Syllabus for the Jenkins course

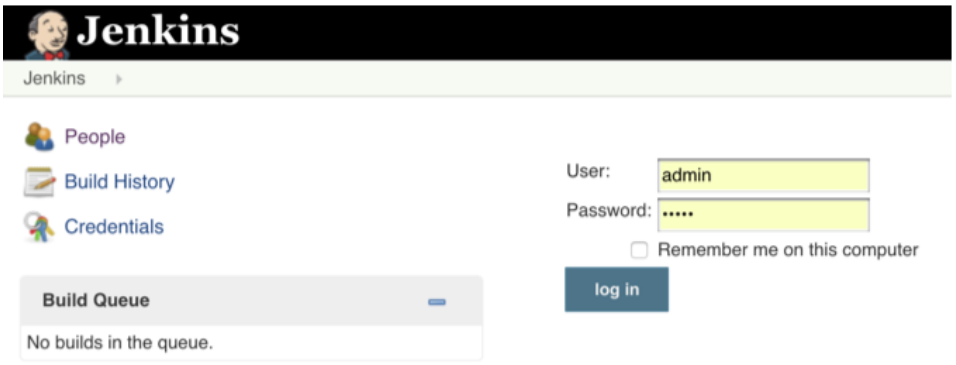
Initial state

Every student is supplied with the credentials for the virtual course PC and could reach it via HTTP + SSH and has an account in the DockerHub (https://hub.docker.com/)

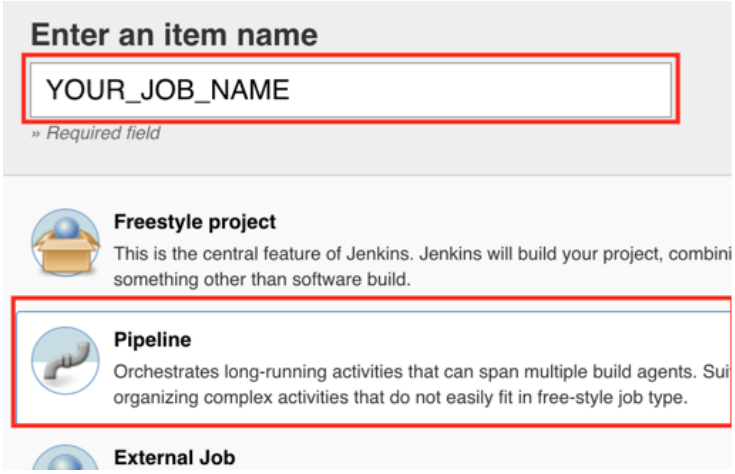
Preliminary items

1. **Get to the course Jenkins environment using the following credentials:**

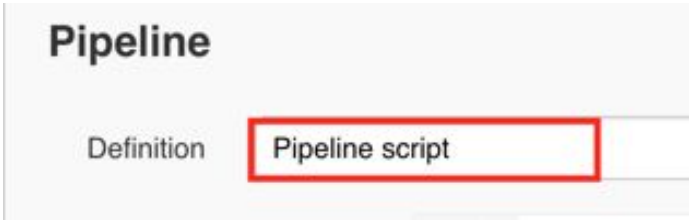
http://<your course PC IP>:8080/
User: *admin*
Password: *admin*



2. **Create a Jenkins Pipeline job:**



3. **Set the pipeline definition as "Pipeline Script" and start creating the code**



Create the Groovy pipeline with the following stages:

1. Stage 'Preparation':
   - Define Maven home variable and connect it to the M3 alias from Global Tool Configuration
   - Source the course test code from the public GIT repository: https://github.com/zivkashtan/course.git

2. Stage 'Creating Package':
   - Using the predefined in stage 1 home variable, create the package with maven
   - Check the result *.war files. The files could be found via the SSH CLI on the filesystem:

sudo su - jenkins
ls -ltr ~/workspace/<YOUR_JOB_NAME>/web/target/
<supervise the file>
exit

3. Stage 'Creating Dockerfile':

- Create the Dokerfile for creating the product image based on 'tomcat:8.0.20-jre8'
- Add to the Dokerfile the *.war file with the product: destination folder in the tomcat is /usr/local/tomcat/webapps/

4. Stage 'Docker build image':
   - Run the docker build command with tag (***<your_dockerhub_username>/time-tracker***) to create an image in the local docker

5. Stage 'Ansible push image':
   - Create/modify the example /home/ubuntu/hosts inventory file via SSH CLI to let the ansible work with the localhost. The file looks like the following example:

[local]
localhost ansible_connection=local

- Create/modify the example /home/ubuntu/docker_push_playbook.yml file via SSH CLI to trigger the ansible do a docker login to your dockehub repository and push there the built image:

```
- name: Ansible Docker PUSH step
  hosts: localhost
  tasks:
    - name: Log into Docker Hub and force re-authorization
      docker_login:
        username: <YOUR_DOCKERHUB_USERNAME>
        password: <YOUR_DOCKERHUB_PASSWORD>
        email: <YOUR_DOCKERHUB_EMAIL>
        reauthorize: yes

    - name: push an image
      docker_image:
        name:  <YOUR_DOCKERHUB_USERNAME>/time-tracker
        tag: latest
        push: yes
```

