```
using Newtonsoft.Json;
using System.Text;
namespace PolytehChatBot
{
    internal class User
    {
        public string Id { get; set; }
        public string UserName { get; set; }
        public string First_Name { get; set; }
        public int currBotMessageId { get; set; }
        public string? QuestionId {  get; set; }
        public string? BufId { get; set; }
        public int State { get; set; }
        public string? group { get; set; }
        public bool isAdmin { get; set; }
        public User(string id, string userName, int
currMessageId,string firstn)
        {
            Id = id;
            UserName = userName;
            this.currBotMessageId =
currMessageId;
            this.State = -1;
            this.group = null;
            this.QuestionId = null;
            this.BufId = null;
            this.isAdmin = false;
            this.First_Name = firstn;
        }
    }
    internal class Question
    {
        public static int Count;
        public string Id { get; set; }
        public string MessageId { get; set; }

        public User FromUser { get; set; }

        public bool isAnswered { get; set; }

        public DateTime Date {  get; set; }

        public Question(string messageId, User
fromUser)
        {
            Count += 1;
            Id = Count.ToString();

            MessageId = messageId;
            FromUser = fromUser;
            this.isAnswered = false;
            this.Date = DateTime.Now;
        }
    }
    internal class Bot
    {
        public static string usersPath =
"assets/users.json";
        public static string questionsPath =
"assets/questions.json";
        public string TOKEN {get; set;}
        public string Id { get; set; }
        public string First_Name { get; set; }
```

```
        public string? Username  { get; set; }
        public static List<User> admins {  get; set;
}
        public static List<User> users { get; set; }


        public static List<Question> questions {
get; set; }


        public Bot(string token, string id, string
first_name, string username)
        {
            this.TOKEN = token;
            this.Id= id;
            this.First_Name = first_name;
            this.Username = username;
            users = new List<User>();
        }
        public void SerializeDataToFile()
        {
            string json1 =
JsonConvert.SerializeObject(users,
Formatting.Indented);
            File.WriteAllText(usersPath, json1);
            string json2 =
JsonConvert.SerializeObject(questions,
Formatting.Indented);
            File.WriteAllText(questionsPath,
json2);
        }
        public void DeserializeDataFromFile()
        {
            if (File.Exists(usersPath))
            {
                string json =
File.ReadAllText(usersPath);
                users =
JsonConvert.DeserializeObject<List<User>>(js
on);


                admins = new List<User>();
                foreach (var user in users)
                    if (user.isAdmin)
admins.Add(user);
            }  else
            {
                users = new List<User>();
                string json =
JsonConvert.SerializeObject(users,
Formatting.Indented);
                File.WriteAllText(usersPath, json);
                Console.WriteLine("Файл users не
найден. Создан новый");
            }
            Question.Count = 0;
            if (File.Exists(questionsPath))
            {
                string json =
File.ReadAllText(questionsPath);
                questions =
JsonConvert.DeserializeObject<List<Question
>>(json);
                if (questions.Count!=0)
                    Question.Count =
Convert.ToInt32(questions[^1].Id);
            }
            else
            {
                questions = new List<Question>();
```

```
            string json =
JsonConvert.SerializeObject(questions,
Formatting.Indented);
            File.WriteAllText(questionsPath,
json);
            Console.WriteLine("Файл questions
не найден. Создан новый");
        }
    }
    public static string? GetFileText(string
path)
    {
        try { return File.ReadAllText(path); }
        catch (Exception ex)
        {
            Console.WriteLine($"Ошибка в
чтении файла: {ex.Message}");
            return null;
        }
    }
    public async Task StartAsync()
    {
        this.DeserializeDataFromFile();
        Console.WriteLine($"Юзеров в
системе: {users.Count}");
        Console.WriteLine($"БОТ запущен");

        Scheduler scheduler = new
Scheduler(3000);
        scheduler.Run();
        int update_id = 0;
        while (true)
        {
            List<Update> updates =
TgApi.GetUpdates(update_id);
            if (updates?.Count != 0)
            {
                var update = updates[^1];
                update_id = update.Update_Id + 1;
                User foundUser = users.Find(user
=> user.Id ==
                (update.Message?.Chat?.Id ??
update.Callback_Query.Message.Chat.Id));
                if (foundUser is null){
                    users.Add(new User(
                        (update.Message?.Chat?.Id ??
update.Callback_Query.Message.Chat.Id),

(update.Message?.Chat?.Username is null ?
