РУКОВОДСТВО ПО ЭКСПЛУАТАЦИИ СИСТЕМЫ «Сервис генерации естественного языка»

Версия 1.0 • 01/06/2024

СОДЕРЖАНИЕ

> В каждую модель, предоставляемую нашим клиентам и партнерам, мы встраиваем уникальный водяной знак. Это необходимо для установления факта утечки модели от клиента или партнера. Просим вас ответственно относиться к обеспечению безопасности хранения модели.

1 — О ПРОДУКТЕ

«Сервис генерации естественного языка» от МТС AI — это передовое решение, основанное на технологиях генеративного искусственного интеллекта (Generative AI), предназначенное для автоматизации и оптимизации обработки неструктурированных данных. Это инновационное решение позволяет бизнесам и организациям эффективно решать задачи, связанные с распознаванием, анализом и преобразованием текстовой информации в ответ на запросы пользователей (промпты).

ФУНКЦИОНАЛ ПРОДУКТА:

- генерация текста: автоматизированное создание уникального контента, статей, отчетов, описаний продуктов и многое другое с высоким уровнем естественности и релевантности;
- анализ текста: извлечение смысла, классификация текстов по темам, определение тональности, распознавание именованных сущностей и анализ эмоциональной окраски;
- автоматический перевод: перевод текста с одного языка на другой с поддержкой множества языков;
- семантический поиск: поиск по смыслу в больших объемах текста, позволяющий находить наиболее релевантную информацию по запросам пользователей;
- поддержка диалогов: разработка и внедрение чат-ботов, виртуальных ассистентов, способных вести беседу, отвечать на вопросы и выполнять команды на естественном языке;
- обработка и анализ больших данных: анализ больших объемов текстовой информации для выявления трендов, паттернов и получения ценных бизнес-инсайтов.

2 — ОСОБЕННОСТИ И ПРЕИМУЩЕСТВА

Основные преимущества Сервиса генерации естественного языка от МТС AI включают:

- **Безопасность данных:**

Работа и поддержка в контуре клиента обеспечивают высокий уровень защиты данных и результатов обучения.

- **Полное управление:**

Возможность обучения на собственных данных клиентов, что позволяет полностью контролировать источники информации и сценарии использования.

- **Высокую производительность:**

LLM обрабатывают и понимают естественный язык лучше любой другой технологии, существовавшей до сих пор, что обеспечивает высокую точность и эффективность решений.

3 — ОСНОВНЫЕ ТЕРМИНЫ В ДОКУМЕНТЕ

Термин	Определение
Искусственный интеллект (ИИ)	Это область компьютерных наук, занимающаяся созданием машин, способных выполнять задачи, требующие человеческого интеллекта, например, восприятие, рассуждение, обучение и решение проблем.
Машинное обучение	Отрасль искусственного интеллекта, в которой машины обучаются выполнять задачи, анализируя и обобщая данные, а не благодаря прямому программированию.
Генеративные модели	Это типы искусственного интеллекта, которые могут создавать новый контент, например тексты, изображения и музыку. Они учатся на больших объемах данных и затем генерируют новый контент, имитируя то, что они видели.
Дообучение (Fine-tuning)	Процесс настройки предобученной модели под конкретную задачу путем дополнительного обучения на более мелком или специфическом наборе данных.
Нейронная сеть	Компьютерная модель, вдохновленная биологическими нейронами, используемая для распознавания паттернов и обработки информации в задачах машинного обучения.
Токен	Минимальная единица текста (например, слово или символ), используемая в обработке естественного языка для анализа и генерации текста.
Натуральный язык (NLP)	Область ИИ, занимающаяся взаимодействием между компьютерами и людьми на естественных языках, таких как русский или английский.
API	Набор правил и инструментов для взаимодействия программного обеспечения. API позволяет различным приложениям обмениваться данными и функциональностью. (Application Programming Interface)
Docker	Платформа для автоматизации развёртывания приложений в контейнерах, что обеспечивает их изоляцию и независимость от среды выполнения.
JSON	Лёгкий формат обмена данными, легко читаемый человеком и парсируемый компьютером. (JavaScript Object Notation)

