**МИНИСТЕРСТВО ОБЩЕГО И ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**РОСТОВСКОЙ ОБЛАСТИ**

**ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ РОСТОВСКОЙ ОБЛАСТИ**

**РОСТОВСКИЙ-НА-ДОНУ КОЛЛЕДЖ СВЯЗИ И ИНФОРМАТИКИ**

**ОТЧЕТ**

**ПО УЧЕБНОЙ ПРАКТИКЕ**

**УП.01.01 «Разработка модулей программного обеспечения для компьютерных систем»**

Специальность **09.02.07 «Информационные системы и программирование»**

Студента 3-го курса, группы ИС-34

форма обучения: очная

Девицкий Георгий Сергеевич

ФИО

Место практики **Ростовский-на-Дону колледж связи и информатики**

Срок практики с «25» марта 2025 г. по «04» апреля 2025 г.

Руководитель практики Кошкина А. А.

подпись ФИО

Итоговая оценка по практике\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ростов-на-Дону

2025 г.

**МИНИСТЕРСТВО ОБЩЕГО И ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**РОСТОВСКОЙ ОБЛАСТИ**

**ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕРОСТОВСКОЙ ОБЛАСТИ**

**РОСТОВСКИЙ-НА-ДОНУ КОЛЛЕДЖ СВЯЗИ И ИНФОРМАТИКИ**

**Аттестационный лист**

**по учебной практике**

Студент Девицкий Георгий Сергеевич

обучающийся на **3** курсе в группе **ИС-34**

форма обучения: **очная**

по специальности: **09.02.07 «Информационные системы и программирование»**

прошел(ла) учебную практику

**УП.01.01 Учебная практика «Разработка модулей программного обеспечения для компьютерных систем»**

в объеме **78** часов с «**25**» марта **2025** г. по «0**4**» апреля **2025**г.

в ГБПОУ РО «РКСИ»

Виды и объем работ, выполненные студентом в период практики, формирующие у обучающегося умения и приобретение первоначального практического опыта (таблица 1).

Таблица 1 – Виды и объем работ

|  |  |  |  |
| --- | --- | --- | --- |
| Вид работ | Объём работ, час | Отметка о выполнении | |
| оценка | подпись руководителя |
| Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием | **8** |  |  |
| Разрабатывать программные модули в соответствии с техническим заданием | **32** |  |  |
| Выполнять отладку программных модулей с использованием специализированных программных средств | **16** |  |  |
| Выполнять тестирование программных модулей | **16** |  |  |
| Осуществлять рефакторинг и оптимизацию программного кода | **6** |  |  |

Качество выполнения работ в соответствии с технологией и предъявляемыми требованиями.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Итоговая оценка по практике \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики \_\_\_\_\_\_\_\_\_\_\_\_\_ /Кошкина А.А. /«\_\_\_» \_\_\_\_\_\_\_\_\_\_2025г.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Содержание  Введение 3  1.Проектирование программного продукта 5  1.1 Диаграмма связей 5  1.2 Диаграмма сценариев использования 6  1.3 Прототипирование и дизайн 7  1.4 Проектирование БД 10  2 Разработка программного продукта 11  2.1 Описание средств разработки 11  2.2 Разработка пользовательского интерфейса 12  2.3 Создание базы данных 17  2.4 Взаимодействие с базой данных 20  3 Тестирование программного продукта 21  3.1 Описание используемых видов тестирования 22  3.2 Тест-план 22  3.3 Тест-кейсы 24  Заключение 27  Приложение А 28  Приложение Б 29  Приложение В 33 | | | | | | | | | | |
|  |  |  |  |  | РКСИ.УП25.09.02.07.УП.04.01.4422.00ПЗ | | | | | |
|  |  |  |  |  |
| *изм* | *лист* | *№ докум.* | *Подпись* | *Дата* |
| *Разраб.* | | *Девицкий Г. С.* |  |  | УП 04 | *Лит* | | | *Лист* | *Листов* |
| *Провер.* | | *Кошкина А. А.* |  |  |  |  |  | *2* | *39* |
| *Реценз.* | |  |  |  | *ИС-34* | | | | |
| *Н. контр* | |  |  |  |
| *Утверд.* | |  |  |  |

Введение

В рамках данной работы был разработан Telegram-бот, предназначенный для помощи пользователям в отслеживании и улучшении качества сна. Современный ритм жизни, характеризующийся высоким уровнем стресса, повышенными умственными нагрузками и постоянным использованием цифровых устройств, зачастую приводит к различным нарушениям сна. В таких условиях создание специализированного бота, способного предоставлять персонализированные рекомендации и полезные материалы для нормализации сна, представляется особенно актуальным.

Актуальность данного проекта подтверждается статистическими данными, согласно которым значительная часть взрослого населения сталкивается с теми или иными проблемами сна. Многие люди испытывают трудности с засыпанием, страдают от бессонницы или не высыпаются, что в конечном итоге негативно сказывается на их продуктивности, настроении и общем состоянии здоровья. При этом далеко не все осознают важность полноценного сна или знают эффективные способы улучшения его качества. Разработанный бот призван решить эти проблемы, предлагая пользователям удобный инструмент для мониторинга сна и доступа к проверенным методикам его нормализации.

Основной целью разработки стало создание многофункционального, но при этом простого в использовании инструмента, который позволит пользователям лучше понимать особенности своего сна и работать над его улучшением. Бот предоставляет возможность регистрации с указанием персональных данных, что позволяет давать более точные рекомендации. После регистрации пользователь получает доступ к нескольким ключевым функциям: библиотеке материалов для улучшения сна (включающей аудиосказки, расслабляющую музыку), информационному разделу с описанием фаз и видов сна, а также умному будильнику. Особое внимание уделено простоте интерфейса и удобству взаимодействия, чтобы ботом могли пользоваться люди с разным уровнем технической грамотности.

Целевая аудитория бота достаточно широка и включает в себя как людей, уже столкнувшихся с проблемами сна, так и тех, кто хочет их предотвратить. Это могут быть студенты, испытывающие стресс во время сессии, офисные работники с ненормированным графиком, молодые родители, а также все, кто заботится о качестве своего отдыха и продуктивности. Для каждой из этих категорий бот предлагает соответствующие их потребностям решения.

