# Retake Exam: Miroslav Kostov (Practice), 2022/11/20

Цялостно решение

Решението е изцяло автоматизирано като машините се вдигат в следната последователност: *monitoring* -> *containers* -> *pipelines* със следните идеи:

- да има работещ Prometheus server, когато на containers и *pipelines* машините се инсталират monitoring agent-ите (node\_exporter);

- *containers* машинaта да е up & running, за да може Jenkins (разположен на pipelines) да си инсталира необходимите неща на него, превръщайки го в slave node;заб. Решението е базирано на разбирането, че чрез Vagrant машините се разгръщат последователно и действията с дадена машина започват, след като тези с предишната са приключили изцяло.

Използвани скриптове

Основен Vagrant скипт : *Vagrantfile*

Скриптове, които се изпълняват на всички node-ове:

add-hosts.sh - добавя IP адресите и hostname-овете (цели и кратки) в hosts файла на отделните машини;

add-software.sh - увеличава основният *DefaultTimeoutStartSec* на машината, инсталира Java, Git, jq и tree;

Скриптове, които се изпълняват на *monitoring* node-a:

install-and-setup-prometheus-and-grafana.sh - Инсталира Prometheus и Grafana, прилага конфигурациите за тях от monitoring папката, убива процесите, заемащи техните портове (ако има такива) и ги стартира. Прави опит за импорт на datasource, dashboard и panels в Grafana (последното не стана от раз);

Скриптове, които се изпълняват на *containers* node-a:

docker-setup.sh - Инсталира Docker компоненти и Docker Compose, прилага конфигурация от docker папката, за да се eкспортват данни към Prometheus, стартира docker deamon;

provision-scripts/setup\_gitea\_bat\_mitko.sh - Добавя Gitea като контейнер чрез Docker Compose и docker-compose-gitea.yml файла от docker папката, добавя gitea user, прави локално хранилище и import-ва хранилището от изпитното задание, добавя Webhook за Jenkins, прави потребителя vagrant собственик на хранилището. Решението е заимствано от домашно #5 нас курса;

install-nodeexporter.sh - Инсталира и стартира monitoring agent на машината;

Скриптове, които се изпълняват на *pipelines* node-a:

install-jenkins.sh - Инсталира Jenkins;

provision-scripts/setup\_jenkins\_bat\_mitko.sh - Инсталира Jenkins plugins и Jenkins CLI, създава credentials за vagrant и Docker Hub, добавя containers машината като slave node, добавя pipeline-a за приложенито от изпитното задание. Решението е заимствано от домашно #5 на курса;

install-nodeexporter.sh - Инсталира и стартира monitoring agent на машината;

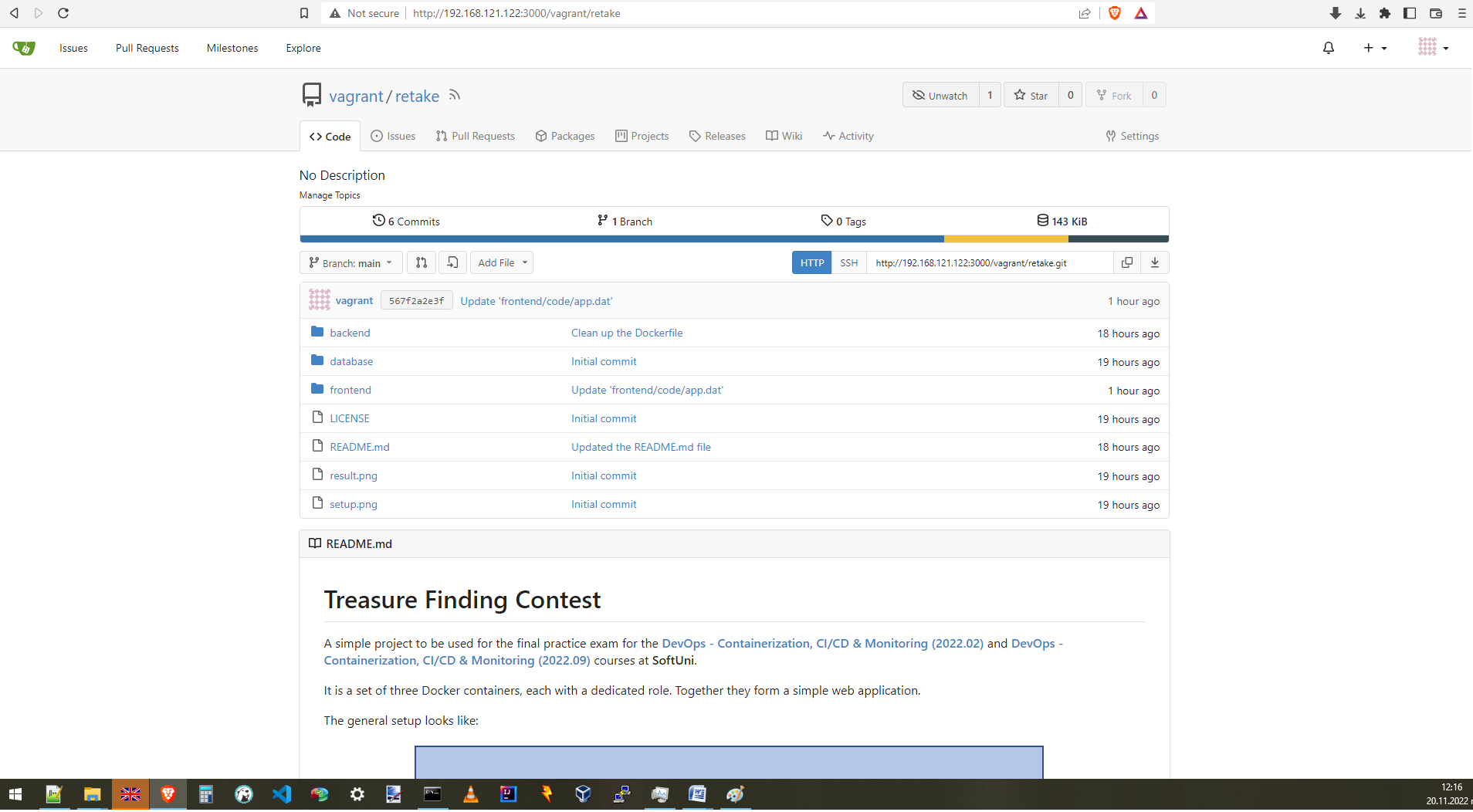
Скриптове, които не се изпълняват:

provision-scripts/install-and-setup-elasticstack.sh

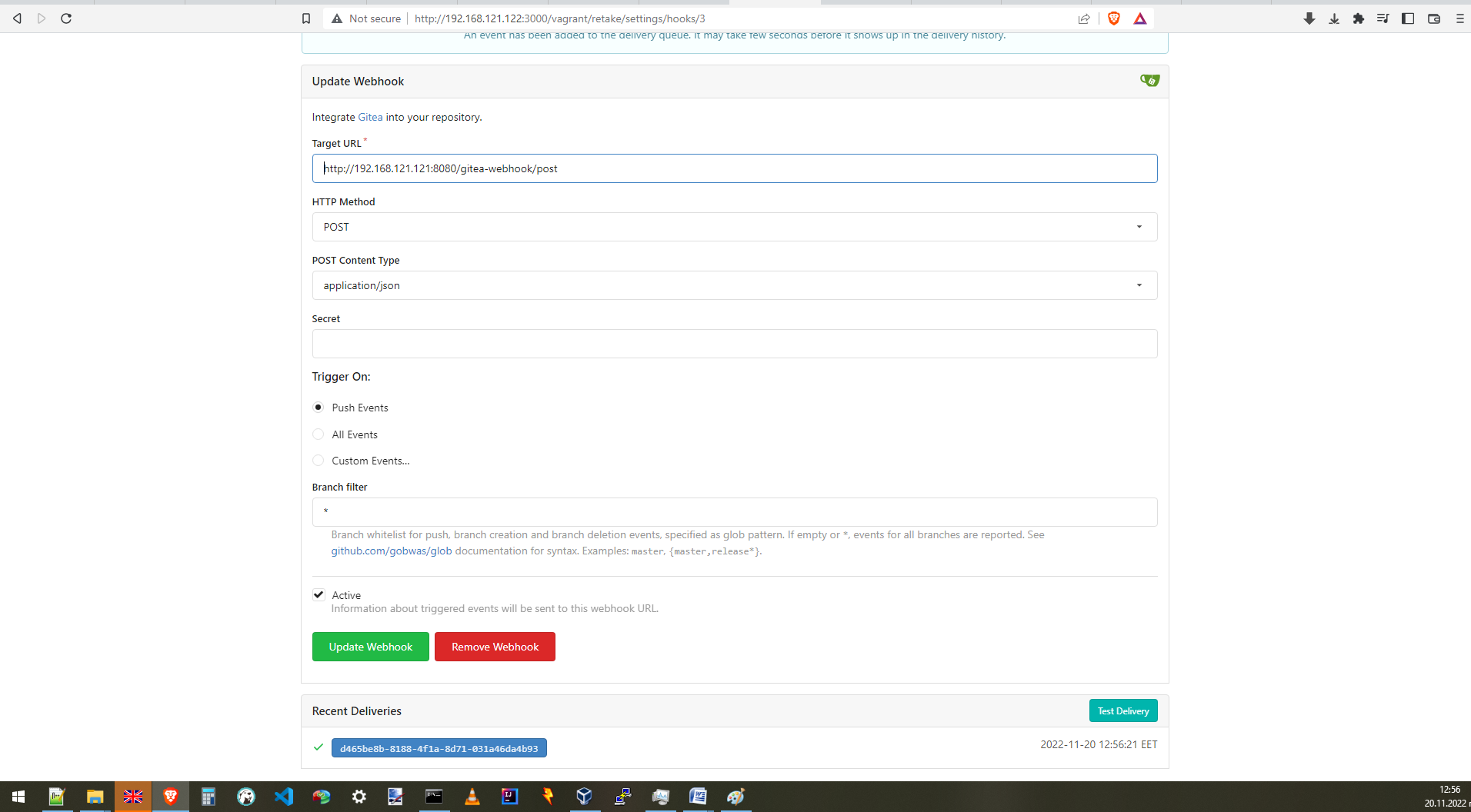
install-metricbeat.sh

Екранни снимки

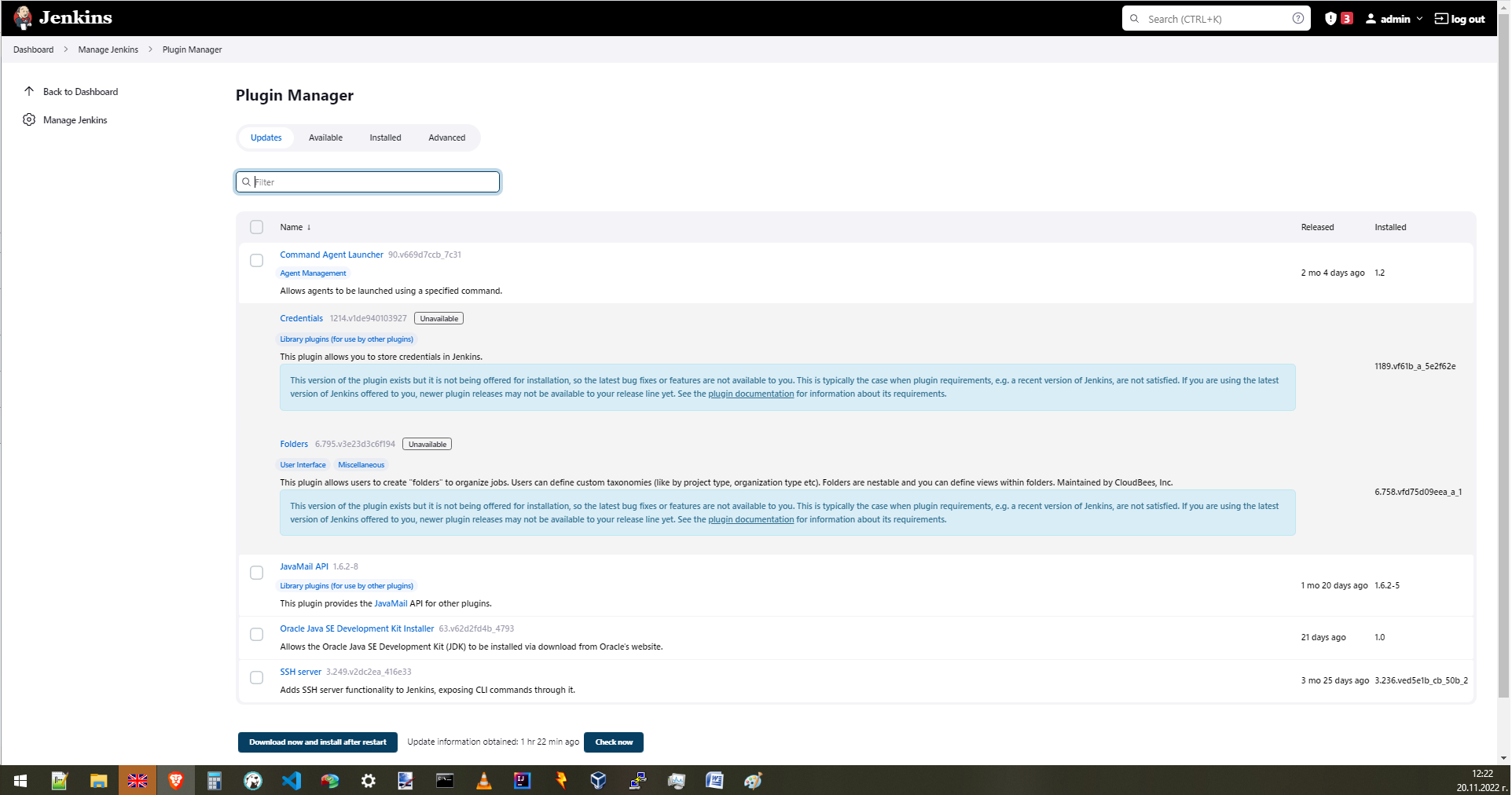
* **(T202, 3 pts)** Local copy of [**https://github.com/shekeriev/treasure-finding-contest**](https://github.com/shekeriev/treasure-finding-contest) as **retake** repository



