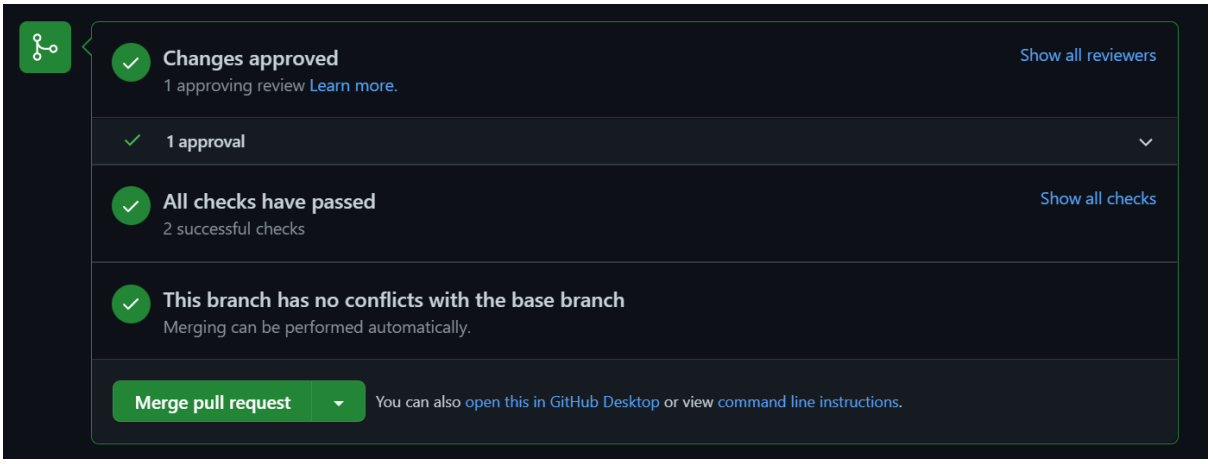
Configuración/mantenimiento del pipeline y su vínculo con el tablero

La idea para este obligatorio es seguir las prácticas de CI/CD en la mayor medida posible. En cuanto a CI si bien no estuvimos desarrollando en ramas que vivían 1 dia y haciendo commits a main una vez por dia (porque escapaba el alcance del proyecto), sí comenzamos con la automatización de la integración para llegar a la integración continua.

Cuando uno de los miembros integraba con develop, automáticamente se ejecutaba un build, se corrían tests unitarios y se corrían automáticamente también los tests de escenarios escritos con Specflow. Este fue el pipeline que construimos en la primera entrega

En esta segunda entrega comenzamos a escribir código tanto para implementar features nuevas usando user stories así como para resolver bugs.

Para lograr optimizar el flujo de desarrollo nuestro tablero sufrió grandes cambios (acomodándose a BDD). De la mano con esto comenzamos a automatizar ciertas cosas utilizando GitHub Actions y eso será lo que explicaremos a continuación.



En la imagen anterior se puede apreciar “All checks have passed. 2 successful checks” esto hace referencia a nuestro pipeline. Los dos chequeos a los que hace referencia en ese caso son a la corrida de los tests unitarios automatizada.

Lo que sucede paso a paso es que se instalan las dependencias, se hace el build, se corren los tests unitarios, se corren los tests de escenarios y se sube la cobertura a codecov.

El vínculo que hay con el tablero es que si bien no automatizamos la movida de las cards, cuando se ejecutaba automáticamente los tests unitarios la card la colocabamos en la columna con ese nombre y ahi podia volver para atrás a application implementation si es que había fallos o quedarse esperando por el code review. Una vez se hacía la review la card se podía mover hacia refactor si es que habían cambios por hacer o ir directamente a done con la review del miembro que hacía de PO.

Por último queremos cerrar mencionando los grandes beneficios que vemos con esto. Ya a nosotros nos ahorró considerable tiempo de configurar y ejecutar builds y tests siendo 3

personas implementando 2 features. Creemos que es impresionante lo útil y beneficioso que se transforma en equipos de tecnología con decenas o cientos de integrantes implementando incontables features con miles de commits. A fin de cuentas lo que queremos es ahorrar tiempo y dinero, CI nos permite ambas de una manera muy elegante.