1. **Общая информация**

**Цель проекта:**

Разработать Telegram-бота, который поможет наладить эффективное взаимодействие между администраторами и стажерами. Бот будет выполнять функции сбора контактных данных, обмена документами, коммуникации и организации задач.

**Назначение системы:**

Система предназначена для автоматизации процессов работы HR отдела и оптимизации временных издержек при постановлении задачи, приеме на работу или увольнении с работы, получении необходимых сведений. Повысить эффективность и качество адаптации стажеров, сократив временные и организационные затраты администраторов на рутинные операции и улучшив коммуникацию между всеми участниками процесса.

* **Автоматизировать** рутинные процессы (сбор данных, ответы на частые вопросы, напоминания).
* **Стандартизировать** процесс onboarding-а (чтобы каждый стажер получал одинаково полную и структурированную информацию).
* **Централизовать** информацию и документооборот (исключить ситуацию, когда договор «где-то в чате»).
* **Ускорить**коммуникацию (стажер всегда знает, к кому и по какому вопросу обратиться).
* **Обеспечить** легкий доступ к учебным материалам и задачам для стажера.
* **Снизить**нагрузку на администраторов и HR, освободив их время для более важных и персонализированных задач.

**Функциональные требования:**

0. Роли в приложении:

* Администратор (админ) - пользователь с наивысшими привилегиями, имеет функционал осмотра всех стажеров, видит списки, задачи и дедлайны каждого;
* Стажер - пользователь-стажер, видит только себя и свои задачи;
* Суперадминистратор (суперадмин) - контролирует регистрацию админов, приходит сообщение, одобряющее или не одобряющее регистрацию администратора, удаление с роли админа, следит за базой данных.

1. Сбор контактных данных

* Кнопка "Поделиться контактом";
* После нажатия, бот должен запросить у пользователя разрешение на получение его username;
* Username стажера должен быть передан администратору.

2. Отправка резюме

* Функционал отправки резюме;
* Пользователь должен иметь возможность загрузить резюме в формате PDF или DOCX;
* Бот передает загруженное резюме на утвержденный контакт администратора.

3. Отправка сообщений от администратора

* Сообщения от "Администратора";
* Администратор может отправлять сообщения любому стажеру через бота;
* Имя отправителя будет отображаться как "Администратор стажировки".

4. Управление задачами

* Создание и отслеживание задач;
* Администратор может создавать задачи для одного или нескольких стажеров;
* У каждой задачи может быть установлен конкретный дедлайн;
* Стажеры получают уведомление о новой задаче и ее дедлайне.

5. Отправка результатов

* Прикрепление документов;
* Стажеры имеют возможность прикреплять файлы (документы, скриншоты и т.д.) и отправлять их через бот;
* Файлы должны быть переданы в утвержденное хранилище или контакт администратора.

**Нефункциональные требования:**

1. Безопасность

Доступ к управлению ботом (административные функции) только для уполномоченных лиц.

2. Производительность

Бот должен отвечать на команды пользователя не более чем за 5 секунды.

3. Интерфейс

* Удобный и интуитивно понятный интерфейс для пользователей;
* Легкий доступ к основным функциям через кнопки и команды.

4. Системные требования

Совместимость с Telegram API и актуальными версиями мессенджера.

**Технические детали разработки:**

1. Языки программирования и технологии

* Backend: Любой;
* Внешние сервисы: для хранения файлов могут использоваться облачные хранилища или что угодно.

2. Инфраструктура

* Размещен на сервере с доступом в интернет. (сервер можем предоставить);
* Использование баз данных для хранения информации о задачах и их статусах.

**Разработка и тестирование:**

* Дизайн и проектирование архитектуры бота: до 01.03.2025;
* Реализация функционала: до 20.04.2025;
* Тестирование всех функций и исправление ошибок: до 12.05.2025.

### **Потенциальные уязвимости:**

### **1. Уязвимости Веб-сервера и Веб-хука**

* **Подделка веб-хуков (Webhook Spoofing):**
* **Раскрытие IP-адреса сервера:**

### **2. Уязвимости Приложения (Кода Бота)**

* **Небезопасная обработка пользовательского ввода (Injection Flaws):**
* **Небезопасная десериализация данных:**
* **Неправильное управление доступом (IDOR & Privilege Escalation):**
* **Утечка чувствительной информации:**

### **3. Уязвимости Инфраструктуры (Docker, Ubuntu)**

* **Привилегированный контейнер**
* **Уязвимости в базовом образе Docker**

### **4. Уязвимости Данных (SQLite)**

* **Хранение секретов в открытом виде**
* **Отсутствие шифрования базы данных**

1. **Реализованные функции:**

**2.1 Общая структура проекта:**

Telegram-бот, развернутый на сервере Ubuntu и работающий через веб-хуки, получает обновления от Telegram в реальном времени. Каждый раз, когда пользователь отправляет сообщение или выполняет действие, Telegram отправляет HTTP-запрос на заранее настроенный URL сервера. Сервер с ботом принимает этот запрос, обрабатывает данные, и в зависимости от заложенной логики выполняет определенные действия: отправляет ответное сообщение, обрабатывает команды, взаимодействует с базой данных или внешними API.

Бот разворачивается через Docker Compose, используя конфигурацию из docker-compose.yml. Сервис работает в изолированном контейнере с подключенными томами (volumes) для данных и переменными окружения из .env. При запуске (docker-compose up) система автоматически создает сеть, монтирует файлы и запускает контейнер с ботом, который взаимодействует с внешними API (Telegram и др.). Ошибки мониторятся через логи (docker-compose logs), а пересборка образа выполняется с очисткой кэша (--no-cache).

База данных (далее - БД) реализована через SQLlite и SQLAlchemy, локально хранящие в себе данные, ключи и тд.

