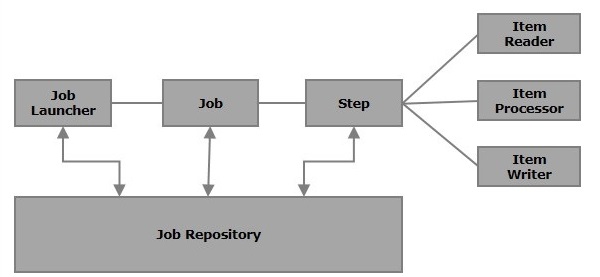
This is a spring boot application to demonstrate how to import the data from file via the spring batch and stored into DB. Able to be query via Restful API. The results are paginated.

The flow for the process

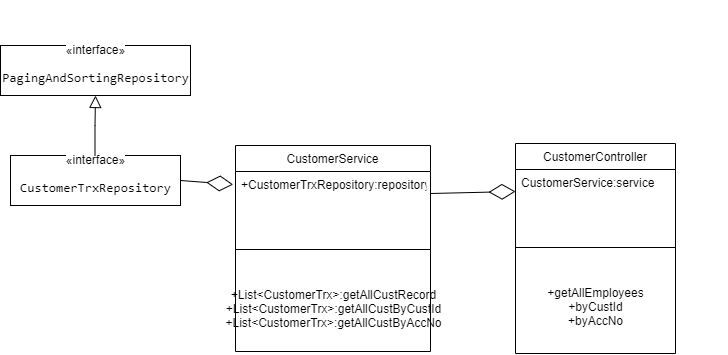
Graphical user interface

Description automatically generated

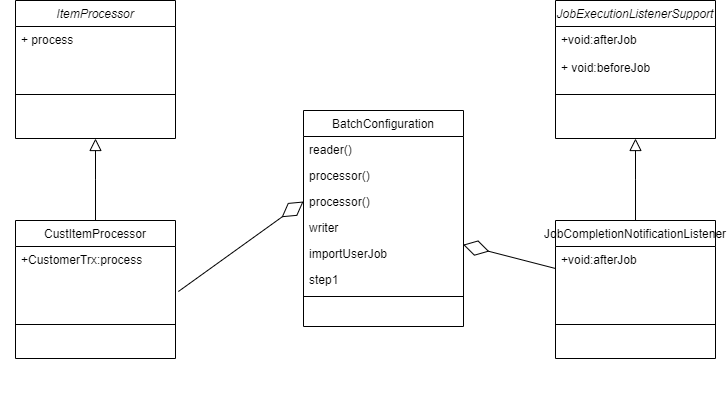
The standard component in spring batch



Basic class diagram for spring batch



Class diagram for rest component



**Restful API**:

This application provide 2 type of rest api, One is define via

@RepositoryRestResource, another is via @RestController

@RepositoryRestResource will create a HATEOAS service with spring JPA. HATEOAS provide pagination and search function.

@RestController will not create a HATEOAS,pagination and search are done in repository object.

**Rest API for @RepositoryRestResource**

1.HATEOS top level service

<http://localhost:8080/>

1. To see all customer records

[http://localhost:8080/custTrx{?page,size,sort}](http://localhost:8080/custTrx%7b?page,size,sort%7d)"

1. Info for search option

<http://localhost:8080/custTrx/search>

1. To search by customer id

[http://localhost:8080/custTrx/search/findByCustId{?custId,page,size,sort}](http://localhost:8080/custTrx/search/findByCustId%7b?custId,page,size,sort%7d)

1. To search by account number

[http://localhost:8080/custTrx/search/findByAccNumber{?accNumber,page,size,sort}](http://localhost:8080/custTrx/search/findByAccNumber%7b?accNumber,page,size,sort%7d)

**Rest API for @RestController**

Retrieved all customer

[http://localhost:8080/customers/{?page,size,sort}](http://localhost:8080/customers/%7b?page,size,sort%7d)

Search by customer id

[http://localhost:8080/customers/custId{?custId, page,size,sort}](http://localhost:8080/customers/%7b?page,size,sort%7d)

Search by account number

[http://localhost:8080/customers/accNo{?accNumber, page,size,sort}](http://localhost:8080/customers/%7b?page,size,sort%7d)

**Additional limitation on this design**

If the spring batch process was running on a monolith system with intense usage of database to read and write data. It might cause the performance issues and deadlock if the same table was access by spring batch and application at the same time.

**Suggest solution:**

Microservice architecture with independence database bundle with the microservices could reduce the burden of the database

Implement CQRS/ES design pattern could help to improve the scalability and performance of the application. The downside is the complexity of the design pattern, and might not suitable for every use case.

Graphical user interface, application

Description automatically generated