**Пояснительная записка к проекту:**

Telegram-бот для управления задачами (To-Do List)

**Авторы**:  Маркин Даниил, Кузин Сергей, Иванова Альмира

1. Введение

Проект представляет собой **многофункционального Telegram-бота** для управления задачами (To-Do List) с использованием **SQLite** в качестве базы данных. Бот позволяет пользователям создавать, редактировать, удалять задачи, устанавливать напоминания, добавлять описания к задачам и отслеживать их статусы.

**Цель проекта**:

Разработать удобный инструмент для планирования задач с расширенным функционалом, включая хранение данных в SQLite и добавление подробных описаний задач.

**Задачи:**

- Реализовать интерактивное меню с кнопками для управления задачами.

- Настроить систему хранения данных в **SQLite**.

- Добавить функционал **напоминаний** с асинхронной проверкой времени.

- Внедрить возможность добавления **описаний** к задачам.

- Обеспечить обработку ошибок и обратную связь для пользователя.

- Оптимизировать производительность бота для стабильной работы.

**2. Техническая реализация:**

2.1. **Используемые технологии:**

- **Язык программирования:** Python 3.10+  

- **Библиотеки**:

  - `aiogram` – для работы с Telegram API и асинхронного взаимодействия.  

  - `sqlite3` – хранение данных в SQLite.

  - `datetime` и `asyncio` – обработка времени и асинхронных задач.  

- **Хранение данны**х: Локальная база данных SQLite (`todo\_bot.db`).

**2.2. Структура базы данных**

**Таблица `tasks`:**  

- `id` (INTEGER, PRIMARY KEY) – уникальный идентификатор задачи.

- `user\_id` (INTEGER) – ID пользователя в Telegram.

- `title` (TEXT) – название задачи.

- `description` (TEXT) – подробное описание задачи (новое поле).

- `status` (TEXT) – статус задачи ("Не начата", "В процессе", "Выполнена").

- `reminder` (TEXT) – дата и время напоминания (формат: `YYYY-MM-DD HH:MM`(Формат измениться)).

- `created\_at` (TEXT) – дата создания задачи.

**2.3. Основные компоненты**

1. **Система состояний (FSM):**

   - Реализована для поэтапного взаимодействия (например, добавление задачи → ввод названия → описание → установка напоминания).  

2. **Меню бота:**

   - Кнопки: `➕ Добавить задачу`, `📋 Список задач`, `✅ Завершить`, `❌ Удалить`.

3. **Работа с задачами:**

   - Добавление/удаление, изменение статуса.

   - Добавление **описания** к задаче (новая фишечка).

   - Напоминания в формате `YYYY-MM-DD HH:MM` (изменяться, но я хз как).

4. **Обратная связь:**

   - Уведомления об успешных действиях и ошибках (например, некорректный формат даты).

**4.2 Особенности реализации**

  - Данные хранятся в структурированной таблице, что ускоряет поиск и фильтрацию.

  - Автоматическое создание базы при первом запуске.

- **Описания задач:**

  - Пользователь может добавить детальное описание к задаче (максимум 500 символов).

- **Асинхронность**:

  - Проверка напоминаний каждую минуту без блокировки основного потока.

**3.2. Сценарии использования**

1. **Добавление задачи:**

   - Пользователь нажимает `➕ Добавить задачу` → вводит название → описание (опционально) → устанавливает напоминание.  

2. **Просмотр задач:**

   - Отображение списка с нумерацией, статусами, описаниями и датами.  

3. **Напоминания**:

   - Бот отправляет сообщение в указанное время и обновляет статус задачи.

**3.3. Обработка ошибок**

- Некорректный ввод даты: \*"Ошибка: используйте формат YYYY-MM-DD HH:MM"\*.

- Пустой список задач: **"У вас нет активных задач!"**.

**4. Этапы разработки**

1. **Проектирование**:

   - Создание схемы базы данных SQLite.

   - Дизайн FSM-состояний для новых полей (описание).  

2. **Реализация**:

   - Написание ядра бота (обработчики команд, меню).

   - Интеграция SQLite и функционала описаний.  

3. **Тестирование**:

   - Проверка корректности сохранения данных в SQLite.

   - Тесты на одновременные запросы от пользователей.  

4. **Деплой**:

   - Запуск бота на хостинг сервере.

**5. Заключение**

Telegram-бот для управления задачами успешно реализован как удобное и надежное решение для повышения личной продуктивности. Проект сочетает простой интерфейс с мощным функционалом, включая гибкие напоминания и детальное описание задач. Использование SQLite обеспечивает стабильное хранение данных, а модульная архитектура позволяет легко расширять функционал. Бот готов к использованию и имеет хороший потенциал для дальнейшего развития через интеграции с популярными сервисами и добавление новых возможностей для пользователей.