```java

package acme.features.manager.assignment;

import java.util.Collection;
import java.util.HashSet;
import java.util.Set;
import java.util.stream.Collectors;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import acme.client.data.models.Dataset;
import acme.client.services.AbstractService;
import acme.entities.project.Assignment;
import acme.features.manager.project.ManagerProjectRepository;
import acme.roles.Manager;

@Service
public class ManagerAssignmentListService extends AbstractService<Manager, Assignment> {

    // Internal state ---------------------------------------------------------

    @Autowired
    private ManagerProjectRepository repository;

    // AbstractService interface ----------------------------------------------


    @Override
    public void authorise() {
        boolean status;
        int masterId;

        masterId = super.getRequest().getData("masterId", int.class);
        Manager manager = this.repository.findManagerByProjectId(masterId);
        status = super.getRequest().getPrincipal().getActiveRoleId() == manager.getId();

        super.getResponse().setAuthorised(status);
    }

    @Override
    public void load() {
        int id = super.getRequest().getData("masterId", int.class);
        Collection<Assignment> objects = this.repository.findAllAssignmentsOfAProjectById(id).stream().distinct().collect(Collectors.toList());
        objects = this.deleteDuplicated(objects);

        super.getResponse().addGlobal("masterId", id);
        super.getBuffer().addData(objects);
    }

    @Override
    public void unbind(final Assignment assignment) {

        assert assignment != null;

        Dataset dataset;

        dataset = super.unbind(assignment, "project", "userStory");

        dataset.put("projectTitle", assignment.getProject().getTitle());
        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              boolean res = false;
74              String pair = assignment.getProject() + "-" + assignment.getUserStory();
75              if (!uniquePairs.contains(pair)) {
76                  uniquePairs.add(pair);
77                  res = true;
78              }
79              return res;
80          }).collect(Collectors.toList());
81      }
82
83  }
84
```