UNIVERSIDADE DA CORUÑA

Calidad en Sistemas de Información
Calidad de Software

Code review

1. Introducción

Las revisiones de código son una parte más de las revisiones de calidad. Aplicaciones de *Source Code Management* como GitLab o GitHub permiten hacer Merge Requests (también llamadas Pull Requests) para incorporar cambios de una rama en otra. Estas peticiones permiten, entre otras opciones:

- Revisar y comentar los cambios y sugerir mejoras.
- Controlar cuando se hacen las incorporaciones y quién ha de dar las autorizaciones necesarias.
- Incorporar informes de herramientas de Integración Continua (CI) para comprobar que los cambios pasan las pruebas necesarias dentro de la misma petición de cambio.
- Automatizar procesos como el aplanar todos los commits del cambio en un solo commit (*squash*) o el borrar la rama de origen cuando se complete la incorporación de los cambios en la rama objetivo.

2. Objetivos

El objetivo del trabajo es usar `git` y el flujo de trabajo git-flow para implementar nuevas funcionalidades en una aplicación. Se ha de familiarizar también con el uso de GitHub y se han de poner en práctica los conceptos de revisión de código durante la realización de la implementación.

Para la realización de la práctica se usará GitHub. La cuenta que se use para la realización de la práctica ha de tener el correo electrónico del alumno de la UDC.

Para la creación de los repositorios de los grupos se hará uso de GitHub Classroom. Se accederá a la tarea a través de un enlace que se proporcionará en el Moodle de la asignatura. Uno de los componentes del grupo ha de acceder al enlace y después de seleccionar su login procederá a la creación de un nuevo grupo. Una vez se ha realizado este paso, el resto de los componentes accederán al enlace y, después de seleccionar sus respectivos logins, han de unirse al grupo creado por su compañero.

Una vez finalizado este paso, todos los componentes del grupo tendrán acceso a un repositorio privado en GitHub, que habrá sido inicializado con un proyecto base con el que empezar a trabajar.

2.1. Proyecto base

Este proyecto en un ejemplo sencillo de un servicio REST que expone los siguientes endpoints:

- `/user/all`
- `/user/<id>`
- `/user/new`
- `/user/search`

El proyecto está implementado usando el Spring Framework, y usa la base de datos H2 para hacer persistencia de los datos (la configuración inicial usa una base de datos en memoria que se pierde al cerrar la aplicación)

Lo primero que debéis hacer es familiarizaros con el proyecto y como está implementado. Se ha de mirar también la documentación de Spring y hacer alguno de los tutoriales como pueden ser

Building a RESTful Web Service como ejemplo de un sencillo servicio REST (sin persistencia), o Accessing Data with JPA sobre como implementar persistencia en Spring. El proyecto incluye un pequeño `README` y un archivo `HELP.md`, que ha sido generado por el propio Spring Initializer.

Además, es recomendable hacer uso de herramientas para acceder a servicios REST, como por ejemplo Postman, para acceder al servicio y comprobar su funcionamiento.

2.2. Funcionalidad a implementar

Una empresa de venta online quiere crear una aplicación para llevar la gestión de parte de su negocio. Esta aplicación de implementará haciendo uso de servicios REST para la realización de las distintas operaciones. En esta sección se detallan los casos de usos que hay que implementar, divididos por cada de las entidades que han de implementarse:

Client

Se ha de de dar soporte para almacenar información de clientes, permitiendo añadir nuevos clientes, modificar algún cliente ya existente o recuperar la información de un cliente. Para cada cliente hay que almacenar la información referente al mismo, como puede ser el nombre y apellidos, dirección, métodos de pago, etc, necesarias para la gestión de los mismos. En particular:

- Ha de ser posible registrar un nuevo cliente en el sistema.
- Ha de ser posible modificar la información de algún cliente ya existente.
- Ha de ser posible recuperar la información de algún cliente en particular.

