МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

##### ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

**ЛАБОРАТОРНАЯ РАБОТА №4**

**по дисциплине**

**«Тестирование программного обеспечения»**

Вариант №3409

##### ***Выполнил:*** Студент группы P3334 Баянов Равиль Динарович

#### Преподаватель:

##### Бострикова Дарья

##### Константиновна

Санкт-Петербург

2025

**Оглавление**

[**Задание** 3](#_Toc199878142)

[**Описание конфигурации JMeter для нагрузочного тестирования** 4](#_Toc199878143)

[**Графики пропускной способности приложения, полученные в ходе нагрузочного тестирования** 5](#_Toc199878144)

[**Выводы по выбранной конфигурации аппаратного обеспечения** 6](#_Toc199878145)

[**График изменения времени отклика от нагрузки для выбранной конфигурации, полученный в ходе стресс-тестирования системы** 8](#_Toc199878146)

[**Вывод** 9](#_Toc199878147)

# **Задание**

С помощью программного пакета [Apache JMeter](http://jmeter.apache.org/) провести нагрузочное и стресс-тестирование веб-приложения в соответствии с вариантом задания.

В ходе нагрузочного тестирования необходимо протестировать 3 конфигурации аппаратного обеспечения и выбрать среди них наиболее дешёвую, удовлетворяющую требованиям по максимальному времени отклика приложения при заданной нагрузке (в соответствии с вариантом).

В ходе стресс-тестирования необходимо определить, при какой нагрузке выбранная на предыдущем шаге конфигурация перестаёт удовлетворять требованиями по максимальному времени отклика. Для этого необходимо построить график зависимости времени отклика приложения от нагрузки.

Приложение для тестирования доступно только во внутренней сети кафедры.

Если запрос содержит некорректные параметры, сервер возвращает HTTP 403.

Если приложение не справляется с нагрузкой, сервер возвращает HTTP 503.

**Параметры тестируемого веб-приложения:**

* URL первой конфигурации ($ 2300) - [http://stload.se.ifmo.ru:8080?token=495351469&user=-2105057934&config=1](http://stload.se.ifmo.ru:8080/?token=495351469&user=-2105057934&config=1);
* URL второй конфигурации ($ 3100) - [http://stload.se.ifmo.ru:8080?token=495351469&user=-2105057934&config=2](http://stload.se.ifmo.ru:8080/?token=495351469&user=-2105057934&config=2);
* URL третьей конфигурации ($ 4200) - [http://stload.se.ifmo.ru:8080?token=495351469&user=-2105057934&config=3](http://stload.se.ifmo.ru:8080/?token=495351469&user=-2105057934&config=3);
* Максимальное количество параллельных пользователей - 13;
* Средняя нагрузка, формируемая одним пользователем - 40 запр. в мин.;
* Максимально допустимое время обработки запроса - 640 мс.

# **Описание конфигурации JMeter для нагрузочного тестирования**

Весь проект делится на 3 конфигурации, они отличаются между собой только параметром config, который мы будем передавать в GET запросе. В разделах *config*

указывается общая информация о тесте, такая как: общее количество пользователей (потоков), Ramp-up period, который указывает нам в течение, какого времени запустятся все потоки и Loop Count – кол-во тестов, которое мы хотим запустить.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Содержимое, созданное искусственным интеллектом, может быть неверным.

Затем, в каждой конфигурации отдельно настраиваются samplers, timers, assertions и listeners.

Http Request – запросы к самому серверу (sampler).

Здесь указывается порт, который мы слушаем и указываем сервер, которому будем посылать запросы.

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Duration Assertions – порог, который помогает отслеживать допустимое время отклика сервера (Assertion).

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

Constant Throughput Timer – это таймер, который позволяет приблизить значение пропускной способности всего теста к определенному значению (timer).

Изображение выглядит как текст, снимок экрана, дисплей, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

View Results in Table – это компонент, позволяющий видеть обработку запросов в виде таблице (listener).

Изображение выглядит как текст, снимок экрана, число, Параллельный

Содержимое, созданное искусственным интеллектом, может быть неверным.

Graph Result – компонент, который позволяет увидеть результаты тестирования в виде графика.

Изображение выглядит как текст, снимок экрана, линия, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

Также в конфигурации присутствуют компонент для оценки общей картины тестирования, они помогают сравнить 3 конфигурации между собой.

Response Time Graph – компонент, который строит график зависимости времени отклика от всего времени.

Aggregate Graph – компонент, который позволяет увидеть график минимального, максимального, среднего значения наших конфигураций по времени отклика.

Aggregate Report – то, же что и aggregate graph, только позволяет нам увидеть результат в табличном виде и сохранить его в csv формат.

# **Графики пропускной способности приложения, полученные в ходе нагрузочного тестирования**

Config 1

Изображение выглядит как текст, снимок экрана, линия, диаграмма

Содержимое, созданное искусственным интеллектом, может быть неверным.

Config 2

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Config 3

Изображение выглядит как текст, снимок экрана, линия, График

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, логотип, диаграмма

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, График, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как снимок экрана, текст, Прямоугольник, диаграмма

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как линия, График, текст, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, программное обеспечение, График

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как снимок экрана, диаграмма, График, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

# **Выводы по выбранной конфигурации аппаратного обеспечения**

Проанализировав данные графики, можем с уверенностью сказать, что нам подходит только конфигурация под номером 3, так как именно в ней мы не получили ни одной ошибки при тестировании. `При этом конфигурация 2 в целом почти уложилось в нужный нам диапазон времени отклика, но всё равно недостаточно. Заметим, также что в 3 конфигурации сильно падает пропускная способность по сравнению с другими конфигурациями, но это не мешает нам выбрать 3 конфигурацию так как, пропускная способность 2 и 1 конфигурации хоть и постоянная, но она всё же не укладывается в рамки нашего варианта.

**Описание конфигурации JMeter для стресс-тестирования**

Для стресс-тестирования остановимся на 3 конфигурации, так как только она удовлетворяет всем нужным условиям. Попробуем увеличить количество пользователей до 300, уберём assertion, а всё остальное останется таким же.

Изображение выглядит как текст, снимок экрана, число, программное обеспечение

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, число, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

# **График изменения времени отклика от нагрузки для выбранной конфигурации, полученный в ходе стресс-тестирования системы**

310 пользователей

Изображение выглядит как линия, График, диаграмма, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, логотип, диаграмма

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как линия, График, текст, диаграмма

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, программное обеспечение, линия, График

Содержимое, созданное искусственным интеллектом, может быть неверным.

Видим, что при 310 пользователях мы получаем ошибку 503. Попробуем теперь уменьшить количество юзеров, чтобы сайт успевал обрабатывать нагрузку.

Возьмём 300 пользователей.  
Изображение выглядит как текст, снимок экрана, Шрифт, логотип

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, линия, снимок экрана, График

Содержимое, созданное искусственным интеллектом, может быть неверным.

Заметим, что теперь сервер справляется с нагрузкой, но мы всё равно не укладываемся во временной интервал, данный нам по варианту.

Возьмём теперь 150 пользователей.

Изображение выглядит как текст, снимок экрана, программное обеспечение, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, График, линия, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Теперь видим, что сайт справляется с нагрузкой и мы укладываемся в диапазон 640 мс времени отклика.

# **Вывод**

Выполнив данную лабораторную работу, я на практике познакомился с нагрузочным и стресс тестированием. Вспомнил как правильно конфигурировать программу JMeter для правильного тестирования. Также выяснил какая конфигурация больше подходит к моему заданию.