REST Testing

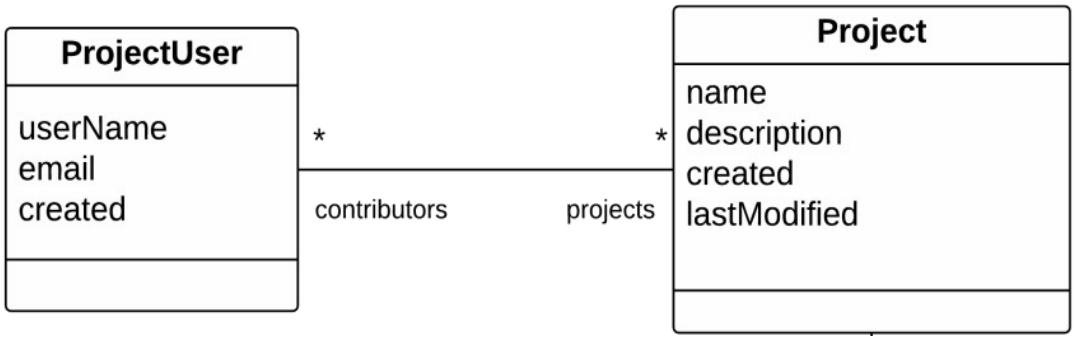
General part

- Explain the difference between Unit test and (REST) API testing.
- What kind of things do you test with API testing.
- Explain the four HTTP methods used for create, read, update, and delete in a REST based API.

Practical part

1. Download the test project from github (https://github.com/sofusalbertsen/ExamPrep)
2. Make an empty database named “ExamPrep”
3. Go into the persistence.xml file and edit the username and password to your own.

The model below is an initial model for a system than can handle *Users* and *Projects* assigned to *Users* (contributors to the project).



The REST service that you are given has the following endpoints:

/users

/projects

Tasks:

/Users is already implemented, but /projects is not implemented yet.

Implement the following actions in the /projects endpoint (ProjectRestService.java):

- Create a project
- Get all projects

- Get a project
- Assign a user to a project[1]

Make tests that (not necessary in the following order) :

- Test CRUD methods on the entities.
- Create a user and a project.
  Add the user to the project.
  Check that the user is added to the project, and vice versa.
- Create two tasks to the project.
  Check that they are assigned.
  Delete one and verify that the given task description is the same.

As you can see, none of the services have exception mappers to handle server side errors gracefully.
Implement mappers that map a given exception into the JSON data format as shown in the example:

{"code": 404, "message": "Project with the given ID is not in the database"}

- Implement a ProjectNotFoundException class.
- Implement a ProjectNotFoundExceptionMapper if a project with the given ID does not exist.
- Implement an ExceptionMapper that will Map all Exceptions not having their own ExceptionMapper
- Implement an ExceptionMapper that will Map NotFoundExceptions to the relevant JSON-Response

1 Be aware of the cyclic reference from User to Project when assigning