Функционал бота включает несколько основных модулей. Модуль регистрации собирает базовую информацию о пользователе (пол, возраст, вес) и выясняет наличие конкретных проблем со сном. Раздел с медиаматериалами содержит подборку контента, способствующего быстрому

засыпанию и глубокому сну, включая сказки для взрослых, звуки. Информационный раздел знакомит пользователей с теорией сна, объясняя важность различных его фаз и предлагая советы по гигиене сна. Также реализована возможность связи с администратором для получения индивидуальных консультаций.

Особое внимание при разработке было уделено этапу тестирования, в ходе которого проверялась корректность работы всех функций, удобство навигации и устойчивость системы к нагрузкам. Тестирование включало как автоматизированные проверки основных сценариев использования, так и сбор обратной связи от реальных пользователей. Это позволило выявить и устранить потенциальные проблемы до запуска бота, обеспечив стабильную работу и положительный пользовательский опыт. В перспективе планируется расширение функционала за счет добавления дневника сна и системы аналитики, что сделает бот еще более полезным инструментом для тех, кто заботится о качестве своего отдыха.

1 Проектирование программного продукта

Проектирование программного обеспечения – это процесс создания программных систем, который включает анализ требований, разработку архитектуры, детальное проектирование, программирование, тестирование и документацию. Он направлен на создание эффективных и надежных решений, соответствующих потребностям пользователей. Процесс требует знаний в области программирования, системного анализа и управления проектами.

* 1. Диаграмма связей

Диаграмма связей – это графическое представление взаимосвязей между различными элементами системы, такими как объекты, классы или компоненты. Она используется для визуализации и анализа отношений, взаимодействий и зависимостей между этими элементами. Диаграммы связей помогают понять структуру системы, выявить ключевые связи и упростить процесс проектирования и разработки.

На рисунке 1 представлена блок-схема работы Telegram-бота. Взаимодействие пользователя с системой начинается с начального экрана, после чего следует этап авторизации для обеспечения безопасного доступа. После успешного входа пользователь получает доступ к основным функциональным разделам: «Видео для сна», «Типы сна», «Будильник» и «Связь с администратором». Диаграмма наглядно демонстрирует логическую структуру и основные точки взаимодействия пользователя с ботом.



Рисунок 1 – Диаграмма связей для Телеграмм-бота

1.2 Диаграмма сценариев использования

Диаграмма сценариев – это визуальное представление последовательности действий и взаимодействий пользователей с системой для достижения определенной цели. Она иллюстрирует, как пользователи (или другие системы) взаимодействуют с функциональностью программного обеспечения в различных сценариях использования.

На рисунке 2 можно увидеть диаграмму сценария использования Телеграмм-бота. Диаграмма показывает варианты взаимодействия пользователей с системой. «Гость» может начать с «Начать регистрацию» и пройти процесс «Авторизации» и получив доступ к «Услугам, может использовать функции «Видео для сна», «Типы сна», «Будильник».

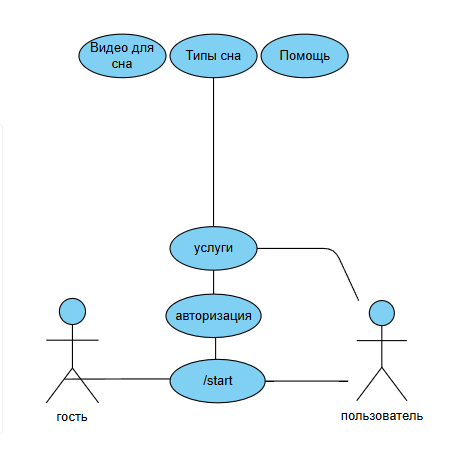


Рисунок 2 – Диаграмма сценариев использования

1.3 Прототипирование и дизайн

Дизайн экранов приложения представлен на рисунке 3.

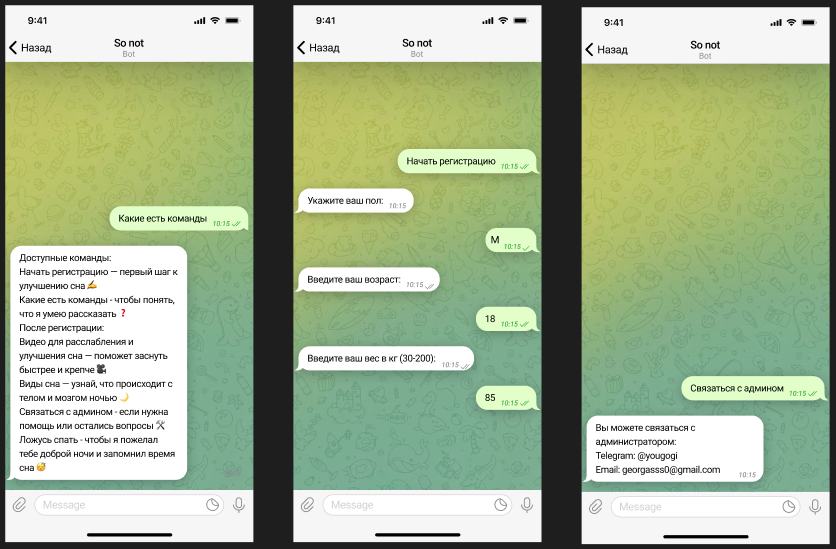


Рисунок 3 - Дизайн главных экранов

Рисунки 4 и 5 показывают, как выглядят экраны связи с администратором и настройки профиля.

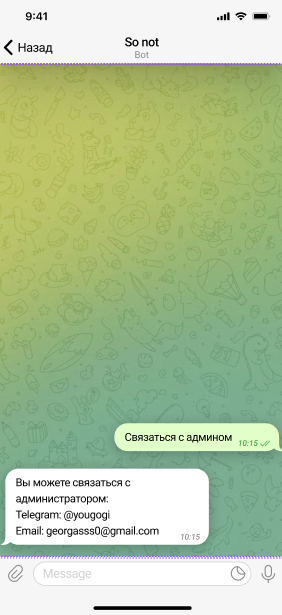


Рисунок 4 - Макет связи с администратором



Рисунок 5 - Макет экрана настройки профиля

На рисунке 6 представлен QR-код, который можно отсканировать, чтобы получить доступ к полному дизайну.



Рисунок 6 - QR-код, для доступа к дизайну

1.4 Проектирование БД

База данных разработана для Telegram-бота, помогающего пользователям улучшить качество сна. Бот предоставляет персонализированные рекомендации, медиатеку расслабляющего контента и инструменты для отслеживания сна. БД построена на SQLite 3 и включает 3 таблицы, связанные между собой.

