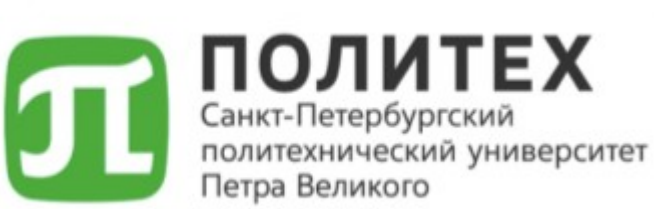
Федеральное государственное автономное образовательное учреждение высшего образования
«Санкт-Петербургский политехнический университет Петра Великого»
Институт компьютерных наук и кибербезопасности
Программная инженерия




# Отчет по курсовой работе

по дисциплине «Конструирование программного обеспечения»

на тему «**Инструмент коллаборативного автоматизированного перевода (CAT-программа)**»

Руководитель Юркин В. А.

Санкт-Петербург
2026

# Оглавление

# Woofie

Инструмент коллаборативного автоматизированного перевода (CAT-программа)

GitHub Репозиторий Woofie: https://github.com/odintsovks/Woofie

GitHub Репозиторий Woofie-Server: https://github.com/odintsovks/Woofie-Server

GitHub Репозиторий Woofie-Client: https://github.com/odintsovks/Woofie-Client

## Участники проекта


Одинцов Кирилл Сергеевич (гр. 5130904/30106)

Дмитриев Арсений Эдуардович (гр. 5130904/30106)

Засульский Егор Андреевич (гр. 5130904/30106)

Маланьин Никита Александрович (гр. 5130904/30106)

Проселков Станислав Павлович (гр. 5130904/30106)

Горбунова Алена Евгеньевна (гр. 5130904/30106)


## Технологический стек

- Серверная часть: Java Spring Framework
- Базы данных: PostgreSQL
- Внешние зависимости: интеграция с сервисами машинного перевода (к примеру «Яндекс.Переводчик» с использованием API-ключа)
- UI: Qt 6

## Определение проблемы

Отсутствие открытых CAT-программ позволяющих организовывать проекты в виде дерева. Вместо этого существующие инструменты ограничиваются табличным отображением единиц перевода, что осложняет процесс, когда последовательность текста зависит от конкретных условий (к примеру «деревья диалогов» в видеоиграх).

# Основные требования

- Возможность использования древовидной структуры для отображения единиц перевода.

Когда на практике переводимый текст зависит от строго заданных условий, я хочу иметь возможность отражать это структурой единиц перевода в CAT-программе, чтобы не путаться в исключительно последовательном табличном отображении

- Возможность делать запросы в сервисы машинного перевода и проверки правописания.

Когда возникает необходимость сверить смысл или правописание, я хочу использовать быстродоступные сервисы машинного перевода и правописания, чтобы не выбиваться из процесса перевода.

- Возможность редактировать единицы перевода в режиме реального времени.

Когда другой пользователь тем или иным образом изменил единицу перевода, я хочу увидеть это изменение без нужды ручного обновления, чтобы всегда видеть актуальное состояние проекта.

Ожидается, что данным набором инструментов (клиентом и сервером) будут пользоваться команды переводчиков текста, которые обычно являются небольшими, но программы строятся с рассчётом на максимум 10к пользователей в сутки. Как правило проекты завершаются меньше, чем за год, период хранения информации соответствующий.

# Разработка архитектуры и детальное проектирование

## Характер нагрузки

- R/W нагрузка - ожидается более высокая интенсивность чтения. При работе с текстом программа-клиент будет активно производить операции вставки в базу данных, но при этом текст должен автоматически синхронизироваться между всеми клиентами, таким образом чтение каждого нового изменения в базу данных растёт с количеством пользователей.
- Объём трафика - подавляюще большую часть трафика будут занимать текстовые данные единиц перевода, оценить объём можно следующим образом:
    - Одну единицу перевода можно оценить в 1 КиБ (достаточно примерно на 1024 / 2 байта на символ / 4 букв на слово = 128 слов в кодировке UTF-8), один пользователь за рабочую неделю сделает примерно 500 изменений в текст (40 часов * 60 / 5 минут на изменение = 480 изменений) изменений в текст, соответственно количество данных на запись оценивается в 1 КиБ * 500 / 7 = 72 КиБ на пользователя в день. При этом количество данных на чтение дополнительно растёт с количеством пользователей, т.к. каждый пользователь должен иметь доступ к истории изменений.
    - Запись:
        - Типовой случай: 72 КиБ/день * 100 пользователей = 7200 КиБ/день = 7 МиБ/день
        - Худший случай: 72 КиБ/день * 10000 пользователей = 720000 КиБ/день = 700 МиБ/день
    - Чтение:
        - Типовой случай: 7 МиБ/день * 100 пользователей = 700 МиБ/день
        - Худший случай: 7 МиБ/день * 10000 пользователей = 70000 МиБ/день = 70 ГиБ/день

    - На практике количество данных на запись/чтение будет ограничено сверху количеством единиц перевода.
- Объёмы дисковой системы - как в случае с объёмом трафика, большую часть дискового пространства будут занимать текстовые данные; в зависимости от размера проекта количество единиц перевода может превышать десятки, если не сотни тысяч. Как правило одна единица требует не больше двух-трёх итераций (черновой вариант + вычитка), соответственно:
    - Типовой случай: 1 КиБ/ед.п. * 10000 ед.п. * 3 итерации = 30000 КиБ = 29 МиБ
    - Худший случай: 1 КиБ/ед.п. * 100000 ед.п. * 3 итерации = 300000 КиБ = 293 МиБ

## API

### 1. Работа с текстом

### GET /translations

Описание: запрос всех единиц перевода (пар "оригинал-перевод")

- Статус: 200 OK
    - Тело:

```
{
  "timestamp": 0,
  "translations": [
    {
      "id": "",
      "sourceText": "",
      "targetText": "",
      "connections": [
        {
          "id": "",
          "description": ""
        },
        // ...
      ]
    },
    // ...
  ]
}
```

Требования по максимальному времени отклика: 5 секунд (запрос производится при запуске клиента, ожидаем что формирование запроса может быть медленным, а время отклика некритично на этапе инициализации программы).

