**Please create project on GitHub with the script (BASH, Ansible, Python, etc.) All tasks need be executed on the VM:**  
  
Example GITHUB account created for the purpose is: **https://github.com/testepamgit/**

user: **testepamgit**

pass: **TestqWerty43#**

To use GITHub without authenticatoin ssh id\_rsa key is imported and used:  
To push a new local GIT repository copy of /*home*/devops/EPAM/ directory  
  
devops@epam:~/EPAM$ sh git\_create.sh epamgittest1

(Here inside web browser added key to GITHub page to be able to authenticate passwordless), commands used to generate the key are below:

#!/bin/bash

# Create and push to a new github repo from the command line.

# Grabs sensible defaults from the containing folder and `.gitconfig`.

# Author: Georgi Georgiev

# hipo@pc-freak.net

#REPONAME='epamtest';

REPONAME="$1";

USER='testepamgit';

GITHUBUSER="$USER";

PWD='TestqWerty43#';

DESCRIPTION='this is a test git commit';

# Gather constant vars

CURRENTDIR=${PWD##\*/}

REPO\_DIR='EPAM';

CURL=$(which curl);

git=$(which git);

API\_URL='https://api.github.com/user/repos';

GITHUBUSER=$(git config github.user)

RANDOM=$((RANDOM%100))  
# install git and curl if they're missing on the system (for that to work must run as root user)

if [ ! -z "$1" ]; then

echo 'OK'

else

echo 'Missing REPONAME argument or other error ... exiting';

exit 1;

fi

rm -rf $HOME/.git

if [ ! -d $HOME/.git ]; then

cd ~

$git init;

echo 'Git Local Repo Initialized';

$git add $HOME/$REPO\_DIR/.;

$git commit $HOME/$REPO\_DIR/. -m "EPAM $RANDOM"

echo "Commiting";

else

echo ".git exists"

fi

echo "Here we go..."

# Curl some json to the github API oh damn we so fancy

$CURL -u ${USER:-${GITHUBUSER}}:"$PWD" $API\_URL -d "{\"name\": \"${REPONAME:-${CURRENTDIR}}\", \"description\": \"${DESCRIPTION}\", \"private\": false, \"has\_issues\": true, \"has\_downloads\": true, \"has\_wiki\": false}"

# Set the freshly created repo to the origin and push

# You'll need to have added your public key to your github account

git config --global user.email "hipodilski@gmail.com"

git config --global user.name "$USER"

$git remote add origin git@github.com:${USER:-${GITHUBUSER}}/${REPONAME:-${CURRENTDIR}}.git

$git push --set-upstream origin master

====================================================

ssh-keygen -t rsa -b 4096 -C "[hipodilski@gmail.com](mailto:hipodilski@gmail.com)"  
eval "$(ssh-agent -s)"  
ssh-add ~/.ssh/id\_rsa  
cat ~/.ssh/id\_rsa

To add a new pository (epamgittest) and push commited files from local .git repo from the VM to remote git repo:  
========================================================================================  
  
sh git\_create.sh epamgittest

It receives one argument which is the repository name to be created.  
  
(Script is a buggy but it basicly does what it is supposed to ... could be significantly improved.)

1. **Add user (Login: devops, pass: testpass) to the system**  
     
     
   # apt-get install –yes sudo  
   *# useradd devops --home /home/devops -s /bin/bash  
   # mkdir /home/devops*

*# chown -R devops:devops /home/devops  
# echo 'devops:testpass' | chpasswd  
  
#!/bin/bash*

*# Create new user/group and add nopasswd login to sudoers*

*# Author: Georgi Georgiev*

*# has to be run sa root - sudo devops*

*# hipo@pc-freak.net*

*u\_id='devops';*

*g\_id='devops';*

*pass='testpass';*

*sudoers\_f='/etc/sudoers';*

*check\_install\_sudo () {*

*if [ $(dpkg --get-selections | cut -f1|grep -E '^sudo') ]; then*

*apt-get install --yes sudo*

*else*

*printf "Nothing to do sudo installed";*

*fi*

*}*

*check\_install\_user () {*

*if [ "$(sed -n "/$u\_id/p" /etc/passwd|wc -l)" -eq 0 ]; then*

*apt-get install --yes sudo*

*apt-get install --yes sudo*

*useradd $u\_id --home /home/$u\_id*

*mkdir /home/$u\_id*

*chown -R $u\_id:$g\_id /home/$u\_id*

*echo "$u\_id:$pass" | chpasswd*

*cp -rpf /etc/bash.bashrc /home/$u\_id*

*if [ "$(sed -n "/$u\_id/p" $sudoers\_f|wc -l)" -eq "0" ]; then*

*echo "$u\_id ALL=(ALL) NOPASSWD: ALL" >> $sudoers\_f*

*else*

*echo "$u\_id existing. Exiting ..";*

*exit 1;*

*fi*

*else*

*echo "Will do nothing because $u\_id exists";*

*fi*

*}*

*check\_install\_sudo;*

*check\_install\_user;  
  
The* ***Virtual Machine {root, devops, epam } users***  *password is:* ***testpass***

1. **Grant admin privileges for user from point 1 and allow him to use “sudo” without entering password**  
     
