```java

package acme.features.manager.assignment;

import java.util.Collection;

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.client.views.SelectChoices;
import acme.entities.project.Assignment;
import acme.entities.project.Project;
import acme.entities.project.UserStory;
import acme.features.manager.project.ManagerProjectRepository;
import acme.roles.Manager;

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

    @Autowired
    private ManagerProjectRepository repository;


    @Override
    public void authorise() {
        boolean status = super.getRequest().getPrincipal().hasRole(Manager.class);

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

    @Override
    public void load() {

        int id = super.getRequest().getData("id", int.class);
        Assignment assigment = this.repository.findAssignmentById(id);

        super.getBuffer().addData(assigment);

    }

    @Override
    public void bind(final Assignment assigment) {
        assert assigment != null;

        super.bind(assigment, "project", "userStory");
    }

    @Override
    public void validate(final Assignment assigment) {
        assert assigment != null;
        boolean updateable = this.repository.findProjectOfAnAssignmentByAssignmentId
(assigment.getId()).isDraft();

        if (!super.getBuffer().getErrors().hasErrors("project")) {
            super.state(!assigment.getProject().isHasFatalErrors(), "project",
"manager.project.form.error.fatal-errors");
            super.state(updateable, "*", "manager.project.form.updateable");
            super.state(assigment.getProject().isDraft() == true, "*",
"manager.project.form.create-denied");
        }

```

```
60      }
61
62      @Override
63      public void perform(final Assignment assigment) {
64          assert assigment != null;
65
66          this.repository.save(assigment);
67      }
68
69      @Override
70      public void unbind(final Assignment assigment) {
71          assert assigment != null;
72
73          Dataset dataset;
74
75          int id = super.getRequest().getPrincipal().getActiveRoleId();
76          SelectChoices projectChoices;
77          SelectChoices userStoriesChoices;
78          Collection<Project> projects = this.repository.findAllProjectsByManagerId(id);
79          Collection<UserStory> userStories = this.repository.findAllUserStoriesOfAManagerById
    (id);
80
81          projectChoices = SelectChoices.from(projects, "title", assigment.getProject());
82          userStoriesChoices = SelectChoices.from(userStories, "title", assigment.getUserStory
    ());
83
84          dataset = super.unbind(assigment, "project", "userStory");
85
86          dataset.put("projects", projectChoices);
87          dataset.put("userStories", userStoriesChoices);
88
89          super.getResponse().addData(dataset);
90      }
91
92  }
93
```