($"undefined{(update.Message?.Chat?.Id ??
update.Callback_Query.Message.Chat.Id)}") :

(update.Message?.Chat?.Username ??
update.Callback_Query?.Message?.Chat?.User
name)),
                        0,
(update.Message?.Chat?.First_Name ??
update.Callback_Query.Message.Chat.First_Na
me)));
                    foundUser = users[^1];}
                if
((update.Message?.Chat?.Username??update.C
allback_Query?.Message?.Chat?.Username) is
not null
                    &&
update.Message?.Chat?.Username!=foundUser.
UserName)
                    foundUser.UserName =
update.Message?.Chat?.Username??
update.Callback_Query.Message.Chat.Userna
me;                 if
```

```
((update.Message?.Chat?.First_Name ?? update.Callback_Query.Message.Chat.First_Name) != foundUser.First_Name)
                foundUser.First_Name = update.Message?.Chat?.First_Name ?? update.Callback_Query.Message.Chat.First_Name;


            foundUser.currBotMessageId = update.Message?.Message_Id ?? update.Callback_Query.Message.Message_Id;


            Console.WriteLine($"+update [id{update_id}, type: {(update.Message!=null?"msg":"cbq")}," +
                $" data: {((update.Message != null ? $"\"{update.Message.Text}\"": update.Callback_Query.Data))}]," +
                $" [user: {foundUser.UserName}," +
                $" name: {foundUser.First_Name}, id{foundUser.Id}]");


            //админ рассылка + вопросы
            if (StateHandler(update.Callback_Query, update.Message, foundUser))
            {
                this.SerializeDataToFile();
                continue;
            }


            if (update.Message != null)

MessageHandler(update.Message,foundUser.currBotMessageId, foundUser);
            else if (update.Callback_Query != null)

CallbackQueryHandler(update.Callback_Query, foundUser.currBotMessageId, foundUser);


                this.SerializeDataToFile();
            }
        }
    }


    public static bool StateHandler(CallbackQuery cbq, Message msg, User user)
    {
        if (msg == null)
        {
            user.State = 0;
            if (cbq != null)
                CallbackQueryHandler(cbq, user.currBotMessageId, user);
            return true;
        }
        if (user.State != 0)
            TgApi.DeleteMessage(msg.Chat.Id, (msg.Message_Id - 1).ToString());
        if (user.State == 1 && user.isAdmin) //рассылка
        {
            foreach (var us in users)
            {
                TgApi.SendMessage(us.Id, "🔔 Сообщение от администратора:");
```

```
            TgApi.CopyMessage(us.Id,
msg.Chat.Id, msg.Message_Id.ToString());
        }
        InlineKeyboard ik = new
InlineKeyboard();
        ik.AddButton("🛠 На главную",
"main1");
        TgApi.SendMessage(msg.Chat.Id,
          $"Сообщение отослано
{users.Count} пользователям чат-бота",
ik.GetMarkup());
        user.State = 0;
        return true;
      }
      else if (user.State == 2) //Вопрос
админу
      {
        user.QuestionId =
msg.Message_Id.ToString();
        questions.Add(new
Question(msg.Message_Id.ToString(), user));
        InlineKeyboard ik = new
InlineKeyboard();
        ik.AddButton("🛠 На главную",
"main1");
        TgApi.SendMessage(msg.Chat.Id,
          $"Ваше обращение записано.
