PRODOTIME – РУКОВОДСТВО ПО УСТАНОВКЕ СЕРВЕРА

ТЕХНИЧЕСКИЕ ТРЕБОВАНИЯ

СЕРВЕРНЫЙ КОМПЬЮТЕР

Серверное приложение Prodotime On-Premises поставляется в виде Docker-контейнера — многоплатформенной виртуальной машины, поддерживаемой ОС Windows и ОС Linux. ОС Linux является предпочтительной платформой, поскольку Docker работает на ней изначально.

Список поддерживаемых версий ОС*:

• Windows Server 2019, Windows 10.
• CentOS 7 или 8
• Debian Stretch 9, Debian Buster 10
• Федора 32, Федора 33.
• Ubuntu Xenial 16.04 (LTS), Ubuntu Bionic 18.04 (LTS), Ubuntu Focal 20.04 (LTS), Ubuntu Groovy 20.10

*все поддерживаемые платформы 64-битные.

Требования к оборудованию для локального сервера

50 пользователей:
• Процессор 1 ГГц
• 2 ГБ оперативной памяти
• Жесткий диск 7200 об/мин
• Ориентировочный объем хранилища данных на 2 месяца: 2 ГБ
• Ориентировочный объем хранилища данных на 6 месяцев: 5 ГБ

500 пользователей:
• 4 ядра ЦП
• 8 ГБ оперативной памяти
• SSD, скорость чтения/записи 200 Мбит/с.

• Ориентировочный объем хранилища данных на 2 месяца: 15 ГБ
• Ориентировочный объем хранилища данных на 6 месяцев: 50 ГБ

Требования к размеру хранилища (все функции включены):
• Хранение базы данных: 20 МБ на пользователя в месяц.

*основано на средних реальных цифрах, предоставленных нашими клиентами.

**Сервер Prodotime не имеет ограничений на количество клиентских подключений, все зависит от производительности оборудования.

КЛИЕНТСКИЙ КОМПЬЮТЕР

Список поддерживаемых версий ОС:
• Windows Vista, 7,8,10, Server 2012,2016,2019 — как 32-битная, так и 64-битная версии.

Требования к оборудованию:
• Процессор 1 ГГц
• 2 ГБ оперативной памяти
• 1 ГБ свободного места на диске.
• Скорость сети 0,2 Мбит.

РАЗВЕРТЫВАНИЕ

ОС WINDOWS/LINUX

• Установите Docker: https://docs.docker.com/engine/install/.
• Для ОС Linux установите docker-compose
• Загрузите файл .yml со страницы https://prodotime.ru/docker-compose-prodotime-distr.yml

• Введите поля в файле .yml, помеченные знаком «[!!!» (примечание: для запуска веб-приложения необходим параметр PUBLIC_URL, введите IP-адрес сервера. Если сервер опубликован во внешней сети (например, в Интернете), используйте здесь внешний IP-адрес.

Это важно для правильного подключения веб-консолей. Публикация сервера и пересылка IP-адресов должны осуществляться администратором сети клиента.). Пример:

• # Версия docker-compose
version: '3.5'
Контейнеры, которые мы собираемся запустить
services:
Наш контейнер Phoenix
on_premise:
  image: registry.prodotime.ru/stingmaster/prodotime-onprem:0.7.1
Параметры сборки для этого контейнера.
  #строить:
  # # Здесь мы определяем, что сборка должна производиться из текущего каталога
  # контекст: .
environment:
Переменные для подключения к нашему серверу Postgres
PUBLIC_URL: 192.168.1.1

• Если вы не используете SMTP-сервер, используйте пробел в параметрах SMTP (SMTP_SERVER, SMTP_USERNAME, SMTP_PASSWORD).
• Запросите ЛИЦЕНЗИЮ (пробную или зарегистрированную) в ДЕМО форме на сайте https://prodotime.ru
• Введите полученное значение лицензионного ключа в параметр LICENSE: в виде одной строки.
• Сохраните изменения в файле .yml, перейдя в папку с .yml файлом, запустите контейнер Docker с помощью команды Терминала:

docker-compose -f docker-compose-prodotime-distr.yml up -d

примечание: если вы хотите проверить журнал приложения, запустите команду docker-compose logs или запустите контейнер без команды -d (он запишет журнал на терминал). Чтобы остановить приложение-контейнер, используйте Ctrl+C.

• Откройте веб-браузер и перейдите по PUBLIC_URL, используйте учетные данные для входа в информацию о лицензии.. Пароль можно изменить позже в меню «Настройки» — «Логин».

ПУБЛИКАЦИЯ ЧЕРЕЗ HTTPS

Доступ к панели управления Prodotime можно получить через HTTPS. Вы можете использовать свой сертификат или создать самозаверяющий сертификат в таком сервисе, как Let’s Encrypt. Для публикации панели мониторинга Prodotime с SSL-соединением потребуется настройка прокси-сервера. Вот примеры инструкций для NGINX.

Идея состоит в том, что вы переходите на https://prodotime.yourcompany.com, он проходит через HTTPS к NGINX на: 443, который проксирует его на http://127.0.0.1:4000 (контейнер on_premise).

Пример файла YML:

```
services:
on_premise:
 …
 environment:
 PUBLIC_URL: prodotime.yourcompany.com # домен, для которого выдан SSL-сертификат
 USE_SSL: "yes"
 PUBLIC_PORT: 443 # эта команда не привязывает контейнер к порту 443
 LIVE_STREAM_PORT: 444
 ports:
 - "4000:4000" # контейнер привязывается к 4000, поэтому NGINX должен проксировать трафик на этом порту
 - "4001:4001"
```

Пример nginx.conf:

```
map $http_upgrade $connection_upgrade {
 default upgrade;
 '' close;
}
# опубликовать основное приложение
server {
 server_name prodotime.yourcompany.com;
 listen 443 ssl http2;
 listen [::]:443 ssl http2;
 # настройки SSL
 ssl_certificate /etc/nginx/ssl/prodotime.yourcompany.com/fullchain.pem;; # путь к сертификату
 ssl_certificate_key /etc/nginx/ssl/prodotime.yourcompany.com/privkey.pem # путь к закрытому ключу
 location / {
 proxy_pass http://localhost:4000;
 proxy_http_version 1.1; # рекомендуется с соединениями поддержки активности
```

```
# https://nginx.org/en/docs/http/ngx_http_proxy_module.html#proxy_http_version
 # Проксирование WebSocket — от https://nginx.org/en/docs/http/websocket.html
 proxy_set_header Upgrade $http_upgrade;
 proxy_set_header Connection $connection_upgrade;
 proxy_read_timeout 150;
 client_max_body_size 100M;
 proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
 }
}
# опубликовать прямую трансляцию
server {
 server_name prodotime.yourcompany.com;
 listen 444 ssl http2;
 listen [::]:444 ssl http2;
 # настройки SSL
 ssl_certificate /etc/nginx/ssl/prodotime.yourcompany.com/fullchain.pem;; # путь к сертификату
 ssl_certificate_key /etc/nginx/ssl/prodotime.yourcompany.com/privkey.pem # путь к закрытому
ключу
 location / {
 proxy_pass http://localhost:4001;
 proxy_http_version 1.1; # рекомендуется с соединениями поддержки активности
# https://nginx.org/en/docs/http/ngx_http_proxy_module.html#proxy_http_version
 # Проксирование WebSocket — от https://nginx.org/en/docs/http/websocket.html
 proxy_set_header Upgrade $http_upgrade;
 proxy_set_header Connection $connection_upgrade;
 proxy_read_timeout 150;
 client_max_body_size 100M;
 proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
 }
}
```