

КАК ЗАДЕПЛОИТЬ САЙТ VITE НА VK CLOUD

КЕРБЕР ЕГОР

19 ФЕВРАЛЯ 2024 Г.

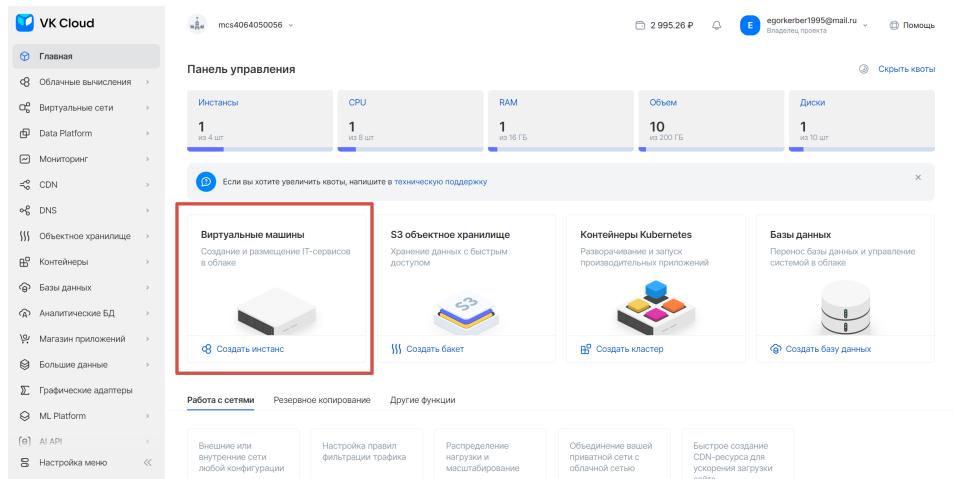
СОДЕРЖАНИЕ

1 Регистрация на vk cloud и добавление правила	3
2 Настройка nginx	7
3 Покупка домена	11
4 Конфигурация vite и docker-compose	14
4.1 Объяснение конфигурации от ChatGPT	18
5 Как зайти на сайт	20
6 Настройка https	21

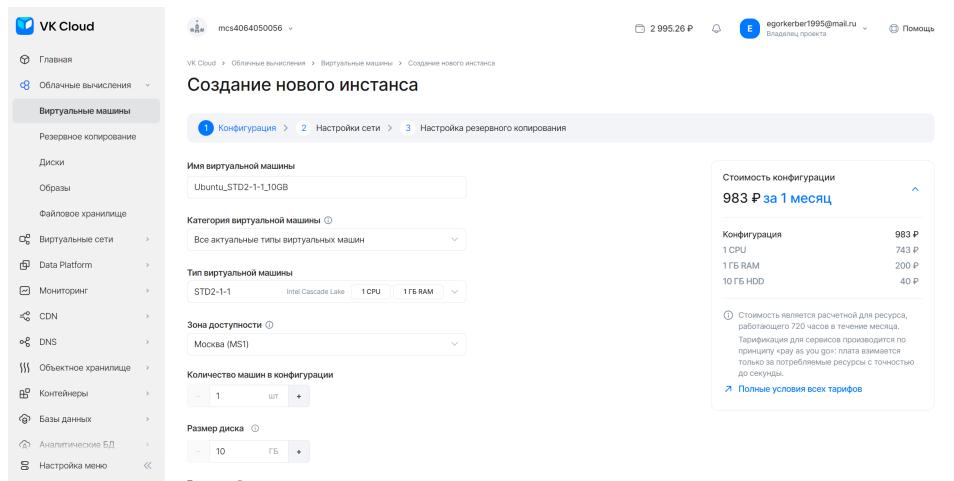
РЕГИСТРАЦИЯ НА VK CLOUD И ДОБАВЛЕНИЕ ПРАВИЛА

1. Регистрируемся на vk cloud

2. Создаем инстанс



3. Желательно выбираем Ubuntu



4. Делаем следующие шаги и все, инстанс создан

5. В разделе 'Виртуальные сети'-'Настройки firewall' добавляем новую настройку firewall

Настройки firewall

Группа	Инстансы	Кол-во правил	Дата изменения
default	1	2	17 февр. 2024, 09:06
ssh	1	1	17 февр. 2024, 09:07
web	1	2	17 февр. 2024, 09:40

6. Обзываем web

Создание группы правил

Имя группы правил : **web**

Описание

Создать группу Отменить

7. Делаем настройку

Настройки

8. Добавляем 2 новых правила

VK Cloud

mcs4064050056

VK Cloud > Cloud Networks > Настройки firewall > web

web

Входящий трафик

Тип	Протокол	Порты	Источник	Группа	Описание
IPv4	TCP	443	0.0.0.0/0	—	...
IPv4	TCP	8000	0.0.0.0/0	—	...