На рисунке 7 изображена таблица users, которая хранит информацию о пользователях с полями: id, tg\_id, gender, height, weight, diseases и is\_registered, где tg\_id – это идентификатор пользователя в Telegram gender, пол – height, рост – weight, вес и is\_registered – статус регистрации, diseases – проблемы со сном.

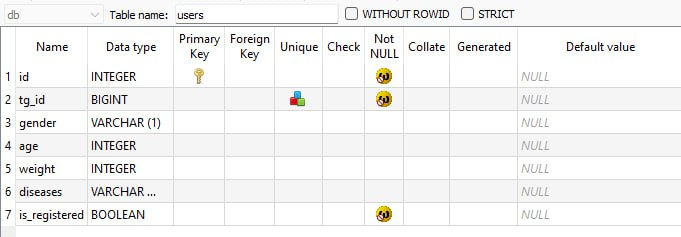


Рисунок 7 – Таблица о пользователях

На рисунке 8 изображена таблица, которая содержит медиа для сна: id, name, description, video\_link и category\_id, где id – это первичный ключ, name – название, description – описание до 100 символов, video\_link – ссылка на видео, а category\_id – внешний ключ связывающий с таблицей sleep\_categories.id.

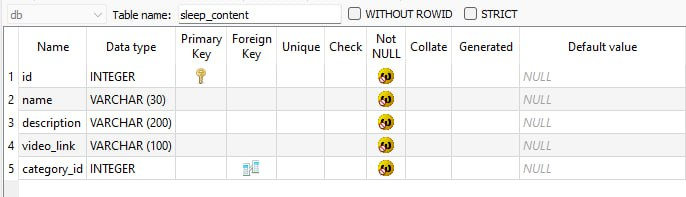


Рисунок 8 – Таблица с медиа

На рисунке 9 показана таблица sleep\_categories, которая включает id и name, где id – первичный ключ, а name – название, например, сказки, 30 символов.

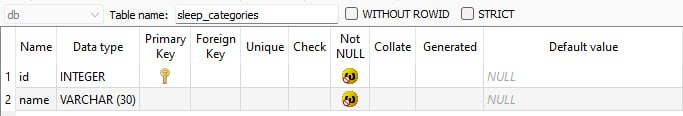


Рисунок 9 – Таблица с контентом для сна

2 Разработка программного продукта

В данном разделе представлены инструменты и технологии, использованные в процессе разработки проекта.

2.1 Описание средств разработки

Проект разработан на Python 3.9+ с использованием современных технологий и инструментов. Основой бота стал асинхронный фреймворк Aiogram 3.x, который предоставил все необходимые инструменты для создания полнофункционального Telegram-бота. Выбор Aiogram обусловлен его высокой производительностью, удобной работой с состояниями (FSM) и простотой интеграции с Telegram API. Для хранения данных пользователей и медиатеки была использована SQLite 3 - легковесная и надежная реляционная база данных, идеально подходящая для проектов такого масштаба.

Разработка велась в профессиональной среде Visual Studio Code,

которая значительно ускорила процесс написания кода благодаря интеллектуальному автодополнению, встроенному отладчику и удобным инструментам для работы с виртуальными окружениями. Для проектирования структуры базы данных и выполнения SQL-запросов использовался DBeaver - мощный универсальный клиент для работы с различными СУБД. На этапе проектирования интерфейса бота активно применялись инструменты визуализации: Figma для создания прототипов интерфейса и дизайна элементов управления, а также Draw.io для построения диаграмм взаимодействия и схемы базы данных.

2.2 Разработка пользовательского интерфейса

Взаимодействие с So not реализовано через систему текстовых команд, обеспечивающую простое и интуитивно понятное управление. Основной функционал бота доступен через ключевые команды, показанные на рисунке 7:

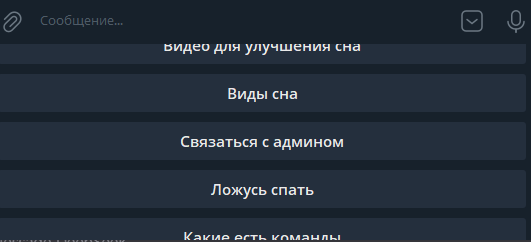


Рисунок 7 - Основные команды

На рисунке 8 представлена команда /start, которая запускает диалог с ботом. Бот предлагает зарегистрироваться, нажав на «Начать регистрацию» для получения списка доступных команд и инструкций по началу работы. Это является отправной точкой для новых пользователей, желающих воспользоваться функционалом So not

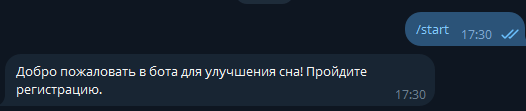


Рисунок 8 - Команда /start

На рисунке 9 показана команда «Начать регистрацию», после нажатия которой бот начинает процесс сбора пользовательских данных, частично необходимых для рекомендации времени для сна. Пользователь вводит данные, отвечая на вопросы бота. Бот проверяет, чтобы данные были введены правильно (например, вес и рост должны быть числами).

Если при вводе данных возникнут ошибки (например, пользователь введет текст вместо числа), бот сообщит об этом и предложит исправить ошибку.

