```
import org.telegram.telegrambots.bots.TelegramLongPollingBot;
import org.telegram.telegrambots.meta.TelegramBotsApi;
import org.telegram.telegrambots.meta.api.methods.send.SendDocument;
import org.telegram.telegrambots.meta.api.methods.send.SendMessage;
import org.telegram.telegrambots.meta.api.methods.send.SendPhoto;
import org.telegram.telegrambots.meta.api.methods.send.SendSticker;
import org.telegram.telegrambots.meta.api.objects.*;
import org.telegram.telegrambots.meta.api.objects.replykeyboard.ReplyKeyboardMarkup;
import org.telegram.telegrambots.meta.api.objects.replykeyboard.buttons.KeyboardButton;
import org.telegram.telegrambots.meta.api.objects.replykeyboard.buttons.KeyboardRow;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;
import org.telegram.telegrambots.updatesreceivers.DefaultBotSession;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.*;
import java.util.Date;
public class Bot extends TelegramLongPollingBot {
    //объект класса доступа к бд
    ClassForGr obj = new ClassForGr();
    //список сохранящий данные вводимые учителем
    String []  TeacherDataForDb=new String[5];

    ArrayList<ArrayList<String>> GraphikForStudentromDb=obj.bd4();
//метод возвращающий токен созданного бота
    @Override
    public String getBotToken() {
        return "1548098213:AAGKsgzudQ2cpd1qKWI41iPS5yD5_L4MjDw";
    }
//метод отсылающий документ по его URL
```

```
public void senddoc(Message message, String httpQuery) {
    InputFile inp = new InputFile(httpQuery);
    SendDocument sd = new SendDocument();
    sd.setChatId(message.getChatId().toString());
    sd.setReplyToMessageId(message.getMessageId());
    sd.setDocument(inp);
    try {
        execute(sd);
    } catch (TelegramApiException e) {
        e.printStackTrace();
    }
}
//метод для отправки фото с заданиями из репозитория гитлаба
public void SENDDOC(Message message, String httpQuery) {
    InputFile inp=new InputFile(httpQuery);
    SendPhoto sd = new SendPhoto();
    sd.setChatId(message.getChatId().toString());
    sd.setReplyToMessageId(message.getMessageId());
    sd.setPhoto(inp);
    try {
        execute(sd);
    } catch (TelegramApiException e) {
        e.printStackTrace();
    }
}
public void sendsticker(Message message, String stid){
    InputFile inp=new InputFile(stid);
    SendSticker st = new SendSticker();
    st.setChatId(message.getChatId().toString());
```

```
        st.setReplyToMessageId(message.getMessageId());
        st.setSticker(inp);
        try {
            execute(st);
        } catch (TelegramApiException e) {
            e.printStackTrace();
        }
    }
//метод отсылающий сообщение
    public void sendMsg(Message message, String text) {
        SendMessage sendMessage = new SendMessage();
        Buttons(sendMessage);
        sendMessage.enableMarkdown(true);
        sendMessage.setChatId(message.getChatId().toString());
        sendMessage.setReplyToMessageId(message.getMessageId());
        sendMessage.setText(text);
        try {
            execute(sendMessage);
        } catch (TelegramApiException e) {
            e.printStackTrace();
        }
    }
    public void sendMsgWithoutButtons (Message message, String text) {
        SendMessage sendMessage = new SendMessage();
        sendMessage.enableMarkdown(true);
        sendMessage.setChatId(message.getChatId().toString());
        sendMessage.setReplyToMessageId(message.getMessageId());
        sendMessage.setText(text);
        try {
```

```
        execute(sendMessage);
    } catch (TelegramApiException e) {
        e.printStackTrace();
    }
}
//еще метод для сообщений
public void sendMsg1(Message message, String text) {
    SendMessage sendMessage = new SendMessage();