+ Добавить правило

Исходящий трафик

Отсутствуют правила

+ Добавить правило

Виртуальные машины с группой правил web

Новое входящее правило

Тип

HTTP

Протокол

TCP

Порты

8000

например, 8080 или 10 - 65535

Удаленный адрес

Все IP-адреса Диапазон IP-адресов Группа безопасности

Добавить описание

Сохранить правило Отменить

Новое входящее правило

Тип

HTTPS

Протокол

TCP

Порты

443

например, 8080 или 10 - 65535

Удаленный адрес

Все IP-адреса Диапазон IP-адресов Группа безопасности

Добавить описание

Сохранить правило Отменить

9. Добавляем правило к сети

Ubuntu_STD2-1-1_10GB

Сети

Имя сети	Имя подсети	IP-адрес	Доменное имя	Настройки Firewall
internet	ext-sub2 Шлюз: 212.111.87.254 CIDR: 212.111.84.0/22	212.111.84.6	- MAC-адрес: fa:16:3e:5d:8d:27	web, default, ssh ... Редактировать подключение Удалить подключение

Настройка подключения

Для правильной настройки подключения прочтите [инструкцию](#).

Все документация
Практические руководства и пошаговые инструкции по работе с VK Cloud

Диагностика VM
Управление виртуальной машиной с помощью VNC-консоли. Просмотр логов сообщений VM

Диагностика и устранение проблем
Способы диагностики и устранения проблем с виртуальными машинами

Шифрование диска
Как настроить шифрование машины с помощью

Ubuntu_STD2-1-1_10GB

Сети

Редактирование подключения

Имя:

Сеть для подключения: Внешняя сеть (internet)

Назначить внешний IP:

Настройки Firewall: default x ssh x web

Сохранить **Отмена**

Это правило нужно для того, чтобы сайт вскоре был доступен

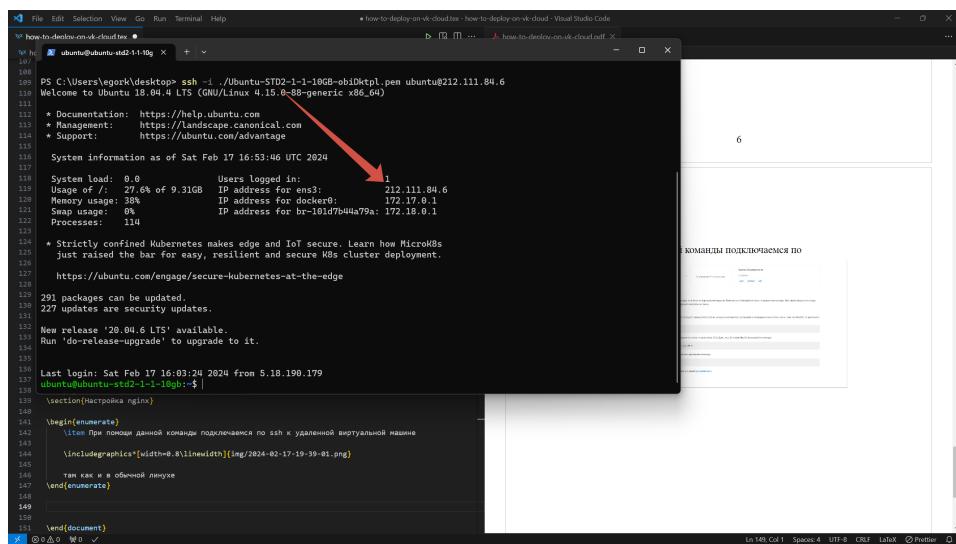
по <http://212.111.84.6:8000/>

1. При помощи данной команды подключаемся по ssh к удаленной виртуальной машине

The screenshot shows the VK Cloud interface under the 'Virtual machines' section. It displays the external IP address (212.111.84.6) and security group information (3 groups: web, default, ssh). Below this, a 'Connection to VM' section provides instructions for SSH access:

- On the previous step, you selected a key for the virtual machine. You need to find the file on your local computer where the key was saved. This file will be used to log in.
- Copy the path to this file.
- Only the root user has full privileges. If your system is Linux, Unix or MacOS, run the command: chmod 400 <path to key>
- First connection to the instance is possible only via the SSH protocol. For Linux, Unix or MacOS, use the command: ssh -i <path to key> ubuntu@212.111.84.6
- To get root privileges, run sudo bash

По следующему адресу кстати будет доступен сайт



Там как и в обычной линуке делаем sudo bash, чтобы удобнее было.

Потом клонируем репозиторий, запускаем контейнер при помощи docker-compose up -d

(-d чтобы работа терминала не зависала только на этом процессе)

далее по необходимости делаем docker exec -it vite_docker sh

чтобы войти в контейнер и выполнить команды npm i и потом npm run dev

Все, сервер поднялся, теперь конфигурируем сам nginx.