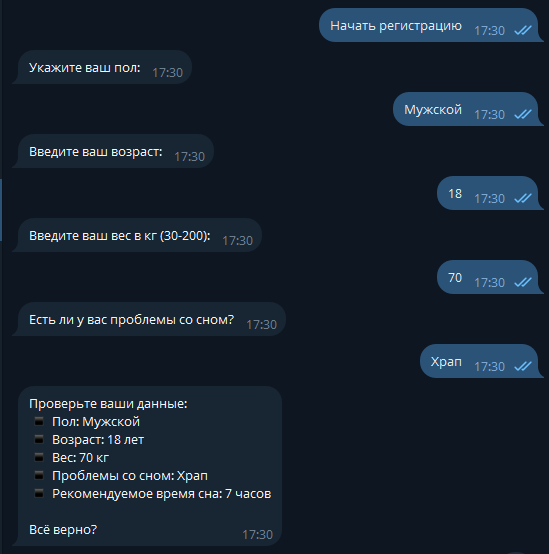
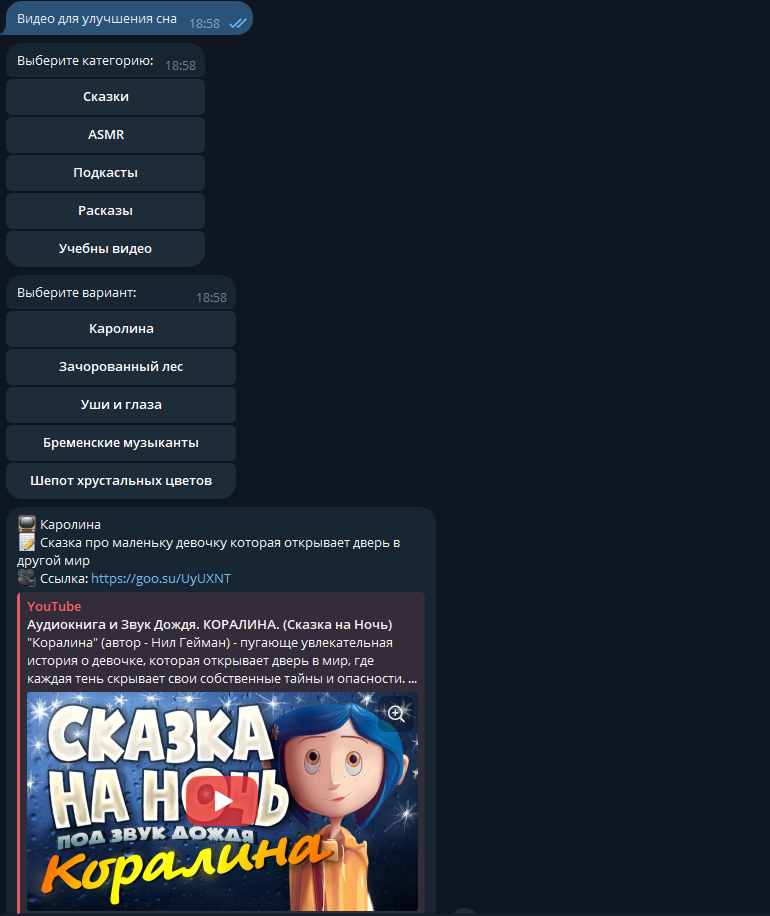


Рисунок 9 - Команда «Начать регистрацию»

На Рисунке 10 представлена команда «Видео для улучшения сна», которая предоставляет пользователю доступ к медиатеке расслабляющего контента. При нажатии на соответствующую кнопку бот отправляет список доступных категорий. Каждая категория содержит подготовленную коллекцию материалов.

После выбора категории пользователь получает список доступных материалов, где каждый элемент представлен отдельной кнопкой. При нажатии на конкретный вариант бот отправляет видео контент с описанием материала.



Рисунки 10 - Команда «Настройка тренировок»

Благодаря команде «Виды сна», пользователь может детально изучить различные режимы сна, их особенности, преимущества и недостатки.

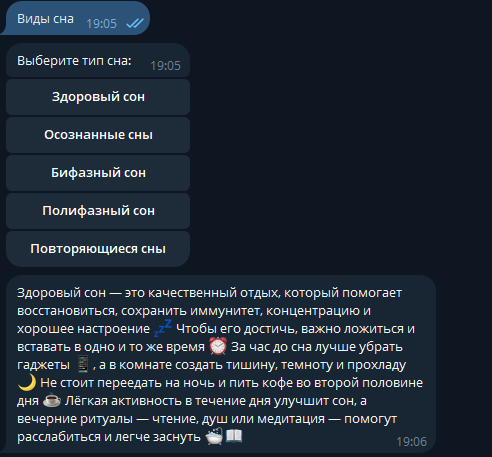


Рисунок 11 - Команда «Виды сна»

На рисунке 12 показана команда  «Связаться с админом», которая предоставляет пользователю способ связаться с администратором So not через Telegram. После ввода команды бот отправляет пользователю username администратора в Telegram и почту. Пользователь может использовать этот username или почту, чтобы написать администратору напрямую и задать вопросы о работе бота, сообщить о проблемах или оставить предложения по улучшению.

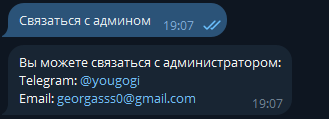


Рисунок 12 - Команда «Связаться с админом»

2.3 Создание БД

База данных (БД) – это ключевой компонент Telegram-бота для управления тренировками. Она хранит информацию о пользователях, упражнениях. В проекте используется MySQL – реляционная СУБД, обеспечивающая надежное хранение данных.

На рисунке 1 показана настройка подключения к базе данных, где настраиваем подключение к SQLite базе данных, используя асинхронный движок SQLAlchemy. База данных будет хранится в файле db.sqlite3, создание фабрик синхронных сессий для работы с БД.

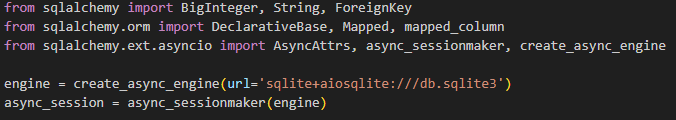


Рисунок 1 – Настройка подключения к базе данных

На рисунке 2 показан базовый класс для моделей, где наследуется от AsyncAttrs для асинхронной работы с атрибутами и от DeclarativeBase для декларативного стиля ORM. Все модели таблиц будут наследоваться от этого класса.



Рисунок 2 – Базовый класс для моделей

На рисунке 3 показана модель пользователя user, где id – первичный ключ (автоинкремент), tg\_id – уникальный ID пользователя в Telegram (BigInteger), gender – пол пользователя (1 символ, может быть NULL), height – рост в см (может быть NULL), weight – вес в кг (может быть NULL), is\_registered – флаг завершения регистрации (по умолчанию False), diseases для хранения информации о нарушениях сна пользователя.

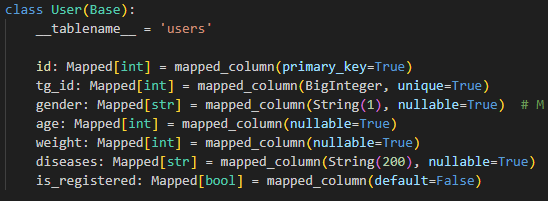


Рисунок 3 – Модель пользователя

На рисунке 4 показана категория сна SleepCategory, где id – первичный ключ, name – название категории (максимум 30 символов).

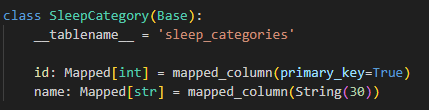


Рисунок 4 – Категория сна

На рисунке 5 показана модель контента, где id – первичный ключ, name – название категории (максимум 30 символов), description – описание (100 символов), video\_link – ссылка на видео (100 символов), category\_id –внешний ключ к таблице категорий.

