**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное**

**учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Математической кибернетики и информационных технологий»

**Отчет по проектному практикуму**

**«Telegram-бот «Напоминалка-бот»»**

Выполнил студент

Группы БВТ2204

Соловов Алексей

Москва

2023

**Введение**

Цель нашей работы создать Telegram-бота, который может предоставить пользователям удобное средство для организации и управления своими задачами, заметками и напоминаниями через платформу Telegram.

Telegram API является мощным инструментом для разработчиков, позволяющим создавать разнообразные приложения и ботов, которые интегрируются с экосистемой Telegram и обогащают пользовательский опыт.

Телеграм-бот (или Telegram-бот) — это программное приложение, которое автоматически выполняет задачи и предоставляет информацию в чате на платформе Telegram. Такие боты создаются разработчиками и могут выполнять разнообразные функции, от простых команд и ответов на вопросы до сложных автоматизированных действий

Телеграм-боты предоставляют ряд удобств, которые делают их привлекательными средствами коммуникации и автоматизации:

* Доступность: Телеграм-боты доступны на платформе Telegram, которая широко используется по всему миру, и их можно использовать на различных устройствах, включая смартфоны, планшеты и компьютеры.
* Простота использования: Взаимодействие с ботами происходит через чат-подобный интерфейс, что делает их интуитивно понятными и легкими в использовании, особенно для пользователей Telegram.
* Мгновенные уведомления: Боты могут отправлять мгновенные уведомления и сообщения, что делает их полезными для напоминаний о важных событиях и задачах.
* Автоматизация задач: Телеграм-боты способны выполнять автоматические задачи и обрабатывать запросы пользователя, что может значительно упростить выполнение повседневных задач.
* Многозадачность: Пользователи могут одновременно взаимодействовать с разными ботами и выполнять разные задачи в одном мессенджере.
* Безопасность: Telegram обеспечивает высокий уровень шифрования данных, что делает использование ботов относительно безопасным способом управления информацией.

Эти преимущества делают телеграм-боты удобными средствами для взаимодействия, организации и автоматизации различных задач и операций.

**Создание телеграм-бота (ход работы)**

Для того, чтобы создать телеграм-бота, нам необходимо импортировать все необходимые библиотеки и модули для работы с ним и другими функциями. Создаем экземпляр бота с использованием библиотеки telebot. Переменная TOKEN хранит токен Telegram-бота, который используется для аутентификации и идентификации бота в Telegram API. Класс User предназначен для хранения информации о каждом пользователе Telegram-бота. Он может записывать и удалять задачи и напоминания для каждого пользователя. Таким образом, класс User помогает боту управлять и хранить персональные задачи и напоминания для разных пользователей. Создаем словарь, для хранения данных.

