РУКОВОДСТВО ПО ЭКСПЛУАТАЦИИ СИСТЕМЫ «Сервис генерации естественного языка»

Версия 1.0 • 01/06/2024

Оглавление

ГЛОССАРИЙ

Термин	Определение
Искусственный интеллект (ИИ)	область компьютерных наук, занимающаяся созданием вычислительных систем, способных выполнять задачи, требующие человеческого интеллекта, такие как восприятие, рассуждение, обучение и решение проблем.
Машинное обучение	раздел искусственного интеллекта, в котором вычислительные системы обучаются выполнять задачи, анализируя и обобщая данные. Обучение происходит без явного программирования специфических инструкций.
Генеративные модели	типы искусственного интеллекта, способные создавать новый контент, включая тексты, изображения и музыку. Такие модели обучаются на больших объемах данных и затем генерируют новый контент, имитируя наблюдаемые данные.
Дообучение (Fine-tuning)	процесс настройки предобученной модели под конкретную задачу путем дополнительного обучения на более мелком и специфическом наборе данных.
Нейронная сеть	математическая модель, состоящая из взаимосвязанных искусственных нейронов, организованных в слои, предназначенная для выполнения задач машинного обучения и обработки данных.
Токен	минимальная единица текста, например, слово или символ. Применяется в обработке естественного языка для анализа и генерации текста.
Натуральный язык (NLP)	область искусственного интеллекта, занимающаяся взаимодействием между вычислительными системами и людьми на естественных языках, таких как русский или английский.
Application Programming Interface (API)	набор правил и инструментов для взаимодействия программного обеспечения. API предоставляет возможность различным приложениям обмениваться данными и функциональностью.
Docker	платформа для автоматизации развёртывания вычислительных приложений в контейнерах. Обеспечивает изоляцию приложений и независимость от среды выполнения.
(JavaScript Object Notation) JSON	лёгкий формат обмена данными. Формат легко читается человеком и парсируется компьютером.

ВВЕДЕНИЕ

Настоящий документ представляет собой руководство по эксплуатации системы Сервис генерации естественного языка.

Руководство описывает:

- общее определение системы;
- функционал системы;
- архитектуру системы;
- эксплуатацию системы через API-запросы.

1 — О ПРОДУКТЕ

Сервис генерации естественного языка — это большая языковая система для генерации и редактирования текстов, суммирования и анализа информации.

Доступны для поставки 3 версии продукта:

- Cotype Light 4k 2.0;
- Cotype Plus 16k 1.0;
- Cotype Pro 8k 1.0.

1.1. Функционал продукта

- Генерация текста: автоматизированное создание контента, включая статьи, отчеты, описания продуктов.
- Анализ текста: извлечение смысла текста, классификация по темам, определение тональности, распознавание именованных сущностей, анализ эмоциональной окраски.
- Автоматический перевод: перевод текста между различными языками.
- Семантический поиск: поиск релевантной информации по смыслу в больших объемах текста.
- Поддержка диалогов: разработка и внедрение чат-ботов и виртуальных ассистентов для ведения бесед, ответов на вопросы и выполнения команд на естественном языке.
- Обработка и анализ больших данных: анализ текстовой информации для выявления трендов и паттернов, предоставление бизнес-инсайтов.

> В каждую систему, 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 системы.

2 — АРХИТЕКТУРА

> Разделы в данной главе дают общее представление о работе системы «Сервис генерации естественного языка», сервисах системы и создаваемых типах данных.

2.1. Сервисы

Таблица 1. Сервисы системы

Сервис	Описание
API Gateway	Компонент, предоставляющий интерфейс для доступа к LLM
Сервис аутентификации и авторизации	Сервис отвечает за хранение пользователей, проверку и выдачу токенов.
Load Balancer	Сервис, отвечающий за распределение нагрузки на LLM-модель.
Backend-App	Компонент, который непосредственно хранит LLM-модель.

API GATEWAY

API Gateway — это компонент, предоставляющий интерфейс для доступа к LLM. Он обрабатывает запросы от внешних сервисов, таких как веб-приложение, взаимодействует с LLM и возвращает ответы обратно внешним системам.
Функционал:

- Обработка входящих запросов;
- Взаимодействие с LLM;
- Возврат ответов внешним сервисам.

