

Департамент образования и науки города Москвы  
Государственное автономное образовательное учреждение высшего  
образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

Макарова Виктория Сергеевна

### **ЛАБОРАТОРНАЯ РАБОТА 3.1**

Docker Compose для мультиконтейнерных приложений  
Направление подготовки

38.03.05 Бизнес-информатика

Профиль подготовки

Аналитика данных и эффективное управление

Курс обучения: 4

Форма обучения: очная

Преподаватель: кандидат технических наук,

доцент Босенко Тимур Муртазович

Москва

2025

**Цель работы:** освоить использование Docker Compose для управления многоконтейнерными приложениями.

**Задачи:**

Разработать систему аналитики посещаемости сайта, которая:

1.Генерирует синтетические данные о посещении сайта в реальном времени.

2.Сохраняет данные в InfluxDB.

3.Визуализирует статистику с помощью Grafana.

4.Разворачивается через Docker Compose.

**Технический стек:**

Backend: Node.js

База данных: InfluxDB •

Визуализация: Grafana •

Контейнеризация: Docker + Docker Compose

Дерево проекта (Рис. 1):

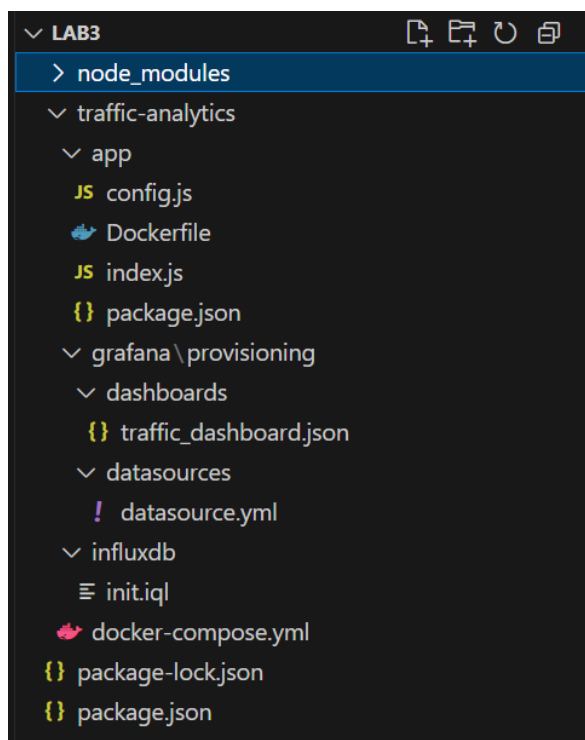


Рис. 1

**Функциональные требования:**

1.Генерация данных: •

Создание записей о посещаемости сайта каждые 2 секунд •

Генерация случайного страницы сайта, id пользователя и времени посещения.

