1 Необходимо установить docker desktop c [сайта](https://www.docker.com/products/docker-desktop/). Установка показана на рисунке

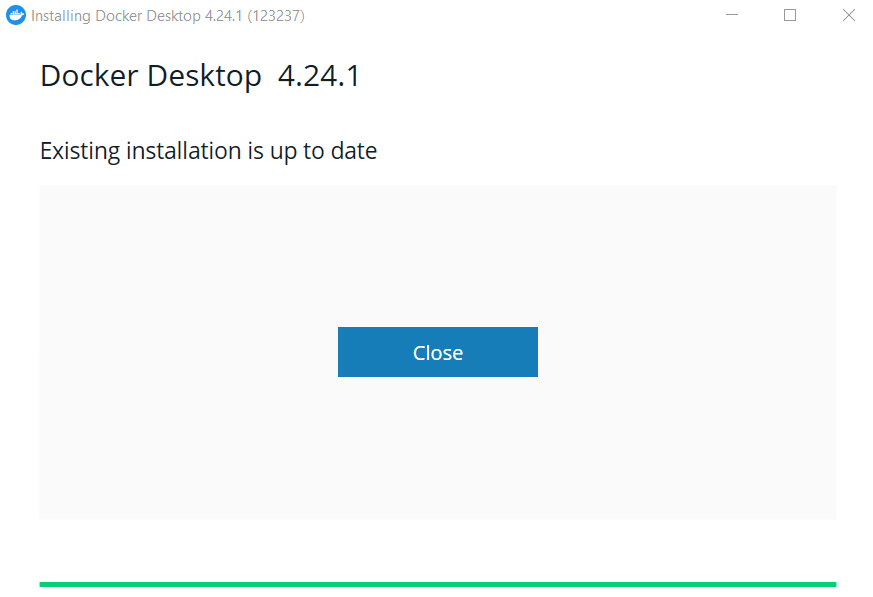


Рисунок 1

2 Для создания API необходимо создать проект Node.js с использованием фреймворка Hono. Для этого следует выполнить команду: npm create hono@latest my-app, чтобы создать проект с предустановленным фреймворком, как показано на рисунке 2.

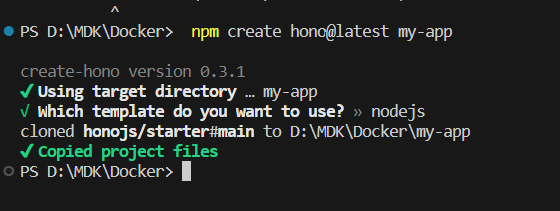


Рисунок 2

3 Для инициализации пакетов Node.js в терминале нужно написать команду:npm i, как показано на рисунке 3.

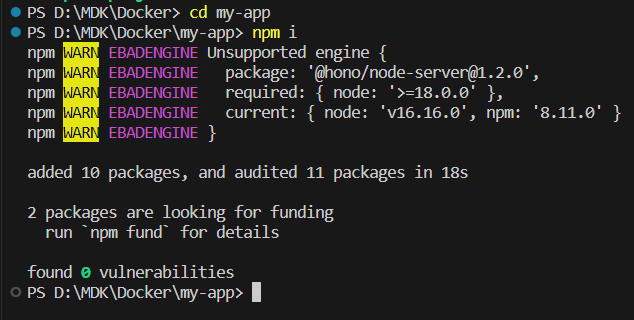


Рисунок 3

4 Для обеспечения взаимодействия с базой данных MySQL, необходимо загрузить дополнительный пакет mysql2, для этого нужно написать в консоле команду npm i mysql2, как показано на рисунке 4.

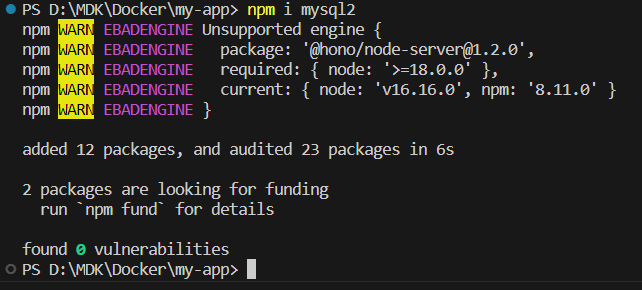


Рисунок 4

5 Необходимо разработать api. Пример API с использованием фреймворка "hono" представлен на рисунке 5.



