```java

package acme.features.manager.project;

import java.util.stream.Stream;

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.Project;
import acme.entities.project.acceptedCurrency;
import acme.roles.Manager;

@Service
public class ManagerProjectCreateService extends AbstractService<Manager, Project> {

    @Autowired
    private ManagerProjectRepository repository;


    @Override
    public void authorise() {

        super.getResponse().setAuthorised(super.getRequest().getPrincipal().hasRole(Manager.class));

    }

    @Override
    public void load() {
        Manager manager;

        manager = this.repository.findManagerByManagerId(super.getRequest().getPrincipal().getActiveRoleId());
        Project project = new Project();
        project.setDraft(true);
        project.setHasFatalErrors(false);
        project.setManager(manager);

        super.getBuffer().addData(project);
    }

    @Override
    public void bind(final Project project) {
        assert project != null;

        super.bind(project, "code", "title", "abstractText", "hasFatalErrors", "cost", "link", "isDraft", "hasFatalErrors");
    }

    @Override
    public void validate(final Project project) {
        assert project != null;

        if (!super.getBuffer().getErrors().hasErrors("cost")) {
            boolean isCurrencyAccepted = Stream.of(acceptedCurrency.values()).map(x -> x.toString().toLowerCase().trim()).anyMatch(currency -> currency.equals(project.getCost().getCurrency().toLowerCase().trim()));

            super.state(isCurrencyAccepted, "cost", "manager.project.form.error.incorrectConcurrency");
```

```
57              super.state(project.getCost().getAmount() >= 0, "cost",
"manager.project.form.error.negative-cost");
58          }
59
60          if (!super.getBuffer().getErrors().hasErrors("code")) {
61              Project existing;
62
63              existing = this.repository.findOneProjectByCode(project.getCode());
64
65              super.state(existing == null, "code", "manager.project.form.error.duplicated");
66          }
67      }
68
69      @Override
70      public void perform(final Project project) {
71          assert project != null;
72
73          this.repository.save(project);
74      }
75
76      @Override
77      public void unbind(final Project project) {
78          assert project != null;
79
80          Dataset dataset;
81
82          dataset = super.unbind(project, "code", "title", "abstractText", "hasFatalErrors",
"cost", "link", "isDraft", "hasFatalErrors");
83
84          super.getResponse().addData(dataset);
85      }
86
87  }
88
```