2.Хранение данных: •

Сохранение всех записей в InfluxDB •

3.Визуализация: •

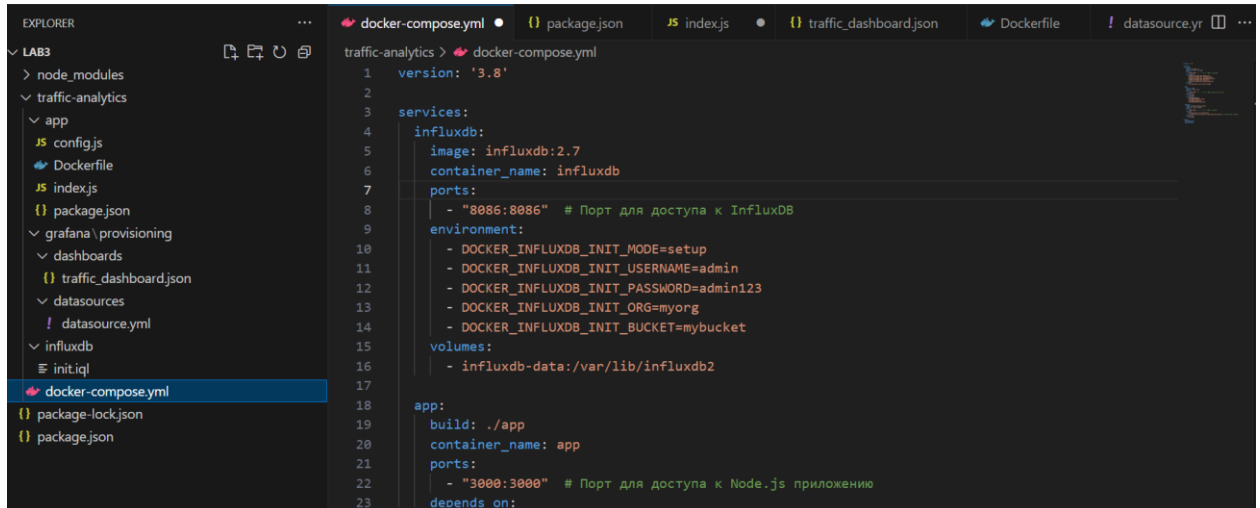
Графики посещаемости по времени

## Ход работы (Вариант st\_94)

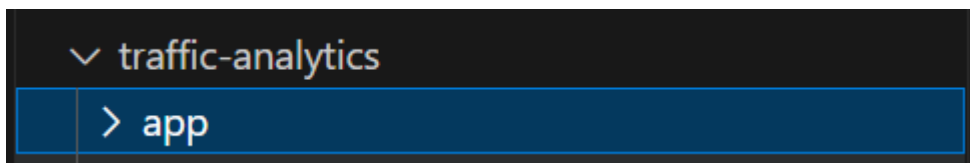
1. Создала новый каталог для проекта traffic-analytics:



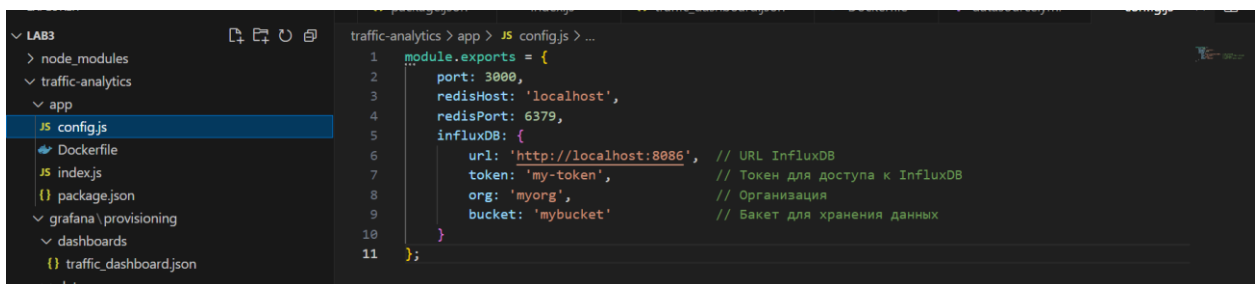
2. В каталоге traffic-analytics был создан файл docker-compose.yml.



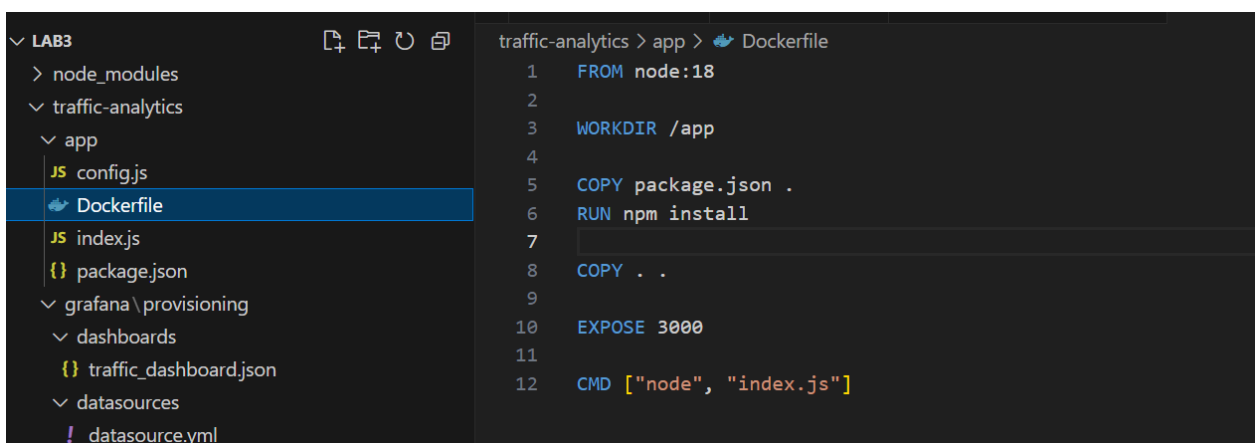
3. В этом же каталоге был создан каталог app.