### POST /translations-fetch-updates

Описание: запрос обновлённых единиц перевода с определённой отметки времени

- Тело запроса:

```
{
"timestamp": 0
}
```

- Статус: *200 OK*

- Тело:

```
[
  {
    "id": "",
    "sourceText": "",
    "targetText": "",
    "connections": [
      {
        "id": "",
        "description": ""
      },
      // ...
    ]
  },
  // ...
]
```

- Статус: *400 Bad Request*
  - Неправильно сформирован запрос; отсутствует поле *timestamp*

Требования по максимальному времени отклика: 1 секунда (ожидается, что программа будет запрашивать свежие обновления текстовых данных раз в 1-2 секунды)

## POST /translations

Описание: добавить единицу перевода

- Тело запроса:

```
{
"sourceText": "",
"targetText": "",
"connections": [
      {
      "id": "",
      "description": ""
      },
      // ...
  ]
}
```

- Статус: *201 Created*
  - Тело:

```
{
   "id": "",
   "sourceText": "",
   "targetText": "",
   "connections": [
     {
         "id": "",
         "description": ""
     },
     // ...
   ]
}
```

- Статус: *400 Bad Request*
    - Неправильно сформирован запрос; отсутствует одно из полей

## GET /translations/{id}

Описание: запросить единицу перевода по заданному индексу

Статус: *200 OK*

- Тело:

```
{
  "sourceText": "",
  "targetText": "",
  "connections": [
    {
      "id": "",
      "description": ""
    },
    // ...
  ]
}
```

•Статус: *404 Not Found*

•Единицы перевода не сущесвтует по заданному индексу

## PUT /translations/{id}

Описание: установить значения единицы перевода по заданному индексу (добавить если не существует)

- Тело запроса:

```
{
  "sourceText": "",
  "targetText": "",
  "connections": [
    {
      "id": "",
```

```
      "description": ""
    },
    // ...
  ]
}
```

- Статус: *200 OK* и *201 Created*
  - Тело

```
{
  "id": "",
  "sourceText": "",
  "targetText": "",
  "connections": [
    {
      "id": "",
      "description": ""
    },
    // ...
  ]
}
```

- Статус: *400 Bad Request*
  - Неправильно сформирован запрос; отсутствует одно из полей.

### DELETE /translations/{id}

Описание: удалить единицу перевода по заданному индексу

- Статус: *204 No Content*
- Статус: *404 Not Found*
  - Единицы перевода не существовало по заданному индексу перед запросом на удаление

## 2. Работа с глоссарием

### GET /glossary

Описание: запрос списка записей глоссария

- Статус: *200 OK*
  - Тело:

```
{
  "timestamp": 0,
  "entries": [
    {
      "id": "",
      "targetTerm": "",
      "sourceTerm": "",
      "definition": ""
    },
```

```
      // ...
    ]
}
```

Требования по максимальному времени отклика: 5 секунда (аналогично с GET /translations)

## POST /glossary-fetch-updates

Описание: запрос обновлённых записей глоссария с определённой отметки времени

- Тело запроса:

```
{
  "timestamp": 0
}
```

- Статус: *200 OK*

  Тело:

```
[
  {
    "id": "",
    "targetTerm": "",
    "sourceTerm": "",
    "definition": ""
  },
  // ...
]
```

- Статус: *400 Bad Request*
  - Неправильно сформирован запрос; отсутствует поле *timestamp*

Требования по максимальному времени отклика: 5 секунд (аналогично с GET /translations-fetch-updates, но ожидаем, что обновления глоссария запрашиваются реже).

## POST /glossary

Описание: добавить запись в глоссарий.

- Тело запроса:

```
{
  "targetTerm": "",
  "sourceTerm": "",
  "definition": ""
}
```

- Статус: *201 Created*
  - Тело:

```
{
  "id": "",
```

```
  "targetTerm": "",
  "sourceTerm": "",
  "definition": ""
}
```

•Статус: *400 Bad Request*

•Неправильно сформирован запрос; отсутствует одно из полей.

## GET /glossary/{id}

Описание: запросить запись из глоссария по индексу

- Статус: *200 OK*
  - Тело:

```
{
  "targetTerm": "",
  "sourceTerm": "",
  "definition": ""
}
```

- Статус: *404 Not Found*
  - Записи глоссария не сущесвтует по заданному индексу

## PUT /glossary/{id}

Описание: изменить запись глоссария по индексу (добавить если не существует)

- Тело запроса:

```
{
  "targetTerm": "",
  "sourceTerm": "",
  "definition": ""
}
```

- Статус: *200 OK* и *201 Created*
  - Тело

```
{
  "id": ""
  "targetTerm": "",
  "sourceTerm": "",
  "definition": ""
}
```

- Статус: *400 Bad Request*
  - Неправильно сформирован запрос; отсутствует одно из полей.

## DELETE /glossary/{id}

Описание:

- Статус: *204 No Content*
- Статус: *404 Not Found*
    - Записи глоссария не существовало по заданному индексу перед запросом на удаление

## 3. Машинный перевод

### GET /translation-services

Описание: запросить информацию о всех доступных сервисах машинного перевода

- Статус: *200 OK*
    - Тело:

```
[
  {
    "id": "",
    "name": ""
  },
  //...
]
```

### POST /translate/{id}

Описание: запросить перевод строки у сервиса машинного перевода по заданному индексу

- Тело запроса:

```
{
  "text": ""
}
```

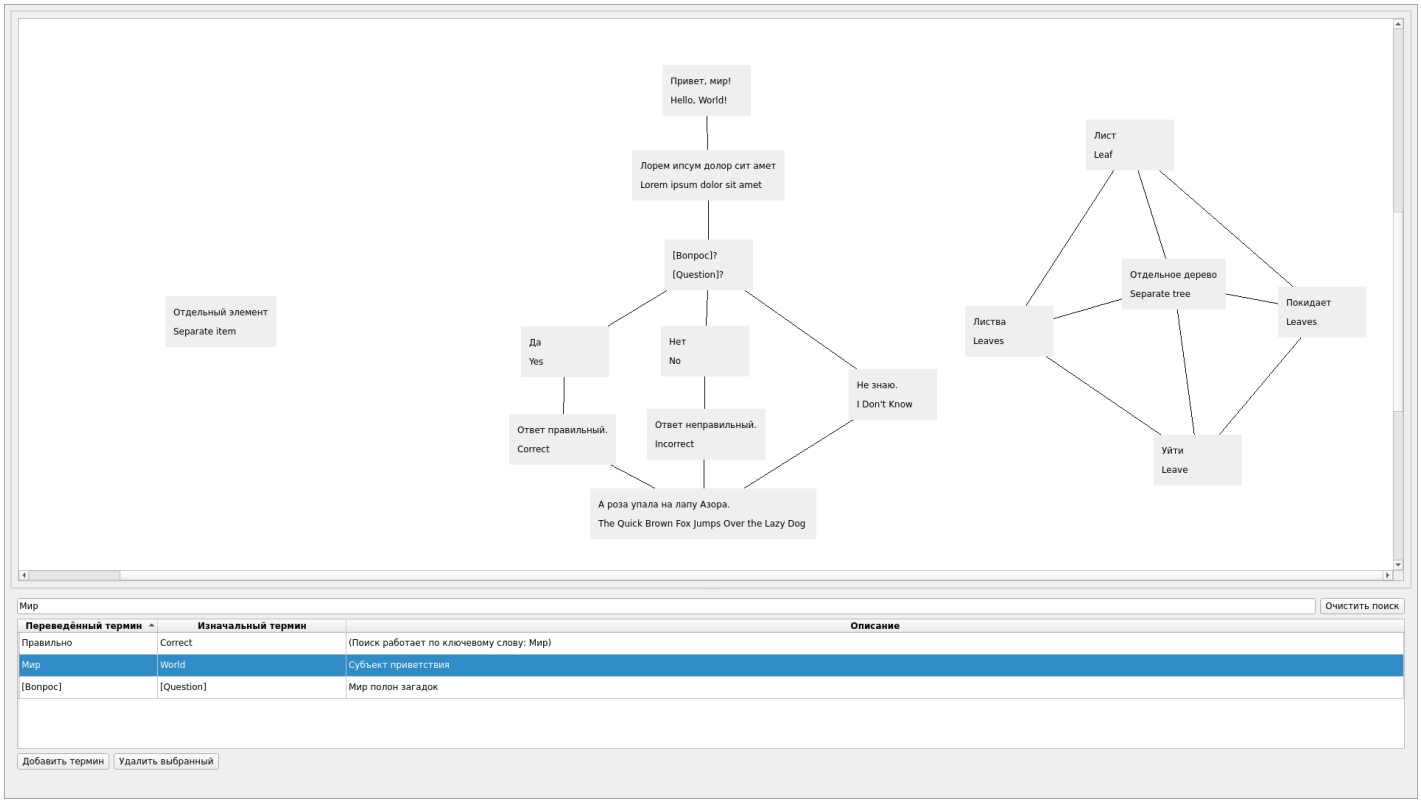
- Статус: *200 OK*
    - Тело:

```
{
  "text": ""
}
```