СЕРВИС АУТЕНТИФИКАЦИИ И АВТОРИЗАЦИИ

Сервис отвечает за хранение пользователей, проверку и выдачу токенов. Каждый входящий запрос в API Gateway инициируется другими сервисами и проходит через этот компонент для проверки подлинности.

Функционал:

- Хранение пользователей в PostgreSQL базе данных;
- Проверка подлинности запросов;
- Выдача токенов аутентификации.

LOAD BALANCER

Сервис, отвечающий за распределение нагрузки на LLM-модель. Он обеспечивает оптимальное использование ресурсов и предотвращает перегрузку системы.
Функционал:

- Распределение входящих запросов;

- Оптимизация использования ресурсов.

BACKEND-APP
Компонент, который непосредственно хранит LLM-модель. Он получает на вход запросы к модели и возвращает ответы.
Функционал:

- Хранение и управление LLM-моделью;
- Обработка запросов к модели;
- Генерация ответов.

3 — ПРОГРАММНЫЕ И АППАРАТНЫЕ ТРЕБОВАНИЯ

3.1. Требования к программному обеспечению рабочей станции

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

Таблица 2. Требования к программному обеспечению

Ресурс	**Требования**
Операционная система	Linux: Ubuntu
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

3.2. Требования к аппаратному обеспечению рабочей станции

Для работы системы необходимо, чтобы выполнялись требования к аппаратным ресурсам.

Система с минимальными требованиями - «Cotype Light 4k 2.0», требования для неё указаны в Таблице 3.

Таблица 3. Минимальные требования к аппаратному обеспечению

Ресурс	**Требования**
CPU	от 8
RAM	от 32 GB
GPU	1x NVIDIA Tesla V100
SSD	500 GB
Видеодрайвер	Driver Version: 525.105.17, CUDA Version: 12.0

Для системы «Cotype Plus 16k 1.0», «Cotype Pro 8k 1.0» и «Cotype Plus 32k 1.0» рекомендуемые параметры указаны в Таблице 4.

Таблица 4. Минимальные требования к аппаратному обеспечению

Ресурс	**Требования**
CPU	от 28
RAM	от 100 GB
GPU	1x NVIDIA A100 80GB
SSD	500 GB - 1 TB
Видеодрайвер	Driver Version: 525.105.17, CUDA Version: 12.0

4 — СПРАВОЧНИК ПО API

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

4.1. COMPLETIONS

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

Таблица 5. Параметры метода COMPLETIONS

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 система 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 система 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 системой.
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 система повторит одну и ту же строку.
presence_penalty		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 система будет говорить о новых темах.

4.2. MODELS

Метод для получения списка доступных вам моделей.

4.3. HEALTH

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

5 — 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:
1. Получите авторизационные данные для вашего приложения.
Данные в виде имени модели и токена будут переданы вместе с образом.
2. Добавьте сценарий получения Access Token в сервис для вызова API.
3. В запросах замените MTSAI_API_KEY на полученный токен и укажите ваше название модели.