2. В папке conf.d делаем конфигурацию при помощи команды nano

/etc/nginx/conf.d/resume.conf



```
ubuntu@ubuntu-std2-1-1-10g: /etc/nginx/conf.d
```

```
GNU nano 2.9.3 /etc/nginx/conf.d/resume.conf
```

```
server {
    listen 80;
    server_name egorkerber.ru; # или ваш домен, если он есть

    location / {
        proxy_pass http://localhost:8000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_redirect off;
    }
}
```

[File '/etc/nginx/conf.d/resume.conf' is unwritable]

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo ^A Mark Text
^X Exit ^R Read File ^N Replace ^U Uncut Text ^T To Spell ^L Go To Line M-E Redo M-G Copy Text

3. Делаем более общую конфигурацию nano /etc/nginx/nginx.conf

```
1 user www-data;
2
3 worker_processes auto;
4
5 pid /run/nginx.pid;
6
7 include /etc/nginx/modules-enabled/*.conf;
8
9
10
11 events {
12     worker_connections 768;
13     # multi_accept on;
14 }
15
16
17 http {
18     ##
19     # Basic Settings
20     ##
21
22     sendfile on;
23     tcp_nopush on;
```

```
18     tcp_nodelay on;
19
20     keepalive_timeout 65;
21
22     types_hash_max_size 2048;
23
24
25     ##
26
27     # SSL Settings
28
29     ##
30
31
32     # Logging Settings
33
34     ##
35
36     access_log /var/log/nginx/access.log;
37     error_log /var/log/nginx/error.log;
38
39     ##
40
41     # Gzip Settings
42
43     ##
44
45     gzip on;
```

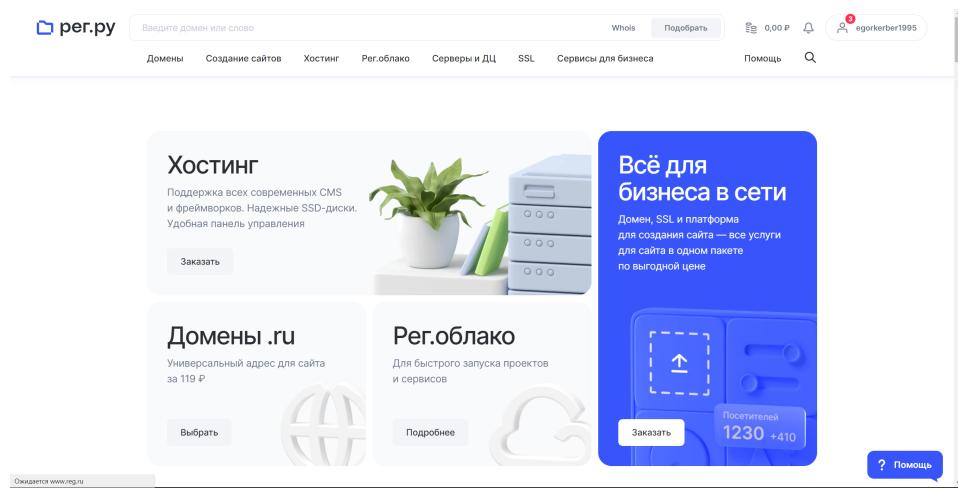
```
45     ##
46     # Virtual Host Configs
47     ##
48
49     include /etc/nginx/conf.d/*.conf;
50     include /etc/nginx/sites-enabled/*;
51 }
52
```