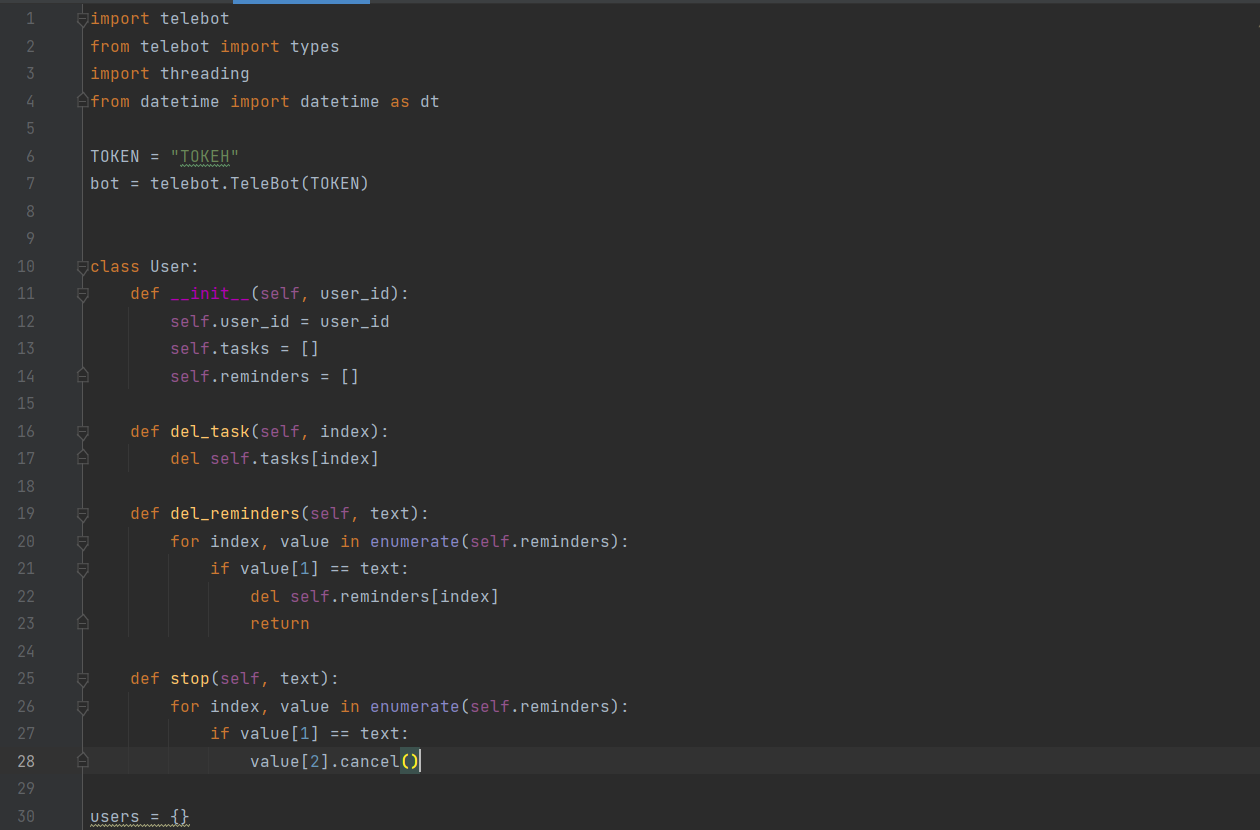


Рисунок 1 – Создание словаря для хранения данных

Прописываем функцию для создания инлайн-клавиатуры.

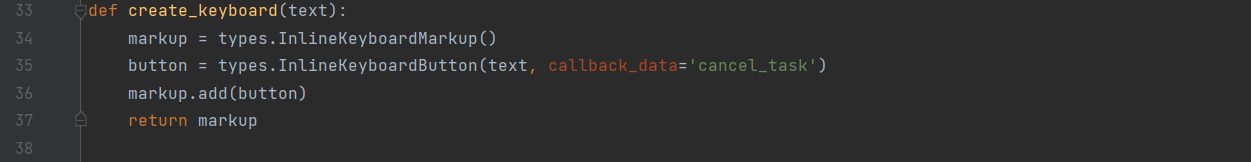


Рисунок 2 – Функция для создания инлайн-клавиатуры

Создаем функцию для обработки колбэк-запросов (callback queries) от пользователя, которые связаны с отменой задачи.

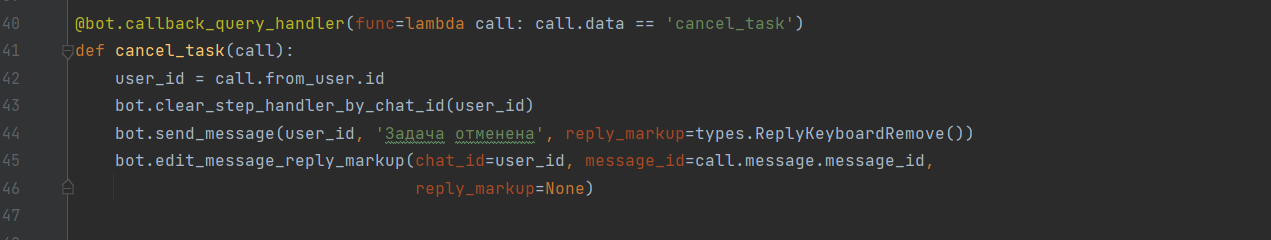


Рисунок 3 – Функция для обработки колбэк-запросов

Далее идет обработка команд /start и /help от пользователя и отправка приветственное сообщение с инструкциями по использованию бота-планировщика задач. Если пользователь новый (его идентификатор не найден в словаре users), то для него создается новый объект класса User.

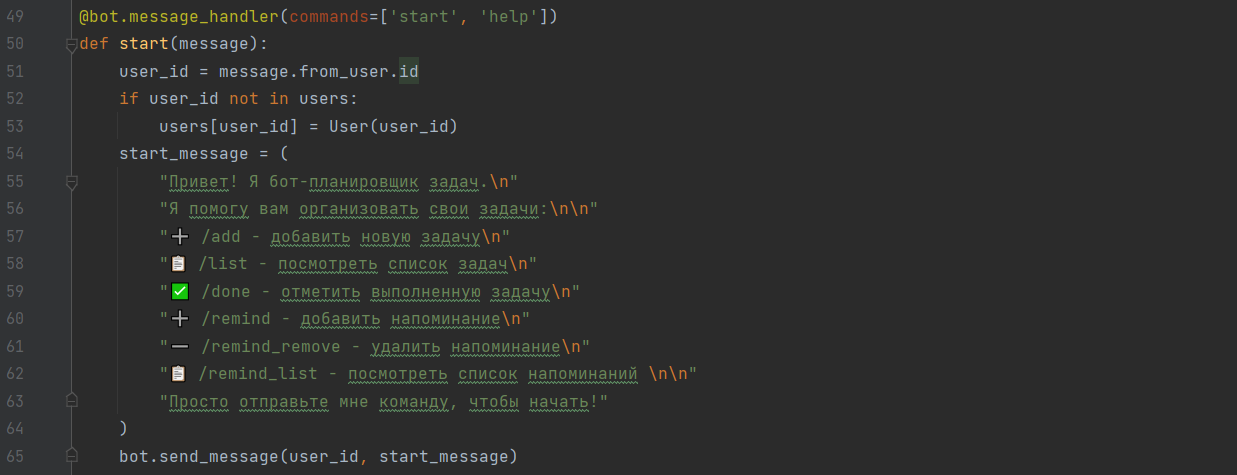


Рисунок 4 – Обработка команд /start и /help

Прописываем в коде команду /add от пользователя для добавления новой задачи.

