

Использование бота StGptBot для генерации тестовых кейсов

Как найти бота

Чтобы найти бота в поиске в телеграмм необходимо ввести его username - @CTT_gpt_bot

Функции бота

- Как найти бота
- Функции бота
 - /start
 - /help
 - /clearSession
 - /getcasejira
 - /getcasewiki
 - /getcaseapi
 - /createPageObject
 - /sendToTestit
 - /createBugReport

/start

- старт работы с ботом

/help

- вывод списка функций бота

/clearSession

- удаление сессии (токены в Jira и Wiki самостоятельно, например, когда ввели ошибочный токен и система его приняла)




/getcasejira

- генерация тестового кейса в Jira по переданному идентификатору сущности (**берется описание задачи и по нему генерируется тестовые кейсы**)

После вызова данной функции необходимо сделать следующее:

1) Выбрать инструмент для генерации тестового кейса (модель gpt):

- YandexGPT (работает лучше всего)
- ChatGPT 3.5 Turbo (работает через проксирование, дольше чем YaGPT, но неплохо генерирует кейсы)
- SberGigachat (странно генерирует кейсы, но надо тестировать)

 Write a message...  

YaGPT

ChatGPT 3.5 Turbo

Sber Gigachat

2) Необходимо передать Jira API Token, чтобы вычитать описание задачи, в формате JiraToken {MyToken}. Пример: JiraToken abcd123. Как получить токен можно почитать тут - [Получение api token в Jira/Wiki/TestIT и работа со swagger для генерации кейсов](#)

Токен живет в системе до следующего календарного дня, поэтому при повторных вызовах метода передавать токен будет не нужно.

3) Передать идентификатор задачи в Jira в формате ERS-XXXX

Как кейсы будут сгенерированы, придет сообщение с готовыми кейсами:

Конечно, вот несколько функциональных тестовых кейсов для требований "Необходимо проработать макеты для редизайна страницы 'Бизнесу'":

1. ****Проверка внешнего вида и оформления страницы:****

- Загрузить страницу "Бизнесу".
- Убедиться, что внешний вид соответствует утвержденным макетам для редизайна.
- Проверить цветовую гамму, шрифты, стилизацию элементов и их расположение.

2. ****Проверка навигации:****

- Проверить работу основного меню.
- Убедиться, что все разделы страницы доступны из главного меню.
- Проверить корректность переходов между разделами и подразделами.

3. ****Проверка контента:****

- Проверить наличие основной информации о продуктах или услугах для бизнеса.
- Убедиться, что контент соответствует требованиям и предоставленным макетам.
- Проверить наличие и корректность ссылок на дополнительные ресурсы или материалы.

Дальше будет предложены следующие варианты:

- Сгенерировать еще дополнительные кейсы (доступно только при выборе yaGPT), позволяет получить еще больше кейсов по переданной ранее информации (/getMoreCases). Необходимо выполнять до отправки результата (/sendToJira), после генерации дополнительных кейсов будет возможность отправить общий набор(его можно вызывать только сразу после получения сгенерированных кейсов, в другой момент вызов данного метода не доступен)
- Опубликовать текущие сгенерированные кейсы в задачу Jira в комментарии через метод /sendToJira (его можно вызывать только сразу после получения сгенерированных кейсов/дополнительно сгенерированных кейсов, в другой момент вызов данного метода не доступен)
- Отправить сгенерированные кейсы в комментарий к другой задаче (например, если необходима отправка кейсов в отдельную задачу на тестирование). Для этого необходимо передать номер тикета, куда отправить кейсы в формате send ERS-XXXX, после чего выполнить /sendToJira
- Отправить сгенерированные кейсы в систему TestIT /sendToTestit

* Если хочешь сгенерировать дополнительные кейсы - выполни **/getMoreCases** Необходимо выполнять до отправки результата!

* Если хочешь опубликовать текущие кейсы комментарием в тикет в Jira - выполни **/sendToJira**

* Если хочешь отправить текущие результаты генерации в другой тикет - передай номер тикета, куда отправить кейсы в формате **send ERS-XXXX**, после чего выполни **/sendToJira**

* Если хочешь отправить текущие результаты генерации TestIT выполни **/sendToTestit**

13:44

После успешной публикации кейсов в комментарии придет отбивка, что кейсы опубликованы, а также в задачу проставится метка **ctt_gpt**.

