# ВВЕДЕНИЕ ПРО НАГРУЗОЧНОЕ ТЕСТИРОВАНИЕ

В промышленной разработке сервисов немаловажную часть занимает исследование их поведения на разных этапах их жизненного цикла. В особенности - непосредственно перед выпуском сервисов в production и уже после выпуска.

Отследить работу конкретной версии приложения и удостовериться, что оно работает так, как было задумано командой разработки, помогают различного рода тесты. Например, модульные тесты помогают отследить корректную работу отдельно взятых методов и компонентов, интеграционные тесты проверяют работу различных модулей в совокупности, допустим, взаимодействие приложения с базой данных. Сквозные тесты проверяют правильную работу всего приложения с точки зрения сценариев конечных пользователей.

Однако данные методы не дают нам ответа на вопрос о том, как сервисы повели бы себя в определенных нестандартных ситуациях, которые произойдут или могут произойти в будущем. Нужным инструментом в таком случае является нагрузочное тестирование, позволяющее выявить узкие места в работе приложения при различных профилях взаимодествия с сервисами, описать, какие пределы в производительности есть у текущей версии приложения для планирования дальнейших действий при разработке, и ответить еще на ряд технических и бизнес вопросов.

Команде разработки необходима информация о производительности сервисов при сценариях работы, отличающихся от текущих, по следующим причинам:

а) бизнес заинтересован в лояльности и хорошем опыте использования сервисов своей аудиторей. Это, в свою очередь, поможет бизнесу развиваться и вкладывать полученные ресурсы в дальнейшее развитие.

б) осуществляя регулярное нагрузочное тестирование, команда разработки имеет представление о возможностях производительности своих сервисов: какие из них хуже работают под нагрузкой, а какие лучше, сколько запросов в секунду выдерживают сервисы и какая у них скорость обработки запросов. Если команда разработки предполагает скорое расширение аудитории своих сервисов, знания о их производительности позволит заранее заложить нужные задачи для оптимизации узких мест и производительности сервисов. Это, в свою очередь, поможет сервисам пережить пики нагрузки без ущерба для пользовательского опыта.

в) механизм нагрузочного тестирования можно встроить в цикл выпуска новых версий сервиса, тем самым заранее будет возможно получать информацию о том, как очередная версия сервиса скажется на его производительности.

г) на ранних этапах разработки новых сервисов нагрузочное тестирование позволит разработчикам сделать выбор в пользу тех или иных технологий или алгоритмов, поможет избежать внесения крупных изменений в код в случае неудовлетворительной производительности на более поздних стадиях разработки проекта.

Разобравшись в проблемах, которые можно решить с помощью нагрузочного тестирования, рассмотрим, какие есть подходы к проведению нагрузочного тестирования и как ими пользоваться.

# МЕТОДОЛОГИЯ НАГРУЗОЧНОГО ТЕСТИРОВАНИЯ

## Требования к сценариям нагрузочных тестов

В зависимости от требований к определенным сценариям, нагрузочные тесты могут проводиться по-разному. Кроме того, интерпретация результатов тестирования также напрямую зависит от этих требований. Разберем некоторые примеры требований:

1. В API-сервисе присутствует метод, производительность которого очень сильно зависит от переданных параметров. Например, имеем метод получения первых “n” документов пользователя. В зависимости от числа “n” метод будет отрабатывать за разное время. С помощью нагрузочного тестирования хотим проверить, как будет изменяться производительность метода вместе с ростом числа запрашиваемых документов. По результатам нагрузочного тестирования сможем подобрать оптимальную верхнюю границу для числа запрашиваемых документов.
2. В связи с приближающимися праздниками ожидается всплеск активности пользователей мессенджера, в два раза превышающий текущий уровень. Требуется через нагрузочное тестирование проверить, сможет ли сервис выдержать двойную нагрузку с примерно таким же соотношением запросов по методам и с такими же параметрами.
3. С целью мониторинга производительности каждой новой версии сервиса, внедряется нагрузочное тестирование со следующими требованиями:
   1. тестирование должно быть встроено в CI/CD pipeline
   2. результаты должны быть загружены в какую-то систему, позволяющую удобно анализировать результаты в сравнении с запусками нагрузочного тестирования при выпуске предыдущих версий
   3. профиль нагрузки должен как можно точнее повторять профиль с production площадки
   4. количество отправляемых запросов должно быть подобрано таким образом, чтобы при нем было задействовано 80-90% того ресурса, который сервис потребляет больше всего

Как можно заметить, в зависимости от требований, выдвинутых сценарию нагрузочных тестов,