    Buttons1(sendMessage);
    sendMessage.enableMarkdown(true);
    sendMessage.setChatId(message.getChatId().toString());
    sendMessage.setReplyToMessageId(message.getMessageId());
    sendMessage.setText(text);
    try {
        execute(sendMessage);
    } catch (TelegramApiException e) {
        e.printStackTrace();
    }
}
//метод для тправки сообщений студента
public void sendMsg2(Message message, String text) {
    SendMessage sendMessage = new SendMessage();
    Buttons2(sendMessage);
    sendMessage.enableMarkdown(true);
    sendMessage.setChatId(message.getChatId().toString());
    sendMessage.setReplyToMessageId(message.getMessageId());
    sendMessage.setText(text);
    try {
        execute(sendMessage);
```

```
        } catch (TelegramApiException e) {
            e.printStackTrace();
        }
    }
    //основной метод обрабатывающий обновления
    @Override
    public void onUpdateReceived(Update update) {
        Message message = update.getMessage();
        if (message != null && message.hasText()) {
            switch (message.getText()) {
                case "/start":
                    sendMsgWithoutButtons(message, "Используйте команды /Teacher или /Student для взаимодействия с программой");
                    break;
                case "/Teacher":
                    sendsticker(message, "CAACAgEAAxkBAAECLA9gdUYZc05h0omyTm8Iv_k2irEoqwACVTMAAtpxZgdUSKRTBteYgR4E");
                        sendMsg(message, "Примените кнопки для составления графика обучения");
                    break;
                case "Группа":
                    sendMsgWithoutButtons(message, "Выберите  группу из списка");
                    sendMsgWithoutButtons(message, obj.bd());
                    break;
                case "Предмет":
                    sendMsgWithoutButtons(message, "Выберите предмет из списка");
                    sendMsgWithoutButtons(message, obj.bd1());
                    break;
                case "Тема":
                    sendMsgWithoutButtons(message, "Выберите тему из списка");
```

```
            sendMsgWithoutButtons(message, obj.bd2());
            break;
            case "Дата выдачи":
            sendMsg(message, "Введите дату выдачи задания"+"\n"+"Формат: Дата выдачи - YYYY-
MM-DD");
            break;
          case "Дата сдачи":
            sendMsg(message, "Введите окончательный срок сдачи задания"+"\n"+"Формат:Сдать
до - YYYY-MM-DD");
            break;
          case "Добавить данные":
            if(TeacherDataForDb[0]==null){
            sendMsg(message, "Необходимо выбрать группу!");}
            else if(TeacherDataForDb[1]==null){
               sendMsg(message, "Необходимо выбрать предмет!");
            } else if(TeacherDataForDb[2]==null){
               sendMsg(message, "Необходимо выбрать тему!");
            }
            else if(TeacherDataForDb[3]==null){
               sendMsg(message, "Необходимо выбрать дату выдачи!");
            }
            else if(TeacherDataForDb[4]==null){
               sendMsg(message, "Необходимо выбрать срок сдачи!");
            } else  {
            sendMsg1(message, "Вы собираетесь добавить следующие данные:  "
                  +"\n"+TeacherDataForDb[0]+"\n"+TeacherDataForDb[1]+"\n"+TeacherDataForDb[2]+
                  "\n"+TeacherDataForDb[3]+"\n"+TeacherDataForDb[4] );}
            break;
          case "Подтвердить":
```

```
                sendMsgWithoutButtons(message, "Введите пароль доступа к базе данных");
                break;
            case "sss":
              obj.bd3("INSERT INTO graphik_for_students(gr_id, grp, subject, tops, dat, datas) "
                    +" VALUES(NULL,