4 — ПРОГРАММНЫЕ И АППАРАТНЫЕ ТРЕБОВАНИЯ

4.1 ТРЕБОВАНИЯ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ РАБОЧЕЙ СТАНЦИИ

Для работы системы необходимо, чтобы выполнялись следующие требования к программному обеспечению:

Ресурс	Требования
Операционная система	Linux: Rocky Linux
Рекомендованная ОС	Ubuntu 24.04 LTS (Noble Numbat) › Ubuntu 22.04.4 LTS (Jammy Jellyfish) › Ubuntu 20.04.6 LTS (Focal Fossa)
Docker	Docker version 24.0.4
Nvidia-Docker	Docker version 24.0.4, build 3713ee1
Интернет	Наличие доступа к Интернет для контейнеров и дополнительных загрузок ПО

4.2 ТРЕБОВАНИЯ К АППАРАТНОМУ ОБЕСПЕЧЕНИЮ РАБОЧЕЙ СТАНЦИИ

Для работы системы необходимо, чтобы выполнялись следующие требования к аппаратным ресурсам:

Ресурс	Требования
CPU	от 8 до 32 шт
RAM	от 32 GB
GPU	1x NVIDIA A100 с 80 GB видеопамяти
SSD	500 GB
Видеодрайвер	Driver Version: 525.105.17, CUDA Version: 12.0

5 — РАЗВЕРТЫВАНИЕ СИСТЕМЫ

> Для развертывания системы в собственной инфраструктуре необходимо выполнить пункты из данного раздела.

1. Скачивание файлов

Скачайте файл docker-compose и env-файл. Поместите их в желаемую директорию.

2. Проверка и конфигурация env-файла

Ознакомьтесь с содержимым env-файла, который передан вместе с docker-compose файлом. Этот файл содержит значения переменных окружения.
При необходимости, сконфигурируйте переменные.

3. Авторизация в Artifactory

Залогиньтесь в Artifactory с использованием вашей учетной записи:

```
docker login artifactory.mts.ai
```

4. Запуск контейнера

Запустите контейнер с помощью команды:

```
docker compose up -d
```

> Убедитесь, что у вас установлен Docker Compose.

5. Проверка работы системы

Проверьте, что система работает корректно, сделав запрос к хосту. Предполагается, что роут открыт на 8000 порту на машине, где были выполнены предыдущие шаги.

```
curl $HOST/v1/chat/completions -H "Content-Type: application/json" -d '{"model": "<MODEL_NAME>", "messages": [{"role": "user", "content": "Когда родился Пушкин?"}], "temperature": 1.0}'
```

5.1 — ПРОВЕРКА И КОНФИГУРАЦИЯ ENV-ФАЙЛА

> Вместе с docker-compose файлом вам будет передан env-файл, который содержит значения переменных окружения. Вам необходимо ознакомиться с этими значениями и при необходимости сконфигурировать их. В файле будут переменные, указанные ниже.

1. Переменные потребления видеопамяти

Переменные потребления видеопамяти (VRAM) используются для мониторинга и управления использованием графической памяти (видеопамяти) на GPU. Эти переменные могут включать текущий объем используемой VRAM, максимальный объем VRAM, доступный для использования, и другие метрики производительности GPU.

Таблица 1.Переменные потребления видеопамяти

Переменная	Определение
EAGER	Параметр, активирующий флаг --enforce_eager для vLLM (true - активирует флаг - enforce_eager, false - не активирует флаг --enforce_eager). Если установлен "True", то CUDA не будет задействован, при false наоборот.
GPUUTIL	Параметр, отвечающий за утилизацию GPU. Соответствует значению, передаваемому в параметр --gpu_memory_utilization. Доля GPU может находиться в диапазоне от 0 до 1. Например, значение 0,5 будет означать использование памяти графического процессора на 50%.

