Министерство науки и высшего образования Российской Федерации ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ОРЕНБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных технологий

Кафедра компьютерной безопасности и математического обеспечения информационных систем

**КУРСОВАЯ РАБОТА**

По языкам программирования

**Проектирование и разработка системы логирования событий в мессенджере Telegram**

ОГУ 10.05.01 3025 914 ОО

Руководитель

К.Р. Джукашев

Студент группы 23КБ(с)РЗПО-1

А.А. Коновалов

« » 20 г.

Оренбург 2025

Утверждаю

Заведующий кафедрой КБМОИС

                           И.В. Влацкая

«        »                         2024 г.

# ЗАДАНИЕ

**на выполнение курсовой работы**

студенту Коновалову Андрею Алексеевичу

специальности 10.05.01 Компьютерная безопасность

по дисциплине «Языки программирования»

1 Тема работы Проектирование и разработка системы логирования событий в мессенджере Telegram

2 Срок сдачи студентом проекта (работы) «    »   \_\_\_\_\_\_\_      2024 г.

3 Цель и задачи работы Разработать программный продукт для сохранения, упорядочивания и отображения истории событий аккаунта Telegram.

Задачи:

1. Разработать структуру хранения логов.
2. Разработать серверную часть системы логирования
3. Спроектировать и разработать интерфейсную часть с использованием технологии WebApp.
4. Реализовать авторизацию пользователя(ей) в аккаунт Telegram через веб-интерфейс.
5. Обеспечить защиту передаваемых данных посредством протокола https.

4 Исходные данные к проекту (работе): Учебная и научная литература содержащая:

- теоретические и практические сведения о платформе разработки .NET и языке C#;

- теоретические и практические сведение о языке разработки JavaScript;

- информацию об основах криптографии и о системах сквозного шифрования;

- документация Telegram Bot Api и Telegram Api

5 Перечень вопросов, подлежащих разработке:

1. Освоить на практике разработку с использованием языков C# и JavaScript.
2. Изучить методы Telegram Api, позволяющие осуществить логирование событий.
3. Изучить методы Telegram Bot Api, позволяющие встроить WebApp в бота Telegram.
4. Разработать и реализовать программный продукт, обеспечивающий сохранение, упорядочивания и отображения истории событий аккаунта Telegram.

Дата выдачи и получения задания

Руководитель «       »                      2024 г.                      К.Р. Джукашев

подпись инициалы фамилия

Студент «       »                      2024 г.                      А.А. Коновалов

подпись инициалы фамилия

# Аннотация

Курсовая работа посвящена созданию серверного приложения, цель которого записывать все события, происходящие на аккаунте Телеграм, а также разработку веб-интерфейса для просмотра сохранённых записей и управлением конфигурацией сохранения. Основные задачи включают разработку серверного приложения, которое отвечает за сохранение или «логирование» событий, Телеграм бота для управления серверным приложением прямо из мессенджера, а также веб-интерфейса встроенного в Телеграм бота.

Работа содержит … листов текста, …

# Содержание

# Введение

# Обозначения и сокращения

**Пользователь** – Телеграм аккаунт, с которого происходит взаимодействие с веб-приложением.

**Аккаунт** – Телеграм аккаунт, с которым происходит взаимодействие.

**Сервер** – серверная часть приложения или “бэкенд”.

**Бот** – Телеграм бот с помощью которого пользователь будет открывать веб-приложение.

**Логи** – журнал. Под журналом имеется в виду структура хранения записей в базе данных. Так же может означать и записи о работе приложения.

**Лог** – запись в журнале. Под записью имеется в виду строка в одной из таблиц базы данных. Так же может означать запись о работе приложения.

**Цель** – Чат аккаунта, чьи события будут сохраняться в логи.

# Техническое задание

Разработать программный продукт интегрированный в мессенджер Телеграм по средствам Telegram API и Telegram Bot Api.

* Создать Телеграм бота.

Управление должно быть реализовано с использованием веб-приложения или Web-App. Через это приложение можно будет ввести свои данные для входа в аккаунт Телеграм и отслеживать события, происходящие на этом аккаунте через это веб-приложение. Один пользователь должен иметь возможность добавить несколько аккаунтов.

* Разработать веб-сайт для мобильного формата.
* Создать кнопку в Телеграм боте для открытия сайта как веб-приложения внутри Телеграм.

Авторизация и аутентификация. Определять от какого пользователя пришёл запрос и проверять не самозванец ли этот пользователь.

* Создание сессии
* Проверка подписи

Сервер должен запускать сессии пользовательских аккаунтов и логировать (запоминать) события, пришедшие на них. Логи должны быть дополняемыми. *Пример: пришло сообщение – лог; сообщение было отредактировано – новый лог со ссылкой на старый (родительский) лог.*

* Разработать систему хранения событий.
* Написать приложение для логирования событий.
* Написать веб-сервер (бэкенд) для предоставления информации (логов) для веб-приложения.
* Реализовать поддержку изменения конфигурации «на горячую» через веб-приложение. В конфигурацию входят: аккаунты для отслеживания, изменение логируемых чатов. Должна быть возможность редактировать эти параметры без перезапуска приложения.

# Выбор методов, способов и средств разработки

Серверное приложение будет разработано на платформе ASP.NET8.0. Благодаря веб-серверу можно будет легко взаимодействовать с серверной частью приложения.