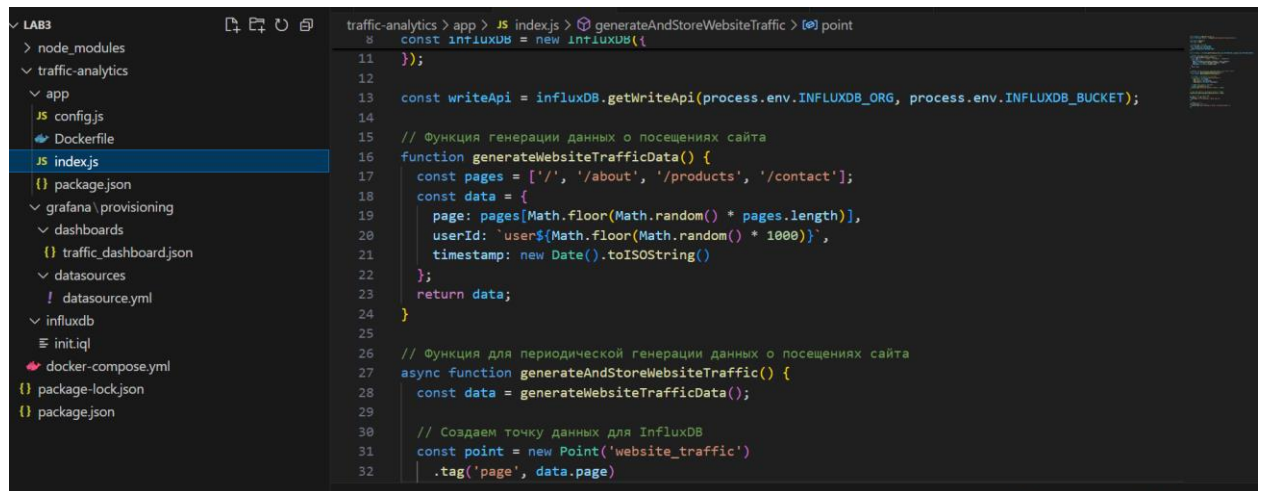
4. В каталоге app был создан файл config.js