2. Переменные авторизации

Переменные авторизации используются для хранения и управления данными, необходимыми для аутентификации и авторизации пользователей в системе. Эти переменные могут включать токены доступа, сессионные идентификаторы, роли пользователей и другие данные, связанные с безопасностью.

Таблица 2. Переменные авторизации

Переменная	Определение
MTSAI_AUTH	Параметр, включающий пользовательскую авторизации. True - активирует пользовательскую авторизацию, false - активирует однотокеновую или dummy авторизацию, какая именно будет зависеть от значения переменной VLLM_AUTH_TOKEN.
VLLM_AUTH_TOKEN	Токен авторизации. Строка произвольной длины, переданная в качестве значения этого параметра, будет являться токеном. Если передать "dummy" - активируется dummy-авторизация.
DB_ID	ID таблицы в Postgres БД, к которой мы хотим подключиться. Необходим при пользовательской авторизации.

> Однотокеновая авторизация (one-token auth) - это авторизация, при которой существует только 1 токен, способный пройти авторизацию (его значение необходимо задать в рамках конфигурации env).
>
> Dummy-авторизация (dummy auth) - это авторизация, при которой любой токен способен пройти авторизацию.
>
> Рекомендуем установить на “false” и использовать однотокеновую авторизацию.

3. Переменные базы данных

Переменные базы данных используются для хранения конфигурационных данных, необходимых для подключения и работы с базой данных. Эти переменные могут включать URL базы данных, имя пользователя, пароль, параметры подключения и другие настройки.

Таблица 3. Переменные базы данных

Переменная	Определение
POSTGRES_ENV_HOST	Хост, на котором развернута Postgres БД "users" пользовательской авторизации. Не используется при ином типе авторизации.
POSTGRES_ENV_PORT	Порт хоста, на котором развернута Postgres БД "users" пользовательской авторизации. Не используется при ином типе авторизации.
PG_PWD	Пароль от базы данных "users" пользовательской авторизации. Не используется при ином типе авторизации.

4. Пример заполнения файла

4.1. Однотокеновая авторизация:

- EAGER: false
- GPUUTIL: 0.9
- MTSAI_AUTH: false
- VLLM_AUTH_TOKEN: <токен авторизации>
- DB_ID: <неважно>
- POSTGRES_ENV_HOST: <неважно>
- POSTGRES_ENV_PORT: <неважно>
- PG_PWD: <неважно>

4.2. Dummy авторизация:

- EAGER: false
- GPUUTIL: 0.9
- MTSAI_AUTH: false
- VLLM_AUTH_TOKEN: dummy
- DB_ID: <неважно>
- POSTGRES_ENV_HOST: <неважно>
- POSTGRES_ENV_PORT: <неважно>
- PG_PWD: <неважно>

> При необходимости скорректируйте значения переменных в env-файле.
>
> После внесения изменений, поместите env-файл в ту же директорию, где находится docker-compose.yml.

6 — ДОСТУП К API

> После успешного развертывания, либо установки в облаке, API модели будет доступно по адресу /v1 на вашей машине.
>
> Например: http://1.1.1.1:8000/v1.

Чтобы подключить API:

Шаг 1. Получите авторизационные данные для вашего приложения.
Данные в виде имени модели и токена будут переданы вместе с образом.
Шаг 2. Добавьте сценарий получения Access Token в сервис для вызова API.
Шаг 3. В запросах замените MTSAI_API_KEY на полученный токен и укажите ваше название модели.