/getcasewiki




- генерация тестового кейса по сценарию на вики по переданному id страницы. (для генерации тестовых кейсов подходят страницы со сценариями от аналитики, например, [2. Получение кнопок-ссылок для блока новостроек в карточке застройщика/ГК](#), страницы без сценариев либо не по формату - не обрабатываются)

Описание поддерживаемых форматов страниц - [Требования к шаблону оформления аналитики для генерации тестовых кейсов по ним через STGptBot](#)

После вызова данной функции необходимо сделать следующее:

1) Выбрать инструмент для генерации тестового кейса (модель gpt):

- YandexGPT (работает лучше всего)
- ChatGPT 3.5 Turbo (в функциональности генерации через сценарии - зависает, лучше не использовать)
- SberGigachat (странно генерирует кейсы, но надо тестировать)

 Write a message...  

YaGPT

ChatGPT 3.5 Turbo

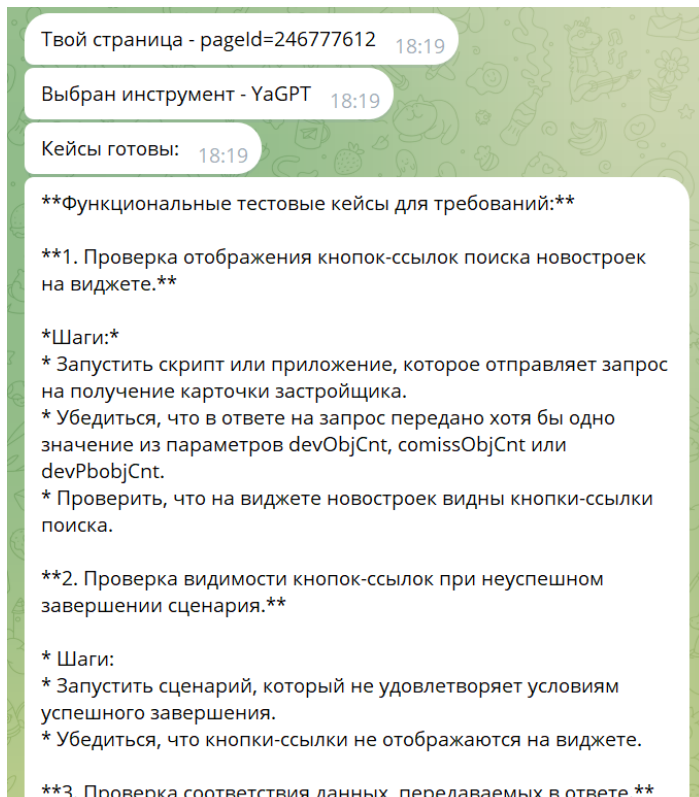
Sber Gigachat

2) Необходимо передать Wiki API Token, чтобы вычитать сценарий, в формате WikiToken {MyToken}. Пример: WikiToken abcd123. Как получить токен можно почитать тут - [Получение api token в Jira/Wiki/TestIT и работа со swagger для генерации кейсов](#)

Токен живет в системе до следующего календарного дня, поэтому при повторных вызовах метода передавать токен будет не нужно.

3) Передать id страницы в Wiki в формате pageid={id}, пример - pageid=246777612 (вытаскиваем из query params URL страницы на вики)

Как кейсы будут сгенерированы, придет сообщение с готовыми кейсами:



Дальше будут предложены следующие варианты:

- Сгенерировать еще дополнительные кейсы (доступно только при выборе yaGPT), позволяет получить еще больше кейсов по переданной ранее информации (/getMoreCases). Необходимо выполнять до отправки результата (/sendToWiki), после генерации дополнительных кейсов будет возможность отправить общий набор (его можно вызывать только сразу после получения сгенерированных кейсов, в другой момент вызов данного метода не доступен)
- Опубликовать текущие сгенерированные кейсы в статью на Wiki в комментарии через метод /sendToWiki (его можно вызывать только сразу после получения сгенерированных кейсов/дополнительно сгенерированных кейсов, в другой момент вызов данного метода не доступен)
- Отправить сгенерированные кейсы в систему TestIT /sendToTestit

* Если хочешь сгенерировать дополнительные кейсы - выполни `/getMoreCases` Необходимо выполнять до отправки результата!

* Если хочешь опубликовать текущие кейсы комментарием в статью на Wiki выполни `/sendToWiki`

21:59

После успешной публикации кейсов в комментарии придет отбивка, что кейсы опубликованы, а также в задачу проставится метка `ctt_gpt`.