Ожидайте ответа.", ik.GetMarkup());
        foreach (var admin in admins)
        {
          TgApi.SendMessage(admin.Id,
$"!! Поступило сообщение от пользователя
[{user.First_Name}, {user.UserName}]");
        }
        user.State = 0;
        return true;
      }
      else if (user.State == 3 &&
user.isAdmin) //выбор вопроса админом
      {
        string questID = msg.Text;
        string SearchQuestion(string qId)
        {
          foreach (var q in questions)
            if (q.Id == questID)
            {
              TgApi.CopyMessage(user.Id,
q.FromUser.Id, q.MessageId);
              return q.Id;
            }
          return null;
        }
        InlineKeyboard ik = new
InlineKeyboard();
        ik.AddButton("🛠 Назад к
вопросам", "questions0");
        if (SearchQuestion(questID) == null)
        {
          user.State = 0;
          TgApi.SendMessage(msg.Chat.Id,
          $"❌ Вопрос не найден:",
ik.GetMarkup());
          return true;
        }
        user.BufId = questID;
        TgApi.SendMessage(msg.Chat.Id,
          $"✏ Напишите ваш ответ:",
ik.GetMarkup());
```

```
            user.State = 4;
            return true;
        }
        else if (user.State == 4 && user.isAdmin) // ответ на вопрос админом
        {
            User ToUser = null;
            foreach (var q in questions)
                if (q.Id == user.BufId)
                {
                    ToUser = q.FromUser;
                    q.isAnswered = true;
                    q.FromUser.QuestionId = null;
                }
            if (ToUser is not null)
            {
                TgApi.SendMessage(ToUser.Id, "!! Ответ от администратора:");
TgApi.CopyMessage(ToUser.Id,user.Id,msg.Message_Id.ToString());
            }
            InlineKeyboard ik = new InlineKeyboard();
            ik.AddButton("🛠 На главную", "main1");
            TgApi.SendMessage(msg.Chat.Id,
                $"✏ Ответ отослан пользователю [{ToUser.First_Name}, {ToUser.UserName}]", ik.GetMarkup());
            user.State = 0;
            return true;
        }
        return false;
    }
    public static void CallbackQueryHandler(CallbackQuery cbq,int message_id,User user)
    {
        var chat_id = cbq.ChatId;
        var callbackData = cbq.Data;
        user.currBotMessageId += 1;

        var ik = new InlineKeyboard();
        if (callbackData.StartsWith("sub"))
        {
            ik.AddButton("⮌ Назад", "other1");

            string sub_gr = callbackData.Split("sub_")[1].ToUpper();
            user.group = sub_gr;
            TgApi.DeleteMessage(chat_id, message_id.ToString());
            TgApi.SendMessage(chat_id,$"Вы подписались на обновления расписания групп {sub_gr}",
                ik.GetMarkup(), isHTML: false);
            return;
        }
        switch (callbackData)
        {
            case "unsub":
                ik.AddButton("⮌ Назад", "other1");
                user.group = null;
                TgApi.DeleteMessage(chat_id, message_id.ToString());
```

```
                TgApi.SendMessage(chat_id,
$"Вы отписались от обновлений
расписания",
                    ik.GetMarkup(), isHTML:
false);break;
            case "prof1":
                ik.AddButton("🏗️ ПГБ", "pgb1");
                ik.AddButton("🚧 БДА", "bda1");
                ik.NextRow();
                ik.AddButton("🎨 АГБ", "agb1");
                ik.AddButton("⚙️ ТАР", "tor1");
                ik.NextRow();
                ik.AddButton("✒️ АЭП", "aep1");
                ik.AddButton("👨‍💻 ПЗТ",
"pzt1");
                ik.NextRow();
                ik.AddButton("↩️ Назад",
"main1");
                TgApi.DeleteMessage(chat_id,
message_id.ToString());
                TgApi.SendMessage(chat_id,
GetFileText("assets/proffesions.txt"),
                    ik.GetMarkup(), isHTML: true);
                break;
            case "pgb1":
                ik.AddButton("↩️ Назад",
"prof1");
                TgApi.DeleteMessage(chat_id,
message_id.ToString());
                TgApi.SendMessage(chat_id,
                    GetFileText("assets/PGB.txt"),
                    ik.GetMarkup(),
                    isHTML: true);
                break;
            case "bda1":
                ik.AddButton("↩️ Назад",
"prof1");
                TgApi.DeleteMessage(chat_id,
message_id.ToString());
                TgApi.SendMessage(chat_id,
                    GetFileText("assets/BDA.txt"),
