

# Проектная работа. Докеризация и деплой сервиса “КупиПодариДай”

В этой проектной работе вам предстоит докеризировать сервис КупиПодариДай. Он состоит из трех частей: бэкенд-сервиса на Node.js, БД (PostgreSQL) и фронтенда на React (раздаётся nginx), а затем развернуть его в Яндекс Облаке.

## Добавьте код приложений в репозиторий

Стартовый код проекта доступен вам в репозитории [web-plus-docker-and-compose](#). Добавьте в него исходный код вашего бэкенда и фронтенда (соответствующие папки есть в репозитории)

## Установите на сервер необходимое ПО

Перед тем, как приступить к выполнению основной части проектной работы, установите на сервере Docker и Docker Compose. Как это сделать мы рассказывали в [этом](#) уроке. Проверьте себя: команды `sudo docker version` и `sudo docker-compose version` должны выполняться без ошибок.

## Подготовьте бэкенд к докеризации

Для этого измените его код так, чтобы JWT-секрет и параметры подключения БД загружались из переменных окружения. Вам потребуются переменные: `JWT_SECRET`, `POSTGRES_PASSWORD`, `POSTGRES_DB`, `POSTGRES_USER`, `POSTGRES_HOST`, `POSTGRES_PGDATA` (директория с данными БД, она должна храниться в разделе “volume” docker или монтироваться на сервер из контейнера).

## Докеризируйте бэкенд

В нём опишите сценарий сборки образа бэкенда в Dockerfile. Собранный образ должен удовлетворять следующим критериям:

- Рабочая директория (WORKDIR) — /app.
- В качестве базового образа используется `node:16-alpine`.
- Сборка осуществляется в два этапа (multi-stage build).  
На первом осуществляется сборка проекта, а на втором — запуск билда в окружении с минимальным набором npm-зависимостей.  
Для установки зависимостей на каждом из этапов используется `npm`. Установка зависимостей осуществляется при помощи команды `npm i` или `npm ci`.  
Обратите внимание: в финальном образе не должно быть исходников (директории `src`), а также `dev`-зависимостей и необходимых конфигов.
- Зависимости проекта корректно кэшируются. Проверьте себя: соберите образ, затем измените номер порта и запустите сборку повторно. Если Dockerfile описан корректно, этап установки зависимостей будет пропущен с пометкой `CACHED`.

```
> docker build . --tag kupipoda
[+] Building 1.0s (14/14) FINISH
...
=> CACHED [builder 4/6] RUN npm
...
=> => naming to docker.io/libr
```

- Запуск проекта осуществляется при помощи pm2.

Для начала добавьте в Dockerfile инструкцию, устанавливающую пакет pm2 глобально при помощи npm.

Для запуска проекта воспользуйтесь pm2 и командой pm2-runtime (специальная команда pm2 для запуска в докере). Конфигурацию запуска опишите в файле экосистемы.

- В Dockerfile указан порт, который по умолчанию использует ("экспозит") ваш сервис.

## Зарегистрируйте домены

Воспользуйтесь нашим сервисом регистрации [доменов](#) и зарегистрируйте 2 новых домена - для фронта и бэка. Используйте для этого раздел "Домены для диплома"

Домены для проекта

Фронтенд

Бэкенд

Домены для диплома

Фронтенд

Бэкенд

Домен для фронтенда дипломного пр

Создав домен, вы не сможете поменять доменное

Создать

Бэкенд разместите на поддомене `api.` доменного имени фронтенда (`api.${your_domain_here}.${zone}`).

## Настройте домены

Для этого вам потребуется написать 2 конфигурации для бэкенда и фронтенда. Для бэкенда настройка производится идентично тому, как вы делали это в предыдущей проектной, а вот для фронтенда будут отличия. Вместо раздачи статики с диска вам потребуется проксировать запросы на порт контейнера так же, как вы раньше делали для бэкенда.

Затем выпустите и настройте SSL-сертификаты при помощи `certbot`, проверьте доступность и корректную работу ваших сервисов по HTTPS.

## Докеризируйте фронтенд

В репозитории проекта есть директория `frontend`. Вот что вам нужно сделать:

1. добавить во `frontend` исходный код фронтенда «КупиПодариДай»,
2. создать в ней поддиректорию `nginx/conf.d` и добавить туда файл `default.conf` со следующим содержимым:

Скопировать код

```
server {  
    listen      80;  
    server_name localhost;  
  
    location / {  
        root    /usr/share/nginx/html;
```

```

        index index.html index.htm;

        # Исправляем роутинг на фронтенде
        try_files $uri $uri/ /index.html;
    }
}

```

Затем в исходном коде проекта измените адрес бэкенд-сервиса на домен для бэкенда, который вы зарегистрировали.

После этого в `Dockerfile` опишите сборку фронтенда в два этапа: для первого потребуется базовый образ `node:16-alpine`, а для второго — `nginx:1.23.1-alpine`. На втором этапе скопируйте билд фронтенда в образ с `nginx`, а также конфиг, который мы добавили выше. Конфиг поместите по пути `/etc/nginx/conf.d` внутри контейнера.

## Опишите файл `docker-compose`

Осталось собрать все три части воедино в файле `docker-compose.yml`.

Опишите в нём три сервиса: `backend`, `frontend`, `database`. Для каждого из них необходимо задать имя контейнера, директорию сборки (контекст). Не забудьте про переменные окружения — их необходимо загружать из одного или нескольких файлов, которые не должны храниться в репозитории.

При этом бэкенд должен быть доступен на порту 4000, `nginx` с фронтендом на 8081.

Используйте для этого директивы раздела `ports` для каждого сервиса.

Порт базы данных должен быть доступен только во внутренней сети и только контейнеру с бэкендом.

Потом добавьте файлы с примерами заполнения `env`. Например, для файла `.env` это будет `.env.example`. В `env.example` приведите примеры значений для каждой переменной. При этом некоторые значения не должны совпадать с настоящими (которые указаны в `.env`). Вот список переменных, значения которых должны отличаться:

- `POSTGRES_USER` (имя пользователя PostgreSQL);
- `JWT_SECRET` (JWT-секрет);
- `POSTGRES_PASSWORD` (пароль пользователя БД);
- `POSTGRES_DB` (имя БД).

Проверьте себя: после выполнения команды `docker-compose up` сервисы запускаются без ошибок, по адресу <http://localhost:8081> открывается фронтенд, а по <http://localhost:4000> — бэкенд, базовая функциональность бэкенда работает корректно.

## Развернуть приложение на сервере

Склонируйте репозиторий на сервер в Яндекс Облаке и запустите ваш сервис при помощи команды `docker-compose up -d`.

Проверьте логи контейнеров и убедитесь, что они запустились без ошибок.

## Расскажите, как найти ваш сервер

Добавьте публичный IP-адрес сервера и домен, по которому к нему можно обратиться, в файл `README.md`.

Для этого воспользуйтесь этим шаблоном:

- IP адрес `x.x.x.x`

- Frontend <https://...>
- Backend <https://...>

После этого сделайте репозиторий публичным.

## **Чек-лист**

Перед отправкой обязательно проверьте работу по чек-листу:

<https://code.s3.yandex.net/web-developer/checklists-pdf/web-plus/checklist-24.pdf>