Рисунок 5

6 В корневой папке необходимо создать файл с названием "Dockerfile", который будет использоваться для создания Docker-образа приложения, как показано на рисунке 6.

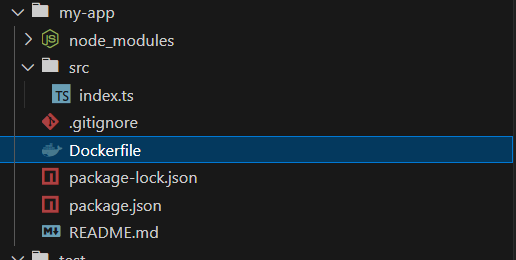


Рисунок 6

7 В dockerfile необходимо написать настройки для создания образа Docker, включая базовый образ, команды для установки зависимостей, копирование файлов проекта и настройки запуска приложения.

Пример dockerfile для Node.js показан на рисунке 7.

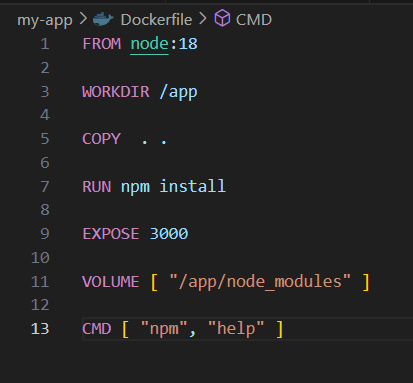


Рисунок 7

Пояснение:

* FROM — задает базовый (родительский) образ.
* WORKDIR — задаёт рабочую директорию для следующей инструкции.
* COPY — копирует в контейнер файлы и папки.
* RUN — выполняет команду и создает слой образа. Используется для установки в контейнер пакетов.
* EXPOSE — указывает на необходимость открыть порт.
* VOLUME — создает точку монтирования для работы с постоянным хранилищем.
* CMD — описывает команду с аргументами, которую нужно выполнить когда контейнер будет запущен. Аргументы могут быть переопределены при запуске контейнера. В файле может присутствовать лишь одна инструкция CMD.

8 В корневой папке вашего проекта создайте файл с именем docker-compose.yaml. Этот файл будет использоваться для определения и управления контейнерами вашего приложения. Показано на рисунке 8.

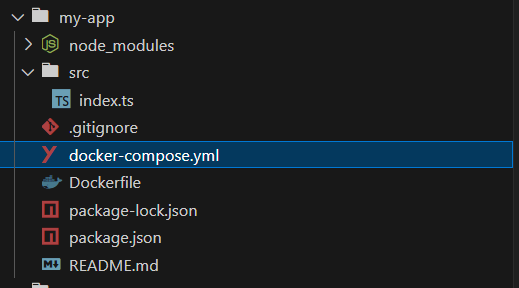


Рисунок 8

9 Нужно отредактировать docker-compose.yaml, описывая контейнеры, их зависимости и настройки. Ниже приведен пример docker-compose.yaml для двух контейнеров: контейнера Node.js приложения и контейнера базы данных MySQL. Пример показан на рисунке 9.

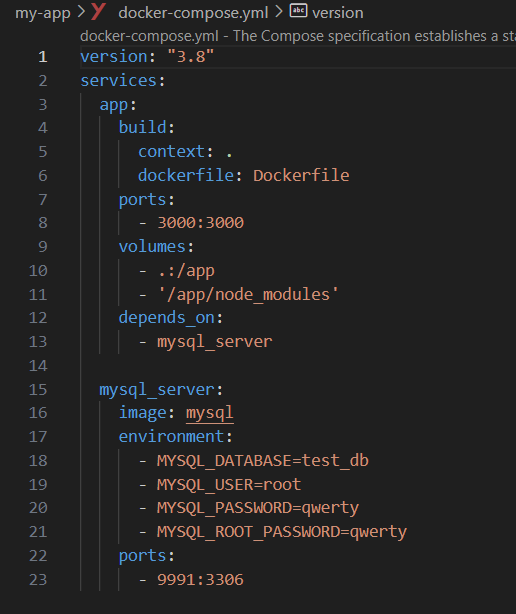


Рисунок 9

В данном примере определены два сервиса: app и mysql\_serve. Сервис app создается на основе ранее описанного DockerFile. Сервис mysql\_serve использует официальный образ MySQL.  
 Поля docker-compose.yaml:

* version: указывает версию схемы docker-compose
* services: здесь начинается описание служб (services), которые будут запущены с использованием Docker Compose. В данном случае, определены две службы: app и mysql\_server.
* app: имя службы
* build:тот раздел определяет настройки для построения
* context: это означает, что Docker Compose будет использовать текущий каталог в качестве контекста для построения образа.
* dockerfile: dockerfile: Это указывает на имя Dockerfile.
* ports: этот раздел определяет порты
* volums: этом разделе определены тома (volumes) для контейнера app. Это позволяет разделять файлы между хостовой машиной и контейнером. Здесь определены два тома:
  + .:/app: Этот том примонтирует текущий каталог хостовой машины к каталогу /app внутри контейнера.
  + '/app/node\_modules': Этот том примонтирует каталог /app/node\_modules внутри контейнера, но без привязки к хостовой машине.
* depends\_on: это означает, что app не будет запущен, пока mysql\_server не будет полностью запущен.
* mysql\_server: это вторая служба
* image: здесь указывается имя образа, который будет использоваться для контейнера mysql\_server
* environment: здесь устанавливаются параметры для настройки MySQL, такие как имя базы данных, имя пользователя и пароли.

10 Для создания и запуска контейнеров,необходимо написать в консоле команду docker-compose up, как показано на рисунке 10.

