# Практика: DOB Модул 8 (автоматизиран подход)

## Задание

Двукомпонентно **Docker**-изирано приложение - **php+apache** и **mysql**.

Да се построи цялостна инфраструктура, включваща следните хостове:

* **Ansible**
* **Docker**
* **Jenkins**
* **Nagios**

Проектът за приложението е в **GutHub**:

<https://github.com/shekeriev/two-docker-images.git>

Да се прави проверка периодично (на всеки 2 мин.) за промени и при наличие на такива, да се генерират нови шаблони и от тях нови контейнери.

**Nagios** да наблюдава базови показатели като **PING** и **SSH** на всички хостове. За контейнерите - да следи дали работят.

## Възможни стъпки

Автоматизацията на провизирането и конфигурирането на хостовете ще бъде изпълнена с помощта на **Ansible** за всички хостове без хоста **ansible.sulab.local**, чието провизиране ще бъде извършено с помощта на **Vagrant**. След приключване на базовото провизиране, **ansible** хоста ще изпълни автоматично всички скриптове, за да подготви и останалата част от инфраструктурата, вкл. добавянето на **docker** хоста като подчинен.

### Подготовка

#### Host

Разархивираме изтегления архив и влизаме в съответната папка, примерно:

**cd /home/devops/DOB-Practice-Prep-Auto**

Стартираме процеса по създаване на машините и първоначалното им базово конфигуриране:

**time vagrant up**

#### Решение на поставената задача

Отваряме <http://localhost:8080/> и въвеждаме **admin** като потребителско име и парола

От главното меню избираме **New Item**

Като тип и име на проекта избираме:

* Name: **Docker-GitHub-Final**
* Type: **Freestyle Project**

Потвърждаваме с бутона **OK**

Правим следните допълнения/настройки:

* Маркираме проекта като **GitHub project**
* Като URL: <https://github.com/shekeriev/two-docker-images.git>
* Натискаме втория бутон **Advanced**, в опцията **Use custom workspace** и въвеждаме: **/vagrant/www-dynamic**
* Включваме опцията **Restrict where this project can be run** и като етикет избираме **docker**

В секцията **Source Code Management** маркираме **Git** и въвеждаме:

* <https://github.com/shekeriev/two-docker-images.git>

В секцията **Build** избираме **Execute shell** и като команди въвеждаме:

* **cd /vagrant/www-dynamic/php**
* **docker image build -t img-php .**

Отново в секцията **Build**, кликваме върху бутона **Add build step** и избираме **Execute shell**. Въвеждаме:

* **cd /vagrant/www-dynamic/mysql**
* **docker image build -t img-mysql .**

Отново в секцията **Build**, кликваме върху бутона **Add build step** и избираме **Execute shell**. Въвеждаме:

* **docker container rm -f dob-http || true**
* **docker container run -d --net dob-network -p 80:80 --name dob-http -v /vagrant/www-dynamic/site:/var/www/html img-php**

Отново в секцията **Build**, кликваме върху бутона **Add build step** и избираме **Execute shell**. Въвеждаме:

* **docker container rm -f dob-mysql || true**
* **docker container run -d --net dob-network --name dob-mysql -e MYSQL\_ROOT\_PASSWORD=12345 img-mysql**

Запазваме с бутона **Save**

Стартираме с бутона **Build Now**

Наблюдаваме процеса на изпълнение

Сега можем от опцията **Configure** в лявото меню да се върнем в настройките на проекта и да добавим проверката по график

В секцията **Build Triggers** включваме опцията **Poll SCM**

В полето **Schedule** задаваме **H/2 \* \* \* \***

Запазваме промените с бутона **Save**

*Можем да направим промяна в* ***GitHub*** *и да наблюдаваме ефекта*

#### Host (преглед на Nagios)

Отваряме сесия в браузъра на локалния хост със следния адрес:

<http://localhost:8081/nagios>

Въвеждаме потребител **nagiosadmin** и парола **Password1**

Преглеждаме наблюдаваните услуги и хостове

#### Host (преглед на docker приложението)

Отваряме сесия в браузъра на локалния хост със следния адрес:

<http://localhost:8082>

Проверяваме работещото уеб приложение