- Статус: *503 Service Unavailable*
    - Сервис перевода не доступен; сервер не получил ответ

Требования по максимальному времени отклика: N/a (полностью зависит от доступности и времени отклика стороннего сервиса)

# Демонстрация работы




# Тестирование

**Тестовые запросы http к серверу:**

**### 1. GET /api/translations (Запрос всех единиц)**

# Ожидаемый статус: *200 OK*

# Ожидаемое тело: { "timestamp": ..., "translations": [...] }

GET http://localhost:8080/api/translations

**### 2. POST /api/translations (Добавление единицы)**

# Ожидаемый статус: *201 Created*

POST http://localhost:8080/api/translations

Content-Type: application/json

{

"sourceText": "Hello world",

```
  "targetText": "Привет мир",
  "connections": [
    { "description": "First connection" }
  ]
}
```

**### 3. POST /api/translations (Ошибка 400 - отсутствует поле)**

```
# Ожидаемый статус: 400 Bad Request
POST http://localhost:8080/api/translations
Content-Type: application/json

{
  "targetText": "Only target"
}
```

**### 4. GET /api/translations/{id} (Запрос по ID)**

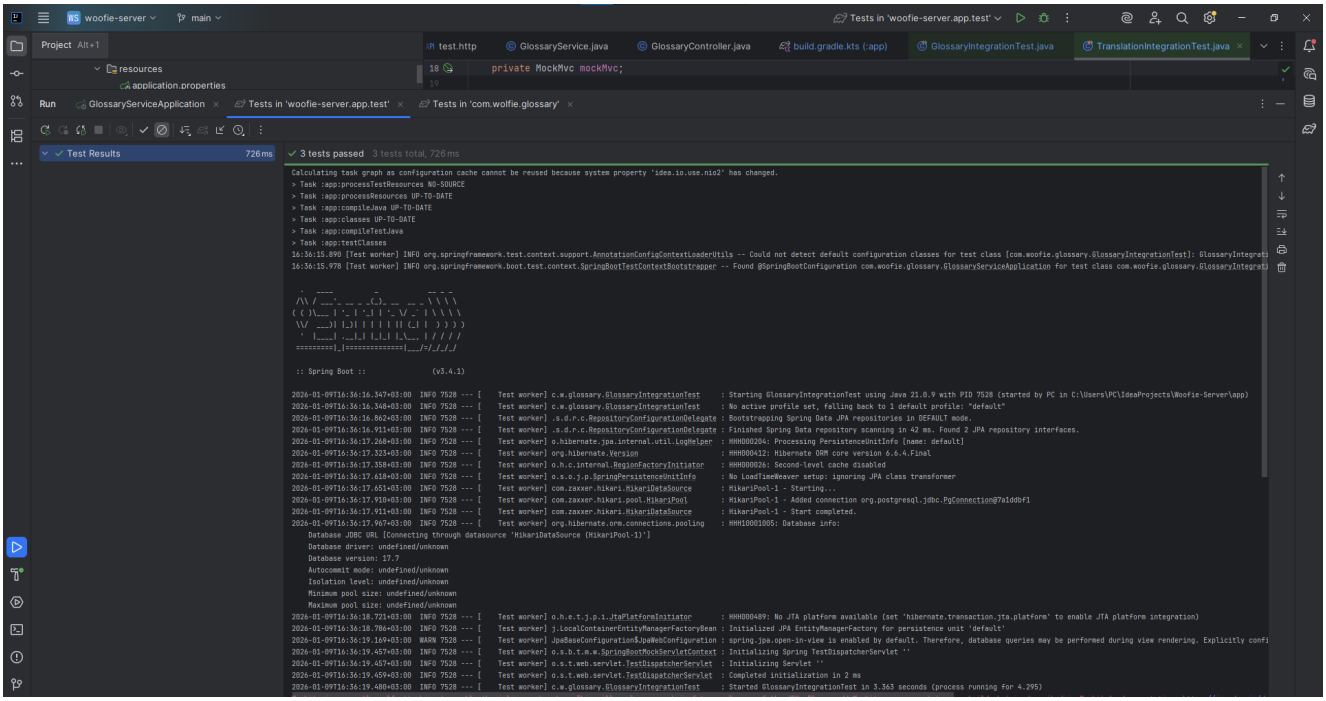
```
# Ожидаемый статус: 200 OK
GET http://localhost:8080/api/translations/2
```