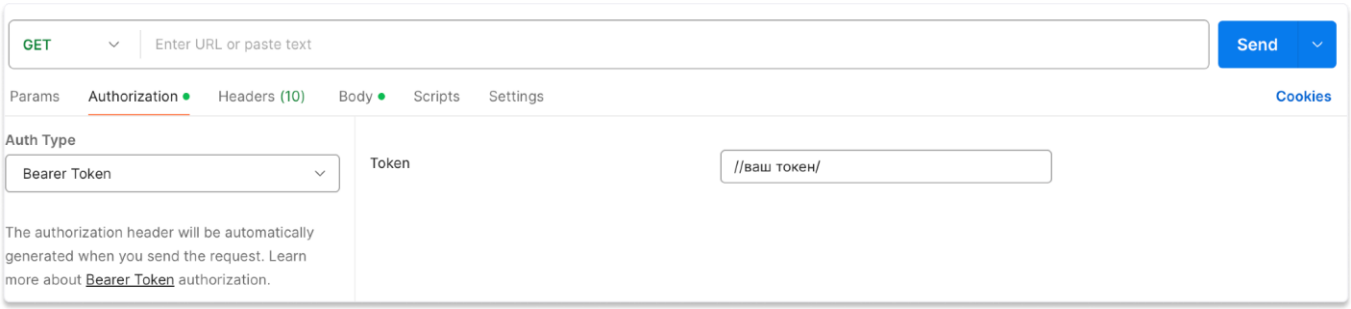

Рисунок 1. Пример заполнения данных в Postman

7 — СПРАВОЧНИК ПО API

> Этот справочник по API предоставляет базовую информацию для работы с продуктом и сообщениями в системе.

COMPLETIONS

Метод предназначен для обработки списка сообщений и генерации ответа на основе входных данных. Он принимает список сообщений в формате чата и возвращает сгенерированное моделью сообщение. Метод может использоваться как для ведения диалогов, состоящих из нескольких последовательных сообщений, так и для выполнения задач, требующих однократного обращения без продолжительного разговора.

Имя	Тип данных	Значение	Описание
messages	string	Текстовый ответ	Метод передачи списка сообщений модели включает в себя объекты с ролями "system", "user" и "assistant", что позволяет учитывать контекст диалога и обеспечивает более точные ответы. Системное сообщение задает поведение ассистента, а передача истории диалога важна для понимания контекста и корректного выполнения запросов пользователя.
model	string	ID	ID модели, к которой вы обращаетесь.

temperature	float	Диапазон температуры — от 0 до 2 > Рекомендованное значение: 0.5	Более низкие значения температуры приводят к более последовательным результатам (например, 0.2), в то время как более высокие значения генерируют более разнообразные и творческие результаты (например, 1.0).
top_p	int	Диапазон — от 0 до 1	Чем ниже значение атрибута, тем более популярные и часто встречающиеся токены модель будет использовать для формирования ответа. > Рекомендуем изменять или этот атрибут, или temperature, но не оба сразу.
max_tokens	int	Натуральное число больше 0	Максимальное количество токенов, которые могут быть сгенерированы в ответ на запрос пользователя. Это позволяет регулировать длину ответа.
n	int	Натуральное число больше 0 > По умолчанию: 1	Количество ответов, которые модель сгенерирует в ответ на ваш запрос.
stream	bool	True/False	Если установить значение true, ответ модели будет возвращаться не целиком сразу, а итеративно, по мере его формирования моделью.

frequency_penalty	int	Натуральное число от -2 до 2	Штраф за частоту — число между -2.0 и 2.0. Положительные значения штрафуют новые токены на основе их текущей частоты в тексте, снижая вероятность того, что модель повторит одну и ту же строку.
presence_penalty	int	Натуральное число от -2 до 2	Положительные значения накладывают штраф на новые токены в зависимости от того, появляются ли они в тексте до сих пор, увеличивая вероятность того, что модель будет говорить о новых темах.

MODELS

Метод для получения списка доступных вам моделей.

HEALTH

Запрос проверяет работоспособность модели. Если модель работоспособна, будет получен ответ [200 OK]. В случае если сервис недоступен по какой-либо причине, ответ не будет получен.