`/getcaseapi`

- генерация тестового кейса для API по swagger и нужному методу

1) Выбери инструмент для генерации тестовых кейсов (выбор через кнопки ниже) 23:27




2) Выбери что ты хочешь сделать:
Генерация тестовых кейсов для проверки API:
`/gettextcase`
Генерация API тестов в формате CURL:
`/getcurltests` 23:27

3) Передай ссылку на api-doc в формате Apidoc {link}. Пример: Apidoc <https://sma-expertise.apps.k8s.uat.domoy.ru/api-docs>. Как найти ссылку на api-doc можно почитать тут - <https://wiki.domrf.ru/pages/viewpage.action?pageId=279049742> 23:27

4) Напиши метод из swagger в формате Method {yourmethod}, пример - Method /api/v1/expertise/type/{full-statement-id} 23:27

После вызова данной функции необходимо сделать следующее:

- 1) Выбрать инструмент для генерации тестового кейса (модель gpt):
 - YandexGPT (работает лучше всего)
 - ChatGPT 3.5 Turbo (**Возможно зависает**)
 - SberGigachat (странно генерирует кейсы, но надо тестировать)

 Write a message...  

YaGPT

ChatGPT 3.5 Turbo

Sber Gigachat

- 2) Выбрать одну из двух функций:

`/gettextcase` - сгенерирует текстовое описание кейсов для проверки API метода

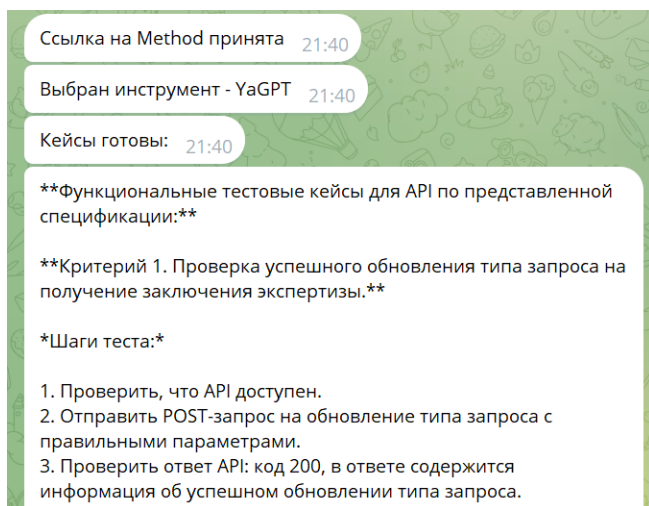
`/getcurltests` - сгенерирует API тесты в формате CURL (которые можно в 2 клика импортировать в Postman), описание как это сделать [Как импортировать CURL запрос в Postman \(из бота CTGptBot\)](#)

3) Необходимо передать ссылку на api-doc в формате Apidoc {link}. Пример: Apidoc <https://sma-expertise.apps.k8s.uat.domoy.ru/api-docs>. Как найти ссылку на api-doc можно почитать тут - <https://wiki.domrf.ru/pages/viewpage.action?pageId=279049742>

Сетевая связность у бота только до адресов UAT контура, поэтому работать ссылки на Apidoc будут только в домене uat.domoy.ru / [fap](#)

4) Передать метод из swagger, для которого необходимо сгенерировать тестовые кейсы в формате Method {yourmethod}, пример - Method /api/v1/expertise/type/{full-statement-id} (**копируем из swagger в точности как там**)

Как кейсы будут сгенерированы, придет сообщение с готовыми кейсами:



Также в случае, если использовалась модель yaGPT, можно вызвать метод `/getMoreCases`, чтобы получить больше кейсов по переданной информации

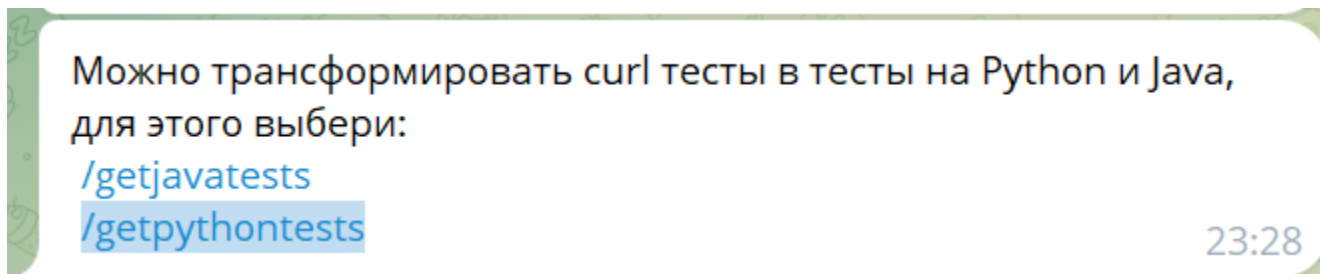
Кейсы сгенерированные кейсы можно отправить в какую либо задачу Jira (выполнив команду `send ERS-XXXX`(номер тикета, куда отправить) и выполнив команду `/sendToJira`