                    ik.GetMarkup(),
                    isHTML: true);
                break;
            case "agb1":
                ik.AddButton("↩️ Назад",
"prof1");
                TgApi.DeleteMessage(chat_id,
message_id.ToString());
                TgApi.SendMessage(chat_id,
                    GetFileText("assets/AGB.txt"),
                    ik.GetMarkup(),
                    isHTML: true);
                break;
            case "tor1":
                ik.AddButton("↩️ Назад",
"prof1");
                TgApi.DeleteMessage(chat_id,
message_id.ToString());
                TgApi.SendMessage(chat_id,
                    GetFileText("assets/TOR.txt"),
                    ik.GetMarkup(),
                    isHTML: true);
                break;
            case "aep1":
```

```
            ik.AddButton("⮌ Назад",
"prof1");
            TgApi.DeleteMessage(chat_id,
message_id.ToString());
            TgApi.SendMessage(chat_id,
                GetFileText("assets/AEP.txt"),
                ik.GetMarkup(),
                isHTML: true);
            break;
        case "pzt1":
            ik.AddButton("⮌ Назад",
"prof1");
            TgApi.DeleteMessage(chat_id,
message_id.ToString());
            TgApi.SendMessage(chat_id,
                GetFileText("assets/PZT.txt"),
                ik.GetMarkup(),
                isHTML: true);
            break;
        case "dorm_rules1":
            ik.AddButton("⮌ Назад",
"dorm0");
            TgApi.DeleteMessage(chat_id,
message_id.ToString());

TgApi.SendDocumentAsync(chat_id,
                filepath:
"assets/dorm_rules.doc",
                caption: "",
                markup: ik.GetMarkup(),
                filename: "Правила
внутреннего распорядка в общежитии.doc");
            break;
        case "dorm0":
            ik.AddButton("Общежитие №1",
"dorm1");
            ik.AddButton("Общежитие №2",
"dorm2");
            ik.NextRow();
            ik.AddButton("📄 Правила
внутреннего распорядка", "dorm_rules1");
            ik.NextRow();
            ik.AddButton("⮌ Назад",
"main1");
            TgApi.DeleteMessage(chat_id,
message_id.ToString());
            TgApi.SendMessage(chat_id,
                GetFileText("assets/dorms.txt"),
                ik.GetMarkup(),
                isHTML: true);
            break;
        case "dorm1":
            ik.AddButton("⮌ Назад",
"dorm0");


            TgApi.DeleteMessage(chat_id,
message_id.ToString());
            TgApi.SendPhotoAsync(chat_id,
                filepath: "assets/dorm1.jpg",
                caption:
GetFileText("assets/dorm1.txt"),
                markup: ik.GetMarkup(),
                isHTML: true);
            break;
        case "dorm2":
            ik.AddButton("⮌ Назад",
"dorm0");
```

```
            TgApi.DeleteMessage(chat_id,
message_id.ToString());
            TgApi.SendPhotoAsync(chat_id,
                filepath: "assets/dorm2.jpg",
                caption:
GetFileText("assets/dorm2.txt"),
                markup: ik.GetMarkup(),
                isHTML: true);
            break;
        case "priem1":
            ik.AddButton("🙏 Проходные
баллы", "scores1");
            ik.NextRow();
            ik.AddButton("⏳ Сроки
вступительной кампании", "camp1");
            ik.NextRow();
            ik.AddButton("📋 План приёма",
"plan1");
            ik.NextRow();
            ik.AddButton("🖨️ Необходимые
документы", "reqdoc1");
            ik.NextRow();
            ik.AddButton("📝 Списки
зачисленных", "spiski1");
            ik.NextRow();
            ik.AddButton("↩️ Назад",
"main1");
            TgApi.DeleteMessage(chat_id,
message_id.ToString());
            TgApi.SendMessage(chat_id, "🖨️
Приёмная кампания",
                ik.GetMarkup(), isHTML: true);
            break;
        case "scores1":
            ik.AddButton("↩️ Назад",
"priem1");
            TgApi.DeleteMessage(chat_id,
message_id.ToString());
            TgApi.SendPhotoAsync(chat_id,
                filepath:
"assets/pass_scores.png",
                caption: "Проходные баллы",
                markup: ik.GetMarkup());
            break;
        case "camp1":
            ik.AddButton("↩️ Назад",
"priem1");
            TgApi.DeleteMessage(chat_id,
message_id.ToString());
            TgApi.SendPhotoAsync(chat_id,
                filepath:
"assets/terms_of_entry.png",
                caption: "Сроки
вступительной кампании",
                markup: ik.GetMarkup());
            break;