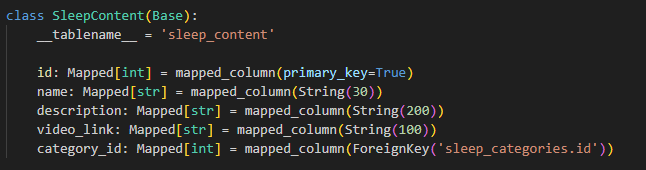


Рисунок 5 – Модель контента

На рисунке 6 показана инициализация базы данных, где функция async\_main создает все таблицы при первом запуске и использует асинхронный контекст, вызывает синхронный метод create\_all через run\_sync.

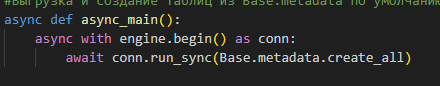


Рисунок 6 – Инициализация базы данных

На рисунке 7 показана инициализация сессии и импорты. Импортируется фабрика асинхронных сессий (async\_session), модели данных (User, SleepCategory, SleepContent) и функция select из SQLAlchemy для построения запросов. Это основа для всех последующих операций с БД.

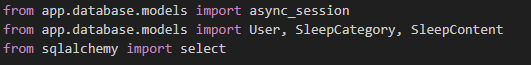


Рисунок 7 – Инициализация сессии и импорты

На рисунке 8 изображено получение пользователя. Создается синхронная сессия через async with, выполняется запрос на поиск пользователя по telegram ID (tg\_id), session.scalar() возвращает один результат или None, типизированный возврат User помогает при разработке.

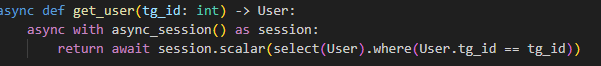


Рисунок 8 – Получение пользователя

На рисунке 9 изображено создание или обновление пользователя, где проверяется существование пользователя, либо обновляет существующие поля, либо создает новую запись. session.add() добавляет новый объект в сессию, session.commit() фиксирует изменения атомарно. Все поля пользователя обновляются единообразно.

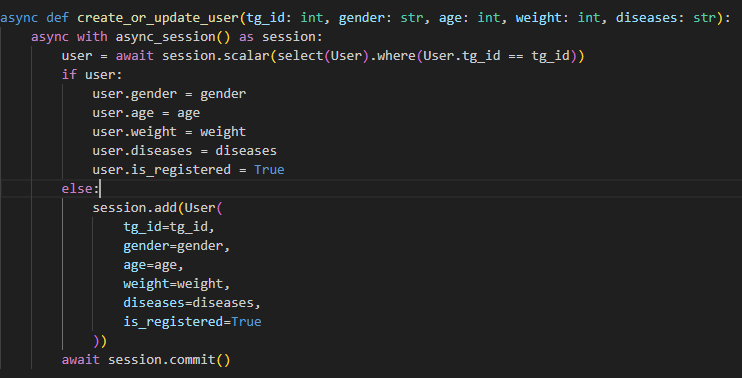


Рисунок 9 – Создание или обновление пользователя

На рисунке 10 показано получение категорий. Простейший запрос без условий, scalars() возвращает итератор объектов category, используется для отображения меню выбора категорий.

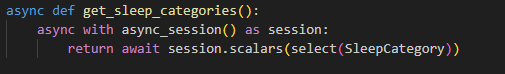


Рисунок 10 – Получение категорий.

На рисунке 11 изображено получение контента сна по категориям. Фильтрация через where по внешнему ключу category. Возвращает все упражнения выбранной категории.

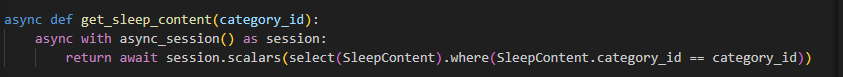


Рисунок 11 – Получение контента сна по категориям

На рисунке 12 изображено получение конкретного контента. Точечный запрос по первичному ключу id, возвращает полные данные упражнения, включая видео-ссылку.

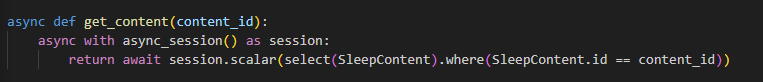


Рисунок 12 – Получение конкретного контента

2.4 Взаимодействие с БД

На рисунке 1 изображен фрагмент кода из файла keyboards.py содержит тип клавиатур, которая играет ключевую роль в организации взаимодействия пользователя с базой данных. Главное меню (main) реализовано через ReplyKeyboardMarkup и включает четыре основные кнопки: "Видео для улучшения сна", "Виды сна", "Связаться с админом", "Какие есть команды" и "Ложусь спать".

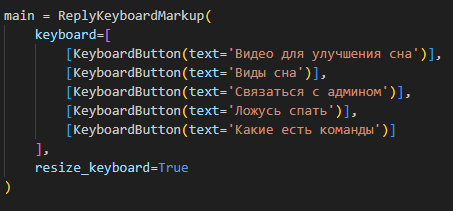


Рисунок 1 – Клавиатура

На рисунке 2 код обрабатывает ввод веса при регистрации: проверяет, что введено число от 30 до 200 кг. Если проверка пройдена, сохраняет вес и переходит к вопросу о проблемах со сном, выводя клавиатуру для выбора. При ошибке просит ввести вес заново. Стандартный шаблон для многоэтапной регистрации в aiogram.

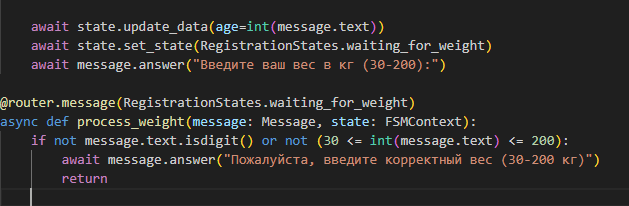


Рисунок 2 – Обработка веса

3 Тестирование ПП

Тестирование - это процесс оценки качества программного обеспечения путем выявления ошибок, дефектов и несоответствий требованиям.

3.1 Описание видов тестов

Программный продукт был протестирован вручную, без использования автоматизации. Тестировщик выполнял сценарии, как обычный пользователь, и внимательно следил за работой системы, фиксируя все ошибки. Этот подход дал возможность лучше понять потребности пользователей, вовремя реагировать на нестандартные ситуации и заметить те детали, которые сложно выявить автоматизированными средствами.

