НИЯУ МИФИ

Лабораторная работа №3:

«Брокеры очередей и тестерование веб API»

Подготовил: Софронов Александр Евгеньевич Б21-525

2023

Сценарий

Сервис простого учета рабочего времени для сотрудников компании. Система позволяет каждому работнику быстро фиксировать время начала и окончания своей работы.

Веб - страница с формой для ввода ID сотрудника и фиксации времени начала или окончания рабочего дня.

Веб API - простые методы для добавления записи о начале и окончании работы сотрудника, используя его ID, а также методы для извлечения и сортировки данных для создания отчетов.

База данных - содержит записи о рабочих интервалах каждого сотрудника с привязкой к их ID, а также информацию для генерации агрегированных отчетов.

RPS (Requests per second) = 50-200 утром и вечером (все начинают работать и все заканчивают работать)

Стек: Простая очередь сообщений для упорядочивания входящих запросов от сотрудников и предотвращения потери данных при высокой нагрузке — RabbitMQ. Использование легковесной, но надежной базы данных, например, SQLite, для хранения данных.

Реализация

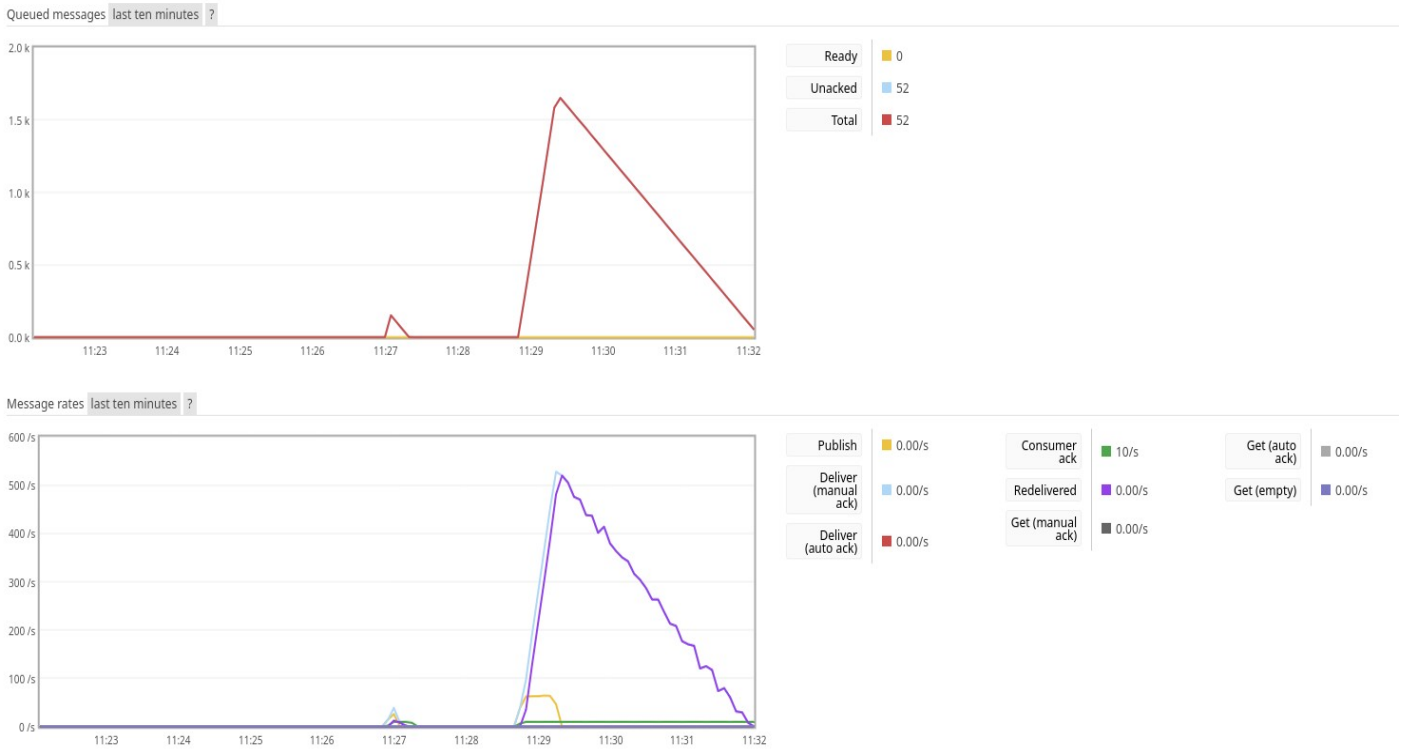
Для реализации поставленной был выбран NodeJS для серверной части и обработки API запросов. В качестве брокера очередей - RabbitMQ, которая имеет хорошие возможности для интеграции с NodeJS бекэндом серверной части. Для нагрузочного тестирования — Newman — консольный запускатор запросов для Postman с прегенерацией запросов на Python.

