1. Введение

1.1 Описание поставляемого Экземпляра ПО с целью экспертной проверки

Экземпляр ПО Умка ИИ с целью экспертной проверки предоставляется в виде файла виртуальной машины с предустановленными сервисами Умка ИИ, тестовой базой данных и наборов тестовых случаев для демонстрации работы **Умка ИИ.**

Правообладатель готов продемонстрировать экземпляр программного обеспечения, для чего просим связаться с нами:

телефон: +7-(495)-308-34-85

почта: umkaaipro@gmail.com

1.2 Системные требования

Системные требования для ПК, на котором будет развёрнута виртуальная машина с системой **Умка ИИ**:

- Операционная система Ubuntu (рекомендуемая версия системы: Ubuntu 22.04).
- 512 Гб памяти на жёстком диске (желательно использование SSD).
- 32 Гб ОЗУ.
- NVIDIA A100 80GB

2. Запуск ПО Умка ИИ

Предварительные требования

1. **Сервер с Linux (Ubuntu или любой другой дистрибутив)**:
    - Имеется SSH-доступ.
    - На сервере установлен Docker и Docker Compose.
        i. **Docker** должен быть установлен как сервис **systemd** (через `apt`), а **не как snap-пакет**. Это гарантирует корректную интеграцию с системными службами.
    - Необходим доступ в интернет для скачивания образов из Docker Hub.
2. **Docker Hub учётная запись**:
    - Необходимо иметь логин и пароль для `docker login` для выгрузки приватных изображений проекта
3. **Файлы проекта**:
    - `docker-compose.yml` — основной файл, в котором описаны сервисы стэка.
    - `Caddyfile` — конфигурационный файл для reverse-прокси (Caddy).

- `.env` — файл с переменными окружения, необходимыми для работы сервисов (БД, Redis, S3, JWT-секреты и др.).

4. **Переменные окружения**:
   - Все необходимые данные (пароли, ключи, хосты) должны быть указаны в `.env` файле.
5. **Привязка домена:**
   - Домен должен быть привязан к IP-адресу основного сервера.
   - Убедитесь, что DNS-записи (A-запись или CNAME) настроены корректно.
6. **Открытые порты:**
   - Предварительно откройте следующие порты на сервере:
     i. 80 и 443 — для Caddy (HTTP и HTTPS).
     ii. 9000 и 9001 — для MinIO (админ-интерфейс и API).
   - После завершения настройки MinIO рекомендуется закрыть порты 9000 и 9001 для обеспечения безопасности.

Шаги по деплою

1. Подготовка файлов проекта на сервере

Создайте целевую директорию для проекта, например:

```
mkdir -p /root/umka_core
cd /root/umka_core
```

2. Скопируйте необходимые файлы на сервер.
Вот необходима иерархия файлов:

```
umka_core/
├── .env
├── docker-compose.yml
├── caddy_server/
│   ├── Caddyfile
```

3. Настройка переменных окружения (.env)

Откройте `.env` для редактирования:

```
nano /root/umka_core/.env
```

- Внесите все нужные значения:
  - `SECRET_KEY_JWT` - Секретный ключ для подписания JWT (JSON Web Token).
  - `ALGORITHM_JWT` - Алгоритм, используемый для шифрования JWT.
  - `ACCESS_TOKEN_EXPIRE_MINUTES` - Время жизни токена доступа в минутах.
  - `HTTPS_IS` - Флаг, указывающий, используется ли HTTPS.