5. В каталоге app был создан файл Dockerfile

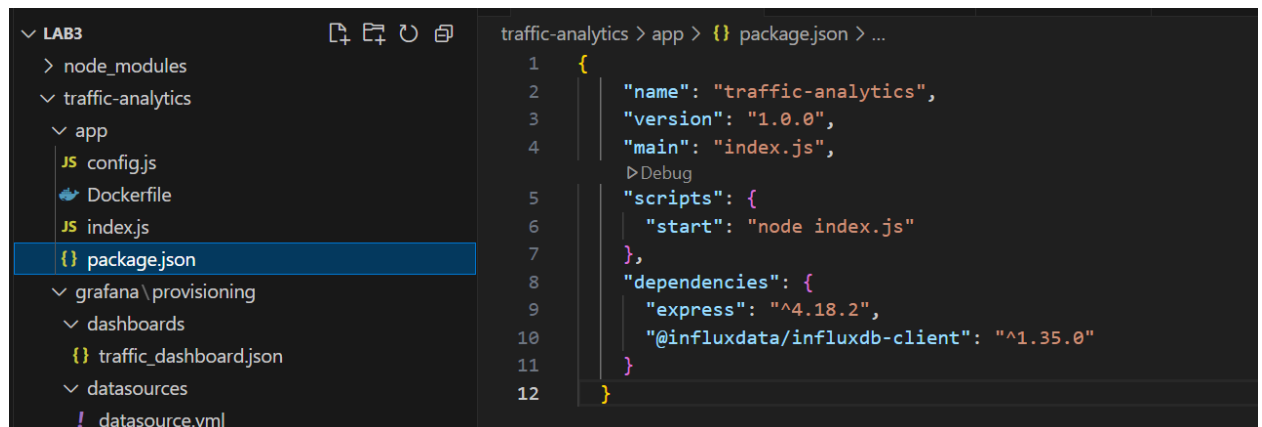


6. В каталоге app был создан файл index.js с генерацией синтетических данных о посещаемости сайта.



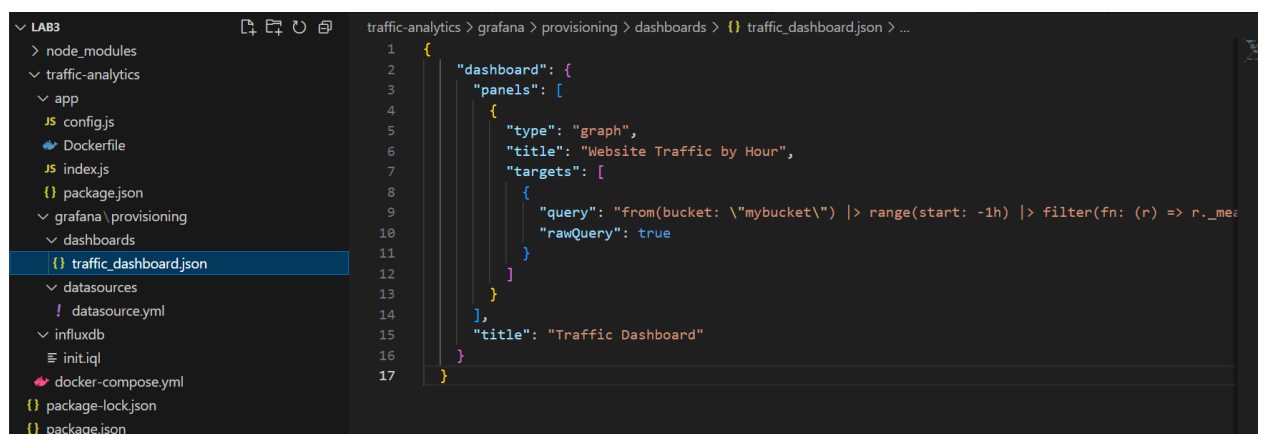
```
traffic-analytics > app > JS index.js > generateAndStoreWebsiteTraffic > [0] point
8  const influxDB = new InfluxDB({
11  });
12
13  const writeApi = influxDB.getWriteApi(process.env.INFLUXDB_ORG, process.env.INFLUXDB_BUCKET);
14
15  // Функция генерации данных о посещениях сайта
16  function generateWebsiteTrafficData() {
17    const pages = ['/', '/about', '/products', '/contact'];
18    const data = {
19      page: pages[Math.floor(Math.random() * pages.length)],
20      userId: `user${Math.floor(Math.random() * 1000)}`,
21      timestamp: new Date().toISOString()
22    };
23    return data;
24  }
25
26  // Функция для периодической генерации данных о посещениях сайта
27  async function generateAndStoreWebsiteTraffic() {
28    const data = generateWebsiteTrafficData();
29
30    // Создаем точку данных для InfluxDB
31    const point = new Point('website_traffic')
32      .tag('page', data.page)
```

7. В каталоге app был создан файл package.json. С записью зависимостей.



```
traffic-analytics > app > {} package.json > ...
1  {
2    "name": "traffic-analytics",
3    "version": "1.0.0",
4    "main": "index.js",
5    "scripts": {
6      "start": "node index.js"
7    },
8    "dependencies": {
9      "express": "^4.18.2",
10     "@influxdata/influxdb-client": "^1.35.0"
11   }
12 }
```