'"+TeacherDataForDb[0]+"','"+TeacherDataForDb[1]+"','"+TeacherDataForDb[2]+ "','"
                    +TeacherDataForDb[3].substring(13) +"','"+
                    TeacherDataForDb[4].substring(10)+"');");
                sendMsgWithoutButtons(message, "Данные добавлены");
                sendMsg(message, "Примените кнопки для составления графика обучения");
                for(int i=0;i<5;i++){
                   TeacherDataForDb[i]=null;
                }
                break;
            case "/Student":
              sendMsg2(message, "Добрый день!");
                sendsticker(message, "CAACAgIAAxkBAAECLA5gdUYZUga6nw6TJTw_uYy1rhgyBAACZwADlp-
MDskGZ8xB2w1PHgQ");
              break;
            case "Получить задание":
                sendMsg2(message, "Отправьте данные в формате: "+"\n"+"Группа//" +
                     "ФИО//номер в списке//предмет//тема");
                break;
        }}
      if (message != null && message.hasText()) {
          String []  StudentDataForDb=new String[5];
          Date dv=null;
          Date df=null;
          StringTokenizer strt=new StringTokenizer(message.getText(), "//");
```

```
        int i=0;
        while(strt.hasMoreTokens()){
           StudentDataForDb[i]=strt.nextToken();
           i++;
        }
        if(StudentDataForDb.length==5&&StudentDataForDb[1]!=null){

           for(ArrayList<String> arl: GraphikForStudentromDb){

if(arl.contains(StudentDataForDb[0])&&arl.contains(StudentDataForDb[3])&&arl.contains(StudentDataF
orDb[4])){
                String datav=arl.get(4);
                String dataf=arl.get(5);
                SimpleDateFormat formatter=new SimpleDateFormat("yyyy-MM-dd", Locale.ENGLISH);
                try{
                   dv=formatter.parse(datav);
                   df=formatter.parse(dataf);
                }
                catch (ParseException e){
                   e.printStackTrace();
                }
                break;
             }
           }

if(dv!=null&&df!=null&&obj.bd5().containsKey(Integer.parseInt(StudentDataForDb[2]))&&obj.bd5().get(
Integer.parseInt(StudentDataForDb[2])).equals(StudentDataForDb[1])){
             if(dv.after(new Date())){
                sendMsgWithoutButtons(message,"Это задание еще не доступно! Вы сможете " +
                     "его получить "+dv.toString());
```

```
            }
            if(df.before(new Date())){
               sendMsgWithoutButtons(message,"К сожалению срок сдачи этого задания истек! Обратитесь к " +
                     "преподавателю для получения необходимой информации ");
            }
            if(!dv.after(new Date())&&!df.before(new Date())){
               sendMsgWithoutButtons(message, "Ваше задание "+StudentDataForDb[1]+": ");
               if(obj.bd6(StudentDataForDb[3],StudentDataForDb[4]).get(0).startsWith("https")){
                  senddoc(message,obj.bd6(StudentDataForDb[3],StudentDataForDb[4]).get(0) );
               }
               else  {sendMsgWithoutButtons(message, obj.bd6(StudentDataForDb[3],StudentDataForDb[4]).get(5));}}
         }
         else {sendMsgWithoutButtons(message, "Ваши данные не были обнаружены в базе данных."+"\n"+
               "Это может быть связано с изменениями состава Вашей группы."+"\n"+
               "Обратитесь к преподавателю для получения необходимой информации");}
      }}
   if (message != null && message.hasText() && obj.groupsfromdb.contains(message.getText())) {
      TeacherDataForDb[0]=message.getText();
      sendMsgWithoutButtons(message, "Группа выбрана!");
   }
   if (message != null && message.hasText() && obj.subjectsfromdb.contains(message.getText())) {
      TeacherDataForDb[1]=message.getText();
      sendMsgWithoutButtons(message, "Предмет выбран!");