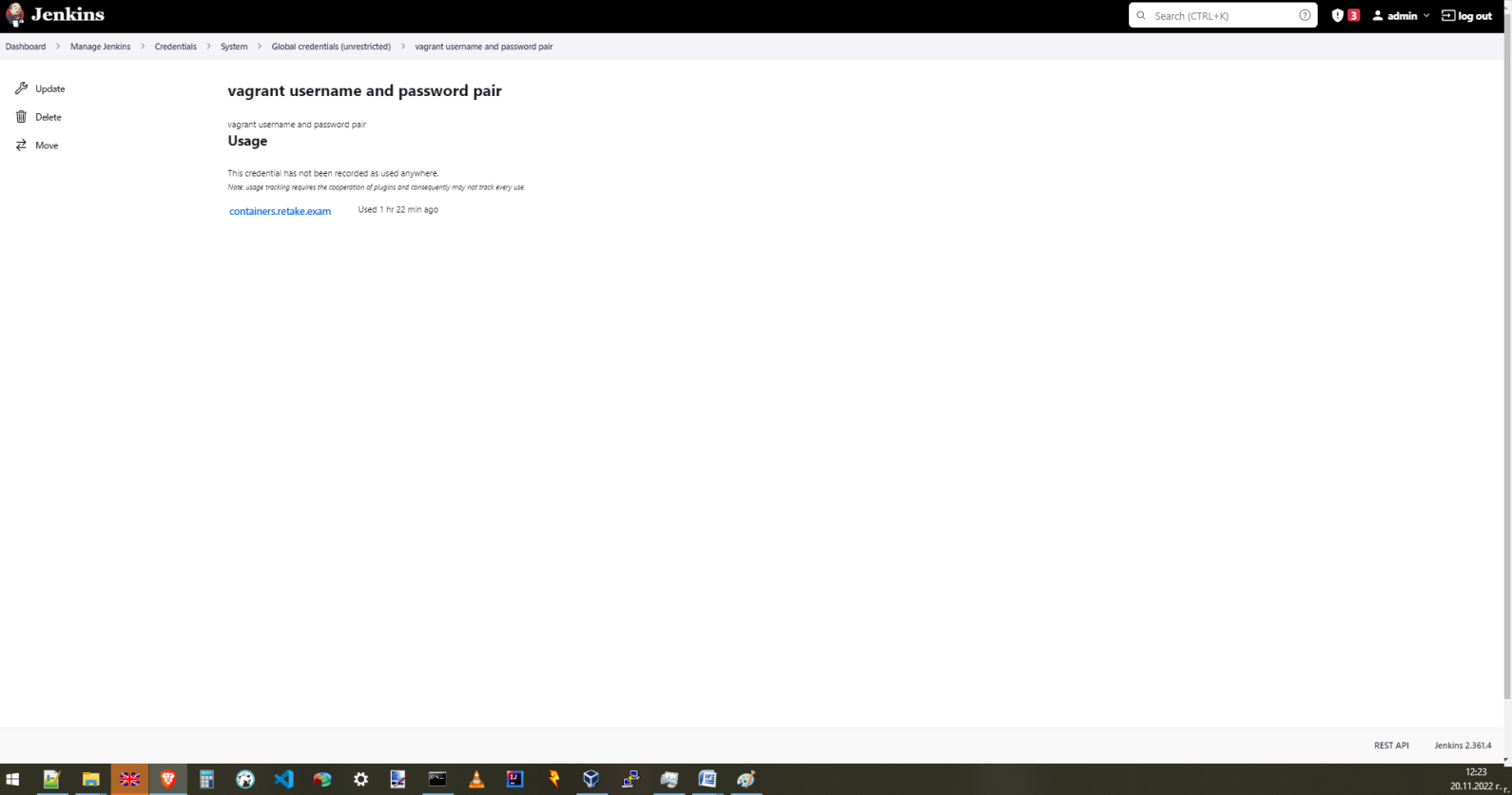
* **(T203, 3 pts)** Configured and tested web hook to **Jenkins**



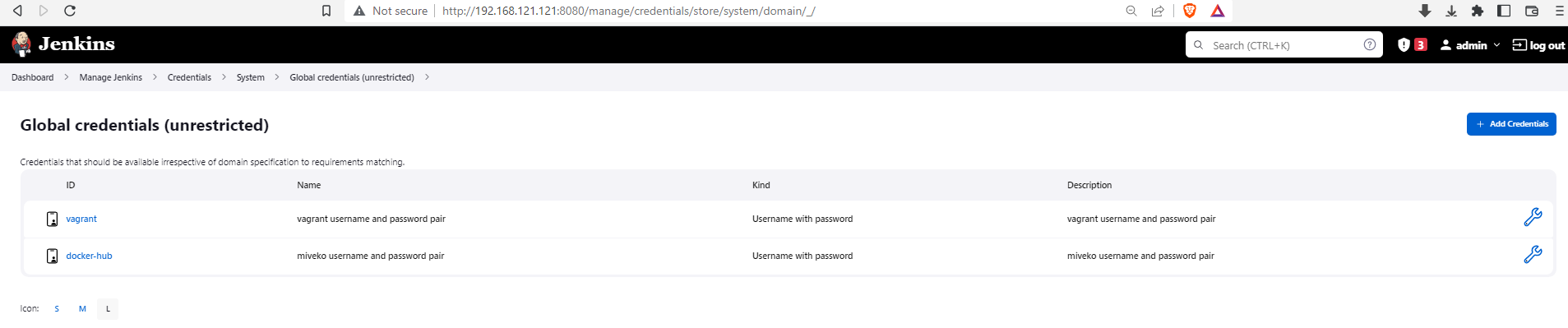
* **(T302, 1 pts)** Additional (at least one) plugin(s) installed and enabled

