Git :

<https://github.com/dzbeda/targil3.git>

jfrog artifactory-OSS

* Jfrog was created using docker-compose file
  + Port 8081 is exposed to port 8081
  + In order to define permanent storage, the container was mapped to local storage under : /data/jfrog/storage
  + Docker compose file can be found under <github-repo>/jfrog/docker-compose.yml
* General repo was created on the Jfrog-Artifactory
  + Repo name : binary-storage
  + Permission were given to super-user
* Artifactory plugin was added to Jenkins

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Docker file

* Docker file can be founder under <github-repo>/Dockerfile
  + Docker is creating a /zip folder were the zip files will be created
  + Items with the highest potential to be updated were included in the end of the file, this in order to avoid as mush as possible layer creation on each docker build
* When running the Docker file as a standalone you will get a print that shows OS type, Architecture and also verified that zip\_job.py was copied to tmp folder
  + The Docker image runs a bash script that gather the info, check if file exists and print the output
  + Bash script can be found under <github-repo>/get\_info.sh

Text

Description automatically generated

Python script

* Python script can be founder under <github-repo>/zip\_job.py
* The scripts checks if environment variable $VERSION exists – if not it will exit
* The script will create file based on array variable ($files)
  + Files will be save to /tmp ( $ file\_path)
  + File extension can be updated ($file\_extension)
* The script will generate ZIP file for each txt file
* The script will create a log for the operation. Log is created under - /tmp/output.log
  + Log file will hold the output of the last run only (was defined as write and not append)

Jenkins job

* Jenkins file declarative file can be found under <github-repo>/Jenkinsfile
* The job create a jenkins agent based on the Docker image
  + The Container will run with –privileged
  + The Container /zip folder will be mapped to server that runs the container under $workspace/zip (This is required in order to upload the files to the jfrog)
* **Stage : Agent information** 
  + This stage will display the Container shows OS type, Architecture and Inform if zip\_job.py was copied to tmp folder  
      
    Text

    Description automatically generated
* **Stage : Build**
  + In this stage zip\_job.py under /tmp will be executed – some information will be printer (env.VERSION , and the files that were create – zip and txt)  
      
    Text, letter

    Description automatically generated
* **Stage : Show Log**
  + This stage will show the log that was generated by the zip\_job python script (the log was generated under /tmp/output.log)  
      
    A screenshot of a computer

    Description automatically generated with medium confidence
* **Stage : Publish**
  + This stage upload the ZIP file to the Jfrog
  + Please note that **I have failed to** assign the env.VERSION to the Artifactory target path. When I have exported the env.VERSION coming from the Container to new variable that can be used by the server hosting the container, it included extra spaces that was not excepted by the Artifactory (I tried few methods but unfortunately I have failed – On line 25 that is comment out you can see that I have exported the env.VERSION)  
      
    Text

    Description automatically generated  
      
    Graphical user interface, text, application

    Description automatically generated
* **Post stage**
  + On job **success** it will send an email to 2 recipients with email subject **Job passed successfully** & the log will be attached  
      
    Graphical user interface, application

    Description automatically generated
  + On job **failure** it will send an email to 2 recipients with email subject **Job failed to run** & the log will be attached  
      
    Graphical user interface, application

    Description automatically generated
  + Once job will finish, successfully or not the $workspace will be deleted from the server running the container.

Jenkins plugins

* For running the job, the following plugins were installed
  + Docker pipeline
  + Artifactory plugin – jfrom
  + Extended E-mail Notification
  + Github

Jenkins Job configuration

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Table

Description automatically generated

Table

Description automatically generated