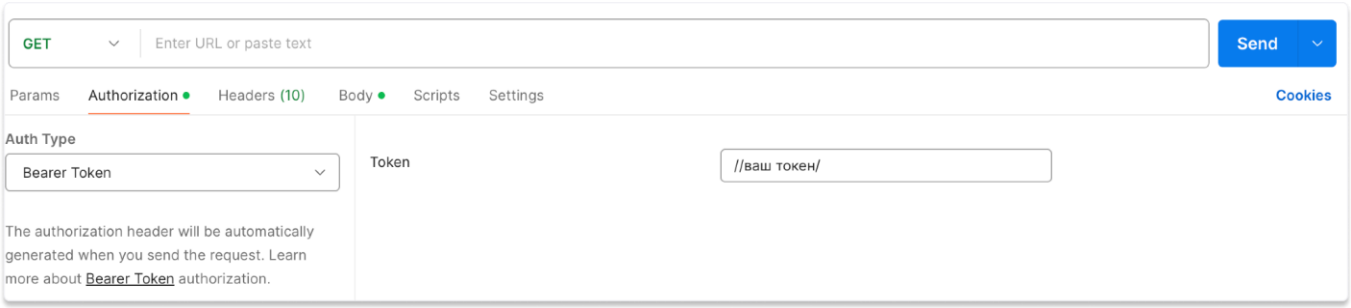

Рисунок 1. Пример заполнения данных в Postman

6 — ЭКСПЛУАТАЦИЯ СЕРВИСА ЧЕРЕЗ API-ЗАПРОСЫ

Заказчик последовательно отправляет HTTP-запросы к сервисам системы 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

6.1. Проверка работоспособности сервиса

6.1.1. Описание запроса

Назначение: Запрос проверяет работоспособность сервиса. Если сервис работоспособен, 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
```

6.1.2. Выполнение запроса

Заполните раздел Headers по данным из таблицы ниже.

Таблица 6. Заголовки для выполнения запроса

Key	Value
Content-Type	application/json

Пример заполнения в Postman представлен на рисунке ниже.

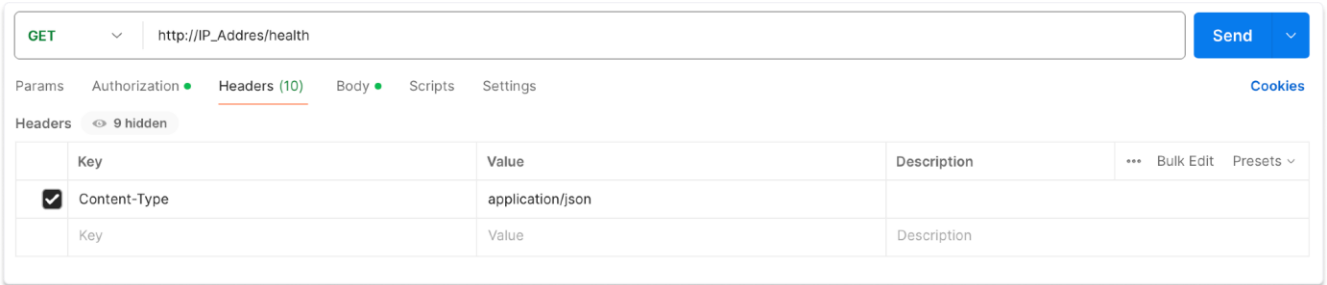

Рисунок 2. Пример заполнения Headers в Postman

Пример cURL-запроса для выполнения запроса из командной строки:

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

6.1.3. Результат выполнения запроса

Если система работоспособна, будет получен ответ [200 OK].

Результат успешного запроса:

Рисунок 3. Пример успешного ответа

6.2. Получение ID модели

6.2.1. 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
```

6.2.2. Выполнение запроса

Заполните раздел Headers по данным из таблицы ниже.

Таблица 7. Заголовки для выполнения запроса

Key	Value
Content-Type	application/json

Пример заполнения в Postman представлен на рисунке ниже.

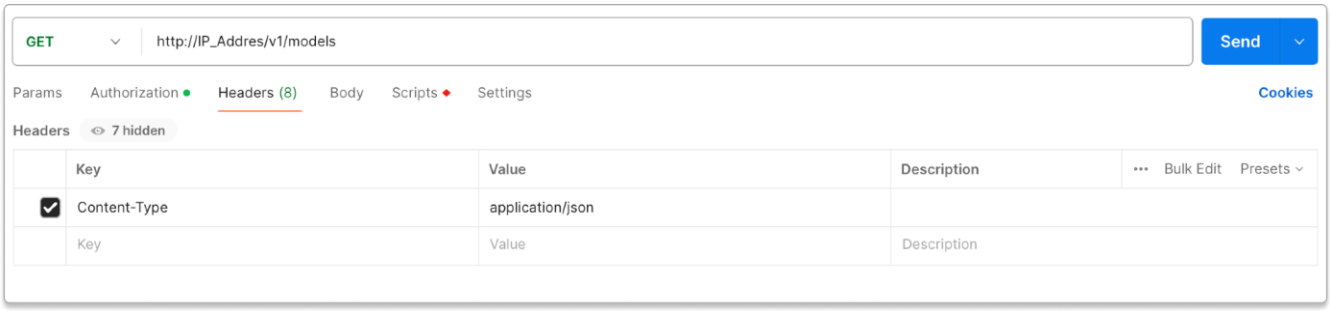

Рисунок 4. Пример заполнения Headers в Postman

Пример cURL-запроса для выполнения запроса из командной строки:

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

6.2.3. Результат выполнения запроса представлен на рисунке ниже. Сохраните себе ID модели в разделе data.

Таблица 8. Результат выполнения запроса

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.

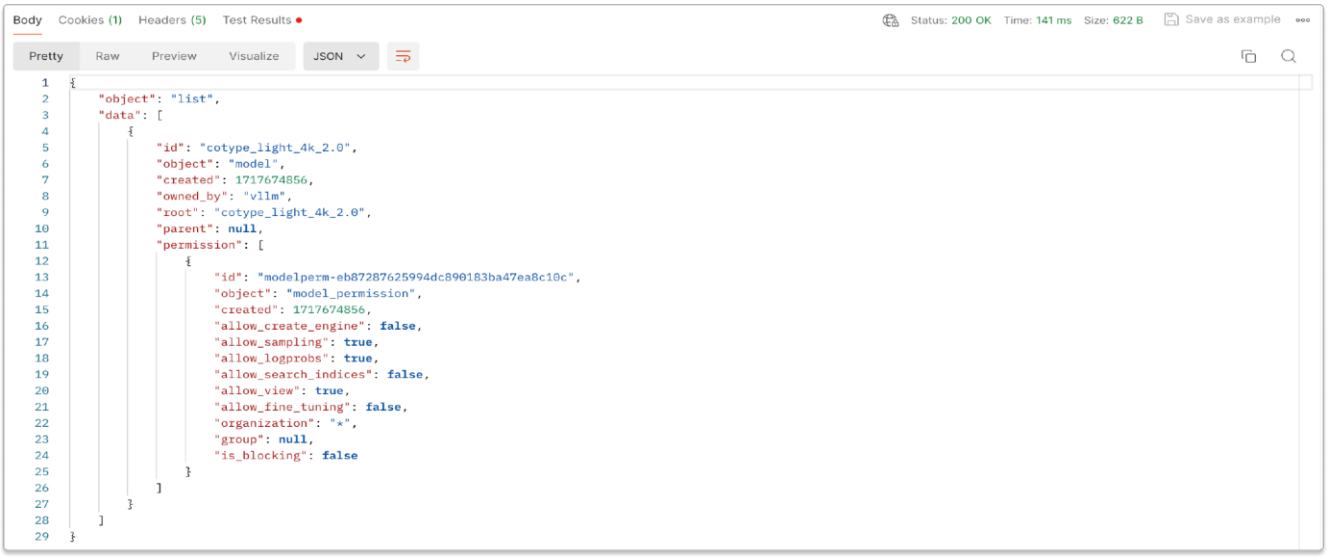

Рисунок 5. Пример успешного ответа

Результат успешного запроса:


```
    "object": "list",
    "data": [
        {
            "id": "/data/RnD/hf_hub/intema_cotype_70b ",
            "object": "model",
            "created": 1715594312,
            "owned_by": "vllm",
            "root": "/data/RnD/hf_hub/intema_cotype_70b",
            "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
                }
            ]
        }
    ]
}
```

6.3.3. Отправка сообщения

6.3.1. 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
```

6.3.2. Выполнение запроса

Заполните раздел Headers по данным из таблицы ниже.

Таблица 9. Заголовки для выполнения запроса

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

6.3.3. Результат выполнения запроса

Таблица 10. 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 ",
    "choices": [
        {
            "index": 0,
            "message": {
                "role": "assistant",
                "content": "Hello! I'm your personal assistant. I'm here to help answer your questions and provide assistance. 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
    }
}
```