Таблица 1 – Тест-план.

|  |  |  |  |
| --- | --- | --- | --- |
| Тест-кейс, № | Название | Описание | Метод  тестирования |
| 1. | Нажатие на кнопку /start | Пользователь нажимает кнопку Start для начала работы с ботом. | Ручное |
| 1. | Нажатие на кнопку «Начать регистрацию» | Пользователь нажимает кнопку Начать регистрацию для ввода данных. | Ручное |
| 3. | **Ввод данных о пользователе** | Пользователь вводит пол, рост, вес (в пределах допустимых значений). | Ручное |
| 4. | Завершение регистрации | Проверка отправки username администратора. | Ручное |
| 5. | Проверка работы кнопки «Начать регистрацию» | Проверка работы кнопки «Начать регистрацию» для зарегистрированного пользователя. | Ручное |

3.3 Тест-кейсы

В таблице 2 представлен тест-кейс, который проверяет, что при нажатии на кнопку Start бот правильно реагирует и отправляет пользователю сообщение с кнопкой для начала регистрации.

Таблица 2 – Тест-кейс №1 «Нажатие на кнопку /start».

|  |  |  |
| --- | --- | --- |
| Название | Нажатие на кнопку /start | |
| Тип тестирования | Ручное | |
| Действие | Ожидаемый результат | - Пройден  - Провален |
| Предусловие | Аккаунт не зарегистрирован | |
| Шаги теста |  | |
| Запуск программы с помощью /start | Успешный запуск программы. Вывод программы для запуска с кнопкой для дальнейшей работы | Пройден |

В таблице 3 представлен тест-кейс, который проверяет, что при нажатии на кнопку «Начать регистрацию» бот правильно реагирует и отображает запрос на ввод данных для регистрации пользователя.

Таблица 3 – Тест-кейс №2 «Нажатие на кнопку «Начать регистрацию»».

|  |  |  |
| --- | --- | --- |
| Название | Нажатие на кнопку «Начать регистрацию» | |
| Тип тестирования | Функциональное | |
| Действие | Ожидаемый результат | - Пройден  - Провален |
| Предусловие | Пользователь нажал на кнопку Start и бот отобразил сообщение с кнопкой Начать регистрацию. | |
| Шаги теста |  | |
| Пользователь нажимает на кнопку «Начать регистрацию» | Бот должен запросить у пользователя следующие данные: пол, рост, вес в рамках допустимых значений. | Пройден |

В таблице 4 представлен тест-кейс в котором тестируется функционал добавления резюме от нового пользователя.

Таблица 4 – Тест-кейс №3 «**Ввод данных о пользователе**».

|  |  |  |
| --- | --- | --- |
| Название | **Ввод данных о пользователе** | |
| Тип тестирования | Ручное | |
| Действие | Ожидаемый результат | - Пройден  - Провален  - Заблокирован |
| Предусловие | Пользователь авторизован | |
| Шаги теста |  | |
| Пользователь вводит свои данные | Бот должен корректно принять все введенные данные, проверить их на соответствие допустимым значениям и отобразить сообщение о завершении регистрации. | Пройден |
| Пользователь вводит некорректные данные | Бот должен отклонить неверные данные (например, рост 30000 или вес 550) и запросить корректные данные, отправив сообщение о том, что данные введены неправильно. | Пройден |

В таблице 5 представлен тест-кейс, который должен завершить процесс регистрации и уведомить пользователя о том, что регистрация прошла успешно.

Таблица 5 – Тест-кейс №4 «Завершение регистрации».

|  |  |  |
| --- | --- | --- |
| Название | Завершение регистрации | |
| Тип тестирования | Ручное | |
| Действие | Ожидаемый результат | - Пройден  - Провален  - Заблокирован |
| Предусловие | Пользователь ввел все данные | |
| Шаги теста |  | |
| Пользователь завершает ввод данных | Бот должен подтвердить успешное завершение регистрации и уведомить пользователя о том, что регистрация прошла успешно | Пройден |

В таблице 6 представлен тест-кейс в котором тестируется проверка кнопки «Начать регистрацию» для уже зарегистрированного пользователя.

Таблица 6 – Тест-кейс №5 «Проверка работы кнопки «Начать регистрацию»».

|  |  |  |
| --- | --- | --- |
| Название | Проверка работы кнопки «Начать регистрацию» | |
| Тип тестирования | Ручное | |
| Действие | Ожидаемый результат | - Пройден  - Провален  - Заблокирован |
| Предусловие | Пользователь зарегистрирован | |
| Шаги теста |  | |
| Открыть бота и ввести «Начать регистрацию» и убедиться что регистрация не начинается заново. | Бот должен отобразить, что пользователь уже зарегистрирован. | Провален |

Заключение

Разработанный с использованием современных технологий, включая Python (aiogram) и SQLite, бот представляет собой удобный инструмент, предоставляющий персонализированные рекомендации по нормализации сна. So not позволяет пользователям указать свои параметры (возраст, пол, вес и имеющиеся проблемы со сном) для получения индивидуально подобранных программ восстановления.

Пользователи получают доступ к обширной базе материалов, включающей аудиосказки, медитации, ASMR-контент и образовательные статьи о природе сна. Каждая рекомендация адаптируется под особенности конкретного пользователя, обеспечивая максимальную эффективность.

В ходе тщательного тестирования подтверждена стабильная работа всех функций бота, включая точность рекомендаций и удобство взаимодействия. Бот гарантирует надежный доступ к персонализированным программам улучшения сна и стабильную работу платформы.

Приложение А

SleepHelperBot - это удобный телеграм-бот, созданный для помощи пользователям в улучшении качества сна. Он предлагает персонализированные рекомендации по нормализации сна на основе индивидуальных параметров пользователя. Пользователь получает доступ к обширной медиатеке, включающей аудиосказки, медитации, ASMR-контент и образовательные материалы о природе сна. Все рекомендации адаптируются под конкретные особенности и потребности пользователя. Бот помогает подобрать оптимальные решения для борьбы с бессонницей, частыми пробуждениями и другими нарушениями сна, делая процесс засыпания более комфортным и эффективным. Простая и интуитивно понятная система взаимодействия позволяет легко получать персонализированные программы для улучшения качества сна.