Структура проекта построена так, что функционал делиться на 3 основные части:

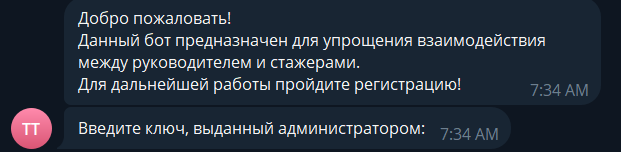
1. Использование бота со стороны стажера
2. Использование бота со стороны админа
3. Использование бота со стороны суперадмина

С кодом Telegram-бота можно ознакомиться в прилагаемом файле.

Сервер, на котором находится бот: 10.0.10.24

**3. Начало работы с ботом**

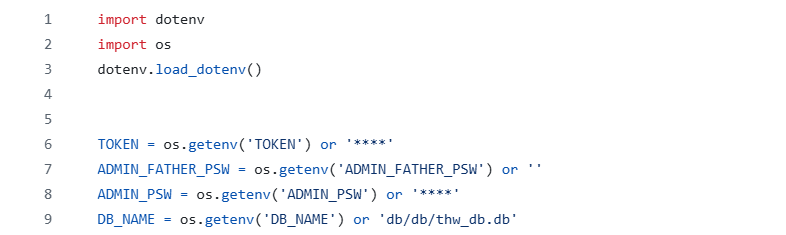
При активации бота командой “/start” будет выведено следующее:



Данный этап идет общим для всех видов пользователей(стажер, админ, суперадмин). Со стороны разработчика ключ указывается в docker-compose.yml файле.



После используется в файле [config.py](http://config.py)



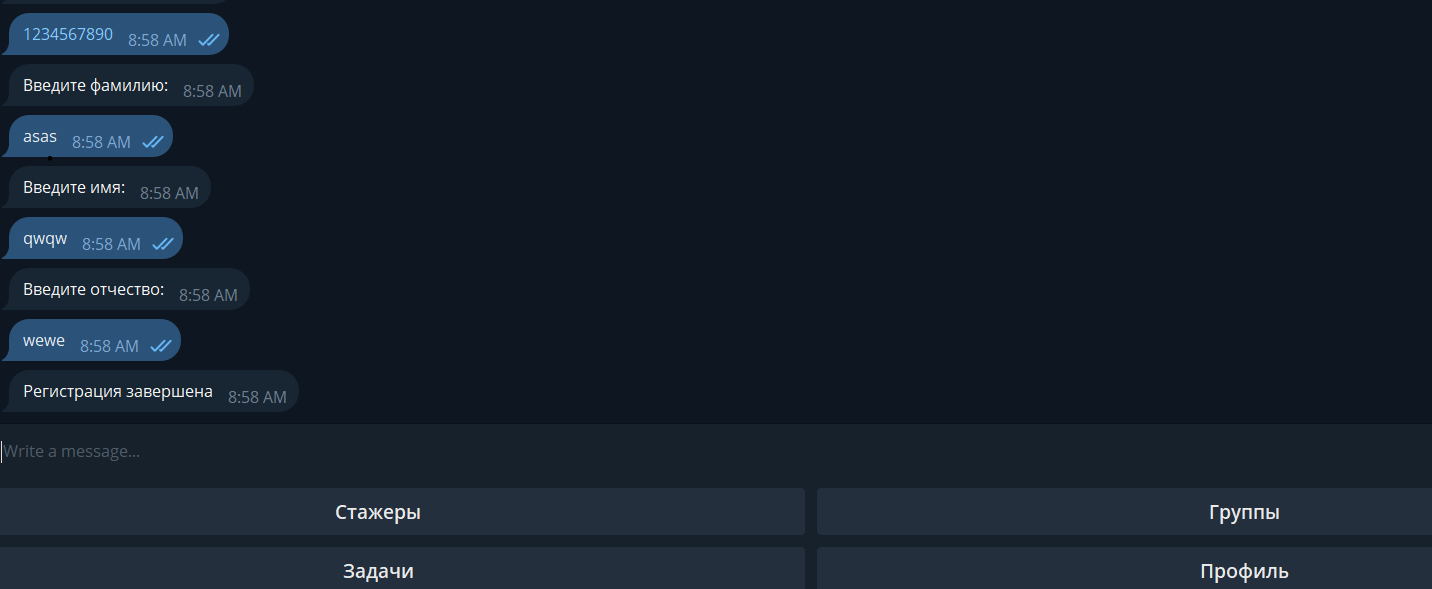
Далее рассмотрим функционал 3 разных пользователей.

*Большинство функций имеют обработку ошибок для стабильной работы бота, о некорректном вводе или выборе команды пользователю приходит уведомление.*

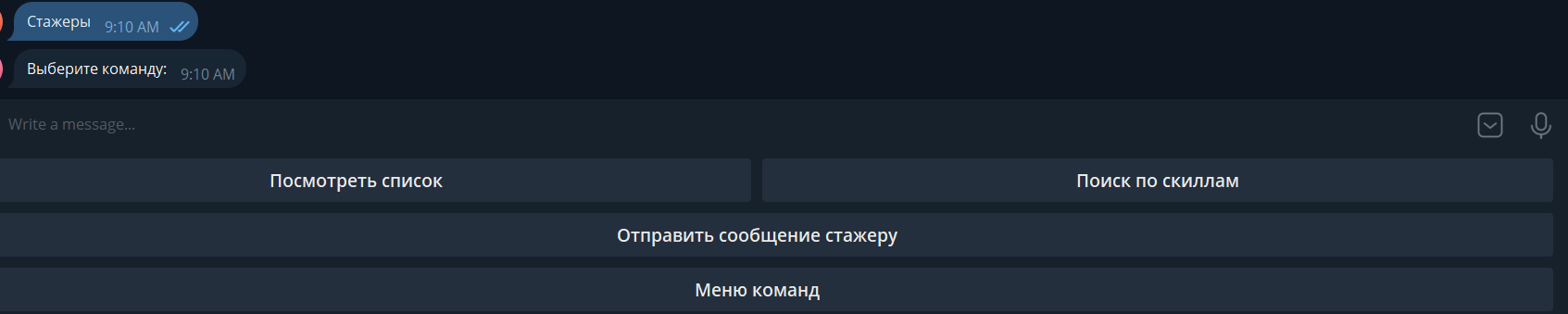


**3.1 Функционал администратора**

После заполнения информации о себе, будут доступны следующие действия:



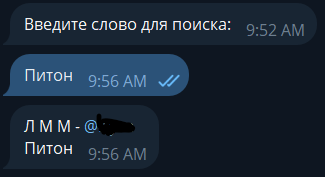
При переходе в стажеры предлагается следующий выбор команд:



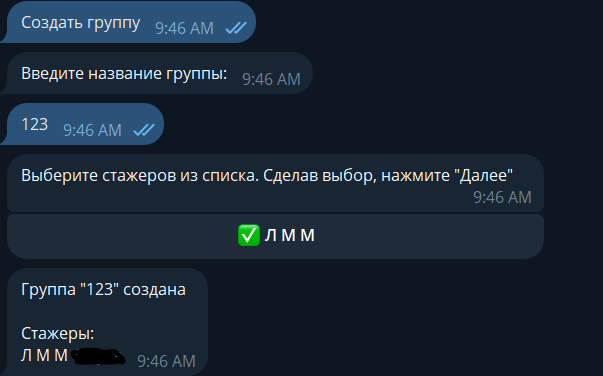
При регистрации новых пользователей администратору приходит уведомление в форме (Зарегистрирован новый пользователь:@username Никнейм):



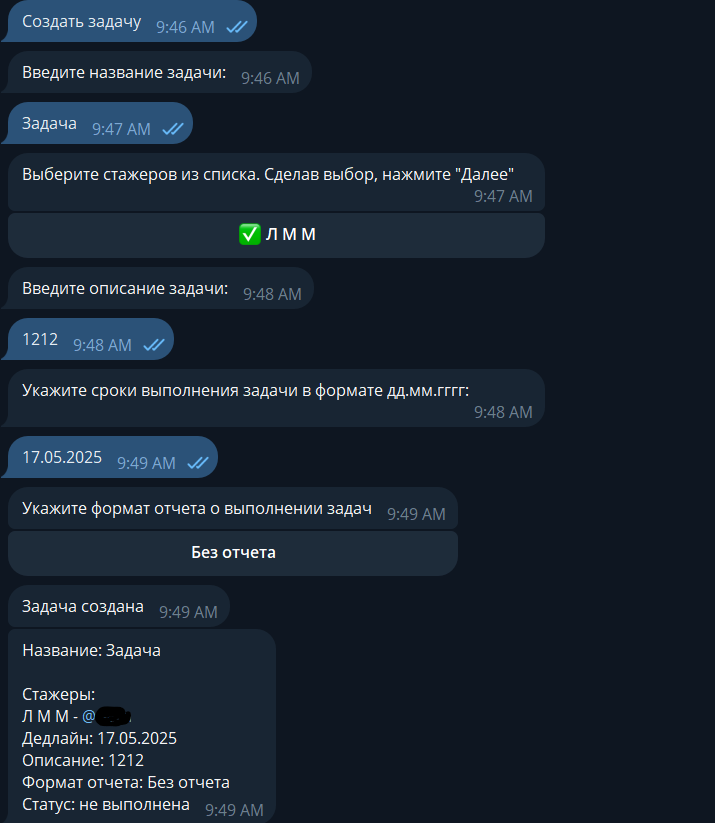
Помимо этого, админ может осуществлять поиск стажеров по скиллам. Поиск реализован на основе поиска с regex, который не чувствителен к целостности записи или регистру.



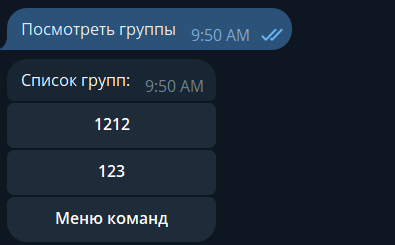
Админы также могут создавать группы из стажеров:



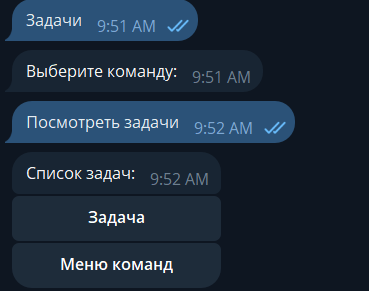
Администраторам доступна опция “Создание задачи”:



Работа команды “Просмотр групп”