**### 5. DELETE /api/translations/{id} (Удаление)**

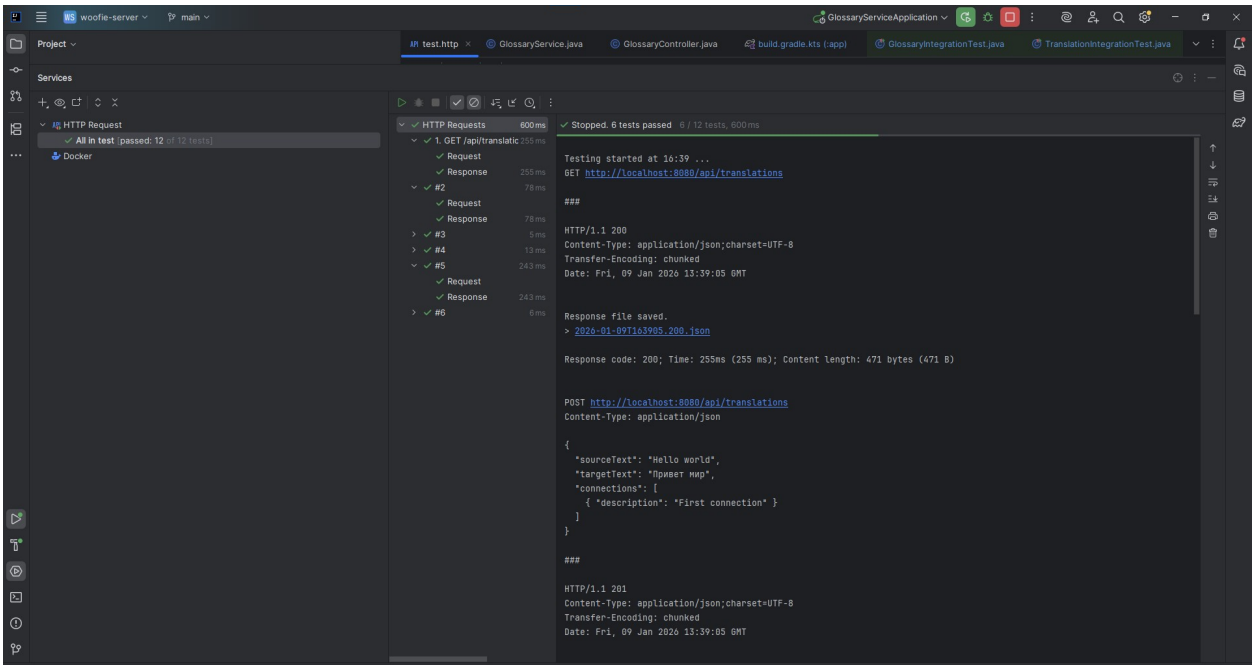
```
# Ожидаемый статус: 204 No Content
DELETE http://localhost:8080/api/translations/1
```

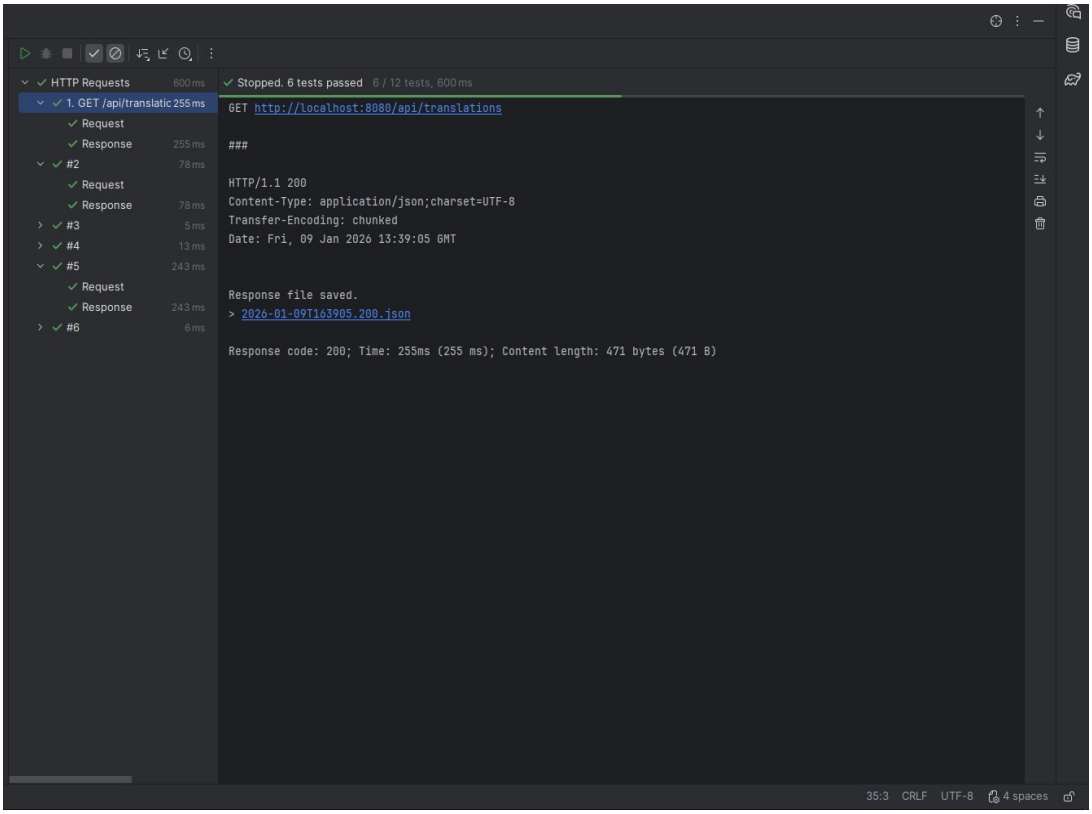
**### 6. DELETE /api/translations/{id} (Ошибка 404 - уже удалено)**

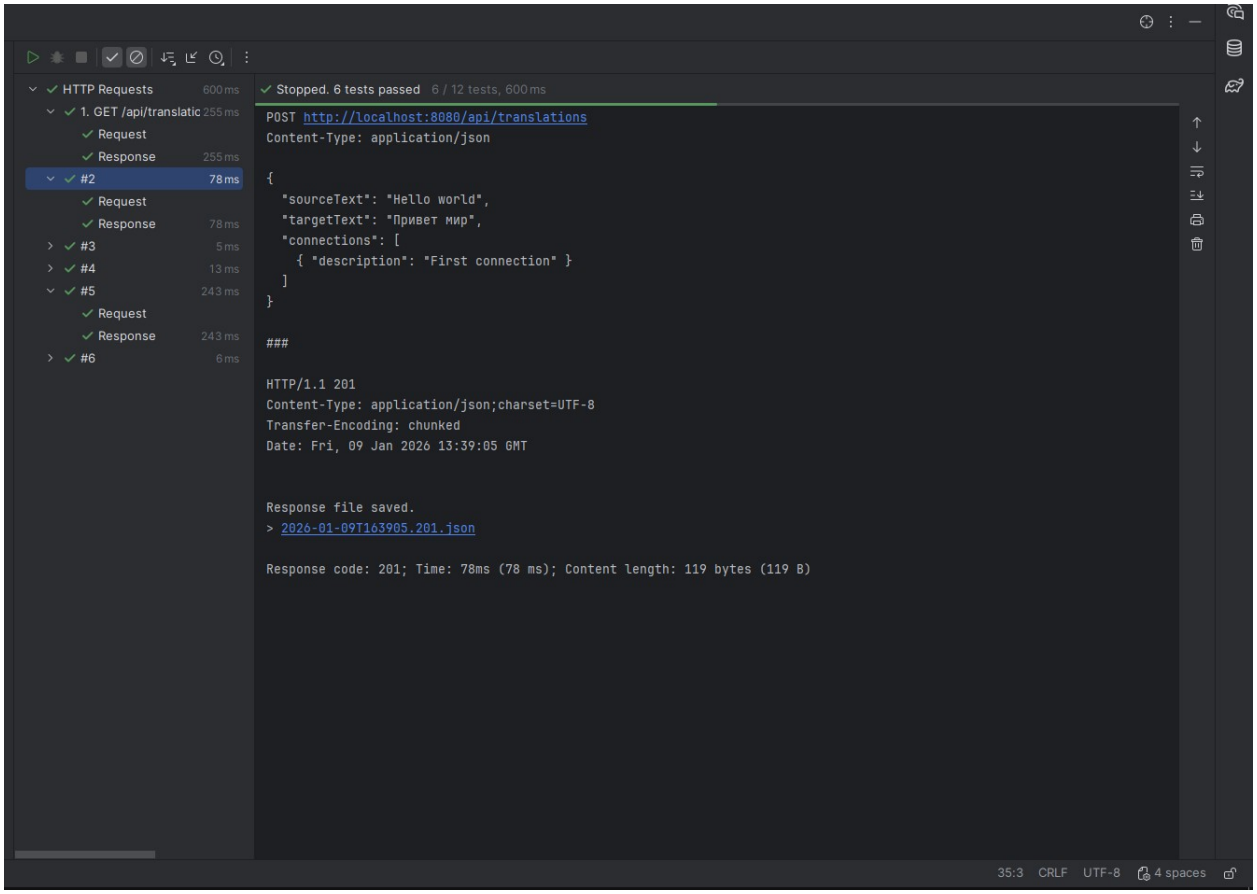
```
# Ожидаемый статус: 404 Not Found
DELETE http://localhost:8080/api/translations/1
```