Product

Ha de ser posible guardar la información de los productos disponibles para su venta. Se ha de poner conocer el nombre del producto, su precio actual, el stock disponible y otra información relevante. Ha de ser posible realizar la siguiente operaciones:

- Ha de ser posible añadir un nuevo producto al catálogo.
- Ha de ser posible modificar la información de un producto existente.
- Ha de ser posible acceder a la información de un producto en particular.
- Ha de ser posible modificar el stock disponible de un producto, añadiendo o restando unidades.

Sale

Ha de guardarse la información de las ventas realizadas, que incluye que producto se ha vendido, a que cliente se ha vendido, cuantas unidades se han vendido y el precio de venta, entre otra información relevante. En particular ha de ser posible:

- Ha de ser posible registrar una nueva venta.
- Ha de ser posible recuperar la información de una venta en particular.
- Ha de ser posible recuperar todas la ventas que se han realizado de un producto.
- Ha de ser posible recuperar todas la compras realizadas pon un cliente en particular.

Employee

Se ha de dar soporte para almacenar la informacion de los empleados de la empresa. Se ha de guardar la información del empleado, como puede ser su nombre, dirección, edad, salario, número de la seguridad social, departamento al que pertenece o quien es su supervisor, o cualquier otra información que se considere relevante. Ha de ser posible:

- Ha de ser posible dar de alta a un nuevo empleado en la empresa.
- Ha de ser posible modificar los datos de un empleado.

- Ha de ser posible recuperar la informacion de un empleado en particular.
- Ha de ser posible recuperar la lista de todos los empleados de un departamento.

2.3. Metodología

Se ha de seguir el flujo de trabajo marcado por git-flow, donde en `main` solo hay código que está en producción, realizándose el trabajo sobre una rama `develop`, de la que parten *feature branches* para cada una de las funcionalidades nuevas.

Antes de incorporar cada una de las ramas de funcionalidad a la rama `develop` se ha de crear una *Merge Request* y se ha de realizar una correcta revisión del código, no incorporando los cambios hasta que todos los problemas hayan sido abordados y se hayan obtenido las autorizaciones pertinentes. Esta es la parte más importante de la práctica, mas allá de que las funcionalidades estén correctamente implementadas o no, pero deben existir revisiones de código.

Cuando todas las funcionalidades estén implementadas e incorporadas se han de incorporar todos los cambios a la rama `main`, señalizando el lanzamiento de la nueva versión en producción. Para facilitar la evaluación se hará *merge* de las ramas sin hacer *fast-forward*, no se ha de aplanar la historia (no se ha de hacer *squash* de los commits) y no se han de borrar las ramas de origen de los cambios.

2.4. Evaluación

La aplicación debe implementar la funcionalidad requerida, además de la calidad del código, en base a que se haya seguido algún estándar de codificación o cada parte del código tiene un formato distinto, si el código es legible y está bien comentado, etc. Es de suponer que si se realiza correctamente el proceso de revisión del código de una manera correcta el resultado ha de ser el adecuado, tanto en funcionalidad como en calidad del código.

Mas allá de las funcionalidades, el mayor peso de la evaluación recaerá en que se haya seguido correctamente la metodología *git-flow* y que, sobre todo, que los procesos de revisión del código se hayan efectuado de manera adecuada.

Para la creación de grupos, se proporciona el enlace al Github Classrooms de la asignatura. **Los grupos pueden ser de 3, 4 o 5 personas. Un primer integrante creará el grupo en Github Classrooms, y el resto se unirán a ese grupo dándole a "Join Team".**

FORMA DE ENTREGA: se ha de enviar un correo a anxo.pvila@udc.es indicando los componentes del grupo (nombres y lógins) y el nombre del grupo.

FECHA DE ENTREGA: Domingo 11 de Mayo.
Se evaluará el contenido del repositorio en esta fecha, aunque se haya enviado el correo con la información con anterioridad.