4. Выполняем команду systemctl restart nginx чтобы запустить nginx

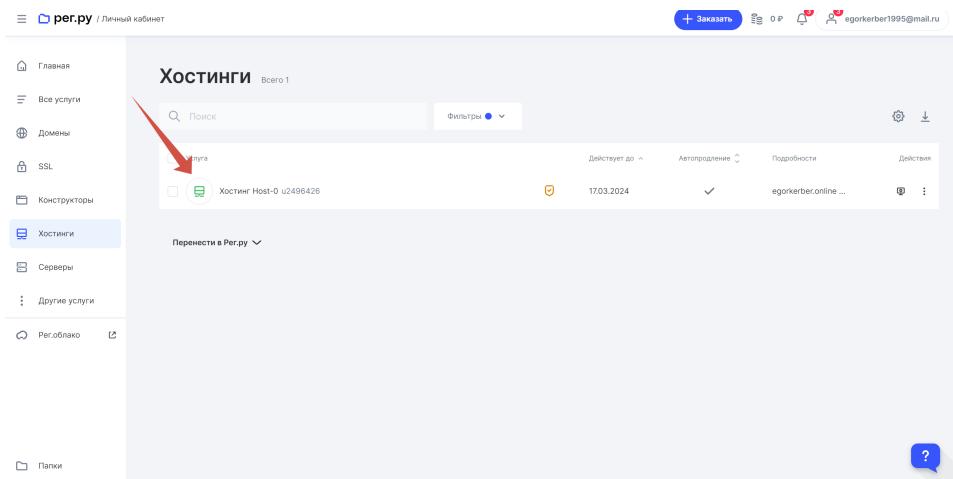
3

ПОКУПКА ДОМЕНА

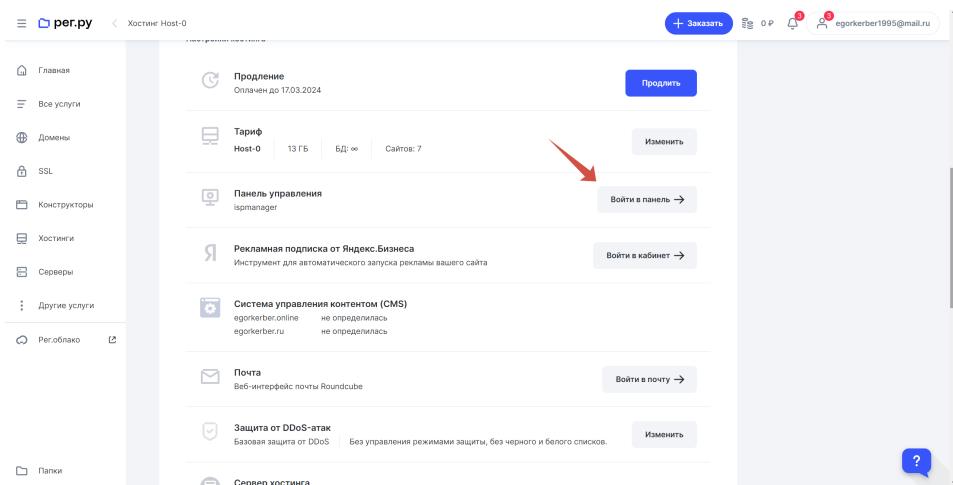
1. Заходим на рег.ру, регистрируемся и т.п. и покупаем домен



2. Потом заходим в личный кабинет, хостинги



3. Заходим в нужный хостинг, мотаем вниз, заходим в панель управления



4. Заходим в управление DNS

The screenshot shows the ispmanager dashboard with the 'Управление DNS' (DNS Management) section selected. On the left sidebar, there is a red arrow pointing to the 'Управление DNS' link under the 'DNS Management' heading. The main area displays a 'Дашборд' (Dashboard) with sections for 'Использование диска' (Disk Usage) and 'Журнал посещений' (Access Log). To the right, there is a 'Ограничения' (Limits) table showing resource usage limits.

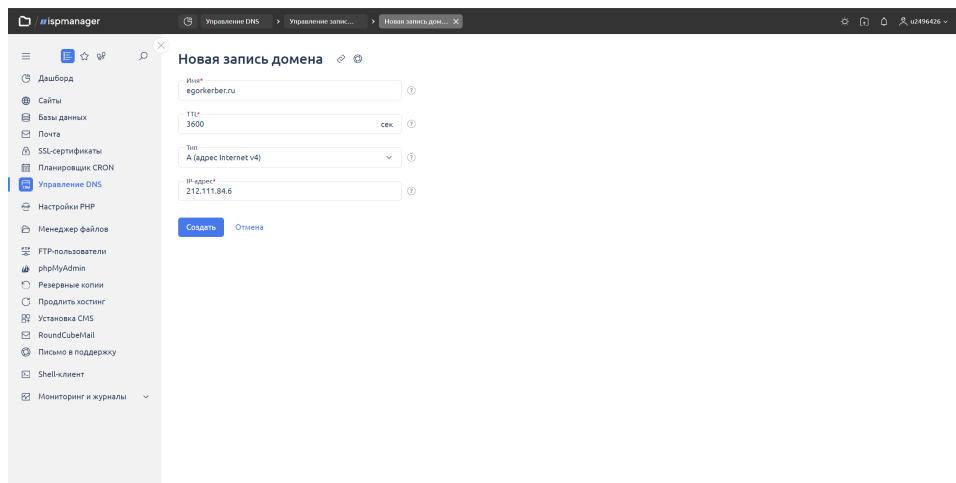
5. Управлять DNS записями

The screenshot shows the 'Управление DNS' (DNS Management) page. A red arrow points to the 'Управление DNS записями' (Manage DNS records) button in the top right corner of the main content area. The page lists domain records for 'egorkerber.online' and 'egorkerber.ru'. For 'egorkerber.ru', the 'Изменить' (Edit) and 'Управление DNS записями' (Manage DNS records) buttons are highlighted.