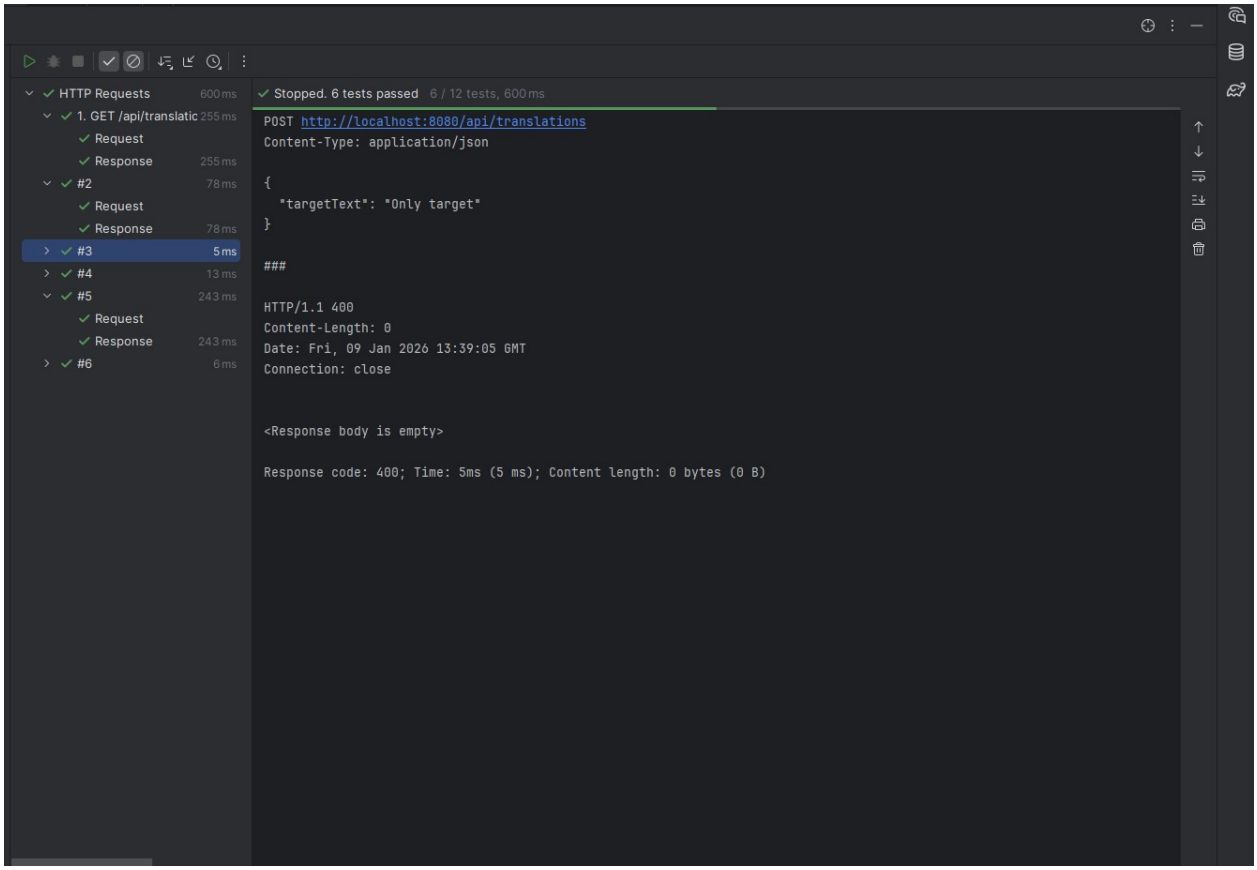


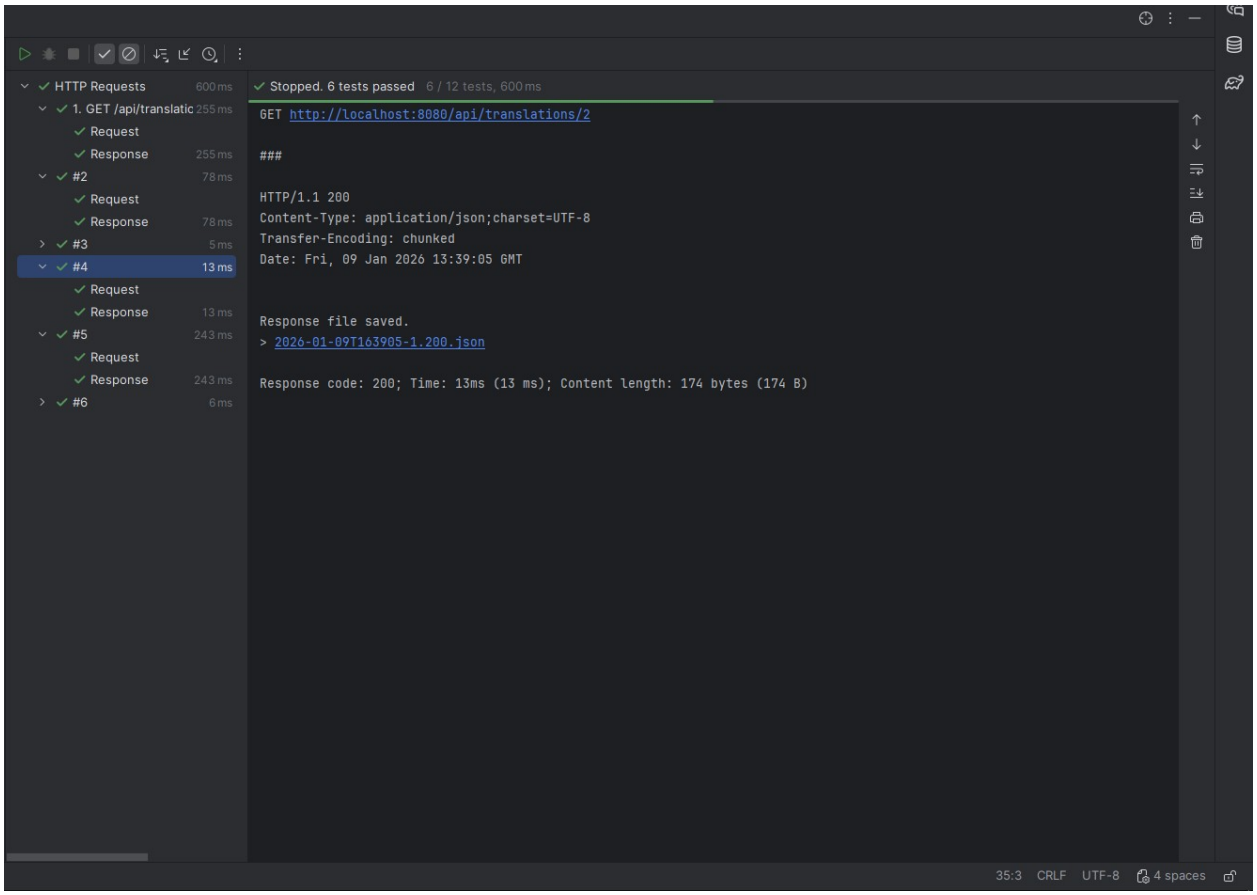

```
2026-01-09T16:36:20.155+03:00  INFO 7528 --- [    Test worker] t.c.s.AnnotationConfigContextLoaderUtils : Could not detect default configuration classes for test class [com.woofie.glossary.TranslationIntegrationTest]: TranslationIntegrat
2026-01-09T16:36:20.157+03:00  INFO 7528 --- [    Test worker] .b.t.c.SpringBootTestContextBootstrapper : Found @SpringBootConfiguration com.woofie.glossary.GlossaryServiceApplication for test class com.woofie.glossary.TranslationIntegra
Hibernate: insert into translations (source_text,target_text,updated_at) values (?,?,?)
Hibernate: insert into translation_connections (description,translation_id) values (?,?)
Hibernate: insert into translation_connections (description,translation_id) values (?,?)
Hibernate: select t1_0.id,t1_0.source_text,t1_0.target_text,t1_0.updated_at from translations t1_0
Hibernate: select c1_0.translation_id,c1_0.id,c1_0.description from translation_connections c1_0 where c1_0.translation_id=?
Hibernate: select c1_0.translation_id,c1_0.id,c1_0.description from translation_connections c1_0 where c1_0.translation_id=?
Hibernate: select c1_0.translation_id,c1_0.id,c1_0.description from translation_connections c1_0 where c1_0.translation_id=?
Hibernate: select t1_0.id,t1_0.source_text,t1_0.target_text,t1_0.updated_at from translations t1_0 where t1_0.id=?
2026-01-09T16:36:20.235+03:00  INFO 7528 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for persistence unit 'default'
2026-01-09T16:36:20.237+03:00  INFO 7528 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Shutdown initiated...
2026-01-09T16:36:20.240+03:00  INFO 7528 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Shutdown completed.
> Task :app:test
BUILD SUCCESSFUL in 8s
4 actionable tasks: 2 executed, 2 up-to-date
```

