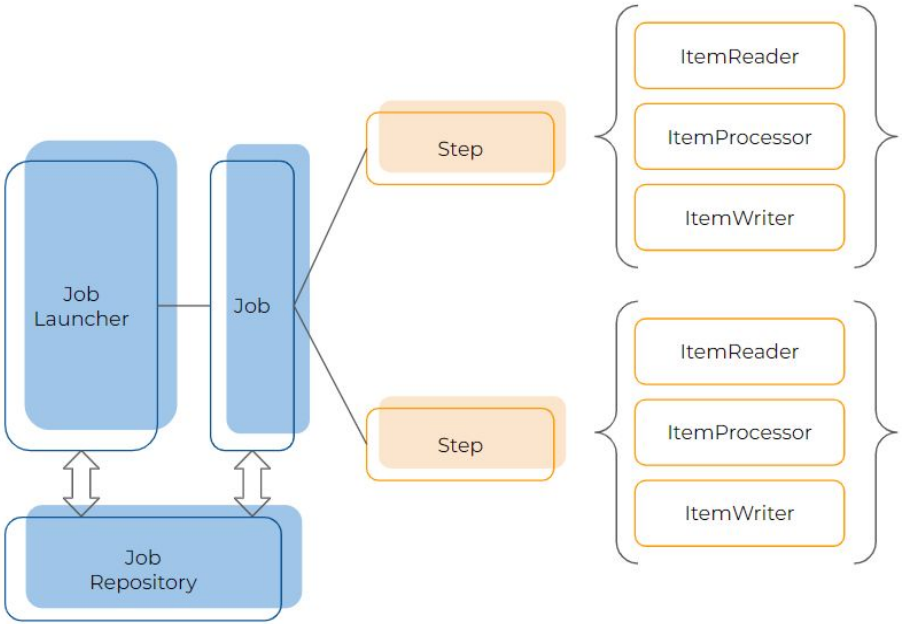
Proyecto Final _ Ana Laura Vázquez Solache

1. Diagrama las partes básicas de una implementación con Spring Batch

El significado literal de **Batch es "Lote"** al español, lo cual nos da una pista de la funcionalidad de este framework, siendo este el **procesamiento de gran cantidad de datos** "por lotes".
Posee también herramientas para **monitorizar** estos procesos, disponer de logs, configuraciones, transacciones, estadísticas, alertas, etc.
Así mismo este posee los siguientes componentes.

Job Repository
La información de los procesamientos queda **almacenada en un repositorio persistente o en memoria.**
Principalmente se utiliza para trabajos de escritura(valores de salida) pero también para comprobar si se ha procesado un fichero previamente, errores,parámetros de algún job, etc.

Job Launcher es el encargado del trigger o detonador de los trabajos, el **job** su significado literal en español es "trabajo", este es un bloque de trabajo que puede estar compuesto por varios **steps**, una vez terminados todos los "pasos" se considera terminado un "trabajo"



Un JOB puede tener muchos pasos, (Step) y estos a su vez deben tener un **Item Reader**(lectura) de entrada y un **Item Writer**(escritura) de salida. Y el **ItemProcessor** es qué tratamiento le quieres dar a esa información. (es opcional ponerlo o no el programa corre y el IDE no marca error)

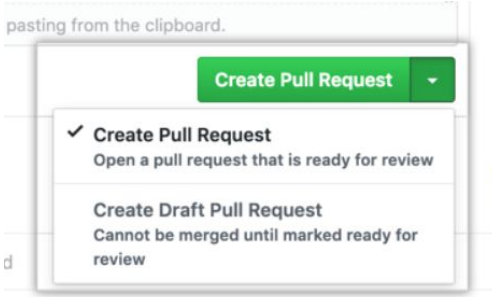
Para el procesamiento por lotes se define un valor denominado **Chunk** en el cual especificamos la cantidad en la que queremos que procese la información "el tamaño del lote".

