```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 id;

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

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

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

    @Override
    public void unbind(final Assignment assignment) {

        assert assignment != null;

        Dataset dataset;

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

        super.getResponse().addData(dataset);
        if (assignment.getProject() != null)
            dataset.put("title", assignment.getProject().getTitle());
        if (assignment.getUserStory() != null)
```

```java
                dataset.put("usTitle", assignment.getUserStory().getTitle());

    }

    private Collection<Assignment> deleteDuplicated(final Collection<Assignment> objects) {
        Set<String> uniquePairs = new HashSet<>();
        return objects.stream().filter(assignment -> {
            String pair = assignment.getProject() + "-" + assignment.getUserStory();
            if (uniquePairs.contains(pair))
                return false;
            else {
                uniquePairs.add(pair);
                return true;
            }
        }).collect(Collectors.toList());
    }

}
```