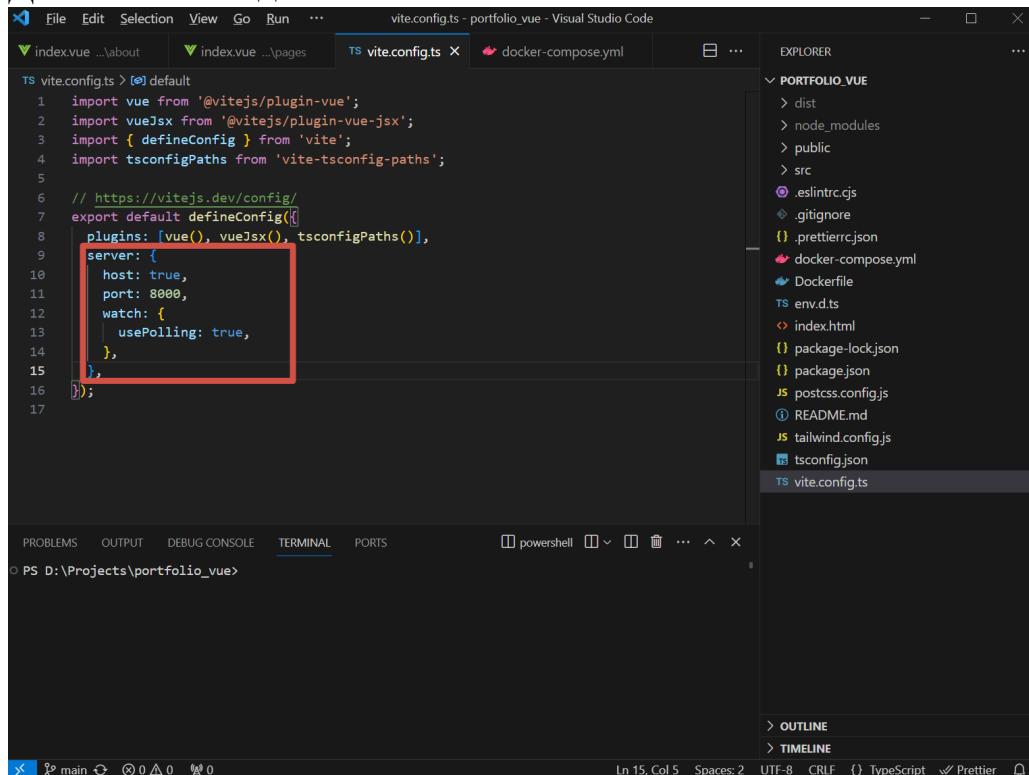
6. Создаем запись

The screenshot shows the 'Управление записями домена - egorkerber.ru' (Manage domain records - egorkerber.ru) page. A red arrow points to the 'Создать запись' (Create record) button at the top left. The main area displays a table of existing DNS records for the 'egorkerber.ru' domain, including entries for A, AAAA, MX, and NS records.

7. Делаем так



Для vite важно добавить это поле



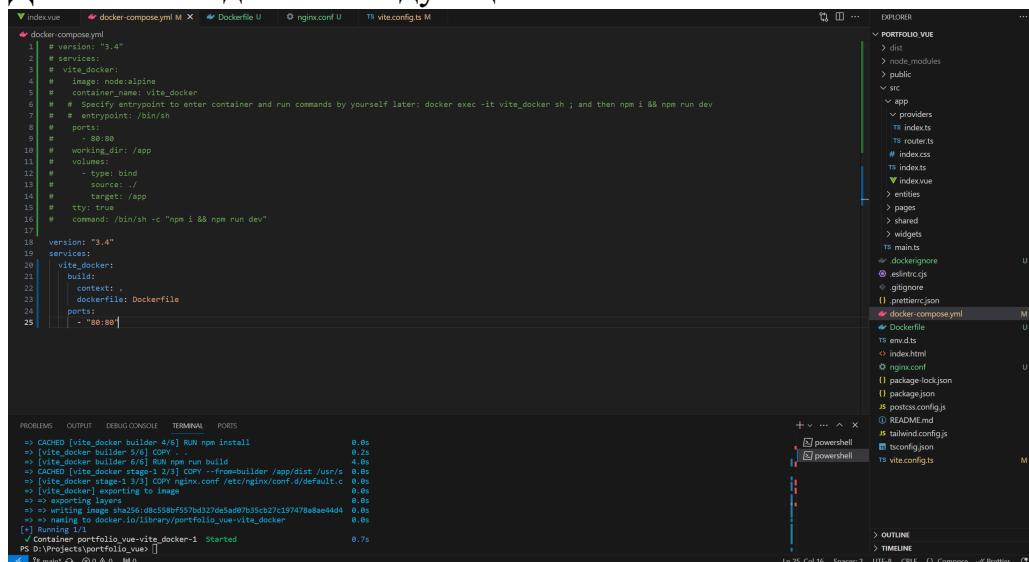
```

1 import vue from '@vitejs/plugin-vue';
2 import vueJsx from '@vitejs/plugin-vue-jsx';
3 import { defineConfig } from 'vite';
4 import tsconfigPaths from 'vite-tsconfig-paths';
5
6 // https://vitejs.dev/config/
7 export default defineConfig({
8   plugins: [vue(), vueJsx(), tsconfigPaths()],
9   server: {
10     host: true,
11     port: 8000,
12     watch: {
13       usePolling: true,
14     },
15   },
16 });
17