8. В каталоге grafana/provisioning создан каталог dashboards с файлом traffic\_dashboard.json.



```
traffic-analytics > grafana > provisioning > dashboards > {} traffic_dashboard.json > ...
1  {
2    "dashboard": {
3      "panels": [
4        {
5          "type": "graph",
6          "title": "Website Traffic by Hour",
7          "targets": [
8            {
9              "query": "from(bucket: \"mybucket\") |> range(start: -1h) |> filter(fn: (r) => r._me",
10             "rawQuery": true
11           }
12         ]
13       },
14     ],
15     "title": "Traffic Dashboard"
16   }
17 }
```

## Запуск контейнеров:

1. Пересоберем и запустим контейнеры

```

PS C:\Users\boxke\OneDrive\Рабочий стол\lab3\traffic-analytics> docker-compose up --build
time="2025-03-07T16:25:33+03:00" level=warning msg="C:\\Users\\boxke\\OneDrive\\Рабочий стол\\lab3\\traffic-analytics\\d
ocker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusio
n"
[+] Running 1/1
  ✓ influxdb Pulled                                2.4s
[+] Building 1.9s (12/12) FINISHED                  docker:desktop-linux
=> [app internal] load build definition from Dockerfile      0.0s
=> => transferring dockerfile: 160B                          0.0s
=> [app internal] load metadata for docker.io/library/node:18 2.1s
=> [app auth] library/node:pull token for registry-1.docker.io 0.0s
=> [app internal] load .dockerignore                        0.0s
=> => transferring context: 2B                                0.0s
=> [app 1/5] FROM docker.io/library/node:18@sha256:ba756f198b4b1e0114b53b23121c8ae27f7ae4d5d95ca4a0554b0649cc9c7 0.0s
=> [app internal] load build context                        0.0s
=> => transferring context: 120B                              0.0s
=> CACHED [app 2/5] WORKDIR /app                          0.0s
=> CACHED [app 3/5] COPY package.json .                    0.0s
=> CACHED [app 4/5] RUN npm install                        0.0s
=> CACHED [app 5/5] COPY . .                               0.0s
=> [app] exporting to image                                0.0s
=> => exporting layers                                       0.0s
=> => writing image sha256:5918da8cbd8ef95207de72eaba877d891e4ed5fa2ad3f510d9d3bdada846c5d7 0.0s

```

```

[+] Running 4/4
  ✓ app           Built
  ✓ Container influxdb Created
  ✓ Container app Created
  ✓ Container grafana Created

