Национальный исследовательский ядерный университет «МИФИ»
Кафедра № 42 «Криптология и кибербезопасность»

Лабораторная работа № 3

«Брокеры очередей и тестирование веб API»

Тимин Александр Б21-515

Цель работы

Приобрести базовые навыки работы с брокерами очередей и тестирования веб API.

Реализуемая система

В рамках лабораторной работы был реализован сервис обращений в управляющую компанию о плачевном состоянии инфраструктуры и коммуникаций. Приложение состоит из веб API, брокера сообщений, сервиса-обработчика запросов и базы данных.

Веб API предоставляет пользовательский интерфейс (HTML-форму) для формирования заявок, принимает их и перенаправляет в очередь брокеру сообщений на основе RabbitMQ. Брокер перенаправляет заявки сервисам обработчикам сообщений (на текущий момент он пока один), который обрабатывает их (для упрощения просто логирует) и сохраняет в базу данных PostgreSQL.

Брокер сообщений необходим, так как в моменты аварий (и неожиданного выпадения снега в декабре) лавинообразно возрастает количество сообщений. Для это был выбран RabbitMQ благодаря его надежности и поддержке сложных сценариев маршрутизации сообщений. Он обеспечивает гарантированную доставку сообщений и эффективное распределение задач между несколькими воркерами, что критично для своевременной обработки обращений о неисправностях и авариях. Кроме того, его возможности управления очередями и подтверждения доставки сообщений помогают избежать потерь данных и обеспечить высокую производительность системы.

Стек технологий: Python, Flask, Pika, RabbitMQ, PostgreSQL, HTML, JS, Docker.

Тестирование

Тестирование веб-API будет проводится при помощи Postman

Реализованный веб-API принимает входящие POST запросы с данными в JSON формате с полями: name, email, subject и message. На рисунке 1 представлен скриншот тестового запроса к веб-API из Postman.

