Цели практической работы

* Ознакомиться с интерфейсом Grafana.
* Поработать с инструментом оркестрации контейнеров docker-compose.

Работа состоит из двух частей.

Часть 1. Сбор метрик

Необходимо собрать более или менее настоящие метрики работы инфраструктуры и сервиса в ней, которые вы впоследствии визуализируете в Grafana. В качестве инфраструктуры выступит ваш рабочий компьютер, в качестве сервиса — сервис предсказаний категории цены товара, который вы делали в практической работе к модулю «Модель как API».

Что нужно сделать

1. Скачайте [Python-скрипт.](https://drive.google.com/file/d/1ArvHTXtMOWtNaUBmUjdv-JcEgBkPOn8M/view?usp=sharing)
2. Добавьте его в проект PyCharm, в котором вы выполняли практическую работу к модулю «Модель как API».
3. Установите все недостающие модули (модуль [psutil](https://pypi.org/project/psutil/" \t "_blank)). Скрипт, который вы скачали, будет постоянно выполнять запросы в ваш сервис и замерять, как долго обрабатывается каждый запрос и сколько запросов удалось выполнить в единицу времени. Кроме того, скрипт периодически запускает бесполезные, но тяжёлые вычисления, создавая дополнительную случайную нагрузку на ваш CPU, чтобы метрики получились более разнообразные.
4. Запустите ваш сервис. С помощью Postman проверьте, что он отвечает на запросы.
5. Запустите скачанный на первом шаге скрипт и расслабьтесь: придётся подождать 12 минут. За это время он выполнит много запросов и произведёт необходимые измерения.  
   *Обратите внимание!* В скрипте есть небольшая хитрость со временем: он делает замеры раз в секунду в течение 720 секунд, то есть 12 минут. Однако метки времени искусственные, с грануляцией раз в минуту. Таким образом, за 12 минут вы получаете наблюдения, аналогичные тем, что были бы за 12 часов. Это небольшой обман, «ускорение» времени, которое подходит для нашего игрушечного примера. Если захотите, можете самостоятельно разобраться, как работает предложенный скрипт, но это необязательно. Цель скрипта — исключительно служебная.
6. Через 12 минут, когда скрипт отработает, он создаст файл *grafana.sql* в той же папке, где размещён. Этот файл — SQL-скрипт, который содержит код инициализации базы данных для ваших метрик. Метрики, которые сгенерированы скриптом, содержат информацию за искусственный двенадцатичасовой интервал, который начался десять часов назад до запуска скрипта и будет продолжаться ещё два часа после (как бы в будущем), чтобы создать видимость процесса, продолжающегося в реальном времени. Поэтому не откладывайте вторую часть практической работы надолго (более чем на два часа). Лучше приступить к ней сразу же, как будут готовы ваши метрики.

Часть 2. Визуализация метрик

Ваша задача — построить такой дашборд:

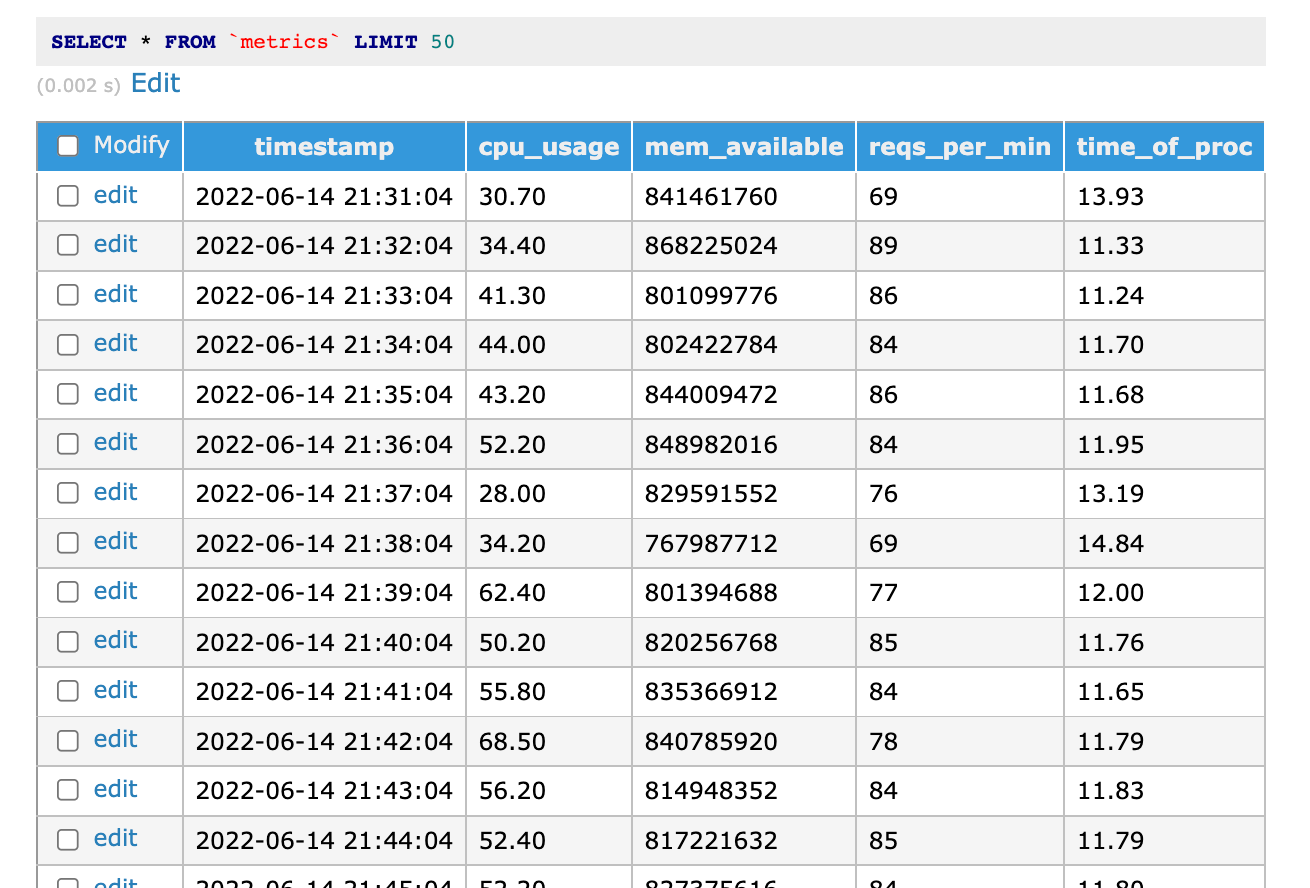


1. Верхний ряд — максимальное, минимальное и среднее время обработки одного запроса (метрика *time\_of\_proc*). Элемент визуализации — stat colored background — можно найти во вкладке suggested элемента выбора визуализации.
2. Второй ряд — график времени обработки запроса (метрика time\_of\_proc) и график объёма доступной оперативной памяти (метрика *mem\_available*).
3. Нижний ряд — график числа запросов, обработанных за минуту (метрика *reqs\_per\_min*), и график нагрузки на CPU, выраженной в процентах (метрика *cpu\_usage*).

Что нужно сделать

Значение всех необходимых метрик вы сгенерировали на первом шаге. Осталось положить их в базу данных и запустить Grafana.

Для этого:

1. Скачайте на компьютер [папку](https://drive.google.com/drive/folders/1n4_sxpkj804NClGfTknR7QczayVDx-sy?usp=sharing). В ней есть подпапка init, а в ней — файл *grafana.sql*. Сейчас этот файл пуст, поэтому замените его тем, который вы сгенерировали в первой части этой практической работы.
2. Запустите терминал (GitBash, если у вас Windows), перейдите в папку (командой cd), которую скачали. Выполните команду docker-compose up. На экране начнут появляться строки. Подождите, пока они не перестанут появляться (или начнут появляться с меньшей скоростью). ***Примечание***: файл docker-compose.yml, описывающий конфигурацию контейнеров, должен находиться в той же папке, в которой вы выполняете команду.
3. Откройте браузер, введите в адресную строку: <http://127.0.0.1:8080/>. Если вы всё сделали правильно, появится окно авторизации в adminer. Введите реквизиты: сервер базы — db, имя базы — *monitoring*, имя пользователя — *grafana*, пароль — *123456*. Когда авторизуетесь, убедитесь, что таблица metrics существует и заполнена данными:
4. Откройте новую вкладку браузера, введите адрес: <http://127.0.0.1:3000/>. Откроется окно авторизации Grafana. Введите логин и пароль: admin, admin (по умолчанию). Grafana тут же предложит установить новый пароль — выбирайте по умолчанию.
5. В интерфейсе Grafana добавьте новый источник данных MySQL (используйте реквизиты из шага 3), создайте новый дашборд и заполните его графиками, как на картинке под заголовком второй части работы. Значения метрик и графики почти наверняка будут отличаться от тех, что представлены на скриншоте.
6. Как только закончите, сохраните дашборд, а после сделайте снимок экрана так, чтобы все графики поместились, а их названия были читаемы.

Что оценивается

1. Программа управления контейнерами Docker установлена.
2. Корректно выбраны метрики для визуализации (с помощью скрипта, который приложен к заданию).
3. Программы — adminer, mysql и grafana — запущены с помощью docker-compose.
4. Источник данных MySQL, который исполняется в отдельном контейнере, добавлен в Grafana.
5. Собран дашборд, состоящий из шести простых графиков. Дашборд визуально соответствует референсу (см. скриншот под заголовком второй части работы), однако цифры и графики должны отличаться.