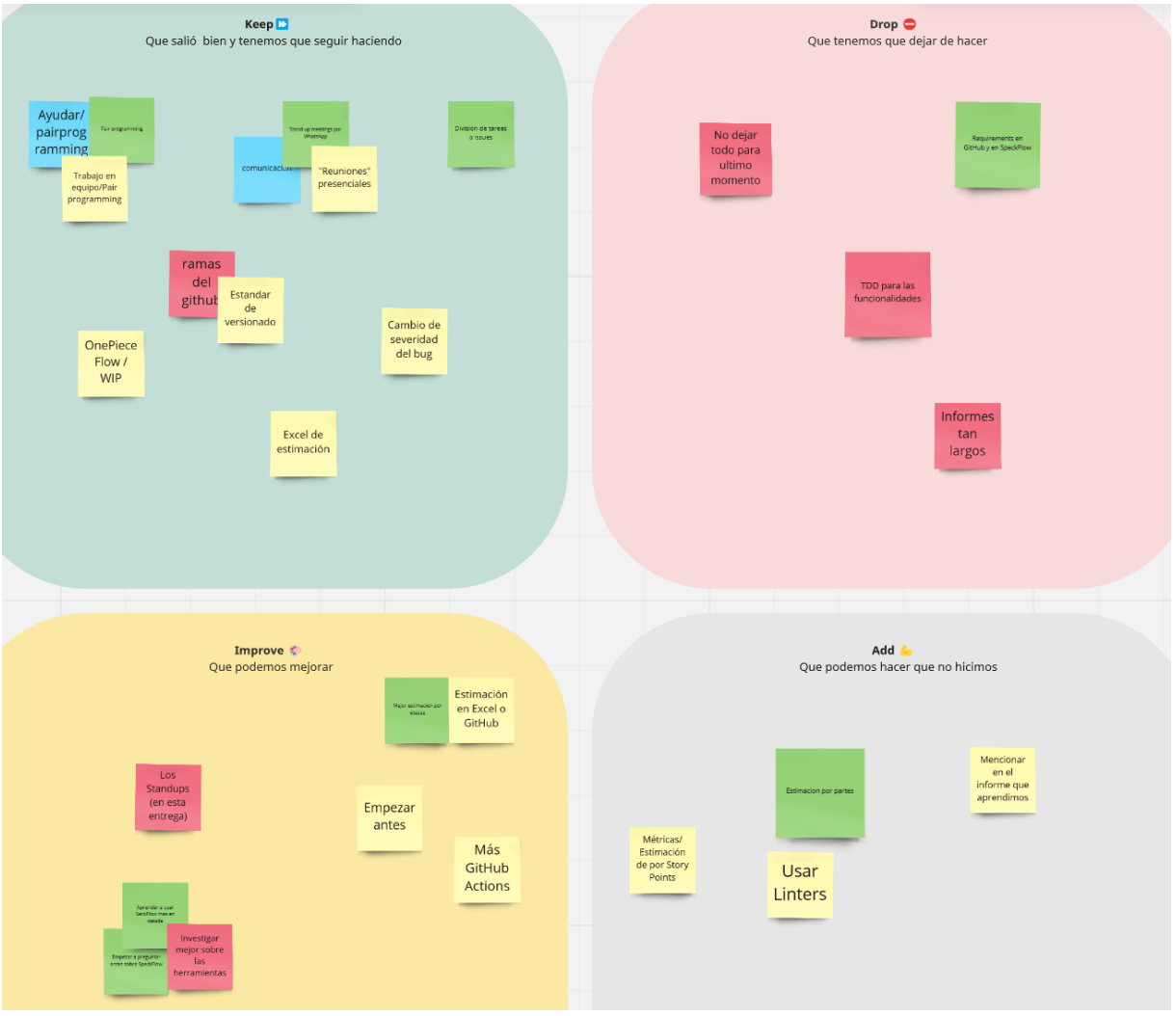
Como se explica en el vídeo, para las cuatro Reviews (dos features nuevas y dos bugs) se definió un Product Owner a alguien que no participó del desarrollo/arreglo de la issue. Cada PO aceptó la US en base a los respectivos criterios de aceptación.

Retrospectiva

Al final de las dos semanas se realizó una retrospectiva con todo el equipo. La idea fue revisar qué cosas podemos mejorar para la siguiente entrega.

La sesión fue grabada y está disponible en el siguiente link: https://www.youtube.com/watch?v=jhQD4z8BgF0

Se utilizó el método DAKI (Drop, Add, Keep, Improve) en un tablero Miro el cual podemos ver a continuación



Se discutieron varios temas en la retrospectiva como se puede observar en el video, donde cada uno propuso ideas de tópicos para discutir luego de unos minutos.

