В каталоге проекта создаем файл Dockerfile, там примерно такое содержимое:

FROM node:18-alpine as builder

WORKDIR /app

COPY package.json .

ENV APP\_PORT=3000

RUN npm install

COPY . .

EXPOSE 3000

CMD ["node", "server.js"]

Создаем файл .dockerignore в каталоге проекта, туда пишем (это не обязательно впрочем):

node\_modules

npm-debug.log

Создаем образ (про точку в середине команды не забыть!):

docker build . -t <наименование образа в докер-хабе>,

например:

docker build . -t kozlovas/mydockerimage

Если ошибок нет – проверяем, должен появиться образ в каталоге докера или командой – docker images “kozlovas/\*” <- будут отображены все мной развернутые образы репозитория.

Или так еще можно:

doker images “kozlovas/graph\*” <- будут отфильтрованы все образы, начинающиеся с “kozlovas/graph”

Пушим образ в DockerHub:

Docker push kozlovas/graphql\_monolit

Если будет ошибка – access denied, то вводим:

docker login

затем логин – kozlovas

пароль – gjkjcfnbrA1$

Делаем файл деплоймента.

Особенности:

- строка подключения к БД – MongoDb – указываем наименование сервиса, который работает с подом MongoDb, в нашем случае – это mongodb-service и порт – внутренний порт Kubernetes, в нашем случае – это стандартный порт MongoDb.

- образ приложения – что сделали ранее.

- порт – можно было оставить и 80, это внутренний порт, но можно и 3000 – какой и был при разработке.

Деплоим контейнер в кубик:

kubectl apply -f graphql\_monolit.yaml

Проверяем – kubectl get pods – должен быть рабочим контейнер graphql\_monolit

Правим ingress-сервис – прописываем новое правило:

- path: /gql/monolit/?(.\*)

pathType: Prefix

backend:

service:

name: graphql-monolit-service

port:

number: 3000

Получаем url для работы через ingress-сервис – <http://localhost:8080/gql/monolit>

**Важная особенность!**

Ingress-сервис НЕ МЕНЯЕТ url – он просто перенаправляет его на нужный сервис, в нашем случае – это graphql-monolit-service. И получаем, что ничего не работает – т.к. к поду обращение происходит по адресу – <http://graphql-monolit-service:3000/gql/monolit>, а такой адрес нигде в приложении не отслеживается.

Решения:

* Править настройки ingress-сервиса – можно дописать модуль rewrite в конфиг сервиса, но это глобальная настройка и тогда будут проблемы с другими уже развернутыми в кубике приложениями, там у них все настроено под ихние пути url.
* Добавить при обработке роутинга константу, в которой будет прописана эта часть адреса url и тогда к ресурсам и страницам будет обращение по аресам с учетом этой части URL. Так и сделано – есть переменная URL\_PREFIX, в которой прописан эта часть адреса - /gql/monolit/. В результате все работает корректно.

Пробовал подменить часть адреса Url при обработке middleware – не сработало.

Все на этом, проверяем – должно работать по адресу – http://localhost:8080/gql/monolit