```

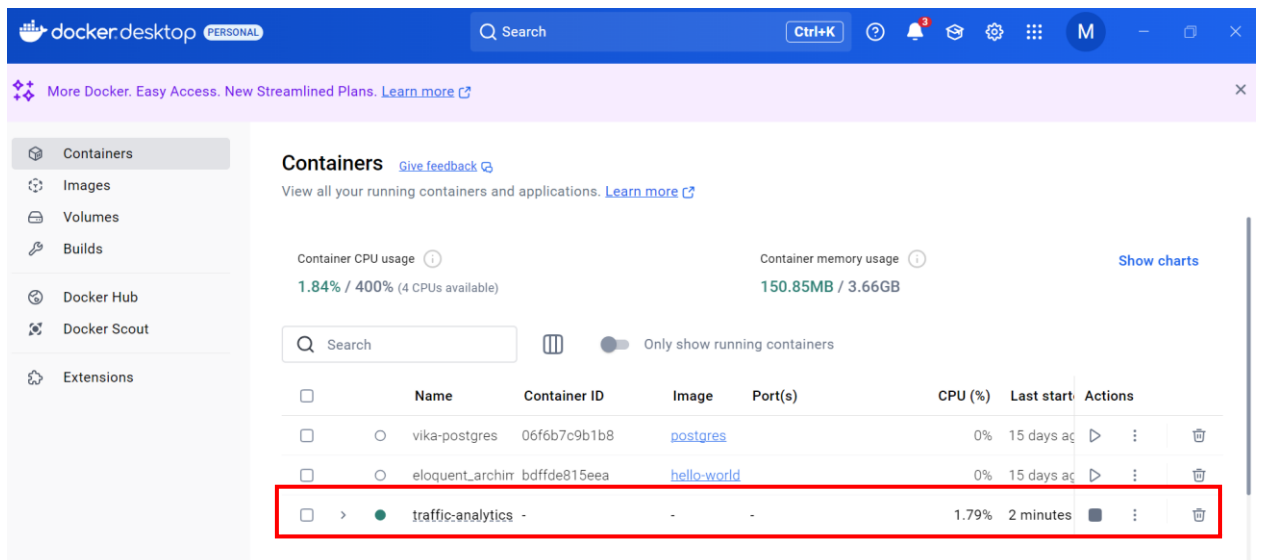
## 2. Генерация данных в реальном времени.

```

app      | Generated website traffic: {
app      |   page: '/about',
app      |   userId: 'user272',
app      |   timestamp: '2025-03-07T13:26:19.710Z'
app      | }
app      | Generated website traffic: { page: '/', userId: 'user80', timestamp: '2025-03-07T13:26:21.710Z' }
app      | Generated website traffic: {
app      |   page: '/contact',
app      |   userId: 'user629',
app      |   timestamp: '2025-03-07T13:26:23.712Z'
app      | }
app      | Generated website traffic: {
app      |   page: '/about',
app      |   userId: 'user468',
app      |   timestamp: '2025-03-07T13:26:25.714Z'
app      | }
app      | Generated website traffic: {
app      |   page: '/about',
app      |   userId: 'user381',
app      |   timestamp: '2025-03-07T13:26:27.716Z'
app      | }
app      | Generated website traffic: {
app      |   page: '/contact',
app      |   userId: 'user946',
app      |   timestamp: '2025-03-07T13:26:29.717Z'

```

## 3. Контейнер запущен успешно



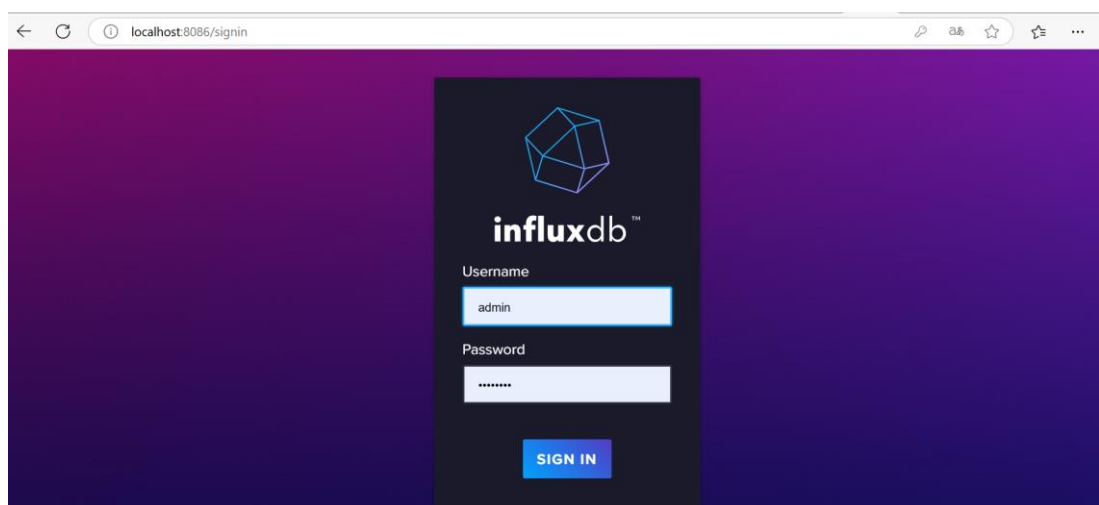
```
PS C:\Users\boxke\OneDrive\Рабочий стол\lab3\traffic-analytics> docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED         STATUS         PORTS
31e8a072da13   grafana/grafana:latest             "/run.sh"                About an hour ago Up 4 minutes   0.0
.0.0:3001->3000/tcp   grafana
7c67ce5bfd38   traffic-analytics-app              "docker-entrypoint.s..." About an hour ago Up 4 minutes   0.0
.0.0:3000->3000/tcp   app
1d199e86e2f4   influxdb:2.7                       "/entrypoint.sh infl..." About an hour ago Up 4 minutes   0.0
.0.0:8086->8086/tcp   influxdb
```