Algunos resultados a destacar:

- Continuar con la posibilidad de realizar pair programing para el que lo desea y coinciden los horarios. *
- Aspectos positivos de usar trunk-based development
- Buena comunicación, seguir haciendo stand ups por WhatsApp *
- Decisión de reducir el tamaño de los documentos en general comparado con la anterior entrega.
- Mejorar los tiempos, incluyendo empezar con más antelación.
- Investigar de antemano alguna herramienta que se proponga usar y no esperar hasta verla en clase únicamente.
- Continuar con el registro de esfuerzo en un archivo Excel, nos resulta más cómodo para todo el equipo.
- Agregar estimación por Story Points a las User Story
- Utilización de los templates definidos para User Story *
- Asignarse las User Story que se encuentren en el TODO, cuando el desarrollador tenga tiempo y disponibilidad para empezar a trabajar en ellas.
- Otros.

También nos parece importante destacar algunas acciones que se tomaron en función de la retrospectiva anterior:

- Utilizar Google Docs para los trabajar colaborativamente en los informes
- Utilizar notas de distintos colores en el Miro dependiendo de la persona
- Los puntos de la lista anterior marcados con un “*”
- Otros

Otro punto importante que queremos modificar para la siguiente entrega es el manejo de las Review. Actualmente la Review es una tarea a parte. Sin embargo para la siguiente entrega se quiere agregar al tablero y parte de la Definition Of Done de las User Stories. Antes de pasar a “Done” tiene que existir la Review.

Análisis de Métricas

La principal métrica que tomamos en cuenta y comentamos en la retrospectiva fue el Work In Progress (WIP), el cual indica cuántas tarjetas en el tablero están en el mismo momento en “Doing” haciendo referencia a cualquier columna que implique estar en proceso de trabajo para nuestro caso.
Lo que queríamos lograr y logramos fue que el WIP nunca sea mayor a 4, ya que contamos con cuatro integrantes en el equipo.

Otra métricas es el Deployment Frequency que en esta caso, como no deployeamos a producción, lo tomamos como la cantidad de veces que hacemos el “deploy” a staging. Cómo hicimos las review juntas al final, para esta entrega el deployment frequency fue de 1 cada dos semanas.

Otra métricas como Lead o Cycle time se podrán agregar cuando estimemos en Story Points.

Registro de Esfuerzo

Como se comentó se registró el esfuerzo en un Excel:

Segunda entrega

	Duración (horas)	Personas	Estimación (esfuerzo)	Esfuerzo (horasXPersona	Diferencia real - estimado
Lectura de definition of Done 2 y creación de tareas	1.5	4		6	
Alta Punto de Carga (Domain y DataAccess)	3	1	2.5	3	0.5
Alta Punto de Carga (BussinessLogic)	1.5	1	2	1.5	-0.5
Alta Punto de Carga (WebApi)	2	1	2	2	0
BDD en Spectflow Creación de Punto de carga	4	1	1.5	4	2.5
Automatización del Build en el repositorio	0.66	1	0.5	0.66	0.16
Automatización del Test en el repositorio	3	1	1	3	2
Arreglar GitHub Action Error con SpecFlow	2.5	1	0.5	2.5	2
Aprender lenguajes	2	1		2	
Instalación solución	2	1		2	
Arreglo de bug de verificacion codigo de reserva (Ivan y Agustina)	6	2	3	12	9
Arreglo de bug de fecha de reserva (Bea y Agustina)	2.5	2	2.5	5	2.5
Baja punto de Carga (Agustina)	1.5	1	2	1.5	-0.5
Retrospectiva	0.5	4		2	2
Reviews	0.25	4	0.5	1	0.5

Persona	Horas pesonales
Fran:	12.5
Agus	11.75
Iván	14.91
Bea	10.5
Ezfuerzo	49.66
Sin contar las tareas de gestión	
Tareas administración	2

Nota: Cuando hagamos el cambio de cómo hacemos la Review, se agrega el tiempo de las Review a las respectivas US.

Estándar de versionado

Se realizaron cambios al estándar de versionado en comparación a lo definido anteriormente:

- Los commits se realizan en inglés
- Cuando se va arreglar un error se crea una rama de forma: “bugfix-nombre”
- Utilización de rama staging para US que pasaron la Review - se va a incorporar al tablero en la siguiente entrega
- Las hotfixes se pueden pushear directamente a main o realizar una rama bugfix