Кейсы можно отправить в TestIT - выполнив команду `/sendToTestit`

В случае, если была выбрана функция `/getcurtests` -придет сообщение с возможностью выбора дополнительной функции:

`/getjavatests` - генерация API тестов на Java + RestAssured (инструкция, как пользоваться сгенерированным кодом будет предоставлена позже)

`/getpythontests`- генерация API тестов на Python + Requests (инструкция, как пользоваться сгенерированным кодом будет предоставлена позже)



В результате придет код API тестов на Python:

Вот как можно преобразовать указанные запросы в API-тесты на Python с использованием библиотеки Requests:

1. **Запрос с корректным идентификатором и данными:**

```
python
import requests

url = "https://sma-expertise.apps.k8s.uat.domoy.ru/api/v1/expertise/files-category/"

payload = {
    "files": [
        {
            "file_name": "example.pdf",
            "category": "Документация"
        }
    ],
    "signature_details": {
        "signed_by": "Иванов Иван Иванович",
        "date": "2023-01-01"
    }
}

headers = {"Content-Type": "application/json"}

response = requests.post(url, json=payload, headers=headers)
assert response.status_code == 200
```

/createPageObject

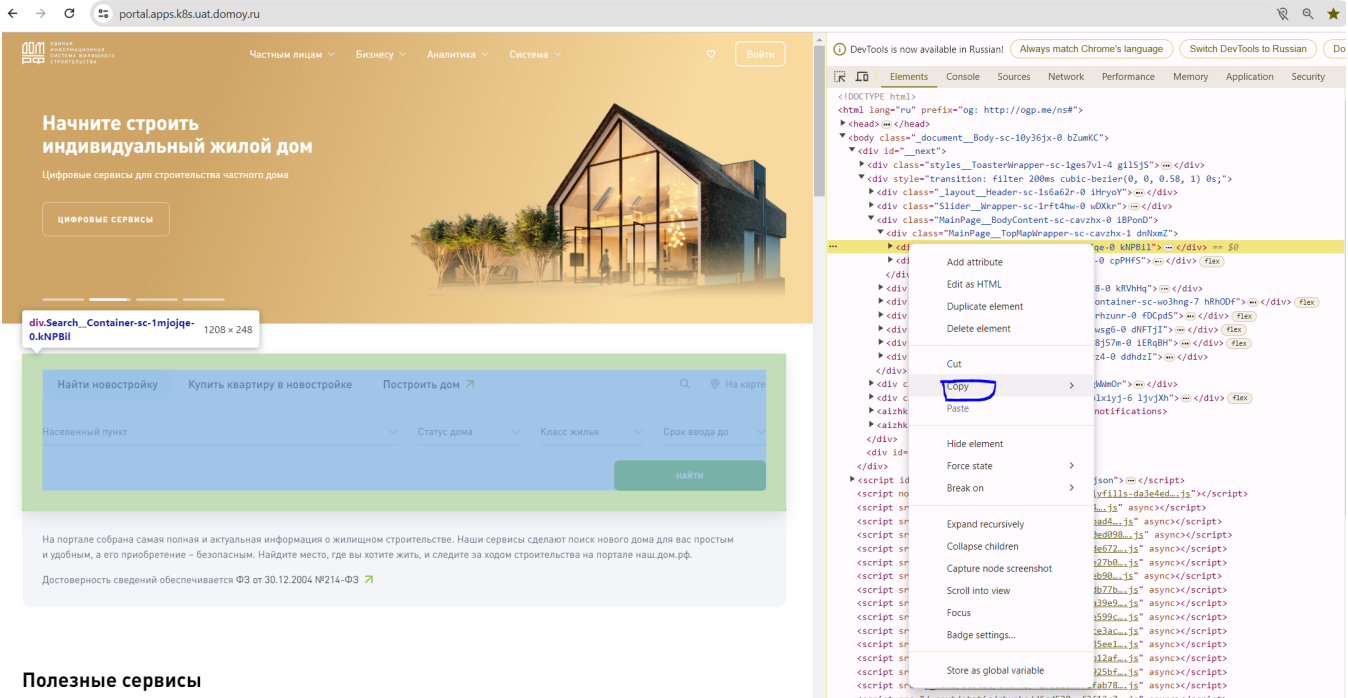
- генерация описания объектов(xpath - локаторов) html-фрагмента страницы для АТ по переданным данным (через wiki)