```

В порте указываем внешний порт (watch usePolling добавил на всякий случай, нужен вроде чтобы hot reload работал или что-то такое)

Для docker-а делаем следующее



```

version: "3.4"
services:
  vite_docker:
    build:
      context: .
    ports:
      - "80:80"

```

(закомментированные настройки - для запуска в режиме разработки)

```

1 # version: "3.4"
2 # services:

```

```
3 # vite_docker:
4 #   image: node:alpine
5 #   container_name: vite_docker
6 #   # Specify entrypoint to enter container and run commands by
7 #   # yourself later: docker exec -it vite_docker sh ; and then npm
8 #   # i && npm run dev
9 #
10 #   # entrypoint: /bin/sh
11 #
12 #   ports:
13 #     - 80:80
14 #
15 #   working_dir: /app
16 #
17 #   volumes:
18 #     - type: bind
19 #       source: ./
20 #       target: /app
21 #
22 #   tty: true
23 #
24 #   command: /bin/sh -c "npm i && npm run dev"
25 #
26
27
28 version: "3.4"
29
30 services:
31 vite_docker:
32   build:
33   context: .
34   dockerfile: Dockerfile
35   ports:
36   - "80:80"
```

Чтобы запустить конфигурацию для продакшна нужна команда
docker-compose up –build -d
Dockerfile:

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure for "PORTFOLIO_VUE". It includes files like Dockerfile, docker-compose.yml, index.html, nginx.conf, package-lock.json, package.json, postcss.config.js, README.md, tailwind.config.js, vite.config.ts, and vite.ts.
- Dockerfile:** The content of the Dockerfile is displayed in the main editor tab.
- Terminal:** The terminal shows the command `docker build -t portfolio_vue-vite_docker .` being run, and the output indicates the build process has started.

```

1 # Stage 1: create NPM packages
2 FROM node:alpine AS builder
3 WORKDIR /app
4 COPY package*.json .
5 RUN npm install
6 COPY . .
7 RUN npm run build
8
9 # Stage 2: copy Nginx c production build
10 FROM nginx:alpine
11 COPY --from=builder /app/dist /usr/share/nginx/html
12 COPY nginx.conf /etc/nginx/conf.d/default.conf
13 CMD ["nginx", "-g", "daemon off;"]

```

```

1 FROM node:alpine AS builder
2 WORKDIR /app
3 COPY package*.json .
4 RUN npm install
5 COPY . .
6 RUN npm run build
7
8 FROM nginx:alpine
9 COPY --from=builder /app/dist /usr/share/nginx/html
10 COPY nginx.conf /etc/nginx/conf.d/default.conf
11 CMD ["nginx", "-g", "daemon off;"]

```

Nginx:

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure for "PORTFOLIO_VUE". It includes:
 - src
 - node_modules
 - public
 - src
 - app
 - providers
 - index.ts
 - router.ts
 - # index.css
 - index.ts
 - index.vue
 - entities
 - pages
 - shared
 - widgets
 - main.ts
 - .dockerignore
 - eslintconfig.js
 - gitignore
 - postcss.config.js
 - README.md
 - tailwind.config.js
 - vite.config.ts
- Code Editor (Top):** Displays the contents of the "nginx.conf" file.
- Terminal (Bottom):** Shows the output of a Docker build command, indicating success with a status of "Started" and a timestamp of "0.7s".

```

1 server {
2
3     listen 80;
4
5     server_name localhost;
6
7     location / {
8
9         root    /usr/share/nginx/html;
10        index   index.html index.htm;
11        try_files $uri $uri/ /index.html;
12    }
13 }

```

```

1 server {
2
3     listen 80;
4
5     server_name localhost;
6
7     location / {
8
9         root    /usr/share/nginx/html;
10        index   index.html index.htm;
11        try_files $uri $uri/ /index.html;
12    }
13 }

```

vite_config:

```

version: "3.4"
services:
  vite_docker:
    image: node:alpine
    container_name: vite_docker
    entrypoint: /bin/sh
    ports:
      - 80:80
    environment:
      - VITE_APP_NAME=portfolio_vue
      - VITE_APP_PORT=80
    volumes:
      - ./src:/app
      - ./node_modules:/node_modules
      - ./public:/public
    command: npm run dev