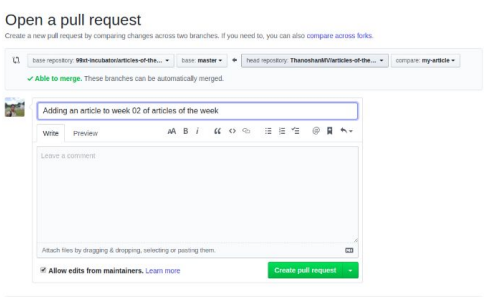
Como buena práctica se recomienda realizar pruebas de "stress" con los datos más realistas posibles.

2. Explicación de los comandos de git.(Pull Request, Fork, Rebase, Stash, Clean, Cherry Pick)

Pull Request Al momento de hacer modificar un código, y utilizar un controlador de versiones Git, y al enviar este código a un sitio web como lo es GItHub, existe este comando a través del cual podemos contribuir a un proyecto grupal, en el cual solicitamos fusionar nuestra rama con la rama main o con el proyecto base. Es como preguntarle al proyecto ¿Puedes tomar mis cambios por favor?

En las imágenes se muestra como se lleva a cabo a través de GItHub, en el cual haciendo el pull request, podemos dejar un mensaje con nuestras modificaciones y estas pueden ser comparadas con el código original.

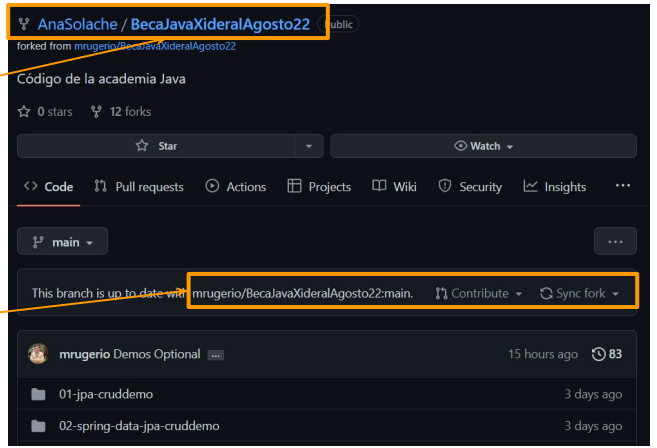




Fork es crear una copia del proyecto original a través del cual también podemos acceder a sus últimos commits, o modificaciones y experimentar libremente en el código sin miedo a editar el proyecto original. (Posteriormente si queremos sugerir nuestros cambios podemos llevar a cabo un **Pull Request.**

Mi repositorio con el “fork” o copia del proyecto original del instructor.

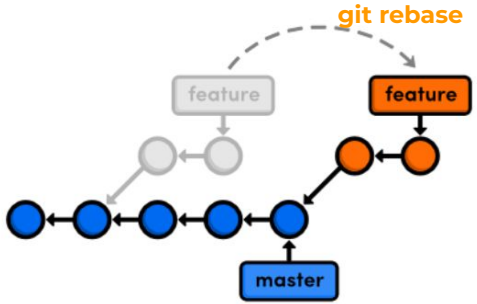
Mantenemos actualizado el repositorio de GitHub con el proyecto original junto con sus modificaciones recientes.



git rebase se parece un poco al merge, sin embargo la principal diferencia radica en que se modifica la historia de Git, así como puede generar información duplicada. Es recomendable solo cuando se está trabajando de forma individual en un proyecto, y para presentar el avance de este en una misma historia lineal.
Pero es peligroso para los trabajos en equipo, ya que se pierde la historia y se combina toda la información siendo difícil identificar puntos claves del proyecto, regresar a ciertas versiones.