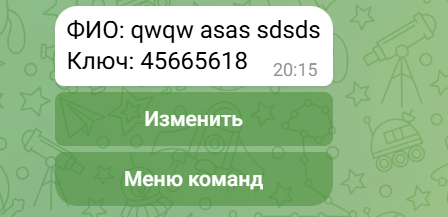
Просмотр всех доступных задач



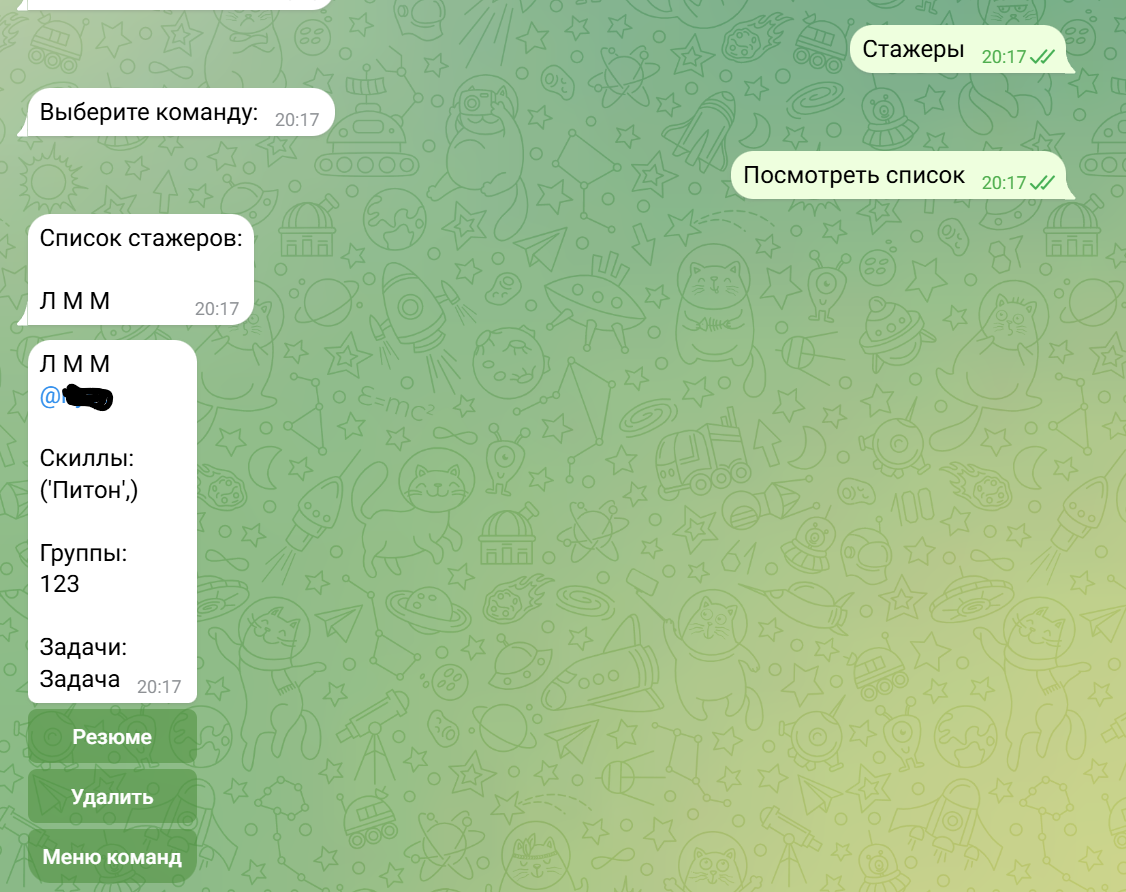
При выборе группы/задачи/стажера из выпадающего списка приходит отдельное сообщение с более подробной информацией.

У админа также есть возможность отправлять личные сообщения стажерам 

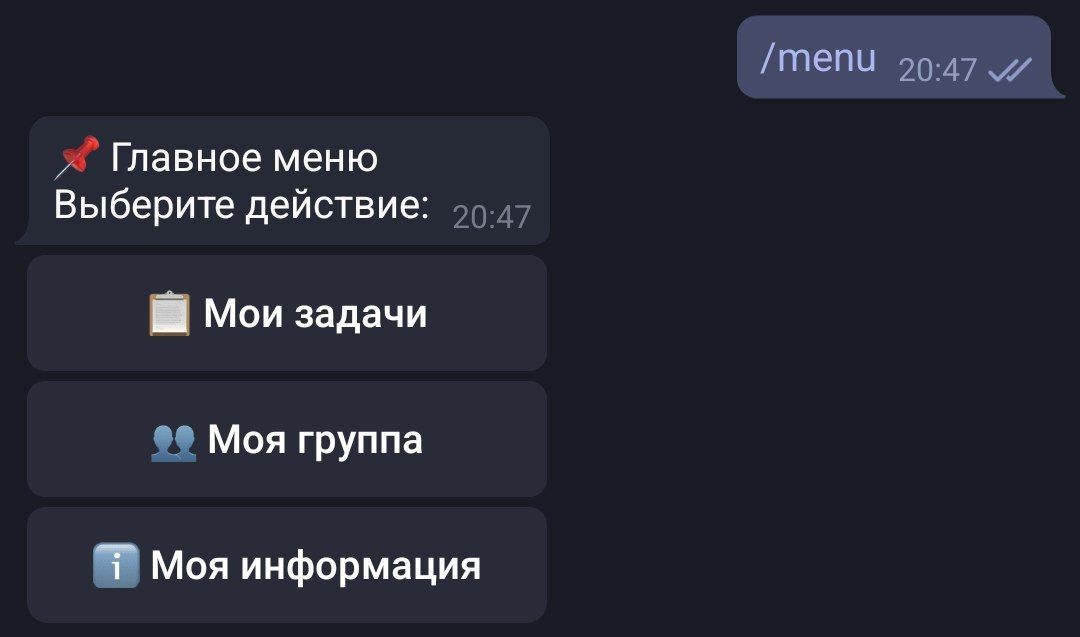
Ключ для стажера админ может получить в своем профиле. Профиль возможно изменить (фамилию, имя или отчество)