В функции add\_task создается инлайн-клавиатура с кнопкой "Отменить добавление задачи", и бот отправляет сообщение с запросом описания задачи, прикрепляя эту клавиатуру.

Затем регистрируется обработчик следующего шага, который будет вызван после ввода пользователем описания задачи.

В функции process\_new\_task полученное описание задачи добавляется в список задач пользователя, и инлайн-клавиатура убирается. После этого бот отправляет сообщение о успешном добавлении задачи.

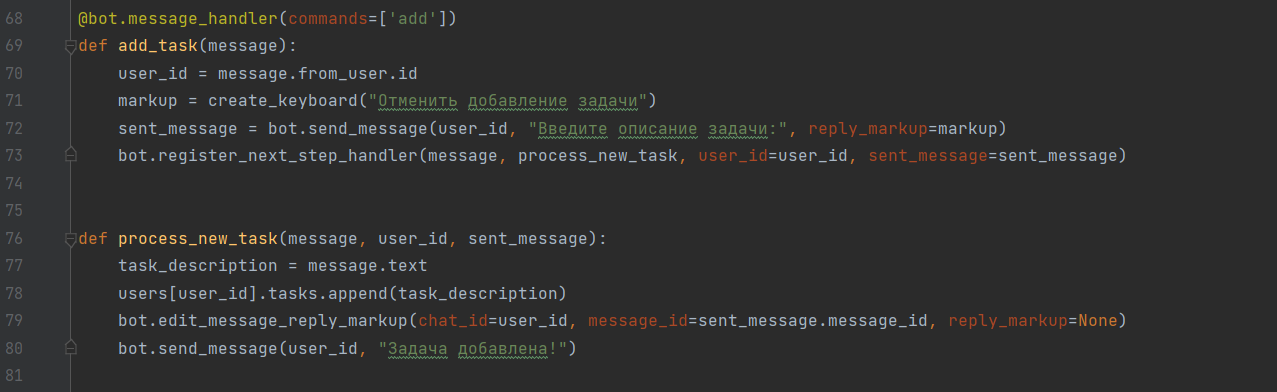


Рисунок 5 - Команда /add от пользователя для добавления новой задачи

Создаем команду /list от пользователя для просмотра списка задач. Бот получает идентификатор пользователя из сообщения. Затем он проверяет, есть ли у пользователя задачи в его списке. Если задачи отсутствуют, бот отправляет сообщение, уведомляя пользователя, что список задач пуст. Если у пользователя есть задачи, бот формирует текстовый список задач, перечисляя их с номерами, и отправляет его пользователю. Каждая задача имеет уникальный номер в списке для удобства идентификации. Этот код позволяет пользователям просматривать список своих задач с помощью команды /list.

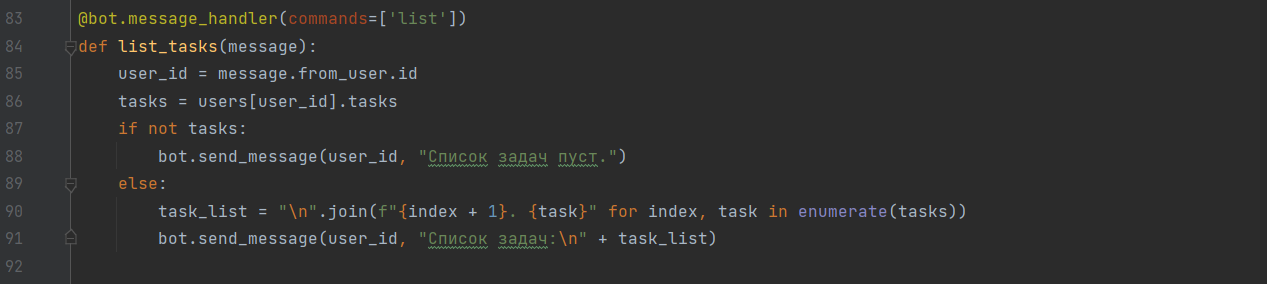


Рисунок 6 – Команда /list

Добавляем команду /done от пользователя для отметки выполненных задач. Бот получает идентификатор пользователя из сообщения и проверяет, есть ли у пользователя задачи в его списке. Если задачи отсутствуют, бот отправляет сообщение, уведомляя пользователя, что список задач пуст, и прекращает выполнение. Если у пользователя есть задачи, бот создает клавиатуру с кнопками задач и одной дополнительной кнопкой "ОТМЕНА". Затем бот отправляет сообщение, предлагая пользователю выбрать задачу для отметки выполнения с помощью этой клавиатуры и регистрирует обработчик следующего шага. В функции process\_done\_task бот обрабатывает выбор пользователя. Если пользователь выбрал задачу, и она совпадает с задачей в списке, задача помечается как выполненная и удаляется из списка. Если выбор сделан некорректно, бот предостерегает пользователя и повторяет запрос.