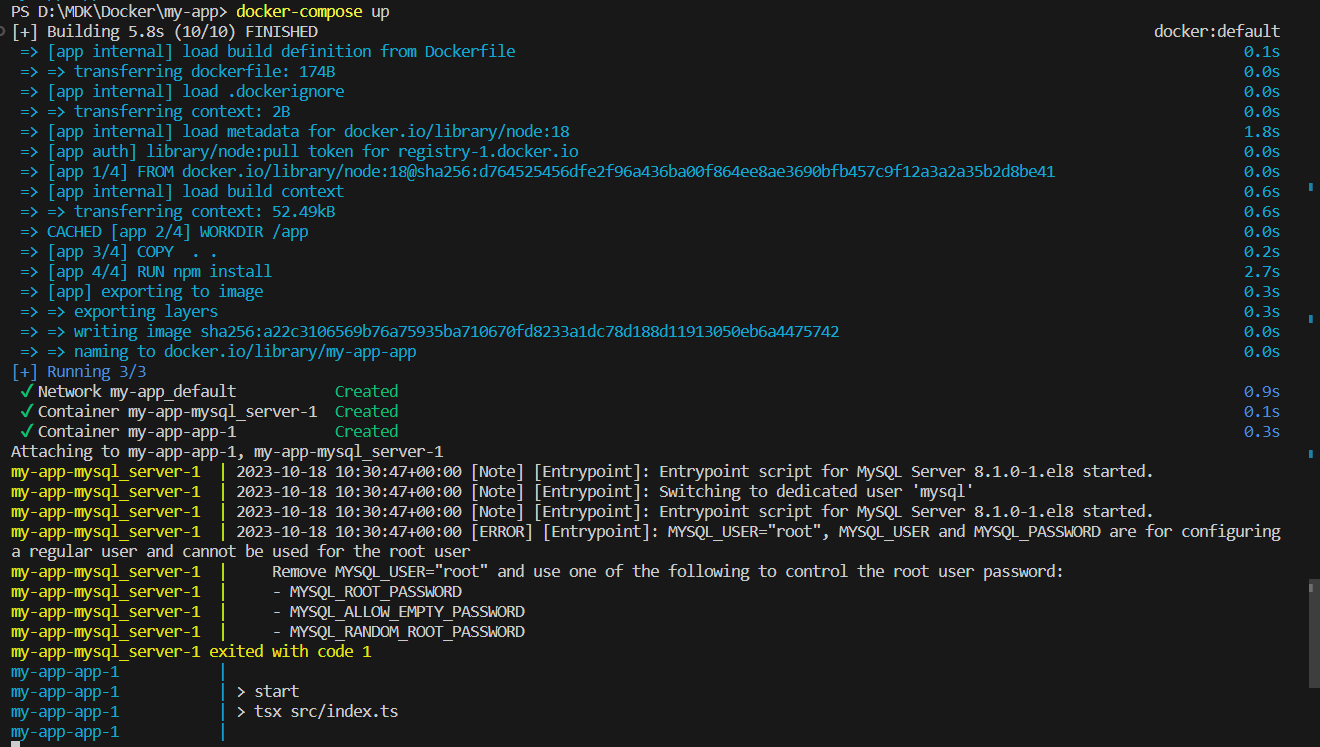


Рисунок 10

11 Необходимо установить соединение с базой данных MySQL, используя учетные данные, указанные в файле docker-compose.yaml. Показано на рисунке 11.

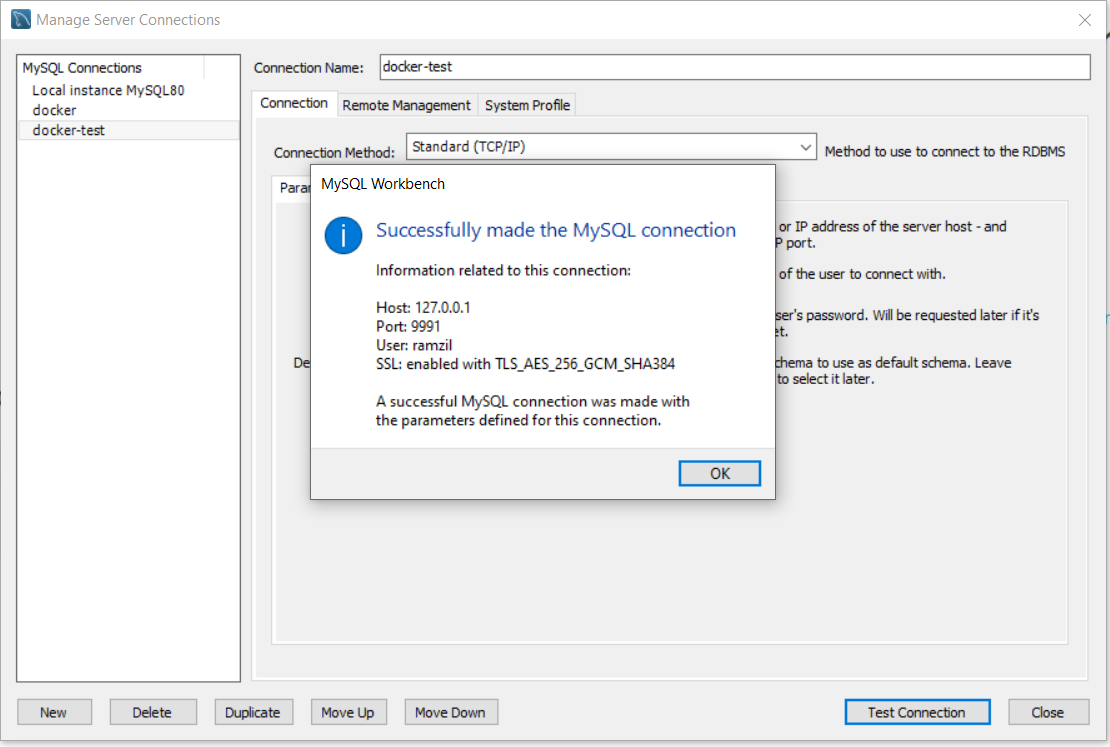


Рисунок 11

12 В базе данных нужно создать таблицу users,как показано на рисунке12.