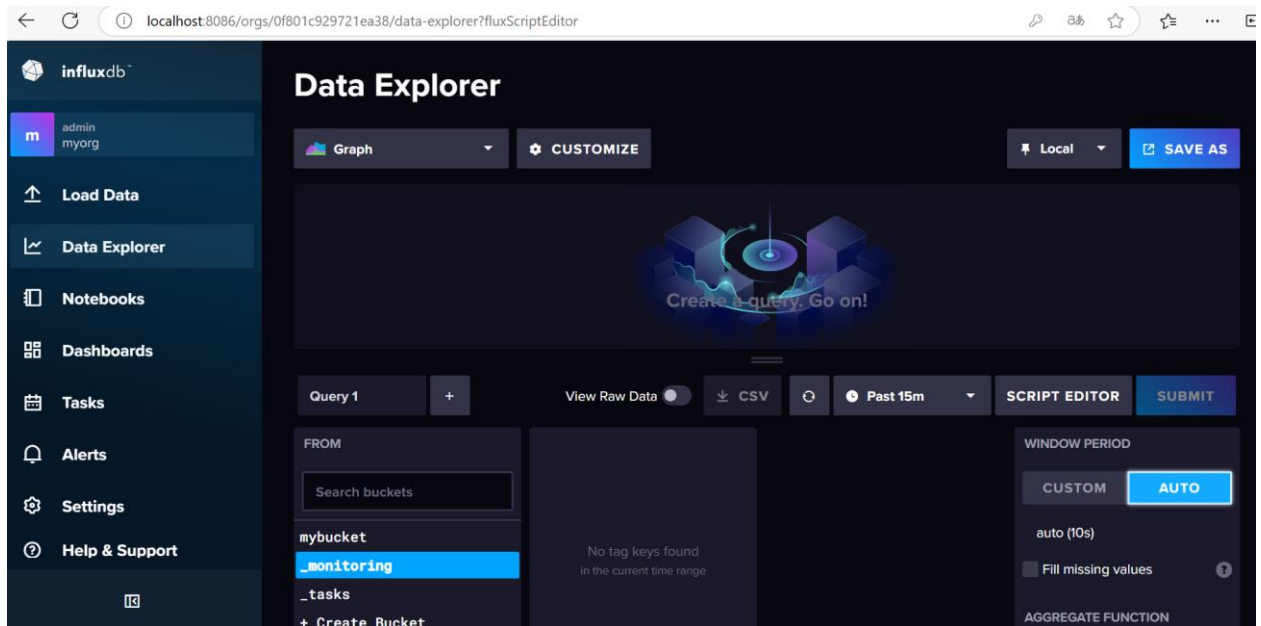
## Проверка доступности сервисов:

