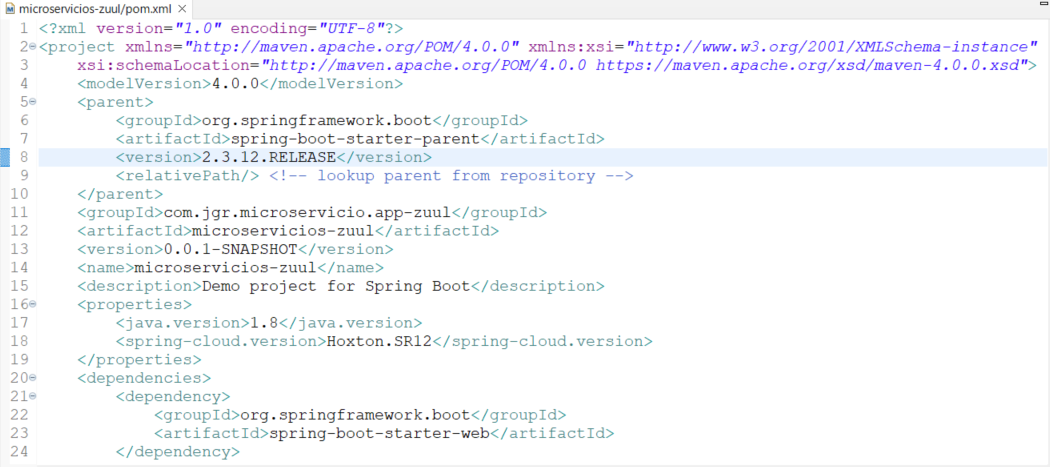
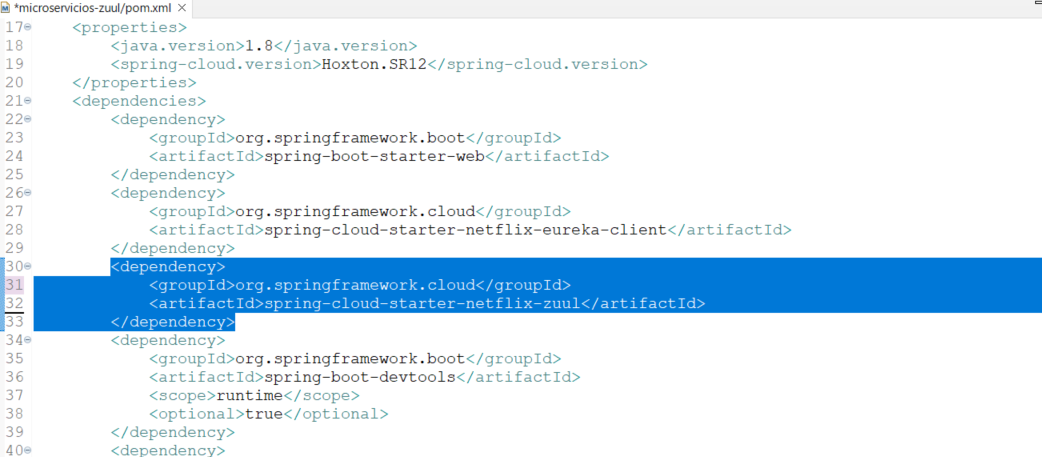
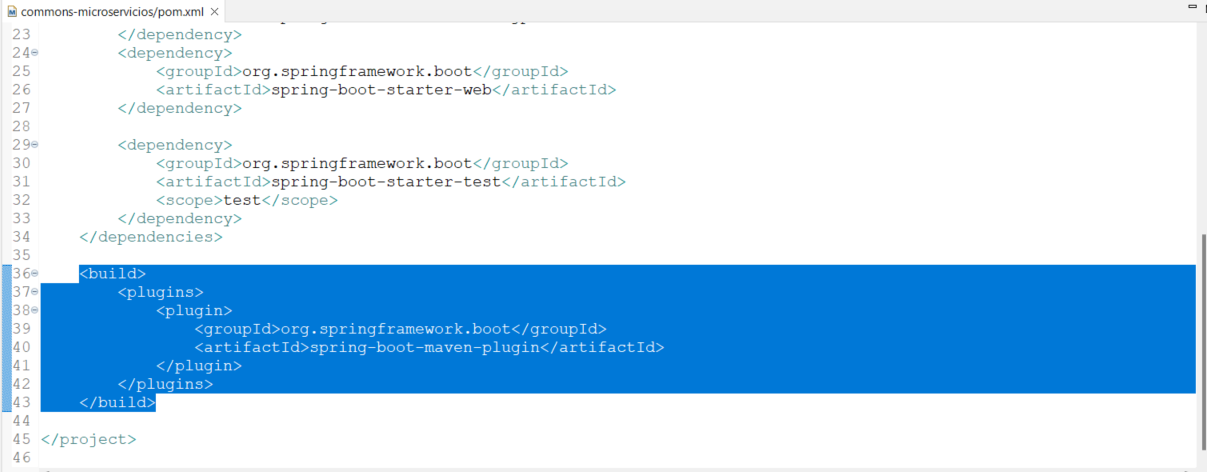
1→Creamos microservicio usuarios

2→Creamos el servidor eureka y los relacionamos en el properties de ambos

3→Creamos el gateway con Zuul, aquí hay que tener cuidado con el pom,cambiamos la version de spring en el pom de zuul:

y luego añadimos en el mismo pom del zuul la dependencia:

4→Creamos un microservicio generico(el commons-microservicios) , le quitamos el starter en el pom del generico para que no se inicie:

esto lo hacemos para poder reutilizarlo en el resto de microservicios. Como no vamos a inyectarlo,sino a heredar de el le quitamos el @Service.

