```java

package acme.features.manager.project;

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.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("hasFatalErrors"))
            super.state(!project.isHasFatalErrors(), "hasFatalErrors",
"manager.project.form.error.fatal-errors");

        if (!super.getBuffer().getErrors().hasErrors("cost"))
            super.state(project.getCost().getAmount() >= 0, "cost",
"manager.project.form.error.negative-cost");

        if (!super.getBuffer().getErrors().hasErrors("code")) {
            Project existing;

            existing = this.repository.findOneProjectByCode(project.getCode());

            super.state(existing == null, "code", "manager.project.form.error.duplicated");
        }
    }

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

        this.repository.save(project);
    }

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

        Dataset dataset;

        dataset = super.unbind(project, "code", "title", "abstractText", "hasFatalErrors",
"cost", "link", "isDraft", "hasFatalErrors");

        super.getResponse().addData(dataset);
    }

}

```