8 — ЭКСПЛУАТАЦИЯ СЕРВИСА ЧЕРЕЗ API ЗАПРОСЫ

Заказчик последовательно отправляет HTTP-запросы к сервисам Cotype через сервис API и получает в ответ пакет данных. Для некоторых запросов предварительно требуется выполнить запрос, чтобы получить сущность, идентификатор которой будет использоваться в следующем запросе.

В данном документе приведена следующая последовательность действий:

1. Проверка работоспособности модели
2. Получение ID модели
3. Отправка модели сообщения и получение ответа от модели

8.1 — ПРОВЕРКА РАБОТОСПОСОБНОСТИ МОДЕЛИ

8.1.1. Описание запроса

Назначение: Запрос проверяет работоспособность модели. Если модель работоспособна, будет получен ответ [200 OK]. В случае если сервис недоступен по какой-либо причине, ответ не будет получен.

Тип запроса: GET

Запрос:

```
http://IP_Addres/health
```

- Где **IP_Address** необходимо заменить на IP-адрес вашей машины. Например:

```
http://1.1.1.1:8000/health
```

8.1.2. Выполнение запроса

Заполните раздел Headers по данным из таблицы ниже.

Key	Value
Content-Type	application/json

Пример заполнения в Postman представлен на рисунке ниже.

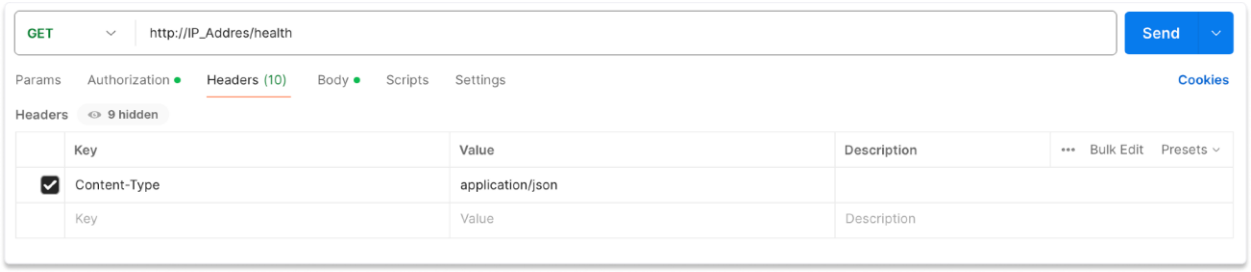

Рисунок 2. Пример заполнения Headers в Postman

Пример cURL-запроса для выполнения запроса из командной строки:

```
curl -X 'GET' \
```

```
'https://{url}/health' \
-H 'accept: application/json'
```

8.1.3. Результат выполнения запроса
Если модель работоспособна, будет получен ответ [200 OK].
Результат успешного запроса:

Рисунок 3. Пример успешного ответа

8.2 — ПОЛУЧЕНИЕ ID МОДЕЛИ

8.2.1. Описание запроса
Назначение: Этот запрос позволяет получить список доступных вам объектов с типом модель.
Тип запроса: GET
Запрос:

```
http://IP_Addres/v1/models
```

- Где **IP_Address** необходимо заменить на IP-адрес вашей машины. Например:

```
http://1.1.1.1:8000/v1/models
```

8.2.2. Выполнение запроса
Заполните раздел Headers по данным из таблицы ниже.

Key	Value
Content-Type	application/json

Пример заполнения в Postman представлен на рисунке ниже.

