**Тестовое задание:**

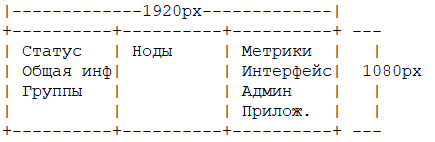
**Разработать представление отображающее состояние инфраструктуры предприятия**.

1. **SQL-запросы:**
   1. **Напишите запрос получающий все группы**

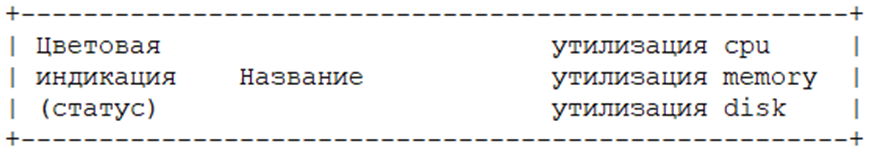
* Отредактируйте файл ./node/sql/groups.sql
* Группы должны содержать все входящие в них ноды.
* Ноды должны содержать информацию о интерфейсах.
* Ноды должны содержать информацию об приложениях.
* Ноды должны содержать информацию об администраторах.
* Ноды и интерфейсы должны содержать информацию о статусе (цвет и описание).
  1. **Напишите запрос, получающий метрики**
* Отредактируйте файл ./node/sql/metrics.sql
* Метрики должны содержать ноды.

1. **Фронтэнд и макет представления:** 
   1. **Общие требования:**

* Реализовать представление на VueJS или ReactJS.
* Реализовать визуальную часть на основе созданного шаблона фигма.
* Представление должно использовать одну из библиотек управления состоянием на ваше усмотрение (Vuex, Pinia, Redux, Mobx и т.д.).
* Представление должно корректно работать в браузерах на основе Google Chrome.
* Для написания стилей можно использовать препроцессоры less/scss/styles.
* Представление должно запрашивать новые данные с backend 1 раз в минуту.
* Для отображения графиков необъодимо использвать библиотеку ChartJS ([chartjs.org](https://www.chartjs.org/)).
  1. **Макет представления в Figma (опционально):**
* Макет должен быть реализован под разрешение 1920х 1080 пикселей.
* Общее описание элементов макета указано в пункте 2.3 и 2.4.
* Стили оформления придумать самостоятельно.
  1. **Требования к оформлению элементов:**
* Схематичное расположение элементов, выбор дизайна представления остаётся на усмотрение разработчика.



* Представление должно быть разбито на 3 равных столбца
* Каждый столбец должен содержать блоки с данными.
* Блоки могут содержать:
* элементы “ноды”
* таблицы
* графики
* строковые элементы
  1. **Описание блоков:**
     + Блок "Статус":
     + Необходимо реализовать статус сервиса - рассчитывается по наихудшему из статусов нод.
     + Должен отображать название и цветовой окрас текущего статуса отображаемых нод (с учетом установленного фильтра групп).
     + Блок "Общая информация":
     + Необходимо реализовать статус сервиса - рассчитывается по наихудшему из статусов нод.
     + Блок "Группы":
     + При клике на группу в столбце 2 и 3 должны отображаться данные только из этой группы.
     + Блок "Ноды":
     + Должен выводить список элементов "нода"

****

* + - Элемент "Нодa":
    - Нода должна отображать название, последние данные метрик (метрика со значением> 85> 95 окрашивается желтым, метрика> 95 окрашивается красных), цветовую индикацию статуса ноды.
    - При клике на ноде в столбце 3 должны отображаться только метрики и данные этой ноды, а в столбце 1 группы в которые входит эта нода.
    - Блок "Метрики":
    - Метрики должны быть реализованы в виде графика chartjs (https://www.chartjs.org/)
    - Блок "Интерфейс":
    - Должен отображать название и статус интерфейса ноды.
    - Блок "Администратор":
    - Должен отображаться данные администратора ноды.
    - Блок "Приложения":
    - Должен отображать данные приложений, выполняющихся на ноде.

1. **Git-репозиторий:**
   1. **Требования:**

* Необходимо создать единый отдельный git-репозиторий для всего проекта.
* Файл макета из фигмы необходимо вложить в проект.
* Структура файлов и папок в репозитории формируется по вашему усмотрению.

1. **Установка и запуск:** 
   1. **Backend:**
      * Установить NodeJS (https://nodejs.org/ru/)
      * Перейти в папку "backend" и выполнить установку зависимостей: "npm i"
      * Запустите backend командой: "npm run start"
      * Проверьте работу в браузере по адресу: <http://127.0.0.1:23456/api>
2. **Дополнительное программное обеспечение:**

Для просмотра таблиц и выполнения тестовых запросов можно воспользоваться бесплатной программой "DB Browser for SQLite” (https://sqlitebrowser.org/)