РУКОВОДСТВО ПО УСТАНОВКЕ 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 установке системы “Сервис генерации естественного языка”.

Руководство описывает:

- общее определение системы;
- функционал системы;
- архитектуру системы;
- установку и настройку системы.

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, 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 — 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 с помощью API-запроса GET/health. Запрос подробно описан в пункте 7.1. Проверка работоспособности модели.

5 — 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.

5.1. 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 5. 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%.

5.2. 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 6. 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.

5.3. 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 7. 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.

5.4. Пример заполнения файла

5.4.1.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>

5.4.2. 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:
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.

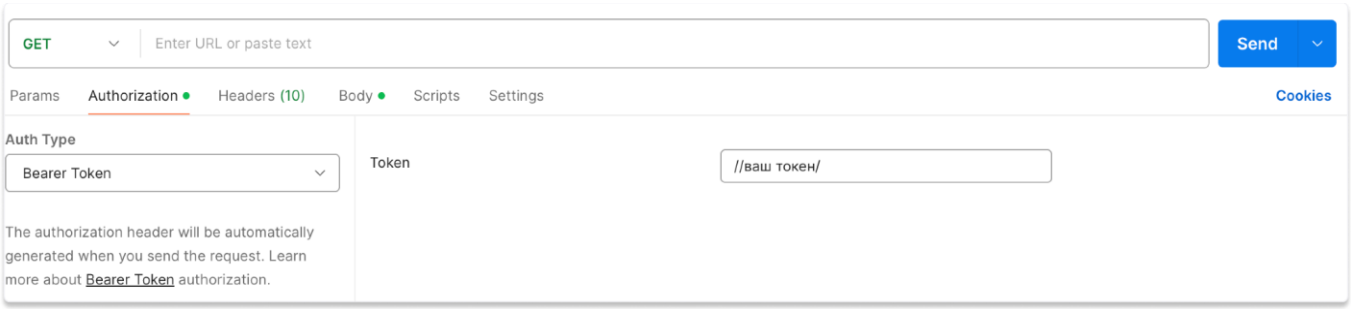

Рисунок 1. Пример заполнения данных в Postman

7 — ПРОВЕРКА РАБОТОСПОСОБНОСТИ СИСТЕМЫ

Заказчик последовательно отправляет HTTP-запросы к сервисам системы через сервис API и получает в ответ пакет данных.

7.1. Проверка работоспособности системы

7.1.1. Описание запроса

Назначение: Запрос проверяет работоспособность системы. Если система 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
```

7.1.2. Выполнение запроса

Заполните раздел Headers по данным из таблицы ниже.

Таблица 8. Заголовки для выполнения запроса

Key	Value
Content-Type	application/json

Пример заполнения в Postman представлен на рисунке ниже.

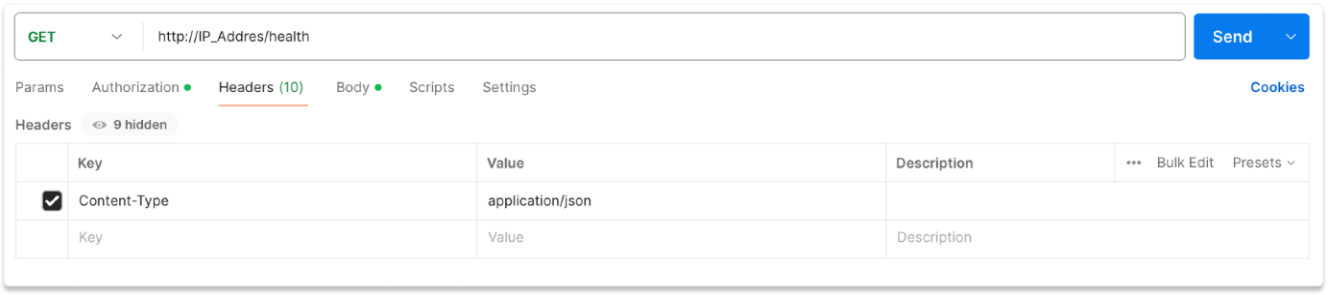

Рисунок 2. Пример заполнения Headers в Postman

Пример cURL-запроса для выполнения запроса из командной строки:

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

7.1.3. Результат выполнения запроса

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

Результат успешного запроса:

Рисунок 3. Пример успешного ответа