Structuring Applications

General part

Explain generally about:

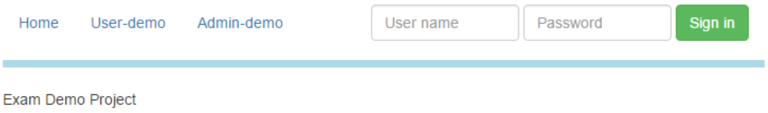
- Explain the purpose(s) of Java Packages.
- Explain how even large programs can be made "understandable" by means of just packages and file-names.
- Explain the "architecture" of both sides of the seed: https://github.com/Lars-m/semesterSeedFall2015.git
- Explain how you could have changed the seed, to provide a better way for storing username and passwords

Practical part

The purpose of this exercise is to show that you hook into an existing piece of code and "twist" it to your needs.

Getting started: Clone the Seed and open the two projects in NetBeans.

1) Change the code to provide an UI as sketched in this figure. Remove all code that is no longer used (leave anything in the components +view2+3 folders, and don't touch the backend yet). For the Home Menu Entry, just show a simple message as sketched on the figure



2) Remove all existing users, and then add two new users (just use the hard coded `UserFacade` for this exercise):

- User-1: userName: **Peter**, password: **test**, role: **User**
- User-2: userName: **Anne**, password: **test**, role: **Admin**

Add the necessary changes to the index.html file so that:

- All Users (even when not signed in) can see the Home menu-item.
- Users logged in with the User-role can also see the User-demo menu-item (Use the `$scope.isUser` property)
- Users logged in with the Amin-role can see the Admin-demo menu-item (Use the `$scope.isAdmin` property)

3) Change the User-demo page

- Change the Backend REST-service to use the following path: ***api/footballclubs*** and return the following hardcoded JSON-array[1]:
  [{"name":"Liverpool", "url":"http://www.liverpoolfc.com"},{"name":"Manchester United","url" : "http://www.manutd.com/"}]
- Change the view to show an unordered list with links for each of the clubs received via the call to: api/footballclubs

4) Change the Admin-demo page

- Change the Backend REST-service to use the following path: ***api/allusers*** and return the following hard-coded JSON-array:
  [{"name": "Jan","mail":"jan@a.dk"},{"name":"Ann","mail":"ann@a.dk"},{"name":"ib","mail":"ib@a.dk"}]
- Change the view to show a table with all the users received via the call to: api/allusers

5) Upload you solution to OPENSHIFT (see note, on the next page)

6) If you have time: Change the test's for the view-2+3 controllers to verify the new behaviour

[1] Hint: create an empty string like " " and paste the content from above into the string, NetBeans will escape all the "'s

Note: this part is NOT a part of the exercise, it's meant as FYI and this section will not be included with the real question.

We know that a lot of things can be tricky when uploading a solution using a remote database provided by OPENSHIFT. This exercise however don't use a Database, so uploading your war-file should be trivial if you done this before.

We suggest that, for the exam you have a Tomcat project ready on OPENSHIFT and a deployment project (the project where you use git to push you war-file up to OPENSHIFT) ready on your development laptop. If this is ready, this step should be as simple as copying your *war-file* into this project and do a:

```
Git add –all
Git commit -m "a message"
Git push
```