Что нужно:

- 1) Создать страницу на wiki, чтобы в название статьи входило сочетание - "Generate", либо взять существующую тестовую статью - [Generate](#).
[Тестовый пример](#)
- 2) Добавить html фрагмента, который мы хотим покрыть локаторами, например:

The screenshot displays a web browser window with the URL portal.apps.k8s.uat.domoy.ru. The page is titled "Инвестиционный калькулятор" (Investment Calculator) and includes a search bar with filters for "Населенный пункт" (Settlement), "Статус дома" (House status), "Класс жилья" (Housing class), and "Срок ввода в эксплуатацию" (Start of operation). A blue box highlights the search bar area. The DevTools console shows the HTML structure of the page, including various divs and script tags. A blue box highlights the search bar area on the page.

Далее копируем html данного блока:



Выбираем сорюOuterHtml (в буфер обмена у нас попадает нужная html)

Далее вставляем ее в нужную статью в тело:



Желательно не отправлять слишком большой объем данных, бот не сможет его обработать и будет ошибка, в случае чего рекомендую разбить на несколько более маленьких объекта и пробовать в несколько итераций

3) Вызываем /createPageObject

1) Вызываем инструмент "yagpt"

2) Передаем Wiki API Token, чтобы вычитать сценарий, в формате WikiToken {MyToken}. Пример: WikiToken abcd123. Как получить token можно почитать тут - [Получение api token в Jira/Wiki/TestIT и работа со swagger для генерации кейсов](#)

Токен живет в системе до следующего календарного дня, поэтому при повторных вызовах метода передавать токен будет не нужно.

3) Передаем prompt в формате Prompt {prompt} (на период отладки данного метода и подбора лучшего варианта prompt для данной задачи). На текущий момент рекомендуется использовать prompt вида: "Prompt Составь локаторы для полей: фео, телефон, емзил, ИНН, должность по xpath no html" с перечислением названий полей, по которым необходимо составить локаторы.

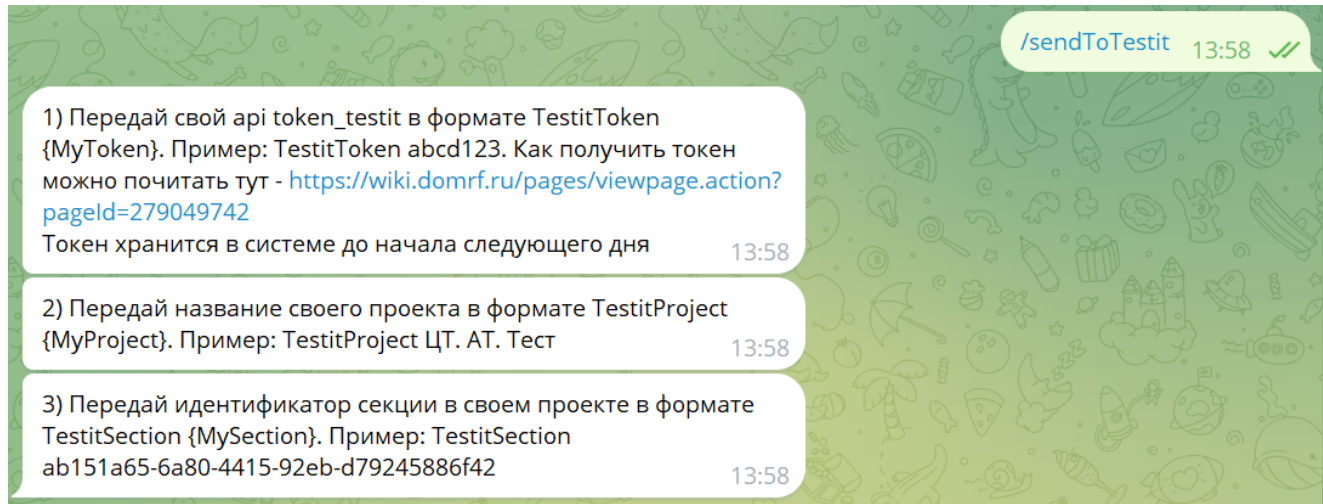
3) Передать id страницы в Wiki в формате pageid={id}, пример - pageid=246777612 (выгаскиваем из query params URL страницы на вики)

