**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет информатика и системы управления

Кафедра системы обработки информации и управления

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №2

«Разработка простого бота для Telegram с использованием языка Python.»

Вариант 20

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-32Б: |  | преподаватель каф. ИУ5 |
| Пчелинцева Д.А. |  | Нардид А.Н. |
| Подпись и дата: 29.10.2025г. |  | Подпись и дата: |

Москва, 2025 г.

**Задание**

Разработка простого бота для Telegram с использованием языка Python.

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

**Текст программы**

**Файл lab\_2.py** (бот, показывающий погоду на завтра)**:**  
import telebot

from telebot import types

import requests

weather\_api\_key = '2cc213560a984e94aa274022251109'

bot = telebot.TeleBot('8254669891:AAE7ePtIM\_oc9VTWcdRNtbFehEO8zV4SoPo')

user\_states = {}  # user\_states[user\_id] = 'waiting\_for\_city'

def menu(message, first\_time=False):

    markup = types.ReplyKeyboardMarkup(resize\_keyboard=True)

    btn1 = types.KeyboardButton('⚡ О боте ⚡')

    btn2 = types.KeyboardButton('☀️ Узнать погоду на завтра ☀️')

    btn3 = types.KeyboardButton('↩️ Выход ↩️')

    markup.row(btn1)

    markup.row(btn2,btn3)

    if first\_time:

        bot.send\_message(

            message.chat.id,

            f'Привет, {message.from\_user.first\_name}! Я могу:\n'

            '1. Рассказать о себе\n'

            '2. Узнать погоду в любом городе на завтра 🌤'

            '3. Выйти из приложения',

            parse\_mode='Markdown'

        )

    bot.send\_message(

        message.chat.id,

        "Выберите действие:",

        reply\_markup=markup

    )

@bot.message\_handler(commands=['start', 'hello', 'Привет'])

def start(message):

    user\_states[message.from\_user.id] = None  # сброс состояния

    menu(message, first\_time=True)

@bot.message\_handler(func=lambda message: True)

def handle\_all\_messages(message):

    user\_id = message.from\_user.id

    text = message.text.strip()

    if user\_states.get(user\_id) == 'waiting\_for\_city':

        if text == '🔙 Вернуться в меню':

            user\_states[user\_id] = None

            menu(message)

        else:

            get\_city\_and\_show\_weather(message)

        return

    # Обработка кнопок меню

    if text == '⚡ О боте ⚡':

        about\_bot(message)

    elif text == '☀️ Узнать погоду на завтра ☀️':

        ask\_for\_city(message)

    elif text=='↩️ Выход ↩️':

        exit\_bot(message)

    elif text == '🔙 Вернуться в меню':

        user\_states[user\_id] = None

        menu(message,first\_time=False)

    else:

        bot.send\_message(message.chat.id, "Пожалуйста, используйте кнопки ниже.", reply\_markup=get\_menu\_markup())

def about\_bot(message):

    bot.send\_message(

        message.chat.id,

        f'Бот создан, чтобы узнавать погоду для вас и ваших близких 🌹\n'

        'Просто выберите "Узнать погоду" и введите название города — я скажу, какая будет погода завтра!'

    )

    menu(message)

def ask\_for\_city(message):

    user\_states[message.from\_user.id] = 'waiting\_for\_city'

    markup = types.ReplyKeyboardMarkup(resize\_keyboard=True)

    btn = types.KeyboardButton('🔙 Вернуться в меню')

    markup.add(btn)

    bot.send\_message(

        message.chat.id,

        '🌆 Введите название города (например, Москва, Лондон, Токио):',

        reply\_markup=markup

    )

def get\_city\_and\_show\_weather(message):

    city = message.text.strip()

    user\_id = message.from\_user.id

    if not city or city == '🔙 Вернуться в меню':

        bot.send\_message(message.chat.id, "Название города не может быть пустым. Попробуйте снова:")

        return

    url = "http://api.weatherapi.com/v1/forecast.json"

    params = {

        'key': weather\_api\_key,

        'q': city,

        'days': 2,

        'lang': 'ru',

        'aq': 'no'