   }
   if (message != null && message.hasText() && obj.topicsfromdb.contains(message.getText())) {
      TeacherDataForDb[2]=message.getText();
      sendMsgWithoutButtons(message, "Тема выбрана!");
```

```
            }
            if (message != null && message.getText().startsWith("Дата выдачи - 20")) {
                TeacherDataForDb[3]=(message.getText());
                sendMsgWithoutButtons(message, "Дата выдачи установлена!");
            }
            if (message != null && message.getText().startsWith("Сдать до - 20")) {
                TeacherDataForDb[4]=(message.getText());
                sendMsgWithoutButtons(message, "Дата сдачи установлена!");
            }




    }
//метод создающий клавиатуру
    public synchronized void Buttons(SendMessage sendMessage){
        ReplyKeyboardMarkup replyKeyBoardMarkup = new ReplyKeyboardMarkup();
        sendMessage.setReplyMarkup(replyKeyBoardMarkup);
        replyKeyBoardMarkup.setSelective(true);
        replyKeyBoardMarkup.setResizeKeyboard(true);
        replyKeyBoardMarkup.setOneTimeKeyboard(false);
        List<KeyboardRow> keyboard = new ArrayList<>();
        KeyboardRow keyboardfirstrow = new KeyboardRow();
        keyboardfirstrow.add(new KeyboardButton("Группа"));
        keyboardfirstrow.add(new KeyboardButton("Предмет"));
        keyboardfirstrow.add(new KeyboardButton("Тема"));
        KeyboardRow keyboardsecondRow = new KeyboardRow();
        keyboardsecondRow.add(new KeyboardButton("Дата выдачи"));
        keyboardsecondRow.add(new KeyboardButton("Дата сдачи"));
        keyboardsecondRow.add(new KeyboardButton("Добавить данные"));
```

```
    keyboard.add(keyboardfirstrow);
    keyboard.add(keyboardsecondRow);
    replyKeyBoardMarkup.setKeyboard(keyboard);
}
//еще метод для клавиатуры
public synchronized void Buttons1(SendMessage sendMessage){
    ReplyKeyboardMarkup replyKeyBoardMarkup = new ReplyKeyboardMarkup();
    sendMessage.setReplyMarkup(replyKeyBoardMarkup);
    replyKeyBoardMarkup.setSelective(true);
    replyKeyBoardMarkup.setResizeKeyboard(true);
    replyKeyBoardMarkup.setOneTimeKeyboard(false);
    List<KeyboardRow> keyboard = new ArrayList<>();
    KeyboardRow keyboardfirstrow = new KeyboardRow();
    keyboardfirstrow.add(new KeyboardButton("Подтвердить"));
    keyboard.add(keyboardfirstrow);
    replyKeyBoardMarkup.setKeyboard(keyboard);
}
//клавиатура для обработки сообщений от студента
public synchronized void Buttons2(SendMessage sendMessage){
    ReplyKeyboardMarkup replyKeyBoardMarkup = new ReplyKeyboardMarkup();
    sendMessage.setReplyMarkup(replyKeyBoardMarkup);
    replyKeyBoardMarkup.setSelective(true);
    replyKeyBoardMarkup.setResizeKeyboard(true);
    replyKeyBoardMarkup.setOneTimeKeyboard(false);
    List<KeyboardRow> keyboard = new ArrayList<>();
    KeyboardRow keyboardfirstrow = new KeyboardRow();
    keyboardfirstrow.add(new KeyboardButton("Получить задание"));
    keyboard.add(keyboardfirstrow);
    replyKeyBoardMarkup.setKeyboard(keyboard);
```

```
    }
//метод возврвщающий имя пользователя нашего бота
    @Override
    public String getBotUsername() {
        return "TeachassBot";
    }


    public static void main(String [] args) {
        ClassForGr o1=new ClassForGr();
        Bot b = new Bot();
       /* System.out.println(b.GraphikForStudentromDb);
        System.out.println(o1.bd6("Дискретная математика", "Теория множеств"));
System.out.println(o1.bd5().toString());*/
        try {
            TelegramBotsApi telgramBotspi = new TelegramBotsApi(DefaultBotSession.class);
            telgramBotspi.registerBot(new Bot());
        } catch (TelegramApiException e) {
            e.printStackTrace();
        }
    }
}
```