```

4.1 Объяснение конфигурации от ChatGPT

Этот файл является файлом конфигурации Docker Compose, который используется для определения и настройки многоконтейнерных приложений. Давайте разберем каждую часть этого файла:

version: "3.4": Это указывает на версию формата файла Docker Compose, которую следует использовать при интерпретации этого файла. В данном случае используется версия 3.4.

services:: Этот ключевой раздел определяет список всех сервисов, которые будут запущены при помощи Docker Compose.

vite_docker:: Это имя сервиса, которое мы определяем. Здесь vite_docker - это произвольное имя, которое мы дали нашему сервису.

image: node:alpine: Это образ Docker, который будет использоваться для запуска нашего сервиса. Здесь мы используем образ node:alpine, основанный на Alpine Linux, который содержит Node.js.

container_name: vite_docker: Это имя контейнера, которое будет присвоено контейнеру, когда он будет запущен.

entrypoint: /bin/sh: Это команда, которая будет выполнена при запуске контейнера. Здесь мы указываем запуск интерактивной оболочки sh в контейнере.

`ports::` Этот параметр определяет порты, которые будут проброшены из контейнера на хостовую машину. В данном случае, порт 8000 контейнера будет проброшен на порт 8000 хостовой машины.

`working_dir: /app:` Это рабочий каталог внутри контейнера, к которому будет сделано изменение после запуска контейнера.

`volumes::` Этот параметр позволяет примонтировать локальные файловые системы или тома внутрь контейнера. Здесь используется тип `bind`, что означает привязку (`mount`) каталога или файла с хостовой машины к каталогу в контейнере.

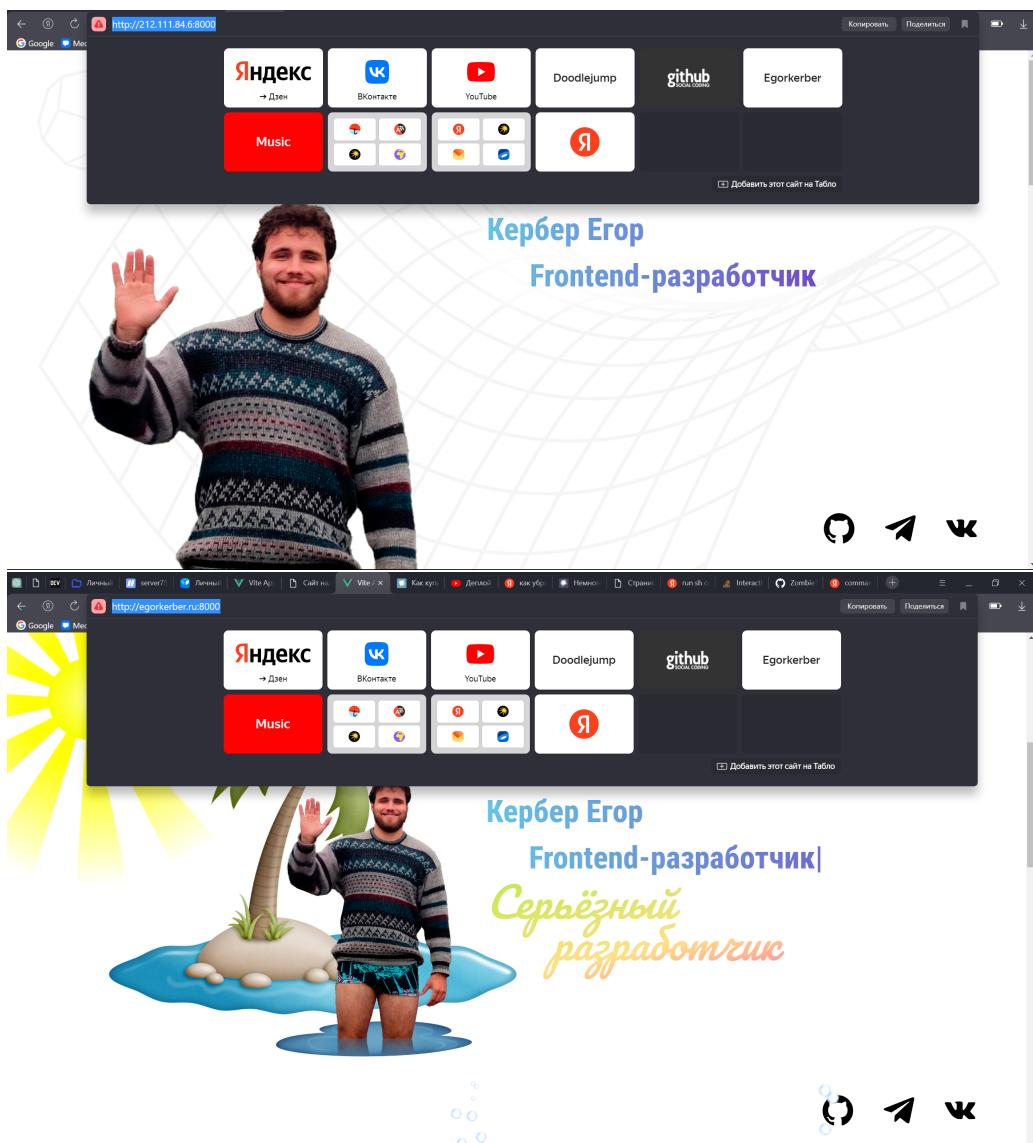
`type: bind:` Это тип тома, который мы используем для монтирования файловой системы хоста.