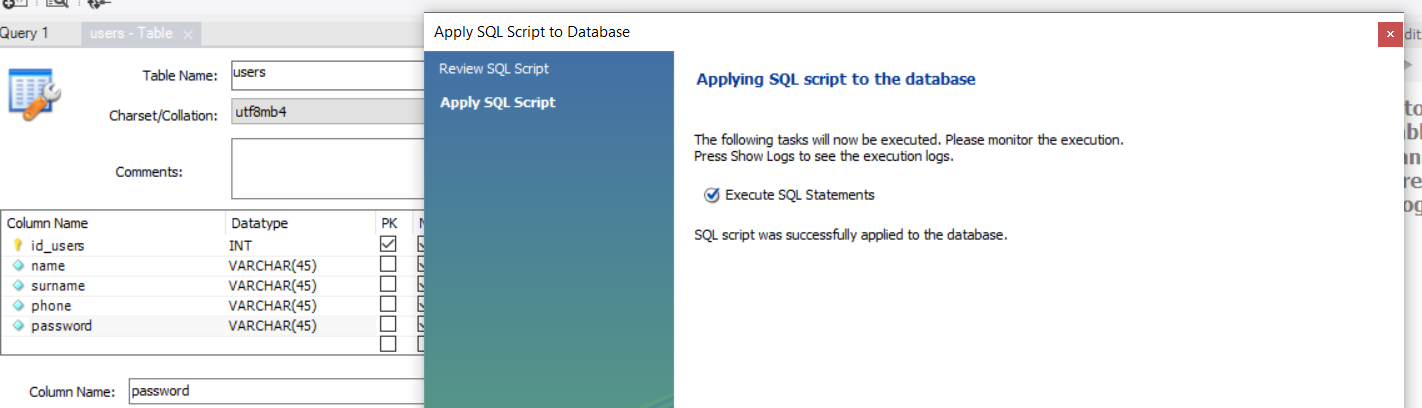


Рисунок 12

13 Нужно заполнить таблицу данными, как показано на рисунке 13.

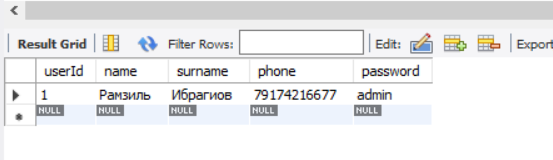
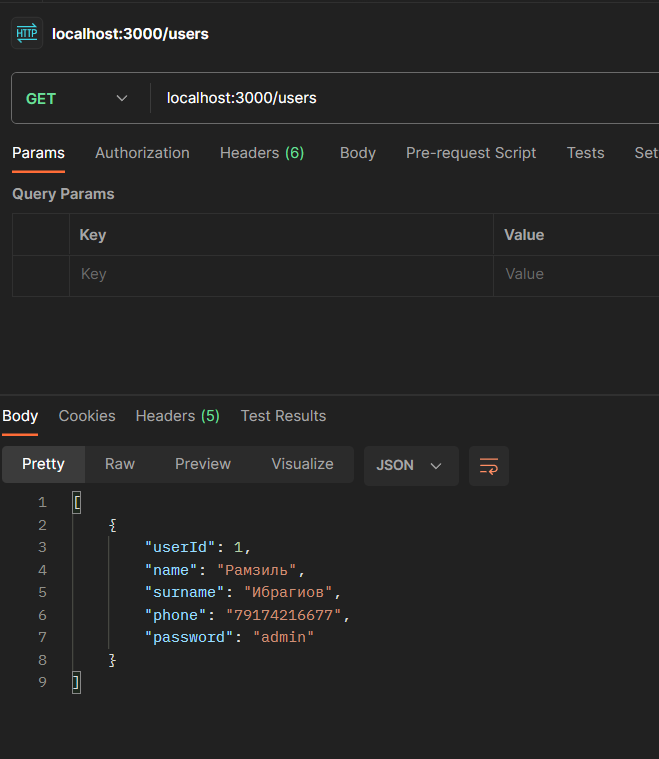


Рисунок 13

14 После выполнения всех указанных инструкций, требуется повторно запустить контейнеры и проверить работу API с использованием программы Postman или в браузере. Смотрите пример на рисунке 14.

  
Рисунок 14