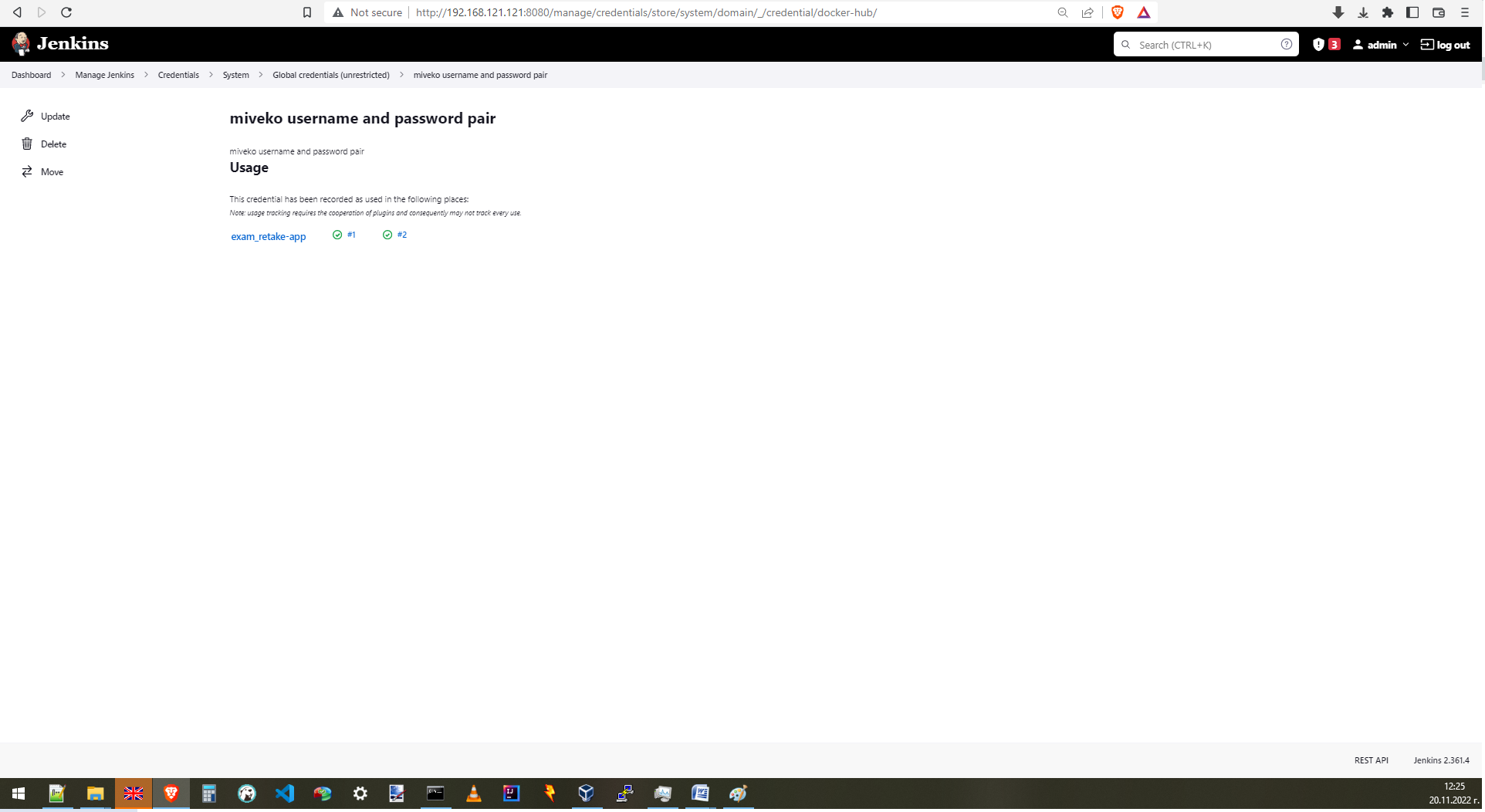


* **(T303, 1 pts)** Added credentials for the **vagrant** user

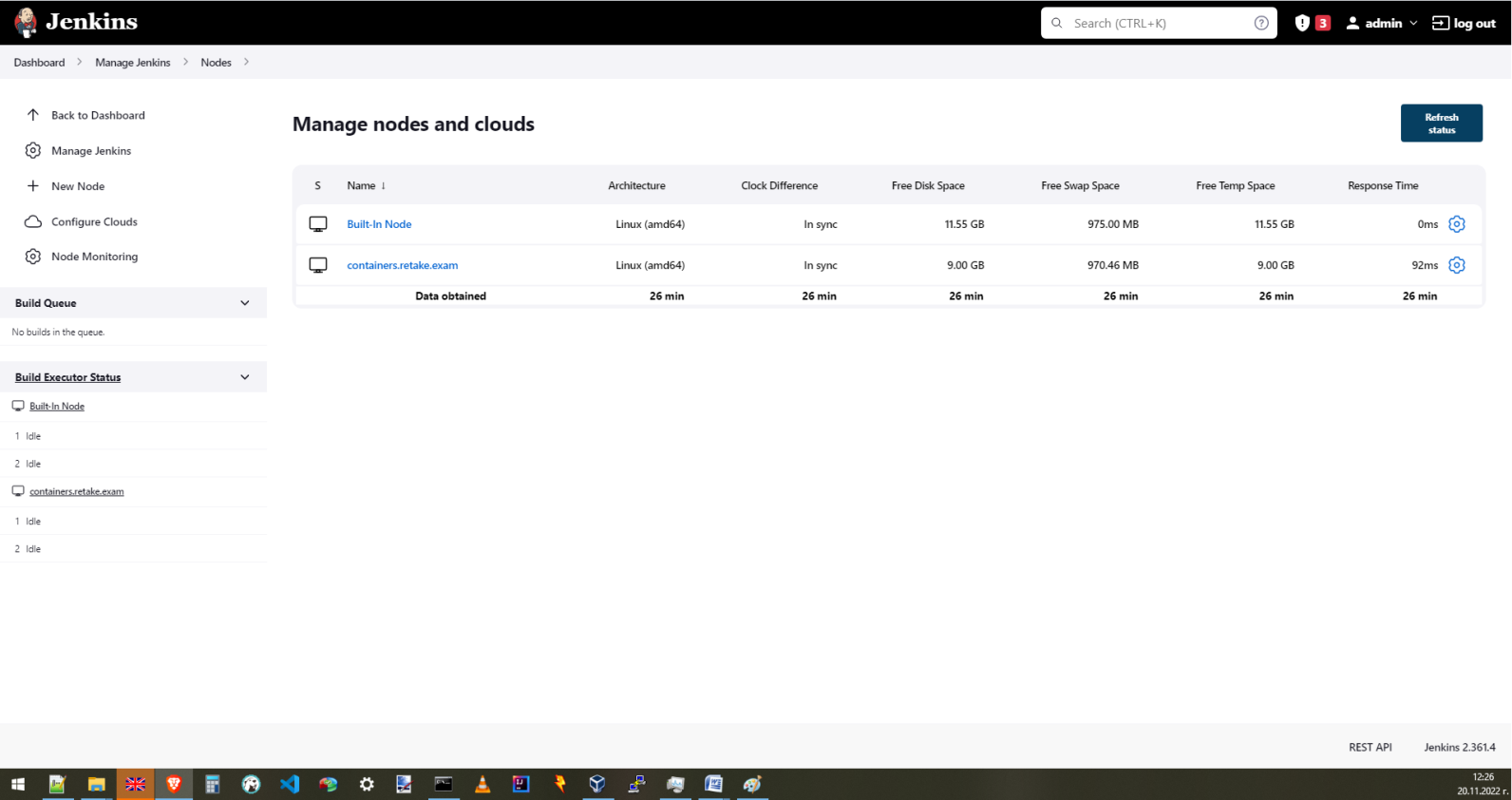


* **(T304, 1 pts)** Added credentials for **Docker Hub**