`source: ./`: Это путь к каталогу или файлу на хостовой машине, который будет примонтирован в контейнер.

`target: /app:` Это место в контейнере, куда будет примонтирован каталог или файл с хостовой машины.

`tty: true:` Этот параметр устанавливает взаимодействие с терминалом (TTY) внутри контейнера, что обеспечивает возможность использования интерактивной оболочки внутри контейнера.

Этот файл Docker Compose определяет контейнер, который будет запущен с образом Node.js, пробрасывает порт 8000, монтирует текущий рабочий каталог хостовой машины внутрь контейнера и запускает интерактивную оболочку sh внутри контейнера.



Чтобы настроить https, заходим в панель хостинга

The screenshot shows the ISPmanager control panel under the 'SSL-сертификаты' section. There are two certificates listed:

- egorkerber.online**: Valid until 2025-02-16, Self-signed, Used.
- egorkerber.ru**: Valid until 2024-09-18, Existing, Used.

Копируем ssl-сертификат и его ключ

The screenshot shows the 'Информация - egorkerber.ru' page. It displays the SSL certificate and private key details:

```

-----BEGIN CERTIFICATE-----
MIIB0DCB9sgAwbAgkMDyQckKs2pPAHxTMfMA0GCSeGSib3DQEBCwUAJMFmxCzAJ
BgNVBAgTAjJFMRkwFwYDQVQKEBhBGvRYWxTAwIuI52LXNlMSkwJwDVQODyBH
bG9YVwTxTwduUdQyQbSMwBEVbUTfMgQ0EgMjAyIDAfFvdyIDAyItTcxITU0Mzd
FwOyIDASMTgxMTU0MzdahBwxGjAYBgNvBAMTExd3dyS1Z29ya2VvYmYvJnJzTMC
-----BEGIN RSA PRIVATE KEY-----
MIUKAIBAAKCaGEauJlyBnTe5Ch7gygtcBHMQUCLasutUpDfFAMs2z4h9MTSB
qrqfTsqPZD/EffPxwCtLvoZERymPPW0oYT5yKc074cwUlJVDWDKuEcw
y3dEfzmv1YkOYCYCZWfCFLaL2f5zbJocRfrafHuiuM6a2zWJS8clscsEJCPfB
OTAzHtgwMDAwMDBaMFmxCzAjBgNvBAYTAkJFMRkwFwYDVQQKExBhG9YVwTxWdu
-----BEGIN SUBJECT-----
Имя (CN)
www.egorkerber.ru
-----BEGIN ISSUER-----
Код страны
-----END ISSUER-----

```

На сервер закидываем в папку /certs egorkerber.crt и egorkerber.key, потом добавляем настройки в docker-compose и nginx

docker-compose.yml

```

1 version: "3.4"
2 services:
3   vite_docker:
4     build:
5       context: .
6     dockerfile: Dockerfile

```

```
7   ports:
8     - 80:80
9     - 443:443
10    volumes:
11      - /certs:/certs
```

nginx.conf

```
1 server {
2   listen 80;
3   listen [::]:80;
4   server_name egorkerber.ru www.egorkerber.ru;
5
6   location / {
7     root /usr/share/nginx/html;
8     index index.html index.htm;
9     try_files $uri $uri/ /index.html;
10  }
11
12  # return 301 https://$server_name$request_uri;
13 }
14
15 server {
16   listen 443 ssl;
17   listen [::]:443 ssl;
18   server_name egorkerber.ru www.egorkerber.ru;
19
20   http2 on;
21
22   ssl_certificate /certs/egorkerber.crt;
```

```
23     ssl_certificate_key /certs/egorkerber.key;
24     ssl_protocols TLSv1.2 TLSv1.3;
25     ssl_prefer_server_ciphers on;
26     ssl_ciphers "EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH"
27     ;
28     ssl_session_cache shared:SSL:10m;
29     ssl_session_tickets off;
30
31     location / {
32         root    /usr/share/nginx/html;
33         index  index.html index.htm;
34         try_files $uri $uri/ /index.html;
35     }
```

Ссылки

<https://dev.to/ysmnikhil/how-to-build-with-react-or-vue-with-vite-and-docker-1a3l>