        case "plan1":
            ik.AddButton("↩️ Назад",
"priem1");
            TgApi.DeleteMessage(chat_id,
message_id.ToString());

TgApi.SendDocumentAsync(chat_id,
                filepath:
"assets/plan_priema.pdf",
                caption: "План приёма",
                markup: ik.GetMarkup(),
                filename:"plan_priema.pdf");
```

```
                break;
            case "reqdoc1":
                ik.AddButton("Детям, оставшимся без попечения родителей", "sirotam1");
                ik.NextRow();
                ik.AddButton("⮌ Назад", "priem1");
                TgApi.DeleteMessage(chat_id, message_id.ToString());
                TgApi.SendMessage(chat_id,

GetFileText("assets/required_documents.txt"),
                    ik.GetMarkup(),
                    isHTML: true);
                break;
            case "sirotam1":
                ik.AddButton("⮌ Назад", "priem1");
                TgApi.DeleteMessage(chat_id, message_id.ToString());
                TgApi.SendMessage(chat_id,

GetFileText("assets/required_documents_sirotam.txt"),
                    ik.GetMarkup(),
                    isHTML: true);
                break;
            case "spiski1":
                ik.AddButton("⮌ Назад", "priem1");
                TgApi.DeleteMessage(chat_id, message_id.ToString());
                TgApi.SendMessage(chat_id,
                    GetFileText("assets/spiski.txt"),
                    ik.GetMarkup(),
                    isHTML: true);
                break;
            case "contacts1":
                ik.AddButton("⮌ Назад", "main1");
                TgApi.DeleteMessage(chat_id, message_id.ToString());

//TgApi.SendLocationAsync(chat_id,
                //    latitude: "53.667466",
                //    longitude: "23.827687");
                TgApi.SendPhotoAsync(chat_id,
                    filepath: "assets/map.jpg",
                    caption: GetFileText("assets/contacts.txt"),
                    markup: ik.GetMarkup(),
                    isHTML: true);
                break;
            case "main1":
                TgApi.DeleteMessage(chat_id, message_id.ToString());
                MessageHandler(new Message { Text = "///main", Chat = new Chat { Id = chat_id } }, message_id, user);
                break;
            case "other1":
                ik.AddButton("⏰ Рассылка обновления расписания", "scheduler1");
                ik.NextRow();
                ik.AddButton("👀 Прохождение профи-теста", "test1");
                ik.NextRow();
```

```
                ik.AddButton("❓ Задать вопрос администратору", "question1");
                ik.NextRow();
                ik.AddButton("↩️ Назад", "main1");
                TgApi.DeleteMessage(chat_id, message_id.ToString());
                TgApi.SendMessage(chat_id, "Прочее: ",
                    ik.GetMarkup(), isHTML: false);
                break;
            case "test1":
TgApi.DeleteMessage(chat_id,message_id.ToString());
                ik.AddButton("↩️ Назад", "main1");
                TgApi.SendMessage(chat_id, $"<a href=\"https://profitest.ripo.by/public/main\"" +
                    $">Перейти на сайт для прохождения профи теста</a>", ik.GetMarkup(), isHTML: true);
                break;
            case "scheduler1":
                TgApi.DeleteMessage(chat_id, message_id.ToString());
                if (user.group is null)
                {
                    ik.AddButton("🏗️ ПГБ", "sub_pgb");
                    ik.AddButton("🚧 БДА", "sub_bda");
                    ik.NextRow();
                    ik.AddButton("🎨 АГБ", "sub_agb");
                    ik.AddButton("⚙️ ТАР", "sub_tor");
                    ik.NextRow();
                    ik.AddButton("🔌 АЭП", "sub_aep");
                    ik.AddButton("👨‍💻 ПЗТ", "sub_pzt");
                    ik.NextRow();
                    ik.AddButton("💳 БСК", "sub_bsk");
                    ik.AddButton("🌱 1 Курс", "sub_p_kurs");
                    ik.NextRow();
                    ik.AddButton("↩️ Назад", "main1");
                    TgApi.SendMessage(chat_id, "Выберите группы для оповещения: ",
                        ik.GetMarkup(), isHTML: false);
                }
                else
                {
                    ik.AddButton("🚫 Отписаться", "unsub");
                    ik.NextRow();
                    ik.AddButton("↩️ Назад", "main1");