git stash este comando almacena temporalmente los cambios que hayas efectuado para regresar a estas modificaciones después. Ocurre principalmente cuando la prioridad del proyecto cambia pero no quieres perder los avances que ya llevas, y no se encuentran completos como para hacer un commit de ellos, el comando es **git stash**

git stash pop este se regresa a la versión en la que estabas trabajando en el momento en que hiciste el stash, y con un **git stash apply** se fusiona con el trabajo que ya estás trabajando actualmente es útil para posteriormente hacer merges sin conflicto. Finalmente para poder hacer un stash es importante que estos se hayan añadido en el git local con un **git add** , y para añadir archivos ignorados con el archivo gitignore, incluir en el comando **git stash -u**




```
$ git status
On branch main
Changes to be committed:

    new file:   style.css

Changes not staged for commit:

    modified:   index.html

$ git stash
Saved working directory and index state WIP
HEAD is now at 5002d47 our new homepage

$ git status
On branch main
nothing to commit, working tree clean
```

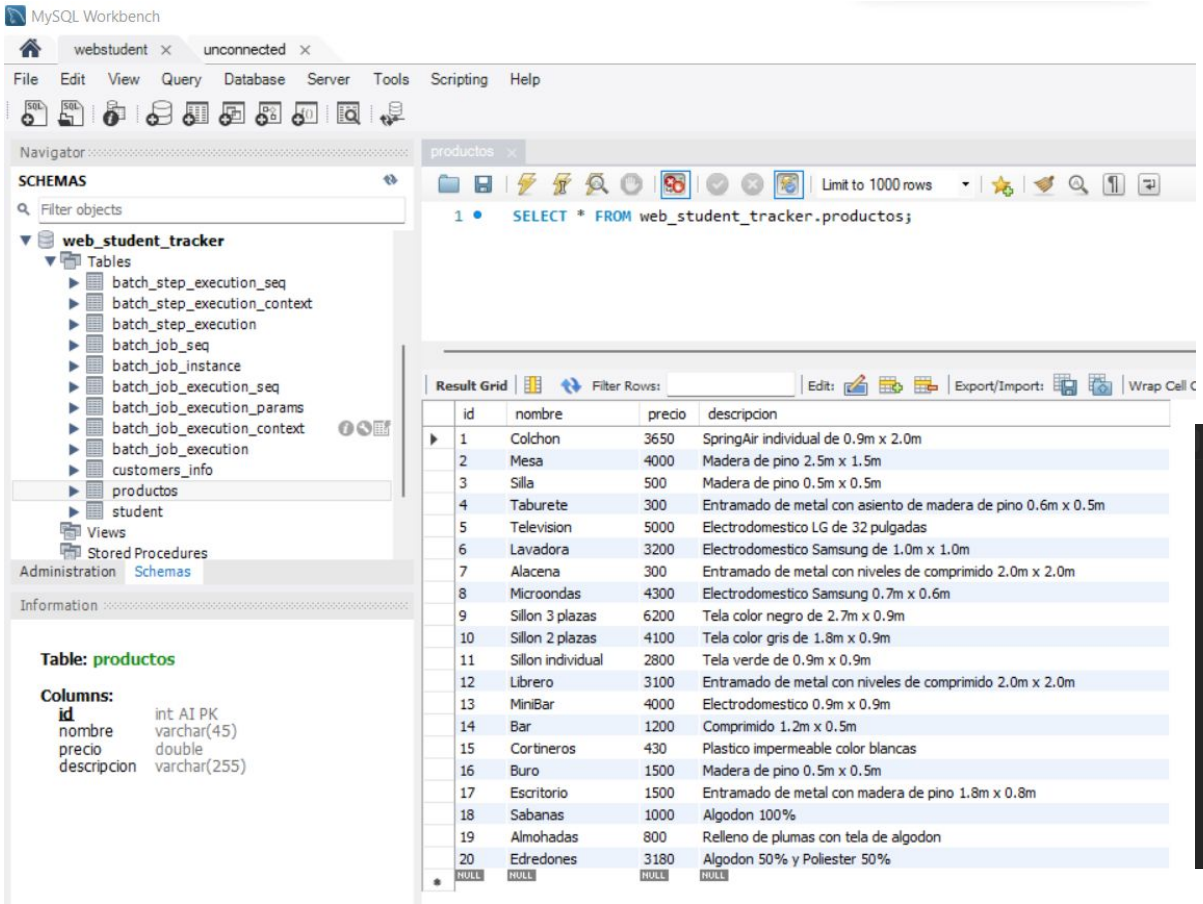
git stash

git clean está pensado para remover documentos que no están siendo usados desde el working directory, los que no han sido enviados al repositorio en commits, y que se encuentran incluidos en el gitignore.
En la linea de comandos se invoca con **git clean.**

Con **git clean -df** borrará archivos y carpetas sin seguimiento, y con **git clean -dx** -force se borrarán archivos ignorados por gitignore.
git clean -fxd se utiliza para borrar desde terminal cambios a los que aún no se les ha hecho commit, pero no se recomienda porque es el equivalente a hacer un **git reset -hard**

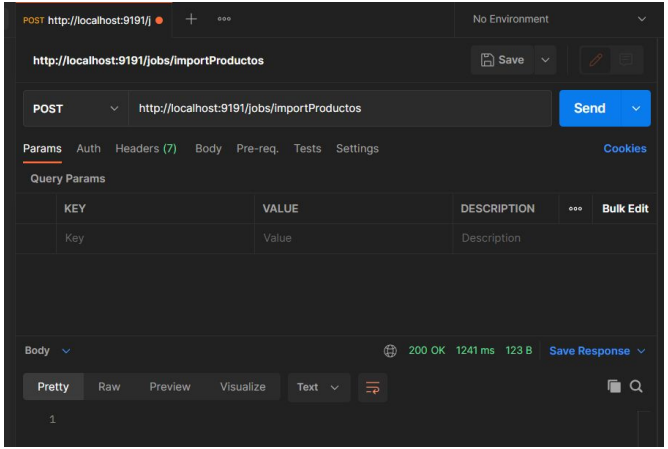
git cherry Pick es elegir los commits de una rama y aplicarla a otra, es útil para deshacer cambios.