Листинг 1 – main.py

import asyncio  
import os  
from aiogram import Bot, Dispatcher  
from app.handlers import router  
from app.database.models import async\_main  
from pathlib import Path  
async def main():  
    await async\_main()  
    bot = Bot(token='7806482724:AAHM2Q-HEfUDryBEok2yysmSloEc37L-iYc') #токен  
    dp = Dispatcher()  
    dp.include\_router(router)  
    try:  
        await dp.start\_polling(bot)  
    except Exception as e:  
        print(f"Error: {e}")  
    finally:  
        await bot.session.close()  
if \_\_name\_\_ == '\_\_main\_\_':  
    try:  
        asyncio.run(main())  
    except KeyboardInterrupt:  
        print('Bot stopped')

Приложение Б

Данный код реализует взаимодействие Telegram-бота с базой данных SQLite через асинхронные запросы SQLAlchemy. Основные функции включают регистрацию пользователей (хранение пола, роста, веса). Код организован по принципу разделения ответственности: models.py определяет структуру БД, requests.py содержит CRUD-операции, а handlers.py реализует бизнес-логику бота.

Листинг 1 – requests.py  
  
# app/database/requests.py  
from app.database.models import async\_session  
from app.database.models import User, SleepCategory, SleepContent  
from sqlalchemy import select  
async def get\_user(tg\_id: int) -> User:  
    async with async\_session() as session:  
        return await session.scalar(select(User).where(User.tg\_id == tg\_id))  
async def create\_or\_update\_user(tg\_id: int, gender: str, age: int, weight: int, diseases: str):  
    async with async\_session() as session:  
        user = await session.scalar(select(User).where(User.tg\_id == tg\_id))  
        if user:  
            user.gender = gender  
            user.age = age  
            user.weight = weight  
            user.diseases = diseases  
            user.is\_registered = True  
        else:  
            session.add(User(  
                tg\_id=tg\_id,  
                gender=gender,  
                age=age,  
                weight=weight,  
                diseases=diseases,  
                is\_registered=True  
            ))  
        await session.commit()  
async def get\_sleep\_categories():  
    async with async\_session() as session:  
        return await session.scalars(select(SleepCategory))  
async def get\_sleep\_content(category\_id):  
    async with async\_session() as session:  
        return await session.scalars(select(SleepContent).where(SleepContent.category\_id == category\_id))  
async def get\_content(content\_id):  
    async with async\_session() as session:  
        return await session.scalar(select(SleepContent).where(SleepContent.id == content\_id))

Листинг 2 – models.py

from sqlalchemy import BigInteger, String, ForeignKey  
from sqlalchemy.orm import DeclarativeBase, Mapped, mapped\_column  
from sqlalchemy.ext.asyncio import AsyncAttrs, async\_sessionmaker, create\_async\_engine  
engine = create\_async\_engine(url='sqlite+aiosqlite:///db.sqlite3')

async\_session = async\_sessionmaker(engine)  
from sqlalchemy import BigInteger, String, ForeignKey  
from sqlalchemy.orm import DeclarativeBase, Mapped, mapped\_column  
from sqlalchemy.ext.asyncio import AsyncAttrs, async\_sessionmaker, create\_async\_engine  
class Base(AsyncAttrs, DeclarativeBase):    
  
class User(Base):  
    \_\_tablename\_\_ = 'users'  
     
    id: Mapped[int] = mapped\_column(primary\_key=True)  
    tg\_id: Mapped[int] = mapped\_column(BigInteger, unique=True)  
    gender: Mapped[str] = mapped\_column(String(1), nullable=True)   
    height: Mapped[int] = mapped\_column(nullable=True)     
    weight: Mapped[int] = mapped\_column(nullable=True)     
    is\_registered: Mapped[bool] = mapped\_column(default=False)  
  
class Category(Base):  
    \_\_tablename\_\_ = 'categories'  
     
    id: Mapped[int] = mapped\_column(primary\_key=True)  
    name: Mapped[str] = mapped\_column(String(30))  
  
class Workout(Base):  
    \_\_tablename\_\_ = 'training'  
     
    id: Mapped[int] = mapped\_column(primary\_key=True)  
    name: Mapped[str] = mapped\_column(String(30))

    description: Mapped[str] = mapped\_column(String(100))  
    videotraining: Mapped[str] = mapped\_column(String(100))  
    category: Mapped[int] = mapped\_column(ForeignKey('categories.id'))

async def async\_main():  
    async with engine.begin() as conn:  
        await conn.run\_sync(Base.metadata.create\_all)

Листинг 3 – handlers.py

from aiogram import F, Router  
from aiogram.types import Message, CallbackQuery, ReplyKeyboardMarkup, KeyboardButton  
from aiogram.filters import CommandStart  
from aiogram.fsm.context import FSMContext  
from aiogram.fsm.state import State, StatesGroup  
from aiogram.types import ReplyKeyboardRemove, InlineKeyboardMarkup, InlineKeyboardButton  
import app.database.requests as rq  
from datetime import datetime, timedelta  
import asyncio  
import app.keyboards as kb  
router = Router()  
class RegistrationStates(StatesGroup):  
    waiting\_for\_gender = State()  
    waiting\_for\_age = State()  
    waiting\_for\_weight = State()  
    waiting\_for\_diseases = State()  
    confirm\_data = State()  
gender\_keyboard = ReplyKeyboardMarkup(  
    keyboard=[  
        [KeyboardButton(text='Мужской'), KeyboardButton(text='Женский')]  
    ],  
    resize\_keyboard=True,  
    one\_time\_keyboard=True  
)  
diseases\_keyboard = ReplyKeyboardMarkup(  
    keyboard=[  
        [KeyboardButton(text='Бессонница')],  
        [KeyboardButton(text='Апноэ')],  
        [KeyboardButton(text='Храп')],  
        [KeyboardButton(text='Нет проблем')]  
    ],  
    resize\_keyboard=True,  
    one\_time\_keyboard=True  
)  
async def check\_registration(message: Message) -> bool:  
    user = await rq.get\_user(message.from\_user.id)  
    return user and user.is\_registered if user else False

Приложение В

Аннотация

So not – это Telegram-бот, созданный для помощи в улучшении качества сна. Он анализирует ваши индивидуальные параметры (возраст, пол, вес, особенности сна) и предоставляет персонализированные рекомендации: аудиосказки, медитации, ASMR и полезные материалы о здоровом сне.

Цель бота – упростить процесс подбора эффективных методик для нормализации сна, предлагая готовые решения на основе ваших данных.