                    TgApi.SendMessage(chat_id, $"Вы подписаны на расписание групп {user.group}",
                        ik.GetMarkup(), isHTML: false);
                }
                break;
            case "faq1":
```

```
                ik.AddButton("🔙 Назад",
"main1");
                TgApi.DeleteMessage(chat_id,
message_id.ToString());
                TgApi.SendMessage(chat_id,
                   GetFileText("assets/faq.txt"),
                   ik.GetMarkup(),
                   isHTML: true);
                break;
             case "adminpan1":
                ik.AddButton("📓 Вопросы от
пользователей", "questions0");
                ik.NextRow();
                ik.AddButton("⚡ Админ-
рассылка", "admin0");
                ik.NextRow();
                ik.AddButton("🔙 Назад",
"main1");
                if (user.isAdmin)
                {
                   TgApi.DeleteMessage(chat_id,
message_id.ToString());
                   TgApi.SendMessage(chat_id,
$"-- 🖍 Админ панель 🖍 --",
ik.GetMarkup(), isHTML: false);
                }
                break;
             case "question1":
                TgApi.DeleteMessage(chat_id,
message_id.ToString());
                if (user.QuestionId != null)
                {
                   ik.AddButton("🔙 Назад",
"main1");
                   ik.NextRow();
                   ik.AddButton("🔁 Отозвать
вопрос", "cancelquestion1");
                   TgApi.SendMessage(chat_id,
$"🖍 Вы уже обращались с вопросом к
администратору, " +
                      $"отмените его либо
дождитесь ответа.", ik.GetMarkup(),
isHTML: false);
                   break;
                }
                user.State = 2;
                ik.AddButton("❌ Отмена",
"main1");
                TgApi.SendMessage(chat_id,
$"🖍 Напишите ваш вопрос
администратору:", ik.GetMarkup(), isHTML:
false);
                break;
             case "cancelquestion1":
                ik.AddButton("🔙 Назад",
"main1");
                TgApi.DeleteMessage(chat_id,
message_id.ToString());
                TgApi.SendMessage(chat_id,
$"🖍 Ваш вопрос отозван:", ik.GetMarkup(),
isHTML: false);
                user.QuestionId = null;
                break;
             case "questions0":
                if (user.isAdmin)
                {
                   user.State = 3;
                   string text = "Неотвеченные
вопросы: \n";
```

```
                foreach (var question in
questions){
  if (!question.isAnswered)
                    text += $"ID:
{question.Id}, от
[{question.FromUser.First_Name},{question.F
romUser.UserName}], " +
                    $"msg_id:
{question.MessageId}, время: " +

$"[{question.Date.Hour}.{question.Date.Minut
e} " +

$"{question.Date.Day}.{question.Date.Month}
.{question.Date.Year}]\n";
                }
                TgApi.DeleteMessage(chat_id,
message_id.ToString());
                TgApi.SendMessage(chat_id,
text, isHTML: false);
                ik.AddButton("❌ Отмена",
"main1");
                TgApi.SendMessage(chat_id,
$"✏️ Введите ID вопроса: ", ik.GetMarkup(),
isHTML: false);
            }
            break;
        case "admin0":
            user.State= 1;
            ik.AddButton("❌ Отмена",
"main1");
            TgApi.DeleteMessage(chat_id,
message_id.ToString());
            TgApi.SendMessage(chat_id,
$"✏️ Введите сообщение для
рассылки",ik.GetMarkup(), isHTML: false);
            break;
        case "delete1":
            TgApi.DeleteMessage(chat_id,
message_id.ToString());
            break;
        default:
          TgApi.AnswerCallbackQuery(cbq.Id,
"Пока не готово.... :(");
            break;
    }
  }
  public static void
MessageHandler(Message message, int
message_id, User user)
  {
    var chat_id = message.Chat.Id;
    string text = message.Text.ToString();
    user.currBotMessageId += 1;

    var ik = new InlineKeyboard();

    if (text.StartsWith("/deop ") &&
user.isAdmin)
    {
      var us = text.Split("/deop ")[1];
      if (us == "all")
        foreach (var to_user in Bot.users)
          to_user.isAdmin = false;
      else
      {
        foreach (var to_user in Bot.users)
          if (to_user.UserName == us)
          {
```

```
                to_user.isAdmin = false;
                TgApi.SendMessage(chat_id, $"❌ Админские полномочия у пользователя {to_user.First_Name} убраны", markup: ik.GetMarkup(), isHTML: true);}}return;}
            switch (text){
                case "/start":
                    ik.AddButton("🐐 Начать", "main1");
