# Jenkins & Docker Assignment

1. Consider the attached project. It is the SpringBoot project with Spring Data and MySQL support.
2. Import it is Spring Tool Suite and configure it locally to run it as Spring Boot App.
3. You may need to configure MySQL credentials and database name.
4. Execute the project locally and access the app at <http://localhost:portnumber>
5. Once, it is working fine in local development environment; do the following as a part of deploying the app to docker as container:
   1. Push the app source in internal GIT server. Pl. ask your mentor for the Internal GIT server URL.
   2. Configure Jenkins locally to pull the source from internal GIT repository
   3. Jenkins should build the project and create the deployable (war/jar)
   4. Create a docker file and docker-compose.yaml such that, when you run the following command:
      1. docker-compose up
      2. It should deploy the attached project at <http://localhost>
   5. Hints:
      1. Docker Compose file should:
         1. Create ngix container
         2. Create mysql container
         3. Configure attached “app” to run in docker containers
6. Spring Boot & MySQL Project
   1. 
7. Once the app is deployed in docker; do the following to demonstrate the Continuous Deployment:
   1. Change the source code in IndexController.java to add one more entity to database in line 22 (in if condition):
      1. postRepository.save( **new** Post(1, "This is first POST!!!!", **new** Date()));
   2. Change the source code in IndexController.java to one more attribute or model (in if condition):
      1. model.addAttribute("newfeature", "I am running from Docker");
   3. Change the source code in view template i.e. index.ftl to display the value of attribute, “newfeature”
      1. <p>${newfeature}</p>
8. Make it sure that project is running locally in development environment without errors.
9. If it running locally without errors, push the changes to the internal GIT repository which was connected in 5.a
10. If 5.b was done correctly, Jenkins will automatically pull the code updates from internal GIT repo and build and deploy the project with updated code.
11. Now, when you visit <http://localhost>; you should see the changes in the browser window and new record (2 records) in MySQL table (you have configure the same in step 3)