    - SAME_SITE - Политика Cookie. Значение strict указывает, что Cookie отправляется только с запросами с того же сайта.
    - ALLOWS_ORIGIN - Указывает разрешённый источник для CORS (Cross-Origin Resource Sharing).
    - NEXT_PUBLIC_API_URL - URL-адрес Backend API, используемого клиентской частью приложения.
    - DB_HOST - Хост для подключения к базе данных.
    - DB_NAME - Имя базы данных.
    - DB_USER - Имя пользователя базы данных.
    - DB_PASSWORD - Пароль для подключения к базе данных.
    - REDIS_PASSWORD - Пароль для подключения к серверу Redis.
    - REDIS_DB - Номер базы данных в Redis (по умолчанию 0).
    - REDIS_HOST - Хост Redis-сервера.
    - REDIS_PORT - Порт для подключения к Redis. По умолчанию используется 6379.
    - ACCESS_KEY_S3 - Ключ доступа к MinIO. Пока не заполнен. На данном этапе должен быть пустым.
    - SECRET_KEY_S3 - Секретный ключ для доступа к MinIO. На данном этапе должен быть пустым.
    - BUCKET_NAME_S3_UPLOAD - Имя bucket для загрузки файлов. На данном этапе должен быть пустым.
    - S3_HOST - Хост для доступа к MinIO.
    - S3_HOST_EXTERNAL - Внешний URL-адрес для доступа к MinIO.
    - S3_PATH_PREFIX - Префикс пути для файлов в MinIO. Нужен для внешнего доступа к s3 через caddy.
    - MINIO_ROOT_USER - Имя администратора MinIO.
    - MINIO_ROOT_PASSWORD - Пароль администратора MinIO.
    - VLLM_SERVER_URL - URL-адрес сервера для работы с LLM (Large Language Model).
    - WHISPER_SERVER_URL - URL-адрес сервера для работы с Whisper (модель для транскрибации).
- Сохраните файл после редактирования.

4. Авторизация в Docker Registry (если образы приватные)

Выполните авторизацию в Docker Hub с приложенными логином и паролем `docker login -u "<нужный_docker_hub_username>" -p "<нужный_docker_hub_password>"`

```
docker login -u umkaai
dckr_pat_p7UmmUmC9sA04bSpnJBrH-pYjiA
```

`Добавить pull изображений из docker hub первоначальный`

5. Инициализация Docker Swarm (если не сделано ранее)

Инициализируйте Swarm (если у вас одиночный сервер):
`docker swarm init`

- Если Swarm уже инициализирован (например, в предыдущих деплоях), можно пропустить этот шаг или игнорировать ошибку о том, что Swarm уже существует.

6. Добавление Caddyfile в Docker Config (для Docker Swarm)

В `docker-compose.yml` предполагается использование `docker config`.

6.1 Создайте конфиг на основе `Caddyfile_sample`:

```
docker config rm caddyfile_config || true
docker config create caddyfile_config Caddyfile
```

- Здесь мы сначала пытаемся удалить предыдущий конфиг (если был), а затем создаём новый.

6.2 Настройка Caddyfile:

- **`your_domain.ru`:** Укажите домен, привязанный к вашему серверу. Это будет основная точка входа для всех запросов.
- **`handle_path /web-api/*`:** Запросы, начинающиеся с `/web-api/`, перенаправляются на `backend_api:8000`. Здесь `backend_api` — имя контейнера, указанное в `docker-compose.yml`.
- **`handle /s3/*`:** Запросы для S3 обрабатываются MinIO на порту **9000**, при этом путь `/s3` удаляется из запроса (с помощью `uri strip_prefix /s3`).
- **`handle /s3-admin/*`:** Административные запросы к MinIO перенаправляются на порт **9001**.
- **`handle`:** Все остальные запросы перенаправляются на фронтенд (порт **3000**, контейнер `frontend`).

7. Деплой проекта через Docker Stack

Выполните команду деплоя

```
export $(cat .env | xargs)

docker stack deploy -c docker-compose.yml my_stack
```

Здесь `my_stack` — имя стэка, под которым будут запускаться все сервисы.
После этого Docker будет скачивать необходимые образы (если их ещё нет) и запускать контейнеры.

8. Настройка MinIO

Откройте браузер и перейдите по адресу:
arduino
Копировать код
`http://<ваш_ip>:9001`

1.
    - Используйте `MINIO_ROOT_USER` и `MINIO_ROOT_PASSWORD` для входа.
2. Создайте бакет (bucket) для хранения данных.
3. Сгенерируйте **access token** для приложения:
    - Перейдите в раздел настройки MinIO.
    - Создайте токены доступа (Access Key и Secret Key).