Lo hacemos lo mas generico posible,

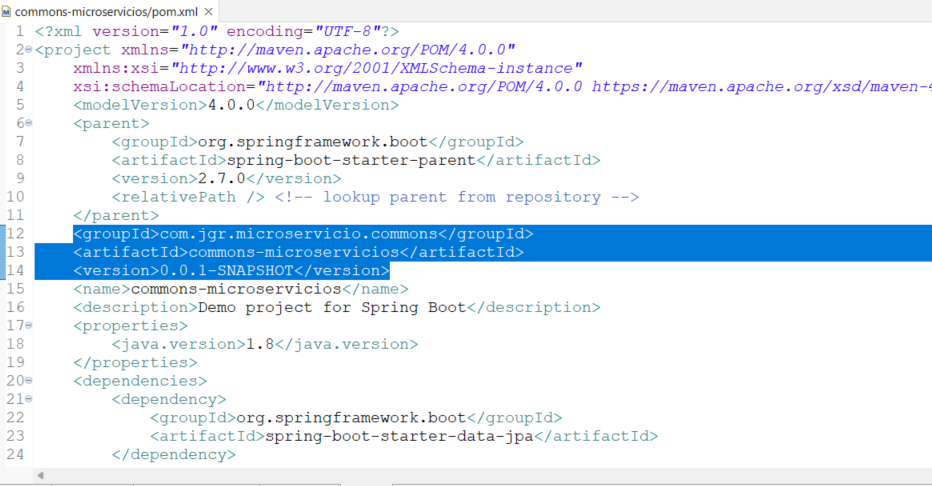
**public** **class** CommonServiceImpl<E,R **extends** CrudRepository<E,Long>> **implements** ICommonService<E>{

como parametros de entrada tiene E, que indica que es generico.

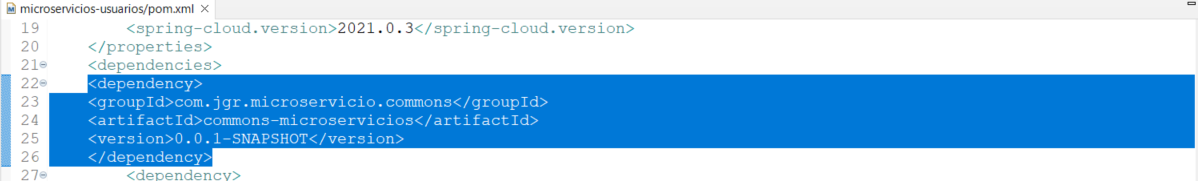
R es el nombre del repositorio que le vamos a pasar. Extiende de CrudRepository y como parametros le pasamos el nombre de la entidad “E” y el tipo de dato que tiene la clave, en este caso un Long. El IcommonService es el nombre de la interfaz que en este caso se llama CommonService,

5→modificamos el pom de microservicios-usuarios para que implemente el service de commons, en vez del suyo propio.

Esto es lo que copiamos del pom de commons:



Y lo incluimos como dependencia en el pom de usuarios:

Modificamos el IAlumnoService para que extienda de IcommonService

5-Creamos un controlador generico, para ello lo creamos en common-microservicios. No ponemos Restcontroller porque se va a heredar. Dejamos los metodos comunes,que son todos menos actualizar porque es el unico que tiene propiedades que no son comunes.

Como parametros de entrada tiene la Entidad E y el Servicio S, que a su vez tiene como parametro el Servicio S.

//le entra una entidad E y un servicio S, este ultimo hereda de ICommonService,que como es generico tambien recibe una

//entidad como parametro de entrada

**public** **class** CommonController<E,S **extends** ICommonService<E>> {

//IMPORTANTE QUE SEA PROTECTED,NO PRIVATE PARA QUE PODAMOS USARLO EN LOS QUE HEREDAN ESTA CLASE GENERICA

@Autowired

**protected** S service;

En el controlador de Alumno indicamos que hereda de CommonController y le pasamos el tipo,que es alumno y el servicio que es IalumnoService, que a su vez hereda de CommonService.

Como repositorio vamos a utilizar el que tiene CommonController, que se llama **protected** S service; que es protegido y no privado para que podamos usarlo

**public** **class** AlumnoController **extends** CommonController<Alumno,IAlumnoService>{