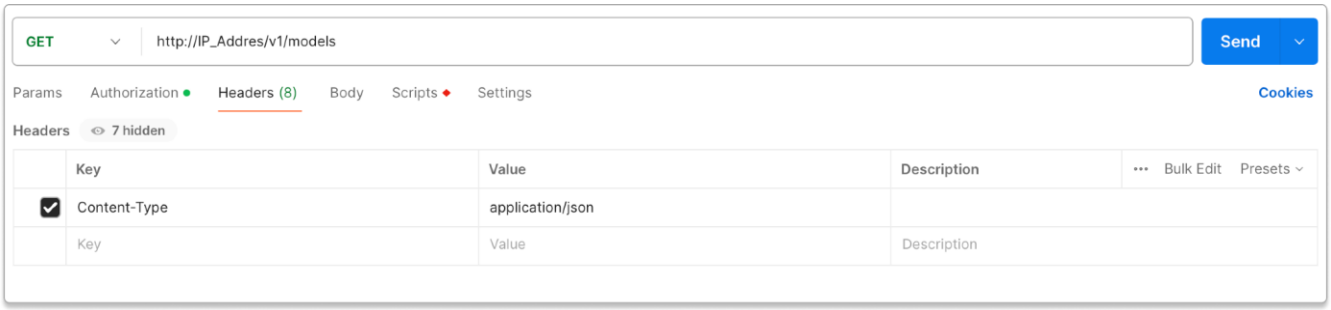

Рисунок 4. Пример заполнения Headers в Postman

Пример cURL-запроса для выполнения запроса из командной строки:

```
curl 'https://{url}/v1/models' \
--header 'Authorization: Bearer Token'
```

8.2.3. Результат выполнения запроса

Сохраните себе ID модели в разделе data.

Параметры ответа	Описание
id	Уникальный идентификатор модели.
object	Обозначает, что в ответе на данный запрос перечислены объекты с типом "model".
created	Unix-временная метка (в секундах) — указывает время создания модели
owned_by	Идентифицирует владельца модели.