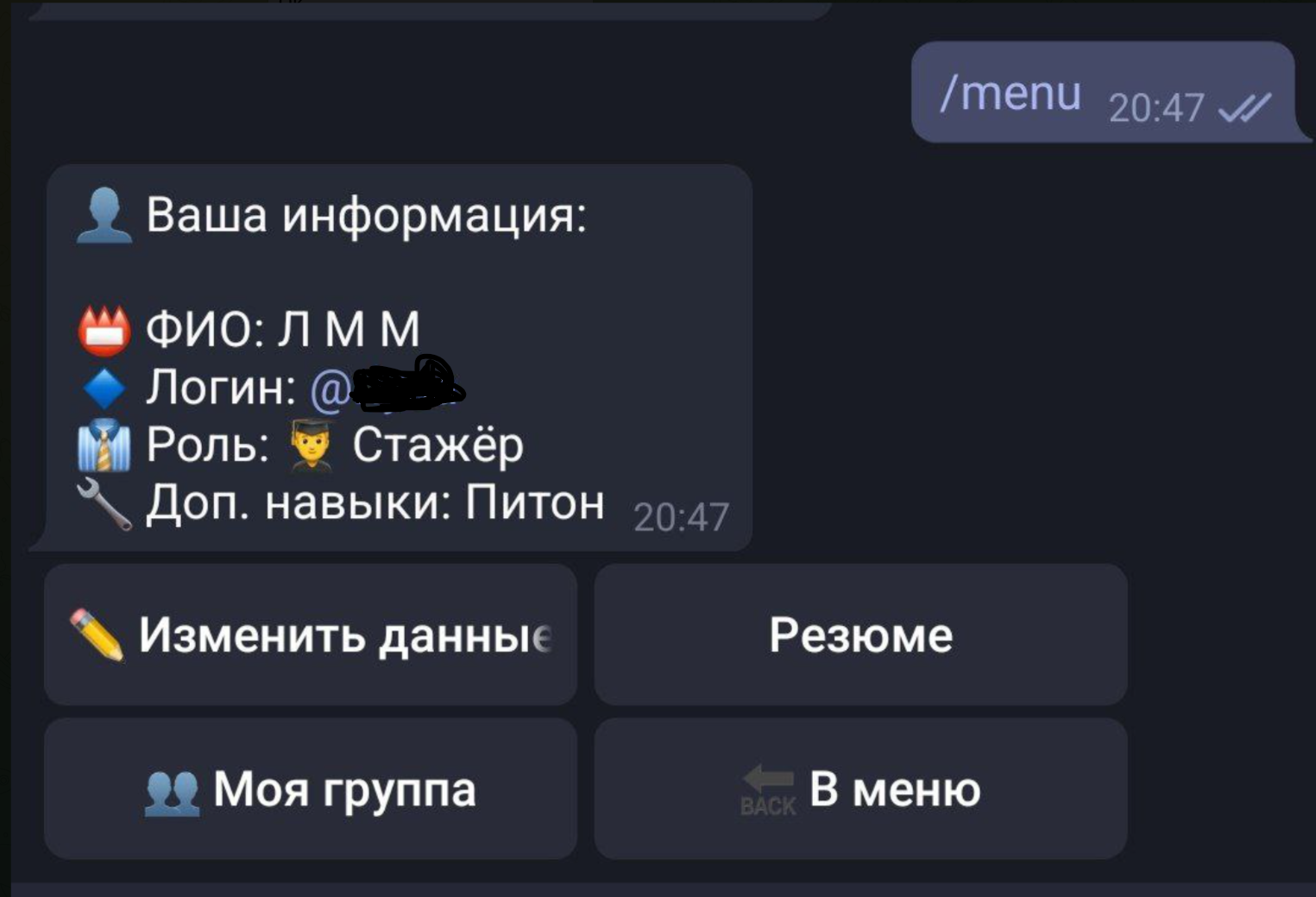
Далее продемонстрирован вывод информации о стажере, выбранном из выпадающего списка. Администратору доступно к просмотру резюме стажера, прикрепленное при регистрации в виде файла, администратор может удалить стажера.

