Задание:

1. Установить Nginx и настроить его на работу с PHP-FPM.

Устанавливаем компоненты:

```
sudo apt install nginx
sudo apt install php
sudo apt install php-fpm
```

Разрешаем автозапуск php-fpm и запускаем его:

```
sudo systemctl enable php8.1-fpm
```

Проверяем открытые порты: `sudo iptables -L -nv`

Открываем порты для TCP и FTP:

```
sudo iptables -I INPUT 1 -p tcp --match multiport --dports 80,443,8080 -j ACCEPT
sudo iptables -I INPUT 1 -p tcp --match multiport --dports 20,21,60000:65535 -j ACCEPT
```

Для сохранения правил ставим пакет iptables-persistent:

```
sudo apt install iptables-persistent
```

Сохраняем правила: `sudo netfilter-persistent save`

Проверяем работу nginx: `systemctl status nginx`

Проверим работу веб-сервера: `sudo curl 10.0.1.12`

Настройка связки NGINX + PHP.

Открываем файл для настройки виртуального домена по умолчанию:

```
sudo nano /etc/nginx/sites-enabled/default
```

В секции **server** редактируем параметр **index** на значение:

```
index index.php index.html index.htm;
```

Внутри секции **server** добавляем:

```
location ~ \.php$ {
    # корневой путь хранения скриптов
    set $root_path /var/www/html;

    # путь до файла сокета для взаимодействия с php-fpm
    fastcgi_pass unix:/run/php/php8.1-fpm.sock;
    fastcgi_index index.php;
    fastcgi_param SCRIPT_FILENAME $root_path$fastcgi_script_name;
    include fastcgi_params;
    fastcgi_param DOCUMENT_ROOT $root_path;
}
```

проверяем: `sudo nginx -t`, перезагружаем `sudo systemctl reload nginx`

Открываем конфигурационный файл PHP-FPM:

```
sudo nano /etc/php/8.1/fpm/pool.d/www.conf
```

Проверяем, что путь до файла сокета такой же, как мы задали в настройках NGINX:

```
listen = /run/php/php8.1-fpm.sock
```

заходим в каталог хранения настроенного сайта: `cd /var/www/html`

Создаем index.php `sudo touch index.php`

Открываем его для редактирования: `sudo nano index.php`

Вносим: <?php phpinfo(); ?>

Для проверки переходим в браузере по IP сервера: http://10.0.1.12/, видим сводную информацию по PHP – Ок.

2. Установить Apache. Настроить обработку PHP. Добиться одновременной работы с Nginx.

Устанавливаем Apache: `sudo apt install apache2`

Устанавливаем модуль php для apache `sudo apt install libapache2-mod-php`

настраиваем порты: `sudo nano /etc/apache2/ports.conf`

```
Listen 8080 # порт 80 занят nginx

#<IfModule ssl_module>
#       Listen 443 #порт слушает nginx
#</IfModule>
#<IfModule mod_gnutls.c>
#       Listen 443
#</IfModule>
```

Настраиваем обработку сначала для php в модуле

`sudo nano /etc/apache2/mods-available/dir.conf`

```
<IfModule dir_module>
    DirectoryIndex index.php index.html ...
</IfModule>
```

Настраиваем основной файл конфигурации Apache:

```
<Directory /var/www/*/www> # указывает на путь, для которого мы хотим задать настройки
    AllowOverride All # позволяет переопределить все настройки с помощью файла .htaccess
    # Options задает некоторые настройки: Indexes разрешает списки каталогов, ExecCGI разрешает запуск cgi скриптов
    Options Indexes ExecCGI FollowSymLinks
    Require all granted # предоставляет всем доступ к сайтам в данном каталоге
</Directory>
. . . #в конце добавляем:
<IfModule setenvif_module>
    SetEnvIf X-Forwarded-Proto https HTTPS=on
</IfModule>
#этой настройкой мы при получении заголовка X-Forwarded-Proto со значением https задаем
#переменную $_SERVER['HTTPS'] равную on. Данная настройки критична для функционирования некоторых CMS.
```

Проверяем конфиг: `apachectl -t` - Ok.

Запрещаем mpm_event: `a2dismod mpm_event`

(по умолчанию, apache2 может быть установлен с модулем мультипроцессовой обработки **mpm_event**. Данный модуль не поддерживает php 7 и выше)

Разрешаем модуль мультипроцессовой обработки mpm_prefork:

`sudo a2enmod mpm_prefork`

Разрешаем модуль php: `a2enmod php8.1`

Разрешаем модуль setenvif: `a2enmod setenvif`

Разрешаем модуль rewrite: `sudo a2enmod rewrite`

Разрешаем автозапуск Apache: `sudo systemctl enable apache2`

Перезапускаем Apache: `sudo systemctl restart apache2`

Для проверки переходим в браузере по IP сервера: http://10.0.1.12:8080/, видим сводную информацию по PHP, с помощью инструментов разработчика F12 убеждаемся, что сервер – Apache.

3. Настроить схему обратного прокси для Nginx (динамика - на Apache).

Меняем конфигурацию nginx:
в секции location комментируем index.php,
в секции location ~ \.php$ комментируем настройки для работы через php-fpm,
добавляем:

```
        location ~ \.php$ {
            proxy_pass http://127.0.0.1:8080;
            proxy_redirect off;
            proxy_set_header   Host $host;
            proxy_set_header   X-Forwarded-Proto $scheme;
            proxy_set_header   X-Real-IP        $remote_addr;
            proxy_set_header   X-Forwarded-For  $proxy_add_x_forwarded_for;
        }
```

проверяем конфиг: sudo nginx –t
перезапускаем nginx: `sudo systemctl reload nginx`
Для проверки переходим в браузере по IP сервера: http://10.0.1.12

4. Установить MySQL. Создать новую базу данных и таблицу в ней.

Устанавливаем бесплатный форк MySQL: `sudo apt install mariadb-server`
Включаем автозапуск `systemctl enable mariadb`
Устанавливаем пароль root `sudo mysqladmin -u root password`
Запускаем систему под рутом `mysql -uroot -p`
Создаем новую базу: `CREATE DATABASE test_data_base DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8_general_ci;`
Устанавливаем привилегии: `GRANT ALL PRIVILEGES ON test_data_base.* TO root@localhost IDENTIFIED BY 'test' WITH GRANT OPTION;`

5. * Установить пакет phpmyadmin и запустить его веб-интерфейс для управления MySQL

`sudo apt install php-mysql php-mysqli`

Создаем новый файл конфигурации для nginx:

```
server {
        listen  80;
        server_name phpmyadmin.local;
        set $root_path /usr/share/phpmyadmin;

        root $root_path;

        location / {
                index index.php;
        }

        location ~ \.php$ {
                fastcgi_pass unix:/run/php/php8.1-fpm.sock;
                fastcgi_index index.php;
                fastcgi_param SCRIPT_FILENAME $root_path$fastcgi_script_name;
                include fastcgi_params;
                fastcgi_param DOCUMENT_ROOT $root_path;
        }
}
```

Прописываем в hosts: 127.0.1.1   phpmyadmin.local
Для проверки переходим в браузере по IP сервера: http://10.0.1.12
Видим веб-интерфейс phpmyadmin – Ok.

6. * Настроить схему балансировки трафика между несколькими серверами Apache на стороне Nginx с помощью модуля ngx_http_upstream_module.
7.

Для балансировки нескольких серверов нужно прописать в файл конфигурации nginx все сервера с указанием их портов и веса, то есть приоритета обращений, вида:

```
upstream backend {
        server 127.0.0.1:8080 weight=2;
        server 127.0.0.1:8081;
```