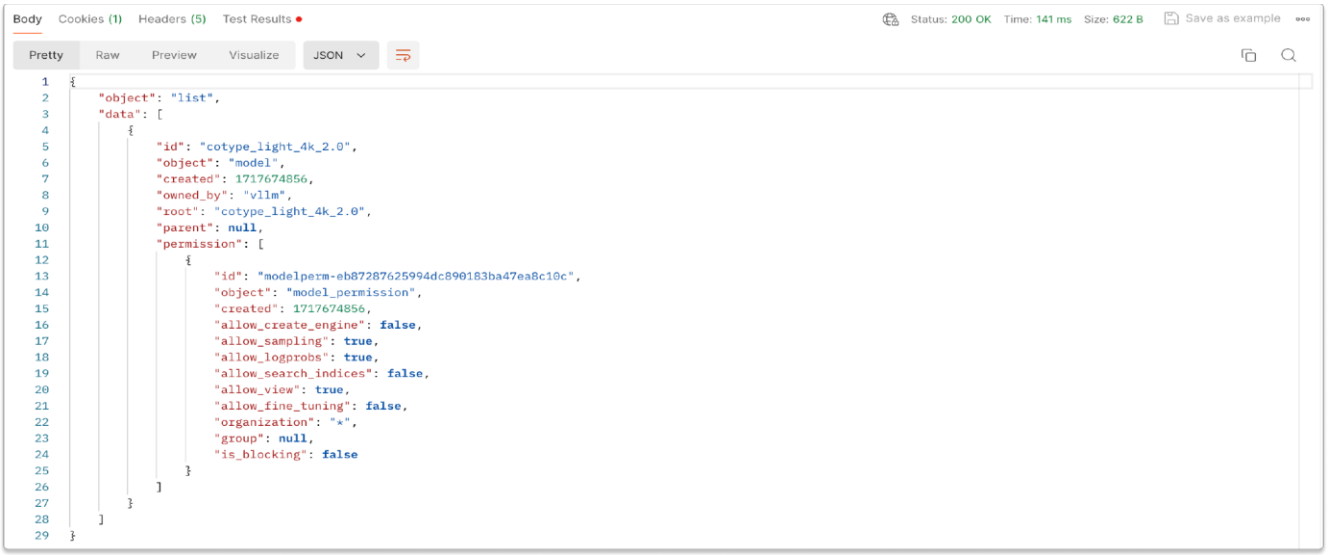

Рисунок 5. Пример успешного ответа

Результат успешного запроса:


```
"object": "list",
"data": [
  {
    "id": "/data/RnD/hf_hub/intema_cotype_70b_arabic",
    "object": "model",
    "created": 1715594312,
    "owned_by": "vllm",
    "root": "/data/RnD/hf_hub/intema_cotype_70b_arabic",
    "parent": null,
    "permission": [
      {
        "id": "modelperm-a18e2b7ef70a4e60b72502331c34966b",
        "object": "model_permission",
```

```
            "created": 1715594312,
            "allow_create_engine": false,
            "allow_sampling": true,
            "allow_logprobs": true,
            "allow_search_indices": false,
            "allow_view": true,
            "allow_fine_tuning": false,
            "organization": "*",
            "group": null,
            "is_blocking": false
        }
    ]
  }
 ]
}
```


8.3 — ОТПРАВКА СООБЩЕНИЯ

8.3.1. Описание запроса

Назначение: Этот запрос предназначен для обработки списка сообщений и генерации ответа на основе входных данных.

Тип запроса: POST

Запрос:

```
http://IP_Addres/v1/chat/completions
```

- Где **IP_Address** необходимо заменить на IP-адрес вашей машины. Например:

```
http://1.1.1.1:8000/v1/chat/completions
```

Запрос выполняется с параметрами, указанными в Таблице 1 и полученными в запросе models.

Название параметра	Описание	Значение
model	Идентификатор модели	Например,thread_abc123

8.3.2. Выполнение запроса

Заполните раздел Headers по данным из таблицы ниже.

Key	Value

Content-Type	application/json

В разделе Body введите код ниже, заменив значение model ID вашей модели.

```
{"model": "//имя вашей модели", "messages": [
{"role": "system", "content": "You are a helpful assistant."},
{"role": "user", "content": "Hello! Answer in English."}]}'
```

Пример заполнения представлен на рисунке ниже.

Рисунок 6. Пример заполнения раздела Body

Пример cURL-запроса для выполнения запроса из командной строки:

```
curl  https://{url}/v1/chat/completions \
-H "Content-Type: application/json" \
-H "Authorization: Bearer Token" \
-d '{"model": "/data/RnD/hf_hub/intema_cotype_70b_arabic", "messages": [
{"role": "system", "content": "You are a helpful assistant."},
{"role": "user", "content": "Hello! Answer in English."}]}'
```

8.3.3. Результат выполнения запроса

Параметры ответа	Описание
message	Метод передачи списка сообщений модели включает в себя объекты с ролями "system", "user" и "assistant", что позволяет учитывать контекст диалога и обеспечивает более точные ответы. Системное сообщение задает поведение ассистента, а передача истории диалога важна для понимания контекста и корректного выполнения запросов пользователя.

Рисунок 7. Пример успешного ответа

Результат успешного запроса:

{
"id": "cmpl-cf6531a8efb44ffba8d01dfd4e03863f",
"object": "chat.completion",
"created": 1715603793,
"model": "/data/RnD/hf_hub/intema_cotype_70b_arabic",
"choices": [
{
"index": 0,
"message": {
"role": "assistant",
"content": "Hello! I'm 3rbi, your personal assistant developed by the research team at Intema. I'm here to help answer your questions and provide assistance in Arabic. However, since you asked me to respond in English, I'll be happy to do so! What's on your mind, and how can I assist you today?"
},
"logprobs": null,
"finish_reason": "stop",
"stop_reason": 128009
}
],
"usage": {
"prompt_tokens": 71,
"total_tokens": 140,
"completion_tokens": 69

```
  }
}
```

Генеральный директор
ООО «МТС ИИ»

Калинин А.Л.