git cherry pick “nombre del commit” es recomendable solo en algunos casos ya que puede generar duplicados. se pueden utilizar también **git cherry pick --no-comit** (para que no genere el commit automático en la historia y solo añada las modificaciones)

3. Escribir un código en el que se utilice el framework de Spring Batch



Para el proyecto final se llevó a cabo el procesamiento de datos de un archivo CSV Comma Separated Values,el cual utilizó como Jobrepository a MySQL, demostrando en la imagen la funcionalidad de framework con las tablas generadas por el framework para monitorizar los procesos así como el proceso concretado en la tabla de productos.

Esto se concretó al llevar a cabo el trigger a través de la petición POST con la ruta URL especificada en el controller.



4. Llevar a cabo un Servicio REST como interfaz gráfica que se alimente de nuestra API REST

En la primera imagen podemos ver el código llevado a cabo a través del IDE de Eclipse, y el framework de Spring Batch.
En este cabe resaltar la consola que indica que han sido insertados los productos dentro del job repository.

Este alimentó nuestra base de datos de la cual estamos consumiendo en nuestra API REST, que llevamos a cabo con el framework de Spring Boot, este código también se incluyó en el repositorio del proyecto final, con las modificaciones sobre el entity, utilizando Lombok para optimizar las líneas de código y generar a través de esta librería los getters & setters, así como constructores, entre otros con las notaciones de @Data @NoArgsConstructor @AllArgsConstructor.

En la segunda imagen podemos ver el Servicio Rest que se llevó a cabo con JSP y CSS para consumir nuestro backend que se llevó a cabo con Spring Boot, que gracias al procesamiento de la información con Spring Batch tenemos en nuestra base de datos una lista de 20 productos.