**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №5

«Разработка простого бота для Telegram с использованием языка Python»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-34Б |  | преподаватель каф. ИУ5 |
| Юрченко Кирилл |  | Нардид Анатолий Николаевич |
|  |  |  |
|  |  |  |

Москва, 2024 г.

Постановка задачи

1. Разработайте простого бота для Telegram. Бот должен использовать функциональность создания кнопок.

Текст программы

import telebot  
from telebot.types import InlineKeyboardMarkup, InlineKeyboardButton, CallbackQuery  
  
bot = telebot.TeleBot("7890336716:AAEBYqOCADBQpDDfXpz-WRdR7xKxqP1LOKI")  
  
notes = {}  
  
  
def get\_main\_keyboard():  
 keyboard = InlineKeyboardMarkup(row\_width=2)  
 keyboard.add(  
 InlineKeyboardButton(text="Добавить заметку", callback\_data="add\_note"),  
 InlineKeyboardButton(text="Мои заметки", callback\_data="view\_notes"),  
 )  
 return keyboard  
  
  
@bot.message\_handler(commands=["start"])  
def start\_command(message):  
 bot.send\_message(message.chat.id,"Бот для заметок. Выберите действие:",reply\_markup=get\_main\_keyboard())  
  
@bot.callback\_query\_handler(func=lambda call: call.data in ["add\_note", "view\_notes"])  
def process\_callback(call: CallbackQuery):  
 user\_id = call.from\_user.id  
  
 if call.data == "add\_note":  
 msg = bot.send\_message(user\_id, "Введите текст заметки:")  
 bot.register\_next\_step\_handler(msg, add\_note\_handler)  
 bot.answer\_callback\_query(call.id)  
  
 elif call.data == "view\_notes":  
 if user\_id in notes and notes[user\_id]:  
 text = "Ваши заметки:\n" + "\n".join([f"{i + 1}. {note}" for i, note in enumerate(notes[user\_id])])  
 keyboard = InlineKeyboardMarkup(row\_width=1)  
 keyboard.add(InlineKeyboardButton("Удалить заметку", callback\_data="delete\_note"))  
 bot.send\_message(user\_id, text, reply\_markup=keyboard)  
 else:  
 bot.send\_message(user\_id, "У вас пока нет заметок.")  
 bot.answer\_callback\_query(call.id)  
  
  
def add\_note\_handler(message):  
 user\_id = message.from\_user.id  
 if user\_id not in notes:  
 notes[user\_id] = []  
 notes[user\_id].append(message.text)  
 bot.send\_message(user\_id, "Заметка успешно добавлена!", reply\_markup=get\_main\_keyboard())  
  
  
@bot.callback\_query\_handler(func=lambda call: call.data == "delete\_note")  
def delete\_note\_handler(call: CallbackQuery):  
 user\_id = call.from\_user.id  
 if user\_id in notes and notes[user\_id]:  
 keyboard = InlineKeyboardMarkup(row\_width=1)  
 for i, note in enumerate(notes[user\_id]):  
 keyboard.add(InlineKeyboardButton(text=note, callback\_data=f"del\_{i}"))  
 bot.send\_message(user\_id, "Выберите заметку для удаления:", reply\_markup=keyboard)  
 else:  
 bot.send\_message(user\_id, "У вас пока нет заметок.")  
 bot.answer\_callback\_query(call.id)  
  
  
@bot.callback\_query\_handler(func=lambda call: call.data.startswith("del\_"))  
def confirm\_delete\_handler(call: CallbackQuery):  
 user\_id = call.from\_user.id  
 note\_index = int(call.data.split("\_")[1])  
  
 if user\_id in notes and 0 <= note\_index < len(notes[user\_id]):  
 deleted\_note = notes[user\_id].pop(note\_index)  
 bot.send\_message(user\_id, f"Заметка '{deleted\_note}' удалена!", reply\_markup=get\_main\_keyboard())  
 else:  
 bot.send\_message(user\_id, "Ошибка: заметка не найдена.")  
 bot.answer\_callback\_query(call.id)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 bot.infinity\_polling()