

```
1
2 package acme.features.manager.assignment;
3
4 import java.util.Collection;
5 import java.util.HashSet;
6 import java.util.Set;
7 import java.util.stream.Collectors;
8
9 import org.springframework.beans.factory.annotation.Autowired;
10 import org.springframework.stereotype.Service;
11
12 import acme.client.data.models.Dataset;
13 import acme.client.services.AbstractService;
14 import acme.entities.project.Assignment;
15 import acme.features.manager.project.ManagerProjectRepository;
16 import acme.roles.Manager;
17
18 @Service
19 public class ManagerAssignmentListService extends AbstractService<Manager,
    Assignment> {
20
21     // Internal state
22     -----
23
24     @Autowired
25     private ManagerProjectRepository repository;
26
27     // AbstractService interface
28     -----
29
30     @Override
31     public void authorise() {
32         boolean status;
33         int masterId;
34
35         masterId = super.getRequest().getData("masterId", int.class);
36         Manager manager = this.repository.findManagerByProjectId(masterId);
37         status = super.getRequest().getPrincipal().getActiveRoleId() ==
38             manager.getId();
39
40         super.getResponse().setAuthorised(status);
41     }
42
43     @Override
44     public void load() {
45         int id = super.getRequest().getData("masterId", int.class);
46         Collection<Assignment> objects =
47             this.repository.findAllAssignmentsOfAProjectById(id).stream().distinct
48             ().collect(Collectors.toList());
```

```
45     objects = this.deleteDuplicated(objects);
46
47     super.getResponse().addGlobal("masterId", id);
48     super.getBuffer().addData(objects);
49 }
50
51 @Override
52 public void unbind(final Assignment assignment) {
53
54     assert assignment != null;
55
56     Dataset dataset;
57
58     dataset = super.unbind(assignment, "project", "userStory");
59
60     dataset.put("projectTitle", assignment.getProject().getTitle());
61     dataset.put("userStoryTitle", assignment.getUserStory().getTitle());
62
63     int masterId = super.getRequest().getData("masterId", int.class);
64
65     super.getResponse().addGlobal("masterId", masterId);
66     super.getResponse().addData(dataset);
67
68 }
69
70 private Collection<Assignment> deleteDuplicated(final
Collection<Assignment> objects) {
71     Set<String> uniquePairs = new HashSet<>();
72     return objects.stream().filter(assignment -> {
73         String pair = assignment.getProject() + "-" +
assignment.getUserStory();
74         if (uniquePairs.contains(pair))
75             return false;
76         else {
77             uniquePairs.add(pair);
78             return true;
79         }
80     }).collect(Collectors.toList());
81 }
82
83 }
84
```