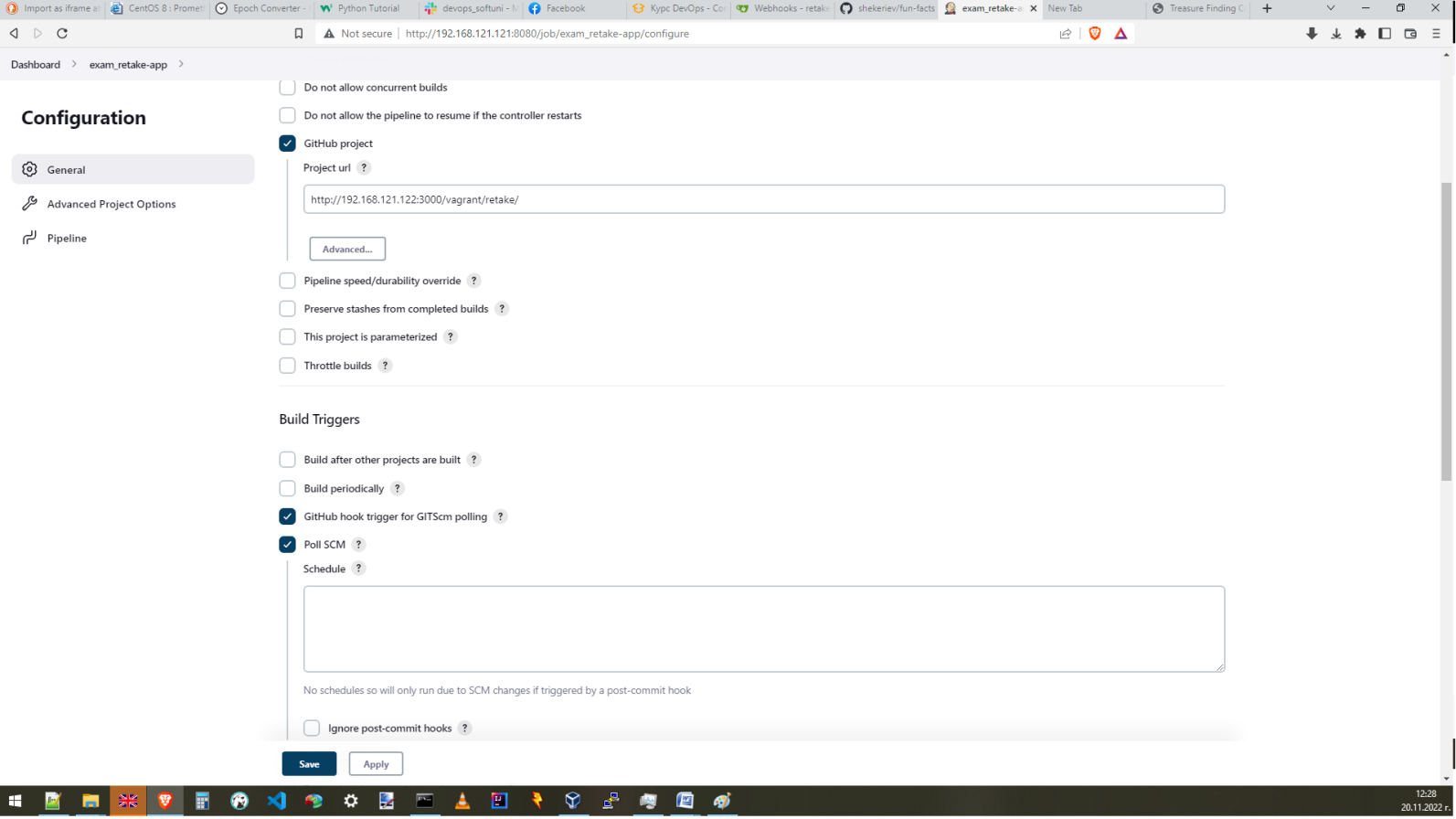


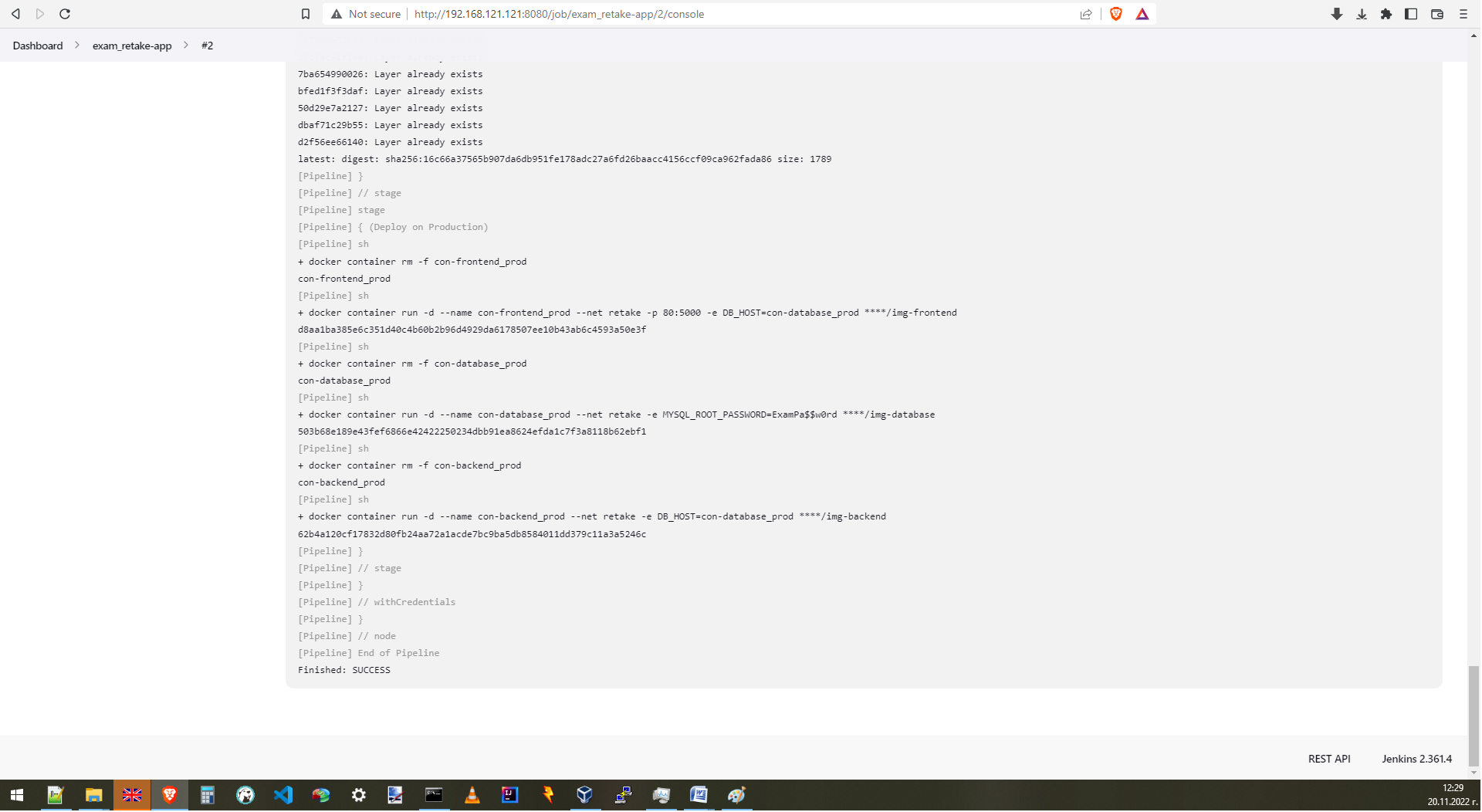


* **(T305, 2 pts)** Added slave (agent) node (the **Docker** host). *On the picture