     
   *echo 'devops ALL=(ALL) NOPASSWD: ALL*

*' >> /etc/sudoers*

*Note: Task 1 & 2 are done via script create\_user.sh also in repository*

1. **Install Jenkins (any way, Docker, .deb, .rpm)**

install\_jenkins.sh is in repository  
  
#!/bin/bash

# Install jenkins and test whether it runs prints password on prompt or send via email

# if email variable is set Jenkins password will be set to your email of choice using mail command

# NOTE: bsd-mailx package should be installed in order for email sent to work and local machine should be running a properly configured

# relay SMTP

# Author: Georgi Georgiev

# hipo@pc-freak.net

email='hipo@mail.com';

add\_repos\_install\_jenkins () {

apt-get install --yes -qq apt-transport-https git curl

wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -

if [ "$(sed -n '/jenkins/p' /etc/apt/sources.list|wc -l)" -eq 0 ]; then

echo 'deb https://pkg.jenkins.io/debian binary/' >> /etc/apt/sources.list

fi

apt-get update -qq && apt-get install --yes -qq jenkins

}

check\_j\_install () {

if [[ $(dpkg --get-selections | cut -f1|grep -i jenkins) ]]; then echo 'succesfully installed';

else printf 'Problem in installing please check'; exit 1;

fi

}

check\_j\_running\_s\_pass () {

if [[ $(ps -e -o command|grep -i jenkins) ]]; then

echo 'Jenkins process working.';

echo ‘... do more here if necessery with some more commands’;

else

echo 'not working log to file' >> jenkins.log

exit 1;

fi

JENKINS\_PASSWORD=`cat cat /var/lib/jenkins/secrets/initialAdminPassword`;  
  
  
  
Deployed Jenkins credentials are:  
========================  
http://localhost:8080/  
Jenkins user:   
admin   
pass:testpass

1. **Create a freestyle Job in Jenkins that will show environment variables.**

I assume here the freestyle job has to be created manually, job is created and made to simply execute the shell command **env** and it executes fine was that the task **?**Jenkins freestyle project is configured to execute shell script as such:  
 **#!/bin/sh**

**echo “Jenskins Environment Variables”**

**echo "BUILD\_NUMBER" :: $BUILD\_NUMBER**

**echo "BUILD\_ID" :: $BUILD\_ID**

**echo "BUILD\_DISPLAY\_NAME" :: $BUILD\_DISPLAY\_NAME**

**echo "JOB\_NAME" :: $JOB\_NAME**

**echo "JOB\_BASE\_NAME" :: $JOB\_BASE\_NAME**

**echo "BUILD\_TAG" :: $BUILD\_TAG**

**echo "EXECUTOR\_NUMBER" :: $EXECUTOR\_NUMBER**

**echo "NODE\_NAME" :: $NODE\_NAME**

**echo "NODE\_LABELS" :: $NODE\_LABELS**

**echo "WORKSPACE" :: $WORKSPACE**

**echo "JENKINS\_HOME" :: $JENKINS\_HOME**

**echo "JENKINS\_URL" :: $JENKINS\_URL**

**echo "BUILD\_URL" ::$BUILD\_URL**

**echo "JOB\_URL" :: $JOB\_URL  
  
echo “===Linux Shell Variables ===”  
env**

The freestyle job is inside the jenkins and called

\*\* - Additional

1. **\*\* Deploy job to the same project on GitHub and allow Jenkins to grub it, and run after point 4 automatically**

I have added a new Token via[**https://github.com/settings/tokens**](https://github.com/settings/tokens) to connect Jenkins to githubToken is: **8481f440fd8fe762fadb84d290b455a990350dd5**For task 5 actually it is necessery to configure GITSCM Polling (in Jenkins Build Triggers section) and then use Webhook on github website but for that you need to have a public IP (setting up DynDNS and a like is an option but I don’t have the time to do it at the moment …).  
Just for the sake of showing up it can be done I’ve Enabled Web Hook pointing to the repository testepamgit (with test URL <http://ci.jenkins-ci.org/github-webhook/>).  
  
Web Hooks are enabled from **Repository → Settings → Integration** and Services  
  
**Conclusion and Notes**

There are many other ways to do the tasks anyways. From the assignment given it was not crear to me, do you want me to create the free style job inside a Jenkins Web interface or you want me to do it via a script, so I made it from the webtool itself, I assume there is also an API which can be used to automate the task but I didn’t have the time to research it further, however as I said in the interview my experience with Jenkins so far is basic though if necessary it can be further learned and done.

P.S. You can use any modern Linux distro, like Debian, Ubuntu, Centos.

Results:

* URL on git repo
* VM image

If additional task is done, Just an URL on Repo is possible

!!! Automation Scripts are into: **/*home/devops/EPAM*** *into the Virtual Guest OS !!!   
List of automation EPAM scripts*

*=====================  
create\_rsa\_key.sh*

*create\_user.sh*

*git\_create.sh*

*install\_jenkins.sh*

*jenkins\_env.sh*

*Vbox VM image with Debian 9 Linux installed is uploaded on:* [*https://drive.google.com/open?id=1R3NSGaRn3Mo3iGScdmJuzOIoCU1zcJHN*](https://drive.google.com/open?id=1R3NSGaRn3Mo3iGScdmJuzOIoCU1zcJHN) *URL on git repo:* ***https://github.com/testepamgit/***