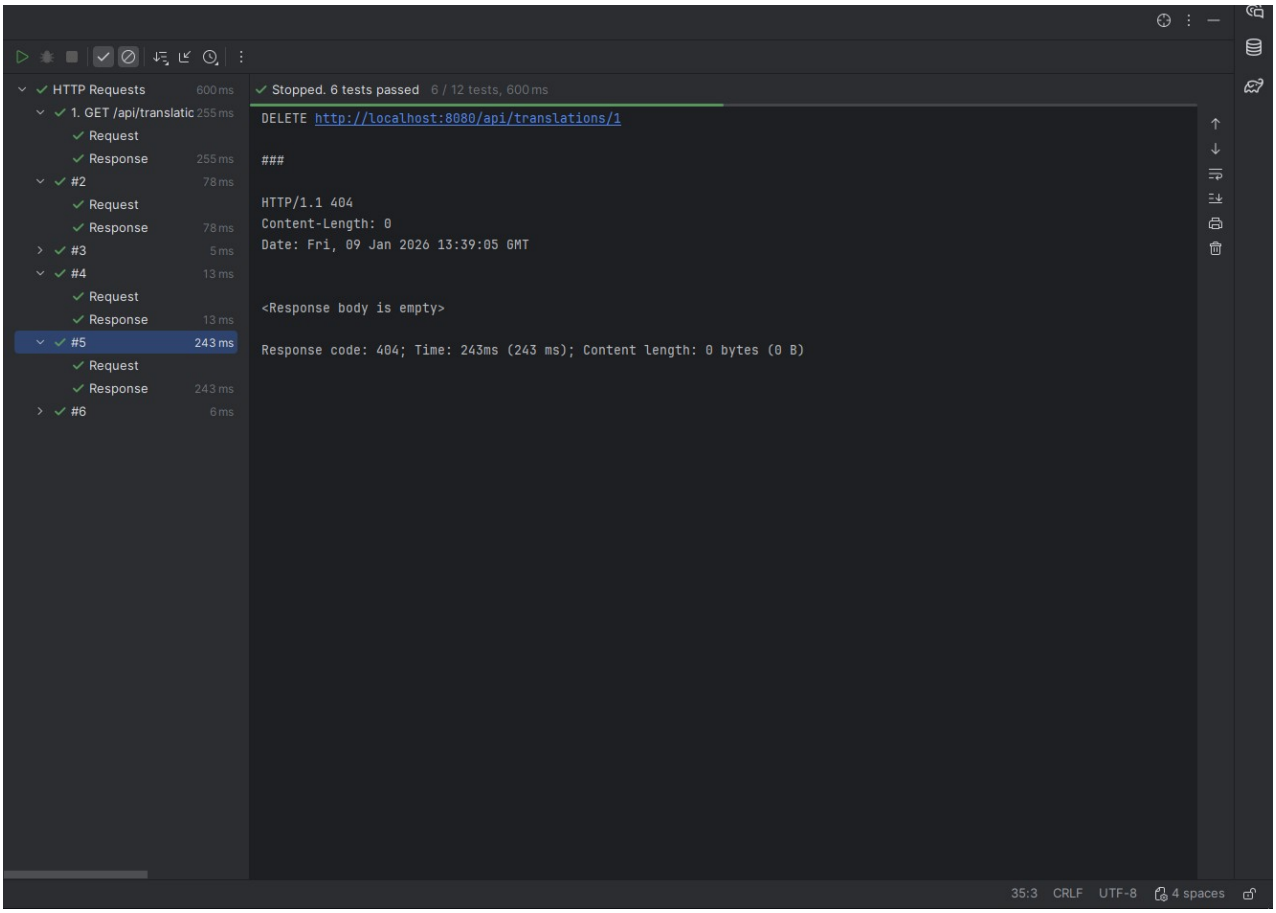


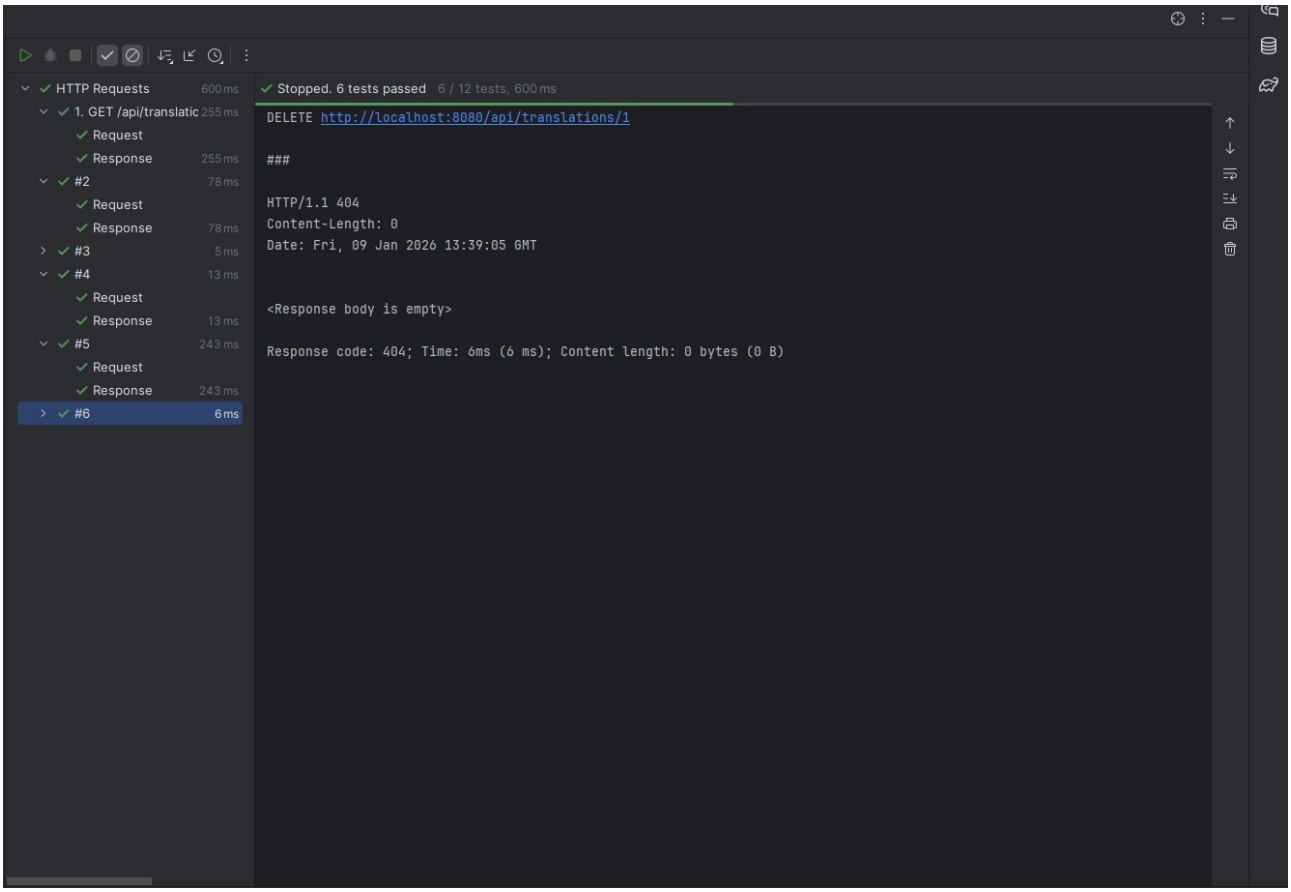

HTTP Requests 600 ms
1. GET /api/translatic 255 ms
Request
Response 255 ms
#2 78 ms
Request
Response 78 ms
#3 5 ms
#4 13 ms
#5 243 ms
Request
Response 243 ms
#6 6 ms
Stopped. 6 tests passed 6 / 12 tests, 600 ms
GET http://localhost:8080/api/translations
###
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Fri, 09 Jan 2026 13:39:05 GMT
Response file saved.
> 2026-01-09T163905.200.json
Response code: 200; Time: 255ms (255 ms); Content length: 471 bytes (471 B)
35:3 CRLF UTF-8 4 spaces


HTTP Requests 600 ms
1. GET /api/translatic 255 ms
Request
Response 255 ms
#2 78 ms
Request
Response 78 ms
#3 5 ms
#4 13 ms
#5 243 ms
Request
Response 243 ms
#6 6 ms
Stopped. 6 tests passed 6 / 12 tests, 600 ms
POST http://localhost:8080/api/translations
Content-Type: application/json
{
"sourceText": "Hello world",
"targetText": "Привет мир",
"connections": [
{ "description": "First connection" }
]
}
###
HTTP/1.1 201
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Fri, 09 Jan 2026 13:39:05 GMT
Response file saved.
> 2026-01-09T163905.201.json