                    TgApi.SendMessage(chat_id, $"👾 Добро пожаловать на борт, {user.First_Name}!" +
                    " Вы обратились к боту-помощнику Гродненского государственного политехнического колледжа!" +
                    " Нажмите \"Начать\" чтобы начать! ", ik.GetMarkup());
                    break;
                case "///main":
                    if (user.isAdmin == true){
                    ik.AddButton("-- 🖍 Админ-панель 🖍 --", "adminpan1");
                    ik.NextRow();  }
                    ik.AddButton("📜 Профессии и специальности", "prof1");
                    ik.NextRow();
                    ik.AddButton("🎁 Наши общежития", "dorm0");
                    ik.NextRow();
                    ik.AddButton("🖨 Приёмная кампания", "priem1");
                    ik.NextRow();
                    ik.AddButton("📞 Контакты и координаты колледжа", "contacts1");
                    ik.NextRow();
                    ik.AddButton("❓ Часто задаваемые вопросы", "faq1");
                    ik.NextRow();
                    ik.AddButton("🔧 Прочее", "other1");
                    //ik.AddButton("🐐 Назад", "main0");
                    TgApi.SendMessage(chat_id,
                    "🛠 Гродненский государственный политехнический колледж\n" +
                    "Выберите интересующий вас вопрос: ", ik.GetMarkup());
                    break;
                case "/op":
                    user.isAdmin = true;
                    ik.AddButton("🐐 Да!", "delete1");
                    TgApi.SendMessage(chat_id, GetFileText("assets/admin.txt"), markup: ik.GetMarkup(), isHTML: true);
                    break;
                default:
                    TgApi.SendMessage(chat_id, "~");
                    break;   }   }   }
    public class Group{
        public string Name { get; set; }
        public long PrevSize { get; set; }
        public string Url { get; set; }
        public Group(string name, long prev_s, string url)
        {
            this.Name = name;
```

```
        this.PrevSize = prev_s;
        this.Url = url;}}
    internal class Scheduler{
        public List<Group> groups { get; set; }
        public int interval { get; set; }
        public Scheduler(int interval) {
            this.groups = new List<Group>{
                new
Group("P_KURS",0,"http://ggpk.by/Raspisanie
/Files/P_KURS.html"),
                new
Group("PGB",0,"http://ggpk.by/Raspisanie/File
s/PGB.html"),
                new
Group("BDA",0,"http://ggpk.by/Raspisanie/Fil
es/BDA.html"),new
Group("AGB",0,"http://ggpk.by/Raspisanie/Fil
es/AGB.html"),
                new
Group("TAR",0,"http://ggpk.by/Raspisanie/Fil
es/TAR.html"),new
Group("AEP",0,"http://ggpk.by/Raspisanie/File
s/AEP.html"),   new
Group("PZT",0,"http://ggpk.by/Raspisanie/File
s/PZT.html"),
                new
Group("BSK",0,"http://ggpk.by/Raspisanie/File
s/BSB.html"),
      };  this.interval = interval; }
        public async void Run(){
            CancellationTokenSource cts = new
CancellationTokenSource();
            await Task.Run(() =>
checkSchedule(cts.Token, interval));}
        async Task
checkSchedule(CancellationToken token, int
interval)  {
            Console.WriteLine("Проверка
расписания запущена");
            while (!token.IsCancellationRequested)
{
                static async Task<long>
GetHtmlPageSizeAsync(string url) {    try
                {      using (HttpClient client = new
HttpClient())  {
       string html = await
client.GetStringAsync(url);
   return
Encoding.UTF8.GetByteCount(html);}}
                catch (Exception) {
                    Console.WriteLine("Произошла
ошибка в получении данных расписания.
Функция отключена.");
                    return 0;}
            }   foreach (Group gr in this.groups){
                long size = await
GetHtmlPageSizeAsync(gr.Url);
                //Console.WriteLine($"Размер
HTML-страницы {gr.Name}: {size} байт");

//Console.WriteLine($"Расписание проверено
в {DateTime.Now}");  if (size == 0) return;
if (gr.PrevSize==0) gr.PrevSize = size;
  else if (gr.PrevSize != size && size!=0) {
Console.WriteLine($"Расписание групп
{gr.Name} поменялось!");gr.PrevSize =
size;foreach (var user in Bot.users){
   if (user.group==gr.Name)
TgApi.SendMessage(user.Id.ToString(), $"⏰
Расписание групп {gr.Name} обновилось!\n"
+      $"{gr.Url}");}} }
                await Task.Delay(interval, token);}
}}}
```