it is displayed as* ***Je (ag)***

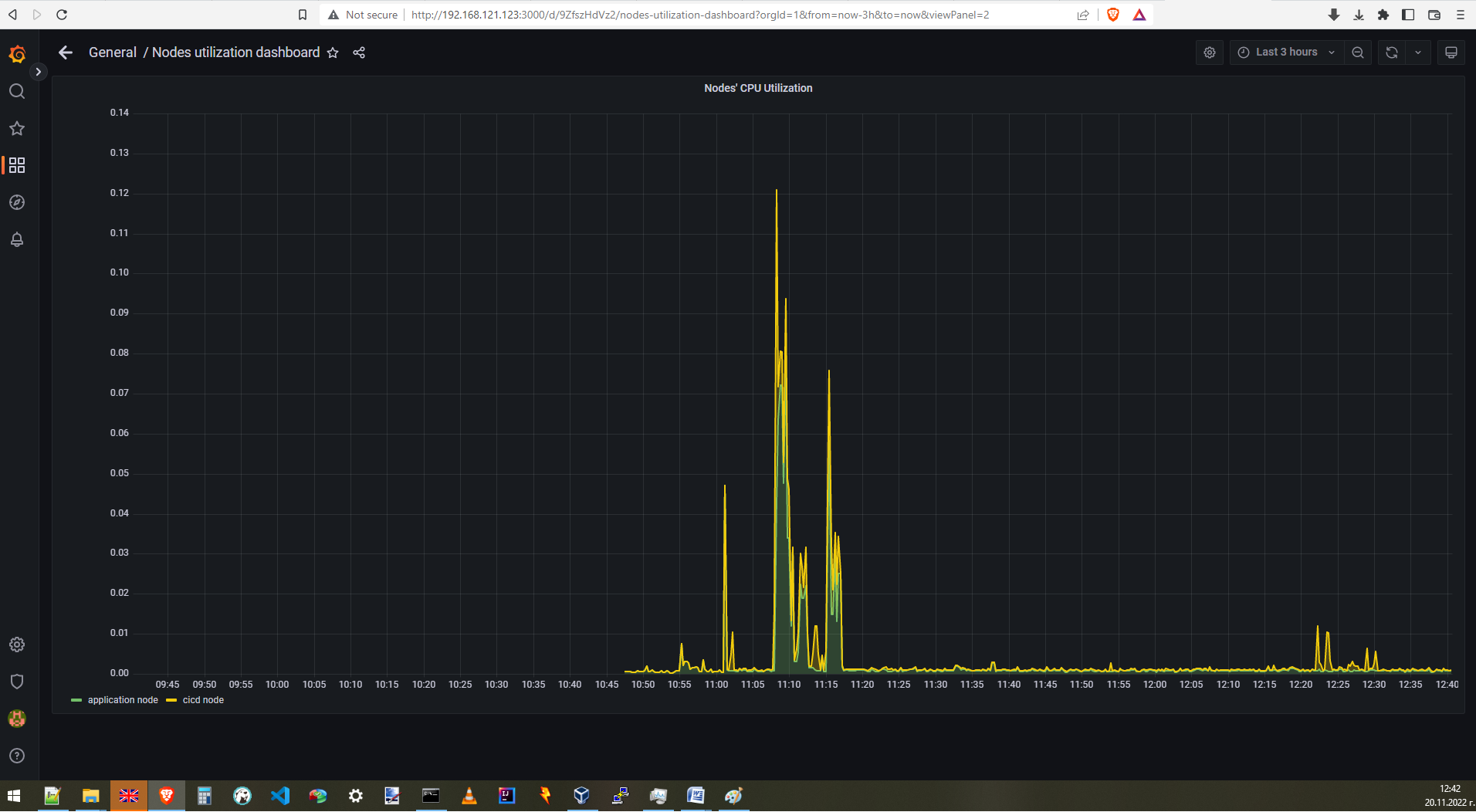


* **(T307, 1 pts)** The pipeline should be configured to be triggered via a webhook and actually executed