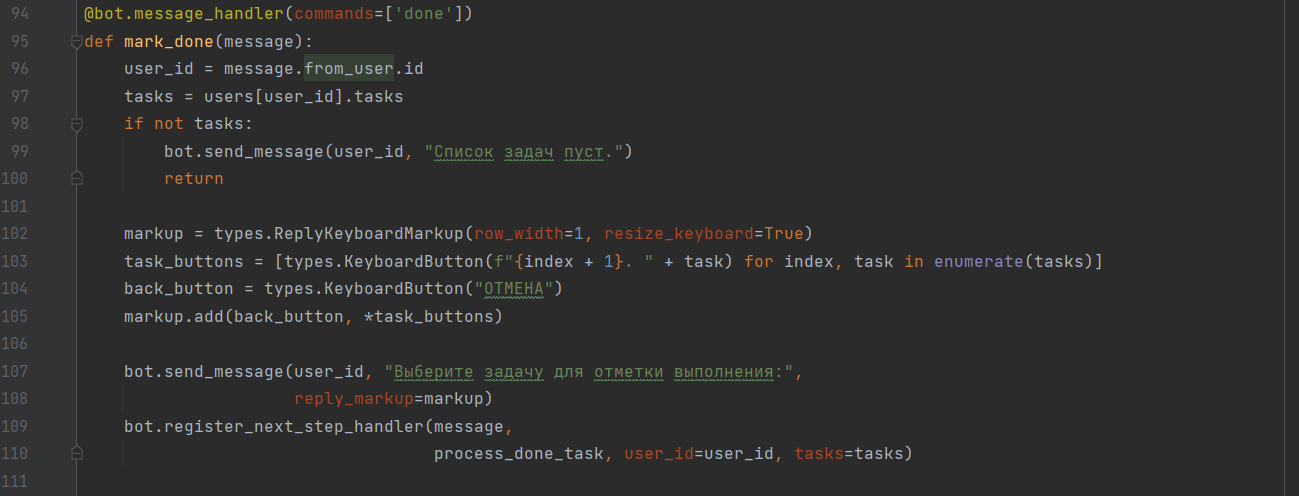


Рисунок 7 – Команда /done

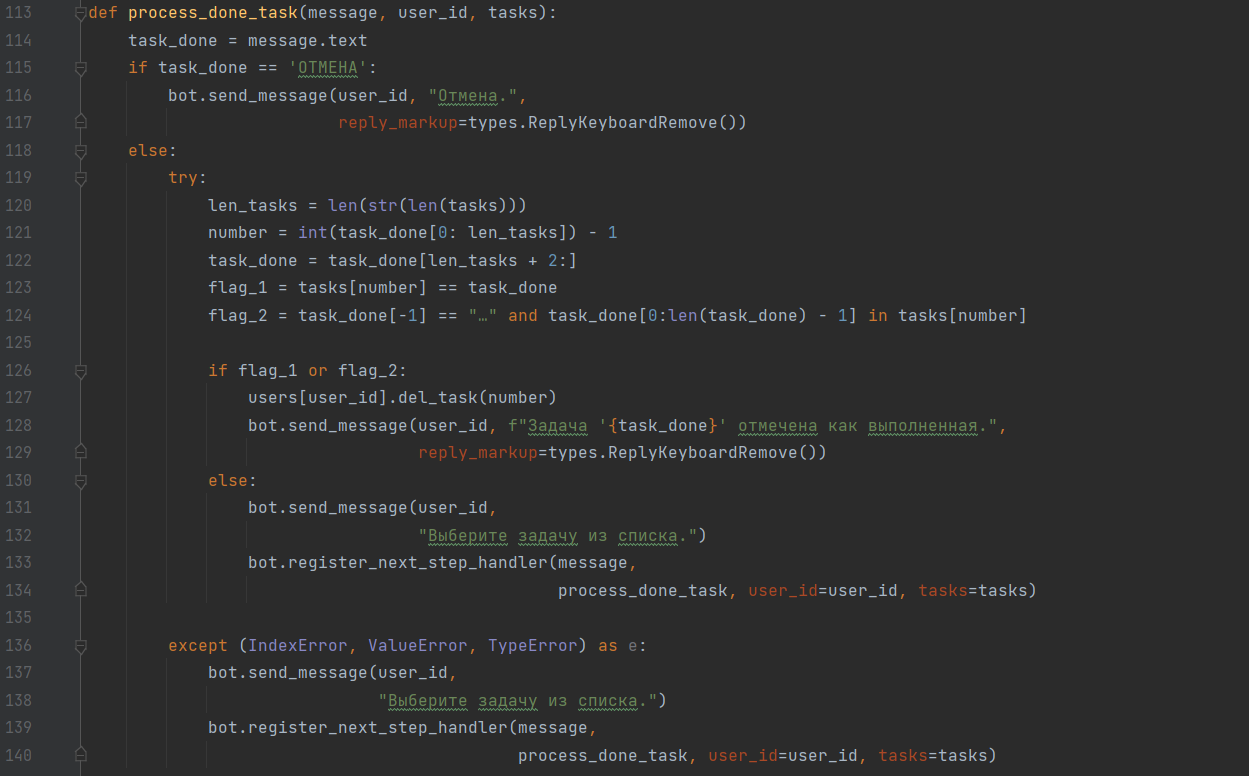


Рисунок 8 – Функция process\_done\_task

После идет обработка команды /remind от пользователя для создания напоминаний. При получении команды, бот запрашивает у пользователя ввод даты и времени для напоминания в формате "ДД.ММ.ГГ ЧЧ:ММ". Если формат ввода неверен, бот предупреждает пользователя.

