UserInput => pageSize(3)=> No of records to be displayed in each page

parameter => pageNo([1],[2],[3],....)=> Service layer would generate
                pageCount(In how many pages the records should be displayed)=>
Service layer would generate

Sequence flow
----
                 scope(session)
Controller =======pageSize=========> Service ====pageSize=======> DAO =========>
Database
   |
   |scope(request)
   | pageNo from link
   | pagesCount
   | listDTO
   |
 report.jsp

code for generating hyperlinks

```
<c:if test="${pageNo>1 }">
      <b><a href='./controller?pageNo=${pageNo-1}&s1=link'>previous</a>
  </b>
</c:if>

<c:forEach var='i' begin='1' end='${pagesCount}' step="1">
      <b><a href='./controller?pageNo=${i}&s1=link'>[${i}]</a>   </b>
</c:forEach>

<c:if test="${pageNo<pagesCount }">
      <b> <a href='./controller?pageNo=${pageNo+1}&s1=link'>next</a>
  </b>
</c:if>
```


Hibernate ReverseEngineering

      Entity class <====== ORM <====== Table already available

To do ReverseEngineering we use Hiberante tools from jbosss.

To install plugin we need to do the following steps
      a. Go to help menu
      b. Go to eclipse market place
      c. In the search box, type as jboss plugin.
      d. install jboss tools 4.2.6 plugin

Steps to perform ReverseEnginnering in eclipse
----
1. Change the perspective from java environment to hibernate
            Open perspective, choose hibernate and click on open

2. Select the project
            Click on file, new option
                  a. choose hibernate.cfg.xml file, provide suitable information
                  b. choose hibernate console configuration and provide suitable
information.

c. choose hiberante reverse engineering file to choose the table for which model should be should generated

3. Now click on run configuration of hibernate
    a. choose hibernate configuration and supply details like
        1. choose the console configuration name
        2. choose the output directory(project name till src folder)
        3. click on check box (Reverse engineer from jdbc connection)
        4. write the package name where the model should be generated(in.ineuron.model)
        5. choose revenge.xml file which is generated in the previous steps.
        6. click on run option.

Advanced ORMapping

There are multiple mismatches b/w Java and RDBMS like, we can keep entity classes in composition or inheritance, but we can't keep table names in inheritance or composition,but we need to map entity classes of composition or inheritance with 1 or more db table by taking the support of "OR" mapping
    a. Component mapping[Entity classes are in composition]
    b. Inheritance mapping[Entity classes are in inheritance]
    c. Collection mapping[Entity classes are having collection with mapping]
    d. Association mapping[Entity classes are in relation like 1---*,*---1,1----1,*----*]

Component Mapping
---
1. The property that is pointing to single column in db table is called "Simple property".
2. The property that is pointing to multiple columns of the db table having sub properties is called "Component property".
3. Component Mapping in java side is represented as "HAS-A" relationship.

eg#1.
```
package in.ineuron.model;

import javax.persistence.Embedded;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "student")
public class StudentInfo {

	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private Integer sid;
	private String sname;
	private Integer sage;
```

```
      @Embedded
      private Address address;

      //setters,getters,toString
}

Address.java
------------
package in.ineuron.model;
import javax.persistence.Embeddable;

@Embeddable
public class Address {
      private String countryName;
      private String stateName;
      private String cityName;

      //setters,getters and toString
}
output
create table student (
       sid integer not null auto_increment,
        cityName varchar(255),
        countryName varchar(255),
        stateName varchar(255),
        sage integer,
        sname varchar(255),
        primary key (sid)
    ) engine=InnoDB
```