* **(T404, 1 pts)** One visualization that shows the CPU load of the monitored hosts, based on the collected information



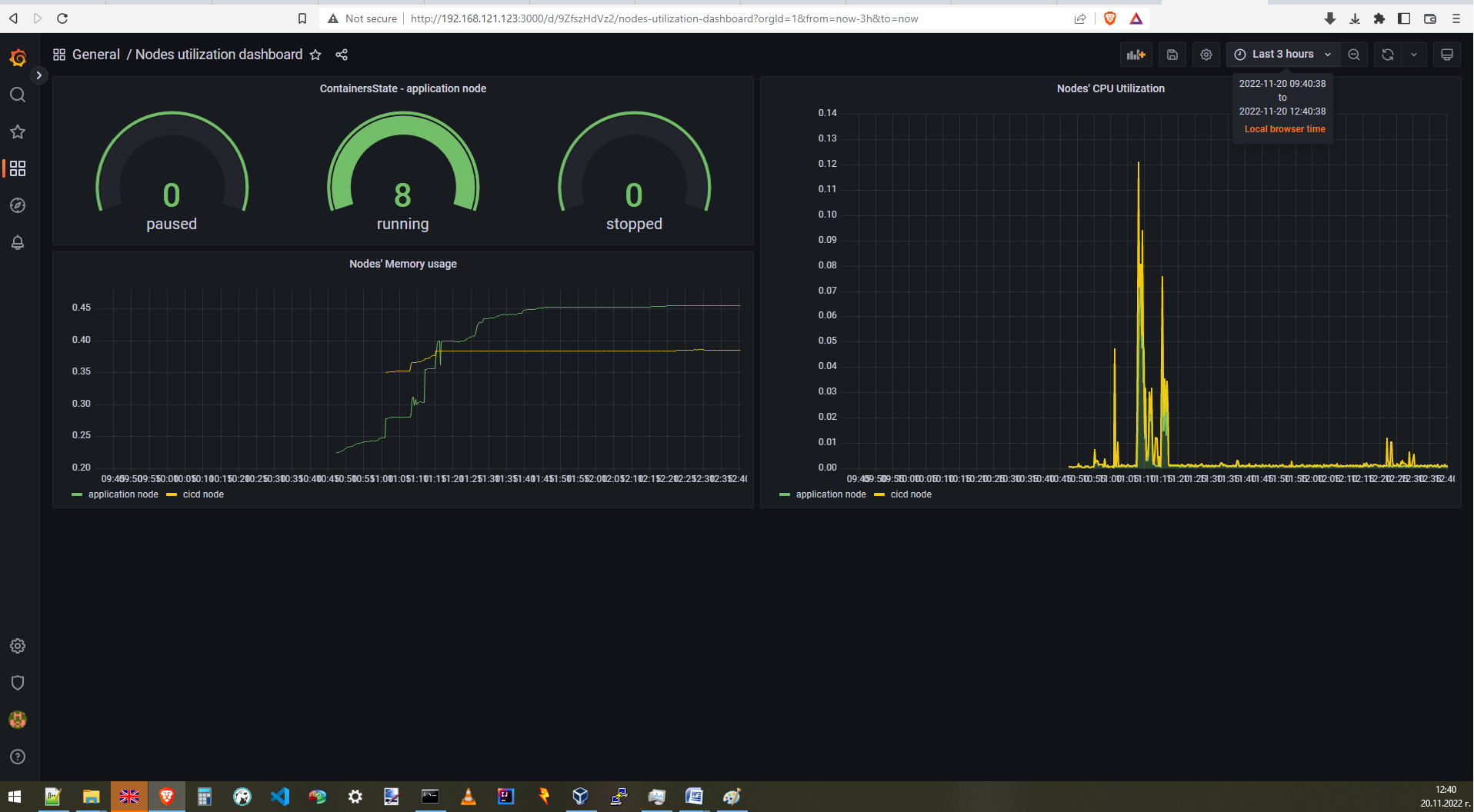
* **(T405, 1 pts)** One visualization that shows the RAM utilization of the monitored hosts, based on the collected information