Для реализации получения событий с Телеграм аккаунта будет использоваться библиотека WTelegram для C#. Её код находится в открытом доступе и её использование бесплатно. Библиотека предоставляет удобный интерфейс для взаимодействия с Telegram API.

Для работы с базой данных будет использоваться Entity Framework, позволяющий удобно работать с данными из БД в представлении обычных объектов классов.

Взаимодействие пользователя с программой будет в веб-приложении, встроенном в Телеграм бота. Веб-приложение будет написано с использованием библиотеки React для JavaScript.

Веб-приложение будет развёрнуто на GitHub pages. Это бесплатный и простой способ развёртывания веб-страницы с SSL сертификатом.

Серверная инфраструктура (приложение и СУБД) будет упакована в docker-compose, что позволит быстро разворачивать его на любом сервере.

# Описание алгоритмов

## Использование технологии WebApp

Открывающееся в Телеграме веб-приложение это по сути своей обычный сайт. Отличие лишь в том, что подключена библиотека, которая выполняет функции авторизации и вызовом некоторого функционала Телеграм. Подробнее о функционале можно узнать в официальной документации Telegram mini apps, но сейчас нас интересует именно авторизация.

Авторизация через Телеграм позволит не реализовывать свою авторизацию, позволит не хранить хеши паролей и т.д. В таблице БД с пользователями мы будет просто хранить user\_id для каждого пользователя, который взаимодействовал с ботом.

Звучит довольно просто, но как этим пользоваться? В подключаемой библиотеке есть поле *initData*. Это поле храни строку с данными о пользователе, откуда и когда он открыл веб-приложение и хеш этой *initData*. Этот хеш используется для аутентификации данных. Благодаря нему, посторонний не сможет воспользоваться нашим API.

Для того, чтобы проверить пришедшую на бэкенд *initData*, нужно выполнить следующие шаги:

* Получить data\_check\_stirng путём сортировки всех полей полученных данных по ключу и записи их в виде:

<Ключ[1]>=<Значение[1]>

<Ключ[2]>=<Значение[2]>

…

<Ключ[n]>=<Значение[n]>

* Получить хеш для *secretKey* по алгоритму HMAC-SHA256, где ключ — это константная строка *“WebApp”*, а хешируемые данные – секретный API ключ Телеграм бота.
* Получить финальный хеш по алгоритму HMAC-SHA-256, где ключ это secretKey, а данные это dataCheckString.

Финальный хеш нужно сравнить с тем, что находится в полученной *initData*. При несовпадении этих хешей, отбрасываем запрос с этой *initData*.

## Авторизация и аутентификация

Авторизация будет осуществляться по заголовку userId в запросе к серверу, а аутентификация в свою очередь по подписи пакета.

Схема такая: при открытии веб-приложения, первым запросом отправляется auth/logIn с данными initData. Важно, что в заголовках должно присутствовать userId: -1, для обозначения того, что клиент ещё не авторизован. В ответе будет находиться строковое поле sessionCode и объектом me. Объект me – это объект информации о пользователе, который открыл веб-приложение. Там и будет тот самый userId, взятый в свою очередь из отправленной initData. Следующие запросы должны содержать в заголовках userId с полученными id.

Аутентификация будет осуществляться через подпись отправляемых данных. Подписывать данные нужно строкой sessionCode. В свою очередь sessionCode имеет время жизни, и сбросить это время жизни можно вызовом любого метода. Если в данный момент нет необходимости вызывать какой-то метод, а сбросить время жизни нужно, вызывается метод auth/ping.

## Добавление аккаунта

Чтобы подключиться к серверам телеграм, я использую библиотеку WTelegram. Для использования методов клиента, нужно создать объект класса Client. В библиотеке WTelegram есть несколько способов авторизации.

1. Можно подключиться к серверам телеграм и последовательно передавать в объект клиента данные (номер телефона, код верификации и т.д.), тогда в конструктор класса Client, нужно передать API\_ID, API\_HASH и путь до директории хранения сессий. API\_ID и API\_HASH являются идентификаторами приложения. Они создаются на сайте <https://my.telegram.org/auth> и привязываются к аккаунту. Эти два параметра необходимы для использования серверов телеграм.
2. Можно передать в конструктор функцию обратного вызова, которая будет возвращать запрашиваемые клиентом значения. Среди запрашиваемых значений могут быть API\_ID, API\_HASH, номер телефона или путь до директории хранения сессий. Удобно использовать этот способ для уже сохранённых сессий, вход в которые был осуществлён ранее.

Будем использовать оба способа в разных ситуациях. Первый для входа в новую сессию, а второй – для запуска уже существующих сессий.

## Назначение цели для отслеживания

Чтобы сервер понимал, какие чаты аккаунта нужно отслеживать, разработана система «целей». После добавления аккаунта, при клике на него открывается список диалогов аккаунта. Слева от аватара диалога, находится чекбокс. Если чекбокс выбран, то этот чат отслеживается. При изменении состояния чекбокса, отправляется запрос на сервер на метод *target/updateTarget* с данными о состоянии чекбокса, и в зависимости от состояния чекбокса, сервер помечает данный диалог как цель и начинает отслеживать, или же наоборот – перестаёт отслеживать.