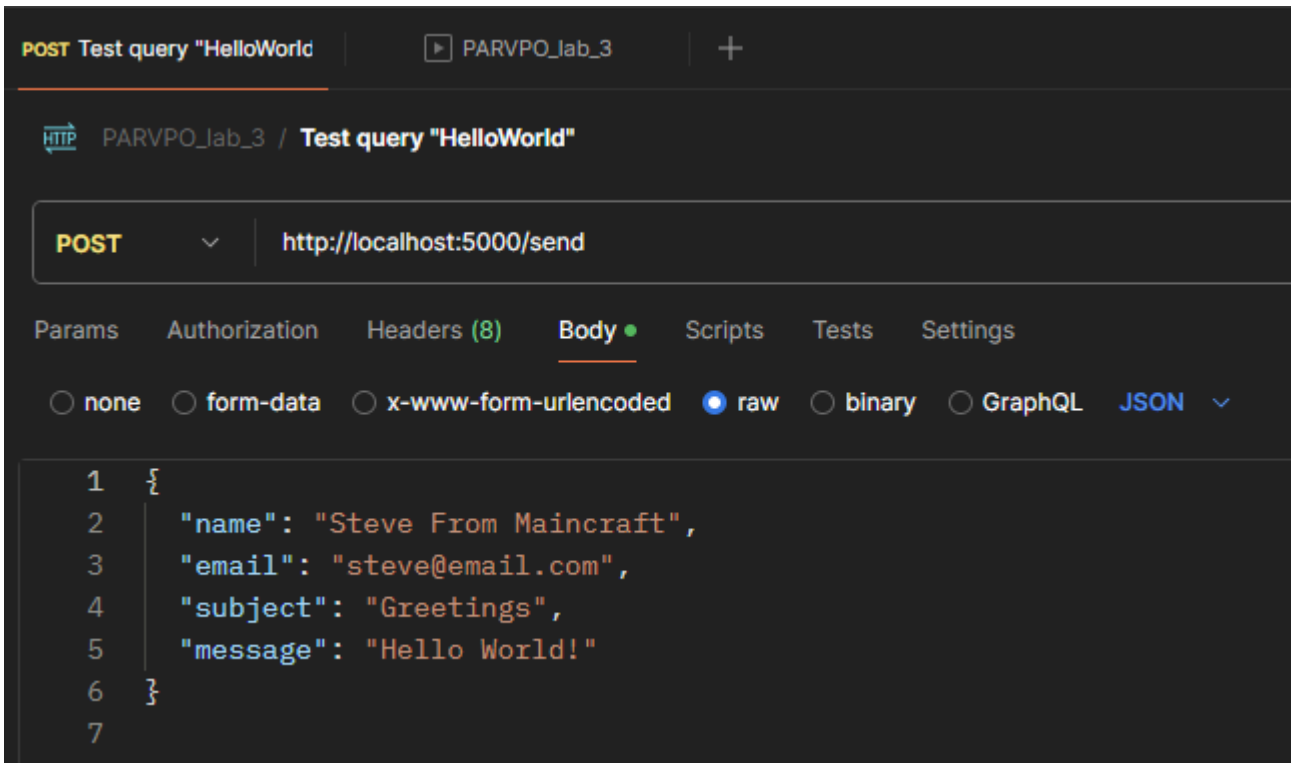

Рисунок 1 – POST-запрос к веб-API

Тестирование проводится для двух сценариев: продолжительная нагрузка (рисунок 2) и короткий всплеск активности (рисунок 3).

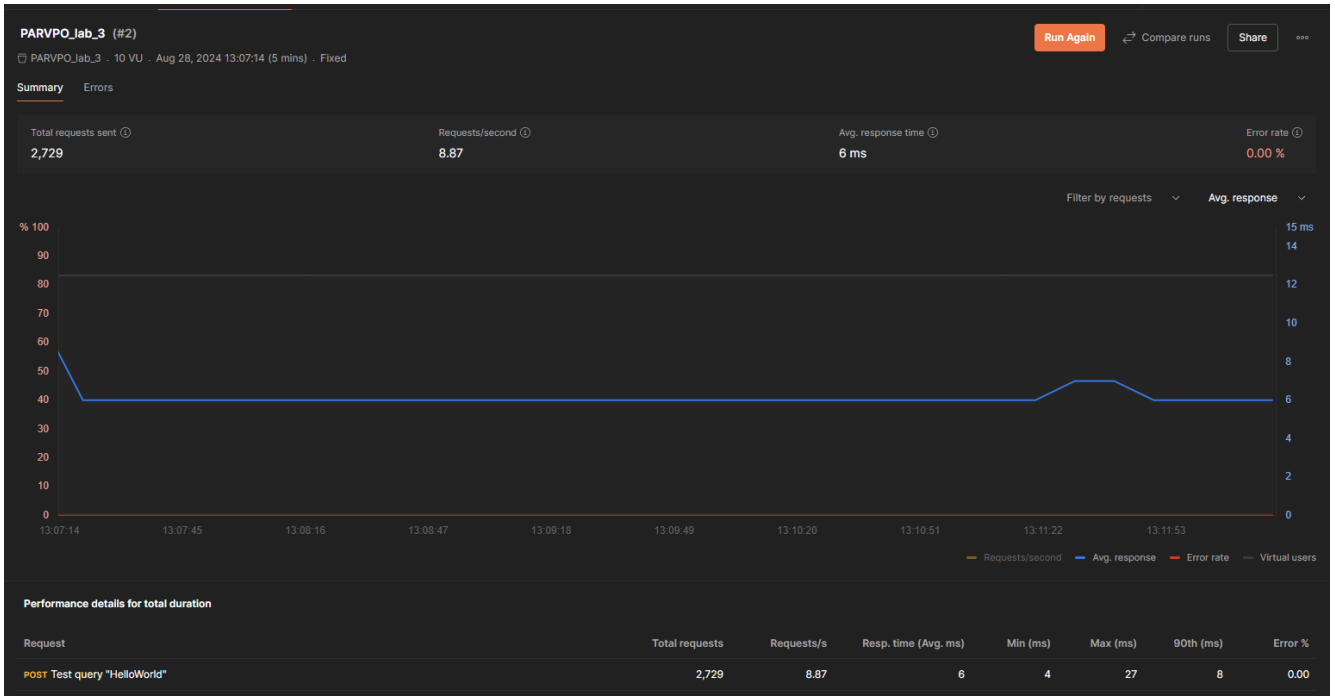

Рисунок 2 – Результаты тестирования по сценарию продолжительной нагрузки

Как видно из рисунка, с сценарием продолжительной нагрузки система справляется без проблем: 10 виртуальных пользователей параллельно отправляют запросы к API, в течение 5 минут было отправлено 2729 запросов, среднее время

ожидания ответа составило 6 миллисекунд, при этом абсолютно все запросы были успешными (более подробные результаты приведены в приложении).

