```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 ManagerAssignmentCreateService 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() {

        Assignment assigment = new Assignment();
        Integer id;

        id = super.getRequest().getData("masterId", int.class);
        Project p = this.repository.findProjectById(id);

        assigment.setProject(p);

        super.getBuffer().addData(assigment);

    }

    @Override
    public void bind(final Assignment assigment) {
```

```
49          assert assigment != null;
50
51          super.bind(assigment, "project.title", "userStory");
52      }
53
54      @Override
55      public void validate(final Assignment assigment) {
56          assert assigment != null;
57
58          if (!super.getBuffer().getErrors().hasErrors("project")) {
59              super.state(!assigment.getProject().isHasFatalErrors(),
    "project", "manager.project.form.error.fatal-errors");
60              super.state(assigment.getProject().isDraft() == true, "*",
    "manager.project.form.create-denied");
61          }
62
63          if (!super.getBuffer().getErrors().hasErrors("userStroy")) {
64              int masterId = super.getRequest().getData("masterId",
    int.class);
65              Collection<UserStory> us =
    this.repository.findAllUserStoriesOfAProjectById(masterId);
66              super.state(!us.contains(assigment.getUserStory()), "userStory",
    "manager.project.form.UsDuplicated");
67          }
68
69      }
70
71      @Override
72      public void perform(final Assignment assigment) {
73          assert assigment != null;
74
75          this.repository.save(assigment);
76      }
77
78      @Override
79      public void unbind(final Assignment assigment) {
80          assert assigment != null;
81
82          Dataset dataset;
83
84          int id = super.getRequest().getPrincipal().getActiveRoleId();
85          SelectChoices userStoriesChoices;
86
87          Collection<UserStory> userStories =
    this.repository.findAllUserStoriesOfAManagerById(id);
88
89          userStoriesChoices = SelectChoices.from(userStories, "title",
    assigment.getUserStory());
90
91          dataset = super.unbind(assigment, "project.title", "userStory");
```

```
 92
 93         int masterId = super.getRequest().getData("masterId", int.class);
 94
 95         dataset.put("masterId", masterId);
 96         dataset.put("userStories", userStoriesChoices);
 97
 98         super.getResponse().addData(dataset);
 99     }
100
101 }
102
```