Когда мы смогли успешно вывести дату и время, бот запрашивает текст напоминания. Затем бот вычисляет разницу во времени между указанным временем напоминания и текущим временем. Если разница положительная (т.е., напоминание в будущем), бот создает таймер, который отправит напоминание по истечении указанного времени. Также информация о напоминании сохраняется в списке напоминаний пользователя. Если указанное время уже прошло, бот сообщает пользователю, что невозможно установить напоминание.

Когда наступает время напоминания, таймер вызывает функцию send\_reminder, которая отправляет пользователю текст напоминания. После этого информация о напоминании удаляется из списка напоминаний пользователя.

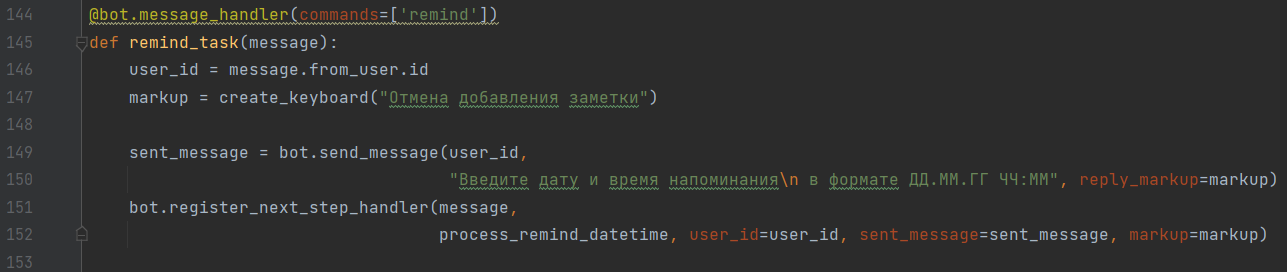


Рисунок 9 - Команда /remind

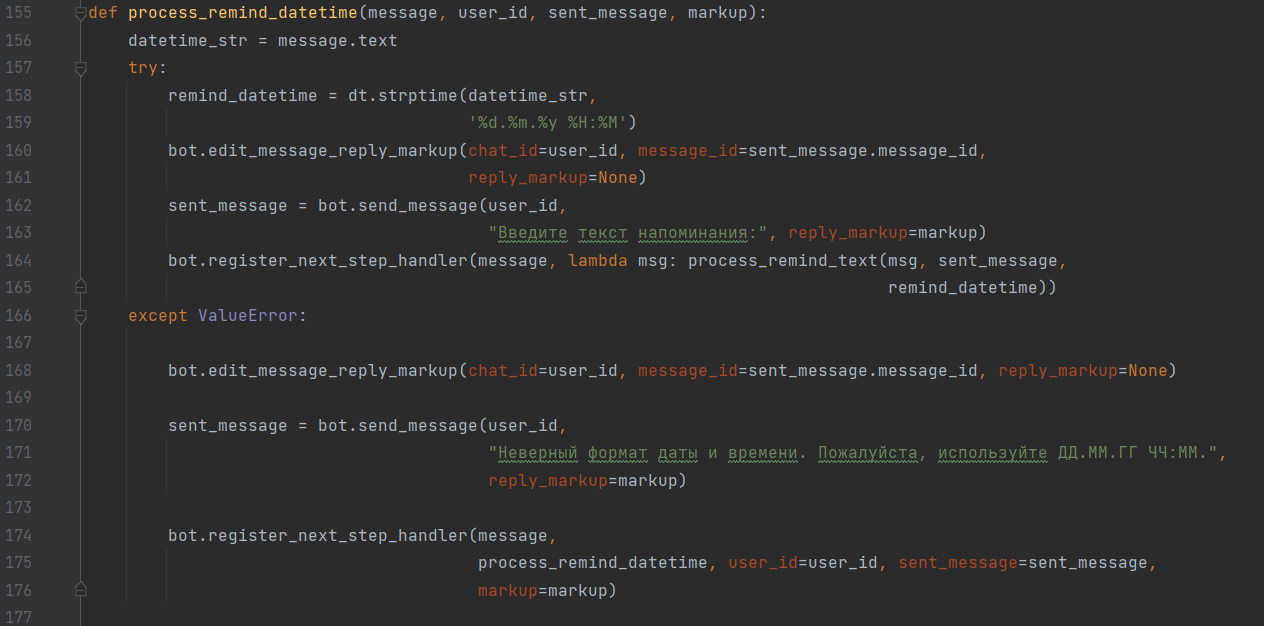


Рисунок 10 - Команда /remind

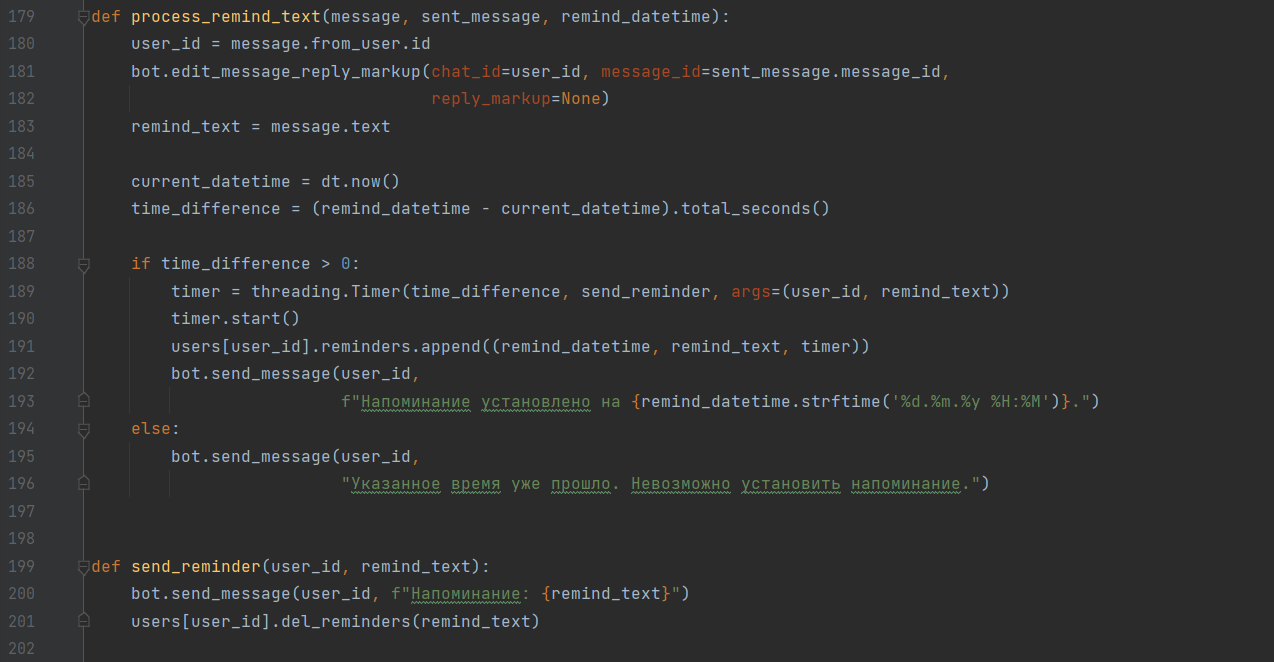


Рисунок 11 - Команда /remind и вызов функции send\_reminder