    }

    try:

        response = requests.get(url, params=params)

        data = response.json()

        if response.status\_code != 200:

            error\_msg = data.get('error', {}).get('message', 'Неизвестная ошибка')

            bot.send\_message(message.chat.id, f"❌ Ошибка: {error\_msg}")

            menu(message,first\_time=False)

            return

        forecast = data['forecast']['forecastday'][1]

        day = forecast['day']

        location = data['location']['name']

        country = data['location']['country']

        avg\_temp = day['avgtemp\_c']

        max\_temp = day['maxtemp\_c']

        min\_temp = day['mintemp\_c']

        condition = day['condition']['text']

        chance\_of\_rain = day['daily\_chance\_of\_rain']

        humidity = day['avghumidity']

        emoji = ("🌤" if "ясно" in condition.lower() else

                 "🌧" if "дождь" in condition.lower() or "ливень" in condition.lower() else

                 "☁️" if "облачно" in condition.lower() else "🌨")

        weather\_msg = (

            f"{emoji} \*\*Погода в {location}, {country} на завтра:\*\*\n\n"

            f"🌡 Средняя температура: {avg\_temp}°C\n"

            f"📈 Максимум: {max\_temp}°C | Минимум: {min\_temp}°C\n"

            f"💧 Влажность: {humidity}%\n"

            f"☔ Вероятность дождя: {chance\_of\_rain}%\n"

            f"📝 Условия: {condition.capitalize()}"

        )

        bot.send\_message(message.chat.id, weather\_msg, parse\_mode='Markdown')

    except requests.exceptions.RequestException:

        bot.send\_message(message.chat.id, "⚠️ Ошибка подключения к сервису погоды.")

    except KeyError:

        bot.send\_message(message.chat.id, "❌ Не удалось найти данные о погоде. Проверьте название города.")

    finally:

        user\_states[user\_id] = None

        menu(message, first\_time=False)

def get\_menu\_markup():

    markup = types.ReplyKeyboardMarkup(resize\_keyboard=True)

    markup.add(types.KeyboardButton('🔙 Вернуться в меню'))

    return markup

@bot.message\_handler(commands=['exit', 'stop', 'end'])

def exit\_bot(message):

    user\_id = message.from\_user.id

    user\_states[user\_id] = None

    remove\_markup = types.ReplyKeyboardRemove()  # убирает клавиатуру

    bot.send\_message(

        message.chat.id,

        f"До новых встреч, {message.from\_user.first\_name}!\n"

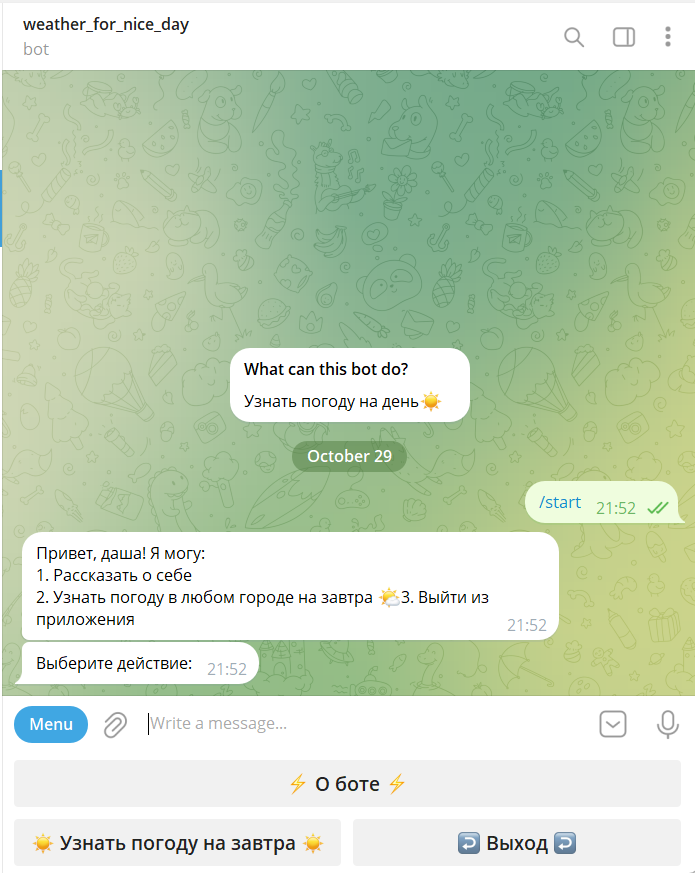
        "Для возобновления работы нажмите /start",

        reply\_markup=remove\_markup

    )

bot.polling(non\_stop=True)

**Скриншоты работы приложения**



**Рис. 1:** программа здоровается с пользователем, показывает, что он умеет и высвечивает кнопки для удобства пользования программой

  
**Рис. 2:** бот рассказывает о себе



**Рис. 3:** бот показывает погоду на следующий день в указанном пользователем городе



**Рис. 4:** выход из приложения

**Ветка с ЛР в моём GitHub:** <https://github.com/daria788/labs_of_pikeap/tree/lab_2>