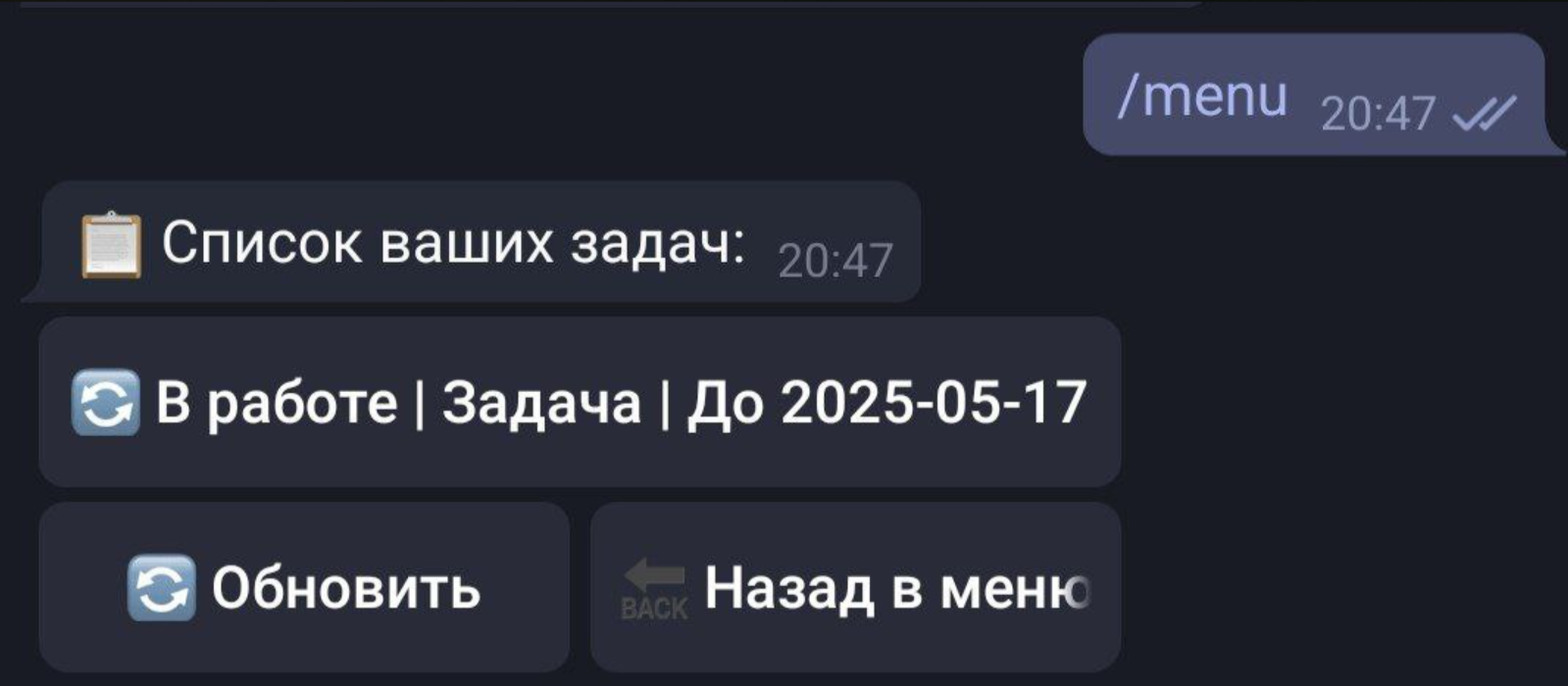


**3.2 Функционал стажера**

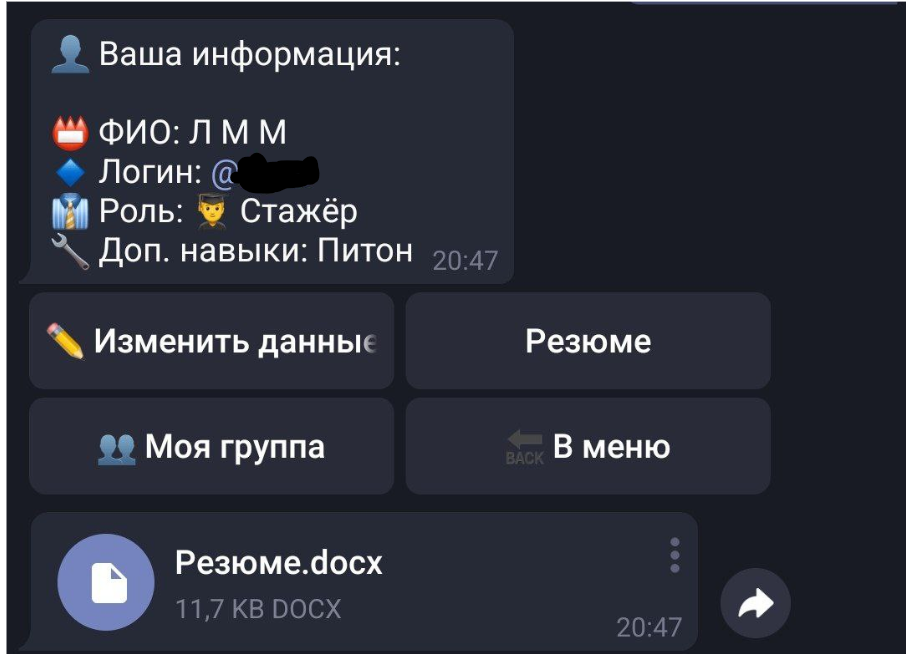
****

Если перейти в раздел “моя информация”, можно увидеть