Затем идет обработка команды /remind\_list от пользователя для просмотра списка его напоминаний. Бот получает идентификатор пользователя из сообщения. Затем он проверяет, есть ли у пользователя активные напоминания в его списке. Если напоминания отсутствуют, бот отправляет сообщение, уведомляя пользователя, что список напоминаний пуст. Если у пользователя есть активные напоминания, бот формирует текстовый список напоминаний, включая дату и время напоминания, а также текст напоминания. Затем бот отправляет этот список пользователю.

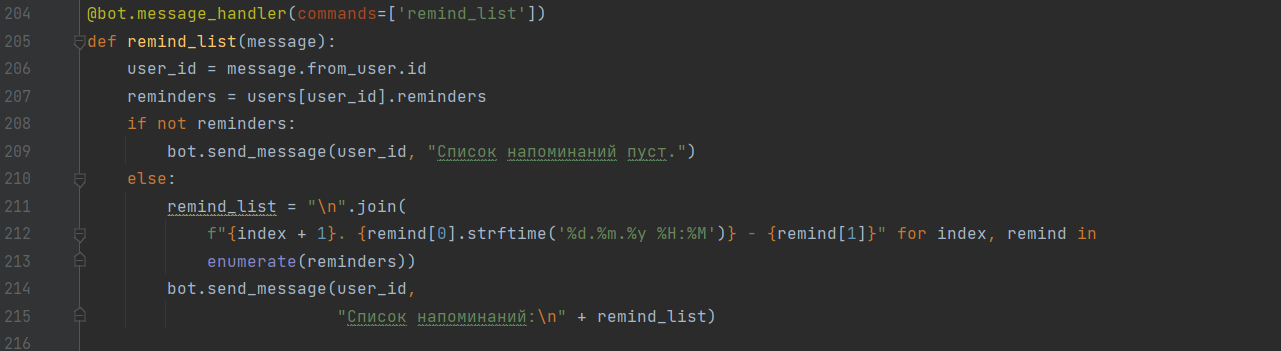


Рисунок 12 - Команда /remind\_list

Добавляем команду /remind\_remove от пользователя для удаления напоминаний. Бот получает идентификатор пользователя из сообщения и список его активных напоминаний. Если у пользователя нет активных напоминаний, бот отправляет сообщение, уведомляя пользователя, что список напоминаний пуст.

Если у пользователя есть активные напоминания, бот создает клавиатуру с кнопками напоминаний и одной дополнительной кнопкой "Назад". Затем бот отправляет сообщение, предлагая пользователю выбрать напоминание для удаления с помощью этой клавиатуры и регистрирует обработчик следующего шага.