Запишите названия бакета и токены в `.env`:

```
S3_BUCKET_NAME=<имя_бакета>

S3_ACCESS_KEY=<access_key>

S3_SECRET_KEY=<secret_key>
```

---

9. Закрытие портов 9000 и 9001

После настройки MinIO, закройте порты для безопасности:

Используйте `ufw` (если установлен):

```
sudo ufw deny 9000

sudo ufw deny 9001
```

Либо настройте правила через `iptables` или любой другой файрвол.

10. Удалите текущий стек

Прежде чем перезапускать проект, остановите текущие сервисы:

```
docker stack rm my_stack
```

Убедитесь, что все контейнеры и сервисы стека остановлены:

```
docker stack ps my_stack
```

- Если вывод пустой — стек удалён.

11. Выполните повторный деплой

Запустите стек с учетом обновленных файлов:

```
export $(cat .env | xargs)
```

```
docker stack deploy -c docker-compose.yml my_stack
```

3. Запуск приложения

Для запуска приложения необходимо ввести в адресную строку браузера адрес и нажать клавишу «Enter» на клавиатуре. После этого откроется окно авторизации оператора.

Данные авторизации:

- Пользователь: **super@admin.com**.
- Пароль: super_passwordExt1

Далее необходимо действовать в соответствии с документом «Общее руководство», раздел «3. Начало работы».

3.1 Особенности демонстрационной сборки

Тестовая сборка содержит тестовые данные для авторизации, которые необходимы для использования системы.

4. Структура файлов ПО Умка ИИ

umka_core/
├── .env
├── docker-compose.yml
├── caddy_server/
│ ├── Caddyfile
│ ├── caddy_config/
│ └── caddy_data/

├── db/redis/

├── db/postgres/

└── minio/

└── whisper/

└── llm/

5. Список используемых компонентов и библиотек.

В составе ПО Умка ИИ используются следующе стороннее ПО.

Сторонний компонент/сервис	**Название лицензии**	**Ссылка на лицензию**	**Ссылка на репозиторий**
Redis (open-source edition)	SSPL	https://redis.io/legal/server-side-public-license-sspl/	https://github.com/redis/redis
Docker Engine 24.0	Apache License 2.0	https://github.com/moby/moby/blob/master/LICENSE	https://github.com/moby/moby
Python 3	PSFL	https://github.com/python/cpython/blob/main/LICENSE	https://github.com/python/cpython
Ubuntu 22.04 LTC	GNU GPL, LGPL	https://github.com/canonical/ubuntu.com/blob/main/LICENSE.md	https://github.com/canonical/ubuntu
GNU Compiler Collection (GCC)	GNU GENERAL PUBLIC LICENSE Version 2 GNU LESSER GENERAL PUBLIC LICENSE Version 2.1 GCC RUNTIME LIBRARY EXCEPTION	https://github.com/gcc-mirror/gcc/blob/master/COPYING https://github.com/gcc-mirror/gcc/blob/master/COPYING.LIB https://github.com/gcc-mirror/gcc/blob/master/COPYING.RUNTIME	https://github.com/gcc-mirror/gcc
vLLM	Apache License 2.0	https://github.com/vllm-project/vllm/blob/main/LICENSE	https://github.com/vllm-project/vllm

Сторонний компонент/сервис	Название лицензии	Ссылка на лицензию	Ссылка на репозиторий
PyTorch	BSD 3-Clause License	https://github.com/pytorch/pytorch/blob/main/LICENSE	https://github.com/pytorch/pytorch
MinIO	GNU Affero General Public License v3.0.	https://www.gnu.org/licenses/agpl-3.0.en.html	https://github.com/minio/minio
PostgreSQL 13	The PostgreSQL Licence	https://opensource.org/license/postgresql	https://github.com/postgres/postgres
Caddy	Apache License	https://github.com/caddyserver/caddy/blob/master/LICENSE	https://github.com/caddyserver/caddy

Таблица 1 — Стороннее ПО

Правообладатель готов продемонстрировать экземпляр программного обеспечения, для чего просим связаться с нами:

телефон: +7-(495)-308-34-85

почта: umkaaipro@gmail.com