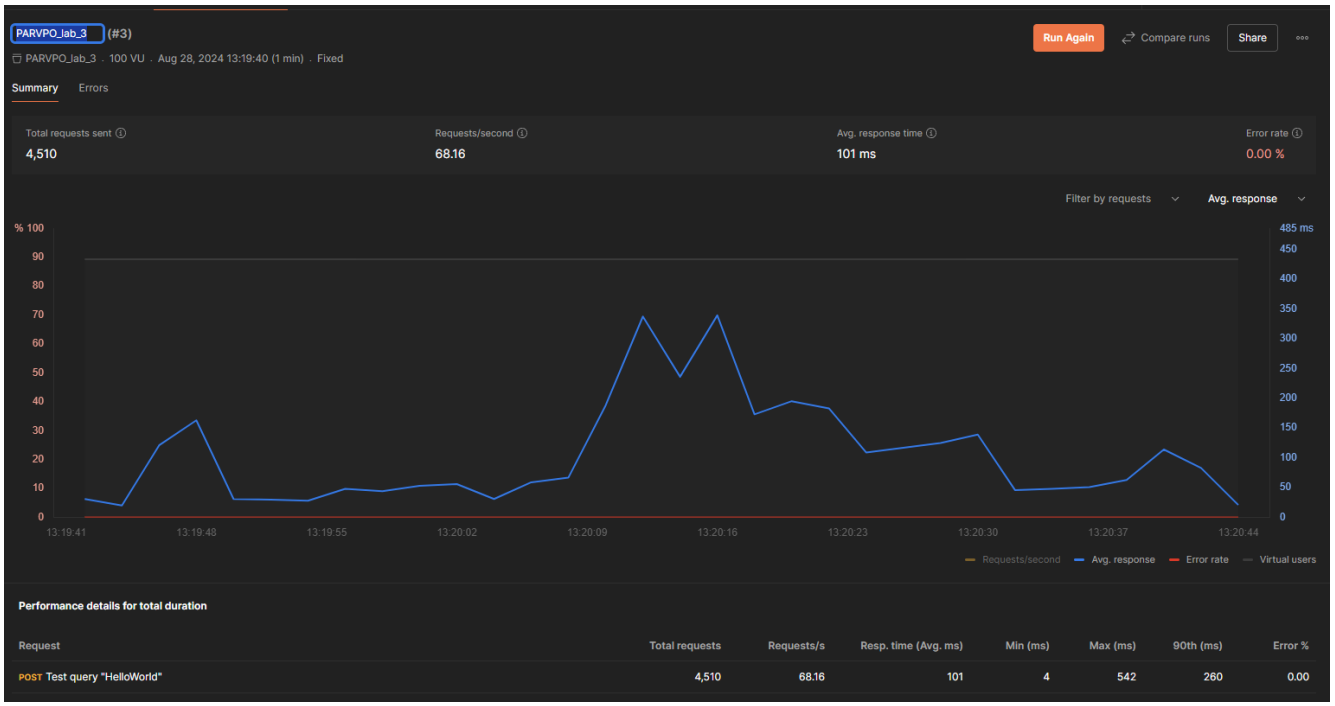

Рисунок 3 – Результаты тестирования по сценарию короткого всплеска активности

Как видно из рисунка, при всплеске активности происходит замедление системы: 100 виртуальных пользователей параллельно отправляют запросы, в течение 1 минуты было отправлено 4510 запросов со средним ожиданием ответа в 101 миллисекунд, при этом все запросы были успешными.

Заключение

В процессе выполнения данной лабораторной работы была придумана концепция и реализован сервис приема и обработки заявок жителей многоквартирного дома, а также проведено нагрузочное тестирование веб-API для двух сценариев: продолжительная нагрузка и всплеск активности. Результаты тестирования показали, что даже при всплеске активности API оперативно отвечает, при этом стабильно успешно.

Приложение

- https://github.com/KATEHOK/PARVPO_labs-6_sem/tree/main – репозиторий проекта (tag: lab_3);
- https://github.com/KATEHOK/PARVPO_labs-6_sem/blob/main/report/PARVPO_lab_3-performance-report-continious_load.pdf – результаты тестирования по сценарию продолжительная нагрузка;
- https://github.com/KATEHOK/PARVPO_labs-6_sem/blob/main/report/PARVPO_lab_3-performance-report-surge_of_activity.pdf – результаты тестирования по сценарию всплеск активности.