**CI/CD Visualizer Project**

**Описание проекта**

Данный проект представляет собой систему визуализации CI/CD пайплайнов с фронтендом на **React** (Next.js), бэкендом на **Go** и базой данных **PostgreSQL**. Система предоставляет возможность управлять пайплайнами и задачами, а также отображает их зависимость и статусы в графическом интерфейсе.

**Требования**

Перед запуском проекта убедитесь, что на вашем компьютере установлены следующие инструменты:

1. [Docker](https://www.docker.com/) и Docker Compose.
2. **Node.js** (рекомендуемая версия 18.x) — требуется для работы Next.js.
3. **Go** (рекомендуемая версия 1.20.x) — требуется для сборки и запуска бэкенда.
4. (Опционально) zip или tar для разархивирования проекта, если он передается в архиве.

**Установка и запуск**

**Шаги для запуска проекта:**

1. **Скачайте и распакуйте проект:**

unzip project-root.zip

cd project-root

Или, если архив в формате.tar.gz:

tar -xvzf project-root.tar.gz

cd project-root

Или, клонируйте репозиторий, перед этим настроев ssh:  
git clone ssh://git@gitverse.ru:2222/Provoin77/IT-Planet\_Pro\_GitVerse.git

cd project-root

1. **Запустите Docker Compose:**

docker-compose up --build

Эта команда создаст и запустит три контейнера:

* + **PostgreSQL** (порт 5433): База данных.
  + **Backend** (порт 8080): Go-приложение.
  + **Frontend** (порт 3000): Next.js-приложение.

1. **Доступ к приложению:**
   * Фронтенд: <http://localhost:3000>
   * Бэкенд: <http://localhost:8080>

**Альтернативный запуск (без Docker)**

1. **Запуск базы данных:**
   * Убедитесь, что PostgreSQL установлен и запущен.
   * Создайте базу данных и инициализируйте её с помощью init\_db.sql:

psql -U your\_username -d your\_database -f init\_db.sql

1. **Запуск бэкенда:**
   * Перейдите в директорию backend:

cd backend

* + Установите зависимости:

go mod tidy

* + Запустите сервер:

go run main.go

1. **Запуск фронтенда:**
   * Перейдите в директорию frontend:

cd frontend

* + Установите зависимости:

npm install

* + Запустите Next.js в режиме разработки:

npm run dev

* + Приложение будет доступно по адресу <http://localhost:3000>.

**Структура проекта**

B:\Games\programi\Dockeer\projectt-root\

├── docker-compose.yml # Файл конфигурации Docker Compose

├── init\_db.sql # Скрипт инициализации базы данных PostgreSQL

├── node\_modules/ # Зависимости Node.js

├── package.json # Список зависимостей Node.js

├── package-lock.json # Закрепленные версии зависимостей

├── test.yaml # Тестовые данные

├── backend/ # Бэкенд на Go

│ ├── .env # Переменные окружения для бэкенда

│ ├── Dockerfile # Dockerfile для бэкенда

│ ├── go.mod # Список зависимостей Go

│ ├── go.sum # Контрольные суммы зависимостей

│ ├── main.go # Основной файл приложения

│ └── mockGenerator.js # Скрипт для генерации mock-данных

├── frontend/ # Фронтенд на Next.js

│ ├── components/ # Реактовые компоненты

│ │ ── PipelineGraph.js # Компонент для графа пайплайнов

│ │

│ ├── pages/ # Страницы приложения

│ │ ├── index.js # Главная страница

│ │ ├── \_app.js # WebSocket для основной страницы

│ │ ├── TaskDetails.js # Страница деталей задач

│ │ └── AnalyticsPage.js # Страница аналитики

│ │ └── ExtendedAnalyticsPage.js # Страница расширенной статистики

│ ├── styles/ # Стили CSS

│ │ └── styles.css # Основные стили

│ ├── .env.local # Переменные окружения для фронтенда

│ ├── Dockerfile # Dockerfile для фронтенда

│ ├── package.json # Список зависимостей Node.js

│ └── package-lock.json # Закрепленные версии зависимостей

└── frontend/node\_modules/ # Зависимости Node.js

└── ... # (Содержимое папки node\_modules)

**Описание контейнеров**

1. **PostgreSQL:**
   * Версия: 13
   * Порт: 5433
   * Файл инициализации базы данных: init\_db.sql
   * Переменные окружения:
     + POSTGRES\_USER: gitverse\_user
     + POSTGRES\_PASSWORD: gitverse\_password
     + POSTGRES\_DB: gitverse\_db
2. **Backend (Go):**
   * Версия Go: 1.20
   * Порт: 8080
   * Переменные окружения (установлены в backend/.env):
     + DB\_HOST: postgres
     + DB\_PORT: 5432
     + DB\_USER: gitverse\_user
     + DB\_PASSWORD: gitverse\_password
     + DB\_NAME: gitverse\_db
3. **Frontend (Next.js):**
   * Версия Node.js: 18
   * Порт: 3000
   * Переменные окружения (установлены в frontend/.env.local):
     + NEXT\_PUBLIC\_API\_URL: http://localhost:8080

**Начальные данные в базе**

После запуска базы данных, скрипт init\_db.sql создаст:

1. Таблицы:
   * user\_role, user, pipeline, task, task\_dependency, task\_log, pipeline\_stat, task\_metrics, access\_control.
2. Начальные данные:
   * Роли пользователей (администратор, разработчик и др.).
   * Пример пользователей.
   * Пример пайплайна и задач:
     + Пайплайн: **Развертывание веб-приложения**.
     + Задачи: **Разработка**, **Тестирование**, **Развертывание**.

**Полезные команды**

**Остановка всех контейнеров:**

docker-compose down

**Перезапуск с пересборкой:**

docker-compose up --build

**Очистка данных PostgreSQL:**

Удалите том postgres\_data:

docker volume rm project-root\_postgres\_data

**Заметки**

1. **Если docker-compose up выдает ошибку:** Проверьте, что порты 3000, 8080, и 5433 не заняты.
2. **Переменные окружения:** Все переменные окружения установлены в .env файлах для каждого сервиса.