Все сервисы живут в отдельных Docker контейнерах. Запуск всей системы происходит с помощью Docker Compose.

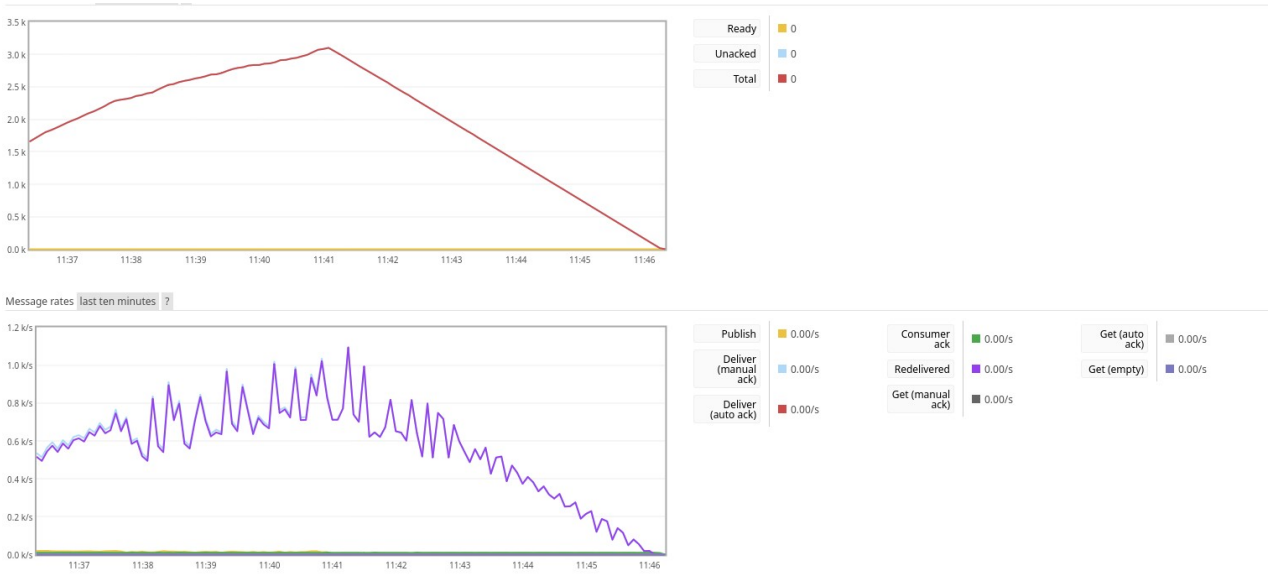
Тестирование

Размер очереди непосредственно сервера (не сервиса) искуственно ограничен до 100 активных соединений. Каждый запрос искуственно растянут на 100мс. Таким образом, пиковая нагрузка, которую способен выдержать сервер после заполнения своей очереди — 10 запросов в секунду.

Для тестирования единовременного всплеска активности было сгенерировано 2000 запросов с интервалами в 5мс — 200 запросов в секунду.



Для продолжительной нагрузки было сгенерировано 7500 запросов с интервалами 40мс — 25 запросов в секунду на протяжении 5 минут.



Тестирование показало, что, благодаря брокеру сообщений, сервис может стабильно обрабатывать поток запросов кратно большей интенсивности, чем пропускная способность сервера.

Заключение

В данной лабораторной работе было успешно создано и протестировано приложение для учёта рабочего времени сотрудников.

Реализация:

- Веб-страница: предоставляет форму для ввода ID сотрудника и фиксации начала и окончания рабочего времени.
- Веб API: включает методы для записи и извлечения данных о рабочих интервалах.
- Брокер сообщений: RabbitMQ используется для упорядочивания запросов и предотвращения их потерь при высокой нагрузке.
- База данных: SQLite для надёжного хранения данных.
- Технологии: NodeJS для серверной части, контейнеризация Docker для управления и запуска сервисов.

Тестирование:

- Ограничение сервера: 100 активных соединений, каждый запрос обрабатывался с задержкой 100 мс, максимальная пропускная способность — 10 запросов в секунду.
- Пиковая нагрузка: 2000 запросов с интервалами 5 мс (200 запросов/с).
- Длительная нагрузка: 7500 запросов с интервалами 40 мс (25 запросов/с) на протяжении 5 минут.

Результаты тестов показали, что за счёт использования RabbitMQ система способна эффективно обрабатывать значительно большее количество запросов, чем пропускная способность сервера. Это подтверждает, что выбранная архитектура позволяет создавать надежные и масштабируемые системы для учёта рабочего времени.

Гит - https://github.com/iGTsan/parvpo/tree/master/lab3