Response code: 201; Time: 78ms (78 ms); Content length: 119 bytes (119 B)
35:3 CRLF UTF-8 4 spaces


HTTP Requests 600 ms
1. GET /api/translatic 255 ms
Request
Response 255 ms
#2 78 ms
Request
Response 78 ms
#3 5 ms
#4 13 ms
#5 243 ms
Request
Response 243 ms
#6 6 ms
Stopped. 6 tests passed 6 / 12 tests, 600 ms
POST http://localhost:8080/api/translations
Content-Type: application/json
{
"targetText": "Only target"
}
###
HTTP/1.1 400
Content-Length: 0
Date: Fri, 09 Jan 2026 13:39:05 GMT
Connection: close
<Response body is empty>
Response code: 400; Time: 5ms (5 ms); Content length: 0 bytes (0 B)


HTTP Requests 600 ms
1. GET /api/translatic 255 ms
Request
Response 255 ms
#2 78 ms
Request
Response 78 ms
#3 5 ms
#4 13 ms
Request
Response 13 ms
#5 243 ms
Request
Response 243 ms
#6 6 ms
Stopped. 6 tests passed 6 / 12 tests, 600 ms
GET http://localhost:8080/api/translations/2
###
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Fri, 09 Jan 2026 13:39:05 GMT
Response file saved.
> 2026-01-09T163905-1.200.json
Response code: 200; Time: 13ms (13 ms); Content length: 174 bytes (174 B)
35:3 CRLF UTF-8 4 spaces