В функции process\_remind\_remove бот обрабатывает выбор пользователя. Если пользователь выбрал напоминание, и оно совпадает с напоминанием в списке, напоминание останавливается и удаляется из списка. Если выбор сделан некорректно, бот предостерегает пользователя и повторяет запрос. Этот код позволяет пользователям удалять свои активные напоминания с помощью команды /remind\_remove.



Рисунок 13 - Команда /remind\_remove

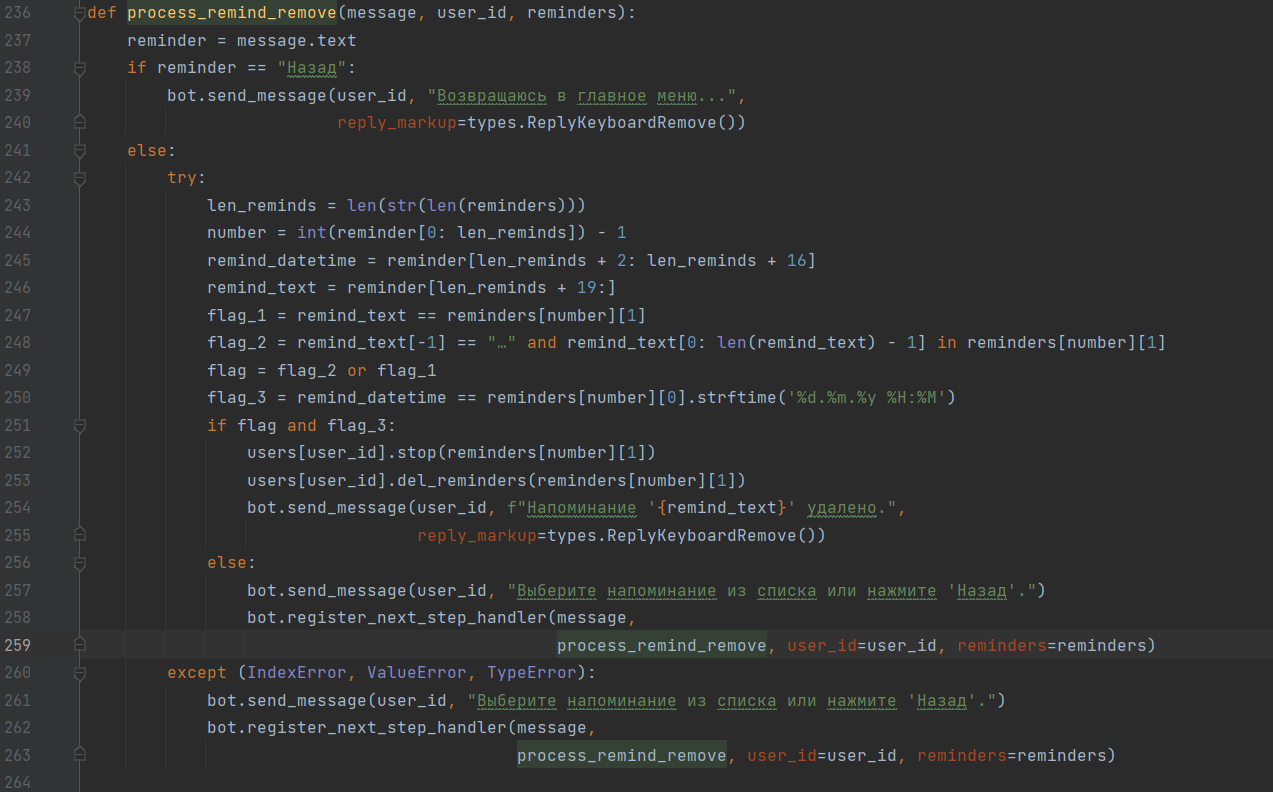


Рисунок 14 - Функция process\_remind\_remove

Далее представлен фрагмент кода, который является точкой входа в программу. Если программа запускается как отдельный скрипт (а не импортируется как модуль), то блок if \_\_name\_\_ == '\_\_main\_\_': выполняется.

В данном случае, код запускает бота с помощью функции bot.polling(none\_stop=True). Бот будет работать непрерывно, прослушивая новые сообщения и команды от пользователей. Этот блок кода обеспечивает непрерывную работу бота и является точкой старта для выполнения основной функциональности бота.

