**Домашна Работа бр. 2   
Анета Станковска  
Индекс: 249012**

**Во рамките на домашна работа бр. 2, се фокусирав на креирање container-и за повеќе сервиси како:**

* **Git engine (Gitea) за управување со репозиториуми,**
* **Jenkins за CI/CD,**
* **Nexus repository за чување артифакти од билдови.**

**За основа ги користев frontend и backend сервисите од првата домашна работа, при што додадов Dockerfile и за frontend, бидејќи претходно имаше само за backend сервисот.**

**Инсталација на Jenkins**

* Како прв чекор, потребно е да се инсталира Docker Desktop на локалната машина. Docker Desktop овозможува креирање и управување со Docker container-и, што е неопходно за подесување на Jenkins.
* Со оглед на тоа што Docker веќе имав инсталирано на мојата машина уште од првата домашна, ја стартувам Docker Desktop апликацијата за да го активирам Docker Daemon-от.
* За да го преземам Docker image-от за Jenkins (што е основата за создавање на Jenkins container), ја извршувам следната команда во терминал: **docker pull jenkins/jenkins:lts**
* Оваа команда го презема официјалниот Jenkins image со верзијата LTS (Long-Term Support), која е стабилна и препорачана за работење со Jenkins во container. Image-от содржи предефинирани поставки за Jenkins и е веќе спремен за употреба.
* Користењето на Docker за Jenkins (containerization на Jenkins) овозможува изолирана околина за Jenkins, што значи дека локалните поставки на машината не влијаат врз Jenkins, а исто така овозможува лесен пренос на Jenkins на различни машини.
* Откако image-от е преземен, следно е креирање на Jenkins container и негово поврзување со локалната мрежа.

A computer screen shot of a black screen

Description automatically generated

**Креирање на docker-compose.yml фајл**

* Во root фолдерот креирав **docker-compose.yml** фајл.
* docker-compose.yml е декларативен фајл кој овозможува дефинирање и управување со повеќе Docker container-и како дел од една апликација или проект. Целта која јас ја имав е со помош на docker-compose.yml фајлот да го исконфигуриррам креирањето на сите потребни containers (за frontend, за backend, за Jenkins, Gitea и Nexus). Овој фајл ја опишува инфраструктурата на апликацијата, вклучувајќи ги container-ите, нивните мрежи, волумени и другите конфигурации потребни за правилно функционирање.
* Во docker-compose.yml фајлот дефинирав повеќе услуги (services) кои се дел од апликацијата:
  + Frontend: Клиентскиот дел на апликацијата, кој е одговорен за корисничкиот интерфејс
  + Backend - Серверскиот дел на апликацијата, кој обработува барања и управува со податоци
  + Jenkins - Алатка за автоматизација на CI/CD процесите
  + Gitea - За управување со Git репозиториуми и верзионирање на кодот
  + Nexus - Централен репозиториум за складирање и менаџирање на артефакти, како што се Docker images и други зависности
* Стартување на container-ите
  + Со една команда. Со командата **docker-compose up** се стартуваат сите container-и дефинирани во docker-compose.yml и се поврзуваат меѓусебно според дефинираните правила
  + Rebuild на container-и. Доколку се направени промени во Dockerfile или во конфигурацијата на container-ите, за да се осигурам дека сите container-и ќе се избилдаат повторно, ја користам командата docker **compose up --build**
  + Стартување во позадина. За да ги стартувам сите container-и во позадина (detached mode) без да се прикажуваат логовите во терминалот, ја користам командата **docker-compose up -d**
* За да се осигурам дека сите container-и ќе се избилдаат соодветно (а на старите ќе им направи rebuild) ја повикувам командата **docker compose up --build**



A screenshot of a computer

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

**Креирање и конфигурација на Jenkinsfile**

* Започнав со креирање на Jenkinsfile во root фолдерот на проектот, односно во Homework\_2 фолдерот.
* Jenkinsfile фајлот го сместив таму со цел да може лесно да пристапи и до двата сервиси (server и client апликациите).
* Во Jenkinsfile ги дефинирав сите потребни конфигурации за Docker, како и чекорите за CI/CD pipeline-от, вклучувајќи ги:
  + Проверка на кодот (Checkout Code): Преземање на најновата верзија на кодот од репозиториумот
  + Билдање на Docker images: Креирање на слики за server и client апликациите
  + Тестирање: Извршување на тестови за осигурување на квалитетот на кодот
  + Публикување на Docker слики: Испраќање на сликите до DockerHub
  + Deploy: Стартување на услугите преку Docker Compose
* За да овозможам Jenkins да ги испраќа Docker сликите, додадов потребни системски креденцијали.
* Во Jenkins UI, навигирав до Manage Jenkins > Credentials и ги внесов следниве информации.
  + DockerHub credentials: Корисничко име и лозинка за најавување во DockerHub
  + Овие креденцијали се користат во Jenkinsfile за автентикација при пуштање на Docker images како дел од CI/CD пипелине-от

**Конфигурација на Jenkins**

* Го стартувам Jenkins со помош на командата: **docker run -p 8082:8080 -v jenkins\_home:/var/jenkins\_home --name jenkins jenkins/jenkins:lts** . Jenkins ни дозволува и да го смениме password-от долколку сакаме за време на setup процесот. Јас го оставам default-ниот
* Параметарот -p 8082:8080 ја поврзува порта 8080 на container-от со локалната порта 8082, овозможувајќи пристап до Jenkins преку <http://localhost:8082>.
* Причина за користење на портата 8082: Портата 8080 е веќе резервирана за фронтенд сервисот, па затоа Jenkins се конфигурира на алтернативна порта
* Преку browser ја отворам страницата: <http://localhost:8082/>
* Во првиот чекор од setup процесот, Jenkins бара администраторски password, кој е достапен во логовите или во фајлот secrets/initialAdminPassword или со командата **docker exec jenkins cat /var/jenkins\_home/secrets/initialAdminPassword**
* По првиот логин, Jenkins нуди избор за инсталација на default plugins, која јас ја одбирам за да ги инсталирам default-ните plugins кои вклучуваат docker, git, pipelines итн
* Jenkins ме прашува дали сакам да креирам кориснички профил или да продолжам како admin
* За сега, одбирам да продолжам како admin.
* Во полето за Jenkins URL внесувам: <http://localhost:8082/>
* Ова овозможува правилно поврзување на Jenkins со други системи и услуги
* Откако ќе ја зачувам конфигурацијата, се прикажува почетниот екран на Jenkins, подготвен за креирање на проекти и CI/CD pipeline-и
* За да може да креираме pipeline со помош на Jenkins, ќе треба внатре во самиот Jenkins да инсталираме docker и docker compose. Една опција е да влеземе директно во Jenkins container-от и да го инсталираме таму, но секојпат кога ќе го рекреираме container-от ќе треба повторно да ја направиме инсталацијата. Наместо тоа, јас во самиот Jenkinsfile како прв чекор сетапирам инсталација на Docker и на Docker Compose за секојпат кога ќе се креира container-от и кога ќе се стартне pipeline, најпрво да го инсталира потребните алатки.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**A screenshot of a computer

Description automatically generated**

Дополнително, ги прикачувам и Dockerfile, docker-compose.yml и Jenkinsfile за преглед.

Source кодот е поставен на моето GitHub репо, во рамки на FINKI-DevOps репозиториум.

Screenshots од дел од командите се ставени во images фолдерот во рамки на самиот репозиториум – Homework\_2 фолдерот.

[ЛИНК до репо](https://github.com/anetastankovska/FINKI-DevOps/tree/main/Homework_2)