HTTP Requests 600 ms
1. GET /api/translatic 255 ms
Request
Response 255 ms
#2 78 ms
Request
Response 78 ms
#3 5 ms
#4 13 ms
Request
Response 13 ms
#5 243 ms
Request
Response 243 ms
#6 6 ms
Stopped. 6 tests passed 6 / 12 tests, 600 ms
DELETE http://localhost:8080/api/translations/1
###
HTTP/1.1 404
Content-Length: 0
Date: Fri, 09 Jan 2026 13:39:05 GMT
<Response body is empty>
Response code: 404; Time: 243ms (243 ms); Content length: 0 bytes (0 B)
35:3 CRLF UTF-8 4 spaces


HTTP Requests 600 ms
1. GET /api/translatic 255 ms
Request
Response 255 ms
#2 78 ms
Request
Response 78 ms
#3 5 ms
#4 13 ms
Request
Response 13 ms
#5 243 ms
Request
Response 243 ms
#6 6 ms
Stopped. 6 tests passed 6 / 12 tests, 600 ms
DELETE http://localhost:8080/api/translations/1
###
HTTP/1.1 404
Content-Length: 0
Date: Fri, 09 Jan 2026 13:39:05 GMT
<Response body is empty>
Response code: 404; Time: 6ms (6 ms); Content length: 0 bytes (0 B)

# Тестирование с нагрузкой

Сервер был протестирован под симулированной нагрузкой с использованием *gatling*. Испытана нагрузка на сценариях, где в течение минуты или более отправляется 1 тысяча, 5 тысяч и 10 тысяч POST-запросов на */api/translations* в секунду.

- 1к POST-запросов:

```
================================================================================
================================================

---- Global Information --------------------------------------------------------|---Total---|-----OK----|----
KO----

> request count                                        |    60,000 |    60,000 |        -

> min response time (ms)                               |         0 |         0 |        -

> max response time (ms)                               |       267 |       267 |        -

> mean response time (ms)                              |         2 |         2 |        -

> response time std deviation (ms)                     |         9 |         9 |        -

> response time 50th percentile (ms)                   |         1 |         1 |        -

> response time 75th percentile (ms)                   |         1 |         1 |        -

> response time 95th percentile (ms)                   |         1 |         1 |        -

> response time 99th percentile (ms)                   |         2 |         2 |        -

> mean throughput (rps)                                |     1,000 |     1,000 |        -

---- Response Time Distribution ------------------------------------------------------------------------------------

> OK: t < 800 ms                                                  60,000   (100%)

> OK: 800 ms <= t < 1200 ms                                            0     (0%)

> OK: t >= 1200 ms                                                     0     (0%)

> KO                                                                   0     (0%)

================================================================================
================================================
```

- 5к POST-запросов:

```
================================================================================
================================================

---- Global Information --------------------------------------------------------|---Total---|-----OK----|----
KO----
```

> request count | 300,000 | 300,000 | -

> min response time (ms) | 0 | 0 | -

> max response time (ms) | 5,425 | 5,425 | -

> mean response time (ms) | 162 | 162 | -

> response time std deviation (ms) | 532 | 532 | -

> response time 50th percentile (ms) | 2 | 2 | -

> response time 75th percentile (ms) | 2 | 2 | -

> response time 95th percentile (ms) | 1,322 | 1,323 | -

> response time 99th percentile (ms) | 4,223 | 4,240 | -

> mean throughput (rps) | 4,918.03 | 4,918.03 | -

---- Response Time Distribution ----------

> OK: t < 800 ms 282,126 (94.04%)

> OK: 800 ms <= t < 1200 ms 1,275 (0.43%)

> OK: t >= 1200 ms 16,599 (5.53%)

> KO 0 (0%)

==========

- 10к POST-запросов:

==========

---- Global Information ----------|---Total---|-----OK----|----KO----

> request count | 600,000 | 268,316 | 331,684

> min response time (ms) | 3 | 301 | 3

> max response time (ms) | 91,182 | 82,066 | 91,182

> mean response time (ms) | 12,042 | 13,366 | 10,971

> response time std deviation (ms) | 11,173 | 8,861 | 12,639

> response time 50th percentile (ms) | 9,463 | 11,284 | 7,711

> response time 75th percentile (ms) | 16,419 | 18,420 | 14,489

> response time 95th percentile (ms) | 29,317 | 29,823 | 28,248

> response time 99th percentile (ms) | 62,624 | 39,025 | 71,010

> mean throughput (rps) | 4,477.61 | 2,002.36 | 2,475.25

---- Response Time Distribution ----------

> OK: t < 800 ms 1,974 (0.33%)

> OK: 800 ms <= t < 1200 ms 2,326 (0.39%)

> OK: t >= 1200 ms 264,016 (44%)

> KO 331,684 (55.28%)

---- Errors ----------

> j.n.ConnectException: connect(..) failed: Cannot assign requested address 305,715 (92.17%)

> i.n.c.ConnectTimeoutException: connection timed out after 10000 ms: /127.0.0.1:8080 16,015 (4.83%)

> Request timeout to 127.0.0.1/127.0.0.1:8080 after 60000 ms 9,943 (3%)

> j.i.IOException: Premature close 11 (0%)

==========

Результаты показывают, что нагрузку в 5 тысяч запросов/с и ниже сервер выдерживает без особых трудностей, подавляющее большинство таких запросов получают ответ меньше, чем за секунду. Когда нагрузка достигает 10 тысяч запросов/с - больше половины запросов не получают ответ (92% из них отбрасывает операционная система из-за длинной очереди), а средняя скорость обработки увеличивается до 12 секунд. Учитывая, что даже 1 тысяча запросов в секунду это нереалистично высокая мерка для потенциального пользования данной программы - за её масштабируемость беспокоиться нет необходимости.

# Сборка и запуск

Для сборки клиента используется утилита *cmake*:
```
cmake -S Woofie -B build
cmake --build build
```

Для упрощения сборки вне контейнера (в частности для подгрузки библиотек и среды разработки QtCreator) можно применить менеджер пакетов и утилиты *nix:*
```
nix develop
```

Предусмотрена сборка через *docker:*
```
docker build -t buildimage
```

В отличие от серверной стороны, ввиду того, что клиент является GUI-приложением, его запуск изнутри контейнера затруднён, если не невозможен.

Для ручного тестирования, сборки и запуска сервера используется *gradle:*
```
gradle test
```

```
gradle build
gradle bootRun
```

Для ручного запуска (пустой) базы данных PostgreSQL так же применяется *nix*:

```
nix run
```

Базу данных и сервер можно собрать и запустить используя *docker-compose*

```
docker-compose up
```