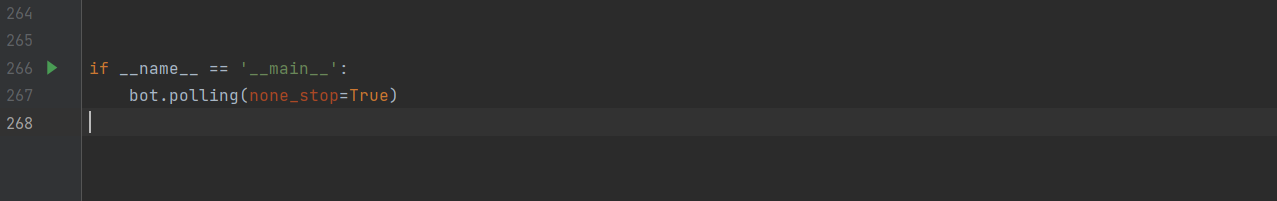


Рисунок 15 - Запуск бота с помощью функции bot.polling(none\_stop=True)

**Результаты тестирования**

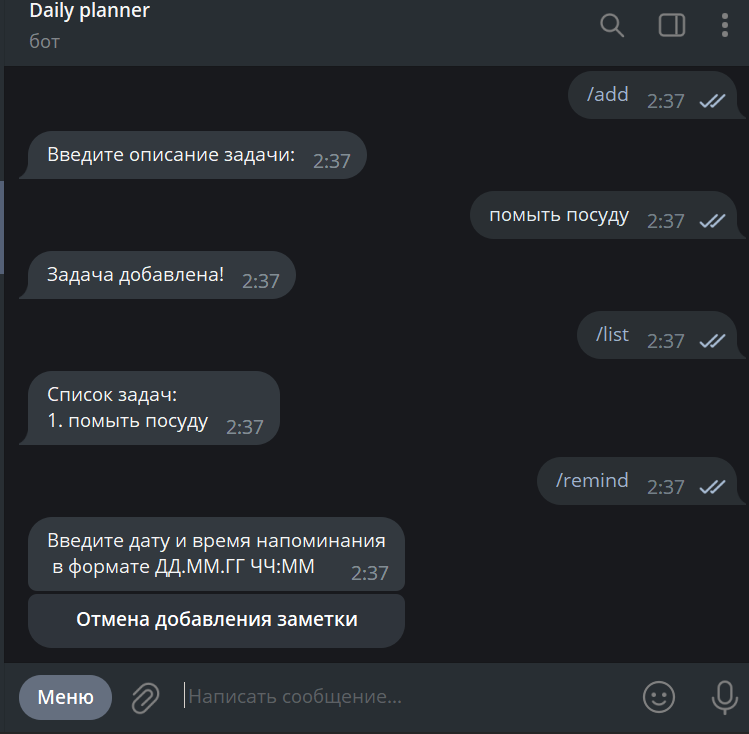
****

Рисунок 16 – Тестирование бота

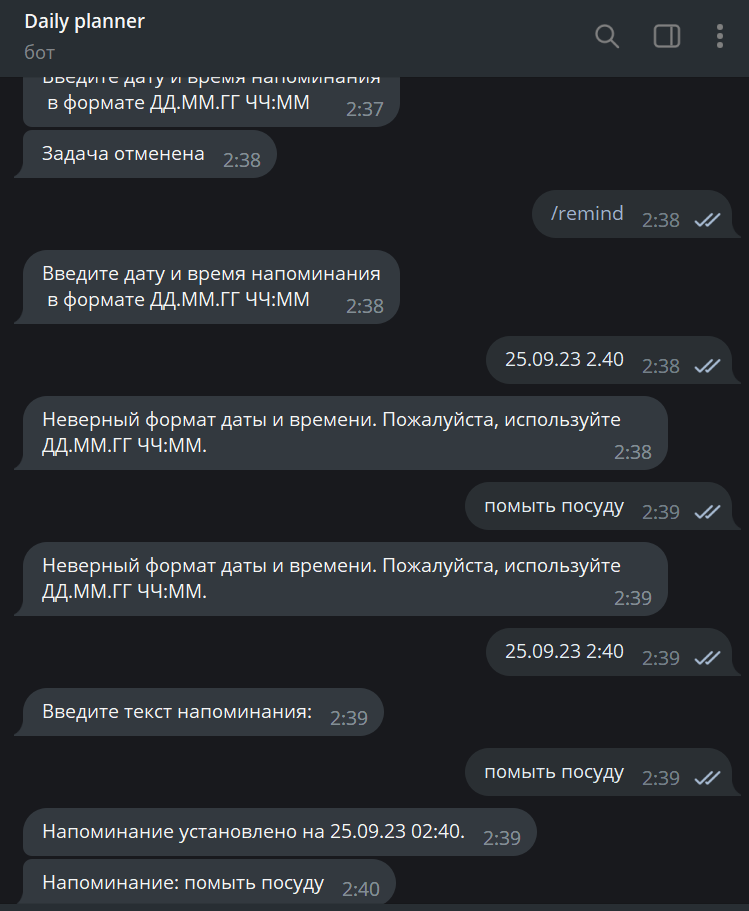
****

Рисунок 17 – Тестирование бота

**Заключение**

Таким образом, из всего вышенаписанного можно сделать вывод, что задание, поставленное в начале практики, было успешно выполнено. В процессе учебной практики был проанализирован и создан Telegram-бот, который представляет собой удобное и функциональное приложение для организации задач и напоминаний. Его основные функции включают в себя добавление, просмотр, отметку выполненных задач, а также создание и управление напоминаниями. Этот бот позволяет пользователям легко вести список дел и не пропускать важные события, благодаря возможности установки напоминаний на определенное время и дату. В заключении можно сказать, что благодаря прохождению данной практики мы смогли приобрести полезные знания и навыки, которые пригодятся нам в будущем.