4) Описание xpath готово

5) Далее будет предложено опубликовать сгенерированные xpath в статью Wiki в комментарии через метод /sendToWiki (его можно вызывать только сразу после получения результатов, в другой момент вызов данного метода не доступен):

/sendToTestit

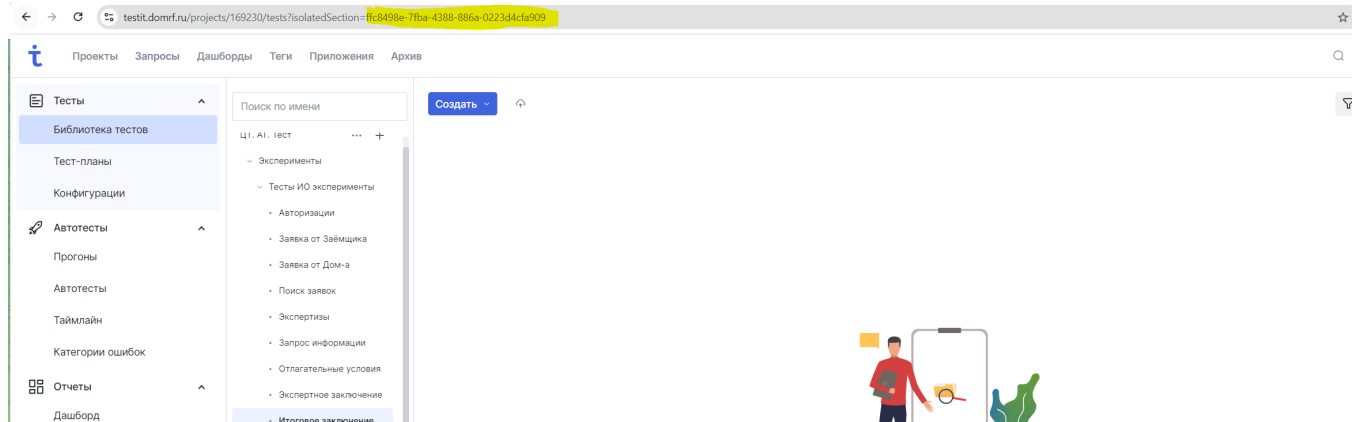
Метод позволяет отправить сгенерированные тестовые кейсы в TestIT и доступен только после успешной генерации кейсов



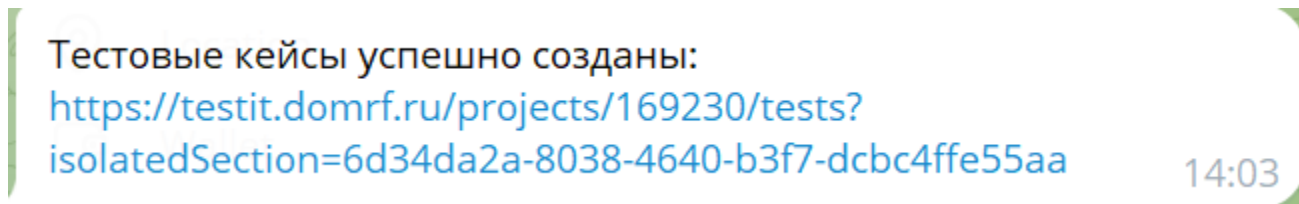
1) Необходимо передать токен по формату **TestitToken{token}**

2) Необходимо передать название своего проекта, куда будут отправлены кейсы по формату **TestitProject{Название проекта}**

3) Передать идентификатор корневой секции, в которую будет добавлена секция с новыми сгенерированными кейсами. Ее можно скопировать из url в адресной строке



После успешной отправки будет выведено сообщение со ссылкой на созданную секцию со сгенерированными кейсами



/createBugReport

Метод позволяет создать баг-репорт в Jira из ассистента, например, для удобной доставки информации в Jira, например, с мобильного устройства при тестировании на девайсах, либо после выполнения сгенерированного ботом теста

1) Вызываем метод /createBugReport

2) Передаем последовательно необходимые поля для заведения бага