Оформление программного документа «Руководство пользователя» произведено по требованиям ЕСПД (ГОСТ 19.101-77 [[1]](#footnote-1)1), ГОСТ 19.103-77 [[2]](#footnote-2)2), ГОСТ 19.104-78\* [[3]](#footnote-3)3), ГОСТ 19.105-78\* [[4]](#footnote-4)4), ГОСТ 19.106-78\* [[5]](#footnote-5)5), ГОСТ 19.505-79\* [[6]](#footnote-6)6), ГОСТ 19.604-78\* [[7]](#footnote-7)7)).

1 Введение

1.1 Область применения

Данное руководство пользователя предназначено для ознакомления с функциональными возможностями телеграм-бота, а также для получения практических рекомендаций по его использованию. Руководство ориентировано на всех, кто заинтересован в отслеживании качества сна и получении персонализированных рекомендаций, основанных на их индивидуальных параметрах и целях.

1.2 Краткое описание возможностей

Телеграм-бот So not представляет собой роль помощника для здорового сна.

Основные возможности бота включают:

* регистрация пользователя;
* ввод основных данных, таких как рост, вес и возраст;
* получение доступа к списку подходящего контента про сон;
* удобная и интуитивно понятная навигация через команды и кнопки.

1.3 Уровень подготовки пользователя

Telegram-бот разработан для пользователей с любым уровнем технической подготовки. Для его использования достаточно базового понимания работы Telegram и умения вести переписку в чатах. Специальных знаний или навыков не требуется. Интерфейс бота интуитивно понятен и прост в использовании.

2 Назначение и условия применения

2.1 Назначение

Telegram-бот So not разработан для комплексного решения проблем со сном.

2.2 Условия применения

Для использования Telegram-бота So not необходимо устройство с доступом в интернет и установленным приложением Telegram. Бот работает в стандартном интерфейсе Telegram и не требует установки дополнительного программного обеспечения. Использовать бота можно в любом месте и в любое время при наличии стабильного интернет-соединения. So not предназначен для индивидуального использования.

3 Подготовка к работе

3.1 Состав и содержание носителя данных

Telegram-бот So not размещен на удаленном сервере и доступен через Telegram. Данные, используемые ботом, хранятся в базе данных, связанной с серверной частью приложения. Это облачное хранилище содержит информацию о пользователях (рост, вес, возраст).

3.2 Порядок загрузки программ и данных

Для начала работы с Telegram-ботом So not необходимо открыть Telegram и перейти по ссылке на бота или найти его через строку поиска. Дополнительная установка не требуется, так как бот работает в облачной среде и доступен напрямую через интерфейс Telegram.

При первом запуске бот предложит зарегистрироваться и ввести основные данные: рост, вес и возраст. Вся последующая загрузка и обновление данных осуществляется через команды и кнопки внутри диалога, без необходимости каких-либо дополнительных действий со стороны пользователя.

3.3 Порядок проверки работоспособности

Проверка работоспособности Telegram-бота So not осуществляется непосредственно через взаимодействие с ним в приложении Telegram. После запуска бота пользователь должен получить приветственное сообщение и доступ к основным функциям, таким как регистрация, ввод параметров и получение доступа к планам тренировок. Успешный отклик бота на команды и корректное отображение информации свидетельствуют о его работоспособности. В случае отсутствия ответа, появления ошибок или некорректного поведения необходимо проверить подключение к интернету, убедиться в актуальности версии Telegram и повторно запустить бота.

4 Описание операций

Взаимодействие с Telegram-ботом So not происходит в диалоговом режиме внутри мессенджера Telegram. Пользователь начинает работу, отправляя команду или выбирая пункт из меню. Каждая операция представляет собой последовательность шагов для ввода данных, получения планов тренировок или получения информации.

Первым этапом является запуск бота. После перехода по ссылке или поиска в Telegram бот отправляет приветственное сообщение и предлагает зарегистрироваться. Пользователь вводит основные данные о себе: рост, вес, возраст.

5 Аварийные ситуации

В процессе работы с Telegram-ботом So not могут возникнуть некоторые проблемы, вызванные различными факторами. Ниже приведены наиболее распространенные ошибки и способы их устранения.

Отсутствие реакции на команды может быть вызвано проблемами с подключением к интернету, устаревшей версией Telegram или ошибками в написании команды. В этом случае проверьте подключение к интернету, обновите приложение Telegram до последней версии и убедитесь, что команда введена правильно.

Некорректное отображение информации может быть связано с ошибками в данных, переданных пользователем, или проблемами на сервере. Проверьте правильность введенных данных и попробуйте перезапустить бот. Если проблема сохраняется, обратитесь в службу поддержки.

Невозможность загрузки медиа может возникнуть из-за проблем с подключением к интернету или временных сбоев на сервере. В этом случае проверьте подключение к интернету и попробуйте повторить запрос позже. Если проблема не исчезает, обратитесь в службу поддержки.

Если бот не отвечает (не работает), это может быть связано с техническими проблемами на стороне сервера или временными сбоями. Подождите некоторое время и попробуйте снова. Если проблема сохраняется, перезапустите бот или обратитесь в службу поддержки.

В случае возникновения любых проблем, рекомендуется в первую очередь проверить подключение к интернету и убедиться в актуальности версии Telegram. Если проблема не исчезает, обратитесь в службу поддержки, используя контактную информацию, предоставленную в интерфейсе бота.

6 Рекомендации по освоению

Освоение работы с Telegram-ботом So not не займет много времени. Для начала ознакомьтесь с приветственным сообщением бота.

Пройдите регистрацию и введите свои основные данные.

В случае возникновения вопросов, обратитесь к данному руководству пользователя.

1. 1) ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов [↑](#footnote-ref-1)
2. 2) ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов [↑](#footnote-ref-2)
3. 3) ГОСТ 19.104-78\* ЕСПД. Основные надписи [↑](#footnote-ref-3)
4. 4) ГОСТ 19.105-78\* ЕСПД. Общие требования к программным документам [↑](#footnote-ref-4)
5. 5) ГОСТ 19.106-78\* ЕСПД. Общие требования к программным документам, выполненным печатным способом [↑](#footnote-ref-5)
6. 6) ГОСТ 19.505-79\* ЕСПД. Руководство пользователя. Требования к содержанию и оформлению [↑](#footnote-ref-6)
7. 7) ГОСТ 19.604-78\* ЕСПД. Правила внесения изменений в программные документы, выполненные печатным способом [↑](#footnote-ref-7)