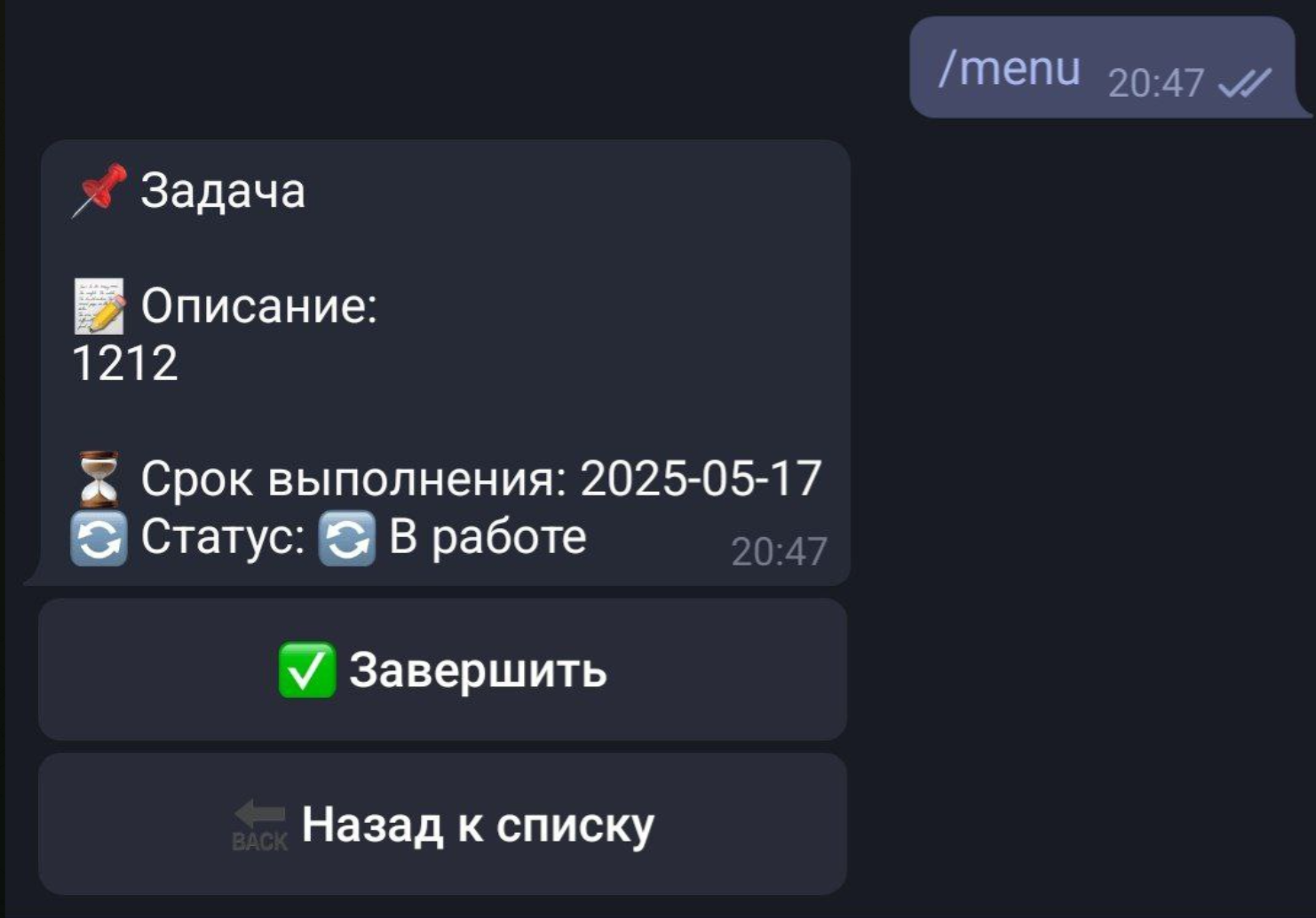
****

****

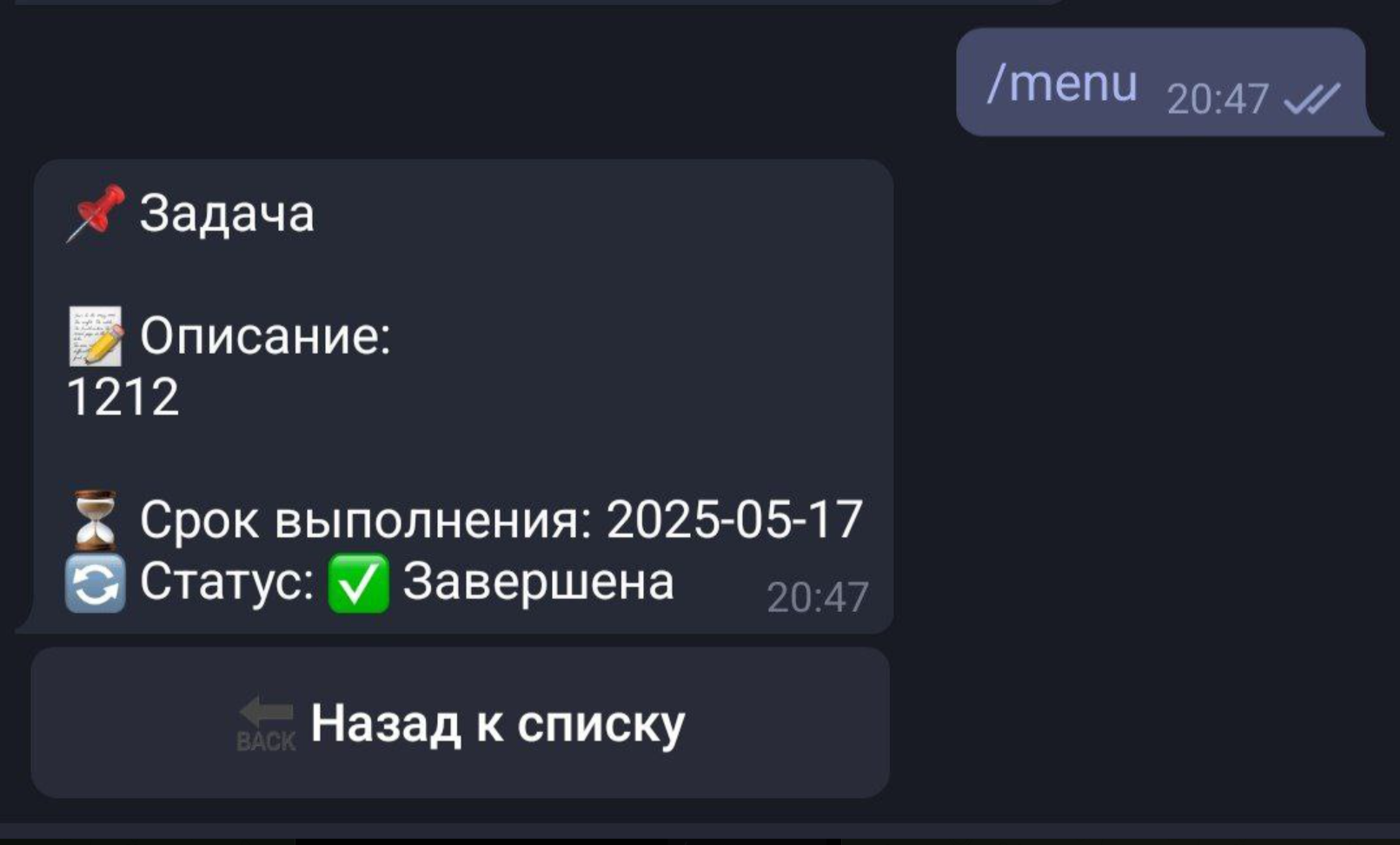
Отображение задач стажера и обновление статуса