- Run command via Jenkins to apply the playbook:

```
ansible-playbook /home/ubuntu/docker_push_playbook.yml -i /home/ubuntu/hosts
```

6. Stage ‘Ansible pull and run image’:
   - Create/modify the example /home/ubuntu/docker_pull_run_playbook.yml file via SSH CLI to trigger ansible to pull your dockehub repository image and run it with the port exposing:

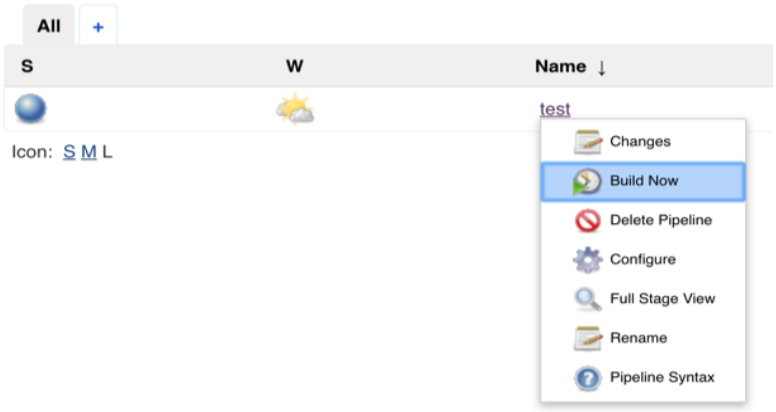
```
- name: Ansible Docker step
  hosts: localhost
  tasks:
    - name: pull an image
      docker_image:
        name: <YOUR_DOCKERHUB_USERNAME>/time-tracker:latest

    - name: Start a container
      docker_container:
        name: time-tracker
        image: <YOUR_DOCKERHUB_USERNAME>/time-tracker:latest
        state: started
        ports:
```

- "80:8080"

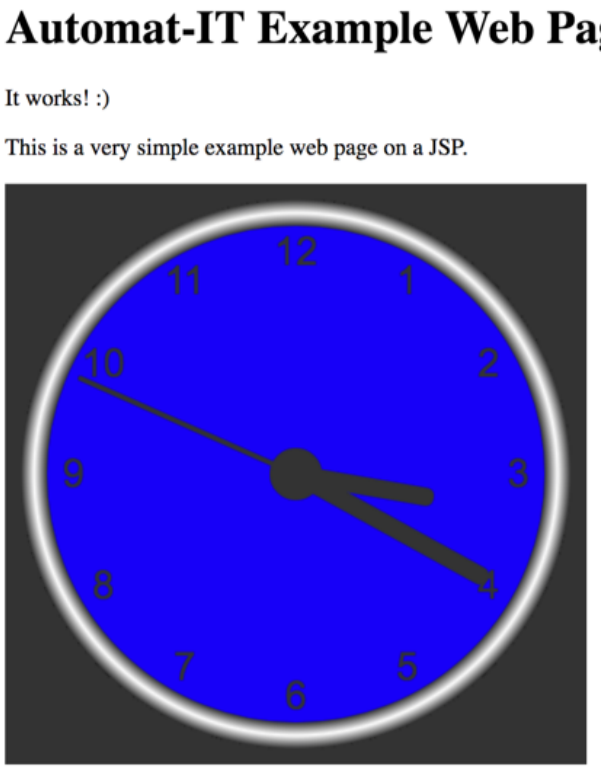
- Run command via Jenkins to apply the playbook:

ansible-playbook /home/ubuntu/docker_pull_run_playbook.yml -i /home/ubuntu/hosts

7. Run the pipeline and see the stdout:



8. Verify the results using the a WEB browser:
   - Go to *http://<your course PC IP>:80/time-tracker-web-0.3.1/*
   - You should see the tremendous Automat-IT example web page:



9. Switch under the jenkis user and modify the source code:

sudo su - jenkins
vi ./workspace/<YOUR_JOB_NAME>/web/src/main/webapp/index.jsp

Put 'green' instead of 'blue':

```
...
function drawFace(ctx, radius) {
  var grad;
  ctx.beginPath();
  ctx.arc(0, 0, radius, 0, 2*Math.PI);
  ctx.fillStyle = 'green';
  ctx.fill();
  grad = ctx.createRadialGradient(0,0,radius*0.95, 0,0,radius*1.05);
...
```

10. Go to Jenkins pipeline and commit out in the Stage 'Preparation' the line with the git clone:

```
stage('Preparation') {
    // Get some code from a GitHub repository
    //git 'https://github.com/zivkashtan/course.git';
    // Get the Maven tool.
```

11. Run the Pipeline and verify the results using the a WEB browser:

- Go to *http://<your course PC IP>:80/time-tracker-web-0.3.1/*
- You should see the fascinating Automat-IT example web page with green color:

Automat-IT Example Web Page

It works! :)

This is a very simple example web page on a JSP.