1. Node.js: <http://localhost:3000>

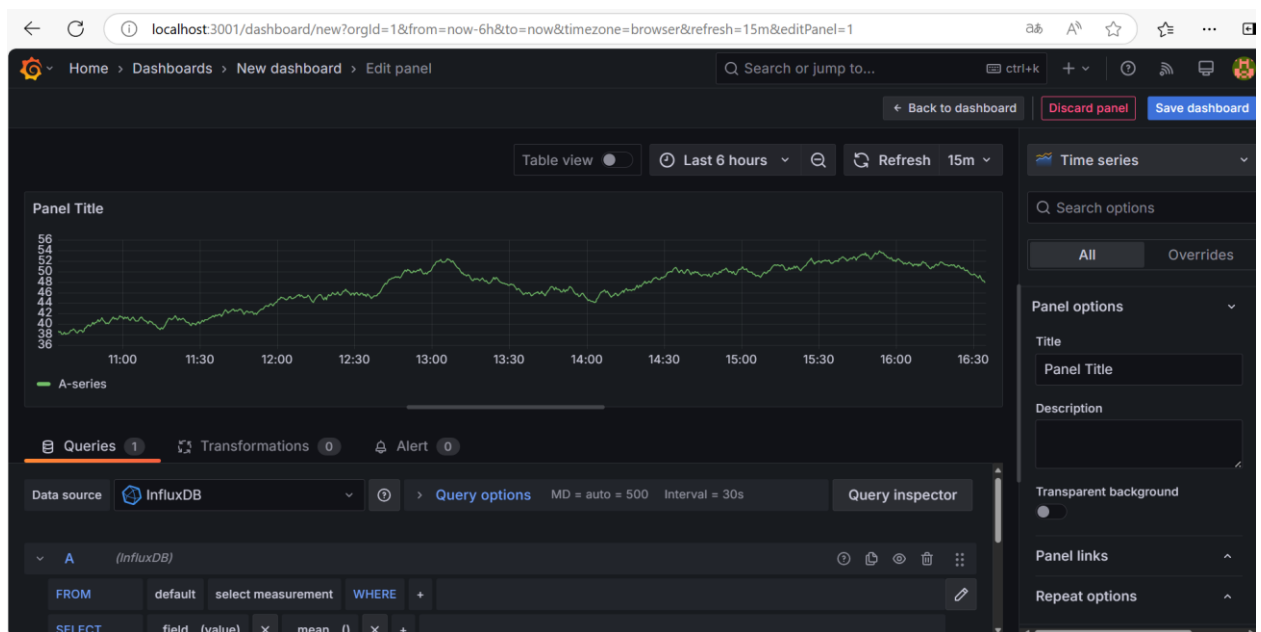


2. InfluxDB: <http://localhost:8086>





3. Grafana: <http://localhost:3001>. График посещаемости сайта.



## Контрольные вопросы:

1. Что такое Docker Compose и для чего он используется?

**Docker Compose** — это инструмент для управления многоконтейнерными приложениями. Он позволяет определять и запускать несколько контейнеров как единое приложение с помощью одного файла конфигурации (docker-compose.yml).

**Для чего используется:**



- a. Упрощение управления сложными приложениями, состоящими из нескольких сервисов (например, веб-сервер, база данных, кэш).
  - b. Автоматизация запуска, остановки и настройки контейнеров.
  - c. Обеспечение согласованности среды разработки, тестирования и production.
2. Какие основные преимущества использования Docker Compose для управления многоконтейнерными приложениями?

- **Упрощение конфигурации:** Все настройки контейнеров описываются в одном файле (docker-compose.yml).
- **Автоматизация:** Запуск и остановка всех сервисов выполняются одной командой.
- **Изоляция среды:** Каждый сервис работает в своем контейнере, что предотвращает конфликты зависимостей.
- **Масштабируемость:** Легко добавлять или удалять сервисы.
- **Согласованность:** Одинаковая конфигурация для всех сред

3. Какие основные разделы и директивы используются в файле docker-compose.yml?

Основные разделы и директивы:

- **version:** Версия формата файла.
- **services:** Определение сервисов (контейнеров). Каждый сервис может включать:
  - **image:** Используемый образ Docker.
  - **build:** Путь к Dockerfile для сборки образа.
  - **ports:** Проброс портов (например, "8080:80").
  - **environment:** Переменные окружения.
  - **volumes:** Подключение томов или директорий.
  - **depends\_on:** Зависимости между сервисами.
  - **networks:** Подключение к пользовательским сетям.

4. Как запустить многоконтейнерное приложение с помощью Docker Compose?

`docker-compose up -d`

5. Как остановить и удалить контейнеры, запущенные с помощью Docker Compose?

**Остановка контейнеров:**

Docker-compose down

Удаление контейнеров:

Docker-compose down --volumes