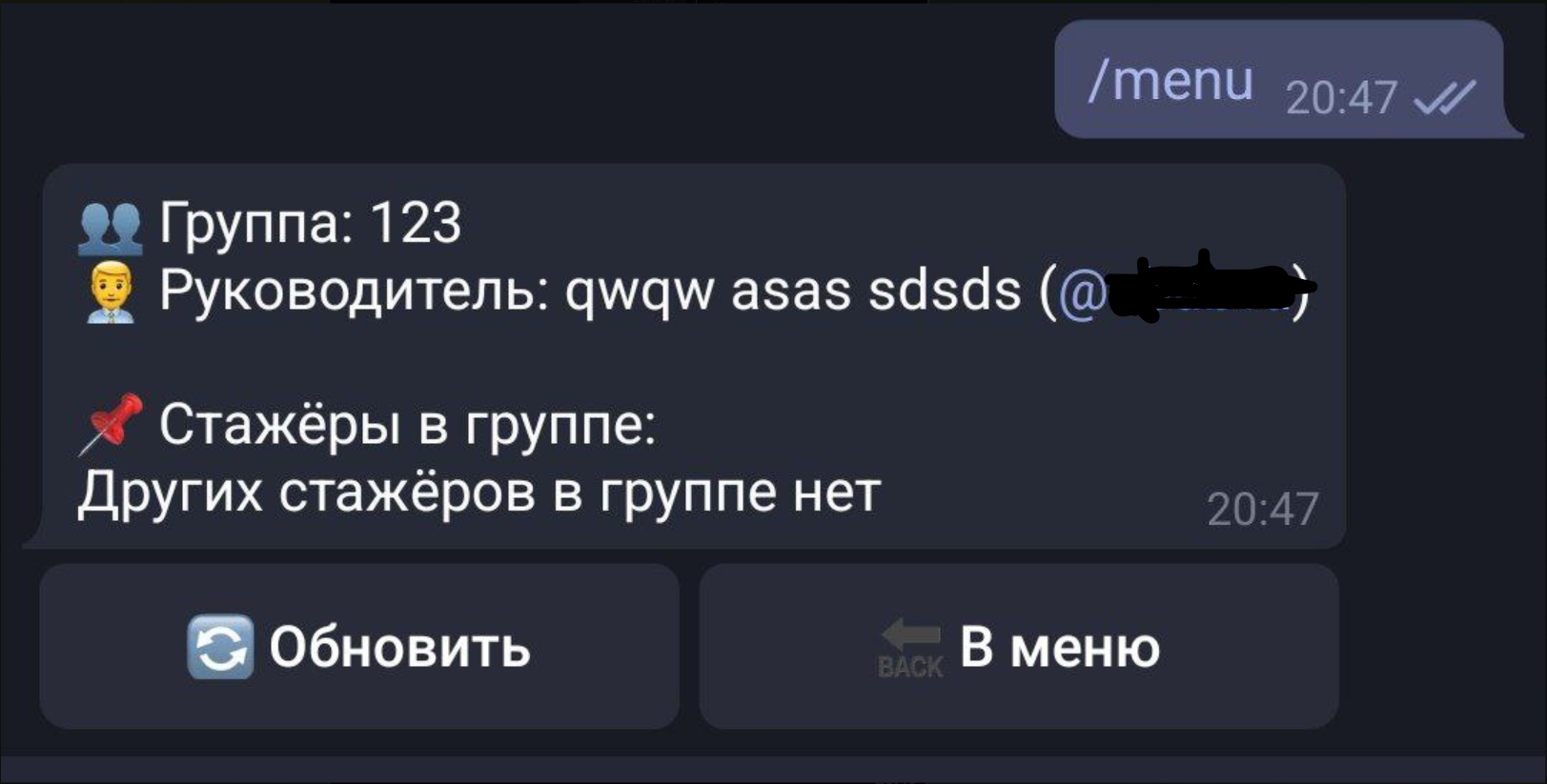


Возможность прикрепления резюме



Возможность завершения задачи



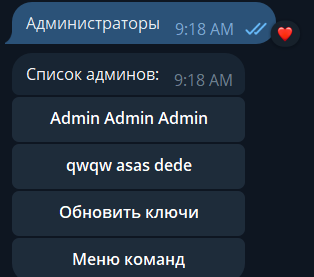


Отображение руководителя и стажеров по команде

**3.3 Функционал суперадмина**

Главным функционалом суперадмина является просмотр пользователей: стажеров и администраторов отдельно.

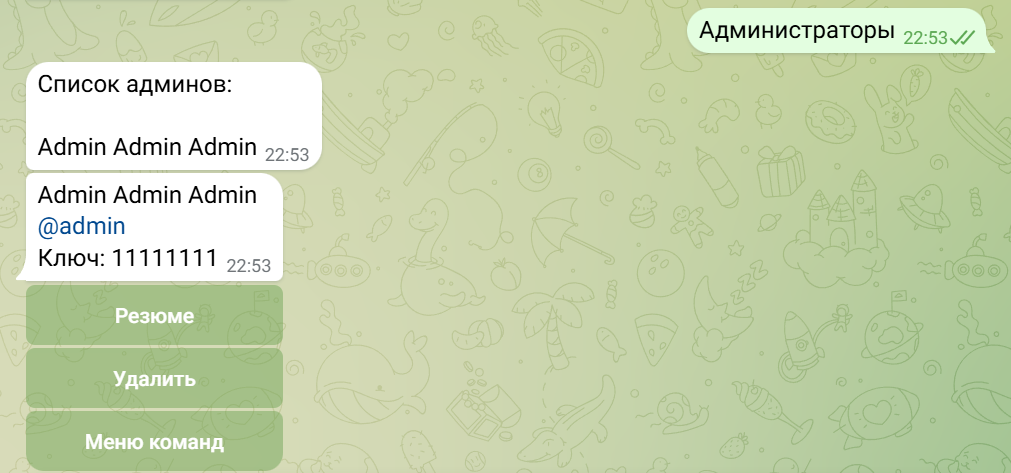




Обновление ключей администраторов происходит по нажатию соответствующей кнопки суперадмином.







Возможность удаления админов

4. Будущее развитие проекта

Проект можно развивать и масштабировать. Одной из идей является добавление подтверждение регистрации администратора главным администратором и стажера администратором, к которому он прикреплен. Также с целью повышения безопасности предлагается добавление функции. которая регулярно обновляла бы ключи администраторов, и сами администраторы по необходимости могли бы их обновлять.

Для упрощения выбора стажеров из списка при создании задачи администратору будет добавлена возможность выбрать группу, при нажатии на кнопку с названием группы будут подсвечены стажеры, состоящие в ней.

Кроме того, планируется переход на полноценную базу данных PSQL, которую планируется развернуть в отдельном контейнере и связать через docker-compose.yml.