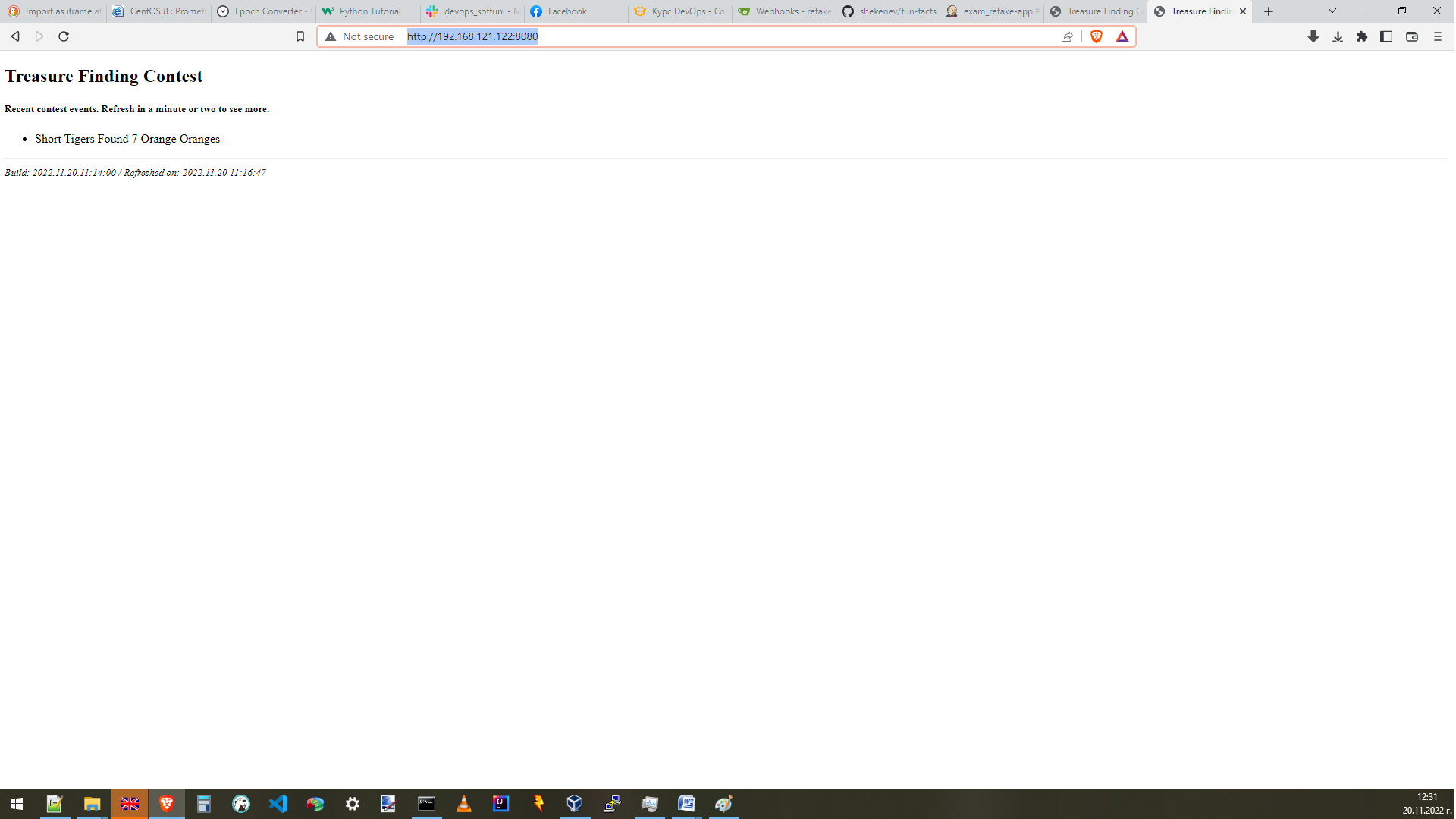
* **(T406, 1 pts)** One visualization that shows the number of containers *(any state – running, paused, etc.)* on the Docker host, based on the collected information



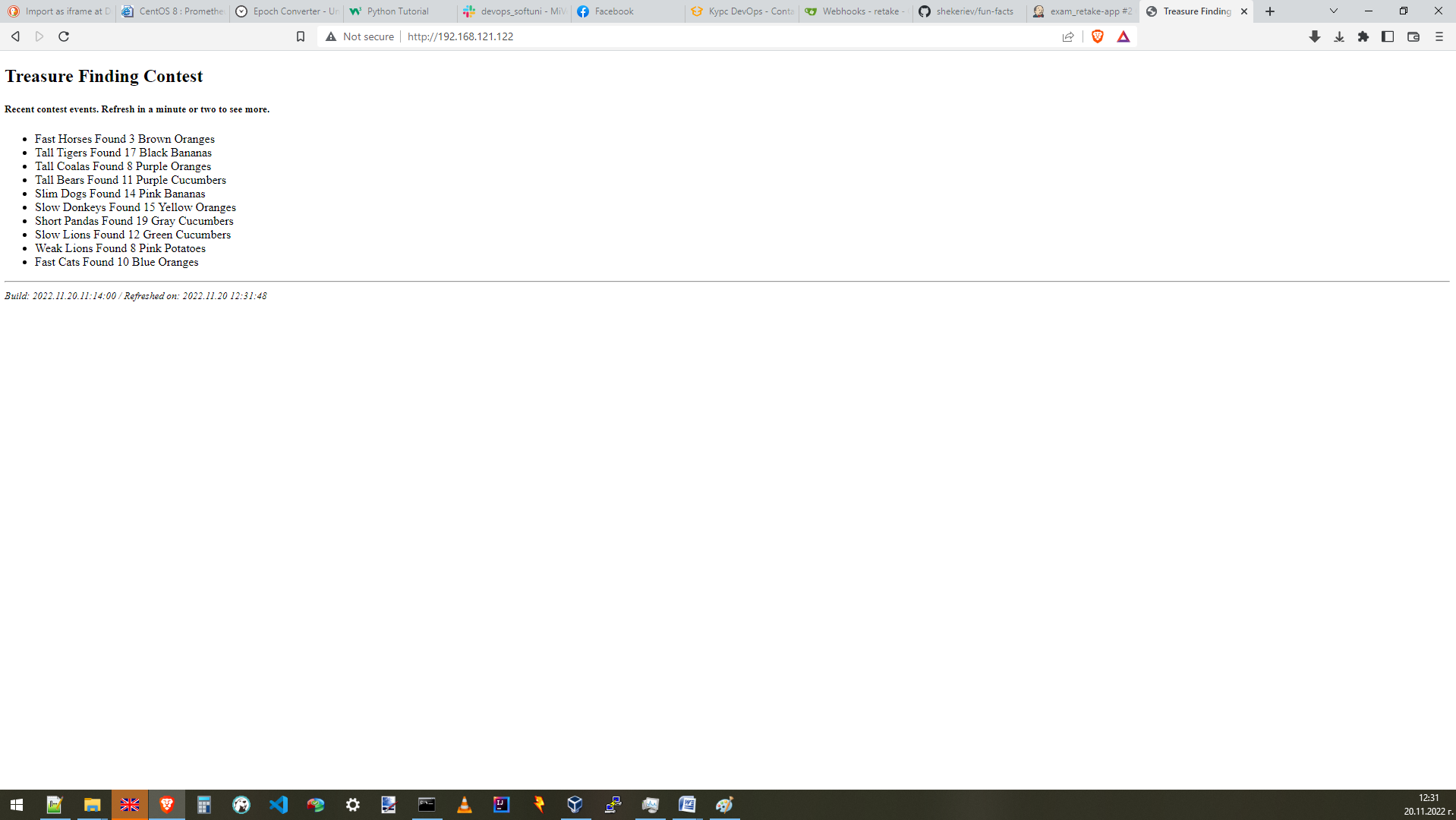
* **(T407, 2 pts)** A dashboard that includes all three visualizations



* **(T503, 3 pts)** Working containerized application in **test mode** as expected *(for a valid and expected output, check the repository you cloned earlier)*

**

* **(T504, 3 pts)** Working containerized application in **production mode** as expected *(for a valid and expected output, check the repository you cloned earlier)*



Други Srceenshost:  
